DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CS3351– DIGITAL PRINCIPLES AND COMPUTER ORGANIZATION

Question Bank

II YEAR CSE

SYLLABUS

CS3351 DIGITAL PRINCIPLES AND COMPUTER ORGANIZATION **L T P C**
3 0 2 4

COURSE OBJECTIVES:

- To analyze and design combinational circuits.
- To analyze and design sequential circuits
- To understand the basic structure and operation of a digital computer.
- To study the design of data path unit, control unit for processor and to familiarize with the hazards.
- To understand the concept of various memories and I/O interfacing.

UNIT I COMBINATIONAL LOGIC **9**

Combinational Circuits – Karnaugh Map - Analysis and Design Procedures – Binary Adder – Subtractor – Decimal Adder - Magnitude Comparator – Decoder – Encoder – Multiplexers - Demultiplexers

UNIT II SYNCHRONOUS SEQUENTIAL LOGIC **9**

Introduction to Sequential Circuits – Flip-Flops – operation and excitation tables, Triggering of FF, Analysis and design of clocked sequential circuits – Design – Moore/Mealy models, state minimization, state assignment, circuit implementation - Registers – Counters.

UNIT III COMPUTER FUNDAMENTALS **9**

Functional Units of a Digital Computer: Von Neumann Architecture – Operation and Operands of Computer Hardware Instruction – Instruction Set Architecture (ISA): Memory Location, Address and Operation – Instruction and Instruction Sequencing – Addressing Modes, Encoding of Machine Instruction – Interaction between Assembly and High Level Language.

UNIT IV PROCESSOR **9**

Instruction Execution – Building a Data Path – Designing a Control Unit – Hardwired Control, Microprogrammed Control – Pipelining – Data Hazard – Control Hazards.

UNIT V MEMORY AND I/O **9**

Memory Concepts and Hierarchy – Memory Management – Cache Memories: Mapping and Replacement Techniques – Virtual Memory – DMA – I/O – Accessing I/O: Parallel and Serial Interface – Interrupt I/O – Interconnection Standards: USB, SATA

45 PERIODS

PRACTICAL EXERCISES: 30 PERIODS

1. Verification of Boolean theorems using logic gates.

2. Design and implementation of combinational circuits using gates for arbitrary functions.

3. Implementation of 4-bit binary adder/subtractor circuits.

4. Implementation of code converters.

5. Implementation of BCD adder, encoder and decoder circuits

6. Implementation of functions using Multiplexers.

7. Implementation of the synchronous counters

8. Implementation of a Universal Shift register.

9. Simulator based study of Computer Architecture

COURSE OUTCOMES:

At the end of this course, the students will be able to:

CO1 : Design various combinational digital circuits using logic gates
CO2 : Design sequential circuits and analyze the design procedures
CO3 : State the fundamentals of computer systems and analyze the execution of an instruction
CO4 : Analyze different types of control design and identify hazards
CO5 : Identify the characteristics of various memory systems and I/O communication

TOTAL: 75 PERIODS

TEXT BOOKS:

1. M. Morris Mano, Michael D. Ciletti, "Digital Design : With an Introduction to the Verilog HDL, VHDL, and System Verilog", Sixth Edition, Pearson Education, 2018.

2. David A. Patterson, John L. Hennessy, "Computer Organization and Design, The Hardware/Software Interface", Sixth Edition, Morgan Kaufmann/Elsevier, 2020.

REFERENCES:

1. Carl Hamacher, Zvonko Vranesic, Safwat Zaky, Naraig Manjikian, "Computer Organization and Embedded Systems", Sixth Edition, Tata McGraw-Hill, 2012.

2. William Stallings, "Computer Organization and Architecture – Designing for Performance", Tenth Edition, Pearson Education, 2016.

3. M. Morris Mano, "Digital Logic and Computer Design", Pearson Education, 2016.

BLOOM TAXANOMY LEVELS

BTL1: Creating., BTL2: Evaluating., BTL3: Analyzing., BTL4: Applying., BTL5: Understanding., BTL6: Remembering

UNIT I COMBINATIONAL LOGIC 9

Combinational Circuits – Karnaugh Map - Analysis and Design Procedures – Binary Adder – Subtractor – Decimal Adder - Magnitude Comparator – Decoder – Encoder – Multiplexers - Demultiplexers

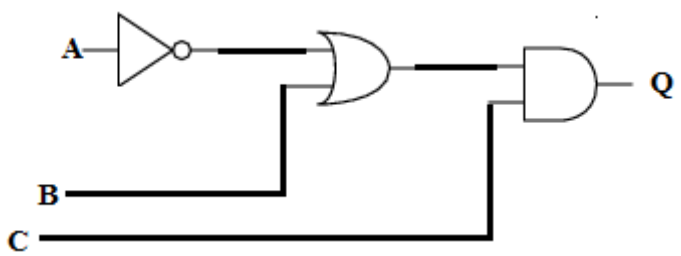
.

PART – A

CO Mapping : CO202.1

S. No.	**Question**	Blooms Taxanomy Level	**Competence**	**PO**
1	**Find the Octal equivalent of the hexadecimal number DC.BA. (May/June 2016)**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3**
2	**What is meant by multilevel gates networks?(May/June 2016)**	**BTL-1**	**Remembering**	**PO1**
3	**Discuss the NOR operation with a truth table. (Nov./Dec. 2015)**	**BTL-1**	**Remembering**	**PO1**
4	**Write short notes on weighted binary codes. (Nov./Dec. 2015)**	**BTL-1**	**Remembering**	**PO1**
5	**Convert $(126)_{10}$ to Octal number and binary number. (Nov./Dec. 2015)**	**BTL-1**	**Remembering**	**PO1**
6	**Prove the following using Demorgan' theorem [(X+Y)'+(X+Y)']'= X+Y (May 2015)**	**BTL-1**	**Remembering**	**PO1**
7	**Convert $(0.6875)_{10}$ to binary. (May 2015)**	**BTL-1**	**Remembering**	**PO1**
8	**Implement AND gate using only NOR gate (December 2014)**	**BTL-1**	**Remembering**	**PO1**
9	**State the principle of duality (December 2014)**	**BTL-1**	**Remembering**	**PO1**
10	**State and prove the consensus theorem. (June 2014)**	**BTL-1**	**Remembering**	**PO1**
11	**Find the octal equivalent of hexadecimal numbers AB.CD. (June 2014)**	**BTL-1**	**Remembering**	**PO1**

12	**Realize XOR gate using only 4 NAND gates. (Dec 2013)**	**BTL-2**	**Understanding**	**PO1, PO2**
13	**Realize JK flip flop using D flip flop. (Dec 2013)**	**BTL-1**	**Remembering**	**PO1**
14	**Convert the following hexadecimal numbers into decimal numbers: (Dec 2012)** **a)263, b)1C3**	**BTL-1**	**Remembering**	**PO1**
15	**What is the significance of BCD code. (Dec 2012)**	**BTL-1**	**Remembering**	**PO1**
16	**Simplify the expression: X = (A'+B)(A+B+D)D'.**	**BTL-1**	**Remembering**	**PO1**
17	**Convert $(11001010)_2$ into gray code.** **b) Convert a Gray code 11101101 into binary code.**	**BTL-1**	**Remembering**	**PO1**
18	**State & prove De-Morgan's theorem.**	**BTL-1**	**Remembering**	**PO1**
19	**Describe the canonical forms of the Boolean function.**	**BTL-1**	**Remembering**	**PO1**
20	**Describe the importance of don't care conditions.**	**BTL-1**	**Remembering**	**PO1**
21	**What is a prime implicant?**	**BTL-1**	**Remembering**	**PO1**
22	**Define the following: minterm and maxterm?**	**BTL-1**	**Remembering**	**PO1**
23	**Minimize the function using K-map: F=∑m(1,2,3,5,6,7).**	**BTL-1**	**Remembering**	**PO1**
24	**Define Karnaugh map.**	**BTL-1**	**Remembering**	**PO1**
25	**Plot the expression on K-map: F (w,x,y) =∑m (0, 1, 3, 5, 6) + d (2, 4).**	**BTL-1**	**Remembering**	**PO1**
26	**Express x + yz as the sum of minterms**	**BTL-1**	**Remembering**	**PO1**
27	**Simplify: a) Y = AB'D + AB'D' b) Z = (A'+B)(A+B).**	**BTL-1**	**Remembering**	**PO1**
28	**What are Universal Gates? Why are they called so?**	**BTL-1**	**Remembering**	**PO1**
29	**Implement OR using NAND only.**	**BTL-1**	**Remembering**	**PO1**
30	**Implement NOR using NAND only.**	**BTL-1**	**Remembering**	**PO1**

PART B				
1	Reduce the expression using Quine McCluskey's method $F(x_1, x_2, x_3, x_4, x_5) = \sum m (0, 2, 4, 5, 6, 7, 8, 10, 14, 17, 18, 21, 29, 31) + \sum d (11, 20, 22)$ **(May/June 2016)**	**BTL-6**	**Creating**	**PO1, PO2, PO3, PO4**
2	Simplify the following switching functions using Quine McCluskey's tabulation method and realize expression using gates F(A,B,C,D) = Σ(0,5,7,8,9,10, 11, 14,15). **(Nov/Dec 2015)**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
3	Simplify the following switching functions using Karnaugh map method and realize expression using gates *F(A,B,C,D) = Σ(0,3,5,7,8,9,10,12,15).* **(Nov/Dec 2015)**	**BTL-1**	**Remembering**	**PO1**
4	(a) Express the following function in sum of min-terms and product of max-terms F(X,Y,Z)=X+YZ **(May 2015)** (b) convert the following logic system into NAND gates only. **(May 2015)**  	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
5	Simply the following Boolean expression in (i) sum of product (ii) product of sum using k-map AC'+B'D+A'CD+ABCD **(May 2015)**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
6	Simplify the Boolean function in SOP and POS F(A,B,C,D)=∑m(0,1,2,5,8,9,10) (**Dec2014)** (ii) plot the following Boolean function in k-map and simplify it. F(w,x,y,z) = ∑m(0,1,2,4,5,6,8,9,12,13,14). (**Dec2014)**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
7	Simply the function F(w,x,y,z)= ∑m(2,3,12,13,14,15) using tabulation method .Implement the simplified using gates.(**Dec2014)**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
8	Minimize the expression using quineMccluskey(tabulation)	**BTL-6**	**Creating**	**PO1,**

	F=∑m(0,1,9,15,24,29,30) +∑d(8,11,31). method **(June 2014)**			**PO2, PO3, PO4**
9	Simplify the following functions using K-map technique **(June 2014)** G=∑m (0,1,3,7,9,11) (ii) f(w,x;y,z)=∑m(0,7,8,9,10,12)+∑d(2,5,13).	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
10	Simplify the given boolean function in POS form using K-map and draw the logic diagram using Only NOR gates F(A,B,C,D)= ∑m (0,1,4,7,8,10,12,15)+d(2,6,11,14). **(Dec2013)** ii)Convert 78.5_{10} into binary. iii)Find the dual and complement of the following Boolean expression Xyz'+x'yz+z(xy+w).	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
11	3.Simplify the Boolean function using QuineMcCluskey method: F (A, B, C, D,E) = ∑m (0,1,3,7,13,14,21,26,28) + ∑d(2,5,9,11,17,24) **(Dec 2013)**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
12	Reduce the following function using K-map technique. **(Dec 2012)** i) f (A, B, C) = ∑m (0,1,3,7) + ∑d (2,5) ii) F (w,x,y,z) = ∑m (0,7,8,9,10,12) + ∑d (2,5,13)	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
13	Simlify the following Boolean function F using Tabulation method. i) F (A, B, C, D) = ∑m (0,6,8,13,14) ,d (A, B, C, D)= ∑m (2,4,10) **(Dec 2012)** ii) F (A, B, C, D) = ∑m (1,3,5,7,9,15) ,d (A, B, C, D)= ∑m (4,6,12,13)	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**

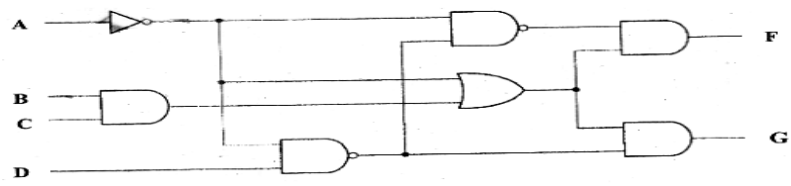
UNIT II

SYNCHRONOUS SEQUENTIAL LOGIC

Introduction to Sequential Circuits – Flip-Flops – operation and excitation tables, Triggering of FF, Analysis and design of clocked sequential circuits – Design – Moore/Mealy models, state minimization, state

assignment, circuit implementation - Registers – Counters.				
PART – A				
CO Mapping : CO202. 2				
S. No.	**Question**	**Blooms Taxanomy Level**	**Competence**	**PO**
1	**Design the combinational circuit with 3 inputs and 1 output. The output is 1 when the binary value of the input is less than 3. The output is 0 otherwise. (May/June 2016)**	**BTL-1**	**Remembering**	**PO1**
2	**Define Combinational circuits. (May/June 2016)**	**BTL-1**	**Remembering**	**PO1**
3	**Draw the truth table of half adder. (Nov./Dec. 2015)**	**BTL-1**	**Remembering**	**PO1**
4	**Write the Data flow description of a 4-bit Comparator. (April/May 2015)**	**BTL-1**	**Remembering**	**PO1**
5	**Implement a 4 bit even parity generator.**	**BTL-1**	**Remembering**	**PO1**
6	**Implement a 4 bit even parity checker.**	**BTL-1**	**Remembering**	**PO1**
7	**Write the data flow description of a 4-bit comparator. (May 2015)**	**BTL-1**	**Remembering**	**PO1**
8	**Implement a full adder with 4×1 multiplexer. (May 2015)**	**BTL-1**	**Remembering**	**PO1**
9	**Implement the following Boolean function using 8:1 multiplexer F(A,B,C)= ∑m(1,3,5,6)(Dec 2014)**	**BTL-1**	**Remembering**	**PO1**
10	**Draw a 2 to 1 multiplexer circuit. (June 2014)**	**(June 2014)**	**Remembering**	**PO1**
11	**What is priority encoder? (Dec 2014)**	**BTL-1**	**Remembering**	**PO1**
12	**Draw the truth table and circuit diagram of 4 to 2 encoder. (Dec 2013)**	**BTL-1**	**Remembering**	**PO1**
13	**Obtain the truth table for BCD to Excess-3 code converter. (Dec 2013)**	**BTL-1**	**Remembering**	**PO1**
14	**Write the stimulus for 2 to 1 line MUX. (June 2012)**	**BTL-1**	**Remembering**	**PO1**
15	**Distinguish between a decoder and a demultiplexer. (June**	**BTL-1**	**Remembering**	**PO1**

	2012)			
16	**Design a 2-bit binary to gray code converter.**	**BTL-1**	**Remembering**	**PO1**
17	**Draw the 4 bit Gray to Binary code converter.**	**BTL-1**	**Remembering**	**PO1**
18	**Draw the 4 bit Binary to Gray code converter.**	**BTL-1**	**Remembering**	**PO1**
19	**Distinguish between combinational logic and sequential logic.**	**BTL-1**	**Remembering**	**PO1**
20	**Implement half Adder using NAND Gates.**	**BTL-1**	**Remembering**	**PO1**
21	**Design a half subtractor.**	**BTL-1**	**Remembering**	**PO1**
22	**Give the truth table for half adder and write the expression for sum and carry.**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
23	**Mention the different type of binary codes.**	**BTL-1**	**Remembering**	**PO1**
24	**What is meant by self-complementing code?**	**BTL-1**	**Remembering**	**PO1**
25	**Draw the logic diagram of a one to four line de-multiplexer.**	**BTL-1**	**Remembering**	**PO1**
26	**List the advantages and disadvantages of BCD code**	**BTL-1**	**Remembering**	**PO1**
27	**Implement a full adder with two half adder.**	**BTL-1**	**Remembering**	**PO1**
28	**Define Tristate gates.**	**BTL-5**	**Evaluating**	**PO4**
29	**Define logic synthesis and simulation.**	**BTL-1**	**Remembering**	**PO1**
	PART B			
1	Implement the following Boolean function with 4 X 1 multiplexer and external gates. Connect inputs A and B to the selection lines. The input requiremnts for the four data lines will be a function of variables C and D these values are obatined by expressing F as a function of C and D for each four cases when AB = 00, 01, 10 and 11. These functions may have to be implemented with external gates. F(A, B, C, D) = Σ (1, 2, 5, 7, 8, 10, 11, 13, 15). **(May/June 2016)**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
2	Design a full adder with x, y, z and two outputs S and C. The circuits performs x+y+z, z is the input carry, C is the output carry and S is the Sum. **(May/June 2016)**	**BTL-6**	**Creating**	**PO1, PO2, PO3**
3	Design a code converter thet converts a 8421 to BCD code. **(Nov./Dec. 2015)**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3,**

				PO4
4	(i) Explain the Analysis procedure. Analyze the following logic diagram. **(April/May 2015)**  (ii) With neat diagram explain the 4-bit adder with carry lookahead.	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
5	(a) Design 2-bit magnitude comparator and write a verilog HDL code. (**Dec 2015**) (b)Implement the following Boolean functions with a multiplexer: F(w,x,y,z)= ∑(2,3,5,6,11,14,15) (c) Construct a 5 to 32 line decoder using 3 to 8 line decoders and 2 to 4 line decoder. **(May 2015)**	**BTL-2**	**Understanding**	**PO1, PO2**
6	Design and implement a 8241 to gray code converter. Realize the converter using only NAND gates (**Dec 2014**)	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
7	Design a circuit that converts 8421 BCD code to Excess-3 **(June 2014)** (b) Implement the following using 8 to 1 multiplexer. **(June 2014)**	BTL4		
8	(i).Realize 4 x 16 decoder using two 3 x 8 decoders with enable input. (ii) Implement the following functions using a multiplexer. F(W,X,Y,Z)= ∑m (0,1,3,4,8,9,15). (Dec 2013)	**BTL-2**	**Understanding**	**PO1, PO2**
9	5.(i).Design a combinational circuit to perform BCD addition. (ii).Design a 4-bit magnitude comparator with three outputs	**(Dec 2013)**	**Creating**	**PO1, PO2, PO3**

	:A<B ,A=B ,A>B. (Dec 2013)			
10	Construct a 4 to 16 line decoder with an enable input using five 2 to 4 line decoders with enable inputs. **(June 2012)**	F(W,X,Y, Z)= ∑m (0,1,3,4,8, 9,15).	**Understanding**	**PO1, PO2**
11	Design a BCD to 7 segment decoder and implement it by using basic gates. **(Dec 2012)**	**BTL-6**	**Creating**	**PO1, PO2, PO3**
12	1. Discuss the need and working principle of Carry Look ahead adder. **(Dec 2012)**	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
13	Design a full adder using 2 half adders.	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**
14	Design a logic circuit that accepts a 4 bit Gray code and converts it into 4 bit binary code.	**BTL-5**	**Evaluating**	**PO1, PO2, PO3, PO4**

UNIT III

Functional Units of a Digital Computer: Von Neumann Architecture – Operation and Operands of Computer Hardware Instruction – Instruction Set Architecture (ISA): Memory Location, Address and Operation – Instruction and Instruction Sequencing – Addressing Modes, Encoding of Machine Instruction – Interaction between Assembly and High Level Language

PART-A

Q. No.	Questions	CO	Bloom's Level
1.	**Write the basic functional units of computer?** The basic functional units of a computer are input unit, output unit, memory unit, ALU unit and control unit	C204.1	BTL1
2.	**Write the basic functional units of computer? (APR/MAY 2017,NOV/DEC 2017)** The basic functional units of a computer are input unit, output unit, memory unit, ALU unit and control unit.	C204.1	BTL1

3.	**What is a bus? What are the different buses in a CPU? [APR/MAY 2011]** A group of lines that serve as a connecting path for several devices is called bus .The different buses in a CPU are 1] Data bus 2] Address bus 3] Control bus.	C204.1	BTL1
4.	**What is meant by stored program concepts?** Stored program concept is an idea of storing the program and data in the memory	C204.1	BTL1
5	**Define multiprogramming?(A.U.APR/MAY 2013)** Multiprogramming is a technique in several jobs are in main memory at once and the processor is switched from job as needed to keep several jobs advancing while keeping the peripheral devices in use.	C204.1	BTL1
6	**What is meant by VLSI technology?** VLSI is the abbreviation for Very Large Scale Integration. In this technology millions of transistors are put inside a single chip as tiny components. The VLSI chips do the function of millions of transistors. These are Used to implement parallel algorithms directly in hardware	C204.1	BTL6
7	**Define multiprocessing?** Multiprocessing is the ability of an operating system to support more than one process at the same time	C204.1	BTL6

8	**List the eight great ideas invented by computer architecture? APR/MAY-2015** • Design for Moore's Law • Use abstraction to simplify design • Make the common case fast • Performance via Parallelism • Performance via Pipelining • Performance via Prediction • Hierarchy of Memory • Dependability via Redundancy	C204.1	BTL1
9	**Define power wall.** • Old conventional wisdom • Power is free • Transistors are expensive • New conventional wisdom: "Power wall" • Power expensive • Transistors"free" (Can put more on chip than can afford to turn on)	C204.1	BTL1
10	**What are clock and clock cycles?** The timing signals that control the processor circuits are called as clocks. The clock defines regular time intervals called clock cycles.	C204.1	BTL1
11	**What is uniprocessor?** A **uniprocessor system** is defined as a computer system that has a single central processing unit that is used to execute computer tasks. As more and more modern software is able to make use of multiprocessing architectures, such as SMP and MPP, the term *uniprocessor* is therefore used to distinguish the class of computers where all processing tasks share a single CPU.	C204.1	BTL1
12	**What is multicore processor?** A multi-core processor is a single computing component with two or more independent actual central processing units (called "cores"), which are the units that read and execute program instructions.The instructions are ordinary CPU instructions such as add, move data, and branch, but the multiple cores can run multiple instructions at the same time, increasing overall speed for programs amenable to parallel computing	C204.1	BTL1

13	**Differentiate super computer and mainframe computer.** A computer with high computational speed, very large memory and parallel structured hardware is known as a super computer.EX: CDC 6600. Mainframe computer is the large computer system containing thousands of IC's. It is a room-sized machine placed in special computer centers and not directly accessible to average users. It serves as a central computing facility for an organization such as university, factory or bank.	C204.1	BTL1
14	**Differentiate between minicomputer and microcomputer.** Minicomputers are small and low cost computers are characterized by Short word size i.e. CPU word sizes of 8 or 16 bits. They have limited hardware and software facilities. They are physically smaller in size.Microcomputer is a smaller, slower and cheaper computer packing all the electronics of the computer in to a handful of IC's, including CPU and memory and IO chips	C204.1	BTL1
15	**What is instruction register?(NOV/DEC 2016)** The instruction register (IR) holds the instruction that is currently being executed. Its output is available to the control circuits which generate the timing signals that control the various processing elements involved in executing the instruction.	C204.1	BTL1
16	**What is program counter?** The program counter (PC) keeps track of the execution of a program. It contains the memory address of the next instruction to be fetched and executed.	C204.1	BTL1
17	**What is processor time?** The sum of the periods during which the processor is active is called the processor time	C204.1	BTL1
18	**Give the CPU performance equation.** CPU execution time for a program **=**Instruction Count **X**Clock cycles per instruction**X**Clock cycle time.	C204.1	BTL1

19	**What is superscalar execution?** In this type of execution, multiple functional units are used to create parallel paths through which different instructions can be executed in parallel. So it is possible to start the execution of several instructions in every clock cycle. This mode of operation is called superscalar execution	C204.1	BTL1
20	**What is RISC and CISC?** The processors with simple instructions are called as Reduced Instruction Set Computers (RISC). The processors with more complex instructions are called as Complex Instruction Set Computers (CISC).	C204.1	BTL1
21	**List out the methods used to improve system performance.** The methods used to improve system performance are • Processor clock • Basic Performance Equation • Pipelining • Clock rate • Instruction set • Compiler	C204.1	BTL1
22	**Define addressing modes and its various types.(nov/dec 2017)** The different ways in which the location of a operand is specified in an instruction is referred to as addressing modes. The various types are Immediate Addressing, Register Addressing, Based or Displacement Addressing, PC-Relative Addressing, Pseudodirect Addressing.	C204.1	BTL1
23	**Define register mode addressing.** In register mode addressing, the name of the register is used to specify the operand. Eg. Add $s3, $s5,$s6.	C204.1	BTL1
24	**Define Based or Displacement mode addressing.** In based or displacement mode addressing, the operand is in a memory location whose address is the sum of a register and a constant in the instruction. Eg. lw $t0,32($s3).	C204.1	BTL1

25	**State Amdahl's Law.** Amdahl's law is a formula used to find the maximum improvement improvement possible by improving a particular part of a system. In parallel computing, Amdahl's law is mainly used to predict the theoretical maximum speedup for program processing using multiple processors. $$\text{Speedup} = \frac{\text{Performance for entire task using the enhancement when possible}}{\text{Performance for entire task without using the enhancement}}$$ Alternatively, $$\text{Speedup} = \frac{\text{Execution time for entire task without using the enhancement}}{\text{Execution time for entire task using the enhancement when possible}}$$	C204.1	BTL1
26	**Define Relative mode addressing.** **(Nov 2014)** In PC-relative mode addressing, the branch address is the sum of the PC and a constant in the instruction. - In the relative address mode, the effective address is determined by the index mode by using the program counter in stead of general purpose processor register. This mode is called relative address mode.	C204.1	BTL1
27	**Distinguish pipelining from parallelism APR/MAY 2015** parallelism means we are using more hardware for the executing the desired task. in parallel computing more than one processors are running in parallel. there may be some dedicated hardware running in parallel for doing the specific task. while the pipelining is an implementation technique in which multiple instructions are overlapped ninexecution.parallelism increases the performance but the area also increases. in case of pipelining the performance and througput increases at the cost of pipelining registers area pipelining there are different hazards like data hazards, control hazards etc.	C204.1	BTL1

28	**Distinguish pipelining from parallelism APR/MAY 2015** parallelism means we are using more hardware for the executing the desired task. in parallel computing more than one processors are running in parallel. there may be some dedicated hardware running in parallel for doing the specific task. while the pipelining is an implementation technique in which multiple instructions are overlapped ninexecution.parallelism increases the performance but the area also increases. in case of pipelining the performance and througput increases at the cost of pipelining registers area pipelining there are different hazards like data hazards, control hazards etc.	C204.1	BTL1
29	**How to represent Instruction in a computer system?MAY/JUNE 2016** Computer instructions are the basic components of a machine language program. They are also known as *macrooperations*, since each one is comprised of a sequences of microoperations. Each instruction initiates a sequence of microoperations that fetch operands from registers or memory, possibly perform arithmetic, logic, or shift operations, and store results in registers or memory. Instructions are encoded as binary *instruction codes*. Each instruction code contains of a*operation code*, or *opcode*, which designates the overall purpose of the instruction (e.g. add, subtract, move, input, etc.). The number of bits allocated for the opcode determined how many different instructions the architecture supports. In addition to the opcode, many instructions also contain one or more *operands*, which indicate where in registers or memory the data required for the operation is located. For example, `add` instruction requires two operands, and a `not` instruction requires one.	C204.1	BTL3
30	**Brief about relative addressing mode.<u>NOV/DEC 2014</u>** **Relative addressing mode -** In the relative address mode, the effective address is determined by the index mode by using the program counter in stead of general purpose processor register. This mode is called relative address mode.	C204.1	BTL1

31	**Distinguish between auto increment and auto decrement addressing mode?** **MAY/JUNE 2016** A special case of indirect register mode. The register whose number is included in the instruction code, contains the address of the operand. Autoincrement Mode = after operand addressing , the contents of the register is incremented. Decrement Mode = before operand addressing, the contents of the register is decrement. We denote the autoincrement mode by putting the specified register in parentheses, to show that the contents of the register are used as the efficient address, followed by a plus sign to indicate that these contents are to be incremented after the operand is accessed. Thus, using register R4, the autoincrement mode is written as (R4)+. As a companion for the autoincrement mode, another mode is often available in which operands are accessed in the reverse order. *Autodecrementmode* The contents of a register specified in the instruction are decremented. These contents are then used as the effective address f the operand. We denote the autodecrement mode by putting the specified register in parentheses, preceded by a minus sign to indicate that the contents of register are to be decremented before being used as the effective address. Thus, we write (R4). This mode allows the accessing of operands in the direction of descending addresses. The action performed by the autoincrement and auto decrement addressing modes can be achieved using two instruction, one to access the operand and the other to increment or to decrement the register that contains the operand address. Combining the two operations in one instruction reduces the number if instructions needed to perform the task.	C204.1	BTL1
32	**If computer A runs a program in 10 seconds and computer B runs the same program in 15 seconds how much faster is A than B?** We know that A is *n* times as fast as B if $$\frac{\text{Performance}_A}{\text{Performance}_B} = \frac{\text{Execution time}_B}{\text{Execution time}_A} = n$$ Thus the performance ratio is $$\frac{15}{10} = 1.5$$ and A is therefore 1.5 times as fast as B. In the above example, we could also say that computer B is 1.5 times *slower*	C204.1	BTL1

	than computer A, since $$\frac{\text{Performance}_A}{\text{Performance}_B} = 1.5$$ means that $$\frac{\text{Performance}_A}{1.5} = \text{Performance}_B$$		
33	**Our favorite program runs in 10 seconds on computer A, which has a 2 GHz clock. We are trying to help a computer designer build a computer, B, which will run this program in 6 seconds. The designer has determined that a substantial increase in the clock rate is possible, but this increase will affect the rest of the CPU design, causing computer B to require 1.2 times as many clock cycles as computer A for this program. What clock rate should we tell the designer to target?** Let's first find the number of clock cycles required for the program on A: $$\text{CPU time}_A = \frac{\text{CPU clock cycles}_A}{\text{Clock rate}_A}$$ $$10 \text{ seconds} = \frac{\text{CPU clock cycles}_A}{2 \times 10^9 \frac{\text{cycles}}{\text{second}}}$$ $$\text{CPU clock cycles}_A = 10 \text{ seconds} \times 2 \times 10^9 \frac{\text{cycles}}{\text{second}} = 20 \times 10^9 \text{ cycles}$$ CPU time for B can be found using this equation: $$\text{CPU time}_B = \frac{1.2 \times \text{CPU clock cycles}_A}{\text{Clock rate}_B}$$ $$6 \text{ seconds} = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{\text{Clock rate}_B}$$	C204.1	BTL1

	$$\text{Clock rate}_B = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{6 \text{ seconds}} = \frac{0.2 \times 20 \times 10^9 \text{ cycles}}{\text{second}} = \frac{4 \times 10^9 \text{ cycles}}{\text{second}} = 4$$ To run the program in 6 seconds, B must have twice the clock rate of A.		
34	**Suppose we have two implementations of the same instruction set architecture. Computer A has a clock cycle time of 250 ps and a CPI of 2.0 for some program, and computer B has a clock cycle time of 500 ps and a CPI of 1.2 for the same program. Which computer is faster for this program and by how much?** We know that each computer executes the same number of instructions for the program; let's call this number *I*. First, find the number of processor clock cycles for each computer: $$\text{CPU clock cycles}_A = I \times 2.0$$ $$\text{CPU clock cycles}_B = I \times 1.2$$ Now we can compute the CPU time for each computer: $$\text{CPU time}_A = \text{CPU clock cycles}_A \times \text{Clock cycle time} = I \times 2.0 \times 250 \text{ ps} = 500 \times I \text{ ps}$$ Likewise, for B: $$\text{CPU time}_B = I \times 1.2 \times 500 \text{ ps} = 600 \times I \text{ ps}$$ Clearly, computer A is faster. The amount faster is given by the ratio of the execution times: $$\frac{\text{CPU performance}_A}{\text{CPU performance}_B} = \frac{\text{Execution time}_B}{\text{Execution time}_A} = \frac{600 \times I \text{ ps}}{500 \times I \text{ ps}} = 1.2$$	C204.1	BTL1

	We can conclude that computer A is 1.2 times as fast as computer B for this program.		
35	**Define CPU execution time and list the types.** **CPU execution time** Also called **CPU time**. The actual time the CPU spends computing for a specific task. **<u>Types:</u>** **User CPU time** The CPU time spent in a program itself. **System CPU time** The CPU time spent in the operating system performing tasks on behalf of the program	C204.1	BTL1
36	**Define response time** **Response time:** Also called **execution time**. The total time required for the computer to complete a task, including disk accesses, memory accesses, I/O activities, operating system overhead, CPU execution time, and so on.	C204.1	BTL1
37	**What is Throughput?** Also called **bandwidth**. Another measure of performance, it is the number of tasks completed per unit time.	C204.1	BTL1
38	**Define Clock cycles:** All computers are constructed using a **clock that determines when events take place in the hardware.** These discrete time intervals are called **clock cycles** (or **ticks**, **clock ticks**, **clock periods**, **clocks**, **cycles**).	C204.1	BTL1

39	**Write Basic performance equation in terms of instruction count (the number of instructions executed by the program), CPI, and clock cycle time.** $$\text{CPU time} = \text{Instruction count} \times \text{CPI} \times \text{Clock cycle time}$$ or, the clock rate is the inverse of clock cycle time: $$\text{CPU time} = \frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}}$$	C204.1	BTL1

40	**Compile given Two C Assignment Statements into MIPS**
	`a = b + c;`
	`d = a - e;`
	Answer
	`add a, b, c`
	`sub d, a, e`
		C204.1	BTL1

41	**Compile givenC Assignment Statement into MIPS**
	
	`f = (g + h) - (i + j);`
	`add t0,g,h # temporary variable t0 contains g + h`
	`add t1,i,j # temporary variable t1 contains i + j`
	`sub f,t0,t1 # f gets t0 -t1, which is`
	`(g + h) - (i + j)`
		C204.1	BTL1

42	**Compile givenC Assignment Statement into MIPS**
	`g = h + A[8];`
	Answer
	The first compiled instruction is
	`lw$t0,8($s3) # Temporary reg $t0 gets A[8]`
	`add$s1,$s2,$t0 # g = h + A[8]`
		C204.1	BTL1

43	**What are the three types of operands in MIPS** **1** .word 2.Memory Operands 3.Constant or Immediate Operands	C204.1	BTL1

44	**Compile givenC Assignment Statement into MIPS**
	
	`A[12] = h + A[8];`
	Answer
	`$t0: lw$t0,32($s3)`
	`# Temporary reg $t0 gets A[8]`
	`add$t0,$s2,$t0`
	`# Temporary reg $t0 gets h + A[8]`
	`sw$t0,48($s3)`
	`# Stores h + A[8] back into A[12]`
		C204.1	BTL1

45	**Write MIPS To add 4 to register `$s3`.** `addi$s3,$s3,4# $s3 = $s3 + 4`	C204.1	BTL1
46	**Define Instruction format** A form of representation of an instruction composed of fields of binary numbers.The numeric version of instructions **machine language** and a sequence of such instructions *machine code*.	C204.1	BTL1
47	**What are the types of instruction format in MIPS** 1. *R-type* (for register) or *R-format*. *2.I-type* (for immediate) or *I-format* *3.J-type or Jump*	C204.1	BTL1
48	**What are the types of instruction in MIPS.(APR/MAY2018)** 1. **Arithmetic instruction** 2. **Data transfer Instruction** 3. **Logical Instruction** 4. **Conditional Branch Instruction** 5. **Unconditional jump Instruction**	C204.1	BTL1

49	**Compile givenC Statement into MIPS**
	`if (i == j) f = g + h; else f = g – h;`
	
	`bne $s3,$s4,Else# go to Else if i ≠ j`
	`add $s0,$s1,$s2# f = g + h (skipped if i ≠ j)`
		C204.1	BTL1

50	**Compile givenC Statement into MIPS**
	
	`while (save[i] == k)`
	`i += 1;`
	`Ans:`
	`Loop: sll$t1,$s3,2# Temp reg $t1 = i * 4`
	`add $t1,$t1,$s6# $t1 = address of save[i]`
	`lw $t0,0($t1)`
	`# Temp reg $t0 = save[i]`
	`bne $t0,$s5, Exit`
	`# go to Exit if save[i] ≠ k`
	`addi $s3,$s3,1# i = i + 1`
	`jLoop# go to Loop`
	`Exit:`
		C204.1	BTL1

51	**State indirect addressing mode give example**.(**APR/May 2017**) Indirect Mode. The effective address of the operand is the contents of a register or main memory location, location whose address appears in the instruction. ... Once it's there, instead of finding an operand, it finds an address where the operand is located. LOAD R1, @R2 Load the content of the memory address stored atregister R2 to register R1.	C204.1	BTL1

PART-B

Q. No.	Questions	CO	Bloom's Level
1.	**i)**Discuss in detail about Eight great ideas of computer Architecture.**(8)** ***Page.No:11-13)*** **ii**) Explain in detail about Technologies for Building Processors and Memory **(8))*(Page.No:24-28)***	C204.1	BTL5
2.	Explain the various components of computer System with neat diagram **(16)**	C204.	BTL5

	.(NOV/DEC2014,NOV/DEC2015,APR/MAY 2016,NOV/DEC 2016,APR/MAY2018)) ***(Page.No:16-17)***	1	
3.	Discuss in detail the various measures of performance of a computer(16) ***(Page.No:28-40)***	C204.1	BTL6
4.	Define Addressing mode and explain the different types of basic addressing modes with an example **(APRIL/MAY2015,NOV/DEC2015,APR/MAY 2016,NOV/DEC 2016,APR/MAY2018)** ***(Page.No:116-117)***	C204.1	BTL5
5.	**i)Discuss the Logical operations and control operations of computer (12)** ***(Page.No:87-89)*** **ii)Write short notes on Power wall(6)** ***(Page.No:40-42)***	C204.1	BTL6
6.	Consider three diff erent processors P1, P2, and P3 executing the same instruction set. P1 has 3 GHz clock rate and a CPI of 1.5. P2 has a 2.5 GHz clock rate and a CPI of 1.0. P3 has a 4.0 GHz clock rate and has a CPI of 2.2. **(APR/MAY 2018)** a. Which processor has the highest performance expressed in instructions per second? b. If the processors each execute a program in 10 seconds, find the number of cycles and the number of instructions. c. We are trying to reduce the execution time by 30% but this leads to an increase of 20% in the CPI. What clock rate should we have to get this time reduction? **(Refer Notes)**	C204.1	BTL5

7.	Explain various instruction format illustrate the same with an example **NOV/DEC2017** ***(Page.No:80-86)***	C204.1	BTL5
8.	Explain direct ,immediate ,relative and indexed addressing modes with example **APR/MAY2018** ***(Page.No:116-117)***	C204.1	BTL5
9.	State the CPU performance equation and the factors that affect performance (8) **(NOV/DEC2014) (Refer Notes)**	C204.1	BTL5
10.	Discuss about the various techniques to represent instructions in a computer system. **(APRIL/MAY2015,NOV/DEC 2017)** ***(Page.No:80-86)***	C204.1	BTL6
11.	What is the need for addressing in a computer system?Explain the different addressing modes with suitable examples.**(APRIL/MAY2015)** ***(Page.No:116-117)***	C204.1	BTL5
12.	**Explain types of operations and operands with examples.(NOV/DEC 2017)** ***(Page.No:63-70)***	C204.1	BTL5
13.	Consider two diff erent implementations of the same instruction set architecture. Th e instructions can be divided into four classes according to their CPI (class A, B, C, and D). P1 with a clock rate of 2.5 GHz and CPIs of 1, 2, 3, and 3, and P2 with a clock rate of 3 GHz and CPIs of 2, 2, 2, and 2. Given a program with a dynamic instruction count of 1.0E6 instructions divided into classes as follows: 10% class A, 20% class B, 50% class C, and 20% class D,	C204.1	BTL5

	which implementation is faster? a. What is the global CPI for each implementation? b. Find the clock cycles required in both cases. **(Refer Notes)**		
14.	To what should the CPI of load/store instructions be reduced in order for a single processor to match the performance of four processors using the original CPI values? **(Refer Notes)**	C204.1	BTL5
15.	Describe the steps that transform a program written in a high-level language such as C into a representation that is directly executed by a computer processor. **(Refer Notes)**	C204.1	BTL4

UNIT IV 9

Instruction Execution – Building a Data Path – Designing a Control Unit – Hardwired Control, Microprogrammed Control – Pipelining – Data Hazard – Control Hazards – Exceptions.

PART-A

Q. No.	Questions	CO	Bloom's Level
1.	**What is pipelining?** The technique of overlapping the execution of successive instruction for substantial improvement in performance is called pipelining. .	C204.3	BTL1
2.	**What is and precise exception?**	C204.3	BTL1

	A precise exception is one in which all instructions prior to the faulting instruction are complete and instruction following the faulting instruction, including the faulty instruction; do not change the state of the machine.		
3.	**Define processor cycle in pipelining.** The time required between moving an instruction one step down the pipeline is a processor cycle.	C204.3	BTL1
4.	**What is meant by pipeline bubble?(NOV/DEC 2016)** To resolve the hazard the pipeline is stall for 1 clock cycle. A stall is commonly called a pipeline bubble, since it floats through the pipeline taking space but carrying no useful work.	C204.3	BTL1
5	**What is pipeline register delay?** Adding registers between pipeline stages me adding logic between stages and setup and hold times for proper operations. This delay is known as pipeline register delay.	C204.3	BTL1
6	**What are the major characteristics of a pipeline?** The major characteristics of a pipeline are: 1. Pipelining cannot be implemented on a single task, as it works by splitting multiple tasks into a number of subtasks and operating on them simultaneously. The speedup or efficiency achieved by suing a pipeline depends on the number of pipe stages and the number of available tasks that can be subdivided	C204.3	BTL6

7	**What is data path?(NOV/DEC 2016,APR/MAY2018)** As instruction execution progress data are transferred from one instruction to another, often passing through the ALU to perform some arithmetic or logical operations. The registers, ALU, and the interconnecting bus are collectively referred as the data path.	C204.3	BTL6
8	**What is a pipeline hazard and what are its types?** Any condition that causes the pipeline to stall is called hazard. They are also called as stalls or bubbles. The various pipeline hazards are: Hazard Control Hazard	C204.3	BTL1
9	**What is Instruction or control hazard?** The pipeline may be stalled because of a delay in the availability of an instruction. For example, this may be a result of a miss in the cache, requiring the instruction to be fetched from the main memory. Such hazards are often called control hazards or instruction hazard.	C204.3	BTL1
10	**Define structural hazards.** This is the situation when two instruction require the use of a given hardware resource at the same time. The most common case in which this hazard may arise is in access to memory	C204.3	BTL1
11	**What is side effect?** When a location other than one explicitly named in an instruction as a destination operand is affected, the instruction is said to have a side effect	C204.3	BTL1
12	**What do you mean by branch penalty?** The time lost as a result of a branch instruction is often referred to as branch penalty	C204.3	BTL1
13	**What is branch folding?** When the instruction fetch unit executes the branch instruction concurrently with the execution of the other instruction, then this technique is called branch folding.	C204.3	BTL1

14	**What do you mean by delayed branching?** Delayed branching is used to minimize the penalty incurred as a result of conditional branch instruction. The location following the branch instruction is called delay slot. The instructions in the delay slots are always fetched and they are arranged such that they are fully executed whether or not branch is taken. That is branching takes place one instruction later than where the branch instruction appears in the instruction sequence in the memory hence the name delayed branching	C204.3	BTL1
15	**Define exception and interrupt.** **Dec 2012,NOV/DEC 14,MAY/JUNE 2016,APR/MAY2018))** **Exception:** The term exception is used to refer to any event that causes an interruption. **Interrupt:** An exception that comes from outside of the processor. There are two types of interrupt. 1. Imprecise interrupt and 2.Precise interrupt	C204.3	BTL1
16	**Why is branch prediction algorithm needed? Differentiate between the static and dynamic techniques. (May 2013,APR/MAY 2015,NOV/DEC 15)** The branch instruction will introduce branch penalty which would reduce the gain in performance expected from pipelining. Branch instructions can be handled in several ways to reduce their negative impact on the rate of execution of instructions. Thus the branch prediction algorithm is needed. **Static Branch prediction** The static branch prediction, assumes that the branch will not take place and to continue to fetch instructions in sequential address order. **Dynamic Branch prediction** The idea is that the processor hardware assesses the likelihood of a given	C204.3	BTL1

	branch being taken by keeping track of branch decisions every time that instruction is executed. The execution history used in predicting the outcome of a given branch instruction is the result of the most recent execution of that instruction.		
17	**What is branch Target Address?** The address specified in a branch, which becomes the new program counter, if the branch is taken. In MIPS the branch target address is given by the sum of the offset field of the instruction and the address of the instruction following the branch	C204.3	BTL1
18	**How do control instructions like branch, cause problems in a pipelined processor?** Pipelined processor gives the best throughput for sequenced line instruction. In branch instruction, as it has to calculate the target address, whether the instruction jump from one memory location to other. In the meantime, before calculating the larger, the next sequence instructions are got into the pipelines, which are rolled back, when target is calculated.	C204.3	BTL1
19	**What is meant by super scalar processor?** Super scalar processors are designed to exploit more instruction level parallelism in user programs. This means that multiple functional units are used. With such an arrangement it is possible to start the execution of several instructions in every clock cycle. This mode of operation is called super scalar execution.	C204.3	BTL1
20	**Define pipeline speedup. [APR/MAY 2012] (A.U.NOV/DEC 2012)** Speed up is the ratio of the average instruction time without pipelining to the average instruction time with pipelining. Average instruction time without pipelining Speedup= Average instruction time with pipelining	C204.3	BTL1

21	**What is Vectorizer?** The process to replace a block of sequential code by vector instructions is called vectorization. The system software, which generates parallelism, is called as vectorizing compiler.	C204.3	BTL1
22	**What is pipelined computer?** When hardware is divided in to a number of sub units so as to perform the sub operations in an overlapped fashion is called as a pipelined computer.	C204.3	BTL1
23	**List the various pipelined processors.** 8086, 8088, 80286, 80386. STAR 100, CRAY 1 and CYBER 205 etc	C204.3	BTL1
24	**Classify the pipeline computers.** Based on level of processing → processor pipeline, instruction pipeline, arithmetic pipelines Based on number of functions→ Uni-functional and multi functional pipelines. Based on the configuration → Static and Dynamic pipelines and linear and non linear pipelines Based on type of input→ Scalar and vector pipelines.**Asf**	C204.3	BTL1
25	**Define Pipeline speedup. (Nov/Dec 2013)** The ideal speedup Time per instruction on unpipelined machine / Number of pipe stages from a pipeline is equal to the number of stages in the pipeline.	C204.3	BTL1

26	**Write down the expression for speedup factor in a pipelined architecture. [MAY/JUNE '11]** The speedup for a pipeline computer is S = (k + n -1) tp Where,K → number of segments in a pipeline,N → number of instructions to be executed. Tp → cycle time	C204.3	BTL1
27	**What are the problems faced in instruction pipeline.** Resource conflicts → Caused by access to the memory by two at the same time. Most of the conflicts can be resolved by using separate instruction and data memories. Data dependency → Arises when an instruction depends on the results of the previous instruction but this result is not yet available. Branch difficulties → Arises from branch and other instruction that change the value of PC (Program Counter).	C204.3	BTL1
28	**What is meant by vectored interrupt? (Nov/Dec 2013)** An interrupt for which the address to which control is transferred is determined by the cause of the exception.	C204.3	BTL1
29	**What is the need for speculation?NOV/DEC 2014** One of the most important methods for finding and exploiting more ILP is speculation. It is an approach whereby the compiler or processor guesses the outcome of an instruction to remove it as dependence in executing other instructions. For example, we might speculate on the outcome of a branch, so	C204.3	BTL3

	that instructions after the branch could be executed earlier. Speculation (also known as *speculative loading*), is a process implemented in Explicitly Parallel Instruction Computing (EPIC) processors and their compiler s to reduce processor-memory exchanging bottlenecks or latency by putting all the data into memory in advance of an actual load instruction		
30	Define Imprecise , Precise interrupt Imprecise interrupt Also called imprecise exception. Interrupts or exceptions in pipelined computers that are not associated with the exact instruction that was the cause of the interrupt or exception. Precise interrupt Also called precise exception. An interrupt or exception that is always associated with the correct instruction in pipelined computers	C204.3	BTL1
31	**What are the advantages of pipelining?MAY/JUNE 2016** The cycle time of the processor is reduced; increasing the instruction throughput.Some combinational circuits such as adders or multipliers can be made faster by adding more circuitry. If pipelining is used instead, it can save circuitry vs. a more complex combinational circuit.	C204.3	BTL1
32	What is Program counter (PC)(Fetching) The register containing the address of the instruction in the program being executed	C204.3	BTL1
33	What is Adder: An adder is needed to compute the next instruction address. The adder is an ALU wired to always add its two 32-bit inputs and place the sum on its output.	C204.3	BTL1

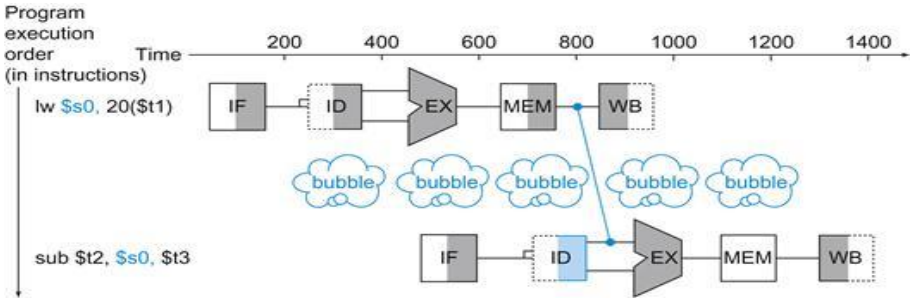
<table>
<tr><td>34</td><td>What is Register file(decoding):
A state element that consists of a set of registers that can be read and written by supplying a register number to be accessed.</td><td>C204.3</td><td>BTL1</td></tr>
<tr><td>35</td><td>Define Sign-extend in data path.
To increase the size of a data item by replicating the high-order sign bit of the original data item in the high-order bits of the larger, destination data item. a unit to sign-extend the 16-bit offset field in the instruction to a 32-bit signed value</td><td>C204.3</td><td>BTL1</td></tr>
<tr><td>36</td><td>Define Shifter:
■ The jump instruction operates by replacing the lower 28 bits of the PC with the lower 26 bits of the instruction shifted left by 2 bits. Simply concatenating 00 to the jump offset accomplishes this shift</td><td>C204.3</td><td>BTL1</td></tr>
<tr><td>37</td><td>What is Delayed branch?
A type of branch where the instruction immediately following the branch is always executed, independent of whether the branch condition is true or false.</td><td>C204.3</td><td>BTL1</td></tr>
<tr><td>38</td><td>What are the control lines of MIPS functions.
<table><tr><th>ALU control lines</th><th>Function</th></tr><tr><td>0000</td><td>AND</td></tr><tr><td>0001</td><td>OR</td></tr><tr><td>0010</td><td>add</td></tr><tr><td>0110</td><td>sub</td></tr></table></td><td>C204.3</td><td>BTL1</td></tr>
</table>

<table>
<tr><td></td><td><table><tr><td>0111</td><td>Set lessthan</td></tr><tr><td>1100</td><td>NOR</td></tr></table></td><td></td><td></td></tr>
<tr><td>39</td><td>Define Don’t-care term
An element of a logical function in which the output does not depend on the values of all the inputs</td><td>C204.3</td><td>BTL1</td></tr>
<tr><td>40</td><td>What are the Function of seven control lines?
<table><tr><th>Signal name</th><th>Effect when deasserted</th><th>Effect when asserted</th></tr><tr><td>RegDst</td><td>The register destination number for the Write register comes from the rt field (bits 20:16).</td><td>The register destination number for the Write register comes from the rd field (bits 15:11).</td></tr><tr><td>RegWrite</td><td>None.</td><td>The register on the Write register input is written with the value on the Write data input.</td></tr><tr><td>ALUSrc</td><td>The second ALU operand comes from the second register file output (Read data 2).</td><td>The second ALU operand is the sign-extended, lower 16 bits of the instruction.</td></tr><tr><td>PCSrc</td><td>The PC is replaced by the output of the adder that computes the value of PC + 4.</td><td>The PC is replaced by the output of the adder that computes the branch target.</td></tr><tr><td>MemRead</td><td>None.</td><td>Data memory contents designated by the address input are put on the Read data output.</td></tr><tr><td>MemWrite</td><td>None.</td><td>Data memory contents designated by the address input are replaced by the value on the Write data input.</td></tr><tr><td>MemtoReg</td><td>The value fed to the register Write data input comes from the ALU.</td><td>The value fed to the register Write data input comes from the data memory.</td></tr></table></td><td>C204.3</td><td>BTL1</td></tr>
<tr><td>41</td><td>What are the Disadvantages of single cycle implementation?
• Although the single-cycle design will work correctly, it would not be used in modern designs because it is inefficient.
• Although the CPI is 1 the overall performance of a single-cycle implementation is likely to be poor, since the clock cycle is too long.
• The penalty for using the single-cycle design with a fixed clock cycle is significant,.
• To implement the floating-point unit or an instruction set with more complex instructions, this single-cycle design wouldn’t work well .</td><td>C204.3</td><td>BTL1</td></tr>
</table>

	• A single-cycle implementation thus violates the great idea of making the **common case fast**.		
42	What is Structural hazard? When a planned instruction cannot execute in the proper clock cycle because the hardware does not support the combination of instructions that are set to execute. If there is a single memory instead of two memories. If the pipeline had a fourth instruction, that in the same clock cycle the first instruction is accessing data from memory while the fourth instruction is fetching an instruction from that same memory. Without two memories, pipeline could have a structural hazard. **<u>To avoid structural hazards</u>** • When designing a pipeline designer can change the design By providing sufficient resources	C204.3	BTL1

43	Define Data Hazards. .(**APR/MAY 2017**)
	Data hazard is also called a **pipeline data hazard**. When a planned instruction cannot execute in the proper clock cycle because data that is needed to execute the instruction is not yet available.
	• In a computer pipeline, data hazards arise from the dependence of one instruction on an earlier one that is still in the pipeline
	• ***<u>Example:</u>***
	add instruction followed immediately by a subtract instruction that uses the sum (`$s0`):
	`add$s0, $t0, $t1`
	`sub$t2, $s0, $t3`
		C204.3	BTL1

44	Define data Forwarding Forwarding is also called as **bypassing**. A method of resolving a data hazard by retrieving the missing data element from internal buffers rather than waiting for it to arrive from programmer-visible registers or memory.	C204.3	BTL1
45	Define load-use data hazard A specific form of data hazard in which the data being loaded by a load instruction has not yet become available when it is needed by another instruction	C204.3	BTL1

46	Define Pipeline stall Pipeline stall is also called as **bubble**. A stall initiated in order to resolve a hazard.  	C204.3	BTL1
47	What is Control Hazard? Control hazard is also called as **branch hazard**. When the proper instruction cannot execute in the proper pipeline clock cycle because the instruction that was fetched is not the one that is needed; that is, the flow of instruction addresses is not what the pipeline expected.	C204.3	BTL1
48	What are theSchemes for resolving control hazards ? 1. Assume Branch Not Taken: 2. Reducing the Delay of Branches: 3. Dynamic Branch Prediction:	C204.3	BTL1
49	**D e** fine Branch delay slot The slot directly after a delayed branch instruction, which in the MIPS architecture is filled by an instruction that does not affect the branch.	C204.3	BTL1
50	Define Correlating , Tournament branch predictor Correlating predictor A branch predictor that combines local behavior of a particular branch and global information about the behavior of some recent number of executed branches. Tournament branch predictor A branch predictor with multiple predictions for each branch and a selection mechanism that chooses which predictor to enable for a given branch	C204.3	BTL1

51	Name control signal to perform arithmetic operation.(APR/MAY 2017) 1.Regdst 2.Regwrite 3.ALU Src	C204.3	BTL1
52	what is ideal cycle per instruction in pipelining?(APR/MAY 2018) With pipelining, a new instruction is fetched every clock cycle by exploiting instruction-level parallelism, therefore, since one could theoretically have five instructions in the five pipeline stages at once (one instruction per stage), a different instruction would complete stage 5 in every clock cycle	C204.3	BTL1

UNIT V MEMORY AND I/O 9

Memory Concepts and Hierarchy – Memory Management – Cache Memories: Mapping and Replacement Techniques – Virtual Memory – DMA – I/O – Accessing I/O: Parallel and Serial Interface – Interrupt I/O – Interconnection Standards: USB, SATA

UNIT V MEMORY & I/O SYSTEMS 9

Memory Concepts and Hierarchy – Memory Management – Cache Memories: Mapping and Replacement Techniques – Virtual Memory – DMA – I/O – Accessing I/O: Parallel and Serial Interface – Interrupt I/O – Interconnection Standards: USB, SATA

.

PART -A

Q. No.	Questions	CO	Bloom's Level
1.	**What is principle of locality?** The principle of locality states that programs access a relatively small	C204.5	BTL1

	portion of their address space at any instant of time		
2.	**Define spatial locality.** The locality principle stating that if a data location is referenced, data locations with nearby addresses will tend to be referenced soon.	C204.5	BTL1
3.	**Define Memory Hierarchy.(MAY/JUNE 2016)** A structure that uses multiple levels of memory with different speeds and sizes. The faster memories are more expensive per bit than the slower memories.	C204.5	BTL1
4.	**Define hit ratio. (A.U.APR/MAY 2013,NOV/DEC 2015)** When a processor refers a data item from a cache, if the referenced item is in the cache, then such a reference is called Hit. If the referenced data is not in the cache, then it is called Miss, Hit ratio is defined as the ratio of number of Hits to number of references. Hit ratio =Total Number of references	C204.5	BTL1
5	**What is TLB? What is its significance?** Translation look aside buffer is a small cache incorporated in memory management unit. It consists of page table entries that correspond to most recently accessed pages. Significance The TLB enables faster address computing. It contains 64 to 256 entries	C204.5	BTL1
6	**Define temporal locality.** The principle stating that a data location is referenced then it will tend to be referenced again soon. .	C204.5	BTL6

7	**How cache memory is used to reduce the execution time. (APR/MAY'10)** If active portions of the program and data are placed in a fast small memory, the average memory access time can be reduced, thus reducing the total execution time of the program. Such a fast small memory is called as cache memory.	C204.5	BTL6
8	**Define memory interleaving. (A.U.MAY/JUNE '11) (apr/may2017)** In order to carry out two or more simultaneous access to memory, the memory must be partitioned in to separate modules. The advantage of a modular memory is that it allows the interleaving i.e. consecutive addresses are assigned to different memory module	C204.5	BTL1
9	**Define Hit and Miss? (DEC 2013)** The performance of cache memory is frequently measured in terms of a quantity called hit ratio. When the CPU refers to memory and finds the word in cache, it is said to produce a hit. If the word is not found in cache, then it is in main memory and it counts as a miss	C204.5	BTL1
10	**What is cache memory?NOV/DEC 2016** It is a fast memory that is inserted between the larger slower main memory and the processor. It holds the currently active segments of a program and their data	C204.5	BTL1
11	**What is memory system? [MAY/JUNE '11] [APR/MAY 2012]** Every computer contains several types of devices to store the instructions and data required for its operation. These storage devices plus the algorithm-implemented by hardware and/or software-needed to manage the stored information from the memory system of computer	C204.5	BTL1
12	**What is Read Access Time? [APR/MAY 2012]** A basic performance measure is the average time to read a fixed amount of information, for instance, one word, from the memory. This parameter is called the read access time	C204.5	BTL1

13	**What is the necessary of virtual memory? State the advantages of virtual memory? MAY/JUNE 2016** Virtual memory is an important concept related to memory management. It is used to increase the apparent size of main memory at a very low cost. Data are addressed in a virtual address space that can be as large as the addressing capability of CPU. Virtual memory is a technique that uses main memory as a “cache” for secondary storage. Two major motivations for virtual memory: to allow efficient and safe sharing of memory among multiple programs, and to remove the programming burdens of a small, limited amount of main memory	C204.5	BTL1
14	**What are the units of an interface? (Dec 2012)** DATAIN, DATAOUT, SIN, SOUT	C204.5	BTL1
15	**Distinguish between isolated and memory mapped I/O? (May 2013)** The **isolated I/O** method isolates memory and I/O addresses so that memory address values are not affected by interface address assignment since each has its own address space. In **memory mapped I/O**, there are no specific input or output instructions. The CPU can manipulate I/O data residing in interface registers with the same instructions that are used to manipulate memory words	C204.5	BTL1
16	**Distinguish between memory mapped I/O and I/O mapped I/O. Memory mapped I/O:** When I/O devices and the memory share the same address space, the arrangement is called memory mapped I/O. The machine instructions that can access memory is used to trfer data to or from an I/O device. **I/O mapped I/O:** Here the I/O devices the memories have different address space. It has special I/O instructions. The advantage of a separate I/O address space is that I/O devices deals with fewer address lines.	C204.5	BTL1

17	**Define virtual memory.(nov/dec 2017)** The data is to be stored in physical memory locations that have addresses different from those specified by the program. The memory control circuitry translates the address specified by the program into an address that can be used to access the physical memory	C204.5	BTL1
18	**What is Semi Random Access?** Memory devices such as magnetic hard disks and CD-ROMs contain many rotating storage tracks. If each track has its own read write head, the tracks can be accessed randomly, but access within each track is serial. In such cases the access mode is semi random.	C204.5	BTL1
19	**What is the use of DMA? (Dec 2012)(Dec 2013,APR/MAY2018)** DMA (Direct Memory Access) provides I/O transfer of data directly to and from the memory unit and the peripheral**.**	C204.5	BTL1
20	**Mention the advantages of USB. (May 2013)** The Universal Serial Bus (USB) is an industry standard developed to provide two speed of operation called low-speed and full-speed. They provide simple, low cost and easy to use interconnect system.	C204.5	BTL1
21	**What is meant by vectored interrupt?(Dec 2013)** Vectored Interrupts are type of I/O interrupts in which the device that generates the interruptrequest (also called IRQ in some text books) identifies itself directly to the processor	C204.5	BTL1

22	**Compare Static RAM and Dynamic RAM.(Dec 2013,APR/MAY2018)** Static RAM is more expensive, requires four times the amount of space for a given amount of data than dynamic RAM, but, unlike dynamic RAM, does not need to be power-refreshed and is therefore faster to access. Dynamic RAM uses a kind of capacitor that needs frequent power refreshing to retain its charge. Because reading a DRAM discharges its contents, a power refresh is required after each read. Apart from reading, just to maintain the charge that holds its content in place, DRAM must be refreshed about every 15 microseconds. DRAM is the least expensive kind of RAM. SRAMs are simply integrated circuits that are memory arrays with a single access port that can provide either a read or a write. SRAMs have a fixed access time to any datum. SRAMs don't need to refresh and so the access time is very close to the cycle time. SRAMs typically use six to eight transistors per bit to prevent the information from being disturbed when read. SRAM needs only minimal power to retain the charge in standby mode. In a dynamic RAM (DRAM), the value kept in a cell is stored as a charge in a capacitor. A single transistor is then used to access this stored charge, either to read the value or to overwrite the charge stored there. Because DRAMs use only a single transistor per bit of storage, they are much denser and cheaper per bit than SRAM DRAMs store the charge on a capacitor, it cannot be kept indefinitely and must periodically be refreshed.	C204.5	BTL1
23	**what is DMA ?(NOV/DEC 2014)** Direct memory access (DMA) is a method that allows an input/output (I/O) device to send or receive data directly to or from the main memory, bypassing the CPU to speed up memory operations. The process is managed by a chip known as a DMA controller (DMAC).	C204.5	BTL1

<table>
<tr>
<td>24</td>
<td>

Differentiate programmed I/O and interrupt i/O..(NOV/DEC2014)

<table>
<tr><th>programmed I/O</th><th>interrupt i/O</th></tr>
<tr><td>Programmed IO is the process of IO instruction written in computer program</td><td>Interrupt Initiated IO is done by using interrupt and some special command.</td></tr>
<tr><td>In Programmed IO technique to transfer data,required constant motoring on peripheral by CPU,once data transfer is initiated, CPU have to wait for next transfer.</td><td>In Interrupt Initiated IO once data transfer initiated ,CPU execute next program without wasting time and the interface keep monitoring the device.</td></tr>
</table>

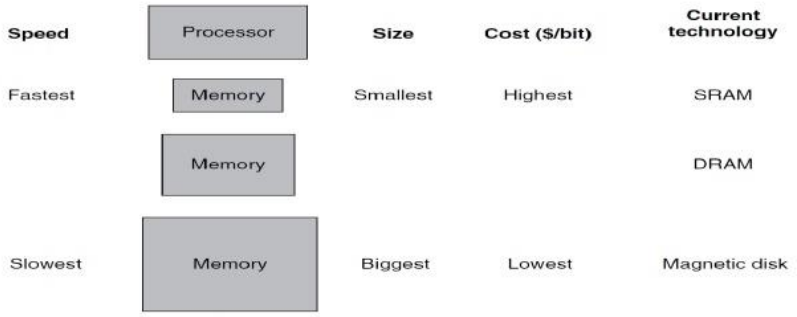
</td>
<td>C204.5</td>
<td>BTL1</td>
</tr>
<tr>
<td>25</td>
<td>

what is the purpose of dirty /modified bit in cache memory.(NOV/DEC2014)

A dirty bit or modified bit is a bit that is associated with a block of computer memory and indicates whether or not the corresponding block of memory has been modified.[1] The dirty bit is set when the processor writes to (modifies) this memory. The bit indicates that its associated block of memory has been modified and has not yet been saved to storage.

</td>
<td>C204.5</td>
<td>BTL1</td>
</tr>
<tr>
<td>26</td>
<td>

What is the need to implement memory as a hierarchy? (APRIL/MAY2015)



The basic structure of a memory hierarchy.

</td>
<td>C204.5</td>
<td>BTL1</td>
</tr>
<tr>
<td>27</td>
<td>

Point out how DMA can improve I/O speed? APRIL/MAY 2015

CPU speeds continue to increase, and new CPUs have multiple processing elements on the same chip.A large amount of data can be processed very quickly Problem in the transfer of data to CPU or even memory in a reasonable amount of time so that CPU has some work to do at all time . Without DMA, when the CPU is using programmed input/output, it is typically fully occupied for the entire

</td>
<td>C204.5</td>
<td>BTL1</td>
</tr>
</table>

<table>
<tr><td></td><td>duration of the read or write operation, and is thus unavailable to perform other work. With DMA, the CPU first initiates the transfer, then it does other operations while the transfer is in progress, and it finally receives an interrupt from the DMA controller when the operation is done.</td><td></td><td></td></tr>
<tr><td>28</td><td>What are the various memory Technologies?NOV/DEC 2015

Memory Technologies

Main memory is implemented from DRAM (dynamic random access memory), while levels closer to the processor (caches) use SRAM (static random access memory). DRAM is less costly per bit than SRAM, although it is substantially slower. The price difference arises because DRAM uses significantly less area per bit of memory, and DRAMs thus have larger capacity for the same amount of silicon;

<table><tr><th>Memory technology</th><th>Typical access time</th><th>$ per GiB in 201</th></tr><tr><td>SRAM semiconductor memory</td><td>0.5–2.5 ns</td><td>$500–$1000</td></tr><tr><td>DRAM semiconductor memory</td><td>50–70 ns</td><td>$10–$20</td></tr><tr><td>Flash semiconductor memory</td><td>5,000–50,000 ns</td><td>$0.75–$1.00</td></tr><tr><td>Magnetic disk</td><td>5,000,000–20,000,000 ns</td><td>$0.05–$0</td></tr></table></td><td>C204.5</td><td>BTL1</td></tr>
<tr><td>29</td><td>What is flash memory?

Flash memory is a type of electrically erasable programmable read-only memory (EEPROM). Unlike disks and DRAM, EEPROM technologies can wear out flash memory bits. To cope with such limits, most flash products include a controller to spread the writes by remapping blocks that have been written many times to less trodden blocks. This technique is called wear leveling.</td><td>C204.5</td><td>BTL3</td></tr>
</table>

30	**In many computers the cache block size is in the range 32 to 128 bytes. What would be the main Advantages and disadvantages of making the size of the cache blocks larger or smaller?** Larger the size of the cache fewer be the cache misses if most of the data in the block are actually used. It will be wasteful if much of the data are not used before the cache block is moved from cache. Smaller size means more misses	C204.5	BTL1
31	**Define USB.** Universal Serial Bus, an external busstandard that supports data transfer ratesof 12 Mbps. A single USB portcan be used to connect up to 127 peripheral devices, such as mice, modems, and keyboards. USB also supportsPlug-and-Play installationandhot plugging. .	C204.5	BTL1
32	**Define Memory latency** **The amount of time it takes to transfer a word of data to or from the memory.**	C204.5	BTL1
33	**Define Memory bandwidth** **Tthe number of bits or bytes that can be transferred in one second. It is used to measure how much time is needed to transfer an entire block of data.**	C204.5	BTL1
34	Define miss Rate. The miss rate (1−hit rate) is the fraction of memory accesses not found in the upper level.	C204.5	BTL1
35	Define Hit rate. Hit rate◊ The fraction of memory accesses found in a level of the memory hierarchy. •	C204.5	BTL1
36	Define miss rate. Miss rate◊ The fraction of memory accesses not found in a level of the memory hierarchy.	C204.5	BTL1

37	Define Hit time. Hit time is the time to access the upper level of the memory hierarchy, which includes the time needed to determine whether the access is a hit or a miss	C204.5	BTL1
38	Define miss penalty The miss penalty is the time to replace a block in the upper level with the corresponding block from the lower level, plus the time to deliver this block to the processor	C204.5	BTL1
39	Define tag in TLB Tag◊ A field in a table used for a memory hierarchy that contains the address information required to identify whether the associated block in the hierarchy corresponds to a requested word.	C204.5	BTL1
40	What are the steps to be taken on an instruction cache miss: 1. Send the original PC value (current PC – 4) to the memory. 2. Instruct main memory to perform a read and wait for the memory to complete its access. 3. Write the cache entry, putting the data from memory in the data portion of the entry, writing the upper bits of the address (from the ALU) into the tag field, and turning the valid bit on. 4. Restart the instruction execution at the first step, which will refetch the instruction, this time finding it in the cache	C204.5	BTL1
41	What is write through cache The simplest way to keep the main memory and the cache consistent is always to write the data into both the memory and the cache. • This scheme is called write-through.	C204.5	BTL1
42	What is write back cache In a write back scheme, when a write occurs, the new value is written only to the block in the cache.	C204.5	BTL1

43	What are the techniques to improve cache performance? Two different techniques for improving cache performance. • One focuses on reducing the miss rate by reducing the probability that two different memory blocks will participate for the same cache location. • The second technique reduces the miss penalty by adding an additional level to the hierarchy. This technique, called multilevel caching	C204.5	BTL1
44	Define dirty bit dirty bit is commonly used. This status bit indicates whether the block is dirty (modified while in the cache) or clean (not modified).	C204.5	BTL1
45	What is TLB. Translation-lookaside buffer (TLB)◊A cache that keeps track of recently used address mappings to try to avoid an access to the page table.	C204.5	BTL1
46	What are the messages transferred in DMA? To initiate the transfer of a block of words , the processor sends, i) Starting address ii) Number of words in the block iii)Direction of transfer.	C204.5	BTL1
47	Define Burst mode. Burst Mode: The DMA controller may be given exclusive(limited) access to the main memory to transfer a block of data without interruption. This is known as Burst/Block Mode. •	C204.5	BTL1
48	Define bus master Bus Master: The device that is allowed to initiate data transfers on the bus at any given time is called the bus master	C204.5	BTL1
49	Define bus arbitration. Bus Arbitration: It is the process by which the next device to become the bus master is selected and the bus mastership is transferred to it.	C204.5	BTL1

50	What are the approaches for bus arbitration? There are 2 approaches to bus arbitration. They are i)Centralized arbitration (A single bus arbiter performs arbitration) ii)Distributed arbitration (all devices participate in the selection of next bus master).	C204.5	BTL1

PART -B

Q. No.	Questions	CO	Bloom's Level
1.	Explain in detail about memory Technologies**(APRIL/MAY2015,NOV/DEC2017) (*Page.No:*-378-383**)	C204.5	BTL5
2.	Expain in detail about memory Hierarchy with neat diagram **(*Page.No:*-374-378**)	C204.5	BTL5
3.	Discuss the various mapping schemes used in cache memory(**NOV/DEC2014) (*Page.No:*-383-397**)	C204.5	BTL6
4.	Discuss the methods used to measure and improve the performance of the cache**.(NOV/DEC 2017) (*Page.No:*-398-417**)	C204.5	BTL6
5.	Explain the virtual memory address translation and TLB with necessary diagram**.(APRIL/MAY2015,NOV/DEC 2015,NOV/DEC 2016,APR/MAY2018) (*Page.No:*-427-452**)	C204.5	BTL5
6.	Draw the typical block diagram of a DMA controller and explain how it is used for direct data transfer between memory and peripherals. **(NOV/DEC 2015,MAY/JUNE 2016,NOV/DEC 2016,MAY/JUN 2018) *Page.No:*-399-402**)	C204.5	BTL5
7.	Explain in detail about interrupts with diagram ***(Page.No:*-436-242**)	C204.5	BTL5

8.	Describe in detail about programmed Input/Output with neat diagram **(MAY/JUN 2018) (Refer notes)**	C204.5	BTL5
9.	Explain in detail about the bus arbitration techniques**.(NOV/DEC2014)**(8) ***(Page.No:*-237-242**)	C204.5	BTL5
10.	Draw different memory address layouts and brief about the technique used to increase the average rate of fetching words from the main memory (8)(**NOV/DEC2014)** **(Refer notes)**	C204.5	BTL5
11.	Explain in detail about any two standard input and output interfaces required to connect the I/O devices to the bus.(**NOV/DEC2014)** ***(Page.No:*-438-452**)	C204.5	BTL5
12.	Explain mapping functions in cache memory in cache memory to determine how memory blocks are placed in cache **(Nov/Dec 2014) (Refer notes)**	C204.5	BTL5
13.	Explain the various mapping techniques associated with cache memories **(MAY/JUNE 2016,MAY/JUN 2018)** **(Refer notes)**	C204.5	BTL5
14.	Explain sequence of operations carried on by a processor when interrupted by a peripheral device connected to it**(MAY/JUN 2018) *(Page.No:*-436-242**)	C204.5	BTL5
15.	Explain virtual memory and the advantages of using virtual memory ***(Page.No:*-427-252**)	C204.5	BTL5

PART-B

Q. No.	Questions	CO	Bloom's Level
1. .	Explain the basic MIPS implementation with binary multiplexers and control lines(16) **NOV/DEC 15 (*Page.No:*244-251**)	C204.3	BTL5
2.	What is hazards ?Explain the different types of pipeline hazards with suitable examples**.(NOV/DEC2014,APRIL/MAY2015,MAY/JUNE 2016,NOV/DEC2017) (*Page.No:303*-324**)	C204.3	BTL5
3.	Explain how the instruction pipeline works. What are the various situations where an instruction pipeline can stall? Illustration with an example? **NOV/DEC 2015,NOV/DEC 2016.(*Page.No:301*-302**)	C204.3	BTL5
4.	Explain data path in detail**(NOV/DEC 14,NOV/DEC2017) (*Page.No:251*-259**)	C204.3	BTL5
5.	Explain dynamic branch prediction **.(*Page.No:321*-323**)	C204.3	BTL5
6.	Explain in detail How exceptions are handled in MIPS architecture.**(APRIL/MAY2015) .(*Page.No:*325-332**)	C204.3	BTL5
7.	Explain in detail about building a datapath(**NOV/DEC2014** **(*Page.No:251*-259**)	C204.3	BTL5
8.	Explain in detail about control implementation scheme(**APR/MAY 2018)** **(*Page.No:***259-271)	C204.3	BTL5

9.	What is pipelining?Discuss about pipelined datapath and control(16)**MAY/JUNE2016 (*Page.No* :**286-303)	C204.3	BTL6
10.	Why is branch prediction algorithm needed?Differentiate between static and dynamic techniques?**NOV/DEC 2016 .(*Page.No:321*-323**)	C204.3	BTL3
11.	Design a simple path with control implementation and explain in detail(**MAY/JUN 2018) (*Page.No:251*-271**)	C204.3	BTL6
12.	Discuss the limitation in implementing the processor path. Suggest the methods to overcome them**(NOV/DEC 2018) (Refer notes)**	C204.3	BTL6
13.	When processor designers consider a possible improvement to the processor datapath, the decision usually depends on the cost/performance trade-off . In the following three problems, assume that we are starting with a datapath where I-Mem, Add, Mux, ALU, Regs, D-Mem, and Control blocks have latencies of 400 ps, 100 ps, 30 ps, 120 ps, 200 ps, 350 ps, and 100 ps, respectively,and costs of 1000, 30, 10, 100, 200, 2000, and 500, respectively.Consider the addition of a multiplier to the ALU. Th is addition will add 300 ps to the latency of the ALU and will add a cost of 600 to the ALU. Th e result will be 5% fewer instructions executed since we will no longer need to emulate the MUL instruction. 1 What is the clock cycle time with and without this improvement? 2 What is the speedup achieved by adding this improvement? 3 Compare the cost/performance ratio with and without this improvement. **(Refer notes)**	C204.3	BTL5
14.	For the problems in this exercise, assume that there are no pipeline stalls and that the breakdown of executed instructions is as follows: add addi not beq lw sw 20% 20% 0% 25% 25% 10% 14.1 In what fraction of all cycles is the data memory used?	C204.3	BTL3

	14.2 In what fraction of all cycles is the input of the sign-extend circuit needed? What is this circuit doing in cycles in which its input is not needed?		
15.	Consider the following loop. loop:lw r1,0(r1) and r1,r1,r2 lw r1,0(r1) lw r1,0(r1) beq r1,r0,loop Assume that perfect branch prediction is used (no stalls due to control hazards),that there are no delay slots, and that the pipeline has full forwarding support. Also assume that many iterations of this loop are executed before the loop exits.	C204.3	BTL3