COEP TECHNOLOGICAL UNIVERSITY (COEP Tech)

A Unitary Public University of Government of Maharashtra
(Formerly College of Engineering Pune (COEP))

END Semester Examination

Programme: B. Tech. **Semester: V**

Course Code: CT 21007 **Course Name: Computer Organization**

Branch: Computer Engineering **Academic Year: 2024-25**

Duration: 3 Hrs. **Max Marks: 60**

Student PRN No. □□□□□□□□□

Instructions:

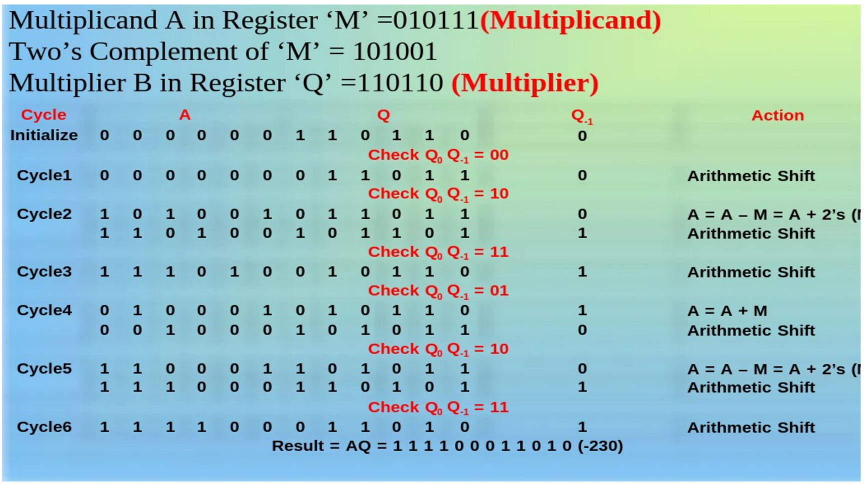
1. **Figures to the right indicate the full marks.**
2. **Mobile phones and programmable calculators are strictly prohibited.**
3. **Writing anything on question paper is not allowed.**
4. **Exchange/Sharing of stationery, calculator etc. not allowed.**
5. **Write your PRN Number on Question Paper.**

			Marks	CO	PO
Q 1		**Solve the following.**			
	a	1. 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? $\text{Clock Rate}_B = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B} = \frac{1.2 \times \text{Clock Cycles}_A}{6\text{s}}$ $\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$ $= 10\text{s} \times 2\text{GHz} = 20 \times 10^9$ $\text{Clock Rate}_B = \frac{1.2 \times 20 \times 10^9}{6\text{s}} = \frac{24 \times 10^9}{6\text{s}} = 4\text{GHz}$ 2. When parallelizing an application, the ideal speedup is speeding up by the number of processors. This is limited by two things: percentage of the application that can be parallelized and the cost of communication. Amdahl's law takes into account the former but not the latter. a. Given that 40% of the first application is parallelizable, how much *overall* system *speedup* would you observe if you parallelized it? b. Given that 99% of the second application is parallelizable, how much *overall* system speedup would you observe if you parallelized it? **a. 1/(0.2 + 0.8 × 0.6 + 0.8 × 0.4/2) = 1/(.2 + .48 + .16) = 1.19** **b. 1/(0.8 + 0.2 × .01 + 0.2 × 0.99/2) = 1/(0.8 + 0.002 + 0.099) = 1.11**	4 (2*2)	1	1, 2, 4, 5, 8, 10, 12

b

1. Multiply each of the following pairs of signed 2's-complement numbers using the Booth's algorithm. In each case, assume that A is the multiplicand and B is the multiplier.

(a) A = 010111 and B = 110110



Multiplicand A in Register 'M' =010111**(Multiplicand)**
Two's Complement of 'M' = 101001
Multiplier B in Register 'Q' =110110 **(Multiplier)**

Cycle	A	Q	Q_{-1}	Action
Initialize	0 0 0 0 0 0	1 1 0 1 1 0	0	
		Check Q_0 Q_{-1} = 00		
Cycle1	0 0 0 0 0 0	0 1 1 0 1 1	0	Arithmetic Shift
		Check Q_0 Q_{-1} = 10		
Cycle2	1 0 1 0 0 1	0 1 1 0 1 1	0	A = A – M = A + 2's (I
	1 1 0 1 0 0	1 0 1 1 0 1	1	Arithmetic Shift
		Check Q_0 Q_{-1} = 11		
Cycle3	1 1 1 0 1 0	0 1 0 1 1 0	1	Arithmetic Shift
		Check Q_0 Q_{-1} = 01		
Cycle4	0 1 0 0 0 1	0 1 0 1 1 0	1	A = A + M
	0 0 1 0 0 0	1 0 1 0 1 1	0	Arithmetic Shift
		Check Q_0 Q_{-1} = 10		
Cycle5	1 1 0 0 0 1	1 0 1 0 1 1	0	A = A – M = A + 2's (I
	1 1 1 0 0 0	1 1 0 1 0 1	1	Arithmetic Shift
		Check Q_0 Q_{-1} = 11		
Cycle6	1 1 1 1 0 0	0 1 1 0 1 0	1	Arithmetic Shift

Result = AQ = 1 1 1 1 0 0 0 1 1 0 1 0 (-230)

2. Repeat the above problem using bit-pair recoding of the multiplier.

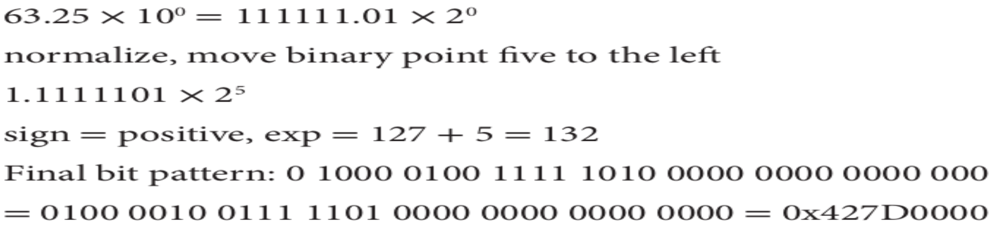
4

(2*2)

2

c

1. Write down the binary representation of the decimal number 63.25 assuming the IEEE 754 single precision format.



$63.25 \times 10^0 = 111111.01 \times 2^0$

normalize, move binary point five to the left

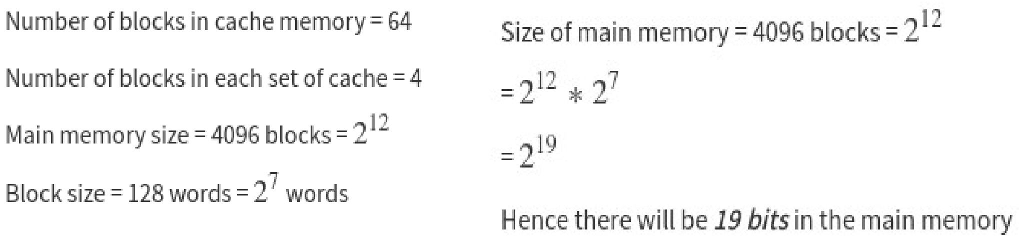
1.1111101×2^5

sign = positive, exp = 127 + 5 = 132

Final bit pattern: 0 1000 0100 1111 1010 0000 0000 0000 000

= 0100 0010 0111 1101 0000 0000 0000 0000 = 0x427D0000

2. A block-set-associative cache consists of a total of 64 blocks, divided into 4-block sets. The main memory contains 4096 blocks, each consisting of 32 words. Assuming a 32-bit byte-addressable address space, how many bits are there in each of the Tag, Set, and Word offset fields?



Number of blocks in cache memory = 64

Number of blocks in each set of cache = 4

Main memory size = 4096 blocks = 2^{12}

Block size = 128 words = 2^7 words

Size of main memory = 4096 blocks = 2^{12}

$= 2^{12} * 2^7$

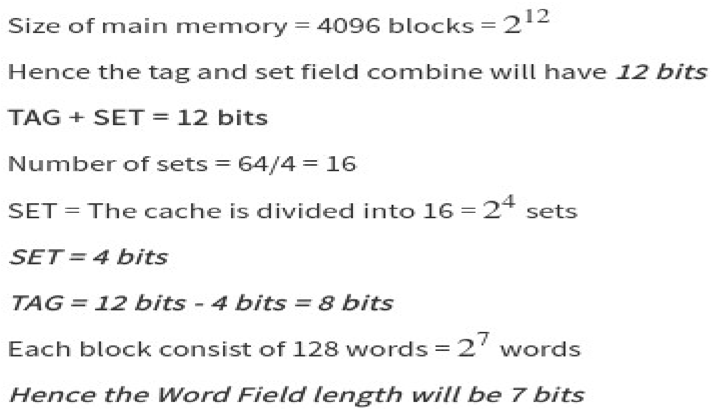
$= 2^{19}$

Hence there will be *19 bits* in the main memory

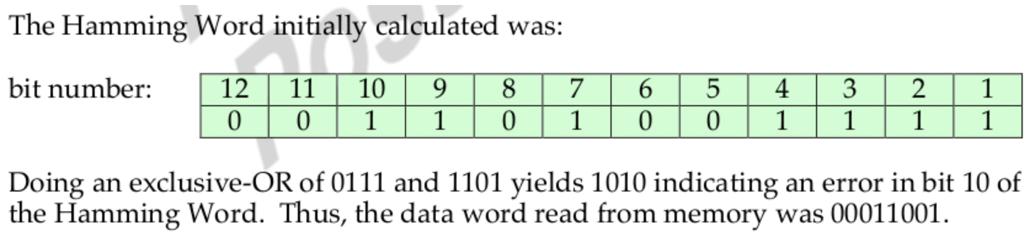
4

(2*2)

2, 3

<table>
<tr><td></td><td></td><td>
Size of main memory = 4096 blocks = 2^{12}
Hence the tag and set field combine will have *12 bits*
TAG + SET = 12 bits
Number of sets = 64/4 = 16
SET = The cache is divided into 16 = 2^4 sets
SET = 4 bits
TAG = 12 bits - 4 bits = 8 bits
Each block consist of 128 words = 2^7 words
Hence the Word Field length will be 7 bits</td><td></td><td></td><td></td></tr>
<tr><td></td><td><b>d</b></td><td>1. A snapshot of the state associated with 2 caches, on 2 separate cores, in a centralized shared memory system is shown below. In this system, cache coherency is maintained with an <i>MSI</i> snooping protocol. You can assume that the caches are direct mapped.
<b>P0:</b>
<table>
<tr><th></th><th>Tag</th><th>Data Word 1</th><th>Data Word 2</th><th>Data Word 3</th><th>Data Word 4</th><th>Coherency State</th></tr>
<tr><td>Block 0</td><td>1000</td><td>10</td><td>20</td><td>30</td><td>40</td><td>M</td></tr>
<tr><td>Block 1</td><td>4000</td><td>500</td><td>600</td><td>700</td><td>800</td><td>S</td></tr>
<tr><td>…</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Block N</td><td>3000</td><td>2</td><td>4</td><td>6</td><td>8</td><td>S</td></tr>
</table>
<b>P1:</b>
<table>
<tr><th></th><th>Tag</th><th>Data Word 1</th><th>Data Word 2</th><th>Data Word 3</th><th>Data Word 4</th><th>Coherency State</th></tr>
<tr><td>Block 0</td><td>1000</td><td>10</td><td>10</td><td>10</td><td>10</td><td>I</td></tr>
<tr><td>Block 1</td><td>8000</td><td>500</td><td>600</td><td>700</td><td>800</td><td>S</td></tr>
<tr><td>…</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Block N</td><td>3000</td><td>2</td><td>4</td><td>6</td><td>8</td><td>S</td></tr>
</table>
i. If P1 writes to Block 1, is Block 1 on P0 invalidated? Why or why not?
<b>No. The tags are different so this is different data.</b>
ii. If P1 brings in Block Z for reading, and no other cache has a copy of Z, which state is it cached in?
<b>It would still be cached in the shared state</b>

2. Assume that you have 4 Gbytes of main memory at your disposal
- 1 Gbyte of the 4 Gbytes has been reserved for process page table storage
- Each page table entry consists of: A physical frame number, a 1 valid bit, a 1 dirty bit, a 1 LRU status bit. Virtual addresses are 32 bits, physical addresses are 26 bits, and the page size is 8 Kbytes. How many process page tables can fit in the 1 Gbyte space?
<b>We can first calculate the number of page table entries associated with each process.</b>
<b>- 8 KB pages implies that the offset associated with a VA/PA is 13 bits (2^{13} = 8KB)</b>
<b>- Thus, the remaining 19 bits are used to represent VPNs and serve as indices to the PT</b>
<b>- Thus, each PT has 2^{19} entries.</b>
<b>Next, each PA is 26 bits:</b>
<b>- 13 bits of the PA come from the offset, the other 13 come from a PT lookup</b>
<b>- Given that each PT entry consists of a PFN, a valid bit, a dirty bit, and a LRU bit, each PT entry will be 2 bytes</b>
<b>Therefore:</b>
<b># of page tables = 2^{30} bytes available x (1 PT entry / 2 bytes) x (1 process / 2^{19} PT entries) # of page tables = 2^{10} or 1024.</b></td><td>4
(2*2)</td><td>3,
4</td><td></td></tr>
</table>

<table>
<tr><td></td><td>e</td><td>1. For the 8-bit word 00111001, the check bits stored with it would be 0111. Suppose when the word is read from memory, the check bits are calculated to be 1101. What is the data word that was read from memory?

The Hamming Word initially calculated was:
<table><tr><td>bit number:</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td></tr><tr><td></td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>
Doing an exclusive-OR of 0111 and 1101 yields 1010 indicating an error in bit 10 of the Hamming Word. Thus, the data word read from memory was 00011001.
2. Compare symmetric multiprocessors with clusters in context of different parameters.
<b>Performance, Availability, Incremental growth, Scaling parameters</b></td><td>4
(2*2)</td><td>5</td><td></td></tr>
<tr><td></td><td>f</td><td>1. Derive an equation to calculate speedup factor by considering a processor without pipeline & a processor with ‘k’ stage pipeline.
$$S_k = \frac{T_{1,n}}{T_{k,n}} = \frac{nk\tau}{[k + (n - 1)]\tau} = \frac{nk}{k + (n - 1)}$$
2. Explain the speedup factor derived in 1 with appropriate example.
<b>Answer sheets example or any other suitable example.</b></td><td>4
(2*2)</td><td>6</td><td></td></tr>
<tr><td>Q 2</td><td>A

B</td><td>Derive an equation for Amdahl’s law.
$$\text{Speedup} = \frac{\text{Execution time for entire task without using the enhancement}}{\text{Execution time for entire task using the enhancement when possible}}$$
$$\text{Execution time}_{\text{new}} = \text{Execution time}_{\text{old}} \times \left((1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}\right)$$
$$\text{Speedup}_{\text{overall}} = \frac{\text{Execution time}_{\text{old}}}{\text{Execution time}_{\text{new}}} = \frac{1}{(1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$
Consider three different processors P1, P2, and P3 executing the same instruction set. P1 has a 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.
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?
<b>a. performance of P1 (instructions/sec) = 3 × 109/1.5 = 2 × 109</b>
<b>performance of P2 (instructions/sec) = 2.5 × 109/1.0 = 2.5 × 109</b>
<b>performance of P3 (instructions/sec) = 4 × 109/2.2 = 1.8 × 109</b>
<b>b. cycles(P1) = 10 × 3 × 109 = 30 × 109 s</b></td><td>6
(3*2)</td><td>1</td><td>1,
2,
4,
5,
8,
10,
12</td></tr>
</table>

cycles(P2) = 10 × 2.5 × 109 = 25 × 109 s

cycles(P3) = 10 × 4 × 109 = 40 × 109 s

c. No. instructions(P1) = 30 × 109/1.5 = 20 × 109

No. instructions(P2) = 25 × 109/1 = 25 × 109

No. instructions(P3) = 40 × 109/2.2 = 18.18 × 109

CPInew = CPIold × 1.2, then CPI(P1) = 1.8, CPI(P2) = 1.2, CPI(P3) = 2.6

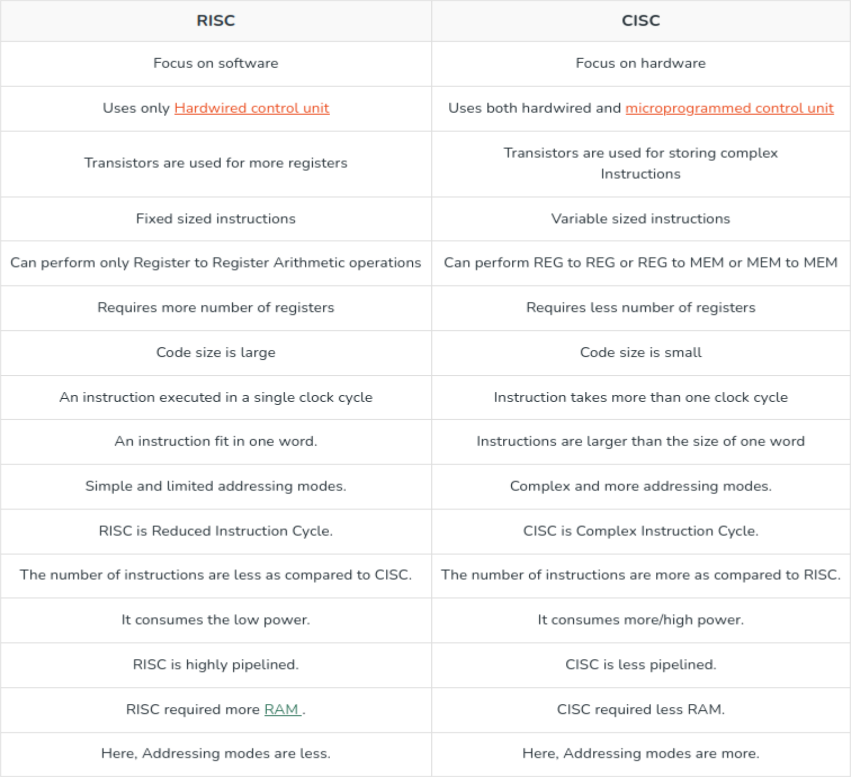
f = No. instr. × CPI/time, then f(P1) = 20 × 109 × 1.8/7 = 5.14 GHz f(P2) = 25 × 109 × 1.2/7 = 4.28 GHz

f(P1) = 18.18 × 109 × 2.6/7 = 6.75 GHz

Q 3

A

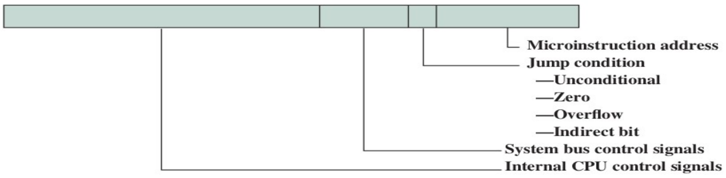
Differentiate between RISC and CISC Styles of processor design.



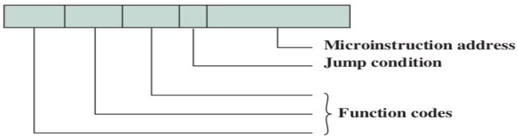
RISC	CISC
Focus on software	Focus on hardware
Uses only Hardwired control unit	Uses both hardwired and microprogrammed control unit
Transistors are used for more registers	Transistors are used for storing complex Instructions
Fixed sized instructions	Variable sized instructions
Can perform only Register to Register Arithmetic operations	Can perform REG to REG or REG to MEM or MEM to MEM
Requires more number of registers	Requires less number of registers
Code size is large	Code size is small
An instruction executed in a single clock cycle	Instruction takes more than one clock cycle
An instruction fit in one word.	Instructions are larger than the size of one word
Simple and limited addressing modes.	Complex and more addressing modes.
RISC is Reduced Instruction Cycle.	CISC is Complex Instruction Cycle.
The number of instructions are less as compared to CISC.	The number of instructions are more as compared to RISC.
It consumes the low power.	It consumes more/high power.
RISC is highly pipelined.	CISC is less pipelined.
RISC required more RAM .	CISC required less RAM.
Here, Addressing modes are less.	Here, Addressing modes are more.

B

What is the overall function of a processor’s control unit? Explain the control unit operations with appropriate block diagram. Differentiate between horizontal and vreticle microinstruction.

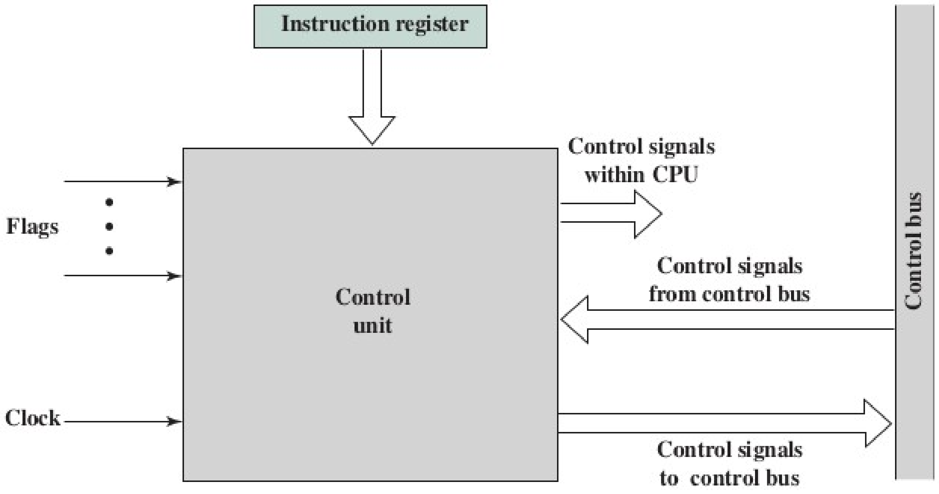
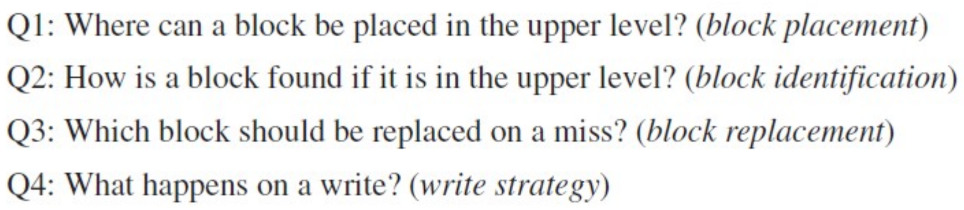


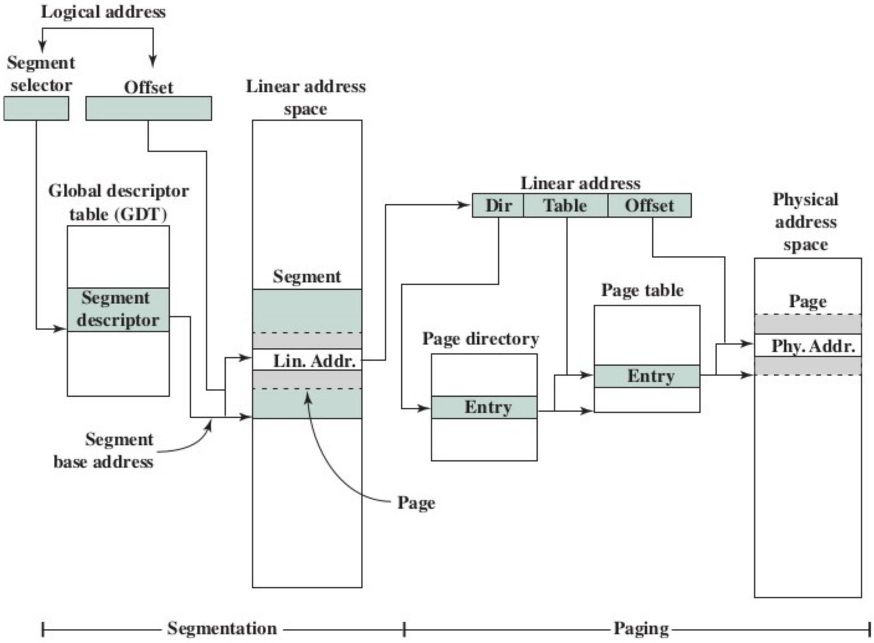
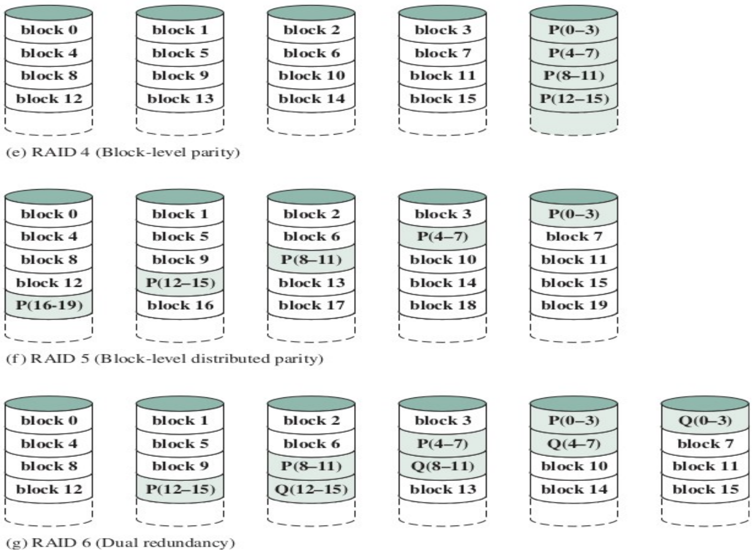
(a) Horizontal microinstruction



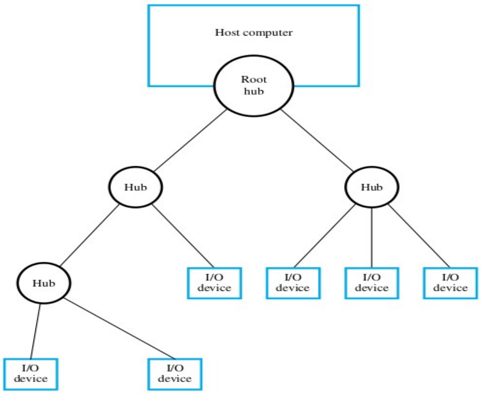
(b) Vertical microinstruction

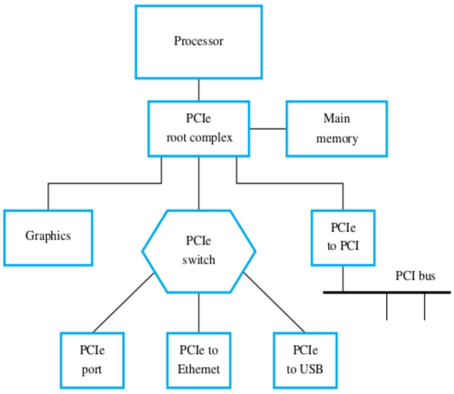
6 (3*2)	2	1, 2, 4, 5, 8, 10, 12

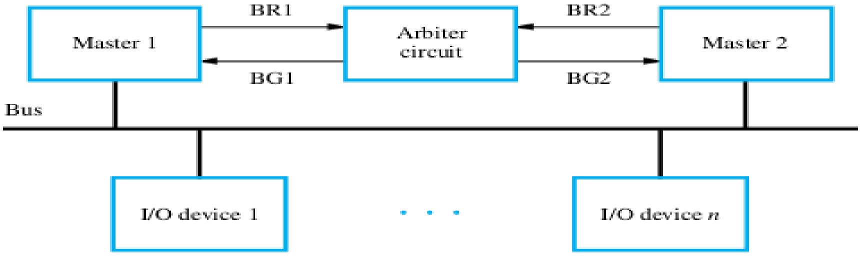
Q 4	A B	 Instruction register Flags Clock Control unit Control signals within CPU Control signals from control bus Control signals to control bus Control bus Discuss the four memory hierarchy questions in context of cache memory.  Q1: Where can a block be placed in the upper level? (*block placement*) Q2: How is a block found if it is in the upper level? (*block identification*) Q3: Which block should be replaced on a miss? (*block replacement*) Q4: What happens on a write? (*write strategy*) For each part of this exercise, assume the initial cache and memory state as illustrated in Figure 1 on Page 4. Each part of this exercise specifies a sequence of one or more CPU operations of the form: **P#: <op> <address> [<value>]** where P# designates the CPU (e.g., P0), <op> is the CPU operation (e.g., read or write), <address> denotes the memory address, and <value> indicates the new word to be assigned on a write operation. **Treat each action among 1 to 6 as independently applied to the initial state as given in Figure 1 on Page 4**. What is the resulting state (i.e., coherence state, tags, and data) of the caches and memory after the given action? Show only the blocks that change; for example, **P0.B0: (I, 120, 00 01)** indicates that CPU P0's block B0 has the final state of I, tag of 120, and data words 00 and 01. Also, what value is returned by each read operation? **Action Performed:** **1.** P0: read 120 **P0.B0: (S, 120, 0020) returns 0020** **2.** P0: write 120 <-- 80 **P0.B0: (M, 120, 0080) & P3.B0: (I, 120, 0020)** **3.** P3: write 120 <-- 80 **P3.B0: (M, 120, 0080)** **4.** P0: write 108 <-- 48 **P0.B1: (M, 108, 0048) & P3.B1: (I, 108, 0008)** **5.** P0: write 130 <-- 78 **P0.B2: (M, 130, 0078) & M: 110 Å 0030 (writeback to memory)** **6.** P3: write 130 <-- 78 **P3.B2: (M, 130, 0078)**	6 (3*2)	3	1, 2, 4, 5, 8, 10, 12

Q 5	**A**	Discuss the working style of paging technique with appropriate block diagrams used in X86 based processors. What is a role of TLB? How many entries are there in different TLBs of Intel Core i7 processor?  Logical address Segment selector Offset Linear address space Global descriptor table (GDT) Segment descriptor Segment Lin. Addr. Segment base address Page Linear address Dir Table Offset Page directory Entry Page table Entry Physical address space Page Phy. Addr. Segmentation Paging Intel Core i7 L1 I 128 entries, 4way SA, L1 D 64 entries 4way & L2 Unified TLB 512 entires 4way SA	6 (3*2)	4	1, 2, 4, 5, 8, 10, 12
	B	Discuss the 4, 5, 6 levels of RAID with appropriate diagrams and differentiate with respect to performance.  block 0 block 4 block 8 block 12 block 1 block 5 block 9 block 13 block 2 block 6 block 10 block 14 block 3 block 7 block 11 block 15 P(0–3) P(4–7) P(8–11) P(12–15) (e) RAID 4 (Block-level parity) block 0 block 4 block 8 block 12 P(16-19) block 1 block 5 block 9 P(12–15) block 16 block 2 block 6 P(8–11) block 13 block 17 block 3 P(4–7) block 10 block 14 block 18 P(0–3) block 7 block 11 block 15 block 19 (f) RAID 5 (Block-level distributed parity) block 0 block 4 block 8 block 12 block 1 block 5 block 9 P(12–15) block 2 block 6 P(8–11) Q(12–15) block 3 P(4–7) Q(8–11) block 13 P(0–3) Q(4–7) block 10 block 14 Q(0–3) block 7 block 11 block 15 (g) RAID 6 (Dual redundancy) Explanation			
Q 6	**A**	Discuss the different types of data dependences. **RAW, WAR, WAW**	6 (3*2)	5	1, 2, 4,

<table>
<tr><td></td><td></td><td>Consider the following code.
L.D F6,32(R2)
L.D F2,44(R3)
MUL.D F0,F2,F4
SUB.D F8,F2,F6
DIV.D F10,F0,F6
ADD.D F6,F8,F2
Explain the stalls occurring due to possible RAW hazards. Discuss the different techniques to eliminate such hazards.
<b>Data forwarding and restructuring code/ re-sequencing of instructions.</b></td><td></td><td></td><td></td></tr>
<tr><td></td><td><b>B</b></td><td>Discuss the following.
1. USB Characteristics
2. PCIE
3. Bus Arbitration

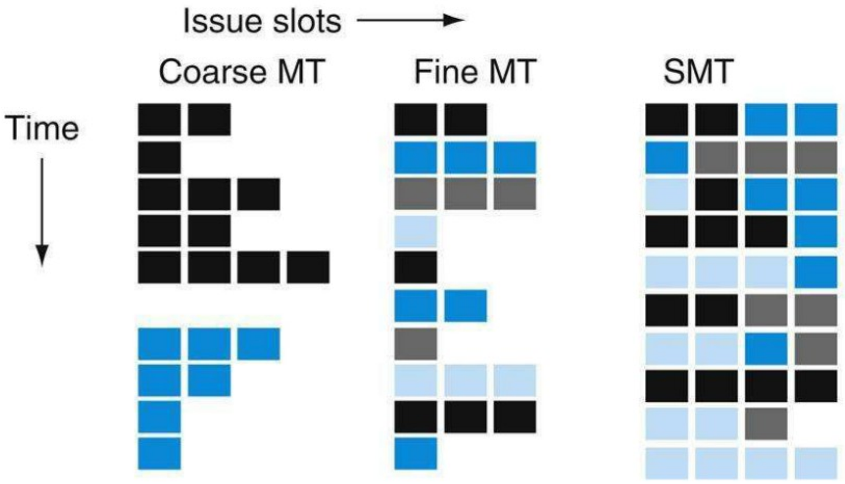




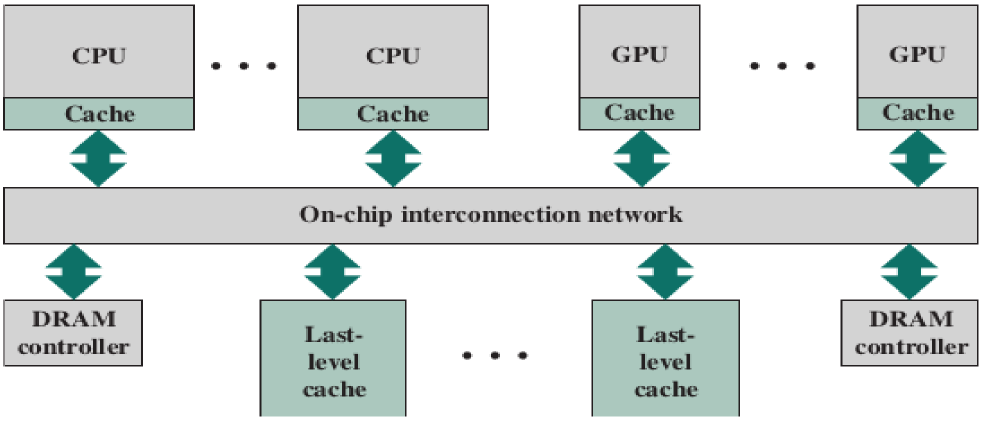


</td><td></td><td></td><td>5,
8,
10,
12</td></tr>
<tr><td><b>Q 7</b></td><td><b>A</b></td><td>Explain different types of hardware multithreading used in different processor architectures. Give appropriate examples.
Without multithreading just superscalar
Coarse G MT
Fine G MT GPU
SMT Intel Core i3, i5, i7

Examples</td><td>6
(3*2)</td><td>6</td><td>1,
2,
4,
5,
8,
10,
12</td></tr>
</table>

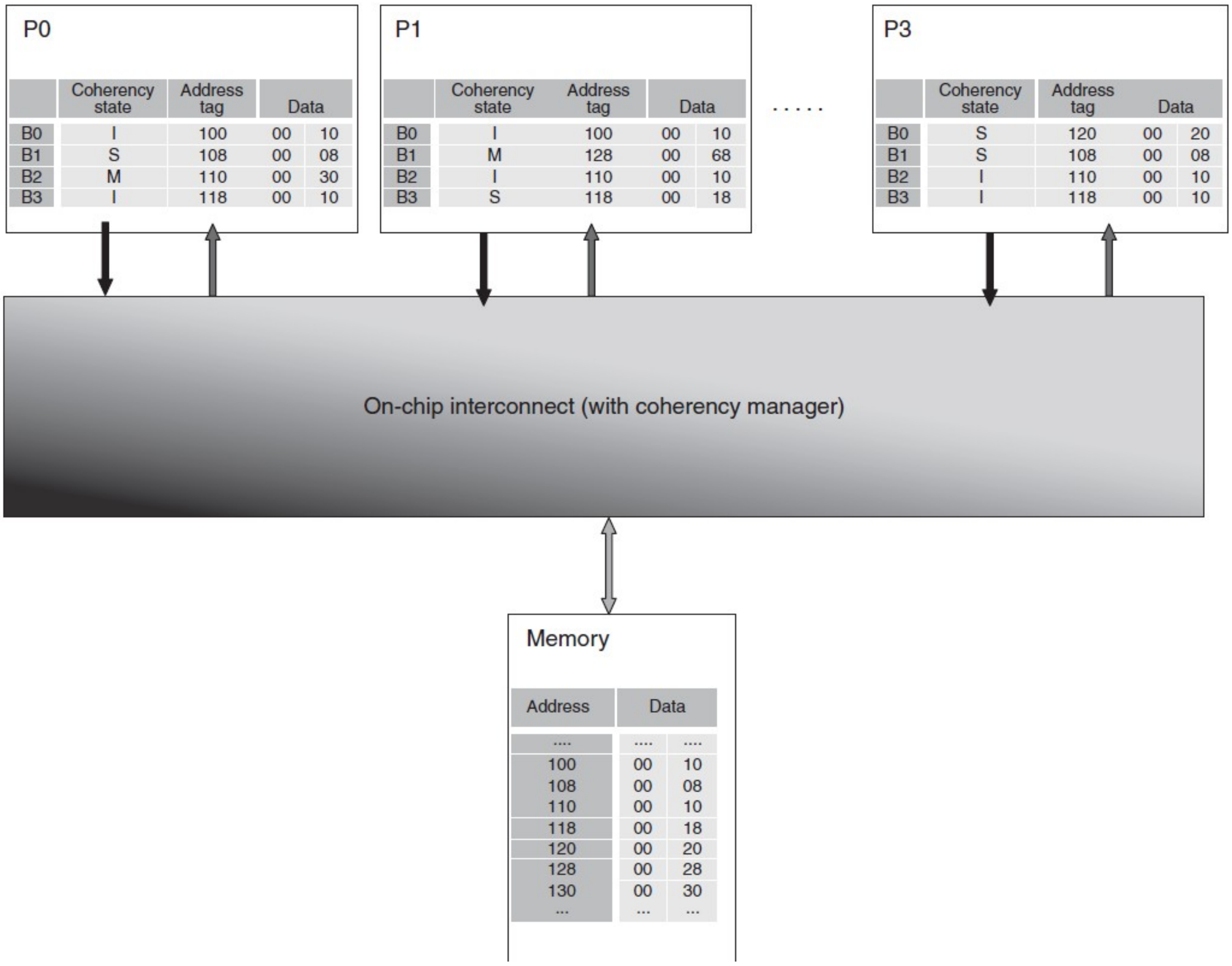


B With an appropriate block diagram, discuss the key features of Heterogeneous System Architecture (HSA).



Key features of the HSA approach include the following:
1. The entire virtual memory space is visible to both CPU and GPU. Both CPU and GPU can access and allocate any location in the system's virtual memory space.
2. The virtual memory system brings in pages to physical main memory as needed.
3. A coherent memory policy ensures that CPU and GPU caches both see an up-to-date view of data.
4. A unified programming interface that enables users to exploit the parallel capabilities of the GPUs within programs that rely on CPU execution as well.

Figure 1 for Q 4 B



P0

	Coherency state	Address tag	Data	
B0	I	100	00	10
B1	S	108	00	08
B2	M	110	00	30
B3	I	118	00	10

P1

	Coherency state	Address tag	Data	
B0	I	100	00	10
B1	M	128	00	68
B2	I	110	00	10
B3	S	118	00	18

P3

	Coherency state	Address tag	Data	
B0	S	120	00	20
B1	S	108	00	08
B2	I	110	00	10
B3	I	118	00	10

Memory

Address	Data	
....	....	....
100	00	10
108	00	08
110	00	10
118	00	18
120	00	20
128	00	28
130	00	30
...	...	...