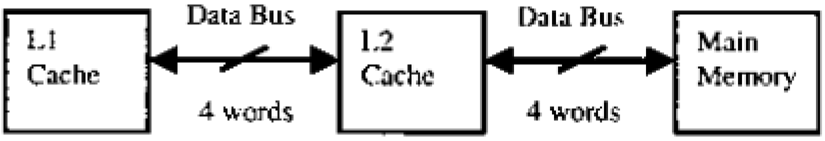
CMR INSTITUTE OF TECHNOLOGY	USN □□□□□□□□□□	

Second Internal Test

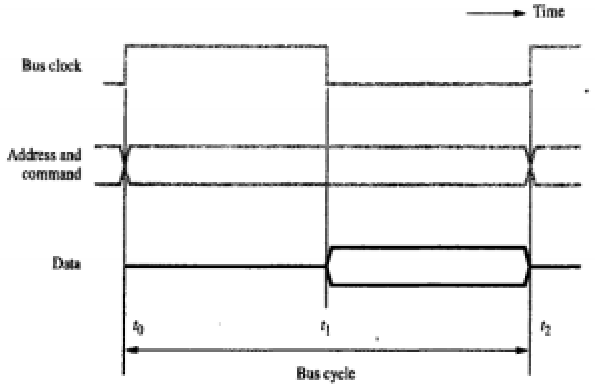
Sub:	COMPUTER ORGANIZATION				Code:	18CS34
Scheme and solution		Sem:	III	Branch:		**ISE**

	Answer Any **FIVE FULL** Questions from Questions 1 to 6	Marks	OBE	
			CO	RBT
1	Discuss in detail about the following bus types with timing diagram i) Synchronous Bus ii) Asynchronous Bus Synchronous Bus: Timing Diagram(2) Explanation(3) Asynchronous Bus: Timing Diagram(2) Explanation(3)	[10]	CO2	L3
2	Three devices, A, B, and C, are connected to the bus of a computer. I/O transfers for all three devices use interrupt control. Interrupt nesting for devices A and B is not allowed, but interrupt requests from C may be accepted while either A or B is being serviced. Suggest different ways in which this can be accomplished in each of the following cases: (a) The computer has one interrupt-request line. (b) Two interrupt-request lines, INTR1 and INTR2, are available, with INTR1 having higher priority. Specify when and how interrupts are enabled and disabled in each case. Case (a):5 marks for explanation Case (b):5 marks for explanation	[10]	CO2	L4
3(a)	Explain about PCI bus Data transfer signals along with its timing diagram. Data signals-3marks Timing diagram:2 marks	[5]	CO2	L3
3(b)	What is the relevance of PCI Device configuration memory in detail?	[5]	CO2	L2

	PCI Device configuration memory definition:2 marks Uses:3 marks			
4(a)	What is DMA controller? How is DMA helping in faster data transfer? DMA controller Definition:2 marks Working:3 marks	[5]	CO2	L2
4(b)	Explain the centralized bus arbitration and distributed bus arbitration schemes with diagrams. Centralized bus arbitration Explanation:2.5 marks Distributed bus arbitration Explanation:2.5 marks	[5]	CO2	L2
5	What is Cache memory? Consider a cache consisting of 128 blocks of 16 words each, for total of 2048(2K) works and assume that the main memory is addressable by 16 bit address. Main memory is 64K which will be viewed as 4K blocks of 16 works each. Explain different cache memory mapping functions with the help of diagrams. Cache memory and its use:1 mark 3 mapping function:5 marks Calculation:2 marks Diagrams:2 marks	[10]	CO2	L3
6(a)	A certain processor uses a fully associative cache of size 16 kB. The cache block size is 16 bytes. Assume that the main memory is byte addressable and uses a 32-bit address. How many bits are required for the *Tag* and the *Index* fields respectively in the addresses generated by the processor? Justify your answer. Correct answer:2 marks Justification:3 marks	[5]	CO2	L4
6(b)	A computer system has an L1 cache, an L2 cache, and a main memory unit connected as shown below. The block size in L1 cache is 4 words. The block size in L2 cache is 16 words. The memory access times are 2 nanoseconds. 20 nanoseconds and 200 nanoseconds for L1 cache, L2 cache and main memory unit respectively.  L1 Cache Data Bus 4 words L2 Cache Data Bus 4 words Main Memory When there is a miss in both L1 cache and L2 cache, first a block is transferred from main memory to L2 cache, and then a block is transferred from L2 cache to L1 cache. What is the total time taken for these transfers?	[5]	CO2	L4

	Correct answer:3 marks Justification:2 marks			

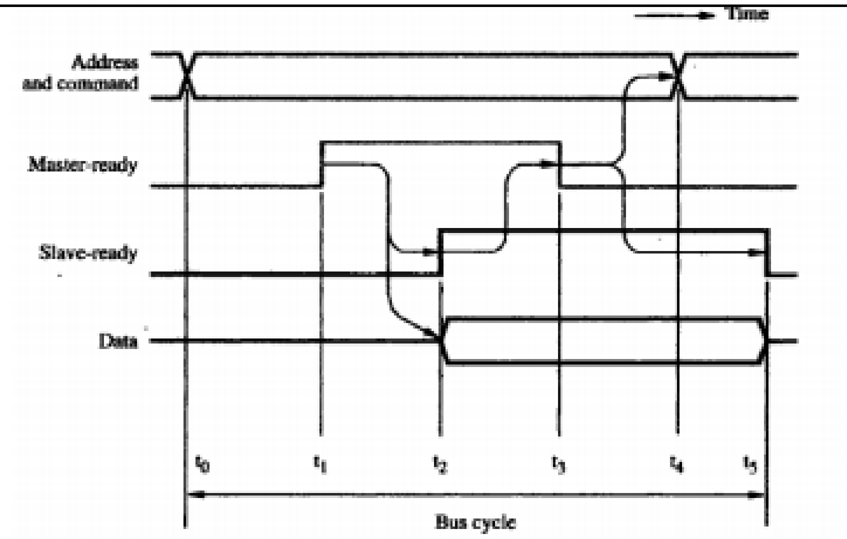
Solution

1(a)	Discuss in detail about the following bus types with timing diagram i) Synchronous Bus ii) Asynchronous Bus **Synchronous Bus** All devices derive timing information from a common clock line Equal time intervals – Bus cycle – High and Low signals Example:  Time Bus clock Address and command Data t_0 t_1 t_2 Bus cycle 2 Marks - Let consider the sequence of events during and input read operations -At time t0 the master places the device address on address bus -Read operation command given to control bus -t0 to t1 allows the device to decode address and read operation to the slave -after t1 slave places the requested input data on the data line after t1 to t2 -At the end of t2, the master 'Strobes' the data on the data lines - Strobe – To capture the values of the data at a given instant and store them into buffer 3 Marks

Asynchronous Bus

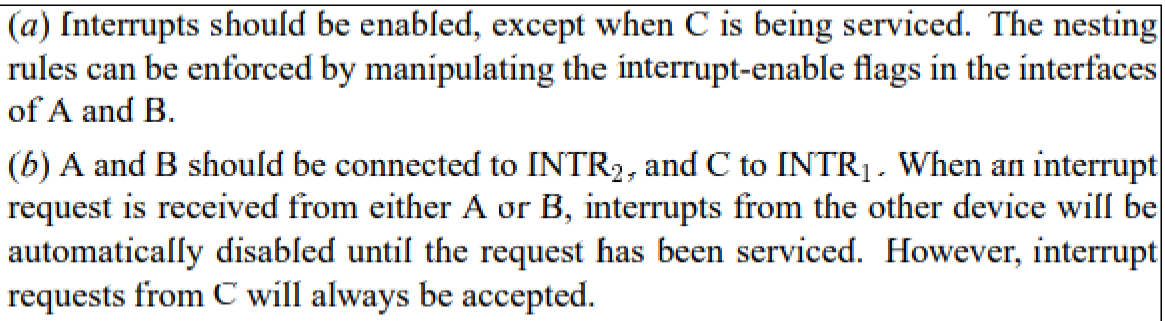
- Alternate to control data transfer on the bus is based on Handshake Signals (Acknowledge) between Master and Slave
- Common clock is replace with master-ready and slave-ready signals

Example: Handshake control of data transfer during an input operation



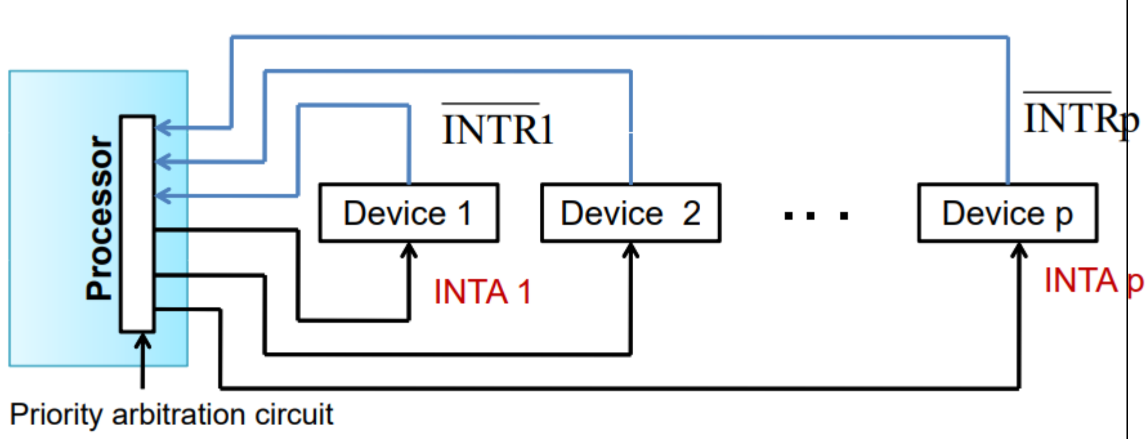
2 Marks

- t0 – The master places the address and command information on the bus
-t1- The master sets the mater-ready line to 1 to inform the data in I/O device and also slave-ready
- Skew – It occurs when two signals simultaneously transmitted from one source arrive at the destination at different times
- t2 – The selected slave decode the address and command information performs the required input operations in the data line
- t3 – The slave-ready signal arrives at the master, indicating that the input data are available on the bus.
- t4- the master removes the address and command information from the bus
- t5- When the device interface receives the 1 to 0 transition of the master-ready signal, it removes the data and the slave-ready signal from the bus. 3 Marks

<table>
<tr><td>2</td><td>

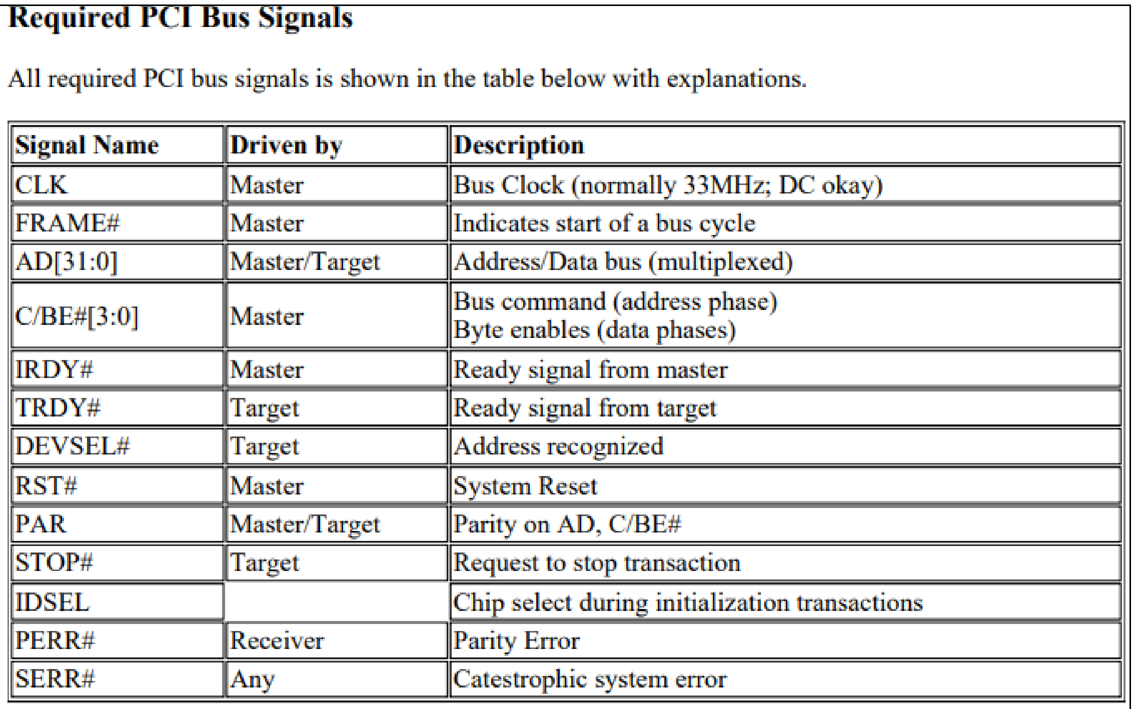
(<i>a</i>) Interrupts should be enabled, except when C is being serviced. The nesting rules can be enforced by manipulating the interrupt-enable flags in the interfaces of A and B.

(<i>b</i>) A and B should be connected to $INTR_2$, and C to $INTR_1$. When an interrupt request is received from either A or B, interrupts from the other device will be automatically disabled until the request has been serviced. However, interrupt requests from C will always be accepted.</td></tr>
<tr><td></td><td>Interrupt Nesting (contd.)


Processor
$\overline{\text{INTR}}$1
$\overline{\text{INTR}}$p
Device 1
Device 2
. . .
Device p
INTA 1
INTA p
Priority arbitration circuit

</td></tr>
<tr><td></td><td>Pre -Emption of low priority Interrupt by another high priority interrupt is known as Interrupt nesting. • Disabling interrupts during the execution of the ISR may not favor devices which need immediate attention. • Need a priority of IRQ devices and accepting IRQ from a high priority device. • The priority level of the processor can be changed dynamically. • The privileged instruction write in the PS (processor status word) , that encodes the processors priority.</td></tr>
<tr><td>3.</td><td></td></tr>
</table>

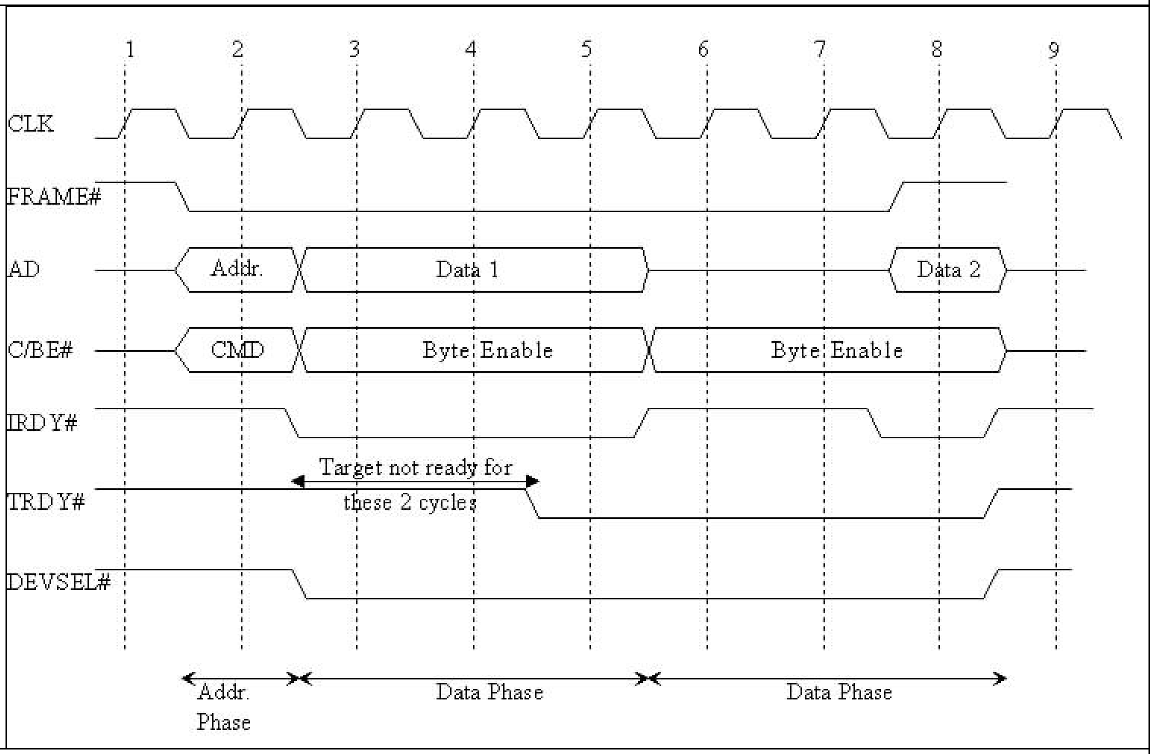
a



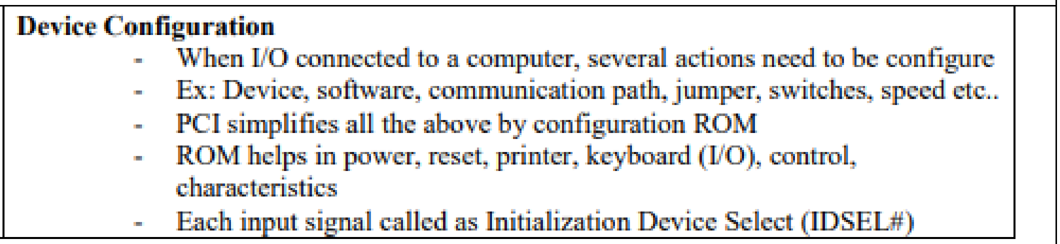
Required PCI Bus Signals

All required PCI bus signals is shown in the table below with explanations.

Signal Name	Driven by	Description
CLK	Master	Bus Clock (normally 33MHz; DC okay)
FRAME#	Master	Indicates start of a bus cycle
AD[31:0]	Master/Target	Address/Data bus (multiplexed)
C/BE#[3:0]	Master	Bus command (address phase) Byte enables (data phases)
IRDY#	Master	Ready signal from master
TRDY#	Target	Ready signal from target
DEVSEL#	Target	Address recognized
RST#	Master	System Reset
PAR	Master/Target	Parity on AD, C/BE#
STOP#	Target	Request to stop transaction
IDSEL		Chip select during initialization transactions
PERR#	Receiver	Parity Error
SERR#	Any	Catestrophic system error

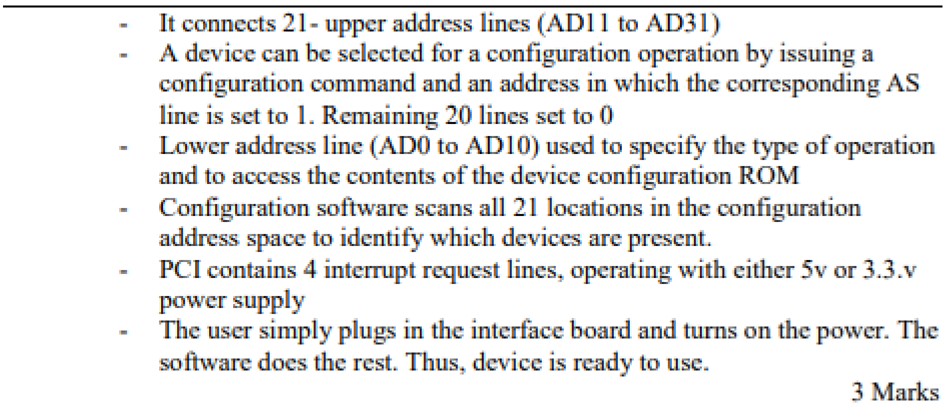
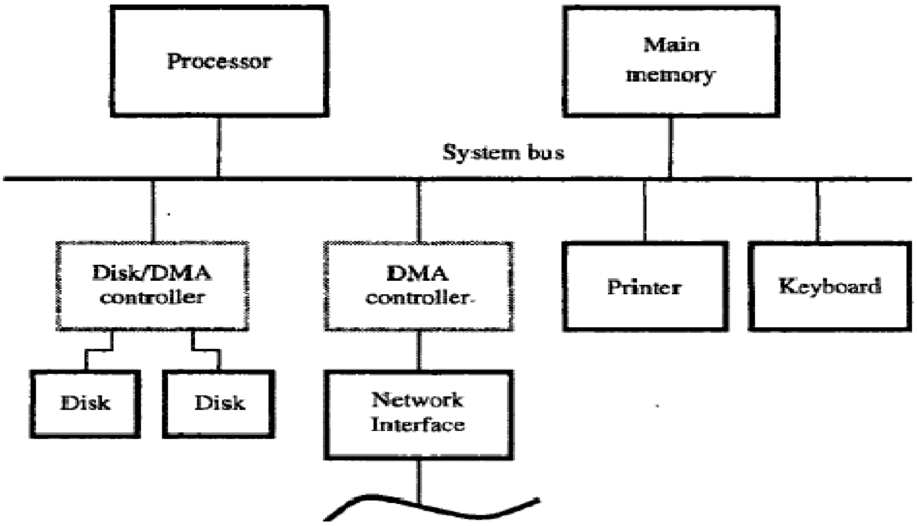


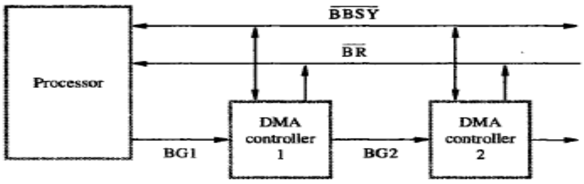
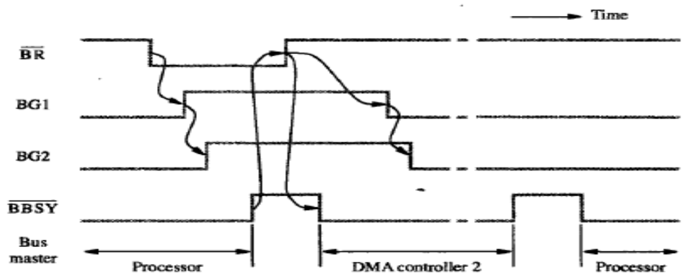
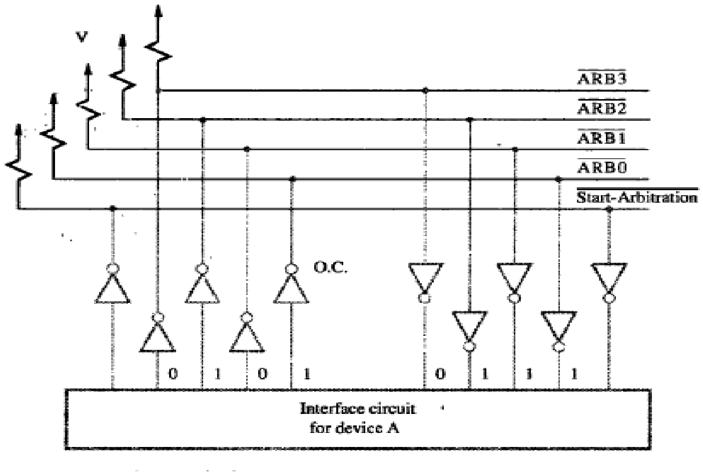
3b

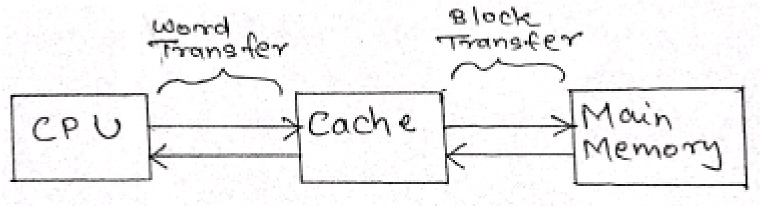


Device Configuration

- When I/O connected to a computer, several actions need to be configure
- Ex: Device, software, communication path, jumper, switches, speed etc..
- PCI simplifies all the above by configuration ROM
- ROM helps in power, reset, printer, keyboard (I/O), control, characteristics
- Each input signal called as Initialization Device Select (IDSEL#)

	 - It connects 21- upper address lines (AD11 to AD31) - A device can be selected for a configuration operation by issuing a configuration command and an address in which the corresponding AS line is set to 1. Remaining 20 lines set to 0 - Lower address line (AD0 to AD10) used to specify the type of operation and to access the contents of the device configuration ROM - Configuration software scans all 21 locations in the configuration address space to identify which devices are present. - PCI contains 4 interrupt request lines, operating with either 5v or 3.3.v power supply - The user simply plugs in the interface board and turns on the power. The software does the rest. Thus, device is ready to use. 3 Marks
4 A	• To transfer large blocks of data at high speed, an alternative approach is used • A special control unit may be provided to allow transfer of a block of data directly between an external device and the main memory, without continuous intervention by the processor. The approach is called Direct Memory Access (DMA) • DMA controller performs the functions that would normally be carried out by the processor when accessing the main memory. 2 Marks  Processor Main memory System bus Disk/DMA controller DMA controller Printer Keyboard Disk Disk Network Interface 3 Marks
4 B	**Two approaches** – Centralized Arbitration – A single bus arbitration performs the required arbitration - Distributed Arbitration – All devices participate in the selection of the next bus master

	 $\overline{BBSY}$ $\overline{BR}$ Processor DMA controller 1 DMA controller 2 BG1 BG2 **Figure 4.20** A simple arrangement for bus arbitration using a daisy chain.  Time $\overline{BR}$ BG1 BG2 $\overline{BBSY}$ Bus master Processor DMA controller 2 Processor **Figure 4.21** Sequence of signals during transfer of bus mastership for the devices in Figure 4.20. 3 Marks  V $\overline{ARB3}$ $\overline{ARB2}$ $\overline{ARB1}$ $\overline{ARB0}$ Start-Arbitration O.C. 0 1 0 1 0 1 1 1 Interface circuit for device A 2 Marks
5	**Cache Memory:-** 1) Cache Memory is very high speed memory used to increase the speed of program by making current program & data available to the CPU at a rapid rate. 2) Access time to cache memory is less compared to main memory. It contains a copy of potions of the main memory. 3) When CPU attempts to read a word from main memory, check is made to determine if the word is in cache. It so, then word is delivered form cache. 4) If word is not there in cache then a block of main memory consisting some word along with that word, is read into cache and the required word is delivered to CPU. This is called “Principle of Locality of Reference”. 5) During a miss if there are no empty blocks in the cache, then some replacement policies such as FIFO, LRU, LFU, etc. are used.



Cache Mapping Technique:-

The different Cache mapping technique are as follows:-

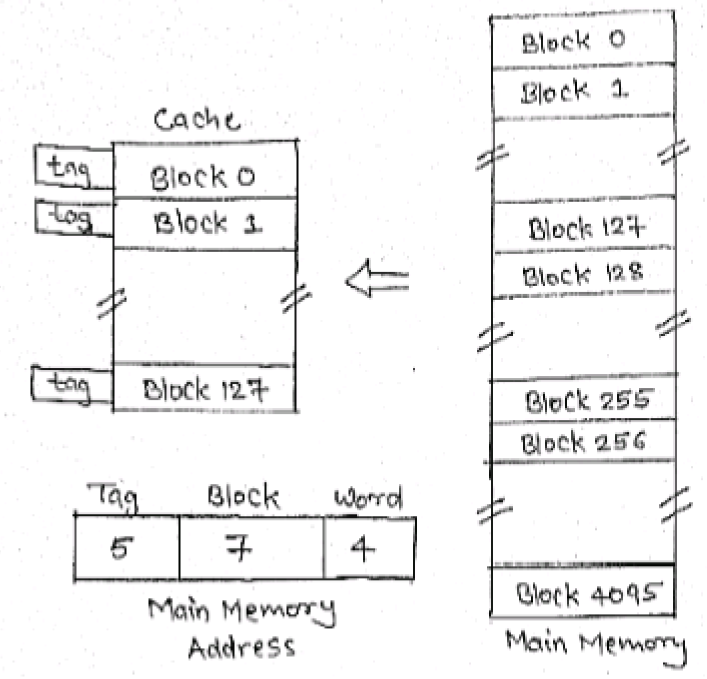
1) Direct Mapping

2) Associative Mapping

3) Set Associative Mapping

Consider a cache consisting of 128 blocks of 16 words each, for total of 2048(2K) words and assume that the main memory is addressable by 16 bit address. Main memory is 64K which will be viewed as 4K blocks of 16 words each.

(1) Direct Mapping:-



1) The simplest way to determine cache locations in which store Memory blocks is direct Mapping technique.

2) In this block J of the main memory maps on to block J modulo 128 of the cache. Thus main memory blocks 0,128,256,….is loaded into cache is stored at block 0. Block 1,129,257,….are stored at block 1 and so on.

3) Placement of a block in the cache is determined from memory address. Memory address is divided into 3 fields, the lower 4-bits selects one of the 16 words in a block.

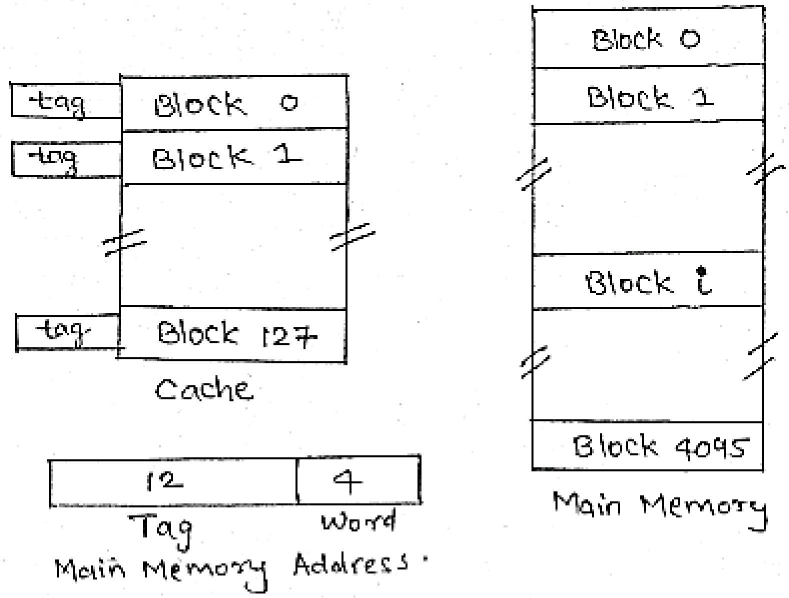
4) When new block enters the cache, the 7-bit cache block field determines the cache positions in which this block must be stored.

5) The higher order 5-bits of the memory address of the block are stored in 5 tag bits associated with its location in cache. They identify which of the 32 blocks that are

mapped into this cache position are currently resident in the cache.

6) It is easy to implement, but not Flexible

(2) Associative Mapping:-



1) This is more flexible mapping method, in which main memory block can be placed into any cache block position.

2) In this, 12 tag bits are required to identify a memory block when it is resident in the cache.

3) The tag bits of an address recevied from the processor are compared to the tag bits of each block of the cache to see, if the desired block is present. This is known as Associative Mapping technique.

4) Cost of an associated mapped cache is higher than the cost of direct-mapped because of the need to search all 128 tag patterns to determine whether a block is in cache. This is known as associative search.

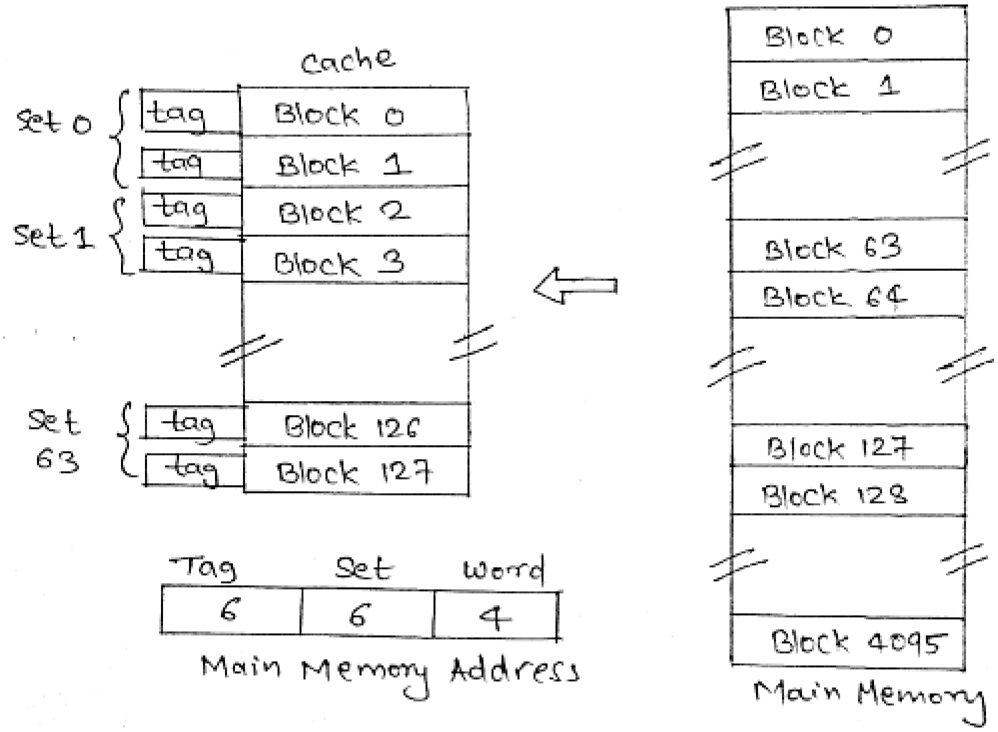
(3) Set-Associated Mapping:-

1) It is the combination of direct and associative mapping technique.

2) Cache blocks are grouped into sets and mapping allow block of main memory reside into any block of a specific set. Hence contention problem of direct mapping is eased , at the same time , hardware cost is reduced by decreasing the size of associative search.

3) For a cache with two blocks per set. In this case, memory block 0, 64, 128,…..,4032 map into cache set 0 and they can occupy any two block within this set.

4) Having 64 sets means that the 6 bit set field of the address determines which set of the cache might contain the desired block. The tag bits of address must be associatively compared to the tags of the two blocks of the set to check if desired block is present. This is two way associative search.

	 Cache Set 0 tag Block 0 tag Block 1 Set 1 tag Block 2 tag Block 3 Set 63 tag Block 126 tag Block 127 Tag Set Word 6 6 4 Main Memory Address Block 0 Block 1 Block 63 Block 64 Block 127 Block 128 Block 4095 Main Memory

6
A
	Given cache block size is 16 bytes, so block or word offset is 4 bits. Fully associative cache of size 16 kB, so line offset should be,
	`= cache size / block size`
	`= 16 kB / 16 B`
	`= 1 k`
	`= 1024`
	`= 10 bits Line or Index Offset`
	Tag bit size would be,
	`= processor address size - (line offset + word offset)`
	`= 32 - 10 - 4`
	`= 18 bits tag size`
	Since, there no option matches, but if we assume that Line Offset is a part of Tag bits, therefore,
	`Tag bits = 18+10 = 28 bits`
	`Line or Index offset = 0 bits (since fully associative cache memory),`
	`Word or block offset = 4 bits`

6 B	Since the block size of L2 cache is 16 words and the bandwidth of mainmem->L2 cache is 4 words, it requires a transfer of 4 words 4 times and then a transfer of required 4 words from L2 cache to L1 cache.

	So total time is 4*(200 + 20) + 1*(20 + 2) = 902 nanoseconds.