VTU Examination-Solution.

III Semester-B.E Examinations-18CS34-Computer Organization

1. a With neat diagram, explain the different processor registers.(8)

Accumulator:
This is the most frequently used register used to store data taken from memory. It is in different numbers in different microprocessors.

Memory Address Registers (MAR):

It holds the address of the location to be accessed from memory. MAR and MDR (Memory Data Register) together facilitate the communication of the CPU and the main memory.

Memory Data Registers (MDR):

It contains data to be written into or to be read out from the addressed location.

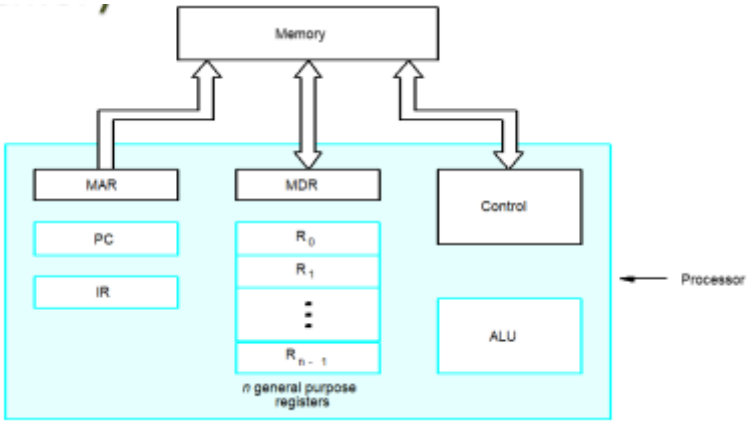
General Purpose Registers:
These are numbered as R0, R1, R2….Rn-1, and used to store temporary data during any ongoing operation.

Program Counter (PC):

Program Counter (PC) is used to keep the track of execution of the program. It contains the memory address of the next instruction to be fetched.

Instruction Register (IR):

The IR holds the instruction which is just about to be executed. The instruction from PC is fetched and stored in IR. As soon as the instruction in placed in IR, the CPU starts executing the instruction and the PC points to the next instruction to be executed.



b. Explain overall SPEC rating for the computer in a program suit(4)

$$SPEC\ rating = \frac{\text{Running time on the reference computer}}{\text{Running time on the computer under test}}$$

$$SPEC\ rating = \left(\prod_{i=1}^{n} SPEC_i\right)^{\frac{1}{n}}$$ Where n is the number of programs in the suite.

c. Explain one address, two address and three address instruction with examples(8)

One-address or 1-Operand instructions

Only one Operand will be specified in the instructions. Accumulator Register will be used as the second Operand

General Format:

Operation	Source/Destination Operand

Example: **ADD A**

Acc <- [Acc]+[A] Meaning : Add the contents of the accumulator with the memory location A ; And Store the result in the accumulator register.

Two-address or 2-Operand instructions

- Two Operands will be specified in the instructions
- General Format:

Operation	Source Operand	Source/Destination Operand

Example: **ADD A, B**

B <- [A]+[B] Meaning : Add the contents of the memory location A and B ; And Store the result in memory location B

Three-address or 3-Operand instructions

Three Operands will be specified in the instructions.

Operation	Source Operand1	Source Operand2	Destination Operand

- General Format:

Example: ADD A, B, C

C <- [A]+[B] Meaning: Add the contents of the memory location A and B ; And Store the result in memory location C

2. a What is Addressing mode? Explain diffrent addressing modes with examples of each mode.(10)

The term addressing modes refers to the way in which the operand of an instruction is specified. Information contained in the instruction code is the value of the operand or the address of the operand. Following are the main addressing modes that are used on various platforms and architectures.

1. Register Addressing Mode
2. Immediate Addressing Mode
3. Direct (or Absolute) Addressing Mode
4. Indirect Addressing Mode
5. Index Addressing Mode
6. Relative Addressing Mode
7. Auto increment Addressing Mode
8. Auto decrement Addressing Mode

1. Register Addressing Mode

- The operand is the content of a processor register. Register name is specified in the instruction.
- Effective Address of the Operand: Register name specified in the instruction

ADD R0, R1 ; R1 <- R0 + R1

Before Executing the Instruction

Registers	Contents
R0	8
R1	2

After Executing the Instruction

Registers	Contents
R0	8
R1	10

2. Immediate Addressing Mode

- The operand is given explicitly in the instruction
- Effective Address of the Operand: Operand value given in the instruction

ADD #10, R1 ; R1 <- 10 + R1

Before Executing the Instruction

Registers	Contents
R1	2

After Executing the Instruction

Registers	Contents
R1	12

3. Direct (or Absolute) Addressing Mode

- The operand is a Memory location. The address of the memory location is given in the instruction explicitly.
- Effective Address of the Operand: Address of the memory location given directly in the instruction

ADD LOCA, R1

Before Executing the Instruction

	Addr	Memory Contents
LOCA	0x1000	8

R1	2

After Executing the Instruction

	Addr	Memory Contents
LOCA	0x1000	8

R1	10

4. Indirect Addressing Mode

Here, neither the operands nor the their addresses are given explicitly. The instruction provides the information from which the address of the operand is determined i.e., the instruction provides effective address of the operand using register or memory location. The indirection is denoted by () sign around register or memory.

- Effective Address of the Operand: (Ri) or (LOCA) is the contents of a register or the memory location whose address appears in the instruction

ADD (LOCA), R1

Before Executing the Instruction

	Addr	Memory Contents
LOCA	0x1000	0x2000
	0x2000	8
R1	2	

After Executing the Instruction

	Addr	Memory Contents
LOCA	0x1000	0x2000
	0x2000	8
R1	10	

5. Index Addressing Mode

- The effective address of the operand is generated by adding a constant value to the contents of a register specified in the instruction. The register in this case is called as Index register.
- The operation is indicated as X(Ri).

Effective Address of the Operand: X+Ri where X is a constant value (signed integer) and Ri is the index register

ADD 5(R1), R2

Before Executing the Instruction

	Addr	Memory Contents
	0x2005	2

R1	0x2000
R2	8

After Executing the Instruction

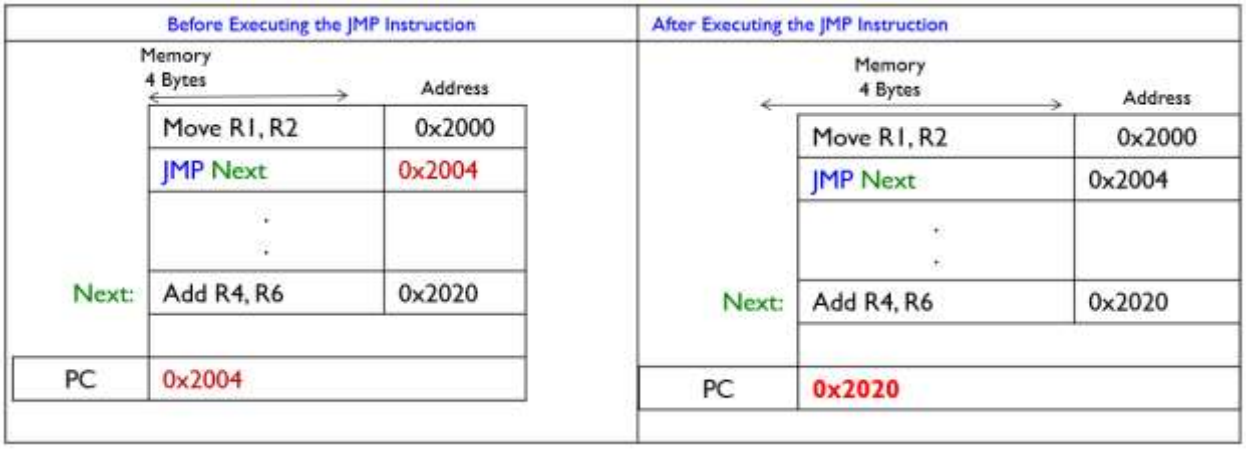
	Addr	Memory Contents
	0x2005	2

R1	0x2000
R2	10

6. Relative Addressing Mode

- In this mode the content of the program counter is added to the address part of the instruction to obtain the effective address.

- Effective Address : X+PC➔ used to address memory location that is X bytes away from the location presently pointed by the program counter. where X is a constant value (signed integer) and PC is the contents of the program counter.



7.Autoincrement Addressing Mode

This is indirect mode with a modification. The effective address of the operand is the contents of a pointer register specified in the instruction. After accessing the operand, the contents of this pointer register is incremented automatically to point to the next entity.
The mode is denoted by (Ri)+, where Ri is the pointer register.
The + sign indicates that Ri is incremented after the operation.

- Example for Autoincrment Addressing Mode

Move (R1)+, R2

2.b Explain shift and rotate instructions with example(10).

- There are many applications that require the bits of an operand to be shifted right or left some specified number of bit positions.
- The details of how the shifts are performed depend on whether the operand is assigned number or some more general binary-coded in formation.
- For general operands, we use a logical shift.

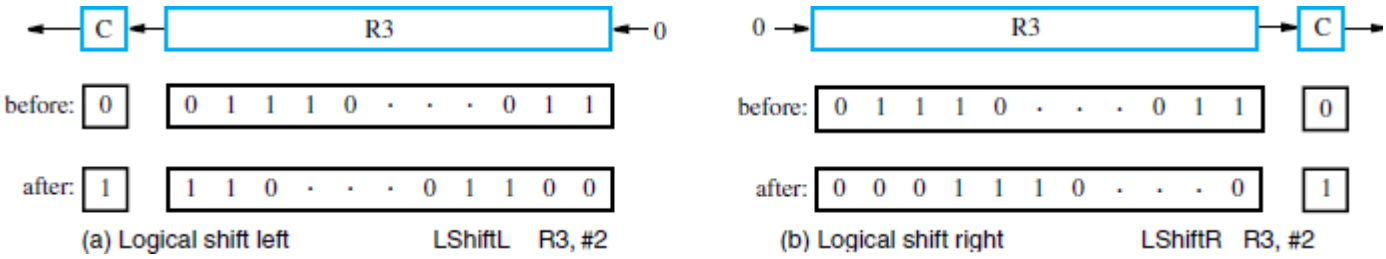
For a number ,we use an arithmetic shift ,which preserves the sign of the number.

LOGICAL SHIFTS

- Two logical shift instructions are

1) Shifting left (LShiftL) &
2) Shifting right (LShiftR).

- These instructions shift an operand over a number of bit positions specified in a count operand contained in the instruction.



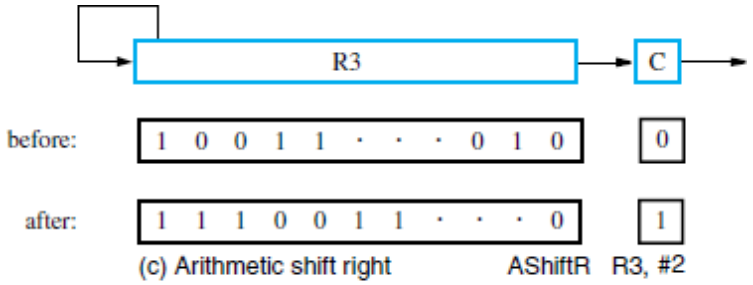
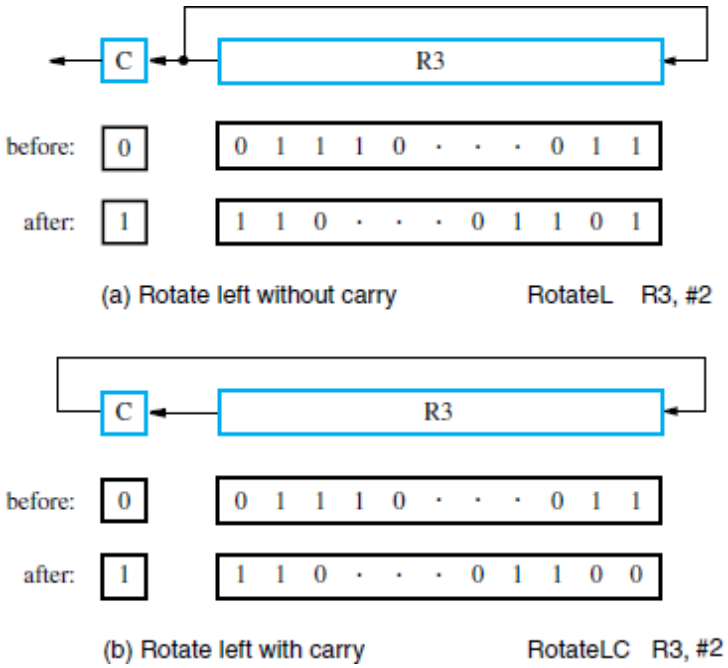
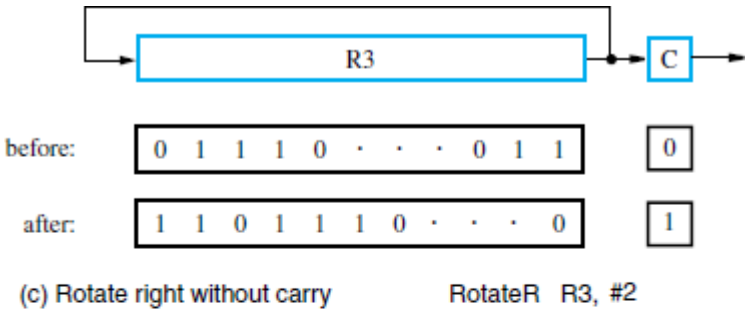


Figure 2.23 Logical and arithmetic shift instructions.

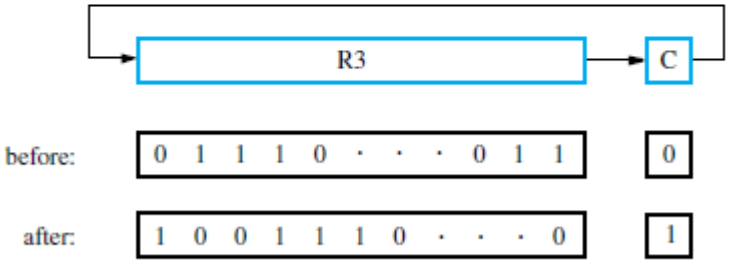
ROTATE OPERATIONS

- In shift operations , the bits shifted out of the operand are lost, except for the last bit shifted out which is retained in the Carry-flag C.
- To preserve all bits a set of rotate instructions can be used.
- They move the bits that are shifted out of one end of the operand back into the other end.
- Two versions of both the left and right rotate instructions are usually provided .In one version, the bits of the operand is simply rotated.
- In the other version, the rotation includes the C flag.





(c) Rotate right without carry RotateR R3, #2



(d) Rotate right with carry RotateRC R3, #2

Figure 2.25 Rotate instructions.

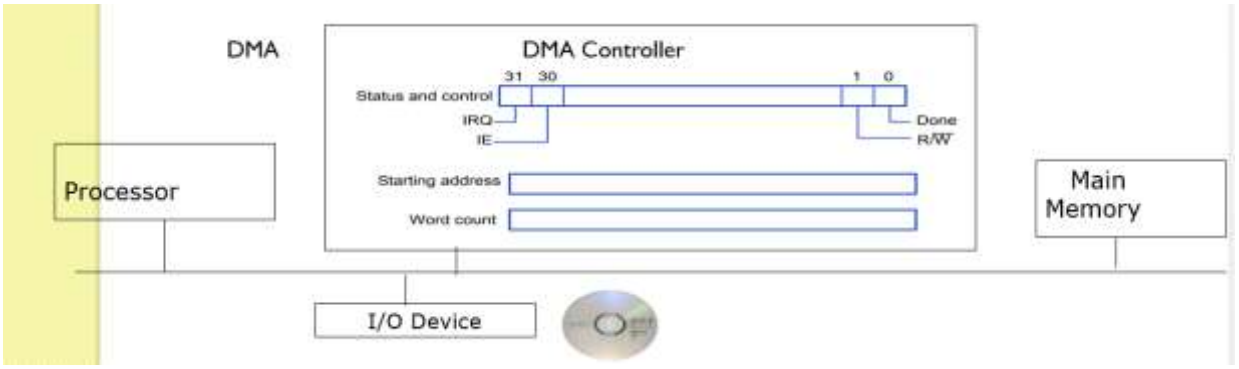
3.a What is DMA? When it is used? Explain it with block diagram. (8)

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, without the continuous intervention by the processor.

To transfer large blocks of data at high speed, DMA approach will be used.

- Direct Memory Access (DMA): A special control unit to transfer a block of data directly between an I/O device and the main memory, without continuous intervention by the processor.
- Control unit which performs these transfers is a part of the I/O device's interface circuit. This control unit is called as a DMA controller.
- DMA controller performs functions that would be normally carried out by the processor:
    - For each word, it provides the memory address and all the control signals.

- To transfer a block of data, it increments the memory addresses and keeps track of the number of transfers.



3.b Explain the term "cycle stealing" and "burst mode" with respect to DMA.(4)

Burst Mode –

- In this mode Burst of data (entire data or burst of block containing data) is transferred before CPU takes control of the buses back from DMAC.
- This is the quickest mode of DMA Transfer since at once a huge amount of data is being transferred.
- Since at once only the huge amount of data is being transferred so time will be saved in huge amount.

```
Percentage of time CPU remains in blocked state = Ty * 100% / Tx + Ty
```

Cycle Stealing Mode –

- Slow IO device will take some time to prepare data (or word) and within that time CPU keeps the control of the buses.
- Once the the data or the word is ready CPU give back control of system buses to DMAC for 1-cycle in which the prepared word is transferred to memory.
- As compared to Burst mode this mode is little bit slowest since it requires little bit of time which is actually consumed by IO device while preparing the data.

Percentage of Time CPU remains blocked :

Percentage of time CPU remains in blocked state = Ty * 100% / Tx

3.c **What is bus arbitration? Explain Distributed bus arbitration.(8)**

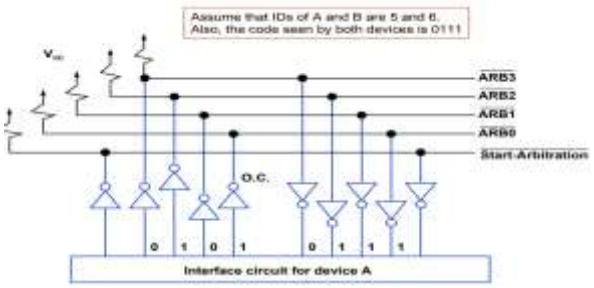
Bus arbitration:
Bus Arbitration – the process by which the next device to become bus master is selected and bus mastership is transferred to it.

distributed bus arbitration.
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, without the continuous
intervention by the processor. To transfer large blocks of data at high speed, DMA approach will be used.

Distributed Bus Arbitration

- No central arbiter used

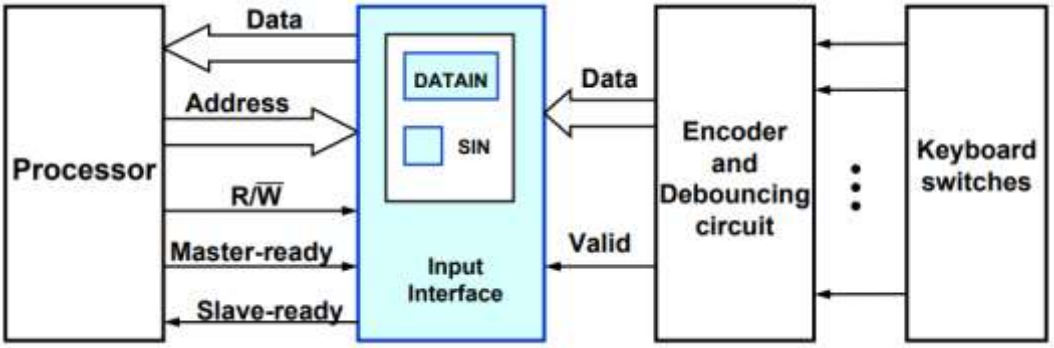
- Each device on bus is assigned a 4-bit identification number.
- When one or more devices request the bus, they assert the Start-Arbitration signal and place their 4-bit ID number on ARB[3..0].
- The request that has the highest ID number ends up having the highest priority.
- Advantages – offers higher reliability (operation of the bus is not dependent on any one device).
- Assume that two devices, A and B, having ID numbers 5 and 6, respectively, are requesting the use of the bus.
- Device A transmits the pattern 0101, and device B transmits the pattern 0110.
- The code seen by both devices is 0111.
- Each device compares the pattern on the arbitration lines to its own ID, starting from the most significant bit.
- If it detects a difference at any bit position, it disables its drivers at that bit position and for all lower-order bits. It does so by placing a 0 at the input of these drivers.
- In the case of our example, device A detects a difference on line ARB 1. Hence, it disables its drivers on lines ARB1 and ARB0.
- This causes the pattern on the arbitration lines to change to 0110, which means that B has won the contention.



4. A. With a block diagram explain how a keyboard is connected to the processor.(8)

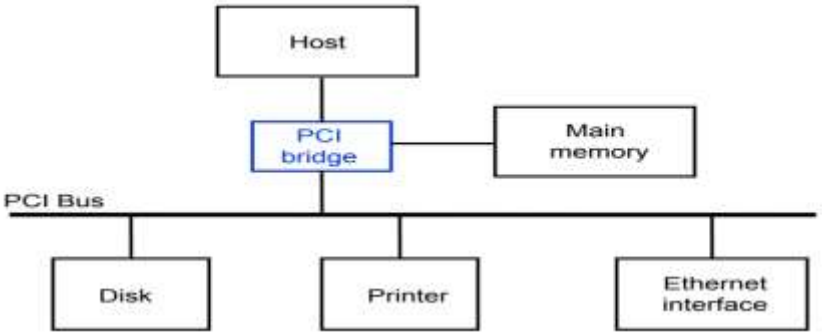
Keyboard to processor connection

- When a key is pressed, the Valid signal changes from 0 o 1, causing the ASCII code to be loaded into DATAIN and SIN to be set to 1.
- The status flag SIN is cleared to 0 when the processor reads the contents of the DATAIN register.



4. B Explain the use of PCI bus in a computer system with neat sketch.(8)

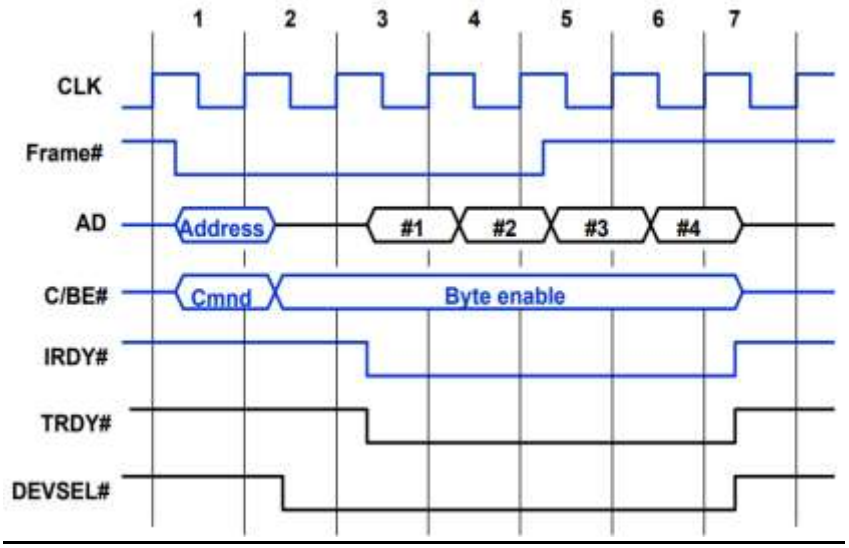
- PCI is developed as a low cost bus that is truly processor independent.
- PCI supports high speed disk, graphics and video devices.
- PCI has plug and play capability for connecting I/O devices.
- To connect new devices, the user simply connects the device interface board to the bus.



- The bus support three independent address spaces: memory, I/O, and configuration.
- The I/O address space is intended for use with processors, such Pentium, that have a separate I/O address space.
- However, the system designer may choose to use memory-mapped I/O even when a separate I/O address space is available
- The configuration space is intended to give the PCI its plug-and-play capability. ‹
  - A 4-bit command that accompanies the address identifies which of the three spaces is being used in a given data transfer operation

Name	Function
CLK	A 33-MHz or 66MHz clock
FRAME#	Sent by the initiator to indicate the duration of a transaction
AD	32 address/data lines, which may be optionally increased to 64
C/BE#	4 command/byte-enable lines (8 for 64-bit bus)
IRDY#, TRDY#	Initiator-ready and Target-ready signals
DEVSEL#	A response from the device indicating that it has recognized its Address and is ready for a data transfer transaction
IDSEL#	Initialization Device Select

A Read Operation on the PCI Bus



4.C What are the design objectives of USB.(4)

☐ Universal Serial Bus, USB is a plug-and-play interface that allows a computer to communicate with peripheral and other devices.

- USB stands for Universal Serial Bus.
- USB supports 3 speed of operation. They are,
  1) Low speed (1.5 Mbps)
  2) Full speed (12 mbps) &
  3) High speed (480 mbps).
- The USB has been designed to meet the key objectives. They are,
  1) Provide a simple, low-cost and easy to use interconnection system.
     This overcomes difficulties due to the limited number of I/O ports available on a computer.
  2) Accommodate a wide range of data transfer characteristics for I/O devices.
     For e.g. telephone and Internet connections
  3) Enhance user convenience through a "plug-and-play" mode of operation.
- **Advantage:** USB helps to add many devices to a computer system at any time without opening the computer-box.

Port Limitation

- Normally, the system has a few limited ports.
- To add new ports, the user must open the computer-box to gain access to the internal expansion bus & install a new interface card.
- The user may also need to know to configure the device & the s/w.

Plug & Play

- The main objective: USB provides a plug & play capability.
- The plug & play feature enhances the connection of new device at any time, while the system is operation.
- The system should
  - → Detect the existence of the new device automatically.
  - → Identify the appropriate device driver s/w.
  - → Establish the appropriate addresses.
  - → Establish the logical connection for communication.

5. A. Draw a neat block diagram of memory Hierarchy in a computer system. Discuss the variation of size, speed and cost per bit in a hierarchy.(8)

Solution:

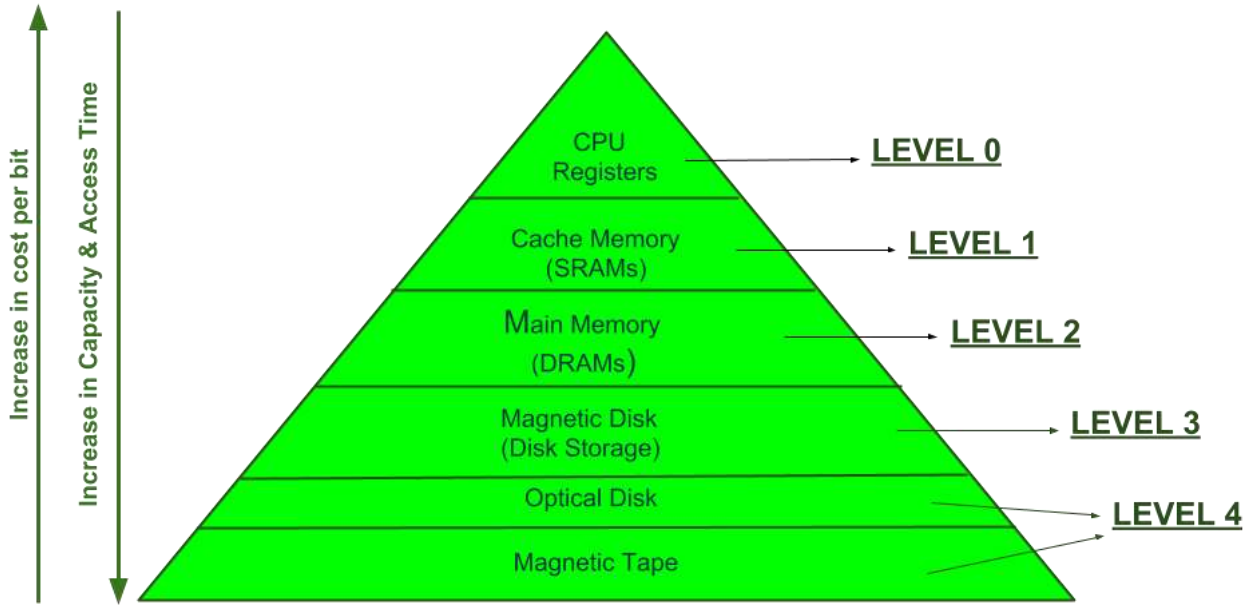
In the Computer System Design, Memory Hierarchy is an enhancement to organize the memory such that it can minimize the access time. The Memory Hierarchy was developed based on a program behavior known as locality of references. The figure below clearly demonstrates the different levels of memory hierarchy.

This Memory Hierarchy Design is divided into 2 main types:

1. **External Memory or Secondary Memory –**
   Comprising of Magnetic Disk, Optical Disk, Magnetic Tape i.e. peripheral storage devices which are accessible by the processor via I/O Module.
2. **Internal Memory or Primary Memory –**
   Comprising of Main Memory, Cache Memory & CPU registers. This is directly accessible by the processor.

We can infer the following characteristics of Memory Hierarchy Design from above figure:

1. **Capacity:**
   It is the global volume of information the memory can store. As we move from top to bottom in the Hierarchy, the capacity increases.
2. **Access Time:**
   It is the time interval between the read/write request and the availability of the data. As we move from top to bottom in the Hierarchy, the access time increases.
3. **Performance:**
   Earlier when the computer system was designed without Memory Hierarchy design, the speed gap increases between the CPU registers and Main Memory due to large difference in access time. This results in lower performance of the system and thus, enhancement was required. This enhancement was made in the form of Memory Hierarchy Design because of which the performance of the system increases. One of the most significant ways to increase system performance is minimizing how far down the memory hierarchy one has to go to manipulate data.
4. **Cost per bit:**
   As we move from bottom to top in the Hierarchy, the cost per bit increases i.e. Internal Memory is costlier than External Memory.



MEMORY HIERARCHY DESIGN

5.B Explain the working of a single transistor dynamic memory cell and the internal organization of 16 mega bit DRAM chip configured as 2M X 8 cells(12)

Solution:

Transistor pairs (T3, T5) and (T4, T6) form the inverters in the latch . For example, in state 1, the voltage at point X is maintained high by having transistors T3 and T6 on, while T4 and T5 are off. Thus, if T1 and T2 are turned on (closed), bit lines b and b‟ will have high and low signals, respectively.

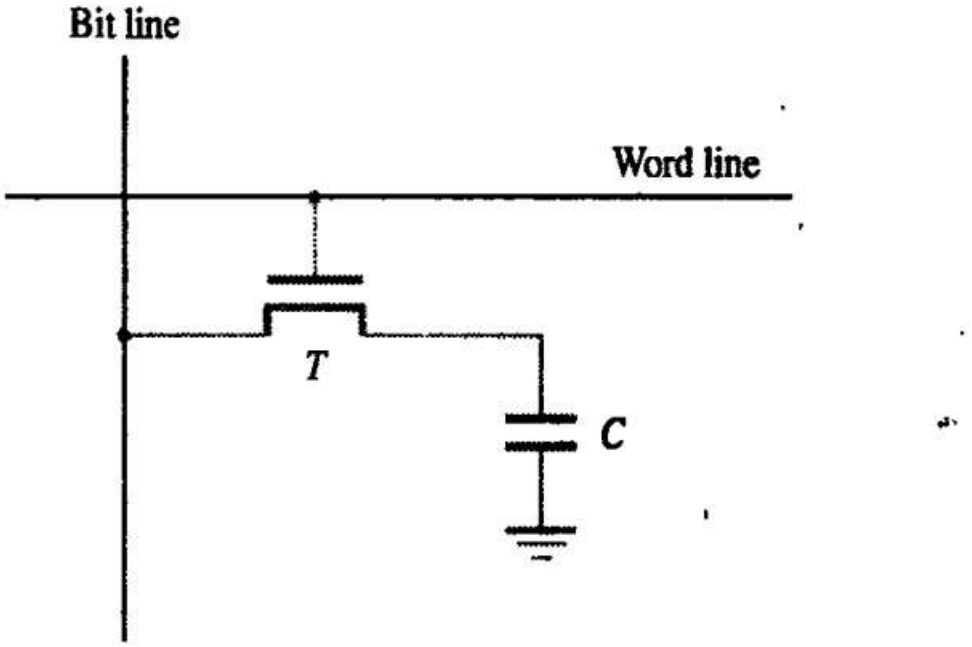
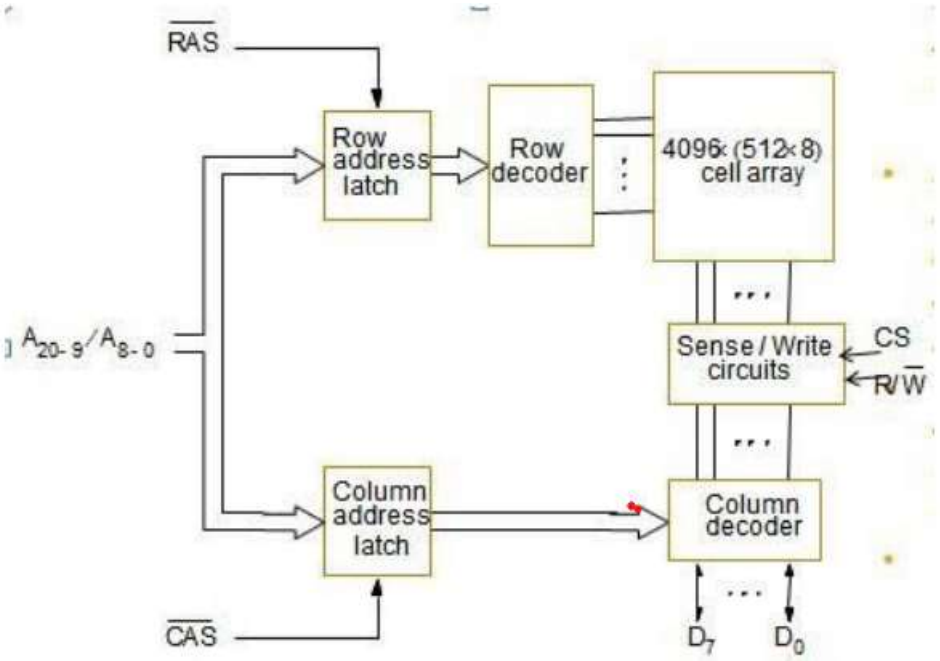


Figure 5.6 A single-transistor dynamic memory cell.

Internal organization of 16 mega bit DRAM chip configured as 2M X 8 cells.

Asynchronous DRAMS Information is stored in a dynamic memory cell in the form of a charge on a capacitor, and this charge can be maintained for only tens of milliseconds. Since the cell is required to store information for a much longer time, its contents must be periodically refreshed by restoring the capacitor charge to its full value. An example of a dynamic memory cell that consists of a capacitor, C, and a transistor, T, is shown below: A sense amplifier connected to the bit line detects whether the charge stored on the capacitor is above the threshold. If so, it drives the bit line to a full voltage that represents logic value 1. This voltage recharges the capacitor to full charge that corresponds to logic value 1. If the sense amplifier detects that the charge on the capacitor will have no charge, representing logic value 0.



A 16-megabit DRAM chip, configured as 2M×8, is shown below. ϒ Each row can store 512 bytes. 12 bits to select a row, and 9 bits to select a group in a row. Total of 21 bits. • First apply the row address; RAS signal latches the row address. Then apply the column address, CAS signal latches the address. • Timing of the memory unit is controlled by a specialized unit which generates RAS and CAS. • This is asynchronous DRAM.

6.A Explain different mapping functions used in catch memory memory.(12)

(i)Direct mapping

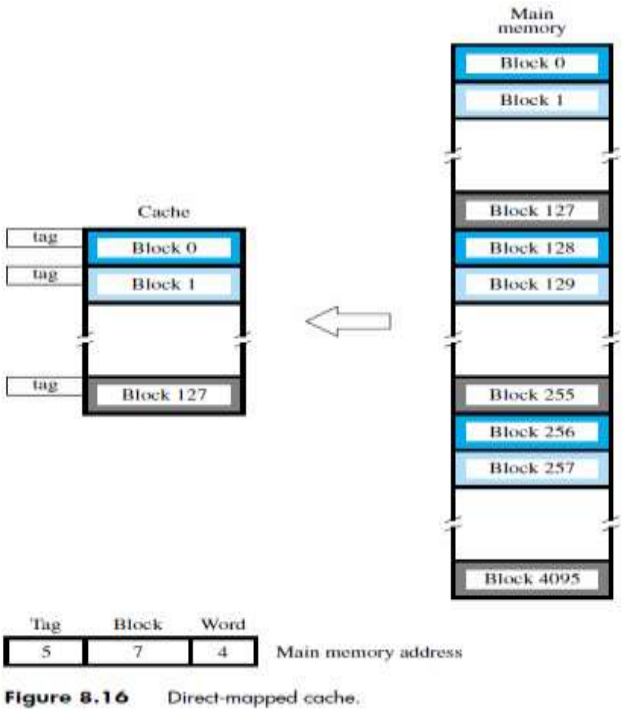


Figure 8.16 Direct-mapped cache.

- Block j of the main memory maps to

j modulo 128 of the cache. 0 maps to 0, 129 maps to 1.

- More than one memory block is mapped onto the same position in the cache.
- May lead to contention for cache blocks even if the cache is not full.
- Resolve the contention by allowing new block to replace the old block, leading to a trivial replacement algorithm.
- The placement of a block in the catch is determined from the memory address.

- Memory address is divided into three fields:

- Low order 4 bits determine one of the 16 words in a block.

- When a new block is brought into the cache, the next 7 bits determine which cache block this new block is placed in.

- High order 5 bits determine which of the possible 32 blocks is currently present in the cache. These are tag bits.

- Simple to implement but not very flexible.

(i) Associative mapping

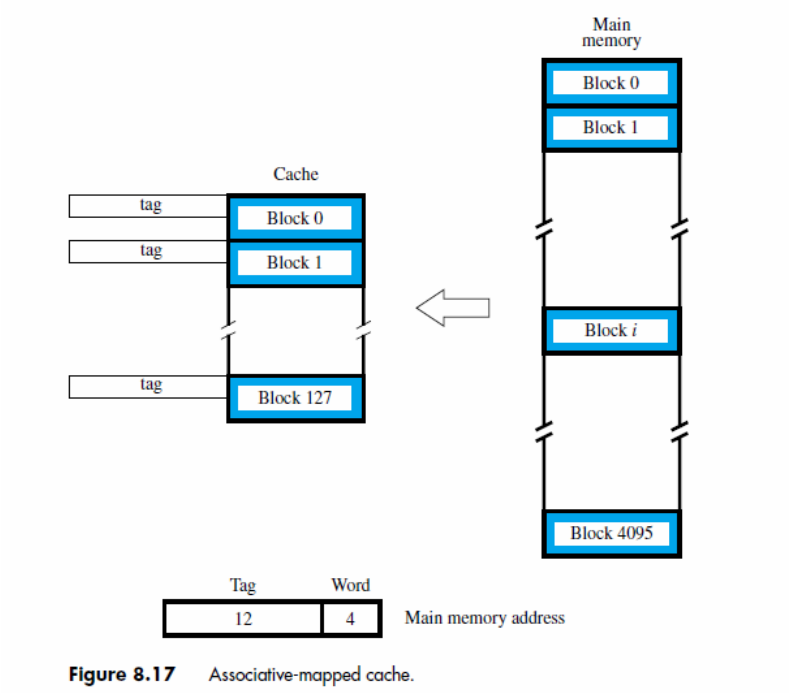


Figure 8.17 Associative-mapped cache.

- Main memory block can be placed into any cache position.
- Memory address is divided into two fields:

- Low order 4 bits identify the word within a block.

- High order 12 bits or tag bits identify a memory lock when it is resident in the cache.

- Flexible, and uses cache space efficiently.
- Replacement algorithms can be used to replace an existing block in the cache when the cache is full.

- Cost is higher than direct-mapped cache because of the need to search all 128 patterns to determine whether a given block is in the cache.

Advantage of associative mapping:

There is flexibility when mapping a block to any block of the cache

Disadvantages of associative mapping:

1. A replacement algorithm must be used to determine which block of cache to swap out

2. More space is needed for the tag field

3. The most important disadvantage is the complex circuitry needed to examine all of the tags in parallel in the cache

Set-Associative mapping

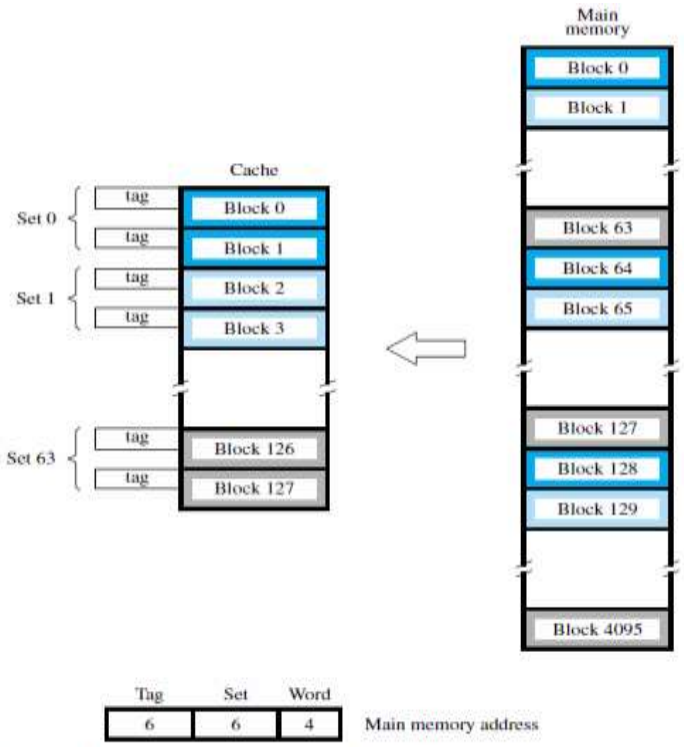


Figure 8.18 Set-associative-mapped cache with two blocks per set.

- Blocks of cache are grouped into sets.
- Mapping function allows a block of the main memory to reside in any block of a specific set.
- Divide the cache into 64 sets, with two blocks per set.
- Memory block 0, 64, 128 etc. map to block 0, and they can occupy either of the two positions.
- Memory address is divided into three fields:

- 6 bit field determines the set number.

- High order 6 bit fields are compared to the tag

fields of the two blocks in a set.

- Set-associative mapping combination of direct and associative mapping.
- Number of blocks per set is a design parameter.

- One extreme is to have all the blocks in one set,requiring no set bits (fully associative mapping).

- Other extreme is to have one block per set, is the same as direct mapping.

6.B what is Page replacement policy? Explain LRU replacement algorithm.(8)

what is Page replacement policy

When a Main Memory (MM) block needs to be brought in while all the Cache memory (CM) blocks are occupied, one of them has to be replaced.

LRU: Replace the block in CM that has not been used for the longest time, i.e., the Least Recently Used (LRU) block. LRU replacement can be implemented by attaching a number to each CM block to indicate how recent a block has been used. Every time a processor reference is made all of these numbers are updated in such a way that the **smaller a number the more recent it was used**, i.e., the LRU block is always indicated by the largest number.

- Block needed: set the number to zero: N=0
- In case of a miss, bring the block needed in (replace an existing CM block if necessary)
- Assume the initial state of the 4 CM blocks is shown below (the underlined block is the least recently used):

Block name	**A**	**B**	**C**	**D**
LRU Counter	0	2	1	3

After block B is referenced,

Block name	**A**	**B**	**C**	**D**
LRU Counter	1	0	2	3

After block C is referenced,

Block name	**A**	**B**	**C**	**D**
LRU Counter	2	1	0	3

After block D is referenced,

Block name	**A**	**B**	**C**	**D**
LRU Counter	3	2	1	0

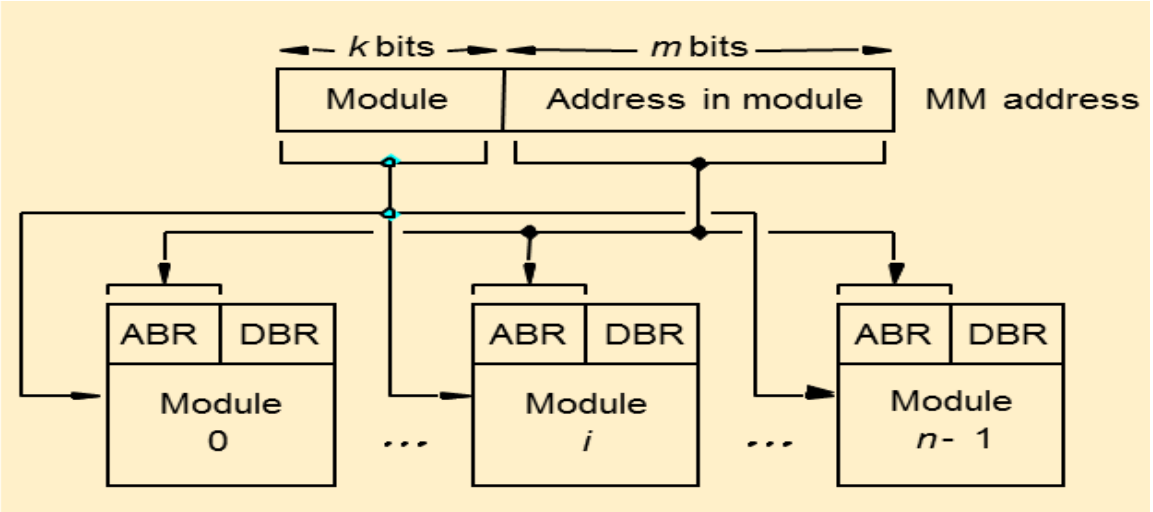
If block E not currently in CM is needed, the LRU CM block A labeled by the largest number 3 will be replaced

Block name	**E**	**B**	**C**	**D**
LRU Counter	0	3	2	1

6. C Explain memory interleaving with necessary diagram.(4)

- Divides the memory system into a number of memory modules. Each module has its own address buffer register (ABR) and data buffer register (DBR).
- Arranges addressing so that successive words in the address space are placed in different modules.

- When requests for memory access involve consecutive addresses, the access will be to different modules.
- Since parallel access to these modules is possible, the average rate of fetching words from the Main Memory can be increased.



- ***<u>Consecutive words are placed in a module.</u>***
- ***High-order k bits of a memory address determine the module.***
- ***Low-order m bits of a memory address determine the word within a module.***
- ***When a block of words is transferred from main memory to cache, only one module is busy at a time.***

7.a perform the following operations on the 5–bit signed bit numbers using 2's complement representation system.(10)

7(a) (i) (-10)+(-13)

(+10) 01010 —2's→ 10110 (-10)

(+13) 01101 —2's→ 10011 (-13)

01001

1

overflow

(ii) (-10) - (+4)

(+10) 01010 —2's→ 10110 (-10)

(+4) 00100 —2's→ 11100

10010

(iii) (-3) + (-8)

(+3) 00011 —2's→ 11101 (-3)

(+8) 01000 —2's→ 11000 (-8)

10101

(iv) (-10) - (+7)

(+10) 01010 —2's→ 10110 (-10)

(+7) 00111 —2's→ 11001

01111

7.B In a carry look ahead addition, explain generate Gi and propagate Pi functions for stage. Using this design explain 4-bit carry look ahead adder.(10)

SOLUTION: CARRY LOOK AHEAD ADDITION

- The logic expression for si (sum) and ci+1(carry-out) of stage i are

si= xi+yi+ci (1)

Ci+1=xiyi+xici+yici (2)

- Factoring (2) into

Ci+1=xiyi+ (xi+yi)

ci

We can write ci+1=Gi+ PiCi where Gi=xiyi and Pi=xi+yi

- The expressions Gi and Pi are called generate and propagate functions.

- If Gi=1, then ci+1=1, independent of the input carry ci. This occurs when both xi and yi are 1. Propagate function means that an input-carry will produce an output-carry when either xi=1 or yi=1.

- All Gi and Pi functions can be formed independently and in parallel in one logic-gate delay.

- Expanding ci terms of i-1 subscripted variables and substituting into the ci+1 expression, we obtain ci+1= Gi+PiGi-1+PiPi-1Gi-2. . . . . .+P1G0+PiPi-1 P0c0

- Conclusion: Delay through the adder is 3 gate delays for all carry-bits

& 4 gate delays for all sum-bits.

- Consider the design of a 4-bit adder. The carries can be implemented as

c1=G0+P0c0

c2=G1+P1G0+P1P0c0

c3=G2+P2G1+P2P1G0+P2P1P0c0

c4=G3+P3G2+P3P2G1+P3P2P1G0+P3P2P1P0c0

- The carries are implemented in the block labeled carry-look ahead logic. An adder implemented in this form is called a *carry-look ahead adder*.
- Limitation: If we try to extend the carry-look ahead adder for longer operands, we run into a problem of gate fan-in constraints.

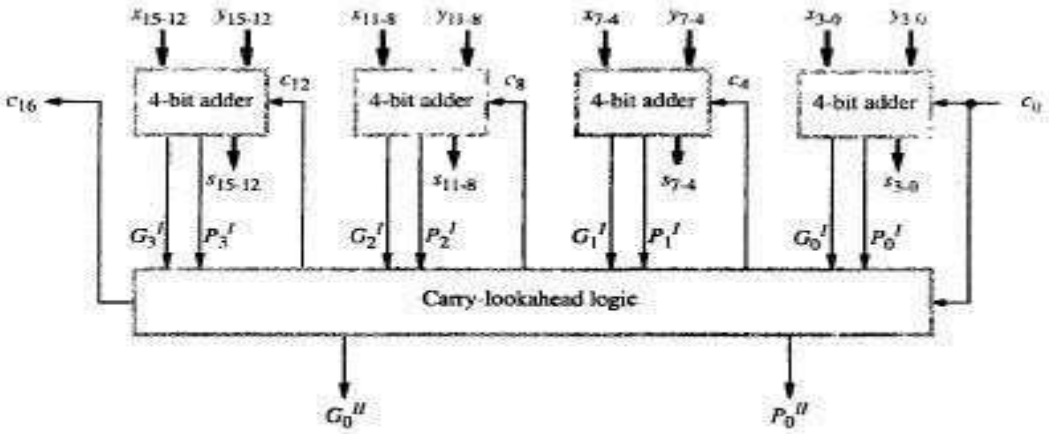


Figure 6.5 16-bit carry-lookahead adder built from 4-bit adders (see Figure 6.4b).

8.a Perform the signed multiplication of numbers +13 and -6 using booth multiplication and bit pair recording method. List the table used.

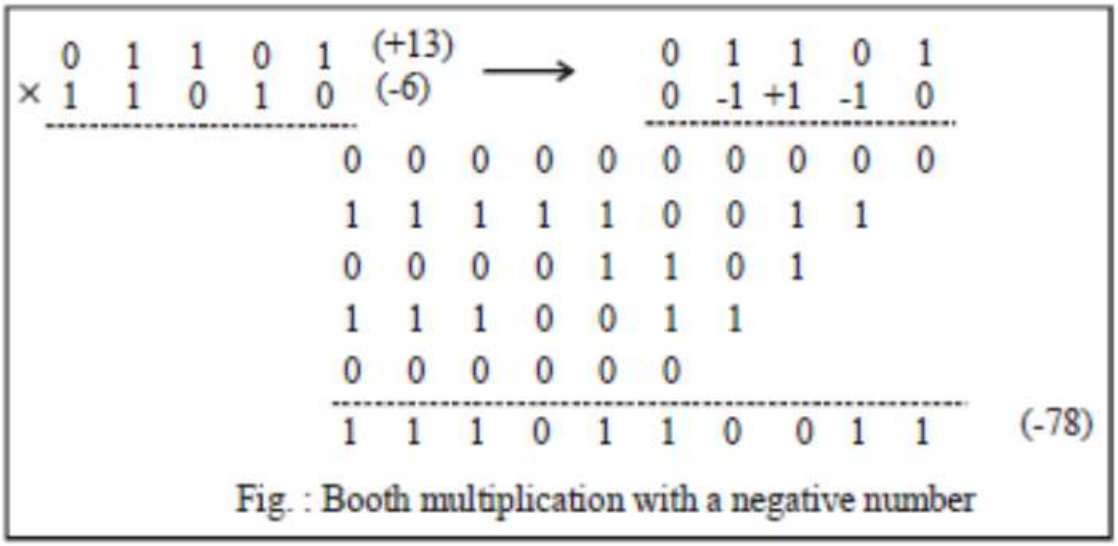
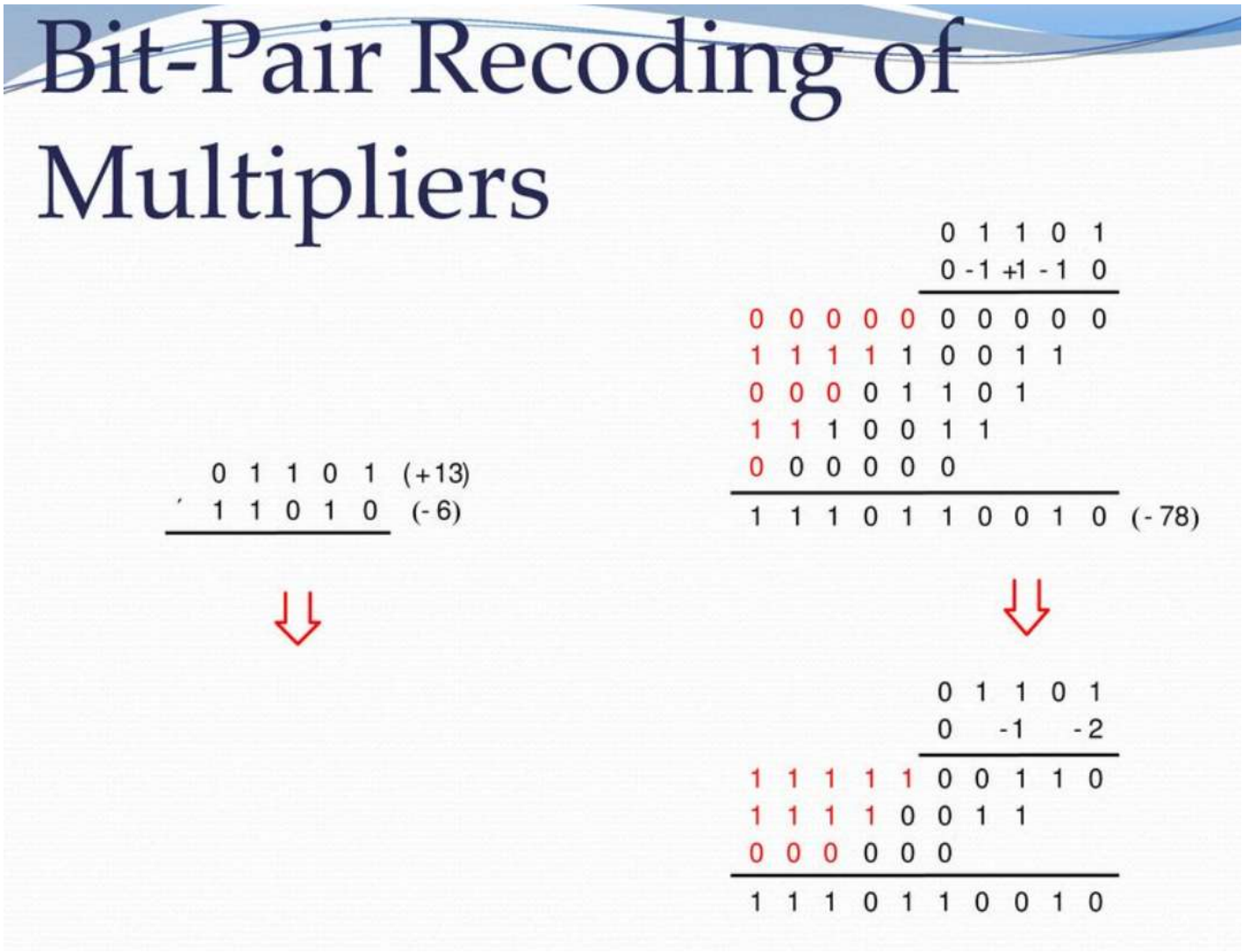


Fig. : Booth multiplication with a negative number



8.b perform division of number 9 by 3(9+3) using the restoring division algorithm.write the steps of algorithm used.

Restoring Division

- **Shift A and Q left one binary position**
- **Subtract M from A, and place the answer back in A**
- **If the sign of A is 1, set q_0 to 0 and add M back to A (restore A); otherwise, set q_0 to 1**

Repeat these steps *n* times

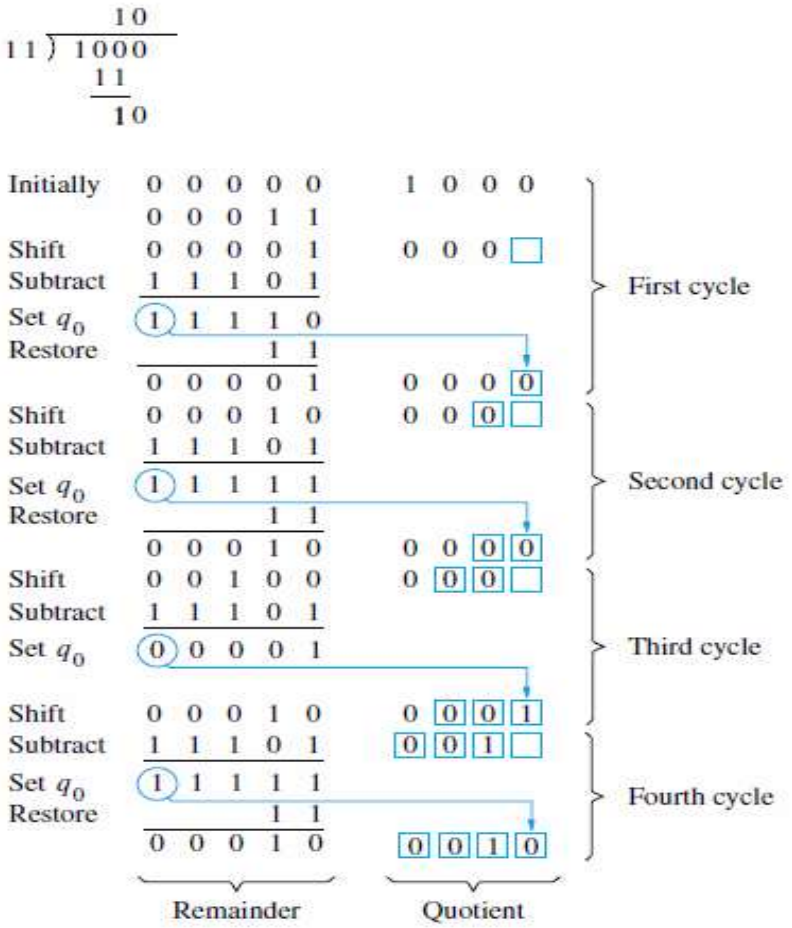
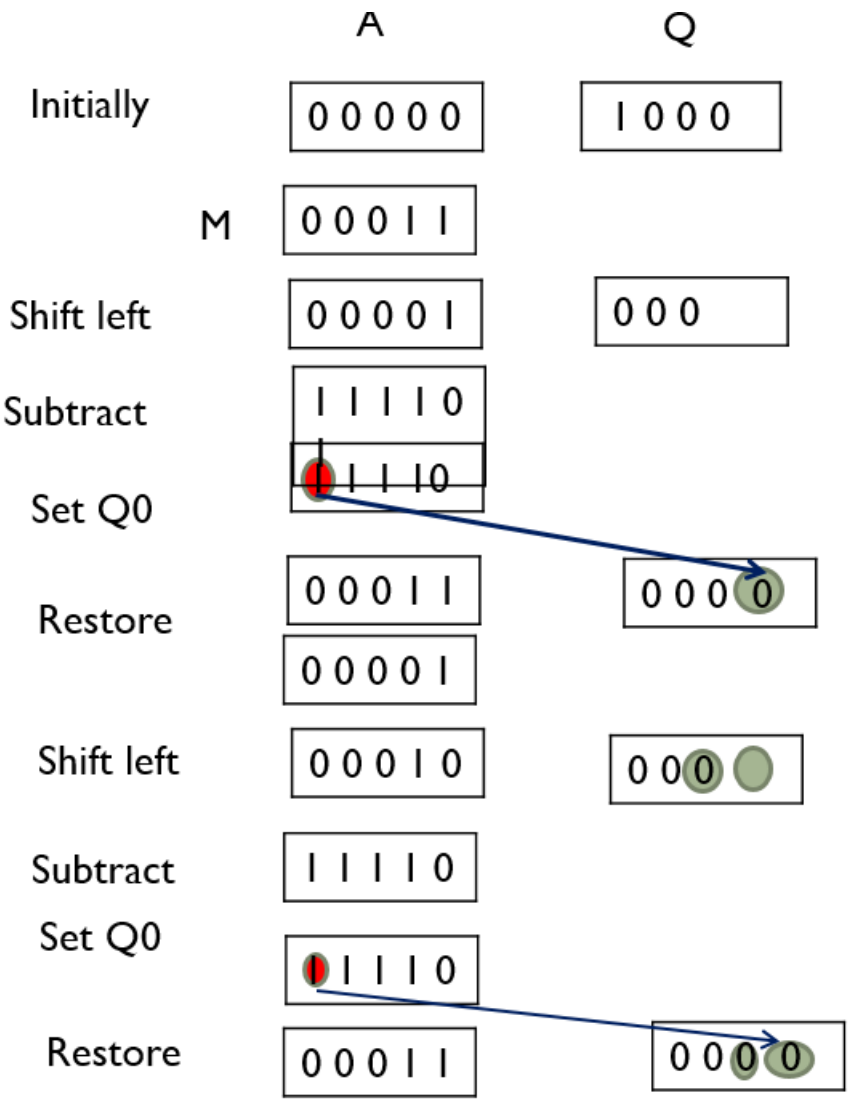


Figure 9.24 A restoring division example.


A
Q
Initially
0 0 0 0 0
1 0 0 0
M
0 0 0 1 1
Shift left
0 0 0 0 1
0 0 0
Subtract
1 1 1 1 0
1 1 1 0
Set Q0
Restore
0 0 0 1 1
0 0 0 0
0 0 0 0 1
Shift left
0 0 0 1 0
0 0 0
Subtract
1 1 1 1 0
Set Q0
1 1 1 1 0
Restore
0 0 0 1 1
0 0 0 0 0

(8)(b) $9 \div 3$

$9 = 1001$

$3 = 11$

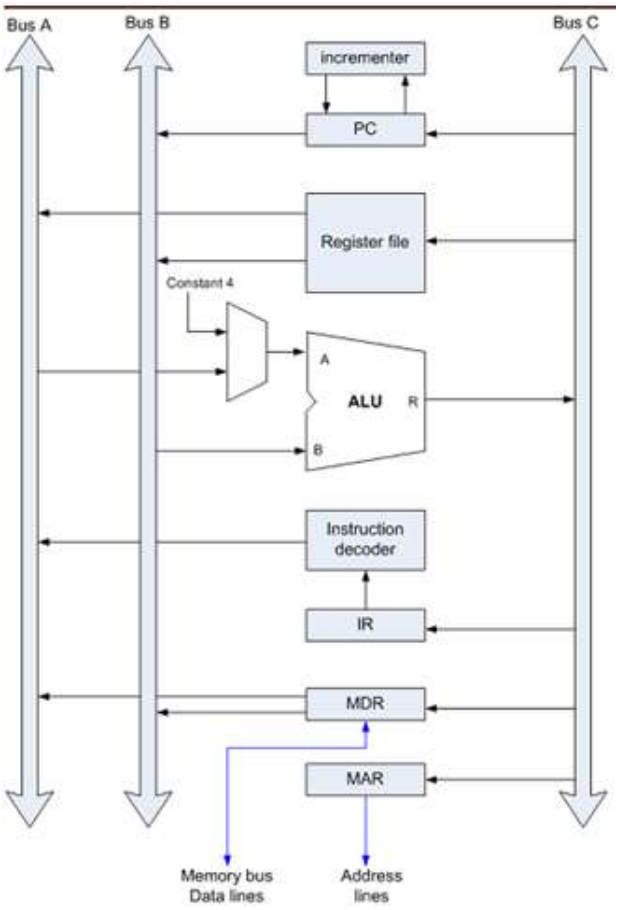
	A	Q	
Initially	00000	1001	
M	00011		
shift	00001	001□	
subtract	11101		first cycle
set q_0	11110		
Restore	00011		
	① 00001	001[0]	
shift	00010	010□	
subtract	11101		
set q_0	11111		second cycle
Restore	00011		
	① 00000	010[0]	
shift	00100	100□	
subtract	11101		
set q_0	① 00001		3rd cycle
		100[1]	
shift	00011	001□	
subtract	11101		
set q_0	① 00000		4th cycle.
		001[1]	

∴ Remainder is 00000 and Quotient is 0011

9.a Draw and explain multiple bus organization. Explain its advantages.

- In single bus structure, control sequences are quite long because only one data item can be transferred over the bus in a clock cycle.
- To reduce the number of steps needed most commercial processors provide multiple internal paths that enable several transfers to take place in parallel.

Solution:



9.B Write and explain the control sequence for execution of unconditional branch instruction.

Step	Action
1	PC_{out} , MAR $_{in}$, Read, Select4,Add, Z_{in}
2	Z_{out} , PC_{in} , Y_{in} , WMFC
3	MDR $_{out}$, IR_{in}
4	Offset-field-of-IR_{out} , Add, Z_{in}
5	Z_{out} , PC_{in} , End

1

Execution of Branch Instructions

- The processing starts as usual, the fetch phase ends in step 3.
- In step 4, offset-value is extracted from IR by instruction-decoding circuit.
- Since updated value of PC is already available in register Y, offset X is gated onto the bus, and an addition operation is performed.
- In step 5, the result, which is the branch address, is loaded into PC.

10.A Draw the block diagram of the control unit organization and describe.

- To execute instructions, the processor must have some means of generating the control signals needed in the proper sequence.
- Two categories: hardwired control and microprogrammed control
- Hardwired system can operate at high speed; but with little flexibility.

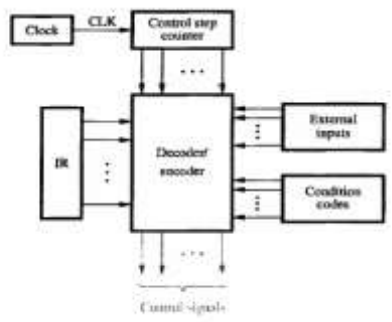


Figure 7.10 Control unit organization.

10.B Explain basic idea of instruction pipelining.

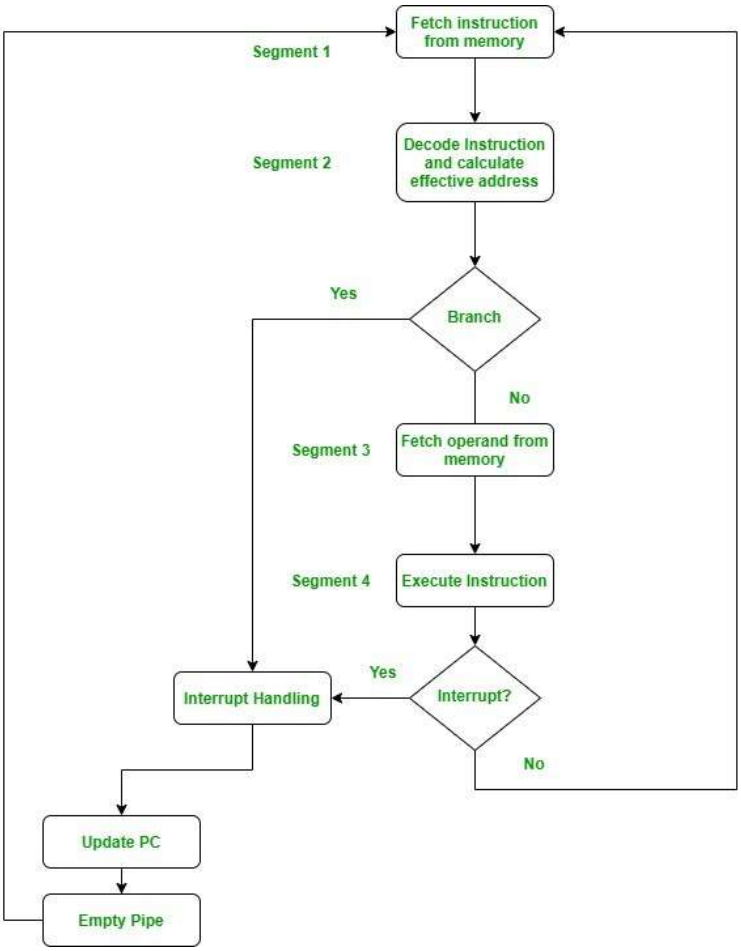
Instruction Pipeline :

In this a stream of instructions can be executed by overlapping fetch, decode and execute phases of an instruction cycle. This type of technique is used to increase the throughput of the computer system. An instruction pipeline reads instruction from the memory while previous instructions are being executed in other segments of the pipeline. Thus we can execute multiple instructions simultaneously. The pipeline will be more efficient if the instruction cycle is divided into segments of equal duration.

In the most general case computer needs to process each instruction in following sequence of steps:

1. Fetch the instruction from memory (FI)
2. Decode the instruction (DA)
3. Calculate the effective address
4. Fetch the operands from memory (FO)
5. Execute the instruction (EX)
6. Store the result in the proper place

The flowchart for instruction pipeline is shown below.


Fetch instruction from memory
Segment 1
Decode Instruction and calculate effective address
Segment 2
Yes
Branch
No
Segment 3
Fetch operand from memory
Segment 4
Execute Instruction
Yes
Interrupt Handling
Interrupt?
No
Update PC
Empty Pipe