CMR
INSTITUTE OF
TECHNOLOGY

VTU Examination December 2017

Sub:	Computer Organization							Code:	15CS34
Date:	30 / 12 / 2017	Duration:	3 hours	Max Marks:	80	Sem:	3	Branch:	CSE/ISE

Answer any FIVE FULL questions, choosing ONE full question from each module

		Marks	OBE	
			CO	RBT
	MODULE 1			
1 (a)	List the steps needed to execute the machine instruction ADD LOC,R0 in terms of transfers between the components and some simple control commands. Assume that the instruction itself is stored in the memory at location INSTR and that this address is initially in register PC. The first two steps might be expressed as: • Transfer the contents of register PC to register MAR. • Issue a Read Command to the memory and then wait until it has transferred the requested word into register MDR. Remember to include the steps needed to update the contents of PC from INSTR to INSTR+1 so that the next instruction can be fetched. • Transfer the contents of register PC to register MAR • Issue a Read command to memory, and then wait until it has transferred the requested word into register MDR • Transfer the instruction from MDR into IR and decode it • Transfer the address LOCA from IR to MAR • Issue a Read command and wait until MDR is loaded • Transfer contents of MDR to the ALU • Transfer contents of R0 to the ALU • Perform addition of the two operands in the ALU and transfer result into R0 • Transfer contents of PC to ALU • Add 1 to operand in ALU and transfer incremented address to PC	[8]	CO1	L3
(b)	What is performance measurement? Explain the overall SPEC rating for the computer in a program suit. The performance measure is the time taken by computer to execute a given benchmark. A non profit organization called System Performance Evaluation Corporation (SPEC) selects and publishes ~~representative~~	[8]	CO3	L2

representative application programs for different application domains, together with test results for many commercially available computers.

The programs selected range from game playing, compiler and database applications to numerically intensive programs in astrophysics and quantum chemistry.

In each case, the program is compiled for the computer under test, and running time on real computer is measured. Simulation is not allowed. The same program is also compiled and run on one computer selected as a reference.

For SPEC 95, the reference is the SUN SPARCstation 10/40.

For SPEC 2000, reference is an UltraSPARC 10 with 300 MHz UltraSPARC-IIi processor.

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

The test is repeated for all the programs in SPEC suite, and geometric mean of results is computed.

Let $SPEC_i$ be the rating for program i in the suite.

Overall SPEC rating is given by:

$$\text{SPEC rating} = \left(\prod_{i=1}^{n} SPEC_i\right)^{\frac{1}{n}}$$

* Geometric mean:- n^{th} root of product of n values

where n is no. of programs in the suite.

2 (a)	With a relevant figure define the little Endian and big Endian assignments.	[4]	CO1	L2

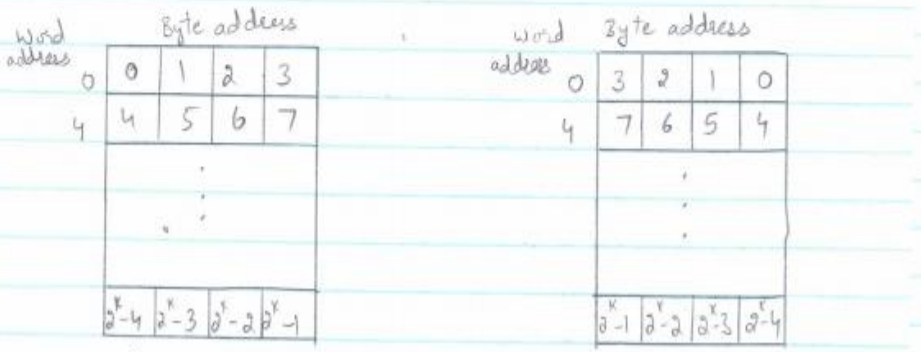
Big-endian and little-endian assignments.

These 2 methods are used for byte addressing. Any one method is selected out of these.

Big-endian assignment - Lower byte addresses are used for more significant bytes (leftmost bytes) of the word.

Little-endian assignment - Lower byte addresses are used for the less significant bytes (rightmost bytes) of the word.

The words 'more significant' and 'less significant' are used in relation to the weights (power of 2) assigned to bits when word represents a number.



(a) Big-endian assignment (b) Little-endian assignment

BYTE & WORD ADDRESSING

(b)	Consider a computer that has a byte addressable memory organized in 32 bit words according to the big Endian scheme. A program reads ASCII characters entered at a keyboard and store them in a successive byte location starting at a location 1000. Show the contents of the two memory words at locations 1000 and 1004 after the name "Johnson" has been entered. (ASCII codes J= 4 AH, o = 6 FH, h = 68 H, n = 6 EH, S = 73 H) (H represents it is a hexadecimal number. Each character takes 1 byte. Each word has 4 bytes or 32 bits. So, location 1000 to 1003 has 4 bytes for storing letters J,o,h,n and, location 1004 to 1007 has 4 bytes to store remaining 3 characters s, o, n) Byte contents in hexadecimal, starting at location 1000 to 1007, will be 4A, 6F, 68, 6E, 73, 6F, 6E. The two words at 1000 and 1004 will be **4A6F686E** and **736F6EXX**.	[4]	CO1	L3
(c)	Write about shift and rotate instruction with neat diagram and example of each.	[8]	CO1	L2

2) Shift and Rotate Instructions.

Shift instructions shift bits of an operand to right or left some specified number of bit positions.

Rotate instructions move the bits that are shifted out of one end of the operand back into the other end.

- Logical Shifts

Shift an operand over a number of bit positions specified in a count operand contained in the instruction. Count operand may be given as an immediate operand or in may be contained in a processor register.

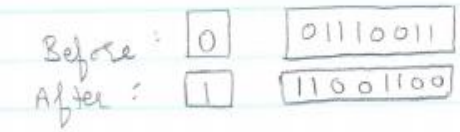
Syntax: LShiftL count, dst
LShiftR count, dst

Bits shifted out are passed through carry flag, C & then dropped.

→ Logical Shift Left

Eg-

LShiftL #2, R0



Logical Shift Left

→ Logical Shift Right.

Eg-

LShiftR #2, R0

Logical Shift Right

- Arithmetic Shifts

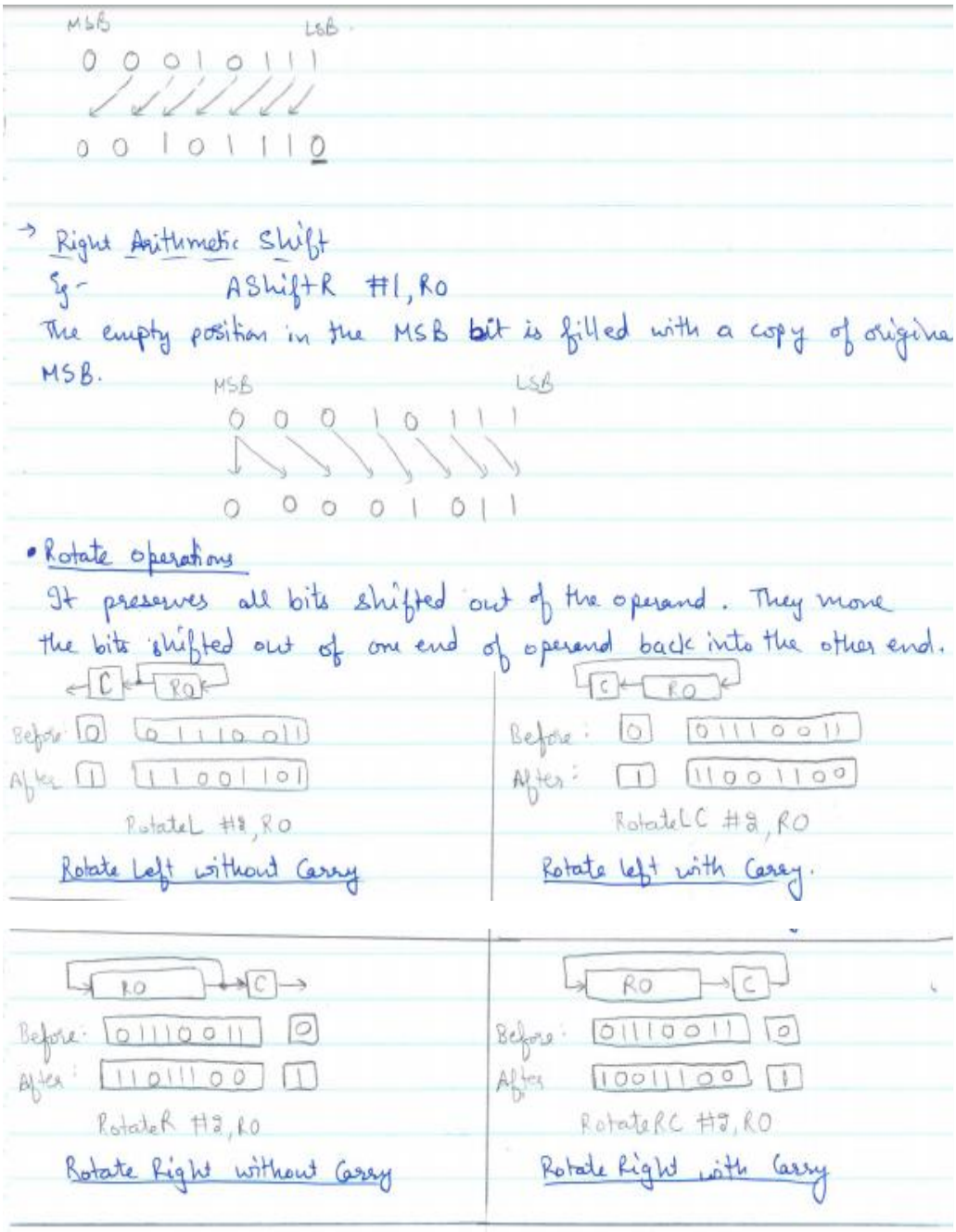
Fillin bits are not always zero as in logical shift operations.

Syntax: AShiftL count, dst
AShiftR count, dst

→ Left Arithmetic Shift

Eg- AShiftL #1, R0

The empty position in the LSB (least significant bit) is filled with a zero.



MODULE 2

3 (a) With supporting diagram, explain the following with respect to interrupts: [6] CO2 L2

i) Vectored interrupts
ii) Interrupt Nesting
iii) Simultaneous interrupts.

i) Vectored interrupts

② VECTORED INTERRUPTS

• Device requesting an interrupt identifies itself directly to the processor.

(4 to 8 bits)

The device sends a special code to the processor over the bus.

- The code contains :
  identification of the device,
  starting address of ISR,
  address of the branch to ISR (if ISR not at that location).

The location pointed to by the interrupting device is used to store the starting address of the interrupt service routine. This address is called interrupt vector. Processor reads it and loads it into PC.

* When the processor is ready to receive interrupt-vector code, it may activate interrupt-acknowledge line, INTA. The I/O device responds by sending its interrupt-vector code and turning off the INTR signal.

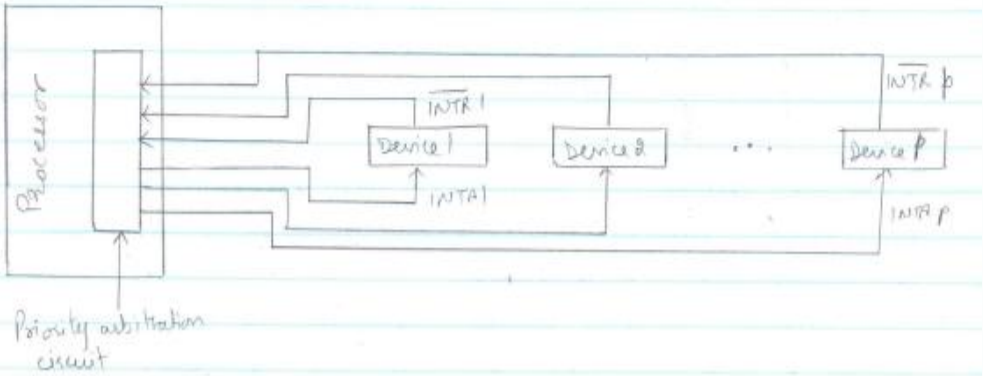
ii) Interrupt Nesting

② INTERRUPT NESTING

- Disabling Interrupts during execution of the ISR may not favor devices which need immediate attention. eg, keeping track of time of day.
- Pre-emption of low priority interrupt by another higher priority interrupt is known as Interrupt nesting.
- Only interrupts requests of higher priority will be accepted during execution of ISR of lower priority interrupt.
- A priority level is assigned to process which is the priority of the program that is currently being executed. Only higher priority interrupts than this are accepted.
- Processor's priority is encoded in a few bits of processor

status word which can be changed by program instructions called privileged instructions, which can be executed only while processor is running in supervisor mode (i.e. when executing OS routines).

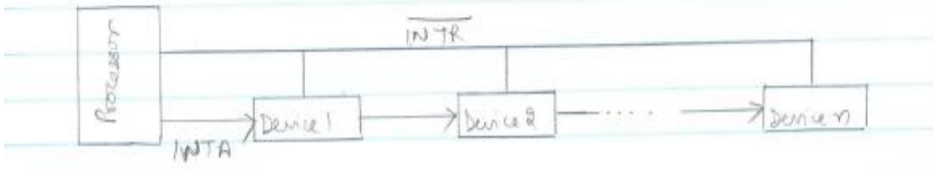
- An attempt to execute a privileged instruction while in user mode leads to a special type of interrupt called a privilege exception.
- A multiple-priority scheme can be implemented by using separate interrupt request and interrupt-acknowledge lines for each device. Each interrupt-request lines is assigned a different priority level. Interrupt requests received over these lines are sent to a priority ~~arbi~~ arbitration circuit in the processor. A request is accepted only if it has a higher priority level than that currently assigned to the processor.



iii) Simultaneous requests

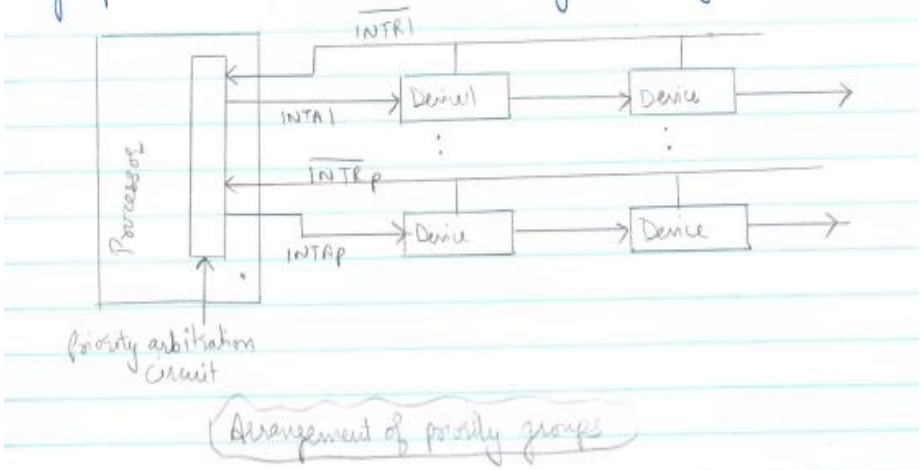
It is same as daisy chaining as daisy chaining handles it.

- The interrupt request line INTR is common to all the devices.
- The interrupt acknowledgement line INTA is connected to devices in a daisy chain way.
- INTA propagates serially through the devices.
- Device that is electrically closest to the processor gets high priority.
- Low priority device may have a danger of starvation.



Daisy chaining with priority group :-

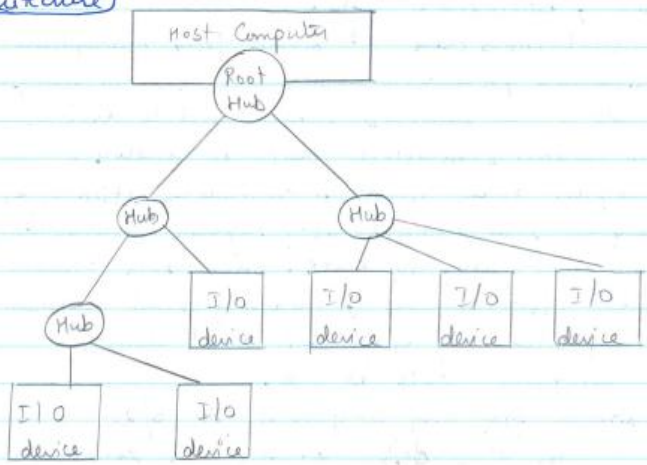
- Combining daisy chaining and interrupt nesting to form priority group.
- Each group has different priority levels and within each group devices are connected in daisy chain way.



Arrangement of priority groups

(b)	Three devices A, B and C are connected to the bus of the 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: i) The computer has only one interrupt request line. ii) Two interrupt request line, INTR1 and INTR2 are available with INTR1 having higher priority. Specify when and how interrupts are enabled and disabled in each case. 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. ii) A and B should be connected to INTR2, and C to INTR1. 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.	[6]	CO2	L3
(c)	Illustrate the tree structure of USB with diagram.	[4]	CO2	L2

USB Architecture



USB Tree Structure

USB is a serial bus that satisfies low cost (less wires flexibility (long distance) requirements. Clock and data information are encoded together and transmitted as a signal. To provide high data transfer bandwidth it high clock frequency. It offers 3 speeds of operation low speed (1.5 megabits/s), full speed (12 megabits USB 2.0 high-speed (480 megabits/s).

USB has tree structure. Each node of the tree has a called hub, which acts as an intermediate control point be the host & I/o devices. Root hub connects the entire to the host computer. Leaves of the tree (I/O devices) are functions. Each hub has a number of ports where may be connected, including other hubs.

Hub copies a message that it receives from its upstrea connection to all its downstream ports (i.e broadcast all I/o devices), but only the addressed device will resp to that message. A message from an I/o device is sen only upstream towards the root of the tree and is seen by other devices. Means, it allows host to c device communication but doesnot allow devices to communicate with each other.

The tree makes it possible to connect large no. of devices to computer though a few ports.

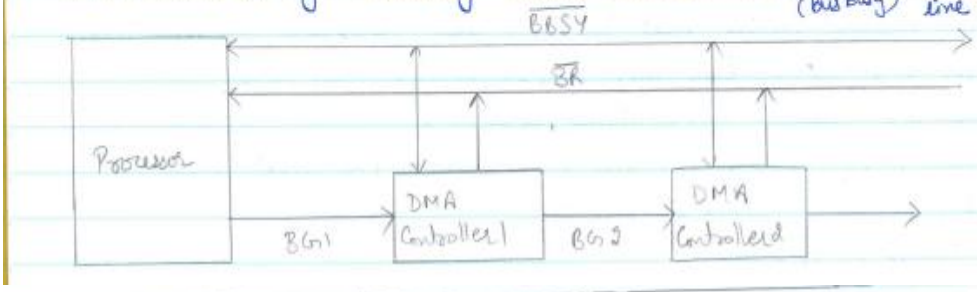
USB operates on polling. A device may send a mes only in response to a poll message from the ho So, no two devices can send messages at same ti

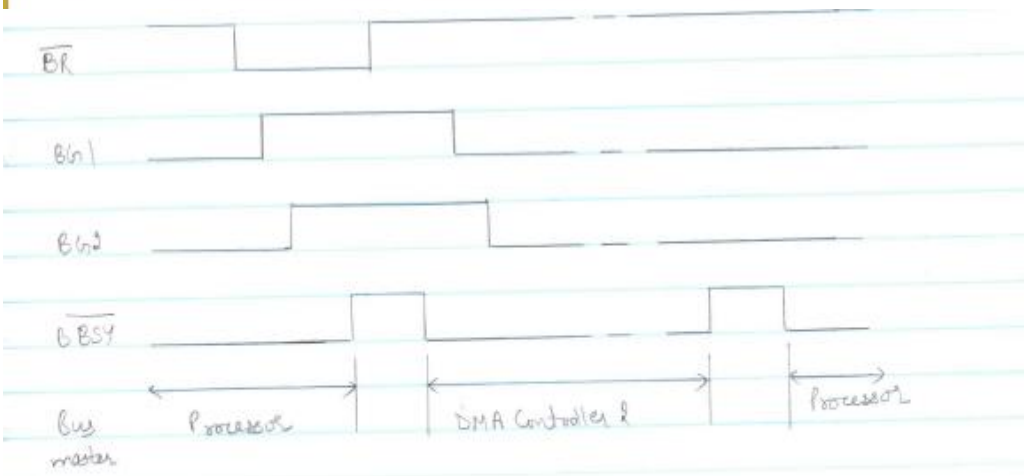
OR

4 (a)	With a neat diagram, explain the centralized arbitration and distributed bus arbitration scheme.	[8]	CO2	L2

- Centralized Bus Arbitration Technique

→ A 'single bus arbiter' performs the required arbitration.
→ Bus arbiter may be the processor or a separate device connected to the bus (eg. DMA controller having highest priority).
→ Initially the processor will act as 'bus master'.
→ Whenever a DMA controller generates a request ($\overline{BR}$), by activating Bus Request line, processor activates Bus-Grant (BG1) signal indicating DMA controller can become a master
→ BG line is connected to all DMA devices using a daisy chain link.
→ If DMA controller 1 has enabled the request, it blocks the signal (BG1) and acts as a master for bus arbitration, thereby disabling the bus for other device use.
→ If DMA controller 1 has not enabled bus arbitration request, it simply forwards BG1 signal to its downstream neighbour DMA controllers by asserting BG2. New bus master activates $\overline{BBSY}$ (Bus Busy) line.



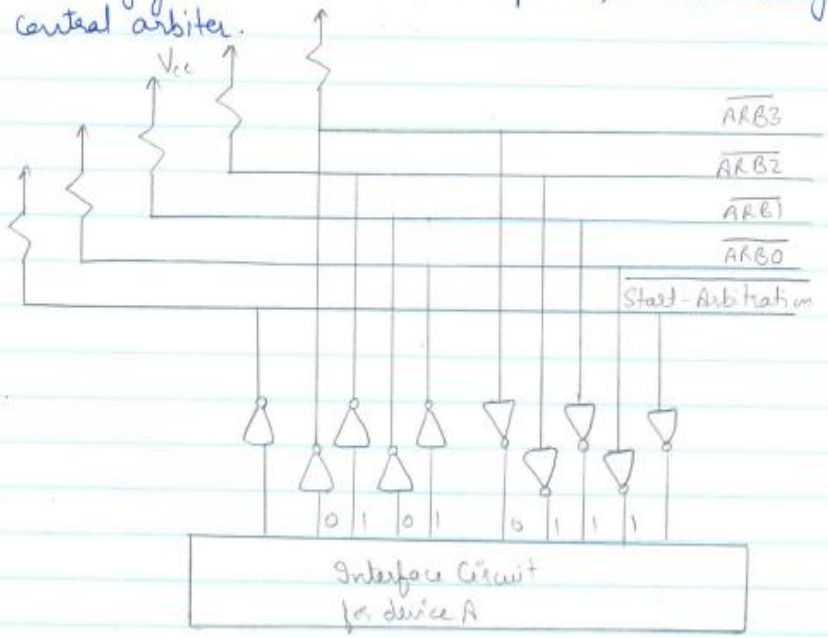


Sequence of signals for Transfer of bus mastership from processor to DMA controller 2

DMA controller 2 requests and acquires the bus mastership of the bus. At this time, it activates the bus busy line to prevent other devices from using the bus at same time. After it finishes its task, when bus is free again, processor acquires the bus if no other DMA controller request is present and activates bus busy line.

Distributed Bus Arbitration

- All devices waiting to use the bus have equal responsibility in carrying out the arbitration process, without using a central arbiter.



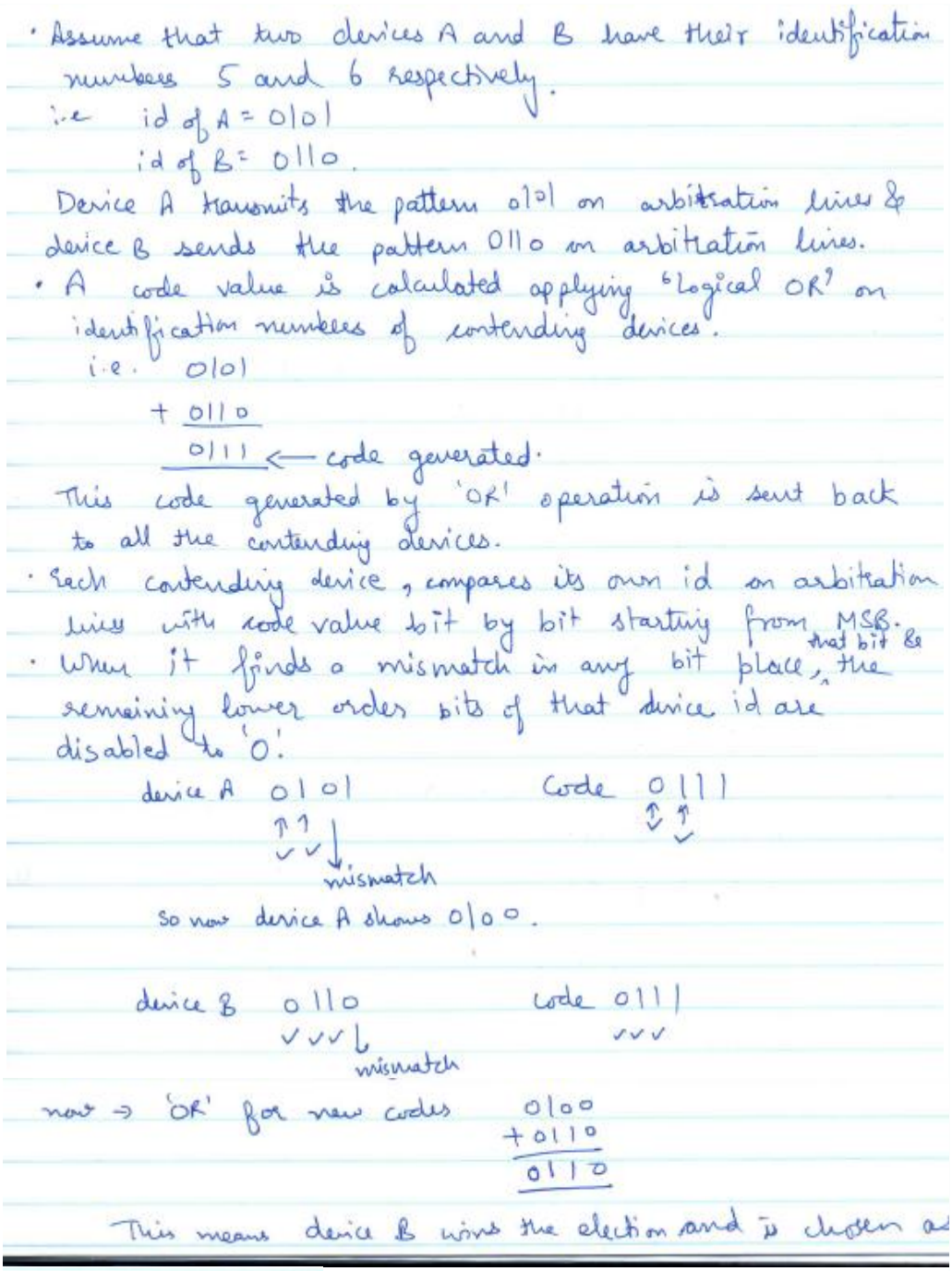
Distributed arbitration scheme

→ Each device is identified by using a 4bit identification number.

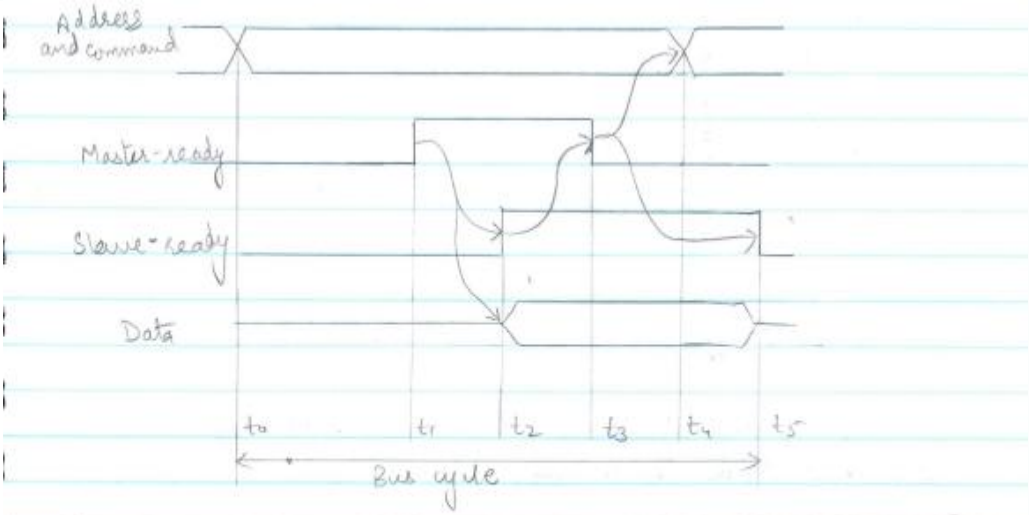
→ Devices start contending for bus by enabling 'Start-arbitration' signal and place their 4-bit identification ~~number~~ on the bus (4 lines $ARB_0 - ARB_3$).

→ Device having highest identification number is selected to get the granted service.

→ Selection procedure:



(b) With neat timing diagram illustrate the asynchronous bus data transfer during an input operation. Use handshake scheme. [8] CO2 L2



Handshake control of data transfer during an input operation

Asynchronous bus - Input

- Data transfer is controlled by a handshake protocol.
- Instead of using common clock, it uses 2 timing control lines named master-ready and slave-ready.
- The master places the address and command information on the bus, and activates Master-ready line at t_1.
- All devices on the bus decode the address and the selected slave informs the master by setting slave-ready signal line after doing required operation. At t_2, selected slave does this. For input operation, it places data from its register on the data lines. It sets slave-ready signal to 1.
- At t_3, slave-ready signal arrives at master indicating the availability of data on the bus. After doing some settings, master strobes the data into its input buffer and drops master-ready signal indicating it has received the data.
- At t_4, master removes the address and command information from the bus.
- At t_5, the device interface observes 1 to 0 transition of master ready signal and removes data and slave ready signals from the bus.

This completes the input transfer. * Delays are caused due to bus skew.

MODULE 3

5 (a) Draw a diagram and explain the working of 16 Megabit DRAM chip configured as 2M x 8. [8] CO5 L3

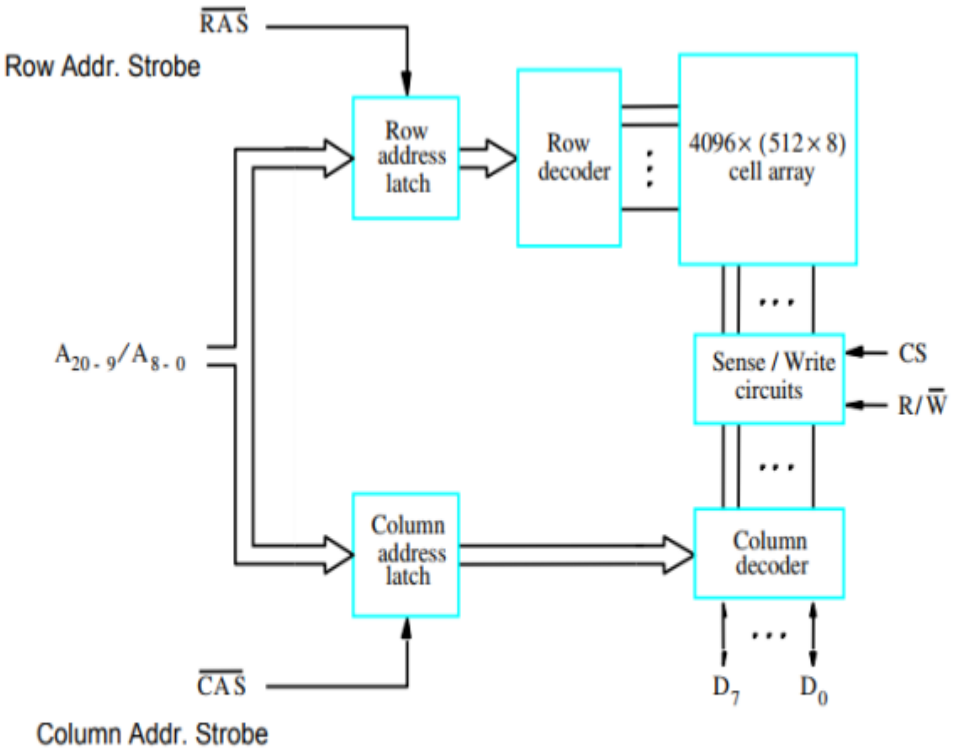


Figure 5.7. Internal organization of a 2M × 8 dynamic memory chip.

In above figure, a 16-megabit (4x4x1024x1024).

DRAM chip is configured as 2M x 8. The cells are organized in the form of 4K x 4K array. The 4096 cells in each row are divided into 512 groups of 8, so that a row can store 512 bytes of data.

- 21 bit address is needed to access a byte in the memory. 21 bit is divided as follows:
  1) 12 address bits are needed to select a row.
     i.e. $A_{8\text{-}0}$ → specifies row-address of a byte.
  2) 9 bits are needed to specify a group of 8 bits in the selected row.
     i.e. $A_{20\text{-}9}$ → specifies column-address of a byte.

- During Read/Write-operation,
  → row-address is applied first.
  → row-address is loaded into row-latch in response to a signal pulse on **RAS'** input of chip.
  (RAS = Row-address Strobe CAS = Column-address Strobe)
- When a Read-operation is initiated, all cells on the selected row are read and refreshed.
- Shortly after the row-address is loaded, the column-address is

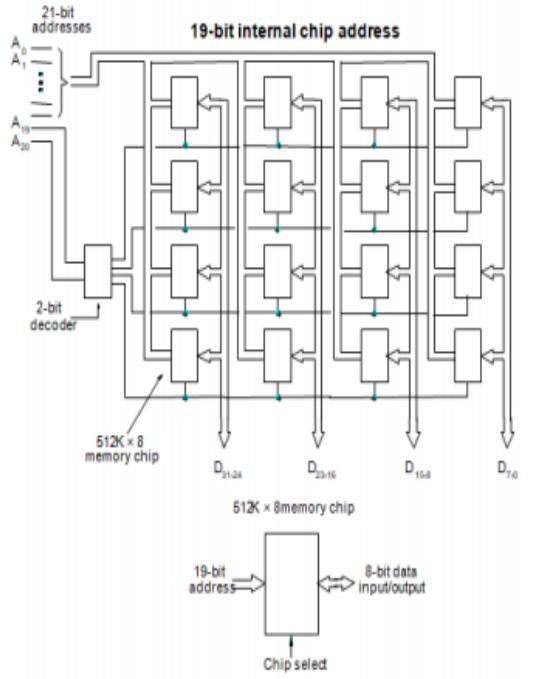
->Applied to the address pins and loaded into Column address latch under control of CAS signal.

- The information in the latch is decoded.
- The appropriate group of 8 Sense/Write circuits is selected.
  R/W'=1(read-operation) → Output values of selected circuits are transferred to data-lines D_0-D_7.
  R/W'=0(write-operation) → Information on D_0-D_7 are transferred to the selected circuits.
- RAS' & CAS' are active-low so that they cause latching of address when they change from high tc low.
- To ensure that the contents of DRAMs are maintained, each row of cells is accessed periodically.
- A special memory-circuit provides the necessary control signals RAS' & CAS' that govern the timing.
- The processor must take into account the delay in the response of the memory.

Fast Page Mode

- Transferring the bytes in sequential order is achieved by applying the consecutive sequence of column-address under the control of successive CAS' signals.
- This scheme allows transferring a block of data at a faster rate.
- The block of transfer capability is called as *fast page mode*.

(b)	Describe organization of an 2M x 32 memory using 512K x 8 memory chips.	[8]	CO5	L3
6 (a)	Discuss in detail the working of set associative mapped cache with two blocks per set with relevant diagram.	[8]	CO5	L2



- *Implement a memory unit of 2M(2097152) words of 32 bits each.*
- *Use 512x8 static memory chips.*
- *Each column consists of 4 chips.*
- *Each chip implements one byte position.*
- *A chip is selected by setting its chip select control line to 1.*
- *Selected chip places its data on the data output line, outputs of other chips are in high impedance state.*
- *21 bits to address a 32-bit word.*
- *High order 2 bits are needed to select the row, by activating the four Chip Select signals.*
- *19 bits are used to access specific byte locations inside the selected chip.*

SET-ASSOCIATIVE MAPPING

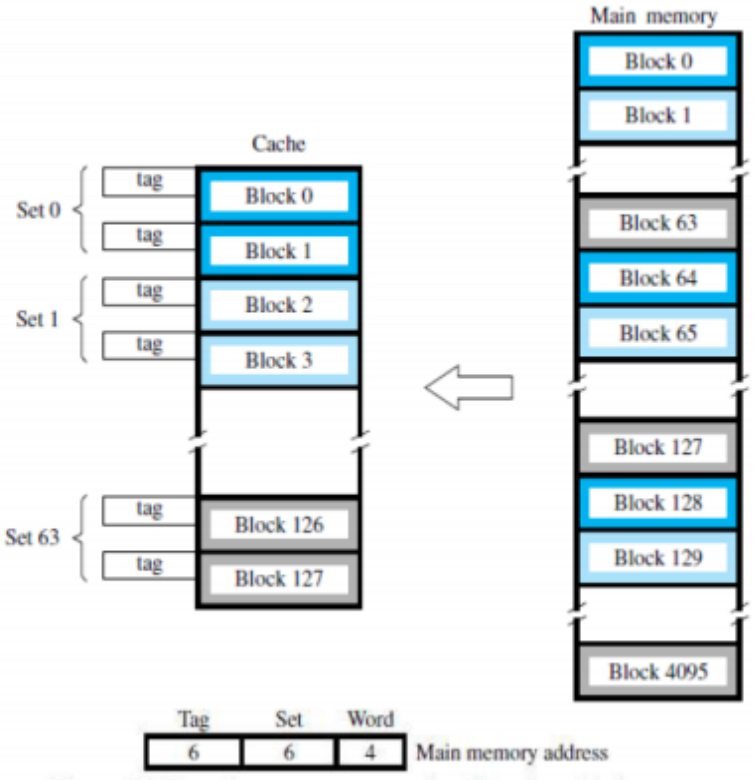


Figure 8.18 Set-associative-mapped cache with two blocks per set.

It is a **combination** of direct mapping and associative mapping techniques.
Blocks of the cache are grouped into **sets**, and the mapping allows a block of the main memory to reside in **any block of a specific set**.
Contention problem of direct mapping is eased by having a few choices for block placement. Hardware cost is reduced by decreasing the associative search.
For example, a cache with two blocks per set. Memory blocks 0, 64, 128, ..., 4032 map into cache set 0, and they can occupy either of the two block positions within this set. 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 field of the address must then be associatively compared to the tags of the two blocks of the set to check if the desired is present. This is two-way associative search.
The number of blocks per set is a parameter that can be selected to suit the requirements of a particular computer. 4 bocks per set need 5-bit set field, 8 blocks per set need 4-bit set field and so on.
128 blocks per set does not require set bits and is known as **full associative** technique.
One block per set is termed as direct mapping. Cache having k blocks per set is referred to as **a k-way-set-associative** cache.
Each block contains a control bit called **valid bit** (different from dirty/modified bit). The valid bits are all set to 0 when power is initially applied to the system or when the main memory is loaded with new programs and data from the disk. The valid bit of a particular cache block is set to 1 the first time this block is loaded from main memory. Whenever a main memory block is updated by a source that bypasses the cache (e.g. DMA transfer), a check is made to determine whether the block being loaded is currently in the cache. If it is, the valid bit is cleared to 0. This ensures that the stale data does not exist in the cache.

(b)	Define the following with respect to cache memory: (i) Valid bit, (ii) Dirty data, (iii) Stale data, (iv) Flush the cache.	[4]	CO2	L1

(i) Valid bit: A bit of information that indicates whether the data in a block is valid (1) or not (0)
(ii) Dirty data: The data in the cache is called dirty data , if it is modified within cache but not modified in main memory.
(iii) Stale data: The data in the cache is called stale data if it is modified in the main memory but the is not updated in cache.

Flush the cache: Forcing dirty data to be written back to the memory before DMA transfer takes place.

(c)	A block-set associative cache consists of a total of 64 blocks divided into 4-blocks sets. The main memory contains 4096 blocks, each consisting of 128 words.	[4]	CO5	L3

i) How many bits are there in a main memory address?
ii) How many bits are there in each of the TAG, SET and the WORD fields?

i) 4096 blocks of 128 words each require 12+7 = 19 bits for the main memory address.
ii)TAG field is 8 bits. SET field is 4 bits. WORD field is 7 bits

MODULE 4

7 (a)	Convert the following pairs of decimal numbers to 5-bit signed 2's complement binary numbers and add them. State whether or not overflow	[4]	CO3	L3

occurs in each case.

i) 5 and 10 ii) -14 and 11 iii) -5 and 7 iv) -10 and -13

i)

```
  00101
+ 01010
-------
  01111
```

No overflow

ii)

```
  10010
+ 01011
-------
  11101
```

No overflow

iii)

```
  11011
+ 00111
-------
  00010
```

No overflow

iv)

```
  10110
+ 10011
-------
  01001
```

Overflow

(b) Design the 16 bit carry look ahead adder using 4-bit adder. Also, unite the expression for C_{i+1}. [8] CO3 L3

$S_i = x_i \oplus y_i \oplus c_i$

$C_{i+1} = x_i y_i + x_i c_i + y_i c_i$

$\Rightarrow C_{i+1} = x_i y_i + (x_i + y_i) c_i$

~~$C_{i+1} = x_i y_i$~~

$C_{i+1} = G_i + P_i c_i$

where

$G_i = x_i y_i$ → Generate function

$P_i = x_i + y_i$ → Propagate function

Carry signal is generated if

- $x_i = 1$ and $y_i = 1$ i.e, $G_i = 1$

or

- $c_{i-1} = 1$ and $(x_i = 1$ or $y_i = 1)$

→ G_i is implemented using AND gate

→ P_i is implemented using XOR gate instead of OR gate. i.e. $P_i = x_i \oplus y_i$ instead of $P_i = x_i + y_i$. Because, only $x_i = y_i = 1$ the output will differ, but in that case $G_i = 1$, so it doesn't matter if $P_i = 0$ or 1.

All G_i and P_i functions can be formed independently and in parallel.

$$C_{i+1} = G_i + P_i C_i$$

$$C_i = G_{i-1} + P_{i-1} C_{i-1}$$

Substituting C_i

$$C_{i+1} = G_i + P_i (G_{i-1} + P_{i-1} C_{i-1})$$

$$= G_i + P_i G_{i-1} + P_i P_{i-1} C_{i-1}$$

Continuing ..... final expression of C_{i+1} in terms of C_0 is

$$C_{i+1} = G_i + P_i G_{i-1} + P_i P_{i-1} G_{i-2} + \dots + P_i P_{i-1} \dots P_1 G_0 +$$

$$\dots P_i P_{i-1} \dots P_0 C_0$$

Consider the design of 4-bit adder.

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$C_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

$$C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0$$

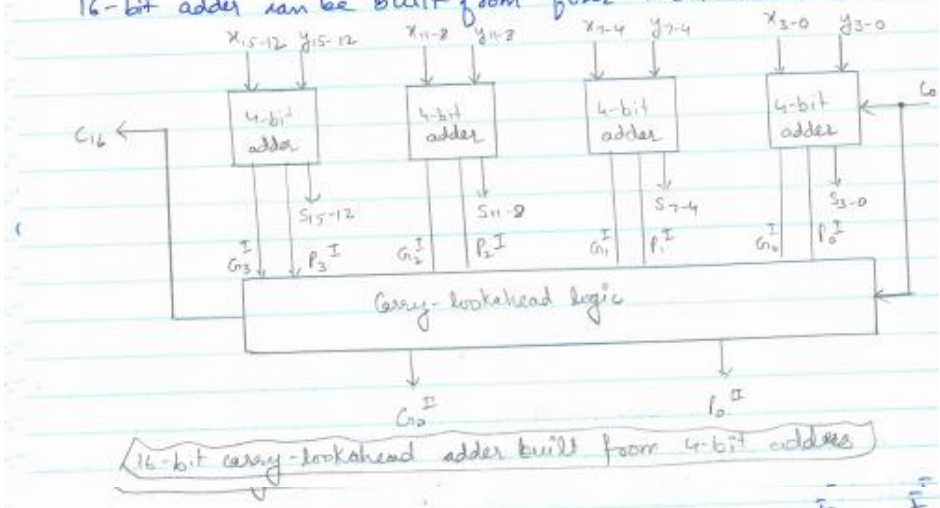
Each carry signal is expressed as a direct Sum of Product (SOP) of C_0 rather than its preceding carry signal.

For C_4 in the 4-bit adder, a fan-in of 5 is required. This is about the limit for practical gates. So, carry lookahead adder cannot be directly extended to larger operand sizes. However, if we cascade a number of 4-bit adders, we can build longer adders. But it would lead to delay in calculating carries of C_4, C_8, C_{12}, C_{16} ... and so on.

The carries C_4, C_8, C_{12} ... ripple through 4-bit adder blocks with 2 gate delays per block, analogous to the way that individual carries ripple through each bit stage in a ripple-carry adder. By using higher-level block generate and propagate functions, it is possible to use the lookahead approach to develop the carries C_4, C_8, C_{12} ... in parallel, in a higher-level carry-lookahead circuit.

Higher-level Generate and Propagate functions

16-bit adder can be built from four 4-bit adder blocks.



16-bit carry-lookahead adder built from 4-bit adders

These blocks provide new output functions defined as G_K^I and P_K^I

where K=0 for first 4 bit block,
K=1 for second 4-bit block and so on

- In first block,

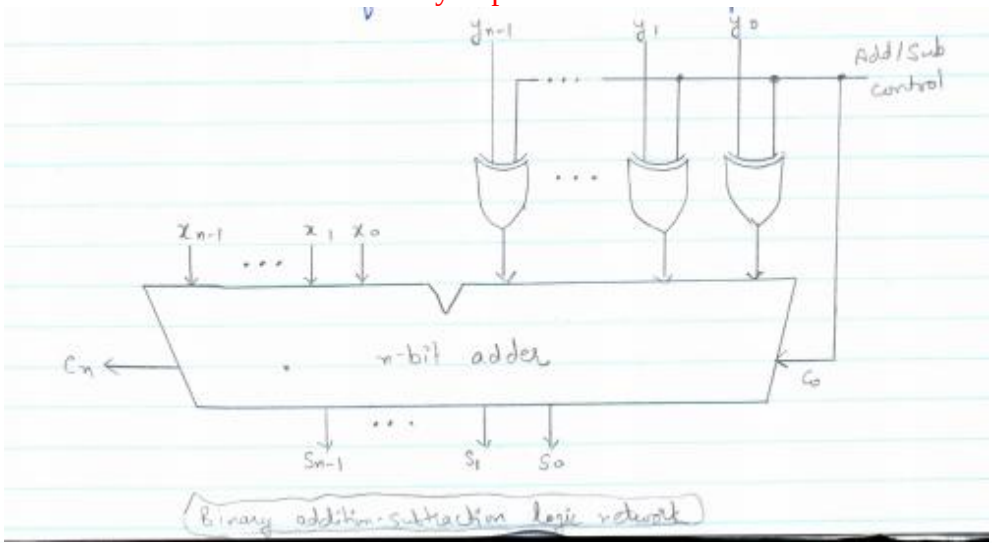
$$P_0^I = P_3 P_2 P_1 P_0$$

$$G_0^I = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0$$

- The first-level G_i and P_i functions determine whether bit stage i generates or propagates a carry, and second level G_K and P_K functions determine whether block K generates or propagates a carry.
- carry C_{16} is formed by one of the carry-lookahead circuits as

$$C_{16} = G_3^I + P_3^I G_2^I + P_3^I P_2^I G_1^I + P_3^I P_2^I P_1^I G_0^I + P_3^I P_2^I P_1^I P_0^I C_0$$

(c) Draw the two n-bit number x and y to perform addition/subtraction. [4] CO3 L2



Binary addition-subtraction logic network

Overflow = $x_{n-1}\, y_{n-1}\, \bar{s}_{n-1} + \bar{x}_{n-1}\, \bar{y}_{n-1}\, s_{n-1}$

Overflow occurs when the carry bits c_n and c_{n-1} are different. Therefore, overflow can be obtained by implementing the expression $c_n \oplus c_{n-1}$ with XOR gate.

For subtraction X−Y on 2's-complement numbers X and Y, we form the 2's complement of Y and add it to X.

The network for addition/subtraction is shown below. Add/Sub input control line is set to 0 for addition & $c_0=0$. (applying Y unchanged). This line is set to 1 for subtraction (Y is complemented) and $c_0=1$.

OR

8 (a) With an example explain the Booths algorithm to multiple two signed operands. [8] CO3 L2

(Any example can be taken)

The Booth algorithm generates a 2n-bit product and treats both positive and negative 2's-complement n-bit operands uniformly.

Steps

(i) Assume that an implied 0 lies to the right of the LSB of multiplier.

(ii) −1 times the shifted multiplicand is selected when moving from 0 to 1, +1 times multiplicand is selected when moving from 1 to 0, as multiplier is scanned from right to left. i.e,

Multiplier Bit i	Bit i−1	Version of multiplicand selected by bit i
0	0	0 × M
0	1	+1 × M
1	0	−1 × M → (2's complement of multiplicand)
1	1	0 × M

Eg- Using booth algorithm, multiply 13 with −6.

+13 → 0 1 1 0 1

−6 → 2's complement of 0 0 1 1 0 is 1 1 0 1 0

0 −1 +1 −1 0

now,

0 1 1 0 1

0 −1 +1 −1 0

0 0 0 0 0 0 0 0 0 0

1 1 1 1 1 0 0 1 1

0 0 0 0 1 1 0 1

1 1 0 0 1 1

0 0 0 0 0 0

1 1 1 0 1 1 0 0 1 0 (−78) in 2's complement.

Advantages

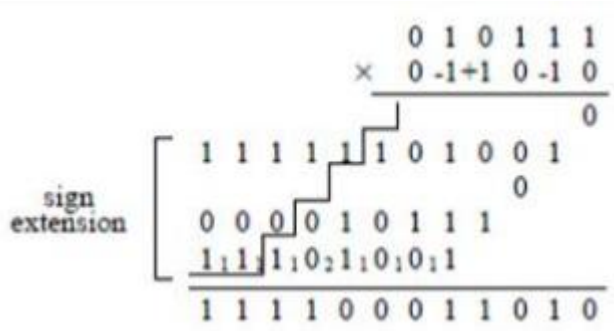
- It handles both positive and negative multipliers uniformly.
- It is more efficient for no. of additions required when the multiplier has a few large block of 1s.

{ When multiplier has its 1s grouped into a few contiguous blocks, only a few versions of the shifted multiplicand (the summands) must be added to generate the product, thus speeding up the multiplication operation. }

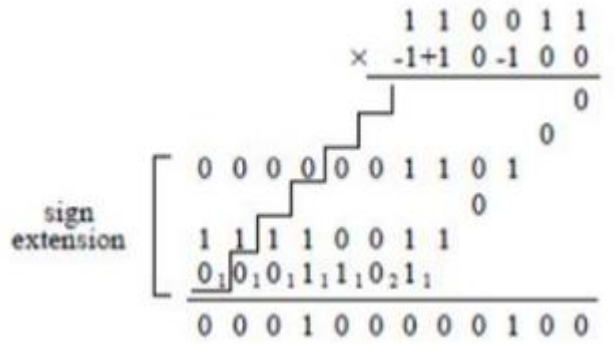
On average, the speed of doing multiplication with the Booth algorithm is the same as with the normal algorithm.

(b)	Multiply each of the following pairs of signed 2's complement number using the Booth algorithm. (A= multiplicand and B= multiplier) i) A= 010111 and B= 110110 ii) A= 110011 and B= 101100 iii) A= 110101 and B= 011011 iv) A= 001111 and B= 001111	[8]	CO3	L3

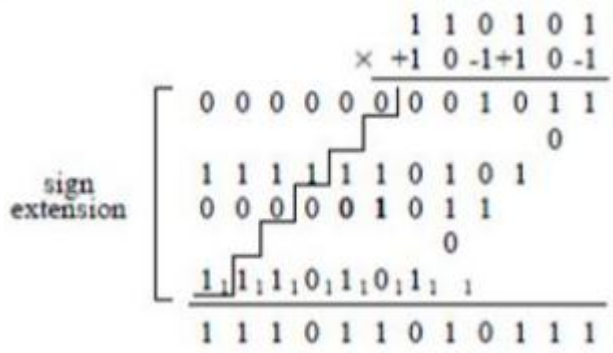
i)



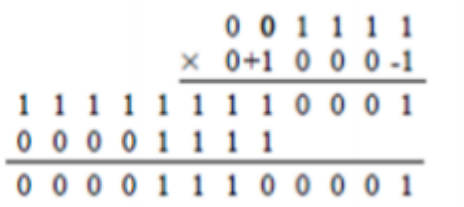
ii)



iii)

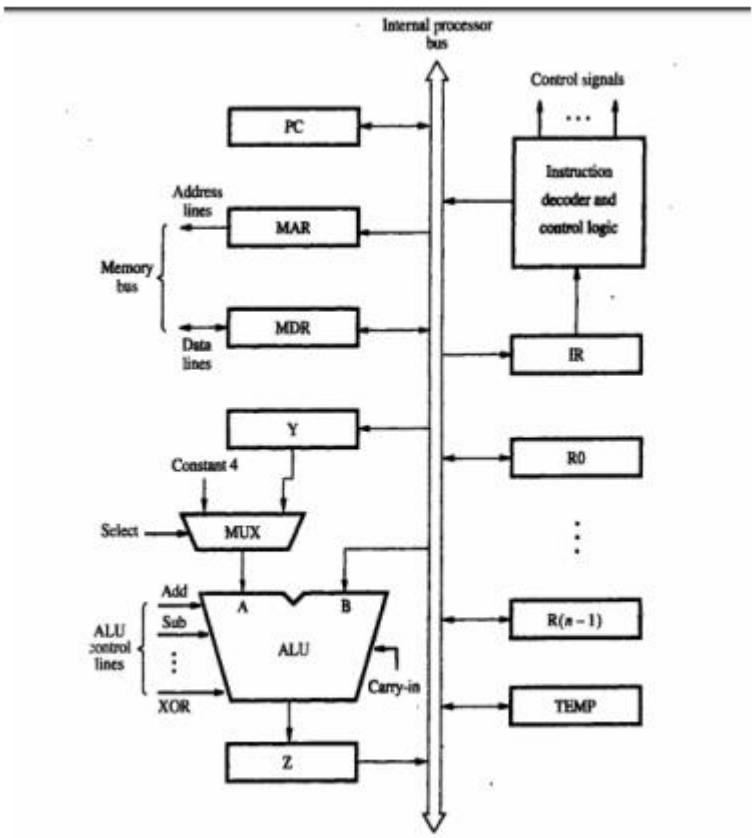


iv)



MODULE 5

9 (a)	Discuss with neat diagram, the single bus organization of the data path inside a processor.	[8]	CO3	L2



SINGLE BUS ORGANIZATION

- ALU and all the registers are interconnected via a **Single Common Bus** (Figure 7.1).
- Data & address lines of the external memory-bus is connected to the internal processor-bus via MDR & MAR respectively. (MDR→ Memory Data Register, MAR → Memory Address Register).
- **MDR** has 2 inputs and 2 outputs. Data may be loaded
  - → into MDR either from memory-bus (external) or
  - → from processor-bus (internal).
- **MAR**'s input is connected to internal-bus;
  MAR's output is connected to external-bus.
- **Instruction Decoder & Control Unit** is responsible for
  - → issuing the control-signals to all the units inside the processor.
  - → implementing the actions specified by the instruction (loaded in the IR).
- Register R0 through R(n-1) are the **Processor Registers**.
  The programmer can access these registers for general-purpose use.
- Only processor can access 3 registers **Y**, **Z** & **Temp** for temporary storage during program-execution.
  The programmer cannot access these 3 registers.
- In **ALU**, 1) 'A' input gets the operand from the output of the multiplexer (MUX).
  2) 'B' input gets the operand directly from the processor-bus.
- There are 2 options provided for 'A' input of the ALU.
- MUX is used to select one of the 2 inputs.
- **MUX** selects either
  - → output of Y or
  - → constant-value 4(which is used to increment PC content).

- An instruction is executed by performing one or more of the following operations:
  1) Transfer a word of data from one register to another or to the ALU.
  2) Perform arithmetic or a logic operation and store the result in a register.
  3) Fetch the contents of a given memory-location and load them into a register.
  4) Store a word of data from a register into a given memory-location.
- **Disadvantage:** Only one data-word can be transferred over the bus in a clock cycle.
  Solution: Provide multiple internal-paths. Multiple paths allow several data-transfers to take place in parallel.

(b) Write the sequence of control steps required for single bus structure for each if the following instructions: [8] CO3 L3

i) Add the contents of memory location NUM to register R1.
ii) Add the contents of memory location whose address is at memory location NUM to register R1.

Assume that each instruction consists of two words. (Note: Change of assumption may result in change of control steps)

i)
1. PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}
2. Z_{out}, PC_{in}, Y_{in}, WMFC
3. MDR_{out}, IR_{in}
4. PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}
5. Z_{out}, PC_{in}, WMFC
6. MDR_{out}, MAR_{in}, Read
7. $R1_{out}$, Y_{in}, WMFC
8. MDR_{out}, Add, Z_{in}
9. Z_{out}, $R1_{in}$, End

ii)
1. PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}
2. Z_{out}, PC_{in}, Y_{in}, WMFC
3. MDR_{out}, IR_{in}
4. PC_{out}, MAR_{in}, Read, Select4, Add, Z_{in}
5. Z_{out}, PC_{in}, WMFC
6. MDR_{out}, MAR_{in}, Read, WMFC
7. MDR_{out}, MAR_{in}, Read
8. $R1_{out}$, Y_{in}, WMFC
9. MDR_{out}, Add, Z_{in}
10. Z_{out}, $R1_{in}$, End

OR

10 (a) Discuss the microwave oven with neat block diagram. [8] CO4 L2

Microwave Oven

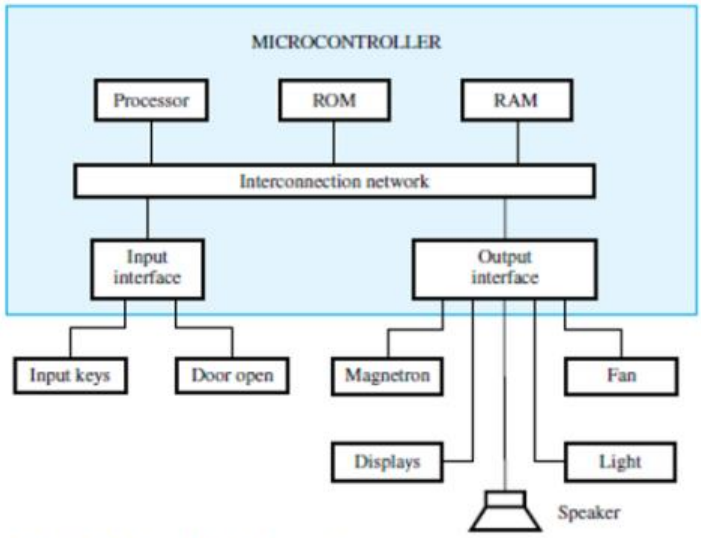


Figure 10.1 A block diagram of a microwave oven.

MICROWAVE OVEN

- Microwave-oven is one of the examples of embedded-system.
- This appliance is based on **magnetron** power-unit that generates the microwaves used to heat food.
- When turned-on, the magnetron generates its maximum power-output.
  Lower power-levels can be obtained by turning the magnetron on & off for controlled time-intervals.
- **Cooking Options** include:
  - → Manual selection of the power-level and cooking-time.
  - → Manual selection of the sequence of different cooking-steps.
  - → Automatic melting of food by specifying the weight.
- **Display (or Monitor)** can show following information:
  - → Time-of-day clock.
  - → Decrementing clock-timer while cooking.
  - → Information-messages to the user.
- **I/O Capabilities** include:
  - → Input-keys that comprise a 0 to 9 number pad.
  - → Function-keys such as Start, Stop, Reset, Power-level etc.
  - → Visual output in the form of a LCD.
  - → Small speaker that produces the beep-tone.
- **Computational Tasks** executed are:
  - → Maintaining the time-of-day clock.
  - → Determining the actions needed for the various cooking-options.
  - → Generating the control-signals needed to turn on/off devices.
  - → Generating display information.

- **Non-volatile ROM** is used to store the program required to implement the desired actions.
  So, the program will not be lost when the power is turned off (Figure 10.1).
- Most important requirement: The microcontroller must have sufficient I/O capability.
  Parallel I/O Ports are used for dealing with the external I/O signals.
  Basic I/O Interfaces are used to connect to the rest of the system.

(b) Discuss the digital camera with neat block diagram. [8] CO4 L2

Digital Camera

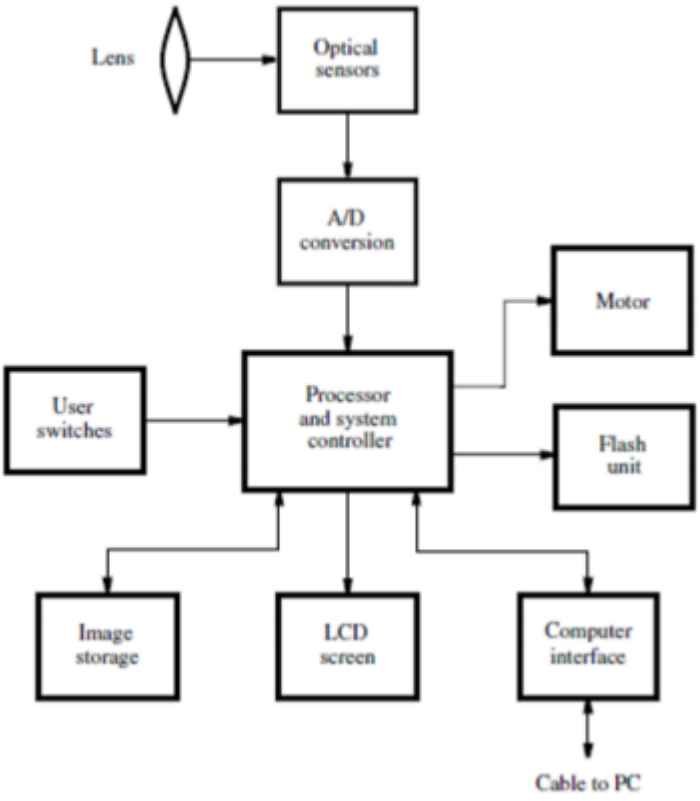


Figure 10.2 A simplified block diagram of a digital camera.

- Digital Camera is one of the examples of embedded system.
- An array of **Optical Sensors** is used to capture images (Figure 10.2).
- The optical-sensors convert light into electrical charge.
  - Each sensing-element generates a charge that corresponds to one **pixel**.
    One pixel is one point of a pictorial image.
    The number of pixels determines the quality of pictures that can be recorded & displayed.
  - **ADC** is used to convert the charge which is an analog quantity into a digital representation.
  - **Processor**
    → manages the operation of the camera.
    → processes the raw image-data obtained from the ADCs to generate images.
  - The images are represented in standard-formats, so that they are suitable for use in computers.
  - Two standard-formats are:
    1) TIFF is used for uncompressed images &
    2) JPEG is used for compressed images.
  - The processed-images are stored in a larger storage-device. For ex: Flash memory cards.
  - A captured & processed image can be displayed on a LCD screen of camera.
  - The number of saved-images depends on the size of the storage-unit.
  - Typically, **USB Cable** is used for transferring the images from camera to the computer.
  - **System Controller** generates the signals needed to control the operation of
    i) Focusing mechanism and
    ii) Flash unit.

(ADC → Analog-to-digital converter, LCD → liquid-crystal display)
(TIFF →Tagged Image File Format, JPEG →Joint Photographic Experts Group)