

part(a) Data Path subsystemsSubsystem Design:-

- The structured design ~~any~~ begins with the concept of hierarchy.
- It is possible to divide any complex function into less complex subfunctions. These may be subdivided further into even simpler subfunctions and so on. The bottom level being commonly referred to as 'leaf-cells'.
- This process is known as top-down design.
- As a system's complexity increases, its organization changes as a different factors become relevant to its creation.
- coupling can be used as a measure of how much submodules interact. clever systems partitioning aims at reducing implicit complexity by minimizing the amount of interaction between subparts; thus independence of design becomes a reality.
- It is crucial that components interacting with high frequency be physically approximate

since one pay severe penalties for long, high-bandwidth interconnects.

→ concurrency, should be exploited - it is desirable that all gates on the chip do useful work most of the time.

→ Because technology changes so fast, the adaption to a new process must occur in a short time. Thus a technology independent description becomes important.

In representing a design, several approaches may be used at different stages of the design process, for example.

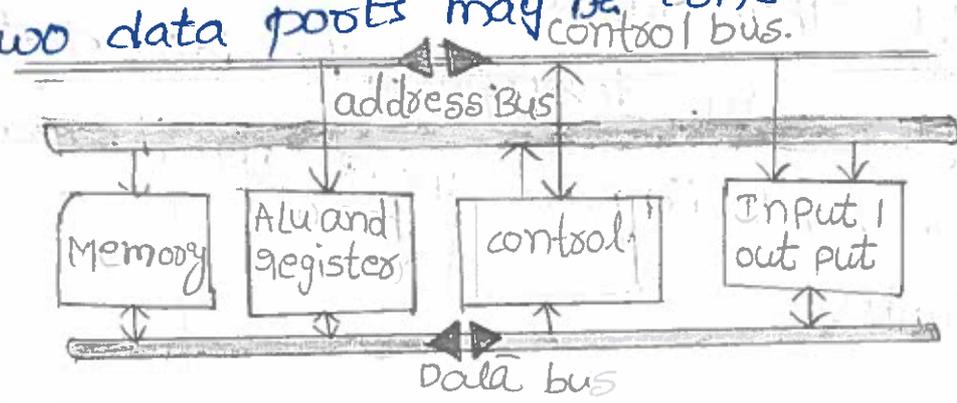
- conventional circuit symbols
- logic symbols
- stick diagram
- any mixture of logic symbol and stick diagram that is convenient at a particular stage.
- mask layout
- architectural block diagrams
- floor plans

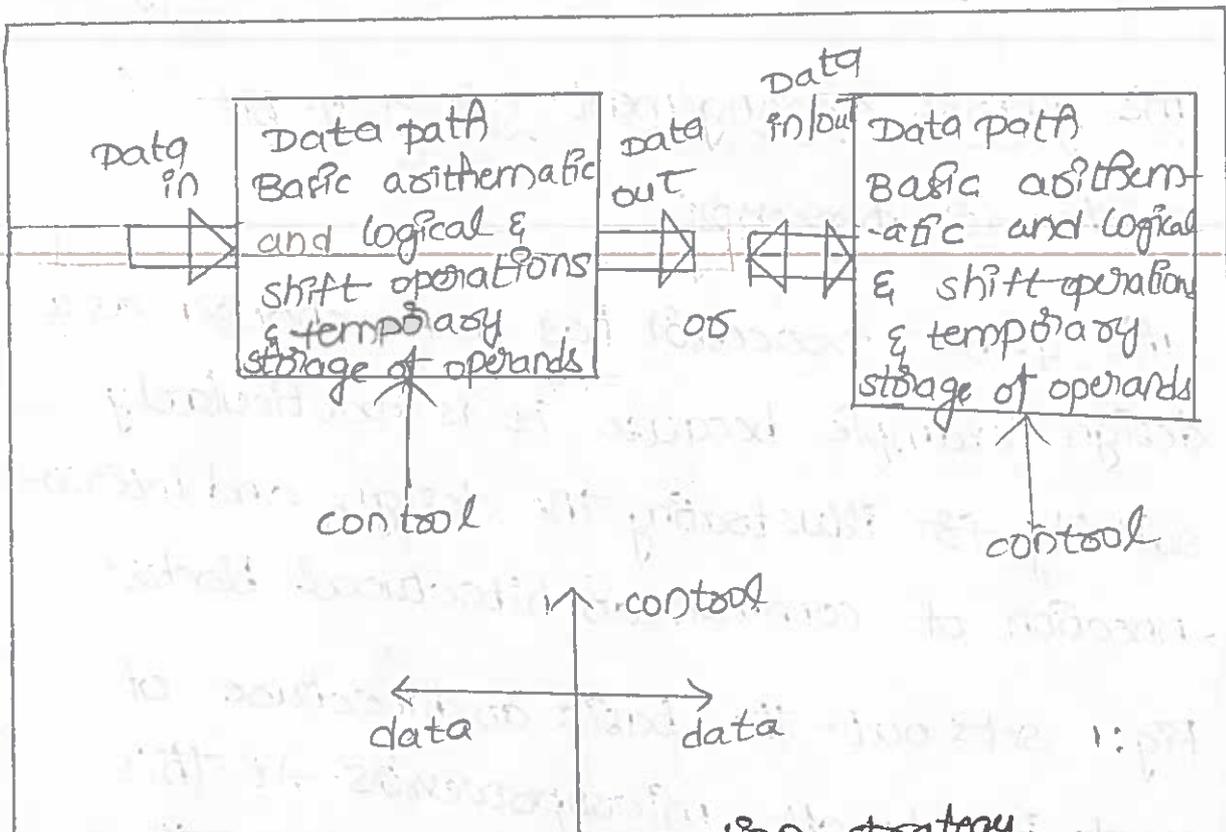
the general arrangement of a 4-bit arithmetic processor;

The 4-bit processor has been chosen as a design example because it is particularly suitable for illustrating the design and interconnection of common architectural blocks.

Fig:1 sets out the basic architecture of most if not all, microprocessors. At this stage we will consider the design of the data path only but matters relevant to other blocks will follow in later chapter.

The data path has been separated out in fig 2 and it will be seen that the structure comprises a unit which processes a data applied at one port and presents its output at a second port. Alternatively two data ports may be considered as





one possible interconnection strategy:
 communication strategy for data path.
 a single bidirectional port exists in the data path. control over the functions to be performed is effected by control signals as indicated.

At this early stage it is essential to evaluate an interconnection strategy to which we will then adhere.

Now we will decompose the data path into a block diagram shown in the main subunits.

In doing this it is useful to anticipate possible floor plan to show the planned relative disposition of the subunits on the chip and thus on the mask layout a block diagram is presented in Fig 3.

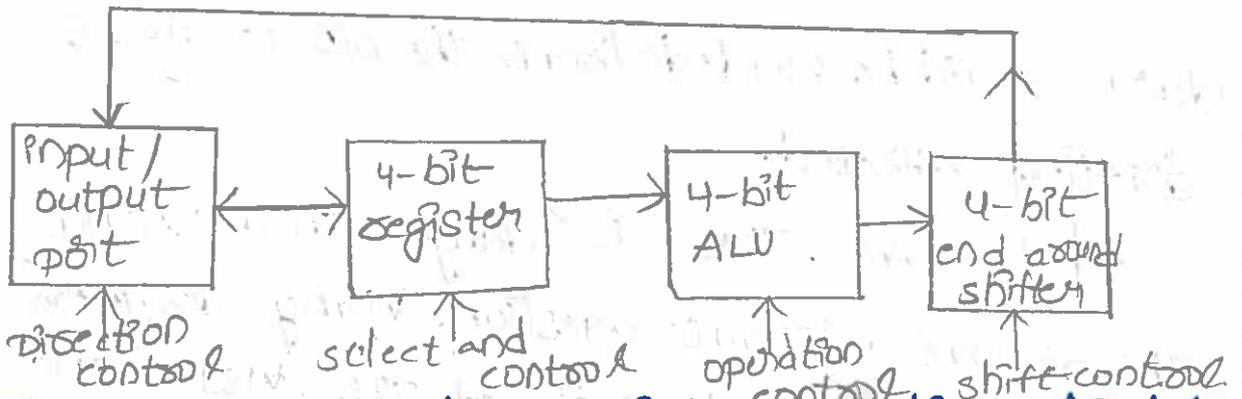


Fig 3:- subunits and basic interconnections for data path shifter:-

The design of a 4-bit shifter

Any general purpose n-bit shifter should be able to shift incoming data by up to n-1 places in a right-shift & left-shift directions. If we now further specify that all shifts should be on an 'end - that all shifts around 'basis, so that any bit shifted out at one end of a data word will be shifted in at that other end of the word then the problem of right shift & left-shift is greatly eased. In fact, a moment's consideration will reveal for a 4-bit word, that a 1-bit shift-right is equivalent to a

3-bit shift left and 2-bit right shift is equivalent to a 2-bit shift left etc.

The nature of the shifter having been decided on, its implementation must then be considered. data could be loaded from the o/p of ALU & shifting effected.

However, there is danger in accepting the obvious without question. Many designers used to the constraints of TTL, MSI and SSI logic would be conditional to think in terms of such standard arrangements.

In this case, the shifter must have,
→ input from a four-line parallel data bus.
→ four output lines for the shifted data
→ means of transferring i/p data to o/p lines with any shift from zero to three bits inclusive.
observe the strategy decided on earlier for the direction data and control. hence

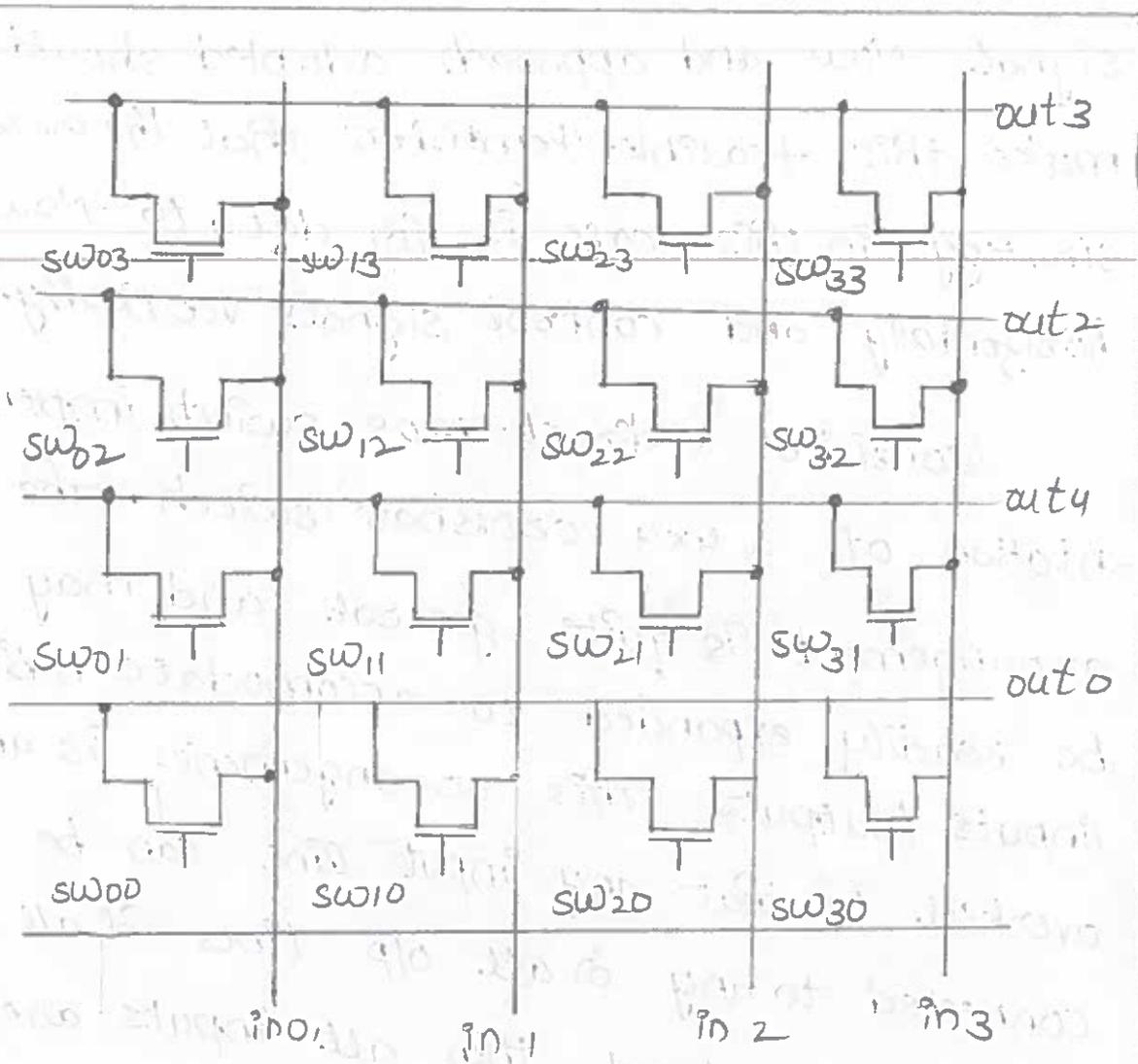
Signal flow and approach adopted should make this feasible. Remember that the overall strategy in this case is for data to flow horizontally and control signals vertically.

Consider a direct mos switch implementation of a 4x4 crossbar switch. The arrangement is quite general and may be readily expanded to accommodate n-bits inputs/outputs. This arrangement is an overkill in that any input line can be connected to any or all o/p lines - if all switches are closed, then all inputs are connected to all outputs in one glorious short circuit.

An adaption of this arrangement recognizes the fact that we can couple the switches gates together.

Multiplexers

~~A study of computers arithmetic process~~



in groups of four and also form four separate groups corresponding to shifts of zero, one, two and three bits. The arrangement is readily adapted so that the in-lines also run horizontally.

Multiplexers:-

A study of computer arithmetic will reveal that the most common requirements

are for addition and subtraction, but that there is also a significant need for a multiplicant capability

Serial - parallel multiplier

This multiplier is the simplest one, the multiplication being considered as succession of additions.

If $A = (a_n a_{n-1} a_{n-2} \dots a_0)$ and

$B = (b_n b_{n-1} b_{n-2} \dots b_0)$

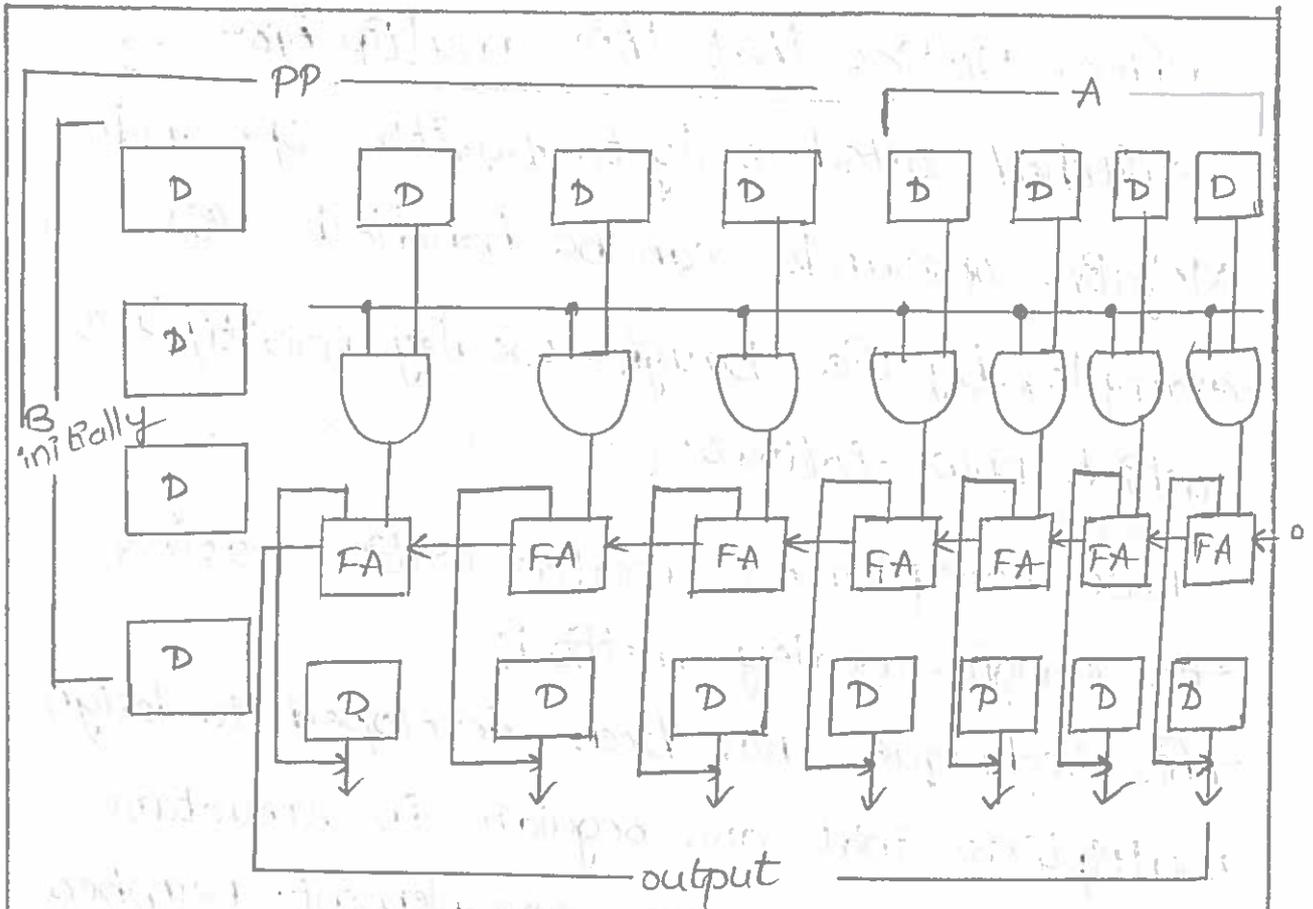
then the product $A \cdot B$ may be expressed as $A \cdot B = (A \cdot 2^n b_n + A \cdot 2^{n-1} b_{n-1} + A \cdot 2^{n-2} b_{n-2} + \dots + A \cdot 2^0 b_0)$.

A possible form of this adder for multiplying four-bit quantities, based on this expression is set. Note that Φ indicates a Φ flipflop and F.A indicates a full adder. The address bit-slice number A is entered in the right-most of the top row of Φ flipflops which are connected to three further Φ flipflops to form a

7-bit shift register to allow the multiplication of number A by $2^1, 2^2, \dots, 2^n$, thus forming the partial product each stage of the process.

The structure under discussion here is suited only to positive or unsigned operands. If the operands are negative and two's complement encoded them.

1. The most significant bit of B will have a negative weight and so a subtraction must be performed as the last step.
2. The most significant bit of A must be replicated since operand A must be expanded to $2N$ bits.



D = D flipflop FA = Full adder
 arrangement of a 4-bit serial-parallel multiplier
the Brown array;

A relative simple form of parallel adder is the Brown array. All partial products $A \cdot b_k$ are computed in parallel, then collected through a cascaded array of carry save adders. At the bottom of the array is used to convert the carry save form to the required form of output. completion time is fixed by the depth of the array, and by the carry propagation characteristic of the

adder. Notice that this multiplier is switched suited only to positive operands.

Negative operands can be handled for example, by the Baugh-woaley multiplier which now follows.

Two complement multiplication using the Baugh-woaley method:-

This technique has been developed to design multipliers that are regular in structure and suited for two complement numbers.

Let us consider two numbers A and B.

$$A = (a_{n-1} \dots a_0) = -a_{n-1}2^{n-1} + \sum_0^{n-2} a_i 2^i$$

$$B = (b_{n-1} \dots b_0) = -b_{n-1}2^{n-1} + \sum_0^{n-2} b_j 2^j$$

The product $A \cdot B$ is given by,

$$A \cdot B = a_{n-1} b_{n-1} 2^{n-2} + \sum_0^{n-2} \sum_0^{n-2} a_i b_j 2^{i+j} - a_{n-1} \sum_0^{n-2} b_j 2^{n+i-1} - b_{n-1} \sum_0^{n-2} a_i 2^{n+i-1}$$

If we use this form, it may be seen that subtraction operations are needed as well as addition. However, the negative terms may be rewritten for example.

$$a_{n-1} \sum_0^{n-2} b_i 2^{n+i-1} = a_{n-1} \left[-2^{n-2} + 2^{n-1} + \sum_0^{n-2} b_i 2^{n+i-1} \right]$$

using this approach, A.B becomes

$$A \cdot B = a_{n-1} b_{n-1} 2^{n-2} + \sum_0^{n-2} \sum_0^{n-2} a_i b_j 2^{i+j} + b_{n-1} \left[-2^{n-2} + 2^{n-1} + \sum_0^{n-2} a_i 2^{n+i-1} \right] + a_{n-1} \left[-2^{n-2} + 2^{n-1} + \sum_0^{n-2} b_i 2^{n+i-1} \right]$$

If we use this form, it may be seen that subtraction operations are needed as well as addition. However, the negative terms may be rewritten for example,

$$a_{n-1} \sum_0^{n-2} b_i 2^{n+i-1} = a_{n-1} \left[-2^{n-2} + 2^{n-1} + \sum_0^{n-2} a_i 2^{n+i-1} \right] + a_{n-1} \left[-2^{n-2} + 2^{n-1} + \sum_0^{n-2} b_i 2^{n+i-1} \right]$$

this equation may be put in a more convenient form by recognizing that,

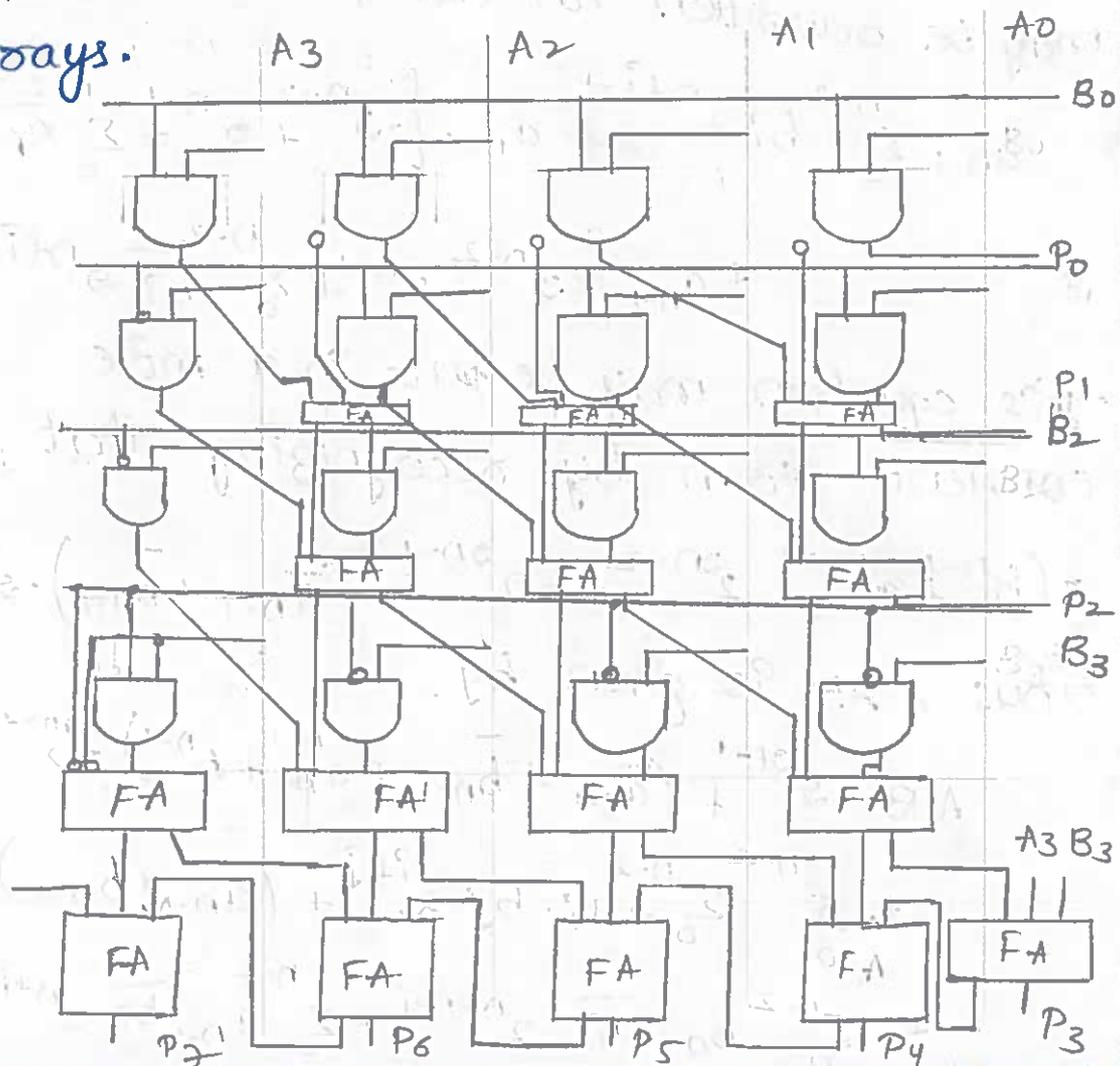
$$-(b^{n-1} + a^{n-1}) 2^{2n-2} = -2^{2n-1} + (\bar{a}_{n-1} + \bar{b}_{n-1}) \cdot 2^{2n-2}$$

thus, AB is given by,

$$A \cdot B = 2^{2n-1} + (\bar{a}_{n-1} + \bar{b}_{n-1} + a^{n-1} + b^{n-1}) 2^{2n-2} + \sum_0^{n-2} \sum_0^{n-2} a_i \cdot b_j \cdot 2^{i+j} + (a_{n-1} + b_{n-1}) 2^{n-1} + \sum_0^{n-2} b_{n-1} \bar{a}_i \cdot 2^{n+i-1} + \sum_0^{n-2} a_{n-1} \bar{b}_i \cdot 2^{n+i-1}$$

since A and B are n -bit operands, their product may extend to $2n$ bits. The first most significant bit is taken into account by the first term $\rightarrow 2^{n-1}$ which is fed to the multiplier as a 1 in the most significant cell.

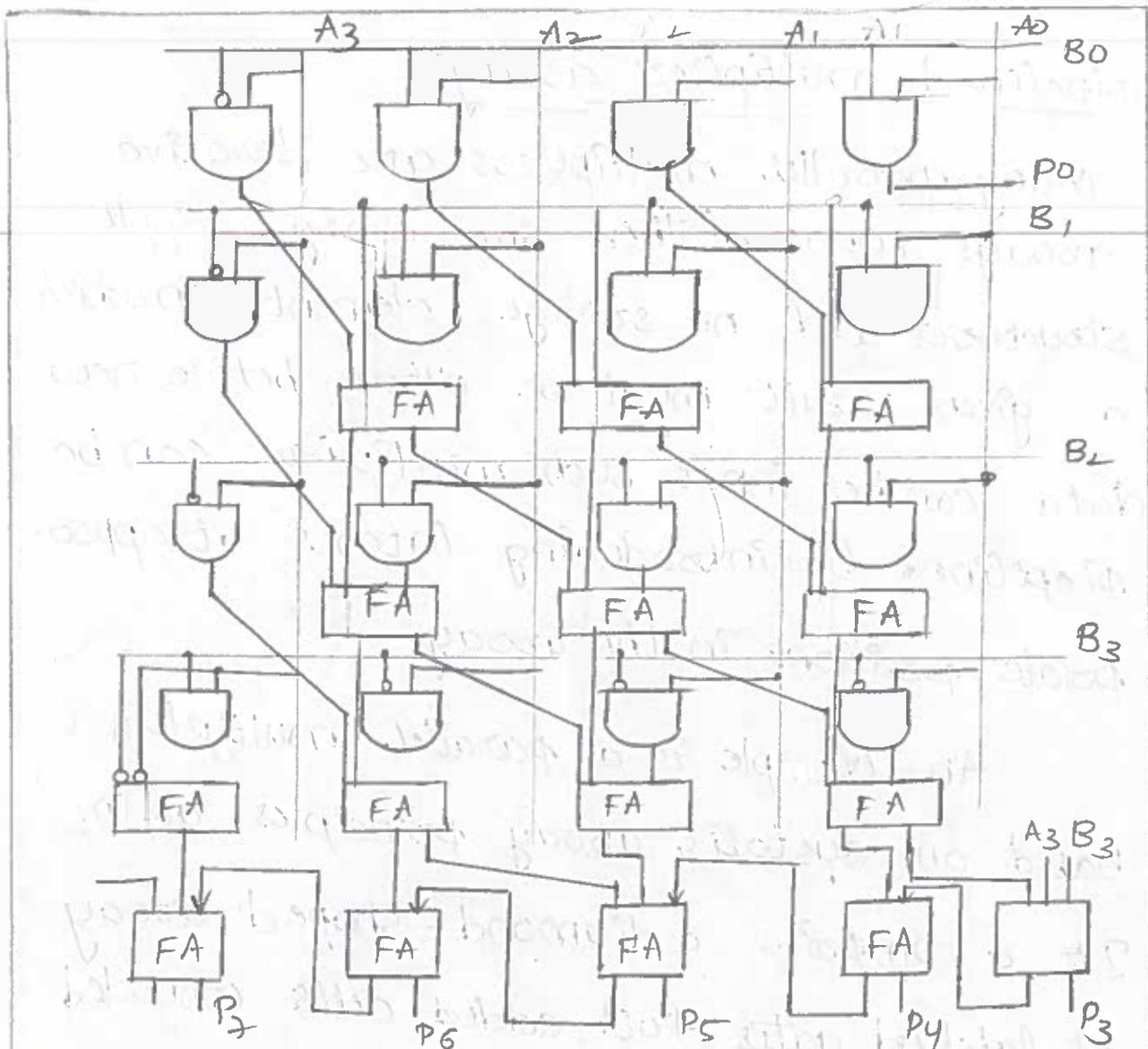
In a serial-parallel multipliers there are as many idle clock cycles as there are as in the multiplicand and the same situations in Basun and Baugh-wooky arrays.



pipelined multiplier array

Many parallel multipliers are iterative arrays some of these are carry-ripple structures with no storage elements, in which a given result must be output before new data can be input. such multipliers can be pipelined by introducing latches at appropriate positions in the array.

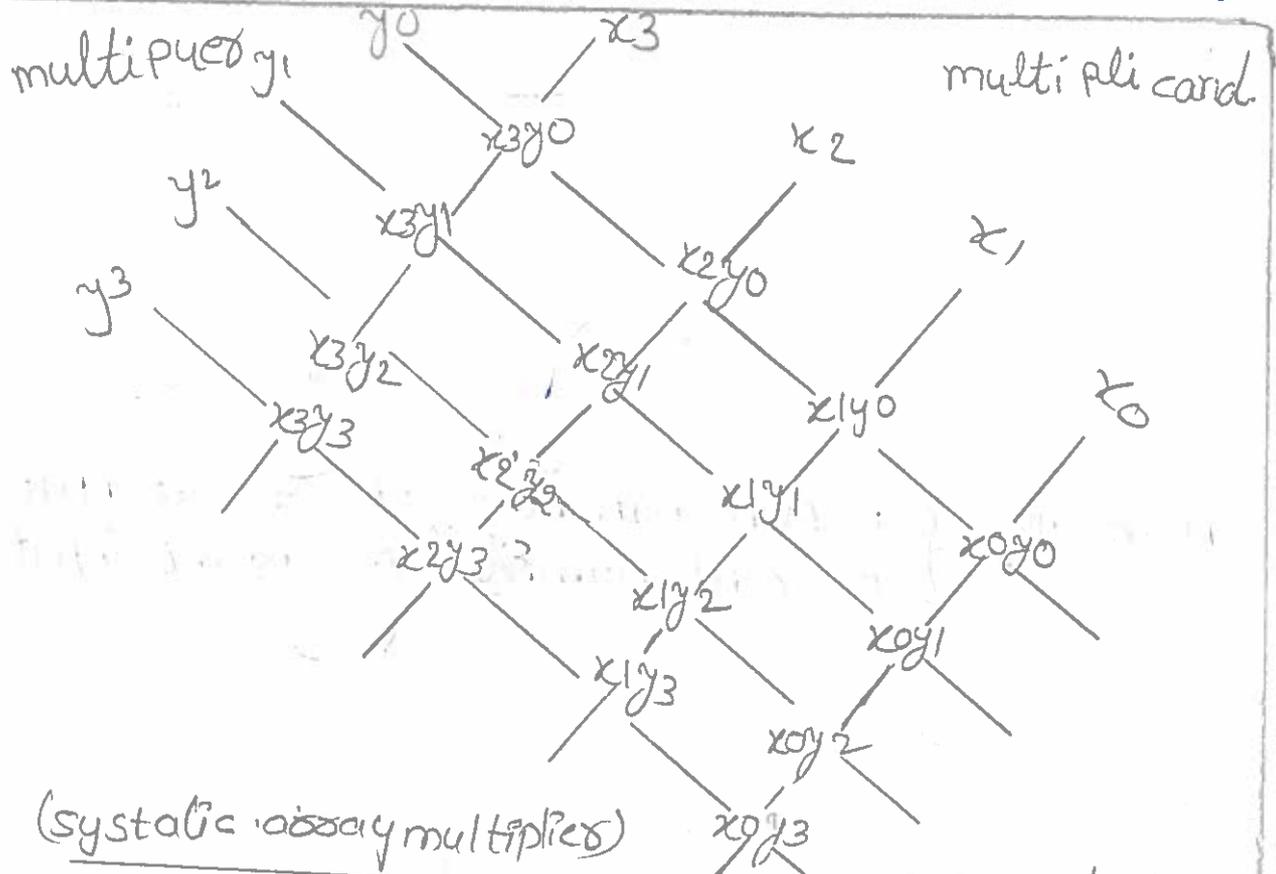
An example is a parallel multiplier based on systolic array principles as in. It consists a diamond-shaped array of latched gates full adder cells connected only to immediately adjacent cells. This has practical advantages as no broadcasting of data right across the multiplier array occurs.



the k th bit of the product

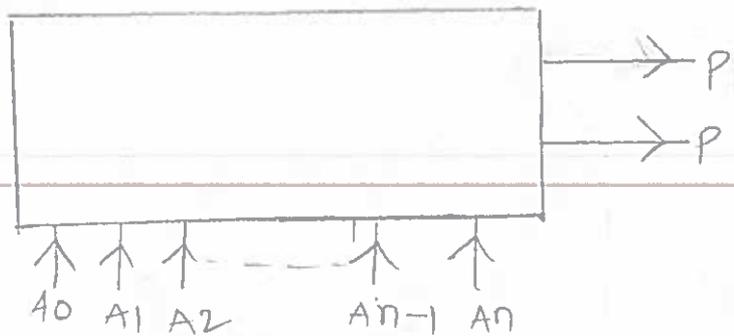
$$P_k = \sum_{i=0}^k x_{k-i} \cdot y_i$$

is formed by adding these components accumulate as P_k down the column carries generated at each stage in the array are passed to the left.

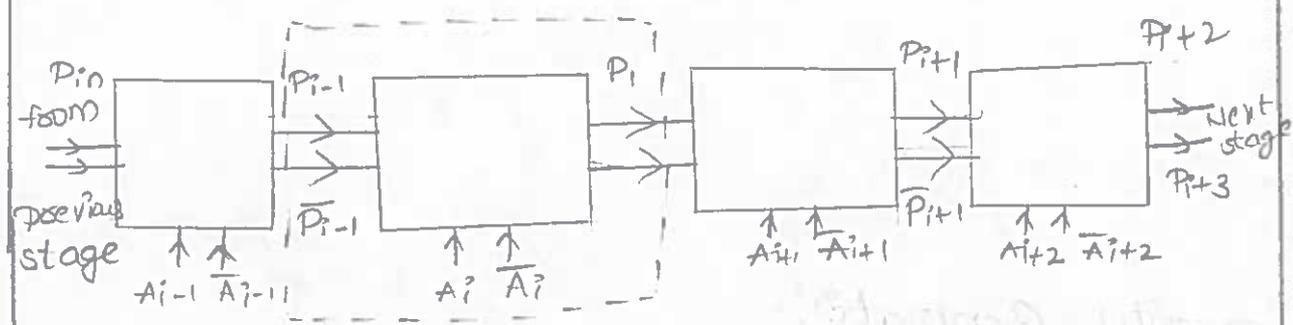


Parity Generator

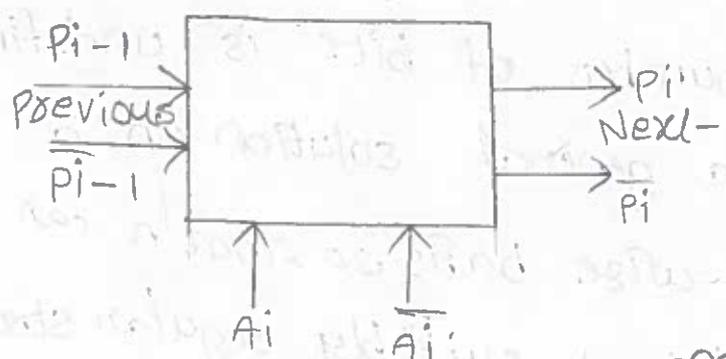
A circuit is to be designed to indicate the parity of a binary number or word. The requirement for an $(n+1)$ -bit input since the number of bits is undefined we must find a general solution on a cascadeable bit-wise basis so that n can have any value. A suitably regular structure is set out. From this, we can standard on basic one-bit cell from which an n -bit parity generator may be formed.



Note $P = \begin{cases} 1 & \text{Even number of 1s at input} \\ 0 & \text{odd number of 1s at input} \end{cases}$



parity generator - structured design approach



parity generator - basic one bit cell.

It will be seen that parity information is passed from one cell to the next and is modified or not by a cell, depending on

the state of the input lines A_i and \bar{A}_i .

A little reflection will readily reveal that the requirements are,

$A_i = 1$ parity is changed $P_i = \bar{P}_{i-1}$

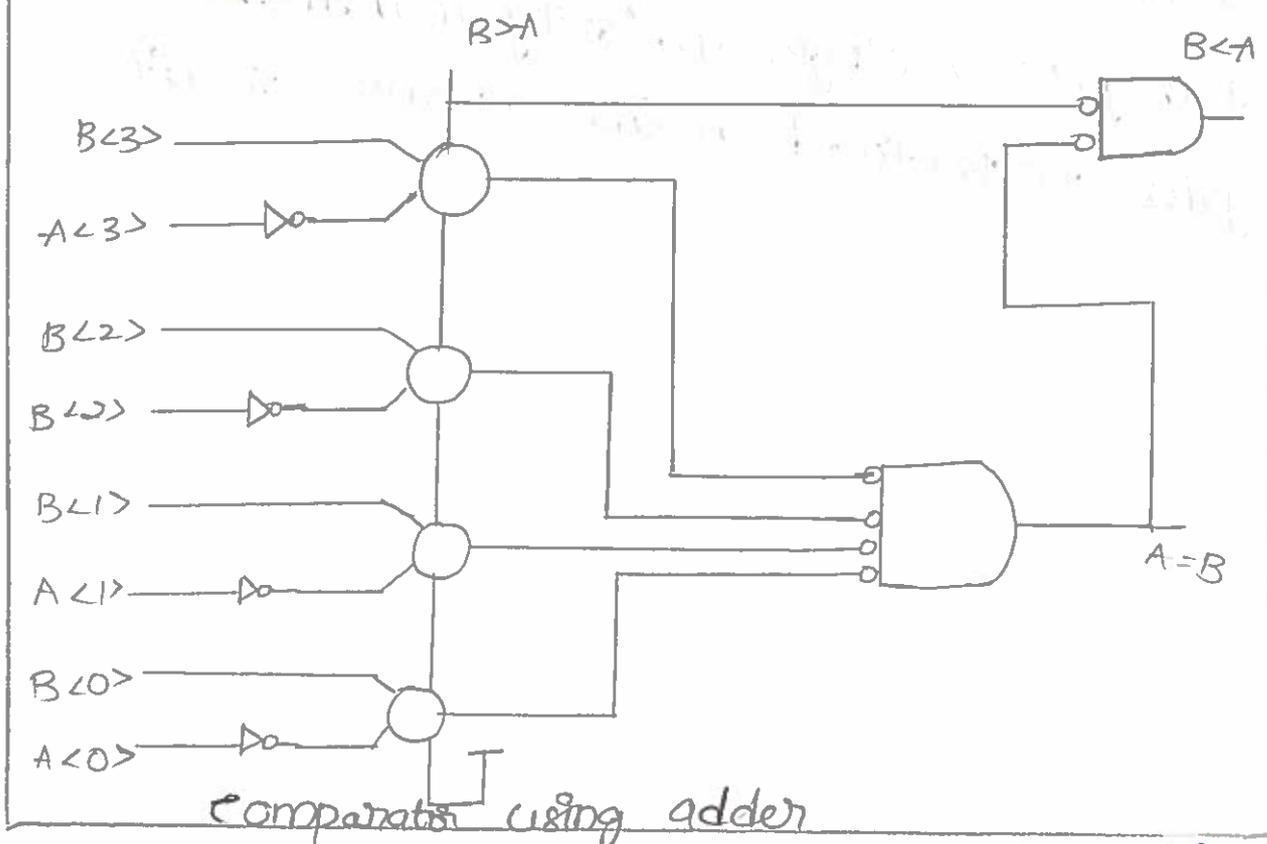
$A_i = 0$ parity is unchanged $P_i = P_{i-1}$

A suitable arrangement for such a cell is given in stick diagram. The circuit implements the function.

$$P_i = \bar{P}_{i-1} \cdot A_i + P_{i-1} \cdot \bar{A}_i$$

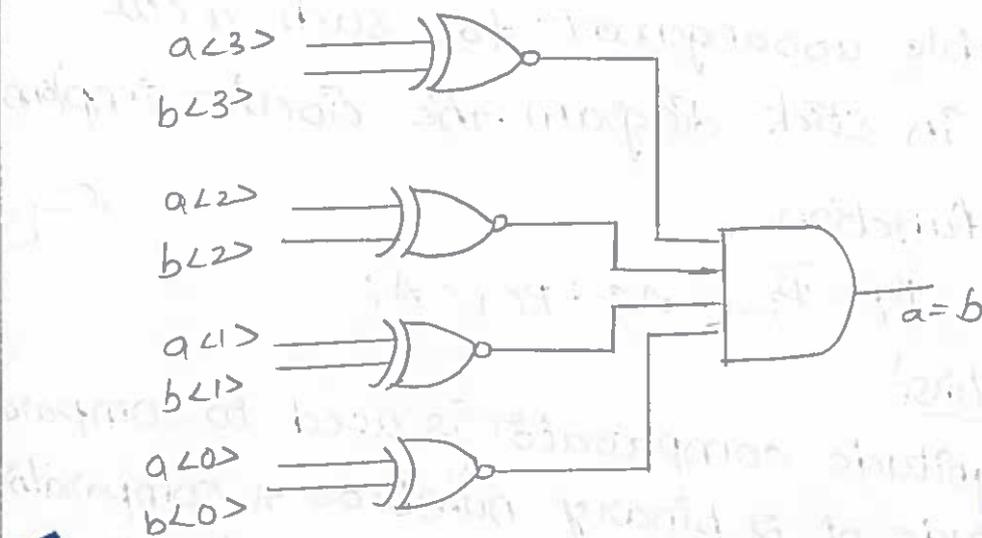
comparators

A magnitude comparator is used to compare the magnitude of 2 binary numbers. A comparator built by using an adder and a complement.

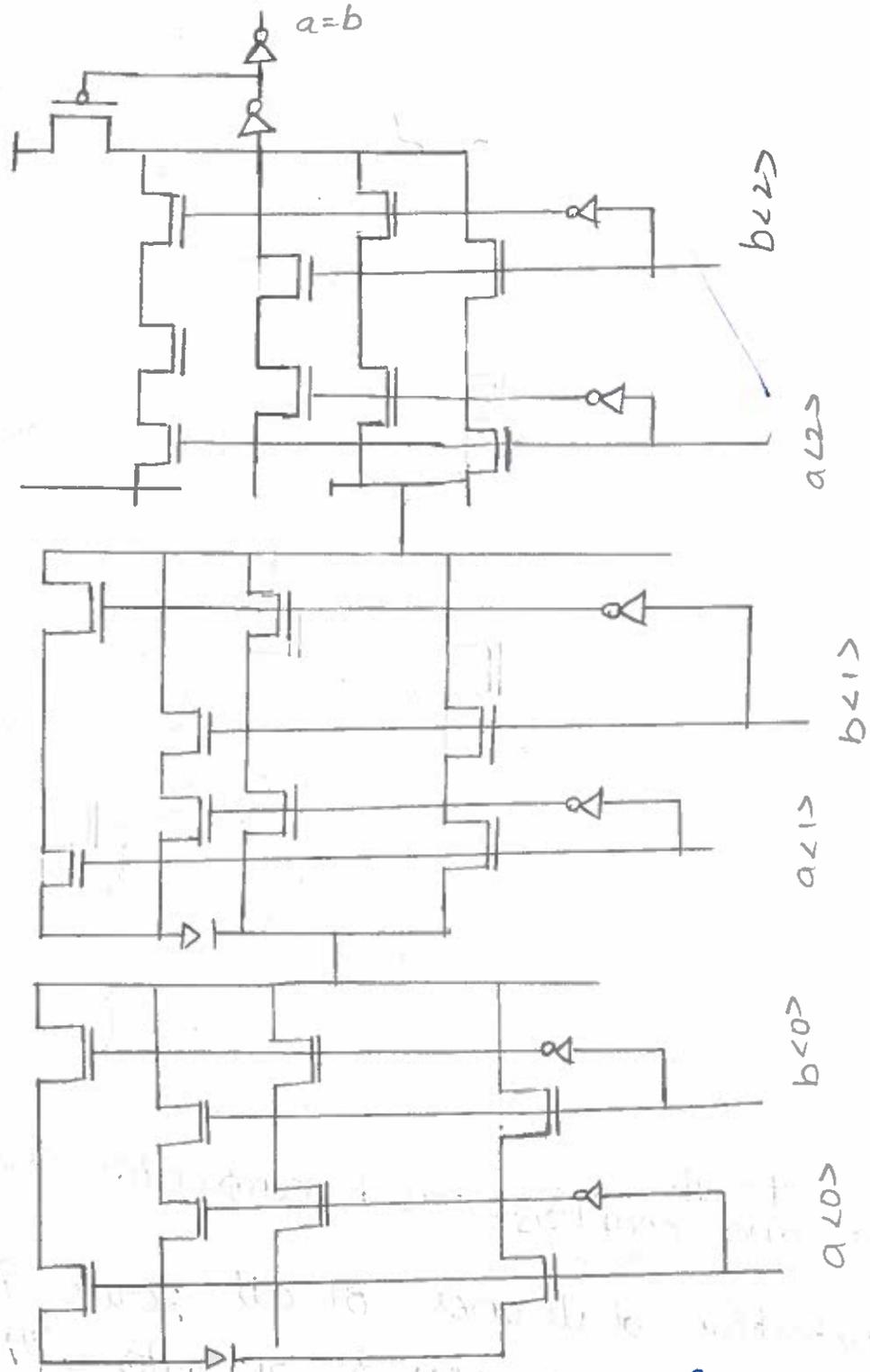


→ A zero detect gives output $A=B$, and final carry output provides $B>A$. signal for other combinations $A<B$ or $A\leq B$ can be generated by suitable arrangement.

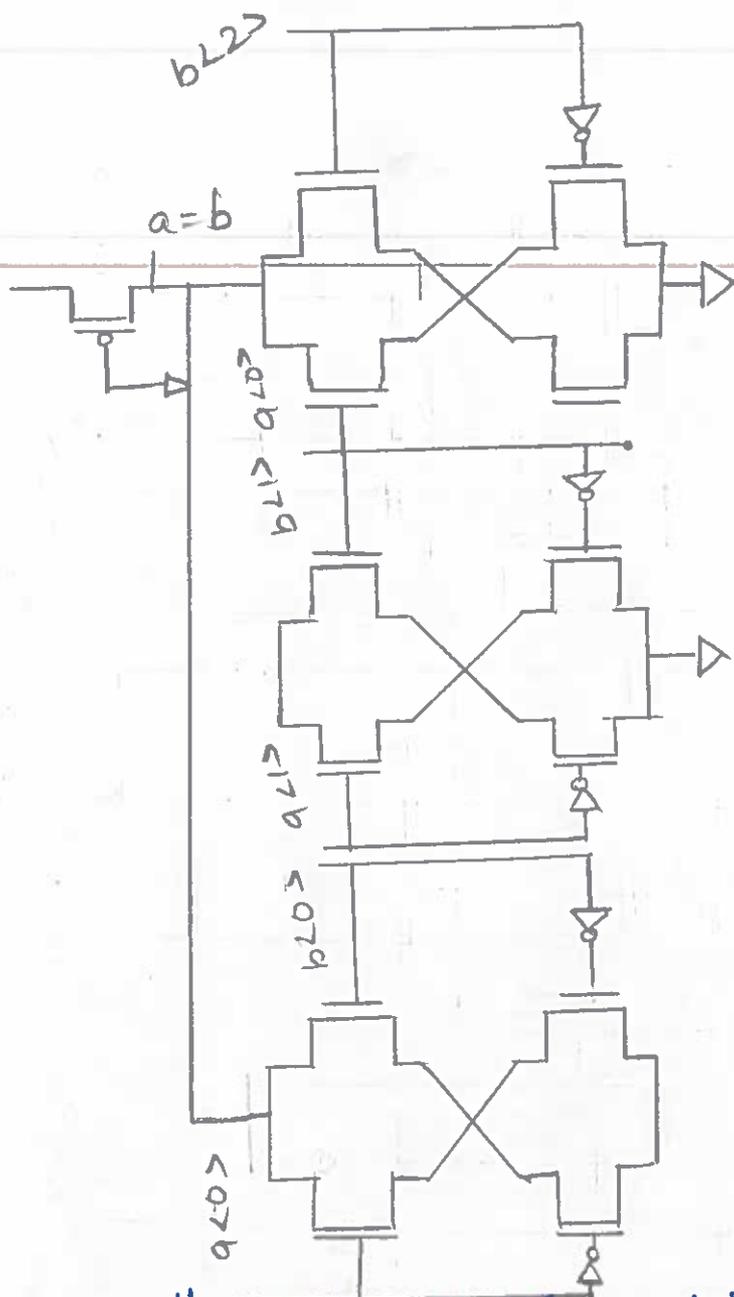
→ XNOR based comparator implementation.



→ A comparator implementation using pass gate. A single polarity transmission gates are preferred in low power circuit.

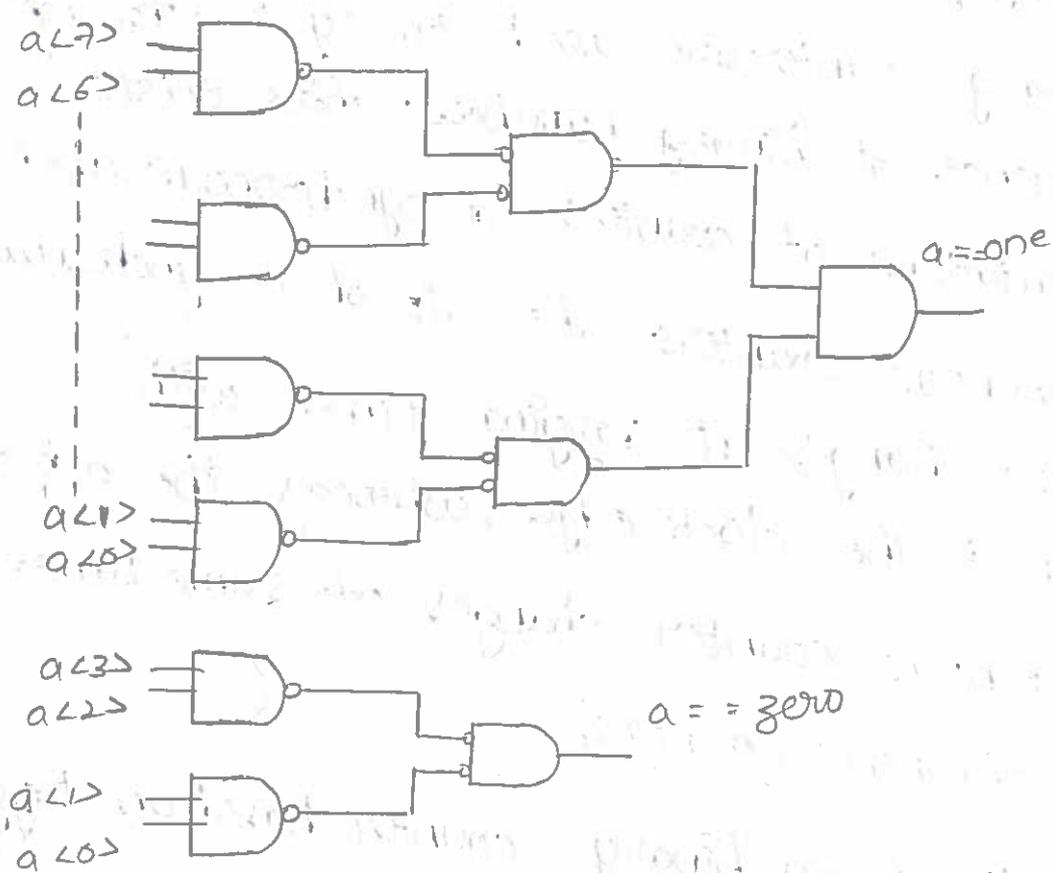


→ comparators implementation by using pseudo nmos it draws very small DC current and is very fast.



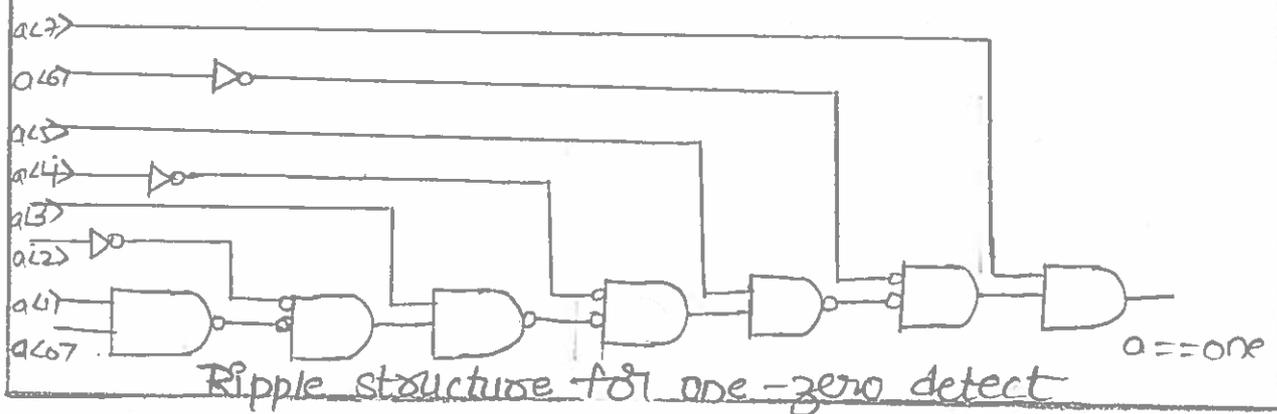
zero/one counters: Pseudo-nmos based comparator circuit

→ Detection of all ones or all zeroes is required for this AND or OR gates with large fanin is required. A tree of AND gates is used for detecting all ones or all zeroes



→ In tree structure NAND and NOR gates are used the delay of the o/p is proportional to $\log(N)$, where N is bit width of the word.

→ Another implementation for one, zero detection is ripple structure. Here the delay from the last changing o/p to zero/one detect is constant one gate delay.



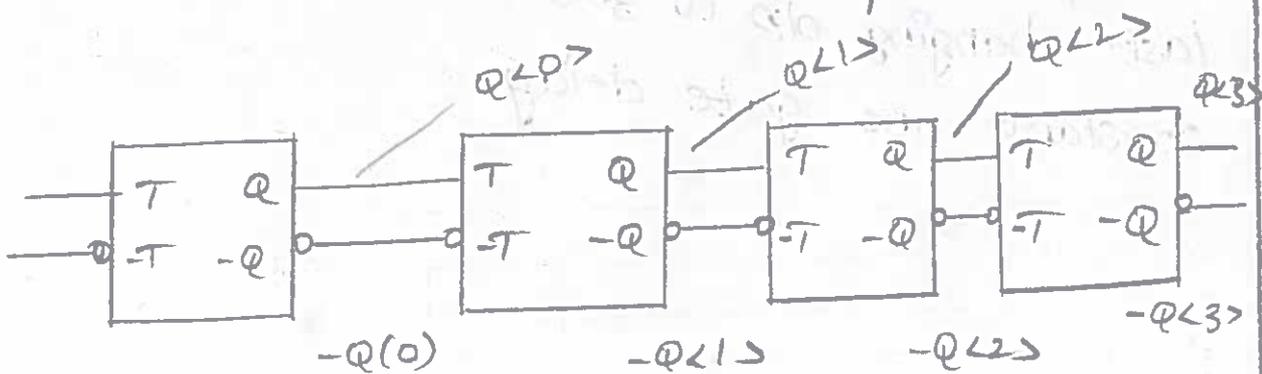
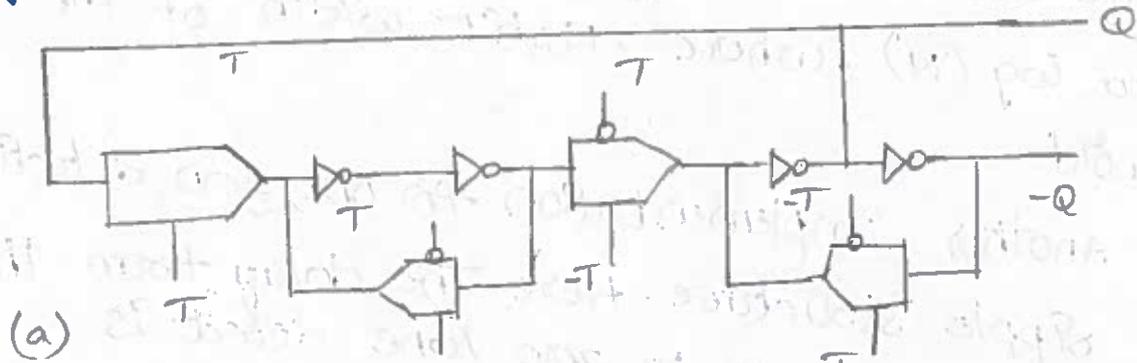
Ripple structure for one-zero detect

Counters

→ Binary counters are used to cycle through a sequence of binary numbers. There exists two variations of counters a synchronous and asynchronous counters. The o/p of a synchronous counters changes at varying times with respect to the clock edge, whereas the o/p of asynchronous counter changes of same time.

Asynchronous counter

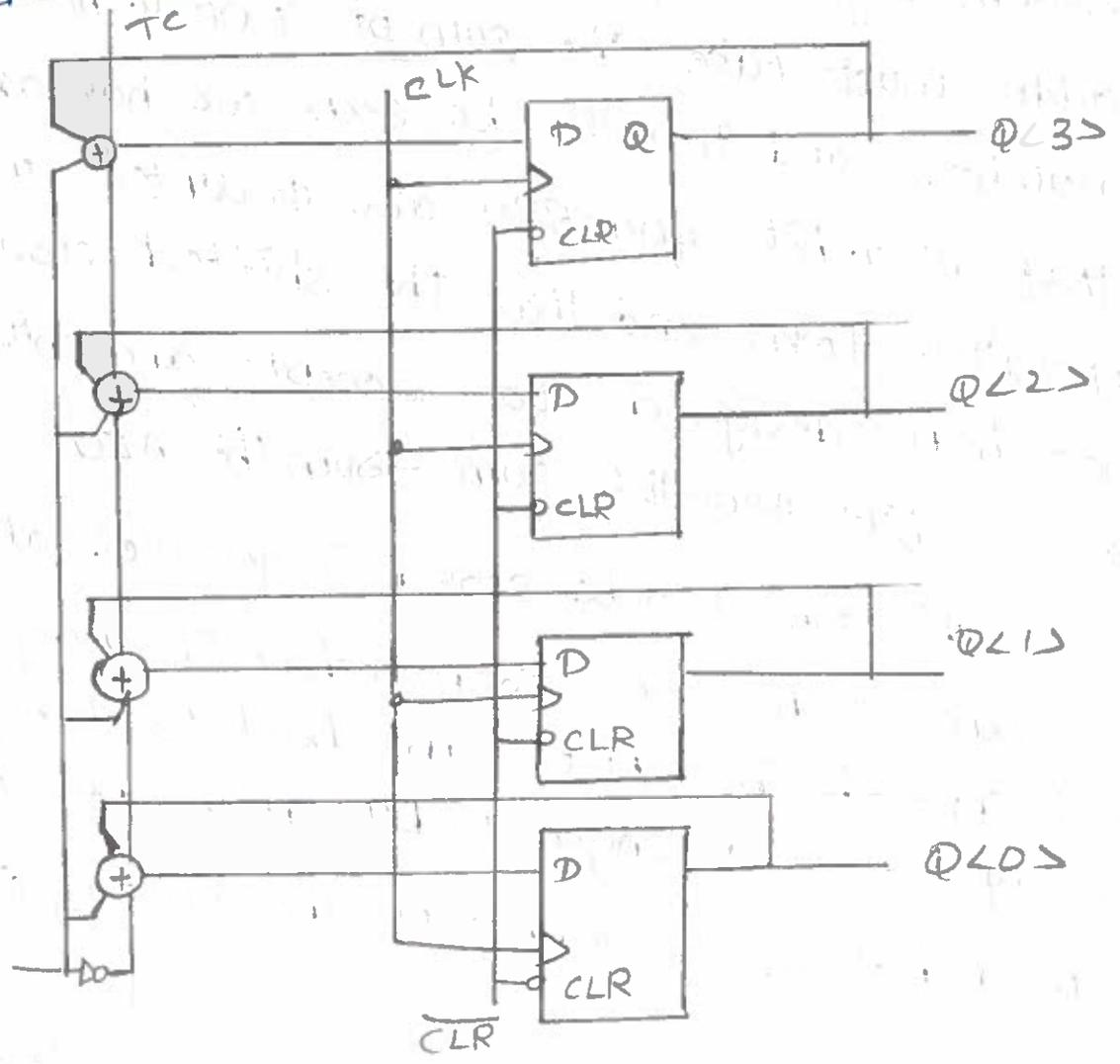
A ripple carry binary counter based on toggle register.



→ the clocking of each stage is done by previous counter stage, therefore last stage will have largest settling time. The circuit has no signal, therefore it is difficult to test the circuit.

Synchronous counters:-

→ A general synchronous up/down counter. An adder and D-type register is used for each bit. The ripple carry from LSB to MSB decides the speed of the counter.



→ Lookahead carry techniques can be used to minimize delay - the same ckt can be used as an increment with slight modifications.

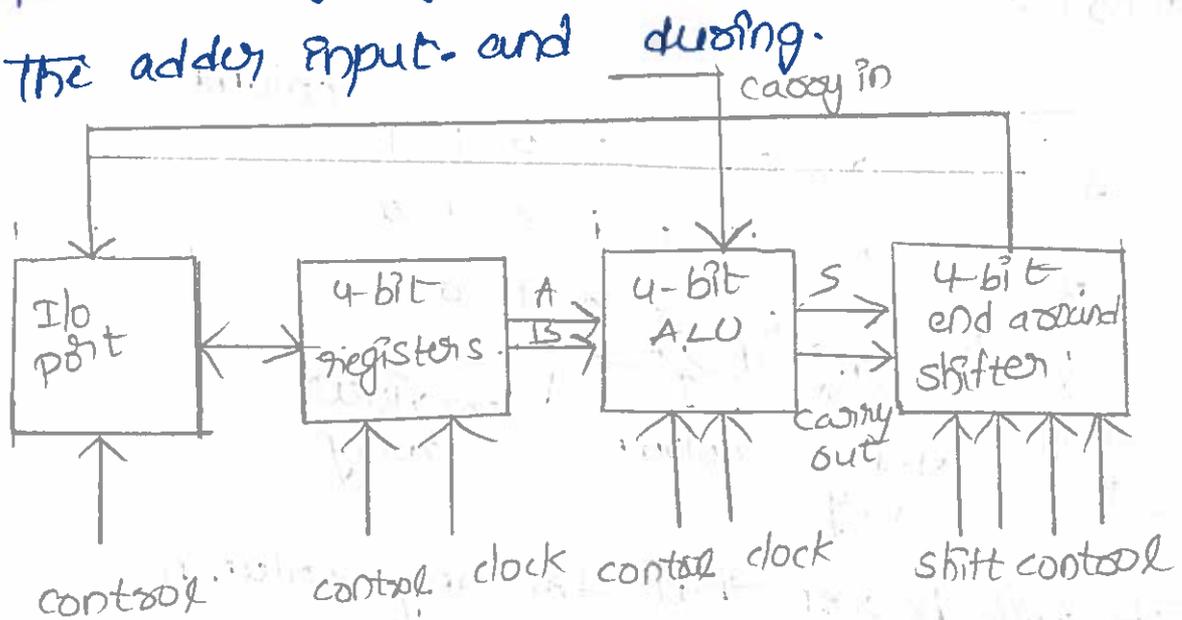
ALU:-

Design of an ALU subsystem:-

The heart of the ALU is a 4-bit adder circuit and it is this which we will actually design, indicating later how it may be readily adapted to subtract and perform logical operations. Obviously, a 4-bit adder must take the sum of two 4-bit numbers, and it will be seen we have assumed that all 4-bit quantities are presented in parallel form and that the shifter circuit has been designed to accept and shift a 4-bit parallel sum from the ALU.

The sum is to be stored in parallel at the output of the adder from where it may be read through the shifter and back to the registers. Thus, a single 4-bit data bus is needed from the adder to the shifter & others

4-bit bus is required anyway - As far as the input to the adder is concerned, the two 4-bit parallel numbers to be added are to be presented in parallel on two 4-bit buses we can also decide on some of the basic aspects of system timing at this stage and will assume clock phase ϕ , as being the phase during signals are fed along buses to the adder input - and during.



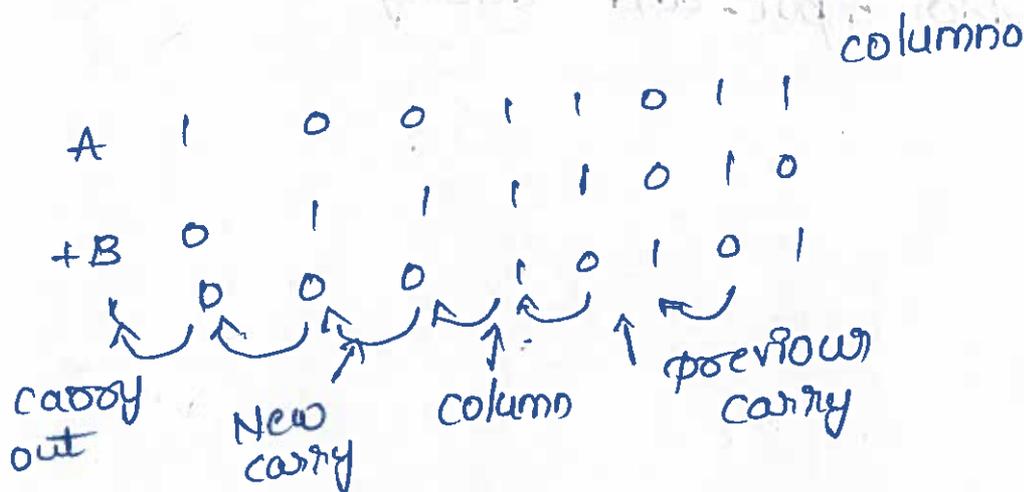
which their sum is stored at the adder output - thus clock signals are required by the ALU the shifter is unlocked but must be connected to four shift control lines. It is also necessary to provide a 'carry out' signals

from the adder and in the general case to provide for a possible 'carry in' signal as.

Address:-

Design of a 4-bit adder:-

In order to derive the requirements for an n -bit adder, let us first consider that addition of two binary numbers $A+B$ as follows.



It will be seen that for any column k there will be three inputs - the corresponding bits of the input numbers, A_k and B_k and the previous 'carry' - carry in (c_{k-1}) . It will also be seen that there are two outputs, the sum (S_k) & a new carry (c_k) .

Truth table for binary adder

Inputs		outputs		
A_k	B_k	previous carry c_{k-1}	sum s_k	New carry c_k
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

conventionally, and assuming that we are not implementing a 'carry look ahead' facilities we may write standard adder equations which fully describe the entire

one form of this eqⁿ is,

$$\text{sum } s_k = H_k \bar{c}_{k-1} + \bar{H}_k c_{k-1}$$

$$\text{New carry } c_k = A_k B_k + H_k c_{k-1}$$

where,

$$\text{Half sum } H_k = \bar{A}_k B_k + A_k \bar{B}_k$$

previous carry is indicated as c_{k-1} and $0 \leq k \leq n-1$ for n -bit numbers.

These equations may be directly implemented as And-or functions δ , most economically, S_k and H_k can be directly implemented with Exclusive or gates.

Address element requirements:-

It reveals that the address requirements may be stated thus,

If $A_k = B_k$ then, $S_k = C_{k-1}$

else, $S_k = \bar{C}_{k-1}$

and for the carry C_k

If $A_k = B_k$ then $C_k = A_k = B_k^*$

else, $C_k = C_{k-1}$

This relationship could also have been stated as,

carry $C_k = 1$ when, $A_k = B_k = 1$

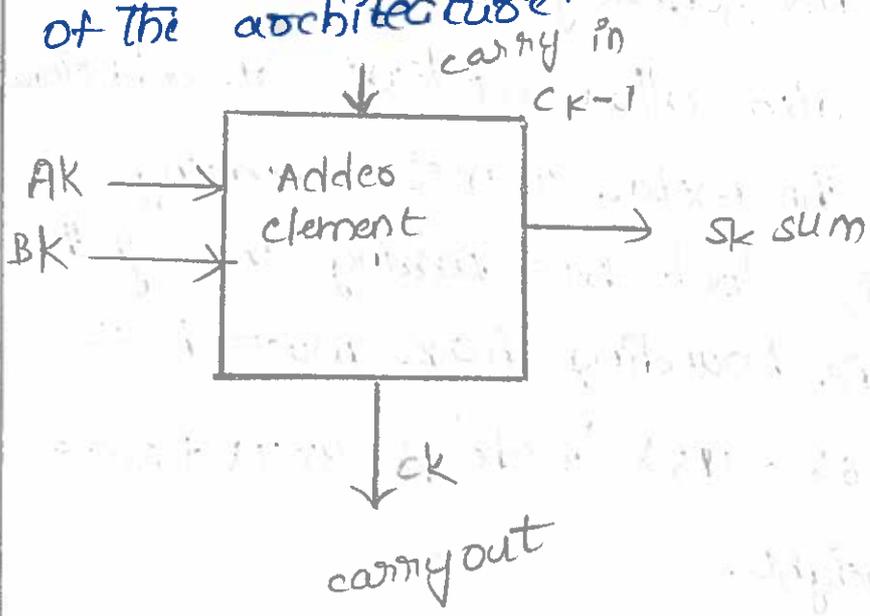
or, $C_k = 0$ when $A_k = B_k = 0$

Standard address element:-

A 1-bit address element may now be represented. Note that any number of such elements may be cascaded to form any size of address and that the element is quite general.

Note also that this standard address

element may itself be composed from a number of replicated subcells - regularities & generality must be aimed at in all levels of the architecture.

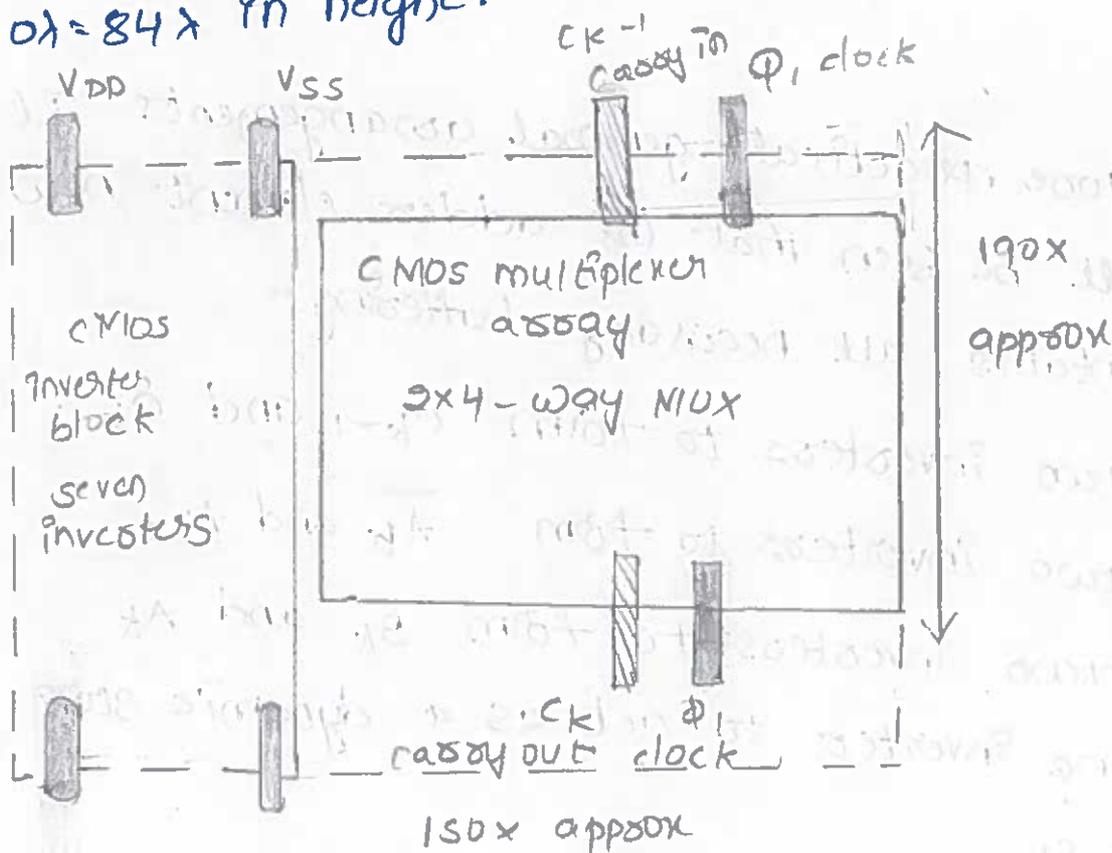


More practical general arrangements It will be seen that the address element now contains all necessary buffering.

- two inverters to form \bar{c}_{k-1} and c_{k-1}
- two inverters to form \bar{A}_k and A_k
- two inverters to form \bar{B}_k and B_k
- one inverter to act as a dynamic store for S_k .

Address element bounding box:-
The CMOS address element as. First estimate the bounding box for the multiplexer area

of the address. Each standard multiplexer cell is $7\lambda \times 11\lambda$ and there is 16 such elements side by side 'horizontally' and four stacked 'vertically'. It also allow at least an additional 6λ width for the metal to metal spacing required by the clock bus passing through the center. Thus the bounding box must be at least $16 \times 7\lambda + 6\lambda = 118\lambda$ wide & $4 \times 11\lambda + 30\lambda + 10\lambda = 84\lambda$ in height.



UNIT - IV

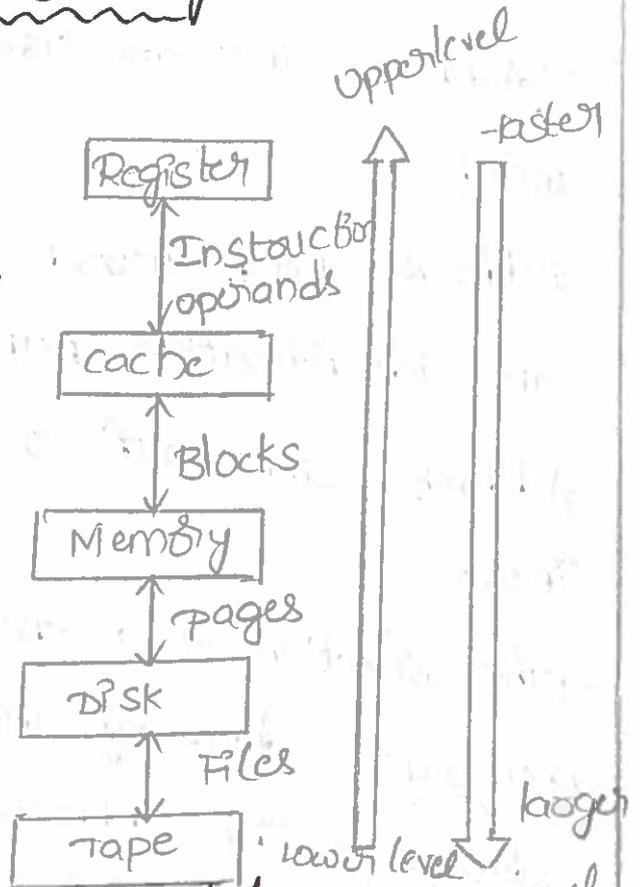
part (b) Array subsystems

Memory Architecture

Levels of Memory Hierarchy

capacity & Access time

- CPU Regs, 100s Bytes
- cache, k Bytes 10-100ns
- Main memory, M Bytes
200ns - 500ns
- Disk, G-Bytes, 10ms
(10,000,000ns)
- Tape infinite
sec - min



Architecture

Fig: Various levels of memory

- In most IC designs various sizes data storage devices (memory) are required.
- Memory cells can be accessed for information transfer to or from any desired location. Hence the name random access memory or simply RAM.

- A memory unit is specified
 - no. of words it contains
 - no. of bits in each word

Word: A memory unit can store binary information in group of bit

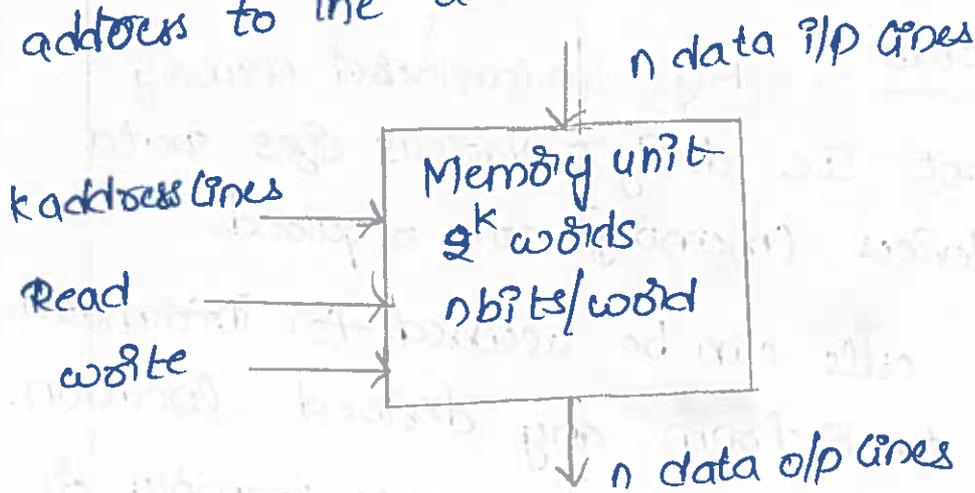
Byte: Group of eight bits.

→ Address lines are used to select one particular word.

Address: Each word in memory is assigned an identification number called an address.

Address starts with 0, 1, 2, 3, ... k Address lines.

→ The selection of a specific word inside the memory is done by applying the k-bit binary address to the address lines.



Memory elements:

→ Memory elements can store a value as

controlled by clock. it may have load signal etc.

→ In CMOS memory is created by,

- (i) capacitance (dynamic)
- (ii) F/B (static)

Memory ^{can be} distinguished on the basis of

→ form of required clock signal

→ How behaviour of data i/p around clock affects the stored value.

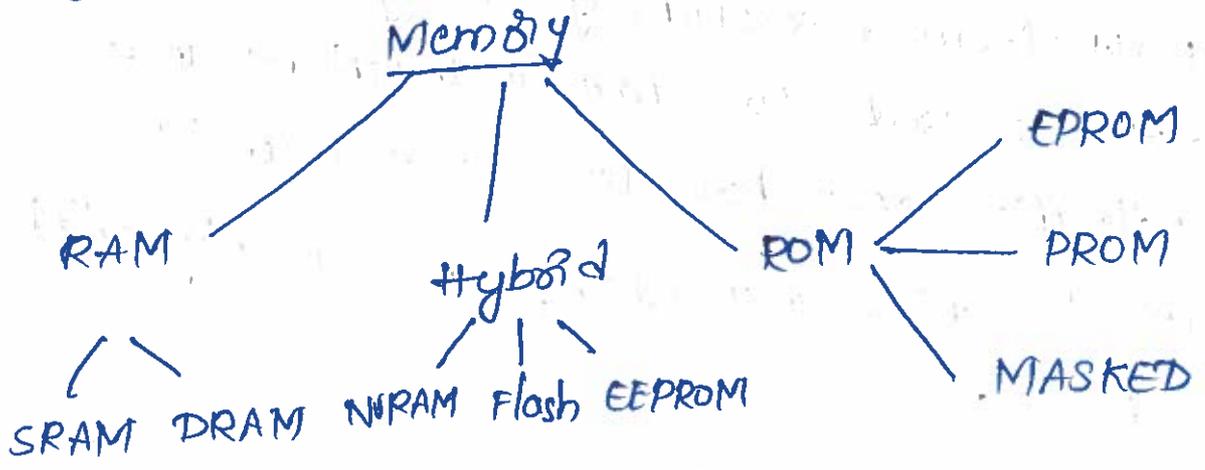
→ When the stored value is presented to the o/p.

Memory element terminology:-

1. Latch: these are transparent, the i/p value is transmitted to the o/p.

2. Flipflops:- these are not transparent reading i/p value and changing its o/p are separate events.

Types of memory in embedded system:-



Many types of Memory devices are available for use in modern computer systems.

for ex:- Embedded system consists of RAM.

RAM: Random Access Memory

→ RAM is defined as memory array with individual bit access.

→ it refers to memory with both Read and write capabilities.

RAM are two types

1) SRAM (static RAM)

↓

→ Holds data as long as power is applied

2) DRAM (dynamic RAM)

→ Must be refreshed periodically

RAM: store data and instructions that are used by the CPU to perform some task.

→ these instructions are usually loaded into RAM from a secondary storage device.

→ It is used to store instructions that tells the CPU how to work with its parts.

these instructions are usually called drivers.

→ The instructions are volatile ^{when} the computer is turned off the information in RAM disappears.

→ The information in RAM needs to be saved to secondary storage before the computer is turned off.

SRAM:- It is widely used in integrated circuits.

It can store data as long as when the power is on.

→ SRAM has 3 different states it can be,

a) Hold where the circuit is idle

b) writing when updating the data

c) Reading when the data has been required.

Advantages:-

→ compatibility with modern logic

→ High speed - fastest of all semiconductors

→ Ease of use

→ High density

Applications:-

→ semiconductor IC applications used widely.

A Pseudo-static RAM / register cell:-

All the storage elements considered as volatile and needed to be periodically refreshed. This is not always convenient and it is necessary to consider the design of a static storage cell which will hold data when the power is on.

considers system timing considerations:-

1. A two-phase non-overlapping clock is assumed and this clock alone will be used throughout the system.
2. clock phases are identified as ϕ_1 and ϕ_2 where ϕ_1 assumed to lead ϕ_2 .
3. Bits are stored to be written to registers, storage elements and subsystems on ϕ_1 of the clock. that is write signals are ANDed with ϕ_1 .
4. Bits of data written into storage elements may be assumed and ϕ_2 signals may be used to refresh stored data where appropriate.

5. In general, delays through datapaths, combinational logic etc are assumed to be less than the interval b/w the leading edge of ϕ_1 of the clock and the leading edge of the following ϕ_2 signal.

6. Bits of data may be read from storage elements on the next ϕ_1 of the clock; that is read signals RD are Anded with ϕ_1 .

Generally RD and WR are mutually exclusive to any one storage element.

Write

Bus enables WR line it is coincidence with ϕ_1 , thus the bit is stored in c_q of Inv 1. and inv 2 gives true o/p.

during every ϕ_2 the clock the stored bit is refreshed through k/b path from the o/p Inv 2 to i/p of Inv. Thus bit will hold as long as ϕ_2 of the clock sep

RD (read)

to read the state of the cell it is only through the RD lines which is also assumed to coincidence with ϕ_1 of the clock and bit will be read on to the bus.

1. WR, RD mutually, exclusive
2. ϕ_2 is used for setup (must not read during ϕ_2 of clock) charge sharing effects.
3. cells must be stackable side by side and top to bottom
4. Allow other bus lines to run through the cell for array computation.

$$A_{\text{cell}} \Rightarrow 5\lambda \times 45\lambda = 225\lambda^2 \text{ (2 buses)}$$

$$1750\lambda^2 \text{ or less (single bus)}$$

$$\therefore A_{\text{cell}} / \text{bit} \approx 10000 \mu\text{m}^2$$

$$\text{max no. of bits } 1.4 \text{ k bits}$$

Dissipation:-

nmos cell uses 2 inverters ① 8:1

② 4:1

Dissipation depends on 'I' drawn and, VDD.

Here, R of 8:1 $90 \text{ k}\Omega$ b/w supply (V_{DD})
 4:1 $50 \text{ k}\Omega$

$$\therefore \text{Avg current} \Rightarrow \frac{1}{2} \left(\frac{5V}{90 \text{ k}\Omega} + \frac{5V}{50 \text{ k}\Omega} \right)$$

$$\approx 80 \mu\text{A}$$

$$\text{dissipation / bit stored} = 80 \mu\text{A} \times 5V$$

$$= 400 \mu\text{W}$$

power dissipat on a single chip for

1.4 kbytes

$$400 \times 1.4 \text{ k} \mu\text{W}$$

$$\Rightarrow 560 \text{ mW}$$

5

voltage}

The cell Non-volatile provided that ϕ_2 signals are present.

Transistor Dynamic and six-transistor static CMOS memory cells:-

Most of the memory cells involved n-type transistors. Now this one is implemented by both n-type and p-type transistors and it is called as CMOS static Memory cell.

Both the dynamic and static elements, use a two bus/bit arrangement so that associated with every bit stored there will be a bit and a bit bar. In both cases buses are precharged to logic 1 before read & write operations takes place.

In fig (1) is a 4-transistor dynamic cell for storing one bit. Each bit is stored on the gate capacitance of n-type transistors T_1 and T_2 and a description of write and read operation follows.

(a) Write operations:-

Both bit and $\bar{\text{bit}}$ buses are precharged to V_{DD} (logic 1) in coincidence with ϕ_1 , of an assumed two phase clock precharging is effected via the p-transistors T_5 & T_6 .

(when $v_{in} = \text{logic } 0$ p-MOS transistors T_5 & T_6 on so $v_{out} = V_{DD}$ it is called precharge state.)

Now with reference to fig (c):-

The appropriate 'column' select line is activated in coincidence with the clock phase ϕ_2 . either the bit or $\bar{\text{bit}}$ line is discharged by the logic levels present on the I/O bus lines, the I/O bus lines acting in this case as a current sink when carrying logic '0'. [Here $v_{in} = \text{logic } 1$, $v_{out} = \text{logic } 0$ so, bit or $\bar{\text{bit}}$ are discharged].

The 'row select' is activated at the same time as 'column select' and the bit line states are written via T_3 & T_4 .

and stored by T_1 & T_2 are interconnected will force them into complementary states while the 'row select' high.

once the select lines are (row, column) deactivated, the stored and remembered until the gate capacitances lose enough charge to drop the 'on' gate voltage below the threshold level for T_1 & T_2 .

2. Read operations:-

once again both bit and $\bar{\text{bit}}$ lines are precharged to V_{DD} via T_5 & T_6 during ϕ_1 , so, that both lines will be at logic 1. Now if, say a 1 has been stored, T_2 will be on and T_1 will be off, and thus the $\bar{\text{bit}}$ line will be discharged to V_{SS} (logic 0). through t_2 and the stored bit reappears on the bit lines.

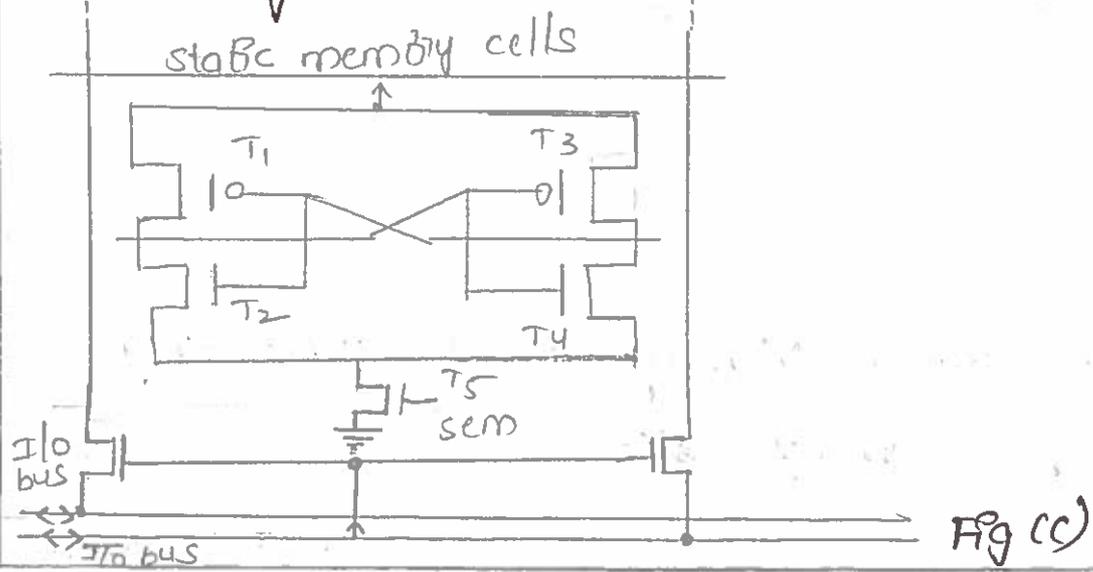
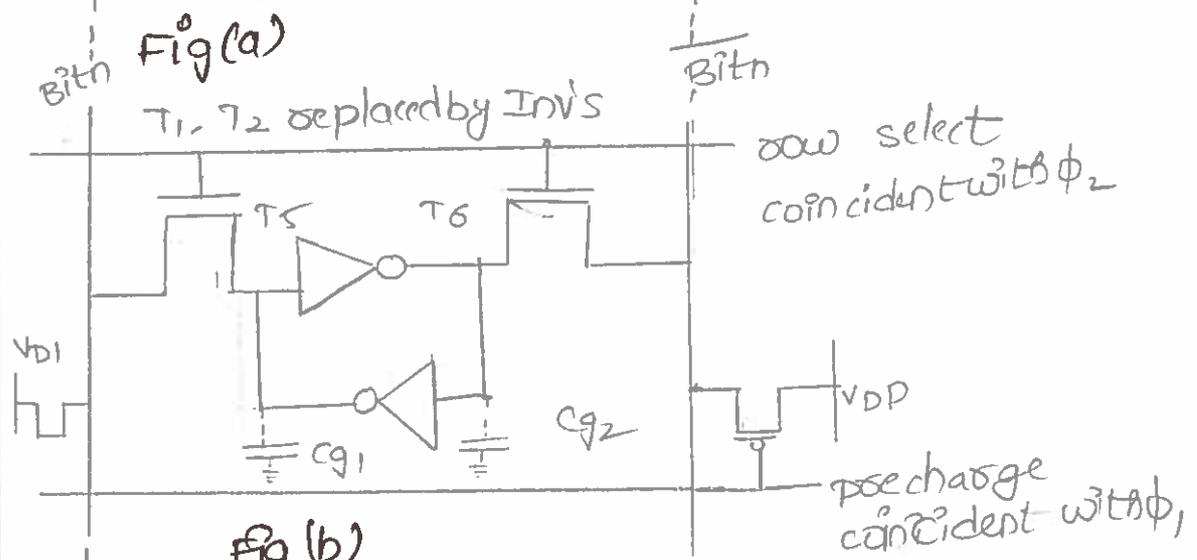
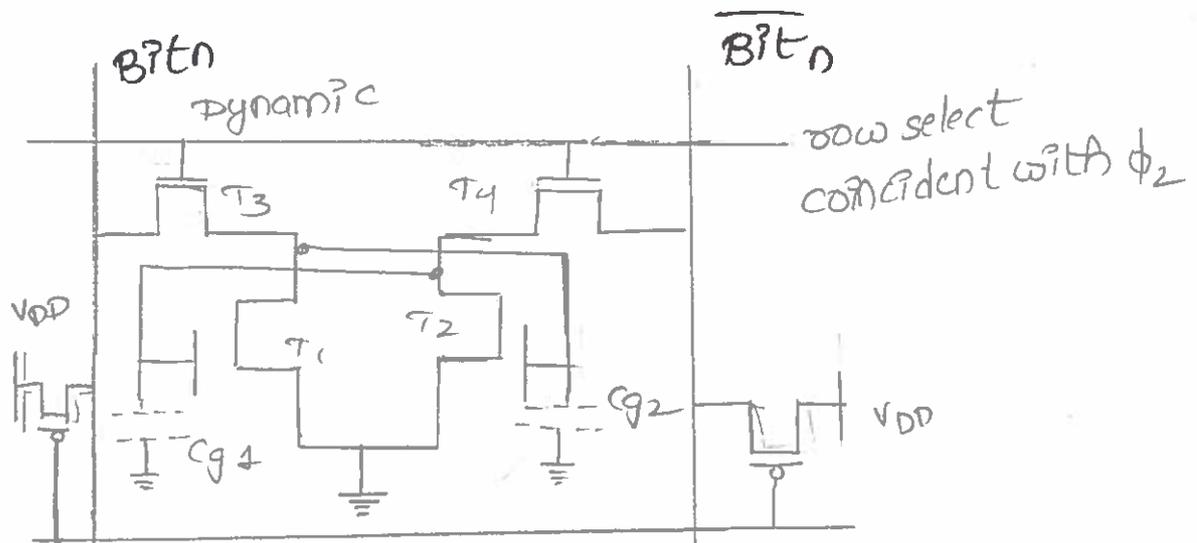
" when such cells are used in RAM arrays, it is necessary to keep the area of each cell

to a minimum and transistors will be minimum size therefore incapable of sinking large charges quickly. RAM arrays therefore generally employ some form of 'sense amplifiers' as shown in Fig (c).

In which T_1, T_2, T_3 & T_4 form a F/F circuit. If we assume the sense line to be inactive, then the state of the bit lines reflected in the charges present on the gate capacitances of T_1 & T_3 w.r.t to V_{DD} such that a 1 will turn off and 0 turn on a Pmos transistor. current flowing from V_{DD} through an on transistor helps to maintain the state of the bit lines and predetermines the state which will be taken up by the sense F/F when the sense line is activated.

single sense amplifier / column will be used to amplify the current sinking capability of the selected memory cell.

Fig (b) indicates an adaption of the basic dynamic cell to form a stable memory cell. Here two additional transistors per bit stored, the transistors T_1 & T_2 can be replaced by an inverter.

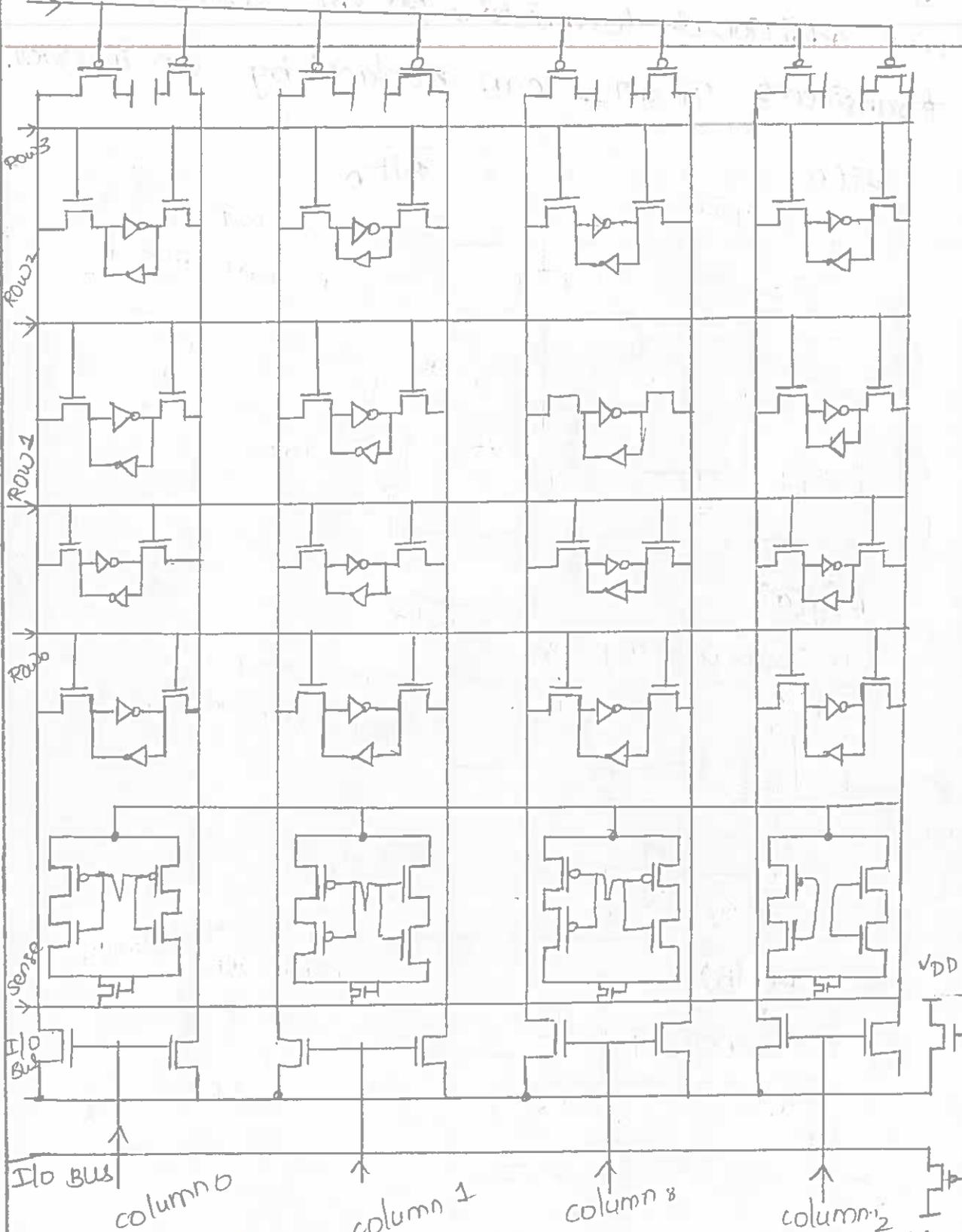


RAM arrays

precharge
V_{DD}

16 bit

CMOS static Memory array



→ CMOS static Memory array consisting
no. of memory cells.

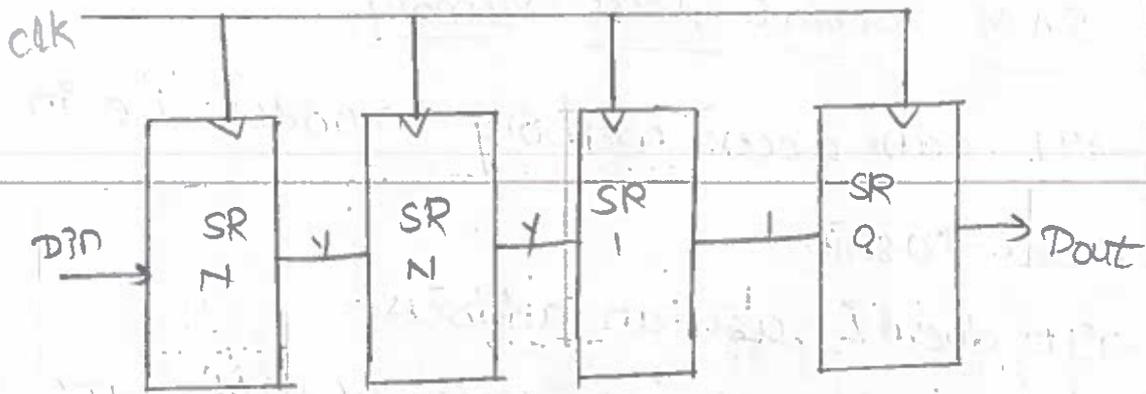
SAM :- Serial Access Memory :-

- > IT can access memory in order i.e in serial fashion
- > IT doesn't use an address
- These are different types of SAM's are available.

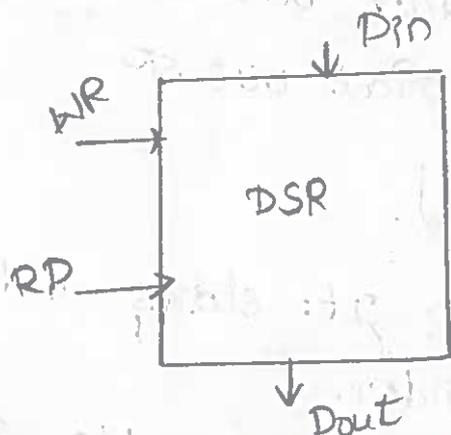
1. Shift Registers
2. Dense Shift Registers
3. Tapped Delay Line Shift Registers
4. Serial in parallel out SR
5. Parallel in serial out SR
6. Queues (FIFO, LIFO).

1. Shift Registers :- It stores and delay the data at 1 bit order.

- > It require clock for enable SR.
- > To achieve N no. of shift registers by cascading N stages of shift registers.
- > Din is the data input
- > Dout is the output data



2. Dense shift registers - It is similar to normal shift registers but it keeps the data in SRAM cells for large shift regs.



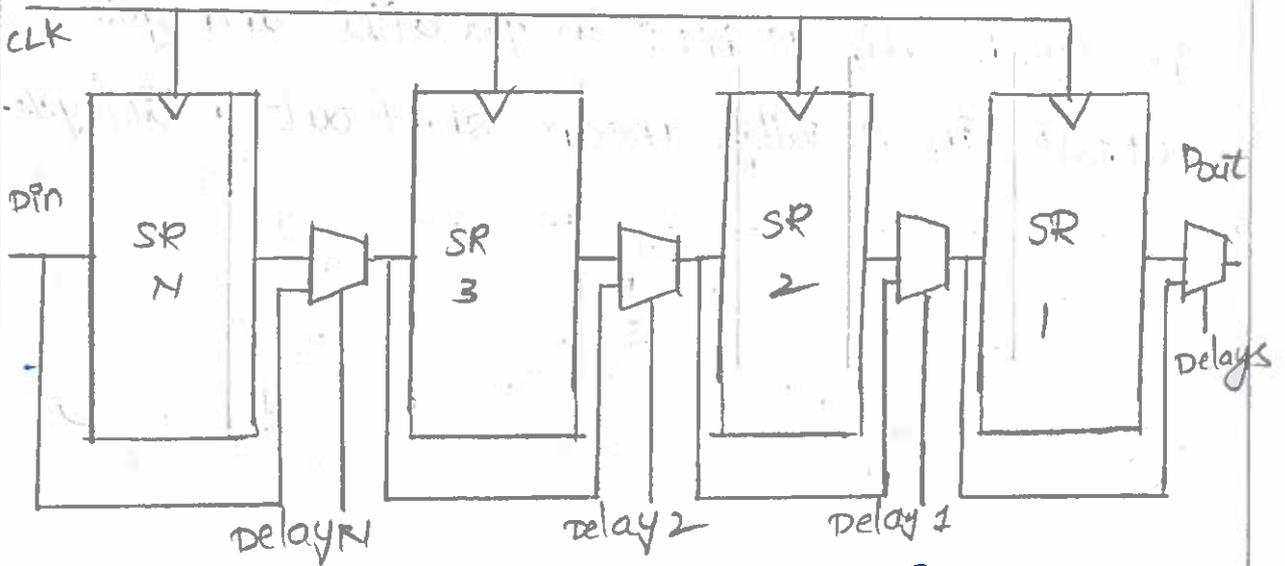
→ It requires additional clocks for read and write operations.

3. Tapped delay line

A Tapped delay line shift register is SR with a programmable N no. of stages

→ It is sent no. of stages with 'N' delay line controls

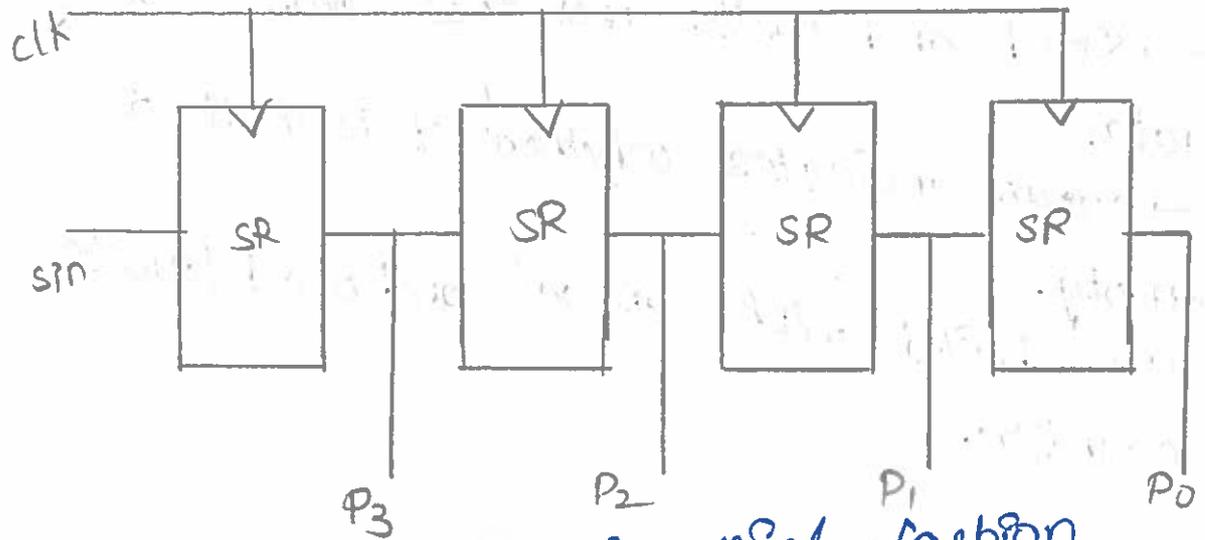
→ These are applied to Multiplexers



→ the i/p data delayed by 1 bit.

4. Serial in Parallel out (SR) (SIPO):-

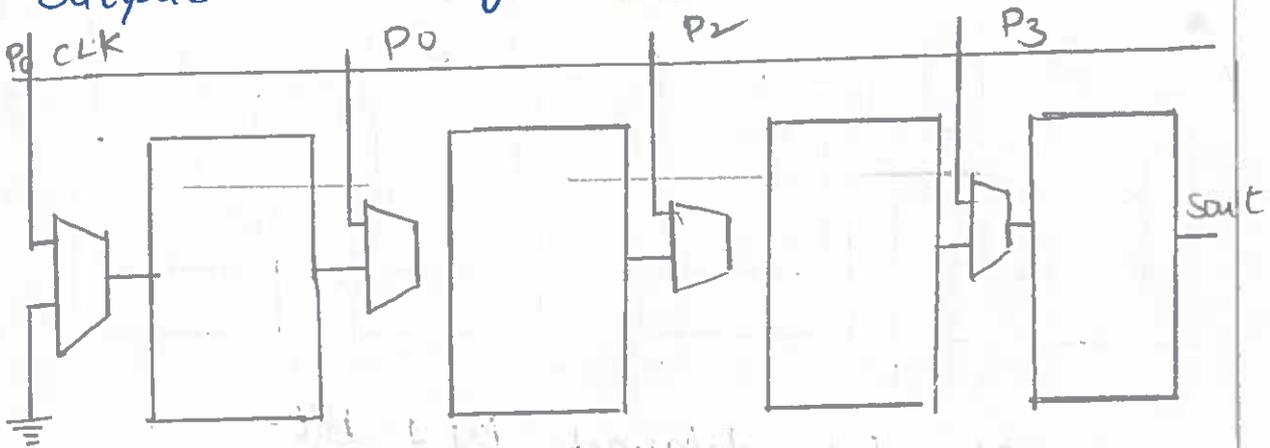
→ It reads 1 bit data in serial order, after N steps presents N. 1 bit parallel o/p's.



→ i/p data applied in serial fashion

(111101) and the o/p will be parallel $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}$

5. PISO:- It is a shift reg, when shift = 0, it loads all N bits in parallel and get output in serially mean shift out 1 bit/cycle.



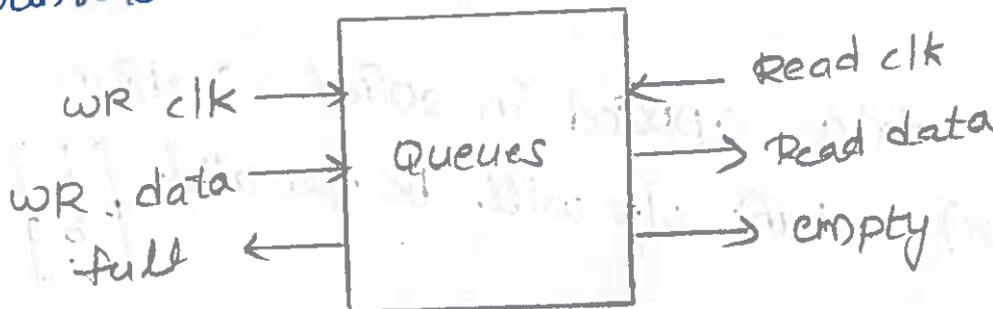
parallel in, serial out

Queues:- It allows data to be read and write at different rates.

→ Read and write uses their own clock data.

→ Queue indicates whether it is full or empty.

→ It build with SRAM and read/write counters.



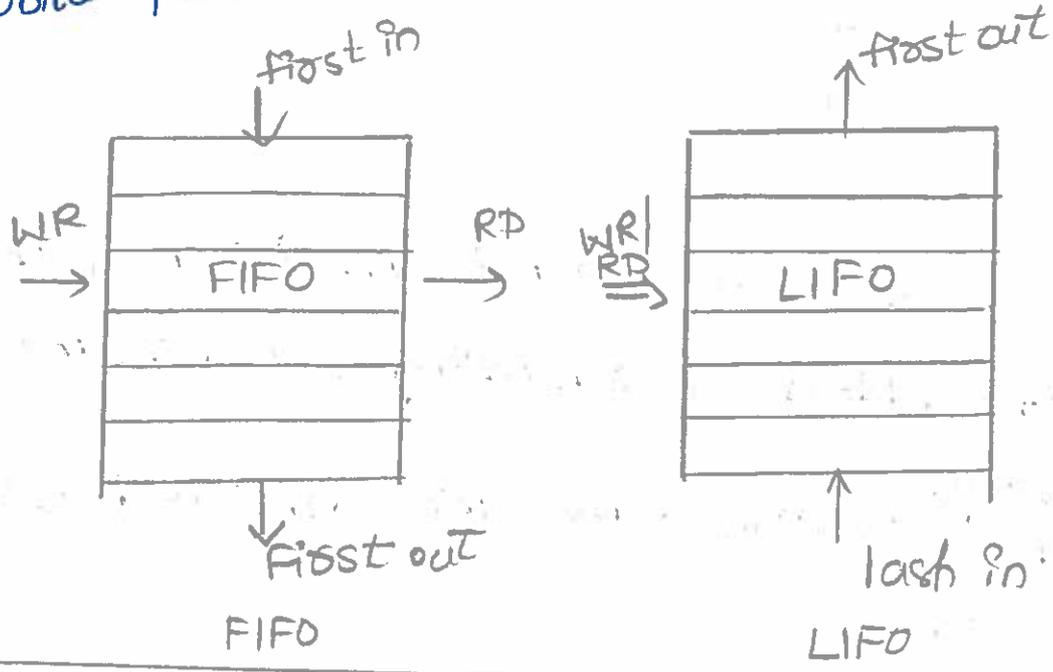
Types of queues: FIFO and LIFO

FIFO:- First in First out

- It initialize read and write pointers to the 1st element.
- If queue is empty, on write, increments write pointer.
- If write almost catches read, queue is full.
- on read, increments read pointer.

LIFO:- Last in first out

- Initialize read and write pointers to the last element.
- It is also called stack
- It use single stack pointer for read and write operations.

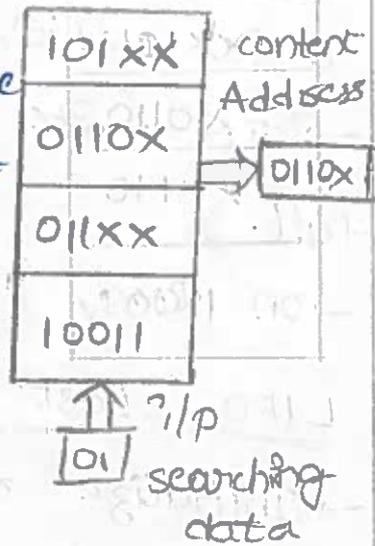


CAM :- content Addressable Memory

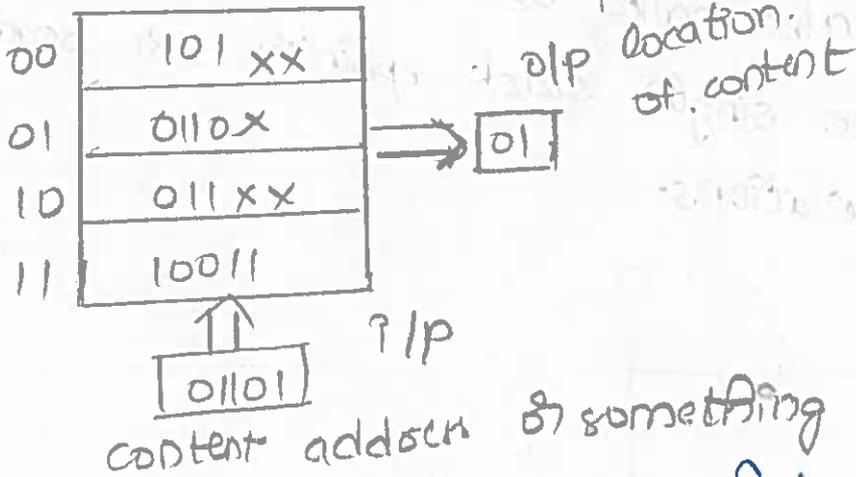
→ It is a special kind of memory
 consider Read operation in traditional memory (RAM) and CAM.

RAM :-

→ input is address location of the content that are interested in it
 → and o/p is content of that address



But in CAM :-



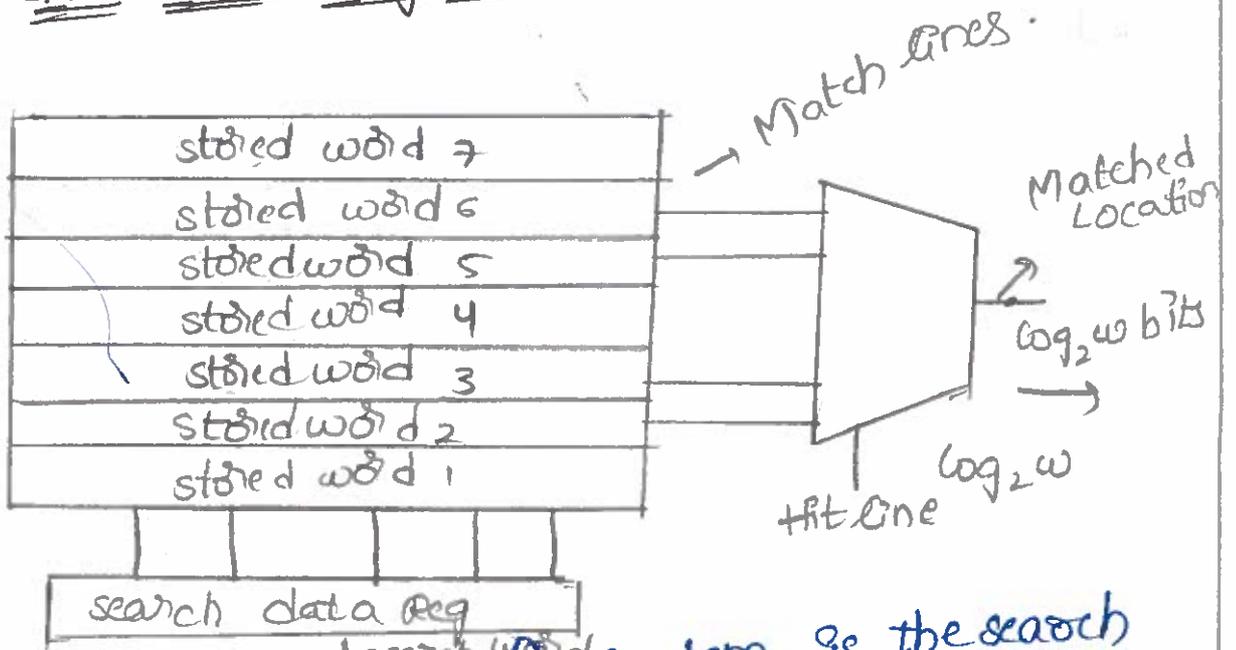
→ It is the reverse, i/p is associated with content address of something stored in the memory.

→ o/p is location where the associated content is stored.

CAM:-

CAM can be used as a search engine we wait to find out matching contents in a data base or table.

CAM Blocks Diagram:-

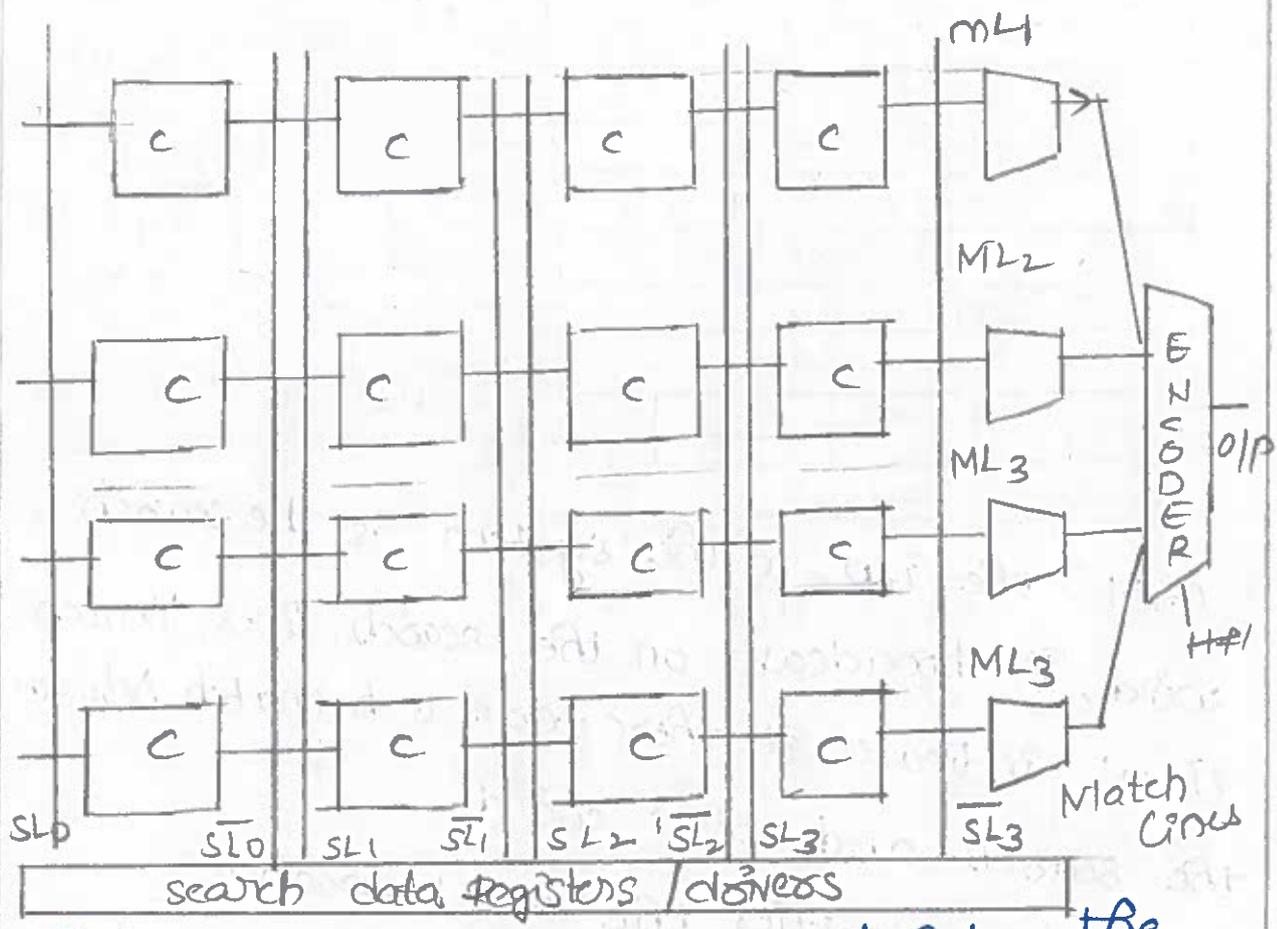


- CAM, the ¹ search word to the system is the search word, if broadcast on the search lines, 'Match lines' indicates if there were a match between the search and stored word.
- > Encoder specifies the match location
- > If multiple matches, a priority encoder selects the 1st match.
- > Hit signal specifies if there is no match.
- > The length of the search word is long range from 36 to 144 bits.

→ Table size ranges are four hundred to 32k and address space is \therefore 7 to 15 bits.

→ Memory size of CAM is 18M bits (single chip)

→ CAM cell consisting of equal to 2 SRAM cells.



→ search data word is loaded into the search data register.

→ All match lines are precharged to high (VDD).

→ search lines drivers broadcast the search word into the diff search lines,

Each CAM compare its stored bit against search bit on the corresponding search lines.

→ Match word that have at least one missing bit, it discharged to GND.

Types

① BCAM :- only store 0's, 1's

APP:- MAC table consultation layer 2 security related VPN

size:- 1MB, 2MB.

② TCAM

stores 0's, 1's, 'x' don't cares

APP:- IP routing (QoS) (PQoS)

4.7MB, 9.4MB, 18.8MB.

Advantages

→ It requires only one clock cycle

→ CAM can be cascaded to increase the memory size.

→ It can adopt new entries.

→ High speed.

Disadvantages:- High cost (several hundred of dollars (CAM) occupy large foot print

on a card board consume excessive power.

ROM Read only Memory or PROM

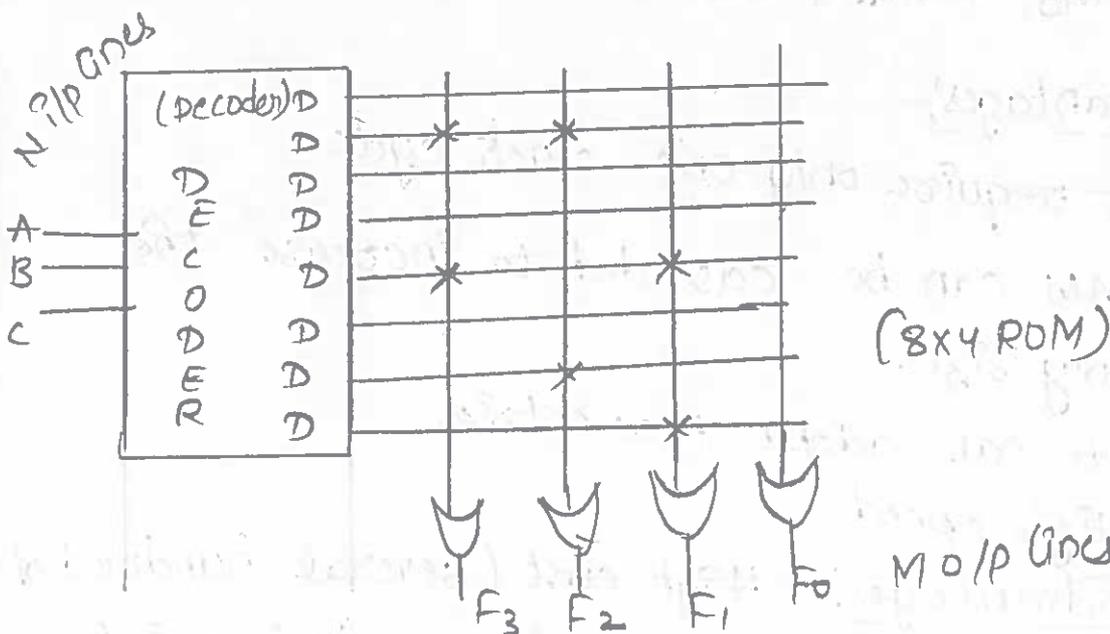
→ It is an important memory in modern computer design.

→ It find applications storing data items that do not change.

→ ROM stores instructions that are used by the CPU. It can also hold programs, that are directly accessed by CPU.

→ the self test can be done, when the computer is turned on.

→ Instructions in ROM can't usually be changed, to change the instructions in ROM by some other special processes.



ROM have \rightarrow N i/p lines

$\rightarrow 2^N$ decoded lines (Minterms).

$\rightarrow M$ o/p lines

\rightarrow It is a Non volatile memory

\rightarrow Fixed AND array is a decoder with 3 i/p's and 8 o/p's.

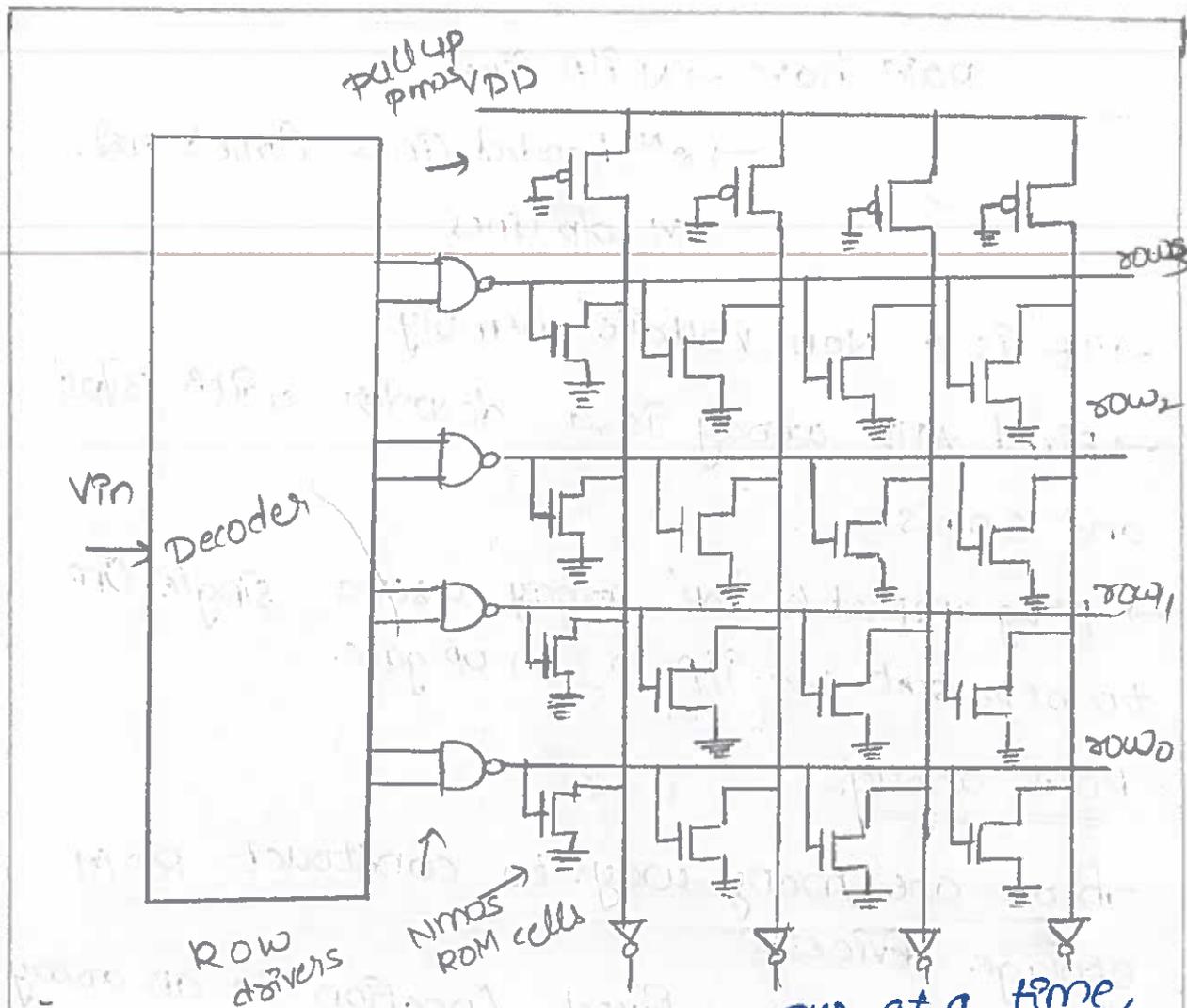
\rightarrow Programmable 'OR' array uses a single line to represent all i/p to an OR gate.

ROM array

There are many ways to construct ROM storage devices.

\rightarrow It is assigned a fixed location in an array of rows and columns.

\rightarrow At each location an NMOS transistor is used to represent '1' bit and the lack of transistor represent zero bit.



→ ROM drivers drive only one row at a time, All the gates in a row of the Nmos ROM cell transistors are attached to the same row driver.

→ when $V_{in} = 0$, At the top of each column, is a pmos transistor that pulls up the column to V_{DD} . and $V_{in} = 1$ Nmos transistors are pull down the entire column to ground.

→ Both NMOS, PMOS are minimum size but $\beta_n = 2.5 \beta_p$

→ so the NMOS transistors win the contest

to pull the entire column to GND.
 → the logic levels created by the context b/w the NMOS and PMOS transistors are not very good. so, an amplifier (inverter) is placed at the base of each column to clean up the signal.

EEPROM

It uses mos circuitry to store data, the stored data as charge & no charge on an insulating layer. (1) (0)

→ Layer is very thin (e.g. 200Å) hence low voltage can be used (15 to 20V) to move charges.

→ It allows selective erasing at register level rather than erasing all the information.

→ erased time is less than 10m sec. compared to erase PROM (EPROM).

→ It is actually ROM but are reprogrammable after erase.

Advantages

- Fast turn around time, Fast access time
- permanent storage of data
- Low power consumption
- Good density

→ on board programming is possible.

Flash Memory:-

→ Entire content of flash memory can be erased by less than second.

→ The erasing process can be done by pulse hence it is called as flash EEPROM.

→ It is a non volatile memory

→ Compared to EEPROM, write can be done in blocks at a time

→ It has high performance to large writes and very poor performance for small writes.

→ It is much more compact than other memory devices.

→ Fast read time than hard disk.

→ storing memory is done by 2 methods
single cell multiple cell

store 1 bit information

store multiple bit

→ It is a semiconductor memory with access time 100ns.

ADP

- Used in Pen drives
- Digital cameras
- Mobile phones, TV settop boxes

Advantages:-

- Non volatile, reprogrammable
- Fast read access time
- Less cost / M Bytes
- very compact.

