UNIT-V

MEMORY DEVICES AND DIGITAL INTEGRATED CIRCUITS

INTRODUCTION

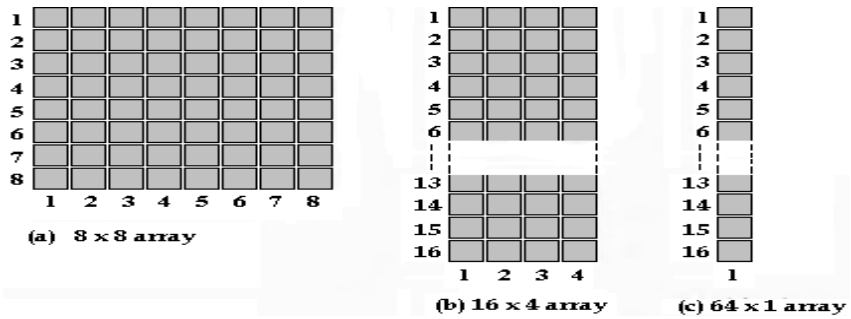
A memory unit is a collection of storage cells with associated circuits needed to transfer information in and out of the device. The binary information is transferred for storage and from which information is available when needed for processing. When data processing takes place, information from the memory is transferred to selected registers in the processing unit. Intermediate and final results obtained in the processing unit are transferred back to be stored in memory.

Units of Binary Data: Bits, Bytes, Nibbles and Words

As a rule, memories store data in units that have from one to eight bits. The smallest unit of binary data is the **bit**. In many applications, data are handled in an 8-bit unit called a **byte** or in multiples of 8-bit units. The byte can be split into two 4-bit units that are called **nibbles**. A complete unit of information is called a **word** and generally consists of one or more bytes. Some memories store data in 9-bit groups; a 9-bit group consists of a byte plus a parity bit.

Basic Semiconductor Memory Array

Each storage element in a memory can retain either a 1 or a 0 and is called a **cell**. Memories are made up of arrays of cells, as illustrated in Figure below using 64 cells as an example. Each block in the memory array represents one storage cell, and its location can be identified by specifying a row and a column.

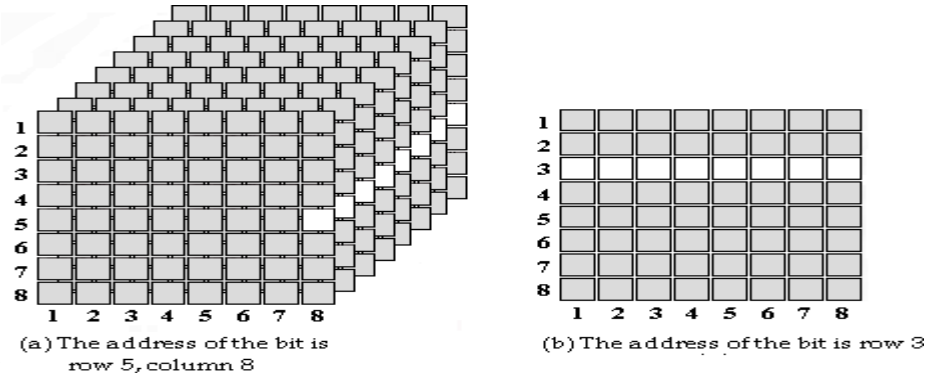


A 64-cell memory array organized in three different ways

Memory Address and Capacity

The *location* of a unit of data in a memory array is called its **address**. For example, in Figure (a), the address of a bit in the 3-dimensional array is specified by the row and

column. In Figure (b), the address of a byte is specified only by the row in the 2-dimensional array. So, as you can see, the address depends on how the memory is organized into units of data. Personal computers have random-access memories organized in bytes. This means that the smallest group of bits that can be addressed is eight.



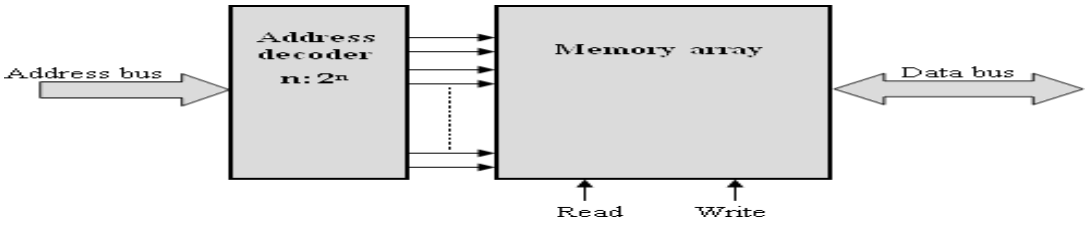
Examples of memory address

The **capacity** of a memoryis the total number of data units that can be stored. For example, in the bit-organized memory array in Figure (a), the capacity is 64 bits. In the byte-organized memory array in Figure (b), the capacity is 8 bytes, which is also 64 bits. Computer memories typically have 256 MB (megabyte) or more of internal memory.

Basic Memory Operations

Since a memory stores binary data, data must be put into the memory and data must be copied from the memory when needed. The write operation puts data into a specified address in the memory, and the read operation copies data out of a specified address in the memory. The addressing operation, which is part of both the write and the read operations, selects the specified memory address.

Data units go into the memory during a write operation and come out of the memory during a read operation on a set of lines called the *data bus*. As indicated in Figure, the data bus is bidirectional, which means that data can go in either directional (into the memory or out of the memory).



Block diagram of memory operation

For a write or a read operation, an address is selected by placing a binary code representing the desired address on a set of lines called the address bus. The address code is decoded internally and the appropriate address is selected. The number of lines in the address bus depends on the capacity of the memory. For example, a 15-bit address code can select 32,768 locations (2_{15}) in the memory; a 16-bit address code can select 65,536 locations (2_{16}) in the memory and so on.

In personal computers a 32-bit address bus can select 4,294,967,296 locations (2_{32}), expressed as 4GB.

Write Operation

To store a byte of data in the memory, a code held in the address register is placed on the address bus. Once the address code is on the bus, the address decoder decodes the address and selects the specified location in the memory. The memory then gets a write command, and the data byte held in the data register is placed on the data bus and stored in the selected memory address, thus completing the write operation. When a new data byte is written into a memory address, the current data byte stored at that address is overwritten (replaced with a new data byte).

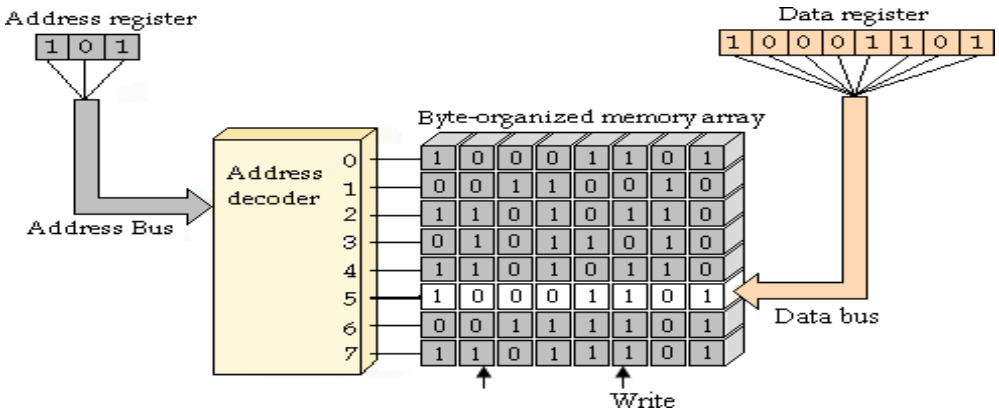


Illustration of the Write operation

Read Operation

A code held in the address register is placed on the address bus. Once the address code is on the bus, the address decoder decodes the address and selects the specified location in the memory. The memory then gets a read command, and a "copy" of the data byte that is stored in the selected memory address is placed on the

data bus and loaded into the data register, thus completing the read operation.

When a data byte is read from a memory address, it also remains stored at that address. This is called *nondestructive* read.

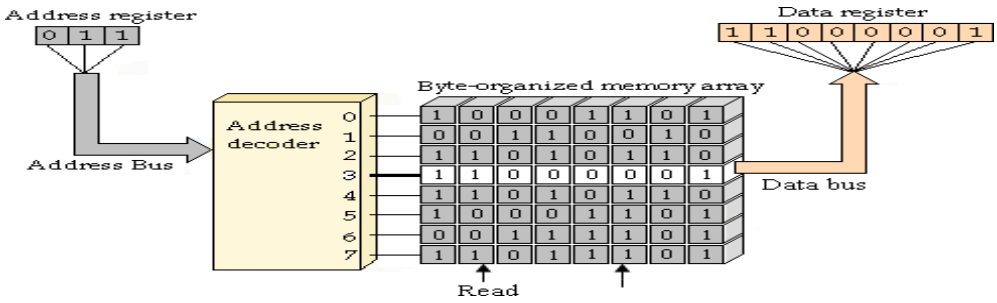


Illustration of the Read operation

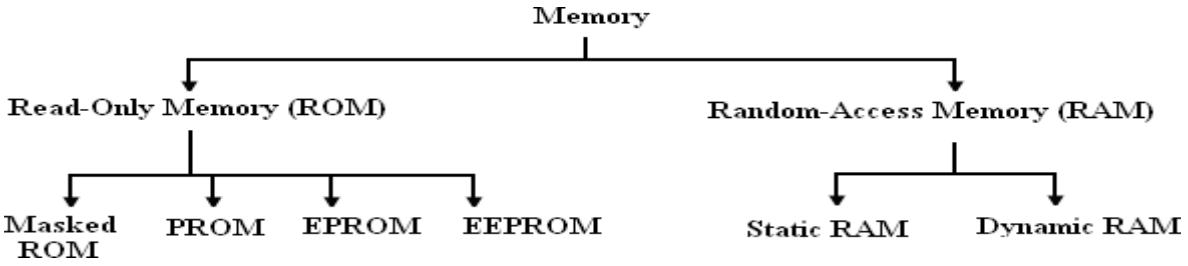
Classification of Memories

There are two types of memories that are used in digital systems:

- Random-Access Memory (RAM),
- Read-Only Memory (ROM).

RAM (random-access memory) is a type of memory in which all addresses are accessible in an equal amount of time and can be selected in any order for a read or write operation. All RAMs have both read and write capability. Because RAMs lose stored data when the power is turned off, they are ***volatile*** memories.

ROM (read-only memory) is a type of memory in which data are stored permanently or semi permanently. Data can be read from a ROM, but there is no write operation as in the RAM. The ROM, like the RAM, is a random-access memory but the term RAM traditionally means a random-access read/write memory. Because ROMs retain stored data even if power is turned off, they are ***nonvolatile*** memories.



Classification of memories

RANDOM-ACCESS MEMORIES (RAMS)

RAMs are read/write memories in which data can be written into or read from any selected address in any sequence. When a data unit is written into a given

address in the RAM, the data unit previously stored at that address is replaced by the new data unit. When a data unit is read from a given address in the RAM, the data unit remains stored and is not erased by the read operation. This nondestructive read operation can be viewed as copying the content of an address while leaving the content intact.

A RAM is typically used for short-term data storage because it cannot retain stored data when power is turned off.

The two categories of RAM are the ***static* RAM** (SRAM) and the ***dynamic* RAM** (DRAM). Static RAMs generally use flip-flops as storage elements and can therefore store data indefinitely *as long as dc power is applied*. Dynamic RAMs use capacitors as storage elements and cannot retain data very long without the capacitors being recharged by a process called **refreshing**. Both SRAMs and DRAMs will lose stored data when dc power is removed and, therefore, are classified as ***volatile memories***.

Data can be read much faster from SRAMs than from DRAMs. However, DRAMs can store much more data than SRAMs for a given physical size and cost because the DRAM cell is much simpler, and more cells can be crammed into a given chip area than in the SRAM.

Static RAM (SRAM)

Storage Cell:

All static RAMs are characterized by flip-flop memory cells. As long as dc power is applied to a static memory cell, it can retain a 1 or 0 state indefinitely. If power is removed, the stored data bit is lost.

The cell is selected by an active level on the Select line and a data bit (l or 0) is written into the cell by placing it on the Data in line. A data bit is read by taking it off the Data out line.

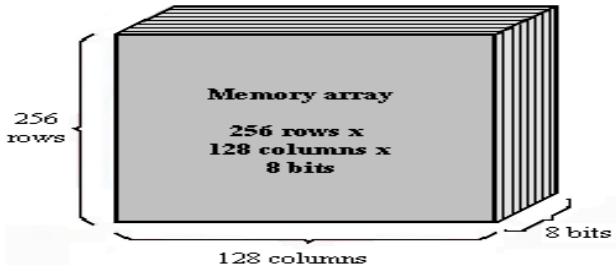
Basic SRAM Organisation:

Basic Static Memory Cell

Array

The memory cells in a SRAM are organized in rows and columns. All the cells in a row share the same Row Select line. Each set of Data in and Data out lines go to each cell in a given column and are connected to a single data line that serves as both an input and output (Data I/O) through the data input and data output buffers.

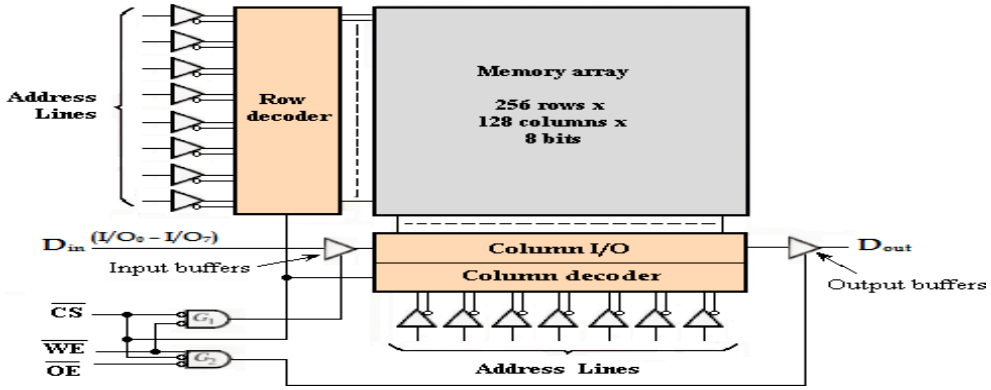
SRAM chips can be organized in single bits, nibbles (4 bits), bytes (8 bits), or multiple bytes (16, 24, 32 bits, etc.). The memory cell array is arranged in 256 rows and 128 columns, each with 8 bits as shown below. There are actually 2_{15} = 32,768 addresses and each address contains 8 bits. The capacity of this example memory is 32,768 bytes (typically expressed as 32 kbytes).



Memory array configuration

Operation:

The SRAM works as follows. First, the chip select, CS, must be LOW for the memory to operate. Eight of the fifteen address lines are decoded by the row decoder to select one of the 256 rows. Seven of the fifteen address lines are decoded by the column decoder to select one of the 128 8-bit columns.



Memory block diagram

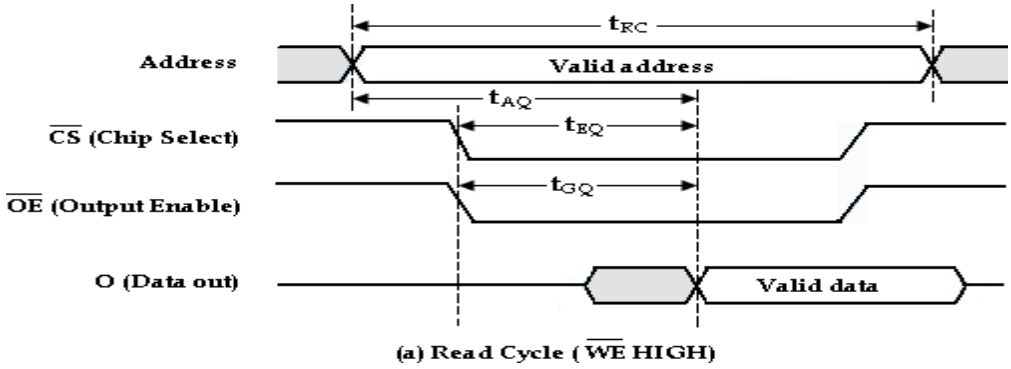
Read: In the READ mode, the write enable input, WE' is HIGH and the output enable, OE' is LOW. The input tristate buffers are disabled by gate **G_1**, and the column output tristate buffers are enabled by gate **G_2**. Therefore, the eight data bits from the selected address are routed through the column I/O to the data lines (I/O_1 through I/O_7), which are acting as data output lines.

Write: In the WRITE mode, WE' is LOW and OE' is HIGH. The input buffers are enabled by gate G_1, and the output buffers are disabled by gate G_2. Therefore the

eight input data bits on the data lines are routed through the input data control and the column I/O to the selected address and stored.

Read and Write Cycles: For the read cycle shown in part (a), a valid address code is applied to the address lines for a specified time interval called the ***read cycle time***, t_{WC}. Next, the chip select (CS) and the output enable (DE) inputs go LOW. One time interval after the DE input goes LOW; a valid data byte from the selected address appears on the data lines. This time interval is called the ***output enable access time***, t_{GQ}. Two other access times for the read cycle are the ***address access time***, t_{AQ}, measured from the beginning of a valid address to the appearance of valid data on the data lines and the **chip enable access time**, t_{EQ}, measured from the HIGH-to-LOW transition of CS to the appearance of valid data on the data lines.

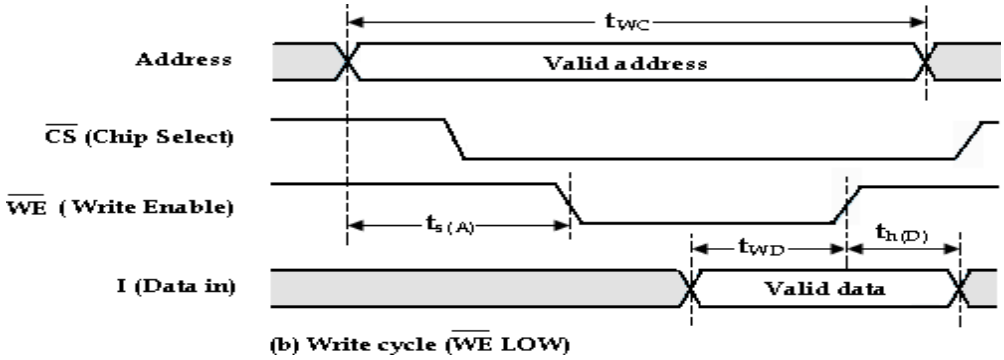
During each read cycle, one unit of data, a byte in this case is read from the memory.



(a) Read Cycle ($\overline{WE}$ HIGH)

For the write cycle shown in Figure (b), a valid address code is applied to the address lines for a specified time interval called the ***write cycle time***, t_{WE} . Next, the chip select (CS) and the write enable (WE) in puts go LOW. The required time interval from the beginning of a valid address until the WE input goes LOW is called the ***address setup time***, $t_{s(A)}$. The time that the WE input must be LOW is the write pulse width. The time that the input WE must remain LOW after valid data are applied to the data inputs is designated t_{WD}; the time that the valid input data must remain on the data lines after the WE input goes HIGH is the data hold time, $t_{h(D)}$.

During each write cycle, one unit of data is written into the memory.



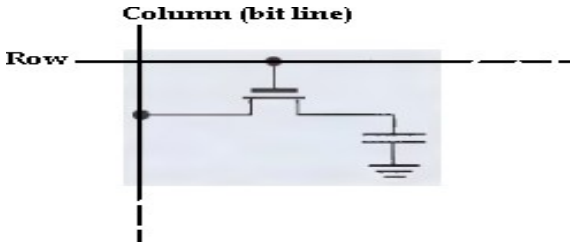
(b) Write cycle ($\overline{WE}$ LOW)

Dynamic RAM (DRAM)

Dynamic RAM Cell:

Dynamic memory cells store a data bit in the form of electric charges on capacitors. The basic storage device in DRAM is not a flip-flop but a simple MOSFET and a capacitor.

The advantage of this type of cell is that it is very simple, thus allowing very large memory arrays to be constructed on a chip at a lower cost per bit. The disadvantage is that the storage capacitor cannot hold its charge over an extended period of time and will lose the stored data bit unless its charge is refreshed periodically. To refresh requires additional memory circuitry and complicates the operation of the DRAM.



DRAM memory cell

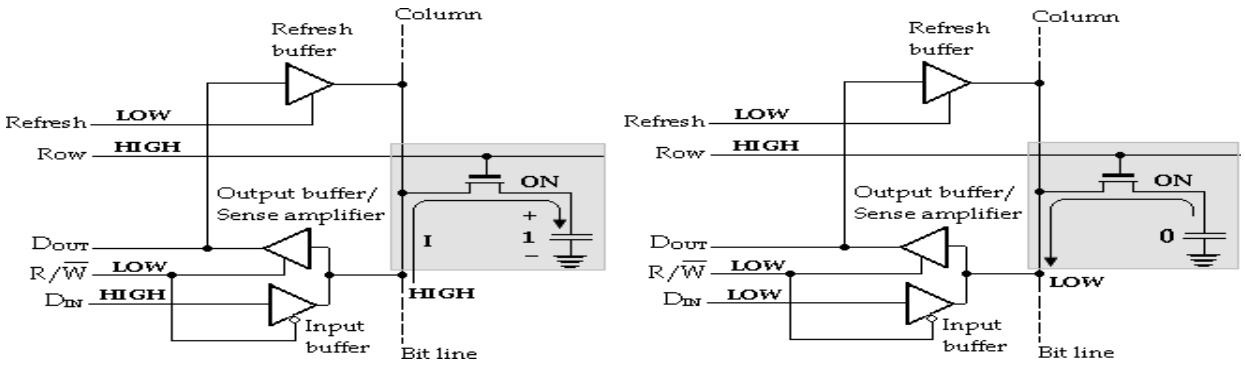
In DRAM memory cell, a bit of data is stored as charge on storage capacitor, where the presence or absence of charge determines the value of the stored bit 1 or 0. The DRAM cell includes a single MOS transistor (MOSFET) and a capacitor. When column line and row line go high, the MOSFET conducts and charges the capacitor. When the column and row lines go low, the MOSFET opens and the capacitor retains its charge. In this way it stores 1 bit.

Operation: The DRAM cell consists of 3 tri-state buffers: Input buffer, Output buffer and refresh buffer. Input and output buffers are enabled and disabled by controlling

R/W' line. When R/W'= 0, input buffer is enabled and output buffer is disabled. When R/W'= 1, input buffer is disabled and output buffer is enabled.

(i) **Write:** To enable write operation R/W' line is made low, which enables input buffer and disables output buffer. To write a 1 into the cell, the D_{IN} line is high and MOSFET is turned ON by a high on the row line. This allows the capacitor to charge to a positive voltage. When 0 is to be stored, a low is applied to the D_{IN} line. The capacitor remains unchanged or if it is storing a 1, it discharges.

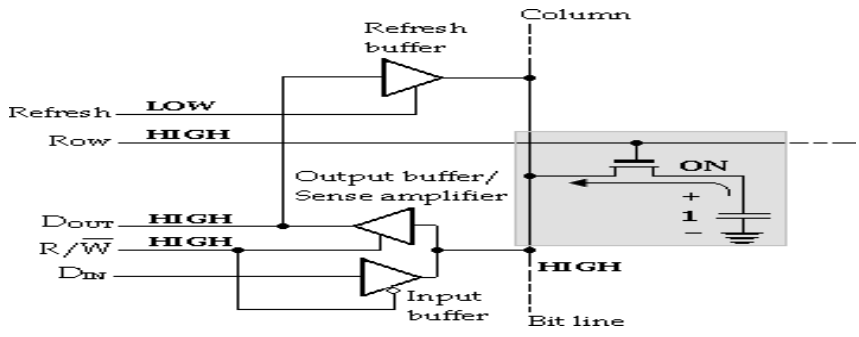
When the row line is made low, the transistor turns OFF and disconnects the capacitor from the data line, thus storing the charge (1 or 0) on the capacitor.



(a) **Writing a 1 into the memory cell** **(b)** **Writing a 0 into the memory cell**

(ii) Read:

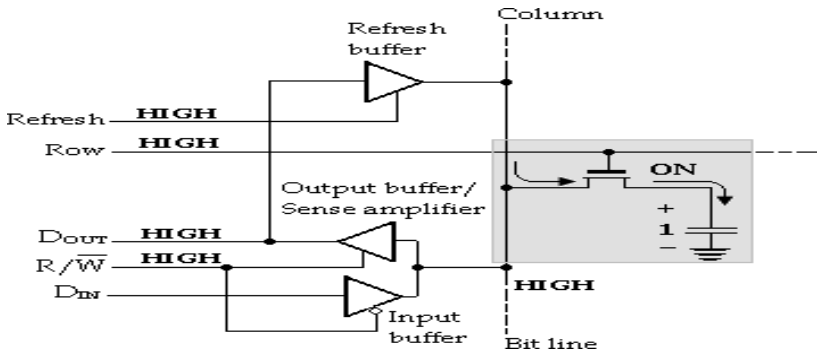
To read data from the cell, the R/W' line is made HIGH, which enables output buffer and disables input buffer. When the row line is made HIGH, the transistor turns ON and connects the capacitor to the D_{OUT} line through output buffer.



Reading a 1 from the memory cell

(iii) **Refresh:** For refreshing the memory cell, the R/W line is HIGH, the row line is HIGH, and the refresh line is HIGH. The transistor turns on, connecting the capacitor to the bit line. The output buffer is enabled, and the stored data bit is

applied to the input of the refresh buffer, which is enabled by the HIGH on the refresh input. This produces a voltage on the bit line corresponding to the stored bit thus refreshing the capacitor.



Refreshing a stored 1

READ- ONLY MEMORIES (ROMS)

A ROM contains permanently or semi-permanently stored data, which can be read from the memory but either cannot be changed at all or cannot be changed without specialization equipment. A ROM stores data that are used repeatedly in system applications, such as tables, conversions, or programmed instructions for system initialization and operation. ROMs retain stored data when the power is OFF and are therefore nonvolatile memories.

The ROMs are classified as follows:

i. Masked ROM (ROM)

ii. Programmed ROM (PROM)

iii. Erasable PROM (EPROM)

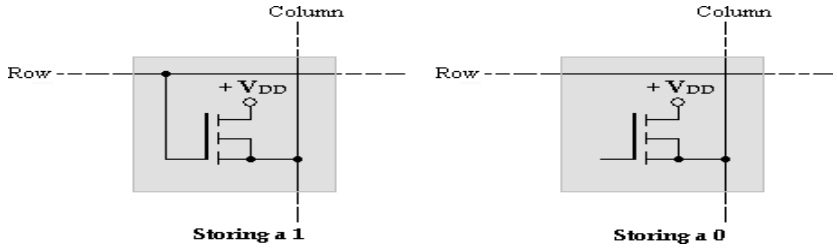
iv. Electrically Erasable PROM (EEPROM)

Masked ROM

The mask ROM is usually referred to simply as a ROM. It is permanently programmed during the manufacturing process to provide widely used standard functions, such as popular conversions, or to provide user-specified functions. Once the memory is programmed, it cannot be changed.

Most IC ROMs utilize the presence or absence of a transistor connection at a row/column junction to represent a 1 or a 0. The presence of a connection from a

row line to the gate of a transistor represents a 1 at that location because when the row line is taken HIGH; all transistors with a gate connection to that row line turn on and connect the HIGH (1) to the associated column lines.



ROM Cells

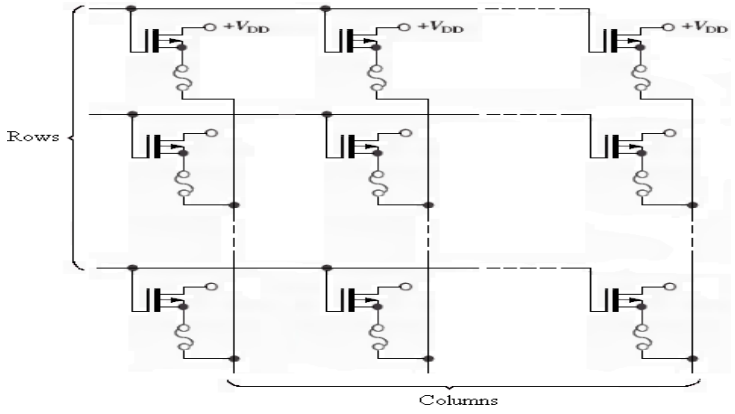
At row/column junctions where there are no gate connections, the column lines remain LOW (0) when the row is addressed.

PROM (Programmable Read-Only Memory)

The PROM (Programmable Read-only memory), comes from the manufacturer unprogrammed and are custom programmed in the field to meet the user's needs.

A PROM uses some type of fusing process to store bits, in which a memory link is burned open or left intact to represent a 0 or a 1. The fusing process is irreversible; once a PROM is programmed, it cannot be changed.

The fusible links are manufactured into the PROM between the source of each cell's transistor and its column line. In the programming process, a sufficient current is injected through the fusible link to bum it open to create a stored O. The link is left intact for a stored 1. All drains are commonly connected to V_{DD}.



PROM array with fusible links

Three basic fuse technologies used in PROMs are metal links, silicon links, and pn junctions. A brief description of each of these follows.

1. **Metal links** are made of a material such as ***nichrome***. Each bit in the memory array is represented by a separate link. During programming, the link is either "blown" open or left intact. This is done basically by first addressing a given cell and then forcing a sufficient amount of current through the link to cause it to open. When the fuse is intact, the memory cell is configured as a logic 1 and when fuse is blown (open circuit) the memory cell is logic 0.
2. **Silicon links** are formed by narrow, notched strips of ***polycrystalline silicon***. Programming of these fuses requires melting of the links by passing a sufficient amount of current through them. This amount of current causes a high temperature at the fuse location that oxidizes the silicon and forms insulation around the now-open link.
3. **Shorted junction**, or avalanche-induced migration, technology consists basically of two pn junctions arranged back-to-back. During programming, one of the diode junctions is avalanched, and the resulting voltage and heat cause aluminum ions to migrate and short the junction. The remaining junction is then used as a forward- biased diode to represent a data bit.

EPROM (Erasable Programmable ROM)

An EPROM is an erasable PROM. Unlike an ordinary PROM, an EPROM can be reprogrammed if an existing program in the memory array is erased first.

An EPROM uses an NMOSFET array with an isolated-gate structure. The isolated transistor gate has no electrical connections and can store an electrical charge for indefinite periods of time. The data bits in this type of array are represented by the presence or absence of a stored gate charge. Erasure of a data bit is a process that removes the gate charge.

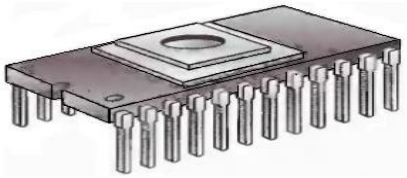
Two basic types of erasable PROMs are the ultraviolet erasable PROM (UV EPROM) and the electrically erasable PROM (EEPROM).

- **UV EPROM:** You can recognize the UV EPROM device by the transparent quartz lid on the package, as shown in Figure below. The isolated

gate in the FET of an ultraviolet EPROM is "floating" within an oxide insulating material. The programming process causes electrons to be removed from the floating gate. Erasure is done by exposure of the memory array chip to high-intensity ultraviolet radiation through the quartz window on top of the package.

The positive charge stored on the gate is neutralized after several minutes to an hour of exposure time. In EPROM's, it is not possible to erase selective information, when erased the entire information is lost. The chip can be reprogrammed.

It is ideally suited for product development, college laboratories, etc.



Ultraviolet Erasable PROM

During programming, address and datas are applied to address and data pins of the EPROM. The program pulse is applied to the program input of the EPROM. The program pulse duration is around 50msec and its amplitude depends on EPROM IC. It is typically 11.5V to 25V.

In EPROM, it is possible to program any location at any time- either individually, sequentially or at random.

EEPROM (Electrically Erasable PROM)

The EEPROM (Electrically Erasable PROM), also uses MOS circuitry. Data is stored as charge or no charge on an insulating layer, which is made very thin (< 200Å). Therefore a voltage as low as 20- 25V can be used to move charges across the thin barrier in either direction for programming or erasing ROM.

An electrically erasable PROM can be both erased and programmed with electrical pulses. Since it can be both electrically written into and electrically erased, the EEPROM can be rapidly programmed and erased in-circuit for reprogramming.

It allows selective erasing at the register level rather than erasing all the information, since the information can be changed by using electrical signals.

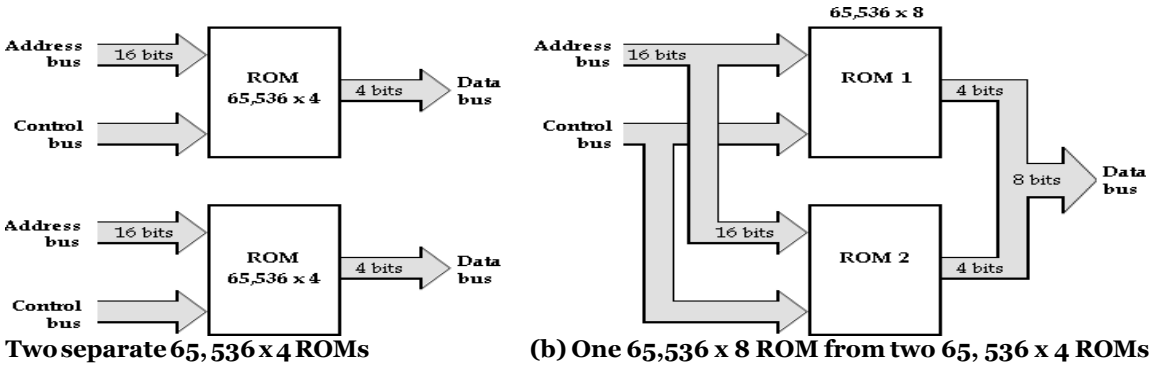
It has chip erase mode by which the entire chip can be erased in 10 msec. Hence EEPROM's are most expensive.

MEMORY EXPANSION

Available memory can be expanded to increase the word length (number of bits in each address) or the word capacity (number of different addresses) or both. Memory expansion is accomplished by adding an appropriate number of memory chips to the address, data, and control buses.

Word Length Expansion

To increase the word length of a memory, the number of bits in the data bus must be increased. An 8-bit word length can be achieved by using two memories each with 4-bit words as illustrated in Figure below. The 16-bit address bus is commonly connected to both memories so that the combination memory still has the same number of addresses (2_{16} = 65,536) as each individual memory. The 4-bit data buses from the two memories are combined to form an 8-bit data bus. Now when an address is selected, eight bits are produced on the data bus-four from each ROM.

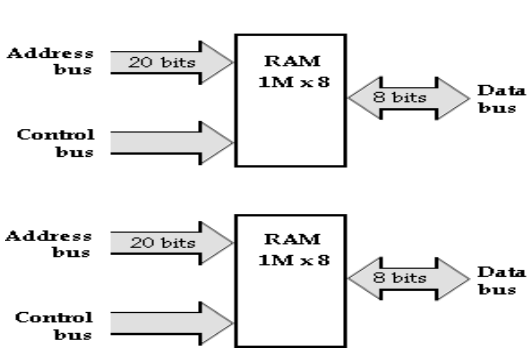


Illustrate of word-length expansion

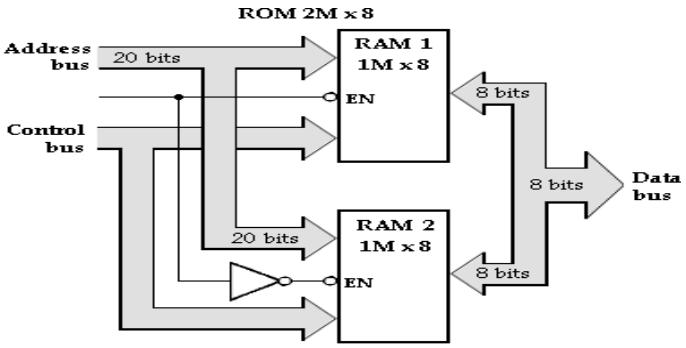
Word-Capacity Expansion: when memories are expanded to increase the word capacity, the number of addresses is increased. Two 1M x 8 RAMs are expanded to form a 2M x 8 memory is shown below.

Each individual memory has 20 address bits to select its 1,048,576 addresses. The expanded memory has 2,097,152 addresses and therefore requires 21 address bits, as shown in part (b). The twenty-first address bit is used to enable the

appropriate memory chip. The data bus for the expanded memory remains eight bits wide.



(a) Individual memories each store 1,048,576 8-bit words

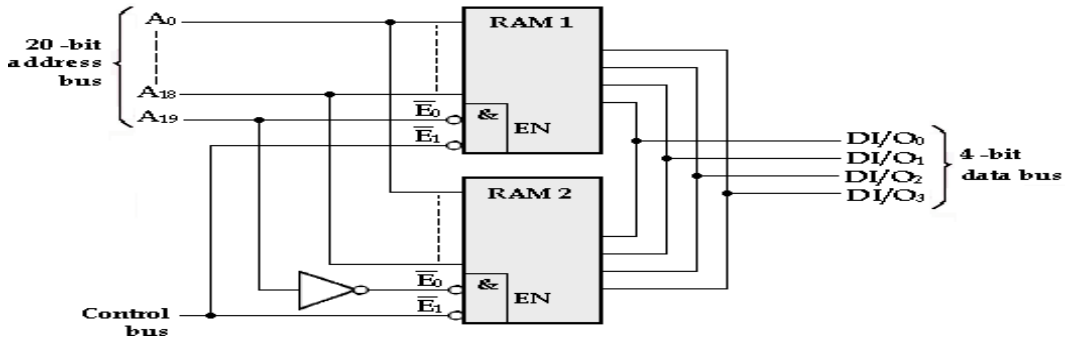


(b) Memories expanded to form a 2M x 8 RAM requiring a 21-bit address bus

Word capacity Expansion

1. Use 512k x 4 RAMs to implement a 1M x 4 memory.

The expanded addressing is achieved by connecting the chip enable (E_0') input to the twentieth address bit (A_{19}). Input (E_1') is used as an enable input common to both memories. When the twentieth address bit (A_{19}) is LOW, RAM 1 is selected (RAM 2 is disabled), and the nineteen lower-order address bits (A_0 –A_{18}) access each of the addresses in RAM 1. When the twentieth address bit (A_{19}) is HIGH, RAM 2 is enabled by a LOW on the inverter output (RAM 1 is disabled), and the nineteen lower-order address bits (A_0 – A_{18}) access each of the RAM 2 addresses.



1M x 4 RAM using 512K x 4 RAM

Advantages of RAM:

1. Fast operating speed **(< 150 nsec)**,

2. Low power dissipation **(< 1mW**),

3. Economy,
4. Compatibility,
5. Non-destructive read-out.

Advantages of ROM:

1. Ease and speed of design,
2. Faster than MSI devices (PLD and FPGA)
3. The program that generates the ROM contents can easily be structured to handle unusual or undefined cases,
4. A ROM's function is easily modified just by changing the stored pattern, usually without changing any external connections,
5. More economical.

Disadvantages of ROM:

1. For functions more than 20 inputs, a ROM based circuit is impractical because of the limit on ROM sizes that are available.
2. For simple to moderately complex functions, ROM based circuit may be costly: consume more power; run slower.

Comparison between RAM and ROM:

S.No	RAM	ROM
1	RAMs have both read and write capability.	ROMs have only read operation.
2	RAMs are volatile memories.	ROMs are non-volatile memories.
3	They lose stored data when the power is turned OFF.	They retain stored data even if power is turned off.
4	**Types**: SRAM, DRAM, EEPROM	**Types**: PROM, EPROM.

Comparison between SRAM and DRAM:

S.No	Static RAM	Dynamic RAM
1	It contains less memory cells	It contains more memory cells per unit area.

	per unit area.	
2	Its access time is less, hence faster memories.	Its access time is greater than static RAM
3	It consists of number of flip-flops. Each flip-flop stores one bit.	It stores the data as a charge on the capacitor. It consists of MOSFET and capacitor for each cell.
4	Refreshing circuitry is not required.	Refreshing circuitry is required to maintain the charge on the capacitors every time after every few milliseconds. Extra hardware is required to control refreshing.
5	Cost is more	Cost is less.

Comparison of Types of Memories:

Memory type	Non- Volatile	High Density	One-Transistor cell	In-system writability
SRAM	No	No	No	Yes
DRAM	No	Yes	Yes	Yes
ROM	Yes	Yes	Yes	No
EPROM	Yes	Yes	Yes	No
EEPROM	Yes	No	No	Yes

PROGRAMMABLE LOGIC DEVICES:

INTRODUCTION:

A combinational PLD is an integrated circuit with programmable gates divided into an AND array and an OR array to provide an AND-OR sum of product implementation. The PLD's can be reprogrammed in few seconds and hence gives more flexibility to experiment with designs. Reprogramming feature of PLDs also makes it possible to accept changes/modifications in the previously design circuits. The advantages of using programmable logic devices are:

1. Reduced space requirements.
2. Reduced power requirements.

3. Design security.
4. Compact circuitry.
5. Short design cycle.
6. Low development cost.
7. Higher switching speed.
8. Low production cost for large-quantity production.

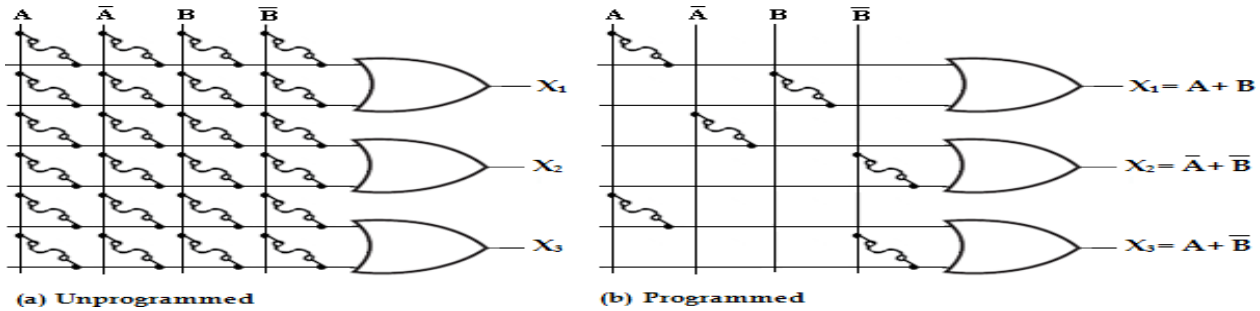
According to architecture, complexity and flexibility in programming in PLD's are classified as—

- PROMs : Programmable Read Only memories,
- PLAs : Programmable Logic Arrays,
- PAL : Programmable Array Logic
- FPGA : Field Programmable Gate Arrays,
- CPLDs : Complex Programmable Logic Devices.

Programmable Arrays: All PLDs consists of programmable arrays. A programmable array is essentially a grid of conductors that form rows and columns with a fusible link at each cross point. Arrays can be either fixed or programmable.

The OR Array: It consists of an array of OR gates connected to a programmable matrix with fusible links at each cross point of a row and column, as shown in the figure below. The array can be programmed by blowing fuses to eliminate selected variables from the output functions. For each input to an OR gate, only one fuse is left intact in order to connect the desired variable to the gate input. Once the fuse is blown, it cannot be reconnected.

Another method of programming a PLD is the antifuse, which is the opposite of the fuse. Instead of a fusible link being broken or opened to program a variable, a normally open contact is shorted by "melting" the antifuse material to form a connection.

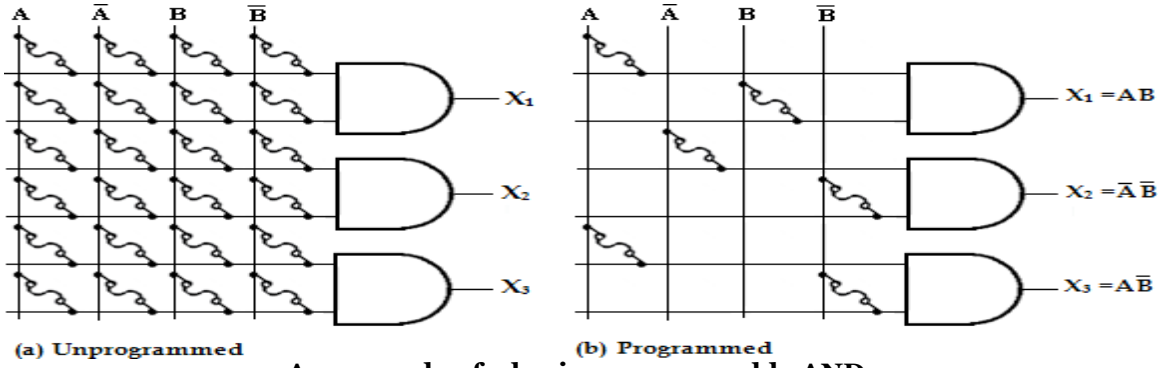


(a) Unprogrammed (b) Programmed

An example of a basic programmable OR array

The AND Array:

This type of array consists of AND gates connected to a programmable matrix with fusible links at each cross points, as shown in the figure below. Like the OR array, the AND array can be programmed by blowing fuses to eliminate selected variables from the output functions. For each input to an AND gate, only one fuse is left intact in order to connect the desired variable to the gate input. Also, like the OR array, the AND array with fusible links or with antifuses is one-time programmable.



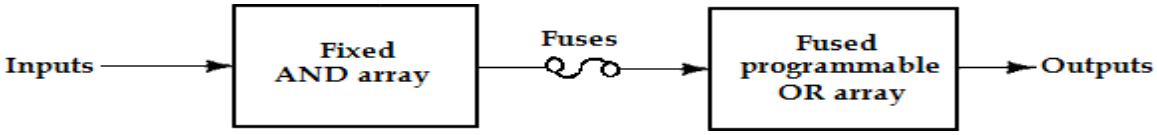
An example of a basic programmable AND array

Classification of PLDs

There are three major types of combinational PLDs and they differ in the placement of the programmable connections in the AND-OR array. The configuration of the three PLDs is shown below.

1. **Programmable Read-Only Memory (PROM):**

   A PROM consists of a set of fixed (non-programmable) AND array constructed as a decoder and a programmable OR array. The programmable OR gates implement the Boolean functions in sum of minterms.



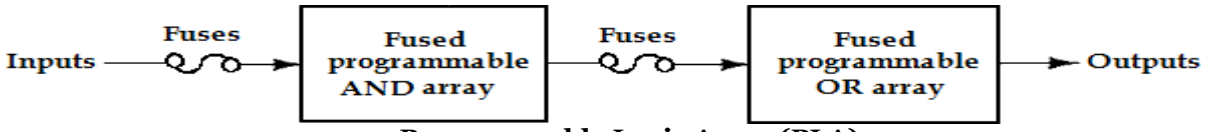
(a) Programmable read- only memory (PROM)

2. **Programmable Logic Array (PLA):**

   A PLA consists of a programmable AND array and a programmable OR array.

The product terms in the AND array may be shared by any OR gate to provide the required sum of product implementation.

The PLA is developed to overcome some of the limitations of the PROM. The PLA is also called an FPLA (Field Programmable Logic Array) because the user in the field, not the manufacturer, programs it.



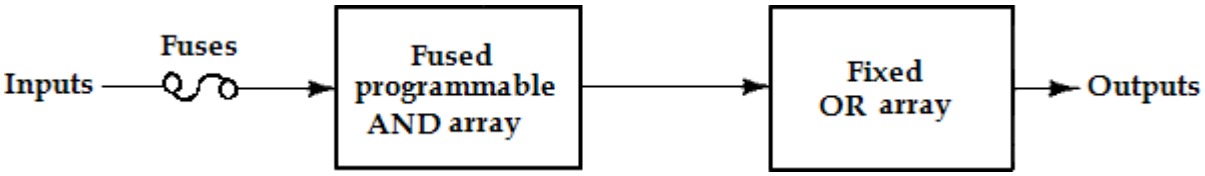
Programmable Logic Array (PLA)

3. Programmable Array Logic (PAL):

The basic PAL consists of a programmable AND array and a fixed OR array.

The AND gates are programmed to provide the product terms for the Boolean functions, which are logically summed in each OR gate.

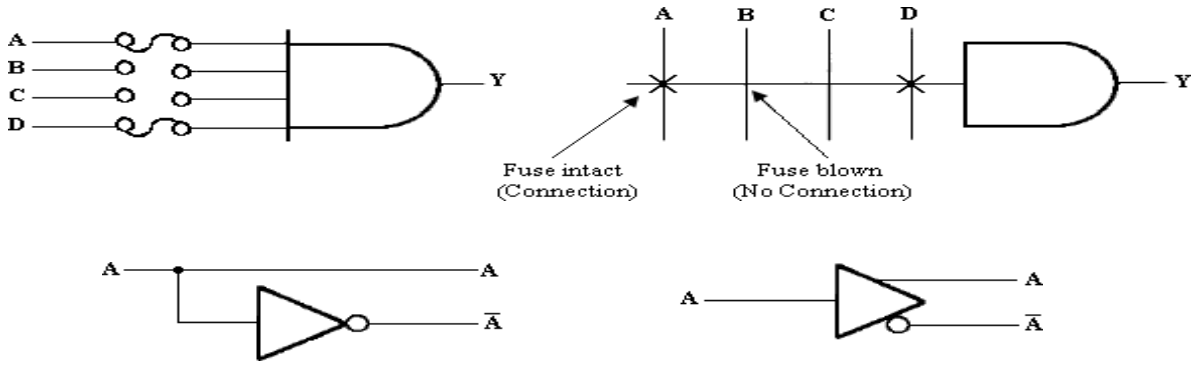
It is developed to overcome certain disadvantages of the PLA, such as longer delays due to the additional fusible links that result from using two programmable arrays and more circuit complexity.

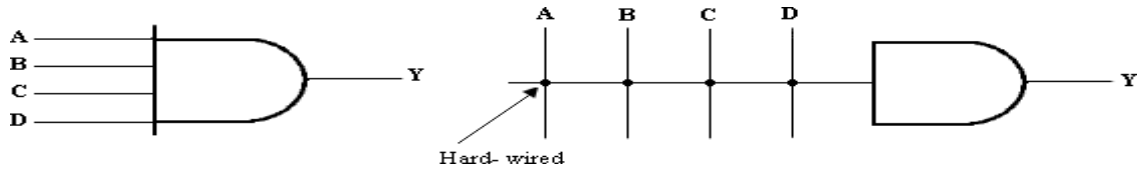


Programmable Array Logic (PAL)

Array logic Symbols:

PLDs have hundreds of gates interconnected through hundreds of electronic fuses. It is sometimes convenient to draw the internal logic of such device in a compact form referred to as **array logic**.

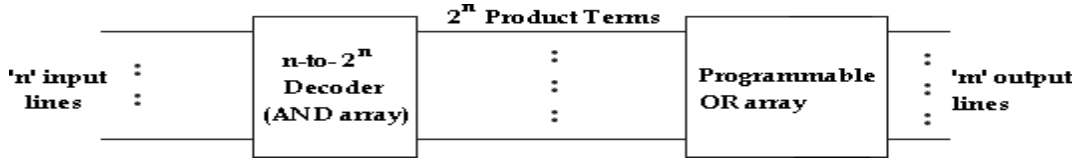




PROGRAMMABLE ROM:

PROMs are used for code conversions, generating bit patterns for characters and as look-up tables for arithmetic functions.

As a PLD, PROM consists of a fixed AND-array and a programmable OR array. The AND array is an n-to-2_n decoder and the OR array is simply a collection of programmable OR gates. The OR array is also called the memory array. The decoder serves as a minterm generator. The n-variable minterms appear on the 2_n lines at the decoder output. The 2_n outputs are connected to each of the 'm' gates in the OR array via programmable fusible links.



2_n x m PROM

4.8.4 Implementation of Combinational Logic Circuit using PROM

1. Using PROM realize the following expression

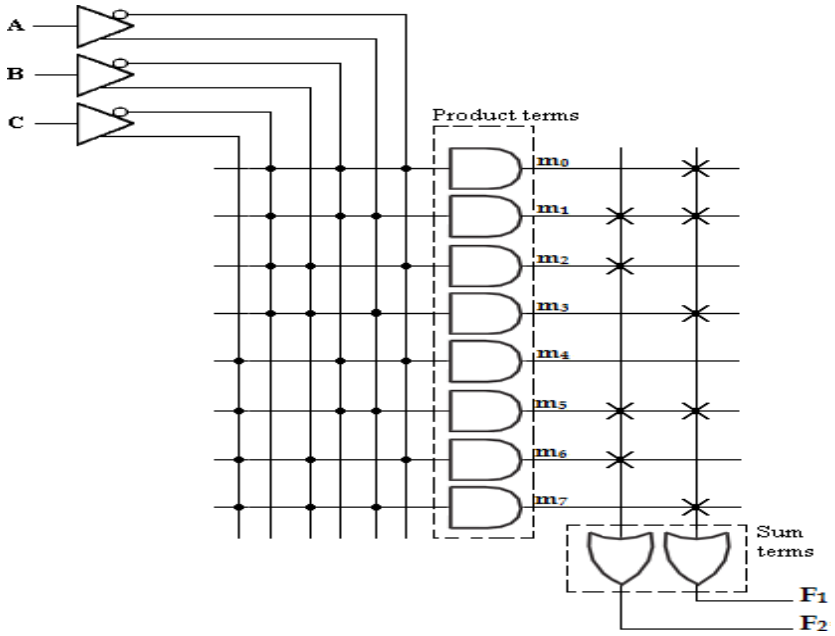
   F_1 (A, B, C) = ∑m (0, 1, 3, 5, 7)

   F_2 (A, B, C) = ∑m (1, 2, 5, 6)

Step1: Truth table for the given function

A	B	C	F_1	F_2
0	0	0	1	0
0	0	1	1	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	0	1
1	1	1	1	0

Step 2: PROM diagram



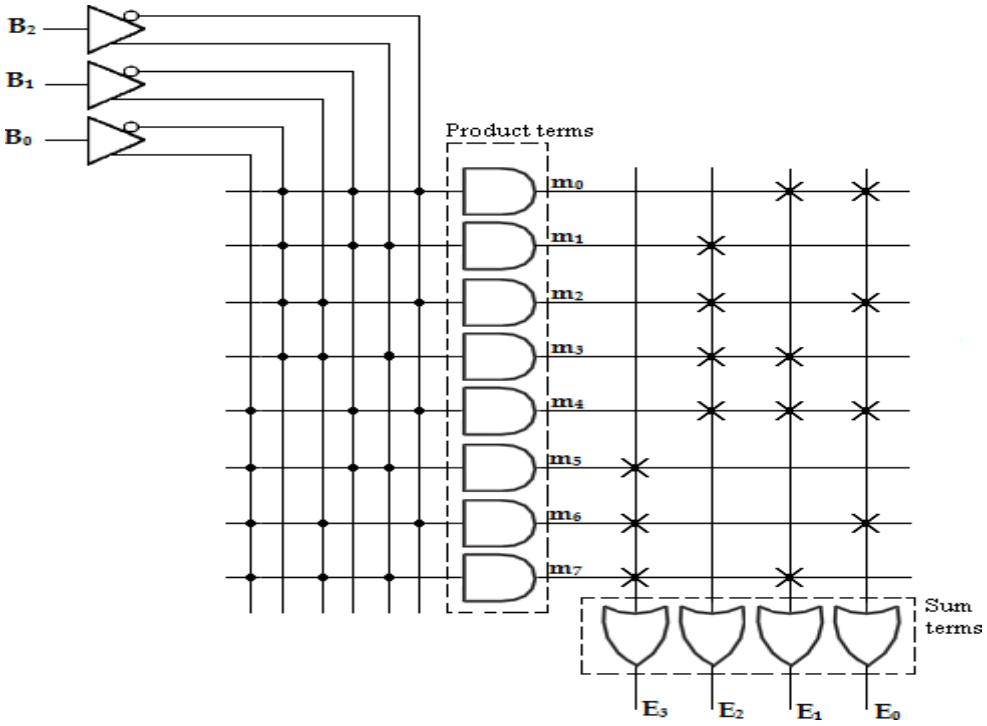
2. Design a combinational circuit using PROM. The circuit accepts 3-bitbinary and generates its equivalent Excess-3 code.

Step1: Truth table for the given function

B_2	B_1	B_0	E_3	E_2	E_1	E_0
0	0	0	0	0	1	1
0	0	1	0	1	0	0
0	1	0	0	1	0	1
0	1	1	0	1	1	0
1	0	0	0	1	1	1

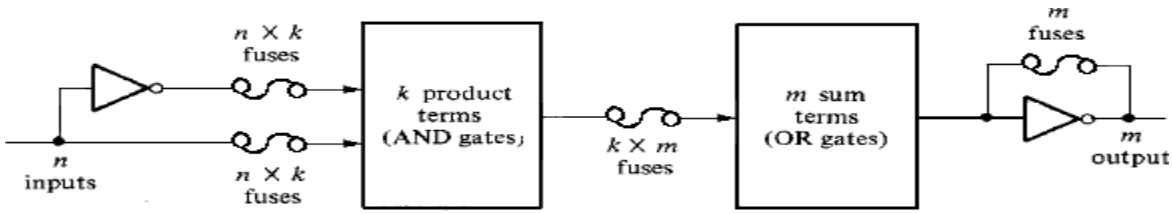
1	0	1	1	0	0	0
1	1	0	1	0	0	1
1	1	1	1	0	1	0

Step 2: PROM diagram



PROGRAMMABLE LOGIC ARRAY: (PLA): The PLA is similar to the PROM in concept except that the PLA does not provide full coding of the variables and does not generate all the minterms.

The decoder is replaced by an array of AND gates that can be programmed to generate any product term of the input variables. The product term are then connected to OR gates to provide the sum of products for the required Boolean functions. The AND gates and OR gates inside the PLA are initially fabricated with fuses among them. The specific boolean functions are implemented in sum of products form by blowing the appropriate fuses and leaving the desired connections.



PLA block diagram

The block diagram of the PLA is shown above. It consists of 'n' inputs, 'm' outputs, 'k' product terms and 'm' sum terms. The product terms constitute a group of 'k' AND gates and the sum terms constitute a group of 'm' OR gates. Fuses are inserted between all 'n'

inputs and their complement values to each of the AND gates. Fuses are also provided between the outputs of the AND gate and the inputs of the OR gates.

Another set of fuses in the output inverters allow the output function to be generated either in the AND-OR form or in the AND-OR-INVERT form. With the inverter fuse in place, the inverter is bypassed, giving an AND-OR implementation. With the fuse blown, the inverter becomes part of the circuit and the function is implemented in the AND-OR-INVERT form.

Implementation of Combinational Logic Circuit using PLA

1. **Implement the combinational circuit with a PLA having 3 inputs, 4 product terms and 2 outputs for the functions.**

   $F_1(A, B, C) = \sum m(0, 1, 2, 4)$

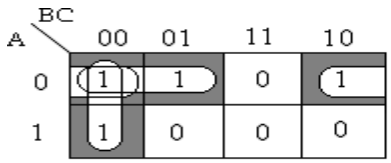
   $F_2(A, B, C) = \sum m(0, 5, 6, 7)$

Solution:

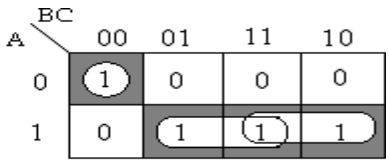
Step 1: Truth table for the given functions

A	B	C	F_1	F_2
0	0	0	1	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	0
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

Step 2: K-map Simplification

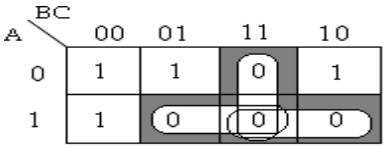


$F_1 = A'B' + A'C' + B'C'$

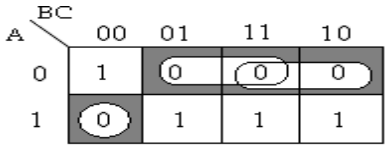


$F_2 = AC + AB + A'B'C'$

this simplification, total number of product term is 6. But we require only 4 product terms. Therefore find out F_1' and F_2'.



$F_1' = AC + BC + AB$



$F_2' = A'C + A'B + A'B'C'$

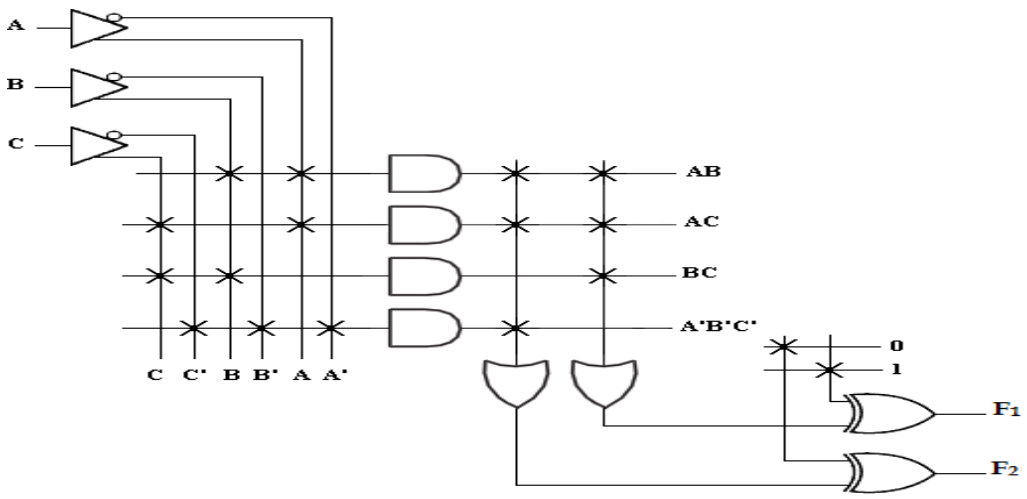
Now select, F_1' and F_2, the product terms are AC, AB, BC and A'B'C'

Step 3: PLA Program table:

	Product term	Inputs			Outputs	
		A	B	C	F_1 (C)	F_2 (T)
AB	1	1	1	-	1	1
AC	2	1	-	1	1	1
BC	3	-	1	1	1	-
A'B'C'	4	0	0	0	-	1

In the PLA program table, first column lists the product terms numerically as 1, 2, 3, and 4. The second column (Inputs) specifies the required paths between the AND gates and the inputs. For each product term, the inputs are marked with 1, 0, or - (dash). If a variable in the product form appears in its normal form, the corresponding input variable is marked with a 1. If it appears complemented, the corresponding input variable is marked with a 0. If the variable is absent in the product term, it is marked with a dash (-). The third column (output) specifies the path between the AND gates and the OR gates. The output variables are marked with 1's for all those product terms that formulate the required function.

Step 4: PLA Diagram



The PLA diagram uses the array logic symbols for complex symbols. Each input and its complement is connected to the inputs of each AND gate as indicated by the intersections between the vertical and horizontal lines. The output of the AND gate are connected to the inputs of each OR gate. The output of the OR gate goes to an EX-OR gate where the other input can be programmed to receive a signal equal to either logic 1 or 0.

The output is inverted when the EX-OR input is connected to 1 ie., **(x ⊕1= x')**. The output does not change when the EX-OR input is connected to 0 ie., **(x ⊕0= x)**.

2. **Implement the combinational circuit with a PLA having 3 inputs, 4 product terms and 2 outputs for the functions.**

   $F_1(A, B, C) = \sum m(3, 5, 6, 7)$

   $F_2(A, B, C) = \sum m(0, 2, 4, 7)$

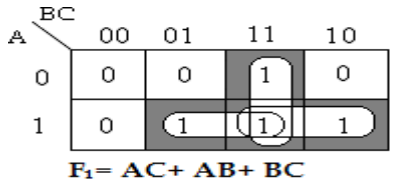
   Solution:

Step 1: Truth table for the given functions

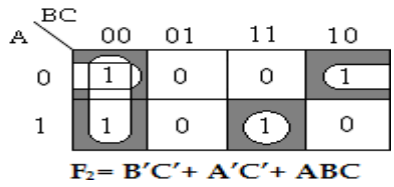
A	B	C	F_1	F_2
0	0	0	0	1
0	0	1	0	0
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0

1	1	0	1	0
1	1	1	1	1

Step 2: K-map Simplification

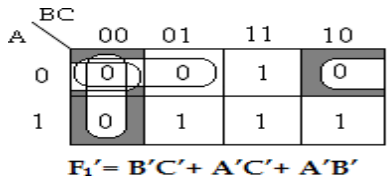


$F_1 = AC + AB + BC$

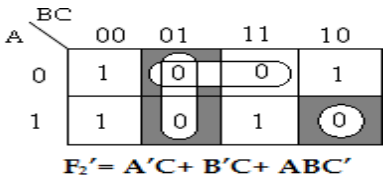


$F_2 = B'C' + A'C' + ABC$

With this simplification, total number of product term is 6. But we require only 4 product terms. Therefore find out F_1' and F_2'.



$F_1' = B'C' + A'C' + A'B'$



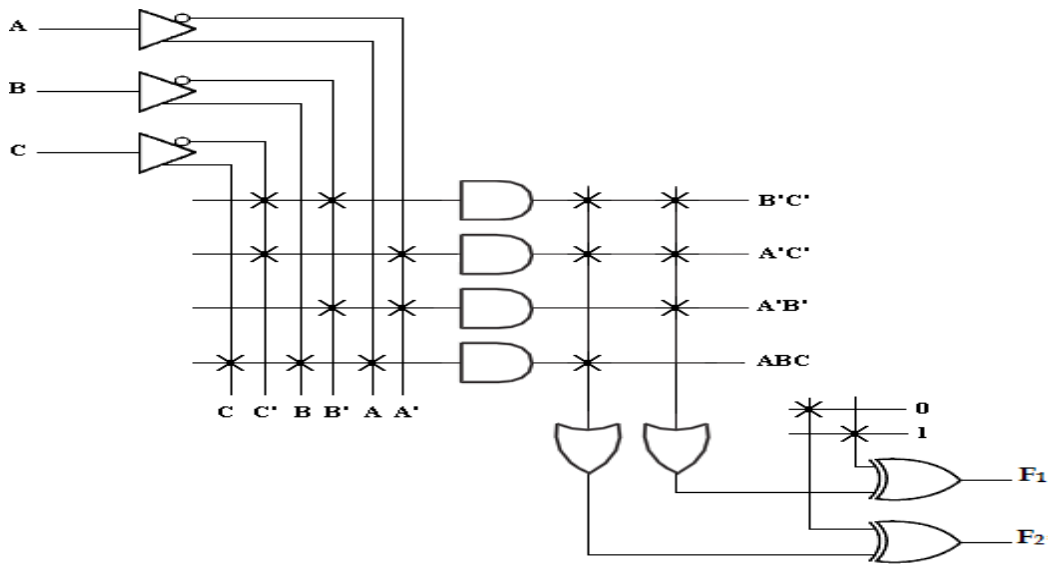
$F_2' = A'C + B'C + ABC'$

Now select, F_1' and F_2, the product terms are **B'C', A'C', A'B' and ABC**.

Step 3: PLA Program table

	Product term	Inputs			Outputs	
		A	B	C	F_1 (C)	F_2 (T)
B'C'	1	-	0	0	1	1
A'C'	2	0	-	0	1	1
A'B'	3	0	0	-	1	-
ABC	4	1	1	1	-	1

Step 4: PLA Diagram



3. Implement the following functions using PLA.

$F_1(A, B, C) = \sum m(1, 2, 4, 6)$

$F_2(A, B, C) = \sum m(0, 1, 6, 7)$

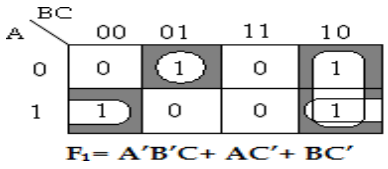
$F_3(A, B, C) = \sum m(2, 6)$

Solution:

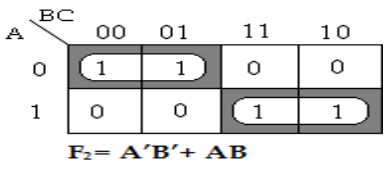
Step 1: Truth table for the given functions

A	B	C	F_1	F_2	F_3
0	0	0	0	1	0
0	0	1	1	1	0
0	1	0	1	0	1
0	1	1	0	0	0
1	0	0	1	0	0
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	1	0

Step 2: K-map Simplification



$F_1 = A'B'C + AC' + BC'$



$F_2 = A'B' + AB$

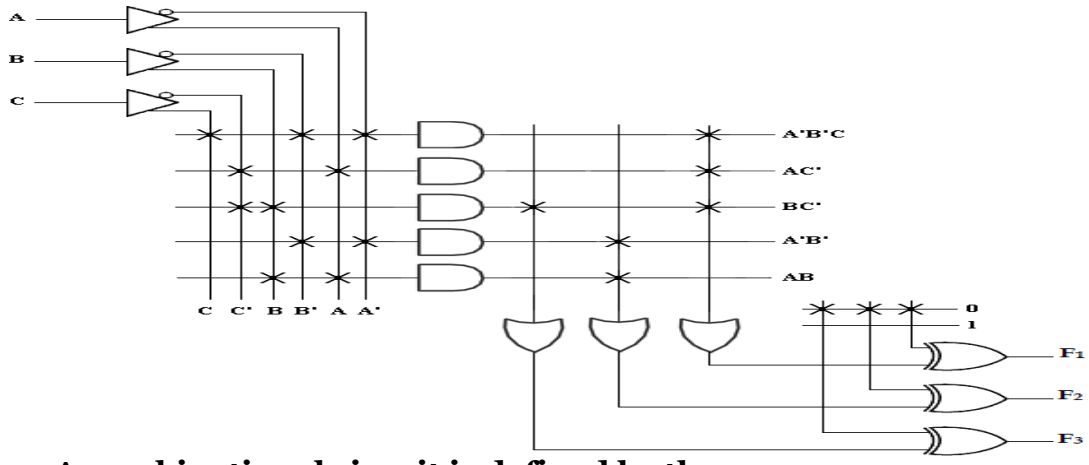
A \ BC	00	01	11	10
0	0	0	0	1
1	0	0	0	1

$F_3 = BC'$

Step 3: PLA Program table

	Product term	Inputs			Outputs		
		A	B	C	F_1 (T)	F_2 (T)	F_3 (T)
A'B'C	1	0	0	1	1	-	-
AC'	2	1	-	0	1	-	-
BC'	3	-	1	0	1	-	1
A'B'	4	0	0	-	-	1	-
AB	5	1	1	-	-	1	-

Step 4: PLA Diagram



4. A combinational circuit is defined by the functions, $F_1(A, B, C) = \sum m(0, 1, 3, 4)$

$F_2(A, B, C) = \sum m(1, 2, 3, 4, 5)$

Implement the circuit with a PLA having 3 inputs, 4 product terms and 2 outputs.

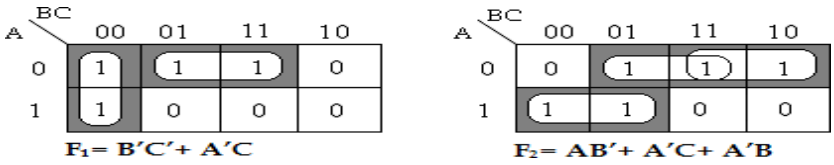
Solution:

Step 1: Truth table for the given functions

A	B	C	F_1	F_2
0	0	0	1	0
0	0	1	1	1

0	1	0	0	1
0	1	1	1	1
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	0	0

Step 2: K-map Simplification

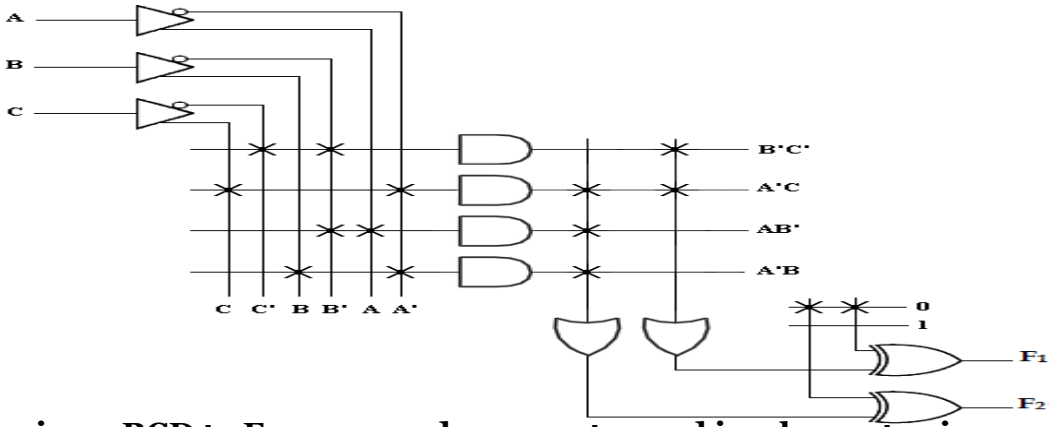


The product terms are **B'C', A'C, AB' and A'B**.

Step 3: PLA Program table

	Product term	Inputs			Outputs	
		A	B	C	F_1 (T)	F_2 (T)
B'C'	1	-	0	0	1	-
A'C	2	0	-	1	1	1
AB'	3	1	0	-	-	1
A'B						

Step 4: PLA Diagram



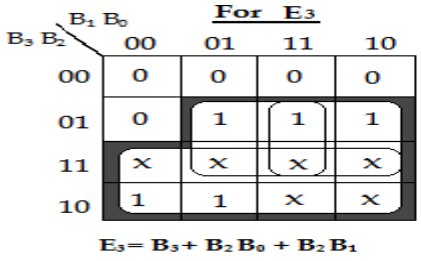
5. Design a BCD to Excess-3 code converter and implement using suitable PLA. Solution:

Step 1: Truth table of BCD to Excess-3 converter is shown below,

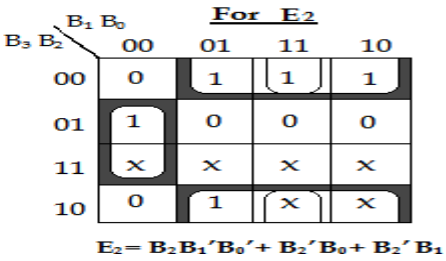
Decimal	BCD code				Excess-3 code			
	B_3	B_2	B_1	B_0	E_3	E_2	E_1	E_0

0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

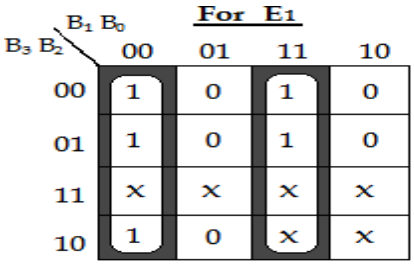
Step 2: K-map Simplification



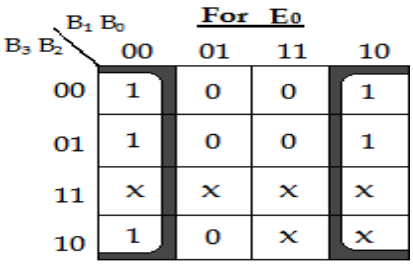
$E_3 = B_3 + B_2B_0 + B_2B_1$



$E_2 = B_2B_1'B_0' + B_2'B_0 + B_2'B_1$



$E_1 = B_1'B_0' + B_1B_0$



$E_0 = B_0'$

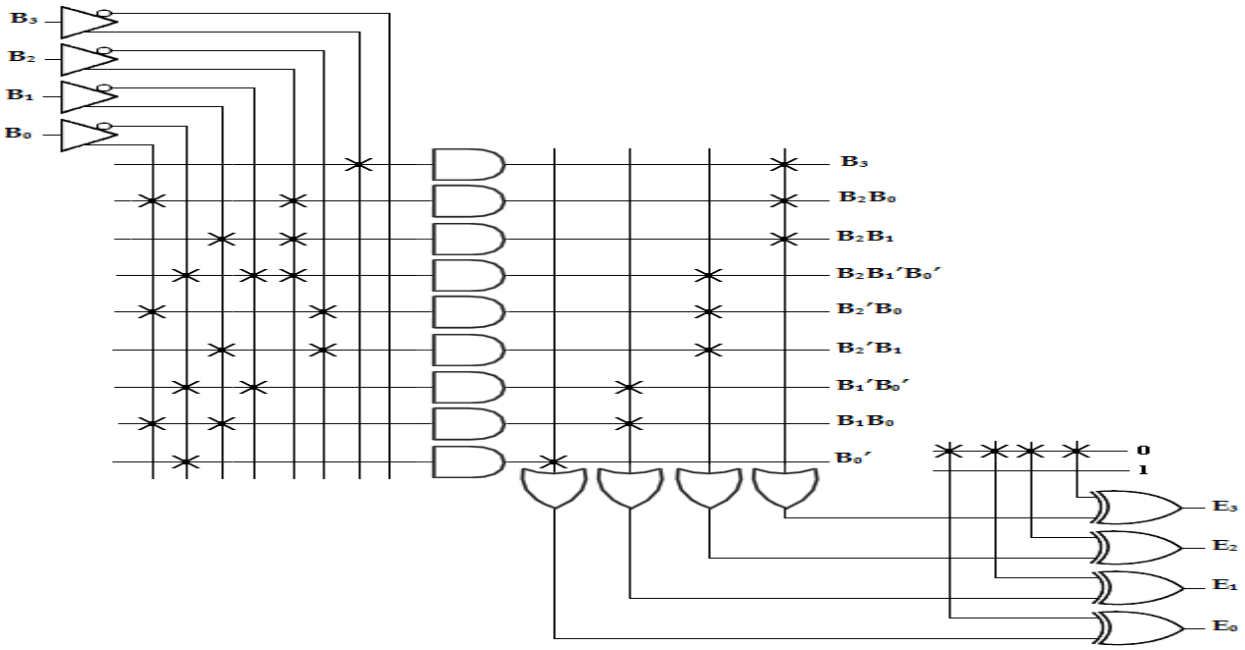
The product terms are **B_3, B_2B_0, B_2B_1, $B_2B_1'B_0'$, $B_2'B_0$, $B_2'B_1$, $B_1'B_0'$, B_1B_0, B_0'**

Step 3: PLA Program table

	Product terms	**Inputs**				**Outputs**			
		B_3	**B_2**	**B_1**	**B_0**	**E_3 (T)**	**E_2 (T)**	**E_1 (T)**	**E_0 (T)**
B_3	1	1	-	-	-	1	-	-	-
B_2B_0	2	-	1	-	1	1	-	-	-
B_2B_1	3	-	1	1	-	1	-	-	-
$B_2B_1'B_0'$	4	-	1	0	0	-	1	-	-
$B_2'B_0$	5	-	0	-	1	-	1	-	-
$B_2'B_1$	6	-	0	1	-	-	1	-	-
$B_1'B_0'$	7	-	-	0	0	-	-	1	-

B_1B_0	8	-	-	1	1	-	-	1	-
B_0'	9	-	-	-	0	-	-	-	1

Step 4: PLA Diagram

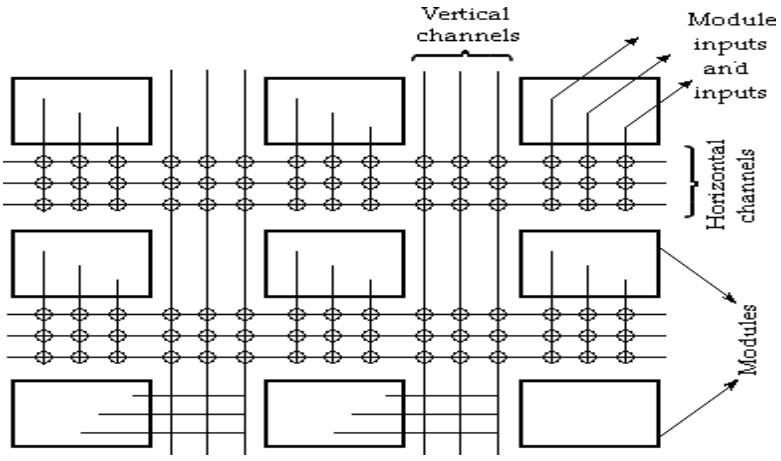


Comparison between PROM, PLA, and PAL:

S.No	PROM	PLA	PAL
1	AND array is fixed and OR array is programmable	Both AND and OR arrays are programmable	OR array is fixed and AND array is Programmable
2	Cheaper and simpler to use	Costliest and complex	Cheaper and simpler
3	All minterms are decoded	AND array can be programmed to get desired minterms	AND array can be programmed to get desired minterms
4	Only Boolean functions in standard SOP form can be implemented using PROM	Any Boolean functions in SOP form can be implemented using PLA	Any Boolean functions in SOP form can be implemented using PLA

FIELD PROGRAMMABLE GATE ARRAY: (FPGA)

Field Programmable Gate Array (FPGA) is a flexible architecture programmable logic device. The word field refers to the ability of the gate arrays to be programmed for specific function by the user. It is a Very Large Scale Integrated (VLSI) circuit constructed on a single piece of silicon. It consists of identical individually programmable rectangular modules as shown in figure below.



Architecture of FPGA

The modules are separated in both horizontal and vertical metallic conductors called *channels*. Each module has vertical and horizontal conductors at its input and output that cross one or more of the channels. Each intersection between the horizontal and vertical conductors marked as a ⊕ in the figure, is a programmable link. These programmable links are used to interconnect the modules and also to program the individual modules.

The content of the modules depends on the type of FPGA. For easy use, the modules need to be programmable into the gates and sequential elements. A module may have both combinational and sequential components.

The logic circuit design procedure using FPGA involves the following steps:

1. Capture the logic circuit to be implemented with a suitable software package, using a library of logic elements which are various configurations of basic modules available in the FPGA. In addition, many FPGA libraries also contain predesigned circuits for multiplexers, encoders, adders and so on. Predesigned circuits make design much easier.
2. ***Functional simulation***: It simulates the circuits to determine whether it is functioning properly.

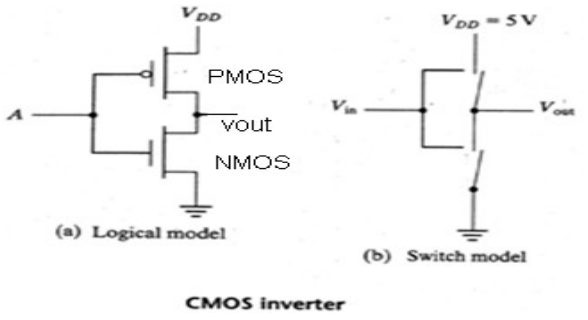
3. Configure and interconnect the modules of the FPGA to produce the desired logic circuit. This may be done automatically by routing software called **router**. Once the routing is over, it is now possible to determine the actual circuit delays which can now be introduced into the simulation model. Now, an accurate simulation of the circuit can be available.

4. ***Programming***: It is a completely automated step in which FPGA interconnections are done. The routing of the devices determined in the previous step is now made into a fuse map. Then, this fuse map is used in conjunction with a device programmer to make the internal device connections.

5. ***Testing***: After programming, it must be tested. If the designed function is not fulfilled, it must be reprogrammed. With careful simulation, reprogramming can be minimized.

TTL and CMOS logic and their characteristics:

Operation of CMOS inverter:



CMOS inverter

INPUT	OUTPUT
A	Vout =A’
0	1
1	0

CMOS - Complementary Metal Oxide Semiconductor

- CMOS inverter consists of Pull-up network (PMOS) and Pull-down network (NMOS).
- If input is 0, PMOS = ON & NMOS = OFF , the Vout is tied with VCCTherefore Vout =1

- If input is 1, PMOS = OFF & NMOS = ON, the Vout is tied with ground Therefore Vout =0

Characteristics of CMOS:

Basic gates: NAND and NOR

Power supply: 5v

Fan out: 12

Power dissipation:

0.002mW Propagation

delay: 1ns Noise immunity:

2.5V

CHARACTERISTICS OF DIGITAL IC'S

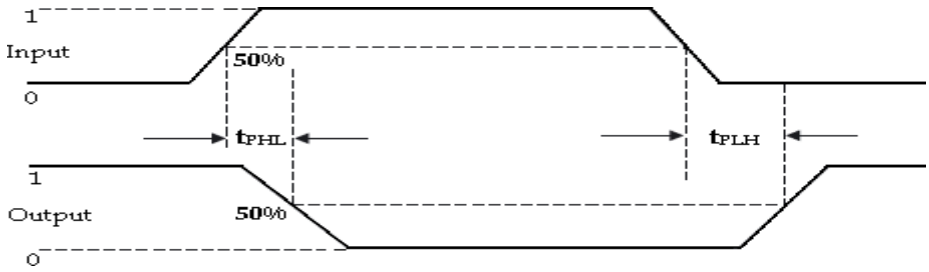
Some of the important parameters or properties of various logic families are listed as follows:

1. Speed of operation (Propagation delays)
2. Power dissipation
3. Fan- in
4. Fan- out
5. Noise Margin
6. Operating temperature
7. Power supply requirements.

1. Speed of Operation:

The speed of operation of an IC is expressed in terms of propagation delays. *Propagation delay is defined as the time taken for the output of a logic gate to change after the inputs have changed.*

It is the transition time for the signal to propagate from input to output. This factor governs the speed of operation of a logic circuit.



Propagation delays

A logic signal always experiences a delay in going through a circuit. Two types of propagation delay times are explained by Figure below, which are defined as t_{PLH}: It is the propagation delay time for a signal to change from logic LOW (0 state) to HIGH (1 state).

a) t_{PHL}: It is the propagation delay time for a signal to change from logic HIGH (1 state) to LOW (0 state).

The delay times are measured by time lapsed between the 50% voltage levels of the input and the output waveforms while making the transition. In general, the two delays t_{PLH} and t_{PHL}, are not necessarily equal and will vary depending on load conditions. The average of the two propagation delays $(t_{PLH} + t_{PHL})/2$ is called the *average delay* and this parameter is used to rate the circuit. It depends on the switching time of the individual transistors or MOSFETs in the circuit.

2. **Power dissipation**: *Power dissipation is the measure of the power consumed by logic gates when fully driven by all inputs*. The average power or the DC power dissipation is the product of DC supply voltage and the mean current consumed from that supply.

3. **Fan in**: *The maximum number of inputs that can be connected to a logic gate without any impairment of its normal operation is referred to as fan in*. For example, if the maximum of eight input loads is connected to a logic gate without any degradation of its normal operation, then its fan-in is 8. The parameter determines the functional capabilities of the logic circuit.

4. **Fan out**:

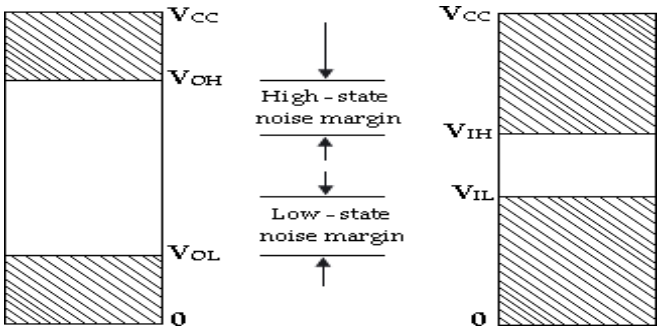
Fan out refers to the maximum number of standard loads that the output of the gate can drive without any impairment or degradation of its normal operation. A standard load is defined as the current flowing in the input of a gate in the same IC family. In a logic circuit a logic gate normally drives several other gates and the input current of each of the driven gates must be supplied from the driving gate. The driving gate must be capable of supplying this current while maintaining the required voltage level. Fan out depends on the output impedance of the driving gate and the input impedance of the driven gate.

Usually the output impedance of a logic gate is made very low, while input impedance is made very high, so that a logic gate can drive many logic gates.

5. **Noise Immunity or Noise Margin**:

The term noise denotes any unwanted signal, such as transients, glitches, hum, etc. Noise sometimes causes the change in the input voltage level, if it is too high, and leads to unreliable operation.

Noise immunity or the noise margin is the limit of noise voltage that may appear at the input of the logic gate without any impairment of its proper logic operation.



(a) Output Voltage range (b) Input Voltage range

Signals for Evaluating Noise Margin

The difference between the operating input logic voltage level and the threshold voltage is the noise margin of the circuit. The manufacture usually quotes the noise margin, which refers to the amplitude of the noise voltage that may cause the logic level to change. In the worst case, a TTL gate functions properly as long as the noise margin is kept less than 0.4V.

6. **Operating Temperature**:

All the ICs are semiconductor devices and they are temperature sensitive by nature. The operating temperature ranges of an IC vary from 0°C to + 70°C for commercial and industrial application, and from –55°C to 125°C for military application.

7. Power Supply Requirements:

The amount of power and supply voltage required for an IC is one of the important parameters for its normal operation. They are different for different logic families. The logic designer should consider these parameters while choosing the proper power supply.

TRANSISTOR – TRANSISTOR LOGIC (TTL)

The most commonly used saturating logic family called the Transistor-Transistor Logic (TTL), has the fast switching speed when compared to other logic families that utilize saturated transistors. The series 54/ 74 TTL family has grown

and evolved into five major families:

i. Standard TTL (74/54 series)
ii. High-speed TTL (74H/54H series)
iii. Low-power TTL (74L/54L series)
iv. Schottky diode clamped TTL (74S/54S series)
v. Low-power schottky TTL (74LS/54LS series)

Although the high-speed and low-power TTL devices are designed for specific applications, all the groups of the family have several common features, and are compatible and capable of interfacing directly with one another.

They have the following typical characteristics in common.

i. Supply voltage is 5V
ii. Logic 0 output voltage level is 0 V to 0.4 V
iii. Logic 1 output voltage level is 2.4 V to 5 V
iv. Logic 0 input voltage level is 0 V to 0.8 V
v. Logic 1 input voltage level is 2 V to 5 V
vi. Noise immunity is 0.4 V

The five different TTL series as mentioned above differ from one another in terms of propagation delay and power dissipation values. The table comprising of the typical values of propagation delay, power consumption, speed-power product, maximum operating frequency, and fan out for different TTL series is given below.

TTL Series Name	Prefix	Fan out	Power Dissipation(mW)	Propagation Delay (ns)	Speed-power Product (pJ)
Standard	74	10	10	9	90
Low-power	74L	20	1	33	33
High-speed	74H	10	22	6	132
Schottky	74S	10	19	3	57
Low power schottky	74LS	20	2	9.5	19

The standard TTL gate was the first version of TTL family. The basic gate was constructed with different resistor values to produce gates with lower power dissipation or higher speed. The propagation delay of saturated logic family largely depends upon two factors—storage time and RC time constants. Reducing the storage time decreases the propagation delay and also reducing the resistor values in the circuit reduces the RC time constant, which in turn reduces the propagation

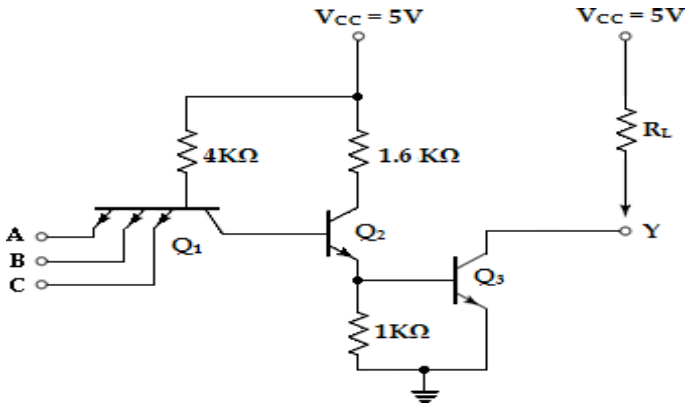
delay. But reduction in resistor values causes higher power consumption. The speed of the gate is inversely proportional to the propagation delay.

All TTL series are available in SSI components and in more complex forms, such as MSI and LSI components. The differences in the TTL series are not in the digital logic that they perform, but rather in the internal construction of the basic NAND gate. In any case, TTL gates in all the available series come in three different types of output configuration:

1. Open- collector output
2. Totem- pole output
3. Three- state output

Standard TTL with Open- Collector Output Configuration

The basic TTL NAND gate is shown in Figure, which is the modified circuit of a DTL gate. Q_1 is a multiple Emitters transistor and the logic inputs are applied to the emitters of Q_1. These emitters behave like the input diodes in the DTL gate, as they form pn junction with their common base. The base-collector junction of Q_1 acts like another pn junction diode, equivalent to the diode D_1 of the DTL gate. The transistor Q_2 replaces the second diode D_2 of the DTL gate. The output of the TTL gate is taken from the open collector of Q_3. A resistor must be connected externally at the collector of Q_3 to VCC, to maintain the output voltage level to high when Q_3 is at cut-off. The external resistor is termed as a *pull-up* resistor.



Open- Collector TTL Gate

The two voltage levels of the TTL circuit are 0.2V for the low level and 2.4 V - 5 V for the high level. If any input is low, the corresponding base emitter junction of

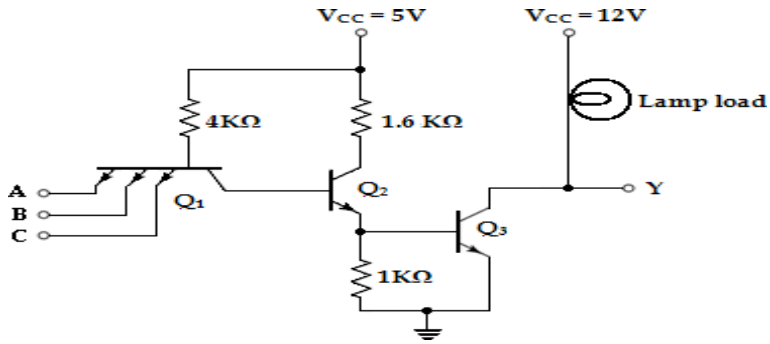
Q_1 becomes forward biased. The voltage level at the base of the transistor Q_1 is 0.2 V plus one V_{BE} drop of 0.7 V, *i.e.*, 0.9 V. This voltage level is not sufficient to drive the transistor Q_2 and Q_3, and they are at cut-off condition. The voltage required at the base of Q_1, to drive Q_2 and Q_3 into saturation should be V_{BE} of Q_3 plus V_{BE} of Q_2 plus one pn junction diode drop of Q_1, *i.e.*, 0.7 V + 0.7 V + 0.7 V = 2.1V. When the output transistor Q_3 cannot conduct, the output voltage level at Y will be high if any external resistor R_L is connected to V_{CC}.

When all the inputs are high, no base emitter junction of Q_1 is forward biased and voltage at the base of Q_1 is higher than 2.1 V. Hence transistor Q_2 is driven to saturation as well as Q_3 provided it has the current through the collector. The collector current may be available from the external pull-up resistance or from the connected loads at the output. The output voltage at Y is V_{CE} (saturation) *i.e.*, 0.2 V. Thus the gate operation conforms the NAND function, as when any of the inputs is low, output Y is high and if all the inputs are high, output Y is low.

You may notice that the TTL gate with an open collector output configuration can be operated without using any external resistor when connected to the inputs of other TTL gates. However, this is not recommended because noise immunity becomes low. Without an external resistor, the output of the gate will be an open circuit when Q_3 is at cut-off. An open circuit to an input of a TTL gate behaves like a high-level input, but a very small amount of noise may change this to a low level. When Q_3 conducts, its collector current will be available from the input of the loading gate through V_{CC}, the 4K resistor, and the forward-biased base-emitter junction.

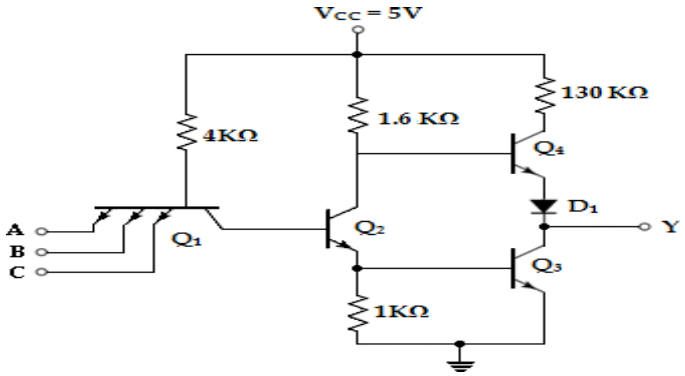
The open collector output configuration has many useful applications. The output may be interfaced with another circuit that has a different supply voltage. The external resistor may be selected of a suitable value according to the supply voltage it is connected to. This facilitates to drive a lamp or a relay which may have a supply voltage other than 5 V used for TTL, directly from the open collector gate as shown in Figure below. When the output transistor is off, no current flows through the lamp or relay and it remains off. When the output transistor Q_3 is on, current path is available for the lamp or relay to make it on. Also, the open collector output

gates can be used for interfacing with gates of another family like CMOS, where supply voltage varies from 3 V to 15 V.



STANDARD TTL WITH TOTEM POLE OUTPUT CONFIGURATION:

The TTL NAND gate with *totem pole* output configuration is shown in Figure below. It is the same circuit as the open-collector gate, except for the output transistor Q_4, a diode D_1, and resistor 130Ω at the collector of Q_4. It is called the *totem pole* output configuration, because the transistor Q_4 sits upon Q_3. The base of the transistor Q_4 is driven from the collector of Q_2, as shown in Figure.



TTL Gate with Totem-Pole Output

The two voltage levels of the TTL circuit are 0.2 V for the low level and 2.4 V – 5V for the high level. If any input is low, the corresponding base emitter junction of Q_1 becomes forward biased. The voltage level at the base of the transistor Q_1 is 0.2 V plus one V_{BE} drop of 0.7 V, *i.e.*, 0.9 V. This voltage level is not sufficient to drive the transistor Q_2 and Q_3, and they are cut-off condition. The voltage required at the base of Q_1, to drive Q_2 and Q_3 into saturation should be V_{BE} of Q_3 plus V_{BE} of Q_2 plus one PN junction diode drop of Q_1, *i.e.*, 0.7 V + 0.7 V + 0.7 V = 2.1V. When Q_2 and Q_3 are off, high base current available for Q_4 to operate and the output Y is logic high. The

currents for the output loads or the gates connected at the output are supplied through transistor Q_4 and its collector resistor 130Ω.

When all the inputs are high, no base-emitter junction of Q_1 is forward biased and voltage at the base of Q_1 is higher than 2.1 V. Hence, transistors Q_2 and Q_3 are driven to saturation. The output voltage at Y is V_{CE} (saturation) *i.e.*, 0.2 V. The voltage at the collector of Q_2 is equal to one V_{BE} drop of Q_3 plus one V_{CE} (saturation) drop of Q_2, *i.e.*, 0.7 V + 0.2 V = 0.9 V. This voltage level is applied to the base of Q_4 and is not sufficient to drive the transistor Q_4. Since, to drive the transistor Q_4, the voltage required at its base is one V_{CE} (saturation) for Q_3 plus one diode drop against D_1 plus one V_{BE} drop of Q_4, *i.e.*, 0.2 V + 0.7 V + 0.7 V = 1.6 V. Hence Q_4 is at cut-off condition.

While Q_3 is in saturation, its collector current is available from the connected loads at the output. Thus the gate operation confirms the NAND function, as when any of the inputs is low, output Y is high and if all the inputs are high, output Y is low.

The output impedance of a gate is normally a resistive plus a capacitive load. The capacitive load consists of the capacitance of the output transistor, the capacitance of any of the fan out gates and any stray capacitance. When the output changes from low state to high state due to a change in the input condition, the output transistor goes from saturation to cut-off state. As soon as the transistor Q_2 turns off and Q_4 conducts because its base is connected to V_{CC} and the total load capacitance, C charges exponentially from the low- to high-voltage level with the time constant of RC, where R is the resistance value at the collector of the output transistor Q_3 to supply voltage V_{CC}, which is also referred to as a pull-up resistor. For totem pole configuration as above, a network consisting of a transistor Q_4, diode D_1, and resistor 130Ω, is connected at the collector of the output transistor Q_3 and is referred to as an *active pull-up* circuit. The active pull-up circuit offers low output impedance and hence the rise time of output is faster and in turn, the propagation delay is reduced to the order of 10 ns. Another advantage of the active pull-up circuit is that because of its low output impedance, it is capable of supplying more currents to the driven gate, and so the fan out capability increases.

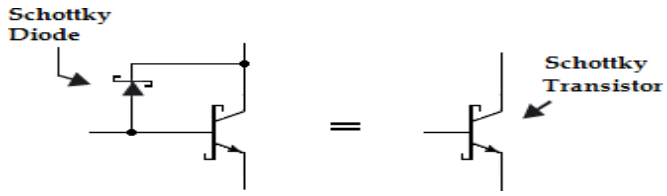
As the capacitive load charges, the output voltage raises and the current in Q_4 decreases, bringing the transistor into the active region. Thus, in contrast to the other transistors, Q_4 is in the active region when a steady state condition is reached. The final value of the output voltage is 5V minus one V_{BE} drop of Q_4, minus one diode drop of D_1, *i.e.*, 5 V – 0.7 V – 0.7 V = 3.6 V. The transistor Q_3 goes into cut-off very fast, but during the initial transition time both Q_3 and Q_4 are on and a peak current is drawn from the power supply. This current spike generates noise in the power supply distribution system. If the change of state is frequent, the transient current spikes increase the power supply current requirement and the average power dissipation of the circuit increases.

A wired logic connection like open-collector gates is not allowed with totem pole output configuration. When two totem poles are wired together with the output of one gate high and the output of other gate is low, an excessive amount of current will be drawn to produce heat and this may cause damage to the transistors in the circuit. Some TTL gates are constructed to withstand the amount of current that is produced under this condition. In any case, the collector current in the low gate may be high enough to move the transistor into the active region and produce an output voltage in the wired connection greater than 0.8 V, which is not a valid binary signal for TTL gates.

SCHOTTKY TTL GATE

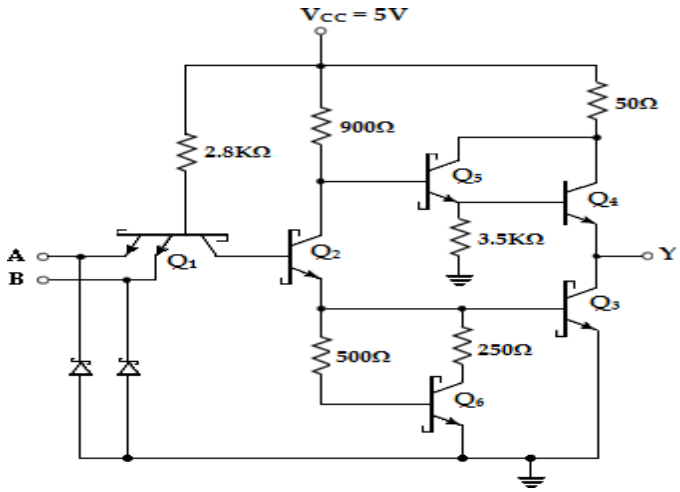
A reduction in storage time results in the reduction of propagation delay. This is due to the time needed for a transistor to come out of its saturation condition delays the switching of the transistor from the on saturation condition to cut-off condition.

Saturation condition can be eliminated by placing a *Schottky diode* between the base and the collector of each saturated transistor in the circuit as shown in Figure below.



The Schottky diode is formed by the junction of a metal and a semiconductor, in contrast to the conventional *pn* diode which is formed by the junction of *p*-type and *n*-type semiconductors. The forward-biased voltage across the Schottky diode is

V as compared to 0.7 V in a conventional diode. The presence of a Schottky diode between the base and collector prevents the transistor from driving into saturation. A transistor with a Schottky diode is referred to as a *Schottky transistor*. The use of Schottky transistors in TTL circuits results in the reduction in propagation delay without sacrificing the power dissipation.



Schottky TTL Gate

A two-input Schottky TTL NAND gate circuit is shown in Figure above. With comparison to the standard TTL gate, all the transistors of the Schottky TTL circuit are of Schottky type, except Q4. Exception is made because the transistor Q_4 does not go to the saturation region but remains at active region. It should be noted that the resistor values have been reduced to further decrease in the propagation delay.

In addition to employing the Schottky transistors and reducing the circuit resistor values, the Schottky TTL circuit includes other modifications over the standard TTL circuit. Two new transistors, Q_5 and Q_6, have been introduced and Schottky diodes are provided at each of the input terminals to ground. The transistors Q_5 and Q_4 are in darlington mode taking care of the diode V_{BE} drops, higher current gain and current output capability, low output impedance and reduction in propagation delay.

The diodes at each input terminal as shown in the circuit are provided to prevent any ringing that may occur in the input lines. Under transient switching conditions, the signal lines appear inductive; this, along with the stray capacitance of the circuit, cause signals to oscillate or ring. When the output of a gate changes its level from high to low, the ringing waveform at the input of the connecting gate may have the excursions below ground as high as 2 to 3 V depending on the line length. The diodes connected to the ground help to clamp the ringing as they conduct when the negative voltage exceeds 0.4 V. When the negative excursion is limited, the positive swing also becomes limited and thus reduces the ringing as well as unwanted switching of the gate.

The emitter circuit of Q_2 in Figure has been modified in Figure by a circuit consisting of a transistor Q_6 and two resistors. The turn-off current spikes are reduced due to this modification, which also helps to reduce the propagation time of the gate.

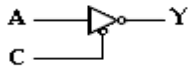
TTL GATE WITH TRI- STATE OUTPUT CONFIGURATION:

All the logic gates have two output states—logic 0 and logic 1. But the tri-state or three-state gate, as its name implies, has three output states as follows.

1. A low-level state or logic 0 state, when the lower transistor in the totem pole is on and the upper transistor is off.
2. A high-level state or logic 1 state, when the lower transistor in the totem pole is off and the upper transistor is on.
3. A third state when both transistors in the totem pole are off. This provides an open circuit or high impedance state which allows the direct wired connection of many outputs on a common line. Three states eliminate the need of open collector gates in common bus configurations.

(a) Three- state buffer state

Input	Enable	Output
1	0	1
0	0	0
x	1	Tristate

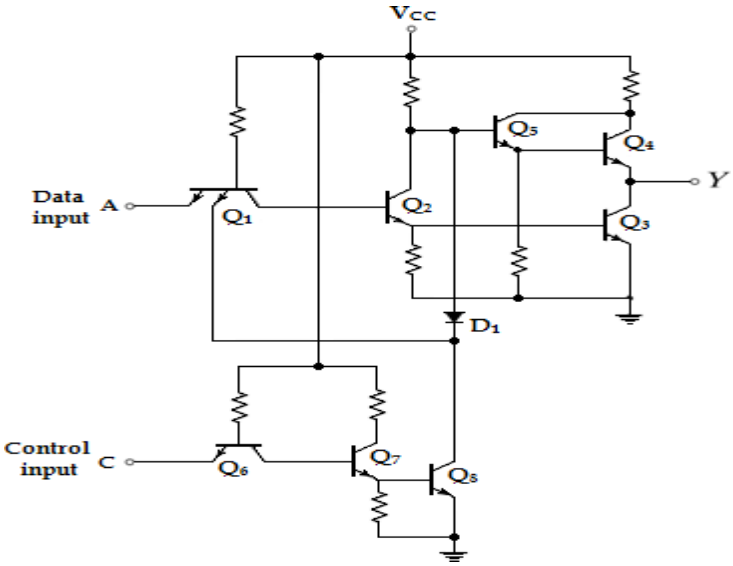


(b) Three- state inverter gate

Input	Enable	Output
1	0	0
0	0	1
x	1	Tristate

Tri-state gates consist of an extra input called a control or enable input. Here, the control input is such that when it is logic 0, the gate performs its normal operation.

When the control input is logic 1, the output of the gate goes to tri-state or high impedance state regardless of the value of input A.



Three- State TTL gate

The circuit diagram of the three-state inverter is shown above. Transistors Q_6, Q_7, and Q_8 associated with the control input form a circuit similar to the open-collector gate. Transistors Q_1- Q_5, associated with the data input, form a totem-pole TTL circuit. The two circuits are connected together through diode D_1. *As* in an open- collector circuit, transistor Q_8 turns off when the control input at *C* is in the low-level state. This prevents diode D1 from conducting. In addition, the emitter in

Q_1 connected to Q_8 has no conduction path. Under this conduction, transistor Q_8 has no effect on the gate and the output in Y depends only on the data input at A.

When the control input is high, transistor Q_8 turns on, and the current flowing from V_{CC} through diode D_1 cause transistor Q_8 to saturate. The voltage at the base of Q_5 is now equal to the voltage across the saturated transistor, Q_8, plus one diode drop, or 0.9V. This voltage turns off Q_5 and Q_4 since it is less than two V_{BE} drops. At the same time, the low input to one of the emitters of Q_1 forces transistors Q_3 (and Q_2) to turn off. Thus, both Q_3 and Q_4 in the totem-pole are turned off and the output of the circuit behaves like an open circuit with very high output impedance.

A three-state bus can be created by wiring several three-state outputs together. At any given time, control input of only one single gate is enabled to transmit its information on the common output bus, and control inputs for all other gates must be disabled at that time instant to keep their outputs at a high impedance state. Extreme care must be taken to select the control input. If two or more control inputs are enabled at the same time, the undesirable condition of two or more active totem pole outputs will arise.

An important feature of most tri-state gates is that the output enable delay is longer than the output disable delay. If a control circuit enables one gate and disables another gate at the same time, then the disable gate enters the high impedance state before the enable gate comes to its action. This eliminates the undesirable situation of both gates being active at the same time.

There is a very small leakage current associated with the high impedance state condition in a tri-state gate. However, this current is so small that as many as 100 tri-state gates can be connected together to form a common bus line without degrading logic behavior of the gates.