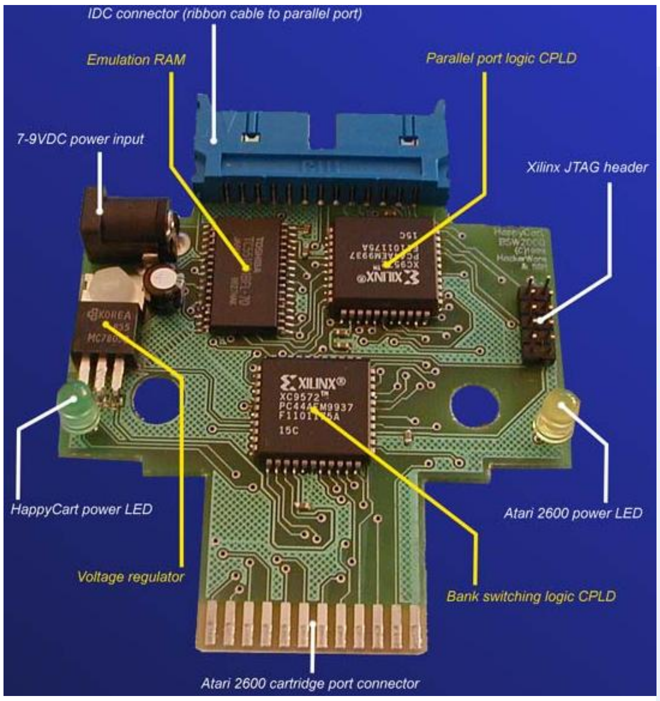
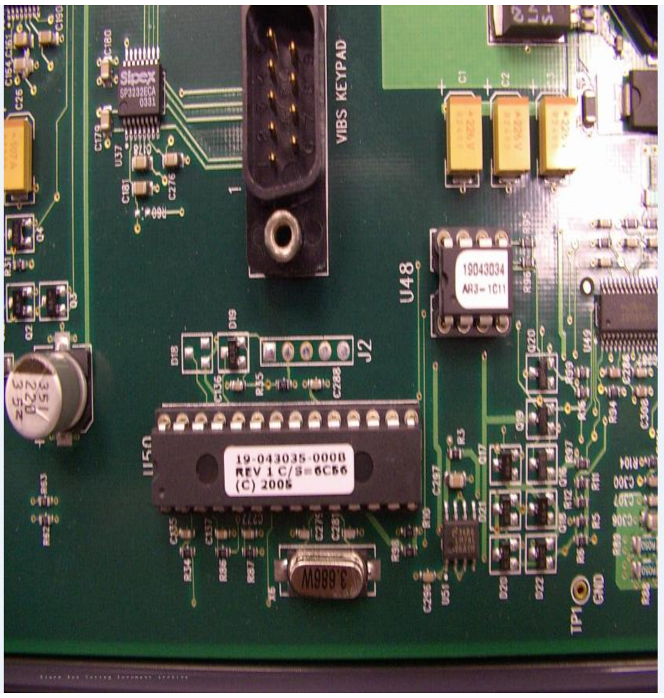
Memory & Programmable Logic Devices





Prepared By: Aso Abdul Adhim Hamah Amin

Index:

OBJECTIVES:-

Note: We focus on PLD

A programmable logic device or PLD is an electronic component used to build reconfigurable digital circuits. Unlike a logic gate, which has a fixed function, a PLD has an undefined function at the time of manufacture. Before the PLD can be used in a circuit it must be programmed, that is, reconfigured.

Form this topic we:

- Discuss types, characteristics and operations of ROM
- To know the basics of Programmable Logic devices and it's types in general and Read Only Memory (ROM) in specific.
- classify and design various ROMs as a kind of PLD and To understand the IC fabrication process

.

Introduction of Memories

- Memory devices are used in electronics system as storage devices
- Computer and others types of systems require the permanent and semi-permanent storage of large amount of binary data
- Microprocessor-based systems rely on storage devices and memories for storing program and for retaining data during processing
- In computer terminology, memory usually refers to RAM and ROM while storage refers to hard disk, floppy disk and CD-ROM
- Programmable logic devices such as complex programmable logic devices (CPLD) and field programmable grid array FPGA) are custom-configured capable to create desired digital circuit

Basic of semiconductor Memory

1. Portion of a system for storing large amount of binary data.
2. Consist of array of latches or capacitors.
3. As a rule memories store data from 1-8 binary data.
4. Basic unit of binary data I computer terminology
   - Bit -1-bit (binary digit)
   - Nibble -4-bit
   - Bytes-8-bit
   - Word-a group of bit or bytes that acts as single entry that can be stored in one memory location
5. In assembly language, a word is specifically defined as two byes.
6. A single binary bit is stored in memory cell.
7. An organized group of cells is called an array.

Memory Device:

Device to which binary information is transferred for storage, and from which information is available for processing as needed.

Memory Unit:

Is a collection of cells capable of storing a large quantity of binary Information?

In digital systems, there are two types of memories:

1. RAM
2. ROM

1. Random-Access Memory (RAM)
RAM is the place in a computer where the operating system, application programs, and data in current use are kept so that they can be quickly reached by the computer's processor.

2. Read-Only Memory (ROM):
ROM is a type of memory that is as fast as RAM, but has two important differences: It cannot be changed, and it retains its contents even when the computer is shut off. It is generally used to start your computer up and load the operating system.

Using a ROM as a PLD: A programmable logic device or PLD is an electronic component used to build digital circuits. Before the PLD can be used in a circuit it must be programmed.

Examples of PLDs : programmable logic array (PLA), programmable array logic (PAL), and field-programmable logic gate array (FPGA). (PAL: Program. AND, fixed OR, PLA: Program. AND/OR)

Basic of Semiconductor Memory

Introduction to Programmable Logic:

Programmable logic refers to a general class of devices which can be configured to perform a variety of logic functions. The devices range from simple PROM, programmable read-only memory, devices which can implement simple combinatorial logic, to PAL's, programmable array logic, to FPGA's, field programmable gate arrays. All these devices share the feature that they are programmed to perform specific functions.
PROMS are usually thought of in the context of non-volatile storage for microprocessor programs, but they can be used for simple combinatorial functions. A PROM has a number of memory locations accessed by address lines. The contents of the memory location are output on data lines. Imagine that we want to implement a 4 bit adder with a PROM. We have 8 address lines. Four of the address lines are assigned to one term, and the other 4 to the other term. We program each memory location of the PROM with the correct sum for each combination of input terms. For example, location 0000 0001 contains 1, location 0010 0010 contains 4, get the idea? This is known as a Look-Up Table, or LUT. The memory resembles a table, and the address lines are used to "look-up" a particular entry.
Programmable logic as we know it today started with devices known as Programmable Array Logic. These devices get their name from the programmable AND array which is part of the device. These devices have pins which can be programmed to be logic inputs. Each pin will be one

logic variable. Each output can be programmed to be active for a particular sum of products of the input terms. Remember from our discussion of combinatorial logic that any logic function can be implemented as a sum-of-products expression. For example, if my PAL has 3 inputs and two outputs, I could program the familiar one-bit full-adder function. One output would be programmed for the logic equation of the sum, and the other of the carry out. These early devices were suitable for combinatorial designs.
This basic idea was extended to devices that were capable of registered and sequential designs by the addition of macrocells after the programmable array. The macrocell contains a flip-flop, the essential ingredient for state machines. Devices with this architecture form the branch of the programmable logic family known as PLD's and CPLDS's (Programmable Logic Devices & Complex Programmable Logic Devices).

What is Programmable Logic?

In the world of digital electronic systems, there are three basic kinds of devices: memory, microprocessors, and logic. Memory devices store random information such as the contents of a spreadsheet or database. Microprocessors execute software instructions to perform a wide variety of tasks such as running a word processing program or video game. Logic devices provide specific functions, including device-to-device interfacing, data communication, signal processing, data display, timing and control operations, and almost every other function a system must perform.

The PLD Advantage:

Fixed logic devices and PLDs both have their advantages. Fixed logic devices, for example, are often more appropriate for large volume applications because they can be mass-produced more economically. For certain applications where the very highest performance is required, fixed logic devices may also be the best choice.

However, programmable logic devices offer a number of important advantages over fixed logic devices, including:

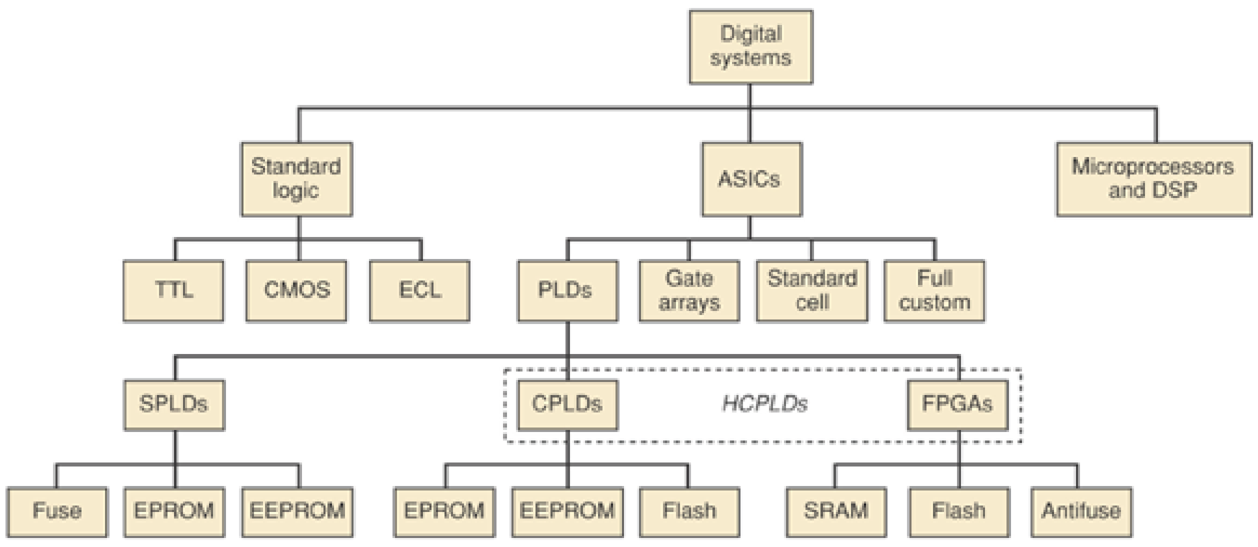
- PLDs offer customers much more flexibility during the design cycle because design iterations are simply a matter of changing the programming file, and the results of design changes can be seen immediately in working parts.
- PLDs do not require long lead times for prototypes or production parts - the PLDs are already on a distributor's shelf and ready for shipment.
- PLDs do not require customers to pay for large NRE costs and purchase expensive mask sets - PLD suppliers incur those costs when they design their programmable devices and are able to amortize those costs over the multi-year lifespan of a given line of PLDs.
- PLDs allow customers to order just the number of parts they need, when they need them, allowing them to control inventory. Customers who use fixed logic devices often end up with excess inventory which must be scrapped, or if demand for their product surges, they may be caught short of parts and face production delays.
- PLDs can be reprogrammed even after a piece of equipment is shipped to a customer. In fact, thanks to programmable logic devices, a number of equipment manufacturers now tout the ability to add new features or upgrade products that already are in the field. To do this, they simply upload a new programming file to the PLD, via the Internet, creating new hardware logic in the system.

There are a wide variety of ICs that can have their logic function "programmed" into them after they are manufactured. Most of these devices use technology that also allows the function to be reprogrammed, which means that if you find a bug in your design, you may be able to fix it without physically replacing or rewiring the device.

A class of devices called *programmable logic devices* (or PLDs) can be thought of as universal logic implementers in the sense that they can be configured (actually programmed) by the user to perform a variety of specific logic functions. So useful and versatile are these PLDs that one might question why any other means of design would ever be considered. Well, the answer is, of course, that there is a time and place for a variety of approaches to design — that is, no one single approach to design satisfies all possible problem situations. However, the option to use PLDs offers the logic designer a wide range of versatile devices that are commercially available for design purposes. Some PLDs are made to perform only combinational logic functions; others can perform both combinational and sequential logic functions. Four commonly used PLDs considered here are the *read-only memory* (ROM) devices and their subgroups, the *programmable logic array* (PLA) devices, the *programmable array logic* (PAL) devices and their subgroups, *and programmable gate arrays* (PGAs) and subgroups. Other PLDs include *erasable programmable logic devices* (EPLDs), including *erasable programmable ROMs, generic array logic* (GAL) devices, and *programmable logic sequencers* (PLSs). Except for PGAs, most of the PLDs mentioned have some commonality, namely a two-level AND/OR configuration. What is connected to the AND/OR network distinguishes one PLD from another.

Historically, *programmable logic arrays (PLAs)* were the first programmable logic devices. PLAs contained a two-level structure of AND and OR gates with user-programmable connections. Using this structure, a designer could accommodate any logic function up to a certain level of complexity using the well-known theory of logic synthesis and minimization. PLA structure was enhanced and PLA costs were reduced with the introduction of *programmable array logic (PAL) devices*. Today, such devices are generically called programmable logic devices (PLDs), and are the "MSI" of the programmable logic industry.

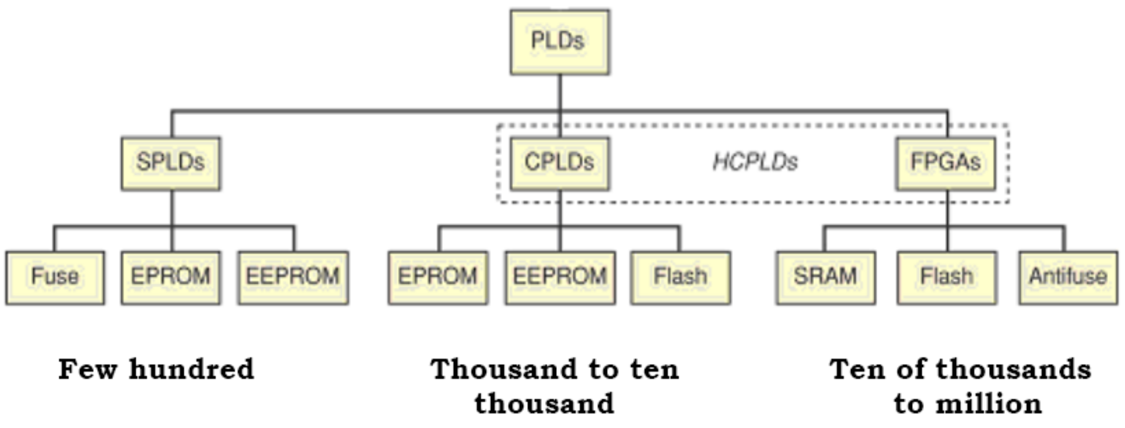
The ever-increasing capacity of integrated circuits created an opportunity for IC manufacturers to design larger PLDs for larger digital-design applications. However, for technical reasons that we'll discuss in \ secref{CPLDs}, the basic two-level AND-OR structure of PLDs could not be scaled to larger sizes. Instead, IC manufacturers devised *complex PLD (CPLD)* architectures to achieve the required scale. A typical CPLD is merely a collection of multiple PLDs and an interconnection structure, all on the same chip. In addition to the individual PLDs, the on-chip interconnection structure is also programmable, providing a rich variety of design possibilities. CPLDs can be scaled to larger sizes by increasing the number of individual PLDs and the richness of the interconnection structure on the CPLD chip.



Programmable Logic Devices Types:

1. Read Only Memory (ROM).
2. Programmable Logic Array (PLA).
3. Programmable Array Logic (PAL).
4. Programmable Gate Array (PGA).

☞ PLDs family tree with process technology and logic gates capacity



Types Programmable Logic Devices Diagram:

Programmable Logic Array (PLA))

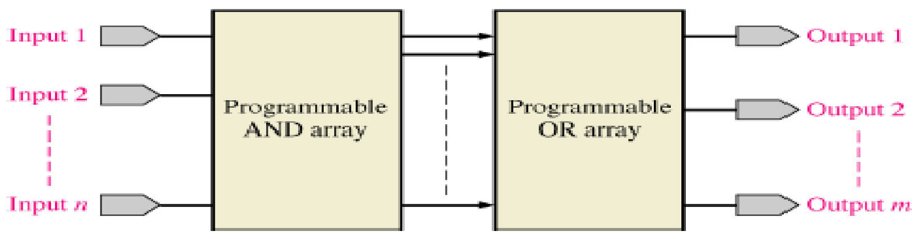


Figure 3--68 Block diagram of a PLA (programmable logic array).

• The PLA was developed to overcome some of the limitations of PROM. The PLA is also called an FPLA (Field programmable logic array) because the user in the field, not the manufacturer, programs it.

Programmable Array Logic (PAL)

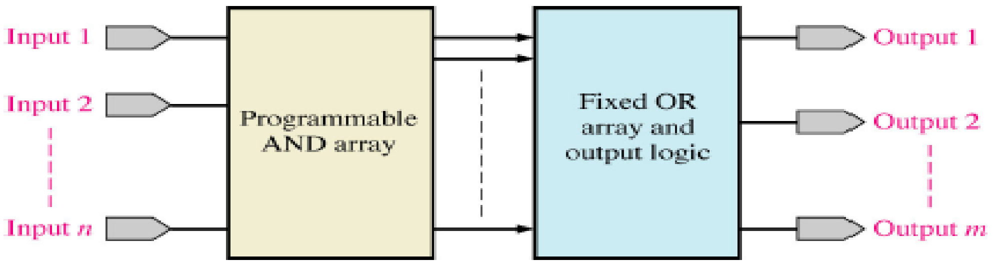


Figure 3--69 Block diagram of a PAL (programmable array logic).

• It was developed to overcome certain disadvantages of PLA, such as longer delays due to the additional fusible links that result from using two programmable arrays and more dificult complexity. The PAL is most common one-time programmable (OTP) logic device and is implemented with bipolar technology (TTL or ECL)

Introduction of PLD types:

Read-only memory (ROM) is a class of storage medium used in computers and other electronic devices. Data stored in ROM cannot be modified, or can be modified only slowly or with difficulty, so it is mainly used to distribute firmware (software that is very closely tied to specific hardware, and unlikely to need frequent updates).

A programmable logic array (PLA) is a kind of programmable logic device used to implement combinational logic circuits. The PLA has a set of programmable AND gate planes, which link to a set of programmable OR gate planes, which can then be conditionally complemented to produce an output. This layout allows for a large number of logic functions to be synthesized in the sum of products (and sometimes product of sums) canonical forms.

The term Programmable Array Logic (PAL) is used to describe a family of programmable logic device semiconductors used to implement logic functions in digital circuits introduced by Monolithic Memories, Inc. (MMI) in March 1978.[1] MMI obtained a registered trademark on the term PAL for use in "Programmable Semiconductor Logic Circuits". The trademark is currently held by Lattice Semiconductor.

PAL devices consisted of a small PROM (programmable read-only memory) core and additional output logic used to implement particular desired logic functions with few components.

Field-Programmable Gate Array (FPGA): is a VLSI circuit whose function is defined by a user's program rather than by the manufacturer of the device (CPEN431)

• Depending on the particular device, the program is either 'burned' in permanently or semi-permanently as part of a board assembly process, or is loaded from an external memory each time the device is powered up.

• The Field-Programmable Gate Arrays provide the benefits of custom CMOS VLSI, while avoiding the initial cost and time delay.

Using specialized machines, PAL devices were "field-programmable". Each PAL device was "one-time programmable" (OTP), meaning that it could not be updated and reused after its initial programming. (MMI also offered a similar family called HAL, or "hard array logic", which were like PAL devices except that they were mask-programmed at the factory.)

In its strictest sense, ROM refers only to mask ROM (the oldest type of solid state ROM), which is fabricated with the desired data permanently stored in it, and thus can never be modified. Despite the simplicity, speed and economies of scale of mask ROM, field-programmability often make reprogrammable memories more flexible and inexpensive. As of 2007[update], actual ROM circuitry is therefore mainly used for applications such as microcode, and similar structures, on various kinds of digital processors (i.e. not only CPUs).

Other types of non-volatile memory such as erasable programmable read only memory (EPROM) and electrically erasable programmable read-only memory (EEPROM or Flash ROM) are sometimes referred to, in an abbreviated way, as "read-only memory" (ROM), but this is actually a misnomer because these types of memory can be erased and re-programmed multiple times.[1] When used in this less precise way, "ROM" indicates a non-volatile memory which serves functions typically provided by mask ROM, such as storage of program code and nonvolatile data

The first three types based on the two level AND-OR cct structure, while the last one uses a more general gate structure to implements ccts. All these devises are available in mask and field programmable versions. We used the Fig (1-A), to illustrate the difference between the ROM, PLA and PAL structures and differ description of PGA structure.

AND Array

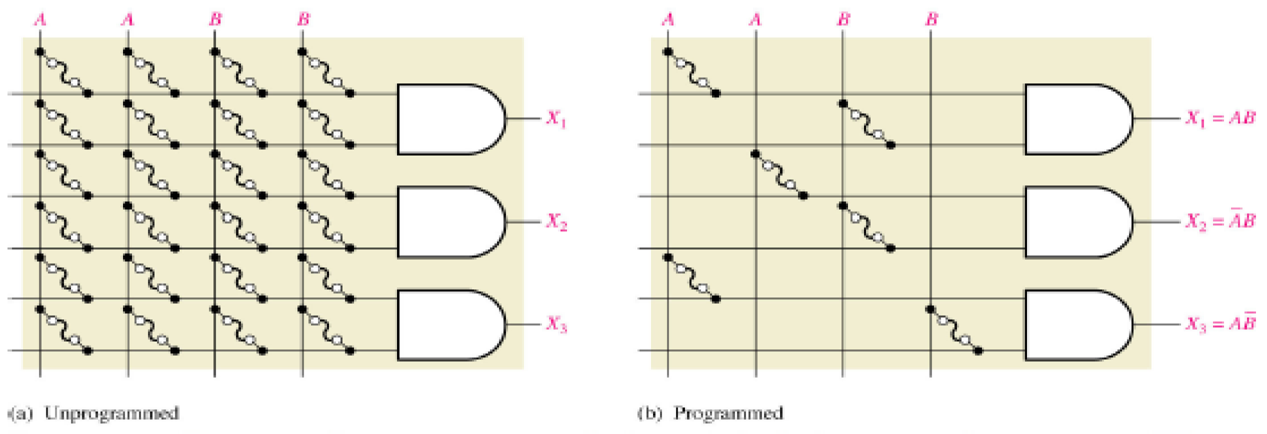


Figure 3--66 An example of a basic programmable AND array.

OR Array

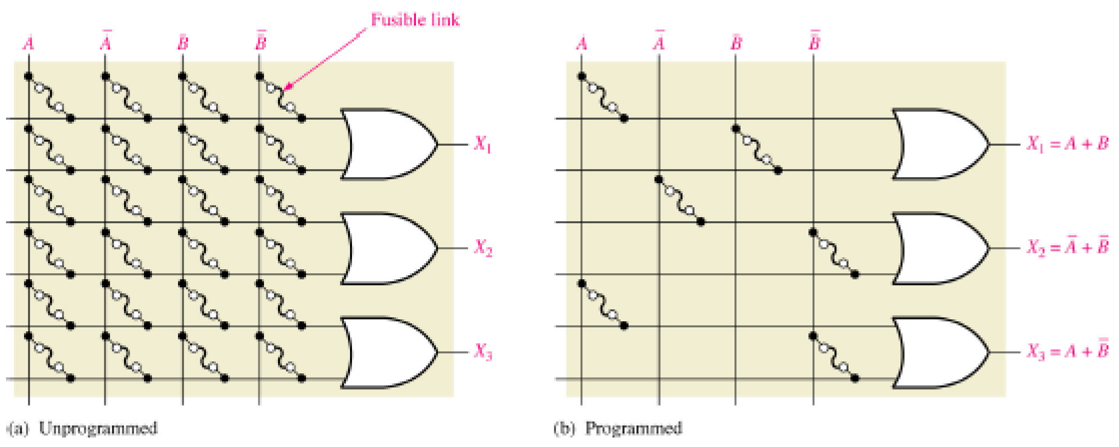


Figure 3--65 An example of a basic programmable OR array.

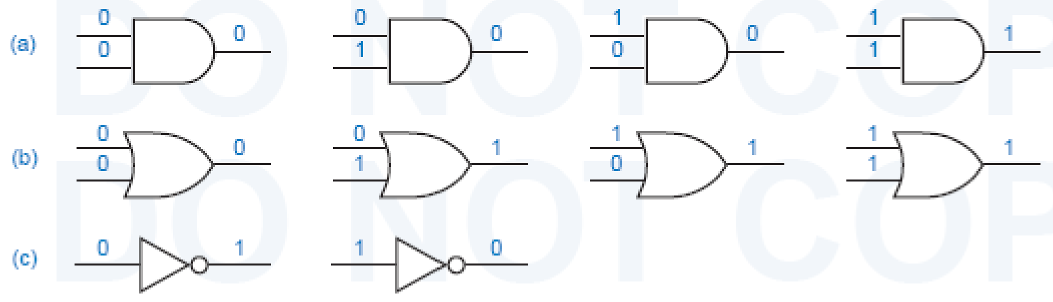
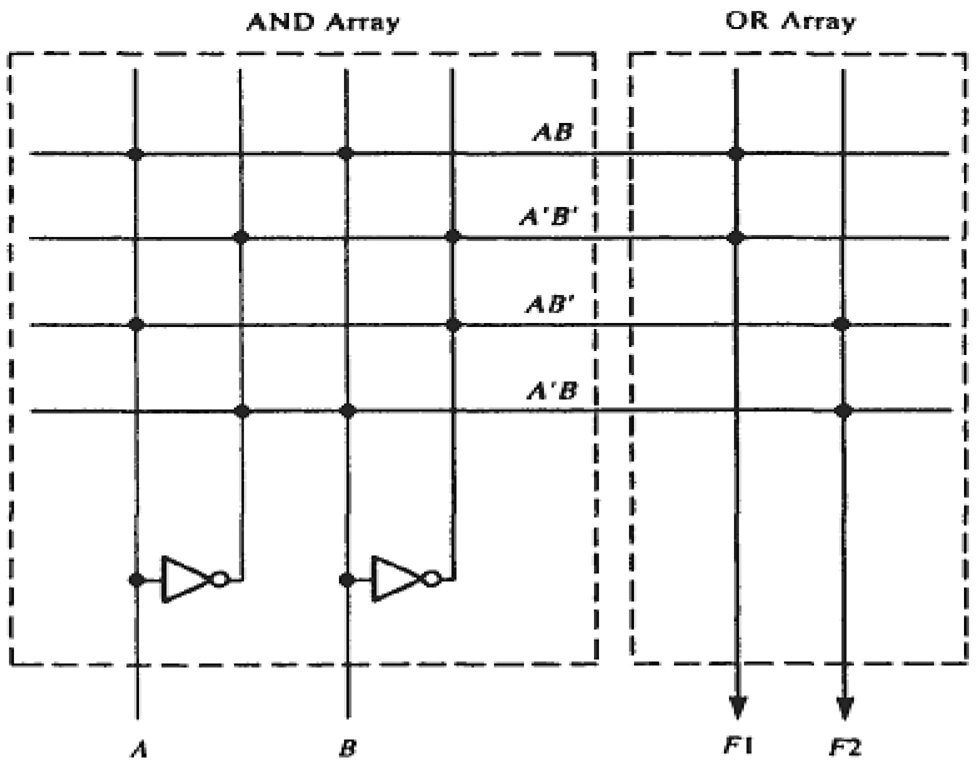


Figure 1-1 Digital devices: (a) AND gate; (b) OR gate; (c) NOT gate or inverter.



Fig(1-A).
Implementation circuit AND-OR

F1 = AB + A'B'
&
F2 = AB' + A'B

The first level is implemented by an AND array. Each column corresponds to an input variable or its complement, and each row corresponds to minterm of the input variables. The second level is implemented by an OR array. Each column corresponds to the OR combination of selected minterm generated by the AND array. We assumed that each row-column intersection of these arrays can be programmed. That's mean the electronic devices at these intersections can be used to either connect or disconnect the row line to the column line, and when connected, the intersection realizes either an AND or an OR operation (depending on the array at which the intersection is located).

a ROM the AND array is fabricated so that all minterms corresponding to the input variables are available, and hence it's not programmable. The OR array programmable to realize the circuit. With PAL the OR array is completely fabricated and it's not programmable while AND array is programmable. In PLA both arrays are programmable, and we will have each device in more details next.

Read Only Memories:

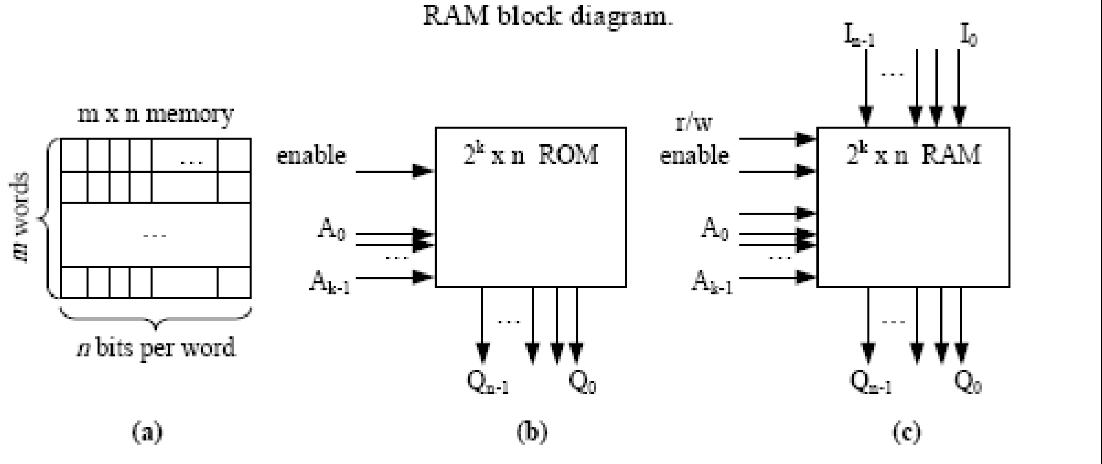
A short introduction: Any embedded system's functionality consists of three aspects: processing, storage, and communication. Processing is the transformation of data, storage is the retention of data for later use, and communication is the transfer of data. Each of these aspects must be implemented. We use processors to implement processing, memories to implement storage, and buses to implement communication. We described common processor types: general-purpose processors, standard single-purpose processors, and custom single-purpose processors. This chapter describes memories.

A memory stores large numbers of bits. These bits exist as m words of n bits each, for a total of m*n bits. We refer to a memory as an mxn ("m-by-n") memory. Log2(m) address input signals are necessary to identify a particular word. Stated another way, if a memory has k address inputs, it can have up to 2k words. n signals are necessary to output (and possibly input) a selected word. To read a memory means to retrieve the word of a particular address, while to write a memory means to store a word in a particular address. Some memories can only be read from (ROM), while others can be both read from and written to (RAM). There isn't much demand for a memory that can only be written to (what purpose would such a memory serve?). Most memories have an enable input; when this enable is low, the address is ignored, and no data is written to or read from the memory.

A ROM is built out of an array of semiconductor devices such as diodes. Bipolar transactions, and filed effect transistors, interconnected to from the AND and OR arrays. The AND array is essentially a decoder circuit that generate all the minterms of the input variables .Each row in the OR array can store one or more bits of data. The ROM device derives its name from the fact that once the data are stored, they can only be read out but cannot be charged (or written into) under usual operating conditions. Special devices are needed to write data into a ROM .We will exam various e types of ROMs later in this section .We will first provide a model for ROM and illustrate the use of ROMs in the implementation of multiple –output combinational logic circuits.>while ROM is a memory device,in the application to be describe here it will be used to construct purely combinational circuits

ROM, or read-only memory, is a memory that can be read from, but not typically written to, during execution of an embedded system. Of course, there must be a mechanism for setting the bits in the memory (otherwise, of what use would the read data serve?), but we call this "programming," not writing. Such programming is usually done off-line, i.e., when the memory is not actively serving as a memory in an embedded system. We usually program a ROM before inserting it into the embedded system. Figure A-1(b) provides a block diagram of a ROM.

Figure 5.1: Basic memories: (a) words and bits per word, (b) ROM block diagram, (c) RAM block diagram.

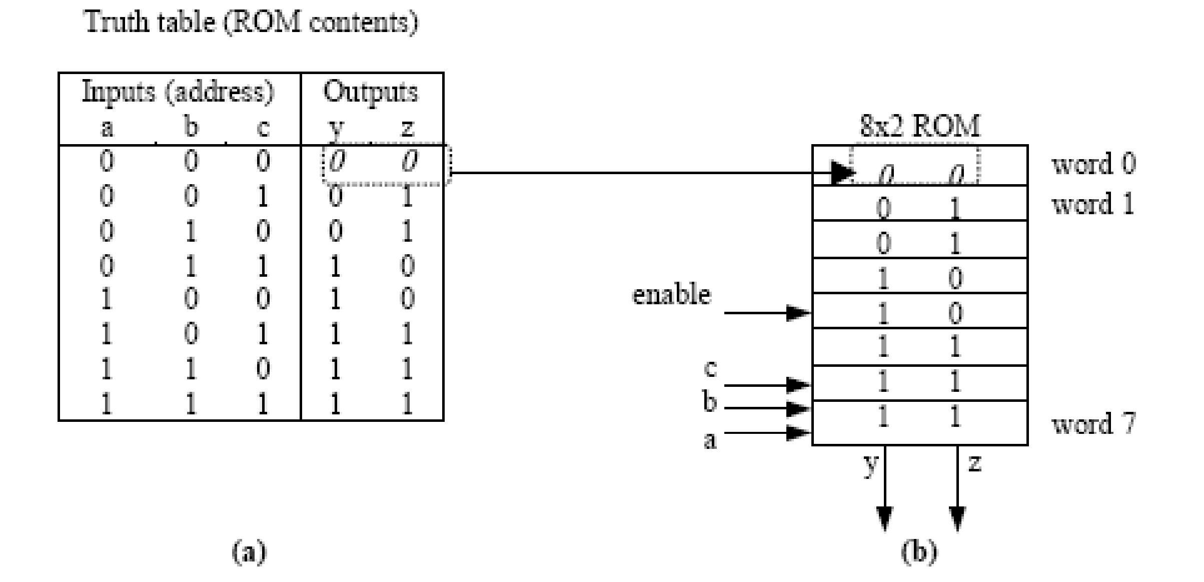


Fig(A-1)

We can use ROM for various purposes. One use is to store a software program for a general-purpose processor. We may write each program instruction to one ROM word. For some processors, we write each instruction to several ROM words. For other processors, we may pack several instructions into a single ROM word. A related use is to store constant data, like large lookup tables of strings or numbers.

Another common use is to implement a combinational circuit. We can implement any combinational function of k variables by using a 2kx 1 ROM, and we can implement n functions of the same k variables using a 2kx n ROM. We simply program the ROM to implement the truth table for the functions, as shown in Figure A-2

Figure 5.2: Implementing combinational functions with a ROM: (a) truth table, (b) ROM contents.



Truth table (ROM contents)

Inputs (address)			Outputs	
a	b	c	y	z
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

8x2 ROM

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

Fig (A-2)

Figure 3 provides a symbolic view of the internal design of an 8x4 ROM. To the right of the 3x8 decoder in the figure is a grid of lines, with word lines running horizontally and data lines vertically; lines that cross without a circle in the figure are not connected. Thus, word lines only connect to data lines via the programmable connection lines shown. The figure shows all connection lines in place except for two connections in word 2. To see how this device acts as a read-only memory, consider an input address of "010." The decoder will thus set word 2's line to 1. Because the lines connecting this word line with data lines 2 and 0 do not exist, the ROM output will read "1010." Note that if the ROM enable input is 0, then no word is read. Also note that each data line is shown as a wired-OR, meaning that the wire itself acts to logically OR all the connections to it.

How do we program the programmable connections? The answer depends on the type of ROM being used. In a mask-programmed ROM, the connection is made when the chip is being fabricated (by creating an appropriate set of masks). Such ROM types are typically only used in high-volume systems, and only after a final design has been determined.

Most other systems use user-programmable ROM devices, or PROM, which can be programmed by the chip's user, well after the chip has been manufactured. These devices are better suited to prototyping and to low-volume applications. To program a PROM device, the user provides a file indicating the desired ROM contents. A piece of equipment called a ROM programmer (note: the programmer is a piece of equipment, not a person who writes software) then configures each programmable connection according to the file. A basic PROM uses a fuse for each programmable connection. The ROM programmer blows fuses by passing a large current wherever a connection should not exist. However, once a fuse is blown, the connection can never be re-established. For this
Reason, basic PROM is often referred to as one-time-programmable device, or OTP.

Another type of PROM is an erasable PROM, or EPROM. This device uses a MOS transistor as its programmable component. The transistor has a "floating gate," meaning its gate is not connected. An EPROM programmer injects electrons into the floating gate, using higher than normal voltage (usually 12V to 25V) that causes electrons to "tunnel" into the gate. When that high voltage is removed, the electrons cannot escape, and hence

Figure 5.3: ROM internals

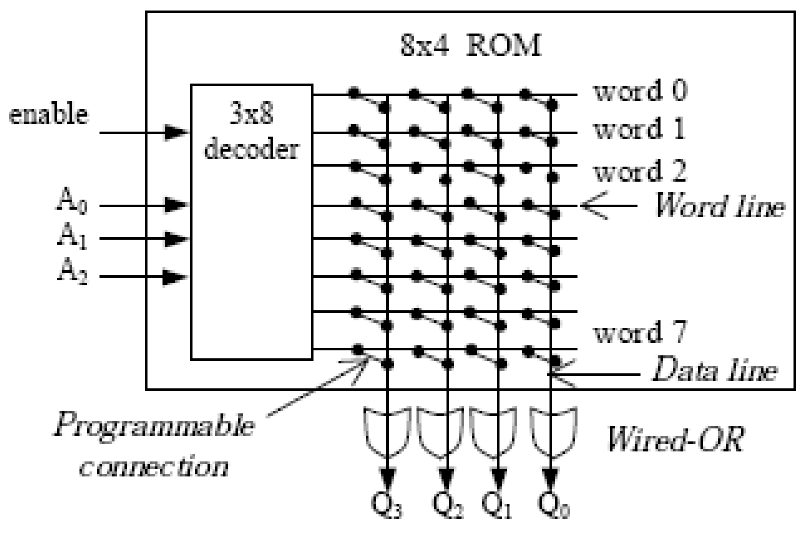
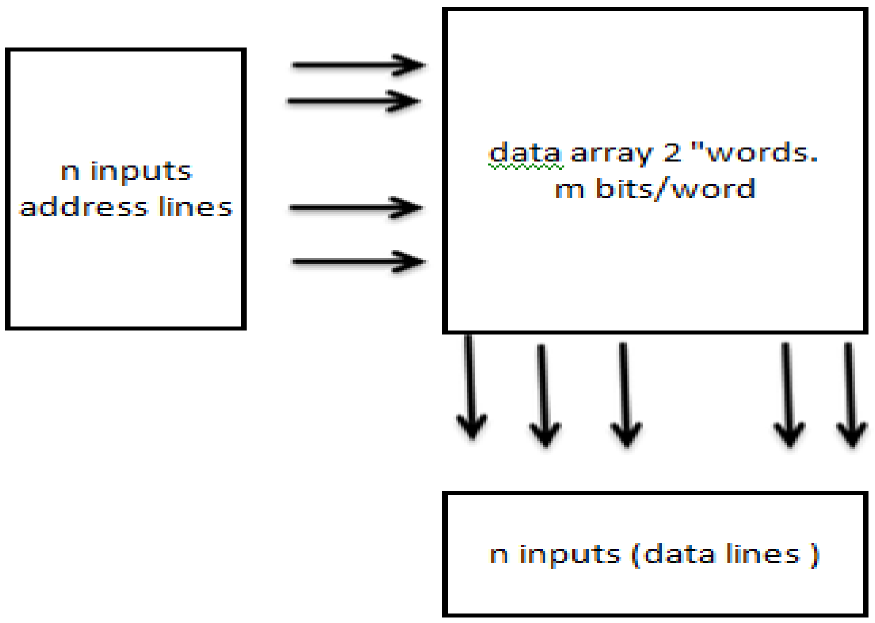


Fig (A-3)

A ROM is an n-input/m-output as illustrated in Fig.(1-B). Bit combinations of the *n*. which can generate 2″ minterms, and the decimal rang will be from 0 to 2″*-1*



(Fig1-B)
Block Diagram showing Read Only Memory and its size 2″ words × m bits.

There are *n* possible addresses to a ROM, there are *2″* possible *words* that can be stored in the ROM, each word being *m* bits in size. Any m-bit output word programmed into the ROM can be selected by the appropriate input address and is nonvolatile — it is stored permanently in the ROM.

ROM Model

The dimensions and size of an n-input/m-output ROM are given by:

2″ × m = (2n)(m)bits

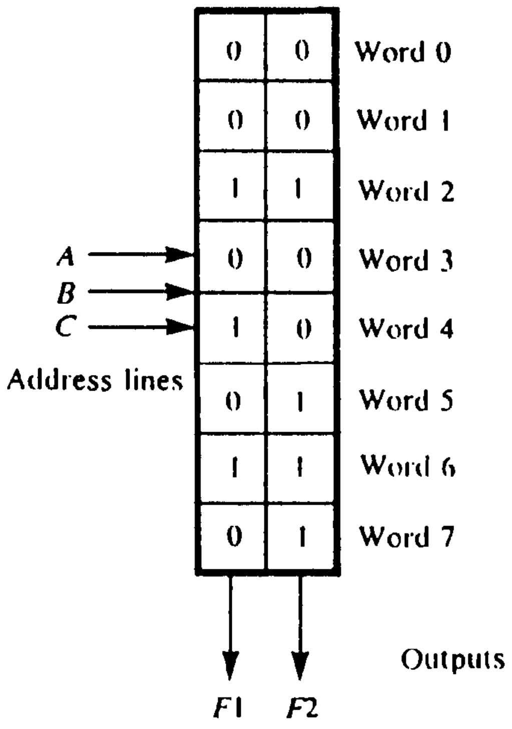
Meaning that *2″* words, each of *m* bits, produce a total of (2")(m) bits. The size of a ROM may be rounded off to the nearest integer power of 2 in K (103) bits of ROM. For example, an 8-input/4-output ROM is represented as a 28 x 4 = 256 x 4 bit = 1,024 bit or I Kbit ROM. The problem with the bit-round off form of representation is that knowledge of the dimensions is lost. Thus, a I Kbit ROM could be any of the following: 27 x 8 = 26 x 16, etc. This can be avoided by giving the ROM size in dimension (2" x m) form, which clearly specifies the number of inputs and outputs.
To implement an n-input, m-output combinational circuit, a 2^n × m ROM is needed. The output portion (m columns) of the truth table is stored in the ROM words. Corresponding to each input

Combination (address), the ROM produced the appropriate output on the output lines; the example below shows the design.

Example:-

This ROM implementation shows two output functions as shown in Fig(1-4).

Fig(1-C)ROM implementation of two-output function.



0	0	Word 0
0	0	Word 1
1	1	Word 2
0	0	Word 3
1	0	Word 4
0	1	Word 5
1	1	Word 6
0	1	Word 7

The two output functions are:

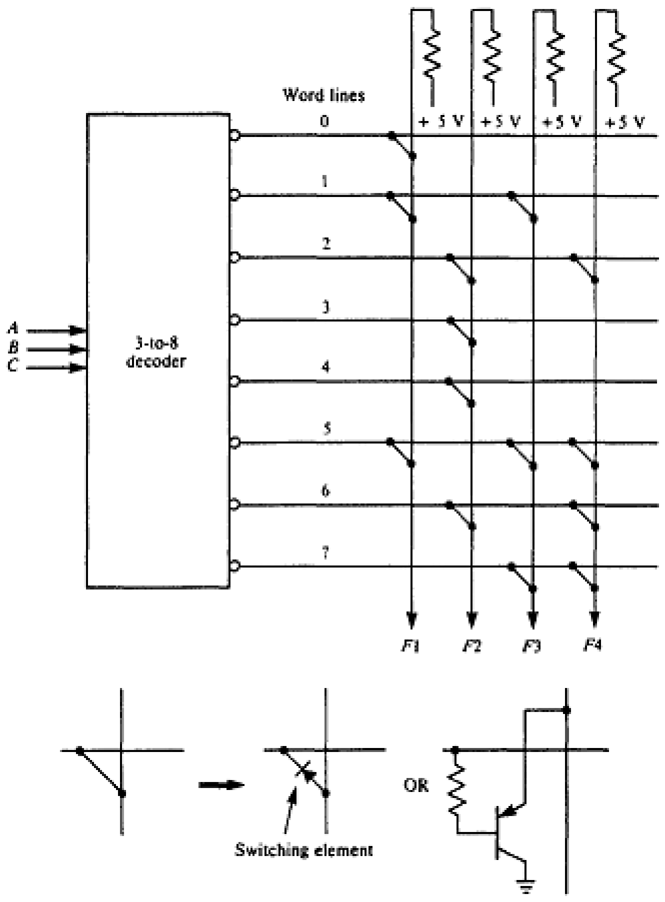
F1 (A,B,C) = $\sum$m (2,4,6)

And since there are three inputs, a ROM with a three input lines and 2^3 or 8 words of storage is needed. Each word should be two bits wide, one bit for each output of the function being implemented

F2 (A,B,C) = Σm (2,5,6,7)

So, 4-Bit ROM word is needed, since there are four outputs from the circuit. The 3-to-8 decoder generates 8 minterms of the three variable functions. Each minterm corresponds to a word line in the ROM. Each vertical line in the memory array corresponds to the output, only those word lines that correspond to the minterms of the output (the output is 0) are connected to the vertical output line by switching element (diode or transistor). As shown in Fig(1-5).

The decoder produces active low outputs, when the selected word line goes low, the switching element connected to that line conduct, driving the corresponding vertical (bit) line low. Thus, each vertical line implements a wired-AND function of all the word lines connected to it. The outputs produced by the circuit thus active low.



Fig(1-D)

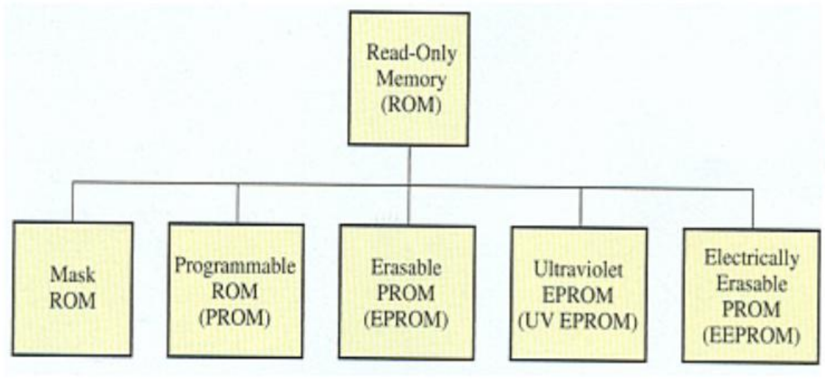
Truth Table

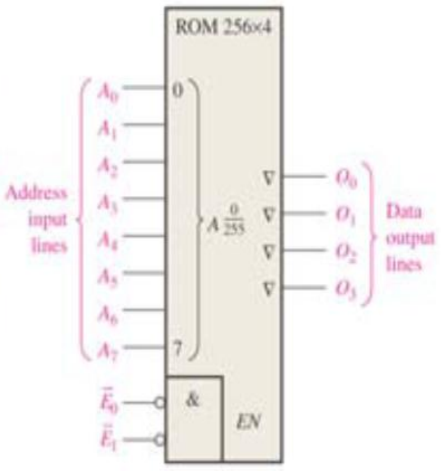
ROMs may differ in a variety of ways, but the main differences center about the manner in which they are programmed, and on whether or not they can be erased and reprogrammed and how this is accomplished.

Types Of ROM:

• Read-only memories (ROMs)................................ Mask programmable OR stage only
• Programmable ROMs (PROMs)............................. User programmable once
• Erasable PROMs (EPROMs)................................... User erasable many times

☞ The ROM family and logic symbol





Both MOS and Bipolar technologies offer ROMs with varying speed. The data access speed of the ROM is the time it takes the ROM to produce as output once the address is input to it. Typical speed are around 50ns for bipolar, and 400ns for MOS ROMs. The PROM types available are shown below:

Bipolar	**MOS**

Mask Programmable PROM	Mask Programmable PROM
Fusible Link	Fusible Link
	Ultraviolet
	Electrically alterable

In A mask-Programmable ROM , the data array is permanently stored during fabrication . This is done by selectively including switching element where a 1 is desired in the data array .the designer of the circuit should provide the ROM program, which is simply the content of the storage array to the IC manufacture. Once the ROM is fabricated, the data array cannot be charged. Mask_-prorgammable ROMs are used when the ROM contents are not expected to change during the lifetime of the ROM .Fusible –link PROMs are manufactured with the switching elements in the place at all the row-Colum intersection of the data array ,with the connection made by "”fusible”" links. To program this ROM the selected links are "Blowm" by sending the appropriate voltage pulse generated by a special links device .known as a ***PROM programmer***. The data once stored cannot be changed. This type of ROM is appropriate when the limited quantities needed do not justify a made-programmable ROM

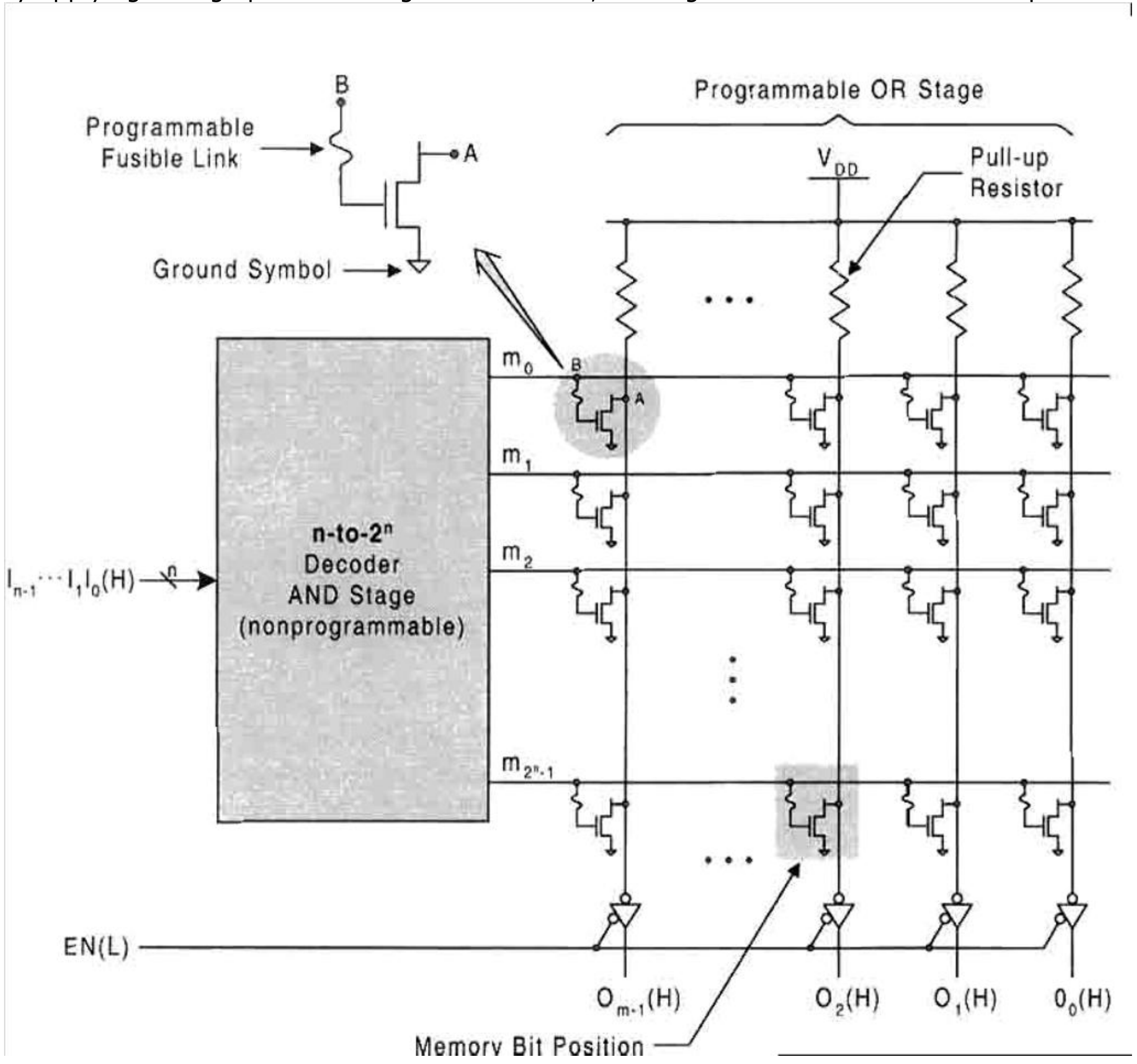
Erasable Programmable Read-Only Memory (EPROM) is a special type of memory that retains its Contents until it is exposed to ultraviolet light.
To write to EPROM, you need a special device called a PROM Programmer or PROM burner (programmer). An EPROM differs from a PROM in that a PROM can be written to only once and cannot be erased.

EPROMs are widely used in personal computers since they enable the manufacturer to change the contents of the PROM before the computer is actually shipped. This means that bugs can be removed
and new versions installed shortly before delivery.

EPROMs use a special charge –storage mechanism to enable or disable the switching element in the data array. A PROM programmer is used to store the charge at the selected switching elements while the EPROMs is programmed .The charge is retained by the EPROM. Thereby retaining the program until the EPROM is erased by using an ultraviolet light. Once erased, the EPROM can be reprogrammed .this type of ROM is useful in the early development phased of Digital circuit design, when it is often necessary to modify the data array

Because the masking process is expensive, the use of mask-programmable ROMs is economically justifiable only if large numbers are produced to perform the same function. When one or a few ROM-type devices are needed to perform certain functions, PROMs can be very useful. Most PROMs are fabricated with *fusible links* on all transistors (or diodes) in the OR memory stage, thereby permitting user programming of the ROM — a write-once capability. Shown in Fig.(1-6) is the circuit for an unprogrammed 2" x *m* PROM

The PROM of Fig. (1-6) is programmed (one time only permitted) with a PROM programmer by applying voltage pulses to target fusible links, causing disconnection of these bit positions.



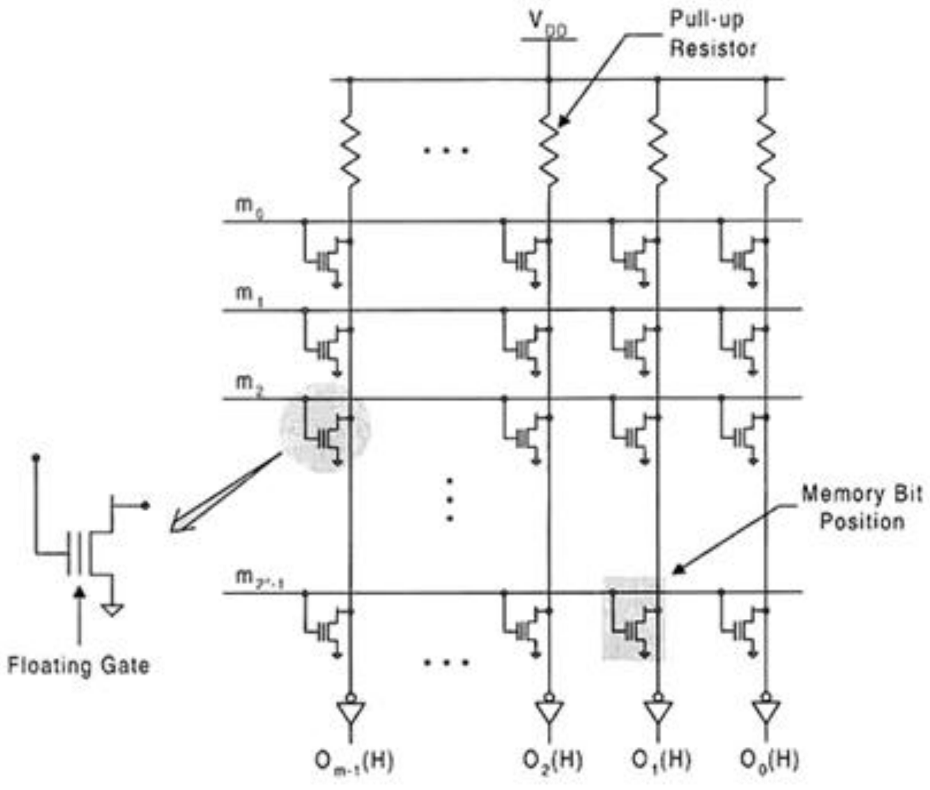
Fig(1-E)

Logic circuit for an unprogrammed *2″* × *m* PROM showing the nonprogrammable decoder and programmable NMOS connections (fusible link) for each normally active bit location in the OR (memory) section. Tri-state drivers provide an enable capability.

The masking process of a mask-programmable ROM places NMOS connections at predetermined (programmed) memory bit positions. The positioned NMOS connections would look similar to those in Fig. 7.2, except their fusible links would be missing. Because the masking process is expensive, mask-programmable ROMs are used only for high-volume commercial applications. Much more useful, generally, are the EPROMs, since they can be programmed, erased, and reprogrammed many times. These devices fall into two main categories: *ultraviolet erasable PROMs* (UVEPROMs) and *electrically erasable PROMs* (EEPROMs). In either case the technology is similar — use is made *of floating-gate NMOS transistors* at each memory bit location, as illustrated by the OR memory stage in Fig. 7.3. Each transistor in Fig. 7.3 has two gates, a connected outer gate, and an inner floating (unconnected) gate that is surrounded by a highly insulating material. Programming occurs when a high positive voltage is applied to the connected gate inducing a negative charge on the floating gate which remains after the high voltage is removed. Then, when a decoder line becomes active (HV), the negative charge prevents the NMOS from being turned ON, thereby maintaining a 1(H) at the memory bit position. This is equivalent to blowing a fusible link in Fig. 7.2. If all floating-gate NMOS in an OR column are so programmed, the output from the inverter is 0(H). But if a decoder line is active to any unprogrammed floating-gate NMOS, that bit position will be pulled to ground 0(H), causing the output to be 1(H) from the inverter — again, the OR function.

Erasure of a programmed floating-gate NMOS occurs by removing the negative charge on its floating gate. This charge can remain on the gate nearly indefinitely, but if the floating gate in a UVEPROM is exposed (through a "window") to ultraviolet light of a certain frequency, the negative charge on the floating gate is removed and erasure occurs. Similarly, if a voltage of negative potential is applied to the outer connected gate of an EEPROM, removal of the negative charge occurs. The technology for the UVEPROMs and EEPROMs differ somewhat as to the manner in which the floating gate is insulated; otherwise, they share much in common.

Technologies other than those just described are used to manufacture PROMs. For example, in *bipolar* PROMs, diodes with fusible links replace the NMOS in Fig. 7.2 with each diode conducting in the *A* -> *B* direction (see blowup of the fusible link). Now, however, the decoder has active low outputs. Then, when a decoder output becomes active, 1(L) = 0(H), a bit position for a connected diode is pulled low, 0(H). A disconnected (programmed) diode maintains a 1(H) at the bit position when selected. If all diode bit positions in an OR column are disconnected, then the output for that column must be 0(H) from the inverting tri-state driver. However, if one (or more) of the bit position diodes in an OR column is left connected and is selected by an active low decoder line, the output will be 1(H) from the tri-state driver. The diode technology, used extensively in the early stages of ROM development, is now in less common use than MOS technology.



Fig(1-F)
Logic circuit for an unprogrammed OR (memory) section of an EPROM illustrating the floating-gate NMOS transistor technology required for a program/erase/program cycle.

Both Bipolar and MOS ROMs are available in various configuration ranging in capacity from 256 bits to 64 k Bits. Typical word lengths are 4,8 and 16 bits. The access time range from 15 to 100 ns for Bipolar ROMs and from 100 to 450 ns for MOS ROMs. ROMs of greater storage capacity and shorter access time are continually being announced.

PROM Applications:-

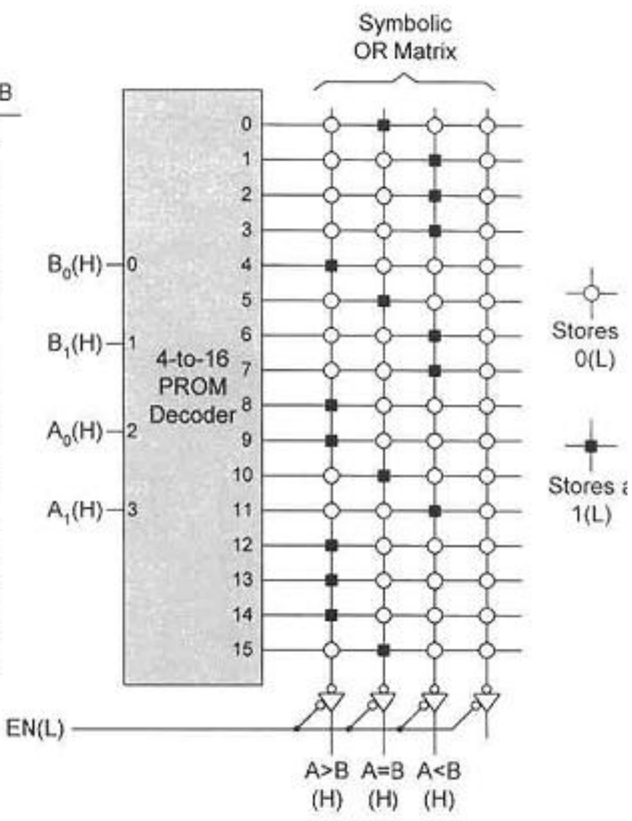
- Comparator…

The AND section of any ROM device is nonprogrammable decoder, and since a decoder is a minterm code generator the following requirement must be met for ROM programming:

To program a ROM, all input and output date must be represented in canonical form.

AB	A_1	A_0	B_1	B_0	A>B	A=B	A<B
00	0	0	0	0	0	1	0
01	0	0	0	1	0	0	1
02	0	0	1	0	0	0	1
03	0	0	1	1	0	0	1
10	0	1	0	0	1	0	0
11	0	1	0	1	0	1	0
12	0	1	1	0	0	0	1
13	0	1	1	1	0	0	1
20	1	0	0	0	1	0	0
21	1	0	0	1	1	0	0
22	1	0	1	0	0	1	0
23	1	0	1	1	0	0	1
30	1	1	0	0	1	0	0
31	1	1	0	1	1	0	0
32	1	1	1	0	1	0	0
33	1	1	1	1	0	1	0

(a)



Fig(1-G)

A 32 × 8 Bipolar PROM (82S23/82S123

Programmable Read-Only Memory (PROM)

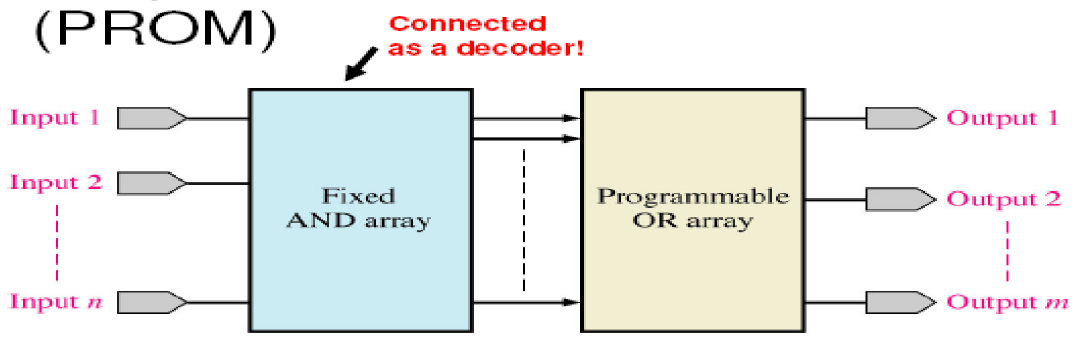


Figure 3--67 Block diagram of a PROM (programmable read-only memory).

• The PROM is used primarily as an addressable memory and not as a logic device because of limitations imposed by fixed AND gates.

Fig-PROM diagram

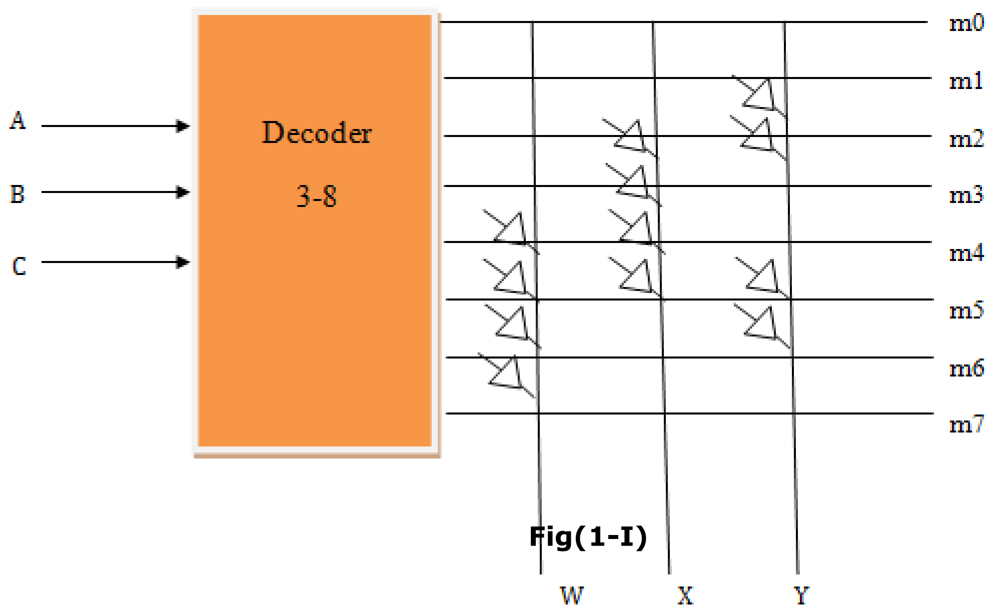
This fact is illustrated by considering the noncascadable 2-bit comparator represented by truth table in Fig(1-9), by setting GT=1 when A>B, EQ=1 when A=B, and LT=1 when A<B, or by setting these inputs to zero otherwise. Shown in Fig(1-9) is a $2^{٤}$ × 4 PROM that has been programmed to function as the 4-input/3-output noncascadable 2-bit comparator.

Programming a PROM that functions as a cascadable 2-bit comparator would require a PROM with dimensions of 2 × 3, a significant increase in hardware. In fact to do so might seem to be hardware overkill, considering that 7-to-128 line decoder section would be required.

- Code Converter…

The ROM can be programmed to convert digital codes. Fig(1-10) shows a 3-bit BCD to Gray converter using ROM when :

W = A
X = A + B
Y = B + C



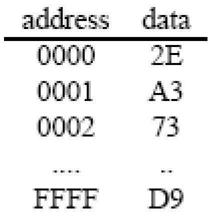
Fig(1-I)

$2^3 \times 3$ ROM to convert 3-bit BCD-to-Gray

IC Fabrication:-

Microprocessor systems use memory ICs to store programs and data. This lecture describes common semiconductor memory devices and how they are organized in microprocessor memory systems. After this lecture you should be able to select the appropriate type of memory device for different applications, combine memory ICs to form memory arrays, and design address decoders using SSI decoders and PLDs.

ROM :The simplest memory IC is a ROM (read-only memory). A ROM can be described as a combinational logic circuit that implements an arbitrary *B*-bit function of *N* bits .The *N*-input bits are known as the *address* and the *B outputs* bits are the *data* .such a device is refer to as a $2^N$by B memory .For example a 4096 by 8 (4kx8)ROM would have 12 address inputs and 8 data pins
Like any combinational logic circuit ROM can be described using a lookup table .The following table shows some of the contents (in hexadecimal) of a hypothetical byte-wide device



address	data
0000	2E
0001	A3
0002	73
....	..
FFFF	D9

Integrated circuit:
a collection of one or more gates fabrication on a single silicon chip is called ***integrated circuit*** (IC) large ICs with tens of millions of transistors may be half an inch or more on a side .while small ICs may be less than one-tenth of an inch a side .
Regardless of it is size .an IC is initially part of much larger, circular Wafer uo to ten inch in diameter, containing dozens to hundred s of replicas of the same IC.All of the IC chips on the wafer are fabricated at the same time .like pizzas that are eventually sold by slice .except in this case each pics (IC) chip is called a *die.* after the Wafer fabricated the dice are tested I place on the Wafer and defective ones are marked .Then the Wafer is sliced up to produce the individual dice , and the marked once are discarded (compare with pizza make who sells the pieces ,even the one without enough pepperoni!).Each unmarked dice is mounted in a package .its pads are connected to the package pins, and the packaged IC is subjected to final test is shipped to a customer
Some people use term of IC to refer to silicone die .some use “chip” to refer to the same thing “chip” to refer, still other sue “IC” or “chip” refer to the combination of the silicon die and it is package .Digital designer tend to use tow terms interchangeably .and they really do not care what they are talking about .they don’t require a precise definition. Since they are only looking at the functional and electrical behavior of these things

Fig (1-11) shows the steps in the IC manufacturing process. The process starts off with a thin (10 mil think) slice of P-type semiconductor material, about two to five inches in diameter called a Wafer. Hundreds of identical circuits are fabricated on the Wafer.

Using multi step process. The the cut into individual dice, each dice corresponding to an IC

The fabrication process consists of the following steps:

- Wafer surface preparation.
- Epitaxial growth.
- Diffusion.
- Metallization.
- Probe test.
- Scribe and break.
- Packaging.
- Final test.

We must develop the circuit diagram before fabrication. Once the circuit is designed, it is usually simulated to verify its functional, timing, and loading characteristics. If these characteristics are acceptable, the circuit is brought to the placement and routing stage.

Although the steps vary depending on the process and the technology used to manufacture the IC, the diffusion can be classified in to ***isolation***, ***base,*** and ***emitter diffusion*** stages .these are the stages in which the corresponding terminals of the transistor are fabricated; each diffusion stage corresponds to one or more mask and etch operation.
By now the wafer contains several identical die with all the circuit components formed on each die .The wafer is then subjected to photon etching tom open the window to provide the connection between the components .The interconnection are made (i.e. through metallization) by vacuum deposition of a thin film of aluminum over the entire wafer .followed by another mask and etch operation to remove unnecessary interconnections.
Among the dice now son the wafer, some may be defective as a result of imperfection in the wafer or in the fabrication process .selected dice are now tested to mark the failing ones. The percentage of good dies obtained is called the "yield "of the fabrication process.
The wafer is now scribed by a diamond-tipped tool along the bound aries of the dice to separate them in to individual chips.
Each die is then tested and mounted on a leader, and pins are attahce4d and package either in inline or flat package
Circuit layout and mask preparation are the most time –consuming and error –prone stage in the fabrication process and hence contribute most to the design cost .When the circuit is very complex the circuit layout require thousands of labor-hours, even the use of CAD tools

References:-

1. PLD BY Mantick Devreleri -Yard Doc Dr Mutlu BOZTEPE
2. Digital Design. third edition By John F. Wakerly
3. Introduction To Logic Design – 2nd Edition. By Sajjan G. Shjiva
4. Engineering Digital Design – 2nd Edition. By Richard F. Tinder.
5. Internet: wikipedia.org.