Digital System Design

Lecture – 6

Memory Device (ROM and PLA)

Memory Device

- Device to which binary information is transferred for storage
- And from which information is available for processing as needed

Memory Unit

- A collection of cells capable of storing a large quantity of binary information

Types of Memories

In digital systems, there are two types of memories:

1. RAM
2. ROM

ROM (Read Only Memory)

- Non-volatile
- Retains its contents even when the computer is shut off
- Generally used to start the computer up and load the operating system
- **A memory device in which a fixed set of binary information is stored**
- **The binary information must first be specified by the user**
- **Then embedded in the unit to form the required interconnection pattern**

ROM (Contd.)

- Includes both the decoder and the OR gates within a single IC package
- Particular function implementation is done by **"programming"**
- Very often used to implement complex combinational circuit in one IC package
- Thus eliminates all interconnecting wires
- Once a pattern is established, it remains fixed even when the power goes off
- **ROM has special internal links that can be fused or broken**

ROM Structure (Contd.)

- Consists of **n input lines** and **m output lines**
- **2^n distinct addresses** possible with **n inputs**
- Each bit combination of the input variables called as **address**
- Each bit combination coming out of the output lines known as **word**
- Number of **bits per word** = number of output lines **m**
- So, **a word equals m bits**
- An output word can be selected by a unique address
- A ROM is defined by number of words **(2^n)** and the number of bits **(m)** per word.
- $2^n \text{ X } m$

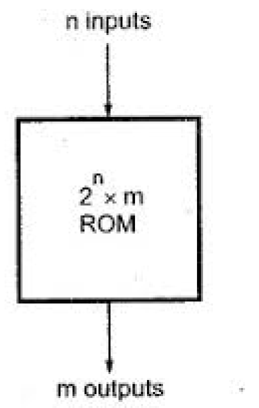
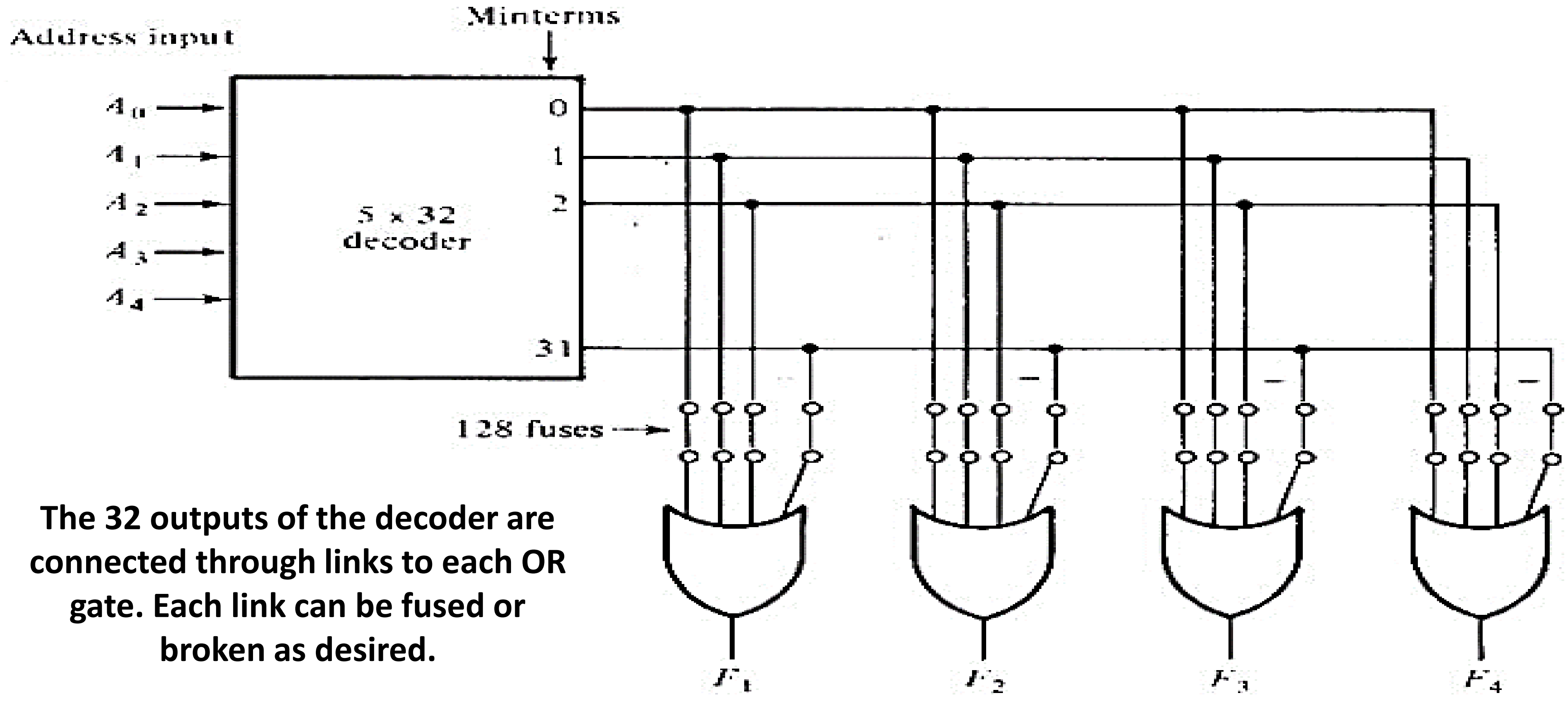


Fig. 3.82 Block diagram of ROM

An Example of a ROM

- Consider a 32×8 ROM i.e, **256-bit ROM**
- The unit consists of 32 words of 8 bits each
- Here $32=2^5$,means 5 input lines
- Input 00000 means word number 0 will be selected
- Input 11111 means word number 31 will be selected
- **Design a 2048-bit ROM having word size 8 bits each?**
- **Design a 2048-bit ROM having word size 4 bits each?**

Internal Logic Construction of a 32 X 4 ROM



The 32 outputs of the decoder are connected through links to each OR gate. Each link can be fused or broken as desired.

Combinational Logic Implementation Using ROM

For an n-input, m-output combinational circuit, we need $a\ 2^n$ X m ROM

$$F_1(A_1, A_0) = \Sigma(1, 2, 3)$$

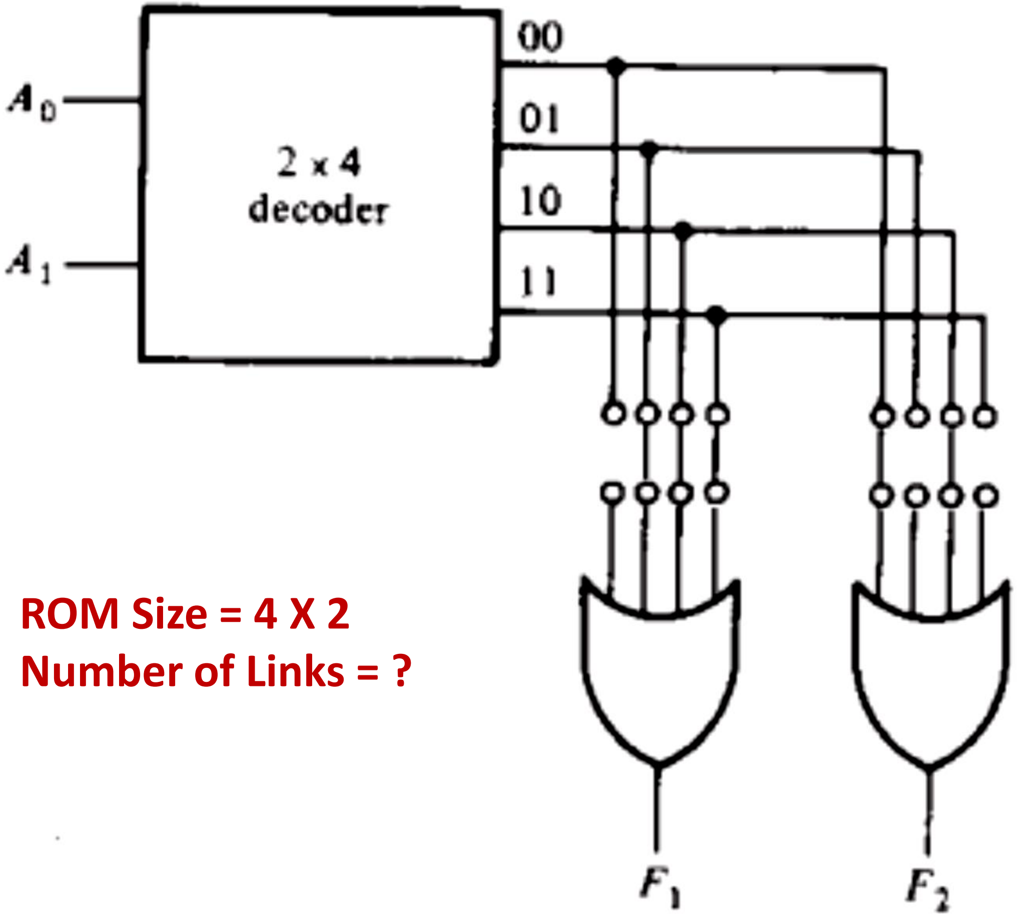
$$F_2(A_1, A_0) = \Sigma(0, 2)$$

A_1	A_0	F_1	F_2
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0

(a) Truth table

When a combinational circuit is implemented using ROM, the functions must be expressed in sum of minterms or by a truth table.
The truth table gives all the information for *programming* a ROM.

ROM with AND-OR Gates



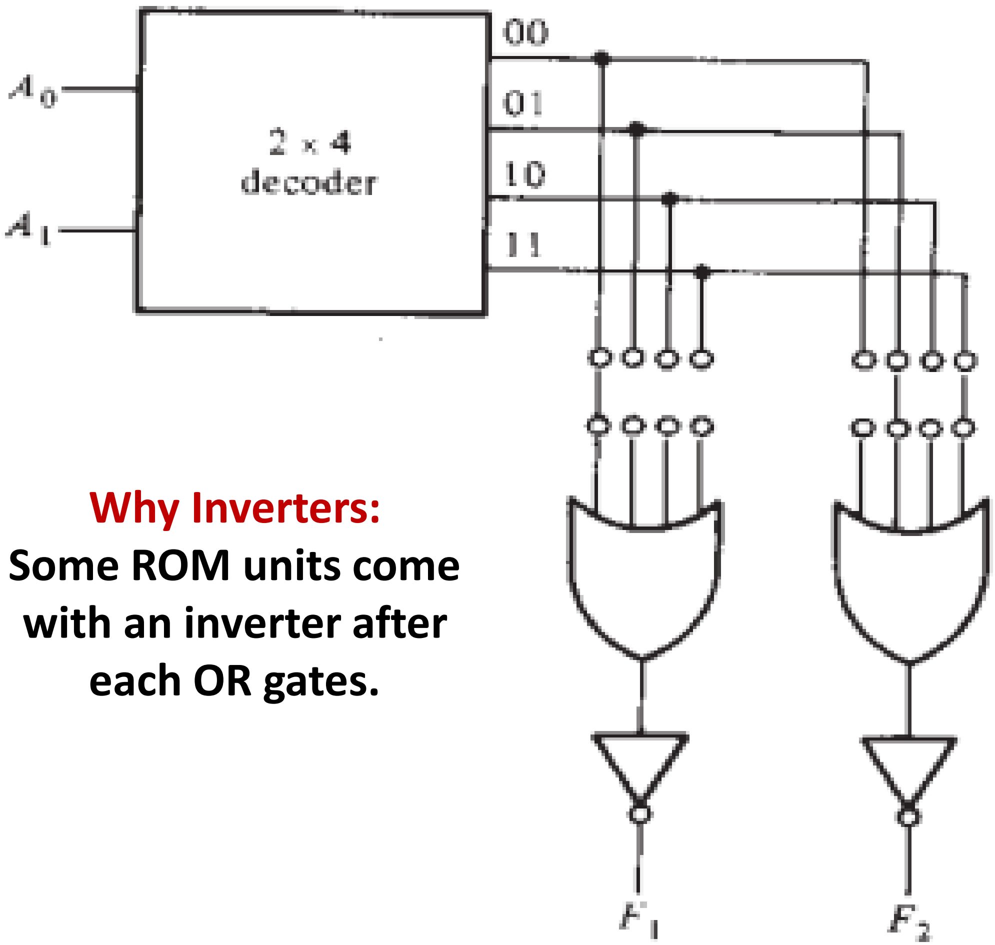
(b) ROM with AND-OR Gates

ROM Size = 4 X 2
Number of Links = ?

A_1	A_0	F_1	F_2
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0

(a) Truth table

ROM with AND-OR-Invert Gates



(c) ROM with AND-OR-Invert Gates

Why Inverters:
Some ROM units come with an inverter after each OR gates.

A_1	A_0	F_1	F_2
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0

(a) Truth table

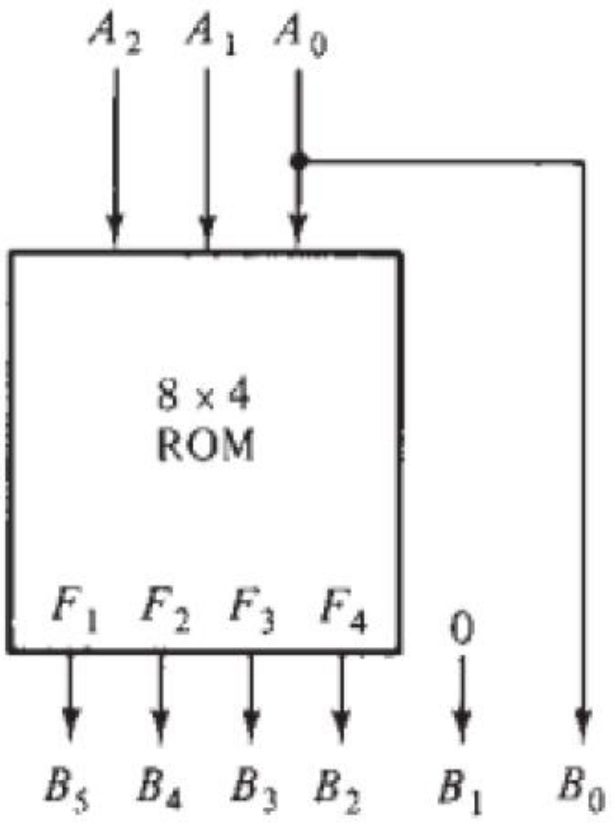
Example

Design a combinational circuit using a ROM. The circuit accepts a 3 bit number and generates an output binary number equal to the square of the input number.

TABLE 5-5
Truth Table for Circuit of Example 5-3

Inputs			Outputs						
A_2	A_1	A_0	B_5	B_4	B_3	B_2	B_1	B_0	Decimal
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	1
0	1	0	0	0	0	1	0	0	4
0	1	1	0	0	1	0	0	1	9
1	0	0	0	1	0	0	0	0	16
1	0	1	0	1	1	0	0	1	25
1	1	0	1	0	0	1	0	0	36
1	1	1	1	1	0	0	0	1	49

Example (Contd.)



(a) Block diagram

A_2	A_1	A_0	F_1	F_2	F_3	F_4
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	1
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	1
1	1	1	1	1	0	0

(b) ROM truth table

The three inputs specify eight words of 4-bits each.

Types of ROMs

ROMs may be programmed in two ways:

-mask programming

-programmable read only memory (PROM)

Mask Programming:

- Done by the manufacturer during the last fabrication process of the unit.
- The manufacturer makes the mask for the paths to produce 1’s or 0’s according to the customers truth table.
- It is economical only if large quantities of the same ROM configuration are to be manufactured.

Types of ROMs (Contd.)

PROM (Programmable Read-Only Memory):

- Economical for small quantities of ROMs
- The links in PROM are broken as per application
- User can use his own laboratory to achieve the desired relationship between input address and stored words
- Special units called **PROM programmers** are available commercially to facilitate the procedure
- **PROM is a hardware procedure.**
- So, **hardware procedure for ROM or PROM is irreversible**
- Once programmed, permanent fixed pattern and cannot be altered
- Unit must be discarded if the bit pattern is to be changed

Types of ROMs (Contd.)

ERASABLE PROM (EPROM):

- Can be restructured to the initial value even though it has been changed previously
- When placed under a special ultraviolet light for a given period time, the short wave radiation discharges the internal gates that serve as constant
- After erasure, the ROM returns to its initial state and can be reprogrammed
- Some ROMs can be erased with electrical signal instead of ultraviolet light which are called **Electrically Alterable ROMs (EAROMs)**

Why it is called Read-Only Memory?

- ***Memory*** Designates a storage unit
- ***Read*** signifies the contents of a word specified by an address in a storage unit which is placed at the output terminals
- So, a ***memory unit*** with a fixed word pattern that can be ***read out*** upon application of a given address
- The bit pattern in the ROM is permanent and cannot be changed during **normal operation**

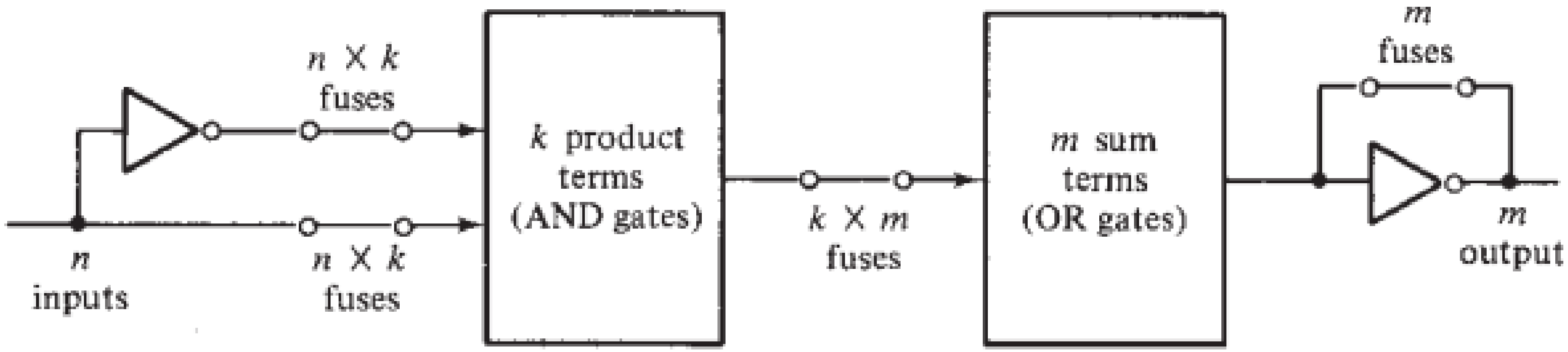
Usage of ROM

- To implement complex combinational circuits from truth tables
- Converting from one binary code to another (e.g., ASCII to EBCDIC and vice versa)
- For arithmetic functions such as multipliers
- For displaying characters in a cathode-ray tube
- For applications which require a large number of inputs and outputs
- In the design of control units of digital systems
- A control unit that utilizes a ROM to store binary control information is called a **microprogrammed control unit**

Programmable Logic Array (PLA)

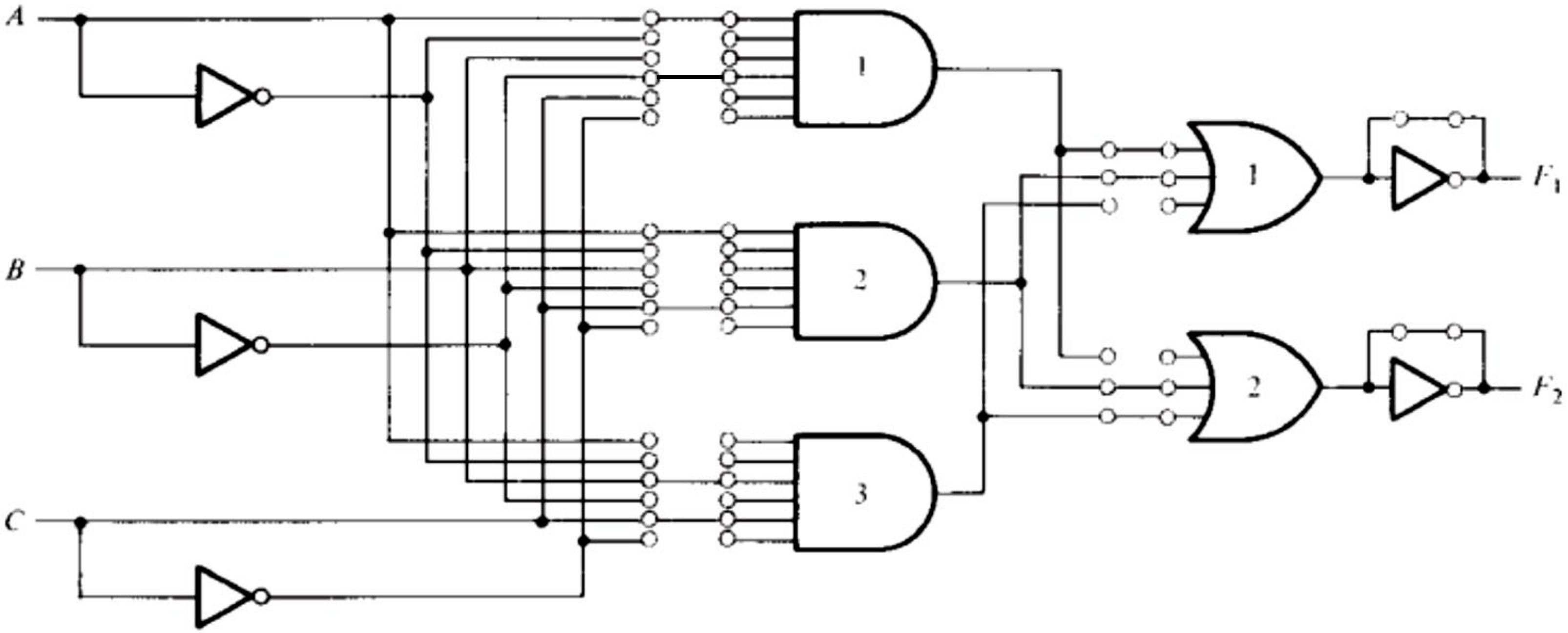
- **Programmable Logic Array(PLA) is a fixed architecture logic device with programmable AND gates followed by programmable OR gates.**
- More economical to use when the **don’t care conditions are excessive**
- Similar to ROM, but does not provide full decoding i.e., **does not generate all the minterms**
- The **decoder is replaced by a group of AND gates**, where each can be programmed to generate a product term of the input variables
- The **AND and OR gates** inside the PLA has **links** among them
- Functions are implemented in sum of products form by opening appropriate links and leaving the desired connections

Programmable Logic Array (Contd.)



- **n inputs, m outputs, k product terms, m sum terms**
- Product terms constitute a group of k AND gates
- Sum terms constitute a group of m OR gates
- **The size of a PLA = n X m X k**
- A typical PLA contains 16 inputs, 48 product terms and 8 outputs
- **Number of programmed links = 2n X k + k X m + m**, whereas that of a **ROM is 2^n X m**

An example of PLA



Inputs, n = 3
Product terms, k = 3
Outputs, m = 2
So, PLA size= ?

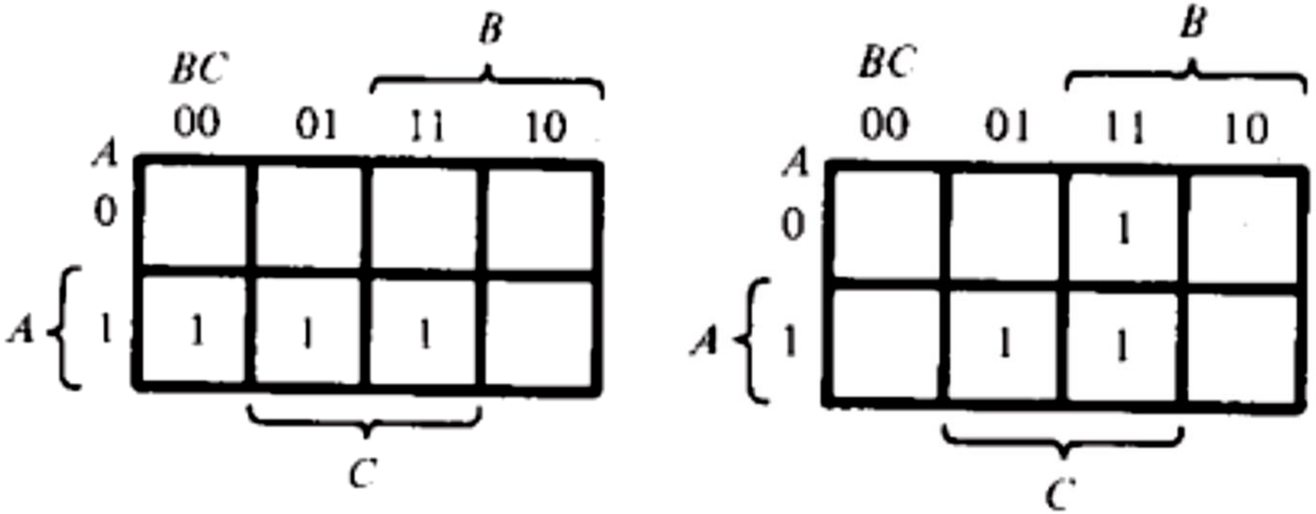
- As with ROM, PLA can be mask programmable and field programmable
- With a mask programmable PLA, customer must submit a PLA program table to the manufacturer
- Field programmable PLA is called FPLA and it is like as PROM

PLA Implementation Example

Consider the following combinational circuit as a truth table:

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

(a) Truth table



$F_1 = AB' + AC$

$F_2 = AC + BC$

(b) Map Simplification

PLA Implementation Example (Contd.)

$F_1 = AB' + AC$

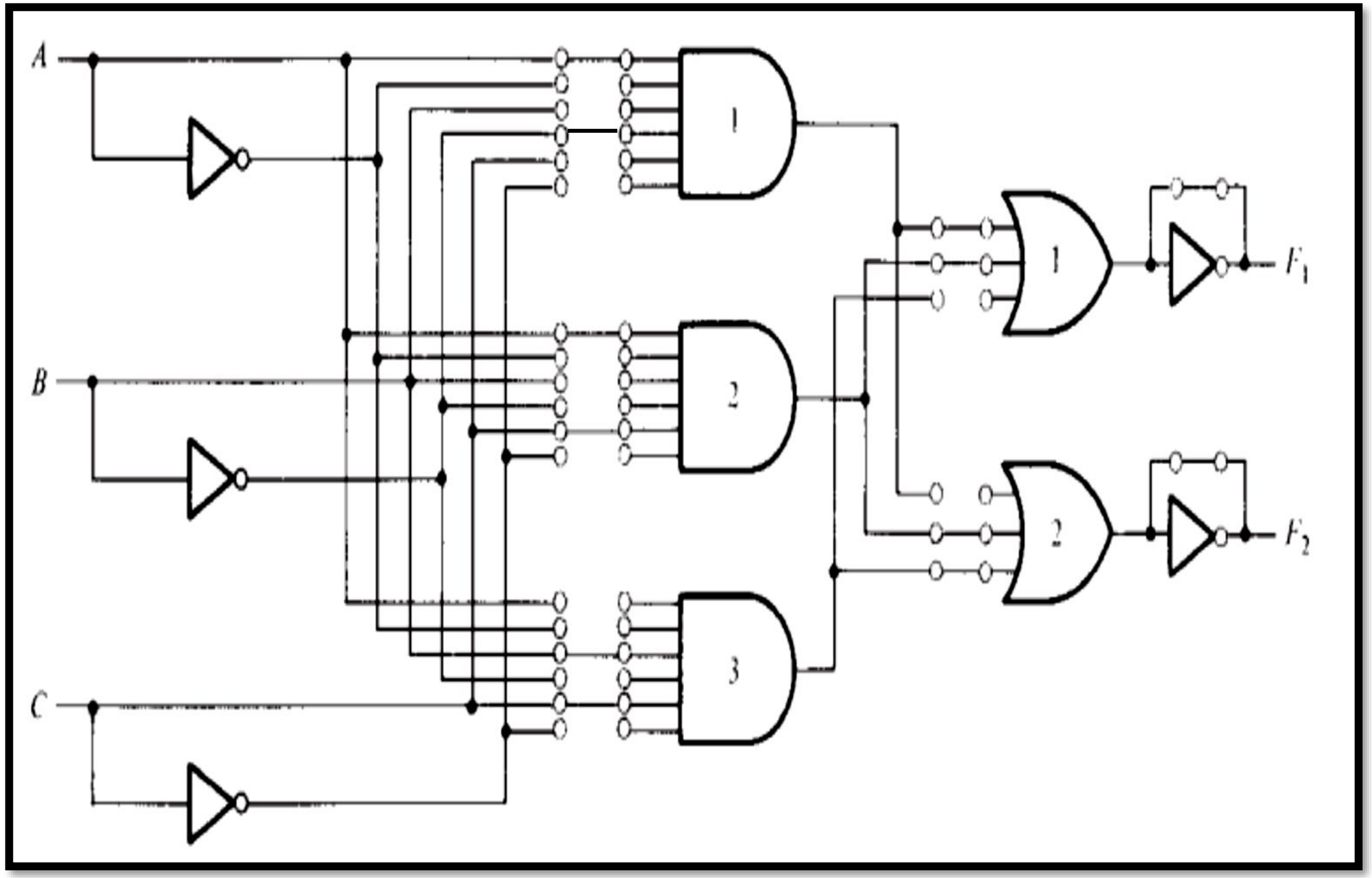
$F_2 = AC + BC$

T: if output inverter is to be bypassed
C: For complement

	Product term	Inputs A	B	C	Outputs F_1	F_2	
AB'	1	1	0	–	1	–	
AC	2	1	–	1	1	1	
BC	3	–	1	1	–	1	
					T	T	T/C

(c) PLA program table

PLA Implementation Example (Contd.)



(d)PLA Diagram

	Product term	Inputs A	Inputs B	Inputs C	Outputs F_1	Outputs F_2	
AB'	1	1	0	–	1	–	
AC	2	1	–	1	1	1	
BC	3	–	1	1	–	1	
					T	T	T/C

Designing a Digital System with PLA

- Reduce the number of distinct product terms
- The number of literals in a product is not important since we have all input variables
- Both the truth value and the complement value should be simplified
- See which can be expressed with fewer product terms
- And which one provides product terms that are common to other functions

PLA Implementation Example - 2

A combinational circuit is defined by the functions:

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

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

Implement the circuit with a PLA having three inputs, four product terms and two outputs.

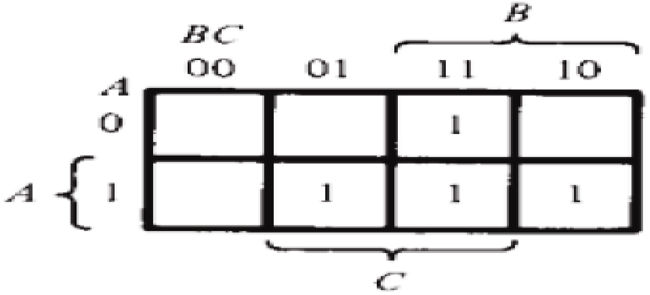
PLA Implementation Example – 2 (Contd.)

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

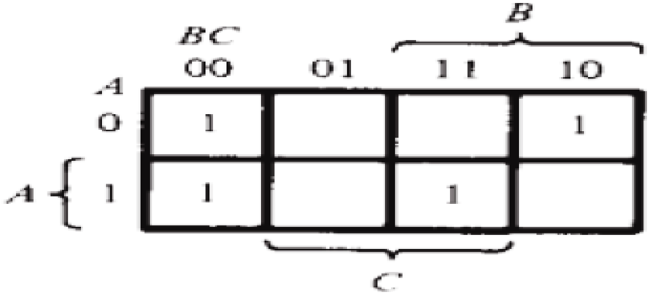
$$F_2(A, B, C) = \Sigma(0, 2, 4, 7)$$

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

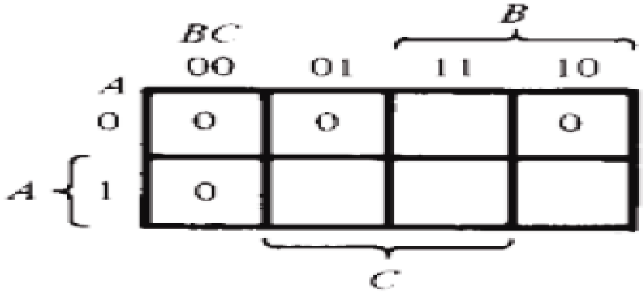
$$F_2 = B'C' + A'C' + ABC$$



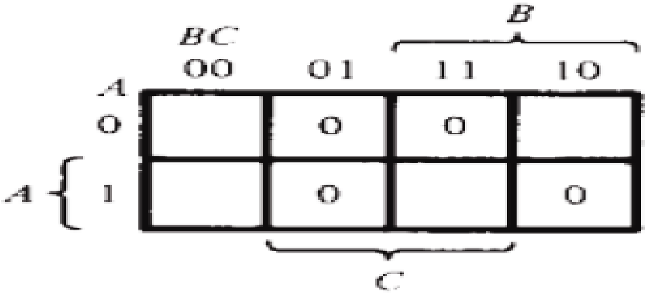
$F_1 = AC + AB + BC$



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



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



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

PLA Implementation Example – 2 (Contd.)

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

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

PLA program table

	Product term	Inputs			Outputs	
		A	B	C	F_1	F_2
$B'C'$	1	–	0	0	1	1
$A'C'$	2	0	–	0	1	1
$A'B'$	3	0	0	–	1	–
ABC	4	1	1	1	–	1
				T/C	C	T

***Draw the PLA Circuit Diagram.

Applications of PLA

- PLA is used to provide control over datapath.
- PLA is used as a counter.
- PLA is used as a decoders.
- PLA is used as a BUS interface in programmed I/O.
- It defines various states in an instruction set, and produces the next state (by conditional branching)