Digital Logic Design
ENEE 244-010x

Lecture 16

Announcements

- Homework 7 due today
- Review sheet and solutions for second midterm are posted (on course webpage and Canvas, respectively)

Agenda

- Last time:
  - Logic Design with Decoders and Multiplexers (5.4, 5.6)
- This time:
  - Programmable Logic Devices (5.7)
  - Programmable Read-Only Memories (PROM) (5.8)
  - Programmable Logic Arrays (PLAs) (5.9)
  - Programmable Array Logic (PAL) (5.10)

General Structure of PLD

- Inputs to the PLD are applied to a set of buffer/inverters. These devices have both the true value of the input as well as the complemented value of the input as its outputs.
- Outputs from these devices are the inputs to an array of and-gates. The AND array generates a set of p product terms.
- The product terms are inputs to an array of or-gates to realize a set of m sum-of-product expressions.

General Structure of PLD

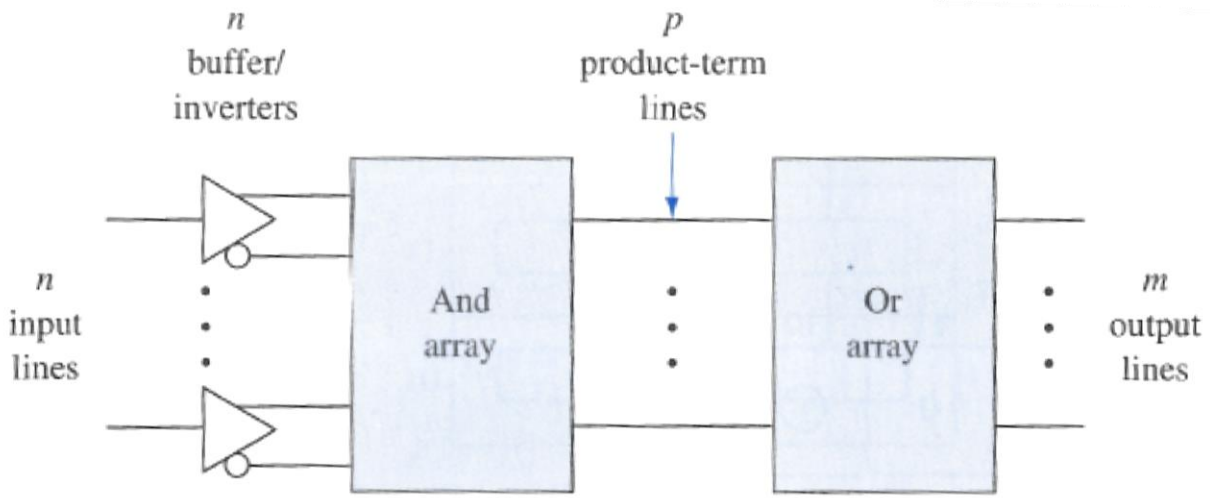


Figure 5.48 General structure of PLDs.

General Structure of PLD

- One or both of the gate arrays are programmable.
- The logic designer can specify the connections within an array.
- PLDs serve as general circuits for the realization of a set of Boolean functions.

Device	AND-array	OR-array
PROM	Fixed	Programmable
PLA	Programmable	Programmable
PAL	Programmable	Fixed

Programming a PLD

- In a programmable array, the connections to each gate can be modified.
- Simple approach is to have each of the gate inputs connected to a fuse.

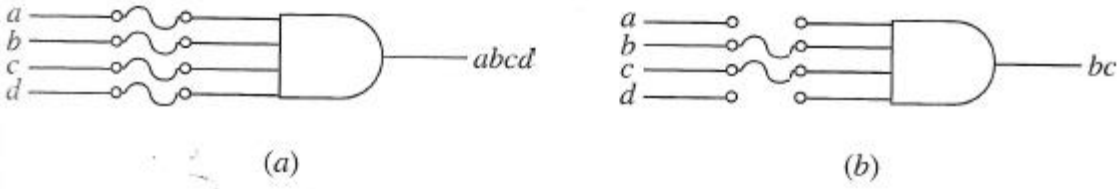


Figure 5.50 Programming by blowing fuses. (*a*) Before programming. (*b*) After programming.

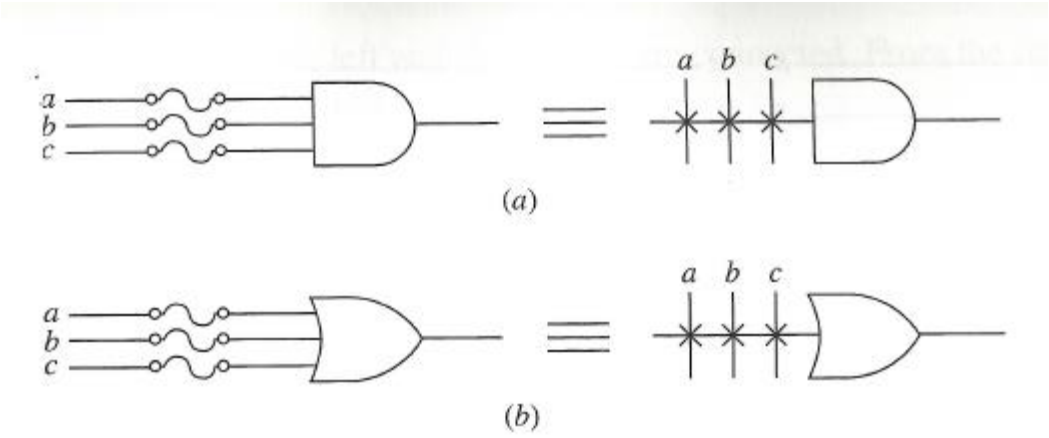
- Gate realizes the product term $abcd$.
- To generate the product term bc we remove the a, d connections by blowing the corresponding fuses.
- Thus, programming is a hardware procedure. Specialized equipment called programmers is needed to carry out the programming of a PLD.

Programming a PLD

- Erasable PLD—connections can be reset to their original conditions and then reprogrammed.
  - Can be achieved by exposing the PLD to ultraviolet light or using electrical signals
- PLDs programmed by a user are called field programmable.
- User can also specify the desired connections and supply the information to the manufacturer.
- Such PLDs are called mask programmable.

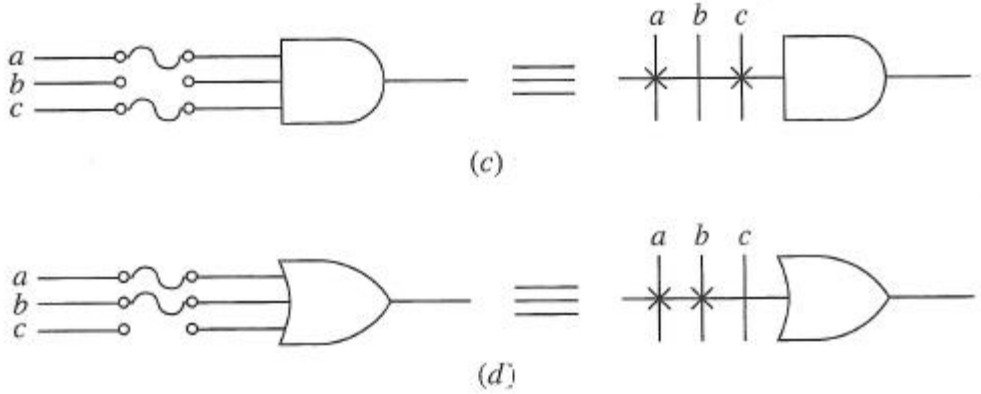
PLD Notation

- Simplified notation. Each gate has only a single input line.
- Inputs are indicated by lines at right angles to the single gate lines.
- A cross at the intersection denotes a fusible link is intact.



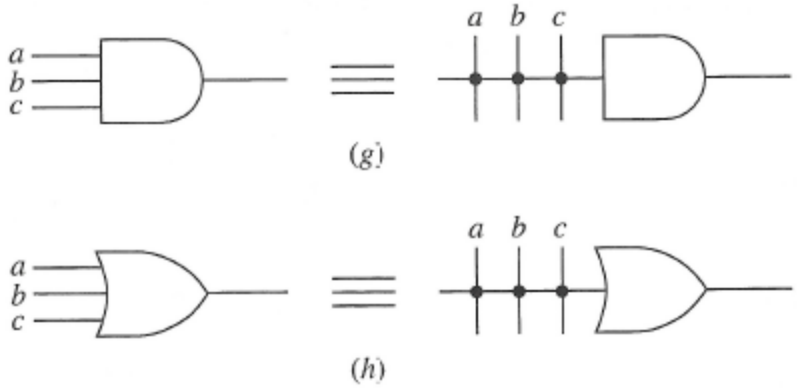
PLD Notation

- Lack of cross indicates the fuse is blown or no connection exists.

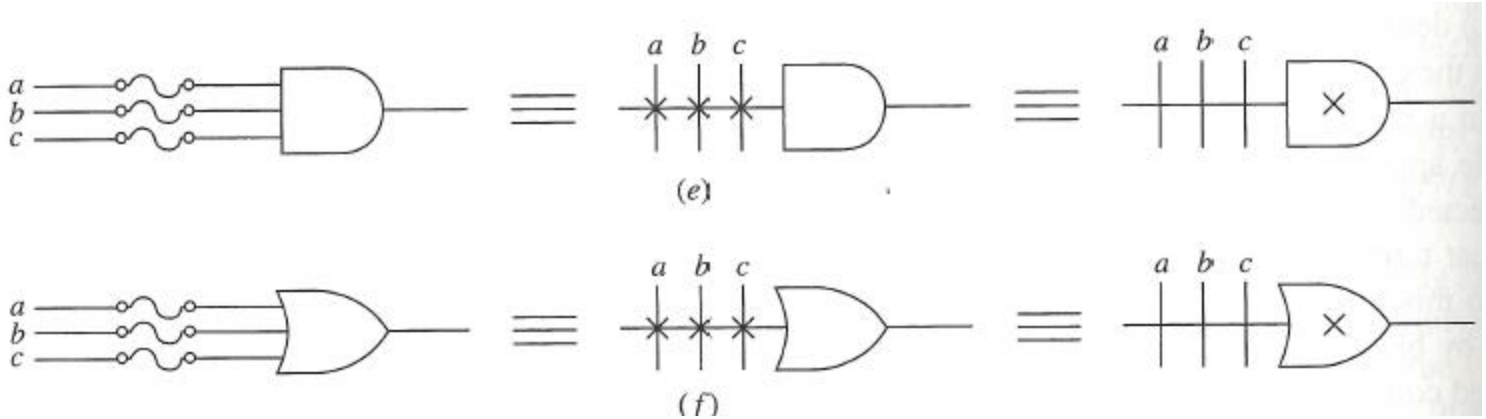


PLD Notation

- The occurrence of a hard-wired connection that is not fusible is indicated by a junction dot.



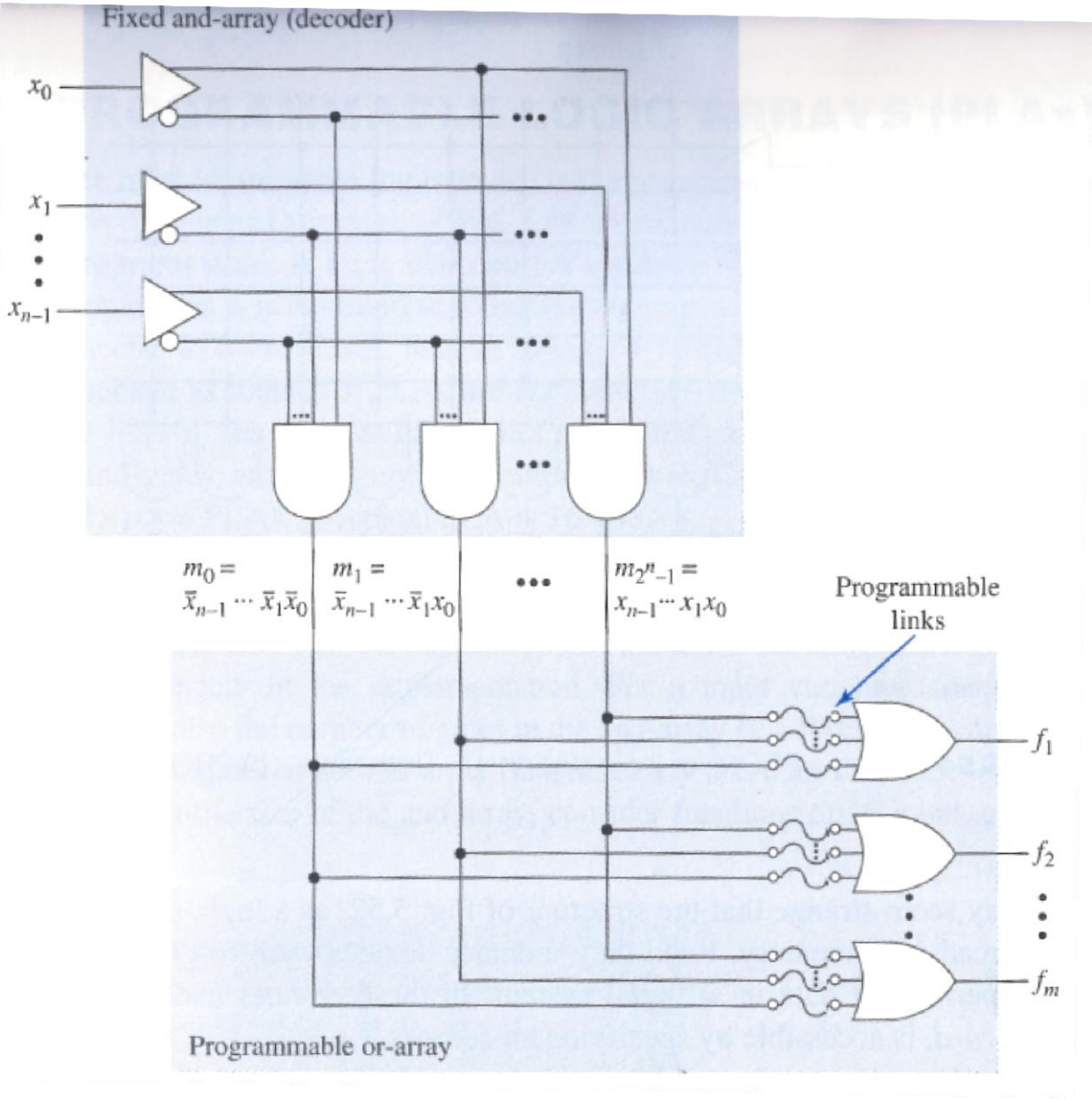
- For the special case when all the input fuses to a gate are kept intact, a cross is placed inside the gate symbol.



Programmable Read-Only Memory (PROM)

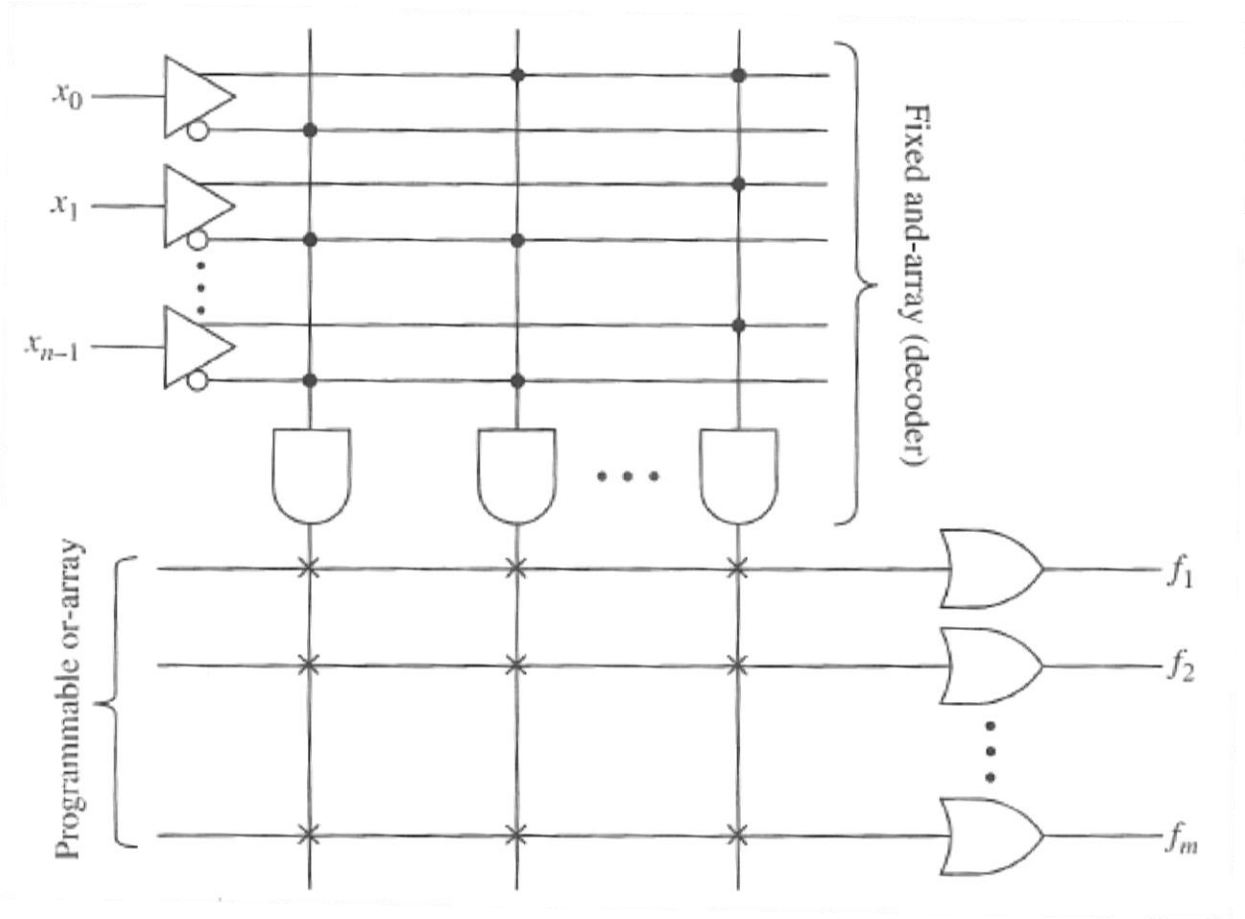
- AND-array with buffer/inverter is an n-to-2^n-line decoder.
- OR-array is a collection of programmable or-gates.
- Decoder is a min-term generator.
- n-variable minterms appear on the 2^n lines at the decoder output. These are also known as word lines.
- n input lines called address lines, m output lines called bit lines.
- $2^n \times m$ PROM.
- Realization of Boolean expressions same as realization using decoder discussed previously.

PROM Structure



Logic Diagram

PROM Structure



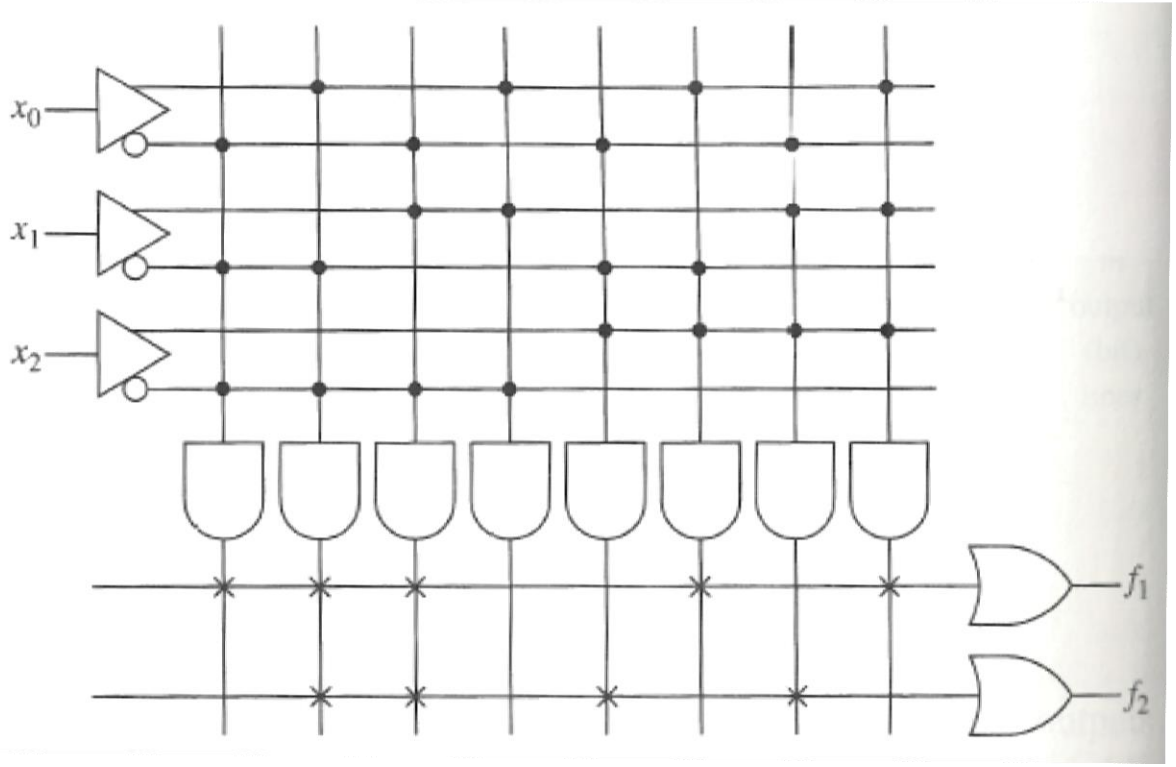
PLD Notation

Example

Realizing $f_1(x_2, x_1, x_0) = \sum m(0,1,2,5,7)$

$f_2(x_2, x_1, x_0) = \sum m(1,2,4,6)$

x_2	x_1	x_0	f_1	f_2
0	0	0	1	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	0
1	0	0	0	1
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0



Why is it called PROM?

- 3-bit input combination to the x_0, x_1, x_2 lines is regarded as an address of one of the word lines.
- As a consequence of selecting a given word line, a pattern of 0's and 1's, a word, as determined by the fusible connections to the selected word line appears at the bit lines of the device.
- This 0-1 pattern is considered the word stored at the address associated with the selected word line.
- E.g. the word stored at address $x_2x_1x_0 = 100$ is $f_1f_2 = 01$.
- "Read only": The fact that the connections associated with the fusible links normally cannot be altered once they are formed.

Programmable Logic Array

- PLAs are characterized by three numbers:
  - Number of input lines n
  - Number of product terms that can be generated p (the number of AND gates)
  - Number of output lines m
- $n \times p \times m$ PLAs
- Typical PLA is $16 \times 48 \times 8$.

Programmable Logic Array

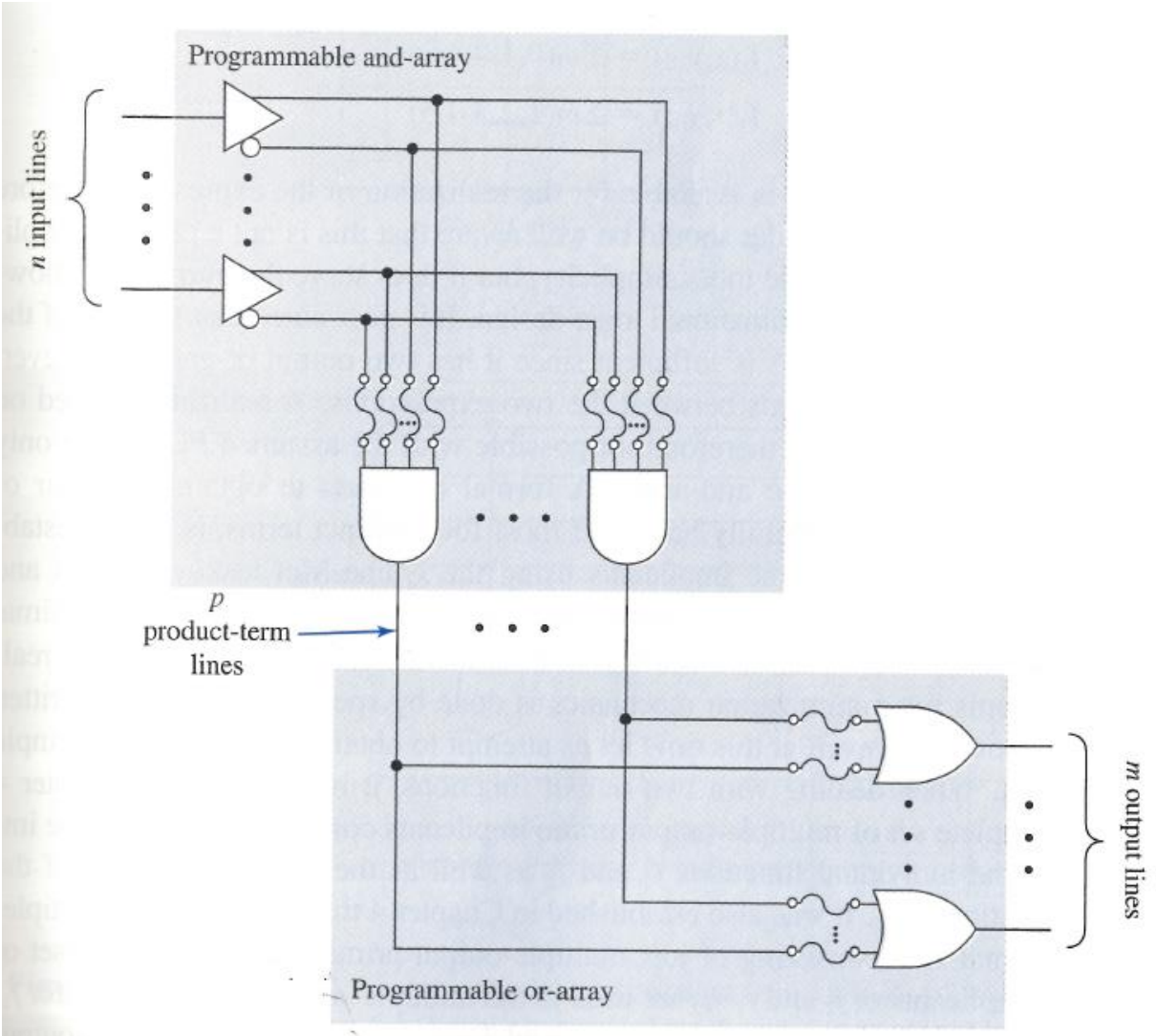


Figure 5.55 Logic diagram of an $n \times p \times m$ PLA.

Programmable Logic Array

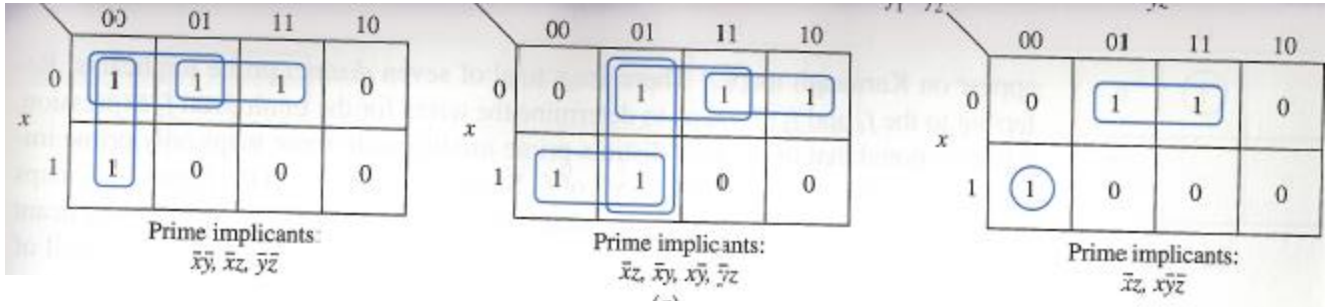
- In many logic design situations not all the minterms are needed for a realization.
- For n input variables, 2^n minterms.
- This is the number of gates in the AND-array of a PROM.
- PLA's with 16 input lines
  - $2^{16} = 65{,}536$ minterms
  - In a $16 \times 48 \times 8$ PLA only 48 product terms.
- $2n$ inputs appear at each AND gate.
- For our examples, assume $3 \times 4 \times 2$ PLA.

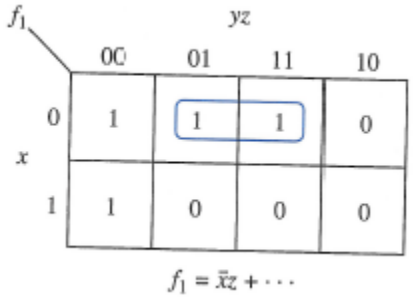
PROM vs PLA

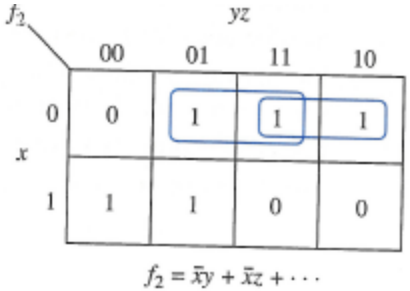
- PROM: realization of a set of Boolean functions is based on minterm canonical expressions.
  - No minimization necessary.
- PLA: the AND gates are capable of generating product terms that are not necessarily minterms.
  - Realization using PLA is based on sum-of-product expression that may not be canonical.
  - Logic designer is bounded by the number of product terms that are realizable by the AND-array.
  - Simplifications is necessary.

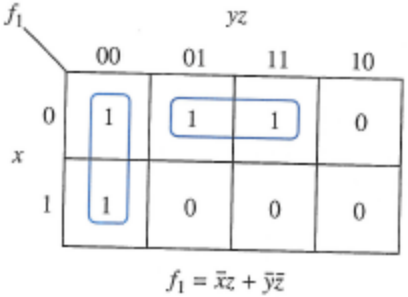
Logic Design Example

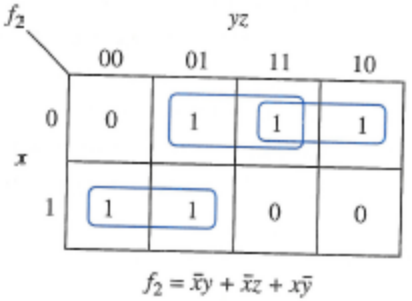
f_1 f_2 $f_1 \cdot f_2$

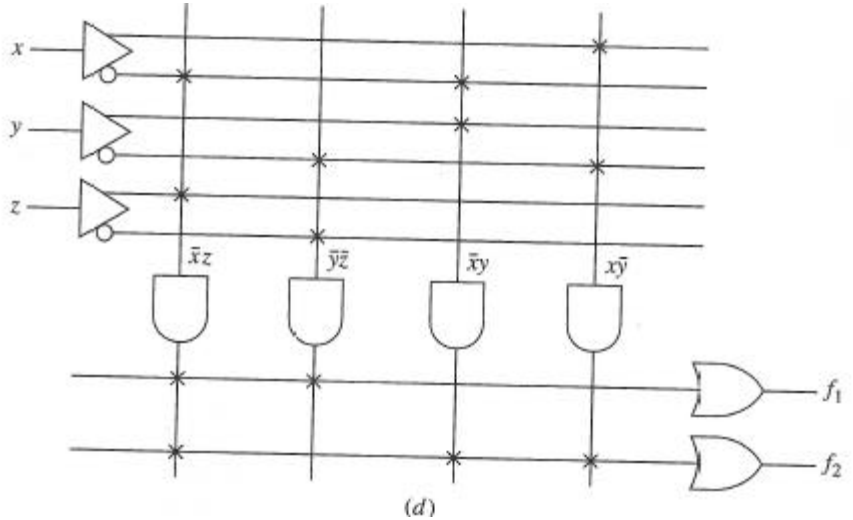






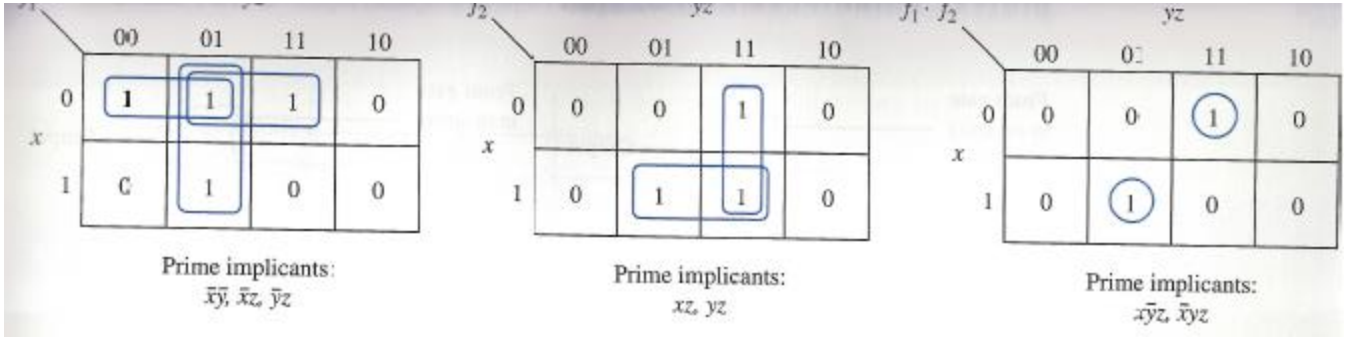


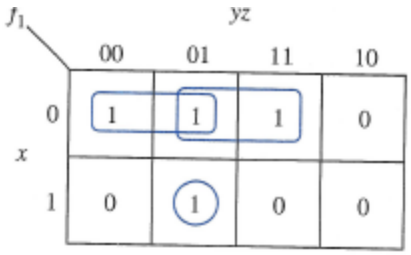




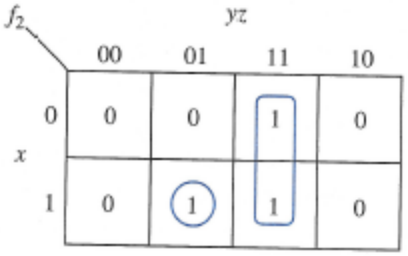
Logic Design Example

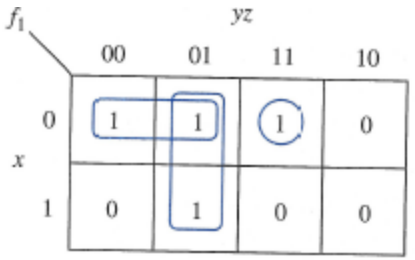
f_1 f_2 $f_1 \cdot f_2$

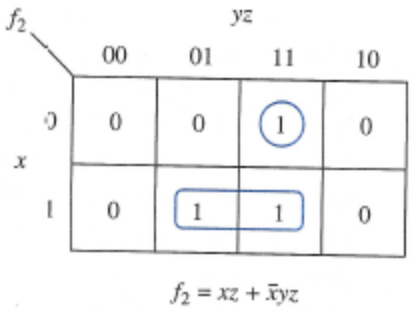


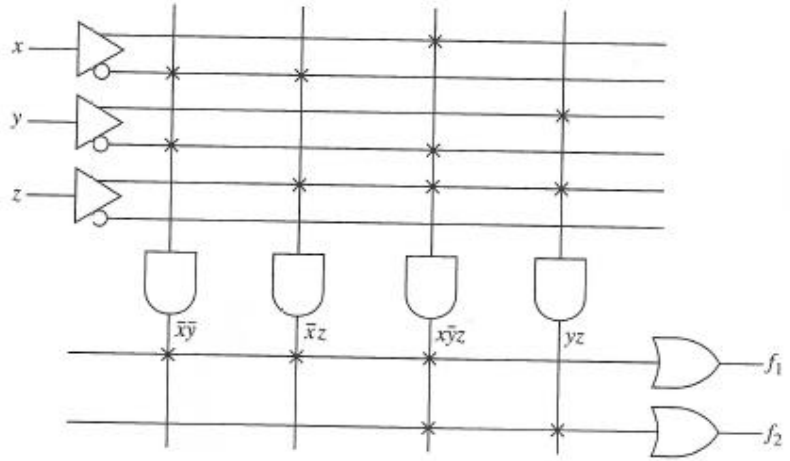


(b)









Additional Features

- For greater flexibility, PLAs make provision for either a true output or a complemented output.

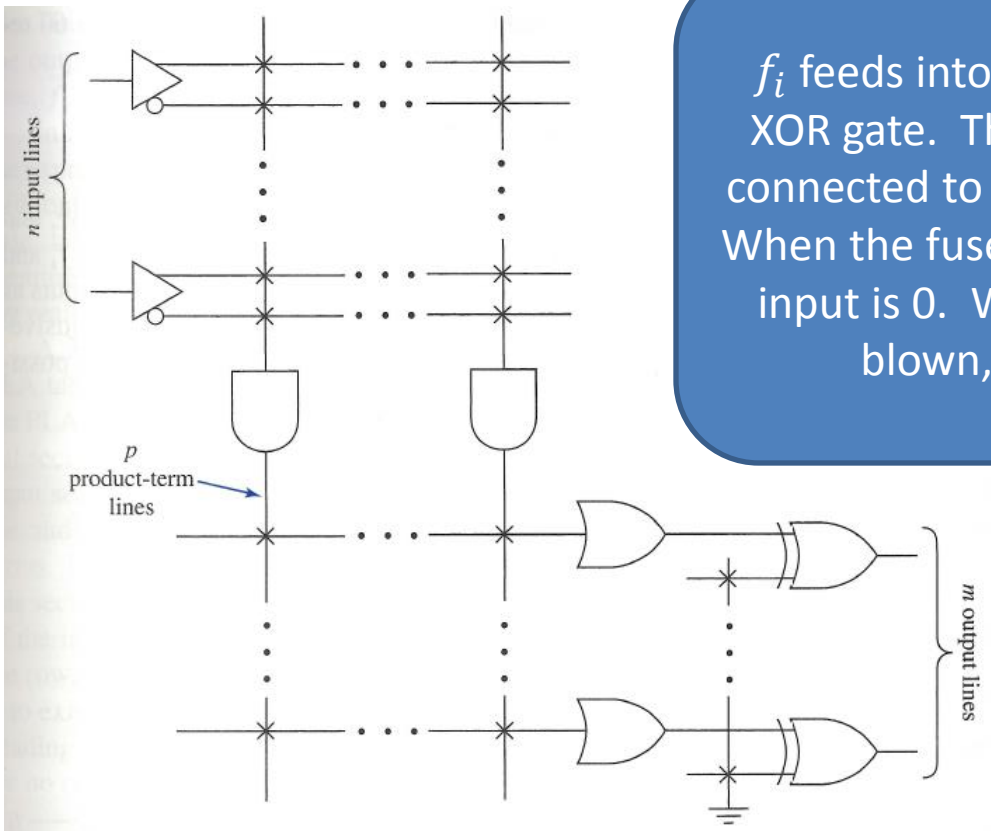


Figure 5.59 General structure of a PLA having true and complemented output capability.

f_i feeds into one input of an XOR gate. The other input is connected to a pull-up resistor. When the fuse is left intact, the input is 0. When the fuse is blown, input is 1.

Example of Use of Complemented Functions

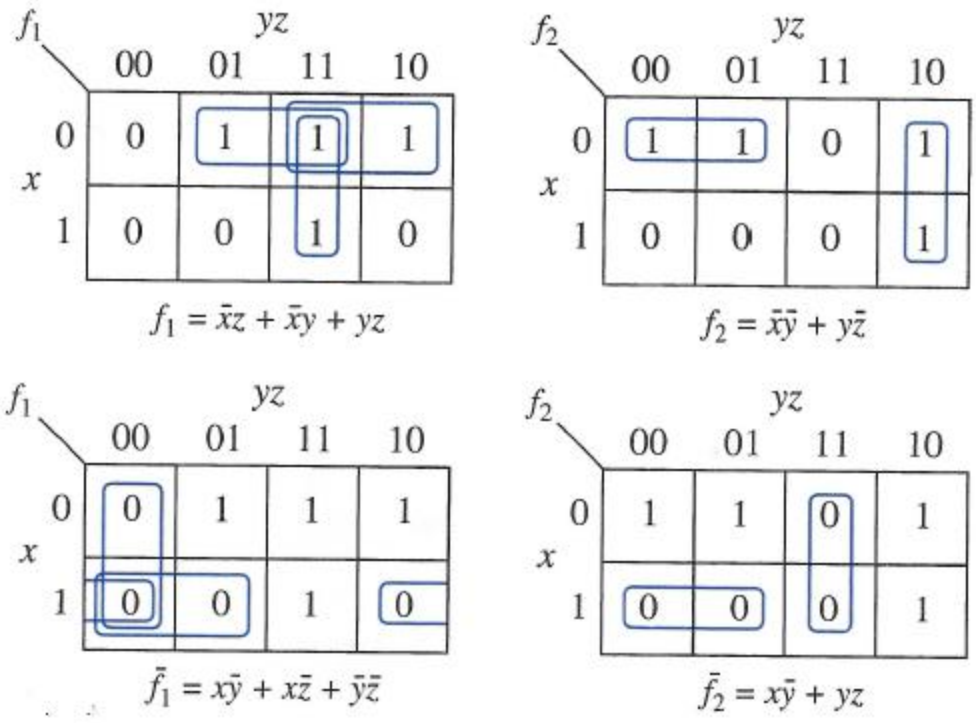


Figure 5.60 Karnaugh maps for the functions $f_1(x,y,z) = \Sigma m(1,2,3,7)$ and $f_2(x,y,z) = \Sigma m(0,1,2,6)$.

Example of Use of Complemented Functions

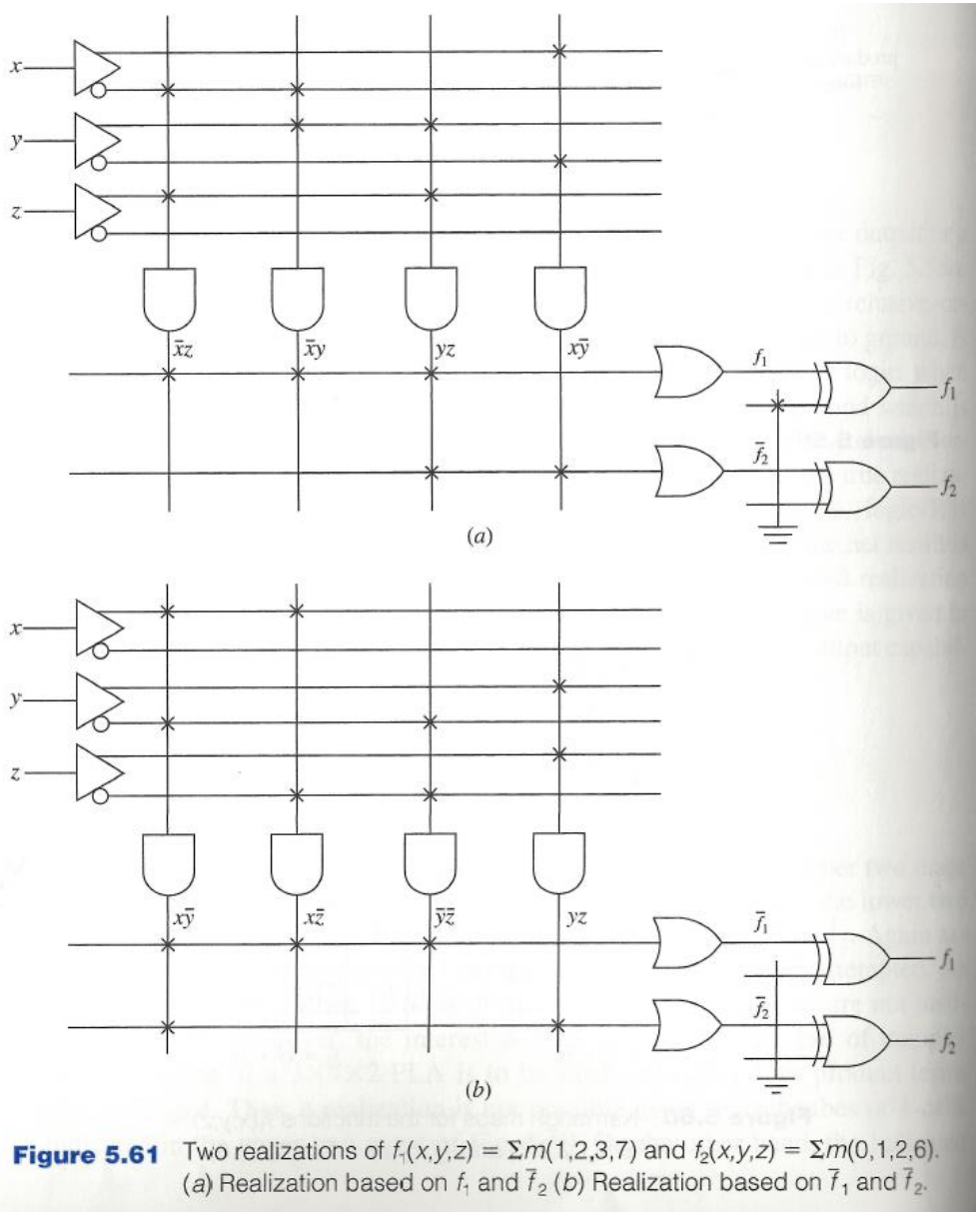


Figure 5.61 Two realizations of $f_1(x,y,z) = \Sigma m(1,2,3,7)$ and $f_2(x,y,z) = \Sigma m(0,1,2,6)$. (*a*) Realization based on f_1 and $\bar{f}_2$ (*b*) Realization based on $\bar{f}_1$ and $\bar{f}_2$.

PLA Table

- A common way of specifying the connections in a PLA.
- 3 sections: input section, output section, T/C section.
- Each product term is assigned a row in the table.
  - Input section indicates connections between inputs to AND-array.
  - Output section indicates connections between outputs of AND-array and inputs to the OR-array.
  - T/C section indicates how the exclusive or gates are programmed.
    - T—true output is used.
    - C—output should be complemented.

Product term	Inputs x	y	z	Outputs f_1	f_2
$x\bar{y}$	1	0	–	1	1
$x\bar{z}$	1	–	0	1	–
$\bar{y}\bar{z}$	–	0	0	1	–
yz	–	1	1	–	1
			T/C	*C*	*C*

Programmable Array Logic (PAL) Devices

- OR-array is fixed by the manufacturer of the device.
  - PAL device is easier to program and less expensive than the PLA.
  - Less flexible.

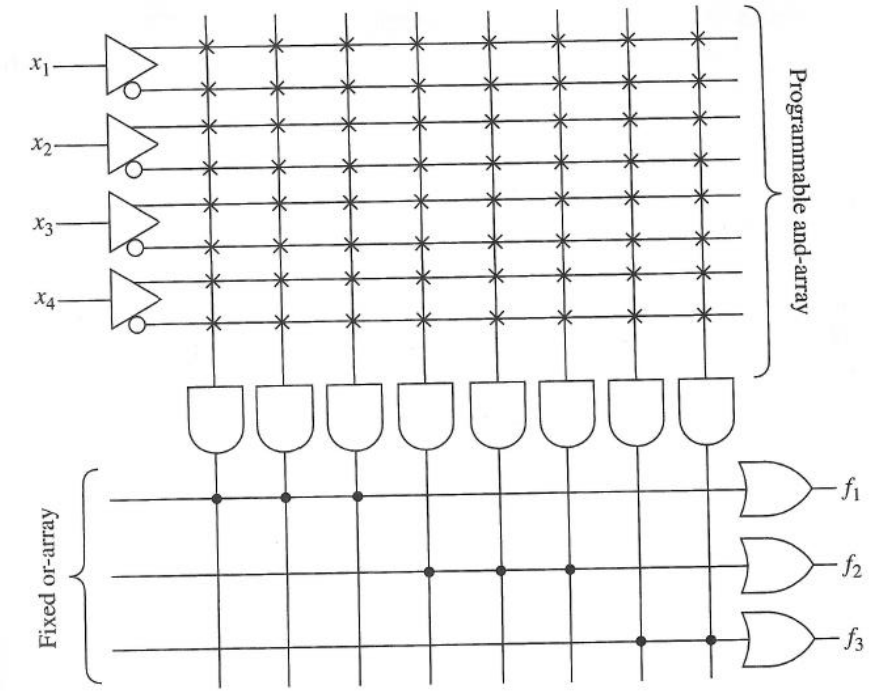


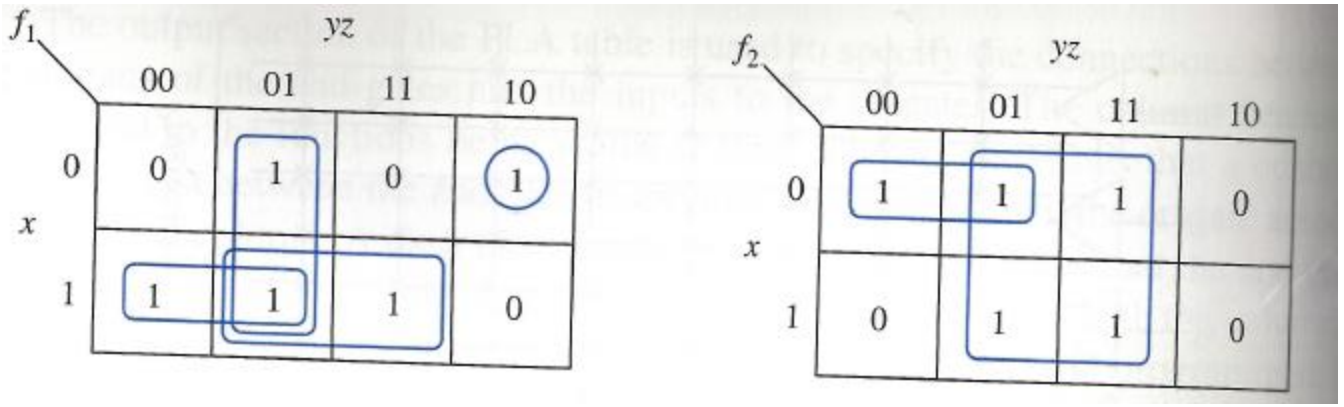
Figure 5.62 A simple four-input, three-output PAL device.

  - For our examples:
    - 4-input, 3-output PAL device
    - Three Boolean expressions can be realized in which two expressions can have at most 3 product terms and one expression can have at most 2 product terms.

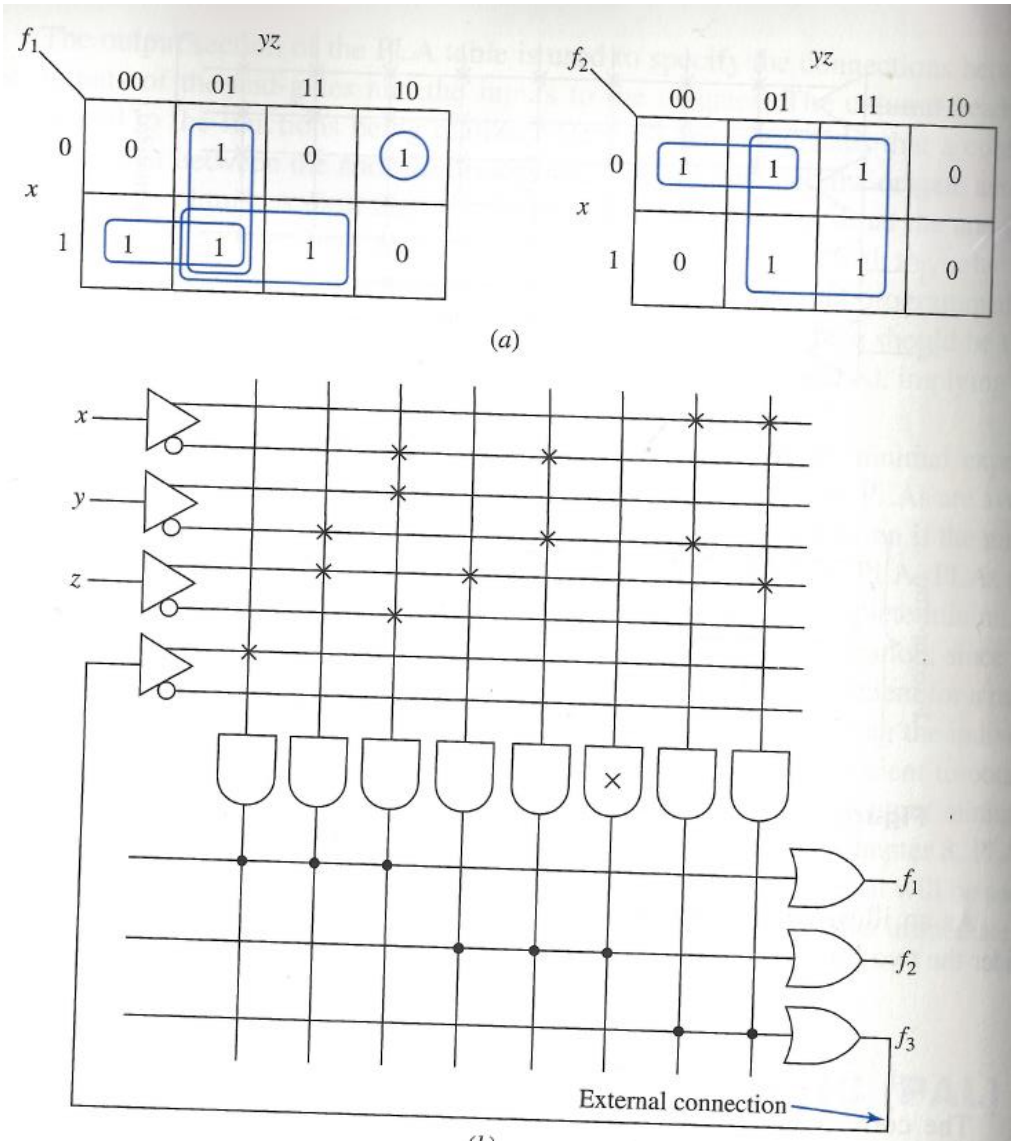
Example of Logic Design with PAL

- Consider

$$f_1(x, y, z) = \sum m(1,2,4,5,7)$$
$$f_2(x, y, z) = \sum m(0,1,3,5,7)$$



- Minimal sums:

$$f_1(x, y, z) = x\overline{y} + xz + \overline{y}z + \overline{x}y\overline{z}$$
$$f_2(x, y, z) = z + \overline{x}\,\overline{y}$$

Example of Logic Design with PAL