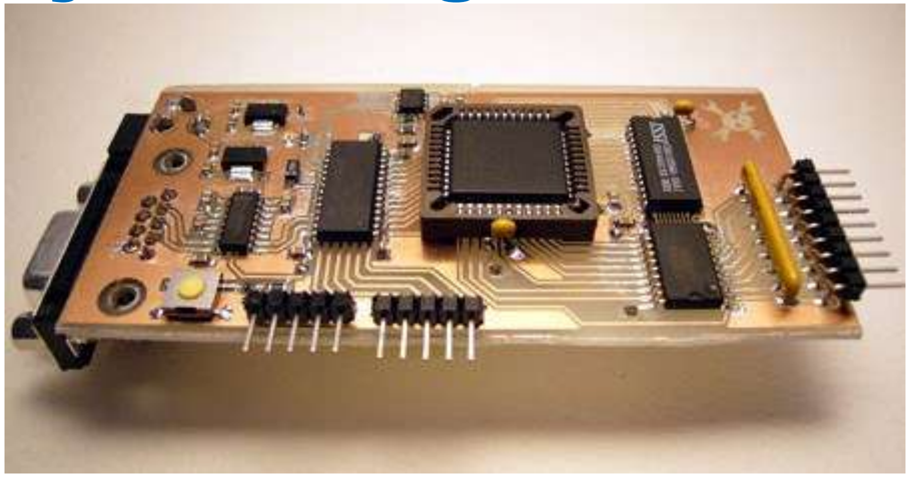
EEE 211/ETE 211
Digital Logic Design

Memory and Programmable Logic



Dr. Mohammad Monirujjaman Khan
Assistant Professor

Dept. of Electrical Engineering and Computer Science
North South University

Read-Only Memory

- ***A read-only memory (ROM) is essentially a memory device in which permanent binary information is stored.***

- ***The binary information must be specified by the designer and is then embedded in the unit to form the required interconnection pattern.***

- ***Once the pattern s established, it stays within the unit even when power is turned off and on again.***

Read-Only Memory

- ***A block diagram of a ROM is shown below. It consists of k address inputs and n data outputs.***
- ***The number of words in a ROM is determined from the fact that k address input lines are needed to specify 2^k words.***

•The inputs provide the address for memory, and the outputs give the data bits of the stored word that is selected by the address.

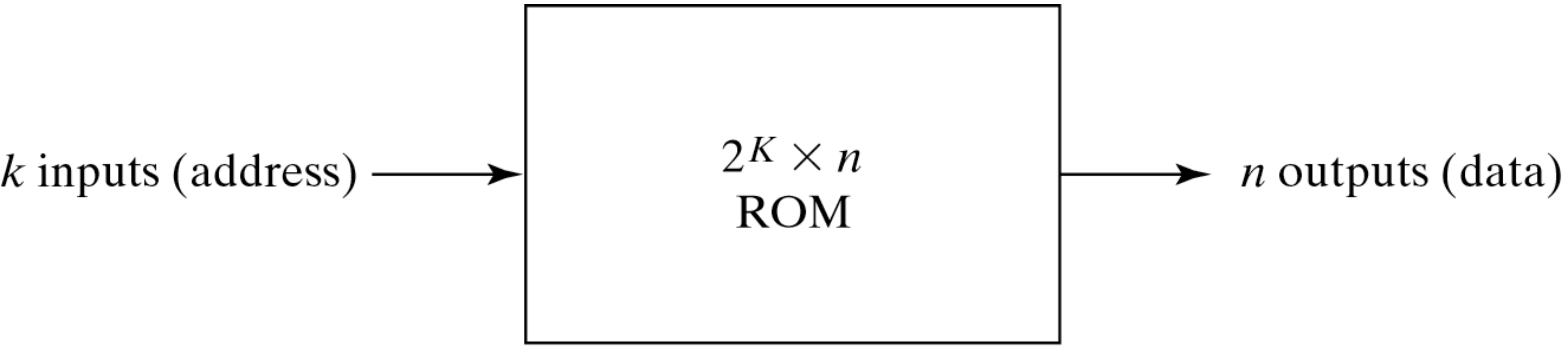


Fig. 7-9 ROM Block Diagram

Construction of ROM

- *Each output of the decoder represents a memory address.*
- *Each OR gate must be considered as having 32 inputs.*
- *A 2^k X n ROM will have an internal k X 2^k decoder and n OR gates*.

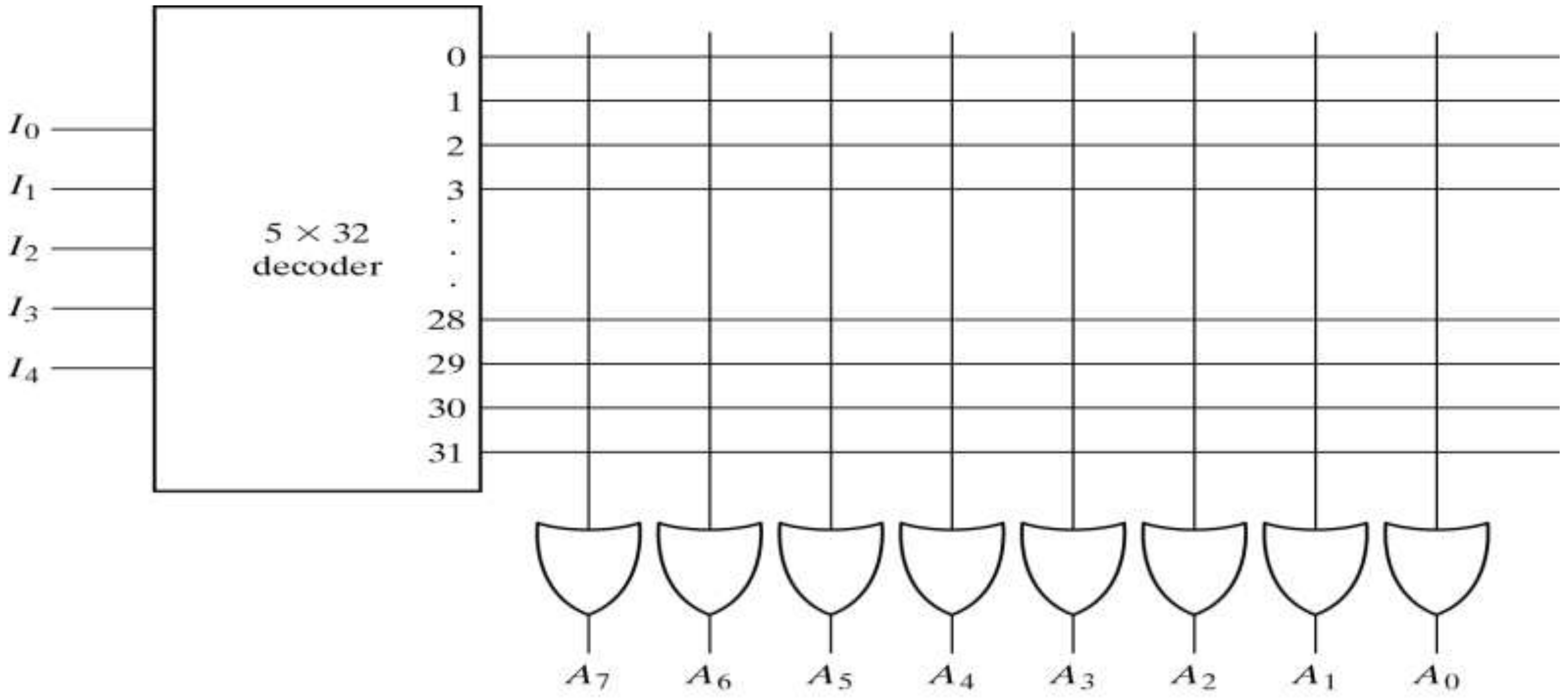


Fig. 7-10 Internal Logic of a 32 × 8 ROM

Truth table of ROM

- *A programmable connection between to lines is logically equivalent to a switch that can be altered to either be close or open.*

- *Intersection between two lines is sometimes called a cross-point.*

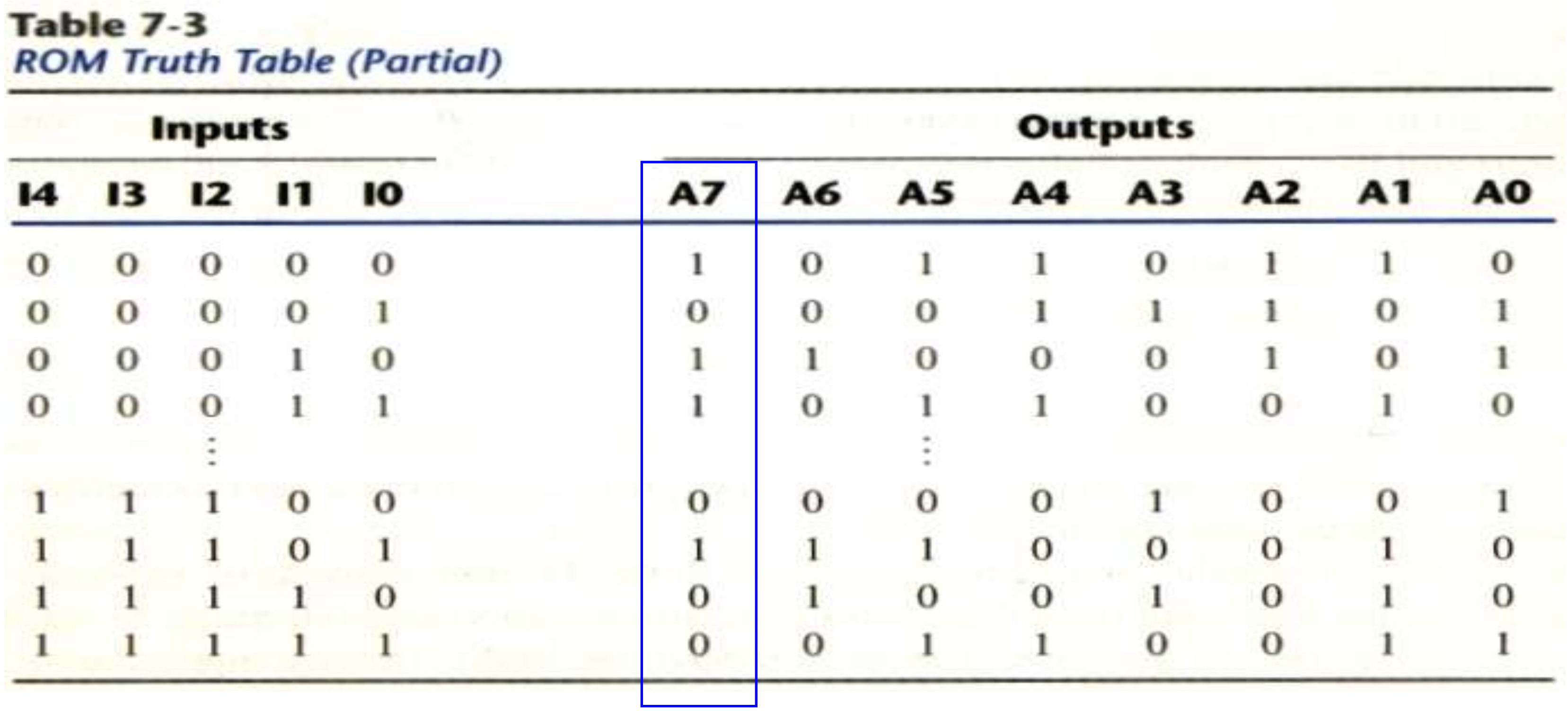


Table 7-3
ROM Truth Table (Partial)

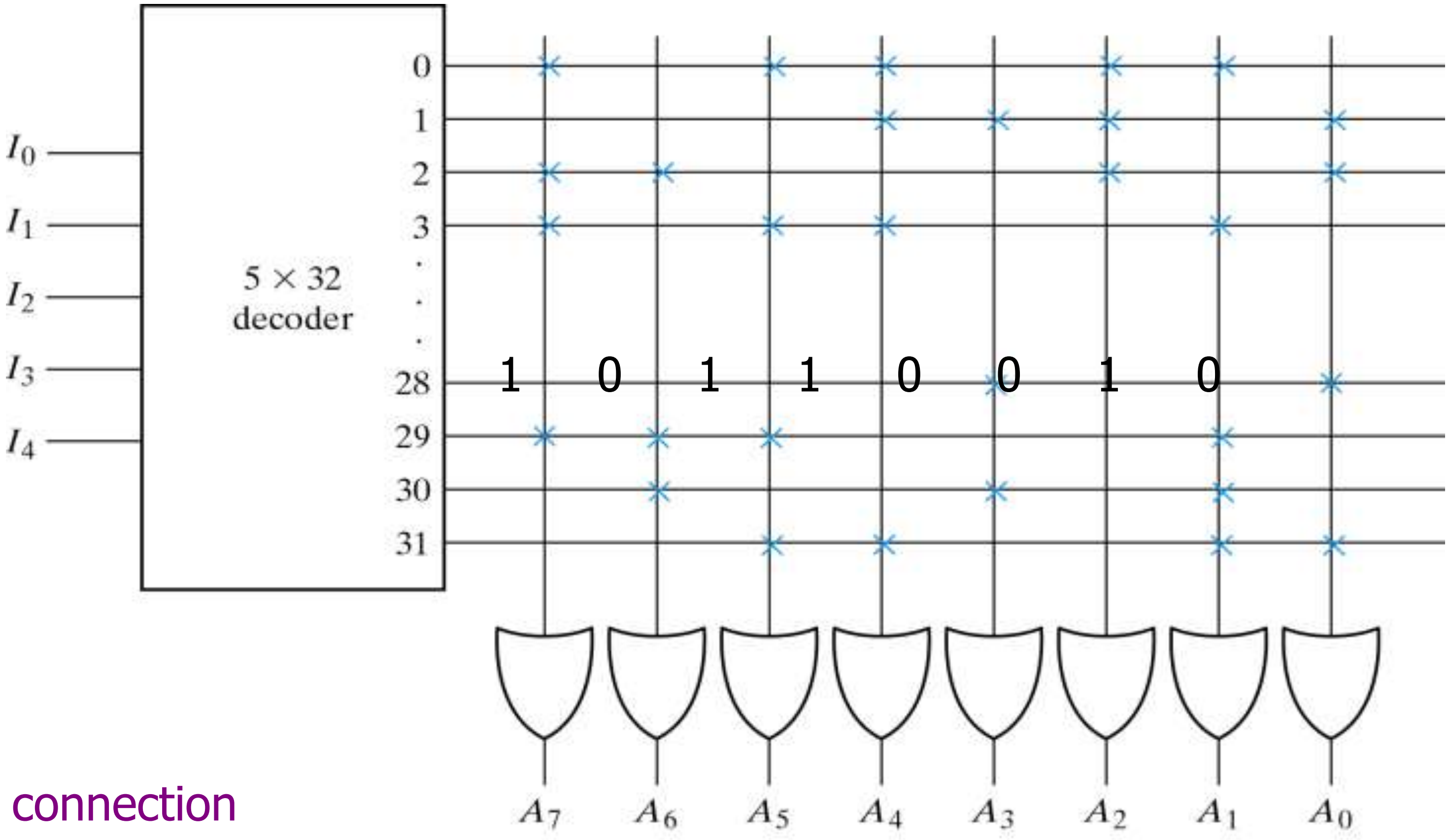
Inputs					Outputs							
I4	I3	I2	I1	I0	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	1	0	1	1	0	1	1	0
0	0	0	0	1	0	0	0	1	1	1	0	1
0	0	0	1	0	1	1	0	0	0	1	0	1
0	0	0	1	1	1	0	1	1	0	0	1	0
		⋮					⋮					
1	1	1	0	0	0	0	0	0	1	0	0	1
1	1	1	0	1	1	1	1	0	0	0	1	0
1	1	1	1	0	0	1	0	0	1	0	1	0
1	1	1	1	1	0	0	1	1	0	0	1	1

Programming the ROM

0 → no connection

1 → connection

Address 3 = 10110010 is permanent storage using fuse link



X : means connection

Fig. 7-11 Programming the ROM According to Table 7-3

Combinational circuit implementation

- The internal operation of a ROM can be interpreted in two way:
- ***First, a memory unit that contains a fixed pattern of stored words.***
- ***Second, implements a combinational circuit.***

$$A_7(I_4, I_3, I_2, I_1, I_0) = \Sigma(0,2,3\ldots,29)$$

Sum of minterms

output A_7

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.

Derive truth table first

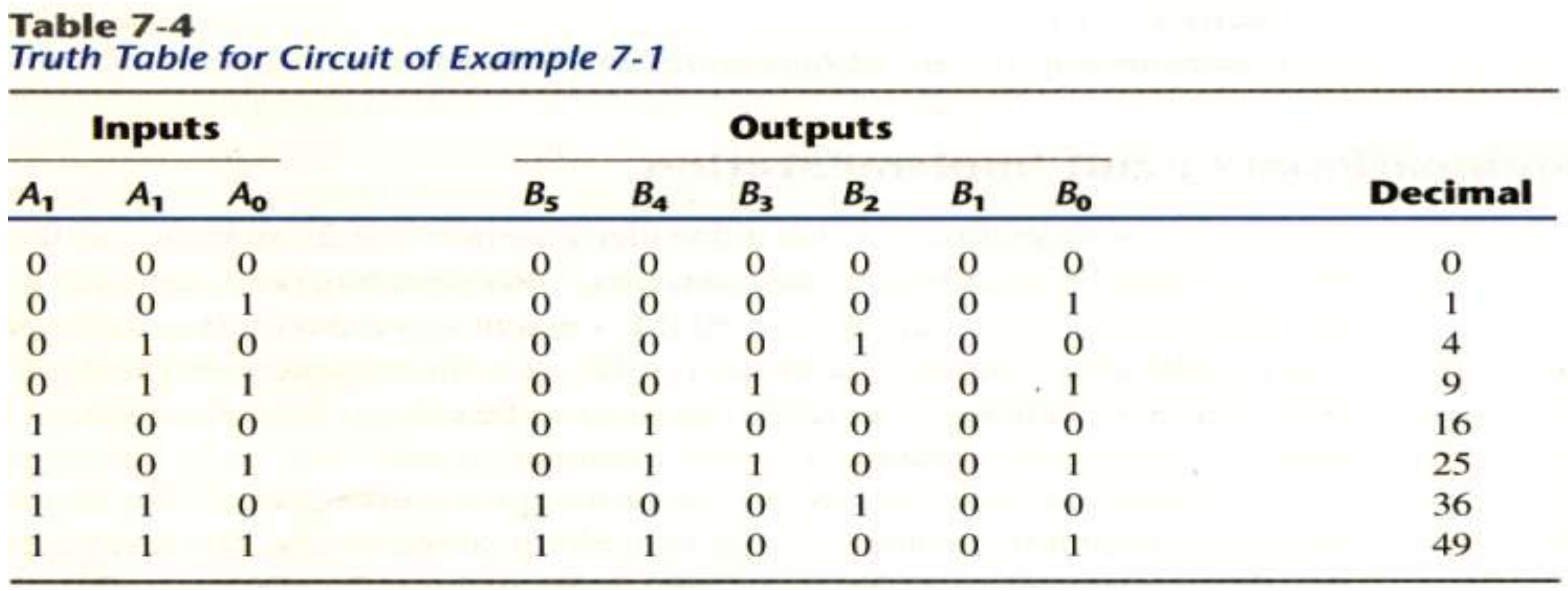
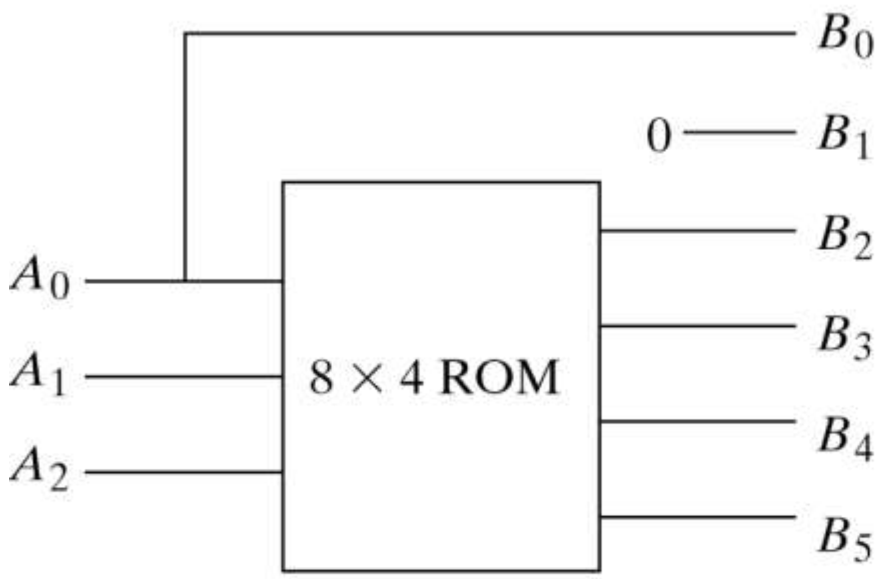


Table 7-4
Truth Table for Circuit of Example 7-1

Inputs			Outputs						
A_1	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



(a) Block diagram

A_2	A_1	A_0	B_5	B_4	B_3	B_2
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

Fig. 7-12 ROM Implementation of Example 7-1

Types of ROMs

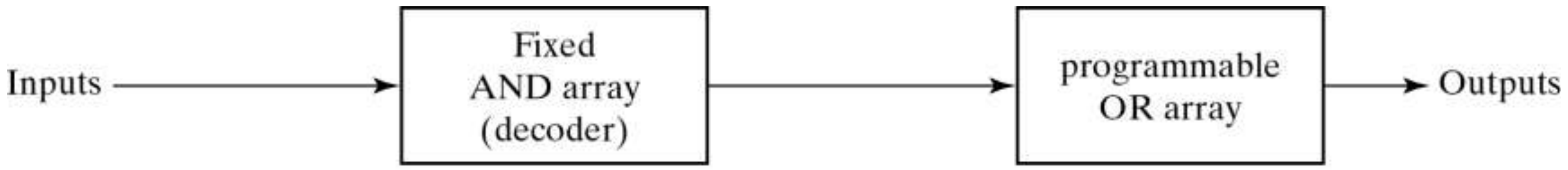
The required paths in a ROM may be programmed in four different ways.

1. Mask programming: fabrication process
2. Read-only memory or PROM: blown fuse /fuse intact
3. Erasable PROM or EPROM: placed under a special ultraviolet light for a given period of time will erase the pattern in ROM.
4. Electrically-erasable PROM(EEPROM): erased with an electrical signal instead of ultraviolet light.

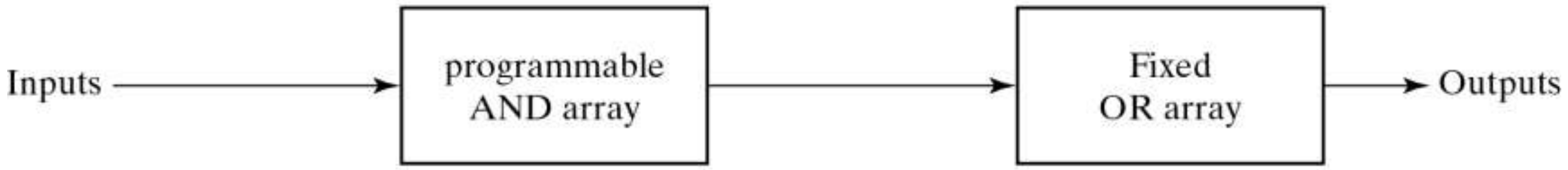
Combinational PLDs

- 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.
- PROM: fixed AND array constructed as a decoder and programmable OR array.
- PAL: programmable AND array and fixed OR array.
- PLA: both the AND and OR arrays can be programmed.

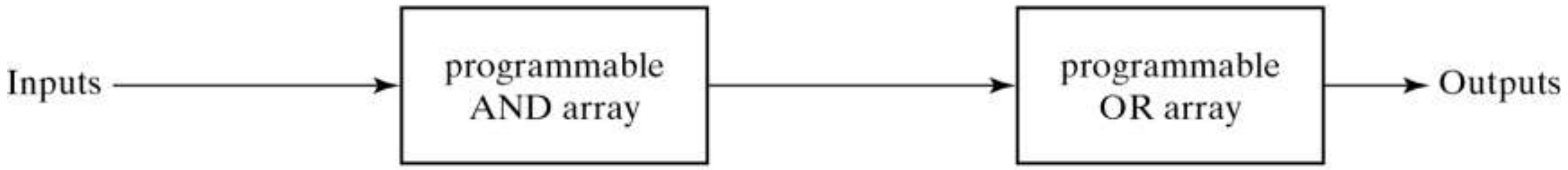
Combinational PLDs



(a) Programmable read-only memory (PROM)



(b) Programmable array logic (PAL)



(c) Programmable logic array (PLA)

Fig. 7-13 Basic Configuration of Three PLDs

Programmable Logic Array

- The decoder in PROM is replaced by an array of AND gates that can be programmed to generate any product term of the input variables.
- The product terms are then connected to OR gates to provide the sum of products for the required Boolean functions.
- The output is inverted when the XOR input is connected to 1 (since $x \oplus 1 = x'$). The output doesn't change and connect to 0 (since $x \oplus 0 = x$).

- ***The PLA is similar in concept to the PROM, except that the PLA does not provide full decoding of the variables and does not generate all the minterms.***

Programmable Logic Array (PLA)

F1 = AB'+AC+A'BC'

F2 = (AC+BC)'

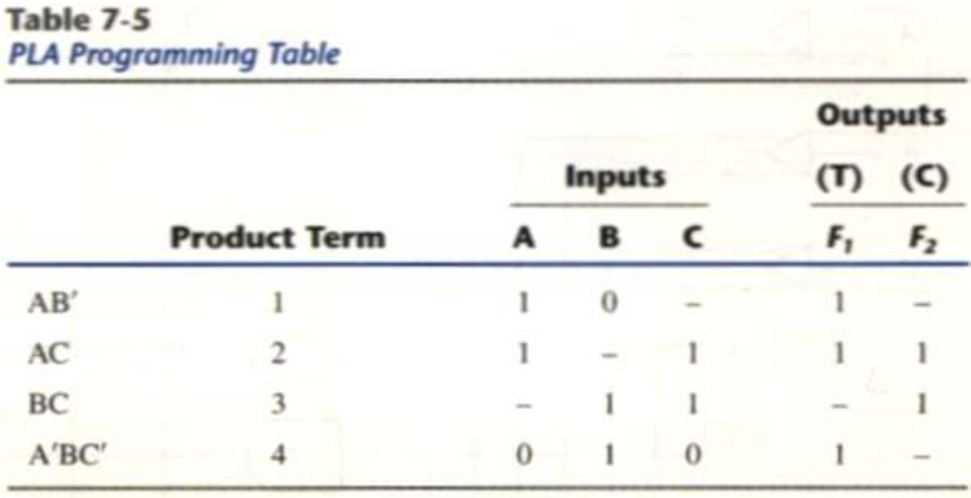


Table 7-5
PLA Programming Table

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

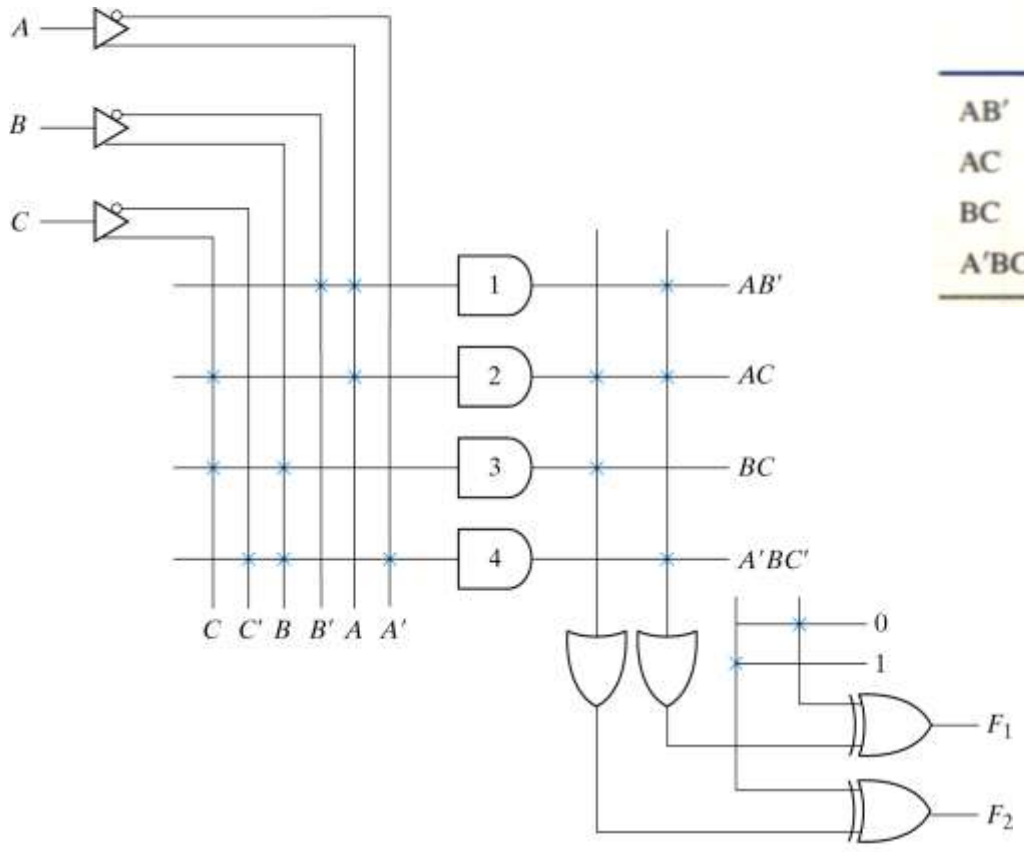


Fig. 7-14 PLA with 3 Inputs, 4 Product Terms, and 2 Outputs

Programming Table:

- First: lists the product terms numerically
- Second: specifies the required paths between inputs and AND gates
- Third: specifies the paths between the AND and OR gates
- For each output variable, we may have a T(ture) or C(complement) for programming the XOR gate

Simplification of PLA:

- Careful investigation must be undertaken in order to reduce the number of distinct product terms, PLA has a finite number of AND gates.
- Both the true and complement of each function should be simplified to see which one can be expressed with fewer product terms and which one provides product terms that are common to other functions.

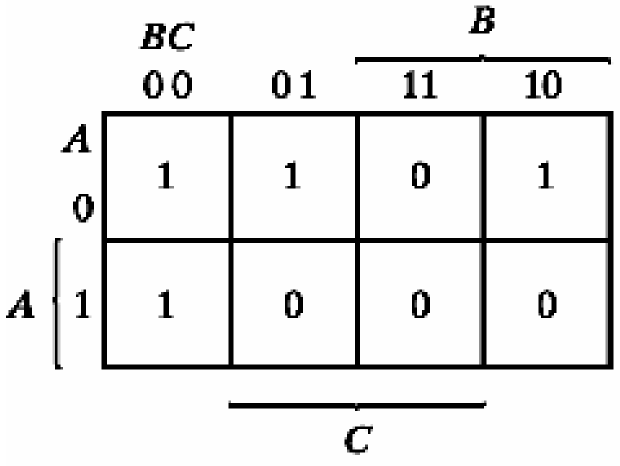
Example

Implement the following two Boolean functions with a PLA:

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

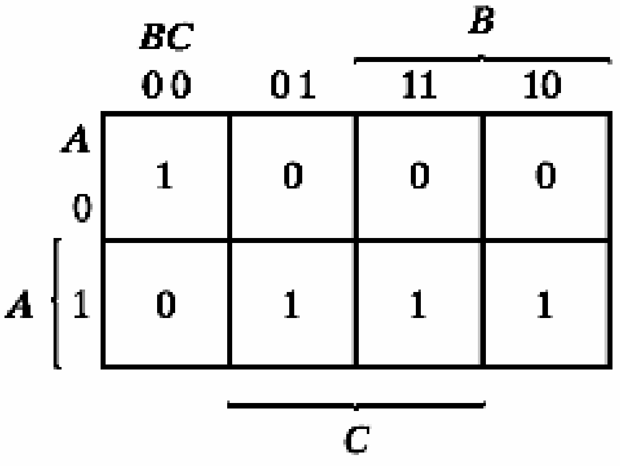
$$F_2(A, B, C) = \sum(0, 5, 6, 7)$$

The two functions are simplified in the maps



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

$F_1 = (AB + AC + BC)'$



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

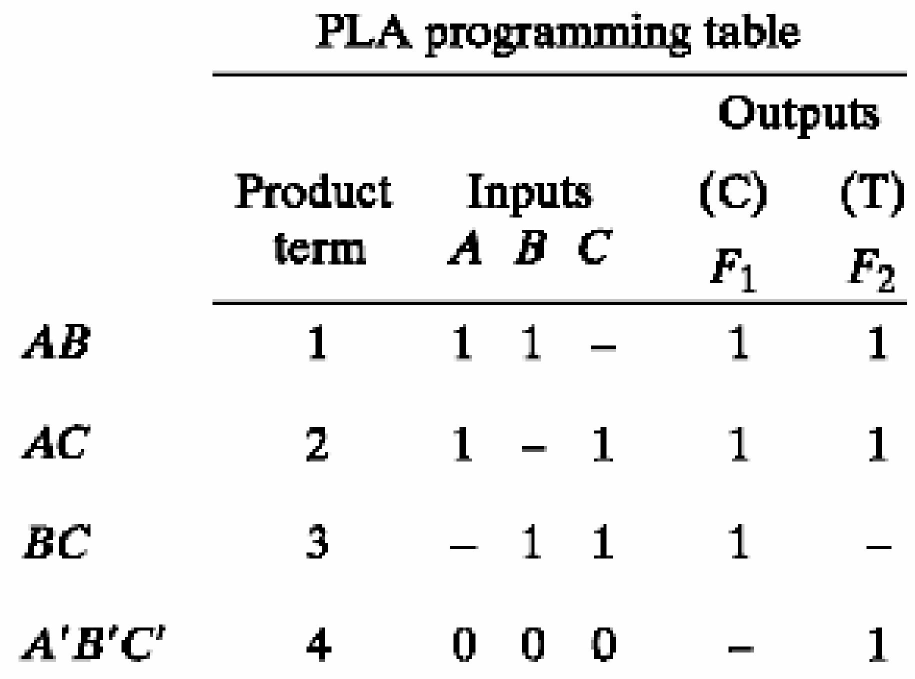
$F_2 = (A'C + A'B + AB'C')'$

PLA table by simplifying the function

- Both the true and complement of the functions are simplified in sum of products.
- We can find the same terms from the group terms of the functions of F_1, F_1',F_2 and F_2' which will make the minimum terms.

F1 = (AB + AC + BC)’

F2 = AB + AC + A’B’C’

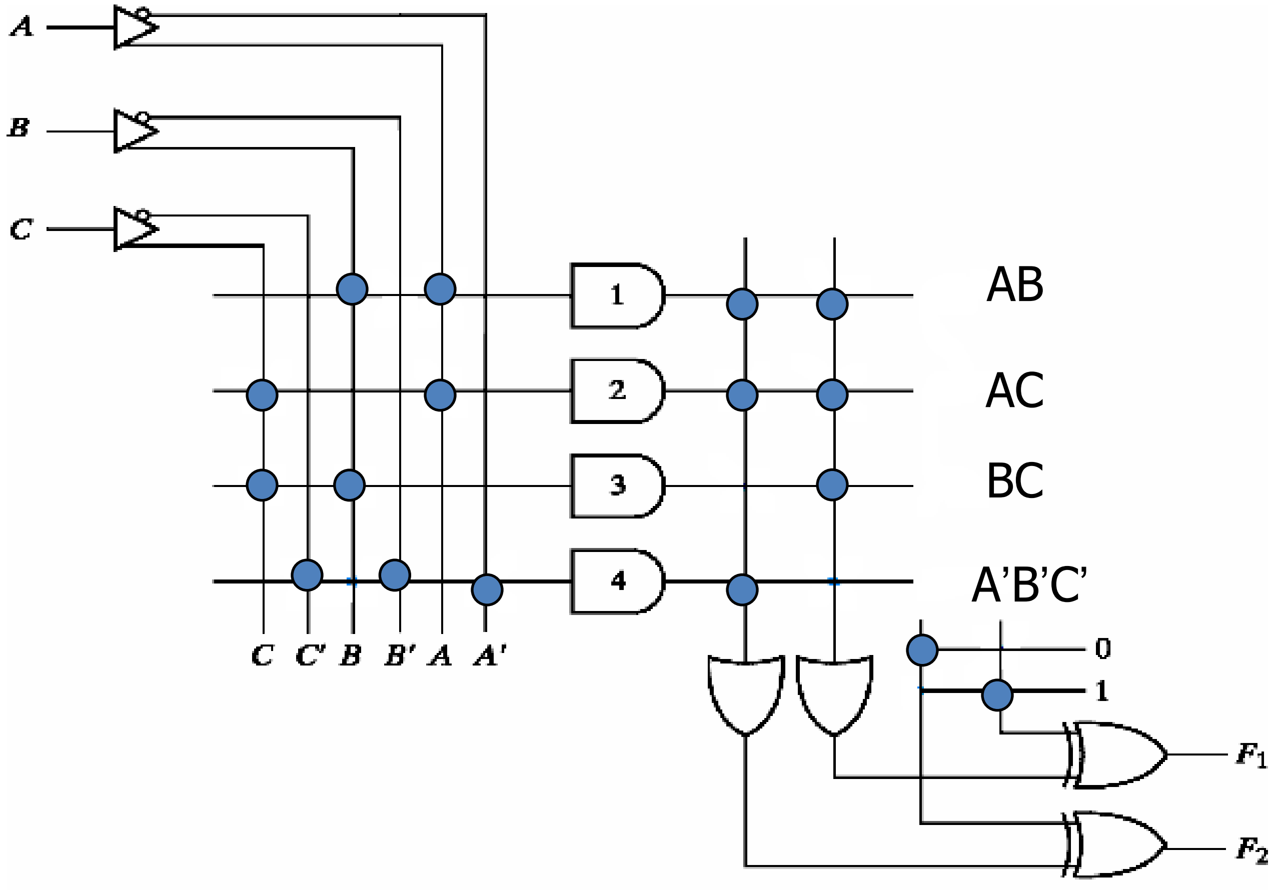


PLA programming table

	Product term	Inputs			Outputs (C)	Outputs (T)
		A	B	C	F_1	F_2
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

Fig. 7-15 Solution to Example 7-2

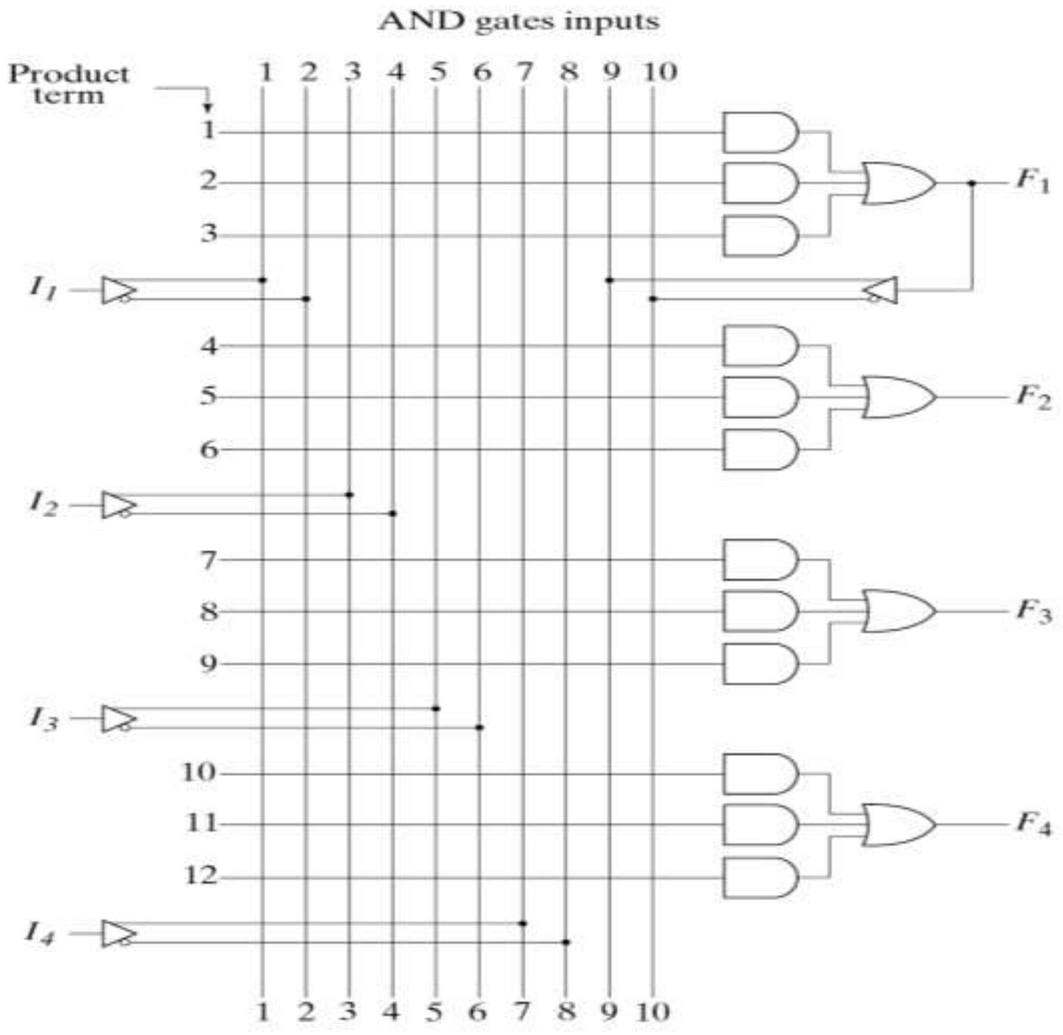
PLA implementation



Programmable Array Logic

- The PAL is a programmable logic device with a fixed OR array and a programmable AND array.

Because only the AND gates are programmable, the PAL is easier to program than, but is not as flexible as, the PLA



PAL with Four Inputs, Four Outputs, and Three-Wide AND-OR Structure

PAL

- When designing with a PAL, the Boolean functions must be simplified to fit into each section.
- Unlike the PLA, a product term cannot be shared among two or more OR gates. Therefore, each function can be simplified by itself without regard to common product terms.
- The output terminals are sometimes driven by three-state buffers or inverters.

Example

$w(A, B, C, D) = \sum(2, 12, 13)$

$x(A, B, C, D) = \sum(7, 8, 9, 10, 11, 12, 13, 14, 15)$

$y(A, B, C, D) = \sum(0, 2, 3, 4, 5, 6, 7, 8, 10, 11, 15)$

$z(A, B, C, D) = \sum(1, 2, 8, 12, 13)$

Simplifying the four functions as following Boolean functions:

$w = ABC' + A'B'CD'$

$x = A + BCD$

$y = A'B + CD + B'D'$

$z = ABC' + A'B'CD' + AC'D' + A'B'C'D = w + AC'D' + A'B'C'D$

PAL Table

- z has four product terms, and we can replace by w with two product terms, this will reduce the number of terms for z from four to three.

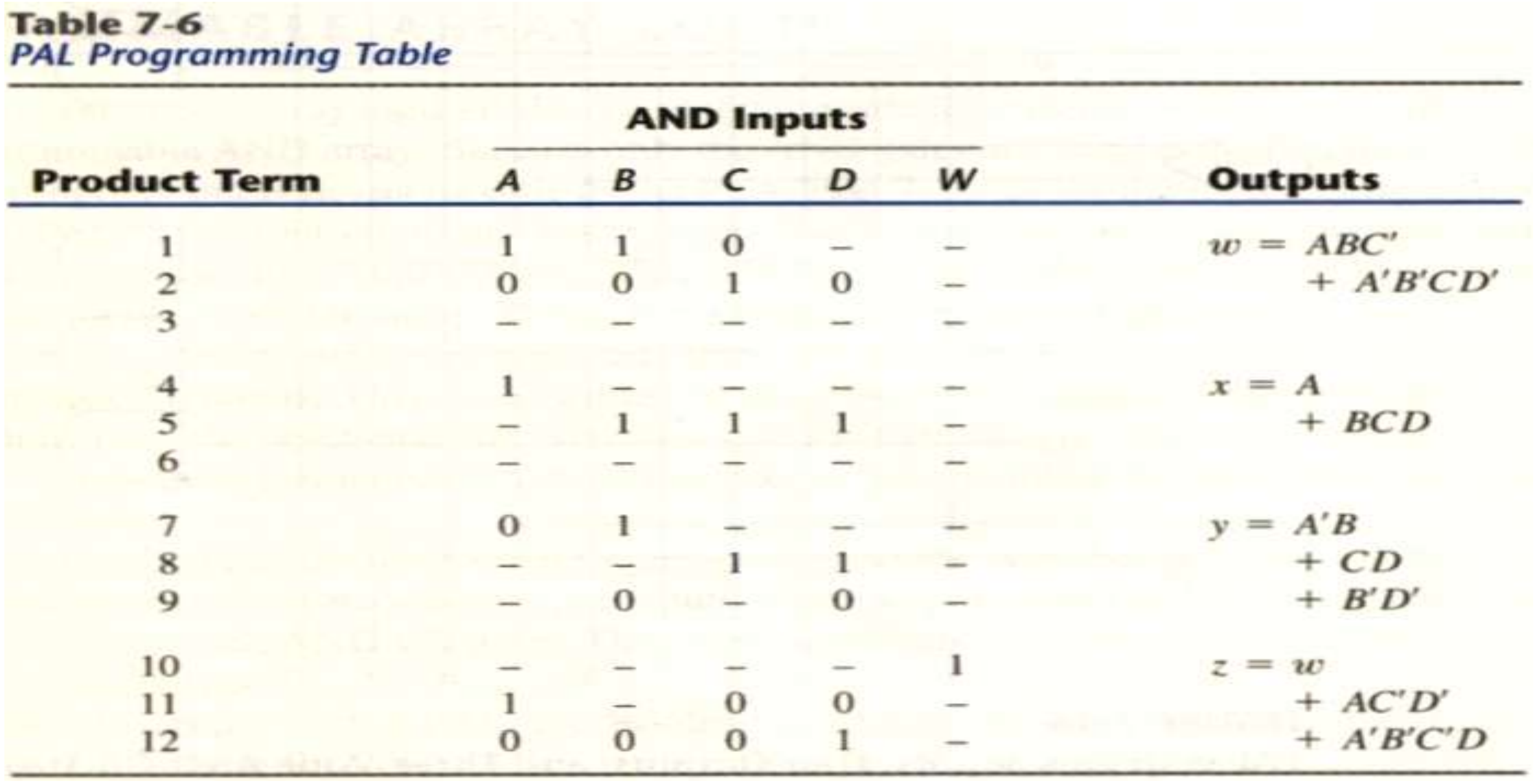
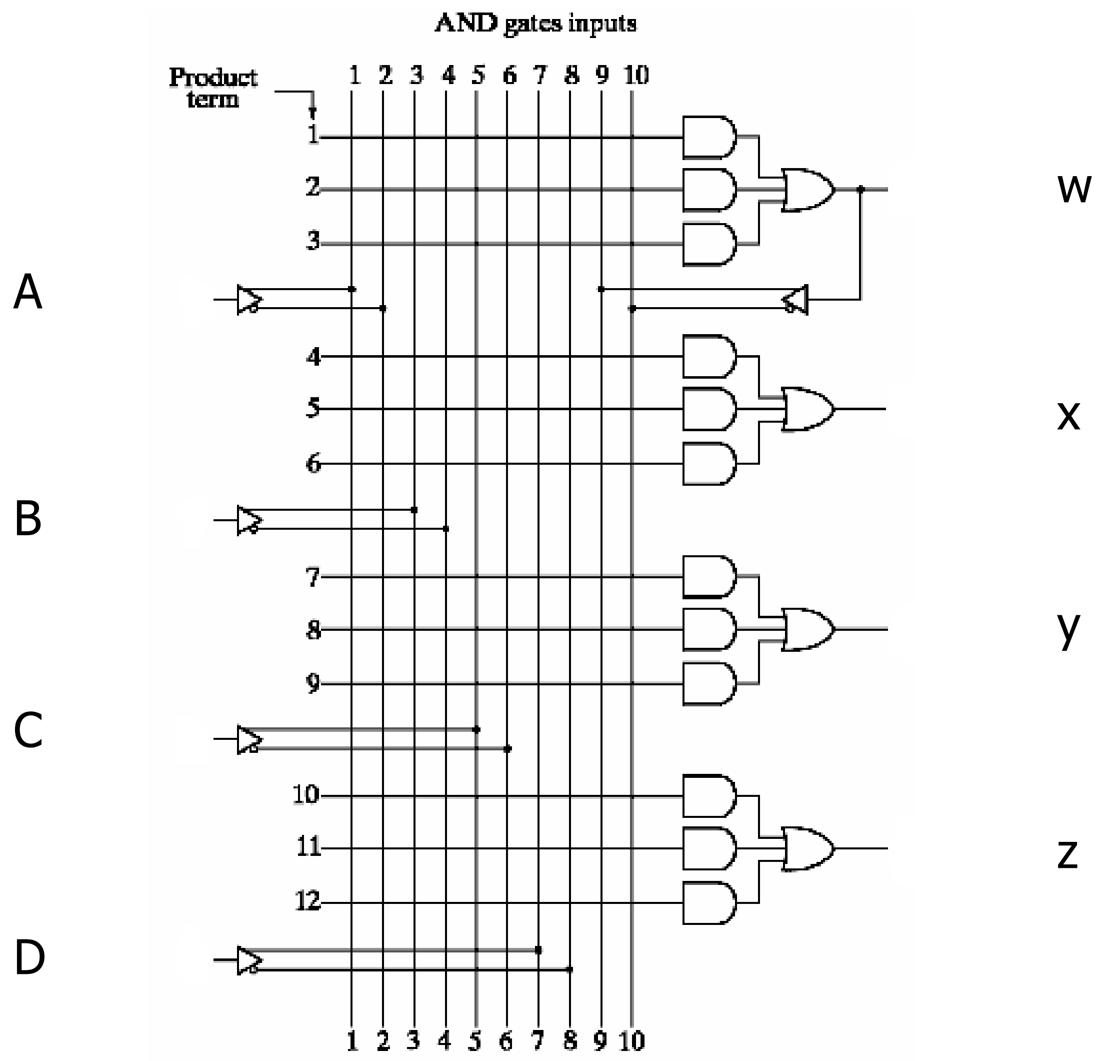


Table 7-6
PAL Programming Table

	AND Inputs					
Product Term	**A**	**B**	**C**	**D**	**W**	**Outputs**
1	1	1	0	–	–	$w = ABC'$
2	0	0	1	0	–	$+ A'B'CD'$
3	–	–	–	–	–	
4	1	–	–	–	–	$x = A$
5	–	1	1	1	–	$+ BCD$
6	–	–	–	–	–	
7	0	1	–	–	–	$y = A'B$
8	–	–	1	1	–	$+ CD$
9	–	0	–	0	–	$+ B'D'$
10	–	–	–	–	1	$z = w$
11	1	–	0	0	–	$+ AC'D'$
12	0	0	0	1	–	$+ A'B'C'D$

PAL implementation



Fuse map for example

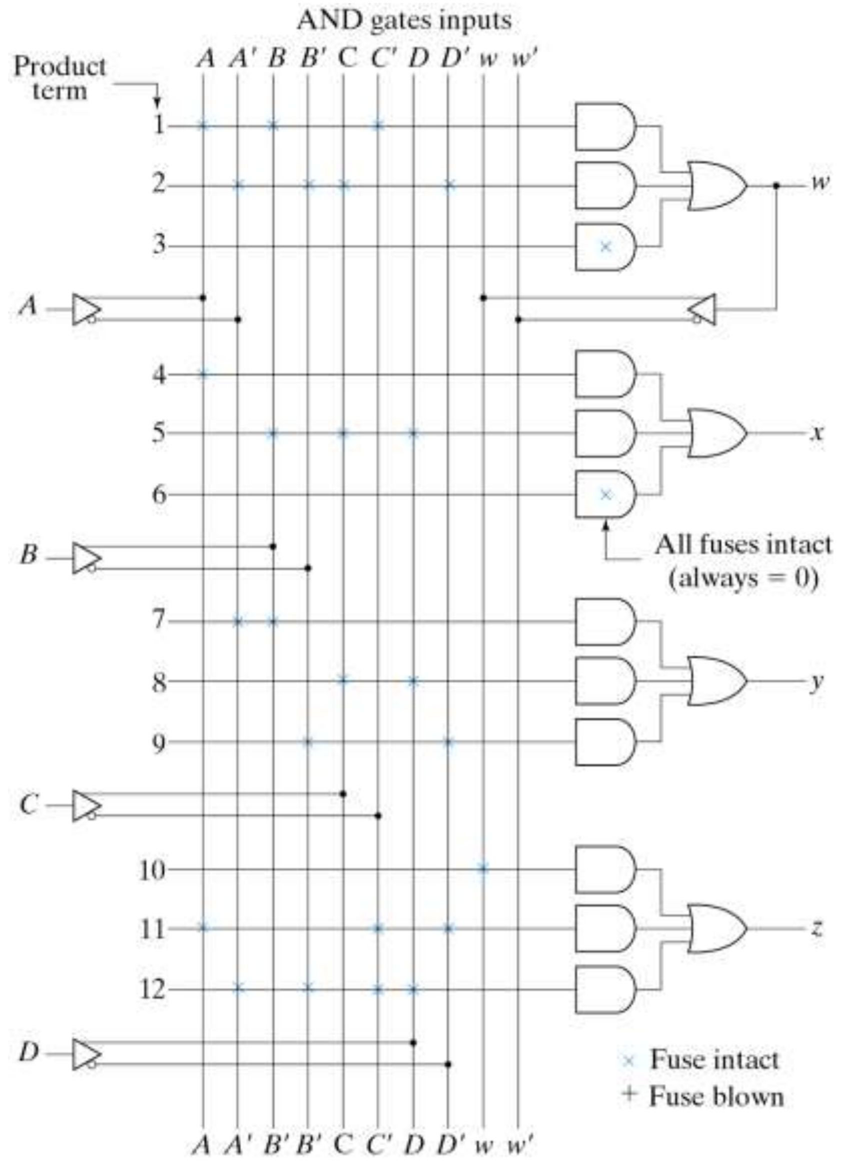


Fig. 7-17 Fuse Map for PAL as Specified in Table 7-6

Sequential Programmable Devices

- Sequential programmable devices include both gates and flip-flops.
- There are several types of sequential programmable devices, but the internal logic of these devices is too complex to be shown here.
- We will describe three major types without going into their detailed construction.

Sequential Programmable Devices

1. Sequential (or simple) Programmable Logic Device (SPLD)
2. Complex Programmable Logic Device (CPLD)
3. Field Programmable Gate Array (FPGA)

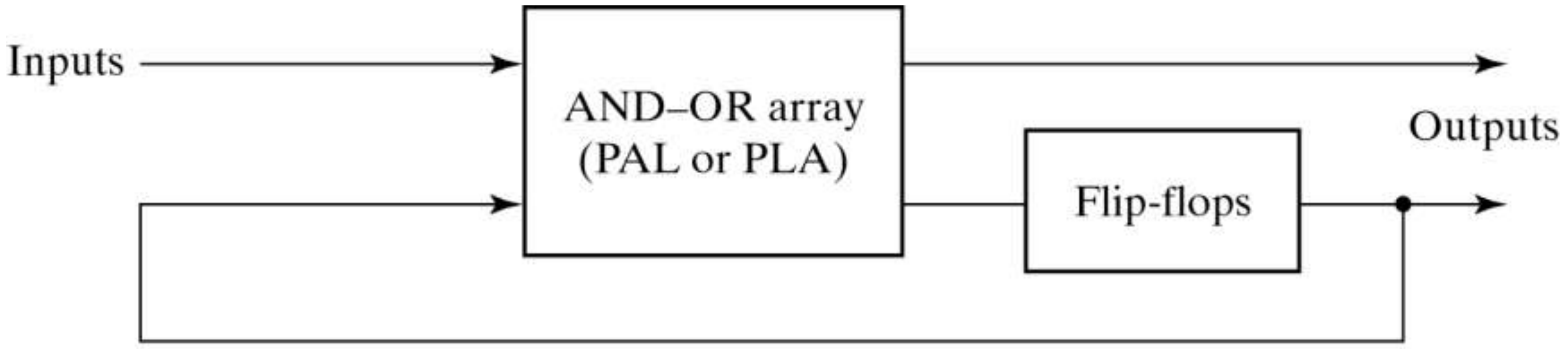


Fig. 7-18 Sequential Programmable Logic Device

Field-programmable Logic Sequencer (FPLS)

- The first programmable device developed to support sequential circuit implementation is the field-programmable logic sequencer(FPLS).
- A typical FPLS is organized around a PLA with several outputs driving flip-flops.
- The flip-flops are flexible in that they can be programmed to operate as either JK or D type.
- The FPLS did not succeed commercially because it has too many programmable connections.

SPLD

- Each section of an SPLD is called a macrocell.
- A macrocell is a circuit that contains a sum-of-products combinational logic function and an optional flip-flop.
- We will assume an AND-OR sum of products but in practice, it can be any one of the two-level implementation presented in Sec.3-7.

Macrocell

- Fig.7-19 shows the logic of a basic macrocell.
- The AND-OR array is the same as in the combinational PAL shown in Fig.7-16.

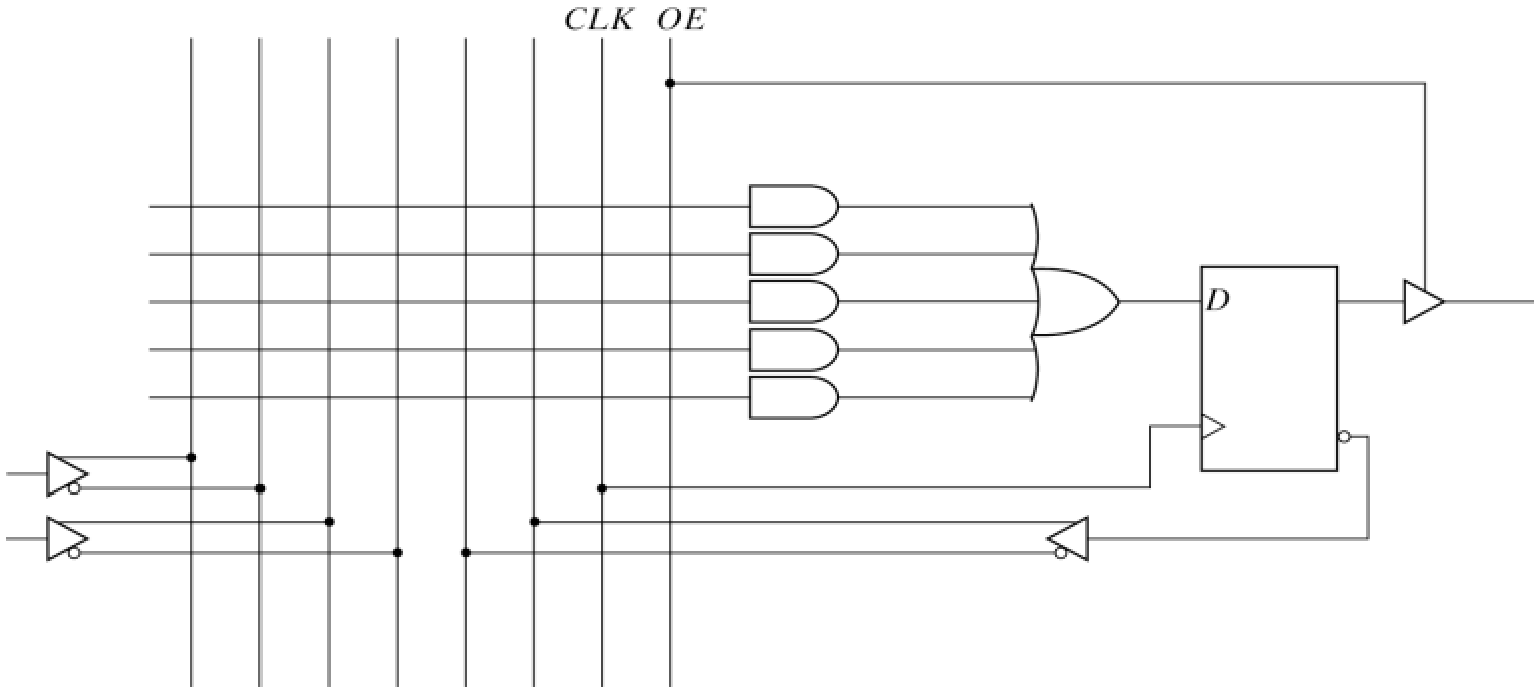


Fig. 7-19 Basic Macrocell Logic

CPLD

- A typical SPLD has from 8 to 10 macrocells within one IC package. All the flip-flops are connected to the common CLK input and all three-state buffers are controlled by the EO input.
- The design of a digital system using PLD often requires the connection of several devices to produce the complete specification. For this type of application, it is more economical to use a complex programmable logic device (CPLD).
- A CPLD is a collection of individual PLDs on a single integrated circuit.

CPLD

- General configuration of a CPLD consists of multiple PLDs interconnected through a programmable switch matrix.
- 8 to 16 macrocell per PLD.

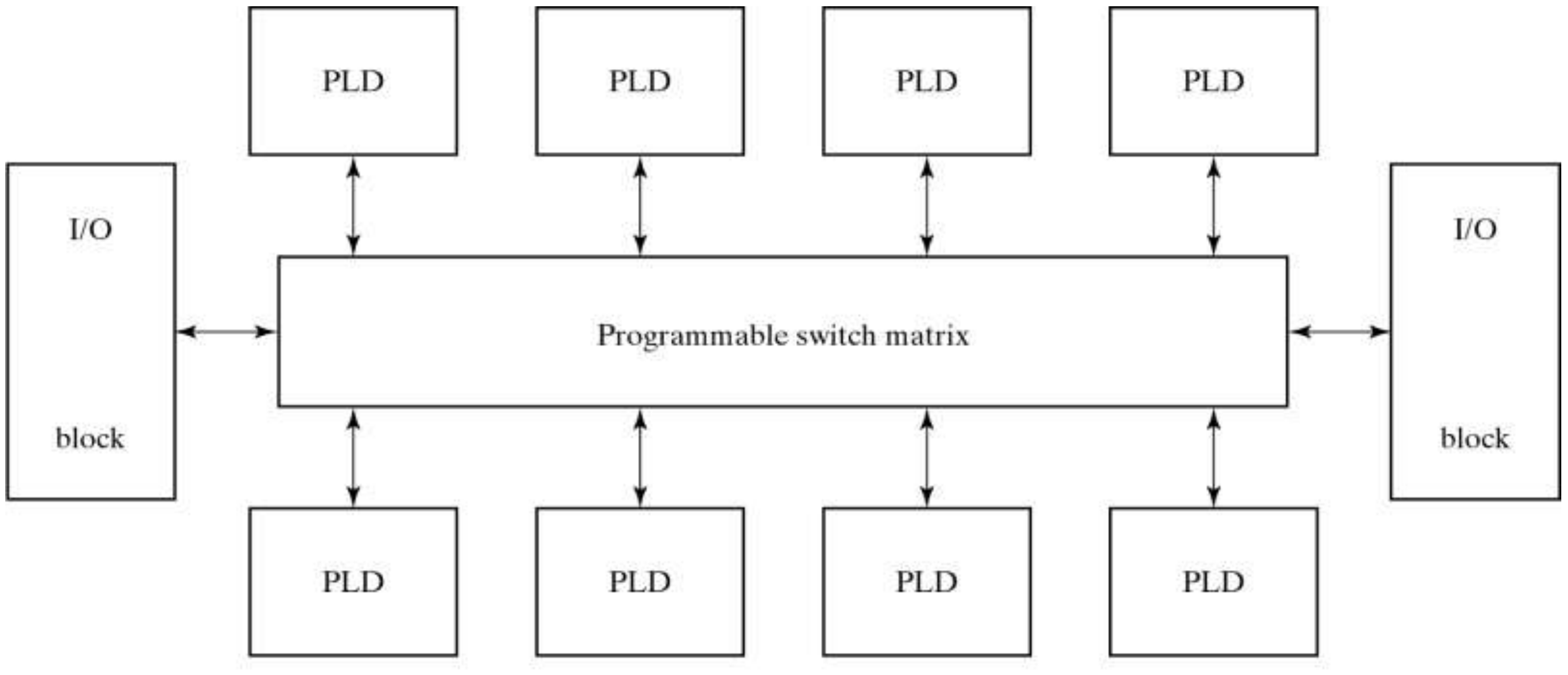


Fig. 7-20 General CPLD Configuration

Gate Array

- The basic component used in VLSI design is the gate array.
- A gate array consists of a pattern of gates fabricated in an area of silicon that is repeated thousands of times until the entire chip is covered with the gates.
- Arrays of one thousand to hundred thousand gates are fabricated within a single IC chip depending on the technology used.

FPGA

- FPGA is a VLSI circuit that can be programmed in the user's location.
- A typical FPGA logic block consists of look-up tables, multiplexers, gates, and flip-flops.
- Look-up table is a truth table stored in a SRAM and provides the combinational circuit functions for the logic block.