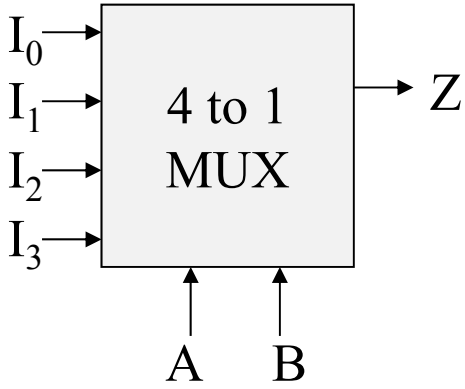
Unit 9

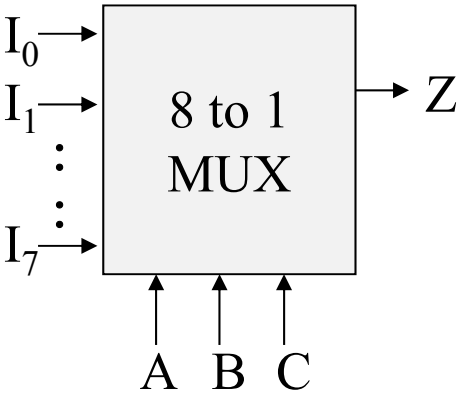
Multiplexers, Decoders, and Programmable Logic Devices

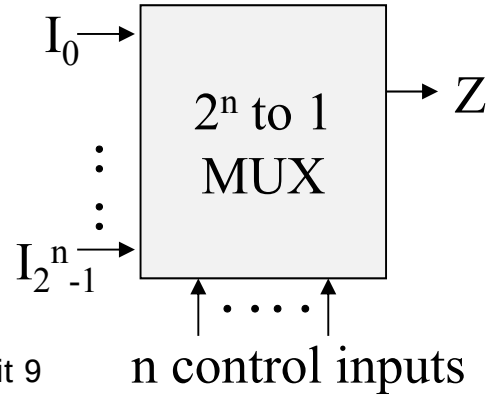
Outline

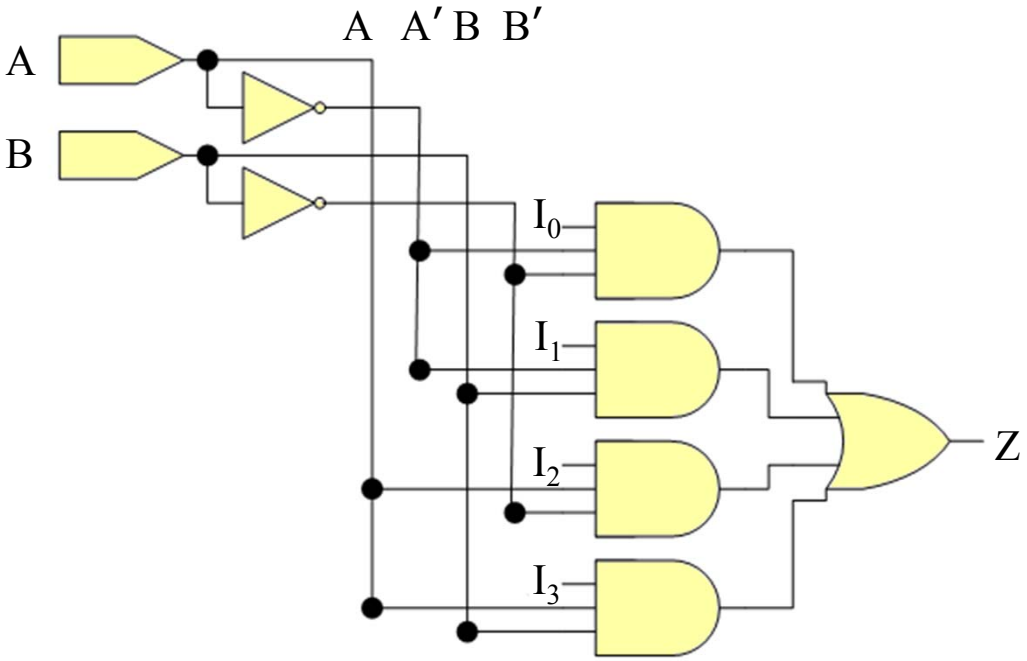
- Multiplexers
- Three state buffers
- Decoders
- Encoders
- Read Only Memories (ROMs)
- Programmable logic devices
- Field Programmable Gate Arrays (FPGAs)

Multiplexers (1/5)









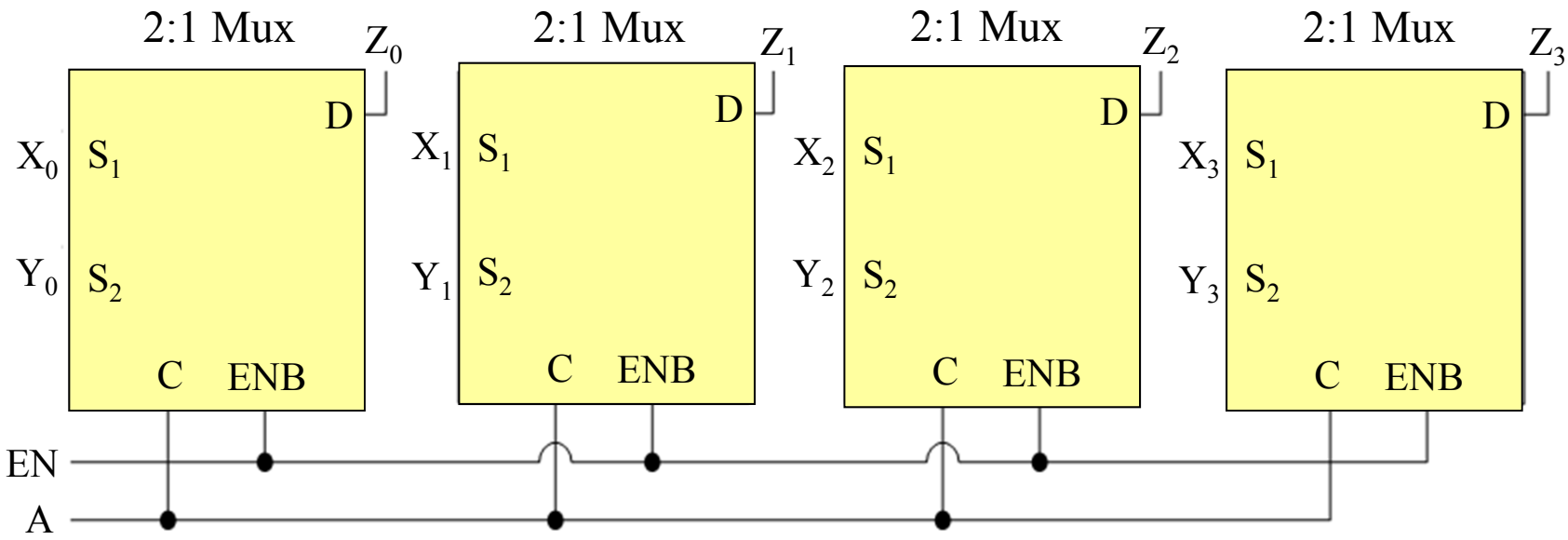
$$4-1: Z = A'B'I_0 + A'BI_1 + AB'I_2 + ABI_3$$

$$8-1: Z = A'B'C'I_0 + A'B'CI_1 + A'BC'I_2 + A'BCI_3 + AB'C'I_4 + AB'CI_5 + ABC'I_6 + ABCI_7$$

$$2^n-1: Z = \sum_{k=0}^{2^n-1} m_k I_k$$

Multiplexers (2/5)

Quad multiplexer to select data

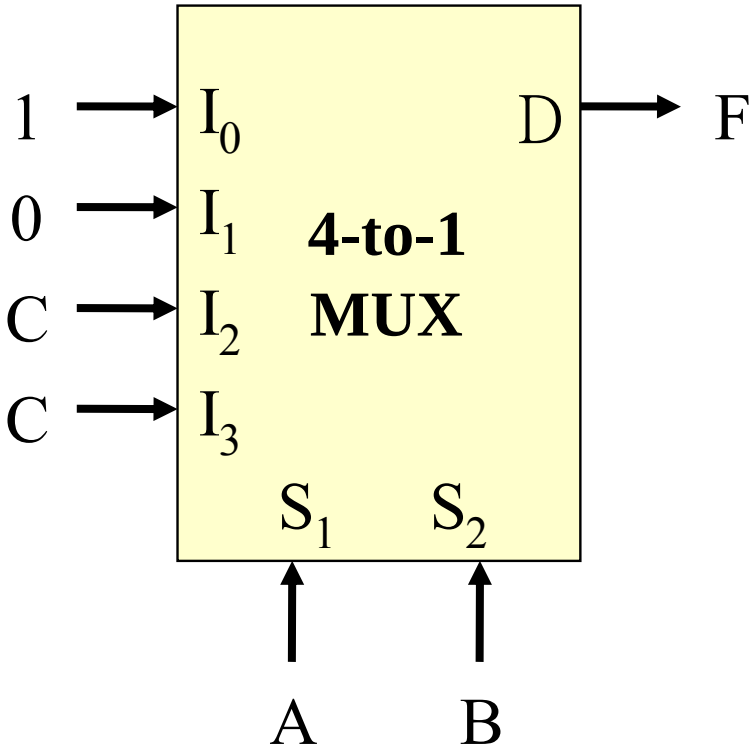


Multiplexers (3/5)

4-to-1 MUX to realize 3-variable function

Ex :

$$F(A,B,C) = A'B' + AC$$
$$= A'B' + AC(B + B')$$
$$= A'B' \cdot 1 + AB'C + ABC$$

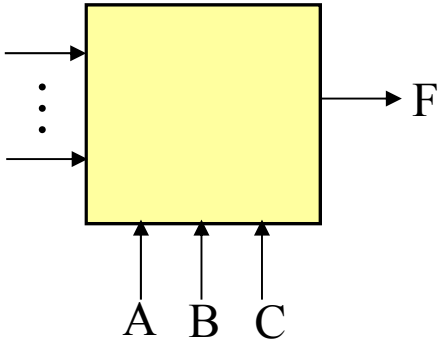


Multiplexers (4/5)

8-to-1 MUX to realize 4-variable function

$$F = A'B'C' + A'BC + ABD' + AC'D' + B'CD$$

$$= A'B'C'(D+D') + A'BC(D+D') + AB(C+C')D' + A(B+B')C'D' + (A+A')B'CD$$



$A'B'C' = 1$

$A'B'C = D$

$A'BC' = 0$

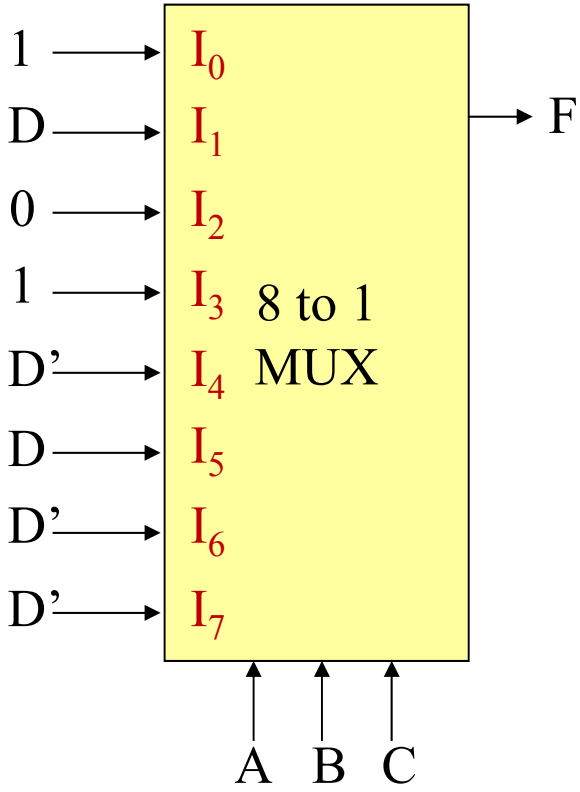
$A'BC = 1$

$AB'C' = D'$

$AB'C = D$

$ABC' = D'$

$ABC = D'$



Multiplexers (5/5)

	A	B	C	D	F	
I_0	0	0	0	0	1	1
	0	0	0	1	1	
I_1	0	0	1	0	0	D
	0	0	1	1	1	
I_2	0	1	0	0	0	0
	0	1	0	1	0	
I_3	0	1	1	0	1	1
	0	1	1	1	1	
I_4	1	0	0	0	1	$\overline{D}$
	1	0	0	1	0	
I_5	1	0	1	0	0	D
	1	0	1	1	1	
I_6	1	1	0	0	1	$\overline{D}$
	1	1	0	1	0	
I_7	1	1	1	0	1	$\overline{D}$
	1	1	1	1	0	

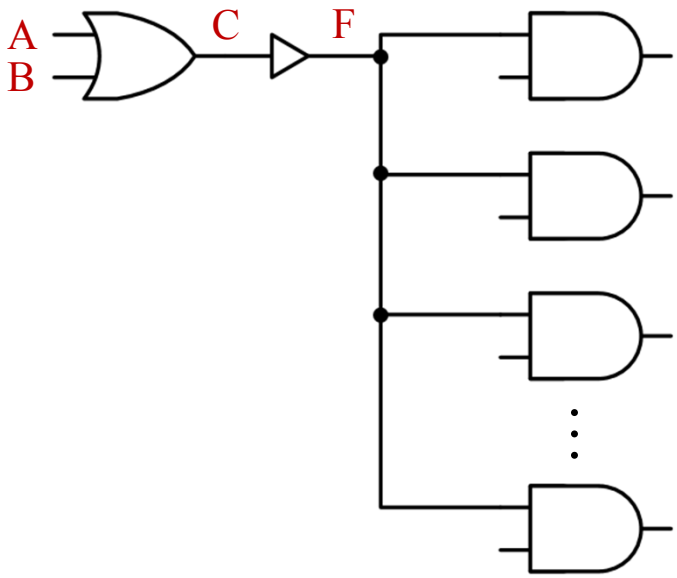
$$F = A'B'C'(D+D') + A'BC(D+D') + AB(C+C')D' + A(B+B')C'D' + (A+A')B'CD$$

V_1	V_2	$\cdots$	V_n	F			
0	0	$\cdots$	0	0	0	1	1
0	0	$\cdots$	1	0	1	0	1
				$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
				0	V_n	$\overline{V_n}$	1

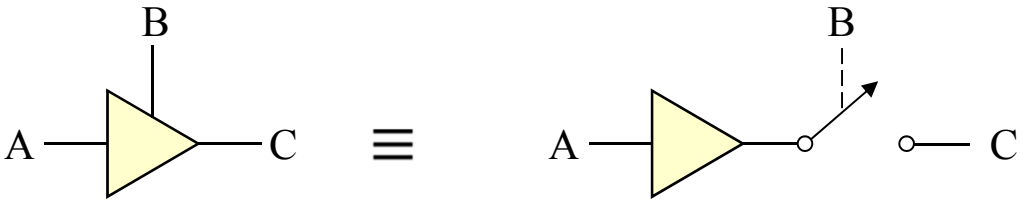
$$F = A'B'C'+A'B'CD+A'BC+AB'C'D'+AB'CD+ABC'D'+ABCD'$$
$$= A'B'C'+A'BC+ABD'+AC'D'+B'CD$$

Three-State Buffers (1/4)

Buffers: to increase the driving capability of a gate output



Tri-state (three-state) Buffers: permits gate outputs to be connected together

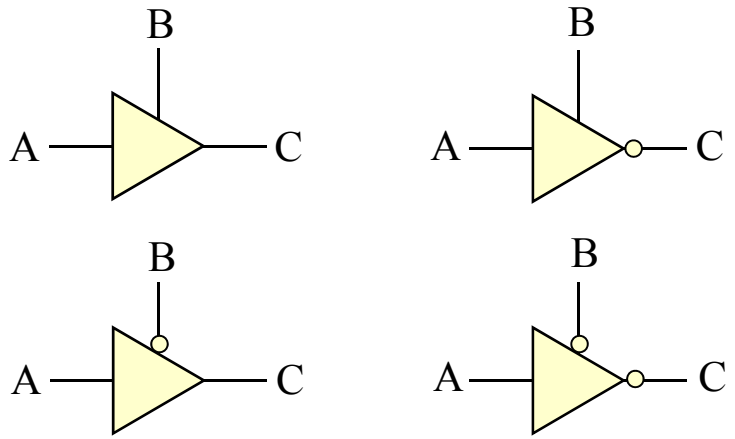


Three-State Buffers (2/4)

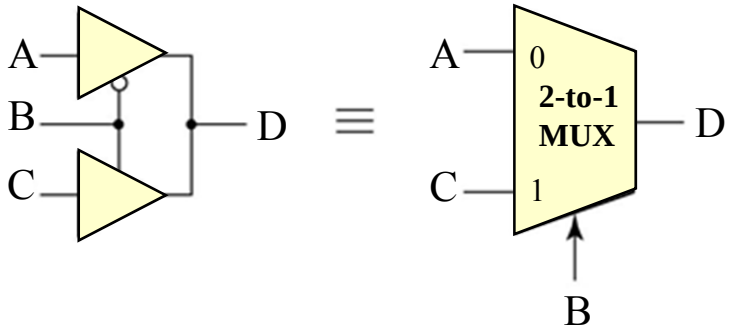
Truth table

B	A	C
0	0	Z
0	1	Z
1	0	0
1	1	1

Truth Table

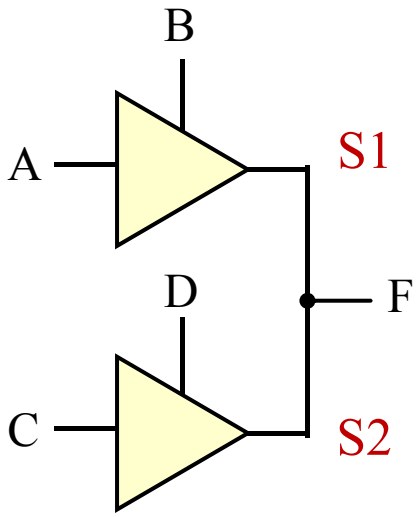


Data selection using three-state buffers



Three-State Buffers (3/4)

Circuits with two three-state buffers

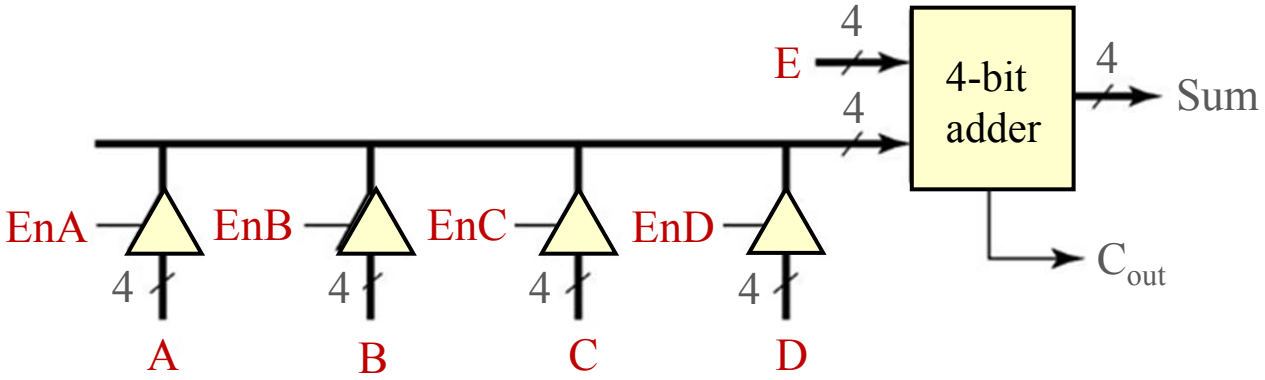


S1	S2			
	X	0	1	Z
X	X	X	X	X
0	X	0	X	0
1	X	X	1	1
Z	X	0	1	Z

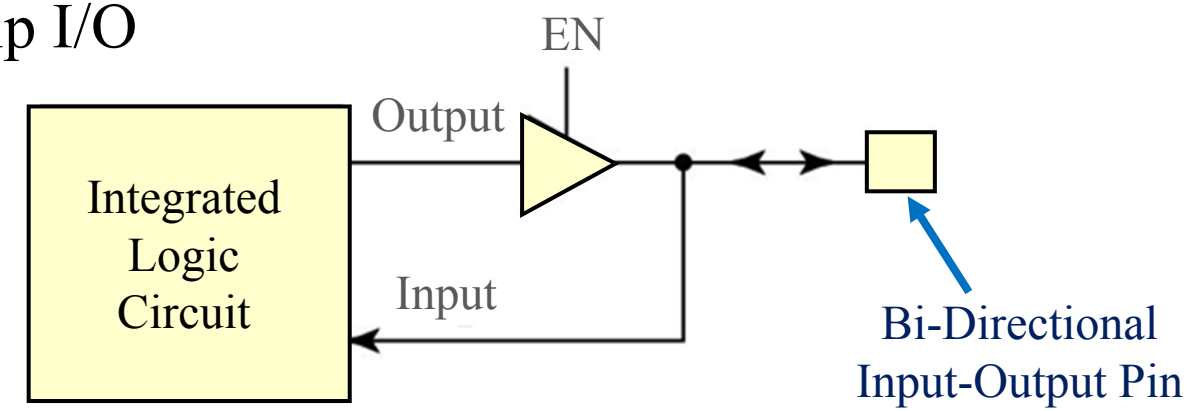
Three State Buffers (4/4)

Applications

1. Bus

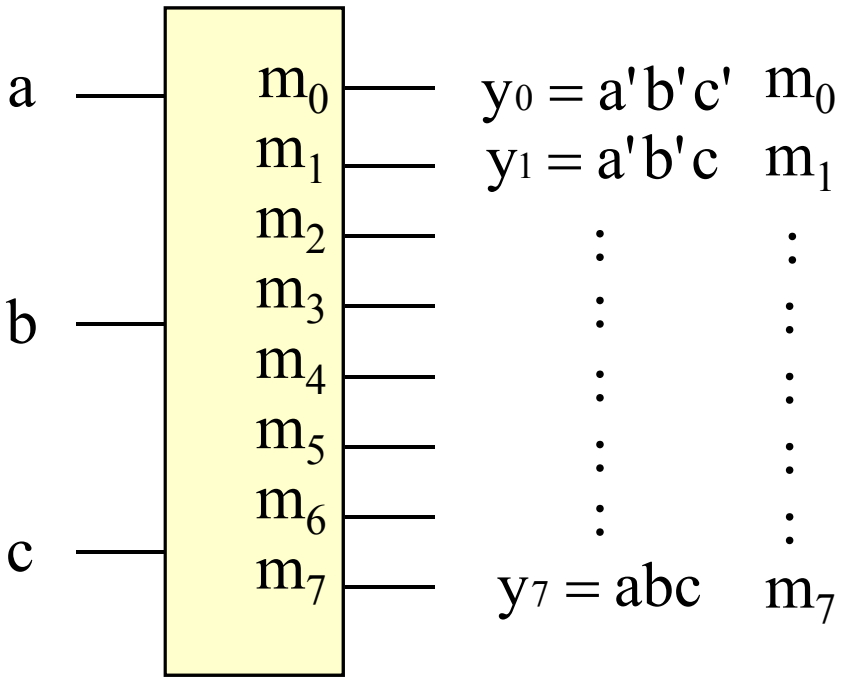


2. Chip I/O



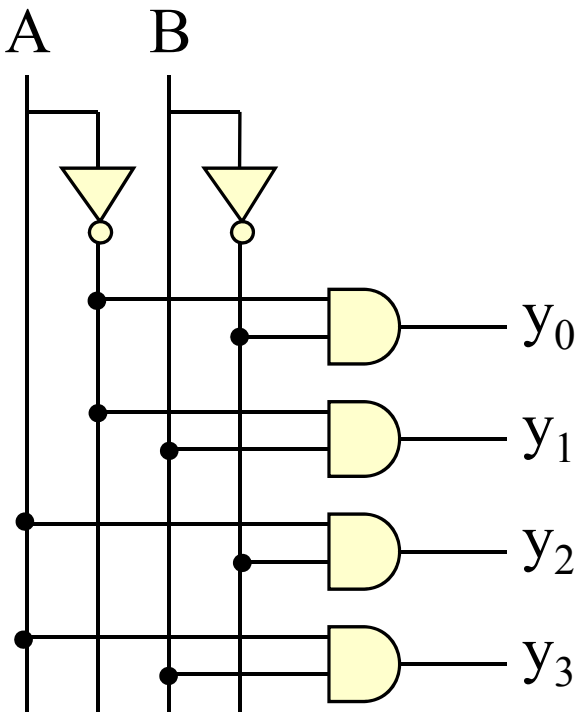
Decoders (1/4)

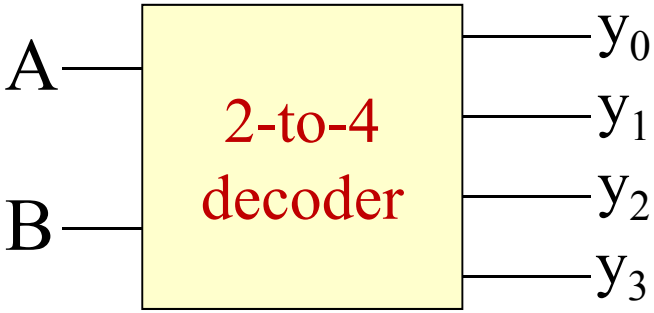
Generate all of the minterms of inputs

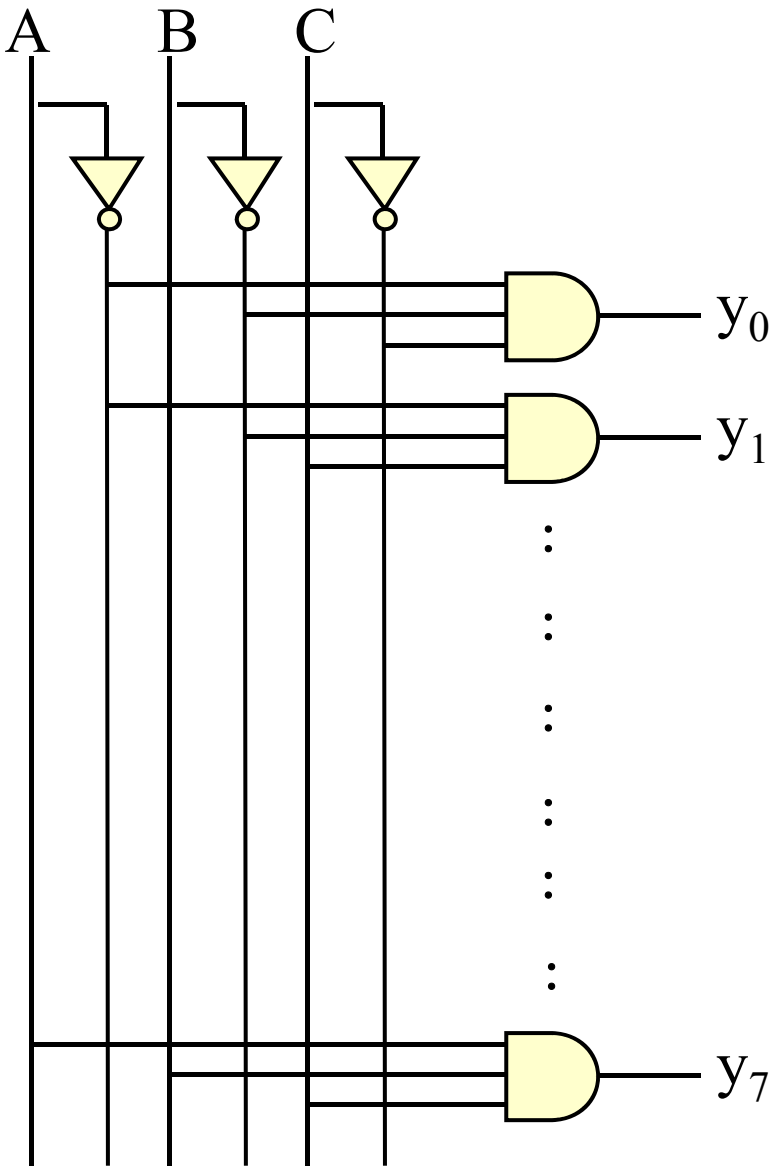


a	b	c	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	1	0	··	··	··	··	··	0
0	0	1	0	1						:
0	1	0	:		1					:
0	1	1	:			1				:
1	0	0	:				1			:
1	0	1	:					1		:
1	1	0	:						1	0
1	1	1	0	··	··	··	··	··	0	1

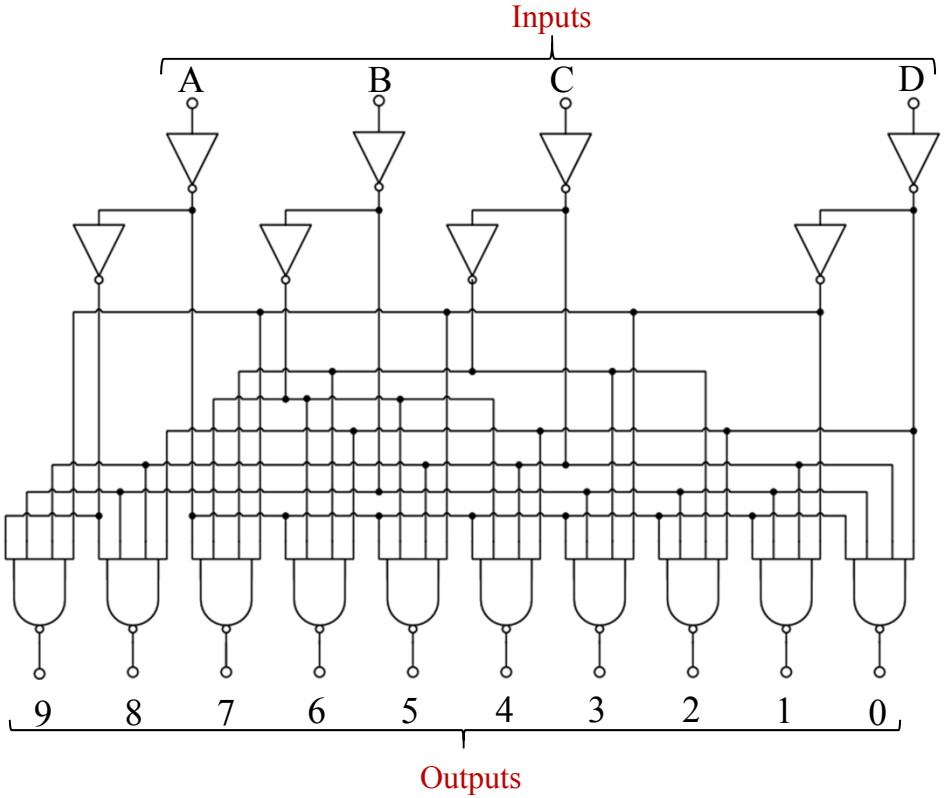
Decoders (2/4)

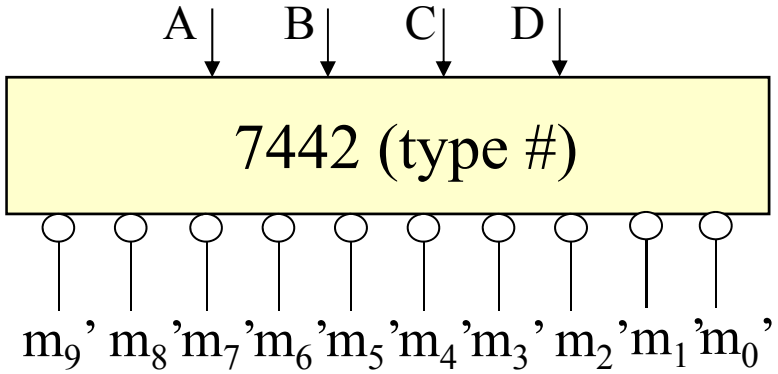






Decoders (3/4)



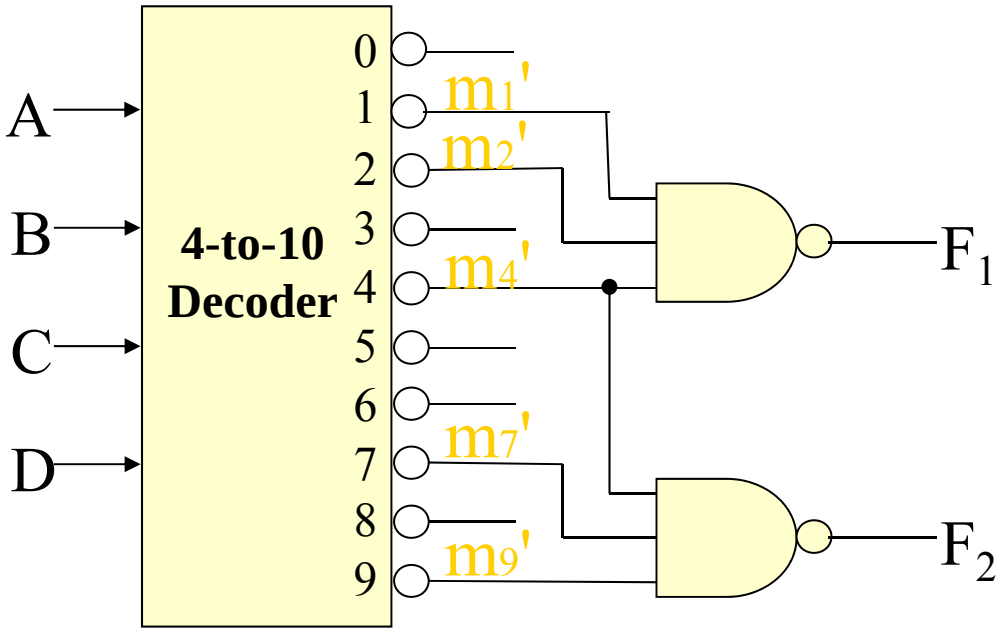


BCD input				Decimal output									
A	B	C	D	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0
1	0	1	0	1	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	1	1	1	1	1	1
1	1	0	1	1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1

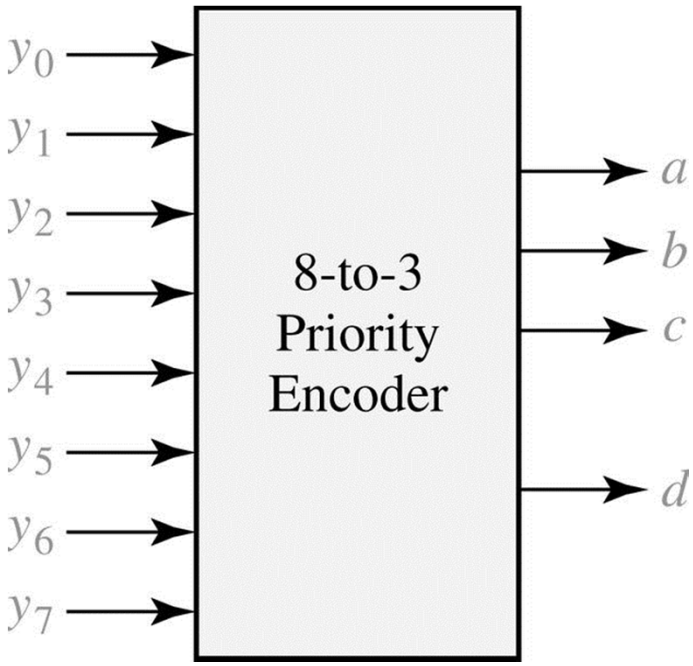
Decoders (4/4)

Realization of a multiple output circuit using a decoder

$$F_1 = m_1 + m_2 + m_4 \qquad F_2 = m_4 + m_7 + m_9$$



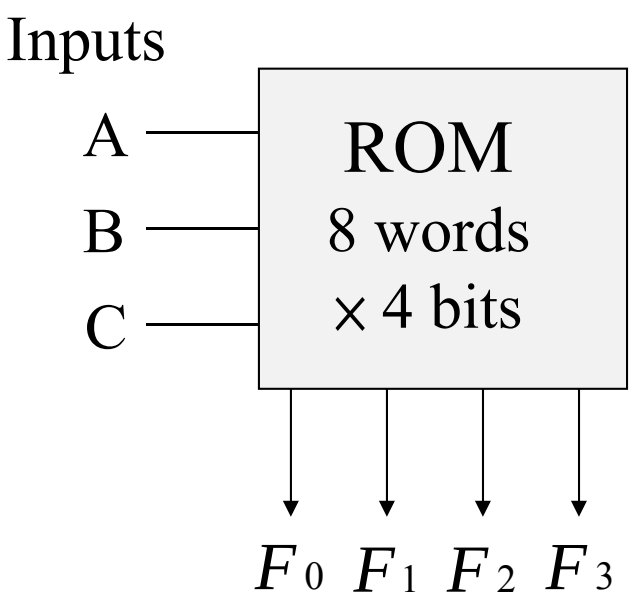
Encoders



y0	y1	y2	y3	y4	y5	y6	y7	a	b	c	d
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
X	1	0	0	0	0	0	0	0	0	1	1
X	X	1	0	0	0	0	0	0	1	0	1
X	X	X	1	0	0	0	0	0	1	1	1
X	X	X	X	1	0	0	0	1	0	0	1
X	X	X	X	X	1	0	0	1	0	1	1
X	X	X	X	X	X	1	0	1	1	0	1
X	X	X	X	X	X	X	1	1	1	1	1

Read Only Memories (1/5)

An LSI circuit to realize multiple output Boolean function(s)



A	B	C	F_0	F_1	F_2	F_3
0	0	0	1	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	1	0	1	1	0
1	0	0	0	1	1	0
1	0	1	1	1	1	0
1	1	0	0	0	1	1
1	1	1	1	1	1	1

Stored in ROM (2^3 words of 4 bits each)

$$F_0 = m_0 + m_1 + m_5 + m_7$$

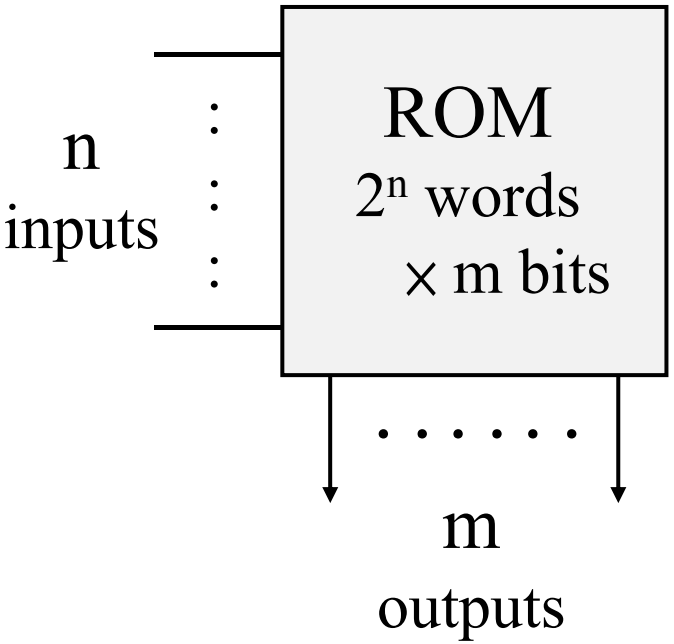
$$F_1 = m_2 + m_3 + m_4 + m_5 + m_7$$

$$F_2 = m_0 + m_1 + m_3 + m_4 + m_5 + m_6 + m_7$$

$$F_3 = m_6 + m_7$$

Read Only Memories (2/5)

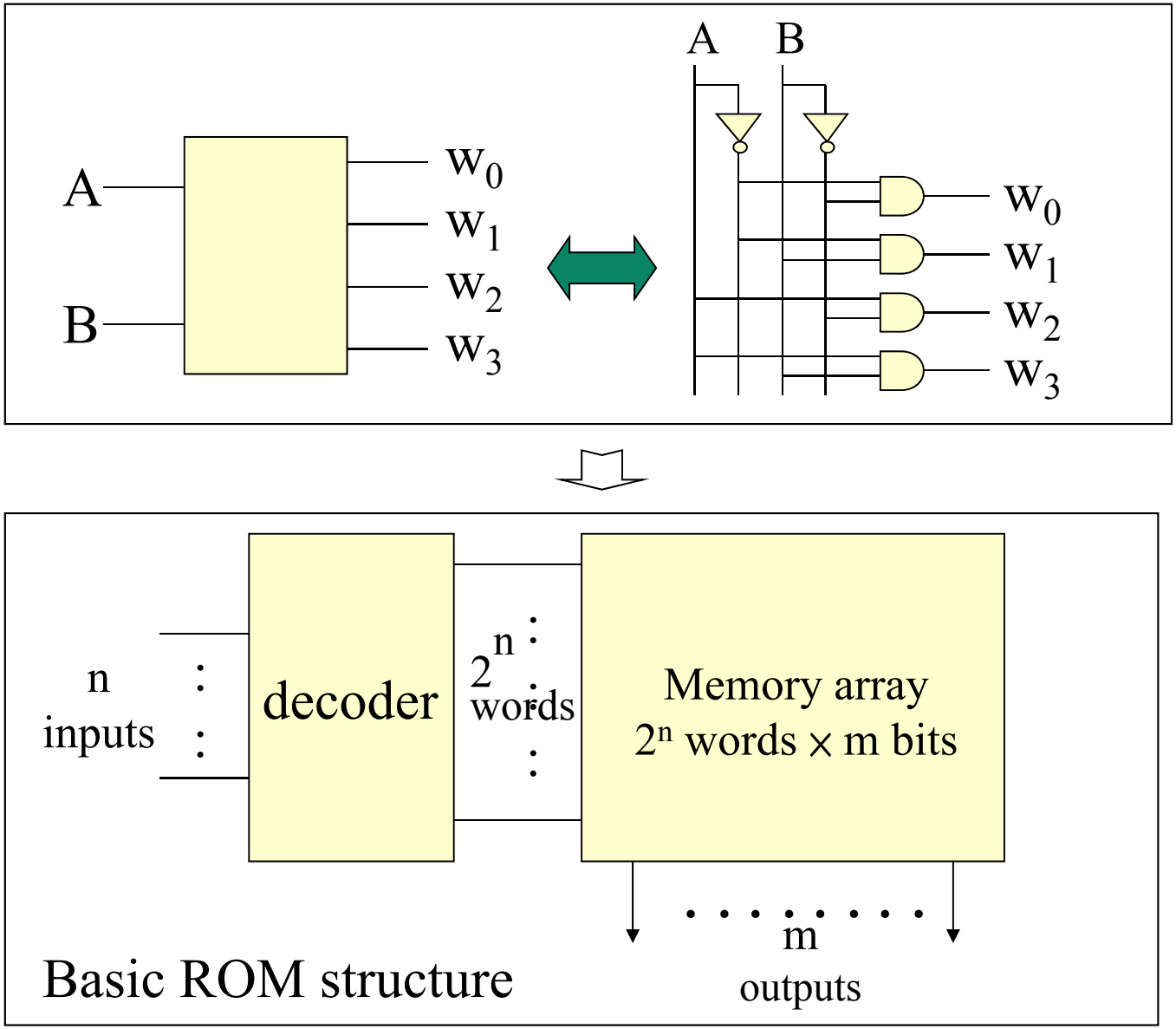
General Form



n inputs				m outputs				
				F_0	⋯	⋯	⋯	F_{m-1}
0	⋯	0	0	0	0	⋯	1	0
0	⋯	0	1	:				:
:			:	:				:
:			:	:				:
:			:	:				:
:			:	:				:
1	⋯	1	1	1	0	⋯	1	0

2^n words

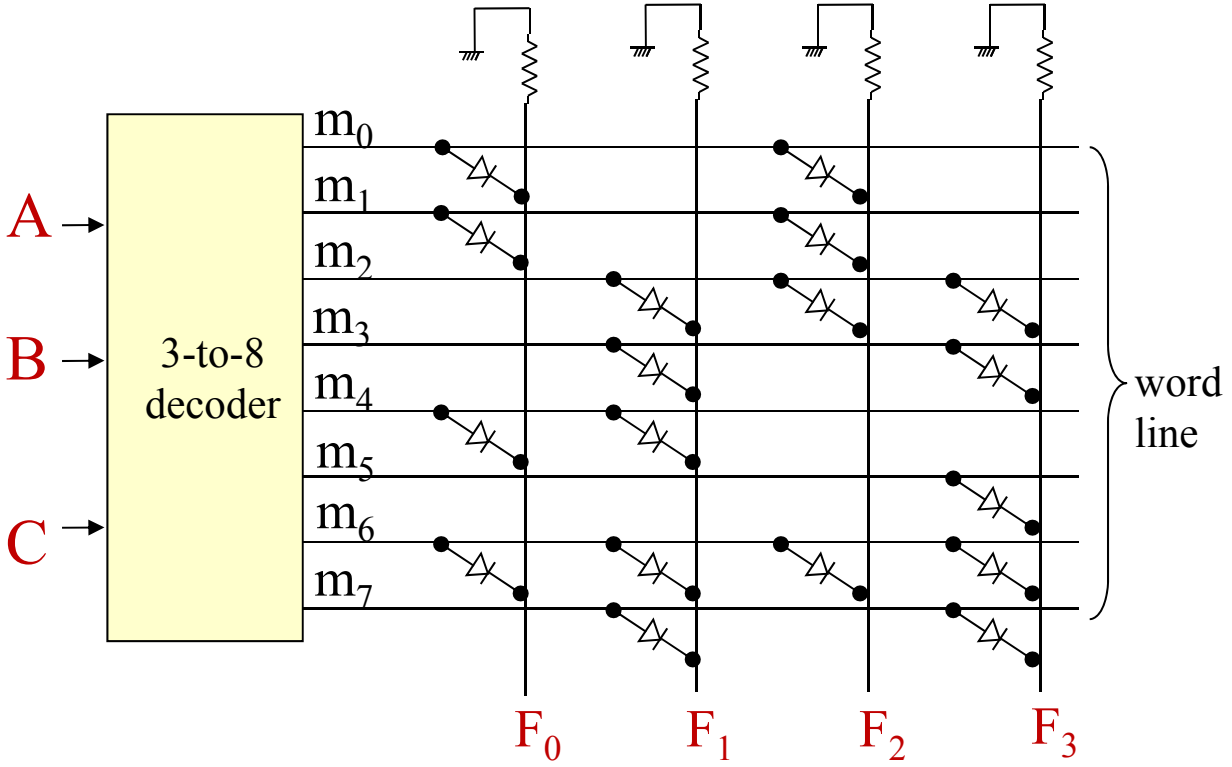
Read Only Memories (3/5)

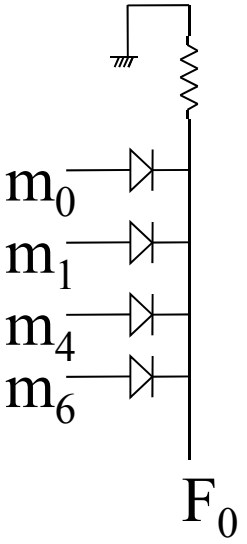


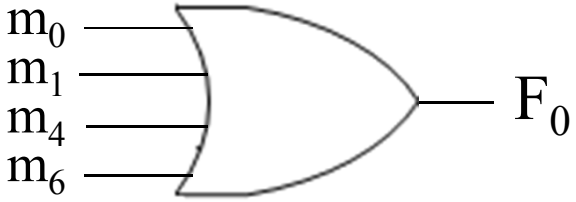
Read Only Memories (4/5)

ROM as logic devices (use decoder and diodes)

A	B	C	F_0	F_1	F_2	F_3
0	0	0	1	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	1	1
0	1	1	0	1	0	1
1	0	0	1	1	0	0
1	0	1	0	0	0	1
1	1	0	1	1	1	1
1	1	1	0	1	0	1





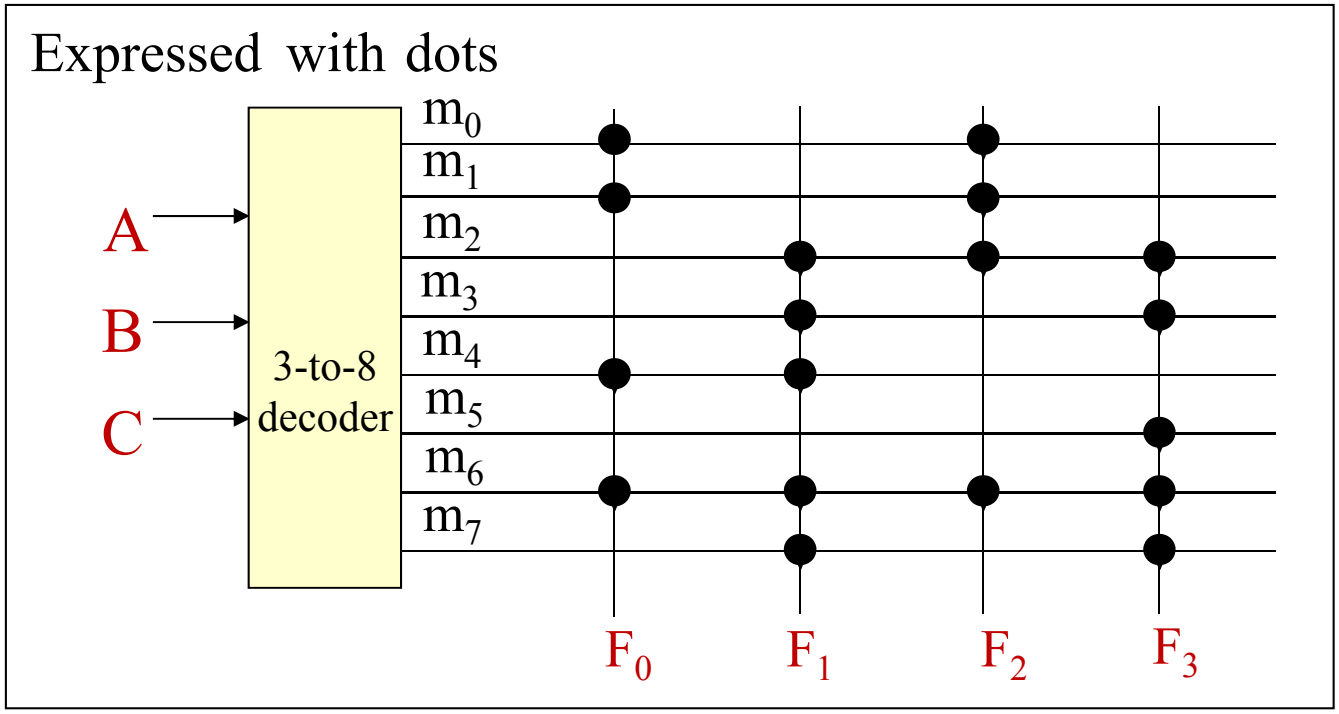


$$F_0 = m_0 + m_1 + m_4 + m_6$$

Read Only Memories (5/5)

$$F_0 = \sum m(0,1,4,6) \qquad F_1 = \sum m(2,3,4,6,7)$$

$$F_2 = \sum m(0,1,2,6) \qquad F_3 = \sum m(2,3,5,6,7)$$

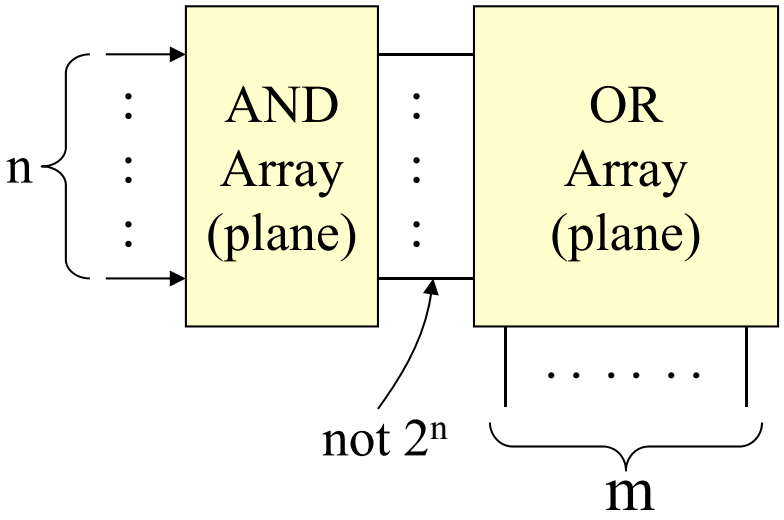


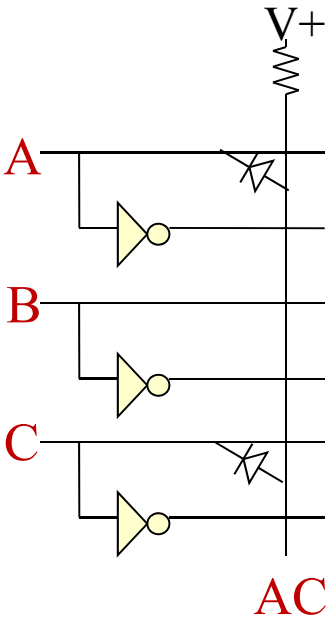
(1) Use "mask" to program ROM
(2) EPROM (Erasable Programmable) UV light
(3) EEPROM (Electrically Erasable)

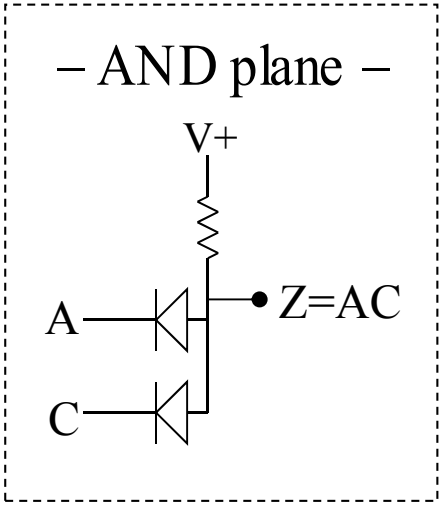
Programmable Logic Devices (1/7)

Various kinds: PLA , PAL

PLAs: n × m realizes m functions with n inputs







2-level SOP implementation

ROM directly implements truth table

If any of A, C is not activiated,

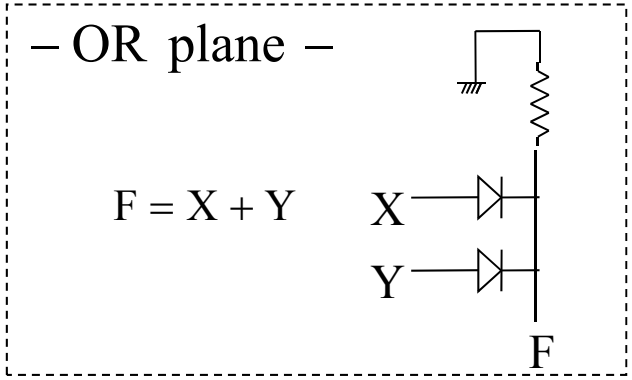
A = GND or C = GND

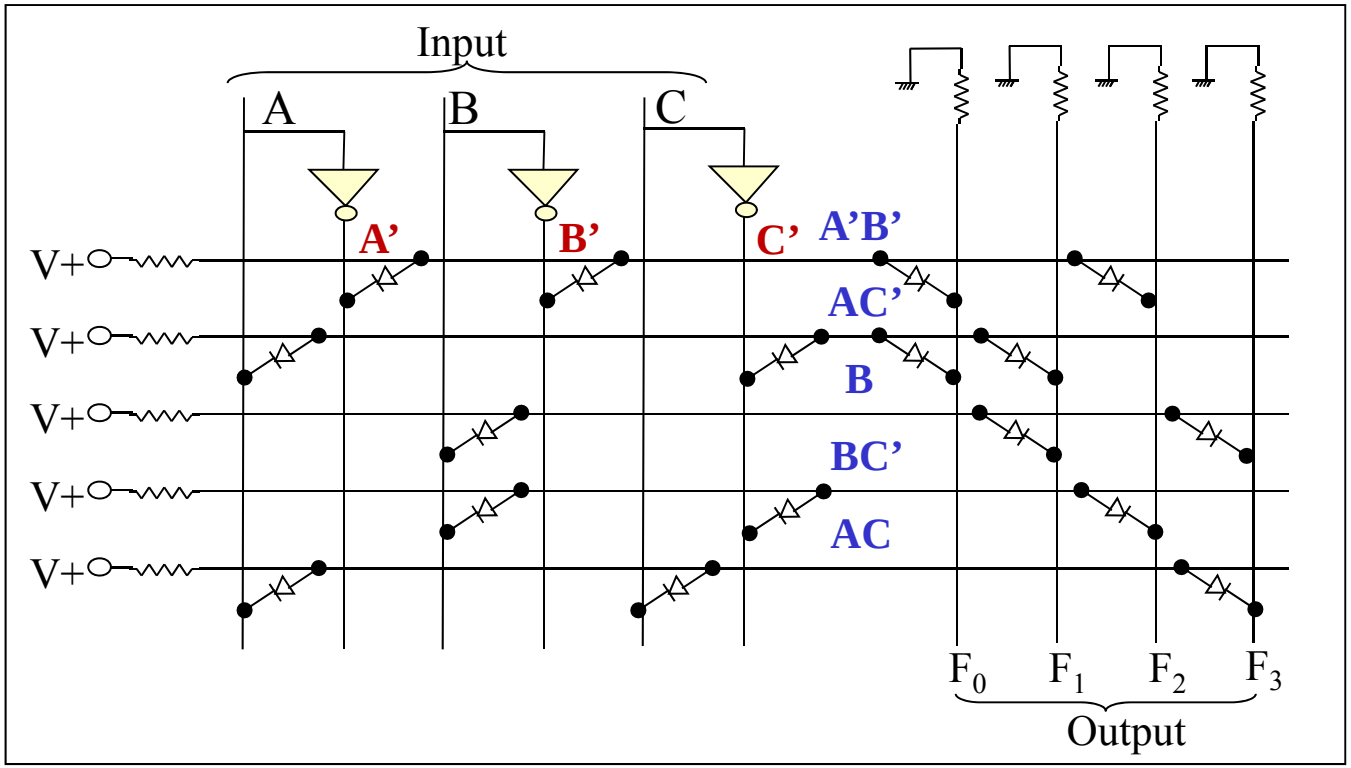
i.e., A = 0 or C = 0

Z=0

Programmable Logic Devices (2/7)

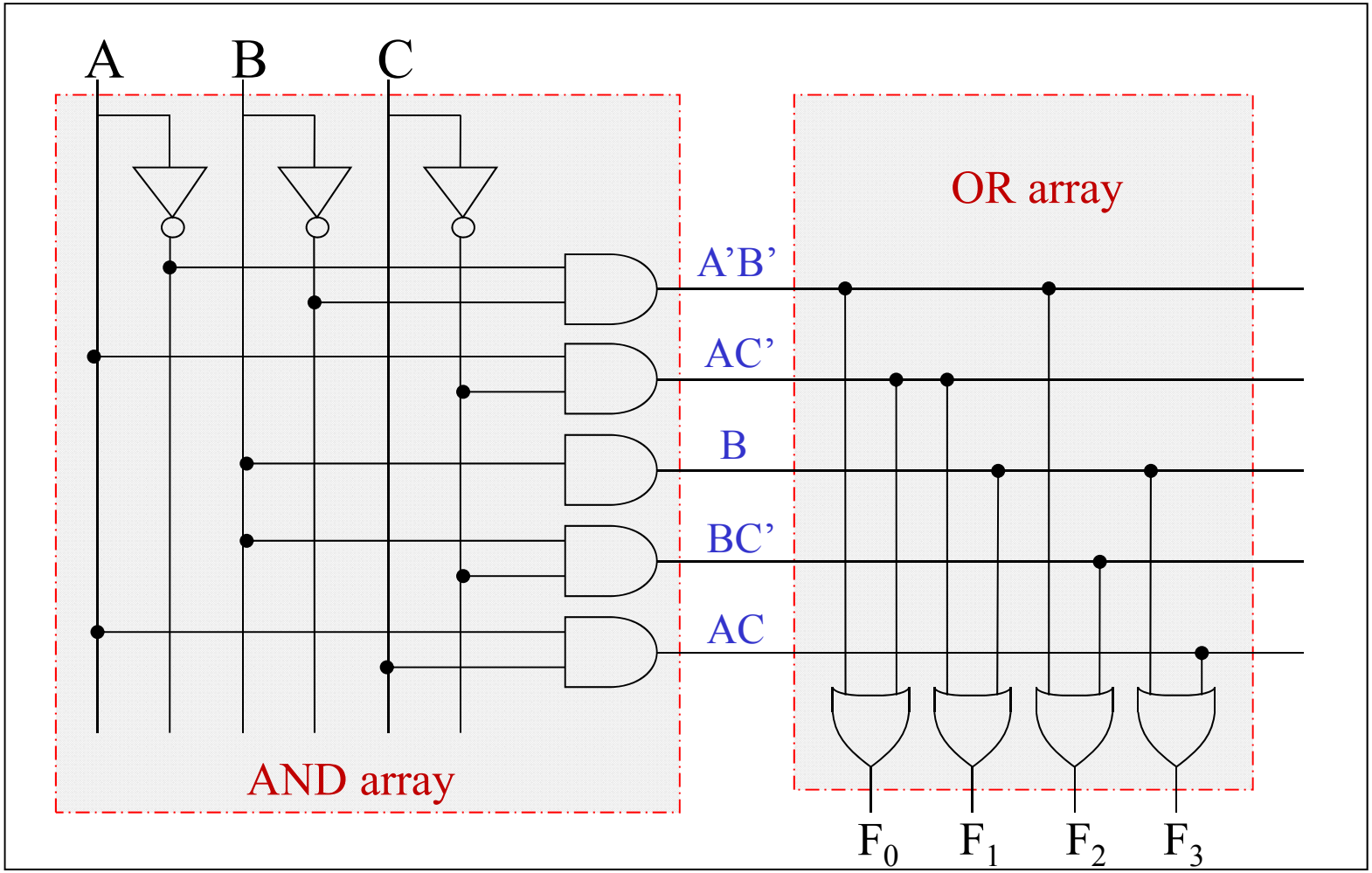
	A	B	C	F_0	F_1	F_2	F_3
A' B'	0	0	–	1	0	1	0
AC'	1	–	0	1	1	0	0
B	–	1	–	0	1	0	1
BC'	–	1	0	0	0	1	0
AC	1	–	1	0	0	0	1





Programmable Logic Devices (3/7)

Equivalent AND - OR Array



Programmable Logic Devices (4/7)

Example

$$F_1 = \sum m(2,3,5,7,8,9,10,11,13,15)$$

$$F_2 = \sum m(2,3,5,6,7,10,11,14,15)$$

$$F_3 = \sum m(6,7,8,9,13,14,15)$$

Minimized multiple output expressions (using K-Map)

$$F_1 = a'bd + abd + ab'c' + b'c$$

$$F_2 = c + a'bd$$

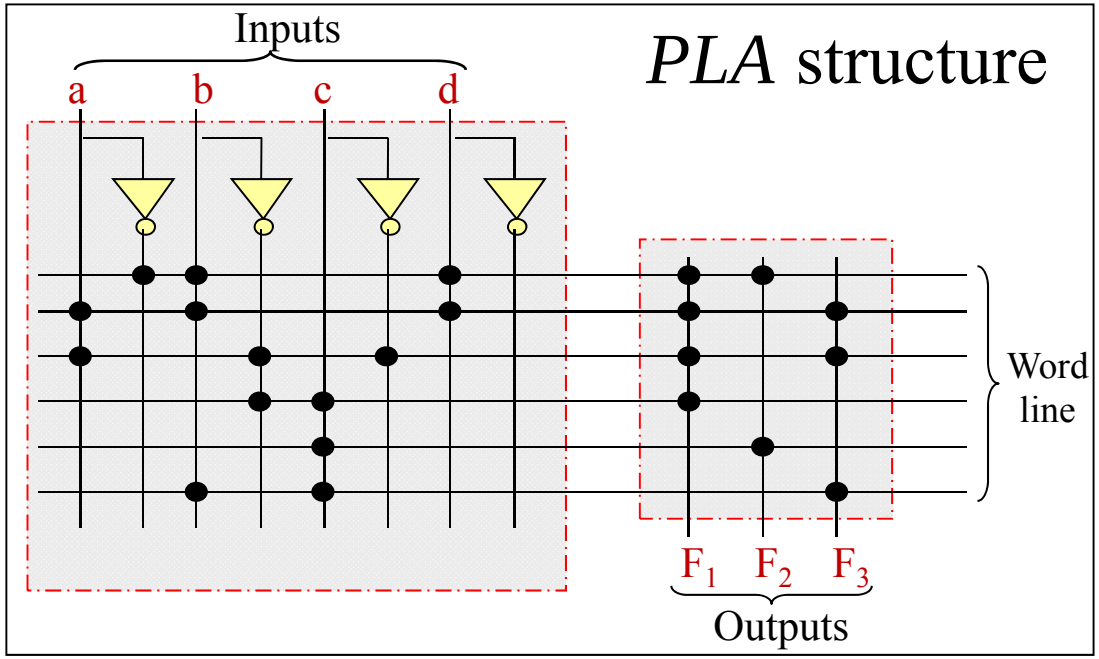
$$F_3 = bc + ab'c' + abd$$

PLA table

a	b	c	d	F_1	F_2	F_3
0	1	–	1	1	1	0
1	1	–	1	1	0	1
1	0	0	–	1	0	1
–	0	1	–	1	0	0
–	–	1	–	0	1	0
–	1	1	–	0	0	1

Programmable Logic Devices (5/7)

a	b	c	d	F_1	F_2	F_3
0	1	–	1	1	1	0
1	1	–	1	1	0	1
1	0	0	–	1	0	1
–	0	1	–	1	0	0
–	–	1	–	0	1	0
–	1	1	–	0	0	1



a b c d = 0 1 1 1 row 1,5,6 selected, $\Rightarrow F_1 = 1(1st) + 0(5th) + 0(6th) = 1$

$F_2 = 1 + 1 + 0 = 1$

$F_3 = 0 + 0 + 1 = 1$

= 0 0 0 1 no rows selected, $\Rightarrow F_1F_2F_3 = 000$

= 1 0 0 1 row 3 selected, $\Rightarrow F_1F_2F_3 = 101$

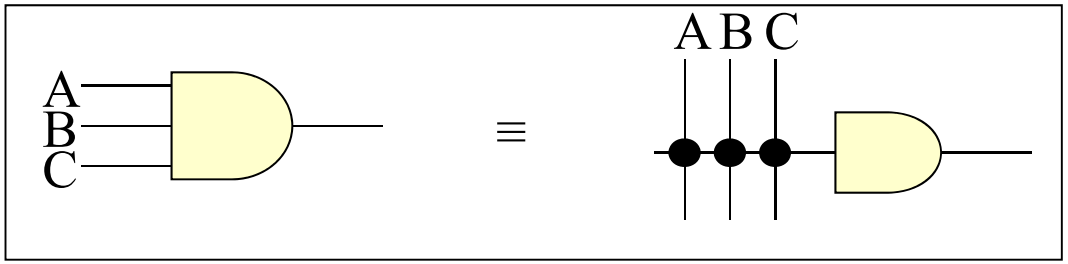
◎ Mask programmable PLA

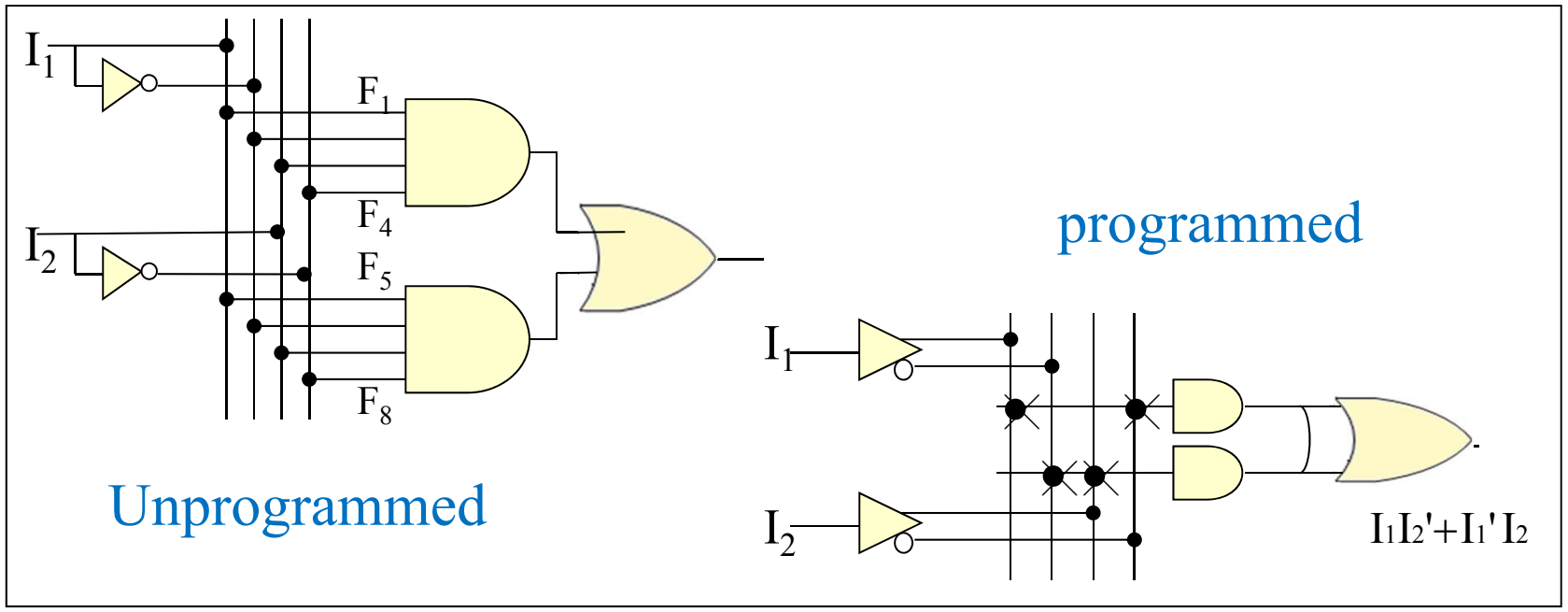
◎ Field programmable PLA

Programmable Logic Devices (6/7)

PALs: n × m realizes m functions with n inputs

A special case of PLA: (AND array programmable
OR array NOT programmable)



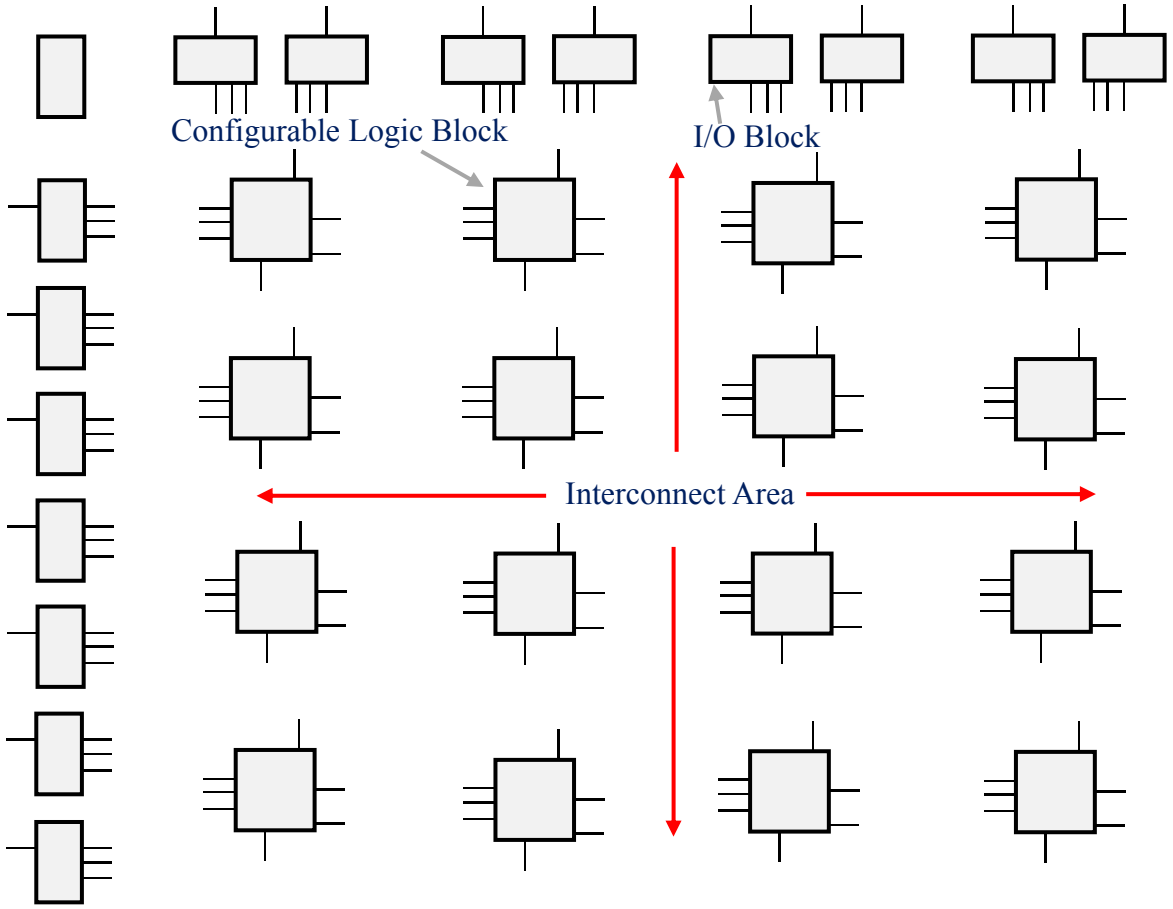


PLD Summary

	AND Array	OR Array
PLA	Programmable	Programmable
PAL	Programmable	Fixed
ROM	Fixed	Programmable
Not Programmable	Fixed	Fixed

Field Programmable Gate Arrays (1/4)

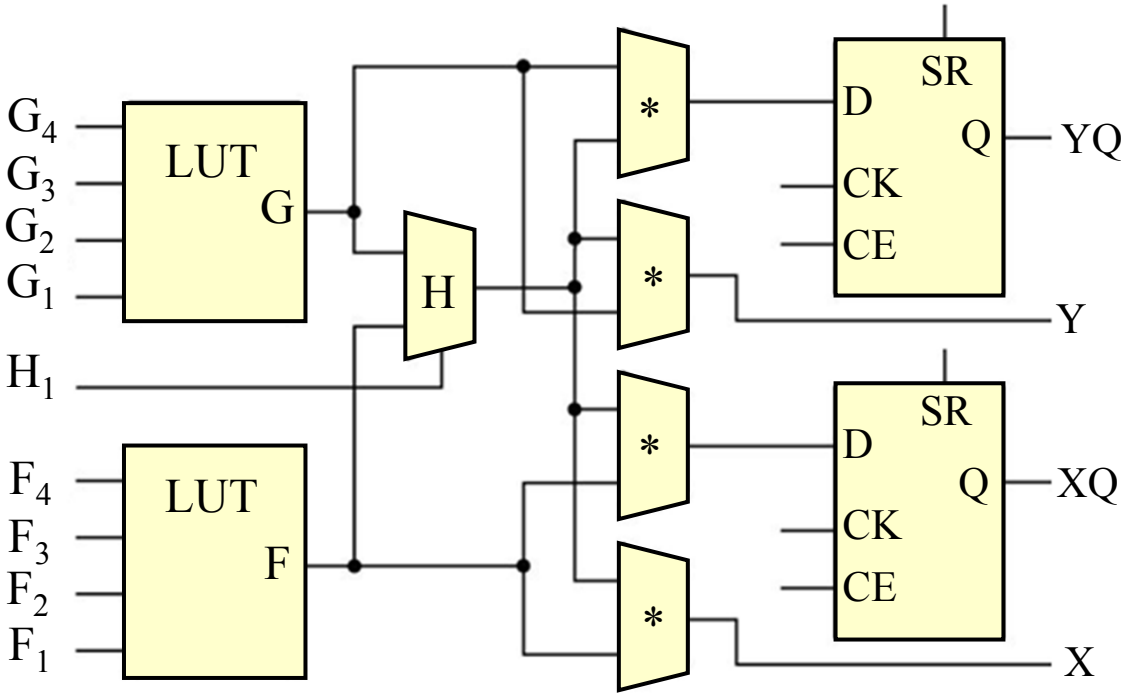
An IC that contains an array of identical logic cells with programmable interconnections



Layout of a typical FPGA

Field Programmable Gate Arrays (2/4)

CLB: configurable logic block



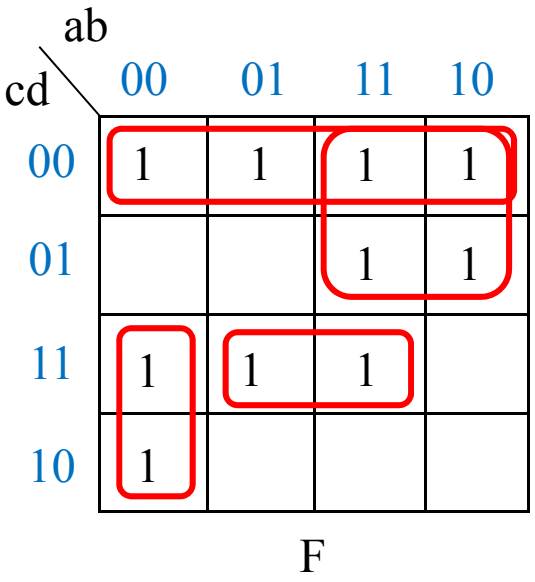
* = Programmable MUX

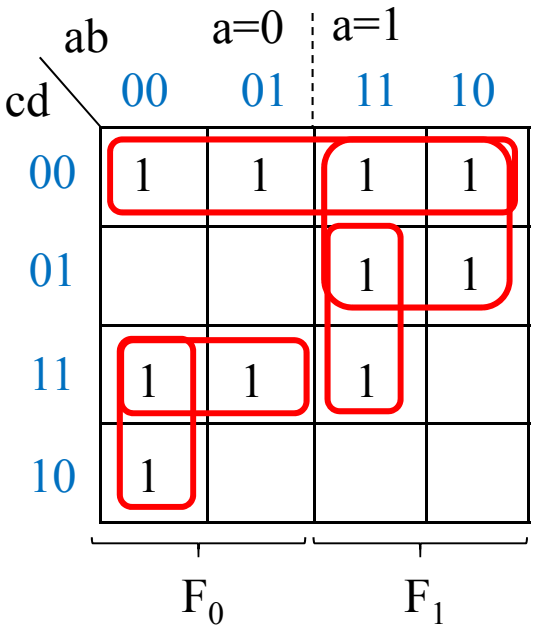
Field Programmable Gate Arrays (3/4)

Decomposition of switching functions

Shannon's expansion theorem

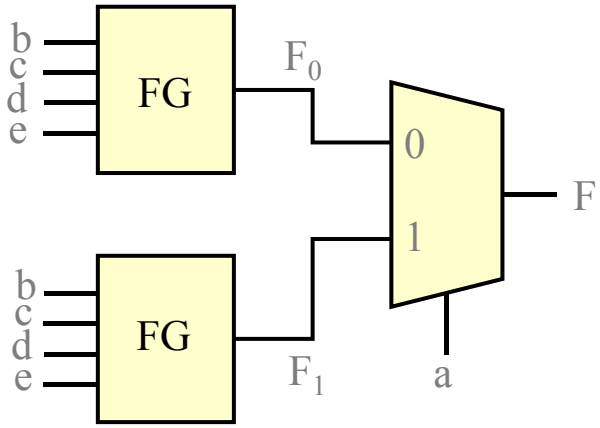
$$F(x_1, x_2, \ldots x_{i-1}, x_i, x_{i+1}, \ldots) = x_i' F_0 + x_i F_1$$



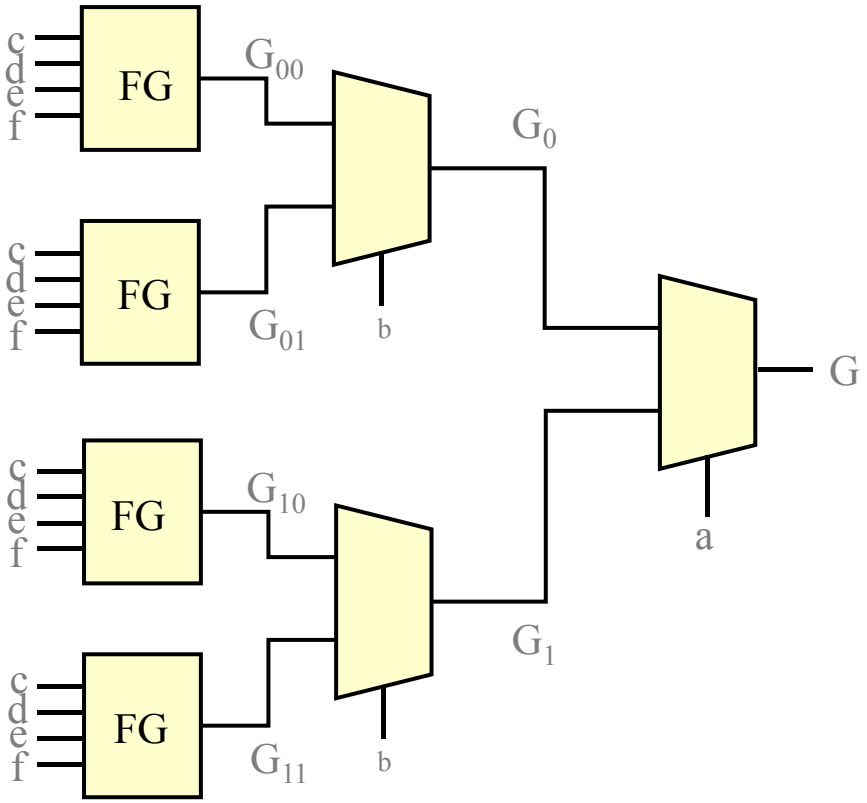


Field Programmable Gate Arrays (4/4)

Realization of 5- and 6-variable functions with 4-variable function generators (FG)



(a) 5-variable function



(b) 6-variable function