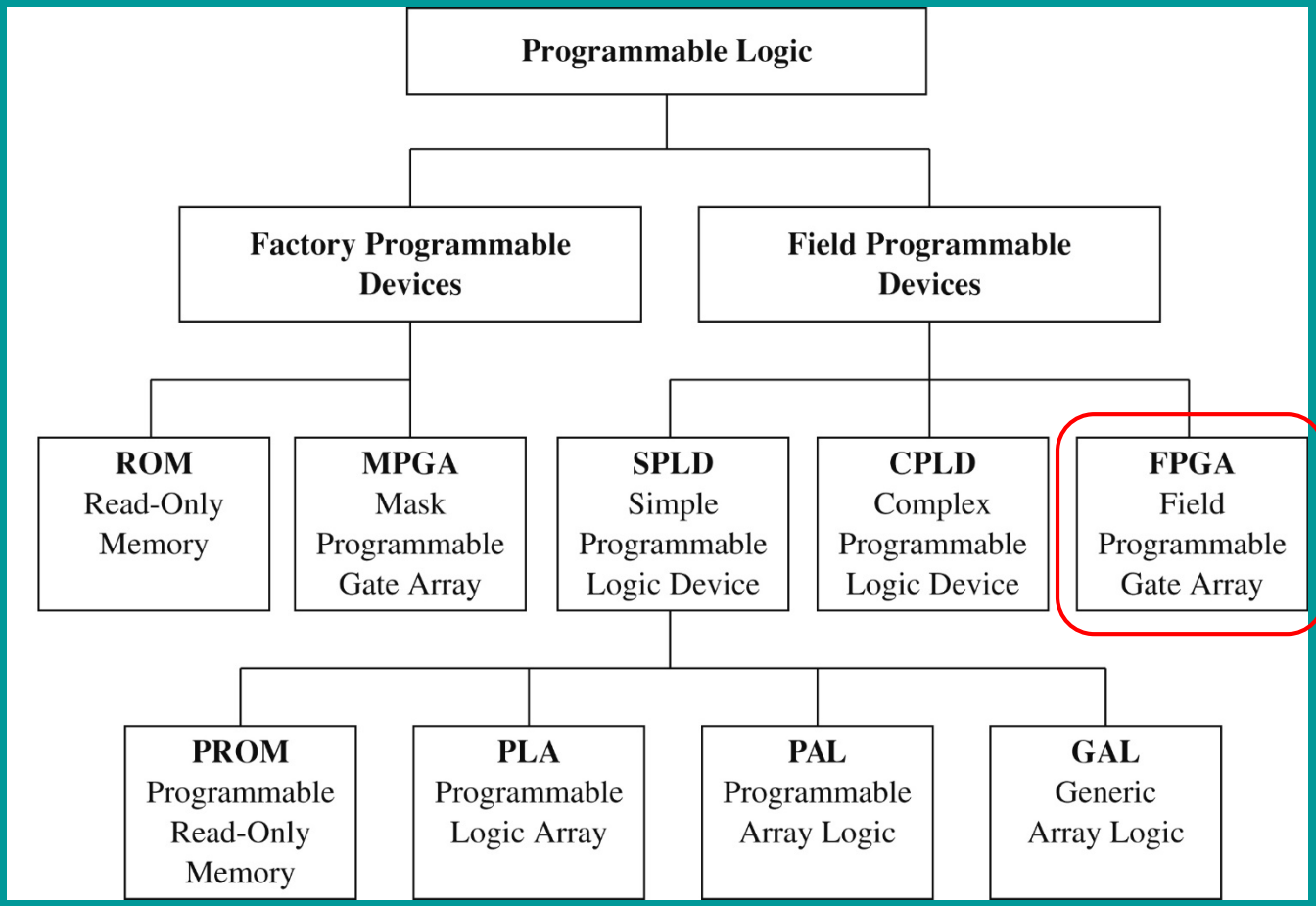
Programmable Logic Devices

Roth Text: Chapter 3 (sections 3.1-3.3)
Nelson Text: Chapter 5 (combinational)
Chapter 11 (sequential)

Programmable logic taxonomy



Lab Device

TABLE 3-1: A Comparison of Programmable Devices

	SPLD	CPLD	FPGA
Density	Low Few hundred gates	Low to Medium 500 to 12,000 gates	Medium to High 3,000 to 5,000,000 gates
Timing	Predictable	Predictable	Unpredictable
Cost	Low	Low to Medium	Medium to High
Major Vendors	Lattice Semiconductor Cypress AMD	Xilinx Altera	Xilinx Altera Lattice Semiconductor Actel
Example Device Families	**Lattice Semiconductor** GAL16LV8 GAL22V10 **Cypress** PALCE16V8 **AMD** 22V10	**Xilinx** CoolRunner XC9500 **Altera** MAX	**Xilinx** Virtex Spartan **Altera** Stratix **Lattice** Mach ECP **Actel** Accelerator

History of Programmable Logic

- Programmable Logic Arrays ~ 1970
  - Incorporated in VLSI devices
  - Can implement any set of SOP logic equations
    - Outputs can share common product terms
- Programmable Logic Devices ~ 1980
  - MMI Programmable Array Logic (PAL)
    - 16L8 – combinational logic only
      - 8 outputs with 7 programmable PTs of 16 input variables
    - 16R8 – sequential logic only
      - 8 registered outputs with 8 programmable PTs of 16 input variables
  - Lattice 16V8
    - 8 outputs with 8 programmable PTs of 16 input variables
      - Each output programmable to use or bypass flip-flop
  - Complex PLDs – arrays of PLDs with routing network
- Field Programmable Gate Arrays ~ 1985
  - Xilinx Logic Cell Array (LCA)
- CPLD & FPGA architectures became similar ~ 2000
  - Incorporation of RAMs and other specialized cores
    - Programmable system-on-chip

Programming Technologies

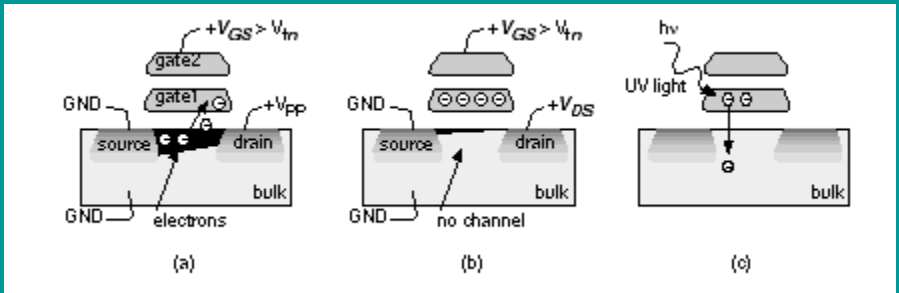
- PLAs were mask programmable
- PALs used fuses for programming
- Early PLDs & CPLDs used floating gate technology
  - Erasable Programmable Read Only Memory (EPROM)
    - Ultra-violet erasable (UVEPROM)
    - Electrically erasable (EEPROM)
    - Flash memory came later and was used for CPLDs
- FPGAs used RAM for programming
- Later trends
  - Fuses were replaced with anti-fuses
    - Better reliability
  - Large CPLDs went to RAM-based programming

Programming Technologies

- PLAs were mask programmable
- PALs used fuses for programming
- Early PLDs & CPLDs used floating gate technology
  - Erasable Programmable Read Only Memory (EPROM)
    - Ultra-violet erasable (UVEPROM)
    - Electrically erasable (EEPROM)
    - Flash memory came later and was used for CPLDs
- FPGAs used RAM for programming
- Later trends
  - Fuses were replaced with anti-fuses
    - Better reliability
  - Large CPLDs went to RAM-based programming

Programming Technologies

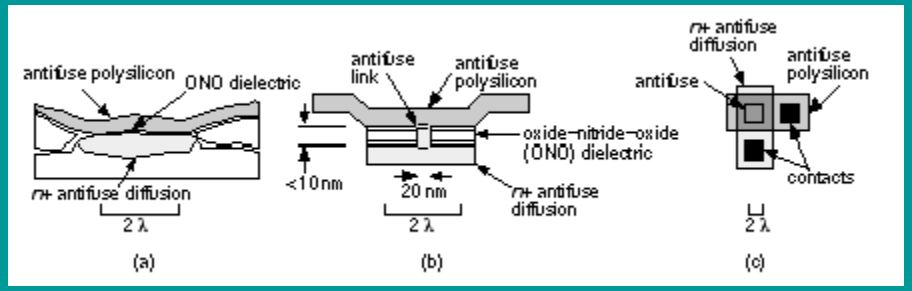
- Floating gate technologies
  - Non-volatile but re-usable
    - UV EPROM, EEPROM, and flash memory
  - In-System Programmable (ISP)
    - EEPROM and flash memory
  - In-System Re-programmable (ISR)
    - Flash memory



EPROM transistor – Programming voltage forces charge into floating gate.
(trapped until given energy to escape)
[Altera EPLDs, Xilinx EPLDs]

Programming Technologies

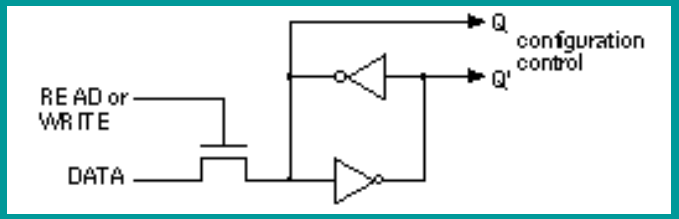
- Fuse/anti-fuse
  - Non-volatile but not re-usable
  - One Time Programmable (OTP)



Antifuse - Open until programming current melts dielectric (one time only)
[Actel PLICE©]

Programming Technologies

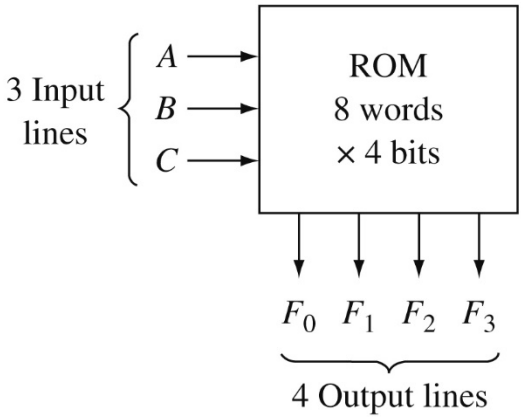
- RAM
  - Volatile – must configure after power-up
  - In-System Re-programmable (ISR)
  - Run-Time Reconfiguration (RTR)
    - dynamic reconfiguration while system is operating



SRAM cell

Logic functions in read-only memory (ROM)

FIGURE 3-2:
An 8-Word × 4-Bit ROM



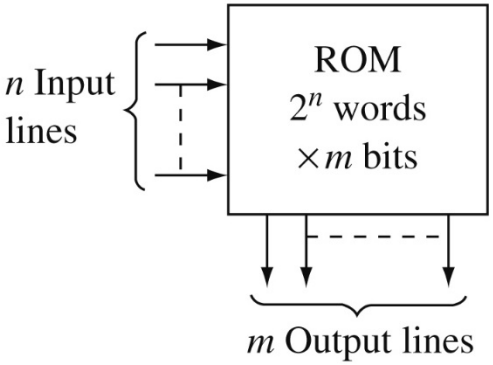
(a) Block diagram

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

Typical data stored in ROM (2^3 words of 4 bits each)

(b) Truth table for ROM

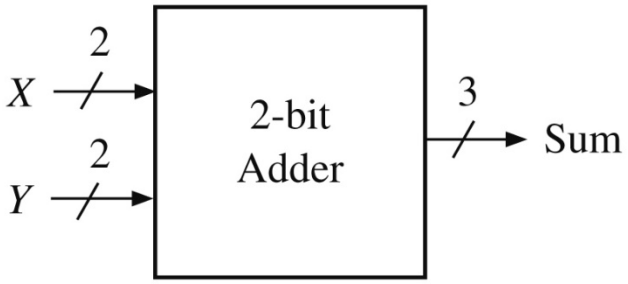
FIGURE 3-3:
Read-Only Memory with *n* Inputs and *m* Outputs



n Input Variables	*m* Output Variables
00 · · · 00	100 · · · 110
00 · · · 01	010 · · · 111
00 · · · 10	101 · · · 101
00 · · · 11	110 · · · 010
⋮	⋮
11 · · · 00	001 · · · 011
11 · · · 01	110 · · · 110
11 · · · 10	011 · · · 000
11 · · · 11	111 · · · 101

Typical data array stored in ROM (2^n words of *m* bits each)

FIGURE 3-4: Block Diagram and Truth Table of a 2-Bit Adder



X_1	X_0	Y_1	Y_0	S_2	S_1	S_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

FIGURE 3-5: ROM Implementation of a 2-Bit Full Adder

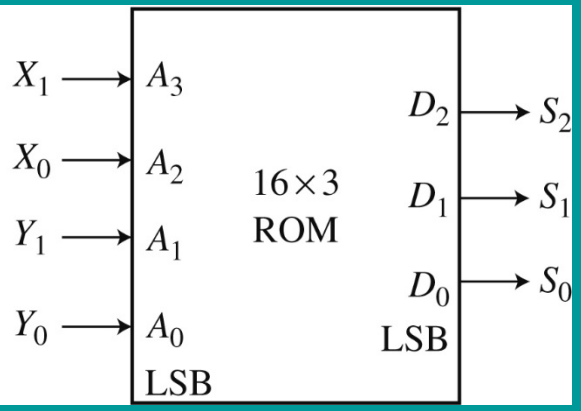
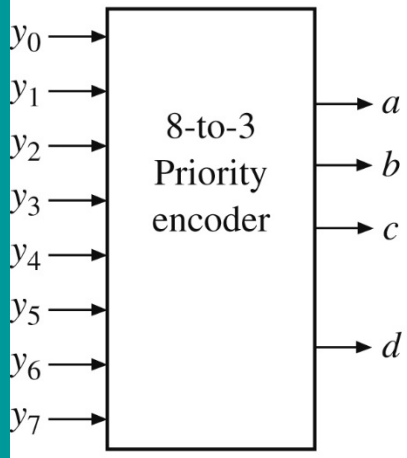


FIGURE 3-6: 8-to-3 Priority Encoder

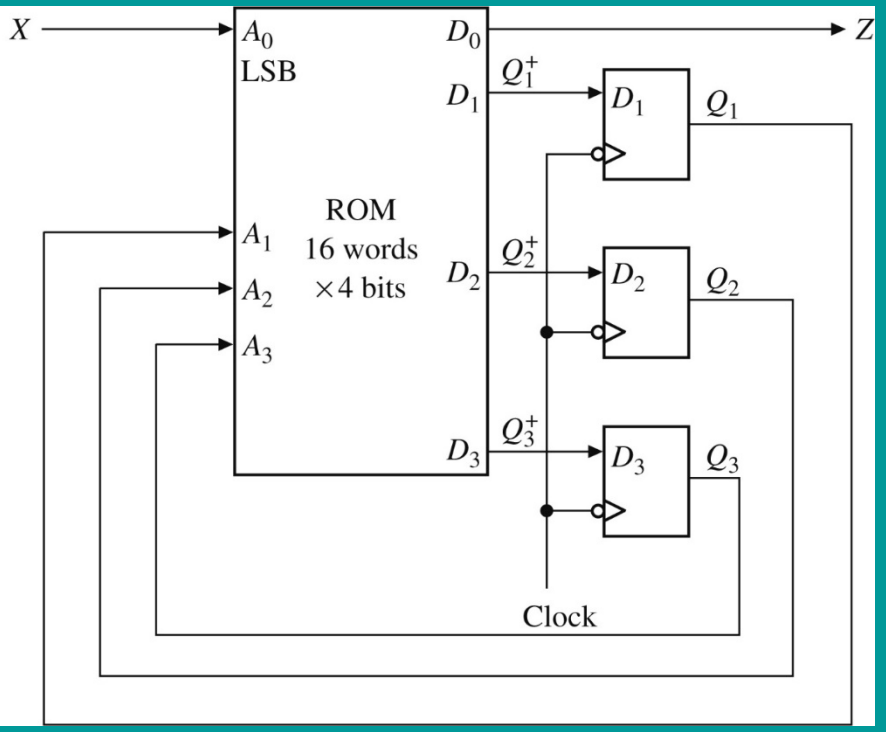


y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	a	b	c	d
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
X	1	0	0	0	0	0	0	0	0	1	1
X	X	1	0	0	0	0	0	0	1	0	1
X	X	X	1	0	0	0	0	0	1	1	1
X	X	X	X	1	0	0	0	1	0	0	1
X	X	X	X	X	1	0	0	1	0	1	1
X	X	X	X	X	X	1	0	1	1	0	1
X	X	X	X	X	X	X	1	1	1	1	1

FIGURE 3-7:
State Table for a Sequential Circuit

PS	NS $X=0$	NS $X=1$	Z $X=0$	Z $X=1$
S_0	S_1	S_2	1	0
S_1	S_3	S_4	1	0
S_2	S_4	S_4	0	1
S_3	S_5	S_5	0	1
S_4	S_5	S_6	1	0
S_5	S_0	S_0	0	1
S_6	S_0	—	1	—

FIGURE 3-8:
Realization of a Mealy Sequential Circuit with a ROM



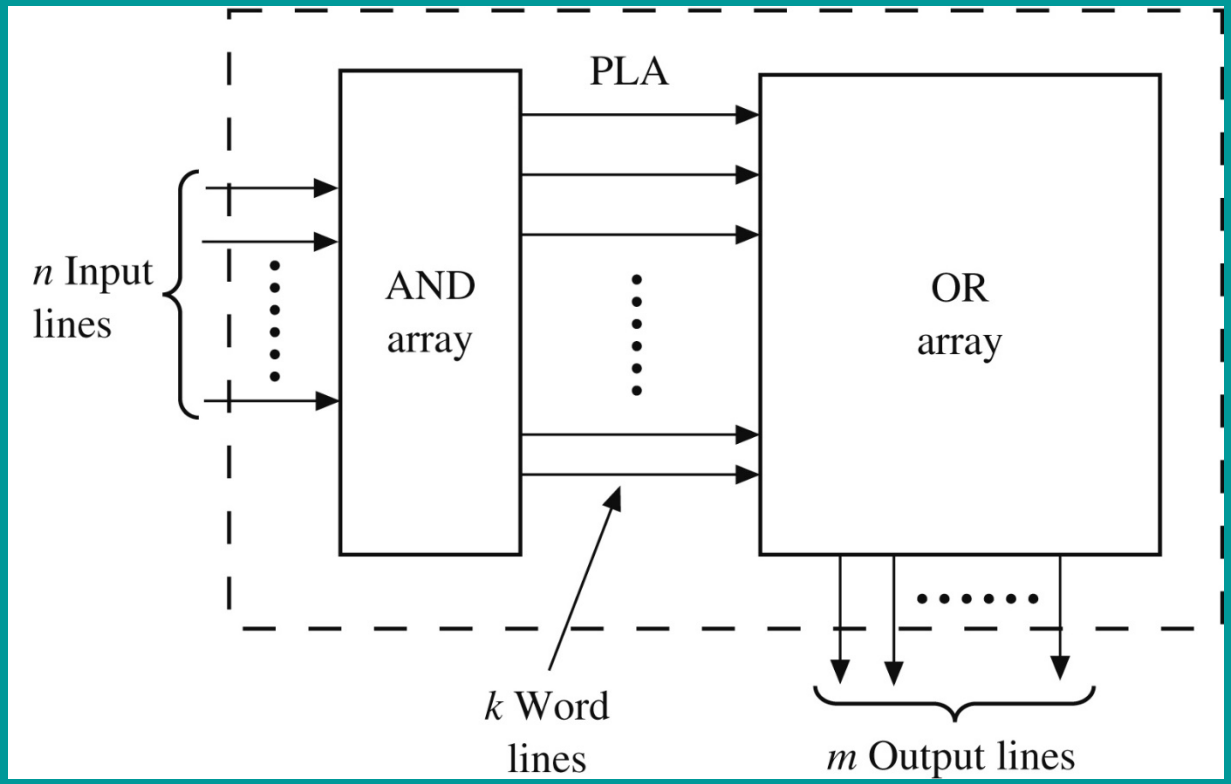
ROM on next slide

TABLE 3-2: ROM Truth Table

Q_3	Q_2	Q_1	X	Q_3^+	Q_2^+	Q_1^+	Z
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	1	1
0	0	1	1	1	0	0	0
0	1	0	0	1	0	0	0
0	1	0	1	1	0	0	1
0	1	1	0	1	0	1	0
0	1	1	1	1	0	1	1
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1
1	1	0	0	0	0	0	1
1	1	0	1	0	0	0	0
1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	0

Programmable logic array structure

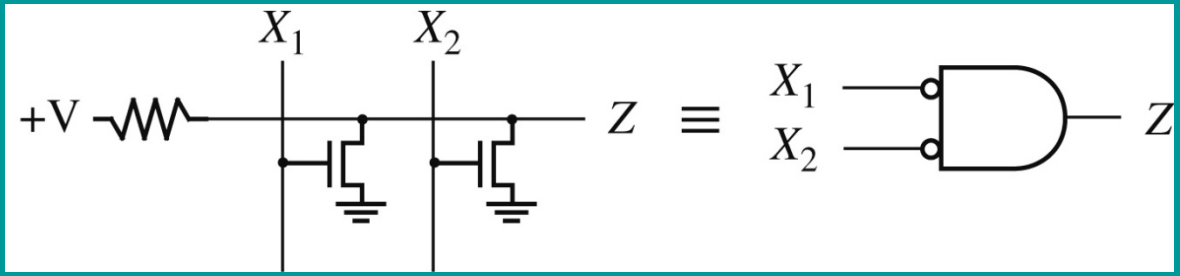
Implement sum of products logic expressions



Each one "product" of the inputs

Each one "sum" of the products

NOR function in programmable logic



Xi = 0 turns transistor OFF (transistor = open circuit)
Xi = 1 turns transistor ON (transistor shorts Z to ground/0)
+V pulls Z up to 1 if not shorted to ground

Truth Table:

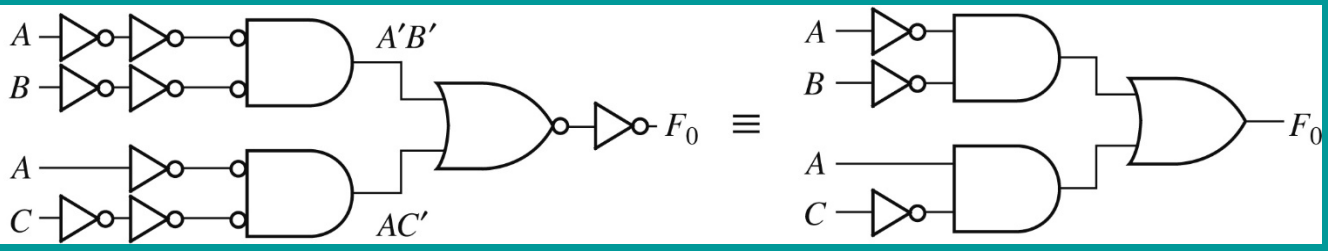
X1	X2	Z	
0	0	1	- both transistors OFF/Z pulled up to +V
0	1	0	- transistor 2 ON/shorts Z to ground
1	0	0	- transistor 1 ON/shorts Z to ground
1	1	0	- both transistors ON/short Z to ground

Manipulate sum of products form to use NOR-NOR structures

PLA implements SOP forms

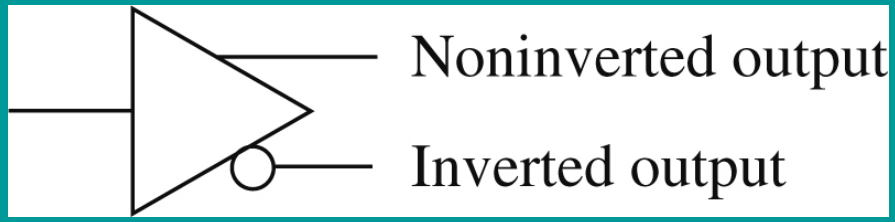
$$F_0 = \bar{A}\bar{B} + A\bar{C}$$

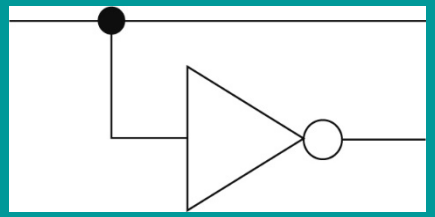
FIGURE 3-13:
Conversion of NOR-NOR to AND-OR



products sum

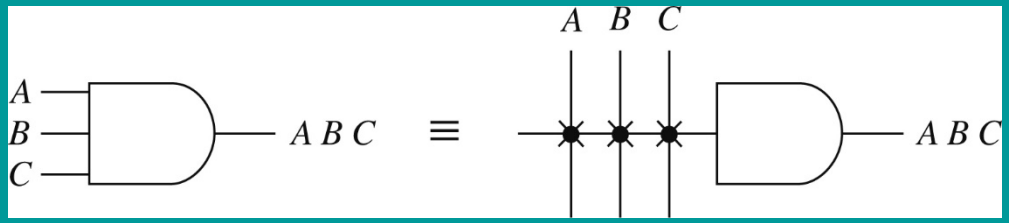
Equivalent AND-OR form



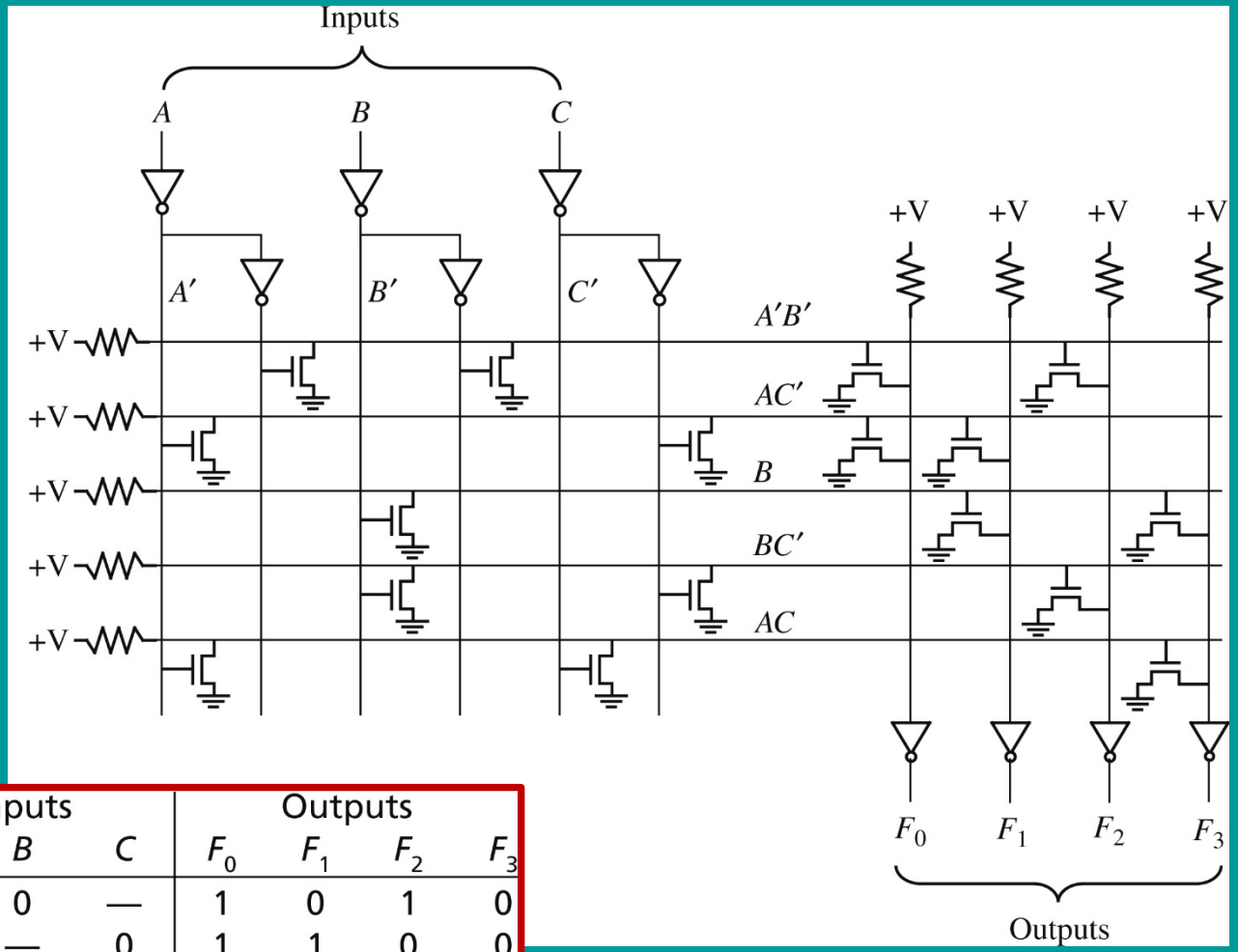


Inputs and complements typically provided in PLA

Compact representation of product

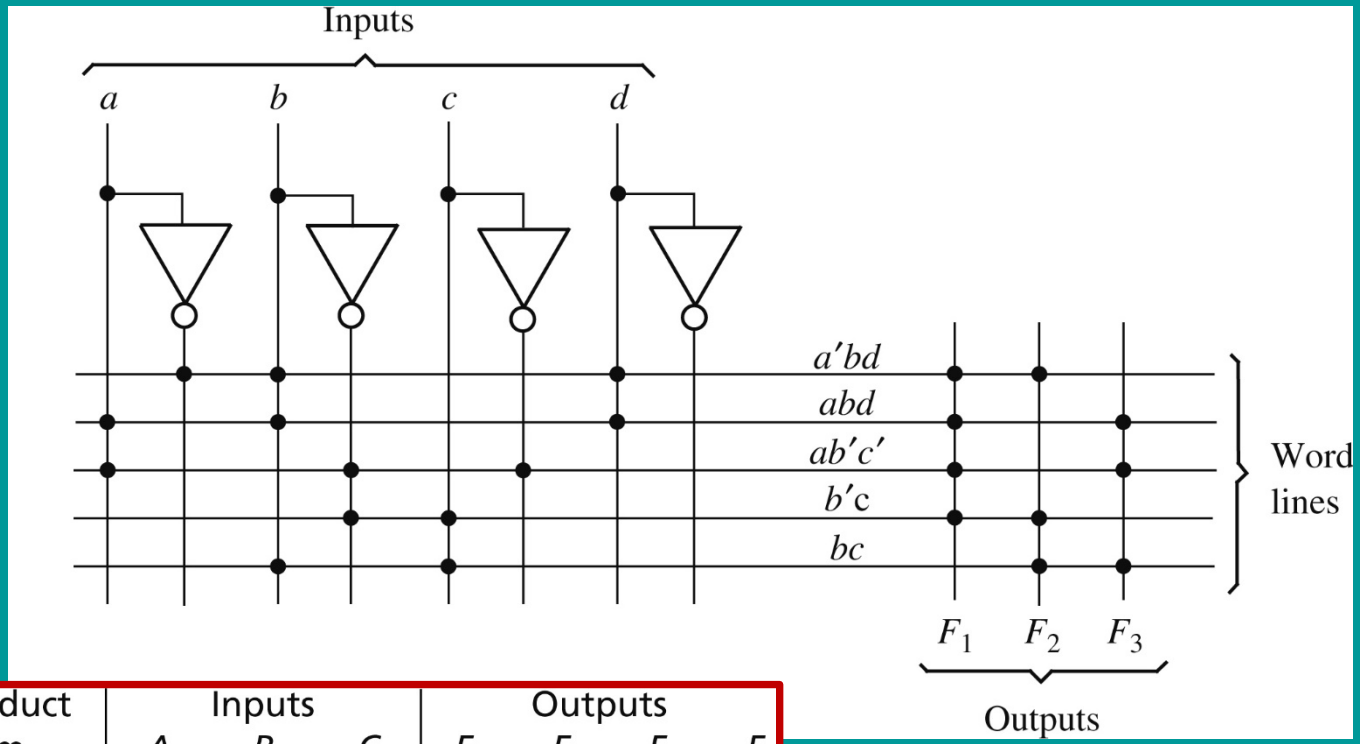


PLA with 3 inputs/5 products/4 sums



Product Term	Inputs A	B	C	Outputs 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

Compact representation of previous PLA circuit



Product Term	Inputs A	B	C	Outputs 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

Minimize multiple functions concurrently
(minimize # product terms)

FIGURE 3-14:
Multiple-Output Karnaugh Maps

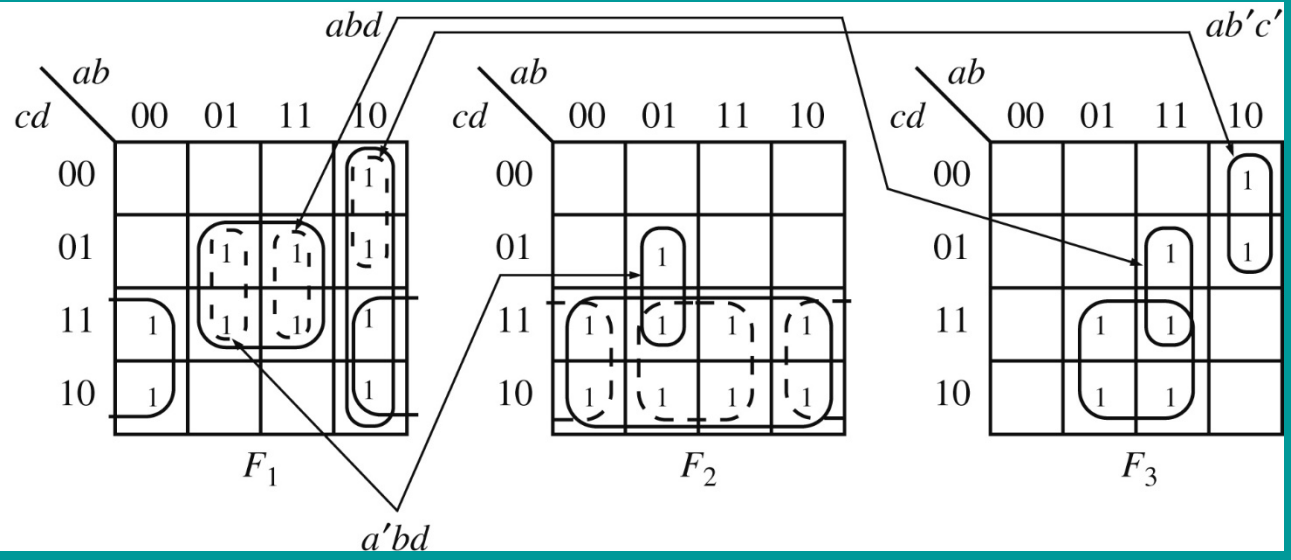
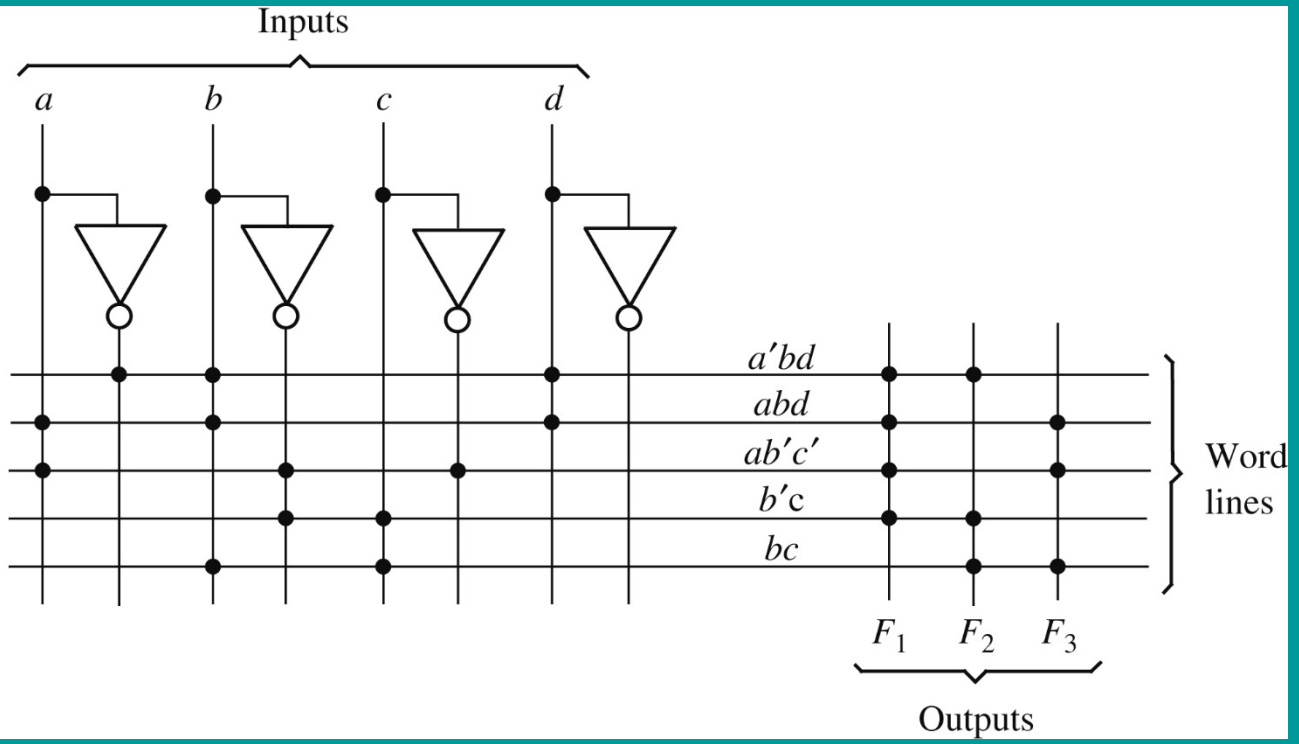


TABLE 3-4:
Reduced PLA Table

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	1	0
—	1	1	—	0	1	1

FIGURE 3-15:
PLA Realization of Equations (3-4)



Sequential circuit implementation

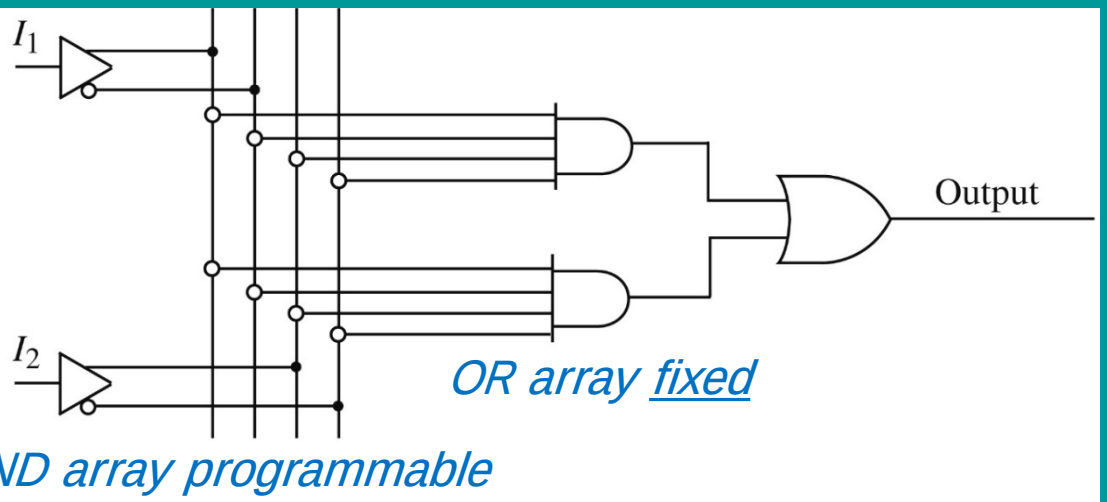
BCD to Excess-3 Converter

TABLE 3-5: PLA Table

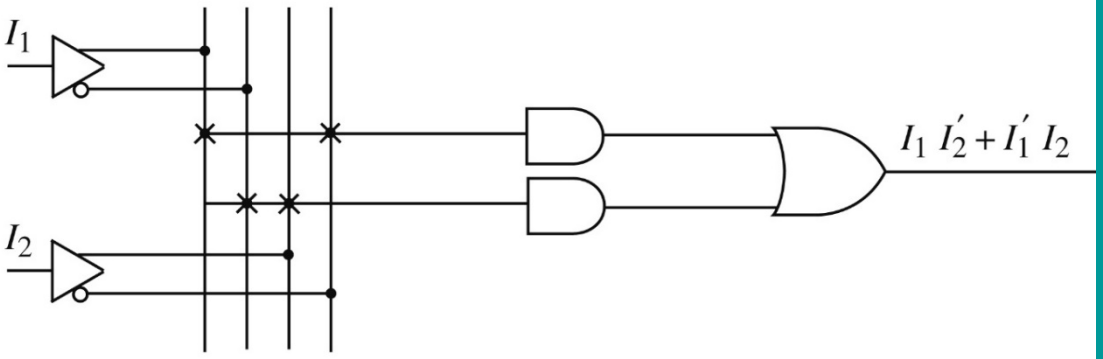
Product Term	Q_1	Q_2	Q_3	X	Q_1^+	Q_2^+	Q_3^+	Z
Q_2'	—	0	—	—	1	0	0	0
Q_1	1	—	—	—	0	1	0	0
$Q_1Q_2Q_3$	1	1	1	—	0	0	1	0
$Q_1Q_3'X'$	1	—	0	0	0	0	1	0
$Q_1'Q_2'X$	0	0	—	1	0	0	1	0
$Q_3'X'$	—	—	0	0	0	0	0	1
Q_3X	—	—	1	1	0	0	0	1

Programmable Array Logic (PAL)

FIGURE 3-16:
PAL Segment



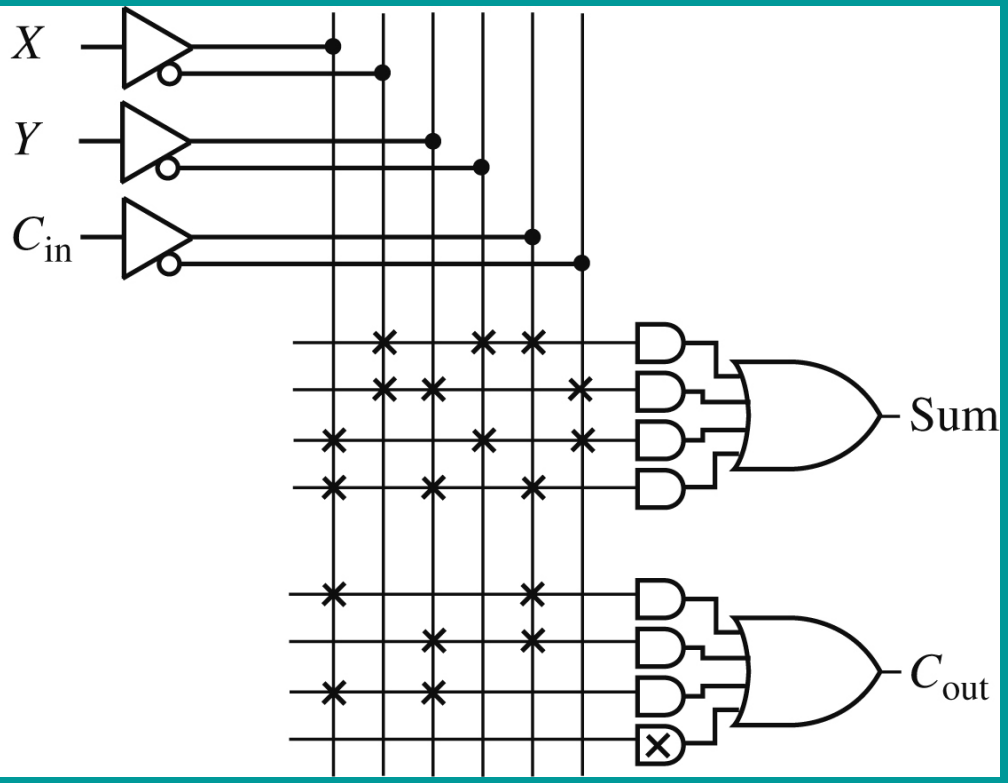
(a) Unprogrammed



(b) Programmed

Full adder with a PAL

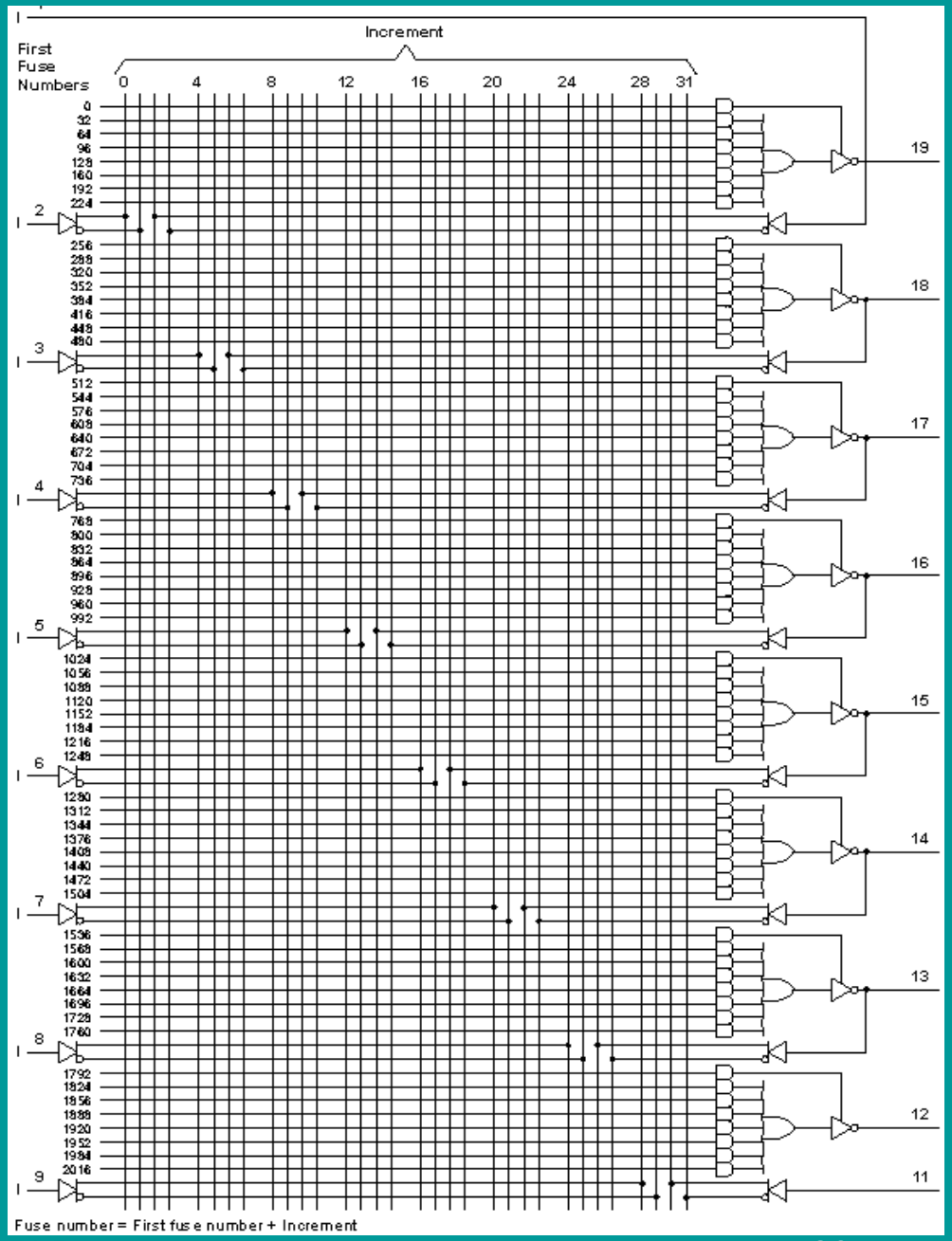
FIGURE 3-17:
Implementation of a Full Adder Using a PAL



PALs

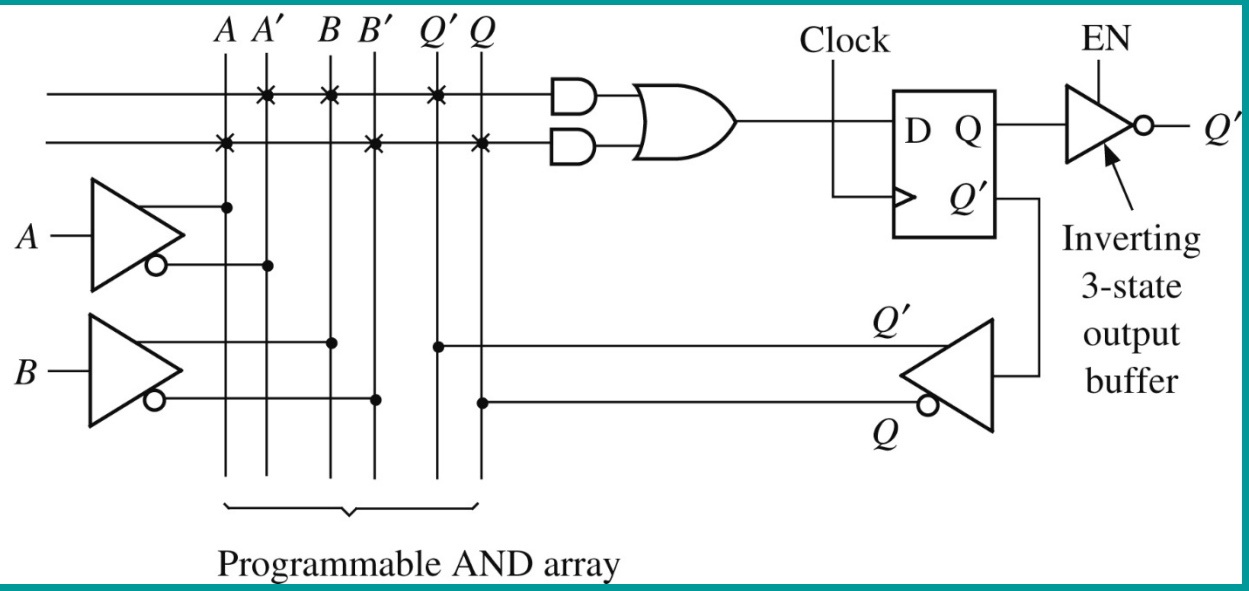
16L8 – combinational logic

- 10 to 16 inputs, each with true and complement signal
- 2 to 8 outputs, each with
  - 7 product terms can AND any of up to 16 inputs or their complements
  - Tri-state control product term for inverting output buffer
    - When output in tri-state, I/O pin can be used as input
      - High impedance output with no signal driven

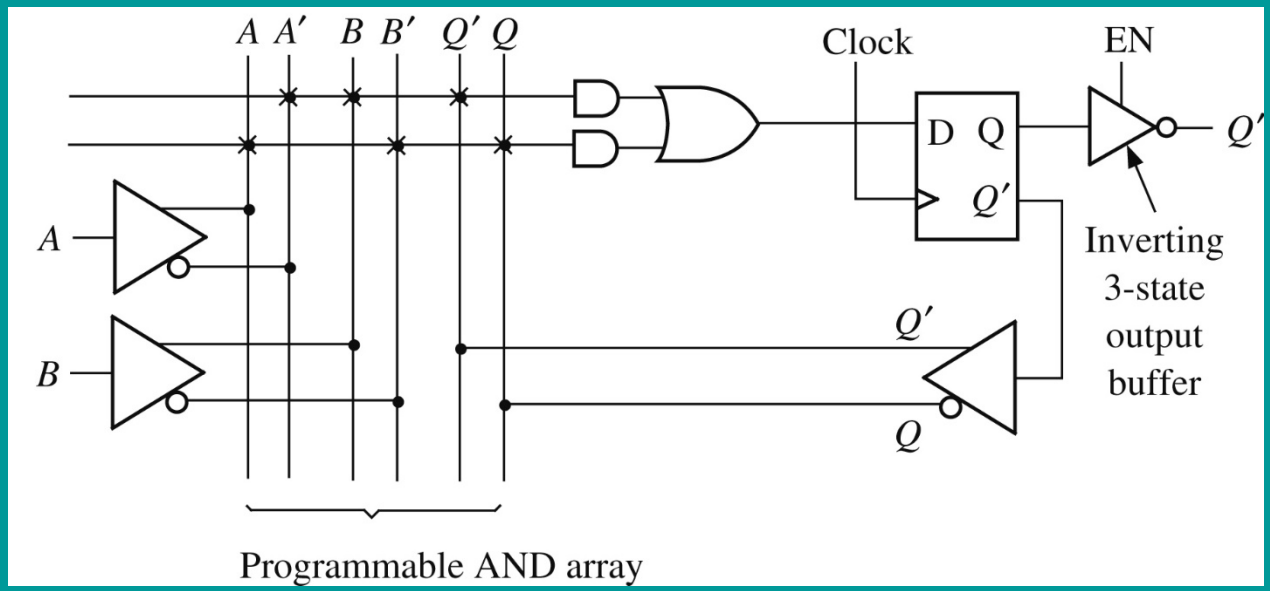


Sequential PAL

FIGURE 3-18:
Segment of a Sequential PAL



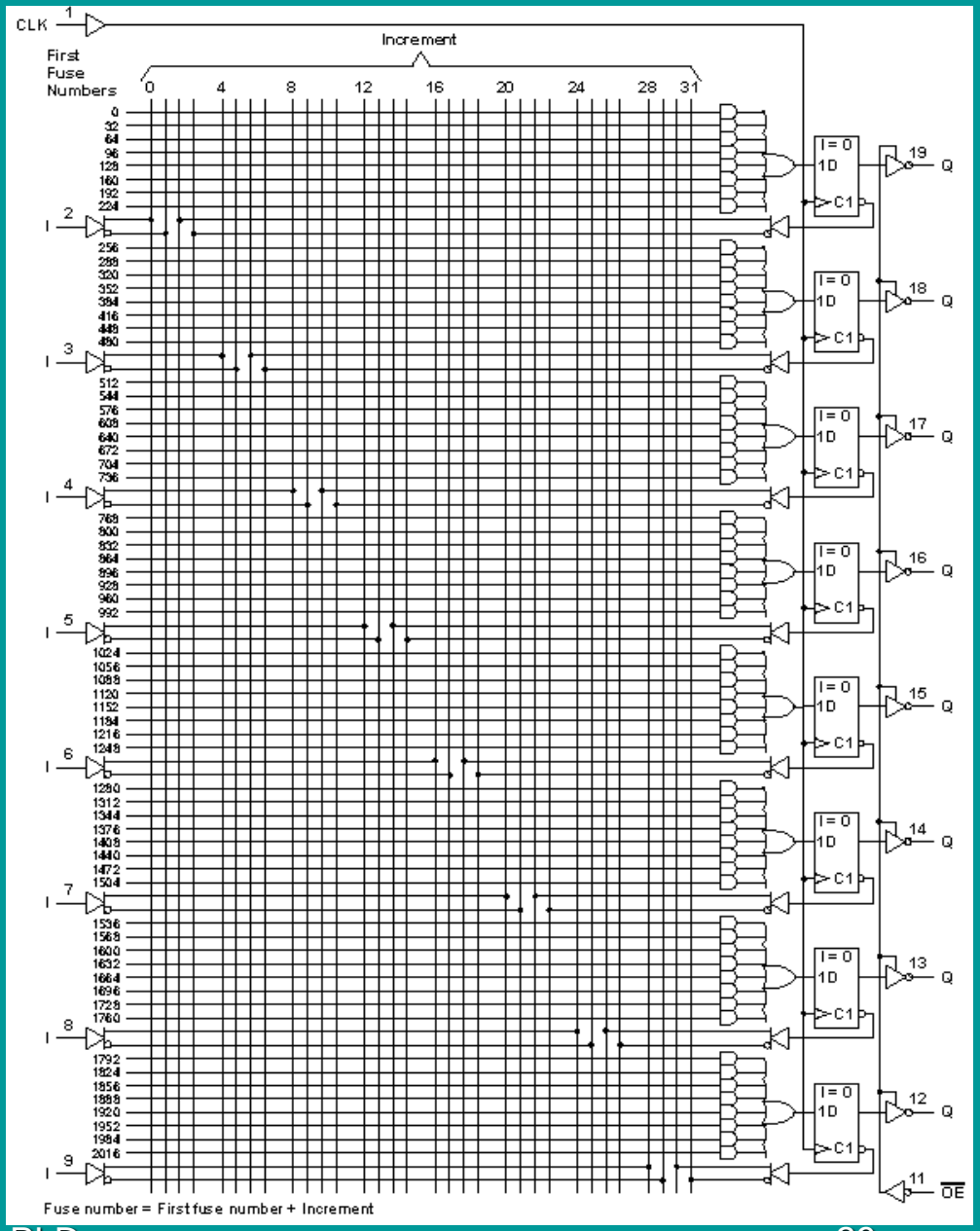
Sequential circuit with a PAL



PALs

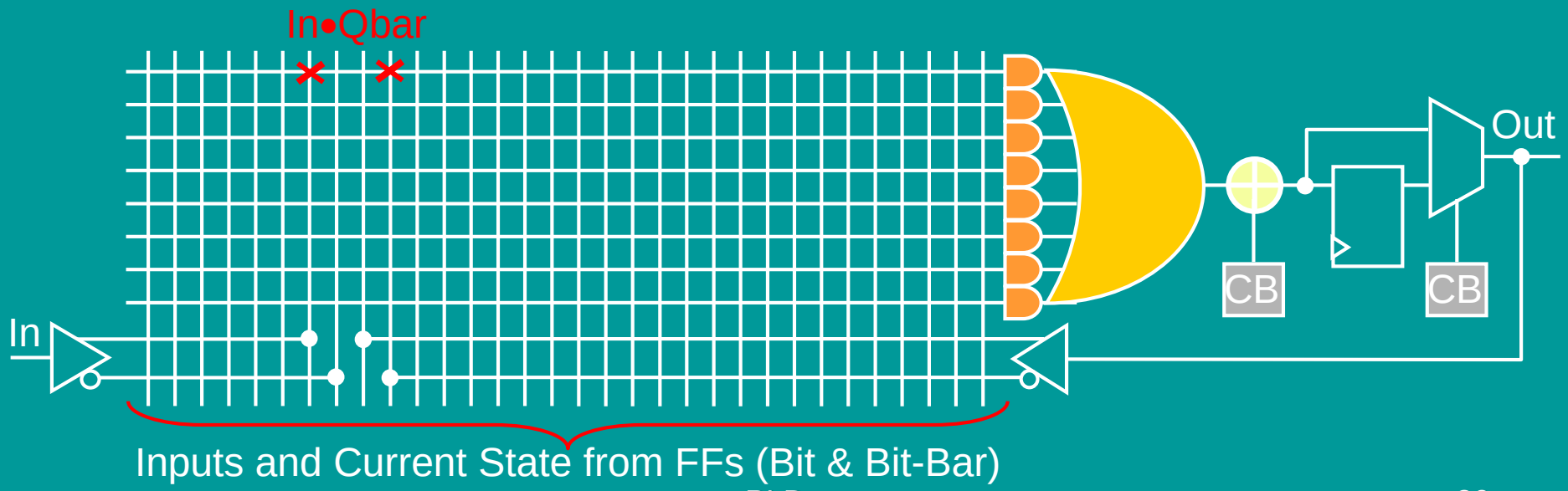
16R8 – sequential logic

- 8 inputs, each with true & complement
- 8 outputs, each with
  - D flip-flop
    - With feedback for FSMs
  - 8 product terms that can AND any of:
    - 8 inputs or their complements
    - 8 feedbacks or their complements from D flip-flops
- One clock for all FFs
- One tri-state control for all outputs



PLD Basic Structure

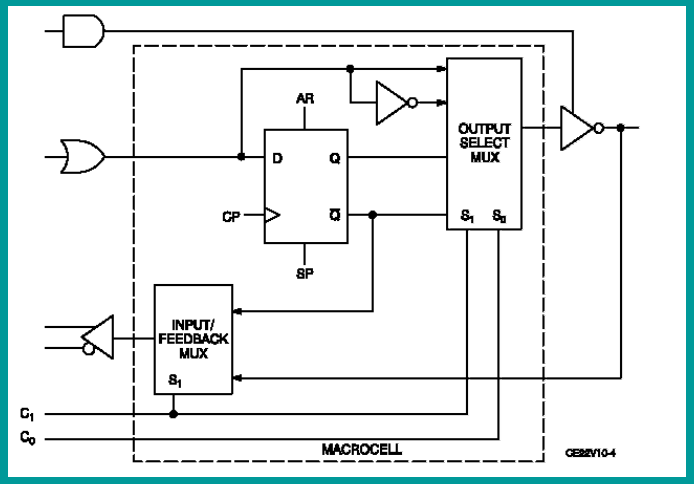
- Programmable product terms (AND plane)
  - AND gates can connect to any input/FF bit or bit-bar
- Fixed OR plane determine maximum # PTs
- Programmable macrocell
  - XOR gate selects SOP or POS for fewer PTs
  - FF for sequential logic or bypass for combinational logic
  - Feedback current state into array for FSM design

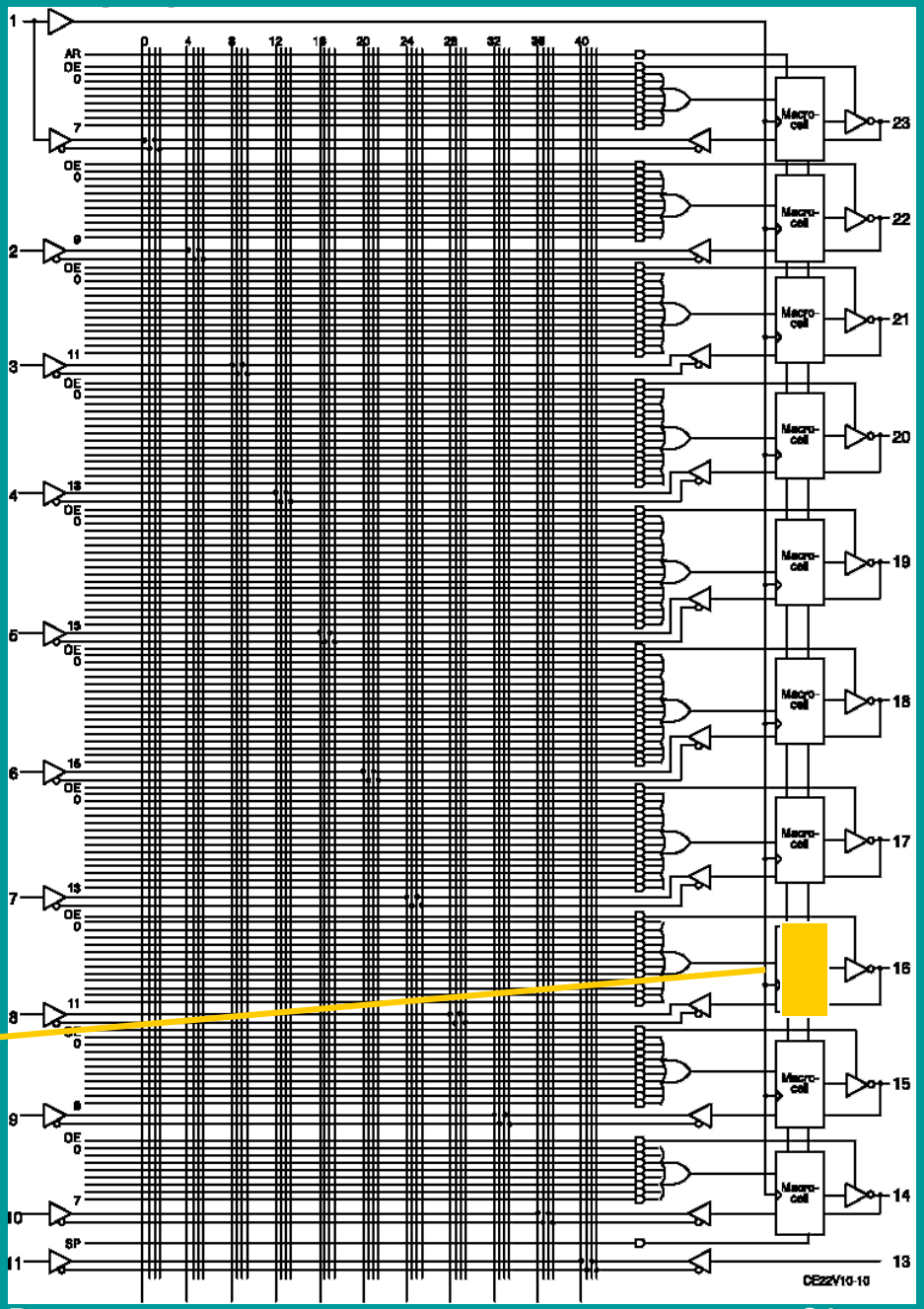


PLDs

22V10 replaced all PALs

- Combinational and/or sequential logic
  - Macrocell program bits C0, C1
- Up to 22 inputs w/complement
- Up to 10 outputs, each with
  - Macrocell
  - 8-16 product terms
  - Tri-state control product term
- Global
  - preset & clear PTs
  - clock





22V10: "V" => versatile outputs

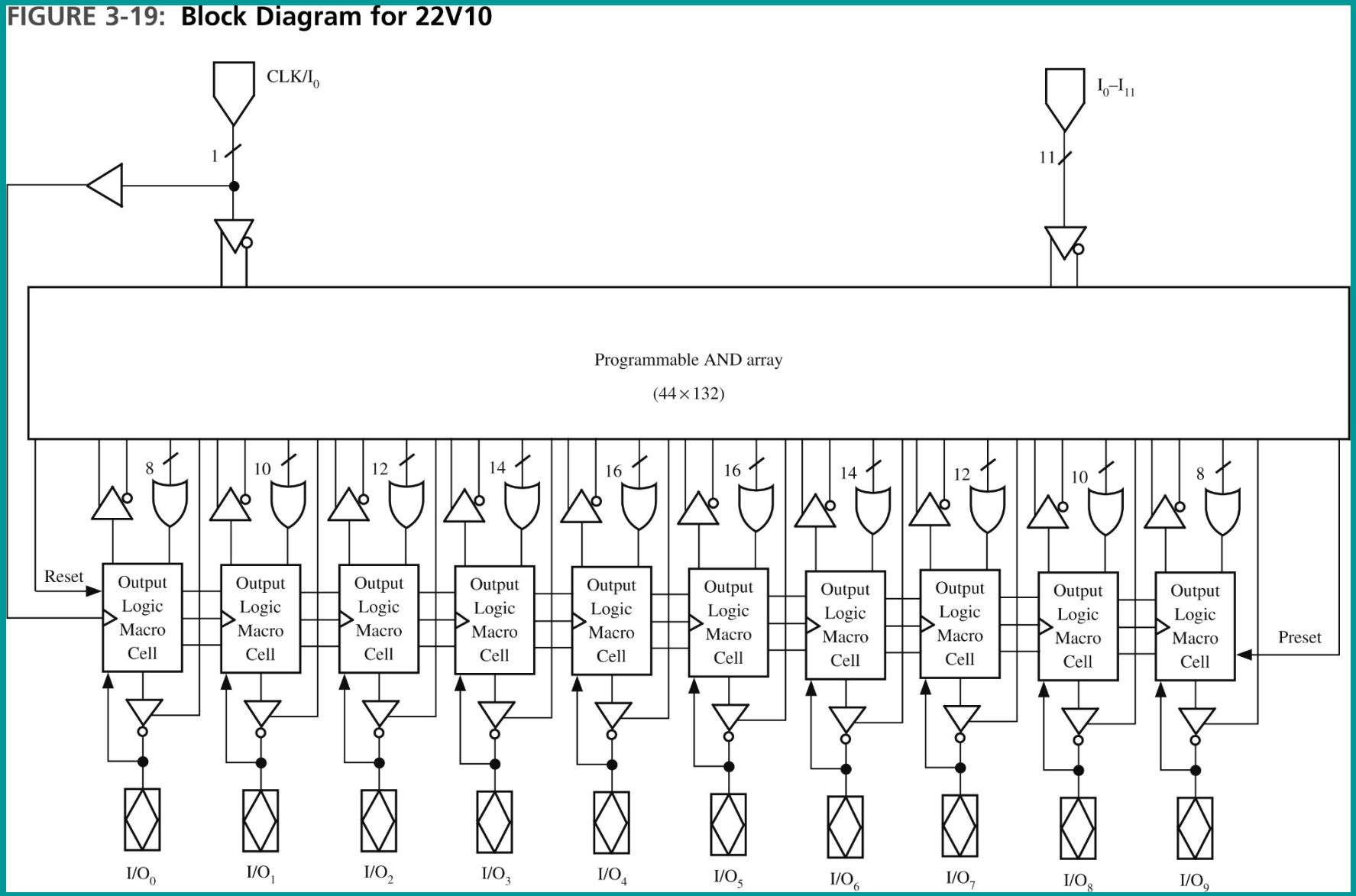
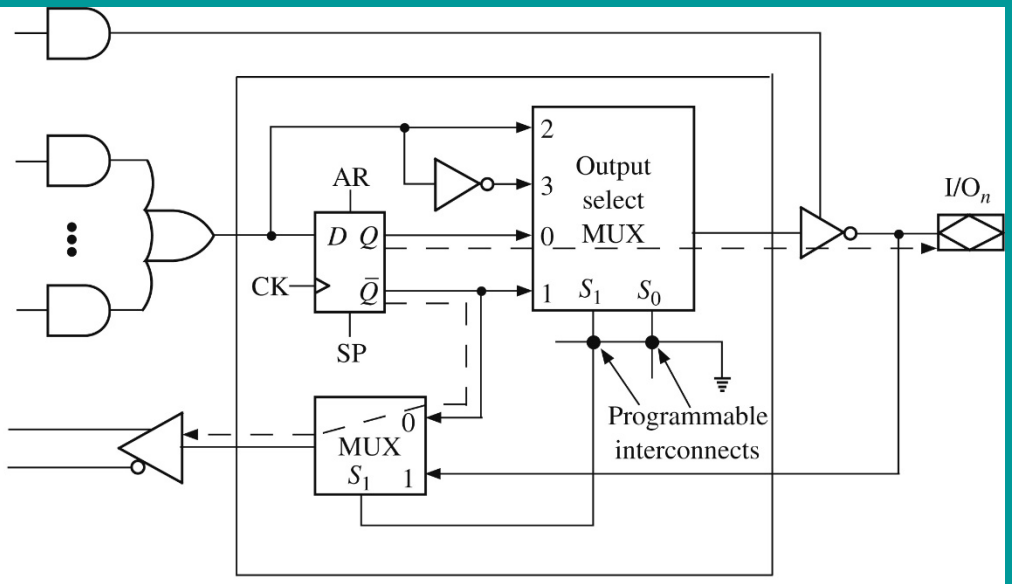
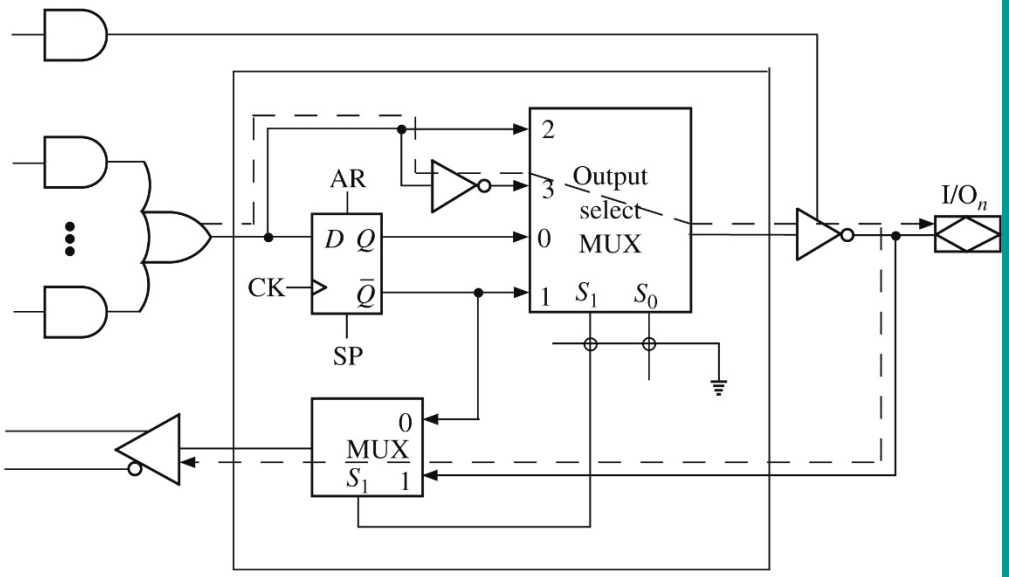


FIGURE 3-20: PLD Output Macrocell

of 22V10



(a) Paths with $S_1 = S_0 = 0$



(b) Paths with $S_1 = S_0 = 1$

CPLD implementation of a Mealy machine

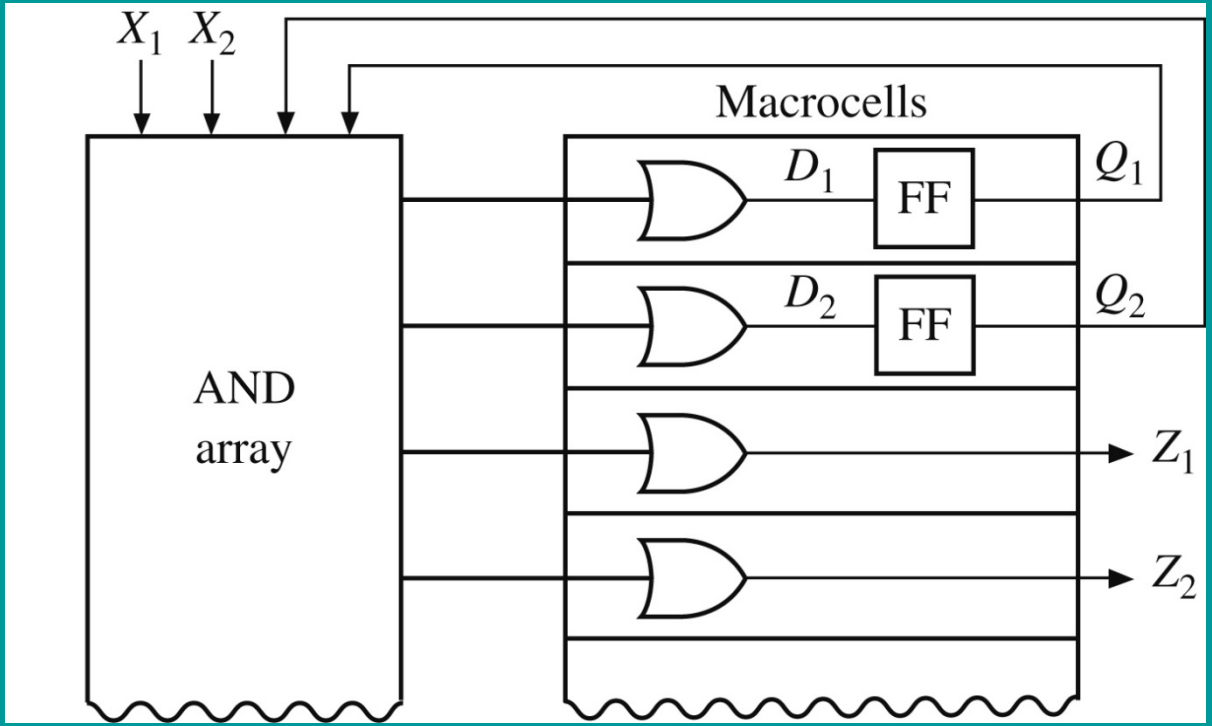


FIGURE 3-24: ***N*-Bit Parallel Adder with Accumulator**

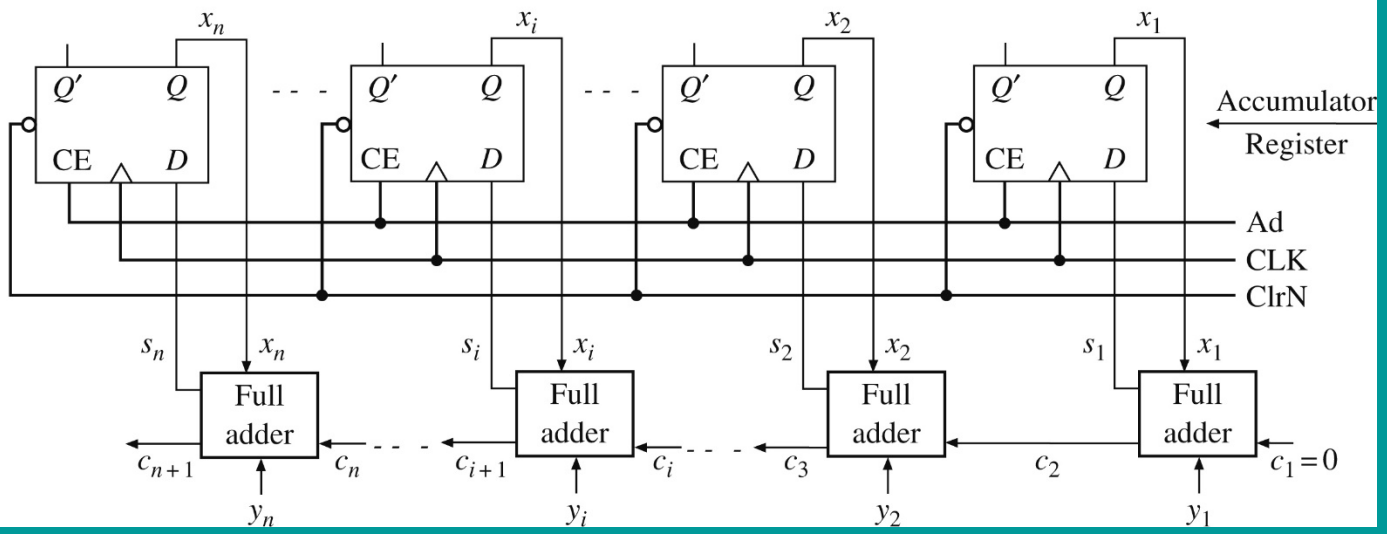
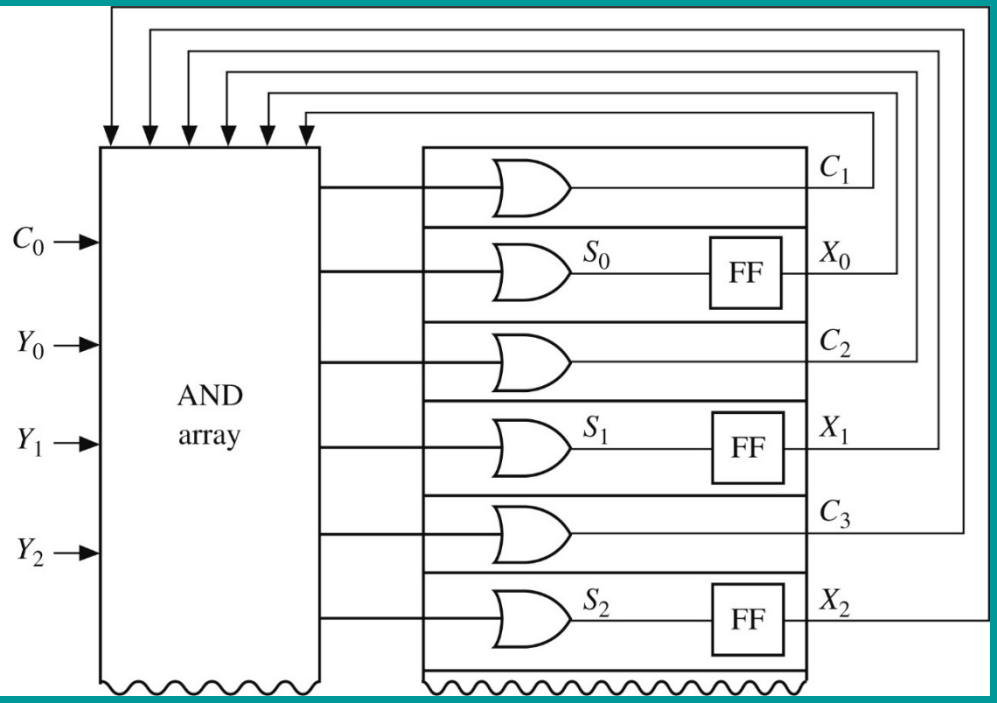


FIGURE 3-25: **CPLD Implementation of a Parallel Adder with Accumulator**



Complex PLDs (CPLDs)

- An array of PLDs
  - Global routing resources for connections
    - PLDs to other PLDs
    - PLDs to/from I/O pins
- Example: Cypress 39K
  - Each Logic Block (LB) similar to a 22V10
  - Each cluster of 8 LBs has two 8K RAMs & one 4K dual-port RAM/FIFO
    - Programmable Interconnect Modules (PIMs) provide interconnections
  - Array of up to 24 clusters with global routing

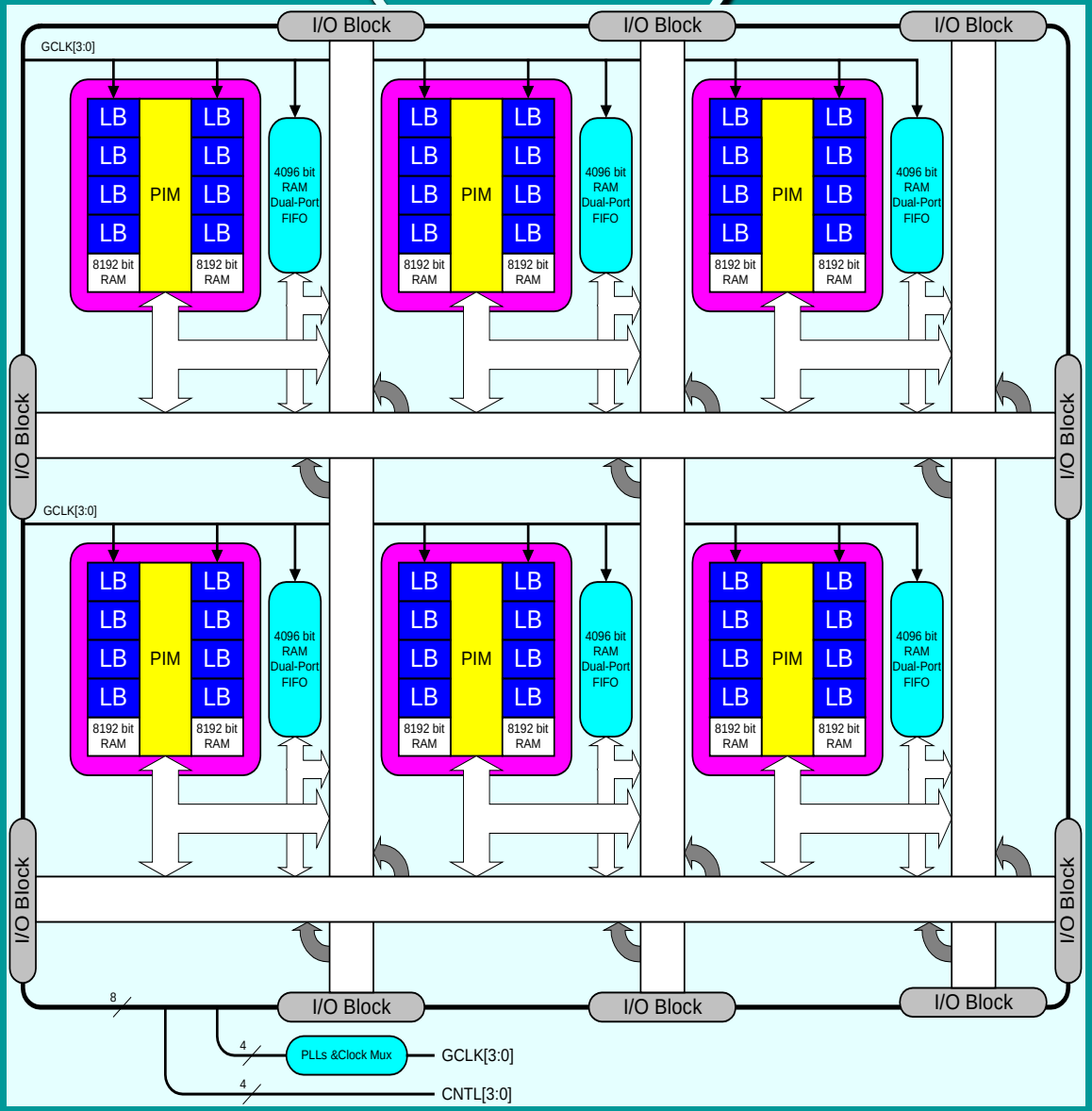


FIGURE 3-21: **Architecture of Xilinx CoolRunner XCR3064XL CPLD**

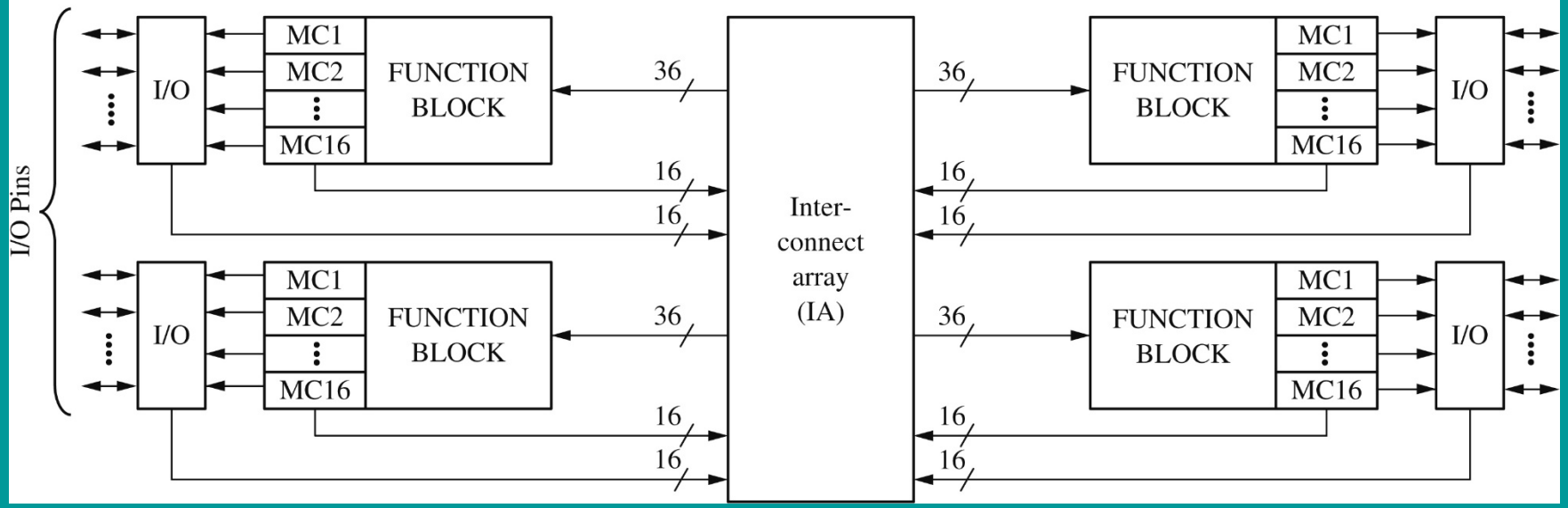
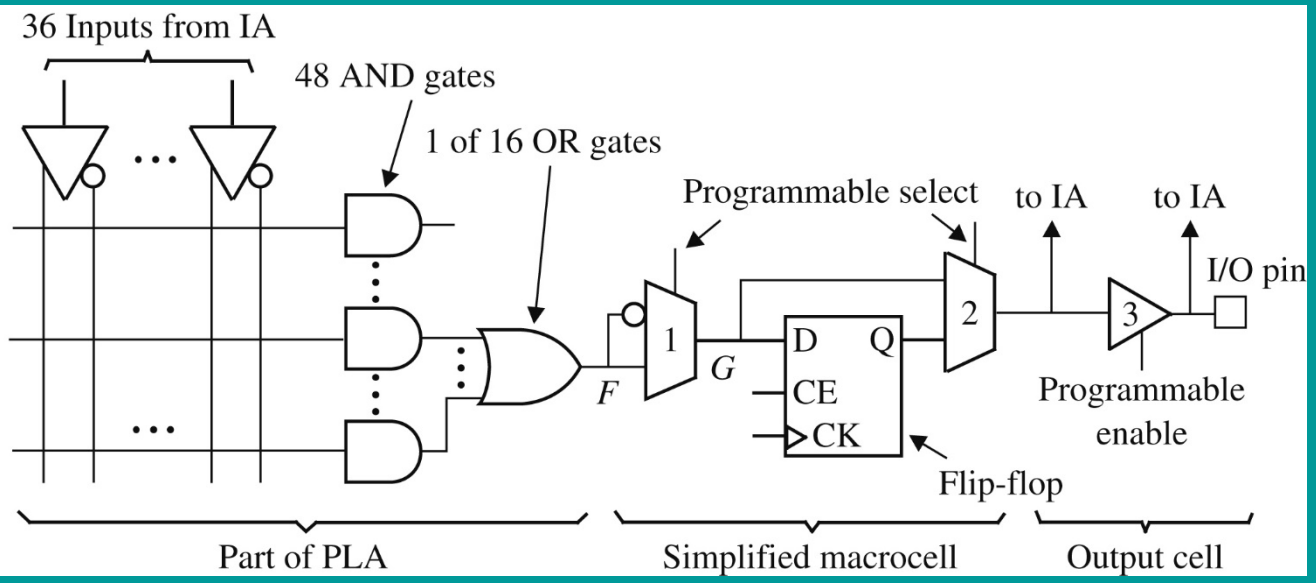


FIGURE 3-22: CPLD Function Block and Macrocell (Simplified Version of XCR3064XL)



Altera MAX architecture
(PAL-based logic modules)

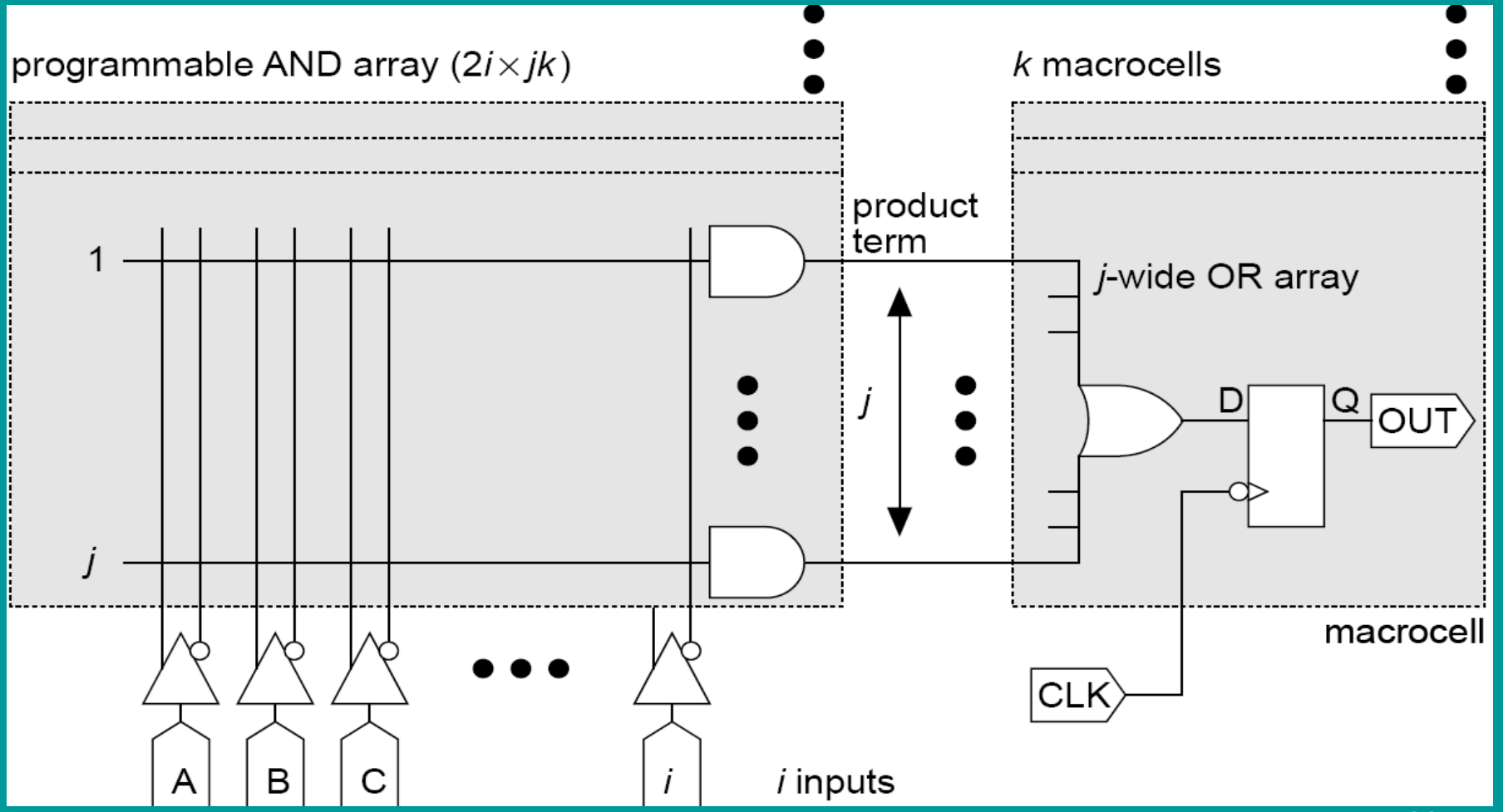


TABLE 3-6: Major CPLDs and their Approximate Capacity

Vendor	CPLD family	Gate Count
Xilinx	CoolRunner-II	750 to 12K
	CoolRunner XPLA3	750 to 12K
	XC9500XV	800 to 6400
	XC9500	800 to 6400
	XC9500XL	800 to 6400
Atmel	CPLD ATF15	750 to 3000 usable gates
	CPLD-2 22V10	500 usable gates
Cypress	Delta39K	30K to 200K
	Flash370i	800 to 3200
	Quantum38K	30K to 100K
	Ultra37000	960 to 7700
	MAX340 high-density EPLDs	600 to 3750
Lattice Semiconductor	ispXPLD 5000MX	75K to 300K
	ispMACH 4000B/C/V/Z	640 to 10,240
Altera	MAX II	240 to 2210 logic elements
	MAX3000	600 to 10K usable gates
	MAX7000	600 to 10K usable gates