Introduction to Programmable Logic Devices

(Class 7.2 – 2/28/2013)

CSE 2441 – Introduction to Digital Logic

Spring 2013

Instructor – Bill Carroll, Professor of CSE

Today’s Topics

- Complexity issues
  - Implementation
  - Design
- Programmable logic devices
  - Simple Programmable Logic Devices (SPLDs)
    - Programmable Logic Arrays (PLAs)
    - Programmable Array Logic (PALs)
  - Programmable Read Only Memory (PROM)
  - Complex Programmable Logic Devices (CPLDs)
  - Field Programmable Logic Devices (FPGAs)

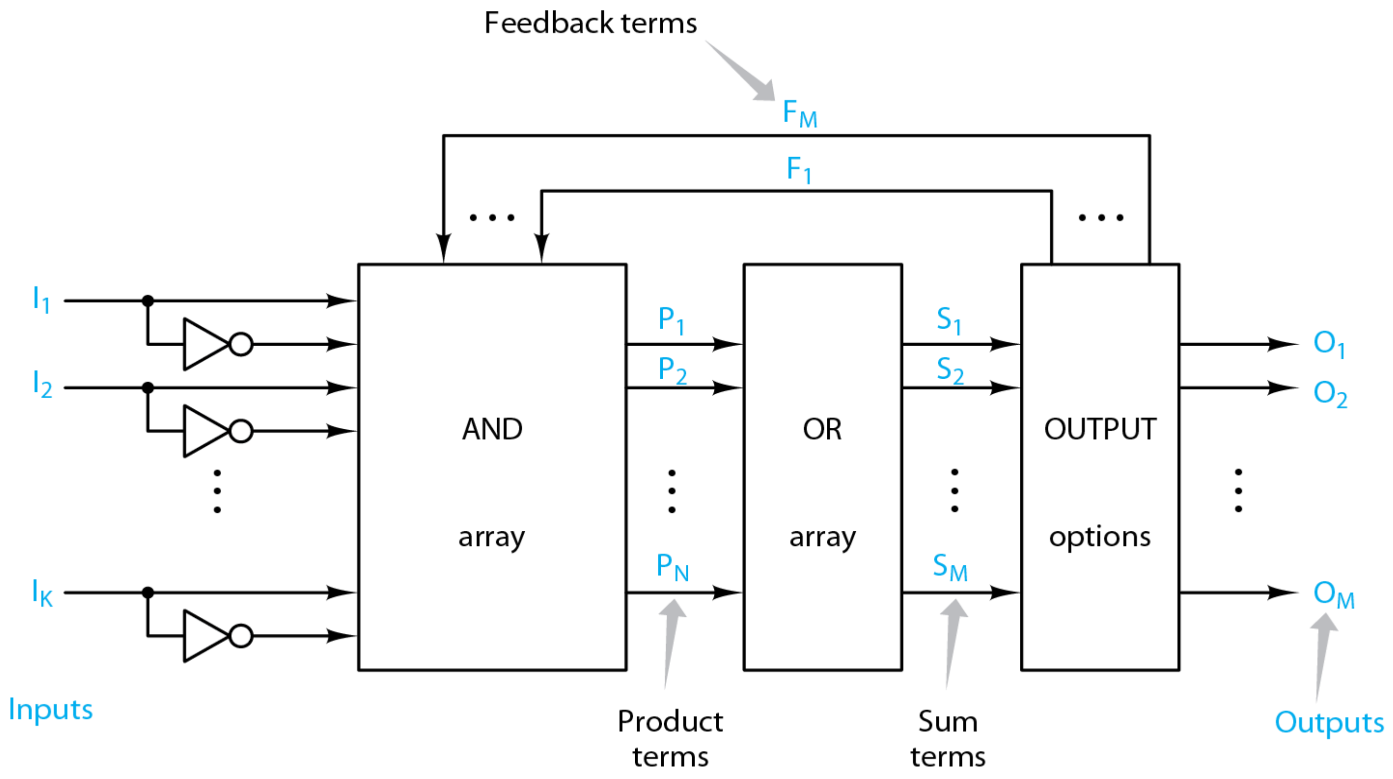
Issues In
Digital Circuit Implementation and Design

- Gates per integrated circuit
- Pin limitations
- Wiring complexity
- Speed
- Heat dissipation
- Design time
- More-complex functionality
- Testability
- Cost

Advances In
Digital Circuit Implementation and Design

- Higher levels of integration
- Larger packages
- More pins per package
- New and improved technologies
- Multilayer printed circuit boards
- Application Specific ICs (ASICS)
- Programmable logic
- CAD tools
- Automated testing

Basic Programmable Logic Array Organization



Two-Level AND-OR Arrays

Realization of $f(a,b,c) = abc' + b'c$

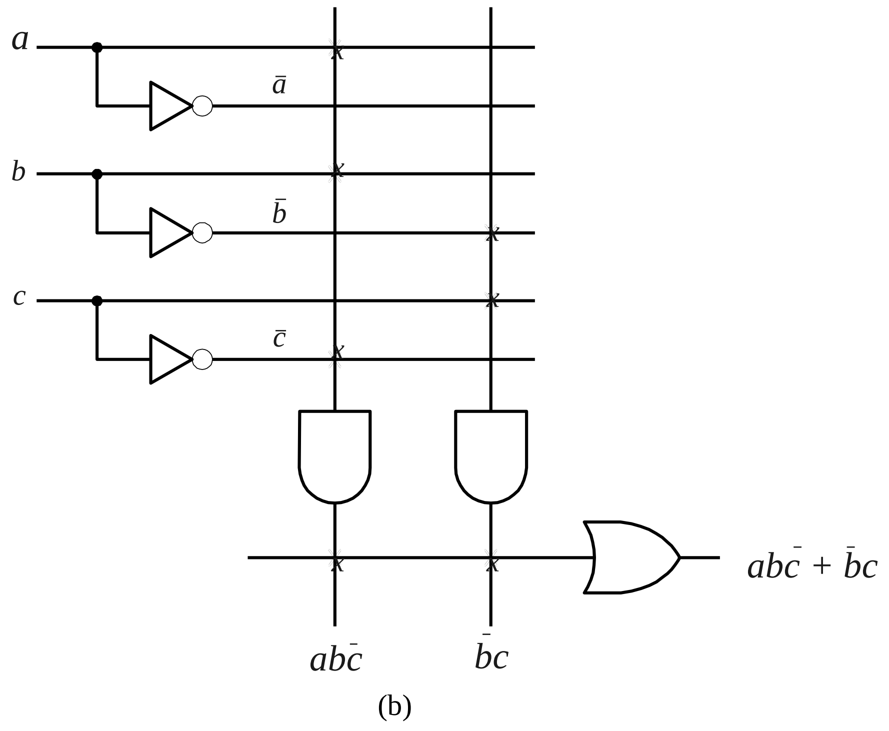


Figure 5.5

Multiple Functions Realized by an AND-OR Array

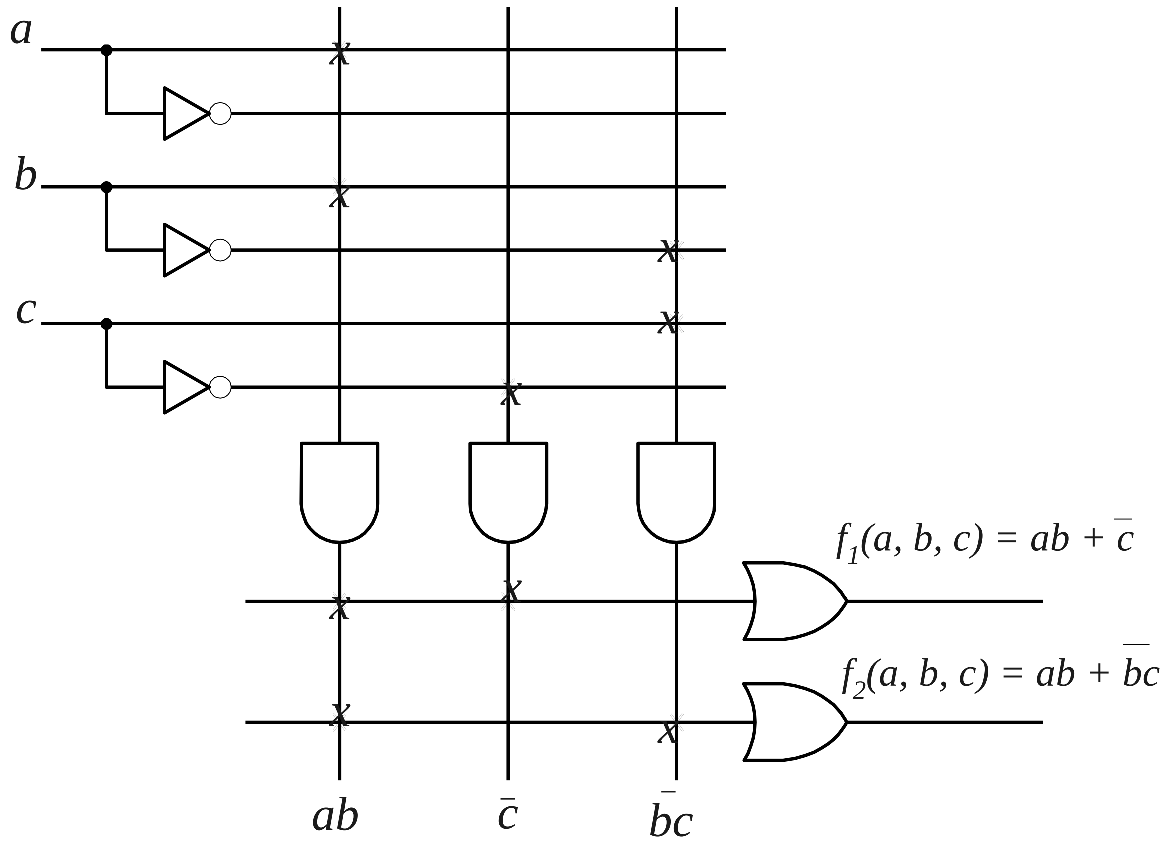


Figure 5.6

PLA Design -- Example 5.1

- Design a PLA to realize the following 3 logic functions and show the internal connections.

$$f_1(A,B,C,D,E) = A'B'D' + B'CD' + A'BCDE'$$
$$f_2(A,B,C,D,E) = A'BE + B'CD'E$$
$$f_3(A,B,C,D,E) = A'B'D' + B'C'D'E + A'BCD$$

- Since there are 5 inputs, there must be at least 5 inputs to the PLA, each of which must be both complemented and uncomplemented.

  There are a total of 7 unique product terms in the preceding 3 expressions. So the PLA must generate at least 7 product terms.

  Since 3 functions are being realized, there must be 3 sum (OR) terms generated.

Example 5.1 (continued)

The PLA organization is shown in Fig. 5.7.

Table 5.1 shows the connections that must be made in the AND and OR arrays. In the table, the product term numbers correspond to the AND gate numbers in Fig. 5.7, each connected to one vertical product line, on which a product term is generated.

In the AND array portion of the table

- a 0 indicates that the complement of the variable is connected to the product line
- a 1 indicates the the uncomplemented input is connected to the product line
- an $\times$ indicates that neither is connected to the product line

For the OR array, a 1 indicates a connection and a 0 indicates no connection.

PLA for Example 5.1

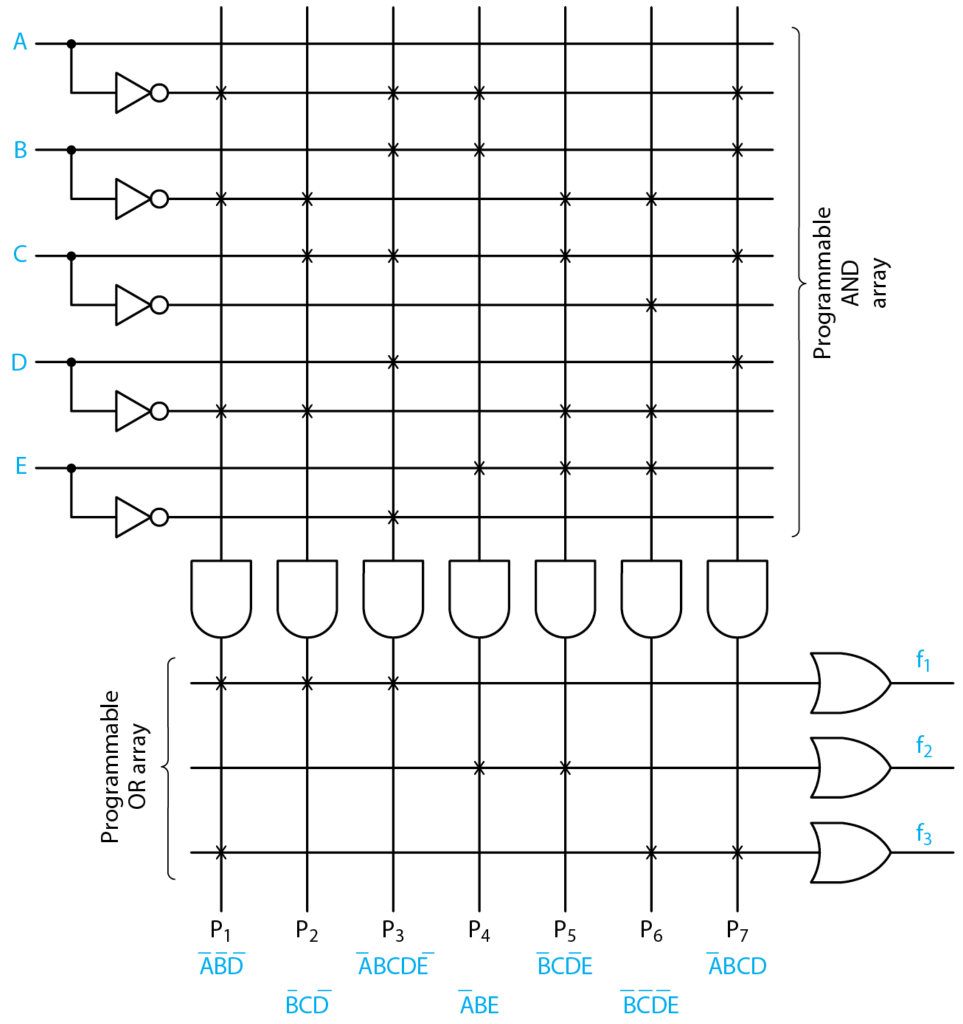
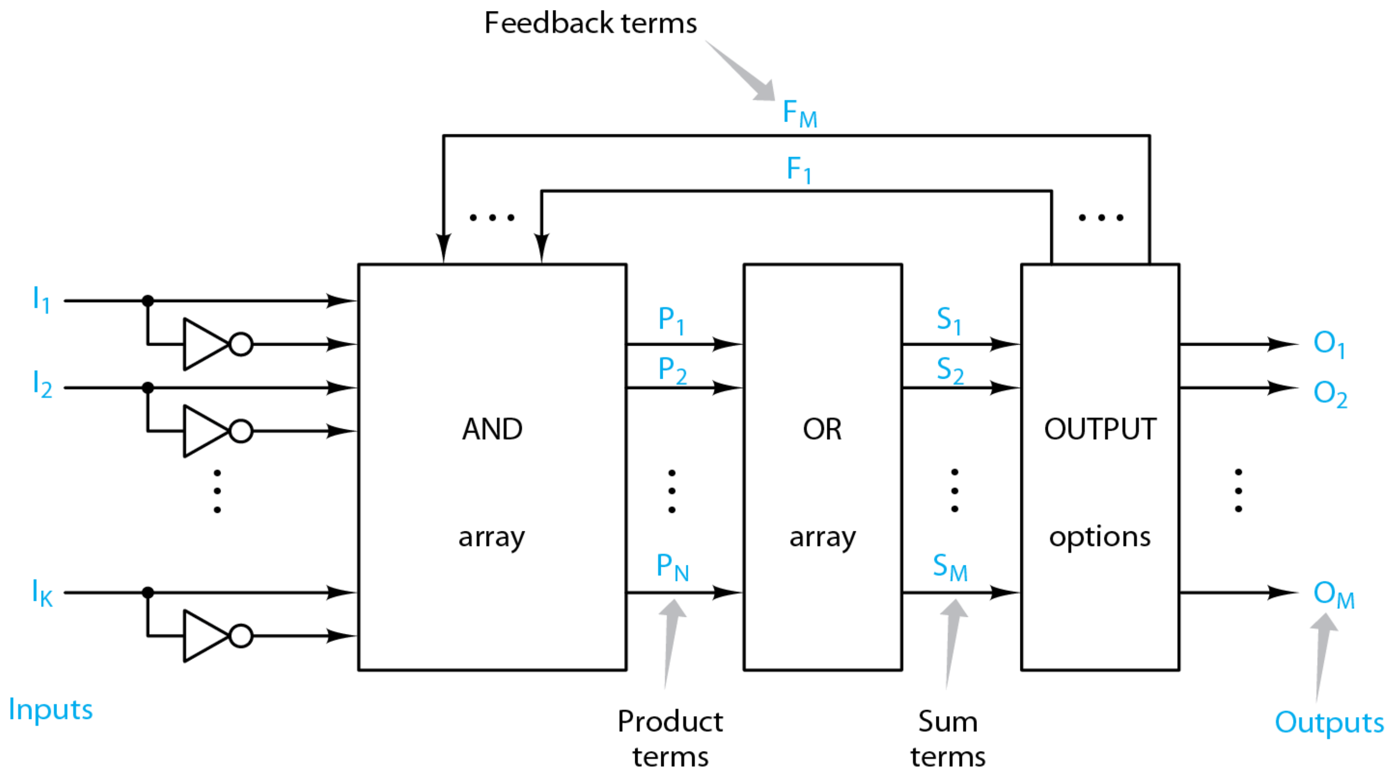


Figure 5.7

Table 5.1 -- PLA Table for Example 5.1

Product Term		AND Array Inputs *ABCDE*	OR Array Outputs $f_1f_2f_3$
1	*A’B’D’*	00×0×	101
2	*B’CD’*	×010×	100
3	*A’BCDE’*	01110	100
4	*A’BE*	01× ×1	010
5	*B’CD’E*	×0101	010
6	*B’C’D’E*	×0001	001
7	*A’BCD*	0111×	001

Basic Programmable Logic Array Organization



Output Polarity Options

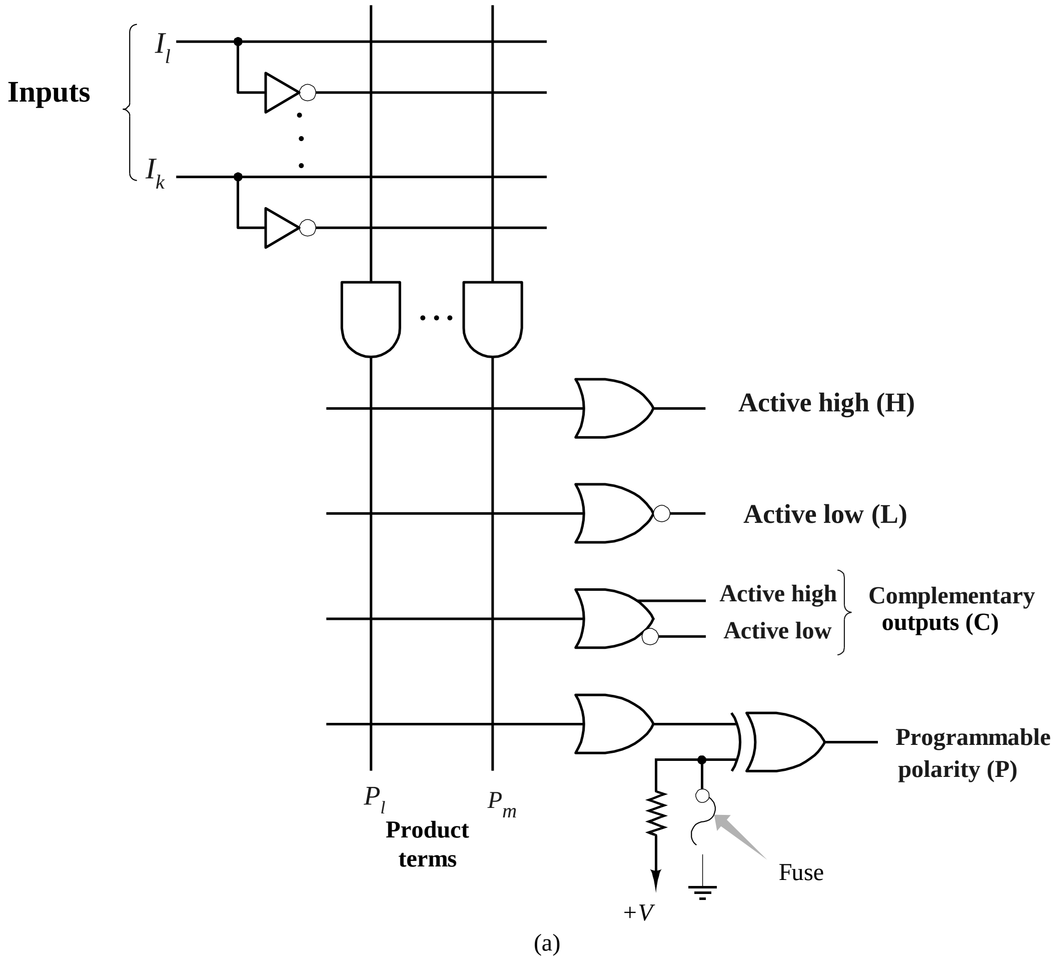
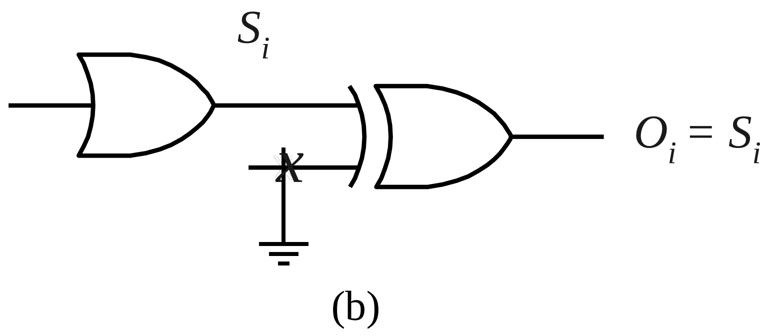


Figure 5.11

Output Polarity Options(continued)



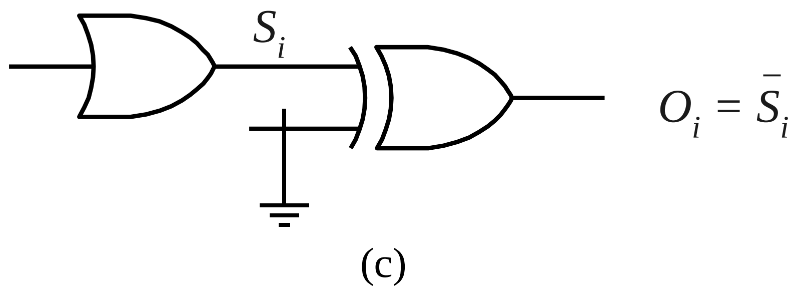


Figure 5.11

Implementation of SOP and POS Forms

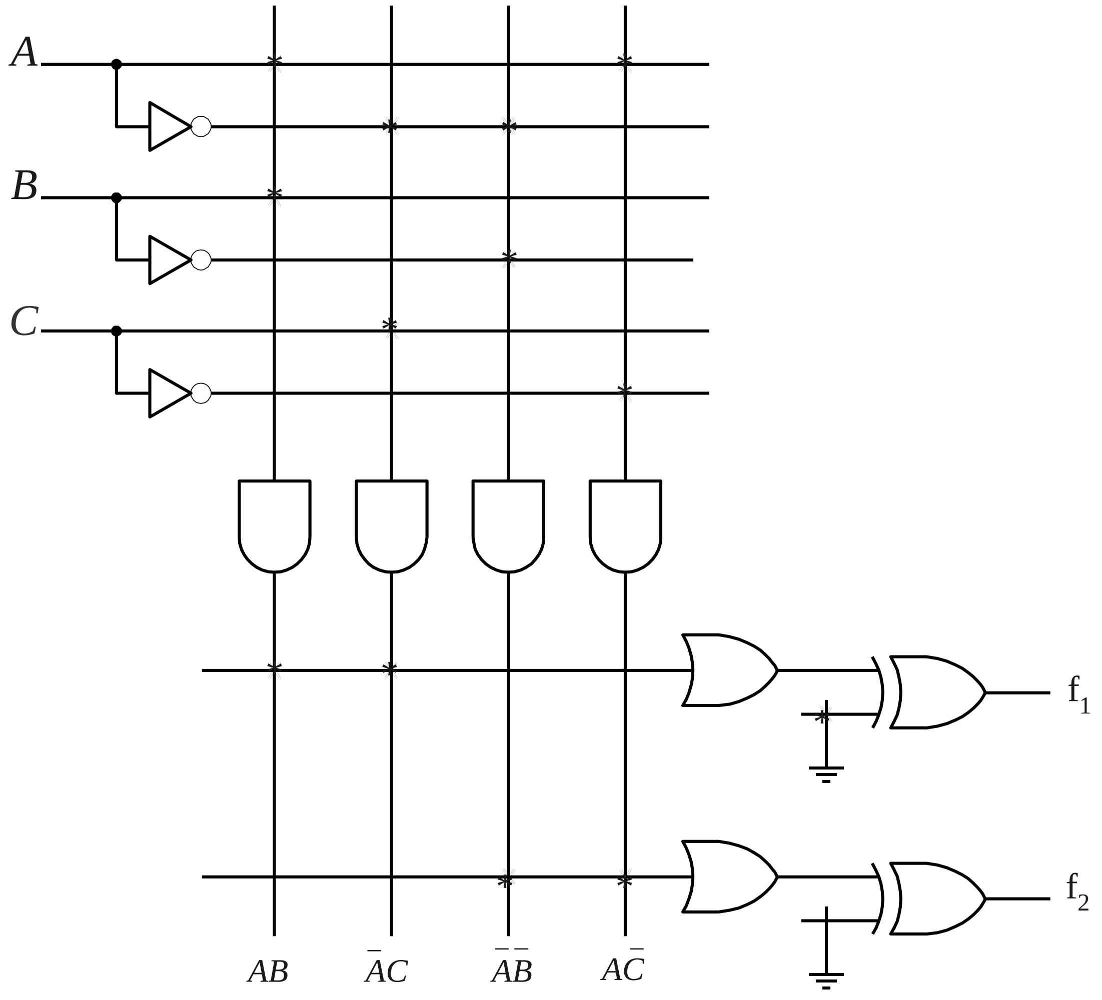


Figure 5.12

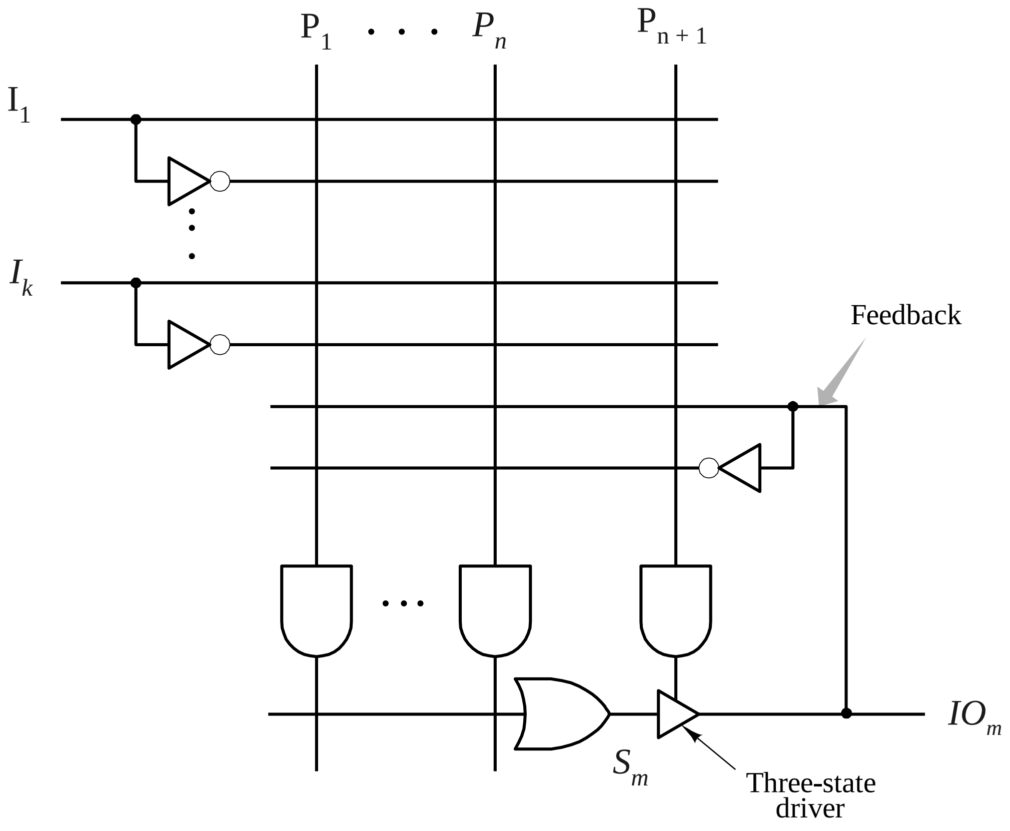
Bidirectional Pins and Feedback Lines

- A bidirectional pin is driven by a *three-state driver*, whose control line is connected to one of the product terms.

- When the control line is 1, the driver is said to be enabled and functions as a short circuit, as shown in Fig. 5.13b. In this case the sum term is driven onto the pin, which therefore functions as an output. This value is fed back to the AND array, where it can be used to form product terms.

- When the driver control line is 0, the driver is disabled and functions as an open circuit, as shown in Fig. 5.13c. This disconnects the sum term from the pin, which, through the feedback line, now becomes an input to the AND array.

Bidirectional Pins



(a)

Figure 5.13

Bidirectional Pins(continued)

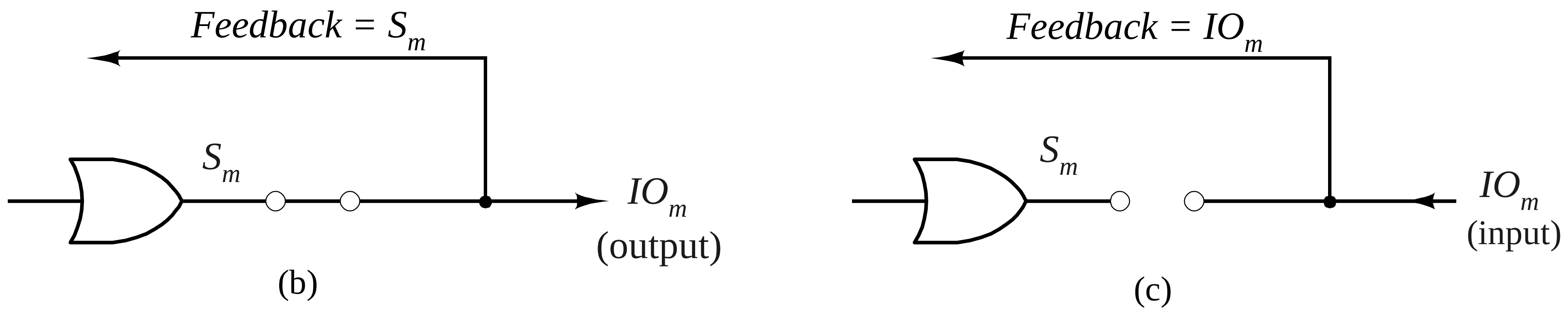


Figure 5.13

Two-bit ripple-carry adder Example 5.3

- Implement a 2-bit ripple-carry adder, as shown in Fig. 5.14a, using a programmable logic array having 4 dedicated input pins, 3 dedicated output pins, and 3 bidirectional pins.

The standard logic equations for one state, i, of an n-bit full-adder are the following:

$S_i = A_iB_iC_{i-1} + A_iB'_iC'_{i-1} + A'_iB_iC'_{i-1} + A'_iB'_iC_{i-1}$

$C_i = A_iB_i + A_iC_{i-1} + B_iC_{i-1}$

where A_i and B_i are the data inputs and C_{i-1} the carry input to stage i, S_i is its sum output and C_i the carry output. For a ripple-carry adder, the carry-out of one stage is connected to the carry input of the next stage, as shown in Fig. 5.14a.

Example 5.3(continued)

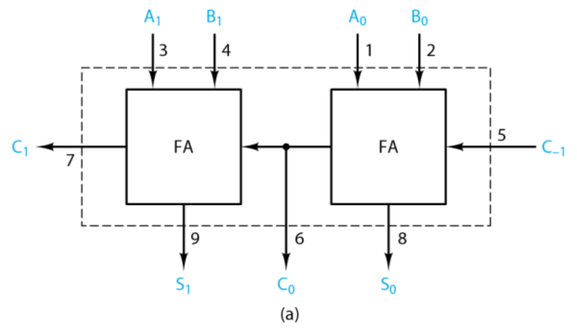
- Figure 5.14b shows the PLA implementation of the block diagram in Fig. 5.14a.

  Since the adder requires 5 inputs, and there are only 4 dedicated input pins, bidirectional pin 5 is used as another input.

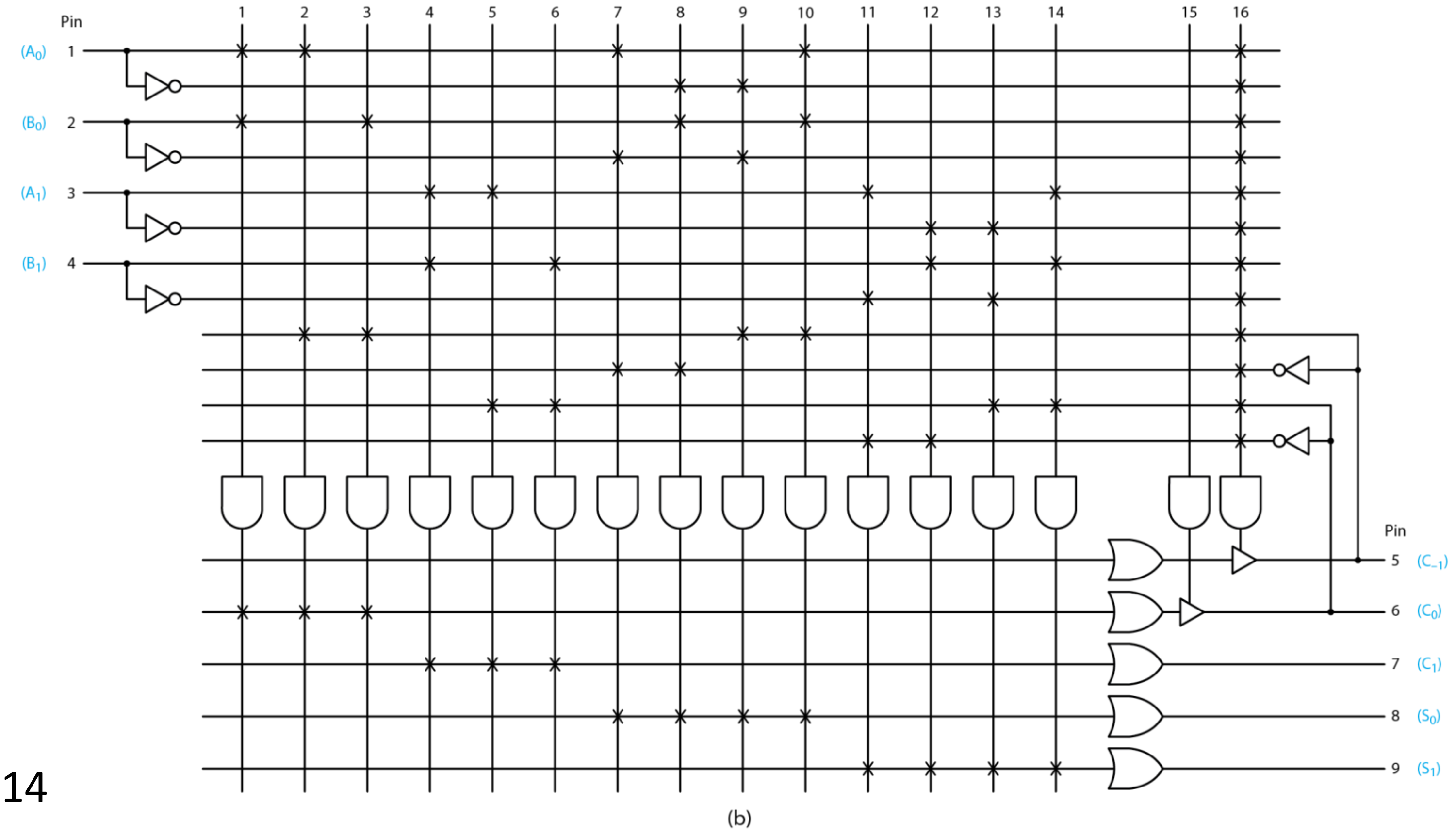
  The driver of pin 5 is disabled by product line 16 by leaving all its fuses intact.

  Carry term C_0 is used to compute terms S_1 and C_1 through the feedback line from pin 6, allowing C_0 to be combined with A_1 and B_1 by the preceding equations.

Two-bit Ripple-carry Adder – Example 5.3



(a)



(b)

Figure 5.14

Programmable Read-only Memory

- A PROM comprises a fixed AND array and a programmable OR array, as illustrated in Fig. 5.21.

  The AND array generates all 2^n possible minterm products of its n inputs and therefore often referred to as an n-to-2^n decoder.

  The OR array allows any combination of product terms to be included in each sum term. The canonical sum of products form of any function can be realized directly from its truth table or minterm list.

- Figure 5.22 illustrates the organization of most typical commercial PROMs.

Programmable Read-only Memory (PROM)

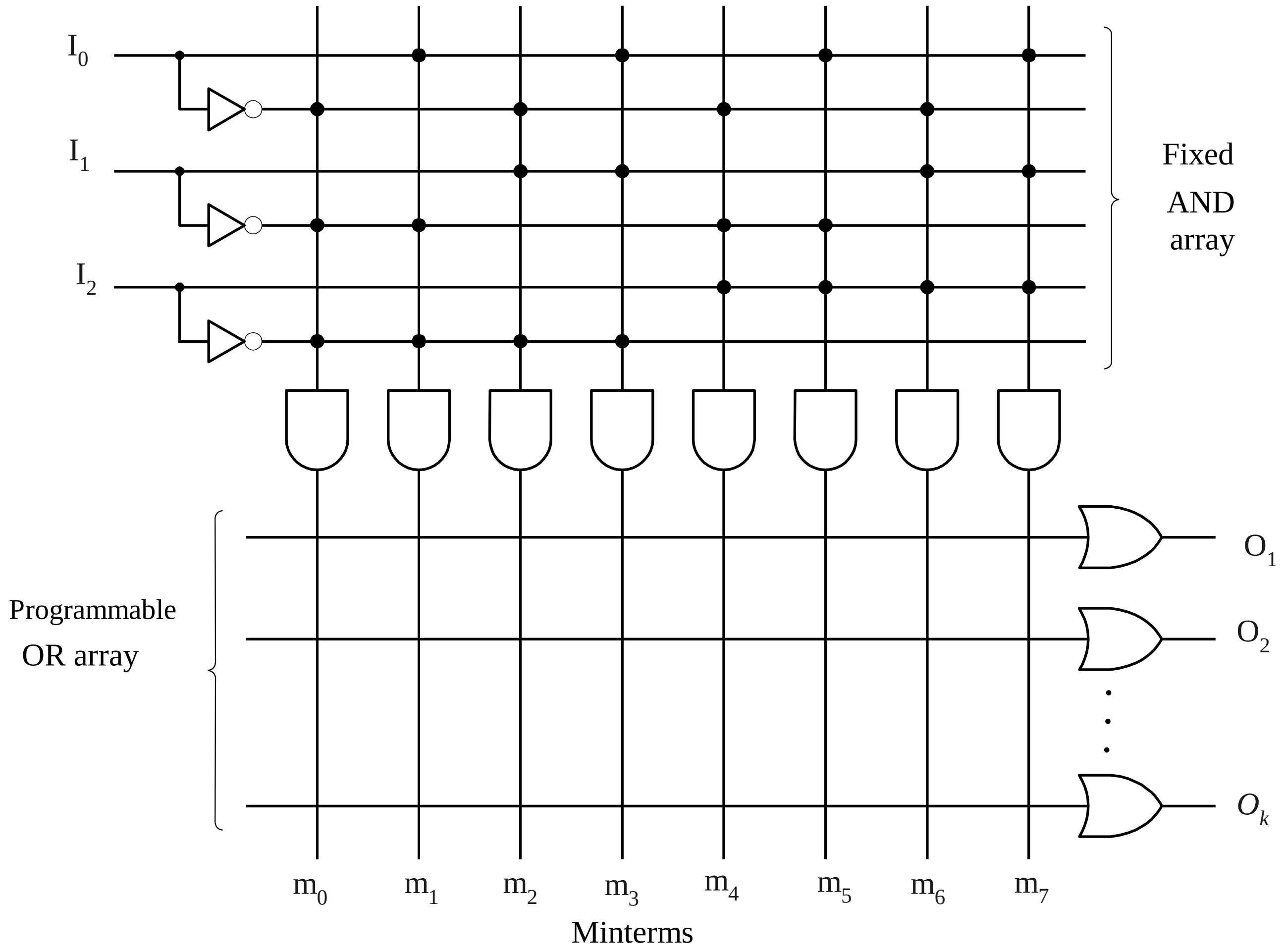


Figure 5.21

General Configuration of a Commercial PROM

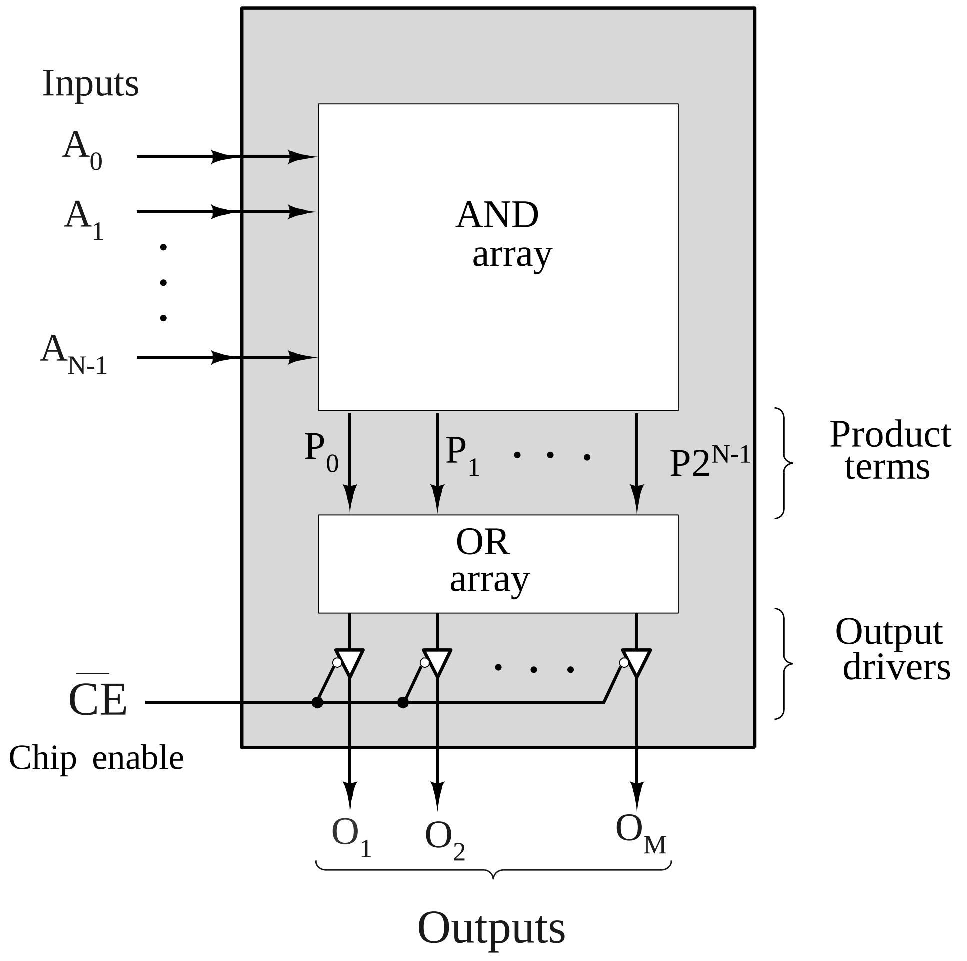


Figure 5.22

Realizing Logic Functions with PROMs

- Each output of a PROM is capable of realizing any arbitrary switching function by connecting that output to the minterms of the function.

  To realize a given switching function with a PROM, first express the function in canonical SOP form or else derive the truth table of the function.

  Then, each of the minterms of the function is connected to the desired OR term to produce the canonical SOP form.

- It should be noted that the use of a commercially available PROM would be very inefficient when only a small number of minterms is needed, unless minimizing chip count is the primary goal.

PROM Realization -- Example 5.6

Realize the following 3 switching functions with a 3-input, 3-output PROM.

$$f_1(A, B, C) = AB + B'C$$

$$f_2(A, B, C) = (A + B' + C)(A' + B)$$

$$f_3(A, B, C) = \mathrm{A + BC}$$

First, convert each function to canonical SOP form.

$$f_1(A, B, C) = ABC' + ABC + A'B'C + AB'C$$

$$= \sum m(1, 5, 6, 7)$$

$$f_2(A, B, C) = (A + B' + C)(A' + B + C')(A' + B + C)$$

$$= \prod M(2, 4, 5) = \sum \mathrm{m}(0, 1, 3, 6, 7)$$

$$f_3(A, B, C) = AB'C' + ABC' + AB'C + ABC + A'BC$$

$$= \sum m(3, 4, 5, 6, 7)$$

PROM Realization for Example 5.6

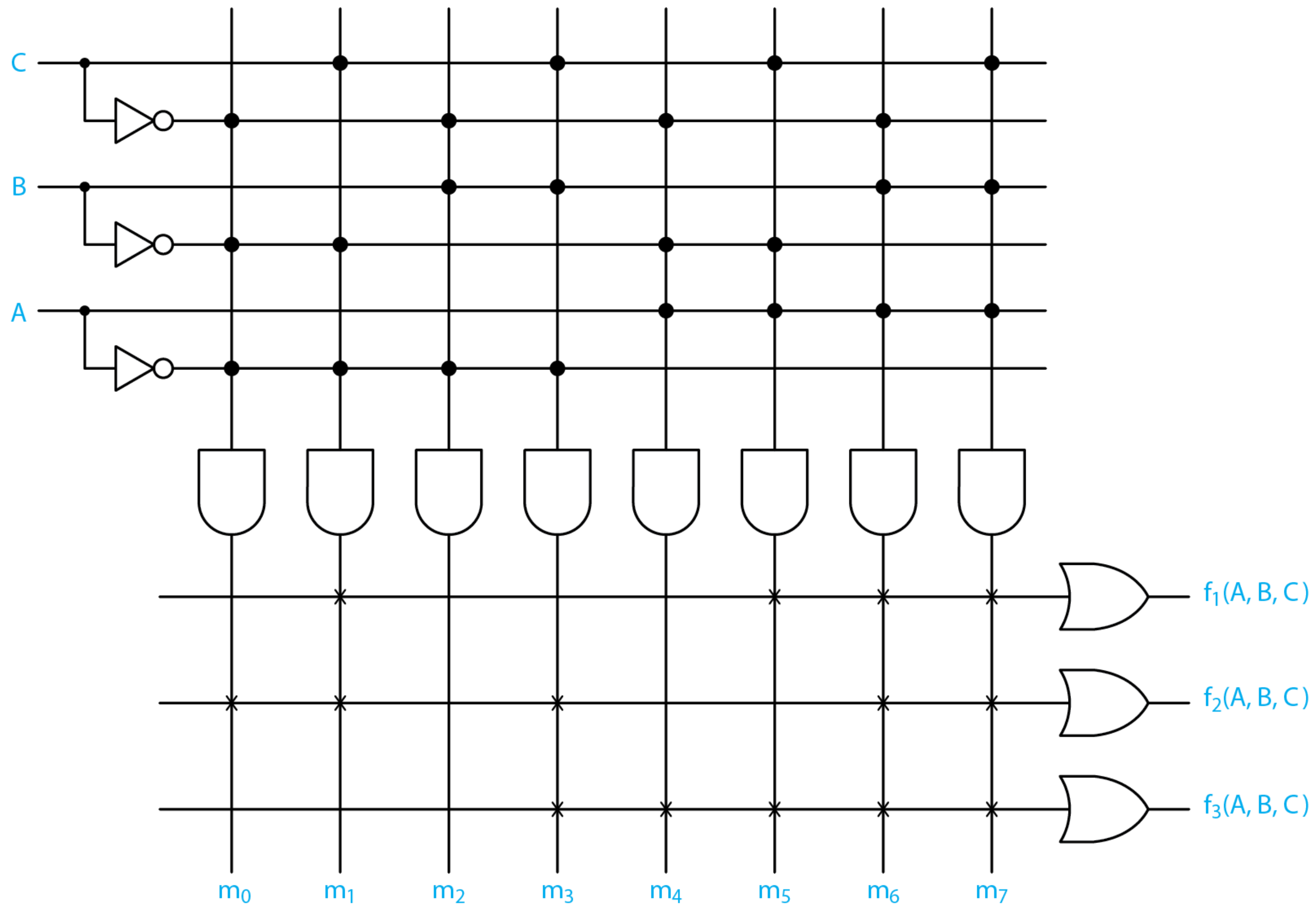


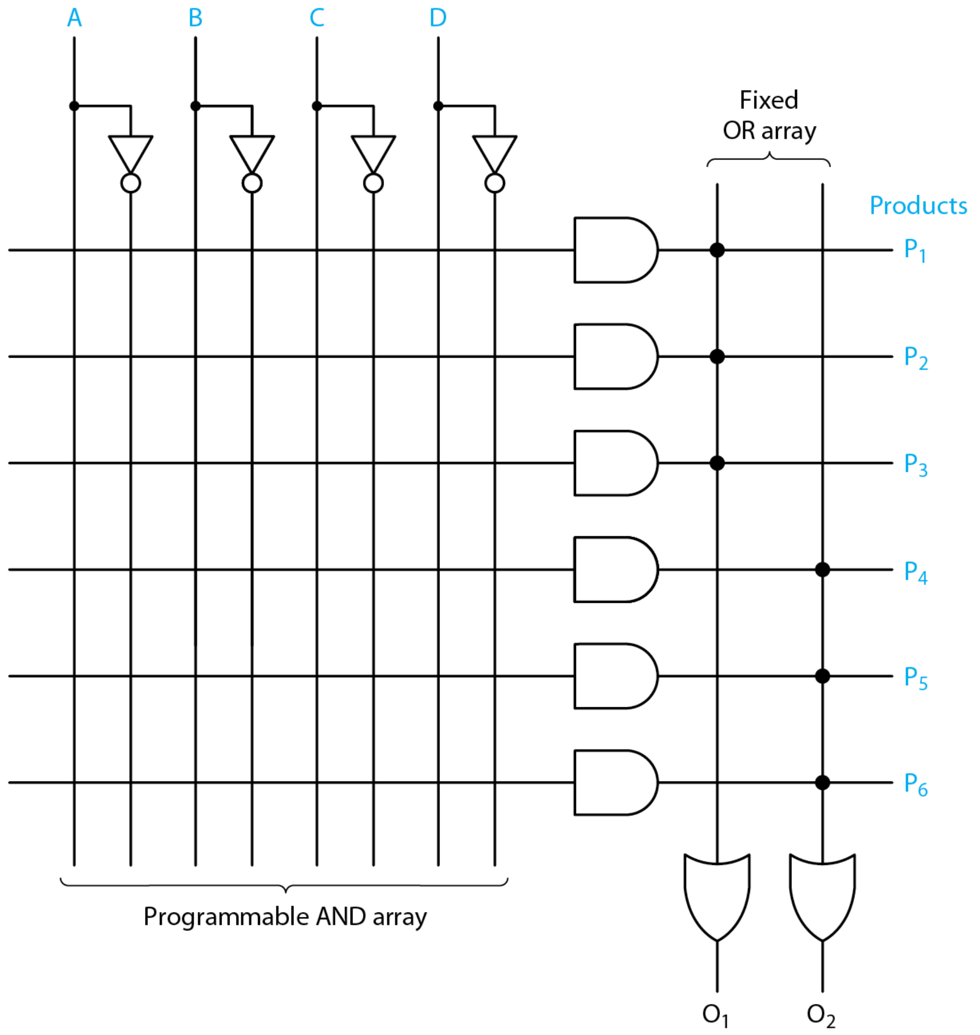
Figure 5.23

Lookup Tables (LUT)

- A common application of PROMs is the *lookup table*, in which a function is stored in tabular form with its arguments used as an index into the table to retrieve the value of the function for those arguments.

- Since truth tables can be readily realized by PROMs, lookup tables are implemented by writing them in truth table format and then realizing the truth table with a PROM.

Programmable Array Logic (PAL)



Example 5.10 – PAL Design Example

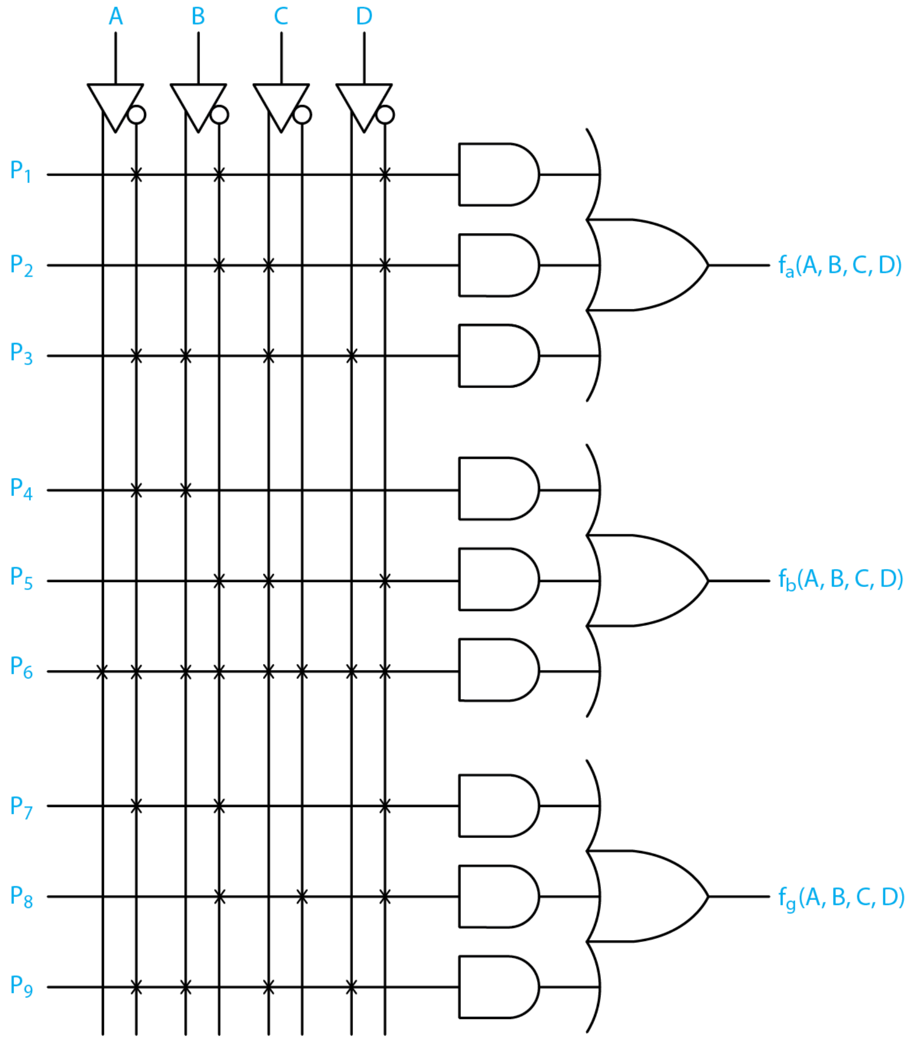
Realize the following functions with a PAL

$f_\alpha(A,B,C,D) = A'B'D' + B'CD' + A'BCD$

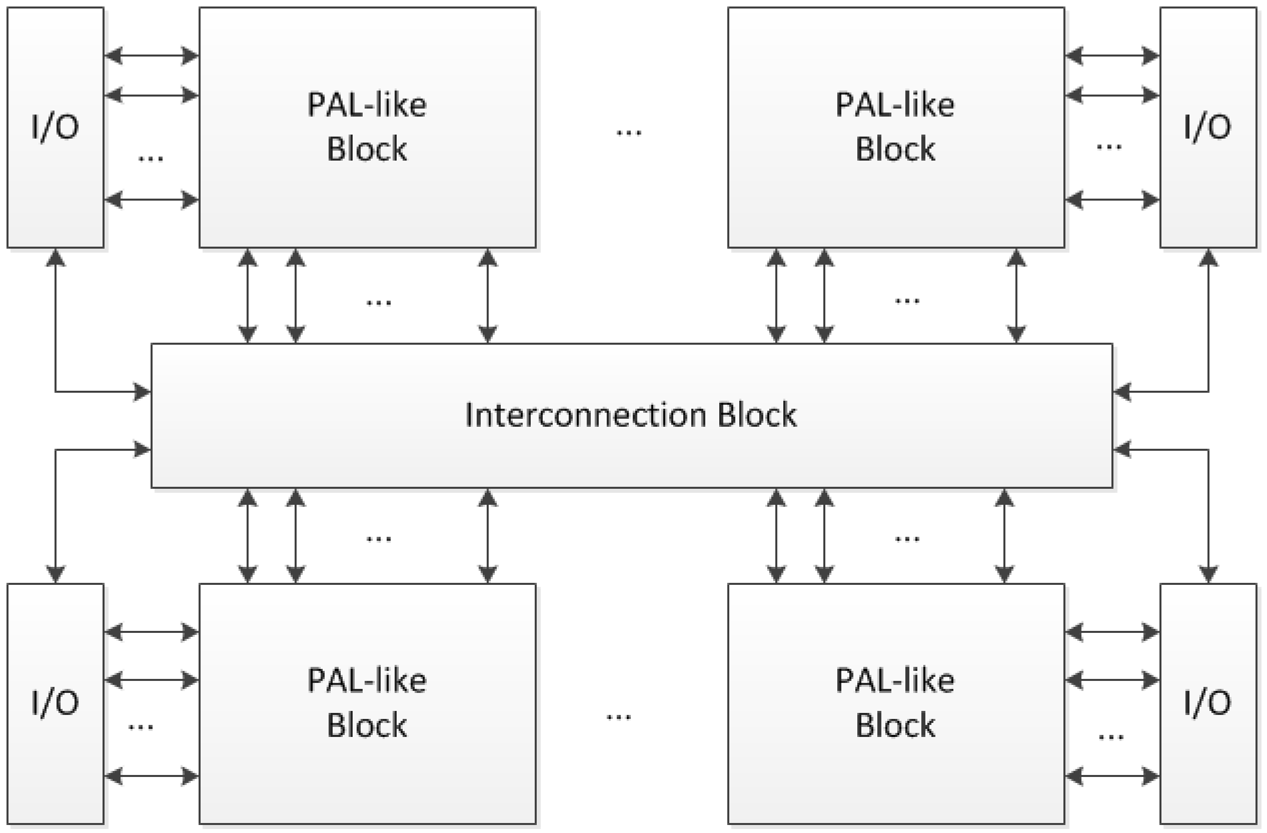
$f_\beta(A,B,C,D) = A'B + B'CD'$

$f_\gamma(A,B,C,D) = A'B'D' + B'C'D' + A'BCD$

Example 5.10 -- PAL Realization



Complex Programmable Logic Devices (CPLDs)

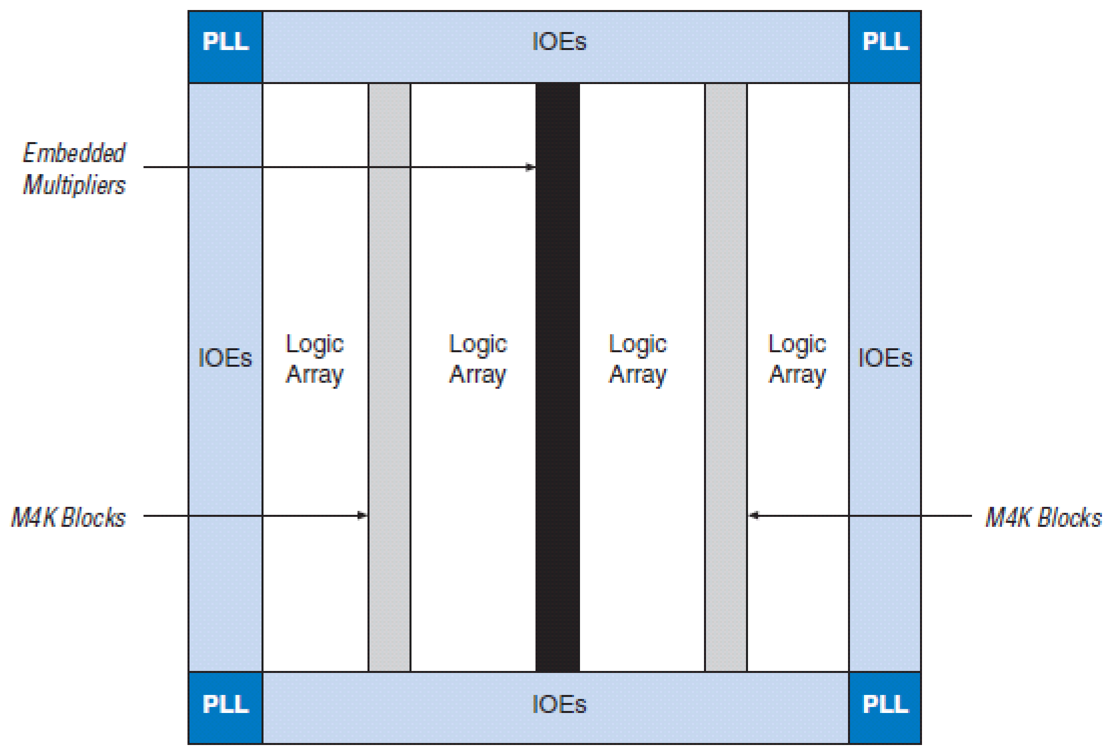


Summary of Programmable Logic Devices

- PLA – AND array and OR array are programmable
- PAL – AND array programmable, OR array fixed
- PROM – AND array fixed, OR programmable
- CPLD – programmable array of PALs
- FPGA – programmable array of logic elements

Cyclone II FPGA*

Figure 2–1. Cyclone II EP2C20 Device Block Diagram



LAB – Logic Array Block, PLL – Phase Locked Loop, IOEs – I/O Elements

*From Cyclone II Device Family Data Sheet

Cyclone II FPGA Family Features*

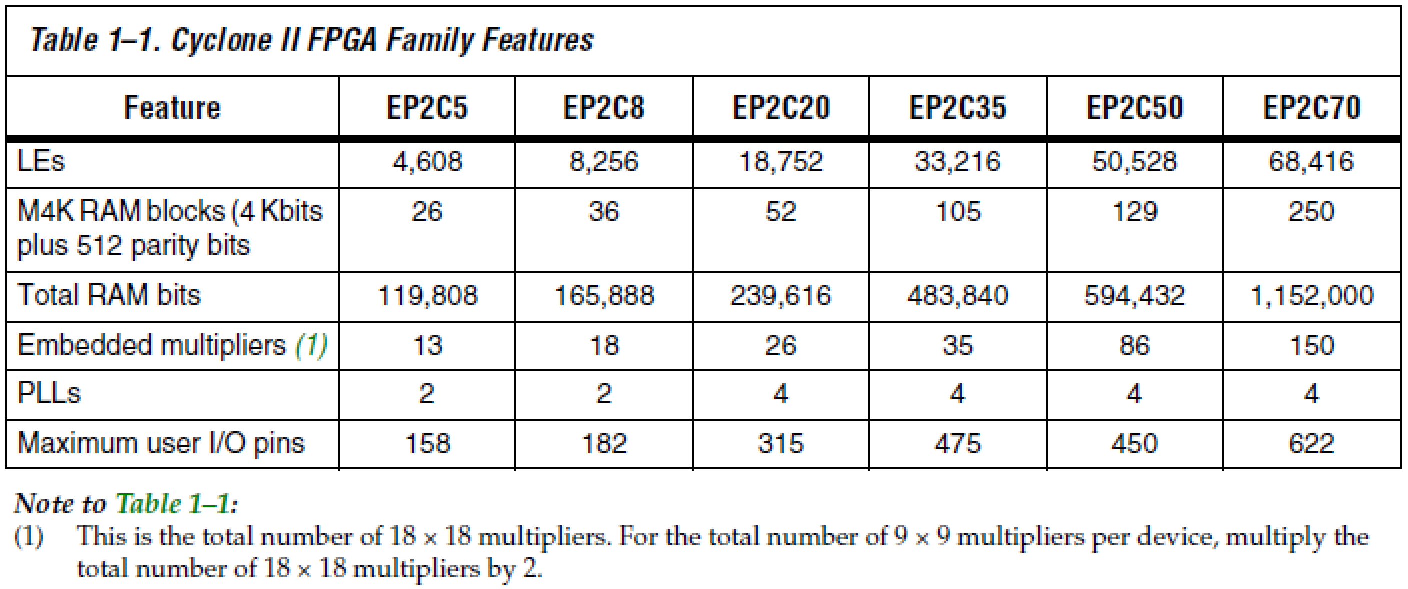


Table 1–1. Cyclone II FPGA Family Features

Feature	EP2C5	EP2C8	EP2C20	EP2C35	EP2C50	EP2C70
LEs	4,608	8,256	18,752	33,216	50,528	68,416
M4K RAM blocks (4 Kbits plus 512 parity bits	26	36	52	105	129	250
Total RAM bits	119,808	165,888	239,616	483,840	594,432	1,152,000
Embedded multipliers *(1)*	13	18	26	35	86	150
PLLs	2	2	4	4	4	4
Maximum user I/O pins	158	182	315	475	450	622

Note to Table 1–1:

(1) This is the total number of 18 × 18 multipliers. For the total number of 9 × 9 multipliers per device, multiply the total number of 18 × 18 multipliers by 2.

*From Cyclone II Device Family Data Sheet

Cyclone II Package Options*

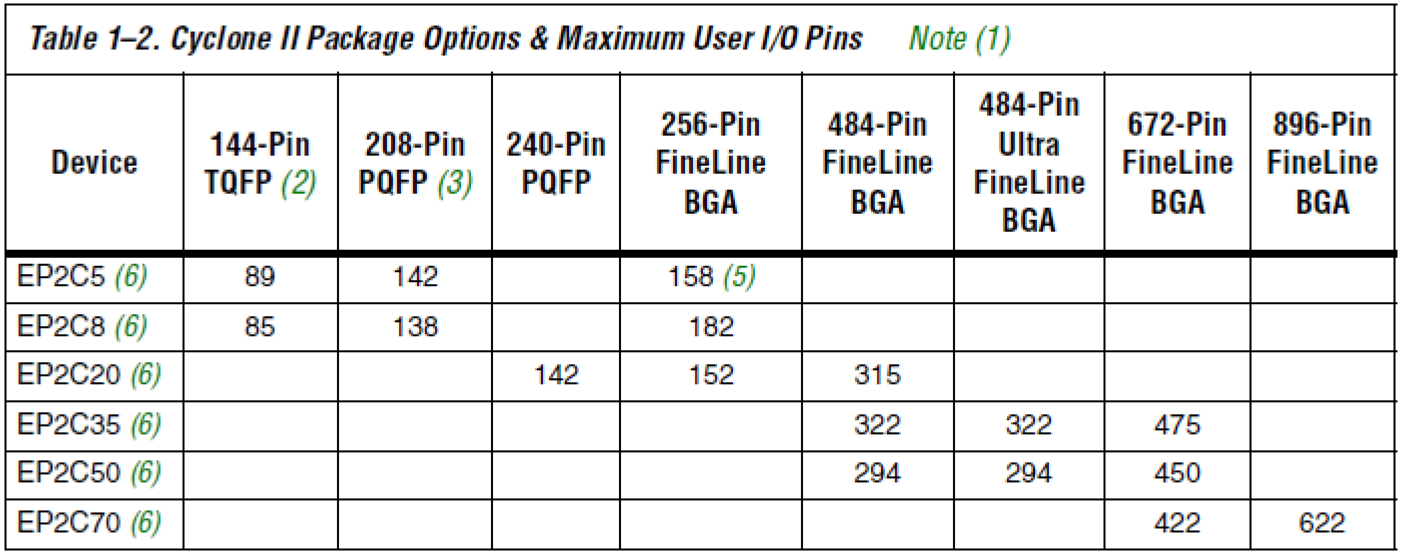


Table 1–2. Cyclone II Package Options & Maximum User I/O Pins Note (1)

Device	144-Pin TQFP *(2)*	208-Pin PQFP *(3)*	240-Pin PQFP	256-Pin FineLine BGA	484-Pin FineLine BGA	484-Pin Ultra FineLine BGA	672-Pin FineLine BGA	896-Pin FineLine BGA
EP2C5 *(6)*	89	142		158 *(5)*				
EP2C8 *(6)*	85	138		182				
EP2C20 *(6)*			142	152	315			
EP2C35 *(6)*					322	322	475	
EP2C50 *(6)*					294	294	450	
EP2C70 *(6)*							422	622

*From Cyclone II Device Family Data Sheet

Cyclone II Device Resources*

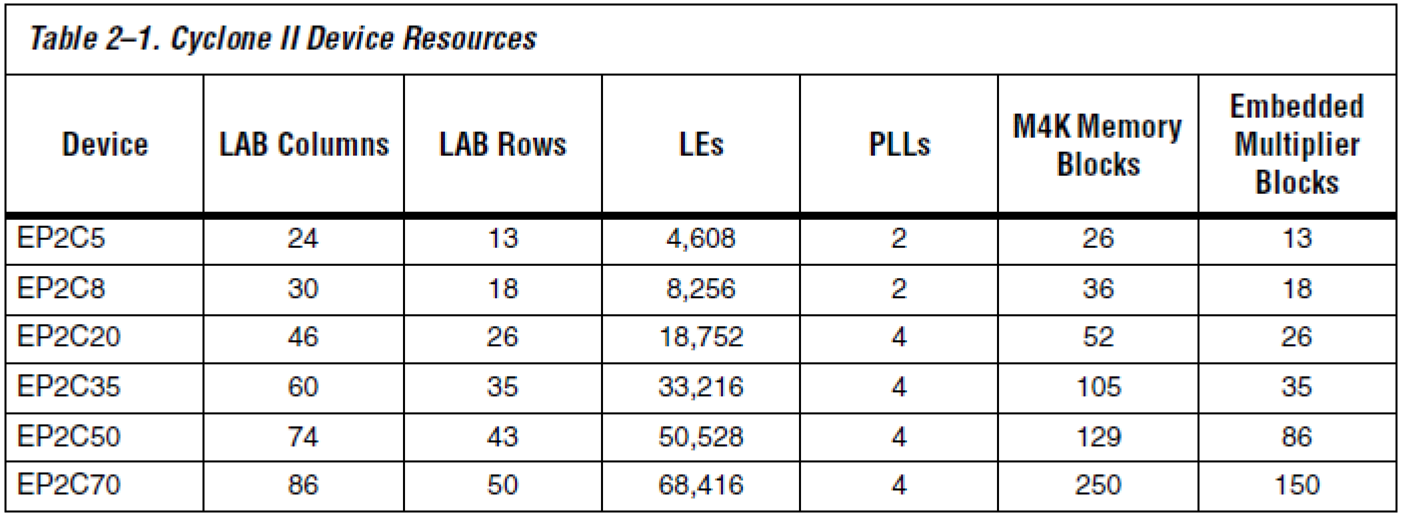


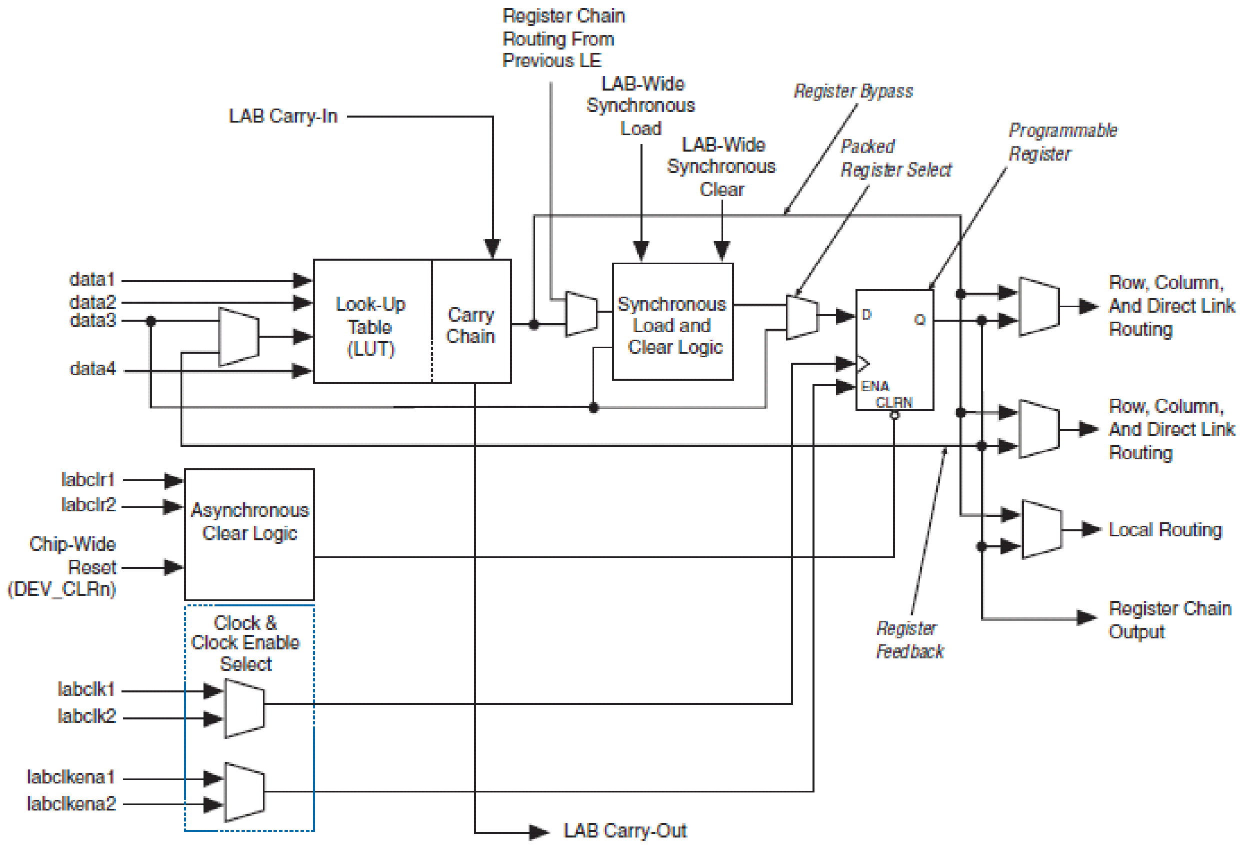
Table 2–1. Cyclone II Device Resources

Device	LAB Columns	LAB Rows	LEs	PLLs	M4K Memory Blocks	Embedded Multiplier Blocks
EP2C5	24	13	4,608	2	26	13
EP2C8	30	18	8,256	2	36	18
EP2C20	46	26	18,752	4	52	26
EP2C35	60	35	33,216	4	105	35
EP2C50	74	43	50,528	4	129	86
EP2C70	86	50	68,416	4	250	150

*From Cyclone II Device Family Data Sheet

Cyclone II Logic Element Architecture*

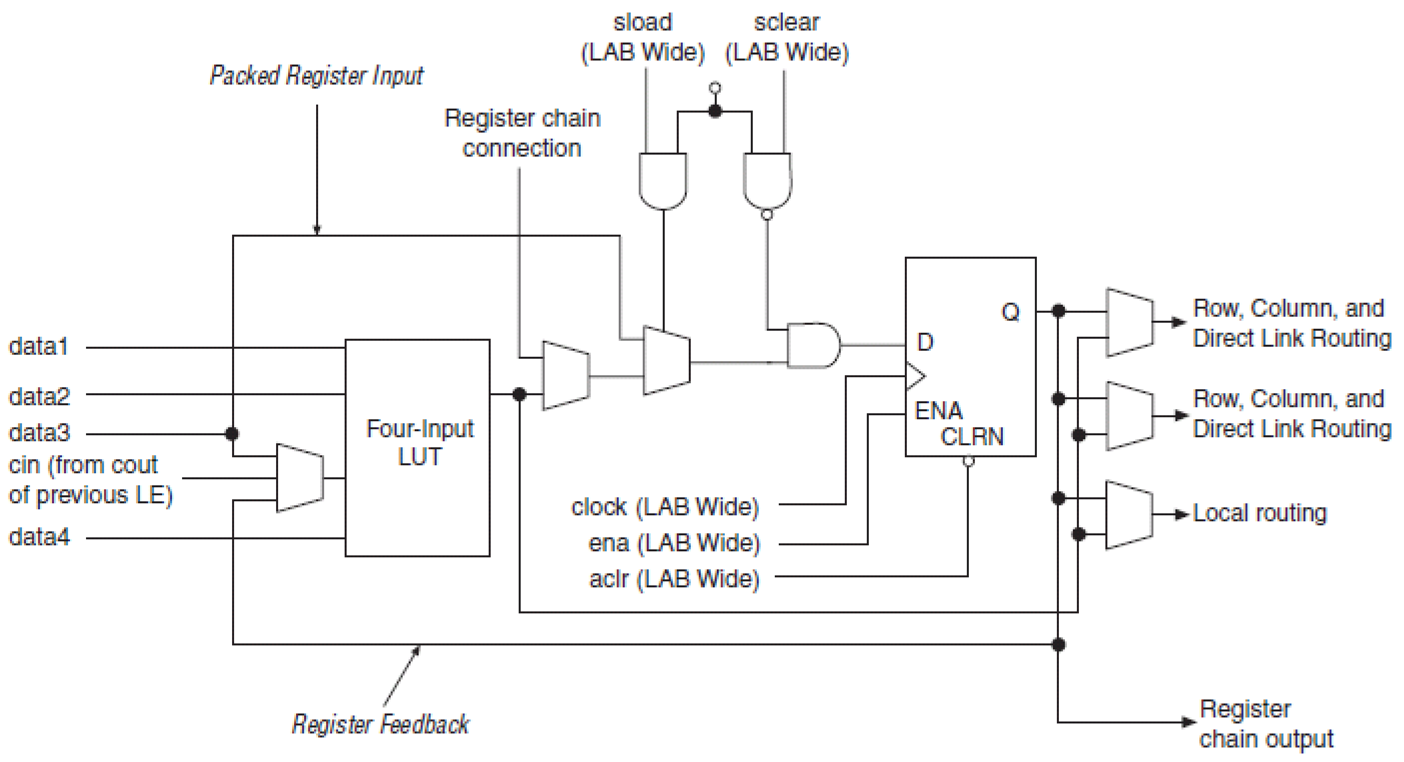
Figure 2–2. Cyclone II LE



*From Cyclone II Device Family Data Sheet

Cyclone II Logic Element in Normal Mode*

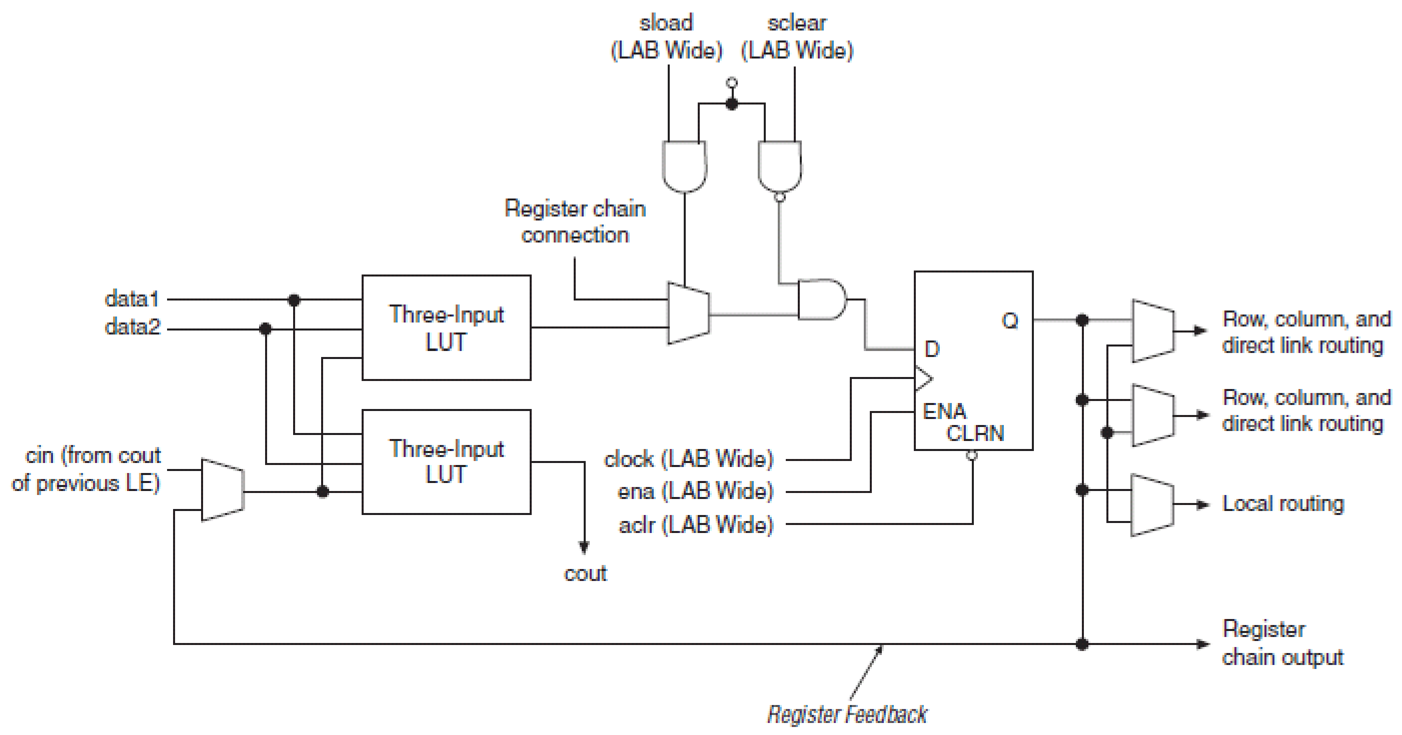
Figure 2–3. LE in Normal Mode



*From Cyclone II Device Family Data Sheet

Cyclone II Logic Element in Arithmetic Mode*

Figure 2–4. LE in Arithmetic Mode



*From Cyclone II Device Family Data Sheet