國立勤益科技大學

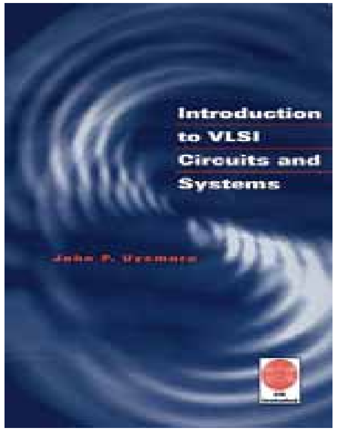
National Chin-Yi University of Technology

Introduction to VLSI Circuits and Systems

積體電路概論

Chapter 12

Arithmetic Circuits in CMOS VLSI



賴秉樑
Dept. of Electronic Engineering
National Chin-Yi University of Technology

Fall 2007

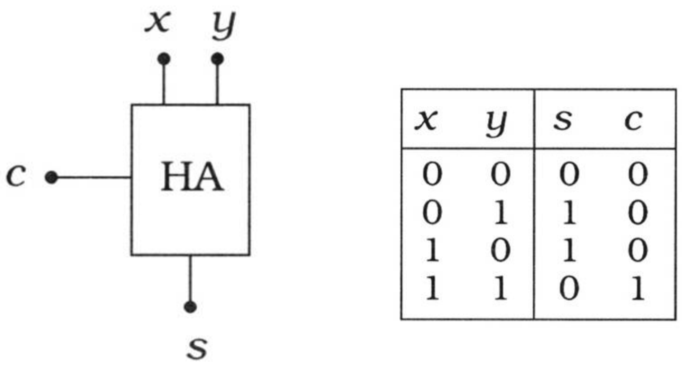
Outline

- Bit Adder Circuits
- Ripple-Carry Adders
- Carry Look-Ahead Adders
- Other High-Speed Adders
- Multipliers

Half-Adder Circuits

- Consider two binary digit *x* and *y*, and the binary sum is denoted by *x* + *y* such that

$$\begin{aligned} 0+0&=0 \\ 0+1&=1 \\ 1+0&=1 \\ 1+1&=10 \end{aligned} \qquad (12.1)$$



x	y	s	c
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Figure 12.1 Half-adder symbol and operation

- A half-adder has 2 inputs (*x* and *y*) and 2 outputs (the sum *s* and the carry-out *c*)

$$\begin{aligned} s &= x \oplus y \\ c &= x \cdot y \end{aligned} \qquad (12.2)$$

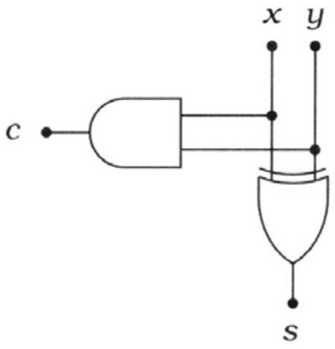


Figure 12.2 Half-adder logic diagram

Full-Adder Circuits

- Adding *n*-bit binary words

$$\begin{array}{rr} & a_3a_2a_1a_0 \\ + & b_3b_2b_1b_0 \\ \hline c_4 & s_3s_2s_1s_0 \end{array} \qquad (12.3)$$

- In the standard carry algorithm, each of the *i*-th columns (*i* = 0, 1, 2, 3) operates according to the full-adder equation

$$\begin{array}{rr} & c_i \\ & a_i \\ + & b_i \\ \hline c_{i+1} & s_i \end{array} \qquad (12.4)$$

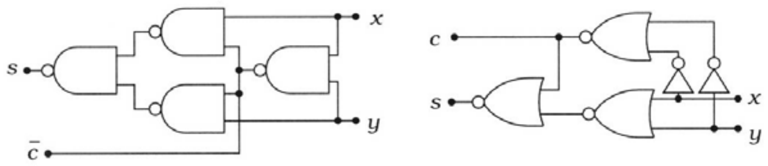
- Expressions for the network are

$$s_i = a_i \oplus b_i \oplus c_i \qquad (12.5)$$

$$c_{i+1} = a_i \cdot b_i + c_i \cdot (a_i \oplus b_i)$$

or

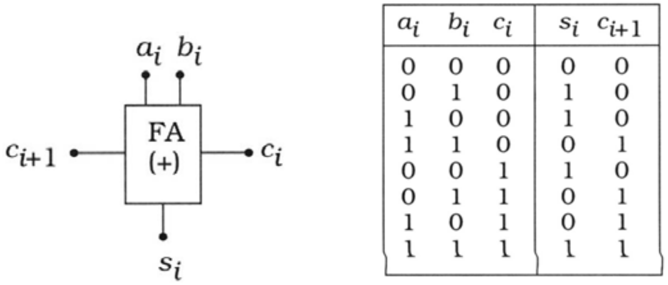
$$c_{i+1} = a_i b_i + c_i \cdot (a_i + b_i) \qquad (12.6)$$



(a) NAND2 logic

(b) NOR-based network

Figure 12.3 Alternate half-adder logic networks



a_i	b_i	c_i	s_i	c_{i+1}
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Figure 12.4 Full-adder symbol and function table

Complementary Pass-transistor Logic

- Dual-rail complementary pass-transistor logic (CPL)

$$a_i \oplus b_i \ \ and \ \ \overline{a_i \oplus b_i} \qquad (12.7)$$

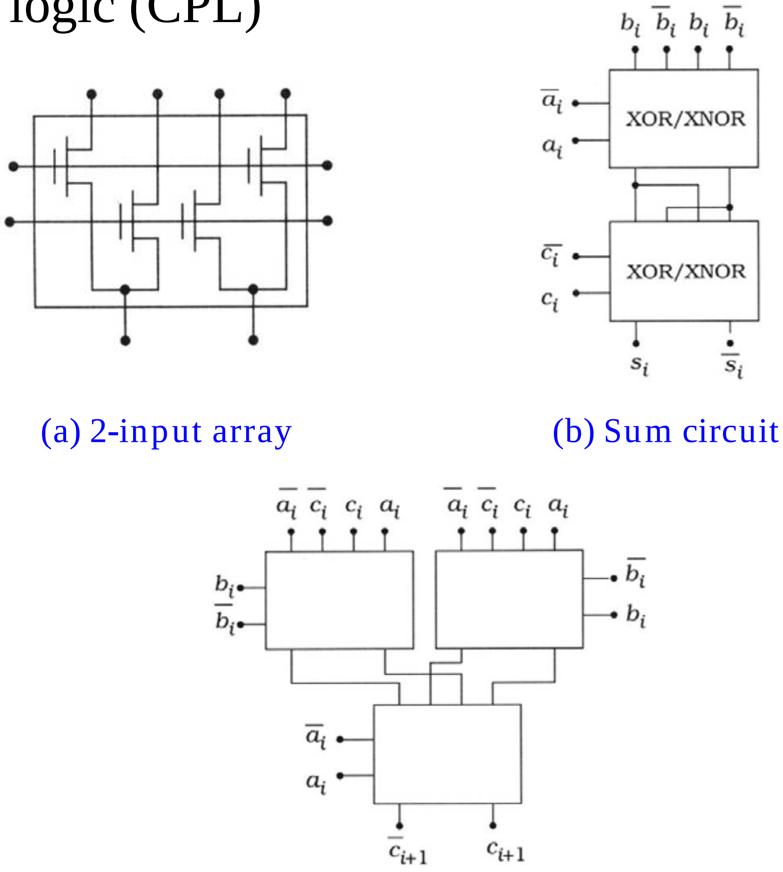
$$s_n = \overline{(a_i \oplus b_i)} \cdot c_i + (a_i \oplus b_i) \cdot \overline{c_i} \qquad (12.8)$$

$$\overline{a_i} \cdot b_i + \overline{b_i} \cdot \overline{c_i} \qquad (12.9)$$

$$\overline{b_i} \cdot c_i + a_i \cdot b_i \qquad (12.10)$$

$$\overline{a_i} \cdot \overline{b_i} + b_i \cdot \overline{c_i} \qquad (12.11)$$

$$b_i \cdot c_i + a_i \cdot \overline{b_i} \qquad (12.12)$$



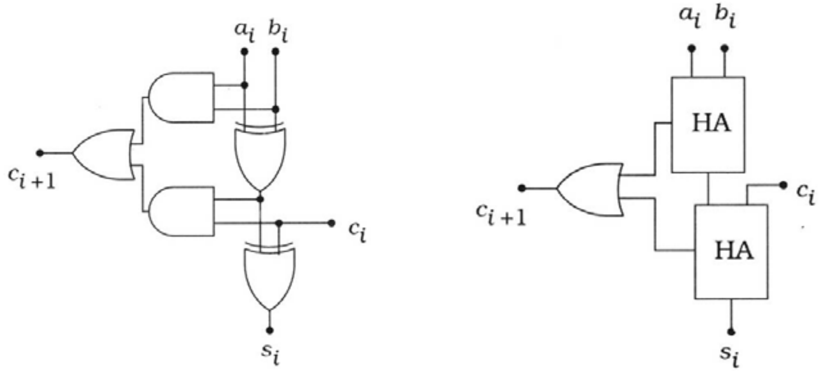
(a) 2-input array

(b) Sum circuit

(c) Carry circuit

Figure 12.5 CPL full-adder design

Full-Adder Circuits (1/2)



(a) Gate-level logic

(b) HA-based design

Figure 12.6 Full-adder logic networks

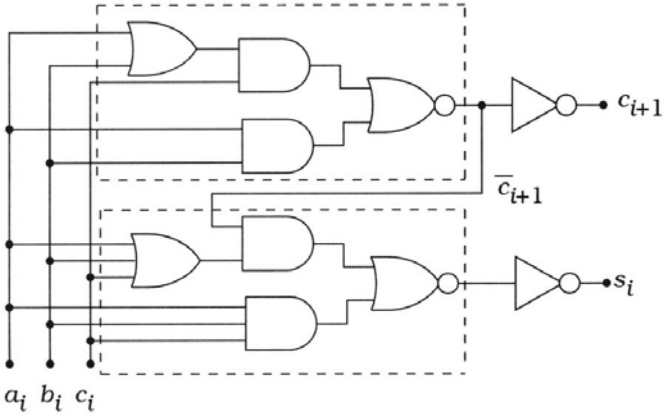
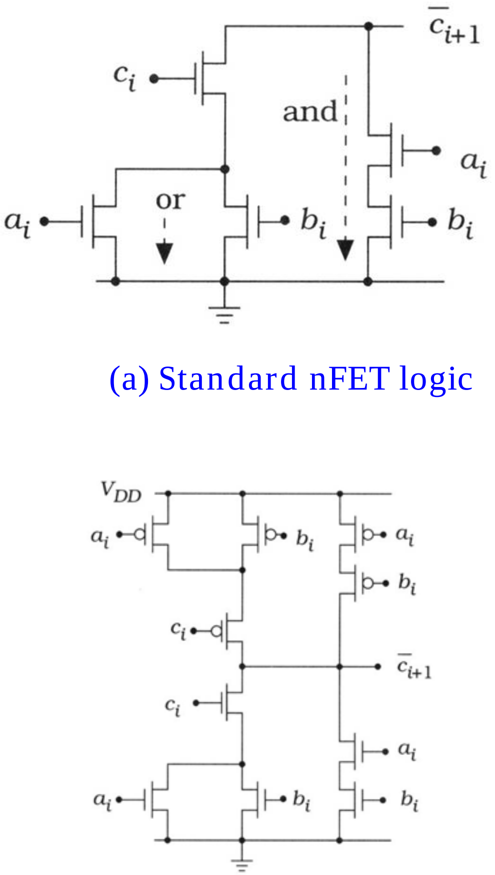


Figure 12.7 AOI full-adder logic



(a) Standard nFET logic

(b) Mirror circuit

Figure 12.8 Evolution of carry-out circuit

Full-Adder Circuits (2/2)

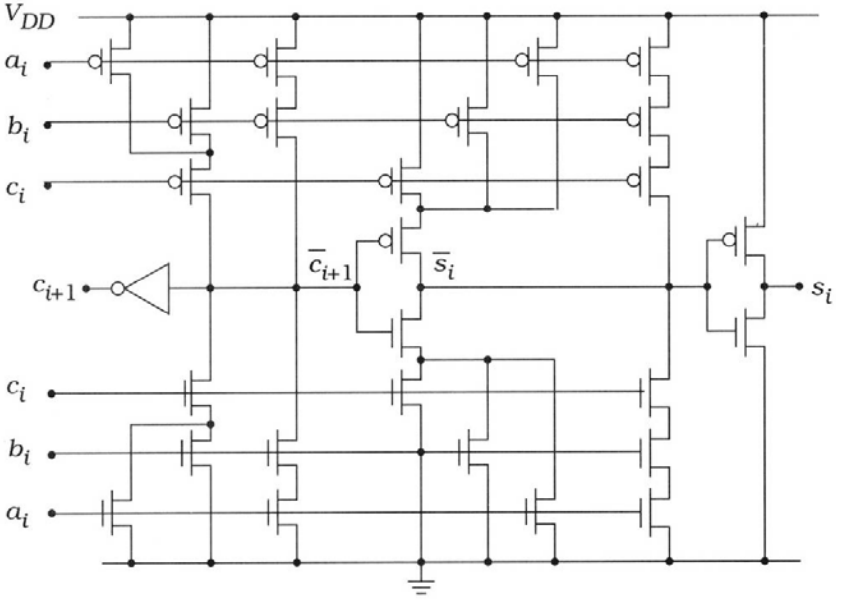


Figure 12.9 Mirror AOI CMOS full-adder

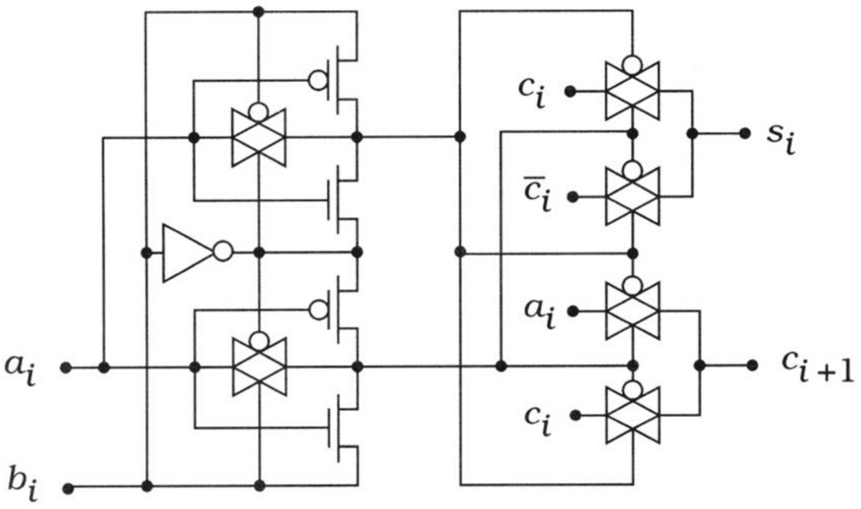


Figure 12.10 Transmission-gate full-adder circuit

Outline

- ❒ Bit Adder Circuits
- ❒ Ripple-Carry Adders
- ❒ Carry Look-Ahead Adders
- ❒ Other High-Speed Adders
- ❒ Multipliers

Ripple-Carry Adders

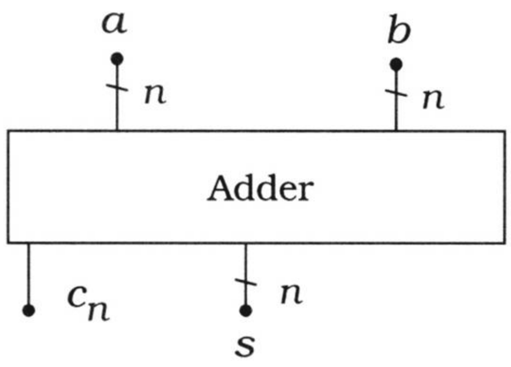


Figure 12.11 An n-bit adder

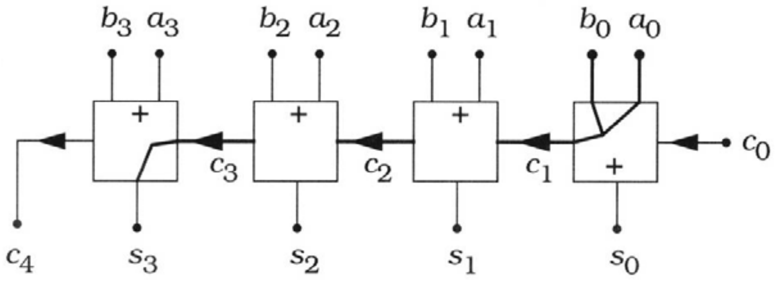


Figure 12.13 Worst-case delay through the 4-bit ripple adder

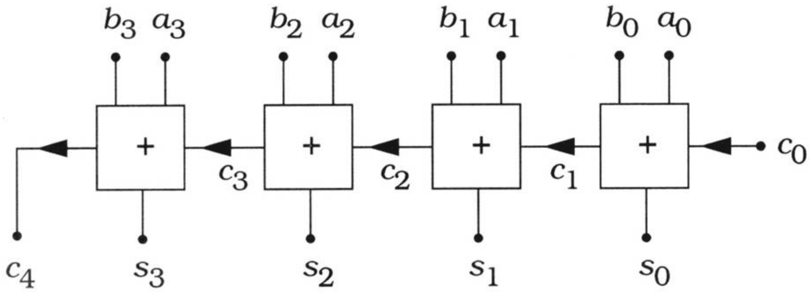


Figure 12.12 A 4-bit ripple-carry adder

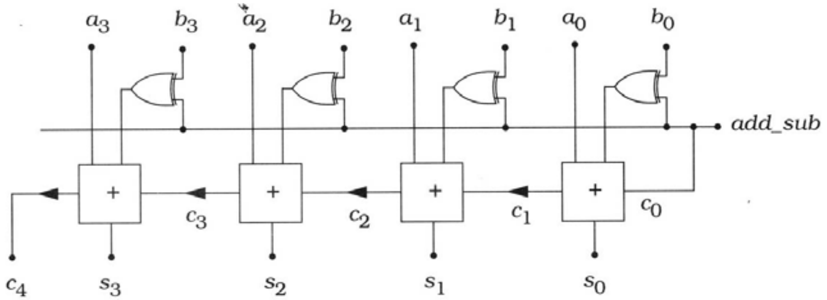
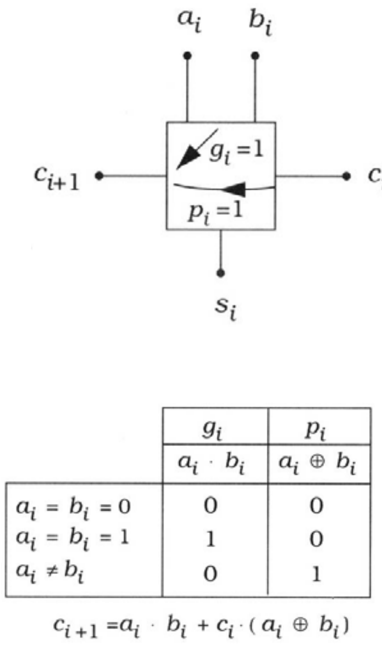


Figure 12.14 4-ibt adder-subtractor circuit

Outline

- ❒ Bit Adder Circuits
- ❒ Ripple-Carry Adders
- ❒ Carry Look-Ahead Adders
- ❒ Other High-Speed Adders
- ❒ Multipliers

Carry Look-Ahead Adders (1/3)



	g_i	p_i
	$a_i \cdot b_i$	$a_i \oplus b_i$
$a_i = b_i = 0$	0	0
$a_i = b_i = 1$	1	0
$a_i \neq b_i$	0	1

$c_{i+1} = a_i \cdot b_i + c_i \cdot (a_i \oplus b_i)$

Figure 12.15 Basis of the carry look-ahead algorithm

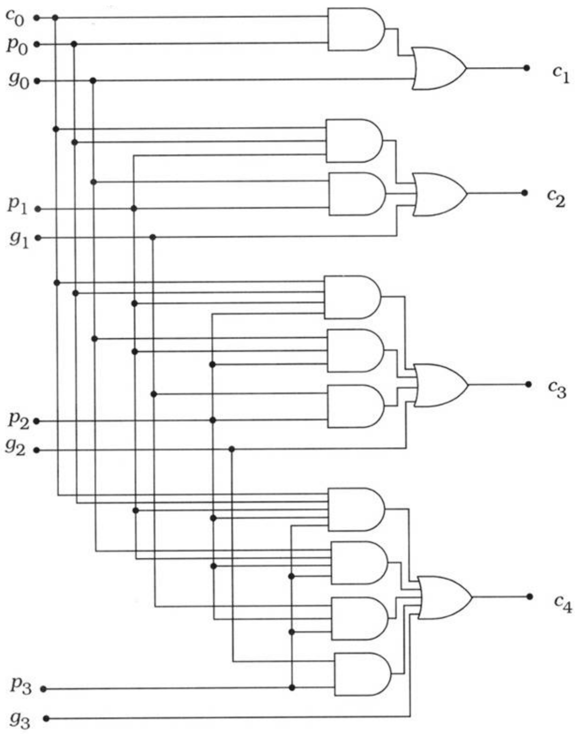


Figure 12.16 Logic network for 4-bit CLA carry bits

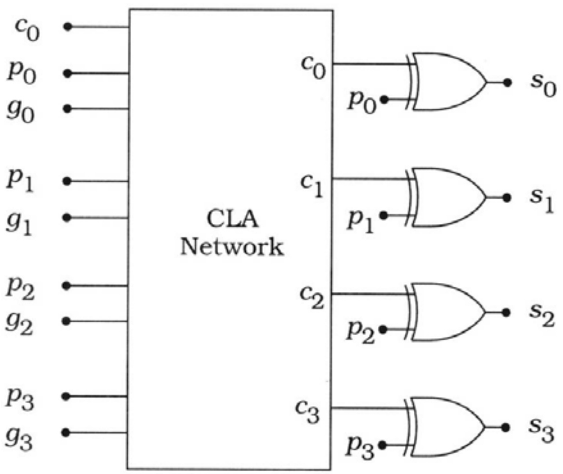
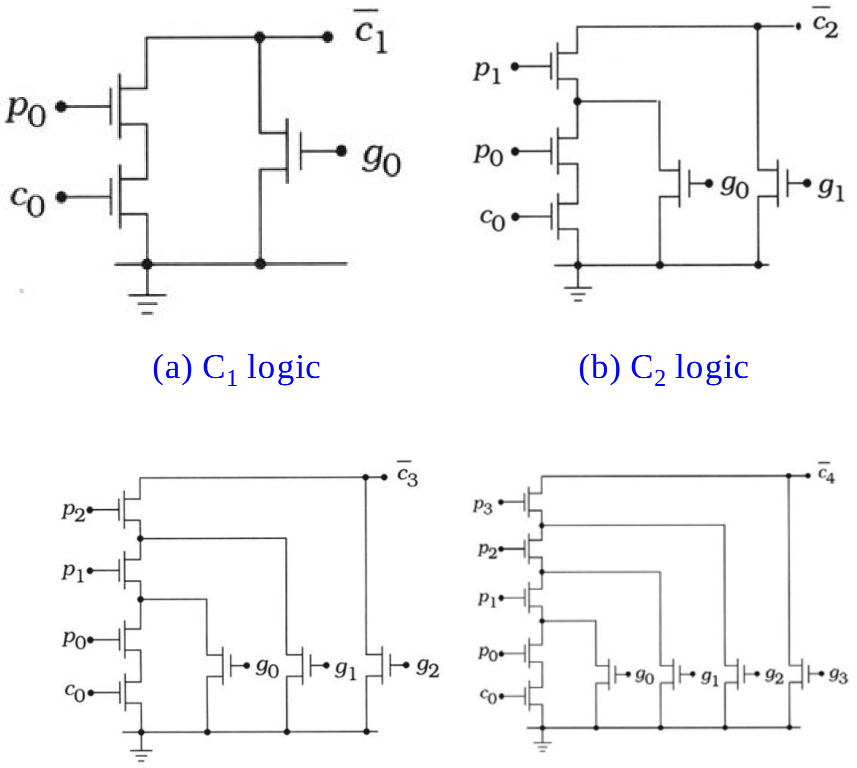


Figure 12.17 Sum calculation using the CLA network

Carry Look-Ahead Adders (2/3)



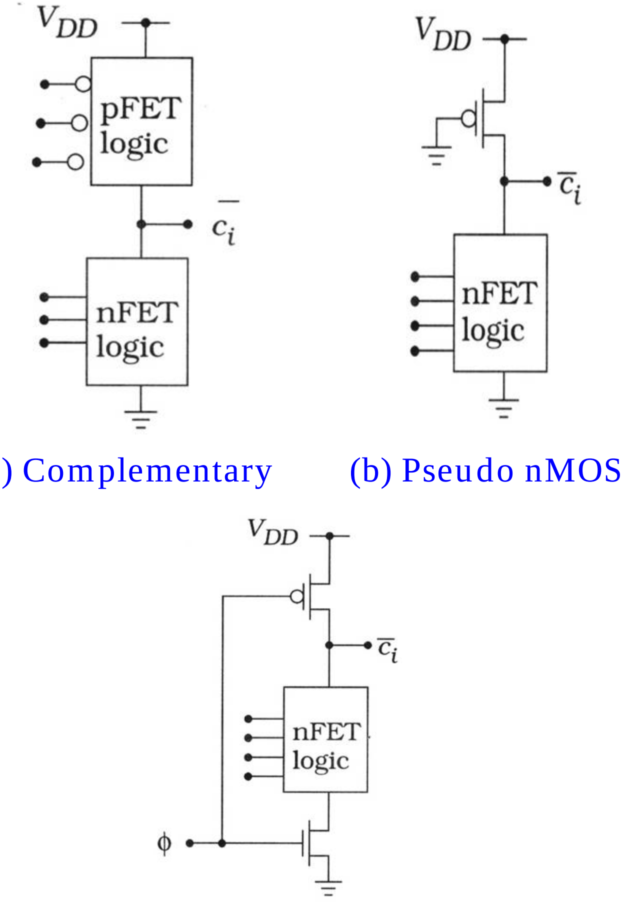
(a) C_1 logic

(b) C_2 logic

(c) C_3 logic

(d) C_4 logic

Figure 12.18 nFET logic arrays for the CLA terms



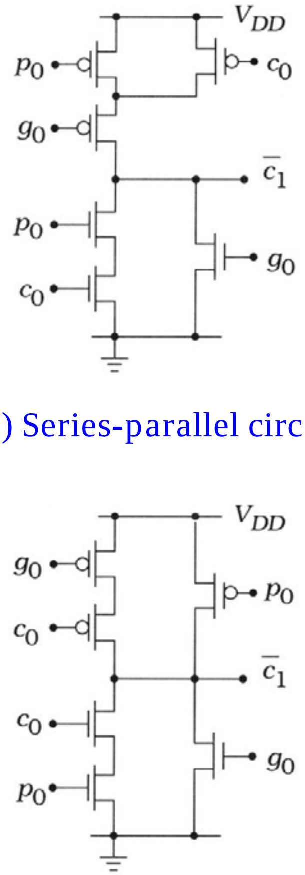
(a) Complementary

(b) Pseudo nMOS

(c) Dynamic

Figure 12.19 Possible uses of the nFET logic arrays in Figure 12.18

Carry Look-Ahead Adders (3/3)



(a) Series-parallel circuit

(b) Mirror equivalent

Figure 12.20 Static CLA mirror circuit

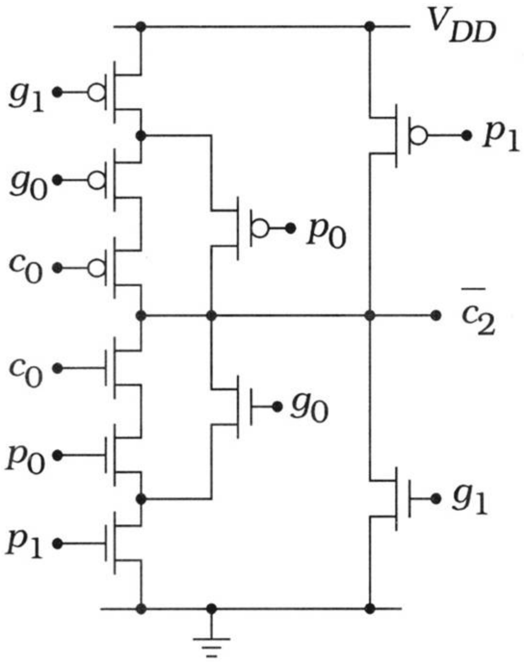


Figure 12.21 Static mirror circuit for C_2

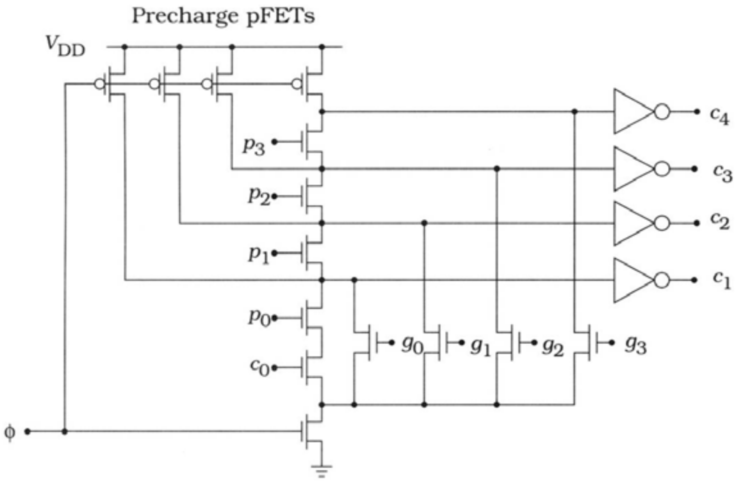
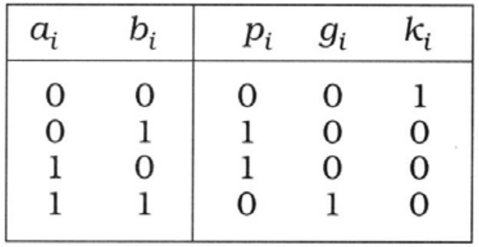


Figure 12.22 MODL carry circuit

Manchester Carry Chains



a_i	b_i	p_i	g_i	k_i
0	0	0	0	1
0	1	1	0	0
1	0	1	0	0
1	1	0	1	0

Figure 12.23 Propagate, generate, and carry-kill values

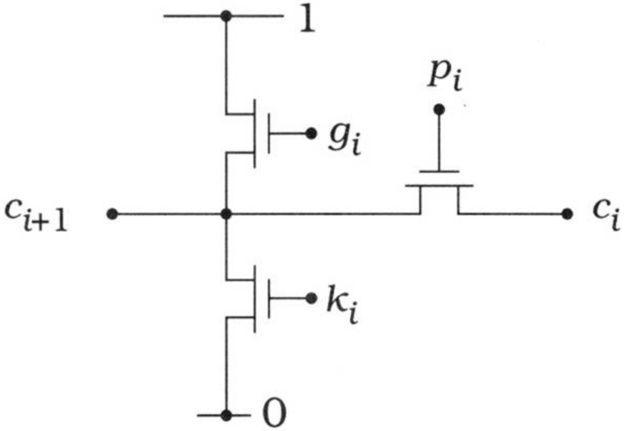
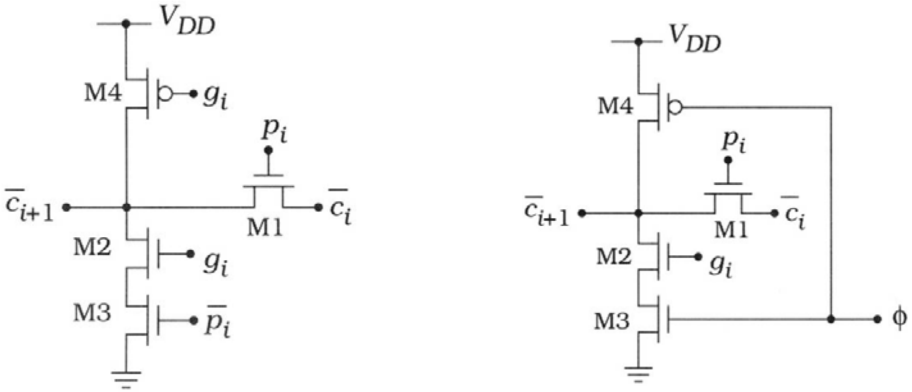


Figure 12.24 Switching network for the carry-out equation



(a) Static circuit

(b) Dynamic circuit

Figure 12.25 Manchester circuit styles

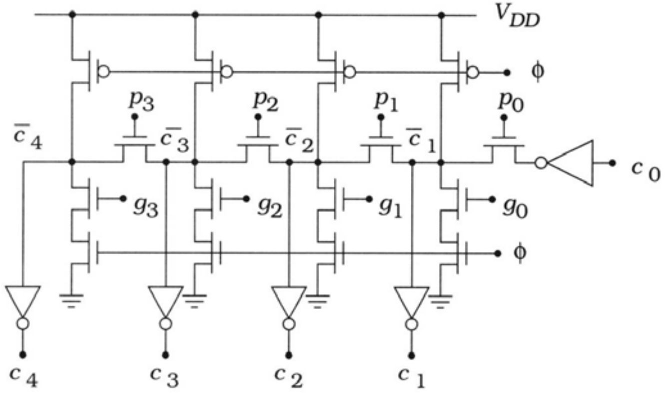


Figure 12.26 Dynamic Manchester carry chain

Extension to Wide Adders (1/2)

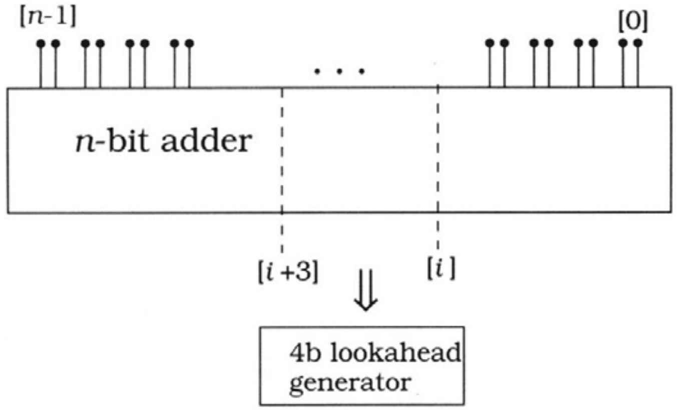
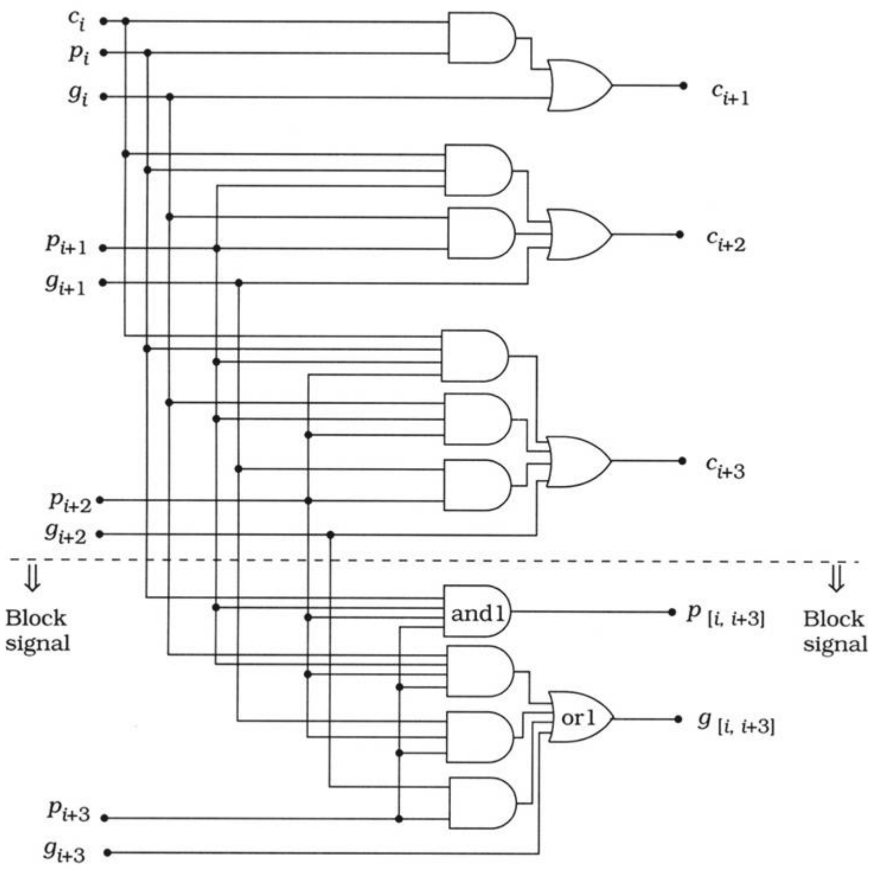


Figure 12.27 An n-bit adder network



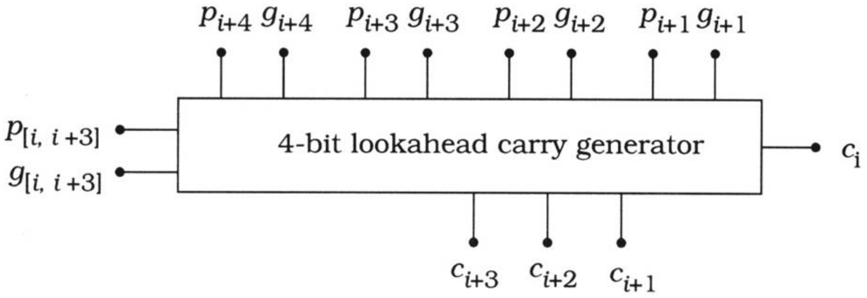
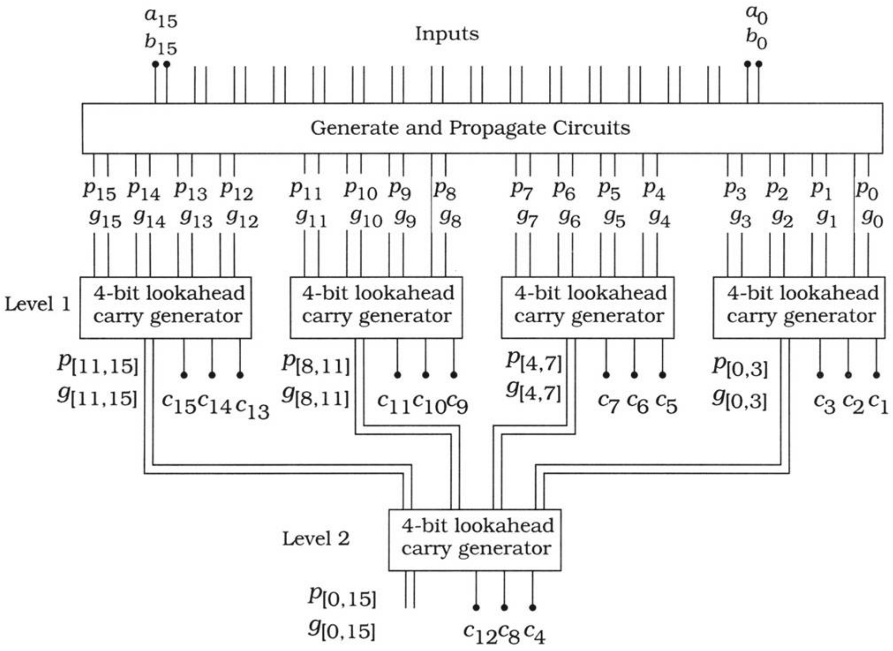


Figure 12.28 4-bit lookahead carry generator signals

Figure 12.29 Block lookahead generator logic

Extension to Wide Adders (2/2)



a_{15}
b_{15}
Inputs
a_0
b_0
Generate and Propagate Circuits
p_{15} p_{14} p_{13} p_{12} p_{11} p_{10} p_9 p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0
g_{15} g_{14} g_{13} g_{12} g_{11} g_{10} g_9 g_8 g_7 g_6 g_5 g_4 g_3 g_2 g_1 g_0
Level 1
4-bit lookahead carry generator
$p_{[11,15]}$
$g_{[11,15]}$
$c_{15}c_{14}c_{13}$
$p_{[8,11]}$
$g_{[8,11]}$
$c_{11}c_{10}c_9$
$p_{[4,7]}$
$g_{[4,7]}$
c_7 c_6 c_5
$p_{[0,3]}$
$g_{[0,3]}$
c_3 c_2 c_1
Level 2
4-bit lookahead carry generator
$p_{[0,15]}$
$g_{[0,15]}$
$c_{12}c_8c_4$

Figure 12.30 Multilevel CLA block scheme for a 16-bit adder

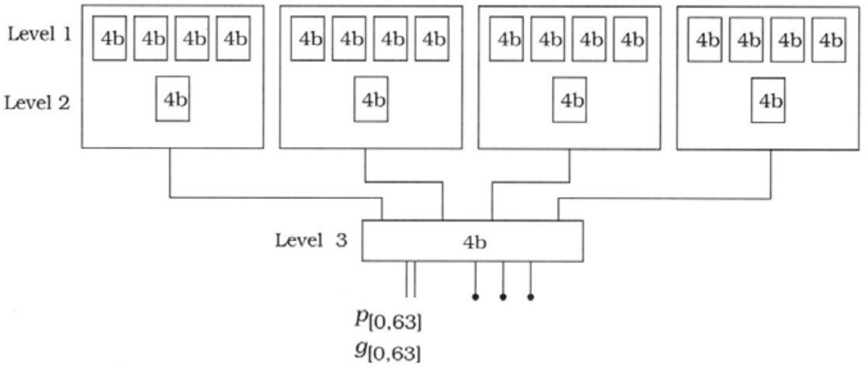
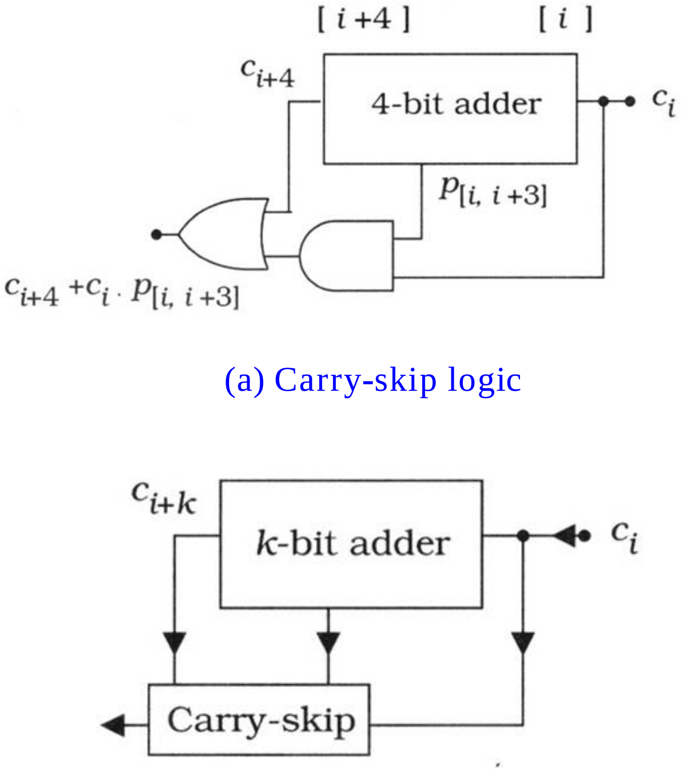


Figure 12.31 64-bit CLA architecture

Outline

- ❒ Bit Adder Circuits
- ❒ Ripple-Carry Adders
- ❒ Carry Look-Ahead Adders
- ❒ Other High-Speed Adders
- ❒ Multipliers

Carry-Skip Circuits



(a) Carry-skip logic

(b) Generalization

Figure 12.32 Carry-skip circuitry

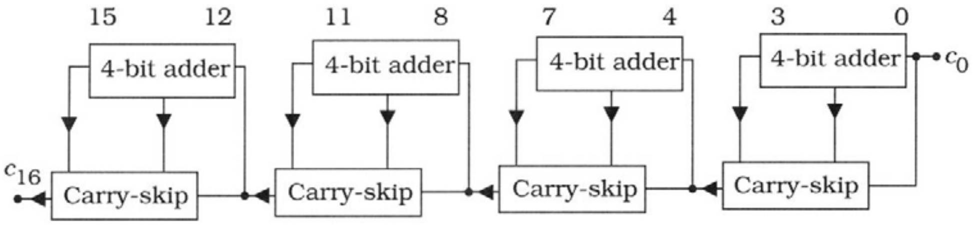


Figure 12.33 A 16-bit adder using carry-skip circuits

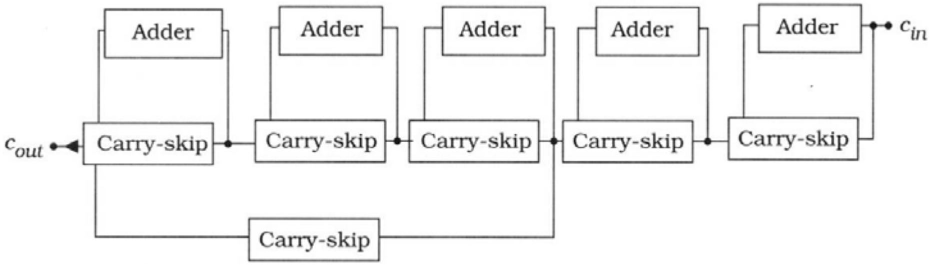


Figure 12.34 A 2-level carry-skip adder

Carry-Select Adders

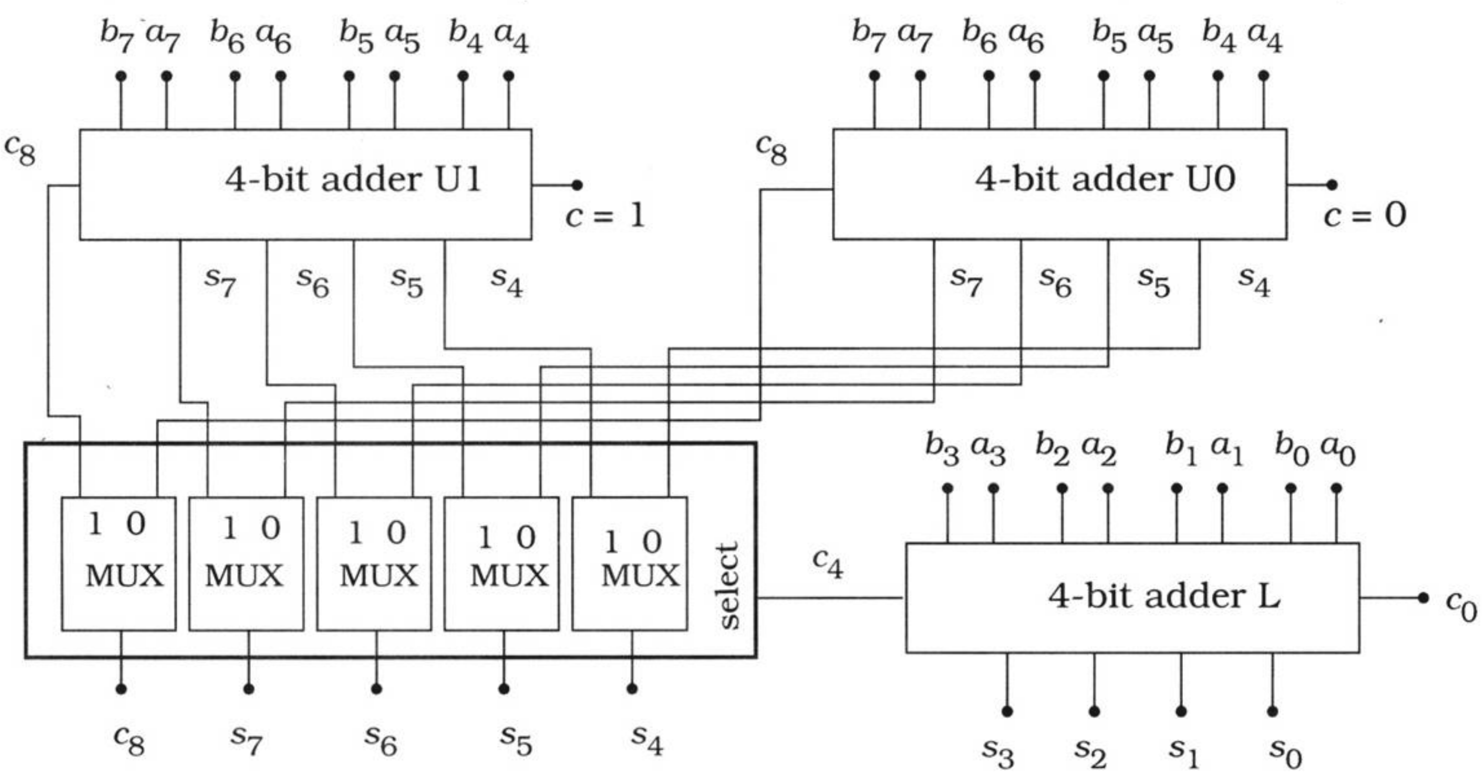
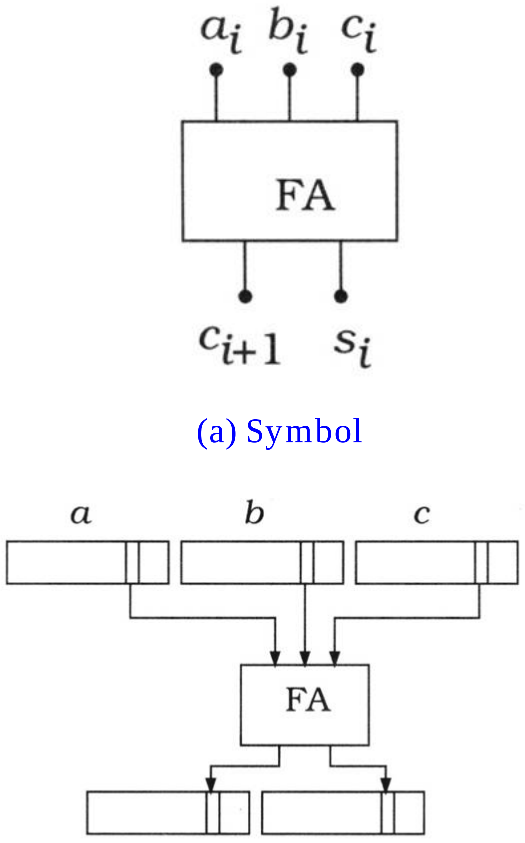


Figure 12.35 8-bit carry-select adder

Carry-Save Adders



(a) Symbol

(b) 3-to-2 reduction

Figure 12.36 Basic of a carry-save adder

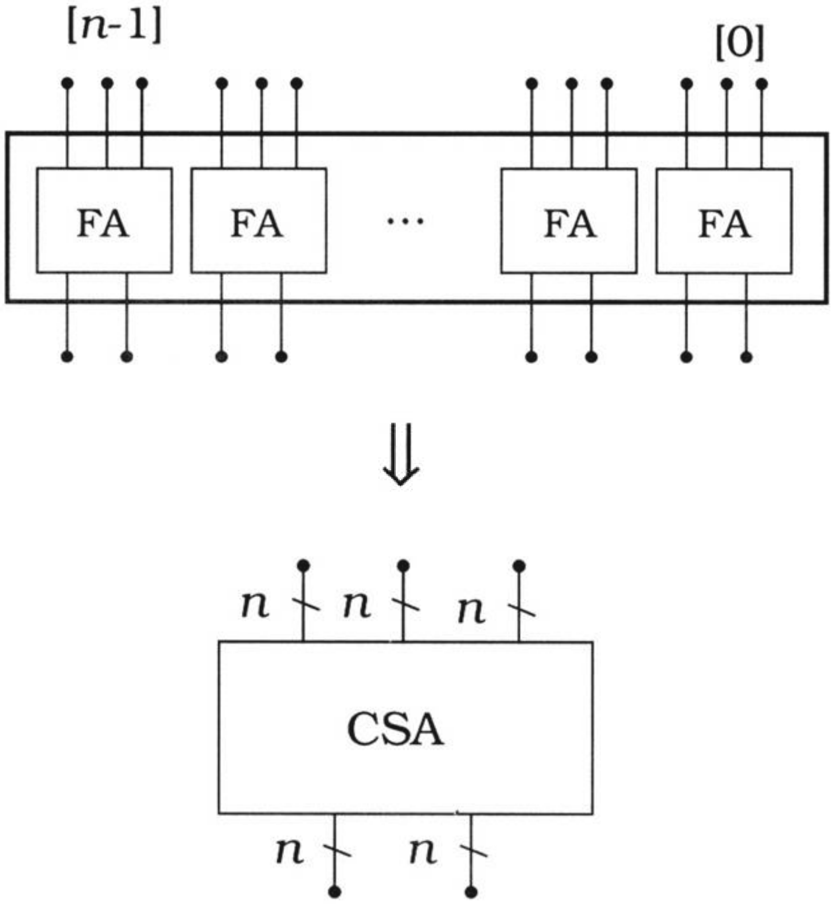


Figure 12.37 Creation of an n-bit carry-save adder

Outline

- ❒ Bit Adder Circuits
- ❒ Ripple-Carry Adders
- ❒ Carry Look-Ahead Adders
- ❒ Other High-Speed Adders
- ❒ Multipliers

Multipliers (1/2)

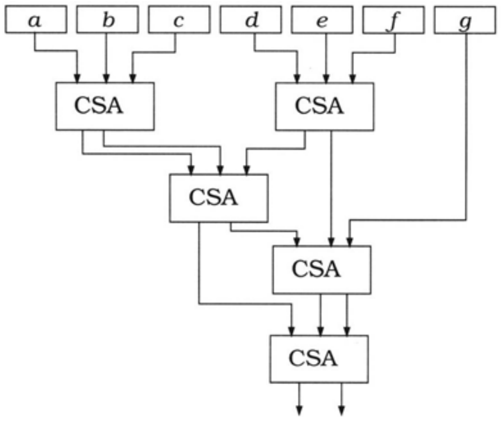
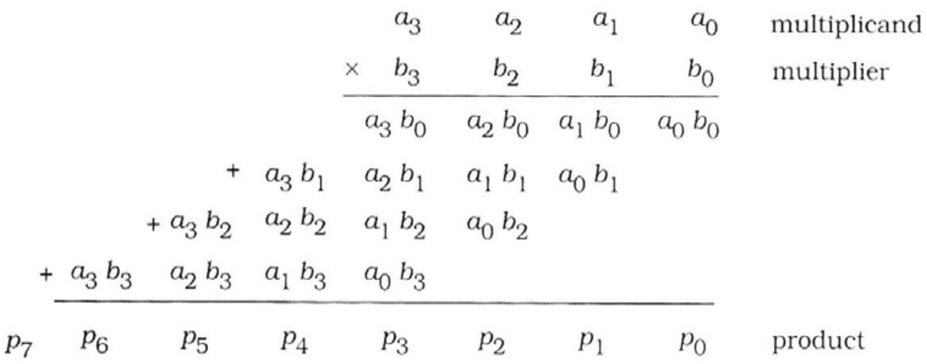


Figure 12.38 A 7-to-2 reduction using carry-save adders



				a_3	a_2	a_1	a_0	multiplicand
				$\times\ b_3$	b_2	b_1	b_0	multiplier
				$a_3 b_0$	$a_2 b_0$	$a_1 b_0$	$a_0 b_0$	
			$+\ a_3 b_1$	$a_2 b_1$	$a_1 b_1$	$a_0 b_1$		
		$+\ a_3 b_2$	$a_2 b_2$	$a_1 b_2$	$a_0 b_2$			
	$+\ a_3 b_3$	$a_2 b_3$	$a_1 b_3$	$a_0 b_3$				
p_7	p_6	p_5	p_4	p_3	p_2	p_1	p_0	product

Figure 12.40 Multiplication of two 4-bit words

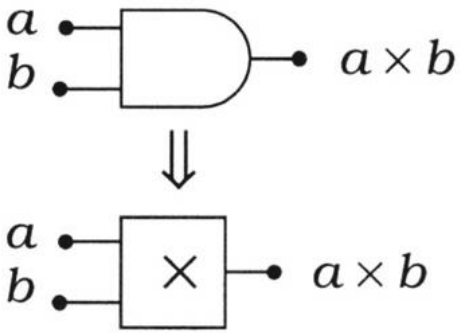


Figure 12.39 Bit-level multiplier

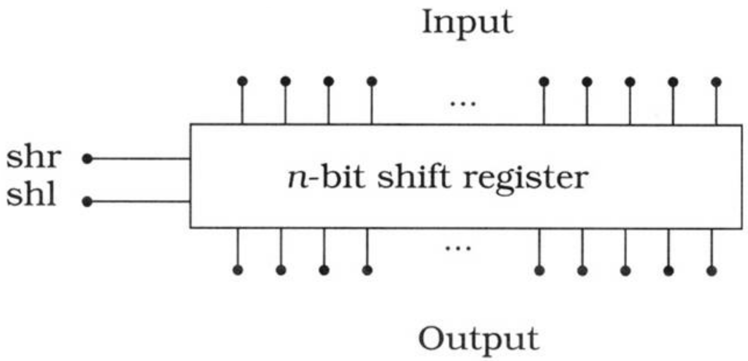


Figure 12.41 Shift register for multiplication or division by a factor of 2

Multipliers (2/2)

$$\begin{array}{rrrrrrrrrl}
 & & & & a_3 & a_2 & a_1 & a_0 & & \text{multiplicand}\\
 & & & \times & b_3 & b_2 & b_1 & b_0 & & \text{multiplier}\\
\hline
 & & & & (a_3 & a_2 & a_1 & a_0)\times b_0 & & (a\times b_0)\,2^0\\
 & & & (a_3 & a_2 & a_1 & a_0)\times b_1 & & & (a\times b_1)\,2^1\\
 & & (a_3 & a_2 & a_1 & a_0)\times b_2 & & & & (a\times b_2)\,2^2\\
+ & (a_3 & a_2 & a_1 & a_0)\times b_3 & & & & & (a\times b_3)\,2^3\\
\hline
p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0 & & \text{product}
\end{array}$$

Figure 12.42 Alternate view of multiplication process

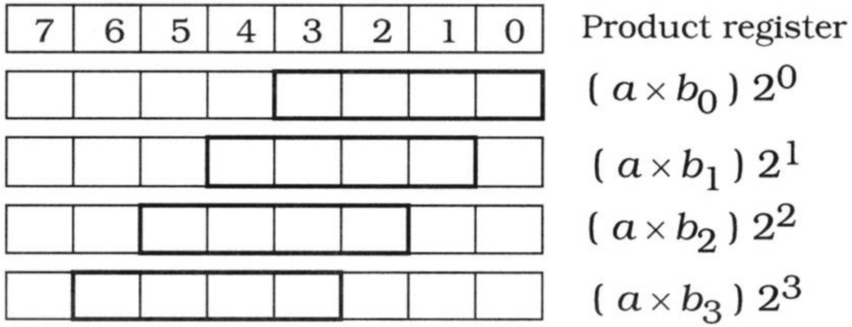
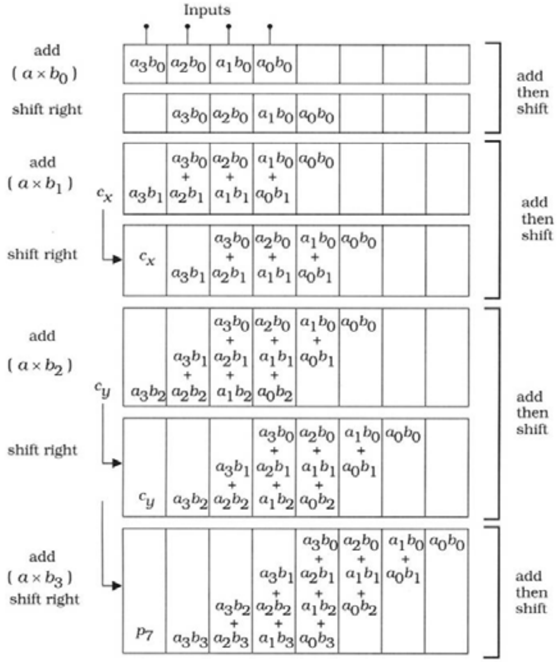


Figure 12.43 Using a product register for multiplication



Inputs
add $(a\times b_0)$
shift right
add then shift
add $(a\times b_1)$
c_x
add $(a\times b_2)$
c_y
add $(a\times b_3)$
p_7
a_3b_0 a_2b_0 a_1b_0 a_0b_0
a_3b_1 a_2b_1 a_1b_1 a_0b_1
a_3b_2 a_2b_2 a_1b_2 a_0b_2
a_3b_3 a_2b_3 a_1b_3 a_0b_3

Figure 12.44 Shift-right multiplication sequence

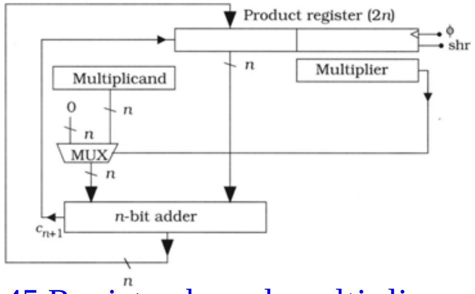


Figure 12.45 Register-based multiplier network

Array Multipliers

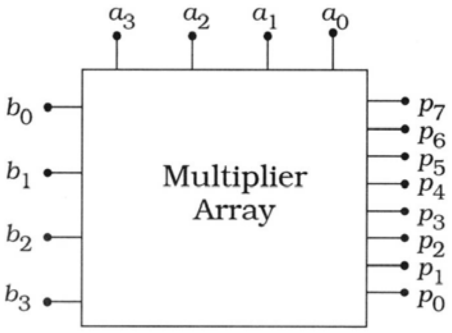


Figure 12.46 An array multiplier

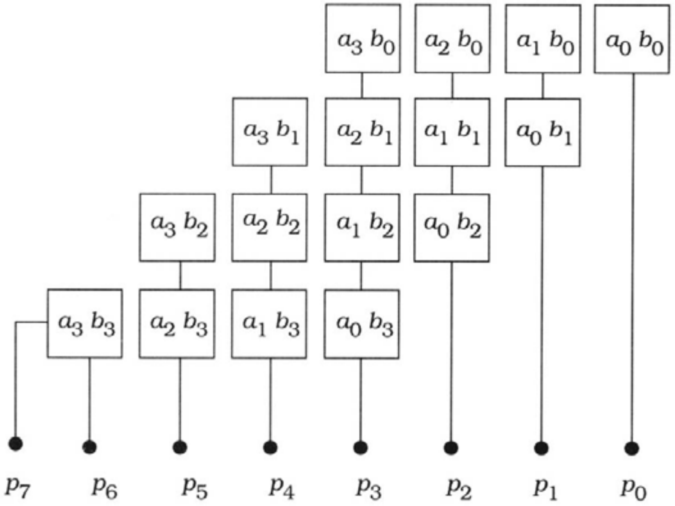


Figure 12.47 Modularized view of the multiplication sequence

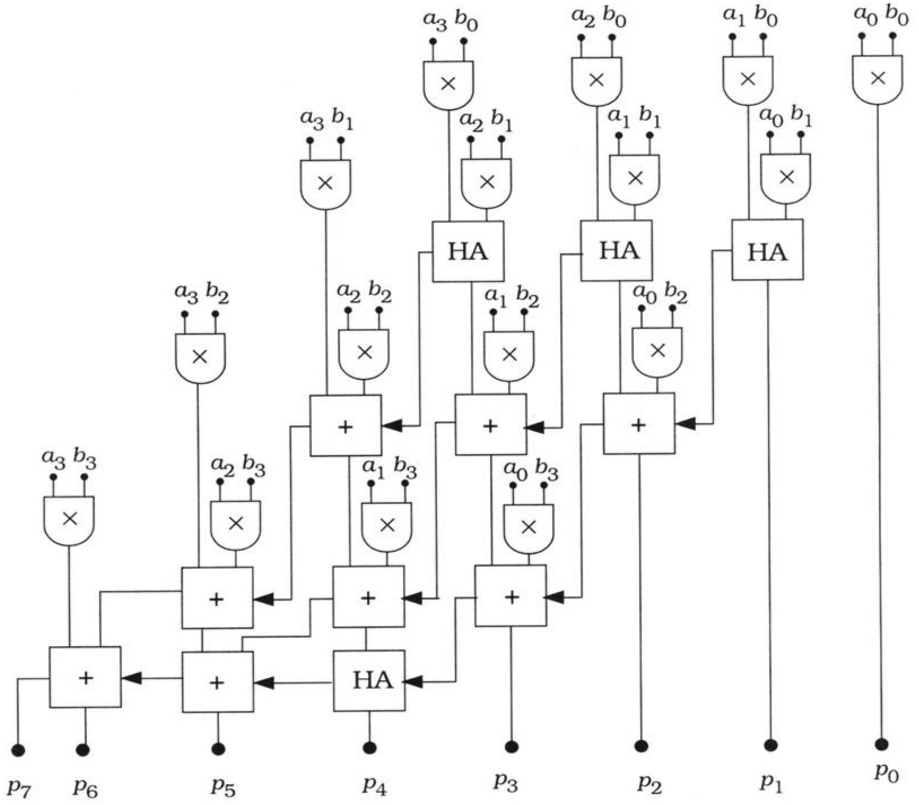


Figure 12.48 Details for a 4 X 4 array multiplier

Other Multipliers

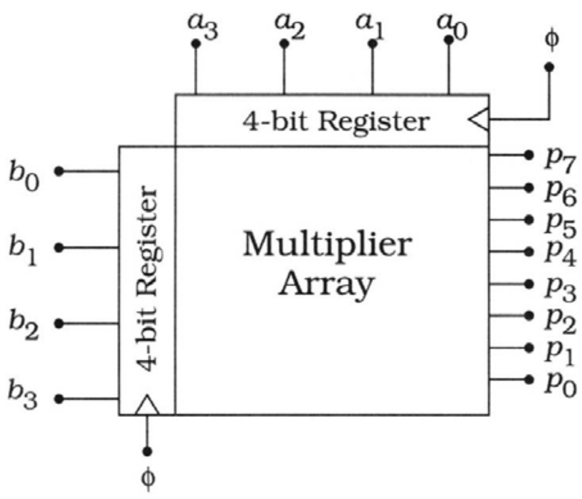
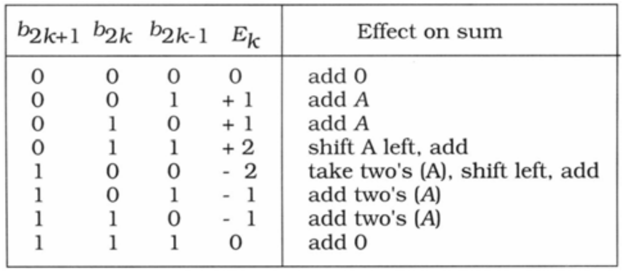


Figure 12.49 Clocked input registers



b_{2k+1}	b_{2k}	b_{2k-1}	E_k	Effect on sum
0	0	0	0	add 0
0	0	1	+ 1	add *A*
0	1	0	+ 1	add *A*
0	1	1	+ 2	shift A left, add
1	0	0	- 2	take two's (A), shift left, add
1	0	1	- 1	add two's (*A*)
1	1	0	- 1	add two's (*A*)
1	1	1	0	add 0

Figure 12.51 Summary of Booth encoded digit operations

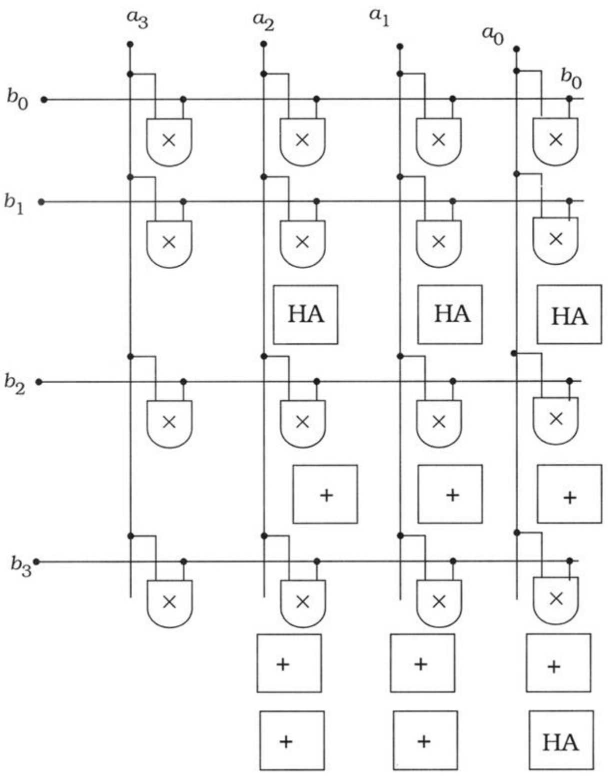


Figure 12.50 Initial cell placement for the array