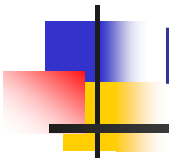
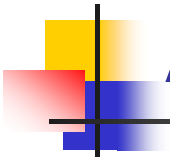


Binary Arithmetic Element



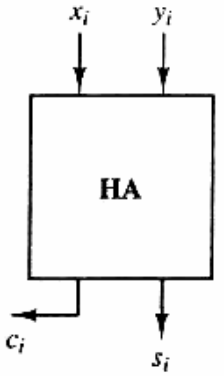
Agenda

- Basic Binary Adder Circuit
- Binary Adder Module
- High Speed Adder Module
- Binary Subtraction Circuit
- Arithmetic Overflow Detection
- Computer Arithmetic Unit

Basic Binary Adder Circuit

- Half Adder

$$s_i = x_i \oplus y_i$$
$$c_i = x_i y_i$$



(a)

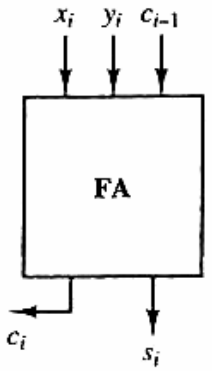
x_i	y_i	c_i	s_i
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

(b)

Basic Binary Adder Circuit

- Full Adder

$$s_i = x_i \oplus y_i \oplus c_{i-1}$$
$$c_i = x_i y_i + x_i c_{i-1} + y_i c_{i-1}$$



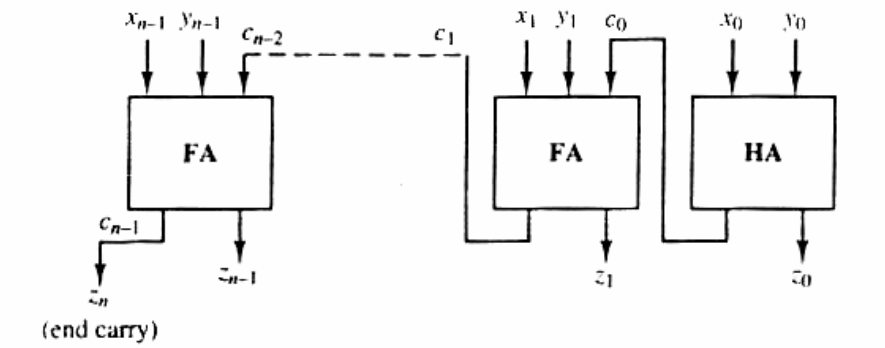
(d)

x_i	y_i	c_{i-1}	c_i	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

(e)

Basic Binary Adder Circuit

- Pseudo Parallel Adder
  - Employs n-1 Full-adder and 1 Half-adder



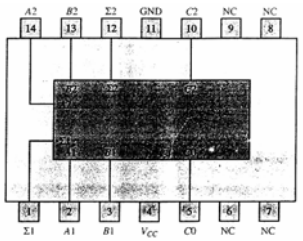
the two input data words. The operation to be performed is

$$\begin{array}{rl} & X \\ + & Y \\ \hline & Z \end{array} \qquad \begin{array}{l} (x_{n-1}x_{n-2}\cdots x_1x_0)_2 \\ +(y_{n-1}y_{n-2}\cdots y_1y_0)_2 \\ \hline (z_nz_{n-1}z_{n-2}\cdots z_1z_0)_2 \end{array}$$

Binary Adder Module

- 7482 Two-bit Adder

Inputs				Outputs					
				When C0 = L			When C0 = H		
A2	A1	B2	B1	C2	Σ2	Σ1	C2	Σ2	Σ1
L	L	L	L	L	L	L	L	L	H
L	L	L	H	L	L	H	L	H	L
L	L	H	L	L	H	L	L	H	H
L	L	H	H	L	H	H	H	L	L
L	H	L	L	L	L	H	L	H	L
L	H	L	H	L	H	L	L	H	H
L	H	H	L	L	H	H	H	L	L
L	H	H	H	H	L	L	H	L	H
H	L	L	L	L	H	L	L	H	H
H	L	L	H	L	H	H	H	L	L
H	L	H	L	H	L	L	H	L	H
H	L	H	H	H	L	H	H	H	L
H	H	L	L	L	H	H	H	L	L
H	H	L	H	H	L	L	H	L	H
H	H	H	L	H	L	H	H	H	L
H	H	H	H	H	H	L	H	H	H



Binary Adder Module

$$
\begin{aligned}
C_1 &= C_0 \cdot A_1 + C_0 \cdot B_1 + A_1 \cdot B_1 \\
\Sigma_1 &= C_0 \cdot \bar{C}_1 + A_1 \cdot \bar{C}_1 + B_1 \cdot \bar{C}_1 + A_1 \cdot B_1 \cdot C_0 \\
&= \bar{C}_1(C_0 + A_1 + B_1) + A_1 \cdot B_1 \cdot C_0 \\
&= (\bar{C}_0 + \bar{A}_1)(\bar{C}_0 + \bar{B}_1)(\bar{A}_1 + \bar{B}_1)(C_0 + A_1 + B_1) + A_1 \cdot B_1 \cdot C_0 \\
&= (\bar{C}_0 + \bar{A}_1 \cdot \bar{B}_1)(\bar{A}_1 + \bar{B}_1)(C_0 + A_1 + B_1) + A_1 \cdot B_1 \cdot C_0 \\
&= [\bar{C}_0(A_1 + B_1) + C_0 \cdot \bar{A}_1 \cdot \bar{B}_1](\bar{A}_1 + \bar{B}_1) + A_1 \cdot B_1 \cdot C_0 \\
&= \bar{C}_0 \cdot \bar{A}_1 \cdot B_1 + \bar{C}_0 \cdot A_1 \cdot \bar{B}_1 + C_0 \cdot \bar{A}_1 \cdot \bar{B}_1 + A_1 \cdot B_1 \cdot C_0 \\
&= C_0 \oplus A_1 \oplus B_1
\end{aligned}
$$

$$
\begin{aligned}
C_2 &= C_1 \cdot A_2 + C_1 \cdot B_2 + A_2 \cdot B_2 \\
\Sigma_2 &= C_1 \oplus A_2 \oplus B_2
\end{aligned}
$$

The 7482 pseudo-parallel adder module.

Binary Adder Module

- 7483 4-Bits Adder Module

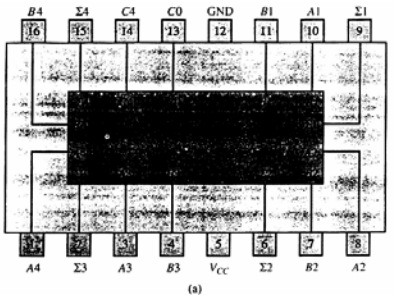
$$
\begin{aligned}
P_i &= (\overline{\bar{B}_i \cdot \bar{A}_i})(A_i + B_i) \\
&= (\bar{A}_i + \bar{B}_i)(A_i + B_i) \\
&= A_i \oplus B_i \\
\Sigma_i &= P_i \oplus C_{i-1} \\
&= A_i \oplus B_i \oplus C_{i-1}
\end{aligned}
$$

and

$$
\begin{aligned}
C_1 &= \overline{\overline{(\bar{C}_0 \cdot \bar{A}_1 \cdot \bar{B}_1)} + \overline{(\bar{A}_1 + \bar{B}_1)}} \\
&= \overline{(\bar{C}_0 \cdot \bar{A}_1 \cdot \bar{B}_1)} \cdot (A_1 + B_1) \\
&= (C_0 + (A_1 \cdot B_1)) \cdot (A_1 + B_1) \\
&= C_0 \cdot A_1 + C_0 \cdot B_1 + A_1 \cdot B_1
\end{aligned}
$$

In a similar manner, we may find

$$C_i = C_{i-1} \cdot A_i + C_{i-1} \cdot B_i + A_i \cdot B_i$$



(a)

High Speed Adder

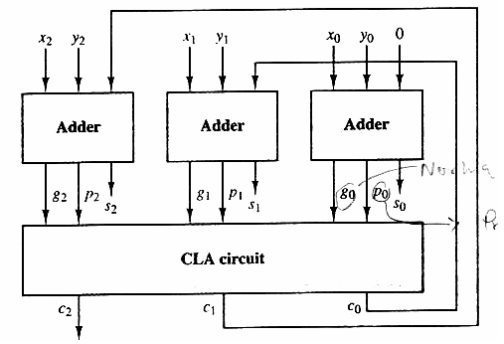
- Fully Parallel Adder

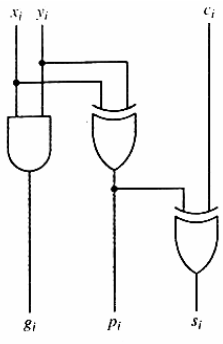
$$
\begin{aligned}
c_0 &= x_0 y_0 \\
c_1 &= x_1 y_1 \bar{c}_0 + x_1 y_1 c_0 + x_1 \bar{y}_1 c_0 + \bar{x}_1 y_1 c_0 \\
&= x_1 y_1 + (x_1 \oplus y_1) c_0 \\
&= x_1 y_1 + (x_1 \oplus y_1)(x_0 y_0) \\
c_2 &= x_2 y_2 + (x_2 \oplus y_2) c_1 \\
&= x_2 y_2 + (x_2 \oplus y_2)[x_1 y_1 + (x_1 \oplus y_1)(x_0 y_0)] \\
&= x_2 y_2 + (x_2 \oplus y_2)(x_1 y_1) + (x_2 \oplus y_2)(x_1 \oplus y_1)(x_0 y_0)
\end{aligned}
$$

High Speed Adder

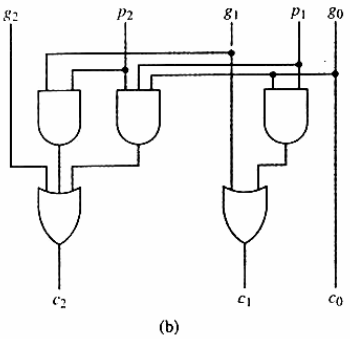
- Carry Look-Ahead Adder

$$
\begin{aligned}
c_0 &= g_0 \\
c_1 &= g_1 + p_1 c_0 \\
&= g_1 + p_1 g_0 \\
c_2 &= g_2 + p_2 c_1 \\
&= g_2 + p_2 g_1 + p_2 p_1 g_0
\end{aligned}
$$





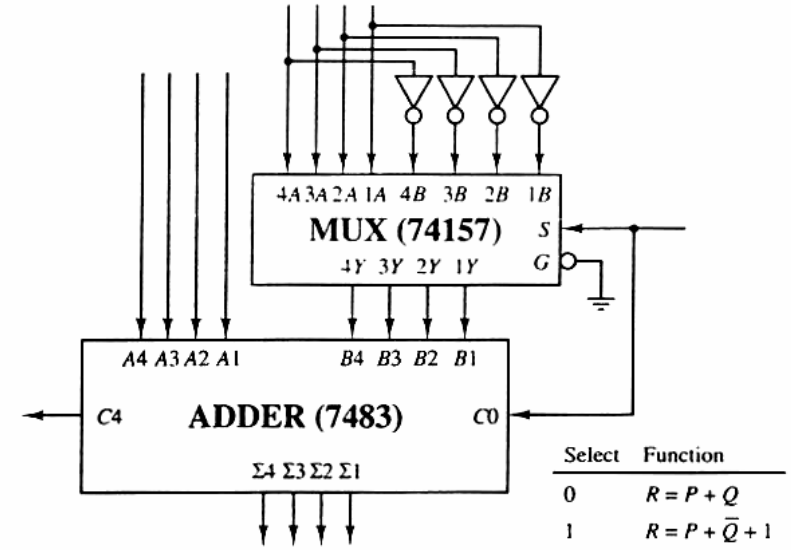
(a)



(b)

Binary Subtraction Circuit

$$
\begin{aligned}
(R)_2 &= (P)_2 - (Q)_2 \\
&= (P)_2 + (-Q)_2 \\
&= (P)_2 + [Q]_2 \\
&= (P)_2 + (\bar{Q})_2 + 1
\end{aligned}
$$



Arithmetic Overflow Detection

$$-2^{n-1} \le N \le 2^{n-1} - 1$$

Adder Inputs			Adder Outputs		Overflo
a_{n-1}	b_{n-1}	c_{n-2}	c_{n-1}	s_{n-1}	V
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	1	0	1
1	1	1	1	1	0

Arithmetic Overflow Detection

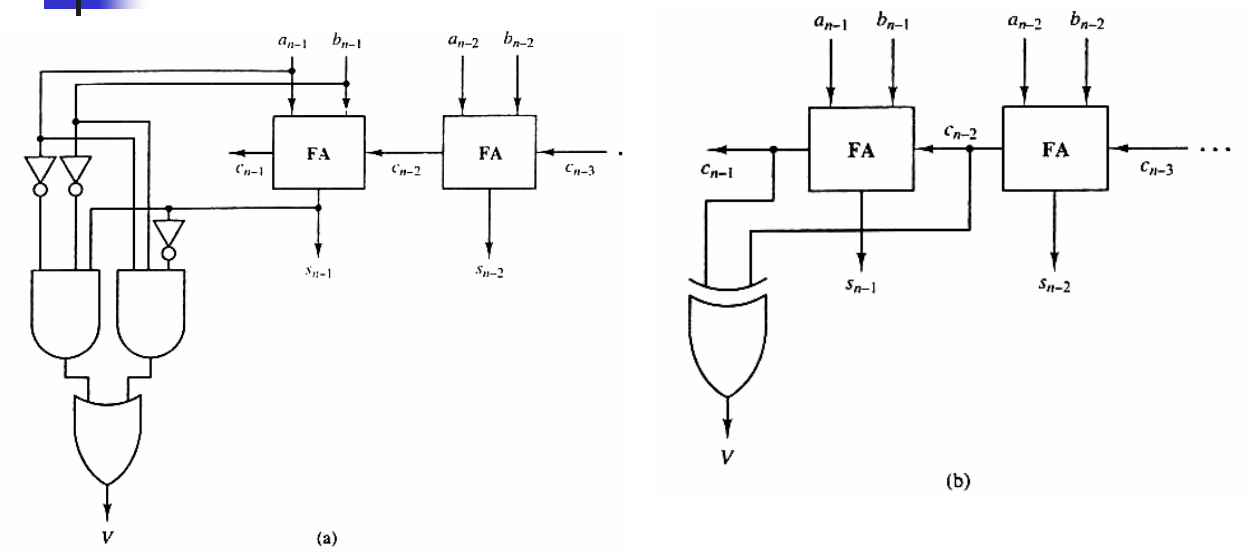
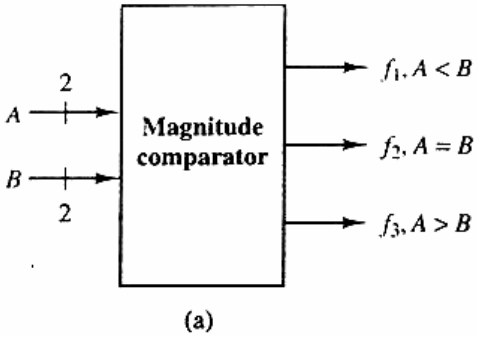


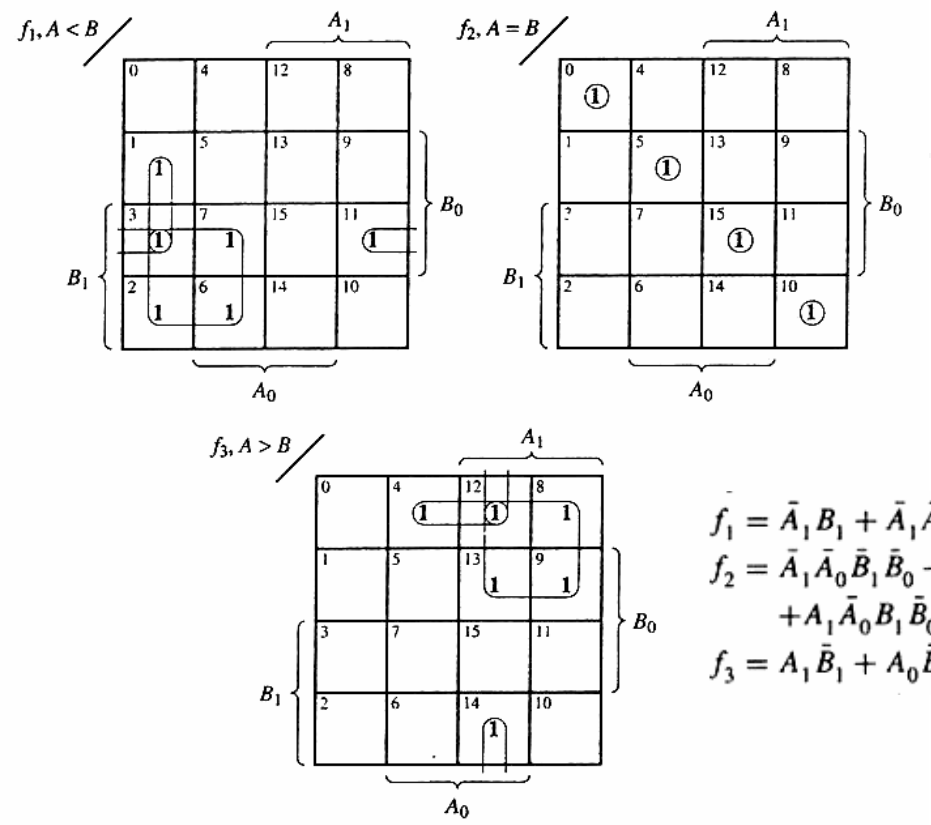
Figure 4.42 Two's complement overflow detection. **(a)** Using sign bits. **(b)** Using carry bits.

Design a comparator that will compare the two words $A = (A_1A_0)_2$ and $B = (B_1B_0)_2$ in binary code.



(a)

i	A_1	A_0	B_1	B_0	f_1	f_2	f_3
0	0	0	0	0	0	1	0
1	0	0	0	1	1	0	0
2	0	0	1	0	1	0	0
3	0	0	1	1	1	0	0
4	0	1	0	0	0	0	1
5	0	1	0	1	0	1	0
6	0	1	1	0	1	0	0
7	0	1	1	1	1	0	0
8	1	0	0	0	0	0	1
9	1	0	0	1	0	0	1
10	1	0	1	0	0	1	0
11	1	0	1	1	1	0	0
12	1	1	0	0	0	0	1
13	1	1	0	1	0	0	1
14	1	1	1	0	0	0	1
15	1	1	1	1	0	1	0



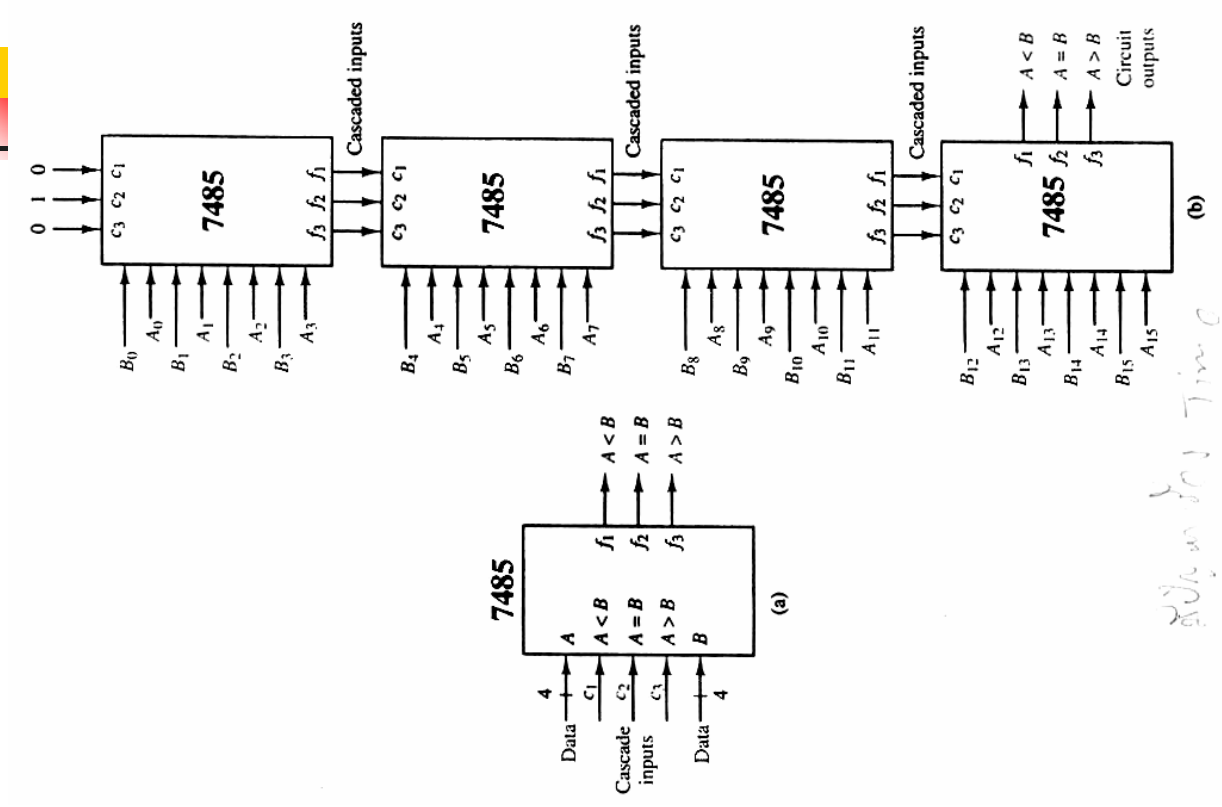
(c)

$$f_1 = \bar{A}_1 B_1 + \bar{A}_1 \bar{A}_0 B_0 + \bar{A}_0 B_1 B_0$$
$$f_2 = \bar{A}_1 \bar{A}_0 \bar{B}_1 \bar{B}_0 + \bar{A}_1 A_0 \bar{B}_1 B_0 + A_1 \bar{A}_0 B_1 \bar{B}_0 + A_1 A_0 B_1 B_0$$
$$f_3 = A_1 \bar{B}_1 + A_0 \bar{B}_1 \bar{B}_0 + A_1 A_0 \bar{B}_0$$

Use the 7485 to construct a 16-bit magnitude comparator.

Comparing inputs				Cascading inputs			Outputs		
A3, B3	A2, B2	A1, B1	A0, B0	A > B	A < B	A = B	A > B	A < B	A = B
A3 > B3	×	×	×	×	×	×	H	L	L
A3 < B3	×	×	×	×	×	×	L	H	L
A3 = B3	A2 > B2	×	×	×	×	×	H	L	L
A3 = B3	A2 < B2	×	×	×	×	×	L	H	L
A3 = B3	A2 = B2	A1 > B1	×	×	×	×	H	L	L
A3 = B3	A2 = B2	A1 < B1	×	×	×	×	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 > B0	×	×	×	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 < B0	×	×	×	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	L	L	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	H	L	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	H	L	L	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	×	×	H	L	L	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	H	L	L	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	L	H	H	L

(b)



A Computer Arithmetic Unit

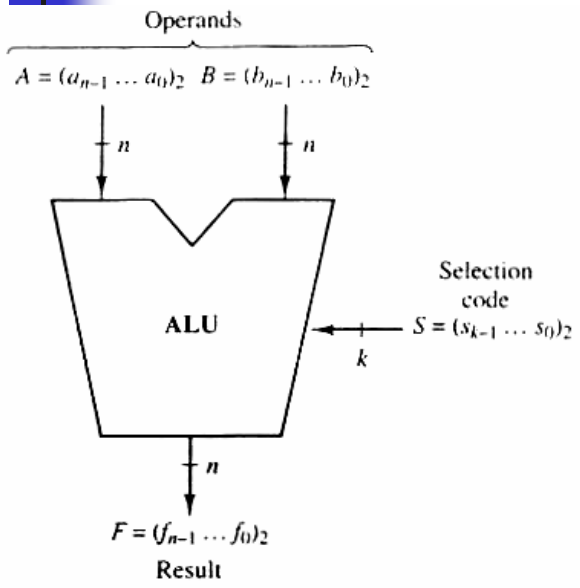
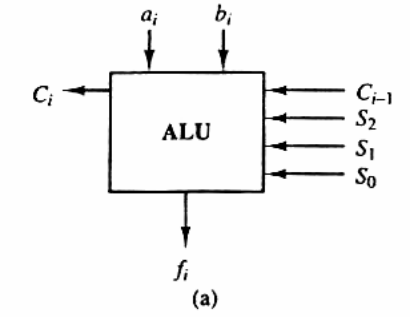
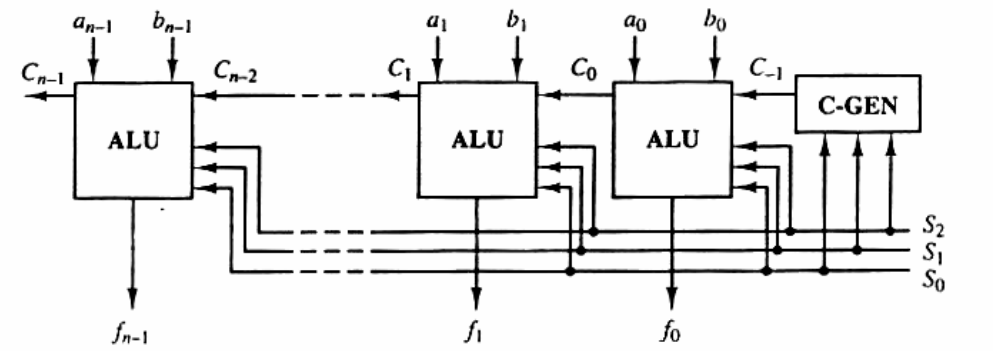


Figure 4.47 ALU logic symbol.

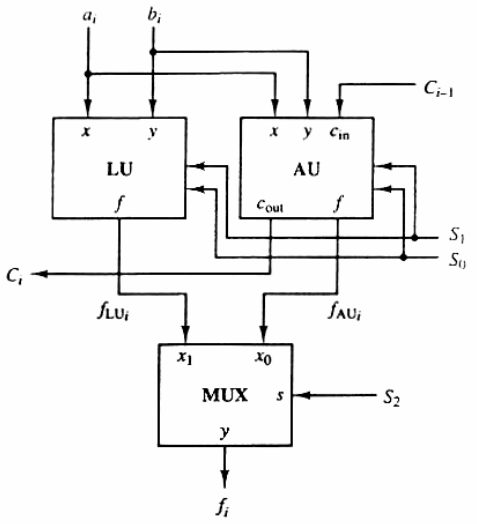
Selection Code			ALU Function
S_2	S_1	S_0	
0	0	0	$F = A + B$
0	0	1	$F = A - B$
0	1	0	$F = A + 1$
0	1	1	$F = A - 1$
1	0	0	$F = A \cap B$
1	0	1	$F = A \cup B$
1	1	0	$F = \bar{A}$
1	!	1	$F = A \oplus B$

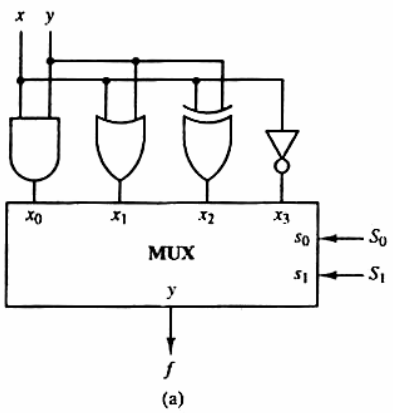
A Computer Arithmetic Unit





A Computer Arithmetic Unit



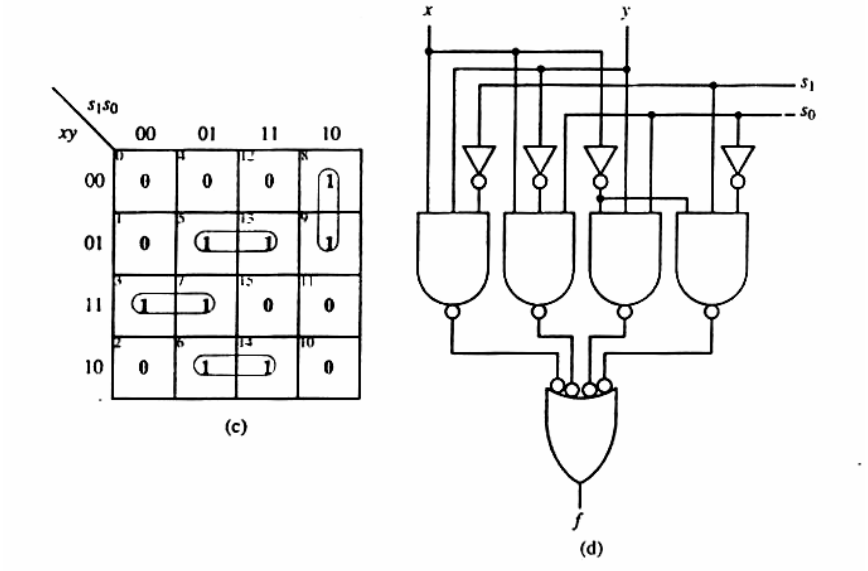


S_1	S_0	x	y	F_{LU}	
0	0	0	0	0	
0	0	0	1	0	x AND y
0	0	1	0	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	0	1	1	x OR y
0	1	1	0	1	
0	1	1	1	1	
1	0	0	0	1	
1	0	0	1	1	NOT x
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	0	
1	1	0	1	1	x XOR y
1	1	1	0	1	
1	1	1	1	0	

(b)

Figure 4.49 One-bit ALU partitioned into separate arithmetic and logic units.

A Computer Arithmetic Unit



A Computer Arithmetic Unit

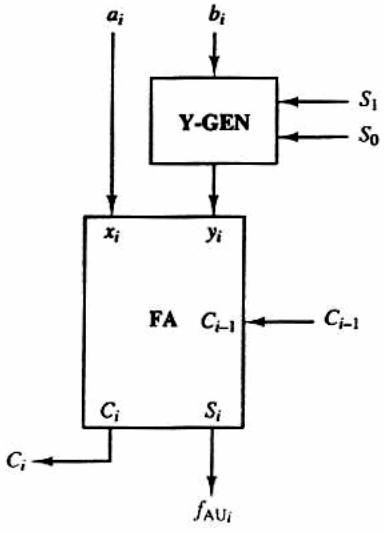


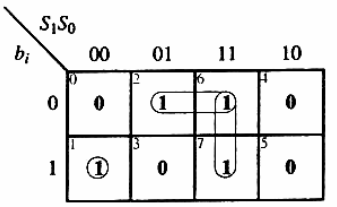
TABLE 4.6 VALUES OF y_i AND C_{-1} FOR THE ARITHMETIC FUNCTIONS

Function	S_1	S_0	y_i	C_{-1}
Add	0	0	b_i	0
Subtract	0	1	$\bar{b}_i$	1
Increment	1	0	0	1
Decrement	1	1	1	0

A Computer Arithmetic Unit

S_1	S_0	b_i	y_i	
0	0	0	0	Add
0	0	1	1	
0	1	0	1	Subtract
0	1	1	0	
1	0	0	0	Increment
1	0	1	0	
1	1	0	1	Decrement
1	1	1	1	

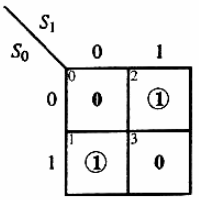
(a)



(b)

S_1	S_0	C_{-1}	
0	0	0	Add
0	1	1	Subtract
1	0	1	Increment
1	1	0	Decrement

(a)



(b)