Cairo University
Faculty of Graduate Studies for Statistical Research

Computer Science Department

Cairo University

CS504

Digital Logic & Computer Organization

Lecture 8

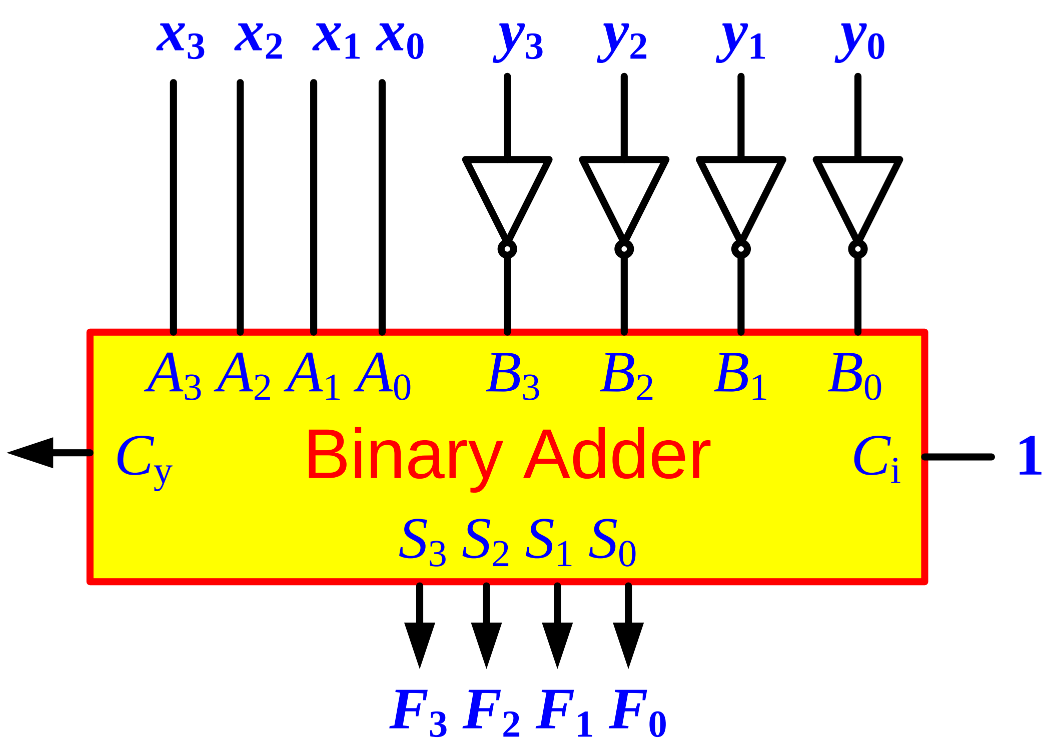
Lecture Outline (Chapter 4)

- ★ **Binary Adder (Section 4.5)**
  - Binary Subtractor
  - Binary Adder / Subtractor
  - Overflow On Signed And Unsigned
  - Overflow Detection
  - Binary Adder/Subtractor With Overflow Detection
- ★ **Decimal Adder (Section 4.6)**
  - BCD Addition
  - BCD Adder
- ★ **Binary Multiplier (Section 4.7)**
- ★ **Binary Magnitude Comparator (Section 4.8)**

Binary Subtractor

★ **Use 2's complement with binary adder**

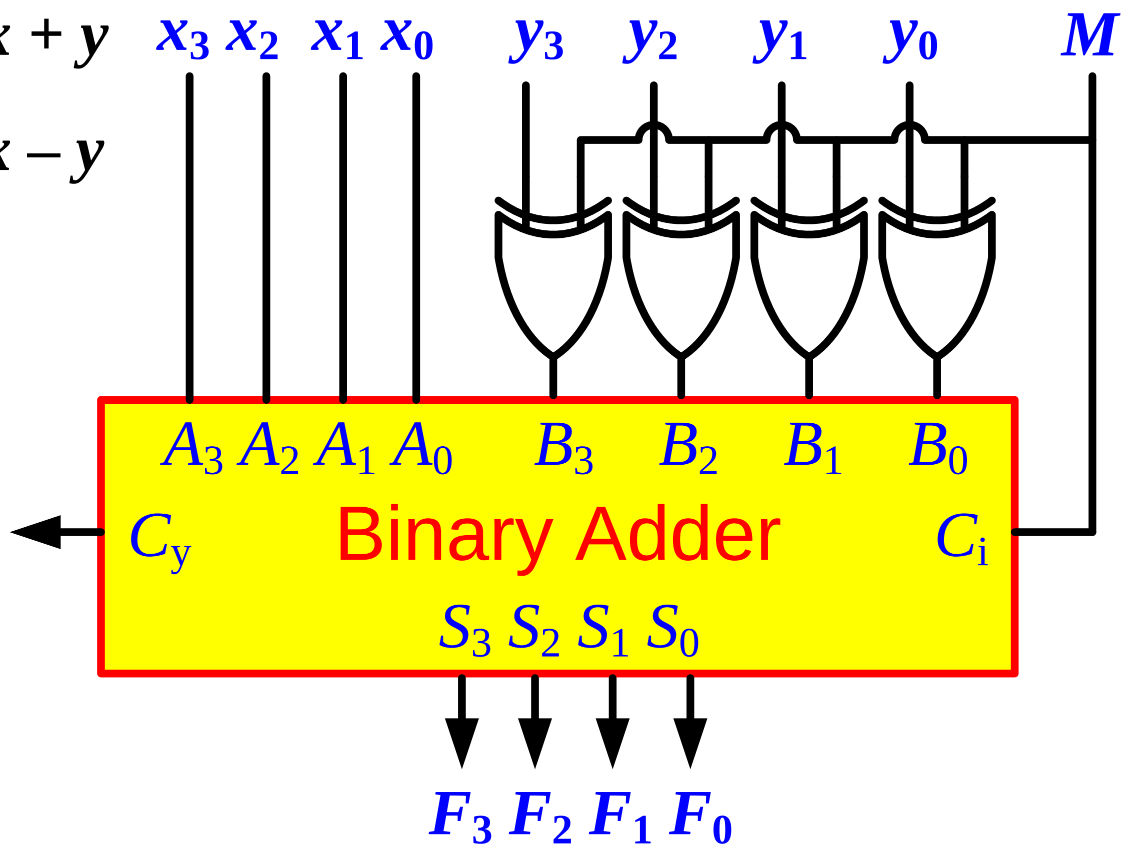
- $x - y = x + (-y) = x + y' + 1$



Binary Adder / Subtractor

★ ***M*: Control Signal (Mode)**

- $M = 0 \Rightarrow F = x + y$
- $M = 1 \Rightarrow F = x - y$



Overflow On Signed And Unsigned

- ★ Overflow is a problem in digital computers because the number of bits (n) that hold the number is finite and a result that contains n+1 bits cannot be accommodated.

- ★ When two unsigned numbers are added, an overflow is detected from the end carry out of the most significant bit (MSB) position.

- ★ When two signed numbers are added, the sign bit is treated as part of the number and the end carry does not indicate an overflow.

- ★ An overflow can't occur after an addition if one number is positive and the other is negative but it occurs if the two numbers added are both positive or both negative.

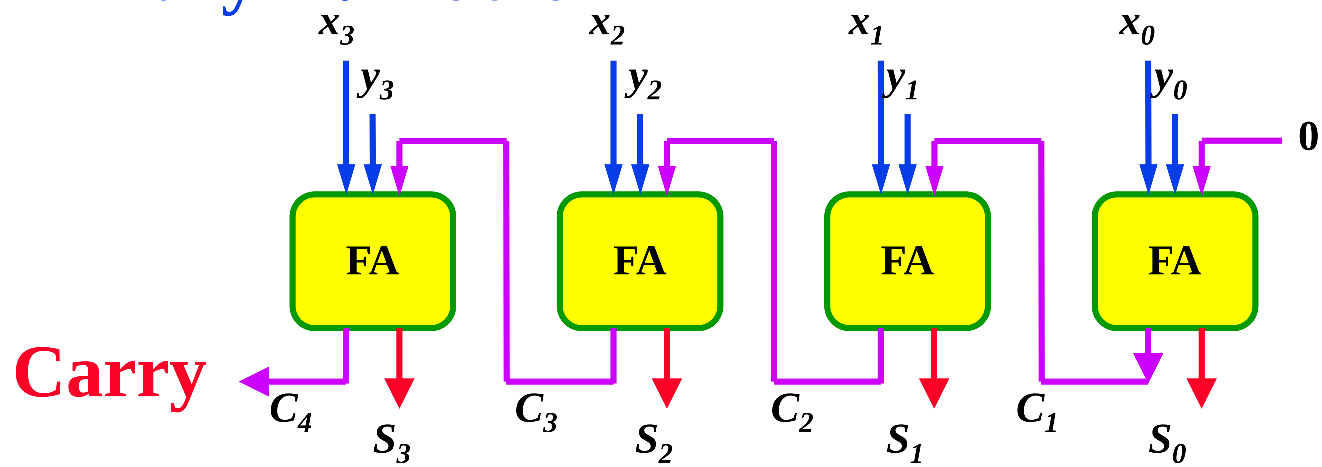
Overflow On Signed

- An overflow can be detected by observing the carry into the sign bit position and the carry out of the sign bit position.
- If only these two carries are not equal, an overflow has occurred.
- Two signed binary numbers, +70 and +80, are stored in two 8-bit registers.
- The sum of the two numbers is +150, exceeds the capacity of 8-bit register. This is also true for -70 and -80

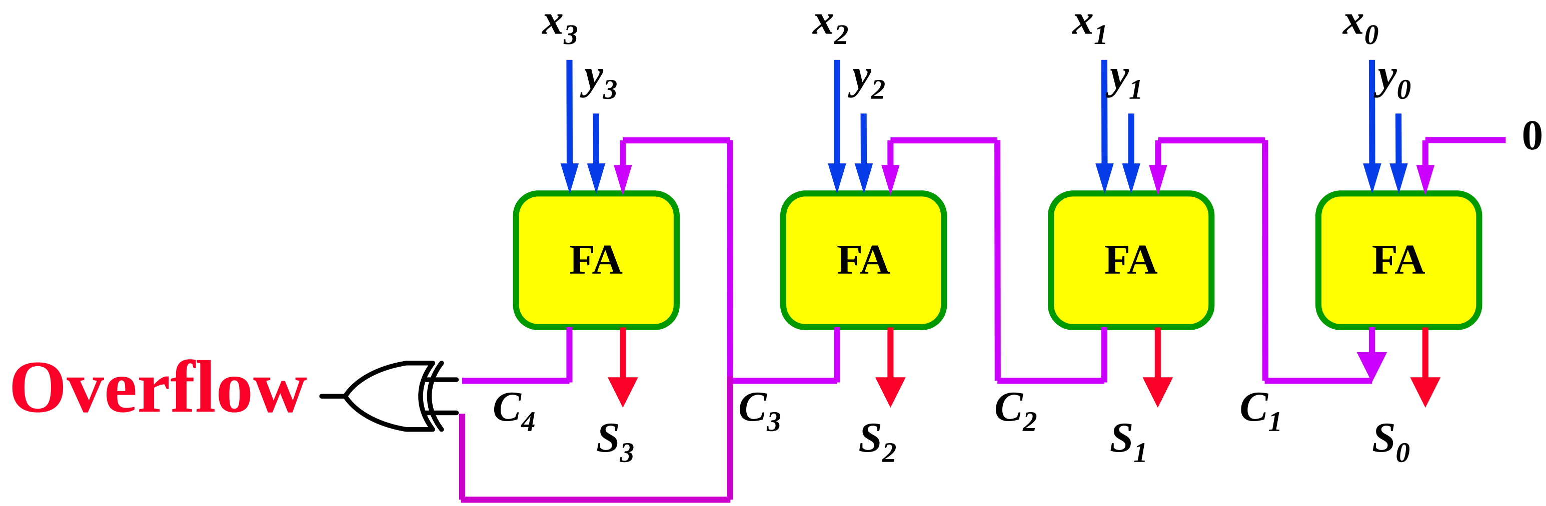
carries:	0 1	carries:	1 0
+70	0 1000110	−70	1 0111010
+80	0 1010000	−80	1 0110000
+150	1 0010110	−150	0 1101010

Overflow Detection

- ★ **Unsigned Binary Numbers**



- ★ **2's Complement Numbers**



Binary Adder/Subtractor With Overflow Detection

M = 1 → Subtractor **M = 0 → Adder**

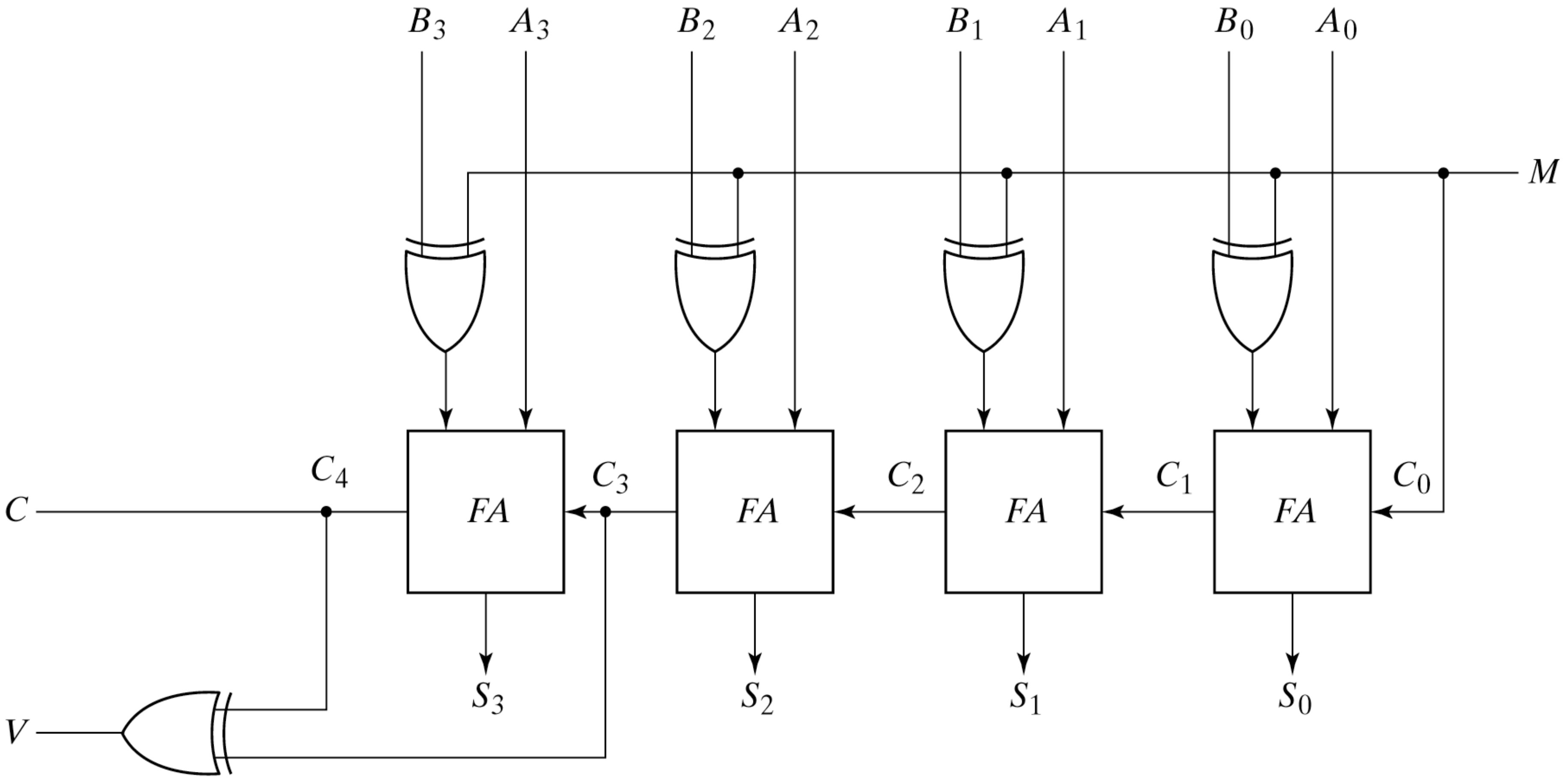


Fig. 4-13 4-Bit Adder Subtractor

Decimal Adder

- Computers or calculators that perform arithmetic operations directly in the decimal number system represent decimal numbers in binary coded form
- Add two BCD's
  - 9 inputs: two 4-bit BCD's and one carry-in
  - 5 outputs: one 4-bit BCD and one carry-out
- Design approaches
  - A truth table with 2^9 = 512 entries
  - The sum <= (9 + 9 + 1) = 19 where 1 being an input carry
  - Use 4-bit binary Adder
    - Convert the binary sum to BCD sum

BCD Addition

Example : Evaluate the following operations in BCD System

- 3 + 4
- 4 + 8
- 148 + 576

$$
\begin{array}{r}
3 \\
+\ 4 \\
\hline
7
\end{array}
\qquad
\begin{array}{c}
\xrightarrow{BCD} \\
\xrightarrow{BCD} \\
\xleftarrow{Decimal}
\end{array}
\qquad
\begin{array}{r}
0011 \\
+\ 0100 \\
\hline
0111
\end{array}
$$

BCD Addition (2)

Example : Evaluate the following operations in BCD System

- 3 + 4
- 4 + 8
- 148 + 576

$$4 \xrightarrow{BCD} 0100$$

$$+\,8 \xrightarrow{BCD} +\,1000$$

$$1100 \longrightarrow \text{Error}$$

$$+\,0110 \longleftarrow$$ We must add 6 (0110) to the result

$$12 \xleftarrow{Decimal} 00010010$$

BCD Addition (3)

Example : Evaluate the following operations in BCD System

- 3 + 4
- 4 + 8
- 184 + 576

$$\begin{array}{rccrrr}
184 & \xrightarrow{BCD} & & 0001 & 1000 & 0100 \\
+\,576 & \xrightarrow{BCD} & + & 0101 & 0111 & 0110 \\
\hline
 & & & 0111 & 10000 & 1010 \\
 & & + & & 0110 & 0110 \\
\hline
760 & \xleftarrow{Decimal} & & 0111 & \cancel{1}0110 & \cancel{1}0000
\end{array}$$

BCD Addition (4)

Notes

1 - In BCD Addition , we add (0110)=(6) if the result value was greater than (1001)=(9) or if the result was more than 4 digits

In previous Example we added 0110 when the result was

A - greater than 9 (1001)

B - more than 4 digits (10000)

Result more than 4 digit is greater than 9 (1001) ☺

BCD Adder

- ★ 4-bits + 4-bits + C_{in}
- ★ Operands and Result: 0 to 9

$$\begin{array}{rrrrr} & & & & C_{in} \\ + & x_3 & x_2 & x_1 & x_0 \\ + & y_3 & y_2 & y_1 & y_0 \\ \hline C_{out} & S_3 & S_2 & S_1 & S_0 \end{array}$$

Table 4.5
Derivation of BCD Adder

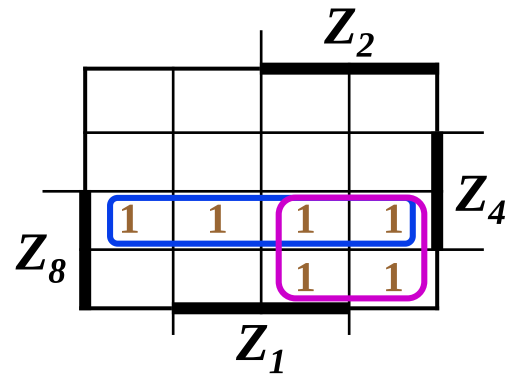
Binary Sum					BCD Sum					Decimal
K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

BCD Adder (2)

★ Correcting Binary Adder's Output by (+6)

- If the result is between 'A' and 'F'
- If K = 1

$Z_8\ Z_4\ Z_2\ Z_1$	*Err*
0 0 0 0	0
1 0 0 0	0
1 0 0 1	0
1 0 1 0	1
1 0 1 1	1
1 1 0 0	1
1 1 0 1	1
1 1 1 0	1
1 1 1 1	1



$$Err = K + Z_8 Z_4 + Z_8 Z_2$$

BCD Adder (3)

- ★ A decimal parallel adder that adds n decimal digits needs n BCD adder stages.
- ★ The output carry from one stage must be connected to the input carry of the next higher-order stage.

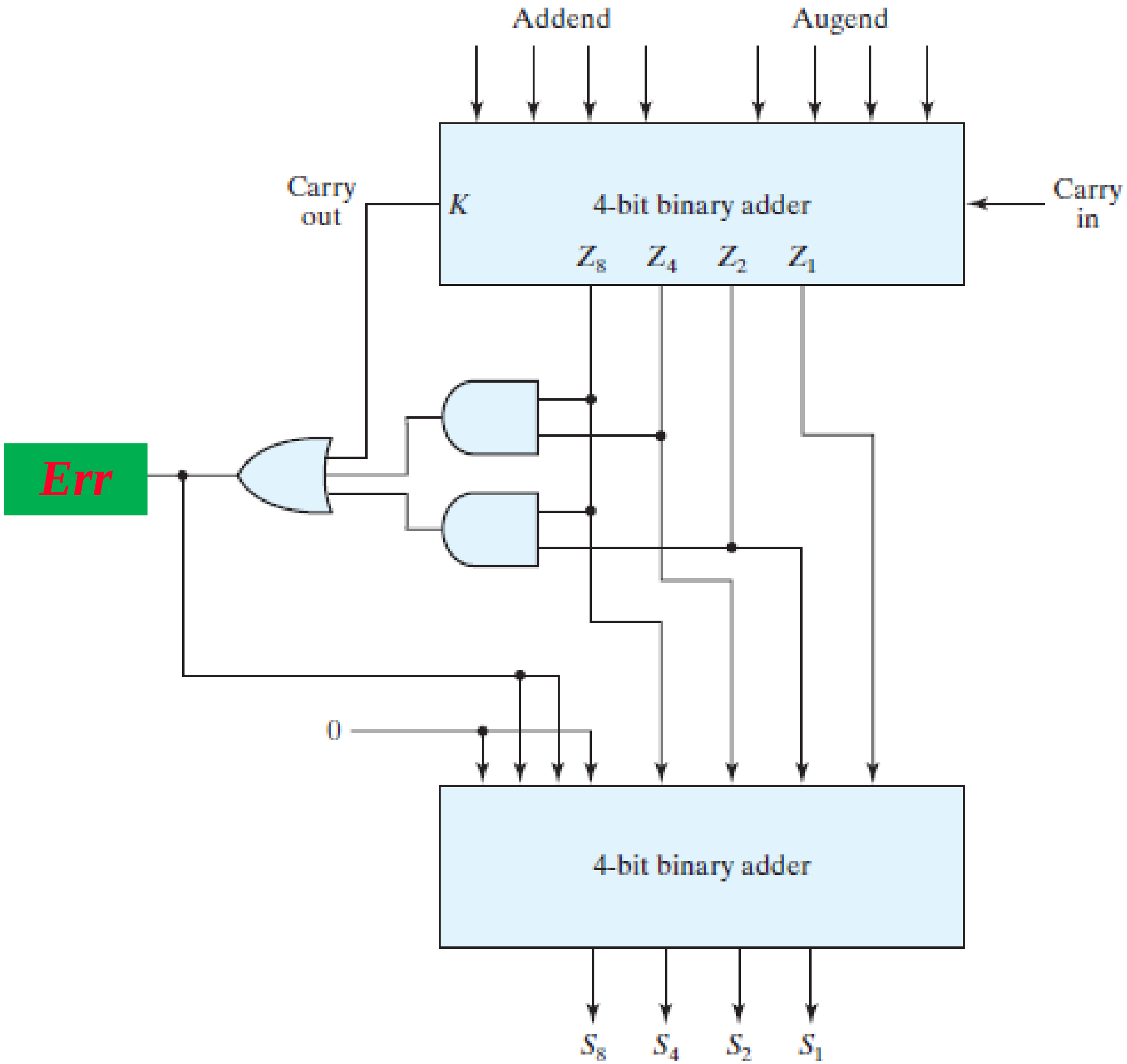


FIGURE 4.14
Block diagram of a BCD adder

Binary Multiplier

- Usually there are **more bits** in the partial products and it is necessary to use **full adders** to produce the sum of the partial products.

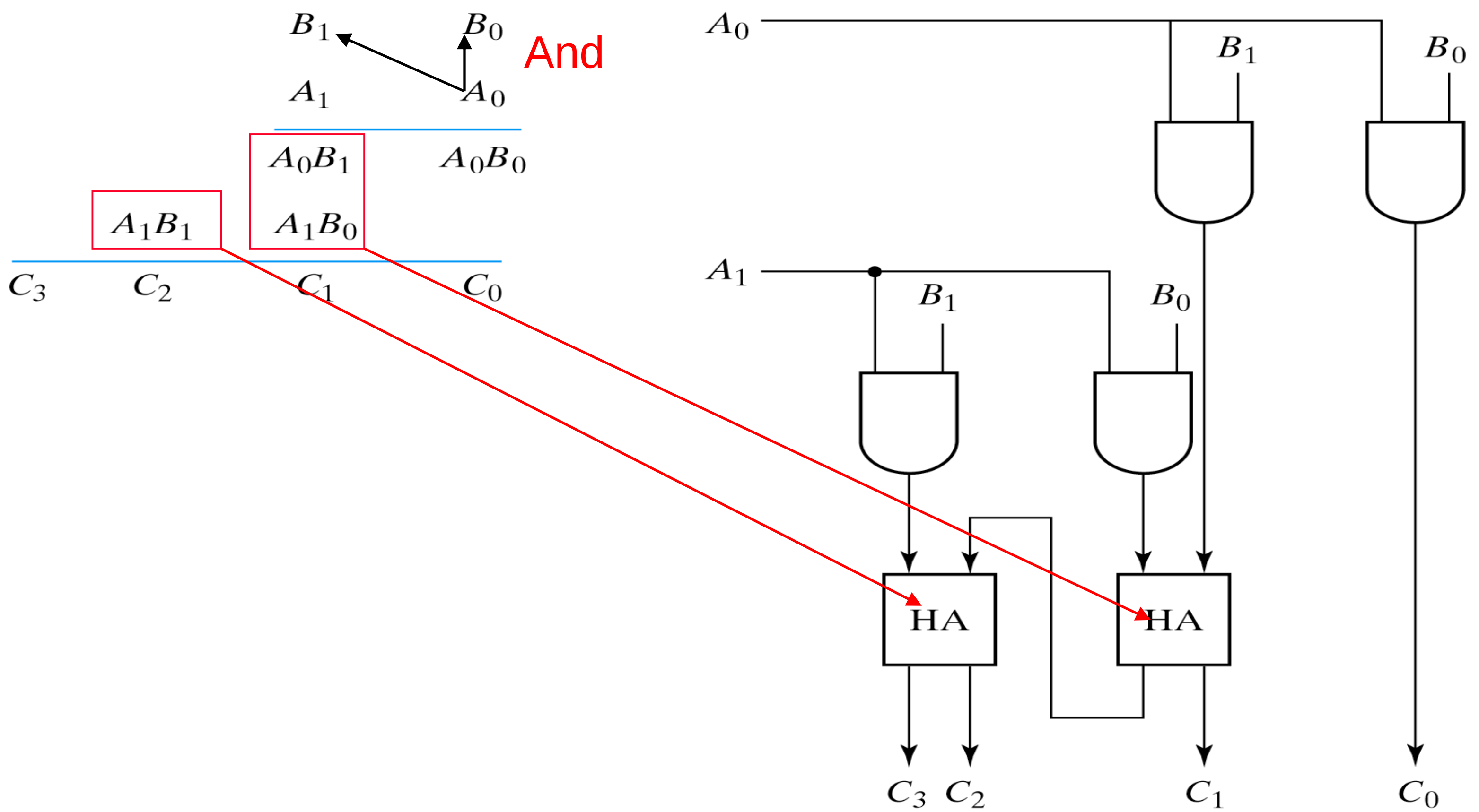


Fig. 4-15 2-Bit by 2-Bit Binary Multiplier

4-bit by 3-bit Binary Multiplier

★ **For J multiplier bits and K multiplicand bits we need (J X K) AND gates and (J − 1) K-bit adders to produce a product of J+K bits.**

★ **K=4 and J=3, we need 12 AND gates and two 4-bit adders.**

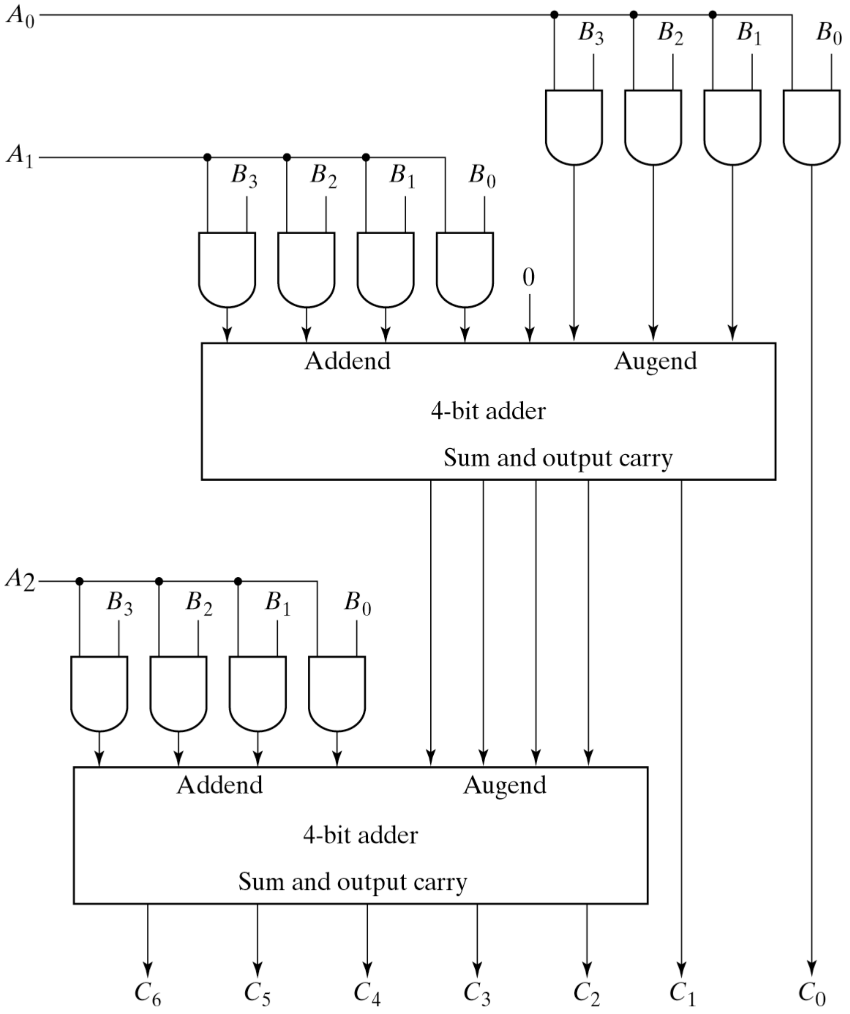


Fig. 4-16 4-Bit by 3-Bit Binary Multiplier

Binary Magnitude Comparator

- ★ **We inspect the relative magnitudes of pairs of the most significant bit (MSB).**

- ★ **If equal, we compare the next lower significant pair of digits until a pair of unequal digits is reached.**

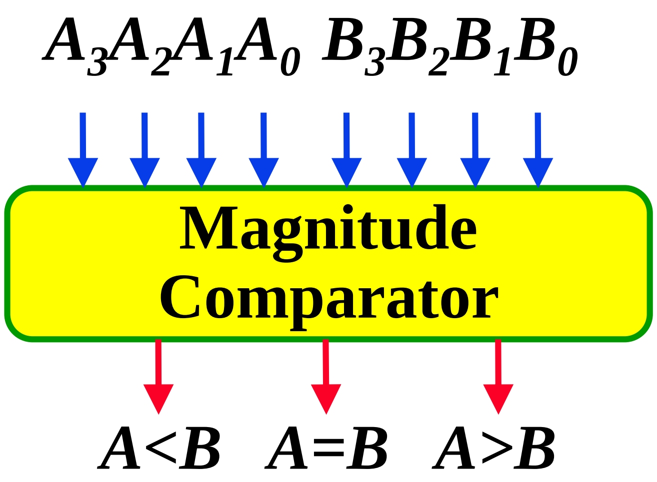
- ★ **If the corresponding digit of A is 1 and that of B is 0, we conclude that A>B.**

- ★ **If the corresponding digit of A is 0 and that of B is 1, we conclude that A<B.**

Binary Magnitude Comparator (2)

★ Compare 4-bit number to 4-bit number

- **3 Outputs: < , = , >**
- **Expandable to more number of bits**



$$x_3 = \overline{A}_3\,\overline{B}_3 + A_3\,B_3$$

$$x_2 = \overline{A}_2\,\overline{B}_2 + A_2\,B_2$$

$$x_1 = \overline{A}_1\,\overline{B}_1 + A_1\,B_1$$

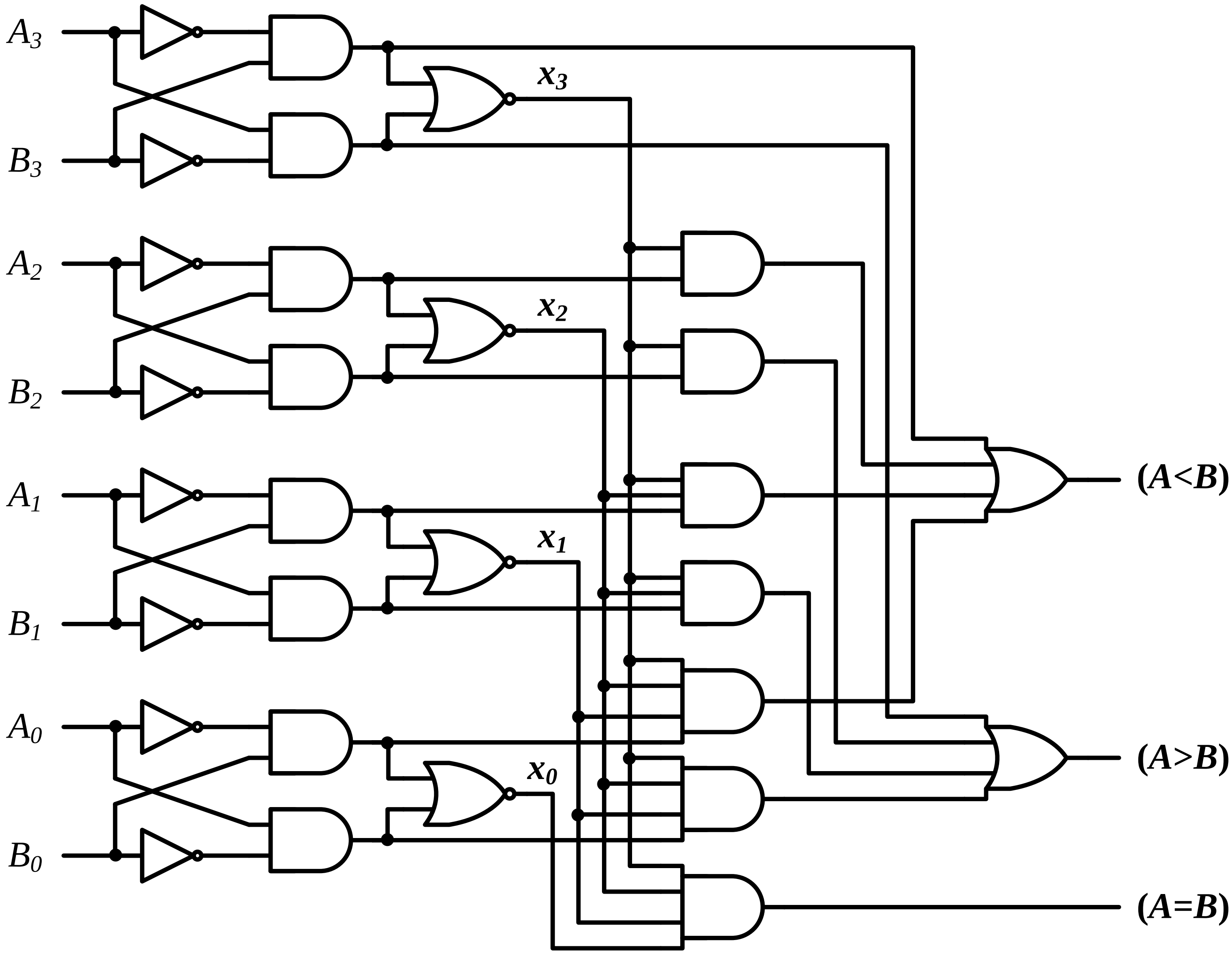
$$x_0 = \overline{A}_0\,\overline{B}_0 + A_0\,B_0$$

$$(A = B) = x_3\,x_2\,x_1\,x_0$$

$$(A > B) = A_3\,\overline{B}_3 + x_3\,A_2\,\overline{B}_2 + x_3\,x_2\,A_1\,\overline{B}_1 + x_3\,x_2\,x_1\,A_0\,\overline{B}_0$$

$$(A < B) = \overline{A}_3\,B_3 + x_3\,\overline{A}_2\,B_2 + x_3\,x_2\,\overline{A}_1\,B_1 + x_3\,x_2\,x_1\,\overline{A}_0\,B_0$$

Binary Magnitude Comparator (3)



The End

Questions?