Chapter4: Combinational Logic

Lecture4- Design Decimal Adder and Binary Multiplier Circuits

Engr. Arshad Nazir, Asst Prof
Dept of Electrical Engineering
SEECS

Objectives

- Design Decimal Adder and Binary Multiplier Circuits

Decimal Adder

- Computers or calculators that perform arithmetic operations directly in the decimal number system represent decimal number in binary coded form (BCD).
- An adder for such a computer must employ arithmetic circuits that accept binary coded decimal numbers and present results in the same code
- To add two BCD digits, we require
  - 9 inputs: eight inputs for two BCDs and one carry-in. Since four bits are required to code each decimal digit
  - 5 outputs: four outputs for one BCD and one carry-out

BCD Adder

- To design a BCD adder we require a truth table with 2^9 entries (since 9 inputs). This may become too difficult to work with so we device some easy alternative to design a BCD adder. This is shown in table 4-5
- Each digit doesn't exceed 9 and so the output sum can't exceed, 9+9+1 = 19 (two BCD digits and input carry)
- Suppose we apply two BCD digits to a 4-bit binary adder
- The adder will form the sum in binary and produces a result that ranges from 0 through 19
- These binary numbers are listed in table 4-5 and are labeled by symbols K, Z_8, Z_4, Z_2 and Z_1. (K is the output carry). But the output sum must be represented in BCD (not binary) and should appear in the form listed in the columns C, S_8, S_4, S_2 and S_1 under BCD sum
- The problem is to find a rule by which binary sum is converted to corresponding BCD sum

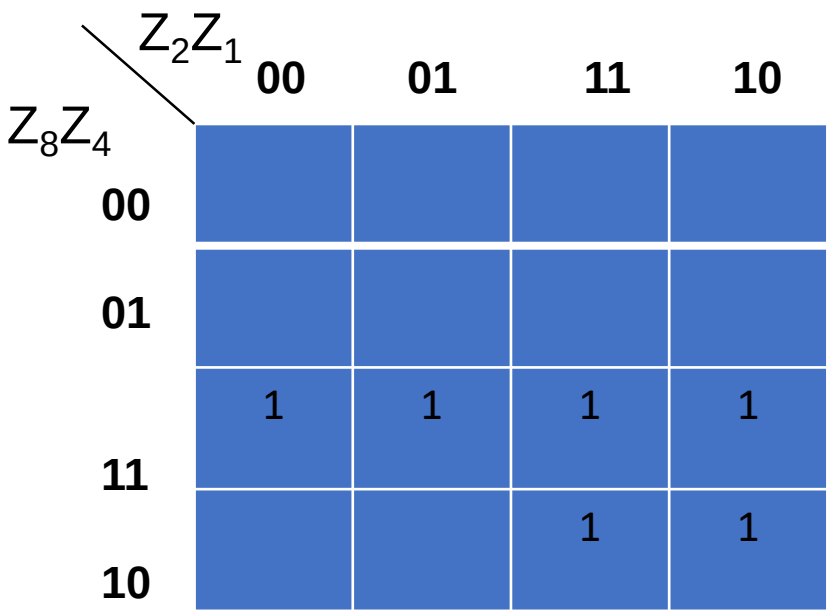
Derivation of a BCD Adder

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

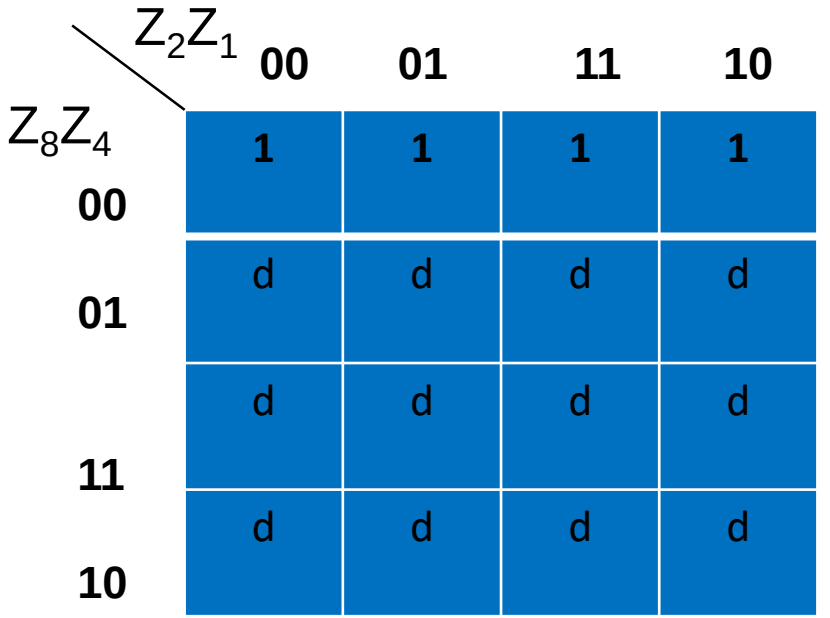
Correction for BCD Adder

- From the table it is apparent that when binary sum is equal to or less than 1001 (decimal 9), the corresponding BCD number is identical and therefore no correction is needed.
- Modifications are needed if the sum is greater than 1001 as we get non-valid BCD representation. The addition of binary 6 (0110) to binary sum converts it to the correct BCD representation and also produces an output carry as required (refer to section 1-7)
- The logic circuit that detects the necessary correction can be derived from the table entries. Correction (adding decimal 6 or binary 0110) is needed when the binary sum has an output
  - $K = 1$
  - $Z_8Z_4 = 1$
  - $Z_8Z_2 = 1$

Map Simplification of the Function C



k=0



k=1

$$\mathbf{C(K,}Z_{8,}Z_{4,}Z_{2,}Z_{1}\mathbf{)= K+}\ Z_8Z_4 + Z_8Z_2$$

Correction for BCD Adder

- This condition for correction and an output carry can be expressed as
  - $C = K + Z_8Z_4 + Z_8Z_2$
- In figure 4-14, the output carry generated from the bottom adder can be ignored since this is already available at the output carry terminal
- A decimal parallel adder that adds n decimal digits (in BCD form) needs n BCD adder stages with output carry from one stage connected to the input carry of next higher-order stage

BCD Adder

Numbers that need correction (add 6) are:

K	Z_8	Z_4	Z_2	Z_1	
0	1	0	1	0	(10)
0	1	0	1	1	(11)
0	1	1	0	0	(12)
0	1	1	0	1	(13)
0	1	1	1	0	(14)
0	1	1	1	1	(15)
1	0	0	0	0	(16)
1	0	0	0	1	(17)
1	0	0	1	0	(18)
1	0	0	1	1	(19)

$$C = K + Z_8Z_4 + Z_8Z_2$$

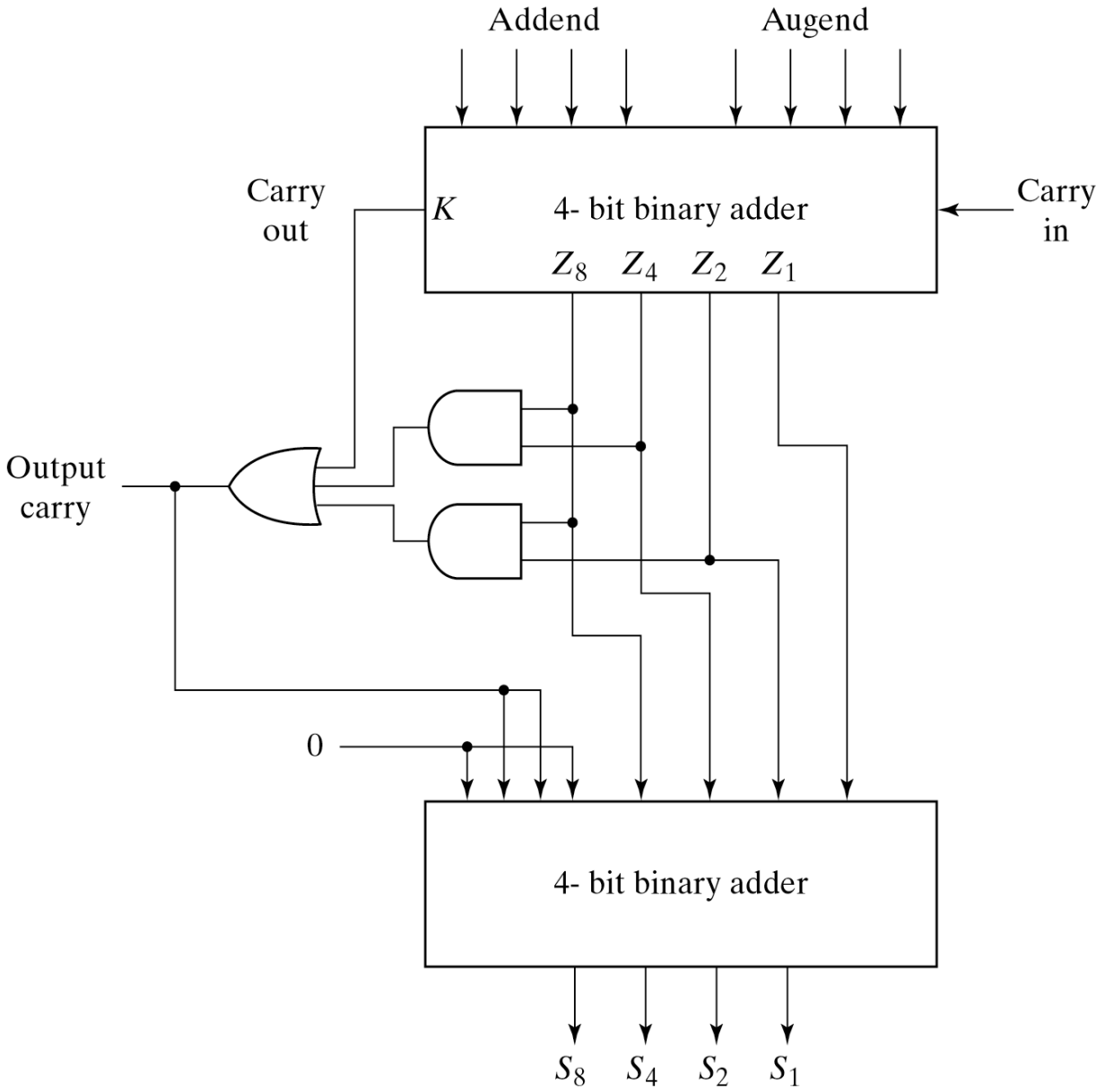


Fig. 4-14 Block Diagram of a BCD Adder

Binary Multiplier

- Multiplication of the binary number is performed in the same way as in decimal numbers. The multiplicand is multiplied by each bit of the multiplier starting from the least significant bit
- Each such such multiplication forms a partial product. Successive partial products are shifted one position to the left. The final product is obtained from the sum of the partial products
- Consider the binary multiplication of 2-bit numbers where B_1 and B_2 are multiplicand bits and A_1 and A_0 are multiplier bits. $C_3\,C_2\,C_1\,C_0$ are the product bits

Binary Multiplier

- The first partial product is formed by multiplying A_0 by B_1 B_0. The multiplication of any two bits produces a 1 if both bits are 1, otherwise it produces a 0. This is identical to AND operation. Therefore partial product can be implemented with an AND gates as shown in figure 4-15
- The second partial product is formed by multiplying A_1 by B_1 B_0 and shifted one position to the left
- The two partial products are added with two half adder (HA) circuits. If there are more bits in the partial products then we use full adders to produce the sum of the partial products
- The least significant bit of the partial product doesn't have to go through an adder since it is formed by the output of the first AND gate

$$
\begin{array}{cccc}
 & & B_1 & B_0 \\
 & & A_1 & A_0 \\
\hline
 & & A_0B_1 & A_0B_0 \\
 & A_1B_1 & A_1B_0 & \\
\hline
C_3 & C_2 & C_1 & C_0
\end{array}
$$

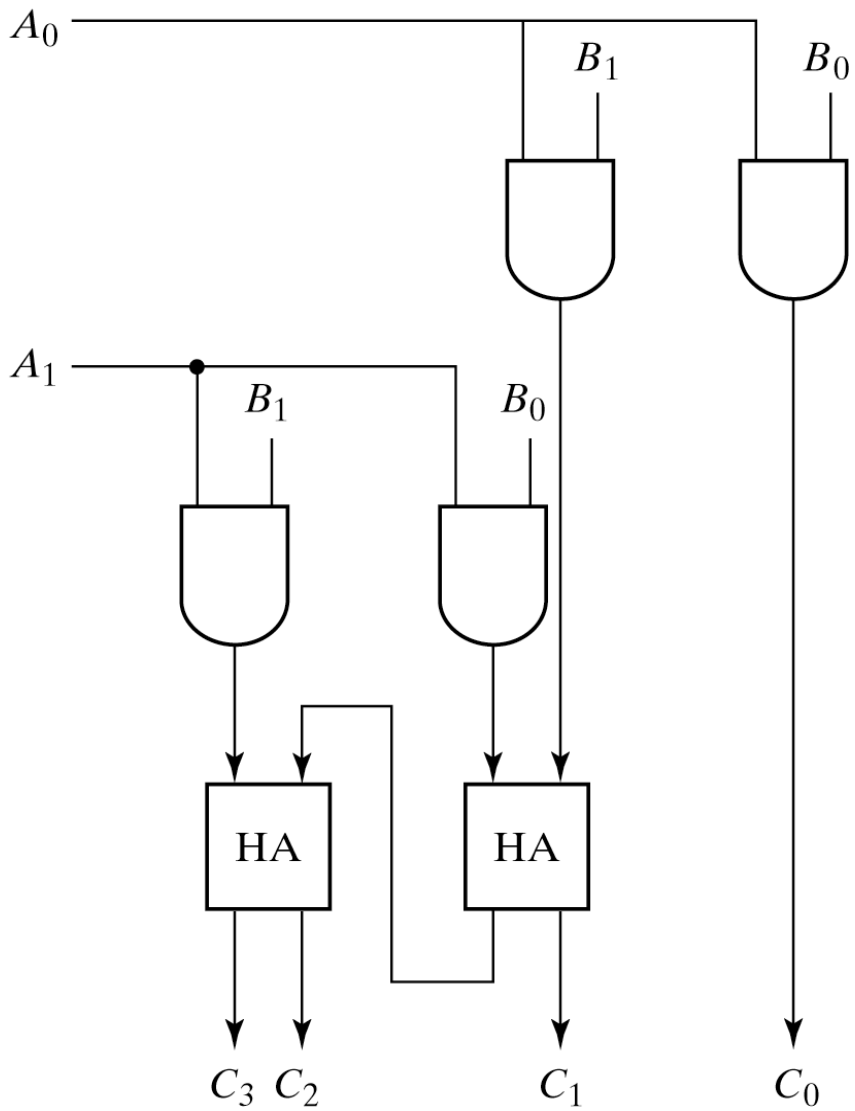


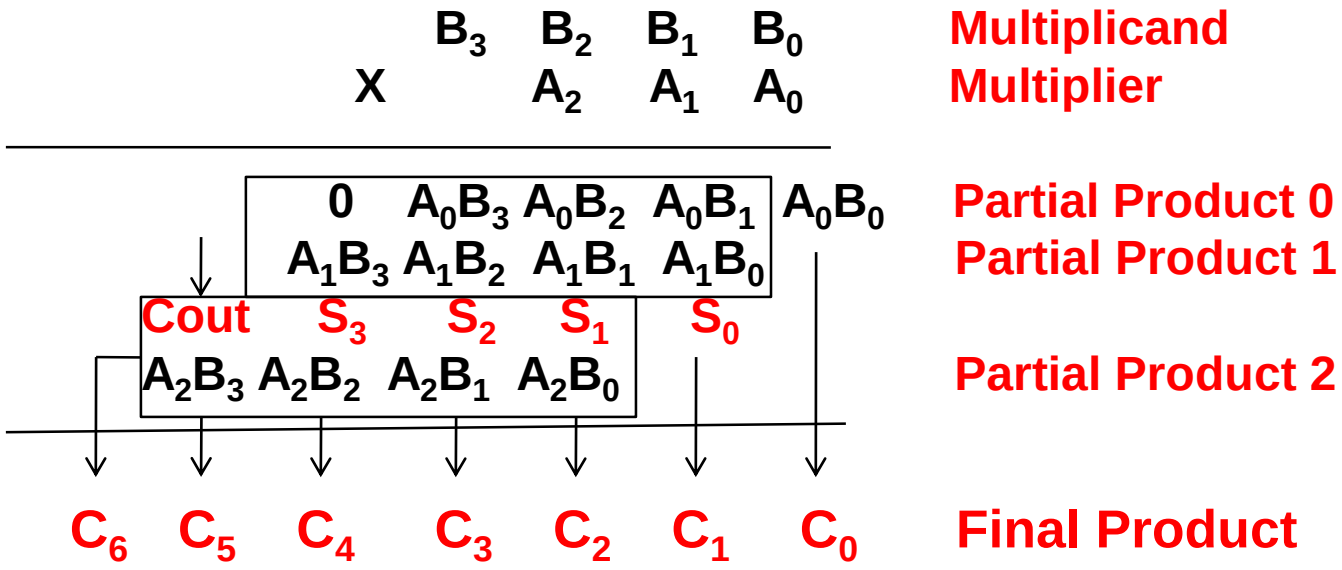
Fig. 4-15 2-Bit by 2-Bit Binary Multiplier

K-bit by J-bit Binary Multiplier

- A combinational circuit for binary multiplier with more bits can be constructed in a similar way. A bit of the multiplier is ANDed with each bit of the multiplicand in as many levels as there are bits in the multiplier
- The binary output in each level of AND gate is added with the partial product of the previous level to form a new partial product. The last level produces the product
- 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

4-bit by 3-bit binary multiplier

- Consider a multiplier circuit that multiplies a binary number of four bits by a number of three bits.
- Let the multiplicand is represented by B_3 B_2 B_1 B_0 and the multiplier by A_2 A_1 A_0
- Since K = 4 and J = 3, we need 12 AND gates and two 4 – bit adders to produce a product of seven bits



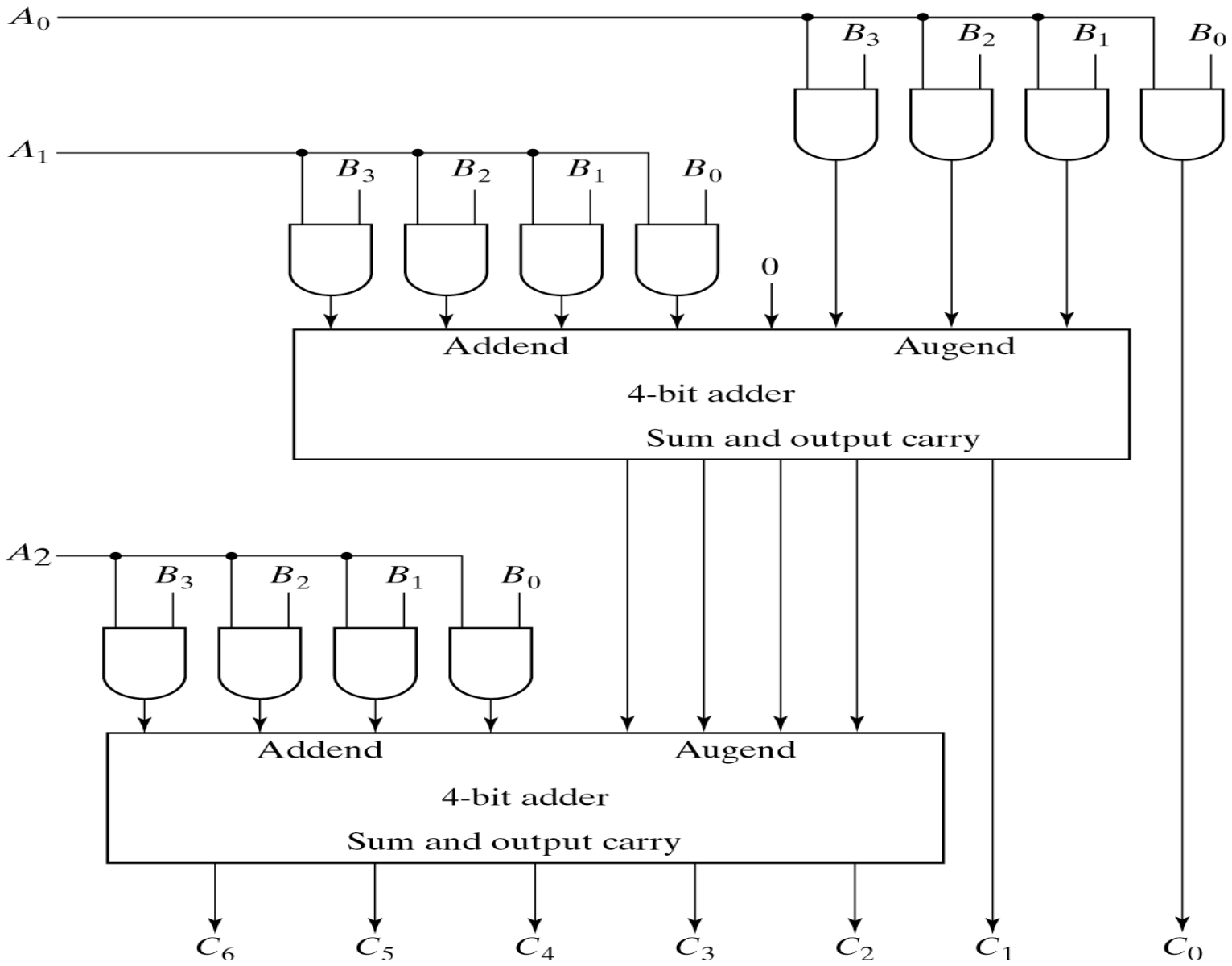


Fig. 4-16 4-Bit by 3-Bit Binary Multiplier

The End