Combinational Logic

1. Introduction.
2. Analysis and Design Procedure for Combinational Logic Circuits.
3. Types of Combinational Circuit.

Introduction

1. Combinational Logic Circuits are circuits designed by using different types of logic gates.
2. A logic gate is a fundamental building block of any electronic circuit.
3. In other word **Combinational Logic Circuits are memoryless digital logic circuits** whose output at any instant in time depends only on the combination of its inputs.
4. Means these circuits do not make use of any memory or storage device.

Introduction

The digital system consists of two types of circuits as –

a) **Combinational circuits**
b) **Sequential circuits**

Introduction

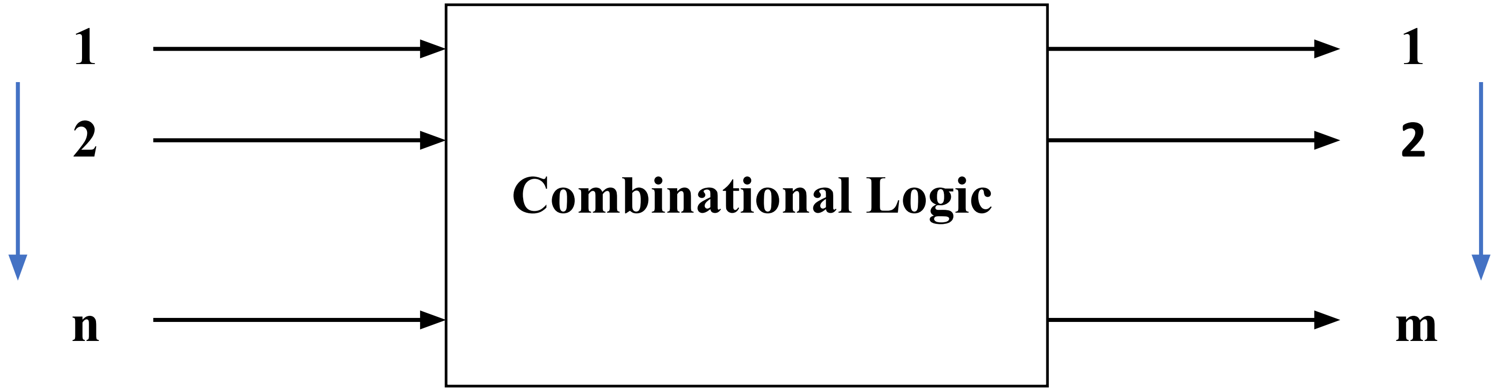
Combinational circuit consists of logic gates whose output at any time is determined from the present combination of inputs. The logic gate is the most basic building block of combinational logic circuit.

A combinational circuit consists of:

- **Input variables**
- **Logic gates**
- **Output variables**

Introduction

The function implemented by combinational circuit is depend upon the Boolean expressions. The combinational circuit having **‘n’** inputs and **‘m’** outputs. The **‘n’** number of inputs shows that there are **2^n** possible combinations of bits at the input.



Introduction

At all combinational circuit can be designed by the following steps of design process- 3

1. Stated problem.
2. Ascertain the input and output variables.
3. The input and output variables are allocated letter symbols.
4. Construction of a truth table to encounter input -output requirements.
5. Writing Boolean expressions for numerous output variables in relations of input variables.

Introduction

6. The basic Boolean expression is acquired by any method of minimization — algebraic, or tabulation method or Karnaugh map method.

7. A logic diagram is understood from the simplified Boolean expression using logic gates.

- The logic gates are combined in such a way that the output state depends completely on the input states. Combinational logic circuits have no memory, timing or feedback loops, there operation is prompt. A combinational logic circuit performs an operation assigned logically by a Boolean expression or truth table.

Introduction

- The logic gates are combined in such a way that the output state depends completely on the input states.
- Combinational logic circuits have no memory, timing or feedback loops, there operation is prompt. A combinational logic circuit performs an operation assigned logically by a Boolean expression or truth table.

Introduction

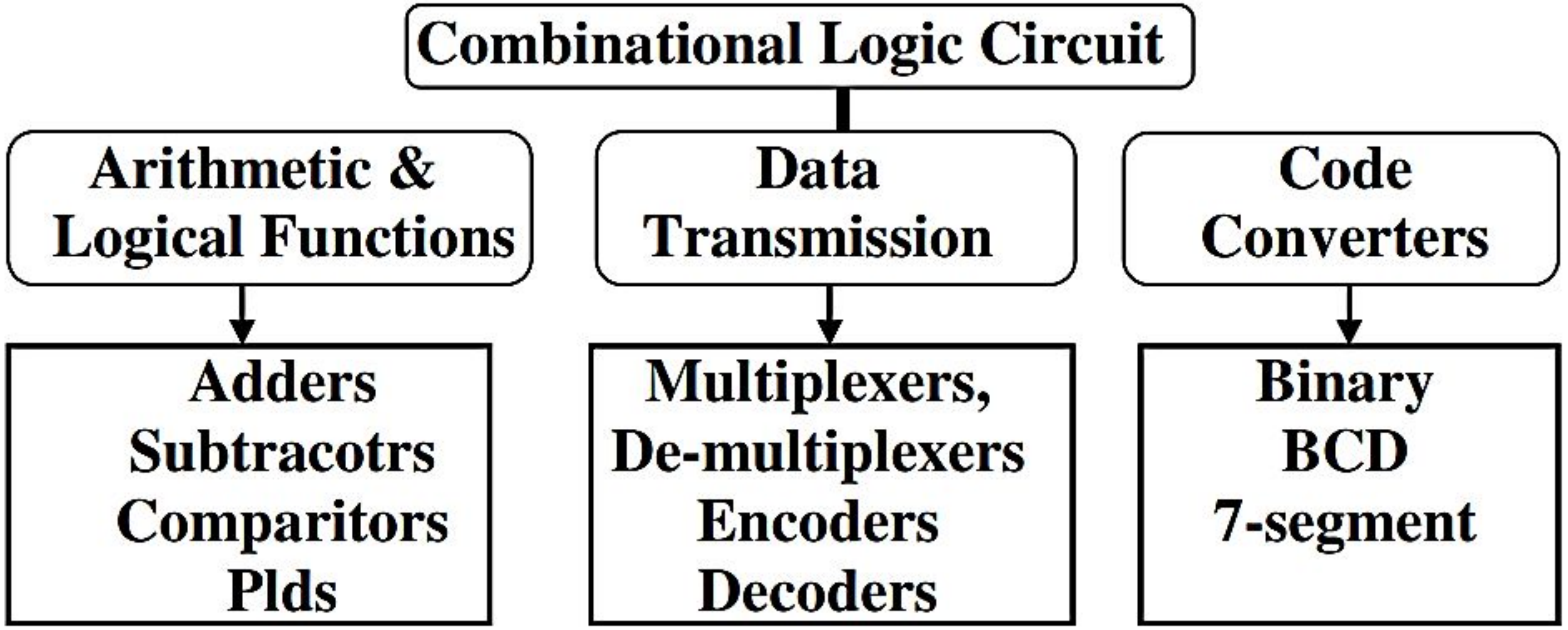
- The **three main methods** of specifying the function of a combinational logic circuit as –

1. **Boolean Algebra:** This forms the algebraic expression viewing the operation of the logic circuit for each input variable whichever True or False that results in a logic **‘‘1’’** output.
2. **Truth Table:** A truth table expresses the function of a logic gate by providing a brief list that **shows all the output** states in tabular form for each possible combination of input variable that the gate could meet.

Introduction

3. Logic Diagram: This is a graphical representation of a logic circuit that displays the wiring and connections of each individual logic gate, signified by a specific graphical symbol that implements the logic circuit.

Classification of Combinational Logic

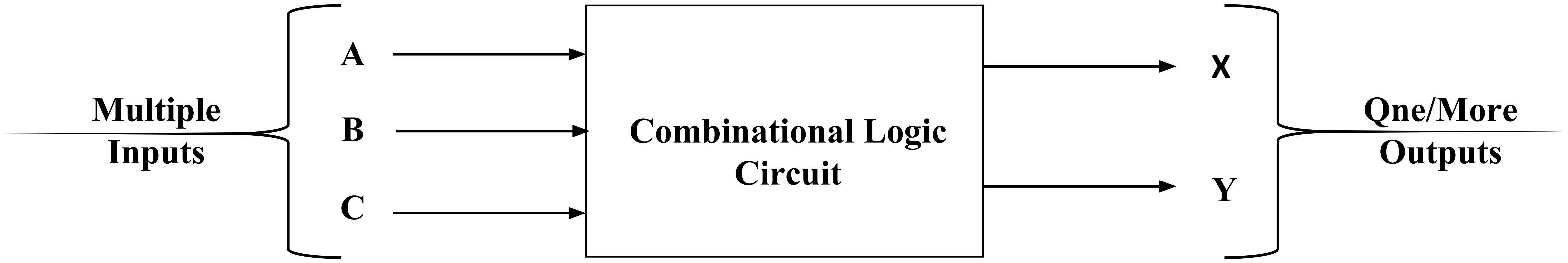


Classification of Combinational Logic

One of the greatest uses of combinational logic is in **Multiplexer and De-multiplexer** type circuits. Here, multiple inputs or outputs are connected to a common signal stripe and logic gates are used to decode an address to select a single data input or output switch.

Multiple Input Combinational Circuit

- Multi input gates can be made by constructing gates of the same type with fewer inputs. The figure below shown how a three input AND gate can be made out of two input AND gates. The same logical standard applies - the output goes **‘‘low’’ (0)** if any of the inputs are made **“high” (1).**



Multiple Input Combinational Circuit

- Combinational Logic Circuits are made up from basic logic **NAND, NOR or NOT gates** that are connected together to produce more complex switching circuits. These logic gates are the building blocks of combinational logic circuits.
- An example of a combinational circuit-
- Decoder, which converts the binary code data present at its input into a number of diverse output lines, one at a time producing an equivalent decimal code at its output.

Multiple Input Combinational Circuit

- The number of possible input states is equal to two to the power of the number of inputs: 6

Number of possible input states = 2^n

Where,

n = Number or inputs

- This increase in the number of possible input states obviously allows for more complex gate performance. Instead of simply inverting a single “high” or “low” logic level, the output of the gate will be determined by whatsoever combination of 1‘s and 0‘s is present at the input stations.

Arithmetic Circuits – Basic Building Blocks

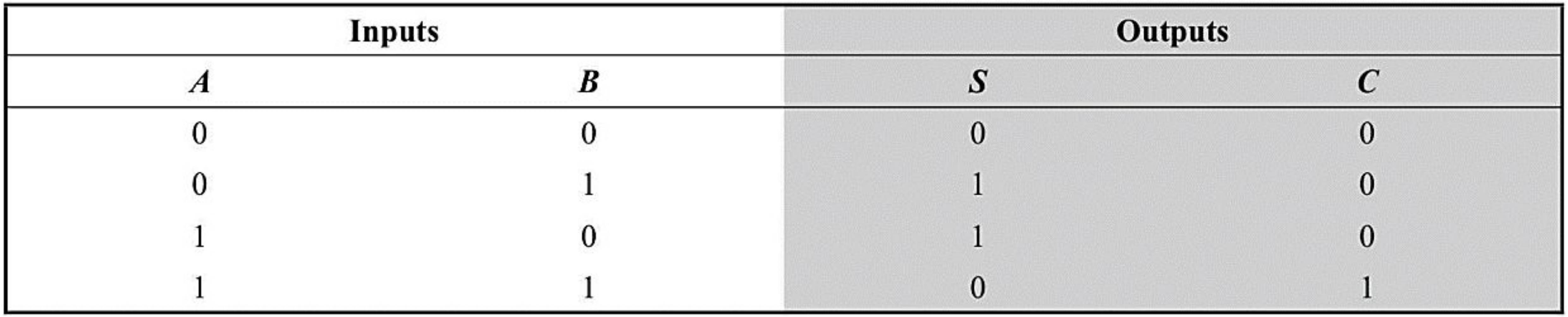
- **Half-Adder**

A half-adder is an arithmetic circuit block that can be used to add two bits. Such a circuit thus has two inputs that represent the two bits to be added and two outputs, with one producing the SUM output and the other producing the CARRY.

The truth table of a half-adder, showing all possible input combinations and the corresponding outputs. The Boolean expressions for the SUM and CARRY outputs are given by the equations

Arithmetic Circuits – Basic Building Blocks

- **Half-Adder**



Inputs		Outputs	
A	*B*	*S*	*C*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Arithmetic Circuits – Basic Building Blocks

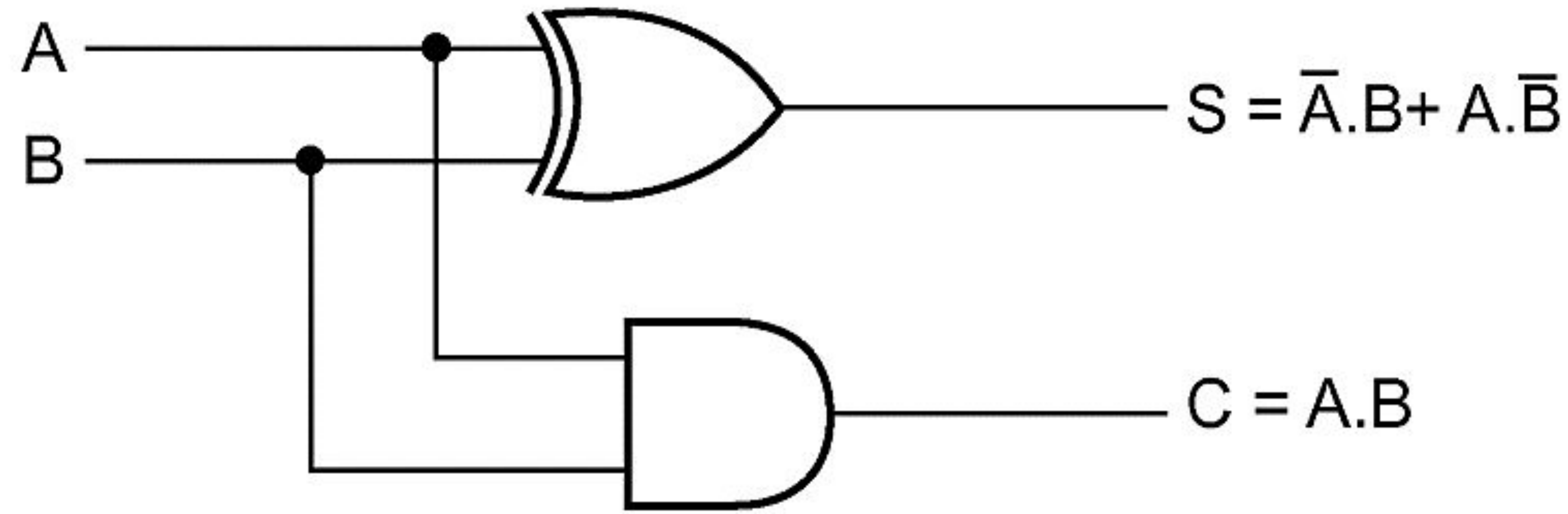
- **Half-Adder**

SUM S = A’.B +A.B’

CARRY C = A.B

Arithmetic Circuits – Basic Building Blocks

- **Half-Adder**



Arithmetic Circuits – Basic Building Blocks

- **Full-Adder**

- A full adder circuit is an arithmetic circuit block that can be used to add three bits to produce a SUM and a CARRY output. Such a building block becomes a necessity when it comes to adding binary numbers with a large number of bits.
- The full adder circuit overcomes the limitation of the half-adder, which can be used to add two bits only. Let us recall the procedure for adding larger binary numbers.

Arithmetic Circuits – Basic Building Blocks

- **Full-Adder**

- We begin with the addition of LSBs of the two numbers. We record the sum under the LSB column and take the carry, if any, forward to the next higher column bits.
- As a result, when we add the next adjacent higher column bits, we would be required to add three bits if there were a carry from the previous addition. We have a similar situation for the other higher column bits

Arithmetic Circuits – Basic Building Blocks

- **Full-Adder**

Inputs			Outputs	
A_n	B_n	C_{n-1}	S_n	C_n
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Arithmetic Circuits – Basic Building Blocks

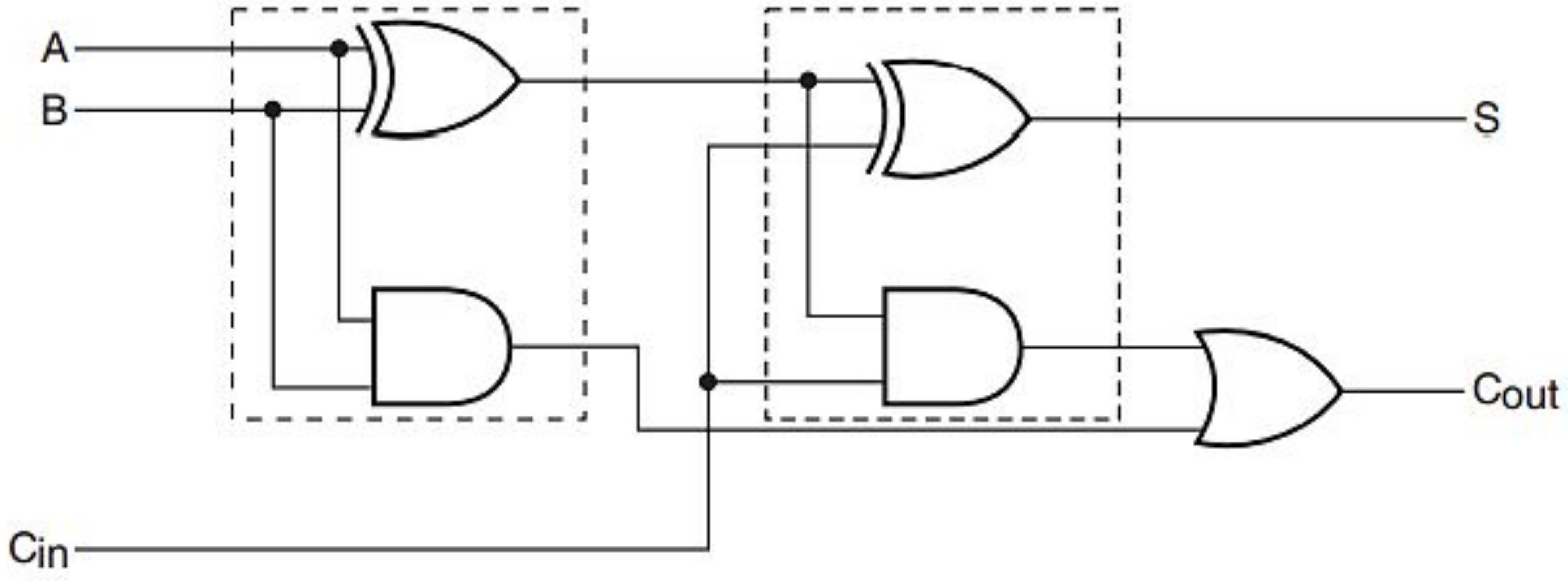
- **Full-Adder**

$$S = \overline{A}.\overline{B}.C_{\text{in}} + \overline{A}.B.\overline{C}_{\text{in}} + A.\overline{B}.\overline{C}_{\text{in}} + A.B.C_{\text{in}}$$

$$C_{\text{out}} = \overline{A}.B.C_{\text{in}} + A.\overline{B}.C_{\text{in}} + A.B.\overline{C}_{\text{in}} + A.B.C_{\text{in}}$$

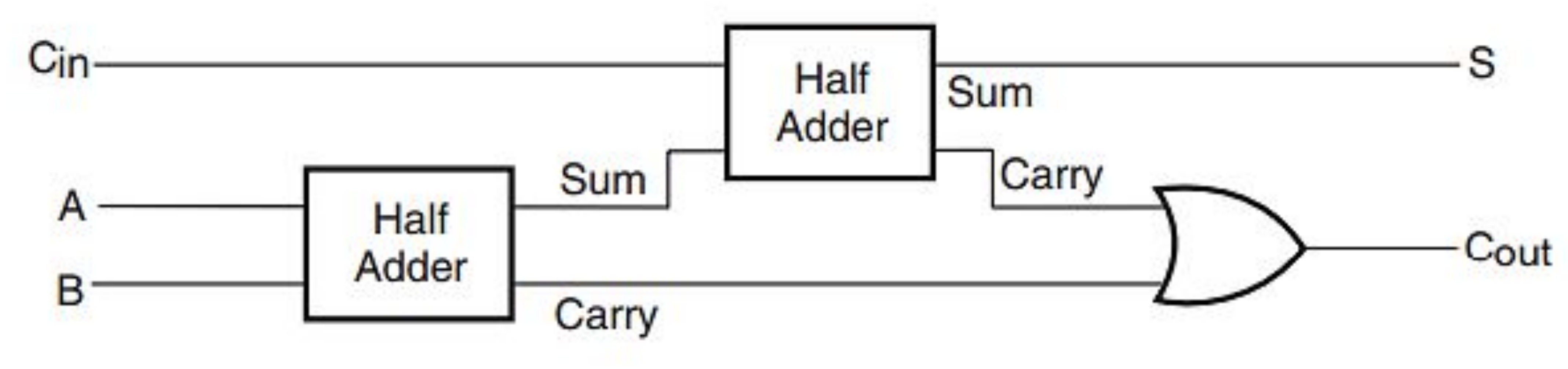
Arithmetic Circuits – Basic Building Blocks

- **Full-Adder**



Arithmetic Circuits – Basic Building Blocks

- **Full-Adder**

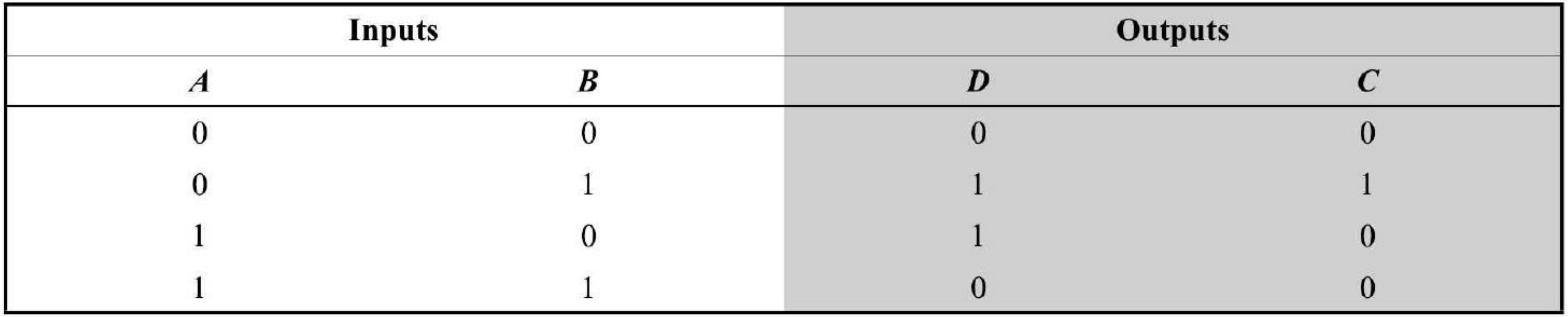


Arithmetic Circuits – Basic Building Blocks

- **Half-Substractor**

A half-subtractor is a combinational circuit that can be used to subtract one binary digit from another to produce a DIFFERENCE output and a BORROW output. The BORROW output here specifies whether a ‘1’ has been borrowed to perform the subtraction.

Arithmetic Circuits – Basic Building Blocks

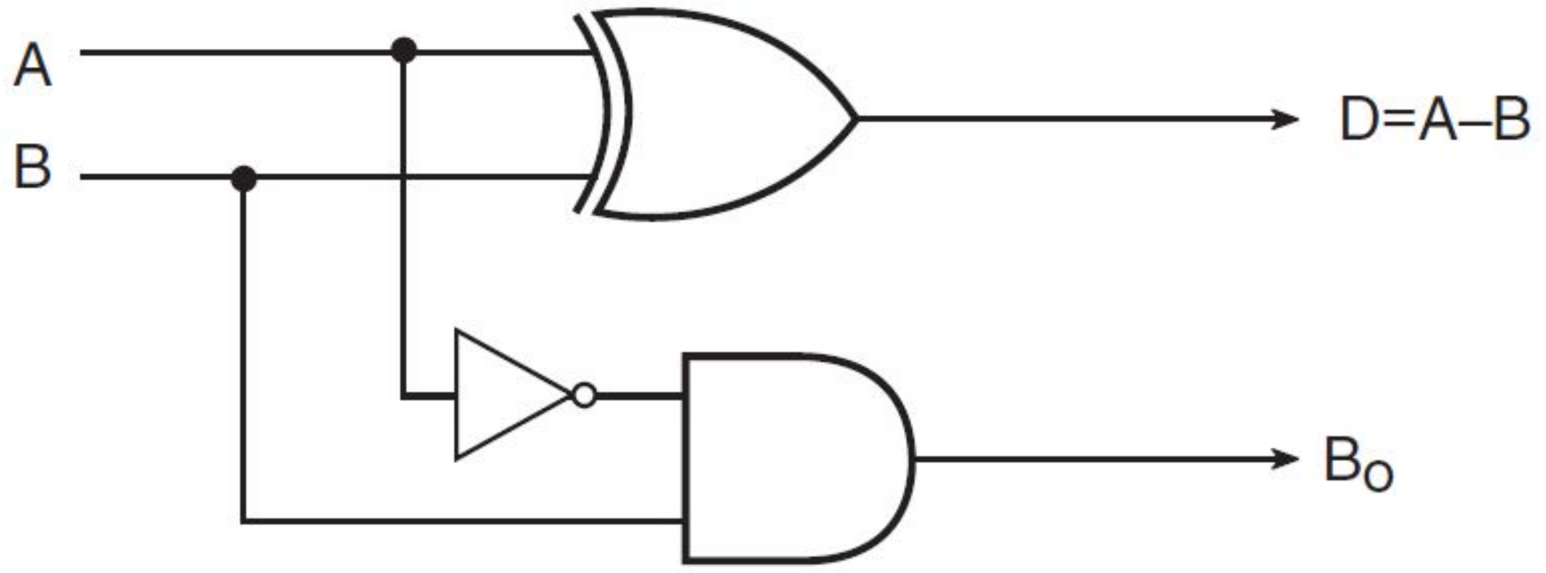


Inputs		Outputs	
A	***B***	***D***	***C***
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Arithmetic Circuits – Basic Building Blocks

$$D = \overline{A}B + A\overline{B} = A \oplus B$$
$$C = \overline{A}B$$

Arithmetic Circuits – Basic Building Blocks



Arithmetic Circuits – Basic Building Blocks

- **Full-Substractor**

A full subtractor performs subtraction operation on two bits, a minuend and a subtrahend, and also takes into consideration whether a ‘1’ has already been borrowed by the previous adjacent lower minuend bit or not.

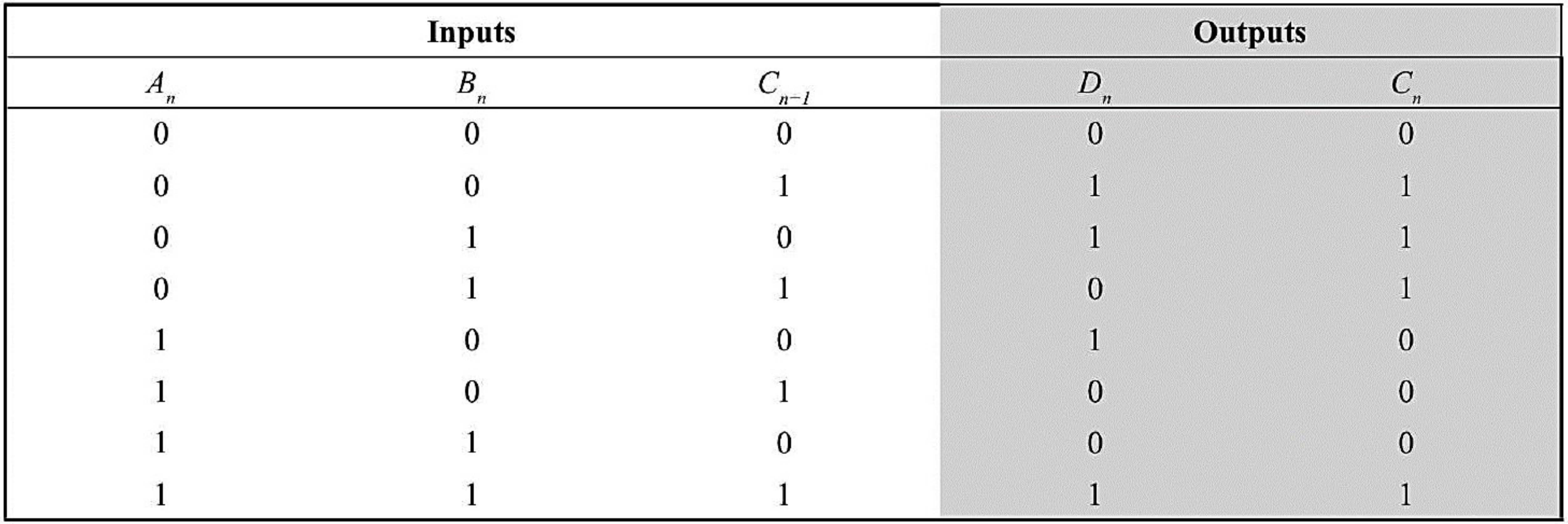
As a result, there are three bits to be handled at the input of a full subtractor, namely the two bits to be subtracted and a borrow bit designated as Bin.

Arithmetic Circuits – Basic Building Blocks

- **Full-Substractor**

There are two outputs, namely the DIFFERENCE output D and the BORROW output Bo. The BORROW output bit tells whether the minuend bit needs to borrow a ‘1’ from the next possible higher minuend bit.

Arithmetic Circuits – Basic Building Blocks



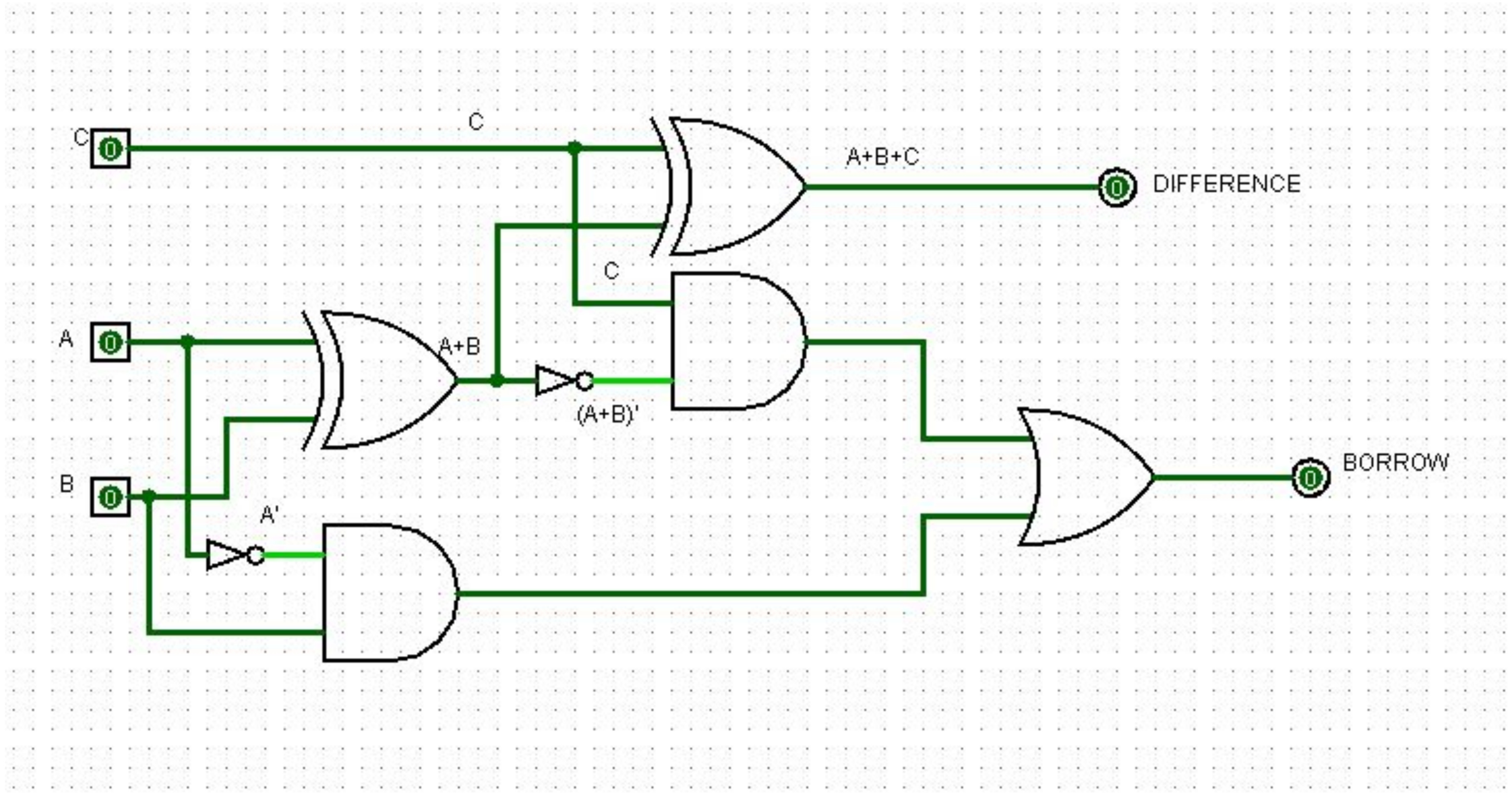
Inputs			Outputs	
A_n	B_n	C_{n-1}	D_n	C_n
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Arithmetic Circuits – Basic Building Blocks

- $$\begin{aligned} D &= A'B'C + A'BC' + AB'C' + ABC \\ &= A'(B'C + BC') + A(B'C' + BC) \\ &= A'(B \oplus C) + A(B \oplus C)' \\ D &= A \oplus B \oplus C \end{aligned}$$

$$\begin{aligned} B &= A'B'C + A'BC' + A'BC + ABC \\ &= A'(B'C + BC') + BC(A' + A) \\ B &= A'(B \oplus C) + BC \end{aligned}$$

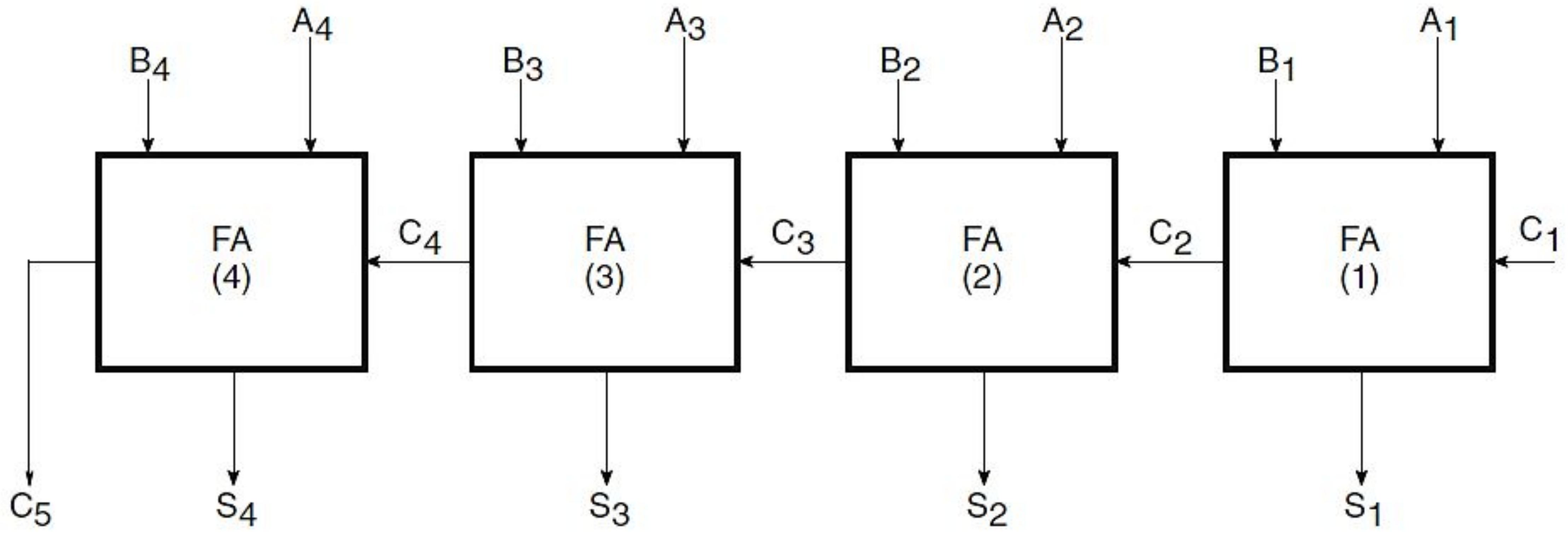
Arithmetic Circuits – Basic Building Blocks



Arithmetic Circuits – Four bit binary adder

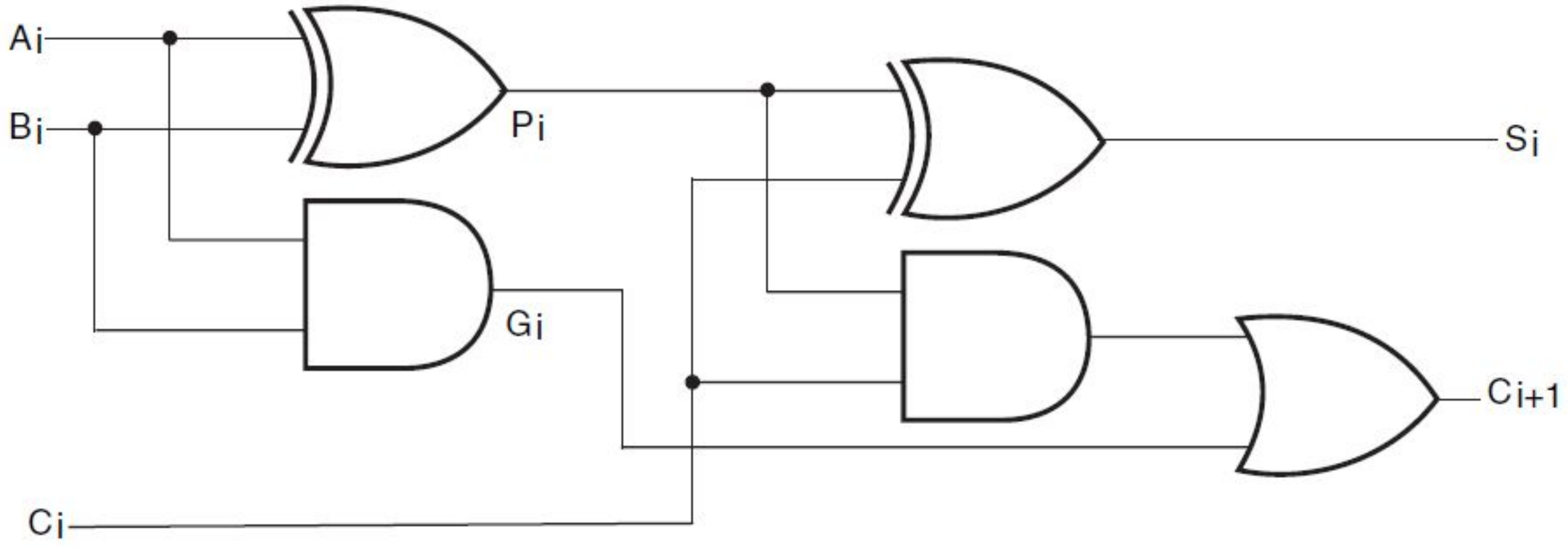
- Each of the adders is composed of four full adders (FAs) connected in cascade. The block schematic arrangement of a four-bit adder is represented in next slide for reference and furtherdiscussion.
- This type of adder is also called a parallel binary adder because all the bits of the augend and addend are present and are fed to the full adder blocks simultaneously.
- Theoretically, the addition operation in various full adders takes place simultaneously.

Arithmetic Circuits – Four bit binary adder



Arithmetic Circuits – Four bit binary adder

- Each Full adder unit



Arithmetic Circuits – Binary Multiplier

- Binary Multiplier is a combinational circuit that performs Binary Multiplication.
- Binary Multiplication is done in the same way as Decimal multiplication is done.
- Suppose 15 * 13

```
    15
    13
 ------
    45
 +15--
 ------
   195
```

Arithmetic Circuits – Binary Multiplier

Multiplicand	Multiplier	Product
0	0	0
0	1	0
1	0	0
1	1	1

Arithmetic Circuits – Binary Multiplier

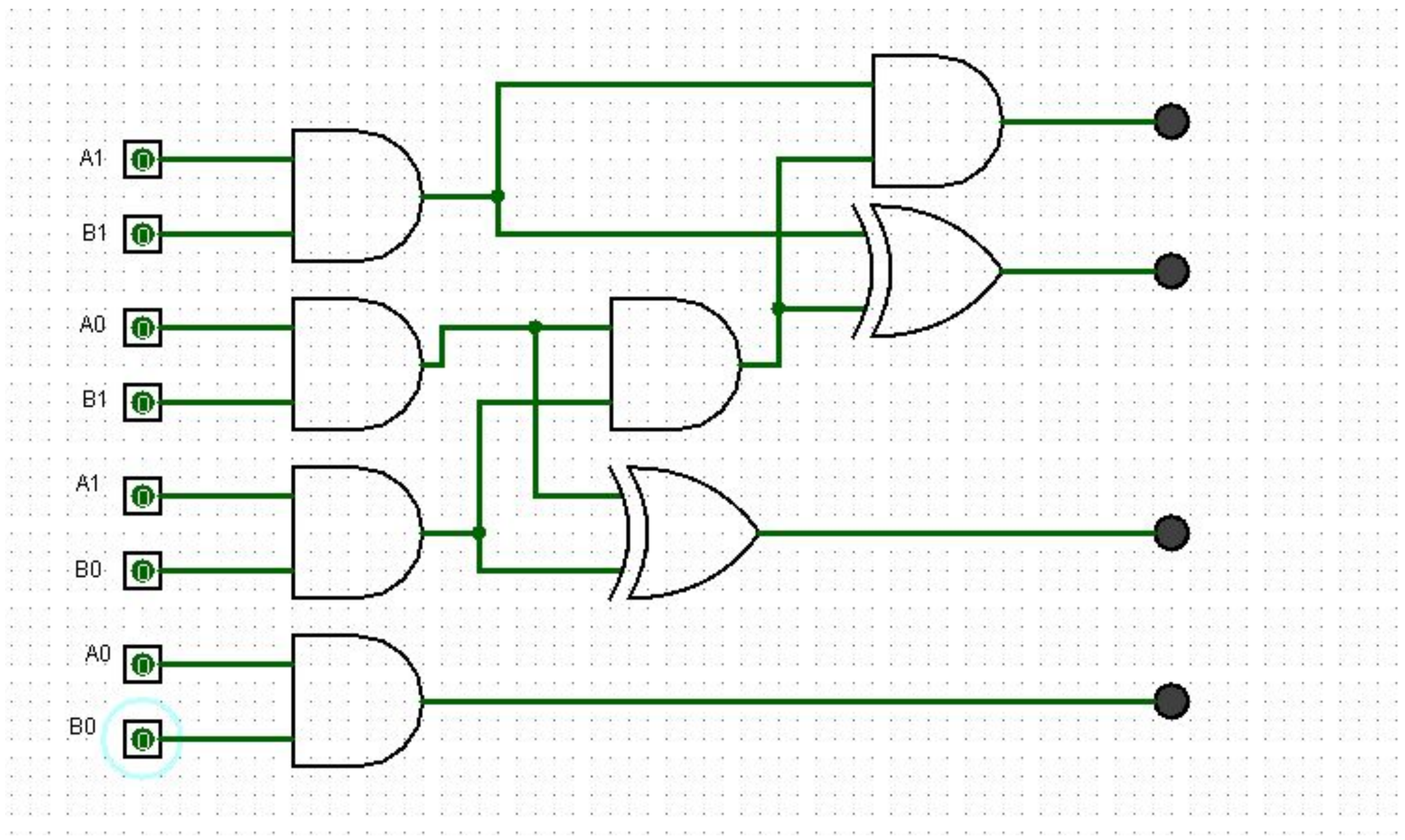
$$\begin{tikzcd}
A_1 \arrow[d] \arrow[dr] & A_0 \arrow[d] \arrow[dl] \\
B_1 & B_0
\end{tikzcd}$$

$$\begin{array}{cccl}
\hline
 & A_1 B_0 & A_0 B_0 & \text{-- Partial Products} \\
A_1B_1 & A_0B_1 & \text{--} & \\
\hline
P_2 & P_1 & P_0 &
\end{array}$$

Arithmetic Circuits – Binary Multiplier

$$\begin{array}{ccccccccl}
 &&&1&0&0&1&&\text{Multiplicand}\\
 &&\times&1&1&0&1&&\text{Multiplier}\\ \hline
 &&&1&0&0&1&\text{I}&\\
 &&0&0&0&0&&\text{II}&\text{Partial Products}\\
 &1&0&0&1&&&\text{III}&\\
1&0&0&1&&&&\text{IV}&\\ \hline
1&1&1&0&1&0&1&&\text{Final Product}
\end{array}$$

Arithmetic Circuits – Binary Multiplier

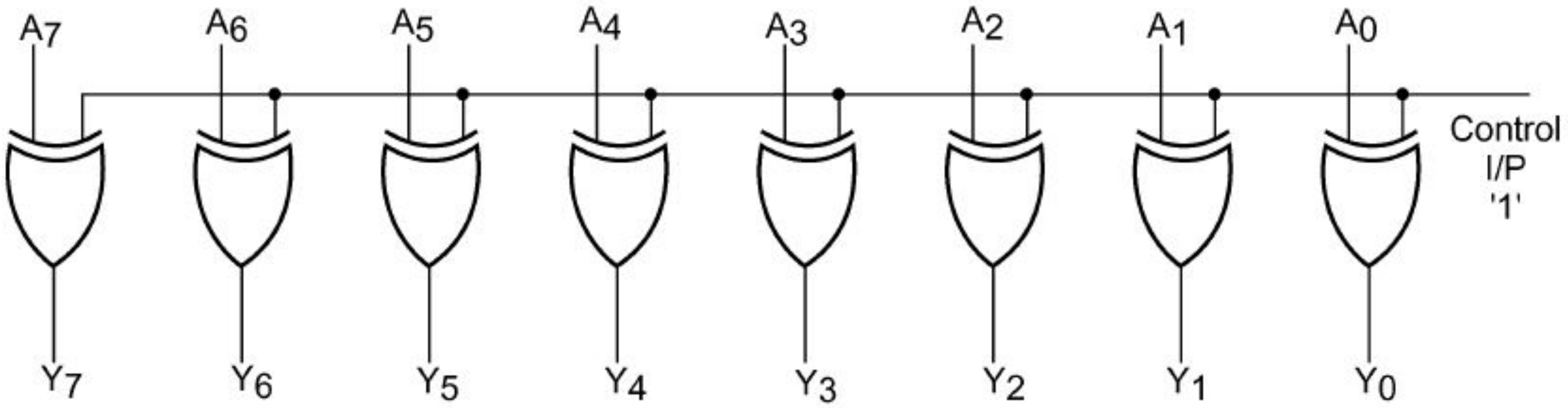


Arithmetic Circuits – 1’ Complement OR **1-bit controlled Inverter**

- A controlled inverter is needed when an adder is to be used as a subtractor.
- As outlined earlier, subtraction is nothing but addition of the 2’s complement of the subtrahend to the minuend. Thus, the first step towards practical implementation of a subtractor is to determine the 2’s complement of the subtrahend. And for this, one needs firstly to find 1’s complement.
- A controlled inverter is used to find1’s complement. A one-bit controlled inverter is nothing but a two-input EX-OR gate with one of its inputs treated as a control input.

Arithmetic Circuits – 1' Complement OR 1-bit controlled Inverter

- When the control input is LOW, the input bit is passed as such to the output. (Recall the truth table of an EX-OR gate.)
- When the control input is HIGH, the input bit gets complemented at the output.



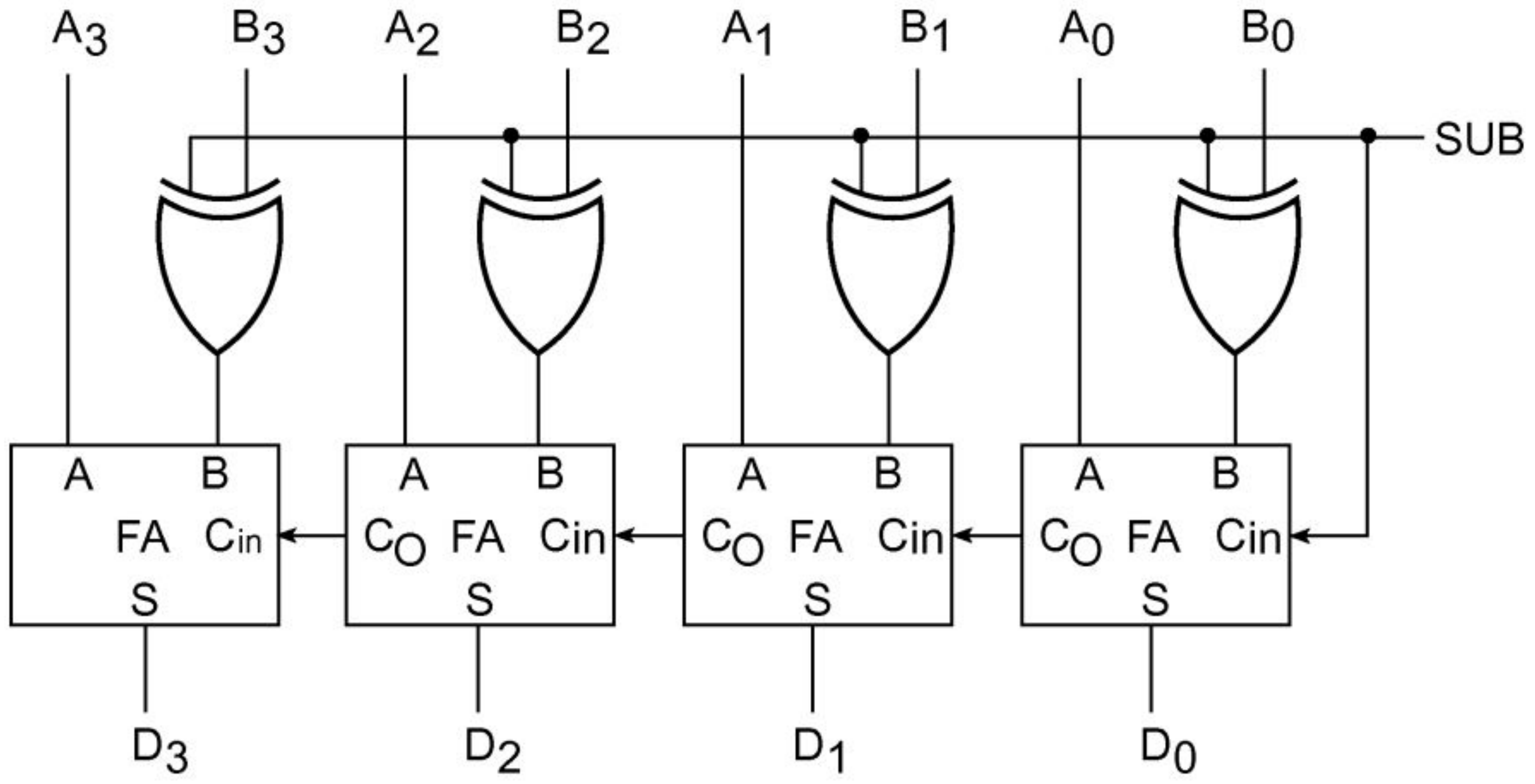
Arithmetic Circuits – Adder-Substractor

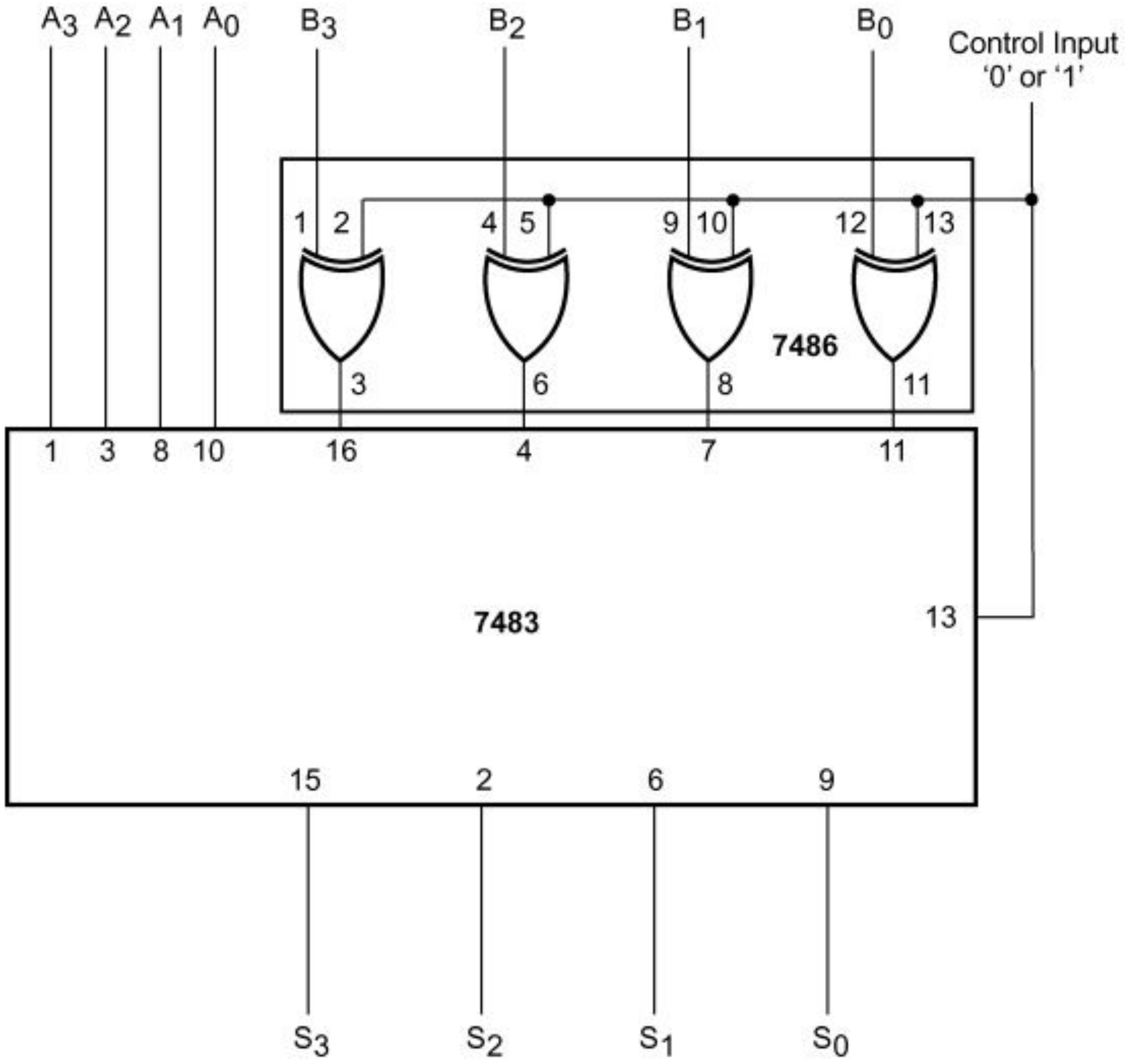
- Subtraction of two binary numbers can be accomplished by adding 2’s complement of the subtrahend to the minuend and disregarding the final carry, if any.
- If the MSB bit in the result of addition is a ‘0’, then the result of addition is the correct answer.
- If the MSB bit is a ‘1’, this implies that the answer has a negative sign. The true magnitude in this case is given by 2’s complement of the result of addition.

Arithmetic Circuits – Adder-Substractor

- The control input here is referred to as the SUB input. When the SUB input is in logic ‘0’ state, the four bits of the binary number (B3, B2, B1, B0) are passed on as such to the B inputs of the corresponding full adders.
- The outputs of the full adders in this case give the result of addition of the two numbers. When the SUB input is in logic ‘1’ state, four bits of one of the numbers, (B3 B2 B1 B0) in the present case, get complemented.
- If the same ‘1’ is also fed to the CARRY-IN of the LSB full adder, what we finally achieve is the addition of 2’s complement and not 1’s complement.

Arithmetic Circuits – Adder-Substractor





A_3
A_2
A_1
A_0
B_3
B_2
B_1
B_0
Control Input
'0' or '1'
7486
1
2
3
4
5
6
9
10
8
12
13
11
7483
1
3
8
10
16
4
7
11
13
15
2
6
9
S_3
S_2
S_1
S_0

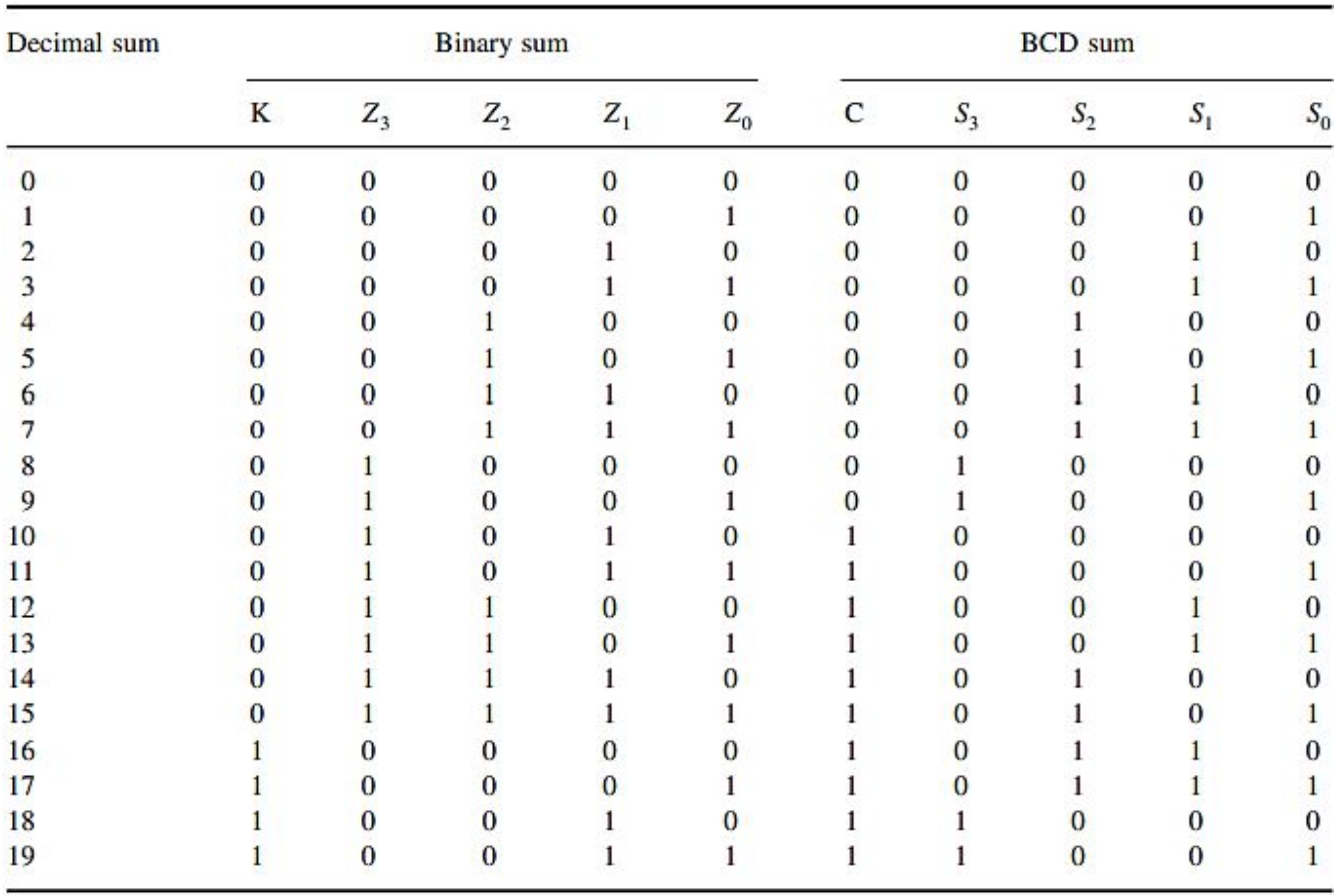
BCD Adder

- A BCD adder is used to perform the addition of BCD numbers. A BCD digit can have any of the ten possible four-bit binary representations, that is, 0000, 0001, , 1001, the equivalent of decimal numbers 0, 1, ….. , 9.
- When we set out to add two BCD digits and we assume that there is an input carry too, the highest binary number that we can get is the equivalent of decimal number 19 (9 + 9 +A BCD adder is used to perform the addition of BCD numbers.

BCD Adder

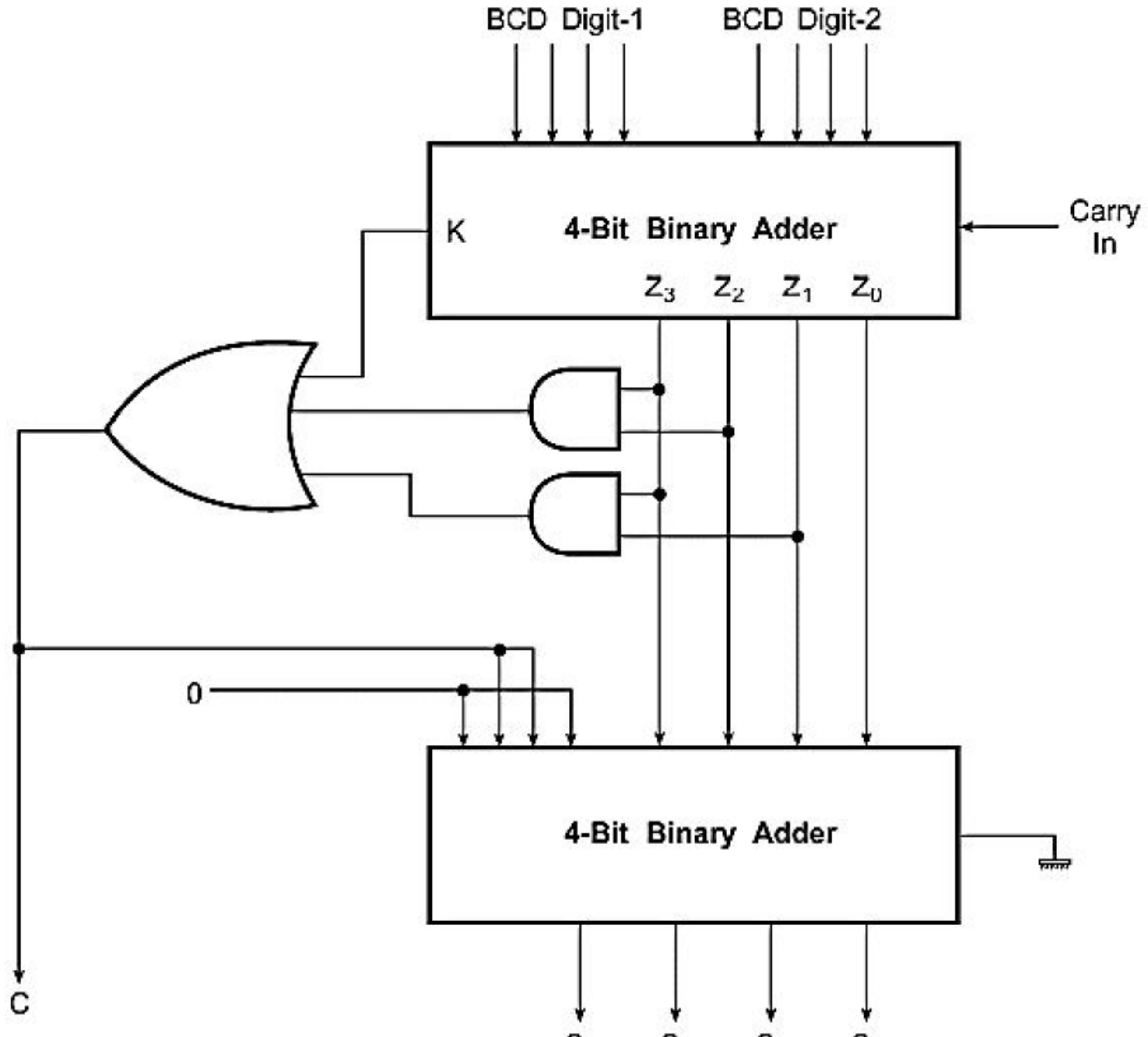
- A BCD digit can have any of the ten possible four-bit binary representations, that is, 0000, 0001, ... , 1001, the equivalent of decimal numbers 0, 1,…... , 9. When we set out to add two BCD digits and we assume that there is an input carry too, the highest binary number that we can get is the equivalent of decimal number 19 (9 + 9 + 1).

BCD Adder

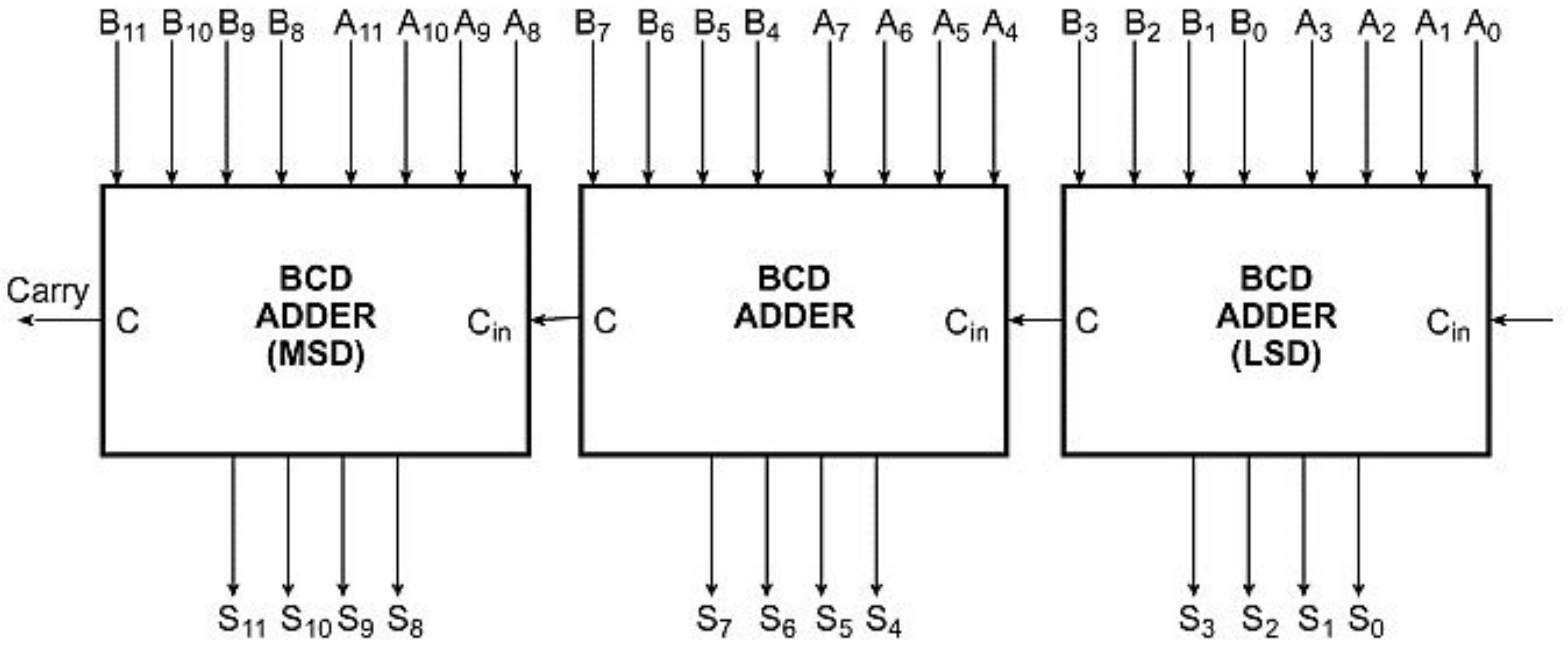


Decimal sum	Binary sum					BCD sum				
	K	Z_3	Z_2	Z_1	Z_0	C	S_3	S_2	S_1	S_0
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1
2	0	0	0	1	0	0	0	0	1	0
3	0	0	0	1	1	0	0	0	1	1
4	0	0	1	0	0	0	0	1	0	0
5	0	0	1	0	1	0	0	1	0	1
6	0	0	1	1	0	0	0	1	1	0
7	0	0	1	1	1	0	0	1	1	1
8	0	1	0	0	0	0	1	0	0	0
9	0	1	0	0	1	0	1	0	0	1
10	0	1	0	1	0	1	0	0	0	0
11	0	1	0	1	1	1	0	0	0	1
12	0	1	1	0	0	1	0	0	1	0
13	0	1	1	0	1	1	0	0	1	1
14	0	1	1	1	0	1	0	1	0	0
15	0	1	1	1	1	1	0	1	0	1
16	1	0	0	0	0	1	0	1	1	0
17	1	0	0	0	1	1	0	1	1	1
18	1	0	0	1	0	1	1	0	0	0
19	1	0	0	1	1	1	1	0	0	1

BCD Adder


BCD Digit-1
BCD Digit-2
4-Bit Binary Adder
K
Carry In
Z_3 Z_2 Z_1 Z_0
0
4-Bit Binary Adder
C

BCD Adder



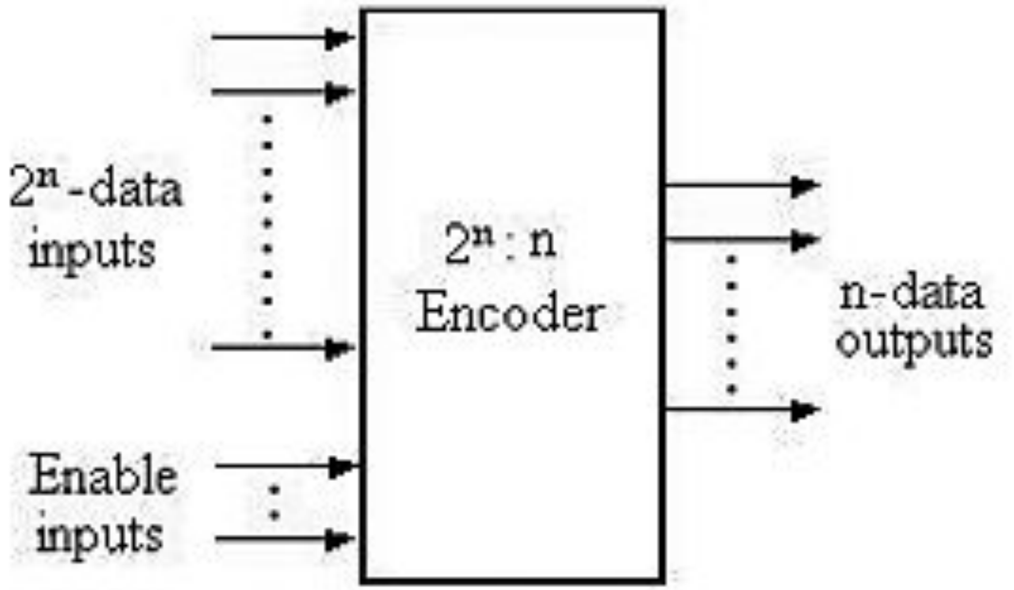
B_{11} B_{10} B_9 B_8 A_{11} A_{10} A_9 A_8
B_7 B_6 B_5 B_4 A_7 A_6 A_5 A_4
B_3 B_2 B_1 B_0 A_3 A_2 A_1 A_0
Carry
C
BCD ADDER (MSD)
C_{in}
C
BCD ADDER
C_{in}
C
BCD ADDER (LSD)
C_{in}
S_{11} S_{10} S_9 S_8
S_7 S_6 S_5 S_4
S_3 S_2 S_1 S_0

Encoder

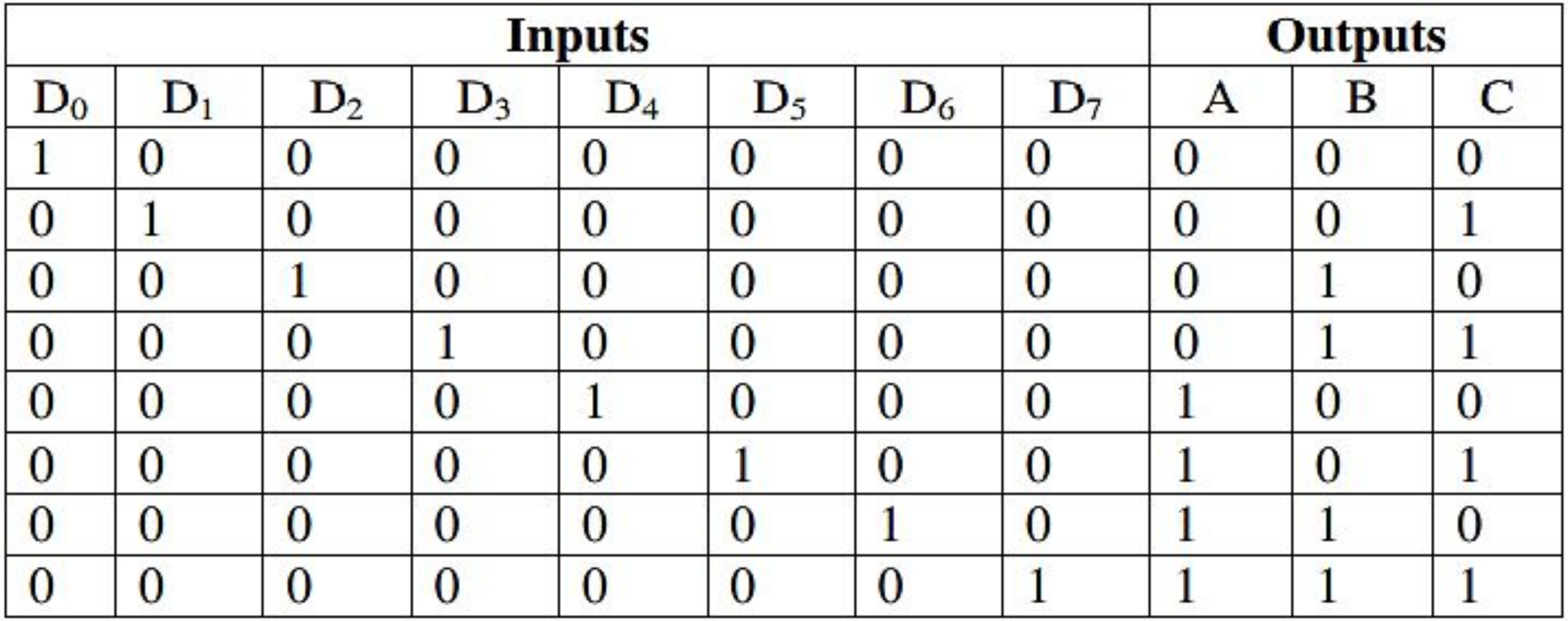
- An encoder is a digital circuit that achieves the inverse operation of a decoder. Therefore, the opposite of the decoding process is called encoding. An encoder is a combinational circuit that converts binary information from $\mathbf{2^n}$ input lines to a maximum of **‘n’** exclusive output lines.

Encoder

- It has **2^n** input lines, only one ''1'' is active at any time and **'n'** output lines. It encodes one active inputs to a coded binary output with 'n' bits. In an encoder, the number of outputs is less than the number of inputs.

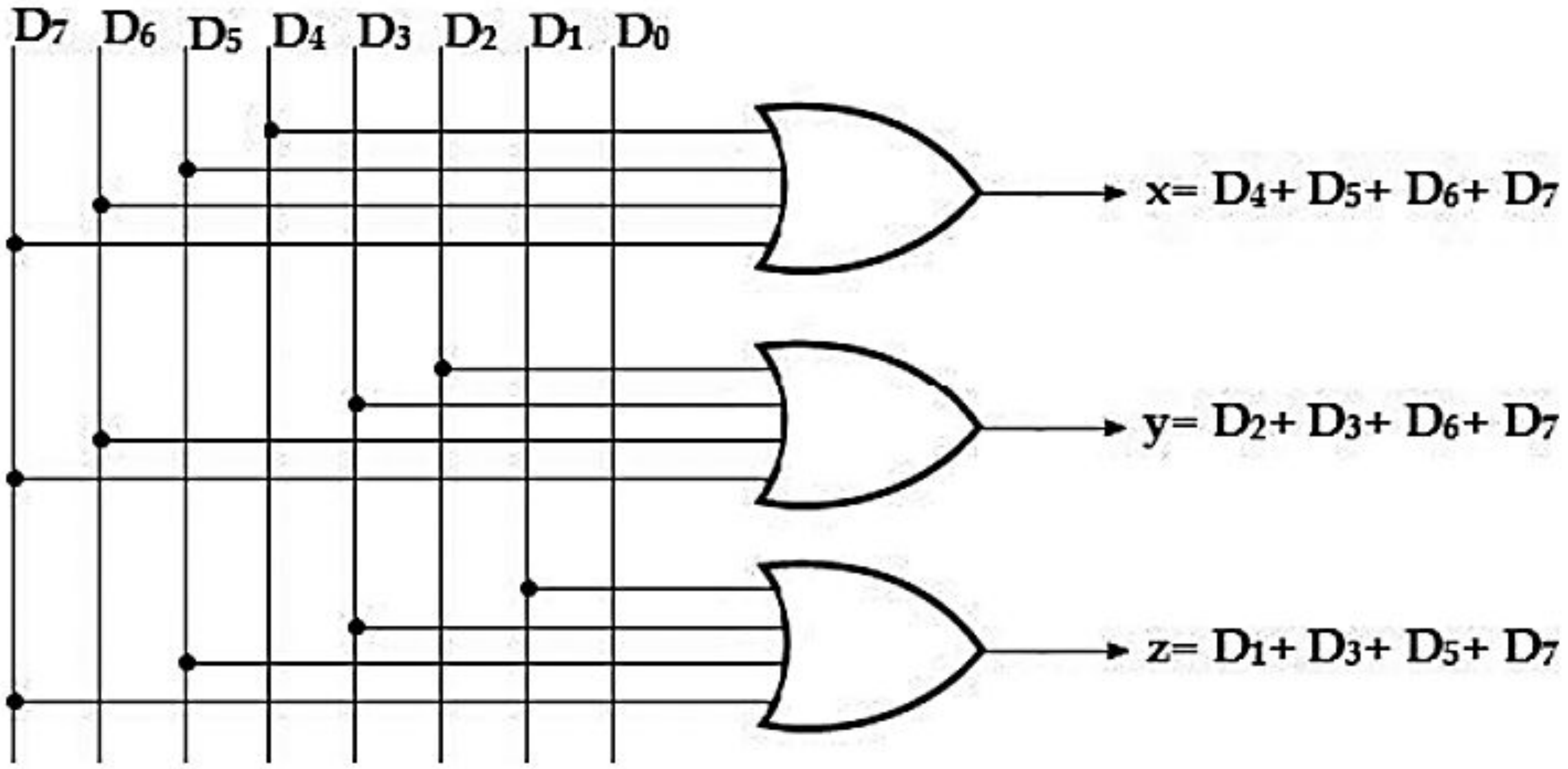


Encoder



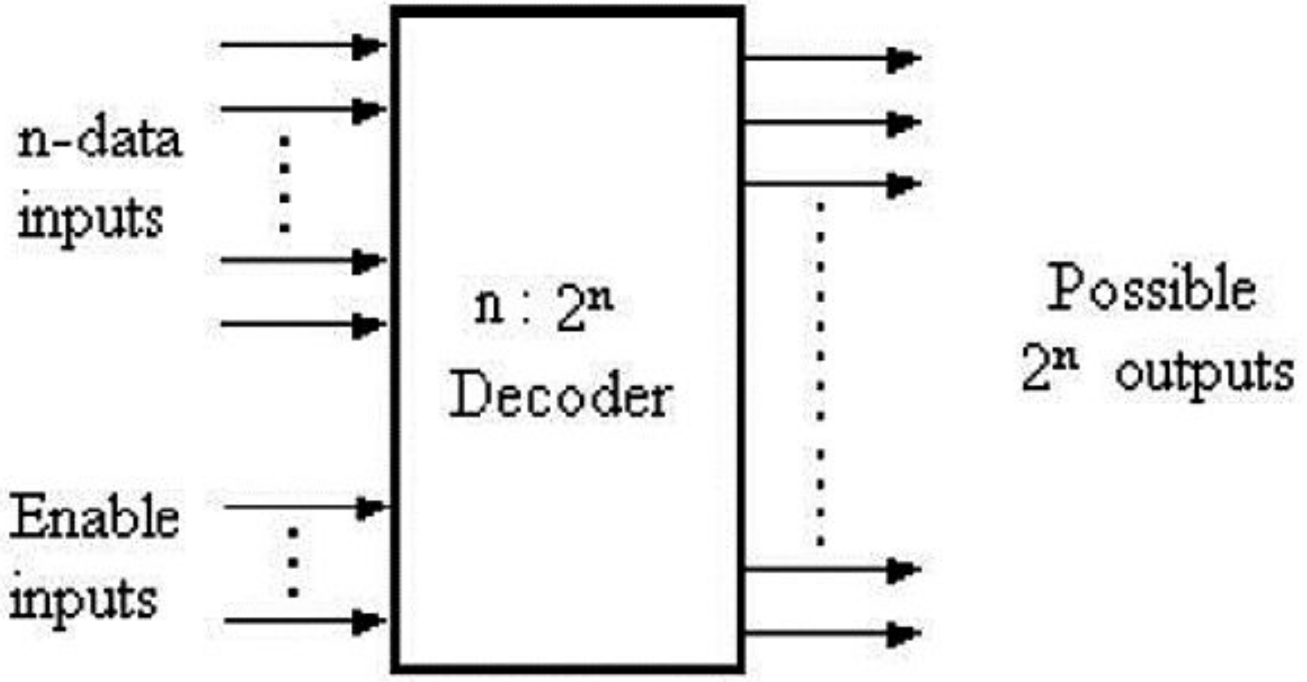
Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	A	B	C
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Encoder



Decoder

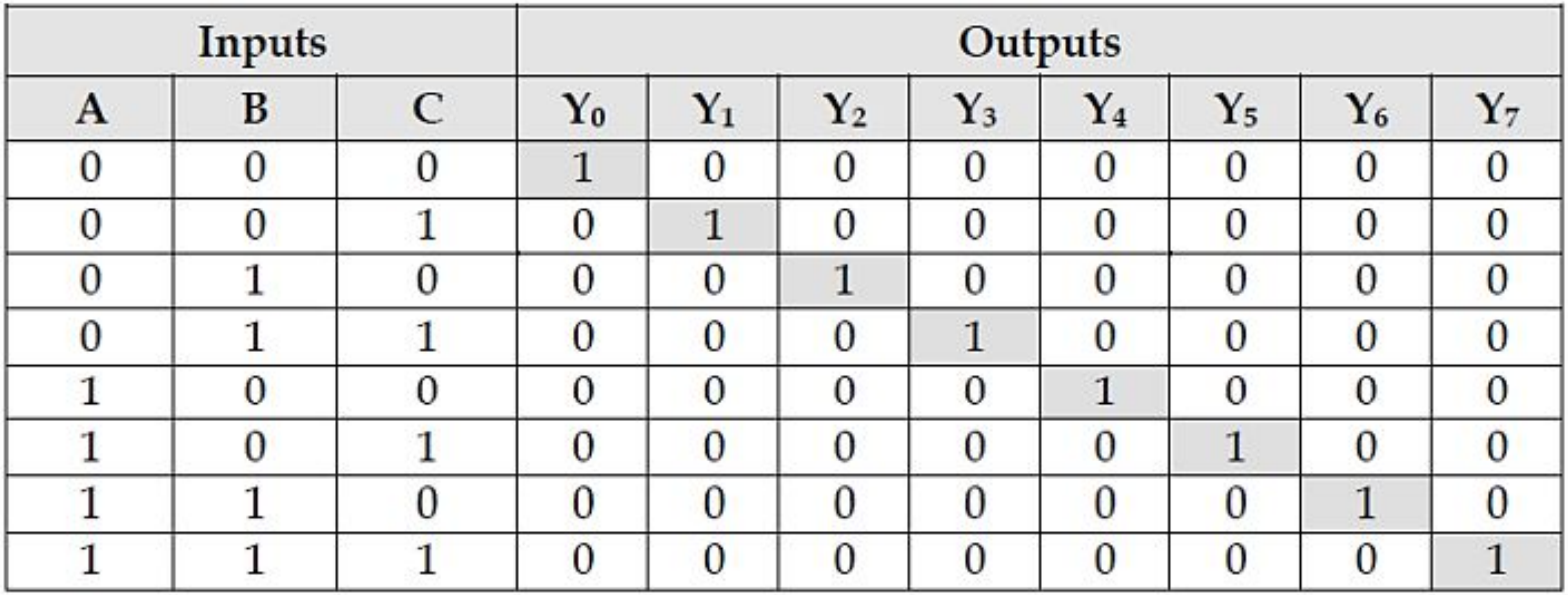
- A decoder is a combinational circuit that converts binary information from ‘n’ input lines to a maximum of ‘‘**2^n**’’ unique output lines.



Decoder

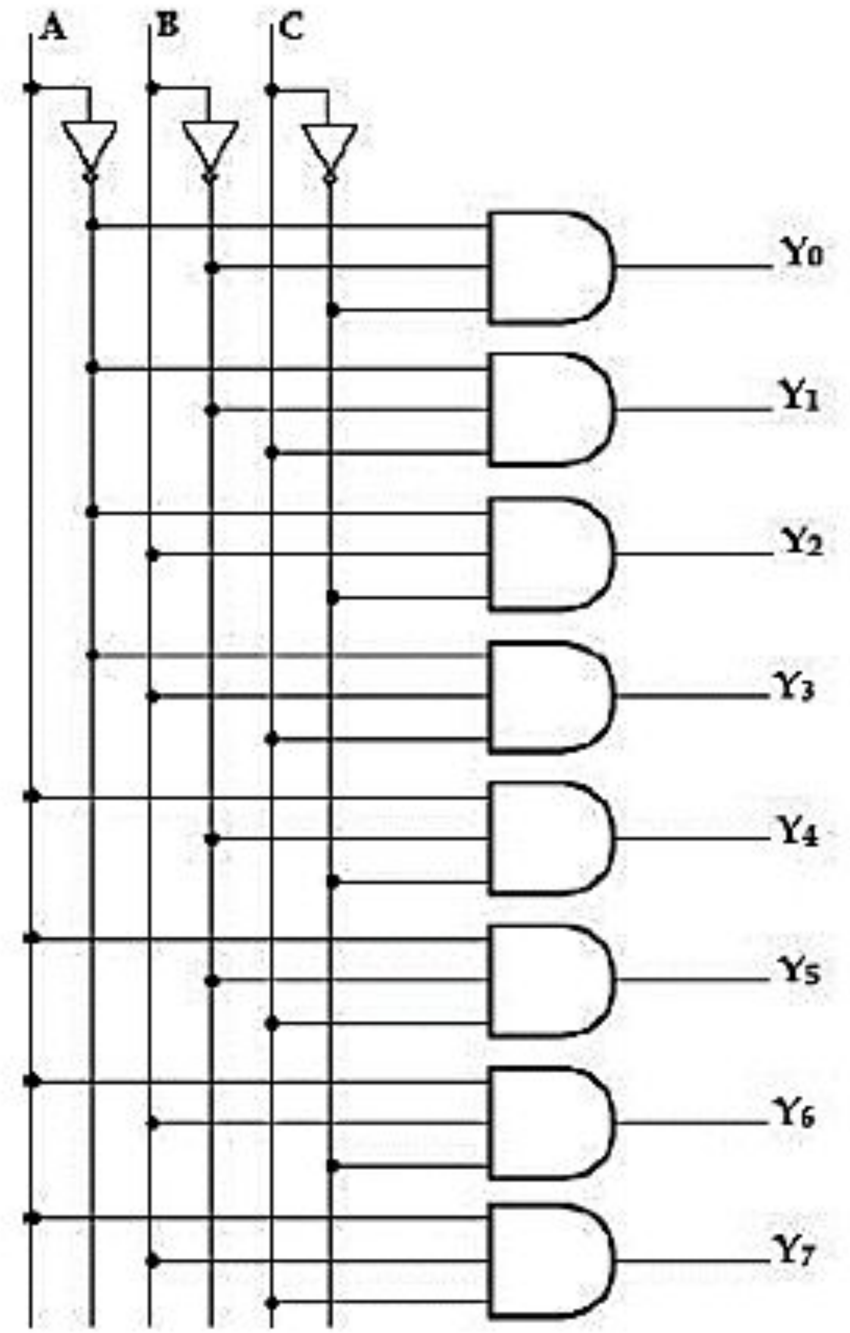
- The encoded information is presented as ‘n’ inputs producing ‘‘$\mathbf{2^{n}}$’’ possible outputs. The ‘‘$\mathbf{2^{n}}$’’ output values are from 0 through ‘‘$\mathbf{2^{n-1}}$’’.
- A decoder is provided with enable inputs to activate decoded output based on data inputs. When any one enable input is unasserted, all outputs of decoder are disabled.

Decoder



Inputs			Outputs							
A	B	C	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Decoder

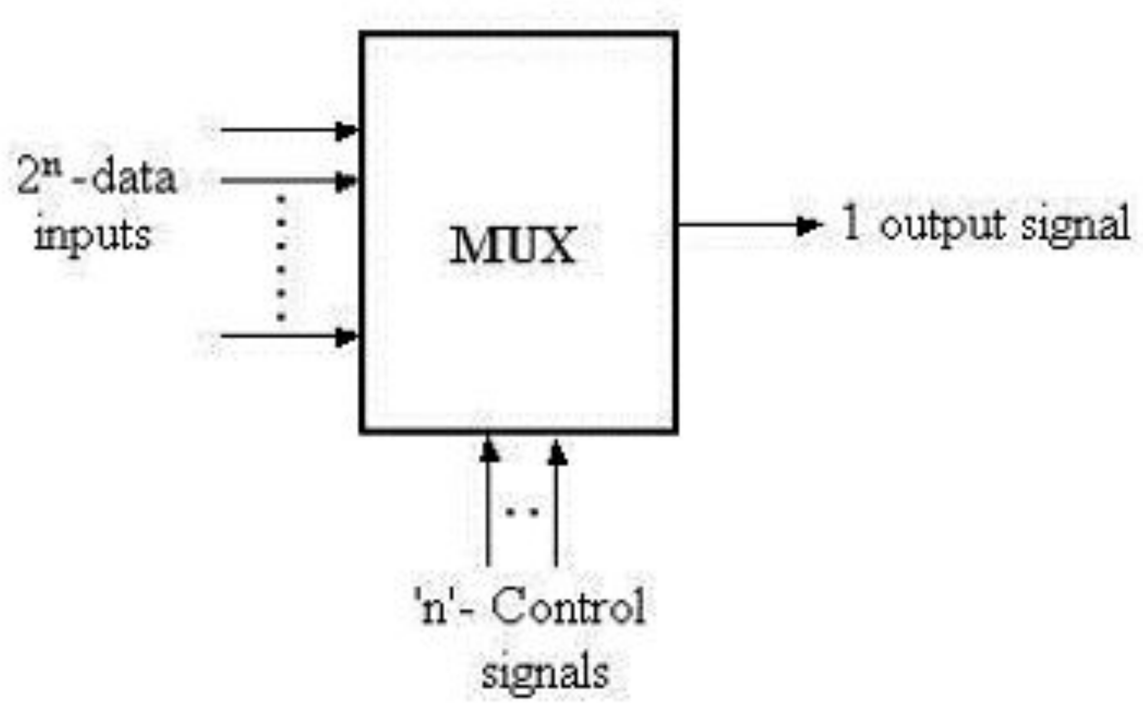


Multiplexer (Data Selector)

- A multiplexer or MUX, is a combinational circuit with more than one input line, one output line and more than one selection line.
- A multiplexer selects binary information present from one of many input lines, depending upon the logic status of the selection inputs, and directions it to the output line.
- Generally, there are ‘‘$\mathbf{2^n}$’’ input lines and **n** selection lines whose bit combinations determine which input is selected.

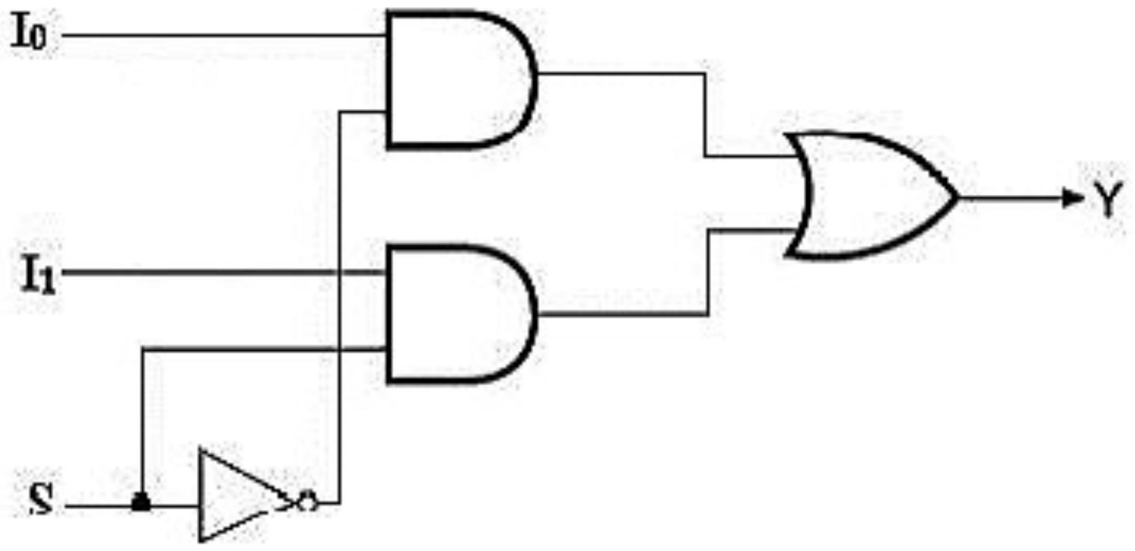
Multiplexer (Data Selector)

- The multiplexer is labelled as MUX in block diagrams. A multiplexer is also called a data selector, since it selects one of many inputs and leads the binary information to the output line.



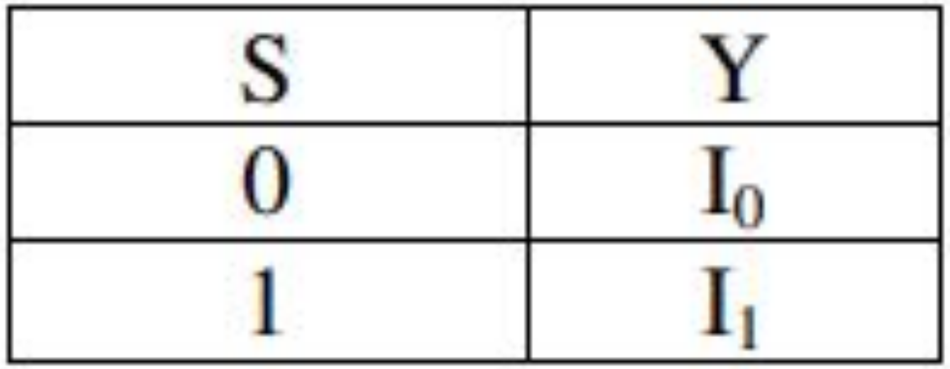
Multiplexer (Data Selector) – 2-to-1 line Multiplexer

- This circuit has two data input lines, one output line and one selection line. When S= 0, the upper AND gate is enabled and I0 has a path to the output.
- When S=1, the lower AND gate is enabled and I1 has a path to the output



Multiplexer (Data Selector) –
2-to-1 line Multiplexer

- The multiplexer acts like an electronic switch that selects one of the two sources.

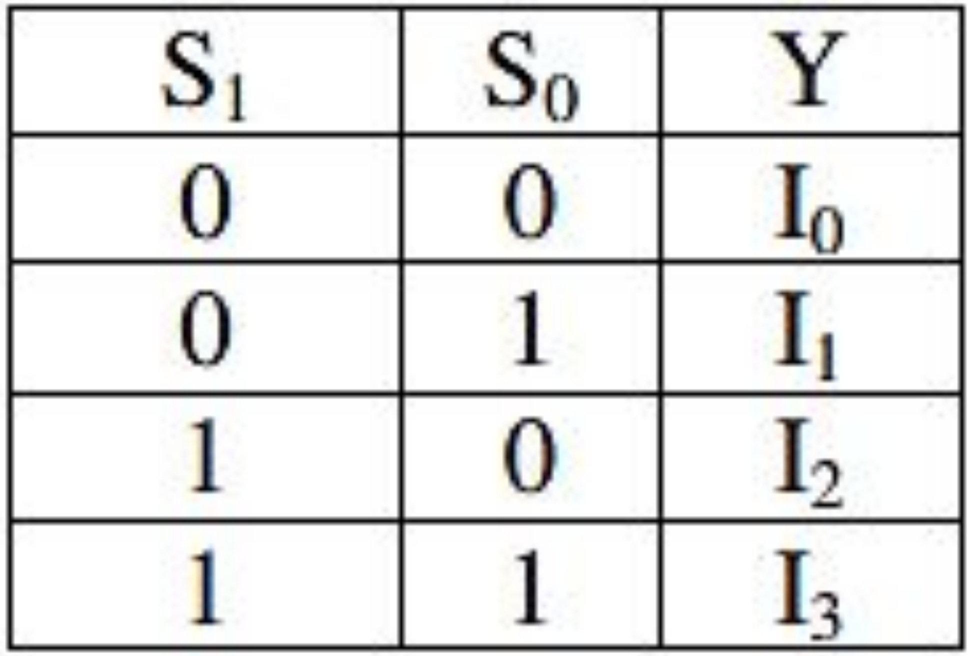


S	Y
0	I_0
1	I_1

Multiplexer (Data Selector) – 4-to-1 line Multiplexer

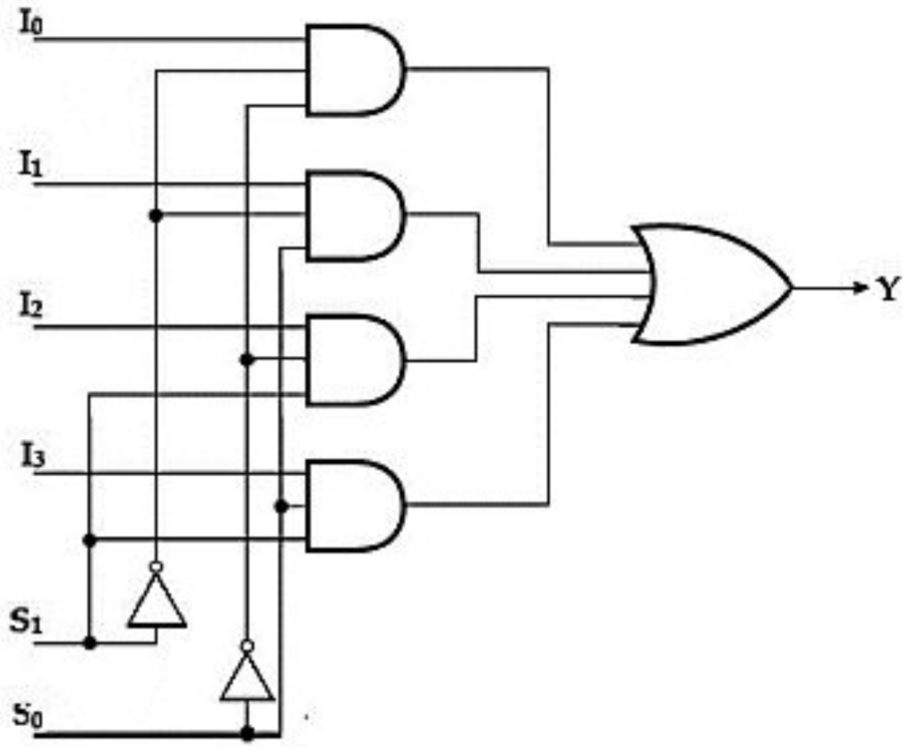
- A 4-to-1-line multiplexer has four (**2^n**) input lines, two (**n**) select lines and one output line.
- It is the multiplexer comprising of four input channels and information of one of the station can be selected and transmitted to an output line according to the select inputs combinations.
- Selection of one of the four input station is possible by two selection inputs.
- Each of the four inputs I0 through I3, is applied to one input of AND gate. Selection lines S1 and S0 are decoded to select a particular AND gate. The outputs of the AND gate are applied to a single OR gate that provides the 1-line output.

Multiplexer (Data Selector) –
4-to-1 line Multiplexer



S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

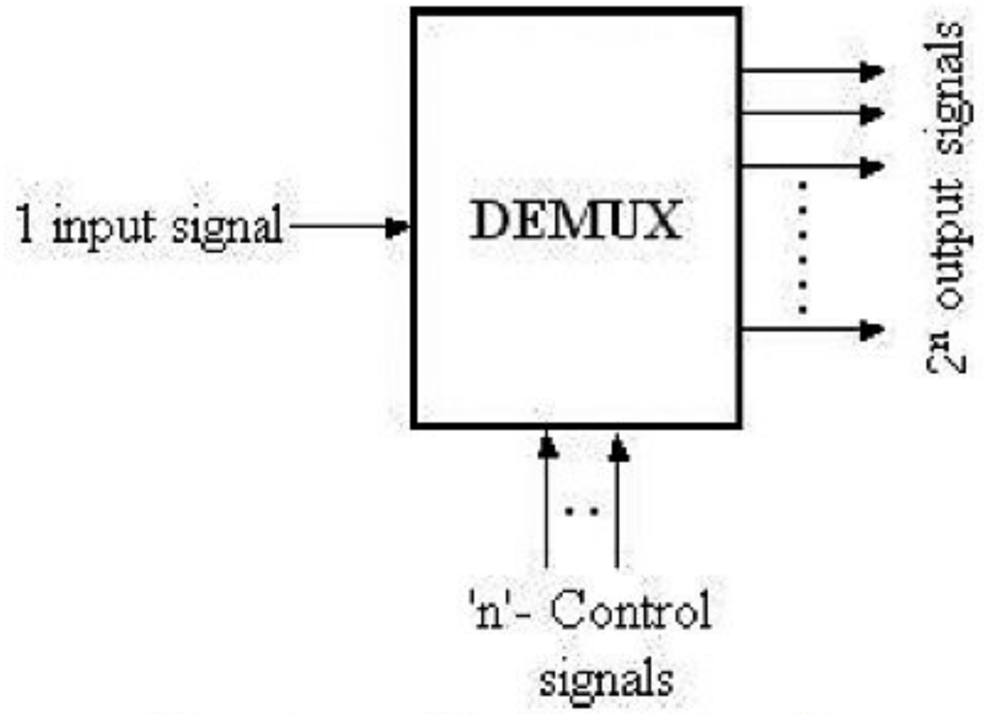
Multiplexer (Data Selector) –
4-to-1 line Multiplexer



$$Y = I_0S_1'S_0' + I_1S_1'S_0 + I_2S_1S_0' + I_3S_1S_0.$$

Demultiplexer

- Demultiplex means one into many. Demultiplexing is the process of taking information from one input and transmitting the same over on of several outputs

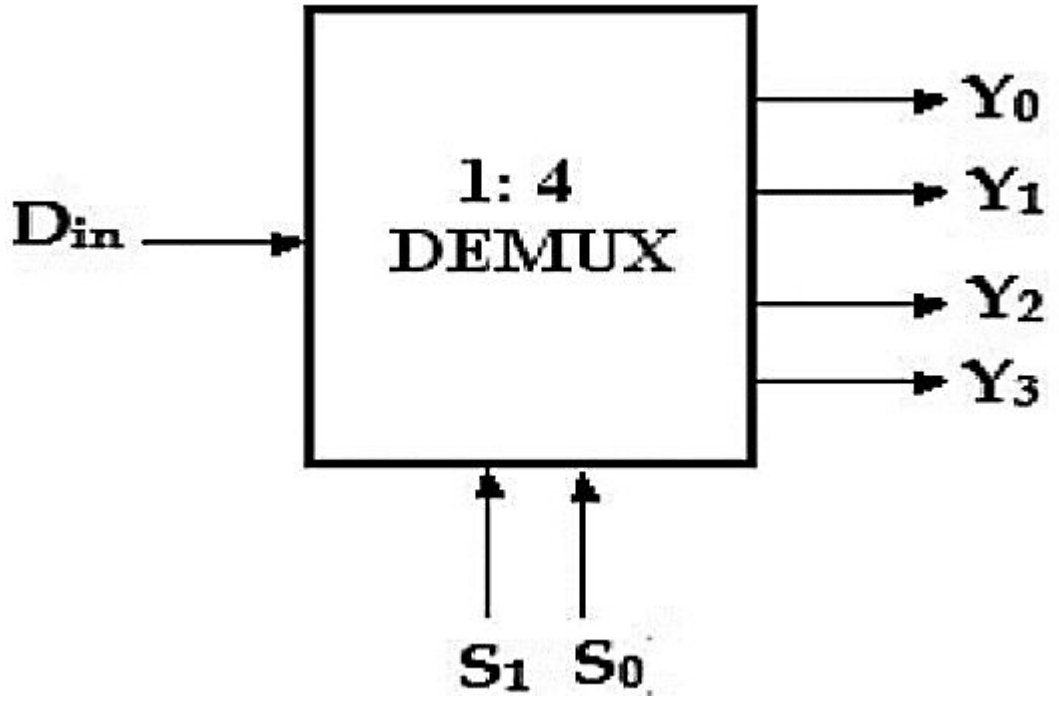


Demultiplexer

- A decoder is similar to a demultiplexer, with one exception-there is no data input.
- The only inputs are the Select signals.
- Decoder is a logic circuit that converts n-bit binary input code into m output lines.
- The decoders presented here are called **n-to-m** line decoders where **m < 2n**.
- Each output line will be activated for only one of the possible combination of inputs.

Demultiplexer –
1-to-4 line Demultiplexer

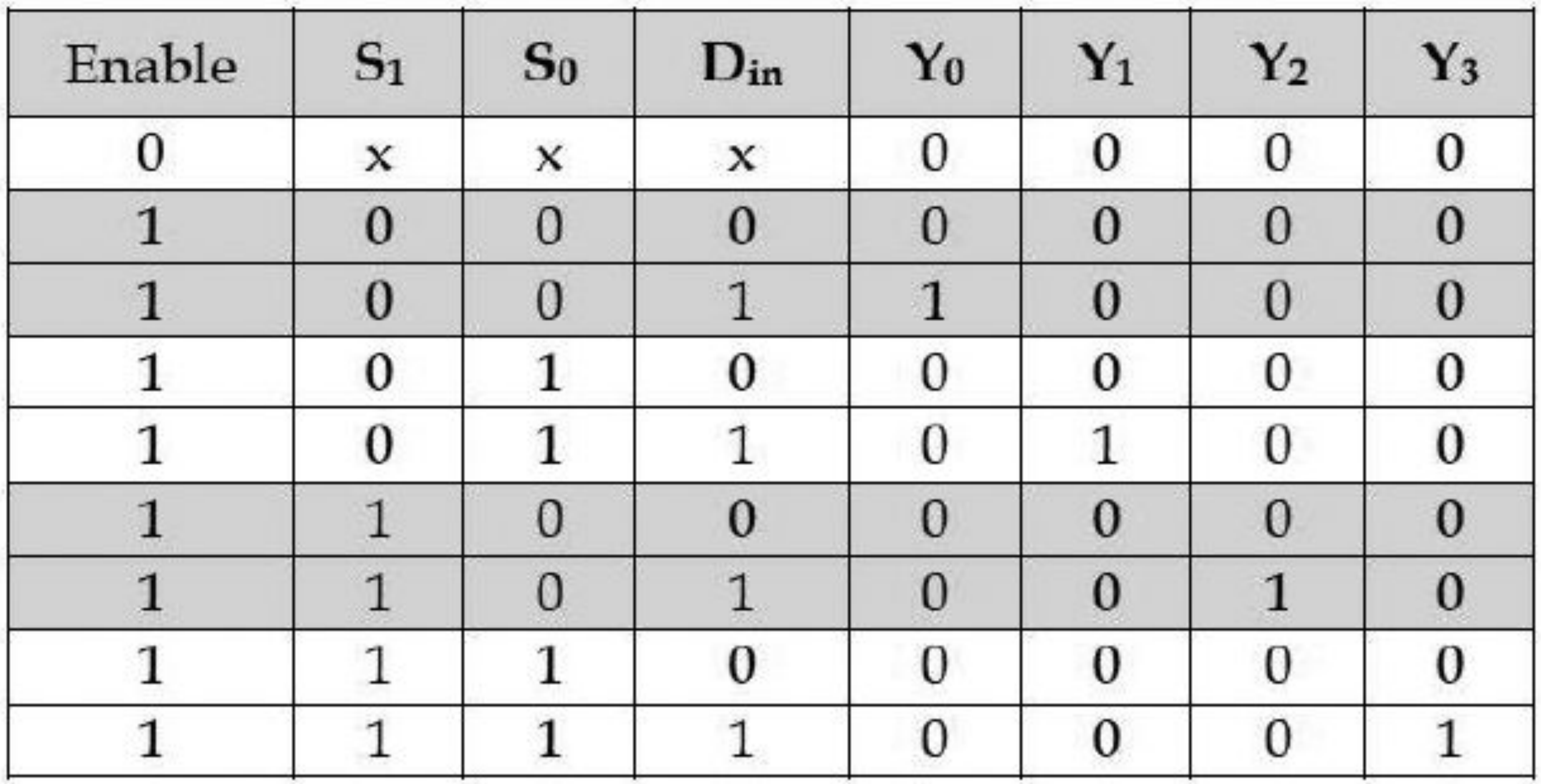
- A 1-to-4 demultiplexer has a single input, Din, four outputs (Y0 to Y3) and two select inputs (S1 and S0).



Demultiplexer –
1-to-4 line Demultiplexer

- The input variable Din has a path to all four outputs, but the input information is directed to only one of the output lines. The truth table of the 1-to-4 demultiplexer is shown below.

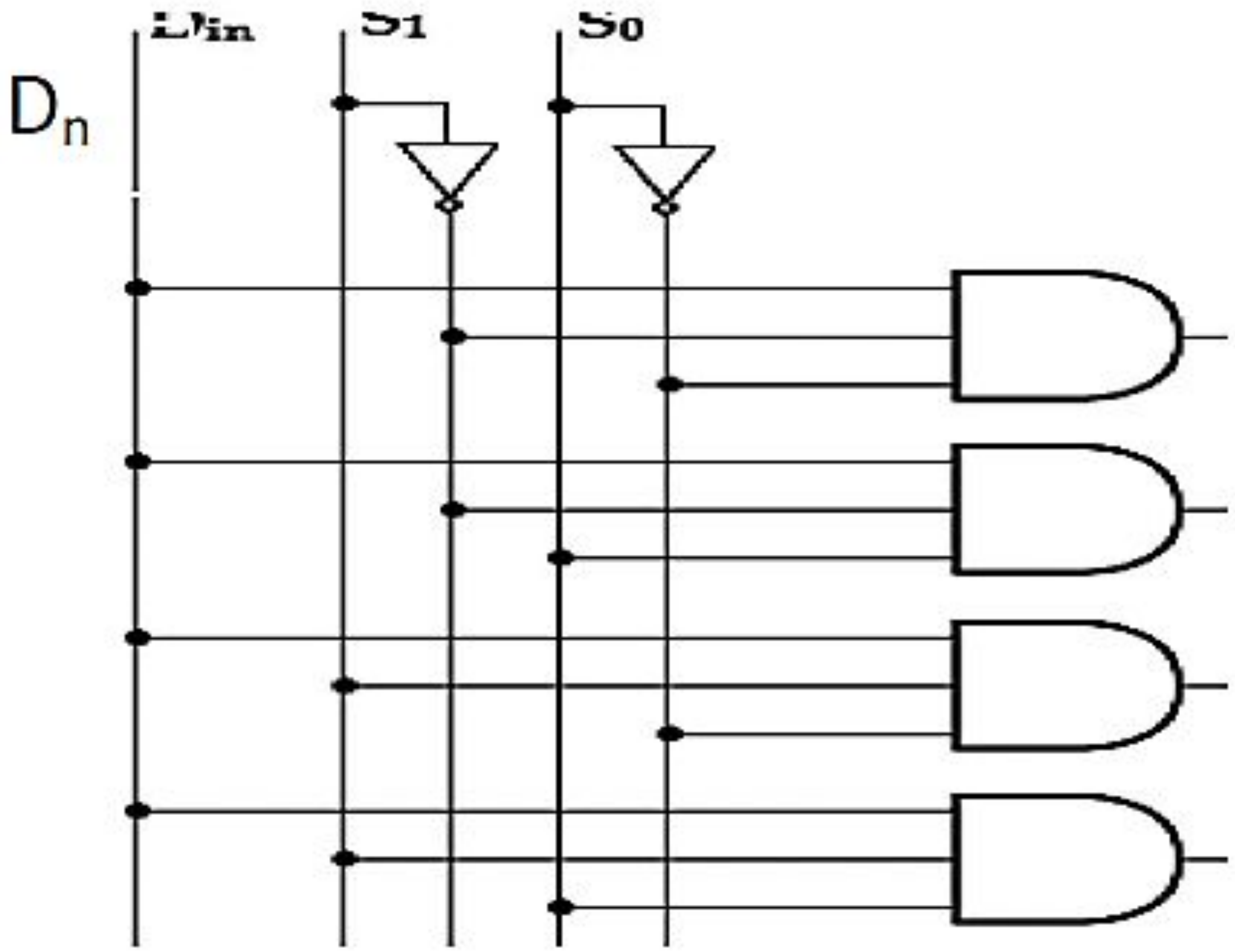
Demultiplexer –
1-to-4 line Demultiplexer



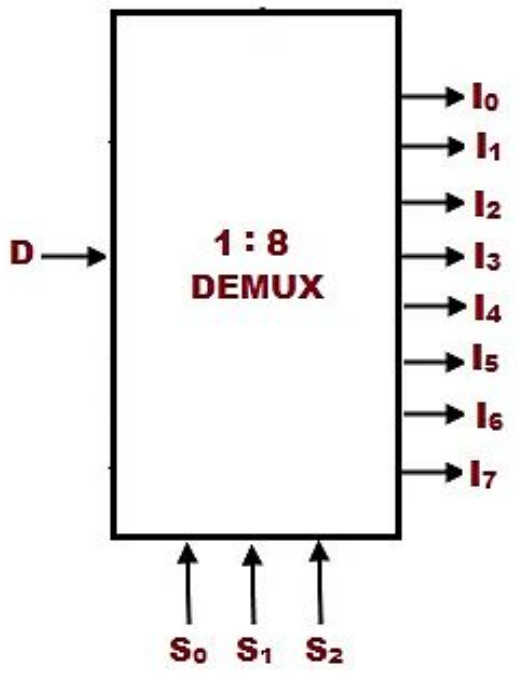
Enable	S_1	S_0	D_{in}	Y_0	Y_1	Y_2	Y_3
0	x	x	x	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	1	0	0
1	1	0	0	0	0	0	0
1	1	0	1	0	0	1	0
1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	1

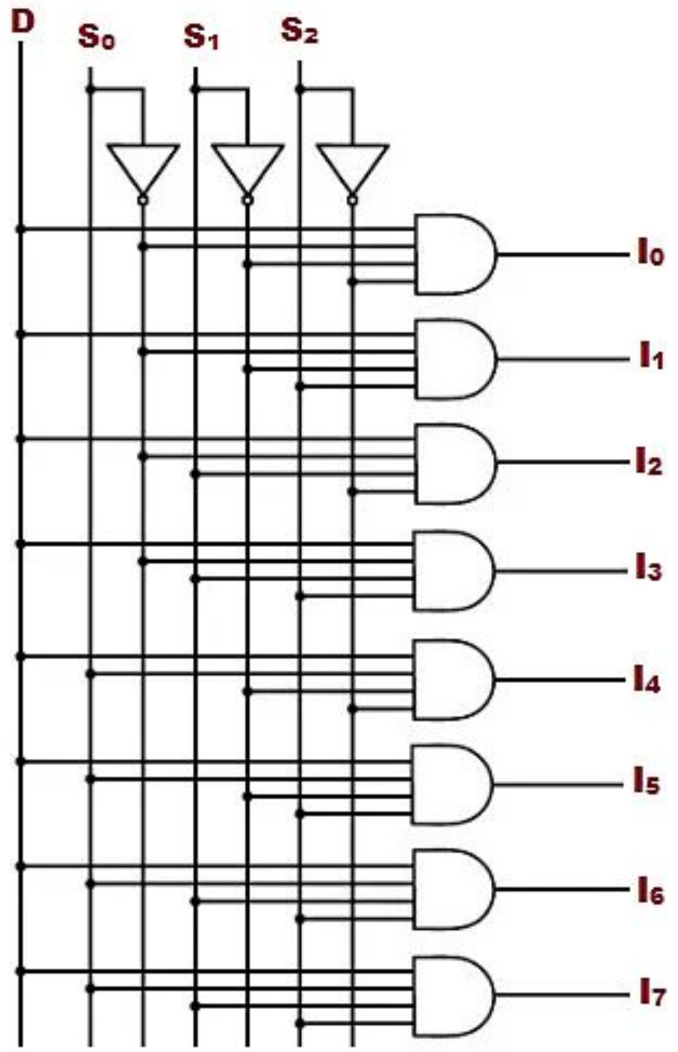
Demultiplexer –

1-to-4 line Demultiplexer

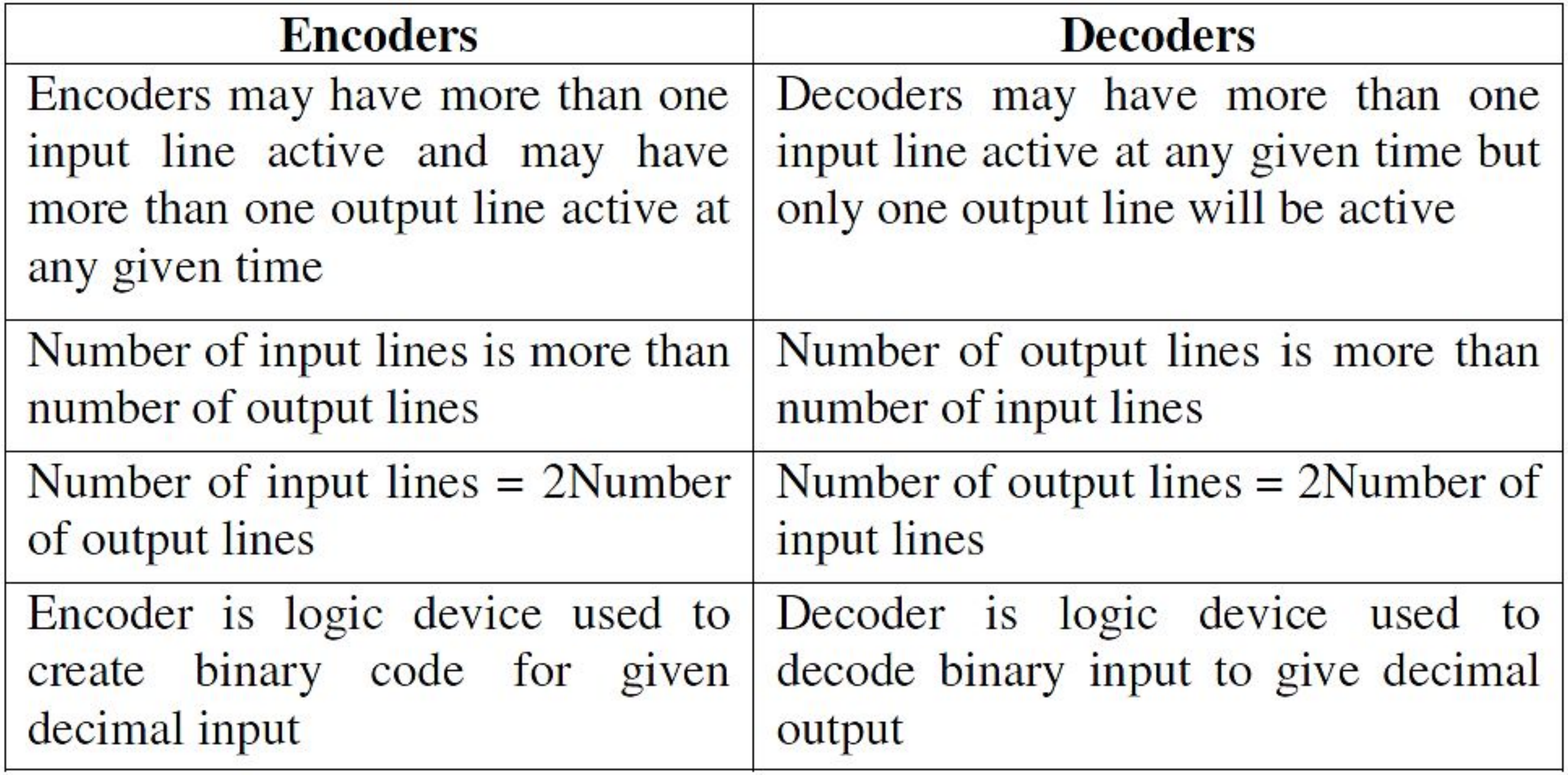


Demultiplexer –
1-to-8 line Demultiplexer





Encoder & Decoder

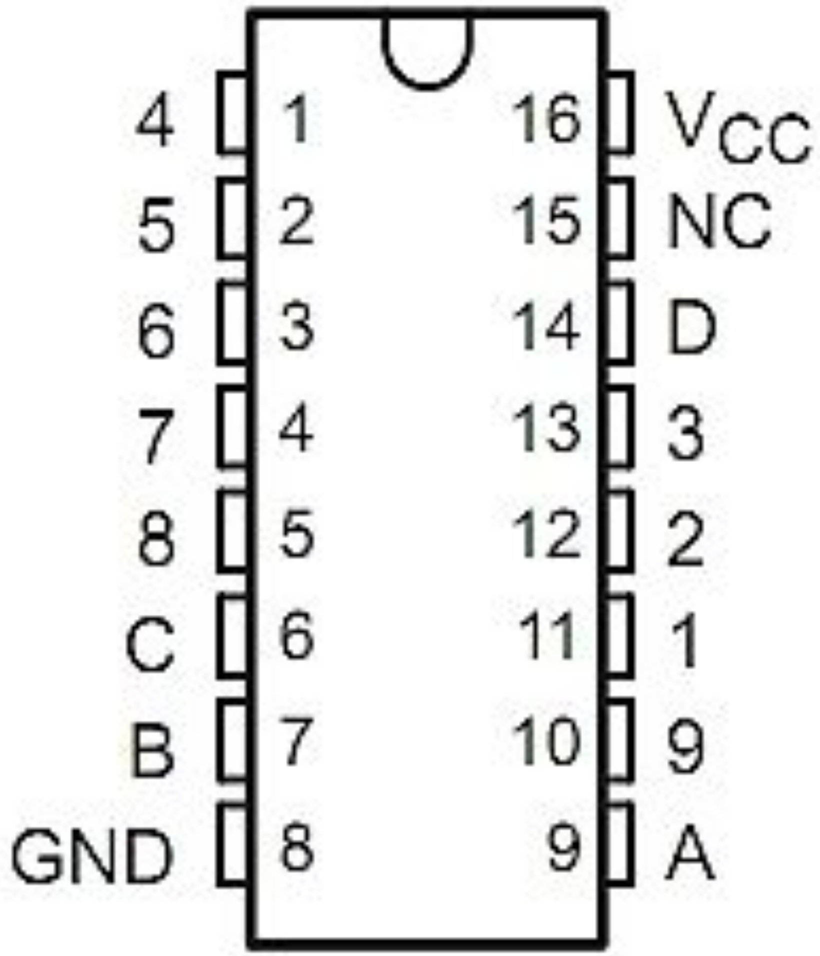


Encoders	Decoders
Encoders may have more than one input line active and may have more than one output line active at any given time	Decoders may have more than one input line active at any given time but only one output line will be active
Number of input lines is more than number of output lines	Number of output lines is more than number of input lines
Number of input lines = 2Number of output lines	Number of output lines = 2Number of input lines
Encoder is logic device used to create binary code for given decimal input	Decoder is logic device used to decode binary input to give decimal output

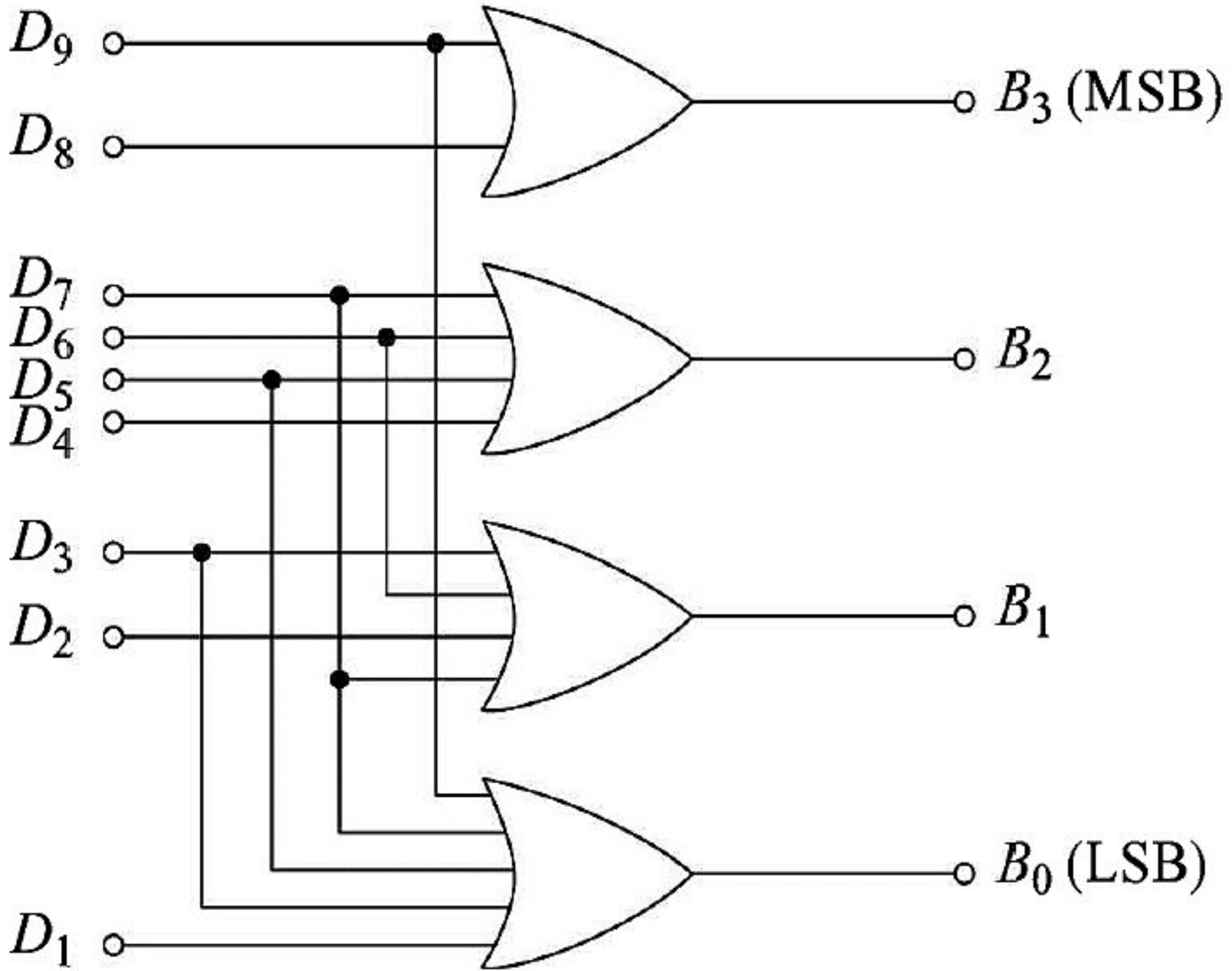
Decimal to BCD Encoder

Input	Output			
Decimal digit	B_3	B_2	B_1	B_0
D_0	0	0	0	0
D_1	0	0	0	1
D_2	0	0	1	0
D_3	0	0	1	1
D_4	0	1	0	0
D_5	0	1	0	1
D_6	0	1	1	0
D_7	0	1	1	1
D_8	1	0	0	0
D_9	1	0	0	1

Decimal to BCD Encoder – IC 7447



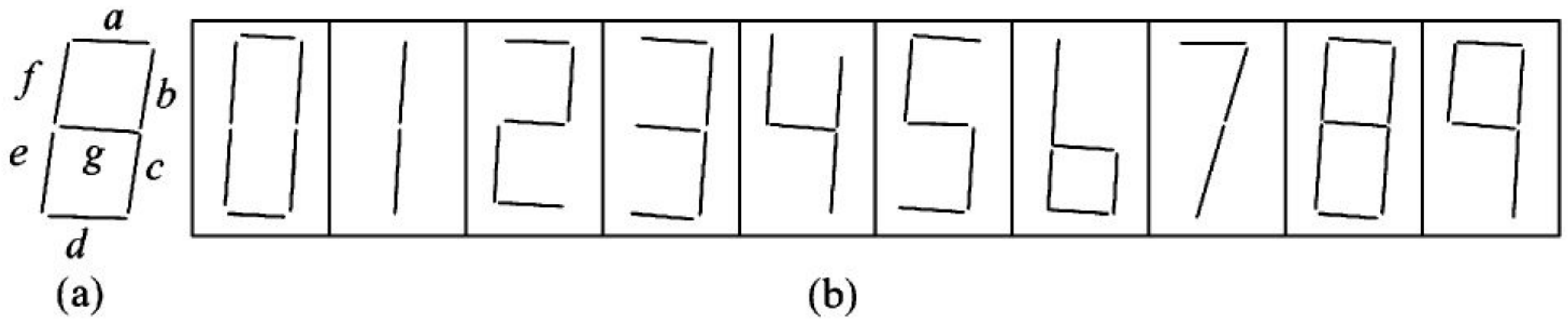
Decimal to BCD Encoder

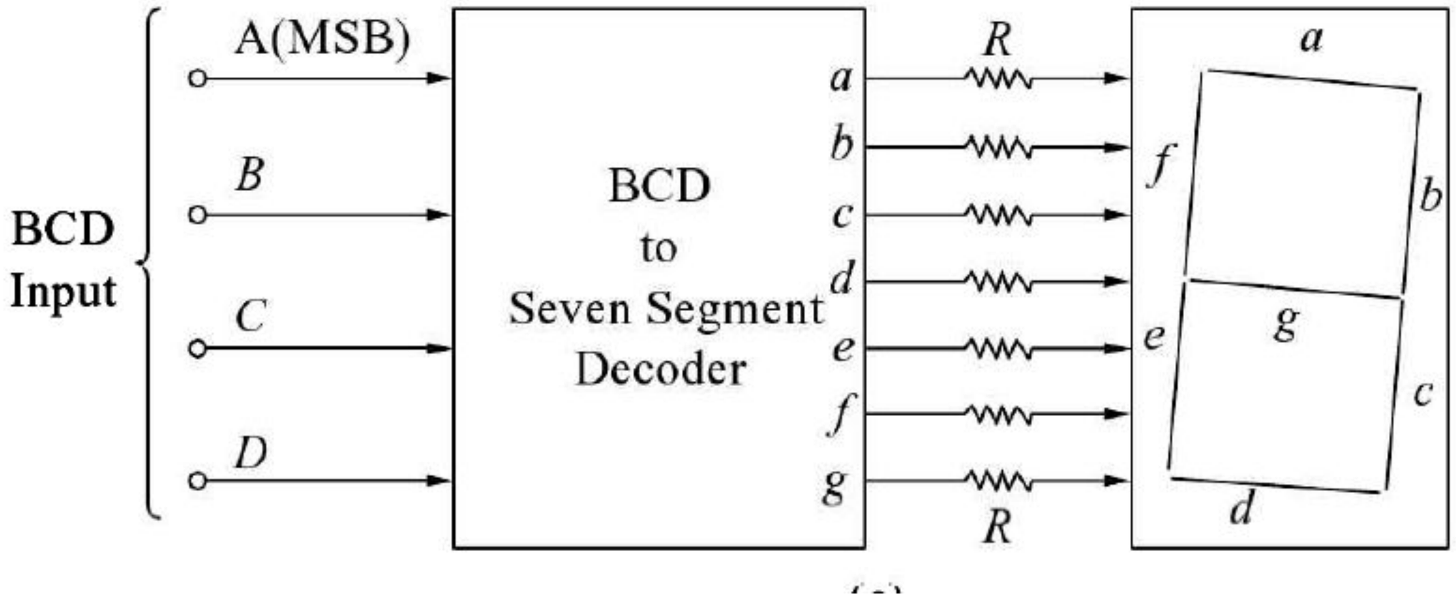


BCD to 7 – segment Decoder

- A digital display that consists of seven LED segments is commonly used to display decimal numerals in digital systems. Most familiar examples are electronic calculators and watches where one 7-segment display device is used for displaying one numeral 0 through 9.
- For using this display device, the data has to be converted from some binary code to the code required for the display. Usually, the binary code used is natural BCD.

BCD to 7 – segment Decoder

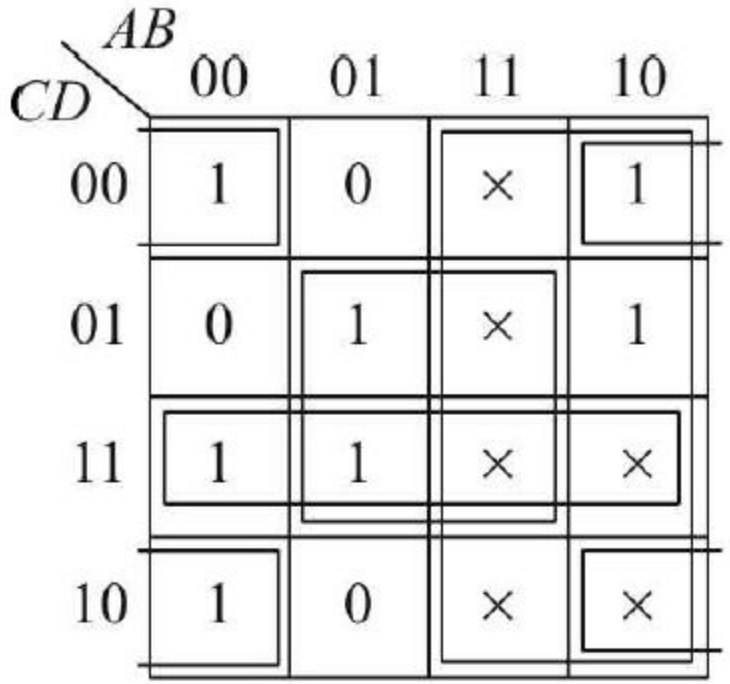




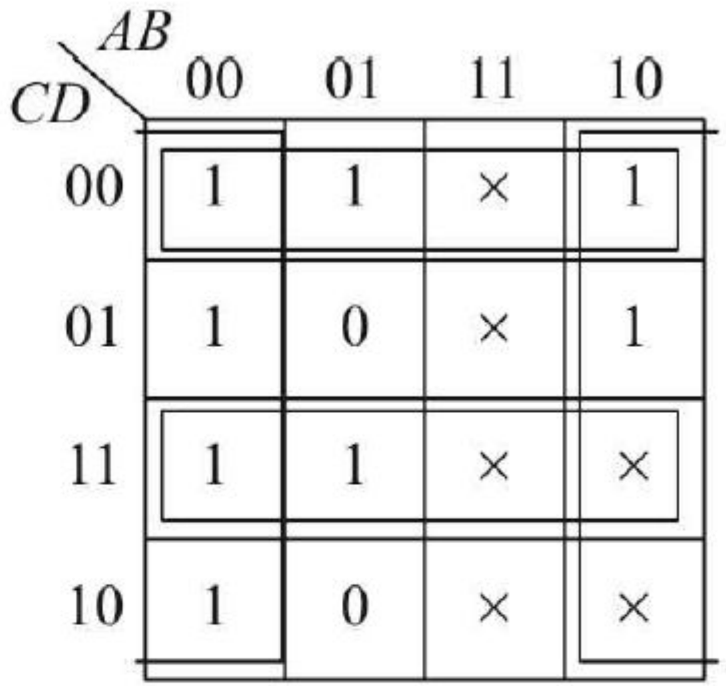
BCD to 7 – segment Decoder – Truth Table

Decimal digit displayed	Inputs				Outputs						
	A	*B*	*C*	*D*	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	0	0	1	1

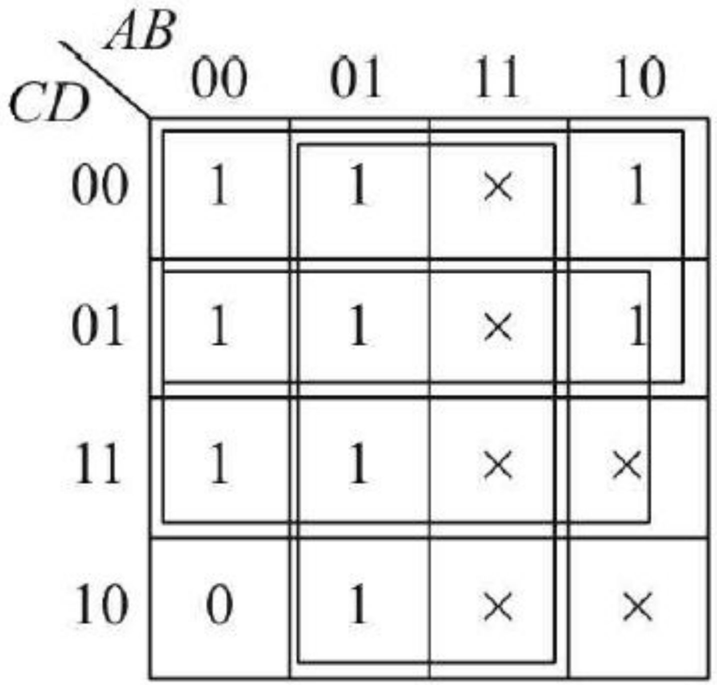
BCD to 7 – segment Decoder – K-map



CD \ AB	00	01	11	10
00	1	0	×	1
01	0	1	×	1
11	1	1	×	×
10	1	0	×	×

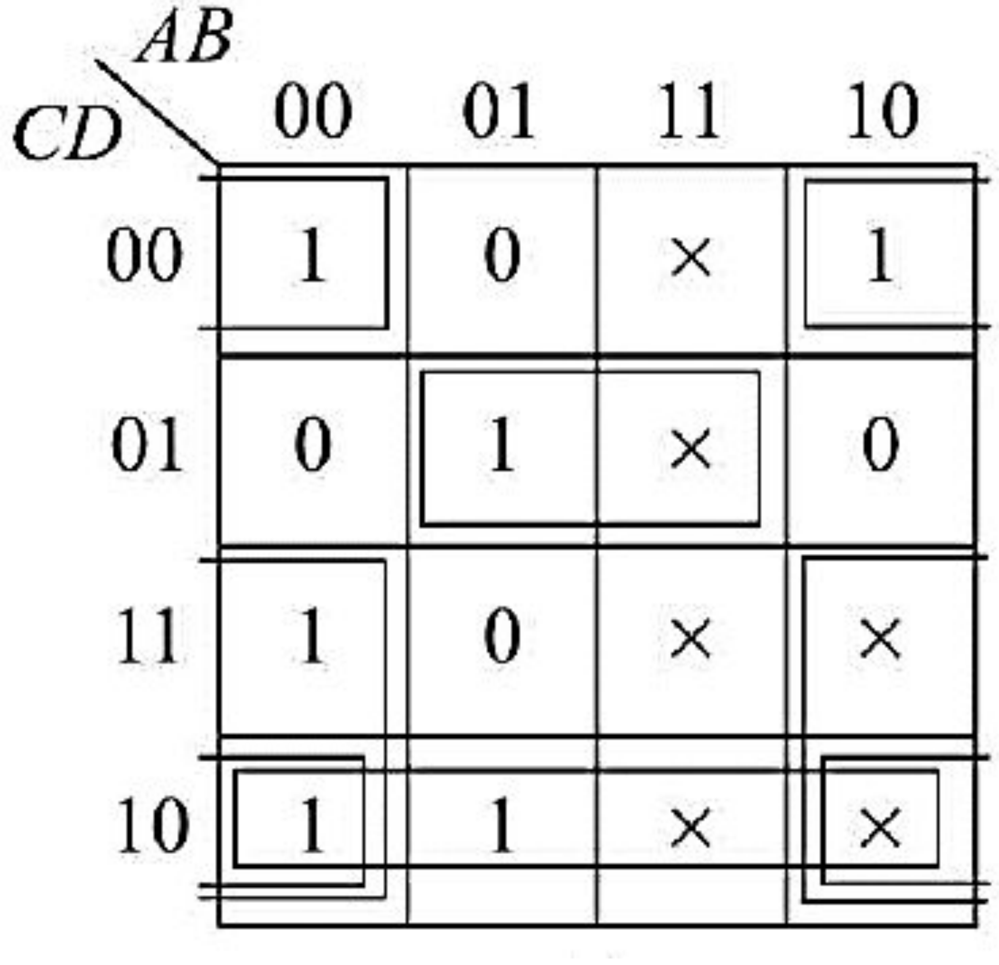


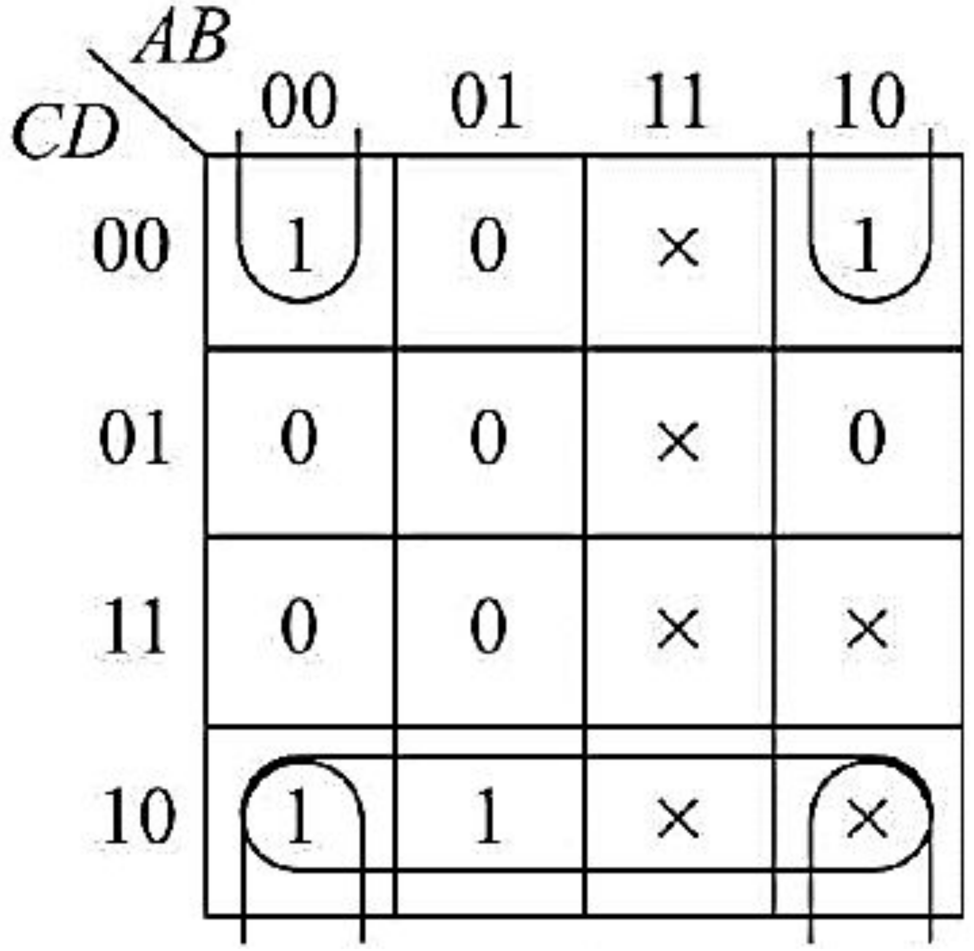
CD \ AB	00	01	11	10
00	1	1	×	1
01	1	0	×	1
11	1	1	×	×
10	1	0	×	×



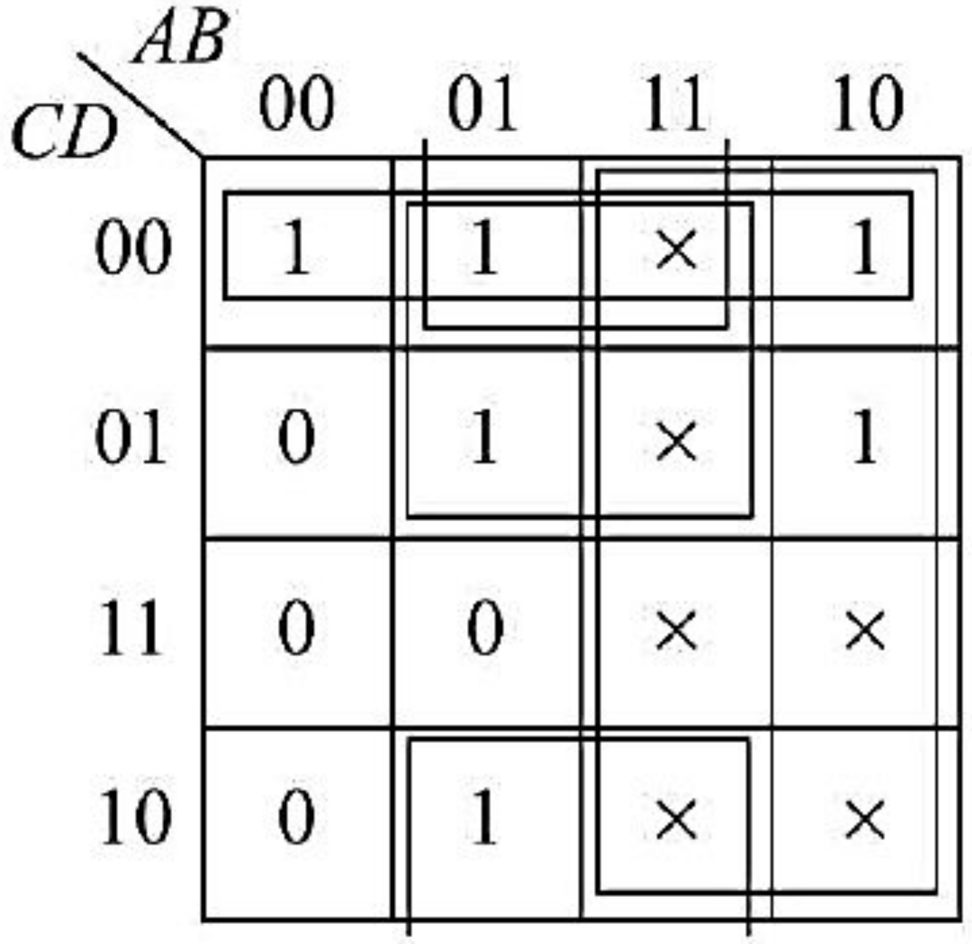
CD \ AB	00	01	11	10
00	1	1	×	1
01	1	1	×	1
11	1	1	×	×
10	0	1	×	×

BCD to 7 – segment Decoder – K-map

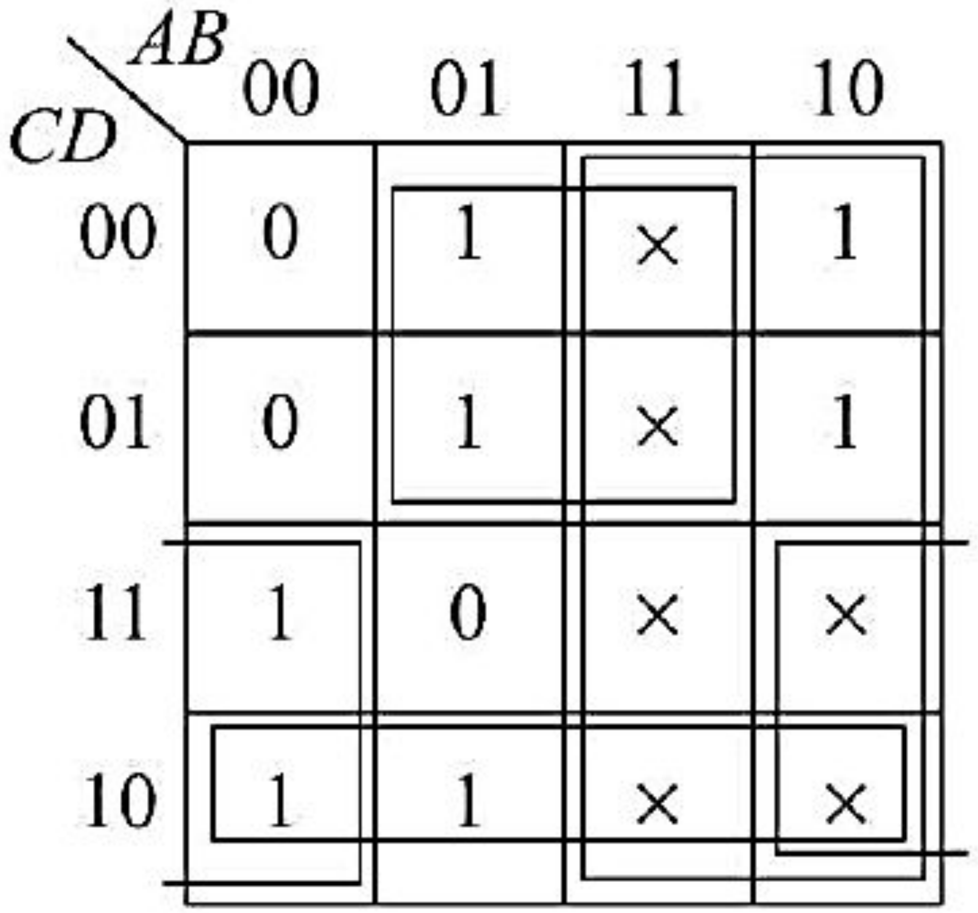




BCD to 7 – segment Decoder – K-map



CD \ AB	00	01	11	10
00	1	1	×	1
01	0	1	×	1
11	0	0	×	×
10	0	1	×	×



CD \ AB	00	01	11	10
00	0	1	×	1
01	0	1	×	1
11	1	0	×	×
10	1	1	×	×

BCD to 7 – segment Decoder – K-map

$$a = \overline{B}\,\overline{D} + BD + CD + A$$

$$b = \overline{B} + \overline{C}\,\overline{D} + CD$$

$$c = B + \overline{C} + D = \overline{\overline{B}C\overline{D}}$$

$$d = \overline{B}\,\overline{D} + C\overline{D} + \overline{B}C + B\overline{C}D$$

$$e = \overline{B}\,\overline{D} + C\overline{D}$$

$$f = A + \overline{C}\,\overline{D} + B\overline{C} + B\overline{D}$$

$$g = A + B\overline{C} + \overline{B}C + C\overline{D}$$