Combinational Logic

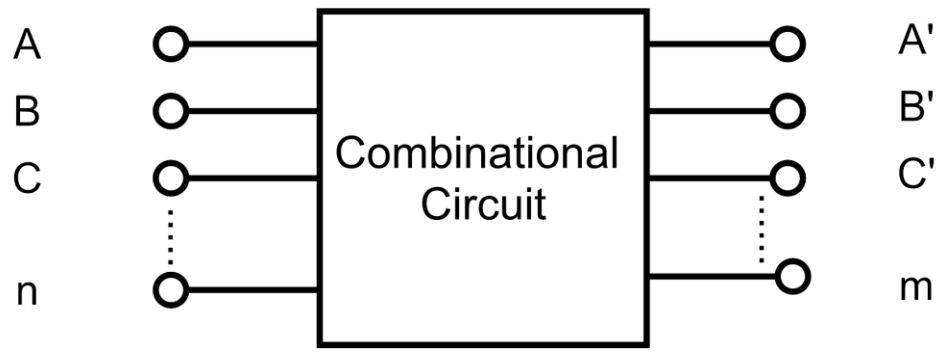
Logic Design of Digital Systems (300-1209) section 1

LECTURE 05

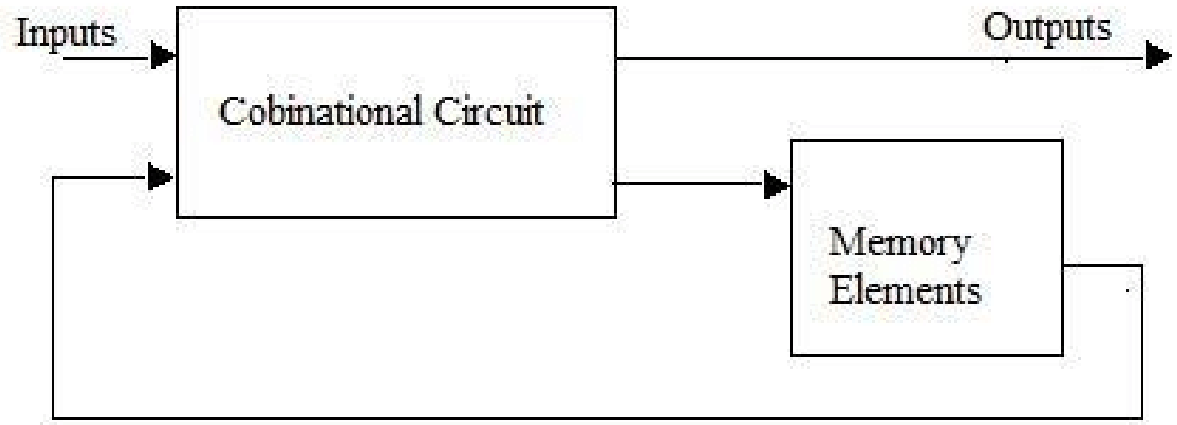
KRISADA PHROMSUTHIRAK

Logic Circuits for Digital Systems

Logic circuits for digital systems may be combinational or sequential. A combinational circuit consists of logic gates whose outputs at any time are determined directly from the present combination of inputs without regard to previous inputs.

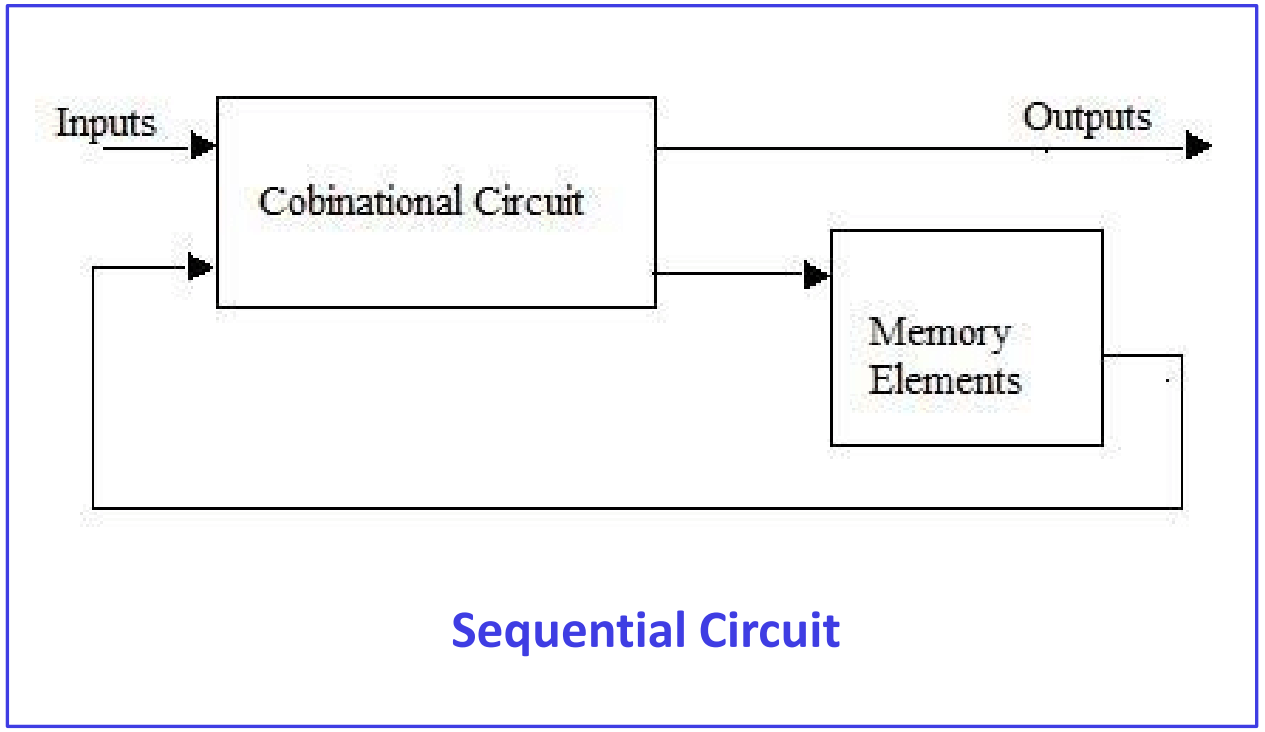


Combinational Circuit



Sequential Circuit

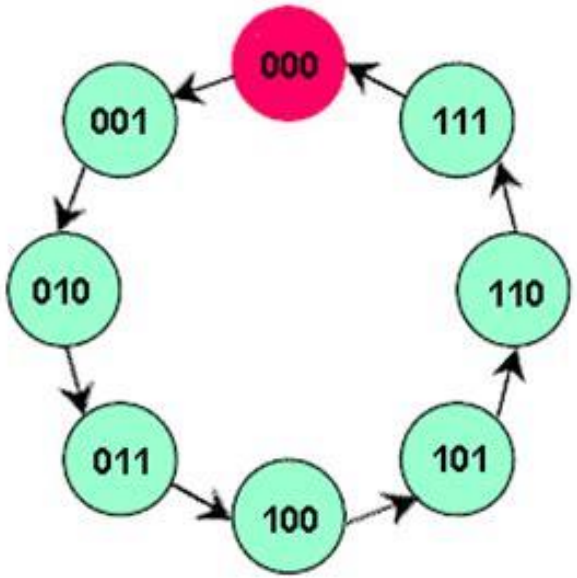
Logic Circuits for Digital Systems



Sequential Circuit

We'll learn this after the combinational logic chapter ends :)

Example

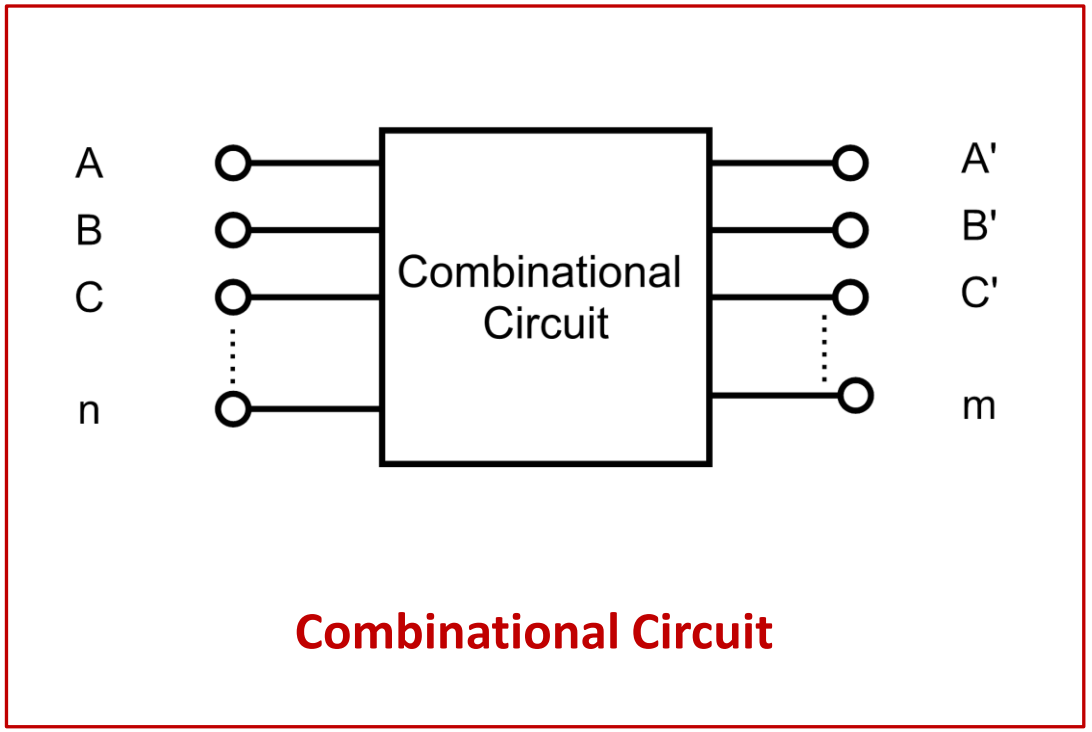


0

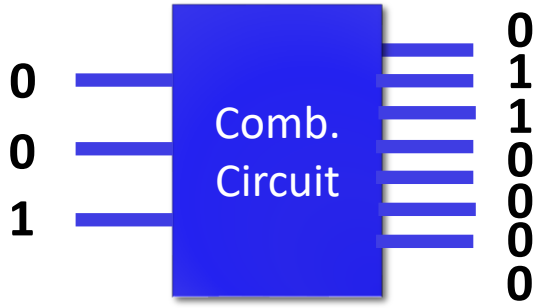
3 bits counter

These types of circuits have a memory unit to store the past output

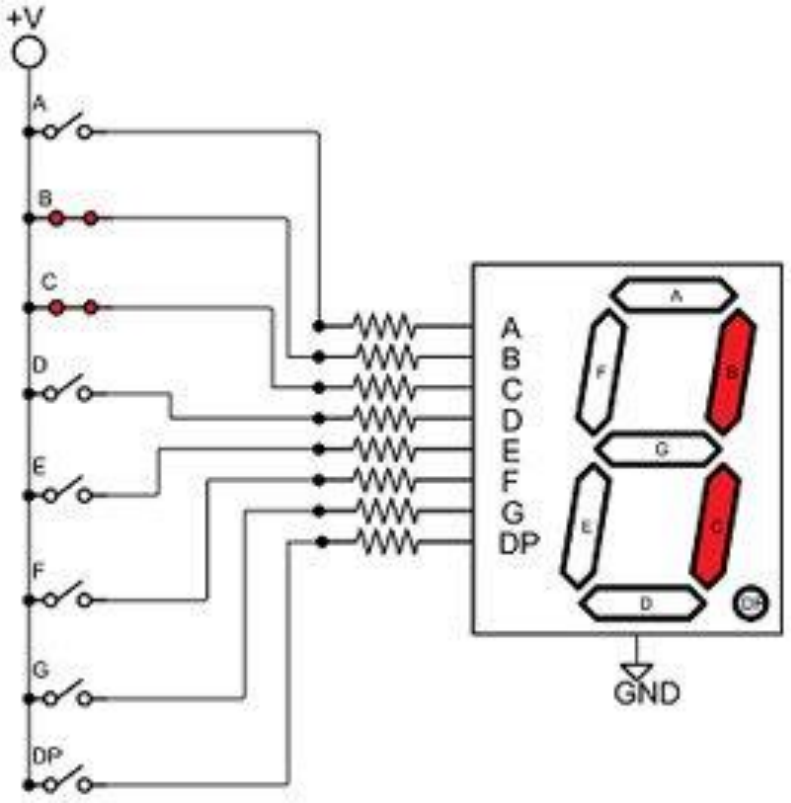
Logic Circuits for Digital Systems



Combinational Circuit



Example

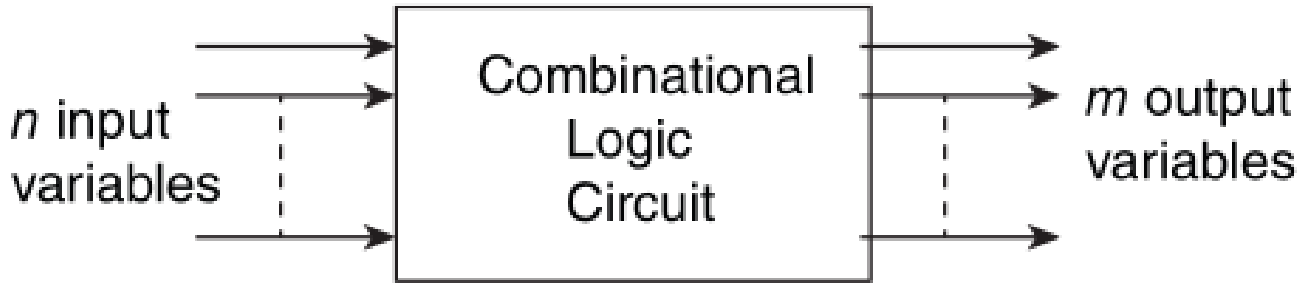


7-Segment Display

Combinational Circuit

Design Procedure

2^n

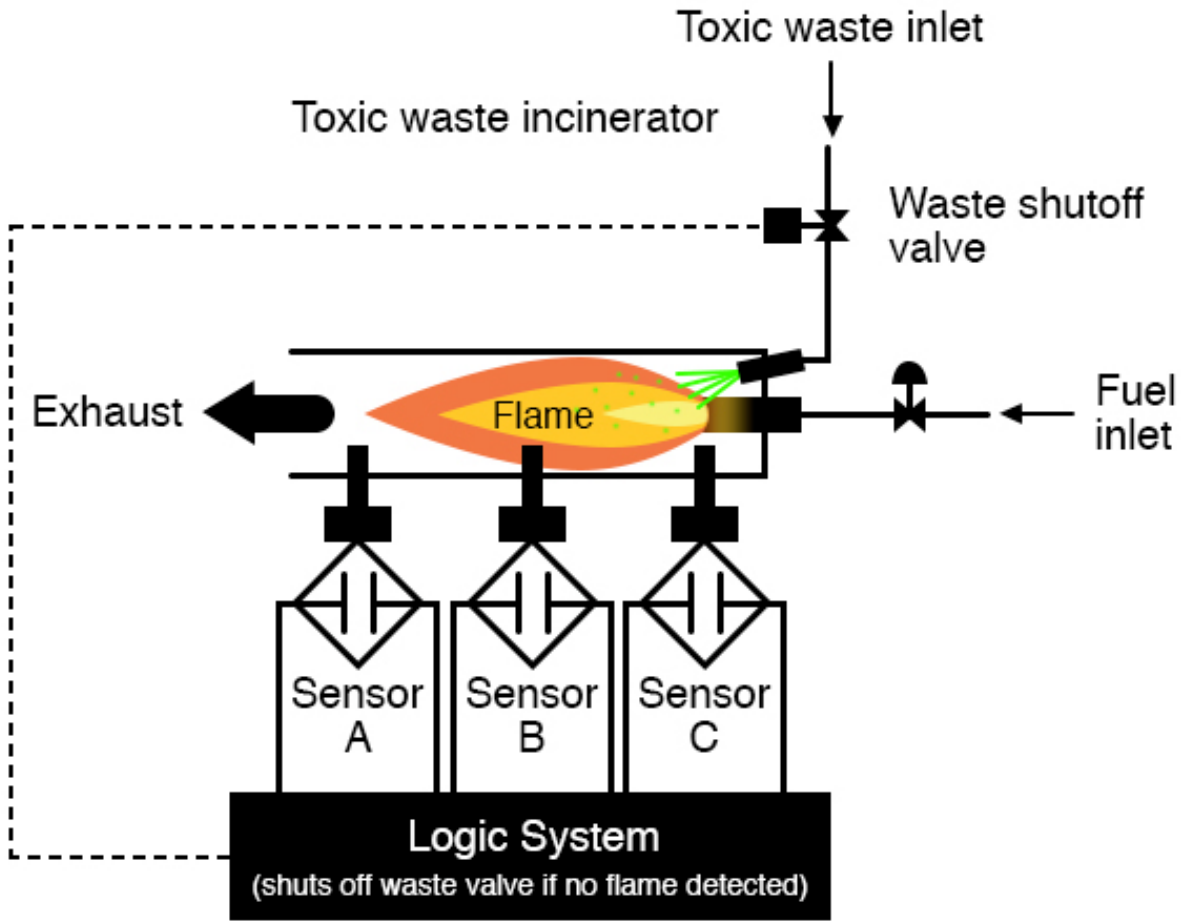


2^m

1. The problem is stated.
2. The number of available input variables and required output variables is determined.
3. The input and output variables are assigned letter symbols.
4. The truth table that defines the required relationships between inputs and outputs is derived.
5. The simplified Boolean function for each output is obtained.
6. The logic diagram is drawn.

Combinational Circuit

Design Procedure



Example

Sensor Inputs

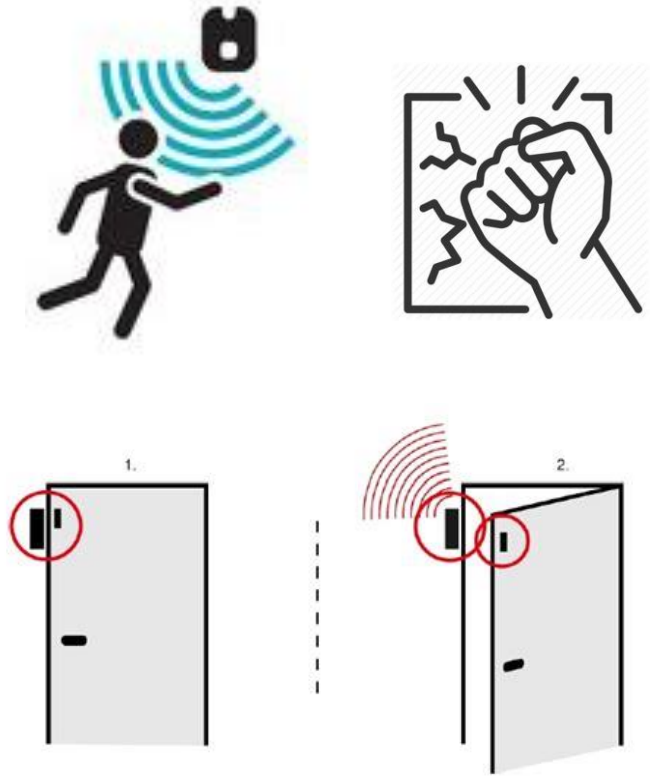
A	B	C	Output
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Output = 0 (close valve)

Output = 1 (open valve)

Combinational Circuit

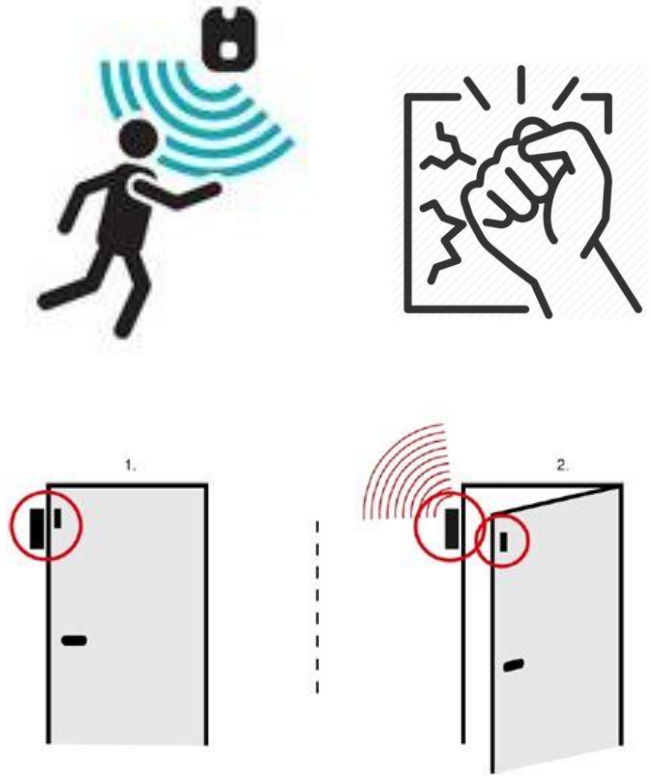
Example: Sum of Product



A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Combinational Circuit

Example: Product of Sum



A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Combinational Logic: Adders

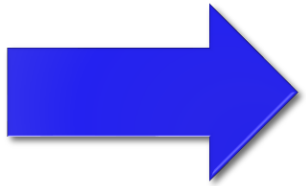
Digital computers perform a variety of information-processing tasks. Among the basic functions encountered are the various arithmetic operations. The most basic arithmetic operation, no doubt, is the addition of two binary digits.

Binary Addition

Combinational Logic: Half-Adder

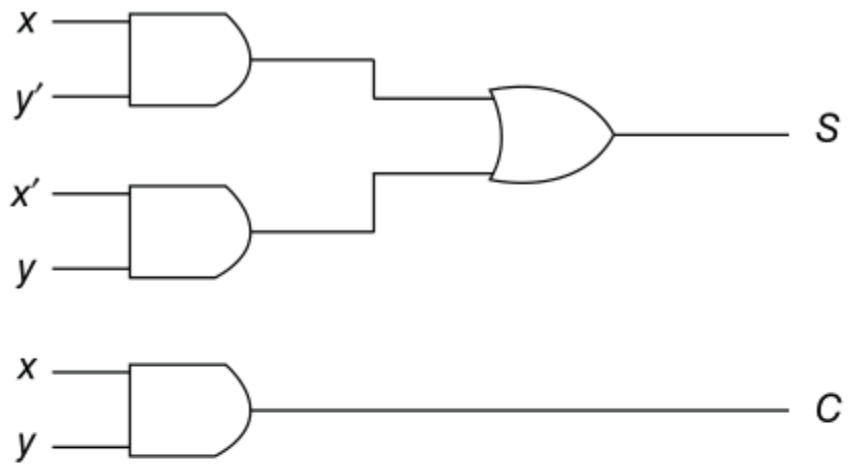
From the verbal explanation of a half-adder, we find that this circuit needs **two binary inputs** and **two binary outputs**. The input variables designate the **augend** and **addend bits**; the output variables produce the sum and carry. It is necessary to specify two output variables because the result may consist of two binary digits. We arbitrarily assign **symbols x and y to the two inputs** and **S (for sum) and C (for carry) to the outputs**.

X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

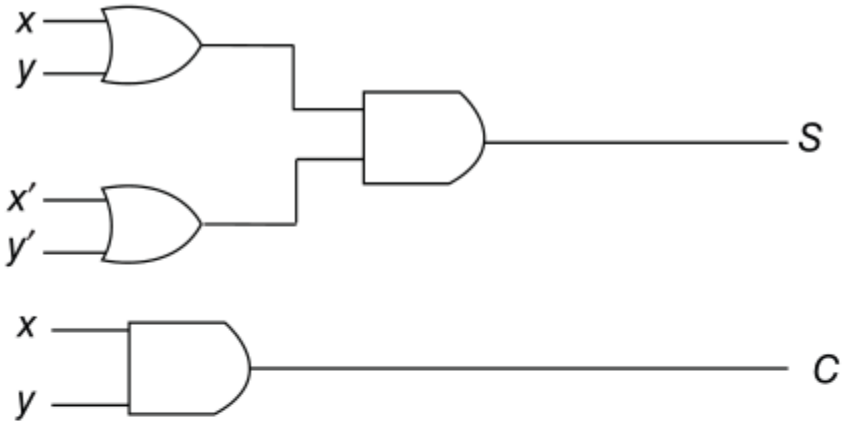


Boolean function

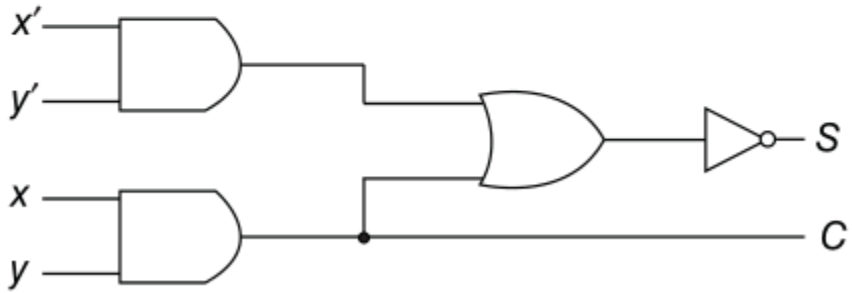
Combinational Logic: Half-Adder



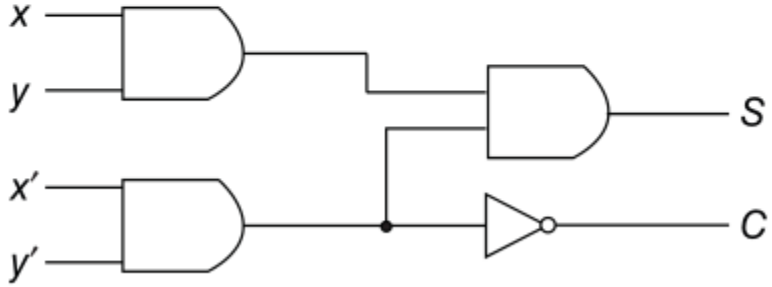
(a) $S = xy' + x'y$
$C = xy$



(b) $S = (x + y)(x' + y')$
$C = x\,y$

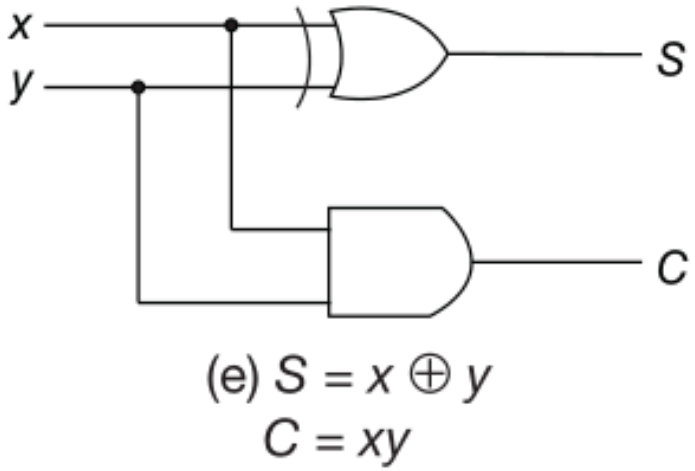


(c) $S = (C + x'y')'$
$C = xy$



(d) $S = (x + y)(x' + y')$
$C = (x' + y')'$

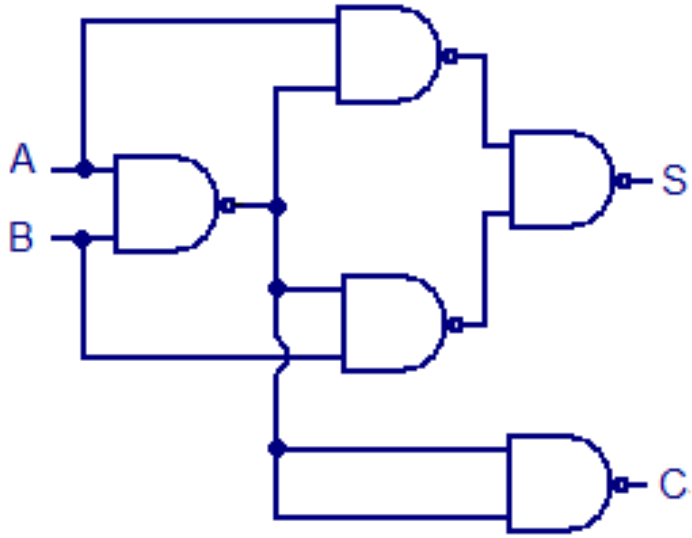
Combinational Logic: Half-Adder



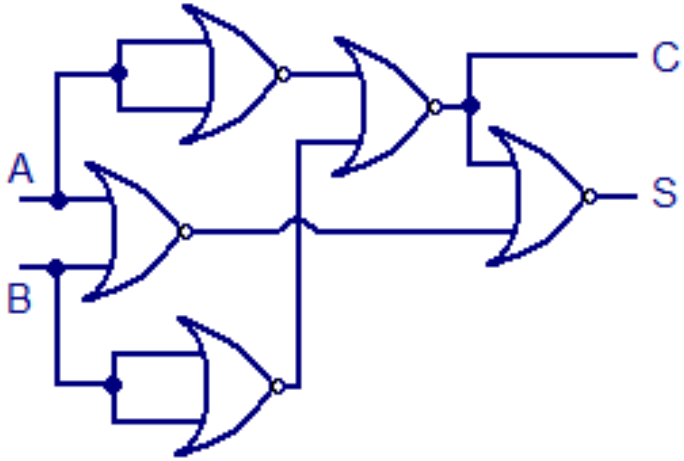
(e) $S = x \oplus y$

$C = xy$

Simplest Half-Adder



Half adder using NAND logic



Half adder using NOR logic

Combinational Logic: Full-Adder

A full-adder is a combinational circuit that forms the arithmetic sum of three input bits. It consists of **three inputs** and **two outputs**. Two of the input variables, denoted by x and y represent the two significant bits to be added. The **third input**, z **represents the carry from the previous lower significant position**.

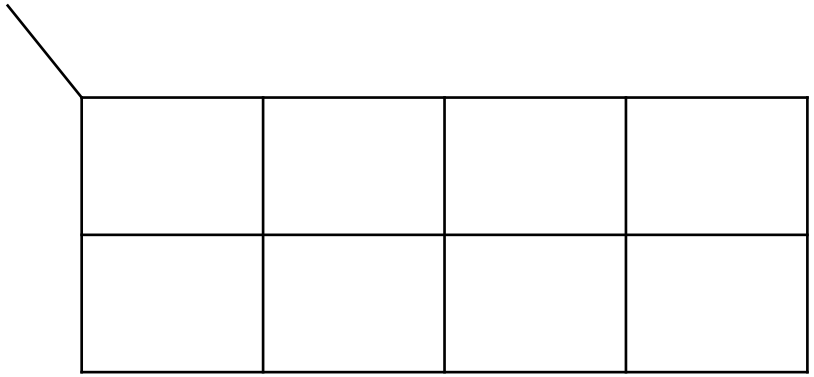
X	Y	Z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

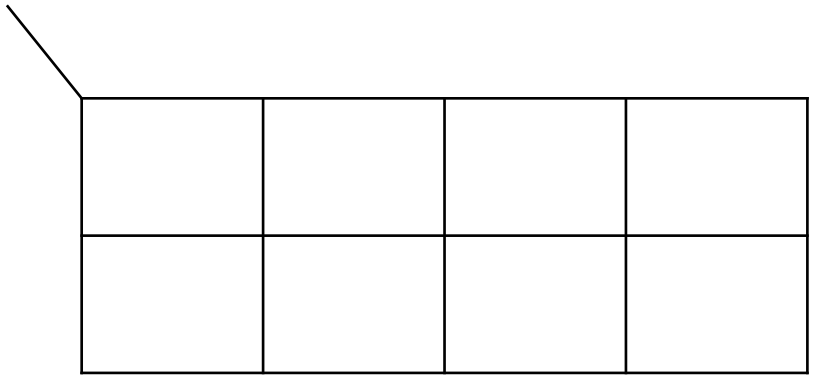
Boolean function

Sum of minterms

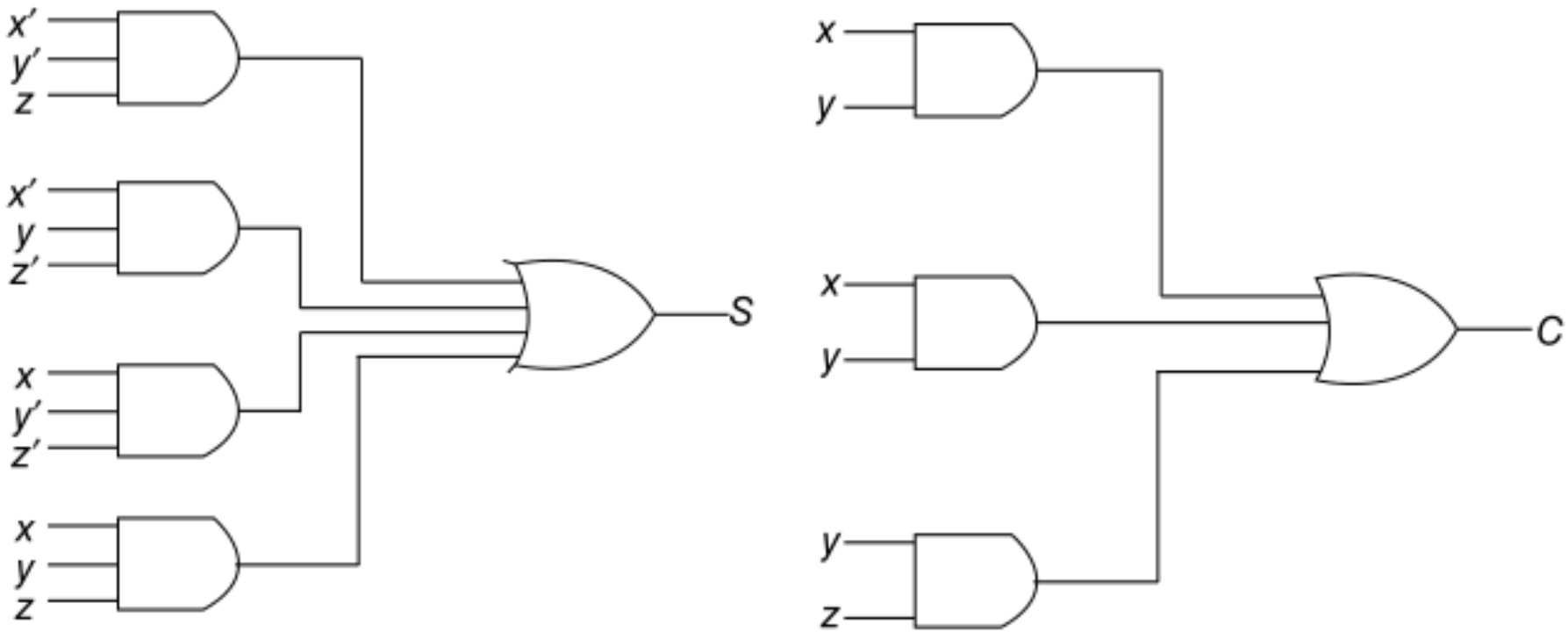
Combinational Logic: Full-Adder

X	Y	Z	C	S
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



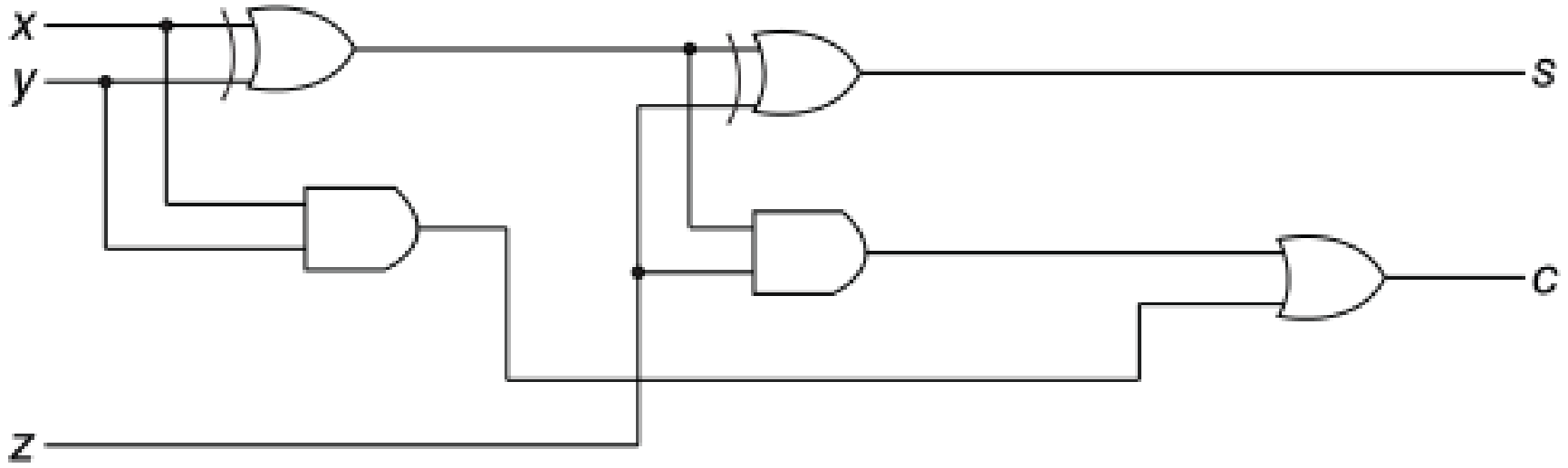


Combinational Logic: Full-Adder



Implementation of full-adder in sum of products

Combinational Logic: Full-Adder



Implementation of full-adder with two half-adders and an OR gate

Combinational Logic: Subtractors

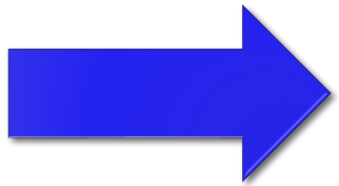
The subtraction of two binary numbers may be accomplished by **taking the complement of the subtrahend and adding it to the minuend**. By this method, the subtraction operation becomes an addition operation requiring full-adders for its machine implementation.

If the minuend bit is smaller than the subtrahend bit, a 1 is borrowed from the next significant position. The fact that a 1 has been **borrowed** must be conveyed to the next higher pair of bits by means of a binary signal coming out (output) of a given stage and going into (input) the next higher stage.

Combinational Logic: Half-Subtractor

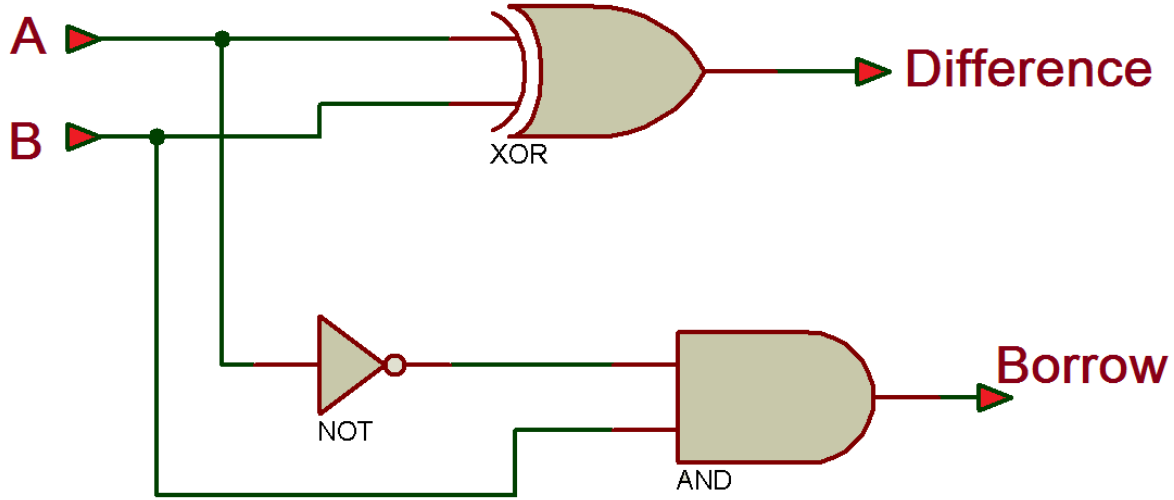
A half-subtractor is a combinational circuit that subtracts two bits and produces their difference. It also has an output to specify if a 1 has been borrowed. Designate the minuend bit by x and the subtrahend bit by y.

X	Y	B	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

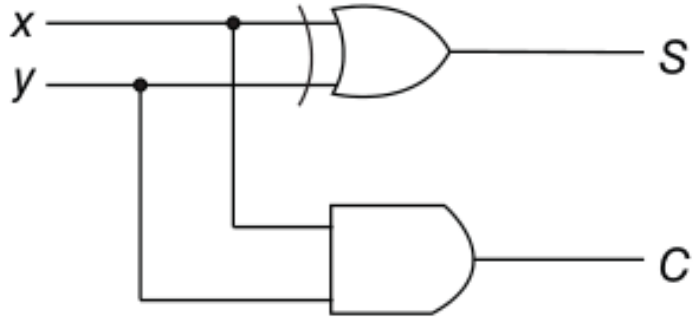


Boolean function

Combinational Logic: Half-Subtractor



Half-Subtractor



Half-Adder

Combinational Logic: Full-Subtractor

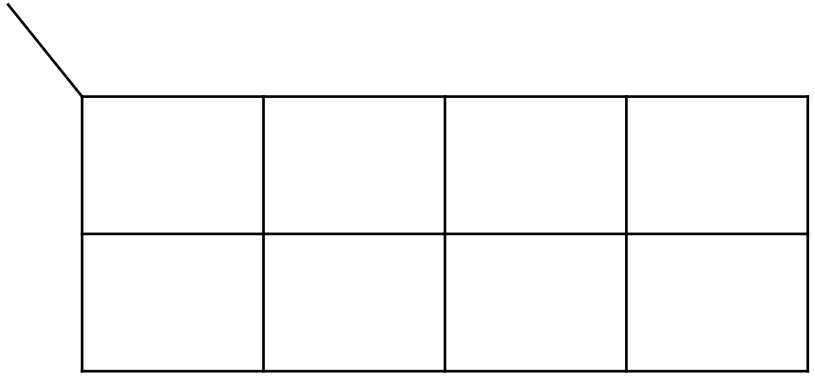
A full-subtractor is a combinational circuit that performs a subtraction between two bits, taking into account that a 1 may have been borrowed by a lower significant stage. This circuit has three inputs and two outputs. The three inputs, x, y, and z, denote the minuend, subtrahend, and previous borrow, respectively. The two outputs, D and B, represent the difference and output borrow, respectively.

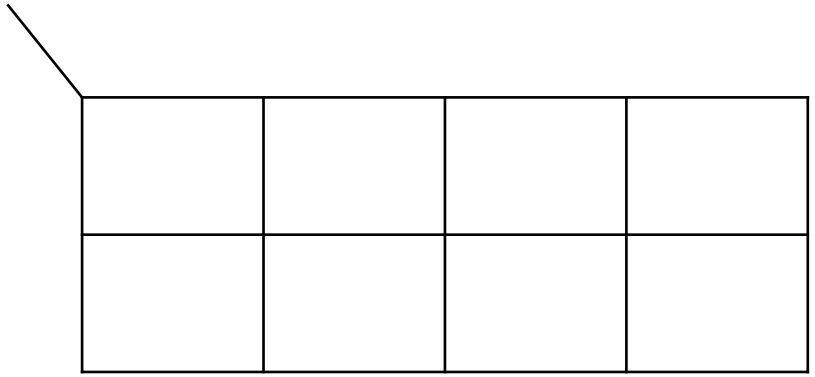
X	Y	Z	B	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Boolean function

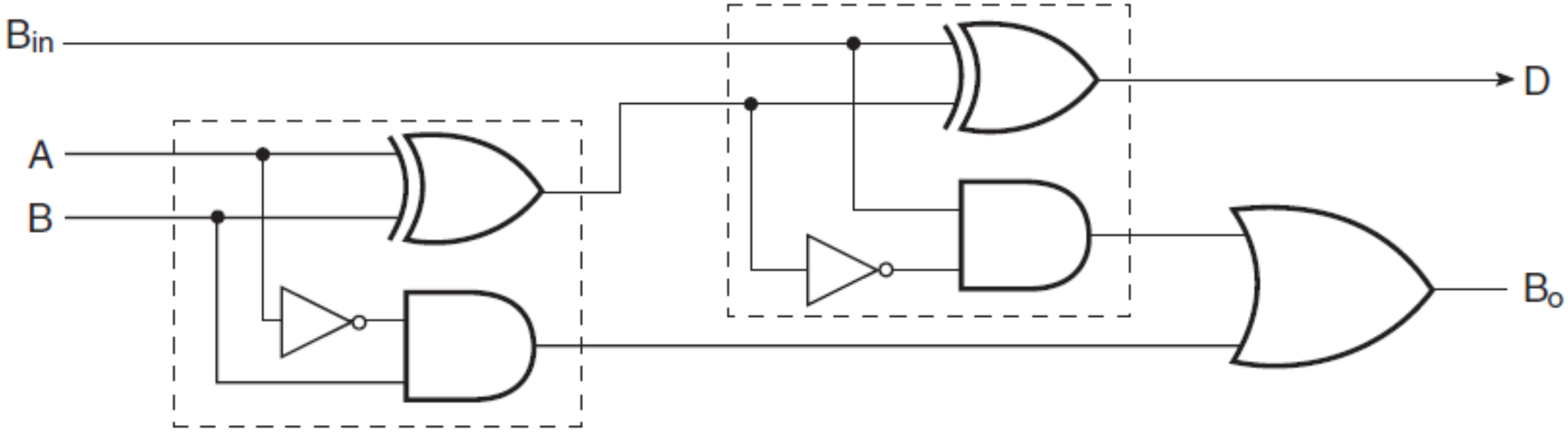
Combinational Logic: Full-Subtractor

X	Y	Z	B	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1





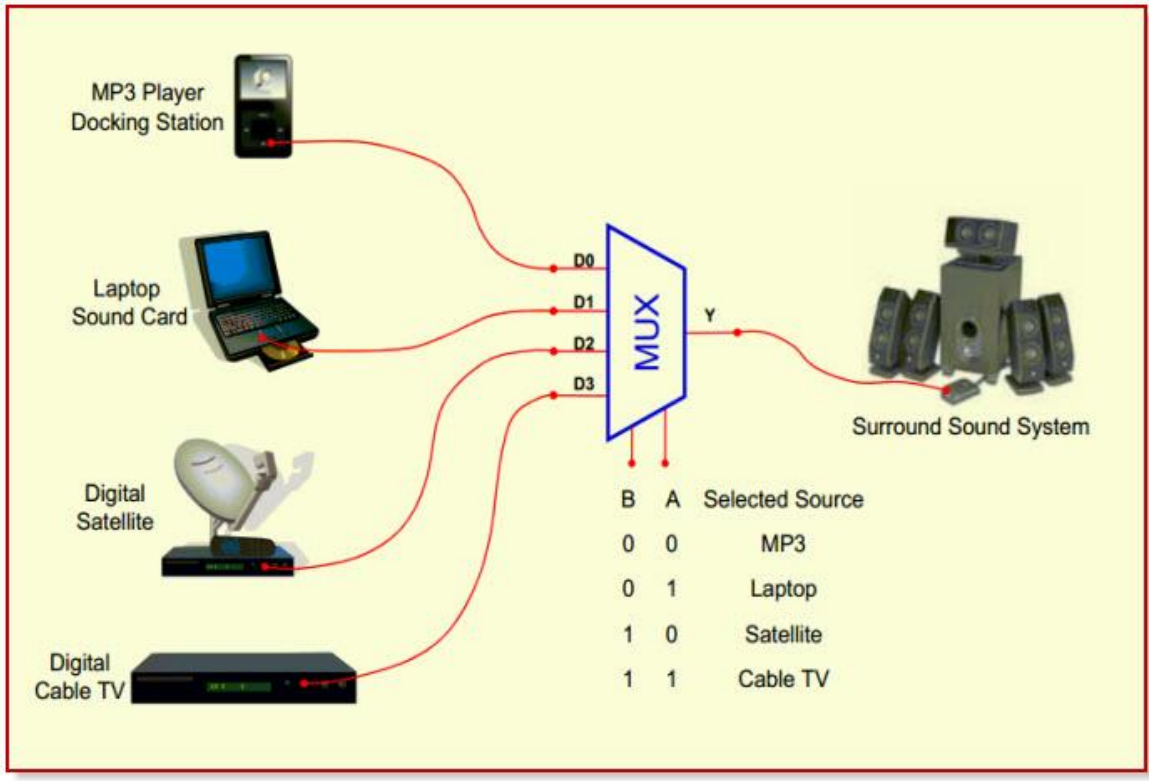
Combinational Logic: Full-Subtractor

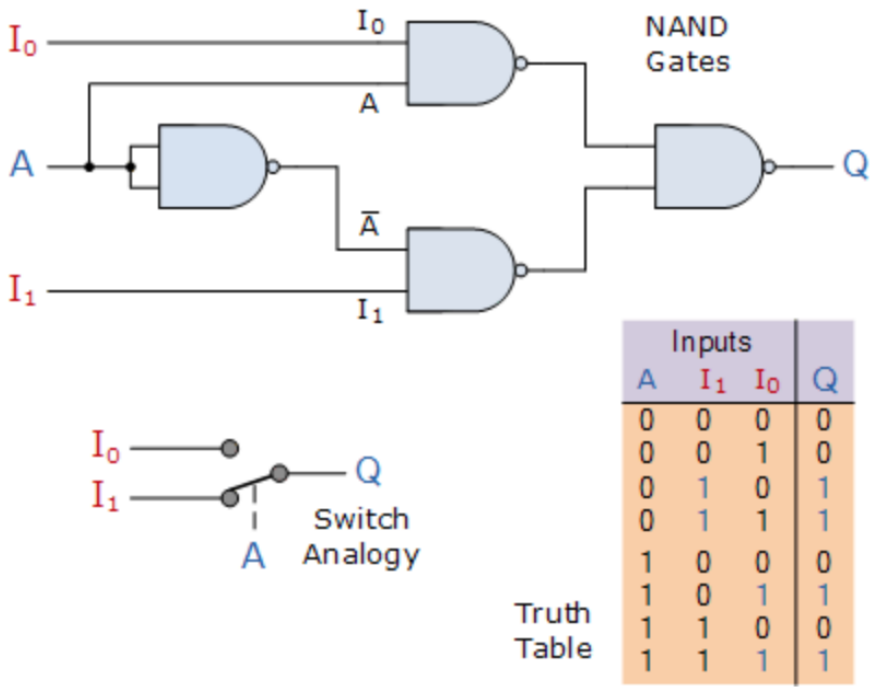


Implementation of full-subtractor with two half-subtractors and an OR gate

Combinational Logic: Code Conversion: [Encoder]

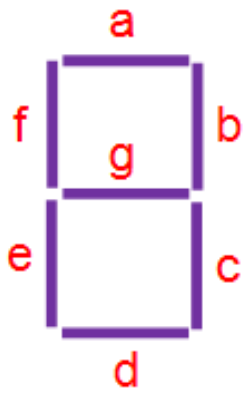
The availability of a large variety of codes for the same discrete elements of information results in the use of different codes by different digital systems. It is sometimes necessary to use the output of one system as the input to another.

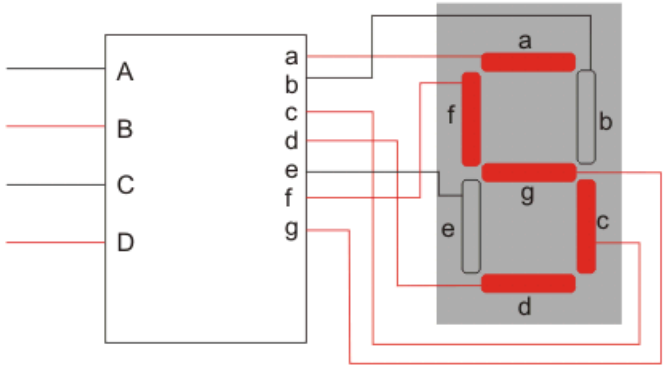


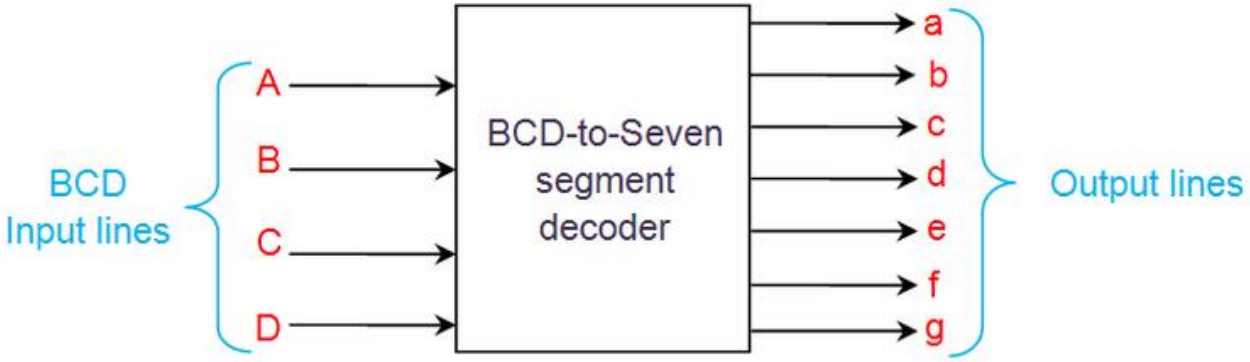


https://www.electrically4u.com/code-converter-types-truth-table-and-logic-circuits/

Combinational Logic: Code Conversion: [BCD to 7-Segment]







4-bits 7

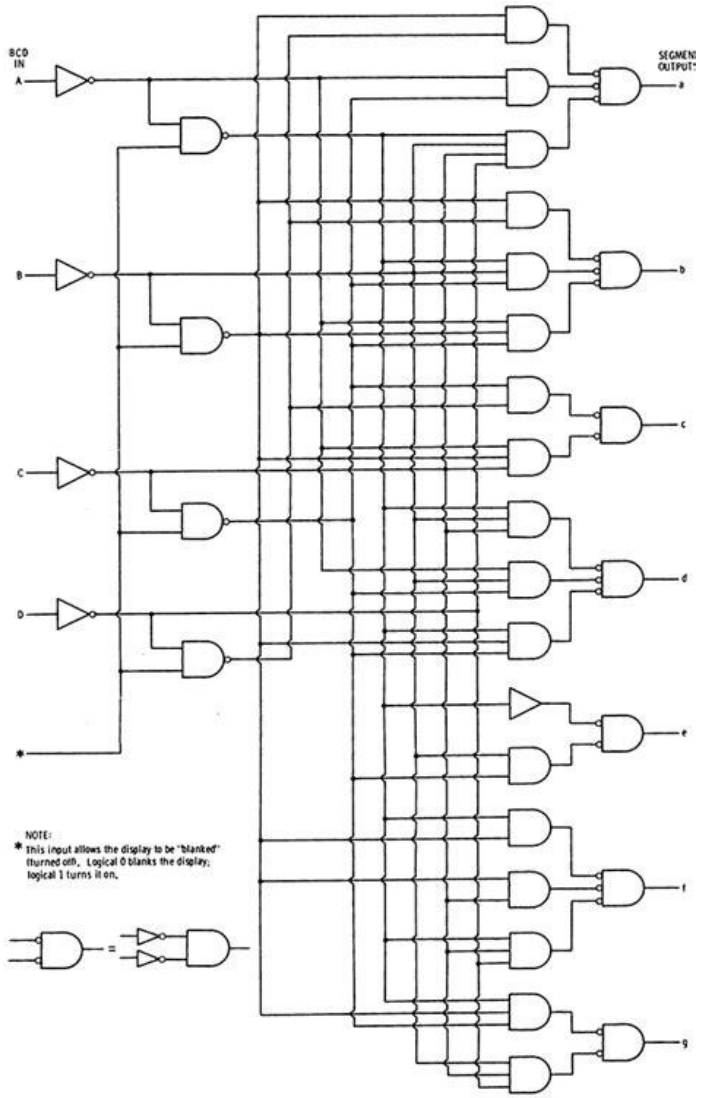
Decimal Digit	Input lines				Output lines							Display pattern
	A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	0	1	1	1	1	1	1	0	
1	0	0	0	1	0	1	1	0	0	0	0	
2	0	0	1	0	1	1	0	1	1	0	1	
3	0	0	1	1	1	1	1	1	0	0	1	
4	0	1	0	0	0	1	1	0	0	1	1	
5	0	1	0	1	1	0	1	1	0	1	1	
6	0	1	1	0	1	0	1	1	1	1	1	
7	0	1	1	1	1	1	1	0	0	0	0	
8	1	0	0	0	1	1	1	1	1	1	1	
9	1	0	0	1	1	1	1	1	0	1	1	

https://www.geeksforgeeks.org/bcd-to-7-segment-decoder/

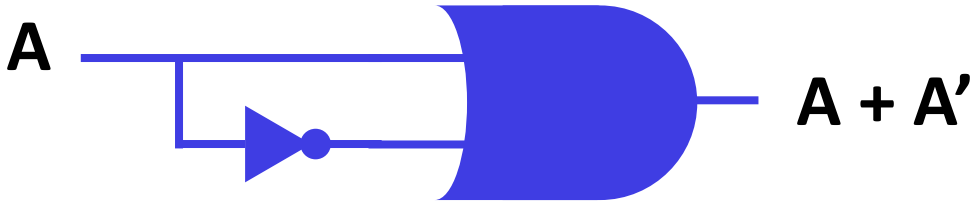
Combinational Logic: Code Conversion: [BCD to 7-Segment]

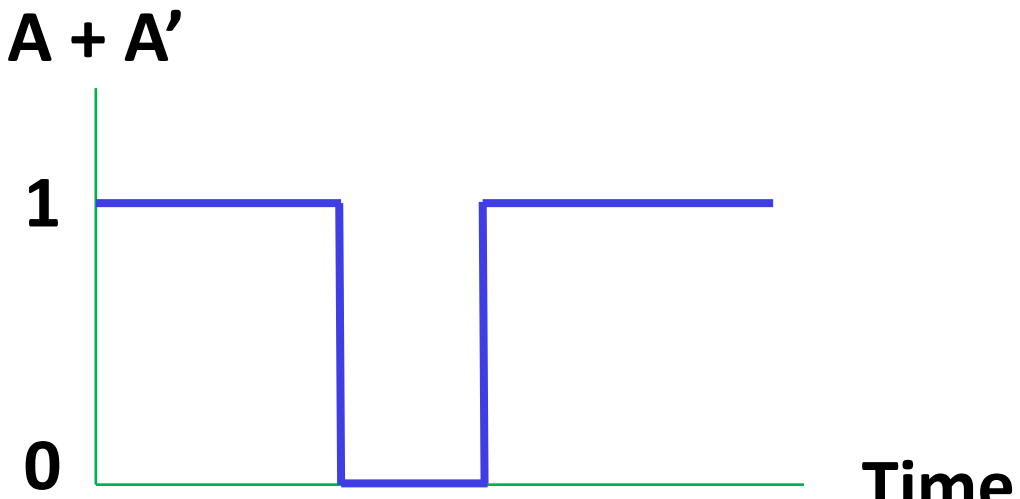
Decimal Digit	Input lines				Output lines							Display pattern
	A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	0	1	1	1	1	1	1	0	
1	0	0	0	1	0	1	1	0	0	0	0	
2	0	0	1	0	1	1	0	1	1	0	1	
3	0	0	1	1	1	1	1	1	0	0	1	
4	0	1	0	0	0	1	1	0	0	1	1	
5	0	1	0	1	1	0	1	1	0	1	1	
6	0	1	1	0	1	0	1	1	1	1	1	
7	0	1	1	1	1	1	1	0	0	0	0	
8	1	0	0	0	1	1	1	1	1	1	1	
9	1	0	0	1	1	1	1	1	0	1	1	

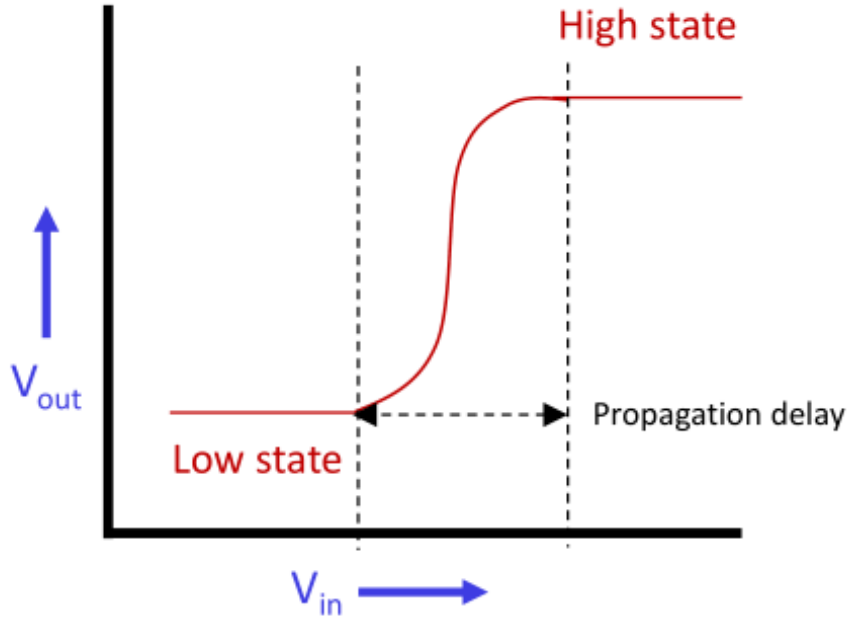
Question: What do we do with input 1110 ?



Hazards in Combinational Circuits (Problem in Practical)







Practical digital circuit

From lecture 1

Class work ออกแบบวงจร Combinational Logic จาก Boolean Function ต่อไปนี้

1. ให้นักศึกษาทำการหาตารางค่าความจริง (Truth Table) และลดรูปวงจร (Simplification) ที่กำหนดให้ต่อไปนี้โดยจัดให้อยู่ในรูปแบบ Sum of Product

$$f(A,B,C,D) = \sum(\ 1,3,7,11,15 \ )$$

with don't care condition

$$d(w,x,y,z) = \sum(\ 0,2,5 \ )$$

ผลลัพธ์

- ❑ Truth Table
- ❑ Boolean Function (Simplification)

หมายเหตุ

- ❑ ให้เขียนขั้นตอนและที่มาของผลลัพธ์ในแต่ละขั้นตอนให้ชัดเจน

Class work ออกแบบวงจร Combinational Logic จาก Boolean Function ต่อไปนี้

2. ให้นักศึกษาทำการหาตารางค่าความจริง (Truth Table) และลดรูปวงจร (Simplification) ที่กำหนดให้ต่อไปนี้โดยจัดให้อยู่ในรูปแบบ Sum of Product

$$f(A,B,C,D,E) = \sum(4,6,9,11,13,15,16,17,18,21,27,29,31)$$

with don't care condition

$$d(v,w,x,y,z) = \sum(0,2,25)$$

ผลลัพธ์

- ❑ Truth Table
- ❑ Boolean Function (Simplification)

หมายเหตุ

- ❑ ให้เขียนขั้นตอนและที่มาของผลลัพธ์ในแต่ละขั้นตอนให้ชัดเจน

Class work ออกแบบวงจร Combinational Logic จาก Boolean Function ต่อไปนี้

3. ให้นักศึกษาทำการหาตารางค่าความจริง (Truth Table) และลดรูปวงจร (Simplification) ที่กำหนดให้ต่อไปนี้โดยจัดให้อยู่ในรูปแบบ Product of Sum

$$f(A,B,C,D) = \sum(1,9,12,13,14)$$

with don't care condition

$$d(w,x,y,z) = \sum(3,6,15)$$

ผลลัพธ์

- ❑ Truth Table
- ❑ Boolean Function (Simplification)

หมายเหตุ

- ❑ ให้เขียนขั้นตอนและที่มาของผลลัพธ์ในแต่ละขั้นตอนให้ชัดเจน

Class work ออกแบบวงจร Combinational Logic จาก Boolean Function ต่อไปนี้

4. ให้นักศึกษาทำการหาตารางค่าความจริง (Truth Table) และลดรูปวงจร (Simplification) ที่กำหนดให้ต่อไปนี้โดยจัดให้อยู่ในรูปแบบ Product of Sum

$$f(A,B,C,D,E) = \sum(4,6,9,11,13,15,16,17,18,21,27,29,31)$$

with don't care condition

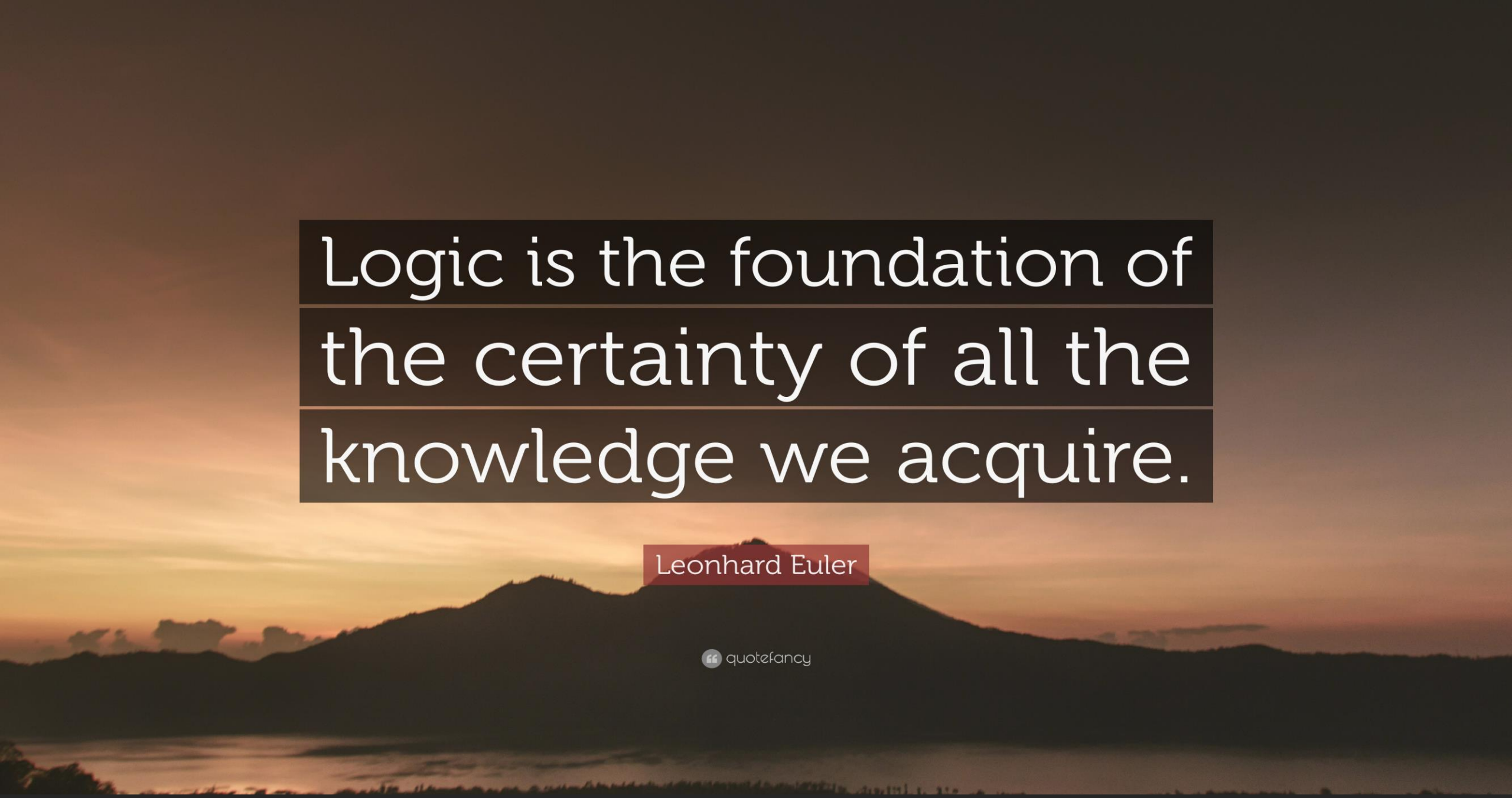
$$d(v,w,x,y,z) = \sum(0,2,25)$$

ผลลัพธ์

- ❑ Truth Table
- ❑ Boolean Function (Simplification)

หมายเหตุ

- ❑ ให้เขียนขั้นตอนและที่มาของผลลัพธ์ในแต่ละขั้นตอนให้ชัดเจน


Logic is the foundation of the certainty of all the knowledge we acquire.
Leonhard Euler
quotefancy