Digital circuits and logical design

数字电路与逻辑设计

5 Combinational Logic Analysis

第五章 组合逻辑分析

Combinational logic(组合逻辑) — the output at any instant of time depends only on what the inputs are at that time.

Sequential logic (时序逻辑) — the output will depend not only on the present input but also on the past history — what has happened earlier.

5-1 Basic Combinational Logic Circuits

5-2 Implementing Combinational Logic

5-3 The Universal Property of NAND and NOR Gates

5-4 Combinational Logic Using NAND and NOR Gates

5-5 Logic Circuit Operation with Pulse Waveform Inputs

System Application Activity

5.0 Analysis and Design of Combinational Logic Circuits

组合逻辑电路的分析与设计

5.0.1 Analysis of Combinational Circuits 组合电路分析

Analysis is a procedure from a logic circuit to function descriptions.

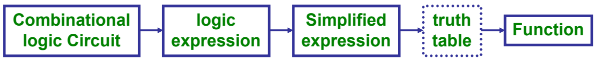
Purpose: Analysis is used to determine the behavior of a logical circuit , to verify that the behavior of a circuit matches its specification , or to assist in converting the circuit to a different form，either to reduce the number of gates or to realize it with different elements.

The three steps of analysis are listed as follows:

Step 1: Write out the algebraic expression of a logic circuit.
Step 2: Formalize the truth table .
Step 3: Describe the function or behavior of a logic circuit.

The three steps of analysis are listed as follows:

Step 1: Write out the algebraic expression of a logic circuit.
Step 2: Formalize the truth table .
Step 3: Describe the function or behavior of a logic circuit.



Combinational logic Circuit → logic expression → Simplified expression → truth table → Function

Example : Find a simplified logic expression and circuit for network of Fig (a).

Analysis:

$$Y(A,B,C)=(P_5+P_6)'$$
$$=(P_1P_2+(P_3+P_4))'$$
$$=((AB)'(A'+C)'+(BC+B'C'))'$$

Simplify:

$$Y=(AB+A'+C)(BC+B'C')'$$
$$=(B+A'+C)(B'C+BC')$$
$$=BC'+A'B'C+A'BC'+B'C$$
$$=BC'+B'C \qquad (1)$$
$$=B\oplus C \qquad (2)$$

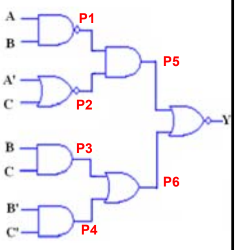


Fig (a)

This function has been reduced to a single exclusive-OR gate.

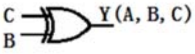


Fig (b)

Example : Find a simplified logic expression and circuit for network of Fig (a).

Analysis:

$$Y(A,B,C) = \overline{P_5 + P_6}$$
$$= \overline{P_1P_2 + (P_3 + P_4)}$$
$$= \overline{\overline{AB}\,\overline{\bar{A}+C} + (BC + \bar{B}\,\bar{C})}$$

Simplify:

$$Y = (AB + \bar{A} + C)\,\overline{BC + \bar{B}\,\bar{C}}$$
$$= (B + \bar{A} + C)(\bar{B}C + B\bar{C})$$
$$= B\bar{C} + \bar{A}\,\bar{B}C + \bar{A}B\bar{C} + \bar{B}C$$
$$= B\bar{C} + \bar{B}C \qquad （1）$$
$$= B \oplus C \qquad （2）$$

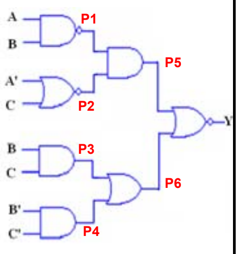


Fig (a)

This function has been reduced to a single exclusive-OR gate.

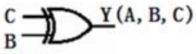


Fig (b)

Example

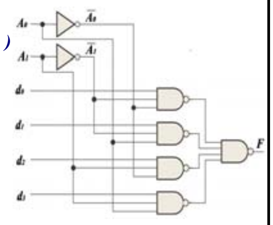
Analyze the logic *function* for the following circuit.

1. Write out the logic expression

$$F = \overline{\overline{(d_0\bar{A}_1\bar{A}_0)}\;\overline{(d_1\bar{A}_1A_0)}\;\overline{(d_2A_1\bar{A}_0)}\;\overline{(d_3A_1A_0)}}$$
$$= d_0(\bar{A}_1\bar{A}_0) + d_1(\bar{A}_1A_0) + d_2(A_1\bar{A}_0) + d_3(A_1A_0)$$
$$= d_0m_0 + d_1m_1 + d_2m_2 + d_3m_3$$

2. Formalize the truth table

A_1A_0	F
m_0	d_0
m_1	d_1
m_2	d_2
m_3	d_3



3. This is a 1-of-4 data selector/multiplexer.

5.0.2 Design (Synthesis) of Combinational Circuits

组合电路设计（综合）

Digital circuits are designed by transforming a word descriptions of function into a set of logic equations and then realizing the equations with logic elements.

The design process for combinational circuits is shown as follows :

1: Represent each of the inputs and outputs in variable.

2: Formalize the design specification either in the form of truth table or of algebraic expressions.

3: Simplify the description (algebraic expressions) in terms of gates .

4: Implement the circuits with components.

Example

Design a full subtracter, that is a circuit which computes a-b-c, where c is the borrow from the next less significant digit and produces a difference , D, and a borrow, P from the next more significant bit.

Solution 1:

(1) showing the truth table

(2) simplifying logic expression

(3) drawing the logic circuit

(1) the truth table

a b c	D P
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

(2) *logic expression*

$$D = \bar{a}\,\bar{b}\,c + \bar{a}\,b\,\bar{c} + a\,\bar{b}\,\bar{c} + a\,b\,c$$

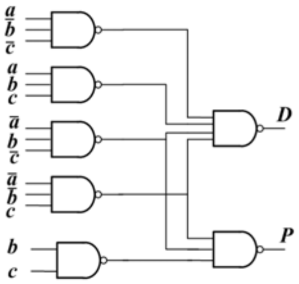
$$P = \bar{a}\,\bar{b}\,c + \bar{a}\,b\,\bar{c} + b\,c$$

(2) *logic expression*

$$D = \bar{a}\,\bar{b}\,c + \bar{a}\,b\,\bar{c} + a\,\bar{b}\,\bar{c} + a\,b\,c$$

$$P = \bar{a}\,\bar{b}\,c + \bar{a}\,b\,\bar{c} + b\,c$$

(3) *logic circuit*



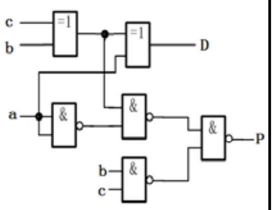
Solution 2:

(2) *logic expression*

$$D = \bar{a}\,\bar{b}\,c + \bar{a}\,b\,\bar{c} + a\,\bar{b}\,\bar{c} + a\,b\,c$$
$$= (\bar{a}\,\bar{b} + a\,b)\,c + (\bar{a}\,b + a\,\bar{b})\,\bar{c}$$
$$= \overline{(a \oplus b)}\,c + (a \oplus b)\,\bar{c}$$
$$= a \oplus b \oplus c$$

$$P = \bar{a}\,\bar{b}\,c + \bar{a}\,b\,\bar{c} + b\,c$$
$$= \bar{a}\,(\bar{b}\,c + b\,\bar{c}) + b\,c$$
$$= \bar{a}\,(b \oplus c) + b\,c$$

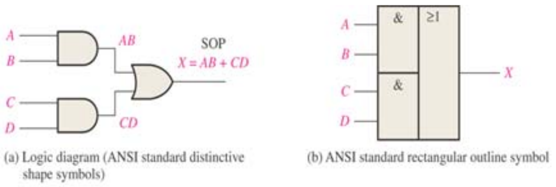
(3) *logic circuit*



5-1 Basic Combinational Logic Circuits

5.1.1 AND-OR Logic

Figure 5-1 (a) shows an AND-OR circuit consisting of two 2-input AND gates and one 2-input OR gate; Figure 5-1(b) is the ANSI standard rectangular outline symbol. The Boolean expression for the AND gate outputs and the resulting SOP expression for output X are shown on the diagram. In general, an AND-OR circuit can have any number of AND gates each with any number of inputs.

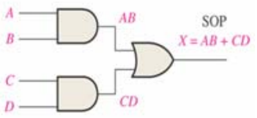


(a) Logic diagram (ANSI standard distinctive shape symbols)

(b) ANSI standard rectangular outline symbol

Figure 5–1 An example of AND-OR logic

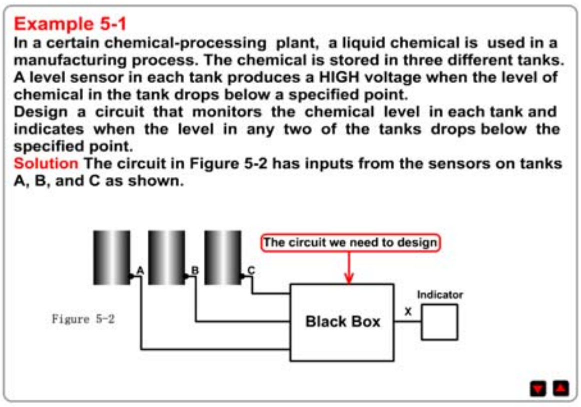
The truth table for the 4-input AND-OR logic circuit is shown in Table 5-1. The intermediate AND gate outputs (the AB and CD columns) are also shown in the table.



For a 4-input AND-OR logic circuit, the output X is HIGH(1) if both input A and input B are HIGH(1) or both input C and input D are HIGH(1).

Table 5-1

INPUTS						OUTPUT
A	B	C	D	AB	CD	X
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	1	1	0	1	1
0	1	0	0	0	0	0
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	1	1	0	1	1
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	0	1	1	0	1	1
1	1	0	0	1	0	1
1	1	0	1	1	0	1
1	1	1	0	1	0	1
1	1	1	1	1	1	1



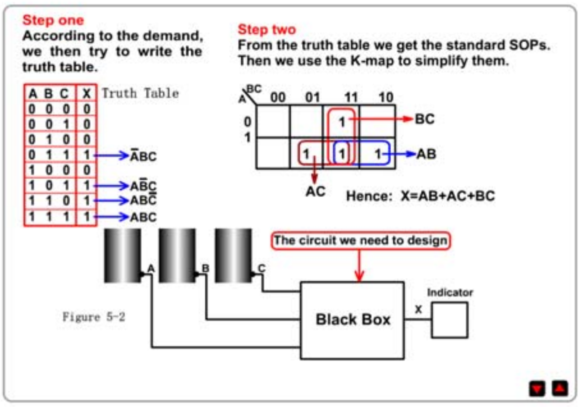
Example 5-1

In a certain chemical-processing plant, a liquid chemical is used in a manufacturing process. The chemical is stored in three different tanks. A level sensor in each tank produces a HIGH voltage when the level of chemical in the tank drops below a specified point.

Design a circuit that monitors the chemical level in each tank and indicates when the level in any two of the tanks drops below the specified point.

Solution The circuit in Figure 5-2 has inputs from the sensors on tanks A, B, and C as shown.

Figure 5-2



Step one

According to the demand, we then try to write the truth table.

Truth Table

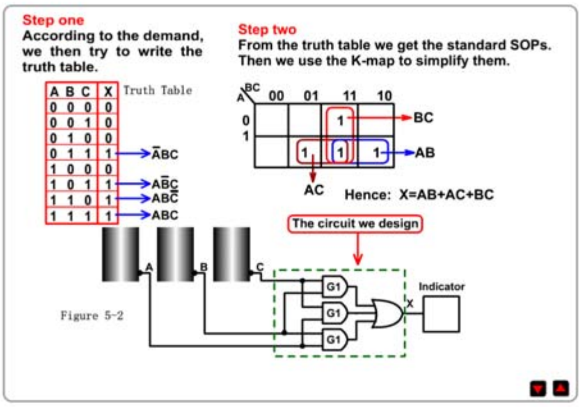
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Step two

From the truth table we get the standard SOPs. Then we use the K-map to simplify them.

Hence: X=AB+AC+BC

Figure 5-2



Step one

According to the demand, we then try to write the truth table.

Truth Table

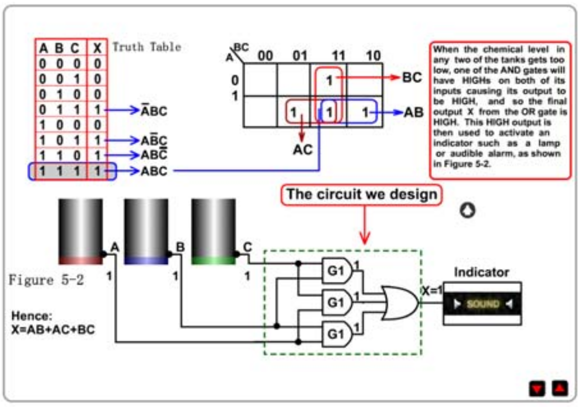
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Step two

From the truth table we get the standard SOPs. Then we use the K-map to simplify them.

Hence: X=AB+AC+BC

Figure 5-2



Truth Table

A	B	C	X
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

When the chemical level in any two of the tanks gets too low, one of the AND gates will have HIGHs on both of its inputs causing its output to be HIGH, and so the final output X from the OR gate is HIGH. This HIGH output is then used to activate an indicator such as a lamp or audible alarm, as shown in Figure 5-2.

Figure 5-2

Hence:
X=AB+AC+BC

5.1.2 AND-OR-Invert Logic

When the output of an AND-OR circuit is complemented (inverted), it results an AND-OR-Invert circuit.

POS expressions can be implemented with AND-OR-Invert logic. This is illustrated as follows, starting with a POS expression and developing the corresponding AND-OR-Invert expression.

$$X = (\overline{A}+\overline{B})(\overline{C}+\overline{D}) = (\overline{AB})(\overline{CD}) = \overline{\overline{(\overline{AB})(\overline{CD})}} = \overline{\overline{\overline{AB}}+\overline{\overline{CD}}} = \overline{AB+CD}$$

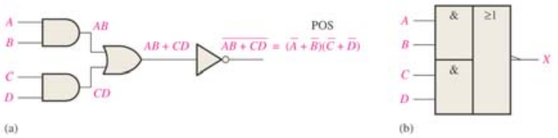
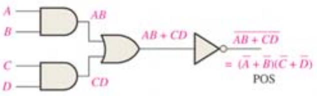

A
B
C
D
AB
CD
AB + CD
POS
$\overline{AB+CD} = (\overline{A}+\overline{B})(\overline{C}+\overline{D})$
(a)
A
B
C
D
&
&
≥1
X
(b)

Figure 5–3 An AND-OR-Invert circuit produces a POS output.

The truth table for the 4-input AND-OR-Invert logic circuit is shown in Table (a).



For a 4-input AND-OR-Invert logic circuit, the output X is LOW(0) if both input A and input B are HIGH(1) or both input C and input D are HIGH(1).

Table (a)

INPUTS						OUTPUT
A	B	C	D	AB	CD	X
0	0	0	0	0	0	1
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	0	1
0	1	1	0	0	0	1
0	1	1	1	0	1	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	1	1	0

EXAMPLE

In a certain chemical-processing plant, a liquid chemical is used in a manufacturing process. The chemical is stored in three different tanks. A level sensor in each tank produces a LOW voltage when the level of chemical in the tank drops below a specified point.

Modify the circuit in Figure 5–2 to operate with the different input levels and still produce a HIGH output to activate the indicator when the level in any two of the tanks drops below the critical point. Show the logic diagram.

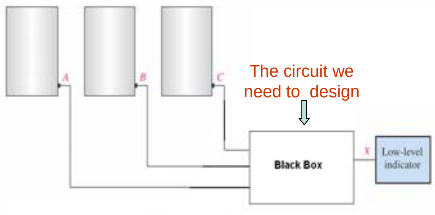
Solution

The circuit in Figure (a) has inputs from the sensors on tanks A, B, and C as shown.

Step one

According to the demand, we then try to write the truth table.

A	B	C	X
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0



The circuit we need to design

⇩

Figure (a)

Step one

According to the demand, we then try to write the truth table.

A	B	C	X
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Step two

Map directly from the truth table to a Karnaugh map.

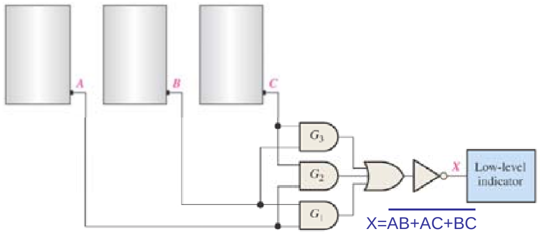
A \ BC	00	01	11	10
0	1	1	0	1
1	1	0	0	0

AC BC AB

Then we use the K-map to simplify them.

Hence: $X=\overline{AB+AC+BC}$

When the chemical level in any two of the tanks gets too low, each AND gate will have a LOW on at least one input causing its output to be LOW and, thus, the final output X from the inverter is HIGH. This HIGH output is then used to activate an indicator such as a lamp or audible alarm, as shown in Figure (b).

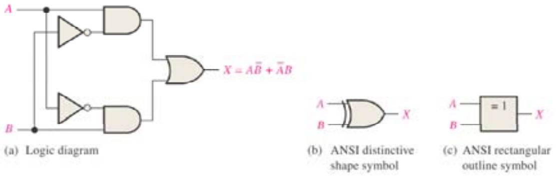


$X=\overline{AB+AC+BC}$

Figure (b)

5.1.3 Exclusive-OR Logic

Exclusive-OR logic diagram and symbols



(a) Logic diagram (b) ANSI distinctive shape symbol (c) ANSI rectangular outline symbol

The output expression for the circuit is

$$X = A\overline{B} + \overline{A}B$$

truth table for an exclusive-OR

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Notice that the output is HIGH only when the two inputs are at opposite levels.

A special exclusive-OR operator ⊕ is often used, so the expression $X=A\overline{B}+\overline{A}B$ can be stated as "X is equal to A exclusive-OR B" and can be written as

$$X = A \oplus B$$

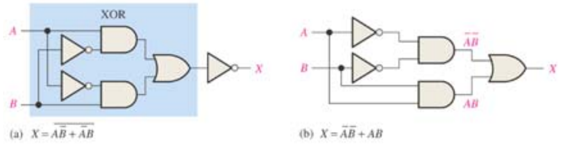
5.1.4 Exclusive-NOR Logic

The complement of the exclusive-OR function is the exclusive-NOR, which is derived as follows:

$$X = \overline{A\overline{B} + \overline{A}B} = \overline{A\overline{B}}\,\overline{\overline{A}B} = (\overline{A}+B)(A+\overline{B}) = \overline{A}\,\overline{B} + AB$$

Notice that the output X is HIGH only when the two inputs A and B, are at the same level.

Two equivalent ways of implementing the exclusive-NOR:



(a) $X = \overline{A\overline{B} + \overline{A}B}$ (b) $X = \overline{A}\,\overline{B} + AB$

truth table for an exclusive-NOR

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

Notice that the output X is HIGH only when the two inputs A and B, are at the same level.

A special exclusive-NOR operator ⊙ is often used, so the expression $X=\overline{A}\,\overline{B}+AB$ can be stated as "X is equal to A exclusive-NOR B" and can be written as

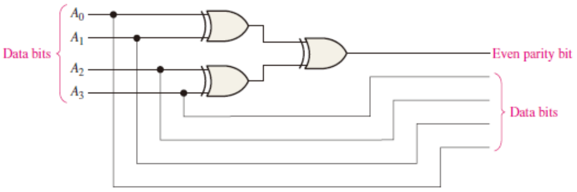
$$X = \overline{A \oplus B} = A \odot B$$

EXAMPLE

Use exclusive-OR gates to implement an even-parity code generator for an original 4-bit code.

Solution

The following circuit produces a 1 output when there is an odd number of 1s on the inputs in order to make the total number of 1s in the output code even. A 0 output is produced when there is an even number of 1s on the inputs.

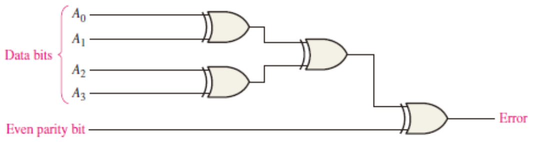


EXAMPLE

Use exlusive-OR gates to implement an even-parity checker for the 5-bit.

Solution

The following circuit produces a 1 output when there is an error in the five-bit code and a 0 when there is no error.



5-2 Implementing Combinational Logic

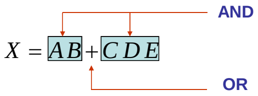
In this section, examples are used to illustrate how to implement a logic circuit from a Boolean expression or A truth table. Minimization of a logic circuit is also included.

5.2.1 From a Boolean Expression to a logic circuit

For every Boolean expression there is a logic circuit, and for every logic circuit there is a Boolean expression.

Let's implement the following expression:

$$X = AB + CDE$$

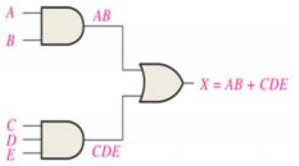


AND

$X = \boxed{AB}+\boxed{CDE}$

OR

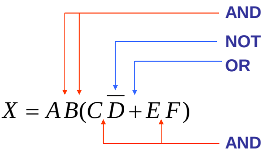
Logic circuit for *X = AB + CDE*:



Let's implement the following expression:

$$X = AB(C\overline{D}+EF)$$

Logic circuit for *X = AB(CD̅ + EF)*:



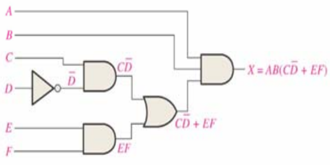
AND

NOT

OR

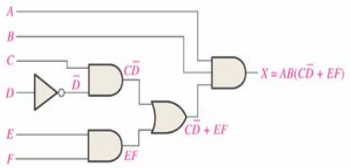
$X = AB(C\overline{D}+EF)$

AND

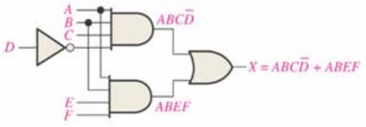


Logic circuit for *X = AB(CD̅ + EF)=ABCD̅+ABEF*

Logic circuit for *X = AB(CD̅ + EF)*:



Logic circuit for *X = ABCD̅ + ABEF* :



5.2.2 From Truth Table to a Logic Circuit

You can write the SOP expression from the truth table and then implement the logic circuit.

Design a logic circuit to implement the operation specified in the truth table of Table (A) .

The Boolean SOP expression obtained from the truth table by ORing the product terms for which X=1 is

$$X = \overline{A}BC + A\overline{B}\,\overline{C}$$

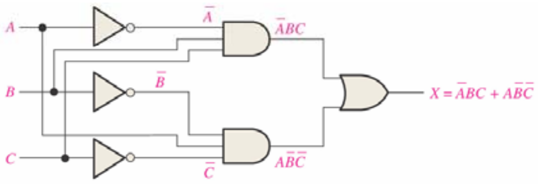
Table (A)

INPUTS			OUTPUT	PRODUCT
A	B	C	X	TERM
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}BC$
1	0	0	1	$A\overline{B}\,\overline{C}$
1	0	1	0	
1	1	0	0	
1	1	1	0	

The Boolean SOP expression obtained from the truth table by ORing the product terms for which X=1 is

$$X = \overline{A}BC + A\overline{B}\,\overline{C}$$

The implementation of this logic function is illustrated in the following Figure.



EXAMPLE

Design a logic circuit to implement the operation specified in the truth table of Table (B) .

Solution

Notice that X=1 for only three of the input conditions. Therefore, the logic expression is

$$X = \overline{A}BC + A\overline{B}C + AB\overline{C}$$

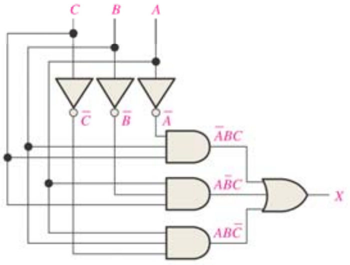
Table (B)

INPUTS			OUTPUT	PRODUCT
A	B	C	X	TERM
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}BC$
1	0	0	0	
1	0	1	1	$A\overline{B}C$
1	1	0	1	$AB\overline{C}$
1	1	1	0	

Notice that X=1 for only three of the input conditions. Therefore, the logic expression is

$$X = \overline{A}BC + A\overline{B}C + AB\overline{C}$$

The logic gates required are three inverters, three 3-input AND gates and one 3-input OR gate. The logic circuit is shown in the following Figure.



EXAMPLE

Develop a logic circuit with four input variables that will only produce 1 output when exactly three input variables is 1s.

Solution

Out of sixteen possible combinations of four variables, the combinations in which three are exactly three 1s are listed in Table (C).

The product terms are ORed to get the following expression:

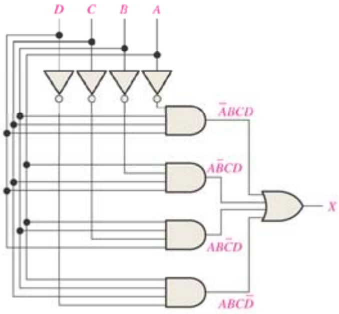
$$X = \overline{A}BCD + A\overline{B}CD + AB\overline{C}D + ABC\overline{D}$$

Table (C)

INPUTS				PRODUCT TERM
A	B	C	D	
0	1	1	1	$\overline{A}BCD$
1	0	1	1	$A\overline{B}CD$
1	1	0	1	$AB\overline{C}D$
1	1	1	0	$ABC\overline{D}$

$$X = \overline{A}BCD + A\overline{B}CD + AB\overline{C}D + ABC\overline{D}$$

The expression is implemented in the following figure with AND-OR logic.



EXAMPLE

Reduce the combinational logic circuit in the following Figure to a minimum form.

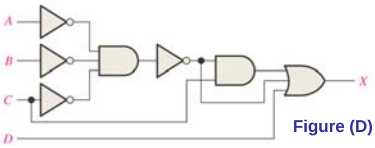


Figure (D)

Solution

The expression for the output of the circuit is

$$X = (\overline{\overline{A}\,\overline{B}\,\overline{C}})C + \overline{\overline{A}\,\overline{B}\,\overline{C}} + D$$

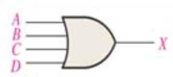
Applying DeMorgan's theorem and Boolean algebra.

$$X = (\overline{\overline{A}\,\overline{B}\,\overline{C}})C + \overline{\overline{A}\,\overline{B}\,\overline{C}} + D = (\overline{\overline{A}\,\overline{B}\,\overline{C}})(C+1) + D$$

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

$$= A + B + C + D$$

The simplified circuit is a 4-input OR gate.



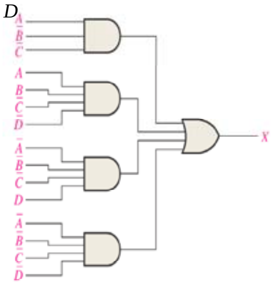
EXAMPLE

Minimize the combinational logic circuit in the following Figure Inverts for the complemented variables are not shown.

Solution

The output expression is

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



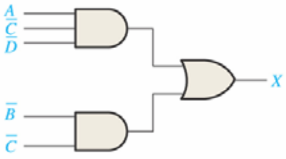
Map the SOP expression and group the cells:

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

The resulting minimum SOP expression is

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

Logic circuit for $X = A\overline{C}\overline{D} + \overline{B}\overline{C}$:



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

$A\overline{C}\,\overline{D}$ $\overline{B}\,\overline{C}$ X

5-3 The Universal Property of NAND and NOR Gates

In this section, the universal property of the NAND gate and the NOR gate is discussed.

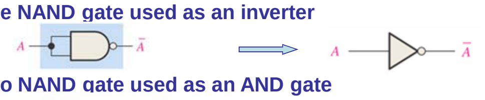
5.3.1 The NAND gate as a Universal Logic Element

NAND gates can be used produce any Logic function.

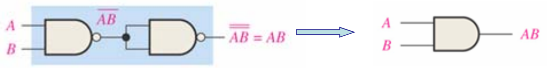
The NAND gate is a universal gate because it can be used produce the NOT, the AND, the OR, and NOR functions.

Universal application of NAND gates

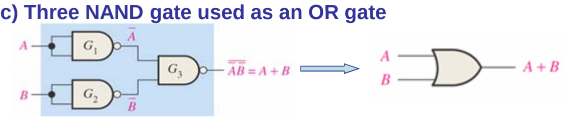
(a) One NAND gate used as an inverter



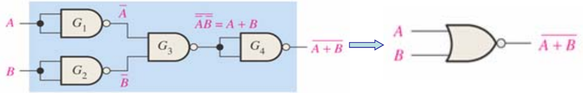
(b) Two NAND gate used as an AND gate



(c) Three NAND gate used as an OR gate



(d) Four NAND gate used as an NOR gate



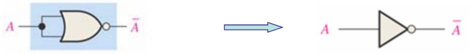
5.3.2 The NOR gate as a Universal Logic Element

NOR gates can be used produce any Logic function.

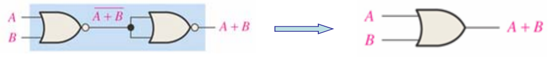
Like the NAND gate, the NOR gate can be used produce the NOT, AND, OR, and NAND functions.

Universal application of NOR gates

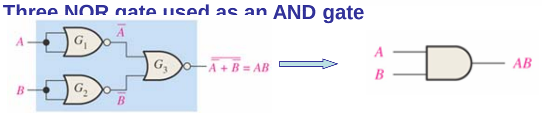
(a) One NOR gate used as an inverter



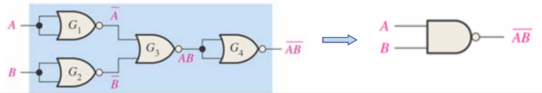
(b) Two NOR gate used as an OR gate



(c) Three NOR gate used as an AND gate



(d) Four NOR gate used as an NAND gate



5-4 Combinational Logic Using NAND and NOR Gates

NAND and NOR logic gates can be used to implement a logic function.

5.4.1 NAND logic

A NAND gate can function as either a NAND or a negative-OR because, by DeMorgan's theorem.

$$\boxed{\overline{AB}} = \overline{A} + \overline{B}$$

NAND

negative-OR

NAND logic for X=AB+CD.

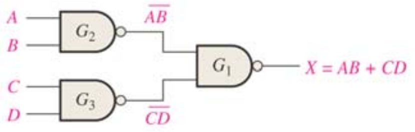


Figure (a)

Consider the NAND logic in Figure (a).

The output expression is developed in the following steps:

$$X = \overline{(\overline{AB})(\overline{CD})}$$

$$= \overline{\overline{AB}} + \overline{\overline{CD}}$$

$$= AB + CD$$

5.4.2 NOR logic

A NOR gate can function as either a NOR or a negative-AND because, by DeMorgan’s theorem.

$$\overline{A+B} = \boxed{\overline{A}\,\overline{B}}$$

NOR

negative-AND

NOR logic for X=(A+B)(C+D).

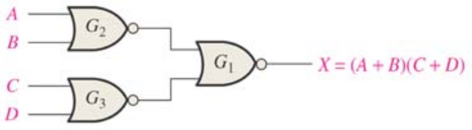


Figure (b)

Consider the NOR logic in Figure (b).

The output expression is developed as follows:

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

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

$$= (A+B)(C+D)$$

5-5 Logic Circuit Operation with Pulse Waveform Inputs

Several examples of general combinational logic circuits with pulse waveform inputs are examined in this section.

The operation of any gate is the same regardless of whether its inputs are pulsed or constant levels. The nature of the inputs does not alter the truth table of a circuit.

The following is a review of the operation of individual gates:

1. The output of an AND gate is HIGH only when all inputs are HIGH at the same time.
2. The output of an OR gate is HIGH only when at least one of its input is HIGH.
3. The output of a NAND gate is LOW only when all inputs are HIGH at the same time.
4. The output of a NOR gate is LOW only when at least one of its input is HIGH.

EXAMPLE

Determine the final output waveform X for the circuit in Figure (A) ,with input waveforms A, B, and C as shown.

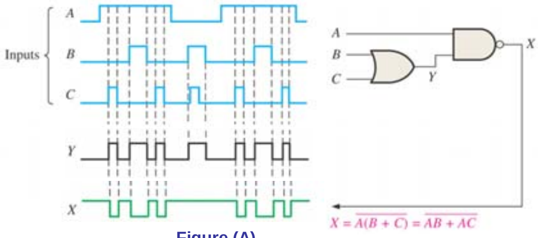


Figure (A)

solution

The output waveforms are shown in Figure (A).

The output X is LOW when both A and B are HIGH or when both A and C are HIGH.

EXAMPLE

Draw the timing diagram for the circuit in Figure (B) showing the outputs of G_1, G_2, and G_3 with the input waveforms, A, and B, as indicated.

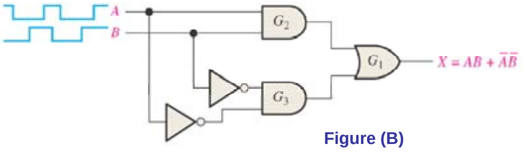


Figure (B)

solution

When both inputs are HIGH or when both inputs are LOW, the output X is HIGH as shown in Figure (C). Notice that this is an exclusive-NOR circuit. The intermediate outputs of G_2 and G_3 are also shown in Figure (C).

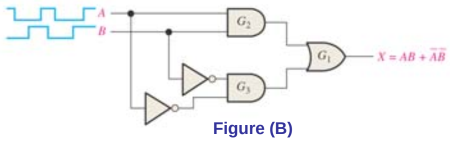


Figure (B)

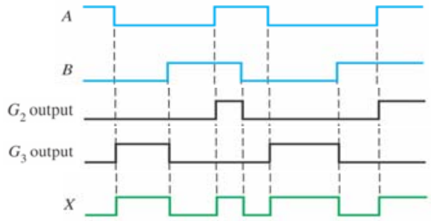
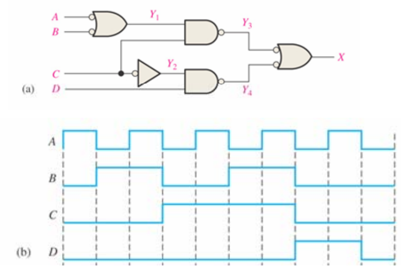


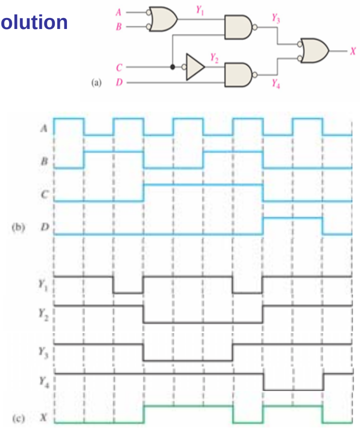
Figure (C)

EXAMPLE

Determine the output waveform X for the logic circuit in Figure (a) by first finding the intermediate waveform at each of points Y_1, Y_2, Y_3, and Y_4. The input waveforms are *shown in Figure (b).*



solution



All the intermediate waveforms and final output waveform are shown in the timing diagram of Figure (c).

EXAMPLE

Determine the output waveform X for the circuit in Figure (D) ,directly from the output expression.

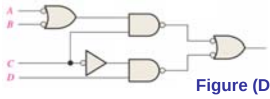


Figure (D)

solution

The output expression for the circuit is developed in Figure (E).

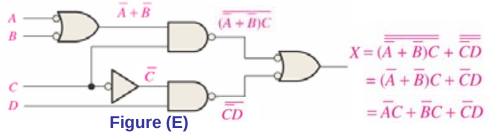

A
B
C
D
$\overline{A} + \overline{B}$
$\overline{(\overline{A} + \overline{B})C}$
$\overline{C}$
$\overline{\overline{C}D}$
$X = \overline{\overline{(\overline{A} + \overline{B})C}} + \overline{\overline{\overline{C}D}}$
$= (\overline{A} + \overline{B})C + \overline{C}D$
$= \overline{A}C + \overline{B}C + \overline{C}D$

Figure (E)

The SOP form indicates that the output is HIGH when A is LOW and C is HIGH or when B is LOW and C is HIGH or when C is LOW and D is HIGH.

The result is shown in Figure (F). The corresponding product terms for each waveform condition that results in HIGH output are indicated.

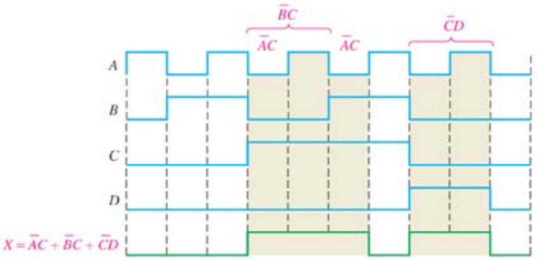
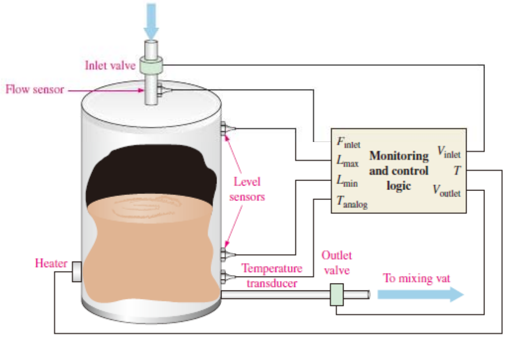
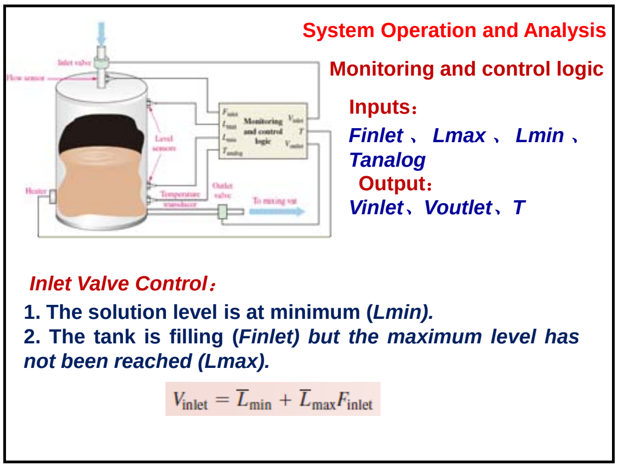


Figure (F)

System Application Activity

Tank Control





System Operation and Analysis

Monitoring and control logic

Inputs：

Finlet 、 Lmax 、 Lmin 、 Tanalog

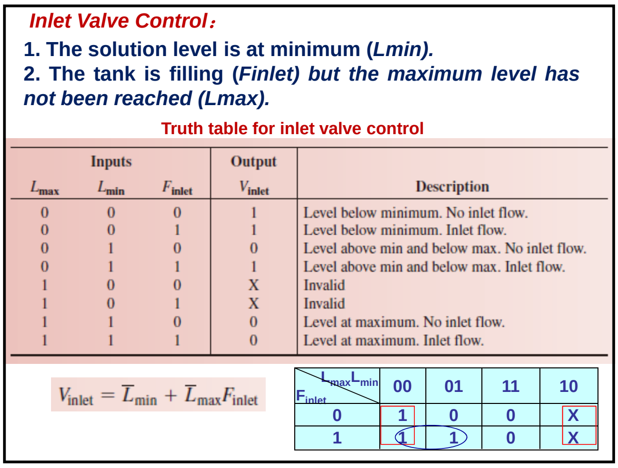
Output：

Vinlet、Voutlet、T

Inlet Valve Control：

1. The solution level is at minimum (*Lmin).*
2. The tank is filling (*Finlet) but the maximum level has not been reached (Lmax).*

$$V_{inlet} = \overline{L}_{min} + \overline{L}_{max}F_{inlet}$$



Inlet Valve Control：

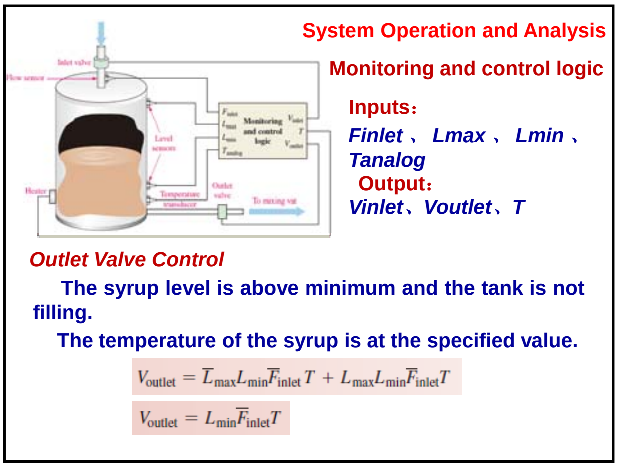
1. The solution level is at minimum (*Lmin).*
2. The tank is filling (*Finlet) but the maximum level has not been reached (Lmax).*

Truth table for inlet valve control

Inputs			Output	
L_{max}	L_{min}	F_{inlet}	V_{inlet}	Description
0	0	0	1	Level below minimum. No inlet flow.
0	0	1	1	Level below minimum. Inlet flow.
0	1	0	0	Level above min and below max. No inlet flow.
0	1	1	1	Level above min and below max. Inlet flow.
1	0	0	X	Invalid
1	0	1	X	Invalid
1	1	0	0	Level at maximum. No inlet flow.
1	1	1	0	Level at maximum. Inlet flow.

$$V_{inlet} = \overline{L}_{min} + \overline{L}_{max}F_{inlet}$$

F_{inlet} \ $L_{max}L_{min}$	00	01	11	10
0	1	0	0	X
1	1	1	0	X



System Operation and Analysis

Monitoring and control logic

Inputs：

Finlet 、 Lmax 、 Lmin 、 Tanalog

Output：

Vinlet、Voutlet、T

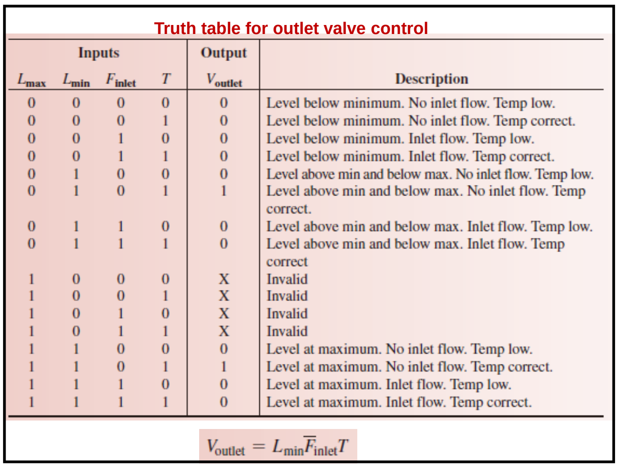
Outlet Valve Control

The syrup level is above minimum and the tank is not filling.

The temperature of the syrup is at the specified value.

$$V_{outlet} = \overline{L}_{max}L_{min}\overline{F}_{inlet}T + L_{max}L_{min}\overline{F}_{inlet}T$$

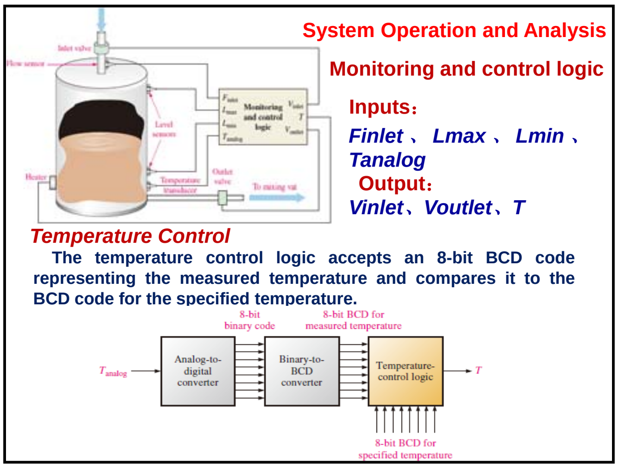
$$V_{outlet} = L_{min}\overline{F}_{inlet}T$$



Truth table for outlet valve control

Inputs				Output	
L_{max}	L_{min}	F_{inlet}	T	V_{outlet}	Description
0	0	0	0	0	Level below minimum. No inlet flow. Temp low.
0	0	0	1	0	Level below minimum. No inlet flow. Temp correct.
0	0	1	0	0	Level below minimum. Inlet flow. Temp low.
0	0	1	1	0	Level below minimum. Inlet flow. Temp correct.
0	1	0	0	0	Level above min and below max. No inlet flow. Temp low.
0	1	0	1	1	Level above min and below max. No inlet flow. Temp correct.
0	1	1	0	0	Level above min and below max. Inlet flow. Temp low.
0	1	1	1	0	Level above min and below max. Inlet flow. Temp correct
1	0	0	0	X	Invalid
1	0	0	1	X	Invalid
1	0	1	0	X	Invalid
1	0	1	1	X	Invalid
1	1	0	0	0	Level at maximum. No inlet flow. Temp low.
1	1	0	1	1	Level at maximum. No inlet flow. Temp correct.
1	1	1	0	0	Level at maximum. Inlet flow. Temp low.
1	1	1	1	0	Level at maximum. Inlet flow. Temp correct.

$$V_{outlet} = L_{min}\overline{F}_{inlet}T$$



System Operation and Analysis

Monitoring and control logic

Inputs：

Finlet 、 Lmax 、 Lmin 、 Tanalog

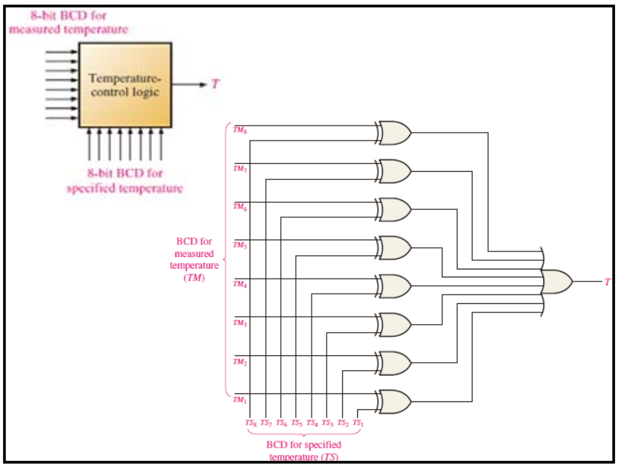
Output：

Vinlet、Voutlet、T

Temperature Control

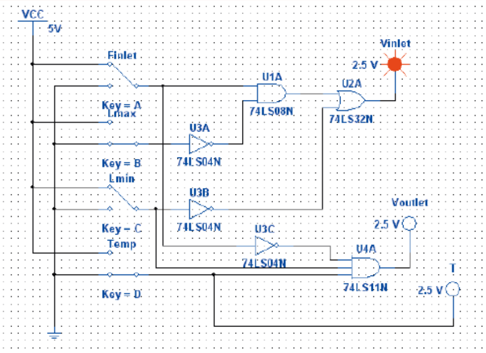
The temperature control logic accepts an 8-bit BCD code representing the measured temperature and compares it to the BCD code for the specified temperature.

8-bit binary code
8-bit BCD for measured temperature
T_{analog}
Analog-to-digital converter
Binary-to-BCD converter
Temperature-control logic
T
8-bit BCD for specified temperature

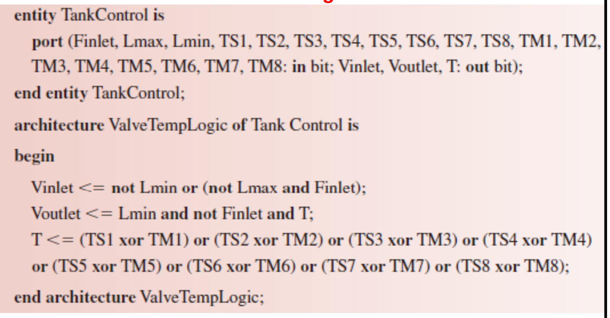


8-bit BCD for measured temperature
Temperature-control logic
T
8-bit BCD for specified temperature
TM_8
TM_7
TM_6
TM_5
TM_4
TM_3
TM_2
TM_1
BCD for measured temperature (TM)
TS_8 TS_7 TS_6 TS_5 TS_4 TS_3 TS_2 TS_1
BCD for specified temperature (TS)
T

Simulation of the Valve Control Logic



VHDL Code for Tank Control Logic



```
entity TankControl is
    port (Finlet, Lmax, Lmin, TS1, TS2, TS3, TS4, TS5, TS6, TS7, TS8, TM1, TM2,
    TM3, TM4, TM5, TM6, TM7, TM8: in bit; Vinlet, Voutlet, T: out bit);
end entity TankControl;
architecture ValveTempLogic of Tank Control is
begin
    Vinlet <= not Lmin or (not Lmax and Finlet);
    Voutlet <= Lmin and not Finlet and T;
    T <= (TS1 xor TM1) or (TS2 xor TM2) or (TS3 xor TM3) or (TS4 xor TM4)
    or (TS5 xor TM5) or (TS6 xor TM6) or (TS7 xor TM7) or (TS8 xor TM8);
end architecture ValveTempLogic;
```