Digital circuit design notes

Vikas Dhiman for ECE275[1]

February 3, 2025

[1] The notes are from the following books [1, 2]. Not intended for distribution outside the class.

Contents

Chapter 1

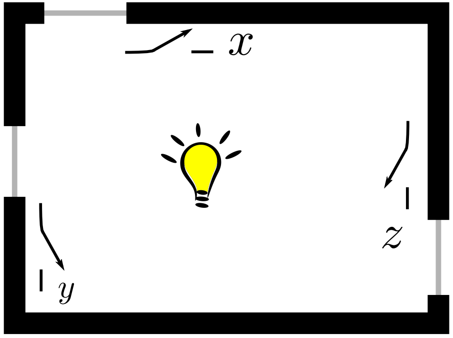
Boolean Algebra

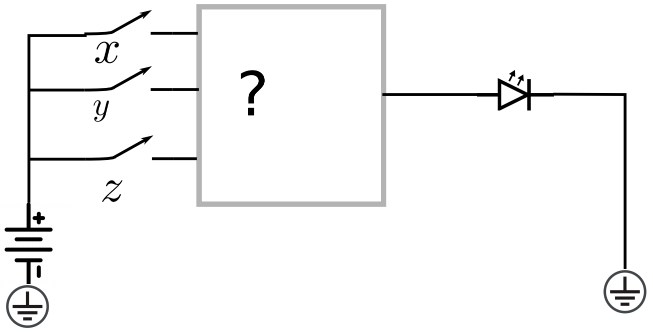
1.1 Learning objectives

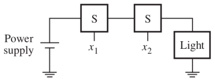
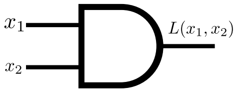
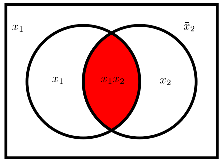
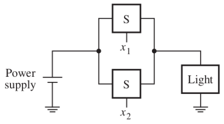
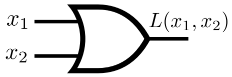
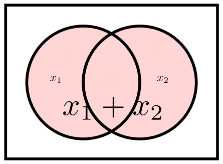
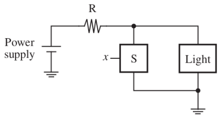
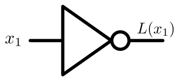
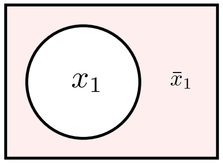
1. Representing digital circuits
2. Converting between different notations: Boolean expression, logic networks and switching circuits
3. Converting between different logic network specifications: truth table, minterm, maxterms, product of sums canonical form and sum of product canonical form.
4. Introduce truth tables as Behavioral Verilog
5. This handout has 11 homework problems totaling to 140 marks

1.2 Motivating Problem

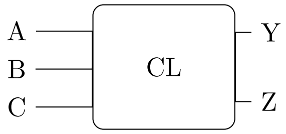
Example 1. *Assume that a large room has three doors and that a switch near each door controls a light in the room. It has to be possible to turn the light on or off by changing the state of any one of the switches.*





<table>
<tr><th>Name</th><th>C/Verilog</th><th>Boolean expr.</th><th>Truth Table</th><th>Switching circuit</th><th>(ANSI) symbol</th><th>Venn diagram</th></tr>
<tr><td>AND Gate</td><td><code>L = x1 & x2</code></td><td>$L = x_1 \cdot x_2 = x_1 x_2$</td><td>$\begin{array}{cc|c} x_1 & x_2 & x_1 \cdot x_2 \\ \hline 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{array}$</td><td> </td><td> </td><td> </td></tr>
<tr><td>OR Gate</td><td><code>L = x1 | x2</code></td><td>$L = x_1 + x_2$</td><td>$\begin{array}{cc|c} x_1 & x_2 & x_1 + x_2 \\ \hline 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{array}$</td><td> </td><td> </td><td> </td></tr>
<tr><td>NOT Gate</td><td><code>L = ∼ x1</code></td><td>$L = \bar{x}_1 = x_1'$</td><td>$\begin{array}{c|c} x_1 & \bar{x}_1 \\ \hline 0 & 1 \\ 1 & 0 \end{array}$</td><td> </td><td> </td><td> </td></tr>
</table>

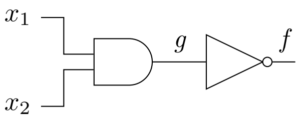
1.3 Digital circuits or networks



$$Y = F(A, B, C) \qquad Z = G(A, B, C)$$

1.4 Two input networks

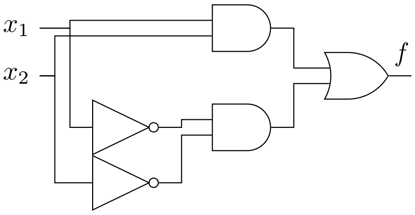
Example 2. *Convert the following (ANSI) network into a Boolean expression, a truth table and a Venn diagram.*



Example 3. *Convert the following Boolean expression into a (ANSI) network, a truth table and a Venn diagram:*

$$f = \overline{x_1 + x_2}$$

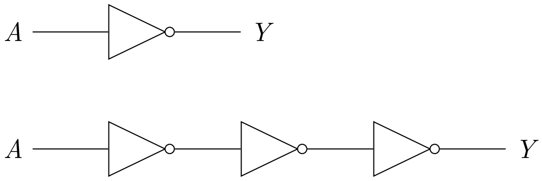
Problem 1 (10 marks)**.** *Convert the following (ANSI) network into a Boolean expression, a truth table and a Venn diagram.*



Example 4. *Convert the following Boolean expression into a network, a truth table and a Venn diagram:*

$$f = x_1\bar{x}_2 + \bar{x}_1 x_2$$

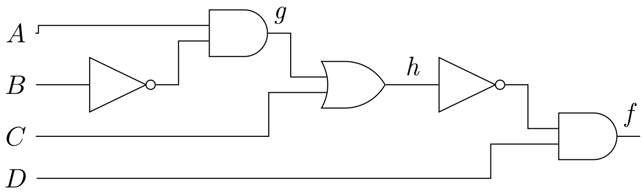
Problem 2 (5 marks)**.** *Can two different circuits have the same truth table? Can two different truth tables have the same circuit? Consider the following two circuits for example*



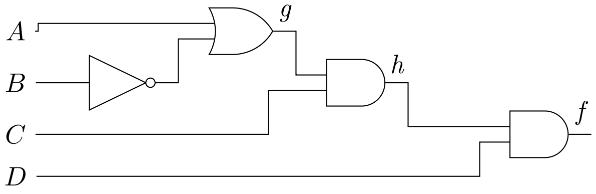
Remark 1. *Truth tables and Venn diagrams define* what *the combinational circuit should do. Truth tables define output for every input. Boolean expression and networks define* how *to achieve the desired input output relationship.*

1.5 Multi-input networks

Example 5. *Convert the following (ANSI) network into a Boolean expression and a truth table.*



Problem 3 (20 marks)**.** *Convert the following (ANSI) network into a Boolean expression and a truth table.*



1.6 Minterms and Maxterms

1.6.1 Minterms

Minterm is a product involving all inputs (or complements) to a function. Every row of a truth table has a corresponding minterm. Minterm is true if and only if the corresponding row in the table is active.

Minterms defined as follows for each row of a two input truth table:

A	B	minterm	minterm
			name
0	0	$\bar{A}\bar{B}$	m_0
0	1	$\bar{A}B$	m_1
1	0	$A\bar{B}$	m_2
1	1	AB	m_3

Consider a two input circuit whose output Y is given by the truth table:

A	B	Y	minterm	minterm
				name
0	0	0	$\bar{A}\bar{B}$	m_0
0	1	1	$\bar{A}B$	m_1
1	0	0	$A\bar{B}$	m_2
1	1	1	AB	m_3

then $Y = \bar{A}B + AB = m_1 + m_3 = \sum(1,3)$.

This also gives the *sum of products canonical form.*

Example 6. *What is the minterm m_{13} for a 4-input circuit with inputs x, y, z, w (ordered from MSB to LSB).*

Problem 4 (5 marks)**.** *What is the minterm m_{23} for a 5-input circuit with inputs a, b, c, d, e (ordered from MSB to LSB).*

Example 7. *Convert the following 4-input truth table into sum of minterms and sum of products canonical form.*

minterm name	*A*	*B*	*C*	*D*	*f*
m_0	*0*	*0*	*0*	*0*	*0*
m_1	*0*	*0*	*0*	*1*	*1*
m_2	*0*	*0*	*1*	*0*	*0*
m_3	*0*	*0*	*1*	*1*	*0*
m_4	*0*	*1*	*0*	*0*	*0*
m_5	*0*	*1*	*0*	*1*	*1*
m_6	*0*	*1*	*1*	*0*	*0*
m_7	*0*	*1*	*1*	*1*	*0*
m_8	*1*	*0*	*0*	*0*	*0*
m_9	*1*	*0*	*0*	*1*	*0*
m_{10}	*1*	*0*	*1*	*0*	*0*
m_{11}	*1*	*0*	*1*	*1*	*0*
m_{12}	*1*	*1*	*0*	*0*	*0*
m_{13}	*1*	*1*	*0*	*1*	*1*
m_{14}	*1*	*1*	*1*	*0*	*0*
m_{15}	*1*	*1*	*1*	*1*	*0*

Problem 5 (10 marks)**.** *Convert the following 4-input truth table into sum of minterms and sum of products canonical form.*

minterm name	*A*	*B*	*C*	*D*	*f*
m_0	*0*	*0*	*0*	*0*	*0*
m_1	*0*	*0*	*0*	*1*	*0*
m_2	*0*	*0*	*1*	*0*	*0*
m_3	*0*	*0*	*1*	*1*	*1*
m_4	*0*	*1*	*0*	*0*	*0*
m_5	*0*	*1*	*0*	*1*	*0*
m_6	*0*	*1*	*1*	*0*	*0*
m_7	*0*	*1*	*1*	*1*	*1*
m_8	*1*	*0*	*0*	*0*	*0*
m_9	*1*	*0*	*0*	*1*	*0*
m_{10}	*1*	*0*	*1*	*0*	*0*
m_{11}	*1*	*0*	*1*	*1*	*1*
m_{12}	*1*	*1*	*0*	*0*	*0*
m_{13}	*1*	*1*	*0*	*1*	*1*
m_{14}	*1*	*1*	*1*	*0*	*1*
m_{15}	*1*	*1*	*1*	*1*	*0*

1.6.2 Maxterms

Maxterm is a sum involving all inputs (or complements) to a function. Every row of a truth table has a corresponding maxterm. Minterm is false if and only if the corresponding row in the table is active.

Maxterms are defined as follows for each row of a two input truth table:

A	B	maxterm	maxterm name
0	0	$A + B$	M_0
0	1	$A + \bar{B}$	M_1
1	0	$\bar{A} + B$	M_2
1	1	$\bar{A} + \bar{B}$	M_3

Consider a two input circuit whose output Y is given by the truth table:

A	B	Y	maxterm	maxterm name
0	0	0	$A + B$	M_0
0	1	1	$A + \bar{B}$	M_1
1	0	0	$\bar{A} + B$	M_2
1	1	1	$\bar{A} + \bar{B}$	M_3

then $Y = (A + B)(\bar{A} + B) = M_0M_2$.

Writing a functional specification in terms of minterms is also called product of sums canonical form.

Example 8. *Write the maxterm M_{11} for 4-input Boolean function with the ordered inputs A, B, C, D.*

Example 9. *Convert the following 4-input truth table into product of maxterms and product of sums canonical form.*

maxterm name	*A*	*B*	*C*	*D*	*f*
M_0	0	0	0	0	0
M_1	0	0	0	1	0
M_2	0	0	1	0	0
M_3	0	0	1	1	1
M_4	0	1	0	0	0
M_5	0	1	0	1	0
M_6	0	1	1	0	0
M_7	0	1	1	1	1
M_8	1	0	0	0	0
M_9	1	0	0	1	0
M_{10}	1	0	1	0	0
M_{11}	1	0	1	1	1
M_{12}	1	1	0	0	0
M_{13}	1	1	0	1	1
M_{14}	1	1	1	0	1
M_{15}	1	1	1	1	0

Problem 6 (10 marks)**.** *Convert the following 4-input truth table into product of maxterms and products of sums canonical form.*

maxterm name	*A*	*B*	*C*	*D*	*f*
M_0	0	0	0	0	0
M_1	0	0	0	1	1
M_2	0	0	1	0	1
M_3	0	0	1	1	1
M_4	0	1	0	0	1
M_5	0	1	0	1	0
M_6	0	1	1	0	1
M_7	0	1	1	1	1
M_8	1	0	0	0	0
M_9	1	0	0	1	1
M_{10}	1	0	1	0	1
M_{11}	1	0	1	1	1
M_{12}	1	1	0	0	0
M_{13}	1	1	0	1	1
M_{14}	1	1	1	0	1
M_{15}	1	1	1	1	0

Example 10. *Write the 3-input truth table for the function* $f = m_2 + m_3 + m_7$.

Problem 7 (10 marks)**.** *Write the 3-input truth table for the function* $f = M_4 M_5 M_7$.

Problem 8 (10 marks). *Write the truth table for the function* $f = \bar{A}B\bar{C} + AB\bar{C}$.

1.7 Karnaugh maps

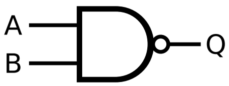
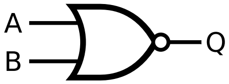
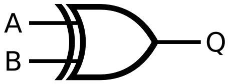
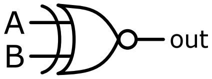
Two input K-maps

Three input K-maps

Four input K-maps

Five input K-maps

1.8 More Gates and notations summary

Name	C/Verilog	Boolean expr.	Truth Table	(ANSI) symbol	K-map
NAND Gate	`Q = ~(x1 & x2)`	$Q = \overline{x_1 \cdot x_2} = \overline{x_1 x_2}$	x_1 x_2 $\overline{x_1 \cdot x_2}$ 0 0 1 0 1 1 1 0 1 1 1 0		B\A 0 1 0 1 1 1 1 0
NOR Gate	`Q = ~(x1 | x2)`	$Q = \overline{x_1 + x_2}$	x_1 x_2 $\overline{x_1 + x_2}$ 0 0 1 0 1 0 1 0 0 1 1 0		B\A 0 1 0 1 0 1 0 0
XOR Gate	`Q = x1 ^ x2`	$Q = x_1 \oplus x_2$	x_1 x_2 $x_1 \oplus x_2$ 0 0 0 0 1 1 1 0 1 1 1 0		B\A 0 1 0 0 1 1 1 0
XNOR Gate	`Q = ~(x1 ^ x2)`	$Q = \overline{x_1 \oplus x_2}$	x_1 x_2 $\overline{x_1 \oplus x_2}$ 0 0 1 0 1 0 1 0 0 1 1 1		B\A 0 1 0 1 0 1 0 1

Verilog HDL

HDL = Hardware description language

HDL does not have functions, instead it has modules
Verilog instead of using "main function"
it uses top-level module

HDL :
1. Verilog HDL
2. VHDL: VHSIC Hardware description language
   VHSIC: Very high speed integrated circuts

Verilog was developed in 1984, IEEE standard in 1995

System Verilog was adopted as IEEE standard in 2008

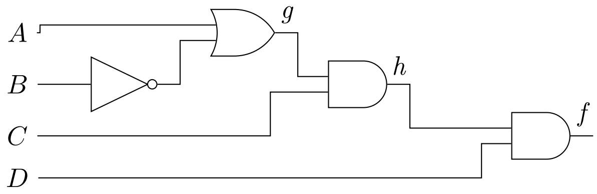
VHDL was developed by DOD in 1981, IEEE standard in 1987

Keywords:
1. module: defines a new module (similar to C functions)

2. input: defines a signal/bus to be input
3. output: ...... to be output
4. logic: this was introduced in system verilog
wire/reg in verilog
5. assign: continuous assignment
Inside always block there are two more kinds
of assignments: blocking and non-blocking. We will
learn about them later

Example 11. *Convert the following Boolean expression into a K-map.* $f = \overline{A\bar{B} + C}D$

Problem 9 (10 marks)**.** *Convert the following logic circuit into a K-map.*



1.9 Boolean Algebra

1.9.1 Axioms of Boolean algebra

1. $0 \cdot 0 = 0$ ✓
2. $1 + 1 = 1$ Diff

3. $1 \cdot 1 = 1$ ✓
4. $0 + 0 = 0$ ✓
5. $0 \cdot 1 = 1 \cdot 0 = 0$ ✓
6. $\bar{0} = 1$ Diff
7. $\bar{1} = 0$
8. $x = 0$ if $x \neq 1$
9. $x = 1$ if $x \neq 0$

$\underline{x \in \{0,1\}}$

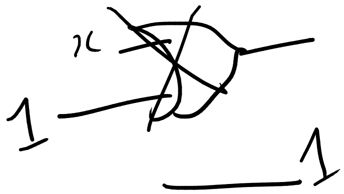
1.9.2 Single variable theorems (Prove by drawing K-maps)

1. $x \cdot 0 = 0$ ✓

2. $\underline{x + 1 = 1}$ Diff

$0+1=1$

$1+1=1$



3. $x \cdot 1 = x$ ✓

4. $x + 0 = x$ ✓

5. $x \cdot x = x$ Diff

x	x	$x \cdot x$	$x+x$
0	0	0	0
1	1	1	1

$2x$

$x+x = x$

x^2

$x \cdot x = x$

6. $x + x = x$ Diff

7. $x \cdot \bar{x} = 0$ Diff

x	$\bar{x}$	$x \cdot \bar{x}$
0	1	0
1	0	0

8. $x + \bar{x} = 1$

x	$\bar{x}$	$x+\bar{x}$
0	1	1
1	0	1

9. $\bar{\bar{x}} = x$

Remark 2 (Duality)**.** *Swap + with · and 0 with 1 to get another theorem*

1.9.3 Two and three variable properties (Prove by K-maps)

1. Commutative: $x \cdot y = y \cdot x$, $x + y = y + x$

2. Associative: $x \cdot (y \cdot z) = (x \cdot y) \cdot z$, $x + (y + z) = (x + y) + z$

3. Distributive: $x \cdot (y + z) = x \cdot y + x \cdot z$, $x + y \cdot z = (x + y) \cdot (y + z)$

4. Absorption: $x + x \cdot y = x$, $x \cdot (x + y) = x$

5. Combining: $x \cdot y + x \cdot \bar{y}$, $(x+y) \cdot (x+\bar{y}) = x$

6. DeMorgan's theorem: $\overline{x \cdot y} = \bar{x} + \bar{y}$, $\overline{x+y} = \bar{x} \cdot \bar{y}$.

7. Concensus:

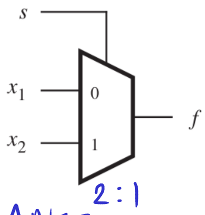
   (a) $x + \bar{x} \cdot y = x + y$

   (b) $x \cdot (\bar{x} + y) = x \cdot y$

   (c) $x \cdot y + y \cdot z + \bar{x} \cdot z = x \cdot y + \bar{x} \cdot z$

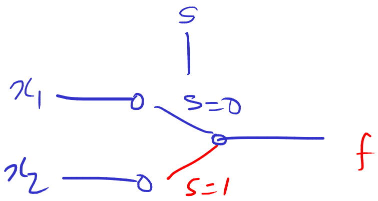
   (d) $(x+y) \cdot (y+z) \cdot (\bar{x}+z) = (x+y) \cdot (\bar{x}+z)$

Example 12 (Multiplexer)**.** *Multiplexer is a circuit used to select one of the input lines x_1 and x_2 based only select input s. When $s = 0$, x_1 is selected, x_2 is selected otherwise. Find a boolean expression and a circuit for multiplexer*



ANSI symbol

s	$f(s, x_1, x_2)$
0	x_1
1	x_2



s	x_1	x_2	f	
0	0	0	0	0
0	0	1	0	1
0	1	0	1	2
0	1	1	1	3
1	0	0	0	4
1	0	1	1	5
1	1	0	0	6
1	1	1	1	7

$f = x_1$ if $s = 0$

$f = x_2$ if $s = 1$

$$f = \underbrace{m_2 + m_3 + m_5 + m_7}$$

Example 13. *Simplify $f = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}B\bar{C}$ using boolean algebra.*

Problem 10 (30 marks, Exercise 2.14 [1])**.** *Simplify the following Boolean equations using Boolean theorems.*

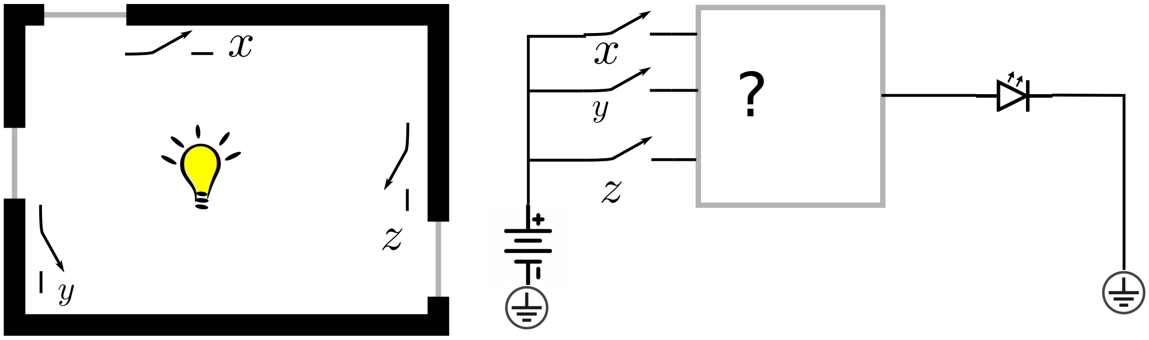
$$Y = \bar{A}BC + \bar{A}B\bar{C} \tag{1.1}$$

$$Y = \overline{ABC} + A\bar{B} \tag{1.2}$$

$$Y = ABC\bar{D} + A\overline{BCD} + (\overline{A + B + C + D}) \tag{1.3}$$

Example 14. *Simplify $f = \bar{A}\bar{A}\bar{C} + \bar{A}\bar{B}C$ using K-maps.*

Example 15. *Assume that a large room has three doors and that a switch near each door controls a light in the room. It has to be possible to turn the light on or off by changing the state of any one of the switches.*



Problem 11 (20 marks, Exercise 2.38 [1])**.** *An M-bit thermometer code for the number k consists of k 1's in the least significant bit positions and M – k 0's in all the more significant bit positions. A binary-to-thermometer code converter has N inputs and* 2^{N-1} *outputs. It produces a* 2^N*–1 bit thermometer code for the number specified by the input. For example, if the input is 110, the output should be 0111111. Design a 3:7 binary-to-thermometer code converter. Give a simplified Boolean equation for each output.*

1.10 Logic minimization

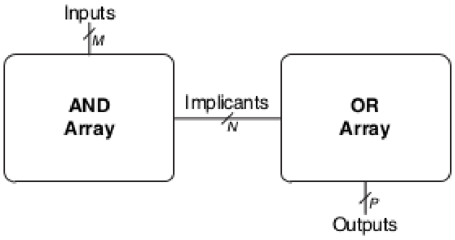
1.11 Logic minimization

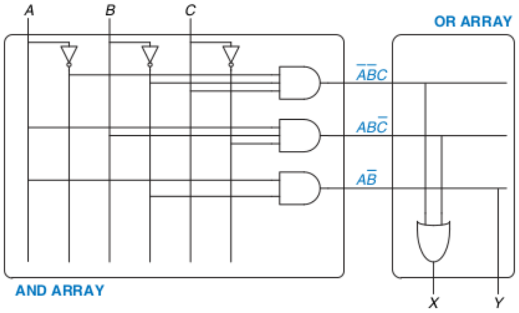
A general optimization criteria for multi-level logic are to Minimize some combination of:

1. Area occupied by the logic gates and interconnect;
2. the Critical Path Delay of the longest path through the logic;
3. the Degree of Testability of the circuit, measured in terms of the percentage of faults covered by a specified set of test vectors, for an appropriate fault model (Eg., single stuck faults, multiple stuck faults, etc.);
4. Power consumed by the logic gates.

In this course, we will start with two-level multi-input circuits and a criteria based on the number of gates/transistors/diodes.

1.12 Programmable Logic Arrays





1.13 Two-level circuits

The cost that we are going to consider in this class depend upon:

1. Number of gates.
2. Number of input to the gates.

More gates need more transistors, more area on the chip. More-inputs the gate need more transistors within each gate. Number of gate inputs can be considered secondary criterion to the number of gates.

Example 16. *Find the cost of the following Boolean expression* $X = \bar{A}\bar{B}C + AB\bar{C} + A\bar{B}$.

Problem 12. *Find the cost of the following Boolean expression* $X = A\bar{B}C + \bar{A}B\bar{C} + \bar{B}C$.

1.14 Terminology for K-maps

Running Example: $f = \sum m(0,1,2,3,7) = \bar{x}_1 + x_1x_2x_3$.

Literal A single variable or its complement. Example: $\bar{x}, x_1, x_2, x_3$

Implicant A product term which is true for a function. All minterms are implicants. Example: $x_1x_2x_3$, $\bar{x}_1$, $m_0 = \bar{x}_1\bar{x}_2\bar{x}_3$, $\bar{x}_1x_3$, $\bar{x}_1\bar{x}_3$.

Prime Implicant An implicant that cannot be combined into fewer literals. Example: $\bar{x}_1, x_2x_3$.

Essential Prime Implicant An implicant that cannot be combined into fewer literals. Example: x_2x_3.

Cover : List of Prime Implicants that account for all $f = 1$.

Cost : Number of gates (excluding not gate on literals) and number of inputs to each gate.

Example 17. *Find minimum cost expression for the function* $f(x_1, x_2, x_3) = \prod M(4,5,6)$

Problem 13. *Find minimum cost expression for the function* $f(x_1, x_2, x_3) = \prod M(2, 5, 6)$

1.14.1 Incompletely specified functions or Don't cares

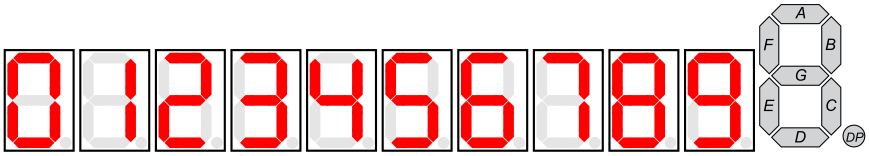


Figure 1.1: 7 Segment Representations of Each Integer

BCD Value				LED Segment
D_3	D_2	D_1	D_0	E
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	d
1	0	1	1	d
1	1	0	0	d
1	1	0	1	d
1	1	1	0	d
1	1	1	1	d

Example 18. *Find minimum cost expression for the function*

$$f(x_1, \ldots, x_4) = \sum m(2, 4, 5, 6, 10) + D(12, 13, 14, 15)$$

Problem 14. *Find minimum cost expression for the function*

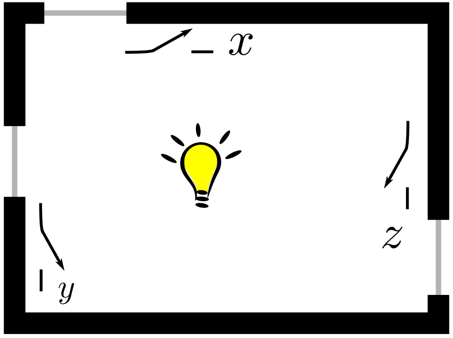
$$f(x_1, \ldots, x_4) = \sum m(0,2,4,6,7,8,9,13) + D(1,12,15)$$

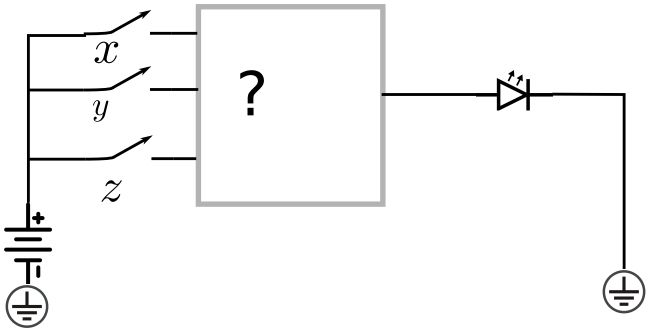
1.15 A few more Boolean problems

Example 19. *Simplify the following Boolean expression:*

$$f = x_1\bar{x}_3\bar{x}_4 + x_2\bar{x}_3\bar{x}_4 + x_1\bar{x}_2\bar{x}_3$$

Example 20. *Assume that a large room has three doors and that a switch near each door controls a light in the room. It has to be possible to turn the light on or off by changing the state of any one of the switches.*





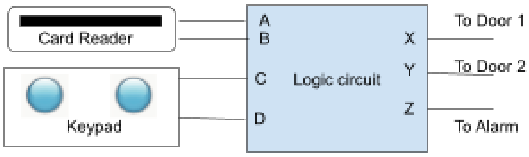
Problem 15. *A simple security system for two doors consists of a card reader and a keypad.*

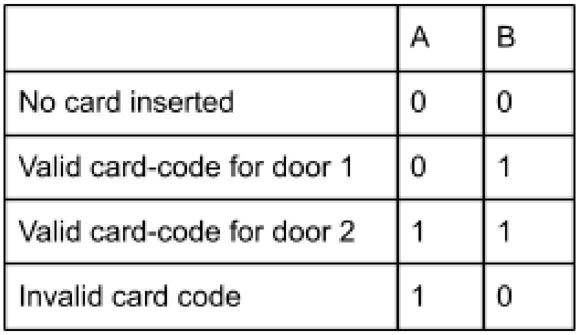
A person may open a particular door if he or she has a card containing the corresponding code and enters an authorized keypad code for that card. Note that card-code and keypad-code are different. The outputs from the card reader are given in the table below.

To unlock a door, a person must hold down the proper keys on the keypad and, then, insert the card in the reader. The authorized keypad code for door 1 is 10, and the authorized keypad code for door 2 is 11. If the card has an invalid code or if the wrong keypad code is entered, the alarm will ring when the card is inserted. If the correct keypad code is entered, the corresponding door will be unlocked when the card is inserted.

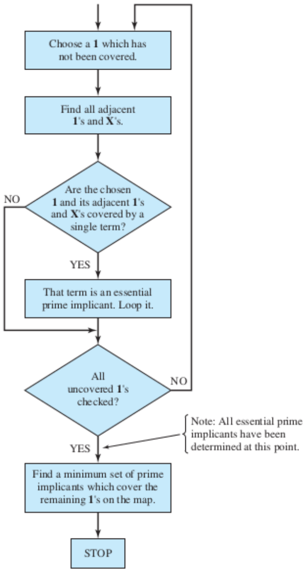
Design the logic circuit for this simple security system. Your circuit's inputs will consist of a card code AB, and a keypad code CD. The circuit will have three outputs XYZ (if X is 1, door 1 will be opened; if Y is 1, door 2 will be opened; if Z 1, the alarm will sound).

Find the minimal cost two-level circuit using K-maps for X, Y, Z. Provide the minimal cost. (It can be either of SOP/POS forms)





	A	B
No card inserted	0	0
Valid card-code for door 1	0	1
Valid card-code for door 2	1	1
Invalid card code	1	0


Choose a 1 which has not been covered.
Find all adjacent 1's and X's.
Are the chosen 1 and its adjacent 1's and X's covered by a single term?
NO
YES
That term is an essential prime implicant. Loop it.
All uncovered 1's checked?
NO
YES
Note: All essential prime implicants have been determined at this point.
Find a minimum set of prime implicants which cover the remaining 1's on the map.
STOP

Example 21.

CD \ AB	00	01	11	10
00	d	1		1
01		1	1	1
11		d1	d	
10		1		1

Problem 16. *Find the minimum SOP (sum of products) and POS (product of sum) expression for the function* $f(a,b,c,d) = \prod M(5,7,13,14,15) \cdot \prod D(1,2,3,9)$

Problem 17. *Read Chapter 2 up to Section 2.7 of Harris and Harris textbook. Write a statement saying that you have read and understood the chapter. [5 marks]*

Problem 18. *If the Sum of Products (SOP) form for* $\bar{f} = AB\bar{C} + \bar{A}\bar{B}$*, then give the Product of Sums (POS) form for* f*. [10 marks]*

Problem 19. *Use DeMorgan's Theorem to find* f *if* $\bar{f} = (A + \bar{B}C)D + EF$*. [10 marks]*

Row	x_1	x_2	x_3	f
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

Table 1.1: Truth table for a 3-way light switch

Problem 20. *For the function* $f = AB\bar{C} + BD$,

1. *Write the Truth table. [10 marks]*
2. *Write* f *in Sum of Products form. [10 marks]*
3. *Write* f *in canonical minterm form. [10 marks]*
4. *Write* f *as Product of Sums. [10 marks]*
5. *Write* f *in canonical maxterm form. [10 marks]*

Problem 21. *Implement the function in Table 1.1 using only NAND gates. [10 marks]*

Problem 22. *Implement the function in Table 1.1 using only NOR gates. [10 marks]*

Problem 23. *Find the minimum-cost Sum of Products (SOP) and Product of Sums (POS) forms for the function* $f(x_1, x_2, x_3) = m(1, 3, 4, 5)$. *Chose the minimum-cost expression by comparing Product of Sums (POS) and Sum of Products (SOP) forms. [10 marks]*

Problem 24. *Find the minimum-cost Sum of Products (SOP) and Product of Sums (POS) forms for the function* $f(x_1, x_2, x_3) = \sum m(1, 5, 7) + D(2, 4)$. *[10 marks]*

Problem 25. *Find the minimum-cost Sum of Products (SOP) and Product of Sums (POS) forms for the function* $f(x_1, x_2, x_3, x_4) = \prod M(1, 2, 4, 5, 7, 8, 9, 10, 12, 14, 15)$. *Chose the minimum-cost expression by comparing Product of Sums (POS) and Sum of Products (SOP) forms. [10 marks]*

Problem 26. *Find the minimum-cost Sum of Products (SOP) and Product of Sums (POS) forms for the function* $f(x_1, x_2, x_3, x_4) = \sum m(2, 8, 9, 12, 15) + D(1, 3, 6, 7)$. *Chose the minimum-cost expression by comparing Product of Sums (POS) and Sum of Products (SOP) forms. [10 marks]*

Problem 27. *Derive a minimum-cost realization of the four-variable function that is equal to 1 if exactly two or exactly three of its variables are equal to 1; otherwise it is equal to 0. [10 marks]*

Problem 28. *Find the minimum-cost Sum of Products (SOP) and Product of Sums (POS) forms for the function* $f(x_1, \ldots, x_5) = \sum m(1, 3, 4, 6, 8, 9, 11, 13, 14, 16, 19, 20, 21, 22, 24, 25) + D(5, 7, 12, 15, 17, 23)$. *Chose the minimum-cost expression by comparing Product of Sums (POS) and Sum of Products (SOP) forms. [10 marks]*

Chapter 2

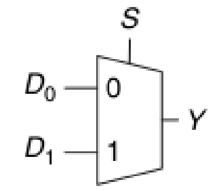
Adders, Muxes and Decoders

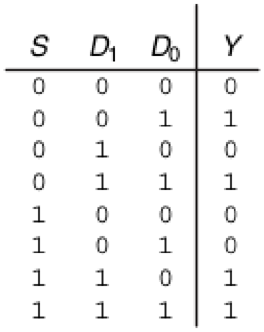
2.1 Objectives

1. Design combinational circuits using multiplexers and decoders

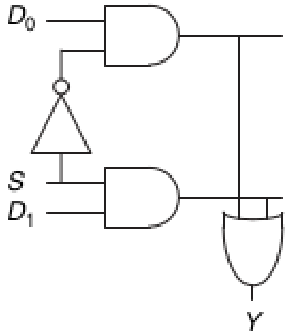
2.2 Design combinational circuit using multiplexers [1, Section 2.8.1]

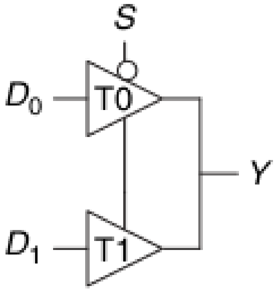
2.2.1 Review: 2to1 Multiplexer (MUX)





S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1





$Y = D_0\overline{S} + D_1 S$

2.2.2 Wider multiplexers

Draw the symbol for a 4:1 MUX, an 8:1 MUX and a $2^N : 1$ MUX and write corresponding Boolean expressions.

Example 22. *Design a circuit for* $Y = A\bar{B} + \bar{B}\bar{C} + \bar{A}BC$ *using a 8:1 MUX.*

Remark 3. *A* $2^N : 1$ *MUX can be used to program any N-input logic function.*

Example 23. *Design a circuit for* $Y = A\bar{B} + \bar{B}\bar{C} + \bar{A}BC$ *using a 4:1 MUX and NOT gates only.*

Remark 4. *A* $2^{N-1} : 1$ *MUX can be used to program any N-input logic function, if we use literals on the input side.*

Example 24. *Design a circuit for* $Y = \bar{A}C + \bar{A}B + B\bar{D}$ *using a 8:1 MUX and NOT gates only. Also design using 4:1 MUX and other gates. fewest gates.*

2.3 Encoders and Decoders

Example 25. *Draw the symbol and the truth table for 2:4 decoder. Also write the logic expressions.*

Example 26. *Draw the symbol and the truth table for 3:8 decoder, 4:16 decoder and* $N : 2^N$ *decoder. Also write the logic expressions.*

Example 27. *Design a circuit for a XOR gate using a 2:4 decoder and an OR gate.*

Example 28. *Design a circuit for* $Y = A\bar{B} + \bar{B}\bar{C} + \bar{A}BC$ *using a 3:8 decoder and an OR gate.*

2.3.1 (Priority) Encoders

Example 29. *Draw symbol and truth table for a 4:2 priority encoder.*

Example 30. *Draw symbol and truth table for a 8:3 priority encoder.*

Chapter 3

Number System

3.1 Place value number system

- What is a place value number system?

- What are some examples?

- What are some non-examples?

- What is a radix (or base)?

- How to convert between different radix in place value system?

- What are some commonly used number systems in computer engineering?

3.1.1 Decimal number system

3.1.2 Binary numbers

3.1.3 Conversion between different radix

Problem 29. *Convert the following binary numbers to decimal:* $(11110)_2$*,* $(100111)_2$*.*

Conversion from decimal to binary The value is in decimal because we find it easy to do calculations in decimal numbers. Decimal values can be converted back to Binary representation by *repeated division* by 2 while noting down the remainder. Allow me to use / sign to denote both quotient and remainder after division. Let's convert $(22)_{10}$ back to binary:

$$\begin{aligned}
22/2 &= (11, 0) && \text{11 is the quotient and 0 is the remainder}\\
11/2 &= (5, 1) && \text{5 is the quotient and 1 is the remainder}\\
5/2 &= (2, 1)\\
2/2 &= (1, 0)\\
1/2 &= (0, 1)
\end{aligned}$$

Read the remainders from bottom to top and right them as left to right, to form the resultant binary number $(22)_{10} = (10110)_2$.

Problem 30. *Find the binary representation for decimal numbers:* 123 *and* 89*. Show your work.*

3.2 Hexadecimal numbers

Numbers with base 16 are called Hexadecimal numbers. From 0 to 9 the symbols are same as decimal numbers. From 10 to 15, Hexadecimal numbers use A to F.

$$A = 10, B = 11, C = 12, D = 13, E = 14, F = 15$$

. Example, $(10AD)_{16} = 1 \times 16^3 + 10 \times 16^1 + 13 = 4096 + 160 + 13 = 4269$.

3.3 Octal numbers

Numbers with base 8 are called octal numbers. Example, $(354)_8 = 3 \times 8^2 + 5 \times 8 + 4 = 192 + 40 + 4 = 236$.

3.4 Hexadecimal/octal to binary and vice-versa

Normally, if you have to convert between a number of base r_1 to a number of base r_2, we will have to convert it via decimal numbers. Convert from base r_1 to decimal and then from decimal to r_2.

Since Hexadecimal base 16 is an exact power of 2 ($16 = 2^4$). Conversion between Hexadecimal to binary is easy. You can group 4 binary digits from right to left and convert each group of 4 binary digits to a single Hexadecimal digit and back. Example, $(10110)_2 = (0001\_0110)_2 = (16)_{16}$. To convert back. Take example, $(10AD)_{16} = (0001\_0000\_1010\_1101)_2 = (1\_0000\_1010\_1101)_2$.

Problem 31. *Find the binary and decimal values of the following Hexadecimal numbers* $(A25F)_{16}$*,* $(F0F0)_{16}$*.*

Similarly octal to binary can proceed by grouping 3-binary digits at a time. Example, $(354)_8 = (011\_101\_100)_2$.

Problem 32. *Find the binary and decimal values of the following Octal numbers* $(3751)_8$ *and* $(722)_8$*.*

3.5 Signed binary numbers

Signed numbers include both negative and positive numbers. There three common signed number representations

1. Sign magnitude representation

2. One's complement

3. Two's complement

3.5.1 Sign-magnitude representation

The Most significant (left most) *bit* (binary digit) represents sign ($0 = +$ and $1 = -$), the rest represent the magnitude. Example, a 5-bit number $(11010)_2$ in signed magnitude representation has the value of $(-1010)_2 = -10$. Note that $+10$ has to be represented by a leading 0 at the most significant bit (MSB) $+10 = (01010)_2$. Hence, the number of bits have to be specified.

Problem 33.

- *Write down all possible 4-digit binary numbers and corresponding decimal values if they are in signed magnitude format? What is the minimum and maximum value?*

- *What is the minimum and maximum value of n-digit signed binary number in sign-magnitude format?*

3.5.2 One's complement negation

You can convert a positive number (say +10) to negative number by applying a negative sign in front of it (-(+10) = -10). It is more evident from taking negative of a negative number (-(-10) = +10). In case of sign-magnitude representation, the "negative operator" flips the sign bit. The next two signed number representations (1's complement and 2's complement) are designed around specific negative operator definitions.

Negate $13_{10} = 01101_2$ using 5-bit one's complement.

Negate -13_{10} using 5-bit one's complement.

3.5.3 One's complement binary numbers

In one's complement representation, the negative operation is obtained by flipping all the bits of the binary number. Example, a 5-bit one's complement of $+10 = (01010)_2$ is $(10101)_2 = -10$. Note that flipping bits is equivalent to subtracting the number from $(11111)_2$, hence the name. You can also confirm that double negative operator yields back the same number.

Problem 34.

- *Write down all possible 4-digit binary numbers and corresponding decimal values if they are in sign magnitude format? What is the minimum and maximum value?*

- *What is the minimum and maximum value of n-digit signed binary number in one's complement?*

Problem 35. *Determine the decimal values of the following 1's complement 6-digit binary numbers :*

1. *01101110*

2. *10101101*

Problem 36. *Convert the decimal numbers -17 and +23 into the 6-digit one's complement binary numbers and try adding them. What adjustments will you need to make to get the right result's (23-17=6) in binary representation.*

3.5.4 Two's complement negation

In two's complement representation, the n-digit negative number is obtained by subtracting the positive number from 2^n. Example, two's complement of 5-digit binary number $+10 = (01010)_2$ is $2^5 - 10 = 22 = (11000)_2$. An easier algorithm to get two's complement goes via one's complement. Note that $(11111)_2 = 2^5 - 1$. We can get two's complement by adding 1 to one's complement. To get two's complement:

1. Flip all the bits. (Same as taking one's complement).
2. Add 1 to the number.

Negate $13_{10} = 01101_2$ using 5-bit two's complement.

Negate -13_{10} using 5-bit two's complement.

How to convert one's complement number representation into sign-magnitude numbers?

1. Check if the number is positive or negative. Even for one's complement representation, or two's complement representation, if the MSB (Most-significant bit) is 1, then the number is negative, otherwise positive.

2. If positive: For positive numbers, two's complement, one's complement and sign magnitude are the same. No conversion between different representation is needed. 2.b If negative: For negative numbers. Flip the bits of 1's complement. Once you flip the 1's complement bits of a negative number, you get the corresponding positive number.

3. We still want to represent the original negative number. So we set the MSB of sign-magnitude representation to 1. Since the range (min and max) for both n-bit 1's complement and sign-magnitude are the same (between $-(2^{n-1}-1)$ and $2^{n-1}-1$), you can always represent 8-bit 1's complement numbers with needing to extend the 8-bit number to 9-bits.

Example: Convert 8-bit one's complement 10101010 to 8-bit sign-magnitude Let number n = 10101010

1. Is the number +ve or -ve: It is negative because it starts with 1.

2. The number is not positive.

3. Take the 1's complement of the negative number to get the positive part. i.e. Flip the bits: -n = 01010101 or n = -(01010101)

4. We got the positive part of the number, but we want to represent the original negative number, so we set the MSB bit one. Hence, the equivalent sign-magnitude representation is:
   n = 11010101

3.5.5 Two's complement representation

Problem 37. *Determine the decimal values of the following 2's complement 6-digit numbers :*

1. *01011110*

2. *10010111*

Problem 38. *Convert the decimal numbers -17 and +23 into the 6-digit two's complement binary numbers and try adding them. What adjustments will you need to make to get the right result's (23-17=6) in binary representation.*

Problem 39. *Convert the decimal numbers 73, 23, -17, and -163 into signed 8-bit numbers in the following representations:*

1. *Sign and magnitude*
2. *1's complement*
3. *2's complement*

3.5.6 Arithmetic overflow

Problem 40. *Consider addition of 4-digit two's complement binary numbers*

1. $1010_2 + 1101_2$
2. $1011_2 + 1100_2$

In which of the two case overflow happens? Can you come up with a rule to "easily" detect overflow?

Rules for detecting arithmetic overflow:

1. Adding numbers of different signs never produces an overflow.

2. Adding numbers of the same sign may produce an overflow

   (a) Wrong approach: Adding two negative 2's complement numbers always produces an additional carry-over 1, but that in itself isn't an overflow. An example, the range of 4-bit 2's complement numbers is between -8 to +7. Adding -3 to -4 in 2's complement is 1101 + 1100 produces an additional carry over 1. You can ignore the additional carry-over 1 to get the correct answer 1001 = -7 which is within range -8 to 7.

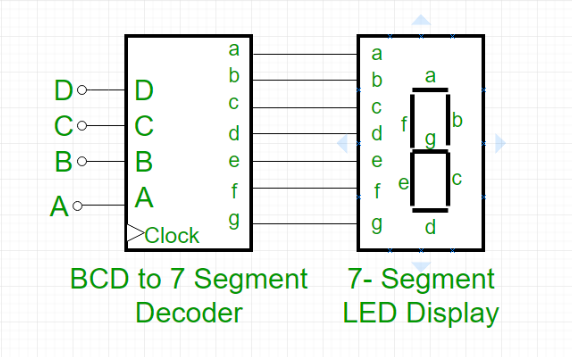
   (b) Approach 1: The easiest way for now to detect overflow is if adding two -ve numbers results in a +ve number, or adding +ve numbers results in a -ve number.

   (c) Approach 2: You can also do a range test in decimal based range test. The range of n-bit 2's complement numbers is between -2^{n-1} and $2^{n-1}-1$. For 5-bit 2's complement numbers, it is between -16 and 15. For 6-bit 2's complement numbers, it is between -32 and 31.

   (d) Approach 3: You can also check the carry-overs of the most significant two bits. If they match, i.e. 0 and 0, or 1 and 1, then there is no overflow. If they do not match, i.e. 0 and 1 or 1 and 0, then there is an overflow.

3.6 Binary coded decimal

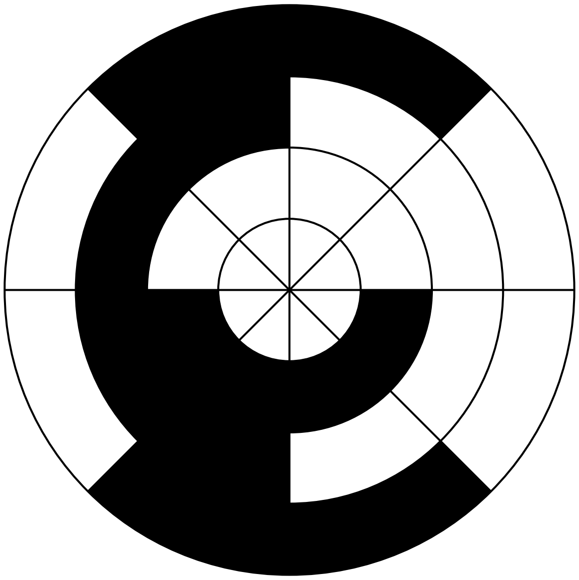
In Binary coded decimal (BCD), each decimal digit is represented by 4 bits. For example, 1047 = $(0001\_0000\_0100\_0111)_{\mathrm{BCD}}$. It is useful in input-output applications where the number has to be either displayed as decimal or received as decimal.



Problem 41. *Convert 11, 23, 35, 57 and 103897 to BCD?*

3.7 Gray code

A sequence of binary numbers where only one bit changes when the number increases by 1. It is helpful in applications like wheel encoders



Problem 42. *Write all possible 3-bit binary numbers in gray-code*

Always show your work/process. Correct final answer is worth less than the correct process. Submit digitally via brightspace.

Problem 43. *Convert the each of the following numbers into binary, decimal, hexadecimal, octal numbers. Show your work. Just filling in the values is not enough. (8 × 6 marks)*

	Binary	*Decimal*	*Hexadecimal*	*Octal*
a)	1010_2			
b)	10_0110_2			
c)		329_{10}		
d)		741_{10}		
e)			$7D_{16}$	
f)			$EC3A_{16}$	
g)				351_8
h)				2563_8

Problem 44. *Convert the each of the following numbers into decimal, 8-bit sign-magnitude binary, 8-bit one's complement binary and 8-bit two's complement binary. Show your work. (6 × 6 marks)*

	Decimal	*Sign-magnitude*	*One's complement*	*Two's complement*
a)	-79_{10}			
b)	-110_{10}			
c)				0110_1110_2
d)				1011_1101_2
e)			0110_1101_2	
f)			1001_1010_2	

Problem 45. *Convert the decimal numbers to 6-bit two's complement binary and then add them. Check if the addition causes overflow (3×6 marks).*

1. $-16_{10} - 7_{10}$
2. $19_{10} - 5_{10}$
3. $-4_{10} - 29_{10}$

Problem 46. 1. *Convert* 299_{10} *to binary coded decimal (BCD). (2 marks)*

2. *Convert* $1001\_0111\_0101_{BCD}$ *to decimal. (2 marks)*
3. *Convert* 0110_1101_{BCD} *to binary. (4 marks)*

Bibliography

[1] Sarah L Harris and David Harris. *Digital design and computer architecture*. Morgan Kaufmann, 2022.

[2] Brown Stephen and Vranesic Zvonko. *Fundamentals of digital Logic with Verilog design*. McGraw Hill, 2022.