

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 2.1: Truth table for a 3-way light switch

**Problem 2.10** (10 marks). Find the minimum-cost Sum of Products (SOP) and Product of Sums (POS) forms for the function  $f(x_1, x_2, x_3) = \sum m(1, 5, 7) + D(2, 4)$ .

**Problem 2.11** (10 marks). 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.

**Problem 2.12** (10 marks). 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.

# Chapter 3

## Review

### 3.1 Syllabus covered

- ✓ Binary numbers
- ✓ Generate minterms, maxterms, SOP canonical form and POS canonical forms and convert between them
- ✓ Understand and use the laws and theorems of Boolean Algebra
- ✓ Perform algebraic simplification using Boolean algebra
- ✓ Simplification using K-maps
- ✓ Derive sum of product and product of sums expressions for a combinational circuit
- ✓ Convert combinational logic to NAND-NAND and NOR-NOR forms
- Hexadecimal, Sign-magnitude, One's-complement and Two's complement. Conversions between them.
- Design combinational circuits for positive and negative logic
- Design Hazard-free two level circuits and understand Hazards in multi-level circuits
- Compute noise margin of one device
- Describe how tri-state and open-collector outputs are different from totem-pole outputs.
- Different between and limitations of master-slave and edge-triggered flip-flops.
- Compute fan out and noise margin of one device driving the same time
- Know the differences and similarities between PAL, PLA, and ROMs and can use each for logic design
- Design combinational circuits using multiplexers and decoders
- Analyze a sequential circuit and derive a state-table and a state-graph
- Understand the difference between synchronous and asynchronous inputs
- Derive a state graph or state table from a word description of the problem

- Reduce the number of states in a state table using row reduction and implication tables
- Perform a state assignment using the guideline method
- Implement a design using JK, SR, D or T flip-flops
- Analyse and design both Mealy and Moore sequential circuits with multiple inputs and multiple outputs
- Convert between Mealy and Moore designs
- Partition a system into multiple state machines

### **3.1.1 Labs (not questioned in exams)**

- Use computer tools to enter designs graphically and HDL
- Simulate designs using computer tools
- Use computer tools to program gate arrays logic and debug and test

# Chapter 4

## Sample midterm exam

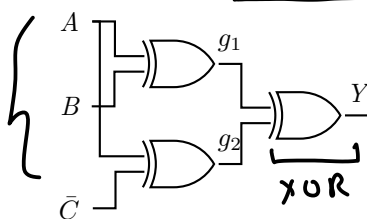
Student Name:

Student Email:

### 4.1 Instructions

- Time allowed is 50 minutes. (This sample exam might be lengthier than the actual exam. Same instruction will apply to the midterm.)
- In order to minimize distraction to your fellow students, you may not leave during the last 10 minutes of the examination.
- The examination is closed-book. One 8x11in cheatsheet is allowed.
- Non-programmable calculators are permitted.
- The maximum number of marks is 80, as indicated; the midterm examination amounts 10% toward the final grade.
- Please colored pen/pencils for K-maps, use a pen or heavy pencil to ensure legibility.
- Please show your work; where appropriate, marks will be awarded for proper and well-reasoned explanations.

**Problem 4.1.** Consider the circuit below



$$Y = g_1 \oplus g_2$$

$$= g_1 \bar{g}_2 + \bar{g}_1 g_2$$

$$Y \text{ XOR } = (\bar{A}B + A\bar{B})(\bar{A}\bar{C} + AC) + (A\bar{B} + \bar{A}B)(\bar{A}\bar{C} + AC)$$

A	B	g <sub>1</sub>
0	0	0
0	1	1
1	0	1
1	1	0

$$g_1 = A \oplus B$$

$$g_2 = A \oplus \bar{C}$$

$$g_1 = m_1 + m_2 = \bar{A}B + A\bar{B}$$

$$g_2 = \bar{A}\bar{C} + A\bar{C} = \bar{A}\bar{C} + AC$$

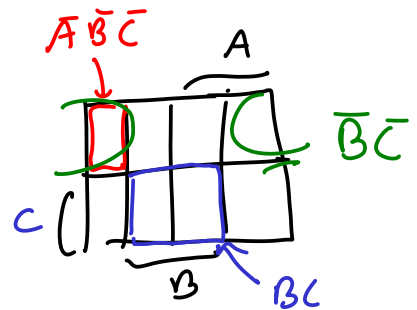
By algebraic manipulation, prove or disprove that  $Y = \bar{B}C + BC$  (10 marks).

**Problem 4.2.** Use the following 4-variable K-map for  $F(A, B, C, D)$ , and find a minimal sum of products (SOP) expression for  $F$  (15 marks)

$$Y = (\bar{A}B + A\bar{B}) \overline{(\bar{A}\bar{C} + AC)} + (\bar{A}\bar{B} + A\bar{A}) (\bar{A}\bar{C} + AC)$$

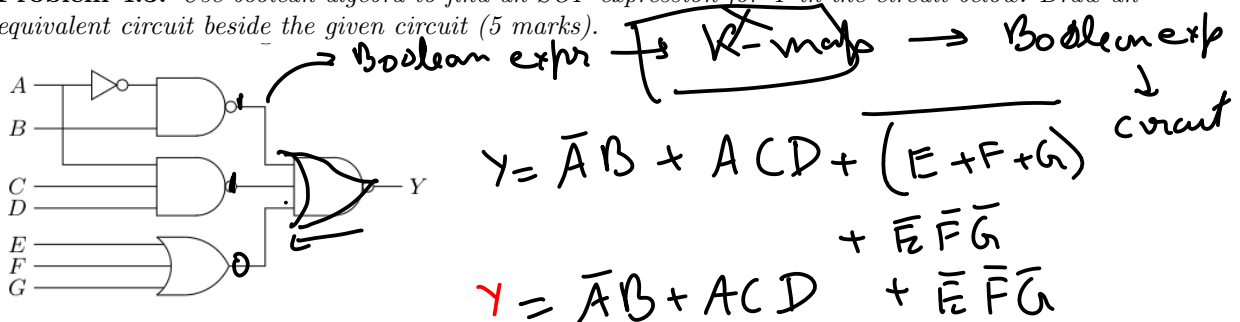
Whenever you see a big not over an expression, apply DeMorgan's theorem.

$$\begin{aligned}
 &= (\bar{A}B + A\bar{B}) ((A+C) \cdot (\bar{A} + \bar{C})) + ((\bar{A} + \bar{B}) \cdot (A + \bar{B})) (\bar{A}\bar{C} + AC) \\
 &= (\bar{A}B + A\bar{B}) (\cancel{A}A + \cancel{C}\bar{A} + \bar{A}\bar{C} + \cancel{C}\bar{C}) + (\bar{A}\cancel{A} + \bar{B}A + \bar{A}\bar{B} + \bar{B}\bar{C}) \cdot (\bar{A}\bar{C} + AC) \\
 &= \bar{A}B(\bar{C}\bar{A} + \bar{A}\bar{C}) + A\bar{B}(\bar{C}\bar{A} + \bar{A}\bar{C}) + \bar{B}A(\bar{A}\bar{C} + AC) + \bar{A}\bar{B}(\bar{A}\bar{C} + AC) \\
 &= \underbrace{\bar{A}B\bar{C}}_{(1)} + \underbrace{\bar{A}A\bar{B}\bar{C}}_{(2)} + \underbrace{A\bar{A}\bar{B}\bar{C}}_{(3)} + \underbrace{A\bar{B}\bar{C}}_{(4)} + \underbrace{A\bar{A}B\bar{C}}_{(5)} + \underbrace{A\bar{B}C}_{(2)} + \underbrace{\bar{A}\bar{B}\bar{C}}_{(4)} + \underbrace{\bar{A}A\bar{B}C}_{(1)} \\
 &= (\bar{A} + A) \bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{B}\bar{C} \\
 &= 1 \cdot \bar{B}\bar{C} + (A + \bar{A}) \bar{B}\bar{C} \\
 &= \bar{B}\bar{C} + \bar{B}\bar{C} \\
 &= \text{RHS} \quad \text{Hence Proved}
 \end{aligned}$$



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

**Problem 4.3.** Use boolean algebra to find an SOP expression for  $Y$  in the circuit below. Draw an equivalent circuit beside the given circuit (5 marks).



**Problem 4.4.** Consider the function  $Y$  given below.

$$Y(A, B, C, D) = \sum m(0, 3, 5, 7, 8, 14) + d(2, 12, 15)$$

1. Draw a K-maps to derive a minimum SOP and POS expressions for  $Y$ . Indicate all essential prime implicants for  $Y$  or  $\bar{Y}$  in your K-maps (20 marks).
2. Sketch a two-level NOR-NOR circuit for  $Y$ . Assume that  $A$ ,  $B$ ,  $C$ , and  $D$  are available in true and complimentary forms (5 marks).
3. Write  $Y$  in Product of sums (POS) canonical form (5 marks).

**Problem 4.5.** Design a minimal SOP circuit to add two two-bit unsigned numbers. Denote the two bits of first number as  $A_1A_0$  and the two bits of second number as  $B_1B_0$ . The result will be a 2-bit sum  $S_1S_0$  and a carry  $C$ . Start with filling out the following truth table (3 example rows are provided) and then use K-maps to find minimal SOP for  $S_1$ ,  $S_0$  and a single carry bit  $C_1$  (20 marks).

$$Y(A, B, C, D) = \sum m(0, 3, 5, 7, 8, 14) + d(2, 12, 15)$$

$\nwarrow$  minterm       $\nwarrow$  don't cares

Prime implicants  $\equiv$  Biggest groups of 1's (and d's)

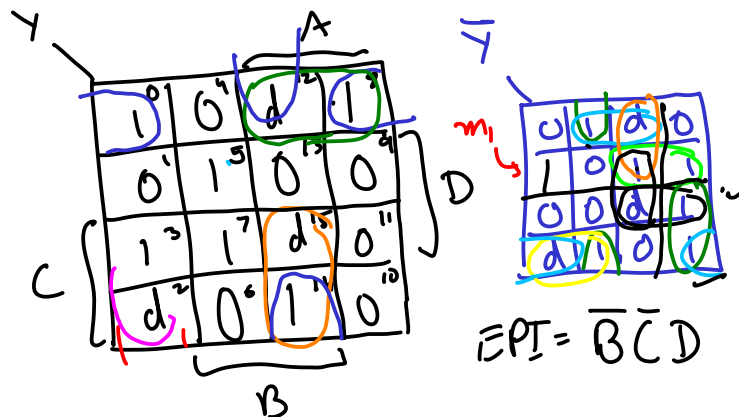
Essential prime implicants

$\equiv$  Prime implicants that uniquely cover some 1's

$$= \bar{A} B D$$

$$Y = \underbrace{\bar{A} B D}_{m_5 + m_7} + \underbrace{\bar{A} \bar{B} C}_{m_3 + m_2} + \underbrace{\bar{B} \bar{C} \bar{D}}_{m_0 + m_8} + \underbrace{A B C}_{m_{14} + m_{15}}$$

$$\bar{Y} = \underbrace{\bar{B} \bar{C} D}_{m_1 + m_9} + \underbrace{A B \bar{C}}_{m_{13} + m_{12}} + \underbrace{A \bar{B} C}_{m_{10} + m_{11}} + \underbrace{\bar{A} B \bar{D}}_{m_6 + m_4}$$



} cover all 1's  
not all d's

$A_1$	$A_0$	$B_1$	$B_0$	$C_1$	$S_1$	$S_0$
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1	0	1	0
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1	1	0	0
1	1	1	0			
1	1	1	1	1	1	0



# Chapter 5

## Number System

$10^3 \quad 10^2 \quad 10^1 \quad 10^0$   
 $\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$   
 $1 \quad 2 \quad 1 \quad 0$   
 $\sqrt{111}$

### 5.1 Place value number system

- What is a place value number system?
- What are some examples?
- What are some non-examples? Roman numeral
- What is a radix (or base)?  $23_{10}$   $11011_2$  ← radix / base
- How to convert between different radix in place value system? decimal ↔ binary
- What are some commonly used number systems in computer engineering?

	base	symbols
binary	2	0/1
decimal	10	0...9
octal	8	0...7
Hexadecimal	16	0.....9, A, B, C, D, E, F
Base 64 encoding	64	$10 + 26 + 26 + \underline{2}$ add'n symbols

:)  
 (: }  
 :- }  
 :-/ }

Binary Representation	Decimal Value
$10111_2$	$23_{10}$
$23_{10}$	

multiply + add

int 23

← Repeated division

$$\begin{array}{r}
 1 \quad 0 \quad 1 \quad 1 \quad 1 \\
 \times_4 \quad \times_3 \quad \times_2 \quad \times_1 \quad \times_0 \\
 2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0 \\
 \hline
 16 + 0 + 4 + 2 + 1 = 23
 \end{array}$$

Binary → Decimal → Base 3<sup>30</sup>  
 $10111_2 \rightarrow 23_{10} \rightarrow 212_3$

$$\begin{array}{r|rr}
 3 & 23 & \\
 \hline
 3 & 7 & 2 \\
 3 & 2 & 1 \\
 \hline
 & 0 & 2
 \end{array}$$

### 5.1.1 Decimal number system

### 5.1.2 Binary numbers

### 5.1.3 Conversion between different radix

**Example 5.1.** 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:

$$22/2 = (11, 0)$$

11 is the quotient and 0 is the remainder

$$11/2 = (5, 1)$$

5 is the quotient and 1 is the remainder

$$5/2 = (2, 1)$$

$$2/2 = (1, 0)$$

$$1/2 = (0, 1)$$

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

**Example 5.2.** Find the binary representation for decimal numbers: 123 and 89. Show your work.

For any radix/base that is a power of 2, for example octal has base  $2^3 = 8$  and Hexadecimal has base  $2^4 = 16$ , you do not need repeated division for conversion from binary to HexD and vice-versa

$$(10AD)_{16} = 0010\_0000\_1010\_1101_{16}$$

$$1101011_2 = 0110\_1011_2 = 6B_{16}$$

## 5.2 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,  $(10AD)_{16} = 1 \times 16^3 + 10 \times 16^1 + 13 = 4096 + 160 + 13 = 4269$ .

## 5.3 Octal numbers

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

$$\begin{array}{c} \downarrow \\ 011\_101\_100_2 \end{array}$$

## 5.4 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 5.1.** 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 5.2.** Find the binary and decimal values of the following Octal numbers  $(3751)_8$  and  $(722)_8$ .

## 5.5 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

$$0 \leftarrow +13 \quad 1 \leftarrow -13$$

$$\underline{01101}_2 \quad \underline{11101}_2 \neq 29_{10}?$$

$$\uparrow \quad \underline{00001}_2 = +1 \quad \underline{10001}_2 = -1$$

### 5.5.1 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 5.3.** • 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?

8-bit sign-magnitude representation  
What range of numbers can it represent?

8-bit unsigned representation

$$\underline{1111111}_2 \quad \text{to} \quad \underline{0111111}_2$$

$$-127_{10} \quad \quad \quad +127_{10}$$

255 numbers

$$\underline{00000000}_2 \quad \text{to} \quad \underline{11111111}_2$$

$$0_{10} \quad \text{to} \quad +255_{10}$$

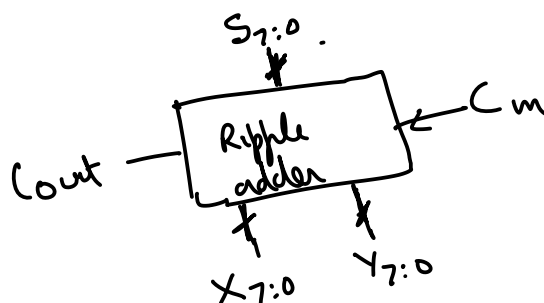
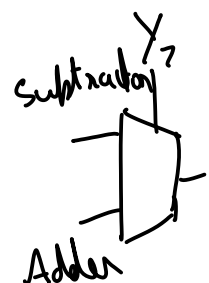
256 numbers

$$-0_{10} = \underline{10000000}_2 = +0_{10} = \underline{00000000}_2$$

### 5.5.2 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.



$\times$  = Bus  
= multiple wires

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

$$+13_{10} = 01101_2$$

$$\begin{array}{r} 11111_2 \\ 01101_2 \\ \hline 10010 \end{array}$$

In 1's complement

$$+13_{10} = 01101_2$$

$$-13_{10} = 10010_2$$

$$\rightarrow 11111_2 = 0_{10}$$

### 5.5.3 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 5.4.** • Write down all possible <sup>but</sup> 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?

sign magnitude

1's complement

$$\begin{array}{l} 0000 \\ 0001 \\ 0010 \\ \vdots \end{array}$$

$$\begin{array}{l} 0 \\ 1 \\ 2 \\ \vdots \end{array}$$

$$\begin{array}{l} 0 \\ 1 \\ 2 \\ \vdots \end{array}$$

$$\begin{array}{l} 0111 \\ 1000 \\ 1001 \\ \vdots \end{array}$$

$$\begin{array}{l} 7 \\ -0 \\ -1 \\ \vdots \end{array}$$

$$\begin{array}{l} 7 \\ -7 \\ -6 \\ \vdots \end{array}$$

$$1111$$

$$-7$$

$$-0$$

$$7 + (-6) = \begin{array}{r} 0111 \\ 1001 \\ \hline 1000 \\ 1111 \\ 0111 \\ 1001 \\ \hline 0001_2 = +1_{10} \end{array}$$

$$-6 = (1111_2 + 1) - 0110_2$$

$$= 10000_2 - 0110_2$$

2's complement

Addition of numbers of opposite sign requires ✓ and addition of +1

**Problem 5.5.** Determine the decimal values of the following 1's complement 6-digit binary numbers

1.  $01101110$

2.  $10101101$

**Problem 5.6.** *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.*

#### 5.5.4 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:  
 $n = 11010101$

### 5.5.5 Two's complement representation

**Problem 5.7.** Determine the decimal values of the following 2's complement 6-digit numbers :

1. 01011110
2. 10010111

**Problem 5.8.** *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 5.9.** *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*

### 5.5.6 Arithmetic overflow

**Problem 5.10.** *Consider addition of 4-digit two's complement binary numbers*

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



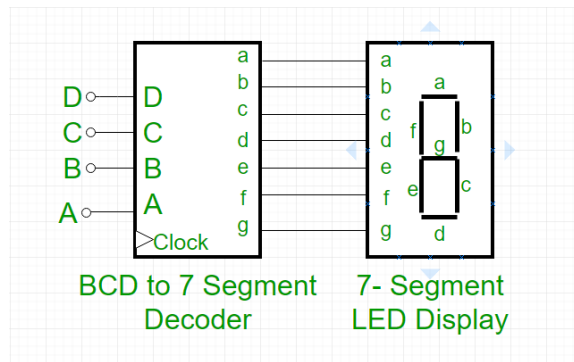
*In which of the two case overflow happens? Can you come up with a rule to “easily” detect overflow?*

**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.

## 5.6 Binary coded decimal

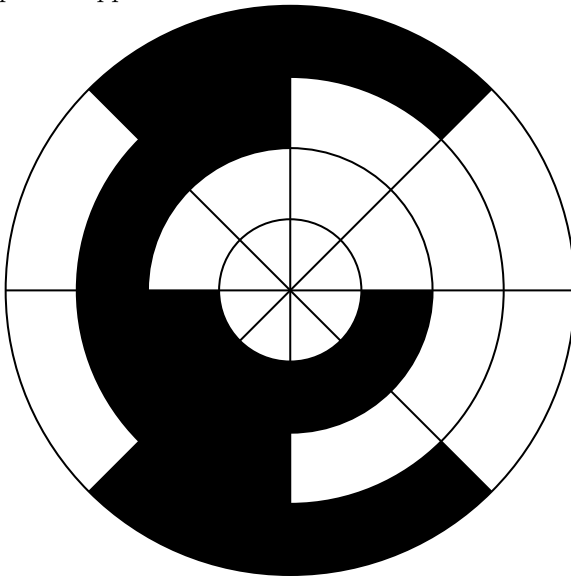
In Binary coded decimal (BCD), each decimal digit is represented by 4 bits. For example,  $1047 = (0001\_0000\_0100\_0111)_{BCD}$ . It is useful in input-output applications where the number has to be either displayed as decimal or received as decimal.



**Problem 5.11.** Convert 11, 23, 35, 57 and 103897 to BCD?

## 5.7 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 5.12.** 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 5.13.** Convert the each of the following numbers into binary, decimal, hexadecimal, octal numbers. Show your work. Just filling in the values is not enough. ( $8 \times 6$  marks)

	Binary	Decimal	Hexadecimal	Octal
a)	1010 <sub>2</sub>			
b)	10_0110 <sub>2</sub>			
c)		329 <sub>10</sub>		
d)		741 <sub>10</sub>		
e)			7D <sub>16</sub>	
f)			EC3A <sub>16</sub>	
g)				351 <sub>8</sub>
h)				2563 <sub>8</sub>

**Problem 5.14.** 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. ( $6 \times 6$  marks)

	Decimal	Sign-magnitude	One's complement	Two's complement
a)	-79 <sub>10</sub>			
b)	-110 <sub>10</sub>			
c)				0110_1110 <sub>2</sub>
d)				1011_1101 <sub>2</sub>
e)			0110_1101 <sub>2</sub>	
f)			1001_1010 <sub>2</sub>	

**Problem 5.15.** *Convert the decimal numbers to 6-bit two's complement binary and then add them. Check if the addition causes overflow ( $3 \times 6$  marks).*

1.  $-16_{10} - 7_{10}$

2.  $19_{10} - 5_{10}$

3.  $-4_{10} - 29_{10}$

**Problem 5.16.** 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)*