Question 1 (40 points: 15 + 10+ 15):

(a) Convert the following binary into (i) decimal, (ii) octal et (iii) hexadecimal
10100001111.1101

(i) to decimal
$(10100001111)_2$
$= 1 \times 2^{10} + 0 \times 2^{9} + 1 \times 2^{8} + 0 \times 2^{7} + 0 \times 2^{6} + 0 \times 2^{5} + 0 \times 2^{4} + 1 \times 2^{3} + 1 \times 2^{2} + 1 \times 2^{1} + 1 \times 2^{0}$
$= 1024 + 0 + 256 + 0 + 0 + 0 + 0 + 8 + 4 + 2 + 1$
$= 1295$

$(0.1101)_2$
$= 1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4}$
$= 0.5 + 0.25 + 0 + 0.0625$
$= 0.8125$

$(10100001111.1101)_2$ = ***1295.8125***

(ii) to Octal
$(10100001111)_2$
= 010 100 001 111.110 100
= ***2417.64***

Binary	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6

(ii) to Hexadecimal
$(10100001111)_2$
= 0101 0000 1111.1101
= ***50F.D***

Binary	Hexa
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E

(b) Perform the following operation using 10's complement

$$\mathbf{(162)}_{10} - \mathbf{(27)}_{10}$$

$[N]_r = r^n - (N)_r$
$n = 3$
$r = 10$
$N = 027$

$[027]_{10} = 10^3 - (27)_{10}$
$= 973$
10's Complement of $(27)_{10} = 973$

$(162)_{10} - (27)_{10} = (162)_{10}$ + 10's Complement of $(27)_{10}$

$= (162)_{10} + (973)_{10}$

$= (135)_{10}$

$$\begin{array}{r} 1\,1 \\ 162 \\ +\ 973 \\ \hline \cancel{1}\,135 \end{array}$$

(c) Convert the following numbers into binary and perform the arithmetic operations in (i) and (ii) using signed binary numbers with 2's complement. Use 7 bits to represent the integer part of decimal numbers and the sign bit. Use three bits to represent the fractional part.

(i) $(4.5)_{10} - (9)_{10}$

(ii) $(8.5)_{10} + (9)_{10}$

$(4.5)_{10}$ = 0000100.100

$(9)_{10}$ = 0001001.000

2's of $(9)_{10}$ = 1110111.000

$(8.5)_{10}$ = 0001000.100

4/2 = 2 rem **0**
2/2 = 1 rem **0**
1/2 = 0 rem **1**

8/2 = 4 rem **0**
4/2 = 2 rem **0**
2/2 = 1 rem **0**
1/2 = 0 rem **1**

9/2 = 4 rem **1**
4/2 = 2 rem **0**
2/2 = 1 rem **0**
1/2 = 0 rem **1**

0.5 * 2 = **1**.0

(i) $(4.5)_{10} - (9)_{10}$

= 0000100.100 - 0001001.000
= 0000100.100 + 2's of (0001001.000)
= 0000100.100 + 1110111.000
= ***1111011.100***
= - (0000100.100)
= - $(4.5)_{10}$

```
   1
  0000100.100
+ 1110111.000
-------------
  1111011.100
```

(ii) $(8.5)_{10} + (9)_{10}$

= 0001000.100 + 0001001.000
= ***0010001.100***
= $(17.5)_{10}$

```
   1
  0001000.100
+ 0001001.000
-------------
  0010001.100
```

Question 2: (10 + 30 + 5 + 15 = 60 points)

Design a 4- bit combinational circuit 2's complementer. The circuit generates at the output the 2's complement of the input binary numbers.
Answer the following questions:

(i) Complete the following truth table. A,B,C,D indicate the input binary number to be complemented using 2's complement and W,X,Y, Z represent the output 2's complement of the input binary number. The variable D is the least significant bit and A is the most significant bit of the binary number.

A	*B*	*C*	*D*	*w*	x	y	z
0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1
0	0	1	0	1	1	1	0
0	0	1	1	1	1	0	1
0	1	0	0	1	1	0	0
0	1	0	1	1	0	1	1
0	1	1	0	1	0	1	0
0	1	1	1	1	0	0	1
1	0	0	0	1	0	0	0
1	0	0	1	0	1	1	1
1	0	1	0	0	1	1	0
1	0	1	1	0	1	0	1
1	1	0	0	0	1	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	0	1	0
1	1	1	1	0	0	0	1

(ii) Simplify the Boolean function **W** in its **Sum of Products form** using K-Map

	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	0	0	0	0
10	1	0	0	0

W = A'B + A'D + A'C + AB'C'D'

(iii) Show that the Boolean function **W** can be constructed using exclusive-OR gates

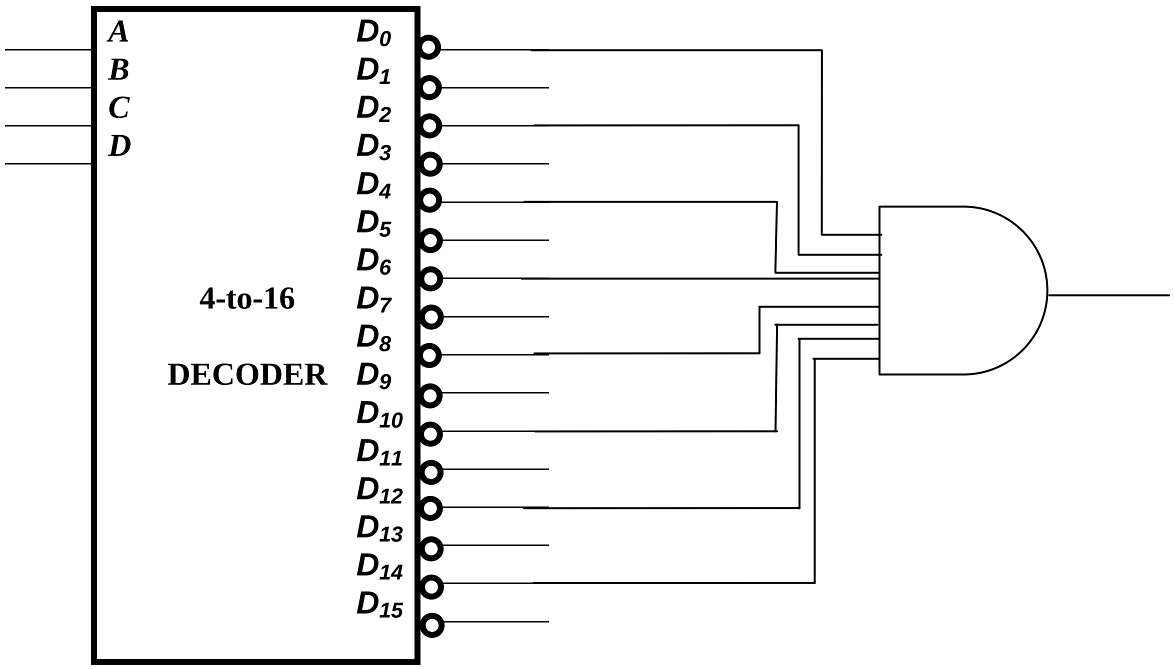
$W = A'B + A'D + A'C + AB'C'D'$
$\quad = A' (B + D + C) + A\, B'C'D'$

$B'C'D' = (B + C + D)'$
$X = (B + C + D)$

$W = A' (B + D + C) + A\, (B + C + D)'$
$W = A'\, X + A\, X'$

$W = A \oplus X$
$W = A \oplus (B + D + C)$

(iv) Implement the Boolean function **Z** in its **Product of Sums form** with a decoder constructed with NAND gates (see figure below) and external gate (s) connected to the decoder outputs.



Sum of Products
Z = Σm(1,3,5,7,9,11,13,15)

Z = ΠM(0,2,4,6,8,10,12,14)

Z_{SoP}= A'B'C'D + A'B'CD + A'BC'D + A'BCD + AB'C'D + AB'CD + ABC'D + ABCD

Z'= A'B'C'D' + A'B'CD' + A'BC'D' + A'BCD' + AB'C'D' + AB'CD' + ABC'D' + ABCD'

Z'' = Z_{PoS}
(A+B+C+D)(A+B+C'+D)(A+B'+C+D)(A+B'+C'+D)(A'+B+C+D)(A'+B+C'+D)(A'+B'+C+D)(A'+B'+C'++D)