Parshvanath Charitable Trust's
A. P. SHAH INSTITUTE OF TECHNOLOGY
(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)
(Religious Jain Minority)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING DATA SCIENCE

UNIT TEST-I-Solution

Class: SE **Semester: III** **Subject: DLCA**

Q. No.	Questions	Marks
Q.1	**Attempt any Two.**	
(a)	Determine BCD, Excess -3 and gray code for the given number $(46.3)_7$.	[05]

Q.1.

(a) $(46.3)_7$ convert to BCD, Excess-3 & Gray code.

$(46.3)_7 \rightarrow$ BCD.

$4\times7^1 + 6\times7^0 + 3\times7^{-1}$

$= 28 + 6 + 3/7$

$= (34.4285)_D$

$= (0011\ 0100\ .\ 0100\ 0010\ 1000\ 0101)_{BCD}$

$(46.3)_7 \rightarrow$ Excess-3.

```
  0011 0100 . 0100 0010 1000 0101
+ 0011 0011   0011 0011 0011 0011
---------------------------------
  0110 0111   0111 1101 1011 1000
```

$\therefore (46.3)_7 = (0110\ 0111\ 0111\ 1101\ 1011\ 1000)_{Ex-3}$

$(46.3)_7 \rightarrow$ Gray

$(46.3)_7 = (34.4285)_D$

2	34	0
2	17	1
	8	0
2	4	0
2	2	0
2	1	1
2	0	

0.4285×2	0.857	0
0.857×2	1.714	1
0.714×2	1.428	1
0.428×2	0.856	0
0.856×2	1.712	1

$(100010.01101)_2$

$1\oplus0\oplus0\oplus0\oplus0\oplus1\oplus0\ .\ 0\oplus1\oplus1\oplus0\oplus1$

$(110011.01011)_{Gray} = (46.3)_7$

(b) Compute Y = $\overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$. [05]

Q.1.

(b) $\overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$

$= \overline{A}BC + A\overline{B}C + AB$ $\{\because \overline{A} + A = 1\}$

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

$= \overline{A}BC + A(B + C)$ $\{\because A + \overline{A}B = A + B\}$

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

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

$= B(A + C) + AC$

$= AB + BC + AC$

(c) Determine octal, binary and hexadecimal code for the given number $(1473.45)_{10}$. [05]

3. $(1473.45)_{10}$: Octal, Binary, Hexadecimal

Solution:

1) Converting Decimal to Binary

2	1473	1
2	736	0
2	368	0
2	184	0
2	92	0
2	46	0
2	23	1
2	11	1
2	5	1
2	2	0
2	1	1
	0	

0.45×2	0.9	0
0.9×2	1.8	1
0.8×2	1.6	1
0.6×2	1.2	1
0.2×2	0.4	0
0.4×2	0.8	0

$(.011100)_2$

$\therefore$ (10111000001.011

$(10111000001)_2$

2) Decimal to Octal : $(1478.45)_{10}$

8	1473	1	0.45×8	3.6	3
8	184	0	0.6×8	4.8	4
8	23	7	0.8×8	6.4	6
8	2	2	0.4×8	3.2	3
	0	$(2701)_8$	0.2×8	1.6	1
				$(34631)_8$	

3) Decimal to Hexadecimal.

$(1473.45)_{10}$

16	1473	1	0.45×16	7.2	7
16	92	$12 \rightarrow C$	0.2×16	3.2	3
16	5	5	0.2×16	3.2	3
	0		0.2×16	3.2	3
	$(5\,12\,1)_{16}$			$(7333)_{16}$	

$\therefore \ (5\underline{12}1.7333)_{16}$

$(5C1.7333)_{16}$

(d)	Compute Y = AB + A(B+C) + B(B+C)	[05]

1) $AB + A(B+C) + B(B+C)$

$= \underline{AB + AB} + AC + \underline{B.B} + BC$

$= \underline{AB} + AC + \underline{B} + \underline{BC}$

$= B(A+1+C) + AC$

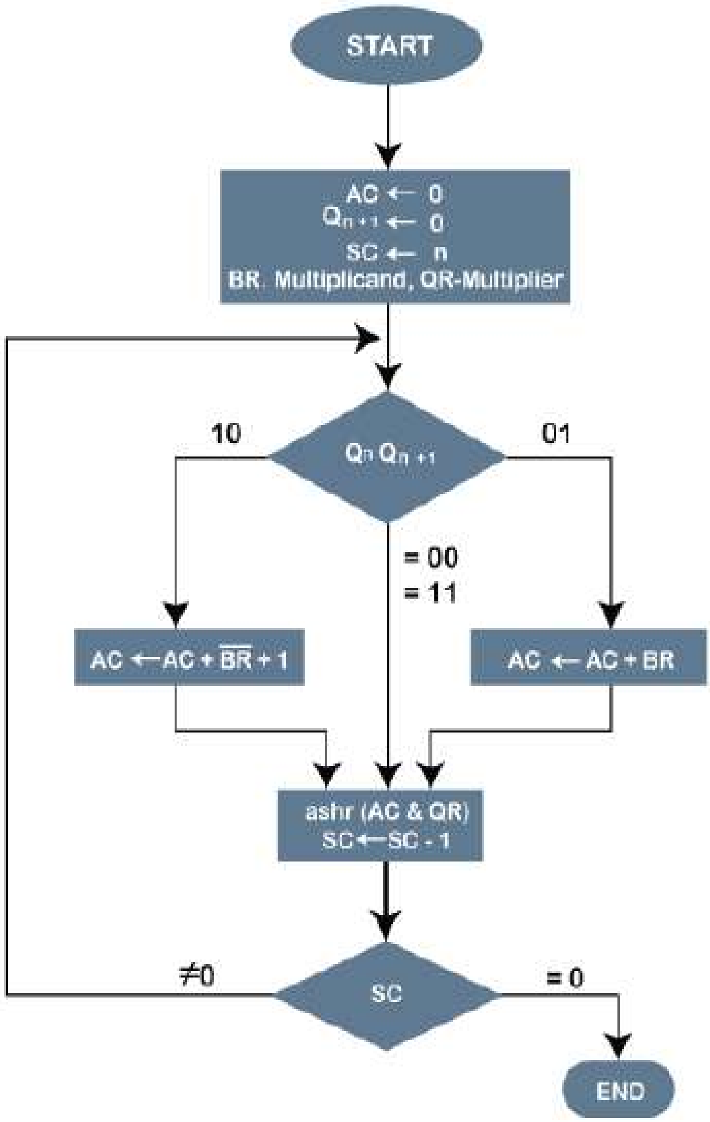
$= B.1 + AC$

$= \boxed{B + AC}$

Q.2	**Attempt any Two.**	
(a)	Illustrate Booths multiplication algorithm and solve $(-7)_{10} \times (3)_{10}$.	[10]

BOOTHS ALGORITHM

The booth algorithm is a multiplication algorithm that allows us to multiply the two signed binary integers in 2's complement, respectively. It is also used to speed up the performance of the multiplication process. It is very efficient too.



Working on the Booth Algorithm

1. Set the Multiplicand and Multiplier binary bits as BR and QR, respectively.
2. Initially, we set the AC and Q_{n+1} registers value to 0.

3. SC represents the number of Multiplier bits (QR), and it is a sequence counter that is intialised to the value n which is number of multiplier bits and in the algorithm it is continuously decremented till it reaches to 0.
4. A Qn represents the last bit of the QR, and the Q_{n+1} shows the incremented bit of Qn by 1.
5. On each cycle of the booth algorithm, Q_n and Q_{n+1} bits will be checked on the following parameters as follows:
    i. When two bits Q_n and Q_{n+1} are 00 or 11, we simply perform the arithmetic shift right operation (ashr) over AC , QR and Q_{n+1}
    ii. If the bits of Q_n and Q_{n+1} is shows to 01, the multiplicand bits (BR) will be added to the AC (Accumulator register). After that, we perform the right shift operation over AC , QR and Q_{n+1} .
    iii. If the bits of Q_n and Q_{n+1} is shows to 10, the multiplicand bits (BR) will be subtracted from the AC (Accumulator register). After that, we perform the right shift operation over AC , QR and Q_{n+1} .
6. After every right shift operation the sequence counter is decremented and the operation continuously works till the sequence counter value reaches 0.
7. Results of the Multiplication binary bits will be stored in the AC and QR registers.

For BR.

∵ −7 is a −ve number, we have to take 2's complement of −7 to get the value of BR.

$$\therefore \quad 0111 \rightarrow \begin{array}{r} 1000 \\ +\quad 1 \\ \hline 1001 \end{array} \rightarrow BR$$

∴ BR = 1001

QR = 3 = 0011 {∵ 3 is +ve, no need to take its 2's complement}

	AC	QR	Q_{n+1}	SC	
	0000	0011	0	4	1st cycle.
	+ 0111				
	0111	0011	0		
SR	0011	1001	1	3	2nd cycle.
SR	0001	1100	1		
	+ 1001				
	1010	1100	1	2	3rd cycle
SR	1101	0110	0		
SR	$\boxed{1110}$	$\boxed{1011}$	0	1	4th cycle

MSB

check if the MSB = 1 or 0.

if MSB = 1 ⇒ Invalid ANS. ⇒ 2's Complement

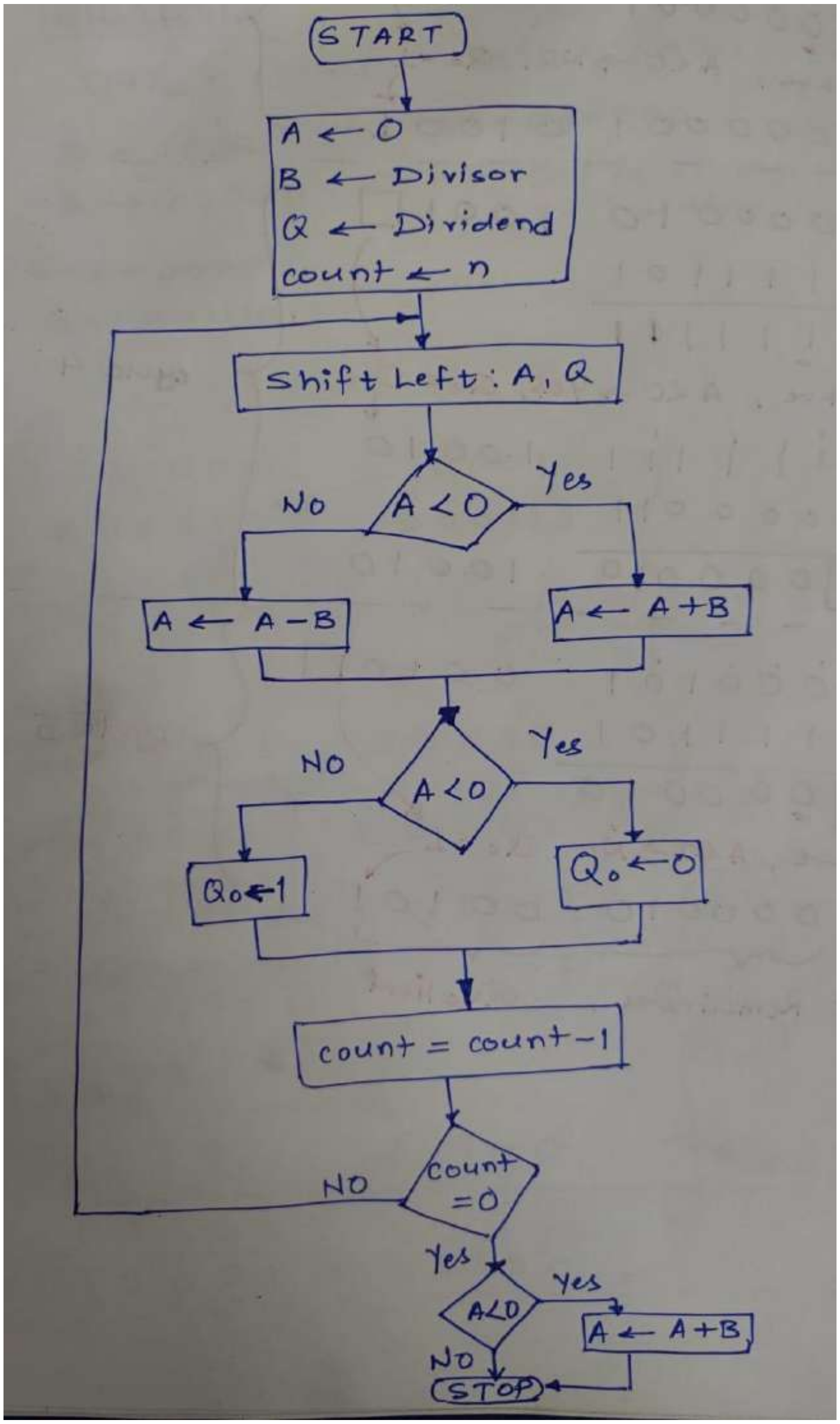
if MSB = 0 ⇒ Valid ANS

1110 1011

1's Complement → 0001 0100

2's Complement → 0001 0101 ⇒ −21

(b)	Illustrate Non-Restoring method of binary division with algorithm and divide $(12)_{10}$ by $(3)_{10}$.	[10]



START
A ← 0
B ← Divisor
Q ← Dividend
count ← n
Shift Left : A, Q
A < 0
Yes
No
A ← A − B
A ← A + B
A < 0
Yes
NO
Q0 ← 1
Q0 ← 0
count = count − 1
count = 0
NO
Yes
A < 0
Yes
A ← A + B
NO
STOP

Q) Dividend = 12 → 1100 → Q

Divisor = 3 → 0011

B = 00011 ; $\bar{B}+1$ = 11101

A = 00000

count (n) = 4

	A	Q	
	00000	1100	cycle 1
LS	00001	100□	
	+ve, $A<0$ → NO ⇒ A−B		
A−B	00001		
	+ 11101		
	11110	100□	
	−ve, $A<0$ → Yes ⇒ $Q_0 = 0$		
	11110	1000	
LS	11101	000□	cycle 2
	−ve, $A<0$ → Yes ⇒ A+B		
A+B	11101		
	+ 00011		
Ignore → $\boxed{1}$	00000		
	+ve, $A<0$ → No ⇒ $Q_0 = 1$		
	00000	0001	
LS	00000	001□	cycle 3
	+ve, $A<0$ → NO ⇒ A−B		
A−B	00000		
	+ 11101		
	11101	001□	
	−ve, $A<0$ → yes ⇒ $Q_0 = 0$		
	~~00000~~	0010	
	11101	0010	
LS	11010	010□	cycle ~~3~~ 4
	−ve, $A<0$ → Yes ⇒ A+B		
A+B	11010		
	+ 00011		
	11101	010□	
	−ve, $A<0$ → Yes ⇒ $Q_0 = 0$		
	11101	0100	
	−ve, $A<0$ → Yes, ⇒ A+B		
A+B	11101		
	+ 00011		
Ignore → $\boxed{1}$	00000	0100	
	Remainder	Quotient	

(c)	i. Determine IEEE 754 single and doubleprecision floating point representation for $(85.125)_{10}$.	[10]

$(85.125)_{10} \rightarrow$ Binary [~~by continuous division~~]

$85 \rightarrow 1010101 \rightarrow$ [by continuous division by 2]

$0.125 \rightarrow 001 \rightarrow$ [continuos multiplication by 2]

$(85.125)_{10} = (1010101.001)_2$

[shift the binary point exactly after the 1st bit.

$= (1.\underbrace{010101001}_{\text{Mantissa}})_2 \times 2^6$

[2^6 because the binary point is shifted by 6 bits.

$2^6 \rightarrow E$. $\therefore E = 6$

(i) Single precesion

$E = 6$

- sign $\rightarrow$ +ve $\rightarrow 0$
- $E' = E + 127 \Rightarrow \cancel{E+}$

$E' \Rightarrow 6 + 127 = 133$

Convert $(133)_{10}$ to binary.

$(133)_{10} \rightarrow (10000101)_2 \rightarrow E'$

(85.125) is +ve no.

$\therefore$ Sign bit is 0.

$$\underbrace{0}_{\text{sign 1-bit}}\ \underbrace{10000101}_{\text{Exponent (8 bit)}}\ \underbrace{\underbrace{010101001}_{9\text{ bits}}\ 00000000000000}_{23\text{ bits - Mantissa}}$$

(ii) Double Precision:

$E' = E + 1023$

$E' = 6 + 1023 = 1029$

Convert $(1029)_{10}$ to binary.

~~1029~~ $E' = (10000000101)_2$

(85.125) is +ve no. $\therefore$ sign bit = 0.

D.P. Representation →

$$\underbrace{0}_{\substack{\text{sign}\\ \text{(1bit)}}}\ \underbrace{10000000101}_{\substack{\text{Exponent}\\ \text{(11 bits)}}}\ \underbrace{\underbrace{010101001}_{9\text{ bits}}\ 0000000000000000000000000000000000000000000}_{\substack{\text{Mantissa}\\ \text{(52 bits)}}}$$

ii. $(DDCC)_H + (BBAA)_H$ without converting to any other base.

② $(DDCC)_H + (BBAA)_H = (?)_H$

$$\begin{array}{cccccc}
 & 1 & 1 & 1 & & \\
 & (13)_H & (13)_H & (12)_H & (12)_H & \\
 & (11)_H & (11)_H & (10)_H & (10)_H & \\
\hline
1 & 9 & 9 & 7 & 6 &
\end{array}$$

$$\begin{array}{r} 12 \\ +10 \\ \hline 22 \end{array}$$

$22 = 1 \times 16 + 6$

$\boxed{1 + 12 + 10 = 23}$

$23 = 1 \times 16 + 7$

$\boxed{1 + 13 + 11 = 25}$

$25 = 1 \times 16 + 9$

$(DDCC)_H + (BBAA)_H = (19976)_H$

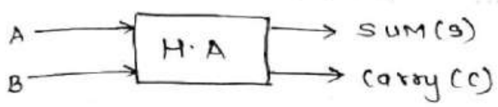
Q.3	**Attempt any One.**	
(a)	Implement Half adder and Full adder with the help of truth table, boolean expressions, K map and basic gates.	[10]
(b)	Implement 8:1 Multiplexer and 1:8 Demultiplexer with the help of truth table.	[10]

HALF ADDER:

→ It is a combinational logic ckt with 2 i/p's and 2 o/p's.

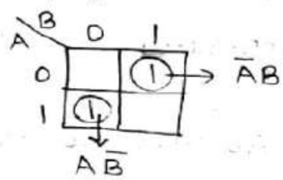
→ It is the building block for addition of 2 single bit nos.

→ This ckt has 2 o/p's namely carry & sum.

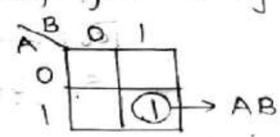


INPUT		OUTPUT	
A	B	S	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

K map for sum.



K map for carry

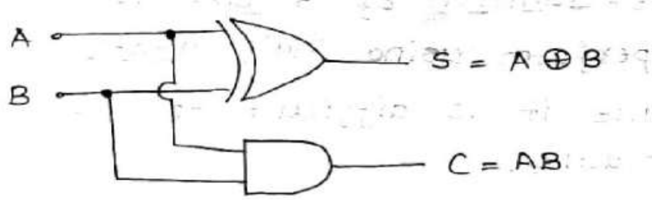


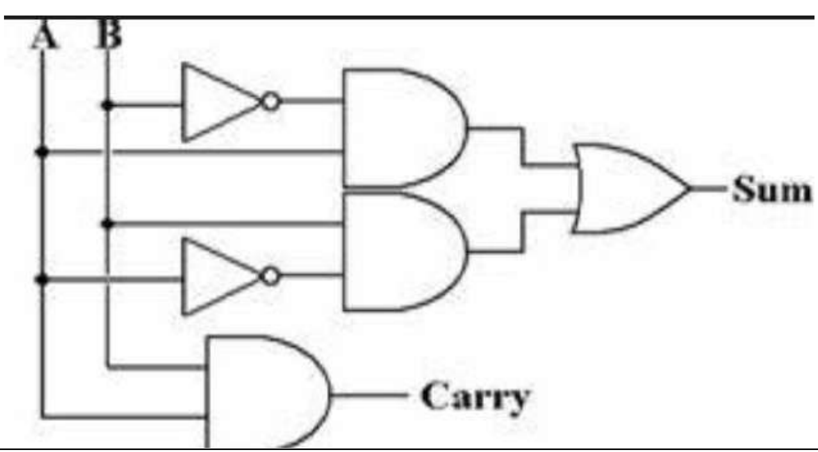
$S = A\bar{B} + \bar{A}B$

$= A \oplus B$

$C = AB$.

Implementation.



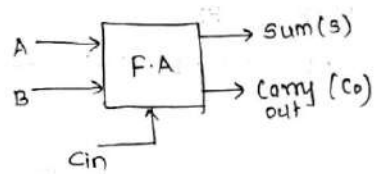


FULL ADDER.

→ To overcome drawback of half adder ckt., a single 3 bit adder ckt called full adder is developed.

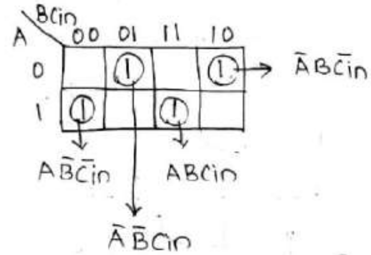
→ It can add 2 one bit nos. A and B and carry Cin.

→ It is a 3 i/p and 2 o/p combinational ckt.

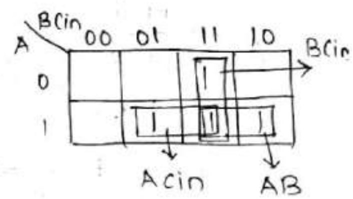


INPUTS			OUTPUTS	
A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

kmap for Sum



K map for Cout



Cout = AB + ACin + BCin

$S = A\bar{B}\bar{C}in + \bar{A}\bar{B}\,Cin + ABCin + \bar{A}B\bar{C}in$

$= A \oplus B \oplus C_{in}$

Implementation.