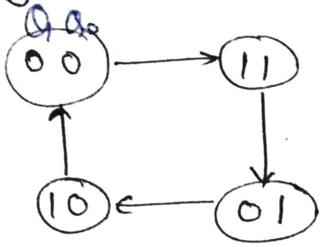
Counter using 2 different FF's

Consider the following state diagram which is to be designed using T-FF for MSB and XY for LSB. The behaviour of XY is given below.



x	y	Q_n
0	0	0
0	1	Q
1	0	Q'
1	1	1

Sol^n

Q_1 Q_0

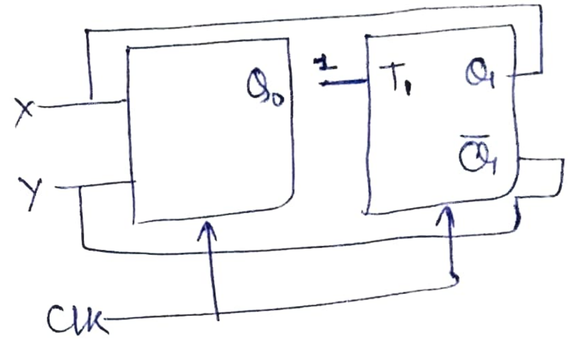
XY ← → TFF

Q_1	Q_0	Q_{1N}	Q_{0N}	T	X	Y
0	0	1	1	1	1	Φ
0	1	1	0	1	Φ	0
1	0	0	0	1	0	Φ
1	1	0	1	1	Φ	1

x	y	Q_n	Q_{n+1}	~~X~~	~~Y~~
0	0	0	0		
0	0	1	0		
0	1	0	0		
0	1	1	1		
1	0	0	1		
1	0	1	0		
1	1	0	1		
1	1	1	1		

Q	Q_n	X	Y
0	0	0	Φ
0	1	1	Φ
1	0	Φ	0
1	1	Φ	1

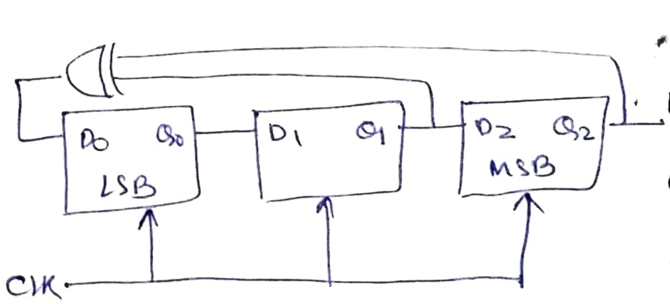
$T = 1 \quad X = \overline{Q_1} \quad Y = Q_1$



Procedure

1. First draw state table
2. Find the inputs of flipflop for this we can use excitation table and characteristic table.
3. Realise the counter.

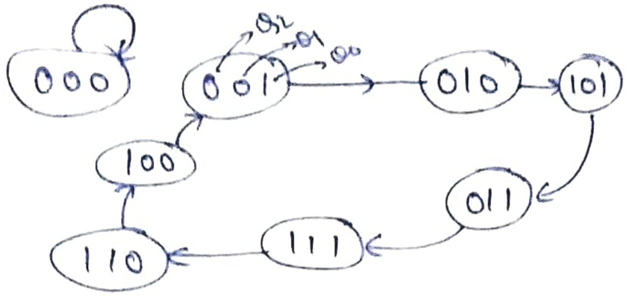
Model on analysis counting states and sequence generation



→ D-FF provide delay of 1 clock cycle.

Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	1	0
0	1	0	1	0	1	1	0	1
0	1	1	1	1	1	1	1	1
1	0	0	0	0	1	0	0	1
1	0	1	0	1	1	0	1	1
1	1	0	1	0	0	1	0	0
1	1	1	1	1	0	1	1	0

$Q_{0N} = Q_1 \oplus Q_2$
$Q_{1N} = D_1 = Q_0$
$Q_{2N} = D_2 = Q_1$



→ It could act as mod 7 Counters.

→ If initial state is 001 then what will the state after
- 4 clock = 111 = 4%7
- 10 clock = 011 = 10%7
- 117 clock = (117 %7) = 5 = 110

→ If output tapped at Q_2 what would be the sequence obtained, initial state = 001

Ans → 0010111 0010111

at Q_1

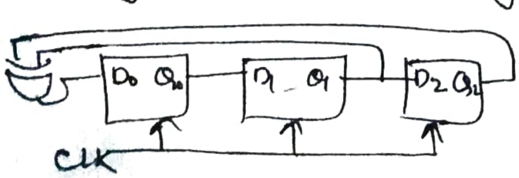
010 1110 010 1110 ...

at Q_0

1011100 1011100 ...

→ Counters can also acts as Sequence generators.

Deriving the Clock frequency



Given, $T_{FF} = 15$ ns
$T_{comb} = 5$ ns

which of the following clock frequency ensures proper counting?

a) 40 MHz b) 60 MHz c) 90 MHz
d) 300 MHz

Sol^n for synchronous counter;

$T_{CLK} \geq T_{FF} + T_{comb}$

$T_{CLK} \geq 15\text{ ns} + 5\text{ ns}$

$T_{CLK} \geq 20\text{ ns}$

frequency $\leq \frac{1}{20\text{ ns}}$

freq. $\leq 0.5 \times 10^9$

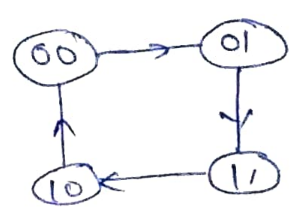
freq ≤ 50 MHz

a) 40 MHz is the answer.

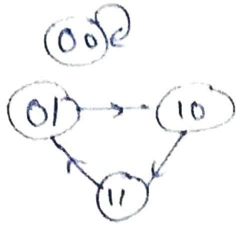
Self Starting and free running

→ A counter is said to be self starting if it is possible to enter counting loops irrespective of initial state.

→ A counter is said to be free running if it contains all possible states in counting loop.

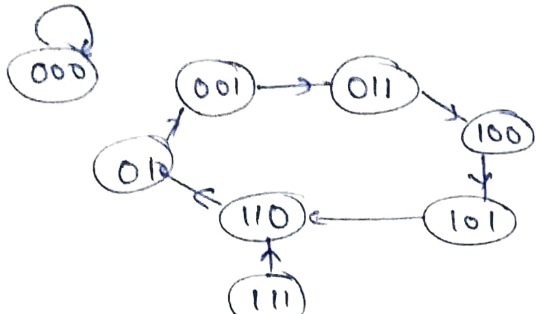
a) 

Self starting & free running

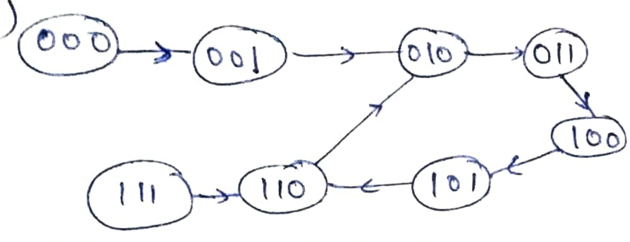
b) 

Not a self starting bcoz if (00) is initial state we can't reach in loop.

→ Not free running

c) 

→ Not free running and not self starting

d) 

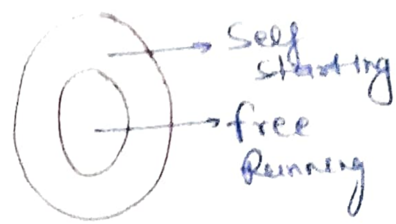
→ Self starting but ~~not~~ free running

Q In d) above what no. of Clock cycle require to enter in counting ~~sequ~~ loop if your current state is (000)

Ans 2 clocks to reach (010)

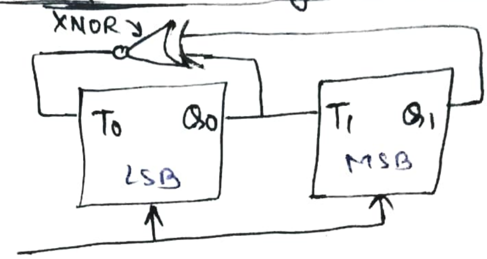
Imp → If a counter is free running then it must be self starting but converse is not always true.

Imp → If counter is not self starting, it neither be free running.

→ 

Example on self starting and free running counter

Ex! Identify if the following counter is self starting or free running.



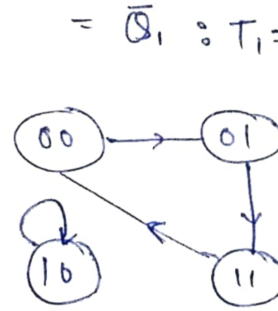
Soln

Q_1	Q_0	Q_{1N}	Q_{0N}	T_1	T_0
0	0	0	1	0	1
0	1	1	1	1	0
1	0	1	0	0	0
1	1	0	0	1	1

$T_1 = Q_0 \quad T_0 = Q_0 \odot Q_1$

$Q_{1N} = Q_1 : T_1 = 0 \qquad Q_{0N} = Q_0 : T_0 = 0$

$= \bar{Q}_1 : T_1 = 1 \qquad = \bar{Q}_0 : T_0 = 1$

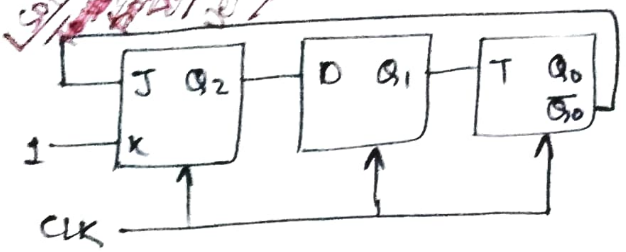


→ Not self starting nor free running.

Counter using 3 different FF's

Consider the following counter

Initially $Q_2Q_1Q_0 = 000$



Q. What will be the state after 4 clocks?

a) 000 b) 010 c) 011 d) 101

Q. Modulus of the counter.

a) 4 b) 5 c) 6 d) 7

Sol^n State table

Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	0	1	1
1	1	0	0	1	1
1	1	1	0	1	0

$Q_{1N} = D = Q_2$

$Q_{0N} = T = Q_0 \quad (T=0)\ Q_1 = 0$

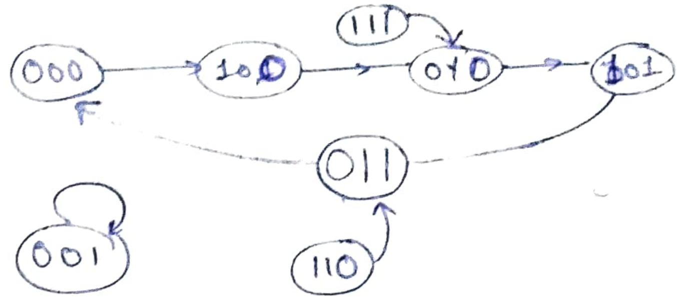
$= \overline{Q_0} \quad (T=1)\ Q_1 = 1$

$Q_{2N} \Rightarrow$

J	K	Q_N
0	1	0
1	1	$\overline{Q_N}$

$J = \overline{Q_0}$

$Q_N = Q_{2N}$] here



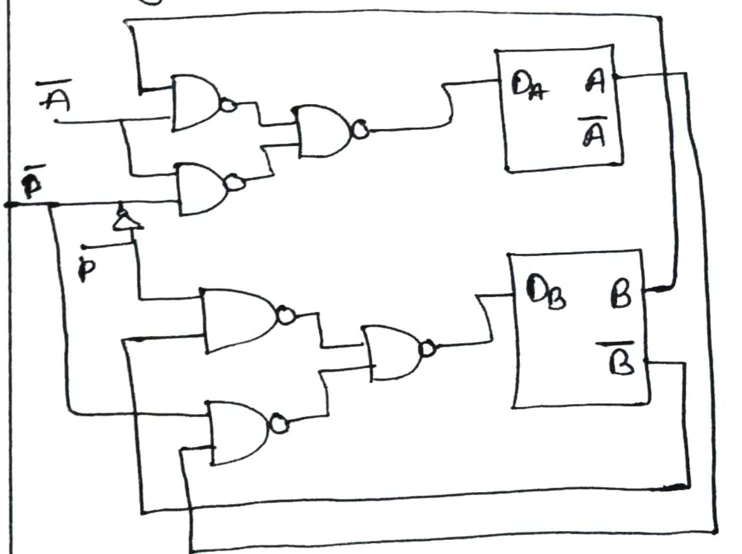
→ It is not free running and self starting.

① c) 011 after 4 clocks

② b) 5 modulus counter.

Example on Combinational ckts and FF's

Q. Consider the following counter, if P=0, the counting sequence of AB is?



Sol^n

A	B	A_n	B_n
0	0	1	0
0	1	1	0
1	0	0	1
1	1	0	1

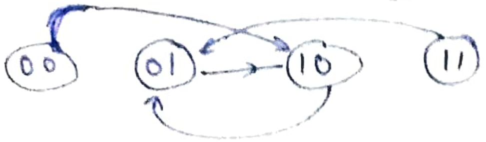
$B_n = D_B = A$

$A_n = D_A = \overline{A}$

$D_A = \overline{A}B + \overline{A}\,\overline{P} = \overline{A}\cdot(B+\overline{P})$

$D_A = P\overline{B} + A\overline{P}$

Put $P=0$, $D_A = \overline{A}$, $D_B = A$



→ Acting as mod 2 counter.

Asynchronous Counters

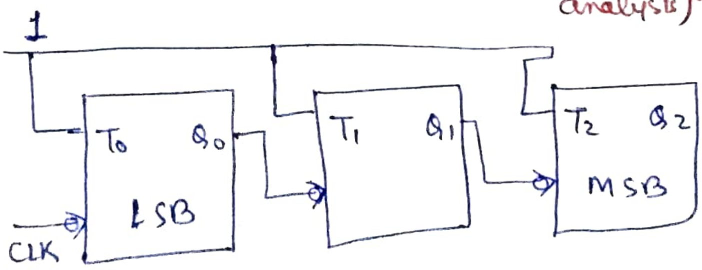
→ The asynchronous are also called as ripple counter.
The basic flipflop is T and basic counting is binary up counting.

→ The up counters are used for implementing incrementation.

→ Due to simpler design, asynch. counters are preffered in IC counter fabrication.

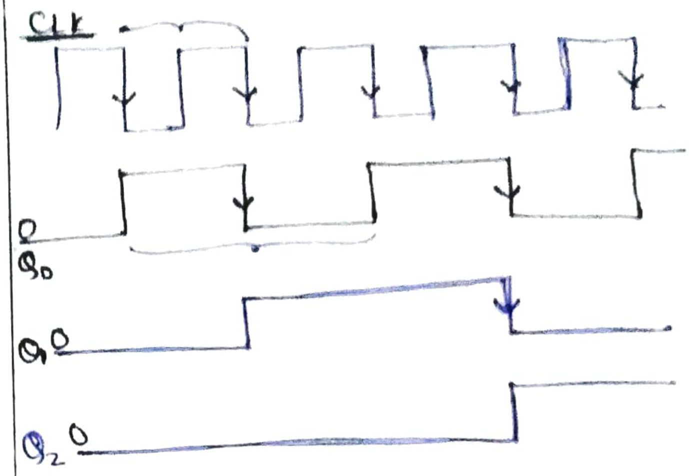
Ex → IC 7490 is a decade counter i.e %10 counter.
IC 7492 is hexadecimal counter i.e %16 counter.

Mod 8 up counter (Imp for remembering analysis)



→ Clock is −ve edge triggered at T_0, T_1, T_2 is given as input 1.

So, on analyze the behaviour of Q_2, Q_1, Q_0 in time clock we get,



We assume that each Q_0, Q_1, Q_2 is initially zero.

Q_1 is working according of cycle of Q_0 and Q_2 working according to clock cycle of Q_1.

So,

Time period of $Q_0 = 2 * T_{CLK}$

freq $Q_0 = \frac{1}{2} f_{CLK}$

Time period of $Q_1 = 2 * T_{Q_0}$

freq $Q_1 = \frac{1}{2} * f_{Q_0}$

$= \frac{1}{4} f_{CLK}$

Time period of $Q_2 = 2 * T_{Q_1}$

freq $= \frac{1}{2}$ freq Q_1

$= \frac{1}{8}$ freq CLK

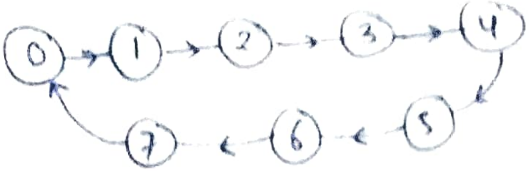
$Q_{0N} = \overline{Q_0}$; every clock

$Q_{1N} = \overline{Q_1}$; Q_0 changes $1 \to 0$

$Q_{2N} = \overline{Q_2}$; Q_1 changes $1 \to 0$

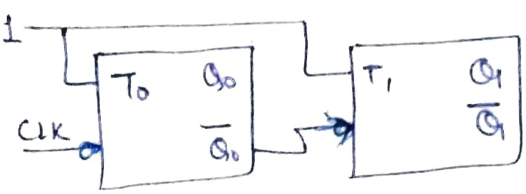
* $1 \to 0$ means −ve edge cycles.

Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0



- → mod 8 upcounter.
- → It is free running and self starting both.
- → up counter bcoz eachtime we are going to above state.
- → On changing (−ve) edge to (+ve) edge triggered and Q_2, Q_1, Q_0 as $\overline{Q_2}, \overline{Q_1}$ and $\overline{Q_0}$ then behaviour remain same.

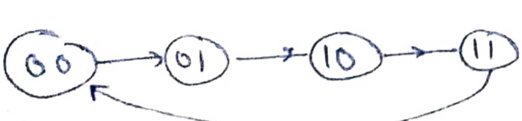
Mod 4 up counter



$Q_{0N} = \overline{Q_0}$; for every clock

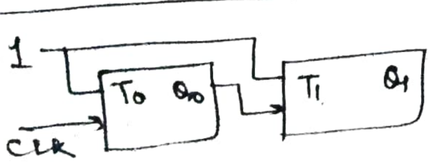
$Q_{1N} = \overline{Q_1}$; for $\overline{Q_0} : 0 \to 1$
$Q_0 : 1 \to 0$

Q_1	Q_0	Q_{1N}	Q_{0N}
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0



mod 4 up counter.

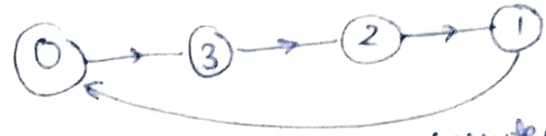
Mod 4 down Counter



$Q_{0N} = \overline{Q_0}$ for every clock

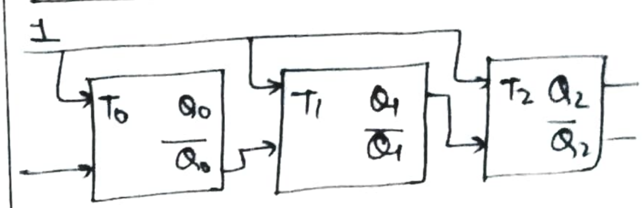
$Q_{1N} = \overline{Q_1}$; $Q_0 : 0 \to 1$

Q_1	Q_0	Q_{1N}	Q_{0N}
0	0	1	1
0	1	0	0
1	0	0	1
1	1	1	0



mod 4 down counter.

Mod 8 random Counter



If initial state $Q_2Q_1Q_0 = 101$. What will be the state after 4 cycles?

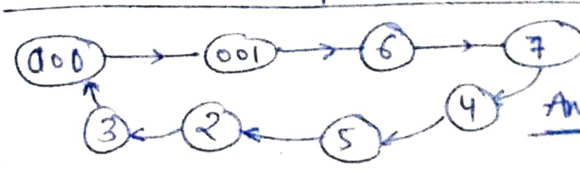
Sol^n → After 4 cycles at 001

$Q_{0N} = \overline{Q_0}$: for every cycle

$Q_{1N} = \overline{Q_1}$: for $\overline{Q_0} : 0 \to 1$)
$Q_0 : 1 \to 0$

$Q_{2N} = \overline{Q_2}$: for $Q_1 : 0 \to 1$

Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}
0	0	0	0	0	1
0	0	1	1	1	0
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	1	0	1
1	0	1	0	1	0
1	1	0	1	1	1
1	1	1	1	0	0



Ans-1

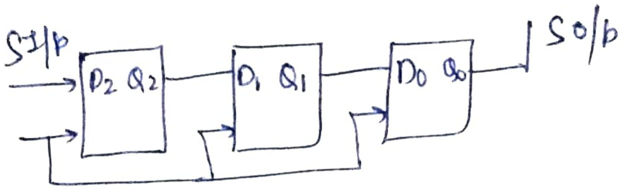
Application of flip flops

① **Shift register :** used as sequential memory
Ex: accumulator in up

② **Counters :** ① used to count no of pulses.
② used as frequency divider.

3 bit shift register

It requires 3 D-FF.



SI/p → Serial Input
SO/p → Serial output

→ we can also use other than D-FF but its most considerable flipflop.

CLK	SI/p	Q_2	Q_1	Q_0
0	—	0	0	0
1	1	1	0	0
2	0	0	1	0
3	1	1	0	1

we give I/p as 101

3 clock cycles

In a serial input n-bit shift register, we need n-clock cycles to put put n-bits on shift register.

CLK	Q_2	Q_1	Q_0	S0/p
0	1	0	1	1
1	0	1	0	0
2	0	0	1	1

2 clock cycles

for n-bit shift register for serial output we need (n-1) clock cycles.

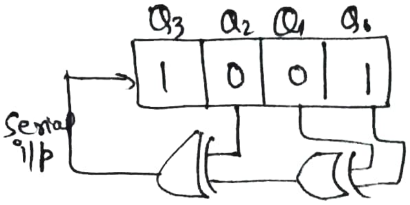
imp
Total clock pulses required to put n-bits on shift register and then takes output from it in serial I/p or o/p manner

$= n + n - 1 = \underline{2n-1}$

In case of parallel in/put and parallel o/p. It doesn't support shift register.

Examples on shift right register :

Ex-1. In the following right shift register, determine the no. of clocks required to bring it to the initial state of '1001'.

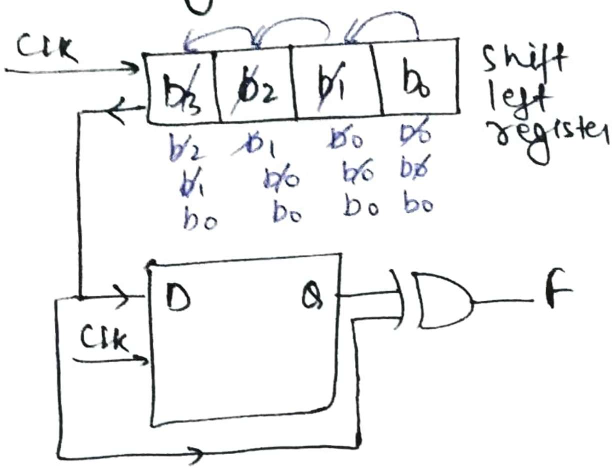


CLK	SI/p	Q_3	Q_2	Q_1	Q_0
0	–	1	0	0	1
1	1	1	1	0	0
2	1	1	1	1	0
3	0	0	1	1	1
4	1	1	0	1	1
5	0	0	1	0	1
6	0	0	0	1	0
7	1	1	0	0	1

After 7 clocks

Binary to gray Converter

Q Determine the function of following circuit.



Soln

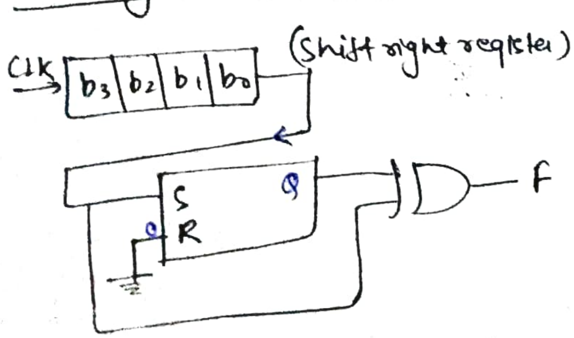
CLK	F (o/p)
0	$0 \oplus b_3 = b_3 = O_3$
1	$b_3 \oplus b_2 = O_2$
2	$b_2 \oplus b_1 = O_1$
3	$b_1 \oplus b_0 = O_0$
4	$b_0 \oplus b_0 = 0$
	0 0

b_3 b_2 b_1 b_0

O_3 O_2 O_1 O_0 (say)

then, $O_3 = b_3$ and these all represent gray code correspond to binary code.

Finding 2's complement



S	R	Mode	Q
0	0	Latch	{1, 0}
1	0	Set	1

Initially $Q = 0$ (given)

$b \oplus 0 = b$

$b \oplus 1 = \bar{b}$

Note → watch the video after completing number system and write notes below.

Gate 2001 (Counting sequence)

$Q_0 = 1,\ Q_1 = Q_2 = 0,$

State of ckt $= 4Q_2 + 2Q_1 + Q_0$

Note → Easy and do it by self.

Number System

Base 1 → 0 (Abacus)

Base 10 → (0–9)

Base 11 → (0–9), A

Base 12 → (0–9), A, B

Base 16 → (0–9), A, B, C, D, E, F

Base 2 → 0, 1

Base-8 → (0–7)

Ex → $(012\,6)_6$ is not possible bcoz all no. in base 6 should be less than 6.

Convert a number in base (b_1) to base (b_2).

(i) If given number is in power of 2 then,

$b_1 \to$ binary $\to b_2$

(ii) If given number is not in power of 2, then,

$b_1 \to$ decimal $\to b_2$

Conversion to base 10

If you ask to convert base $b_1 \to b_2$

then

① convert $b_1 \to$ Base 10

② convert Base 10 — b_2

$(a\,b\,c)_x = (\ z\ )_{10}$

$a \times x^2 \quad b \times x^1 \quad c \times x^0$

$(z)_{10} = (ax^2 + bx + c)_{10}$

Ex → $(123)_7 = (z)_{10}$

$z = 3 + 2 \times 7 + 7^2$

$z = 3 + 14 + 49$

$z = (66)_{10}$

$(12AB)_{16} = (z)_{10}$

$\Rightarrow z = B + A \times 16 + 2 \times 16^2 + 3 \times 16^3$

$\Rightarrow z = 11 + 160 + 512 + 3 \times 16^3$

$(1010)_2 = (\ )_{10}$

$\Rightarrow 0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3$

$= 2 + 8$

$= (10)_{10}$

10 → $1 \times 2 + 0 = 2$ | 101 → $(2 \times 2 + 1) = 5$ | 1010 → $(5 \times 2) + 0 = 10$.

Start from most significant bit

$(125)_{10} \to (\ )_{10}$

125

$1 \times 10 + 2$

$= 12\ 5$

$12 \times 10 + 5 = (125)_{10}$

You can convert any base to base 10

$(1212)_3 = (50)_{10}$

$1 \times 3 + 2 = 5$

$5 \times 3 + 1 = 16$

$16 \times 3 + 2 = 50$

Conversion from base 10

Base 10 → b_2

Ex → $(11)_{10}$ → $(\ )_2$

2	11	1
2	5	1
2	2	0
2	1	1
	0	

$(1011)_2$

To convert $(x)_{10} \to (\ )_2$ we take $\lceil \log_2 x \rceil$ steps.

$(32)_{10} - (\ )_2$

2	32	0
2	16	0
2	8	0
2	4	0
2	2	0
2	1	1
	0	

$(100000)_2$

Short Approach

But we do the same using another approach if given base is in power of 2

Ex → $\boxed{(16 \to 2), (4 \to 2), (32 \to 2)}$

Ex → $(1056)_{16} - (\ )_2$

$16 \to 2^{4} \to 4$ bits

$\underset{0001}{1}\ \underset{0000}{0}\ \underset{0101}{5}\ \underset{0110}{6}$] Make pairs of 4 bits

$= (1000001010110)_2$

Ex → $(\underline{100}\,\underline{1110}\,\underline{1111})_2 = (\ )_{16}$

0100 → 4, 1110 → E, 1111 → F

$(4EF)_{16}$

~~Imp~~

① Always start pairing from LSB

② If bits less than required add 0 bits.

→ Also check after conversion the no. is valid or not

$(273)_8 \to (\ )_2$

$\underset{010}{2}\ \underset{111}{7}\ \underset{011}{3}$] 3 bits pair $(2^3 = 8)$

$= (10111011)_2$

for (Base) $8 \to 2 \to 16$

→ Convert Base 8 → Base 16 using base 2

→ (Base) 3 can convert convert to base 9, 27 using same approach 2-4, 2-8, 2-16 conversion occurs.

$(32)_{10} \to (\ )_2$

16	32	0
16	2	2
	0	

$= (20)_{16}$

$= \overline{0010}\,\overline{0000}$

$= (100000)_2$

we use $10 \to 16 \to 2$ approach

$(65)_{10} \to (\ )_2$

32	65	1
32	2	2
	0	

$= (21)_{32}$

$\underset{00010}{2}\ \underset{00001}{1}$

$= (1000001)_2$

Min. no. of bits required for inter conversion.

Ex $(6728)_{10} \to (\quad)_2$

first we discuss a general concept to convert into base x.

$(\underline{1}\ \underline{2}\ \underline{3}\ \underline{4} \ldots \underline{n})_x$

Consider a number with n digits and each digit can have max. value as $(x-1)$ and weight of each digit is given by

$(\underline{x^{n-1}}\ \underline{x^{n-2}}\ \underline{x^{n-3}}\ \underline{\ }\ \ldots\ \underline{x^0})$

Maximum no. possible

$(x-1)x^{n-1} + (x-1)x^{n-2} + \ldots + (x-1)x^0$

$= (x-1)[x^{n-1} + x^{n-2} + \ldots x^0]$

$= (x-1)(1 + x + x^2 + \ldots x^{n-1})$

$= \dfrac{(x-1)\,1(x^n - 1)}{(x-1)}$

$= \boxed{x^n - 1}$ ✓

Ex → for Base 2

$(\underline{1}\ \underline{1}) = (2^2 - 1) = 3$

$(11)_2 = 3$

max value with 2 digit = 3

for Base 10

$(\underline{9}\ \underline{9}) = (10^2 - 1)$

$= 99$

$\underline{9}\ \underline{9}\ \underline{9} = (10^3 - 1)$

$= 999$

Any no. with base x and n digits should fall in range $0 \to (x^n - 1)$.

so, for $(6728)_{10}$

$\boxed{(x^n - 1)_{10} \geq (6728)_{10}}$ → both are in base 10

$x^n \geq 6729$

$n \log_x x \geq \log_x 6729$

$\boxed{n \geq \log_x (6729)}$ here $x = 2$

$n \geq 12.$ something.

So, Ans is 13 bcoz we need min. no. of bits and we can't take the faction.

→ In case of minimum go with ceil and in case of maximum go with floor.

→ If number not given in decimal then convert it into decimal first, bcoz equation is applicable for decimal only

Ex → Calculate min. no. of bits in $(516)_7 \to$ Base 4

$= (5\times7^2 + 1\times7 + 6) \leq 4^n - 1.$

Q what is the min. no. of digits required to represent a 32 digit decimal no. in binary.

→ Radix and base both are same things.

→ we are doing the question in general form.

Say, we have k-digit no. of base 'b' and convert it to n-digit base x.

largest no. possible $= b^k - 1$
and number (largest no.) must be exist in range of base x.

So, $b^k - 1 \le x^n - 1$

$b^k \le x^n$

$\Rightarrow \boxed{n \geq k \log_x b}$

$\Rightarrow \boxed{n \geq 32 \log_2 10}$

(iii) 10 digit base 8 → base 2

$n \geq 10 \log_2 8$

$\boxed{n \geq 30}$.

(iii) 10 digit base 8 → base 4

$n \geq 10 \log_4 8$

$n \geq 10 \log_{2^2} 2^3$

$n \geq 10 \times \frac{3}{2}$

$n \geq 15$

$[\log_{a^k} a^{k'}] = \frac{k'}{k}$

Q If $(11)_2 + (22)_3 + (33)_4 + (44)_5 = (abc)_6$. find a, b, c.

Best method is convert everything to base 10.

$(11)_2 = 1 \times 2^1 + 1 \times 2^0 = (3)_{10}$

$(22)_3 = 2 + 6 = (8)_{10}$

$(33)_4 = 3 + 12 = (15)_{10}$

$(44)_5 = 4 + 20 = (24)_{10}$

On adding all we get
$(50)_{10}$

Convert $(50)_{10}$ to base 6

$$\begin{array}{r|l l}
6 & 50 & 2 \\
\hline
6 & 8 & 2 \\
\hline
6 & 1 & 1 \\
\hline
 & 0 &
\end{array} \quad = (122)_6$$

$a = 1, \; b = c = 2$

$(3)_{10} = (3)_6$, $(15)_{10} = (23)_6$

$(8)_{10} = (12)_6$ $(24)_{10} = (40)_6$

On adding all

$$\begin{array}{c|c}
 & 3 \\
\hline
1 & 2 \\
\hline
2 & 3 \\
\hline
4 & 0 \\
\hline
(8) &
\end{array}$$

(8) → not lies in base 6
So assume it as base 10
Convert to base 6

$(8)_{10} = (12)_6$

So

$$\begin{array}{cc}
 & 3 \\
1 & 2 \\
2 & 3 \\
4 & 0 \\
\hline
(8)_{10} & 2 \\
(12) &
\end{array}$$

Ans. 122

Q find possible value of x.

$(123)_5 = (x8)_y$

Ans

Step 1 → find all condition.

Step 2 → Convert into base 10

Soln ① $x < y$, $y > 8$

② $1\times5^2 + 2\times5^1 + 3\times5^0 = xy + 8$

$\Rightarrow 38 = xy + 8$

$xy = 30$

x	y	Satisfy Condition
1	30	✓
2	15	✓
3	10	✓
5	6	×
6	5	×
10	3	×
15	2	×
30	1	×

Possible value of x = 1, 2, 3

Q $(123)_x = (12x)_3$ find x.

① $x > 3$ from LHS

$x < 3$ from RHS

So, we can say that there is no value of x exist.

Imp

If you solve it without checking condition you will definitely get an Answer as,

$x^2 + 2x + 3 = 9 + 6 + x$

$\Rightarrow x^2 + x - 12 = 0$

$\Rightarrow x = 3, -4$.

→ But Answer is value of x doesn't exist.

Q Determine the base of the systems from following relations:

a) $38 + 43 = 80$

Apply bases of all,

$(38)_b + (43)_b = (80)_b$

$\Rightarrow 3b + 8 + 4b + 3 = 8b$

$\Rightarrow 7b + 11 = 8b$

$\Rightarrow \boxed{b = 11}$

→ Arithmatic operation can be performed on same bases. So we apply same bases to each numbers.

→ value of base must satisfy the condition. Here the only condn is $x > 8$.

b) Roots of $x^2 - 11x + 22 = 0$ are 3 and 6.

Product of roots $= \frac{c}{a} = (3_b)\times(6_b)$

sum of roots $= -b/a = (3_b) + (6)_b$

$\boxed{3_b + 6_b = \frac{(22)_b}{(1)_b}}$ — ①

$(3)_{10} + (6)_{10} = (2b + 2)_{10}$

$2b + 2 = 18$

$b = 8$ Satisfy the eqn ①

③ $\frac{66}{6} = 11$

$\frac{6b + 6}{6} = b + 1 \Rightarrow \boxed{b + 1 = b + 1}$

$b \rightarrow$ can be any value

→ but b must be greater than 6 and can hold any value (>6)

Imp **Ex-8**

Q In a number system, the range is $[-3, 3]$, represented as C, B, A 0, 1, 2, 3. Express $(102)_{10}$ in this number system.

→ Base = 7 of number system, bcoz of 7 numbers in range.

7	102
7	15 — $-3 = C$
7	2 — 1
	0 — 2

$= \underline{(21C)_7}$

$2\times7^2 + 1\times7 - 3 = 102$

Complementary number System

Advantages:

Subtraction can be done using addition

Base - b → (b-1)'s complement (diminished radix complement); b's complement (radix complement)

for a number x, n → no. of digits

(b-1)'s complement → $(b^n-1)-x$

b's complement → $b^n - x$

Ex → Base 2

1's Comp : $\bar{x} = (2^n-1)-x$

2's comp : $\bar{x} = 2^n - x$

Base 3:

2's complement : $\bar{x} = (3^n-1)-x$

3's complement : $\bar{x} = 3^n - x$

So, value of 2's complement is different in both base 2 and base 3.

Base 10:

9's complem.: $(10^n-1)-x$

10's complem.: (10^n-x)

Similarly we can compute complements for each base.

Examples → Base 10

$x = (685)_{10}$

9's complement $= (10^3-1)-658$
$= 999-685$
$= 314$

$x = (214)_{10}$

9's comp. $= 999-214$
$= \underline{785}$

→ for 9's complement subtract the no. from (999):

→ for 10's complement add 1 to 9's complement.

Base 2: $x = (1011)_2$

$n = 4$ (no. of digits)

1's comp $= (2^4-1) - 1011$
$= (1111)-(1011)$
$= (0100)_2$

To get 1's complement just flip the bits in the number.

→ XOR of any number with (111...1) gives 1's complement.

Base: 16 $x = (1234)_{16}$

15's comp. $= (16^4 - 1) - x$

$= FFFF - 1234$

$= EDCB$

$(b^n - 1)$ is the max. no. represent by base b with 'n' digits

16' complement $= EDCB + 1$

$= EDCC$

why we use complements?

We perform subtraction using addition as,

Suppose we want to perform

$y - x$

$\Rightarrow y + (-x)$

To find negation of no. 'x',

$-x = (b^n - 1) - x = \bar{x}$

$\Rightarrow \bar{x} + x = (b^n - 1) - x + x$

$\Rightarrow x + \bar{x} = (b^n - 1)$

and since $x + \bar{x} \to 0$ so,

$(b^n - 1) \to 0$ (forcefully suppose)

for Base 2: $(2^n - 1) = 0$

$\Rightarrow 1111 \ldots 1$ considered as 0

meanwhile value should not be '0' but it is so it's a disadvantage.

for Base 3: $(3^n - 1)$ 222 ... 2

and it will consider as zeroes.

and 000 ... 0 also zero. So there are two patterns that represent 0, as same as base 2.

Now consider b's complement instead of (b-1)'s complement,

So, $y - x = y + (-x)$ (say)

$-x = \bar{x} = (b^n - x)$

$\Rightarrow x + (\bar{x}) = b^n = 0$

↳ Additive Inverse

x and $\bar{x}$ are n bits number but b^n is $(n+1)$ bit number.

$b^n = 1$ 'n zeroes']: 1 followed by n zeroes.

So, eliminate 1 to make it as 1 bit number so, '0' is represent only by 1 pattern $00 \ldots 0$.

So, in any base we can use radix complement for subtraction

Subtraction in diminished radix complement:

Base 10.

9's complement

10's complement

① 9's complement

Ex → $97 - 23$

$\Rightarrow 97 + (-23)$

9's comp. of $23 = 99 - 23$

$= 76$

$\Rightarrow 97 + 76 = 173$

Explanation $x - y = x + \bar{y}$

$\Rightarrow x + (10^2 - 1 - y)$

$\Rightarrow 10^2 - 1 + (x - y)$

$\Rightarrow 99 + (x - y) = x + \bar{y}$

$\Rightarrow 99 + (x - y) = 173$

$\Rightarrow 73 = (99 - 100) + (x-y)$

$\Rightarrow (x-y) = 73+1$
$= 74$

Or
$$\begin{array}{r} 97 \\ +76 \\ \hline 1\,73 \\ +1 \\ \hline 74 \end{array}$$
end around carry
74 Ans

$(x-y) = x+\bar{y} = x+(b^n-1-y)$
$= b^n-1+(x-y)$

there we be a carry if
$x+\bar{y} > b^n-1$, b^n contains an extra bit and total $(n+1)$ bits.

So, $\boxed{x-y \geq 1}$ → carry generated
Imp $\boxed{x-y \leq 0}$ → no carry

If you have carry then remove the carry bit represented by b^n and add 1 means

$\cancel{b^n}-1+(x-y)+1$
$=(x-y)$ and we get the answer but if no carry generated then to get $(x-y)$ we have to remove (b^n-1) and for this we take (b-1) complement of $b^n-1+(x-y)$ as

$(b^n-1)-(b^n-1+x-y)$
$\Rightarrow -(x-y)$
and then negate $(x-y)$.

Ex: $23-97$

$\Rightarrow 23+(-97)$
$x+(-y)$

$$\begin{array}{r} \bar{y} = 99 \\ -97 \\ \hline 02 \end{array}$$

$x+\bar{y} = 23+02 = 25$

$x+(99-y) = 99+(x-y)$
Since in 25 we have no carry means
$-(x-y) = (25)$ 9'complement
so, $(x-y) = -(99-24)$
$= -74$

→ So here is two cases with carry or without carry.

Examples on diminished radix Complement.

(b-1)'s complement

1. Base 10

87 − 6

Sol^n
$$\begin{array}{r} 87 \\ -06 \\ \hline \end{array}$$

9's complement of 06 = 93

then,
$$\begin{array}{r} 87 \\ +93 \\ \hline 1\,80 \\ +1 \\ \hline 81 \end{array}$$
Carry obtained
Ans!!

87 → x
93 → $b^n - 1$ y
(1)80 → $(b^n - 1) + (x - y)$
x +1 x Add 1
81 → (x−y)

(2) 06
− 87

9's complement (87) = 12

06
+ 12
18] No carry

9's complement of 18 and then negate it
= − (9' comp. of 18)
= − (81) Ans

(3) 456
− 213

9' com. of 213 = 786

456
786] Carry
(1)242
+1
243 Ans

(4) 213
− 456] No carry

9's comp. (456) = 543

213
+ 543
756

Ans = − (9' comp. of 756)

$\boxed{-243}$ Ans

Base 2 (b−1) : 1's complement
1's complement.

1011 → 11
− 0101 → 5

1's comp (0101) = 1010

1011
+ 1010
(1)0101] with carry
+1
0110 → (6)

(2) 0101 → 5
− 1011 → 11

1's comp. of (1011) → 0100

0101
+ 0100
1001

Ans − (1' comp. of 1001)
= − (0110)
= − 6

(3) 10100
− 00111

1's comp. (00111) = (11000)

10100
11000
(1)01100
+ 1
01101

Ans = 13

Note → Number of digits must be same in both operand. In case if both are not equal then add 0 on vacant digits.

Base 3 : (b-1)'s complement

2's complement

$$\begin{array}{r} 212 \\ -121 \\ \hline \end{array}$$

(2's comp. of 121) = 101

$$\begin{array}{r} 212 \\ +101 \\ \hline 1\,020 \\ +1 \\ \hline 021 \end{array}$$

$$\begin{array}{r} 2 \\ +1 \\ \hline 3 \end{array}$$ → Not in base 3

So assume it in base 10 and convert it to base 3.

$(3)_{10} = (10)_3$

because it there

②

$$\begin{array}{r} 121 \\ -212 \\ \hline \end{array}$$

2's comp. of 212 = 010

$$\begin{array}{r} 121 \\ 010 \\ \hline 201 \end{array}$$

2's comp. of (201) = (021)

Ans −(021).

Examples on subtraction in radix complement

b's complement

Base 10

$$\begin{array}{r} 97 \\ -23 \\ \hline \end{array}$$

10's comp. of 23 = 77

$$\begin{array}{r} 97 \\ 77 \\ \hline 1\,74 \end{array}$$

No need to do end around carry just discard it.

Ignore

→ Just discard the carry if obtained.

(i) $(x-y) \geq 0$ we get a carry and ignore it.

(ii) $(x-y) < 0$, no carry just take approach given below.

$$\begin{array}{r} 23 \\ -97 \\ \hline \end{array}$$

10's comp. (97) = 03

$$\begin{array}{rl} 23 & \rightarrow x \\ +03 & \rightarrow 10^2 - y \\ \hline 26 & \rightarrow 10^2 - (x+y) \end{array}$$

10's comp of (26) = 64

Ans = −64.

②

$$\begin{array}{r} 456 \\ -132 \\ \end{array}$$

10's comp. (132) = 868

$$\begin{array}{r} 456 \\ +868 \\ \hline 1\,324 \end{array}$$

324 Ans.

discard

Base 2 : 2's complement

$$\begin{array}{rl} 1011 & \rightarrow 11 \\ -0101 & \rightarrow 5 \\ \hline \end{array}$$

2's comp (0101) = 1011

$$\begin{array}{r} 1011 \\ +1011 \\ \hline 1\,0110 \end{array}$$ → 6 Ans.

discard

→ We can say that b's complement is better than (b-1)'s complement bcoz of No end around carry.

Base 3 (3's complement)

$(212)_3$
$-(121)_3$

3's complement (121) = 102

$$\begin{array}{r} 212 \\ 102 \\ \hline (1)\,021 \end{array} \qquad \begin{array}{r} 2 \\ +2 \\ \hline 4 \end{array} \qquad (4)_{10} = (2)_3 \quad 2 = (11)_3$$

discard ⇒ 021 Ans.

→ Similarly we done do all the problems regarding all the bases.

Summary of subtraction using complements in case of unsigned integers

① (b-1)'s complement

The subtraction of two n-digits unsigned numbers M−N (N≠0) in base 'b' can be done as follows:

1) Add M to (b-1)'s complement of N. This performs $M + (b^n - 1 - N) = b^n - 1 + (M-N)$

2) If $(M-N) \geq 1$, then sum will produce an end carry b^n will be discarded and '1' is added (EAC) to gives $(M-N)_0$

3) If $(M-N) < 1$, then sum doesn't produce an end carry. To obtain the answers in familiar form, take (b-1)'s complement of the sum and place a negative sign in front.

b' complement [Not need to read]

In 1) replace (b-1)'s by b's, $M + (b^n - 1 - N)$ by $M + (b^n - N)$

2) Replace $(M-N) \geq 0$

3) Replace $(M-N) < 0$

→ Just do the examples and relate the theory. b's complement might have some incomplete theory. So don't need to read this

Signed Number Representation

Binary Numbers
- Unsigned Number
- Signed Numbers
  - Sign magnitude form
  - signed 1's comp.
  - signed 2's comp. (Imp)

Number
- Integer (10) → fixed point representation
- fraction (.014) → fixed point representation
- Integer with fraction (10.14) → floating point represent.

→ In fixed point representation the position of decimal is fixed either rightmost or leftmost side of number.

Example on signed number representation (watch the video again & make notes on next page)

3 bits	Unsigned	Signed Mag.	1's	2's
000	0	0	0	0
001	1	1	1	1
010	2	2	2	2
011	3	3	3	3
100	4	-0	-3	-4
101	5	-1	-2	-3
110	6	-2	-1	-2
111	7	-3	0	-1

Look at the page on RHS of this page

In signed representation
1 → use to represent (-ve) at MSB
0 → ———————— (+ve) at MSB

+ve number representation always remain same everywhere.

→ 0 represented as (0) and (-0) but both work as same and also (-0) ~~decreases~~ disadvantage.

→ we required additional hardware.

~~for 1's complement.~~

→ $y = -y$ means, negative number can be represented as 1's comp of positive counterpart.

-0 = have positive counter part as 0 [000] and 1's complement of 000 is [111] = -3,

so, $-0 = -3$.

for (-3) [111] we have 3 as a (+ve) part and its 1's complement is 1's comp. of (011) as 100 equals to 4 but we can't place it bcoz 4 is not present in upper (½)

So, we know

$x - x = x + \bar{x} = 0$

$x + \bar{x} = x + (2^n - 1) - x = 0$

$\Rightarrow 2^n - 1 = 0$

$\Rightarrow 1111\ldots n = 0.$

So all 1's can be assumed as 0 in 1's complement signed magnitude representation

→ So disadvantage is 0 get two combinations.

→ Adv. that both addition and subtraction need same hardware.

→ Also we have to take care of end around carry.

for 2's complement

take positive part and compute 2's complement

001 → 1 (+ve part of 101)
2's comp → (111)
So, 111 represent -1.

010 → 2 (+ve part of 110)
110 → 2's complement (-2)

011 → 3 (positive part of 111)
101 → (-3)

000 → 0

Cont.

→ Notes taken are not sufficient to justify the concept. So we did it again

Examples on signed number representation

	Unsigned	Signed mag.	1's	2's
0 0 0	0	0	0	0
0 0 1	1	1	1	1
0 1 0	2	2	2	2
0 1 1	3	3	3	3
1 0 0	4	−0	−3	−4
1 0 1	5	−1	−2	−3
1 1 0	6	−2	−1	−2
1 1 1	7	−3	0	−1

→ In unsigned we talk only about positive no. greater than equals to 0.

→ In Signed repres. '0' at MSB shows no. is positive and '1' at MSB shows (−ve) numbers.

011 → (3) (+ve) 111 → (−3) (−ve)

→ Signed representation holds signed mag., 1's and 2's complement all the 3.

In sign mag upper half represent positive and lower half represent negative.

−0' doesn't make sense but it happens and it also waste the combination (100) becoz (000) is already alloted to 0.

→ To check sign we need additional hardware.

→ So we generally don't use it.

→ 1's complement

In 1's complement, negative number is going to represent 1's complement of positive part

Ex (−1) represent 1's complement of (1).

$1 = 001$ 1's comp. $= 110$

$(-1) = (-2)$

−3 represent 1's comp. (3)

$(\overline{011}) = (100) = (-0)$

−2 represent 1's comp. of (2)

$(\overline{010}) = 101 = (-1)$

we can't represent (−4) bcoz it doesn't present in upper half.

So, to fill value in front of (−3), we do,

$$x - x = x + \bar{x} = 0$$

$$x + \bar{x} = (2^n - 1) - x + x$$

$$x + \bar{x} = (2^n - 1) = 0$$

1 1 1 1 .. n represent 0

(−0) represents 1's comp of 0

$(\overline{000}) = (111)$

So we can put 0 in all 1's.

→ So, 0 get two combinations here. (Disadvantage)

→ But here we don't need different hardware for both add or sub parts

→ It have also disadv. that it has end around carry.

→ So, it also useful to some extent only.

2's complement

(-1) represent as 2's complement of (+ve) part

2's(001) = 111 represent -1

-2 represented by 2's comp. of 010

110 → -2

Similarly 101 represent -3

for (-4) 2's comp. of (100) = 100

So, 100 is used as -4.

→ Only 1 combination used for 0.

→ Due to -4 range get increased.

→ No additional hardware require.

→ No end around carry.

→ So its the best way for operation.

Weighted Code

Unsigned representation is a weighted code bcoz

$\underset{2^2}{\downarrow}$ $\underset{2^1}{\downarrow}$ $\underset{2^0}{\downarrow}$ each bit have some weight and we

can decode the number using bit combination and weights

Ex 010 can be decoded as

$0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 2$

Similarly signed. mag. is also a weighted code with MSB fixed for sign.

S	2^1	2^0

$1\,0\,0 = (-1)^1 (2^1 \times 0 + 2^0 \times 0)$
$= -0$

S = 1 here.

S = {0, 1} only two values can hold.

1's and 2's complement both are non weight and weighted respectively.

for 2's complement

-2^2	2^1	2^0	
1	0	0	= -4
1	0	1	= -3
0	0	1	= 1
0	1	0	= 2

→ Points to write while doing PYQ's

Ranges of signed no. representation

	US	Signed mag	1's	2's
0000	0	0	0	0
0001	1	1	1	1
0010	2	2	2	2
0011	3	3	3	3
0100	4	4	4	4
0101	5	5	5	5
0110	6	6	6	6
0111	7	7	7	7
1000	8	−0	−7	−8
1001	9	−1	−6	−7
1010	10	−2	−5	−6
1011	11	−3	−4	−5
1100	12	−4	−3	−4
1101	13	−5	−2	−3
1110	14	−6	−1	−2
1111	15	−7	0	−1

Range: for n-bits Number

Unsigned : $[0 \to (2^n - 1)]$

Signed Mag. : $(-(2^{n-1}-1)) \to (2^{n-1}-1))$

1's Complement : $(-(2^{n-1}-1) \to (2^{n-1}-1))$

2's complement : $(-(2^{n-1}) \to (2^{n-1}-1))$

→ If size or no. of bits can't be represented by register called as overflow condition.

Ex → 1

Find the min. of bits to represent given no. in 2's complement.

1. $(+32)_{10}$
2. $(-32)_{10}$

In 2's complement range is

-2^{n-1} to $(2^{n-1}-1)$

(1)

$2^{n-1}-1 \geq 32$

$2^{n-1}-1 \geq 2^5$

$2^{n-1} \geq 33$

$n-1 \geq \log_2 33$

$n-1 \geq 5.$

$n \geq 6.\ldots$

$n = 7$ Ans

(2)

$-2^{n-1} \leq -32$

$2^{n-1} \geq 32$

$n-1 \geq 5$

$\boxed{n \geq 6}$ $\boxed{n = 6}$ Ans

Ex-2 Find the max. +ve no. that can be represented by 10 bits in 2's complement.

Max no $= (2^{n-1}-1)$

$= 2^9 - 1$

$= 511$

Sign. bit Extension

Suppose you want to store a 4 bit no. in 8 bit register

0	0	0	0	0	1	1	0

↳ Remaining 4 bits filled as 0's but it happends only in case of unsigned numbers.

In signed magnitude representation,

$-6 = (1110)$

$\boxed{0\ 0\ 0\ 0 \mid 1\ 1\ 1\ 0}$

↳ It changes the entire no. to (+14) from (−6) because MSB is the sign bits that becomes '0' here.

So, we have to do something as

1110 (−6)

$\boxed{1\ 0\ 0\ 0 \mid 0\ 1\ 1\ 0}$

So, first bit of register is sign. bit and we fill 0's in between.

1's complement

① 0110 → +6

Positive no. doesn't make any difference so fill all remaining bits with 0.

② (1001) → −6

then take positive part means 6 as 0110, fill all the vacancies with 0 as

0000 0110 and then

takes 1's complement.

$\boxed{1\ 1\ 1\ 1 \mid 1\ 0\ 0\ 1}$ → −6

means in 1's complement, instead of '0' fills 1.

2's complement : It is weighted code.

$$\begin{array}{cccc} -2^3 & 2^2 & 2^1 & 2^0 \\ 1 & 0 & 0 & 1 \end{array}$$

$-8+1 = 7$.

$$\begin{array}{ccccc} -2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ 1 & 1 & 1 & 0 & 1 \end{array}$$

$-16+8+4+1 = -3$

$$\begin{array}{cccc} -2^3 & 2^2 & 2^1 & 2^0 \\ 1 & 1 & 0 & 1 \end{array}$$

$-8+4+1 = -3$

$$\begin{array}{ccc} -2^2 & 2^1 & 2^0 \\ 1 & 0 & 1 \end{array}$$

$-4+1 = -3$

If we have a chain of 1's in MSB we can replace all 1's with a single one and we get the same ans.

→ for positive no. in 2's complement is same as others.

→ for negative no

$$\begin{array}{cccc} -2^3 & +2^2 & 2^1 & 2^0 \\ 1 & 1 & 0 & 1 \end{array} = -3$$

$-8+4+1 = -3$

Put 1's in vacant slots

$\boxed{1\ 1\ 1\ 1 \mid 1\ 1\ 0\ 1}$

all 1's can merge to a single 1's. So doesn't matter.

imp → FFF3 is a 2's complement no. convert it into base 10.

1111 1111 1111 0011

↳ (1)0011

All bits convert to single 1

$-16+3 = -(13)$ Ans

Ex → A 2's complement number $N = P_3 P_2 P_1 P_0$ is transformed as $P_3 P_3 P_3 P_2 P_1' P_0 P_0 1$ which of the following operation is performed on this number.

Sol^n

$P_3 \ P_3 \ P_3 \ P_2 \ P_1 \ P_0 \ 1$

1 1 → If P_3 is -ve

or

0 0 → If P_3 is +ve

So, we can replace all P_3 with single P_3

$\boxed{P_3 \ P_2 \ P_1 \ P_0 \ 1}$

As 15 can be written as

$1 \times 10 + 5 = 15$ [for decimal]

So $P_3 \ P_2 \ P_1 \ P_0 \ 1$ equals to

$\underline{P_3 P_2 P_1 P_0} \ 1$

$2 \times (\underbrace{P_3 P_2 P_1 P_0}_{N}) + 1$

$2 \times N + 1$

↳ bcoz no. is binary.

So operation is $\underline{2N+1}$.

Overflow

→ If you add two n bits no. then result could be as max (n+1) digits, so that 1 extra digit is overflow.

① Unsigned No.

4 bits $(0 \rightarrow 2^4 - 1)$

$$\begin{array}{rl} 1000 & \rightarrow 8 \\ 1100 & \rightarrow 12 \\ \hline (1)0100 & \rightarrow 20 \end{array}$$

$$\begin{array}{rl} 1010 & \rightarrow 10 \\ 0110 & \rightarrow 6 \\ \hline (1)0000 & \rightarrow 16 \end{array}$$

→ In case of unsigned carry indicates overflow.

② 2's complement

[-8 to 7] using 4 digits.

$$\begin{array}{rl} 1000 & \rightarrow -8 \\ 1100 & \rightarrow -4 \\ \hline 10100 & \rightarrow (-12) \end{array}$$

Here carry indicates overflow

$$\begin{array}{rl} 1010 & \rightarrow (-6) \\ 0110 & (6) \\ \hline 10000 & (0) \end{array}$$

Here carry generated but we can ignore it. So it is not overflow.

So, in signed representation carry doesn't always posses overflow.

$$\begin{array}{rl} 0111 & - 7 \\ 0001 & - 1 \\ \hline 1000 & \rightarrow 8 \end{array}$$] No carry

but 8 doesn't lies in range (-8, 7). So its overflow.

→ On adding two positive no. you get a negative no. or on adding no negative no. you get a positive no then there is overflow.