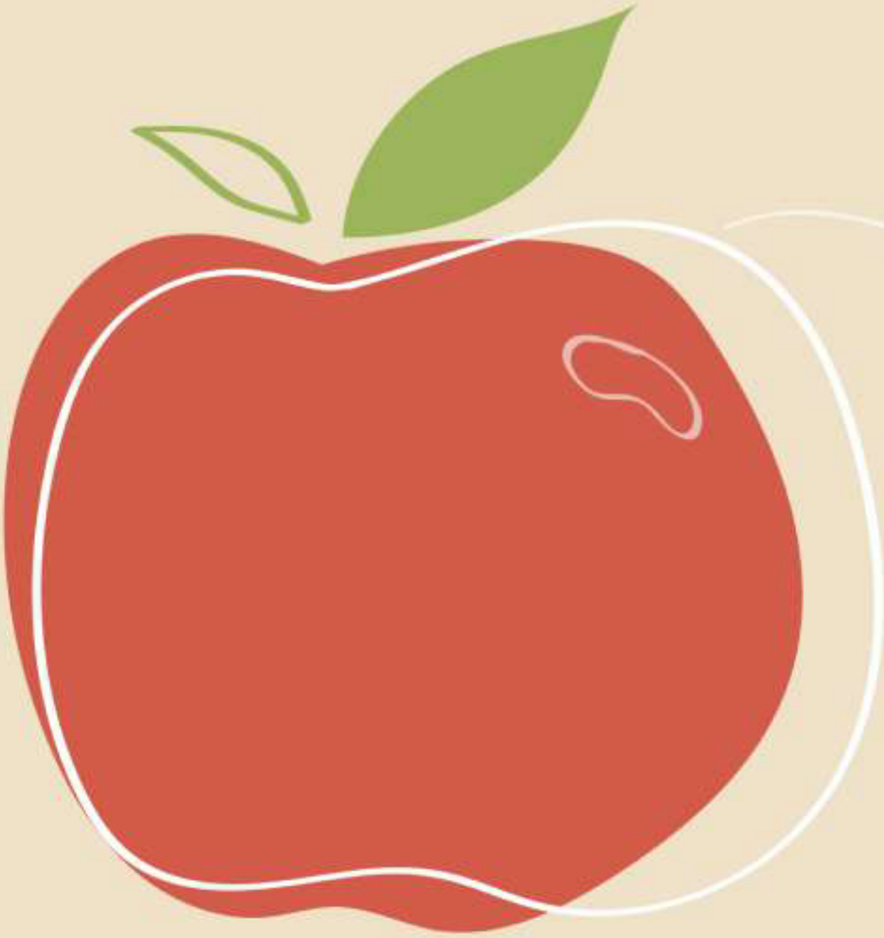
CIE-1 : unit + 1st half of unit-2



Unit - 1

BINARY NUMBERS

- unsigned
- signed
  - i) sign & magnitude
  - ii) 1s complement
  - iii) 2s complement

Sign and Magnitude :

consider MSB : 0101 : zero : +ve , one : -ve

MSB → 0 (sign), 101 (magnitude)

Representation :

b_3	b_2	b_1	b_0	sign & mag	1s comp	2s comp
0	1	1	1	+7	+7	+7
0	1	1	0	+6	+6	+6
0	1	0	1	+5	+5	+5
0	0	1	0	+2	+2	+2
0	0	0	0	+0	+0	+0
1	0	0	0	-0	-7 (1111)	-8
1	0	0	1	-1	-6	-7
1	0	1	0	-2	-5	-6
1	0	1	1	-3	-4	-5
1	1	0	0	-4	-3	-4

1	1	0	1	-5	-2	-3
1	1	1	0	-6	-1	-2
1	1	1	1	-7	-0	-1
				[0, 7] [-7, -0]	[0,7] [-7, -0]	[0, 7] [-8, -1]
				$-2^{n-1}-1$ to $2^{n-1}-1$	$-2^{n-1}-1$ to $2^{n-1}-1$	-2^{n-1} to $2^{n-1}-1$

1s complement is obtained by subtracting the number from $2^n - 1$ ie $(2^n - 1)$ - number

for 4 bit : $2^4 - 1$

2s complement is obtained by subtracting the number from 2^n ie 2^n - number

for 4 bit : 2^4

for unsigned : 5 = 0 1 0 1

$2^3 \; 2^2 \; 2^1 \; 2^0$

AKA : 8 4 2 1 code

for signed : 1 1 0 1

$\boxed{-2^3} \; 2^2 \; 2^1 \; 2^0$

to convert it : $-1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

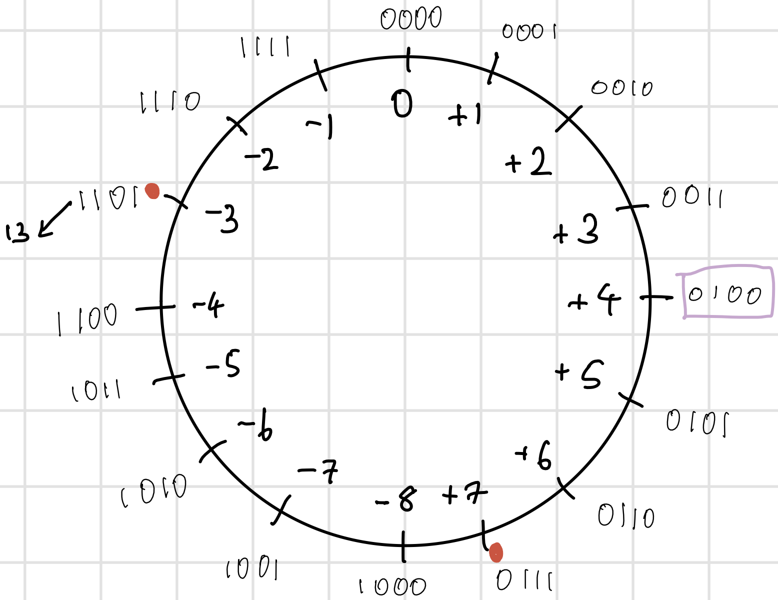
$\Rightarrow -8 + 4 + 0 + 1$

$= -3$

$$\text{weight} = (-b_{n-1} \times 2^{n-1}) + \sum_{i=0}^{n-2} (b_i \times 2^i)$$

2s complement : no ambiguity cause it doesn't have 2 diff representations for zero

4 - bit



to calculate +7 $\boxed{-3}$ = +4 $\quad$ ∴ 13 steps from 7 (~~clockwise~~)

steps : 1- locate numbers on circle

2- take binary representation of the second number

3- move those many numbers clockwise on the circle from the position of the first number

subtract (-7 - 5) ⇒ -7 + 5 = -2

Multiplication of binary numbers

$$
\begin{array}{cccccccccl}
 & & & & & P_3 & P_2 & P_1 & P_0 & \\
 & & & & & 1 & 1 & 0 & 1 & \text{multiplicand (13)} \\
 & & & & \times & 1 & 0 & 1 & 1 & \text{multiplier (11)} \\
\hline
 & & & & & 1 & 1 & 0 & 1 & \\
 & & & 1 & 1 & 1 & 0 & 1 & x & \\
 & & 1 & 0 & 0 & 0 & 0 & x & x & \\
 & 1 & 1 & 1 & 0 & 1 & x & x & x & \\
\hline
 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & \\
 & P_7 & P_6 & P_5 & P_4 & P_3 & P_2 & P_1 & P_0 &
\end{array}
$$

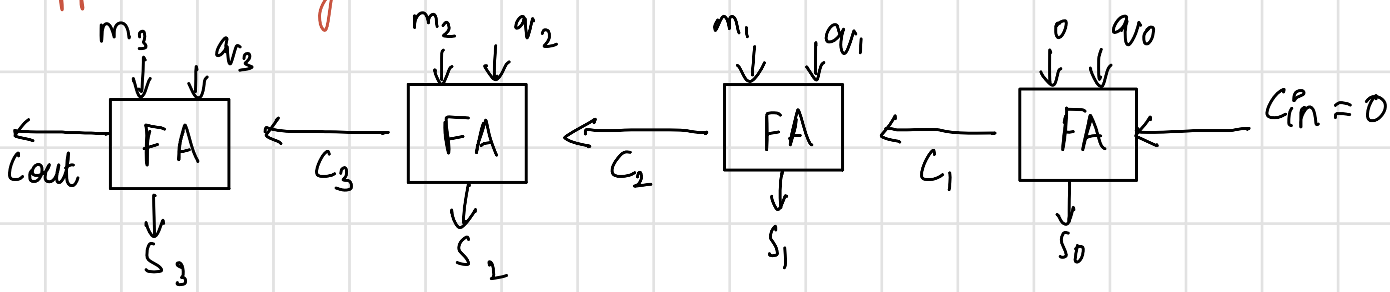
consider multiplicand, keep it as it is.
consider multiplier, in each step starting from the rightmost bit, considering the weight (therefore shift by 1 in each step) multiply that bit with the multiplicand and finally add all

flip-flop : stores 1 bit
register : multiple flip flops, can store multiple bits

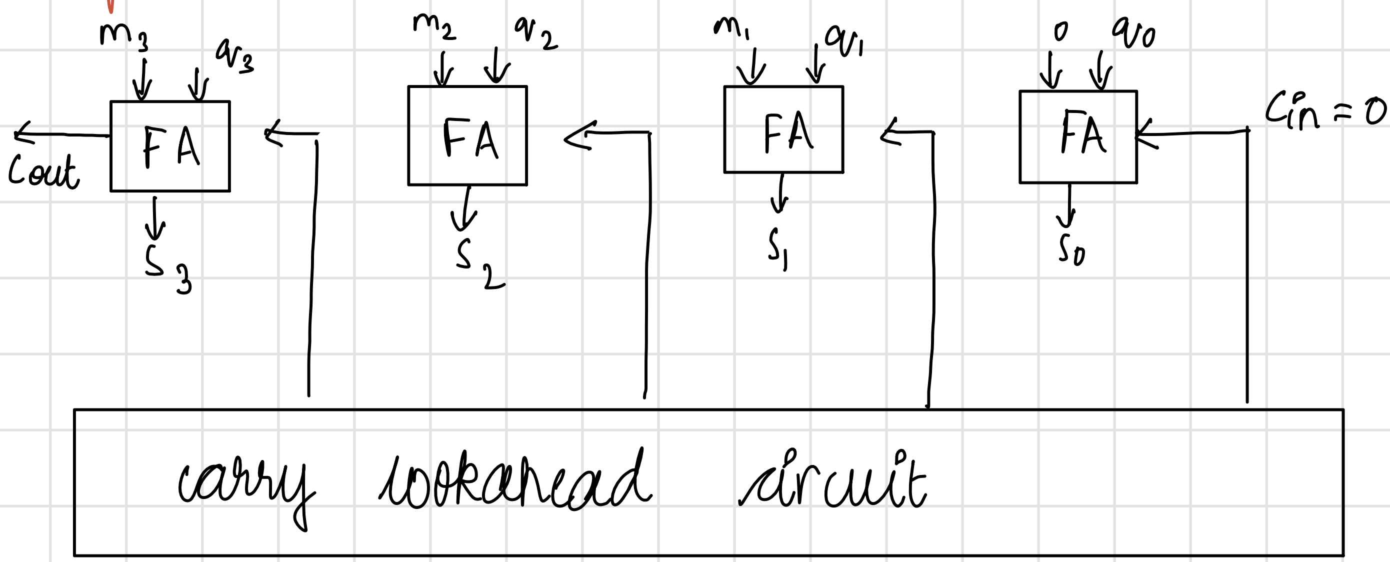
full adder : adds 1 bit
parallel adder : collection of full adders

PARALLEL ADDERS:

Ripple carry adder :-



carry lookahead circuit :-



→ more space, but less time ie less delay

Sequential Circuit Multiplier :-

For each bit in Q

$q_i \to 0 \to$ no addition
perform right shift

$q_i \to 1 \to$ add M with A
perform right shift

Arithmetic right shift :

used with signed numbers

keep MSB as it is and shift the rest like usual

eg: 1 1 0 0 (−4) (its in 2s comp form)
1 1 1 0 → −2

Example:

M = 1101 multiplicand , Q = 1011 multiplier

M is 1101

C	a_3	a_2	a_1	a_0	q_3	q_2	q_1	q_0	
0	0	0	0	0	1	0	1	1	
	1	0	1	1					A + M
0	1	0	1	1	1	0	1	1	
0	0	1	0	1	1	1	0	1	rightshift
	1	1	0	1					A + M
1	0	0	1	1	1	1	0	1	
0	1	0	0	1	1	1	1	0	rightshift
0	0	1	0	0	1	1	1	1	zero so no A+M direct right shift
	1	1	0	1					A + M
1	0	0	0	1	1	1	1	1	
0	1	0	0	0	1	1	1	1	right shift → ANSWER

note: no of operations to be performed is equal to the no of bits, here → its 4

Booths Multiplier:

A − multiplicand
B − multiplier
Q − B
Q_0 − LSB of Q register
Q_{-1} − 1 bit variable / register
A_{cc} − holds intermediate result

Q_0	Q_{-1}	A
0	0	+ 0.M
0	1	+ 0. M
1	0	− 0. M
1	1	+ 0. M

multiply

M Q
6 x 7 = 42

6 = 0110 2s = 1010
7 = 0111 2s : 1001

A				Q				Q_{-1}	
0	0	0	0	0	1	1	1	0	↳ -1 x M
1	0	1	0						
1	0	1	0	0	1	1	1	0	↳ right shift
1	1	0	1	0	0	1	1	1	↳ 0 x M, right shift
1	1	1	0	1	0	0	1	1	↳ right shift
1	1	1	1	0	1	0	0	1	↳ 1 x M
0	1	1	0						
(1) 0	1	0	1	0	1	0	0	1	↳ right shift
0	0	1	0	1	0	1	0	0	
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0		

maintain

⟶ final ans without Q_0

Fast Multiplier

Multiplier Bits Block			Recoded 1-bit pair		2 bit booth	
i+1	i	i-1	i+1	i	Multiplier Value	Partial Product
0	0	0	0	0	0	Mx0
0	0	1	0	1	1	Mx1
0	1	0	1	-1	1	Mx1
0	1	0	1	0	2	Mx2
1	0	0	-1	0	-2	Mx-2
1	0	1	-1	1	-1	Mx-1
1	1	0	0	-1	-1	Mx-1
1	1	0	0	0	0	Mx0

Q multiply 13 x (-6)
-6 : 1010
2s : 1010

-6:

1 1 | 1 0 1 0 | 0

0 0 −1 1 −1 0

2^1 2^0 2^1 2^0 2^1 2^0

$\underbrace{0}_{(0\times2^1+0\times2^0)}$ $\underbrace{-1}_{(-1\times2^1+1\times2^0)}$ -2 $(-1\times2^1+0\times2^0)$

finish

Division :

```
              0  4  1  1  → quotient
divisor → 3 ) 1  2  3  4  → dividend
              1  2  ↓
              -------
                 0  3
               - 0  3  ↓
                 -------
                    0  4
                  - 0  3
                  -------
remainder →            1
```

2 methods :

1. restoring division
2. non restoring division

1	2	3	4
10^3	10^2	10^1	10^0

restoring for decimals :

now, $1234 - 3 \times 10^3 = -1766$, $q_3 = 1$

↓ restoring

$-1766 + 3 \times 10^3 = 1234$, $q_3 = 0$

$1234 - 3 \times 10^2 = 934$, $q_2 = 1$

$934 - 3 \times 10^2 = 634$, $q_2 = 2$

$634 - 3 \times 10^2 = 334$, $q_2 = 3$

$334 - 3 \times 10^2 = 34$, $q_2 = 4$

$34 - 3 \times 10^2 = -266$, $q_2 = 5$

$-266 + 3 \times 10^2 = 34$, $q_2 = 4$

$34 - 3 \times 10^1 = 4$, $q_1 = 1$

$4 - 3 \times 10^1 = -26$, $q_1 = 2$

↑ restore

$-26 + 3 \times 10^1 = 4$, $q_1 = 1$

$4 - 3 \times 10^0 = 1$, $q_0 = 1$

↑

STOP → if its less than divisor here, its 3

restoring for binary division :

$Q = 01010_{(2)}$ → dividend

$M = 00011_{(2)}$ → divisor

$-M = 11101$ (2s complement of M)

	A (n+1 bits)	Q (n bits) $q_3\ q_2\ q_1\ q_0$	
shift left	0 0 0 0 0	1 0 1 0	
sub M from A	0 0 0 0 1	0 1 0 _	
	1 1 1 0 1		
	$\boxed{1}$ 1 1 1 0	0 1 0 $\boxed{0}$	I
	0 0 0 0 1	0 1 0 $\boxed{0}$	I
left shift	0 0 0 1 0	1 0 0 _	I
sub M	1 1 1 0 1		I
or this	$\boxed{1}$ 1 1 1 1	1 0 0 $\boxed{0}$	II
restore (add M)	0 0 0 1 1		II
× $\boxed{1}$	0 0 0 1 0	1 0 0 0	III
left shift	0 0 1 0 1	0 0 0 _	III
sub M	1 1 1 0 1		III
× $\boxed{1}$	$\boxed{0}$ 0 0 1 0	0 0 0 $\boxed{1}$	IV
left shift	0 0 1 0 0	0 0 1 _	IV
sub M	1 1 1 0 1		IV
× $\boxed{1}$	$\boxed{0}$ 0 0 0 1	0 0 1 $\boxed{1}$	IV
	(1)	(3)	

→ restoring : works for +ve & -ve nos

→ non restoring : works for +ve nos only last step might need restoration

non restoring :

Q = 01010

M = 00011

-M = 11101

	A					Q			
shift left	0	0	0	0	0	1	0	1	0
sub M	0	0	0	0	1	0	1	0	_
	1	1	1	0	1				
	1	1	1	1	0	0	1	0	0
Shift left	1	1	1	0	0	1	0	0	_
sub M	1	1	1	0	1				
x $\boxed{1}$	1	1	1	0	1	1	0	0	0
Shift left	1	1	0	1	1	0	0	0	_
sub M	1	1	1	0	1				
x $\boxed{1}$	1	1	0	0	0	0	0	0	0
Shift left	1	0	0	0	0	0	0	0	
sub M	1	1	1	0	1				
x $\boxed{1}$	0	1	1	0	1	0	0	0	1

gaive to add M if its -ve result

FIXED PT & FLOATING PT :

fixed : radix point is fixed, faster, limited range
floating : radix pt is dynamically adjusted, large range of values
↳ largest no
↳ smallest no

fixed	Floating
1.25	1.1
23.71	12.12
118.75	123.123

→ need to represent as exponent, mantissa

eg : $1.234 = 1.234 \times 10^0$

$12.123 = 1.2123 \times 10^1$

$123.123 = 1.23123 \times 10^2$

The process of doing this → AKA normalisation

0 → +ve
1 → -ve

sign bit	exponent	mantissa
1 bit	4 bits	6 bits

examples (binary) :

1. $1.0011 = 1.0011 \times 2^0$
2. $1011.001001 = 1.011001001 \times 2^3$
3. $-11.10 = -1.110 \times 2^1$
4. $0.001101 = 1.101 \times 2^{-2}$

implicit normalisation

explicit

implicit & explicit normalisation :

implicint : $(-1)^{sign} \times 1.M \times 2^E$

M → mantissa
E → exponent

eg : $1.0011 = 1.0011 \times 2^0$

0	0000	000011

IEEE format :

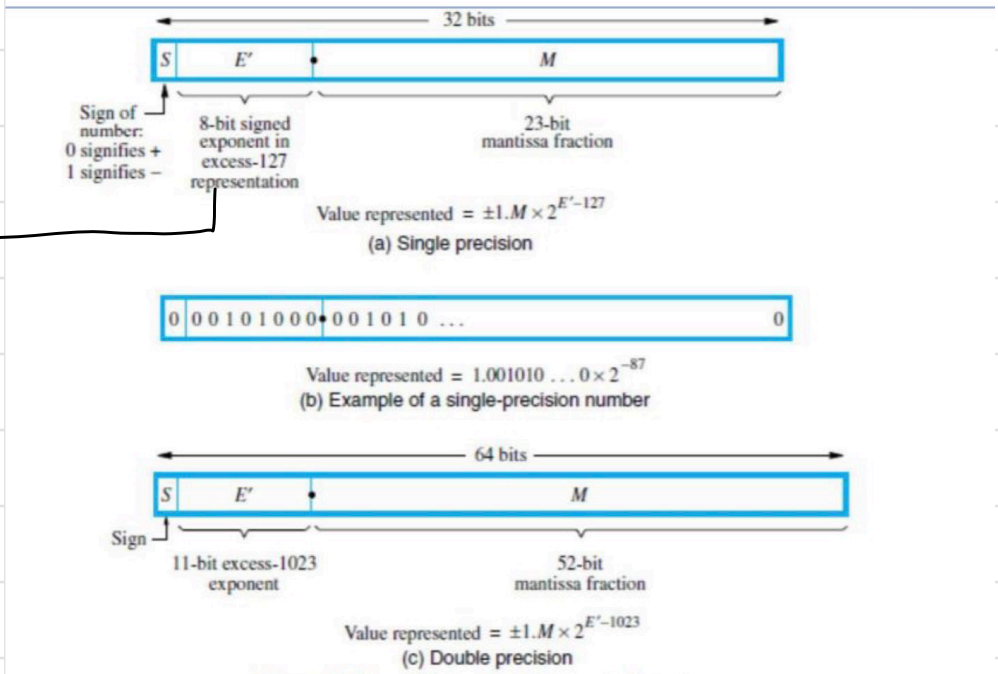


Figure 9.26 IEEE standard floating-point formats.

biased exponent
-126 to 127

Observations based on :-

	sign	exp	mantissa	represents
1)	0 or 1	00000000	M = 0	± 0
2)	0 or 1	11111111	M = 0	± ∞
3)	0 or 1	$1 \leq E \leq 254$	$M \neq 0$	$(-1)^S \times 1.M \times 2^E$ ↳ implicit
4)	0 or 1	E = 0	$M \neq 0$	$(-1)^S \times 0.M \times 2^E$ ↳ explicit
		E = 255	$M \neq 0$	NaN

1   8   32 bits

0	10000011	1100 00

→ find decimal form

exp = 131 − 127 = 4

$1.1100 \times 2^4 = 11100 = \underline{\underline{28}}$

explicit demonstration : $5.625_{(10)}$

$$5.625 = \underset{2^2\ 2^1\ 2^0}{101} . \underset{2^{-1}\ 2^{-2}\ 2^{-3}}{101}$$

$0.625 \times 2 = 1.25 \quad 1$

$1.25 \times 2 = 0.5 \quad 0$

$0.5 \times 2 = 1 \quad 1$

$= 0.101101 \times 2^3$

$(-1)^{sign} \times 0.M \times 2^E$

0	1 0 1 1 0 1	0 1 1

↳ unbiased for explanation

explicit gives you one decimal of exponent or mantissa & hence if you're using high nos you'll get a diff ans
∴ implicit is better

Adding floating pt numbers :

1) chose the no with smaller exponent & shift its mantissa right by a number of steps = diff in exponents
2) set exponent of smaller = larger
3) perform addition / subtraction on mantissa
4) Normalize if necessary

1) Add 01001.110
+ 010010.1001

$1.001110 \times 2^3 = 0.1001110 0 \times 2^4$

$1.00101001 \times 2^4 \quad + 1.00101001 \times 2^4$

$= 1.1100110 \times 2^4$

Multiplication:

1. add exponents & subtract 127 to maintain excess 127 format
2. multiply mantissa, determine sign
3. normalize

division:

1. subtract exponents, add 127
2. divide mantissa, determine sign
3. normalize.

K-MAP (karnaugh map)

variable	cells	shape
2	4	2x2
3	8	2x4
4	16	4x4
5+	3D kmap	

2 variable

	A \ B	$\bar{B}$ 0	B 1
$\bar{A}$	0	0_0	1_1
A	1	1_2	1_3

A	B	F (OR gate)
0	0	0
0	1	1
1	0	1
1	1	1

3 variable

		$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
	A \ BC	00	01	11	10
$\bar{A}$	0	0	1	3	2
A	1	4	5	7	6

4 variable

		$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
	AB \ CD	00	01	11	10
$\bar{A}\bar{B}$	00	0	1	3	2
$\bar{A}B$	01	4	5	7	6
AB	11	12	13	15	14
$A\bar{B}$	10	8	9	11	10

note: grey code means 1 bit differs in consecutive cells

Q) $y = A\bar{B}C + \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + \bar{A}\bar{B}\bar{C}$

A	B	C	y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

A \ BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	1 (0)	1 (1)	1 (3)	(2)
A	1 (4)	1 (5)	(7)	(6)

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

Q) F(A, B, C, D) = 1 for

A	B	C	D	
0	0	0	0	✓
0	1	0	0	✓
0	0	1	0	✓
0	1	0	1	✓
0	1	1	0	✓
1	1	0	1	✓
1	1	1	0	✓

AB \ CD	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	1 (0)	(1)	(3)	1 (2)
$\bar{A}B$	1 (4)	1 (5)	(7)	1 (6)
AB	(12)	1 (13)	(15)	1 (14)
$A\bar{B}$	X (8)	X (9)	(11)	(10)

$$\boxed{\bar{A}\bar{D} + C\bar{D}B + \bar{C}DB}$$

x → indicates don't care

- consider don't care only if it can be used to make a larger group
- you should not consider a group of only don't cares

Q)

Input			Output		
x	y	z	A	B	C
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	0	1	1
1	0	1	1	0	0
1	1	0	1	0	1
1	1	1	1	1	0

for output A

	A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$	0	(0)	(1)	1 (3)	(2)
A	1	(4)	1 (5)	1 (7)	1 (6)

for output B

	A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$	0	(0)	1 (1)	(3)	1 (2)
A	1	1 (4)	(5)	1 (7)	(6)

for output C

	A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$	0	1 (0)	(1)	(3)	1 (2)
A	1	1 (4)	(5)	(7)	1 (6)

for $A = BC + AC + AB$

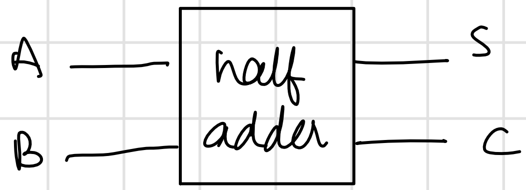
for $B = \bar{A}\bar{B}C + A\bar{B}\bar{C} + ABC + AB\bar{C}$

for $C = \bar{C}$

unit - 2

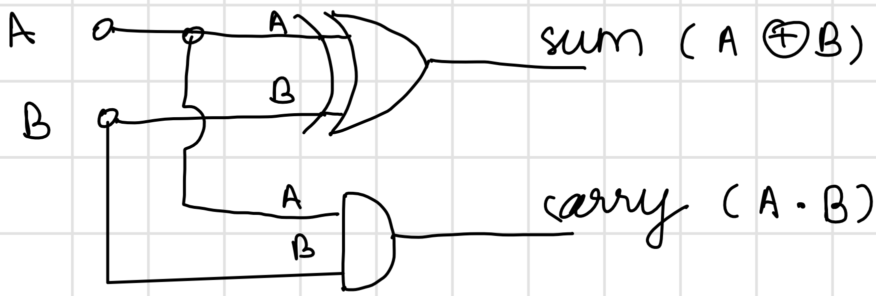
HALF ADDER

A	B	sum	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



$sum = A \oplus B$

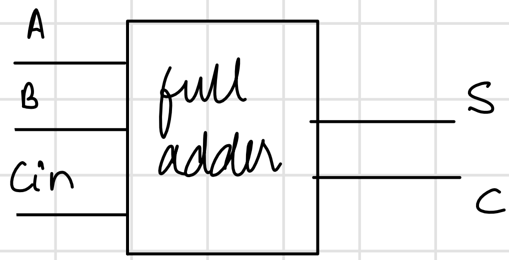
$carry = A \cdot B$



KMAP :

FULL ADDER

A	B	C_{in}	Sum	carry
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



$$\text{sum} = A \oplus B \oplus C$$

$$\text{carry} = (A \cdot B) + (B \cdot C_{in}) + (C_{in} \cdot A)$$