→ Decimal to Binary
÷2

→ Decimal to Octal
÷8

→ Decimal to Hexadecimal
÷16

→ Binary to Octal
-- group of 3 bits

→ Binary to Hexadecimal
group of 4 bits

→ Octal to Binary
covert each to 3 bits

→ Octal to Hexa
→ binary ↗

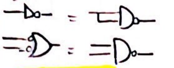
→ Octal to Decimal
multiply each digit with power of 8

→ Hexadecimal to Octal
→ decimal ↗

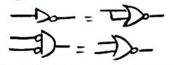
→ Binary to Gray Code
1 0 0 1
XOR
1 1 0 1

→ Gray Code to Binary
1 1 1 0
1 0 1 1

→ NAND Gate
↳ make bubbles at o/p of AND & i/p of OR



→ NOR Gate
↳ same as above and
↳ apply NOT gates where bubble
↳ cancel NOT gate in series



Minterm : 1 Maxterm : 0 for Maxterm $x=0$, $x'=1$

★ Converting function into sum of minterms / sum of product SOP
EXAMPLE: $A+B'C$
→ expand expression into sum of AND terms
→ check if each term has all variables;
$A(B+B')(C+C') + B'C(A+A')$
$ABC + ABC' + AB'C + AB'C' + A'B'C$ → SOP form
m_7 m_6 m_5 m_4 m_1
$\Sigma(1,4,5,6,7)$ → canonical form

★ Converting function into product of maxterms / POS
EXAMPLE: $xy+x'z$
→ converting into OR term using $A+BC=(A+B)(A+C)$ distributive law.
$F=(x'+y)(x+z)(y+z)$
→ check if each term has all variables.
$(x'+y+zz')(x+z+yy')(y+z+xx')$
$F=(x+y+z)(x+y'+z)(x'+y+z)(x'+y+z')$ → POS form
$\Pi(0,2,4,5)$ → canonical form.

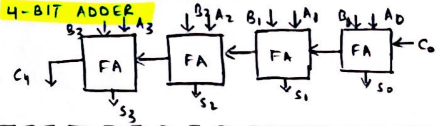
→ POSTULATES

Postulate 2 (a) $x+0=x$ (b) $x\cdot 1=x$
Postulate 5 (a) $x+x'=1$ (b) $x\cdot x'=0$
Theorem 1 (a) $x+x=x$ (b) $x\cdot x=0$
Theorem 2 (a) $x+1=1$ (b) $x\cdot 0=0$
Theorem 3, involution $(x')'=x$
Commutative (a) $x+y=y+x$ (b) $xy=yx$
Distributive (a) $x(y+z)=xy+xz$ (b) $x+yz=(x+y)(x+z)$
Demorgan (a) $(x+y)'=x'y'$ (b) $(xy)'=x'+y'$

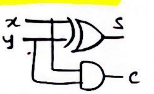
→ $A+A=A$
→ $A\cdot A=A$
→ $A+AB=A$
→ $A(A+B)=A$
→ $A+\bar{A}B=A+B$

Associative (a) $x+(y+z)=(x+y)+z$ (b) $x(yz)=(xy)z$
Absorption (a) $x+xy=x$ (b) $x(x+y)=x$
→ $A\cdot(\bar{A}+B)=AB$

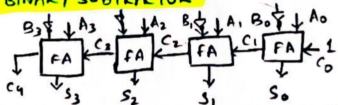
4-BIT ADDER



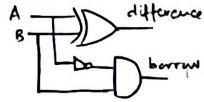
HALF ADDER



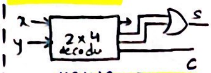
BINARY SUBTRACTOR



HALF SUBTRACTOR

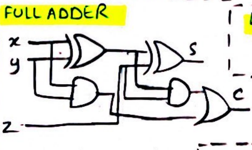


HALF ADDER

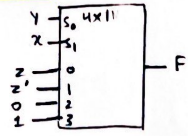


USING DECODER

FULL ADDER

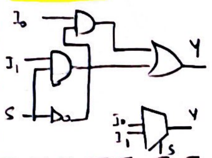


4x1 MUX

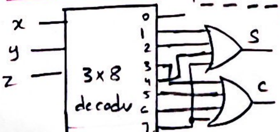


	x	y	z	F	
0	0	0	0	0	F = z
	0	0	1	1	
1	0	1	0	1	F = z'
	0	1	1	0	
2	1	0	0	0	F = 0
	1	0	1	0	
3	1	1	0	1	F = 1
	1	1	1	1	

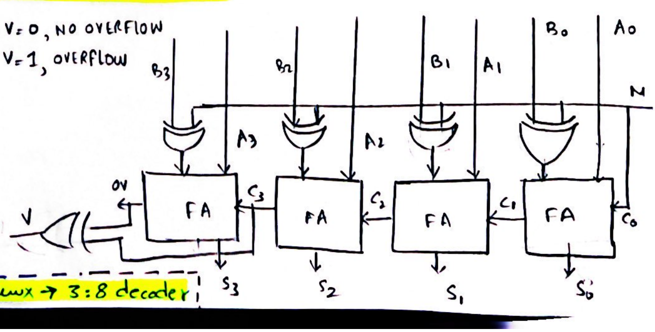
2x1 MUX



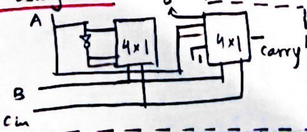
→ using 3x8 decoder



BINARY ADDER SUBTRACTOR
→ if V=0, NO OVERFLOW
→ if V=1, OVERFLOW



→ using MUX

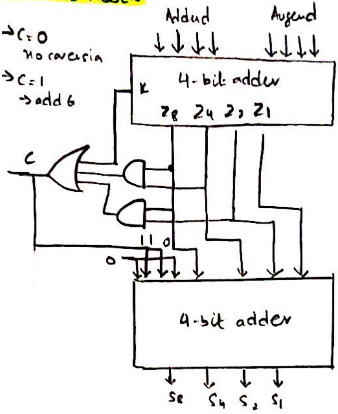


1:4 Demux → 2:4 decoder // 1:8 demux → 3:8 decoder

DECIMAL ADDER

→ C=0 no conversion

→ C=1 → add 6

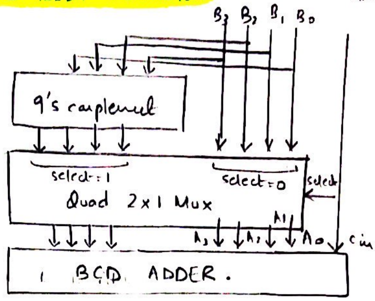


BCD-ADDER SUBTRACTOR

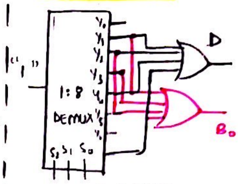
*not necessarily 9's complement

Mode = 0 for ADD

Mode = 1 for SUBTRACT



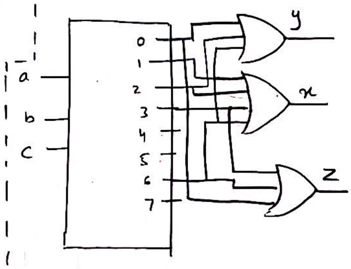
FULL SUBTRACTOR



→ using 4x1 MUX

Gray-code to Excess 3-Code.

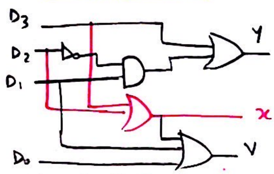
→ using 3x8 decoder.



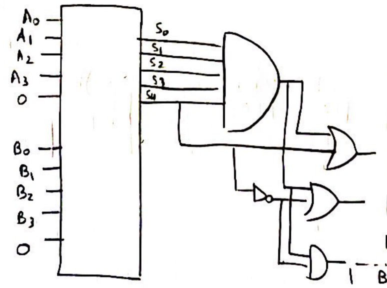
~~5 BIT SUBTRACTOR~~

K+

PRIORITY ENCODER



5 BIT SUBTRACTOR

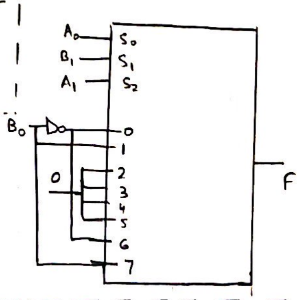


EQUALITY CHECKER

→ using 8x1 MUX

$A_1A_0 = B_1B_0$ (2 bit binary)

high when A=B



Decimal	2421	Gray code	excess 3
0	0000	0000	0011
1	0001	0001	0100
2	0010	0011	0101
3	0011	0010	0110
4	0100	0110	0111
5	1011	0111	1000
6	1100	0101	1001
7	1101	0100	1010
8	1110	1100	1011