COSC 130 (11/13)

- Truth Table

Inputs　　　output

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

2^n = number of rows
- n is # inputs

$A \wedge B = Q$

- Converting to equations
  - Sum of Products (SoP)
    - OR = A+B　A | B　A v B
    - AND = A•B　A & B
    - NOT = $\bar{A}$　A'
    - Run thru AND$^{(\bullet)}$ gate then OR$^{(+)}$ gate

OR gate

	A	B	Q
	0	0	0
$\bar{A}B$	0	1	1
$A\bar{B}$	1	0	1
AB	1	1	1

→ $\bar{A}B + A\bar{B} + AB = Q$

SOP
1. Invert inputs that are 0
2. Up/down a row → ADD (OR)
3. Across a row → MULTIPLY (AND
4. Look where output = 1

if A=0, B=1　$\bar{0}1 + 0\bar{1} + 01$

$11 + 00 + 01 = 1 + 0 + 0 = 1$

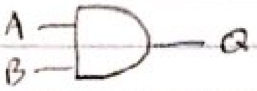
- Product of Sums (PoS)
  1. Look for ouput = 0
  2. Across row → OR (add)
  3. Up/down row → AND (multiply)
  4. Invert inputs = 1

OR gate

	A	B	Q
A+B	0	0	0
	0	1	1
	1	0	1
	1	1	1

→ A+B = Q

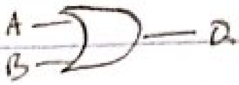
AND Gate



$Q = AB$

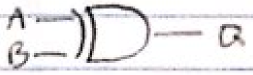
$Q = A \& B$

OR Gate



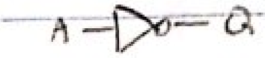
$Q = A + B$ $Q = A | B$

XOR Gate



$Q = A \oplus B$

NOT Gate

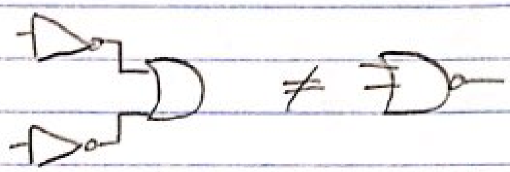


$Q = \sim A$ $Q = A'$

$Q = \bar{A}$ $Q = \neg A$

○ inverter bubble can be added to anything

- De Morgan's Law



- Can change OR's → AND's, vice ve
  - Where they're bubbles, remove
  - Where no bubbles, add

NOR

$Q = (A + B)' \Rightarrow Q = (\bar{A} \cdot \bar{B})$

$Q = (A + B)'$

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

$Q = (\bar{A} \cdot \bar{B})$

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

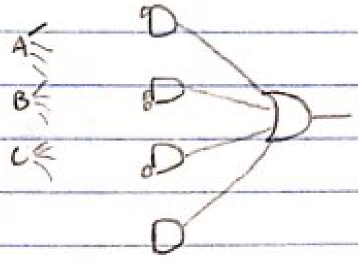
MAKE THIS

$C = 0 \Rightarrow Q = A$

$C = 1 \Rightarrow Q = B$

USE SOP

A	B	C	Q
0	0	0	0
0	0	1	0
0	1	0	0 SOP
0	1	1	1 ←
1	0	0	1 ← SOP
1	0	1	0
1	1	0	1 ← SOP
1	1	1	1 ← SOP

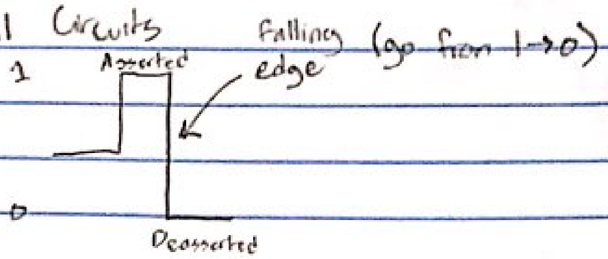


$Q = \bar{A}BC + A\bar{B}\bar{C} + AB\bar{C} + ABC \Rightarrow$ 4 AND's

Cose 130 (11/15)

Sequential Circuits

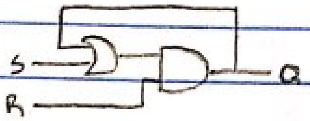
Latch I



- If output is fed back as input, it's sequential

SR Latch Truth

S_{et}	R_{eset}	Q
0	0	Q_{prev}
0	1	0
1	0	1
1	1	0



Sum Truth

Left	Right	S
0	0	0
0	1	1
1	0	1
1	1	0

Carry Truth

Left	Right	C
0	0	0
0	1	0
1	0	0
1	1	1

Data Latch (D-Latch)

(D) Data	(E) Enable	Q
0	0	Q_{prev}
0	1	0
1	0	Q_{prev}
1	1	1

E=1, get D

E=0, get Q_{prev}

D-Flip-Flop (DFF)

Rising edge, falling edge

Finite State Machine

- E.x.: Lamp with L-M-H bulb (need 2 bits 0-3)

State L	State R	Switch	$Q_{new\ L}$	$Q_{new\ R}$
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

$N_L = s(C_L \oplus C_r) + C_L s$

$N_R = C_r \oplus s$