

# Lecture 1: Introduction to EEE/CSE 120

Bahman Moraffah

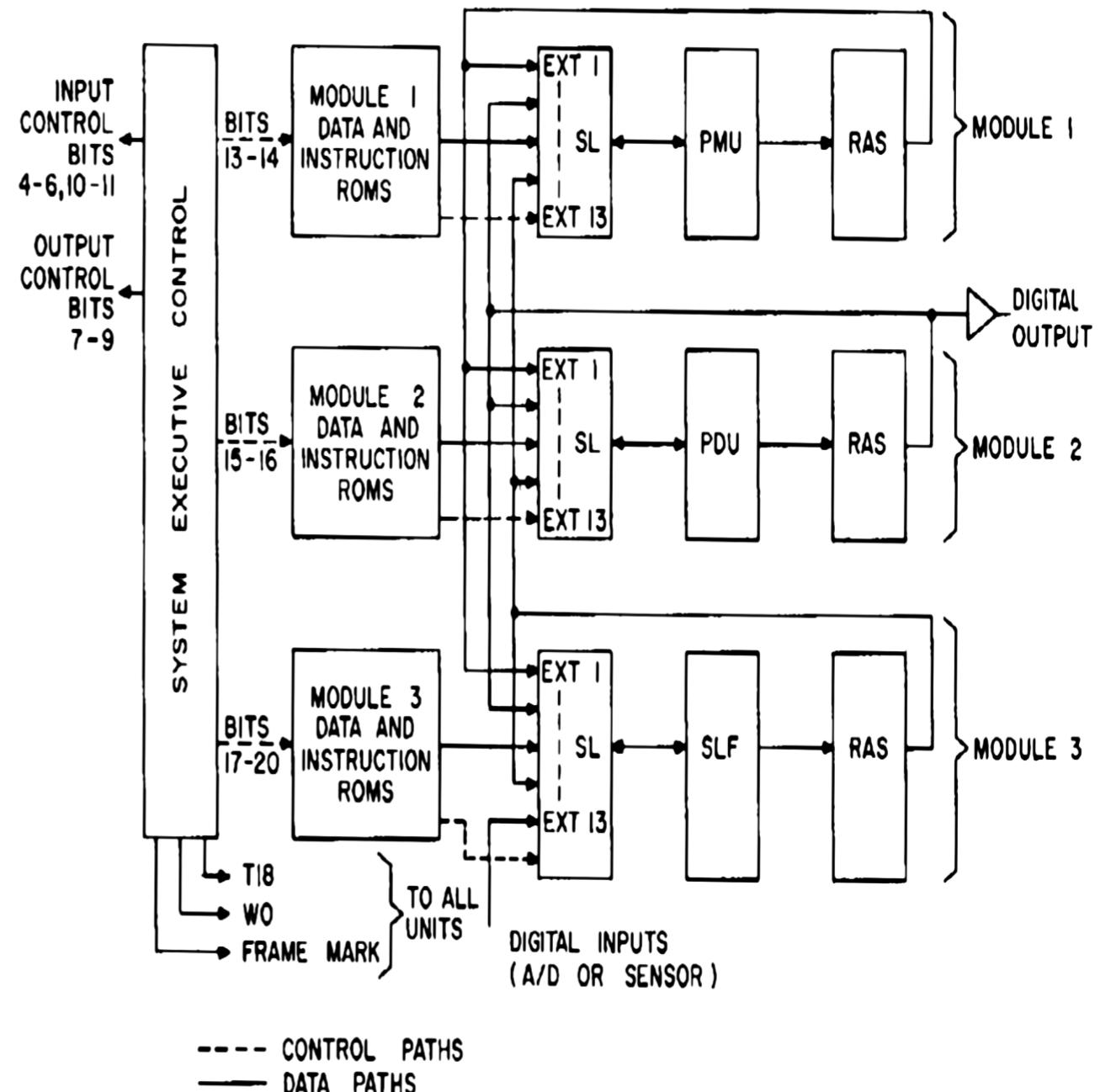
Electrical, Computer, and Energy Engineering  
Arizona State University

# Why do we learn digital circuit analysis

- We cannot store information in an analog way.
- Computers often chain logic gates together, by taking the output from one gate and using it as the input to another gate.
- Circuits enables computers to do more complex operations than they could accomplish with just a single gate.
- Logic gates are used to define the state of a system that has many inputs and outputs so that more complex units are created such as arithmetic units, shift registers, memory elements etc.
- Programs are written which manipulate these complex units giving what you see on the screen of your computer etc.

# Cool Example: CPU

- 8 bit CPU
- Princeton Architecture  
(von Neumann model )
- Few 8 bit registers
- Microcode sequence



70's CPU

# How this class will work

- Instructor: Dr. Bahman Moraffah
- Grader: William Shih
- UGTA: Alden Davison
- TAs: TBD
- Class will be in-person or via Zoom
- Office Hours are virtual, and you should join using its own zoom meeting (It is different from Zoom link for the lectures)
- All of us should keep checking both Canvas and Course website for updates!
- Class is cumulative, so keep up with the materials and assignments!
- 5-6 assignments + Lab reports. They all must eb submitted on Canvas. No email or in-person submission will be accepted.

# Logistics

- **Class meet:** Tuesday and Thursday  
In-Person (Alphabetically) or through zoom

Find the zoom link on Canvas!

- **Office Hours:** T TH 9:30-10:15 am (Virtually through zoom).

The link for office hours is  
different from the lectures!

- **Course**  
**Website:** [https://bmoraffa.github.io/EEE\\_CSE120\\_Fall2020.html](https://bmoraffa.github.io/EEE_CSE120_Fall2020.html)
- **Canvas:** <https://canvas.asu.edu>
- **My Email:** Bahman.Moraffah@asu.edu

# Exams and Quizzes

Exams/Quizzes	Date
Quiz 1	September 1
Quiz 2	October 1
Midterm	October 20
Quiz 3	November 3
Quiz 4	November 1
Final Exam	As scheduled by ASU

*subject to change  
+ mini quizzes*

Exams will be through Lockdown Browser. Try to familiarize yourself with it. This method records video and sound as you take quizzes and exams. You will be required to use this method to take quizzes and exams whether you are in person or participating remotely.

All quizzes and exams are closed book and notes.  
Exam and quiz dates are subject to change.

# Course Grading

	<b>Distributions</b>
Lab Report	25%
Assignment	10%
Quizzes and Attendance	15%
Capstone Project	10%
Midterm	20%
Final Exam	20%

- For the letter grade check [syllabus](#).
- I will not curve, but I will provide a lot of opportunities to earn extra credit.
- Extra Credit:** I need volunteers to take notes each class, type it up and send it to me so it can be uploaded for the entire class. Each student can scribe at most 2 lectures.
- Incorrect Work & Correct Answer = NO CREDIT.
- No Work & Correct Answer = NO CREDIT.

# ASU Sync

This course uses Sync. ASU Sync is a technology-enhanced approach designed to meet the dynamic needs of the class. During Sync classes, students learn remotely through live class lectures, discussions, study groups and/or tutoring.

- To access live sessions of this class go to the Canvas shell for this course on the side bar click on zoom and choose the lecture, you should attend.
- Classroom attendance is limited to 50% of the capacity of the room. Therefore, it may be required that you attend one day per week in person and one day per week online. For more information check the syllabus.
- If you cannot physically be on campus due to travel restrictions or personal health concerns, you will be able to attend your classes via ASU Sync during the fall semester. If you will not be on-campus for the fall semester, you are expected to contact your professors to make accommodations.
- You must attend lectures either in-person or through zoom (your choice!). Note that attendance is mandatory.

# Labs

- 5 Labs + Capstone Project
- Lab instructions are posted on Canvas.
- Capstone will be posted and discussed thoroughly in class
- **Lab Reports:**
  - Lab results (schematic diagrams, timing diagrams) will be filled into a lab template. Lab templates will be posted on Canvas.
  - Lab templates have to be completed and submitted individually. No group submissions will be accepted.
  - Copying full reports or sections of other students, except for data generated as a group effort, is considered an academic integrity violation and will be reported.
  - Students must indicate their lecture session (instructor and meeting time) as well as the names of their lab partners on the lab submission.
  - Submissions must be in electronic format (doc or pdf, no individual jpegs) and must be submitted via the submission link on Canvas. No paper or email submissions of lab reports will be accepted.
  - Late lab submissions will be penalized at a rate of 10% per day late, up to a maximum penalty of 50%. No lab reports will be accepted after 5 working days, unless there is a valid excuse.
- **Capstone Project:**
  - This lab has to be performed individually, not as a group.
  - Students have to pick a one-hour time slot within their session to demonstrate a working finite state machine design, implemented in programmable logic, to the TA, and explain the operation to the TA to be graded and approved for completion.

# Course in one glance

Thu, Aug 20	Syllabus, Introduction to EEE 120 & Electrical Fundamentals
Tue, Aug 25	Logical and Binary Systems, AND-OR, NAND-NOR Logic, Truth Tables, Realizations
Thu, Aug 27	Number Systems, Addition
Tue, Sep 1	Half Adder, Full Adder, Multi-bit Adder
Thu, Sep 3	2's Complement Representation, 2's Complement Arithmetic
Tue, Sep 8	Boolean Algebra I
Thu, Sep 10	Boolean Algebra II, SOP & POS Forms

Thu, Sep 17	The Uniting Theorem, Karnaugh Maps
Tue, Sep 22	Karnaugh Maps, Min SOP & Min POS, Don't Cares
Thu, Sep 24	MUX's, Decoders
Tue, Sep 29	MUX and DEC as Function Generators, PROMs
Tue, Oct 6	Tri State & Open Collector Buffers
Thu, Oct 8	Sequential Logic, Latches
Tue, Oct 13	Flip Flops

Thu, Oct 22	Synchronization and Registers
Tue, Oct 27	Synchronous Counters
Thu, Oct 29	Synchronous Machine Design, Moore Machine
Thu, Nov 5	Mealy Machines
Tue, Nov 10	Design Project
Tue, Nov 24	Brainless Microprocessor
Thu, Nov 26	No Class: Thanksgiving Recess
Tue, Dec 1	CPU Architecture and Microprocessor Systems

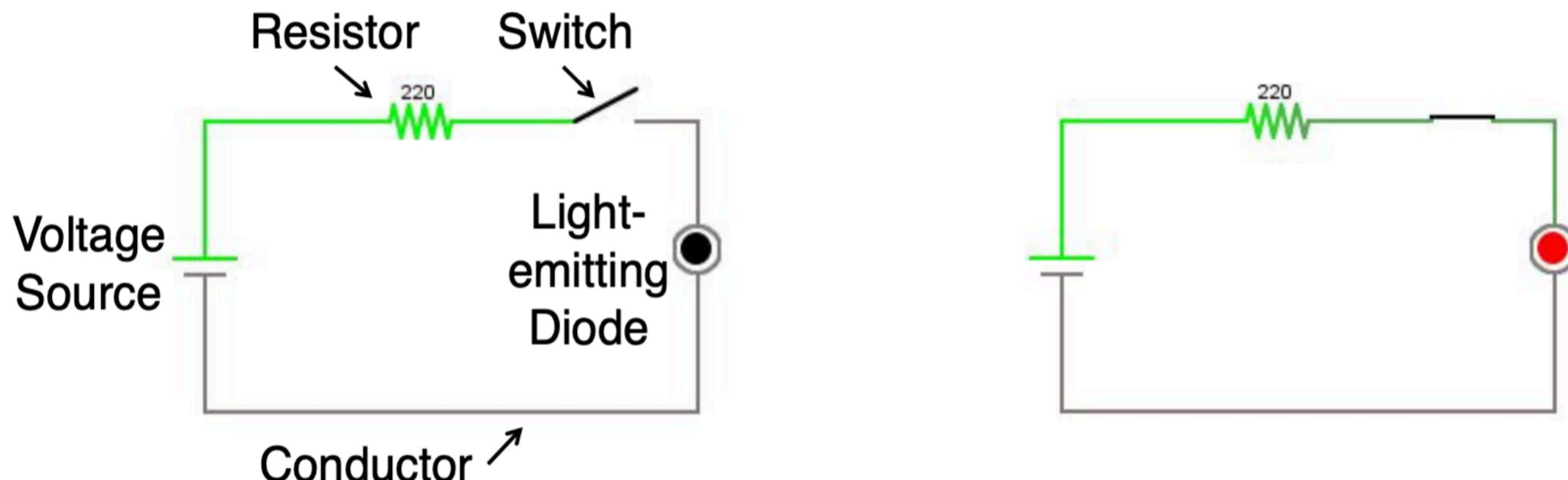
# Electrical Fundamentals

- “**Charge**” carriers are the reason why we can observe “electric” phenomena
- **Electrons** are charge carriers in metals
- **Conductors**: materials having lots of freely movable charge carriers
- **Insulators**: materials that have no freely movable charge carriers
- **Energy source**: Charge carriers with a potential energy
- **Voltage**: Electrical potential – energy needed to move a unit charge from point ‘a’ to point ‘b’
- **Current**: Amount of charge that flows per second in a conductor
- Number of charges must be **constant** at all time – closed circuit necessary (Law of Conservation of Energy)

# Why digital design

- In 17<sup>th</sup> century electricity was discovered.
- Built a switch to turn things on and off
  - They were huge switches
- Switches:
  - Insulators: Prevent the flow of electricity (Rubber)
  - Conductors: Allow the electricity to flow
  - Vacuum Tube: Amplify current/voltage and act like a switch
    - Very expensive and not durable
  - Semi-conductors
    - Make Transistors (Bell labs)
    - Better than Vacuum tubes most of the times
    - Disadvantage: Electromagnetic pulses don't work on vacuum tubes but works on transistors
- Transistors:
  - Analog → Continuous time
  - Digital → Discrete time ( We only deal with {0,1} )
- Electric circuits: Circuit diagrams are a schematic representation of the physical circuit elements and wires.

# Convention



Switch off = circuit open  
**LED off**

Switch on = circuit closed  
**LED on**

Circuit Open (FALSE) (0)	Switch off
Circuit Closed (TRUE) (1)	Switch on

ATTN:

Active high → True = 1

Active low → True = 0

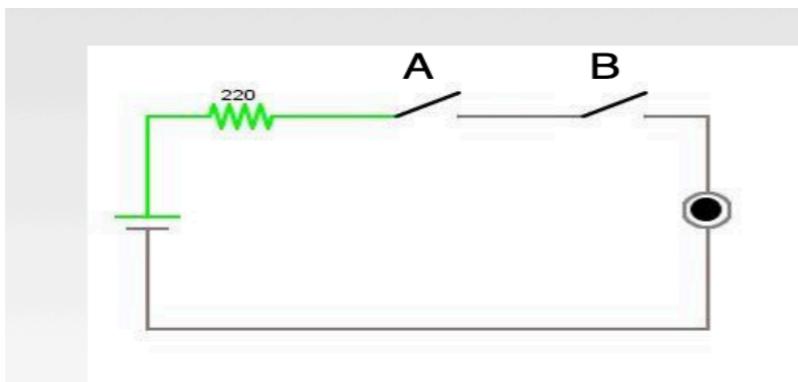
# Switches: Connection

## □ Series Connection

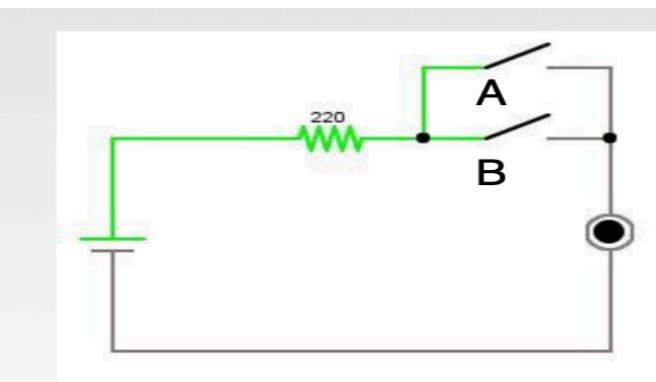
- If switches are back to back
- If two switches are in series, then circuit is closed only if switch A and switch B are both closed

## □ Parallel Connection

- If switches share the same head and tail
- If two switches are in parallel, then circuit is closed if switch A or switch B is closed



Switch A	Switch B	LED
Off	Off	Off
Off	On	Off
On	Off	Off
On	On	On



Switch A	Switch B	LED
Off	Off	Off
Off	On	On
On	Off	On
On	On	On

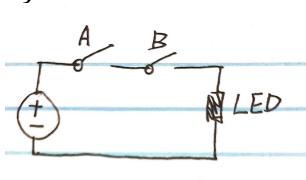
Truth Table

# Next Lecture

- Truth Tables
- And/OR/NAND/ NOR/XOR/NXOR Gates
- Truth table for each of the gates
- Analysis of the gates

How to connect switches?

1) . series: back to back



Switch off  $\leftrightarrow$  False  $\leftrightarrow$  0

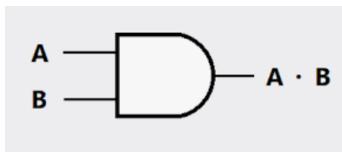
Switch on  $\leftrightarrow$  True  $\leftrightarrow$  1

For the above series, only A open or only B open, the LED will not on. LED is on if both A and B are closed.

4 possibilities:

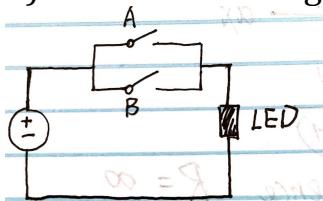
A	B	Y=Output
0	0	0
0	1	0
1	0	0
1	1	1

Truth Table shows all possibilities of outputs.



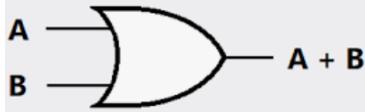
**"AND" gate** only true when both A and B are true.

2) . Parallel: sharing the same head and tail.



Two switches are in parallel, so output is on if the switches are both .

A	B	Y=Output
0	0	0
0	1	1
1	0	1
1	1	1



**“OR” gate** will be true if A is true or B is true.

**Question:** How do we design a truth table that allows the LED to turn off/on with each switch being independent of the other?

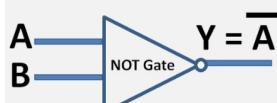
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0



**“XOR” gate**(not series, not parallel) gives true output when the number of true inputs is odd.

Explanation:

- LED is on when 1). Switch A is on **and** Switch B is off(not on) **OR**
- 2). Switch A is off(not on) **and** Switch B is on



A	Y
0	1
1	0

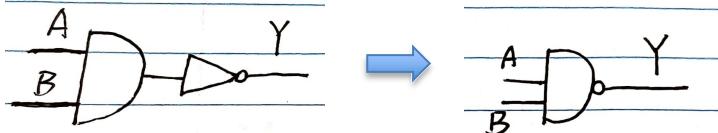
**“NOT” gate** is one the outputs the opposite state as what is input.



**“Buffer” gate** is the one that outputs equal to its inputs.

A(Input)	Y(Output)
0	0
1	1

“AND” gate combines with “NOT” gate: **“NAND” gate** (means not “AND”)



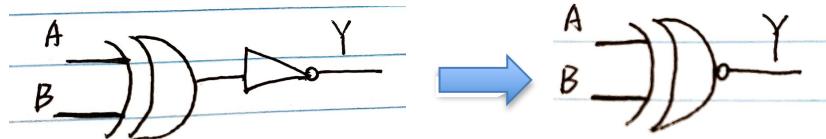
A	B	Y(final output)	C(before "not")
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

“OR” gate combines with “NOT” gate: **“NOR” gate** (means not “or”)



A	B	Y(final output)	C(before “not”)
0	0	1	0
0	1	0	1
1	0	0	1
1	1	0	1

"XOR" gate combines with "NOT" gate: "XNOR" gate



A	B	Y(final output)	C(before "not")
0	0	1	0
0	1	0	1
1	0	0	1
1	1	1	0

The number of possibilities for outputs depends on how many input(n):

The number of output:  $2^n$

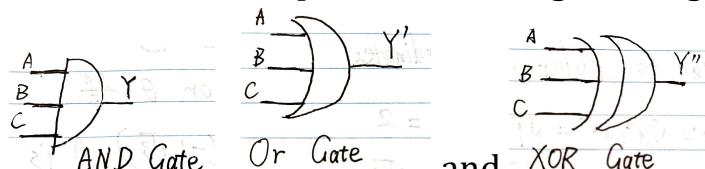
Eg. 2 inputs --- 4 outputs

3 inputs --- 8 outputs

4 inputs --- 16 outputs

### Example:

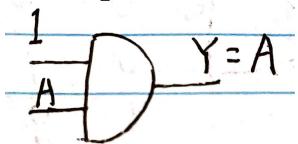
There are three inputs A, B, and C go through the following three gates



AND Gate, Or Gate and XOR Gate, list the outputs for truth table for Y, Y', and Y''.

A	B	C	Y	Y'	Y''
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	0	1	1
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	1	1	1

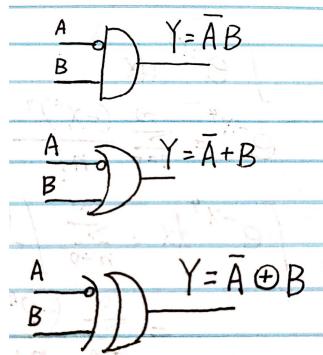
### Example 1.



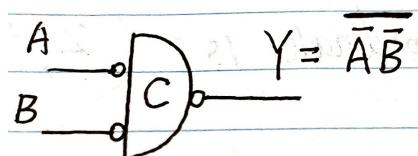
The output Y depends on A: A  
so the output Y=A



### Example 2.



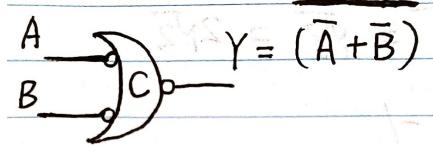
### Example 3.



A	B	Y	C
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

(the final outputs are the same as "OR" gate)

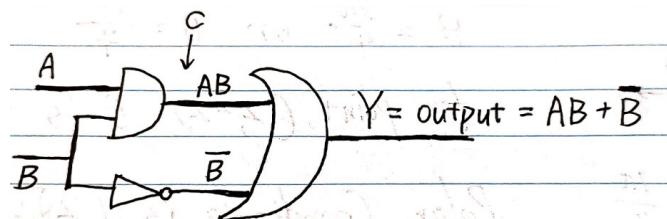
#### Example 4.



A	B	Y	C
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

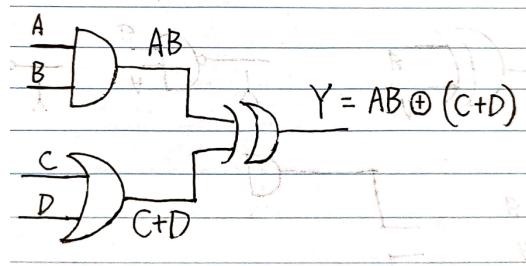
(the final outputs are the same as "AND" gate)

#### Example 5.



A	B	Y	C=AB
0	0	1	0
0	1	0	0
1	0	1	0
1	1	1	1

### Example 6.



A	B	C	D	Y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	1	0	0	0
1	0	0	0	0
1	1	0	0	1
1	0	1	0	1
1	0	0	1	1
0	1	1	0	1
0	1	0	1	1
0	0	1	1	1
1	1	1	0	0
1	1	0	1	0
1	0	1	1	1
0	1	1	1	1
1	1	1	1	0

Explanation:

- 1) . For "And" gate, its output is AB, which means only when A and B are 1, the output is 1;
- 2) . For "OR" gate, its output is C+D, which means when A is 1 and B is 0, B is 1 and A is 0, or both A and B are 1, the output is 1;
- 3) . For the final output Y, it is "XOR" gate, which means when the values of inputs are different(A is 1 and B is 0, or A is 0 and B is 1), the output Y is 1

### Lecture 3

EEE/CSE 120 : Number System.

$$(654)_{10} = 4 \times 10^0 + 5 \times 10^1 + 6 \times 10^2 = \sum_{i=0}^n c_i 10^i$$

coeff.                          base                          exponent

base = 10  $\leftrightarrow$  Decimal  $\leftrightarrow c_i < 10$

$$c_i \in \{0, 1, 2, \dots, 9\}$$

base = 2  $\leftrightarrow$  Binary  $\leftrightarrow c_i < 2 \Rightarrow c_i \in \{0, 1\}$

$$(\dots)_2$$

$$\sum_{i=0}^n c_i 2^i$$

Q: 654 in terms of powers of Two?  
 (Binary rep. of 654)

<u>Binary</u>	<u>Decimal</u>	<u>Binary</u>	<u>Decimal</u>
$2^0$	1	$2^8$	256
$2^1$	2	$2^9$	512
$2^2$	4	$2^{10}$	1024
$2^3$	8	$2^{11}$	2048
$2^4$	16	.	.
$2^5$	32	.	.
$2^6$	64		
$2^7$	128		

$$654 = 1 \times 2^9 + 1 \times 2^7 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

↓  
 $0 \times 2^8$

$$(1010001110)_2$$

$\begin{matrix} \uparrow & \uparrow \\ 2^1 & 2^0 \end{matrix}$

base $=2^3=8 \rightarrow$  octal (oct)  $\leftrightarrow c_i \in \{0, 1, 2, \dots, 7\} (c_i < 8)$

base $=2^4=16 \leftrightarrow$  HEXADECIMAL (HEX)  $\leftrightarrow c_i < 16$

$$\begin{matrix} 0-9 \\ 10 \leftrightarrow A \\ 11 \leftrightarrow B \end{matrix}$$

$$\begin{matrix} 12 \leftrightarrow C \\ 13 \leftrightarrow D \\ 14 \leftrightarrow E \end{matrix}$$

$$15 \leftrightarrow F$$

$$(1 \mid 0 \ 1 \ 1 \mid 0 \ 1 \ 1)_2$$

$\downarrow \uparrow \uparrow$   
 $2^2 \ 2^1 \ 2^0$

Example :

$$654 = (0 \mid 1 \ 0 \ 1 \ 0 \mid 0 \ 0 \ 1 \ 1 \ 1 \ 0)_2$$

Q: What is the oct?

$\underbrace{0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2}_2$

$\downarrow$

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

$\underbrace{\quad\quad\quad}_6$

$$(1 \ 2 \ 1 \ 6)_8$$

EX:

$$654 = (0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ | \ 1 \ 1 \ 1 \ 0)_2$$

$1 \times 2^1 = 2$

$2^3 = 8$

$0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 = 14$

HEX ?

E

$$(2 \ 8 \ E)_{16}$$

Example :  $(B \ 3 \ 6 \ E)_{16}$  binary?

$$E = 14 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \leftrightarrow (1 \ 1 \ 1 \ 0)_2$$

$$6 = 1 \times 2^2 + 1 \times 2^1 \leftrightarrow (0 \ 1 \ 1 \ 0)_2$$

$$3 = 1 \times 2^1 + 1 \times 2^0 \leftrightarrow (0 \ 0 \ 1 \ 1)_2$$

$$B = 11 \longleftrightarrow 2^3 + 1 \times 2^1 + 1 \times 2^0 \leftrightarrow (1011)$$
$$(1011 \text{ } 00 \text{ } 11 \text{ } 0 \text{ } 11 \text{ } 0 \text{ } 1110)_2$$

$$654 \begin{array}{r} | \\ \hline Q: 327 \end{array}$$

$$\begin{array}{r} | \\ R = 0 \end{array}$$

least significant  
digit

$$654 = 2 \times 327 + 0$$

$$327 = 2 \times 163 + 1$$

$$(1010001110)_2$$

$$\begin{array}{r} & 2 \\ R & | \\ & 654 \end{array}$$

most significant

$$\text{Ex: } \begin{array}{r} +1 \\ 7 \\ \hline 12 \end{array}$$

binary:  $\begin{array}{r} 1 \\ + 0 \\ \hline \end{array}$   $11 \rightarrow 1 \times 2^0 + 1 \times 2^1 = 11$

$+ 0 0 + 0 \rightarrow 2 +$

$$\begin{array}{r} 1 \quad | \quad 0 \quad | \\ \hline \end{array}$$

$1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 = 13$

$\checkmark$

$$\begin{array}{r} +1 \\ 0 \\ \hline 1101 \end{array}$$

How do we define negative numbers?

II Signed bit -

negative  $\rightarrow 1 (0 \ 0 \ 1) \rightarrow -1$

positive  $0 \ 0 \ 0 \ 1 \rightarrow +1$

"0"  $\rightarrow 1 \ 0 \ 0 \ 0 \quad \} \text{ two rep.}$   
 $0 \ 0 \ 0 \ 0$

$-1 \rightarrow 1 \ 0 \ 0 \ 1$

$1 \rightarrow 0 \ 0 \ 0 \ 1$

$\overline{0 \ 1 \ 0 \ 1} \rightarrow 2^3 + 2^1 = 10 \quad \times$

## ② offsetting

$$-8, 7 \xrightarrow{+8} 0, 15$$

We still have  $a + (-a) \neq 0$

$$\rightarrow \boxed{0} \quad 1 \quad 1 \quad 1$$

## ③ 2's Complement.

$$\begin{array}{r} 1 \quad 0 \quad 0 \quad 0 \\ - \quad 0 \quad 0 \quad 1 \\ \hline \boxed{1} \quad 0 \quad 0 \quad 1 \end{array}$$

Rule of thumb: To get a negative number from a positive number :

① Find a number which if added to the original number  $\rightarrow 111$

② Add 1 to the number  $\rightarrow 001$

## Lecture 4:

# EEE/SE 120 : 2's complement and its Arithmetic

- HW 1 is due Sep 3
- Lab 0 is due Sep 14
- Quiz 1 is on Sep 8
- Office hours T-Th 9:30<sup>AM</sup>
- Join the Lab zoom meeting if there is an issue w/ Labs.
- When you scribe you need to send the notes to me within 2 days

Defining negative numbers:

### 1 Sign bit

Neg  $\rightarrow$   $\boxed{1} \ 0 \ 0 \ 1 \rightarrow -1$

pos  $\rightarrow$   $\boxed{0} \ 0 \ 0 \ 1 \rightarrow +1$

problems: 1) "0" has two rep.  
2)  $1 + (-1) \neq 0$

### 2 Offsetting

problem:  $1 + (-1) \neq 0$

### 3 2's Complement

1. zero has one rep.

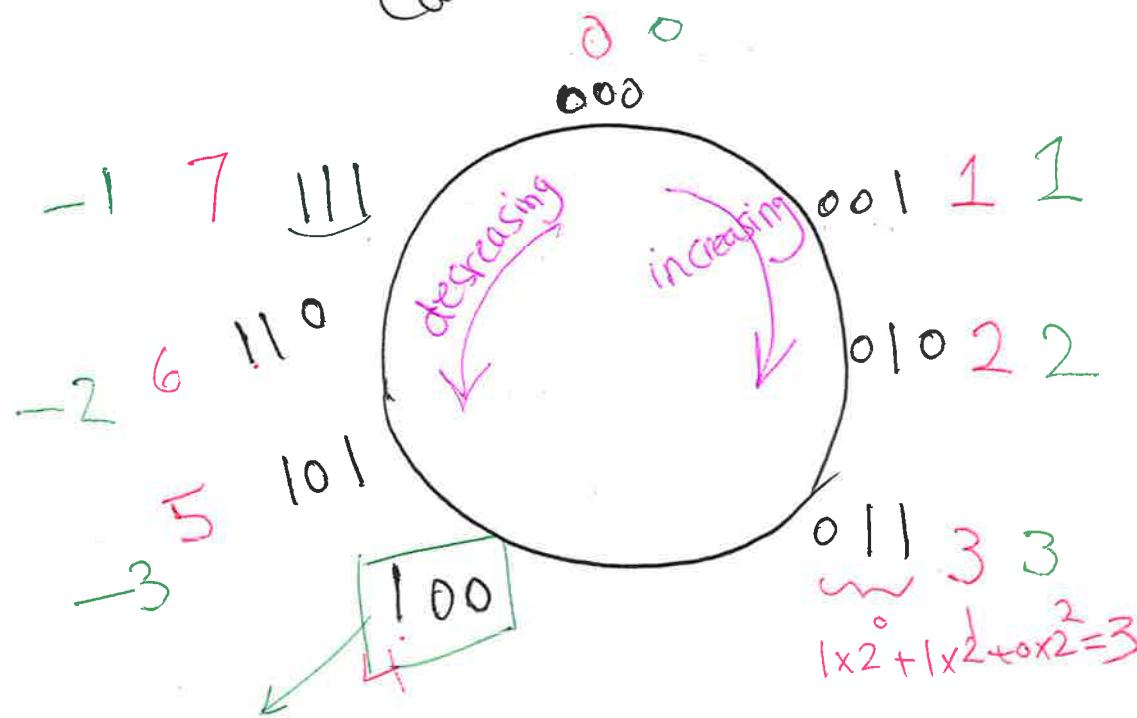
2.  $1 + (-1) = 0$

3. Signed bit is part of the number

4. positive numbers stay the same.

$$\begin{array}{r}
 (+3)_{10} \\
 + (-3)_{10} \\
 \hline
 (0)_{10}
 \end{array}
 \quad \longleftrightarrow \quad
 \begin{array}{c}
 (1\ 0\ 1\ 1)_2 \\
 \boxed{1} \ \boxed{0} \ \boxed{1} \\
 \hline
 \cancel{\boxed{1}} \quad \cancel{1} \quad \cdot \quad 0 \ 0 \ 0
 \end{array}
 \quad \rightarrow (-3)_{10}$$

ignore.  
Carry out



-4 → define this  
to be -4

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

How to get a negative number from a positive number?

Rule: ① Find the number which added to the original number that results in 111

② add 1 to that number

Example:

$$\begin{array}{r} & \boxed{1} & \boxed{1} \\ & + & \\ 0 & | & | \\ + & \boxed{1} & \boxed{0} & \boxed{1} \\ \hline \boxed{1} & 0 & 0 & 0 = 111 + 001 \\ & \underbrace{\quad\quad\quad}_{111} & & \\ & & 001 & \\ \hline & \boxed{1} & 0 & 0 \end{array}$$

Another way of writing the rule:

- 1 Flip all the bits  
of the original number +  $\begin{array}{r} 0 \ 1 \ 1 \\ - 1 \ 0 \ 0 \\ \hline 1 \ 1 \ 1 \end{array}$   
(one's complement)
- 2 Add 1 to it.

Example:  $(0 \ 1 \ 1) = 3_{10}$

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

- ① Flipping the bits : 1 0 0
- ② Add 1 to it

$$\begin{array}{r} 1 \ 0 \ 0 \\ + 0 \ 0 \ 1 \\ \hline 1 \ 0 \ 1 \rightarrow (-3)_{10} \end{array}$$

Example: 2's complement of a negative number!

$$\underline{(1 \ 0 \ 1)}_2 = (-3)_{10}$$

① flipping bits  $\rightarrow (0 \ 1 \ 0)_2$

② Add 1 to it

$$\begin{array}{r} 0 & 1 & 0 \\ + & & \\ \hline 0 & 0 & 1 \end{array}$$

$$\underline{\quad \quad \quad}$$

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

Example: 2's complement of 000?

① flipping the bits  $\rightarrow \underline{\quad \quad \quad}$

② To add 1 to it  $\rightarrow \underline{\quad \quad \quad}$

Example:

$$\begin{array}{r} \boxed{1} \ 0 \ 1 \\ \xrightarrow{\quad} 1 \ 0 \ 1 \ 0 \\ \textcircled{2} \ 0 \ 1 \ 0 \\ \qquad\quad 0 \ 0 \ 1 \\ \hline 0 \ 1 \ 1 \end{array} \rightarrow +3$$

(-3)<sub>10</sub> is converted to binary. The result is 1010. A circled 2 is above the second column from the left. The result is 011, which is +3.

Example:

$$\begin{array}{r} 1 \ 0 \ 0 \ 0 \\ \underbrace{\quad\quad\quad}_{(-8)} \\ (-8)_{10} \end{array} \quad (4 \text{ bit})$$

2's Compl.

$$\begin{array}{r} \textcircled{1} \ 0 \ 1 \ 1 \ 1 \\ \textcircled{2} \ 1 \ 1 \ 1 \ 1 \\ \hline 0 \ 0 \ 0 \ 1 \\ \hline 1 \ 0 \ 0 \ 0 \end{array} \rightarrow +8$$

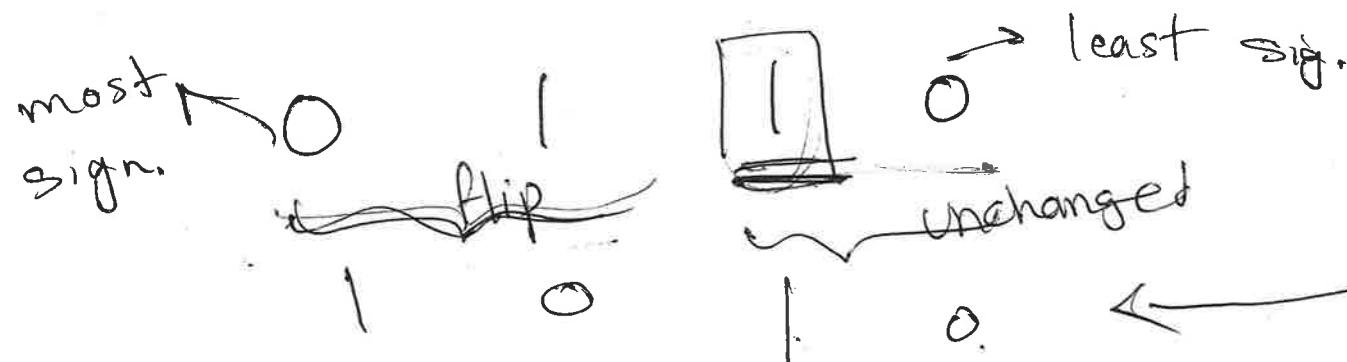
Example : 0 1 1 0

1 Flip all the bits : 1 0 0 1

2 To add 1 to it  $\begin{array}{r} + \\ \hline 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 1 & 0 \end{array}$

Quick way of Computing 2's Complement :

Spot the first "1" and leave everything the same up to that "1" and then flip the rest of the bits to the left



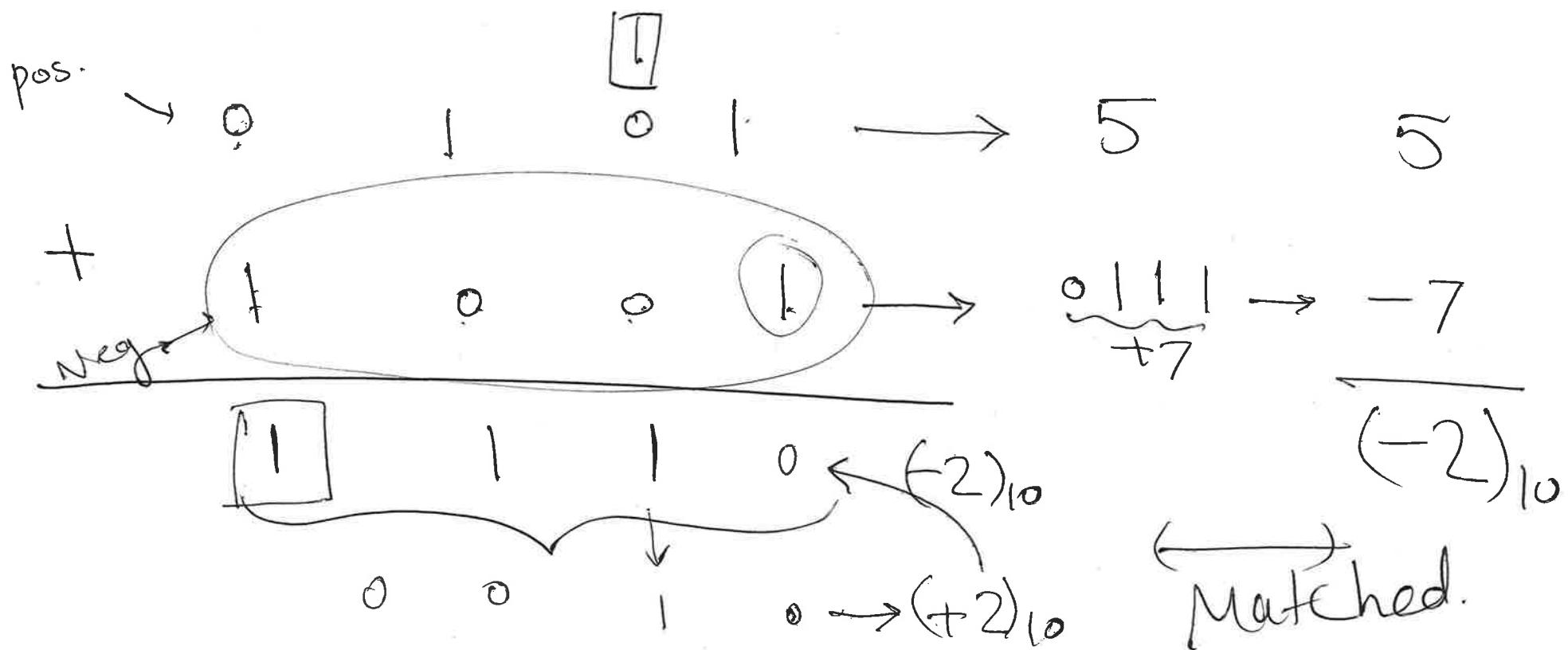
## Example

Signed binary

↳ Addition :

① numbers can be pos.  
or neg.

② Addition doesn't care  
about sign.



Example: Neg carry in  

$$\begin{array}{r}
 \text{Neg} \xrightarrow{\quad} \text{carry in} \\
 + \\
 \text{Neg} \xrightarrow{\quad} \\
 \hline
 \end{array}
 \begin{array}{cccc}
 0 & 1 & 1 & 1 \\
 | & | & | & | \\
 \hline
 \end{array}
 \begin{array}{c}
 2 \\
 \cancel{-9} \\
 \text{Don't match}
 \end{array}$$

pos Neg pos Neg  
 pos Neg Neg Post  
 we might have OF.  
 NO OF.

when we look at the most sign. bit  
 if carry in  $\neq$  carry out  $\Rightarrow$  OF.

Example : Add 2's complement of the following numbers and determine when we of

$$\begin{array}{r}
 \boxed{0} \\
 1 \ 0 \ 0 \ 0 \rightarrow -8 \\
 + 0 \ 1 \ 0 \ } \rightarrow 5 \\
 \hline
 \end{array}
 \quad
 \begin{array}{r}
 \boxed{1} \xrightarrow{\text{carryin}} \\
 0 \ 1 \ 0 \ 0 \rightarrow +4 \\
 + 0 \ 1 \ 1 \ 0 \rightarrow +6 \\
 \hline
 \end{array}
 \quad
 \begin{array}{r}
 1 \ 1 \ 1 \ 1 \\
 + 1 \ 1 \ 1 \ 0 \\
 \hline
 \end{array}$$

$$\begin{array}{r}
 \boxed{0} \quad 1 \ 0 \ 1 \ (\underline{-3})_{10} \\
 \text{Carry in} = \text{Carry out} \quad \boxed{0} \quad 1 \ 0 \ 1 \ 0 \ (\underline{10})_{10} \\
 \text{Carry out} \neq \text{Carry in} \\
 \downarrow \\
 0 \ 0 \ 1 \ 1 \rightarrow -3
 \end{array}$$

$$\begin{array}{r}
 \text{Carry out} \neq \text{Carry in} \\
 \downarrow \\
 \boxed{0} \ 1 \ 1 \ 0 \quad +6 \\
 \text{OF.}
 \end{array}$$

$$1010 = -6$$

$$\begin{array}{r} \text{Carry in} \downarrow \\ \boxed{1} \quad \boxed{0} \quad | \quad | \quad | \quad | \rightarrow (0001) = 1 \rightarrow (-1)_{10} \end{array}$$

$$\begin{array}{r} + \\ \boxed{1} \quad | \quad | \quad | \quad 0 \rightarrow (0010) = 2 \rightarrow (-2)_{10} \end{array}$$

$$\begin{array}{r} \hline \boxed{1} \quad | \quad | \quad 0 \quad | \\ \hline \end{array} \quad \begin{array}{l} \text{Match.} \\ \nwarrow (-3)_{10} \\ \hline \end{array} \quad \begin{array}{l} \text{Carryout} \\ \swarrow \\ \text{Carry in} = \text{Carry out} \end{array} \Rightarrow \text{No OF.}$$

?s  
comp.

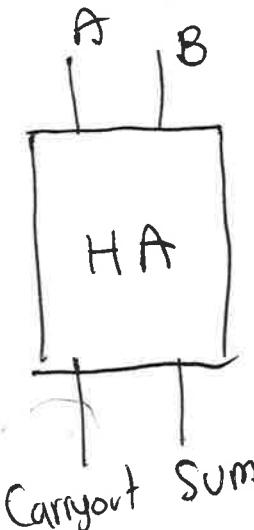
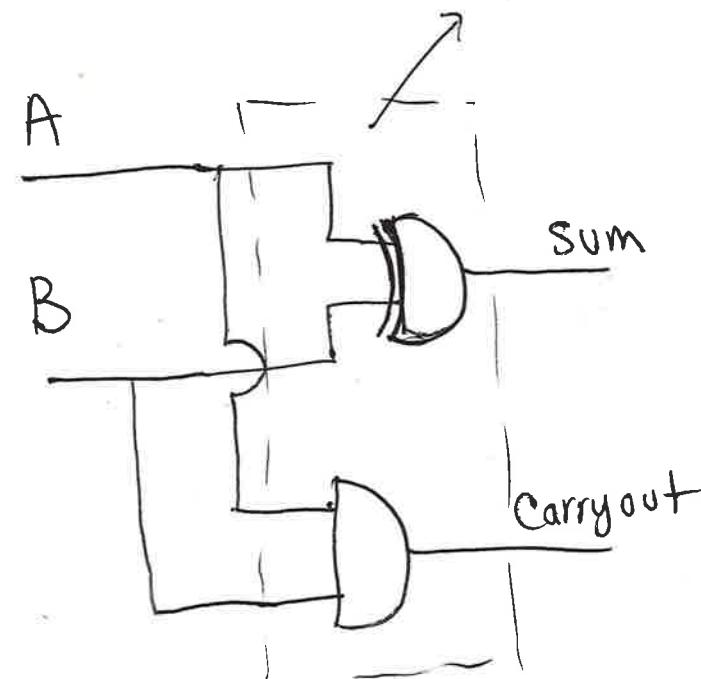
$$(0011) = 3$$

Lectures 5:  
 EEE / KSE 120 : Adders :

A	B	Sum	Carryout
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

input      XOR      AND      outputs

HALF ADDER



"Sum  $\Leftrightarrow$  XOR"

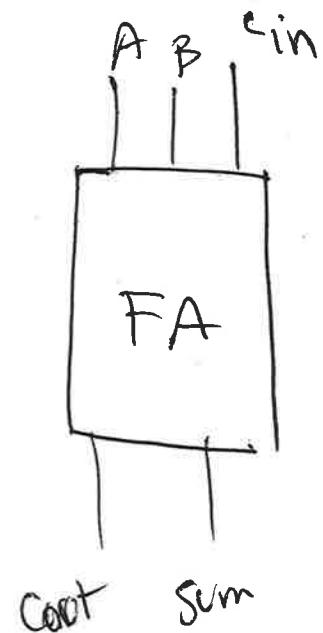
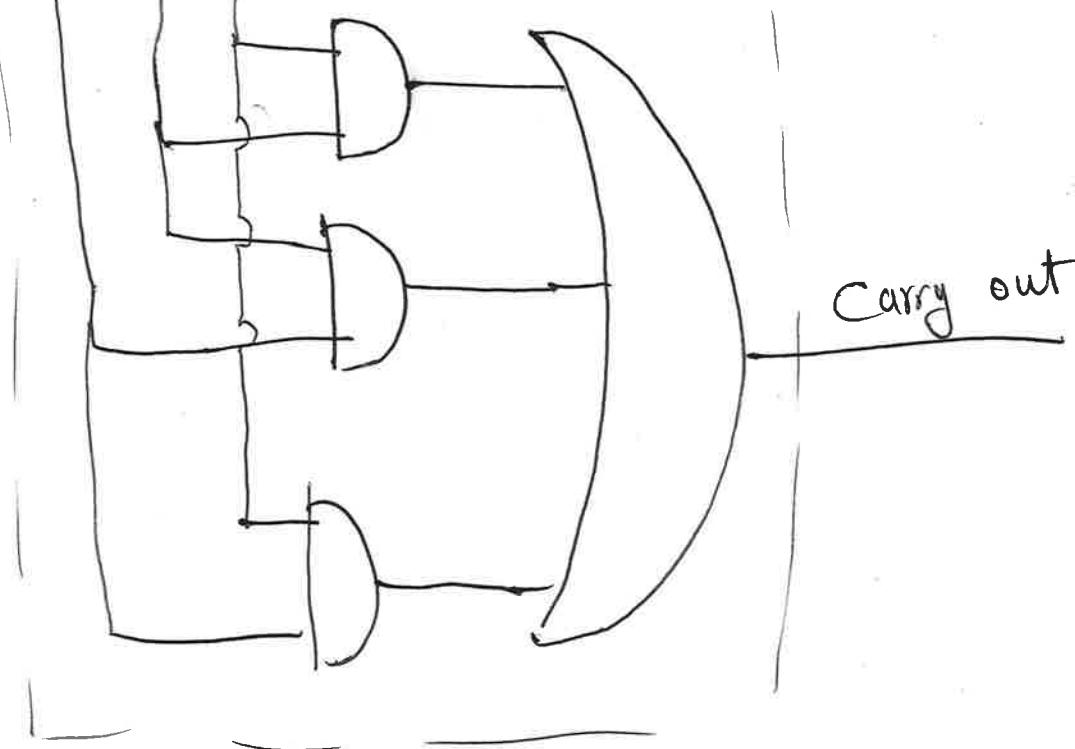
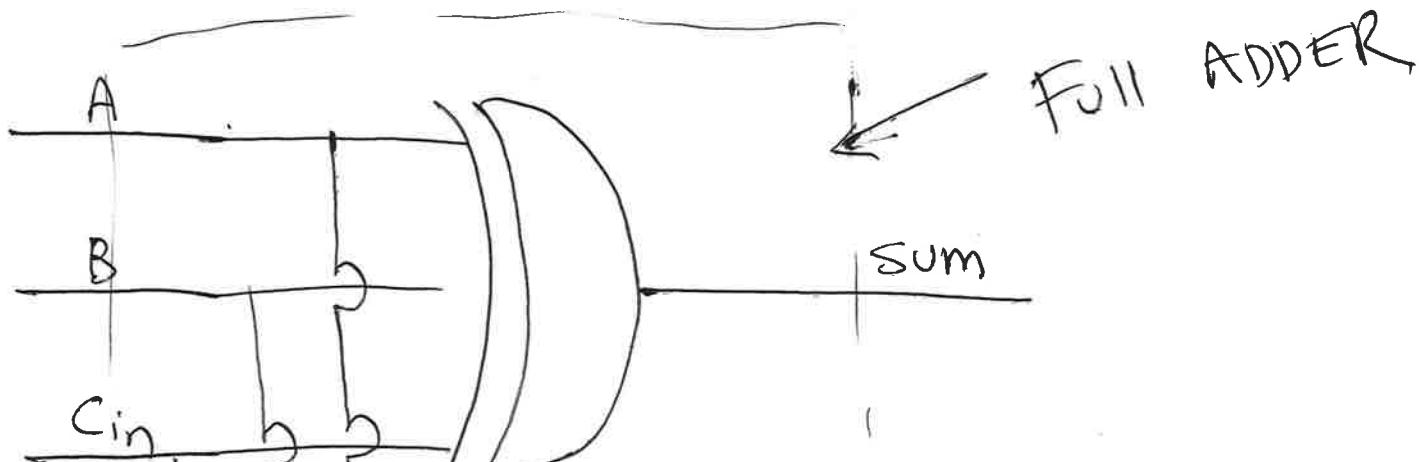
3 bit binary

A	B	Cin	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1 * $\bar{A} \bar{B} C_{in}$
1	0	0	1	0
1	0	1	0	1 * $\bar{A} \bar{B} C_{in}$
1	1	0	0	1 * $A \bar{B} \bar{C}_{in}$
1	1	1	1	1 * $A B C_{in}$

V.O.R

$$C_{out} = \underbrace{\bar{A} \bar{B} C_{in}} + \underbrace{A \bar{B} C_{in}} + \underbrace{A B \bar{C}_{in} + A B C_{in}} \\ (\bar{A} \bar{B} + A \bar{B}) C_{in} = A C_{in} + B C_{in}$$

$C_{in}$	$\bar{C}_{in}$	$C_{in} + \bar{C}_{in}$
0	1	1
1	0	1



$$(\bar{A}B + A\bar{B})C_{in} = AC_{in} + BC_{in}$$

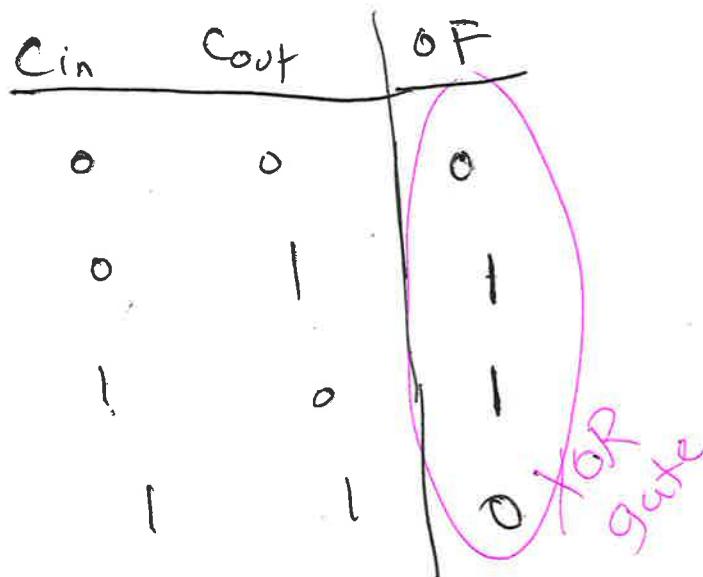
$$(\bar{A}B + \underbrace{AB + A\bar{B} + A\bar{B}}_{=0} )C_{in}$$

$$\cancel{(\bar{A} + A)B} + A(\cancel{B + \bar{B}})C_{in} = (A + B)C_{in} = AC_{in} + BC_{in}$$

$$C_{out} = AC_{in} + BC_{in} + AB$$

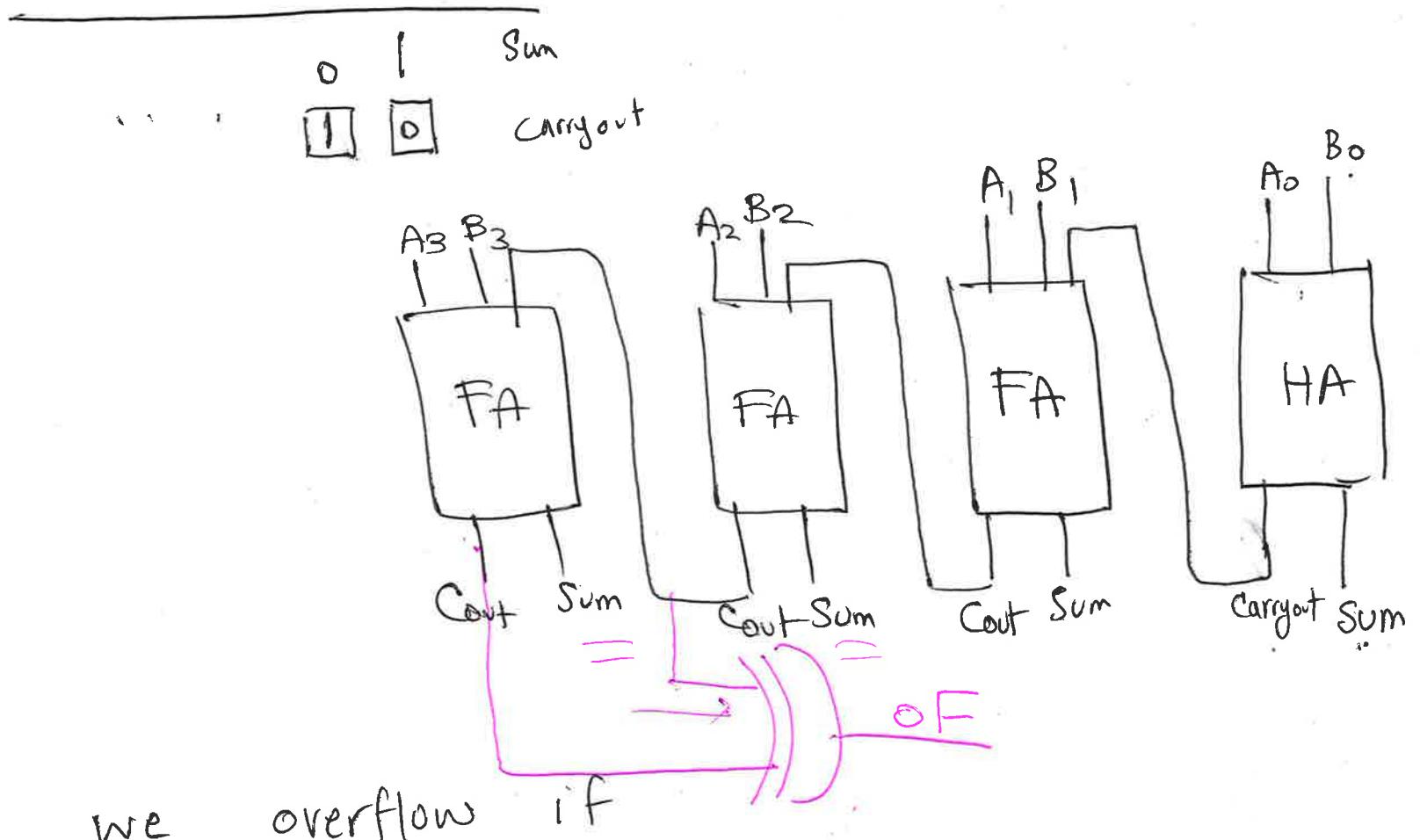
$$\begin{array}{r}
 \text{pos.} \quad 0 \quad 0 \\
 \text{Cin} \quad \downarrow \quad 0 \quad 0 \\
 \text{pos.} \quad 1 \quad 0 \\
 \hline
 \text{Sum} \quad 1 \quad 0
 \end{array}$$

$$\begin{array}{r}
 \text{Neg} \quad 1 \quad 1 \\
 \text{Neg} \quad 1 \quad 1 \\
 \text{Neg} \quad 1 \quad 1 \\
 \hline
 \text{Neg} \quad 1 \quad 0 \\
 \hline
 \text{Cin} \quad \downarrow \quad 1 \quad 0 \\
 \text{pos.} \quad 1 \quad 0 \\
 \hline
 \text{Sum} \quad 1 \quad 0 \\
 \text{Carryout} \quad 0 \quad 0 \\
 \text{OF} \quad \underline{\quad} \quad \checkmark \\
 \hline
 \end{array}$$



E      A  
0      1      F      0.

+      !      0.      !      b



Carry in  $\neq$  Carry out

$$+ \begin{pmatrix} 0 & 1 & 0 & 1 \end{pmatrix} = A = A_3 A_2 A_1 A_0$$

$$+ \begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix} = B = B_3 B_2 B_1 B_0$$


---

$A + B$   
 $A - B$

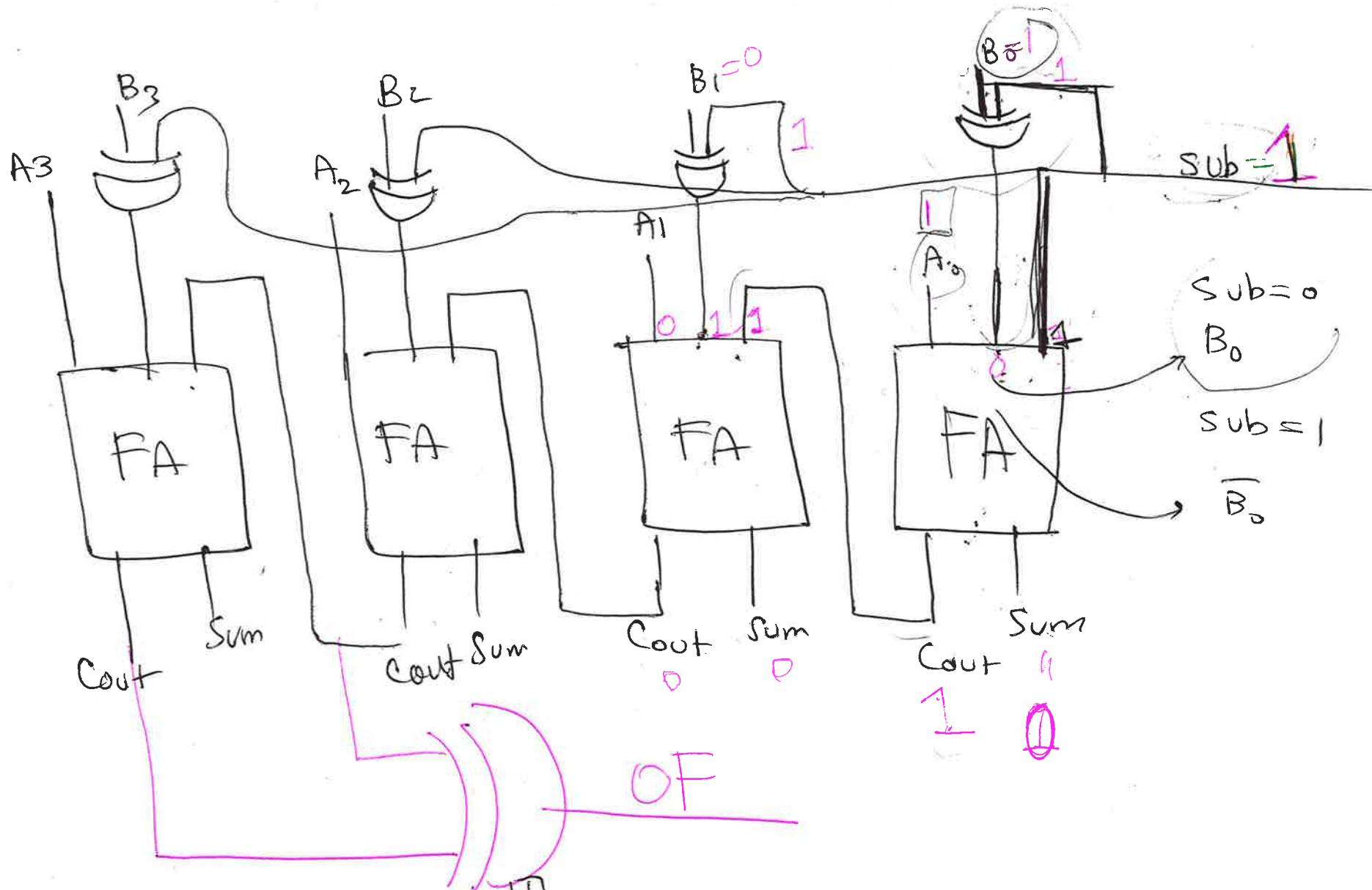
Reminder: 2's complement

① Flip all the bits

② to add 1 to it

$$\begin{array}{r}
 & & 0 & 1 & 1 & 0 \\
 + & & 0 & 0 & 0 & 1 \\
 \hline
 & & 0 & 1 & 1 & 1
 \end{array}$$

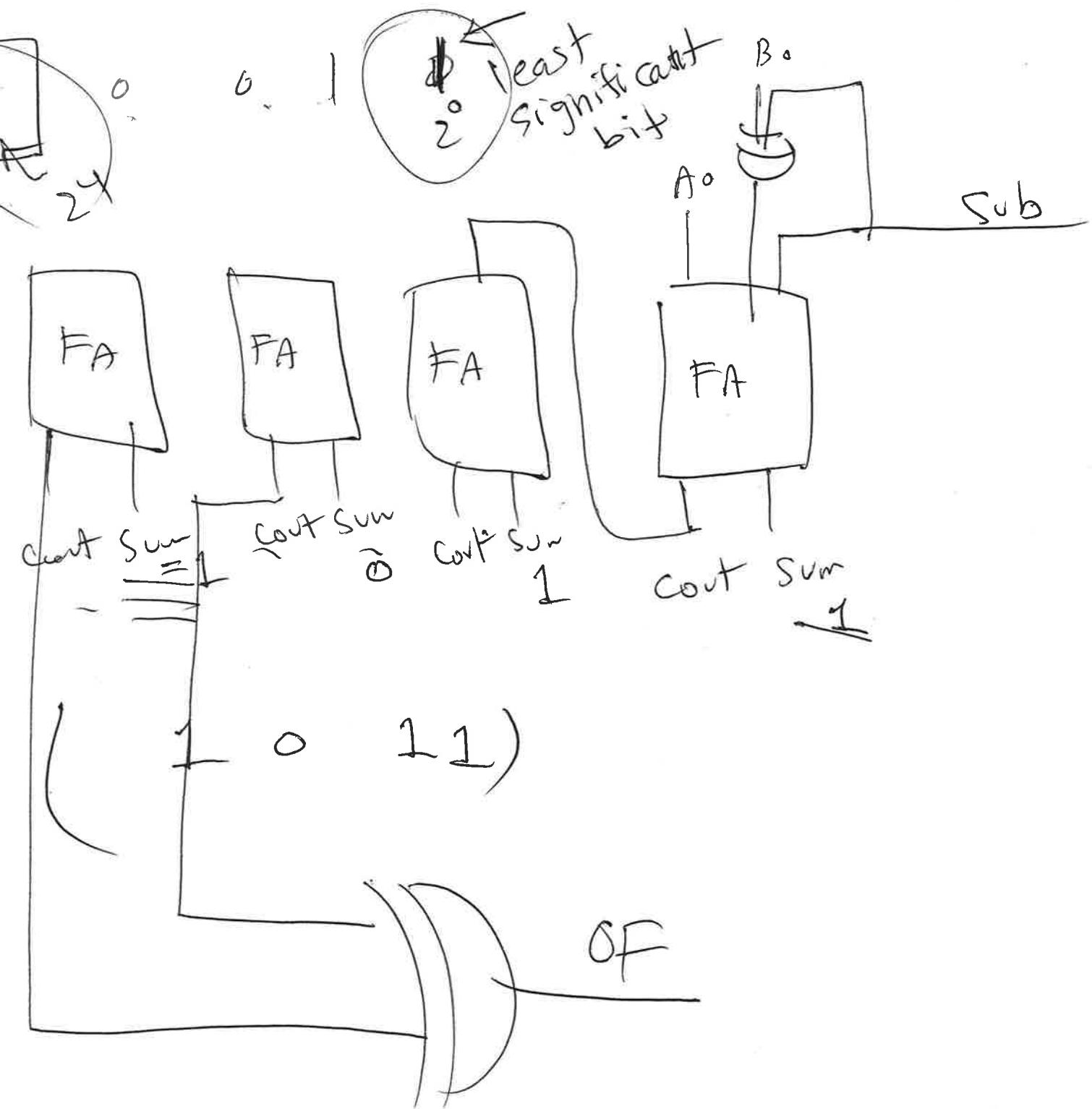
Subtractor = { Sub = 0 ( $\Leftrightarrow$  we're adding)  
Sub = 1 ( $\Leftrightarrow$  subtracting)



$$\begin{array}{r}
 A = 0 \ 1 \ 0 \ 1 \\
 + B = 1 \ 0 \ 0 \ 1 \\
 \hline
 \end{array} \rightarrow \text{Is : } 0 \ 1 \ 1 \ 0$$

Sum: 0 1 1 1

most  
sig.

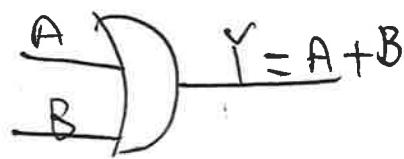
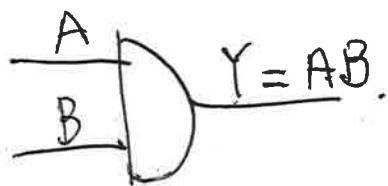


Lecture 6:

EEE/CSE 120 : Boolean Algebra

[EEE/CSE 120 - Quiz 1] ← title  
← Email

bahman.moraffah @ asu . edu



A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

$$1 \oplus 1 = 0$$

$$1 + 1 = 1$$

$$5 + 7 = 12 \text{ (closure)}$$

$$5 + 0 \leftarrow = 5$$

$$5 \cdot 1 = 5$$

$$\rightarrow 2 \cdot 3 \cdot 4 = 2 \cdot (3 \cdot 4) \neq (2 \cdot 3) \cdot 4$$

$$= (2 \cdot 4) \cdot 3$$

$$2 \cdot (3 + 4) = 2 \cdot 3 + 2 \cdot 4$$

"group"

### ① Closure :

$$\underbrace{1 + 1 = 1}_{\text{"OR" gate}}, \quad \underbrace{1 \oplus 1 = 0}_{\text{XOR gate}}$$

### ② Associativity

$$A \cdot B \cdot C = (A \cdot B) \cdot C = A \cdot (B \cdot C) = (C \cdot A) \cdot B$$

### ③ Neutral element

$$A \cdot B = A$$

?

A	B	$\#AB$
0	0	0
0	1	0
1	0	0
1	1	1

$B=1$

$$A + B = A$$

?

A	B	$\#A+B$
0	0	0
0	1	1
1	0	1
1	1	1

$$A \oplus B = A$$

?

$$B = 0$$



EX: prove that  $A \cdot (B+C) = A \cdot B + A \cdot C$

$$A \cdot (B+C) = (A+B) \cdot (A+C)$$

A	B	C	<u><math>A \cdot (B+C)</math></u>	$A \cdot B + A \cdot C$
0	0	0	0	0
1	0	0	1	0
1	1	0	1	1
1	0	1	1	1

## Summary :

$$A \cdot 0 = 0$$

$$\underbrace{A \cdot 1 = A}_{\text{AND gate}}$$

$$A \cdot A = A$$

$$\underbrace{A \cdot 0 = 0}_{\text{AND Gate}}$$

$$A + 1 = 1$$

$$\underbrace{A + A = A}_{\text{"OR" gate}}$$

$$A + 0 = A$$

$$1 + 1 = 1$$

Example: "OR" gate

line	A	B	$A+B$
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1

$$Y = \text{output} = (\overline{A}\overline{B}) + (\overline{A}\overline{B}) + (AB) = A+B$$

Sum of products

$$= \sum m(1, 2, 3)$$

min-term

Example:

$$Y = (\overline{A}\overline{B}) + (\overline{A}\overline{B}) + (AB) = A+B$$

Canonical form

Simplified form

## Sum of Products Rule : (SOP)

① Find the lines in truth table for which the output is "1".

② Write down the product of "ALL" input variables and invert those are "zero".

" ALL " input variables  
" zero "   
" Canonical Form "

③ Sum them all .

EECE 120 - Quiz 1 - resubmit

Example:

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

"SOP"

Canonical form

$$Y = \text{output} = \overline{A} \overline{B} + A \overline{B}$$
$$= A \oplus B$$

Simplified.

Lecture 7  
EEE/CSE 120: Boolean Algebra II

1 Closure  $(1+1=1, 1 \oplus 1=0)$

2 Associativity

$$A \cdot B \cdot C = (A \cdot B) \cdot C = A \cdot (B \cdot C) = (C \cdot A) \cdot B$$

3 Neutral element, (Identity)

$$AB = A$$

?

$$B = 1$$

$$A+B = A$$

$$B = 0$$

$$A \oplus B = A$$

$$B = 0$$

4 Inverse element

$$A \cdot B = 0$$

{}

$$A\bar{A} \neq 0$$

$$A+B = 1$$

$$B = \bar{A}$$

$$A \oplus B = 1$$

{}

## ⑤ Commutativity

$$A \cdot B = B \cdot A$$

$$A + B = B + A$$

## ⑥ Distributivity

$$\textcircled{1} \quad A \cdot (B + C) = A \cdot B + A \cdot C$$

$$\textcircled{2} \quad A + (B \cdot C) = (A + B) \cdot (A + C)$$

Example : "OR" gate

	A	B	$Y = A + B$
0	0	0	0
1	0	1	1 $\leftarrow \bar{A}B$
2	1	0	1 $\leftarrow A\bar{B}$
3	1	1	1 $\leftarrow AB$

$$\begin{aligned} 00 &\rightarrow 0 \\ 01 &\rightarrow 1 \\ 10 &\rightarrow 2 \\ 11 &\rightarrow 3 \end{aligned}$$

$$Y = (\bar{A}B + A\bar{B} + AB) = ?$$

Canonical form

$$\sum m(1, 2, 3)$$

min-term

$\downarrow 0$

$\underbrace{A + B}$   
Simplified form

$$Y = \bar{A}B + \underbrace{A\bar{B} + AB}_{\downarrow} = \bar{A}B + A \cdot (\cancel{\bar{B} + B})$$

(Distributivity)

$$= \bar{A}B + A \cdot 1$$

(Inverse)

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

(Identity, Neutral)

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

(Distributivity)

$$= 1 \cdot (B + A)$$

(Inverse)

$$= (B + A)$$

(Identity)

$$= \underbrace{A + B}$$

(Commutativity)

\*  $A \cdot (B + C) = A \cdot B + A \cdot C$

\*  $A + (B \cdot C) = (A + B) \cdot (A + C)$

Example :  $F(a, b, c) = \bar{a}bc + a\bar{b}c + ab\bar{c} + abc$

$$= \bar{a}bc + a\bar{b}c + ab(c + \cancel{\bar{c}})$$

$$= \bar{a}bc + \underbrace{a\bar{b}c + ab}$$

$$= \bar{a}bc + a \cdot (\bar{b}c + b)$$

$$= \bar{a}bc + a \cdot (\bar{b} + b) \cdot (c + b)$$

$$= \bar{a}bc + a \cdot (c + b)$$

$$= \underbrace{\bar{a}bc + a \cdot c}_{+ a \cdot b}$$

$$= (\bar{a}b + a) \cdot c + a \cdot b$$

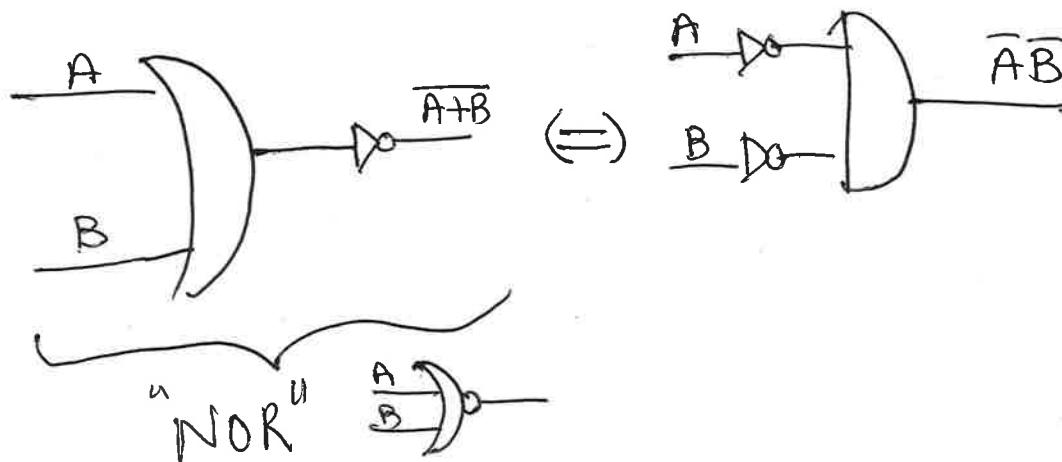
$$= (\cancel{\bar{a}} + a) (\cancel{b} + a) \cdot c + a \cdot b$$

$$= b \cdot c + a \cdot c + a \cdot b$$

## DeMorgan's Law:

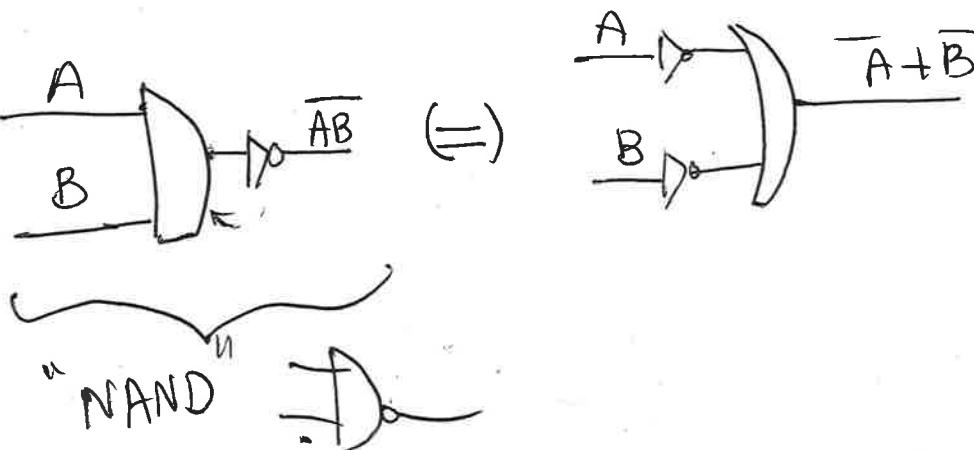
①

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

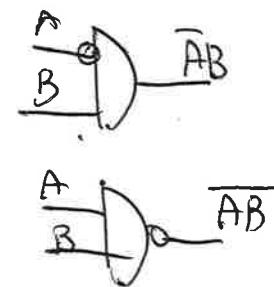


②

$$\overline{AB} = \overline{A} + \overline{B}$$



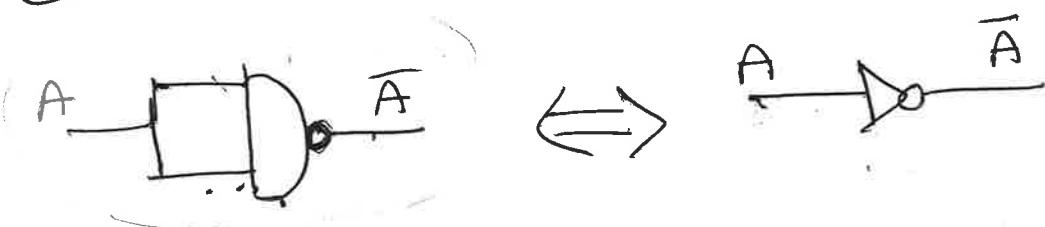
$$\overline{AB} \neq \overline{A}\overline{B}$$



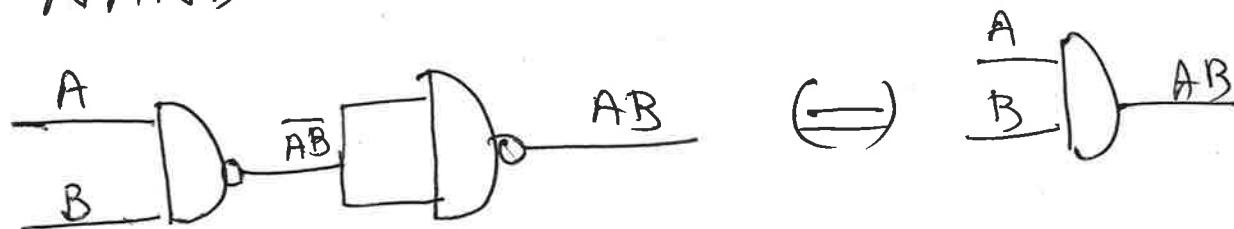
Q: Can we use "NAND" gates or "NOR" gates to build any circuit? YES!

"NAND":

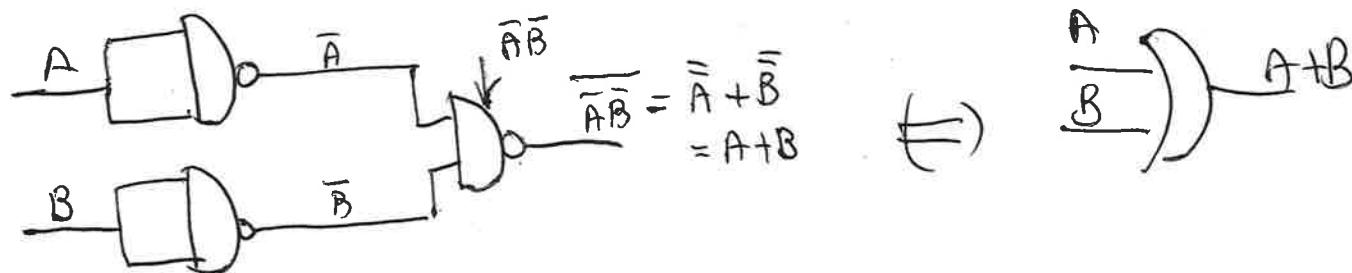
① NAND to build a NOT gate:



② NAND to build AND



③ NAND gate to build OR



Def: We call a gate "Functionally complete" if we can use that gate to build "AND", "OR", "NOT" gates.

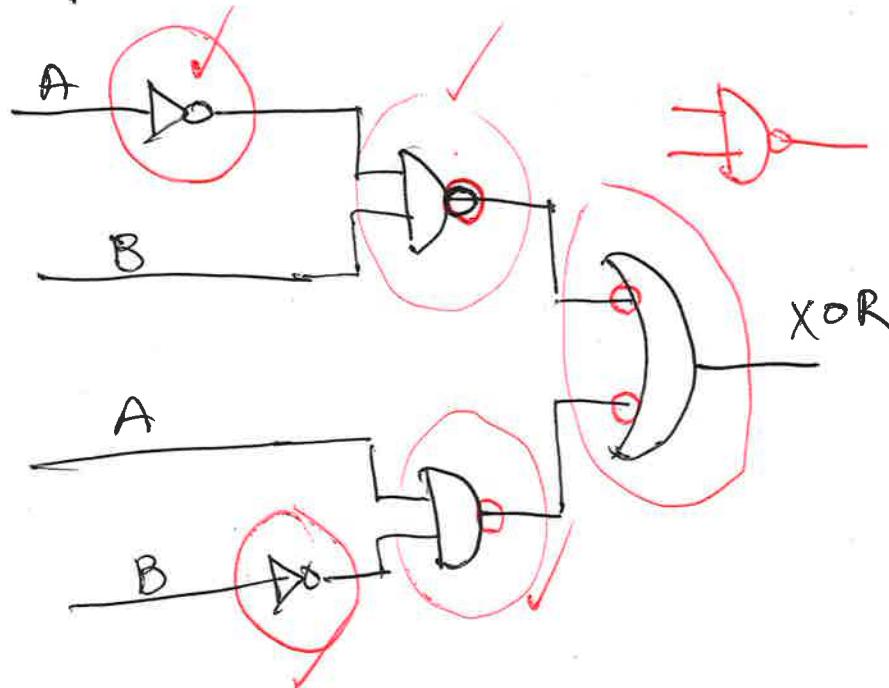
Example: NAND is functionally complete.

NOR is functionally complete.

Example : XOR gate  $\rightarrow$  build XOR gate in terms of NAND gates.

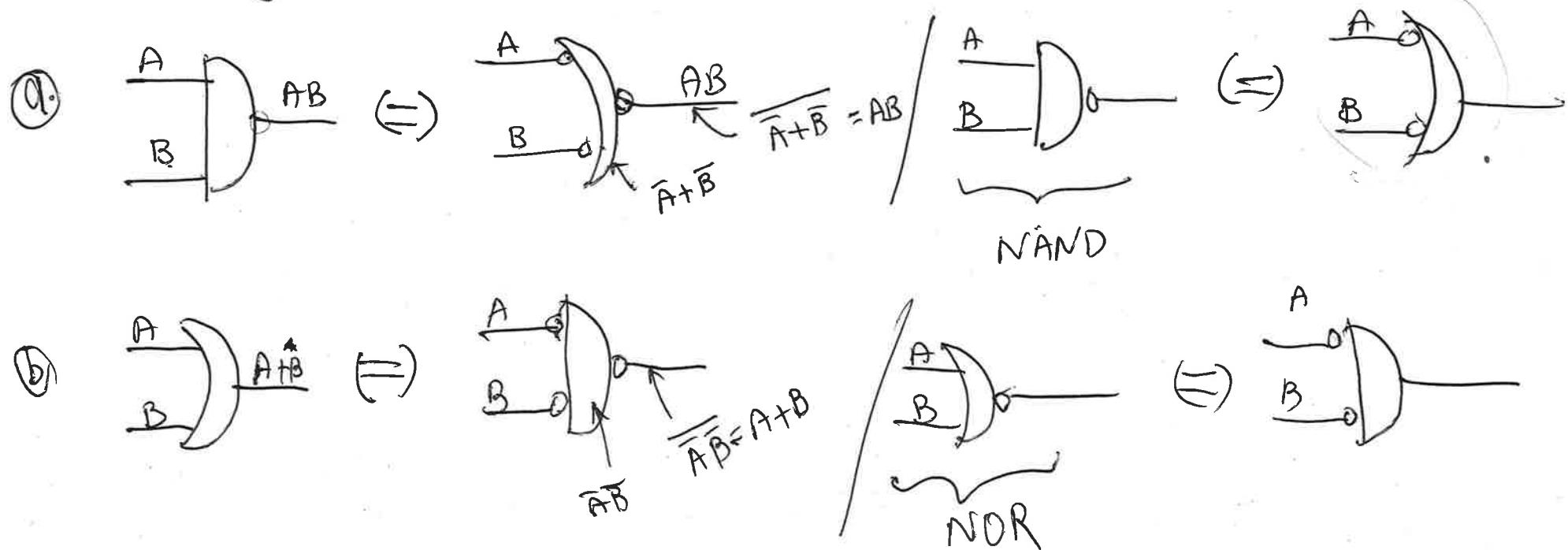
A	B	XOR
0	0	0
0	1	1 $\leftarrow \bar{A}B$
1	0	1 $\leftarrow A\bar{B}$
1	1	0

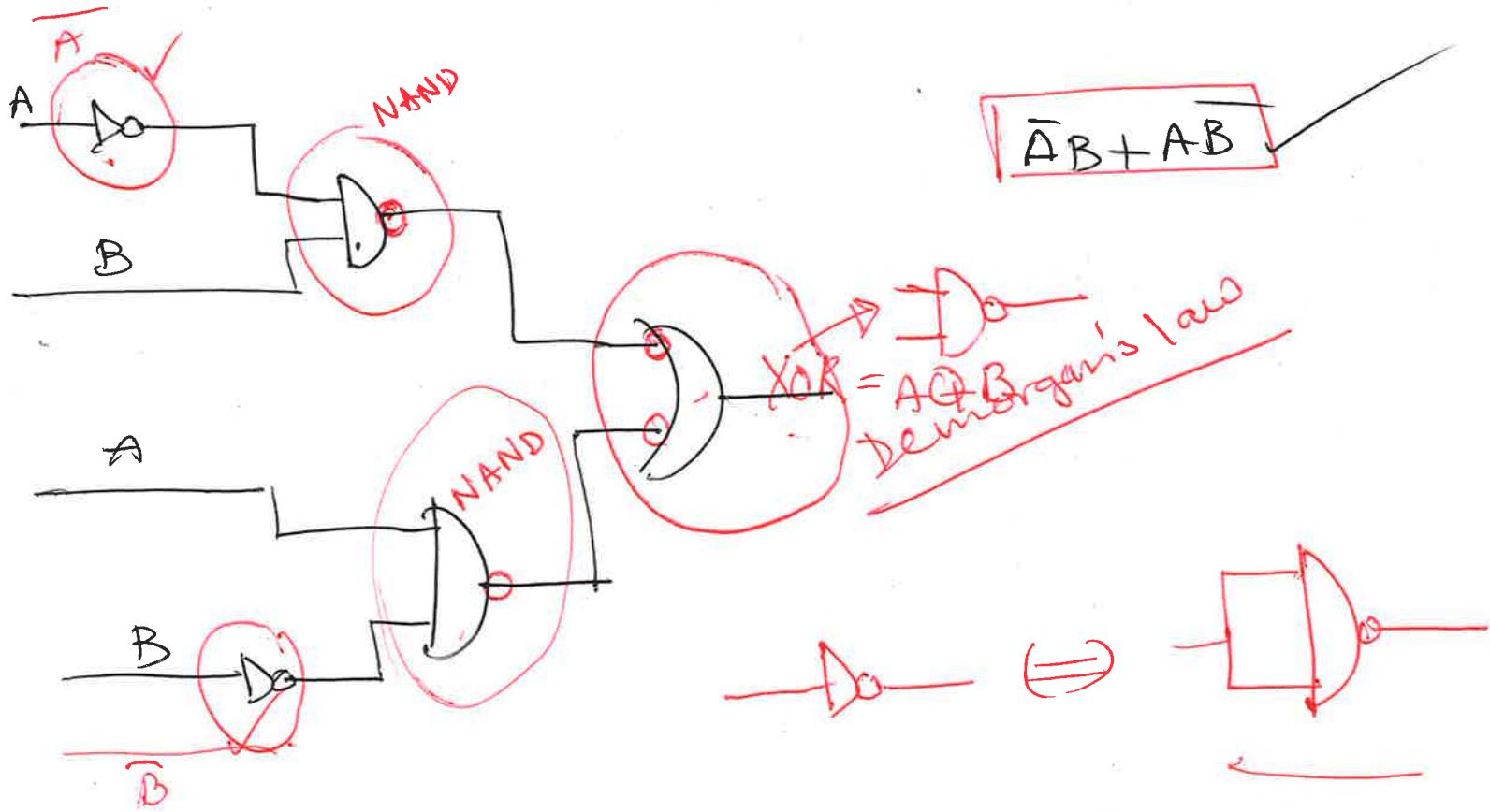
$$\text{XOR} = \bar{A}B + A\bar{B}$$



Steps to build a circuit in terms of "NAND" gates or "NOR" gates:

- ① Find the equation representing the gate or gates
- ② Draw the equation w/o paying attention to the NAND or NOR.
- ③ Play the bubble game :





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

$\bar{A}$

$\bar{B}$

$A$

$B$

## Lecture 8:

EEE/CSE 120:

"Sum of Product" & Product of sum

- ① HW2 is due today → HW3 is up tonight
- ② Lab 0 is due today
- ③ office hours T/TH 9:30 - 10:15 AM
- ④ Short Quiz 1 is on Thursday

Example: NAND gates , NOR gates functionally complete

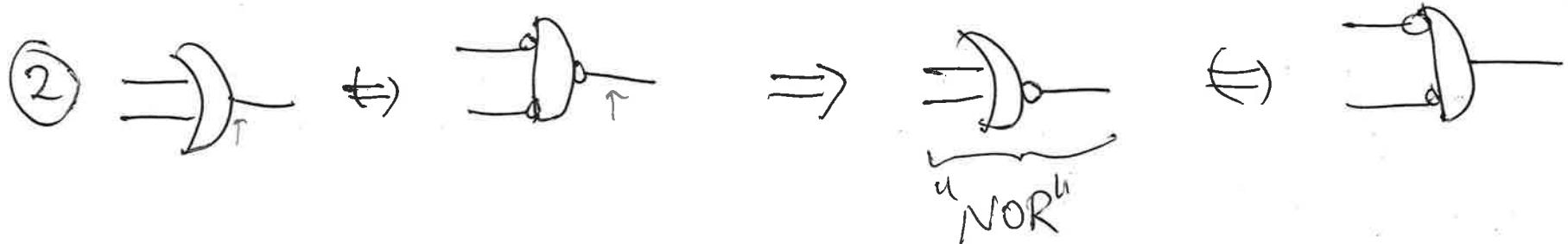
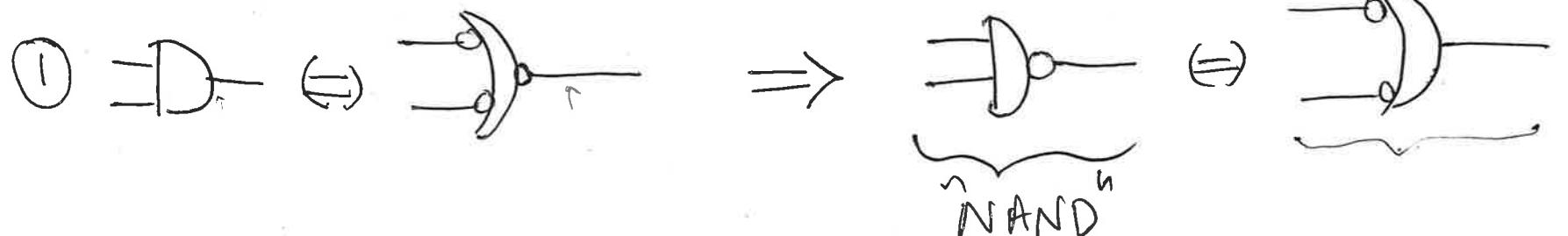
How to build any circuit using NAND /NOR gates!

1 write down the function using truth table and draw it "normally"

2 play the bubble game

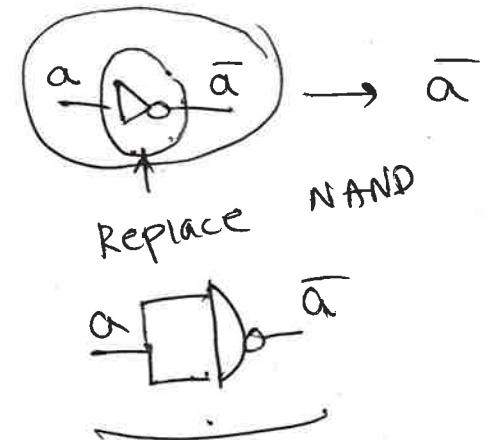
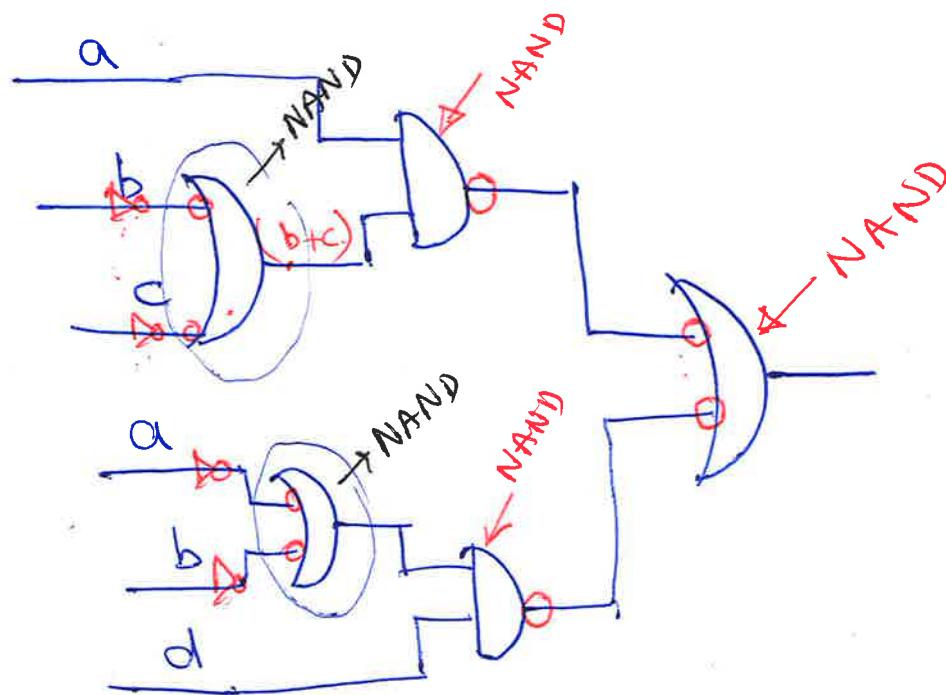
$$a \odot \bar{a} \Leftrightarrow a \triangleright \bar{a}$$

Rule:

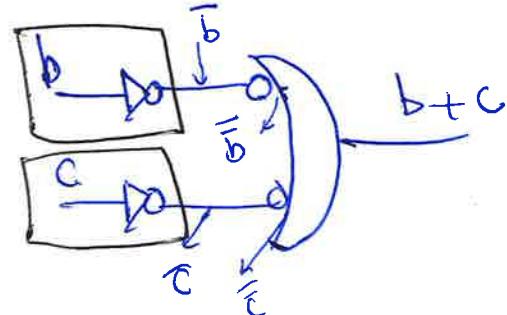


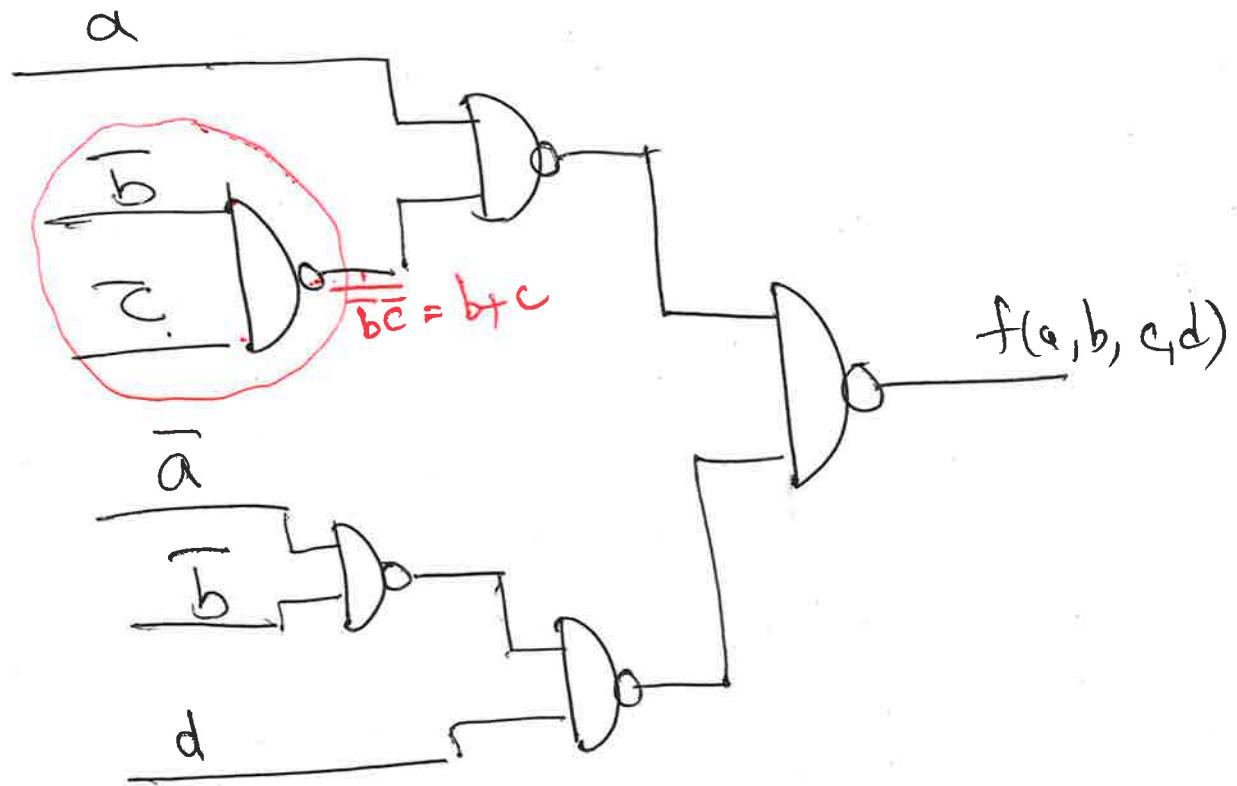
Example : Assume we have access to all inputs as well as their complement, draw the following circuit using only "NAND" gates!

$$f(a, b, c, d) = \underline{a(b+c)} + \underline{d(a+b)}$$

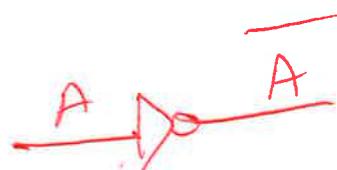
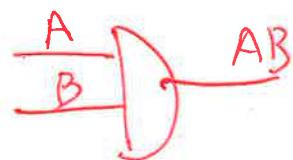
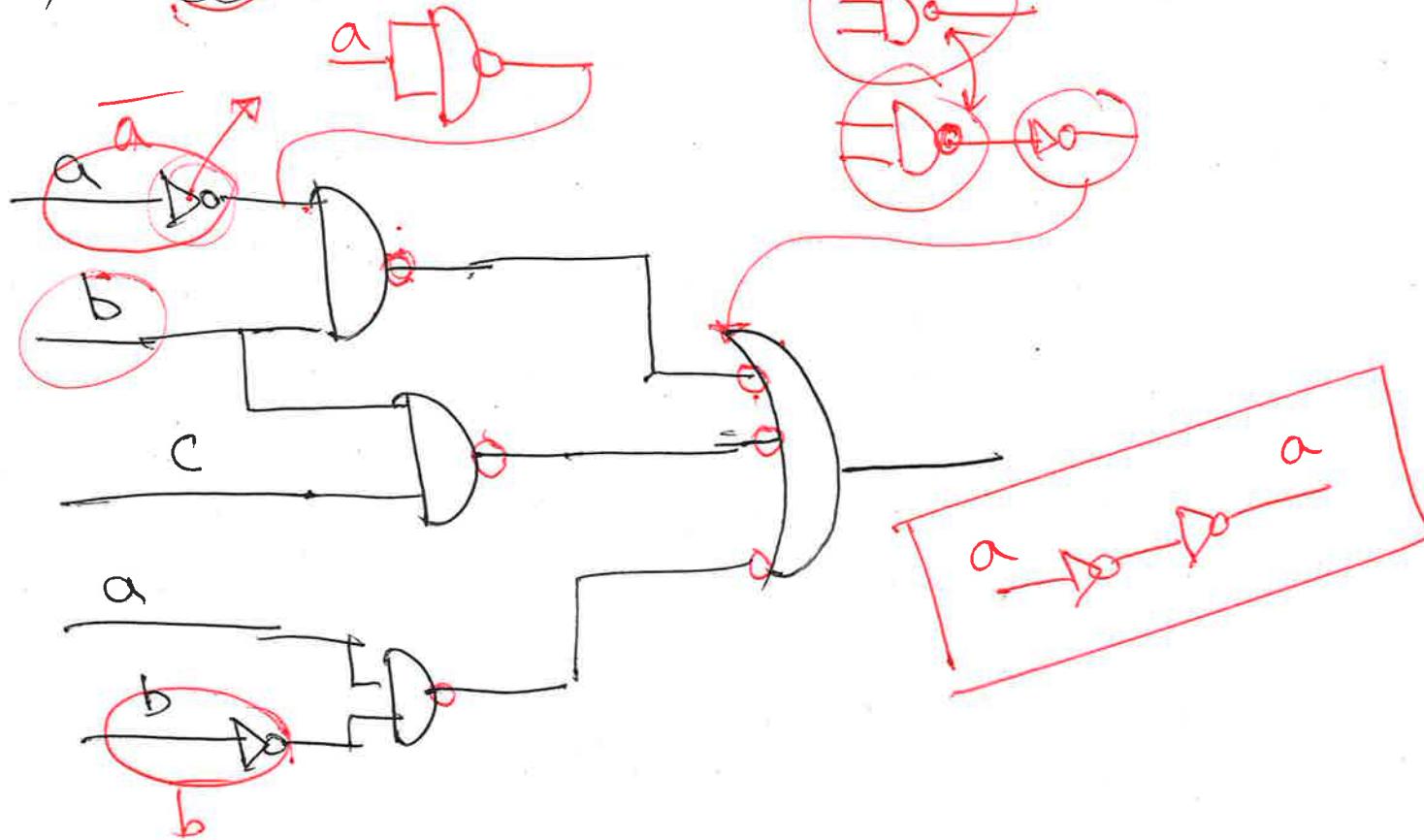


$$\overline{b} \quad \overline{c} \quad b+c$$

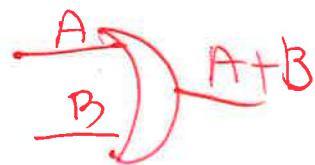




$$f(a, b, c) = \overline{ab} + bc + \overline{ab}$$



$$\overline{AB} = \overline{A} + \overline{B}$$



How to do product of sum:

- ① look at the lines of the truth table for which function is "0".
- ② write the sum of the input variables and invert those which are "one".
- ③ multiply all the sums.

$$f(a,b,c) = \underbrace{a\bar{b}c}_{\text{sum}} + \underbrace{ab\bar{c}}_{\text{sum}} + \underbrace{\bar{a}bc}_{\text{sum}} \quad (\text{canonical})$$

$$f(a,b,c) = a\bar{b}c + \underbrace{bc}_{\text{sum}}$$

Example:

line #	A	B	C	Y
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

OUTPUT

pos?

$$(A + B + \bar{C})$$

$$(\bar{A} + B + C)$$

$$Y = (A + B + \bar{C})(\bar{A} + B + C)$$

$$Y = \prod M(1, 4)$$

max-term

Max-term repr.

$$\sum_{i=1}^n a_i = a_1 + a_2 + \dots + a_n$$

$$\prod_{i=1}^n a_i = a_1 \cdot a_2 \cdot \dots \cdot a_n$$

$$1+2+\dots+n = \sum_{i=1}^n i$$

$$n! = 1 \times 2 \times \dots \times n = \prod_{i=1}^n i$$

Example :  $F(a,b,c) = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{c} + \bar{b}\bar{c}$

- a) write the Canonical SOP.
- b) write the Canonical POS.
- c) write the function as min-term expression.
- d) write the function as max-term expression.

$$(1+1)^{+1} = 1$$

line#	a	b	c	$Y = F(a,b,c)$
0	0	0	0	1 $\leftarrow \bar{a}\bar{b}\bar{c}$
1	0	0	1	1 $\leftarrow \bar{a}\bar{b}c$
2	0	1	0	0 $\leftarrow (\bar{a}+\bar{b}+c)$
3	0	1	1	1 $\leftarrow \bar{a}bc$
4	1	0	0	0 $\leftarrow (\bar{a}+b+c)$
5	1	0	1	1 $\leftarrow a\bar{b}c$
6	1	1	0	0 $\leftarrow (\bar{a}+\bar{b}+c)$
7	1	1	1	0 $\leftarrow (\bar{a}+b+\bar{c})$

$$F(a,b,c) = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}c + \bar{a}b\bar{c}$$

$$\begin{aligned} a=0, b=0, c=0 \rightarrow F &= \underbrace{1 \times 1 \times 1}_1 + \underbrace{1 \times 0 \times 1}_0 + \underbrace{1 \times 0 \times 0}_0 \\ &= 1 \end{aligned}$$

$$\begin{aligned} a=0, b=1, c=1 \rightarrow F &= \underbrace{1 \times 1 \times 1}_1 + \underbrace{1 \times 1 \times 0}_1 + \underbrace{0 \times 1 \times 0}_0 \\ &= 1 \end{aligned}$$

$$\underbrace{1 \times 0 \times 0}_0 + \underbrace{1 \times 1}_1 + \underbrace{0 \times 1}_0 = 1$$

$$\begin{aligned} a=1, b=0, c=1 \rightarrow F &= \underbrace{0 \times 0 \times 0}_0 + \underbrace{0 \times 1}_0 + \underbrace{1 \times 1}_1 = 1 \end{aligned}$$

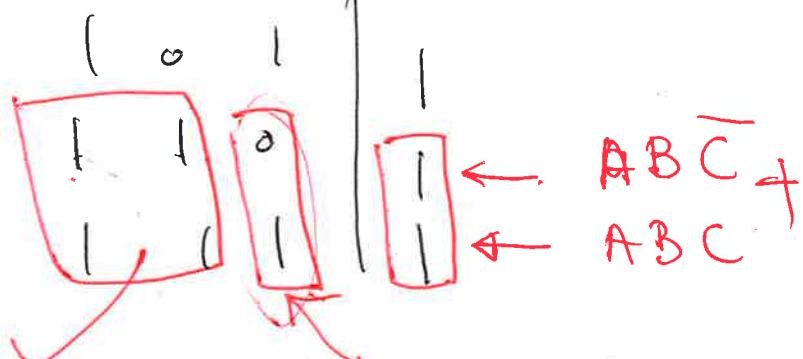
a)  $Y = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}c + \bar{a}bc + abc$  (SOP)

b)  $Y = (a+b+c)(\bar{a}+b+c)(\bar{a}+\bar{b}+c)(\bar{a}+\bar{b}+\bar{c})$  (POS)

c)  $F(a, b, c) = \sum m(0, 1, 3, 5)$

d)  $F(a, b, c) = \prod M(2, 4, 6, 7)$

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1



$A \oplus B$  are  
not changing variable

Demorgan's  
law

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

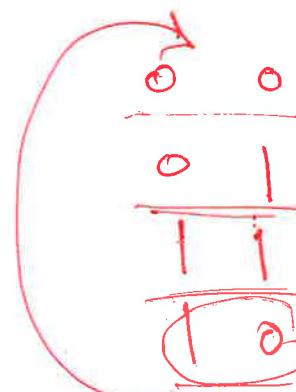
$$\overline{AB} = \overline{A} + \overline{B}$$

$$\overline{AB} = \overline{A} \cdot \overline{B}$$

$$\overline{AB}$$

$$AB [C + \overline{C}] = \underline{\underline{AB}}$$

$$\begin{array}{cc} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{array}$$



gray code!

$$\begin{aligned}
 y &= f(x) \\
 &= x^2 + 2 \\
 &= 5^2 + 2 \\
 &= 27
 \end{aligned}$$

## Short Quiz 1 :

Assume both inputs and their complement are given.

Draw the following circuit using only "NAND" gates

$$F(a,b,c,d,e,f) = a(bc + de) + f(cd + be)$$

## Lecture 9 :

EEE | CSE 120 : Karnough Map (k-map)

- ① HW3 is up on Canvas (due: Oct 1)
- ② Lab 1 is up on Canvas (due: Sep 28)
- ③ office hours T/TH 9:30 - 10:15 AM.
- ④ Start installing Lockdown browser for exams.

Example: line

	A	B	C	Y
0	0	0	0	1 $\leftarrow \bar{A}\bar{B}\bar{C}$
1	0	0	1	0
2	0	1	0	1 $\leftarrow \bar{A}B\bar{C}$
3	0	1	1	1 $\leftarrow \bar{A}BC$
4	1	0	0	0
5	1	0	1	1 $\leftarrow A\bar{B}C$
6	1	1	0	1 $\leftarrow ABC$
7	1	1	1	1 $\leftarrow A\bar{B}\bar{C}$

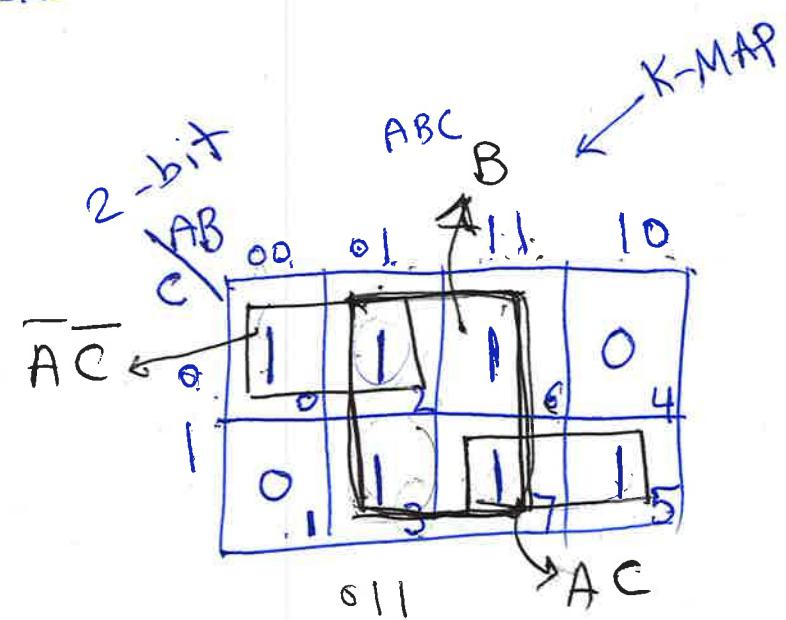
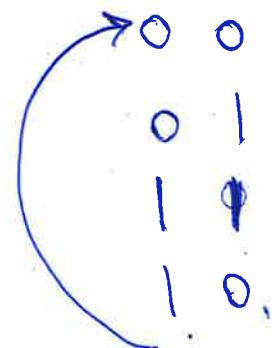
A and B are not changing  
AB

C is changing variable

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

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

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

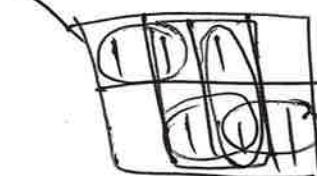


## Rules of minimum sum of product:

1 Find all the ones on the map and form the K-map.

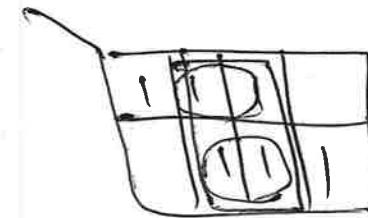
2 group all the ones into groups of  $2^n$  elements  
(vertically / Horizontally but NOT diagonally) ↴

groups are called implicants!



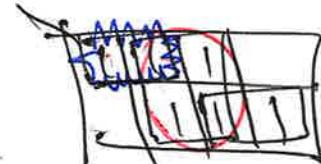
3 Find the largest groups possible

↳ prime implicants.

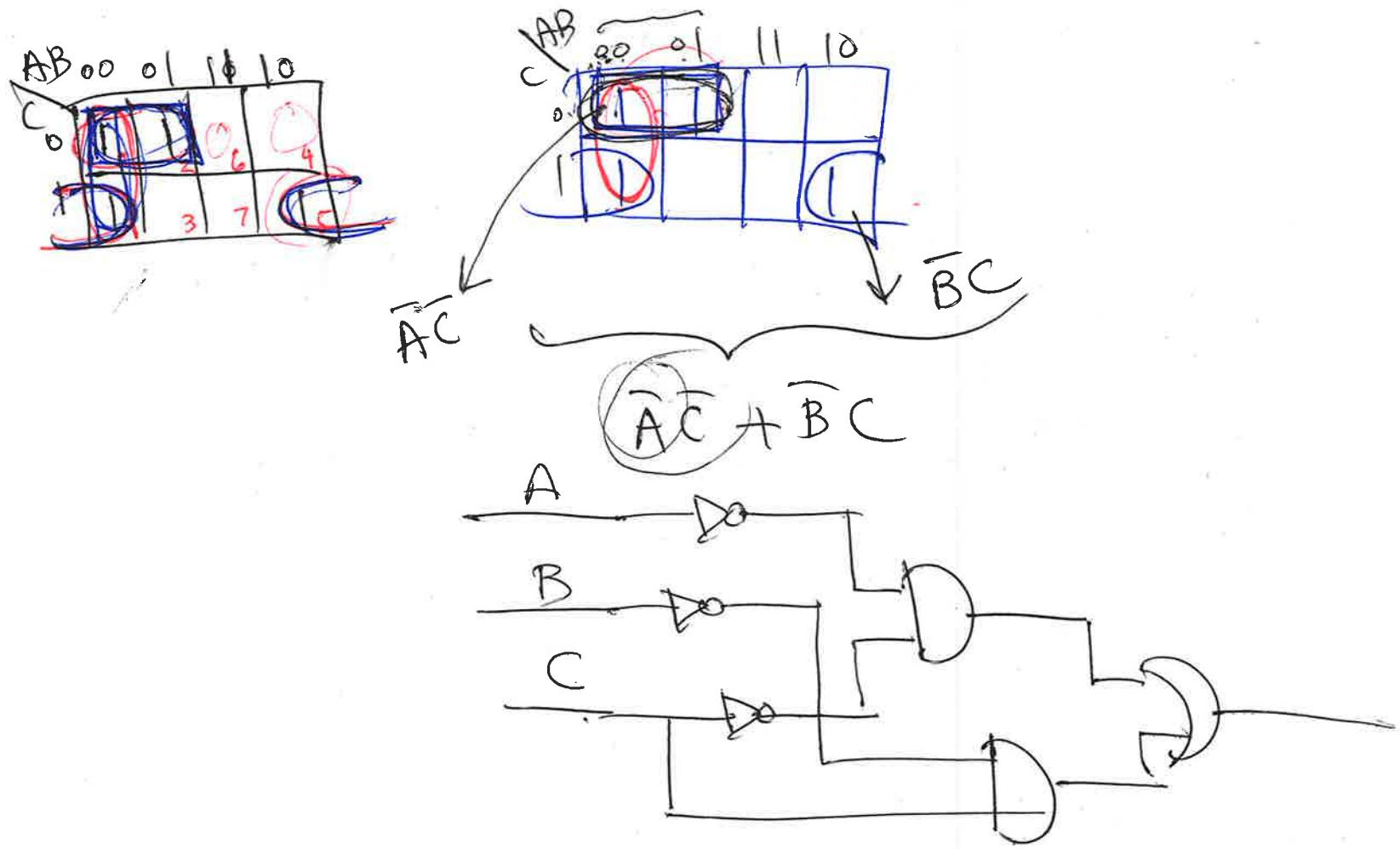


4 Find groups that have at least a single "one" that is NOT shared w/ another group.

↳ essential prime implicants



5 Add the product terms for the products of the eliminated changing variable. (essential prime implicants)



## Example: Full Adder

#	A	B	ein	Cout
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1 $\leftarrow \bar{A}BC_{in}$
4	1	0	0	0
5	1	0	1	1 $\leftarrow A\bar{B}C_{in}$
6	1	1	0	1 $\leftarrow AB\bar{C}_{in}$
7	1	1	1	1 $\leftarrow ABC_{in}$

$$Y = \bar{A}BC_{in} + A\bar{B}C_{in} + AB\bar{C}_{in} + ABC_{in}$$

Q: Simplify Y

- ① Boolean Alg.
- ② K-MAP.

3 inputs  $\Rightarrow$  3-variable K-MAP.

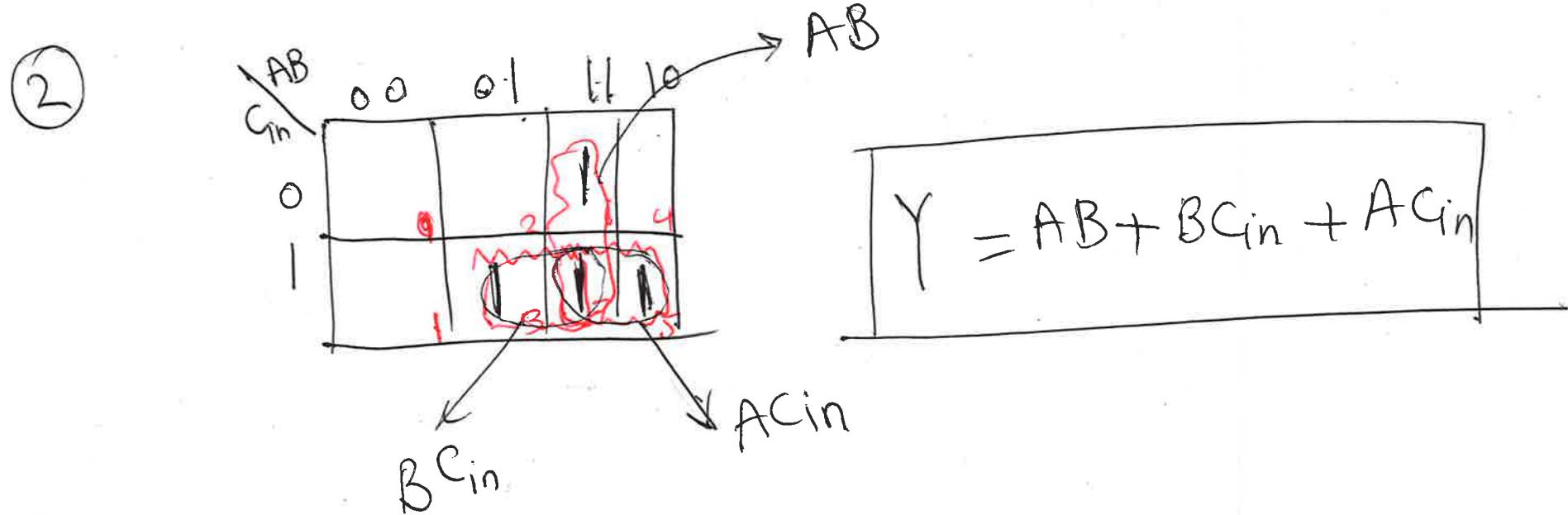
$$\sum m(3, 5, 6, 7)$$

$$\textcircled{1} \quad Y = \underline{\underline{\bar{A}B C_{in}}} + \bar{A}\bar{B}C_{in} + A\bar{B}\bar{C}_{in} + \underline{\underline{ABC_{in}}} + \bar{A}\underline{\underline{BC_{in}}} + \underline{\underline{ABC_{in}}}$$

$\cancel{A + A = A}$

$$BC_{in} [\bar{A} + A] + AC_{in} [\bar{B} + B] + AB [\bar{C}_{in} + C_{in}]$$

$$= BC_{in} + AC_{in} + AB$$



Sep 22, 2020

## Lecture 10 : Karnough Map (K-MAP) cont.

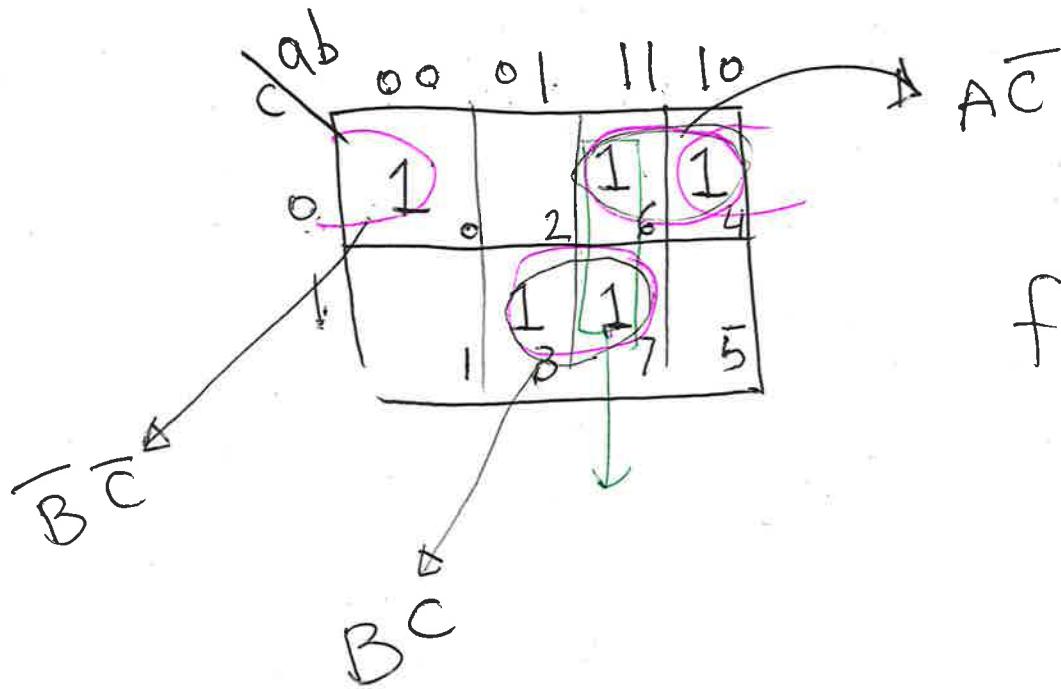
- ① Lab office hours |

Monday	5:00-6:00 PM	(AZ time)
Wednesday	12-1:00 PM	
Friday	2:30-4:00 PM	
- ② Office hours 9:30 - 10:15 Am (T/TH)
- ③ Have Quartus installed on your computer.

Example : minimum sum of product of f

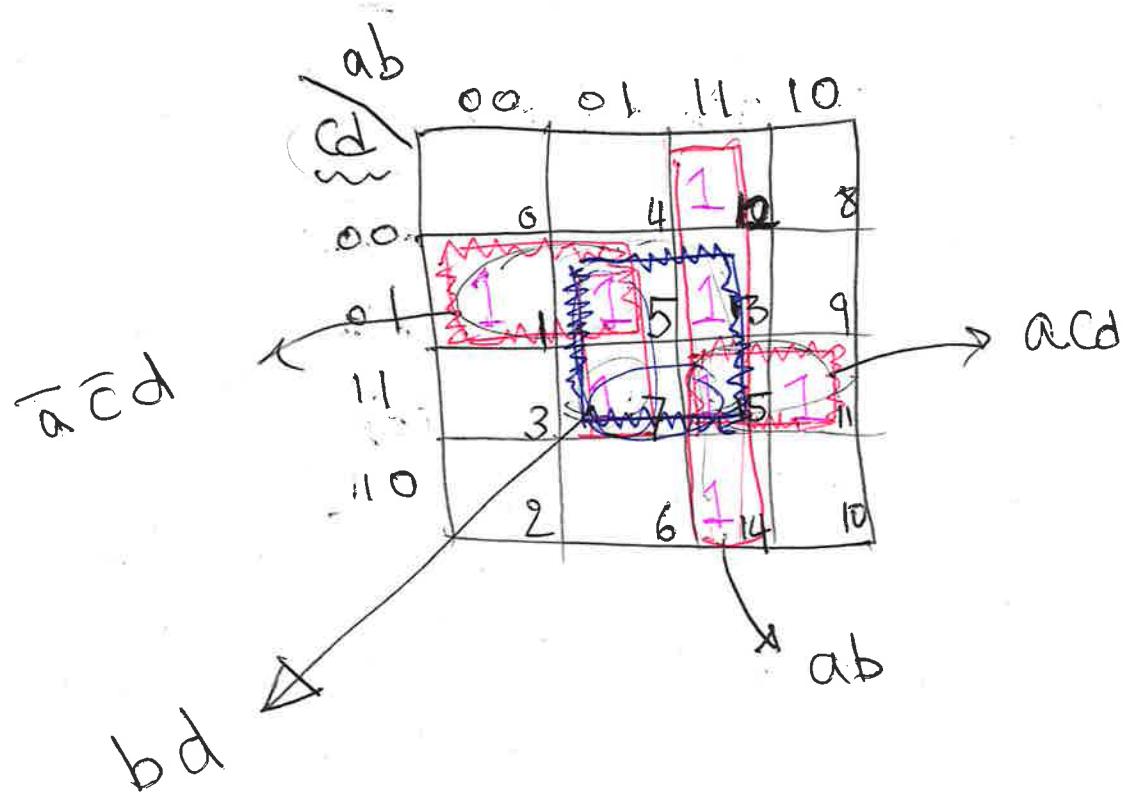
$$f(a,b,c) = \sum m(0, 3, 4, 6, 7)$$

K-MAP  $\rightarrow$  3-var



$$f(a,b,c) = \overline{B}\overline{C} + Bc + A\overline{C}$$

Example :  $f(a,b,c,d) = \sum m (1, 5, 7, 11, 12, 13, 14, 15)$



4-var K-MAP

$$11\ 00 \rightarrow 12$$

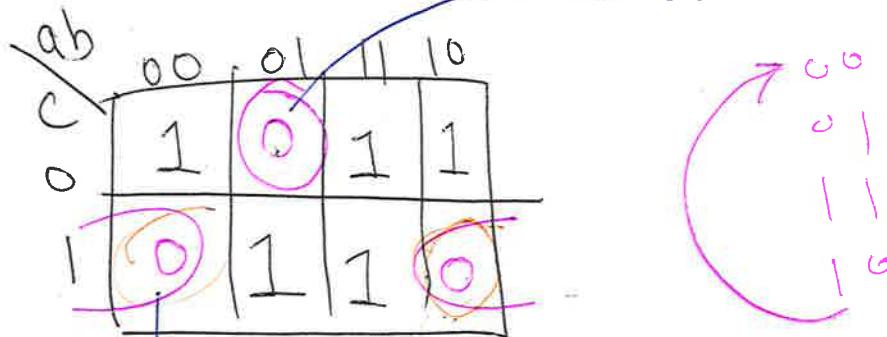
minimum sum of product

$$f(a,b,c,d) = \underline{acd} + \underline{\bar{a}\bar{c}d} + \underline{bd} + \underline{ab}$$

$$\overline{\overline{f}} = f$$

~~b(a+d)~~  
sum

Example:  $f(a,b,c) = \sum m(0, 3, 4, 6, 7)$



$$\bar{b}\bar{c} \quad \bar{f}(a,b,c) = \bar{a}b\bar{c} + \bar{b}c$$

$$f(a,b,c) = \overline{\bar{f}(a,b,c)} = \overline{(\bar{a}b\bar{c} + \bar{b}c)}$$

Demorgan's law:

$\overline{A+B} = \bar{A}\bar{B}$	$\checkmark$
$\overline{AB} = \bar{A} + \bar{B}$	

$$f(a,b,c) = (\overline{\bar{a}b\bar{c}})(\overline{\bar{b}c}) = (a+\bar{b}+c)(\bar{b}+\bar{c})$$

## How to write min Pos :

- ① group all "zeros"  $\rightarrow$  essential Prime implicants  
and write the product
- ② writing the product terms and ignoring the  
changing variable  $\rightarrow$  sum them up  $\Rightarrow \bar{f}$
- ③ Use Demorgan's law to get  $f \Rightarrow f$

Example :

1	2	3
4	5	6
7	8	9
0		

Output = 1 , if we press  
multiples of 3 (including "0")

Q: Design a circuit that does that !

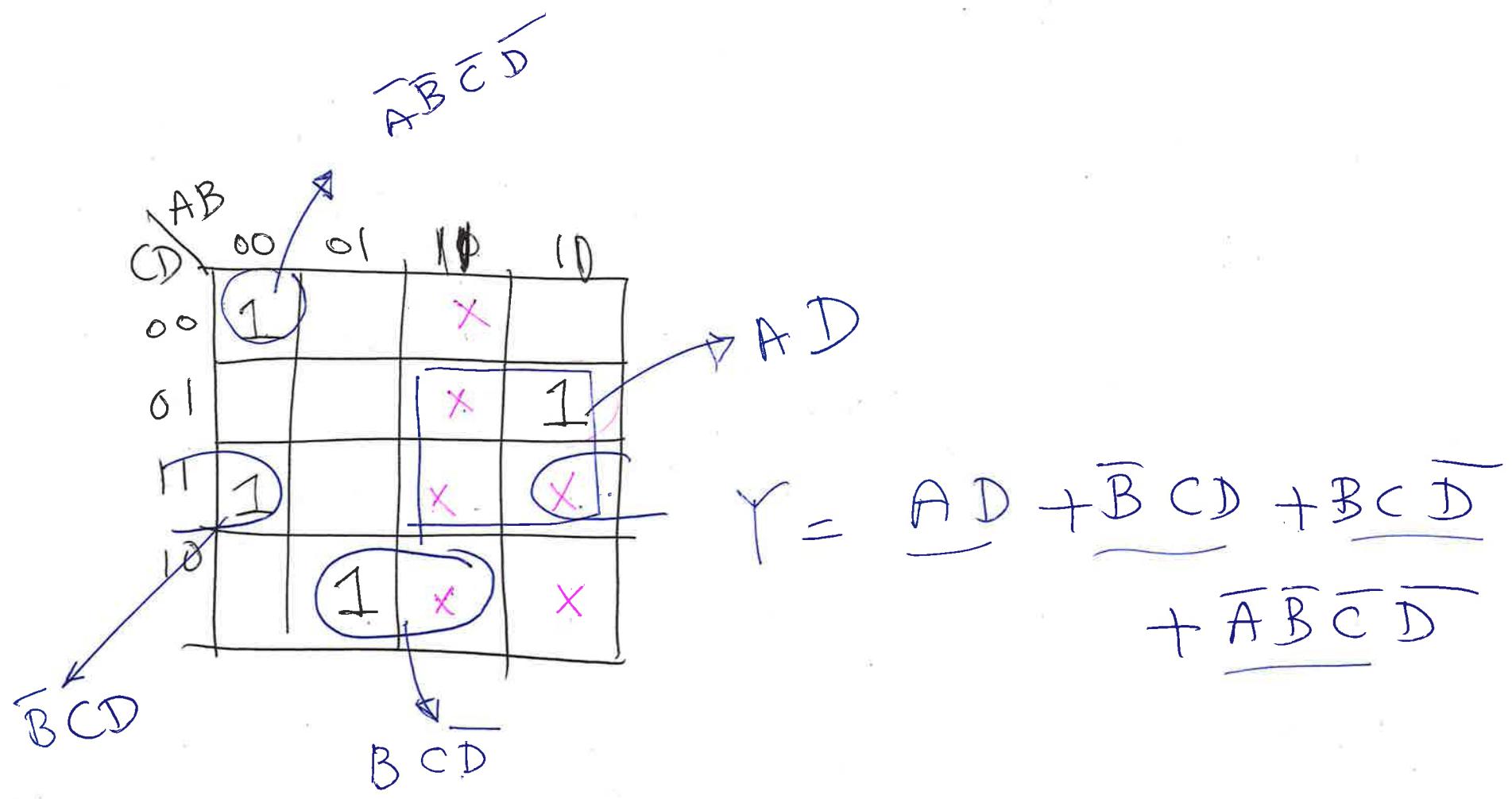
#	A	B	C	D	Y=output
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	X
11	1	0	1	1	X
12	1	1	0	0	X
13	1	1	0	1	X
14	1	1	1	0	X
15	1	1	1	1	X

$\text{SOP}$

$$Y = \sum m(0, 3, 6, 9) + \sum d(10, 11, 12, 13, 14, 15)$$

"don't care"

can either be "1" or "0"



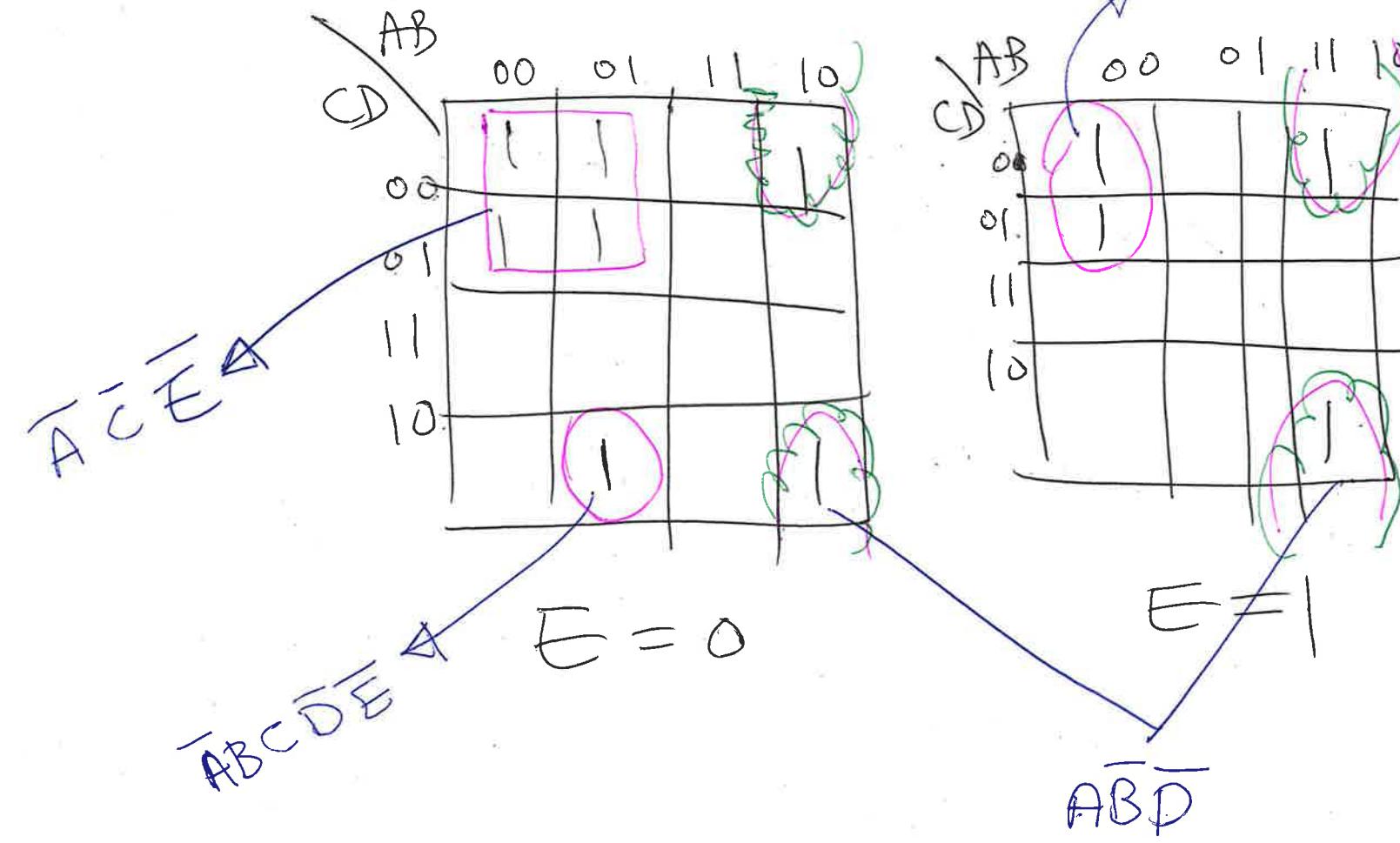
$$f(a, b, c) = \sum m(4, 6, 7) + \sum d(1, 5)$$

		ab	00	01	11	10
		c	0	1	1	0
a	0		X			
	1			X		

$$f(a, b, c) = a$$

=

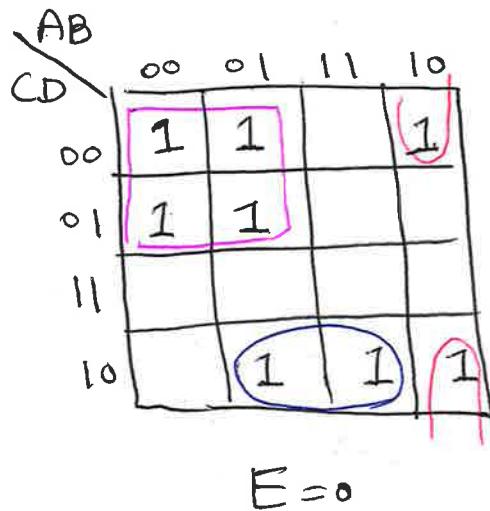
Example :  $F(A, B, C, D, E)$



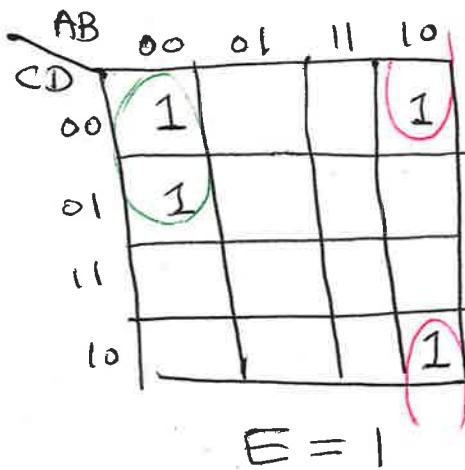
$$\begin{aligned}F(A, B, C, D, E) = & \overline{A}\overline{C}\overline{E} + \overline{A}B\overline{C}\overline{D}\overline{E} + \overline{A}\overline{B}\overline{C}E \\& + \overline{A}\overline{B}D\end{aligned}$$

## Karnaugh MAP (5 variables)

Find minimum SOP for the following 5-variable K-MAP:  $F(A, B, C, D, E) = ?$



$$F = \overline{A}\overline{C}\overline{E} + BC\overline{D}\overline{E} + \overline{A}\overline{B}\overline{C}E + A\overline{B}\overline{D}$$



(pay attention to color code.  
Each color represent an essential prime implicant.)

## Lecture 11 : Multiplexers & Decoders (Sep 24, 2020)

- ① what are the Multiplexers (Encoders) ?
- ② what are the demultiplexers (Decoders) ?
- ③ How to use them ?

### Announcement :

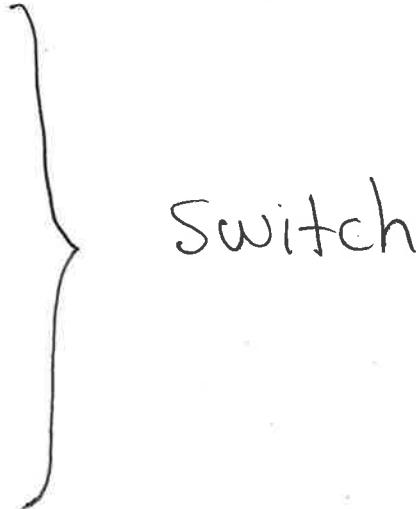
- HW3 is due Oct 1.
- Short Quiz 2 is on Oct 1.
  - It is exactly the same as short quiz 1 and must be submitted on canvas!

If (select 0)

$$Y = I_0$$

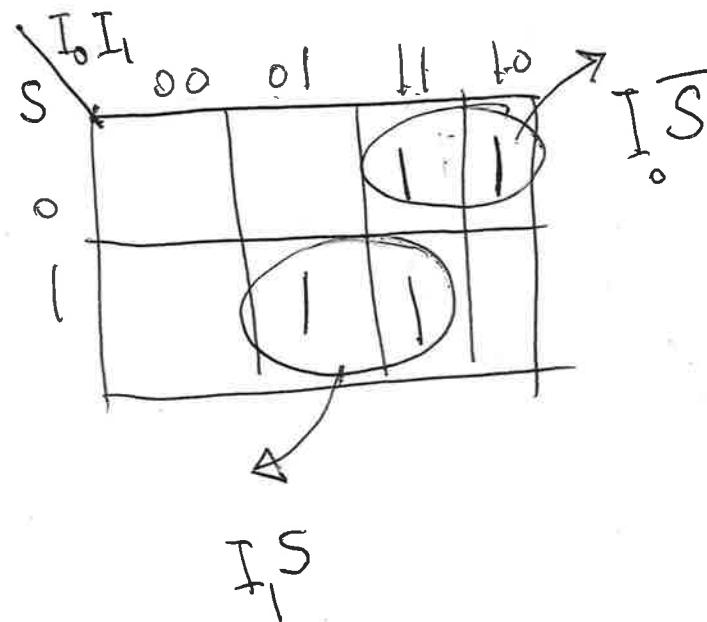
else (select 1)

$$Y = I_1$$



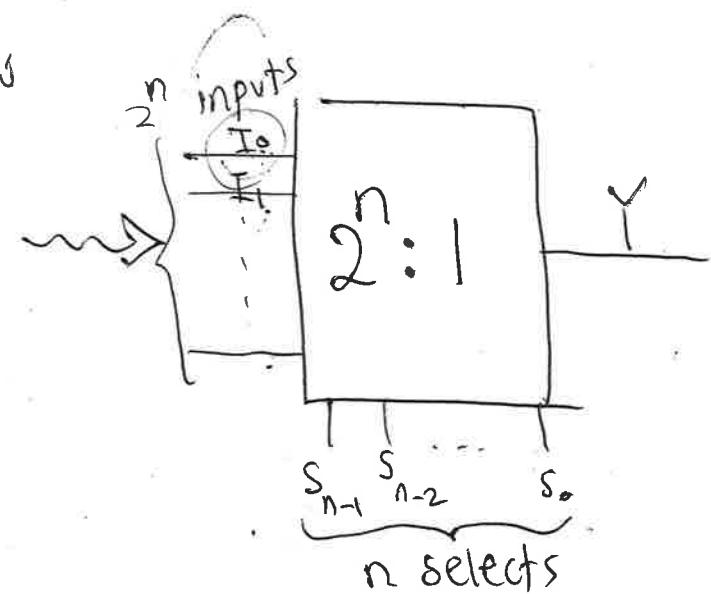
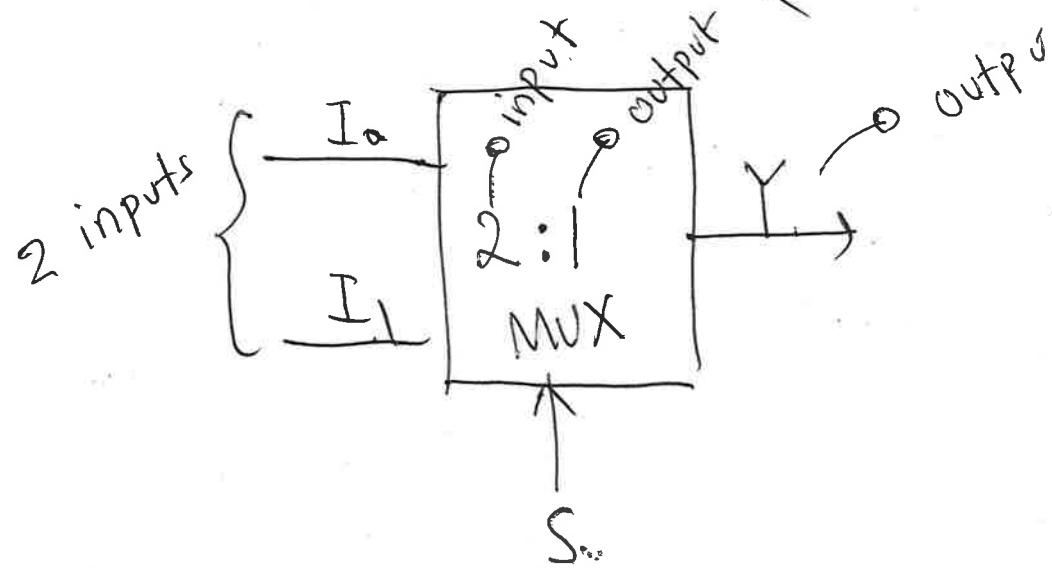
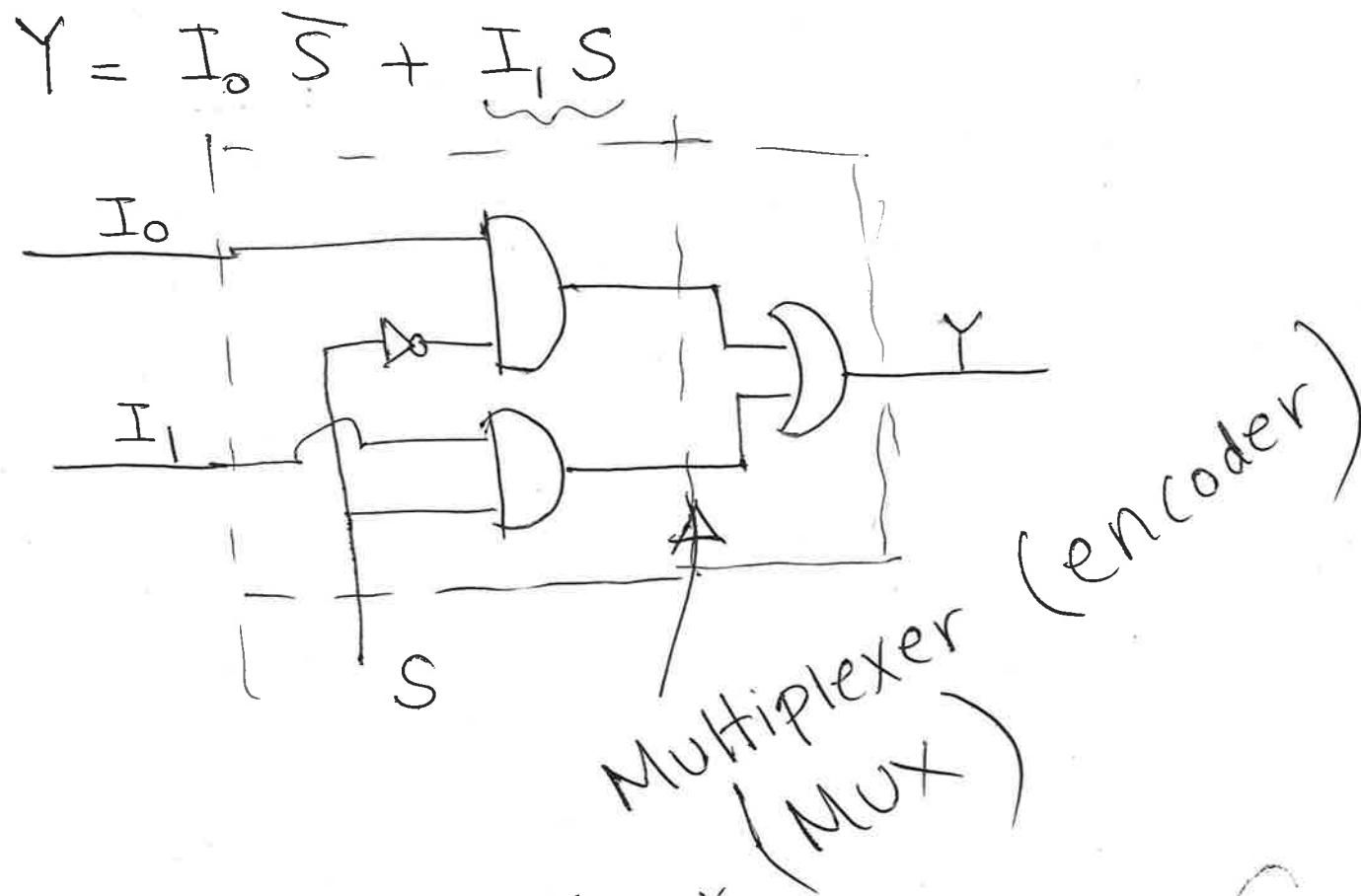
select

I <sub>0</sub>	I <sub>1</sub>	S	Y = output
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1 (T)	1 ↑
1	0	0	1 ↑
1	0	1	0
1	1	0	1 ↑
1	1	1	1 ↑



$$Y = I_0 \bar{S} + I_1 S$$

binary number  
represents the index



Think of Multiplexers as signal routing devices.

Application: CPU

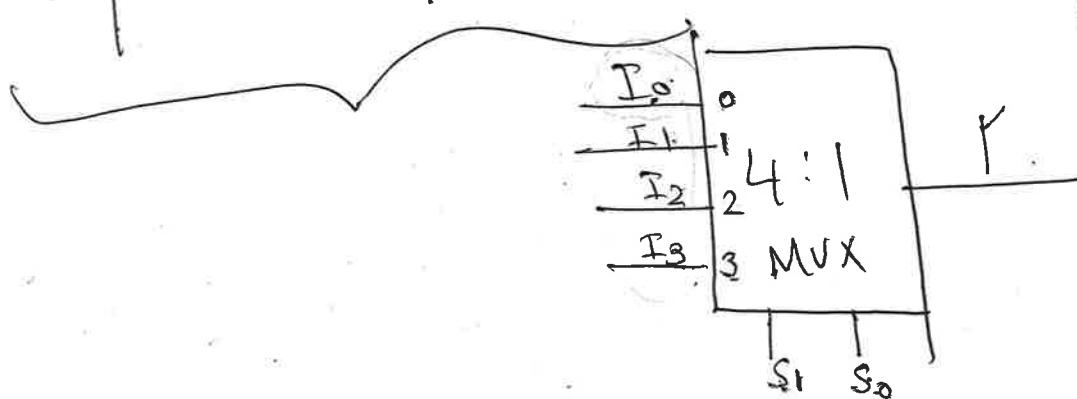
How to build  $4:1$  MUX?

$(I_0, I_1, I_2, I_3)$

#line	MSB $S_1$	LSB $S_0$	$Y = \text{output}$
0	0	0	$I_0$
1	0	1	$I_1$
2	1	0	$I_2$
3	1	1	$I_3$

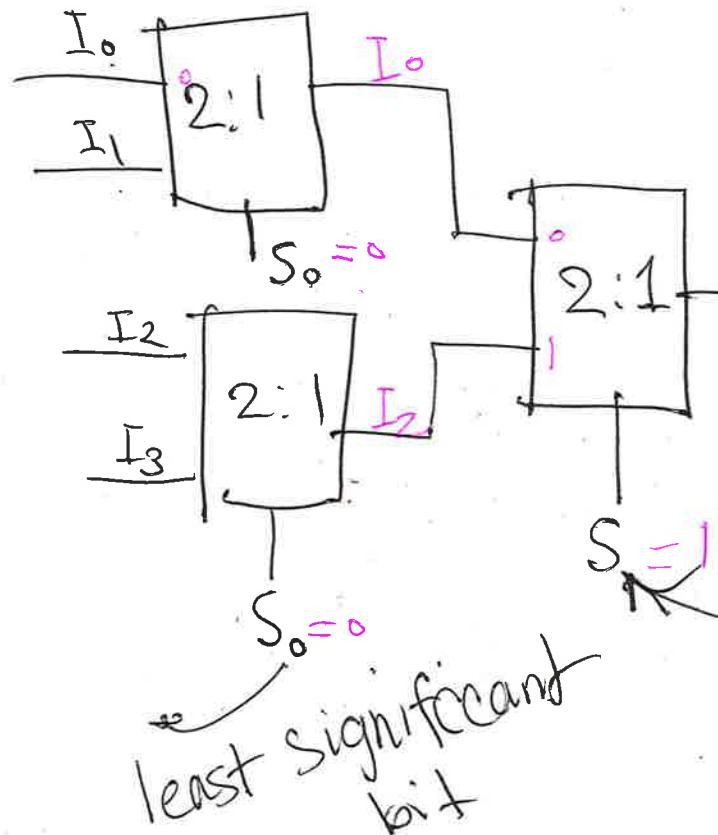
$$Y = I_0 \overline{S_1} \overline{S_0} + I_1 \overline{S_1} S_0 +$$

$$I_2 S_1 \overline{S_0} + I_3 S_1 S_0$$



$$f(a_1, b, c) = \underbrace{\sum m(1, 2, 4, 5)}_{\text{represent the output}} + \underbrace{\sum d(6)}_{\text{not input}}$$

Q: Can we build a 4:1 MUX using 2:1 Mixers?



Example:

$$\left| \begin{array}{l} S_1 = 1 \\ S_0 = 0 \end{array} \right. \rightarrow I_2$$

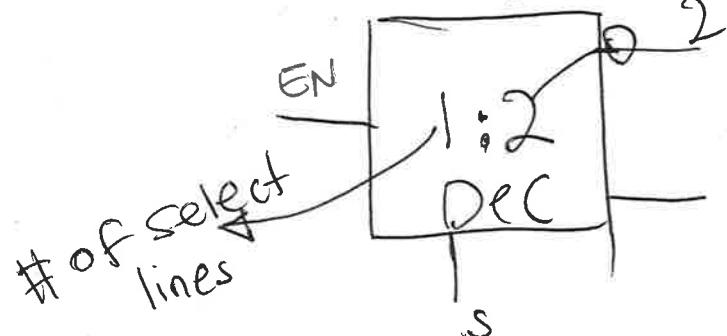
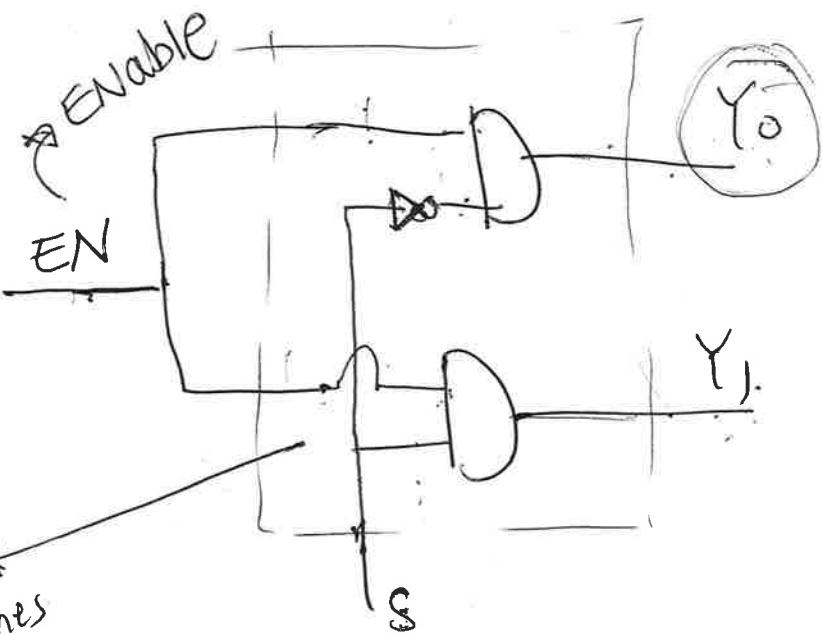
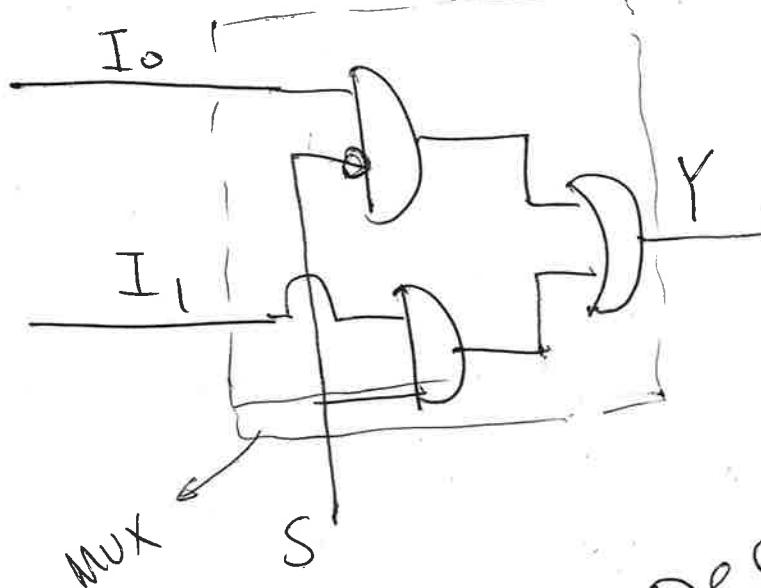
		$S_1$	$S_0$	$Y$
0	0			$I_0$
0	1			$I_1$
1	0			$I_2$
1	1			$I_3$

		$S$	
0			$I_0$
1			$I_1$

# Demultiplexers (Decoders)

① one input & multiple outputs

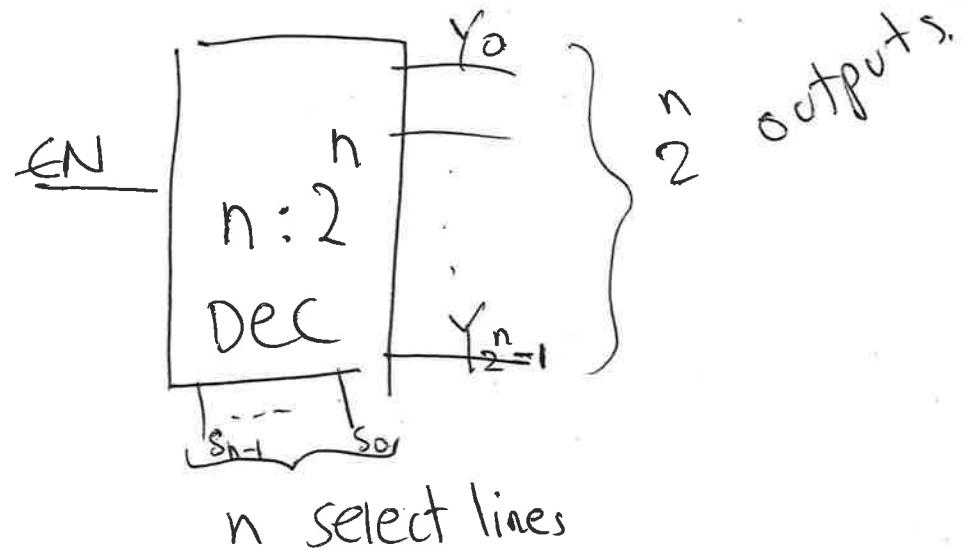
② select line(s) determine which output is active



Decoder  
#select lines

$EN = 0 \Rightarrow$  Decoder is not on

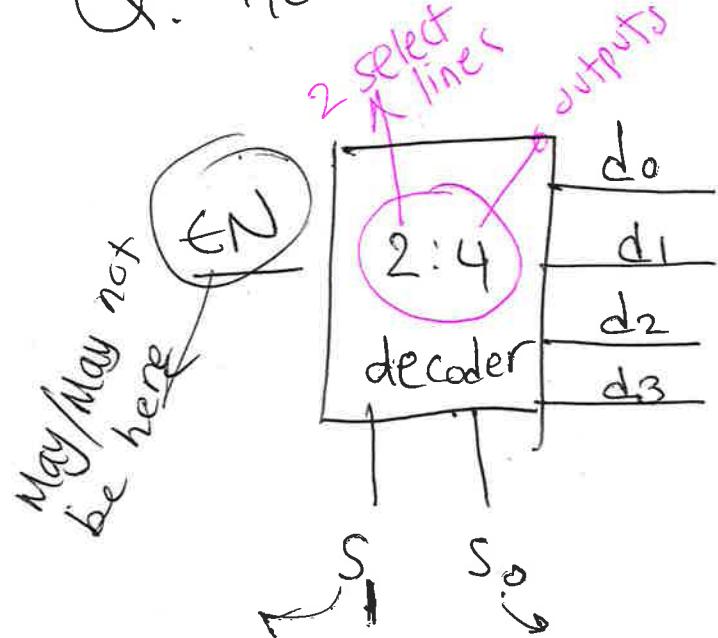
$EN = 1 \Rightarrow \begin{cases} S = 0 \Rightarrow Y_0 \\ S = 1 \Rightarrow Y_1 \end{cases}$



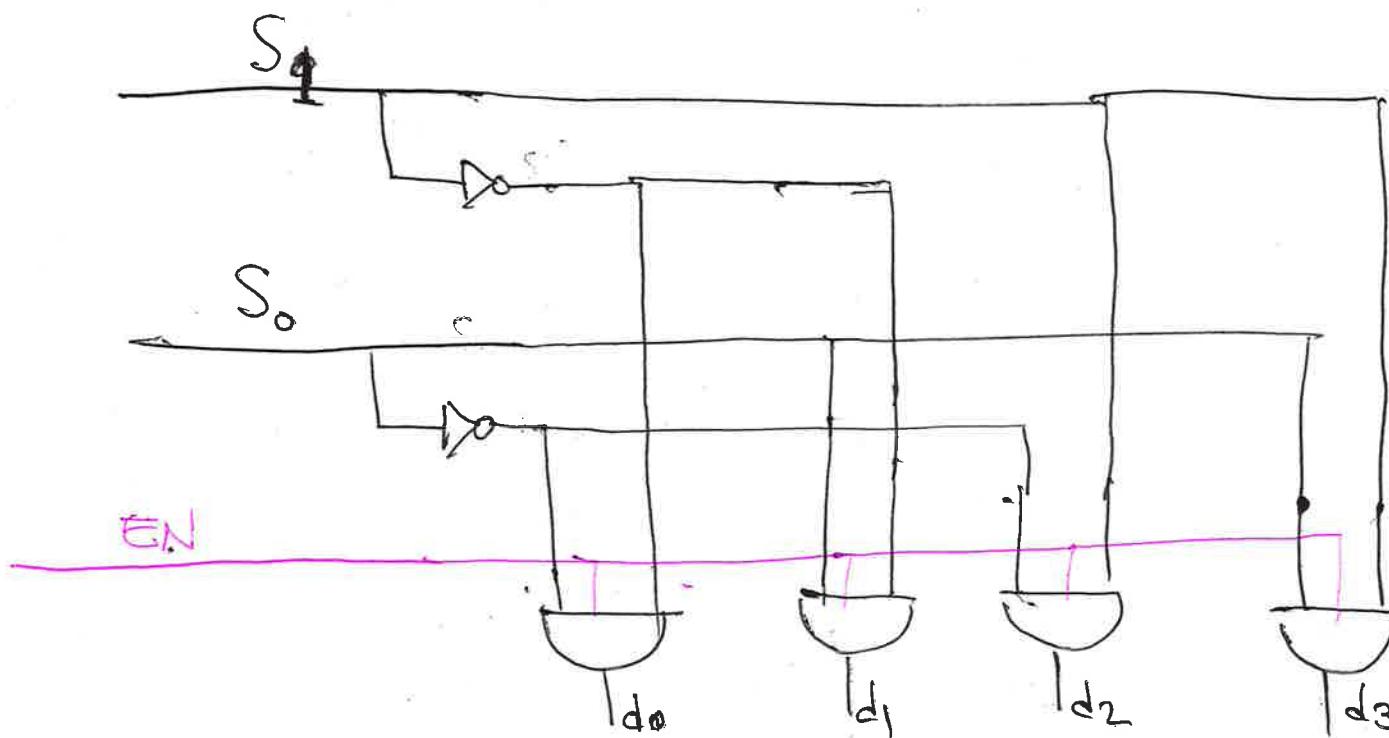
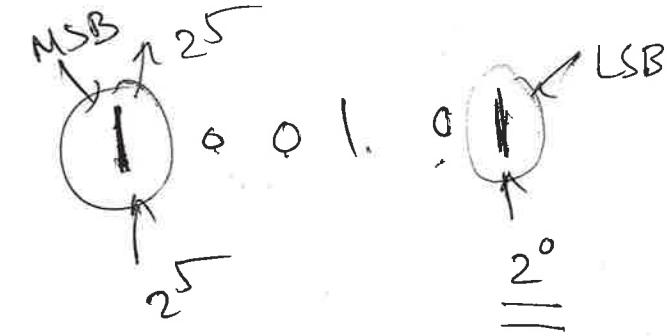
$\bar{EN}=1$  says, we generate output.

$\bar{EN}=0 \Rightarrow$  output = 0

Q: How to build 2:4 decoders?



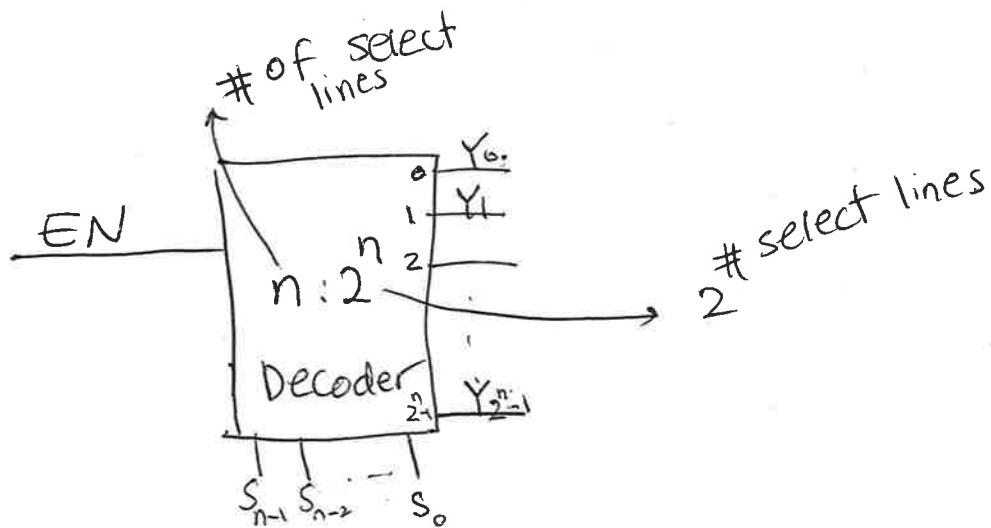
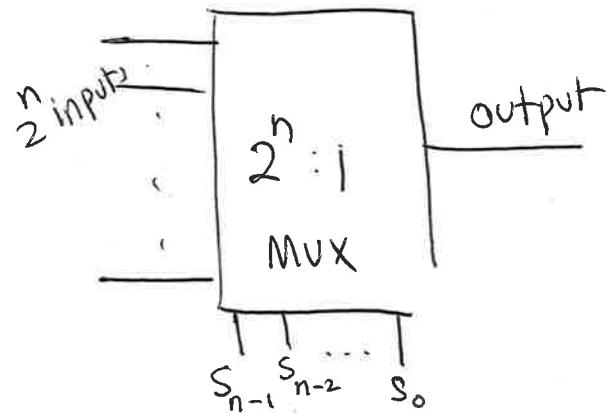
$$\begin{aligned} d_0 &\leftarrow \overline{S_1} \overline{S_0} \\ d_1 &\leftarrow \overline{S_1} S_0 \\ d_2 &\leftarrow S_1 \overline{S_0} \\ d_3 &\leftarrow S_1 S_0 \end{aligned}$$



Sep 29, 2020

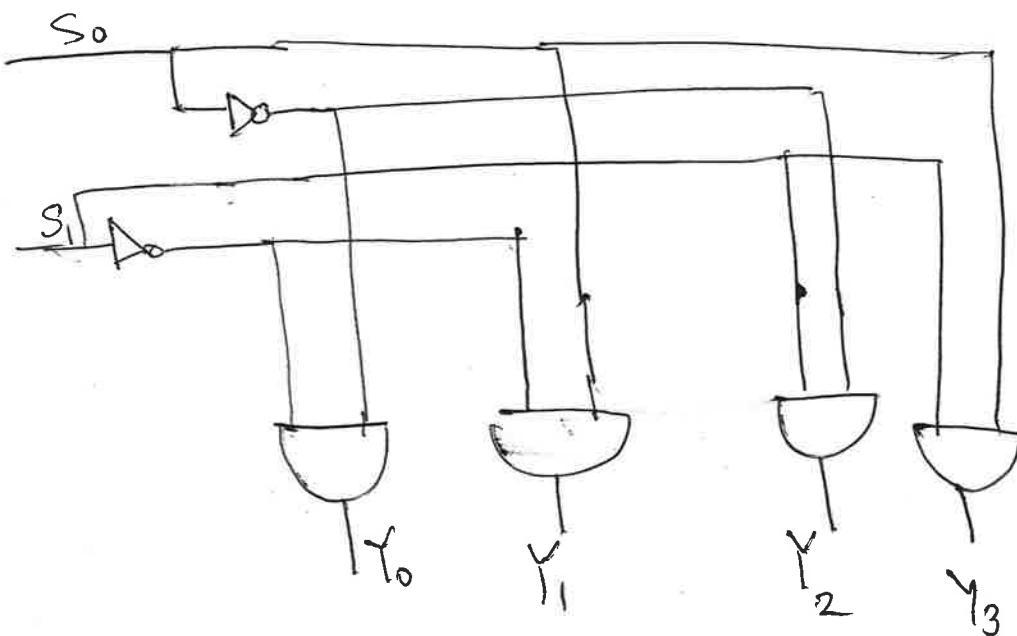
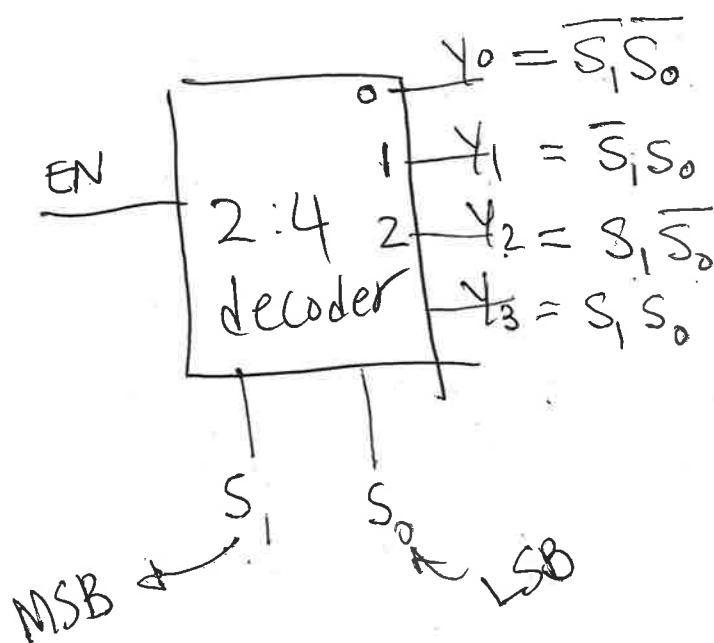
## Lecture 12: What are the applications of MUXes & Decoders?

- HW 3 is due Oct 1.
- Short Quiz 2 is on Oct 1.
- Lab 1 due is extended.

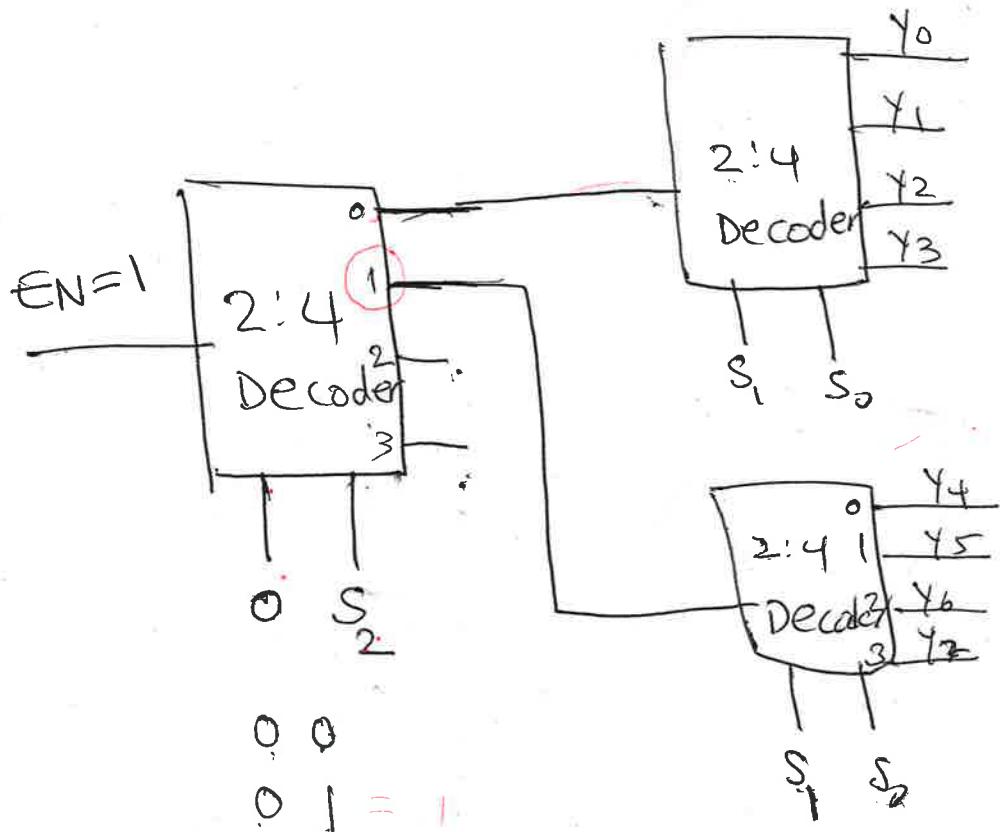


$2^n$  inputs  $\Leftrightarrow$  only one output

Q: How to build a 2:4 decoder



Q: Can we build a 3:8 decoder using 2:4 decoders?



$S_2$	$S_1$	$S_0$	Output
0	0	0	$y_0$
0	0	1	$y_1$
0	1	0	$y_2$
0	1	1	$y_3$
1	0	0	$y_4$
1	0	1	$y_5$
1	1	0	$y_6$
1	1	1	$y_7$

$y_i \in \{0, 1\}$

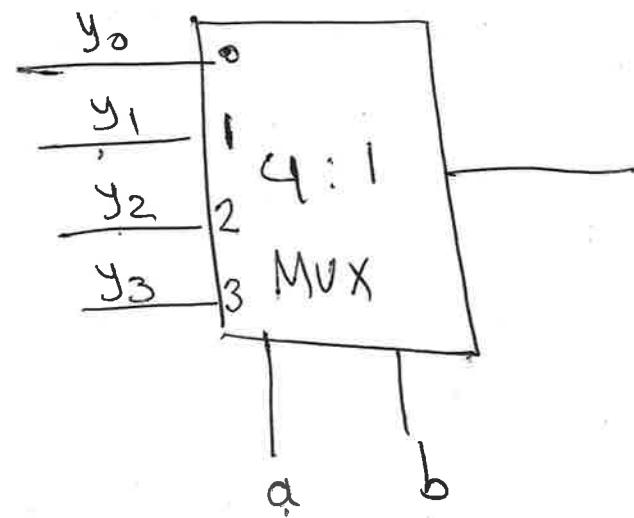
$$\underbrace{OS_2}_2 = S_2$$

Any logic function w/ "n" inputs can be implemented w/  
 $2^n$ :1 Multiplexer.

Example:

a	b	Y=output
0	0	$y_0$
0	1	$y_1$
1	0	$y_2$
1	1	$y_3$

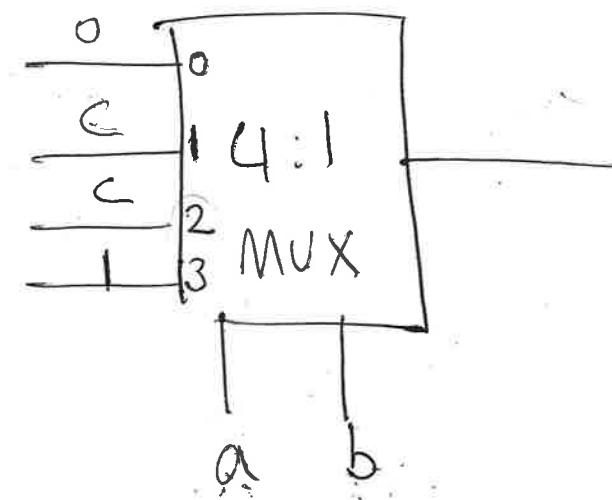
$$y_i \in \{0,1\}$$



Example : 3 inputs , output = majority voting  $\Rightarrow$  use 4:1 MUX

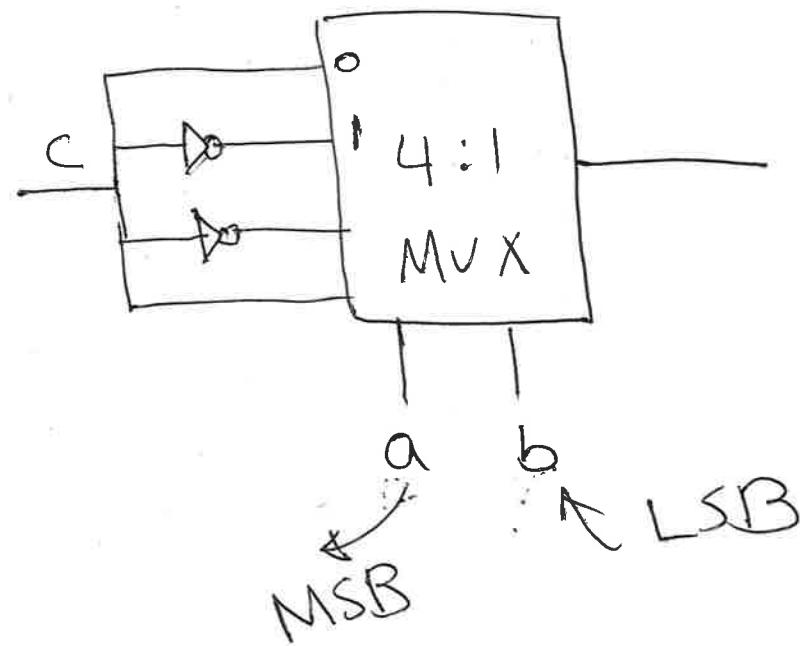
to build the circuit

a	b	c	output=Y   output
0	0	0	0   0
0	0	1	0   0
0	1	0	0   C
0	1	1	1   C
1	0	0	0   C
1	0	1	1   C
1	1	0	1   C
1	1	1	1   1

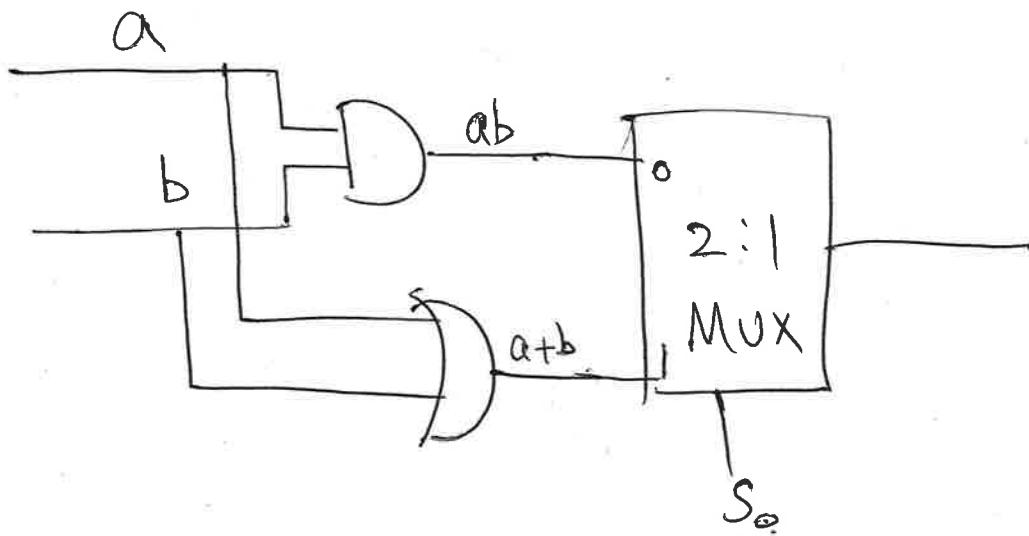


Example : (parity)

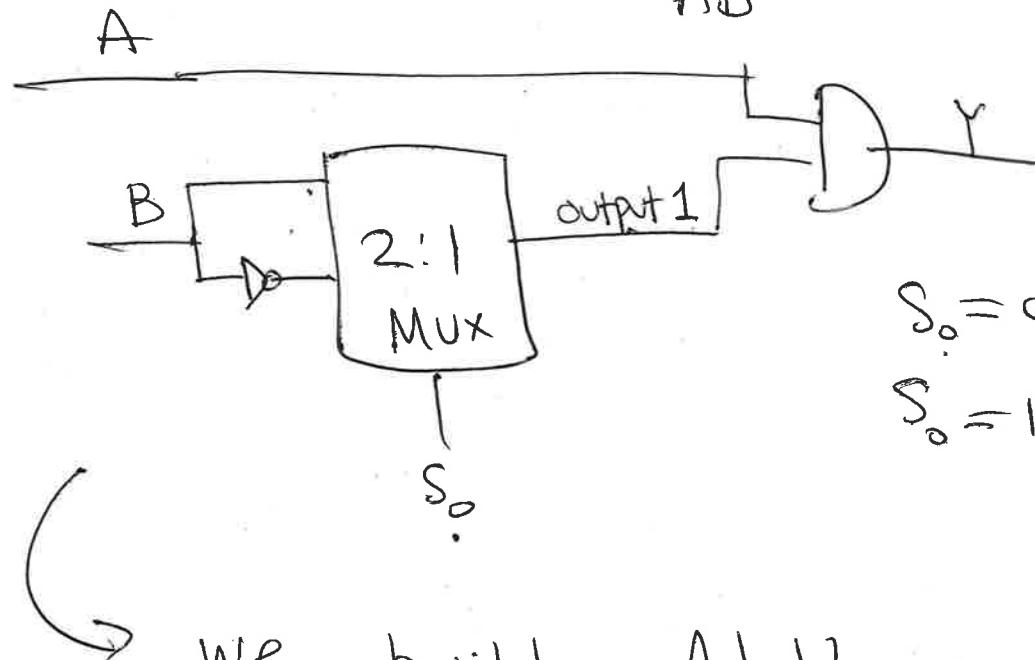
a	b	c	$y = \text{output}$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1



Example : use a mux to build a circuit that allows one of the operations ("AND", "OR") to be chosen and applied to an input.



Example: Use a Mux to build a circuit that evaluates either "A and B" or "A and not B".



$$S_0 = 0 \Rightarrow \text{output 1} = B \Rightarrow Y = AB$$

$$S_0 = 1 \Rightarrow \text{output 1} = \overline{B} \Rightarrow Y = A\overline{B}$$

we build

ALU      logic unit

Arithmetic

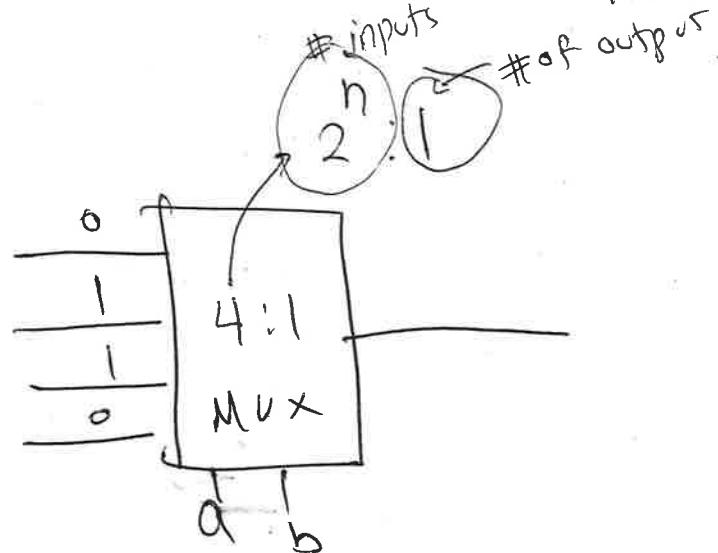
Carry out a range of Calculations

on its input.

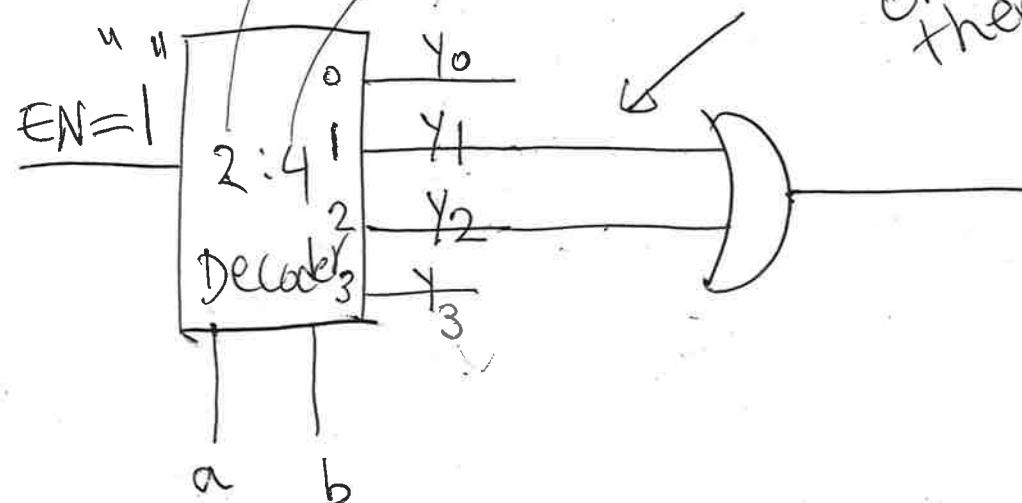
Example: Use 2:4 decoder to build XOR function.

	a	b	$Y = \text{XOR}^u$
0	0	0	0
1	0	1	1 $\leftarrow \bar{ab}$
2	1	0	1 $\leftarrow ab$
3	1	1	0

$$Y = \bar{a}\bar{b} + a\bar{b} + \bar{a}b$$



$$Y = \# \text{of select lines}^2 \text{ select line}$$



wire ones and input them to an XOR gate.