Lecture 1: Introduction to EEE/CSE 120

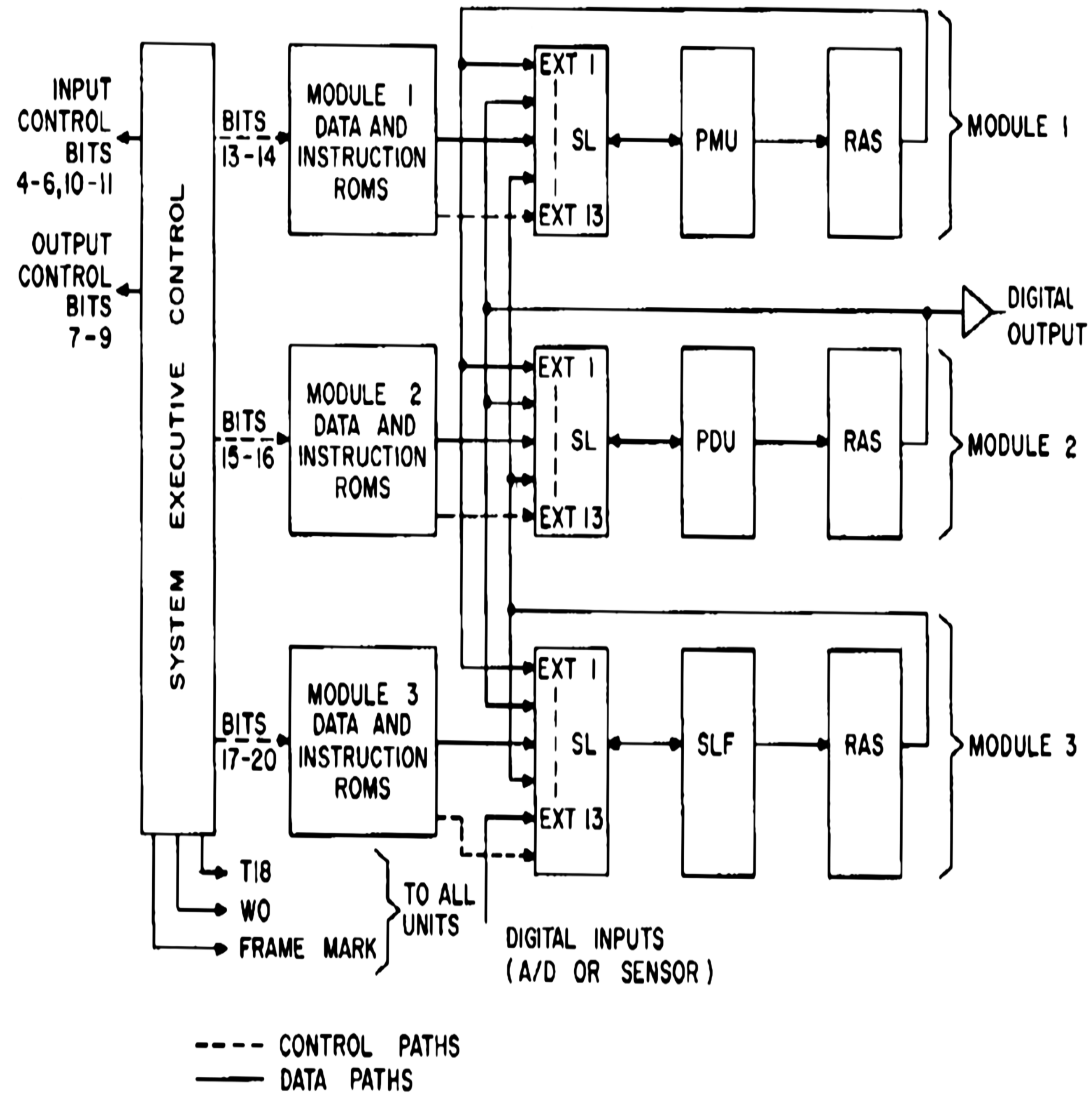
Bahman Moraffah
Electrical, Computer, and Energy Engineering
Arizona State University

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
- 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 though 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

	Distributions
Lab Report	25%
Assignment	10%
Quizzes and Attendence	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	Synchrounous 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)

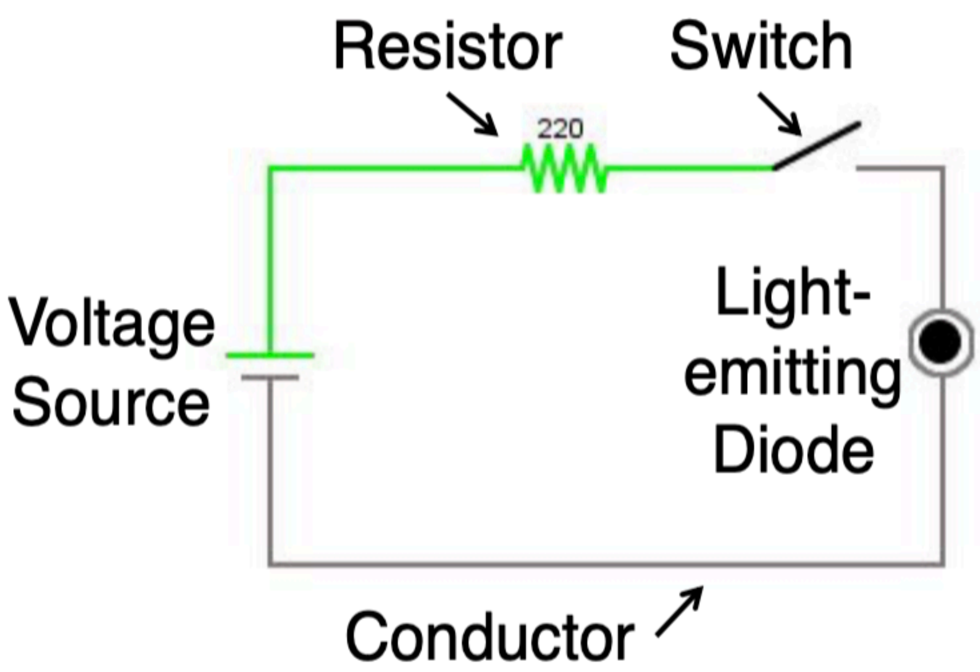
Why digital design

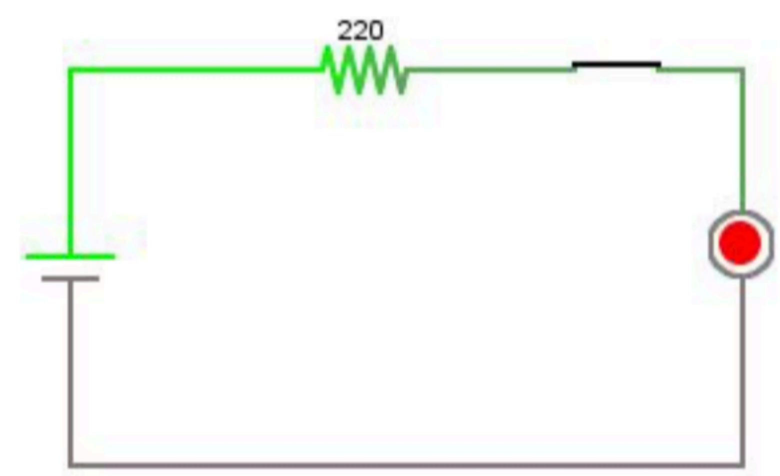
- In 17th century electricity was discovered.
- Built a switch to turn things on and off
  - They were huge switches

- 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

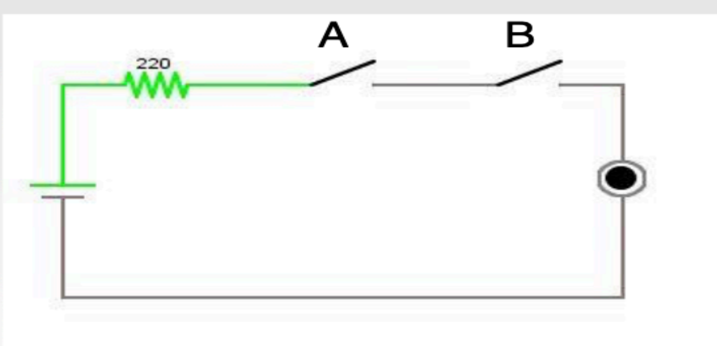
Switches: Connection

❑ Series Connection

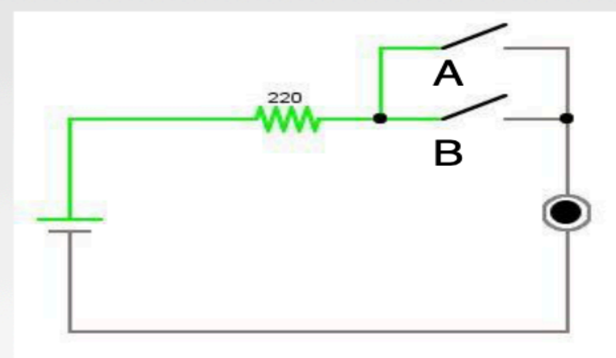
- If switches are back to back
- If two switches are in series, then circuit is closed only if switch A and switch B are both closed

❑ 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

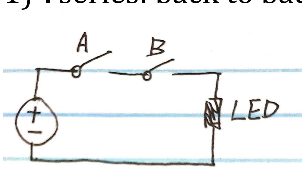
EEE/CSE 120: Digital Design Fundamentals Fall 2020

Lecture 2: Gates and Truth Tables

Lecturer: Dr. Bahman Moraffah Scribes: Xinyi Ma

How to connect switches?

1) . series: back to back



Switch off ⟺ False ⟺ 0

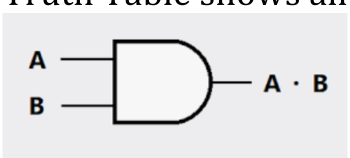
Switch on ⟺ True ⟺ 1

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:

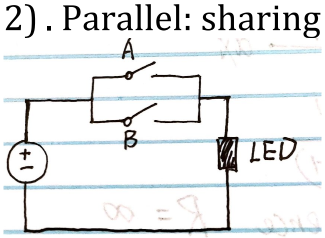
A	B	Y=Output
0	0	0
0	1	0
1	0	0
1	1	1

Truth Table shows all possibilities of outputs.



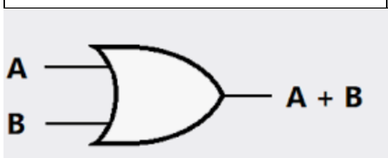
"AND" gate only true when both A and B are true.

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



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

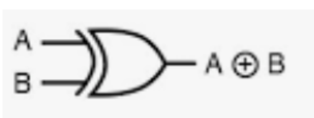
A	B	Y=Output
0	0	0
0	1	1
1	0	1
1	1	1



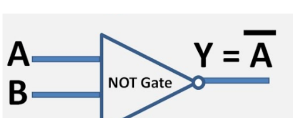
"OR" gate will be true if A is true or B is true.

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

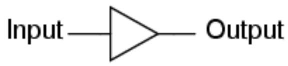


"XOR" gate(not series, not parallel) gives true output when the number of true inputs is odd.
Explanation:
LED is on when 1). Switch A is on **and** Switch B is off(not on) **OR**
2). Switch A is off(not on) **and** Switch B is on



A	Y
0	1
1	0

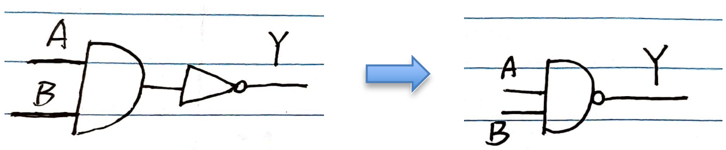
"NOT" gate is one the outputs the opposite state as what is input.



"Buffer" gate is the one that outputs equal to its inputs.

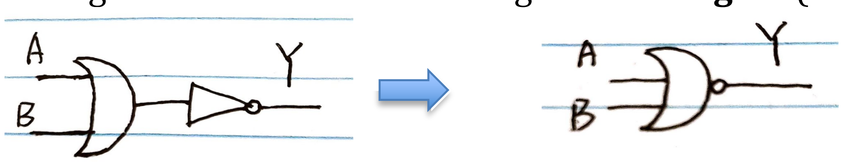
A(Input)	Y(Output)
0	0
1	1

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



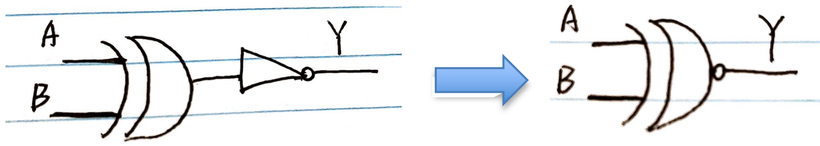
A	B	Y(final output)	C(before"not")
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

"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

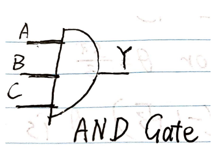
"XOR" gate combines with "NOT" gate:**"XNOR" gate**



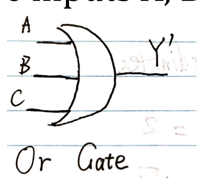
A	B	Y(final output)	C(before"not")
0	0	1	0
0	1	0	1
1	0	0	1
1	1	1	0

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

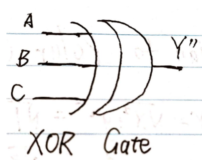
Example:
There are three inputs A, B, and C go through the following three gates



,



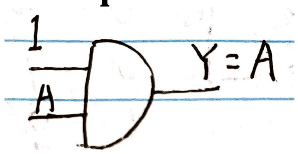
and



, list the outputs for truth table for Y, Y', and Y''.

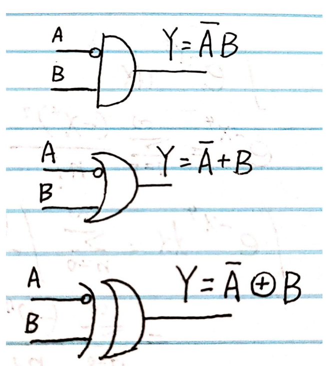
A	B	C	Y	Y'	Y''
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	0	1	1
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	1	1	1

Example 1.

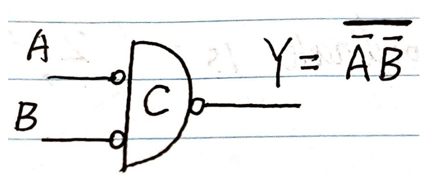


The output Y depends on A: A $\nearrow$ 0 $\rightarrow$ Y=0
so the output Y=A $\searrow$ 1 $\rightarrow$ Y=1

Example 2.



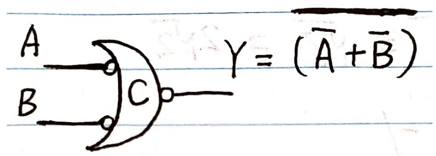
Example 3.



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)

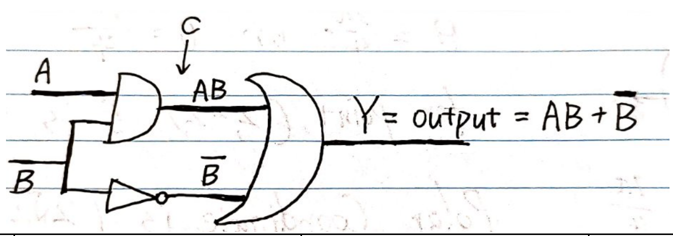
Example 4.



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

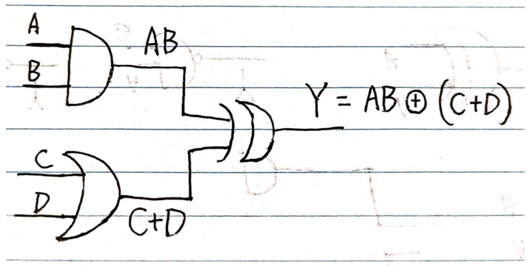
(the final outputs are the same as "AND" gate)

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.

$$(6\,5\,4)_{10} = \underbrace{4}_{\text{coeff.}} \times 10^0 + 5 \times 10^1 + 6 \times \underbrace{10}_{\text{base}}^{\boxed{2}\,\text{exponent}} = \sum_{i=0}^{n} c_i\, 10^i$$

$10^2 \quad 10^1 \quad 10^0$

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

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

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

$(\ldots)_2$

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

Q: 654 in terms of powers of Two?

(Binary rep. of 654)

Binary	Decimal
2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32
2^6	64
2^7	128

Binary	Decimal
2^8	256
2^9	512
2^{10}	1024
2^{11}	2048
$\vdots$	

$$654 = 1\times \underbrace{2^9 +}_{0\times 2^8} \underline{\underline{1}}\times 2^7 + 1\times 2^3 + \underline{1}\times 2^2 + 1\times 2^1 + 0\times 2^0$$

$$(1\,0\,1\,0\,0\,0\,1\,1\,1\,0)_2$$

$2^1 \quad 2^0$

$\underline{\text{base} = 2^3} = 8 \longrightarrow$ octal (oct) $\longleftrightarrow c_i \in \{0,1,2,\dots,7\}$ $(c_i < 8)$

$\underline{\text{base} = 2^4} = 16 \longleftrightarrow$ HEXADECIMAL (HEX) $\longleftrightarrow c_i < 16$

0 – 9

$10 \longleftrightarrow A$	$12 \longleftrightarrow C$	$15 \longleftrightarrow F$
$11 \longleftrightarrow B$	$13 \longleftrightarrow D$	
	$14 \longleftrightarrow E$	

$$(1\,|\,0\;1\;1\,|\,\underset{2^2}{0}\;\underset{2^1}{1}\;\underset{2^0}{1})_2$$

<u>Example</u>:

$$654 = (001\,|\,0\;1\;0\,|\,0\;0\;1\,|\,1\;1\;0)_2$$

Q: What is the Oct?

$$\underbrace{001}_{1}\quad \underbrace{0\times2^0+1\times2^1+0\times2^2}_{2}\quad \underbrace{001}_{1}\quad \underbrace{0\times2^0+1\times2^1+1\times2^2}_{6}$$

$$(1216)_8$$

EX:

HEX ?

$$654 = (\underbrace{0\,0\,1\,0}_{1\times 2^1 = 2}\ |\ \underbrace{1\,0\,0\,0}_{2^3 = 8}\ |\ \underbrace{1\,1\,1\,0}_{\substack{0\times 2^0 + 1\times 2^1 \\ + 1\times 2^2 + 1\times 2^3 = 14}})_2$$

E

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

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

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

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

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

$B = 11 \longleftrightarrow 1\times 2^3 + 1\times 2^1 + 1\times 2^0 \longleftrightarrow (1\ 0\ 1\ 1)$

$(1\ 0\ 1\ 1\quad 0\ 0\ 1\ 1\quad 0\ 1\ 1\ 0\quad 1\ 1\ 1\ 0)_2$

654	2	Q: 327	R = 0
327	2	163	1
163	2	81	1
81	2	40	1
40	2	20	0
20	2	10	0
10	2	5	0
5	2	2	1
2	2	$\boxed{1}$	0

R 2 | 654 R

least significant digit

most significant

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

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

$\vdots$

$$(1\,0\,1\,0\,0\,0\,1\,1\,1\,0)_2$$

Ex:

$$\begin{array}{r} 5 \\ +^{1}\ 7 \\ \hline 12 \end{array}$$

binary:

$$\begin{array}{rcccc} & & 1 & & \\ & 1 & 0 & 1 & 1 \\ + & 0 & 0 & 1 & 0 \\ \hline & 1 & 1 & 0 & 1 \end{array}
\quad
\begin{array}{l} \longrightarrow 1\times 2^0 + 1\times 2^1 + 0\times 2^2 + 1\times 2^3 = 11 \\ \longrightarrow \quad 2 \quad + \\ \hline \quad 13 \end{array}$$

$$\underbrace{1\times 2^0}_{1} + 0\times 2^1 + \underbrace{1\times 2^2}_{4} + \underbrace{1\times 2^3}_{8} = 13$$

✓

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

How do we define negative numbers ?

$\boxed{1}$ Signed bit.

negative

$1 \; (0 \; 0 \; 1) \longrightarrow -1$

$0 \; 0 \; 0 \; 1 \longrightarrow +1$

positive

"0" $\rightarrow$ $\left.\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}\right\}$ Two rep.

$$
\begin{array}{rcccccc}
 & & & & & 1 & \\
-1 & \rightarrow & 1 & 0 & 0 & 1 & \\
1 & \rightarrow & 0 & 0 & 0 & 1 & \\
\hline
 & 0 & 1 & 0 & 1 & 0 & \longrightarrow 2^3+2^1=10
\end{array}
$$

X

[2] offsetting

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

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

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

[3] 2's Complement.

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

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

[1] Find a number which if added to the original number $\longrightarrow$ 111

[2] Add 1 to the number $\longrightarrow$ 001

Lecture 4:

EEE/CSE 120 : 2's complement and it's Arithmatic

- HW 1 is due sep 3
- Lab0 is due Sep 14
- Quiz 1 is on Sep 8
- office hours T-Th 9:30 AM
- 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

Defening negative numbers:

$\boxed{1}$ 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) $1+(-1) \neq 0$

$\boxed{2}$ offsetting

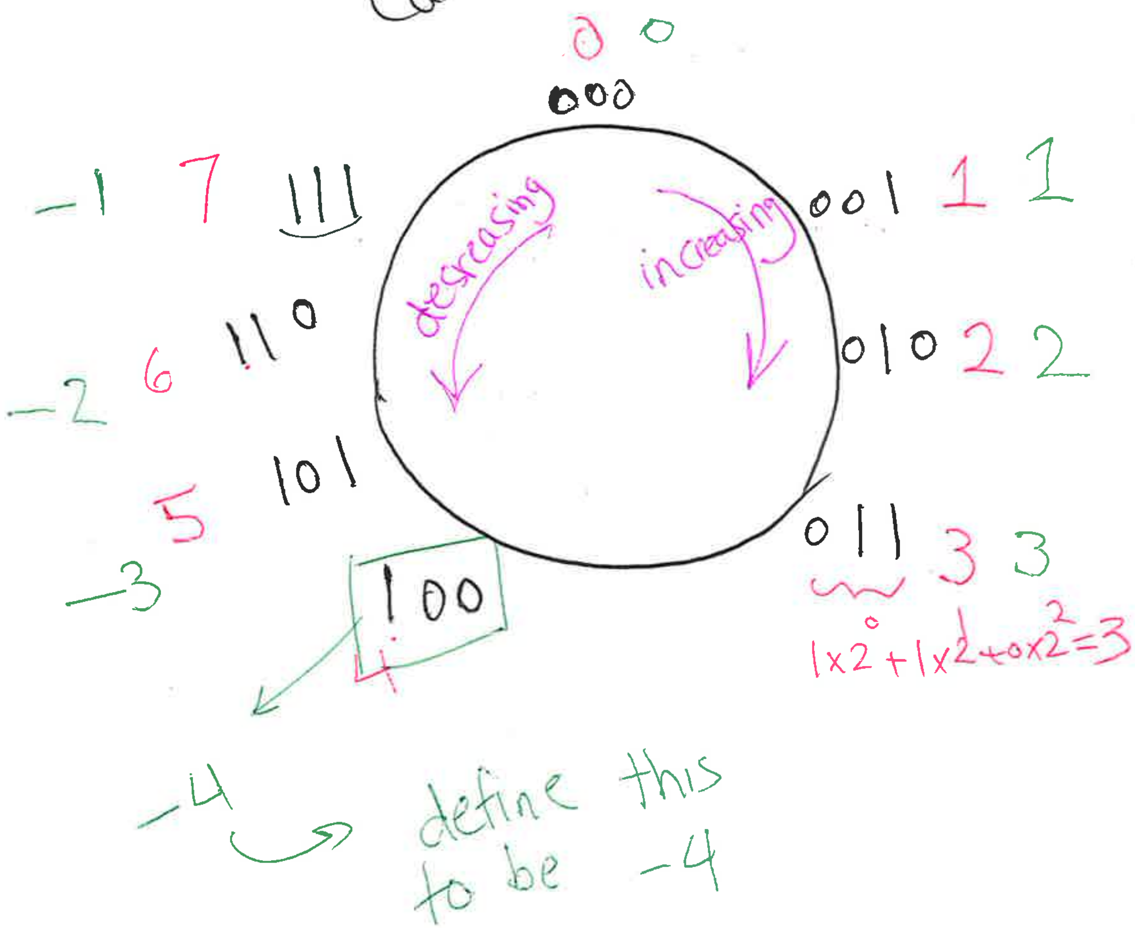
problem: $1+(-1) \neq 0$

$\boxed{3}$ 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.

$$\begin{array}{r} (+3)_{10} \\ +\ (-3)_{10} \\ \hline (0)_{10} \end{array} \quad \longleftrightarrow \quad \begin{array}{r} (011)_2 \\ \boxed{1}\boxed{0}\boxed{1} \\ \hline \cancel{1}\ 000 \end{array} \rightarrow (-3)_{10}$$

ignore carry out



unsigned numbers

Signed number

How to get a negative number from a positive number?

Rule: $\boxed{1}$ Find the number which added to the original number that results in 111

$\boxed{2}$ add 1 to that number

Example:

$$\begin{array}{r} 0\ 1\ 1 \\ +\ \boxed{1}\ \boxed{0}\ \boxed{1} \\ \hline \boxed{1}\ 0\ 0\ 0 \end{array} = 111 + 001$$

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

Another way of writhing the rule :

$\boxed{1}$ Flip all the bits of the original number (one's complement)

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

$\boxed{2}$ 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

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

Example: 2's Complement of a negative number!

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

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

② Add 1 to it

$$
\begin{array}{r}
0\ 1\ 0 \\
+\ 0\ 0\ 1 \\
\hline
\underbrace{\boxed{0}\ 1\ 1}_{1\times 2^0 + 1\times 2^1 = 3}
\end{array}
$$

Example: 2's Complement of 000?

① flipping the bits $\longrightarrow$ 1 1 1

② To add 1 to it $\longrightarrow$

$$
\begin{array}{r}
1\ 1\ 1 \\
0\ 0\ 1 \\
\hline
\cancel{1}\ 0\ 0\ 0
\end{array}
$$

Example: $\boxed{1}\ 0\ 1$ $\quad (-3)_{10}$

① $0\ 1\ 0$

② $\begin{array}{r} 0\ 1\ 0 \\ 0\ 0\ 1 \\ \hline 0\ 1\ 1 \end{array} \longrightarrow +3$

Example: $\underbrace{1\ 0\ 0\ 0}_{(-8)_{10}}$ (4 bit)

2's Compl.: ① $0\ 1\ 1\ 1$

② $\begin{array}{r} {\scriptstyle 1\ 1\ 1\ \ } \\ 0\ 1\ 1\ 1 \\ 0\ 0\ 0\ 1 \\ \hline 1\ 0\ 0\ 0 \end{array} \longrightarrow +8$

Example : 0 1 1 0

$\boxed{1}$ Flip all the bits : 1 0 0 1

$\boxed{2}$ to add 1 to it

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

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

most sign. 0 1 $\boxed{1}$ 0 least sig.

flip unchanged

1 0 1 0

Example

Signed binary

↳ Addition :

① numbers can be pos. or neg.

② Addition doesn't care about sign.

$\boxed{1}$

pos. → 0 1 0 1 → 5   5

\+ 1 0 0 1 → $\underbrace{0111}_{+7}$ → −7

Neg

$\boxed{1}$ 1 1 0 ← $(-2)_{10}$ $(-2)_{10}$

0 0 1 0 → $(+2)_{10}$

Matched.

Example:

$$
\begin{array}{cccccl}
 & \overset{\text{Neg}}{\boxed{1}}\ \overset{\text{carry in}}{\boxed{0}} & 0 & 0 & 1 & \to \underbrace{0\,1\,1\,1}_{7} \to (-7) \\
+ & \underset{\text{Neg}}{\boxed{1}} & 1 & 1 & 0 & \to \underbrace{0\,0\,1\,0}_{2} \leftrightarrow (-2) \\
\hline
\text{Carry out} \leftarrow & \cancel{\boxed{1}} & \multicolumn{4}{l}{\underbrace{0\ \ 1\ \ 1\ \ 1}} \neq -9
\end{array}
$$

Don't match

overflow.

pos Neg
pos Neg
we might have OF.

pos Neg
Neg post
NO OF.

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

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

$$
\begin{array}{r}
\boxed{0} \quad\quad\quad\quad\quad\quad \\
1\ 0\ 0\ 0 \rightarrow -8 \\
+\ 0\ 1\ 0\ 1 \rightarrow 5 \\
\hline
\boxed{0}\ \underbrace{1\ 1\ 0\ 1}_{(-3)_{10}}\ \ (-3)_{10}
\end{array}
$$

Carry in = Carry out

$0\ 0\ 1\ 1 \rightarrow -3$

$$
\begin{array}{r}
\boxed{1} \xrightarrow{\text{carry in}} \quad\quad\quad \\
0\ 1\ 0\ 0 \rightarrow +4 \\
+\ 0\ 1\ 1\ 0 \quad +6 \\
\hline
\boxed{0}\ 1\ 0\ 1\ 0\ \ (10)_{10}
\end{array}
$$

carry out

Carry out $\neq$ carry in

OF.

$\underbrace{0110}_{+6}$

$1010 = -6$

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

Carry in

$$\begin{array}{cccccl}
 & \boxed{1} & \boxed{1} & & & \\
 & 1 & 1 & 1 & 1 & \rightarrow (0\,0\,0\,1) = 1 \rightarrow (-1)_{10} \\
+ & 1 & 1 & 1 & 0 & \rightarrow (0\,0\,1\,0) = 2 \rightarrow (-2)_{10} \\
\boxed{1} & & & & & \\
\hline
\boxed{1} & \underbrace{1 \quad 1 \quad 0 \quad 1} & & & & \leftarrow (-3)_{10} \ \text{Match.} = (-3)_{10}
\end{array}$$

carryout

Carry in = Carry out $\Longrightarrow$ NO OF.

2's comp

$(0\,0\,1\,1) = 3$

Lecture 5:

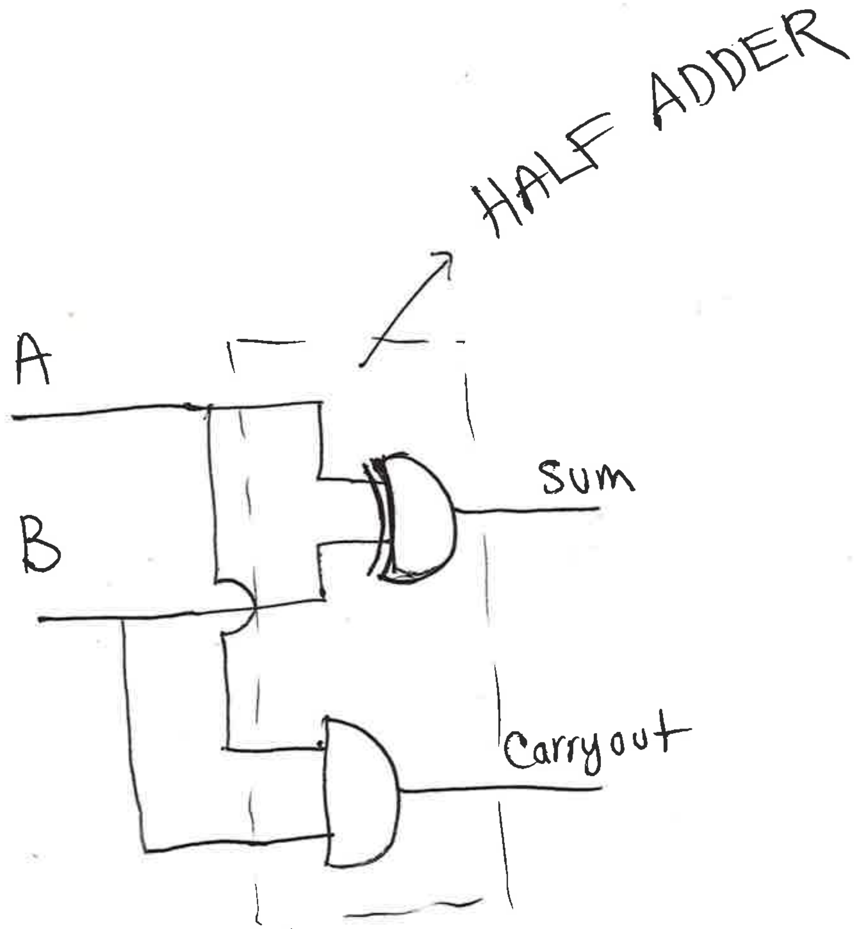
EEE/CSE 120: Adders:

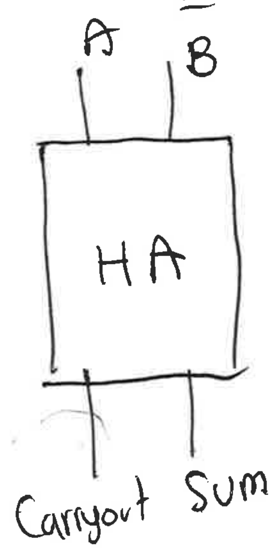
A	B	Sum	Carryout
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

input outputs

XOR AND

"Sum ⇔ XOR"





3 bit binary

Cin ← carryin

Cout = Carry out

A	B	Cin	Sum	Carry out	
0	0	0	0	0	
0	0	1	1	0	
0	1	0	1	0	
0	1	1	0	1	* $\bar{A} B C_{in}$ +
1	0	0	1	0	
1	0	1	0	1	* $A \bar{B} C_{in}$ +
1	1	0	0	1	* $A B \overline{C_{in}}$ +
1	1	1	1	1	* $A B C_{in}$

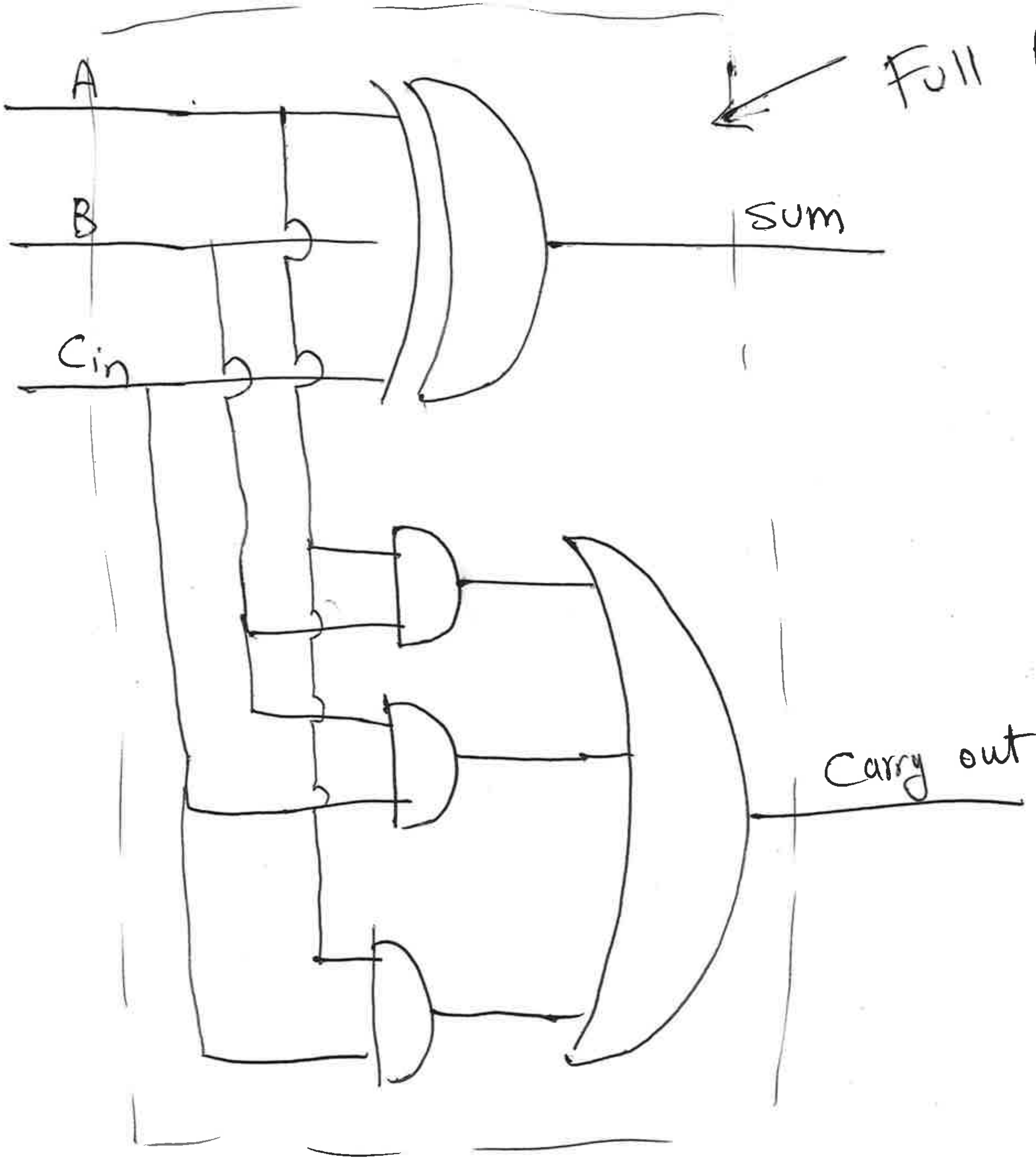
Sum: XOR

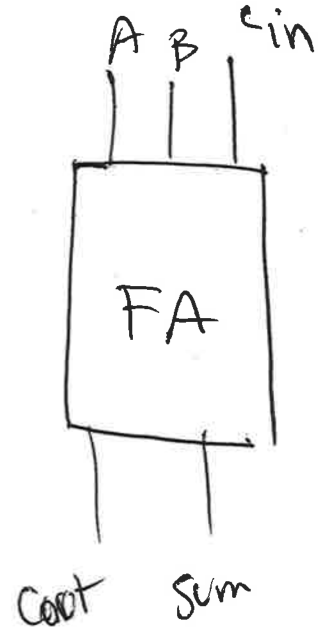
$$C_{out} = \underbrace{\bar{A} B C_{in} + A \bar{B} C_{in}}_{(\bar{A}B + A\bar{B})C_{in} = AC_{in} + BC_{in}} + \underbrace{A B \overline{C_{in}} + A B C_{in}}_{AB[\overline{C_{in}} + C_{in}]}$$

$\overline{C_{in}} + C_{in} = 1$

C_{in}	$\overline{C_{in}}$	$C_{in} + \overline{C_{in}}$
0	1	1
1	0	1

FULL ADDER





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

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

$$\underbrace{(\bar{A} + A)}_{1}B + A\underbrace{(B + \bar{B})}_{1}C_{in} = (A+B)C_{in} = AC_{in} + BC_{in}$$

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

pos.	0	0
Cin	0	0
pos.	1	0
Sum	1	0
Carryout	0	0

OF

Neg	1	1
	1	1
Neg Cin	1	0
	1	0
	1	1

OF

Neg	1	1
pos.	0	0
Cin	1	0
	0	1
	1	0

Cin	Cout	OF
0	0	0
0	1	1
1	0	1
1	1	0

XOR gate

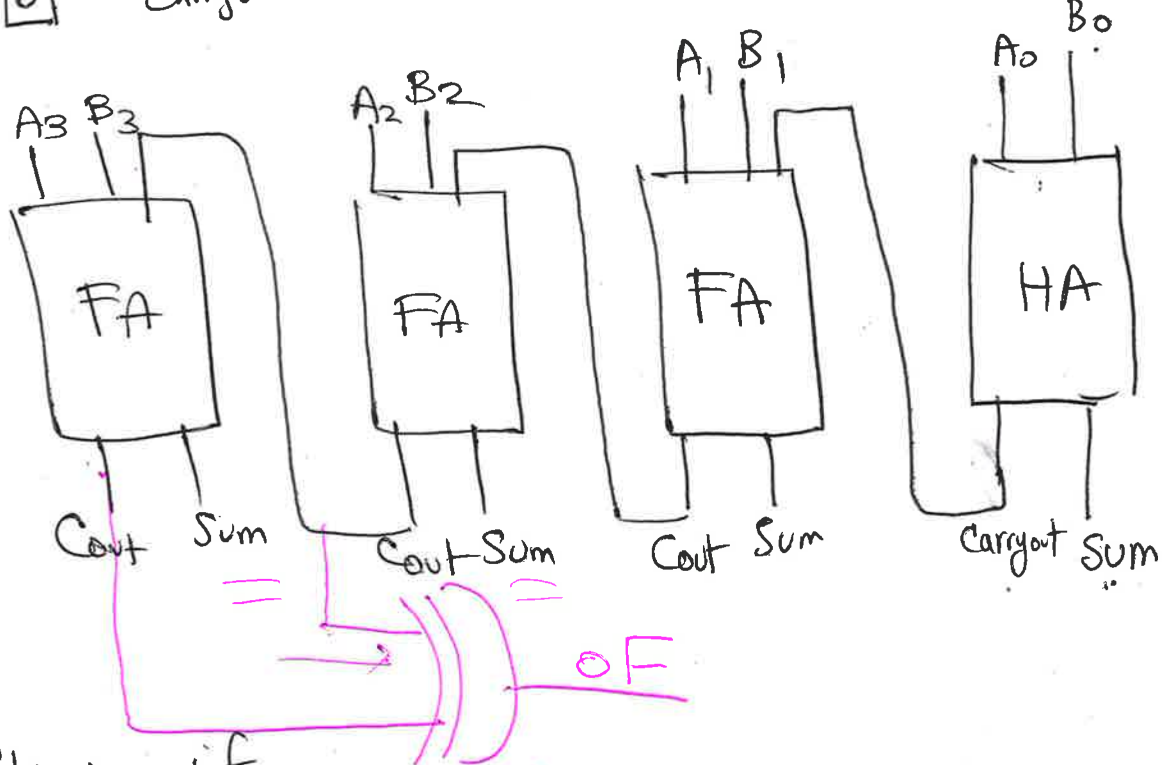
$\boxed{1}$ $\boxed{0}$

0 1 1 0

\+ 1 0 1 1

0 1 Sum

$\boxed{1}$ $\boxed{0}$ carryout



we overflow if

Carry in ≠ Carry out

$$\begin{array}{r} 0\ 1\ 0\ 1 = A = A_3 A_2 A_1 A_0 \\ +\ 1\ 0\ 0\ 1 = B = B_3 B_2 B_1 B_0 \\ \hline \end{array}$$

$A + B$

$A - B$

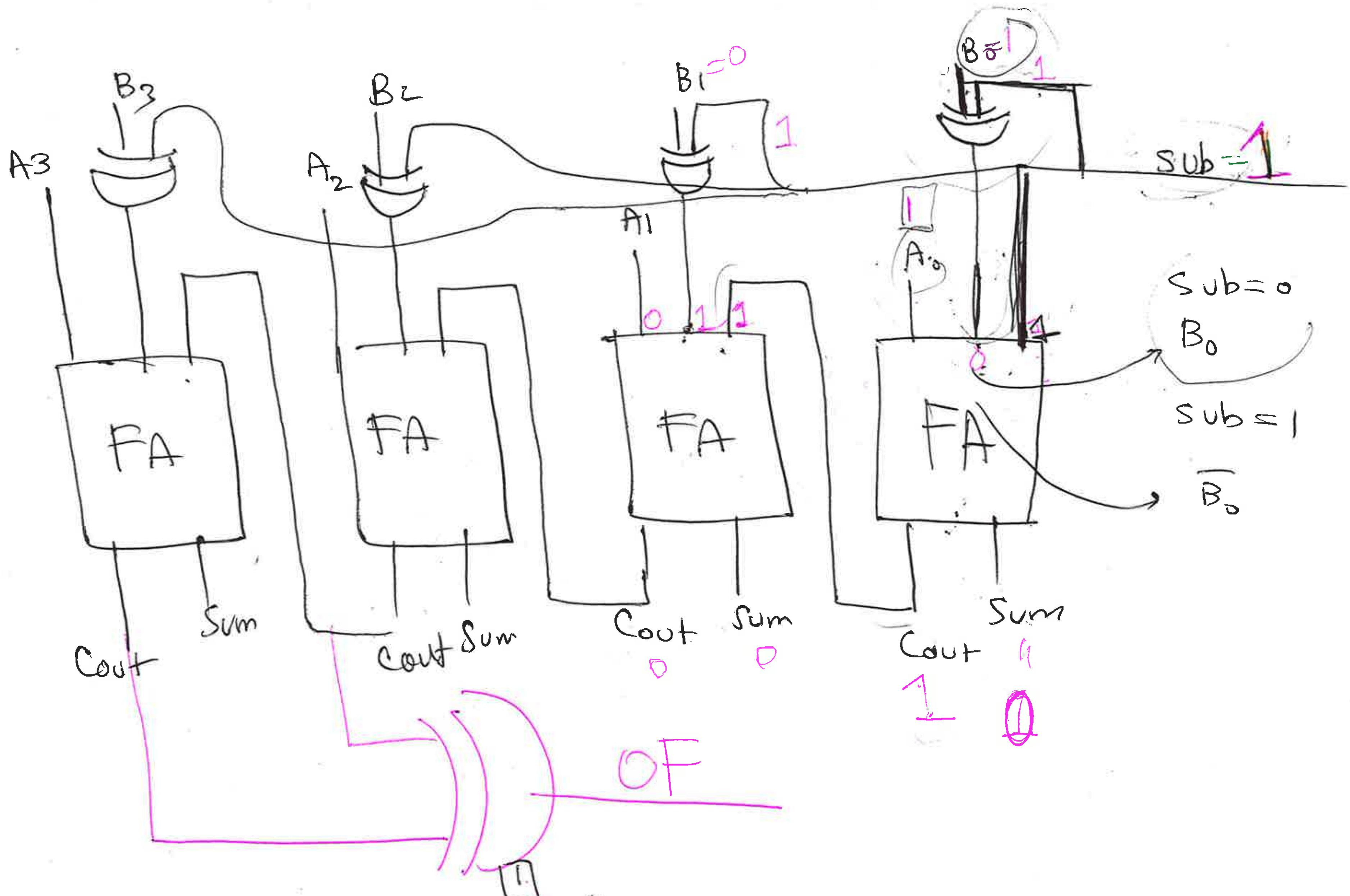
<u>Reminder</u>: 2's complement

(1) Flip all the bits

(2) to add 1 to it

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

$$\text{Subtractor} = \begin{cases} \text{Sub} = 0 \iff \text{we're adding} \\ \text{Sub} = 1 \iff \text{subtracting} \end{cases}$$



A = 0 1 0 1

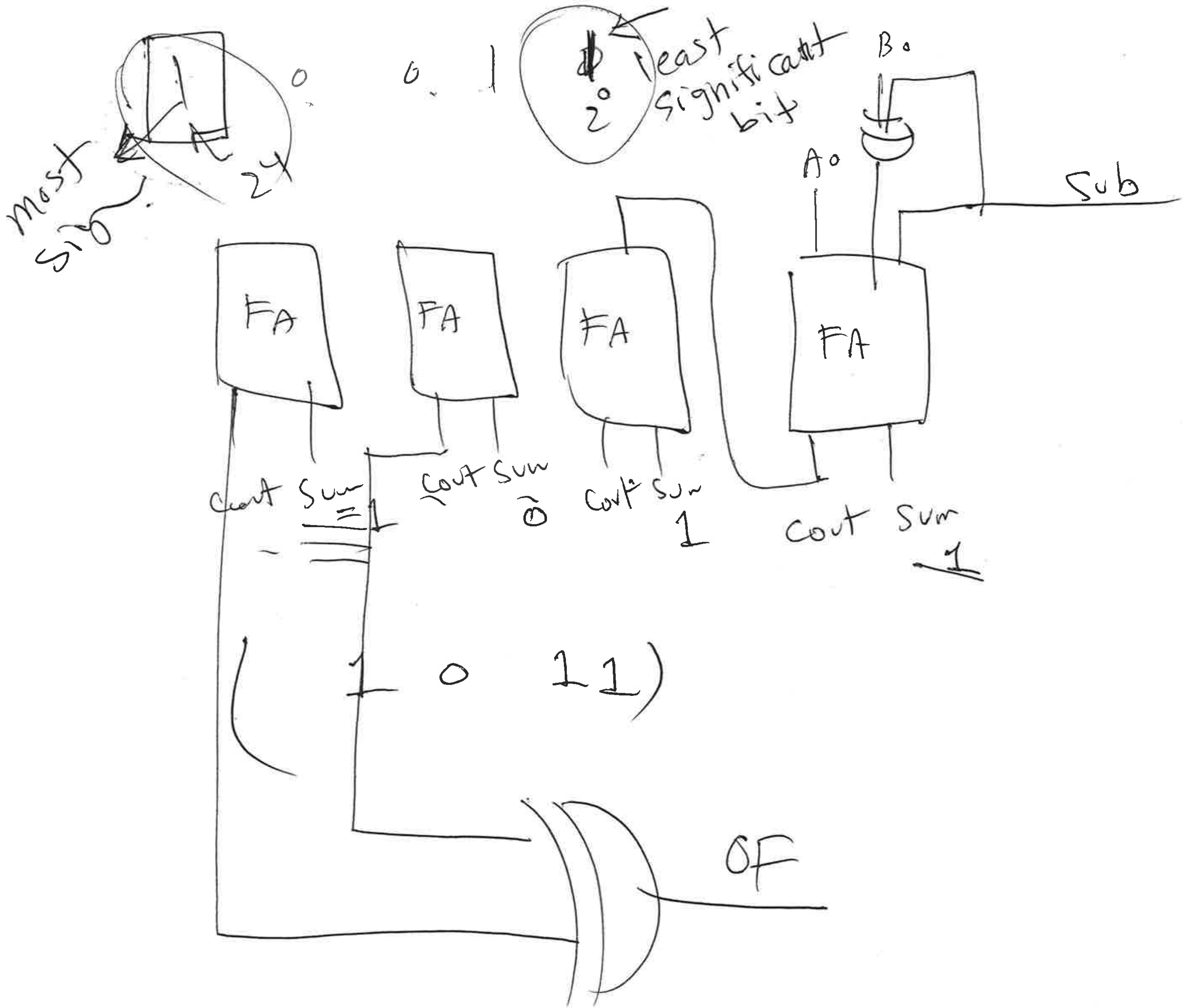
\+ B = 1 0 0 1

0

→ 1's : 0 1 1 0

0 0 0 1

0 1 1 1


most sig
1
2^4
0
0
1
0
least significant bit
2^0
B0
A0
Sub
FA
FA
FA
FA
Cout Sum
=1
Cout Sum
0
Cout Sum
1
Cout Sum
1
(1 0 11)
OF

Lecture 6:

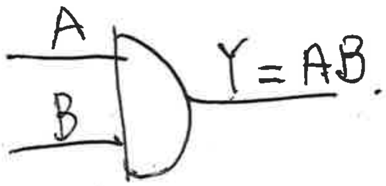
EEE/CSE 120 : Boolean Algebra

EEE/CSE 120 – Quiz 1

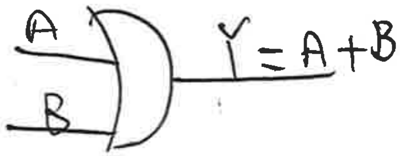
title

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

$1 \oplus 1 = 0$

$1 + 1 = 1$

$5 + 7 = 12$ (closure)

$5 + 0 = 5$

$5 \cdot 1 = 5$

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

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

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

"group"

① Closure :

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

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

$$A \cdot B = A \qquad A + B = A \qquad A \oplus B = A$$

? ? ?

A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

$B=1$

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

$B=0$

<u>EX</u>: prove that

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

A	B	C	$A\cdot(B+C)$	$A\cdot B+A\cdot C$
0	0	0		
	1			
	1			
	1			

Summary:

$$\underbrace{\begin{matrix} A \cdot 0 = 0 \\ A \cdot 1 = A \end{matrix}}_{\text{AND gate}}$$

$$\underbrace{\begin{matrix} A \cdot A = A \\ A \cdot 0 = 0 \end{matrix}}_{\text{AND gate}}$$

$$\underbrace{\begin{matrix} A + 1 = 1 \\ A + A = A \end{matrix}}_{\text{"OR" gate}}$$

$$A + 0 = A$$

$$1 + 1 = 1$$

Example: "OR" gate

line	A	B	A+B
0	0	0	0
1	0	1	1 ← $\bar{A}B$ +
2	1	0	1 ← $A\bar{B}$ +
3	1	1	1 ← AB

$$Y = \text{output} = \underbrace{\bar{A}B + A\bar{B} + AB}_{\text{Sum of products}} = A+B$$

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

min-term

0
1
.
.
.
7
.
.
15

Example: $Y = \underbrace{\bar{A}B + A\bar{B} + AB}_{\text{Canonical form}} = \underbrace{A+B}_{\text{Simplified form}}$

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".

"Cananoical Form"

3. Sum them all.

EEE/CSE 120 - Quiz 1 - resubmit

Example:

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

"SOP"

$$Y = \text{output} = \underbrace{\bar{A}B + A\bar{B}}_{\text{Canonical form}}$$

$$= \underbrace{A \oplus B}_{\text{Simplified.}}$$

Lecture 7

EEE/CSE 120 : Boolean Algebra II

$\boxed{1}$ Closure $(1+1=1\ ,\ 1\oplus 1=0)$

$\boxed{2}$ Associativity

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

$\boxed{3}$ Neutral element, (Identity)

$$AB \overset{?}{=} A \qquad A+B \overset{?}{=} A \qquad A\oplus B \overset{?}{=} A$$

$$B=1 \qquad B=0 \qquad B=0$$

$\boxed{4}$ Inverse element

$$\underbrace{\underbrace{A\cdot B}_{A\bar{A}=0\ \checkmark} \overset{?}{=} 0 \qquad A+B \overset{?}{=} 1 \qquad A\oplus B \overset{?}{=} 1}_{B=\bar{A}}$$

(5) Commutativity

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

$$A + B = B + A$$

(6) Distributivity

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

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

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	$\leftarrow AB$

$$00 \rightarrow 0$$
$$01 \rightarrow 1$$
$$10 \rightarrow 2$$
$$11 \rightarrow 3$$

$$Y = \underbrace{\bar{A}B + A\bar{B} + AB}_{\text{Canonical form}} \overset{?}{=} \underbrace{A + B}_{\text{Simplified form}}$$

$$\boxed{\sum m(1,2,3)}$$

min-term

10

$$Y = \bar{A}B + A\bar{B} + AB = \bar{A}B + A\cdot(\underbrace{\bar{B}+B}_{1})$$ (Distributivity)

$$= \bar{A}B + A\cdot 1$$ (Inverse)

$$= \bar{A}B + A$$ (Identity, Neutral)

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

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

$$= (B+A)$$ (Identity)

$$= \underbrace{A+B}$$ (Commutativity)

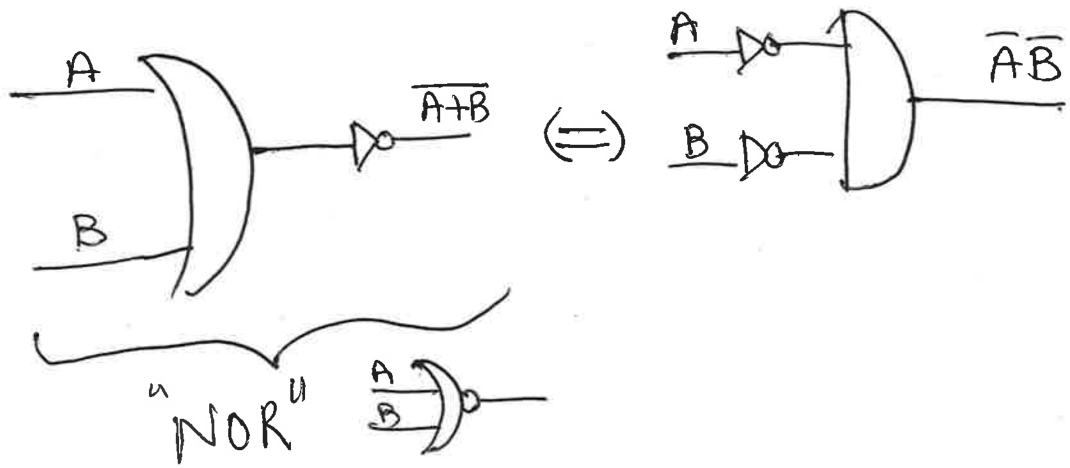
* $A\cdot(B+C) = A\cdot B + A\cdot C$
* $\boxed{A+(B\cdot C) = (A+B)\cdot(A+C)}$

Example:

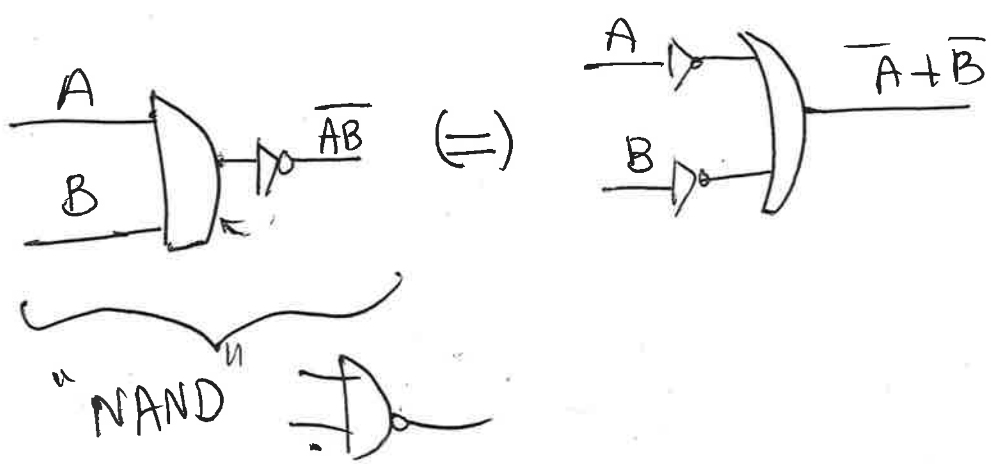
$$
\begin{aligned}
F(a,b,c) &= \bar{a}bc + a\bar{b}c + \underbrace{ab\bar{c} + abc} \\
&= \bar{a}bc + a\bar{b}c + ab(\underbrace{c+\bar{c}}_{1}) \\
&= \bar{a}bc + \underbrace{a\bar{b}c + ab} \\
&= \bar{a}bc + a\cdot(\underbrace{\bar{b}c + b}) \\
&= \bar{a}bc + a\cdot(\underbrace{\bar{b}+b}_{1})\cdot(c+b) \\
&= \bar{a}bc + a\cdot(c+b) \\
&= \underbrace{\bar{a}bc + a\cdot c} + a\cdot b \\
&= \underbrace{(\bar{a}b + a)}\cdot c + a\cdot b \\
&= (\underbrace{\bar{a}+a}_{1})\underbrace{(b+a)}\cdot c + a\cdot b \\
&= b\cdot c + a\cdot c + a\cdot b
\end{aligned}
$$

Demorgan's law :

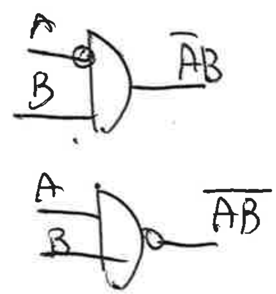
① $\overline{A+B} = \bar{A}\,\bar{B}$



② $\overline{AB} = \bar{A} + \bar{B}$



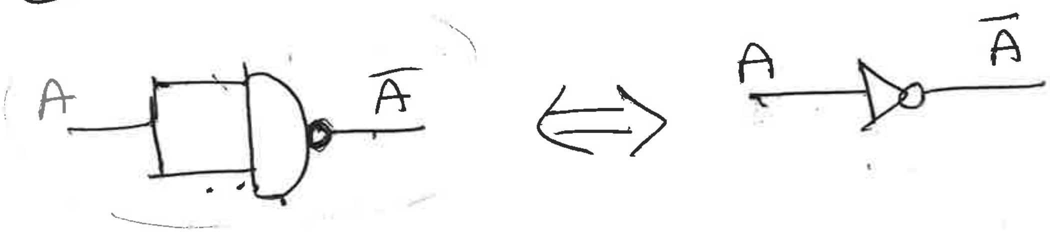
$\bar{A}\bar{B} \neq \overline{AB}$



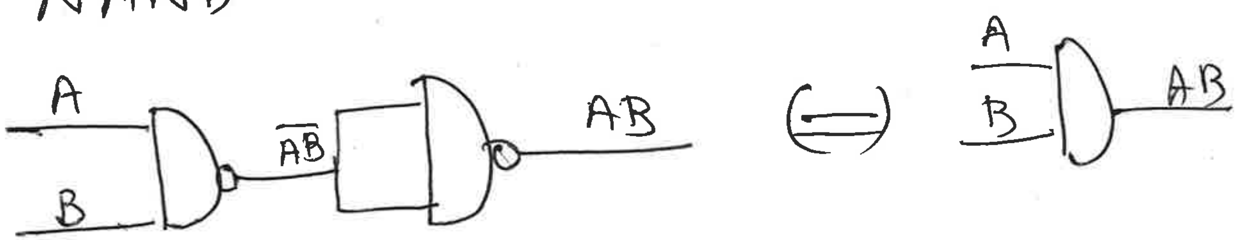
Q: Can we use "NAND" gates or "NOR" gates to build any circuit? YES!

"NAND":

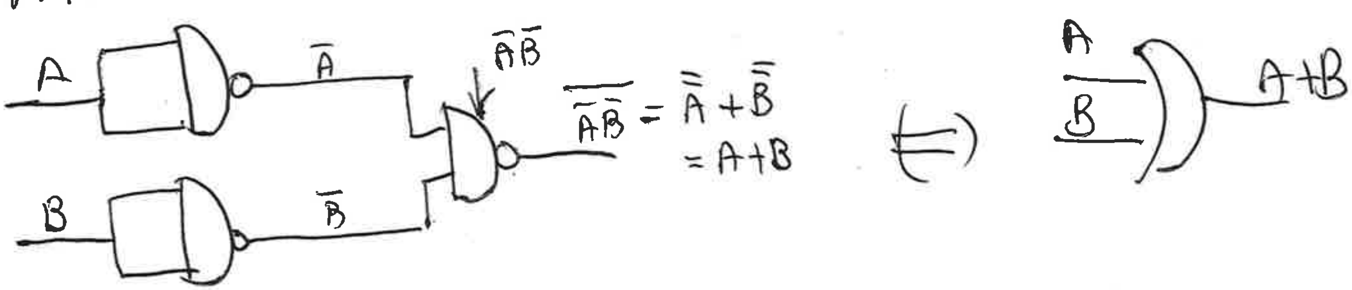
① NAND to build a NOT gate:



② NAND to build AND



③ NAND gate to build OR



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

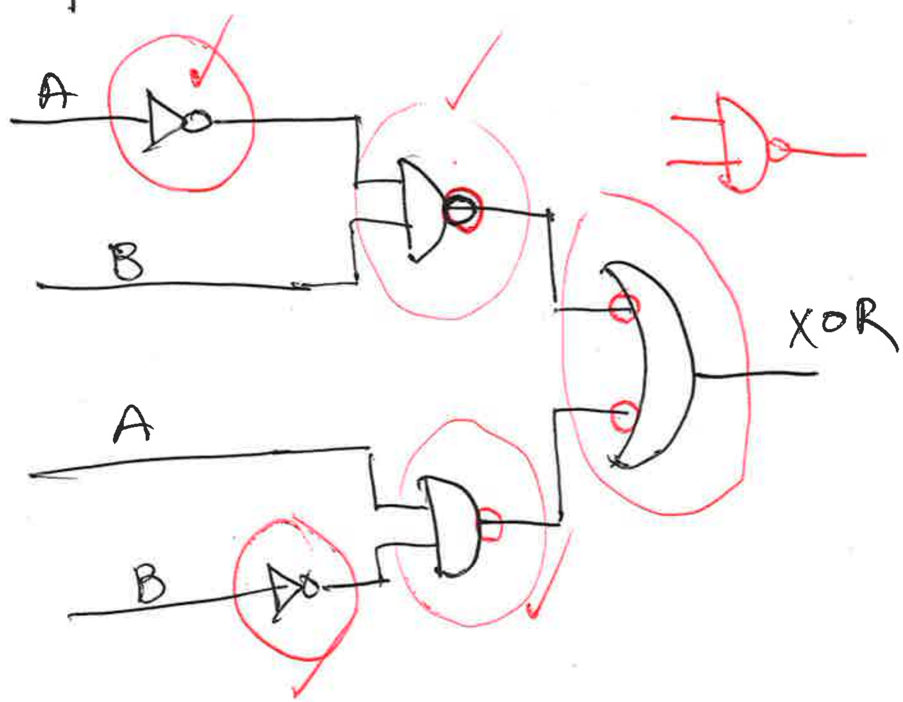
Def: We call a gate "Functionally complete" if we can use that gate to build "AND", "OR", "Not" gates.

Example: NAND is functionally complete.
NOR is functionaly complete.

Example: XOR gate $\longrightarrow$ 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

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

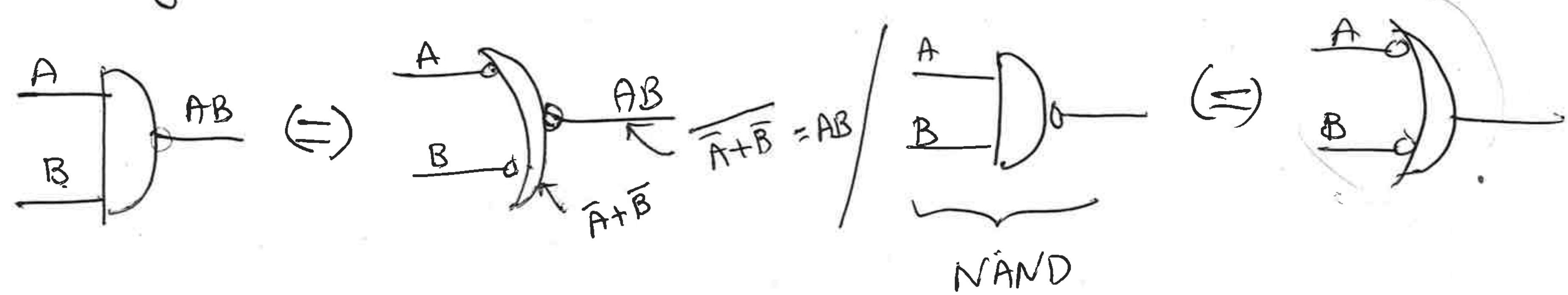


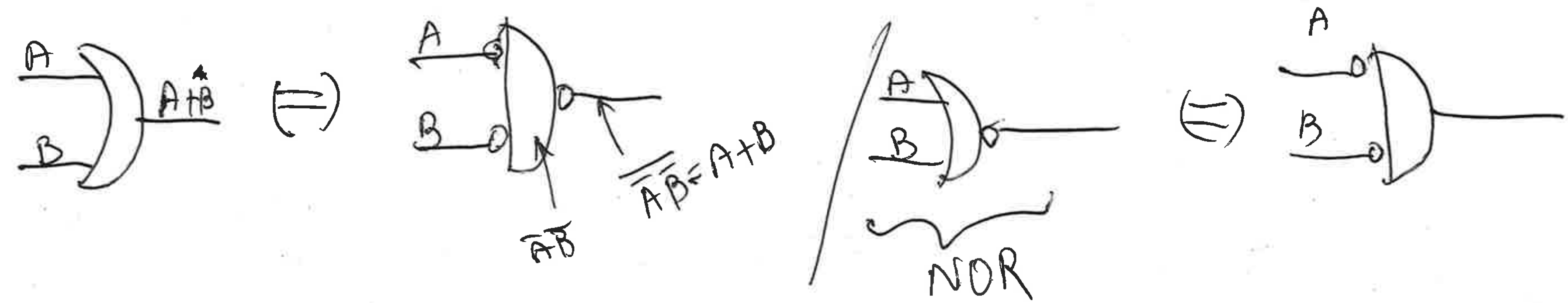
Steps to build a circuit in terms of "NAND" gates or "NOR" gates:

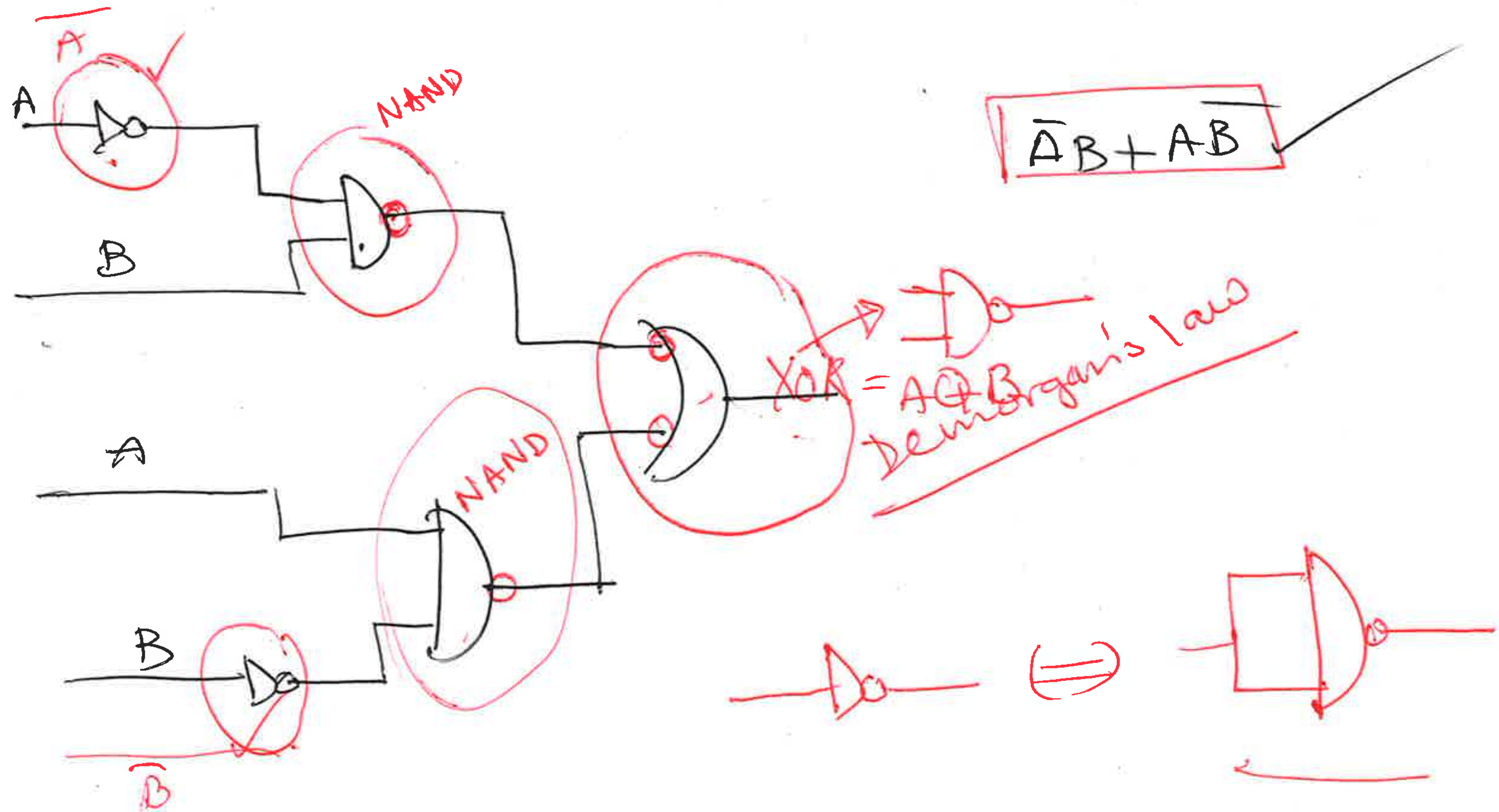
1. Find the equation representing the gate or gates

2. Draw the equation w/o paying attention to the NAND or NOR.

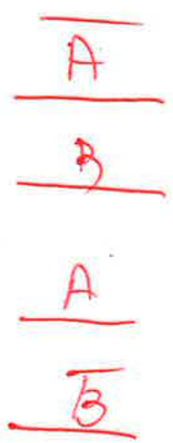
3. Play the bubble game:

a. 

b. 



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



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

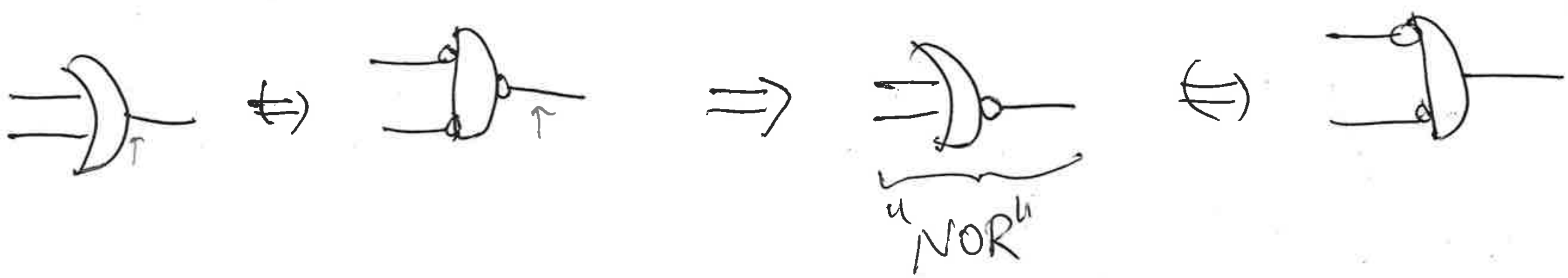
How to build any circuit using NAND / NOR gates!

$\boxed{1}$ write down the function using truth table and draw it "normally"

$\boxed{2}$ play the bubble game

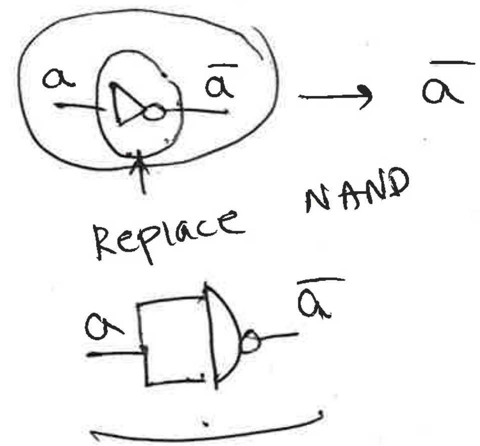
Rule:

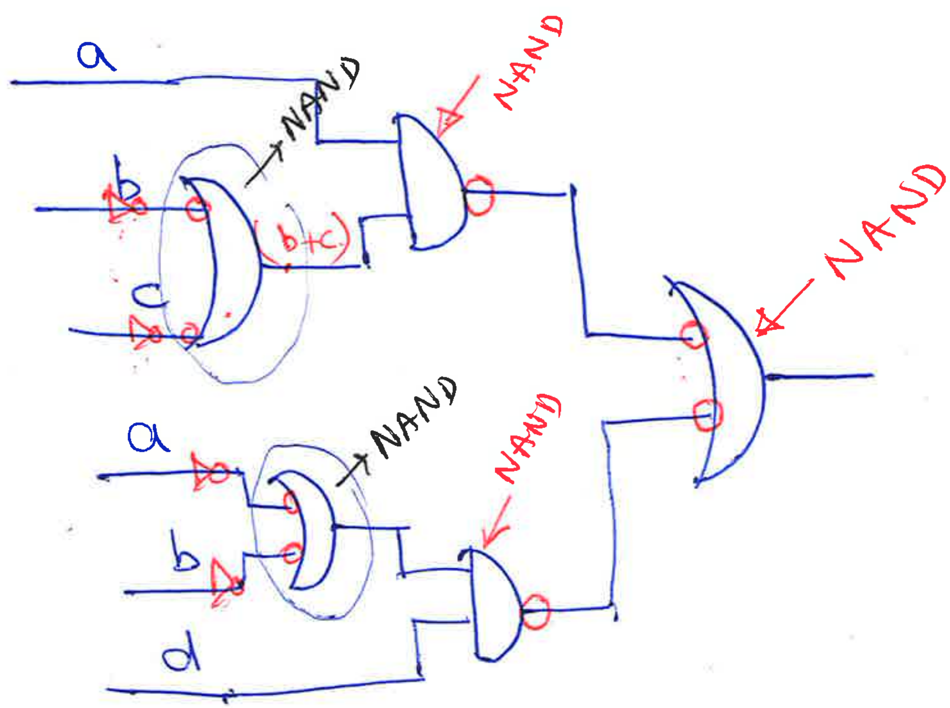
① 

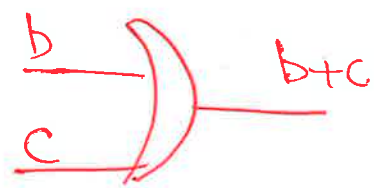
② 

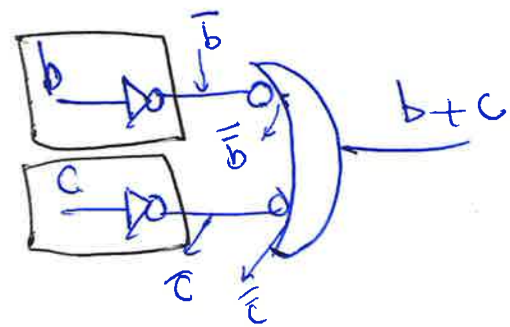
Example: Assume we have access to all inputs as well as their complement, Draw the following circuit using only "NAND" gates!

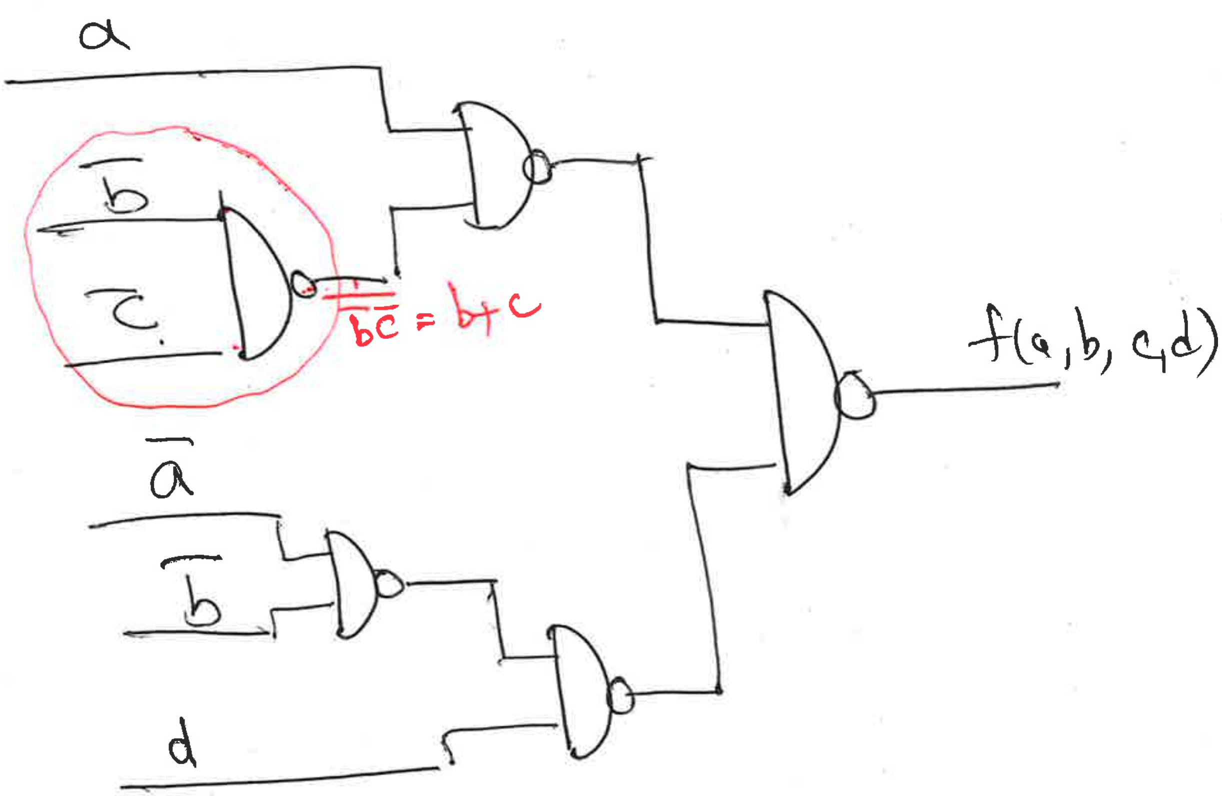
$$f(a,b,c,d) = a(b+c) + d(a+b)$$





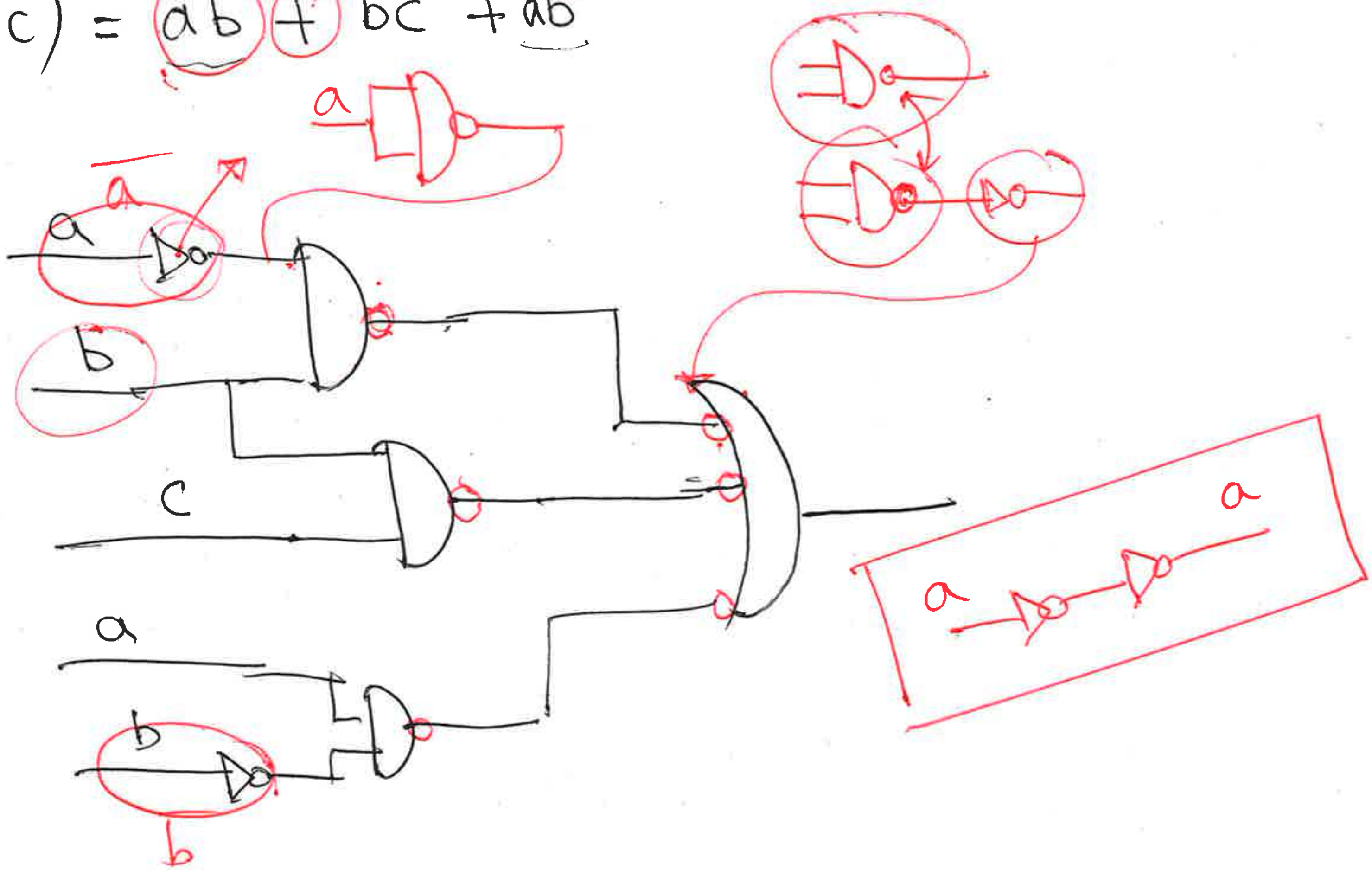


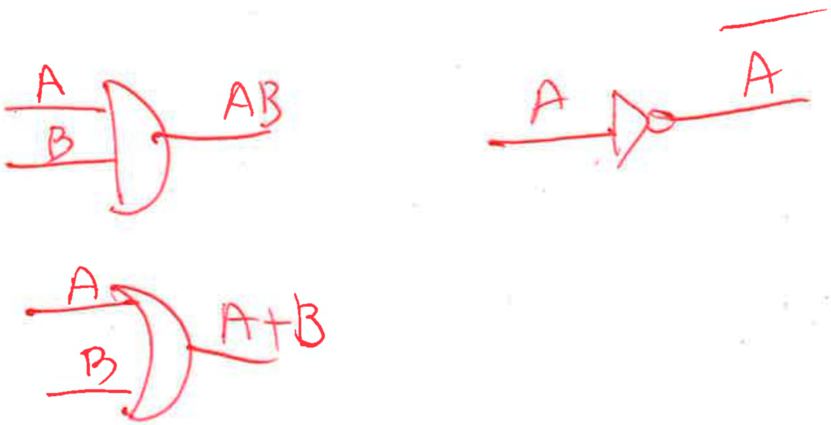




a
b̄
c̄
b̄c̄ = b+c
ā
b̄
d
f(a, b, c, d)

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





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

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.

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

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

Example:

output

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

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

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

POS ?

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

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

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

001 100

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

Max-term

Max-term repr.

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

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

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

$\underbrace{(1+1)}_{1} + 1 = 1$

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.

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	$0 \leftarrow (a+\bar{b}+c)$
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	$0 \leftarrow (\bar{a}+\bar{b}+\bar{c})$

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

$a=0, b=0, c=0 \rightarrow F = \underbrace{1\times1\times1}_{1} + \underbrace{1\times0}_{0} + \underbrace{1\times0}_{0}$
$= 1$

$F(0,1,1) = \underbrace{1\times1\times1}_{1} + \underbrace{1\times1}_{1} + \underbrace{0\times1}_{0} = 1$

$a=0, b=1, c=1$

$\underbrace{1\times0\times0}_{0} + \underbrace{1\times1}_{1} + \underbrace{0\times1}_{0} = 1$

$a=1, b=0, c=1$

$\overbrace{0\times0\times0}^{0} + \overbrace{0\times1}^{0} + \overbrace{1\times1}^{1} = 1$

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

(b) $Y = (a+\bar{b}+c)(\bar{a}+b+c)(\bar{a}+\bar{b}+c)(\bar{a}+\bar{b}+\bar{c})$ (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
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$AB\bar{C} + ABC$

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

A & B are not changing

changing variable

0	0
0	1
1	0
1	1

0	0
0	1
1	1
1	0

grey code!

Demorgan's law

$$\overline{A+B} = \overline{A}\,\overline{B} \quad \neq \cancel{\overline{A+B}}$$

$$\overline{AB} = \overline{A} + \overline{B} \quad \neq \cancel{\overline{A}\,\overline{B}}$$

$y = f(x) = x^2+2$

$y = f(5)$
$= 5^2+2$
$= 27$

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 Lock down browser for exams.

Example:

line #	A	B	C	Y	
0	0	0	0	1	← $\bar{A}\bar{B}\bar{C}$
1	0	0	1	0	
2	0	1	0	1	← $\bar{A}B\bar{C}$
3	0	1	1	1	← $\bar{A}BC$
4	1	0	0	0	
5	1	0	1	1	← $A\bar{B}C$
6	1	1	0	1	← $AB\bar{C}$
7	1	1	1	1	← ABC

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

A and B are not changing → AB

C is changing variable

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

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

00
01
11
10

K-MAP

2-bit

ABC

C \ AB	00	01	11	10
0	1 (0)	1 (2)	1 (6)	0 (4)
1	0 (1)	1 (3)	1 (7)	1 (5)

$\bar{A}\bar{C}$

B

AC

011

Rules of minimum sum of product:

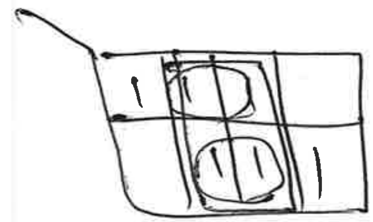
1. Find all the ones on the map and form the K-MAP.

2. group all the ones into groups of 2^n elements (vertically / Horizantally but NOT diagonlly)

groups are called implicants!

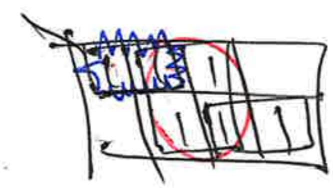
3. Find the largest groups possible

→ prime implicants.

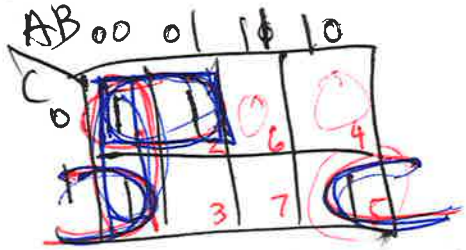


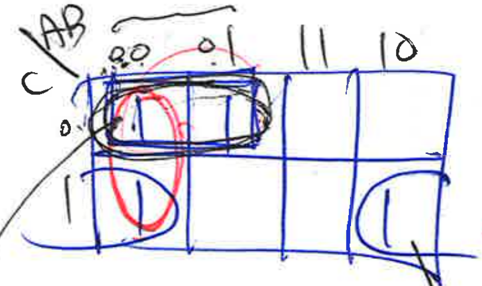
4. Find groups that have at least a single "one" that is NOT shared w/ another group.

→ essential prime implicants



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

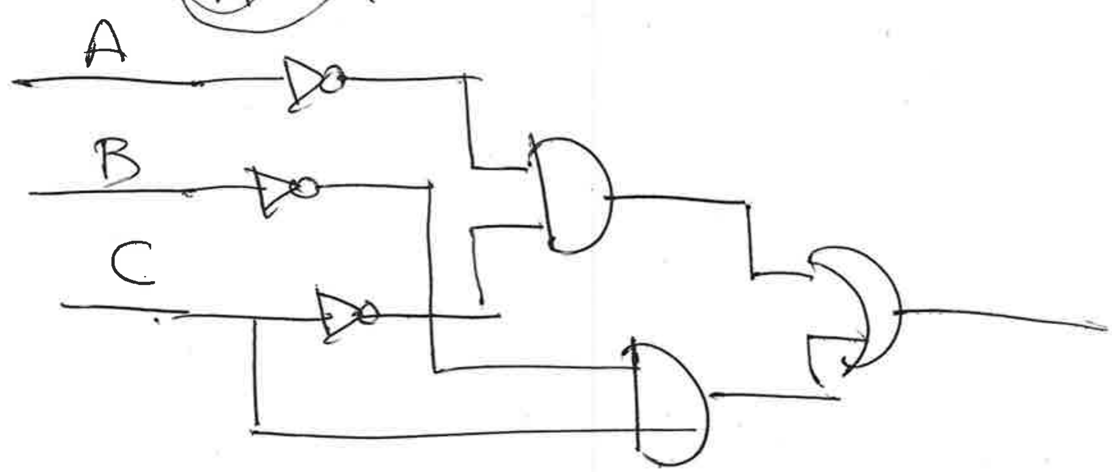




$\bar{A}\bar{C}$

$\bar{B}C$

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



Example: Full Adder

#	A	B	C_{in}	C_{out}	
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	$\leftarrow AB\overline{C_{in}}$
7	1	1	1	1	$\leftarrow ABC_{in}$

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

Q: Simplify Y

1. Boolean Alg.
2. K-MAP.

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

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

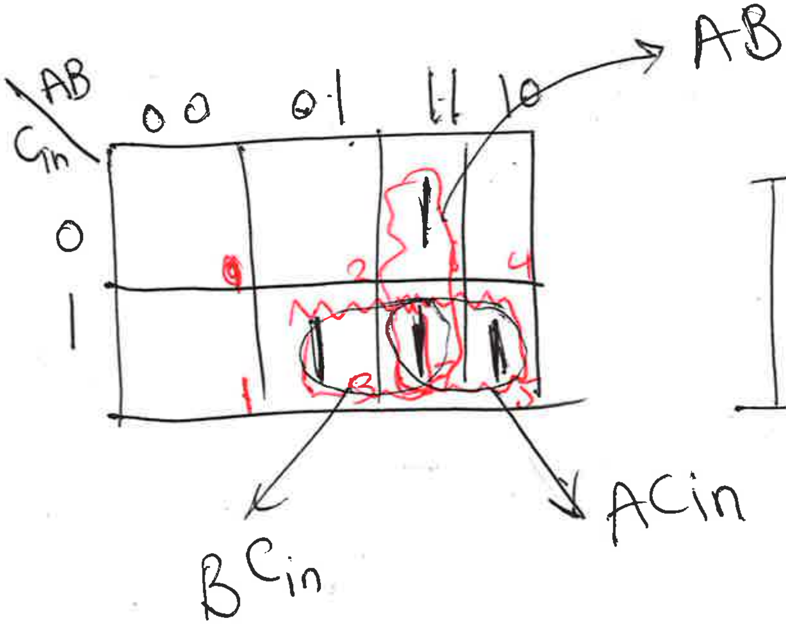
① $Y = \bar{A}BC_{in} + A\bar{B}C_{in} + AB\bar{C}_{in} + ABC_{in} + ABC_{in} + ABC_{in}$

$A + A = A$

$$BC_{in}\underbrace{[\bar{A} + A]}_{1} + AC_{in}\underbrace{[\bar{B} + B]}_{1} + AB\underbrace{[\bar{C}_{in} + C_{in}]}_{1}$$

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

②



$$\boxed{Y = AB + BC_{in} + AC_{in}}$$

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	

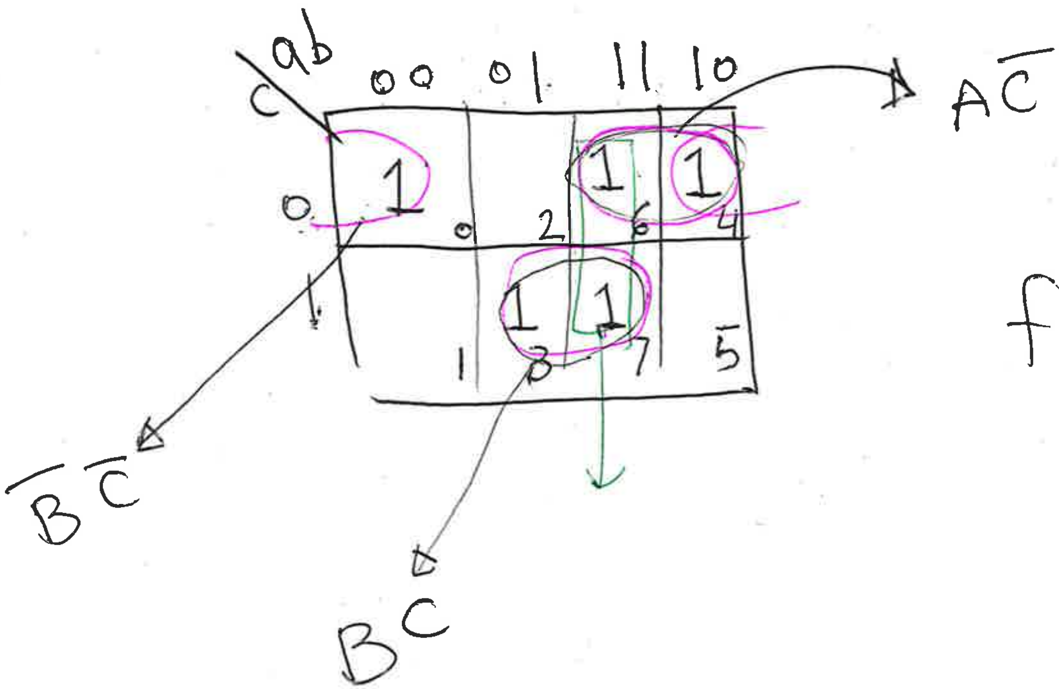
② office hours 9:30 - 10:15 Am (T/TH)

③ Have Quartus installed on your computer.

Example: minimum sum of product of

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

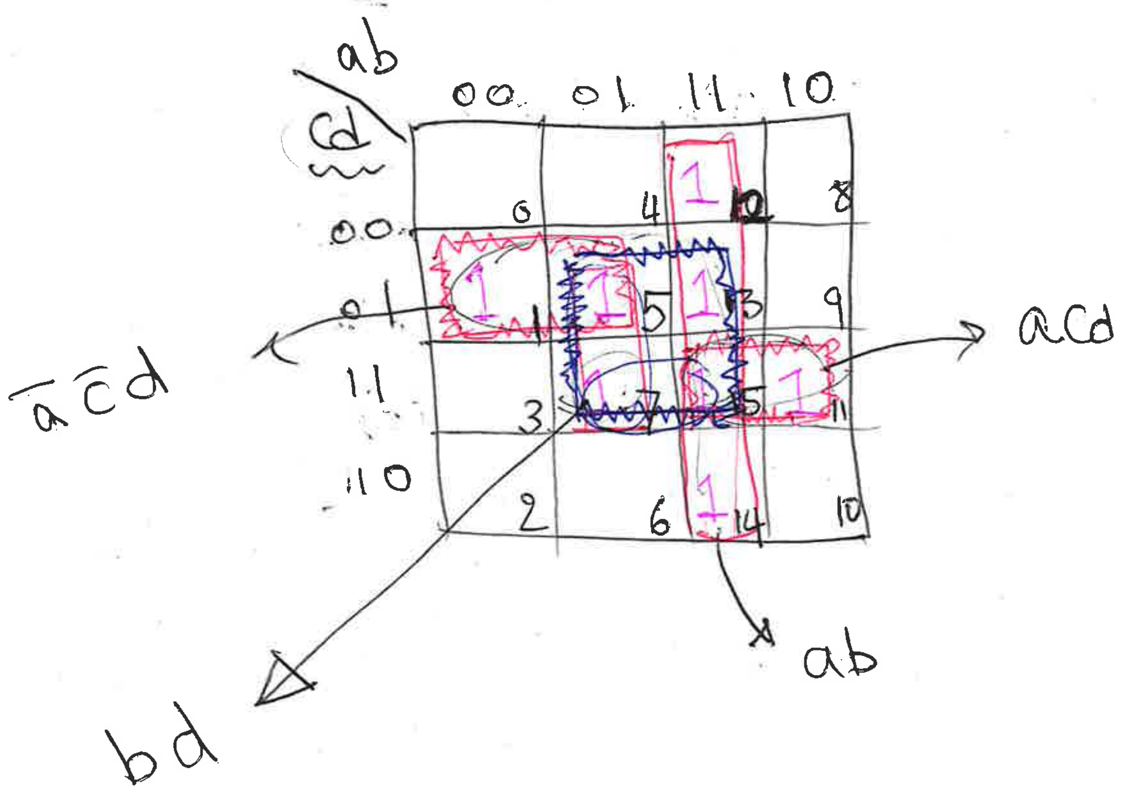
K-MAP → 3-var



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

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

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

$\cancel{b(a+d)}$

sum

$\overline{F} = f$

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

$c \backslash ab$	00	01	11	10
0	1	0	1	1
1	0	1	1	0

$\bar{a} b \bar{c}$

$\bar{b} c$

00
01
11
10

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

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

Demorgan's law:

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

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

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

How to write min Pos :

① group all "zeros" → essential Prime implicants and write the product

② writing the product terms and ignoring the changing variable → sum them up ⇒ $\overline{f}$

③ Use Demorgan's law to get f ⇒ f

Example:

1	2	3
4	5	6
7	8	9
	0	

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

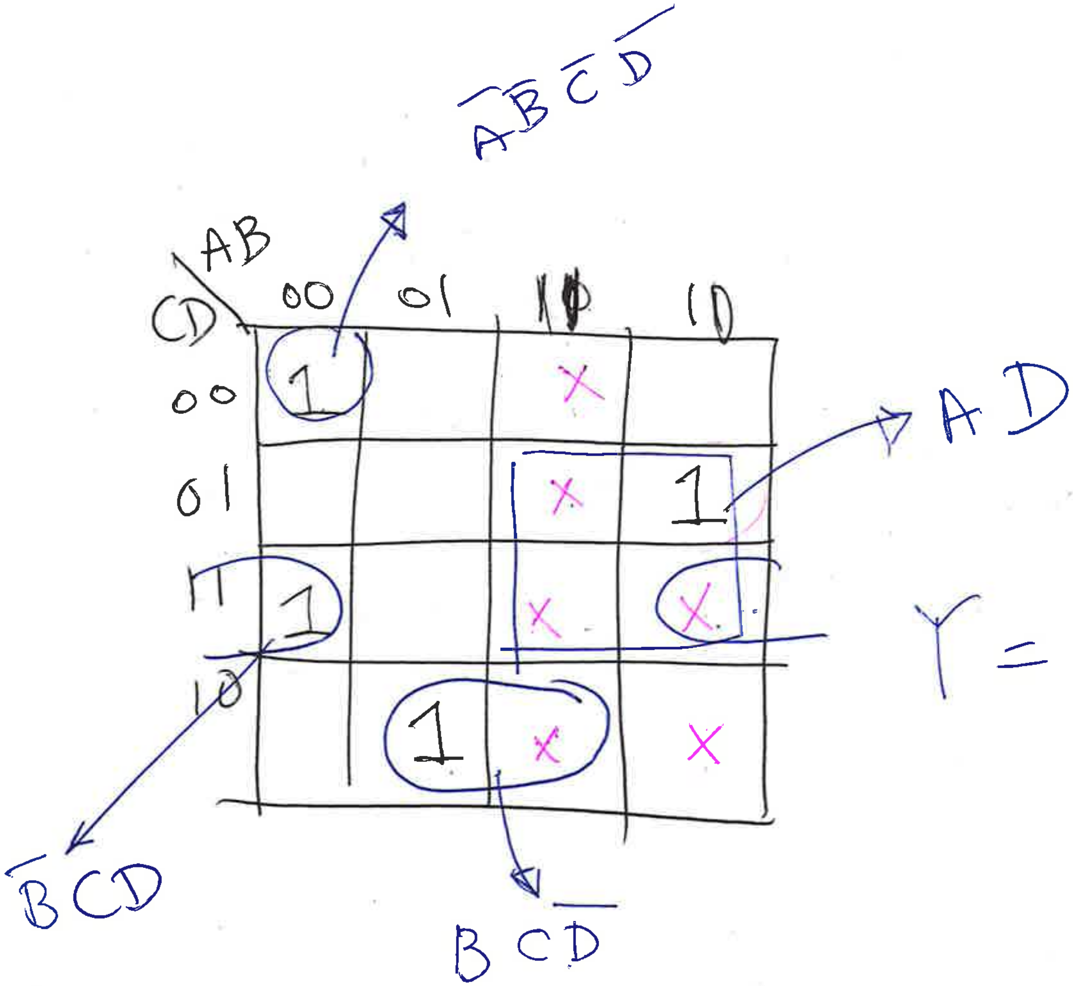
Q: Design a circuit that does that!

#	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

SOP

$$Y = \sum m(0,3,6,9) + \underbrace{\sum d(10,11,12,13,14,15)}_{\text{"don't care"}}$$

"don't care" → can either be "1" or "0"



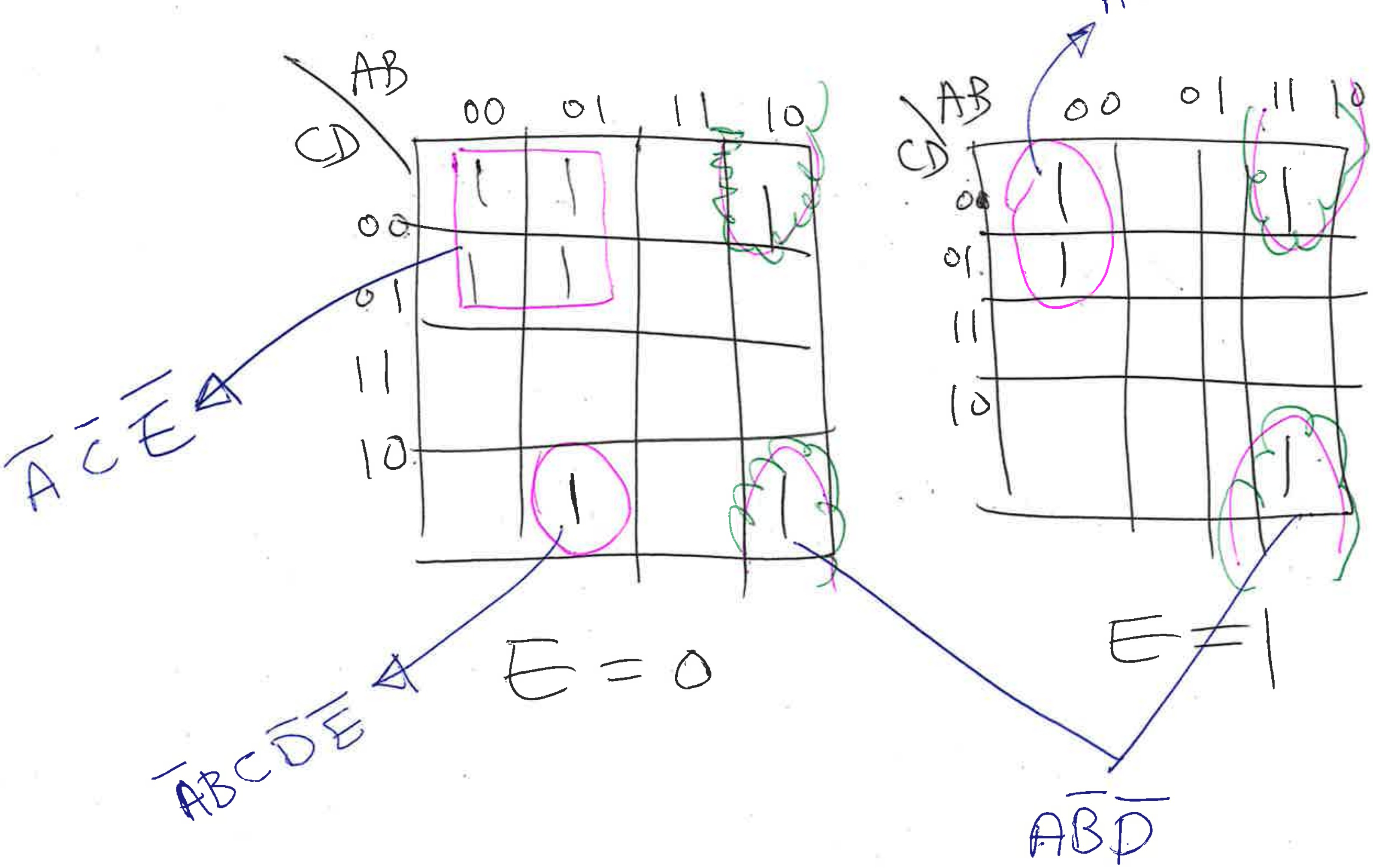
$$Y = AD + \bar{B}CD + BC\bar{D} + \bar{A}\bar{B}\bar{C}\bar{D}$$

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

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

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

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



$$F(A,B,C,D,E) = \bar{A}\bar{C}\bar{E} + \bar{A}BC\bar{D}\bar{E} + \bar{A}\bar{B}\bar{C}E + A\bar{B}\bar{D}$$

Karnough MAP (5 variables):

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

CD \ AB	00	01	11	10
00	1	1		1
01	1	1		
11				
10		1	1	1

E=0

CD \ AB	00	01	11	10
00	1			1
01	1			
11				
10				1

E = 1

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

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

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

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

Annocement:

- 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)

$Y = I_1$

Switch

S → select

I_0	I_1	S	Y = output
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

S \ $I_0 I_1$	00	01	11	10
0			1	1
1		1	1	

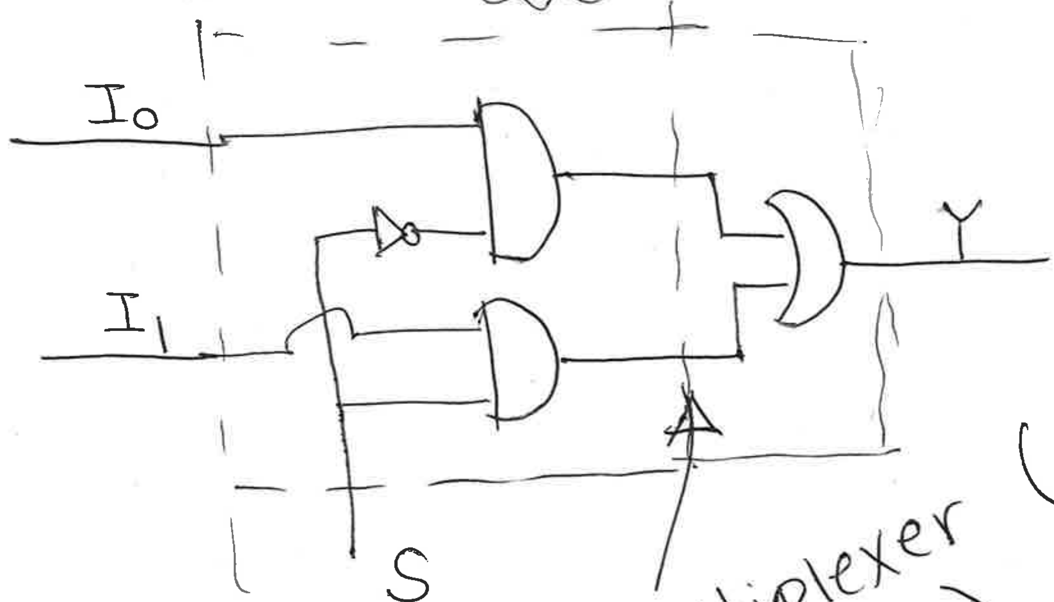
$I_0 \overline{S}$

$I_1 S$

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

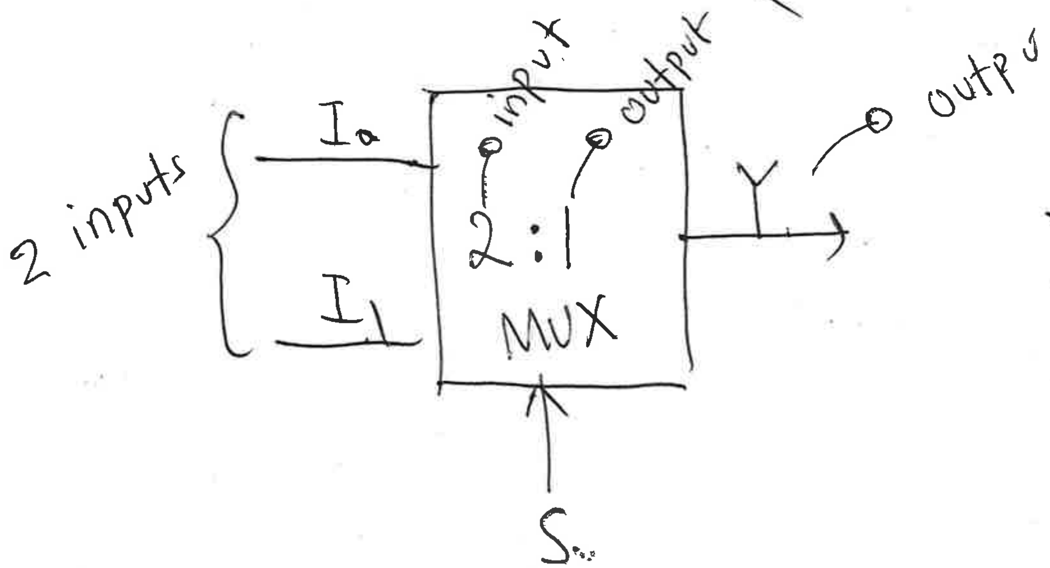
binary number represents the index

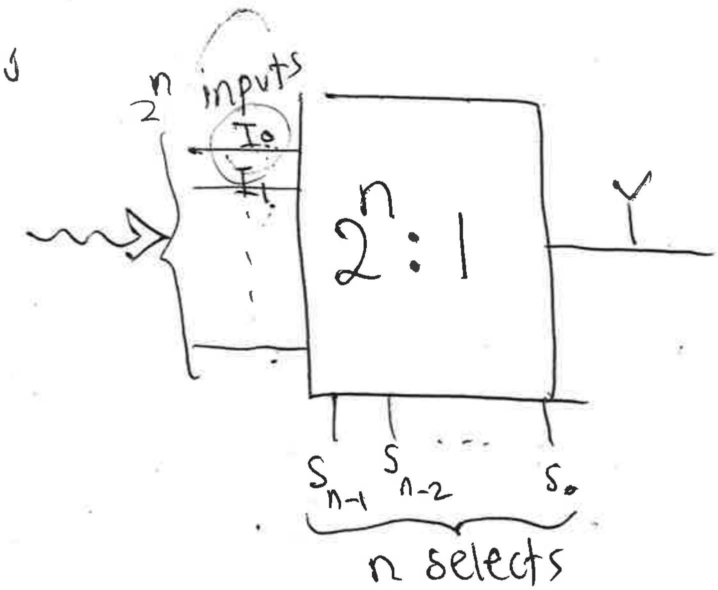
$$Y = I_0 \overline{S} + I_1 S$$



Multiplexer (encoder)

(MUX)





Think of Multiplexers as signal routing devices.

<u>Application</u>: CPU

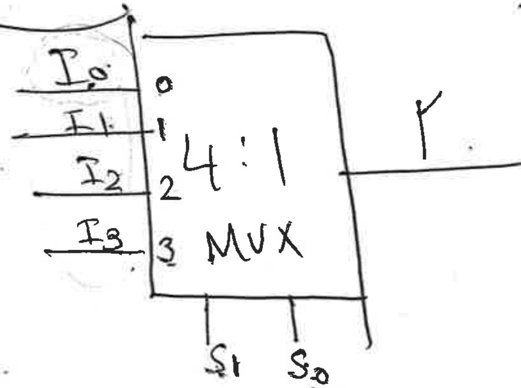
How to build $\underset{2^n}{4}:1$ Mux? (I_0, I_1, I_2, I_3)

2^n ← # of select lines

# line	S_1 (MSB)	S_0 (LSB)	Y = output
0	0	0	I_0
1	0	1	I_1
2	1	0	I_2
3	1	1	I_3

$$Y = I_0 \underbrace{\overline{S_1}\,\overline{S_0}}_{0\,0} + I_1 \underbrace{\overline{S_1}\,S_0}_{0\,1} +$$

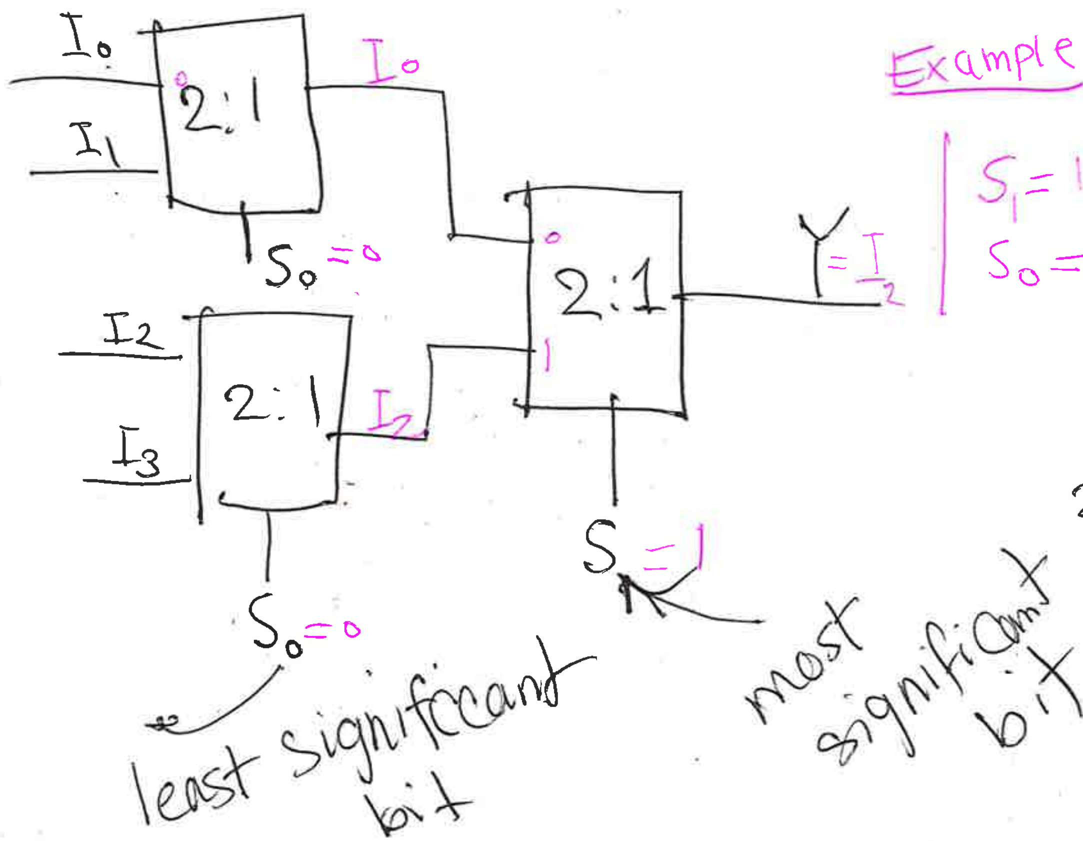
$$I_2 \underbrace{S_1\,\overline{S_0}}_{1\,0} + I_3 \underbrace{S_1\,S_0}_{1\,1}$$



$$f(a,b,c) = \underbrace{\sum m(1,2,4,5) + \sum d(0,6)}_{\text{represent the output}}$$

not input

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



Example:

$S_1 = 1$
$S_0 = 0$ $\rightarrow I_2$

4:1 Mux

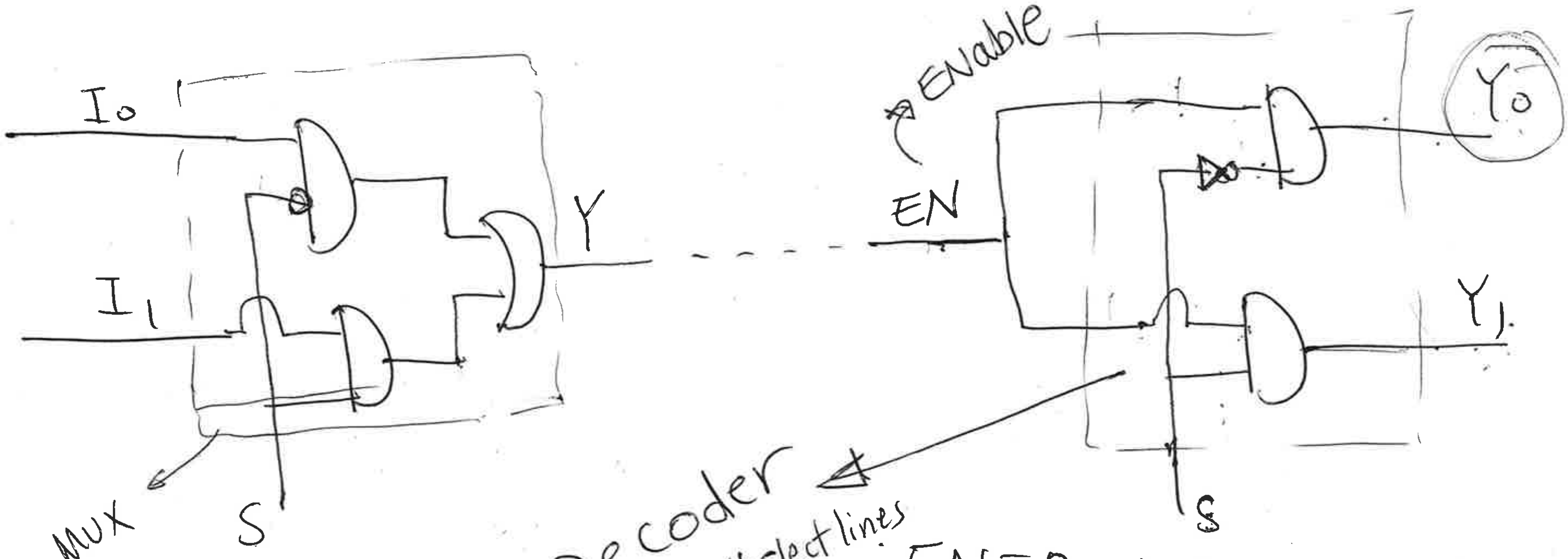
S_1	S_0 (LSB)	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

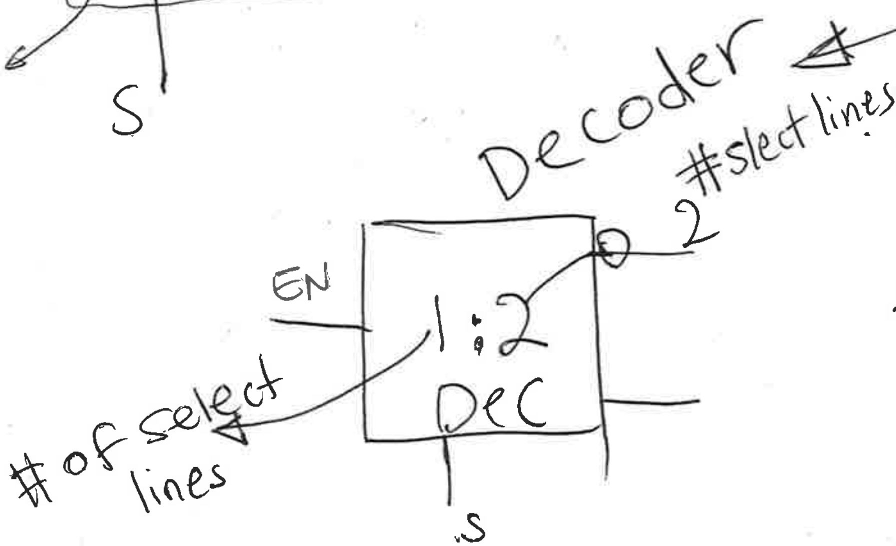
2:1 Mux

S	
0	I_0
1	I_1

Demultiplexers (Decoders)

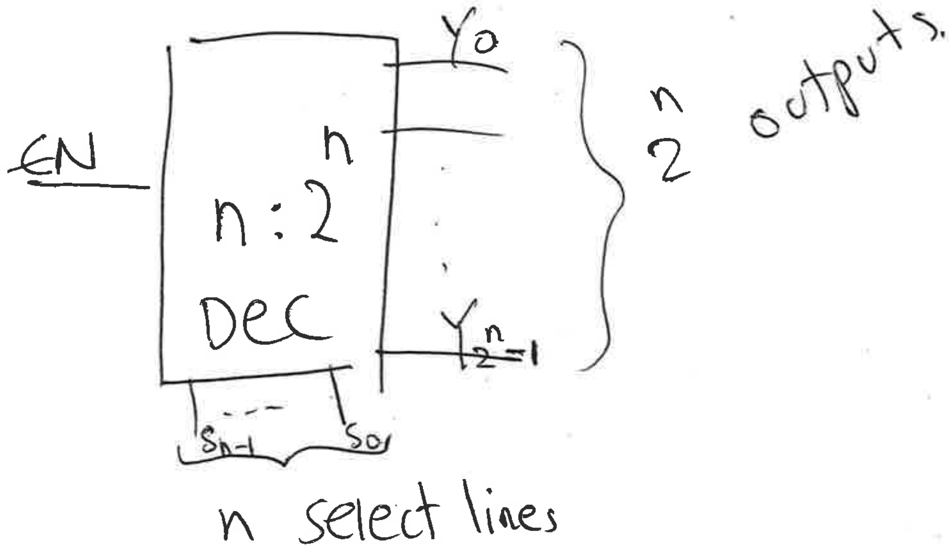
1. one input & multiple outputs
2. select line(s) determine which output is active





EN=0 ⇒ Decoder is not on

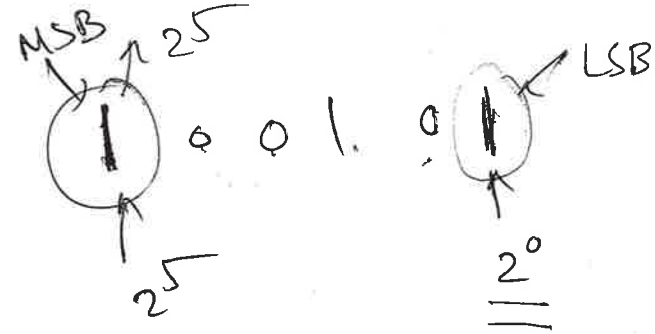
EN=1 ⇒ { S=0 ⇒ Y_0 ; S=1 ⇒ Y_1 }

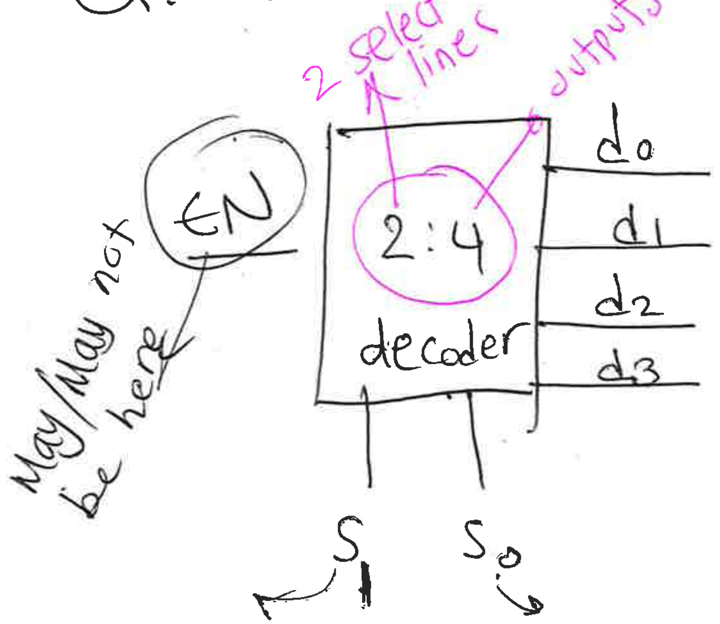


EN=1 says, we generate output.

EN=0 $\Longrightarrow$ output = 0

Q: How to build 2:4 decoders?



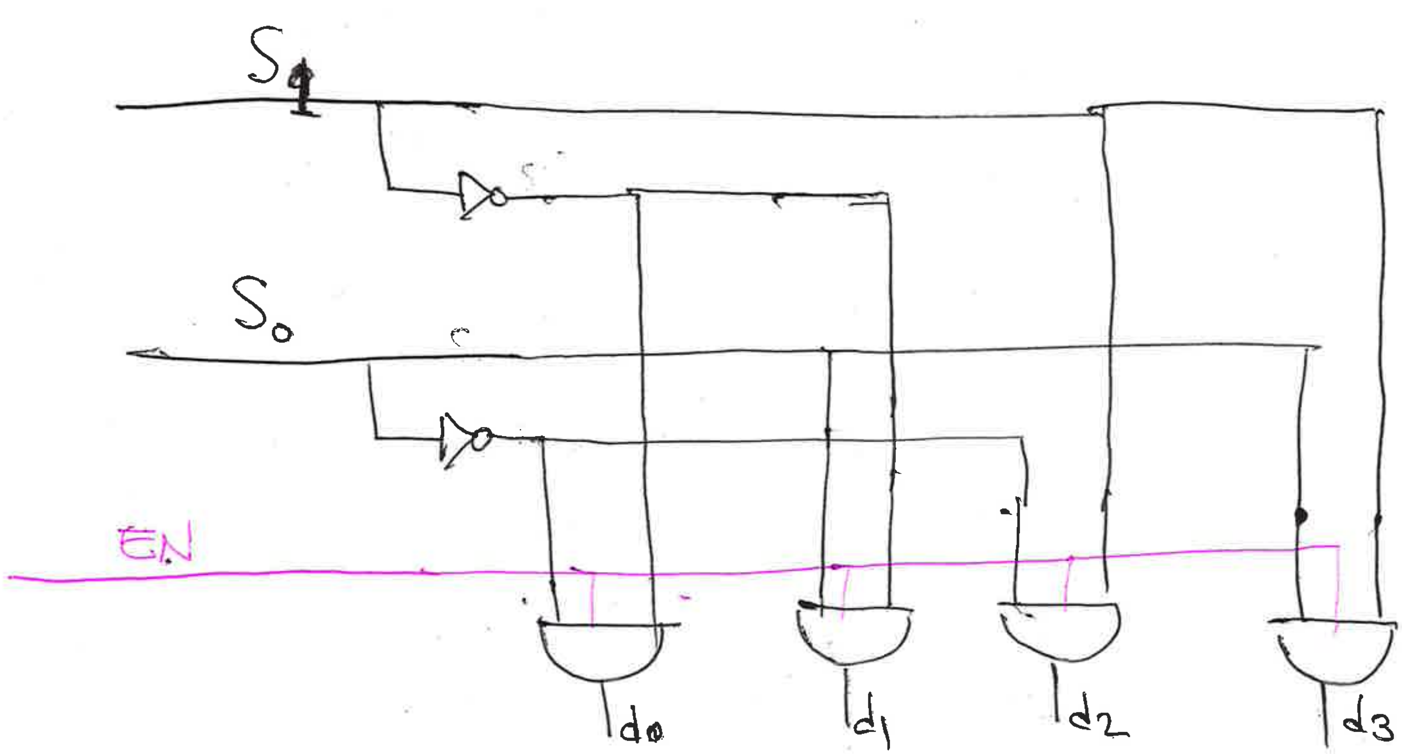


$d_0 \leftrightarrow \overline{S_1}\,\overline{S_0}$

$d_1 \leftrightarrow \overline{S_1}\, S_0$

$d_2 \leftrightarrow S_1\, \overline{S_0}$

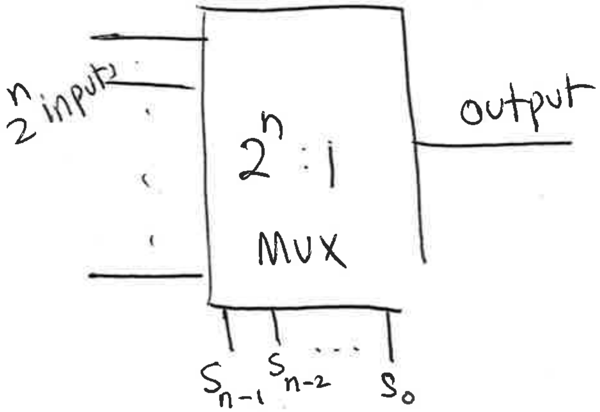
$d_3 \leftrightarrow S_1\, S_0$

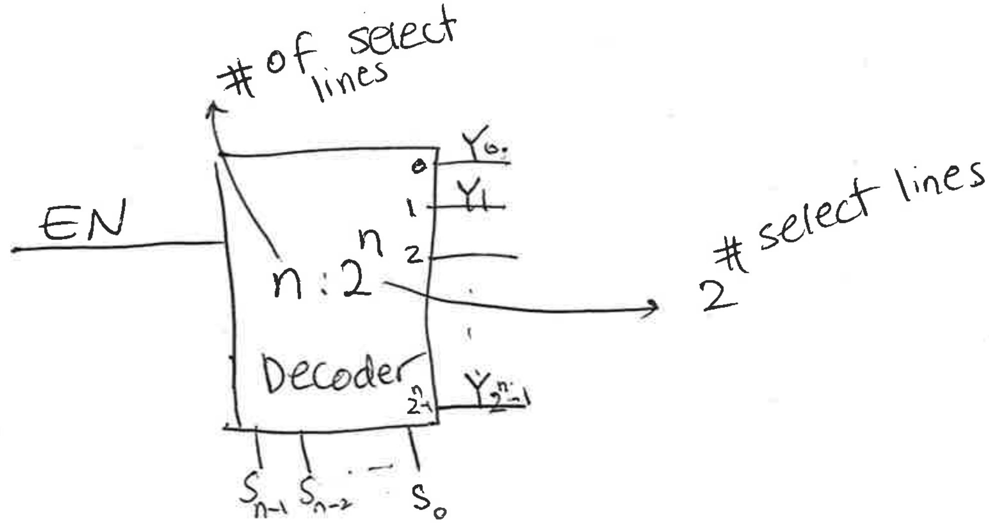


Sep 29, 2020

Lecture 12: What are the applications of Muxes & Decoders?

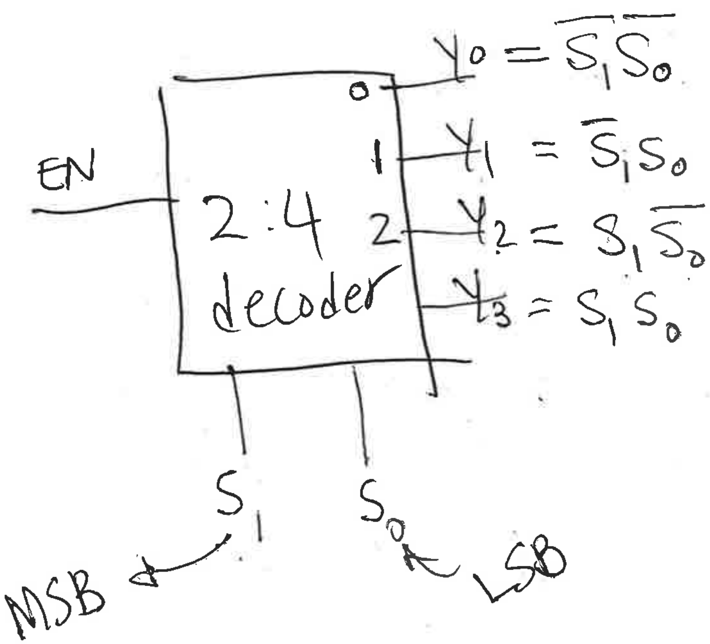
- HW 3 is due oct 1.
- Short Quiz 2 is on oct 1.
- Lab 1 due is extended.

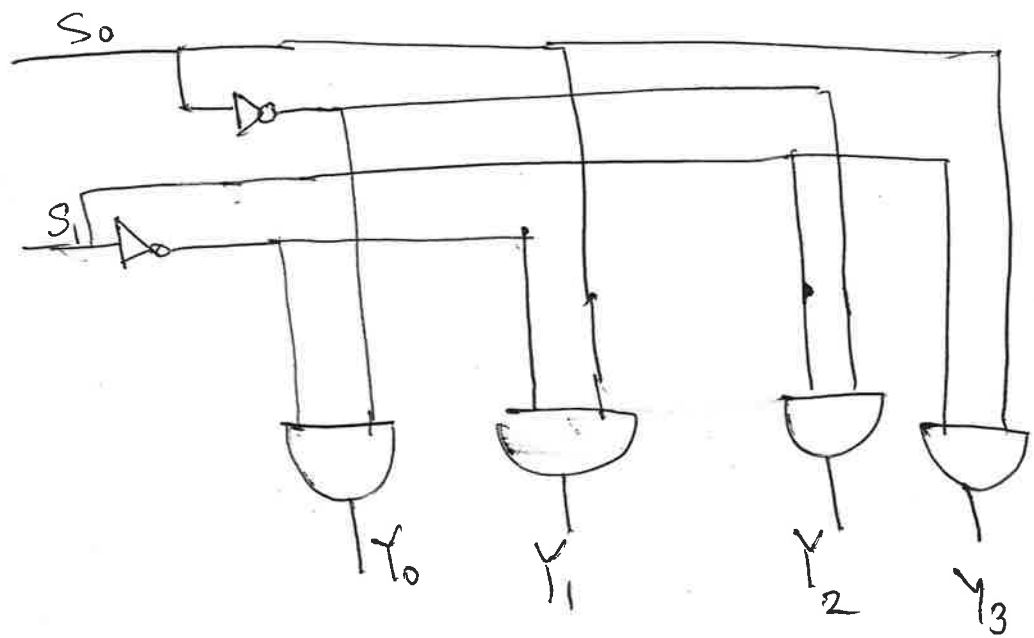




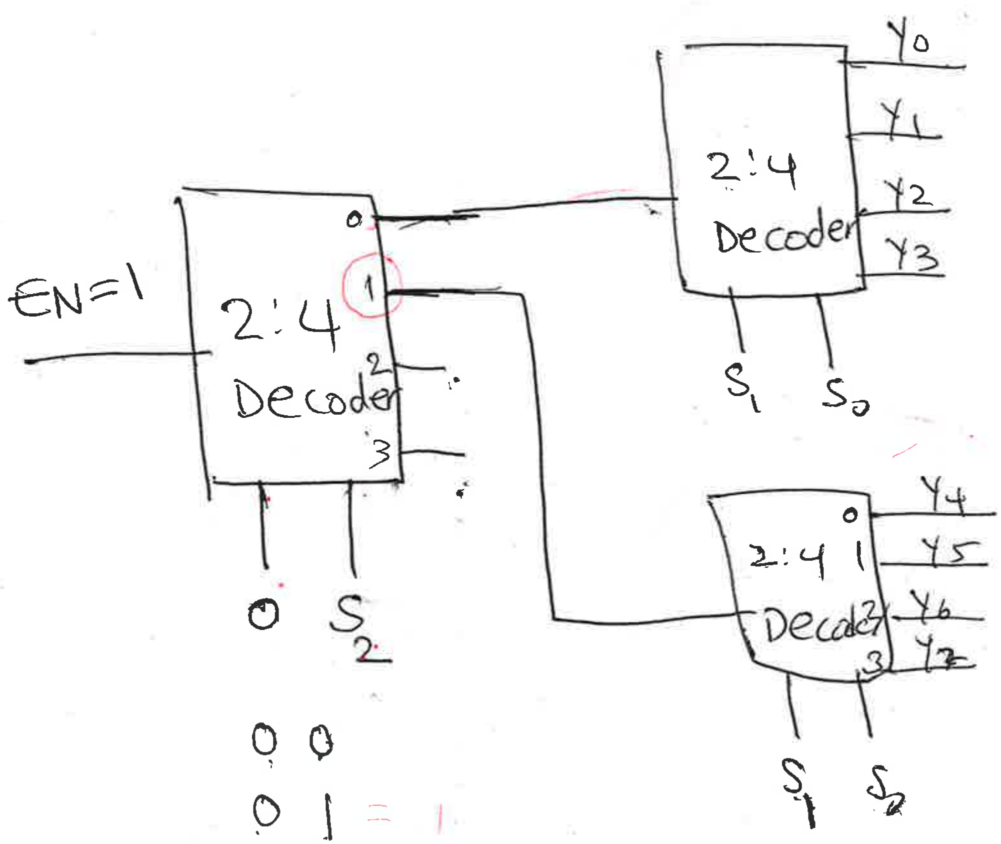
2^n inputs $\Longleftrightarrow$ only one output

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

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

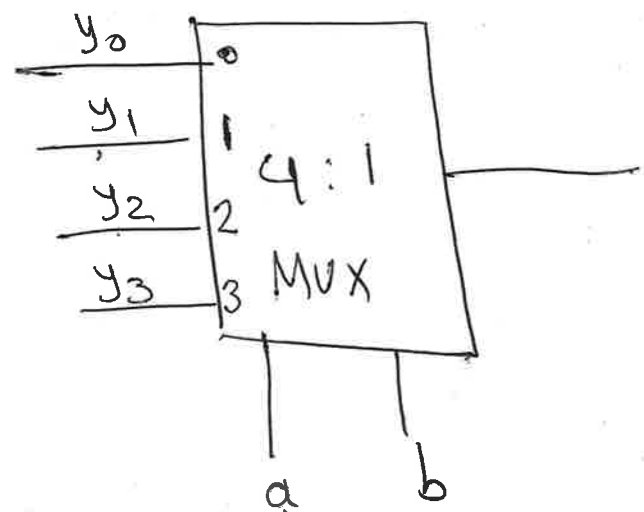
$$\underbrace{0S_2} = S_2$$

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

Example:

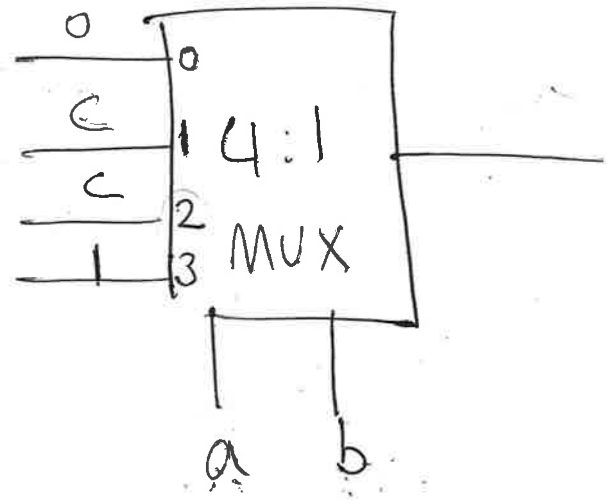
a	b	Y=output
0	0	y_0
0	1	y_1
1	0	y_2
1	1	y_3

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



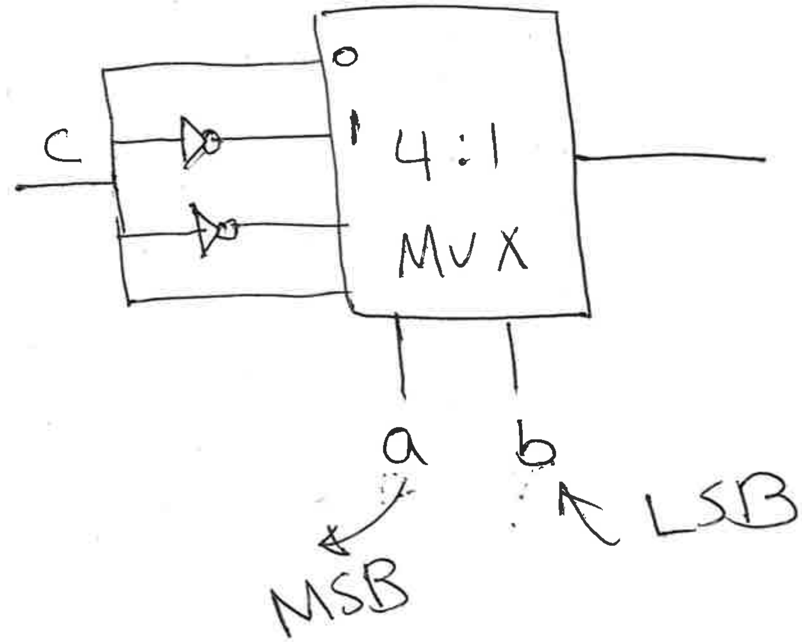
Example : 3 inputs , output = majority voting ⟹ use 4:1 Mux to build the circuit

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	1
1	1	1	1	1

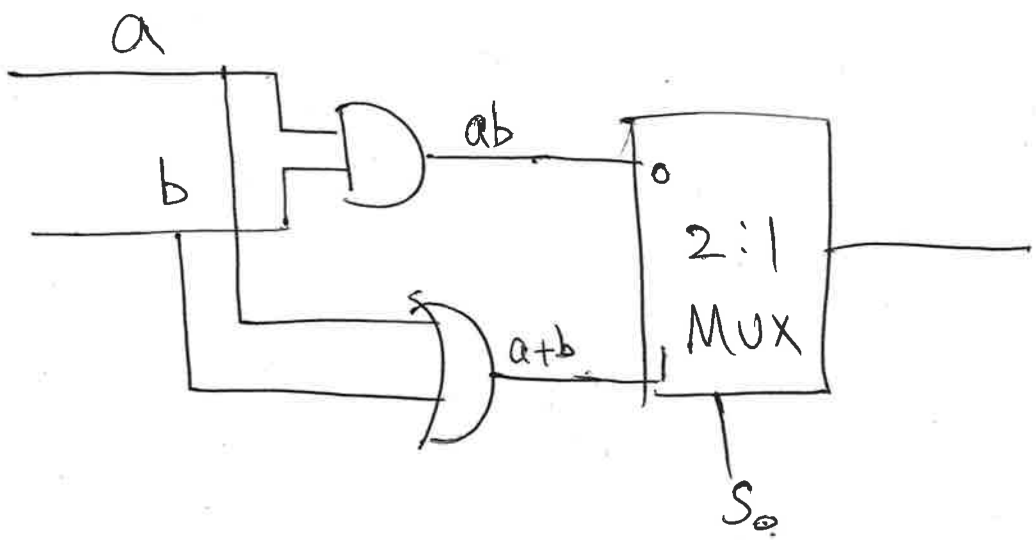


Example: (parity)

a	b	c	Y=output	
0	0	0	0	c
0	0	1	1	c
0	1	0	1	$\overline{c}$
0	1	1	0	$\overline{c}$
1	0	0	1	$\overline{c}$
1	0	1	0	$\overline{c}$
1	1	0	0	c
1	1	1	1	c

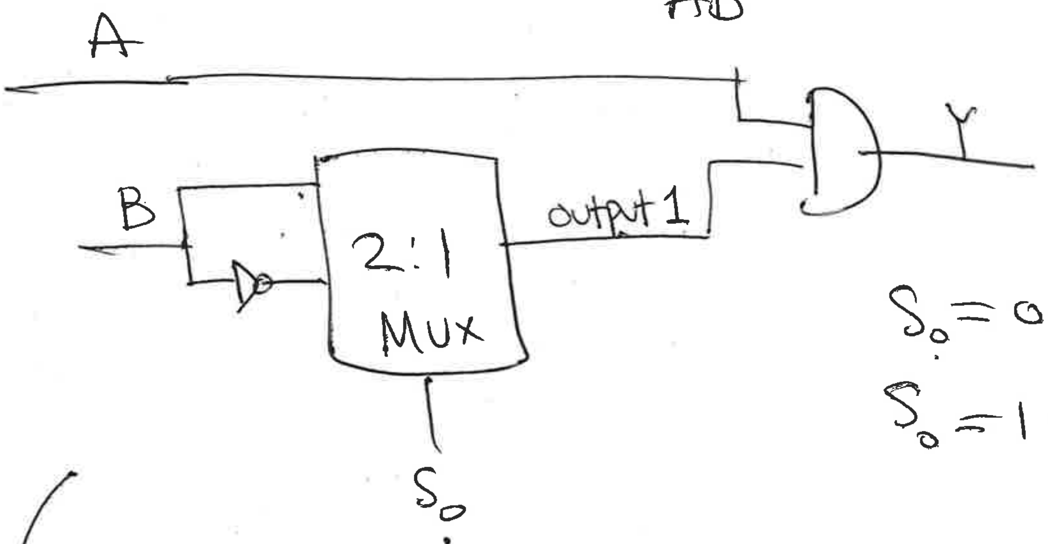


Example : use a mux to build a circuit that allows one of the operations ("AND", "OR") to be chosen and applied to an input.



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

$\underbrace{\text{A and B}}_{AB}$ or $\underbrace{\text{A and not B}}_{A\overline{B}}$



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

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

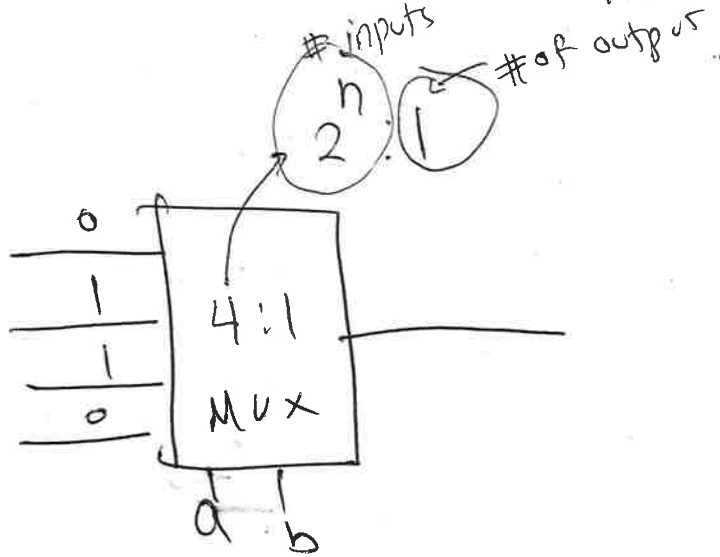
we build ALU

Arithmitic logic unit

Carry out a range of Calculations on its input.

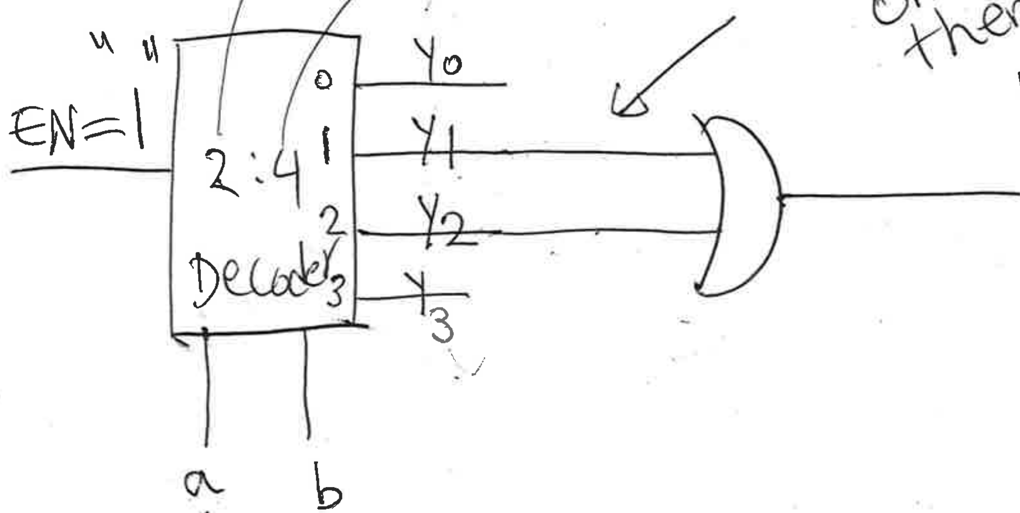
Example : Use 2:4 decoder to build XOR function.

	a	b	Y="XOR"	
0	0	0	0	
1	0	1	1	← $\bar{a}b$
2	1	0	1	← $a\bar{b}$
3	1	1	0	



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

# of select lines
2 # select lines



Oct 1, 2020

Lecture 13 : Programmable logic devices and Tri-state Buffer

programmable logic devices (PLD)

→ Integrated circuits that contain "AND", "OR" gates

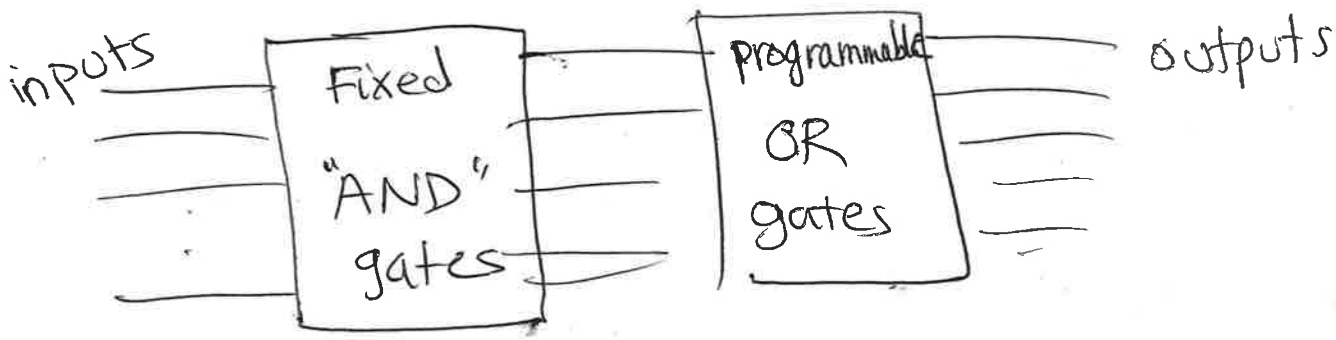
Programming : entring info into these devices is Called Programmig

X : Indicate Programmig

Types of PLD:

1. Programmable read only memory (PROM)
2. Programmable array logic (PAL)
3. Programable logic array (PLA)

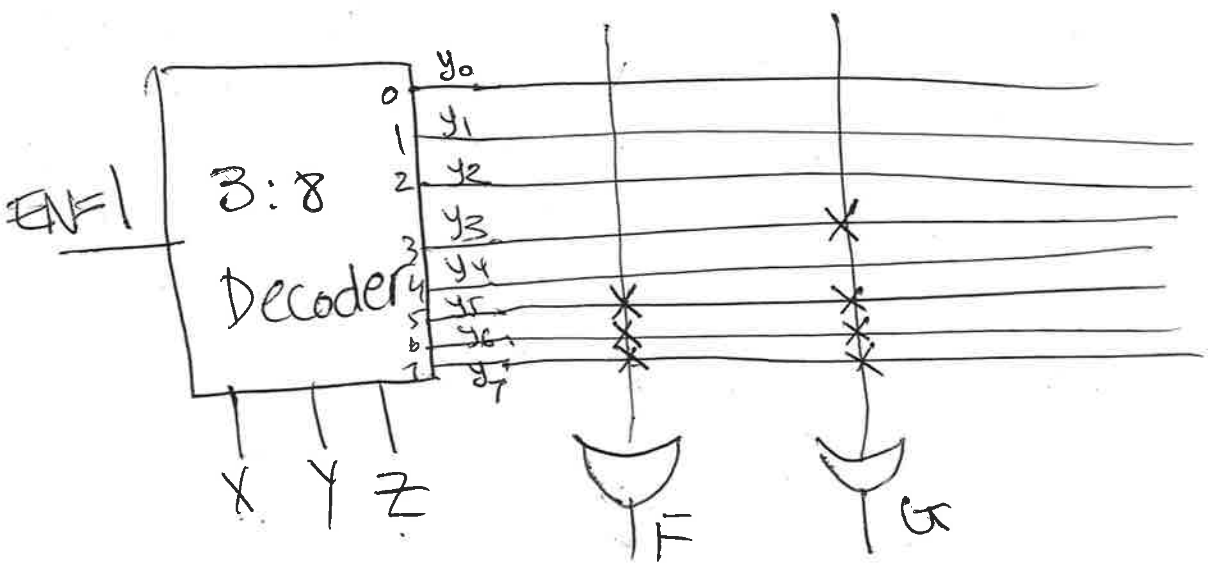
1 PROM :



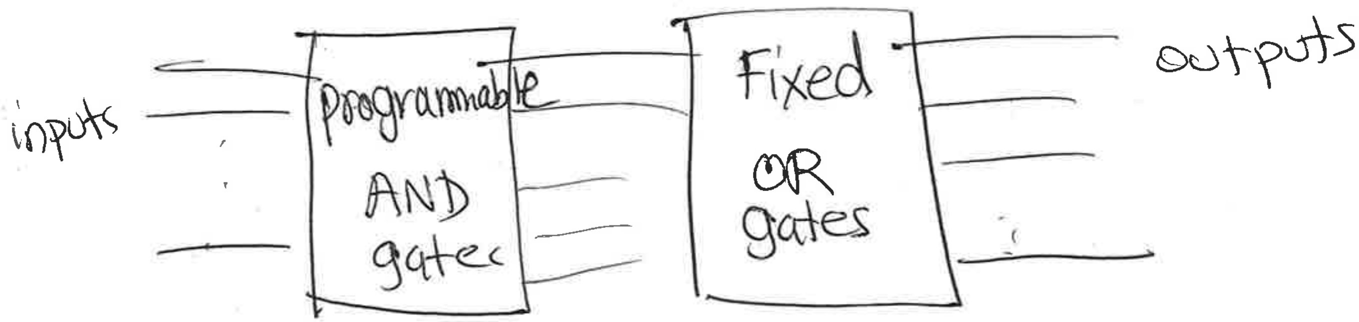
Example : Implement the following outputs using PROM

$$F(X,Y,Z) = \sum m(5,6,7)$$

$$G(X,Y,Z) = \sum m(3,5,6,7)$$



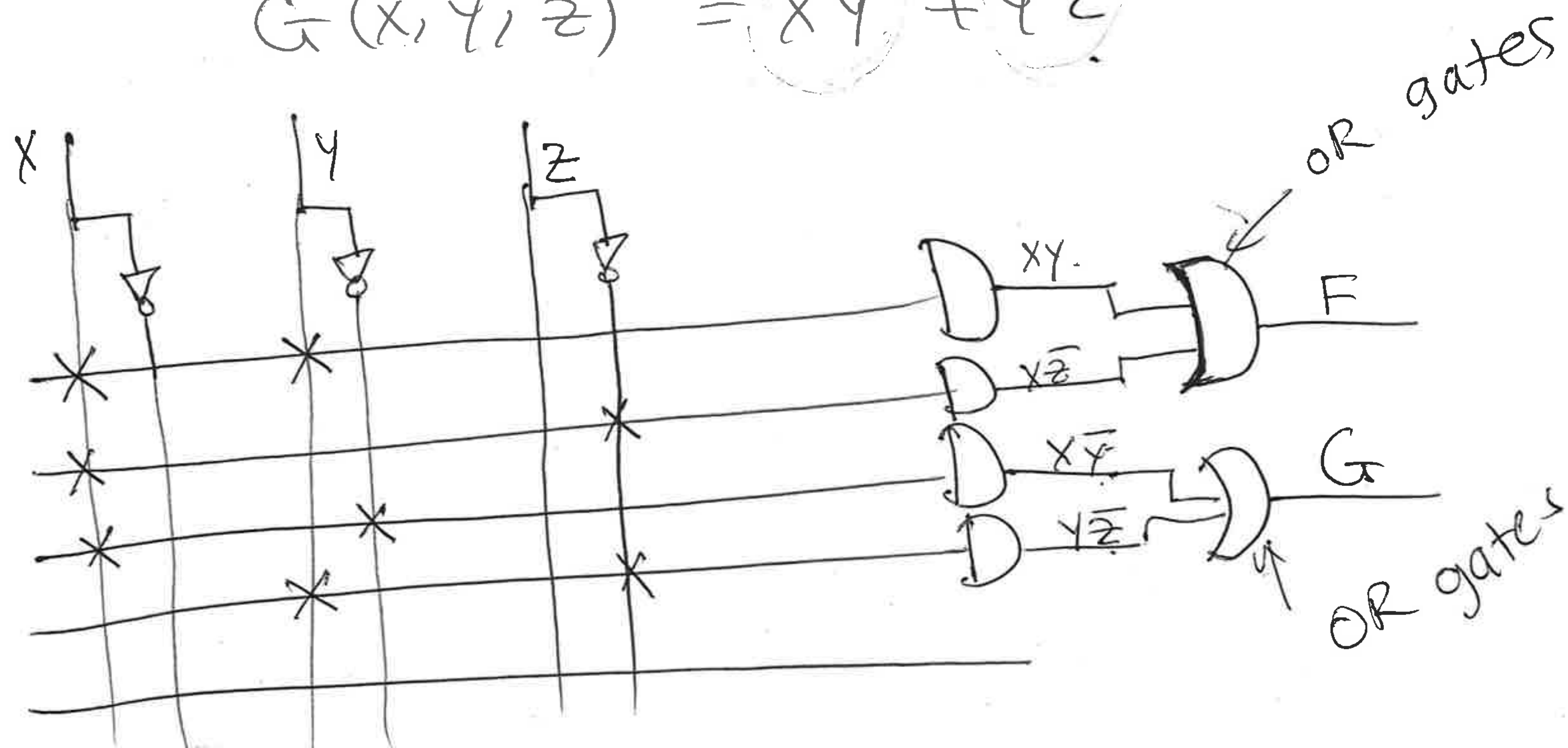
2 PAL



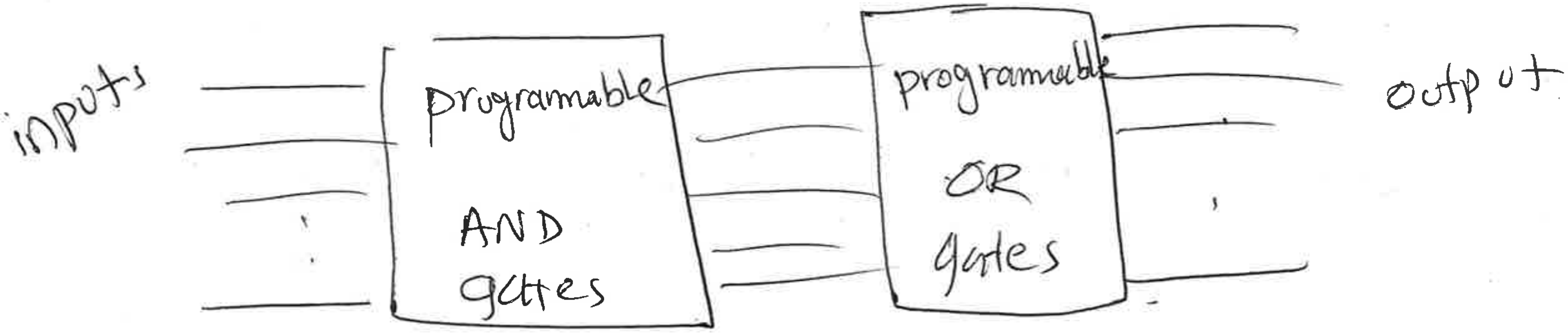
Example: Build the following functions using PAL.

$$F(X,Y,Z) = XY + X\overline{Z}$$

$$G(X,Y,Z) = X\overline{Y} + Y\overline{Z}$$



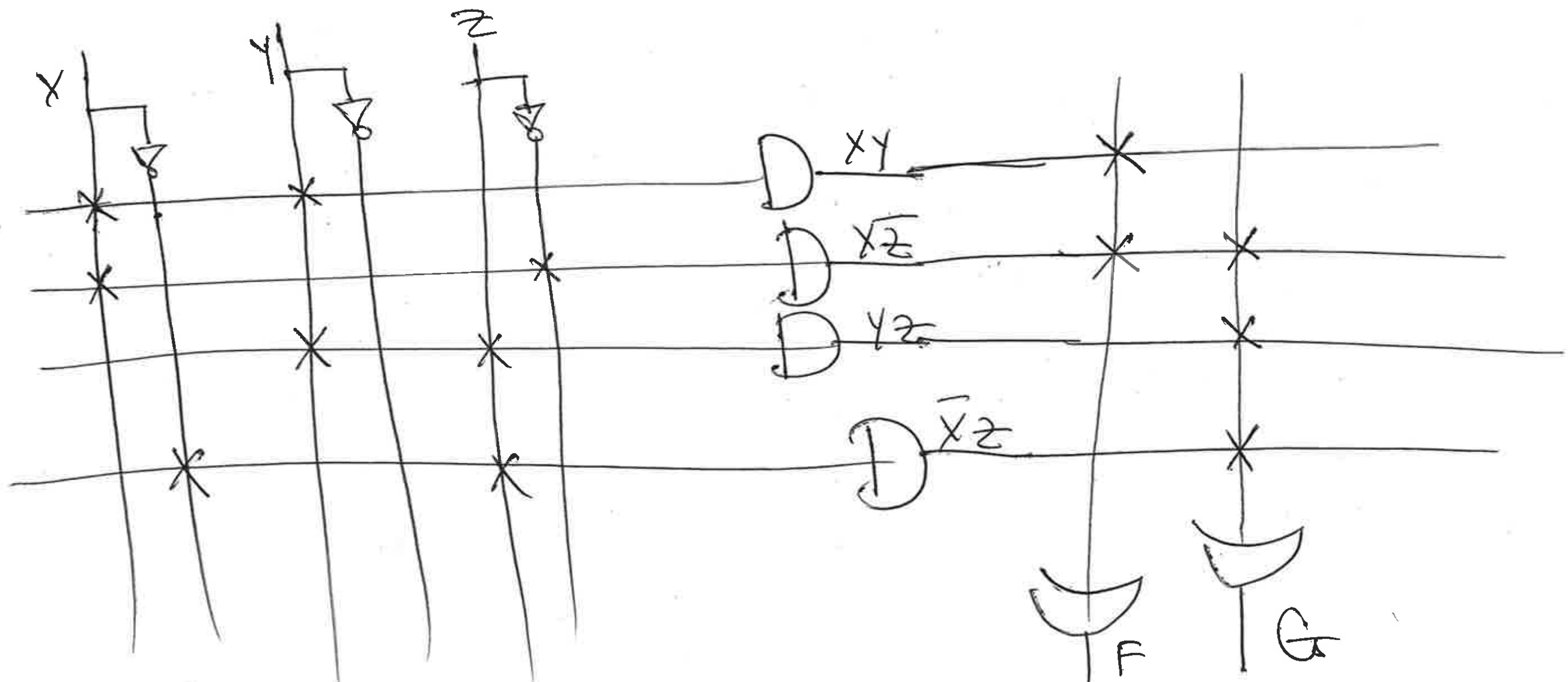
$\boxed{3}$ PLA



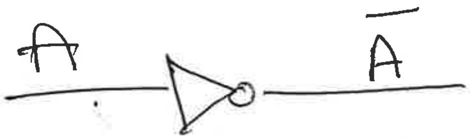
Example :

$$F(X,Y,Z) = XY + X\bar{Z}$$

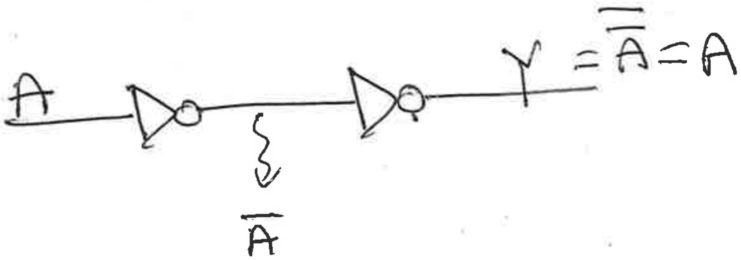
$$G(X,Y,Z) = X\bar{Z} + YZ + \bar{X}Z$$



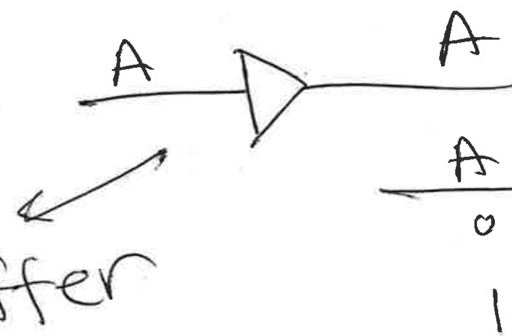
Tri-State Buffer:



A	Y = output
0	1
1	0

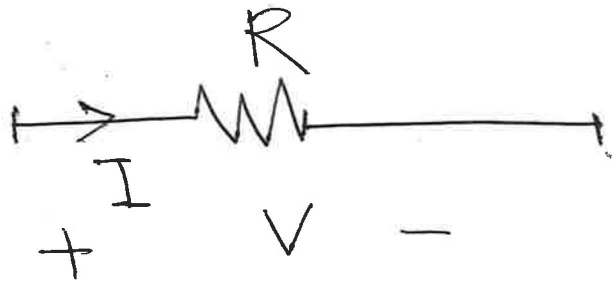


$\Rightarrow$

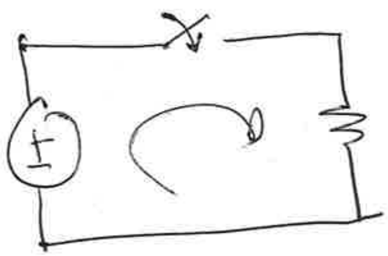


A	Y
0	0
1	1

digital amplification.



$$V = RI \Rightarrow R = \frac{V}{I}$$



switch open $\Rightarrow I = 0$

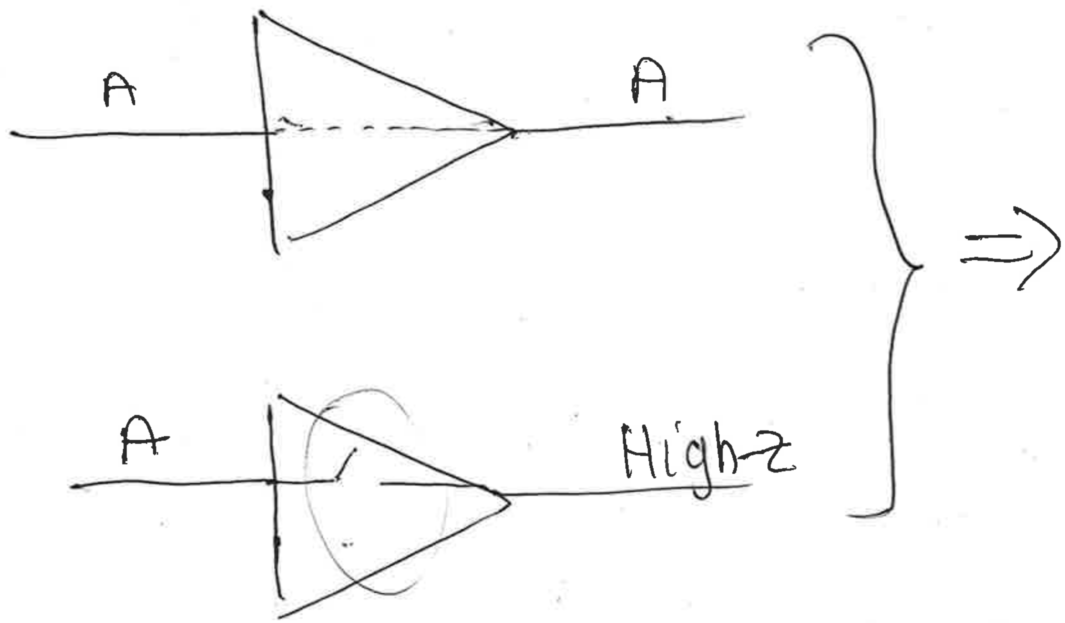
$$R = \frac{V}{0} = +\infty$$

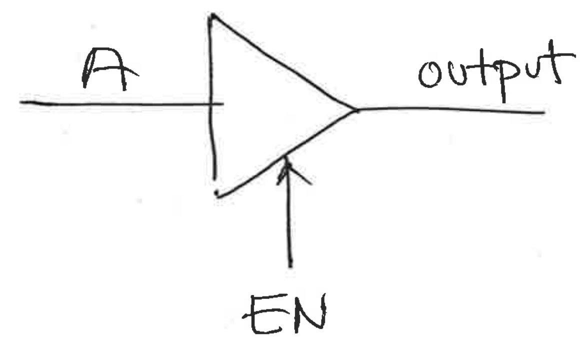
O.C. $\Rightarrow \underbrace{R = \infty}_{\text{high R "}\Omega\text{"}}$

high-z ← impedence

Tri-state Buffer :

→ Can be thought of as an input controlled switch w/ an output that can electronically be turned "ON" or "off".





$EN = 1 \Rightarrow output = A$

$EN = 0 \Rightarrow output = High\text{-}z$

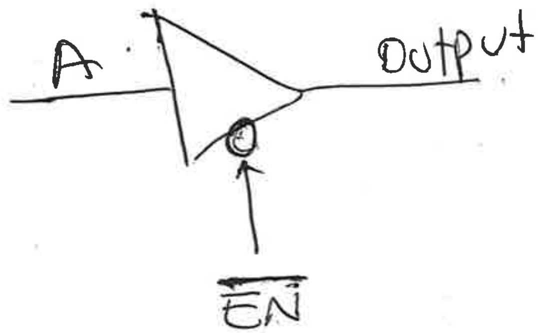
EN	A	output
0	0	High-z
0	1	High z
1	0	0
1	1	1

"Active-high" Tri-state Buffer

Active low Tri-state Buffer

EN=0 ⇒ output = A

EN=1 ⇒ ouput = High-Z

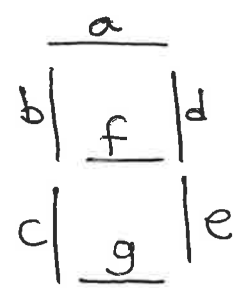


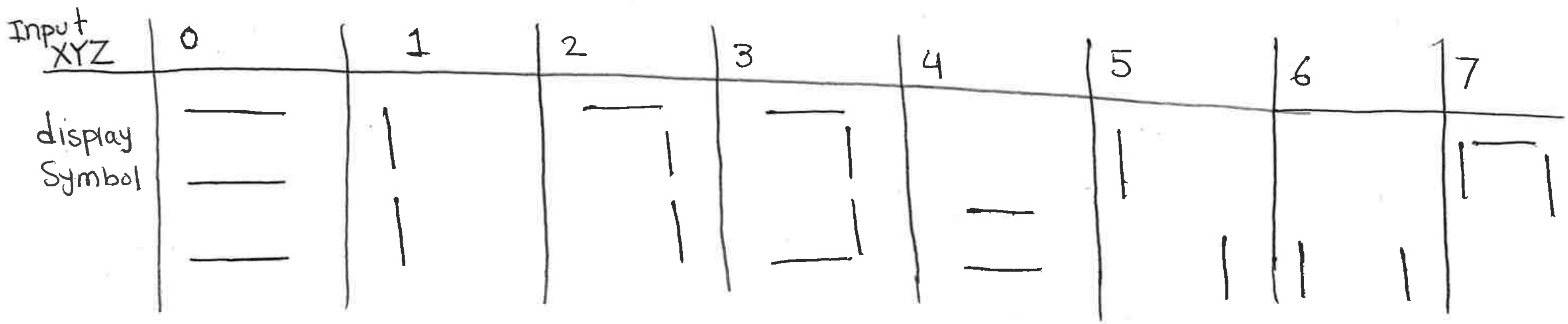
EN	A	output
0	0	0
0	1	1
1	0	High-Z
1	1	High-Z

Short Quiz 2 g

We'd like to design a display that can show the required output:

The display looks like





(A) What is minimum Sum of Product to display segment "a"?

(B) What is minimum Sum of Product to display segment "f"?

Quiz2:

Solution

Indecies that contain segment "a".

(A) $f(x,y,z) = \sum m(0, 2, 3, 7)$

z \ xy	00	01	11	10
0	1	1		
1		1	1	

$\bar{x}\bar{z}$

yz

$$f(x,y,z) = \bar{x}\bar{z} + yz$$

(B) $g(x,y,z) = \sum m(0, 4)$

z \ xy	00	01	11	10
0	1			1
1				

$\bar{y}\bar{z}$

$$g(x,y,z) = \bar{y}\bar{z}$$

Lecture 14 : oct.6, 2020

EEE/CSE 120 : Latches & flipflops

- HW 4 is uploaded
- Lab 2 is on canvas
- office hours T/TH 9:30 - 10:15 AM
- Get ready for mid-term
  - lockdown browser
  - Review your notes

So far; input $\Rightarrow$ output

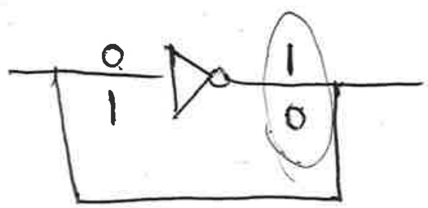
↳ Store the logic value!

– ability to program storage module

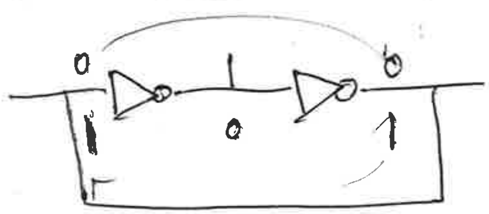
– ability to read the stored value.

Idea: Feed the output back to the input "feed back"

[1] First attempt:

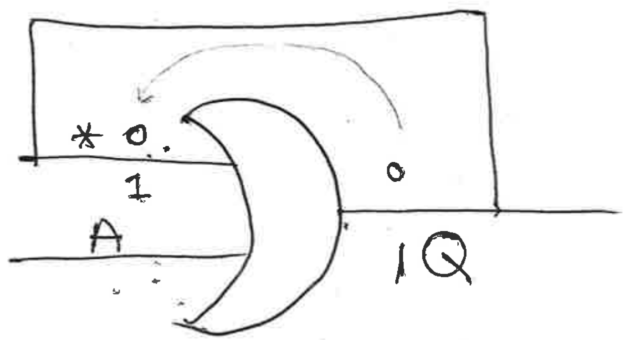


oscillates between "0" & "1" $\Rightarrow$ unstable



$\Rightarrow$ Stable

$\boxed{2}$ Second attempt



$\underline{A = 0}$: $* = 0 \Rightarrow Q = 0$
$* = 1 \Rightarrow Q = 1$

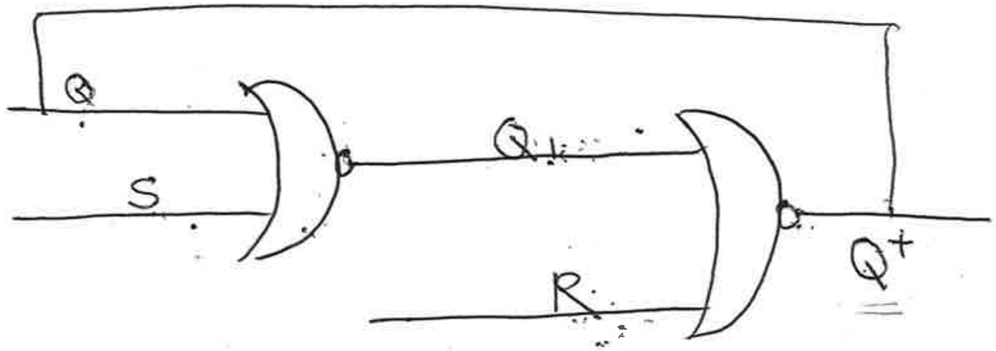
<u>bi-stable</u>

$\underline{A = 1}$: $* = 0 \Rightarrow Q = 1$
$* = 1 \Rightarrow Q = 1$

output switches to 1

$\Rightarrow$ cannot program "0".

3 Third Attempt



- $Q = 0$
  - $S = 0$ → $Q_1 = 1$
    - $R = 0$ → $Q^+ = 0$
    - $R = 1$ → $Q^+ = 0$
  - $S = 1$ → $Q_1 = 0$
    - $R = 0$ → $Q^+ = 1$
    - $R = 1$ → $Q^+ = 0$

- $Q = 1$
  - $S = 0$ → $Q_1 = 0$
    - $R = 0$ → $Q^+ = 1$
    - $R = 1$ → $Q^+ = 0$
  - $S = 1$ → $Q_1 = 0$
    - $R = 0$ → $Q^+ = 1$
    - $R = 1$ → $Q^+ = 0$

Set $S=1$:

$Q_1 = 0$

$\Downarrow$

Q^+ only depends on R

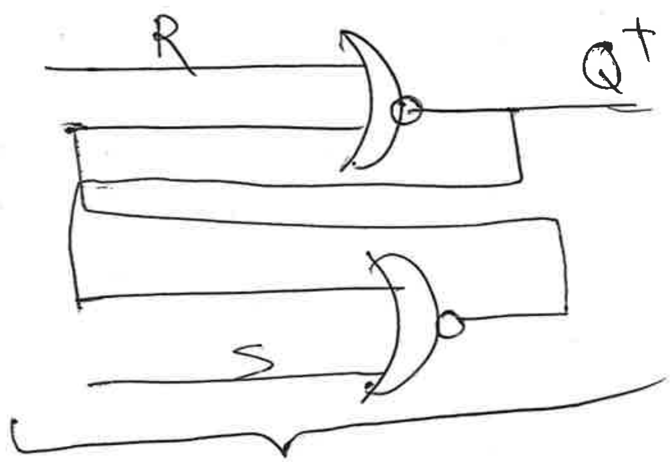
$R=0 \Rightarrow Q^+ = 1$

$R=1 \Rightarrow Q^+ = 0$

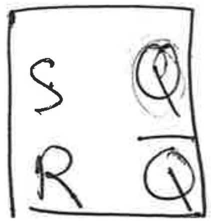
S	R	Q	Q^+ ← Qnext	
0	0	0	0	bi-stability ⟹ Storage part
0	0	1	1	
0	1	0	0	← Program "0"
0	1	1	0	
1	0	0	1	← Program "1"
1	0	1	1	
1	1	0	0	Exclude R=1, S=1
1	1	1	0	

State transition table

S	R	Q^+
0	0	Hold data (Q)
0	1	0
1	0	1
1	1	X



SR latch

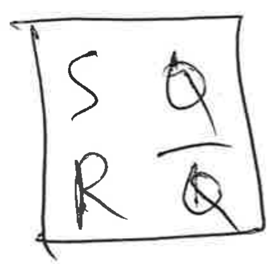


SR-latch

S	R	Q^t
0	0	Storage
0	1	0
1	0	1
1	1	X

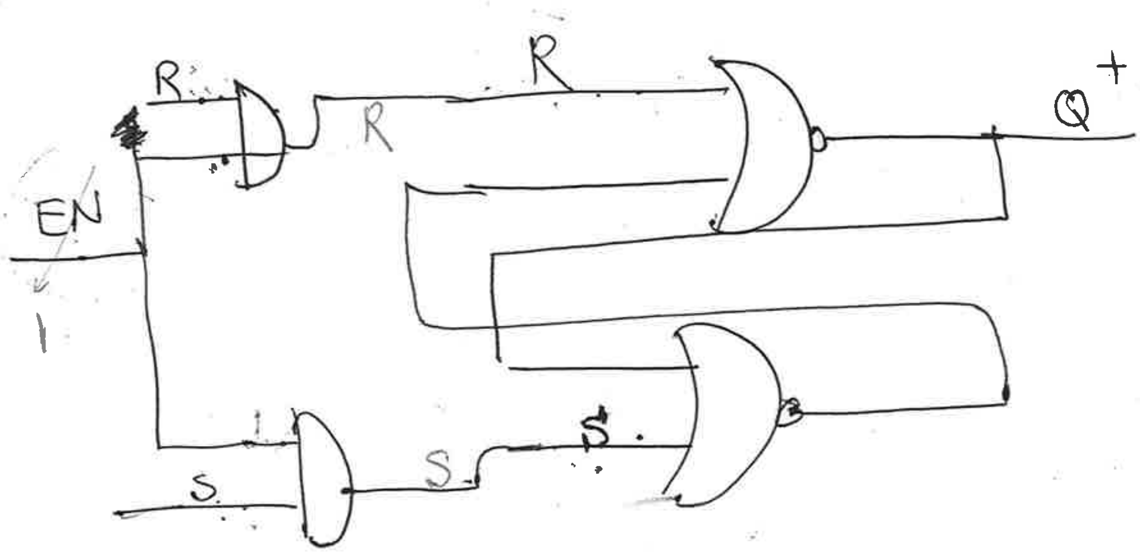
Observations :

- If $S = R = 0$ ⇒ stable states
- If switch S to 1 (temporarily) ⇒ $Q^t \to 1$
  If we switch S back to zero ⇒ Q^t stays at 1
  ⇒ "Set"
- If switch R to 1 (temprorily) ⇒ $Q^t = 0$
  switch back to 1 ⇒ Q^t stays at " "
  ⇒ Reset



SR-latch

SR latches w/ enable :



EN=1 ⇒ SR-latch

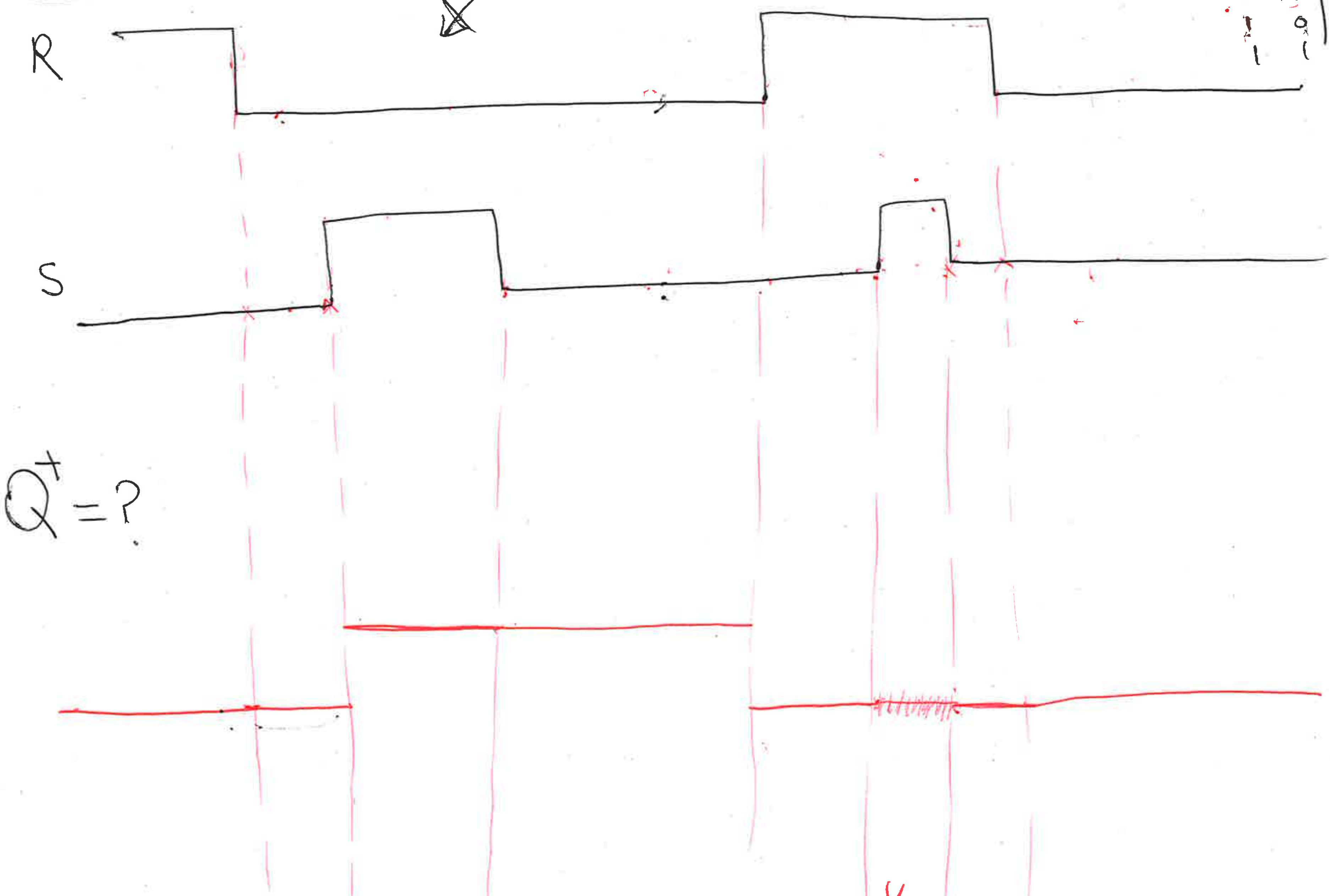
EN=0 : storage mode

EN	S	R	Q^+
0	X	X	storage mode
1	0	0	storage
1	0	1	0
1	1	0	1
1	1	1	X Forbidden

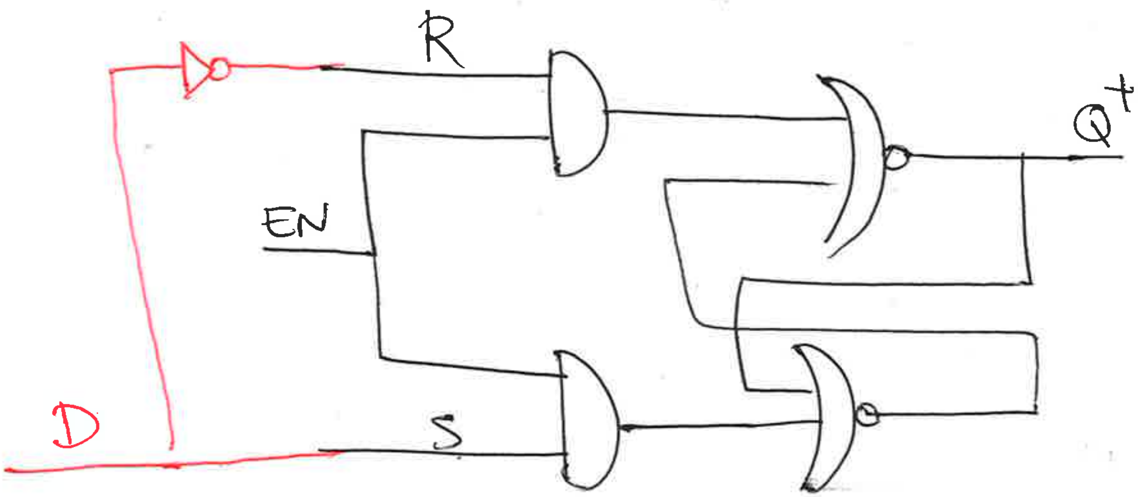
Example:

Timing diagrams

S	R	Q^+
0	0	storage
0	1	0
1	0	1
1	1	X



"level sensitive"



EN	S	R	Q^t
0	X	X	Storage "Q"
1	0	0	Q (storage)
1	0	1	0
1	1	0	1
1	1	1	X

EN	D	Q^t
0	X	Storage
1	0	0
1	1	1

=>

EN	D	Q^t
0	X	Storage
1	D	D

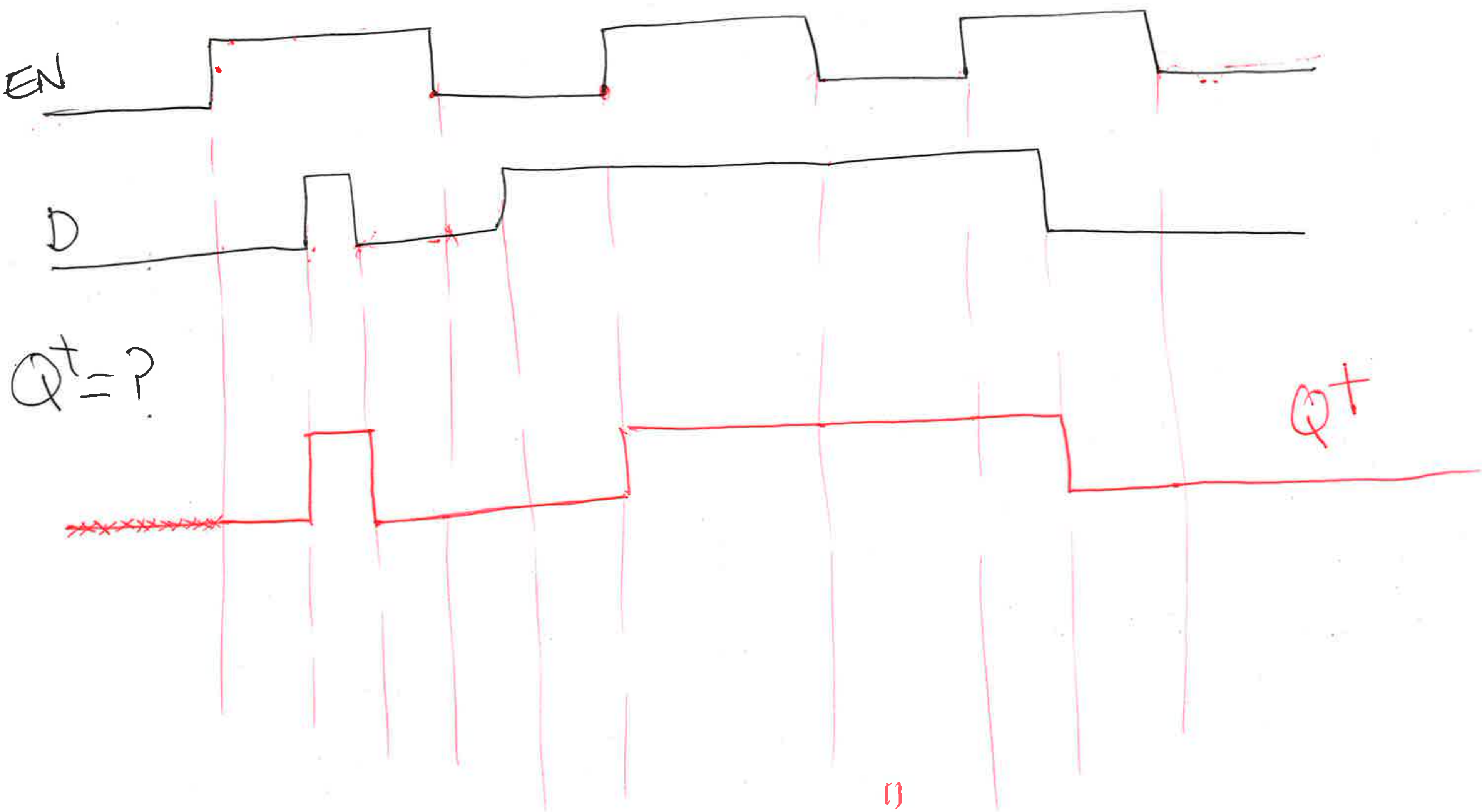
state transition table

$\boxed{Q^t = D}$

behavioral eq.

$\boxed{\text{D-latch}}$

Example:



"level sensitive"

Lecture 15 : Flip Flops

oct 8, 2020

- Lab office hours next week :

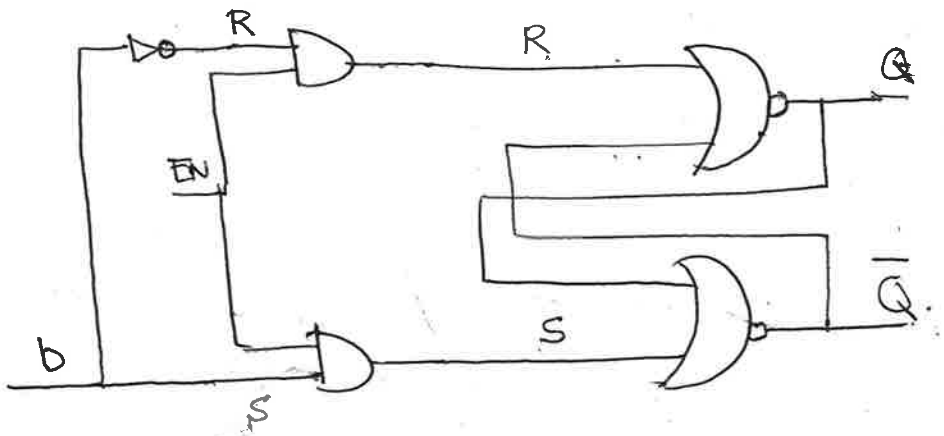
  Monday 5-6 pm

  Wednesday 12-1:30 pm

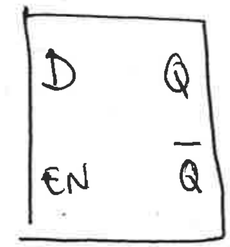
  Friday 2:30- 4:30pm

- Make sure you have lockdown browser

Last time :



D - latch



EN=1 ⇒ Active high
high D-latch

EN=0 ⇒ low D-latch

Next state

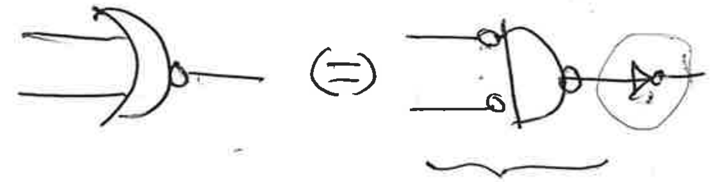
EN	D	Q^+
0	x	storage Q
1	0	0
1	1	1

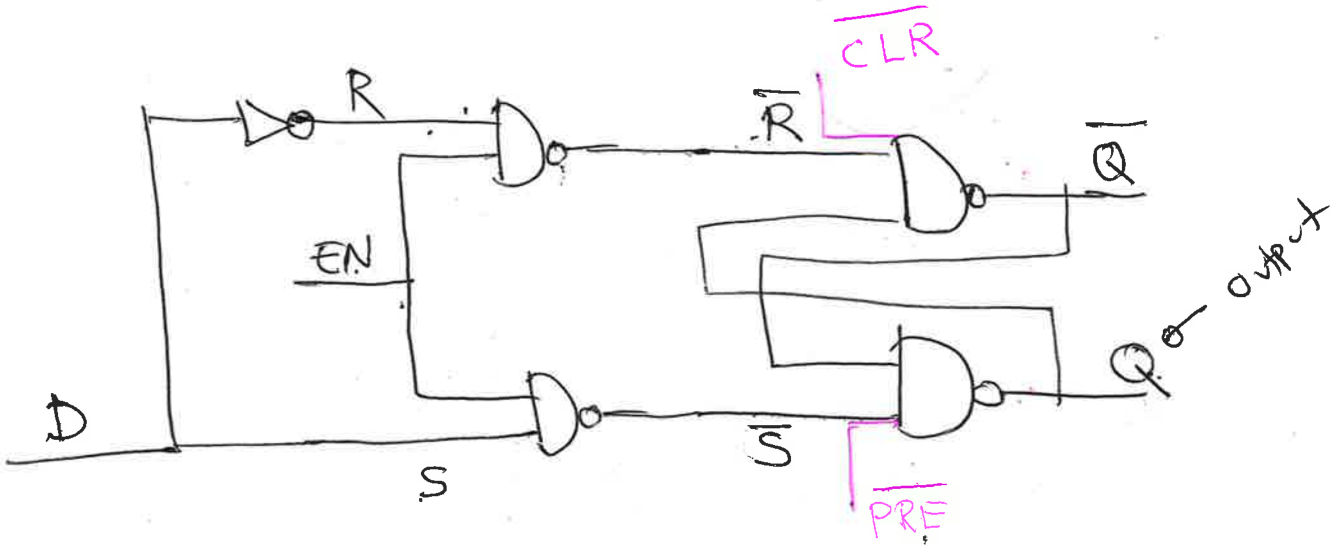
D D

high D-latch

$$\Rightarrow \underbrace{Q^+ = D}_{\text{behavioral eq.}}$$

D-latch using "NAND" gates





EN=1 ⟹ latch working

EN=0 ⟹ Storage mode

EN	D	Q^+
0	x	Q
1	D	D

$\overline{CLR}=0 \Rightarrow \overline{Q}=1 \Rightarrow \boxed{Q=0}$

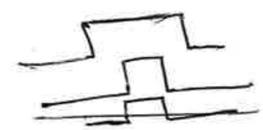
$\Updownarrow$

$CLR=1$

$\overline{PRE}=0 \Rightarrow \boxed{Q=1}$

$\Updownarrow$

$PRE=1$

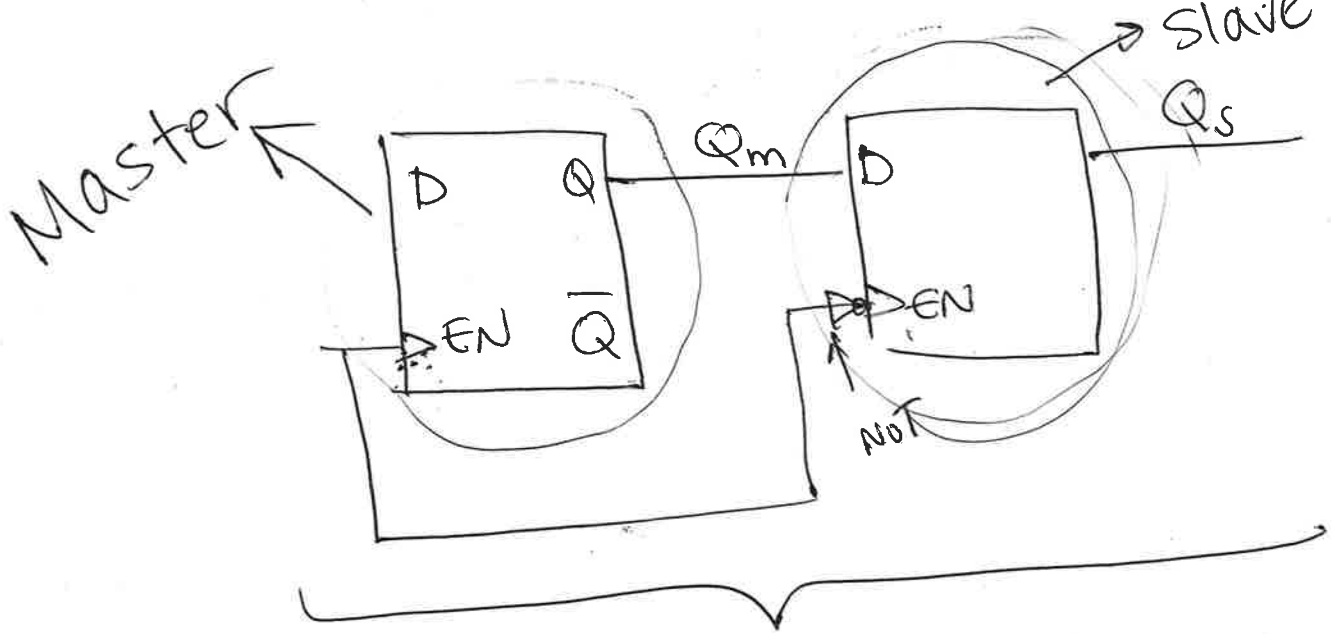


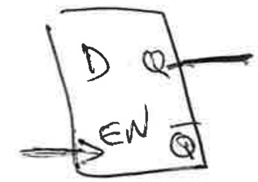
D-latches are level sensitive ⟹ output will be proprogrammed "EN=1"

Transparency.

Q: Can we design an "EN" that only enables when signal changes $\underbrace{0 \rightarrow 1}_{\text{positive edge}}$ or $\underbrace{1 \rightarrow 0}_{\text{Negative edge}}$

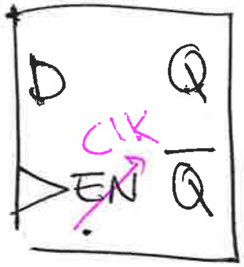
How do we do this ?





sequence of latches

D_flip flop



positive edge – D-flip flop

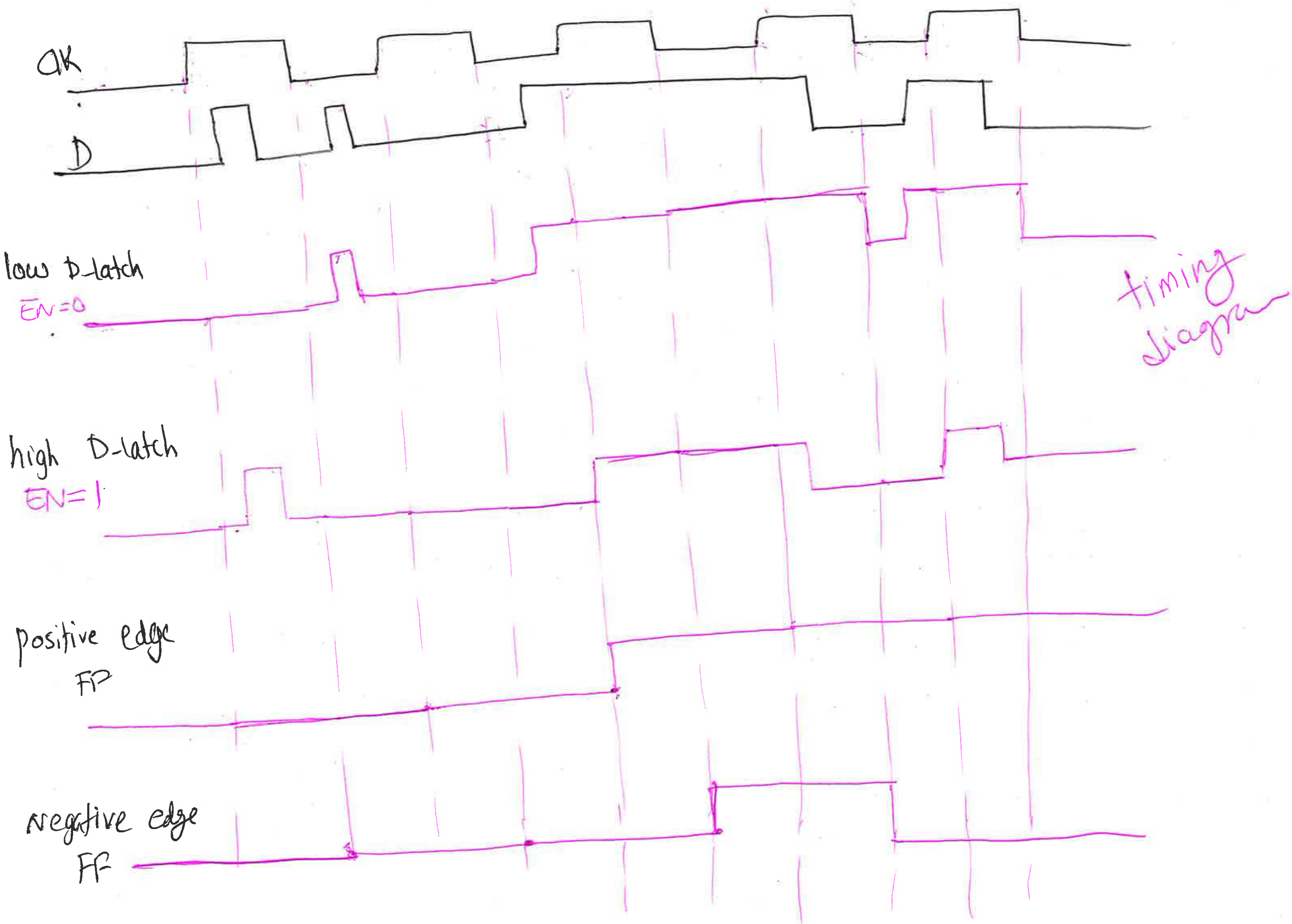
negative edge – D-flip flop

$$Q^+ = D$$

as long as ~~neg~~ rising edge $(0 \rightarrow 1)$

EN	D	
0	X	storage
1	D	D

Example: D-flip flop



Other types of FFs

① JK flip flop

J	K	Q^+ ← Next state
0	0	Storage mode "Q"
0	1	0
1	0	1
1	1	$\overline{Q}$

State transition table

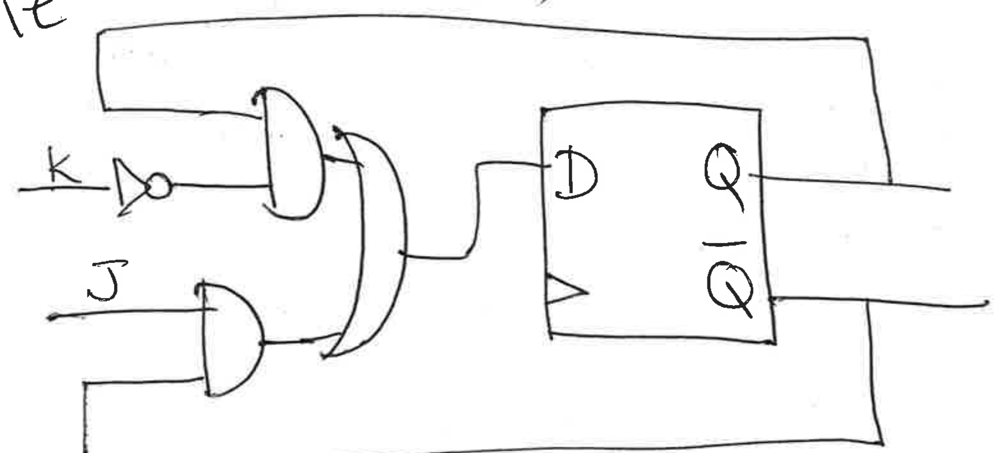
K-MAP for Q^+

Q \ JK	00	01	11	10
0			1	1
1	1			1

$J\overline{Q}$

$\overline{K}Q$

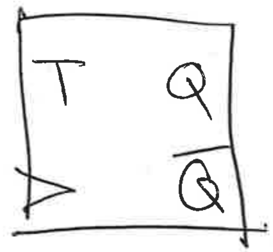
$$\boxed{Q^+ = J\overline{Q} + \overline{K}Q}$$

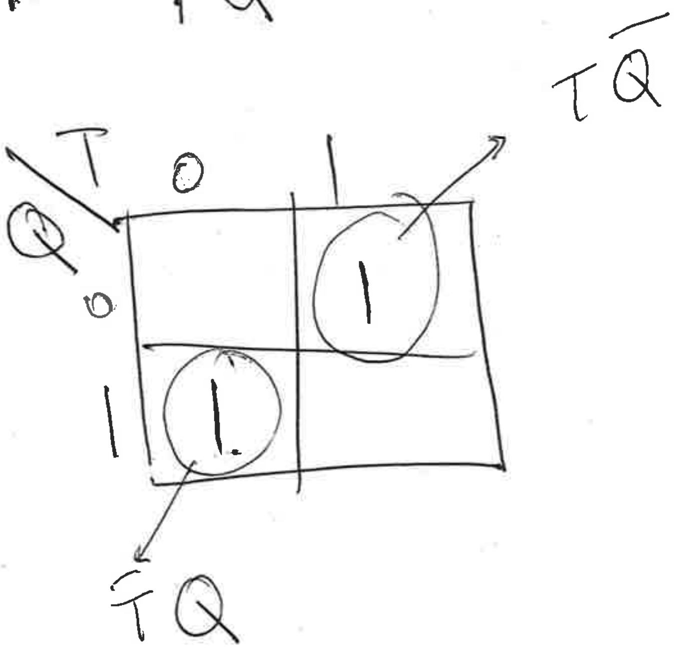


JK flip-flop

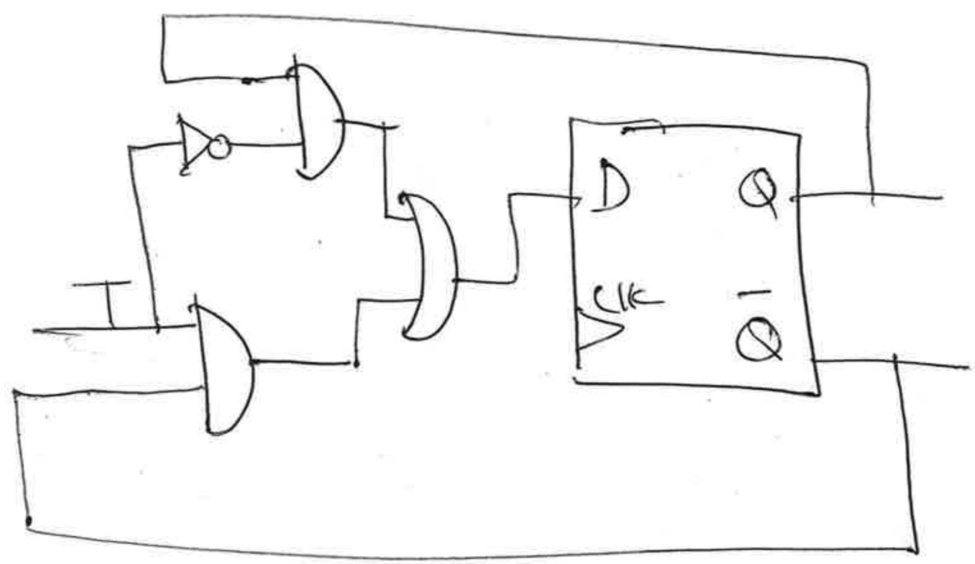
② Toggle flip flop

T	Q^+
0	Q
1	$\bar{Q}$





$$Q^+ = T\bar{Q} + \bar{T}Q$$



Toggle flip flop.

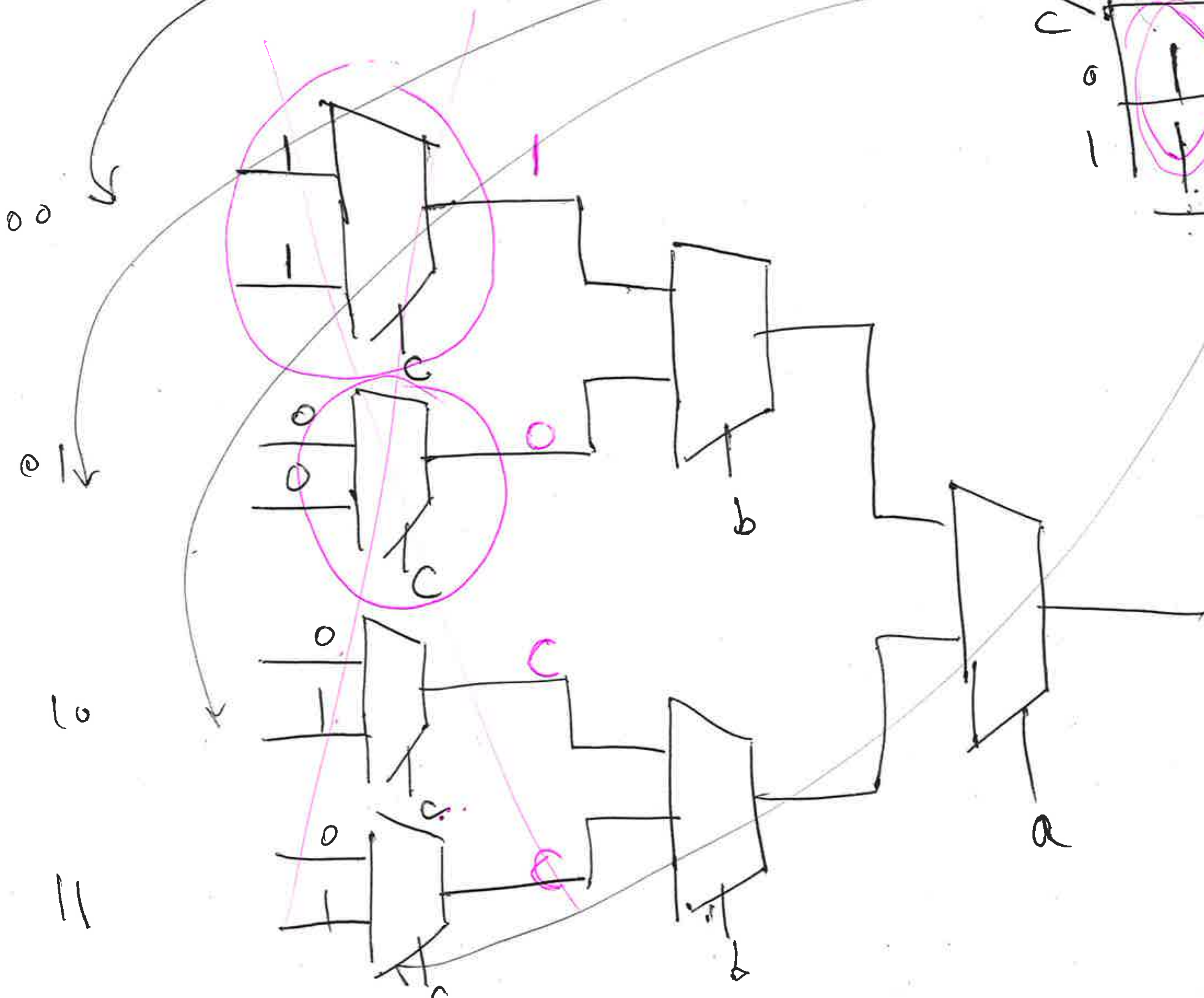
Example:

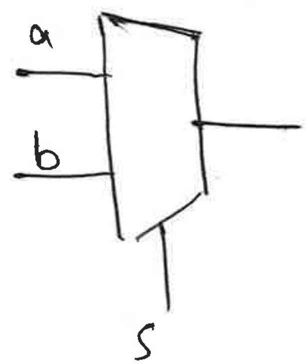
$$f(a,b,c) = \sum m(0,1,5,7)$$

Using MUX to build this function.

c \ ab	00	01	11	10
0	1			
1	1		1	1

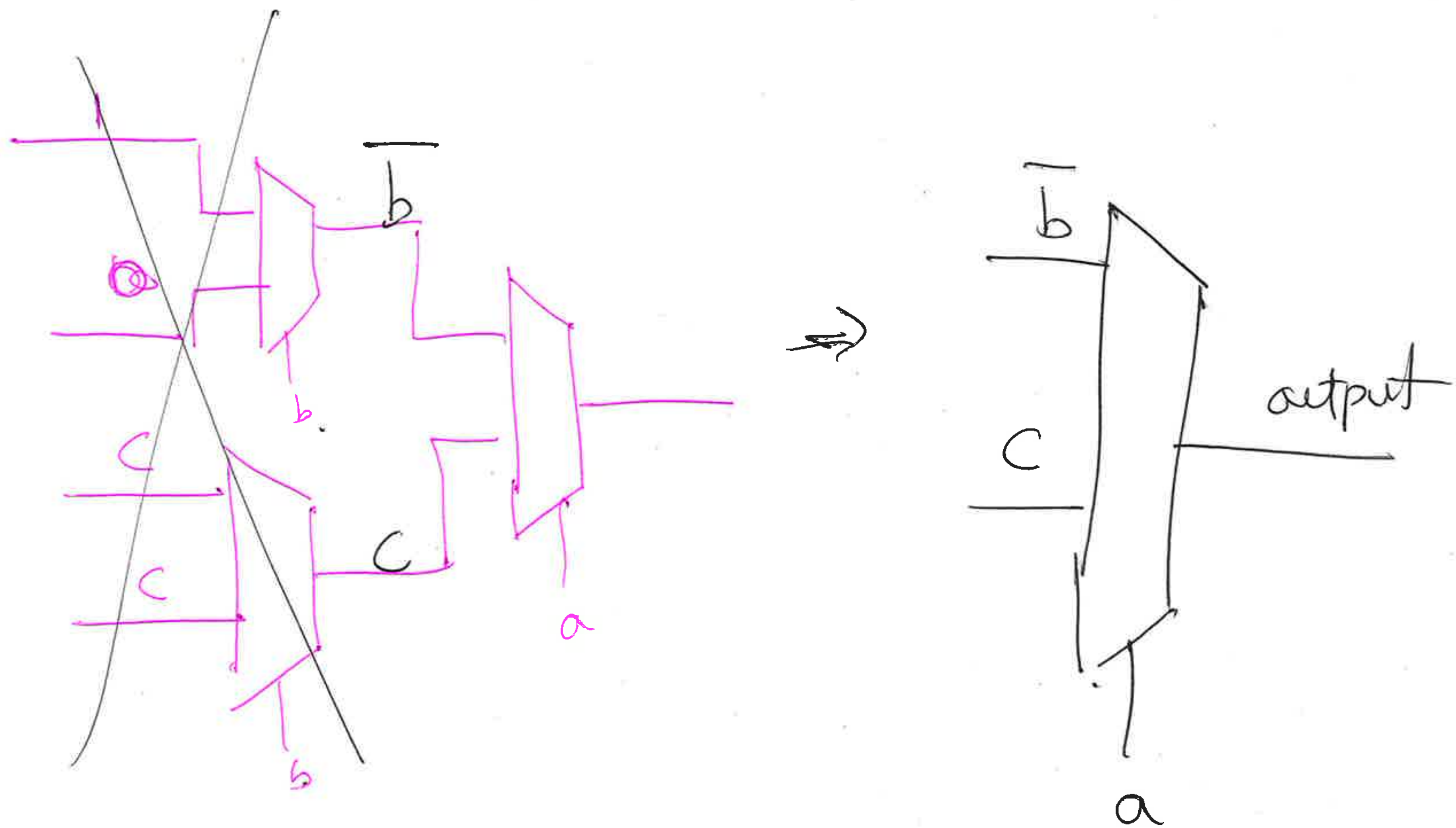
$\bar{a}\bar{b} + ac$





$s = 0 \Rightarrow y = a$

$s = 1 \Rightarrow y = b$



$\bar{b}$
b
c
c
b
c
a
$\bar{b}$
c
output
a

Oct 13, 2020

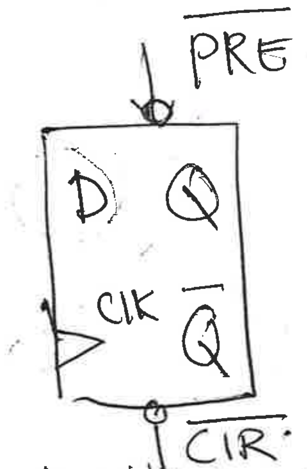
EEE/CSE 120 : Registers

- Office hours T/TH 9:30-10:15 AM
- Lab office hours | Wed : 12-1:30 pm
  | Fri : 2:30-4:30 pm

- Lab 3 is uploaded

- Make sure you have lock-down
  - Mock exam!

FF:

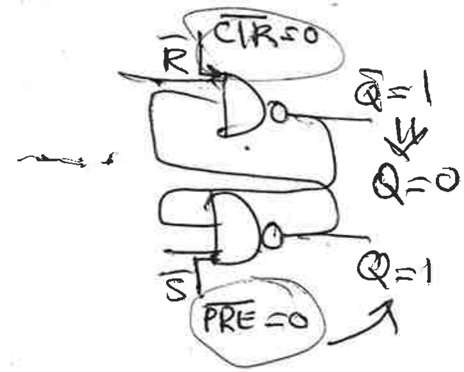


positive edge FF

$0 \rightarrow 1$

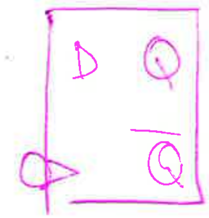
$\overline{CLR} = 0 \Rightarrow Q = 0$

$\overline{PRE} = 0 \Rightarrow Q = 1$



$$Q^+ = D$$

Next State

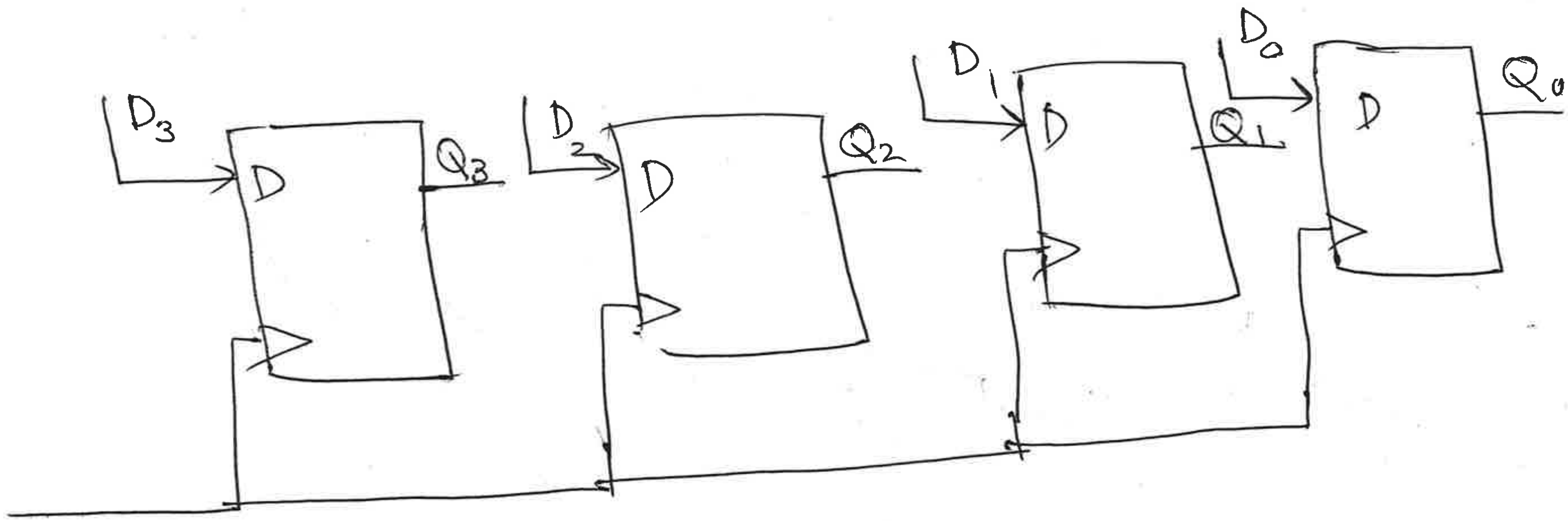


Negative edge

$1 \rightarrow 0$

Register: Any combination of flip flops is called a register!

Example : (parallel in - parallel out)



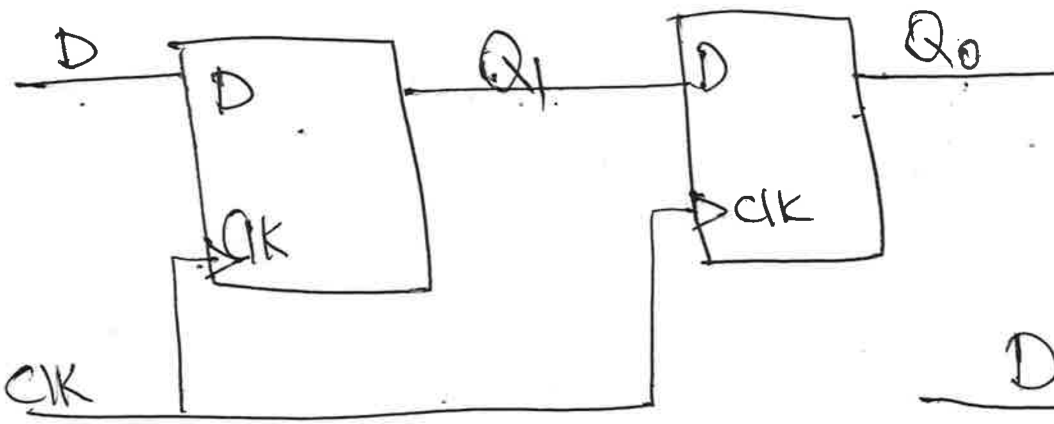
Same clock $\Longrightarrow$ synchronous.

$D_3 D_2 D_1 D_0$ (4 bit)

$Q_3 Q_2 Q_1 Q_0$ (output)

Each clock comes in 4-bit binary number and loads 4 bit out.

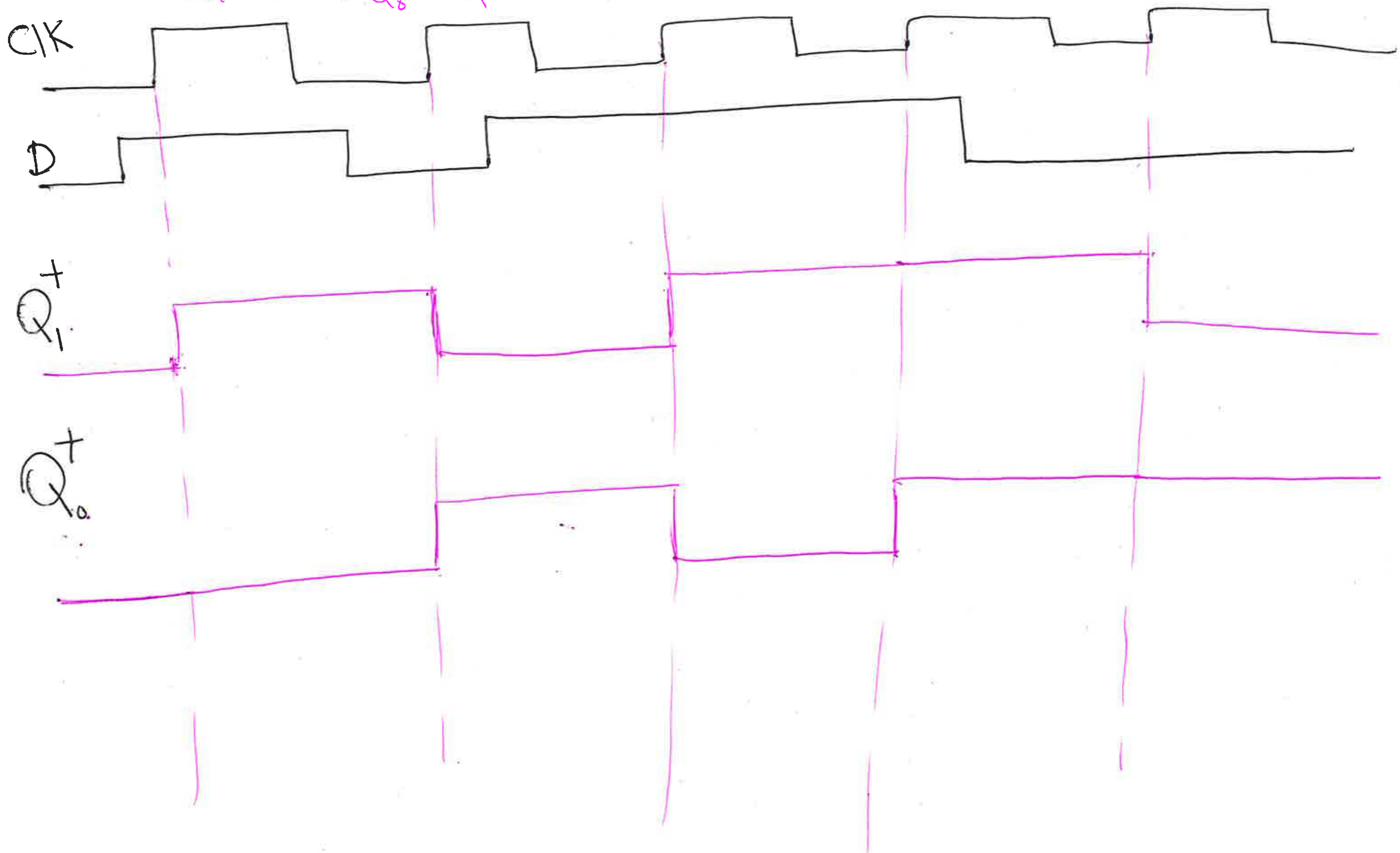
Example : (Shift register)



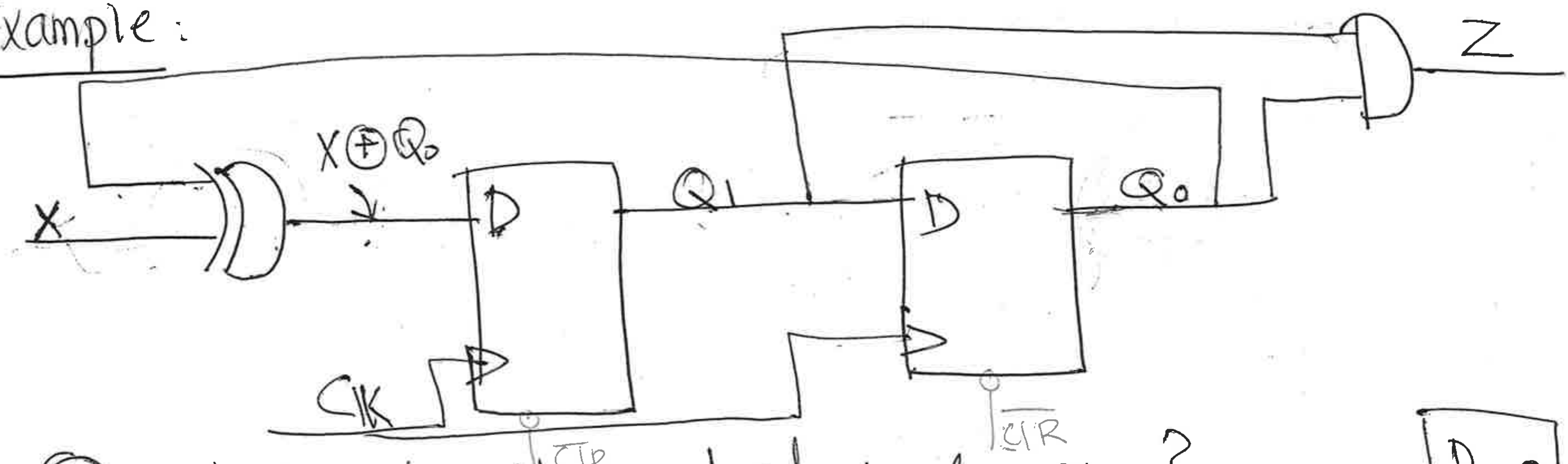
$$\begin{cases} Q_1^+ = D \\ Q_0^+ = Q_1 \end{cases}$$

D	Q_1	Q_0	Q_1^+	Q_0^+
0	0	0	0	0
0	0	1	0	0
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	1
1	1	1	1	1

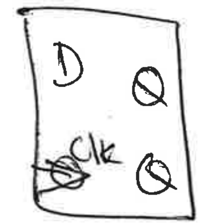
$Q_1^+ = D$, $Q_0^+ = Q_1$



Example:



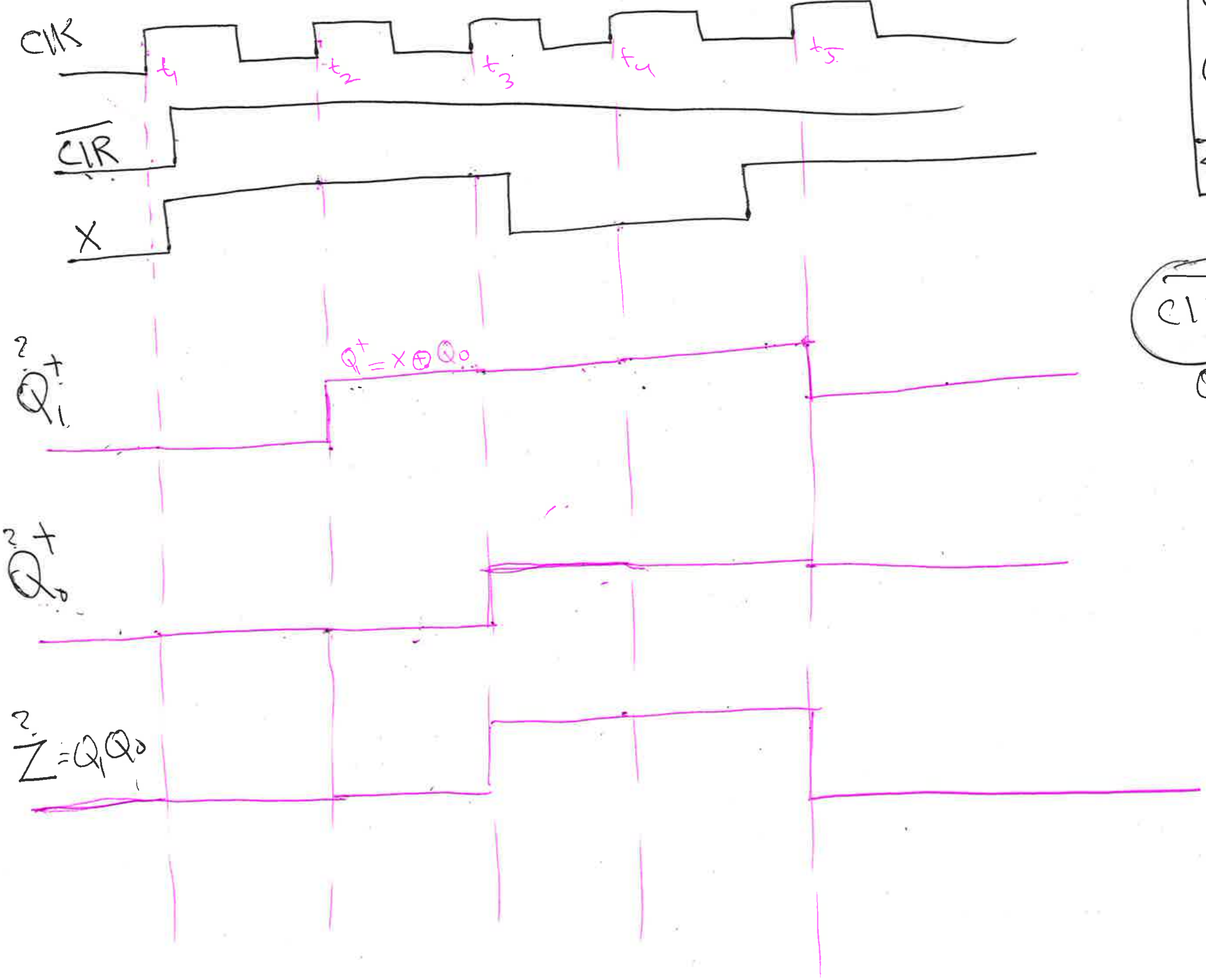
① what is the behaivioral eq.?



$Q^+ = D$

next states
$$Q_1^{+} = X \oplus Q_0$$
$$Q_0^{+} = Q_1$$

output $Z = Q_1 Q_0$



$Q_1^+ = X \oplus Q_0$

$Q_0^+ = Q_1$

$Z = Q_1 Q_0$

$ClR = 1$

$\overline{ClR} = 0 \Rightarrow$

output $= 0$

Example: Register that follows these equations:

Next state

$$Q_2^+ = Q_1 \oplus Q_2$$

$$Q_1^+ = Q_0$$

$$Q_0^+ = X + Q_2$$

output

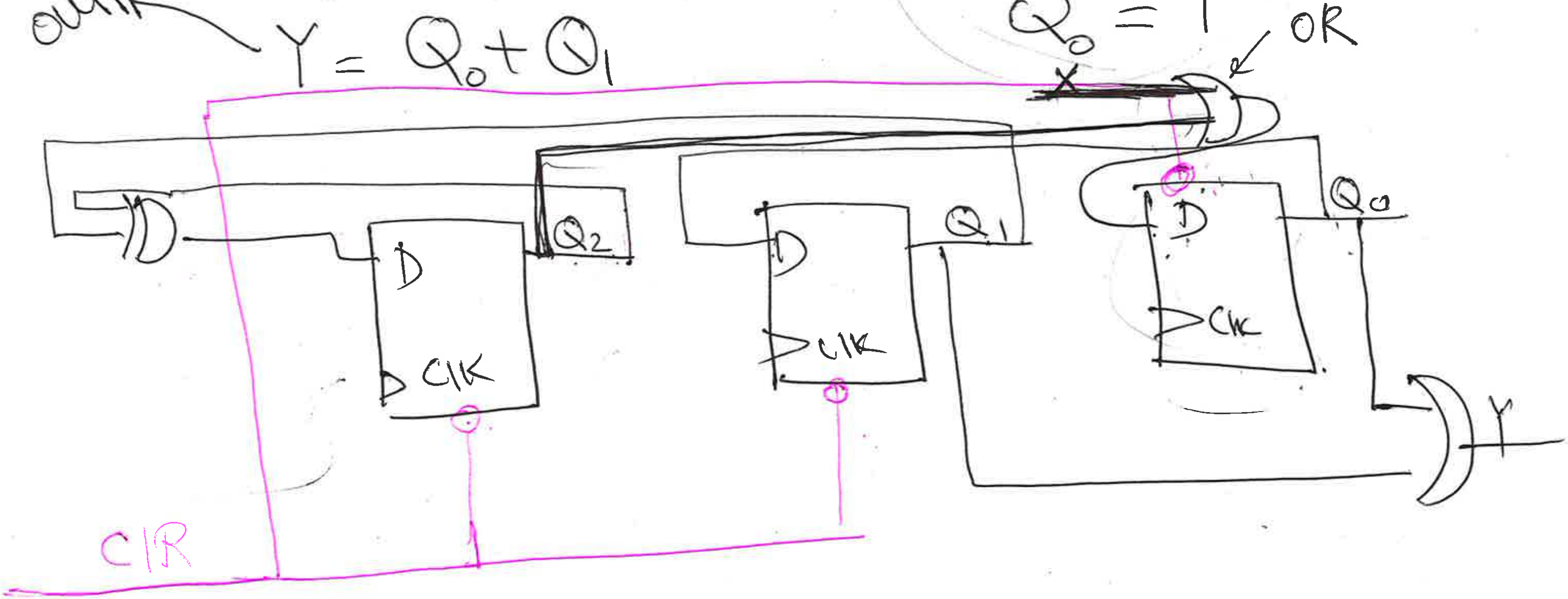
$$Y = Q_0 + Q_1$$

Assuming

$Q_2 = 0$

$Q_1 = 0$

$Q_0 = 1$



ClR = 1

(Synchronous, CLKS are connected to the same clk.)

EEE/CSE 120 : Review for Midterm

office hours :

$\boxed{\text{Monday } 3{:}00 - 4{:}00 \text{ pm}}$

Example 1: Add # (Signed)

Cin

$\boxed{1}$ 1 1 0

  1 0 0 1 1 → $(-13)_{10}$ (01101) → +13

\+ 0 1 1 1 0 $(14)_{10}$

Cout

$\boxed{1}$ (0 0 0 0 1)   +1

$C_{in} = C_{out} \Rightarrow$ NO OF.

cin

$\boxed{0}$ 0 0

  1 0 0 1 0 → $\overbrace{(01110)}^{14}$ → −14

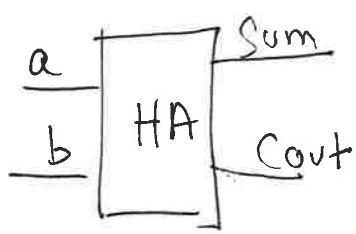
2^4 2^3 2^2 2^1 2^0

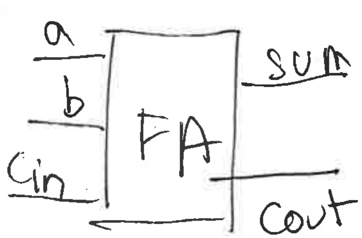
\+ 1 1 0 0 1 → $\underbrace{(00111)}_{7}$ → −7

$\boxed{1}$ (0 1 0 1 1)

cout

$C_{in} \neq C_{out} \Rightarrow$ OF.





Example 2 :

$$(213)_{16} \longrightarrow (\qquad)_{10} \text{ Decimal}$$

$16^2 \quad 16^1 \quad 16^0$

$$3 \times 16^0 + 1 \times 16^1 + 2 \times 16^2$$

$$(327)_{10} \longrightarrow \text{HEX}$$

$$(A)_{10} = \sum_{i=1} c_i a^i = (\;\;\;)_a$$

$327 \div 16 = 20$, R 7

$20 \div 16 = 1$, R 4

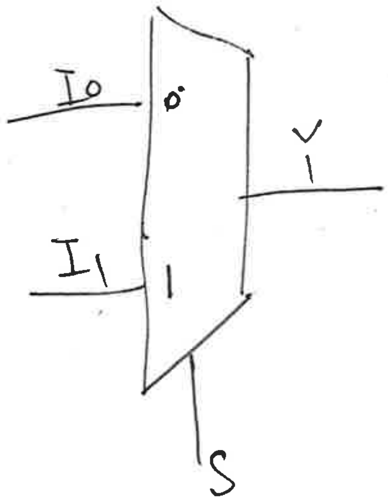
$$(147)_{16}$$

$$\underbrace{1 \times 16^2 + 4 \times 16^1 + 7 \times 16^0}_{327}$$

$$(F729)_{16} \longleftrightarrow \text{binary}$$

$16 \to 2^4$

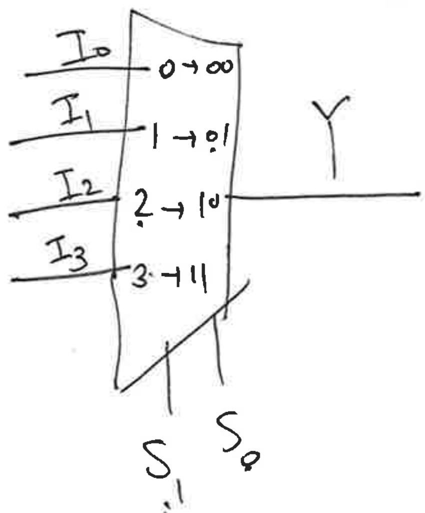
$$(1111 \; 0111 \; 0010 \; 1001)_2$$



$$S = 0 \Rightarrow Y = I_0$$

$$S = 1 \Rightarrow Y = I_1$$

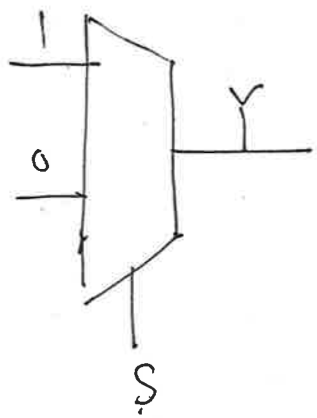
$$Y = \bar{S} I_0 + S I_1$$



$$Y = \bar{S}_1 \bar{S}_0 I_0 + \bar{S}_1 S_0 I_1 + S_1 \bar{S}_0 I_2 + S_1 S_0 I_3$$

Is a Mux functionally complete?

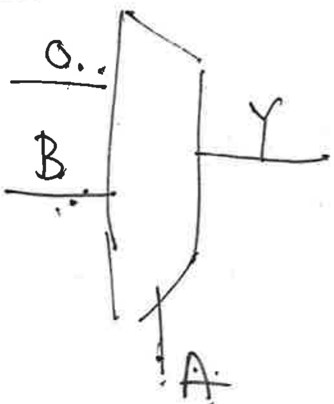
① Invertor



$S=0 \Rightarrow Y=1$

$S=1 \Rightarrow Y=0$ $\rightsquigarrow$ $Y=\overline{S}$ ✓

② AND

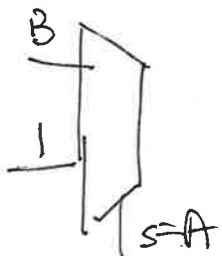


$A=0 \Rightarrow Y = 0B = AB$

$A=1 \Rightarrow Y = B = \underbrace{1}_{A} \cdot B = AB$

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

③ OR

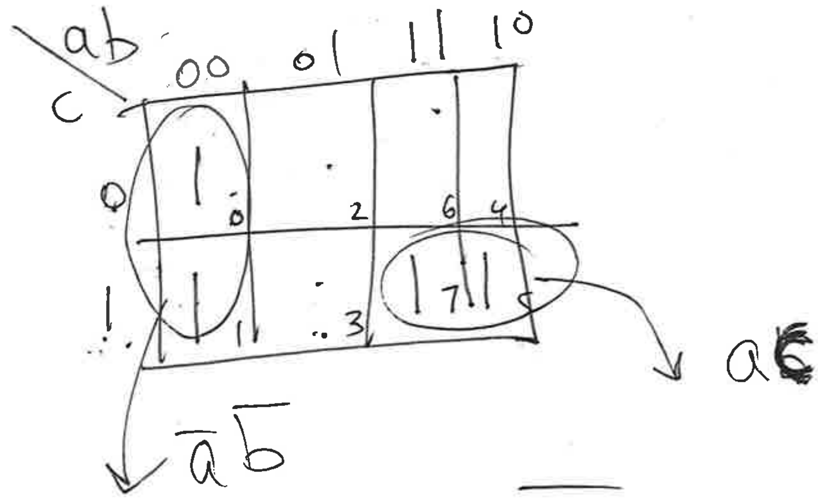


$$\left.\begin{array}{l} A=0 \Rightarrow Y=B \\ A=1 \Rightarrow Y=1 \end{array}\right\} A+B$$

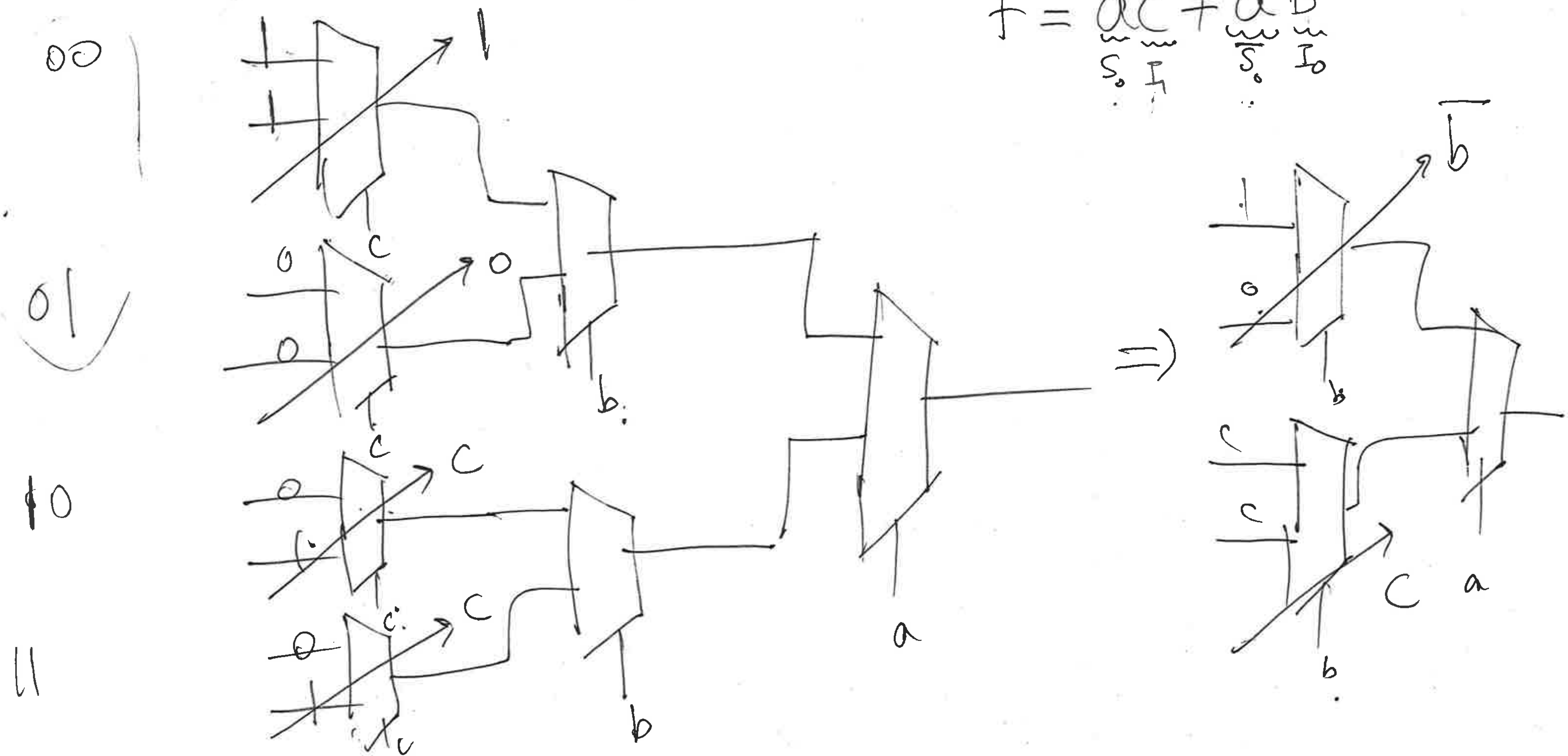
0	0	0
0	1	1
1	0	1
1	1	1

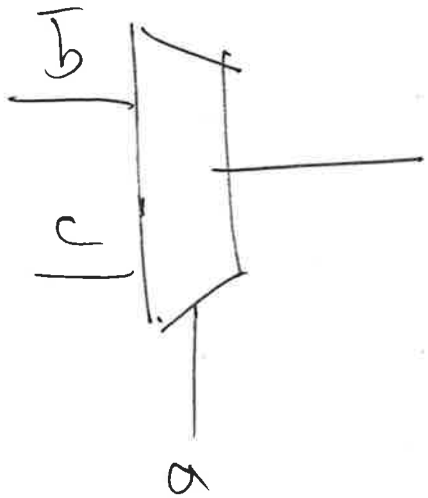
$f(a,b,c) = \sum m(0,1,5,7)$

- use 7 multiplexers
- use 3 muxes
- use 1 mux.

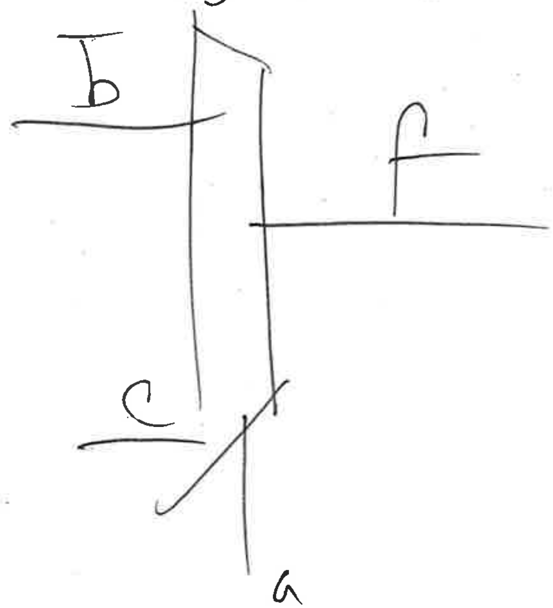


$$f = \underbrace{a}_{S_0}\underbrace{c}_{I_1} + \underbrace{\bar{a}}_{\bar{S}_0}\underbrace{\bar{b}}_{I_0}$$





$$f = \underbrace{a}_{s}\,\underbrace{c}_{I_1} + \underbrace{\bar{a}}_{\bar{s}}\,\underbrace{\bar{b}}_{I_0}.$$



Example 4 :

$$f(a,b,c,d) = \sum m(0,2,5,10,15) + \sum d(4,6,7,8,13)$$

① Find minimum SoP ?

② Find minimum PoS ?

①

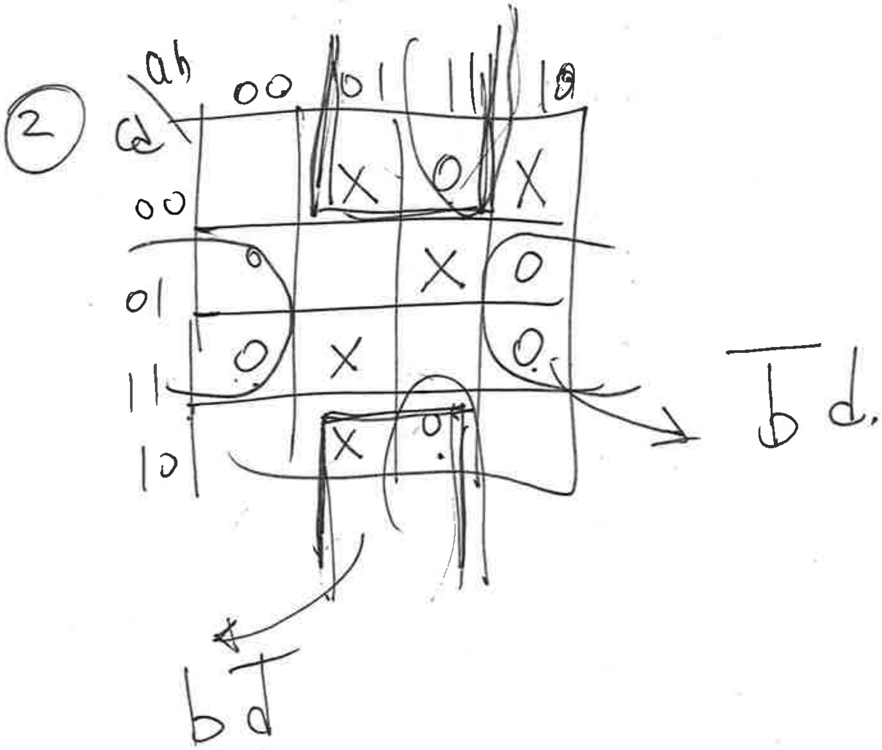
cd \ ab	00	01	11	10
00	1	X		X
01		1	X	
11		X	1	
10	1	X		1

$\rightarrow \bar{b}\,\bar{d}$

$\rightarrow bd$

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

(2)



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

$$f = \bar{\bar{f}} = \overline{\left(\bar{b}d + b\bar{d}\right)}$$

$$= \left(b + \bar{d}\right)\left(\bar{b} + d\right)$$

oct 22, 2020

EEE/CSE 120 : Design sequential Finite state Machines

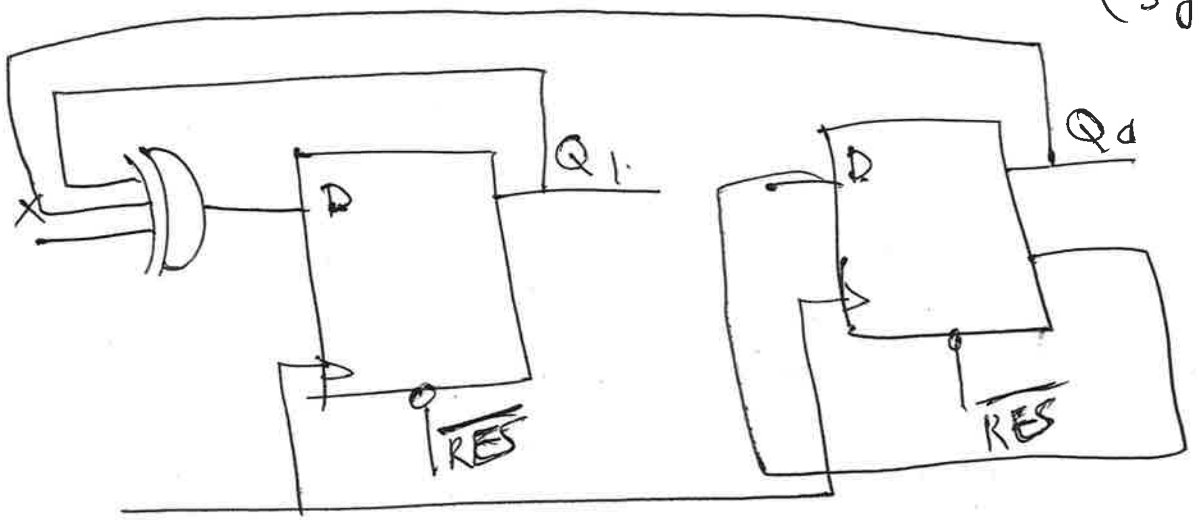
- office Hrs T/TH 9:30-10:15 AM
- Midterm redo for extra credit up to 10 points
  starting today at 6pm
  Till Friday (tomorrow) at 5:59 pm.

- HW 5 is due on oct 29

Sequential Circuit

↳ A bunch of ffs together.

↳ all FFs get the same clks at the same time (synchronous)



⇝ ("Counter")

$$\begin{cases} Q_1^+ = X \oplus Q_1 \oplus Q_0 \\ Q_0^+ = \overline{Q_0} \end{cases}$$

Designing a Sequential FSM :

① State definition table.

state : Any combination of "0"s and "1"s

3 FFs $\rightsquigarrow$ $2^3 = 8$ states

* Sometimes we don't use some of these states (Don't Cares)

② Draw state transition diagram.

↳ graphically represents how one state goes to another state encountering a clock pulse.

(3) State transition table

↳ tells us if these are inputs and current states where we would go next!

(4) Design the next state logic.

Example: Build a counter that counts down from $\underbrace{3 \text{ to } 0}_{\text{Decimal \#}}$ stopping at 0

3, 2, 1, 0, 0, . . .

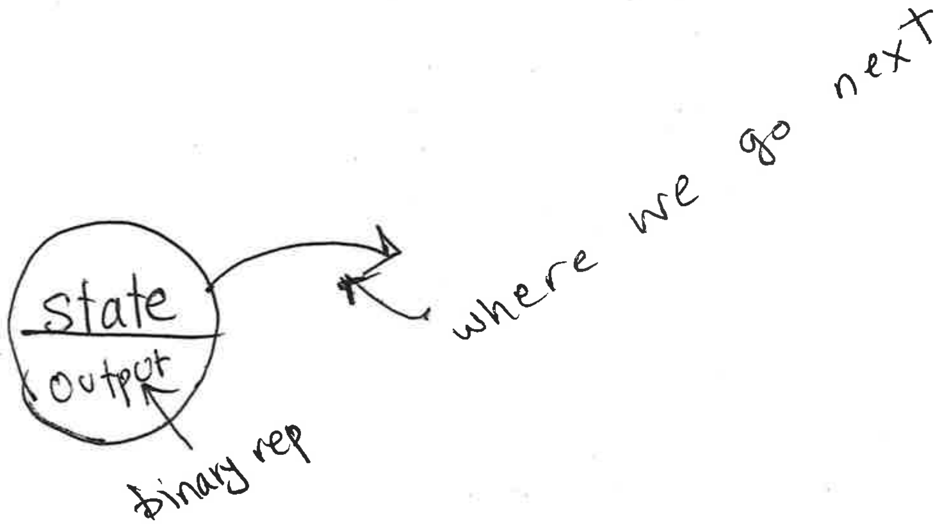
1

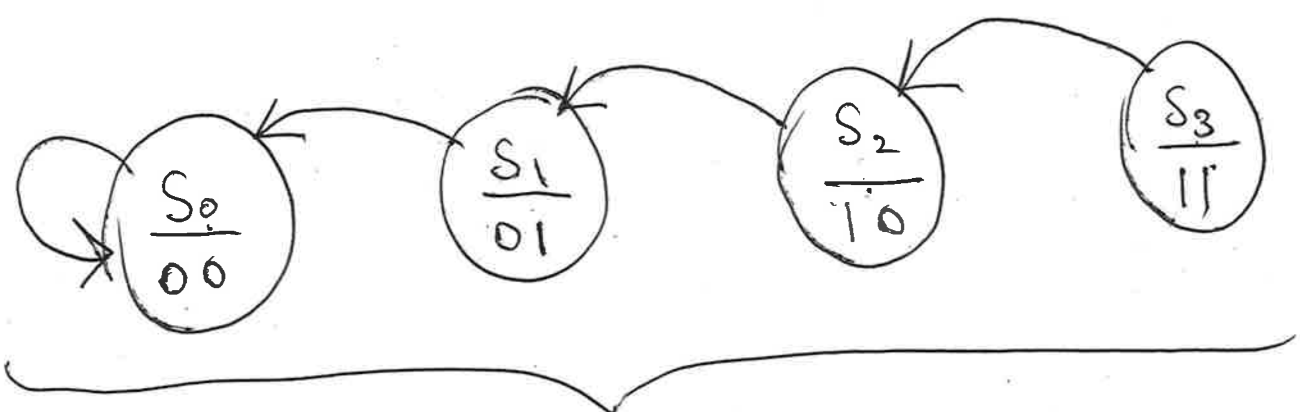
States	Definition	Binary rep. of states
S_0	Count = 0	0 0
S_1	Count = 1	0 1
S_2	Count = 2	1 0
S_3	Count = 3	1 1
		Q_1 Q_0

Two digits needed.

state def. table

② State transition diagram.



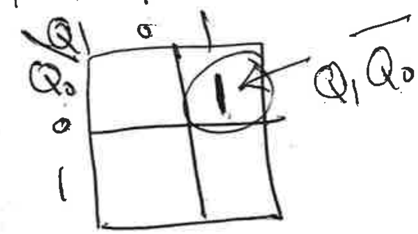


State transition diagram.

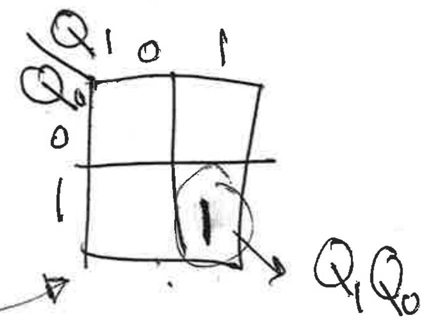
③ State transition table

state	Q_1	Q_0	Q_1^+	Q_0^+	next state
S_0	0	0	0	0	S_0
S_1	0	1	0	0	S_0
S_2	1	0	0	1	S_1
S_3	1	1	1	0	S_2

K-MAP for Q_0^+

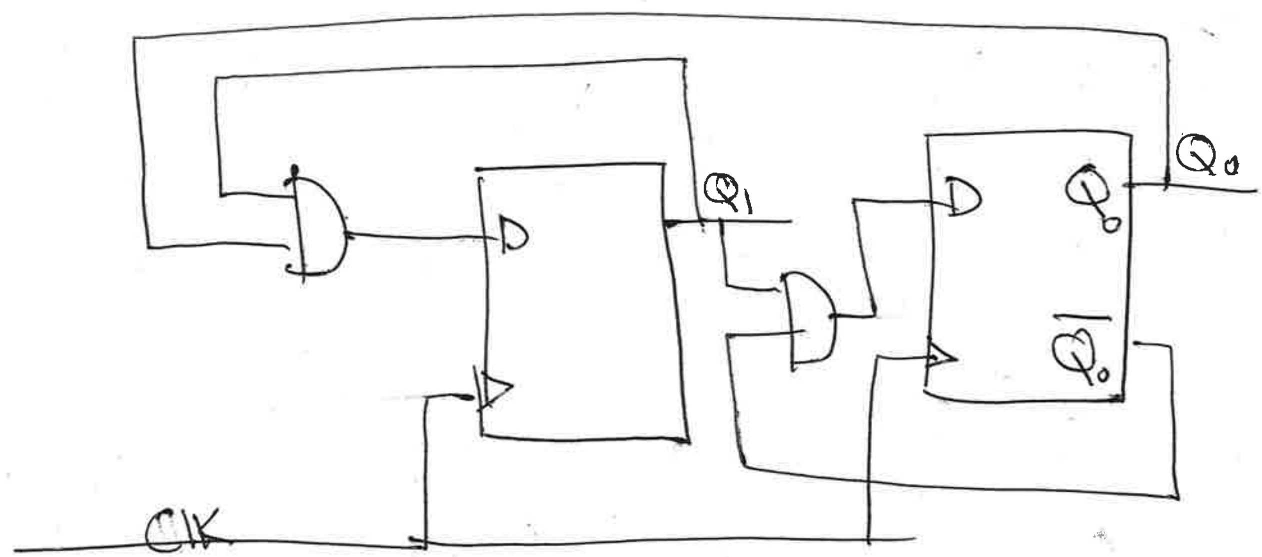


K-Map for Q_1^+



④ Design the circuit

$$\begin{cases} Q_1^+ = Q_1 Q_0 \\ Q_0^+ = Q_1 \overline{Q_0} \end{cases}$$



Example: Build a circuit that can count up / down

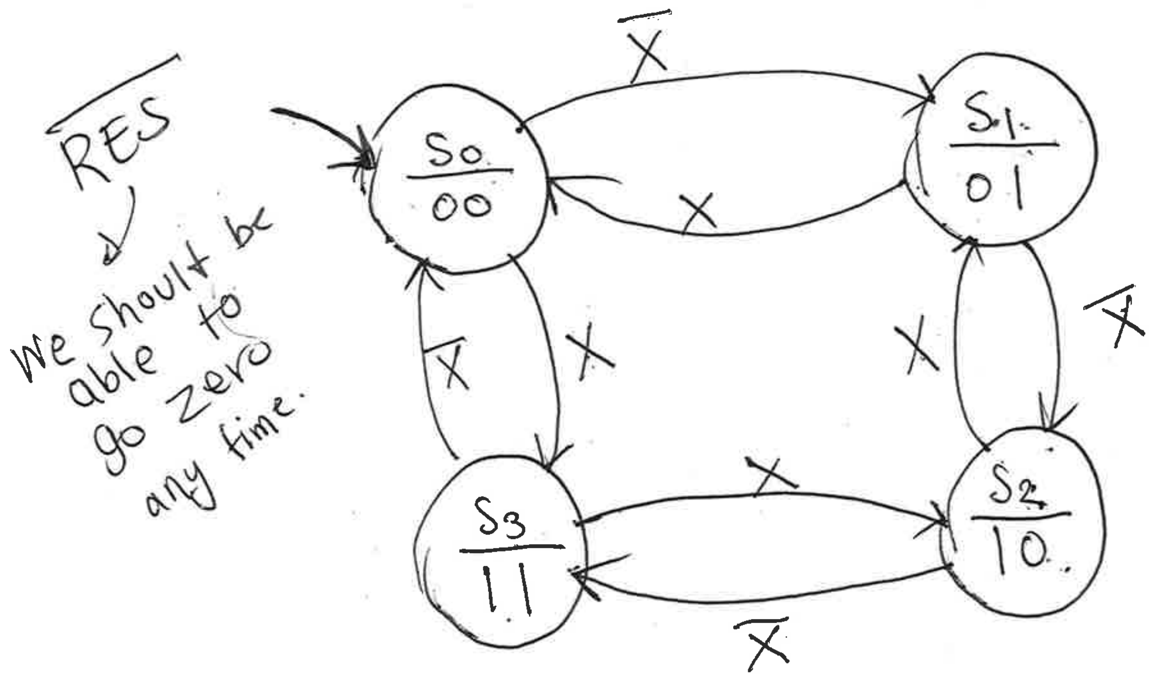
① State def. table

States	Def	binary rep
S_0	count = 0	0 0
S_1	count = 1	0 1
S_2	count = 2	1 0
S_3	count = 3	1 1

Input ⇝ $X = 0$: Counting up

$X = 1$: Counting down

(2) Draw transition diagram.



- prob. def: $3 \xrightarrow[\text{up}]{\text{counting}} 0$
- prob. def.: $0 \xrightarrow[\text{down}]{\text{counting}} 3$

(3) State transition table.

States	input X	Q_1	Q_0	Q_1^+	Q_0^+
S_0	0	0	0	0	1
S_0	1	0	0	1	1
S_1	0	0	1	1	0
S_1	1	0	1	0	0
S_2	0	1	0	1	1
S_2	1	1	0	0	1
S_3	0	1	1	0	0
S_3	1	1	1	1	0

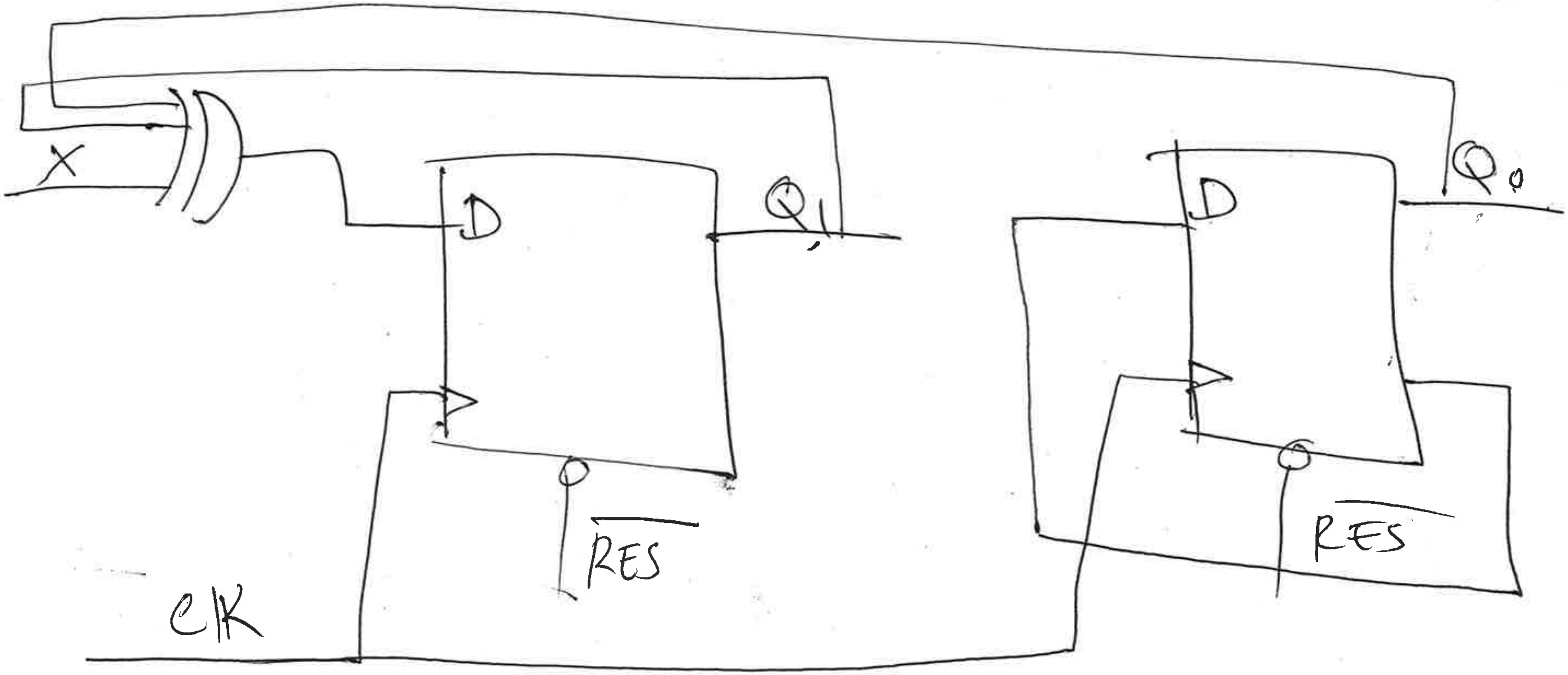
Design :

Q_0 \ XQ_1	00	01	11	10
0		1		1
1	1		1	

For $Q_1^+ = X \oplus Q_1 \oplus Q_0$

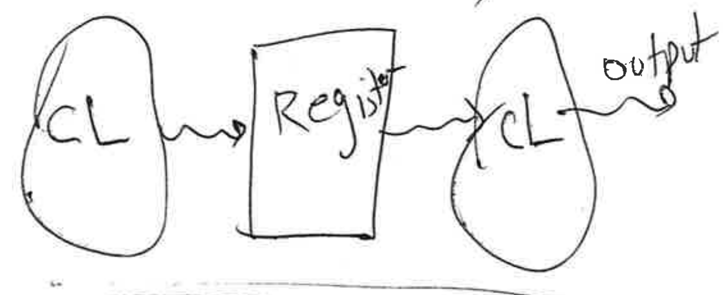
Q_0 \ XQ_1	00	01	11	10
0	1	1	1	1
1				

For $Q_0^+ = \overline{Q_0}$



- $2^{\underbrace{\#FFs}_{\text{Finite}}} = \#\text{of } \underbrace{\text{states}}_{\#\text{Finite}} \Rightarrow$ Called Finite state machine (FSM)

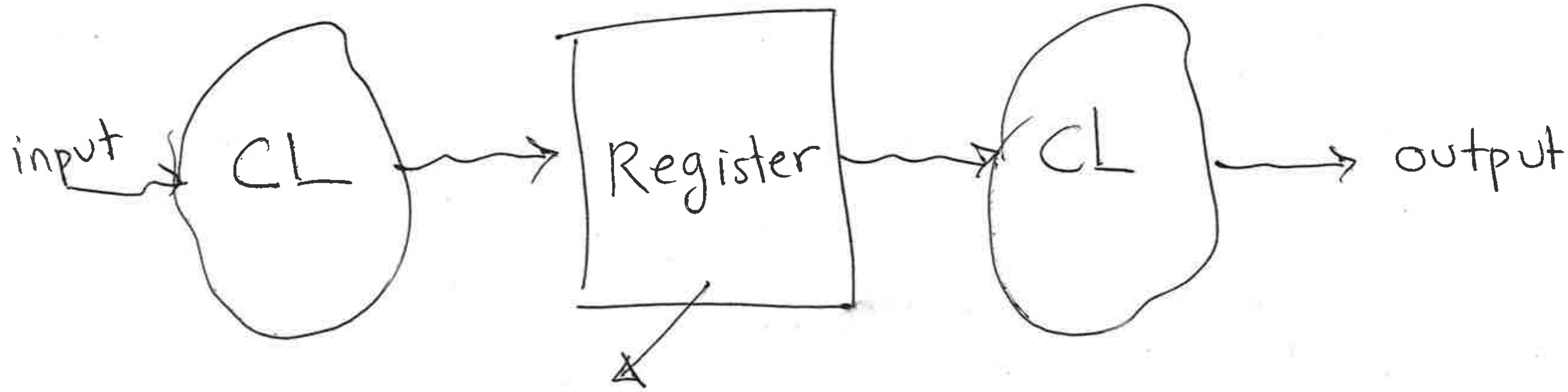
FSM → Moore Machine
FSM → Mealy Machine



EEE/CSE 120 : Finite State Machines

* HW 5 is due oct 29

* Poll → due oct 31
  → Canvas → Quizzes → poll.



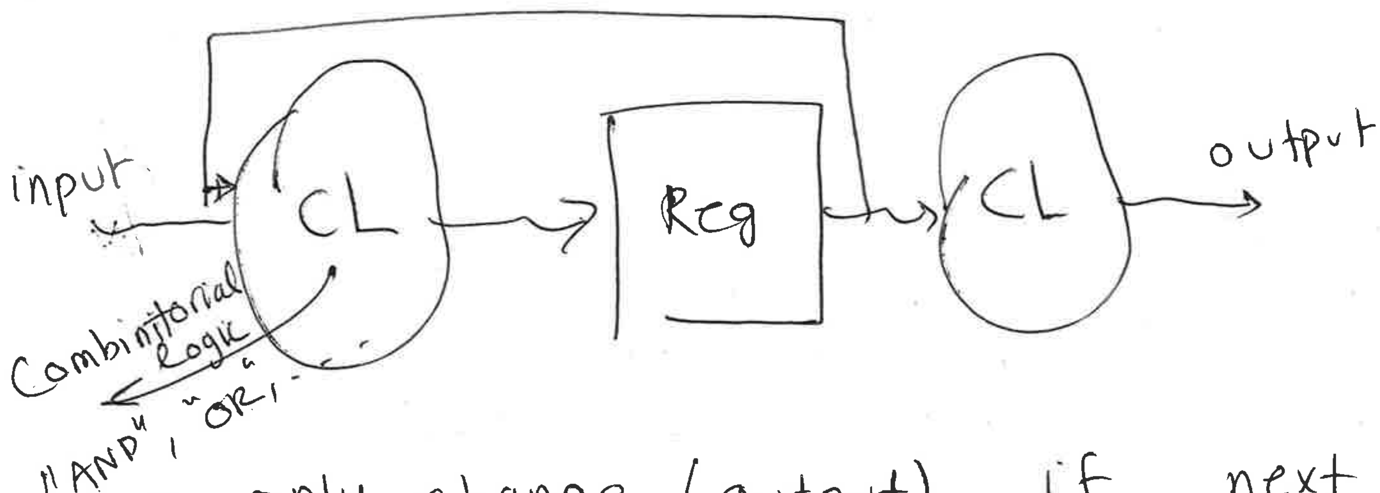
$$2^{\#\text{ of FFs}} = \#\text{ of states}$$

<u>FSM</u> → Moore Machine

→ Mealy Machine

Moore Machine:

A FSM is Moore if there is no direct Connection from input to output.



* only change (output) if next clock happens.
(unless there is a reset)

Example: Design a Moore machine Such that

Key pad w/ $\begin{cases} X=0 \\ X=1 \end{cases}$

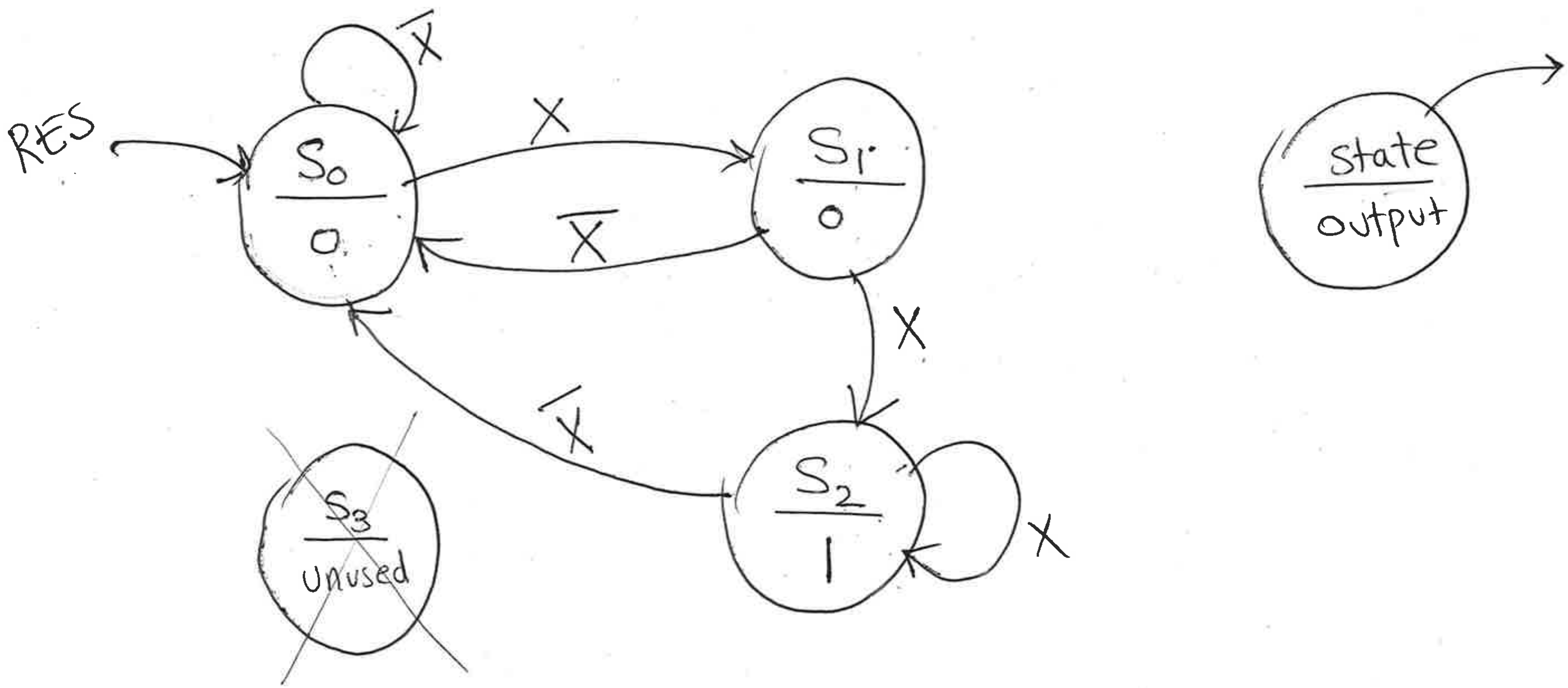
press "1" Twice in a row then the Safe opens up.

① State def. table:

states	name (def)	binary rep.		output
S_0	IDLE	0	0	0
S_1	One - one	0	1	0
S_2	two - Ones	1	0	1
S_3	unused	1	1	X
		Q_1	Q_0	

input: $\begin{cases} X=0 \\ X=1 \end{cases}$

② Draw the state diagram



③ State transition table

states	input=X	Q_1	Q_0	Q_1^+	Q_0^+	Output
S_0	0	0	0	0	0	0
S_0	1	0	0	0	1	0
S_1	0	0	1	0	0	0
S_1	1	0	1	1	0	0
S_2	0	1	0	0	0	1
S_2	1	1	0	1	0	1
S_3	0	1	1	X	X	X
S_3	1	1	1	X	X	X

depends on the current states and not the future states!

④ Design the circuit

For Q_1^+

Q_0 \ XQ_1	00	01	11	10
0			1	
1		X	X	1

XQ_1

XQ_0

$$Q_1^+ = XQ_0 + XQ_1$$

For Q_0^+

Q_0 \ XQ_1	00	01	11	10
0				1
1		X	X	

$$Q_0^+ = X\overline{Q}_1\overline{Q}_0$$

For output

Q_0 \ XQ_1	00	01	11	10
0		1	1	
1		X	X	

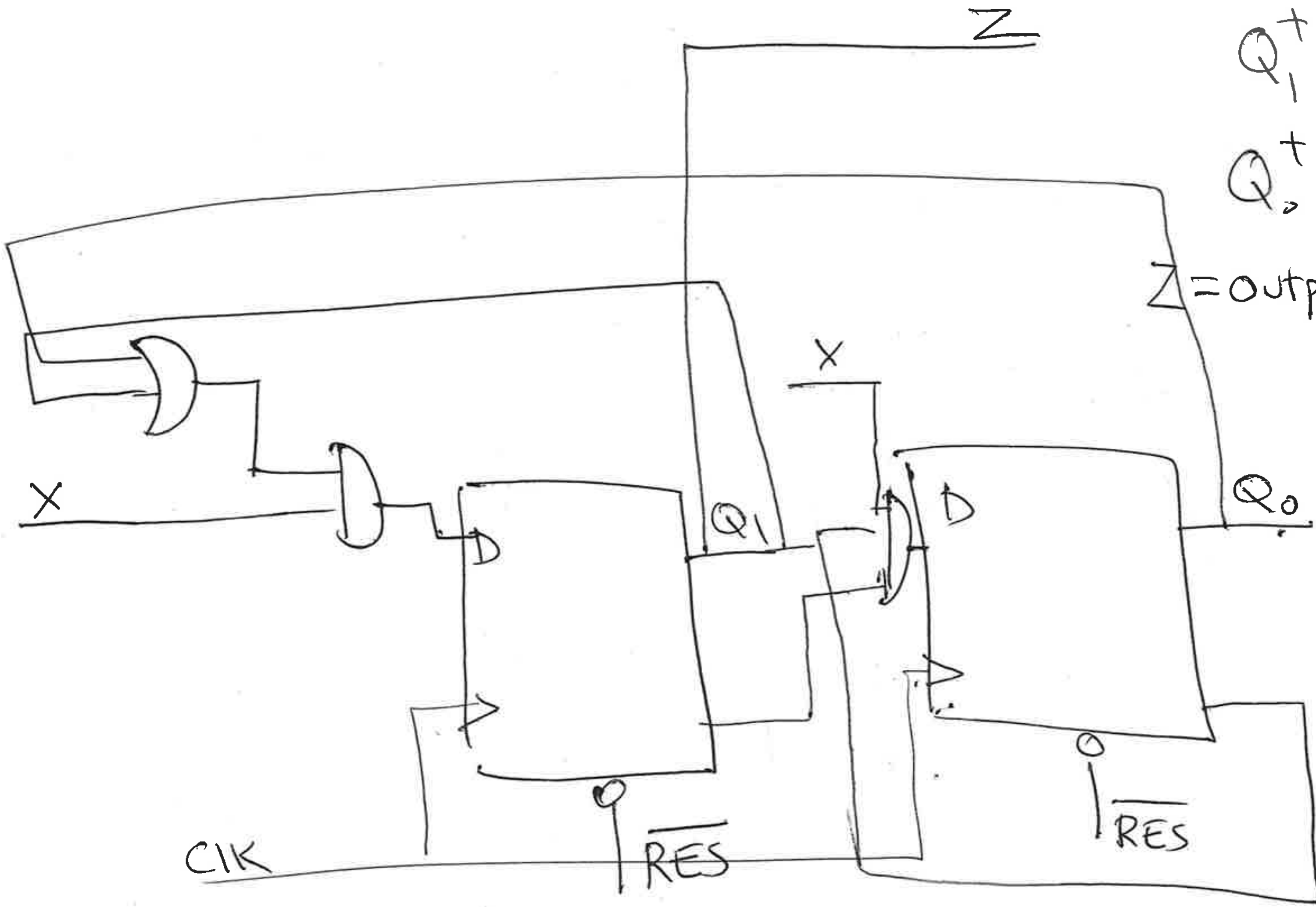
$\underbrace{\text{output} = Q_1}$

is only a funct. of current state Q_0, Q_1.

$$Q_1^+ = X Q_0 + X Q_1$$

$$Q_0^+ = X \overline{Q_1}\,\overline{Q_0}$$

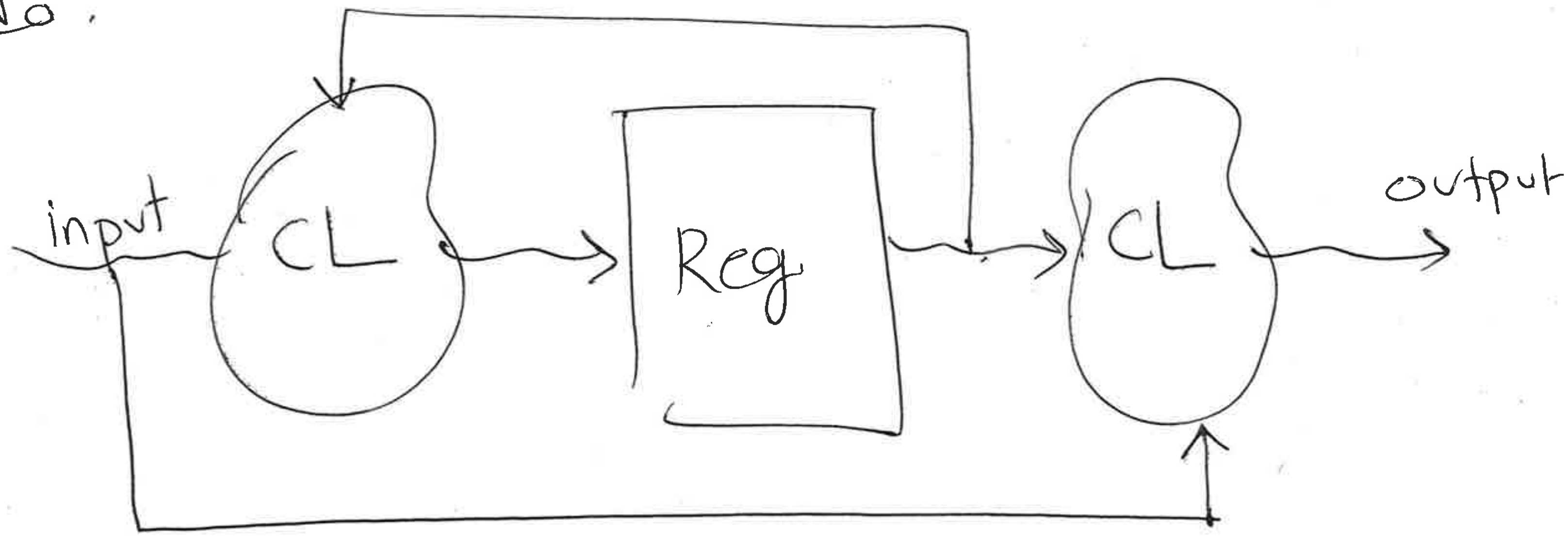
$$Z = \text{output} = Q_1$$



Mealy Machine :

→ Is a FSM that the output depends on the states of the flip flops as well as the input.

- Mealy machine can do whatever moore machines can do.



Moore Machine : Input change ⇒ for output to change we have to wait for the next clock

Mealy Machine : Input change ⇒ output change.

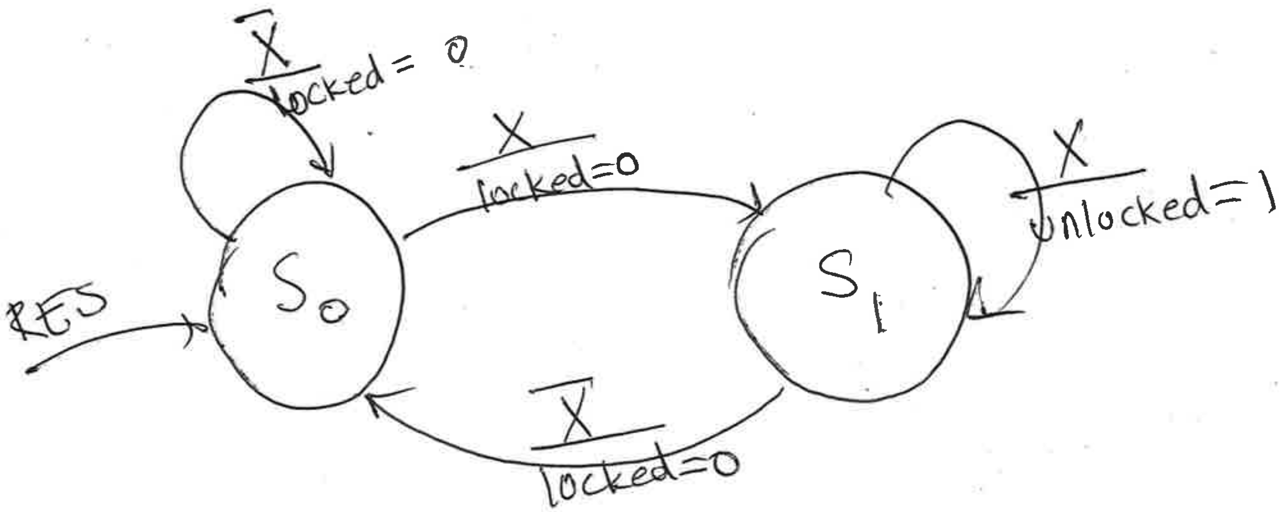
<u>Example</u>: Key pad $\begin{cases} X=0 \\ X=1 \end{cases}$, we press two ones in a row, then we open up the safe.

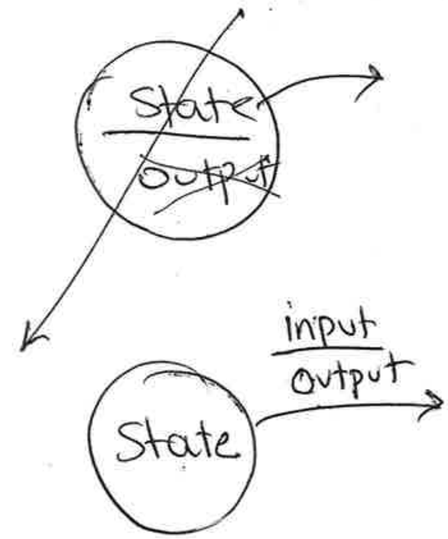
Design a Mealy Machine.

(1) State def. table.

States	State def	binary rep.
S_0	IdLE	0
S_1	Count the first one	1

② Draw state diagram





- Moore output is the output of flip flop and the output state

- Mealy machine the output is a function of inputs.

③ State transition table

input X	Q	Q^+	output
0	0	0	0
0	1	0	0
1	0	1	0
1	1	1	1

output → This depends on input

④ Design the circuit

For Q^+

Q \ X	0	1
0		1
1		1

→ $Q^+ = X$

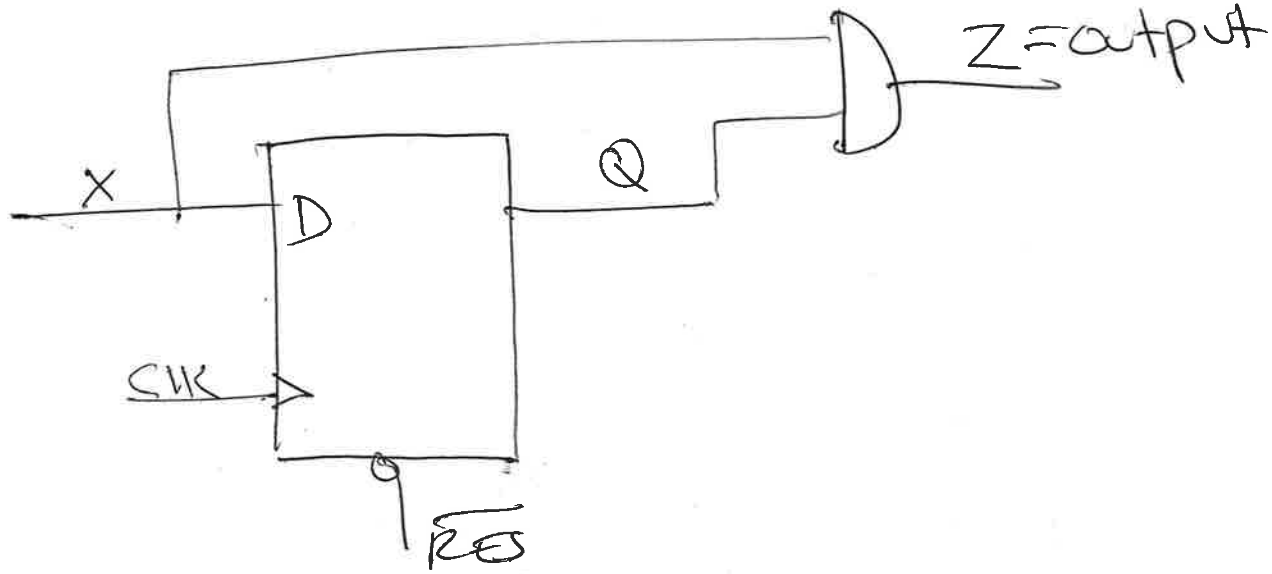
For output

Q \ X	0	1
0		
1		1

output $= XQ$

$Q^+ = X$

$Z = \text{output} = XQ$

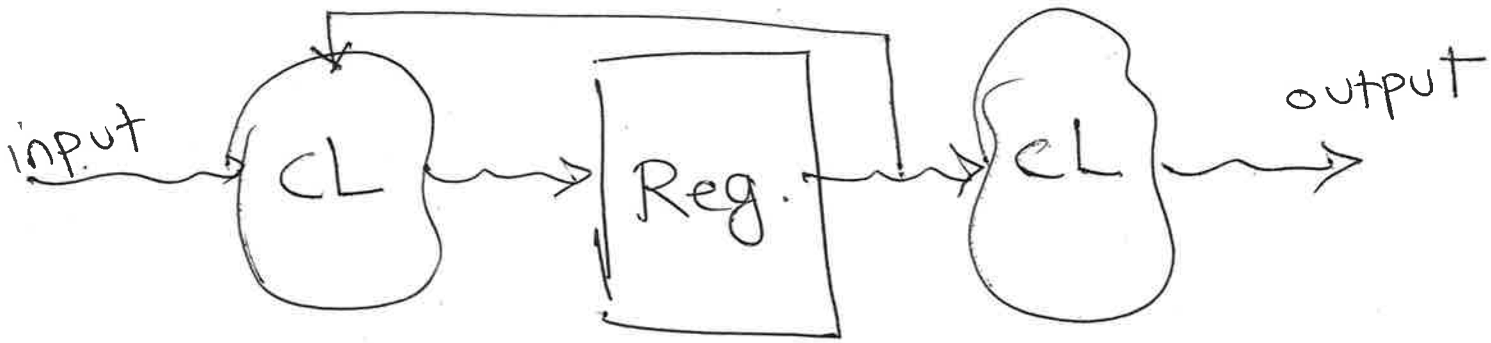


- There is timing issue → synchronization ← Adding one ff.

EEE/CSE 120 : Moore and Mealy Machines Oct 29, 2020

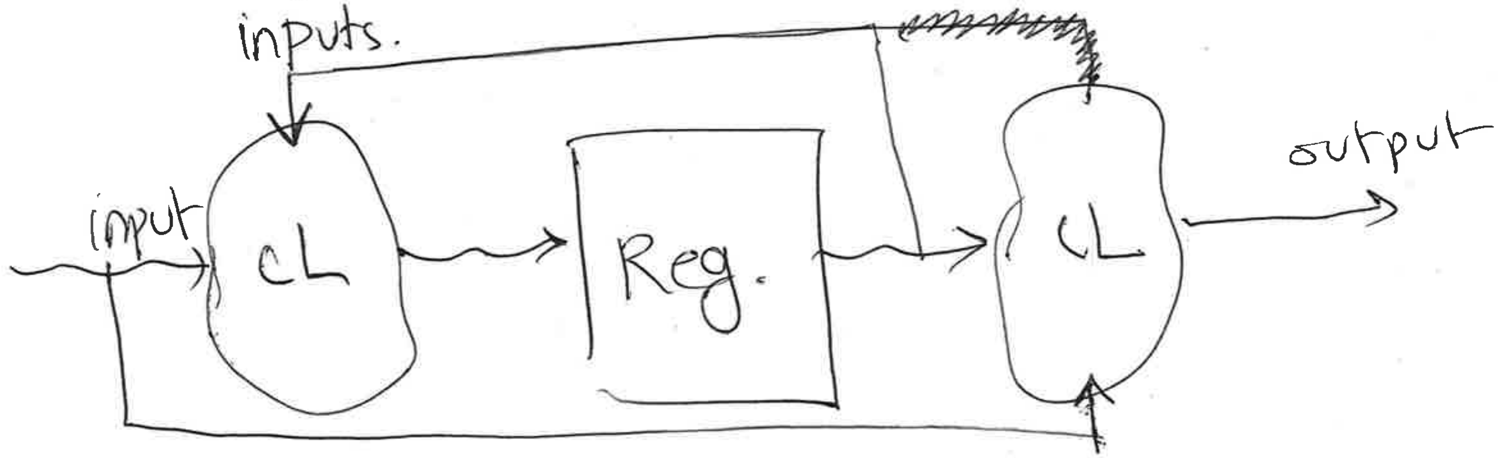
① Moore Machine :

FSM, there is no direct relationship between input & output



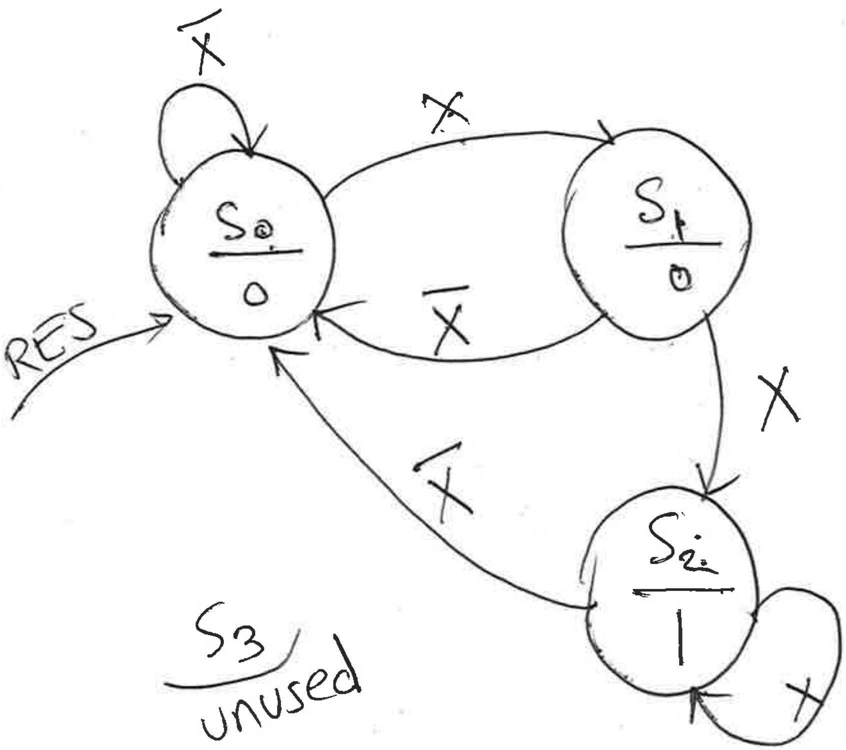
② Mealy Machine :

FSM : output change depends on both states & inputs.



Review of the example: Keypad $\begin{cases} X=0 \\ X=1 \end{cases}$, safe opens up if we have two "1"s in a row.

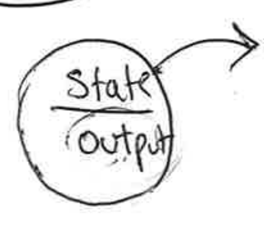
Moore Machine



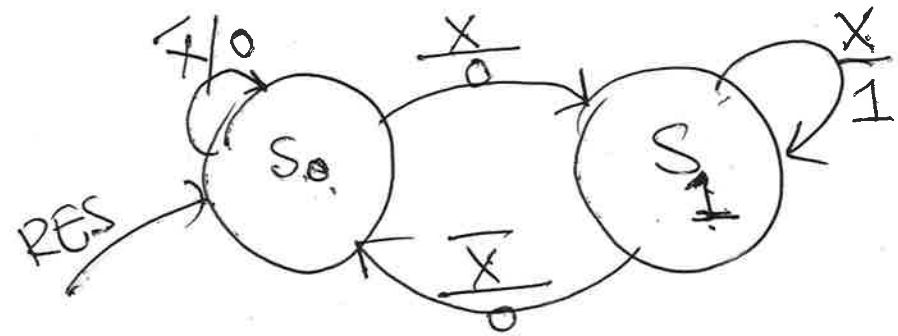
S_3 unused

$\underbrace{\hspace{6em}}$

2 Flip flops

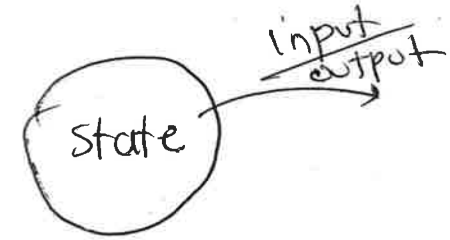


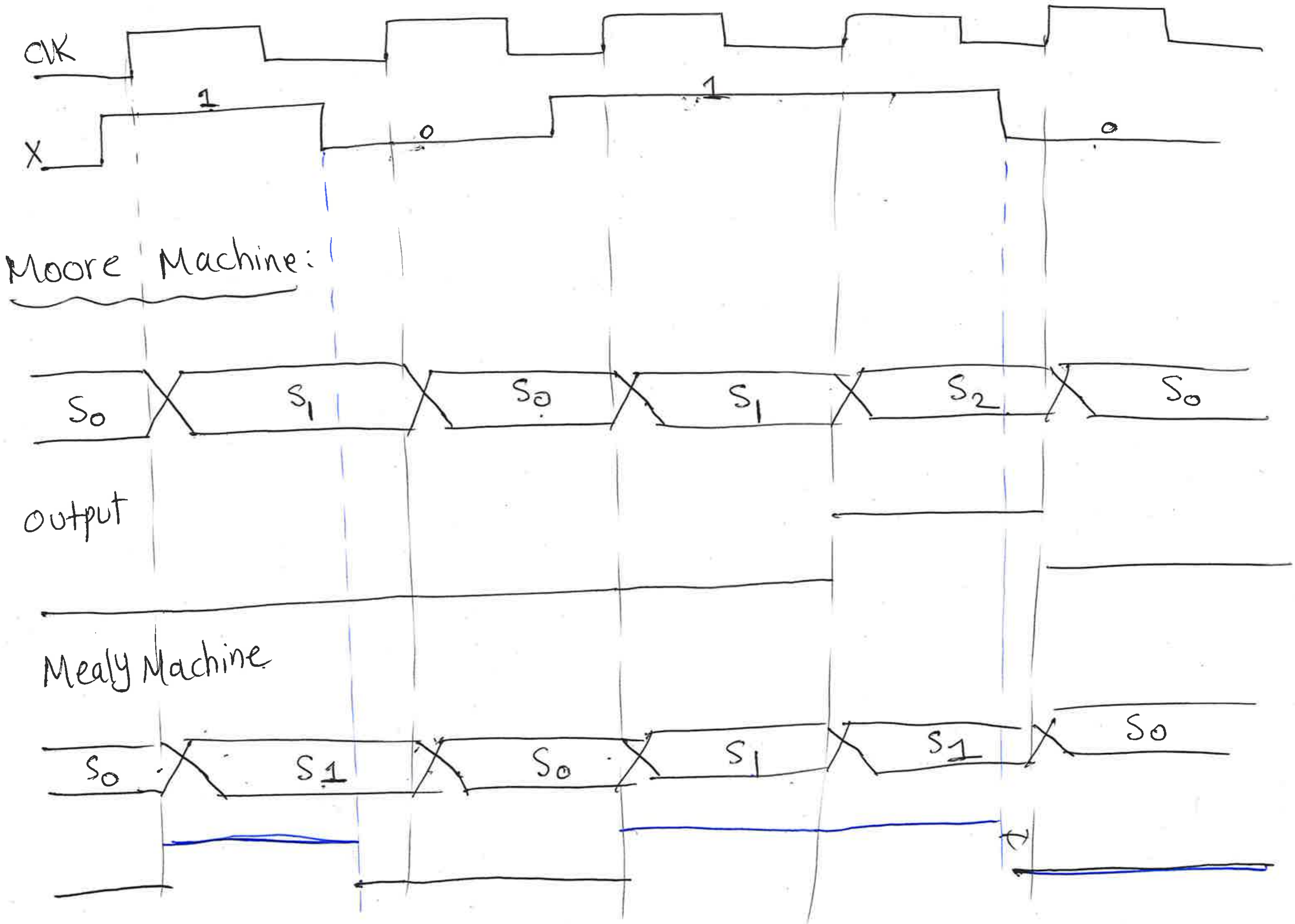
Mealy Machine



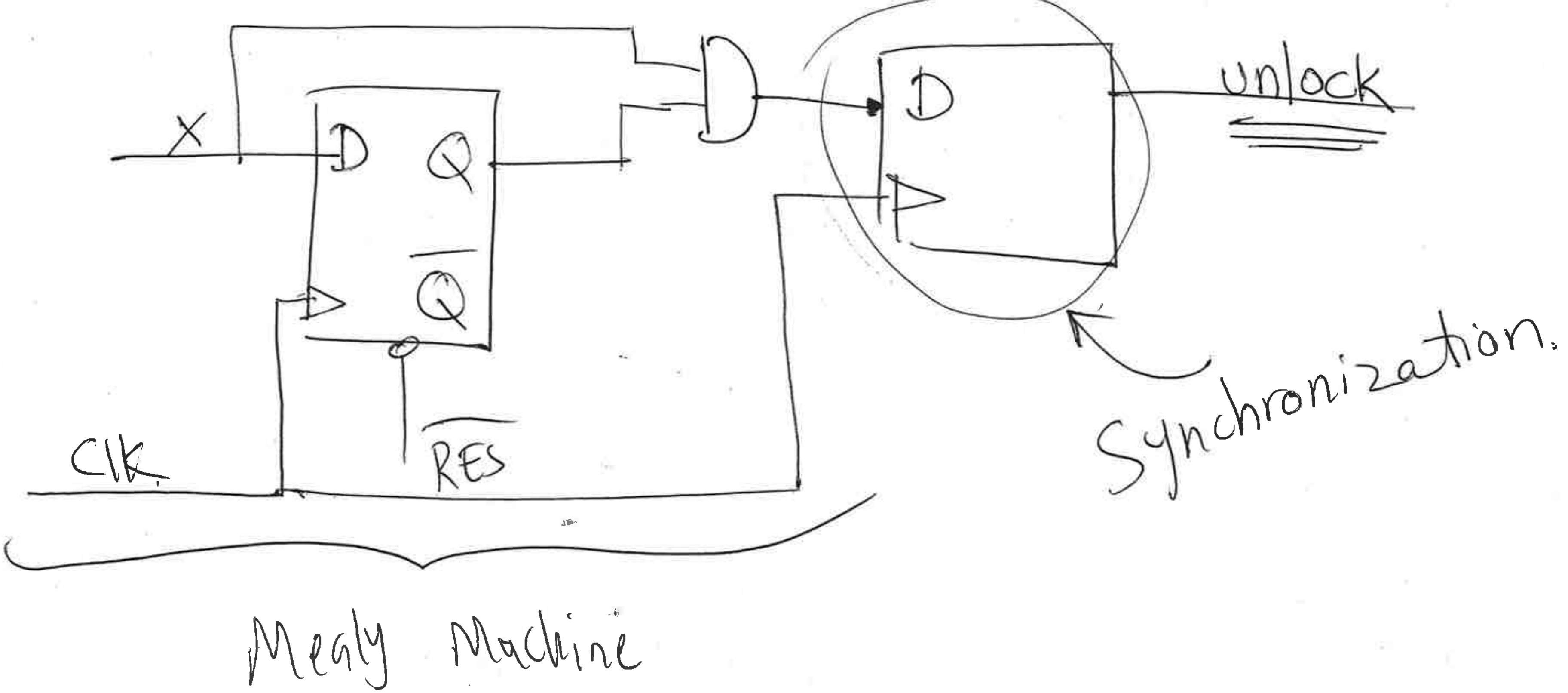
$\underbrace{\hspace{6em}}$

1 Flip flop.




CLK
X
1
0
1
0
Moore Machine:
S_0
S_1
S_0
S_1
S_2
S_0
output
Mealy Machine
S_0
S_1
S_0
S_1
S_1
S_0

- Mealy Machines use less number of ffs.,
  - however, we often have timing issue!
    - Asynchronous.
      - by adding another FF.
        - Makes analysis much more difficult.



Example: Lets say we have sensor that Can tell us if someone enters or exists

inputs = $\begin{cases} e & \text{entering} \\ x & \text{exiting} \end{cases}$

- Max # of people in the room = 3
- only one person at a time can leave /enter.
- If the room is full, red light on

  If the room is empty, light is off

Q: Design a light Controller that can do the above.

① state def. table:

state	def.	binary rep.	
S_0	light off & room empty	0	0
S_1	light on & one person	0	1
S_2	light on & two people	1	0
S_3	light on & 3 people	1	1
		Q_1	Q_0

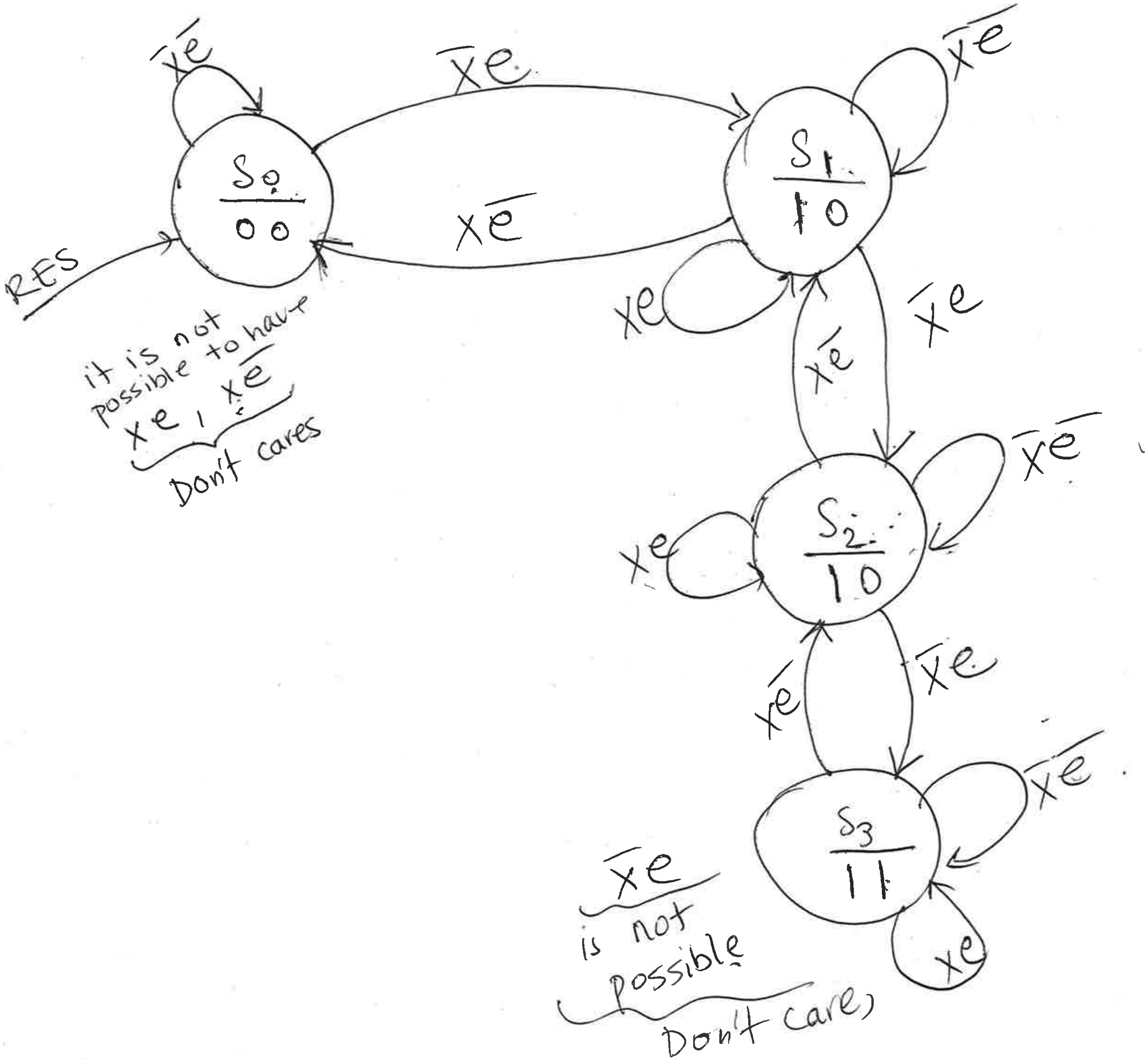
inputs

$e \in \{0,1\}$ $\quad x \in \{0,1\}$

outputs

light on/off $\quad$ light red/not

② state diagram



③ State transition table.

	Q_1	Q_0	X	E	Q_1^+	Q_0^+	L (light)	R (red light)
state S_0	0	0	0	0	0	0	0	0
	0	0	0	1	0	1	0	0
	0	0	1	0	X	X	X	X
	0	0	1	1	X	X	X	X
S_1	0	1	0	0	0	1	1	0
	0	1	0	1	1	0	1	0
	0	1	1	0	0	0	1	0
	0	1	1	1	0	1	1	0
S_2	1	0	0	0	1	0	1	0
	1	0	0	1	1	1	1	0
	1	0	1	0	0	1	1	0
	1	0	1	1	1	0	1	0
S_3	1	1	0	0	1	1	1	1
	1	1	0	1	X	X	X	X
	1	1	1	0	1	0	1	1
	1	1	1	1	1	1	1	1

output only depends on current state "NOT" the next state

④ For Q_1^+

$XE \backslash Q_1Q_0$	00	01	11	10
00			1	1
01		1	X	1
11	X		1	1
10	X			

$Q_1\bar{X}$

Q_1E

$Q_0\bar{X}E$

$$\boxed{Q_1^+ = Q_1\bar{X} + Q_1E + Q_0\bar{X}E}$$

For Q_0^+

$XE \backslash Q_1Q_0$	00	01	11	10
00		1	1	
01	1		X	1
11	X	1	1	
10	X		1	1

Q_1Q_0

$\bar{Q}_0\bar{X}E$

$Q_1X\bar{E}$

$\bar{Q}_0XE$

$$\boxed{Q_0^+ = Q_1Q_0 + \bar{Q}_0XE + Q_1X\bar{E} + \bar{Q}_0\bar{X}E}$$

For L

$XE \backslash Q_1Q_0$	00	01	11	10
00		1	1	1
01		1	X	1
11	X	1	1	1
10	X	1	1	1

Q_0

Q_1

$$\boxed{L = Q_1 + Q_0}$$

For R

$XE \backslash Q_1Q_0$	00	01	11	10
00			1	
01			X	
11	X		1	
10	X		1	

$$\boxed{R = Q_1Q_0}$$

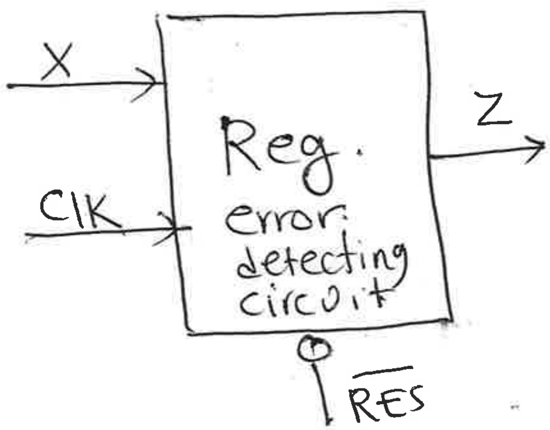
Lecture 21

EEE / CSE 120 : Capstone project (NOV 3, 2020)

- Office Hrs { T : 9:30 - 10:15 AM (only this week)
  TH 3:15 - 4:00 pm

- Capstone project is uploaded
- HW6 is uploaded
- Get ready for Quiz 2
  - Registers
  - timing diagram.
- Midterm is graded and grades will be uploaded soon.

Example : Design a Mealy machine for an error detector.



- Single input
- Single output.

output $Z = 1$ (error)

error : Two zeros in a row "00" or three ones in a row "111"

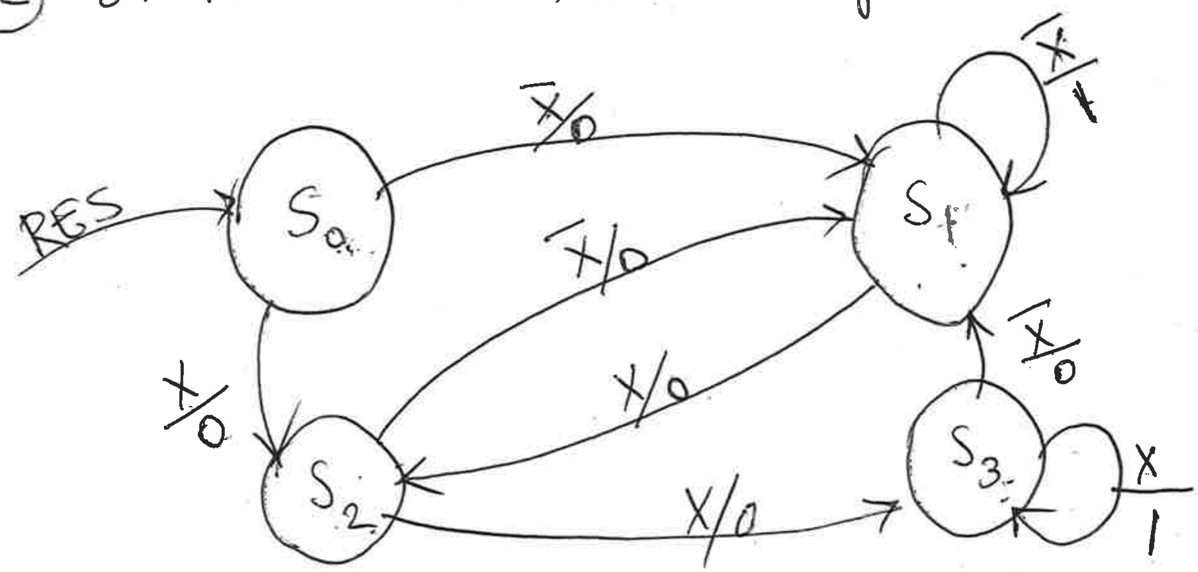
Example :

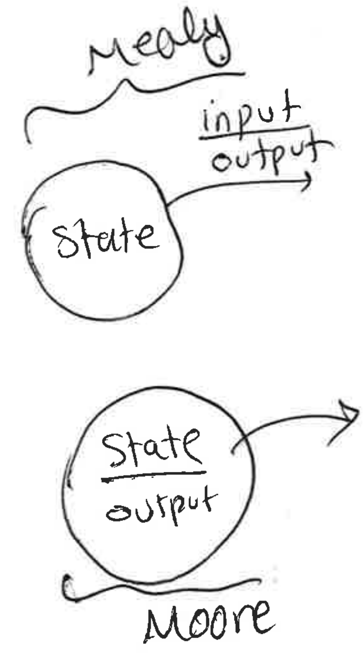
X	0	0	1	1	1	1	1	0	0	0	1	1	1	0	0
Z	0	1	0	0	1	1	1	0	1	1	0	0	1	0	1

① State def. table

State	Def.	binary rep.	
		Q_1	Q_0
S_0	IDLE	0	0
S_1	one-zero	0	1
S_2	one-one	1	0
S_3	two-ones	1	1

② State transition diagram.





③ state transition table

	Q_1	Q_0	X	Q_1^+	Q_0^+	Z = output
S_0	0	0	0	0	1	0
	0	0	1	1	0	0
S_1	0	1	0	0	1	1
	0	1	1	1	0	0
S_2	1	0	0	0	1	0
	1	0	1	1	1	0
S_3	1	1	0	0	1	0
	1	1	1	1	1	1

④ Design

K-Map for Q_1^+

$X \backslash Q_1Q_0$	00	01	11	10
0				
1	1	1	1	1

$$Q_1^+ = X$$

K-Map Q_0^+

$X \backslash Q_1Q_0$	00	01	11	10
0	1	1	1	1
1			1	1

$\overline{X}$

Q_1

$$Q_0^+ = Q_1 + \overline{X}$$

For output Z

$X \backslash Q_1Q_0$	00	01	11	10
0		1		
1			1	

$\overline{Q}_1 Q_0 \overline{X}$

$Q_1 Q_0 X$

$$Z = \overline{Q}_1 Q_0 \overline{X} + Q_1 Q_0 X$$

$Q_1^+ = X$

$Q_0^+ = Q_1 + \overline{X}$

$Z = \overline{Q}_1 Q_0 \overline{X} + Q_1 Q_0 X$