Chapter 7: A Primer On Digital Logic

Basic Computer Architecture

Outline

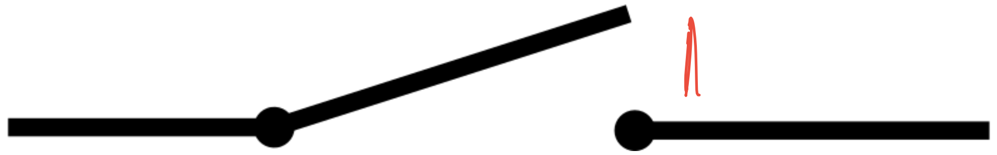
* Transistors and Gates
* Combinational Logic
* Sequential Logic
* SRAM/ DRAM Cells

Disclaimer: This chapter is only to get a high level overview …

* We assume some background in logic gates, transistors, combinational and sequential logic
* The aim of this chapter is to only provide a high level overview
* For a deeper understanding consult any of the classic textbooks on digital logic

Atoms and Molecules of Circuits

The Transistor

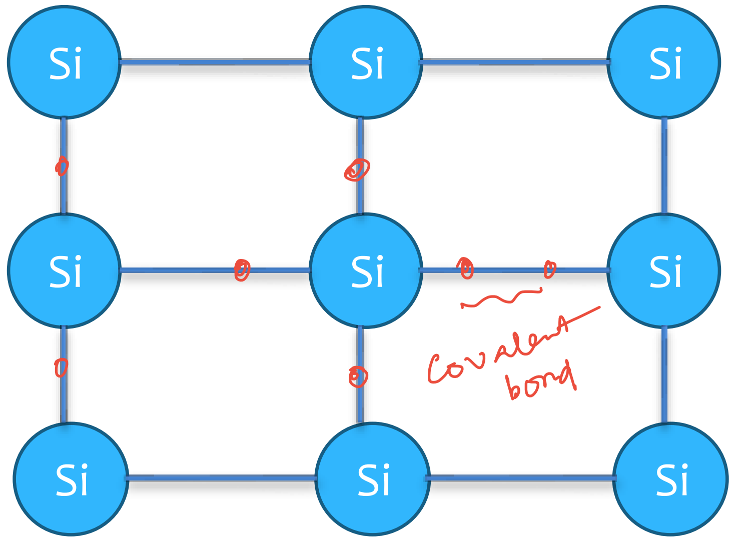


- It is just a **switch**.
- It is either switched on (current can flow), or switched off (no current flow)

How is a Transistor Made?

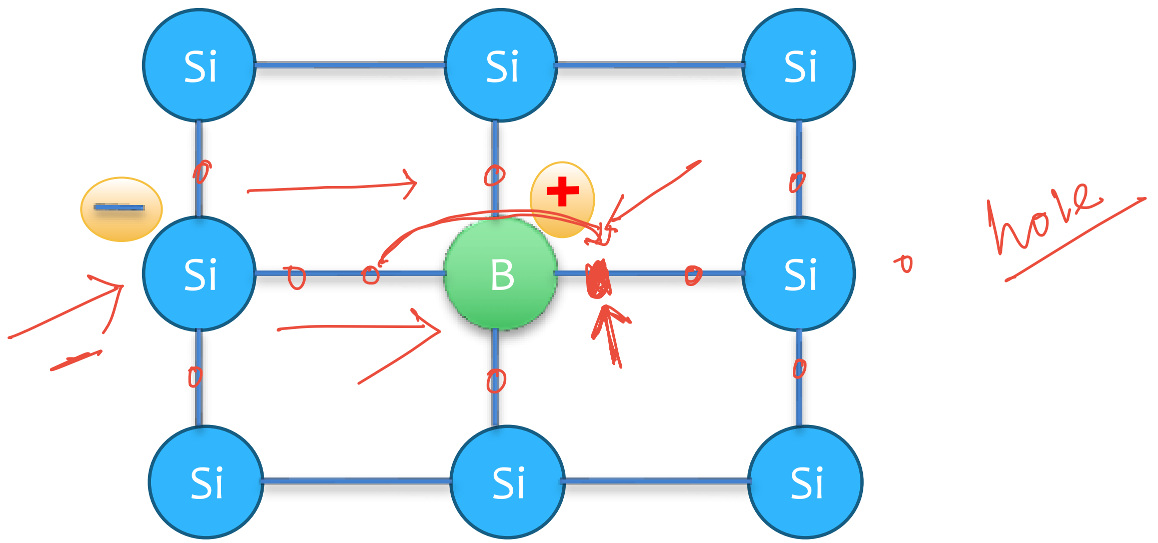
- It is made of Silicon
  - Silicon is a semi-conductor (neither conductor or insulator).
  - We can change its properties:
  - Add a little bit of impurities → doping
- Dope it with Group III elements (3 valence electrons)
  - Boron, Aluminum and Gallium
  - It is called a p-type semiconductor
- Or, dope it with Group V elements (5 valence electrons)
  - Phosphorus or Arsenic
  - It is called a **n-type** semiconductor

Silicon Lattice



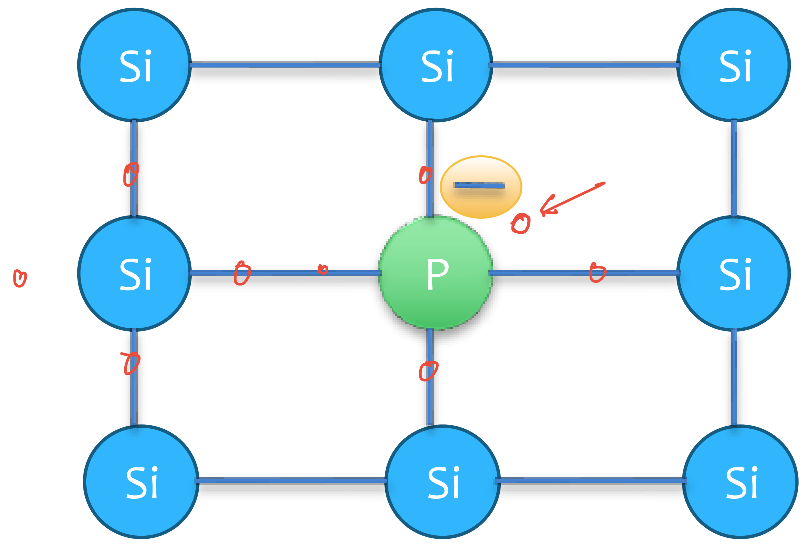
* This is a typical silicon lattice. Each Si atom is connected to 4 other atoms via covalent bond

P-Type Doping



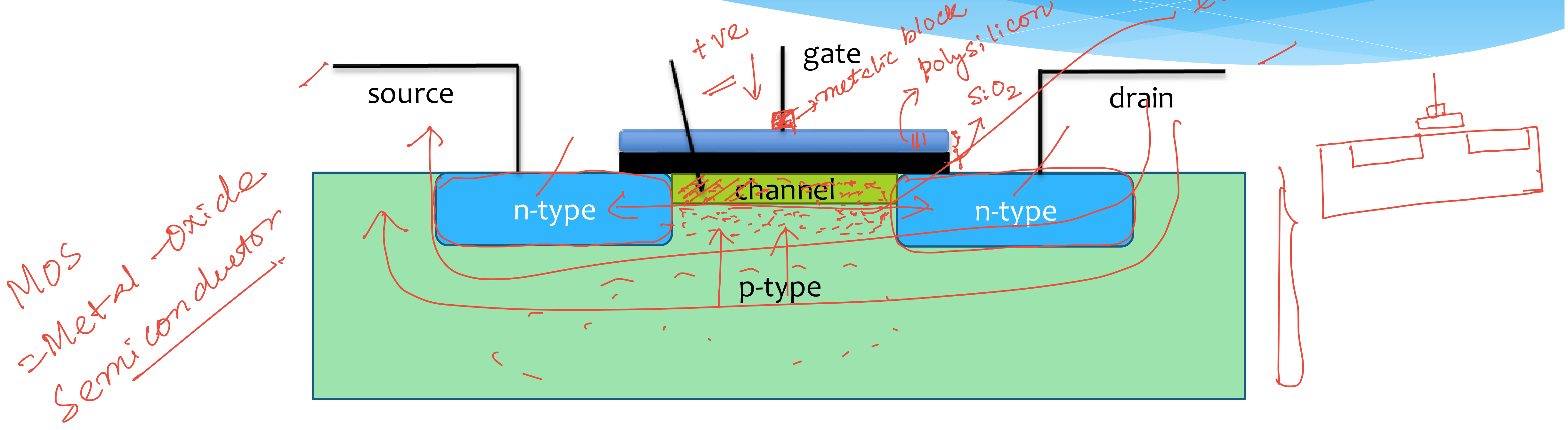
- If there is a Boron atom in the lattice. It will create bonds with the rest of the atoms.
- There will be one less electron (hole)
- Holes can flow. They are associated with +ve charge. (see the animation)

N-Type Doping



* If there is a Phosphorus atom in the lattice. It will create bonds with the rest of the atoms.
* There will be one more electron (electron)
* Electrons can flow. They are associated with -ve charge. (see the animation)

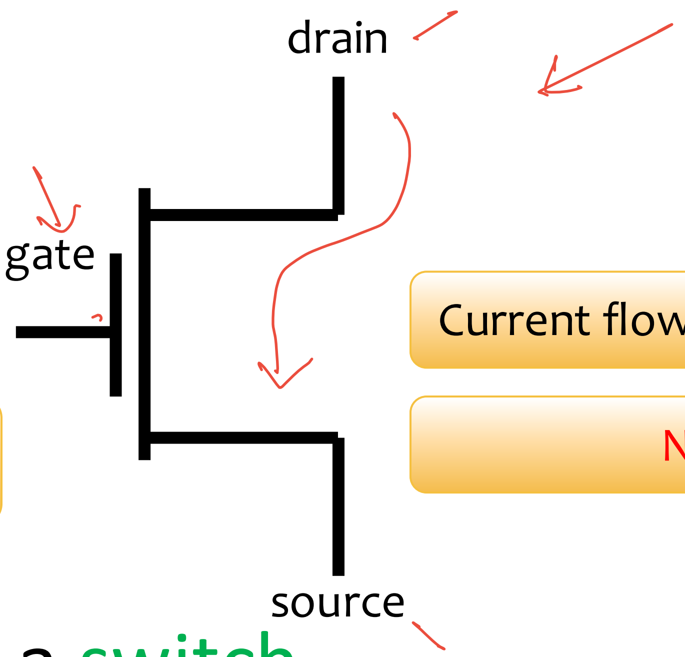
Let us make a transistor.



* NMOS transistor → Put two n-type wells in a p-type substrate
* Now, assume that we apply +ve charge to the gate.
* Channel → This forms because electrons move towards the positive charge. Forms a conductive layer that can conduct current.

NMOS Transistor

Metal-Oxide-Semiconductor



Apply a +ve voltage

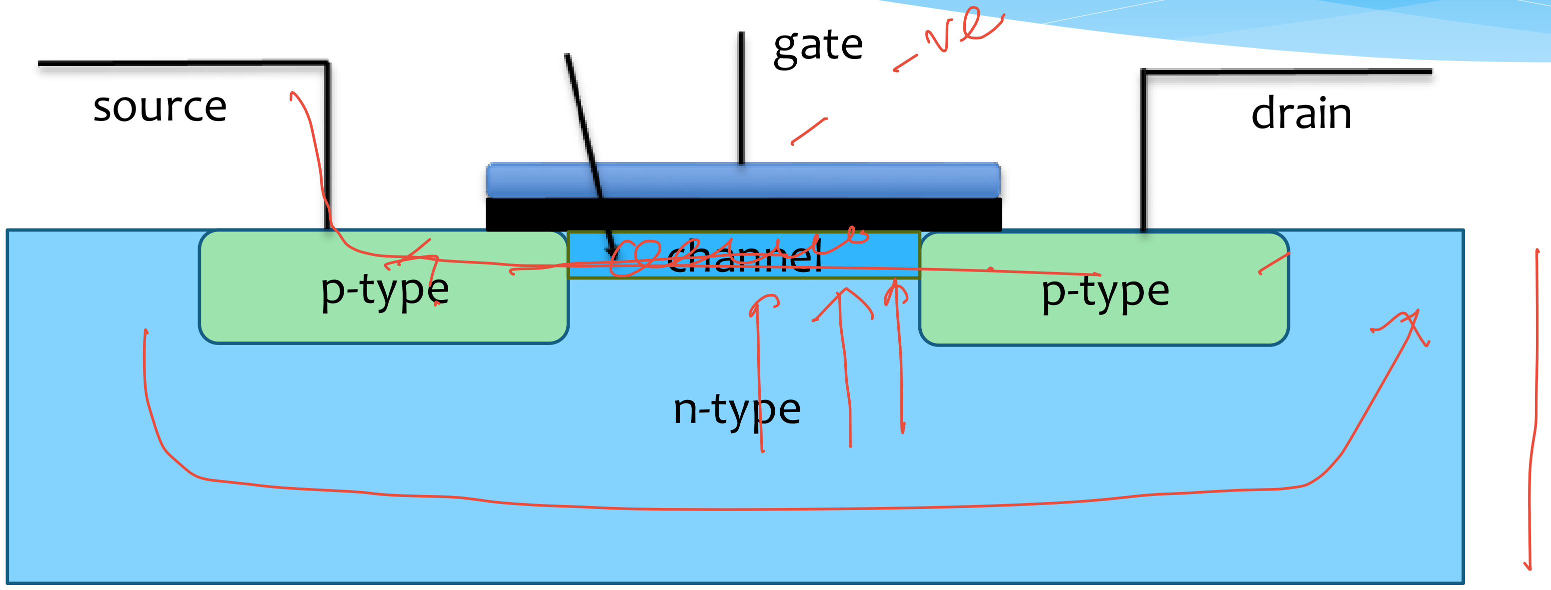
Current flows from the drain to source

Apply a 0 or -ve voltage

No current flow.

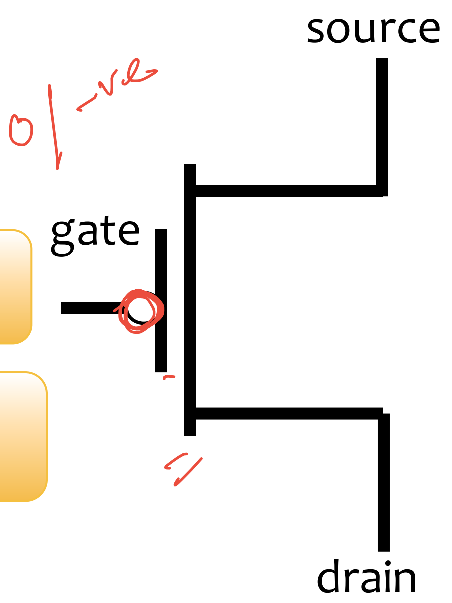
- We thus have a switch
  - Apply a +ve voltage at the gate → make the transistor conduct
  - Apply a –ve or 0 voltage at the gate → transistor is off (no current flow across it)

PMOS Transistor



- PMOS transistor → Put two p-type wells in an n-type substrate
- Now, assume we apply 0 or -ve charge to the gate.
- Channel → This forms because holes move towards the gate. This channel can conduct current.

PMOS Transistor



Apply a 0 or -ve voltage

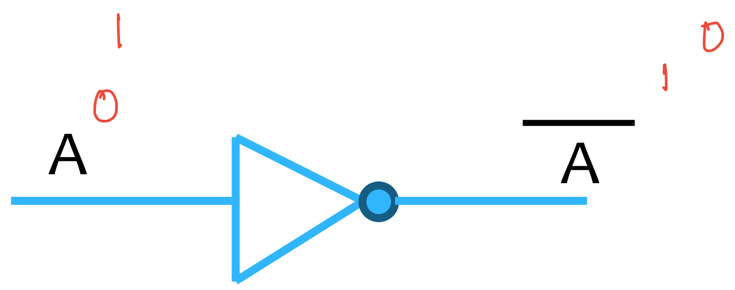
Apply a +ve voltage

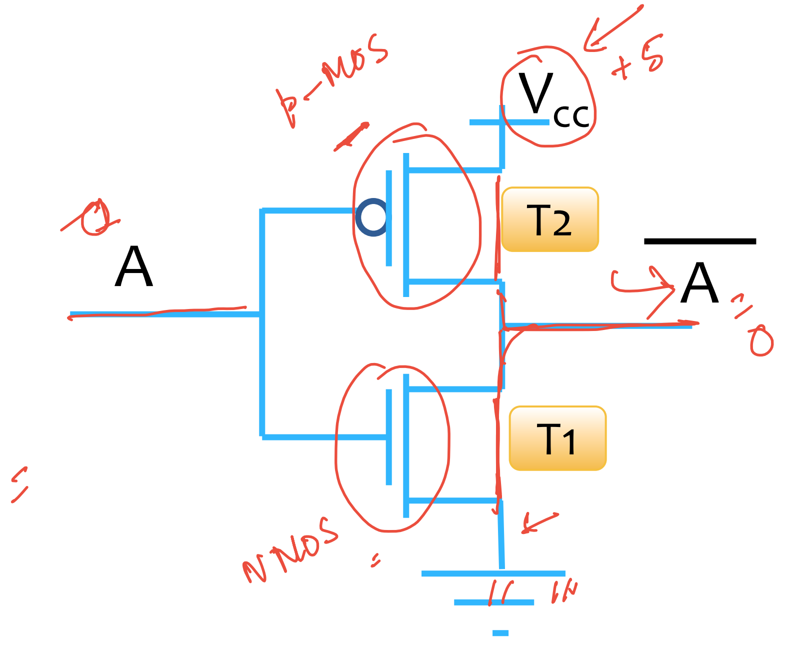
Current flows from the source to the drain

No current flow.

- We thus have a switch
  - Apply a +ve voltage at the gate → transistor is off (no current flow across it)
  - Apply a –ve or 0 voltage at the gate →transistor conducts

Let us make an inverter



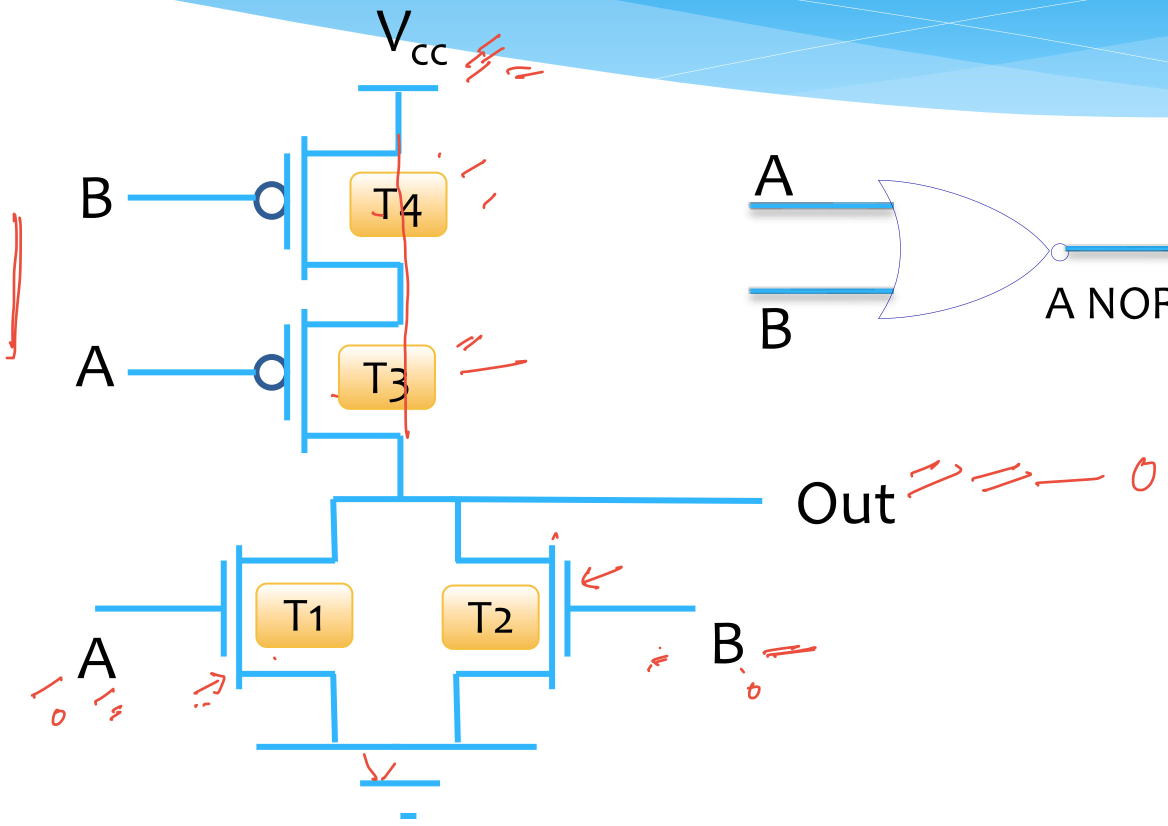


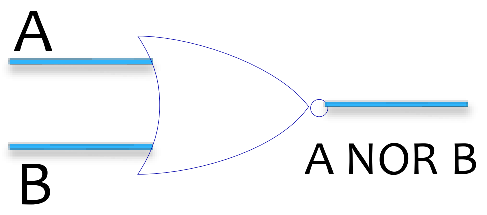
If A = 1 (+ve voltage)
T1 is on, T2 is off
Thus, the output will be connected to the ground, and it will be a logical 0

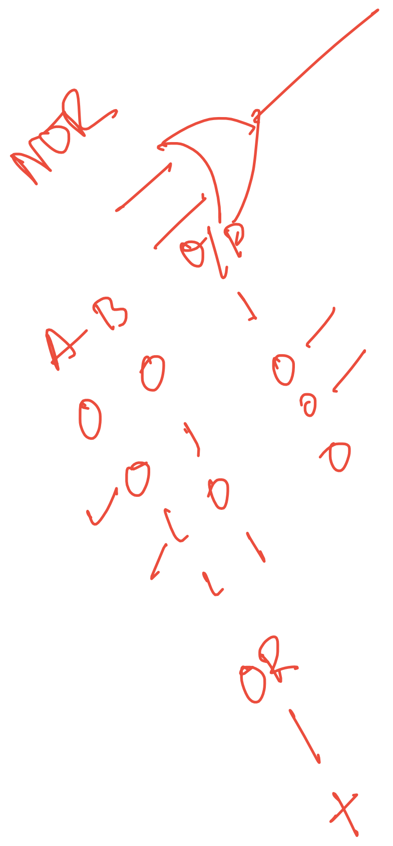
If A = 0 (0 voltage)
T1 is off, T2 is on
Thus, the output will be connected to the supply (Vcc), and it will be a logical 1

Complementary Metal-Oxide-Semiconductor

NOR Gate





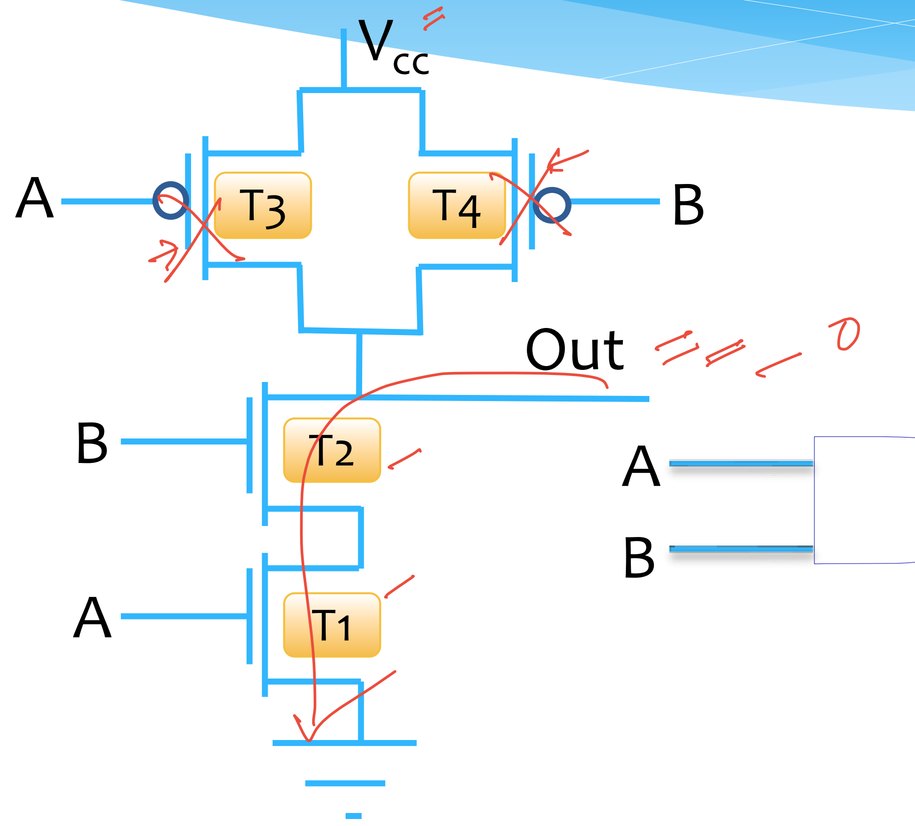


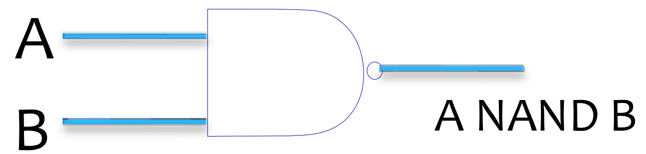
A	B	T1	T2	T3	T4	Out
0	0	off	off	on	on	1
1	0	on	off	off	on	0
0	1	off	on	on	off	0
1	1	on	on	off	off	0

NAND

A	B	O/P
0	0	1
0	1	1
1	0	1
1	1	0

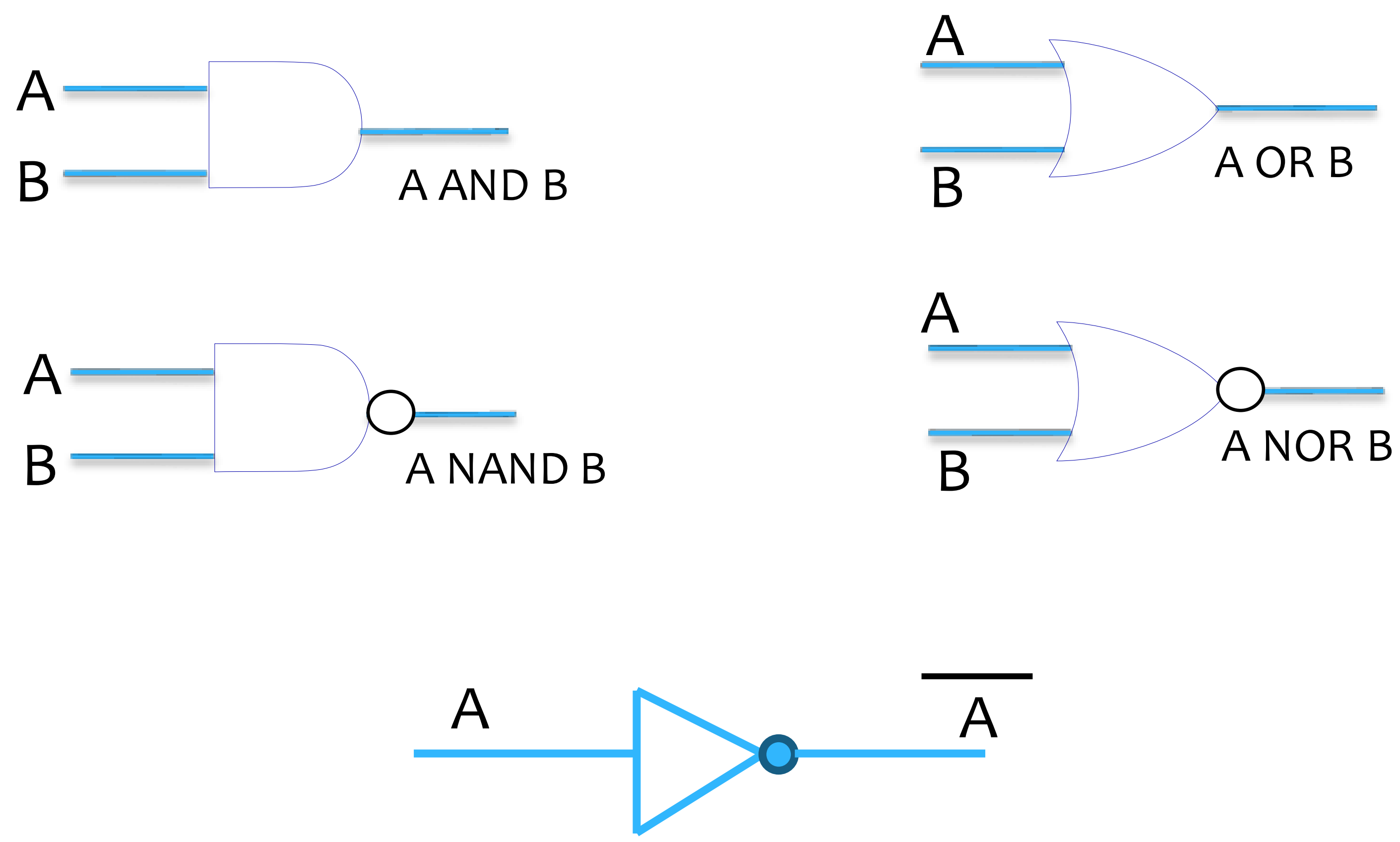
NAND Gate





A	B	T1	T2	T3	T4	Out
0	0	off	off	on	on	1
1	0	on	off	off	on	1
0	1	off	on	on	off	1
1	1	on	on	off	off	0

Summary of Logic Gates

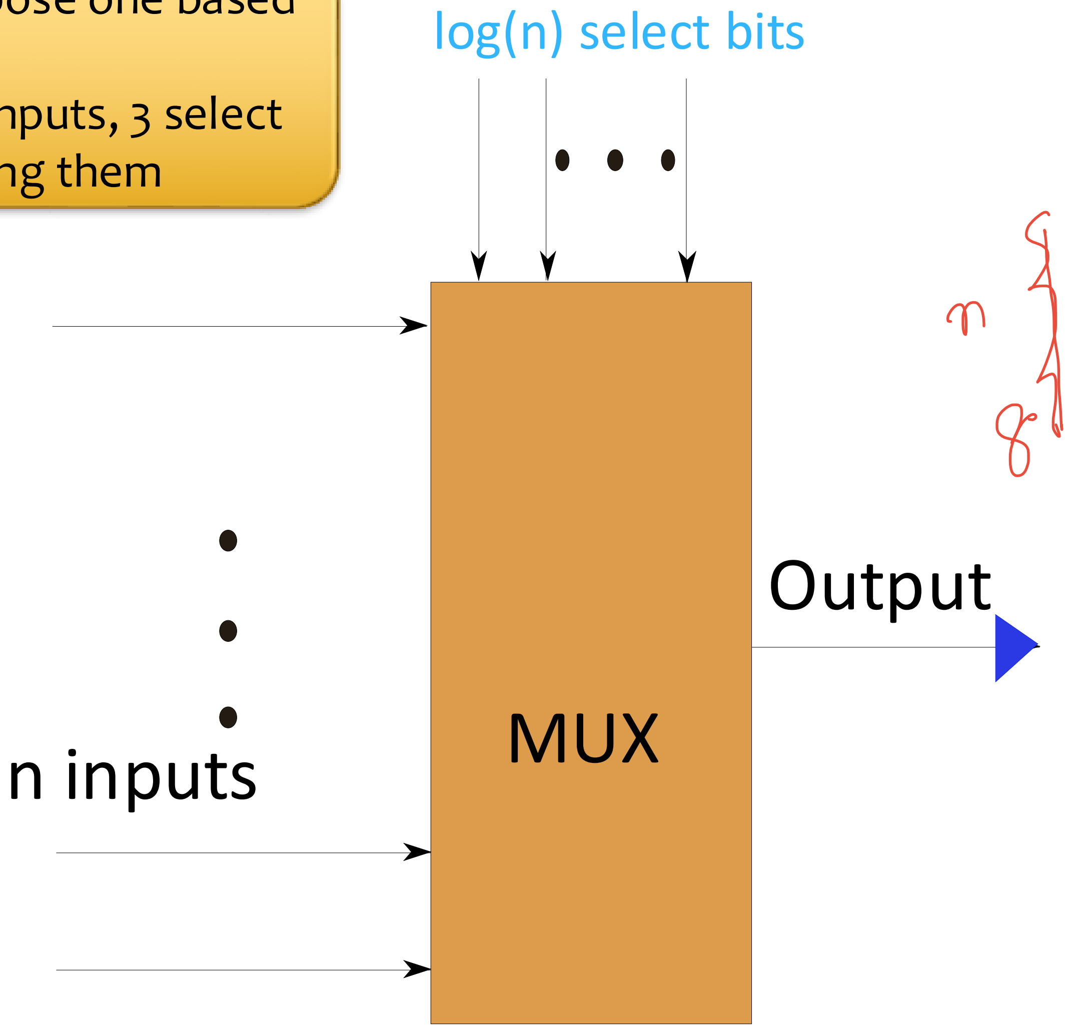


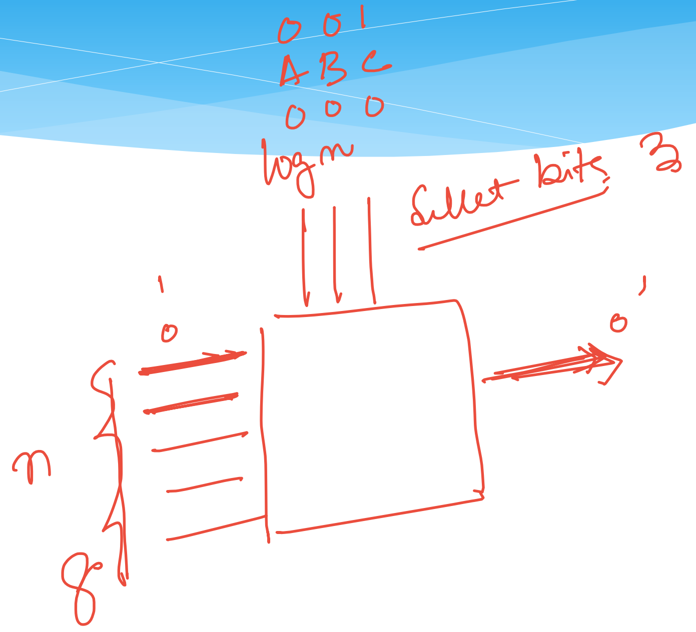
Outline

- Transistors and Gates
- Combinational Logic
- Sequential Logic
- SRAM/ DRAM Cells

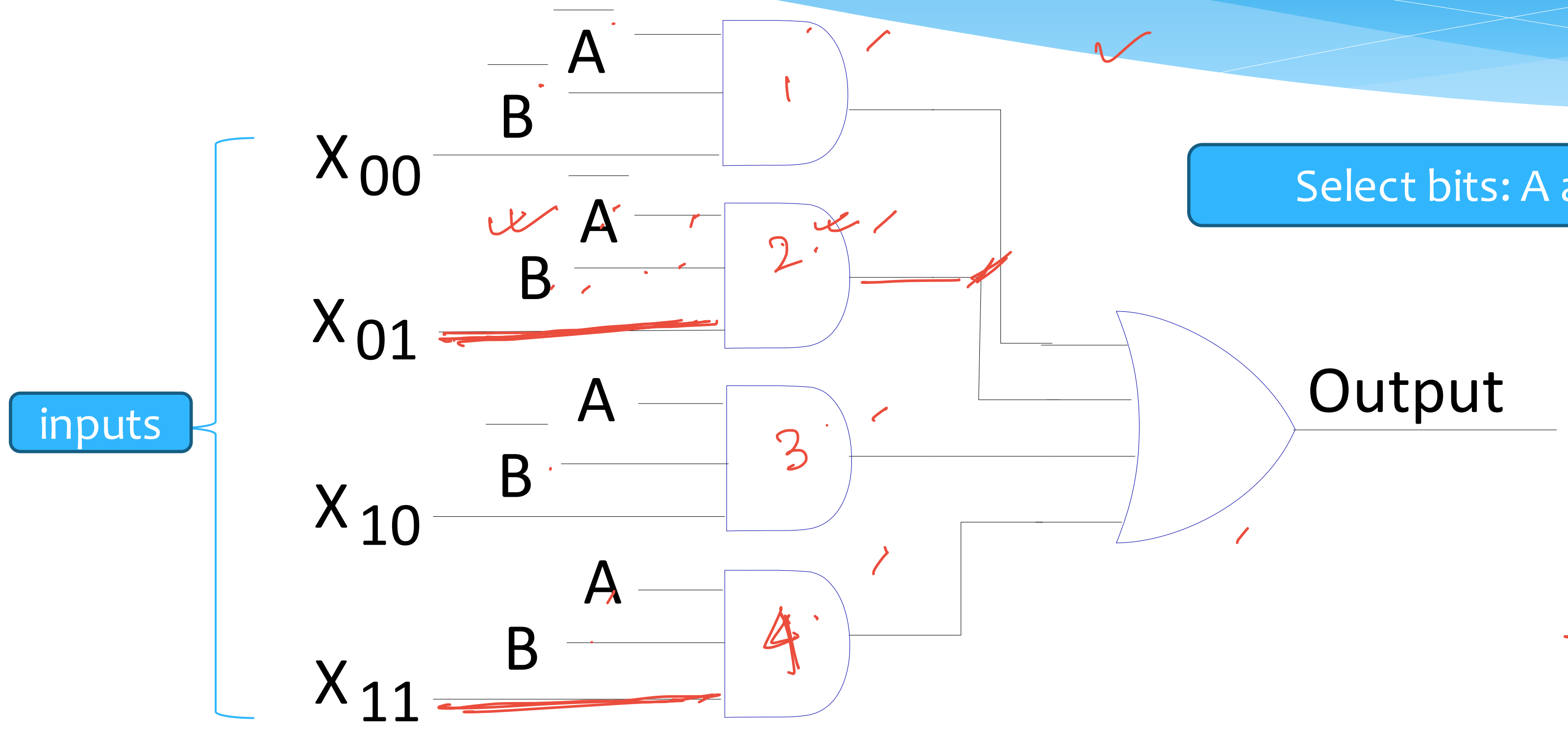
Multiplexer

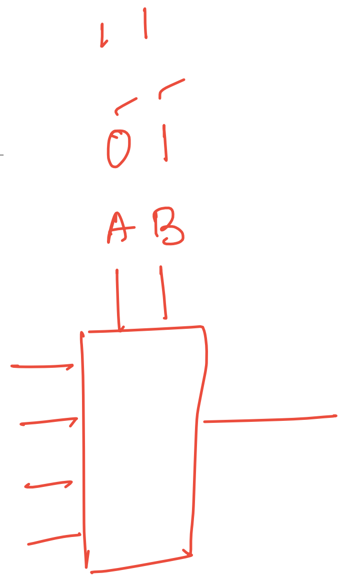
Given *n* inputs, choose one based on the select bits.
Example: Given 8 inputs, 3 select bits, choose 1 among them





Design of a Multiplexer

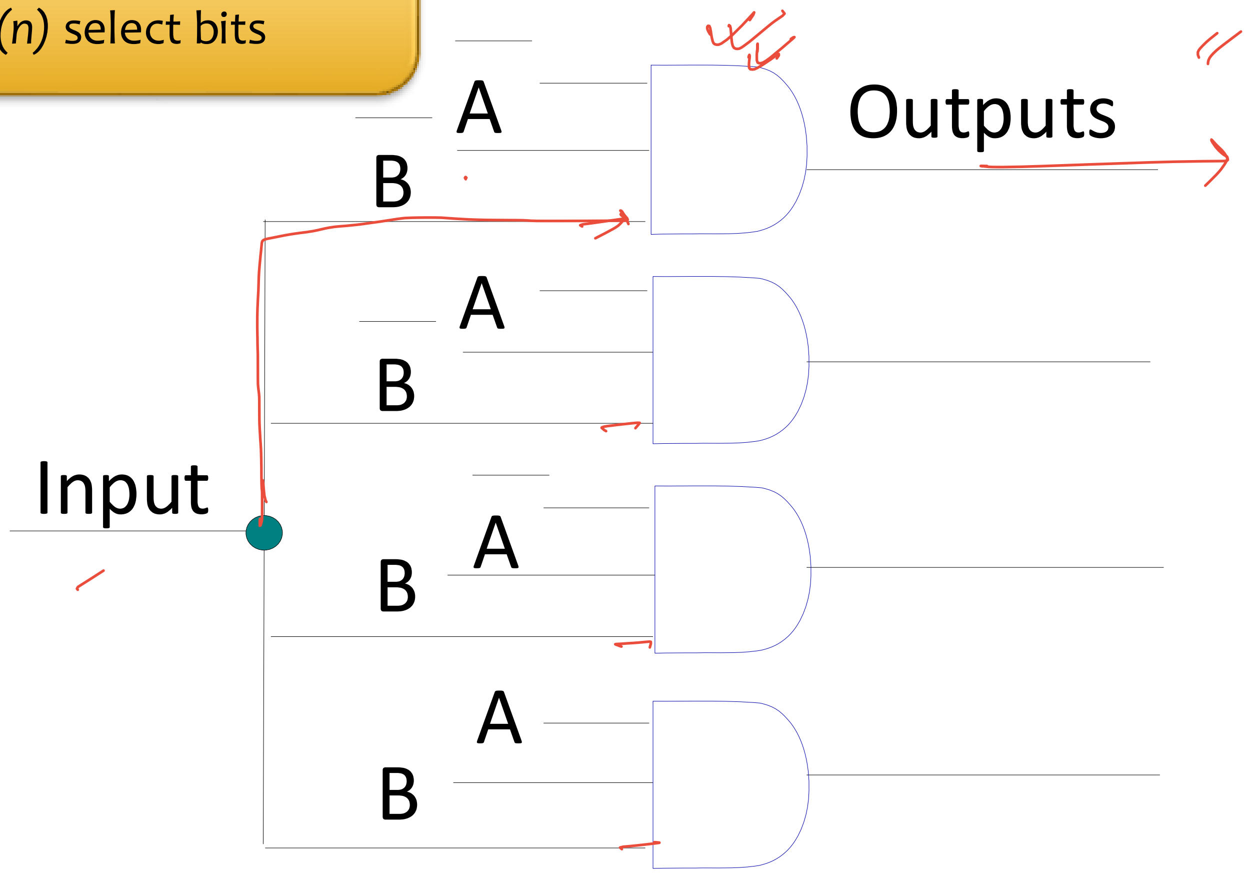


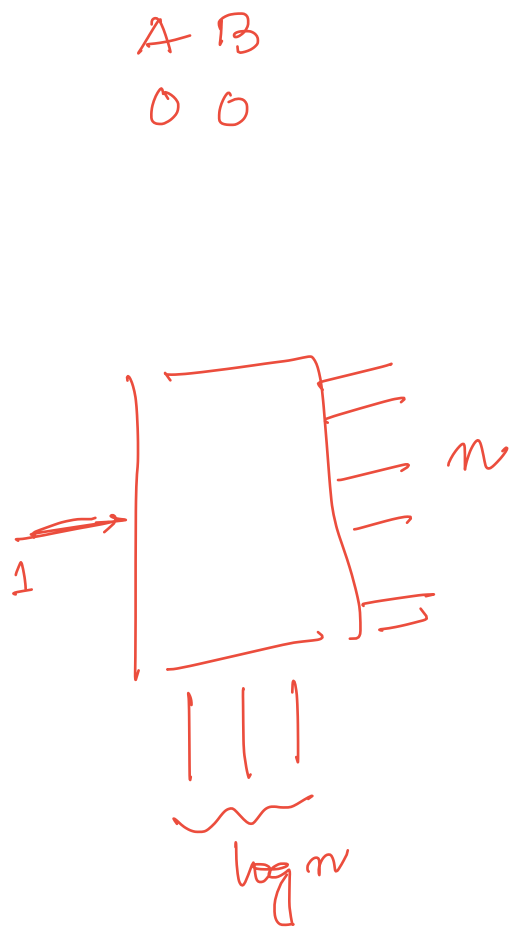


- Generate all combinations of select bits
  - $\overline{A}.\overline{B}, A.\overline{B}, \overline{A}.B, A.B$
  - Only one of the combinations is true.
  - For this combination the output of the AND gate is equal to the input (X_{AB}), rest of the outputs of the AND gates are 0
  - The output is equal to (X_{AB} OR 0 OR 0 OR 0) = X_{AB}

Demultiplexer

Given *1* input, make it appear on one of *n* outputs. The output is decided by *log(n)* select bits



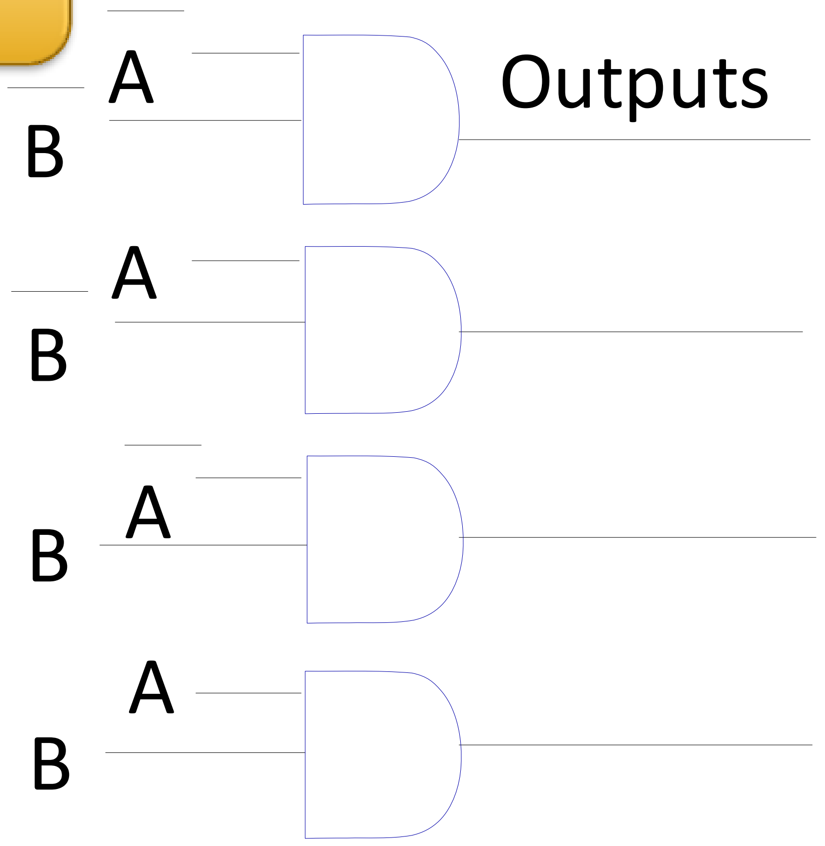


Working of a Demultiplexer

- Same logic
  - Generate all combinations of A, B, $\bar{A}$, and $\bar{B}$
  - Only one combination is TRUE
  - The output of that AND gate is equal to: X AND 1 = X
    Here, X is the input
  - The outputs of the rest of the AND gates are **0**

Decoder

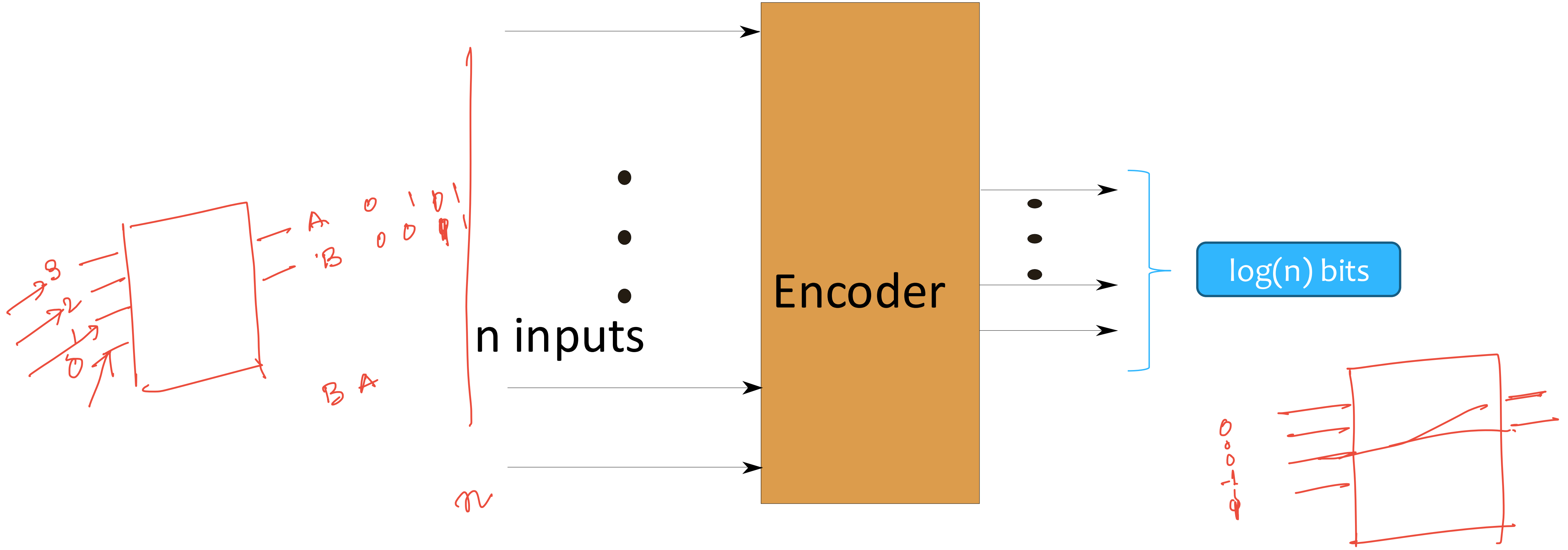
Set one of the *n* outputs to 1. The output is decided by *log(n)* select bits



Operation is similar to a demultiplexer

Encoder

Given *n* inputs, assume only one of them is 1. Find its id.
For example: if the 6th input out of 8 inputs is 1. The output should be 110



Example of a 3-bit Encoder

Output bits

Input bit	Bit2	Bit1	Bit0
b0	0	0	0
b1	0	0	1
b2	0	1	0
b3	0	1	1
b4	1	0	0
b5	1	0	1
b6	1	1	0
b7	1	1	1

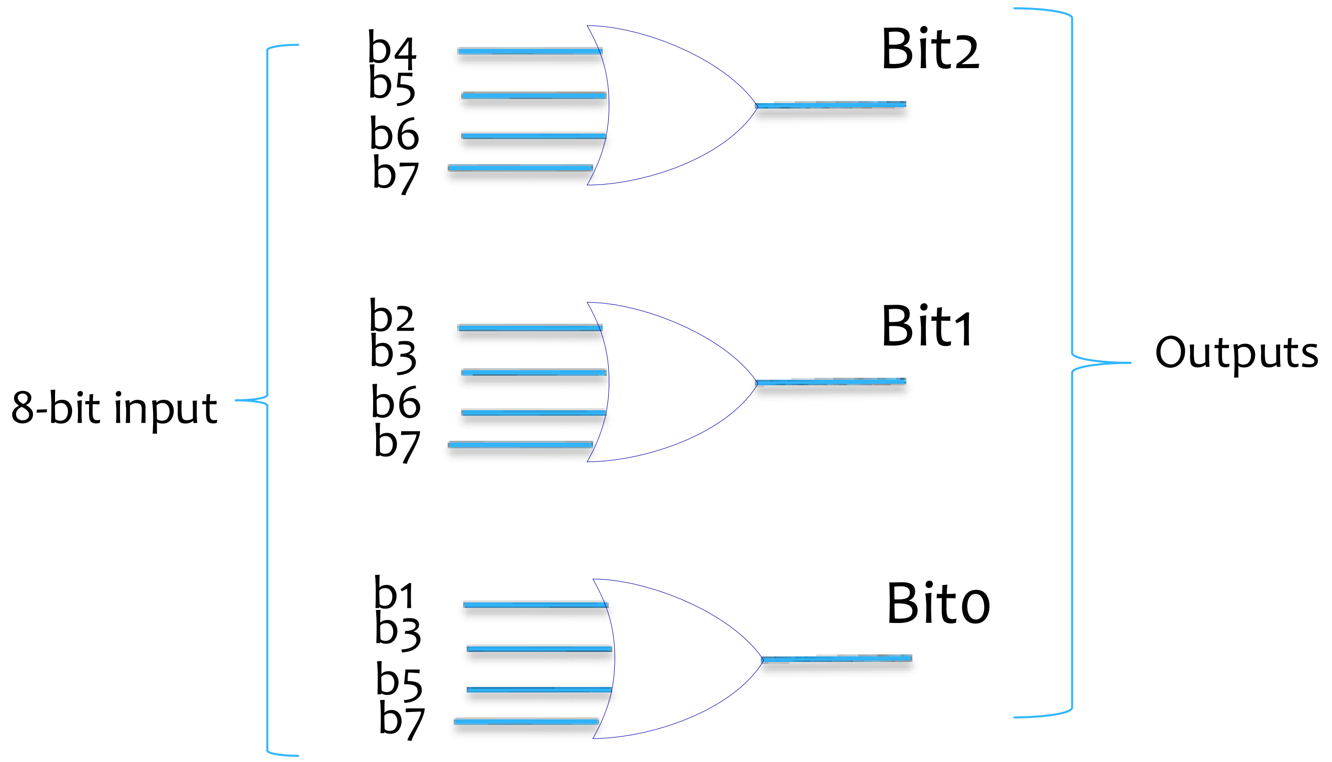
Here, the + symbol stands for OR

Bit2 = b4 + b5 + b6 + b7

Bit1 = b2 + b3 + b6 + b7

Bit0 = b1 + b3 + b5 + b7

Example of a 3-bit Encoder - II



Outline

- Transistors and Gates
- Combinational Logic
- Sequential Logic
- SRAM/ DRAM Cells

S-R Latch

S : Set
R : Reset

S=1 R=0
S=0 R=1
S=0 R=0
$\boxed{S=1 \quad R=1}$

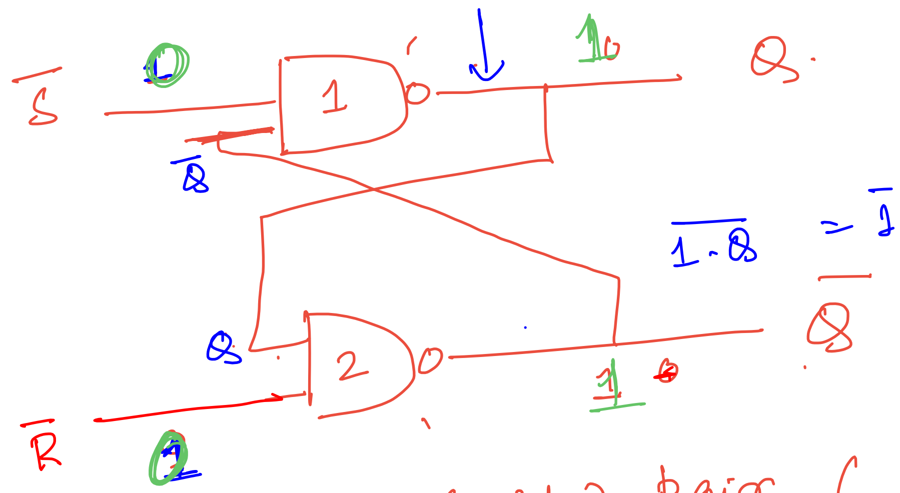
Q=1 $\bar{Q}$=0 — Set
Q=0 $\bar{Q}$=1 — Reset
main~~tain~~ old values.
✗ not desired state

$\overline{1\cdot\bar{Q}} = \bar{1} + \bar{\bar{Q}} = 0 + Q = Q$

NAND

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

At least one i/p = 0
↪ o/p = 1

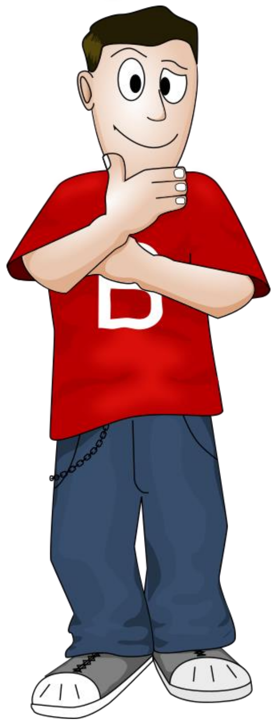


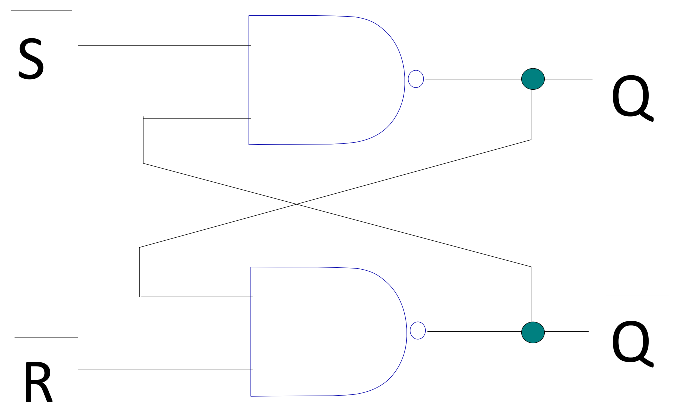
$\overline{1\cdot Q} = \bar{1} + \bar{Q} = 0 + \bar{Q} = \bar{Q}$

Cross-coupled pair (Latch)

SR Latch

Use a cross-coupled pair of NAND gates to store a stable value.





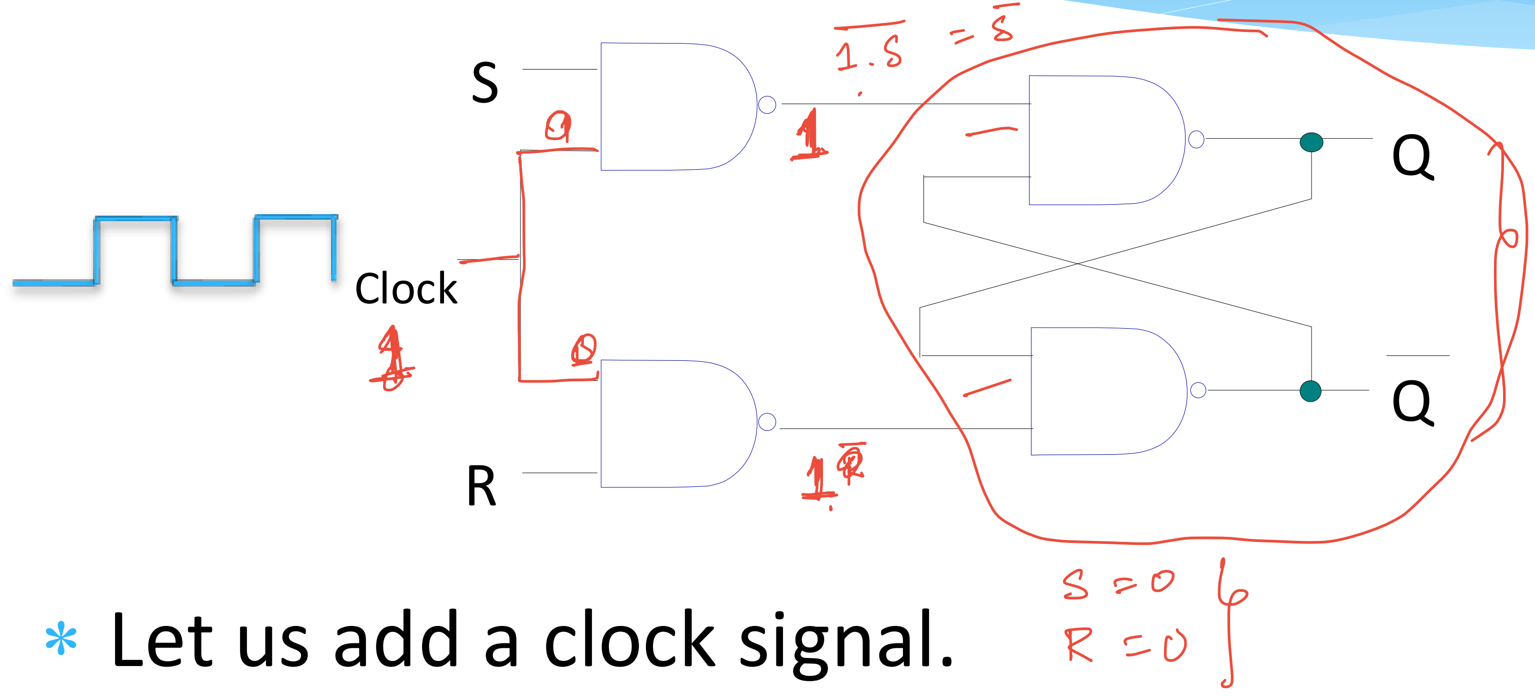
S → Set
R → Reset

* S = 1, R = 0, Q = 1
* S = 0, R = 1, Q = 0
* S = 0, R = 0, <maintain old values>

Set the values

Understand these points

Clocked SR Latch

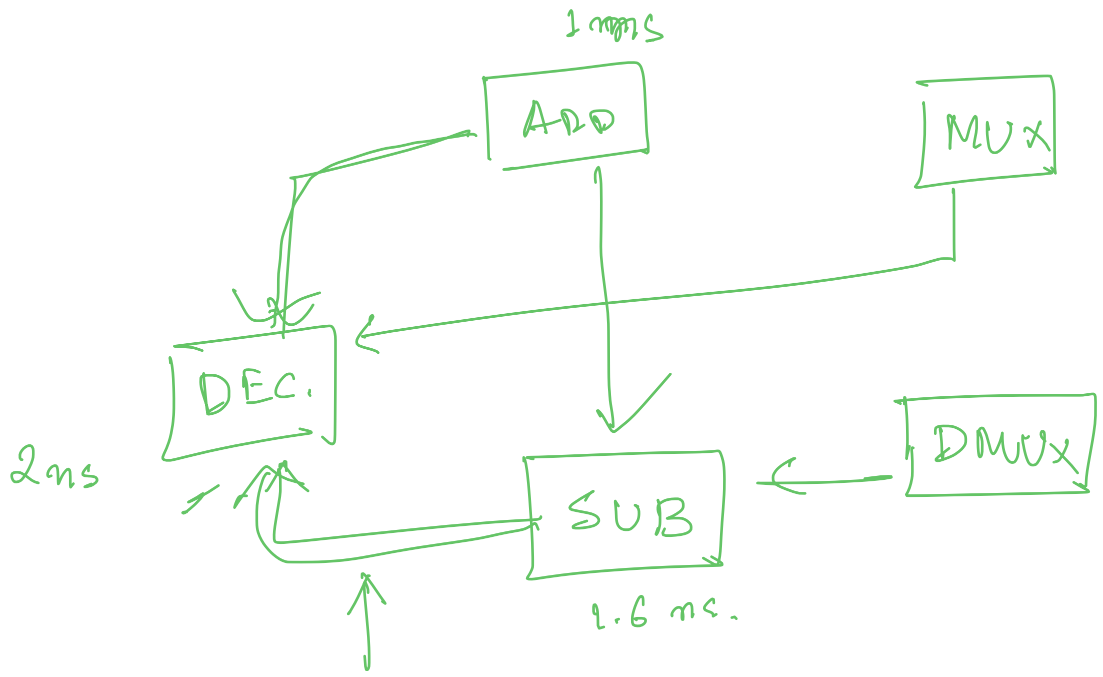


freq = $\frac{1}{\text{clock period}}$

200 KHz

S = 1   R = 0

* Let us add a clock signal.
  * When the clock is 1, outputs of the NAND gates are $\bar{S}$ and $\bar{R}$ respectively (same as the classic SR latch)
  * Clock is 0 → they are 1 and 1 respectively (maintain old values)


1 ~~ns~~ ns
ADD
MUX
DEC.
SUB
DMUX
2ns
1.6 ns.