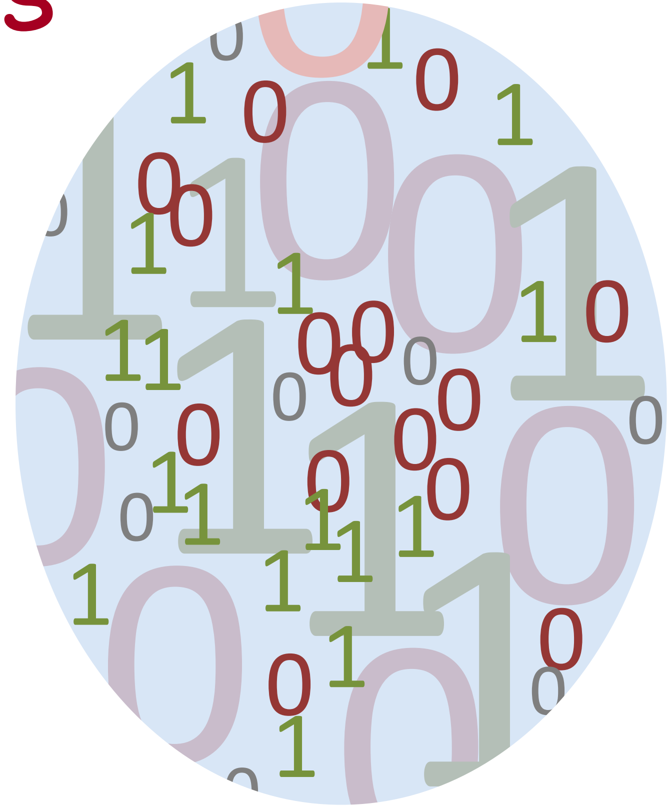
6.002x CIRCUITS AND ELECTRONICS

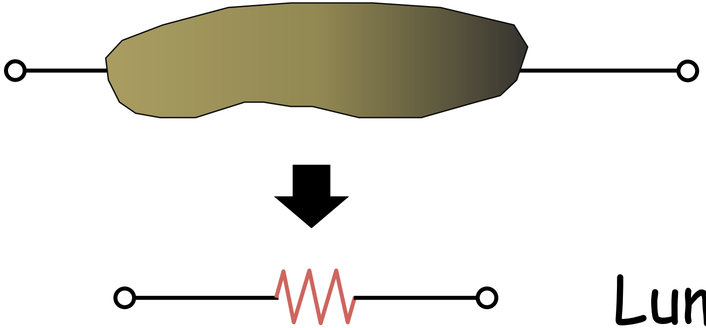
# The Digital Abstraction



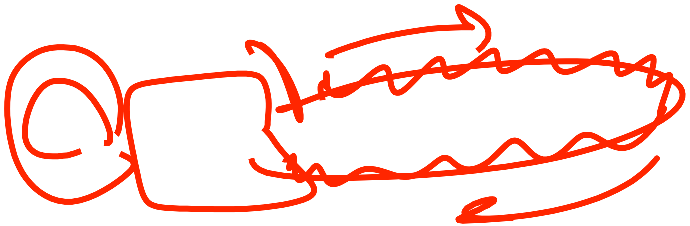
Reading: Chapter 5 of A&L

# Review

- Discretize matter by observing lumped matter discipline



Lumped Circuit Abstraction



- Analysis tool kit

KVL/KCL, composition, node, superposition, Thévenin, Norton

all elts linear

# In this Sequence

Discretize value → Digital abstraction

lump value

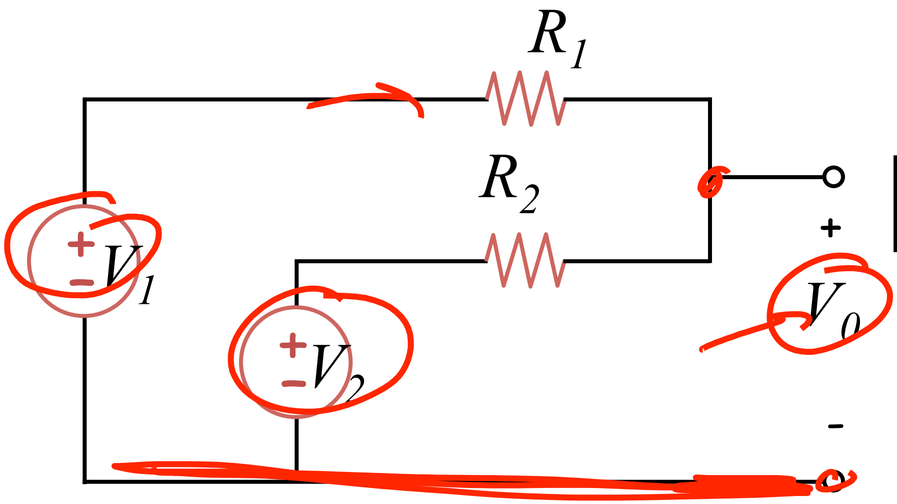
Interestingly, we will see shortly that the tools learned in the previous three lectures are sufficient to analyze simple digital circuits

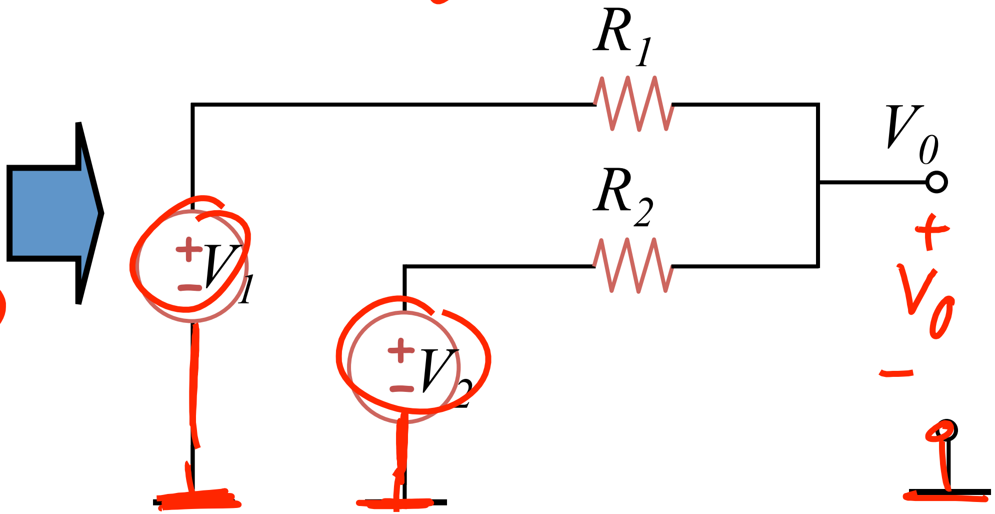
Reading Chap 5

# But first, why digital?

## In the past …

Analog signal processing



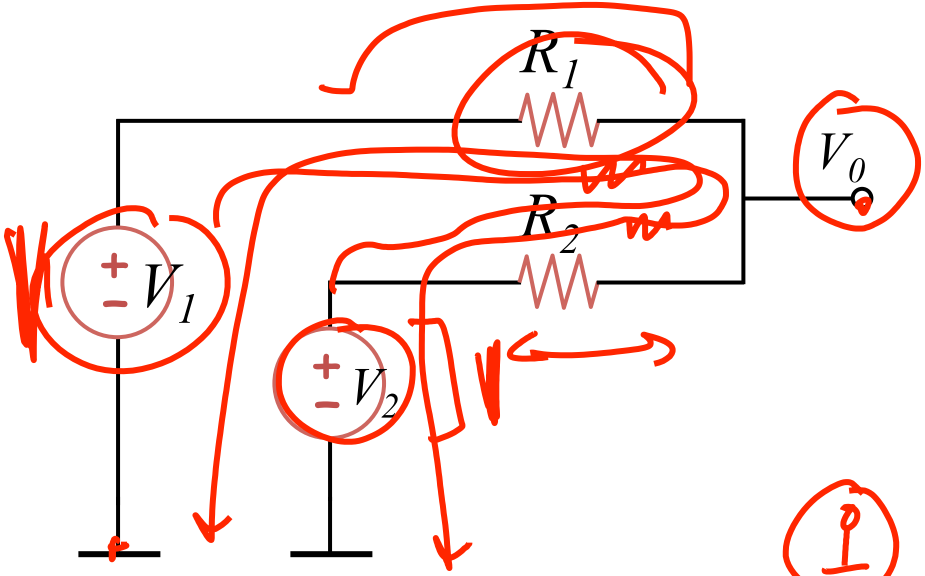




$V_1$ and $V_2$ might represent the outputs of two sensors, for e.g.

Shorthand notation (from node method)

# Why digital?

Analog signal processing




linear

By superposition

$$V_0 = \frac{R_2}{R_1+R_2} V_1 + \frac{R_1}{R_1+R_2} V_2$$

If $R_1 = R_2$

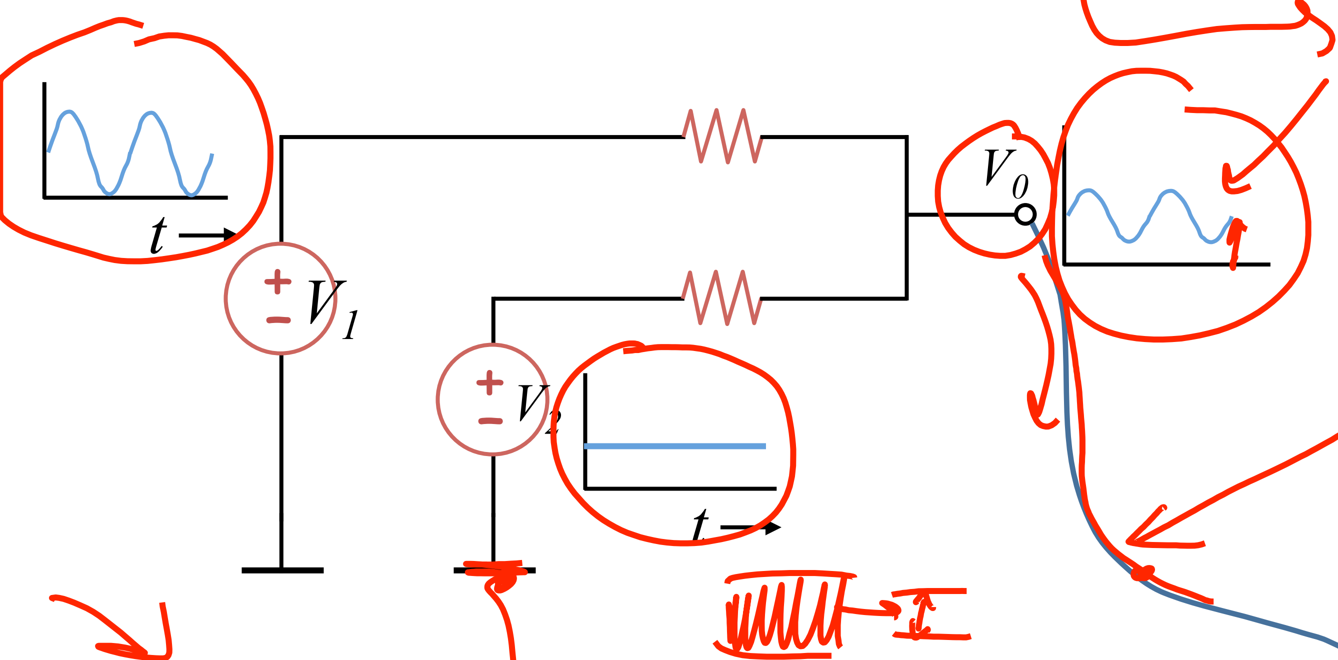
$$V_0 = \frac{V_1 + V_2}{2}$$

$V_1$, $V_2$

The above is an “adder” circuit.

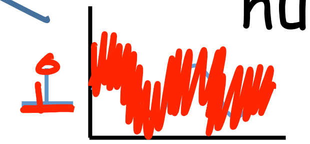
# Noise Problem with Analog

lacks noise immunity




add some noise on the wire

~~Reseiver~~: ckt
huh?



Noise hampers our ability to distinguish between small differences in value — e.g. between 3.1V and 3.2V.

# Idea: Value Discretization (or lumped values)

Restrict values to be one of two

HIGH LOW

5V 0V

TRUE FALSE

1 0

"1" "0"

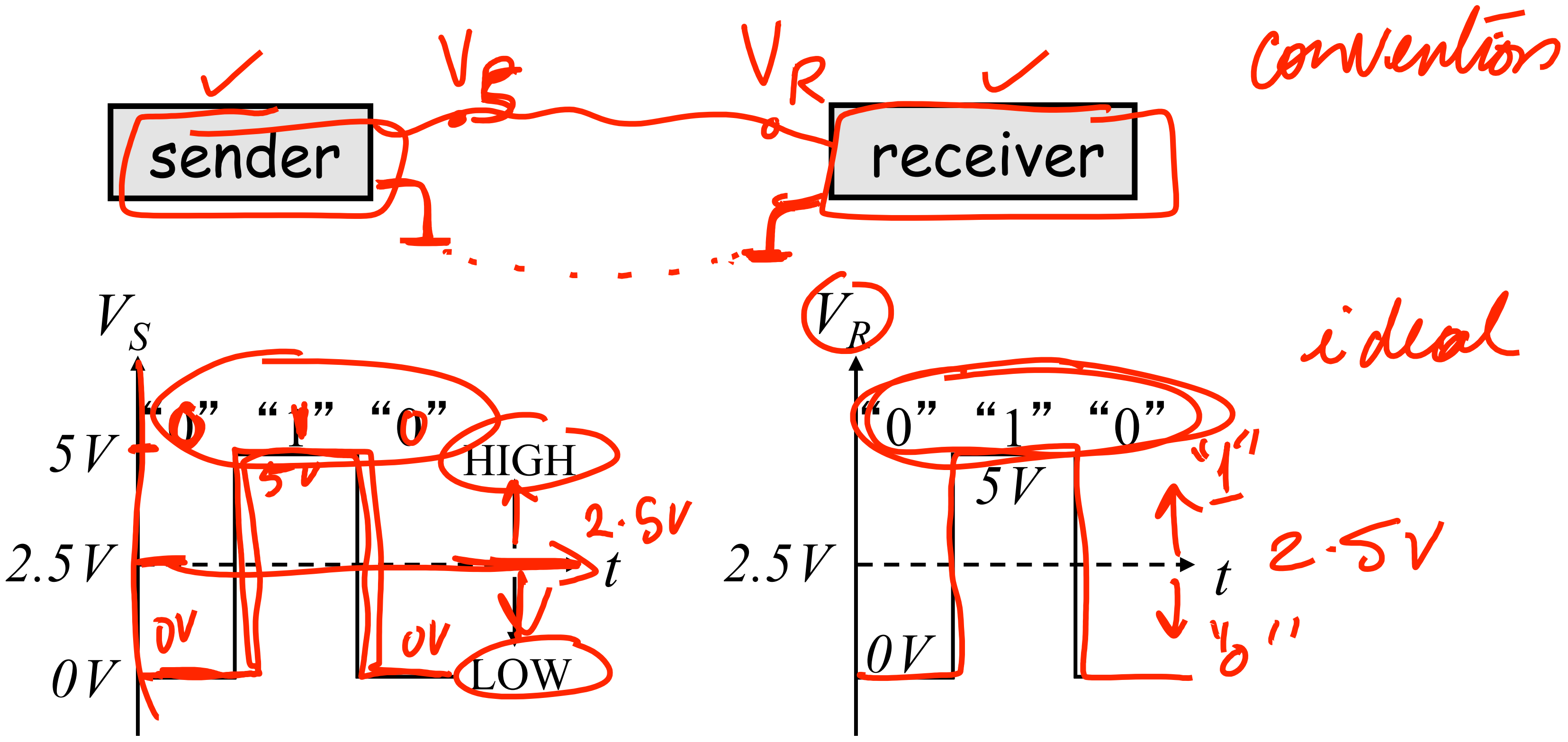
modern world

1V 0V

...like two digits 0 and 1

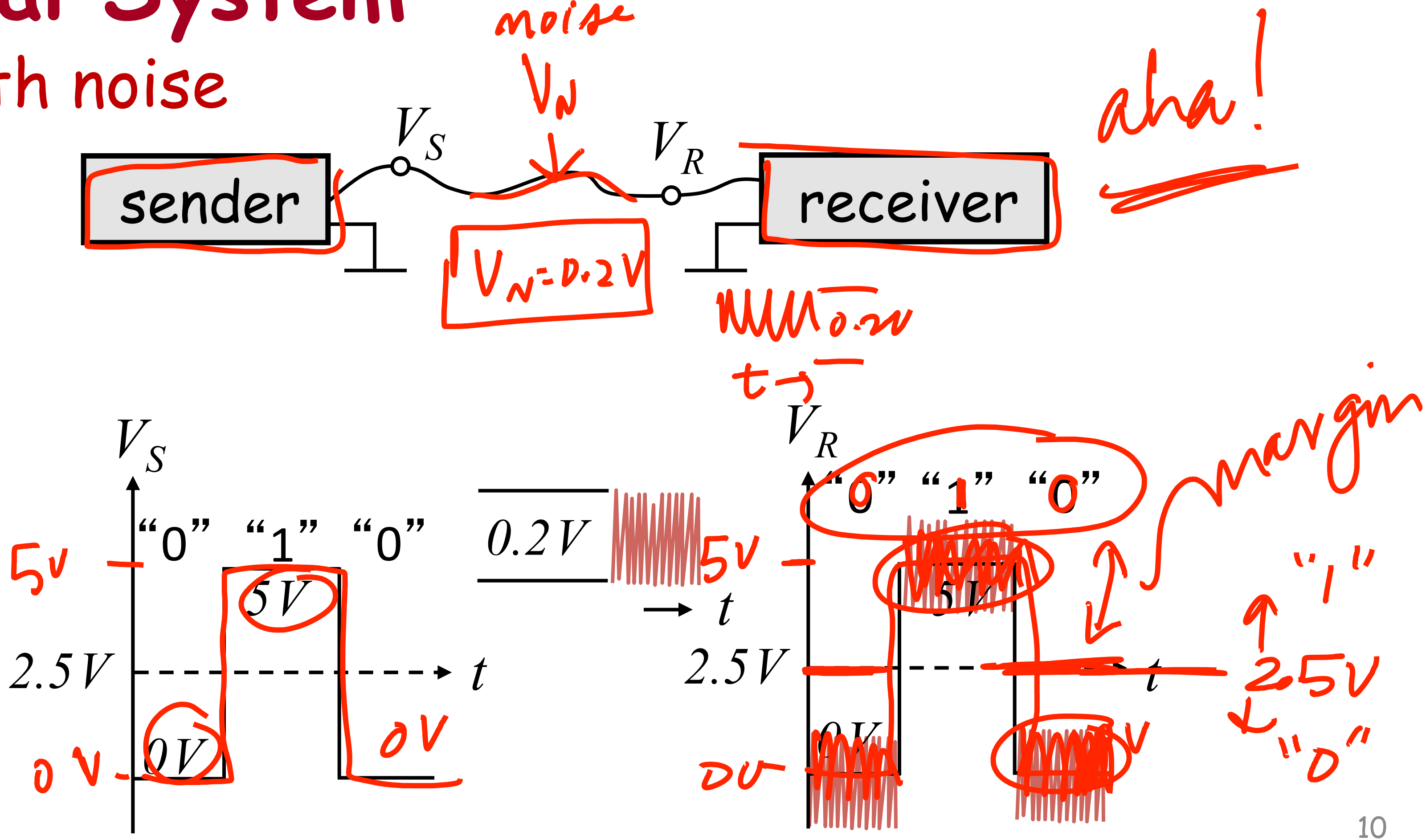
# Why is this discretization useful?

# Digital System

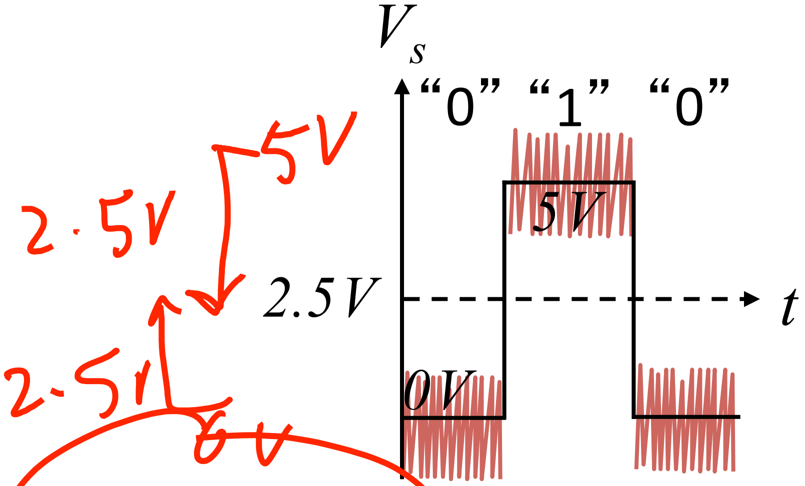


# Digital System

## With noise



# Digital System

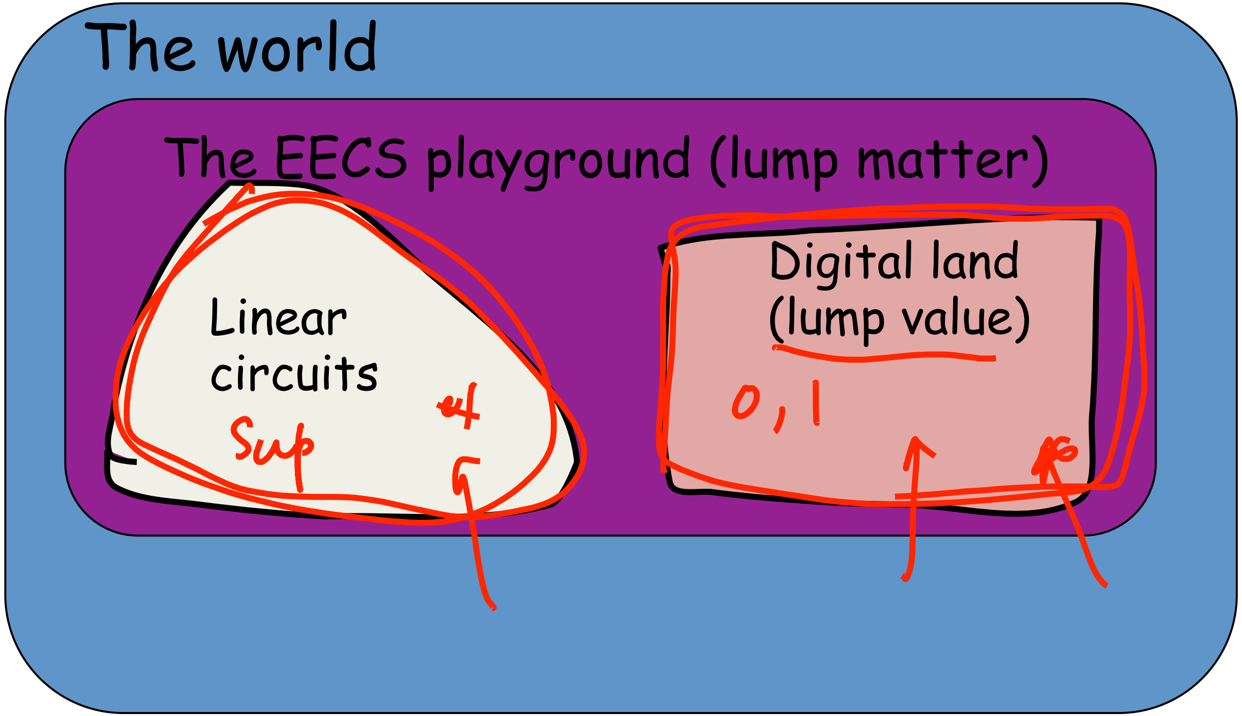



Better noise immunity → Lots of "noise margin"

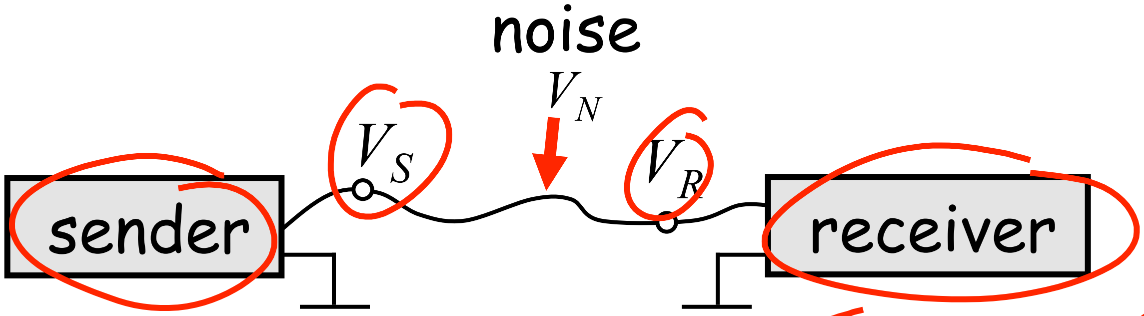
For "1": noise margin $\underbrace{5V \text{ to } 2.5V}$ $= 2.5V$

For "0": noise margin $\underbrace{0V \text{ to } 2.5V}$ $= 2.5V$

# The Big Picture



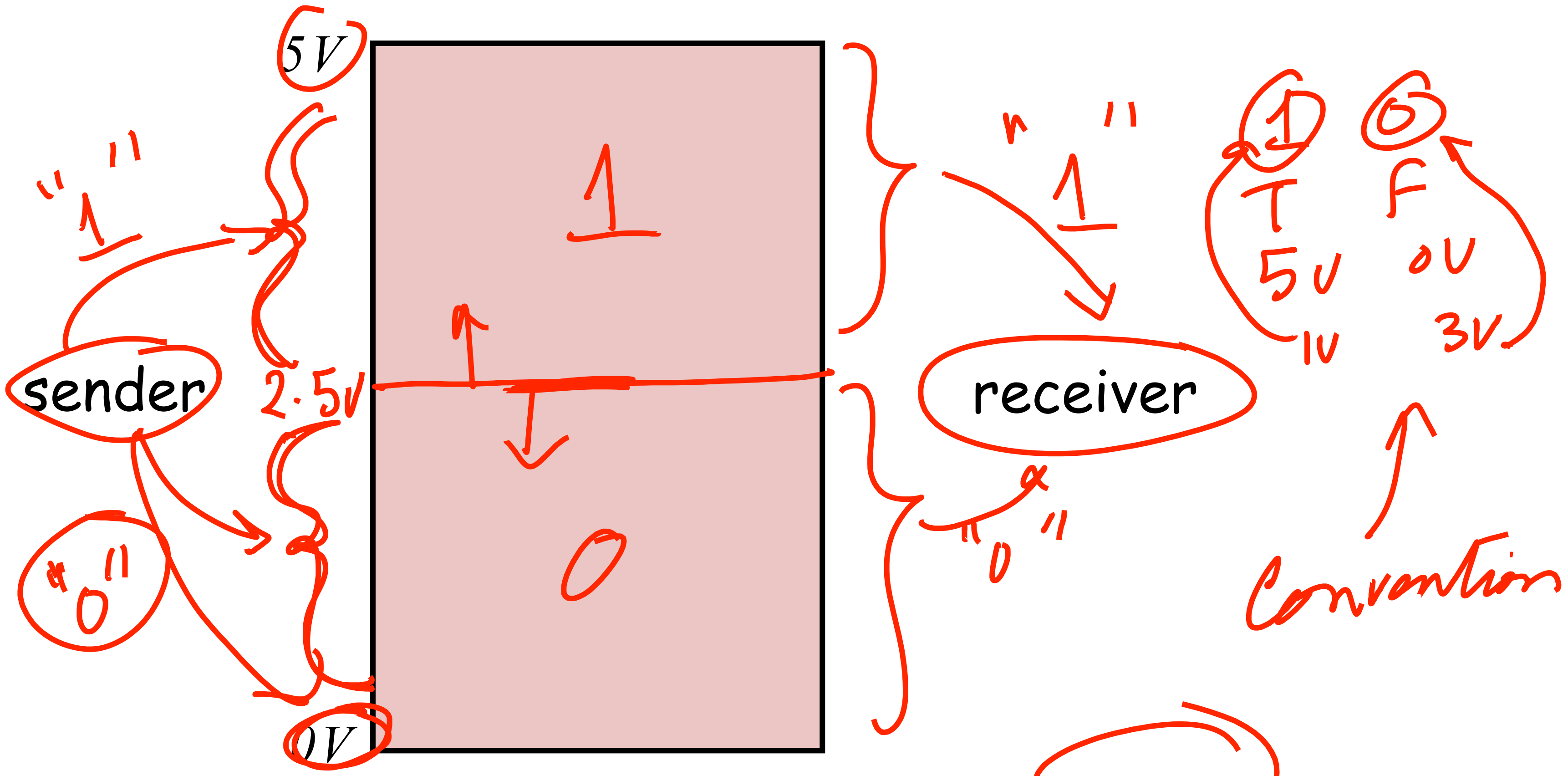
# Digital System Sender-Receiver Contract




here is a 1

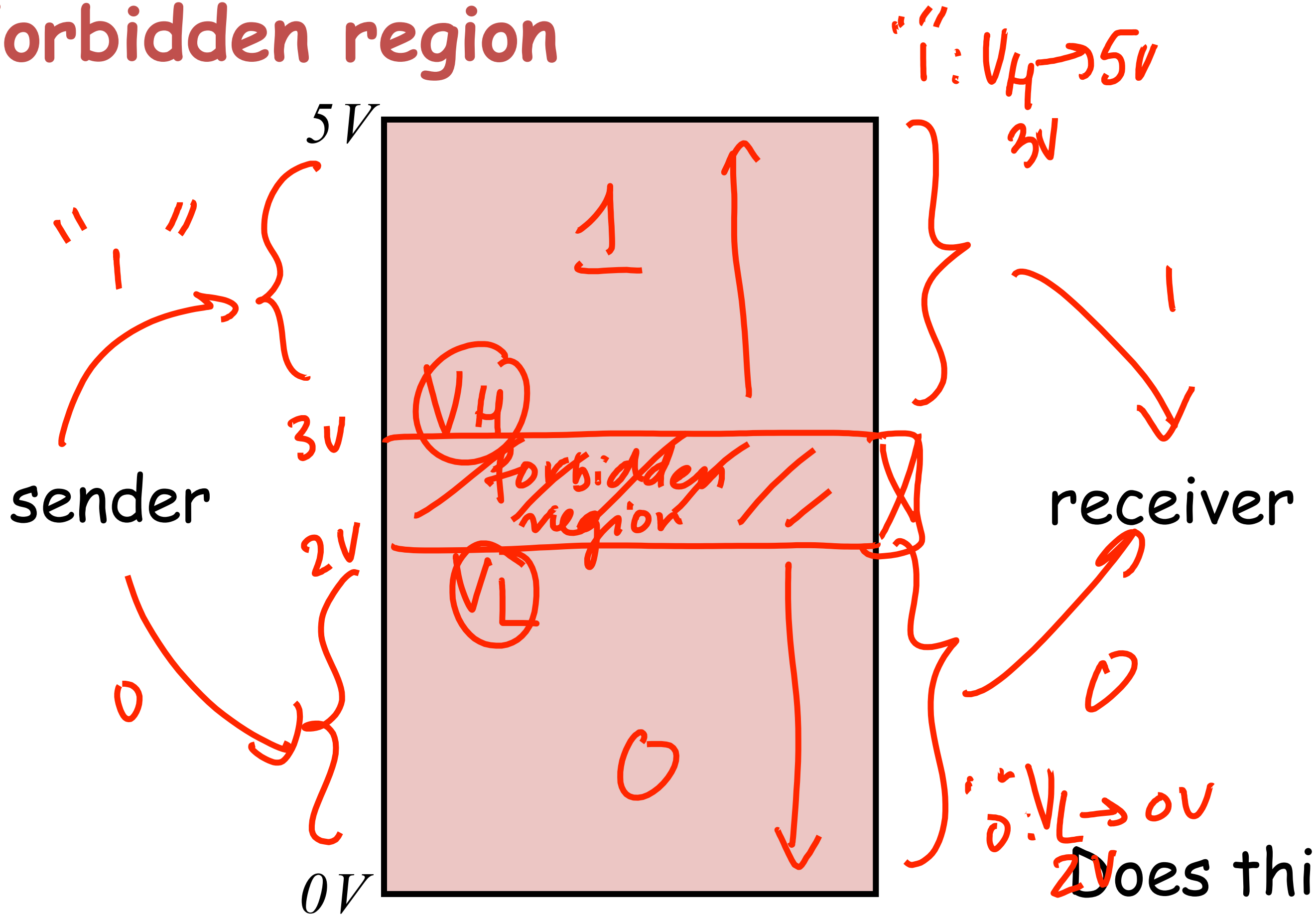
I see a 1 hmmm?

# Voltage Thresholds and Logic Values




But, but, but … What about 2.5V?

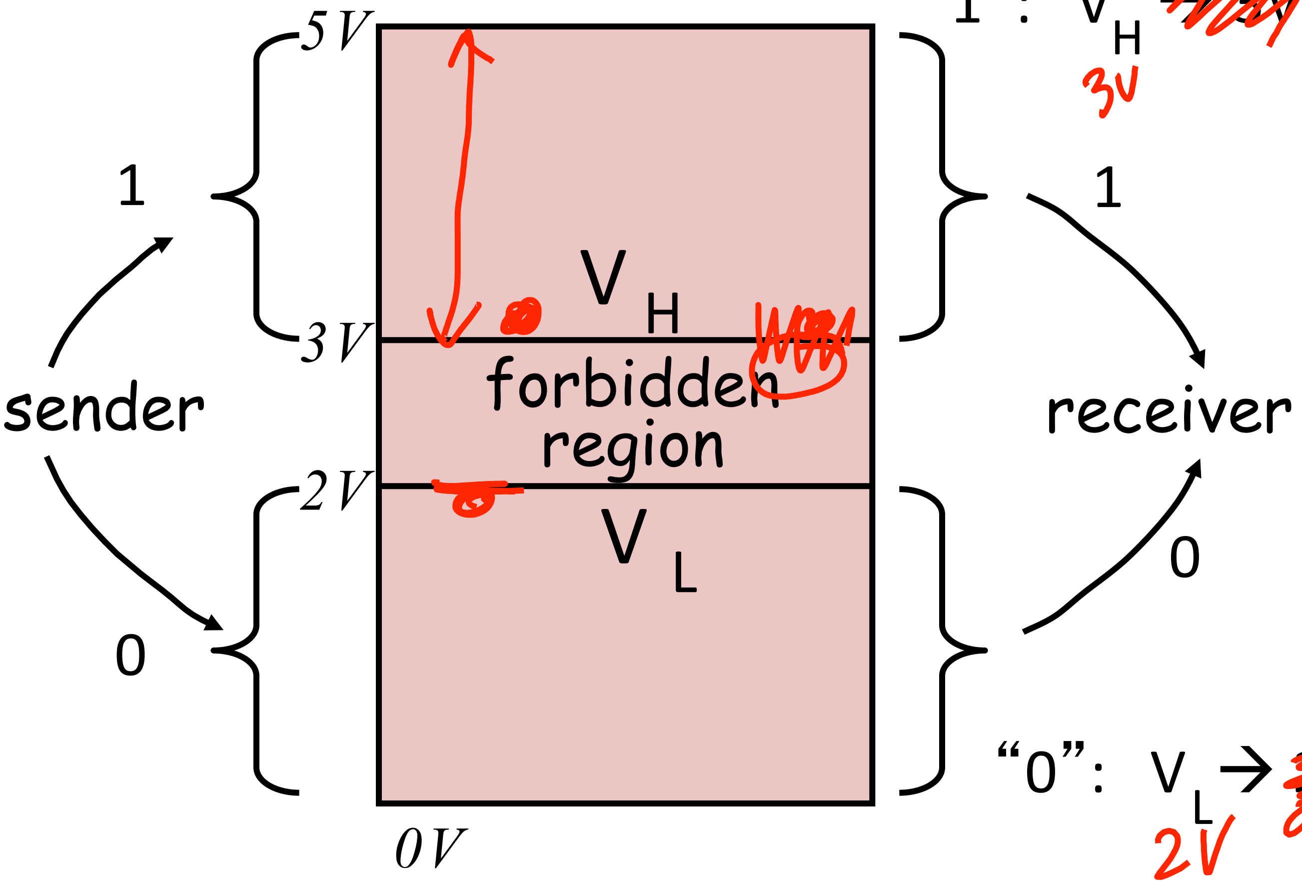
# Hmmm… Idea! Create “no man’s land” or forbidden region




Remember, we can do so with impunity because it is our choice as to what discipline we agree on in our digital playground

Does this work?

# "No Man's Land" or Forbidden Region




"1": $V_H$ → ~~3V~~ 5V

3V

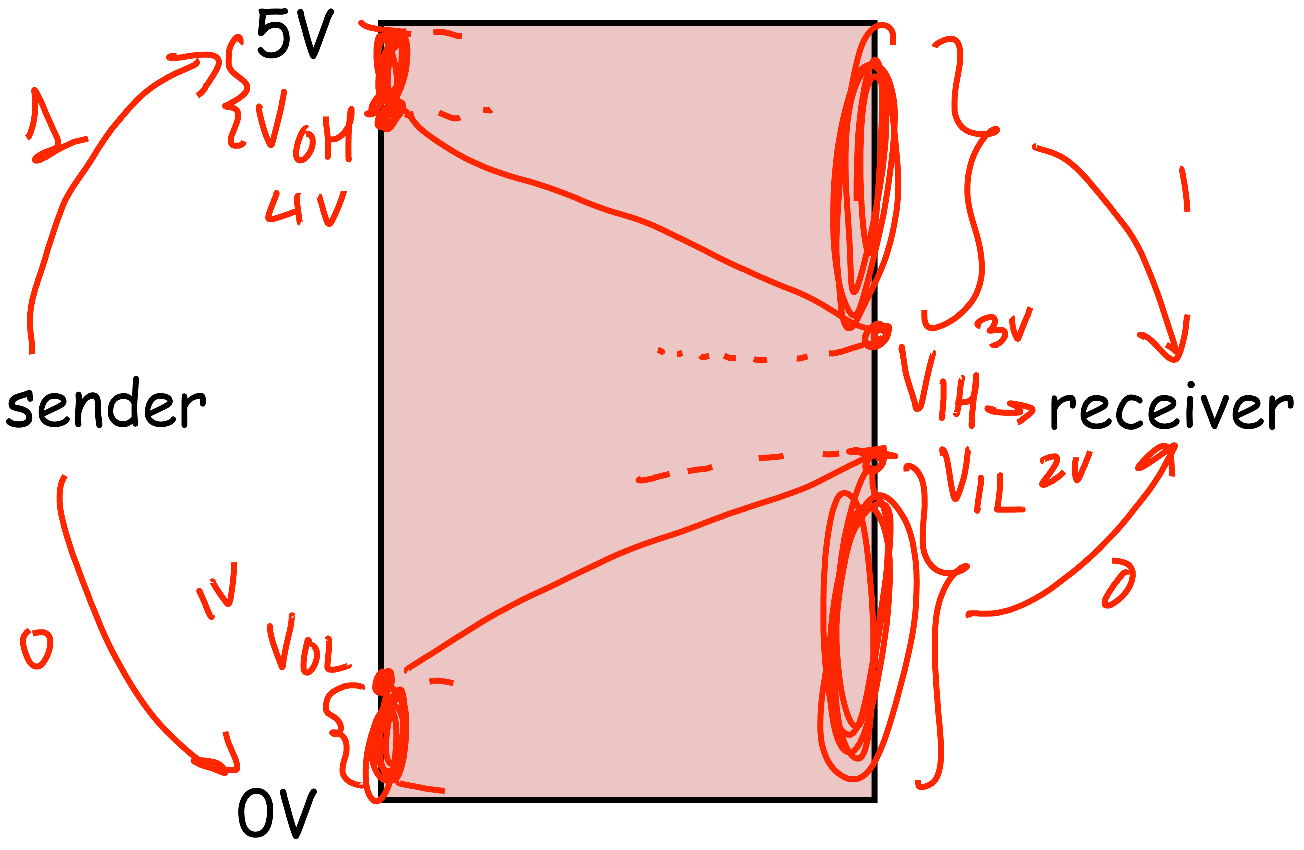
"0": $V_L$ → ~~2V~~ 0V

2V

**Where's the noise margin?**

**What if the sender sent 1: $V_H$** 3V

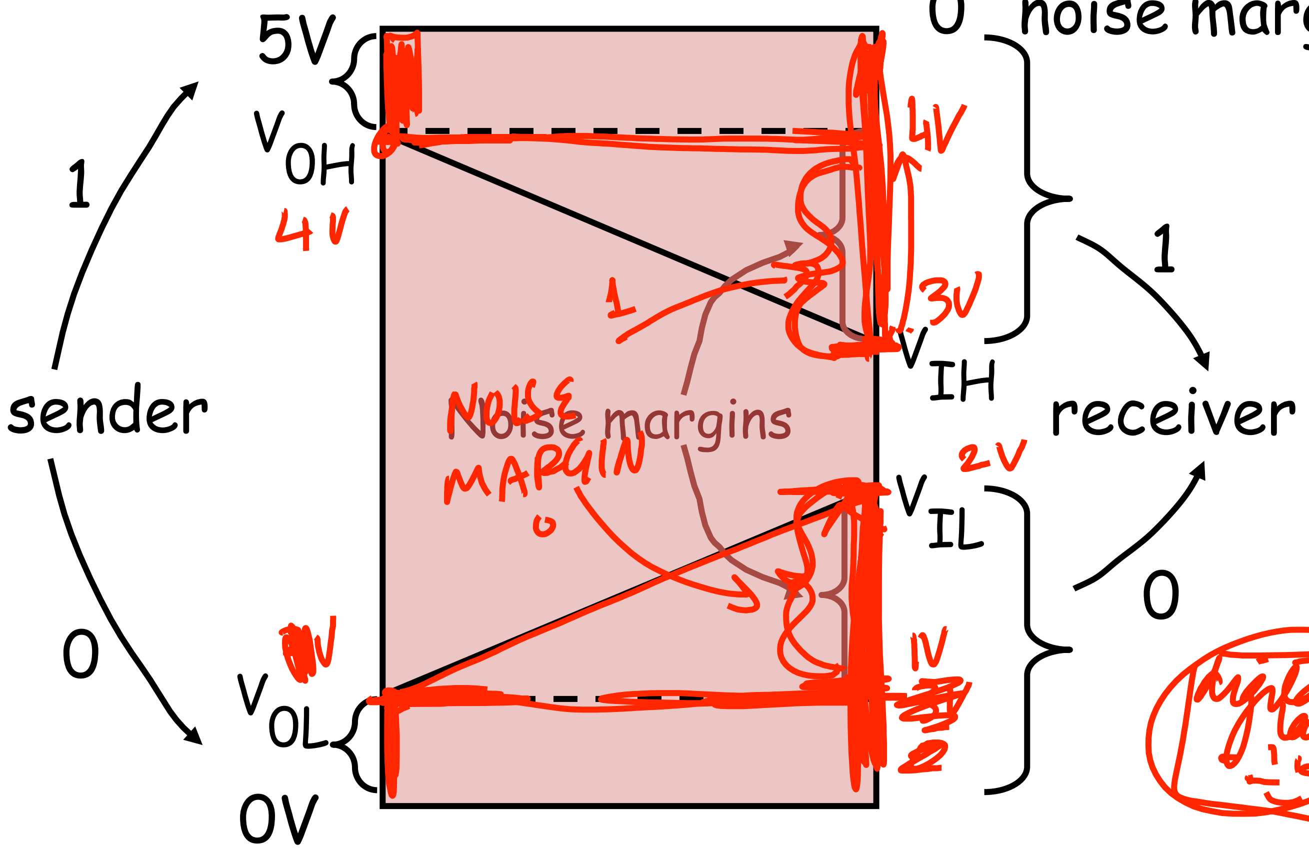
# Hold the Sender to Tougher Standards!



# Noise Margins

“1” noise margin: $V_{OH} - V_{IH}$

“0” noise margin: $V_{IL} - V_{OL}$

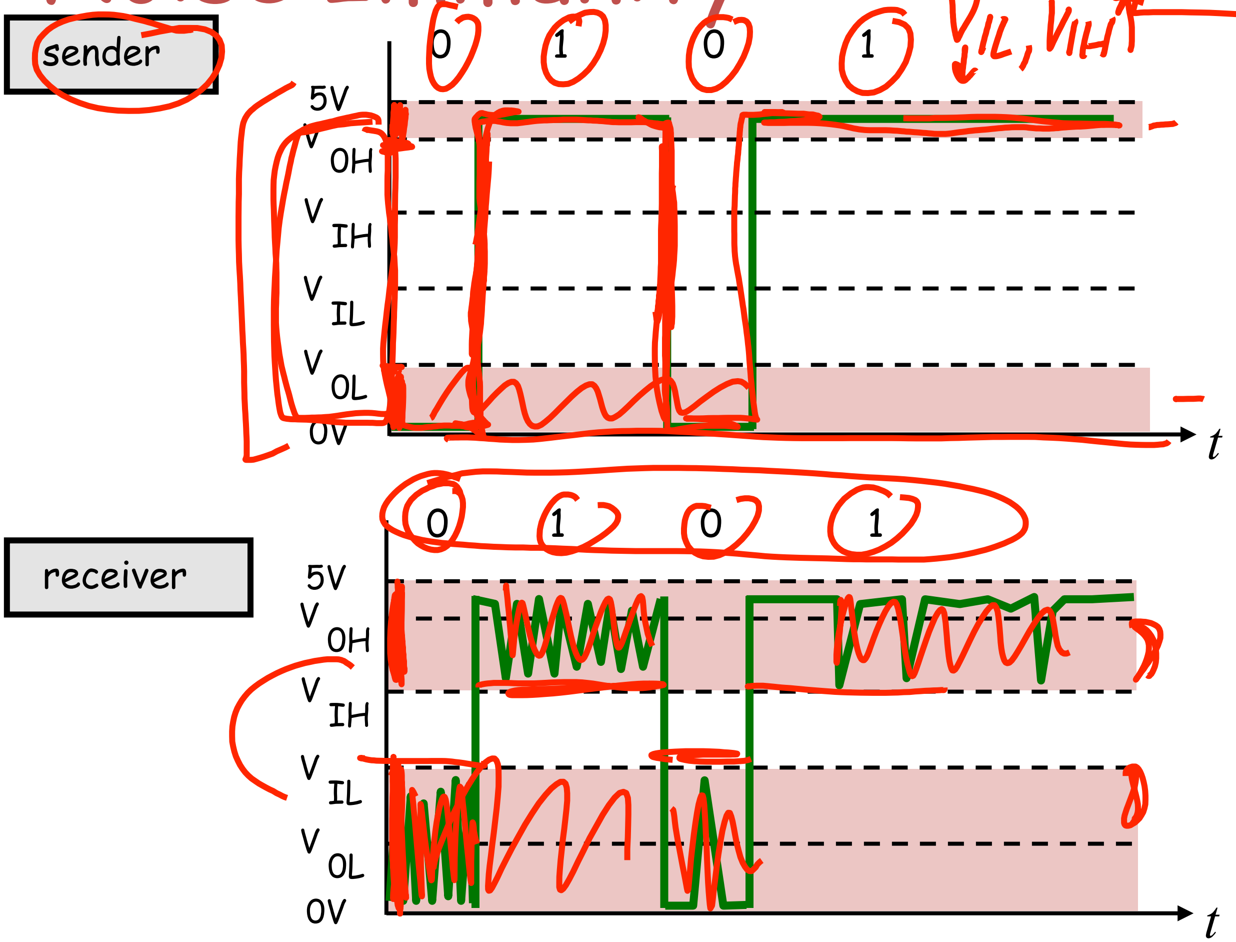



Together, the $V_{OH}$, $V_{IH}$, $V_{OL}$, $V_{IL}$ ~~thresholds define~~ a discipline or standard that digital devices follow so they can talk to each other

digital land

# Noise Immunity

IN → S → OUT $V_{OH}, V_{OL}$




Digital systems follow static discipline: if inputs to the digital system meet valid input thresholds, then the system guarantees its outputs will meet valid output thresholds.

# Processing Digital Signals

Recall, we have only two values –

**1 , 0** ⇨ Map naturally to logic: T, F

⇨ Can also represent numbers

0 - 9

9065?

What is 1011?

Check Chapter 5.6 of A&L

0, 1

# Processing Digital Signals

## Boolean Logic

⇨ If X is true and Y is true
Then Z is true, else Z is false.

F T
X → 0, 1
Y
Z binary vars.

$Z = X$ AND $Y$

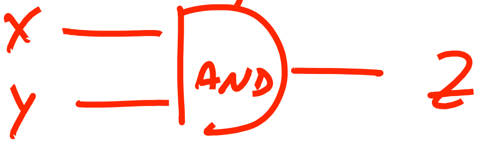
boolean equation → $\boxed{Z = X \cdot Y}$

| Z | X | Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0 | 1 |
| 0 | 1 | 0 |

x, y, z
digital signals
"0" or "1"

# Processing Digital Signals

$z = x \cdot y$ boolean equations

→ 

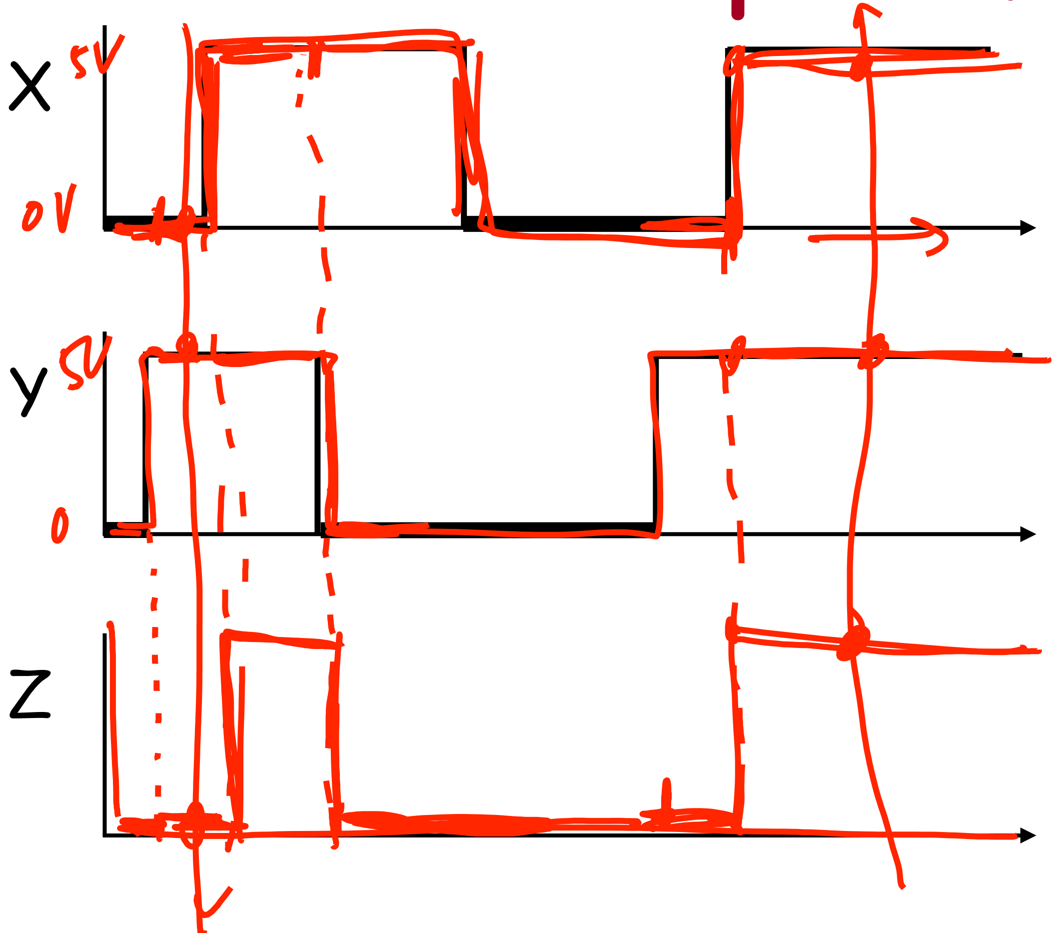

AND gate

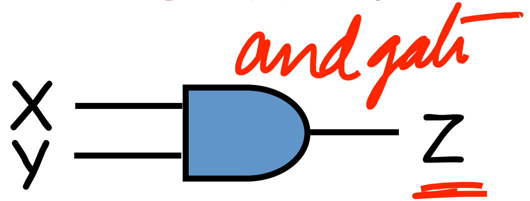
→ Truth table representation

| x | y | z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

enumerate input combinations

# What is the Output Of This Gate?

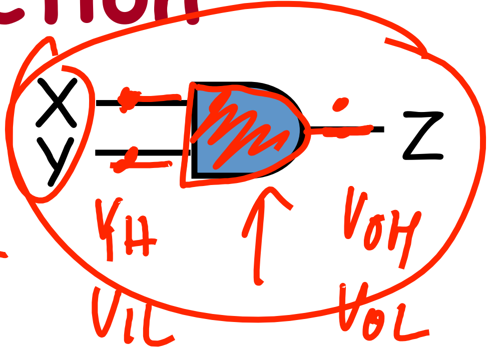



$$Z = X \cdot Y$$

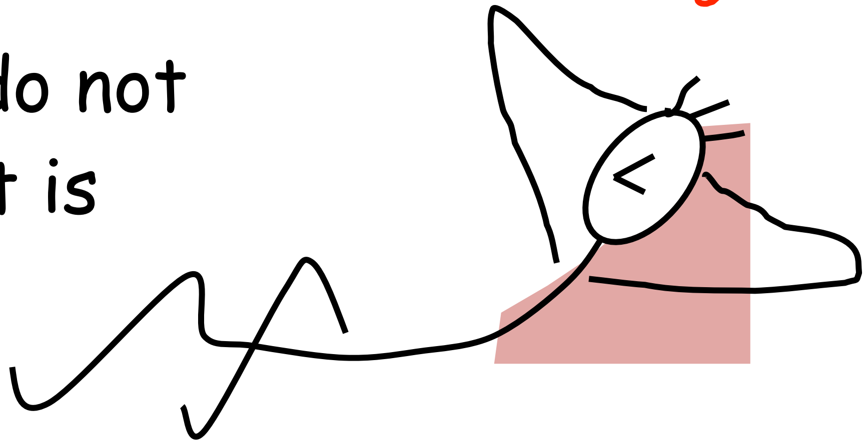
# Combinational Gate Abstraction

- Adheres to static discipline
- Outputs are a function of inputs alone.

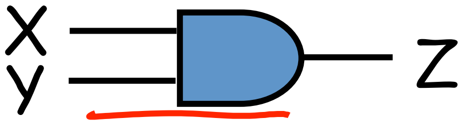



$Z = X \cdot Y$

Digital logic designers do not have to care about what is inside a gate.



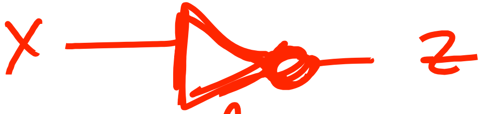
# Logic Gates




$Z = X \cdot Y$

AND gate

| X | Y | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

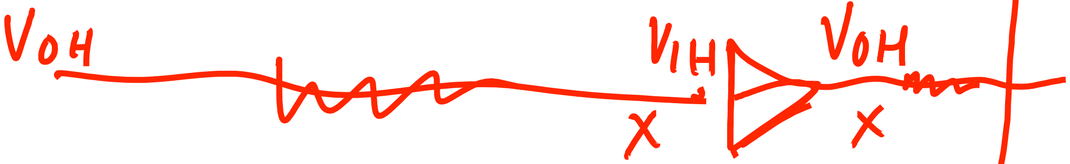


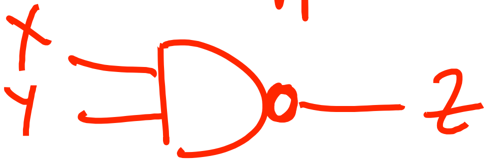

Inversion

NOT gate
inverter

$Z = \overline{X}$

| X | Z |
|---|---|
| 0 | 1 |
| 1 | 0 |


NAND gate

$Z = \overline{X \cdot Y}$

| X | Y | Z |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

# Another Gate Example

If (A is true) OR (B is true)

then C is true

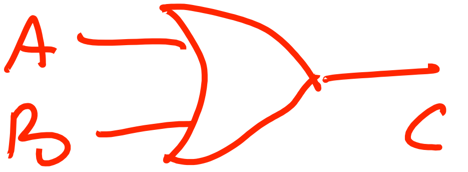
else C is false

~~gates~~

→ $C = A + B$ boolean eq

OR

| A | B | C |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

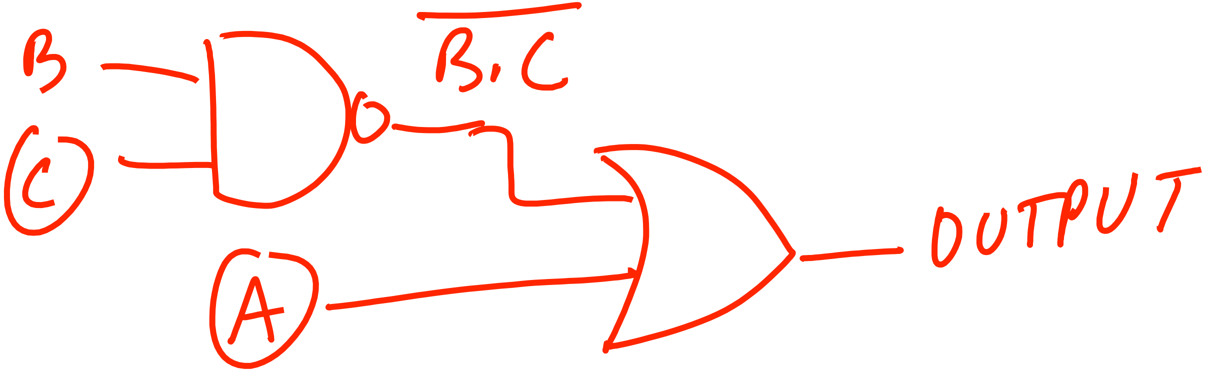
→ 


OR gate

# Digital Circuits

Implement: $\boxed{\text{output} = A + \overline{B \cdot C}}$

0, 1



# Representing Numbers

5 999

Numbers larger than 1 can be represented using multiple binary digits and coding, much like using multiple decimal digits to represent numbers greater than 9.

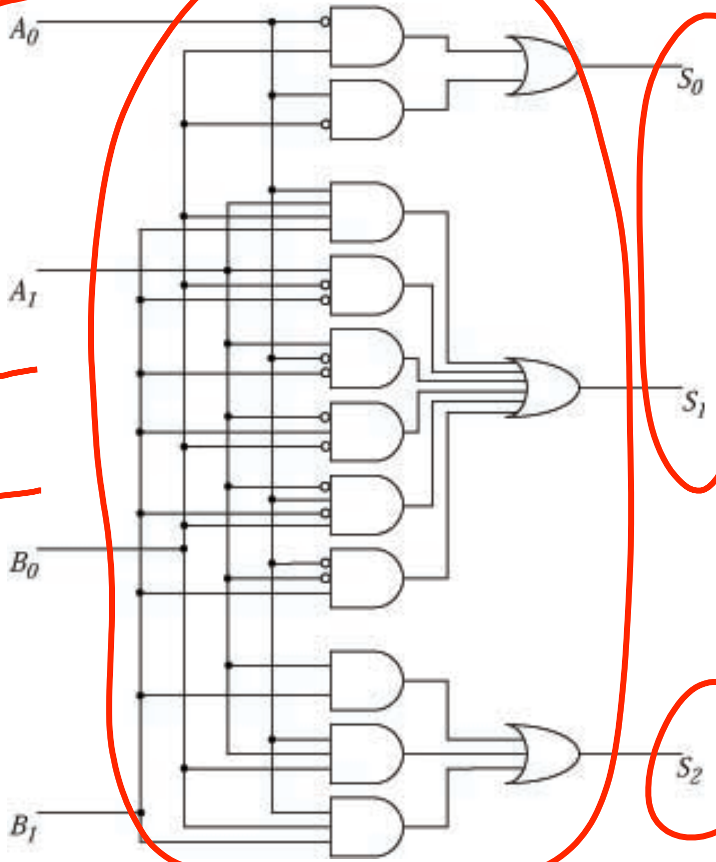
decimal

$$921_{base10} = 9\times\underbrace{10^2}_{100} + 2\times\underbrace{10^1}_{10} + 1\times\underbrace{10^0}_{1}$$

The binary number 101 has decimal value:

$$101_{base\,2} = 1\times\underbrace{2^2}_{4} + 0\times\underbrace{2^1}_{2} + 1\times\underbrace{2^0}_{1} = 5_{base\,10}$$

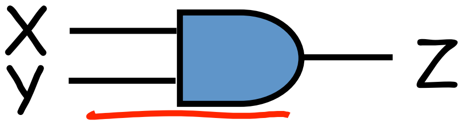
# A Two-Bit Adder Circuit




adder

digital design
6.004x

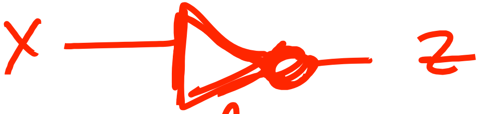
# Logic Gates




$Z = X \cdot Y$

AND gate

| X | Y | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

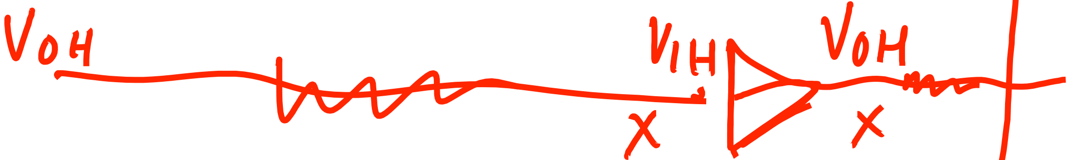


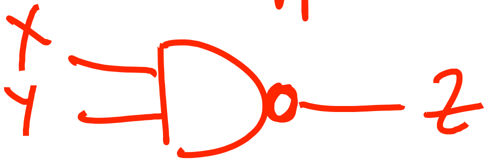

Inversion

NOT gate
inverter
$Z = \overline{X}$

| X | Z |
|---|---|
| 0 | 1 |
| 1 | 0 |


buffer




NAND gate
$Z = \overline{X \cdot Y}$

| X | Y | Z |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

# Another Gate Example

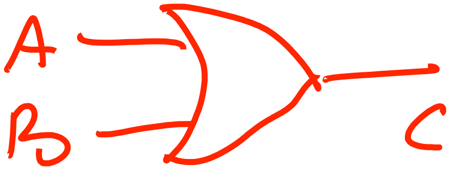
If (A is true) OR (B is true)
then C is true
else C is false

~~gates~~

→ $C = A + B$ boolean eq

OR

| A | B | C |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

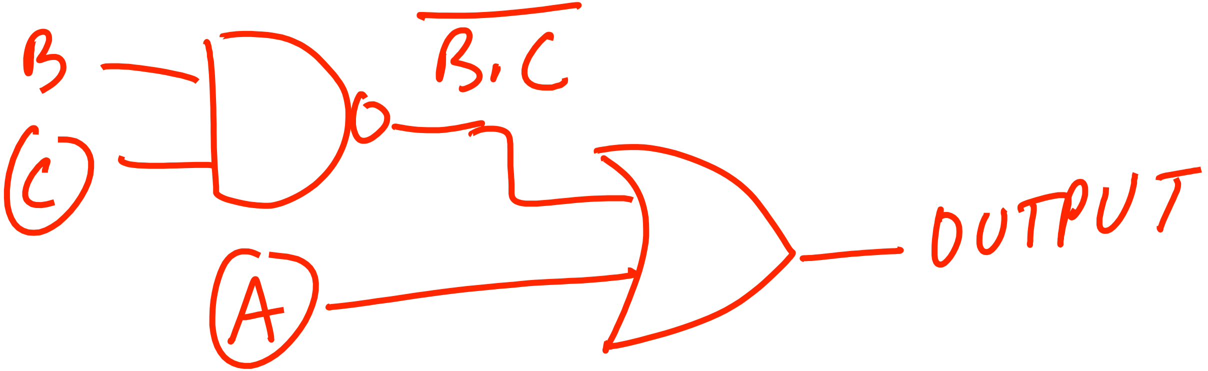
→ 


OR gate

# Digital Circuits

Implement: $\boxed{\text{output} = A + \overline{B \cdot C}}$

0, 1



# Representing Numbers

5 999

Numbers larger than 1 can be represented using multiple binary digits and coding, much like using multiple decimal digits to represent numbers greater than 9.

decimal $921_{base10} = 9 \times 10^2 + 2 \times 10^1 + 1 \times 10^0$

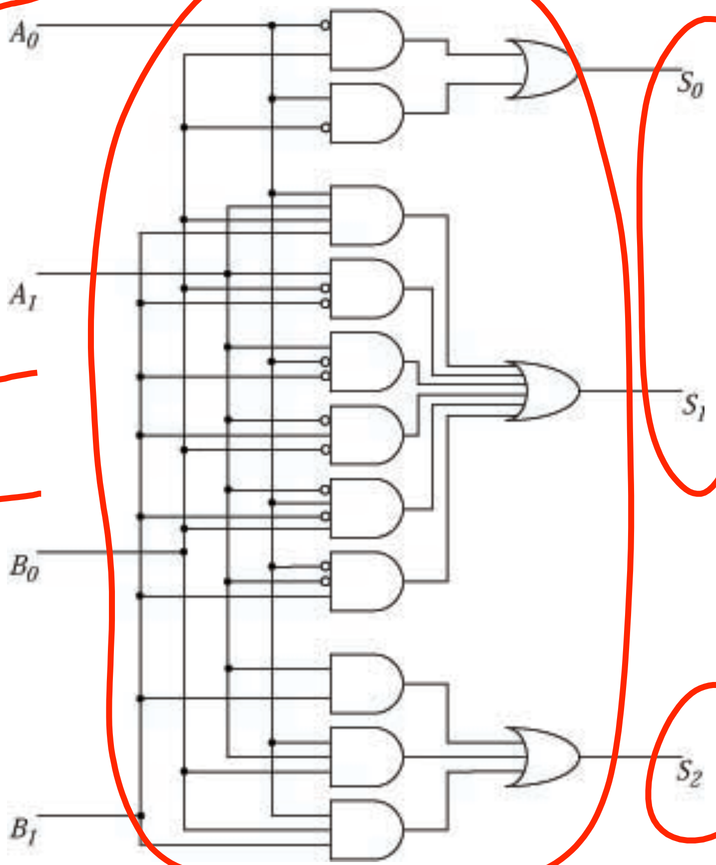
100 10 1

The binary number 101 has decimal value:

$101_{base 2} = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 5_{base 10}$

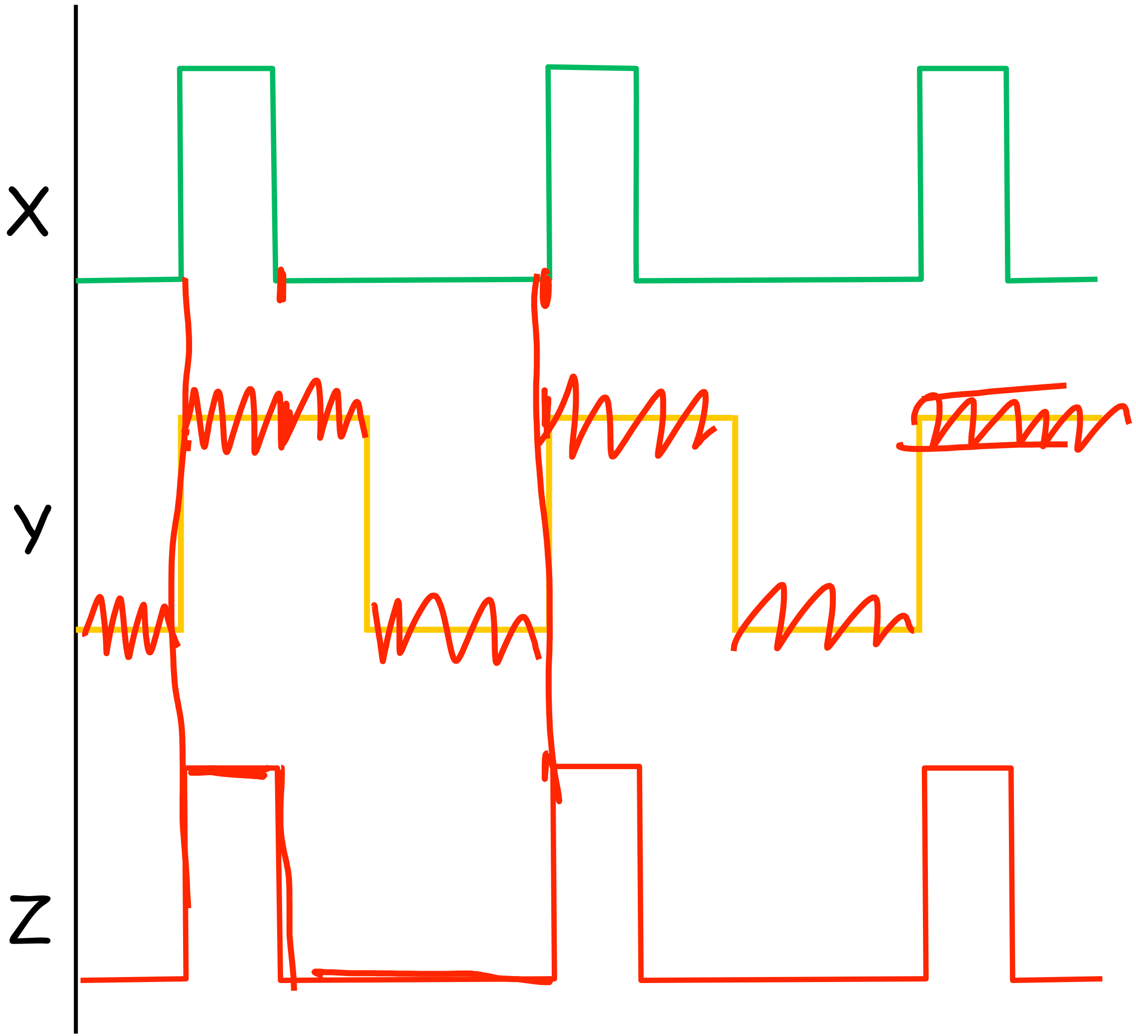
4 2 1

# A Two-Bit Adder Circuit



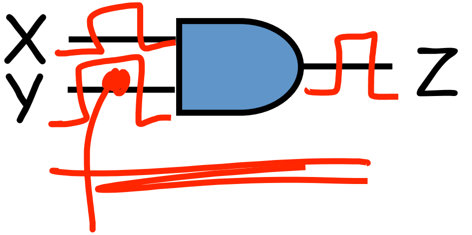

adder

digital design
6.004x




Demo

$Z = X \cdot Y$




noise