# VE270 RC3

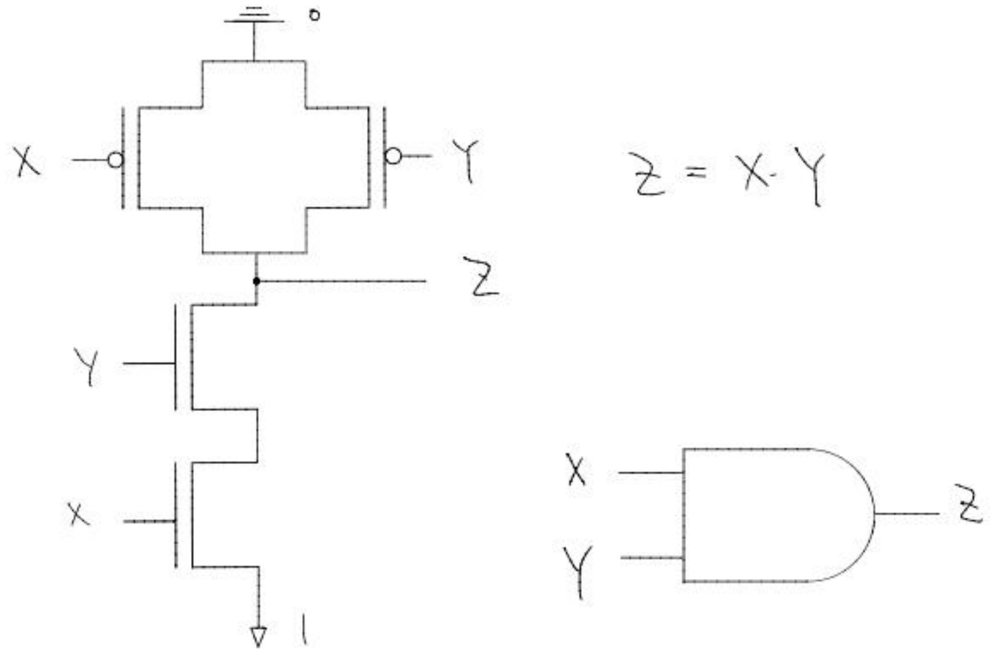
VE270 TA GROUP

2020.6.3

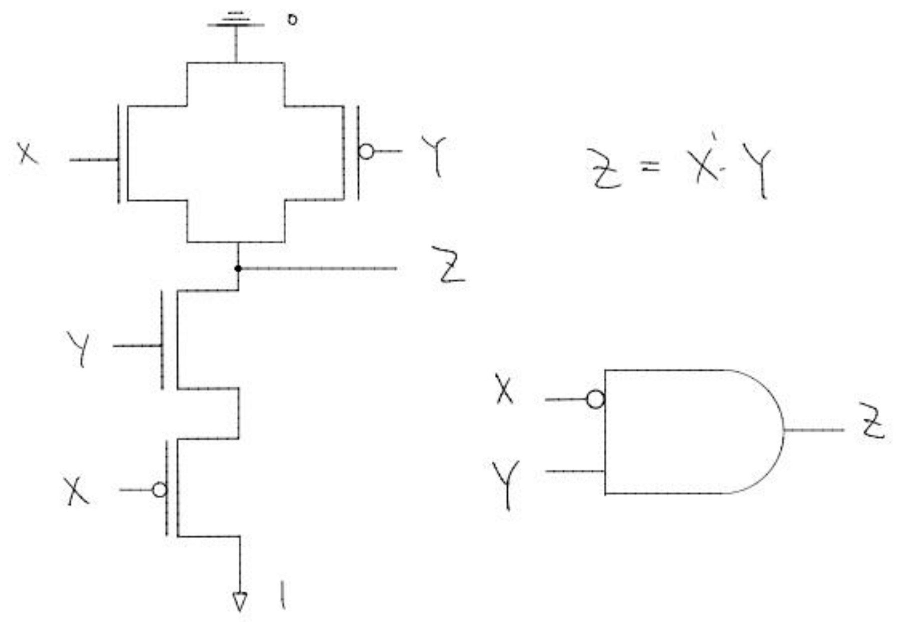
## Feedback of Quiz 2 and Homework 2

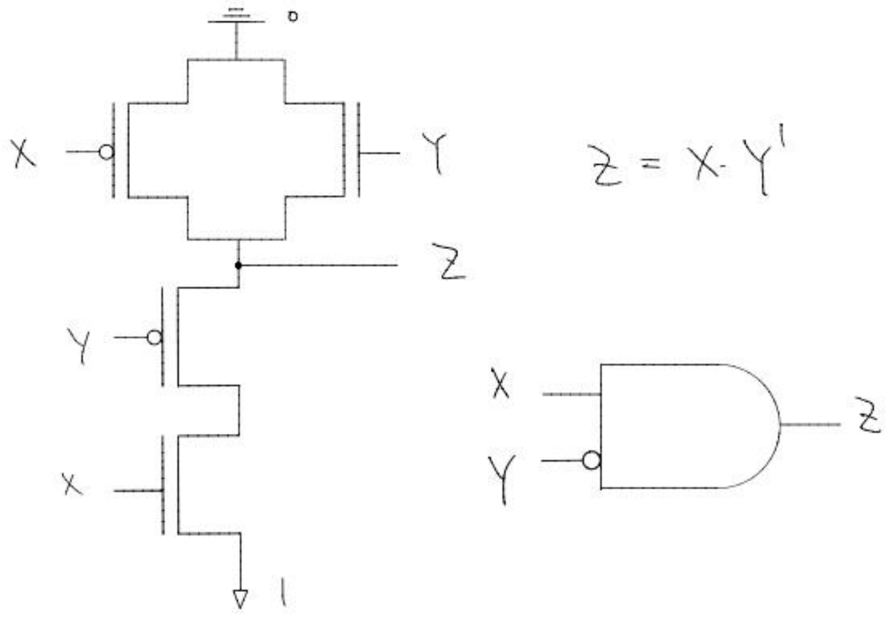
1. When the problem doesn't specify, inverters are always neglected when counting gate delay and number of transistors.

We take AND gate as an example.

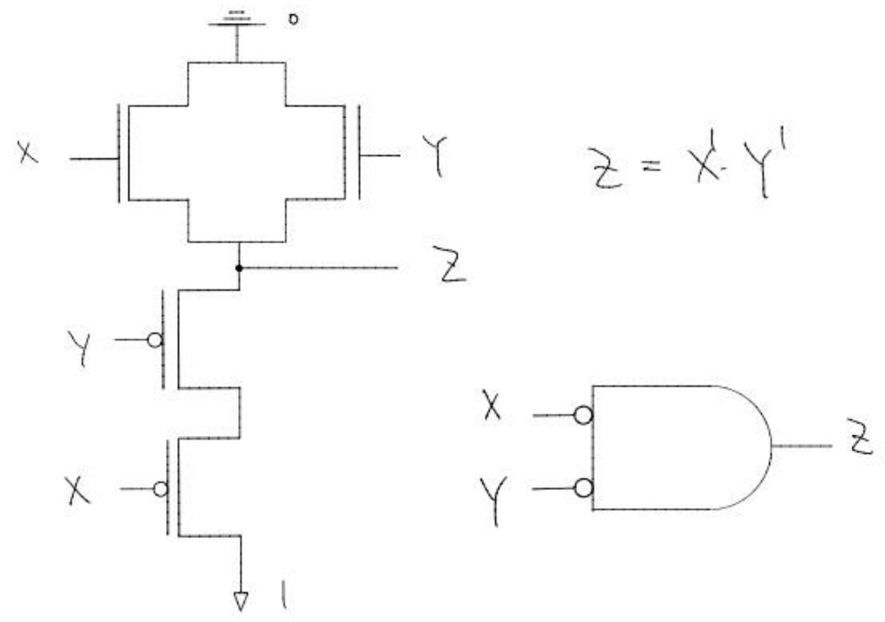


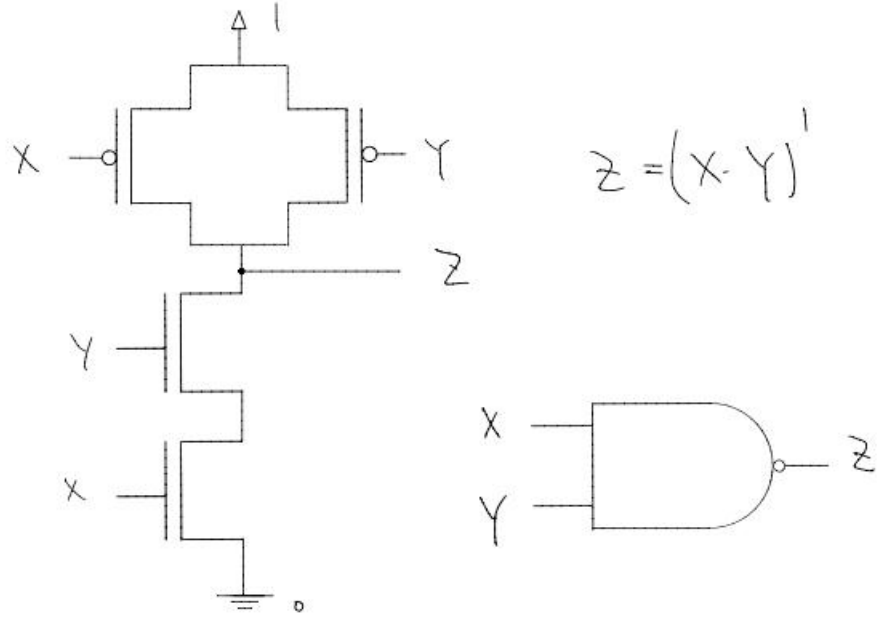
# Feedback of Quiz 2 and Homework 2





# Feedback of Quiz 2 and Homework 2



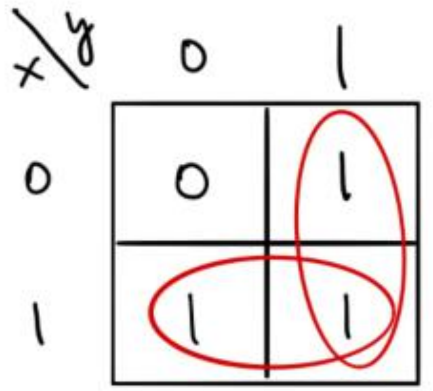


# Feedback of Quiz 2 and Homework 2

2. Arithmatic method to simplify boolean equations

Do not forget: $x + x'y = x + y$ $x + xy = x$

You can also use K-map to verify your solution



$$F = x + xy + x'y$$
$$= x + y$$

# Outline

1. Combinational Circuit
2. Building Blocks
    1. MUX
    2. Half Adder
    3. Full Adder
    4. Carry-ripple Adder
    5. Encoder/Decoder
    6. Tri-state Buffer
    7. ALU
3. SR latch

## Combinational Circuit

- Depends only upon the present combination of its inputs
- Output changes only and immediately when inputs change
- Sequential Circuit: feedback

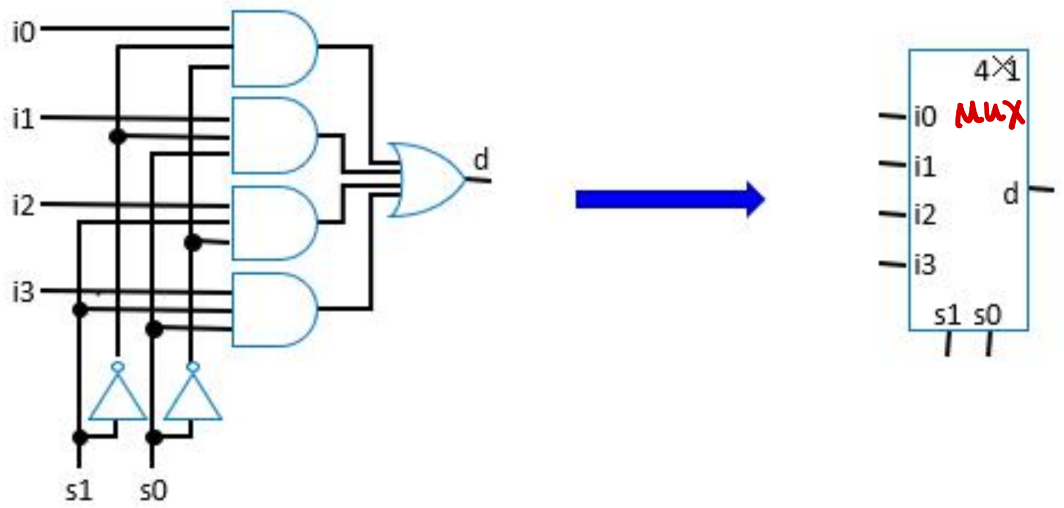
## Labeling

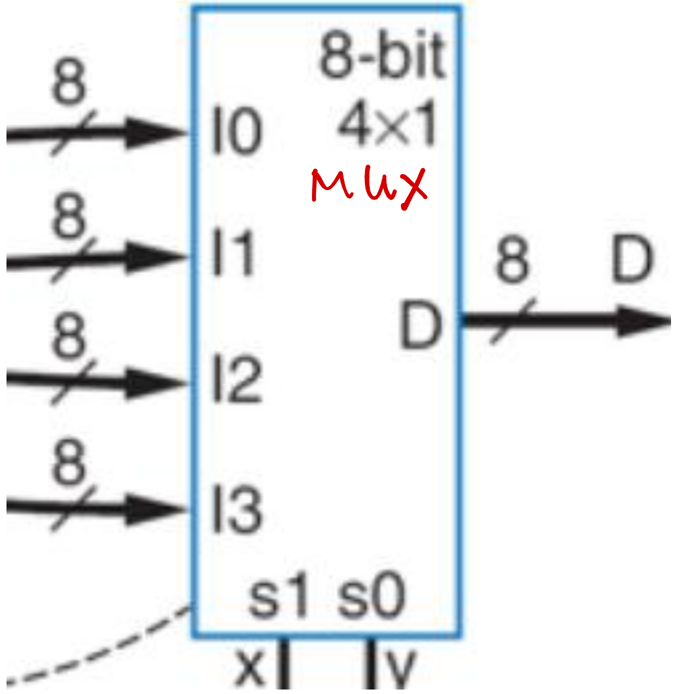
1. The name of the block, e.g. Full Adder, Half Adder, MUX, Decoder…
2. The number of bits if it is more than 1
3. The number of inputs and outputs for MUX, Decoder

In addition, you should name all the input and output in the block according to the tradition.

# Mux

- Use select bits to determine the route for the output
- N inputs and $\log_2(N)$ select bits → 1 output

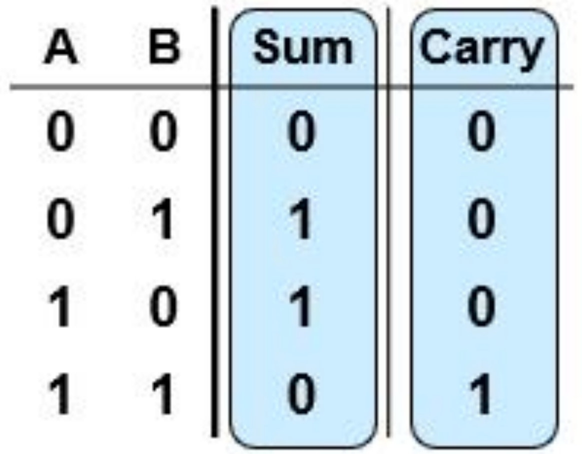




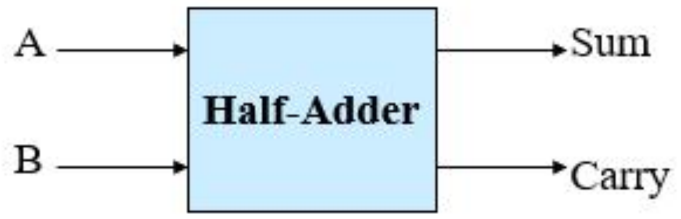
**Label is important!!!!!**

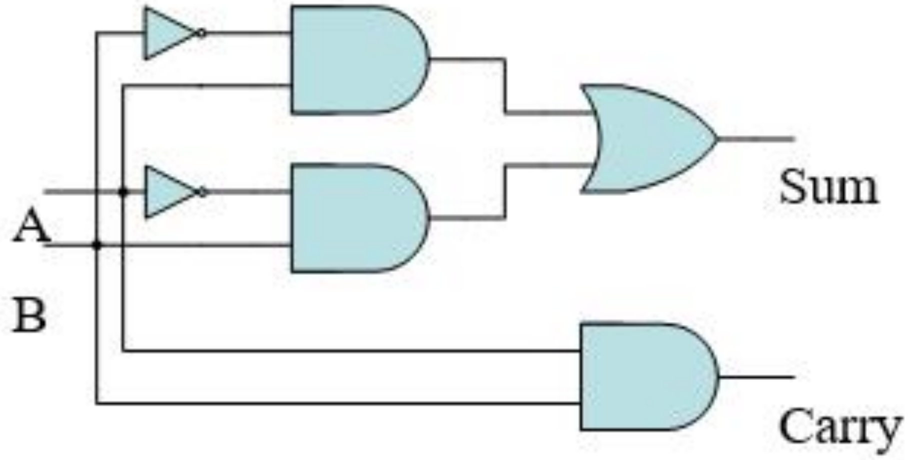
# Half Adder

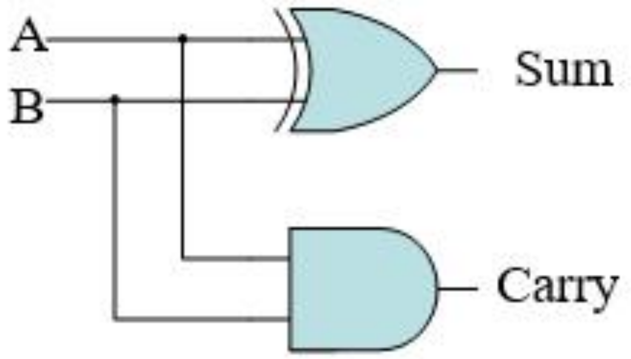
- Addition of two bits



| A | B | Sum | Carry |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

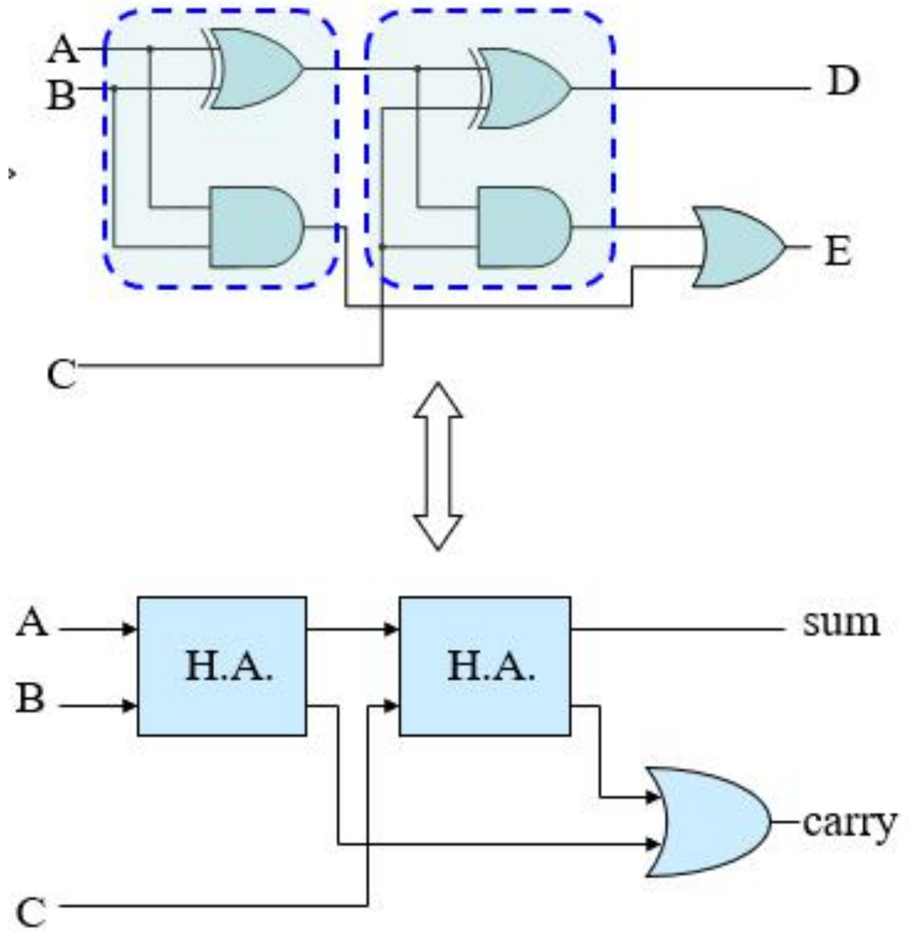






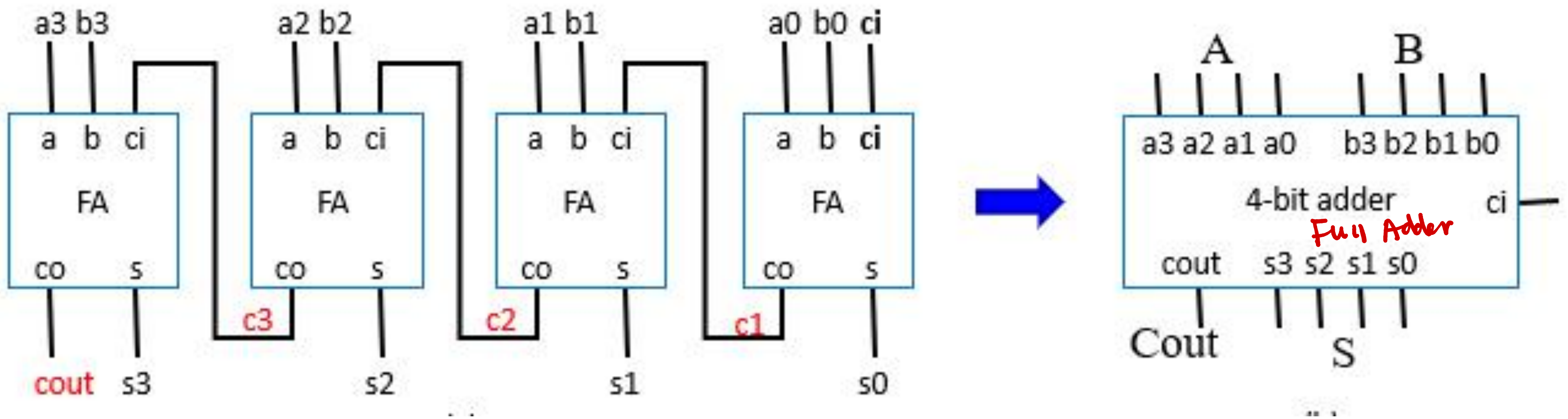
## Full Adder

- Addition for 3 bits

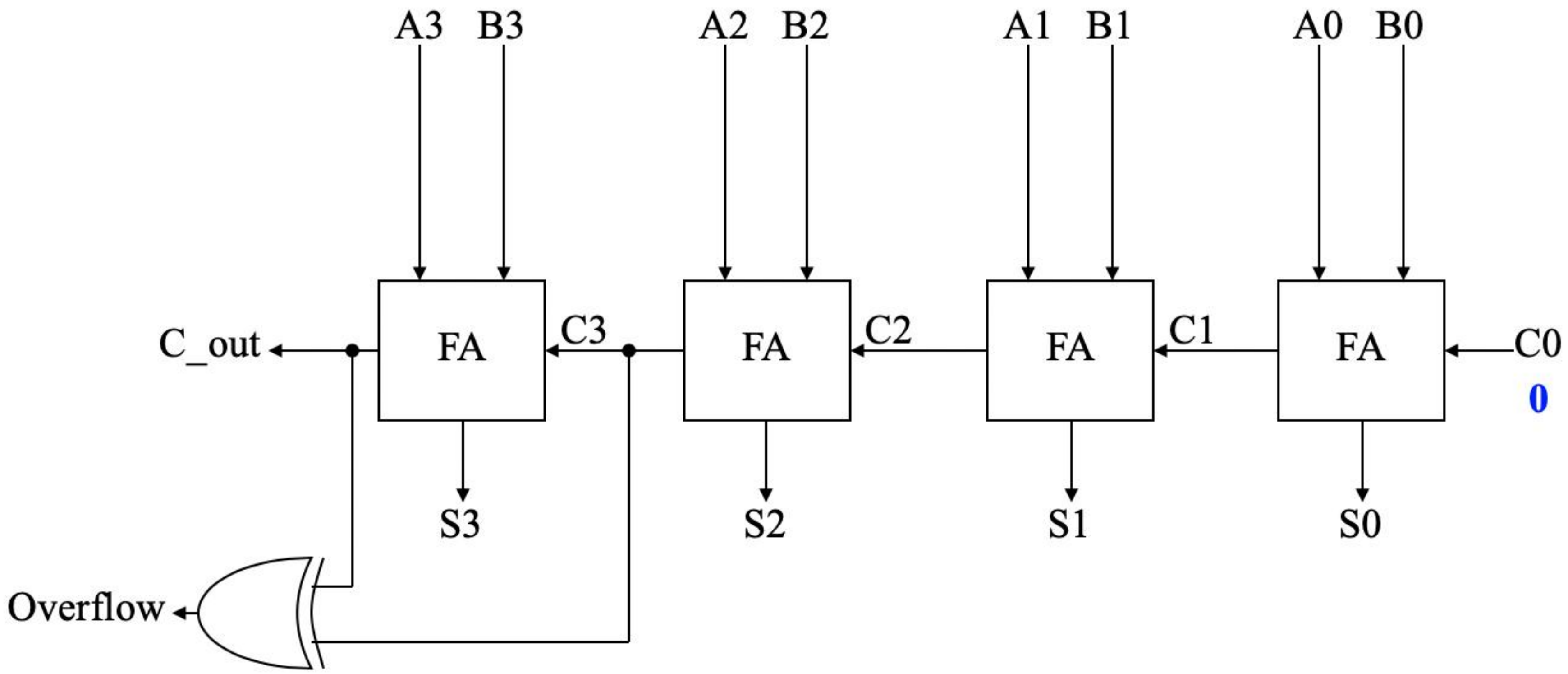


# Carry-ripple Adder

- Combination of FAs
- Easily built for adding N bits

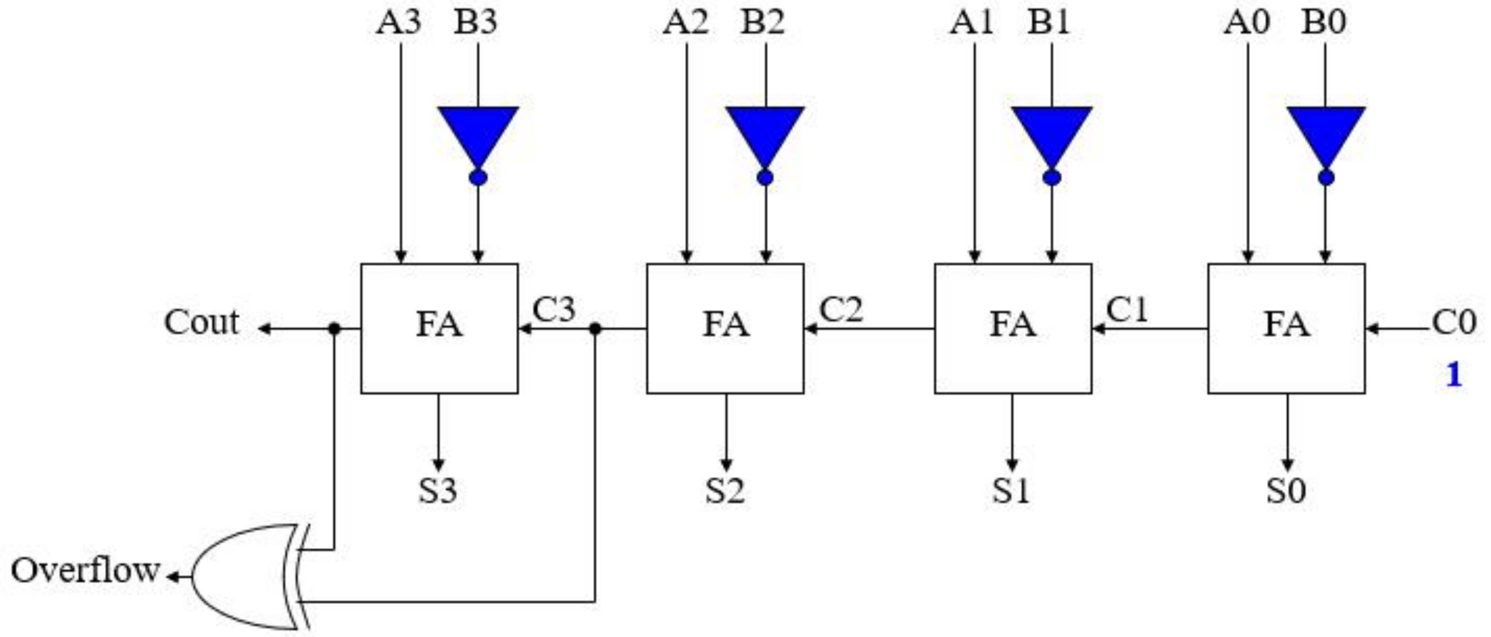


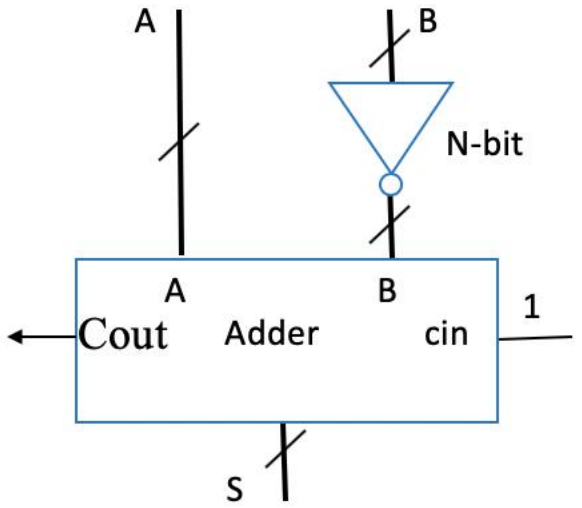
# 4-Bit 2's complement Adder



# 4-Bit 2's complement Subtractor

A - B = A + inverse(B) + 1





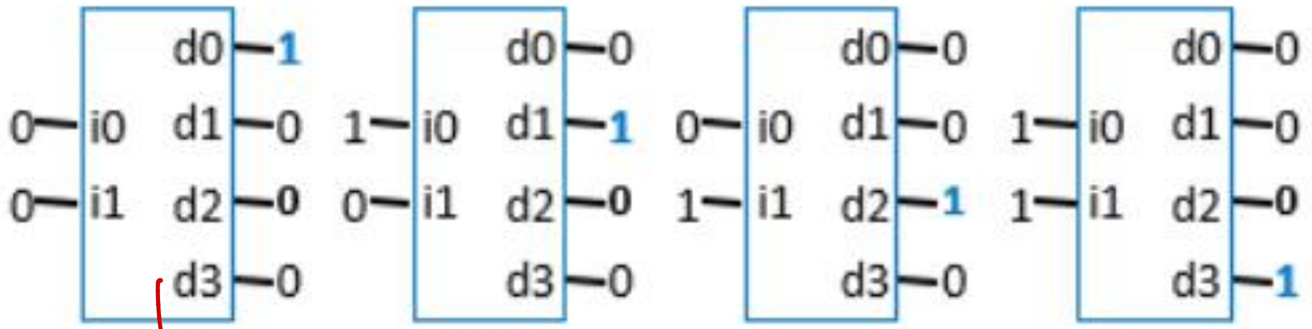
## Encoder

| Inputs | | | | | | | | Outputs | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $D_0$ | $D_1$ | $D_2$ | $D_3$ | $D_4$ | $D_5$ | $D_6$ | $D_7$ | x | y | z |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

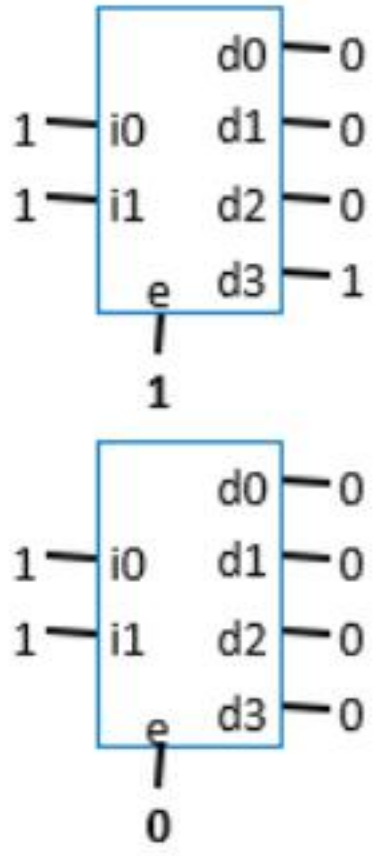
Not recommended to use since the output is undefined when multiple input is turned on.

## Decoder

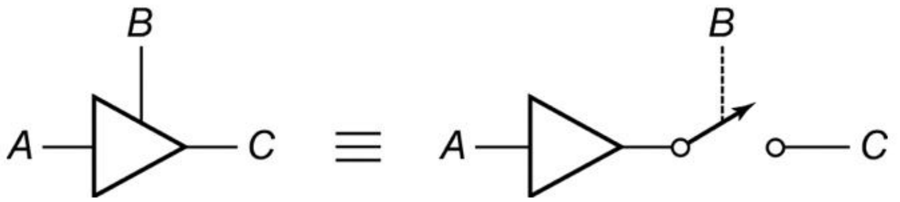
- N inputs, 2^N outputs
- Enable e
- **Label is important !!!**


0 i0 0 i1 d0 1 d1 0 d2 0 d3 0
1 i0 0 i1 d0 0 d1 1 d2 0 d3 0
0 i0 1 i1 d0 0 d1 0 d2 1 d3 0
1 i0 1 i1 d0 0 d1 0 d2 0 d3 1

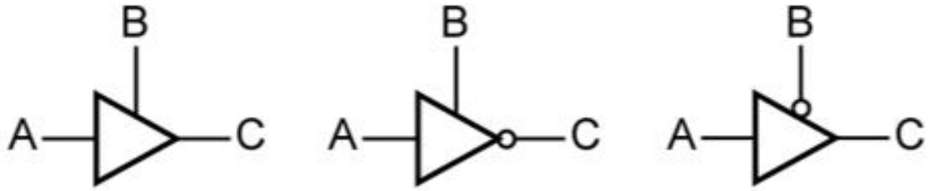
2×4
decoder.


1 i0 1 i1 e 1 d0 0 d1 0 d2 0 d3 1
1 i0 1 i1 e 0 d0 0 d1 0 d2 0 d3 0

## Tri-state Buffer



Open Circuit

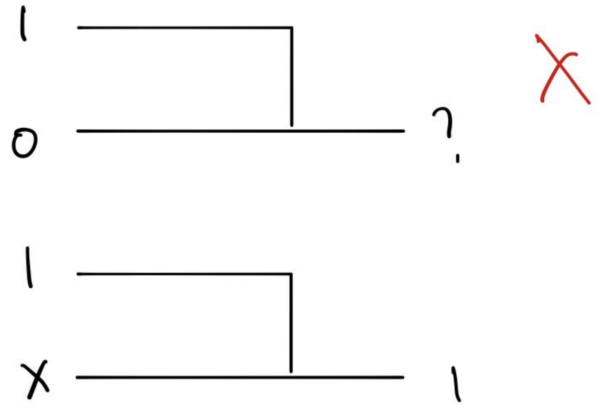


| B | A | C |
|---|---|---|
| 0 | 0 | Z |
| 0 | 1 | Z |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

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

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

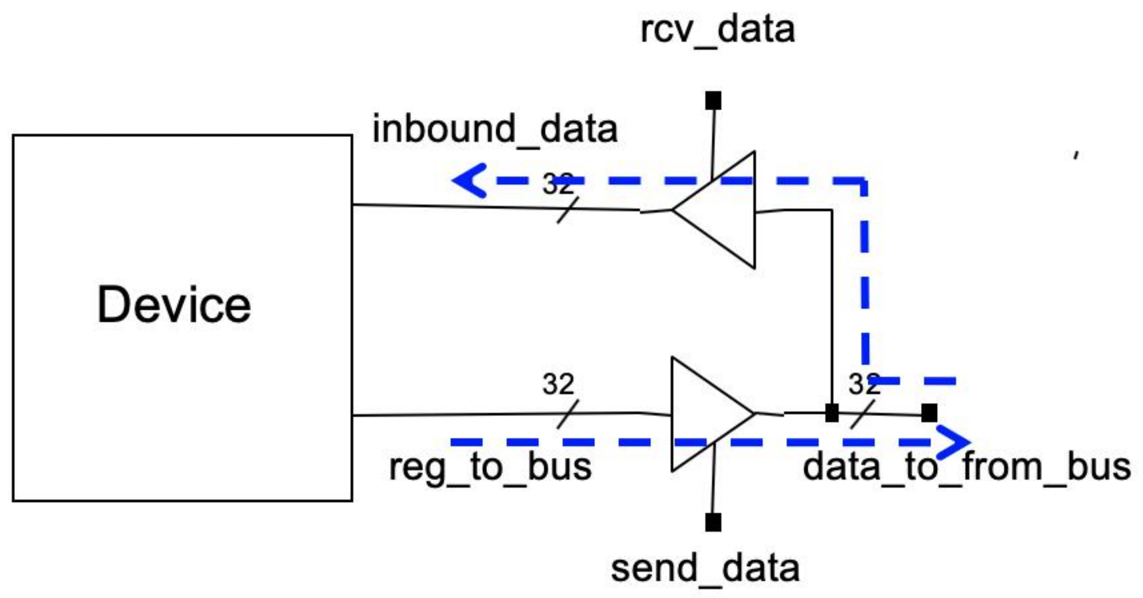
Difference between 0 and Z



# Tri-state Buffer Application

Bidirectional I/O Port

Aviod Multi-driver



# ALU

- Mux
- Decoder + tri-state

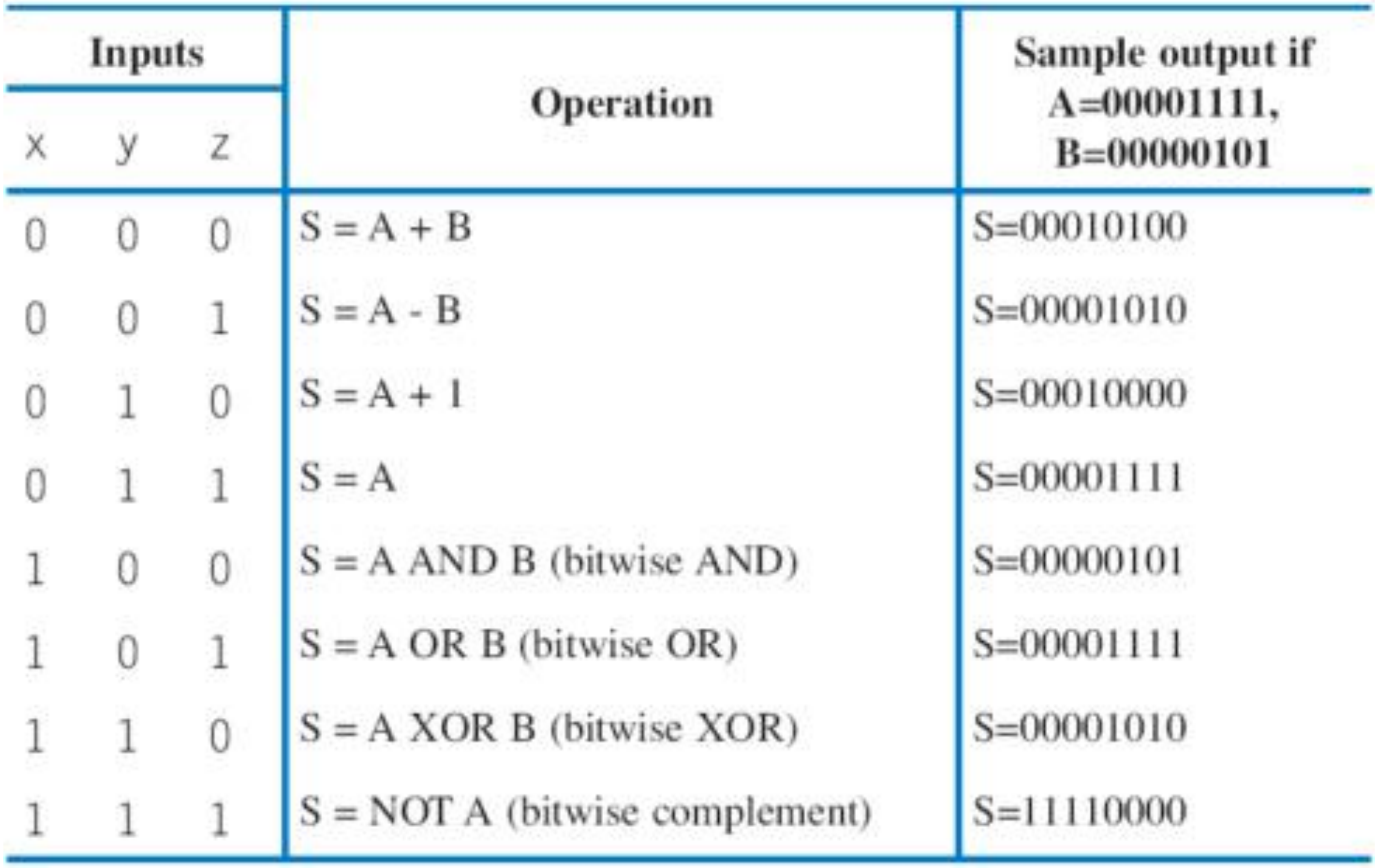


TABLE 4.2 Desired calculator operations

| Inputs | | | Operation | Sample output if A=00001111, B=00000101 |
|---|---|---|---|---|
| x | y | z | | |
| 0 | 0 | 0 | S = A + B | S=00010100 |
| 0 | 0 | 1 | S = A - B | S=00001010 |
| 0 | 1 | 0 | S = A + 1 | S=00010000 |
| 0 | 1 | 1 | S = A | S=00001111 |
| 1 | 0 | 0 | S = A AND B (bitwise AND) | S=00000101 |
| 1 | 0 | 1 | S = A OR B (bitwise OR) | S=00001111 |
| 1 | 1 | 0 | S = A XOR B (bitwise XOR) | S=00001010 |
| 1 | 1 | 1 | S = NOT A (bitwise complement) | S=11110000 |

## Exercise

2.54 A museum has three rooms, each with a motion sensor (`m0`, `m1`, and `m2`) that outputs `1` when motion is detected. At night, the only person in the museum is one security guard who walks from room to room. Create a circuit that sounds an alarm (by setting an output `A` to `1`) if motion is ever detected in more than one room at a time (i.e., in two or three rooms), meaning there must be one or more intruders in the museum. Start with a truth table.

Using Building blocks

You can use:

MUX

Decoder

Half Adder

Full Adder

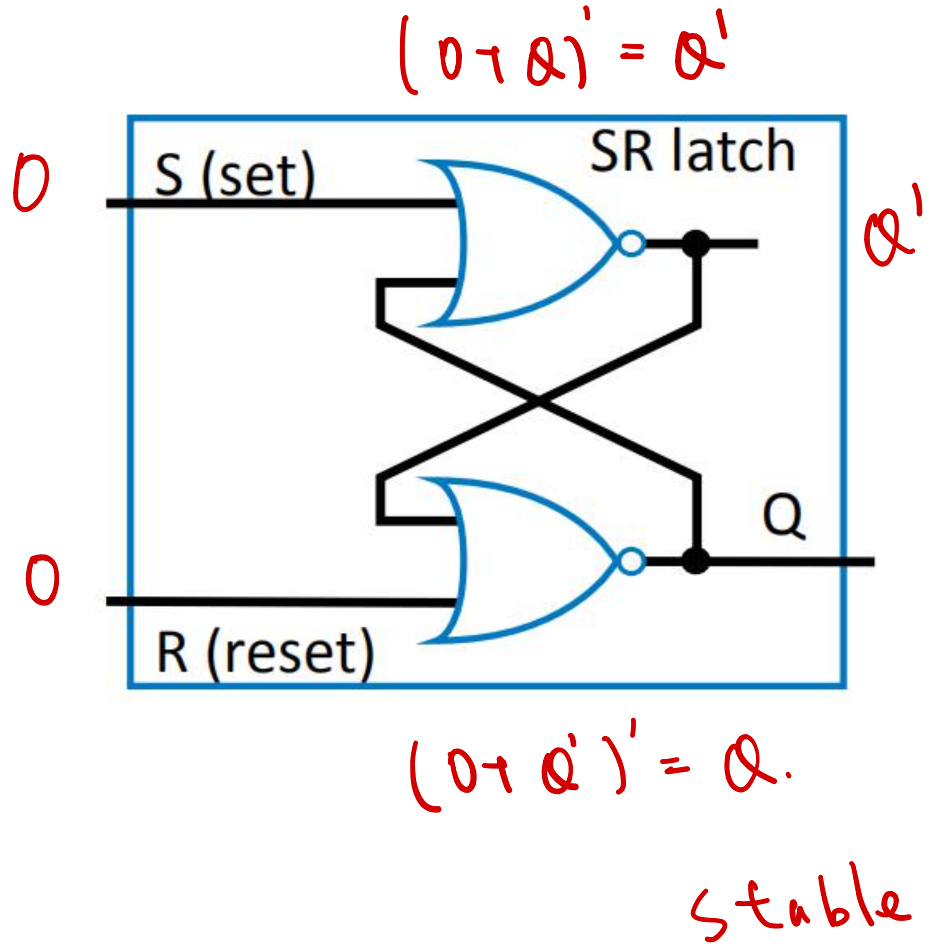
## Sequential Circuit

- A digital circuit whose output depends not only upon the present input values, but also the history of input and output values.

Non-ideal gate behavior: delay

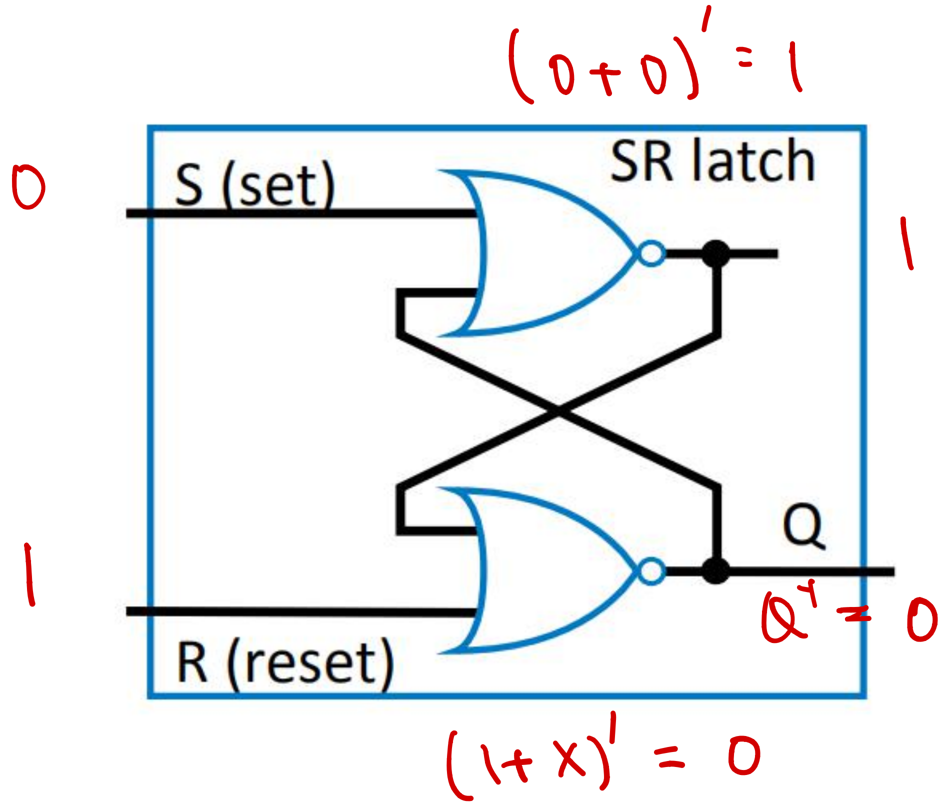
Outputs don't change immediately after inputs change

# SR latch



| S(t) | R(t) | Q(t) | Q(t+Δ) $\longrightarrow Q^+$ |
|---|---|---|---|
| 0 | 0 | 0 | |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | |
| 1 | 0 | 1 | |
| 1 | 1 | 0 | |
| 1 | 1 | 1 | |

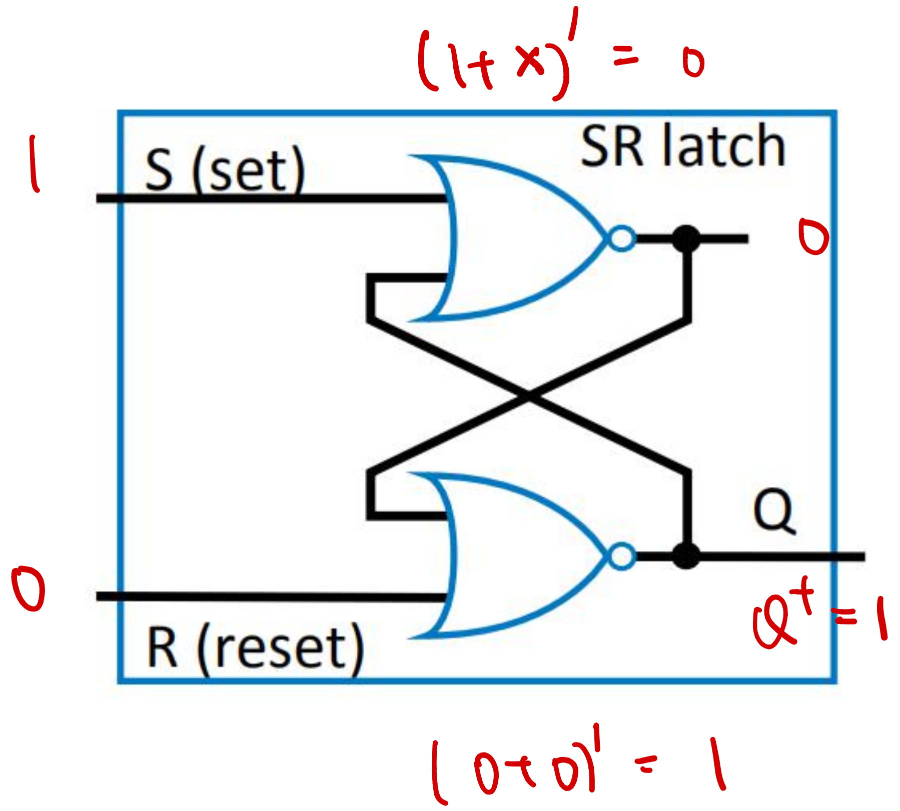
# SR latch



| S(t) | R(t) | Q(t) | $Q(t+\Delta) \longrightarrow Q^+$ | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | hold |
| 0 | 0 | 1 | 1 | |
| 0 | 1 | 0 | | |
| 0 | 1 | 1 | | |
| 1 | 0 | 0 | | |
| 1 | 0 | 1 | | |
| 1 | 1 | 0 | | |
| 1 | 1 | 1 | | |

stable after release $(S=0, R=0)$

## SR latch

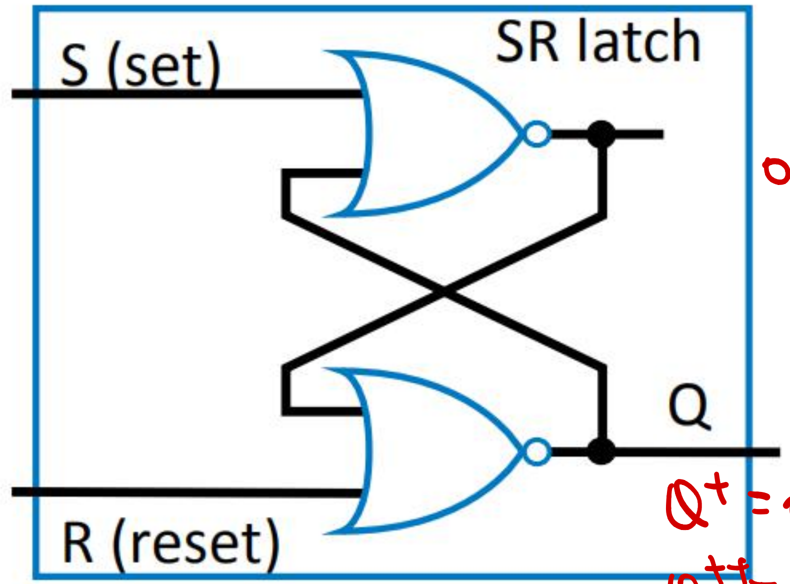


Stable after release

| S(t) | R(t) | Q(t) | Q(t+Δ) $\longrightarrow Q^+$ | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | hold |
| 0 | 0 | 1 | 1 | |
| 0 | 1 | 0 | 0 | reset |
| 0 | 1 | 1 | 0 | |
| 1 | 0 | 0 | | |
| 1 | 0 | 1 | | |
| 1 | 1 | 0 | | |
| 1 | 1 | 1 | | |

## SR latch

$(1+x)' = 0$



1

1

$0 \to 1 \to 0 \cdots$

$Q^+ = 0$

$Q^{++} = 1$

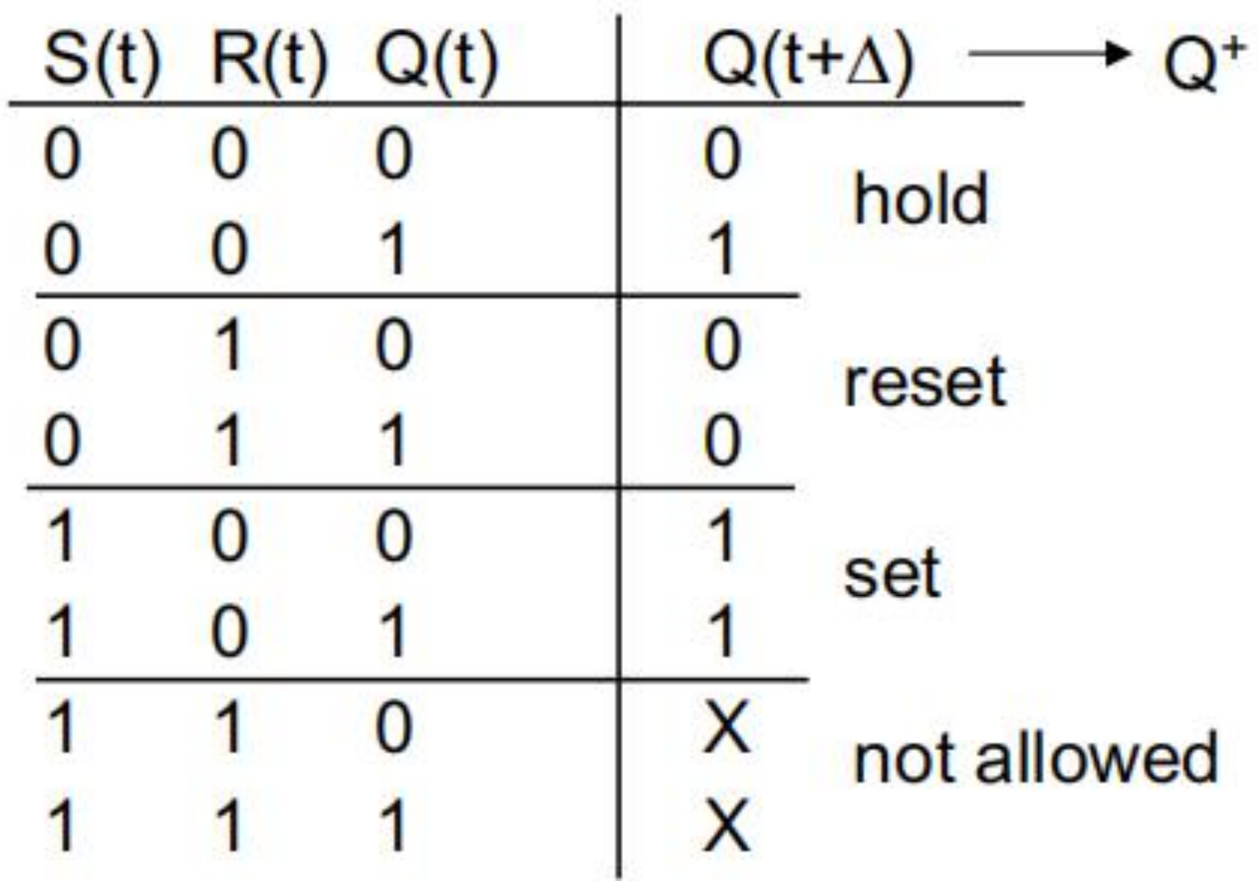
$\vdots$

$(1+x)' = 0$

| S(t) | R(t) | Q(t) | Q(t+Δ) $\to Q^+$ | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | hold |
| 0 | 0 | 1 | 1 | |
| 0 | 1 | 0 | 0 | reset |
| 0 | 1 | 1 | 0 | |
| 1 | 0 | 0 | 1 | set |
| 1 | 0 | 1 | 1 | |
| 1 | 1 | 0 | | |
| 1 | 1 | 1 | | |

Oscillate after release. unstable

## SR latch



| S(t) | R(t) | Q(t) | $Q(t+\Delta)$ | $\longrightarrow Q^+$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | hold |
| 0 | 0 | 1 | 1 | |
| 0 | 1 | 0 | 0 | reset |
| 0 | 1 | 1 | 0 | |
| 1 | 0 | 0 | 1 | set |
| 1 | 0 | 1 | 1 | |
| 1 | 1 | 0 | X | not allowed |
| 1 | 1 | 1 | X | |

Thank you