The logic of computers

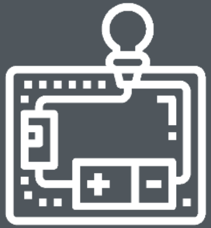
starting in 5:00

"To be, or not to be …"

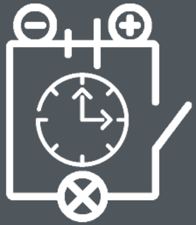
- Shakespeare

Dr. Goran Soldar
Dr. Khuong An Nguyen

Boolean algebra



Combinational circuit

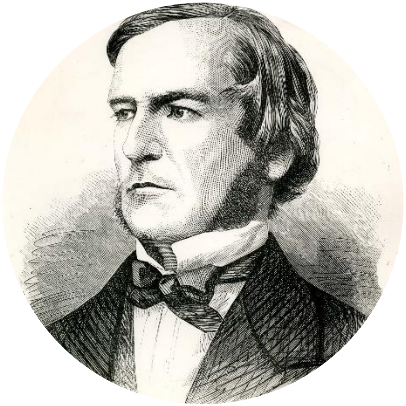


Sequential circuit

1. Boolean algebra.
2. Combinational circuit.
3. Sequential circuit.

Boolean algebra

1. Algebra on 0 and 1.
2. Has its root in philosophy.



1. What we start

Axioms: assuming basic objects and operations are true.

2. What we derive

Laws and theorems: manipulating Boolean expressions

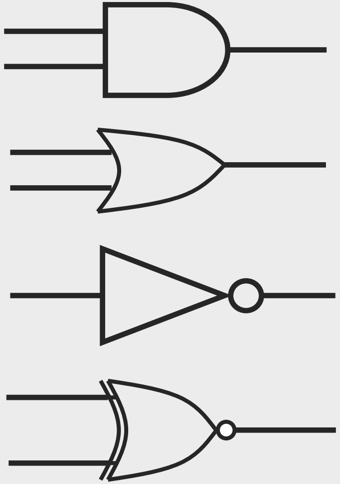
3. What we build

Circuit: deriving complex digital design.

Further reading

“The Mathematical analysis of Logic”. George Boole, 1847

Ordinary vs Boolean algebra

	Ordinary algebra	Boolean algebra
Inputs	1	false 0
Operators		 AND OR NOT XOR

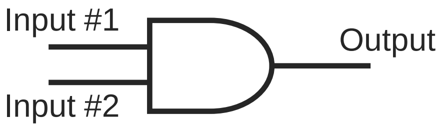
Boolean algebra

Combinational logic

Sequential logic

AND operation

Symbol



Boolean Expression

Given Input #1 = A; Input #2 = B; Output = Q

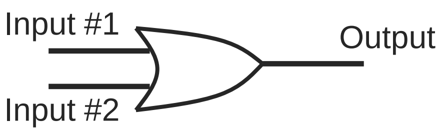
$$Q = A \bullet B$$

Truth table

Input #1	Input #2	Output
True (1)	True (1)	True (1)
True (1)	False (0)	False (0)
False (0)	True (1)	False (0)
False (0)	False (0)	False (0)

OR operation

Symbol



Boolean Expression

Given Input #1 = A; Input #2 = B; Output = Q

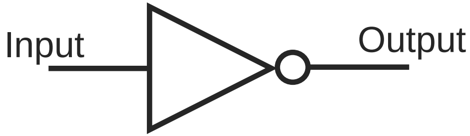
$$Q = A + B$$

Truth table

Input #1	Input #2	Output
True (1)	True (1)	True (1)
True (1)	False (0)	True (1)
False (0)	True (1)	True (1)
False (0)	False (0)	False (0)

NOT operation

Symbol



Expression

Given Input = A; Output = Q

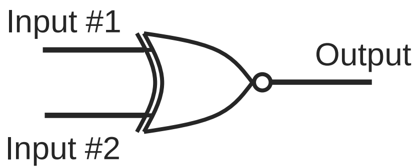
$$Q = \overline{A}$$

Truth table

Input	Output
True (1)	False (0)
False (0)	True (1)

XOR (Exclusive-OR) operation

Symbol



Boolean Expression

Given Input #1 = A; Input #2 = B; Output = Q

$$Q = A \oplus B = \overline{A}\,B + A\,\overline{B}$$

Truth table

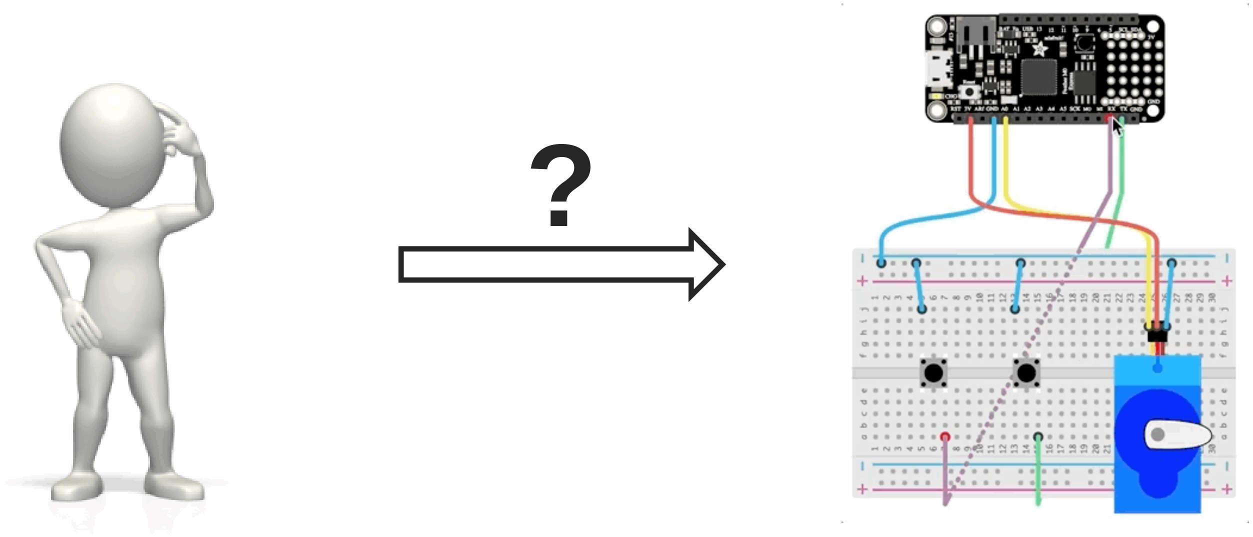
Input #1	Input #2	Output
True (1)	True (1)	False (0)
True (1)	False (0)	True (1)
False (0)	True (1)	True (1)
False (0)	False (0)	False (0)

Boolean algebra

Combinational logic

Sequential logic

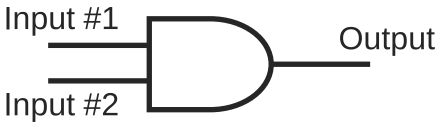
Designing a circuit



Truth table

1. Lists all possible inputs and outputs.
2. Two circuits are identical, if they share the same truth table.

AND operator



Truth table

Input #1	Input #2	Output
True (1)	True (1)	True (1)
True (1)	False (0)	False (0)
False (0)	True (1)	False (0)
False (0)	False (0)	False (0)

Step 1: Writing the truth table

Specification:

A circuit that accepts 3 inputs, and returns True, only when one input is False.

Writing the truth table:

A circuit with n inputs will always have 2^n rows (possibilities).

Truth table

Input A	Input B	Input C	Output
True (1)	True (1)	True (1)	False (0)
True (1)	True (1)	False (0)	True (1)
True (1)	False (0)	True (1)	True (1)
True (1)	False (0)	False (0)	False (0)
False (0)	True (1)	True (1)	True (1)
False (0)	True (1)	False (0)	False (0)
False (0)	False (0)	True (1)	False (0)
False (0)	False (0)	False (0)	False (0)

Step 2: Deriving Boolean expressions

1. Find all rows with output True (1).

2. **For each row:**

   if the input X is 1, write X

   if the input X is 0, write $\overline{X}$

Truth table

Input A	Input B	Input C	Output	Expression
True (1)	True (1)	True (1)	False (0)	
True (1)	True (1)	False (0)	True (1)	$A.B.\overline{C}$
True (1)	False (0)	True (1)	True (1)	$A.\overline{B}.C$
True (1)	False (0)	False (0)	False (0)	
False (0)	True (1)	True (1)	True (1)	$\overline{A}.B.C$
False (0)	True (1)	False (0)	False (0)	
False (0)	False (0)	True (1)	False (0)	
False (0)	False (0)	False (0)	False (0)	

Boolean algebra

Combinational logic

Sequential logic

Step 3: Sum of the products

OR (+) all the product of the rows together.

$$A.B.\overline{C} + A.\overline{B}.C + \overline{A}.B.C$$

Truth table

Input A	Input B	Input C	Output	Expression
True (1)	True (1)	True (1)	False (0)	
True (1)	True (1)	False (0)	True (1)	$A.B.\overline{C}$
True (1)	False (0)	True (1)	True (1)	$A.\overline{B}.C$
True (1)	False (0)	False (0)	False (0)	
False (0)	True (1)	True (1)	True (1)	$\overline{A}.B.C$
False (0)	True (1)	False (0)	False (0)	
False (0)	False (0)	True (1)	False (0)	
False (0)	False (0)	False (0)	False (0)	

Expression simplification

Axioms

- Closure, given a, b ∈ B :
  $a + b \in B$
  $a \bullet b \in B$

- Commutative, given a, b ∈ B
  $a + b = b + a$
  $a \bullet b = b \bullet a$

- Identity, given 0, 1 ∈ B :
  $a + 0 = a$
  $a \bullet 1 = a$

- Distributive :
  $a + (b \bullet c) = (a + b) \bullet (a + c)$
  $a \bullet (b + c) = a \bullet b + a \bullet c$

- Complement :
  $a + \bar{a} = 1$
  $a \bullet \bar{a} = 0$

Laws

- Operation with 0 and 1 :
  $X + 0 = X$ $\quad X \bullet 1 = X$
  $X + 1 = 1$ $\quad X \bullet 0 = 0$
- Idempotent :
  $X + X = X$ $\quad X \bullet X = X$
- Involution :
  $\overline{(\bar{X})} = X$
- Complementarity :
  $X + \bar{X} = 1$ $\quad X \bullet \bar{X} = 0$
- Commutative :
  $X + Y = Y + X$ $\quad X \bullet Y = Y \bullet X$
- Associative :
  $(X + Y) + Z = X + (Y + Z)$
  $(X \bullet Y) \bullet Z = X \bullet (Y \bullet Z)$
- Distributive :
  $X \bullet (Y + Z) = (X \bullet Y) + (X \bullet Z)$
  $X + (Y \bullet Z) = (X + Y) \bullet (X + Z)$

Circuit design steps

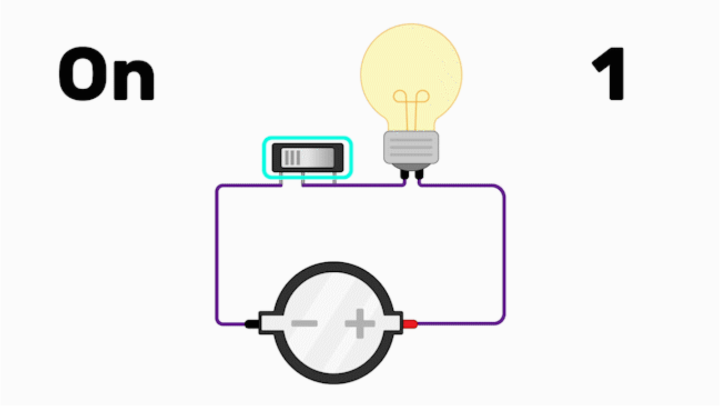
Specification

1. Write the truth table
2. Deriving the Boolean expression
3. Simplifying the expression

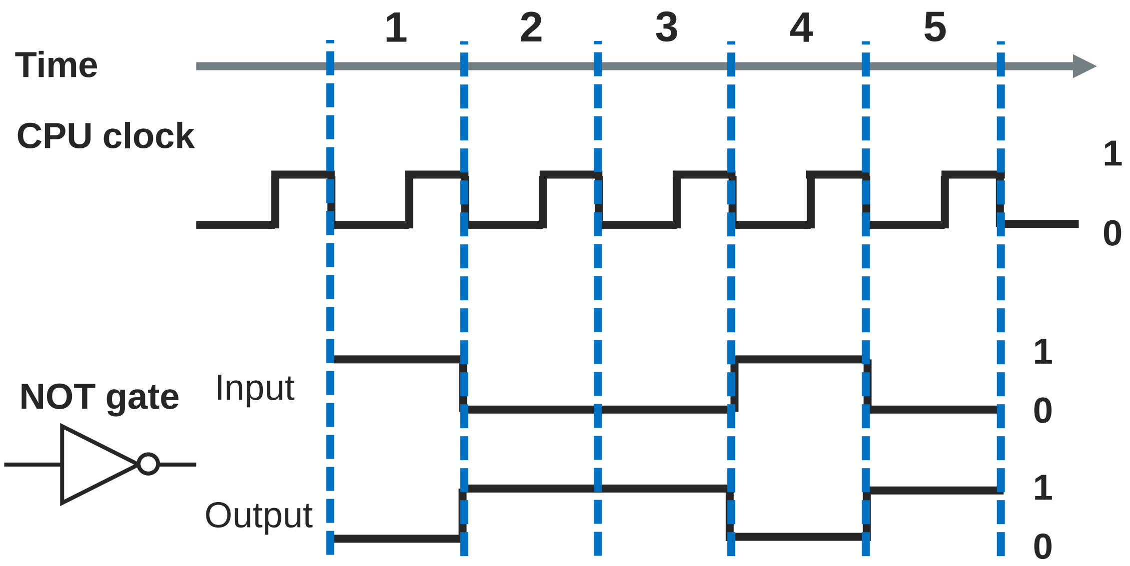
Circuit

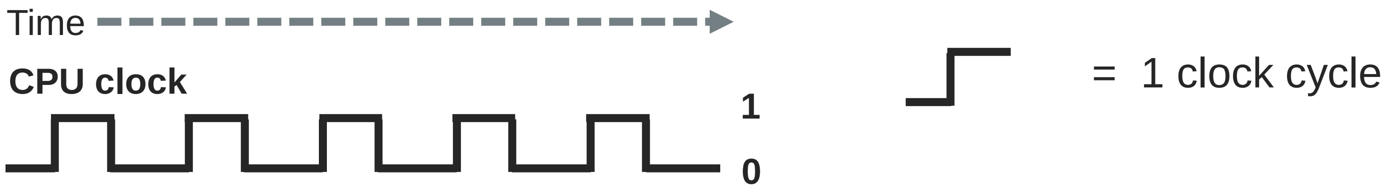
Signal delay

1. We often assume the signal output happens immediately.
2. CPU speed limitation: cannot perform more calculations per second than its capability.

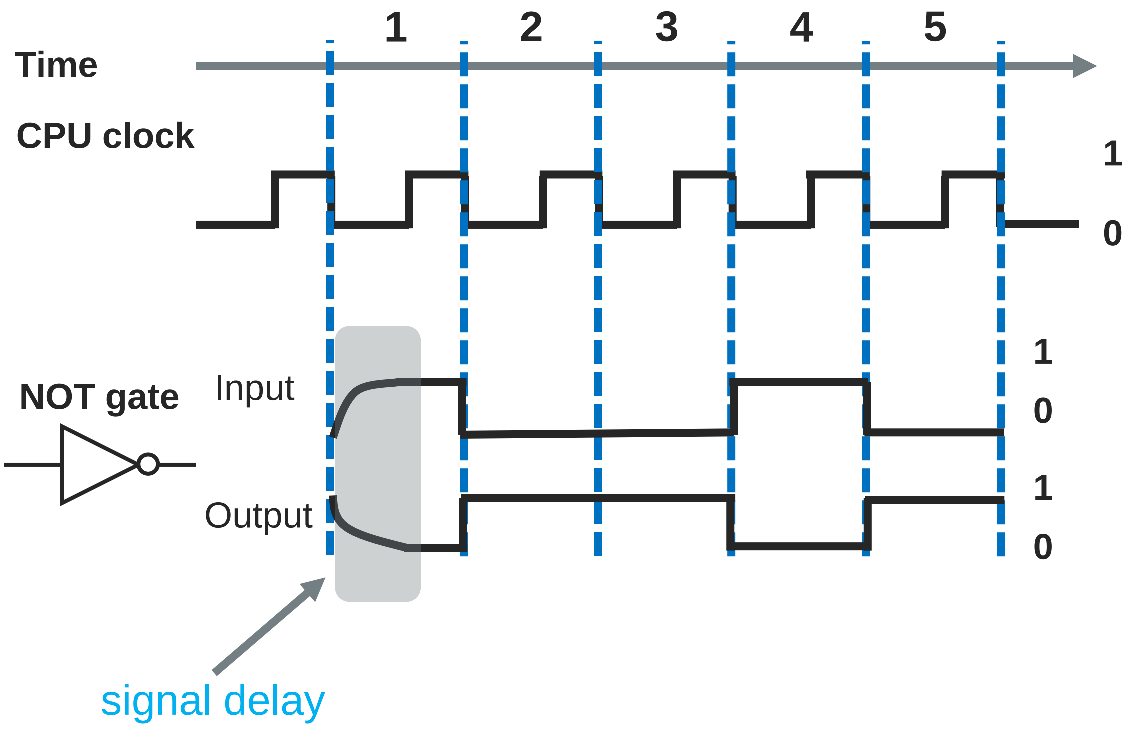


Electronic signal (in theory)





Electronic signal (in reality)



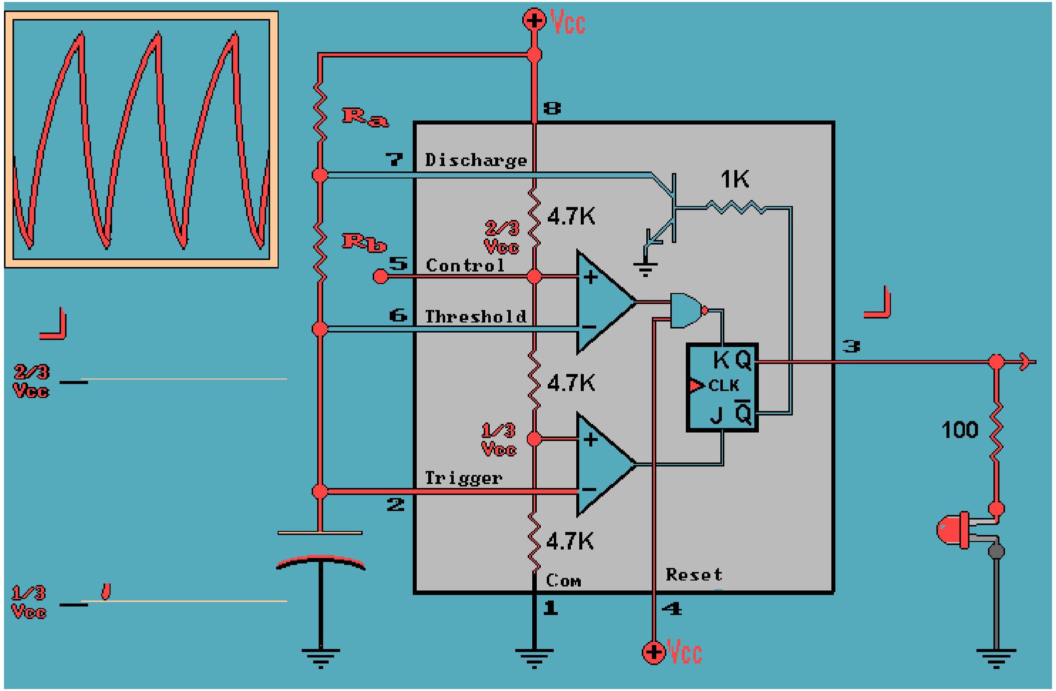
Boolean algebra

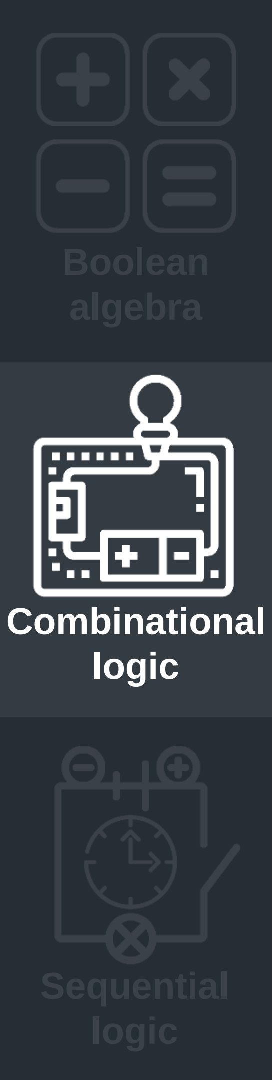
Combinational logic

Sequential logic

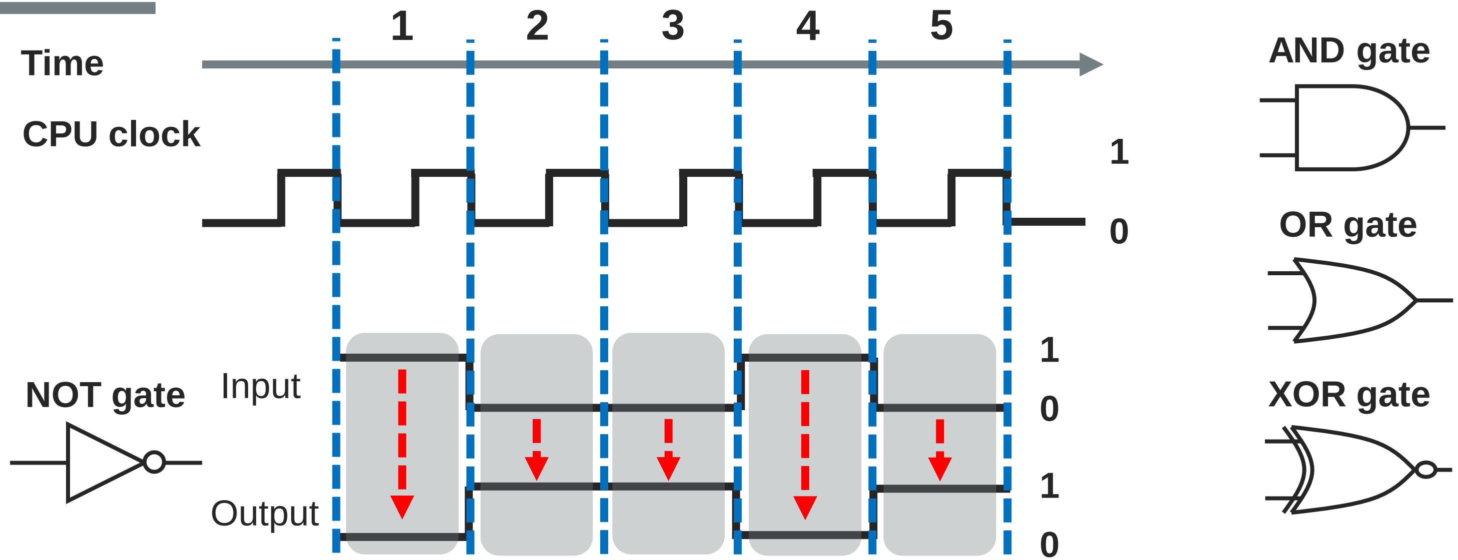
Processing delay

Complex circuit takes time to process the inputs.



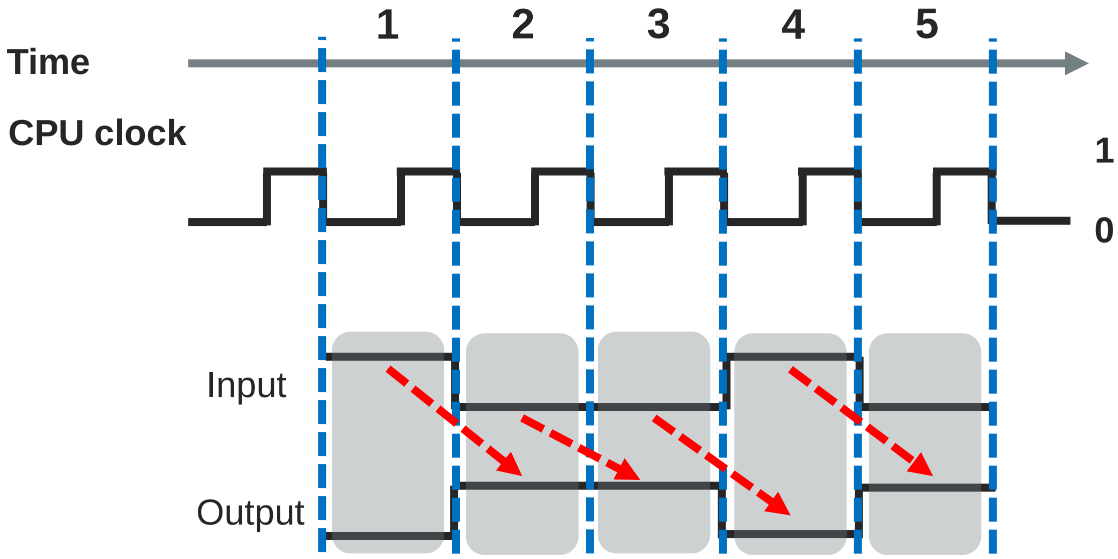


Combinational logic



1. Takes some inputs, and produces some outputs.
2. All processing happens within one time unit.
3. output[t] = function(input[t]).

Sequential logic



Output depends on the input, and the previous states
output[t] = function(input[t - 1]).

Boolean algebra

Combinational logic

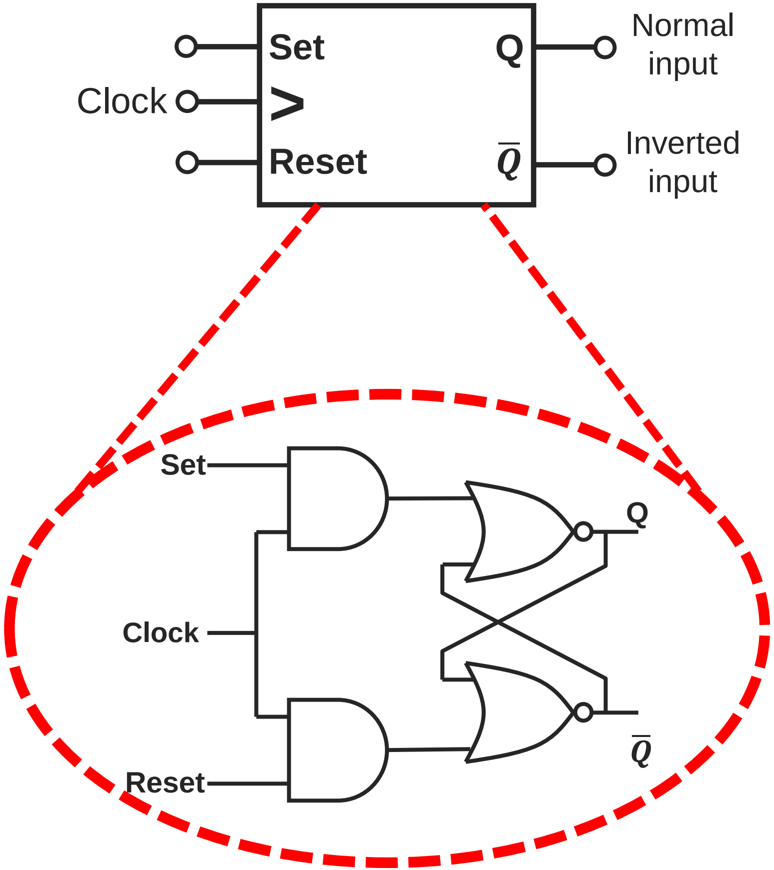
Sequential logic

S-R Flip-Flop

Set state	Reset state
Q = 1	**Q = 0**
$\overline{Q}$ = 0	**$\overline{Q}$ = 1**

1. 1-bit storage device.
2. Output can be set to store 0 or 1, depending on the inputs.
3. Flip-Flop retains its outputs, even without input signals.

S-R Flip-Flop



Truth table

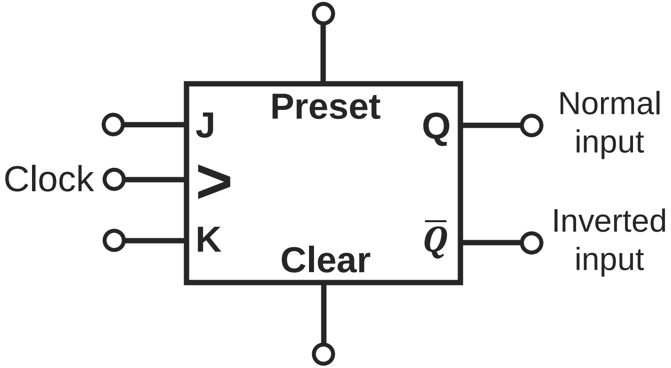
Set	Reset	Q	$\overline{Q}$
0	0	no change	
0	1	1	0
1	0	0	1
1	1	invalid	

Boolean algebra

Combinational logic

Sequential logic

J-K Flip-Flop

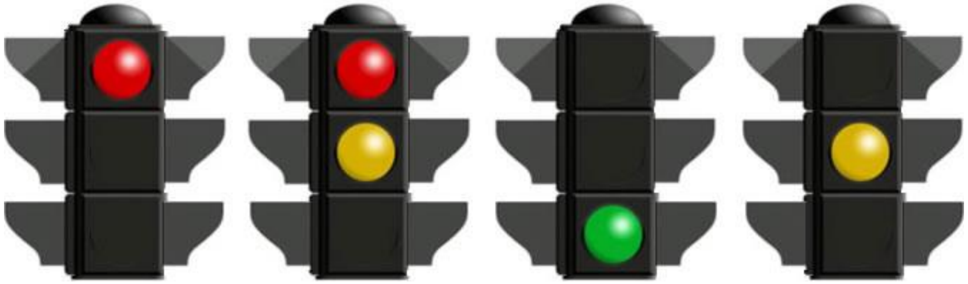


Truth table

J	K	Q	$\overline{Q}$
0	0	no change	
0	1	1	0
1	0	0	1
1	1	toggle	

1. An improvement over S-R Flip-Flop.
2. No invalid state.
3. When J = K = 1, the output toggles between 0 and 1.

Traffic light example



RED | RED AMBER | GREEN | AMBER

Questions, feedback

Cockcroft building
C519 (Khuong)
C537 (Goran)

K.A.Nguyen@brighton.ac.uk
G.Soldar@brighton.ac.uk

https://khuong.uk

(SMBC comic)