Review – Logic Circuits

Dae Hyun Kim

EECS
Washington State University

Boolean Logic/Algebra

- Deals with two values (0, 1)

- NOT (one input, one output)
  - NOT 0 = 1
  - NOT 1 = 0

- AND (two inputs, one output)
  - 0 AND 0 = 0
  - 0 AND 1 = 0
  - 1 AND 0 = 0
  - 1 AND 1 = 1

- OR (two inputs, one output)
  - 0 OR 0 = 0
  - 0 OR 1 = 1
  - 1 OR 0 = 1
  - 1 OR 1 = 1

Boolean Logic/Algebra

- Laws

$$A \cdot B = B \cdot A$$
$$A \cdot A = A$$
$$A \cdot 0 = 0$$
$$A \cdot 1 = A$$

$$A + B = B + A$$
$$A + A = A$$
$$A + 0 = A$$
$$A + 1 = 1$$

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

$$A \cdot \bar{A} = 0$$
$$A + \bar{A} = 1$$
$$\bar{\bar{A}} = A$$

Boolean Logic/Algebra

- De Morgan's Law

$$\overline{F(x_1, \dots, x_n, AND, OR)} = F(\overline{x_1}, \dots, \overline{x_n}, OR, AND)$$

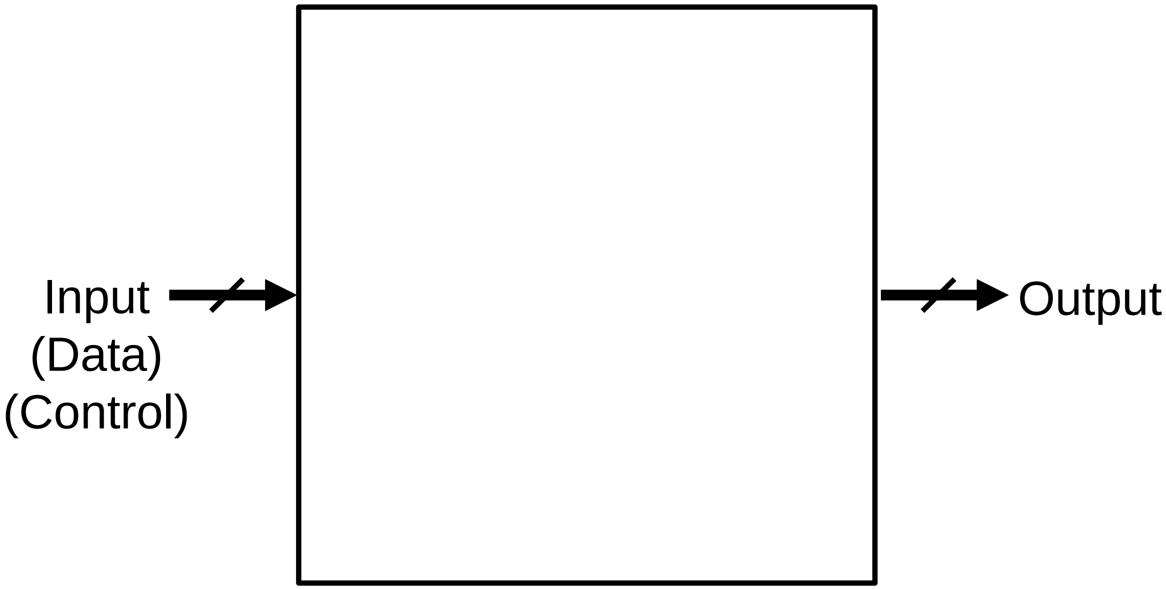
$$\overline{A \cdot B} = \bar{A} + \bar{B}$$
$$\overline{A + B} = \bar{A} \cdot \bar{B}$$
$$\overline{A \cdot (B + C + D \cdot E \cdot (F + G))} = \bar{A} + (\bar{B} \cdot \bar{C} \cdot (\bar{D} + \bar{E} + (\bar{F} \cdot \bar{G})))$$

- XOR, XNOR

$$Y = A \oplus B = A \cdot \bar{B} + \bar{A} \cdot B$$
$$Y = \overline{A \oplus B} = A \cdot B + \bar{A} \cdot \bar{B}$$

$$A \oplus B = B \oplus A$$

Systems

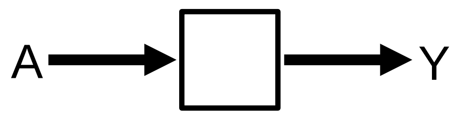


Basic Gates (Cells) and Signals

- We can categorize gates based on
  - (# inputs, # outputs)
  - Logic types (combinational, sequential)
  - ...

- Signals
  - 0, 1, Z (high-impedance), X (don't-care), U (unknown)
  - In general, 0 and 1 are always available as primary inputs (PIs).

Gates – (1, 1)



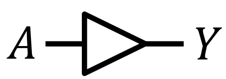
A	Y
0	0
1	0

Constant 0

$$Y = 0$$

A	Y
0	0
1	1

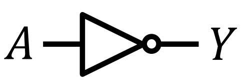
Buffer



$$Y = A$$

A	Y
0	1
1	0

Inverter



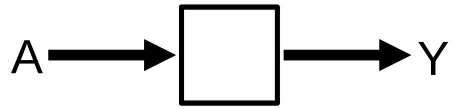
$$Y = \bar{A}$$

A	Y
0	1
1	1

Constant 1

$$Y = 1$$

How to Derive the Boolean Equations



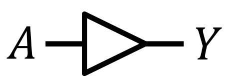
A	Y
0	0
1	0

$$Y = \bar{A} \cdot 0 + A \cdot 0 = 0$$

$$Y = 0$$

A	Y
0	0
1	1

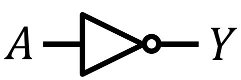
$$Y = \bar{A} \cdot 0 + A \cdot 1 = A$$



$$Y = A$$

A	Y
0	1
1	0

$$Y = \bar{A} \cdot 1 + A \cdot 0 = \bar{A}$$



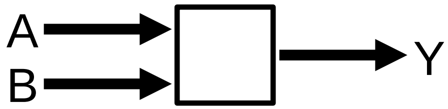
$$Y = \bar{A}$$

A	Y
0	1
1	1

$$Y = \bar{A} \cdot 1 + A \cdot 1 = 1$$

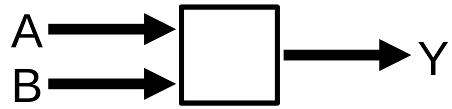
$$Y = 1$$

Gates – (2, 1)



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

Gates – (2, 1)



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

$Y = 0$

$Y = A \cdot B$ (AND)

$Y = \overline{A \rightarrow B}$ $(A \cdot \bar{B})$

$Y = A$

$Y = \overline{B \rightarrow A}$ $(B \cdot \bar{A})$

$Y = B$

$Y = A \oplus B$ (XOR)

$Y = A + B$ (OR)

$Y = \overline{A + B}$ (NOR)

$Y = \overline{A \oplus B}$ (XNOR)

$Y = \bar{B}$

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

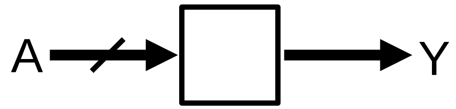
$Y = \bar{A}$

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

$Y = \overline{A \cdot B}$ (NAND)

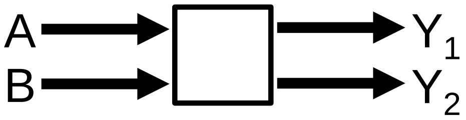
$Y = 1$

Gates – (n, 1)



- n-input AND: $Y = A_1 \cdot A_2 \cdot \cdots \cdot A_n$
- n-input NAND: $Y = \overline{A_1 \cdot A_2 \cdot \cdots \cdot A_n}$
- n-input OR: $Y = A_1 + A_2 + \cdots + A_n$
- n-input NOR: $Y = \overline{A_1 + A_2 + \cdots + A_n}$
- n-input XOR: $Y = A_1 \oplus A_2 \oplus \cdots \oplus A_n$ (1 if #1's is odd)
- n-input XNOR: $Y = \overline{A_1 \oplus A_2 \oplus \cdots \oplus A_n}$ (1 if #1's is even)

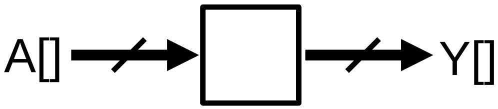
Gates – (2, 2)



- Half-adder

A	B	S	CO
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Gates – (3, 2)

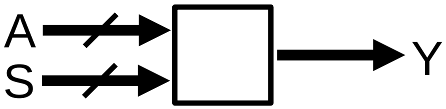


- Full-adder

A	B	CI	S	CO
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Gates – (n, 1)

- Multiplexer (MUX)



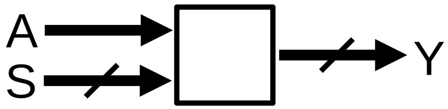
- 4:1 MUX

$$Y = \overline{S_1} \cdot \overline{S_0} \cdot A_0 + \overline{S_1} \cdot S_0 \cdot A_1 + S_1 \cdot \overline{S_0} \cdot A_2 + S_1 \cdot S_0 \cdot A_3$$

- k-bit n:1 MUX

Gates – (n, m)

- Demultiplexer (DEMUX)



- 1:4 DEMUX

S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	A
0	1	0	0	A	0
1	0	0	A	0	0
1	1	A	0	0	0

Gates – (0, 0), (0, 1), (1, 0)

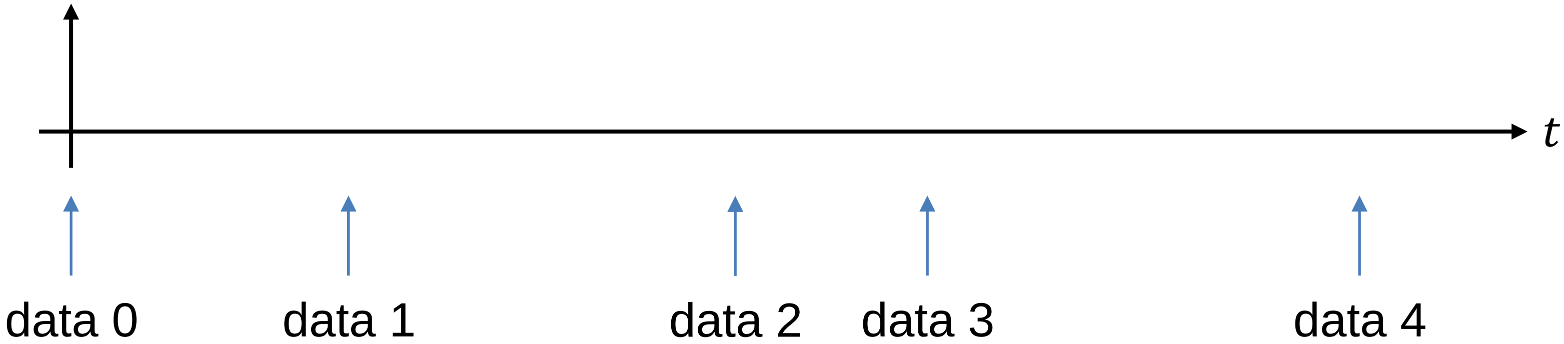
- (0, 0)
  - Fill cells (used for filling white space in a layout)
  - Decap cells
  - Tap cells
  - ...

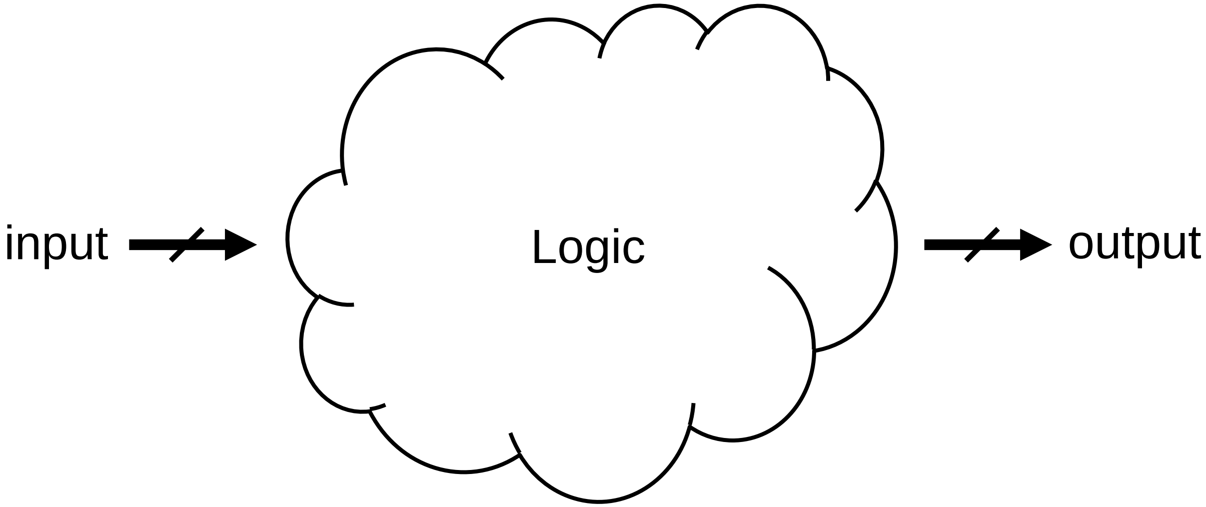
- (0, 1)
  - Constant-output cells (used to connect a signal to ground, power)
  - ...

- (1, 0)
  - Antenna cells (used to fix antenna rule violations)
  - ...

Synchronization

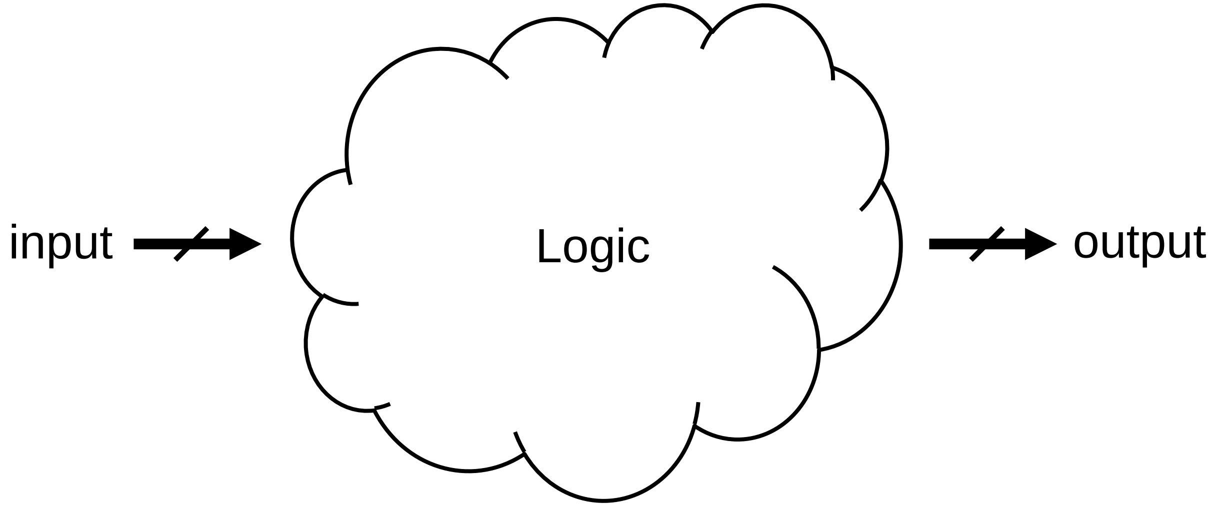
- Timing diagram





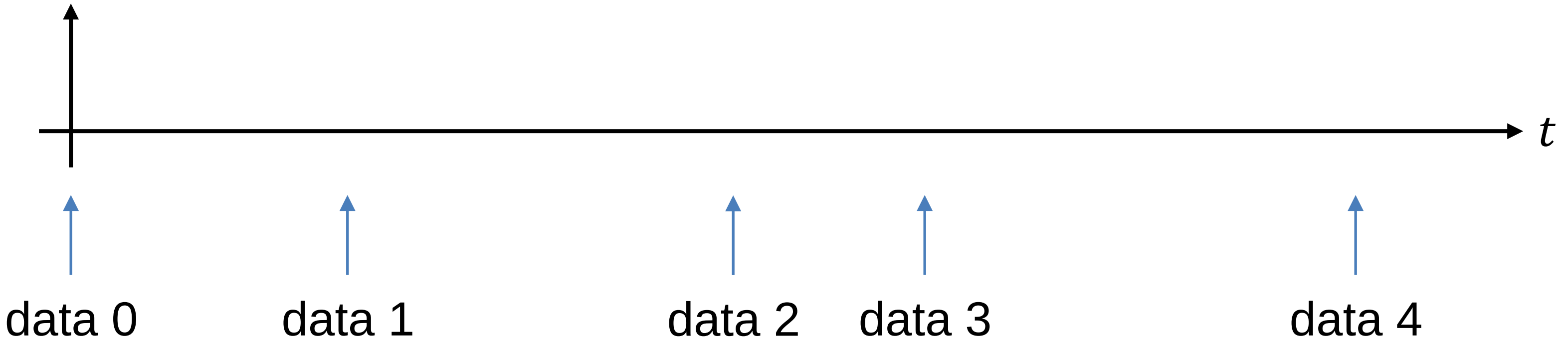
Synchronization

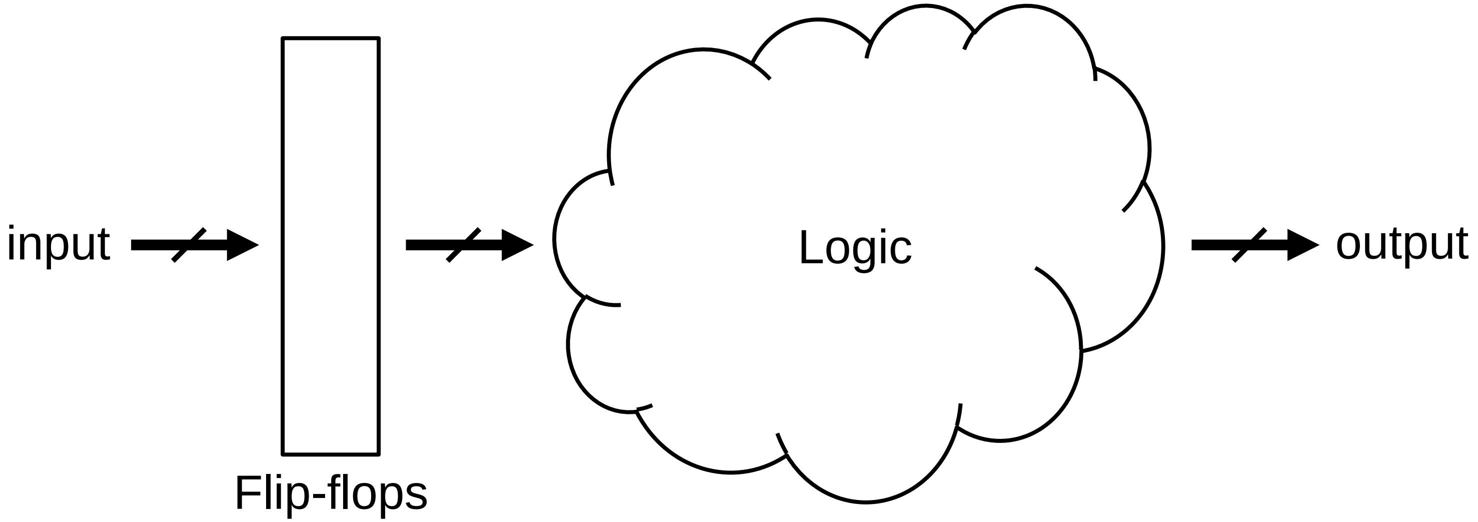
- Signal path
  - A traversal
    - from a primary input to a primary output
    - from a primary input to a flip-flop
    - from a flip-flop to a primary output
    - from a flip-flop to a flip-flop
- Different signal paths have different delays.
- Signal delays are data-dependent.



Synchronization

- Timing diagram





Synchronization

- Timing diagram