Introduction to Computer and Programming

Lecture 10

Yue Zhang
Westlake University

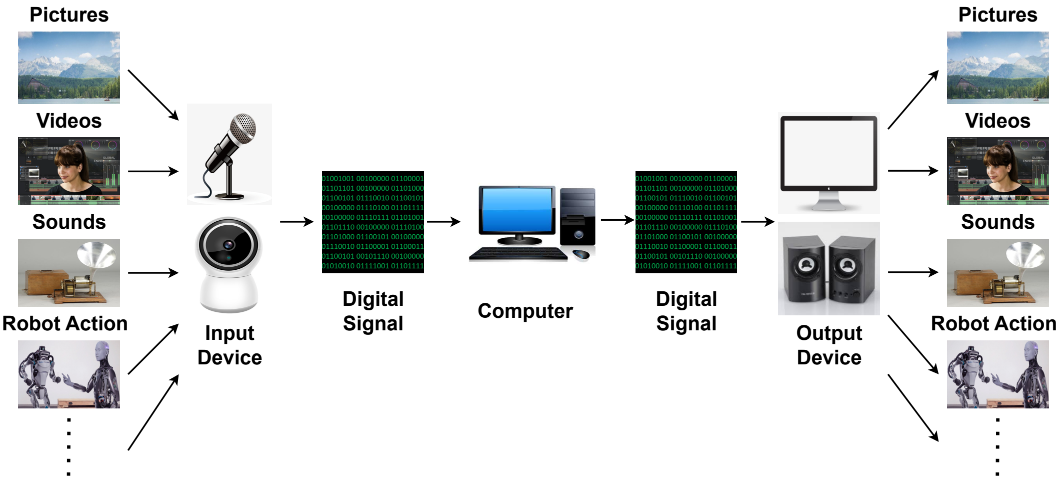
August 1, 2023

WestlakeNLP

Chapter 10.

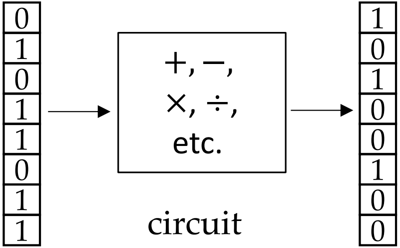
Gates and Circuits

The Digitization



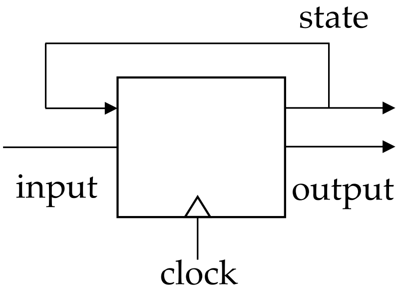
Basic Functions in Digital Devices

Combinational Circuits



$$\boxed{y = f(x)}$$

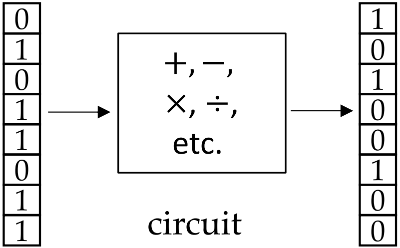
Sequential Circuits



$$\boxed{y_t = s_t = f(x_t, s_{t-1})}$$

Combinational Circuits

Combinational Circuits



$$y = f(x)$$

Define f

f is a mapping between all possible inputs and all possible outputs.

Combinational Circuits

- Use 2-bit input, 1-bit output for example.

f_1

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	0

f_2

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

f_3

x_1	x_2	y
0	0	0
0	1	0
1	0	1
1	1	0

…… ……

f_N

x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	1

Combinational Circuits

- How many 2-bit $\rightarrow$ 1-bit functions are there in total?

f_1

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	0

f_2

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

f_3

x_1	x_2	y
0	0	0
0	1	0
1	0	1
1	1	0

......

f_N

x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	1

2-bit input $\rightarrow$ 4 combinations

1-bit output $\rightarrow$ 2 cases

$\therefore 2^4 = 16$ functions.

WestlakeNLP

Combinational Circuits

- How many N-bit $\rightarrow$ 1-bit functions are there in total?

Combinational Circuits

- How many N-bit $\rightarrow$ 1-bit functions are there in total?

  N-bit input $\rightarrow 2^N$ combinations

  1-bit output $\rightarrow$ 2 cases

  $\therefore 2^{2^N}$ functions.

Combinational Circuits

- How many N-bit $\rightarrow$ 1-bit functions are there in total?

$$\begin{aligned}&\text{N-bit input} \rightarrow 2^N \text{ combinations}\\&\text{1-bit output} \rightarrow 2 \text{ cases}\\&\therefore 2^{2^N} \text{ functions.}\end{aligned}$$

- How many N-bit $\rightarrow$ M-bit functions are there in total?

WestlakeNLP

Combinational Circuits

- How many N-bit $\rightarrow$ 1-bit functions are there in total?

$$\text{N-bit input} \rightarrow 2^N \text{ combinations}$$
$$\text{1-bit output} \rightarrow 2 \text{ cases}$$
$$\therefore 2^{2^N} \text{ functions.}$$

- How many N-bit $\rightarrow$ M-bit functions are there in total?

$$\text{N-bit input} \rightarrow 2^N \text{ combinations}$$
$$\text{M-bit output} \rightarrow 2^M \text{ cases}$$
$$\therefore (2^M)^{2^N} \text{ functions.}$$

Combinational Circuits

- How can we implement arbitrary N-bit $\rightarrow$ M-bit functions?
  - Break the M bit $\rightarrow$ 1 bit independent **circuits**.

$$y = x_1 + x_2$$

x_1	x_2	y_1	y_2
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

=

x_1	x_2	y_1
0	0	0
0	1	0
1	0	0
1	1	1

+

x_1	x_2	y_2
0	0	0
0	1	1
1	0	1
1	1	0

  - Further break the circuits into the combinations of smaller units – **gates**.

WestlakeNLP

Combinational Circuits

- Combinational circuits are also called combinational logic. Gates are also called logical gates.

Combinational Circuits

- Combinational circuits are also called combinational logic.
  Gates are also called logical gates.
- Why? – Boolean Algebra.

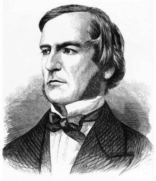


Figure: 1850s George Boole

"the mathematics of logic"

inputs are $\{0, 1\}$

representing $\{False, True\}$

- All N-bit $\rightarrow$ 1-bit functions can be viewed as boolean functions.
- including $+, -, \times, \div$ and more complicated functions.

Boolean Algebra

All 2-bit $\rightarrow$ 1-bit functions.

x_1	x_2																
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

Boolean Algebra

All 2-bit $\rightarrow$ 1-bit functions.

x_1	x_2																
0	0	$\boxed{0}$	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	$\boxed{0}$	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	$\boxed{0}$	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	$\boxed{0}$	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

$\downarrow$

0

Boolean Algebra

All 2-bit $\to$ 1-bit functions.

x_1	x_2																
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

0

AND
||
$x_1 and\ x_2$

Boolean Algebra

All 2-bit $\rightarrow$ 1-bit functions.

x_1	x_2																
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
		0	AND ‖ $x_1 and\ x_2$		x_1												

Boolean Algebra

All 2-bit → 1-bit functions.

x_1	x_2																
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
		0	AND $\Vert$ x_1 *and* x_2		x_1		x_2										

Boolean Algebra

All 2-bit $\rightarrow$ 1-bit functions.

x_1	x_2																
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
		0	AND $\parallel$ $x_1 and\ x_2$		x_1		x_2		OR $\parallel$ $x_1 or\ x_2$								

Boolean Algebra

All 2-bit $\rightarrow$ 1-bit functions.

x_1	x_2																
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
		0	AND $\parallel$ $x_1 and\ x_2$		x_1		x_2		OR $\parallel$ $x_1 or\ x_2$	NOR $\parallel$ $\neg(x_1 or\ x_2)$							

Boolean Algebra

All 2-bit $\rightarrow$ 1-bit functions.

x_1	x_2																
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
		0	AND ‖ $x_1 and\ x_2$		x_1		x_2		OR ‖ $x_1 or\ x_2$	NOR ‖ $\neg(x_1 or\ x_2)$		$\neg x_2$					

Boolean Algebra

All 2-bit $\rightarrow$ 1-bit functions.

x_1	x_2																
0	0	$\boxed{0}$	$\boxed{0}$	0	$\boxed{0}$	0	$\boxed{0}$	0	$\boxed{0}$	$\boxed{1}$	1	$\boxed{1}$	1	$\boxed{1}$	1	1	1
0	1	$\boxed{0}$	$\boxed{0}$	0	$\boxed{0}$	1	$\boxed{1}$	1	$\boxed{1}$	$\boxed{0}$	0	$\boxed{0}$	0	$\boxed{1}$	1	1	1
1	0	$\boxed{0}$	$\boxed{0}$	1	$\boxed{1}$	0	$\boxed{0}$	1	$\boxed{1}$	$\boxed{0}$	0	$\boxed{1}$	1	$\boxed{0}$	0	1	1
1	1	$\boxed{0}$	$\boxed{1}$	0	$\boxed{1}$	0	$\boxed{1}$	0	$\boxed{1}$	$\boxed{0}$	1	$\boxed{0}$	1	$\boxed{0}$	1	0	1
		$\downarrow$ 0	$\downarrow$ AND $\Vert$ $x_1 and\ x_2$		$\downarrow$ x_1		$\downarrow$ x_2		$\downarrow$ OR $\Vert$ $x_1 or\ x_2$	$\downarrow$ NOR $\Vert$ $\neg(x_1 or\ x_2)$		$\downarrow$ $\neg x_2$		$\downarrow$ $\neg x_1$			

Boolean Algebra

All 2-bit $\rightarrow$ 1-bit functions.

x_1	x_2																
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
		0	AND $\Vert$ $x_1 \, and \, x_2$		x_1		x_2		OR $\Vert$ $x_1 \, or \, x_2$	NOR $\Vert$ $\neg(x_1 \, or \, x_2)$		$\neg x_2$		$\neg x_1$		NAND $\Vert$ $\neg(x_1 \, and \, x_2)$	

Boolean Algebra

All 2-bit $\to$ 1-bit functions.

x_1	x_2																
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
		0	AND $\parallel$ $x_1 and\ x_2$		x_1		x_2		OR $\parallel$ $x_1 or\ x_2$	NOR $\parallel$ $\neg(x_1 or\ x_2)$		$\neg x_2$		$\neg x_1$		NAND $\parallel$ $\neg(x_1 and\ x_2)$	1

Boolean Algebra

AND

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

OR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

NOT

x_1	y
1	0
0	1

AND, OR and NOT are **universal**.

All N-bit $\rightarrow$ 1-bit functions can be denoted as their combinations.

WestlakeNLP

Boolean Algebra

- Using AND, OR and NOT to denote all functions.

x_1	x_2	x_3	x_4	y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

1) Read out all the 1 values in y.
2) Represent them as compositions of x.
3) Use or to join them.

$$y = \overline{x_1}x_2x_3x_4 + x_1x_2\overline{x_3}x_4 + x_1x_2x_3x_4$$

where $\overline{x_1} = \text{NOT } x_1$

$$x_1x_2 = x_1 \text{ AND } x_2$$

$$x_1 + x_2 = x_1 \text{ OR } x_2$$

WestlakeNLP

Boolean Algebra

- Using AND, OR and NOT to denote all functions.
  - You can use boolean algebra to simplify functions.

$$\begin{aligned} y &= \overline{x_1} x_2 x_3 x_4 + x_1 x_2 \overline{x_3} x_4 + x_1 x_2 x_3 x_4 \\ &= (\overline{x_1} x_3 + x_1 \overline{x_3} + x_1 x_3) x_2 x_4 \\ &= (x_1 + x_3) x_2 x_4 \end{aligned}$$

- You can learn boolean algebra by further reading relevant books and courses.
- Simplified equations simplify circuits.

Boolean Algebra

PROPERTY	AND	OR
Commutativity	$AB = BA$	$A + B = B + A$
Associativity	$(AB)C = A(BC)$	$(A + B) + C = A + (B + C)$
Distributivity	$A(B + C) = (AB) + (AC)$	$A + (BC) = (A + B)(A + C)$
Identity	$A1 = A$	$A + 0 = A$
Complement	$A(\overline{A}) = 0$	$A + \overline{A} = 1$
De Morgan's law	$\overline{AB} = \overline{A} + \overline{B}$	$\overline{A + B} = \overline{A}\,\overline{B}$

WestlakeNLP

Combinational Circuits

- How can we build AND, OR and NOT gates?

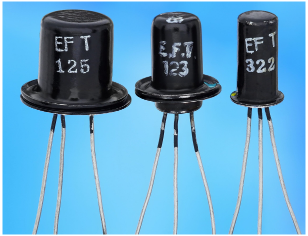


Figure: Transistors

- Transistors are patented 20 years ago before they are invented.

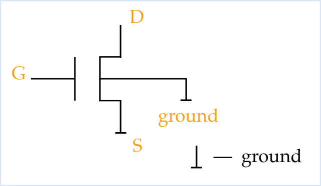
- MOSFET
Metal-oxside field-effect transistor

WestlakeNLP

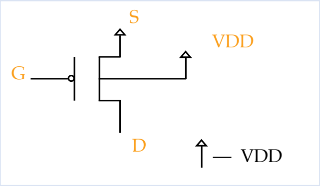
Transistors

- Transistors can be seen as gates.

PULL-DOWN GATE

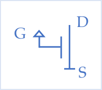


PULL-UP GATE

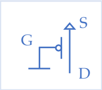


Gates are like switches.

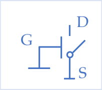
When G = 1,
D = 0.



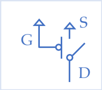
When G = 0,
D = 1.



When G = 0,
D is disconnected.

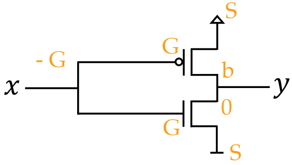


When G = 1,
D is disconnected.

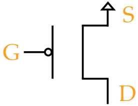


Gates

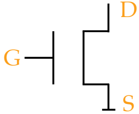
NOT gate

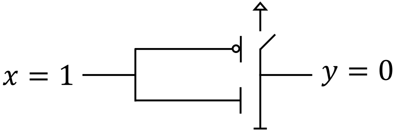


PULL UP

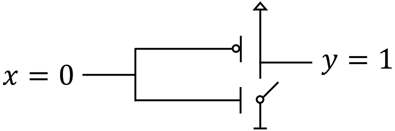


PULL DOWN





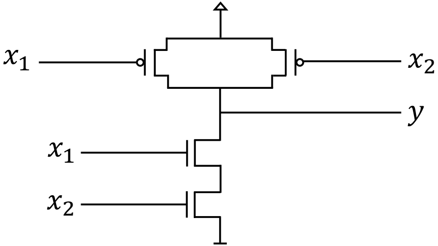
PULL-UP ×
PULL-DOWN √



PULL-UP √
PULL-DOWN ×

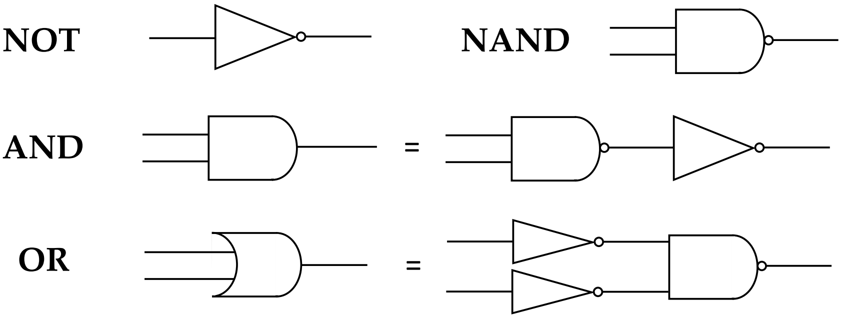
NAND gate

x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	0

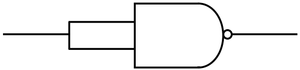


- (y=1) PULL-UP when x_1 or x_2 is 0.
- (y=0) PULL-DOWN when x_1 and x_2 is 1.

Gates



In fact, **NOT** =



Thus, **NAND** is universal !

Combinational Logic

- The final step in circuit design.

x_1	x_2	x_3	x_4	y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

$$
\begin{aligned}
y &= \overline{x_1}x_2x_3x_4 + x_1x_2\overline{x_3}x_4 + x_1x_2x_3x_4 \\
&= (\overline{x_1}x_3 + x_1\overline{x_3} + x_1x_3)x_2x_4 \\
&= (x_1 + x_3)x_2x_4
\end{aligned}
$$

WestlakeNLP

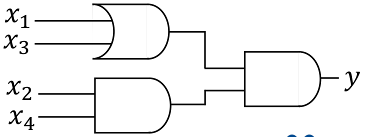
Combinational Logic

- The final step in circuit design.

x_1	x_2	x_3	x_4	y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

$$
\begin{aligned}
y &= \overline{x_1}x_2x_3x_4 + x_1x_2\overline{x_3}x_4 + x_1x_2x_3x_4 \\
&= (\overline{x_1}x_3 + x_1\overline{x_3} + x_1x_3)x_2x_4 \\
&= (x_1 + x_3)x_2x_4
\end{aligned}
$$

$\Downarrow$



WestlakeNLP

Combinational Logic

- Boolean logic can also represent algebraic functions, of course.
- Adder (1-bit)

A	B	Sum	Carry
0	0		
0	1		
1	0		
1	1		

Combinational Logic

- Boolean logic can also represent algebraic functions, of course.
- Adder (1-bit)

A	B	Sum	Carry
0	0	0	0
0	1		
1	0		
1	1		

WestlakeNLP

- Boolean logic can also represent algebraic functions, of course.
- Adder (1-bit)

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0		
1	1		

WestlakeNLP

- Boolean logic can also represent algebraic functions, of course.
- Adder (1-bit)

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1		

WestlakeNLP

Combinational Logic

- Boolean logic can also represent algebraic functions, of course.
- Adder (1-bit)

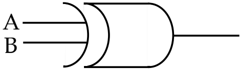
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Combinational Logic

- Consider sum and carry separately.

A	B	Sum
0	0	0
0	1	1
1	0	1
1	1	0

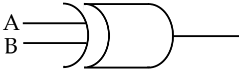
A **XOR** B



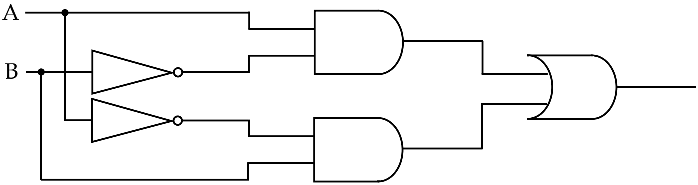
- Consider sum and carry separately.

A	B	Sum
0	0	0
0	1	1
1	0	1
1	1	0

A **XOR** B



(A **AND NOT** B) **OR** (B **AND NOT** A)

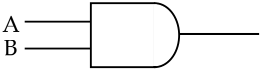


Combinational Logic

- Consider sum and carry separately.

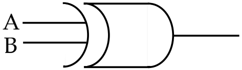
A	B	Carry
0	0	0
0	1	0
1	0	0
1	1	1

A **AND** B

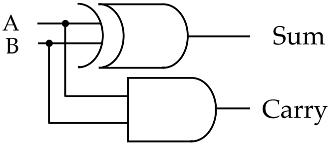


A	B	Sum
0	0	0
0	1	1
1	0	1
1	1	0

A **XOR** B



Together



Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

WestlakeNLP

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
0	0	0	0	0
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

WestlakeNLP

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
0	0	0	0	0
0	0	1	1	0
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

WestlakeNLP

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

WestlakeNLP

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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		
1	1	0		
1	1	1		

WestlakeNLP

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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		
1	1	0		
1	1	1		

WestlakeNLP

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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		
1	1	1		

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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		

WestlakeNLP

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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

$$\text{sum} = \overline{AB}I + \overline{A}B\overline{I} + A\overline{BI} + ABI$$

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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

$$
\begin{aligned}
\text{sum} &= \overline{A}\,\overline{B}I + \overline{A}B\overline{I} + A\overline{B}\,\overline{I} + ABI \\
&= (\overline{A}B + A\overline{B})\overline{I} + (\overline{A}\,\overline{B} + AB)I
\end{aligned}
$$

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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

$$
\begin{aligned}
\text{sum} &= \bar{A}\bar{B}I + \bar{A}B\bar{I} + A\bar{B}\bar{I} + ABI \\
&= (\bar{A}B + A\bar{B})\bar{I} + (\bar{A}\bar{B} + AB)I \\
&= (A \text{ XOR } B)\bar{I} + \overline{(A \text{ XOR } B)}I
\end{aligned}
$$

Combinational Logic

- Also considering carry-in.

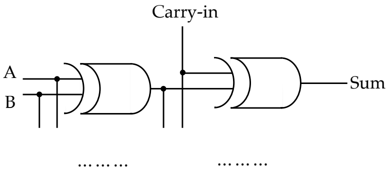
A	B	Carry-in	Sum	Carry-out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$\begin{aligned}
\text{sum} &= \overline{A}\,\overline{B}I + \overline{A}B\overline{I} + A\overline{B}\,\overline{I} + ABI \\
&= (\overline{A}B + A\overline{B})\overline{I} + (\overline{A}\,\overline{B} + AB)I \\
&= (A \text{ XOR } B)\overline{I} + \overline{(A \text{ XOR } B)}I \\
&= (A \text{ XOR } B) \text{ XOR } I
\end{aligned}$$

Combinational Logic

- Also considering carry-in.

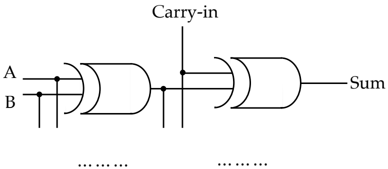
A	B	Carry-in	Sum	Carry-out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



$$
\begin{aligned}
\text{sum} &= \bar{A}\bar{B}I + \bar{A}B\bar{I} + A\bar{B}\bar{I} + ABI \\
&= (\bar{A}B + A\bar{B})\bar{I} + (\bar{A}\bar{B} + AB)I \\
&= (A \text{ XOR } B)\bar{I} + \overline{(A \text{ XOR } B)}I \\
&= (A \text{ XOR } B) \text{ XOR } I
\end{aligned}
$$

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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



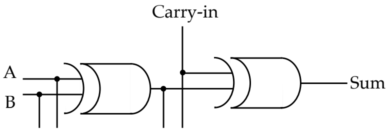
$$\text{Carry-out} = \overline{A}BI + A\overline{B}I + AB\overline{I} + ABI$$

$$\begin{aligned}\text{sum} &= \overline{A}\,\overline{B}I + \overline{A}B\overline{I} + A\overline{B}\,\overline{I} + ABI \\ &= (\overline{A}B + A\overline{B})\overline{I} + (\overline{A}\,\overline{B} + AB)I \\ &= (A \text{ XOR } B)\overline{I} + \overline{(A \text{ XOR } B)}I \\ &= (A \text{ XOR } B) \text{ XOR } I\end{aligned}$$

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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



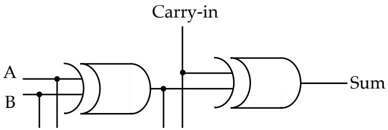
$$
\begin{aligned}
\text{Carry-out} &= \overline{A}BI + A\overline{B}I + AB\overline{I} + ABI \\
&= (\overline{A}B + A\overline{B})I + AB(\overline{I} + I)
\end{aligned}
$$

$$
\begin{aligned}
\text{sum} &= \overline{A}\overline{B}I + \overline{A}B\overline{I} + A\overline{B}\overline{I} + ABI \\
&= (\overline{A}B + A\overline{B})\overline{I} + (\overline{A}\overline{B} + AB)I \\
&= (A \text{ XOR } B)\overline{I} + \overline{(A \text{ XOR } B)}I \\
&= (A \text{ XOR } B) \text{ XOR } I
\end{aligned}
$$

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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



……… ………

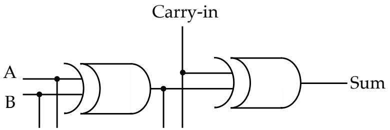
$$\begin{aligned}\text{Carry-out} &= \bar{A}BI + A\bar{B}I + AB\bar{I} + ABI \\ &= (\bar{A}B + A\bar{B})I + AB(\bar{I} + I) \\ &= (A \text{ XOR } B)I + AB(\bar{I} + I)\end{aligned}$$

$$\begin{aligned}\text{sum} &= \bar{A}\bar{B}I + \bar{A}B\bar{I} + A\bar{B}\bar{I} + ABI \\ &= (\bar{A}B + A\bar{B})\bar{I} + (\bar{A}\bar{B} + AB)I \\ &= (A \text{ XOR } B)\bar{I} + \overline{(A \text{ XOR } B)}I \\ &= (A \text{ XOR } B) \text{ XOR } I\end{aligned}$$

Combinational Logic

- Also considering carry-in.

A	B	Carry-in	Sum	Carry-out
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



……… ………

$$\begin{aligned} \text{sum} &= \overline{A}\,\overline{B}I + \overline{A}B\overline{I} + A\overline{B}\,\overline{I} + ABI \\ &= (\overline{A}B + A\overline{B})\overline{I} + (\overline{AB} + AB)I \\ &= (A \text{ XOR } B)\overline{I} + \overline{(A \text{ XOR } B)}I \\ &= (A \text{ XOR } B) \text{ XOR } I \end{aligned}$$

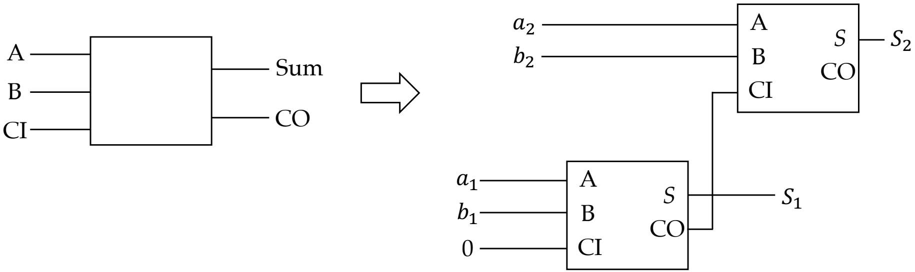
$$\begin{aligned} \text{Carry-out} &= \overline{A}BI + A\overline{B}I + AB\overline{I} + ABI \\ &= (\overline{A}B + A\overline{B})I + AB(\overline{I} + I) \\ &= (A \text{ XOR } B)I + AB(\overline{I} + I) \\ &= (A \text{ XOR } B)I + AB \end{aligned}$$

Combinational Circuits

- Putting adders together.

1-bit Adder

2-bit Adders



WestlakeNLP

Intergrated Circuits

- Chips – A piece of silicon on which multiple gates are made.

Abbreviation	Name	Number of Gates
SSI	Small-scale integration	1 to 10
MSI	Medium-scale integration	10 to 100
LSI	Large-scale integration	100 to 100,000
VLSI	Very-large-scale integration	more than 100,000