Digital systems and basics of electronics

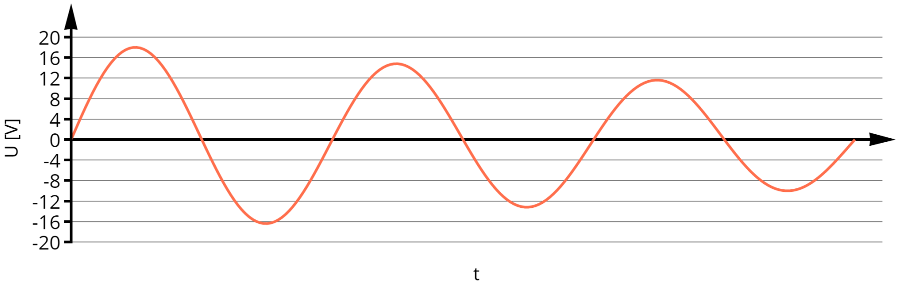
Adam Szmigielski

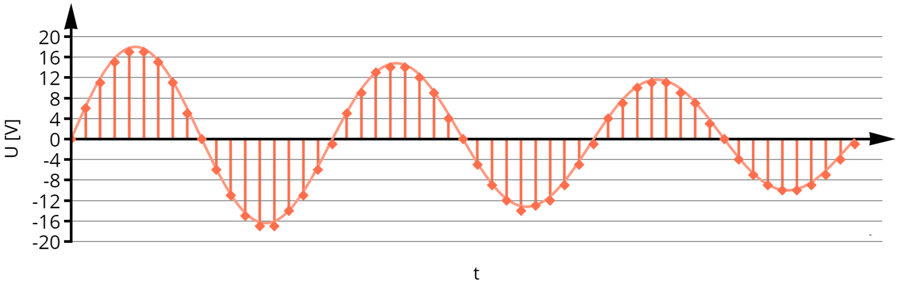
aszmigie@pjwstk.edu.pl

materials: $ftp(public)://aszmigie/SYC/ENG$

Introduction to digital systems - lecture 6

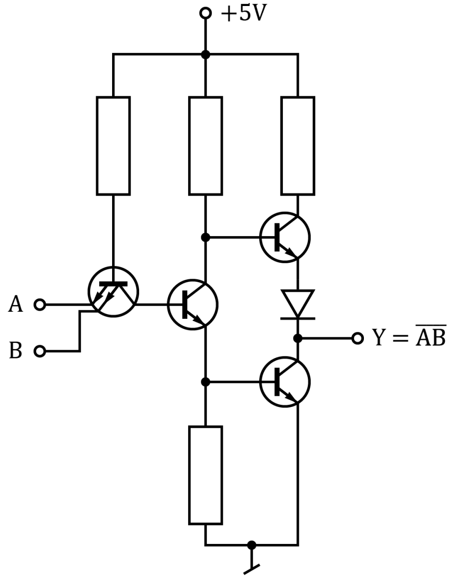
Analog and digital signals





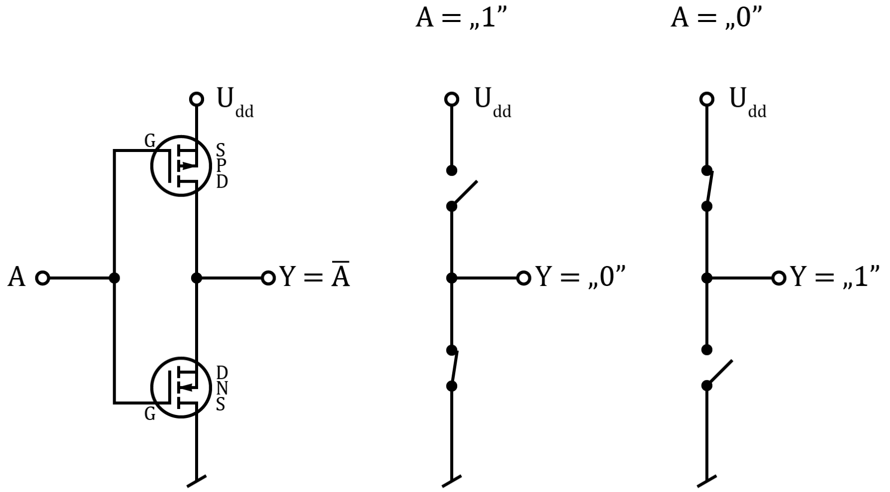
- *Analog signal:* - is any continuous signal for all the time,
- *Discrete-time:* - signal is a sampled version of an analog signal: the value of signal is noted at fixed intervals, rather than continuously.
- *Logic voltage levels:* - the two states of a wire are usually represented by some measurement of an electrical property.

Transistor–Transistor Logic (TTL) standard



- Build on bipolar transistor, supplied with DC voltage about 5V.
- When electrical potential, related to ground, is in interval $0V \div 0,8V$ we have **low state** or **logical "0"**,
- When electrical potential is in interval $2V \div 5V$ we have **high state** or **logical "1"**,
- Electrical potential is in interval $0,8V \div 2V$ - signal is not digital.

Complementary Metal–Oxide–Semiconductor (CMOS) standard



- CMOS uses complementary and symmetrical pairs of p-type and n-type metal oxide semiconductor field effect transistors (MOSFETs) for logic functions. Only one of them is on.
- Relative cheap and easy in production, with high density,
- Can be supplied with voltages $3 \div 18V$,

- Practically in statical state CMOS has no power consumption, energy is lost in switching,
- Logical voltage levels are closed to power supply poles (ground - logical "0", positive pole - logical "1"). Sometimes logical voltage levels are distinguished by percentages - "0- related to $0 - 30\%$, "1' related to $70 - 100\%$ of power supply.

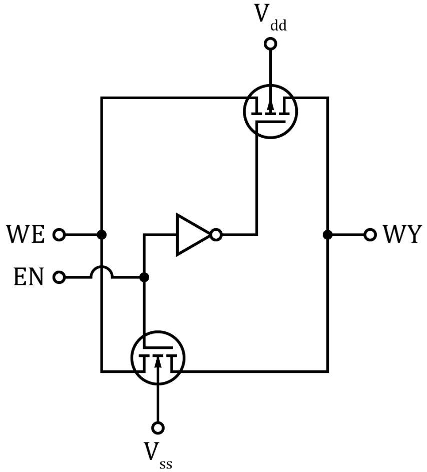
(Low Voltage) CMOS

- There is the tendency to reducie the value of power supply,
- CMOS series supplied with voltage $3,3V$, $2,5V$ or even $1,8V$ are produced,

Third logical state and ópen collector"gates

- Besides logical "0ąnd "1"there is the *third state* called **high impedance state**,
- If any point of circuit in not connected (open circuit) neither to grand nor positive pole of power supply this point is in **high impedance state**,
- To treat high impedance point as logical zero or one it is necessary to connect that point via resistor respectively to ground or positive pole. Resistor linked high impedance point to source is called *pull-up resistor*.
- Special logic gates, which output can be in high impedance state, are produced.

Transmission gate



- Transmission gate is an electronic element. It is a good non-mechanical relay, built with CMOS technology. Sometimes known as an analog gate, analogue switch or electronic relay depending on its use.
- It is made by the parallel combination of an nMOS and a pMOS transistor with the input at the gate of one transistor being complementary to the input at the gate of the other.
- This logic can be used to design multiplexers (and demultiplexers) - see manual 4051.

Boolean algebra

- Boolean algebra (or Boolean logic) is a logical calculus of truth values, developed by George Boole in the late 1830s.
- With the numeric operations of multiplication $x \cdot y$, addition $x + y$, and negation $-x$ replaced by the respective logical operations of conjunction $x \wedge y$, disjunction $x \vee y$, and complement $\neg x$.
- There are several notation used in Boolean logic (algebra)
  - conjunction $\wedge$, alternative $\vee$ and negation $\neg$
  - conjunction $\cap$, alternative $\cup$ and negation $\sim$
  - conjunction $\cdot$, alternative $+$ and negation $-$

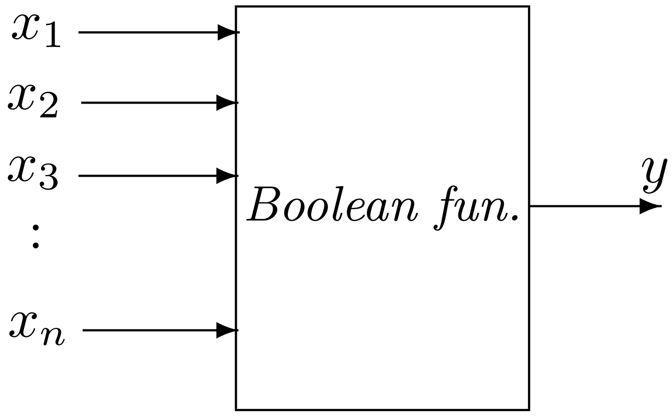
Properties of Boolean algebra

associativity	$(ab)c = a(bc)$	$(a+b)+c = a+(b+c)$
commutativity	$ab = ba$	$a+b = b+a$
distributivity	$a+(bc) = (a+b)\cdot(a+c)$	$a\cdot(b+c) = (ab)+(ac)$
absorption	$a(a+b) = a$	$a+(ab) = a$
complements	$a+\overline{a} = 1$	$a\cdot\overline{a} = 0$

Boolean algebra laws

- de Morgana's law:
  $\overline{a+b} = \overline{a} \cdot \overline{b}$
  $\overline{a \cdot b} = \overline{a} + \overline{b}$
- Rule of grouping:
  $a\overline{b} + ab = a$

Boolean function



- *Boolean n argument function* is an transformation $f : B^n \to B$, where $B = \{0, 1\}$ is an set of function values,
- *Boolean function* is an model of *combinational circuit.*

Description of Boolean functions - truth tables

- One variable function (eg. negation $f(a) = \neg a$)

a	f(a)
0	1
1	0

- Two variable function (eg. conjunction $f(a, b) = a \wedge b$)

a	b	$a \wedge b$
0	0	0
0	1	0
1	0	0
1	1	1

Zeros and ones - binary and decimal representation

a	b	$a \wedge b$
0	0	0
0	1	0
1	0	0
1	1	1

$f^1 = [11]$ - *set of ones* and its binary representation,

$f^0 = \begin{bmatrix} 00 \\ 01 \\ 10 \end{bmatrix}$ - *set of zeros* and its binary representation,

$f^1 = \{3\}$ - *set of ones* and its decimal representation,

$f^0 = \{0, 1, 2\}$ - *set of zeros* and its decimal representation.

Disjunctive Normal Form (DNF)

a	b	$f(a,b)$
0	0	0
0	1	0
1	0	0
1	1	1

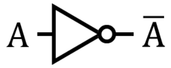
- Disjunctive Normal Form (DNF) is a disjunction, where a clause is a conjunctive of literals. : function f is a sum of products $f = \ldots(\ldots \wedge \ldots \wedge \ldots) \vee (\ldots \wedge \ldots \wedge \ldots) \vee (\ldots \wedge \ldots \wedge \ldots)\ldots,$
- Expression in brocket (product) corresponds to one one ("1")
- In our case: $f = (a \wedge b)$,
- Decimal description: $f(a,b) = \sum\{3\}$.

Conjunctive Normal Form (CNF)

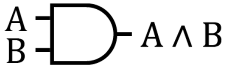
a	b	$f(a,b)$
0	0	0
0	1	0
1	0	0
1	1	1

- Conjunctive normal form (CNF) is a conjunction of clauses, where a clause is a disjunction of literals.: function is a product of sums $f = \ldots(\ldots \vee \ldots \vee \ldots) \wedge (\ldots \vee \ldots \vee \ldots) \wedge (\ldots \vee \ldots \vee \ldots) \ldots,$
- Expression in brocket (sum) corresponds to one one ("0"),
- In our case: $f = (a \vee b) \wedge (a \vee \overline{b}) \wedge (\overline{a} \vee b)$.
- Decimal description: $f(a,b) = \prod\{0,1,2\}$

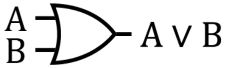
The most popular logic gates



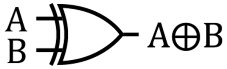
A	NOT A
0	1
1	0



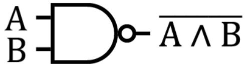
A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1



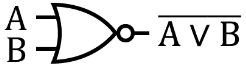
A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1



A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0



A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0



A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0

Task for lab

1. Introduction to *Logisim*,
2. Using only NAND gates realize NOT, AND i OR gates (each function as a separated circuit),
3. Using available in *Logisim* gates realize function:

   $$y = a \cdot \bar{b} \cdot c + a \oplus b + c$$

   Realization should reflect structure of equation.
4. Using “yours” AND OR and NOT realize above function. Realization should reflect structure of equation.