Lecture 1:
Introduction to Digital System Design

Emad Samuel Malki Ebeid

Full Professor at the University of Southern Denmark

Head of SDU Digital and High-Frequency Electronics section

CPU, GPU, FPGA: what is the difference? Hardware vs Software programming?

Figure 2: **Industrial Analogy for CPUs**

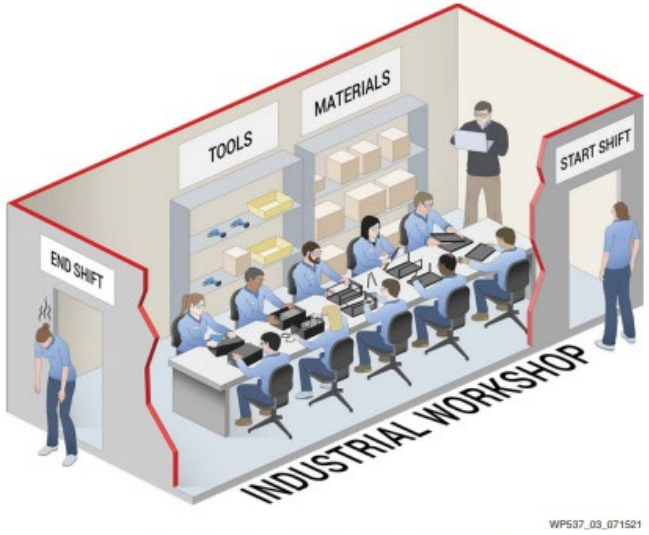


Figure 3: **Industrial Analogy for GPUs**

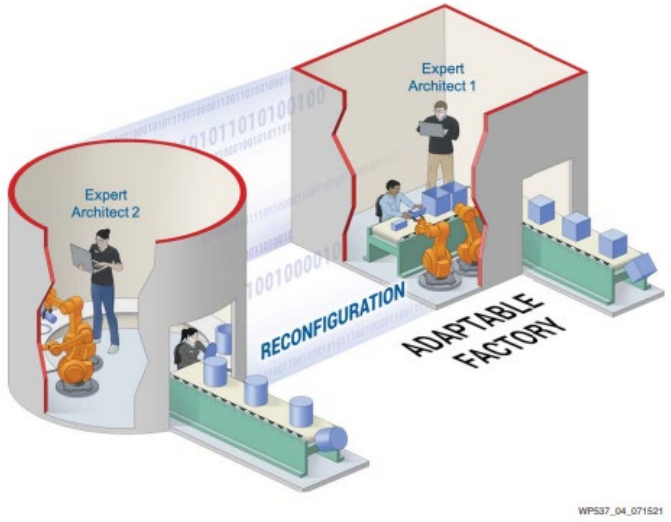
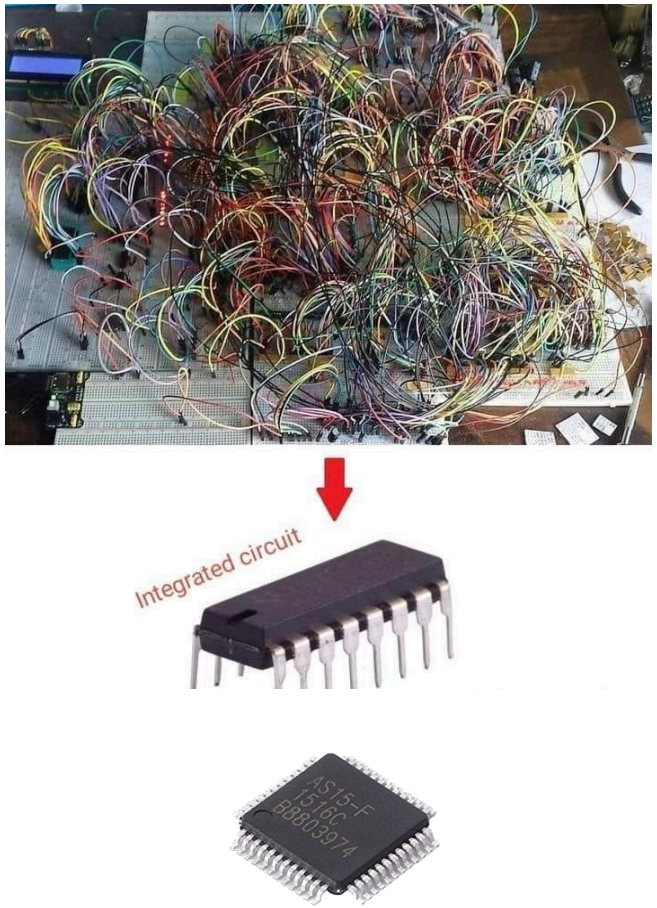


Figure 4: **Industrial Analogy for FPGAs**

Adaptive Computing in Robotics Leveraging ROS 2 to Enable Software-Defined Hardware for FPGAs, 20212
https://www.xilinx.com/content/dam/xilinx/support/documentation/white_papers/wp537-adaptive-computing-robotics.pdf

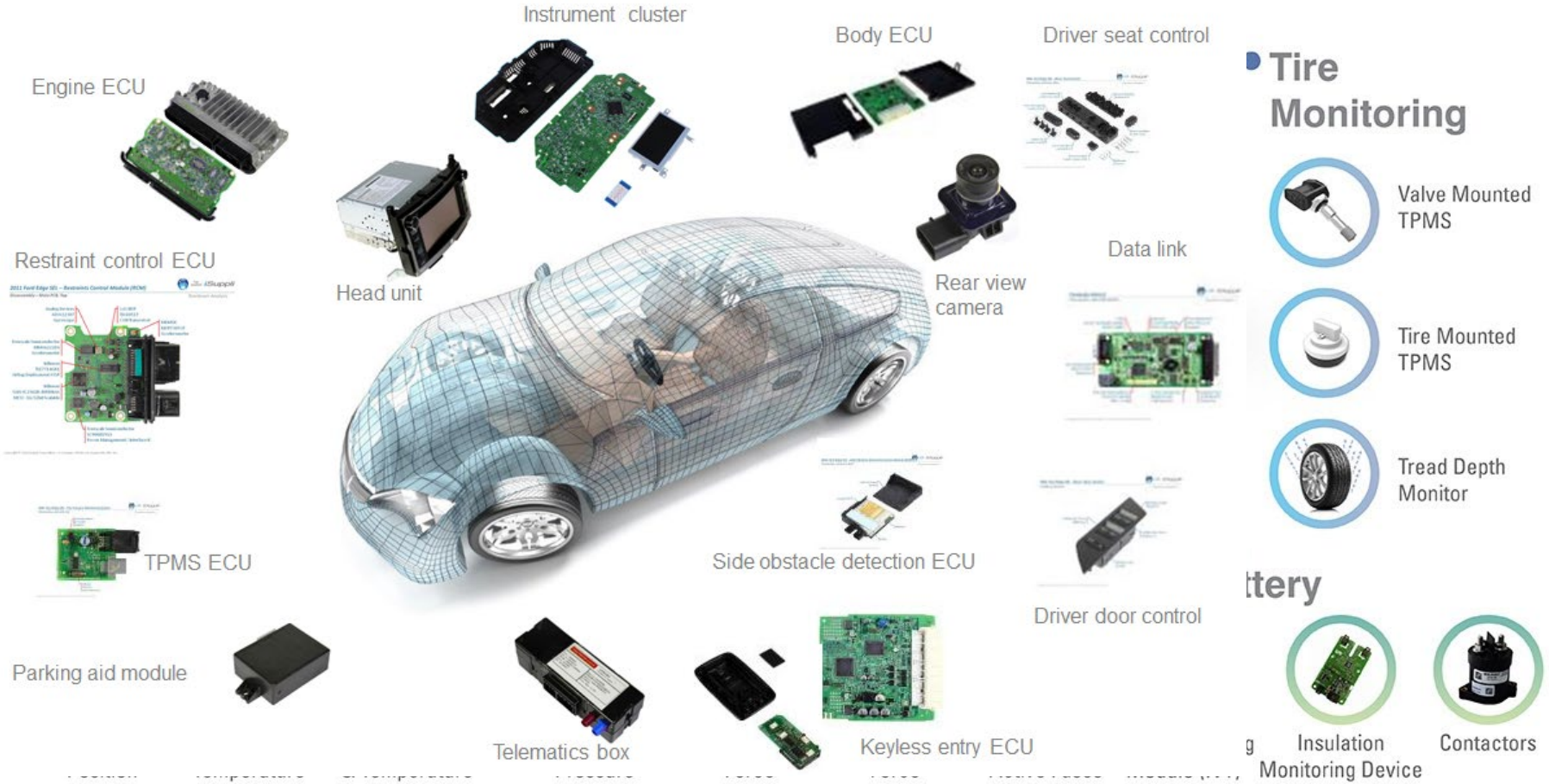
What is an Integrated Circuit/Embedded System/programmable electronics?

- Integrated Circuit (IC) is a micro chip that can function as an amplifier, oscillator, timer, microprocessor, or computer memory.
- IC can hold anywhere from hundreds to millions of transistors, resistors, and capacitors.
- It can perform calculations and store data using either digital or analog technology
- Embedded system is a combination of computer hardware and software designed for functions within a larger system or for a specific function.



Applications

- Automotive

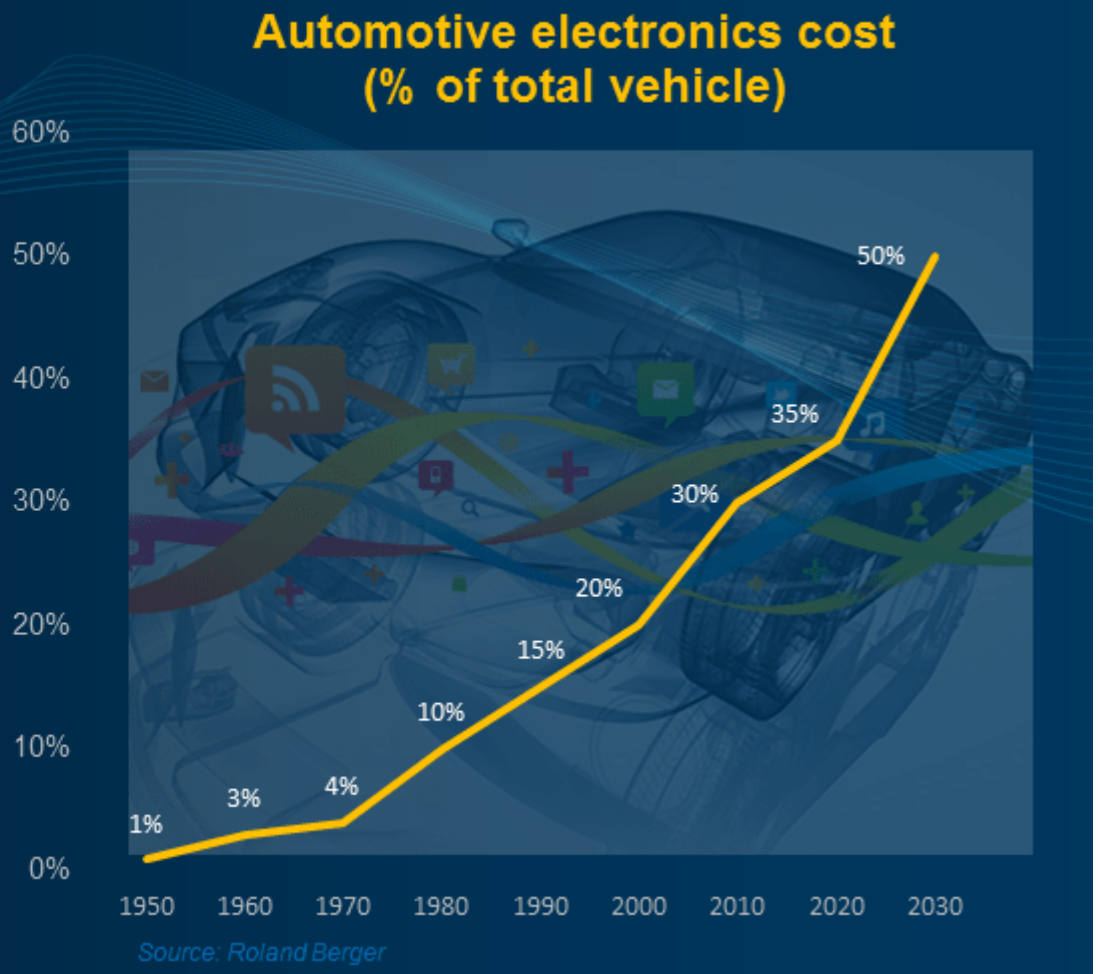


https://www.evengineeringonline.com/

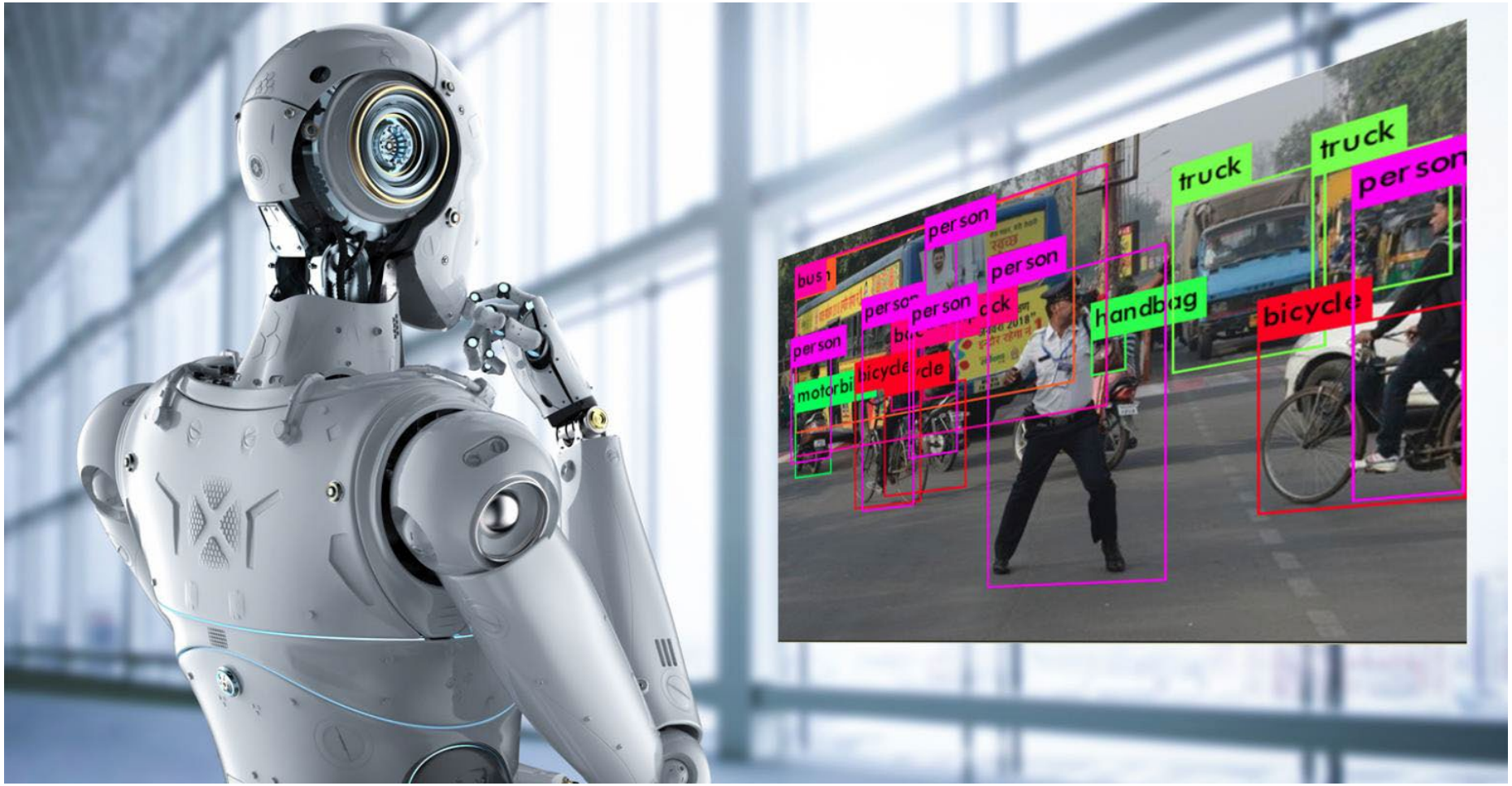
Automotive Market

*"High-end cars will contain more than **$6,000** worth of electronics in five years, driving a **$160 billion** automotive electronics market in 2022"*

Luca De Ambroggi
Principal analyst Automotive electronics
IHS Markit

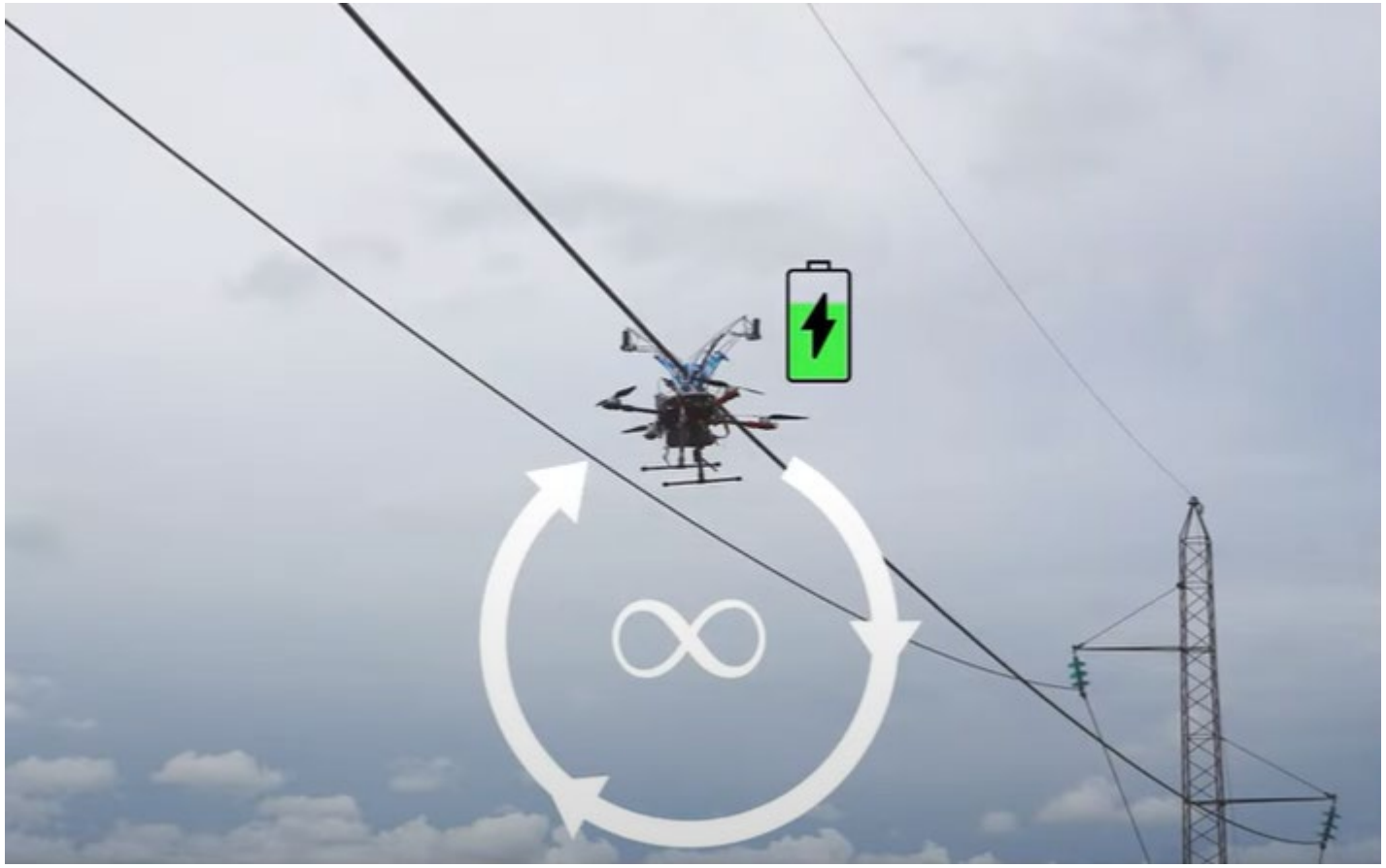


Self-learing robots



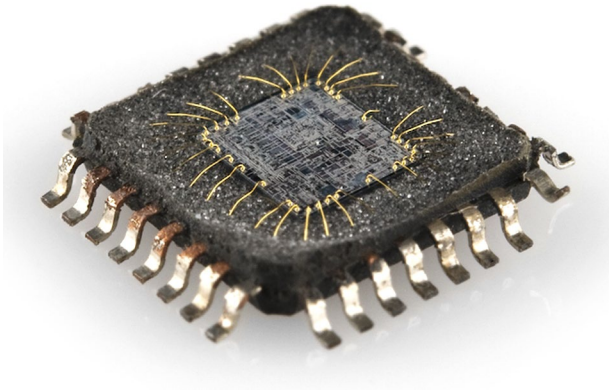
electronicdesign.com

Autonomous drones



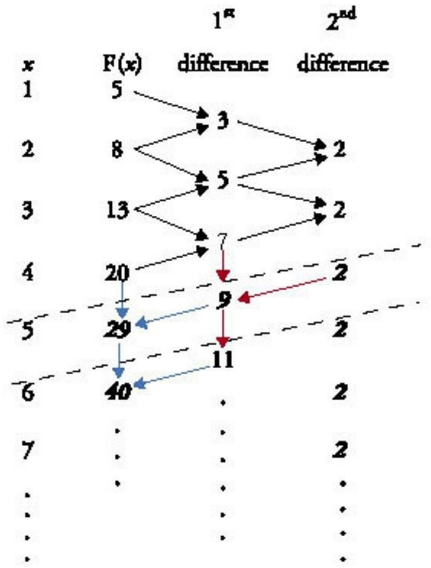
Digital Integrated Circuits

- Why is designing digital integrated circuits different today than it was before?
- Will it change in the future?



The first computer

- Charles Babbage: **Difference Engine** (1834), a large-scale mechanical computing device.
- It allows the values of a ***polynomial function*** (e.g., $x^2 + 4$) to be calculated using simple **addition** only.
- 25000 parts
- Cost £17,470 in 1834!



$F(x) = x^2 + 4$

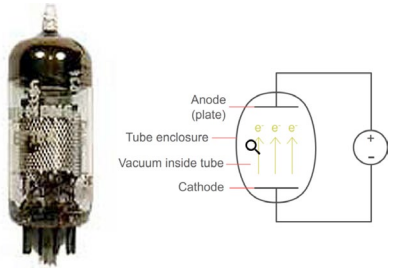
https://www.computerhistory.org/babbage/howitworks/

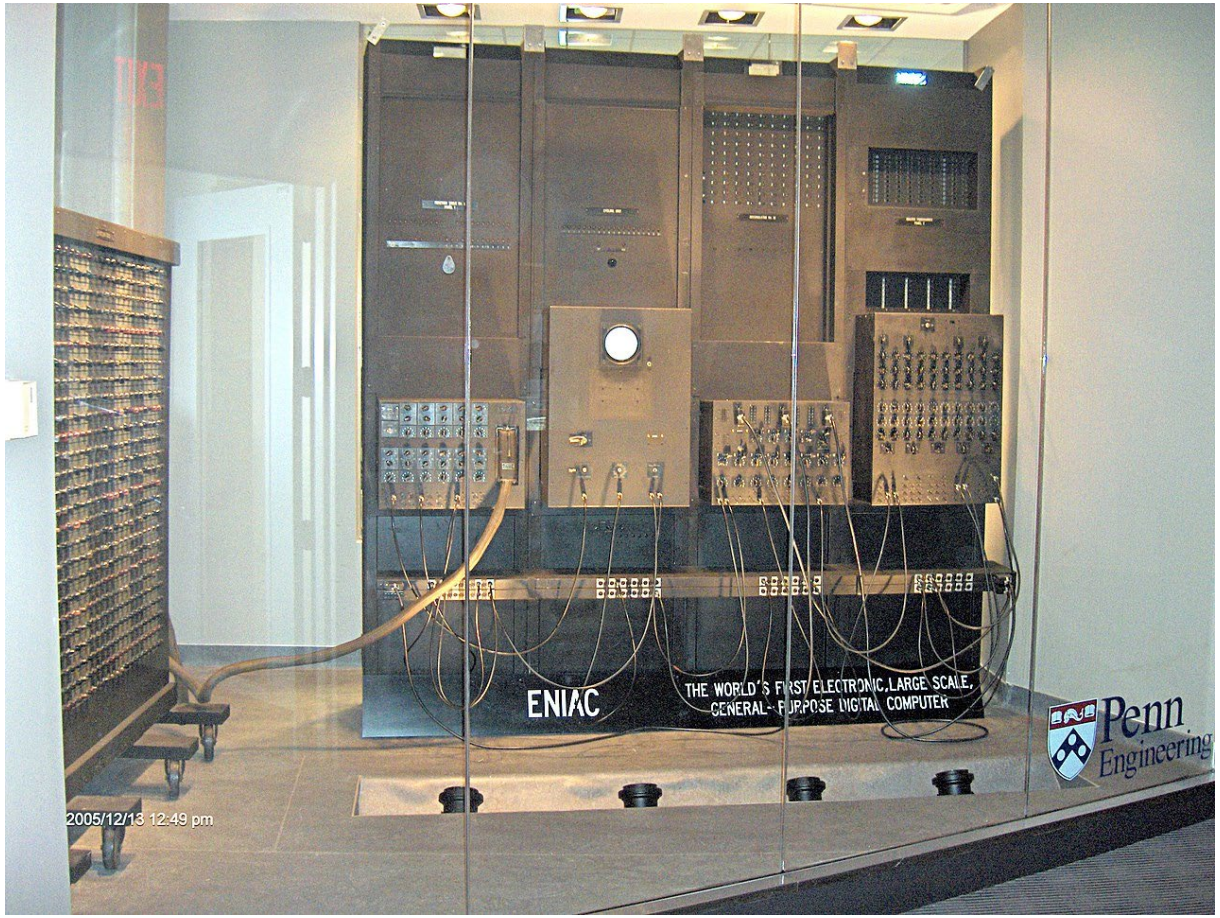
Charles Babbage and His Difference Engine

https://www.youtube.com/watch?v=XSkGY6LchJs

Electronic Numerical Integrator and Computer (ENIAC)

- The first electronic general-purpose computer (1946).
- It has
  - 18,000 vacuum tubes
  - several miles of wiring
  - 40 black eight-foot panels
  - weigh 30 tons
  - occupying the 50-by-30-foot basement of the Moore School of Electrical Engineering at the U Penn.
  - ENIAC could execute up to 5,000 additions per second

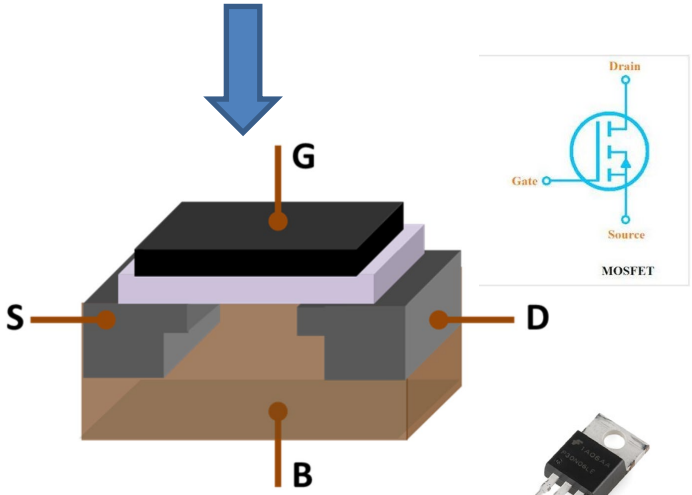


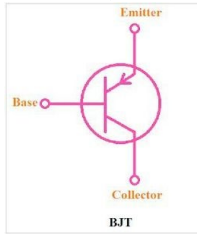


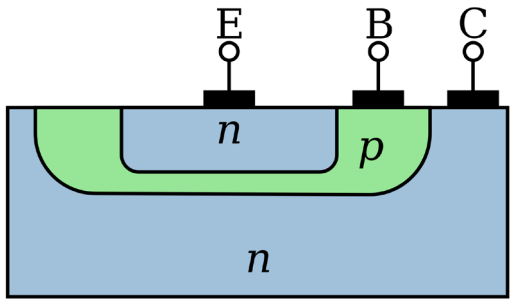
Paul W Shaffer, University of Pennsylvania

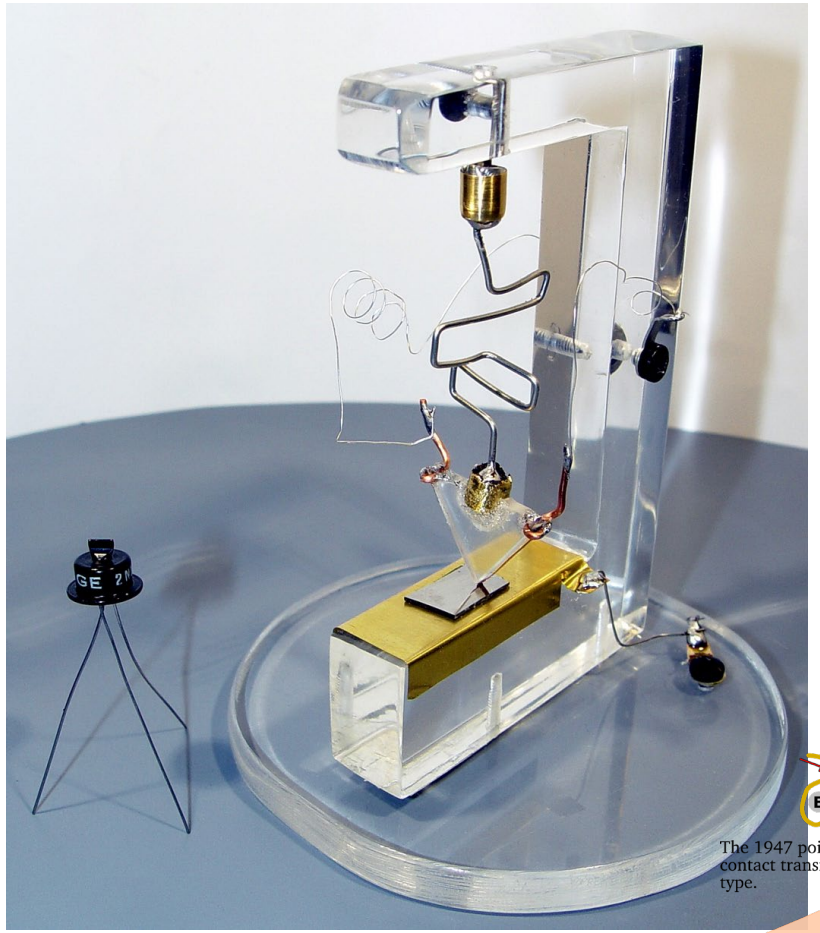
The transistor revolution

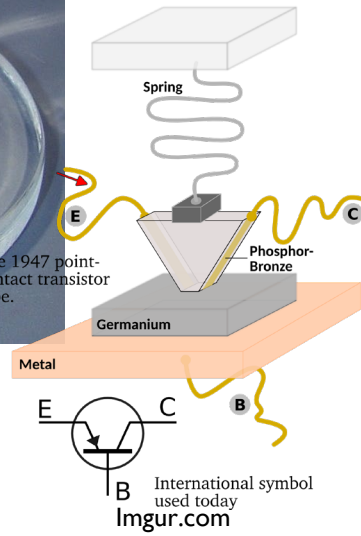
- First transistor developed at Bell Labs (started by Alexander Graham Bell and now is Nokia Bell Lab) in 1947.
  - A point contact transistor
- Bipolar junction transistor in 1948
- MOSFET in 1959







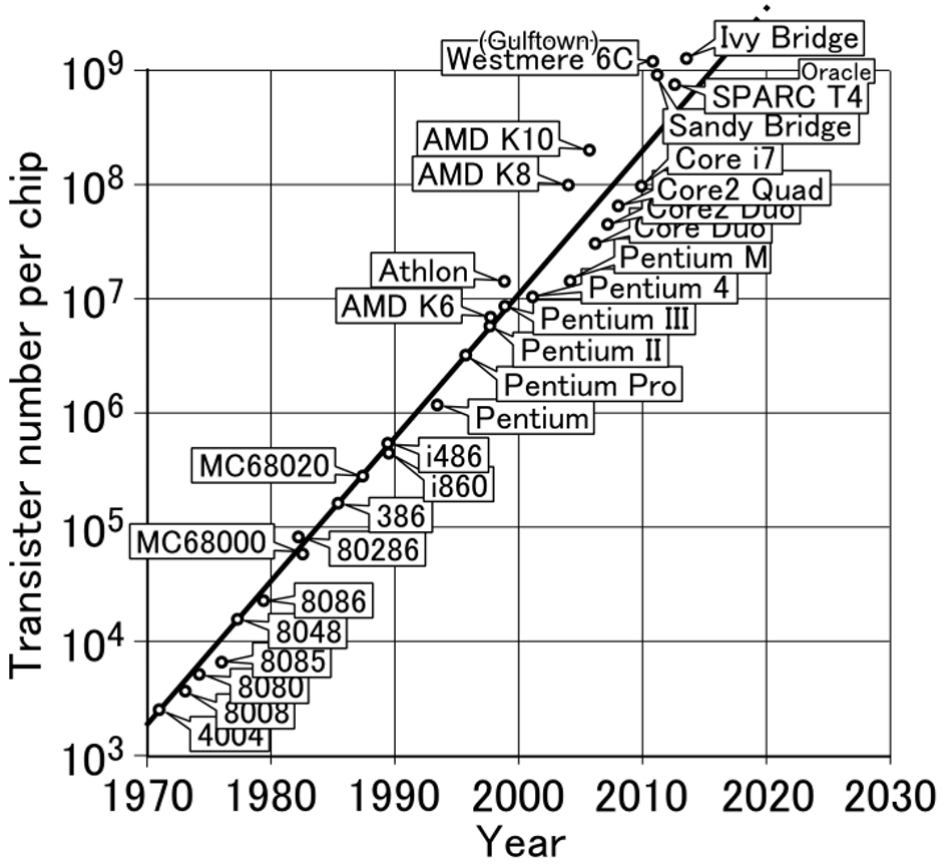




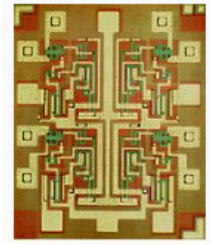
Transistor evolution

- Gordon Moore, Fairchild Semiconductor, 1965 (Intel cofounder in 1968)

(Intel.com, 2018)



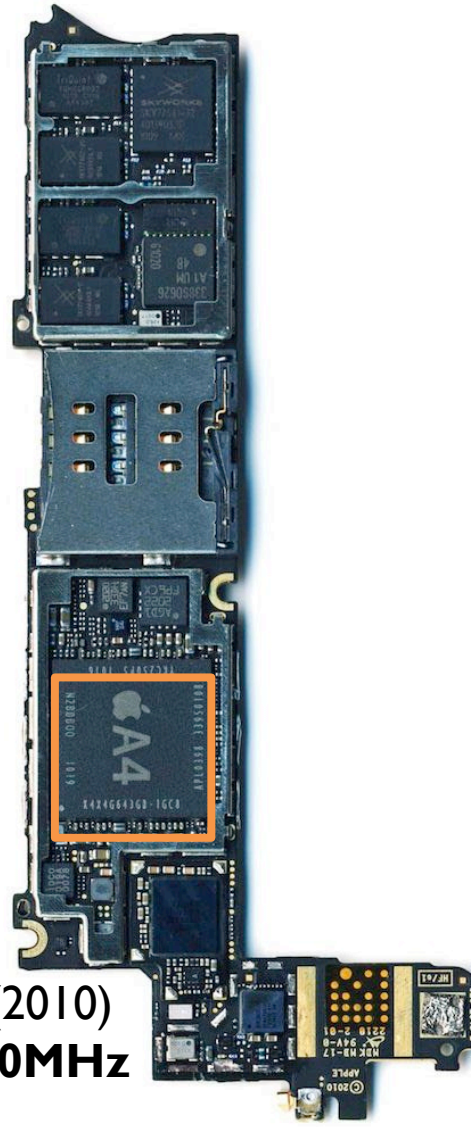
Semiconductor manufacturing processes



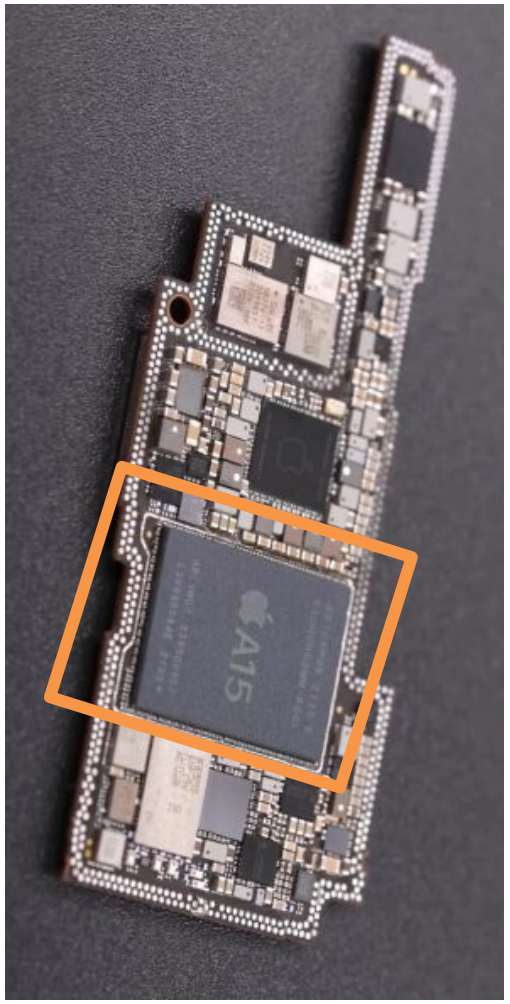
10 µm – 1971
6 µm – 1974
3 µm – 1977
1.5 µm – 1981
1 µm – 1984
800 nm – 1987
600 nm – 1990
350 nm – 1994
250 nm – 1996
180 nm – 1999
130 nm – 2001
90 nm – 2003
65 nm – 2005
45 nm – 2007
32 nm – 2009
22 nm – 2012
14 nm – 2014
10 nm – 2016
7 nm – 2018
5 nm – 2019
3 nm – *~2021*

Moore's article: https://newsroom.intel.com/wp-content/uploads/sites/11/2018/05/moores-law-electronics.pdf

Smartphones



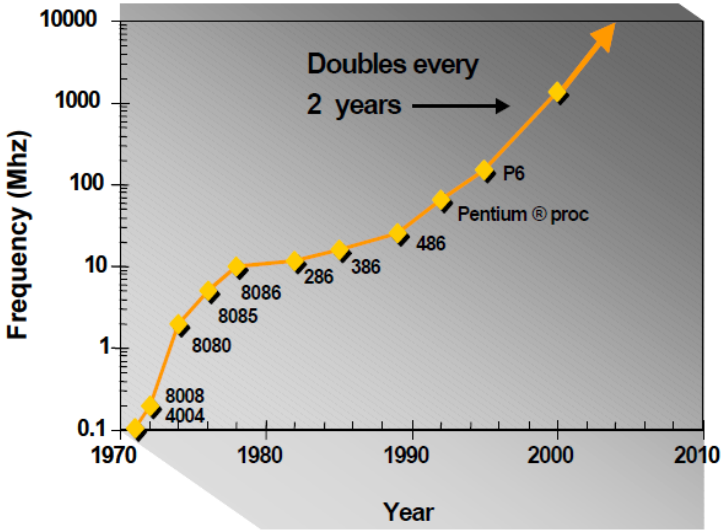
iPhone 4 (2010)
45nm/800MHz



iPhone 13 (2021)
5nm/2.93 GHz

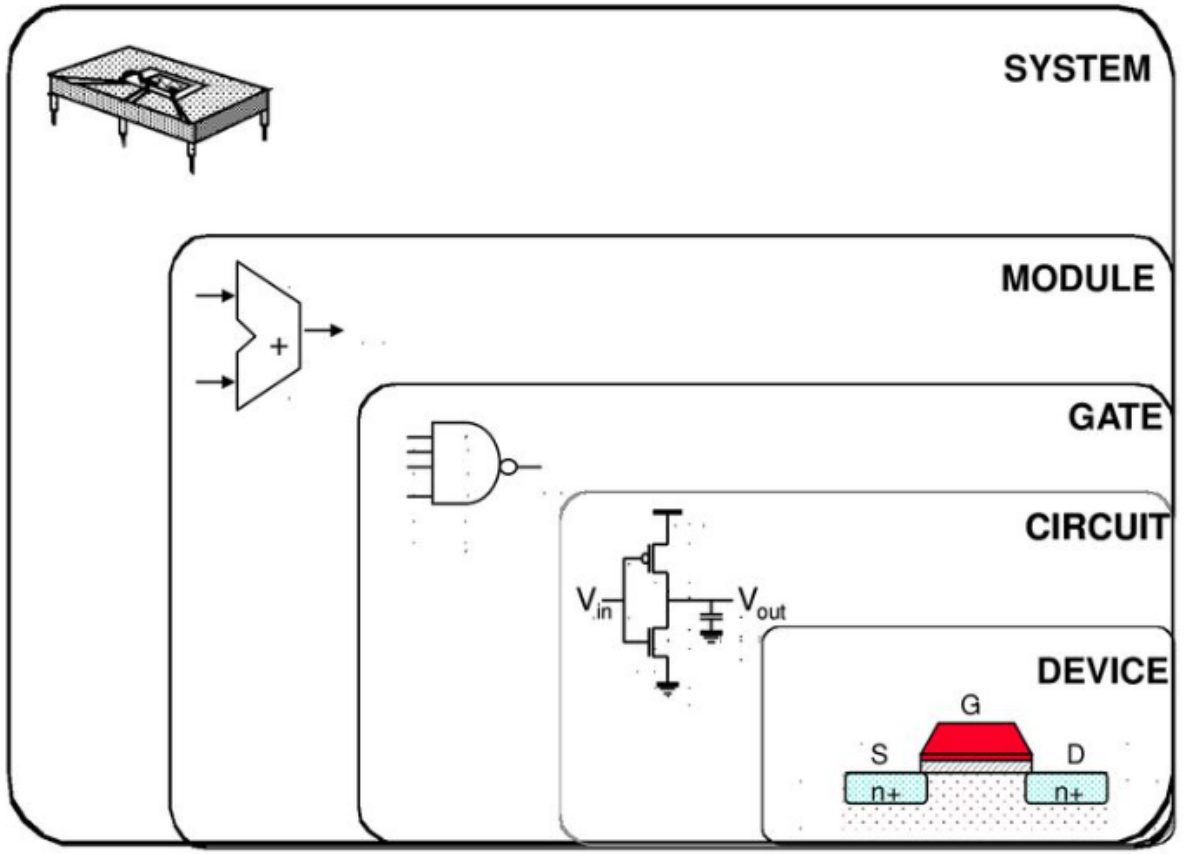
Moore's law

- Number of transistors doubles every two years
- Frequency doubles every two years
- With every generation can integrate 2x more functions per chip; chip cost does not increase significantly
  - Cost of a function decreases by 2x
- But …
  - How to design chips with more and more functions?
  - Design engineering population does not double every two years…
- Hence, a need for more efficient design methods
  - Exploit different levels of abstraction



Abstraction

- Abstraction is to deal with the design complexity
- *Computer-aided design* (CAD) frameworks for digital circuits
  - without it the current design complexity would not have been achievable.
- Design tools include simulation at the various complexity levels, design verification, layout generation, and design synthesis.



Issues in Digital Integrated Circuit Design

Microscopic Problems

- Power Dissipation & distribution
- Clock distribution
- Noise
- Crosstalk
- Reliability
- Manufacturability

Macroscopic Issues

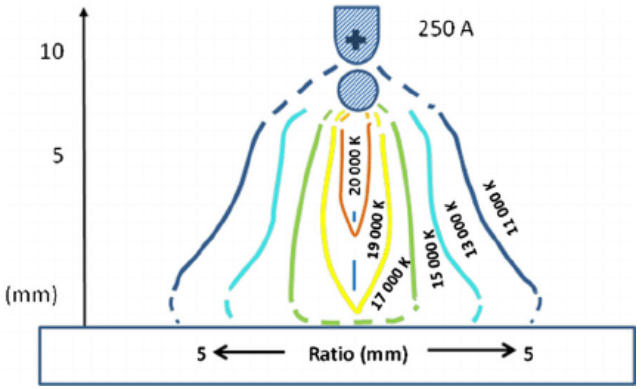
- Time-to-Market
- Millions of Gates
- High-Level Abstractions
- Reuse & IP: Portability
- Predictability
- etc.

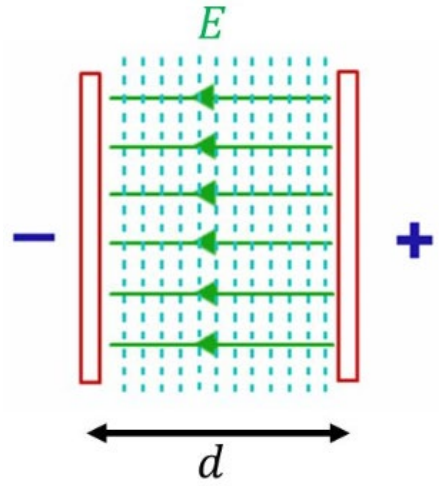
Power dissipation

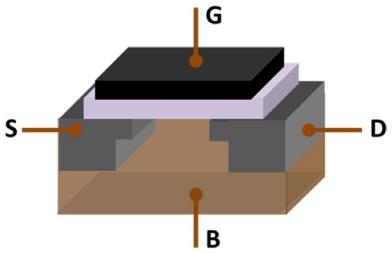
- Power dissipation is a f(speed, area, technology).
- $E = \frac{V}{d}$
- Examples:
  - For welding:

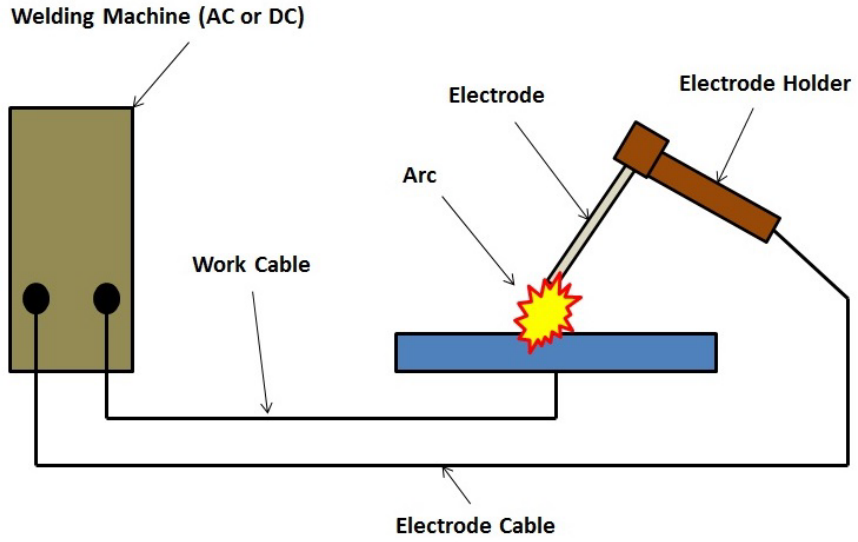
    $E = \frac{5000\ V}{5000\ \mu m} = 1\ V/\mu\ m$
  - Inside the transistor:

    $E = \frac{0.6\ V}{0.06\ \mu m} = 10\ V/\mu\ m$!!!



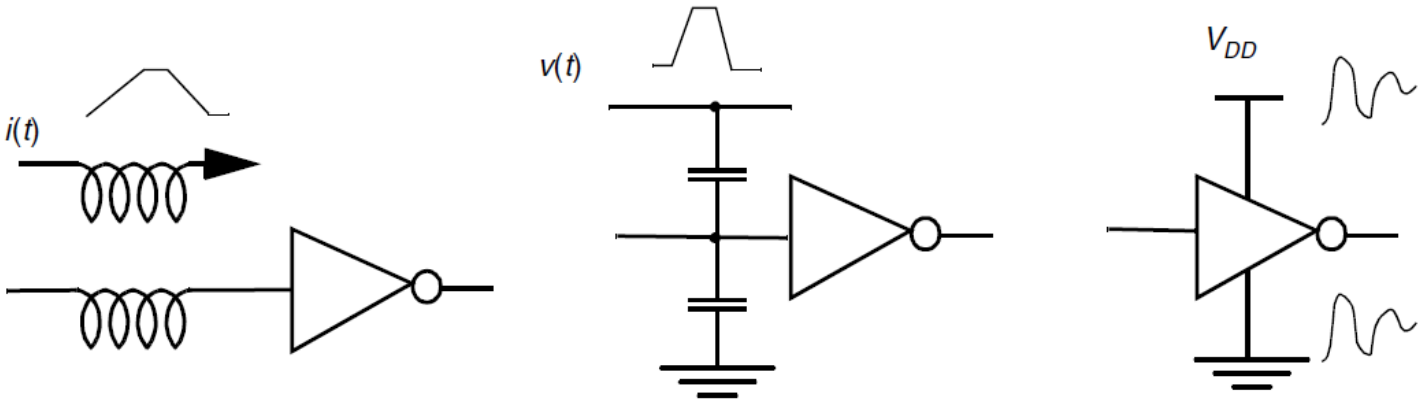






Functionality and Robustness

- Noise in digital circuits means *"unwanted variations of voltages and currents at the logic nodes."*
- *Noise sources in digital circuits:*



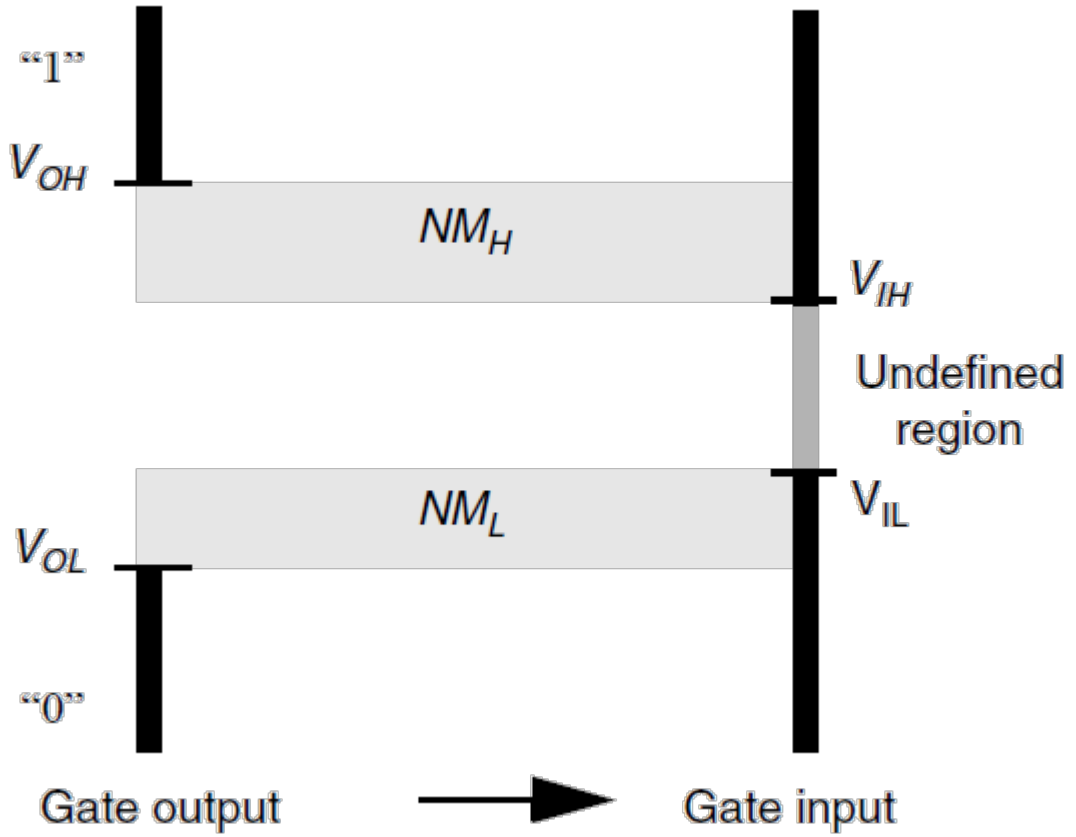
(a) Inductive coupling (b) Capacitive coupling (c) Power and ground noise

Noise Margins

- Digital circuits (DC) perform operations on *logical* (or *Boolean*) variables (0 or 1) that are abstract representation of the real values.
- But phyiscally not
  - $1 \leftrightarrow V_{OH}\ and\ 0 \leftrightarrow V_{OL}$

Noise Margin is how much noise can be added to the logic level and still these level unchanged.

- $NM_H = V_{OH} - V_{IH}$
- $NM_L = V_{IL} - V_{OL}$
- $NM = \min(NM_H, NM_L)$



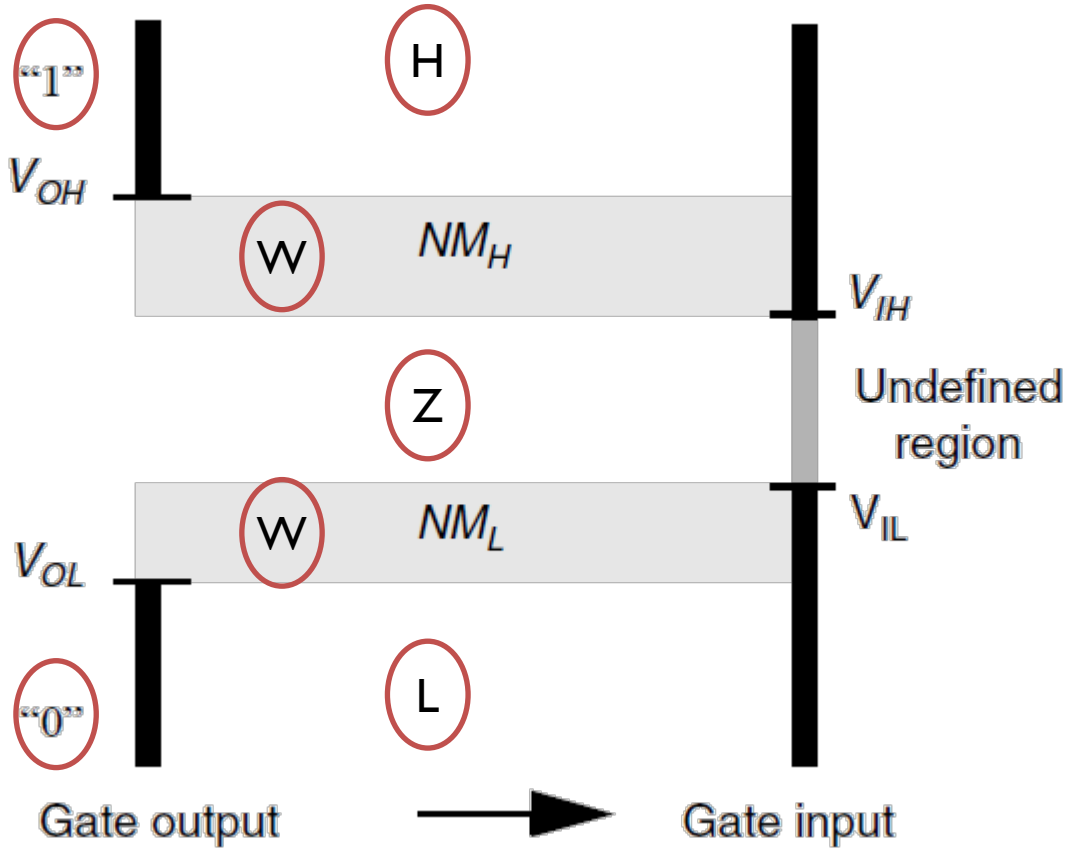
VHDL Programming Language

VHDL

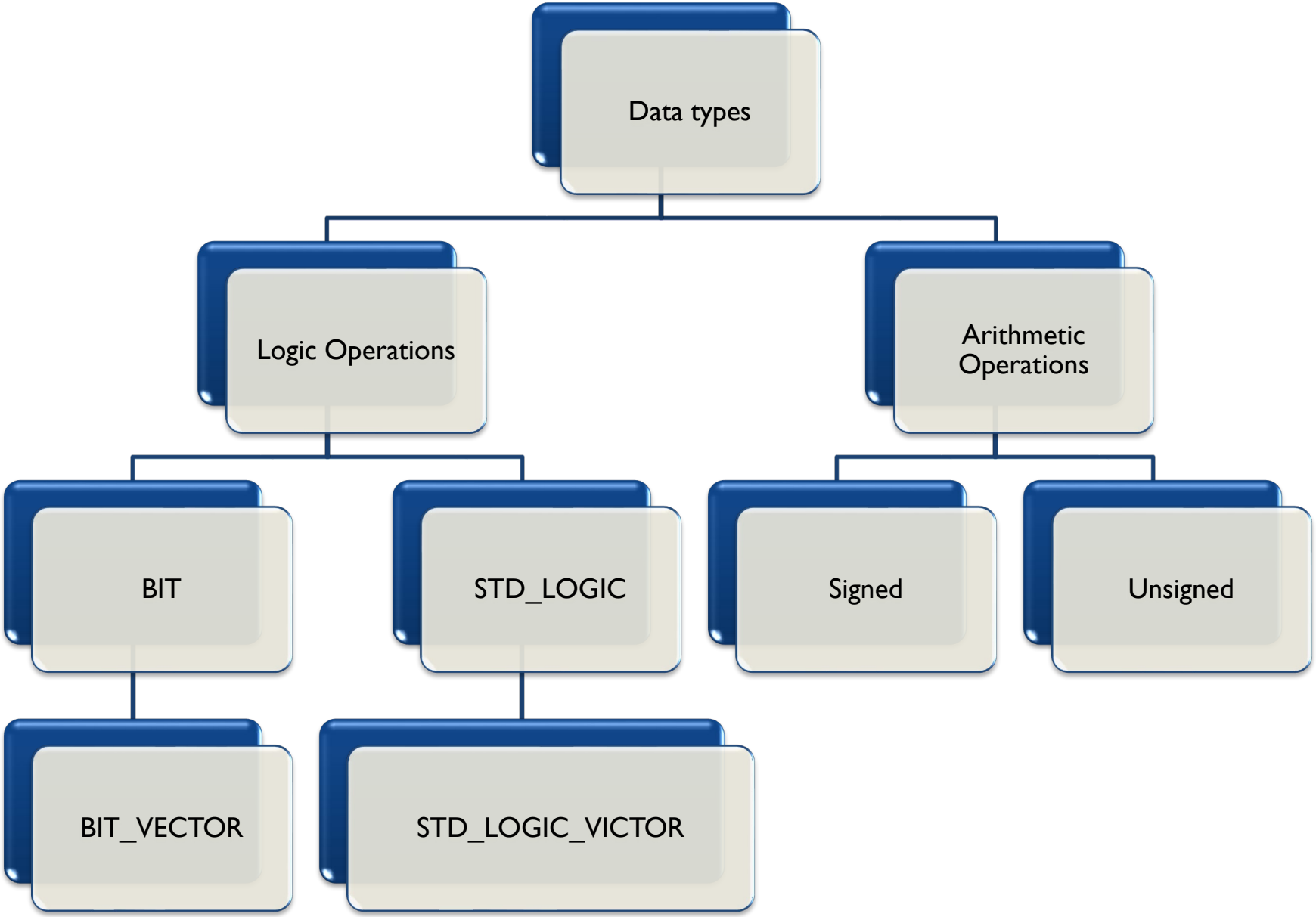
- *Very high speed integrated circuit Hardware Description Language* (VHDL) is a programming language that has been designed and optimized for describing the behavior of digital systems.
- 1980: The USA department of defense (DOD) wanted to make circuit design self documenting.
- 1983: The development of VHDL began with a joint effort by IBM, Texas Instruments and Intermetrics.
- 1987: The Institute of Electrical and Electronics Engineers (IEEE) was presented with a proposal to standardize the language.
- 1993: The VHDL language was revised to IEEE 1076-1993. IEEE published 1164 standards that describes the definitions of logic values to be used.
- 1996: A VHDL package for use with synthesis tools become part of the IEEE 1076 standard, specifically it is 1076.3. This greatly improved the portability of designs between different synthesis vendor tools.

Character	Value
`'U'`	uninitialized
`'X'`	strong drive, unknown logic value
`'0'`	strong drive, logic zero
`'1'`	strong drive, logic one
`'Z'`	high impedance
`'W'`	weak drive, unknown logic value
`'L'`	weak drive, logic zero
`'H'`	weak drive, logic one
`'-'`	don't care

STD_LOGIC



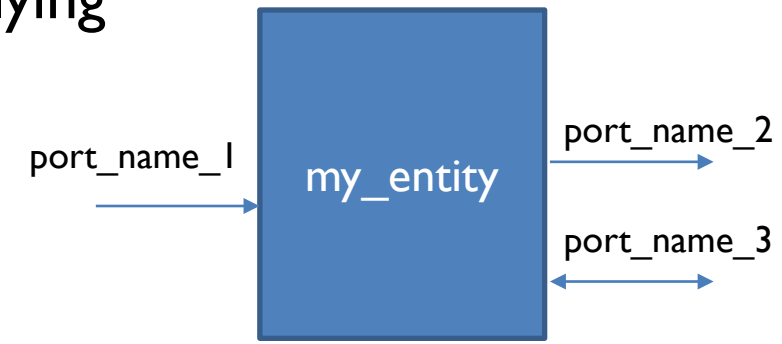
Character	Value
'U'	uninitialized
'X'	strong drive, unknown logic value
'0'	strong drive, logic zero
'1'	strong drive, logic one
'Z'	high impedance
'W'	weak drive, unknown logic value
'L'	weak drive, logic zero
'H'	weak drive, logic one
'-'	don't care


Data types
Logic Operations
Arithmetic Operations
BIT
STD_LOGIC
Signed
Unsigned
BIT_VECTOR
STD_LOGIC_VICTOR

VHDL Design Units

Entity

- The VHDL entity describes how the unit interfaces with the outside world.
- The entity lists the various inputs and outputs of the underlying circuitry.




```
entity my_entity is
port(
     port_name_1 : in    std_logic ;
     port_name_2 : out   std_logic;
     port_name_3 : inout std_logic ); --do not forget the semicolon
end my_entity; -- do not forget this semicolon either
```

VHDL Design Units

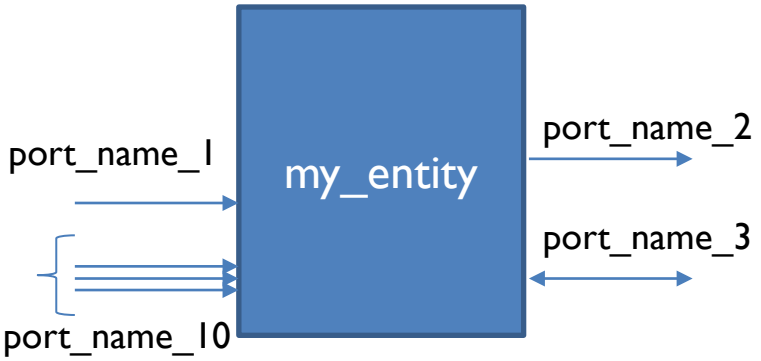
Vector (bus)

A set of similar signals.

port_name_10: in std_logic_vector(2 downto 0);

OR

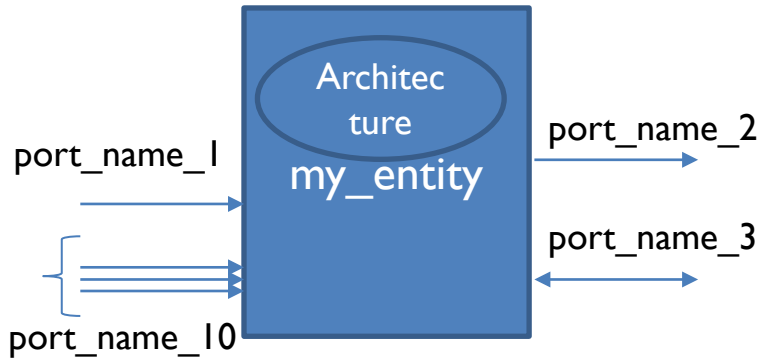
port_name_10: in std_logic_vector(0 to 2);



VHDL Design Units

Architecture:

- The architecture describes what the circuit actually does. In other words, the VHDL architecture describes the internal implementation of the associated entity.
- There is the **data-flow** model, the **behavioral** model, the **structural** model and the **hybrid** models.



Logic Gates

AND

INPUT		OUTPUT
A	B	
0	0	0
1	0	0
0	1	0
1	1	1

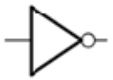
OR

INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	1

XOR

INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	0

NOT

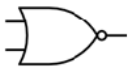


INPUT	OUTPUT
A	
0	1
1	0

NAND

INPUT		OUTPUT
A	B	
0	0	1
1	0	1
0	1	1
1	1	0

NOR



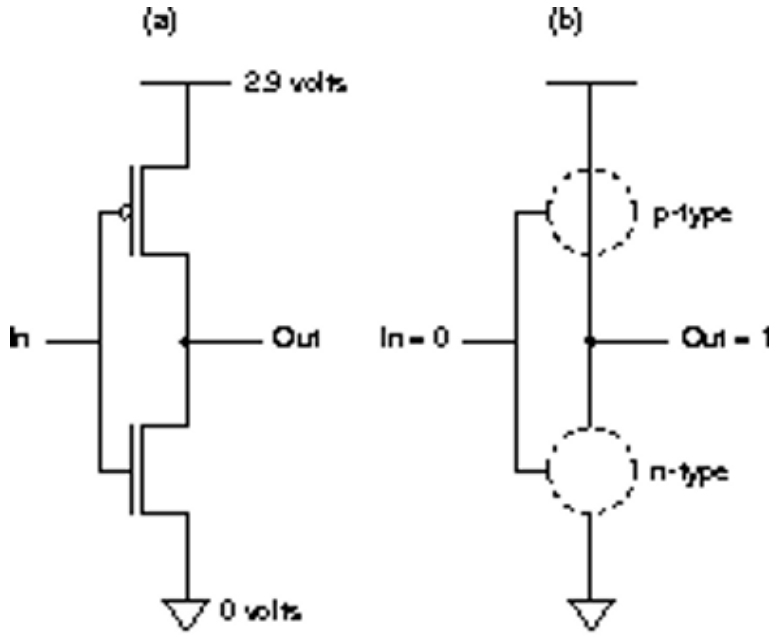
INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	0

XNOR

INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	1

Logic gates implementations

- Inverter



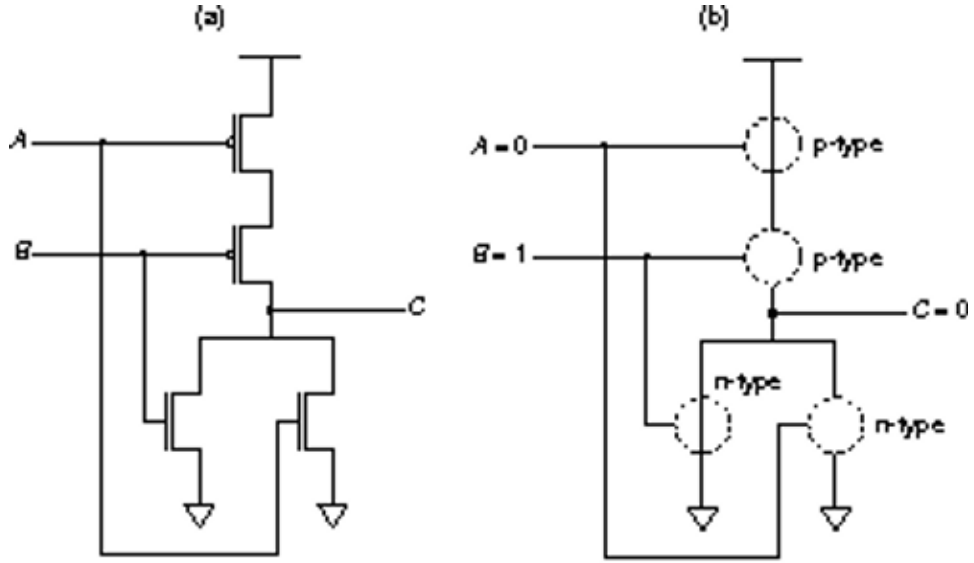
(c)

In	Out
0 volts	2.9 volts
2.9 volts	0 volts

(d)

In	Out
0	1
1	0

- NOR



(c)

A	B	C
0 volts	0 volts	2.9 volts
0 volts	2.9 volts	0 volts
2.9 volts	0 volts	0 volts
2.9 volts	2.9 volts	0 volts

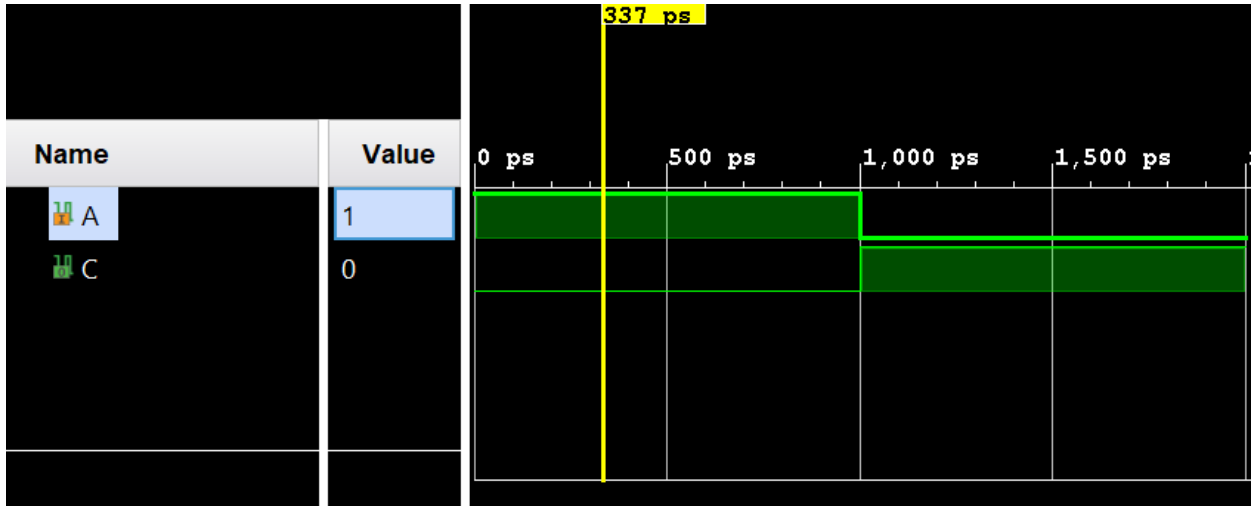
(d)

A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

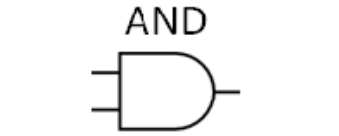
NOT Gate

INPUT	OUTPUT
A	
0	1
1	0

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity NOT_Gate is
    Port ( A : in STD_LOGIC;
           C : out STD_LOGIC);
end NOT_Gate;

architecture Behavioral of NOT_Gate is

begin

C <= not(A);

end Behavioral;
```

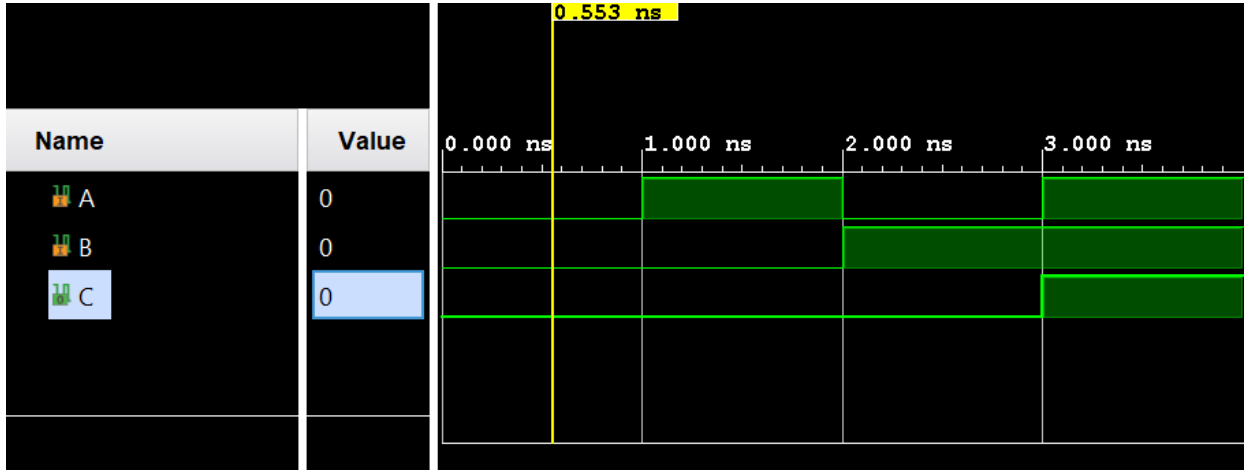


AND Gates



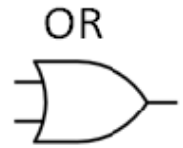
INPUT		OUTPUT
A	B	
0	0	0
1	0	0
0	1	0
1	1	1

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity AND_Gate is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end AND_Gate;

architecture Behavioral of AND_Gate is

begin
C <= A and B;

end Behavioral;
```



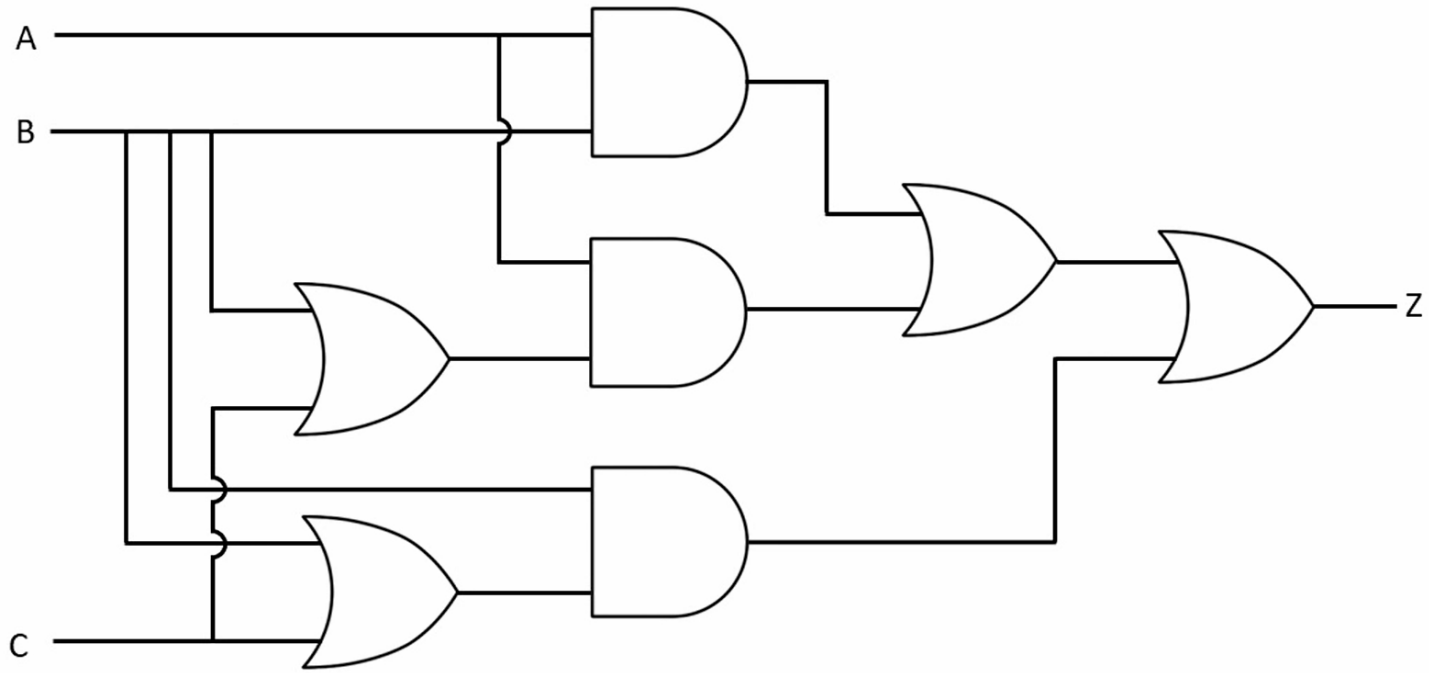
OR Gates

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity OR_Gate is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end OR_Gate;

architecture Behavioral of OR_Gate is

begin
C <= A or B;

end Behavioral;
```



INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	1

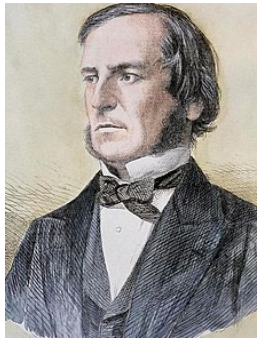
Complex logic circuit



$Z = A \wedge B \vee A \wedge (B \vee C) \vee B \wedge (B \vee C)$

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

The Laws of Boolean Algebra

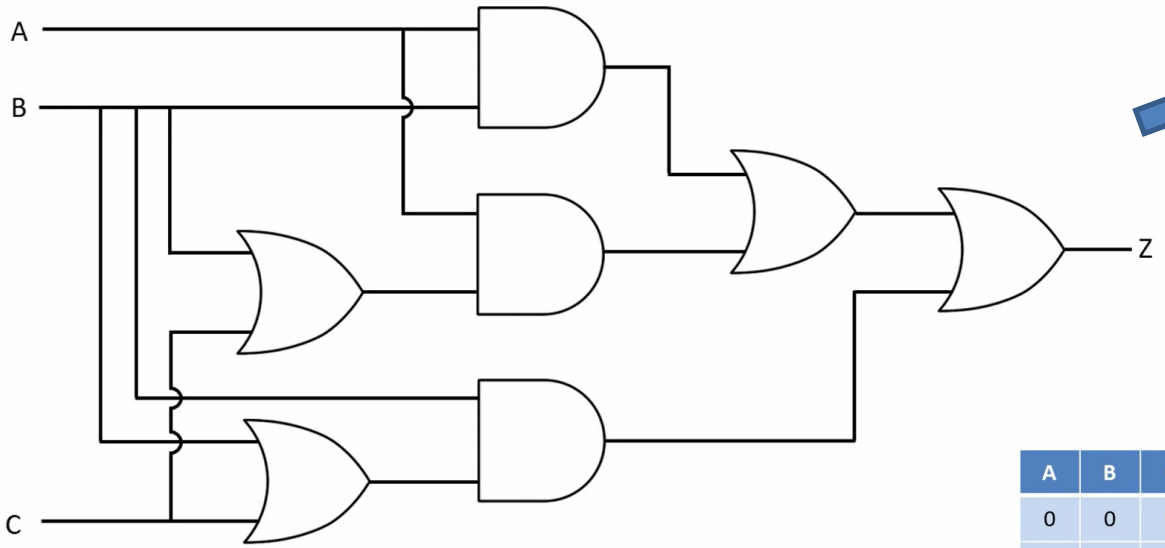


- **George Boole** (1815-1864) invented the Boolean Algebra by codifying logic in algebraic form.
- This work formed the basis for the digital logic that drives today's computer and communications technologies.
- In digital electronics, Boolean Algebra is used to simplify the design of complex circuits.

+ -> or
* -> and
/ -> not

Law/Theorem	Law of Addition	Law of Multiplication
Identity Law	$x + 0 = x$	$x \cdot 1 = x$
Complement Law	$x + x' = 1$	$x \cdot x' = 0$
Idempotent Law	$x + x = x$	$x \cdot x = x$
Dominant Law	$x + 1 = 1$	$x \cdot 0 = 0$
Involution Law	$(x')' = x$	
Commutative Law	$x + y = y + x$	$x \cdot y = y \cdot x$
Associative Law	$x+(y+z) = (x+y)+z$	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$
Distributive Law	$x \cdot (y+z) = x \cdot y+x \cdot z$	$x+y \cdot z = (x+y) \cdot (x+z)$
Demorgan's Law	$(x+y)' = x' \cdot y'$	$(x \cdot y)' = x' + y'$
Absorption Law	$x + (x \cdot y) = x$	$x \cdot (x + y) = x$

Complex logic circuit



$Z = A \wedge B \vee A \wedge (B \vee C) \vee B \wedge (B \vee C)$

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$Z = A \wedge B \vee A \wedge (B \vee C) \vee B \wedge (B \vee C)$

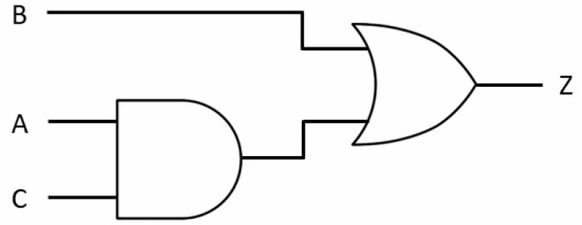
$= A \wedge B \vee (A \wedge B) \vee (A \wedge C) \vee (B \wedge B) \vee (B \wedge C)$

$= A \wedge B \vee (A \wedge B) \vee (A \wedge C) \vee B \vee (B \wedge C)$

$= (A \wedge B) \vee (A \wedge C) \vee B \vee (B \wedge C)$

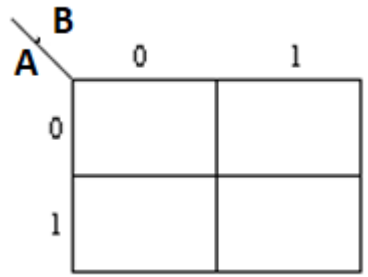
$= (A \wedge B) \vee (A \wedge C) \vee B$

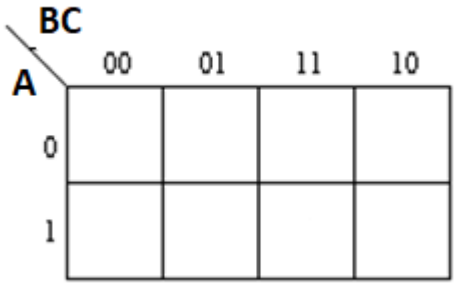
$= (A \wedge C) \vee B$

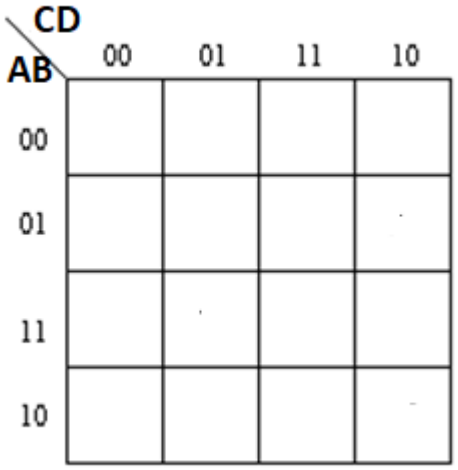


Karnaugh map (K-Map)

- K-MAP is another method of circuit optimization, introduced by Maurice Karnaugh in 1953
- It aims to reduce logic functions more quickly and easily compared to Boolean algebra.





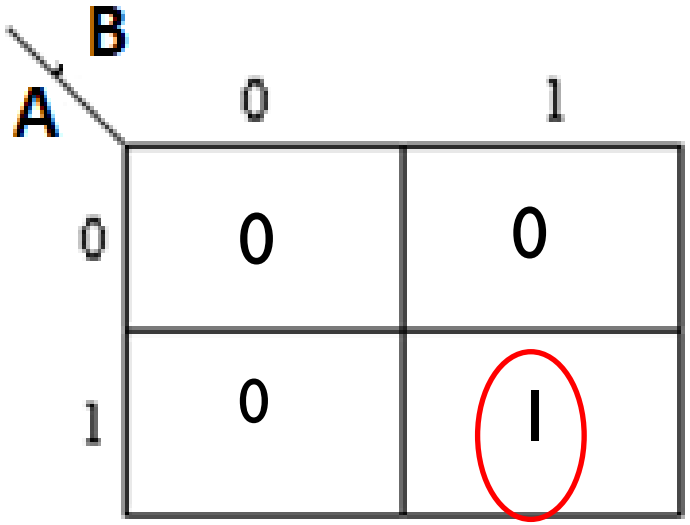


Karnaugh map (K-Map)

- AND

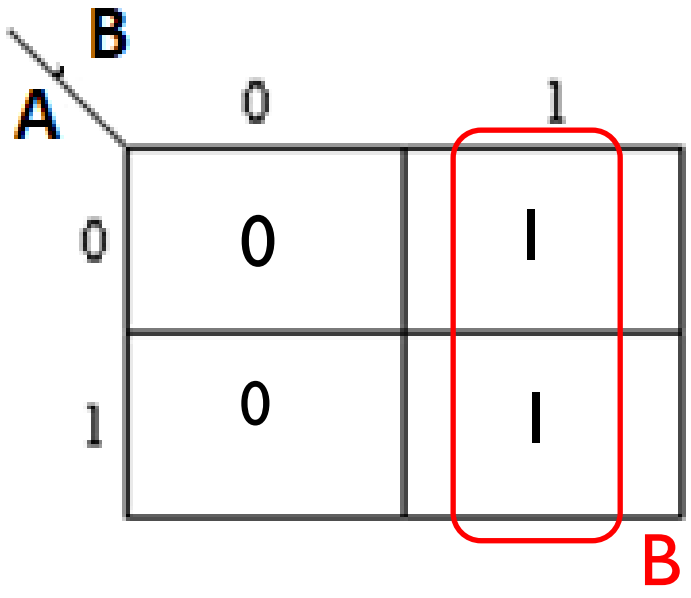
AND

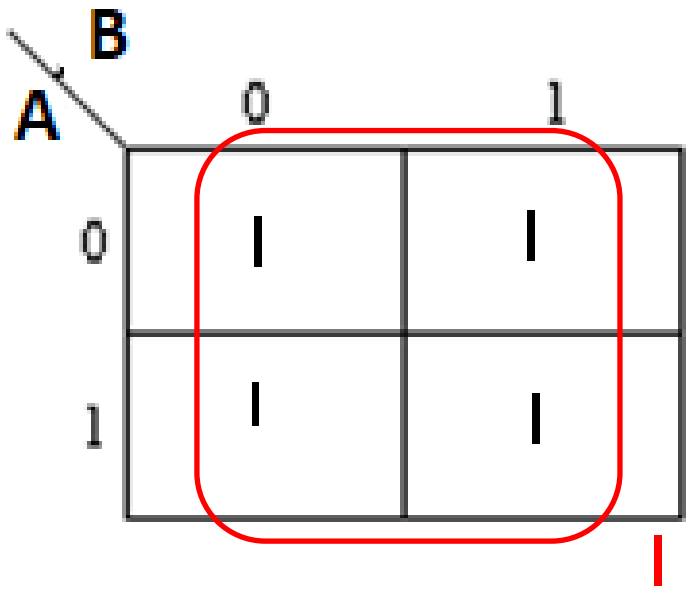
INPUT		OUTPUT
A	B	
0	0	0
1	0	0
0	1	0
1	1	1

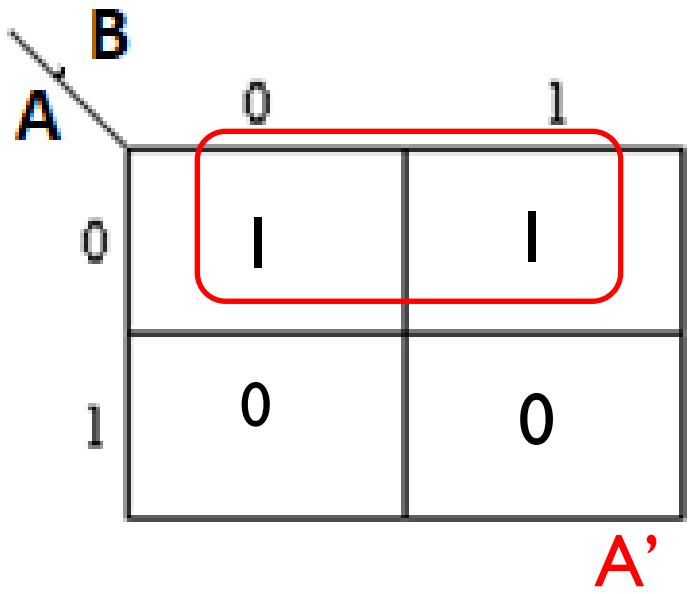


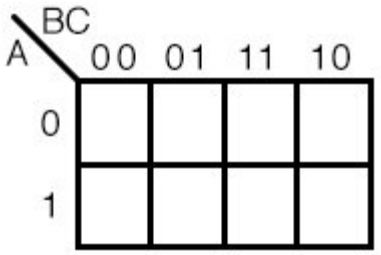
AB

Karnaugh map (K-Map)

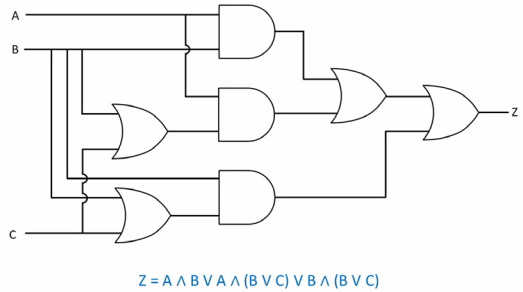








Karnaugh map (K-Map)

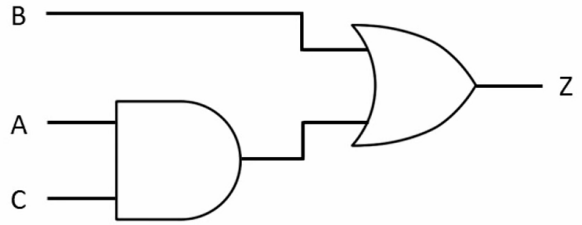


- K-MAP is another method of circuit optimization, introduced by Maurice Karnaugh in 1953, aims to reduce logic functions more quickly and easily compared to Boolean algebra.

BC

A	00	01	11	10
0	0	0	1	1
1	0	1	1	1

B+AC



A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Digital circuit designs in VHDL

Combinational circuits

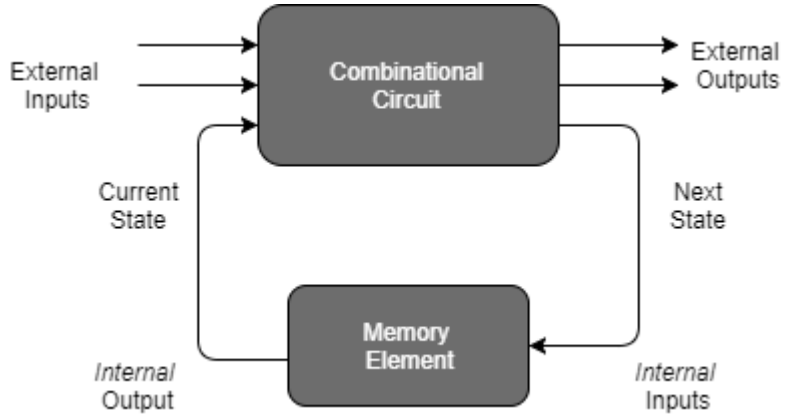
They are defined as the *time independent circuits* which do not depends upon previous inputs to generate any output are termed as combinational circuits.

Examples – Encoder, Decoder, Multiplexer, Demultiplexer, etc.

Sequential circuits

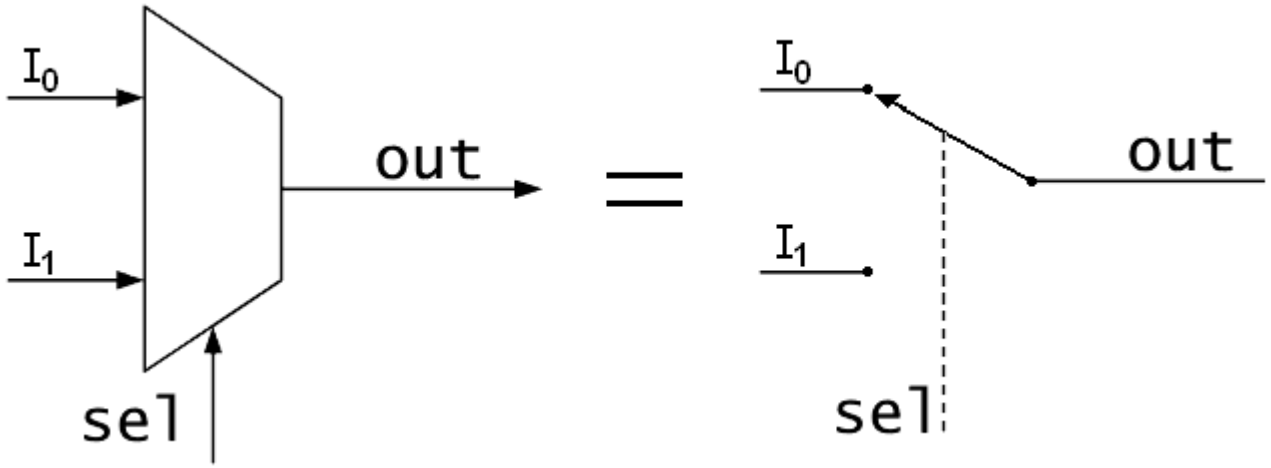
They are those which are *dependent on clock cycles* and depends on present as well as past inputs to generate any output.

Examples – Flip-flops, Counters, Registers, etc.



Multiplixer

- It function is to select one of the inputs and connect it to the output



Sel	I_0	I_1	Out
0	0	0	0
	0	1	0
	1	0	1
	1	1	1
1	0	0	0
	0	1	1
	1	0	0
	1	1	1

10 minutes exercise: Draw the equivalent logic circuit using boolean algebra and k-map

Multiplixer

Answer