



COMPUTER ARCHITECTURE

1 Introduction

1. Introduction :

- ❑ 1.1 CA course
- ❑ 1.2 About computers
- ❑ 1.3 Basic computer structure and operation
- ❑ 1.4 Analog – digital, continuous-discrete
- ❑ 1.5 8 important ideas in computer architecture (and in general)
- ❑ 1.6 Computer realization

1. Introduction :

1.1 CA course

1.2 About computers

1.3 Basic computer structure and operation

1.4 Analog – digital, continuous-discrete

1.5 8 important ideas in computer architecture (and in general)

1.6 Computer realization

- Web classroom: <http://ucilnica.fri.uni-lj.si>
<https://padlet.com/rawall/RAWall>

- MS Teams
 - Team enter code: 277pdvh

- Office hours: on schedule in R2.40, 50

Possible changes will be posted to the web classroom. Announce over email.



Team CA

Instructors

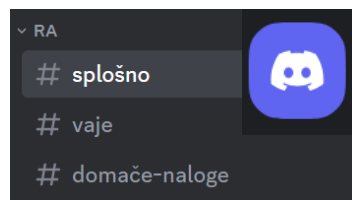


Žiga Pušnik
ziga.pusnik@fri...



Romanela Lajić
romanela.lajic@fr...

Tutors



<https://discord.gg/nmzjQU7me7>



Robert Rozman
rozman@fri.uni-lj.si



■ Literature:

- Lecture content, lab. exercises and slides (also in English)
 - <http://ucilnica.fri.uni-lj.si>

- MS Teams (chat, lecture notes)



- Common (shared) notes – Gdocs

----- Skupni zapiski/Shared course notes -----

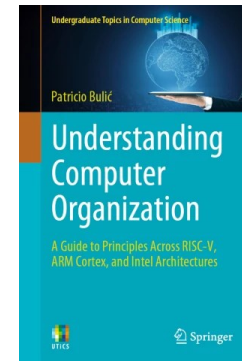
Computer Architecture - Crowd-sourced Shared Notes

Računalniška Arhitektura - Deljeni zapiski za skupno dopolnjevanje

- Basic, includes more comprehensive content than needed:
 - Dušan Kodek: ARHITEKTURA IN ORGANIZACIJA RAČUNALNIŠKIH SISTEMOV, Bi-TIM, 2008



- Additional (only certain parts – 5 copies in library FRI):
 - P. Bulić: Understanding Computer Organization



- Additional (only certain parts – 4 copies in library FRI):
 - Andrew S. Tanenbaum: STRUCTURED COMPUTER ORGANIZATION, Sixth Edition Pearson Prentice Hall, 2013



Important :

- There are no silly questions,
 - silly are just those that don't ask
- You're always welcome
- We all make efforts

Povpr. ocena/max. [št. odg./vsij]†	2022/23-Nosilec†	2021/22-Nosilec†	2020/21-Nosilec†	2019/20-Nosilec†	2018/19-Nosilec†
Predmet†	4.56/5·[173/184]†	4.53/5·[192/196]†	4.65/5·[154/161]†	4.54/5·[149/150]†	4.66/5·[138/164]†
Izvajalec†	4.72/5·[173/184]†	4.75/5·[192/196]†	4.74/5·[154/161]†	4.79/5·[149/150]†	4.77/5·[138/164]†


Surveys (2018/23) - highlights:

- Kahoot! I learned a lot from kahoot and the RAwall.
- **OneNote notebooks** with notes, useful and interesting exercises, Kahoots, good pdfs
- Current news were often mentioned in the lectures, which encouraged critical and independent thinking
- Great emphasis is placed on understanding, and not on learning by heart
- The lecturer's energy, practical (life) examples
- **A good learning system for foreign students**
- ... the assistants and especially the professor can see that they don't care about our knowledge. Because of this, the quality of lectures and exercises is known.

To Improve:

- Theory sometimes becomes difficult to understand, it is difficult to follow new concepts and ideas.
- Lecture material <> exercise
- "Not perfect, which is characteristic of everyone"

What's important in 2024:

- Live lectures and lab sessions (1,2 lab sessions in English – day ?)
 - Previous LAB sessions recordings
 - <https://unilj.sharepoint.com/sites/LAPSYEmbeddedAcademy/>
- Platforms :
 - e-classroom <http://ucilnica.fri.uni-lj.si>
 -  Computer Architecture - Crowd-sourced Shared Notes
 - MS Teams (board notes, communication)
 - Team entry code : 277pdvh
- Important :
 - be active
 - cooperate, talk, ask, comment, ...
 - all major documents are translated to English
 - occasional testing of realtime Slovene-English translation – project ON
 - please help us on English documents (typos, missing content, ...)



<https://dis-slovarcek.ijs.si/>

Na Univerzi v Ljubljani letos študira dobrih 40 tisoč študentov. Devet odstotkov jih prihaja iz tujine.

What's new in 2024:

This year, more than 40,000 students are studying at the University of Ljubljana. Nine percent of them come from abroad.

■ Live lecture recognition to Slovene and translation to English text

- testing of real-time Slovene-English translation – [project ON](#)

■ Access link :

- <https://on.uni-lj.si>
- using your official student e-mail identity (*@student.uni-lj.si).

occasional testing of realtime Slovene-English translation – [project ON](#)

■ Please fill introductory survey :

- <https://1ka.arnes.si/ON-2022>

■ Further instructions will follow

■ Contact:

- tjasa.jelovsek@fri.uni-lj.si

ONLINE NOTES (ON)

RAZVOJ SISTEMA ZA AVTOMATIZIRANO PREVAJANJE SLOVENSКИH PREDAVANJ V TUJE JEZIKE

Projekt **Online Notes (ON)** je namenjen razvoju sistema za avtomatizirano prevajanje slovenskih predavanj v tuje jezike. Razvoj poteka pod vodstvom Fakultete za računalništvo in informatiko Univerze v Ljubljani v sodelovanju s Centrom za jezikovne vire in tehnologije Univerze v Ljubljani. Cilj projekta je povečati dostopnost predavanj za tuje študente doma in v tujini ter za ciljne skupine s senzornimi oviranostmi.

Rezultati projekta bodo koristni za več skupin: prevodi v živo bodo tujim študentom olajšali spremljanje predavanj, ki potekajo v slovenskem jeziku; prepisi v slovenskem jeziku bodo povečali dostopnost gluhim in naglušnim; izvlečki in posnetki predavanj, ki bodo dosegljivi preko posebnega portala, pa bodo odlično dopolnjevali elektronska gradiva za študij.

<https://www.cjvt.si/infrastruktura-podpora/online-notes/>




Računalniška arhitektura

[Predmet](#) [Nastavitve](#) [Sodelujoči](#) [Ocene](#) [Poročila](#) [Več](#) ▼

▼ Splošno - General

 [Anketa o predmetu Računalniška arhitektura \(2023/24\)](#)

 Skrito za udeležence

 [Splošne informacije - General info](#)

MS Teams ekipe: RA VSP 2024/25 - izvedba predmeta (koda: 277pdvh), LAPSy Embedded Academy - stalni FRI HW portal (koda: ty5qjm9)

 [Discord Hardware FRI - skupnost za HW predmete](#)

 [Forum novic - News Forum](#)

 [Forum - RA - vprašanja in odgovori - Q&A](#)

MS Teams: chat, OneNote notebook

The screenshot displays the MS Teams interface with a OneNote notebook open. The notebook is titled "Class Notebook" and is part of a team named "RA VSP 2024/25". The notebook's content is organized into sections and pages. The current page is titled "Kako biti uspešen?" and contains handwritten notes in green, blue, and black ink. The notes include:

- **RADOVEDNOST** (green)
- **RAZUMEVANJE = ZNANJE** (blue)
- **AKTIVNO SODELOVANJE** (black)

The notes are accompanied by a large blue curly bracket on the right side, and a double-headed arrow and the letter 'M' are also visible. The notebook's interface includes a search bar at the top, a ribbon with various editing tools, and a table of contents on the left side of the page.

Chapter related content

<https://padlet.com/rawall/RAWall>

Stalni viri

- RAWall 3yr: Padlet lahko uporabite za svoje potrebe. S prijavo spodaj nam prinesete free padlet.
- RRobi 3yr: FRI E-učilnica. Računalniška arhitektura.
- RRobi 3yr: MS Teams. Team code: y331dds.

Vsebina

- RRobi 3yr: 1.Uvod : 1.1 Predmet RA 1.2 Računalniki včeraj in danes 1.3 Osnove zgradbe in delovanja računalnikov 1.4 Analogno – digitalno, zvezno diskretno 1.5 8 pomembnih idej v računalniški arhitekturi (in širše) 1.6 Praktična realizacija računalnikov

Moja vprašanja

- RAWall 1yr: Občutki ?
- RRobi 2yr: VEGA HPC na top500.org
- RRobi 3yr: Dewesoft & Atlantis

Viri

- RRobi 3yr: Katero srednjo šolo ste končali ? (klik na sliko)

Vprašanja, komentarji

- RRobi 3yr: Premik vprašanj in odgovorov

Tedenska vsebina – 2024/25

Možnost anonimnih vprašanj, aktivnega sodelovanja, izrekanje mnenj

Join at
slido.com
#RA1

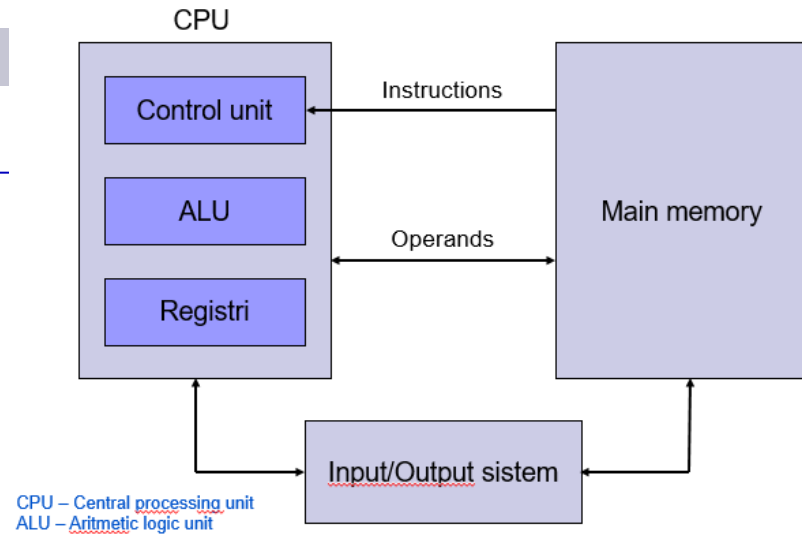


What will you learn on the course of
Computer Architecture?

Lectures, Lab sessions

Lectures content:

- CA-1 Introduction
- CA-2 Development of computing machines
- CA-3 Basic computing principles
- CA-4 Assembly Instructions
- CA-5 Operands - representation of information
- CA-6 Structure and operation of CPU
- CA-7 CPU performance measurement
- CA-8 Memory
- CA-9 Memory hierarchy



Laboratory work contents:

- Learn the **basics of computer architecture** from a practical point of view
- **Understand the inner workings of computers (ARM)** by programming in assembly language
- In-depth view:
 - into **computer operation**
 - into **program execution** on computers

Further knowledge upgrade -> Computer Organization elective course and other related courses

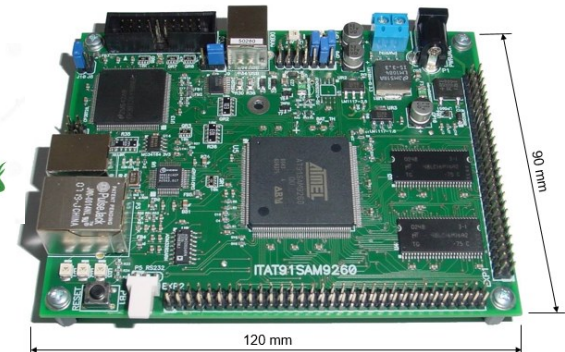
Computer STM32H750-DK



Two practical, interesting LAB sessions

You can get your own computer

- FRI-SMS computer (somewhere in-between)
 - Microcontroller AT91SAM9260 of the ARM9 microcontroller family



Why is Computer Architecture important ?

- 5 questions
and ...
- 5 answers

1. Complete products (HW+SW)

Success stories (HW+SW)



Make your home healthier,
your office more productive

Uncover the simple solutions. With just a small, stylish, cordless
and connected Cube in each room.

Get Your Cubes Now!

Winter 2013 batch available!

Potato Salad
by Zack Danger Brown

Comments 1,127

This project was successfully funded on August 2

Columbus, OH

6,911 backers

\$55,492
pledged of \$10 goal

0 seconds to go

Chipolo - Bluetooth Item Finder for iPhone and Android
by The Chipolo Team

Home Updates 17 Backers 5,329 Comments 1,011

Funded! This project was successfully funded on November 15, 2013

Trbovlje, Slovenia Technology

Chipolo
Nothing is lost.

GO:GLOBAL MEMBER | SPS SK200 AUTUMN BATCH 2014

Chipolo
Finalisti tekmovanja Start:up leta 2016

5,329 backers

\$293,014
pledged of \$15,000 goal

0 seconds to go

Project by
The Chipolo Team
Trbovlje, Slovenia

First created - 0 backed

Has not connected Facebook



COSYLAB



OPEN INSTRUMENTS
FOR EVERYONE



826 backers

\$256,125
pledged of \$50,000 goal

0 seconds to go

Funding period
Jul 22, 2013 - Sep 20, 2013 (60 days)



Project by
Red Pitaya
Newport News, VA

DEWESoft

YEAR WARRANTY

FRI

2. Efficient programming

- Because knowledge on computer architecture and operation leads to more efficient programming (programs).
 - Case 1: program code optimization regarding the operation of caches

```
/* Before */
for (j = 0; j < 100; j = j + 1)
    for (i = 0; i < 5000; i = i + 1)
        x[i][j] = 2 * x[i][j];

/* After */
for (i = 0; i < 5000; i = i + 1)
    for (j = 0; j < 100; j = j + 1)
        x[i][j] = 2 * x[i][j];
```

2. Efficient programming

- Because knowledge on computer architecture and operation leads to more efficient programming (programs).
 - Case 2: program code optimization regarding the parallel execution

us/Iteration	Iterations/sec
2.02500	493827.16
0.53300	1876172.61

```
double results[st];  
  
for(int i = 0; i < st; ++i)  
{  
    results[i] = a[i] * b[i];  
}
```

```
float results[st];  
  
for(int i = 0; i < (st - 8); i += 8)  
{  
    __m256 i_a = _mm256_load_ps(&a[i]);  
    __m256 i_b = _mm256_load_ps(&b[i]);  
    __m256 i_c = _mm256_mul_ps(i_a, i_b);  
    _mm256_store_ps(&results[i], i_c);  
}  
  
for(int i = (st - 8); i < st; ++i)  
{  
    results[i] = a[i] * b[i];  
}
```

Code below is **4-times faster** !

Reference: „Pomen poznavanja računalniške arhitekture“,
avtor Miha Krajnc.

3. Why still assembly?

Assembler e.g. machine language is the only one, that computer understands and can execute.

„who still knows this language?“

3. Why still assembly?

Kje je pomembno znanje arhitekture?

- Če programer ve kako deluje prevajalnik in zbirnik, lahko lažje in hitreje reši napake v kodi.
- Omogoči pisanje **hitrejših programov**.
- Programerji razumejo relativno ceno operacij (**CPI**) in **učinke različnega načina pisanja programa**

Zakaj potem ne programiramo v zbirniku?

V bistvu bomo programirali z posebnimi metodami, ki se neposredno prevedejo v zbirne ukaze.

Additional article in Slovene (e-učilnica): „Pomen poznavanja računalniške arhitekture“, author Miha Krajnc.

3. Why still assembly? One of the answers

[Dejan Črnica, Dewesoft]:

„because it's „polite“ to learn the native language, culture...“

Past Meetup

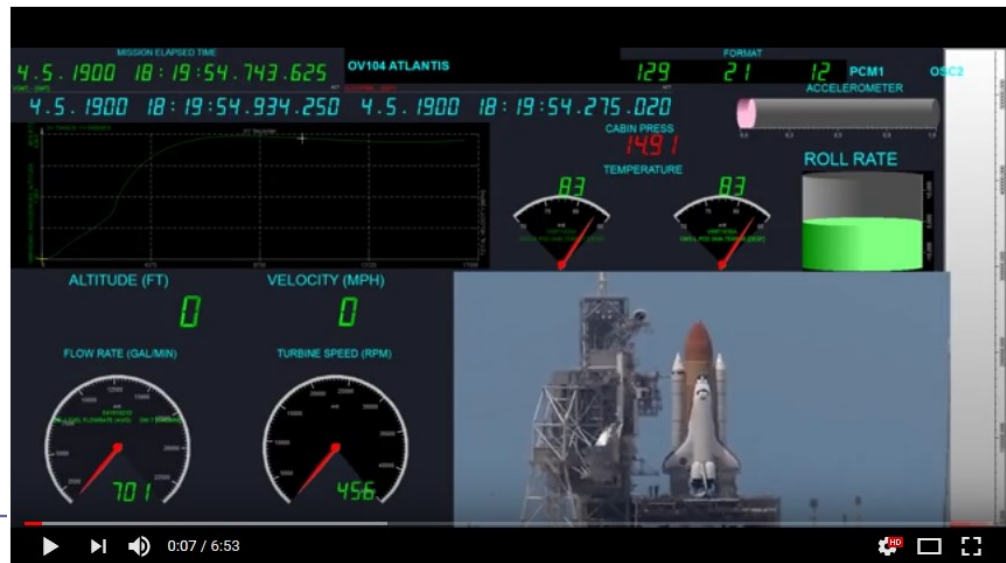
Code optimization on modern processors [Dejan Črnica, Dewesoft]

„in our company developers „speak in assembly...“

Code optimization is important but often overlooked part of a software project. In this talk we will dive deep and discuss when and why to optimize code, how to approach optimization and how to design data structures and algorithms for scalable performance.

*„by knowing the hardware and assembly we can speedup the code by **64x** !!!...“*

Dejan Črnica Dejan Črnica is **lead software engineer at Dewesoft** (<https://www.dewesoft.com/careers>) since 2001. He has designed and implemented core modules of Dewesoft application with particular focus on application performance to keep software in front of competition.

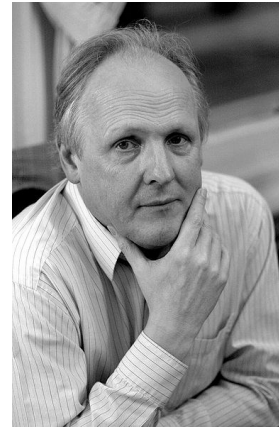
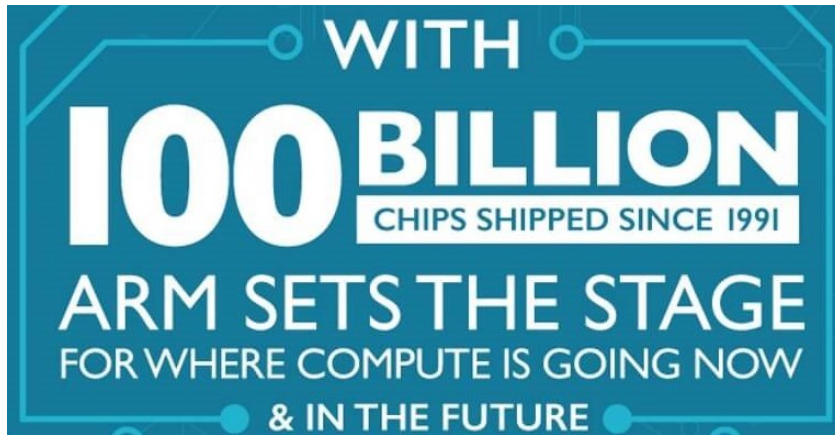


4. Why the ARM architecture?

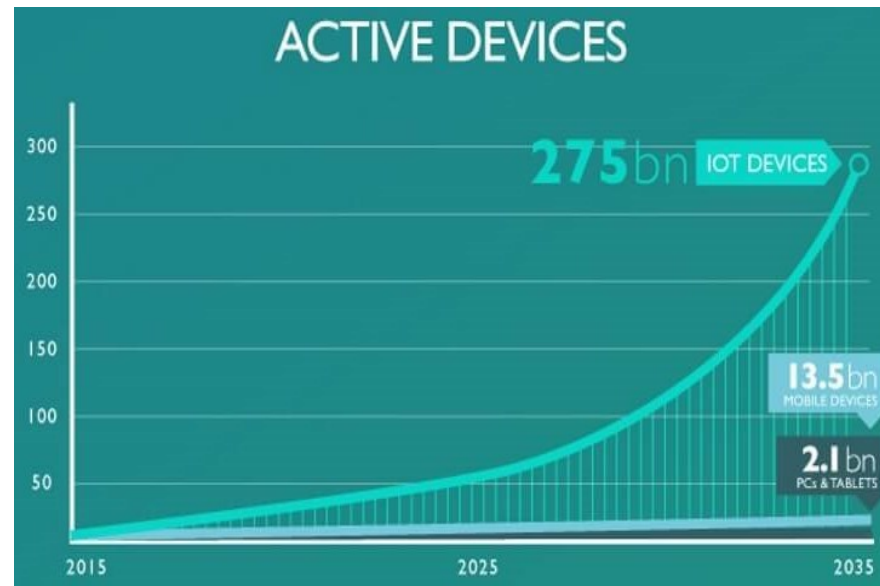
Because ??? ...“

4. Why the ARM architecture?

„Steve Furber on FRI“



principal designer of the BBC Micro and the ARM 32-bit RISC microprocessor.^[15]



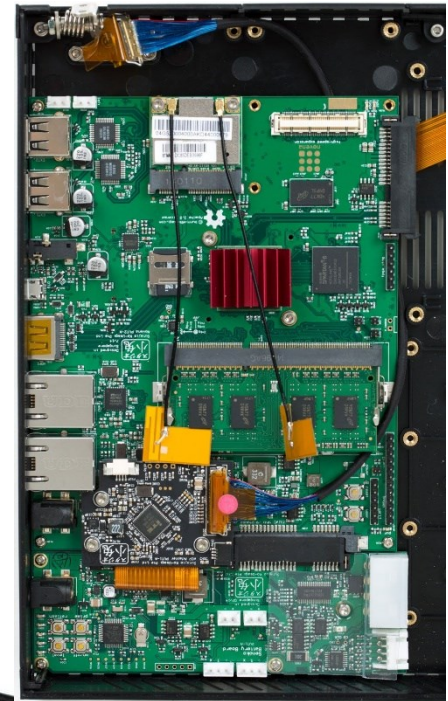
<https://community.arm.com/processors/b/blog/posts/inside-the-numbers-100-billion-arm-based-chips-1345571105>

5. „Black box“ even to experienced engineers?

H AS THE COMPUTER BECOME A BLACK BOX, EVEN TO EXPERIENCED ELECTRICAL ENGINEERS?

Will we be forever reliant upon large, opaque organizations to build them for us? Absolutely not, we say. And to prove our point, we built our very own laptop, from the circuit boards on up.

Admittedly, we did not delude ourselves that we could build a laptop that would be faster, smaller, or cheaper than those of Apple, Dell, or HP. However, we did set out to build a machine powerful and convenient enough to use every day. Fortunately, our dream inspired enough people to crowdfund the effort. Our laptop, which we call Novena, started shipping to backers in January 2015.



*RA – We say
No !!!*

<https://spectrum.ieee.org/novena-a-laptop-with-no-secrets>

1. Introduction :

- ❑ 1.1 CA course

- ❑ 1.2 About computers

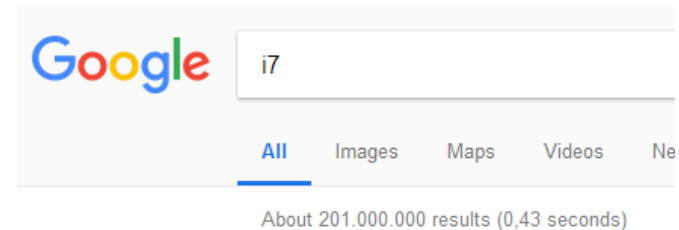
- ❑ 1.3 Basic computer structure and operation

- ❑ 1.4 Analog – digital, continuous-discrete

- ❑ 1.5 8 important ideas in computer architecture (and in general)

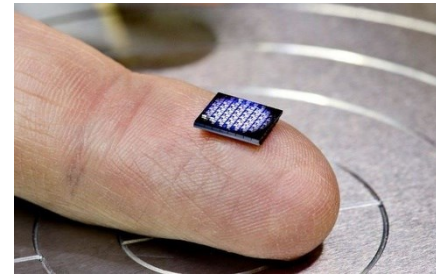
- ❑ 1.6 Computer realization

- Development and use of computers: IT revolution (third revolution in our civilization)
- Extremely rapid evolution over the past 25 years
- Applications that were until recently „impossible“, suddenly became common:
 - Computers in automobiles (autonomous drive)
 - Mobile telephony
 - DNA analysis (The Human Genome Project)
 - The World Wide Web
 - Search engines (Google: i7 \Rightarrow \approx 200.000.000 results in few tenths of a second)
 - AI Chatbots, Assistants (ChatGPT, generative AI, ...)



- Huge difference in computer implementation:

- Supercomputers
- Simple computers on a chip



Pros ?
Cons ?

- Smaller differences in structure
- With every computer, even the simplest, we can calculate everything that can be calculated (is calculable).

- Most powerful computer in the world (in June 2021, second today):
 - SUPERCOMPUTER FUGAKU in Kobe, Japan
 - 7 630 848 cores
 - Performance 537 212 TFLOPS
 - Power consumption 29 899 kW (*Hydro PowerPlant Medvode 26 700 kW*)



<https://www.top500.org/lists/top500/2021/06/>
<https://www.top500.org/lists/top500/2023/06/>

<https://www.r-ccs.riken.jp/en/fugaku/3d-models/>

Potreba po superračunalniških kapacitetah

- Molekulske simulacije: primer klasične molekulske dinamike (MD)
- velikost simuliranega sistema: > 100 000 atomov
- vsak atom ima x,y,z koordinate in V_x , V_y , V_z hitrosti → 600 000 parametrov
- sisteme simuliramo ~1 mikrosekundo s časovnim korakom 2 femtosekunde: 500 000 000 korakov

- 600 000 parametrov * 500 000 000 korakov = **300 tisoč milijard** izračunov



superračunalnik

~ 500 x



domači računalnik



Arnes



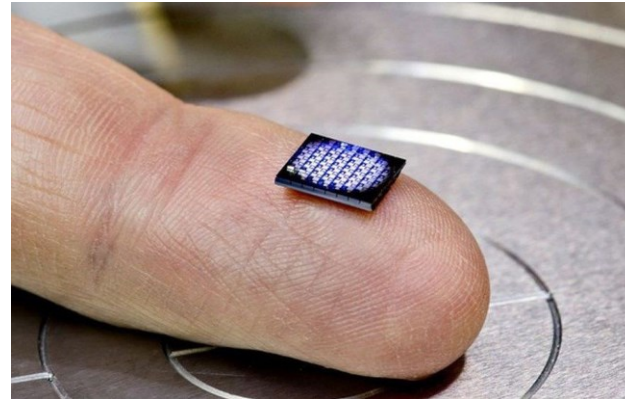
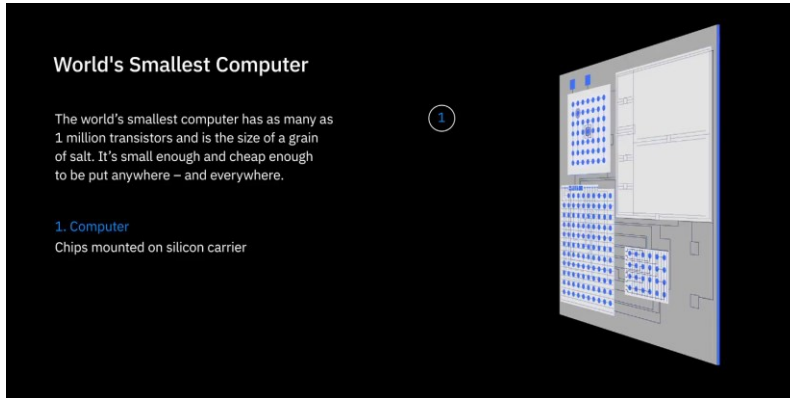
arnes

Slovenian National Supercomputing Network

Slovenian National Supercomputing Network (SLING) is a consortium for the development of grid computing and management of supercomputing infrastructures in Slovenia.

<https://www.sling.si/en/>

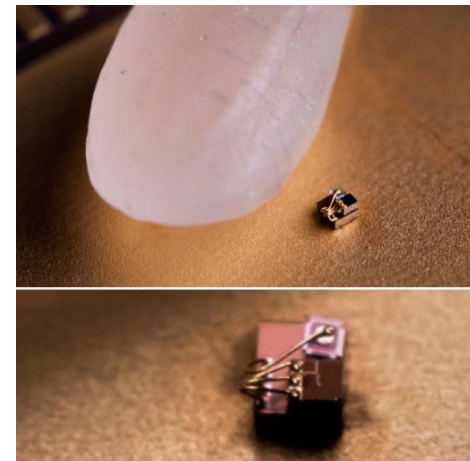
■ Currently the most miniature computer in the world (year 2018): ?



<https://www.research.ibm.com/5-in-5/crypto-anchors-and-blockchain/>

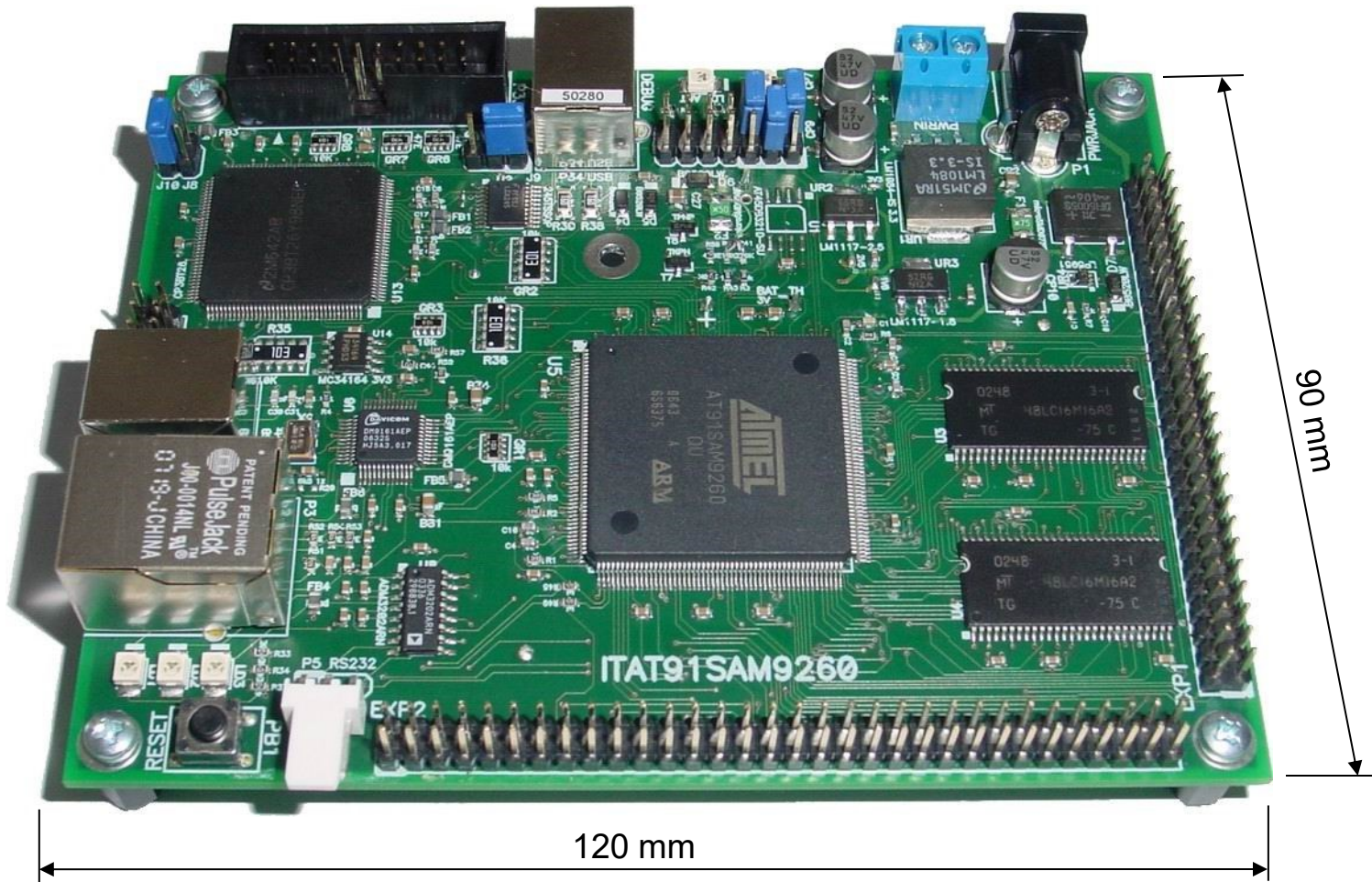
The university on Thursday said its engineers have produced a computer that's **0.3 mm x 0.3 mm** -- it would be dwarfed by a grain of rice. While it drew comparisons to IBM's own 1mm x 1mm computer, Michigan's team said the creation is about more than just size.

- Pros: Low power consumption
- Cons: Low performance



<https://news.umich.edu/u-m-researchers-create-worlds-smallest-computer/>

- FRI-SMS computer (somewhere in-between embedded and HPC)
 - Microcontroller AT91SAM9260 of the ARM9 microcontroller family



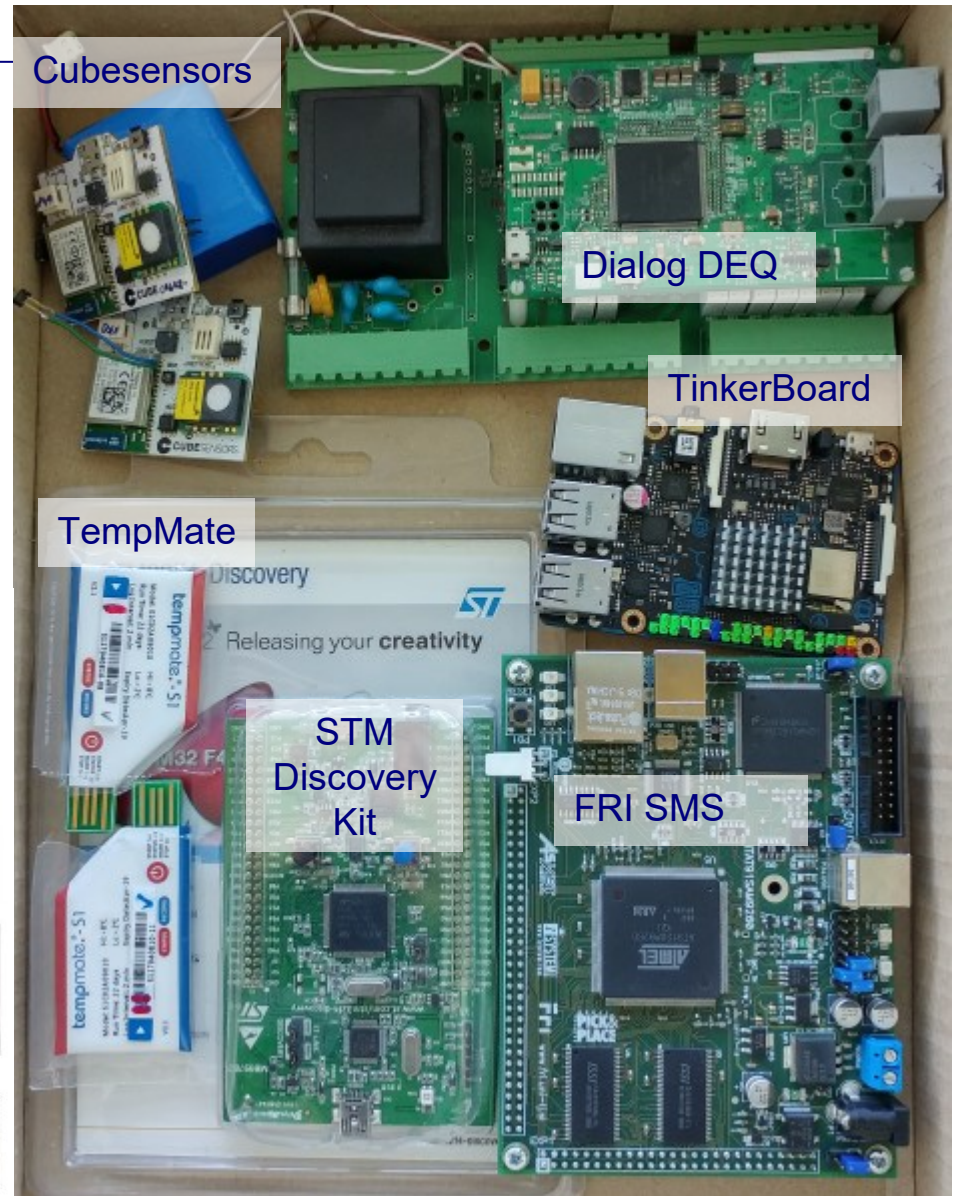
- Nowadays, computers can be attributed into three functional categories:
 - Personal Computers (laptop, tablet, . . .)
 - Servers
 - There are significant differences between servers in price and performance
 - A bit more powerful desktop computers on the low-end
 - Supercomputers with terabytes of main memory and petabytes of external storage on the high-end
 - Embedded computers
 - The most numerous group of computers
 - Microprocessors (or microcontrollers) in automobiles, mobile phones, gaming consoles, household appliances, audio and video equipment, ...

Introduction

Embedded computers (practical examples)

All systems (in right picture) are based on the ARM architecture.

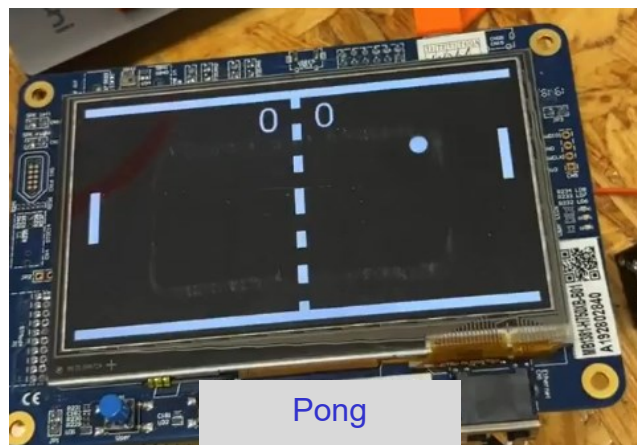
Examples of Embedded Systems



Students' projects on STM32H750 board



Pacman



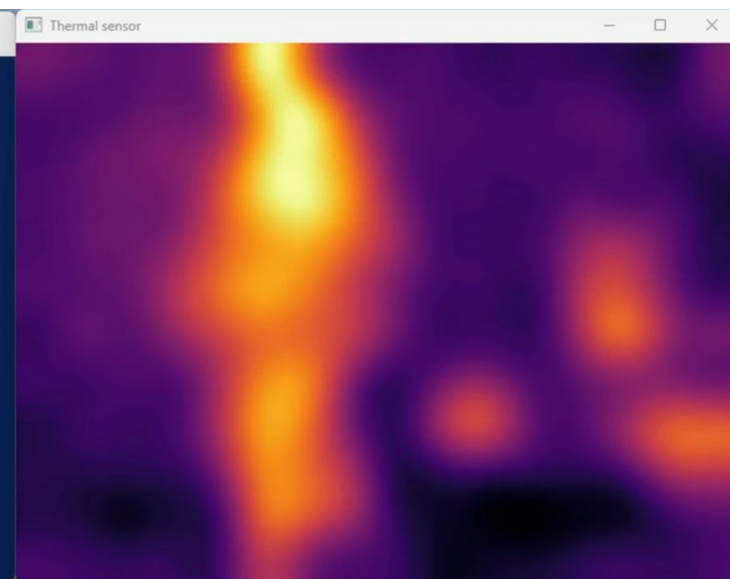
Pong



Termo kamera

```
Windows PowerShell
PS C:\Users\Jakob\git\LIR2> python3 .\classifier.py models/convolutional_model.pt -s COM4 -d
Person present: True
```

Termo kamera – detekcija človeka



1. Introduction :

- ❑ 1.1 CA course
- ❑ 1.2 About computers
- ❑ 1.3 Basic computer structure and operation
- ❑ 1.4 Analog – digital, continuous-discrete
- ❑ 1.5 8 important ideas in computer architecture (and in general)
- ❑ 1.6 Computer realization

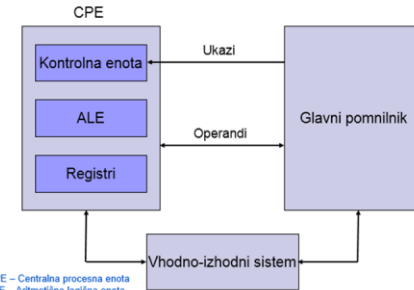
Def: Computer architecture is

- consideration of the programmer's visible computer properties independently of its logical and physical realization [Kodek]
 - „... what programmers see on the assembly language level ...“

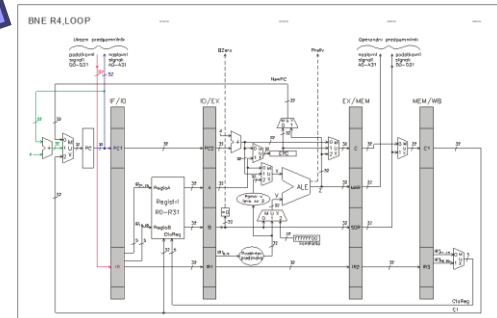
Def: Computer organization (also microarchitecture) :

- explores the logical structure and properties of the computer components and their interconnections [Kodek]
 - „ ... is the architecture of individual components ...“
 - „ ... is closer to the Hardware (HW) level ...“

Zgradba tipičnega računalnika



CPE – Centralna procesna enota
ALE – Aritmetično logična enota



One architecture can be realized with different types of organization and vice versa.

Operation of (digital) computers

- Computer architecture is also the structure of computers as seen by the programmer in assembly code.
- Machine language consists of instructions which can be directly executed by the computer. Those instructions are also called machine code instructions.
- Machine instructions are native instructions built into computers. Computers from different manufacturers can generally have different machine code instructions.

Computer „understands“ own machine instructions only !!!

What is the computer doing ?
(How does it work ?)

Executing instructions !

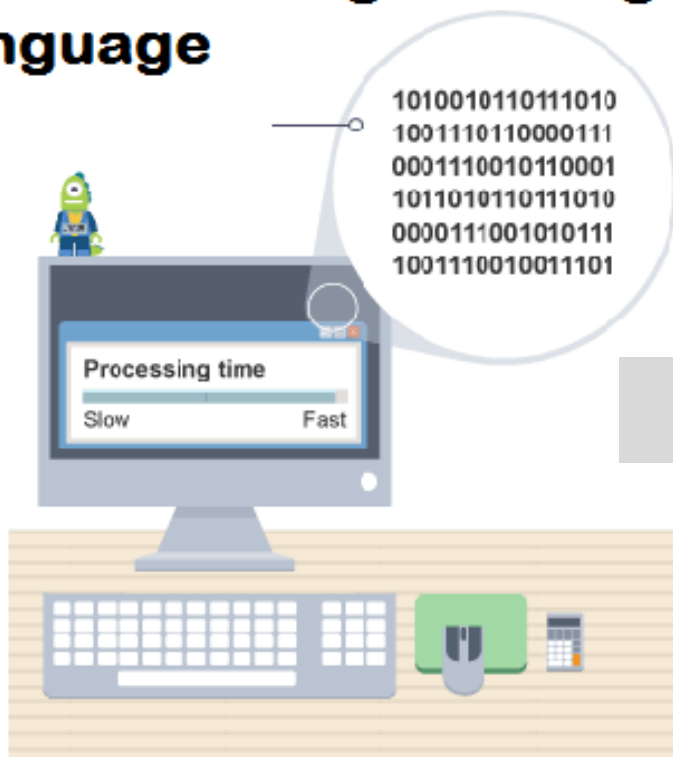
- A digital computer is a machine for solving problems by executing instructions which were set by programmers.
- The sequence of instructions which determines how the machine performs a specific task is called a program.
- The electronic circuit in the computer recognizes and directly executes only a limited set of machine code instructions into which every program has to be translated before the execution.
- Different processors can have different machine code instructions.

- Those basic instructions (machine instructions) are very simple, for example:
 - Addition of two numbers
 - Testing if a number is equal to zero
 - Copy data from one part of the memory to another.
- Any program that is written with some other instructions (e.g. instructions from Java, C++, VisualBasic,...) needs to be changed (translated) into those basic machine code instructions.

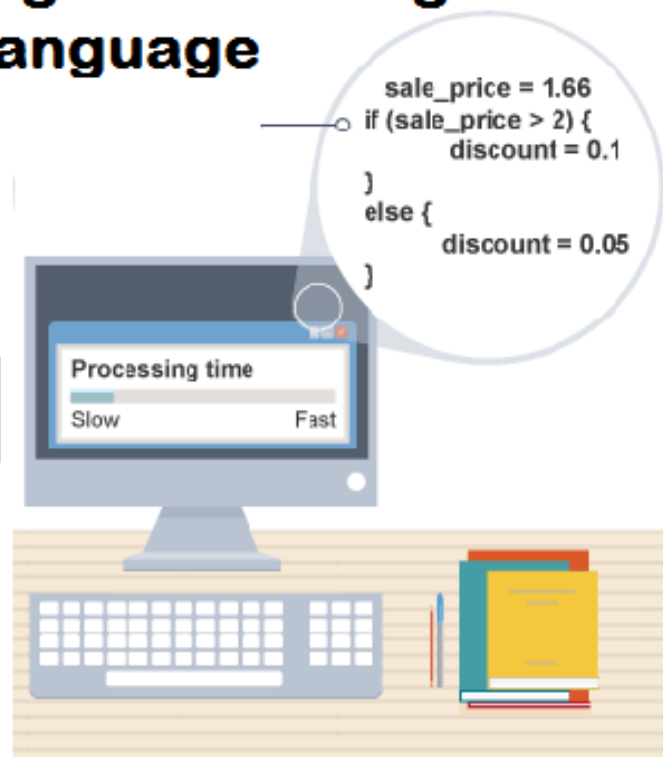
machine language

<-> high-level languages ?

Low Level Programming Language



High Level Programming Language

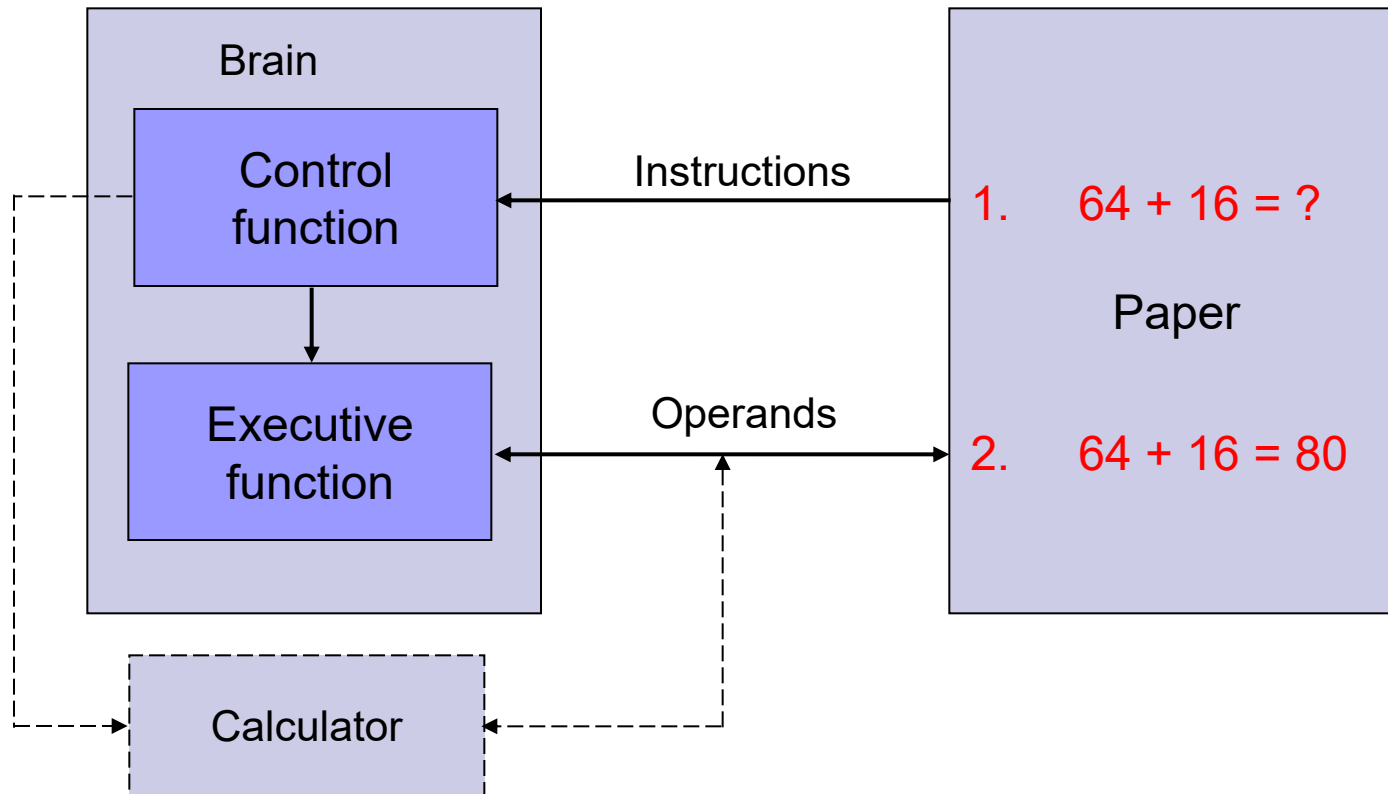


←
Compiler
Interpreter

Portability vs speed ?

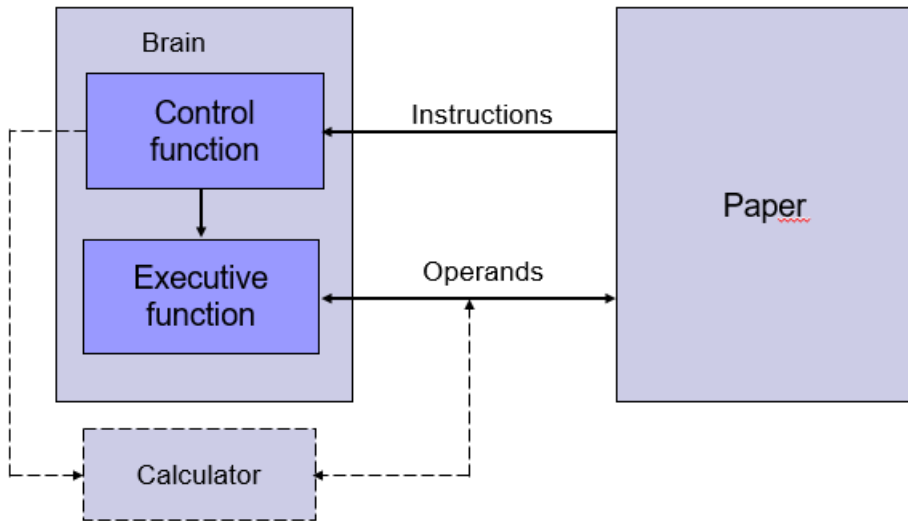
The relation between manual and machine model of computing

Manual computing

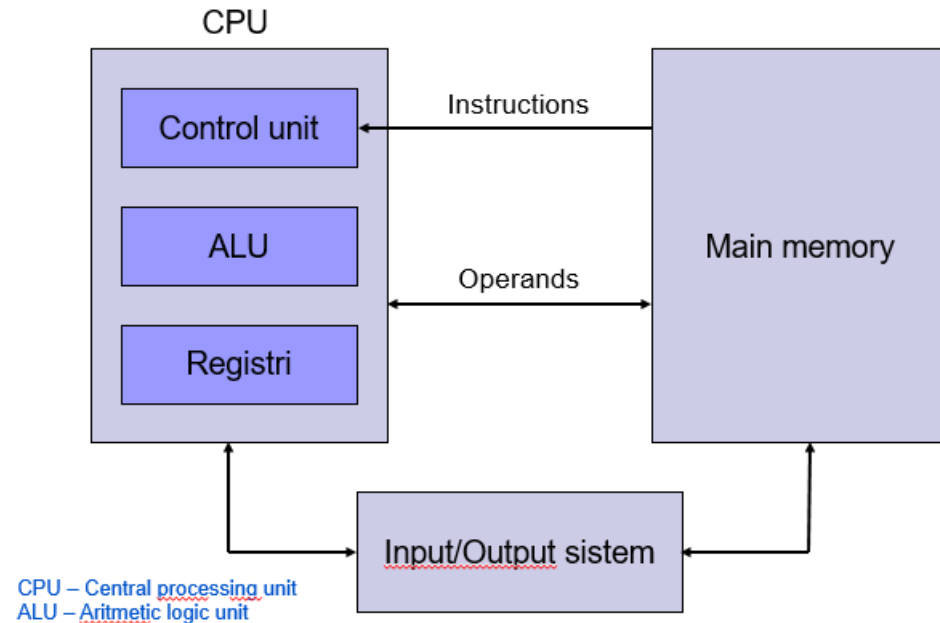


Comparison between models of computing

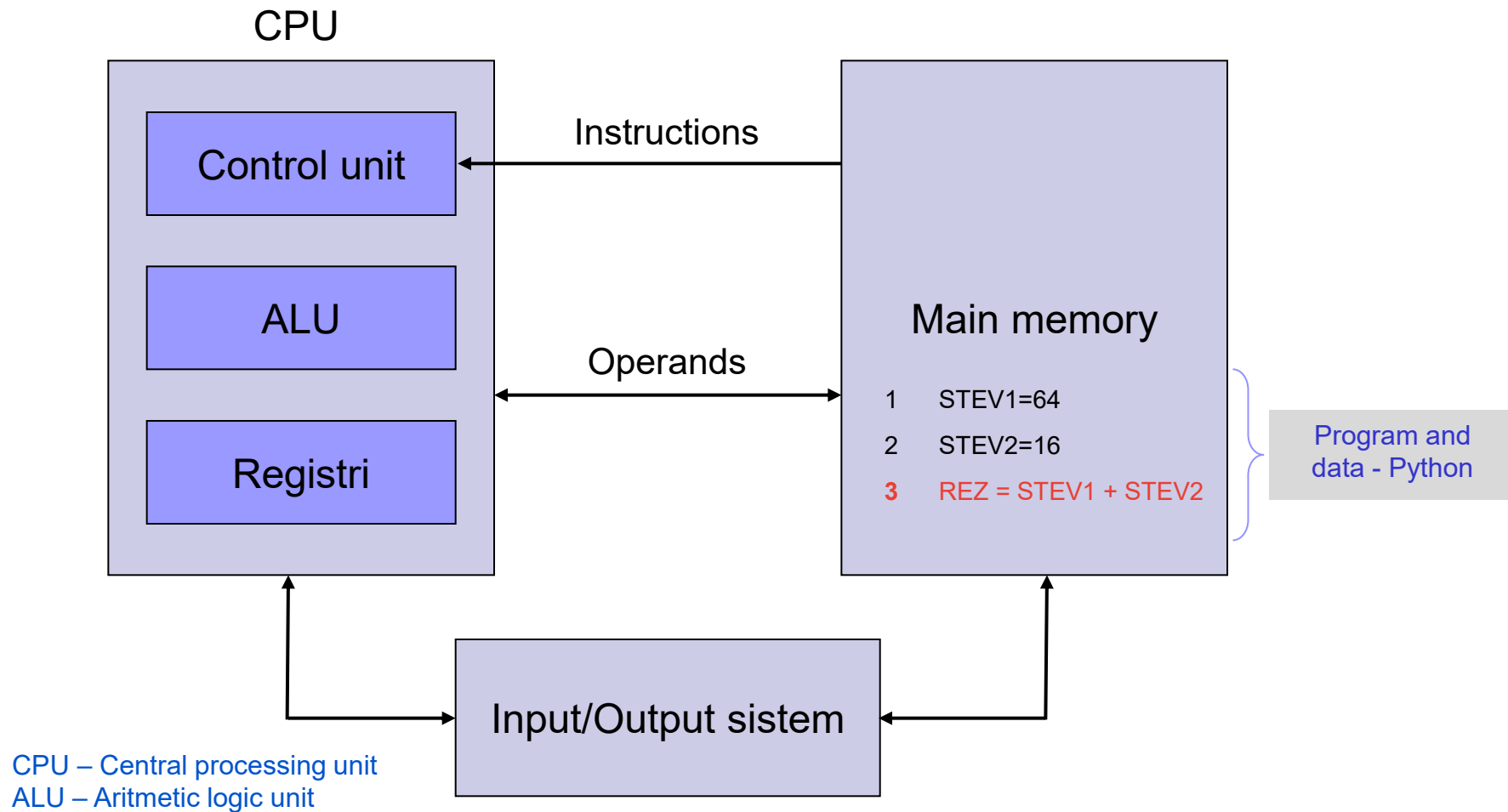
Manual computing



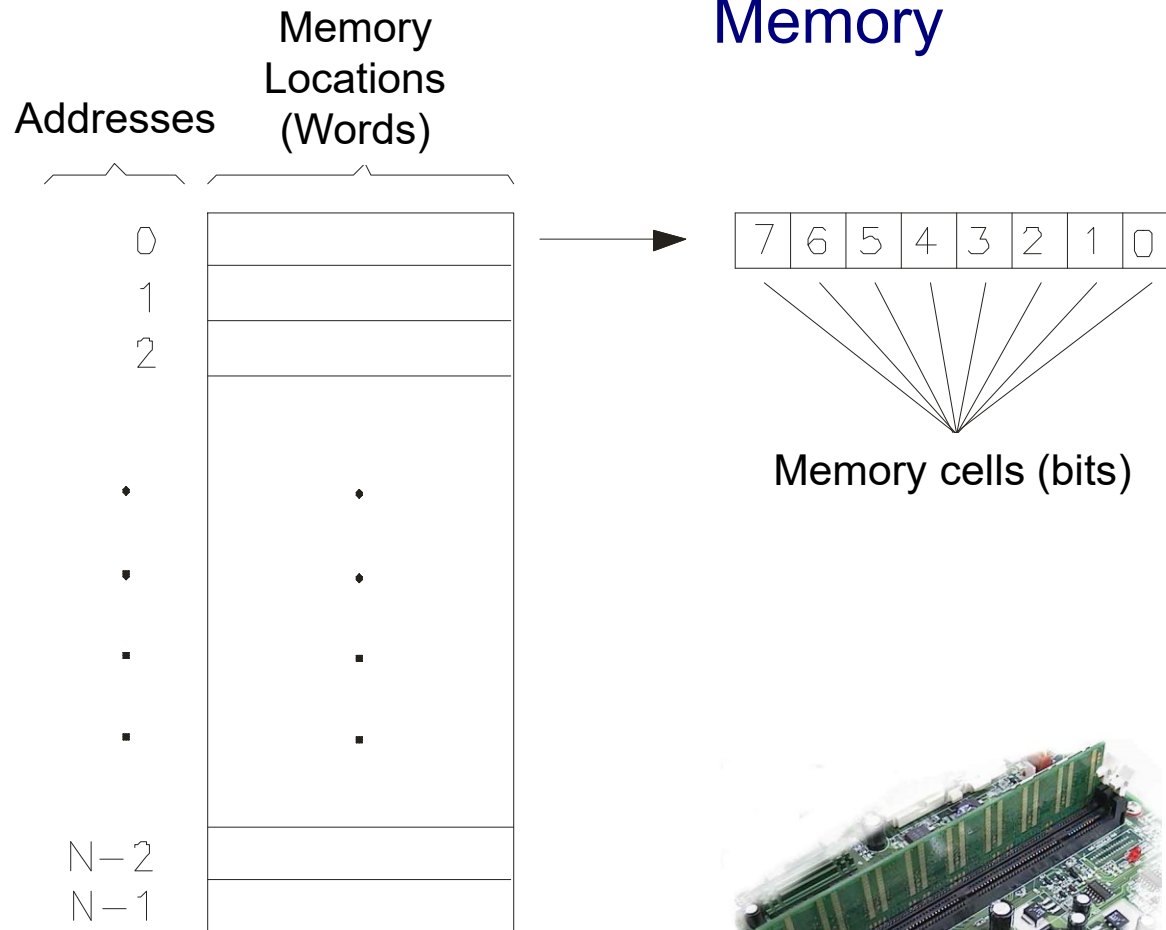
Structure of a typical computer



Structure of a typical computer



Memory



Memory

Logisim EVO Demo

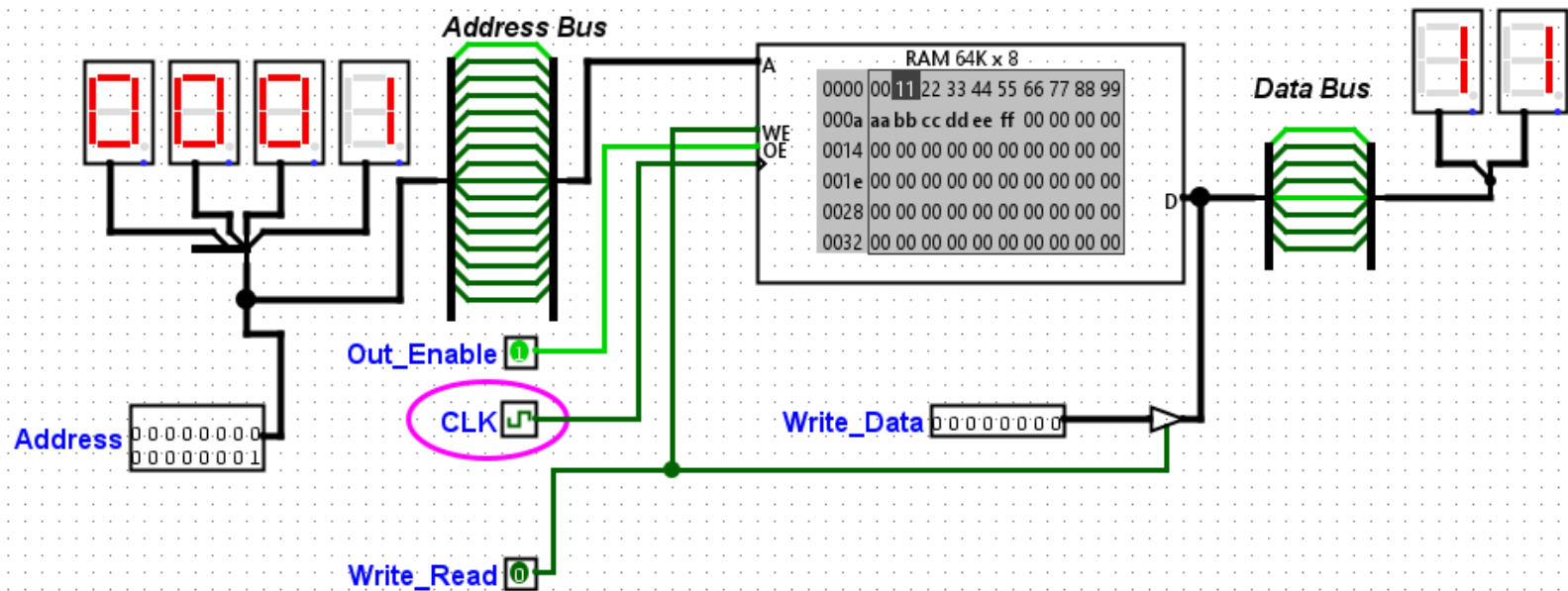
Primer delovanja pomnilnika RAM

Branje :- nastavi naslov (Adress Bus)

- Out_Enable = 1
- Write_Read = 0
- CLK: 1 cikel (dva klika)

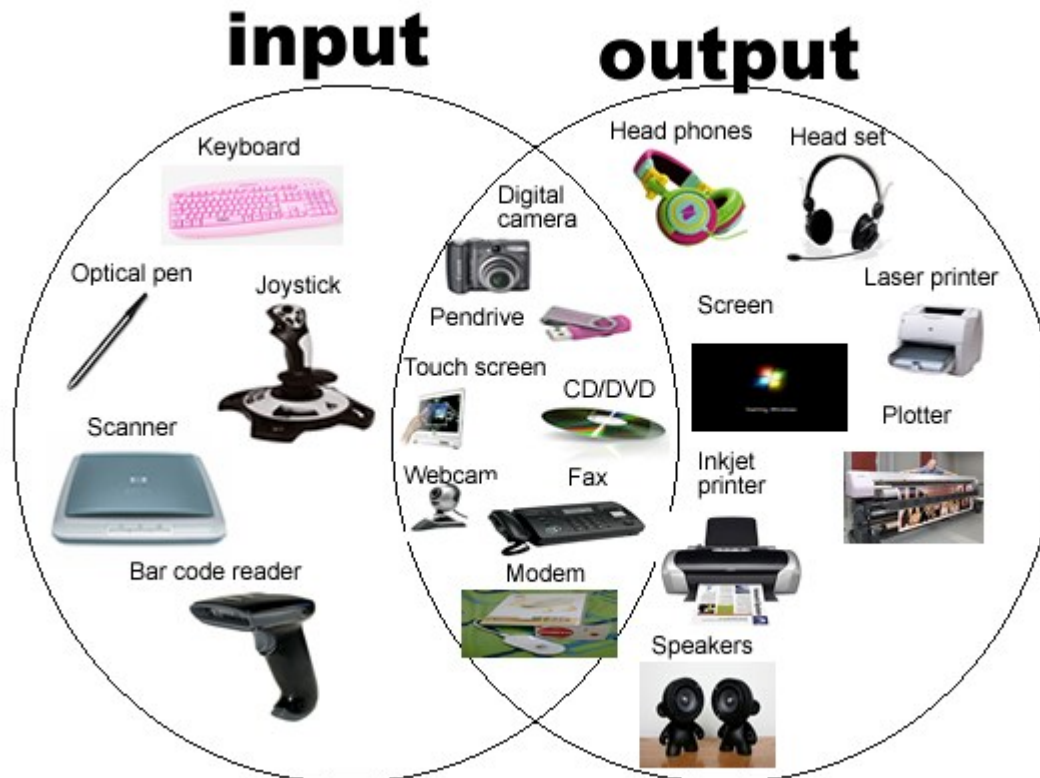
Pisanje :- nastavi naslov (Adress Bus)

- Write_Data = podatek za vpis
- Out_Enable = 0
- Write_Read = 1
- CLK: 1 cikel (dva klika)



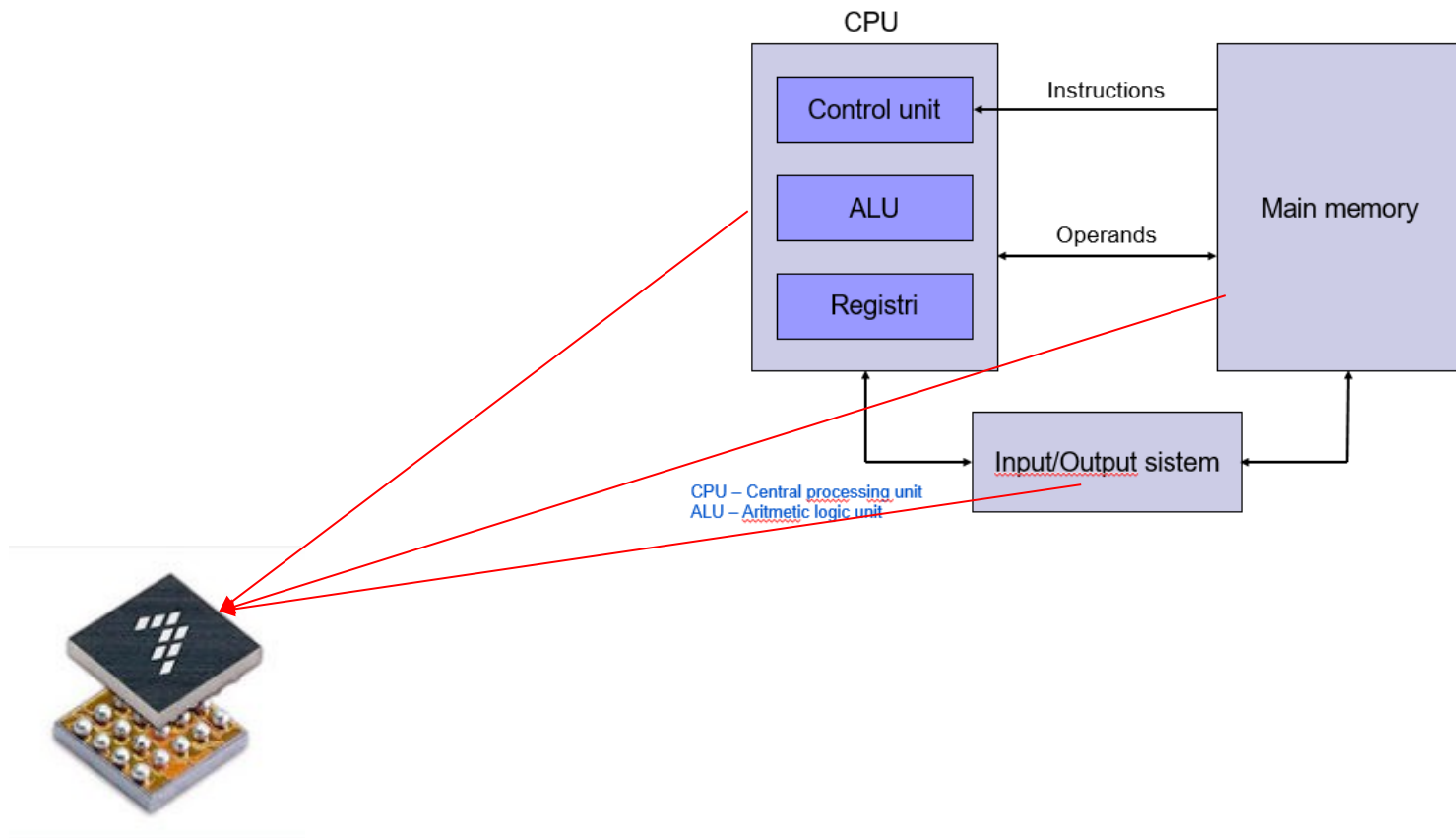
RAM_pomnilnik_demo_EVO.circ

Input-Output system (devices)



Structure of a typical computer

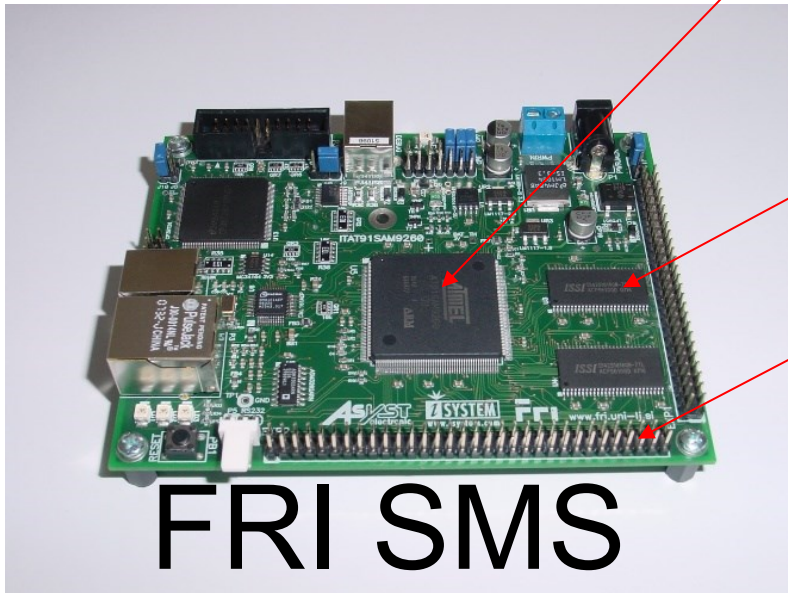
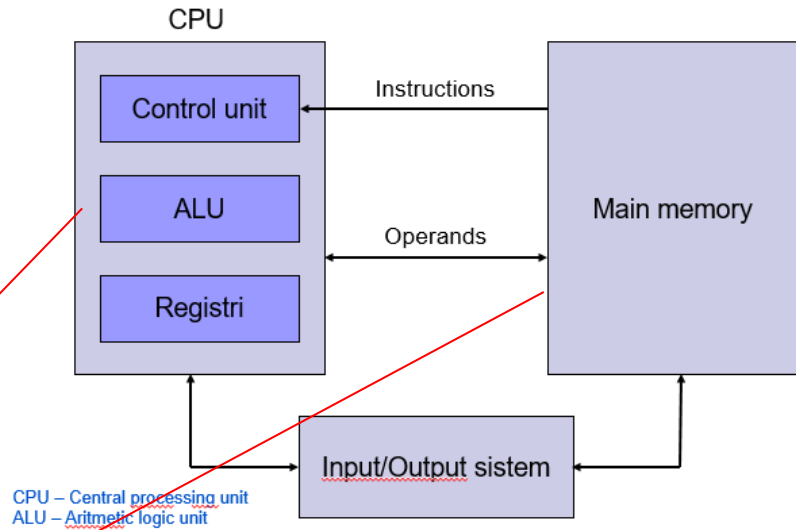
Structure of a typical computer



Microcontroller

Structure of a typical computer

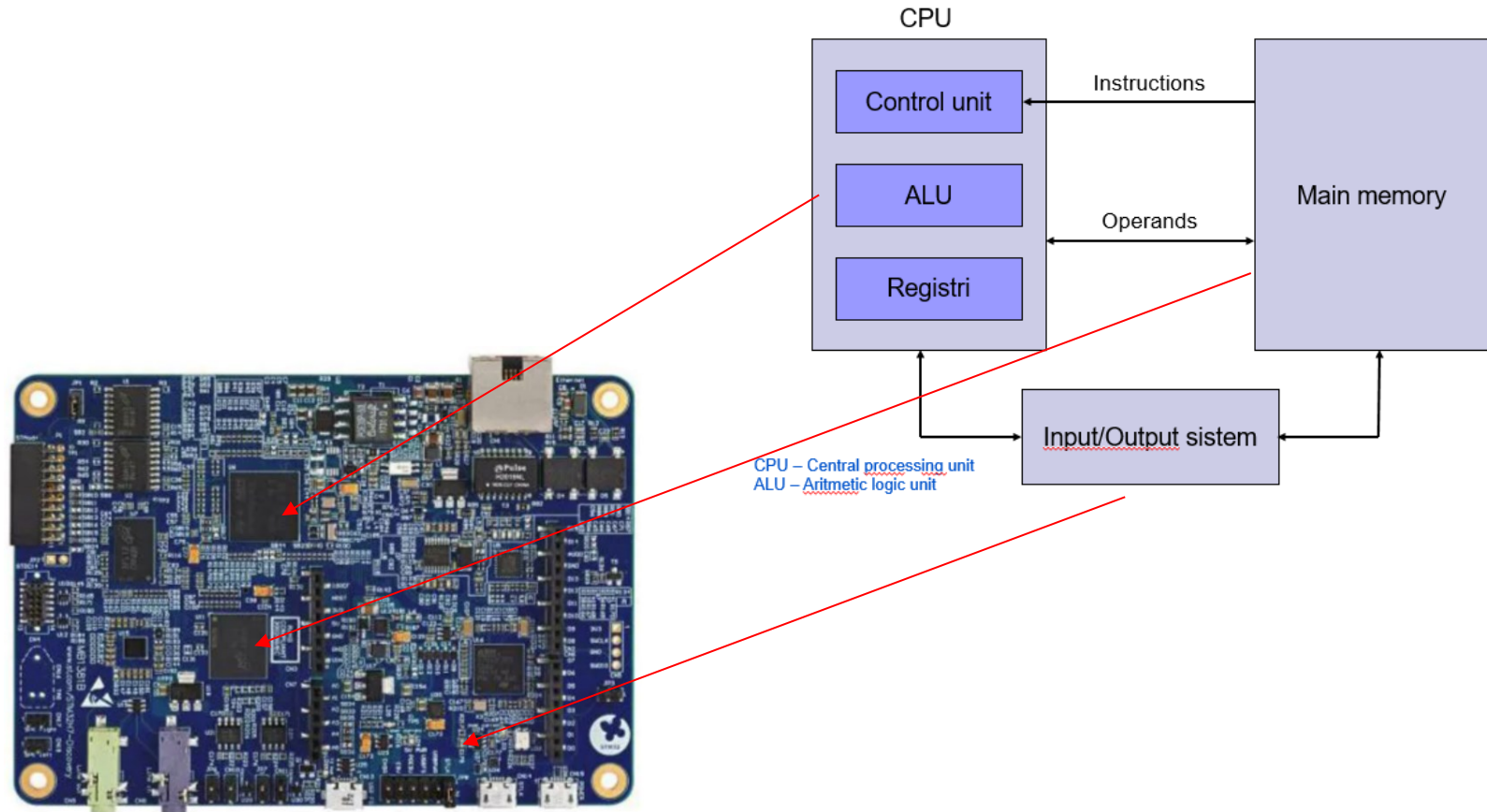
Structure of a typical computer



FRI SMS

Structure of a typical computer

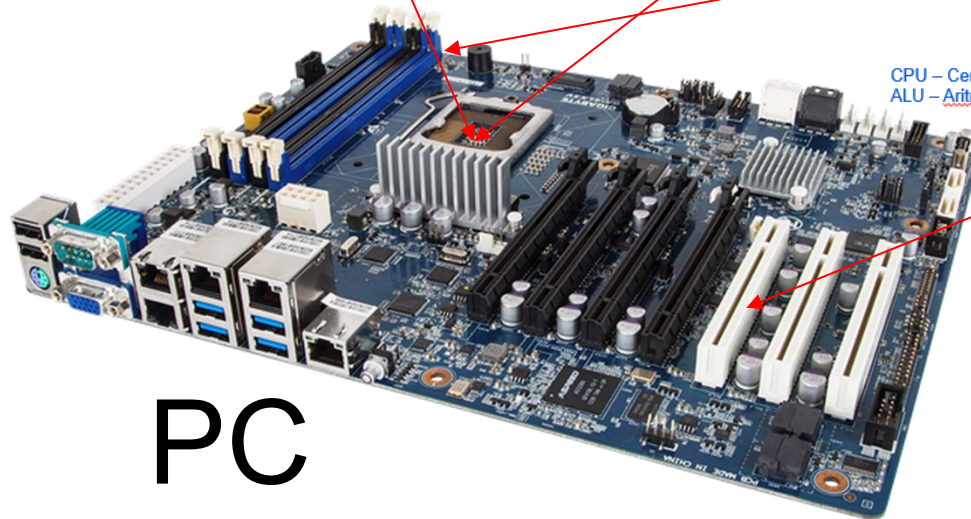
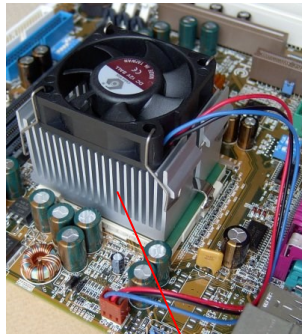
Structure of a typical computer



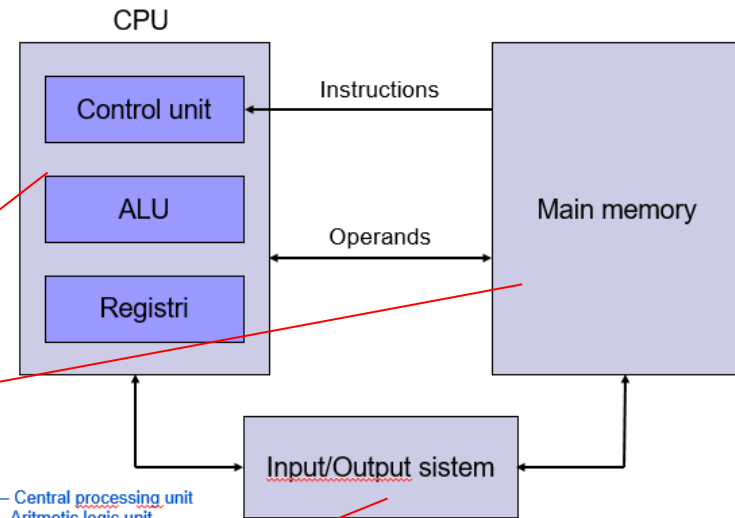
STM32H750-DK

Structure of a typical computer

Structure of a typical computer

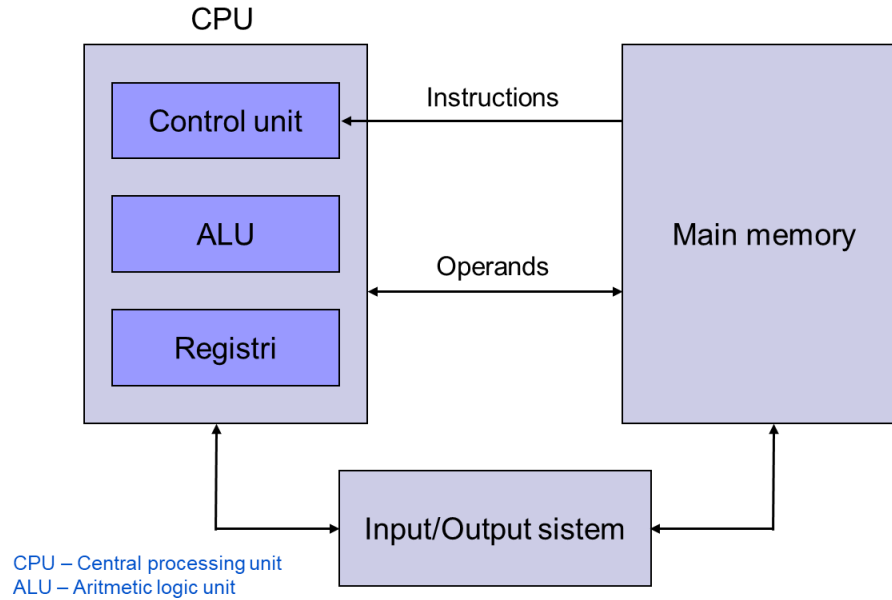


PC



CPU – Central processing unit
ALU – Arithmetic logic unit

Case: Addition of two numbers(1. LAB session)



Zbirni jezik	Opis ukaza	Strojni jezik
<code>adr r0, stev1</code>	$R0 \leftarrow \text{nasl. stev1}$	0xE24F0014
<code>ldr r1, [r0]</code>	$R1 \leftarrow M[R0]$	0xE5901000
<code>adr r0, stev2</code>	$R0 \leftarrow \text{nasl. stev2}$	0xE24F0018
<code>ldr r2, [r0]</code>	$R2 \leftarrow M[R0]$	0xE5902000
<code>add r3, r2, r1</code>	$R3 \leftarrow R1 + R2$	0xE0823001
<code>adr r0, rez</code>	$R0 \leftarrow \text{nasl. rez}$	0xE24F0020
<code>str r3, [r0]</code>	$M[R0] \leftarrow R3$	0xE5803000

Python

- 1 STEV1=64
- 2 STEV2=16
- 3 **REZ = STEV1 + STEV2**

Case: Addition of two numbers(1. LAB session)

Python

Assembler

<http://goo.gl/YXQ5qN>

- 1 STEV1=64
- 2 STEV2=16
- 3 REZ = STEV1 + STEV2



```

STEV1: .word 0x10 // 32-bitna spr.
STEV2: .word 0x40 // 32-bitna spr.
VSOTA: .word 0 // 32-bitna spr.

```

```

adr r0, STEV1 // Naslov od STEV1 -> r0
ldr r1, [r0] // Vsebina iz naslova v r0 -> r1

```

```

adr r0, STEV2 // Naslov od STEV2 -> r0
ldr r2, [r0] // Vsebina iz naslova v r0 -> r2

```

```

add r3,r1,r2 // r1 + r2 -> r3

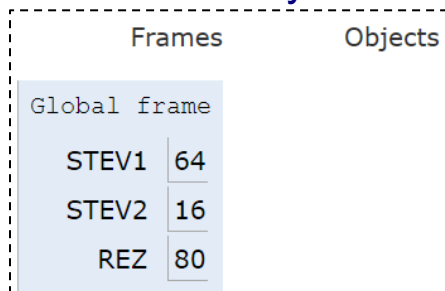
```

```

adr r0,VSOTA // Naslov od VSOTA -> r0
str r3,[r0] // iz registra r3 -> na naslov v r0

```

Variables in
memory



<https://cpulator.01xz.net/?sys=arm&loadasm=share/s9brA0e.s>

Structure of a typical computer and addition of two numbers

LAB -1

Python

- 1 STEV1=64
- 2 STEV2=16
- 3 **REZ = STEV1 + STEV2**



Assembler

```

STEV1: .word 0x10 // 32-bitna spr.
STEV2: .word 0x40 // 32-bitna spr.
VSOTA: .word 0 // 32-bitna spr.

adr r0, STEV1 // Naslov od STEV1 -> r0
ldr r1, [r0] // Vsebina iz naslova v r0 -> r1

adr r0, STEV2 // Naslov od STEV2 -> r0
ldr r2, [r0] // Vsebina iz naslova v r0 -> r2

add r3,r1,r2 // r1 + r2 -> r3

adr r0,VSOTA // Naslov od VSOTA -> r0
str r3,[r0] // iz registra r3 -> na naslov v r0
    
```



Machine language

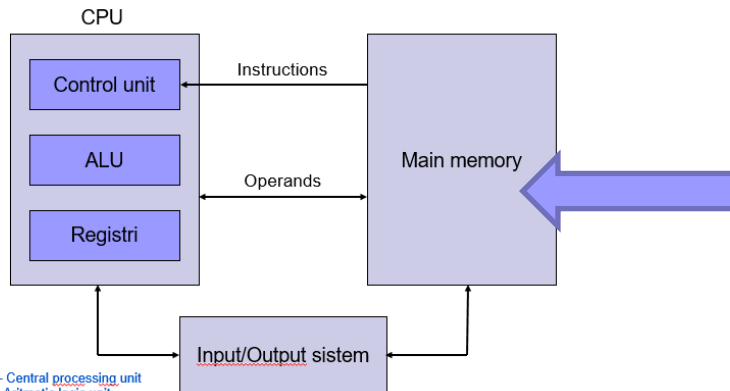
Memory (Ctrl-M)

Go to address, label, or register:

Address	Memory contents and ASCII
00000010	00000000 00000000 00000000 00000000
00000020	00000010 00000040 00000000 e24f0014
00000030	e5901000 e24f0018 e5902000 e0813002
00000040	e24f0020 e5803000 aaaaaaaaa aaaaaaaaa



Structure of a typical computer



CPU – Central processing unit
ALU – Arithmetic logic unit

Python (case: REZ = STEV1 + STEV2)

Adding variables in Python.

<http://goo.gl/YXQ5qN>

Python 2.7

```
1 STEV1=0x40
2 STEV2=0x10
3 REZ = STEV1 + STEV2
→ 4 print (" STEV1 = " + hex(STEV1) + "\n+STEV2 = " + hex(STE
```

Frames

Objects

Print output (drag lower right corner to resize)

Global frame

STEV1	64
-------	----

STEV2	16
-------	----

REZ	80
-----	----

```
STEV1 = 0x40
```

```
+STEV2 = 0x10
```

```
-----
REZ = 0x50
```

Addition of two numbers LAB -1

Assembler

```
STEV1:      .org 0x20
             .word 0x10 // 32-bitna spr.
STEV2:      .word 0x40 // 32-bitna spr.
VSOTA:      .word 0 // 32-bitna spr.

.global _start
_start:

    adr r0, STEV1 // Naslov od STEV1 -> r0
    ldr r1, [r0] // Vsebina iz naslova v r0 -> r1

    adr r0, STEV2 // Naslov od STEV2 -> r0
    ldr r2, [r0] // Vsebina iz naslova v r0 -> r2

    add r3,r1,r2 // r1 + r2 -> r3

    adr r0,VSOTA // Naslov od VSOTA -> r0
    str r3,[r0] // iz registra r3 -> na naslov v r0
end:        b end
```

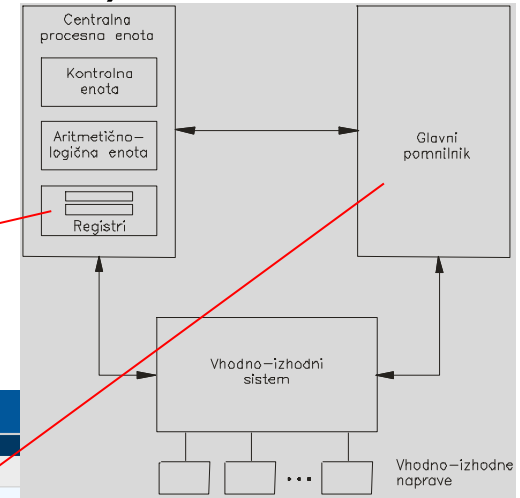
Assembler (case: REZ = STEV1 + STEV2)

CPUlator – online simulator

CPUlator

The screenshot shows the CPUlator simulator interface. On the left, the 'Registers' panel lists registers r0 through r12, sp, lr, pc, cpsr, and spsr. The 'Disassembly (Ctrl-D)' panel shows assembly code with addresses and opcodes. The 'Memory (Ctrl-M)' panel shows memory contents and ASCII values. A red arrow points from the 'Registers' panel to the 'CPUlator' text above.

Address	Opcode	Disassembly
00000000	aaaaaaa	bge 0xfeaaaa98
00000001	aaaaaaa	bge 0xfeaaaa9c
00000002	aaaaaaa	bge 0xfeaaaaa0
00000003	aaaaaaa	bge 0xfeaaaaa4
00000004	aaaaaaa	bge 0xfeaaaaa8
00000005	aaaaaaa	bge 0xfeaaaaac
00000006	andeq r8, r0, r0	
00000007	andeq r8, r0, r0	
00000008	andeq r8, r0, r0	
00000009	andeq r8, r0, r0	
0000000a	andeq r8, r0, r0	
0000000b	andeq r8, r0, r0	
0000000c	andeq r8, r0, r0	
0000000d	andeq r8, r0, r0	
0000000e	andeq r8, r0, r0	
0000000f	andeq r8, r0, r0	
00000010	andeq r8, r0, r0	
00000011	andeq r8, r0, r0	
00000012	andeq r8, r0, r0	
00000013	andeq r8, r0, r0	
00000014	andeq r8, r0, r0	
00000015	andeq r8, r0, r0	
00000016	andeq r8, r0, r0	
00000017	andeq r8, r0, r0	
00000018	andeq r8, r0, r0	
00000019	andeq r8, r0, r0	
0000001a	andeq r8, r0, r0	
0000001b	andeq r8, r0, r0	
0000001c	andeq r8, r0, r0	
0000001d	andeq r8, r0, r0	
0000001e	andeq r8, r0, r0	
0000001f	andeq r8, r0, r0	
00000020	andeq r8, r0, r0	
00000021	andeq r8, r0, r0	
00000022	andeq r8, r0, r0	
00000023	andeq r8, r0, r0	
00000024	andeq r8, r0, r0	
00000025	andeq r8, r0, r0	
00000026	andeq r8, r0, r0	



The 'Memory (Ctrl-M)' panel shows a search bar with the address '20' entered. Below is a table of memory contents and ASCII values.

Address	Memory contents and ASCII
00000010	00000000 00000000 00000000 00000000
00000020	00000010 00000040 00000000 e24f0014
00000030	e5901000 e24f0018 e5902000 e0813002
00000040	e24f0020 e5803000 aaaaaaaaaa aaaaaaaaaa

1. Introduction :

- ❑ 1.1 CA course
- ❑ 1.2 About computers
- ❑ 1.3 Basic computer structure and operation
- ❑ 1.4 Analog – digital, continuous-discrete
- ❑ 1.5 8 important ideas in computer architecture (and in general)
- ❑ 1.6 Computer realization

Comparisons:

Analog – digital

Continuous – discrete

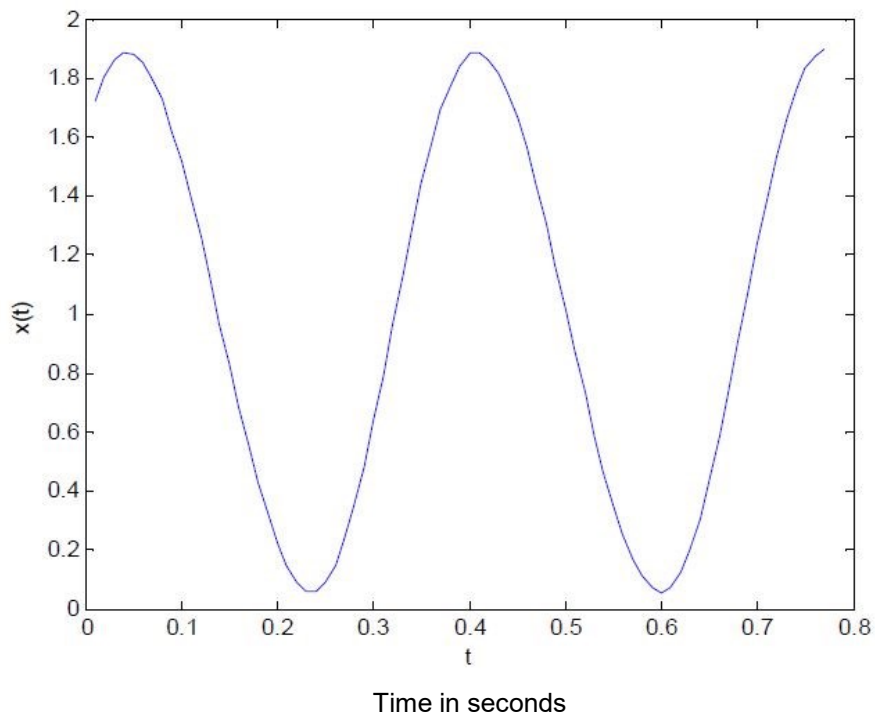
Analog –
- continuous representation



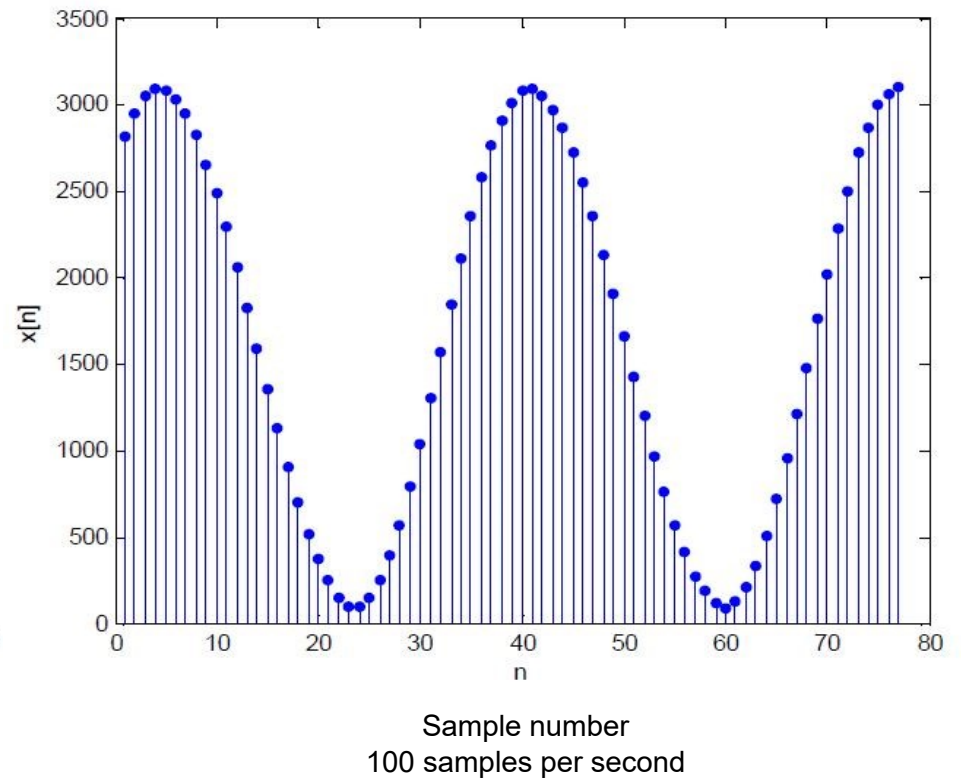
Digital –
- discrete representation



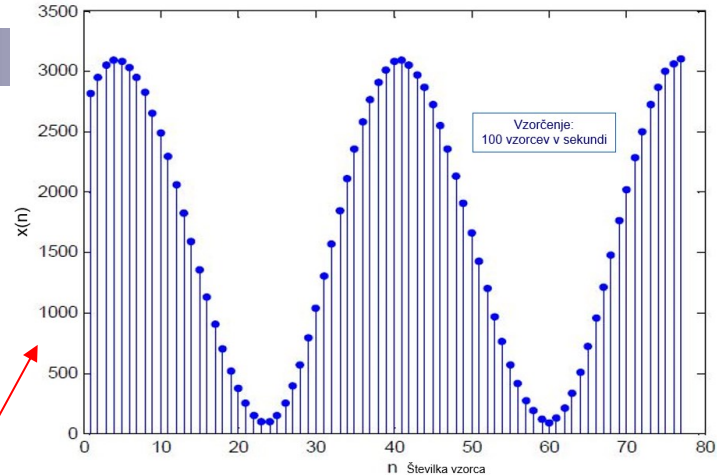
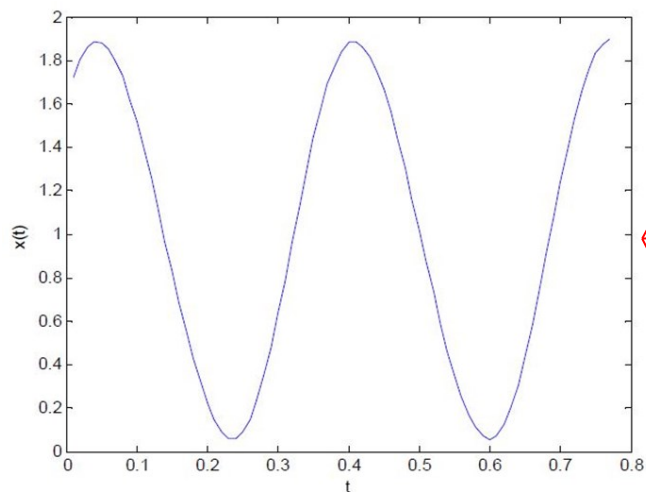
Analog – - continuous representation



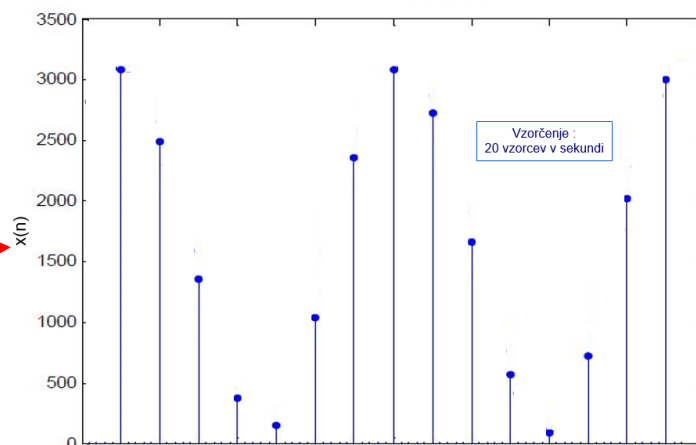
Digital – - discrete representation



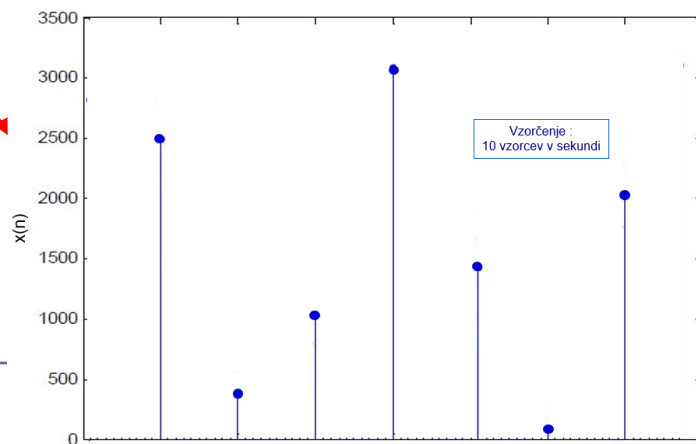
Case 2: Signal sampling



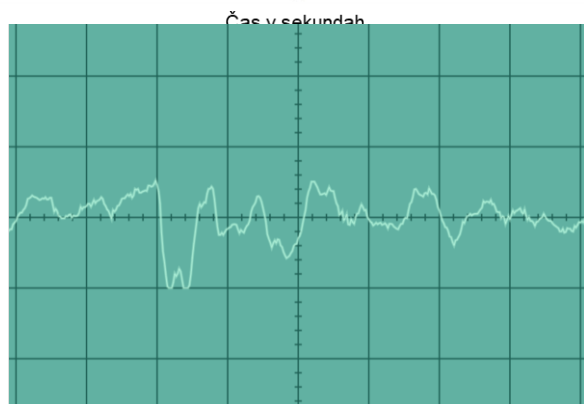
100 samples/sec



20 samples/sec



10 samples/sec

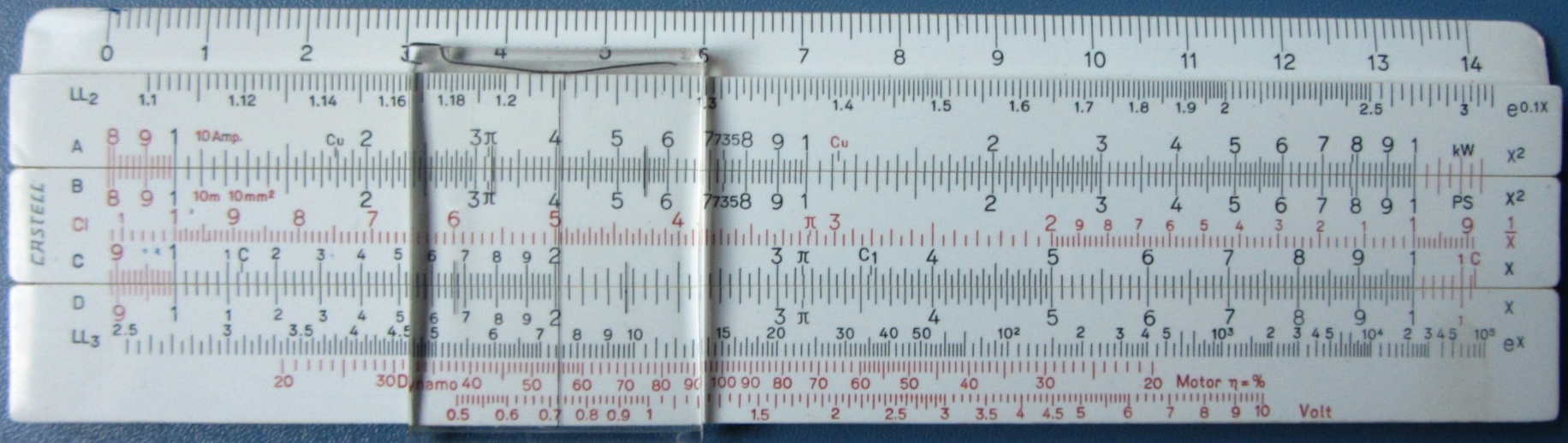




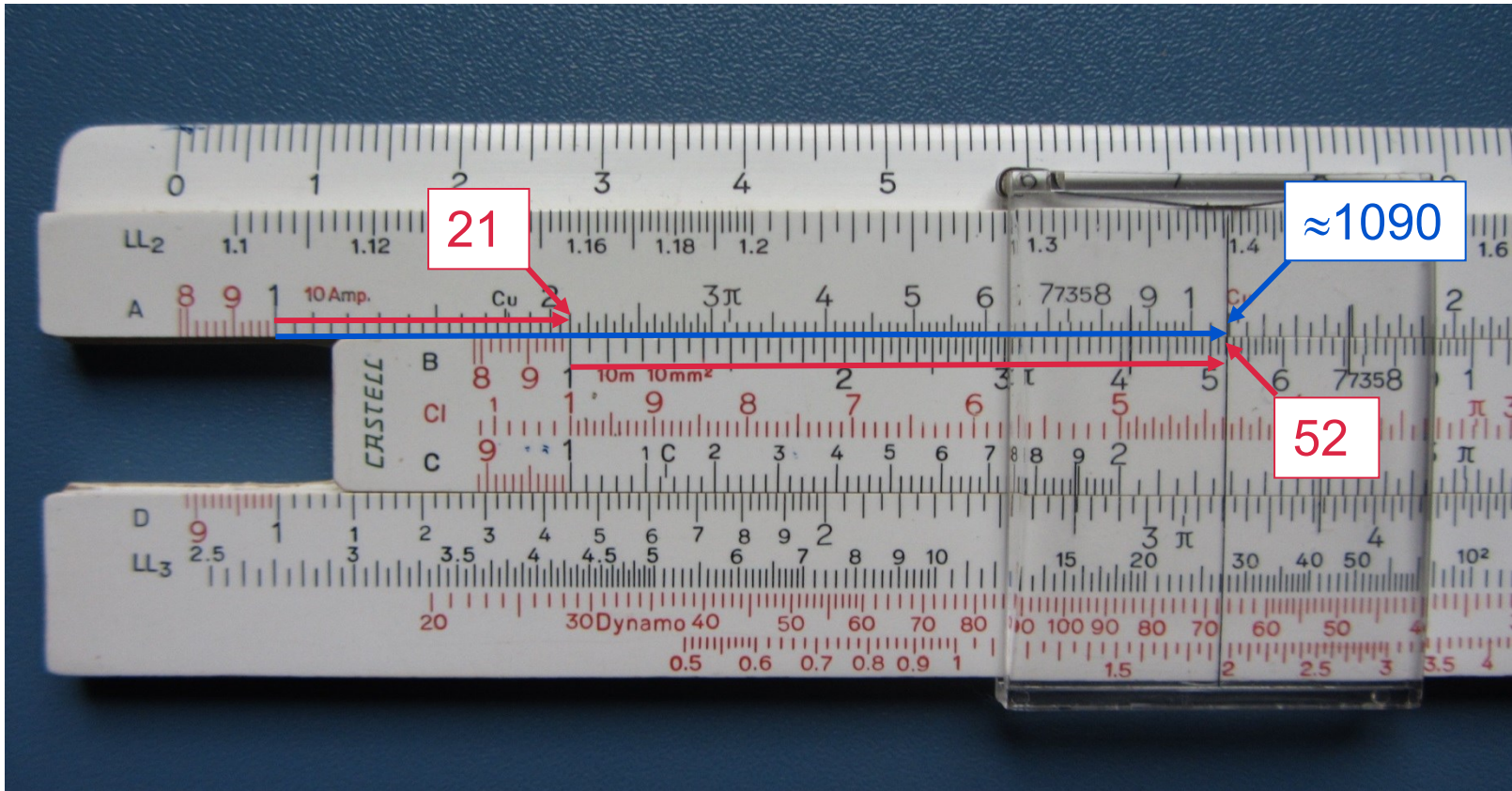
Analog computing – continuous presentation of numbers

Digital computing – discrete presentation of numbers

- Analog computing is carried out by representing numbers with some other physical quantity:
 - With a distance \Rightarrow Logarithmic calculator
 - Idea: $\log_{10}(a \cdot b) = \log_{10} a + \log_{10} b$



- Example of multiplication of 21 x 52 with the logarithmic calculator:



$$21 \times 52 \approx 1090$$

Measured result

$$21 \times 52 = 1092$$

Exact result

Introduction

- using continuous Voltage \Rightarrow Analog amplifier and computer





- Discrete computing with beads
- With digits from 0 to 9

Digital computing

- With digits 0 and 1

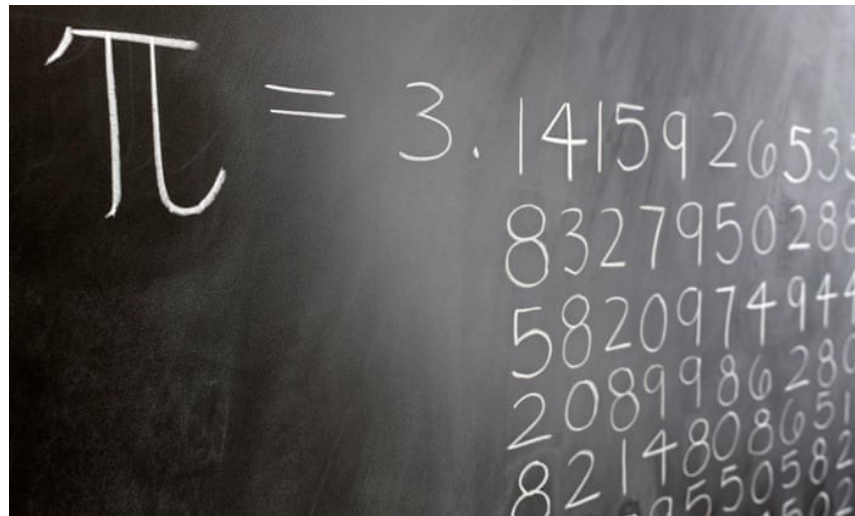


- Binary numeral system:

- base number is 2
- digits 0 and 1

- binary digit = bit
- Bit = one of the two digits (0 or 1) of the binary numeral system
- Digital computer is built on top of a binary numeral system

- Numbers are represented with finite number of bits
- How can you represent/calculate with value of π in computer ?



1. Introduction :

- ❑ 1.1 CA course
- ❑ 1.2 About computers
- ❑ 1.3 Basic computer structure and operation
- ❑ 1.4 Analog – digital, continuous-discrete
- ❑ 1.5 8 important ideas in computer architecture (and in general)
- ❑ 1.6 Computer realization

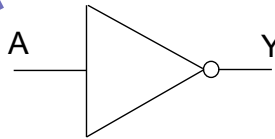
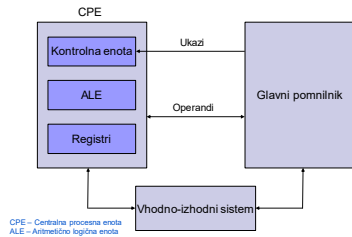
8 important ideas in computer architecture (and broader) [Patt]

- Moore's law
 - Number of transistors in integrated circuits double every 18-24 months
- Abstraction as simplification
 - Design of hardware and software, programming languages, subprograms, ...
- Speed up common procedures
 - It's most profitable to speed up the most common used procedures
- More performance with parallelism
 - Considering the current technology evolution: it's the only way
- Performance with pipelines
 - Effective, transparent way to speed up the CPU
- Performance with speculations
 - „Better work according to some speculation than just do nothing - wait“
- Memory hierarchy
 - Compromise between memory speed and cost
- Reliability with redundancy
 - Cost of the backup system may be lower than the cost of failure

1. Introduction :

- ❑ 1.1 CA course
- ❑ 1.2 About computers
- ❑ 1.3 Basic computer structure and operation
- ❑ 1.4 Analog – digital, continuous-discrete
- ❑ 1.5 8 important ideas in computer architecture (and in general)
- ❑ 1.6 Computer realization

Theoretical model <-> Practical realization



Simbol

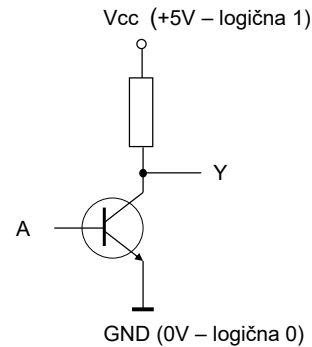
mathematical (logical) view: logic gate

Mathematical ideal
logical level 0,1

electrical realization : electrical circuit

Cons :

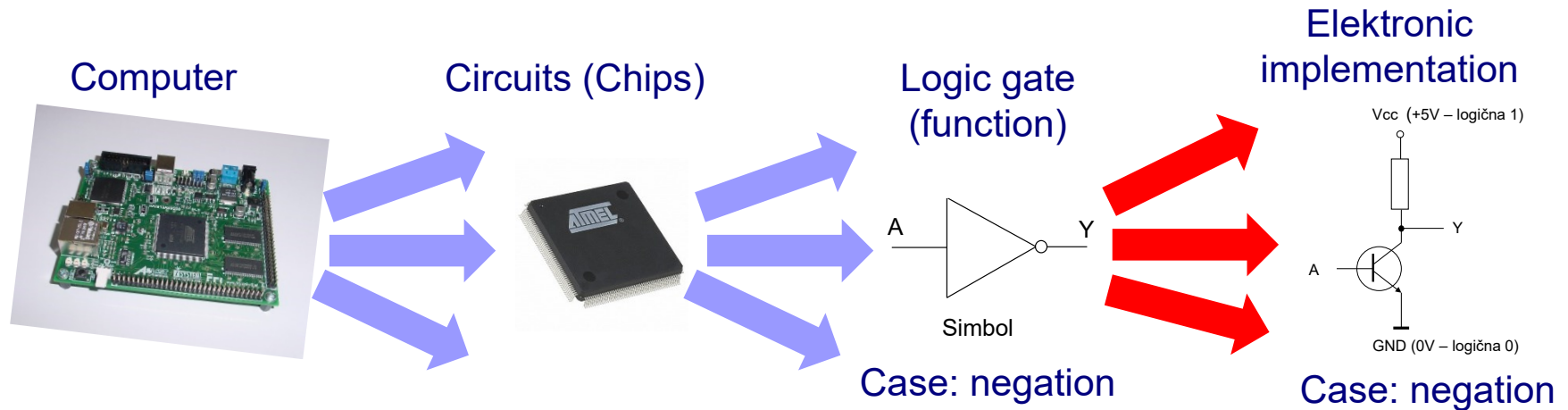
- *analogue voltages*
- *time-delays*
- *noises*



voltage levels $\approx 0V, \approx 3.3 (5) V$

Electrical realization

Physical structure of the computer



Information (instructions and operands) are in computers represented in binary system, with the help of electrical signals

- Two states (symbols) 0 and 1 are represented with two voltage levels.
 - **State 0** can be represented with low voltage (around 0V)
 - **State 1** can be represented with high voltage (up to +5V)
- Simple realization with a switch - example:
 - **State 0** – switch open – low voltage
 - **State 1** – switch closed – high voltage



- One switch can be in two states, state 0 or 1.
- Such a switch can memorize 1 bit of information.
- Basic memory cell can be imagined as such a switch. It shows its state and we can store 1 bit (0 or 1) of information into it.
- If we want to store more than only 1 bit of information, we need more cells.

Realization of switches in the development of digital computers – technology evolution

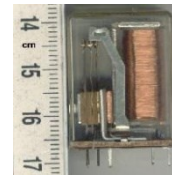
Prefixes for units of measurement

Abbreviation	Name	Value	Exponent (scientific notation)
p	pico	0,000 000 000 001	10^{-12}
n	nano	0,000 000 001	10^{-9}
μ	micro	0,000 001	10^{-6}
m	milli	0,001	10^{-3}
K	kilo	1 000	10^3
M	mega	1 000 000	10^6
G	giga	1 000 000 000	10^9
T	tera	1 000 000 000 000	10^{12}

Realization of switches as the basic building block - summary:

- Electro-mechanical switch

- 1939: Relay,



switching time 1-10ms

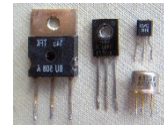
- Electrical switch

- 1945-1955: Vacuum tube,



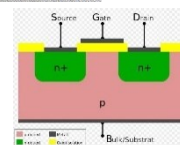
switching time $\sim 5\mu\text{s}$

- 1955: Transistors \rightarrow ,



switching time $\sim 10\text{ns}$

- 1958: Integrated circuit - chip,

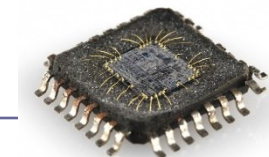


switching time 2-10ns

- 1980: VLSI integrated circuit

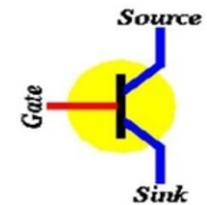
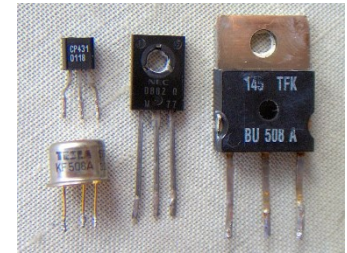
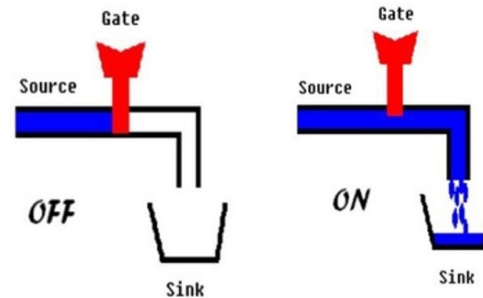
- Very Large Scale Integration

switching time $< 0.1\text{ns}$

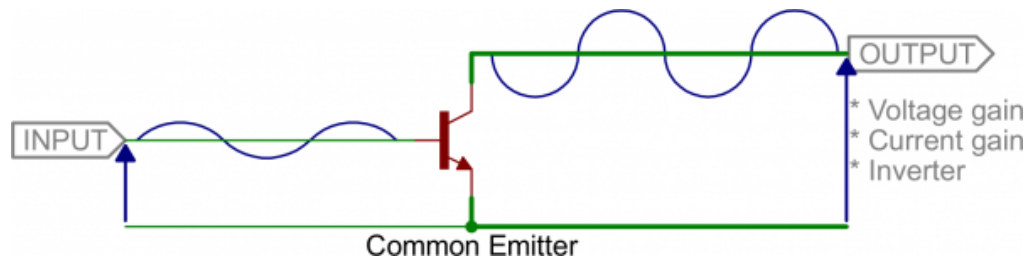


Transistor can be used as :

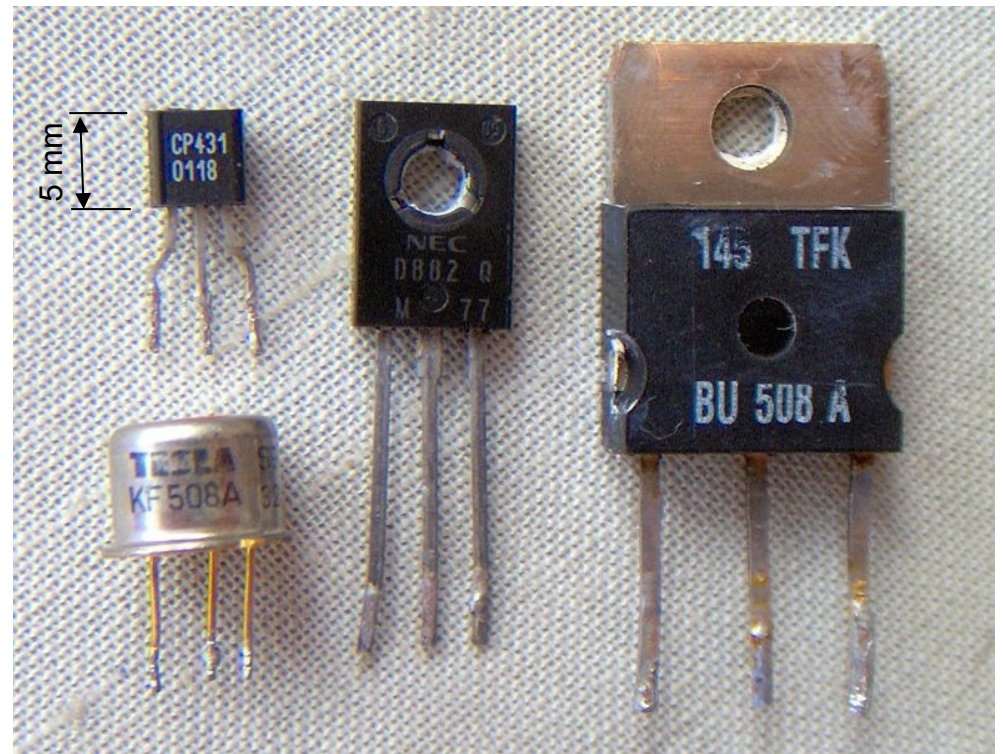
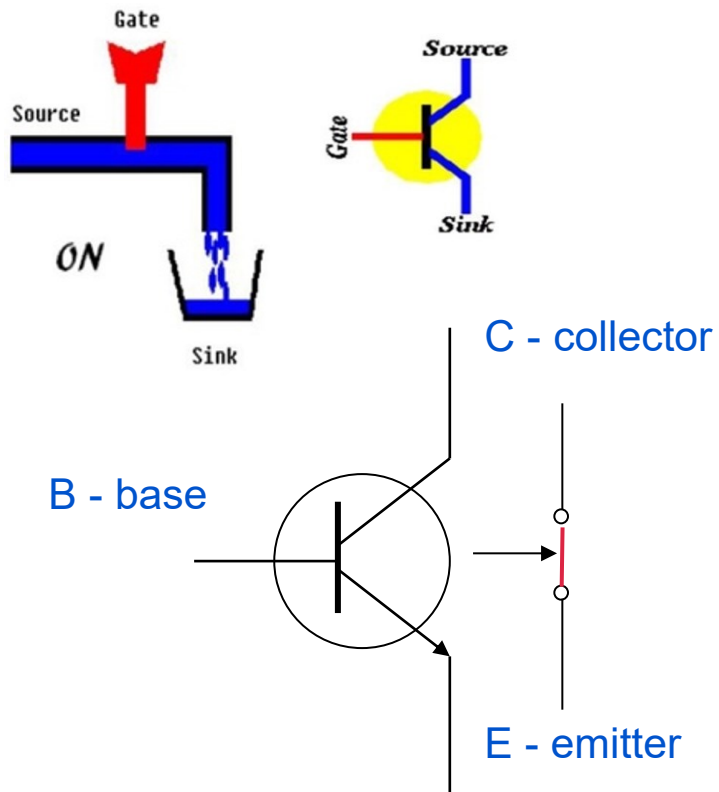
- Switch
 - In digital circuits



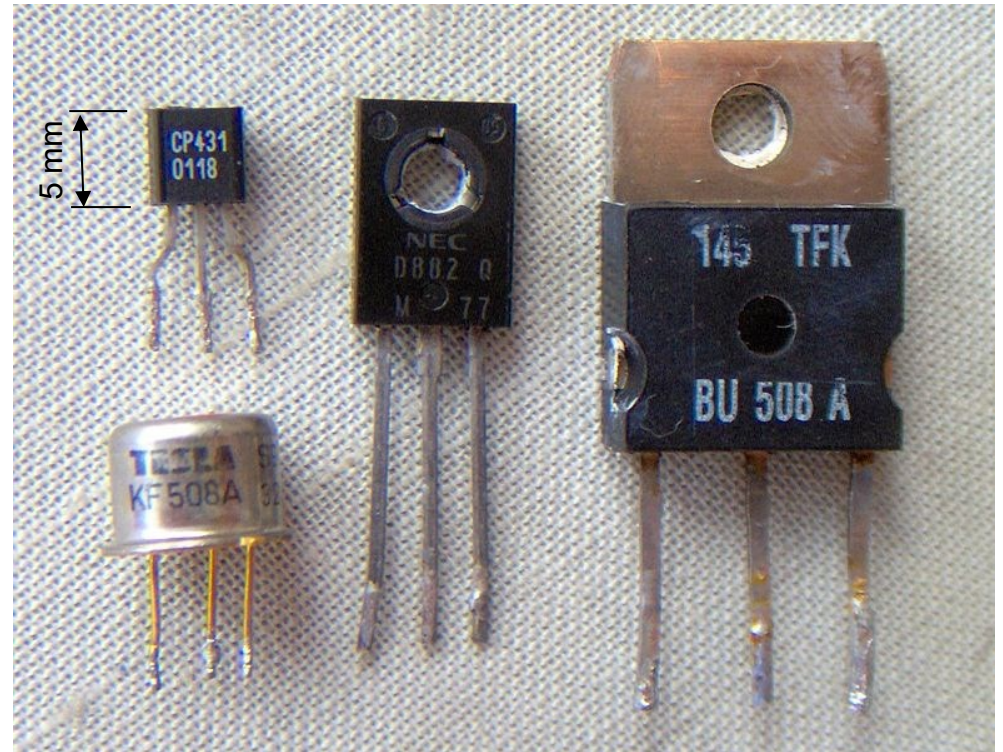
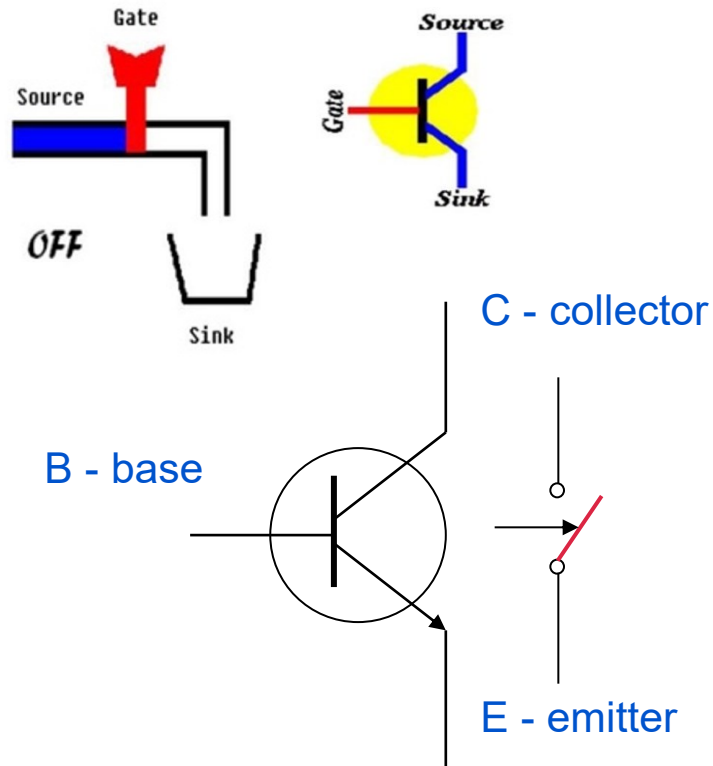
- Signal amplifier
 - Electronic circuits (amplifiers)



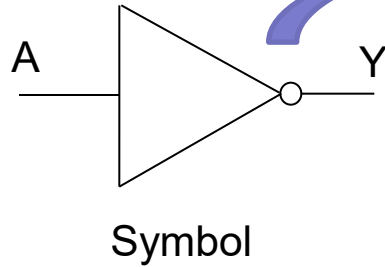
□ Transistor as switch - ON



□ Transistor as switch - OFF



Realization of the logical function NEGATION (NOT)

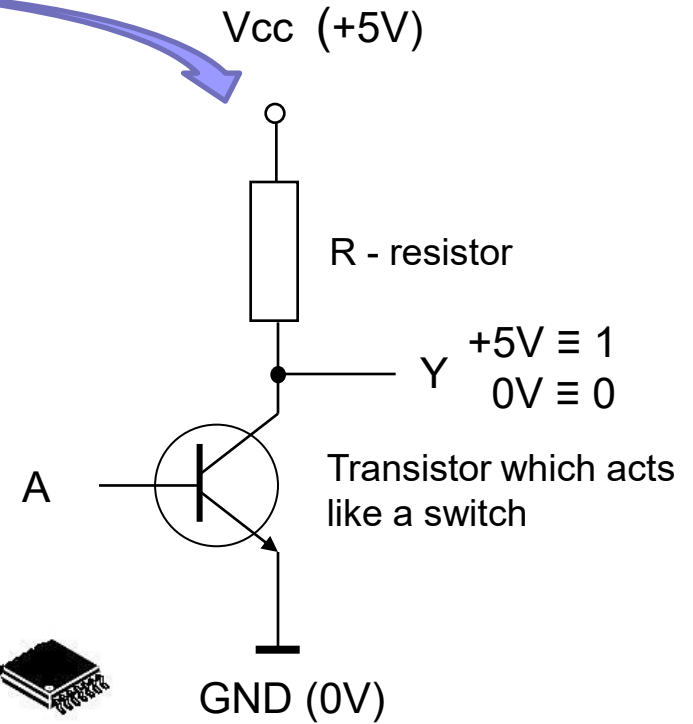
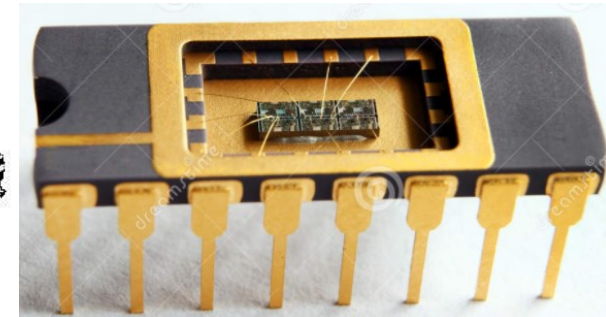
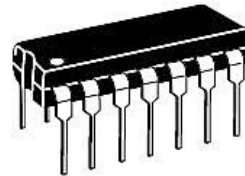
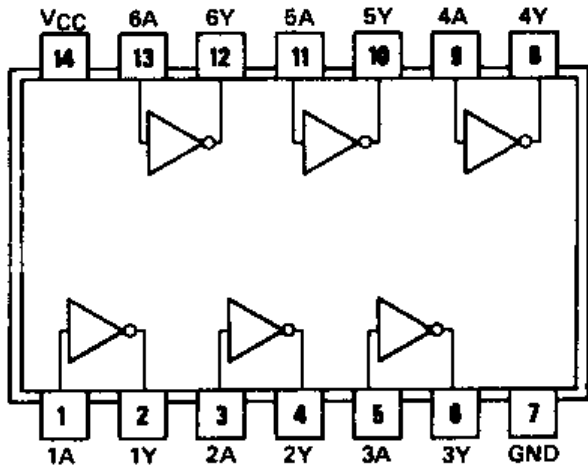


A	Y
0	1
1	0

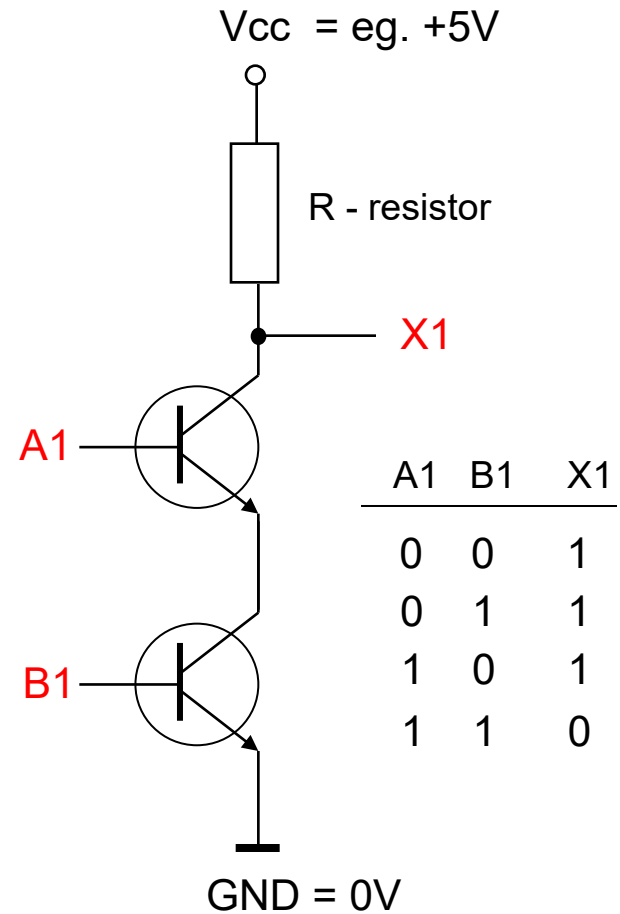
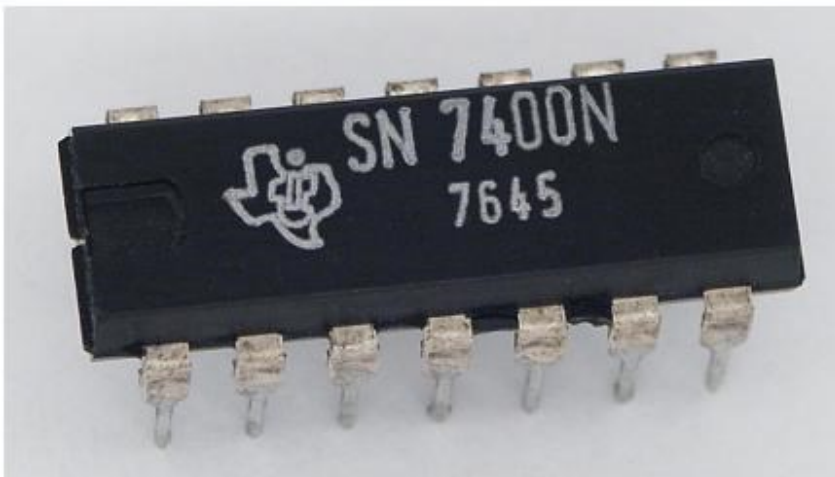
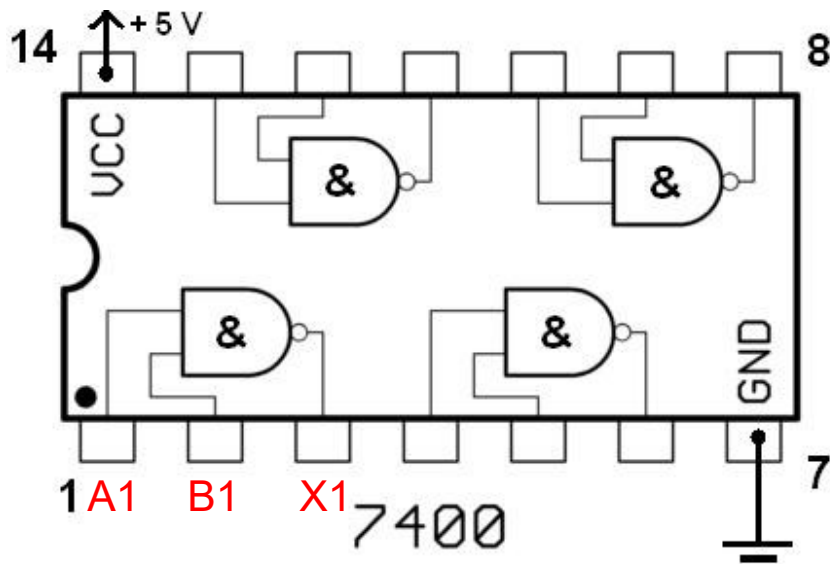
Truth table

IC (Integrated Circuit) with 6 negators

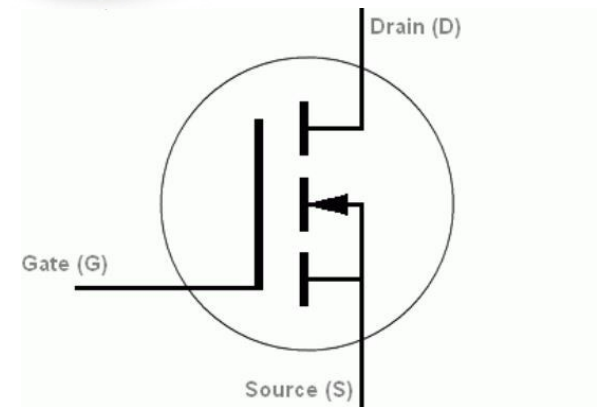
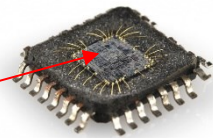
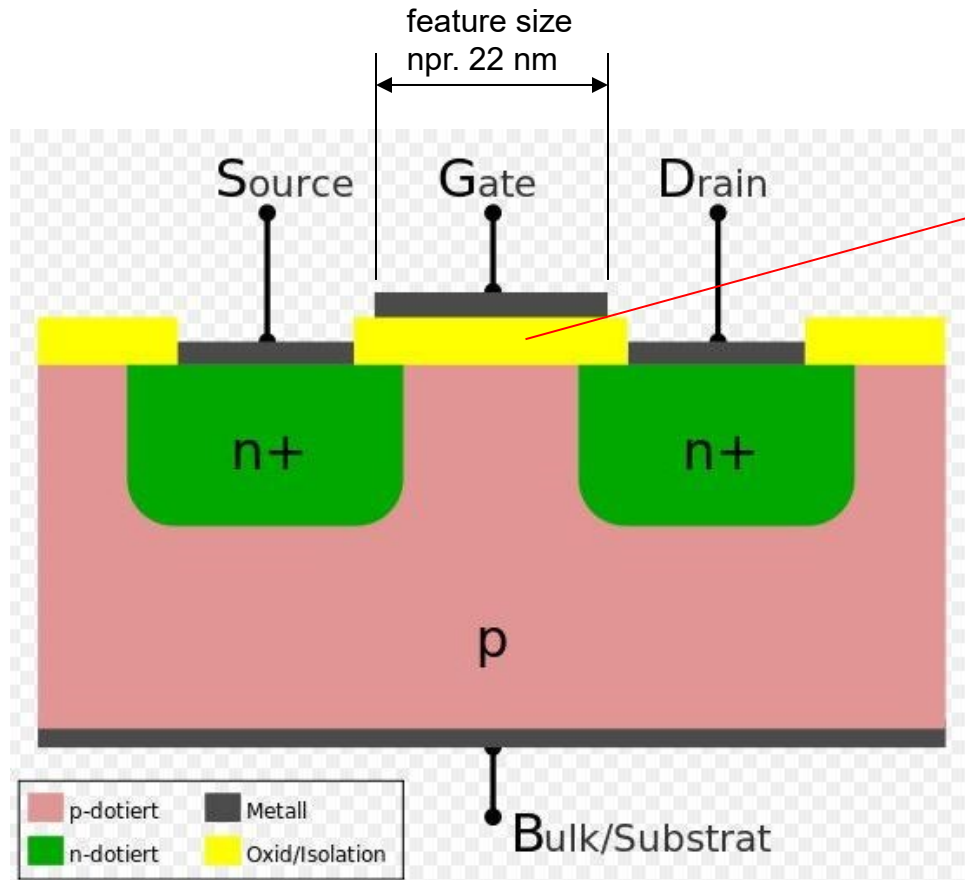
7406



Realization of the logical function NAND (Negated conjunction)



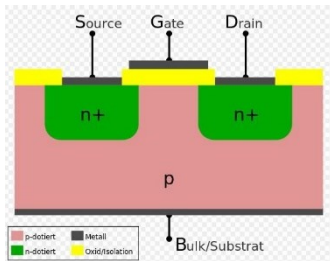
Transistors as a part of the integrated circuit



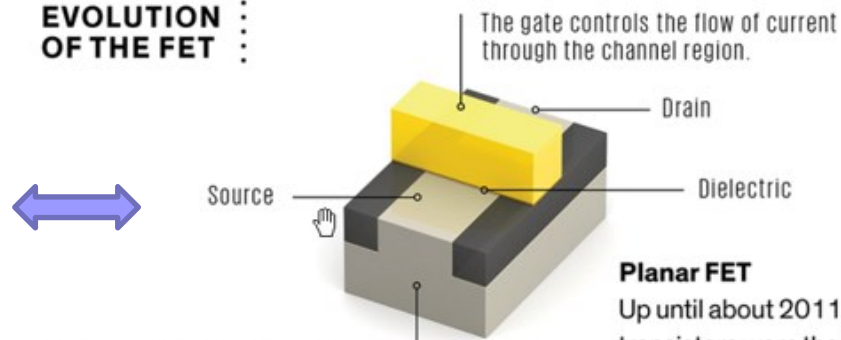
Introduction

Transistors evolution in modern circuits:

- From 2D to 3D -> less space, higher density !!!



EVOLUTION OF THE FET

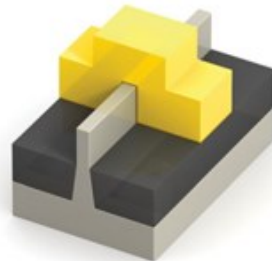


planar 2D (horizontal)

Planar FET

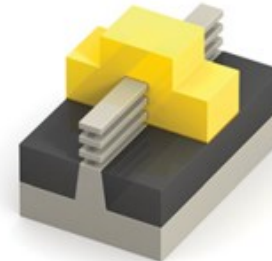
Up until about 2011, planar transistors were the best devices available.

Charge can leak through the channel region and waste power.



FinFET

Surrounding the channel region on three sides with the gate gives better control and prevents current leakage.



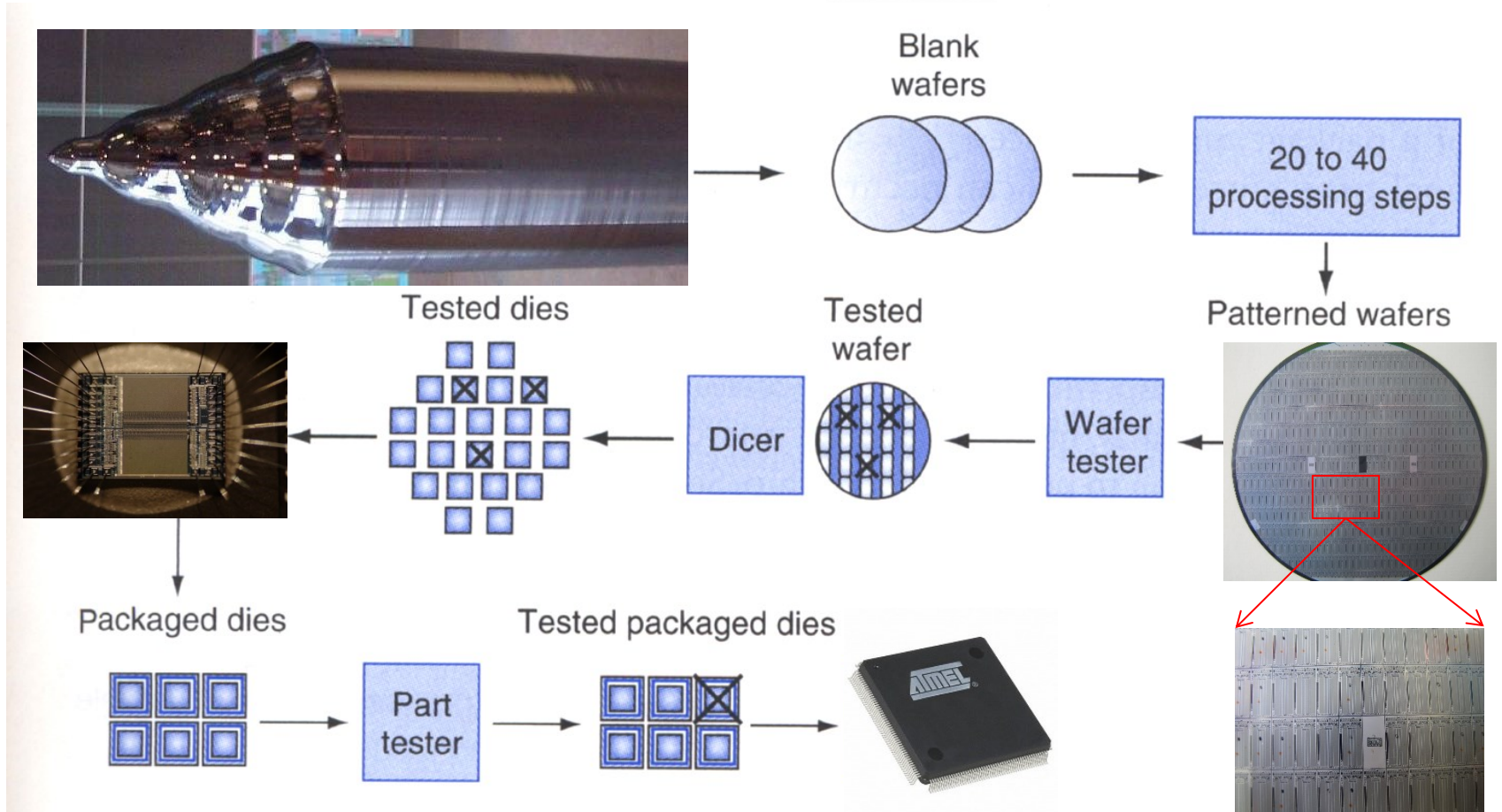
Stacked nanosheet FET

The gate completely surrounds the channel regions to give even better control than the FinFET.

3D transistor (vertical)

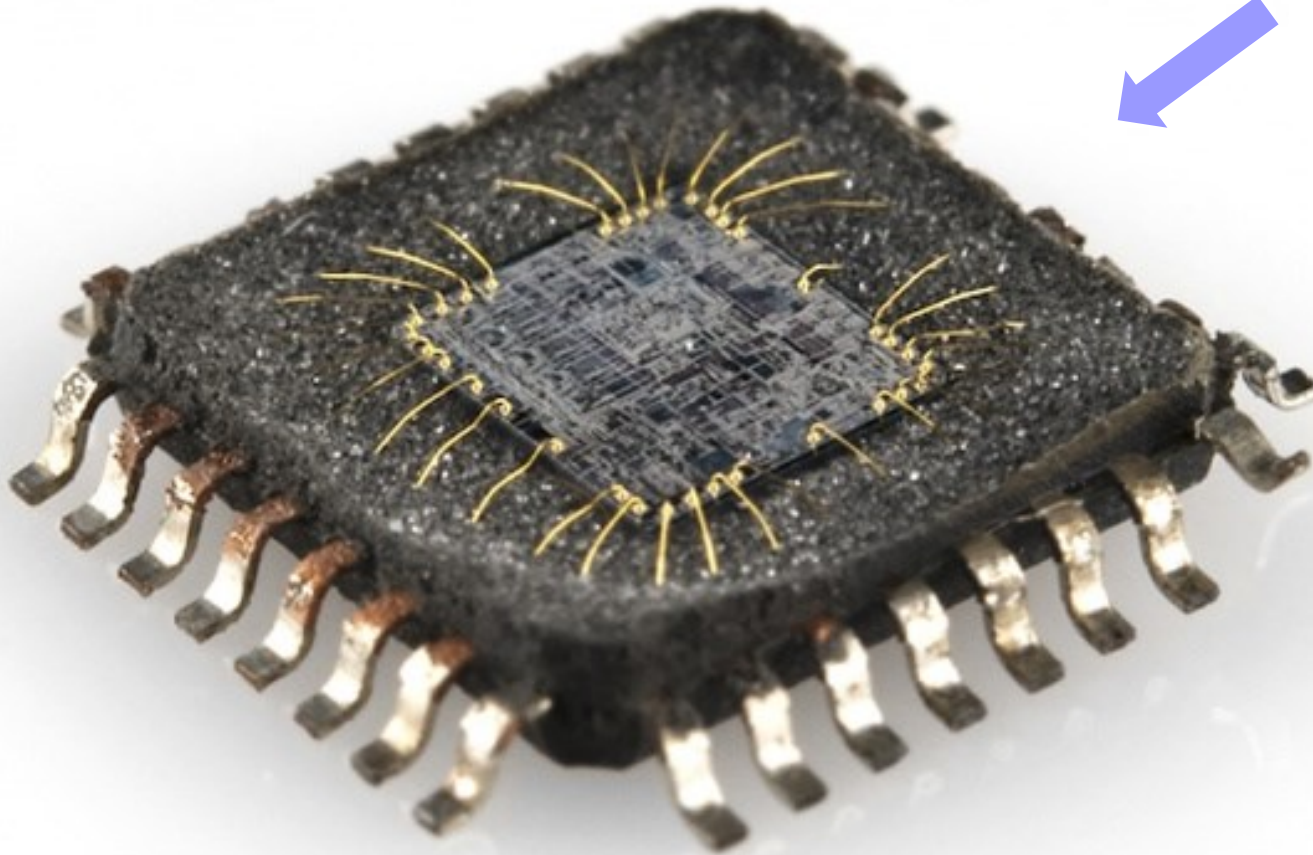
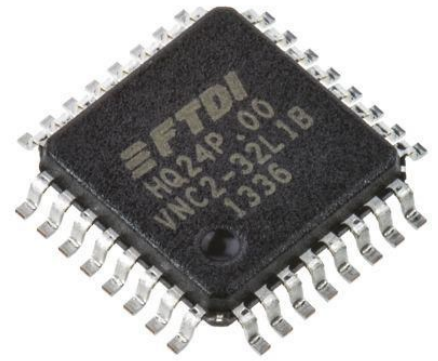
Si atom's diameter is 0.24nm!!!

The process of making a VLSI chip



David A. Patterson, John L. Hennessy:
Computer Organization and Design, Fourth Edition

VLSI chip - inside



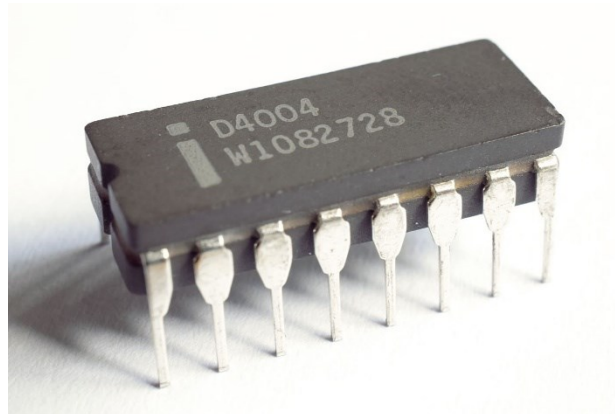
- ? nm process (feature size ? nm)
 - The parameter **feature size** in integrated circuits mostly determines the number of transistors on the integrated circuit and its properties.
 - Determines the smallest size of any object on the integrated circuit.
 - The object can be a part of the transistor, connection wire, space between two objects. The whole transistor is usually bigger.
 - The number of transistors on the chip depends on the size of the transistor. **The number of transistors is increasing quadratically according to the reduction of the parameter feature size**

■ Problems in contemporary VLSI technologies

- Switching speed of transistor is slowly progressing
- Density of transistors is increasing faster -> PARALLELISM
 - Density increase is more and more limited
- Reduction of elements' dimensions -> TROUBLE (heating, noises)
 - Excess heating dissipation -> COOLING
 - Lower resistance to interference

Case 1:

- The first processor on a chip Intel 4004 (year 1971)
 - 2.250 transistors on a die size 3,2 x 4,2 mm
 - 10 μm process (feature size 10 μm = 10×10^{-6} m = 0,00001 m, human hair is approximately 100 μm thick)
 - 16 connectors (pins)
 - Instruction execution time 10,8 μs (= 0,0000108 s) or 21,6 μs
 - Power consumption 1,0 W
 - Price (according to nowadays standards) \$26

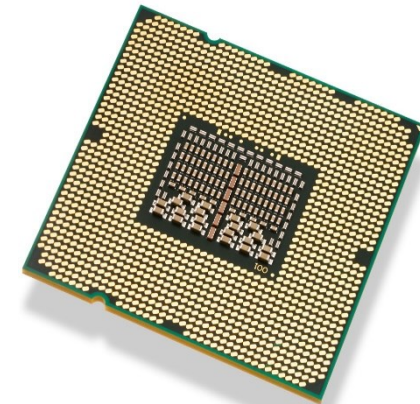
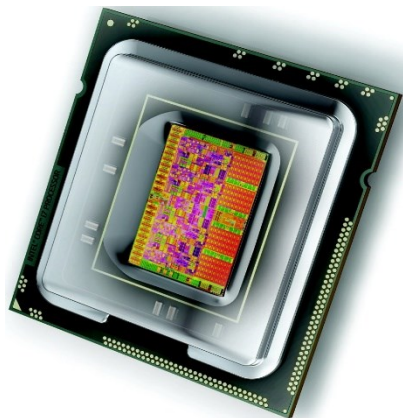
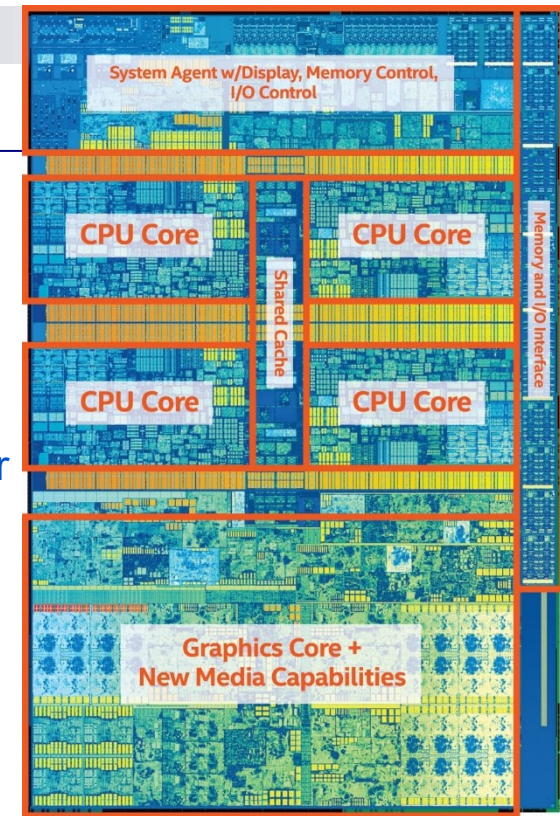


Case 2:

■ Processor Intel i7 7700

(microarchitecture Kaby Lake 7th generation year 2017):

- Number of transistors - Intel doesn't disclose this number
- **14 nm** process ($14\text{nm} = 14 \times 10^{-9} \text{ m} = 0,000000014 \text{ m}$)
- Size of the chip - Intel doesn't disclose this information
- 4 cores (4 processors, 8 threads), graphical processor
- **1155 connectors (pins)**
- Power consumption (TDP) **65 W**
- Recommended Price (Intel) 303 \$ - 312 \$



Case 3:

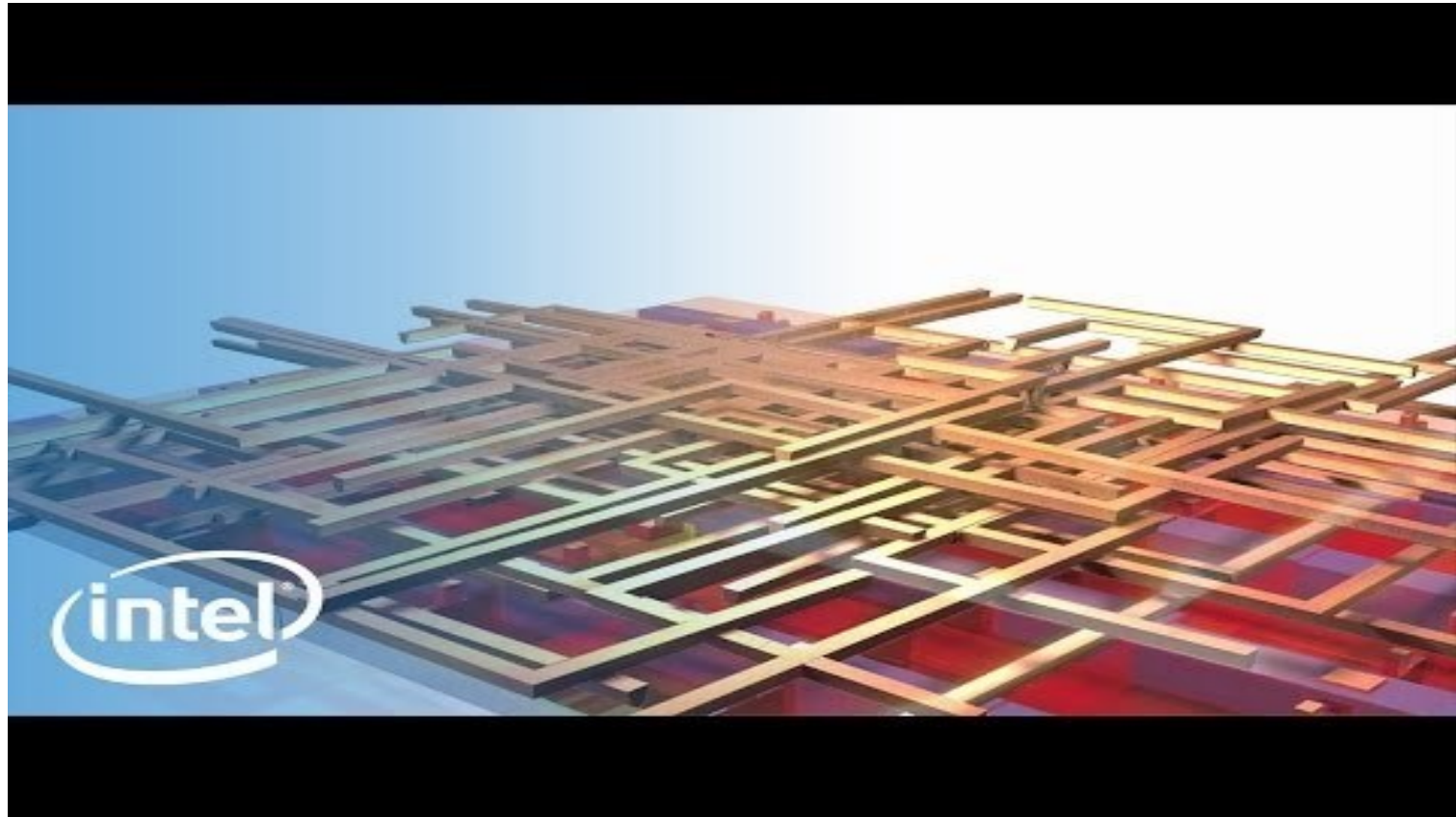
■ Processor Intel i9-11900

(microarchitecture Rocket Lake 11th generation year 2021):

- Number of transistors - Intel doesn't disclose this number
- **14 nm** process ($14\text{nm} = 14 \times 10^{-9} \text{ m} = 0,000000014 \text{ m}$)
- Size of the chip - Intel doesn't disclose this information
- 8 cores (16 threads), graphical processor
- **1200 connectors (pins)**
- Power consumption (TDP) **65 W**
- Recommended Price (Intel) 439 \$ - 449 \$



Intel: The Making of a Chip with 22nm/3D Transistors (Youtube Video)



https://www.youtube.com/watch?v=d9SWNLZvA8g&ab_channel=Intel

Computer Architecture – 1. Introduction

Online quizzes (after lectures) – „Slido“:

- Each time a "podium" is recorded
- Use of student ID number:
 - we track your performance
 - can raise final grade, cannot lower it...
- Possibility of questions, active participation

RA-1 | My Space #RA1
26 Oct – 10 Nov 2022 Engage Your event has finished Share Present

Live polls Audience Q&A Analytics

Slido overview

Share Analytics



Engaged participants

134 out of 164 Slido participants engaged with polls or Q&A.



Q&A engagement

38 out of 164 participants asked a question or voted in Q&A.



Poll engagement

131 out of 164 Slido participants voted in a poll.

Participants asking in Q&A

11

Anonymous / all questions

13 / 15

Total poll votes

Participants up or down voting in Q&A

34

Answered / unanswered questions

14 / 1

Polls with responses

Participants voting in polls

131

Upvotes / downvotes

57 / 0

Average votes per poll

Engaged participants score ⓘ

82%

Q&A engagement score ⓘ

23%

Poll engagement score ⓘ