MANAGEMENT AND ANALYSIS OF PHYSICS DATASET (MOD. A)

Introduction

Introductions

- Andrea Triossi
  Dipartimento di Fisica ed Astronomia
  Via Marzolo 8 – Room 3-44, Lab. 3-24
  andrea.triossi@unipd.it

- Main interests
  - Digital Electronics for physics experiments
  - Reconfigurable computing

- What about you?

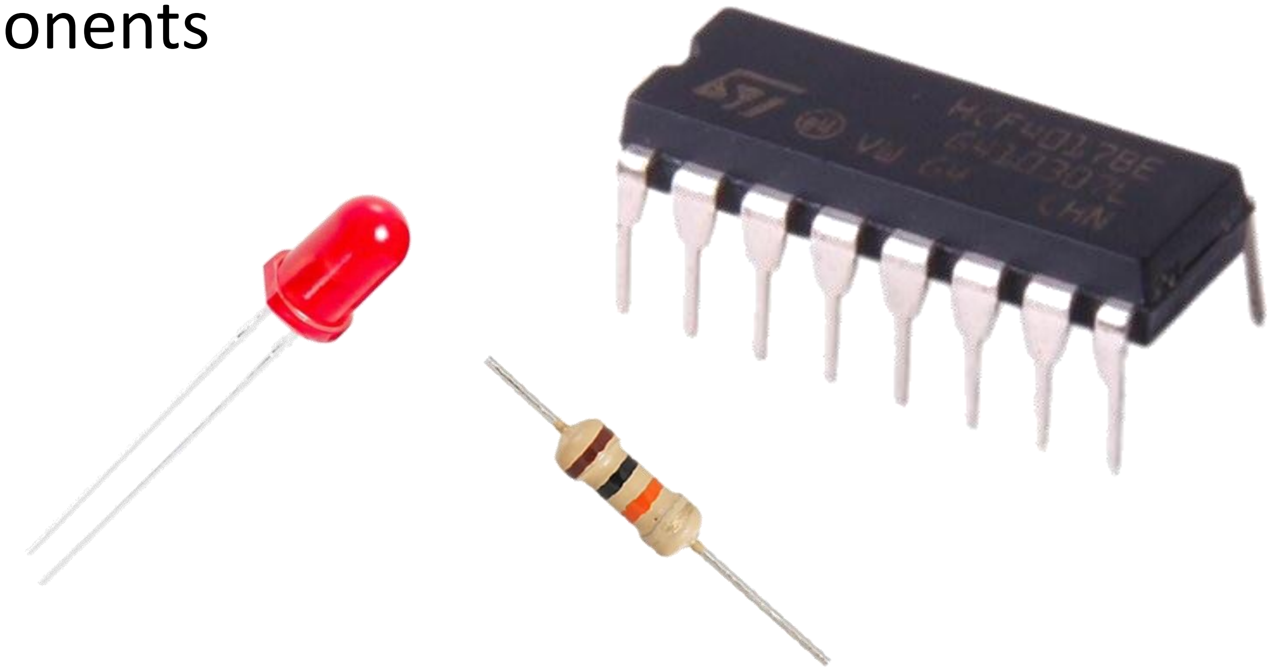
Physics

Computer science

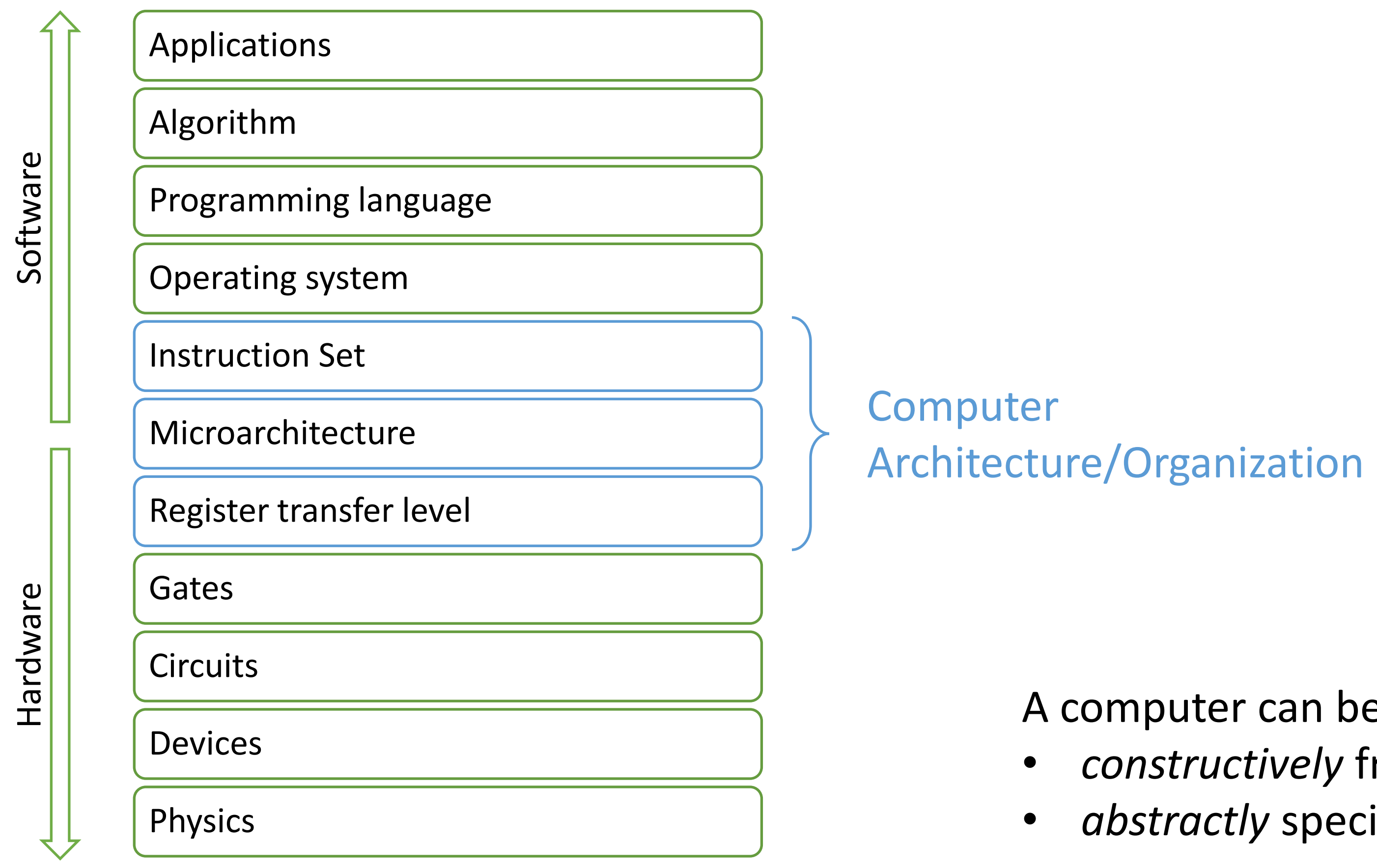
Electronic Engineer

Goal

- Make microprocessors understandable and interesting
- Keep the right level of abstraction
- Present the SAP (simple-as-possible) processor
- Build SAP using digital components
- Program/improve SAP
- Have fun



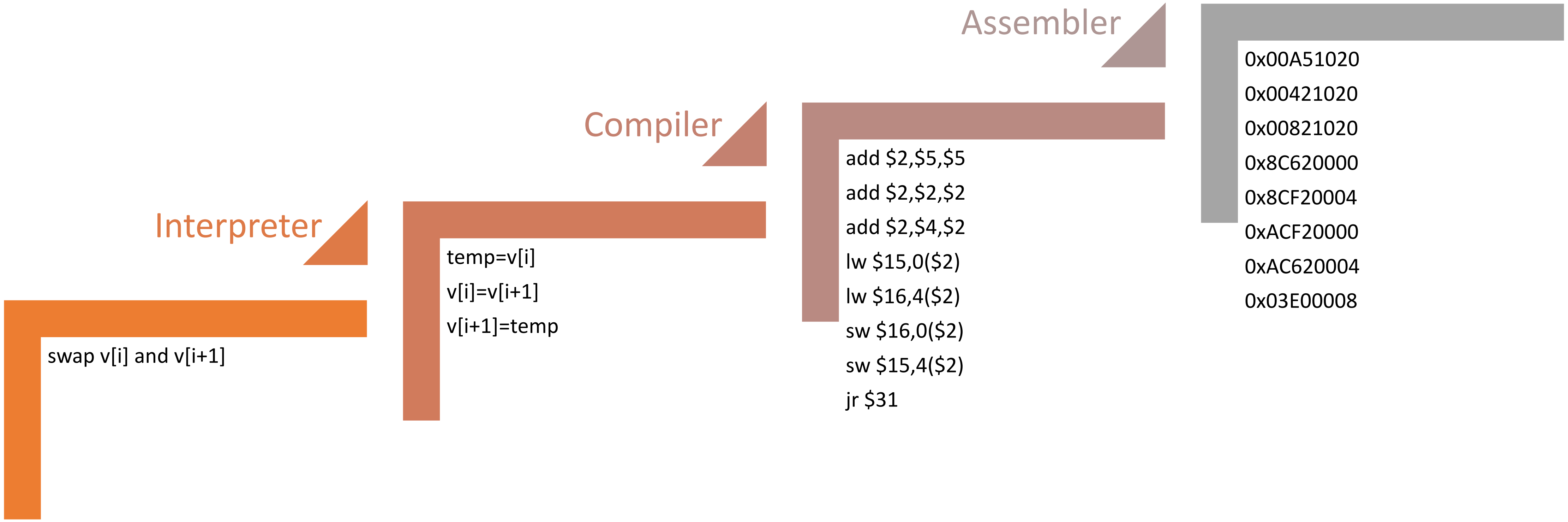
What we will (not) talk about?



A computer can be described:

- *constructively* from low-level chips
- *abstractly* specifying its machine language capabilities

Abstraction in programming



```
swap v[i] and v[i+1]
```

```
temp=v[i]
v[i]=v[i+1]
v[i+1]=temp
```

```
add $2,$5,$5
add $2,$2,$2
add $2,$4,$2
lw $15,0($2)
lw $16,4($2)
sw $16,0($2)
sw $15,4($2)
jr $31
```

```
0x00A51020
0x00421020
0x00821020
0x8C620000
0x8CF20004
0xACF20000
0xAC620004
0x03E00008
```

Very high-level language objectives or tasks

High level language statements

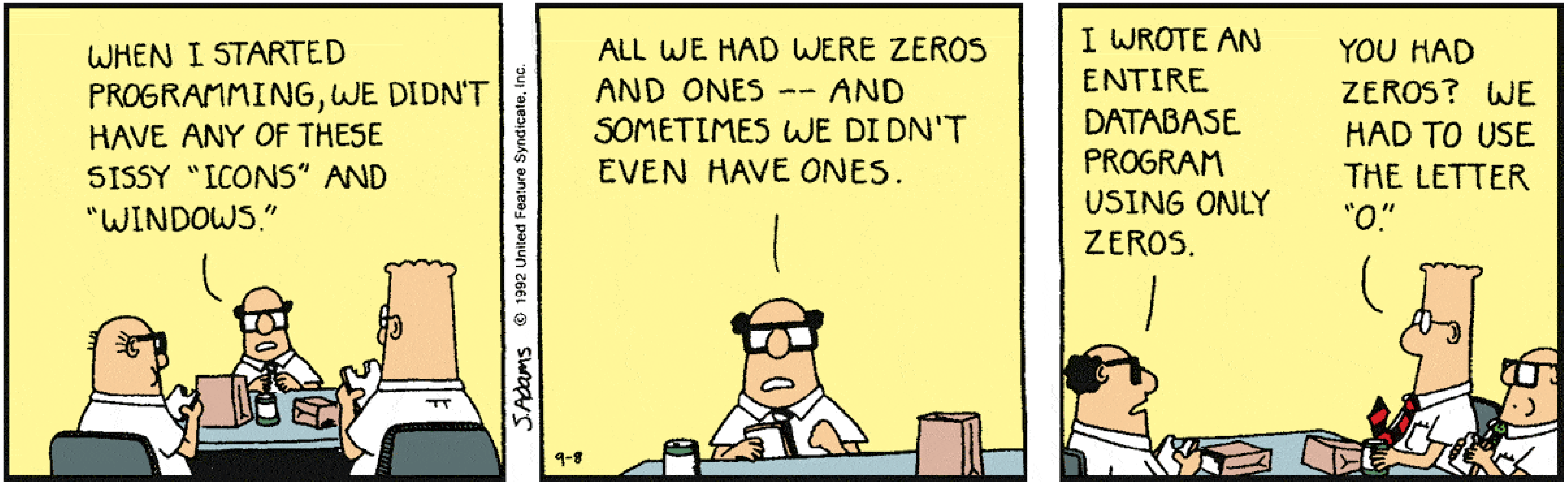
Assembly language instructions

Machine language instructions

- More abstract
- Machine independent
- Easier to write, read, debug or maintain

- More concrete
- Machine specific
- Error prone
- Harder to write, read, debug or maintain

Old school


WHEN I STARTED PROGRAMMING, WE DIDN'T HAVE ANY OF THESE SISSY "ICONS" AND "WINDOWS."
© 1992 United Feature Syndicate, Inc.
S. Adams
9-8
ALL WE HAD WERE ZEROS AND ONES -- AND SOMETIMES WE DIDN'T EVEN HAVE ONES.
I WROTE AN ENTIRE DATABASE PROGRAM USING ONLY ZEROS.
YOU HAD ZEROS? WE HAD TO USE THE LETTER "O."

Machine language

- Agreed-upon formalism designed to code low-low level programs as series of machine instructions
- Using instructions, the processor performs operations (arithmetic, logic, store, fetch, move, test conditions…)
- Machine language aims to be directly executed to a hardware platform to totally control it
- Generality and power of expression, even if desired, don’t belong to machine language
- Machine language is the fine line where hardware and software meet
- Although you will never program in machine language, it is necessary for a complete understanding of computer architecture

Machines

Machine language is designed to manipulate a *memory* using a *processor* and a set of *registers*

Memory

- Collection of hardware devices that store data and instructions
- Array of cells called *words* each having a unique *address*

Processor

- Also called Central Processing Unit (CPU)
- A device able to perform a set of elementary operations
- The operands (results) are binary values that come from (are stored in) registers and memory locations

Registers

- Simplified memory structures able to hold a single value
- High-speed local memory

Languages

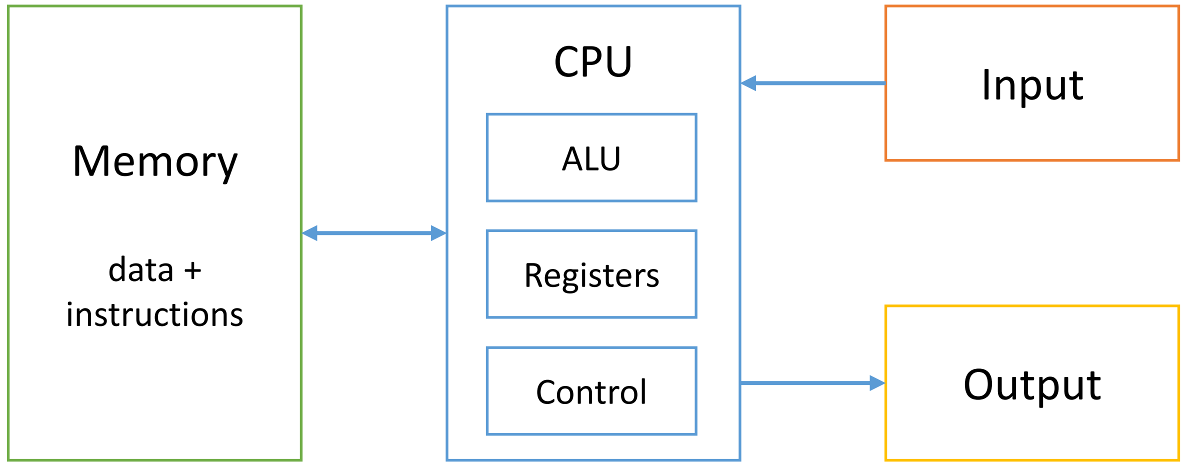
- A series of coded instructions that depends on the hardware specification and a syntax
- For instance
  - 1010001100011001 in a 16-bit computer could be *set R3 to R1 + R9*
  - 4-bit field for the instruction and three 4-bit field for the operands
- Difficult to remember -> *mnemonics:* symbolic label
  - 1010 could represent the mnemonic *ADD*
  - 1010001100011001 -> *ADD R3, R1, R9*
- Symbolic notation is called *assembly*
- Each computer has a different machine language -> Tower of babel
- However, there are similar sets of generic commands

Stored program

- General-purpose computer are capable of running *stored programs* written in machine language
- Computers are based on fixed hardware platform capable of executing a fixed set of instructions
- Instructions can be combined like building blocks yielding a sophisticated program
- The logic of the program is not fixed but stored and manipulated in computer memory like data -> software
- Universal Turing machine and von Neumann machine are based on the concept of *stored program*

Von Neumann architecture

- Practical architecture at the basis of almost all computer platforms today
- A *CPU* interacting with a *memory*, receiving data from some *input* device and sending data to some *output* device
- The memory stores not only the data but also the instructions that tell the computer what to do with data



Memory

- It holds two type of information
  - Data items
  - Programming instructions
- They are treated differently and sometimes they can be stored in separate units but usually they are just words at different addresses of the same random-access memory
- A data word can be either read (retrieved) or written (stored erasing the old value)
- In each step of computer's operation, the CPU fetches (reads) an instruction word, decodes it, executes it and figures out what next

CPU

- It is in charge of executing the instructions of the currently loaded program
- Three main hardware elements are used:
    - Arithmetic Logic Unit (ALU) is built to perform the basic arithmetic and logical operations
    - Registers are used to boost performance when is needed to store results locally
    - Control unit is the decoder of the instructions. It use the extracted information to orchestrate the execution

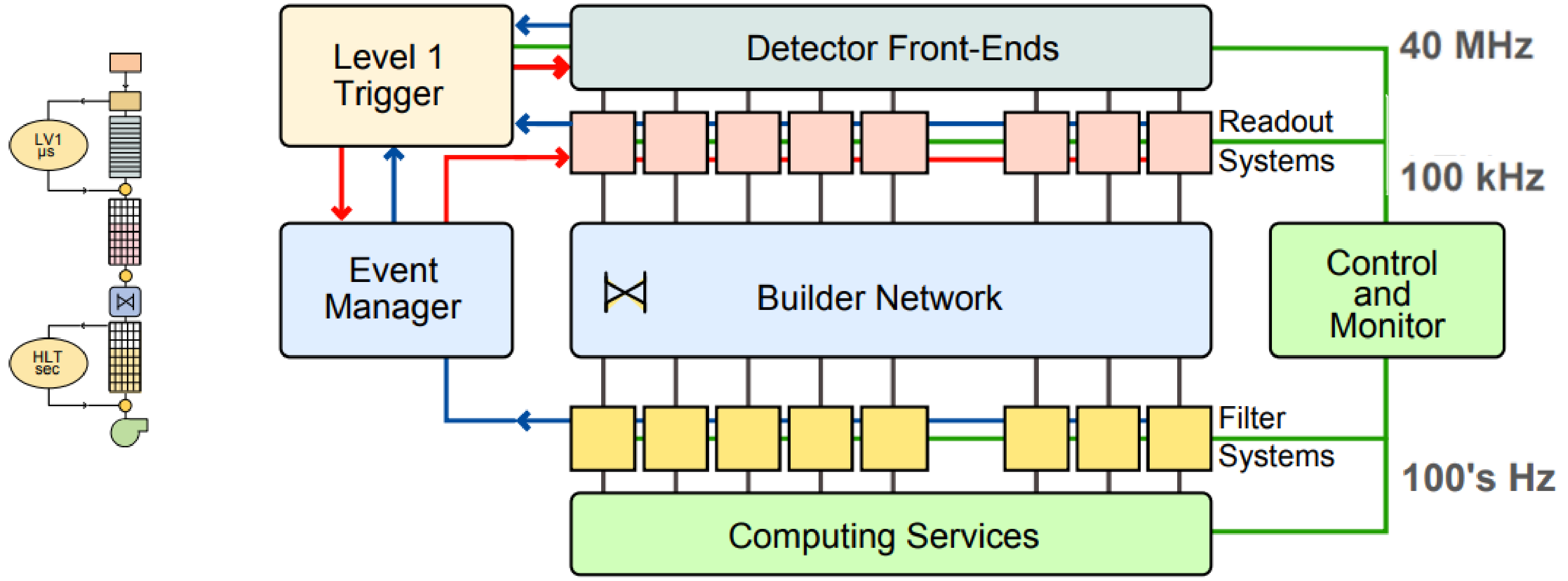
Input and output

- They are used to interact with the external environment
- For instance: screen, keyboard, USB device, etc.
- All those devices are very different, but they are treated in the same way thanks to a technique called *memory-mapping I/O*
- To the CPU they look like normal memory segments
- For input devices the memory map continuously reflects the physical state of the device
- For output devices the memory map continuously drives the physical state of the device
- In this way the design of a CPU is totally independent of the number and nature of the I/O devices
- When a new device is connected a new memory map is allocated at a particular base address. The program has just to manipulate (read/write) such memory

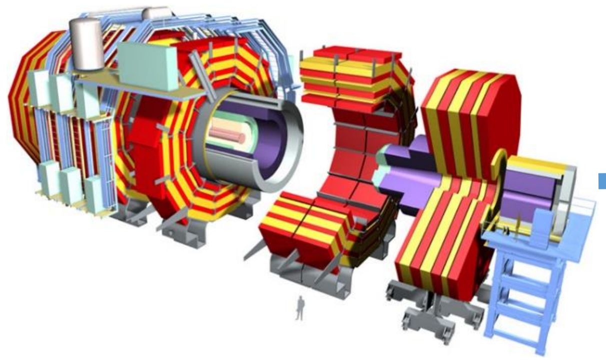
What does this have to do with physics?

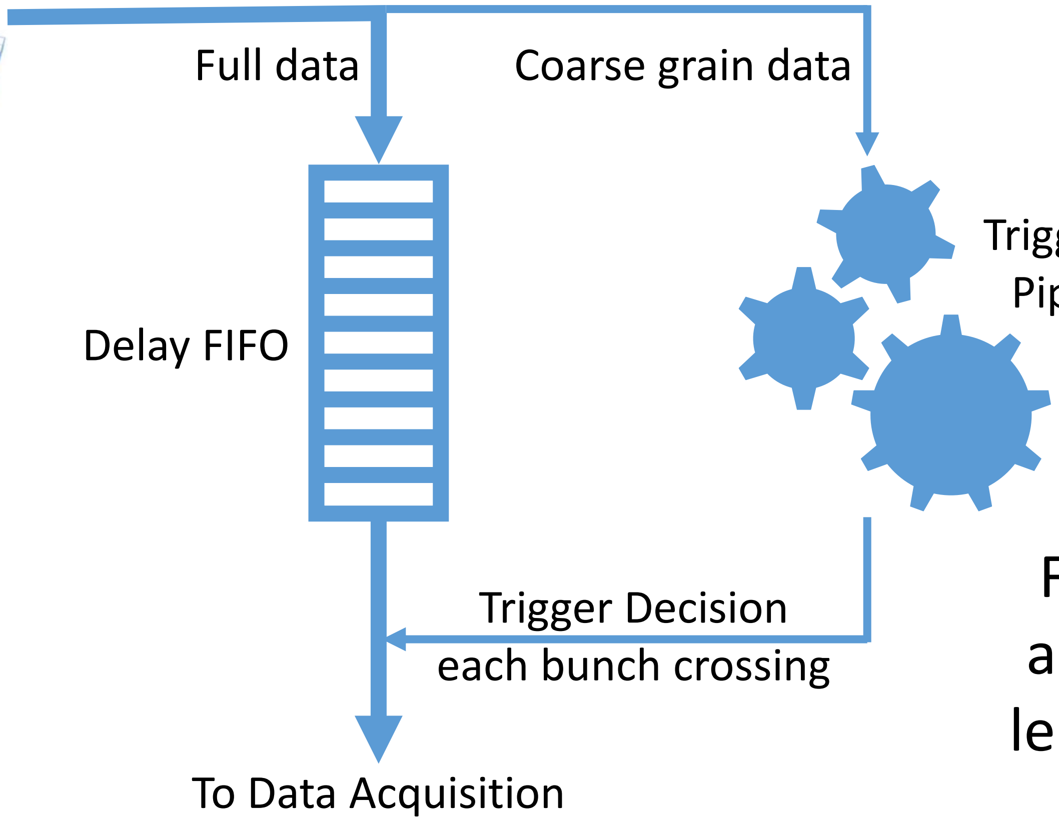
Data acquisition

- Collect the detector information at the full events frequency
- Execute physics selection algorithms on the events read out, in order to accept the ones with the most interesting physics content (achieve large rejection ratio)
- Forward these accepted events to the online services which monitor the performance of the detector and provide the means of archiving the events in mass storage



First-Level trigger

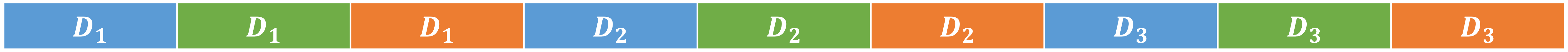




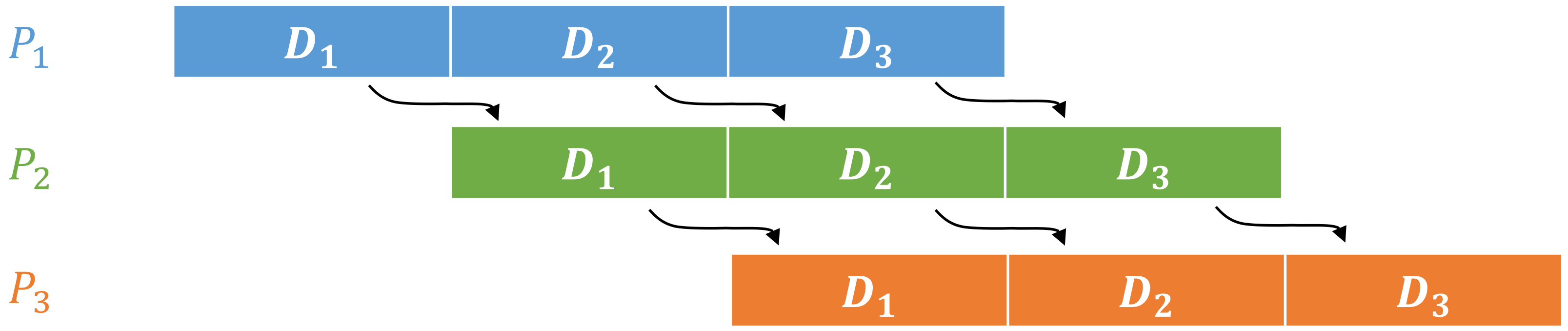
Data are moved synchronously with the accelerator clock:
25 ns bunch crossing

Fixed latency as short as possible to limit the length of the delay FIFO

Pipelining



Sequential execution

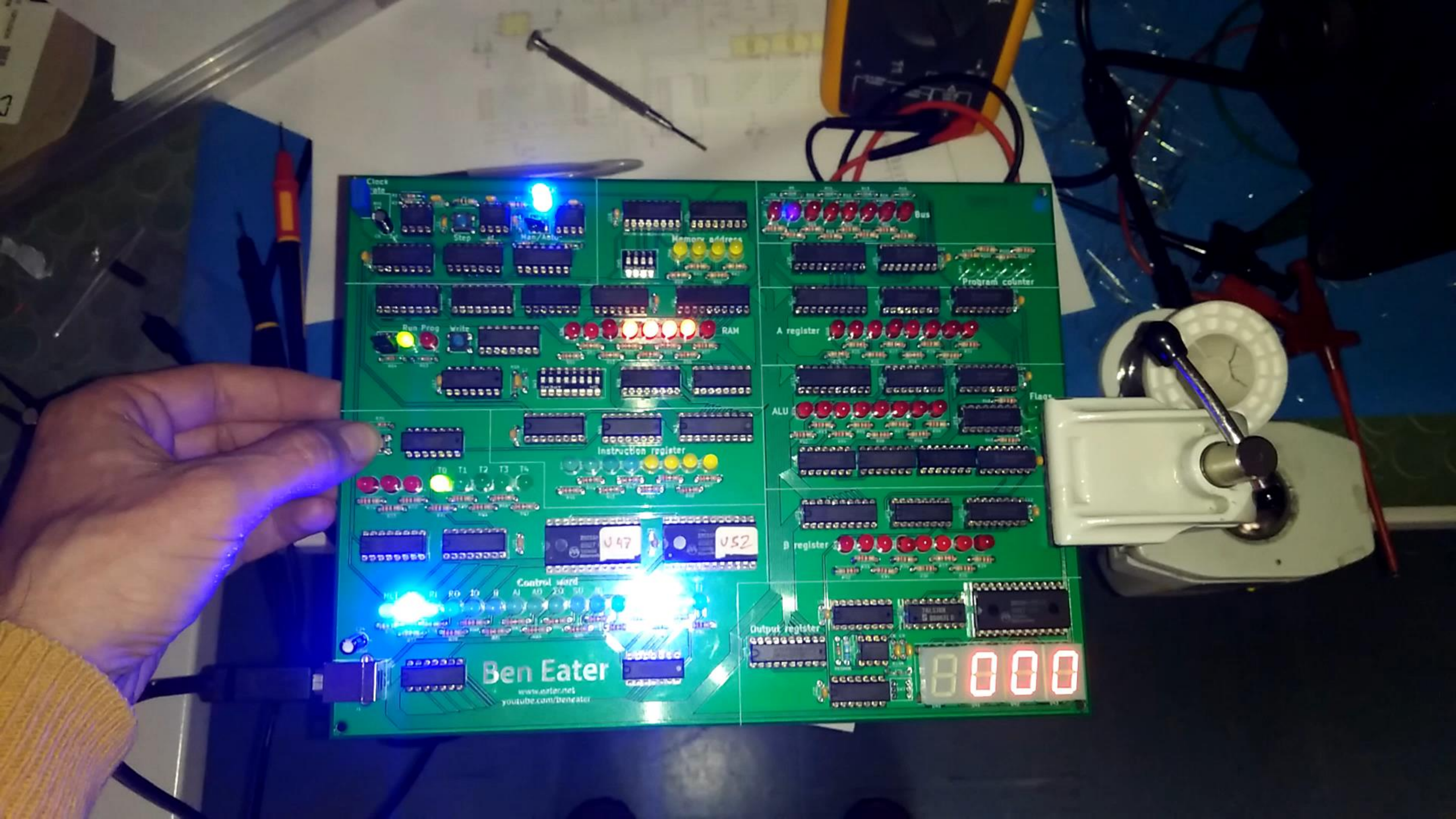


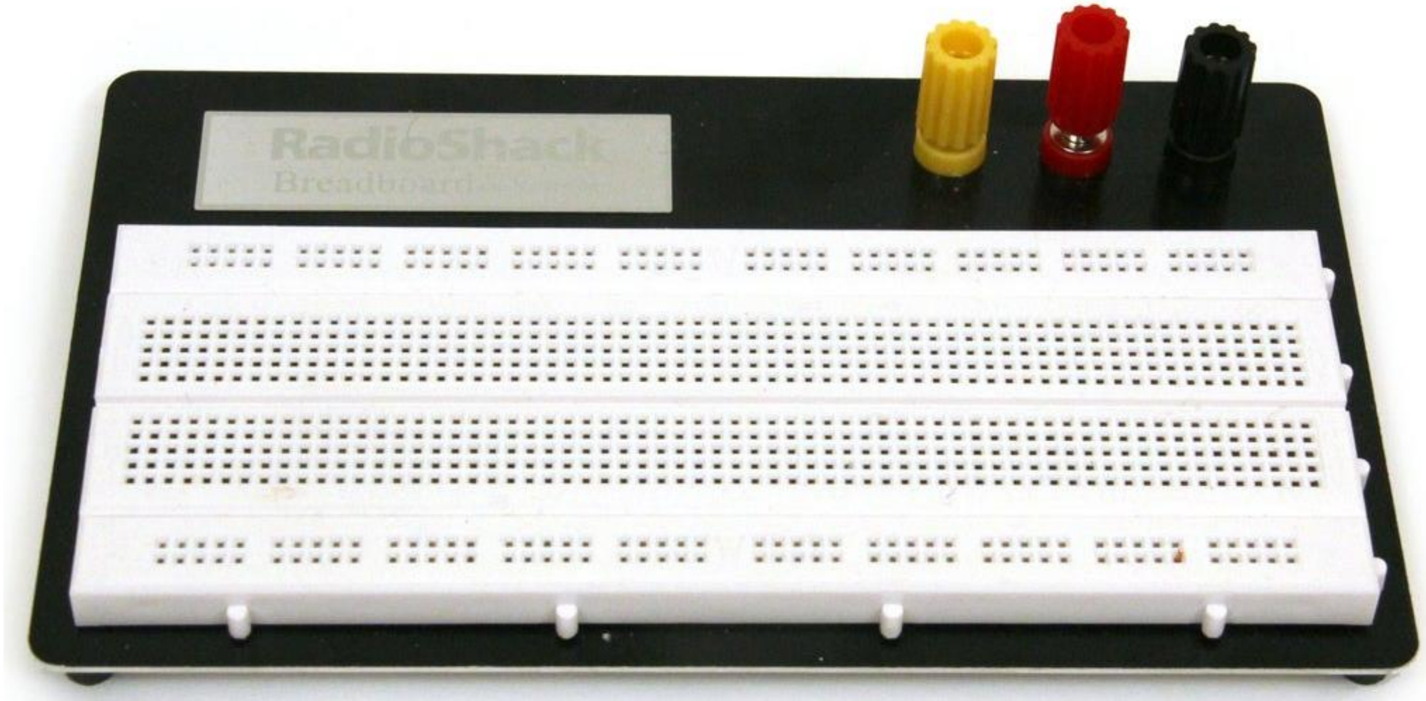
Pipelined execution

Course info

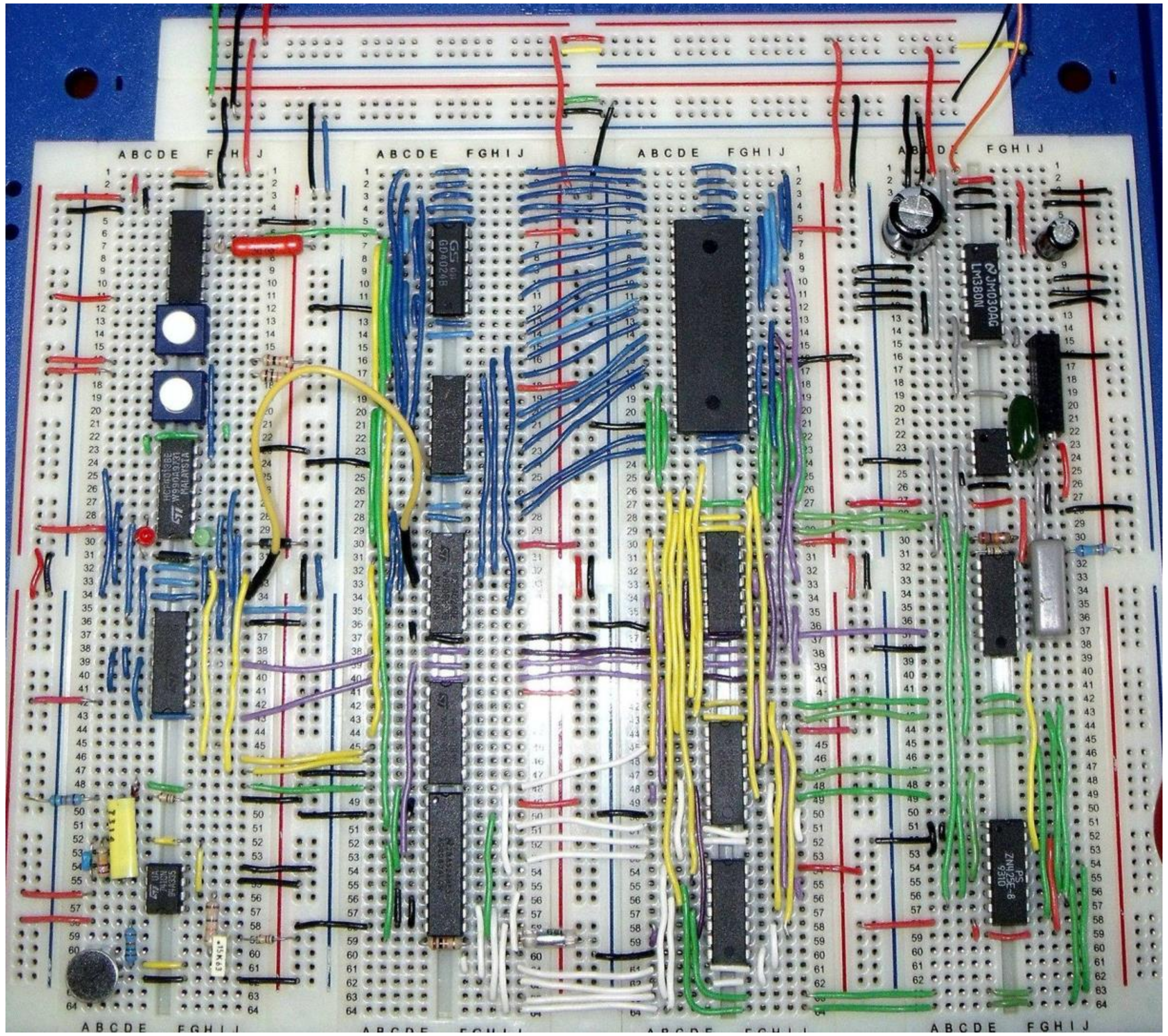
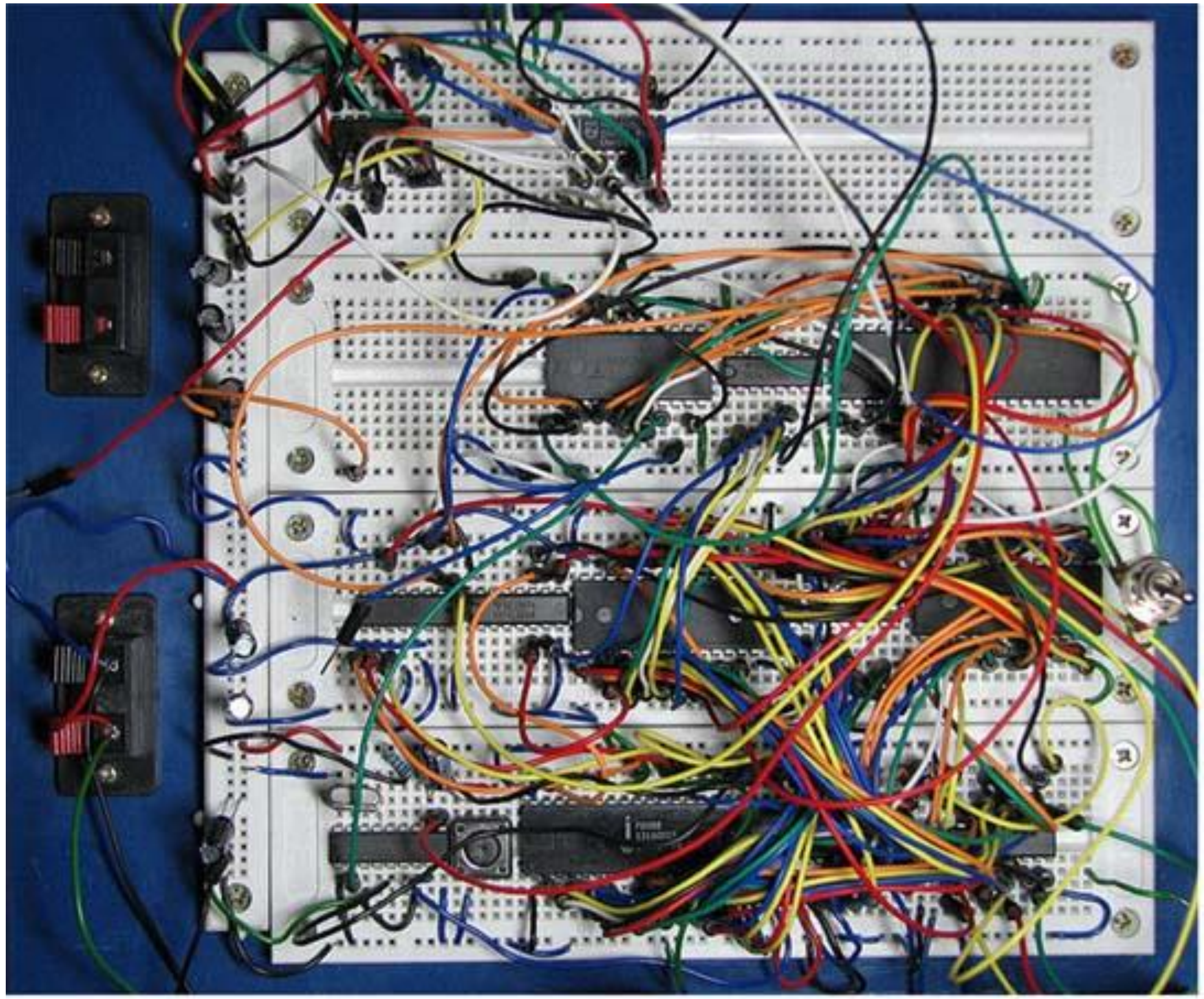
Practical info

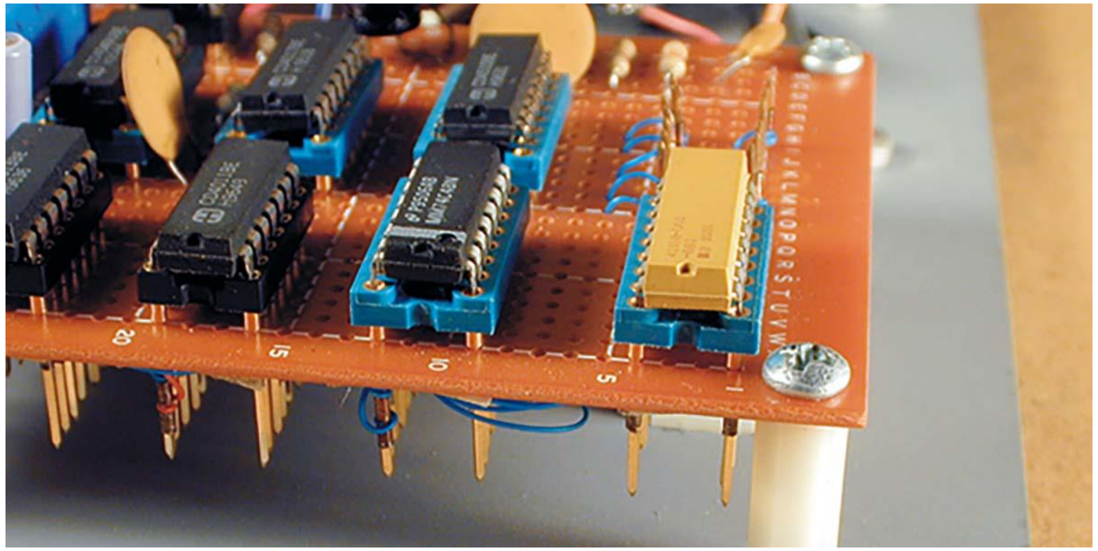
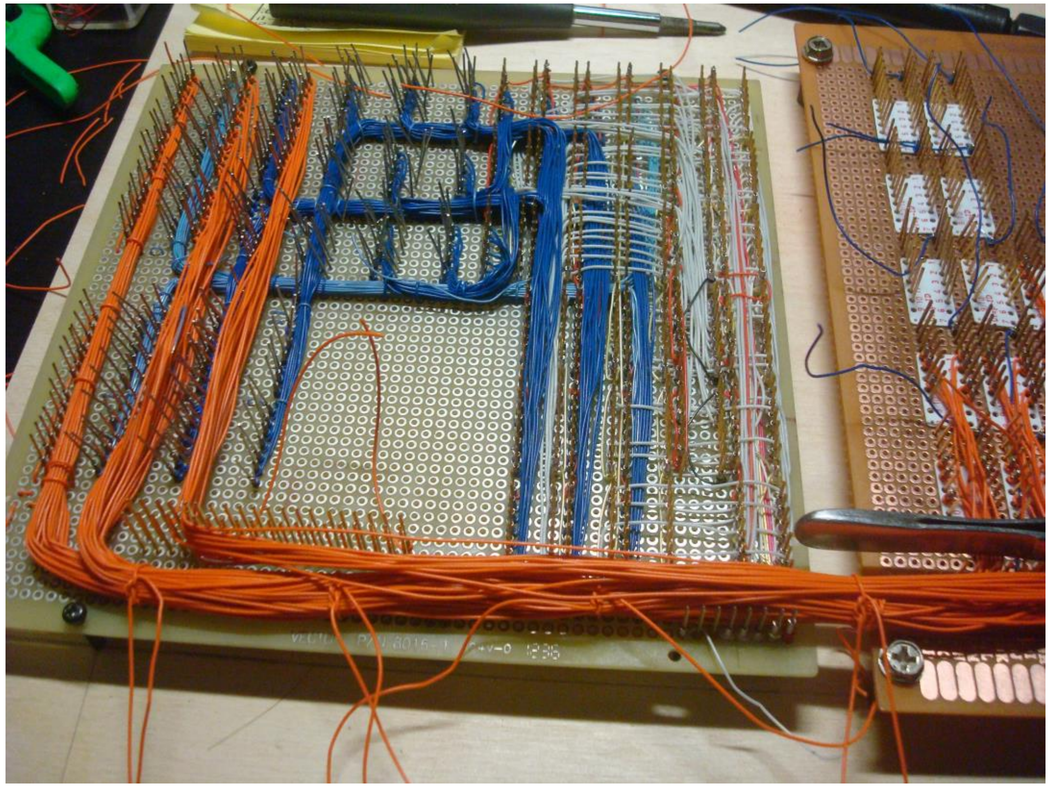
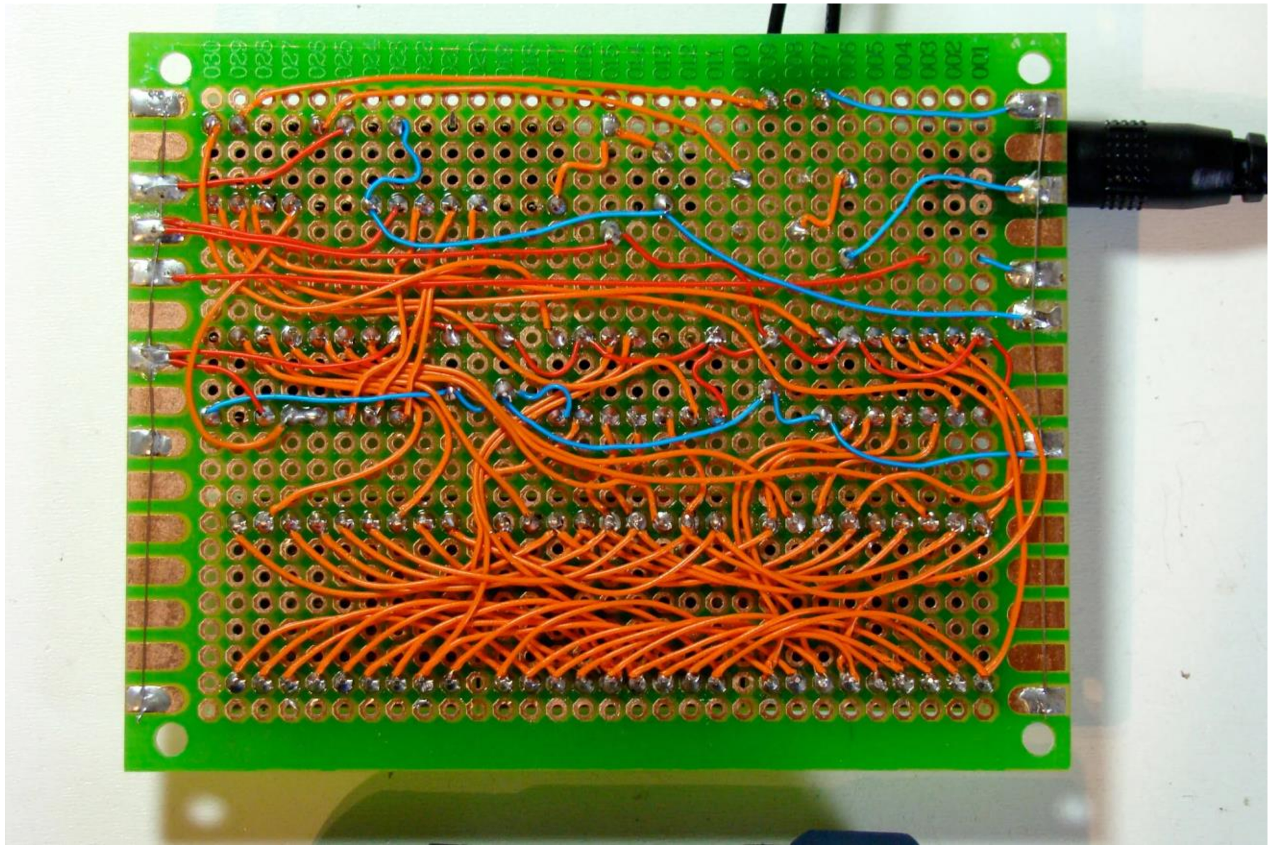
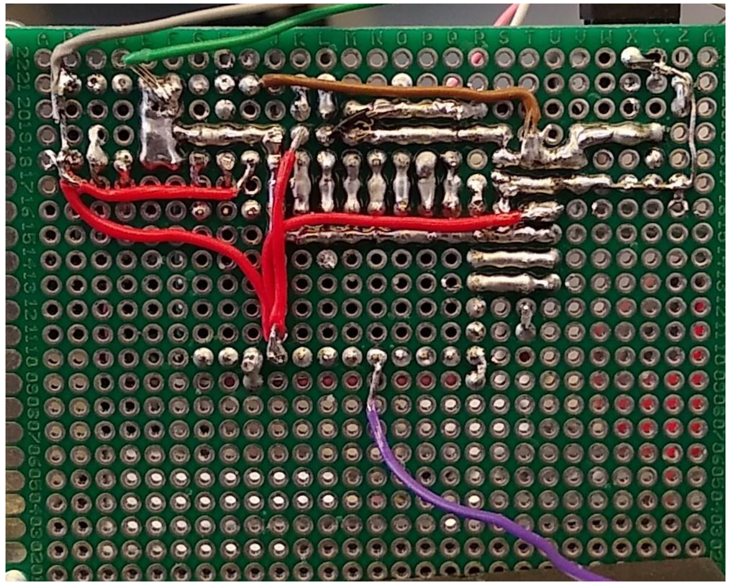
- ∼1/3 lectures (Room B, DFA)
  - Digital electronics without technology details
  - Basic elements of computer architecture
- ∼2/3 lab sessions (Lab 115, Polo didattico, via Loredan 10)
  - Build an 8-bit computer
- Suggested book:
  - Malvino - Brown, Digital Computer Electronics, Glencoe, McGraw-Hill, 1992
- Slides on moodle
  - https://stem.elearning.unipd.it/course/view.php?id=10230
- Zoom available only for foreign students waiting for visa (contact me)


Clock
Step
Man/Auto
Memory address
Bus
Program counter
Run Prog
Write
RAM
A register
ALU
Flags
Instruction register
T0
T1
T2
T3
T4
U47
U52
Control word
B register
Output register
000
Ben Eater
www.eater.net
youtube.com/beneater


RadioShack
Breadboard

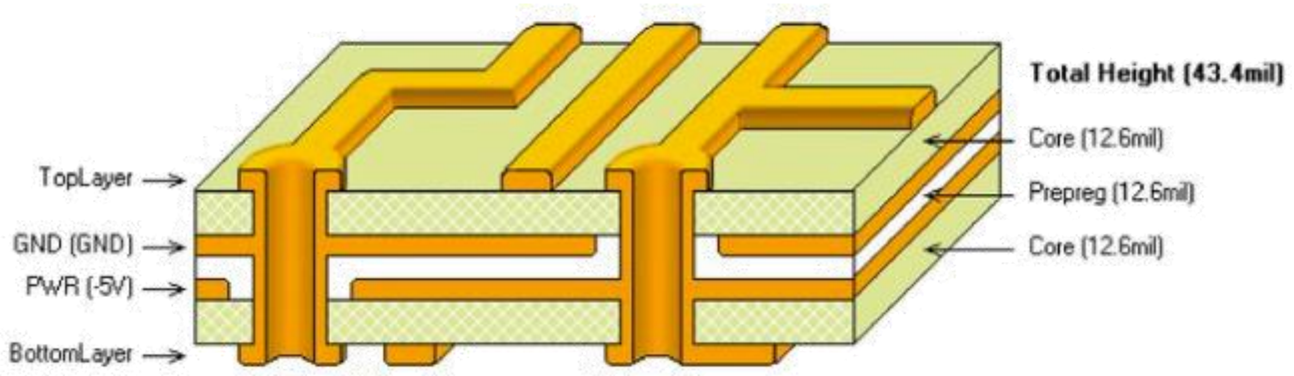
DIGITAL MULTIMETER
-1.8.8.8
IEC61010-1
2000 Counts/Autorange
FUNC.
HOLD
MAX.
Ω
Hz
µA
mA
A
~V
V
OFF
OFF
CE
DURATOOL
D03124
10A
600V CAT III
COM
VΩHz mAµA
10A/600V~ MAX
600V MAX
250mA/600V~ MAX
600V CAT III
10A MAX

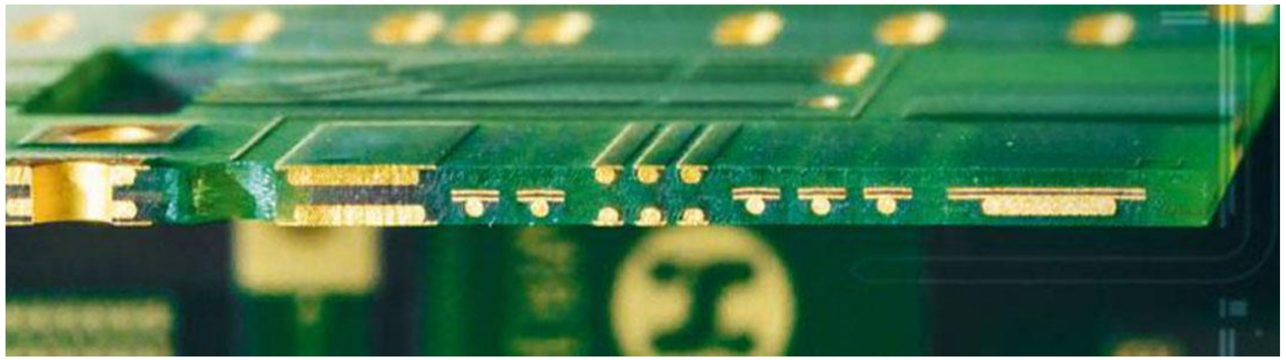



Printed Circuit Board

- A PCB mechanically supports and electrically connects electronic components using conductive tracks (*traces*) etched from copper sheets laminated onto a non-conductive substrate
- Conductors on different layers are connected with plated-through holes called *vias*
- *Pads* are where the components are soldered on the PCB





Exam rules

- The full MOD A exam is divided in a written exam + lab assignments
- Written exam
  - Exercises like the ones proposed during the lessons
  - Can be done also *in itinere* (tentatively beginning of december)
- Lab activity
  - Mandatory attendance
  - Will be done in groups ($\sim$3 students per group)
  - Continuous assessment through lab assignments
  - Optionally a small project will be proposed at the end