Embedded Programming for Beginners

Implementing an embedded application using Arduino

Session 1

Course Goals

- Objective 1
  - Understand the Arduino world and history
  - Learn Arduino internals, peripherals, project options
- Objective 2
  - Understand how to identify components for an embedded project
  - Introduction to datasheets
  - Integrate registers and bitwise operations into the mix
- Objective 3
  - Building applications with open source software
  - Building temperature reading application and integrate it with the data acquisition and the rest of kit components

Key competences

- Thinking
- Relating to others
- Using language symbols and texts
- Managing self
- Participating and collaborating

Let`s meet

- Hi, we are Alex & Nicu
- How about you?
  - Where are you from …
  - How you ended up here ...
  - Is English ok with you ...
  - How was the test …
  - What are your expectation regarding this training …
  - What are your likes/dislikes …
  - Some first date story: funny or sad …

Where to buy stuff?

- https://www.optimusdigital.ro/ro/
- https://ardushop.ro/ro/
- https://cleste.ro/
- https://www.conexelectronic.ro/ro/

conex electronic

Cleşte.ro
tehnologia te prinde

ArduShop

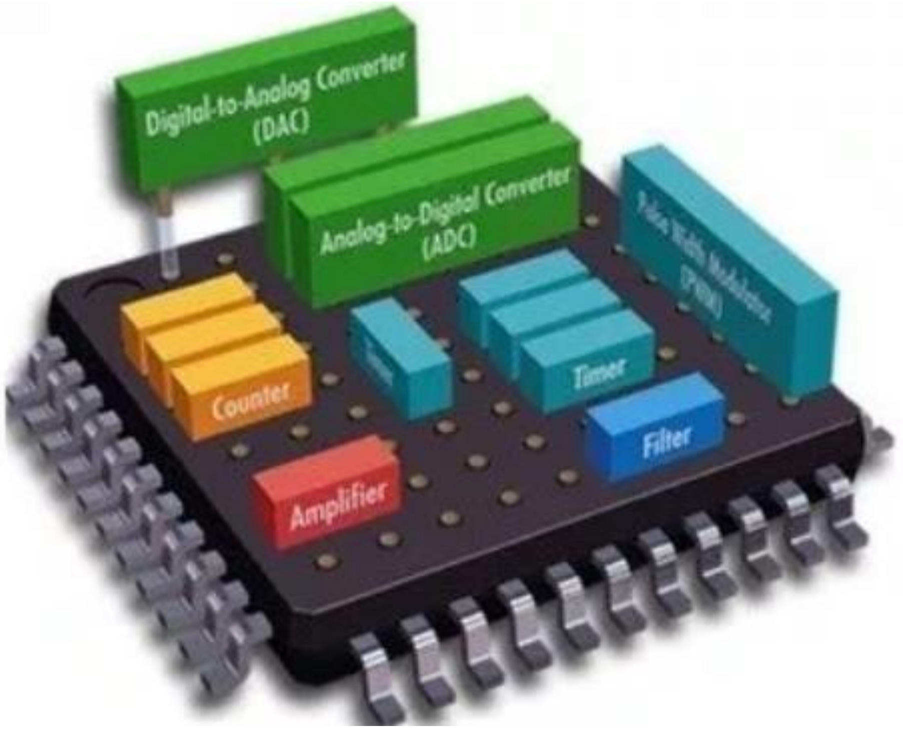
Optimus Digital

Hands-on exercise: Kit Arduino

- Try to make a list with these components:
  - Arduino Uno + Cablu / controller,
  - LED RGB catod comun, LED Monocolor,
  - Rezistor 0.25W 1KΩ, Rezistor 0.25W 4.7KΩ,
  - Buton, Buzzer Pasiv 5V, Breadboard,
  - Fire tata-tata, Fire mama-tata,
  - Condensator electrolitic 10uF,
  - Servomotor,
  - Adaptor MicroSD, Card MicroSD(HC) <= 16GB,
  - Alphanumeric LCD (16x2 characters),
  - Senzor de temperatură DS18B20.

- Make sure that you include as much of them from a single vendor
- Pay attention at the price as well!

Microcontroller


Digital-to-Analog Converter (DAC)
Analog-to-Digital Converter (ADC)
Pulse Width Modulator (PWM)
Counter
Timer
Timer
Filter
Amplifier

Microprocessor vs Microcontroller

- **Microprocessor** consists of only a Central Processing Unit, whereas **Microcontroller** contains a CPU, Memory, I/O all integrated into one chip**.**
- **Microprocessor** is used in Personal Computers whereas **Microcontroller** is used in an embedded system.
- **Microprocessor** uses an external bus to interface to RAM, ROM, and other peripherals, on the other hand, **Microcontroller** uses an internal controlling bus**.**
- **Microprocessors** are based on Von Neumann model **Microcontrollers** are based on Harvard architecture.
- **Microprocessor** is complicated and expensive, with many instructions to process but **Microcontroller** is inexpensive and straightforward with fewer instructions to process.

Microcontroller structures

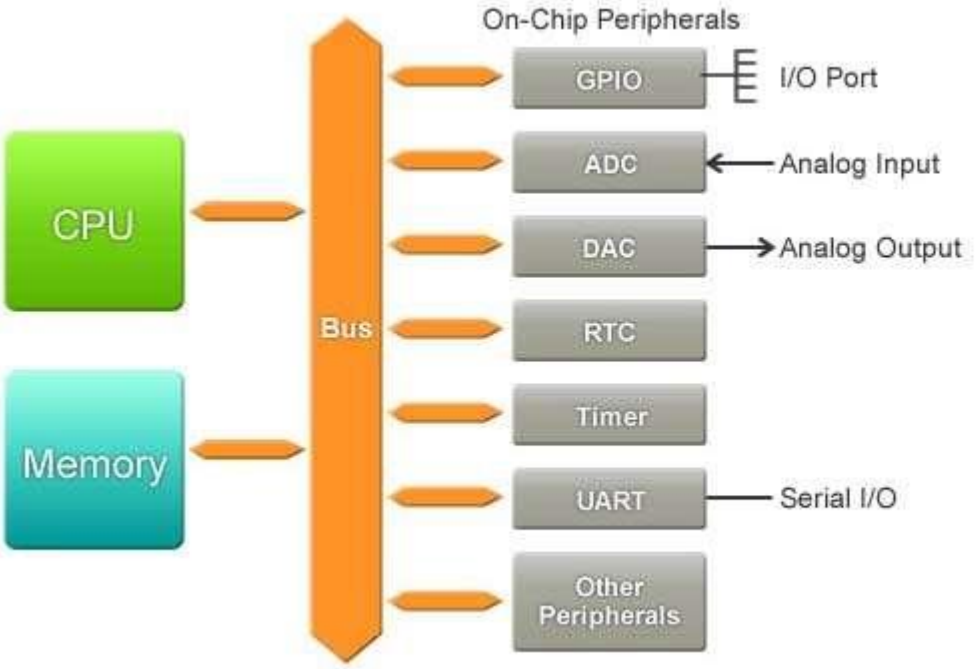
- *µP core:* 8, 16, 32 or 64 bits
- Volatile/non-volatile data memory
- Non-volatile code memory
- GPIO
- Serial communication interfaces
- Ethernet interface
- Graphical display interfaces
- Timers
- ADC & DAC
- Integrated voltage source
- Programming and debugging interfaces

Microcontroller peripherals (1)

- Power supply
- Timers
- Serial exchange ports
- Input/output circuits
- System service specific circuits

Microcontroller peripherals (2)

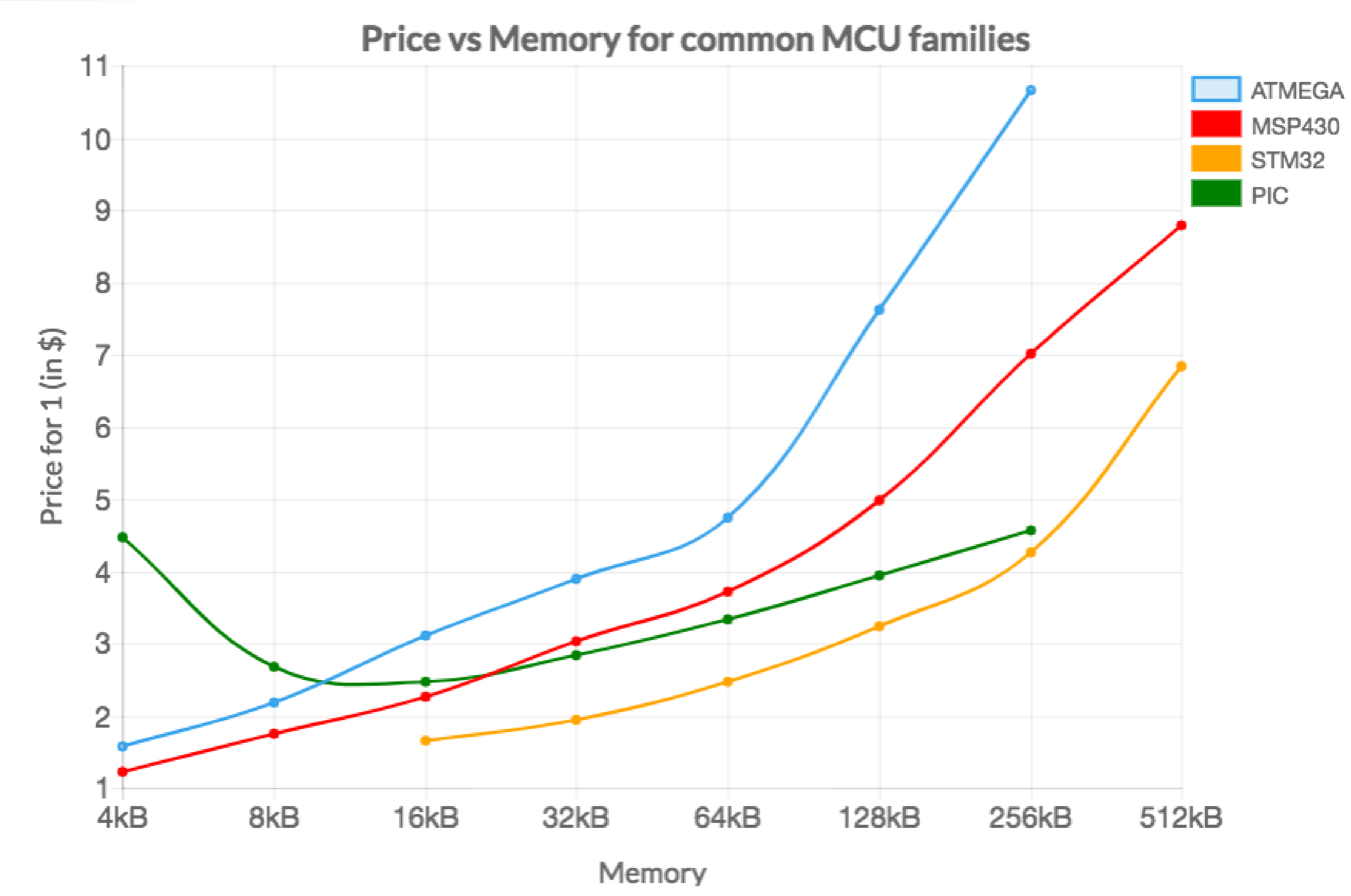
- Interfaces with the outside world
  - GPIO
  - I2C
  - SPI
  - UART
  - Timers
  - USB
  - …



Microcontroller role

- Designed for a specific challenge
- Execution of a particular operation
- Design calculation is a measure of executive functions like length, strength, price and performance.
- Attention to battery consumption
- Software is usually called firmware and is stored on the chip
- Calculates precise outcomes in actual time without any put-off
- Hardware is used for safety and performance and software is used for more flexibility and capabilities

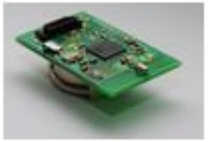
Microcontroller cost



Microcontrollers in industrial use cases

- Aircraft instruments
- Cellular phones and modern cars
- Domestic appliances: stereo, washing machine
- Other automated process

Microcontroller demo projects



- Other resources and ideas available here: https://www.hackster.io/projects/tags/microcontroller

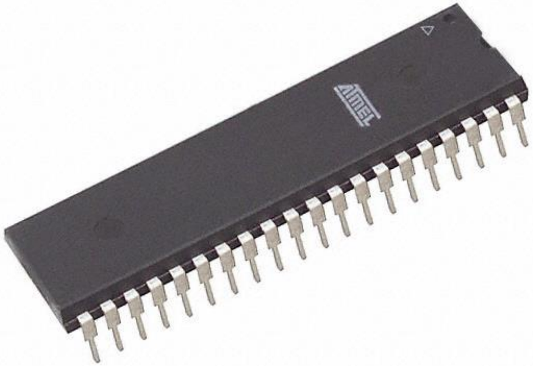
Microchip AVR family

- Harvard architecture
- 8 bits based
- RISC
- Flash, EEPROM, and SRAM on same chip
- 16 bits instructions
- 1 MIPS/MHz throughput

Microchip AVR family – comparison

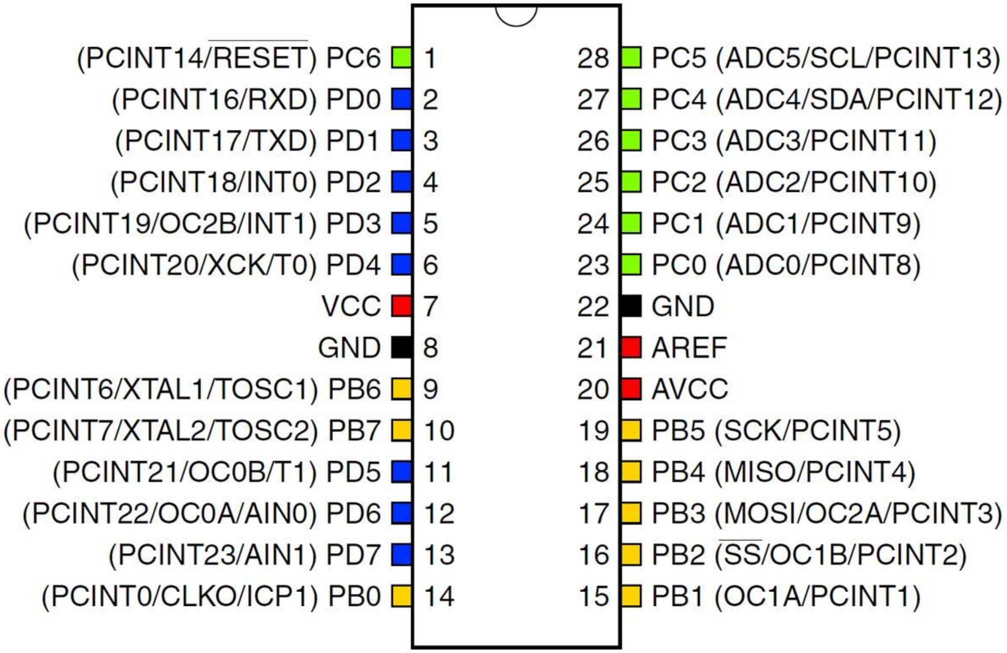
tinyAVR	megaAVR	XMEGA	Application Specific AVR
* 1-8 KB memorie de program * capsulă de 8 - 32 de pini * set limitat de periferice * 4-256 KB memorie de program	* capsulă de 28 - 100 de pini * set extins de instrucţiuni (instrucţiuni pentru înmulţire şi adresare indirectă) * set extins de periferice	* 16-256 KB memorie de program * capsulă de 44 - 100 de pini * interfeţe ca: DMA, “Event System”, şi support pentru criptografie * set extins se periferice	* megaAVR + funcţii speciale: controller de LCD, controller USB, CAN etc. * FPSLIC, un core AVR integrat cu un FPGA.

ATmega328P



- 32 KB Flash (determină dimensiunea maximă a programului care poate fi executat)
- 1 KB EEPROM
- 2 KB RAM
- 20 MHz frecvența maximă de lucru
- Tensiune de alimentare între 1.8V și 5.5 V
- 6 canale de PWM
- 8 canale de ADC, cu rezoluție de 10 biți
- 3 porturi digitale de I/O (GPIO - General Purpose I/O) , fiecare cu 7 sau 8 pini, în total 23 de pini de I/O
- 3 timere (două pe 8 biți și unul pe 16 biți)
- Interfeţe de comunicație seriale: USART, SPI, TWI
- Interfaţe de programare ISP și debug JTAG

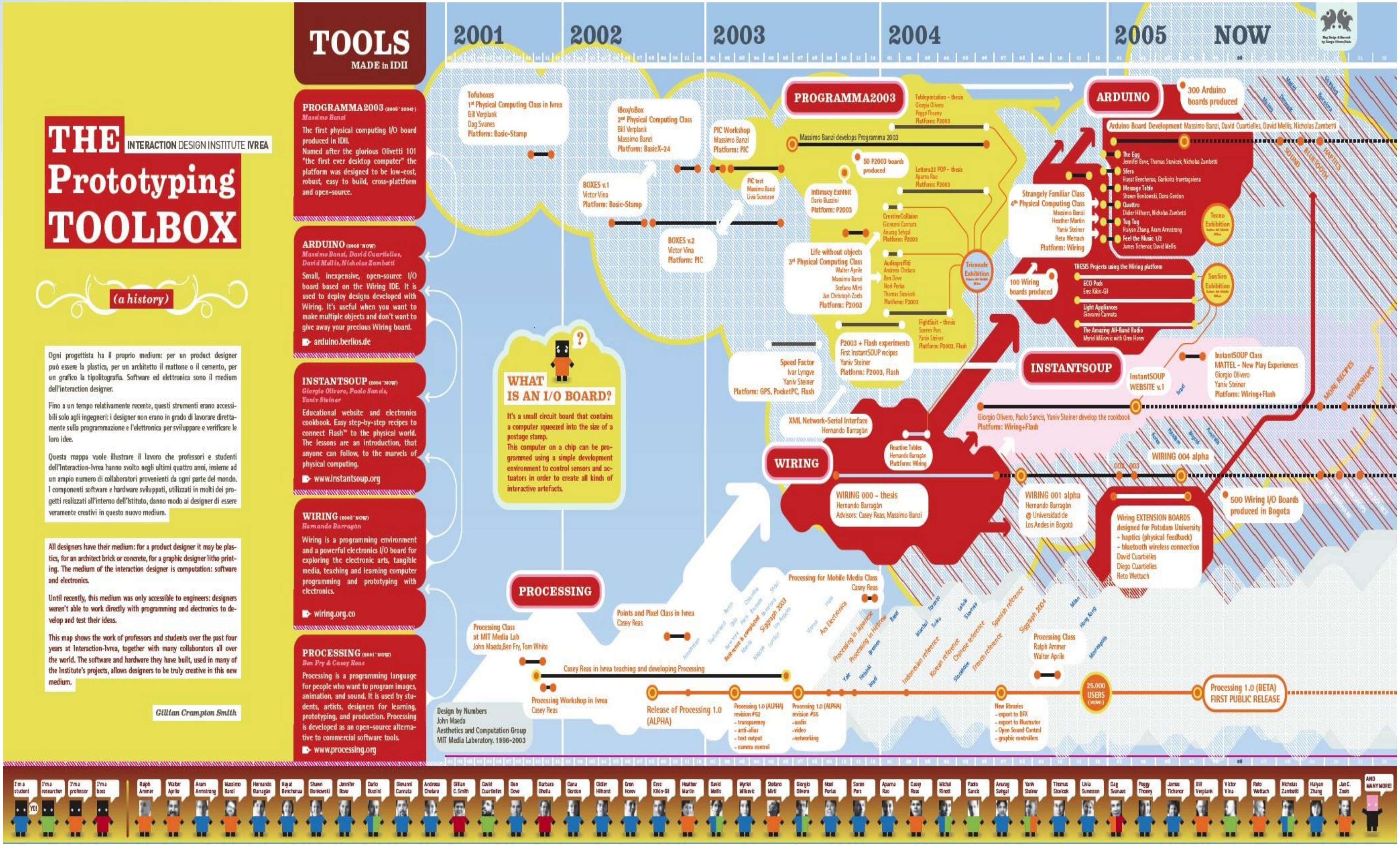
ATmega328P peripherals



Arduino history

- 2005 first Arduino board
- Interaction Design Institute Ivrea (IDII) in Italy
- Arduino – The Revolution of Open Hardware
- Purchase for aprox. US $30
- Or build from scratch
- Connected to all kind of lights, motors, sensors and other devices
- Easy-to-learn programming language available

Arduino infograf



THE INTERACTION DESIGN INSTITUTE IVREA Prototyping TOOLBOX

(a history)

Ogni progettista ha il proprio medium: per un product designer può essere la plastica, per un architetto il mattone o il cemento, per un grafico la tipolitografia. Software ed elettronica sono il medium dell'interaction designer.

Fino a un tempo relativamente recente, questi strumenti erano accessibili solo agli ingegneri: i designer non erano in grado di lavorare direttamente sulla programmazione e l'elettronica per sviluppare e verificare le loro idee.

Questa mappa vuole illustrare il lavoro che professori e studenti dell'Interaction-Ivrea hanno svolto negli ultimi quattro anni, insieme ad un ampio numero di collaboratori provenienti da ogni parte del mondo. I componenti software e hardware sviluppati, utilizzati in molti dei progetti realizzati all'interno dell'Istituto, danno modo ai designer di essere veramente creativi in questo nuovo medium.

All designers have their medium: for a product designer it may be plastics, for an architect brick or concrete, for a graphic designer litho printing. The medium of the interaction designer is computation: software and electronics.

Until recently, this medium was only accessible to engineers: designers weren't able to work directly with programming and electronics to develop and test their ideas.

This map shows the work of professors and students over the past four years at Interaction-Ivrea, together with many collaborators all over the world. The software and hardware they have built, used in many of the Institute's projects, allows designers to be truly creative in this new medium.

Gillian Crampton Smith

TOOLS MADE in IDII

PROGRAMMA2003 (2003-2004?)
Massimo Banzi

The first physical computing I/O board produced in IDII.
Named after the glorious Olivetti 101 "the first ever desktop computer" the platform was designed to be low-cost, robust, easy to build, cross-plattform and open-source.

ARDUINO (2005-NOW)
Massimo Banzi, David Cuartielles, David Mellis, Nicholas Zambetti

Small, inexpensive, open-source I/O board based on the Wiring IDE. It is used to deploy designs developed with Wiring. It's useful when you want to make multiple objects and don't want to give away your precious Wiring board.

arduino.berlios.de

INSTANTSOUP (2004-NOW)
Giorgio Olivero, Paolo Sancis, Yaniv Steiner

Educational website and electronics cookbook. Easy step-by-step recipes to connect Flash™ to the physical world. The lessons are an introduction, that anyone can follow, to the marvels of physical computing.

www.instantsoup.org

WIRING (2003-NOW)
Hernando Barragán

Wiring is a programming environment and a powerful electronics I/O board for exploring the electronic arts, tangible media, teaching and learning computer programming and prototyping with electronics.

wiring.org.co

PROCESSING (2001-NOW)
Ben Fry & Casey Reas

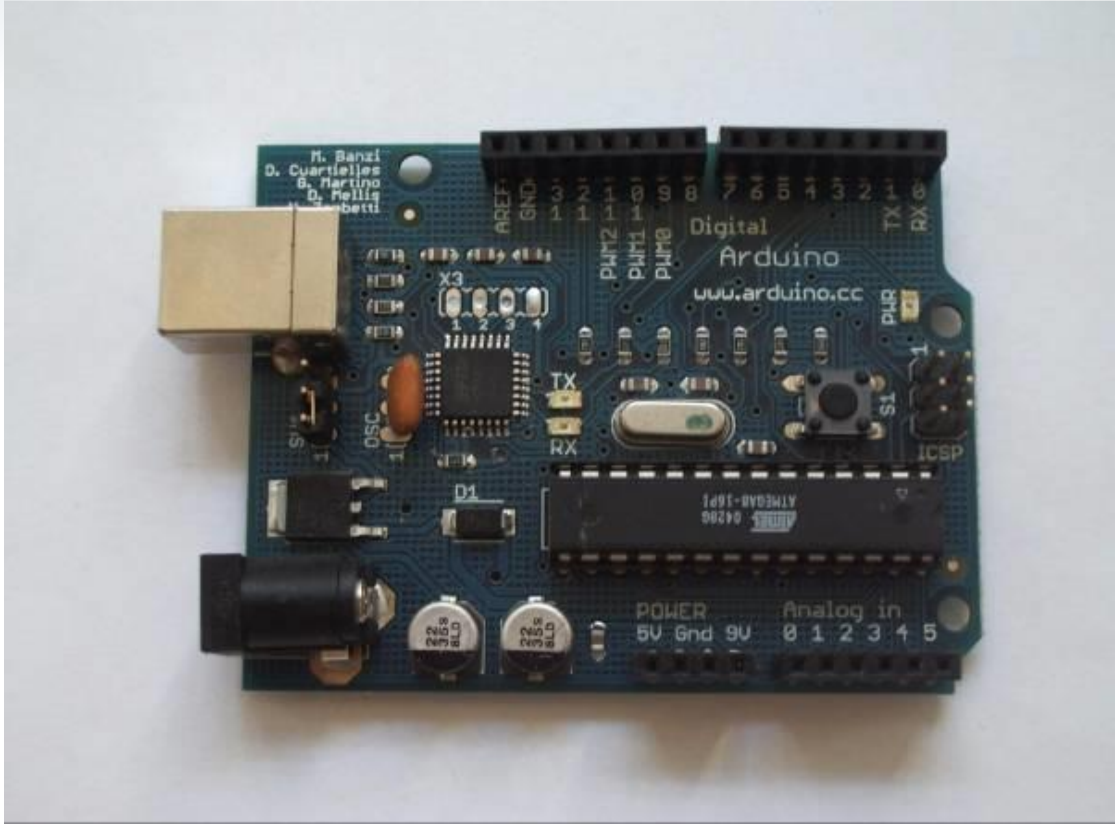
Processing is a programming language for people who want to program images, animation, and sound. It is used by students, artists, designers for learning, prototyping, and production. Processing is developed as an open-source alternative to commercial software tools.

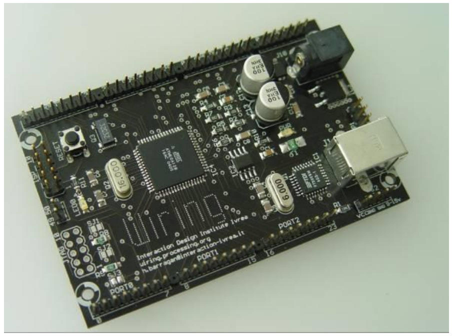
www.processing.org

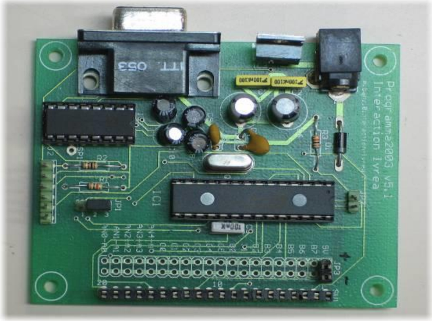
WHAT IS AN I/O BOARD?

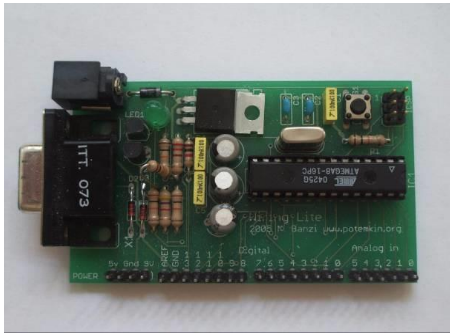
It's a small circuit board that contains a computer squeezed into the size of a postage stamp.
This computer on a chip can be programmed using a simple development environment to control sensors and actuators in order to create all kinds of interactive artefacts.

Why so successful?

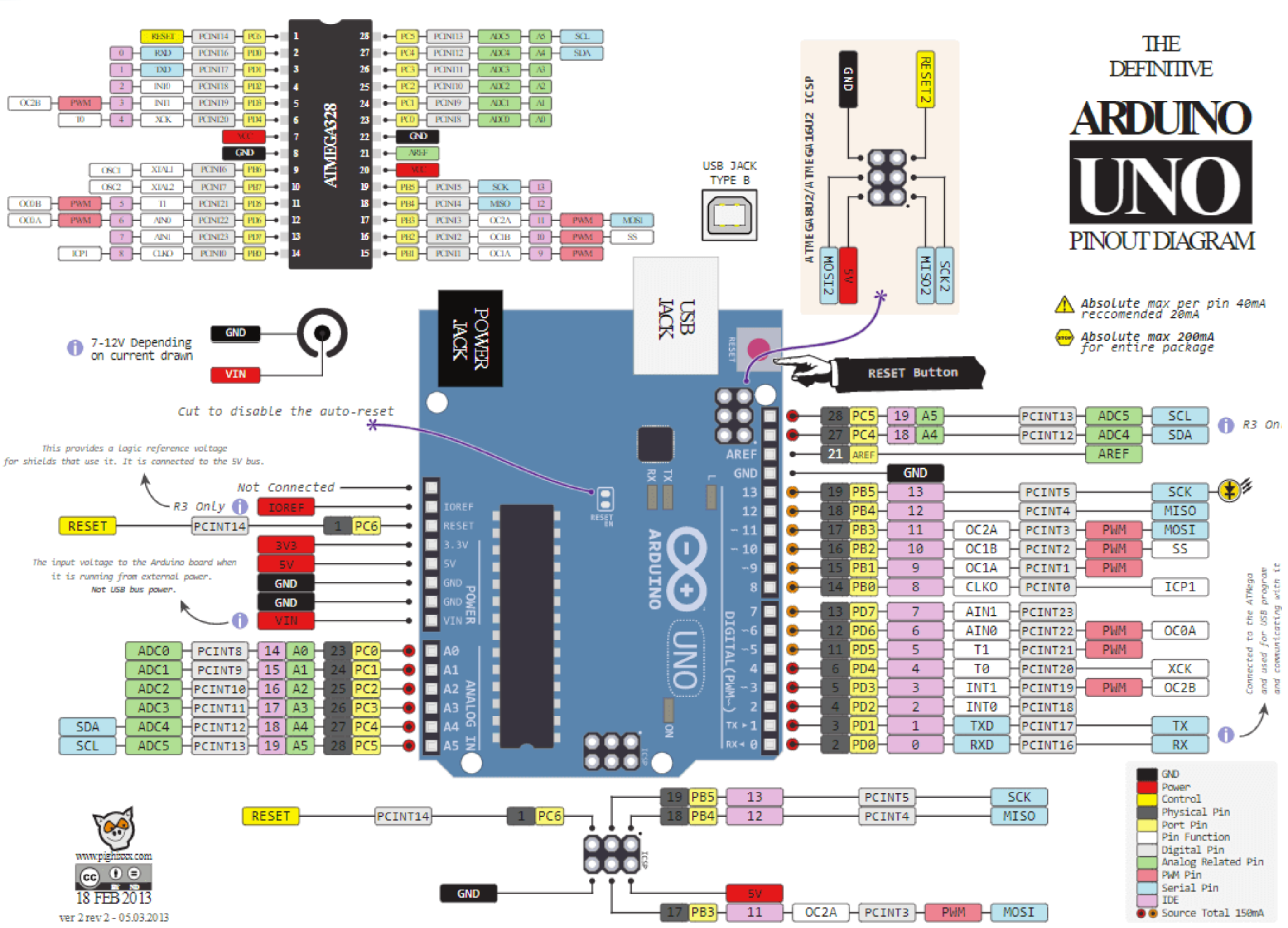








Arduino hardware



Arduino software

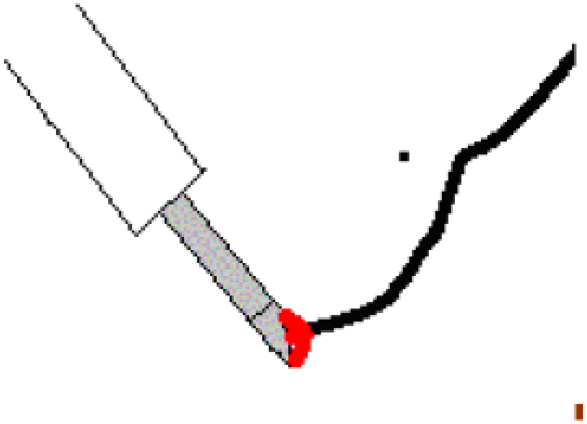
- Any programming language, but C/C++ is preferred
- Arduino IDE, written in Java
- setup(): run only once, used for initializing settings
- loop(): run repeatedly until the power supply is turned off
- GNU toolchain used
- avrdude: converts executable into hexadecimal for upload

```
avrdude -p atmega328p -D -Uflash:w:"main.hex" -v -v -F -c avrisp -P /dev/ttyACM0
```

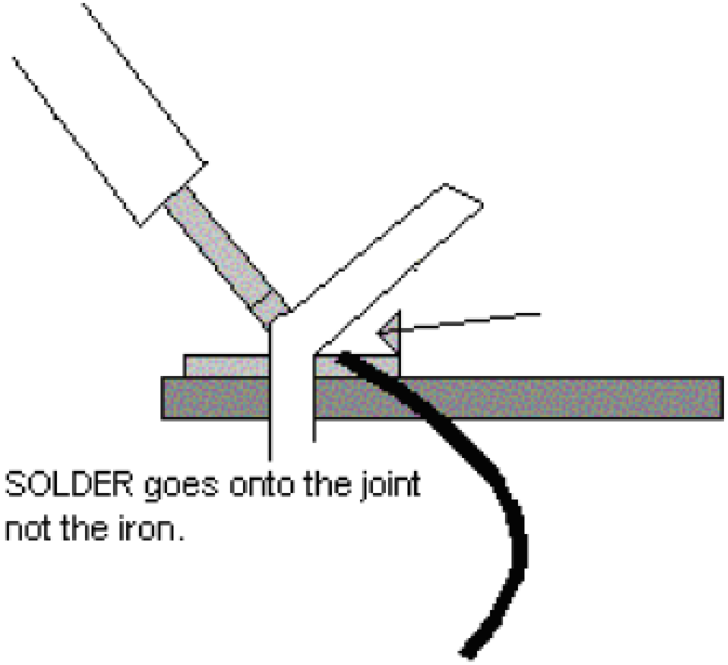
Soldering

- 1. The materials must be clean
- 2. Wipe clean the iron on a moist sponge (the sponge must not be dripping wet!)
- 3. The iron must be tinned with a small amount of solder
- 4. Put the tinned iron onto the joint to heat the joint first
- 5. The joint must be heated (be aware that excessive heat can ruin boards and components)
- 6. Apply the solder to the joint near the soldering iron but not onto the iron itself
- 7. Use enough solder so the solder flows thoroughly around the joint- it takes time for the solder to siphon or capillary around all the gaps
- 8. Remove the solder
- 9. Keep the iron on the joint after the solder for an instant
- 10. Remove the soldering iron last – do not clean the iron, the solder left on it will protect it from oxidizing
- 11. Support the joint while it cools (do not cool it by blowing on it)
- **DO NOT - DO NOT - DO NOT - DO NOT** repeatedly touch and remove the soldering iron on a joint this will never heat the joint properly, HOLD the iron onto the joint until both parts of it COMPLETELY heat through

Soldering technique

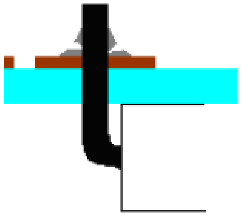


Tin the end of the iron with solder

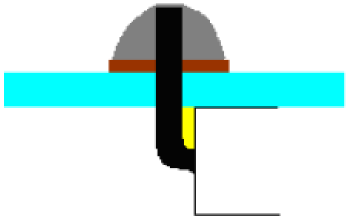


SOLDER goes onto the joint not the iron.

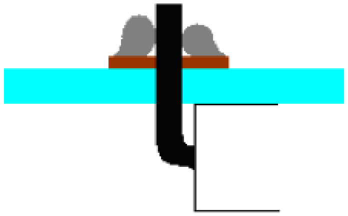
Good and bad solder joints



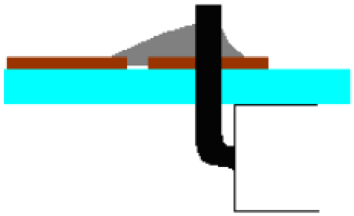
Too little solder, not enough heat



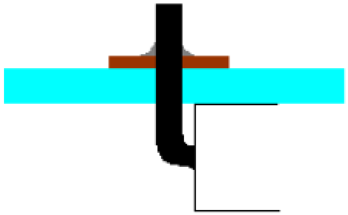
Too much solder



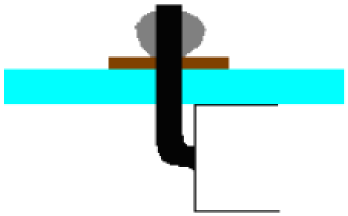
Heated only the pcb track



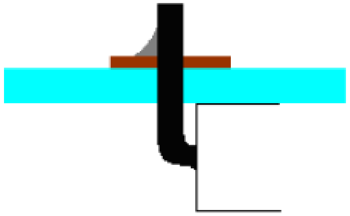
Too much solder,
it has flowed onto another track



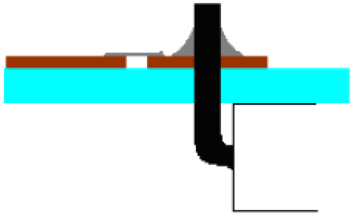
Too little solder



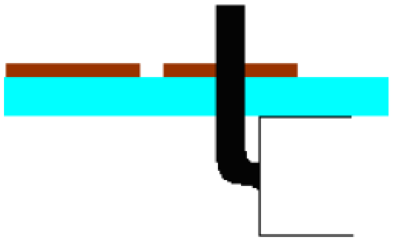
Heated only the leg of the component



Only soldered on one side of the leg

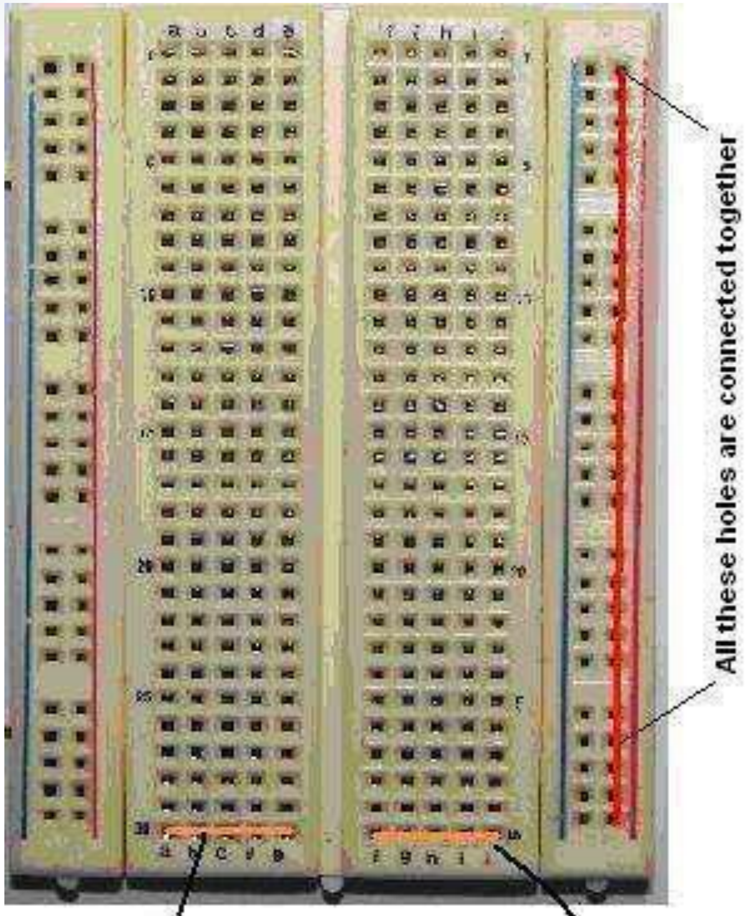


A whisker of solder is touching another track



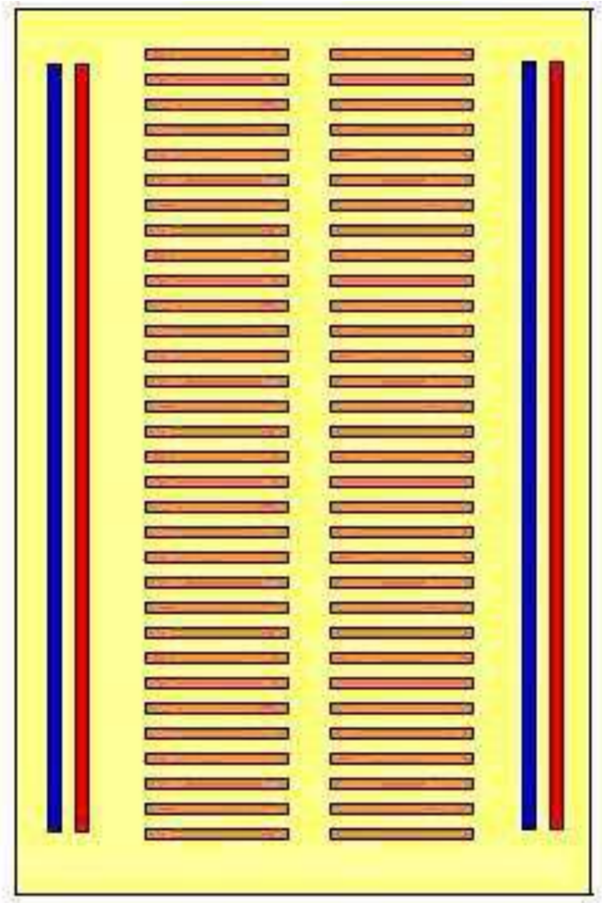
Forgot to solder it!!

Breadboards



All these holes are connected together

These 5 holes are connected, but they are separate to the next 5 holes



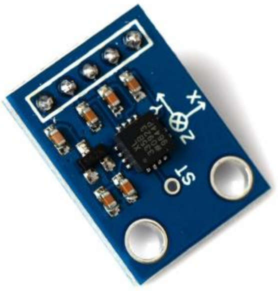
Here are all of the connections on the board

Actuators

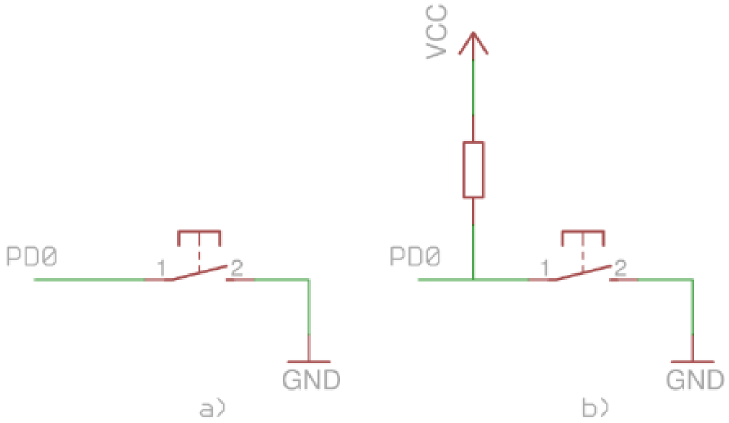
- LED (light)
- Servo motor (motion)
- Speaker (sound)
- DC motor (motion, rotation)
- Solenoid (motion)

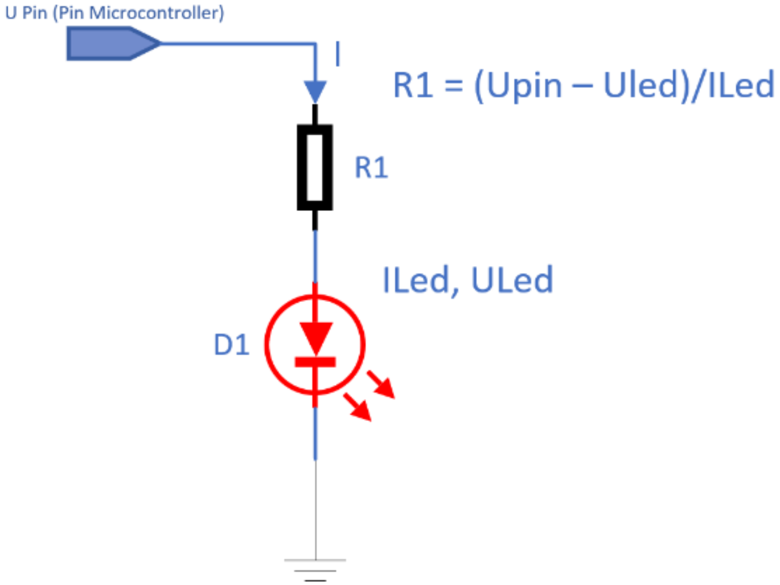
Traductors

- **Digital**
  Push button
  Tilt sensor
  Hall Effect sensor
- PIR motion sensor
  pulse sensor
- **Analog**
  LDR
  Potentiometer
  Flex sensor
  soft potty
- piezo sensor
  Microphone
- muscle sensor
  Distance Sensor
  Ultra Sonic distance sensor
  3 axis Accelerometer
- Gyroscope

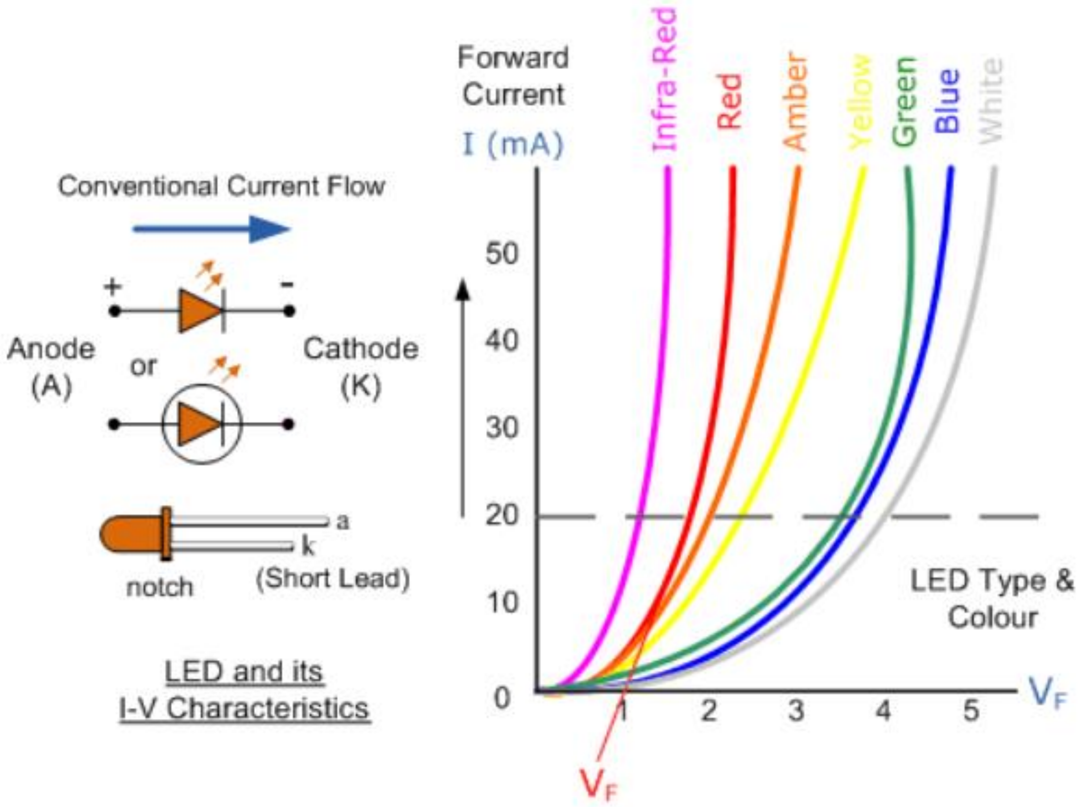


LED, Resistors, Buttons





LEDs Characteristics



Registers operations

Operation	Formula
Write bit on 1	register |= (1 << bit_index)
Write bit on 0	register &= ~(1 << bit_index)
Toggle bit	register ^= (1 << bit_index)
Read bit	register & (1 << bit_index)

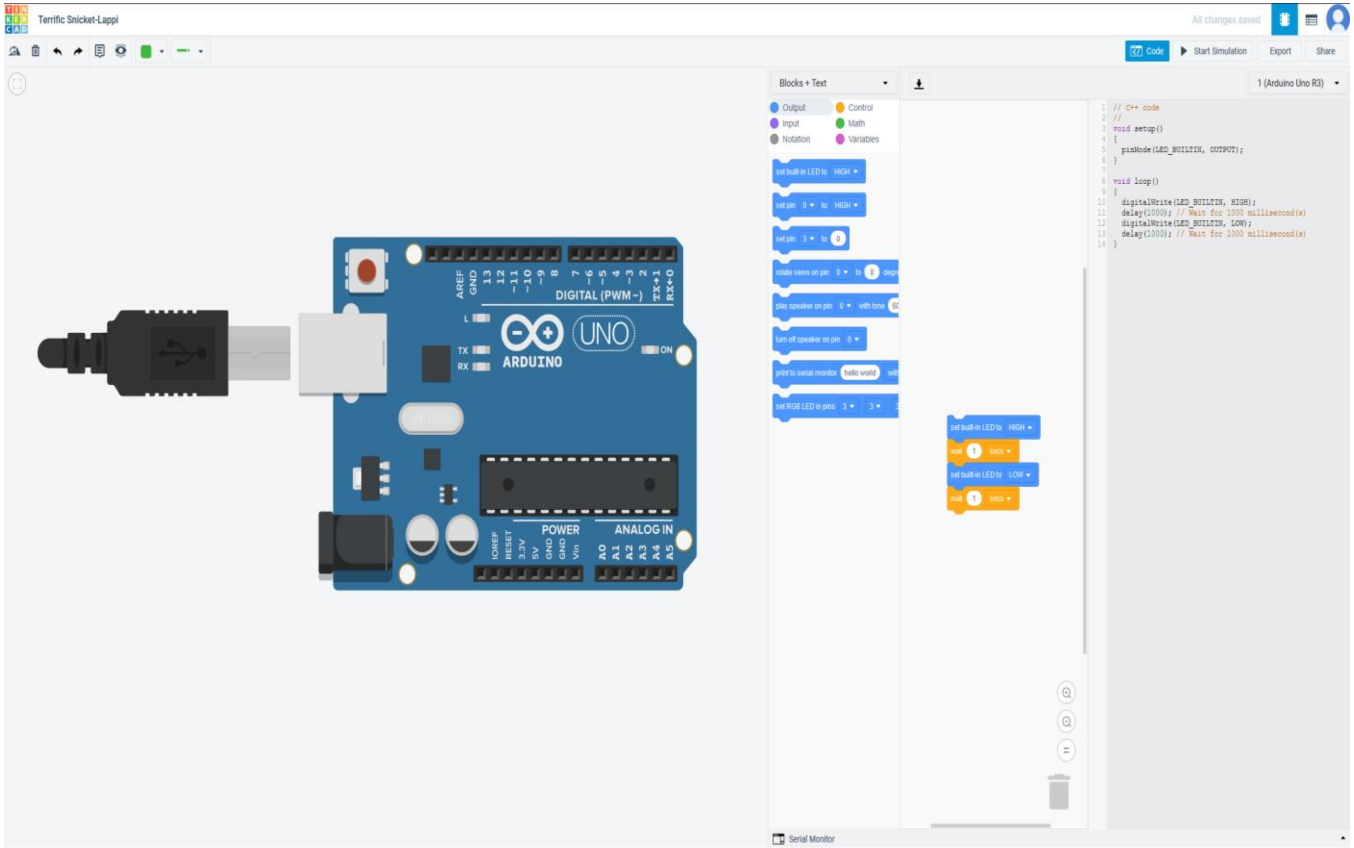
Hands-on exercise: Registers

- Let's assume we have an LED connected to Pin 1 of Port B called PORTB1 or PB1.
  - Try switching on and off the LED?

- Let's assume we have a button connected to Pin 4 of Port D called PORTD4 or PD4
  - Try determine the state of the button: pressed or not

Arduino simulator

- Tinkercad
- 3D modeling
- Arduino UNO simulation



```
// C++ code
//
void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop()
{
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

Hands-on exercise: Hello world

- On your workstation install the Arduino IDE
- Configure the IDE accordingly for the board interaction
- Write your first Hello World application

- Open Tinkercad: https://www.tinkercad.com
- Create a user account
- Reproduce the hardware setup in the simulator

Closing remarks

- Microcontroller vs microprocessor
- AVR family
- Arduino history
- Soldering
- Actuators and traductors
- Arduino registers
- Tinkercad

- Q&A