Lab avanzato elettronica

Michela Greco,

michela.greco@unito.it

(3rd floor, NE, #7476)

Introduction

Taking into account your answers, I will lecture an «Arduino» base-level course for almost everybody.

Some do have Arduino board, most do not have. It is not a problem, that is we will handle using **Tinkercad** which lets you simulate Arduino.

7 «Webex lessons»
Me 27/5 16-18
Gio 28/5 14-16
Ve 29/5 14-17
Lu 1/6 14-17
Me 3/6 16-18
Gio 4/6 14-16
Ve 5/6 14-17

Exam

- Carry out a design among projects I will indicate to you starting “from scratch” or modify one of the project I will introduce.

What does it mean?
-”mount” hw virtually or physically
-programming
-being asked about hw (principle of operation, technical specs)
and sw (Arduino code and related libraries)

1 h long oral on Webex
Weight: 1/3 of total score

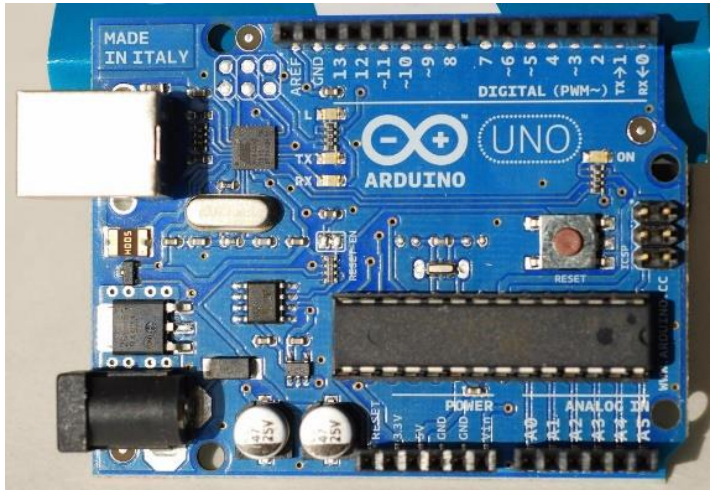
- Reference material: folder_students_arduino on github &googledrive

Where Arduino was patented?

Quote

> *Ciao*
>
> *Si chiama cosi perche quando l'abbiamo inventato eravamo ad ivrea dove tutto si chiama o Arduino oppure Olivetti....*
>
> *Avevamo 5 minuti prima di spedire il circuito in produzione e non avevamo ancora un nome... Ad un certo punto ho detto "chiamiamolo Arduino come il bar dove andiamo all'aperitivo, poi possiamo sempre cambiarlo dopo...."*
>
> *Alla fine è andata bene così*
>
> *Massimo*

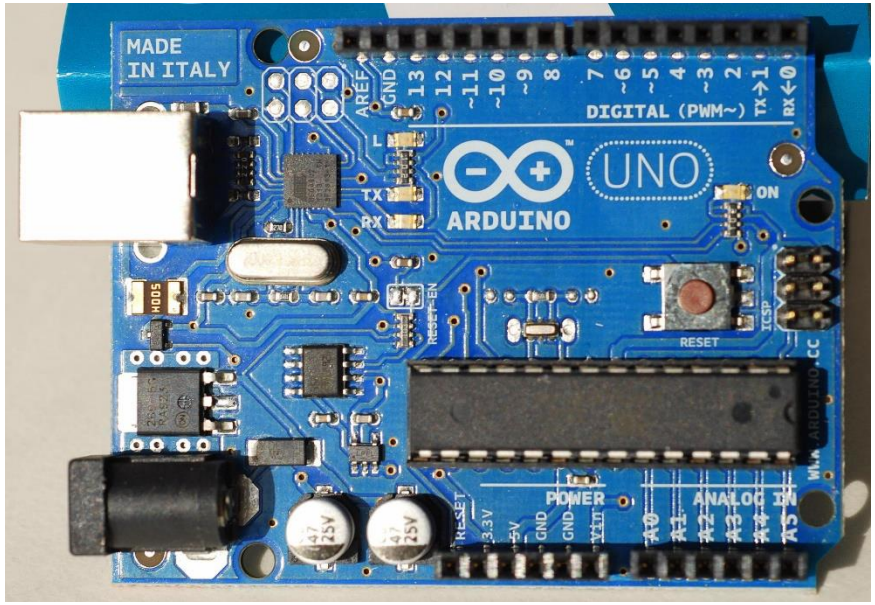
How did it spread?



What is Arduino?

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.

da Arduino.cc



Main features

-microcontrollor
-analogic and digital I/O pins
-Flash memory
-USB port for serial communication

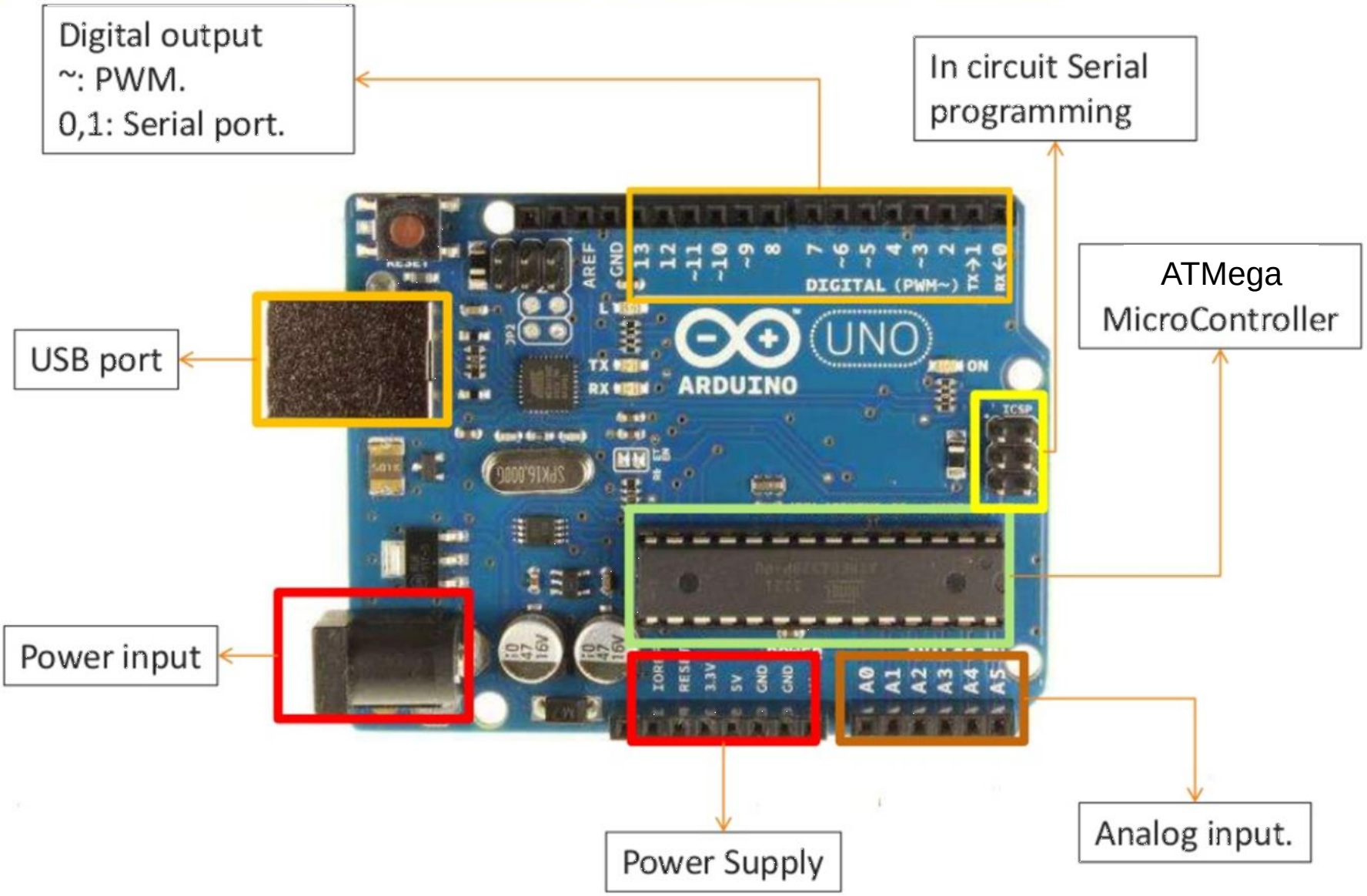
PDIP Atmega 328 microcontroller

Quartz clock , 16 MHz

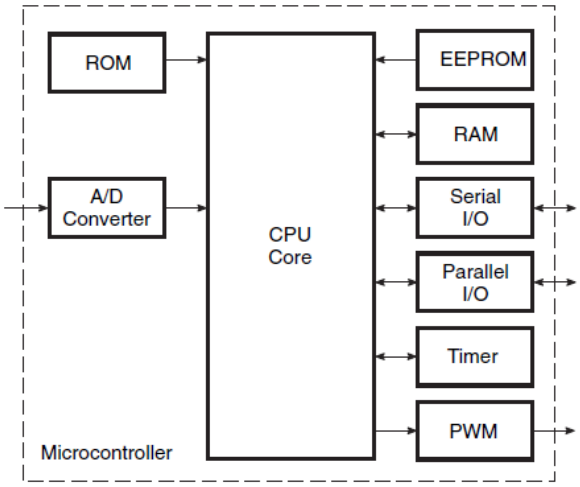
Main functions

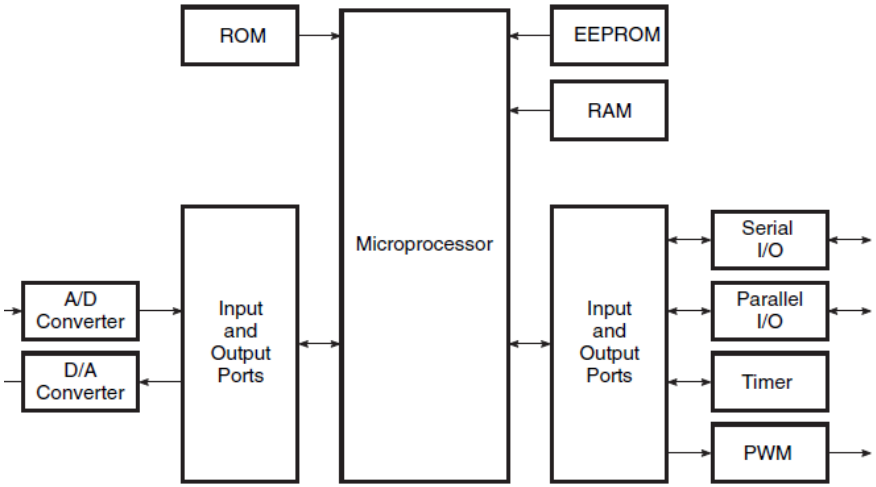
-Read out sensors
- Control peripheries
- Serial communication
- Program

What is Arduino?

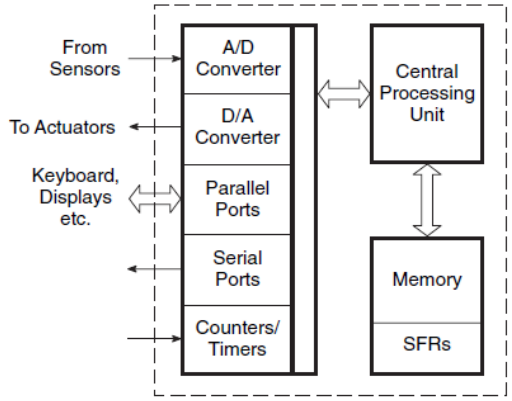


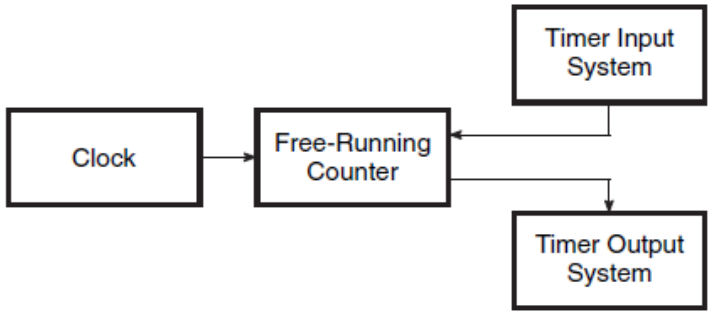
What is a microcontroller?





The microcontroller contains a simplified processor, some memory (RAM and ROM), I/O ports and peripheral devices such as counters/timers, analogue-to-digital converters, etc., **all integrated on a single chip.** This is the main difference wrt microprocessors



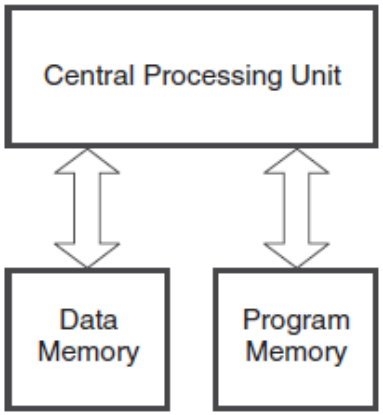


What is a microcontroller?

The microcontroller can be classified according to NUMBER OF BITS, MEMORY (external or embedded) and on INSTRUCTION SET:

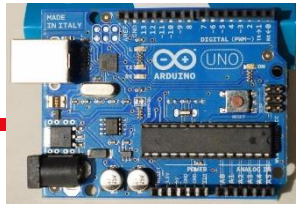
- CISC is referred as “complex instruction set computer”. One valid instruction is enough to replace number of instructions.
- RISC is referred as “reduced instruction set computer”. RISC helps in reducing the operation time of executing the program. It does it by reducing the clock cycle per instruction.

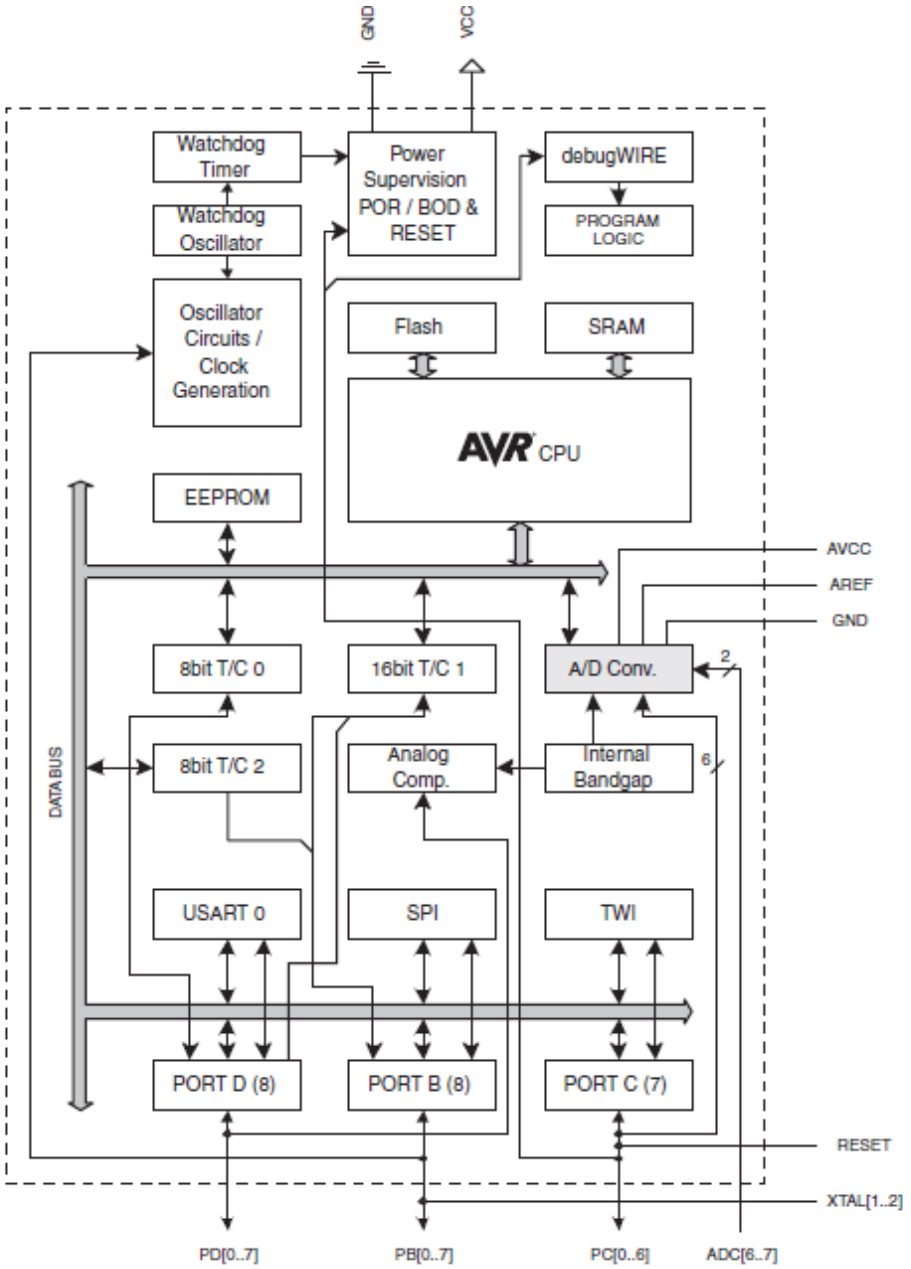
AVR is referred as Advanced Virtual RISC which was produced by Atmel in 1966.
It supports **Harvard Architecture** in which program and data is stored in different spaces of microcontroller and can easily be accessed.
It is considered as earlier types of controllers in which on-chip flash is used for storing program.



In Harvard architecture, instructions and operands can be fetched simultaneously

Arduino: hw





Micocontrollor Atmega 328:

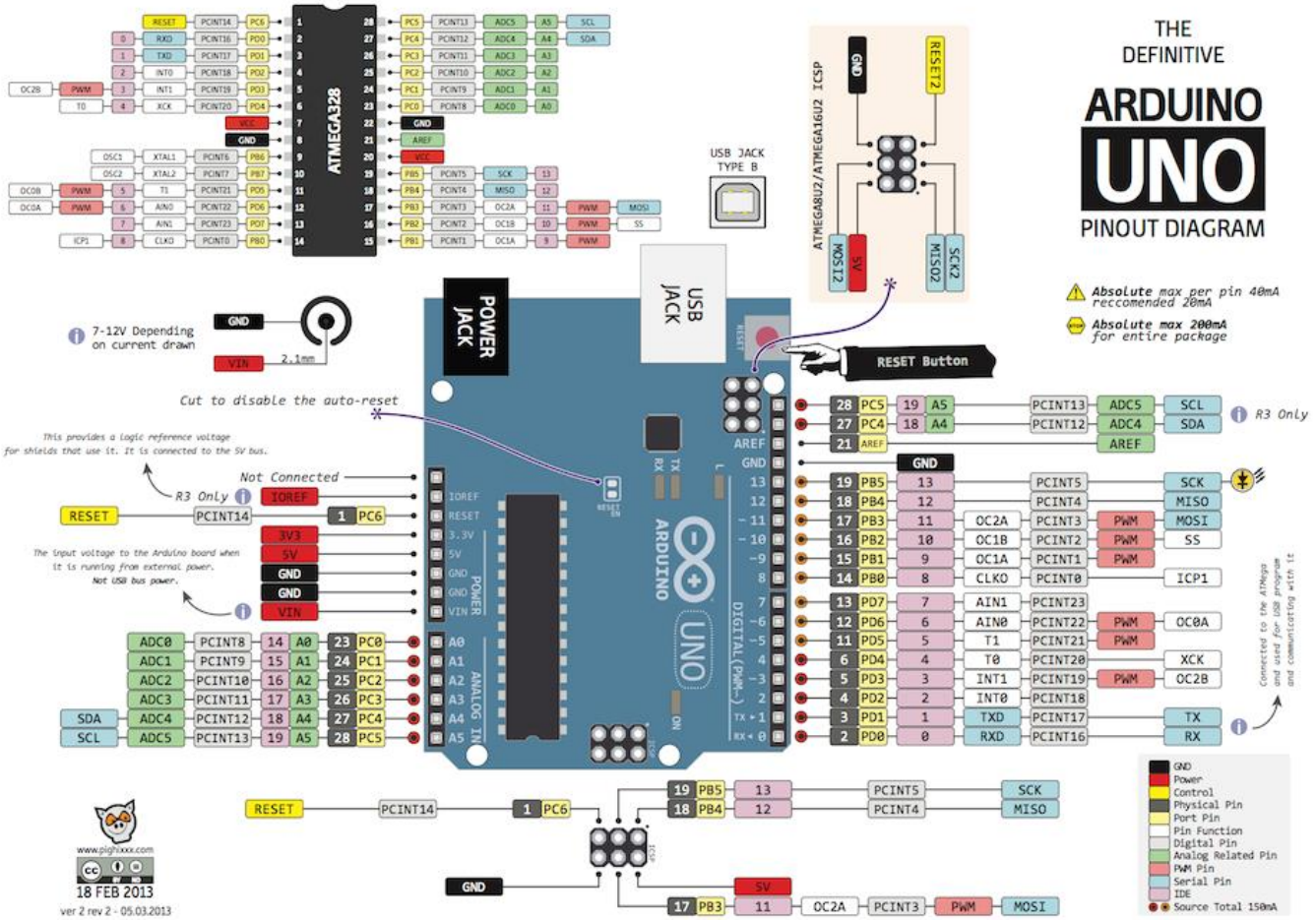
Components:

- CPU
- Memory ROM, EPROM, FLASH
- Data memory: RAM, EEPROM
- Quartz oscillator
- I/O ports
- ADC, DAC, counters, timer...

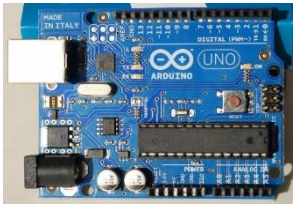
In the folder,
you find DATASHEETs
These are important to be read (and known in some part for the exam)

Arduino: hw

Name	Processor	Operating Voltage/Input Voltage	CPU Speed	Analog In/Out	Digital IO/PWM	EEPROM [KB]	SRAM [KB]	Flash [KB]	USB	UART
Uno	ATmega328	5 V/7-12 V	16MHz	6/0	14/6	1	2	32	Regular	1



Arduino: hw



I/O pins operating at +5 V –max current on pins: 40 mA

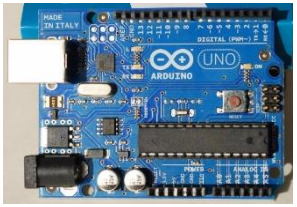
Pin 13 LED : the only actuator built-in to your Arduino Uno
AREF: logical ref value; if you use shields →5V
Reset

6 pins: analogic pins
→each 1024 values (10 bit ADC)
14 pins: digital I/O
→6 of them give 8 bit output with **Pulse Width Modulation (PWM) technique**

Pins for communication
Asynchronous Serial: 0 (RX) and 1 (TX)
SPI (serial peripheral interface, synchronous): 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK)
TWI (two wire serial interface), I^2C (Interintegrated Circuit)
A4 →SDA pin
A5 →SCL pin

→custom libraries(Wire, SPI..)

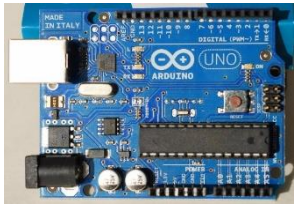
Arduino: 10 ways to destroy an Arduino



1) Shorting I/O Pins to Ground
2) Shorting I/O Pins to Each Other
3) Apply Overvoltage to I/O Pins
4) *Apply External Vin Power Backwards*
5) *Apply >5V to the 5V Connector Pin*
6) *Apply >3.3V to the 3.3V Connector Pin*
7) *Short Vin to GND*
8) *Apply 5V External Power with Vin Load.*
9) *Apply >13V to the Reset Pin*
10) Exceed Total Microcontroller Current

http://www.ruggedcircuits.com/10-ways-to-destroy-an-arduino

Arduino: sw



Arduino is programmable
Custom integrated development environment (IDE) : Java based multiplatform application

- C like Syntax
- Derived and varied wrt **Processing** (https://processing.org/)

A program is made of (only) two parts!

setup() and loop()
which need to be declared
setup(): runs once, when the Arduino is first powered on→ configuration and assignments
loop(): runs continuously after the setup() has completed. It is where you have to implement really your program

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

https://www.arduino.cc/reference/en
http://www.arduino.cc/en/Reference/Libraries

Arduino coding

Data types and operators

Integers
Character
Long
Floating Point

Statement and operators

Statement represents a command, it ends with «;»
Operators are symbols used to indicate a specific function:
Math [+,-,*,/,%,^]
Logic [==,!=,&&,||]
Comparison [=,>,<,!=,<=,>=]
Compound [++,--,+=,-=,*=,/=]

Control statements

If conditioning
Switch case

Arduino: per iniziare

Starter kit

sw :
https://www.arduino.cc/en/Main/Software✓

1.Double-click the Arduino application to open it.
2.Navigate to the LED blink example sketch ('sketch' is what Arduino programs are called). FILE > EXAMPLES > 01.BASICS > BLINK
3.select your board under TOOLS > BOARD menu
4.Choose the serial port your Arduino is connected to from the TOOLS > SERIAL PORT menu.
5.To upload the Blink sketch to your Arduino, press the UPLOAD toggle in the top left corner of the window
You should see a bar indicating the progress of the upload near the lower left corner of the Arduino IDE, and the lights labeled TX and RX on the Arduino board will be blinking. If the upload is successful, the IDE will display the message DONE UPLOADING.

A few seconds after the upload has completed, you should see the yellow LED with an L next to it start blinking

Arduino-Starter Kit projects

(1.get_to_know_your_tools)

2.myfirstcode

3.analog_sensors

4.color_mixing_lamp

5.servo_motor

6.light_theremin

7.keyboard_instrument

8.digital_6min_hourglass

9.motors

10.hbridge

11.LCD

12.knock_lock

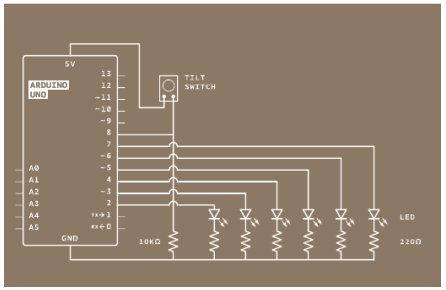
13.touchy-feely_lamp

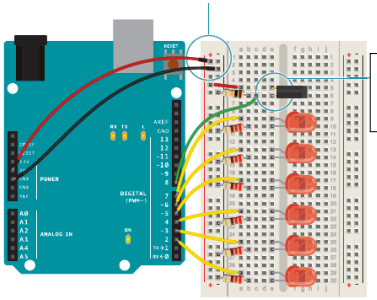
14.tweak_arduino_logo

15.optocouplers

8.digital 6 min hourglass

The **millis()** function keeps track of the of the time your Arduino has been running in milliseconds. The variable is unsigned long, so it can keep memory of about 50 days.
In this project you will build a 6 min digital hourglass.
The tilt switch works just like a regular switch in that it is an on/off sensor. You'll use it here as a digital input. What makes tilt switches unique is that they detect orientation. Typically they have a small cavity inside the housing that has a metal ball. When tilted in the proper way, the ball rolls to one side of the cavity and connects the two leads that are in your breadboard, closing the switch.
With six LEDs, your hourglass will run for 6 min, just as its name implies.

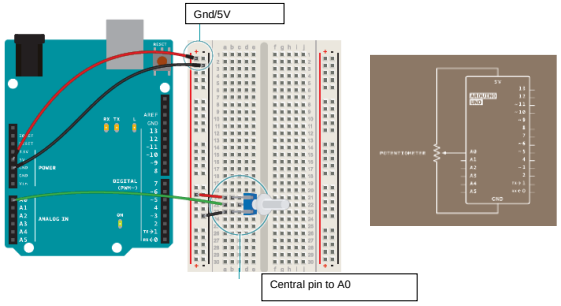




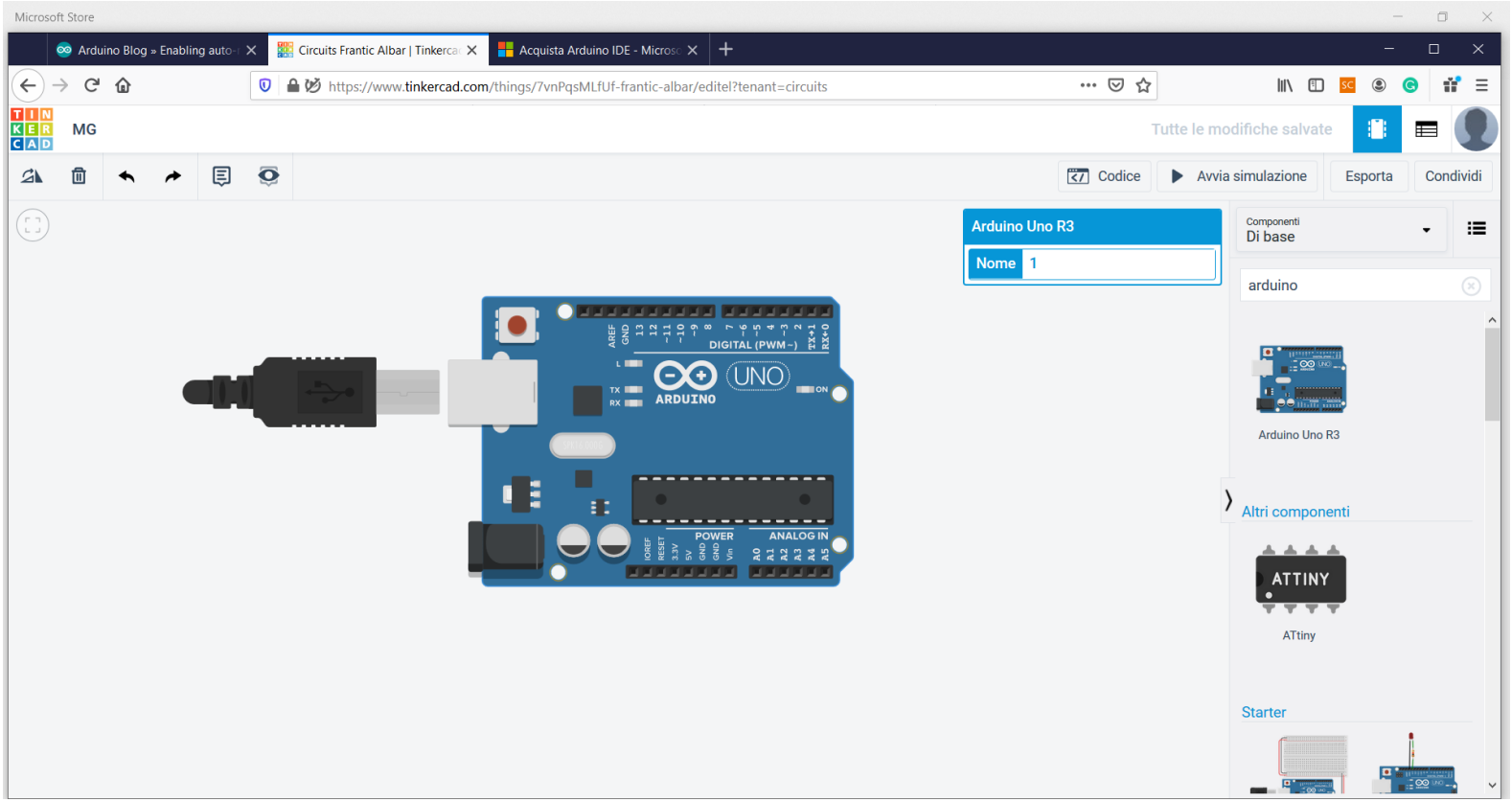
Connect one lead of the tilt switch to 5V. Connect the other to a 10-kilohm resistor to ground. Connect the junction where they meet to digital pin 8.

14. Tweak Arduino logo

Using serial communication, you'll use your Arduino to control a program on your computer.
When you program your Arduino, you're opening a connection between the computer and the microcontroller. You can use this connection to send data back and forth to other applications. The Arduino has a chip that converts the computer's USB-based communication to the serial communication the Arduino uses. Serial communication means that the two computers, your Arduino and PC, are exchanging bits of information serially, or one after another in time.
When communicating serially, computers need to agree on the speed at which they talk to one another. You've probably noticed when using the serial monitor there's a number at the bottom right corner of the window. That number, 9600 bits per second, or baud, is the same as the value you've declared using Serial.begin(). That's the speed at which the Arduino and computer exchange data.
You've used the serial monitor to look at values from the analog inputs; you'll use a similar method to get values into a program you're going to write in a programming environment called Processing. Processing is based on Java, and Arduino's programming environment is based on Processing's. They look pretty similar, so you should feel right at home there.
The most efficient way to send data between the Arduino and Processing is by using the Serial.write() function in Arduino. It's similar to the Serial.print() function you've been using in that it sends information to an attached computer, but instead of sending human readable information like numbers and letters, it sends values between 0-255 as raw bytes. This limits the values that the Arduino can send, but allows for quick transmission of information.
On both your computer and Arduino, there's something called the serial buffer which holds onto information until it is read by a program. You'll be sending bytes from the Arduino to Processing's serial buffer. Processing will then read the bytes out of the buffer. As the program reads information from the buffer, it clears space for more.
When using serial communication between devices and programs, it's important that both sides not only know how fast they will be communicating, but also agree on what is sent and received.



Tinkercad



https://www.tinkercad.com/