Arduino Notebook

Lucas Aguilera

Started on july 13 of 2025

Nota/Note

This document was made by me and is meant to be used by me. That means that this document may include: words in spanish, bad grammar, personal thoughts, curse words and dumb/unnecesary comments. Remember, learning sucks if you don't have fun while you learn, so please enjoy the process, this is NOT a college course, this is YOU teaching YOURSELF something cool (and maybe kinda childish but whatever).

1. Introduction

An Arduino is a micro controller... what is that? It's like a tiny computer. But SBCs are also tiny computers so what's the deal? Well, embedded systems (informatics systems designed to do something very specific inside of a bigger product) usually use micro controllers and single board computers for managing this specific task, but micro controllers are like a 'lite' version of the latter. Think of it like this: Micro controllers are small, usually low power devices optimized for specific tasks and real time control, while SBCs are more powerful, versatile computers that can run full operating systems.

If you care, we could look at the parts of a micro controller: integrated circuit that combines a processor core, memory (RAM and ROM of course), and peripherals. Look, here is an image of an Arduino:

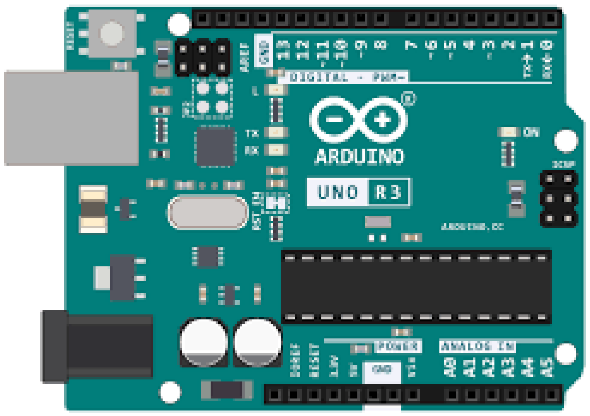

Figura 1: Looks cute

Now lets look at a SBC: a raspberry pi. See how the circuit board has a processor, memory, storage, and more interfaces? That is because its more powerful, and often used as a general purpose computer that can handle a wide range of tasks.

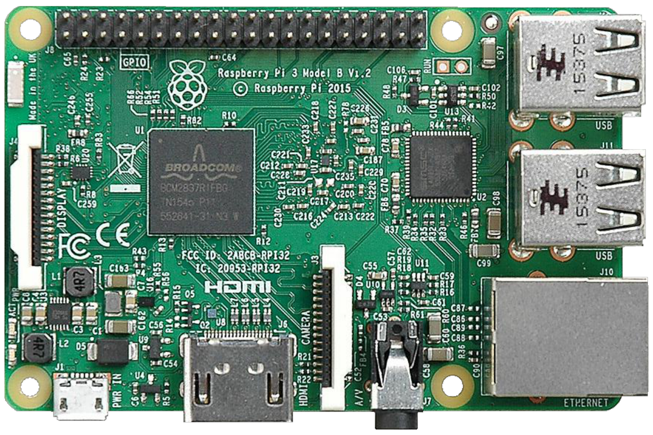

Figura 2: Looks like hell

So, micro controllers are very good with repetitive tasks and hardware management, while SBCs are computers that run OS and have way more processing power (smarter!).
If you a nerd this will help: Micro controllers can't run DOOM, while SBCs can.
Ok, that is enough. We know the difference, but lets focus on micro controllers for now.

An Arduino has:

- I/O pins, digital pins 0-13, analog pins 0-5
- 2 power sources. One is the USB port that can draw power from the USB connection. Another is power jack that inputs DC power of 6-12 volts
- 4 LEDs and reset button. L is the on board LED that connects with digital pin 13. TX and RX are indicators of transmission signal and received signal. When we download a sketch to the Arduino, these two lights blink, indicating that data is being transmitted and received

Of course, to work with an Arduino, we need to code it! for that, we will use the Arduino IDE. Note, this notebook is located in a github repository, where all the projects made by me will be stored, with instructions and explanations of the builds in this document.

Here is a list on how to install and run Ardunio IDE and then load and run a short Arduino script to check if the micro controller is working:

1. Download Arduino IDE in the Arduino page.
2. Connect the board to the pc using the USB cable (the led should turn on), this will also install any driver in your pc.
3. While the Arduino is connected to your computer, open the Arduino IDE, and go to file > examples > Basics > Blinks (or similar).
4. Click Verify to verify code (duh).
5. IMPORTANT: Go to TOOLS > Board > ARDUINO UNO.
6. IMPORTANT: Go to TOOLS > Serial Port > < *Port* >. This step might fail, if this happens, go to Device Manager, then Ports (COM AND LPT) then look for ARDUINO UNO (COM).
7. Finally, click upload to load the script into the Arduino!
8. The onboard LED should start to blink, If so, you did it!

This concludes the introduction to Arduino, now that we have the important stuff out of the way, lets do some cool stuff.

2. About circuits

Ok, so I forgot something very important about working with micro controllers and hardware and sensors and any peripherals ... you kinda need to know a little about circuits.

You can use this chapter as a cheat sheet, where you can learn/remember about concepts or techniques that we'll use.

Electricity is a type of energy, it flows through conductive material. Anything that transforms energy is called a transducer, so, for example, a LED is a transducer. Sensors turn other energies into electricity, and actuators do the opposite. Circuits are like roads, made with cables and transducers.

In circuits, electricity runs from a point of high potential energy (known as the positive), to a point of low potential energy. Ground (GRD) (masa en español), is the point with the lowest potential energy in the whole circuit.

There are two main types of circuits:

- DC (direct current): Electricity circulates only in one direction.
- AC (alternate current): Electricity flow changes direction pretty fucking fast, AC is generally preferred because is easy to generate and transform, while also being pretty efficient. although calculating stuff related to AC is way harder than DC and I don't want to delve into the intricacies of electromagnetism right now.

Terms:

- Current (I): Measured in Amps (A), is the rate of flow of charge in a certain point in time: $\frac{dq}{dt}$.
- Tension (Voltage V): Measured in Volts (V), is the electrical potential difference in a certain point in space.
- Resistance (R or Ω): Measured in Ohms (Ω), is the opposition of a material to the flow of electricity.
- Power (P): Measured in Watts (W) is the rate at which electrical energy is transferred or used in an electrical circuit. It's essentially how quickly energy is used or delivered.
- Ohms Law: $V = I \times R$, $I = \frac{V}{R}$, $R = \frac{V}{I}$.
- Power Formulas: $I = \frac{P}{V}$, $V = \frac{P}{I}$, $P = V \times I$.

Basically:

I (Current) is how much electric charge is flowing — like the amount of water flowing through a pipe.

V (Voltage) is the push that causes the current to flow — like the pressure in the pipe.

R (Resistance) is how much the material resists the flow — like a narrow or rough pipe slowing the water down.

P (Power) is how much energy is being used or transferred per second — like how much work the water is doing (e.g. turning a turbine).

Electricity current follows certain rules: Current flow must remain constant in a closed circuit (of course, energy isn't destroyed), current flow really likes to go fast, so it avoids high resistance, connecting SV to GND directly generates a short circuit, of course, this is a path of low resistance and so, (I) becomes very large.

Remember, you can have:

- Parallel circuits: Current is sum of currents each branch, voltage is same across all branches, total resistance is less than of any individual component.

- Series circuits: Current is same in branch, voltage is sum of drops across branch, total resistance is the sum of individual resistances.

Arduino elements:

- Proto Shield: mini-breadboard, basically used for creating mini prototypes, it connects to the Arduino if you place it on top of it.
- Breadboard: White thing with holes on it, you can use it to create circuits and contraptions without welding.
- Jumper M/M: Cables male to male.
- Resistor: Cute little thing that resists flow (look for color ring markings to determine ohms).
- LED: Light-emitting diode. Two-lead semiconductor light source, remember, positive is longer and connected to a tiny triangle inside the led, while negative is shorter and connected to a flatter edge.
- Push button: Button that when pushed, closes the circuit.

3. LED Blink

Items needed:

- LED DIODE (any color)
- Push Button (pulsador)
- Resistors ()