

Introducción a Arduino

Organización del Computador I

Alexis Tcach

Departamento de Computación - FCEyN
UBA

2^{do} Cuatrimestre 2017

Microcontroller vs Microprocessor

Microcontroller

- tipicamente de 8 Bits
- resuelve bien tareas específicas, e.g. lavarropas, microondas, etc.
- Un programa que se repite, almacenado en ROM
- IO onboard
- Poco Consumo

Microprocessor

- tipicamente de 32/64 Bits
- multiples programas
- Usos complejos, matemática, etc.
- IO externos, por ejemplo placas de video, red, etc.

Algunas marcas y modelos. Atmega, PIC, Cypress, Intel, etc...



■ Pic

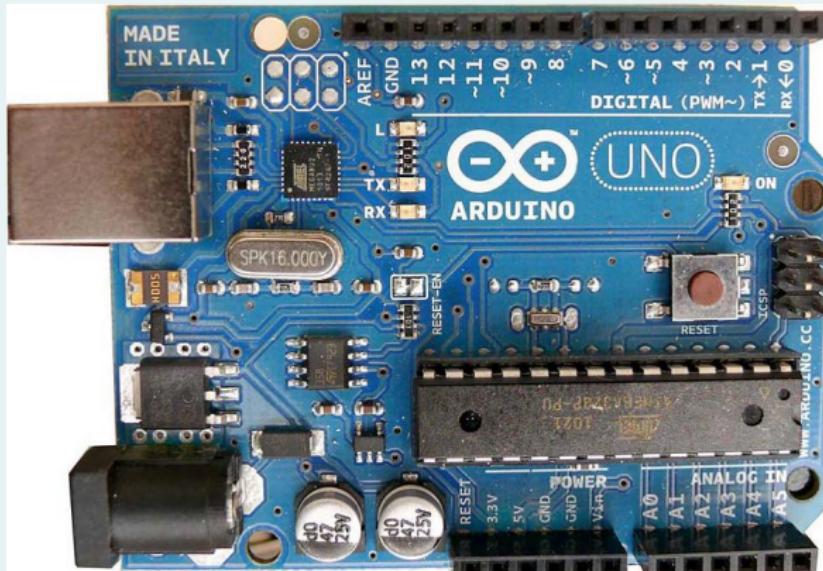
- 8-bit PIC® MCU
- PIC10
- PIC12
- PIC16
- PIC18
- 16-bit PIC® MCU
- PIC24F
- PIC24H
- PIC24E
- 32-bit PIC® MCU
- PIC32

■ Atmega

- AT32UC3L016
- AT32UC3L032
- AT32UC3L064
- AT32UC3L0128
- AT32UC3L0256
- ATUC64L3U
- ATUC128L3U
- ATUC256L3U
- ATUC64L4U
- ATUC128L4U
- ATUC256L4U

Arduino UNO

Esquema

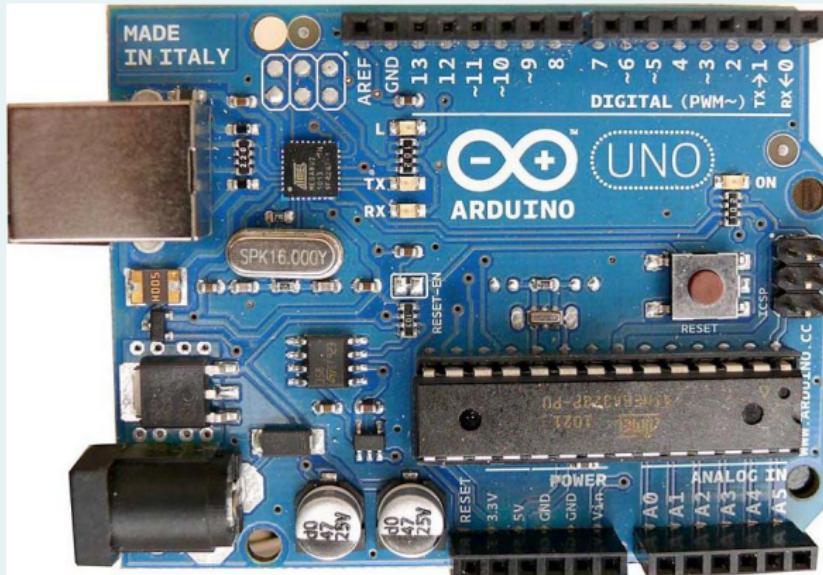


¿Qué es?

- Plataforma de cómputo
- Open Source
- Hardware Abstracted Wiring Language
- Programable por USB
- Popular
- Win-Mac-Linux
- Barato
- Todo en una sola placa
- Placa experimental. No para producción

Arduino UNO

Esquema

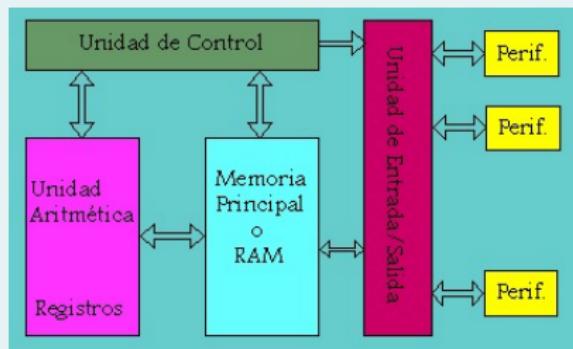


Especificaciones

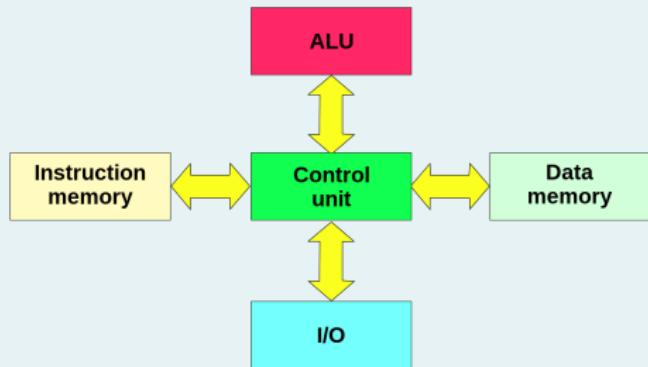
- Basado en ATmega328-8bits
- USB
- RISC, 16Mhz, 20 MIPS
- 2KB RAM !!
- 32 K Memory
- 1KB EEPROM
- 6 Ch 10 Bit A/D
- 13 Ch Digitales
- 3 timers
- 5 modos de ahorro de energía
- Arquitectura *Harvard*

Arquitectura Hardvard vs. Von Newmann

Von Newmann
(Lo que vienen viendo)



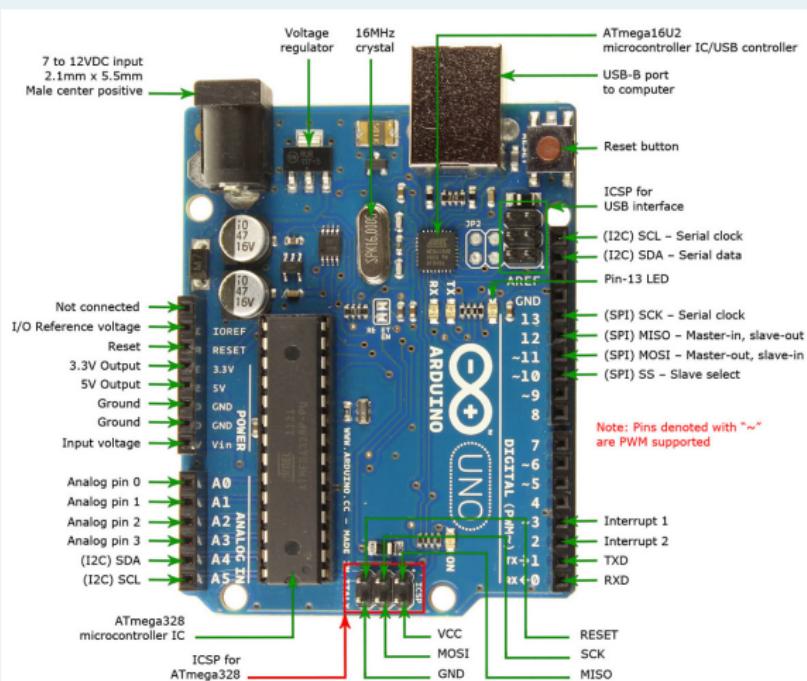
Hardvard



Aunque hoy por hoy casi siempre se usan modelos mixtos

Esquema del Arduino UNO

Esquema



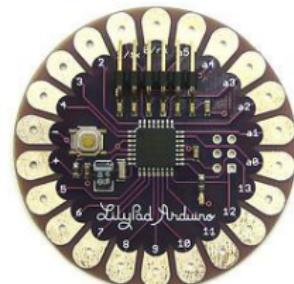
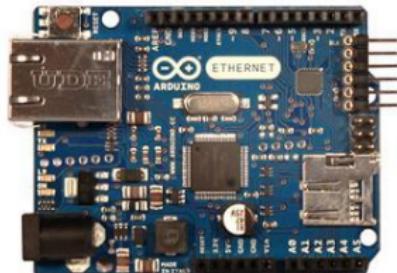
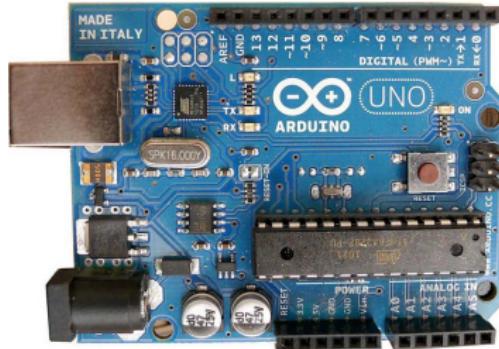
¿Qué podemos hacer?

- Digital IO (Leds, switches)
- Analog IO (resistive sensor data)
- Serial IO (Sensores, GPS, arduino, etc)
- Programable desde la Compu

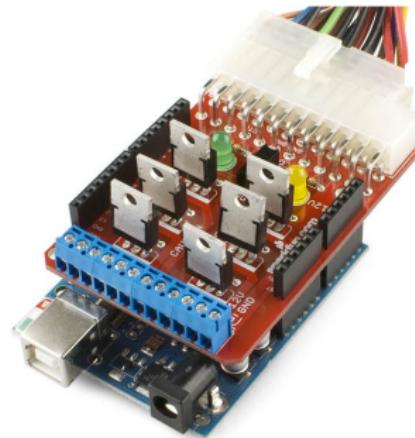
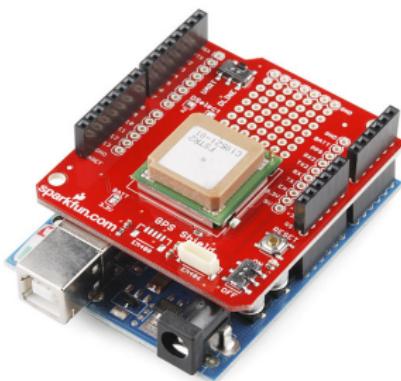
Arduino terminología

- IO Boards-micontrolador
- Shield - add-on boards
- Sketch - el programa
- Sensor - componentes(sensores de luz, etc.)
- Modules - serial data (GPS, etc.)

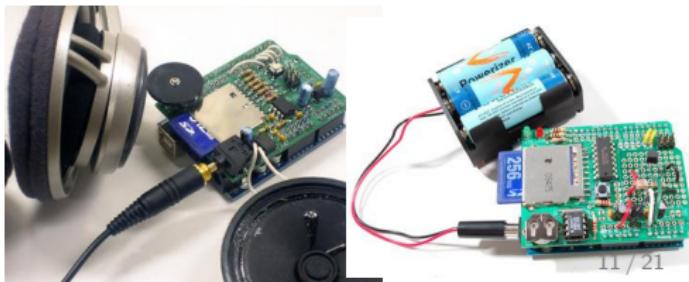
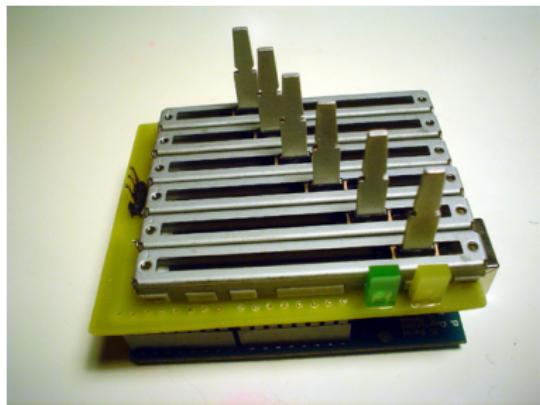
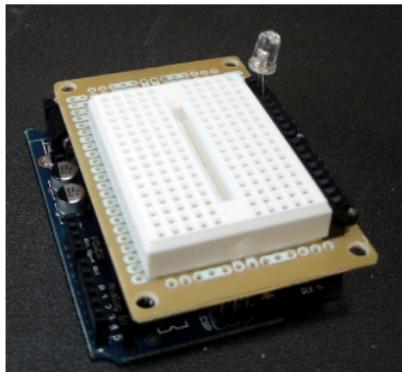
Aspecto



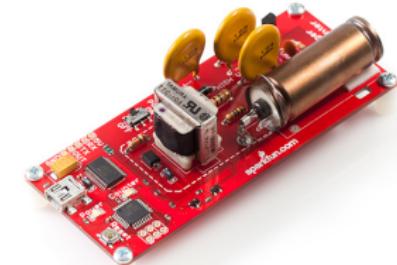
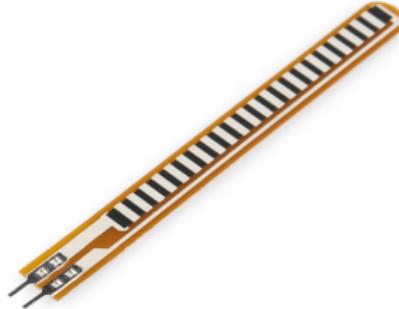
Shields



Shields...



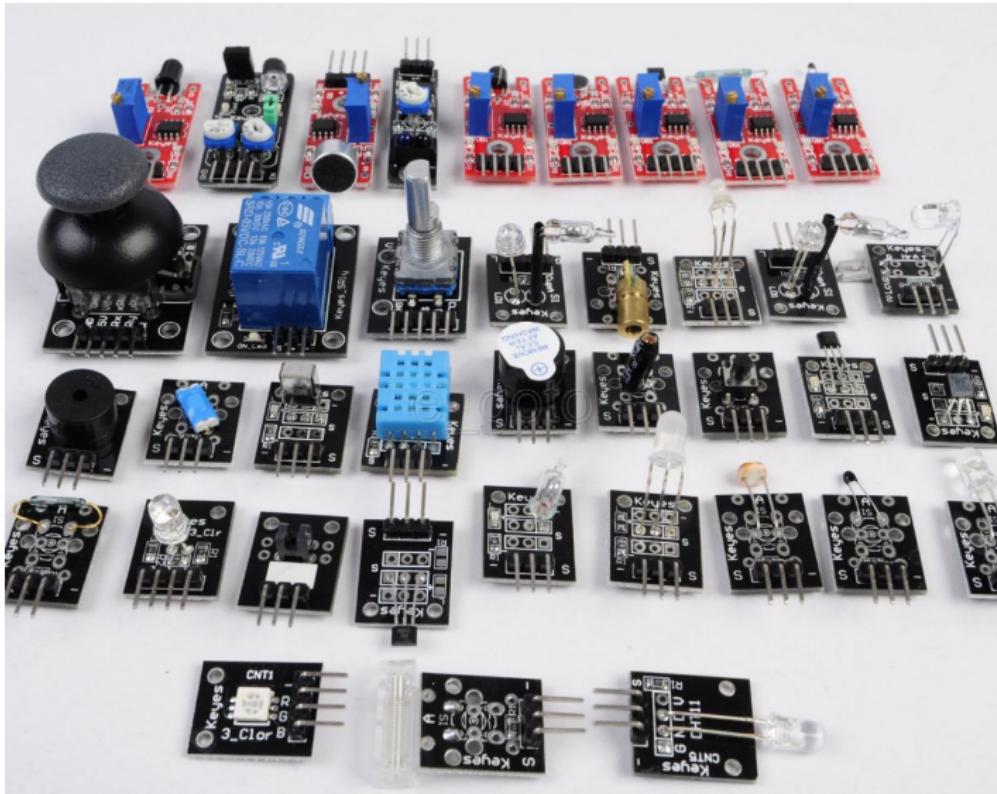
Sensores



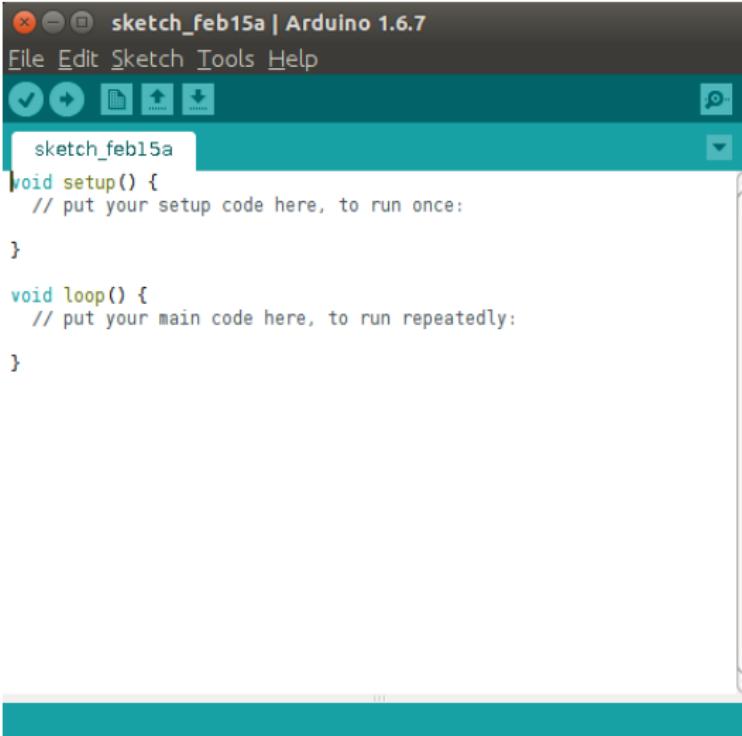
Kits



Kits...



Sketch



The screenshot shows the Arduino IDE interface. The title bar reads "sketch_feb15a | Arduino 1.6.7". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for file operations like Open, Save, and Print. The main workspace displays the following code:

```
sketch_feb15a
void setup() {
    // put your setup code here, to run once:
}

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

A vertical scroll bar is visible on the right side of the code editor.

Sketch

Funciones principales:

- `setup()`

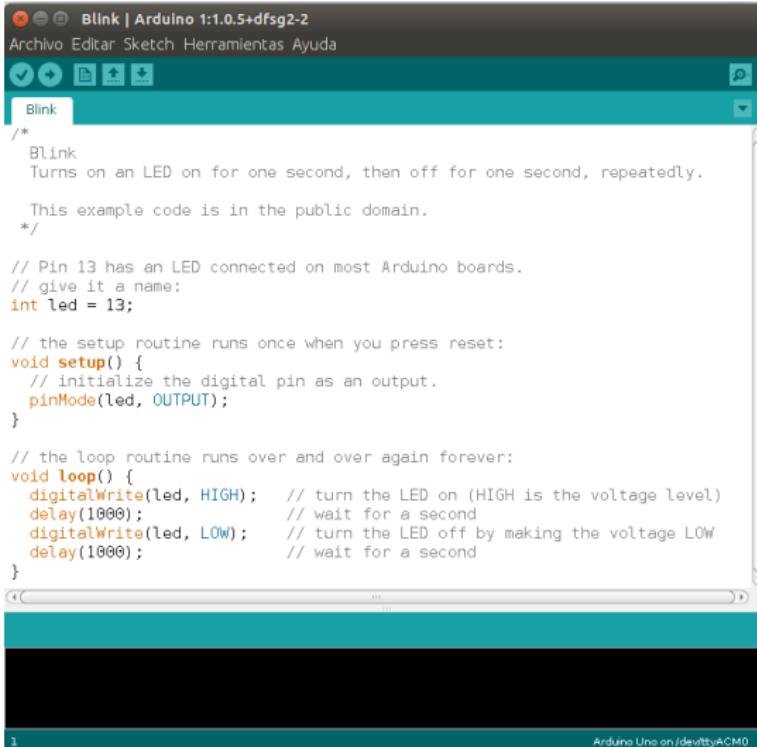
Configuración inicial: selección de pines, sensores, formatos, velocidades, etc.

- `loop()`

Rutina principal, que se ejecuta constantemente

- Fuertemente orientada a interrupciones, por el tipo de uso y por hardware disponible.

Blink



The screenshot shows the Arduino IDE interface with the title bar "Blink | Arduino 1:1.0.5+dfsg2-2". The menu bar includes "Archivo", "Editar", "Sketch", "Herramientas", and "Ayuda". Below the menu is a toolbar with icons for file operations like Open, Save, and Print. The code editor window contains the "Blink" sketch. The code is as follows:

```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 *
 * This example code is in the public domain.
 */
 
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
    // initialize the digital pin as an output.
    pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
    digitalWrite(led, HIGH);      // turn the LED on (HIGH is the voltage level)
    delay(1000);                // wait for a second
    digitalWrite(led, LOW);       // turn the LED off by making the voltage LOW
    delay(1000);                // wait for a second
}
```

At the bottom of the IDE, there is a status bar with the text "Arduino Uno on /dev/ttyACM0".

File -> Examples

BLINK: prender y apagar un led por un segundo

Configuración de Pines

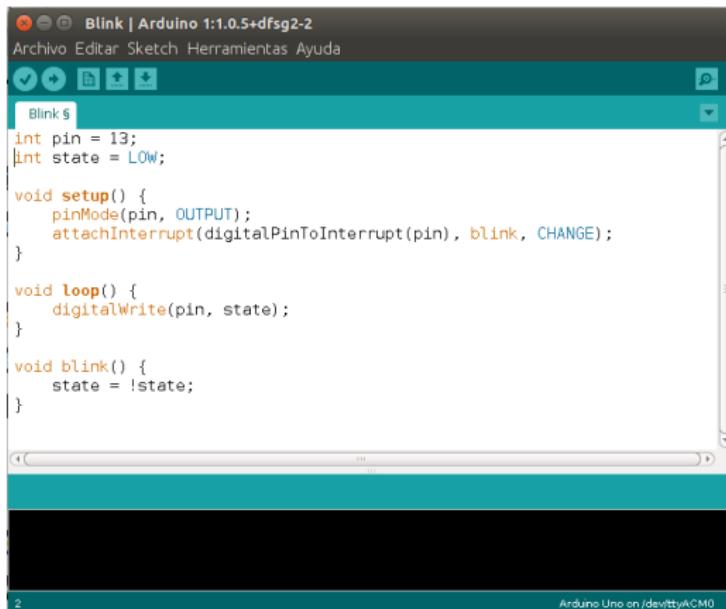
Modo del Pin

```
pinMode(pin, mode)  
// mode: OUTPUT, INPUT or INPUT_PULLUP
```

Interrupciones

```
attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)  
// mode: LOW, CHANGE, RISING, FALLING
```

Blink



The screenshot shows the Arduino IDE interface with the title bar "Blink | Arduino 1:1.0.5+dfsg2-2". The menu bar includes "Archivo", "Editar", "Sketch", "Herramientas", and "Ayuda". Below the menu is a toolbar with icons for file operations. The main window displays the "Blink" sketch:

```
int pin = 13;
int state = LOW;

void setup() {
  pinMode(pin, OUTPUT);
  attachInterrupt(digitalPinToInterrupt(pin), blink, CHANGE);
}

void loop() {
  digitalWrite(pin, state);
}

void blink() {
  state = !state;
}
```

The status bar at the bottom indicates "Arduino Uno on /dev/ttyACM0".

Blink

Blink usando Interrupciones

Tarea

Hacerse una cuenta en en <https://123d.circuits.io/>

¿Preguntas?

