

Introducción al entorno Arduino y ArduinoBlocks

Arduino es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinarios .

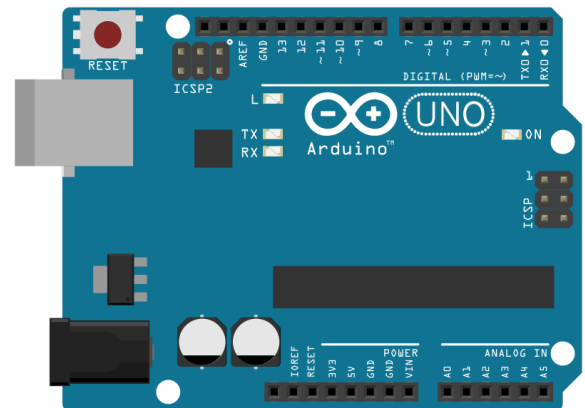
El hardware consiste en una placa con un microcontrolador Atmel AVR y puertos de entrada/salida. Los microcontroladores más usados son el Atmega168, Atmega328, Atmega1280, y Atmega8 por su sencillez y bajo coste que permiten el desarrollo de múltiples diseños.

Arduino puede tomar información del entorno a través de sus entradas analógicas y digitales, puede controlar luces, motores y otros actuadores.

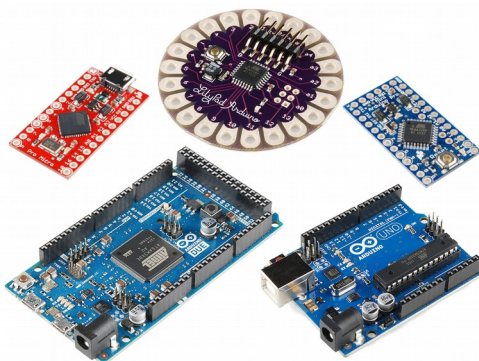
También cuenta con su propio software que se puede descargar de su página oficial que ya incluye los drivers de todas las tarjetas disponibles lo que hace más fácil la carga de códigos desde el computador.

<https://www.arduino.cc/>

Placa Arduino UNO



Otros modelos (Lista completa: <https://www.arduino.cc/en/Main/Products>)



Arduino UNO: Características técnicas

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Alimentación de la placa Arduino

La placa Arduino UNO se puede alimentar de dos formas:

- A través del conector USB: cuando conectamos al ordenador para programarlo o utilizando un “power bank” con conexión USB por ejemplo.
- A través del conector de alimentación externa. La fuente de alimentación conectada debe ofrecer un voltaje DC de 9v a 12v. Internamente la placa Arduino regula la tensión a 5v.

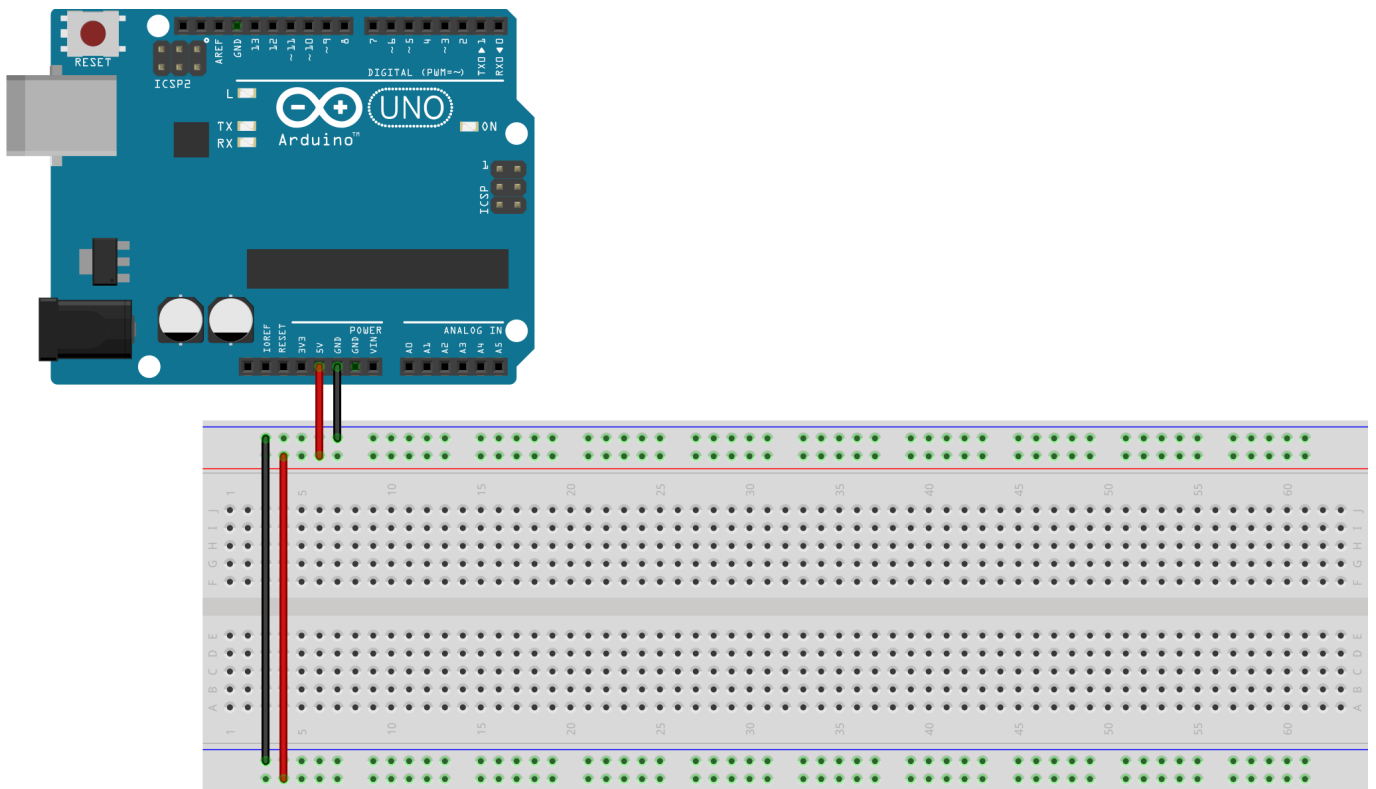
A través de los pines 3.3v , 5v, GND y Vin obtenemos la alimentación para circuitos auxiliares, sensores, shields, etc.

- 3.3v proporciona una tensión de 3.3v y una corriente máxima de 50mA
- 5v proporciona una tensión de 5v y una corriente máxima de 300mA
- GND es el nivel 0v de referencia
- Vin proporciona la tensión de alimentación conectada al conector de alimentación (sin regular)

Algunas formas de alimentar la placa Arduino:



Ejemplo de conexión de la alimentación suministrada en los pines 5v y GND de Arduino para utilizarlo en una placa de prototipos donde conectar sensores, actuadores, etc.:



Entradas y salidas en Arduino UNO

Arduino UNO dispone de múltiples conexiones (pines) de entrada/salida.

- Salida digital Los valores de salida pueden ser 0v (LOW) o 5v (HIGH)
- Entrada digital Se leerá un nivel LOW cuando la entrada esté entre 0 y 2v y se leerá un valor HIGH cuando esté entre 3 y 5v
- Salida analógica Se podrá fijar un valor en la salida de 0 a 5v en un rango de valores de 0 a 255 (0=0v y 255=5v)
- Entrada analógica Se pueden leer valores de entrada entre 0 y 5v en un rango de 0 a 1023 (0=0v y 1023=5v)

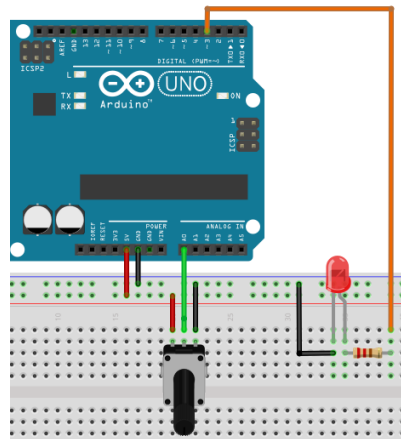
Pines 0...13 : Pueden funcionar como entradas/salidas digitales

Pines (~) 3,5,6,9,10,11: Pueden funcionar como salidas analógicas (PWM)

Pines A0...A5: Pueden funcionar como entradas analógicas o como entradas/salidas digitales

Salida digital (para activar led): pin 3

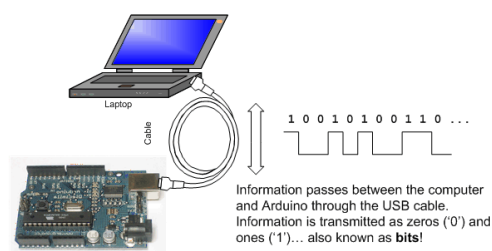
Entrada analógica (para leer el valor del potenciómetro): pin A0



Comunicaciones

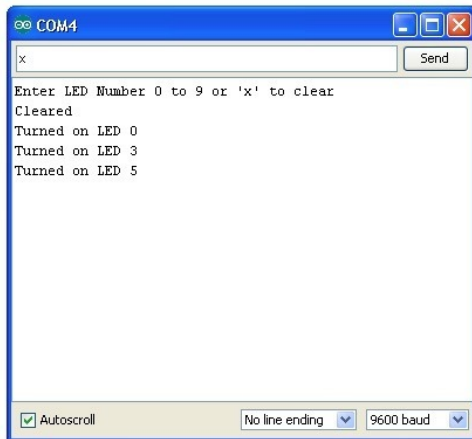
Arduino permite comunicarse con periféricos o con el PC de diferentes formas:

- Puerto serie (COM): Permite una conexión serie (tipo RS232) a través del puerto USB. Utilizada principalmente para programar la placa Arduino desde el ordenador y para comunicar con aplicaciones en un ordenador (consola serie).

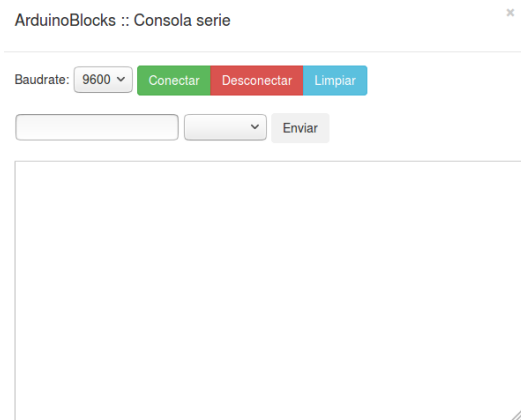


Con una aplicación de terminal o consola serie podemos visualizar los datos recibidos desde Arduino y enviar datos al Arduino fácilmente.

Consola serie de Arduino IDE



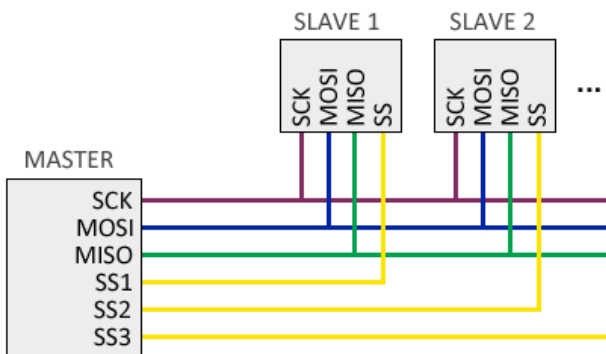
Consola serie de ArduinoBlocks



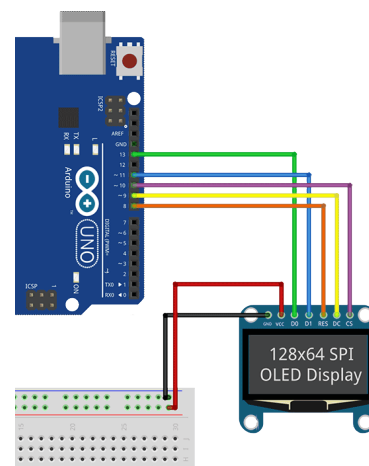
La conexión serie de Arduino está accesible en los pines 0 (RX) y 1 (TX), internamente estos pines están conectados al chip que gestiona el USB, en caso de usar la comunicación serie con otros dispositivos conectados a los pines 0 y 1 se deberá desconectar el USB para evitar que se mezclen los datos y no funcione la comunicación.

- **SPI (Serial Peripheral Interface):** Mediante SPI podemos conectar periféricos compatibles con este protocolo de comunicación. El Bus SPI permite un maestro y múltiples esclavos conectados formando una pequeña red de dispositivos que se conectan entre sí. Los pines de Arduino utilizados para SPI son: 11 (MOSI) , 12 (MISO) y 13 (CLK), por otro lado cada dispositivo conectado al bus SPI necesita un pin más para seleccionarlo (SS/CS → Slave Select/Chip Select). En caso de un sólo dispositivo conectado al bus suele usarse el pin 10 como CS/SS .

Bus SPI con dos dispositivos conectados



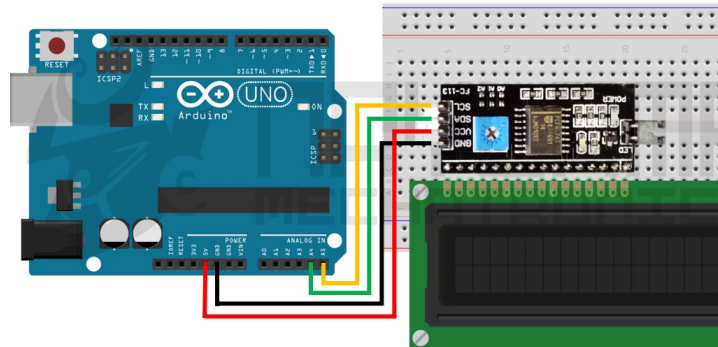
Pantalla OLED conectada por SPI



- I2C: Es otro sistema de bus de comunicaciones en serie. Al igual que SPI permite crear una pequeña red de dispositivos (pensado principalmente para interconectar otros chips en la propia placa o cerca). La comunicación I2C sólo utiliza 2 cables.

Para I2C Arduino utiliza los pines: A4 (SDA) y A5 (SCL)

Ejemplo de pantalla LCD conectada por I2C:



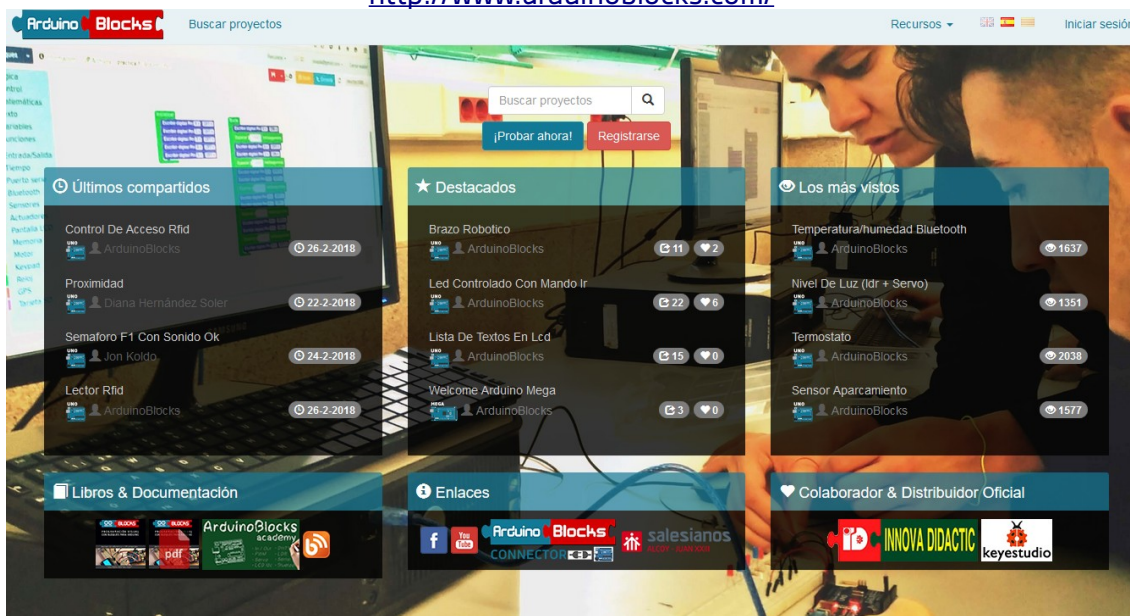
Programación de Arduino

Arduino se programa en lenguaje C++ (con algunas variaciones para simplificarlo).

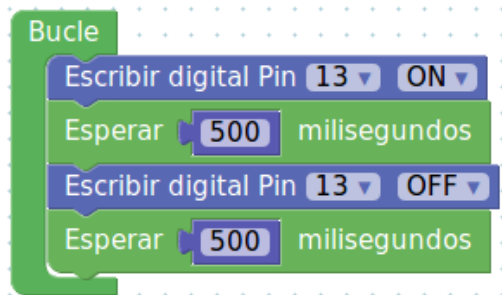
Para programar normalmente se utiliza el IDE (entorno integrado) de Arduino, que permite escribir el código, compilar el programa (crear el programa binario para el procesador Arduino) y grabarlo en la placa Arduino a través del puerto USB. El IDE de Arduino se puede descargar desde la web oficial. Es totalmente libre: <https://www.arduino.cc/en/Main/Software>

Gracias a ArduinoBlocks.com podemos programar Arduino de forma visual con lenguaje de bloques sin necesidad escribir ni una sólo línea de código. De esta forma podemos empezar a realizar proyectos con Arduino de una forma muy rápida y a edades más tempranas.

<http://www.arduinoblocks.com/>



Programa creado con ArduinoBlocks:



Código generado en C++ para Arduino

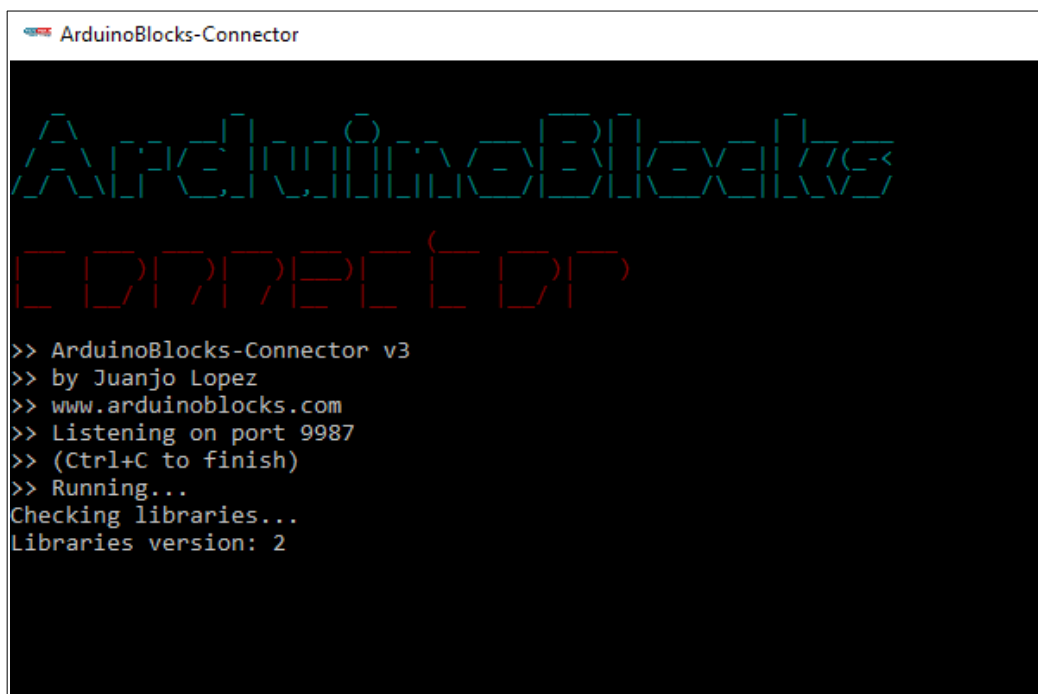
```
void setup()
{
    pinMode(13, OUTPUT);
}

void loop()
{
    digitalWrite(13, HIGH);
    delay(500);
    digitalWrite(13, LOW);
    delay(500);
}
```

Para poder programar la placa Arduino directamente desde el navegador debemos instalar y ejecutar la aplicación ArduinoBlocks-Connector (Windows, Linux, Mac):

<http://www.arduinoblocks.com/web/site/abconnector>

Una vez descargado e instalado simplemente se ejecuta y se puede dejar minimizado. El programa recibirá el código generado por ArduinoBlocks desde el navegador web y se encargará de compilar y subir el programa a la placa Arduino automáticamente de forma transparente para el usuario.



Para más información sobre ArduinoBlocks-Connector y como realizar una compilación remota desde otros sistemas como Android o ChromeOS consultar aquí:

<http://www.arduinoblocks.com/blog/2017/02/14/arduinoblocks-connector/>

Bibliografía y enlaces de interés:

<http://www.arduinoblocks.com>

<https://www.arduino.cc/>

<http://www.keyestudio.com/>

<http://realtterm.sourceforge.net/>

<https://es.wikipedia.org/wiki/I%C2%B2C>

https://es.wikipedia.org/wiki/Serial_Peripheral_Interface

<http://ai2.appinventor.mit.edu>

<http://www.pighixx.com>

<https://learn.sparkfun.com/tutorials/what-is-an-arduino>

<https://www.arduino.cc/en/Tutorial/HomePage>

<https://scratch.mit.edu/>

<http://fritzing.org/home/>

Arduinoblocks en las redes sociales:

<https://www.facebook.com/ArduinoBlocks>

<https://www.youtube.com/channel/UCoJwWGyd8a2pxzJHFdfXYw>

<https://twitter.com/arduinoblocks>

<https://es.linkedin.com/in/arduinoblocks-programaci%C3%B3n-visual-5169a9133>

<http://shop.innovadidactic.com>

NOVEDADES, PROYECTOS Y NUEVOS BLOQUES

<http://www.arduinoblocks.com/blog/>

Contacto:

Juanjo López

info@arduinoblocks.com