



Periféricos y Dispositivos de Interfaz Humana Grado en Ingeniería Informática

SEMINARIO 2: INTRODUCCIÓN AL USO DE LA PLATAFORMA ARDUINO

Objetivos

- I. Conocer qué es la plataforma Arduino: arquitectura, modo de funcionamiento, programación y conexión de periféricos.
- II. Describir un kit de programación para Arduino (Lab Kit, *Cooking Hacks*): componentes electrónicos y funcionalidad.
- III. Instalar y realizar sencillos programas mediante el IDE de programación basado en el lenguaje Processing de Arduino.
- IV. Conocer herramientas adicionales para el diseño de prototipos en Arduino (Fritzing).

Contenido

| | | |
|----|-------------------------------|----|
| 1. | Introducción..... | 2 |
| 2. | Entorno de trabajo..... | 3 |
| 3. | Precauciones y seguridad..... | 3 |
| 4. | Descripción del hardware..... | 3 |
| 5. | Kit de desarrollo..... | 8 |
| 6. | Programación..... | 10 |
| 7. | Complementos..... | 12 |
| 8. | Ejercicios..... | 14 |
| 9. | Referencias..... | 16 |

1. Introducción

Según sus propios creadores (<http://www.arduino.cc/>) [1]: *“Arduino es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de usar. Se creó para artistas, diseñadores, aficionados y cualquiera interesado en crear entornos u objetos interactivos”*.



Arduino se basa en trabajos previos, el primero de ellos es el lenguaje de programación Processing y su entorno de desarrollo, diseñado especialmente para personas sin experiencia en programación. Posteriormente se creó la plataforma Wiring, que es una plataforma hardware para realizar prototipos rápidos de aplicaciones. El lenguaje de programación y las herramientas derivan en último término de C. Sobre este lenguaje se añaden elementos de Processing/Wiring para hacerlo lo más sencillo posible. Mediante este lenguaje se describirá el comportamiento del computador de la placa Arduino. Los ficheros que contienen el texto del programa se denominan *“Sketchs”* (bocetos).

La plataforma está formada por tres componentes principales que la hacen funcionar y ser seguida a nivel mundial por miles de usuarios:

- Hardware: placa electrónica básica para hacer desarrollos rápidos y económicos. Basado en hardware libre: los códigos y esquemas de conexión están disponibles para todo el mundo. Se permite una forma simple de aprender electrónica básica analizando los componentes que forman sus circuitos. También permite ampliar el hardware mediante el uso de otras placas (*shields*) que se conectan a la del microprocesador.
- Entorno de programación: Processing permite de forma muy sencilla la programación del hardware escogido. Además, este entorno es multiplataforma (Windows, Mac y Linux) y únicamente necesita un cable USB para conectarse a la placa.
- Comunidad de usuarios: existen miles de usuarios que participan activamente en nuevos desarrollos, ideas y evaluación. Se puede consultar cualquier duda tanto del hardware como del software de forma libre, tanto en la web oficial de Arduino como en múltiples foros y webs creados por la comunidad.

El éxito y repercusión que ha tenido se debe sobre todo a varios factores: muy bajo coste del material a utilizar, facilidad de entendimiento y manejo para crear aplicaciones concretas y finalmente se trata de un software y hardware abierto, con una licencia que permite su estudio, reproducción y modificación [2].

Arduino permite implementar sistemas integrados de una forma muy sencilla. Utilizándolo se dispone de un hardware genérico, el diseño del hardware se reduce a conectar placas de expansión o shields a la placa base de Arduino. De esta forma se elimina una de las partes que requiere conocimiento más específico y que supone fabricación. Así se permite que el desarrollador se concentre en la programación del firmware y en la funcionalidad del sistema.

El firmware es el código de los elementos programables, ya sea el microcontrolador, DSP o FPGA. Se escribe como texto en un determinado lenguaje, por ejemplo, en el caso de microcontroladores suele ser C. El código del firmware es traducido por un compilador a código máquina interpretable

por el procesador. Este código es una secuencia de instrucciones que ejecutará el procesador. La combinación de estas instrucciones determina la forma en que el dispositivo funciona e interactúa con los demás elementos.

2. Entorno de trabajo

Para trabajar con la placa en el aula de prácticas, entraremos al cargador de imágenes REMBO (el que aparece al encender la máquina) con nuestro usuario y contraseña y con código “tec_em”. Hay que tener en cuenta que esta imagen tiene instalado el software y drivers para la placa Arduino modelo UNO R3. Si se desea usar otra placa, no contamos con privilegios de administrador para instalar los drivers, y por tanto, habrá que usar un portátil.

En caso de usar nuestro portátil, descargaremos el entorno de programación y seguiremos las instrucciones en:

Windows: <http://arduino.cc/es/Guide/Windows>

MAC OS: <http://arduino.cc/es/Guide/MacOSX>

Linux: <http://playground.arduino.cc/Learning/Linux>

3. Precauciones y seguridad

Antes de usar la placa, y para evitar daños en la misma, en el PC o entre nosotros, es conveniente leer la siguiente información [3]:

<http://www.maelabs.ucsd.edu/mae156alib/electronics/DestroyArduino.pdf>

(“10 great ways to destroy an Arduino”, diez formas de destrozar una Arduino).

Este documento se puede encontrar traducido en <http://www.trastejant.es/blog/?p=192>.

4. Descripción del hardware

4.1. Tipos de placas Arduino

Existe una gran variedad de placas de Arduino, variando los parámetros del tamaño de memoria, modelo de microcontrolador, periféricos disponibles, geometría y medidas de la placa, etc. Además de las placas oficiales Arduino, también existe una amplia oferta de clones y versiones a precios aún más reducidos. En la Figura 2 se puede apreciar de forma gráfica la evolución de las placas oficiales de Arduino. La tabla de la Figura 1 resume las características de las principales placas Arduino.

| Nombre | Procesador | Reloj (MHz) | Memoria (kB) Flash/RAM/ ROM | Pines IO/A | Periféricos | Tamaño (mm) |
|-------------|--------------------------|-------------|-----------------------------------|---------------|----------------------------------|-------------|
| UNO | ATmega328 | 16 | 32/2/1 | 14/6 | - | 68,6x53,3 |
| Mega | ATmega2560 | 16 | 256/8/4 | 54/16 | - | 101,6x53,3 |
| Leonardo | ATmega32 | 16 | 32/2,5/1 | 20/12 | | 68,6x53,3 |
| Esplora | ATmega32 | 16 | 32/2,5/1 | | Sensores, TFT color, USART | 165,1x61 |
| Duemilanove | ATmega328 | 16 | 32/2/1 | 14/6 | - | 68,6x53,3 |
| Mini Pro | ATmega328 | 8/16 | 32/2/1 | 14/8 | - | 17,8x33 |
| Micro | ATmega32 | 16 | 32/2,5/1 | 20/1 2 | - | 17,8x48x3 |
| Nano | ATmega328 | 16 | 32/2/1 | 14/8 | - | 18,5x43,2 |
| Fio | ATmega328 | 8 | 32/2/1 | 14/8 | - | 66x27,9 |
| LilyPad | varios | 8 | - | 14/6 | - | 50 |
| Ethernet | ATmega328 | 16 | 32/2/1 | 14/6 | Ethernet | 68,6x53,3 |
| Bluetooth | ATmega168 | | 32/2/1 | 14/6 | Bluetooth | 81,3x53,3 |
| Robot | ATmega32 +ATmega32 | 16 | 32/2,5/1 | 6/4 | SPI, I2C, TFT, sensores | Ø 190 |
| Yún | ATmega32 +MIPS(linux) | 16 | 32/2,5/1 | 14 | Wi-fi, SD, USB | 68,6x53,3 |
| Due | ARM Cortex | 84 | 512/96/0 | 54/12 | DMA, CAN | 101,6x53,3 |

Figura 1. Características de las principales placas Arduino [2].

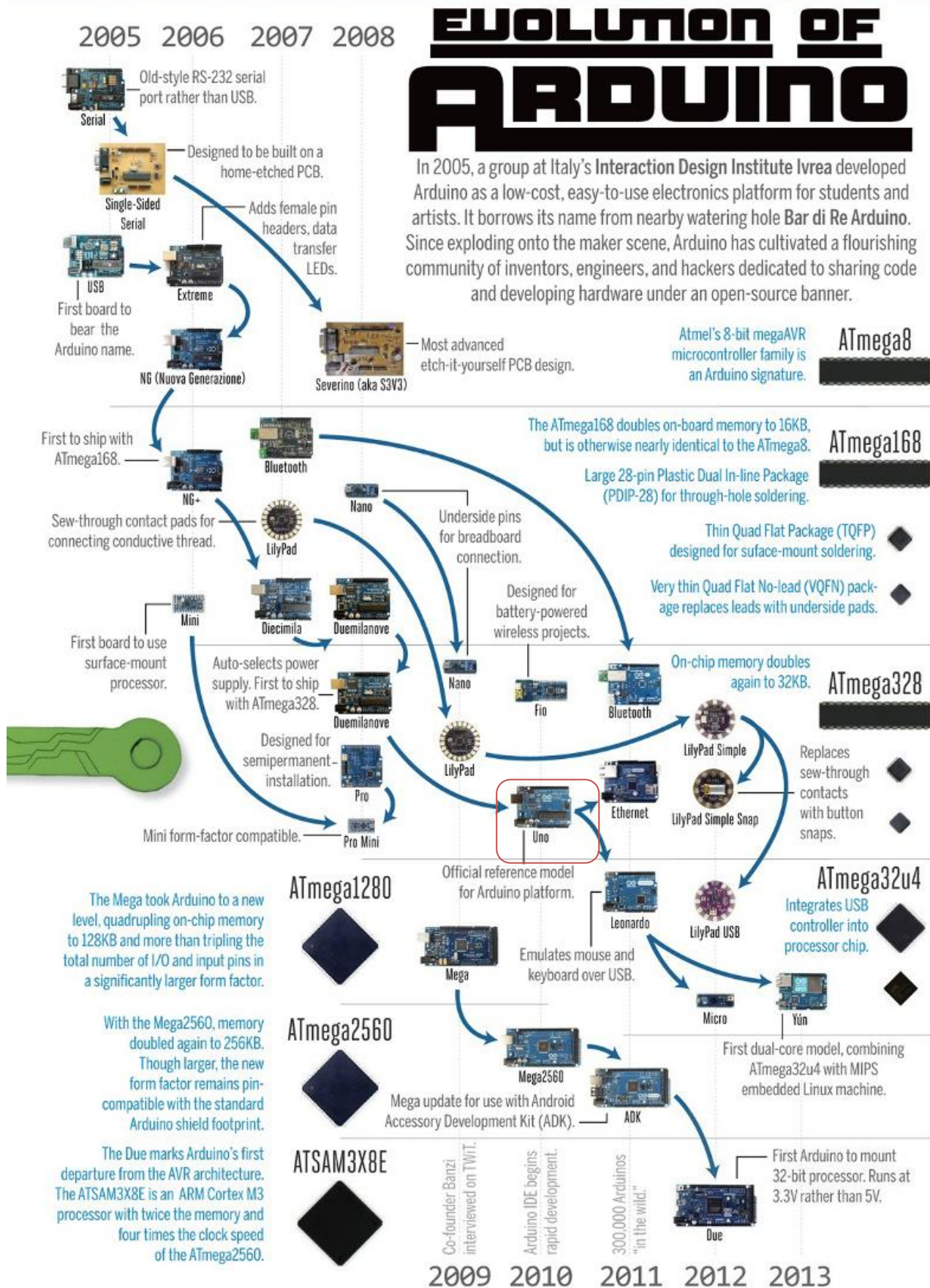


Figura 2. Evolución de las placas Arduino [4].

4.2. Placa Arduino UNO

En el laboratorio de prácticas utilizaremos la placa Arduino UNO. Esta es la placa de referencia del resto de placas de Arduino y la más usada en general. La placa está formada por los componentes descritos en la Figura 3.

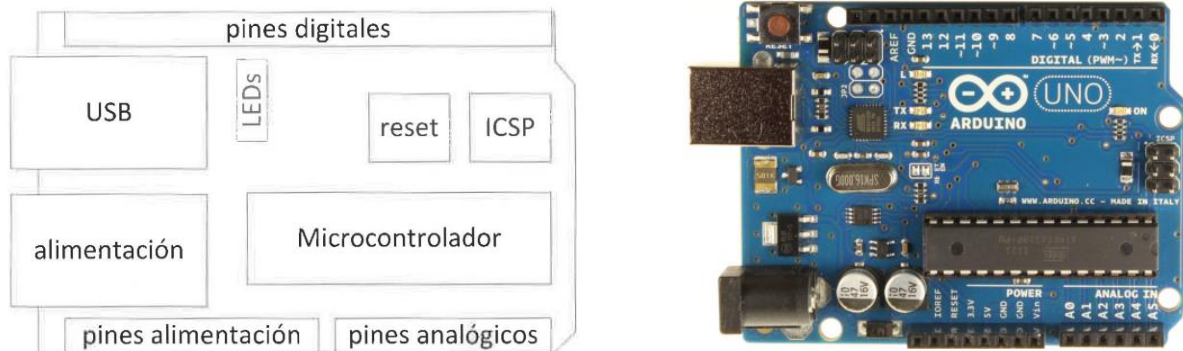


Figura 3. Estructura e imagen de una placa Arduino UNO (R2 a la izquierda y R3 a la derecha).

Microcontrolador: Es un computador en un solo chip. El microcontrolador usado por Arduino UNO es el ATmega328 de la compañía Atmel. Casi todos los Arduino utilizan microcontroladores de la familia ATmega, la placa UNO usa el modelo 328. El ATmega es un microcontrolador basado en una arquitectura Harvard modificada de 8 bits con tecnología RISC (*Reduced Instruction Set Computing*), es decir, con memorias separadas para programa y datos y un conjunto de instrucciones simples para realizar la programación.

Características:

- Voltaje de Operación: 5V
- Memoria Flash: 32 KB (512 bytes para el *bootloader* o gestor de arranque).
- SRAM 2 KB (memoria volátil)
- EEPROM 1 KB (memoria permanente de escritura eléctrica)
- Velocidad del Reloj 16 MHz



Figura 4. Microcontrolador ATmega328 .

Alimentación: la alimentación de la placa puede venir de una fuente externa o a través del cable USB. Si hay alimentación externa se utiliza ésta y un transistor bloquea la alimentación del USB para que no haya conflictos entre ellas; si no hay alimentación externa el transistor permitirá que la alimentación de la placa provenga del USB.

USB: El sistema USB de la placa UNO está formado por otro microcontrolador (ATmega8U2-MU).

Botón Reset: suministra un valor LOW(0V) para reiniciar el microcontrolador.

ICSP: Conector para la programación ICSP (*In Circuit Serial Programming*, o Programación en Serie sobre el Circuito). El ICSP es el sistema utilizado en los dispositivos PIC para programarlos sin necesidad de tener que retirar el chip del circuito del que forma parte.

Conectores: El conector de la parte inferior corresponde a la parte de alimentación y reset (izquierda) y entradas analógicas (derecha, A0-A5). El expansor de la parte superior corresponde a entradas/salidas digitales (0-13), de las cuales 6 pueden actuar como salidas PWM, *Pulse Width Modulation* (3, 5, 6, 9, 10, 11). Se debe tener presente que muchos pines pueden tener varias funciones. Para mantener la compatibilidad con las *shields* o placas de expansión, todas las placas de Arduino intentan respetar la disposición y orden de los conectores.

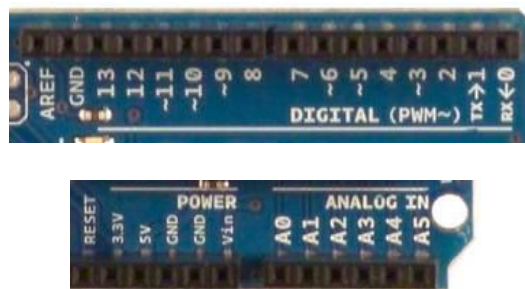
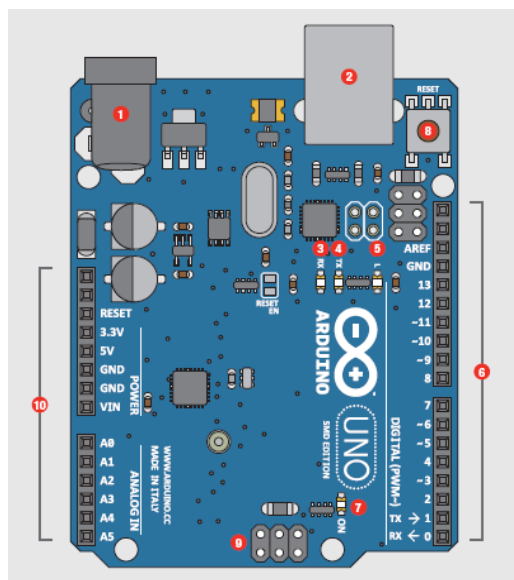


Figura 5. Conectores analógicos y digitales de la placa Arduino.

La siguiente figura resume las características y ubicación de los componentes de Arduino. Los principales elementos se puntualizan en la leyenda de la derecha.



1 - Toma de corriente: Sirve para que Arduino pueda funcionar sin estar conectado a un ordenador

2 - Conexión USB: Permite conectar un ordenador y transmitir información entre el ordenador y Arduino.

6 - Pines digitales: Permiten a Arduino transmitir órdenes a máquinas electrónicas

7 - Luz de encendido: Se enciende cuando Arduino está funcionando.

8 - Botón de reset: Se utiliza cuando se pretende resetear Arduino.

10 - Pines analógicos y de potencia: Se utiliza para recibir información y suministrar energía a las máquinas que se están controlando

Figura 6. Esquema-resumen de los componentes de Arduino.

- x2 Potenciómetro 100 K Ω
- x5 Botones de pulsación
- x1 Sensor LDR (sensor fotoeléctrico)
- x1 Sensor NTC (sensor de temperatura)
- x1 Sensor MCP (sensor de temperatura)
- x10 LEDs rojos
- x10 LEDs verdes
- x1 Cartucho para 6 pilas AA
- x2 Diodo 1N4001
- x3 Registros de desplazamiento
- x3 Acoplador óptico (*optocoupler*)
- x1 Altavoz piezo-eléctrico.
- x1 Servomotor
- x1 Mini-placa de prototipado
- x1 Placa experimental (*protoshield*)
- x1 LCD 16x2
- x1 Matriz LED
- x1 Relé PB4005
- x1 Pack sensor de movimiento: acelerómetro, sensor de inclinación, sensor de vibración.
- x2 Transistor PNP
- x2 Transistor NPN
- x1 Transistor ULN2003

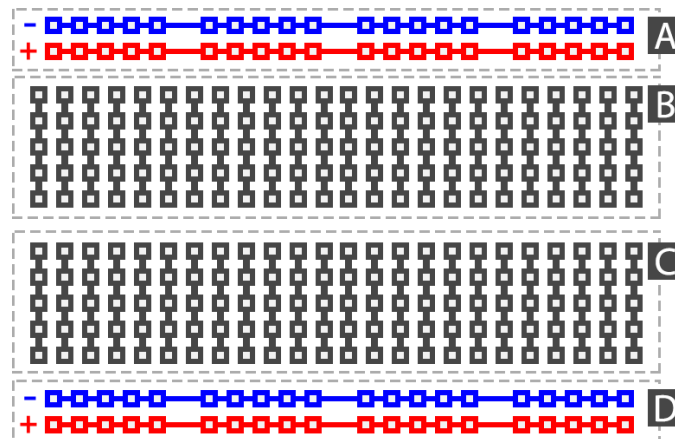
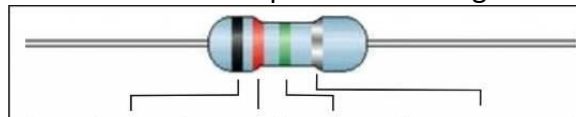


Figura 8. Interconexiones de un bread-board. A, D: bandas de alimentación. B, C: columnas de interconexión.

Para identificar el valor de las resistencias se puede usar la siguiente imagen:



| Color | 1ra. Banda | 2da. Banda | 3ra. Banda Multiplicador | Tolerancia % |
|----------|------------|------------|--------------------------|--------------|
| Negro | 0 | 0 | x1 | |
| Cafe | 1 | 1 | x10 | |
| Rojo | 2 | 2 | x100 | 2% |
| Naranja | 3 | 3 | x1000 | |
| Amarillo | 4 | 4 | x10000 | |
| Verde | 5 | 5 | x100000 | |
| Azul | 6 | 6 | x1000000 | |
| Violeta | 7 | 7 | x10000000 | |
| Gris | 8 | 8 | x100000000 | |
| Blanco | 9 | 9 | x1000000000 | |
| | | | | Dorado 5% |
| | | | | Plata 10% |

Circuitos Básicos

Figura 9. Codificación de las resistencias mediante 4 bandas

6. Programación

6.1. Entorno integrado de desarrollo (IDE)

Arduino dispone de un sencillo entorno de desarrollo que permite escribir programas, compilarlos y transferirlos al *firmware*. Al igual que el resto del entorno Arduino, se puede acceder al código fuente (*open source*), el cual está escrito en C, C++ y Java.

El IDE se compone del propio editor de texto, un conjunto de librerías y las herramientas de compilación y programación necesarias para trabajar. Dispone de versiones para diferentes sistemas operativos: Windows, Linux y Mac OS. El lenguaje de programación empleado (Processing) está basado en C.

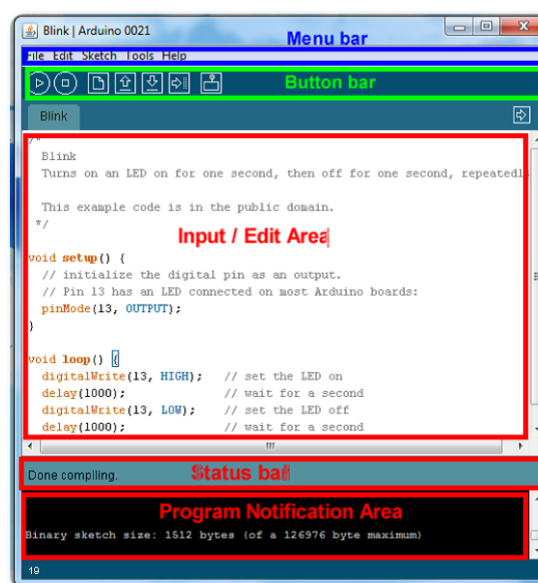


Figura 10. Apariencia del IDE de Arduino

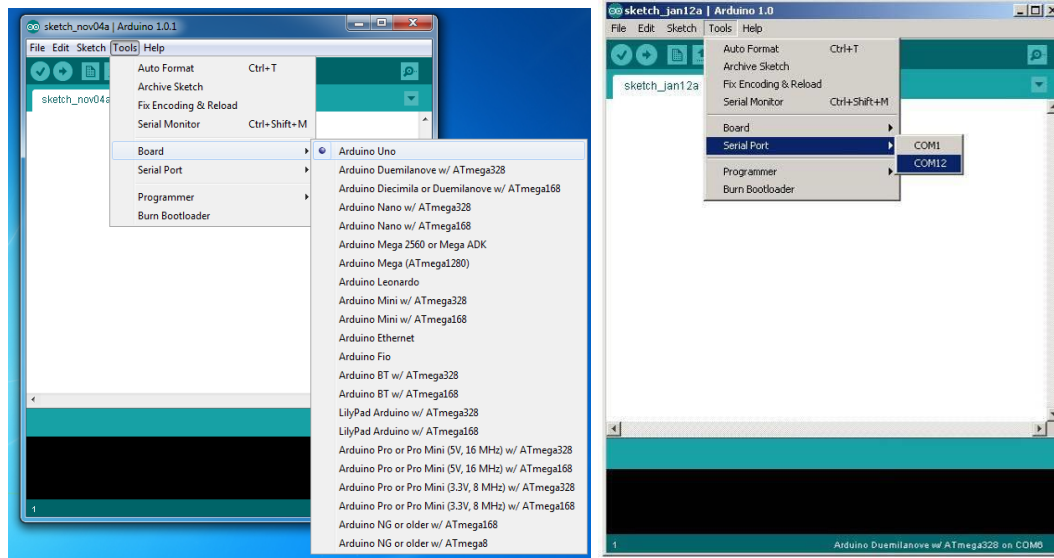


Figura 11. Selección de tipo de tarjeta y puerto serie

6.2. El “Hola Mundo” de Arduino: hacer parpadear un LED

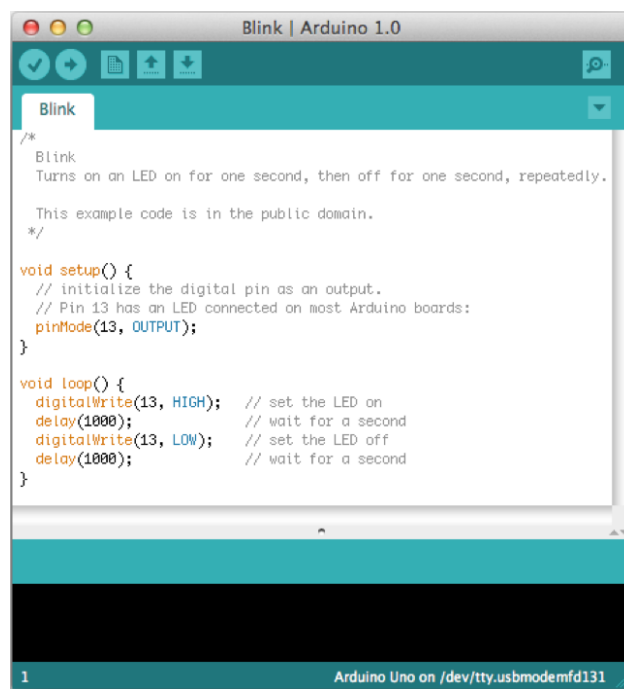


Figura 12. Programa de parpadeo del LED de la salida digital 13 en el IDE de Arduino

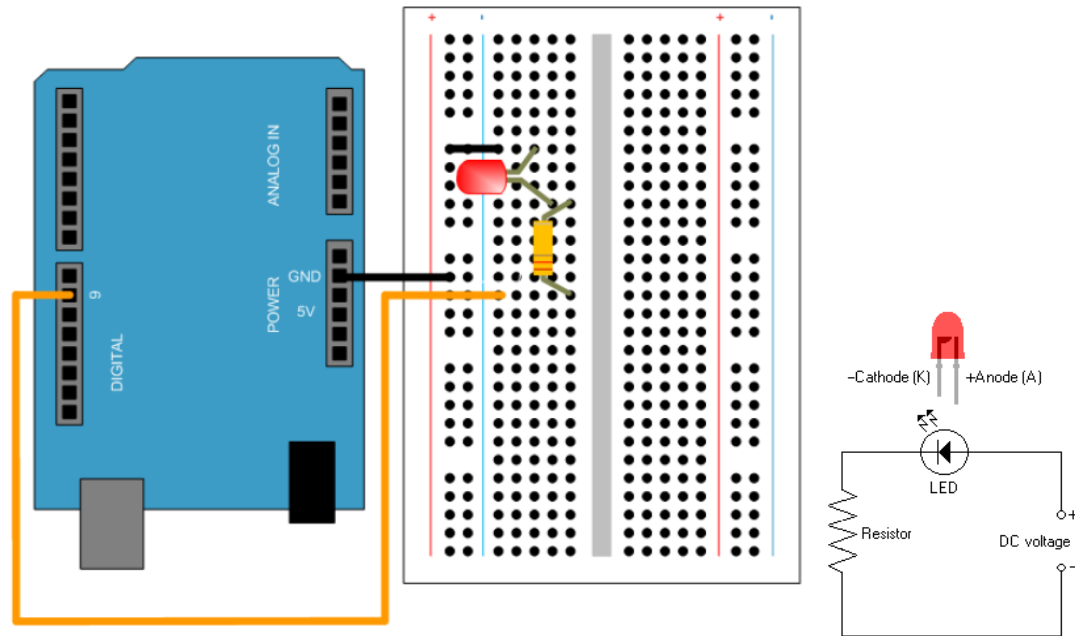


Figura 13. Montaje de un LED en la bread-board

Funciones relacionadas con las E/S digitales:

| | |
|--|---|
| <code>pinMode(pin, mode):</code> | Fija el modo del pin en INPUT o OUTPUT |
| <code>digitalRead(pin):</code> | Consulta el valor del pin (HIGH o LOW) |
| <code>digitalWrite(pin, value):</code> | Escribe un valor en el pin (HIGH o LOW) |

Para hacer funcionar este primer ejemplo en una placa Arduino UNO, se puede seguir la guía oficial de iniciación a Arduino en:

<http://arduino.cc/es/Guide/HomePage>

7. Complementos

7.1. Fritzing

Fritzing es un programa de software libre de automatización de diseño electrónico libre dedicado a la asistencia en el diseño de prototipos de placas de pruebas y su implementación como productos finales (<http://fritzing.org/>).

Mediante este software podemos realizar el diseño de un proyecto de forma estructurada y mostrar de forma gráfica la distribución de los componentes. Fritzing no permite (de momento) simular el funcionamiento del circuito.

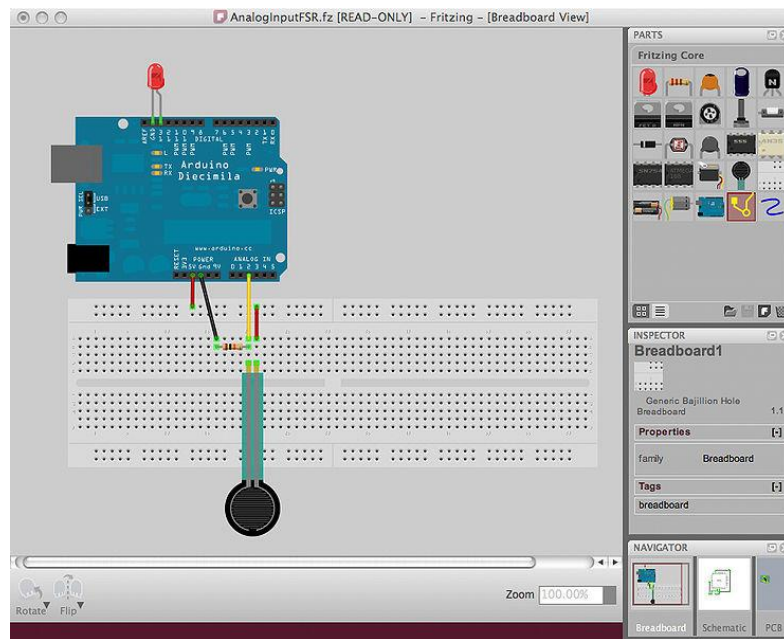


Figura 14. Ejemplo de creación de un prototipo con Fritzing

7.2. 123D Circuits

123D Circuits (<http://123d.circuits.io/users/new>) permite diseñar circuitos electrónicos a través de un navegador en línea. Se pueden crear circuitos propios desde cero o utilizar y modificar plantillas y otros proyectos compartidos públicamente. Precisa registro, pero su uso es gratuito.

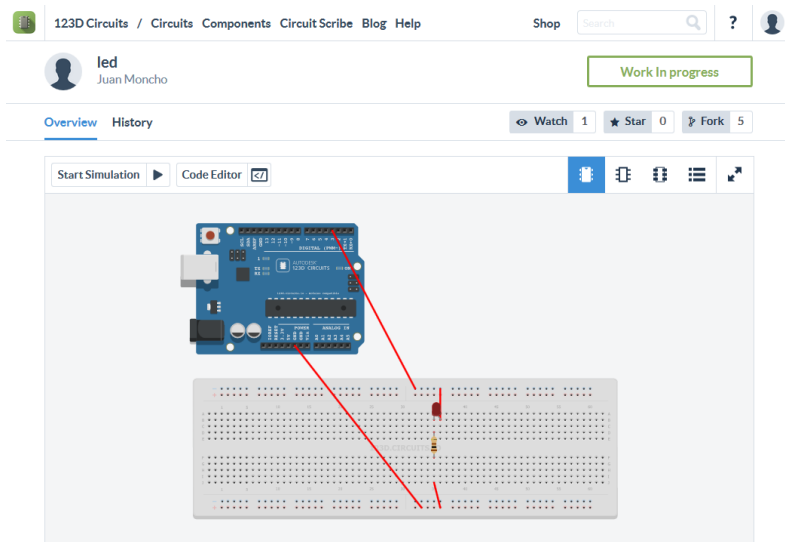


Figura 15. Simulación del parpadeo de un LED con 123DCircuits

Existen otros simuladores disponibles que no precisan una placa física Arduino conectada como Simuino, ArduSim, Emulino, Virtronics, etc. Dada la popularidad en la actualidad de Arduino y su filosofía de software libre, surgen continuamente nuevos recursos disponibles en la Red relacionados con esta plataforma.

8. Ejercicios

1. Identificar los siguientes componentes del LabKit y rellenar la siguiente tabla.

| Componente | Imagen | Función / especificaciones |
|--------------------------------|--------|----------------------------|
| Display LCD | | |
| Servomotor | | |
| Potenciómetro 100K Ω | | |
| Resistencia 1M Ω | | |
| Resistencia 470 Ω | | |
| Altavoz piezo- eléctrico | | |

2. Implementar el programa de parpadeo de LED para que encienda y apague un LED independiente sobre la placa de prototipado conectado a la salida digital 9 (ver Figura 12) a un intervalo de 1.5 segundos. Seguir el siguiente recorrido:
 - a. Crear el esquemático con Fritzing.
 - b. Cargar el programa en Arduino y comprobarlo.
3. Modificar el anterior prototipo para que se encienda únicamente cuando se pulse un botón interruptor que está conectado a la entrada digital 7.

Más información sobre el uso del botón:

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

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

4. Secuencia de leds. Se trata de encender y apagar 4 leds secuencialmente. Los leds deben estar conectados a los pines 5, 6, 7 y 8. Se deben encender y posteriormente apagar los leds desde el pin 5 al 8, con un tiempo de duración de encendido y apagado de 200 milisegundos. Posteriormente, la secuencia de encendido y apagado se invierte. El efecto que se persigue es similar al de "El coche fantástico".
 - a. Representar el esquema de conexiones mediante Fritzing.
 - b. Implementar en una placa real Arduino UNO.

Forma de Entrega:

Subir un fichero (.PDF, .DOC, .ODT o presentación) con la respuesta a cada uno de los ejercicios, incluyendo:

- Ejercicio 1: Tabla rellena
- Ejercicios 2-4:
 - Equema en Fritzing del prototipo (imagen)
 - Fotografía o enlace a vídeo del circuito en ejecución
 - Código fuente del programa transferido.

Forma de Entrega:

Subir un fichero (.PDF, .DOC, .ODT o presentación) con la respuesta a cada uno de los ejercicios, incluyendo:

- Ejercicio 1: Tabla rellena
- Ejercicios 2-4:
 - Equema en Fritzing del prototipo (imagen)
 - Fotografía o enlace a vídeo del circuito en ejecución
 - Código fuente del programa transferido.

9. Referencias

- [1] Arduino Website. *Arduino - HomePage*. Disponible en: <http://arduino.cc/>
- [2] J. R. Lajara Vizcaíno and J. Pelegrí Sebastià, *Sistemas integrados con Arduino*. Barcelona: Marcombo, 2014.
- [3] Rugged circuits. *10 Great Ways to Destroy an Arduino*. Disponible en: <http://www.ruggedcircuits.com/10-ways-to-destroy-an-arduino>
- [4] A. G. González. (2014). *Un documental sobre la Historia de Arduino*. Disponible en: <http://panamahitek.com/un-documental-sobre-la-historia-de-arduino/>
- [5] Cooking Hacks. (2014). *Cooking Hacks. Packs Kits. Lab Kit. Arduino Kit*. Disponible en: <http://www.cooking-hacks.com/arduino-lab-kit>