



Universidad de Valladolid

TRABAJO FIN DE MÁSTER

MÁSTER EN PROFESOR DE EDUCACIÓN
SECUNDARIA OBLIGATORIA Y
BACHILLERATO, FORMACIÓN PROFESIONAL
Y ENSEÑANZAS DE IDIOMAS

Especialidad de Tecnología e Informática

**ACTIVIDADES DE AULA EN EL ÁMBITO
DE LA TECNOLOGÍA CON ARDUINO**

**CLASSROOM ACTIVITIES IN THE FIELD
OF TECHNOLOGY WITH ARDUINO**

Autor:

D. José María Martínez Tejero

Tutor:

Dña. Carmen Hernández

Valladolid, 6 de Junio de 2018

Agradecimientos

En primer lugar, me gustaría agradecer a mis padres María Cruz y Gerardo, dos personas que de manera significativa han marcado mi pasado, que me han enseñado y transmitido unos valores humanos fundamentales, que siempre me han estado apoyando en todo, y que pese a las dificultades que me he encontrado a lo largo de la vida, siempre han hecho que pudiera sacar adelante todo lo que me he ido proponiendo, enseñándome que la felicidad se encuentra dentro uno mismo, y que el esfuerzo y la pasión deben de ser dos de los motores fundamentales de la vida.

En segundo lugar a mi hijo Sergio, principal estímulo y motor de mi vida, por quererme y ayudarme como lo hace. Él es mi razón de vida en muchos aspectos y todo un orgullo para mí.

En tercer lugar para mi pareja Carmen Taranilla, por su ayuda y por estar a mi lado en momentos difíciles. Ella es, sin ninguna duda, artífice de que hoy yo me encuentre escribiendo estas líneas.

En cuarto lugar, a todos mis compañeros del máster y amigos por su ayuda, apoyo y comprensión durante todo este tiempo. ¡¡Sois estupendos chic@s!!.

Y en quinto lugar pero no por ello menos importante, a mi tutora Carmen Hernández, por su inestimable ayuda durante la elaboración de este trabajo. Ella ha sido un referente en este largo proceso, y sus consejos y orientaciones han permitido que bajo un largo trabajo, este TFM vea al fin la luz.

Resumen

La hipertecnificada sociedad del siglo XXI en la que vivimos, rodeados de tecnología miremos allá dónde miremos, ha cambiado nuestra forma de vivir y de relacionarnos. Lo que hace solamente unos pocos años era inimaginable, ya está aquí. Vivimos rodeados de objetos tecnológicos, desde nuestro pequeño robot aspirador, hasta nuestro teléfono móvil, más que un teléfono, una verdadera computadora con multitud de sensores como pueden ser GPS, acelerómetro, giroscopio, magnetómetro, etc, que hacen que la principal función del mismo, que es comunicarnos mediante la voz haya pasado a un segundo plano. En un mundo en el cada minuto que pasas está todo más automatizado, tenemos cada vez más la necesidad de relacionarnos con la tecnología en sus múltiples formas, comprendiendo y aprendiendo su manejo en muchos casos. Una buena manera de acercarnos a la tecnología es la escuela, y dentro de ella asignaturas como pueden ser la tecnología y la robótica.

Este trabajo plantea y desarrolla una serie de actividades, con el objetivo de acercar la robótica y la programación de robots a nuestros jóvenes estudiantes en general, y de 4º ESO en particular. Para ello, utilizaré la plataforma Arduino, y el entorno de programación mBlock.

Palabras clave

Arduino, Arduino UNO R3, mBlock, ESO, Competencias, Actividades.

Abstract

The hyper-technified society of the 21st century in which we live, surrounded by technology, let's look where we look, has changed our way of living and relating. What only a few years ago was unimaginable, is already here. We live surrounded by technological objects, from our little robot vacuum cleaner, to our mobile phone, more than a phone, a real computer with a multitude of sensors such as GPS, accelerometer, gyroscope, magnetometer, etc., which make the main function of it , which is to communicate through the voice has passed into the background. In a world where every minute you spend is more automated, we have more and more the need to relate to technology in its many forms, understanding and learning its management in many cases. A good way to approach technology is school, and within it subjects such as technology and robotics.

This work presents and develops a series of activities, with the aim of bringing robotics and robot programming closer to our young students in general, and of 4th ESO in particular. For this, I will use the Arduino platform, and the mBlock programming environment.

Keywords

Arduino, Arduino UNO R3, mBlock, ESO, Competitions, Activities

Índice

1. Introducción.....	8
1.1. Objetivos del trabajo.....	9
1.2. Estructura de la memoria.....	9
2. Arduino y mBlock.....	10
2.1. ¿Qué es Arduino?.....	10
2.2. Arduino UNO R3.....	16
2.2.1. Descripción y características técnicas.....	17
2.2.2. Diagrama de pines.....	17
2.2.3. Esquema de la placa electrónica.....	19
2.2.4. Programación.....	20
2.3. ¿Qué es mBlock?.....	21
2.3.1. La interfaz de mBlock.....	22
3. Contexto educativo.....	24
3.1. Objetivos generales de la etapa.....	24
3.2. Objetivos específicos de la Tecnología.....	25
3.3. Contenidos.....	27
3.4. Criterios de Evaluación y Estándares de Aprendizaje.....	27
3.5. Justificación pedagógica:.....	30
4. Actividades.....	32
4.1. Actividades de iniciación.....	34
4.1.1. Conociendo nuestra placa Arduino Uno R3.....	34
4.1.2. Encendemos un LED.....	38
4.2. Actividades de trabajo con sensores.....	41
4.2.1. El semáforo.....	41
4.2.2. Los dos pulsadores.....	45
4.2.3. Usemos el potenciómetro.....	51
4.2.4. Controlando el giro de un servomotor.....	54
4.2.5. Control de un motor DC.....	57
4.2.6. ¡Alarma!, ¡Intrusos!.....	63
4.2.7. ¡Mejor con luz!.....	68
4.2.8. Radar de velocidad.....	73
4.3. Actividades de construcción de robots.....	79
4.3.1. ¡No te caigas!.....	79
4.3.2. ¡Pilota tu robot!.....	87
4.4. Evaluación.....	94
4.4.1. Criterios de evaluación.....	94
4.4.2. Herramientas de evaluación.....	95
4.4.3. Evaluación de la unidad y de la labor docente.....	95
5. Conclusiones.....	96
6. Bibliografía y Webgrafía.....	98
7. Anexos.....	100
7.1. Anexo 1.....	100
7.2. Anexo 2.....	102
7.3. Anexo 3.....	104
7.4. Anexo 4.....	106
7.5. Anexo 5.....	108
7.6. Anexo 6.....	110

Capítulo 1 - Introducción

¿Por qué alguien que sea amante de Star-Wars no puede construir su propio robot BB-8?, ¿o su propio R2-D2?, los dos a tamaño real, usando materiales que se pueden encontrar en casa y una plataforma electrónica de código abierto como puede ser Arduino. ¿Por qué no pensar en construir nuestra propia consola de videojuegos del tamaño de una tarjeta de crédito, o un sistema de alarma para puertas y ventanas de nuestra propia casa, o un sistema para controlar las luces de nuestro hogar con sensor de movimiento incluido etc, y todo ello sin tener, en un principio, grandes conocimientos científico-tecnológicos, solamente con ilusión y un poquito de esfuerzo.

¿Quién de nosotros no fue una vez un niño que soñaba con poder construir o controlar todo tipo de artilugios que veíamos principalmente en la televisión?, ¿quién no ha soñado alguna vez con estar dentro de una nave espacial recorriendo el inmenso y desconocido espacio exterior, o poder disparar una pistola láser, o construir una máquina del tiempo y poder viajar en ella, tanto al futuro como al pasado?. Todos estos sueños y pensamientos, y muchos más que aquí no están expresados, nos han acompañado y en algunos casos nos acompañan en diferentes etapas de nuestra vida, y es ahora, cuando la tecnología está a nuestro alcance, cuando podemos convertir muchos de estos sueños en realidad, o al menos incentivar en nuestros hijos, alumnos, conocidos, etc, el uso de esa tecnología, pero no solamente desde un punto de vista de usuario de la misma, si no desde el punto de vista de creadores o inventores de nuestros propios objetos tecnológicos.

Hasta ahora la tecnología, a pesar de que nos rodea por todas partes, ha sido para la mayoría de personas una gran desconocida. Todos sabemos cómo manejar distintos aparatos tecnológicos, desde los teléfonos móviles y ordenadores, hasta la más sencilla de las tostadoras de pan que podemos tener en nuestro hogar. Pero, ¿seríamos capaces de integrar un módulo de control Bluetooth en nuestra tostadora para poder controlarla con nuestro móvil?, seguramente que no en la mayoría de los casos. Estamos demasiado habituados a ser “consumidores” y no “creadores” de tecnología y ese es el cambio que debemos de fomentar. Es ahora, cuando las plataformas de prototipos electrónicos de código abierto están en pleno auge y, cuando sus productos además de baratos, pueden ser aprovechados por todo el mundo, aunque no se tengan grandes conocimientos ni de programación ni de electrónica, el momento de inculcar en los más jóvenes el interés no sólo por el uso de la tecnología, sino por la creación e invención de los propios objetos tecnológicos.

Considero que es la escuela y dentro de ella la ESO, la que puede y debe realizar una importante misión en este sentido. Dentro de la asignatura de Tecnología se puede, además de tratar los temas propios de la asignatura reflejados en la LOMCE, inculcar a los más jóvenes ese espíritu creador y de aprendizaje tecnológico, utilizando para ello una plataforma de prototipos electrónicos de código abierto como puede ser Arduino, y mediante una serie de distintas actividades, conseguir despertar el interés por la tecnología desde un punto de vista práctico, no solamente teórico.

1.1 Objetivos del trabajo

El desarrollo de este trabajo fin de máster busca alcanzar básicamente los siguientes objetivos:

- Analizar Arduino como tecnología para impartir robótica dentro de 4º ESO.
- Analizar y fomentar el uso de mBlock como lenguaje de programación para entornos Arduino.
- Proponer una serie de actividades con Arduino bajo mBlock que potencien la imaginación y la creatividad en la asignatura de tecnología en 4º ESO.

Paralelamente se plantean otra serie de objetivos en este trabajo que complementan a los dos anteriores sirviéndoles de apoyo y, a la vez, de desarrollo. Estos objetivos son:

- Evidenciar la importancia de la robótica, y en concreto de Arduino en la sociedad actual.
- Reflejar las ventajas de la utilización de Arduino en la ESO.
- Reflejar las ventajas de la programación con el entorno mBlock desarrollado para Arduino.

1.2 Estructura de la memoria

La presente memoria está estructurada de la siguiente forma:

1. Introducción: En este capítulo realizo una pequeña introducción en la cual reflejo mi punto de vista personal sobre el acercamiento de la tecnología a todas las personas en general y a los jóvenes en particular, además, de exponer los objetivos que pretendo con este trabajo.
2. Arduino: En este capítulo describo qué es Arduino, en qué se basa. Además, describo la placa Arduino UNO R3 que voy a utilizar en las actividades que presentaré más adelante en este mismo trabajo, así como el lenguaje de programación por bloques mBlock que utilizaré para las mismas.
3. Contexto educativo: Este capítulo está dedicado a los *objetivos, contenidos y estándares* de aprendizaje de la asignatura de Tecnología en el currículum de secundaria, más concretamente en el currículum de 4º de la ESO, y se presenta una justificación pedagógica a la introducción de Arduino en dicho currículum.
4. Actividades: Capítulo dedicado por entero a la descripción de las actividades propuestas y todo el desarrollo de las mismas.
5. Conclusiones: En este capítulo expresaré mis conclusiones finales sobre varios aspectos de este trabajo.
6. Bibliografía y Webgrafía: Este capítulo contendrá las fuentes en las que me he basado a la hora de realizar este trabajo.

Capítulo 2 - Arduino y mBlock

2.1- ¿Qué es Arduino?

Tomando como punto de partida las definiciones sobre Arduino que podemos encontrar tanto en su página oficial [1], en la Wikipedia [2] y otras páginas web donde aprender Arduino, como el blog de Luis Araya [3], podemos definir Arduino como 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 multidisciplinares. En la figura 1 podemos ver el logo de Arduino.



Figura 1: Logo Arduino. Fuente: www.Arduino.cc

Por otro lado, Arduino nos proporciona un software consistente en un Entorno de Desarrollo Integrado (IDE) que implementa el lenguaje de programación de Arduino y el Gestor de Arranque de Dispositivo (bootloader) ejecutado en la placa. Las principales características de este entorno son:

- Su sencillez
- Su facilidad de uso

La figura 2 nos muestra el IDE de Arduino.

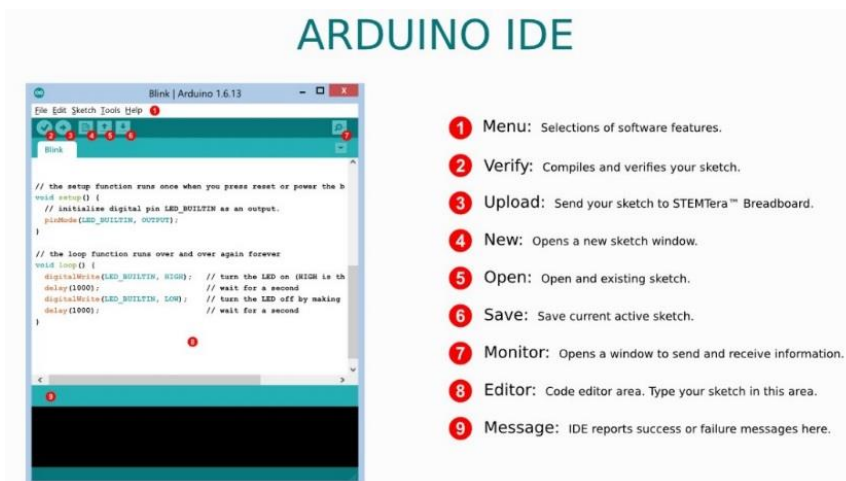


Figura 2: IDE Arduino. Fuente: www.Arduino.cc

¿Para qué sirve Arduino?

Arduino se puede utilizar para desarrollar elementos autónomos, conectando estos elementos al Arduino e interactuar tanto con el hardware como con el software de los mismos. Con Arduino podemos controlar un elemento, por ejemplo un motor que nos suba o baje una persiana de nuestra casa, basándonos en la luz existente en una habitación, gracias a un sensor de luz conectado al Arduino, o bien podemos controlar esa misma persiana desde nuestro teléfono móvil gracias a un controlador bluetooth conectado a nuestro Arduino.

Un ejemplo de los componentes y sensores que podemos conectar a nuestro Arduino lo tenemos en la figura 3.

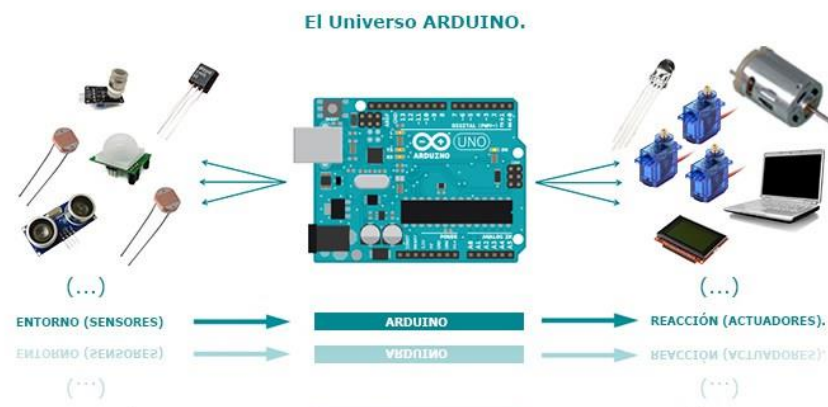


Figura 3: Componentes para Arduino. Fuente: aprendiendoarduino.wordpress.com

¿Qué puede hacer Arduino?

Arduino es una plataforma para programar un microcontrolador y por lo tanto podemos hacer con Arduino todo lo que se puede hacer con un microcontrolador, que dependerá principalmente del tipo del mismo y de los periféricos que le conectemos. El límite está en nuestra imaginación.

Por otro lado, **Arduino es también un sistema autónomo programado que realiza una o varias tareas específicas**, o lo que es lo mismo, un Arduino puede hacer las tareas de un autómatas e intercambiar datos con un SCADA (Supervisión, Control y Adquisición de Datos). Con Arduino también podemos hacer “smart” las cosas, es decir, que se conecten entre ellas mismas, a internet, etc [3].

Algunos ejemplos de dispositivos construidos con Arduino son:

- Control de un telescopio.
- Smart-Watch.
- Un analizador lógico.
- Juego de Ping-Pong.

Hay otro factor importante en el éxito de Arduino, este factor no es otro que la comunidad que apoya todo este desarrollo, comparte conocimiento, elabora librerías para

facilitar el uso de Arduino y publica sus proyectos para que puedan ser replicados, mejorados o servir de base para otro proyecto relacionado.

En resumen, podemos decir que [3]:

Arduino = Hardware + Software + Comunidad

Hardware Arduino

El Hardware Arduino es básicamente una placa con un microcontrolador. Un **microcontrolador** (μC , UC o MCU) no es otra cosa que es un circuito integrado programable, capaz de ejecutar las órdenes que previamente hemos grabado en su memoria. Está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica [3].

Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento (CPU), memoria y periféricos de entrada/salida. Las características de un Microcontrolador:

- Velocidad del reloj u oscilador
- Tamaño de palabra
- Memoria: SRAM, Flash, EEPROM, ROM, etc..
- I/O Digitales
- Entradas Analógicas
- Salidas analógicas (PWM)
- DAC (Digital to Analog Converter)
- ADC (Analog to Digital Converter)
- Buses
- UART
- Otras comunicaciones.

En la figura 4 podemos ver la estructura interna de un microcontrolador.

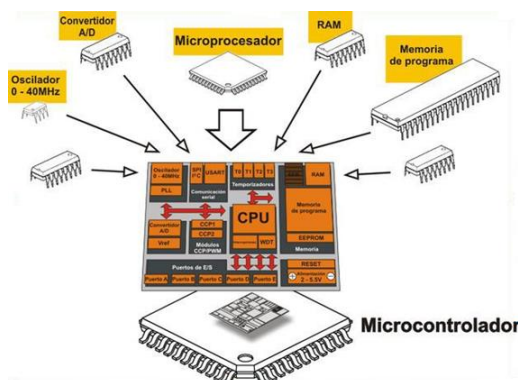


Figura 4: Estructura interna de un MCU. Fuente: aprendiendoarduino.wordpress.com

Arduino también dispone de una amplia variedad de placas y Shields (placas compatibles que se pueden colocar en la parte superior del Arduino y que permiten

extender las capacidades del mismo) para usar, dependiendo de nuestras necesidades. Algunos ejemplos de placas de Arduino son:

- Arduino Uno
- Arduino Mega
- Arduino Ethernet
- Arduino Otto
- Arduino Leonardo
- Arduino Micro
- Arduino Due

En cuanto a Shields, de Arduino:

- Ethernet Shield
- Arduino Wifi Shield
- Arduino Wifi Shield 101
- Arduino GSM Shield
- Arduino Motor Shield

Las Shields se pueden comunicar con Arduino bien por algunos de los pines digitales o analógicos o bien por algún bus como el SPI, I2C o puerto serie, así como usar algunos pines como interrupción. Además estas Shields se alimentan generalmente a través del Arduino mediante los pines de 5V y GND. Los pines que se usan para la conexión de estos Shields podemos verlos en la figura 5.

Es importante destacar que cada Shield de Arduino debe tener el mismo factor de forma que el estándar de Arduino, con un espaciado de pines concreto para que solo haya una forma posible de encajarlo.

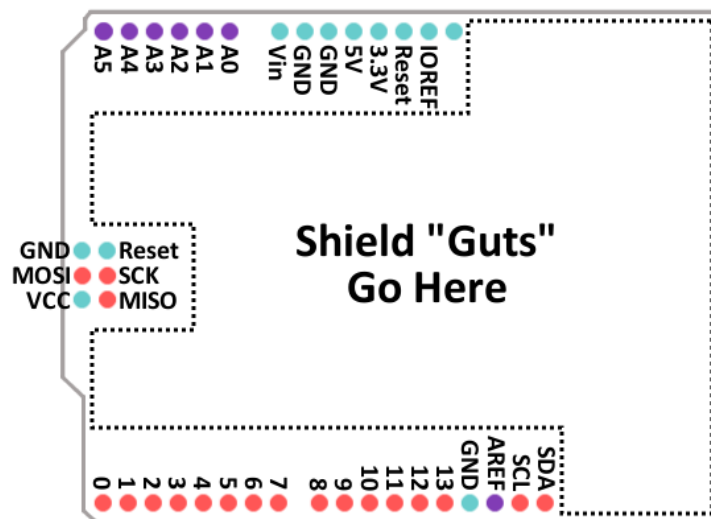


Figura 5: Esquema factor de forma Arduino. Fuente: aprendiendoarduino.wordpress.com

Dentro del mercado, existe también una extensa variedad de placas Arduino y compatibles. Entre las placas Arduino no oficiales, aunque compatibles, algunos ejemplos son:

- [Almond PCB](#)
- [Bambino 210](#)
- [Banguino](#)
- [Boarduino](#)
- [bq ZUM BT-328](#)
- [ChipKIT Fubarino](#)
- [ChipKIT Lenny](#)
- [ChipKIT UNO32](#)

Además del hardware de Arduino tenemos infinidad de placas que son clones de las placas de Arduino y luego están las placas compatibles con Arduino, que son aquellas placas que no están basadas en las placas originales de Arduino (y que pueden usar otros microcontroladores diferentes), pero que se programan igual que Arduino, incluso con el mismo IDE [3].

Dentro del entorno Arduino, podemos encontrar placas de otra serie de fabricantes de chips y controladores compatibles como pueden ser **Microchip** o **Mediatek** y otros fabricantes que se han aliado a Arduino.org, como **ST Microelectronics**, para sacar al mercado nuevos modelos de Arduinos, como por ejemplo el **Arduino Mega**, que muestro en la figura 6.

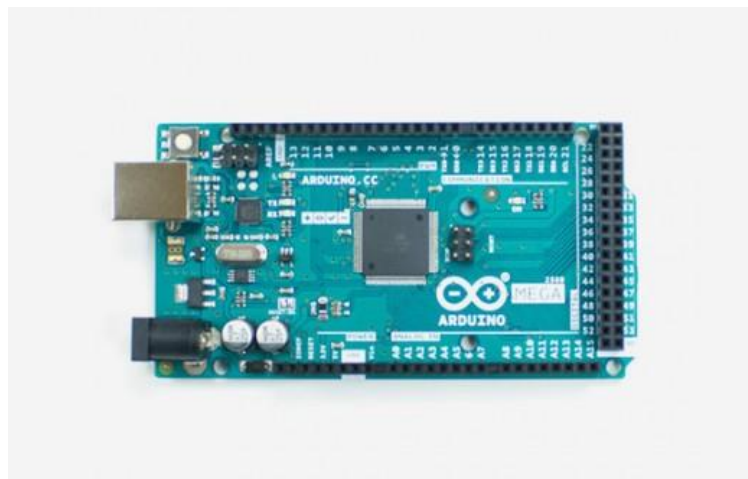


Figura 6: Placa Arduino Mega. Fuente: www.store.arduino.cc

Software Arduino

El software de Arduino es un Entorno de Desarrollo Integrado o IDE (Integrated Development Environment), es decir, una aplicación informática compuesta por un conjunto de herramientas de programación [2].

El IDE de Arduino es un entorno de programación que ha sido empaquetado como un programa de aplicación; es decir, consiste en un editor de código, un compilador, un

depurador y un constructor de interfaz gráfica (GUI). Además incorpora las herramientas para cargar el programa ya compilado en la memoria flash del hardware.

En principio el IDE de Arduino sólo tenía soporte para las placas Arduino y los clones con los mismos microcontroladores que los Arduinos oficiales. A partir de la versión 1.6.2, se han añadido soporte a otros microcontroladores y placas, y la **gestión de librerías y de placas** ha sido muy mejorada respecto a las versiones anteriores [1] y [3].

En la figura 7 podemos ver un ejemplo de IDE Arduino.

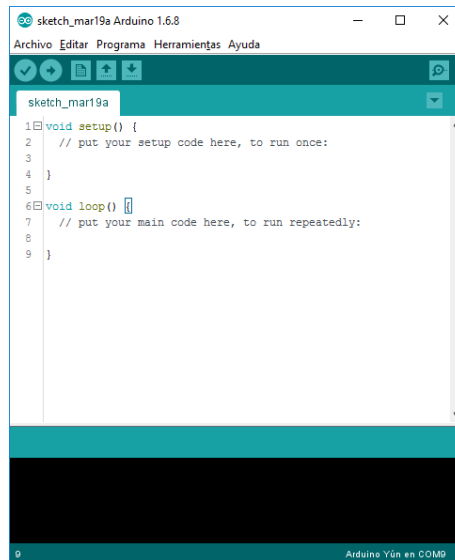


Figura 7: IDE Arduino 1.6.8. Fuente: aprendiendoarduino.wordpress.com

Comunidad Arduino

Sin duda, uno de los factores principales de éxito de Arduino es la comunidad que está apoyando este proyecto, y que día a día difunde contenidos y resuelve las dudas que podamos tener [3]. En Internet tenemos a nuestra disposición, todo tipo de cursos, tutoriales, herramientas de consulta, proyectos, etc... que nos ayudarán a usar Arduino con facilidad.

Los dos primeros sitios web que nos ayudarán a trabajar con Arduino son:

- La página oficial de Arduino [4].
- El playground de Arduino [5].

Arduino playground es una wiki donde todos los usuarios de Arduino pueden contribuir. Es el lugar donde publicar y compartir código, diagramas de circuitos, tutoriales, trucos, cursos, etc, y sobretodo el lugar donde buscar cuando tengamos dudas, problemas, necesitemos encontrar una librería adecuada para nuestro proyecto, etc... Esta es la base de datos de conocimiento por excelencia de Arduino.

También existen lugares no oficiales de Arduino donde resolver nuestras dudas. Dos de ellas son:

- Stackexchange [17].

- Stackoverflow [18].

Otros lugares de encuentro de la comunidad son los diversos portales donde se publican proyectos con Arduino, y por último, también hay espacios locales para la comunidad, son los llamados hacklabs, hackerspace, makerspace, etc. Éstos son, aunque hay ciertas diferencias entre unos y otros, sitios físicos donde gente con intereses en ciencia, nuevas tecnologías y artes digitales o electrónicas se pueden conocer, socializar y colaborar. Estos sitios ponen al alcance de aficionados y estudiantes la infraestructura y ambiente necesarios para desarrollar sus proyectos tecnológicos [3]. Tres ejemplos:

- Hackster [20].
- Arduino project hub [1].
- OpenHardware [21].

2.2- Arduino UNO R3

Arduino Uno R3 es una placa electrónica basada en el microcontrolador ATmega328 [5] y es la placa con la que llevaré cabo las actividades que propondré en este trabajo, debido fundamentalmente a cuatro razones:

- Es una placa bastante potente, pero a la vez fácil de manejar.
- Es la placa más común y extendida del mercado, con todas las ventajas que eso conlleva en cuanto a actividades, tutoriales, resolución de problemas, etc.
- Es una placa muy barata; se puede encontrar en internet desde unos 6 euros.
- El software de programación que voy a utilizar, mBlock, ha sido probado infinidad de veces y funciona perfectamente con esta placa, además de ser uno de los más utilizados últimamente.

Podemos ver la placa con la que vamos a trabajar en la figura 8.

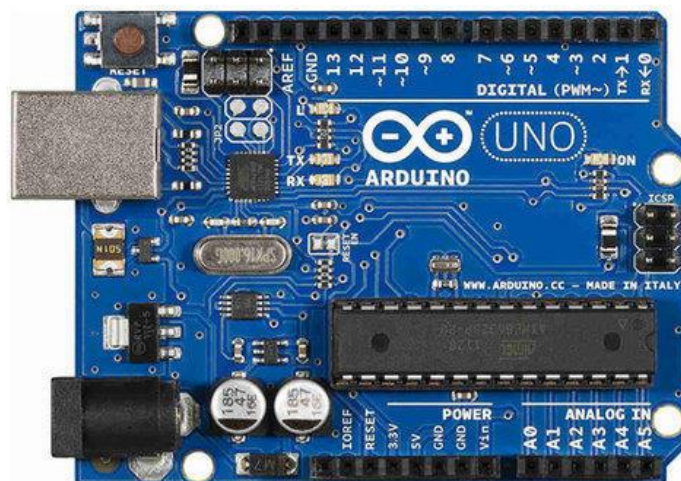


Figura 8: Placa Arduino UNO R3. Fuente: www.indiamart.com

Todo lo relacionado con la placa Arduino Uno R3, su descripción, características técnicas, diagrama de pines, esquema de la placa electrónica y programación que presento, están basados en la página oficial de Arduino Uno [5]:

2.2.1- Descripción y características técnicas

Las características técnicas de la placa electrónica vienen recogidas en la siguiente tabla:

Microcontrolador	ATmega328P
Tensión de funcionamiento	5 V
Voltaje de entrada (recomendado)	7 – 12 V
Voltaje de entrada (límite)	6 – 20 V
Digital pines I/O	14 (de los cuales 6 proporcionan una salida PWM)
PWM Digital pines I/O	6
Pines de entrada analógica	6
Corriente DC por pin I/O	20 mA
Corriente DC para pin 3,3 V	60 mA
Memoria Flash	32 KB ATmega328P de los que 0,5 KB son utilizados por el sector de arranque
SRAM	2 KB ATmega328P
EEPROM	1 KB ATmega328P
Velocidad de reloj	16 MHz
Longitud	68.6 mm
Anchura	53,4 mm
Peso	25 g

2.2.2- Diagrama de pines

La figura 9 nos muestra el diagrama de pines que forman parte de la placa Arduino Uno R3.

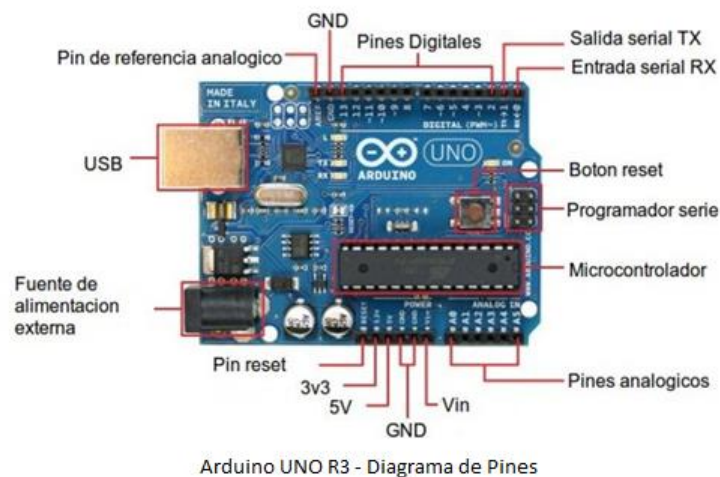


Figura 9: Esquema placa Arduino Uno R3. Fuente www.infootec.net

La placa electrónica Arduino Uno R3 puede ser alimentada de varias formas:

- Con un cable USB conectado al ordenador; el conector es un conector USB tipo B y nos sirve también para programarlo una vez conectado al ordenador.
- Mediante una fuente externa; cuenta con un zócalo donde se conecta un Jack de 2,1mm para conectar un adaptador que se encuentre entre los rangos de 7 – 12v que es la tensión recomendada.

Los pines de la placa Arduino Uno R3 también podemos verlos representados en la figura 10.

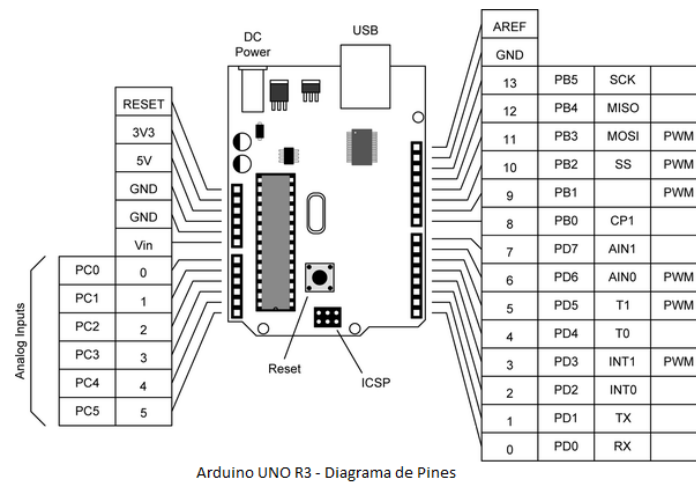


Figura 10: Diagrama de pines Arduino Uno R3. Fuente: /www.infootec.net

Describimos brevemente la función de cada pin:

Pin VIN:

Este pin se puede usar de varias formas:

- Si tenemos una fuente de alimentación conectada mediante un adaptador, lo que podemos hacer es obtener la alimentación para conectar otro dispositivo; pero tenemos que tener en cuenta que la placa no regulará la tensión y obtendremos la misma tensión que tenga el adaptador.
- Si tenemos conectado el USB, la tensión será regulada a 5v.
- Y si tenemos una fuente de alimentación externa como por ejemplo pilas, el borne positivo de la pila ira conectado al pin VIN y el borne negativo de la pila al pin GND; en este caso, si la pila saca 10v la placa regulará la tensión a 5v.

Pin GND:

El pin GND es la tierra.

Pin 5v:

Este pin tiene varias funciones; podemos alimentar la placa mediante este pin, siempre que tengamos la fuente externa regulada a 5v. Por otro lado si tenemos la placa alimentada tanto por el Jack como por USB, se puede alimentar otro componente con una tensión regulada de 5v.

Pin 3.3v:

Por este pin sacamos una tensión de 3.3v que es alimentada mediante el conector Jack o el USB. Los 3.3v se utilizan para alimentar dispositivos que requieren una tensión baja.

Pines de entradas analógicas:

La placa de Arduino cuenta con 6 pines de entradas analógicas, que van desde el pin A0 al A5, que proporcionan 10bits, llamados bits de resolución. La tensión que miden va de 0 a 5v, aunque es posible cambiar su rango usando una función con el pin AREF.

Pin IOREF:

El pin IOREF es una copia del pin VIN y se utiliza para indicar a los demás dispositivos conector a la placa que las tensiones de los pines de entrada y salida son 5v.

Pin RESET:

Este pin tiene el mismo funcionamiento que el botón RESET, se utiliza para reiniciar el microcontrolador.

Pines de entradas y salidas digitales:

Las entradas y salidas digitales son 14 y van desde el pin 0 al 13 y ofrecen una tensión de 5v.

Pines A5 SCL y A4 SDA:

Se pueden utilizar para conectar dispositivos que lleven a cabo comunicaciones mediante la librería Wire.

Pin AREF:

Ofrece un voltaje de referencia para las entradas analógicas.

Pines 1 TX y 0 RX:

Estos pines se utilizan para recibir y transmitir datos en serie.

2.2.3- Esquema de la placa electrónica

Este esquema lo tenemos a nuestra disposición en el **anexo 1** del presente documento.

2.2.4- Programación

El entorno de programación más utilizado es Arduino IDE, descargable desde la página oficial de Arduino [3].

La figura 11 nos muestra la ventana de descarga del software de Arduino.



Figura 11: Descarga de software Arduino. Fuente: www.arduino.cc

Lo único que debemos hacer es seleccionar el sistema operativo desde donde vamos a ejecutar nuestro entorno Arduino IDE, realizar la descarga y la instalación y ya lo tendremos todo preparado para programar el chip.

Yo he elegido otro lenguaje de programación para realizar las actividades con Arduino. Utilizaré el lenguaje de programación por bloques disponible para Arduino, **mBlock**. En el siguiente punto argumento una serie de razones por las que he elegido mBlock frente a otros entornos de programación por bloques para Arduino, como pueden ser **Scratch For Arduino** (S4A) o **Ardublock**. En la figura 12 tenemos representado el logo de mBlock.



Figura 12: Logotipo de mBlock. Fuente: www.mBlock.cc

Este software, al igual que el IDE de Arduino es de libre distribución, y lo podemos encontrar en internet, en la página oficial de mBlock [6].

2.3- ¿Qué es mBloc?

Las páginas oficiales de mBlock [6] y Makeblock [8] nos lo definen como un entorno gráfico de programación basada en el editor Scratch 2.0 para introducir la robótica y la programación en entornos Arduino, de una manera sencilla y divertida, a niños y adolescentes en escuelas y centros de formación.

mBlock = Scratch + Arduino

Es también un lenguaje de programación orientado a la educación en STEAM. Por otro lado, STEM, es el acrónimo en inglés de los nombres de cuatro materias o disciplinas académicas: Science, Technology, Engineering y Mathematics.

Las iniciativas o proyectos educativos englobados bajo la denominación STEM, pretenden aprovechar las similitudes y puntos en común de estas cuatro materias para desarrollar un enfoque interdisciplinario del proceso de enseñanza y aprendizaje, incorporando contextos y situaciones de la vida cotidiana, utilizando para ello todas las herramientas tecnológicas necesarias.

Si a estos conocimientos, le añadimos las Artes, tendremos STEAM [9].

STEAM = STEM + Artes

Algunas de las características de mBlock [6] son:

- Permite programar nuestros robots de forma inalámbrica, bien vía bluetooth o por medio de Wi-Fi 2,4 GHz.
- Nos permite traducir los bloques de Scratch a código fuente de Arduino, lo que nos permite ver el equivalente en código de programa al entorno gráfico que hayamos diseñado, de forma que nos ayuda en la transición de entorno gráfico a código.
- Podemos probar en tiempo real el programa que realicemos en Scratch sin necesidad de grabarlo en la placa.
- Una vez probado, tenemos la posibilidad de grabarlo de forma permanente en nuestro robot.
- Permite desarrollar nuestras propias extensiones y librerías para utilizar distintos componentes y Shields.

Además, mBlock podemos usarlo tanto con nuestras placas Arduino como con los robots educativos de Makeblock:

- *Robot educativo mBot* para los que acaban de iniciarse.

- *Starter Kit Scout* para aquellos que quieran iniciarse con un kit básico.
- *Robot mBot Ranger* para aquellos que necesiten un robot más avanzado.
- *Ultimate Kit*, para aquellos más experimentados.

En cuanto a las ventajas que podemos encontrar en mBlock respecto a otros entornos de programación en bloques para Arduino, son principalmente [8]:

- A diferencia del *Scratch For Arduino* (S4A), en el cual el robot tiene que tener siempre comunicación con el PC (bien por cable o por Bluetooth), con mBlock podemos elegir la forma en que queremos trabajar, ejecutándolo en tiempo real en el Pc (como el S4A) o cargándolo directamente en la placa, con lo que no necesitamos la conexión con el PC. Esto nos aporta una gran ventaja a la hora de construir robots autónomos.
- Nos permite definir qué pines vamos a querer como entradas y cuáles como salidas, a diferencia del S4A en el que vienen predefinidos.
- Nos permite descargar unos paquetes llamados extensiones, que no vienen con la versión standard de mBlock, y que nos permite el trabajo con componentes que nos eran imposibles de utilizar con S4A.

2.3.1- Interfaz mBlock

La interfaz es muy sencilla e intuitiva. Tenemos un ejemplo de la interfaz mBlock en la figura 13.

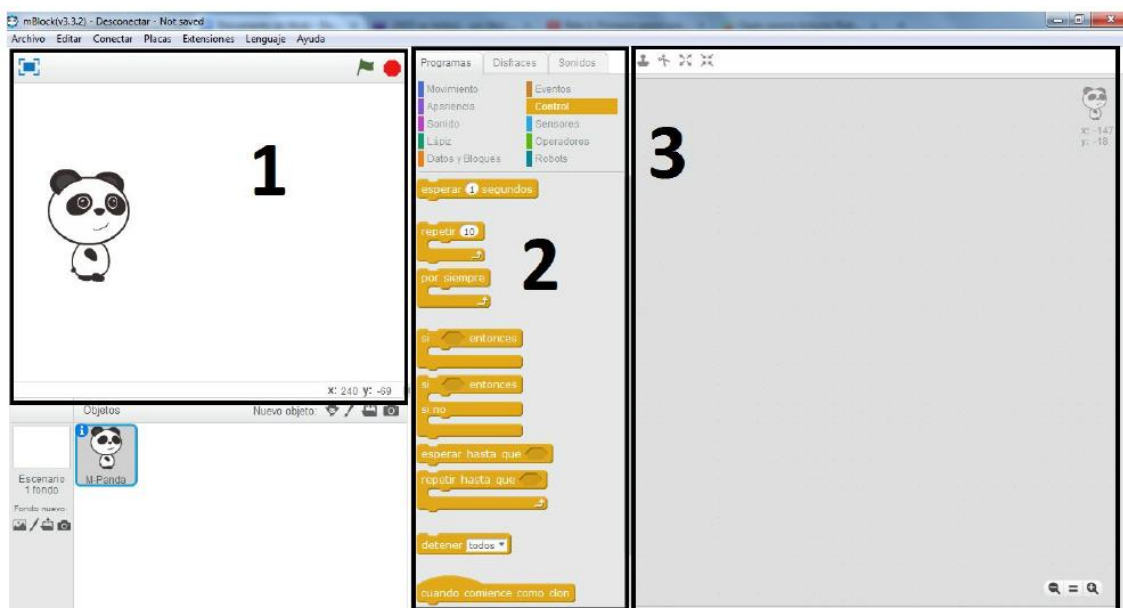


Figura 13: Esquema interfaz mBlock. Fuente: www.atlantistelecom.com

En el lado izquierdo (1) tenemos representado el escenario, que es donde se verá representado el videojuego que programemos. En el centro (2) están los bloques de programación divididos por categorías, es decir, es la parte donde se encuentran las piezas con las que programaremos nuestro robot o videojuego. En el lado derecho de la interfaz (3), es donde se construye el programa, es decir, donde se colocan los bloques que vamos eligiendo en nuestra programación.

Los bloques de programación poseen distintas funcionalidades asociadas a distintos colores, por ejemplo, los “eventos” están asociados al marrón, los “sensores” al azul marino, los “operadores” al verde etc. Estas funcionalidades se pueden dividir en dos grandes grupos:

1. **Bloques para programar videojuegos:** las categorías de bloques que podemos utilizar para programar videojuegos son las de Movimiento, Apariencia, Sonido, Lápiz, Control, Eventos, Sensores, Datos, Operadores y Bloques.
2. **Bloques para programar robots:** para programar robots utilizaremos las categorías de Robots, Operadores, Control, Eventos, Bloques y Datos.

La figura 14 nos muestra las distintas categorías de bloques, con sus respectivos colores y funcionalidades.








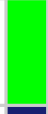


Categoría	Notas	Categoría	Notas
 Movimiento	Mueve objetos y cambia ángulos	 Eventos	Contiene activadores de eventos situado al comienzo de cada grupo de instrucciones
 Apariencia	Controla el aspecto visual del objeto, añade bocadillos de habla o pensamiento, cambia el fondo, ampliar o reducir	 Control	Contiene los bloques de lógica de programación como los bucles, condiciones, repeticiones..
 Sonido	Reproduce ficheros de audio y secuencias programables	 Sensores	Los objetos pueden interactuar con el ambiente que ha creado el usuario
 Lápiz	Control del ancho, color e intensidad del lápiz	 Operadores	Operadores matemáticos, generador aleatorio de números, comparadores..
 Datos y Bloques	Creación de variables, bloques y listas	 Robots	Bloques para controlar el robot, usar sensores, girar motores...

Figura 14: categorías de bloques en mBlock. Fuente: www.atlantistelecom.com

En el capítulo referente a las actividades que vamos a realizar con Arduino y mBlock explicaré más detalladamente el proceso de programación en dicha interfaz y el uso de los distintos bloques a medida que los vayamos necesitando.

Capítulo 3 - Contexto educativo

En el presente trabajo, se exponen algunas propuestas de actividades con Arduino bajo mBlock para llevar a cabo dentro de la asignatura de Tecnología en la Educación Secundaria Obligatoria (ESO), en concreto en cuarto curso, dado que los objetivos generales de estas actividades se basan en los que se detallan en la orden que desarrolla el currículum de Secundaria para Castilla y León [10]. Los contenidos que se pretenden introducir con ayuda de las actividades propuestas también se ajustan en gran medida a los recogidos en el currículum que dicha orden establece. Las actividades que se pretenden introducir son en principio evaluables, ya que con ellas se alcanzan la mayor parte de los contenidos, criterios de evaluación y estándares de aprendizaje evaluables desglosados en la orden.

A continuación se detallan los objetivos, contenidos, criterios de evaluación y estándares de aprendizaje evaluables del currículum de **4º ESO** para la asignatura de Tecnología [10].

3.1- Objetivos Generales de Etapa

La LOMCE [11], en su Artículo III, Capítulo 23, y el Real Decreto 1105/2014, de 26 de diciembre [12], nos definen los objetivos generales para la etapa de secundaria:

La educación secundaria obligatoria contribuirá a desarrollar en los alumnos y las alumnas las capacidades que les permitan:

- a) Asumir responsablemente sus deberes, conocer y ejercer sus derechos en el respeto a los demás, practicar la tolerancia, la cooperación y la solidaridad entre las personas y grupos, ejercitarse en el diálogo afianzando los derechos humanos como valores comunes de una sociedad plural y prepararse para el ejercicio de la ciudadanía democrática.
- b) Desarrollar y consolidar hábitos de disciplina, estudio y trabajo individual y en equipo como condición necesaria para una realización eficaz de las tareas del aprendizaje y como medio de desarrollo personal.
- c) Valorar y respetar la diferencia de sexos y la igualdad de derechos y oportunidades entre ellos. Rechazar los estereotipos que supongan discriminación entre hombres y mujeres.
- d) Fortalecer sus capacidades afectivas en todos los ámbitos de la personalidad y en sus relaciones con los demás, así como rechazar la violencia, los prejuicios de cualquier tipo, los comportamientos sexistas y resolver pacíficamente los conflictos.
- e) Desarrollar destrezas básicas en la utilización de las fuentes de información para, con sentido crítico, adquirir nuevos conocimientos. Adquirir una preparación básica en el campo de las tecnologías, especialmente las de la información y la comunicación.
- f) Concebir el conocimiento científico como un saber integrado, que se estructura en distintas disciplinas, así como conocer y aplicar los métodos para identificar los problemas en los diversos campos del conocimiento y de la experiencia.

- g) Desarrollar el espíritu emprendedor y la confianza en sí mismo, la participación, el sentido crítico, la iniciativa personal y la capacidad para aprender a aprender, planificar, tomar decisiones y asumir responsabilidades.
- h) Comprender y expresar con corrección, oralmente y por escrito, en la lengua castellana y, si la hubiere, en la lengua cooficial de la Comunidad Autónoma, textos y mensajes complejos, e iniciarse en el conocimiento, la lectura y el estudio de la literatura.
- i) Comprender y expresarse en una o más lenguas extranjeras de manera apropiada.
- j) Conocer, valorar y respetar los aspectos básicos de la cultura y la historia propias y de los demás, así como el patrimonio artístico y cultural.
- k) Conocer y aceptar el funcionamiento del propio cuerpo y el de los otros, respetar las diferencias, afianzar los hábitos de cuidado y salud corporales e incorporar la educación física y la práctica del deporte para favorecer el desarrollo personal y social. Conocer y valorar la dimensión humana de la sexualidad en toda su diversidad. Valorar críticamente los hábitos sociales relacionados con la salud, el consumo, el cuidado de los seres vivos y el medio ambiente, contribuyendo a su conservación y mejora.
- l) Apreciar la creación artística y comprender el lenguaje de las distintas manifestaciones artísticas, utilizando diversos medios de expresión y representación.

3.2- Objetivos Específicos de la Tecnología

Los objetivos específicos para la asignatura de Tecnología vienen recogidos tanto en el orden EDU-362 [10] como en el Real Decreto 1105 [12]. Estos objetivos son:

- La asignatura de Tecnología pretende que los alumnos observen en su entorno los objetos y los avances que les rodean y vean en ellos el resultado de un proceso que abarca la ciencia y la técnica, el pensamiento científico y las habilidades prácticas.
- A lo largo de la historia de la humanidad los desarrollos tecnológicos han cambiado en gran medida nuestra forma de vida, dando respuesta a una necesidad, a un anhelo o a una idea. En la educación Secundaria, esta materia busca que los estudiantes comprendan la relación del ser humano con el mundo creado por el hombre, valoren la Tecnología como un proceso ligado íntimamente al ingenio, emprendimiento y habilidad humana. No es posible entender el desarrollo tecnológico sin los conocimientos científicos, como no es posible hacer ciencia sin el apoyo de la tecnología; ambas necesitan de instrumentos, equipos y conocimientos técnicos. En la sociedad actual, todos estos campos están relacionados con gran dependencia unos de otros, pero a la vez cada uno cubre una actividad diferente. La materia Tecnología aporta al alumno “saber cómo hacer” al integrar ciencia y técnica, es decir “por qué se puede hacer” y “cómo se puede hacer”. Por tanto, un elemento fundamental de la tecnología es el carácter integrador de diferentes disciplinas con un referente disciplinar común basado en un modo ordenado y metodológico de intervenir en el entorno.
- El valor educativo de la materia está asociado tanto a su propio contenido como a la metodología. El objetivo final será la resolución de los problemas tecnológicos: desde

la identificación y formulación del problema hasta su solución constructiva mediante un desarrollo que busque la optimización de recursos. Para alcanzar este propósito es necesario integrar los conocimientos científicos y técnicos adquiridos de un modo ordenado y metódico. Con este fin se incluyen una serie de orientaciones metodológicas de carácter general para que sirvan de referencia al profesorado de Tecnología a la hora de concretar la programación del centro:

- Dado el carácter práctico, Tecnología es la materia más indicada para que el alumnado sea consciente de que los contenidos que aprende realmente son aplicables. Esta funcionalidad se va a ver reflejada en el desarrollo de un proyecto en el que los alumnos van a aplicar todos y cada uno de los conocimientos que han ido adquiriendo en forma de contenidos teóricos y problemas o casos prácticos.
- Siempre que se pueda, se aplicarán metodologías activas en las que el protagonista del proceso enseñanza-aprendizaje sea el propio alumno y no el profesor ni los contenidos que se vean en cada momento.
- En cada proyecto técnico los alumnos discutirán sobre diversos aspectos resolutivos, como, por ejemplo, tipo de herramientas que utilizarán, diferentes formas de realizar una tarea, acabados finales, presentación del producto, entre otros. Una cuestión fundamental es crear unos hábitos de trabajo adecuados evitando que realicen la fase de construcción del objeto sin haber realizado las fases previas de diseño y planificación.
- Es importante crear unos hábitos de comportamiento en el espacio de trabajo y organizar las tareas entre los distintos miembros del grupo para poder tener controlado el proceso en todo momento tanto por parte de los alumnos como del profesor.
- Los alumnos aprenden mejor si ven la posibilidad de aplicar en el mundo real los conocimientos adquiridos. En este sentido, es muy importante que se realicen salidas organizadas para que puedan ver la aplicación práctica de la tecnología en la vida real. Así pues, actividades tales como trabajos de investigación sobre soluciones tecnológicas reales, visitas a museos de la ciencia y tecnología, a centros de investigación, parques tecnológicos, estaciones de tratamiento de residuos y depuración, algunos establecimientos industriales, plantas generadoras de energía, etc., les motivarán a la hora de adquirir conocimientos relacionados con estos ámbitos.
- Las tecnologías de la información y la comunicación van a estar presentes en todo momento. No solamente a la hora del aprendizaje del manejo básico de las aplicaciones sino en la utilización práctica de software específico, simuladores, creación de documentación técnica de proyectos, búsqueda de información en Internet, presentaciones de contenidos y otras tareas que el profesor pueda proponer en las que el uso del ordenador sea necesario.
- Por último, tanto en el aula como en el taller se ha de fomentar un clima que potencie la creatividad del alumnado, el desarrollo de su autoestima personal,

la integración de distintos saberes culturales, la asunción de valores éticos y la autonomía personal.

3.3- Contenidos

Los contenidos del currículo de Tecnología de 4º ESO, según la orden EDU-362 [10] y el Real Decreto 1105 [12], se distribuyen en seis bloques:

1. «Tecnologías de la información y la comunicación». En este bloque se desarrollan contenidos vinculados a la crucial importancia que suponen hoy las TIC en la sociedad. En particular, la necesidad de tratar la información, almacenarla y transmitirla. El actual desarrollo de las telecomunicaciones hace necesario el conocimiento de los distintos tipos de comunicación tanto cableada como inalámbrica.
2. «Instalaciones en viviendas». Este bloque trata de enseñar las instalaciones características de una vivienda (componentes, funcionamiento, correcto mantenimiento y consumo energético responsable), señalando como básicas la instalación eléctrica, abastecimiento de agua, saneamiento, calefacción, gas y domótica. Asimismo, proporciona conocimiento sobre la simbología que se utiliza en dichas instalaciones y el ahorro energético.
3. «Electrónica». En este bloque se proporcionan al alumno los conocimientos básicos de la electrónica analógica (componentes y simbología), analizando circuitos elementales, y de la electrónica digital, aplicando el álgebra de Boole a problemas sencillos de puertas lógicas.
4. «Control y Robótica». Los contenidos de este bloque permitirán mostrar al alumno los distintos sistemas automáticos y sus componentes, las características básicas de un robot y los programas para su control.
5. «Neumática e Hidráulica», que proporciona los conocimientos para analizar sistemas neumáticos e hidráulicos: componentes, simbología y principios físicos de funcionamiento.
6. «Tecnología y sociedad», contenidos que permiten conocer el desarrollo tecnológico a lo largo de la historia y su repercusión en el día a día.

3.4- Criterios de Evaluación y Estándares de Aprendizaje Evaluables

También están recogidos en la orden EDU-362 [10] y en el Real Decreto 1105 [12]. Para la asignatura de Tecnología de 4º ESO son siguientes:

CONTENIDOS	CRITERIOS DE EVALUACIÓN	ESTÁNDARES DE APRENDIZAJE EVALUABLES
BLOQUE 1 «Tecnologías de la información y la comunicación»		
<p>Elementos y dispositivos de comunicación alámbrica e inalámbrica. Redes. Tipología. Publicación e intercambio de información en medios digitales. Uso seguro y responsable de los medios de publicación e intercambio de información. Conceptos básicos e introducción a los lenguajes de programación. Uso de ordenadores y otros sistemas de intercambio de información. Diseño asistido por ordenador: Herramientas CAD.</p>	<p>1. Analizar los elementos y sistemas que configuran la comunicación alámbrica e inalámbrica. 2. Acceder a servicios de intercambio y publicación de información digital con criterios de seguridad y uso responsable. 3. Elaborar sencillos programas informáticos. 4. Utilizar equipos informáticos y emplear herramientas de diseño asistido por ordenador para elaborar representaciones de objetos, planos o esquemas técnicos.</p>	<p>1.1. Describe los elementos y sistemas fundamentales que se utilizan en la comunicación alámbrica e inalámbrica. 1.2. Describe las formas de conexión en la comunicación entre dispositivos digitales. 2.1. Localiza, intercambia y publica información a través de Internet empleando servicios de localización, comunicación intergrupala y gestores de transmisión de sonido, imagen y datos. 2.2. Conoce las medidas de seguridad aplicables a cada situación de riesgo. 3.1. Desarrolla un sencillo programa informático para resolver problemas utilizando un lenguaje de programación. 4.1. Utiliza el ordenador como herramienta de adquisición e interpretación de datos, y como realimentación de otros procesos con los datos obtenidos.</p>
BOQUE 2 «Instalaciones en viviendas»		
<p>Instalaciones características: instalación eléctrica, instalación de agua sanitaria, e instalación de saneamiento. Otras instalaciones: calefacción, gas, aire acondicionado, domótica. Normativa, simbología, análisis y montaje de instalaciones básicas. Ahorro energético en una vivienda. Arquitectura bioclimática. Estudio y análisis de facturas domésticas.</p>	<p>1. Describir los elementos que componen las distintas instalaciones de una vivienda y las normas que regulan su diseño y utilización. 2. Realizar diseños sencillos empleando la simbología adecuada. 3. Experimentar con el montaje de circuitos básicos y valorar las condiciones que contribuyen al ahorro energético. 4. Evaluar la contribución de la arquitectura de la vivienda, sus instalaciones y de los hábitos de consumo al ahorro energético.</p>	<p>1.1. Diferencia las instalaciones típicas en una vivienda. 1.2. Interpreta y maneja simbología de instalaciones eléctricas, calefacción, suministro de agua y saneamiento, aire acondicionado y gas. 2.1. Diseña con ayuda de software instalaciones para una vivienda tipo con criterios de eficiencia energética. 3.1. Realiza montajes sencillos y experimenta y analiza su funcionamiento. 4.1. Propone medidas de reducción del consumo energético de una vivienda.</p>
BLOQUE 3 «Electrónica»		
<p>Señal analógica y señal digital. Electrónica analógica. Componentes básicos. Simbología y análisis de circuitos elementales. Montaje de circuitos sencillos. Electrónica digital. Sistemas de numeración: binario y hexadecimal. Álgebra de Boole, operaciones y funciones lógicas.</p>	<p>1. Analizar y describir el funcionamiento y la aplicación de un circuito electrónico y sus componentes elementales. 2. Emplear simuladores que faciliten el diseño y permitan la práctica con la simbología normalizada. 3. Experimentar con el montaje de circuitos elementales y aplicarlos en el proceso tecnológico. 4. Realizar operaciones lógicas empleando el álgebra de Boole en la resolución de problemas tecnológicos sencillos. 5. Resolver mediante puertas lógicas problemas tecnológicos sencillos. 6. Analizar sistemas automáticos, describir sus componentes. 7. Montar circuitos sencillos.</p>	<p>1.1. Describe el funcionamiento de un circuito electrónico formado por componentes elementales. 1.2. Explica las características y funciones de componentes básicos: resistor, condensador, diodo y transistor. 2.1. Emplea simuladores para el diseño y análisis de circuitos analógicos básicos, empleando simbología adecuada. 3.1. Realiza el montaje de circuitos electrónicos básicos diseñados previamente. 4.1. Realiza operaciones lógicas empleando el álgebra de Boole.</p>

<p>Aplicación del álgebra de Boole a problemas tecnológicos básicos. Puertas lógicas. Programas de diseño y simulación para el análisis y la realización de circuitos electrónicos básicos</p>		<p>4.2. Relaciona planteamientos lógicos con procesos técnicos. 5.1. Resuelve mediante puertas lógicas problemas tecnológicos sencillos. 6.1. Analiza sistemas automáticos, describiendo sus componentes. 7.1. Monta circuitos sencillos.</p>
BLOQUE 4 «Control y Robótica»		
<p>Sistemas automáticos, sistemas de lazo abierto y de lazo cerrado, componentes característicos de dispositivos de control. Diseño y construcción de robots. Arquitectura de un robot. Elementos mecánicos, articulaciones, sensores, unidad de control y actuadores. Grados de libertad. Tipos de robots. Características técnicas y aplicaciones. El ordenador como elemento de programación y control. Lenguajes básicos de programación. Aplicación de tarjetas controladoras en la experimentación con prototipos diseñados.</p>	<p>1. Analizar sistemas automáticos, describir sus componentes e identificar los elementos que componen un robot. 2. Montar automatismos sencillos y diseñar y construir un robot sencillo. 3. Desarrollar un programa para controlar un sistema automático o un robot y su funcionamiento de forma autónoma.</p>	<p>1.1. Analiza el funcionamiento de automatismos en diferentes dispositivos técnicos habituales, diferenciando entre lazo abierto y cerrado. 2.1. Representa y monta automatismos sencillos. 3.1. Desarrolla un programa para controlar un sistema automático o un robot que funcione de forma autónoma en función de la realimentación que recibe del entorno.</p>
BLOQUE 5 «Neumática e Hidráulica»		
<p>Los fluidos: fundamentos físicos. El aire comprimido y los fluidos hidráulicos. Análisis de sistemas hidráulicos y neumáticos. Componentes. Simbología. Principios físicos de funcionamiento. Programas de diseño y simulación para el análisis y la realización de circuitos básicos. Aplicación en sistemas industriales.</p>	<p>1. Conocer las principales aplicaciones de las tecnologías hidráulica y neumática. 2. Identificar y describir las características y funcionamiento de este tipo de sistemas. 3. Conocer y manejar con soltura la simbología necesaria para representar circuitos. 4. Experimentar con dispositivos neumáticos y simuladores informáticos.</p>	<p>1.1. Describe las principales aplicaciones de las tecnologías hidráulica y neumática. 2.1. Identifica y describe las características y funcionamiento de este tipo de sistemas. 3.1. Emplea la simbología y nomenclatura para representar circuitos cuya finalidad es la de resolver un problema tecnológico. 4.1. Realiza montajes de circuitos sencillos neumáticos e hidráulicos bien con componentes reales o mediante simulación.</p>
BLOQUE 6 «Tecnología y sociedad»		
<p>El desarrollo tecnológico a lo largo de la historia. Análisis de la evolución de objetos técnicos y tecnológicos importancia de la normalización en los productos industriales. Aprovechamiento de materias primas y recursos naturales.</p>	<p>1. Conocer la evolución tecnológica a lo largo de la historia. 2. Analizar objetos técnicos y tecnológicos mediante el análisis de objetos. 3. Describir los procesos de fabricación, distribución y comercialización de productos tecnológicos, y valorar la repercusión del desarrollo tecnológico en el día a día.</p>	<p>1.1. Identifica los cambios tecnológicos más importantes que se han producido a lo largo de la historia de la humanidad. 2.1. Analiza objetos técnicos y su relación con el entorno, interpretando su función histórica y la evolución tecnológica. 3.1. Elabora juicios de valor frente al desarrollo tecnológico a partir del análisis de objetos, relacionado</p>

Adquisición de hábitos que potencien el desarrollo sostenible. Cambios sociales y laborales asociados al desarrollo tecnológico. Prevención de riesgos laborales. Fabricación, distribución y comercialización de productos tecnológicos.		inventos y descubrimientos con el contexto en el que se desarrollan. 3.2. Interpreta las modificaciones tecnológicas, económicas y sociales en cada periodo histórico ayudándose de documentación escrita y digital.
--	--	---

3.5- Justificación pedagógica: Arduino en la Enseñanza Secundaria Obligatoria

A mi juicio, existen varios factores de importancia a la hora de elegir Arduino como tecnología para la realización de actividades, en especial dentro del bloque de robótica en 4º de la ESO:

- Un factor muy importante es sin duda el precio. Su **bajo coste** hace que las placas Arduino sean muy interesantes, sobre todo para profesores y estudiantes. Por una relativa módica cantidad (menos de 100 euros) podemos comprar el *Arduino Starter Kit Oficial*, un kit muy completo que viene con todos los componentes necesarios para iniciarse en este mundo. También se pueden encontrar kits con un buen número de componentes por menos de 50 euros. Todo esto hace de Arduino una opción muy interesante frente a otras opciones más caras.
- Otro factor importante es su **flexibilidad**, lo que hace que se pueda trabajar con Arduino en casi todas las plataformas, desde Mac OS X a Linux, pasando por Windows [3]. Además, como es de código abierto, han aparecido infinidad de herramientas que hacen fácil su uso, como es el caso de mBlock, herramienta que es la que he elegido para el desarrollo de las actividades con Arduino.
- Otra consecuencia que tiene el hecho de que Arduino sea una **plataforma abierta**, y que se base en código abierto (Open Source) en cuanto al software y el hardware, es que conseguimos [3]:
 - Ampliar la capacidad de desarrollar y construir multitud de proyectos con Arduino.
 - Poder combinar Arduino con otras plataformas o destinarlo a multitud de funciones, como pueden ser la construcción de drones, de impresoras 3D, etc, con lo que los límites de Arduino sólo están en la imaginación de inventores y desarrolladores.
- También podemos utilizar Arduino como una **herramienta para la iniciación a la programación**. Quizá con Arduino sea más fácil, debido a la gran cantidad de tutoriales que hay en la red para realizar pequeños proyectos electrónicos, y a sistemas de programación visuales como son Scratch For Arduino, Ardublock y mBlock, que pueden servir como iniciación [14].

- Además, si somos profesores, Arduino nos será útil para enseñar a los alumnos a conocer los principales componentes y fundamentos electrónicos [14].

Otros factores que considero importantes son los descritos en la página web *The Green Monkey; cinco razones para estudiar robótica* [13] y son los siguientes:

1. Los niños y jóvenes lo encuentran divertido, ya que cada vez es más evidente que los jóvenes responden bien a temas relacionados con la programación de robots, que tienen cada vez más información sobre ello en internet, que es el medio natural de los niños y jóvenes de ahora. Además de que existen varias competiciones y juegos para distintos grupos de edad que pueden canalizar los instintos competitivos de una manera positiva, como por ejemplo la *Liga Nacional de Robótica de Competición* [28].
2. Es una manera muy efectiva de introducir la programación a los estudiantes, ya que aunque la programación pueda ser abstracta, el hecho de realizar programas que luego se van a ver reflejados en la realidad, por ejemplo en la construcción de pequeños robots, puede ser muy motivador.
3. Por otra parte, proporciona habilidades útiles en el empleo futuro ya que habrá una necesidad de que la gente participe en la programación de futuros dispositivos mecánicos y electrónicos.
4. Desmitifica una tecnología compleja, es decir, la tecnología ya no es sólo un conocimiento para ciertas personas o sectores, ahora todo lo que nos rodea es tecnología y es importante conocerla de forma sencilla y lúdica.
5. La robótica también puede ser adecuada para personas con problemas de atención e hiperactividad, ya que se está comprobando que esas personas responden muy bien a las interacciones tranquilas, claras y consistentes que los robots pueden proporcionar.

Por otra parte, el aprendizaje con Arduino es un recurso más, que nos facilita el desarrollo de la robótica y competencias generales [15].

Capítulo 4 - Actividades

Como hemos visto en el capítulo 2 de este trabajo, uno de los factores principales del éxito de Arduino es la existencia de una amplia comunidad que apoya este proyecto, y que día tras día difunde nuevos contenidos en internet. Es por esto que en internet tenemos a nuestra disposición todo tipo de cursos, tutoriales, herramientas de consulta, proyectos, actividades, etc, que nos van a ayudar a usar Arduino de una manera más fácil.

Además, con mBlock y Arduino podemos implementar casi cualquier robot o invento tecnológico que imaginemos, debido, entre otras causas, a la gran variedad de placas, sensores, componentes y Shields que existen en el mercado para Arduino y al gran número de extensiones para poder manejar esos componentes con mBlock.

Teniendo en cuenta los aspectos anteriores, he diseñado en este trabajo una secuencia de actividades con varios objetivos:

1. Introducir e implementar la robótica en 4º ESO, en estudiantes que en general no han tenido antes contacto con la robótica ni la programación.
2. Cumplir con los Criterios de Evaluación y Estándares de aprendizaje comprendidos tanto en la orden EDU-362 [10], como en el Real Decreto 1105 [12], para el bloque de robótica dentro de la asignatura de Tecnología de 4º ESO.
3. Incentivar el autoaprendizaje y el trabajo en equipo en los estudiantes.

Estas actividades son una recopilación de entre las muchas que podemos encontrar en la comunidad Arduino y en internet en general, pero adaptadas y realizadas por mí. Los criterios que me han llevado a elegir la secuencia de actividades que propongo en este trabajo son:

- Se organizan de menor a mayor grado de dificultad, en las cuales los estudiantes van aprendiendo conceptos sobre robots, sensores, programación, etc, e interactuando con componentes que más adelante utilizarán.
- El número de sesiones disponible para poder realizarlas. La asignatura de Tecnología para 4º ESO contiene un total de seis bloques repartidos en tres trimestres [10] [12], de los cuales uno es el de robótica, por lo que he previsto una duración aproximada para este bloque de medio trimestre, es decir, de seis semanas. En 4º ESO, la Tecnología se imparte durante 4 horas semanales, por lo que tendremos unas 24 sesiones. Yo he ajustado las actividades a esas 20 sesiones, aunque este ajuste podrá variarse en caso de necesidad.

He decidido dividir en tres categorías las actividades que propongo en este trabajo. Éstas siguen un orden lógico de menor a mayor dificultad, aunque algunas de ellas, también podrían realizarse de una manera independiente. Las distintas categorías en las que he dividido las actividades son:

- **Actividades de iniciación:** En ellas comenzaremos desde cero, explicando la placa Arduino Uno, software mBlock y cómo conectarlos entre sí para poder trabajar.
- **Actividades de trabajo con sensores:** Son actividades para ir conociendo los tipos de sensores que existen y sus características principales. Con ellas, aprenderemos a manejar entradas y salidas tanto analógicas como digitales y PWM, utilizar pulsadores, fotoresistencias LDR, servomotores, motores de corriente continua, pulsadores, sensores ultrasónicos y de infrarrojos (IR), etc.
- **Actividades de construcción de robots:** Otro escalón más; mediante estas actividades aprenderemos a diseñar, construir y programar dos robots. El primero evitará caerse de una plataforma y el segundo detectará obstáculos y los sorteará.

He puesto un especial cuidado en la elección de esta serie de actividades, para que los estudiantes vayan progresando poco a poco tanto en el conocimiento de la robótica en general y de la placa Arduino Uno R3 en particular (su estructura, sus pines analógicos y digitales, etc), como en los conocimientos de programación con mBlock, desde programas muy sencillos hasta llegar a implementar programas de mayor dificultad, en los cuales deban utilizar distintos objetos y un buen número de variables, bucles y sentencias de programación.

4.1- Actividades de iniciación

4.1.1- Actividad 0: Conociendo nuestra placa Arduino Uno R3

Objetivos

Familiarizarnos con las partes de una placa Arduino UNO, con la conexión de la placa al PC y con la configuración del programa mBlock para poder trabajar con ella.

Competencias

- Comunicación lingüística (CCL).
- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).

Metodología

- Clase magistral.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Una pizarra digital.
- Un cable USB para conectar la placa al PC.

Duración de la actividad

Una sesión de 50-55 minutos.

Descripción de la Actividad

Esta actividad se desarrollará en dos partes:

1. En la primera parte, presentaremos la placa Arduino UNO, haciendo hincapié en las partes más importantes de la misma. Expondremos la imagen de la placa Arduino UNO (figura 1) en la pizarra digital e iremos describiendo parte a parte la misma. También explicaremos a los alumnos la diferencia entre las entradas/salidas digitales y las analógicas, así como las diferencias entre una entrada/salida digital normal y una PWM, para que se utilicen unas y otras, etc.

Presentaremos el programa mBlock, describiremos las partes más importantes de éste, los bloques que podemos utilizar para programar videojuegos y, los que podemos utilizar para programar Arduino. Haremos algunos sencillos ejemplos de programación mediante los cuales iremos explicando conceptos como variables, sentencias y estructuras.

También utilizaremos, si lo consideramos conveniente, el apoyo de vídeos y tutoriales para introducir la programación Arduino con mBlock, [22] y [23].

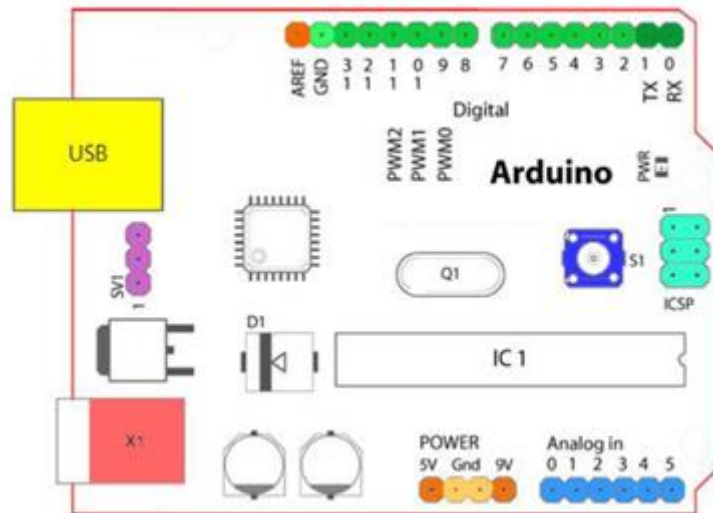


Figura 1: Esquema de pines Arduino Uno. Fuente: <http://ramon-gzz.blogspot.com>

2. En la segunda parte, y antes de empezar a programar nuestro Arduino con mBlock, deberemos que asegurarnos de que nuestra placa está bien conectada al PC. Para ello, seguiremos los siguientes pasos:
 - Conectaremos nuestra placa a un puerto USB de nuestro PC.
 - Accederemos al *Administrador de dispositivos* de nuestro PC y veremos en qué puerto serie está conectado el Arduino. En nuestro caso vemos que es el puerto de serie número 8 (COM8).
 - En el menú *Placas* de mBlock elegimos *Arduino*.
 - En el menú *Extensiones*, nos aseguramos que esté marcada la opción *Arduino*.
 - En el menú *Conectar*, elegimos el puerto serie en el que está conectado nuestro Arduino, el COM8 en nuestro caso.

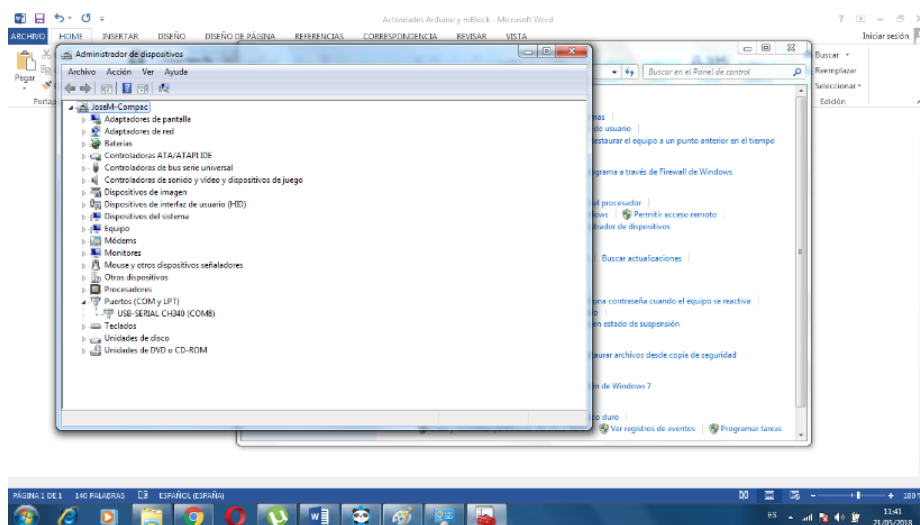


Figura 2: Ventana administrador de dispositivos. Fuente: Propia

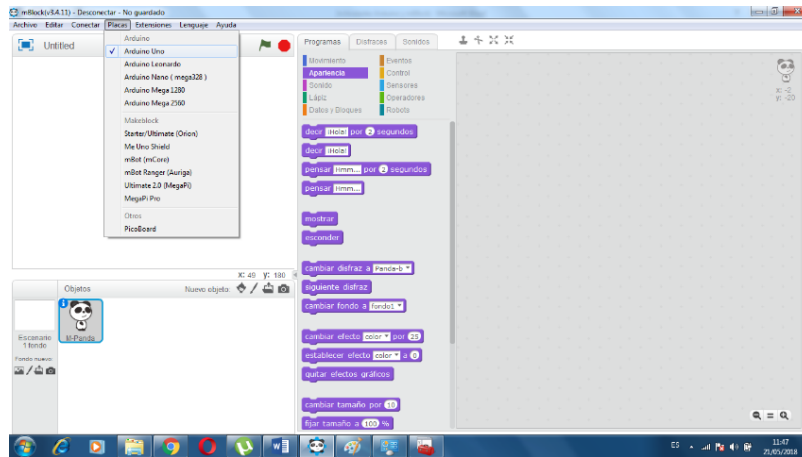


Figura 3: Menú *Placas* mBlock. Fuente: Propia

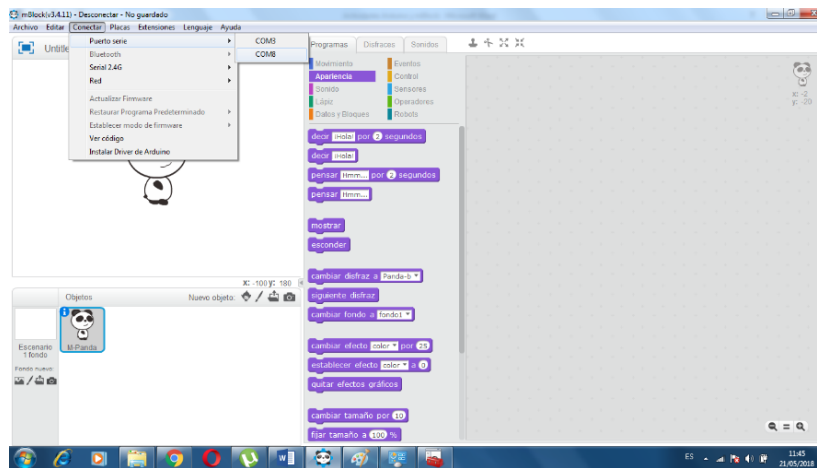


Figura 4: Menú *Conectar* mBlock. Fuente: Propia

Si queremos visualizar el código “real” del Arduino a la vez que programamos, tendremos que tener seleccionada la opción *Arduino Mode* en el menú *Editar*.

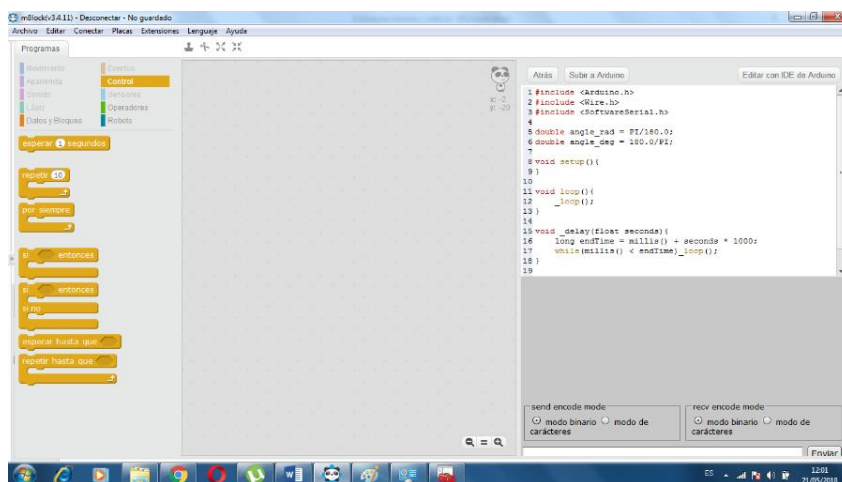


Figura 5: Menú *Editar* mBlock. Fuente: Propia

Una vez realizados todos estos pasos y si no se nos ha presentado ningún problema, tendremos correctamente configurada nuestra placa Arduino Uno para empezar a programar con mBlock.

Evaluación

La segunda parte de la actividad no será evaluable, ya que podrían surgir cualquier tipo de problemas de diversa índole en la conexión de la placa al PC, que entiendo están fuera del ámbito de conocimientos que pretendo adquiriera el alumno.

Los quince últimos minutos de la actividad los dedicaremos a realizar un cuestionario para evaluar los conocimientos iniciales del alumnado. **Anexo 2.**

4.1.2- Actividad 1: Encendemos un LED

Objetivos

En esta actividad aprenderemos cómo encender y apagar un diodo LED mediante la programación de bloques con mBlock. Aprenderemos a utilizar los pines digitales como salidas.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).

Metodología

- Trabajo individual.
- Aprendizaje basado en problemas.
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- Un diodo LED.

Duración Actividad

Una sesión de 50-55 minutos.

Descripción de la Actividad

En esta actividad aprenderemos a usar las salidas digitales de nuestro Arduino Uno encendiendo y apagando un diodo LED. Para ello usaremos el pin digital 13 de nuestra placa. Podríamos utilizar otro cualquiera, pero en este caso, al llevar el pin 13 asociado una resistencia, ya tendremos limitada la corriente que pasa por el diodo (para no quemarlo) sin tener que usar ninguna resistencia externa.

Deberemos tener cuidado al conectar los LEDs, puesto que estos tienen polaridad. Debemos saber que la patilla más larga es el ánodo (+), que conectaremos al pin 13, y la más corta el cátodo (-), que conectaremos a tierra (GND). La figura 1 nos muestra un LED real, con su ánodo y su cátodo.

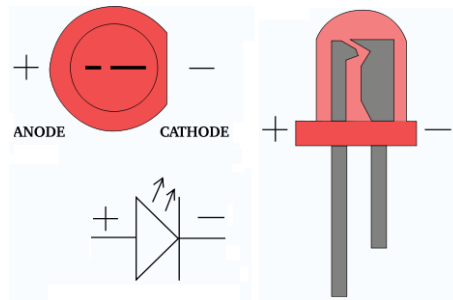


Figura 1: diodo LED. Fuente: www.makinandovelez.wordpress.com

La figura 2 nos muestra la conexión del diodo a la placa y la figura 3 la programación de bloques en mBlock que necesitamos para hacer que el LED se encienda y se apague.

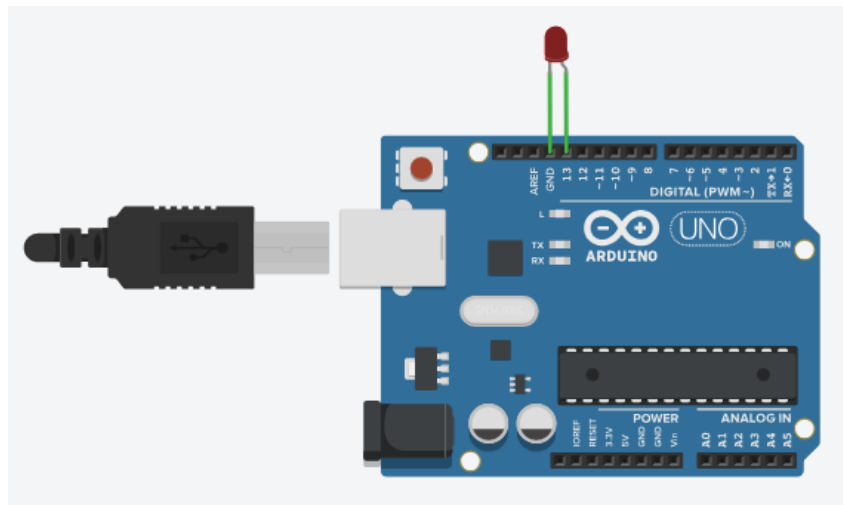


Figura 2: Conexión LED a Arduino: Fuente Propia



Figura 3: Programa LED. Fuente: Propia

En la figura 3 podemos ver que el tiempo que se mantiene, tanto apagado como encendido el LED es de 2 segundos. Cambiando el valor de este tiempo, podremos hacer que el LED parpadee con mayor o menor velocidad.

En las figuras 4 (LED OFF) y 5 (LED ON) podemos ver el resultado obtenido en el circuito real.

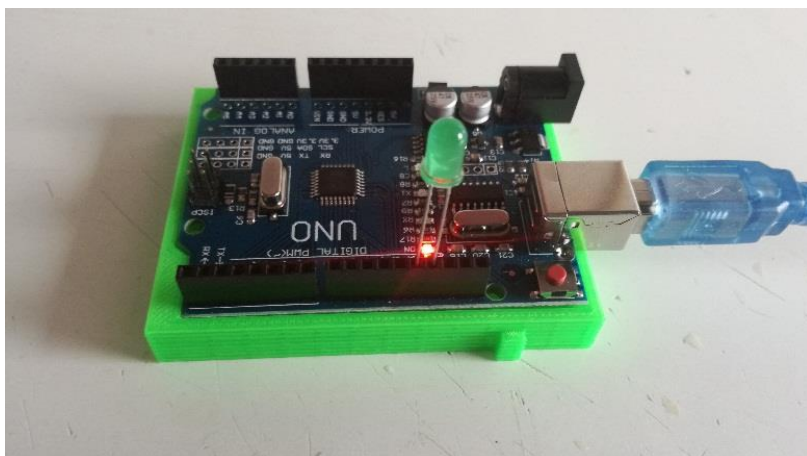


Figura 4: Montaje circuito LED. Fuente: Propia



Figura 5 Montaje circuito LED. Fuente: Propia

Evaluación

La actividad estará correctamente realizada si se enciende y apaga el LED cada 1 o 2 segundos.

4.2- Actividades de trabajo con sensores

4.2.1- Actividad 2: El semáforo

Objetivos

Aprender a construir un semáforo operativo y a utilizar varios pines como salidas. También, aprenderemos a implementar más de un objeto al programar con mBlock.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).
- Competencias sociales y cívicas (CSC).

Metodología

- Trabajo en grupos de 2 personas.
- Aprendizaje basado en problemas.
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 3 diodos LED, uno rojo, uno amarillo y otro verde.
- 3 resistencias de 220Ω
- Una protoboard.
- Cables para realizar las conexiones.

Duración Actividad

Dos sesiones de 50-55 minutos.

Descripción de la Actividad

En esta actividad vamos a utilizar tres salidas digitales de nuestra placa Arduino para construir un semáforo. Éste realizará un ciclo que comenzará con el LED rojo encendido durante cinco segundos, momento en el que se apagará y se encenderá el amarillo, que parpadeará durante dos segundos para después apagarse y dar paso al LED verde que permanecerá encendido otros cinco segundos, con lo que el ciclo volverá a empezar.

Las salidas digitales que utilizaremos serán las correspondientes a los pines 12 para el LED verde, 11 para el LED amarillo y 10 para el LED rojo, pudiéndose elegir otras salidas digitales cualesquiera que posea la placa. Las resistencias irán en serie con los LED, pudiéndose éstas intercalar en el circuito antes o después de los LED, la posición en este caso es indiferente, se utilizan para limitar la corriente por los LED y no quemarlos. El esquema de conexión es el mostrado en la figura 1.

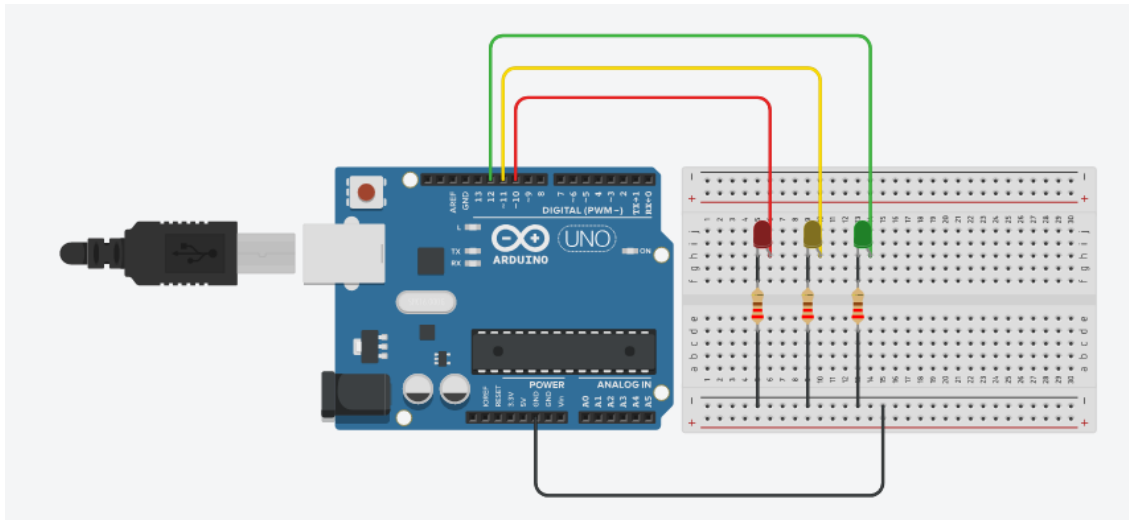


Figura 1: Conexión LEDs a Arduino: Fuente Propia

La programación de mBlock con su diagrama de bloques se presenta en la figura 2. Es importante observar que se pueden establecer variaciones en la secuencia del semáforo, como por ejemplo, hacer parpadear el LED amarillo cuando aún está encendido el rojo para luego apagar los dos al encender el LED verde etc. Esta variante se puede implementar una vez hayamos conseguido hacer funcionar el semáforo.



Figura 2: Programa semáforo. Fuente: Propia

Las figuras 4,5 y 6, nos muestran distintas fases del montaje real del semáforo:

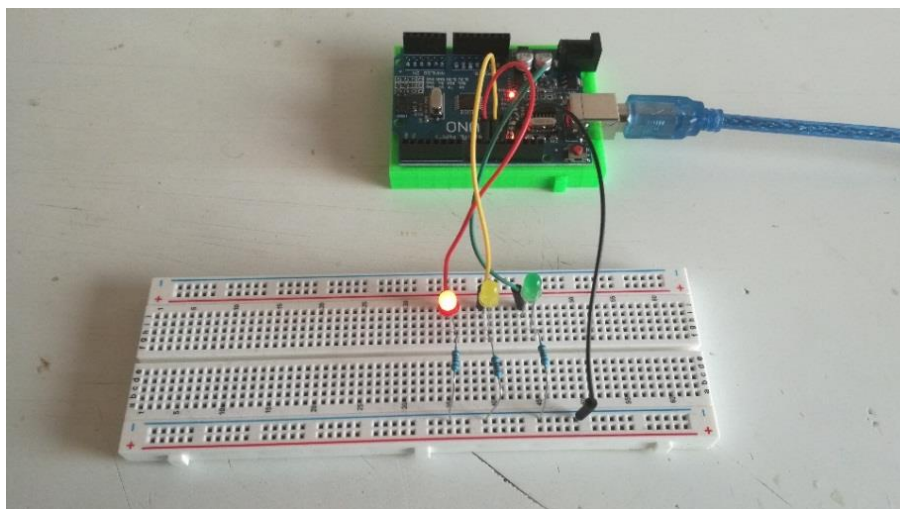


Figura 4: Montaje circuito. Fuente: Propia

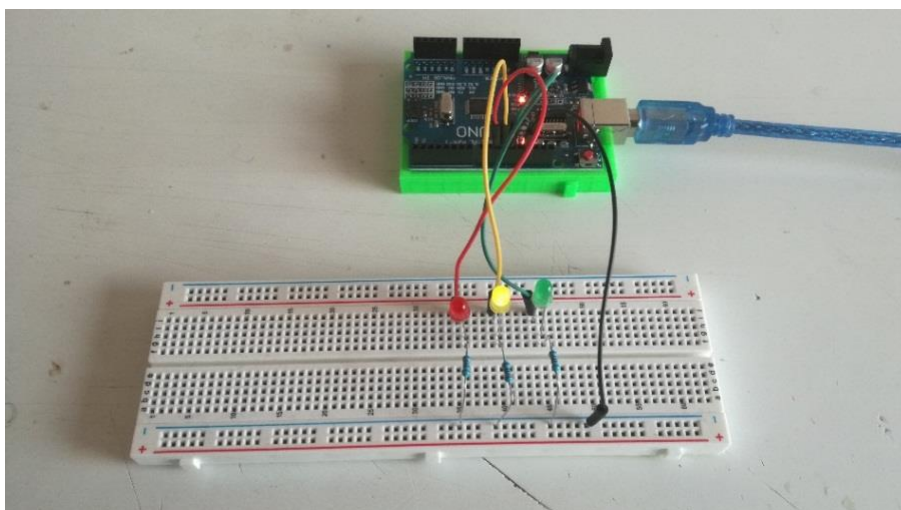


Figura 5: Montaje circuito. Fuente: Propia

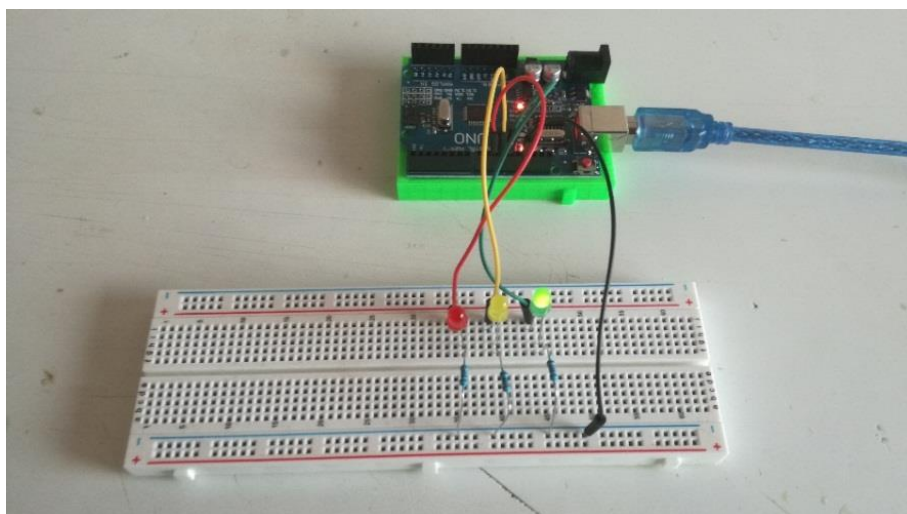


Figura 6: Montaje circuito. Fuente: Propia

Evaluación

Al final de la actividad, el semáforo deberá funcionar correctamente. Se valorará de forma positiva modificaciones en el diseño inicial, que simulen mejor el comportamiento de un semáforo real.

4.2.2 Actividad 3: Los dos pulsadores

Objetivos

Aprender a implementar pulsadores usando entradas digitales, utilizándolos en este caso para encender y apagar un LED.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).
- Competencias sociales y cívicas (CSC).

Metodología

- Trabajo en grupos de 2 personas.
- Clase magistral.
- Aprendizaje basado en problemas.
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 1 diodo LED.
- 1 resistencia de 220Ω .
- 2 resistencias de $4k7\ \Omega$.
- 2 pulsadores.
- Una protoboard.
- Cables para realizar las conexiones.

Duración Actividad

Dos sesiones de 50-55 minutos.

Descripción de la Actividad

Esta práctica está compuesta de dos partes:

1. En la primera parte, realizaremos una breve explicación de lo que es un interruptor y su funcionamiento. Comenzaremos explicando que el interruptor es un dispositivo eléctrico que, al pulsar un botón o girar una palanca, conecta dos contactos para permitir el paso de la electricidad a través de ellos. Los contactos que vamos a utilizar para esta práctica y su esquema eléctrico están representados en la figura 1.

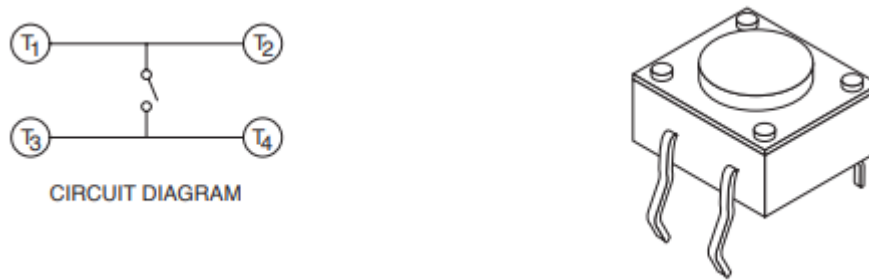


Figura 1: Interruptor- pulsador. Fuente: www.arduino.cc

En cuanto a la conexión de los pulsadores con la placa, aunque nuestra primera intención sea conectar los pines Arduino a una tensión de referencia de 5v o 0v, no podemos conectarlos directamente a la tensión de referencia, como se muestra en la figura 2, ya que esto podría funcionar con el interruptor cerrado, pero ¿qué pasaría con el interruptor abierto?. En este último caso, estamos dejando el pin del Arduino desconectado de cualquier tensión, por lo que el estado en el que se encontrará el pin será un **estado indeterminado** al que solemos llamar **estado de alta impedancia**.

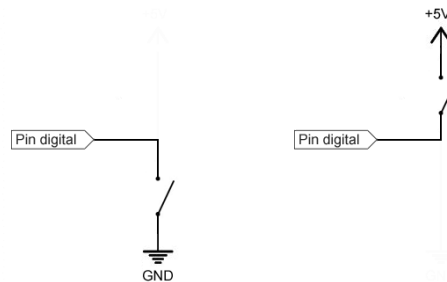


Figura 2: Esquema interruptor. Fuente: www.luisllamas.es

Para solucionar este estado de indeterminación, podríamos pensar en conectar el pin a dos referencias de tensión alternadas en función del estado del interruptor, como podemos ver en la figura 3.

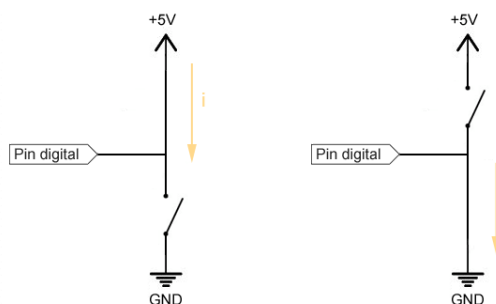


Figura 3: Esquema interruptor. Fuente: www.luisllamas.es

El problema de este montaje es que al pulsar el interruptor estamos conectado directamente los 5v a 0v, es decir, estamos **causando un cortocircuito**. Para evitar

este cortocircuito, vamos a necesitar dos resistencias, las cuales vamos a llamar Resistencia de **Pull-Up** y resistencia de **Pull-Down**, de tal forma que:

- La resistencia de Pull-Up fuerza el pin a HIGH cuando el pulsador está abierto, como podemos apreciar en la figura 4 y, cuando está cerrado el pulsador, fuerza el pin a LOW y la resistencia evita el cortocircuito además de limitar la corriente por el pin.
- La resistencia de Pull-Down fuerza el pin a LOW cuando el pulsador está abierto y a HIGH cuando está cerrado, caso en el que la intensidad que le llega al pin se ve limitada por la resistencia.

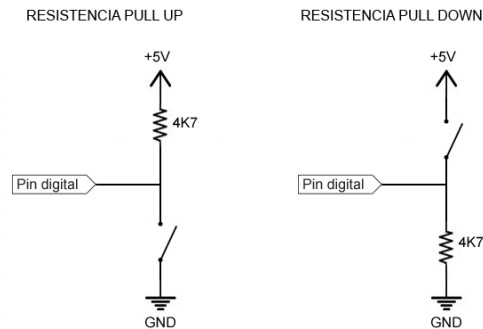


Figura 4 Esquema PULL-UP y PULL DOWN. Fuente: www.luisllamas.es

El valor de las resistencias viene condicionado por la intensidad que pasa al accionar nuestro interruptor y por un concepto llamado “autoridad del Pull/Down”, que está relacionado con el ruido en la medición [5] y [16]. Para la resistencia, se suele utilizar un valor medio de $4k7\Omega$, ya que si la resistencia es muy pequeña, el valor de la intensidad será alto, por lo que se producirán calentamientos y mayores consumos y, si es muy grande, el valor de la intensidad será muy bajo, lo que producirá problemas de ruidos.

Para nuestro montaje, podremos elegir el lugar dónde pongamos las resistencias, dependiendo si al pulsar el interruptor queremos tener un valor LOW en el pin (Pull-Up), o un valor HIGH (Pull-Down). Para una mayor comprensión, realizaremos los dos montajes.

2. En esta segunda parte es donde realizaremos el montaje del circuito eléctrico y la programación del Arduino. Vamos a realizar un mismo montaje utilizando resistencias Pull-Up y Pull-Down. Lo que pretendemos es que, activando el primer pulsador, se encienda el LED y se mantenga encendido hasta que se active el segundo pulsador, momento en el que se apagará.

Para esta práctica, utilizaré como salida el pin digital 10 y como entradas los pines digitales 6 y 7, aunque podríamos elegir otros pines digitales cualesquiera.

La figura 5 nos muestra un montaje en Pull-Up, junto al código mBlock que tendremos que utilizar para conseguir nuestro objetivo, mientras que la figura 6 nos muestra lo mismo, pero utilizando en este caso las resistencias en Pull-Down.

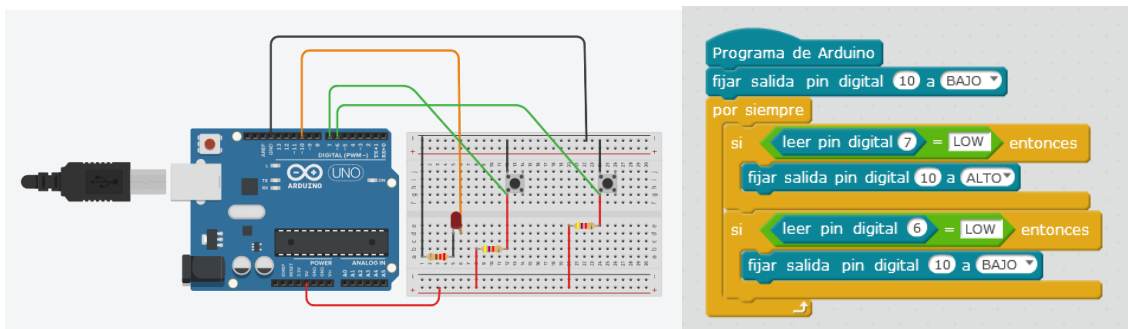


Figura 5: Esquema circuito Pull-Up y código. Fuente: Propia

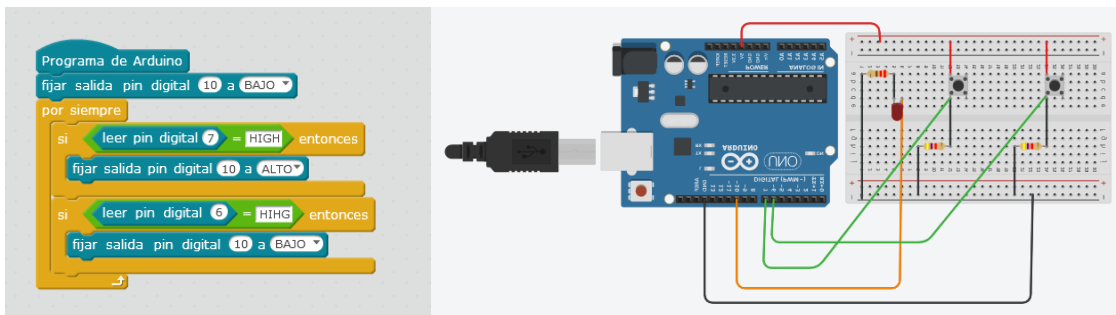


Figura 6: Esquema circuito Pull-Down y código. Fuente: Propia

En las figuras 7 y 8 podemos observar el montaje real del circuito para Pull-Up y en las figuras 9 y 10 para Pull-Down.

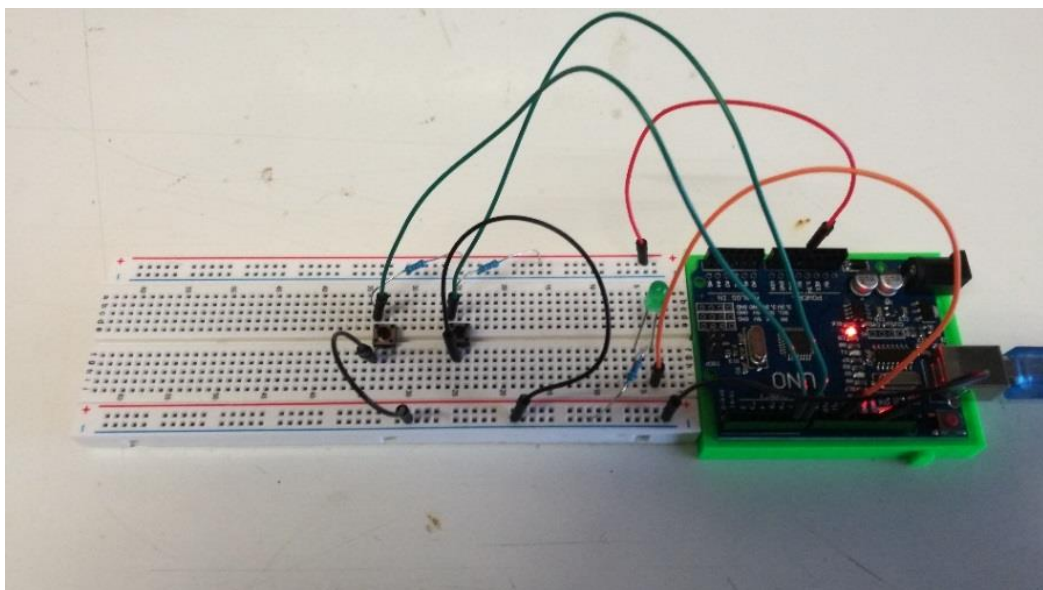


Figura 7: Montaje Pull-Up. Fuente: Propia

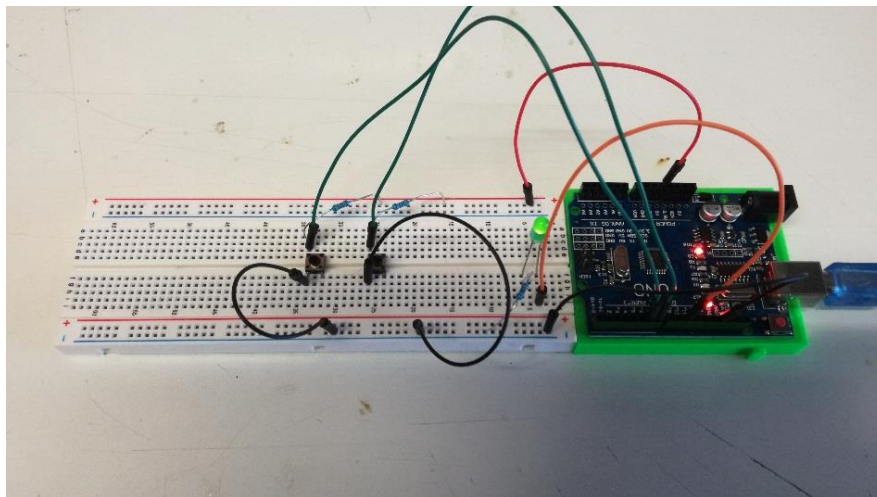


Figura 8: Montaje Pull-Up. Fuente: Propia

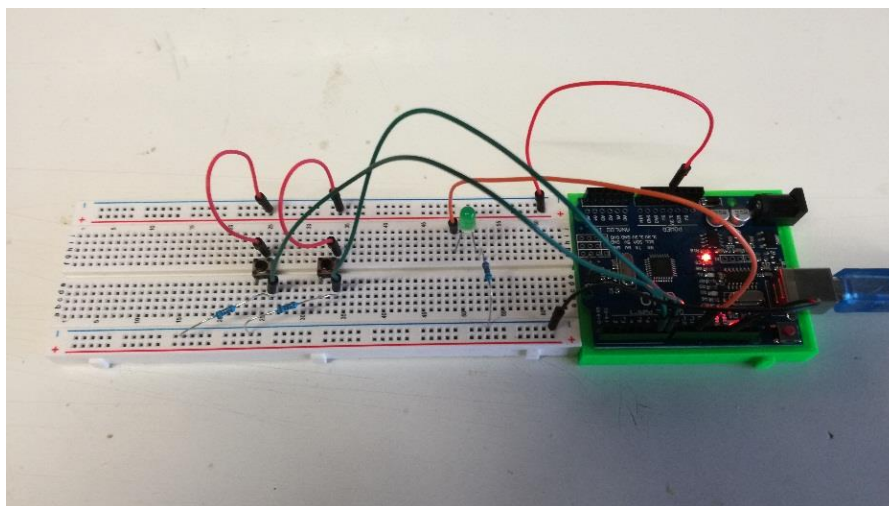


Figura 9: Montaje Pull-Down. Fuente: Propia

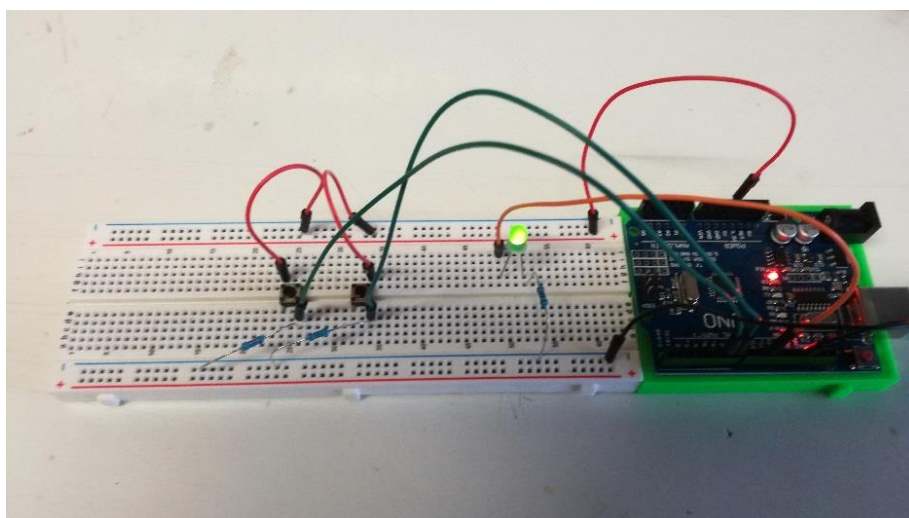


Figura 10: Montaje Pull-Down. Fuente: Propia

Evaluación

Al final de la actividad, utilizaremos la siguiente rúbrica para la evaluación de la misma:

RÚBRICA EVALUACIÓN ACTIVIDAD 3	No le ha quedado claro (1)	Le ha quedado medianamente claro (2)	Le ha quedado muy claro (3)
Comprende el circuito Pull-Up y su funcionamiento			
Comprende el circuito Pull-Down y su funcionamiento			
Sabe programar mBlock para que funcionen correctamente los pulsadores			
Realiza de forma correcta las conexiones de los pulsadores en la Protoboard			
Realiza de forma correcta las conexiones en el Arduino			

Fuente: Propia

4.2.3- Actividad 4: Usemos el potenciómetro

Objetivos

Con esta actividad aprenderemos a utilizar el potenciómetro y las entradas analógicas de nuestro Arduino para variar la luminosidad de un diodo LED.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).

Metodología

- Trabajo individual.
- Aprendizaje basado en problemas.
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 1 LED.
- 1 resistencia de 220 Ω .
- 1 Potenciómetro 0 – 10 K Ω .
- Cables para realizar las conexiones.

Duración Actividad

Una sesión de 50-55 minutos.

Descripción de la Actividad

En esta actividad aprenderemos a utilizar un potenciómetro. Éste irá conectado a un pin analógico de nuestro Arduino que utilizaremos como entrada. Iremos tomando lecturas de esta entrada, y en función de las mismas, iremos regulando la intensidad de luz de un diodo LED que tendremos conectado a un pin digital PWM.

El potenciómetro que vamos a utilizar es el de la figura 1, de 0 a 10k Ω , y que presenta tres terminales: uno de tierra, otro para la alimentación y un tercero que le conectaremos a un pin analógico de nuestra placa, el pin A0.



Figura 1

Los pines analógicos de nuestra placa tienen la capacidad de variar desde 0 a 1014, es decir, el valor 0 se correspondería con 0Ω de nuestro potenciómetro, y el valor 1023 con $10K\Omega$. Para ajustar la lectura de la señal analógica, que variará entre 0 y 1024 (resolución de 10 bits) a una PWM, que varía entre 0 y 255, lo único que tendremos que hacer será dividir por 4. El funcionamiento es sencillo: dependiendo de la posición del potenciómetro, la entrada analógica tendrá un mayor o menor valor, lo que producirá un mayor o menor valor en la salida digital, que producirá a su vez, una mayor o menor intensidad en la luz emitida por el LED.

En las siguientes figuras mostramos las conexiones del circuito (figura 2), el código necesario para realizar esta actividad (figura 3) y la foto del circuito real (figura 4).

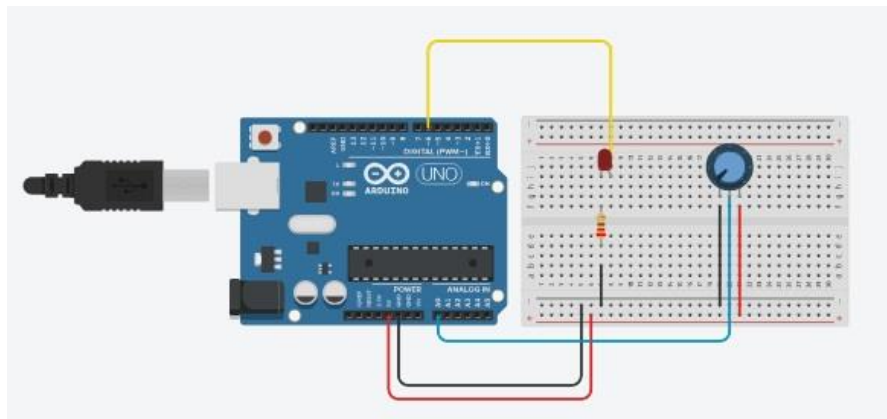


Figura 2: Esquema de montaje. Fuente: Propia



Figura 3: Código mBlock. Fuente: Propia

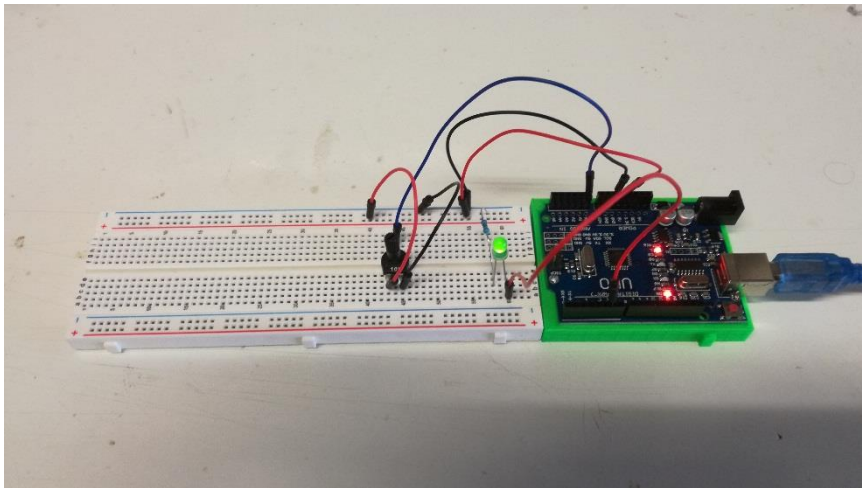


Figura 4: Montaje real. Fuente: Propia

Evaluación

La actividad será correcta si a medida que variamos el potenciómetro, la luminosidad del diodo LED cambia.

4.2.4 Actividad 5: Controlando el giro de un servomotor

Objetivos

Con esta actividad aprenderemos a controlar el giro de un servomotor desde 0° a 180° utilizando Arduino y mBlock.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).

Metodología

- Trabajo individual.
- Aprendizaje basado en problemas.
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 1 Servomotor.
- 1 batería de 9v.
- Cables para realizar las conexiones.

Duración Actividad

Una sesión de 50-55 minutos.

Descripción de la Actividad

En esta actividad aprenderemos a mover un servo con mBlock en un recorrido normal del mismo, es decir, de 0° a 180°. Normalmente la mayoría de los servomotores llevan una pestaña interna que hace de tope y limita el recorrido de 0° a 180°. Muchas veces en robótica se suelen modificar los servos para que puedan llevar a cabo una rotación continua, eliminando esa pestaña, pero en nuestro caso nos limitaremos al movimiento de 0° a 180°.

¿Qué es un servomotor?

Un servo motor [25] es un motor eléctrico que presenta dos características especiales:

- Nos permite mantener una posición que indiquemos, siempre que esté dentro del rango de características del dispositivo.
- Nos permite controlar la velocidad de giro; podemos hacer que espere un tiempo antes de comenzar el movimiento.

Existen varios modelos de servomotor con Arduino. En este caso, nosotros vamos a utilizar un Micro Servo de 9g SG90 de Tower Pro, el cual tenemos representado en la figura 1.

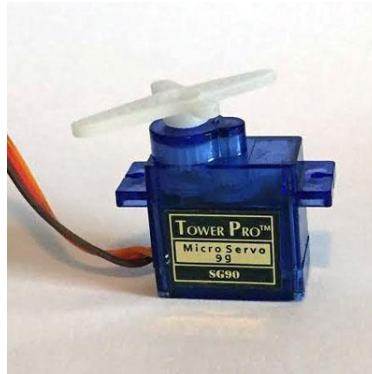


Figura 1; Servomotor SG 990. Fuente: www.digitalab.org

Dos cosas importantes a tener en cuenta con este dispositivo:

- El ángulo de giro; en este caso nos permite hacer un barrido entre -90° y 90° . lo que viene a ser un ángulo de giro de 180° .
- Aunque el servo puede moverse con una resolución de más de 1 grado, éste es el máximo de resolución que vamos a poder conseguir debido a la limitación de la señal PWM que genera nuestra placa Arduino Uno.

Este dispositivo cuenta además con tres cables para su conexión a la placa: Alimentación, Tierra o GND y Señal (que conectaremos a cualquier pin PWM). En la figura 2 podemos ver el esquema de conexión del servo con la placa Arduino.

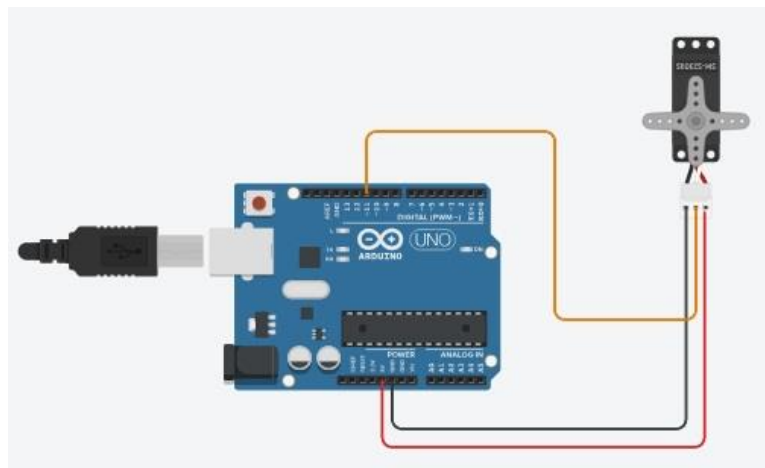


Figura 2: Conexión servo a la placa. Fuente: Propia

Para la parte de programación, necesitaremos una variable en la cual ir almacenando los valores del ángulo que queremos vaya adquiriendo el servo. A esta variable la vamos a llamar *Ángulo*. El código mBlock para esta actividad lo podemos ver en la figura 3



Figura 3: programa mBlock. Fuente: Propia

El funcionamiento del programa es el siguiente: Fijamos la variable ángulo en cero grados, llevamos al servo hasta esa posición y, mientras el ángulo sea menor de 180 grados, iremos moviendo el servo de 10 en 10 grados. Cuando el ángulo sea mayor de 180 grados, se fijará otra vez a cero y el ciclo comenzará de nuevo. La figura 4 nos muestra el circuito y el servo en funcionamiento.

Una precaución que debemos tener en cuenta es la de dar tiempo al servo para que alcance la posición deseada. Para un recorrido total bastará con 2 segundos y para los recorridos parciales, puesto que el ángulo va variando de 10 en 10 grados, nos bastará con una décima de segundo.

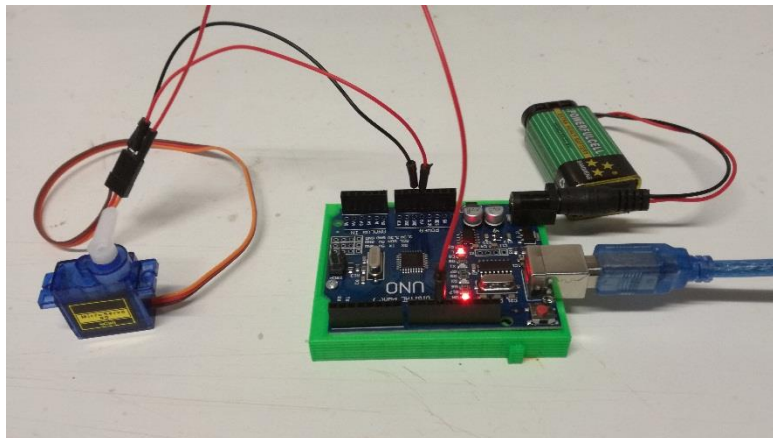


Figura 4: Circuito real. Fuente: Propia

Evaluación

La actividad se considera correcta si el servomotor consigue realizar giros de 0° a 180°.

4.2.5 Actividad 6: Control de un motor DC

Objetivos

Aprender a controlar un motor DC, su sentido de giro y su velocidad mediante el Arduino programando con mBlock. Aprender a utilizar las salidas PWM de la placa.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).
- Competencias sociales y cívicas (CSC).

Metodología

- Trabajo en grupos de 2 personas.
- Clase magistral.
- Aprendizaje basado en problemas.
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 1 Motor DC.
- 1 batería de 9v.
- 1 controlador para motores DC y paso a paso (PAP) L298N.
- Cables para realizar las conexiones.

Duración Actividad

Dos sesiones de 50-55 minutos.

Descripción de la Actividad

En esta actividad aprenderemos a controlar un motor de corriente continua con Arduino. Para controlar este motor, vamos a hacer uso del controlador de motores L298N, ya que si conectáramos directamente nuestro motor a la placa Arduino podríamos quemarla. El controlador L298N es representado en la figura 1 y se caracteriza principalmente porque con él podemos controlar el sentido de giro y la velocidad de hasta dos motores de corriente continua.

Esta actividad la dividiremos en dos partes:

1. En la primera explicaremos qué es un controlador L298N, sus características más importantes y cómo utilizarlo.
2. Realizaremos el montaje electrónico y la programación de la placa con mBlock.

En la figura 2 podemos ver las conexiones entre la placa Arduino y el controlador, además de las salidas para los motores 1 y 2. El funcionamiento es sencillo: El controlador se dirige a través de seis pines, tres para la salida 1 (ENA, IN1, IN2) y tres para la salida 2 (ENB, IN3, IN4). Los pines ENA y ENB vienen de serie conectados a un puente, con lo que si dejamos el puente, lo que podremos controlar son los sentidos de giro del motor 1, mediante la salida 1 (Activando o desactivando IN1 e IN2 mediante dos salidas digitales desde nuestra placa), y del motor 2, mediante la salida 2 (Activando o desactivando IN3 e IN4 mediante dos salidas digitales desde nuestra placa), de tal manera que IN1=HIGH e IN2=LOW hará que el motor 1 gire en un sentido, e IN1=LOW e IN2=HIGH hará que el motor 1 gire en sentido contrario (de igual modo con IN3 e IN4 y el motor 2).

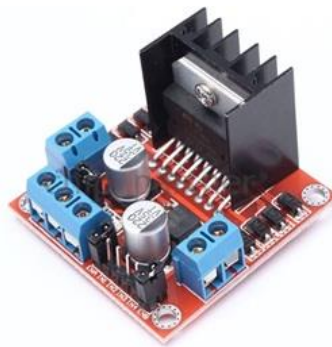


Figura 1: Controlador motores L298N. Fuente: www.gearbest.com

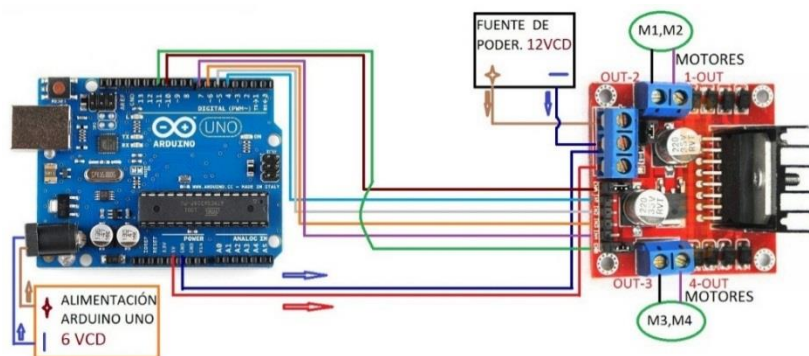


Figura 2: Conexión Arduino – L298N. Fuente: www.ejemplostutorialesorotronik.blogspot.com

Si quitamos los puentes y utilizamos los pines ENA y ENB del controlador, podremos variar la velocidad de giro de los motores 1 y 2. Para lograr esto, deberemos conectar cada pin a una salida digital PWM de nuestro arduino, como podemos observar en la figura 2. Sabemos que una salida PWM es un tipo de salida digital que emula una salida analógica estableciendo 256 valores diferentes (de 0 a 255) [1] y [16].

Para salvar la limitación que tienen la mayoría de los automatismos, y que es que no pueden proporcionar una salida analógica, ni siquiera una salida analógica discretizada (dividida en un rango de valores), los automatismos emplean una especie de “truco”: activar una salida digital durante un tiempo y mantenerla apagada durante el resto. El promedio de la señal durante el tiempo será igual al valor analógico deseado. Un ejemplo de este sistema lo tenemos en la figura 3.

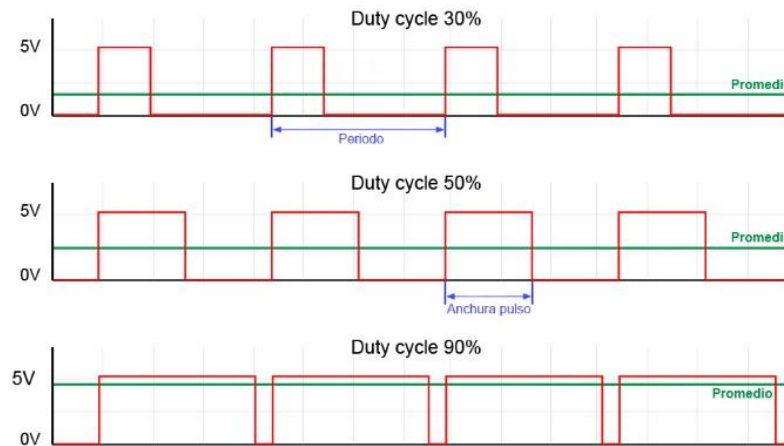


Figura 3: Ciclos PWM. Fuente www.luisllamas.es

Nosotros para esta actividad vamos a utilizar los pines 10 (PWM), 5 y 4. Los tres actuarán como salidas. Mediante los pines 4 y 5 controlaremos el sentido de giro del motor y con el pin 10 su velocidad. En la figura 4 podemos observar el montaje del circuito para esta práctica.

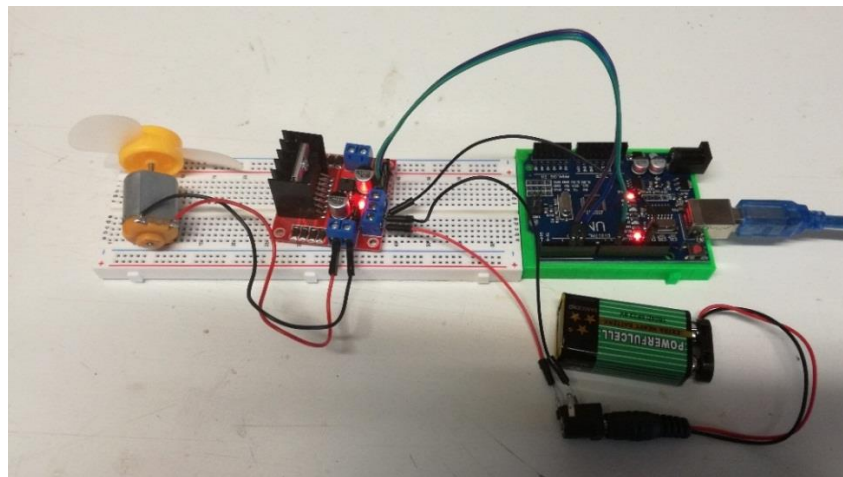


Figura 4: Montaje circuito real. Fuente: Propia

Por otro lado, el motor que vamos a utilizar en esta actividad es un motor básico de corriente continua de 6 voltios. Es el tipo de motor que se suele suministrar en cualquier kit de Arduino. Lo tenemos representado en la figura 5.

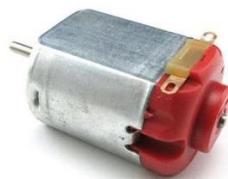


Figura 5: Motor 3v-6v DC. Fuente: www.joom.com

Una vez terminado el montaje del circuito, vamos a proceder con la programación del mismo con mBlock. Tenemos dos formas de programarlo con mBlock:

1. La primera es utilizar tres variables para cada salida del controlador. En este caso he definido los pines 10 (PWM) para la variable ENA, y 11 para ENB, con las que controlaremos le velocidad de giro de los motores, y los pines 4, 5, 6 y 7 para las variables IN1, IN2, IN3, e IN4 respectivamente, mediante las cuales controlaremos el sentido de giro de los motores. El programa es el de la figura 5; si queremos cambiar la velocidad o el sentido de giro de algún motor sólo tendríamos que cambiar el valor de las variables correspondientes, compilar el programa y subirlo a la placa Arduino. Las salidas PWM nos permiten un rango de 0 (valor más bajo, motor parado en este caso) hasta 255 (valor más alto, motor al máximo), en la figura 6 podemos observar que elegimos un valor medio, 125.

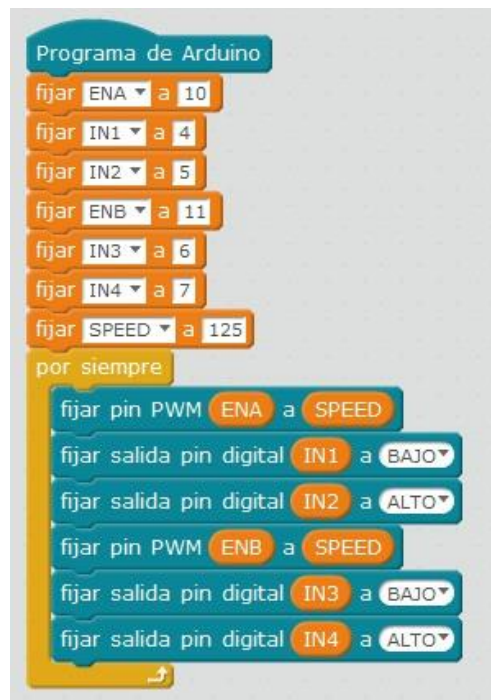


Figura 6: Programa mBlock. Fuente: Propia

2. La segunda es utilizar una librería o una extensión para el controlador, ya que mBlock permite añadir múltiples extensiones y utilizar las funciones predefinidas en la misma. Hemos optado por utilizar una extensión *mBlock_motor_L298N*, desarrollada en este caso por Fernando de Sousa [24]. Esta librería nos permite controlar todos los parámetros del controlador con tres sencillas instrucciones, como podemos observar en la figura 7. La primera instrucción nos sirve para establecer los pines que queremos utilizar y las dos siguientes para establecer el sentido de giro y la velocidad de los motores.

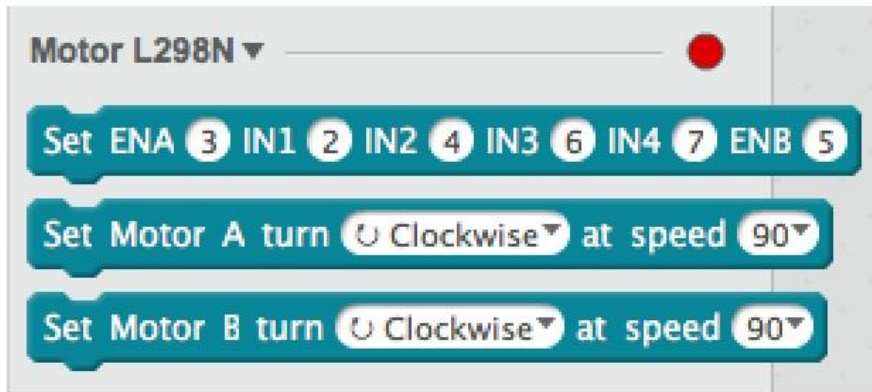


Figura 7: Sentencias mBlock para L298N. Fuente: Fernando de Sousa [24]

El código mBlock utilizando esta librería sería el de la figura 8. En este caso, hemos elegido un valor para la velocidad de 150, ya que la instrucción sólo nos permite elegir algunos valores discretos para la velocidad (0, 50, 70, 100, 150, 200 y 255).



Figura 8: Programa mBlock. Fuente: Propia

Por último, sólo nos queda comprobar el correcto funcionamiento del circuito para las dos formas de implementarlo y esto lo podemos observar mediante la figura 9.

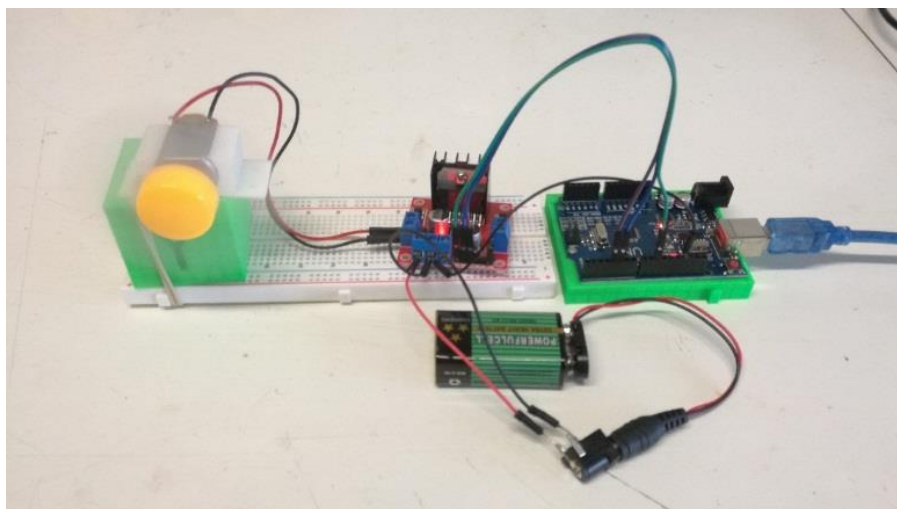


Figura 9: Montaje circuito real. Fuente: Propia

Evaluación

Para la evaluación de esta actividad, utilizaremos la siguiente rúbrica:

RÚBRICA EVALUACIÓN ACTIVIDAD 6	No le ha quedado claro (1)	Le ha quedado medianamente claro (2)	Le ha quedado muy claro (3)
Distingue las entradas digitales PWM y comprende su funcionamiento			
Comprende el funcionamiento del controlador L298N			
Comprende la programación del controlador L298N			
Realiza de forma correcta las conexiones en el controlador L298N			
Sabe resolver el objetivo de la práctica y lo programa con mBlock			
Realiza de forma correcta las conexiones en el Arduino			

Fuente: Propia

4.2.6 Actividad 7: ¡Alarma!, ¡Intrusos!

Objetivos

Con esta actividad aprenderemos a manejar un sensor de movimiento infrarrojo pasivo (sensor PIR) y un zumbador activo.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).
- Competencias sociales y cívicas (CSC).

Metodología

- Trabajo en grupos de 2 personas.
- Aprendizaje basado en problemas.
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 1 sensor PIR.
- 1 zumbador.
- 1 batería de 9v.
- Cables para realizar las conexiones.

Duración Actividad

Dos sesiones de 50-55 minutos.

Descripción de la Actividad

Esta actividad la vamos a realizar en dos partes:

1. En esta primera parte, aprenderemos el funcionamiento de un sensor PIR, en concreto el PIR para Arduino HC-SR501, que podemos ver en la figura 1. Realizaremos el montaje y la programación con mBlock utilizando el objeto *Oso*, haciendo que éste aparezca y desaparezca en función de si hemos o no detectado movimiento. También reproduciremos un sonido al más puro estilo de alarma cuando detectemos movimiento.
2. Añadiremos al circuito anterior un zumbador. Un zumbador es un circuito electroacústico que produce un sonido continuo o intermitente, generalmente en un mismo tono [2]. Se suele utilizar como mecanismo de señalización o aviso en múltiples sistemas. Nosotros utilizaremos el zumbador para Arduino de la figura 2.



Figura 1: PIR HC-SR501. Fuente: www.crelectrons.com



Figura 2: Zumbador Activo 5v. Fuente: www.ebay.es

El HC-SR501 tiene tres terminales para su conexión, uno de alimentación (5v) que conectaremos a un pin de 5v de nuestra placa, otro de tierra (GND) que conectaremos a tierra y otro de señal que conectaremos a uno de los pines digitales de nuestra placa. En esta ocasión elegimos el pin 7. El montaje lo podemos ver en el esquema de la figura 3.

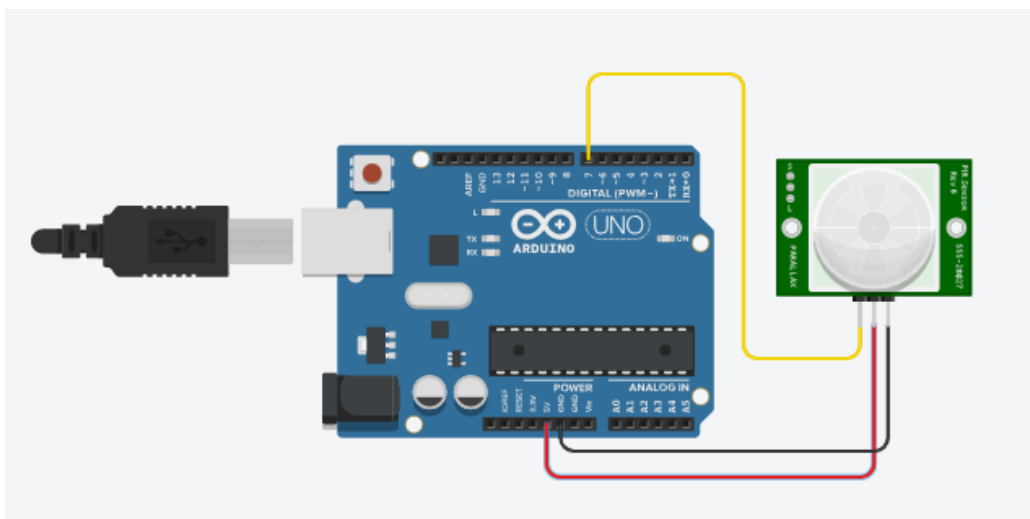


Figura 3: Esquema de montaje. Fuente: Propia

El HC-SR501 actúa de la siguiente manera: cuando no detecta movimiento, su salida está a cero y, cuando detecta un movimiento, pone su salida a uno [26]. El radio de acción se puede regular mediante dos potenciómetros que lleva incorporados. En lo que se refiere a la parte de programación, vamos a utilizar una variable, que llamaremos *detector*, donde

guardaremos un valor uno o cero, dependiendo si el sensor detecta o no movimiento. Para la programación de esta primera parte de la actividad, vamos a utilizar un modo que permite mBlock: subir el programa a la placa Arduino manteniendo la conexión USB. Este modo nos permite utilizar alguna categoría más de bloques, como pueden ser las de *Movimiento* y *Sonidos*. En nuestro programa, cada vez que detectemos un movimiento, aparecerá el objeto *Oso* y se reproducirá el sonido *Eat*. La programación con mBlock nos quedará como muestra la figura 4.

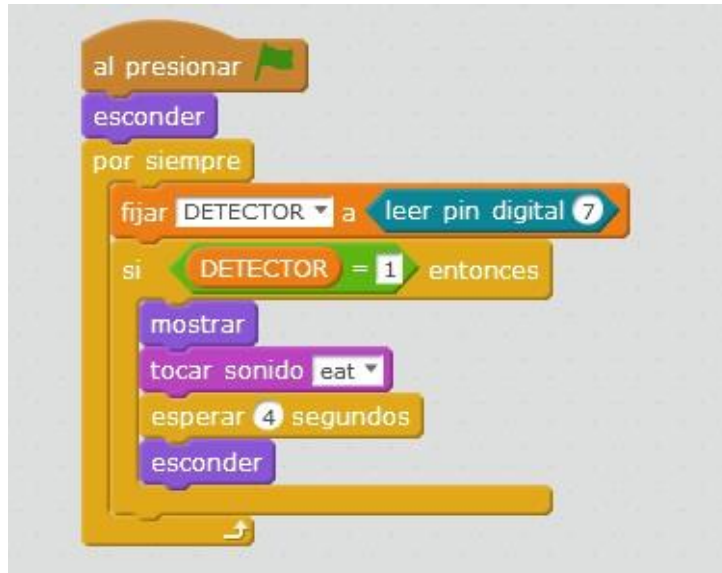


Figura 4: Bloque de programa mBlock. Fuente: Propia

Para la segunda parte de la actividad, sólo nos queda conectar el zumbador al Arduino y cambiar algunas sentencias del programa mBlock.

El zumbador sólo tiene dos terminales, uno positivo, que conectaremos a un pin salida digital PWM de nuestra placa y el negativo que conectaremos a GND. El esquema de conexiones que empleamos es el de la figura 5.

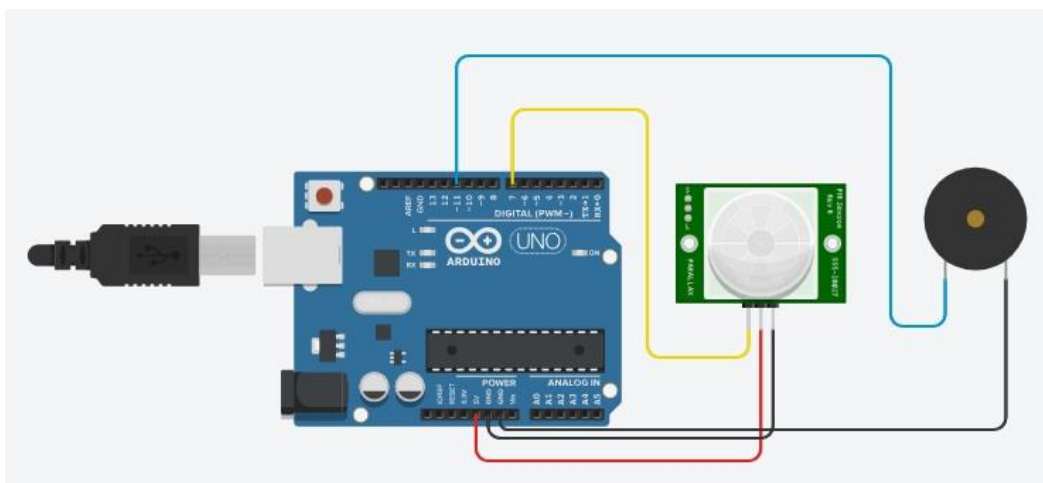


Figura 5: Esquema de montaje. Fuente: Propia

En cuanto a la parte de programación con mBlock, vamos a sustituir la aparición del objeto mBlock “oso” y la sentencia de emisión de sonido, por una sentencia donde activamos el pulsador, como podemos ver en la figura 6.



Figura 6: Bloque de programa. Fuente: Propia

Es importante que para que el circuito nos funcione, esta vez debemos desconectarlo del puerto USB y alimentar la placa con una pila de 9V. El montaje real corresponde a la figura 7.

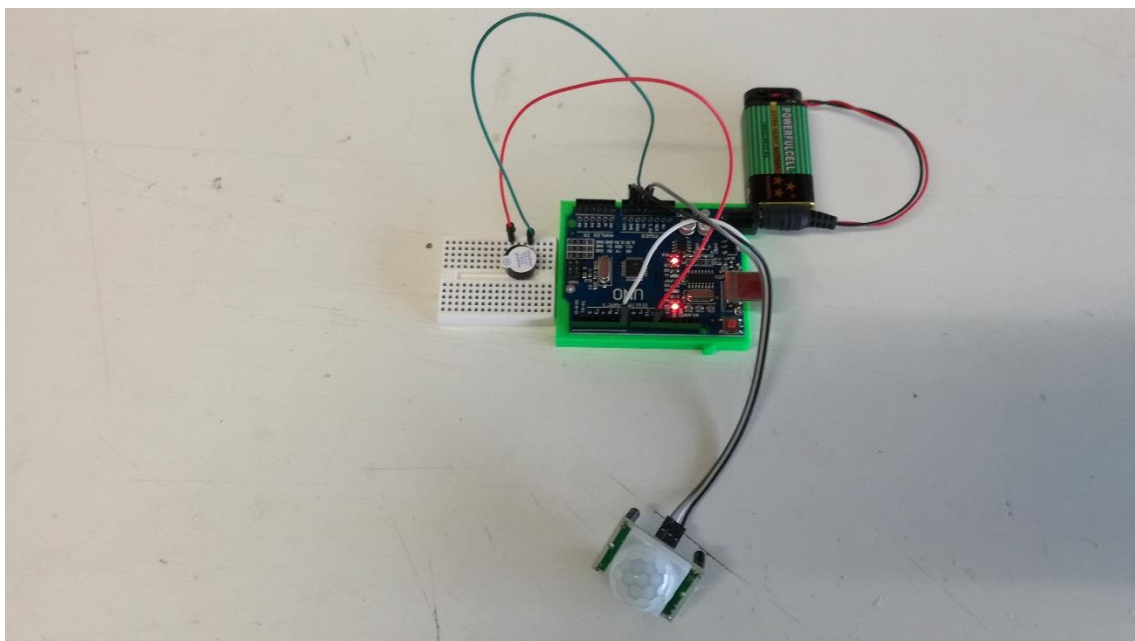


Figura 7: Esquema de montaje real. Fuente: Propia

Evaluación

Para la evaluación de esta actividad, utilizaremos la siguiente rúbrica:

RÚBRICA EVALUACIÓN ACTIVIDAD 7	No le ha quedado claro (1)	Le ha quedado medianamente claro (2)	Le ha quedado muy claro (3)
Sabe cómo funciona un zumbador			
Comprende el funcionamiento del sensor PIR HC-SR501			
Sabe cómo resolver el objetivo de la práctica y lo programa con mBlock			
Realiza de forma correcta las conexiones en la Protoboard			
Realiza de forma correcta las conexiones en el Arduino			

Fuente: Propia

4.2.7 Actividad 8: ¡Mejor con luz!

Objetivos

Con esta actividad aprenderemos a medir la intensidad de luz utilizando una fotoresistencia LDR o fotocélula.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).

Metodología

- Trabajo individual.
- Clase magistral.
- Aprendizaje basado en problemas.
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 1 fotoresistencia LDR.
- 5 LEDES.
- 5 resistencias de 220Ω .
- 1 resistencia de $10k\Omega$.
- Cables para realizar las conexiones.

Duración Actividad

Dos sesiones de 50-55 minutos.

Descripción de la Actividad

En esta actividad aprenderemos a medir la intensidad de luz utilizando una fotoresistencia LDR y una entrada analógica y utilizar la intensidad de la luz que se medirá para controlar un número de LEDES, 5 en este caso.

La fotocélula que vamos a usar está representada en la figura 1, es un resistor dependiente de luz y actúa como una resistencia variable en función de la cantidad de luz que incida sobre ella.



Figura 1: LDR C2795. Fuente: www.taringa.net

Esta fotocélula que vamos a usar presenta una resistencia de alrededor de $50k \Omega$ en total oscuridad y 500Ω en luz brillante [27]. Para convertir este valor de resistencia en algo que podamos medir con un pin analógico, ésta debe ser convertida en un voltaje. La forma más sencilla para hacer esto es compararla con una resistencia fija, como podemos observar en la figura 2.

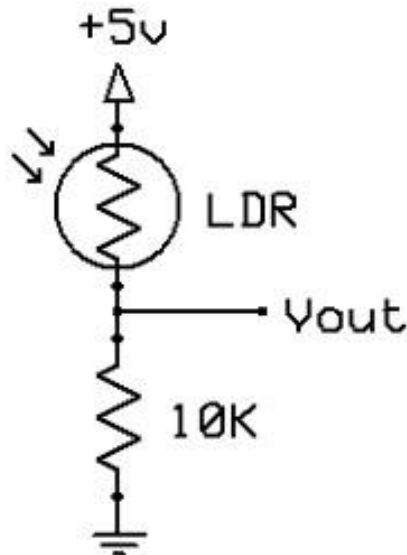


Figura 2: Esquema conexión LDR. Fuente www.dtic.upf.edu

Cuando la luz es muy brillante, la resistencia de la fotocélula es muy pequeña en comparación con la fija, por lo que toda la tensión caerá en la resistencia fija, y V_{out} tenderá a 5v; cuando la luz está apagada sucede al revés, es decir, la resistencia de la fotocélula es mucho mayor que la fija, por lo que V_{out} tenderá a 0v. Vamos a diseñar un circuito para medir la intensidad de la luz que incide en la LDR. Este circuito tendrá 5 LEDS, que dependiendo el nivel de intensidad lumínica se irán encendiendo o apagando. Los cinco LEDS estarán encendidos cuando la intensidad lumínica sea máxima, claridad total, y sólo uno cuando la intensidad sea mínima, que se corresponderá con mucha oscuridad. El esquema del circuito es el de la figura 3.

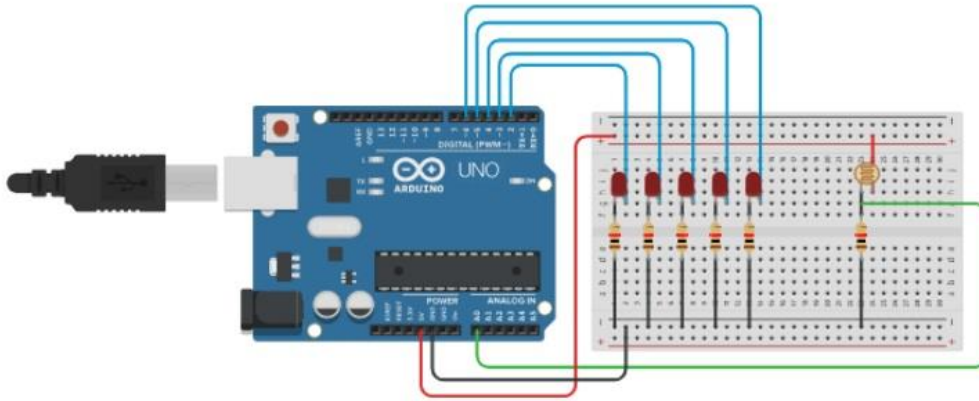


Figura 3: Esquema circuito LDR. Fuente: Propia, realizada con Tinkercad

Para la parte de mBlock, necesitaremos una variable, que llamaremos *LDR*, donde guardar las lecturas de la intensidad en la LDR que se vayan produciendo. Sabemos que al ser una entrada analógica, va a presentar valores comprendidos entre 0 y 1023. Los más cercanos a cero significarán mucha intensidad luminosa y los más cercanos a 1023, oscuridad. Nosotros vamos a dividir ese rango en cinco partes, cada una de ellas regida por un LED, y según sea la intensidad luminosa, se irán encendiendo o apagando más o menos LEDs.



Figura 4: Bloque de programa mBlock. Fuente: Propia

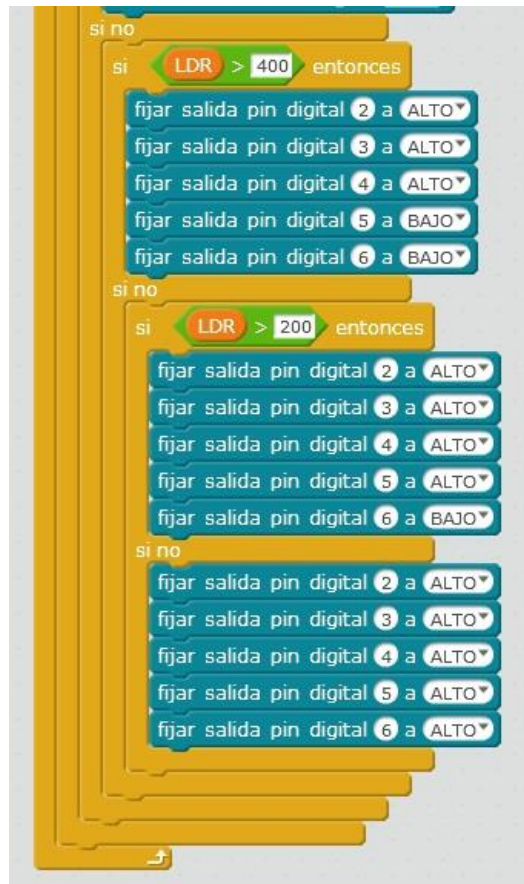


Figura 5: Bloque de programa mBlock. Fuente: Propia

Hemos utilizado los pines digitales 2, 3, 4, 5 y 6 como salidas para activar los LEDS, y el pin analógico A0 como entrada de la señal de la LDR. El programa en mBlock es el que puede verse en las figuras 4 y 5.

En las figuras 6 y 7, podemos apreciar el montaje real del circuito. En la figura 6 observamos todos los LEDS encendidos, como corresponde a una gran intensidad luminosa, y en la figura 7 tapamos la LDR con un dedo (simulando oscuridad), con lo que podemos observar que los LEDS se apagan.

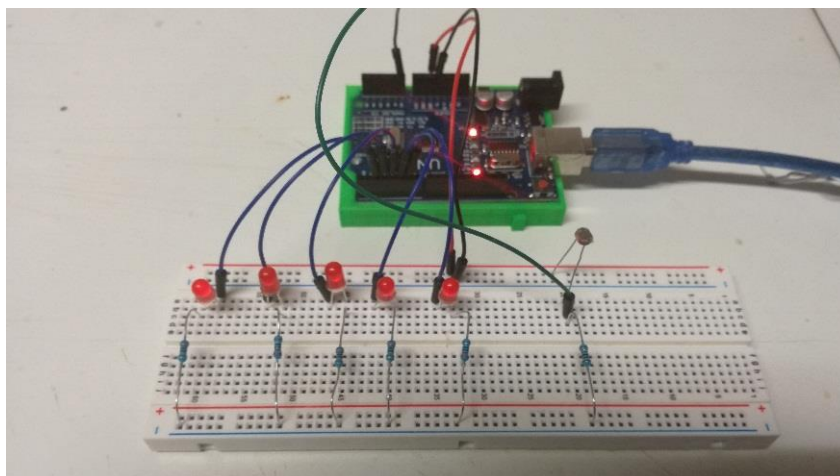


Figura 6: Montaje real LDR. Fuente: Propia

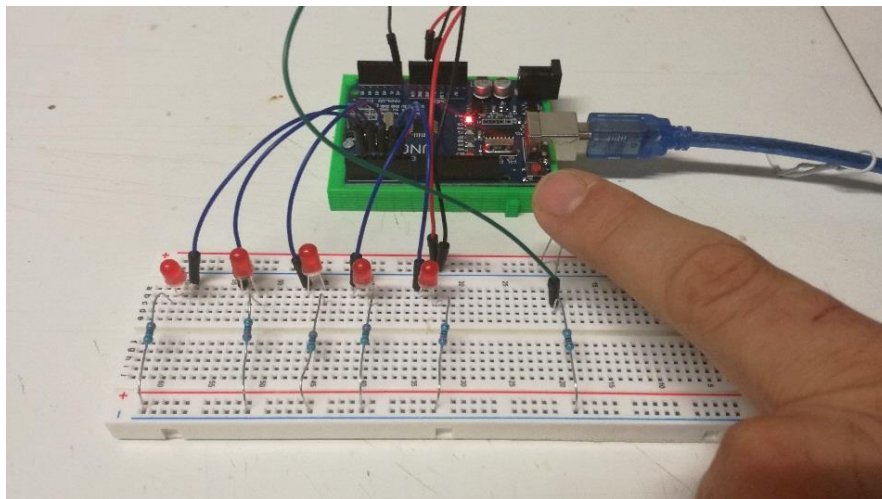


Figura 7: Montaje real LDR. Fuente: Propia

Evaluación

La actividad será correcta si, a medida que aumenta o disminuye la intensidad luminosa en la LDR, aumenta o disminuye el número de LEDs encendidos.

4.2.8 Actividad 9: Radar de velocidad

Objetivos

Construir un radar de velocidad. Aprender a manejar un sensor ultrasónico y un LCD 2x16 (2líneas de 16 caracteres cada una).

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).
- Competencias sociales y cívicas (CSC).

Metodología

- Trabajo en grupos de 2 personas.
- Clase magistral.
- Aprendizaje basado en problemas. Aprendizaje significativo
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 1 sensor ultrasónico HC-SR04 o similar.
- 1 pantalla LCD 2x16
- Cables para realizar las conexiones.
- Materiales para confeccionar soportes (cartón, madera, plástico, etc).

Duración Actividad

Dos sesiones de 50-55 minutos.

Descripción de la Actividad

En esta actividad vamos a construir un radar de velocidad para medir la velocidad de un objeto. Esta velocidad se puede calcular de varias formas y con varios sensores diferentes. Nosotros calcularemos la velocidad utilizando el sensor ultrasónico HC-SR04 de la figura 1, mediante el cual realizaremos mediciones de la distancia del objeto al radar. Una vez obtenidas estas mediciones, mediante unos sencillos cálculos obtendremos la velocidad del objeto y la visualizaremos empleando el control LCD 2x16 de la figura 2.

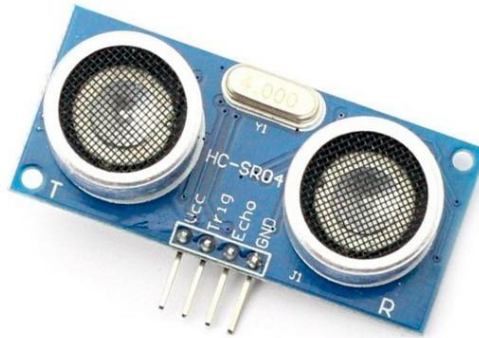


Figura 1: sensor HC-SR04. Fuente: <http://www.cetronic.es>

Este sensor realiza dos funciones principales:

- Puede detectar si un objeto se presenta, al igual que un sensor PIR (Passive Infrared Sensor).
- Puede transmitir la distancia al objeto (en cm).

Este sensor ultrasónico normalmente se emplea para medir distancias ya que puede detectar un objeto desde una distancia de 2 cm hasta 400 cm. Posee cuatro pines para su conexión: alimentación (Vcc), tierra (GND), disparo del ultrasonido (Trig) y recepción del ultrasonido (Echo).

Su funcionamiento es el siguiente [35]: El sensor envía un pulso por el *Trigger* (disparador) a la vez que empieza a contar el tiempo. La salida *Echo* se mantendrá a nivel alto hasta que reciba el eco del pulso enviado por el *Trigger*, momento en el que se parará el tiempo. La distancia obtenida es proporcional a la duración del pulso y puede obtenerse mediante la fórmula:

$$\text{Distancia en centímetros (cm)} = \text{Tiempo medido en microsegundos (us)} \times 0.017$$

El otro dispositivo que vamos a utilizar es el LCD 16x2. Éste es un controlador muy común utilizado ampliamente en proyectos de Arduino [36]. Como este tipo de pantalla requiere muchos pines para ser controlada, debido principalmente a que utiliza un bus en paralelo para comunicarse, hacemos uso de una solución muy fácil y económica para este problema: un adaptador que permite conectar la pantalla al Arduino usando solamente dos líneas digitales a través del bus I2C. El controlador LCD + adaptador I2C se representan en la figura 2.



Figura 2: LCD 2x16 y adaptador I2C. Fuente: <https://www.luisllamas.es>

Nosotros vamos a implementar nuestro radar de la siguiente forma: siempre estaremos leyendo una distancia (dist1) y otra una décima de segundo después (dist2) en nuestro sensor ultrasónico. De esta forma, cuando un objeto entre en el rango de nuestro sensor, se le irán haciendo lecturas de su posición cada décima de segundo. En el momento en el que la distancia resultado de la primera lectura sea menor de 70 cm pasaremos a calcular la velocidad. Esta velocidad la calcularemos mediante la fórmula:

$$velocidad \left(\frac{cm}{seg} \right) = \frac{dist1 - dist2}{0.1}$$

Ahora sólo nos queda enviar el resultado de nuestro cálculo a la pantalla del LDC y con esto tendremos implementado nuestro radar de velocidad.

La conexiones entre nuestra placa Arduino y nuestros sensores las mostramos en la figura 3 y el programa mBlock utilizado para esta actividad en la figura 4.

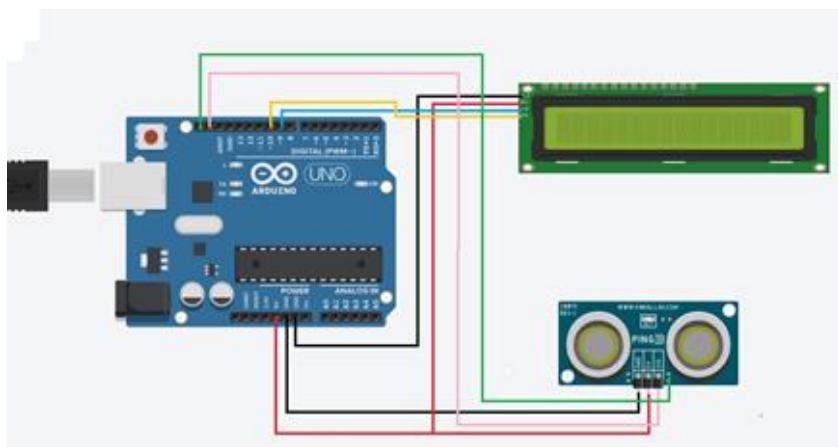


Figura 3: Conexión LCD Bus I2C y HC-SR04. Fuente: Propia

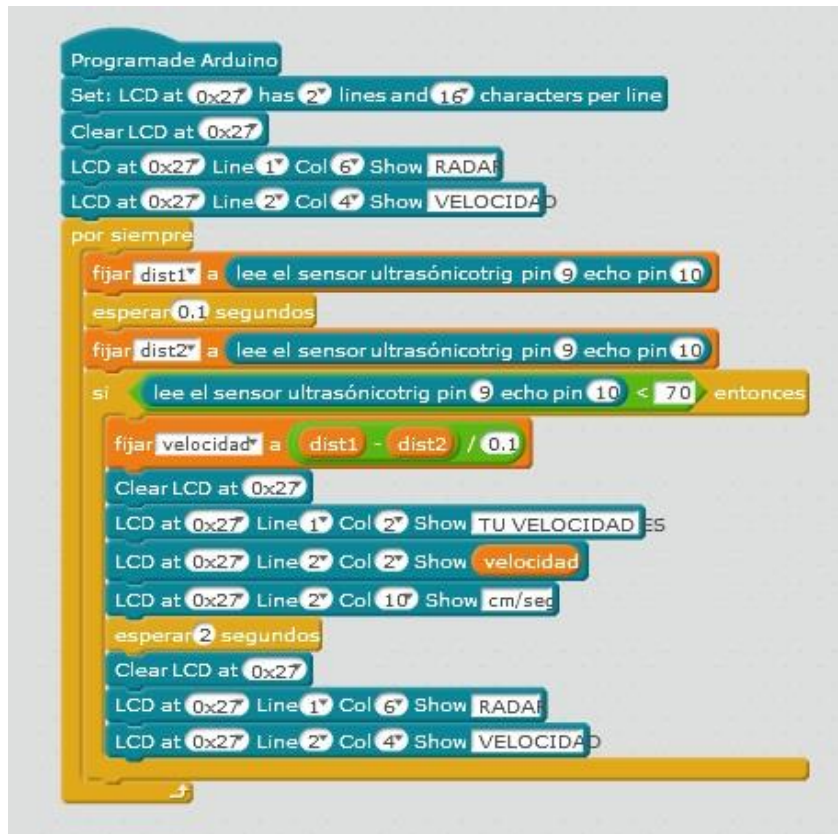


Figura 4: Programa mBlock. Fuente: Propia

Para el programa representado en la figura 4, hemos utilizado tres variables; dos para las distancias (dist1 y dist2) y una para almacenar la velocidad (velocidad). Las figuras 5, 6 y 7 nos muestran diversos detalles de nuestro radar.

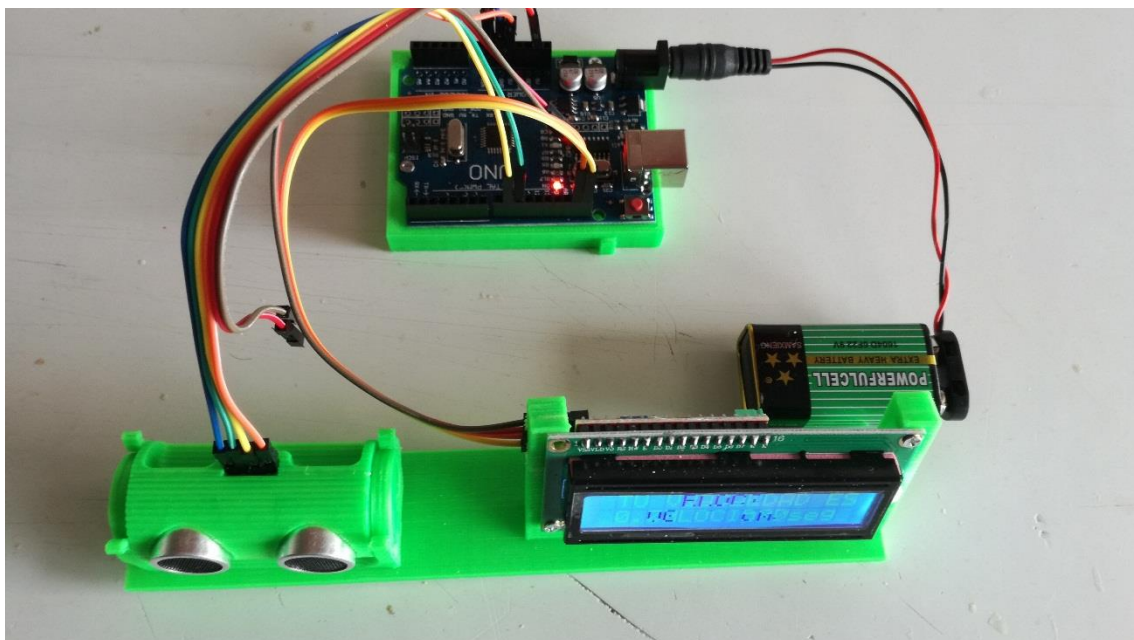


Figura 6: Montaje real del radar. Fuente: Propia

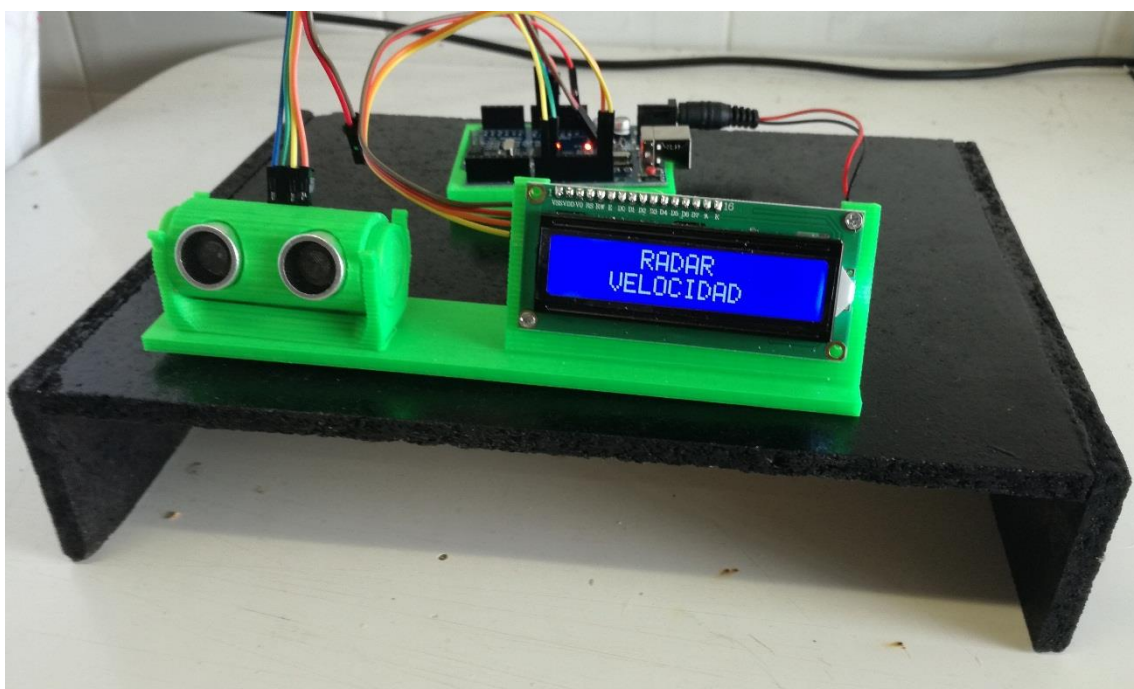


Figura 6: Detalle implantación radar. Fuente: Propia

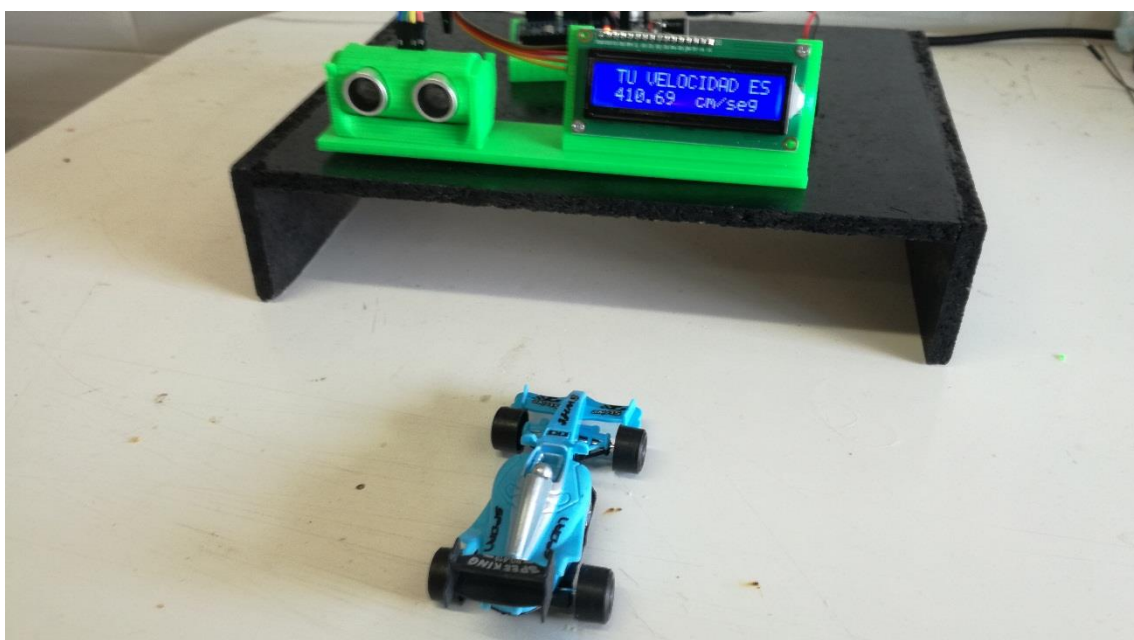


Figura 7: Detalle funcionamiento radar. Fuente: Propia

Evaluación

Para la evaluación de esta actividad, utilizaremos siguiente rúbrica.

RÚBRICA EVALUACIÓN ACTIVIDAD 9	No nos ha quedado claro (1)	Nos ha quedado claro (2)	Nos ha quedado muy claro (3)
Comprendemos el funcionamiento del control LCD I2C			
Comprendemos el funcionamiento del sensor ultrasónico HC-SR04			
Realizamos de forma correcta las conexiones en el Arduino			
Sabemos realizar el cálculo de la velocidad			
Sabemos cómo realizar el programa con mBlock			

Fuente: Propia

4.3- Actividades de construcción de robots

4.3.1 Actividad 10: ¡No te caigas!

Objetivos

Construir un vehículo que evite caerse por los bordes de una superficie. Aprender a manejar un sensor de infrarrojos.

Al final de las sesiones, se establecerá una pequeña competición que será ganada por el vehículo que más tiempo se mantenga encima de la superficie.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).
- Competencias sociales y cívicas (CSC).

Metodología

- Trabajo en grupos de 2 personas.
- Clase magistral.
- Aprendizaje basado en problemas. Aprendizaje significativo
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.
- Gamificación.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 2 motores 3v-6v DC y 2 ruedas.
- 1 chasis estilo vehículo.
- 1 porta baterías.
- 1 módulo L298N.
- 1 protoboard pequeña.
- 2 sensores de infrarrojos IR TCRT5000 o similar.
- 2 baterías de 6v y 9v respectivamente.
- Materiales para confeccionar el vehículo (cartón, madera, plástico, etc).
- Cables para realizar las conexiones.

Duración Actividad

Cuatro sesiones de 50-55 minutos.

Descripción de la Actividad

En esta actividad, el vehículo comenzará a rodar en una dirección cualquiera sobre una superficie y, cuando detecte el final de ésta, dará un giro de 90° aproximadamente y continuará rodando en una nueva dirección hasta que vuelva a detectar de nuevo el final de la superficie y así sucesivamente. Cuando todos los vehículos están contruidos, se establecerá una competición que será ganada por el grupo que haya construido el robot que consiga mantenerse durante más tiempo encima de la superficie.

La actividad comenzará con una breve clase magistral en la que explicaremos que un dispositivo detector de obstáculos infrarrojo es un dispositivo que detecta un objeto mediante la reflexión que produce en él la luz. El motivo de usar luz infrarroja (IR) es que no sea visible para los humanos [16].

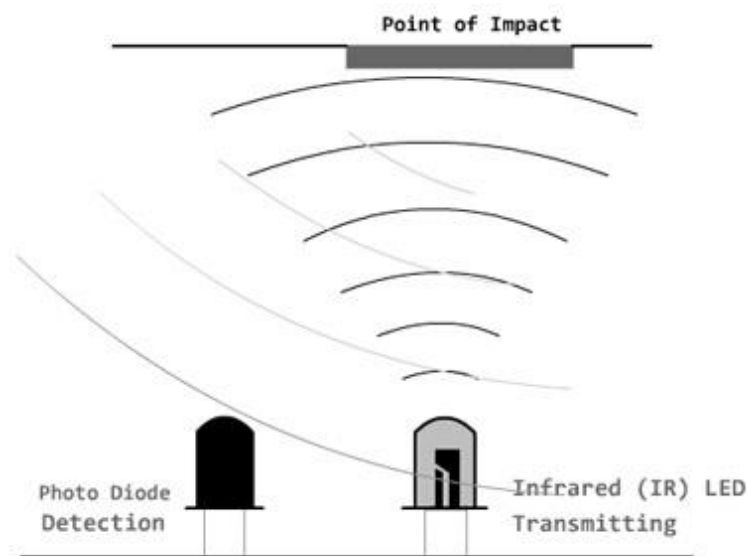


Figura 1: Fundamento sensor Infrarrojos. Fuente: www.luisllamas.es

Estos sensores suelen estar compuestos de un diodo LED IR, encargado de emitir una señal luminosa, y de un fotodiodo que recibe la luz reflejada por el posible obstáculo, como podemos ver en la figura 1. Además, los detectores de obstáculos suelen proporcionarse con una placa de medición que permite obtener la lectura como un valor digital.

Nosotros para esta actividad vamos a utilizar el detector de la figura 2, que es similar al detector infrarrojos TCRT5000, y cuyo funcionamiento se caracteriza por cambiar el nivel de su señal de salida, de 1 a 0 o viceversa, en función de si el fotodiodo recibe la reflexión de la luz emitida por el diodo IR o no. Este sensor actúa a distancias cortas, de 2 a 40 mm, y esa distancia la podemos graduar a través del potenciómetro que lleva incorporado.



Figura 2: Detector IR. Fuente: www.prometec.net

También en esta actividad vamos a hacer uso de dos motores 3v – 6v DC con reductora incorporada y dos ruedas, como podemos ver en la figura 3.



Figura 3: Rueda + motor para vehículo. Fuente: www.prometec.net

Después de esta introducción al sensor IR, tendremos que elaborar un chasis para nuestro vehículo. Éste podemos diseñarlo como queramos y del material que queramos; madera, cartón duro, plástico, e incluso diseñar un modelo 3D con una herramienta de diseño 3D libre como pueda ser *Tinkercad* [31], para después imprimirlo con una impresora 3D, etc. En nuestro caso hemos utilizado una impresora 3D para imprimir el modelo de un chasis que previamente hemos descargado de *Thingiverse* [32]. El chasis utilizado es el mostrado en la figura 4.

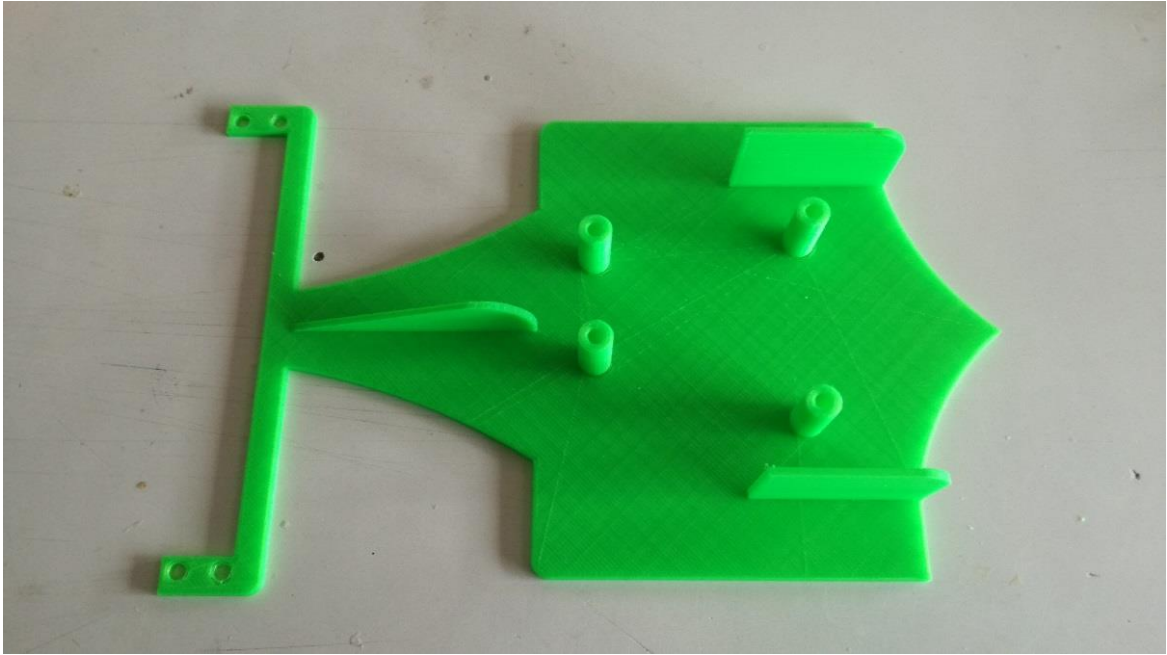


Figura 4: Chasis vehículo. Diseño: Brunetica. Fuente: Propia

Una vez tengamos el chasis, le añadiremos el porta baterías, los dos motores, el controlador L298N, la placa Arduino Uno, una protoboard pequeña y los dos sensores IR, tal y como muestran las figuras 5 y 6.

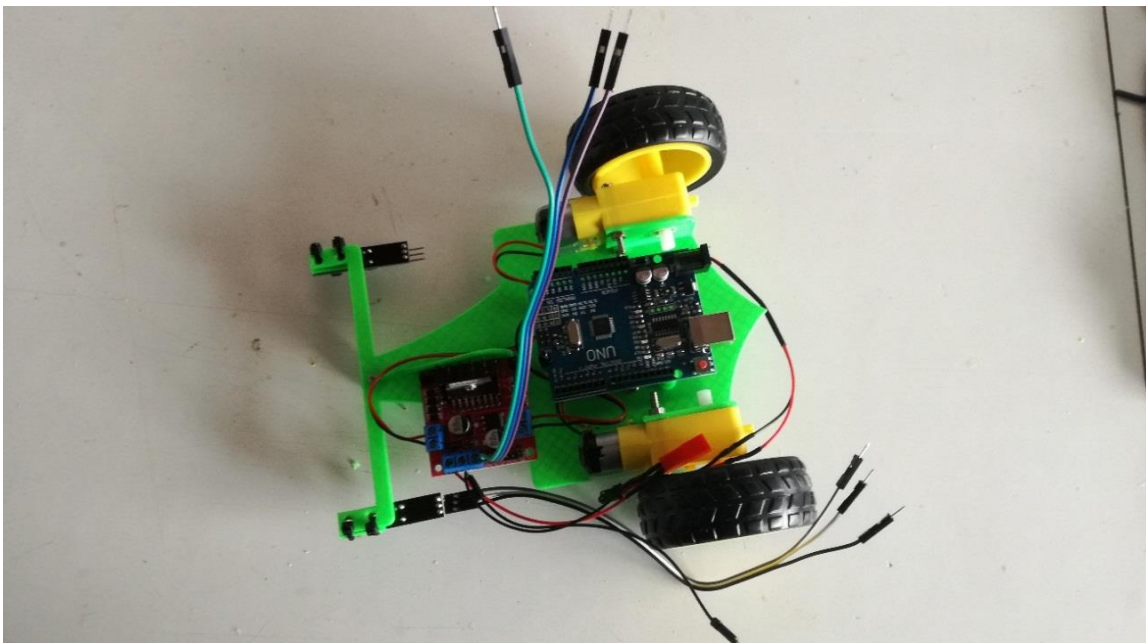


Figura 5: Detalle instalación motores, L298N y placa Arduino Uno. Fuente propia.

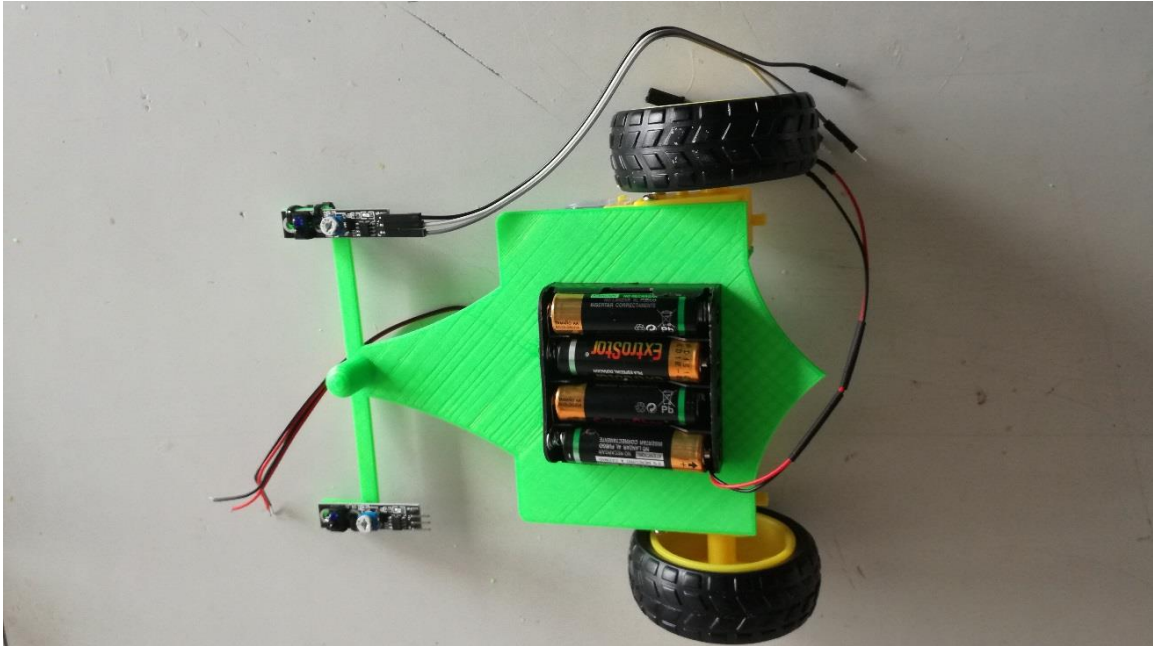


Figura 6: Detalle instalación del porta baterías y sensores IR. Fuente: propia

Una vez montados los componentes, nos queda realizar las conexiones entre los mismos, como muestra la figura 7. En la figura 8 podemos ver el vehículo montado con todas sus conexiones establecidas.

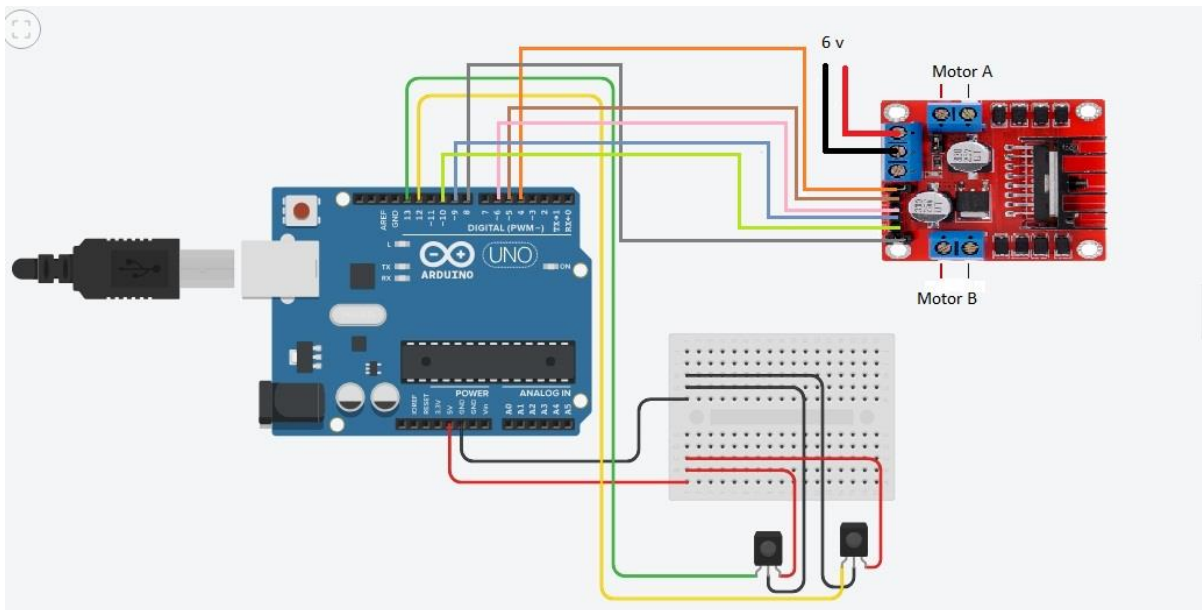


Figura 7: Esquema de conexiones. Fuente: Propia

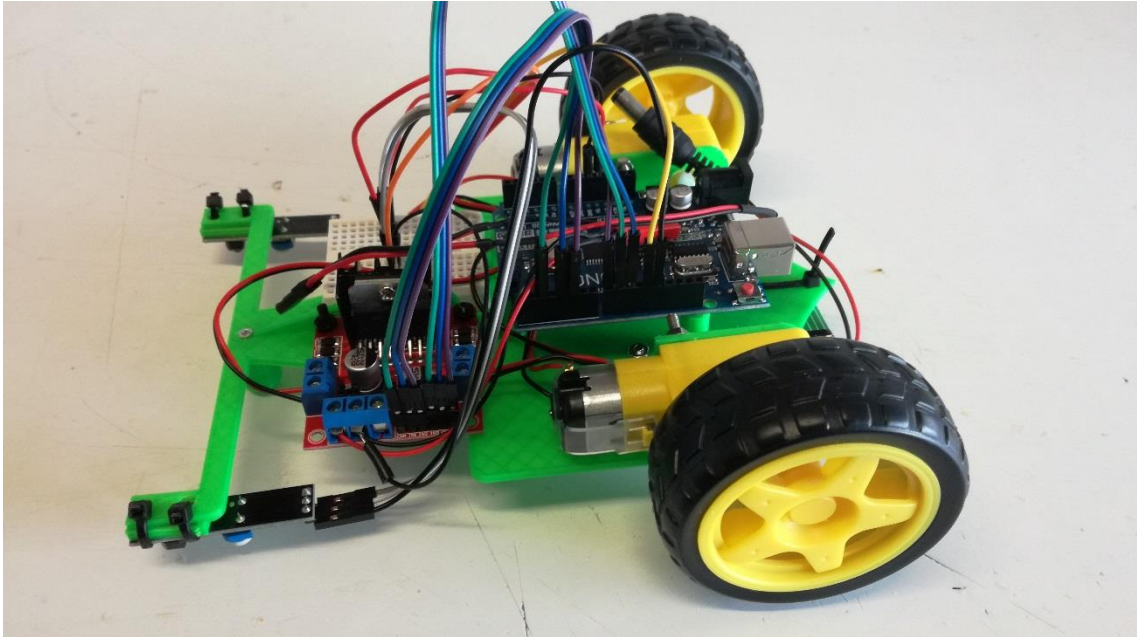


Figura 8: Detalle vehículo montado y conexiones realizadas. Fuente: Propia

Ahora que tenemos montado y preparado nuestro vehículo, vamos a programarlo: Hemos dividido el movimiento del vehículo en 5 partes: adelante, atrás, derecha, izquierda y parar, de tal manera que cada una de esas partes la hemos implementado con un bloque propio. Lo hemos hecho así para poder ver más claramente la parte de programación del vehículo. Los bloques o subrutinas son los pertenecientes a las figuras 9 y 10.

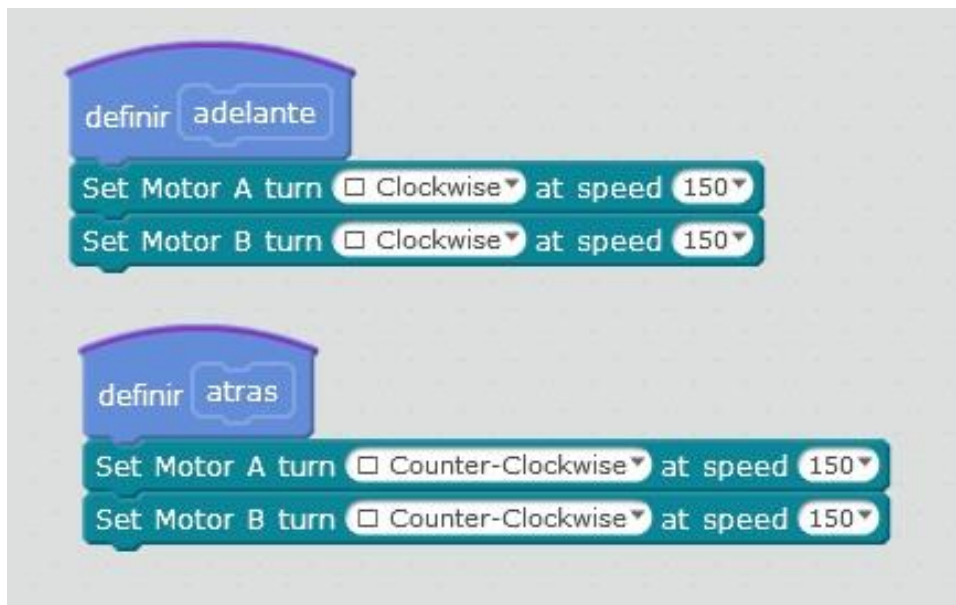


Figura 9. Bloques *Adelante* y *Atrás*. Fuente: Propia

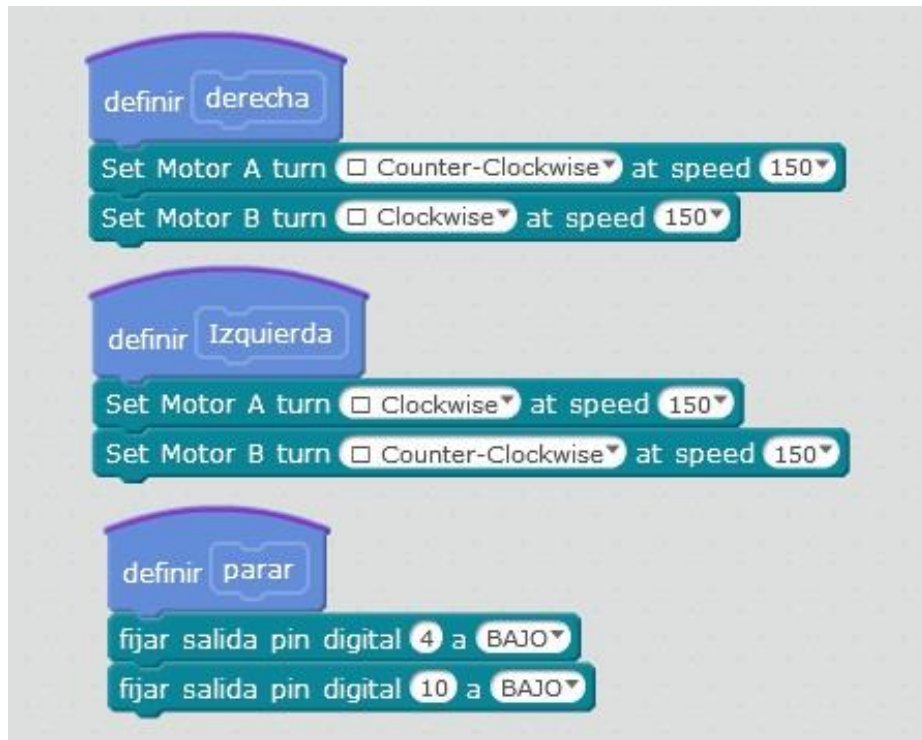


Figura 10: Bloques *Derecha*, *Izquierda* y *Parar*. Fuente: Propia

El movimiento del vehículo lo hemos programado de la siguiente forma: Utilizamos dos variables, *SensorD* y *SensorIZ*, para almacenar las lecturas de los sensores IR. Si los dos están a 1 significa que el vehículo se sale de frente de la plataforma, por lo que le mandamos *Parar*, *Atrás* durante dos segundos; *Derecha*, con lo que realiza un giro de aproximadamente 90° (que es lo que gira en 1 segundo) y *Adelante*, para reanudar la marcha. En el caso de que *SensorD* se ponga a 1 y *SensorIZ* a 0, el vehículo se sale de la plataforma por la derecha, con lo cual le corregimos mandando *Atrás*, durante un segundo y luego *Izquierda* durante otro segundo. En el caso de que *SensorIZ* se ponga a 1 y *SensorD* a 0, ocurriría lo contrario, el vehículo se sale por la izquierda, con lo cual le corregimos mandando *Atrás* durante un segundo y luego *Derecha*, durante otro segundo. Después de estos movimientos, el vehículo sigue su marcha y el ciclo vuelve a empezar. Este bloque de programación podemos verlo representado en la figura 11.

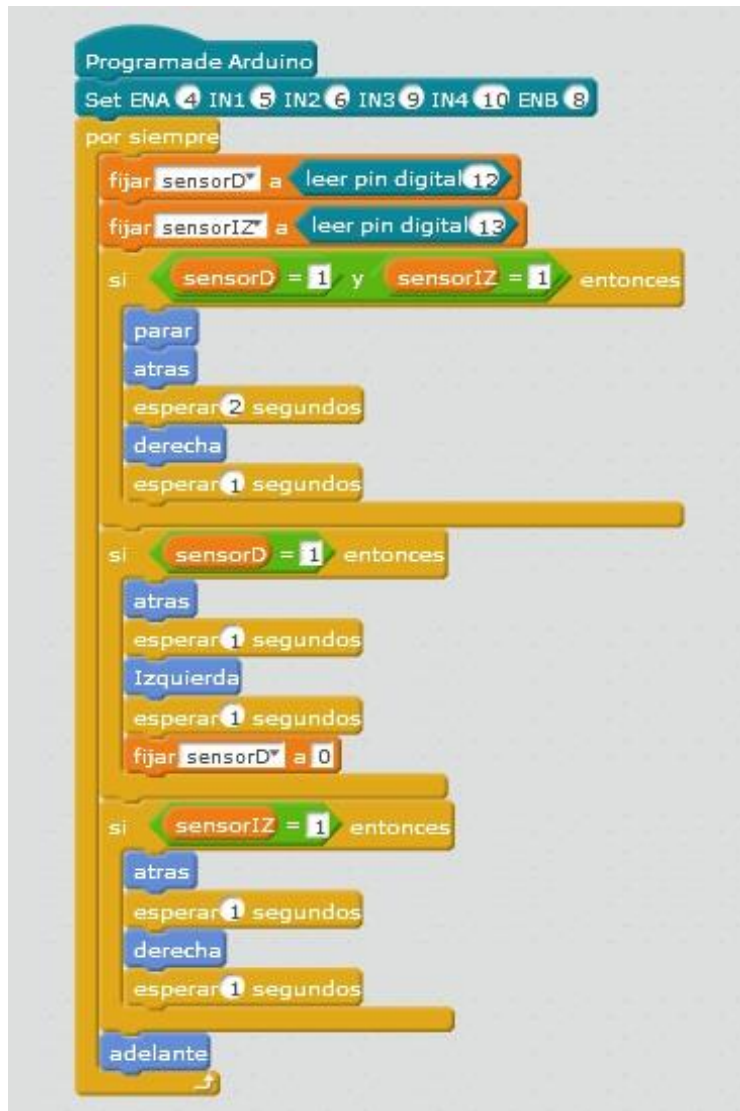


Figura 11: Programa mBlock. Fuente: Propia

Evaluación

Los alumnos realizarán un informe sobre la actividad (**anexo 4**).

4.3.2 Actividad 11: ¡Pilota tu robot!

Objetivos

Aprender a utilizar un dispositivo Bluetooth para controlar un robot desde un Smartphone, mediante una aplicación App de libre distribución.

Construir un pequeño robot para ser dirigido desde la App de nuestro Smartphone.

Al final de las sesiones, se establecerá una pequeña competición que será ganada por el alumno que tarde menos tiempo en dar 2 vueltas a un circuito con obstáculo que se improvisará en el aula-taller.

Competencias

- Competencia matemática y competencias básicas en ciencia y tecnología (CMCT).
- Competencia digital (CD).
- Aprender a aprender (CPAA).
- Competencias sociales y cívicas (CSC).

Metodología

- Trabajo en grupos de 2 personas.
- Clase magistral.
- Aprendizaje basado en problemas.
- Aprendizaje basado en competencias.
- Observación directa y sistemática del alumno.
- Gamificación.

Recursos

- Una placa Arduino Uno.
- Un PC.
- Un cable USB para conectar la placa al PC.
- 2 motores 3v-6v DC.
- 2 ruedas.
- 1 chasis vehículo.
- 1 porta baterías.
- 1 módulo controlador de motores DC L298N.
- 1 protoboard pequeña.
- 1 módulo Bluetooth HC-06
- 2 baterías de 6v y 9v respectivamente.
- Cables para realizar las conexiones.
- 1 Smartphone.

Duración Actividad

Cuatro sesiones de 50-55 minutos.

Descripción de la Actividad

En esta actividad vamos a construir un robot guiado desde un Smartphone. El robot deberá moverse adelante, atrás, derecha e izquierda como mínimo, pudiéndose implementar otras funcionalidades de manera opcional.

Para esta actividad los alumnos podrán tomar como base el vehículo o robot que hayan construido en la actividad anterior (actividad 10), o bien, construir uno diferente.

Nosotros hemos elegido partir del vehículo construido en la actividad 10 para desarrollar la actividad. La parte correspondiente a la utilización y conexión del controlador de motores L298N ha sido explicada con detalle en la actividad anterior.

La sesión comienza con una pequeña clase magistral en la que explicamos cómo conectar el módulo Bluetooth HC-06 (figura 1) a nuestro Arduino y configurarlo para utilizarlo con mBlock.



Figura 1: Módulo Bluetooth HC-06. Fuente: <https://www.ebay.es>

Para poder utilizar este módulo con mBlock, es necesario que nos descarguemos una extensión para poder controlar un módulo Bluetooth. Nosotros vamos a utilizar la extensión *Bluetooth HC06* para esta actividad. En la figura 2 podemos ver las sentencias que nos ofrece esta extensión para poder interactuar con el módulo Bluetooth HC-06.

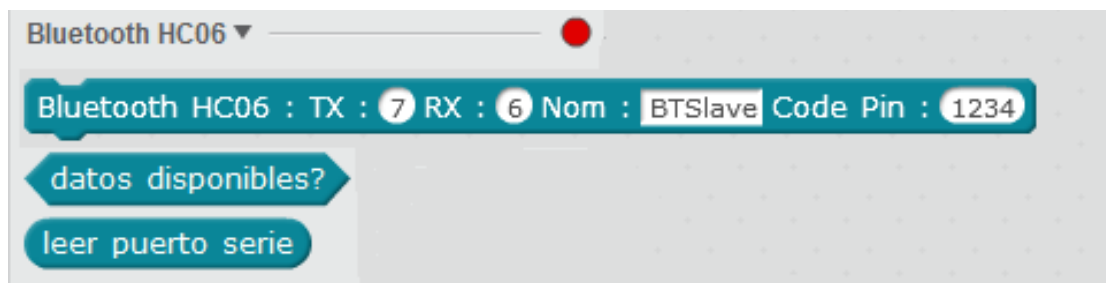


Figura 2: Comandos mBlock para módulo HC-06. Fuente: Propia

Con la primera sentencia establecemos los pines para transmitir y recibir la información, el nombre del dispositivo y la clave para poder conectarse a él (por defecto

es el 1234). La segunda sentencia nos sirve para controlar si hemos recibido algún dato por Bluetooth y la tercera para poder leer el dato, en el caso de que se haya recibido.

Para manejar Arduino mediante un Smartphone hay multitud de aplicaciones (Apps) libres en *Google Play Store* [37]; nosotros hemos optado por la App *Arduino Control Car V2*, desarrollada por *El Profe García* [38]. La interfaz de esta App la tenemos representada en la figura 3.

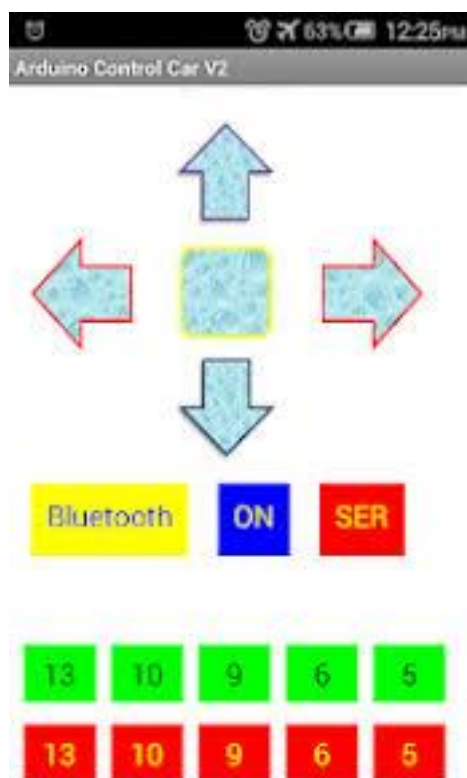


Figura 3: Arduino Control Car V2 de El Profe García. Fuente: <https://play.google.com/store/apps/>

Mediante esta App podemos mandar órdenes a nuestro robot. Con cada flecha o botón que pulsamos, el dispositivo Bluetooth de nuestro teléfono móvil envía un carácter ASCII que es recogido por el módulo HC-06 de nuestro Arduino. Nosotros podemos programar distintas acciones con nuestro robot dependiendo del carácter que recibamos.

La forma de enlazar el módulo HC-06 con nuestro Smartphone es la siguiente:

1. Conectamos el HC-06 al Arduino, éste comenzará a parpadear esperando ser enlazado.
2. Seguidamente activamos el Bluetooth de nuestro teléfono, buscamos el módulo y enlazamos.
3. Abrimos el programa Arduino Control car V2 y seleccionamos la opción Bluetooth (, (en amarillo). Nos aparecerá una ventana con distintos dispositivos como la de la figura 4. Buscamos el nuestro, en este caso el que termina en HC-06 y le seleccionamos. En unos segundos la luz del módulo dejará de parpadear indicando que la conexión está establecida. Ya tenemos nuestro dispositivo preparado para empezar a enviar órdenes.

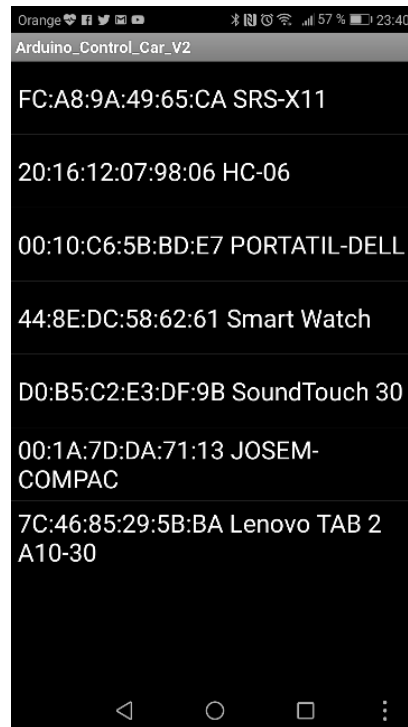


Figura 4: Arduino Control Car V2 de El Profe García. Fuente: Propia

Para la parte de programación con mBlock de nuestro robot, además de la extensión *Bluetooth HC06*, necesitaremos la extensión *L298N* para poder controlar nuestros motores DC.

Para la parte de programación hemos optado por construir unos bloques en los que definimos los movimientos de nuestro robot. Estos bloques están representados en las figuras 5 y 6.

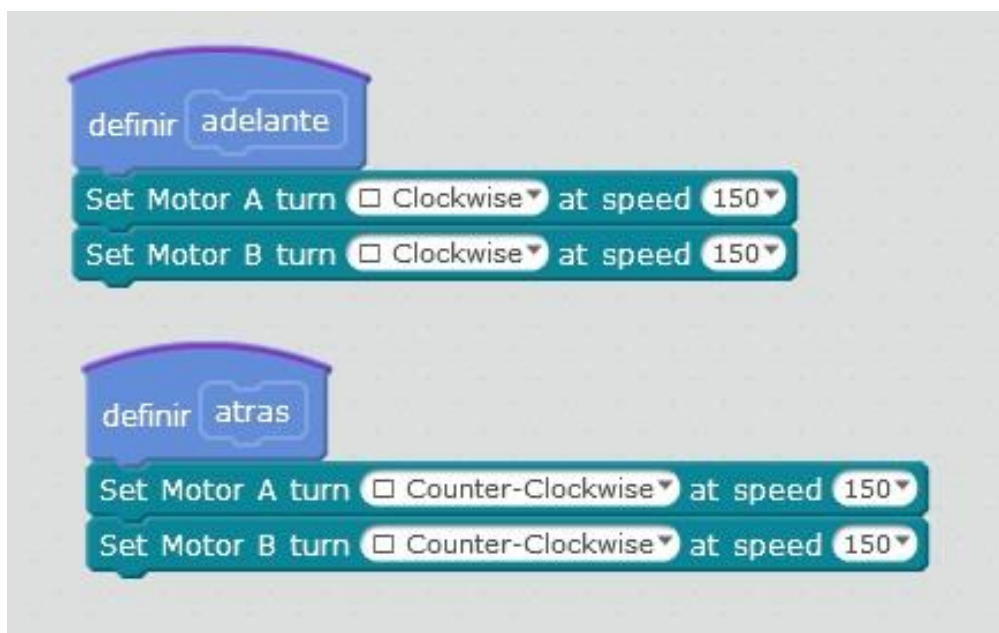


Figura 5. Bloques *Adelante* y *Atrás*. Fuente: Propia

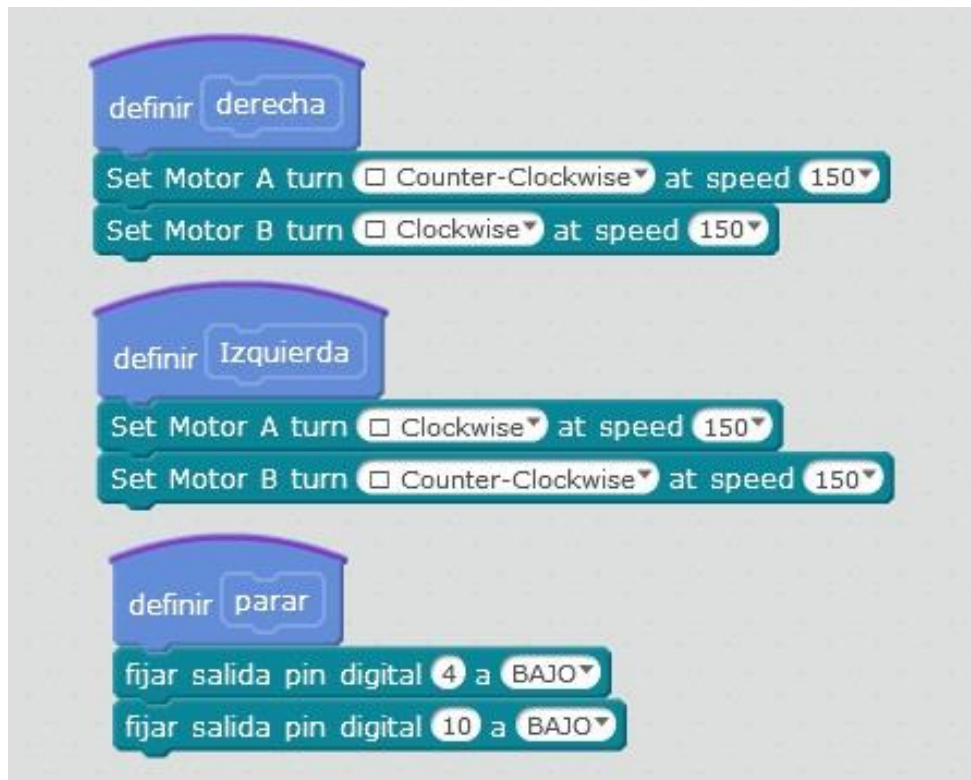


Figura 6: Bloques *Derecha*, *Izquierda* y *Parar*. Fuente: Propia

El código mBlock que hemos programado para controlar nuestro robot es el representado en la figura 7.

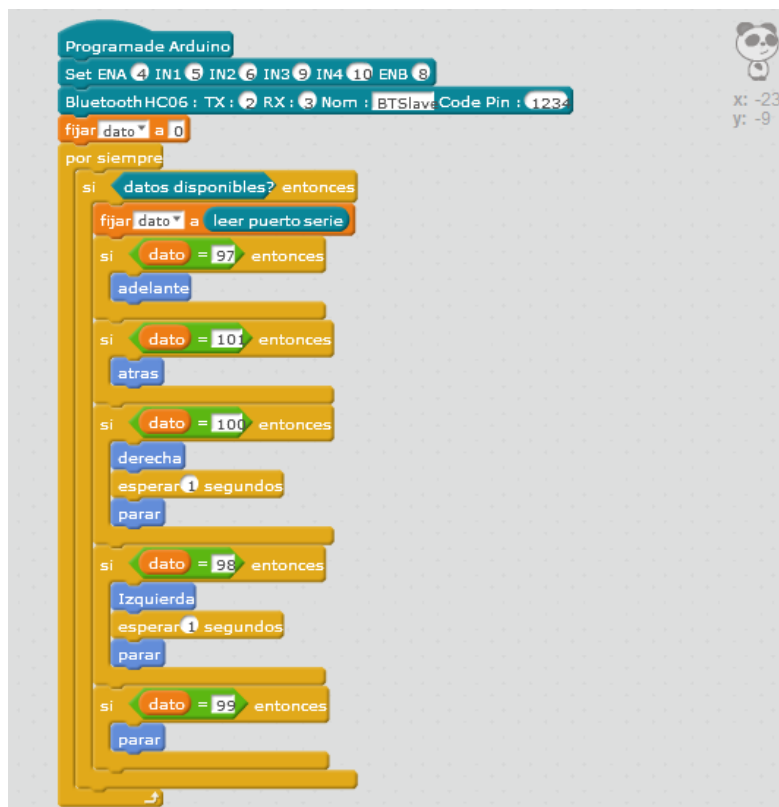


Figura 7: Programa mBlock. Fuente: Propia

El funcionamiento del programa es el siguiente: primeramente definimos dónde tenemos conectados los pines, tanto del módulo Bluetooth HC-06 como del controlador de motores L298N. Seguidamente ejecutamos un bucle en el que estamos a la espera de recibir un carácter desde nuestro Smartphone. En el caso afirmativo y dependiendo del carácter que sea, ejecutaremos una u otra acción:

- Si pulsamos **Flecha Arriba** (adelante) recibiremos el código ASCII 97, que corresponde al carácter “a”. El robot rodará hacia adelante.
- Si pulsamos **Flecha Abajo** (atrás) recibiremos el código ASCII 101, que corresponde al carácter “e”. El robot rodará hacia atrás.
- Si pulsamos **Flecha Izquierda** (izquierda) recibiremos el código ASCII 98, que corresponde al carácter “b”. El robot girará a la izquierda durante 1 segundo y luego se parará a la espera de nuevas órdenes.
- Si pulsamos **Flecha Derecha** (derecha) recibiremos el código ASCII 100, que corresponde al carácter “d”. El robot girará a la derecha durante 1 segundo y luego se parará a la espera de nuevas órdenes.
- Si pulsamos **Botón central** (parar) recibiremos el código ASCII 99, que corresponde al carácter “c”. El robot se parará a la espera de nuevas órdenes.

Estas órdenes se pueden implementar de varias formas diferentes, y su implementación será clave para si queremos ganar la competición final.

En la figura 8 podemos ver el módulo de Bluetooth con la luz fija, lo que significa que ha enlazado con nuestro Smartphone y está a la espera de órdenes.

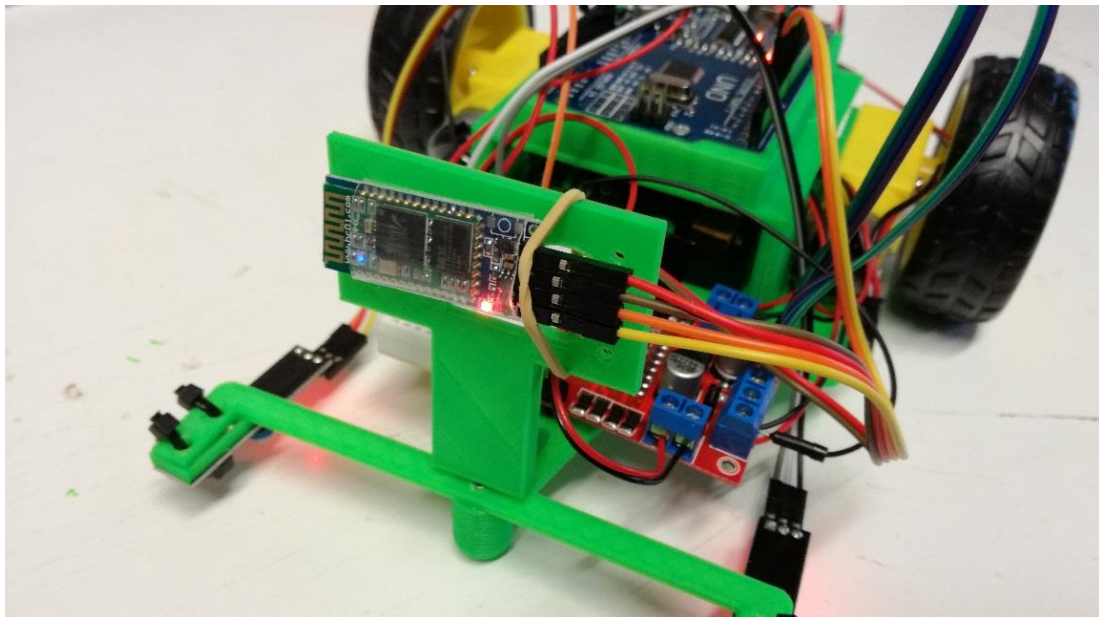


Figura 8: Módulo HC-06 enlazado. Fuente: Propia

La figuras 9 y 10 nos muestran nuestro robot ya terminado y preparado para la acción.

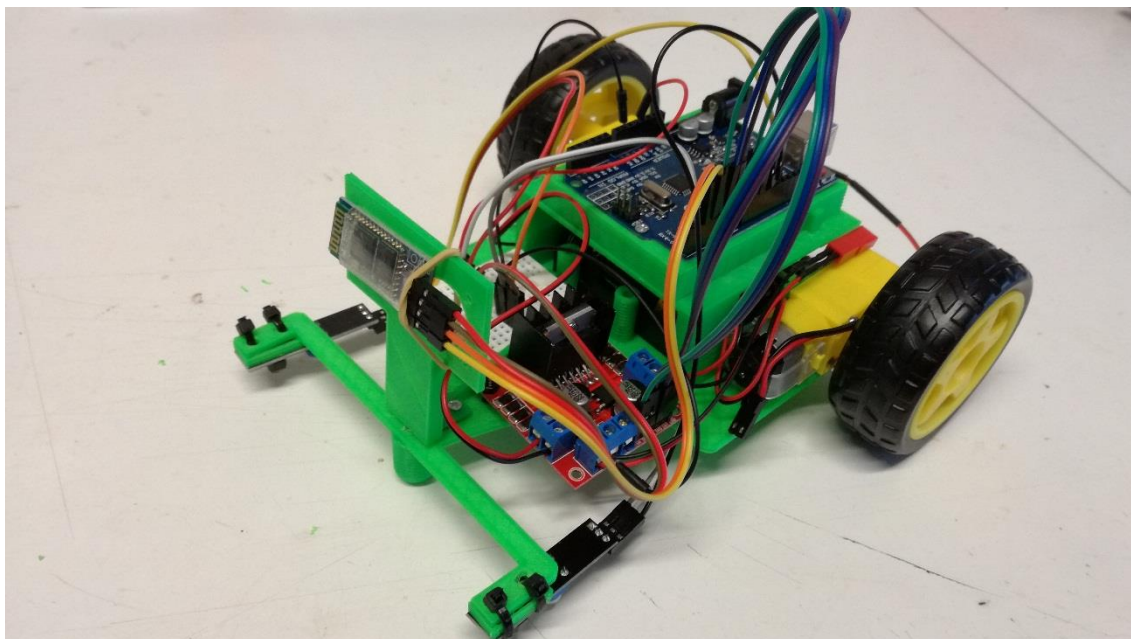


Figura 9: Detalle vehículo. Fuente: Propia

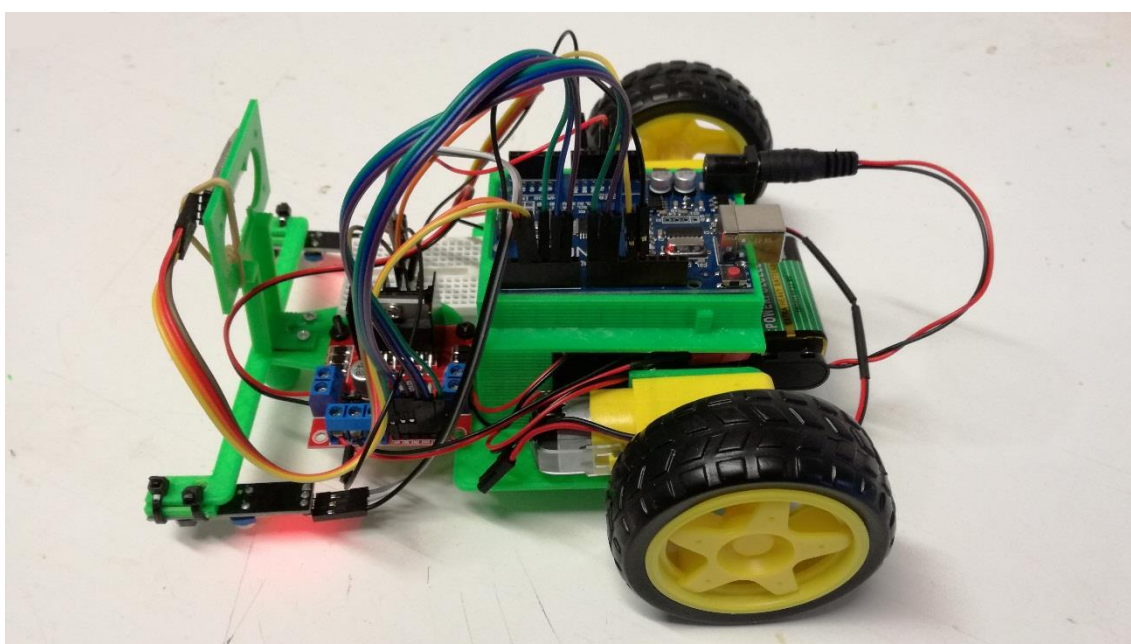


Figura 10: Detalle vehículo conectado. Fuente: Propia

Evaluación

Los alumnos realizarán un informe sobre la actividad (**anexo 4**).

4.4- Evaluación

El objetivo de este proceso es llegar a calificar el aprendizaje de los alumnos lo más objetivamente posible y es por eso que, aunque las diferentes actividades podrían evaluarse individualmente, he decidido establecer un proceso de evaluación continua para todas ellas.

Este proceso comenzará con una **evaluación inicial** que se realizará en la actividad 0 y una **evaluación final** en las actividades 10 y 11. En estas últimas dos actividades los estudiantes deberán utilizar todo lo aprendido anteriormente para poderlas realizar de una forma correcta.

El resto de actividades, aun siendo actividades de aprendizaje, también estarán sometidas a valoración. En ellas se valorará tanto el proceso como la adquisición de los contenidos necesarios para la realización de las actividades finales, constituyendo estas valoraciones un feedback del aprendizaje de los alumnos. En la tabla de la figura 1 recogemos los aspectos del proceso de evaluación.

	Momento	Lo que queremos evaluar	Para qué lo queremos evaluar
Evaluación Inicial	Actividad 0	Conocimientos	<ul style="list-style-type: none"> ▪ Para establecer un nivel de conocimientos inicial
Evaluación Continua	Durante el desarrollo de las actividades	<ul style="list-style-type: none"> ▪ Ritmo de aprendizaje ▪ Trabajo individual y en equipo 	<ul style="list-style-type: none"> ▪ Valorar el proceso de aprendizaje
Evaluación Final	Actividades 10 y 11	<ul style="list-style-type: none"> ▪ Objetivos alcanzados ▪ Progresión realizada ▪ Dificultades 	<ul style="list-style-type: none"> ▪ Valorar la progresión de cada alumno

Figura 1: Proceso de evaluación. Fuente: Propia

4.4.1- Criterios de evaluación

Están basados en la ley [10] [12] y nos van a servir para evaluar el progreso en el aprendizaje de los alumnos.

- Conocer las partes que componen un robot, así como su funcionamiento.
- Saber diseñar, programar y construir un robot sencillo.
- Conocer el lenguaje informático mBlock.
- Saber traducir un problema planteado a lenguaje informático correctamente.
- Conocer la placa Arduino Uno R3.
- Conocer los tipos de sensores más importantes: sensores de luz, sonido y de contacto.
- Saber modificar el diseño de un robot para variar su respuesta frente a estímulos determinados.
- Creatividad e iniciativa en la participación y en la solución de los retos planteados.

4.4.2- Herramientas de evaluación

Estas herramientas nos permitirán recoger datos durante todo este proceso. Las herramientas que utilizaremos son:

- **Ficha de seguimiento:** Basada en la observación directa del alumnado y las actividades, en ella recogeremos todo lo relacionado con la actitud de los alumnos durante las actividades y su progreso en las mismas. La ficha de actividad que utilizaremos es la recogida en el **anexo 3**.
- **Cuestionarios:** Éstos pueden de estilo clásico (en papel), o estar implementados con *Kahoot* [29], *Quizizz* [30], o cualquier otra herramienta de gamificación que creamos conveniente, constituyendo otro elemento indicador del nivel de conocimientos adquirido por los alumnos, ya que nos permiten, entre otras cosas, abarcar todos los aspectos de un tema o una actividad [33]. Sirva como ejemplo el cuestionario incluido en la actividad 0 (ver **anexo 2**).
- **Rúbricas:** El uso de este instrumento de evaluación proporciona ventajas tanto a los estudiantes como al profesorado [34]. Las rúbricas se utilizarán en aquellas actividades que consideremos oportunas y estarán ajustadas a la actividad que queremos evaluar. Las rúbricas están incluidas en la ficha de descripción de las actividades 3, 6.7 y 9.
- **Informes de actividad:** Estos informes contendrán una serie de apartados obligatorios, como son: el título del trabajo, nombre de la institución educativa, asignatura, profesor a quien está dirigido el informe, nombre del alumno o grupo de alumnos, grado escolar, objetivo, introducción, desarrollo, conclusiones y fuentes de información utilizadas. El modelo de informes está incluido en el **anexo 4** que servirá para la valoración de las actividades 10 y 11.

4.4.3- Evaluación de la unidad y de la labor docente

Dentro de función docente, es importante valorar y revisar tanto las diversas actividades propuestas como la labor del profesor en las mismas. Esta valoración por parte del alumnado nos aportará ideas, recursos y feedback sobre todos los aspectos relacionados con el aula y nos permitirá reflexionar sobre las actividades, si son las adecuadas o hay que cambiar alguna, si los resultados han sido los esperados o no, etc.

Para realizar la evaluación de las actividades pasaremos a los alumnos el cuestionario incluido en el **anexo 5**. Este cuestionario nos permitirá entre otros aspectos, conocer la impresión de los alumnos sobre las actividades y detectar algunos problemas que hubieran podido surgir en el desarrollo de las mismas, y deberá ser cumplimentado de una forma anónima ya que esto supone una de las mejores opciones para recoger respuestas objetivas por parte del alumnado.

En cuanto a la labor docente, pasaremos otro cuestionario, para que sea contestado de forma anónima por los alumnos. Este cuestionario está incluido en el **anexo 6**, y nos servirá para poder evaluar nuestra propia labor docente y corregir o cambiar aquellos aspectos que sean necesarios.

Capítulo 5 - Conclusiones

Este trabajo fin de máster propone una serie de actividades de aula en la asignatura de Tecnología en 4º ESO, concretamente dentro del bloque “Control y Robótica”, mediante la plataforma Arduino y un entorno de programación mBlock.

A través del desarrollo de estas actividades, hemos utilizado Arduino como una tecnología para construir pequeños robots autónomos que pueden realizar distintas funciones, dependiendo de los sensores, controladores, actuadores y programaciones que se utilicen en ellos. La variedad de acciones que pueden llevar a cabo depende, en gran medida, de nuestra imaginación y creatividad a la hora de diseñarlos y construirlos.

Llegados a este punto, es importante destacar que las actividades están diseñadas y secuenciadas de menor a mayor grado de dificultad, lo que ofrece al alumno un aprendizaje gradual.

Hemos utilizado el entorno de programación de robot mBlock, un entorno sencillo e intuitivo, fácil de interpretar y utilizar, lo que lo convierte en adecuado para la programación de estos robots.

Por otra parte, con el desarrollo de estas actividades hemos acercado la robótica a nuestros alumnos. La robótica ha pasado de ser una tecnología lejana y, de alguna manera, “sólo para locos y frikis”, a ser cercana a nosotros, de tal manera que hemos construido nuestros propios robots.

Todo esto ha supuesto la consecución de los objetivos que nos hemos propuesto a la hora de realizar este trabajo, de entre los que podemos destacar:

- Analizar Arduino como tecnología para impartir robótica dentro de 4º ESO.
- Analizar el uso de mBlock como lenguaje de programación para entornos Arduino.
- Proponer actividades con Arduino bajo mBlock que potencien la imaginación y la creatividad.
- Proponer mecanismos de evaluación del aprendizaje y de la propia actividad.
- Evidenciar la importancia de la robótica en la sociedad actual y acercarla hasta nosotros.

Otros objetivos que también hemos cumplido con este trabajo son los dispuestos en la orden EDU-362 [10] y el Real Decreto 1105 [12] dentro del bloque de “Control y Robótica” en 4º ESO.

No obstante, a pesar de no haber puesto en práctica este proyecto, y no haber recibido ese feedback, sí hemos diseñado una serie de herramientas para valorar los objetivos propuestos en él a través de las actividades que hemos realizado. Herramientas como *Fichas de Seguimiento*, *Informes de Actividad* y *Cuestionarios*, sencillas y fáciles de aplicar, que proporcionarán al profesor una idea clara del progreso de los alumnos.

Una última reflexión final es la posibilidad de dar continuidad a este trabajo, en la medida que éste pueda servir como base para el desarrollo de nuevas y futuras actividades para los alumnos: actividades que incluyan el uso de nuevos sensores y controladores, actividades dirigidas a la atención a la diversidad, etc.

Capítulo 6 - Bibliografía y Webgrafía

Bibliografía (ordenada por orden de aparición)

- [10] Orden EDU-362-2015 de 4 de mayo currículo ESO Castilla y León.
- [11] Ley Orgánica 8/2013, de 9 de diciembre, para la mejora de la calidad educativa (LOMCE).
- [12] Real Decreto 1105/2014, de 26 de diciembre, por el que se establece el currículo básico de la Educación Secundaria Obligatoria y del Bachillerato.

Webgrafía (ordenada por orden de aparición)

- [1] Página oficial de Arduino: <https://www.arduino.cc/>.
- [2] Página de Wikipedia: <https://es.wikipedia.org/>.
- [3] Blog de Luis Araya 2017: <https://blogluisaraya.wordpress.com/>. Última consulta mayo 2018.
- [4] Página oficial Playground de Arduino: <http://www.playgrondarduino.cc/>.
- [5] Página oficial Arduino Uno: <http://arduino.cl/arduino-uno>.
- [6] Página oficial de mBlock: <http://www.mblock.cc/>.
- [7] Página oficial de Makeblock: <https://www.makeblock.es>.
- [8] Curso de robótica infantil con mBlock de PROMETEC: <https://www.prometec.net/introduccion-mblock/>.
- [9] Página de educación innovadora: <http://www.innovandoeducacion.es/steam-educacion-innovadora/>.
- [13] Página oficial de GreenMonkey: <http://www.thegreenmonkey.es/blog/5-razones-para-aprender-robotica/>
- [14] Blog de Luis del Valle: <https://campus.programarfacil.com/?menu-curso-arduino>. Última entrada mayo 2018.
- [15] Bravo Sánchez, F. A. y Forero Guzmán, A. (2012). La robótica como un recurso para facilitar el aprendizaje y desarrollo de competencias generales. Revista Teoría de la Educación: Educación y Cultura en la Sociedad de la Información. 13(2), 120-136.
- [16] Blog de Luis Llamas: <https://www.luisllamas.es/leer-un-pulsador-con-arduino/>. Última entrada mayo 2018.
- [17] Página de Arduino stackexchange: <http://arduino.stackexchange.com/>.
- [18] Página de Arduino stackoverflow: <http://stackoverflow.com/>.
- [20] Página oficial hackster: <https://www.hackster.io/>.

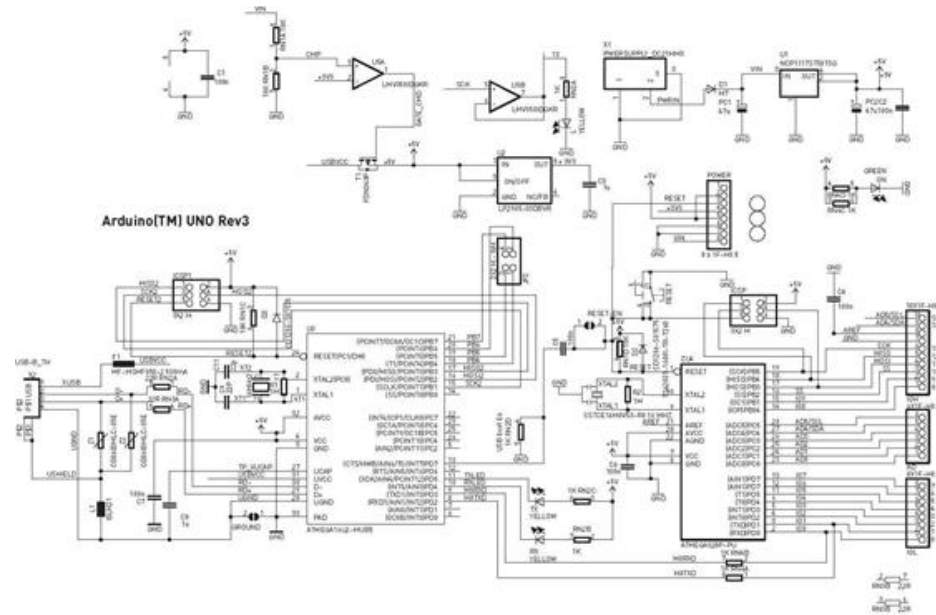
- [21] Página oficial OpenHardware: <https://www.openhardware.io/>.
- [22] Tutorial básico de programación de Arduino con mBlock.
<https://www.youtube.com/watch?v=FiWX9Gpr9KQ&feature=youtu.be>
- [23] Tutorial de programación de Arduino con mBlock.
<https://www.youtube.com/watch?v=FiWX9Gpr9KQ&feature=youtu.be>
- [24] Página oficial de Github. Extensiones desarrolladas por usuarios para utilizar distintos Shields con mBlock.
https://github.com/iFernandoSousa/mBlock_motor_L298N.
- [25] Página web Programarfácil. Tutorial de programación servos con arduino.
<https://programarfácil.com/tutoriales/fragmentos/servomotor-con-arduino/>.
- [26] Página oficial de minitrónica. Uso del sensor PIRE HC-SR501.
<https://www.minitronica.com>.
- [27] Características LDR. Datasheet.
<https://www.electan.com/datasheets/cebek/CE-C2795.pdf>
- [28] Liga nacional de robótica de competición. Página oficial. <http://lnrc.es/>
- [29] Página oficial de Kahhot. <https://kahoot.com>.
- [30] Página oficial de Quizizz. <https://quizizz.com>.
- [31] Página oficial de Tinkercad. <https://www.tinkercad.com>
- [32] Página oficial de Thingiverse. <https://www.thingiverse.com>
- [33] Ventajas y desventajas de los cuestionarios. <https://surveyanyplace.com/es/9-ventajas-y-desventajas-de-los-cuestionarios/>
- [34] Ventajas y desventajas de las rúbricas.
<https://www.realinfluencers.es/2016/03/10/rubricas/>
- [35] Funcionamiento sensor HC-SR04.
<http://bkargado.blogspot.com/2013/09/todosobrehc-sr04.html>.
- [36] Tutorial LCD 2 x16 para Arduino.
<https://www.geekfactory.mx/tutoriales/tutoriales-arduino/lcd-16x2-por-i2c-con-arduino/>.
- [37] Google Play store. Tienda de juegos y apps de Google. <https://play.google.com>.
- [38] Página de Google Play. App Arduino Control Car V2. El profe García.
https://play.google.com/store/apps/details?id=appinventor.ai_el_profe_garcia.Arduino_Control_Car_V2&hl=es.

Capítulo 7 - Anexos

7.1- Anexo 1

Esquema de la placa electrónica

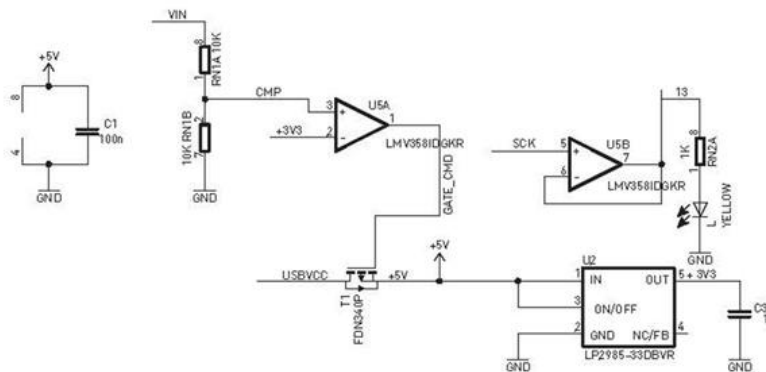
El siguiente esquema de la figura 1 representa toda la electrónica de la placa Arduino UNO R3.



Arduino UNO R3 - Electrónica Completa

Figura 1

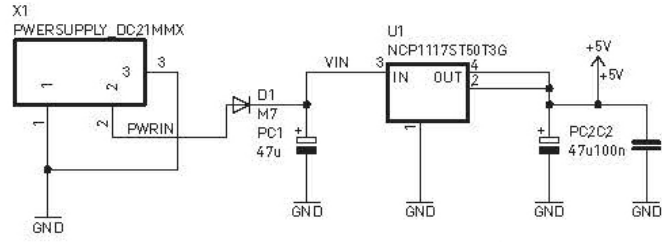
En el siguiente esquema, representado en la figura 2, vemos una parte del anterior esquema electrónico, y representa la dirección de voltaje para alimentar la placa. En él, podemos ver representado "USBVCC" para la alimentación por el puerto USB.



Arduino UNO R3 - Esquema alimentación USB

Figura 2

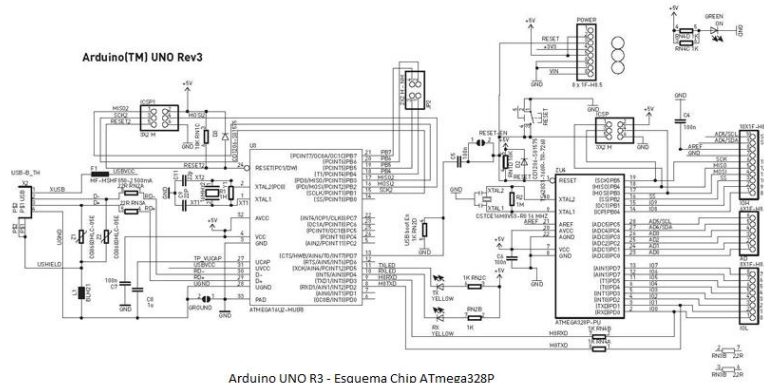
El siguiente esquema de la figura 3, también extraído del primer esquema electrónico de arriba, representa la entrada principal de tensión a través del jack de alimentación.



Arduino UNO R3 - Esquema Alimentación por Jack

Figura 3

Por último, en la figura 4, tenemos representado el esquema donde se procesa toda la información. El chip principal para procesar la información es el ATmega328P.



Arduino UNO R3 - Esquema Chip ATmega328P

Figura 4

7.2- Anexo 2

Nombre del alumno:

Fecha:

Asignatura:

Contesta las siguientes preguntas sabiendo que puede haber 1 ó 2 respuestas válidas y, que cada pregunta mal contestada resta 0,25 puntos.

1. En una placa Arduino Uno:
 - a. Tenemos 13 pines analógicos y 6 digitales.
 - b. Tenemos 13 pines digitales, de los cuales 6 se pueden usar como pines PWM y 6 pines analógicos.
 - c. Tenemos 13 pines digitales que también se pueden usar como pines PWM y 6 digitales.
2. Los pines analógicos de una placa Arduino Uno:
 - a. Se pueden utilizar como entradas y como salidas.
 - b. Sólo se pueden utilizar como entradas.
 - c. Sólo se pueden utilizar como salidas.
3. Los pines digitales de una placa Arduino Uno:
 - a. Se pueden utilizar como entradas y como salidas.
 - b. Sólo se pueden utilizar como entradas.
 - c. Sólo se pueden utilizar como salidas.
4. Los pines analógicos:
 - a. Se caracterizan por leer valores de tensión de 0 a 5 Voltios con una resolución de 10 bits (1024 valores diferentes).
 - b. Se caracterizan por leer valores de tensión de 0 a 5 Voltios con una resolución de 8 bits (256 valores diferentes).
 - c. Se caracterizan por leer valores de tensión de 0 a 10 Voltios con una resolución de 8 bits (256 valores diferentes).
5. ¿Para qué se utilizan los pines PWM?:
 - a. Para emular salidas digitales con una resolución de 8 bits (256 valores diferentes).
 - b. Para emular salidas analógicas con una resolución de 8 bits (256 valores diferentes).
 - c. Las dos anteriores son falsas.
6. El botón **Reset** de Arduino Uno:
 - a. No sirve para nada.
 - b. Se utiliza para resetear el programa grabado en la placa en caso de bloqueos.
 - c. Se utiliza para resetear el programa grabado en la placa y así poder grabar otro programa diferente.
7. Las instrucciones contenidas en una sentencia **Si <condición> Entonces**:
 - a. Se ejecutarán siempre.
 - b. Se ejecutarán si se cumple la condición.
 - c. Se ejecutarán si no se cumple la condición.
8. Las instrucciones contenidas en una sentencia **Repetir Hasta que <condición>**:
 - a. Se ejecutarán siempre.
 - b. Se ejecutarán hasta se cumpla la condición.
 - c. Se ejecutarán mientras no se cumpla la condición.

9. Las instrucciones contenidas en una sentencia ***Repetir <nº veces>***:
- Se ejecutarán siempre 10 veces.
 - Se ejecutarán tantas veces como indique *nº veces*.
 - Se ejecutarán mientras no se cumpla el *nº veces*.
10. Con mBlock podemos crear:
- Variables.
 - Variables y bloques.
 - Variables datos y bloques.

Fuente: propia

7.3- Anexo 3

FICHA DE SEGUIMIENTO																		
ACTIVIDADES CON ARDUINO																		
Actividad	Consecución de objetivos											Participación y actitud	Cuidado del material	Trabajo en grupo	Valoración global			
	0	1	2	3	4	5	6	7	8	9	10					11		
Alumno 1																		
Alumno 2																		
Alumno 3																		
Alumno 4...																		

Fuente: Propia

7.4- Anexo 4

Informe de actividad

Título del trabajo:		
Asignatura:	Curso:	Fecha:
Componentes del grupo de trabajo:		
Objetivos:		
Materiales utilizados:		
Pasos seguidos en el proceso de realización de la actividad:		
Valoración del resultado:		
Dificultades encontradas:		

Funciones de cada participante:

Fuente: Propia

7.5- Anexo 5

Con este cuestionario se pretende valorar las actividades realizadas con Arduino y mBlock dentro del bloque 4 “Control y robótica”. También te pedimos que hagas un esfuerzo por razonar tus respuestas.

CUESTIONARIO ACTIVIDADES CON ARDUINO	
PREGUNTAS	RESPUESTAS
Ponle nota a las actividades (del 1 al 10)	
¿Qué actividad te ha gustado más?, ¿por qué?	
¿Qué actividad te ha gustado menos?, ¿por qué?	
¿Qué actividad te ha resultado más difícil?, ¿por qué?	
¿Consideras cambiar alguna actividad?, ¿cuál?	
¿Te ha gustado programar con mBlock?, ¿por qué?	
¿Te ha gustado utilizar Arduino?, ¿por qué?	

Fuente: Propia

7.6- Anexo 6

CUESTIONARIO LABOR DOCENTE	
PREGUNTAS	RESPUESTAS
¿Crees que el profesor ha explicado de forma clara Arduino?, ¿por qué?	
¿Crees que el profesor ha explicado de forma clara mBlock?, ¿por qué?	
¿Consideras suficiente la ayuda prestada por el profesor durante las actividades?, ¿por qué?	
¿Qué actividad te ha resultado más difícil?, ¿por qué?	
¿Cambiarías algo en las actividades?, ¿por qué?	
¿Ponle nota al profe! (del 1 al 10)	

Fuente: Propia