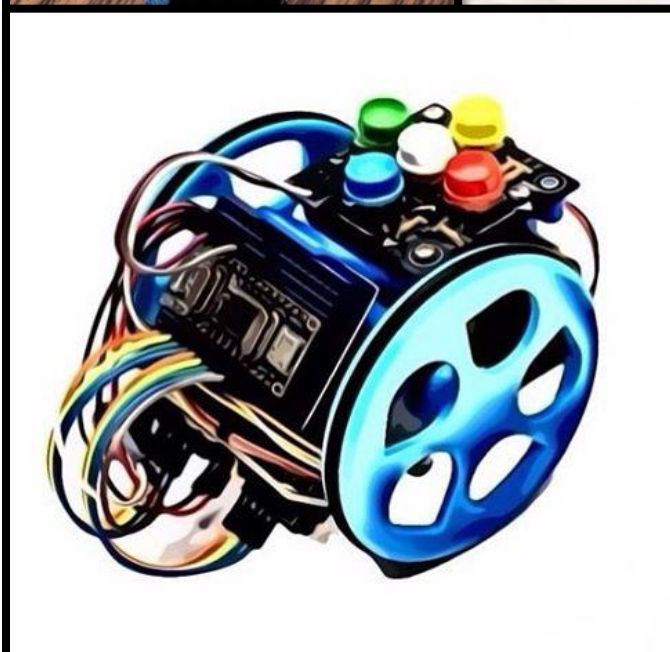
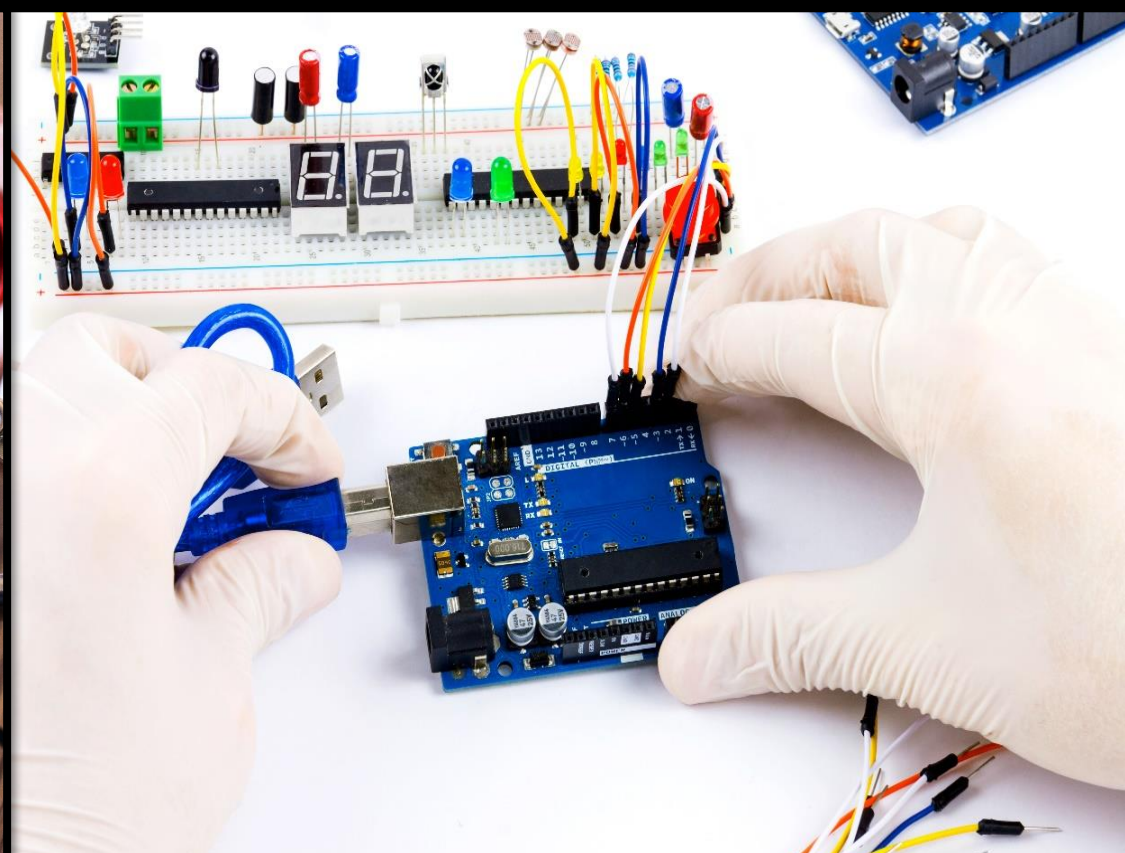


Vehículos No Tripulados



Índice

1. Datos generales	1
2. Introducción	1
3. Descripción de la actividad	3
3.1. El proyecto Escornabot	3
3.2. Tapetes de juego	6
4. Teoría de componentes y técnicas de laboratorio	7
4.1. El zumbador	7
4.2. El motor paso a paso	9
4.3. El multímetro	11
4.4. Técnicas de soldadura	12
5. Proyecto Escornabot	14
5.1. Páginas web de referencia	14
5.2. Lista de materiales	15
5.3. PCB Botonera Escornabot	17
5.4. Esquema electrónico botonera	19
5.5. Comprobación de la botonera	20
5.6. Ajustes de la botonera	21
5.7. Memoria del Escornabot	23
5.8. Programación de los motores paso a paso	24
5.9. Programación final	25
6. Makeblock	25
6.1. Conceptos de repaso	27
6.1.1. Sensor de Ultrasonido	27
6.1.1.1. La ecolocalización	27
6.1.1.2. Ultrasonidos	28
6.1.1.3. Sensor de proximidad HC-SR04	29
6.1.1.4. Algoritmo y funcionamiento del sensor	30
6.2. Programación del mBot desde Arduino IDE	31
6.3. ACTIVIDAD 1 MBOT: BUZZER TEST	32
6.4. ACTIVIDAD 2: SENSOR DE LUMINOSIDAD	32
6.5. ACTIVIDAD 3: SENSOR DE ULTRASONIDOS	32

6.6.	ACTIVIDAD 4: ROBOT AUTÓNOMO CON SENSOR ULTRAOSNIDOS	32
6.7.	ACTIVIDAD 5: AÑADIENDO EXTRAS A NUESTRO ROBOT	32

1. Datos generales

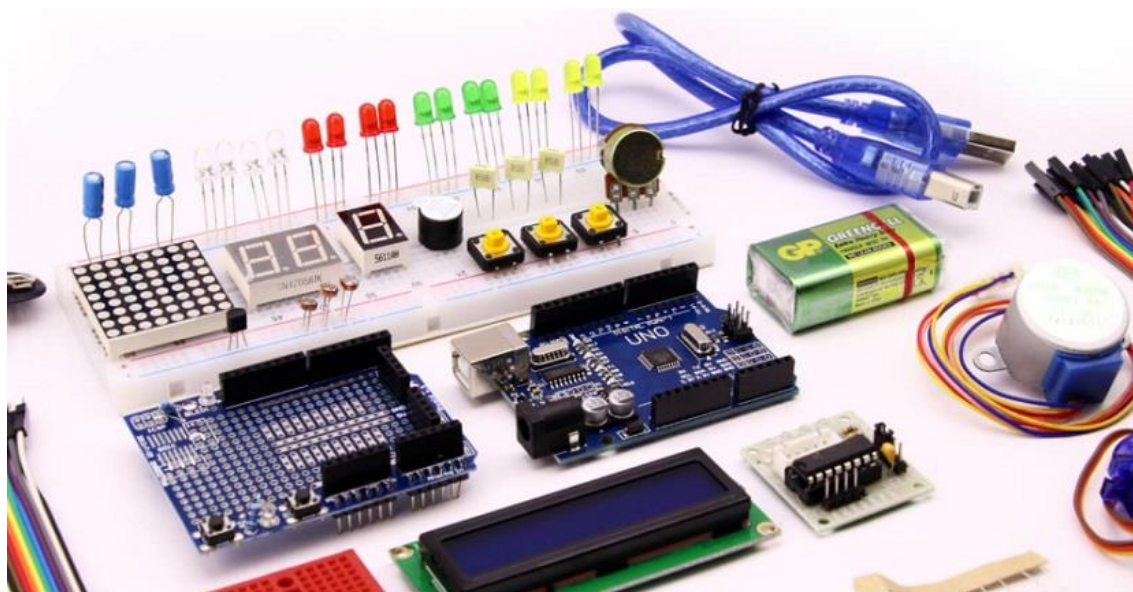
- Título: VEHÍCULOS NO TRIPULADOS
- Organiza: Asociación Educativa y Tecnológica Valle_Seam
- Fecha: 17/12/2021
- Horas lectivas: 3
- Plazas: 20
- Docente: Augusto Samuel Hernández Martín

2. Introducción

La tecnología, y más concretamente la electrónica, ha experimentado una evolución extraordinaria a lo largo de las últimas décadas. Al igual que podía ocurrir con otras muchas áreas de conocimiento de complejidad considerable, el desarrollo de circuitos electrónicos y programación de microcontroladores era una tarea que normalmente solo estaba al alcance de profesionales con una alta cualificación, que por lo general trabajaban como ingenieros en empresas del sector industrial o como investigadores en universidades, organismos en definitiva con capacidad económica para financiar los altos costes de investigación y de fabricación de productos.



Sin embargo, la electrónica al igual que el resto de áreas tecnológicas ha evolucionado favorablemente en los últimos años con la aparición de diversas placas de desarrollo de propósito general muy sencillas de programar, juntamente con la electrónica modular, con lo que ahora disponemos de miles de módulos electrónicos compatibles, fáciles de usar y de conectar unos con otros para formar circuitos complejos y productos terminados.



Las nuevas tecnologías por tanto han permitido que lo que antes era muy complejo, ahora sea totalmente accesible a todas las personas, sin importar la edad, el conocimiento previo o la experiencia. Gracias a todo esto han surgido grandes y variadas comunidades de aficionados al desarrollo de proyectos y fabricación de productos, caseros y no tan caseros, que han ido creciendo de forma exponencial gracias a la inmensa cantidad de proyectos y conocimientos compartidos a modo de repositorios, tutoriales y contenidos de valor, creándose toda una Cultura Maker en torno a proyectos de electrónica, programación, robótica y fabricación digital, caracterizada por el afán de crear, inventar, aprender y compartir.

Si bien existen referencias anteriores a la Cultura Maker que formaron parte de su origen, tales como el movimiento DIY (Do It Yourself) de los años 70, el mérito de la aparición de la Cultura Maker como tal se le reconoce mayormente a la revista Make que se publica bimensualmente desde el año 2005, así como a su fundador Dale Dougherty. Desde entonces la Cultura Maker se ha ido extendiendo a ritmo considerable, pasando de ser una simple comunidad de aficionados a toda una nueva forma de entender la industria, la investigación y la educación.

Ofreciendo un nuevo modelo de aprendizaje basado en la experiencia, el hacer y el desarrollo de proyectos, la Cultura Maker está teniendo una muy buena aceptación por parte de las universidades, colegios y centros de enseñanza en general, de tal manera que lo que antes era impensable encontrar fuera de las aulas de ingeniería, ahora poco a poco se está introduciendo en educación primaria y secundaria, viendo como alumnos de edades tempranas se vuelven capaces de desarrollar proyectos de electrónica muy diversos, tales como robots, reproductores de música, dispositivos para instalaciones de domótica, etc...



3. Descripción de la actividad

3.1. El proyecto Escornabot

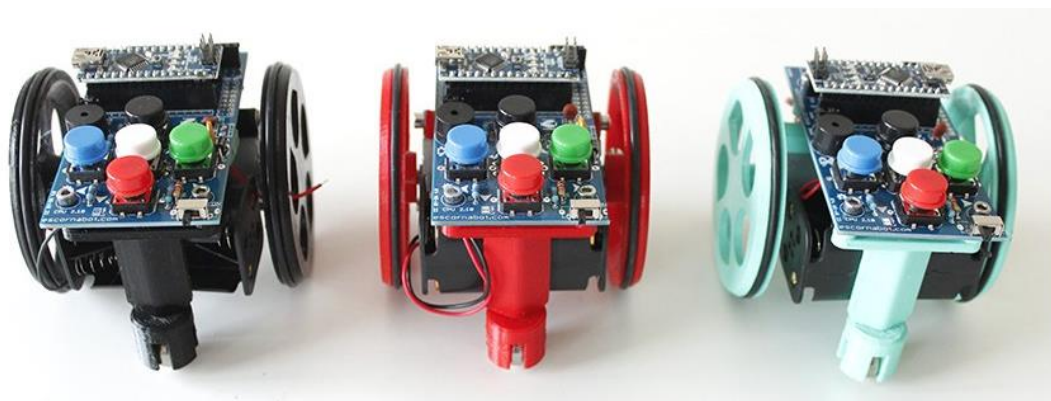
Escornabot es un proyecto de hardware y software libre nacido en Galicia que tiene el objetivo de acercar la robótica maker, la electrónica, la programación y la fabricación digital, a niños de todas las edades. Se puede visitar su web oficial a través del enlace:

<http://escornabot.com/web/>

Asimismo se puede acceder a toda la información y documentación del proyecto compartido (modelos 3D, programación, guías de montaje, etc...) a través del siguiente enlace:

<https://github.com/escornabot>

Existen varias versiones de robots dentro del proyecto Escornabot, y el más básico de la saga permite ser programado mediante botones para ejecutar una secuencia de movimientos con el objetivo de superar un reto o resolver un problema dentro de un tablero de juego.



Esta herramienta viene genial para que alumnos de infantil y primaria comiencen a familiarizarse con sus primeros robots, lo que les permitirá aprender a programar mediante lenguaje direccional, mejorar la orientación espacial y aplicar la lógica para resolver problemas sobre diversos juegos y disciplinas (lengua, inglés, matemáticas, etc...).

Este modo de trabajar de Escornabot nos recuerda mucho al clásico Bee-Bot, una simpática abeja robot que funciona de esta misma manera y que se utiliza mucho para trabajar con alumnos de infantil y primaria.



Sin embargo y a pesar de ser similares en cuanto a funcionamiento, Escornabot presenta claras ventajas respecto a Bee-Bot:

- **Proyecto Maker:** A diferencia de Bee-Bot, Escornabot no se puede comprar en una juguetería, debe fabricarse de cero por uno mismo. Para ello cualquiera puede acceder al repositorio del proyecto, descargar los modelos 3D para imprimir las piezas, realizar el montaje del circuito electrónico a partir de los esquemas y programar el robot para que realice su función correctamente.
- **Proyecto I+D:** Lo dicho anteriormente resulta ideal para transmitir al profesorado y el alumnado metodologías y buenos hábitos sobre desarrollo de proyectos I+D, dado que implica una labor de búsqueda, investigación, revisión de documentación, aparte del desarrollo innovador y creativo que cada uno desee aportar con su propio Escornabot.
- **Proyecto colaborativo:** Como ya hemos dicho, Escornabot es un proyecto compartido que pertenece a una gran comunidad en la que se fomenta no solo el compartir y extender lo que ya está hecho, sino participar y colaborar en su desarrollo para continuar mejorando el robot, por lo que resulta ideal para iniciar a profesores y alumnos en este tipo de proyectos y en el uso de repositorios online.



- **Proyecto para todos:** Escornabot, al igual que Bee-Bot, está pensado para ser utilizado como recurso educativo para niños y niñas de infantil y primaria. No obstante, su desarrollo, fabricación y programación, para nada trivial, es ideal para formar a alumnos de secundaria y bachillerato, por tanto al final no solo supone un mejor aprovechamiento de los recursos, sino que fomenta la creación de sinergias en torno a proyectos en los centros educativos.
- **Proyecto multidisciplinar:** Escornabot es un proyecto muy completo que implica diseño e impresión 3D, diseño de circuitos electrónicos, uso de instrumentos de medición, técnicas de soldadura y programación de microcontroladores, por lo que resulta ideal como curso para mostrar una visión amplia sobre las nuevas tecnologías disponibles para el prototipado de productos y fabricación de recursos educativos.

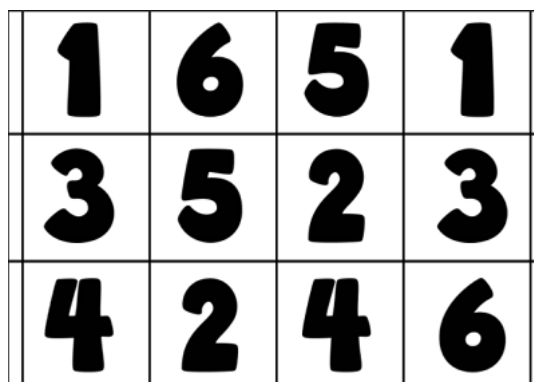
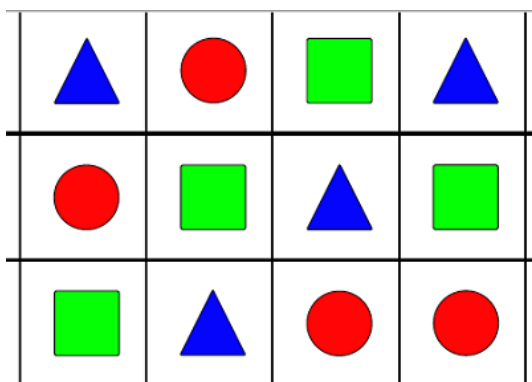
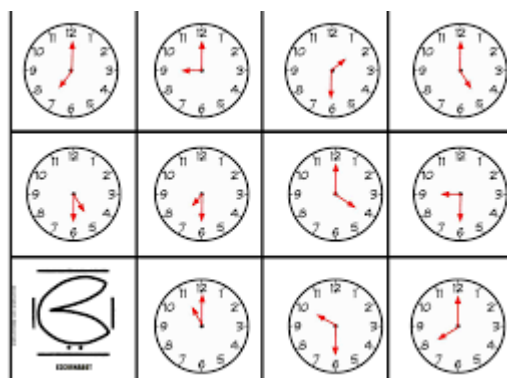
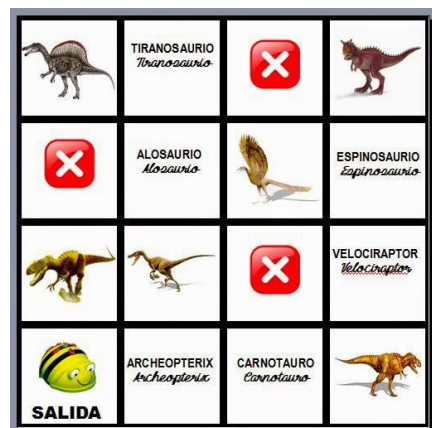


- **Proyecto escalable:** Si bien en este primer curso comenzaremos por desarrollar la versión más sencilla de Escornabot, el proyecto realmente está pensado para ser fácilmente escalable, de modo que el único límite está en tu imaginación. La ventaja de poder modificar a tu gusto el diseño 3D de las piezas, de añadir sensores y modificar o ampliar la programación, hace que las posibilidades sean infinitas.
- **Proyecto económico:** Por si todo lo anterior fuese poco, hay que añadir que construir un Escornabot es mucho más barato que comprar un Bee-Bot, cuyo precio en jugueterías es de 100€, mientras que el Escornabot supone un coste en materiales de aproximadamente 25 €.

3.2. Tapetes de juego

Los tapetes de juego también pueden ser fabricados en el aula por los propios alumnos y profesores. Las posibles temáticas son muy variadas, y por lo general se busca que el juego con Escornabot sea en sí mismo un recurso educativo para los alumnos.

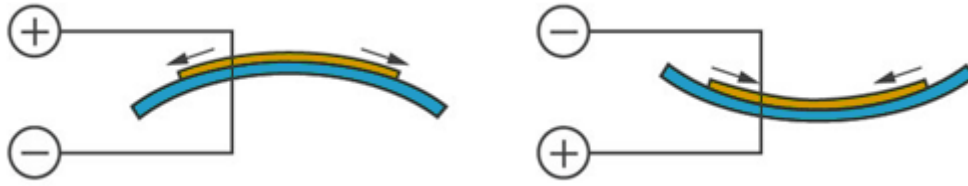
Algunos ejemplos de tapetes pueden ser los siguientes:



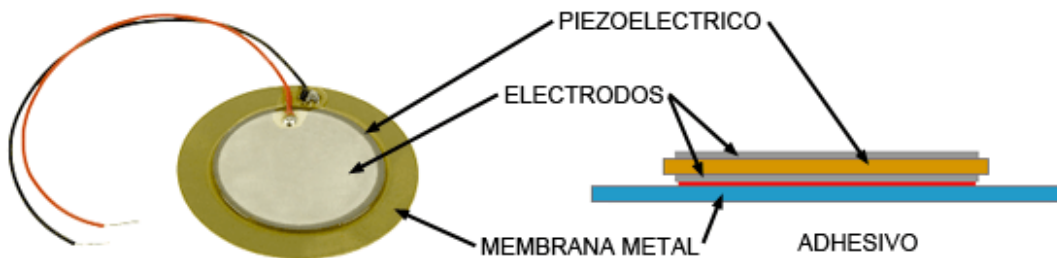
4. Teoría de componentes y técnicas de laboratorio

4.1. El zumbador

El zumbador es un componente sonoro que basa su principio de funcionamiento en la pizeoelectricidad, un fenómeno que sucede con determinados cristales como el cuarzo, que al ser sometidos a una tensión mecánica generan una tensión eléctrica, y también lo contrario, cuando son sometidos a una tensión eléctrica se deforman variando su volumen, volviendo a su posición de reposo cuando se les deja de aplicar tensión.



Si sometemos un material piezoeléctrico a una tensión eléctrica variable se produce una vibración de igual frecuencia que la señal eléctrica. El zumbador aprovecha este fenómeno para hacer vibrar una membrana de metal a la que va adherido el cristal. Si la frecuencia de vibración está dentro de la frecuencia audible (20 Hz a 20 KHz aproximadamente) se produce una onda sonora a la misma frecuencia.



Los zumbadores tienen la ventaja de ser componentes muy baratos y fáciles de manejar, por lo que son ideales para proyectos que requieran de sonido o incluso melodías mediante tonos simples, como por ejemplo alarmas o tonos para pulsaciones. Sin embargo, los zumbadores no son válidos para reproducir pistas musicales, no porque no sean capaces de hacerlo sino porque la calidad del sonido es muy mala, por lo que para este tipo de aplicaciones lo más.

Podemos distinguir principalmente entre dos tipos de zumbadores:

- **Zumbador pasivo**: Sin oscilador interno, lo que implica que debe alimentarse con corriente alterna para poder funcionar. Tienen la desventaja de ser un poco más difíciles de utilizar al tener que ser alimentados directamente a través de una señal eléctrica de frecuencia concreta para producir el tono deseado. Sin embargo, existen librerías en Arduino que nos facilitan mucho el control de zumbadores pasivos. Tienen la gran ventaja de que podemos generar melodías variando la frecuencia de la señal. Se suele distinguir fácilmente por no tener generalmente pegatina en la parte superior, y por tener terminales de igual longitud.
- **Zumbador activo**: Dispone de un oscilador interno, lo que implica que funciona con corriente continua, por lo que resulta más sencillo de utilizar que el zumbador pasivo, ya que simplemente se controla con una señal digital en alta o baja para producir o no sonido. La desventaja es que no puede generar melodías, ya que no se puede variar la frecuencia de la señal para producir diferentes tonos. Se distingue

fácilmente porque suele tener pegatina en la parte superior, además de tener el terminal positivo más largo que el negativo.

Una forma útil de distinguir un zumbador activo de un zumbador pasivo, es alimentándolo a con corriente continua (generalmente 5V). Si el zumbador suena es activo, de lo contrario es pasivo.



Zumbador pasivo



Zumbador activo

4.2. El motor paso a paso

Un motor paso a paso es un tipo de motor de corriente continua, con la importante característica de que nos permite controlar el movimiento rotativo con precisión, gracias a que el estator dispone de varias bobinas que permiten que el rotor gire un ángulo determinado o paso. El motor paso a paso se alimenta a través de una placa electrónica o driver, que energiza las bobinas del estator siguiendo un orden o secuencia determinada, para que de esa forma el rotor gire de forma controlada, avanzando solo el número de pasos que le ha sido indicado.



Una de las características más importantes de un motor paso a paso es su precisión que se mide en número de pasos por vuelta, lo cual depende directamente del número de bobinas. En principio un motor de por ejemplo cuatro bobinas tendría cuatro pasos por vuelta, de ocho bobinas ocho pasos, etc... lo cual en cualquiera de los casos es muy poco para lograr un motor de una precisión mínima aceptable. Debido al considerable tamaño de las bobinas no es posible alcanzar un buen número de pasos aumentando solo el número de las bobinas, y es por eso que a los motores paso a paso se les suele añadir una reductora.

Un motor paso a paso de 4 bobinas con una reductora de por ejemplo 1/64 tiene un total de 256 pasos por vuelta. Este es el caso del motor paso a paso que nosotros vamos a utilizar, el **28BYJ-48**, un motor unipolar de 5 hilos.



Con el fin de ganar el doble de pasos y conseguir un movimiento más suave y preciso, en lugar de excitar las bobinas para que el motor avance paso a paso, vamos a emplear un trquito para trabajar mediante medios pasos, lo que nos daría un total de 512 pasos por vuelta.

Para controlar la secuencia de giro utilizaremos el Arduino conectado a los pines (IN1 – IN4) de la placa controladora del motor. La secuencia de control de giro sería la siguiente:

Paso	Bobina A (IN1)	Bobina B (IN2)	Bobina C (IN3)	Bobina D (IN4)	Imagen
1	ON	OFF	OFF	OFF	

2	ON	ON	OFF	OFF	
3	OFF	ON	OFF	OFF	
4	OFF	ON	ON	OFF	
5	OFF	OFF	ON	OFF	
6	OFF	OFF	ON	ON	
7	OFF	OFF	OFF	ON	
8	ON	OFF	OFF	ON	

4.3. El multímetro

Un **multímetro** (también conocido como polímetro o tester) es un instrumento de medida que sirve para tomar mediciones de diferentes tipos de magnitudes eléctricas. Dispone de un selector para escoger la escala y tipo de medida que se desea tomar, y en función de su posición, el multímetro puede trabajar generalmente de cualquiera de las siguientes maneras:

- **Voltímetro:** Para medir tensiones (Voltios)
- **Amperímetro:** Para medir corrientes (Amperios)
- **Óhmetro:** Para medir resistencias (ohmios)



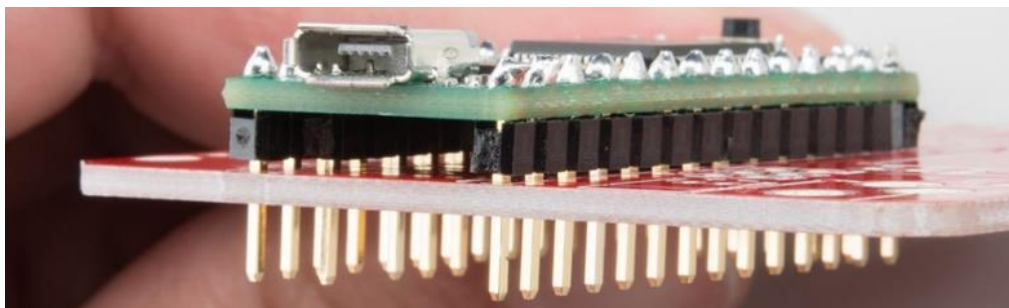
Para conectar las puntas de medida el multímetro dispone de varios conectores. Al común de color negro siempre deberá ir conectado una de las puntas (normalmente el de color negro para no hacer las cosas al revés de la lógica). La punta roja la conectaremos a uno de los conectores rojos disponibles en función de la magnitud que necesitemos medir. Por lo general hay uno o dos conectores para medir amperios con distinto límite máximo, y otro conector para medir tensión y ohmios entre otras magnitudes.

4.4. Técnicas de soldadura

Existen muchas técnicas y herramientas para soldadura de componentes electrónicos, pero a modo de introducción cabe destacar dos categorías en particular correspondientes a los dos grandes grupos de componentes en función de su estructura:

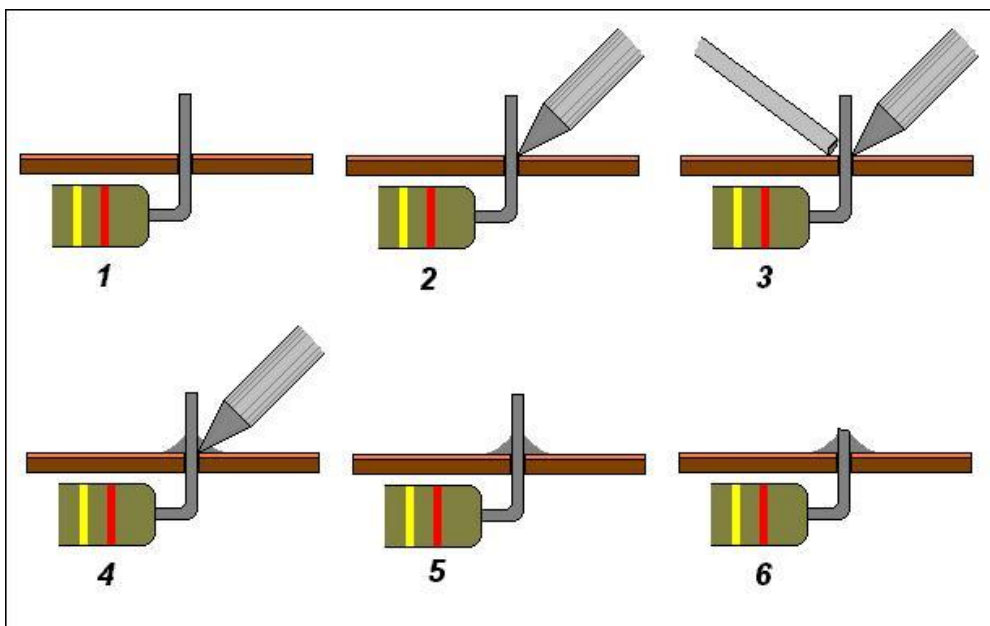
- THD (Through Hole Device):

THD (Through Hole Device) o Dispositivos de Agujero Pasante: son aquellos componentes cuyos pines de conexión atraviesan la placa de circuito impreso (PCB).



La técnica de soldadura utilizada para estos casos es la más común, y los pasos correctos son: apoyar la punta del soldador sobre la pata del componente y el pad de la placa donde se desea soldar, aplicar estaño con cuidado, retirar el estaño manteniendo la punta del soldador, retirar la punta del soldador. Una soldadura correcta tiene forma de

cono y ocupa la mayor superficie posible del pad, pero sin llegar a convertirse en una bola. Véase la siguiente imagen:

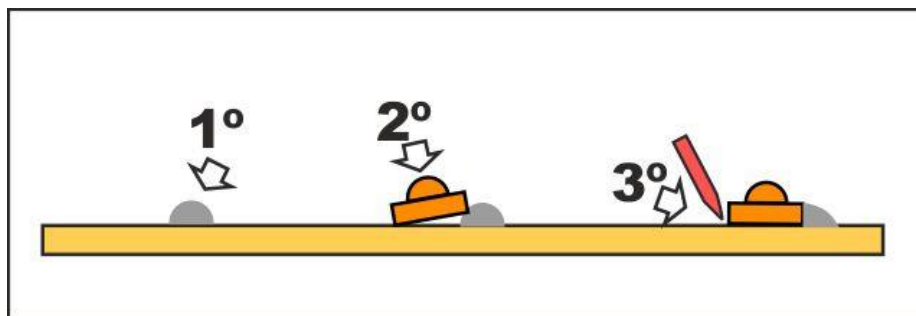


- SMD (Surface Mount Device):

SMD (Surface Mount Device) o Dispositivos de Montaje Superficial: son aquellos componentes cuyos pines de conexión apoyan sobre la superficie de la placa de circuito impreso (PCB).



En estos casos las técnicas pueden ser más variadas en función de si empleamos un soldador común tipo “cautin”, estación de aire caliente, pinza soldadora, bobina de estaño o pasta para soldar, pero en cualquier caso la apariencia final de una buena soldadura SMD debe ser la siguiente:



5. Proyecto Escornabot

5.1. Páginas web de referencia

- Página web oficial:

<http://escornabot.com/web/>

La web oficial de Escornabot. Útil para conocer el origen del proyecto, su evolución y contexto general. Incluye tutoriales de montaje e información técnica de interés para las distintas versiones de Escornabot, aunque para esto último mejor tener un poco de cuidado dado que la página se encuentra un tanto abandonada y desactualizada. Para tutoriales y descarga de archivos se recomienda recurrir a otras fuentes.

- Repositorio del proyecto:

<https://github.com/escornabot>

Escornabot cuenta con su propio repositorio en Github. Desde ahí podrás ver y descargar toda la documentación técnica del proyecto (diseño 3D, códigos Arduino, diseño PCB, esquemas electrónicos, documentos explicativos, etc...). Aunque se mantiene actualizado, desde mi punto de vista la información no está lo suficientemente bien organizada para que el repositorio sea intuitivo. Puede resultar un poco complicado de interpretar sobre todo para personas poco familiarizadas con Github.

- Página web de Pablo Rubio:

<https://pablorubma.cc/>

Aunque Pablo Rubio no pertenece directamente al grupo de personas que desarrollaron el robot Escornabot, su contribución en el proyecto se considera igualmente importante. Tal y como él se define a sí mismo, es un maker de corazón y gran fan de Escornabot. La razón por la que su web es tan útil es que reúne, ordena y mantiene actualizada casi toda la información relativa al proyecto. Sin duda una de las páginas más recomendables para conocer Escornabot y seguir tutoriales.



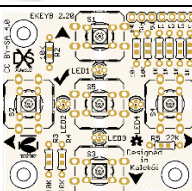




- Gitbook de Escornabot:

<https://catedu.gitbooks.io/escornabots/>

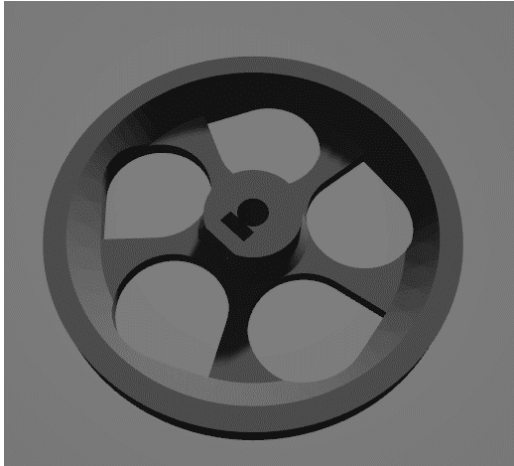
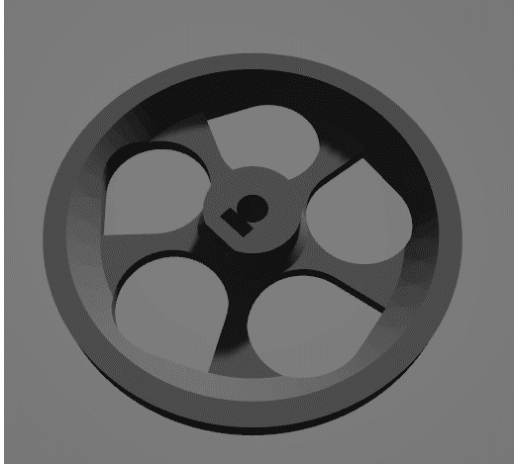
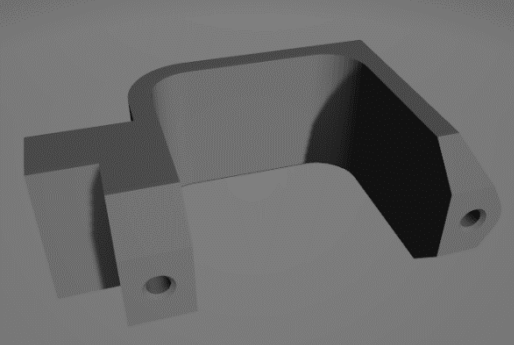
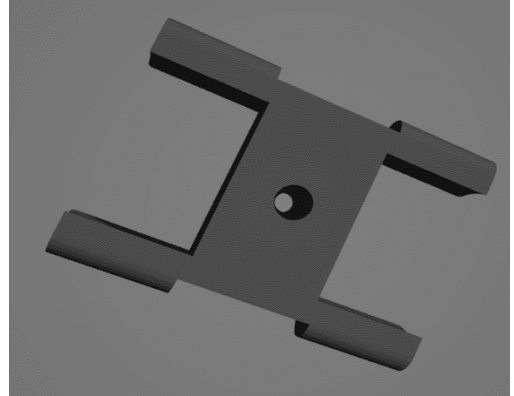
Escornabot dispone también de un GitBook bastante útil que resume y organiza muy bien la información, documentación y tutoriales del proyecto. Muy recomendable para echarle un vistazo en paralelo con la página web de Pablo Rubio.

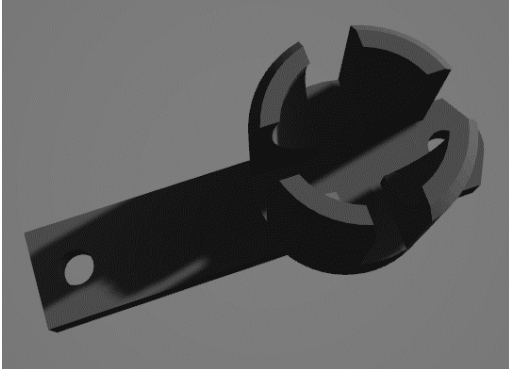
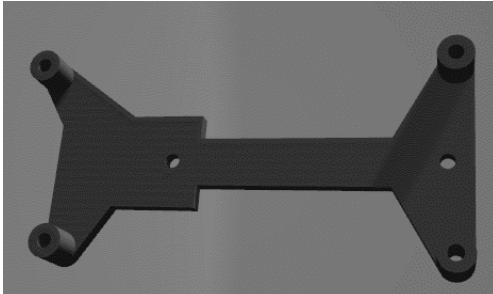
5.2. Lista de materiales

En la siguiente lista se muestra el total de componentes necesarios para fabricar un Escornabot versión DIY:

Imagen	Ud.	Descripción
	1	Arduino Nano
	2	Motor paso a paso 28BYJ-48 + Driver ULN2003
	1	PCB Botonera + Driver 2.20 Singularis con LEDs + componentes
	1	Portapilas 4 X AA
	2	Junta tórica 63 x 3 mm
 Ø 14 mm	1	Bola de acero 14 mm
	1	Kit de tuercas y tornillos: - 16 X Tornillo M3 x 10 mm - 2 X Tuerca M3

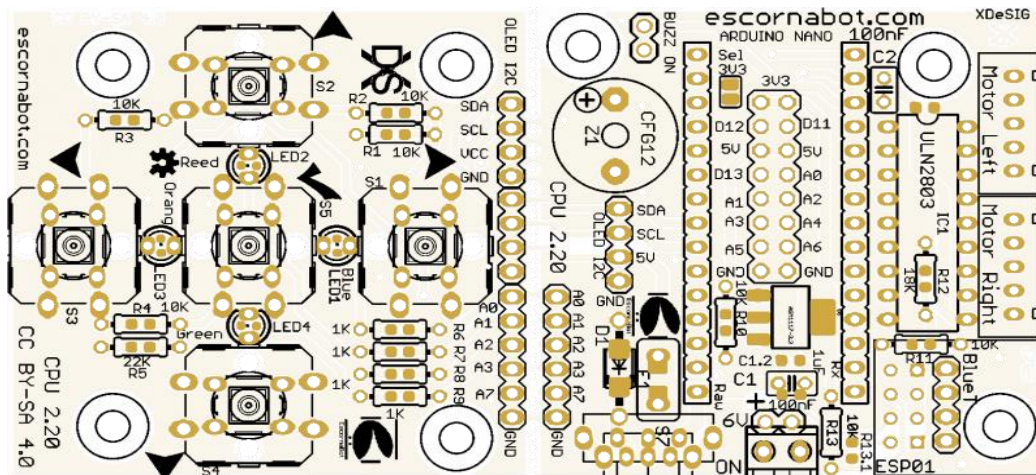
Por otro lado tenemos también el Kit de piezas impresas en 3D, que junto con el listado anterior conforman el total de materiales necesarios para fabricar el robot:

Imagen	Ud.	Descripción
	1	Rueda izquierda <i>wheel-l.stl</i>
	1	Rueda derecha <i>wheel-r.stl</i>
	1	Sujeción pilas <i>battery-bracket.stl</i>
	1	Sujeción motores <i>motoresMotorBracket.stl</i>

	1	<p>Sujeción bola <i>ballcaster-v2.stl</i></p>
	1	<p>Sujeción placa <i>PCBSupport-V2.2.stl</i></p>

5.3. PCB Botonera Escornabot

La placa de circuito impreso (PCB) que vamos a utilizar para construir nuestro Escornabot DIY, se trata de la versión 2.20 diseñada por la empresa XDeSIG:



Los componentes electrónicos que deberemos soldar en la PCB para tener nuestro circuito listo son los siguientes:

Cantidad	Nombre	Huella
2	Condensadores 100 nF	C-EU025-025X050
1	Condensador 22uF SMD 1206	
6	Resistencia 10kOhm	R-EU_0204/7
1	Resistencia 18kOhm	R-EU_0204/7
3	Resistencia 1kOhm	R-EU_0204/7
1	Resistencia 22kOhm	R-EU_0204/7
1	Diodo 1N4001	DO41-10
2	Conectores motores XH2.54 mm 5 pines	XH2.54 5P
5	Pulsadores 12x12 mm + capuchón	
4	Leds de 3mm, uno de cada color	LED3MM
1	Zumbador pasivo CFG12	F/CM12P
1	Fusible PTC PolySwitch 500mA	PFRA.050
1	Puente en H ULN2803A	ULN2803A
1	Zócalo PCB DIP16	
1	Jumper 2.54mm	
1	<i>PinHeader</i> Macho 2P 2.54mm	
2	<i>PinHeader</i> Hembra 2x04P 2.54 mm	
1	<i>PinHeader</i> Hembra 1x04P 2.54 mm	
1	Conector Roscado 5mm 2P	
2	<i>PinHeader</i> Hembra 1x15P 2.54mm	
1	Regulador de tensión 3.3V AMS1117	AMS1117
1	Placa Arduino Nano	
1	Interruptor deslizante	SK12D07 VG2

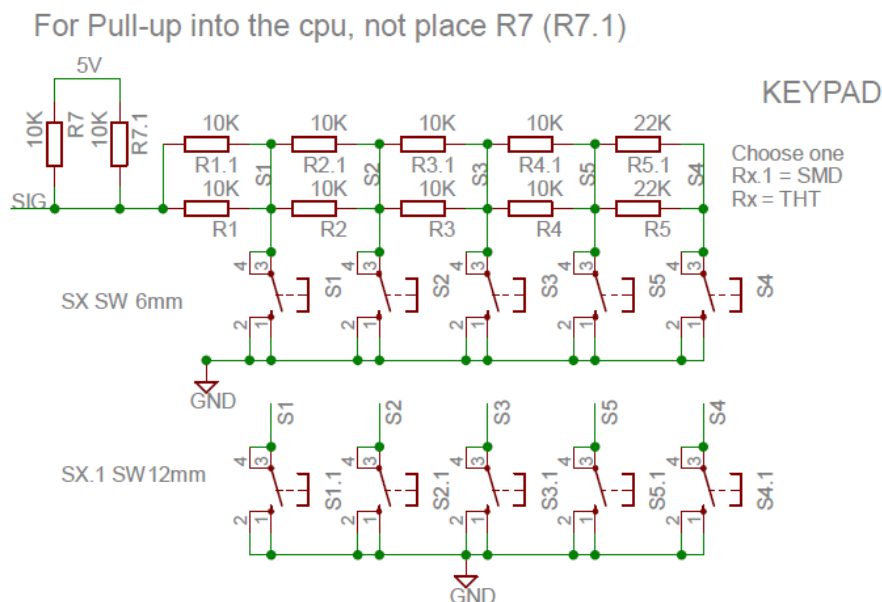
Por otro lado en la siguiente tabla se describe la función de cada uno de los pines de la PCB:

Nombre	E/S	Descripción
5V	IN	Entrada de alimentación 5 V
Sig	OUT	Señal de salida para el estado de la pulsación.
L1	IN	Entrada digital para control del LED 1
L2	IN	Entrada digital para control del LED 2
L3	IN	Entrada digital para control del LED 3
L4	IN	Entrada digital para control del LED 4
GND	IN	Entrada de tierra

Respecto a la soldadura de componentes en la placa, con lo único con lo que tenemos que tener un poquito de cuidado es con los LEDs, dado que a diferencia de los demás componentes los LEDs tienen polaridad y no da igual en qué sentido hagamos la soldadura. Para no cometer errores, fíjate bien en la imagen anterior de la PCB. Los cuatro LEDs tienen el cátodo (negativo) a la izquierda y el ánodo (positivo) a la derecha.

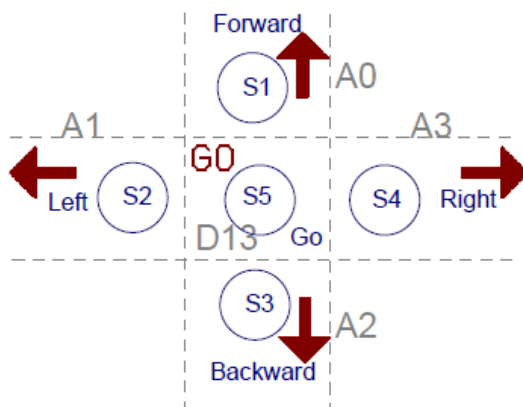
5.4. Esquema electrónico botonera

Las siguientes imágenes muestran el esquema del circuito electrónico de la botonera que emplea el Escornabot versión DIY:



Téngase en cuenta que el esquema muestra dos opciones a elegir entre tipos de resistencias, dado que la PCB versión 2.20 para la botonera de Escornabot permite montaje de resistencias tanto de tipo THD como SMD, lo que significa que en el circuito anterior solo tenemos que quedarnos con una de cada pareja de resistencias (R1 o R1.1, R2 o R2.1, etc...)

Asimismo la siguiente imagen muestra la distribución de pulsadores en la botonera y el identificador utilizado para cada uno de ellos en el circuito:



En el circuito mostrado no se incluyen los LEDs ni las resistencias asociadas a cada uno de ellos, pero son muy sencillos de intuir, ya que se trata solo de controlar el encendido y apagado de 4 LEDs independientes desde la placa Arduino, tal y como estamos ya acostumbrados a hacer.

5.5. Comprobación de la botonera

Analizando el esquema electrónico de la botonera de Escornabot del apartado anterior, nos damos cuenta de que se trata realmente de un circuito muy simple, un divisor de tensión de seis resistencias.

Un divisor de tensión es una forma inteligente de evitar tener que utilizar los cinco pines digitales del Arduino que nos harían falta para controlar los cinco pulsadores de la botonera de forma independiente, y en su lugar emplear un único pin analógico para manejar el teclado entero. ¿Cómo distinguimos entonces un pulsador de otro? Pues gracias a las resistencias conectadas formando un divisor de tensión.

Fíjate que en función del pulsador que accionemos obtenemos un circuito cerrado con más o menos resistencias conectadas en serie, que al sumarse nos dan una resistencia total medida entre los puntos “Sig” y “GND”, distinta para cada pulsador, lo que implica que el valor de tensión en “Sig” será también dependiente de cada pulsador.

Sabiendo esto, lo primero que tenemos que hacer después de haber soldado los componentes a la PCB, es asegurarnos de que la placa funcione correctamente, y en caso de existir algún componente dañado, mal soldado o de habernos equivocado con el código de colores de las resistencias, detectar el problema sobre la marcha para evitar arrastrar errores. La forma de hacer esta comprobación es medir con un multímetro el valor de la resistencia entre “Sig” y “GND” para cada pulsación. Analizando el esquema electrónico de la PCB resulta fácil deducir que los valores esperados son los siguientes:

Pulsador	Resistencia
S1	10 Kohm
S2	20 Kohm
S3	30 Kohm
S5	40 Kohm
S4	62 Kohm

Si midiendo con el multímetro obtenemos los valores anteriores podemos estar seguros de que la PCB funciona correctamente y podemos proseguir tranquilos con el proyecto.

5.6. Ajustes de la botonera

Como ya hemos adelantado, nuestra botonera es básicamente un divisor de tensión que nos da a la salida un valor de voltaje entre 0 y 5V diferente, según el pulsador que accionemos.

Conectaremos la salida de la botonera a una entrada analógica del Arduino. Para hacer un poco de repaso, recordemos que las entradas analógicas del Arduino van directas a un conversor analógico-digital (ADC) con una resolución de 10 bits. Este conversor convierte valores específicos de tensión en secuencias binarias, que por mayor sencillez podemos referirnos a ellas por su correspondiente valor entero decimal.

Al tener 10 bits de resolución, el ADC muestrea, cuantifica y codifica la tensión de entrada de 0 a 5V, y a su salida ofrece un total de 1024 valores diferentes de 0 a 1023, correctamente escalados respecto de la tensión de entrada. Por ejemplo, a un valor de tensión de 0 V le corresponde el número 0, a 5 V el número 1023, a 2.5 V el número 511, etc...

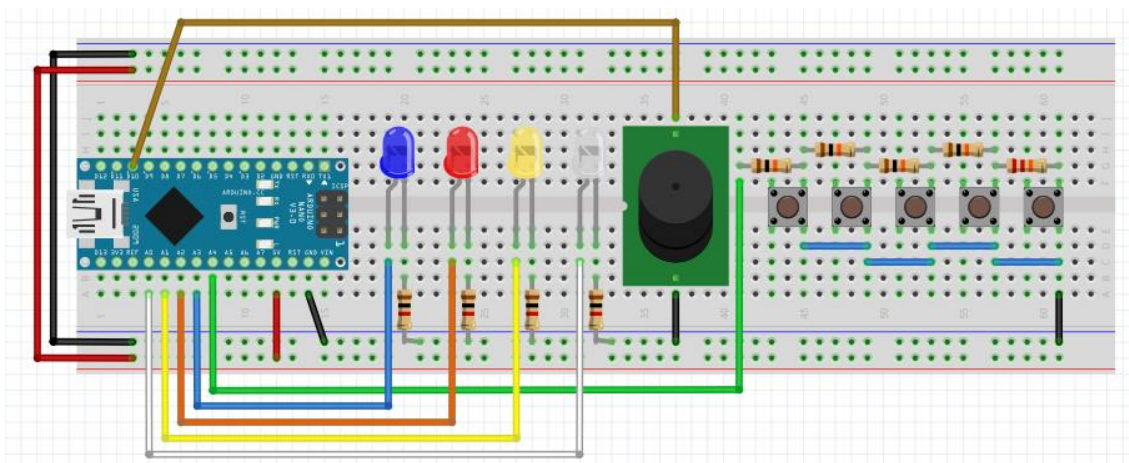
Para poder realizar más adelante la programación del robot, necesitamos por tanto conocer el valor de tensión exacto en el pin analógico para cada pulsador, así como el valor digital asignado por el ADC a cada uno de esos valores de voltaje.

Antes de empezar debemos aclarar que existen dos formas de trabajar con la PCB de Escornabot:

- Sin hacer uso de la entrada de 5V de la PCB, declarando la entrada analógica del Arduino como INPUT_PULLUP.
- Haciendo uso de la entrada de 5V de la PCB, declarando la entrada analógica del Arduino como INPUT normal y corriente.

Es muy importante diferenciar una solución de otra porque los valores de tensión que se obtienen en "Sig" son diferentes en ambos casos. Nosotros aplicaremos la primera solución porque hemos comprobado que los cinco posibles valores de tensión distan un poco más unos de otros, lo cual es mejor ya que se reduce la probabilidad de error.

Lo primero que tenemos que hacer es conectar la PCB Botonera al Arduino. También puedes realizar el mismo circuito en una protoboard por si quieres profundizar un poco más y conocerlo mejor. En cualquier caso, el circuito completo de la botonera con los LEDs y el zumbador sería el siguiente:



Aunque de momento para tomar los valores de tensión de la botonera basta con conectar de la PCB solo los pines de “GND” a la tierra del Arduino y de “Sig” de la entrada analógica “A4”.

A continuación hacemos un programita sencillo para configurar solo la entrada analógica A4 en modo INPUT_PULLUP y lo cargamos en el Arduino.

Hecho eso, ahora podemos medir el valor de tensión con el multímetro entre “Sig” y “GND”, y en una tabla anotar el valor de tensión que llega a “A4” para cada pulsador.

Los valores que nosotros hemos obtenido han sido los siguientes:

Pulsador	Resistencia	Tensión
S1	10 Kohm	1,0 V
S2	20 Kohm	1,6 V
S3	30 Kohm	2,0 V
S5	40 Kohm	2,4 V
S4	62 Kohm	2,8 V
SP	-	4,6 V

Para saber el valor digital asociado a cada valor de tensión podemos hacer un programita sencillo que nos permita leer simplemente el valor de la entrada analógica A4 con cada pulsación, y comparar el valor real con el valor teórico, quedando la tabla finalmente de la siguiente manera:

Pulsador	Resistencia	Tensión	Valor teórico	Valor real
S1	10 Kohm	1,0 V	205	766
S2	20 Kohm	1,6 V	328	881
S3	30 Kohm	2,0 V	410	508
S5	40 Kohm	2,4 V	492	817
S4	62 Kohm	2,8 V	573	680
SP	-	4,6 V	942	1023

5.7. Memoria del Escornabot

A la hora de programar es recomendable tener en cuenta que siempre resulta más sencillo dividir un problema grande en problemas más pequeños y resolver cada uno por separado, que tratar de enfrentar el problema grande de entrada. Pues eso mismo es lo que vamos a hacer con el Escornabot.

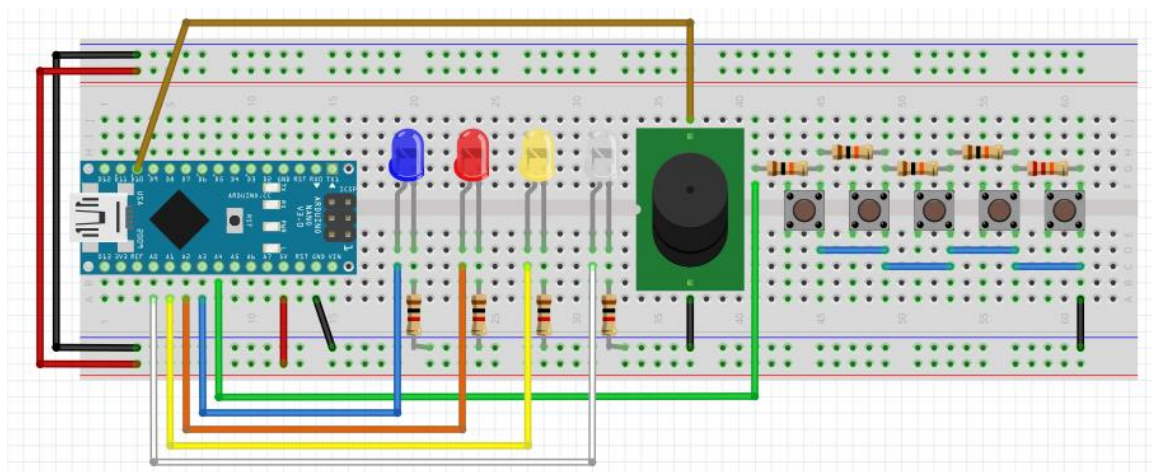
Primero vamos a realizar un programa para implementar la memoria del robot dejando de momento de lado los motores usando LEDs en su lugar que son más sencillos de manejar. El programa deberá permitir almacenar por orden las pulsaciones del teclado en un array que hará la función de memoria. Para cada uno de los cuatro pulsadores de dirección tendremos un LED asignado, de manera que cuando queramos ejecutar la secuencia almacenada pulsando el pulsador central, los LEDs se deberán encender y apagar en el mismo orden en el que los pulsadores fueron accionados.

Este código lo mantendremos intacto en el diseño final del Escornabot, dado que los LEDs los utilizaremos de la misma manera para acompañar los desplazamientos del robot mediante el encendido en cada momento del LED correspondiente.

Una vez conseguido esto, podemos añadir también un zumbador para darle un poco más de vida al robot mediante sonidos. Recomendamos añadir tonos en las siguientes situaciones:

- Un tono de frecuencia determinada para cada vez que se pulse un pulsador de dirección.
- Un tono de frecuencia determinada para cuando se pulse el pulsador central de ejecutar secuencia.
- Un tono de frecuencia determinada a medida que el robot vaya ejecutando cada desplazamiento de la secuencia.
- Un tono de frecuencia determinada cuando el robot termine de ejecutar toda la secuencia y se detenga.

Como circuito electrónico podemos utilizar la misma PCB del Escornabot, o si eres de los que le gusta hacerlo todo desde cero también puedes montar por ti mismo el circuito en la protoboard siguiendo este esquema:



Para revisar tu código podrás encontrar la solución resuelta entre los recursos compartidos del curso.

5.8. Programación de los motores paso a paso

Antes de integrar los motores paso a paso en el Escornabot, vamos primeramente a ver cómo funcionan con un ejemplo sencillo que consista simplemente en hacer girar los motores durante tres segundos en un sentido y durante tres segundos en sentido opuesto de forma repetitiva.

La siguiente tabla muestra las conexiones a realizar entre los pines digitales de los driver de los motores y los pines digitales del Arduino, así como el nombre de la constante que utilizaremos para referirnos a cada uno de los pines en el programa del IDE:

Nombre	Pin motor	Nombre constante	Pin digital Arduino
Motor derecha	IN1	IN1	5
	IN2	IN2	4
	IN3	IN3	3
	IN4	IN4	2
Motor Izquierda	IN1	IN5	9
	IN2	IN6	8
	IN3	IN7	7
	IN4	IN8	6

Para revisar tu código podrás encontrar la solución resuelta entre los recursos compartidos del curso.

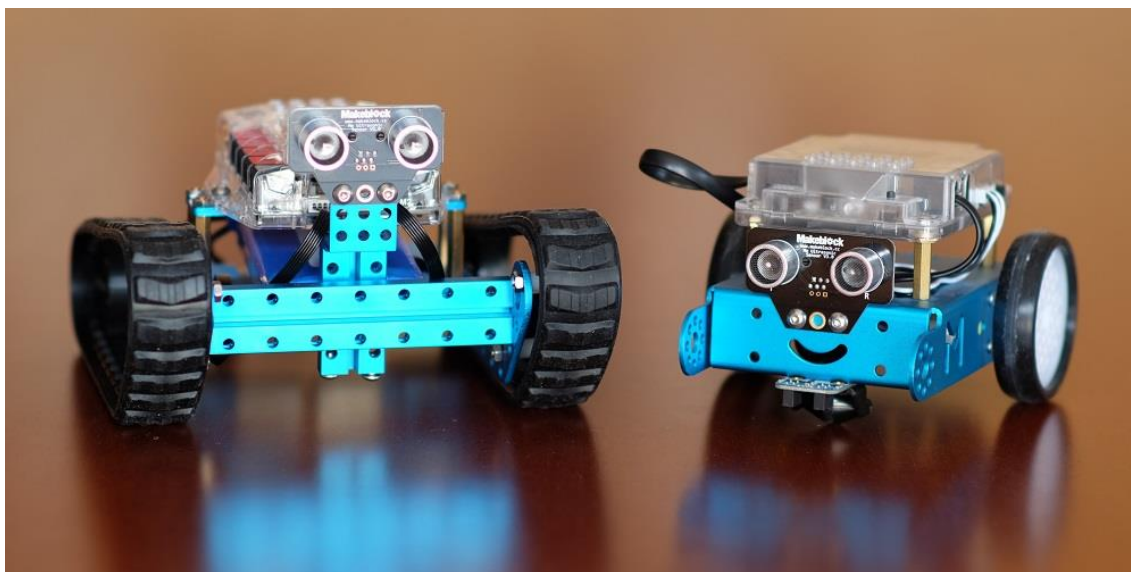
5.9. Programación final

Llegado a este punto deberías tener por un lado el código para la memoria del Escornabot y por otro el código para control de los motores paso a paso. Con estos dos programas realizar la programación final del Escornabot es bastante sencillo, dado que en realidad ya lo tienes todo, a falta solo de pensar un poco para ver como unir ambos códigos.

Trabajaremos este último ejercicio de forma presencial en el curso. Si necesitas acceder a la solución final podrás encontrar el código totalmente resuelto entre los recursos compartidos del curso.

6. Makeblock

Otra alternativa muy empleada en educación son los robots comerciales de MakeBlock. Estos modelos están orientados a reducir al mínimo los errores de montaje electrónicos, contando con conectores RJ25 para la unión de los módulos e integrando todos los sensores necesarios en su placa electrónica central, permitiendo al estudiante centrarse en la programación.

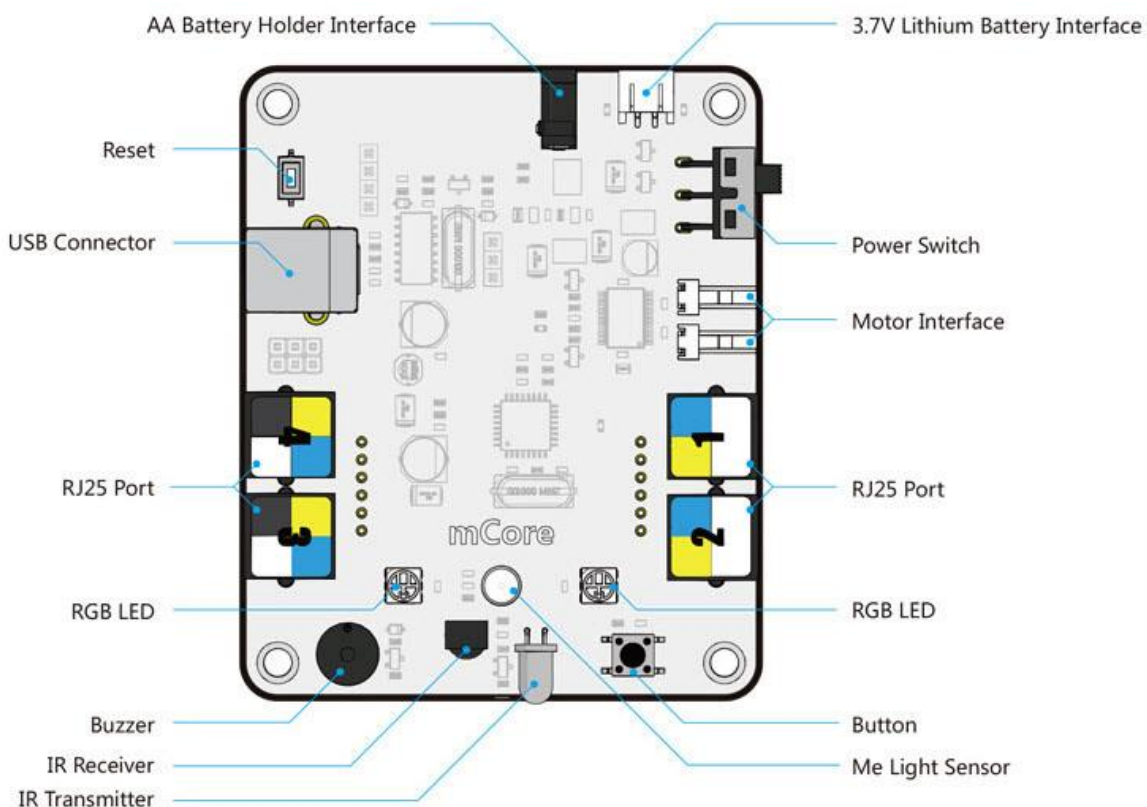


Existen principalmente 2 modelos: el mbot (robot de la derecha) y el makeblock ranger (robot de la izquierda). Ambos cuentan con módulos de sensor de ultrasonidos como los ya vistos en el curso, sensores de infrarrojo para seguimiento de línea, LEDs RGB en su parte superior, sensor de luminosidad, así como un botón integrado. Además, existen múltiples módulos para conectar a dichos robots: teclado matricial, pantalla matricial, servo, motores paso a paso, motores DC, etc.

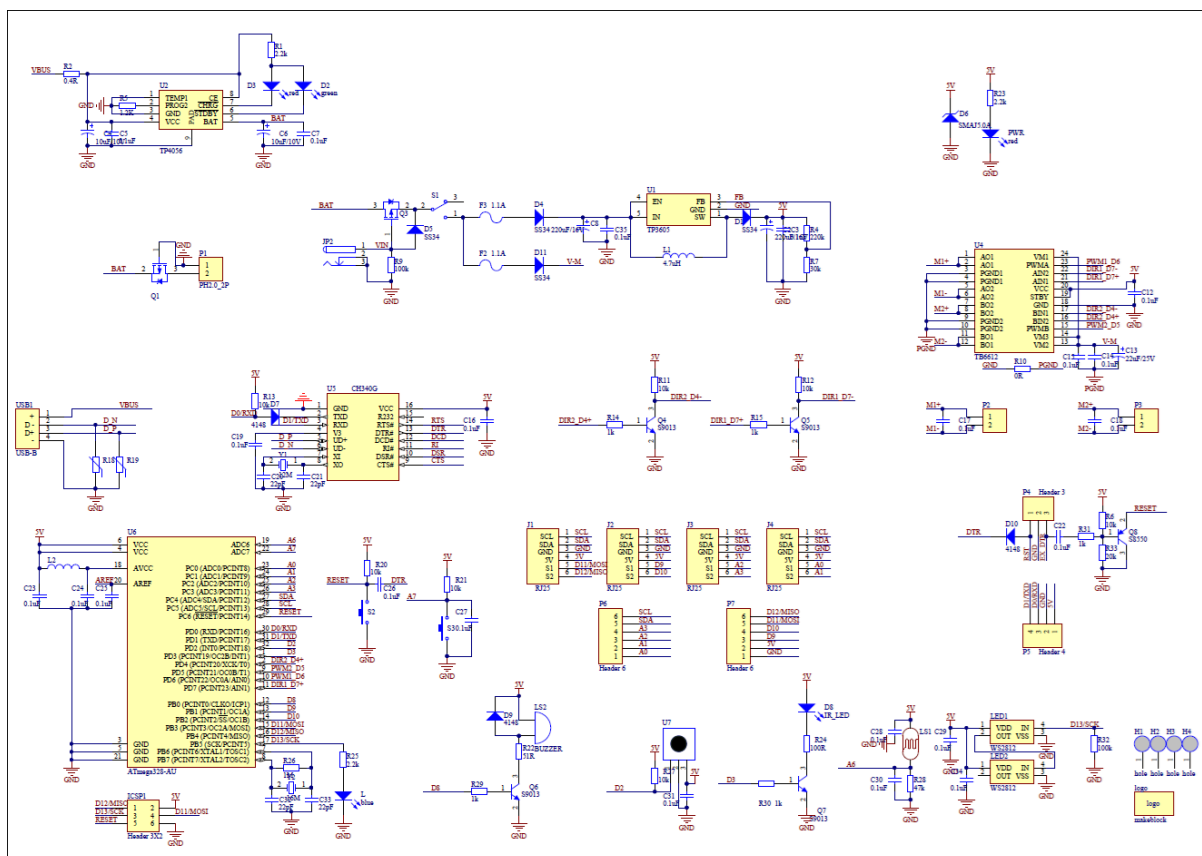
Todos estos sensores se encuentran conectados a los pines del AtMega de la siguiente forma:

COMPONENTE	PIN
2 Leds RGB (WS2812B)	D13
Buzzer	D8
Motor 1	D6, D7
Motor 2	D4, D5
Emisor de infrarrojos	D3
Receptor de infrarrojos	D2
Sensor de luz	A6
Pulsador	A7

Su placa de control (CPU) incorpora un ATmega 328P en el caso del mbot y un ATmega 2560 en el caso del Ranger. Por tanto, ambos dispositivos pueden ser programados desde el IDE de Arduino seleccionando las placas Arduino Uno y Arduino Mega, respectivamente.



El esquemático del mBot Core es el siguiente:

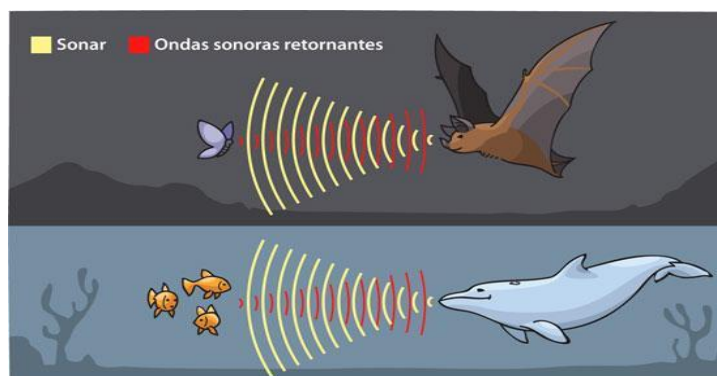


6.1. Conceptos de repaso

6.1.1. Sensor de Ultrasonido

6.1.1.1. La ecolocalización

La ecolocalización es una capacidad que poseen algunos animales, como el murciélago y el delfín, de orientarse y reconocer su entorno mediante la emisión de sonidos, analizando el eco recibido cuando la onda sonora choca y rebota contra un obstáculo. Estos animales pueden conocer la distancia a la que se encuentran los objetos midiendo la diferencia de tiempo entre la señal emitida y la recibida, incluso pueden ser capaces de captar información más específica, como la posición espacial del objeto, su tamaño y otras características.



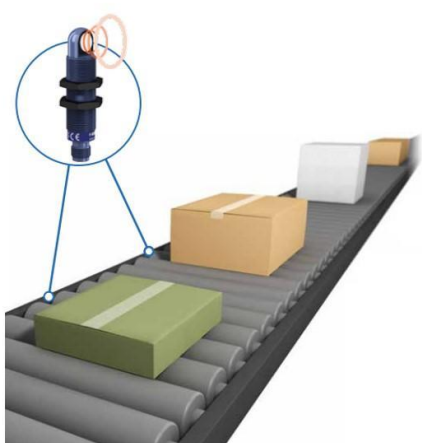
6.1.1.2. Ultrasonidos

Los ultrasonidos son ondas sonoras de frecuencia superior a la perceptible por el oído humano, es decir, superiores a los 20.000 Hz. Recordemos que el oído humano solo es capaz de percibir el sonido de frecuencias comprendidas entre los 20 Hz y los 20.000 Hz.

En la práctica los ultrasonidos son muy útiles en diversos campos como por ejemplo la medicina. Por lo general a todos nos suena el término ecografía. Se trata de imágenes creadas a partir del “eco”. Las máquinas utilizadas para generar ecografías, permiten crear imágenes de órganos internos gracias a los ultrasonidos, ondas sonoras que tras rebotar contra las estructuras corporales retornan al dispositivo, el cual es capaz de crear las imágenes a partir del análisis de la intensidad y tiempo de retorno de las ondas ultrasónicas.



En el sector industrial, los ultrasonidos se emplean comúnmente como sensores para medir distancias o detectar obstáculos, así como para limpieza de determinadas piezas y componentes electrónicos por medio de bañeras de ultrasonidos.

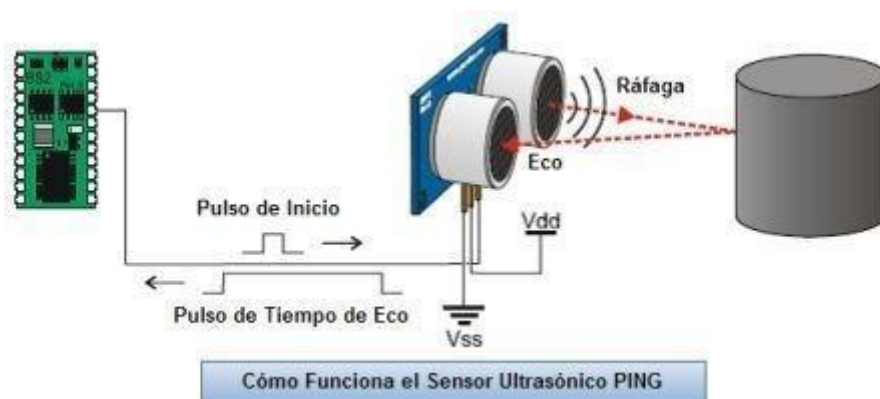


6.1.1.3. Sensor de proximidad HC-SR04

El HC-SR04 es un sensor de proximidad por ultrasonidos que sirve para medir distancias. Su modo de operar se basa en todo lo que ya hemos descrito anteriormente acerca de las ondas ultrasónicas.



El dispositivo cuenta con un elemento transmisor y un elemento receptor. El transmisor emite pulsos ultrasónicos que rebotan contra los objetos cercanos y son recibidos por el receptor. Conociendo la velocidad del sonido y midiendo el tiempo de retorno, somos capaces de calcular la distancia a la que se encuentra el objeto que reflejó la onda sonora.



Las principales características del sensor son las siguientes:

- Alimentación a 5 V
- Rango de medición: de 2 cm a 400 cm (en la práctica se queda en torno a 200 cm)
- Frecuencia del pulso: 40 KHz
- Apertura del pulso: 15 grados
- Señal de disparo: 10 us
- Dimensiones del módulo: 45x20x15 mm

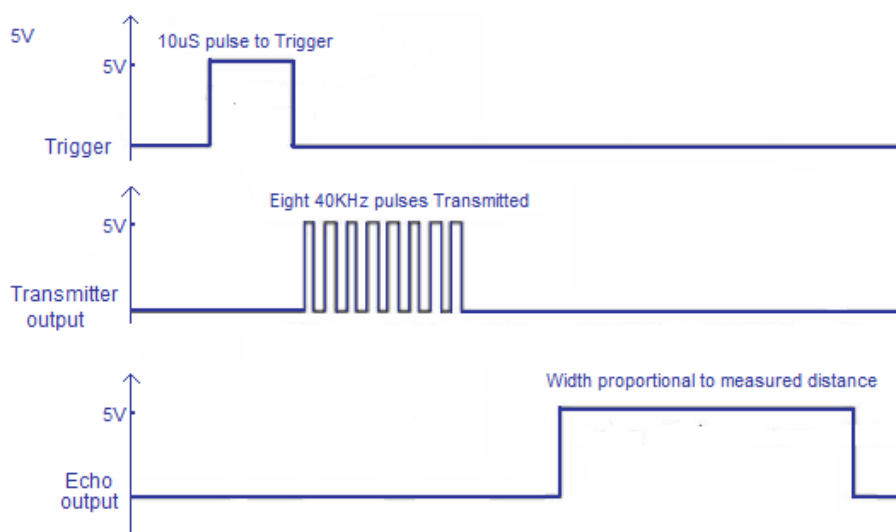
En cuanto a las funciones de sus entradas y salidas tenemos lo siguiente:

Nombre	E/S	Descripción
VCC	IN	Entrada de alimentación 5 V
Trig	IN	Señal de Trigger. Cuando recibe un pulso de 10 us el transmisor emite un sonido de 8 pulsos a 40 KHz
Echo	OUT	Señal de Eco. Emite una pulso digital de duración igual al tiempo que la onda emitida tardó en llegar al receptor
GND	IN	Entrada de tierra

6.1.1.4. Algoritmo y funcionamiento del sensor

Para medir una distancia con el sensor de ultrasonidos HC-SR04 el funcionamiento sería el siguiente:

- Desde cualquier pin digital de Arduino se envía un único pulso de 5V y 10us de duración a la entrada “Trig” del sensor.
- Tras recibir dicha señal, el transmisor del sensor indicado mediante la letra “T” emitirá una onda sonora ultrasónica compuesta de 8 pulsos y frecuencia de 40 KHz.
- La onda sonora rebotará contra un objeto del medio físico y regresará al sensor a través del elemento receptor indicado con la letra “R”.
- Después que el receptor ha recibido los 8 pulsos de la onda sonora, el sensor mide el tiempo que la onda ha tardado en regresar, y comunica dicho dato generando una señal de un único pulso de duración igual al tiempo calculado y la envía a través de la salida “Echo”.



- De vuelta al Arduino, para conocer la duración del pulso emitido por la señal “Echo”, recurrimos a la función “pulseIn()” que explicaremos más adelante en el apartado de programación.

- Ahora que ya tenemos el tiempo de propagación de la onda, podemos calcular fácilmente la distancia recorrida por la misma y por tanto la distancia entre el sensor y el objeto. Sabiendo que la velocidad del sonido es de 343 m/s, el cálculo sería el siguiente:

Primero aplicamos la siguiente conversión de unidades:

$$343 \frac{m}{s} = 34300 \frac{cm}{s} = 0,0343 \frac{cm}{\mu s} = \frac{1 cm}{29,2 \mu s}$$

Lo que significa que la velocidad del sonido recorre 1 cm en 29,2 μs . El espacio recorrido por la onda sonora es el doble de la distancia entre el sensor y el objeto puesto que la onda ha de ir y volver, por tanto la fórmula para calcular dicha distancia será la siguiente:

$$2 \times Distancia = Tiempo \cdot Velocidad$$

$$Distancia = \frac{Tiempo \cdot Velocidad}{2}$$

$$Distancia = \frac{Tiempo}{2 \times 29,2}$$

6.2. Programación del mBot desde Arduino IDE

Para hacer uso del IDE de Arduino para programar nuestro robot, debemos descargar y añadir la librería de MakeBlock a nuestro IDE. Esta puede localizarse en el siguiente enlace:

https://www.makeblock.es/soporte/librerias_makeblock_arduino/

Tras instalarla, podemos localizar diversos ejemplos preparados con los que familiarizarnos con su programación. Únicamente debemos tener en cuenta modificar la cabecera según el Core empleado:

Tipo de Placa	Librería
Orion	MeOrion.h
Baseboard	MeBaseBoard
mCore	MeMCore.h
Shield	MeShield.h
Auriga	MeAuriga.h
MegaPi	MeMegaPi.h

Abriremos algunos de estos ejemplos y los cargaremos para verificar el correcto funcionamiento de nuestro robot. Tras esto, plantearemos diferentes actividades hasta conseguir programar nuestro propio robot de navegación autónoma.

6.3. ACTIVIDAD 1 MBOT: BUZZER TEST

En este primer ejemplo, emplearemos el zumbador incorporado para producir nuestra propia melodía. Para ello, debemos hacer uso de la clase *MeBuzzer* y el método *tone(pin, frecuencia, duración)*.

6.4. ACTIVIDAD 2: SENSOR DE LUMINOSIDAD

En este ejemplo, emplearemos el sensor de luminosidad y leeremos su valor analógico para mostrarlo mediante puerto serie. Se debe hacer uso de la clase *MeLightSensor* y el método *read()*.

6.5. ACTIVIDAD 3: SENSOR DE ULTRASONIDOS

Este ejemplo, consistirá en hacer uso del sensor de ultrasonidos para obtener el valor de la distancia medida y mostrarlo por Puerto Serie.

6.6. ACTIVIDAD 4: ROBOT AUTÓNOMO CON SENSOR ULTRASONIDOS

A continuación, haremos uso de lo aprendido en los ejemplos anteriores para programar un robot que al localizar un objeto a menos de 10 cm, se mueva hacia atrás y gire a la derecha.

6.7. ACTIVIDAD 5: AÑADIENDO EXTRAS A NUESTRO ROBOT

Tras conseguir una correcta navegación del robot, añadiremos una señal luminosa mediante los LEDs RGB, así como una señal sonora con el sensor de ultrasonidos para indicar que se ha encontrado un obstáculo. En este caso, al localizar un obstáculo los LEDs deben iluminarse en ROJO y VERDE, y el zumbador debe sonar. Por el contrario, mientras no exista obstáculo, los LEDs deben estar en verde.