Robot Arduino controlado mediante sensores y con respuestas sobre actuadores

Asier Pérez de Lazarraga Mangado
Grado en Tecnologías de Telecomunicación
Arduino

José López Vicario
Pere Touset Peiró

Junio de 2017

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Robot Arduino controlado mediante sensores y con respuestas sobre actuadores*
Nombre del autor:	*Asier Pérez de Lazarraga Mangado*
Nombre del consultor/a:	*José López Vicario*
Nombre del PRA:	*Pere Touset Peiró*
Fecha de entrega (mm/aaaa):	06/2017
Titulación:	*Grado de Tecnologías de Telecomunicación*
Área del Trabajo Final:	*Arduino*
Idioma del trabajo:	*Castellano*
Palabras clave	*Arduino. Robótica. Sensórica.*

Resumen del Trabajo (máximo 250 palabras):

Las tecnologías IT hoy en día están amplia y totalmente asentadas en ámbitos domésticos y empresariales. En este contexto, están surgiendo nuevos conceptos como el *Big Data* o el *Internet of Thinks* en los que la sensórica ha adquirido un papel fundamental. Y es que los sensores son los encargados de definir el entorno tomando todo tipo de datos de él.

Estos datos como tal no serían de mayor utilidad sin su análisis y/o procesamiento. Microcontroladores o microprocesadores son usados para transformarlos a información útil con la que se pueda tomar alguna decisión con el fin de realizar alguna acción. De esta forma, con una interconexión y comunicación entre dispositivos adecuada, se automatizan simples o complejas tareas, en función de las variables de entrada.

Con todo ello, en este trabajo se implementa un robot gobernado por un microcontrolador Arduino que actúa en función de las señales que reciba como entradas. Además, pretende realizar un estudio de las posibilidades que actualmente hay en el mercado para la ampliación de este robot o el desarrollo de sistemas más complejos, visto que las oportunidades de crecimiento en ésta u otra solución son enormemente amplias.

Y es que, por todo lo expuesto, se concluye que la automatización de procesos o trabajos tiene un recorrido enorme en las que las tecnologías de información y telecomunicación son parte fundamental. Existen además muchas necesidades que pueden ser cubiertas por soluciones que pueden ser desarrolladas con herramientas que están al alcance de todo el mundo.

Abstract (in English, 250 words or less):

Nowadays IT technologies are very present at home and in most enterprises. In this context, new concepts such as Big Data or Internet of Thinks are emerging, in which sensory has acquired a fundamental role. This is because the sensors are the ones in charge of defining the environment taking all type of data from it.

That data would be useless without its analysis and / or processing. Microcontrollers or microprocessors are used to transform the data into useful information with which a decision can be made in order to perform some action. In this way, with an appropriate interconnection and communication between devices, simple or complex tasks, are automated depending on the input variables.

With this in mind, a robot controlled by an Arduino microcontroller is implemented in this work. This robot acts in function of the signals received as inputs. Additionally, it is intended to perform a study of the possibilities that currently exist in the market for the expansion of this robot, or the development of more complex systems. Growth opportunities in this or another solution are very big.

And because of all the above, it is concluded that processes or task automation in which information and telecommunication technologies are fundamental has an extremely promising future. And there are also a lot of needs that can be covered by solutions that can be developed with tools that are accessible to everyone.

Índice

Lista de figuras

Lista de ecuaciones

Lista de tablas

1. Introducción

1.1 Contexto y justificación del Trabajo

En la actualidad hay infinidad de sensores que pueden medir casi la totalidad de las magnitudes físicas o químicas que definen el entorno en el que se encuentran. Este hecho hace que estas mediciones puedan alimentar con gran cantidad de información sistemas ad-hoc o de propósito general. En base a este conocimiento del medio, el sistema puede definirlo con gran exactitud en cada instante con variables que le resulten de interés, y a partir de ello reaccionar de una forma o de otra. Y tanto es así que la sensórica es un pilar fundamental en conceptos como el internet de las cosas (*Internet of Things* en inglés y a menudo conocido por sus siglas IoT), Industria 4.0 o *Smart Cities* que tanto futuro tienen en el mundo de las IT (Tecnologías de la Información por las siglas en inglés de *Information Technology*).

La robótica y la automatización son a su vez, ramas de la ingeniería que pueden aprovecharse de forma más natural de estas funciones. Por sus características, pueden ejercer fácilmente sobre actuadores que modifiquen de forma mecánica (y también eléctrica) manipulando objetos físicos a su alcance. Y es que estos dos campos han evolucionado de los movimientos repetitivos de los que principalmente se basaban en el pasado a interactuar según las variables de entrada en cada momento.

Por otra parte, se ha extendido el movimiento “hágalo usted mismo” (DIY por sus siglas *Do It Yourself*) en el ámbito científico. Su expansión ha posibilitado que surja y se desarrolle Hardware libre con una comunidad muy amplia que da apoyo y soporte a diversas iniciativas de esta índole. Este hecho ha propiciado mejoras constantes y que exista cantidad de información y de repositorios para este tipo de material electrónico. Un claro exponente de estas ideas, y uno de los sistemas que más aceptación ha tenido, es Arduino.

Por tanto, uniendo todos estos conceptos, se pretende construir un robot que se desplace controlado por Arduino capaz de interactuar con el medio. Este prototipo podrá ser desarrollado para tareas más complejas en el futuro incluyendo más inputs y outputs progresivamente según las necesidades que se vayan detectando.

1.2 Objetivos del Trabajo

Los objetivos principales del proyecto de este Trabajo de Fin de Grado son:

- Construir un dispositivo robot controlado por Arduino que tenga capacidad de desplazamiento.

- Implementar una aplicación capaz de controlar sus movimientos de forma pasiva/preventiva o activa.

Además, se definen los siguientes objetivos secundarios:

- El equipo deberá estar dotado de varios sensores de diferente tipo y estar interconectado y ser accesible mediante tecnología inalámbrica, por medio de Bluetooth.
- Para conseguir la interactividad, se desarrollará una aplicación a cargar en el Arduino que defina las salidas en función de las entradas.
- Debe ser ampliable. Es decir, que el sistema sea capaz de adaptarse fácilmente a nuevas necesidades que surjan a posteriori. Por tanto debe ser un sistema abierto que deberá estar bien documentado para su mantenimiento, versiones posteriores e incluso posibles modificaciones.

1.3 Enfoque y método seguido

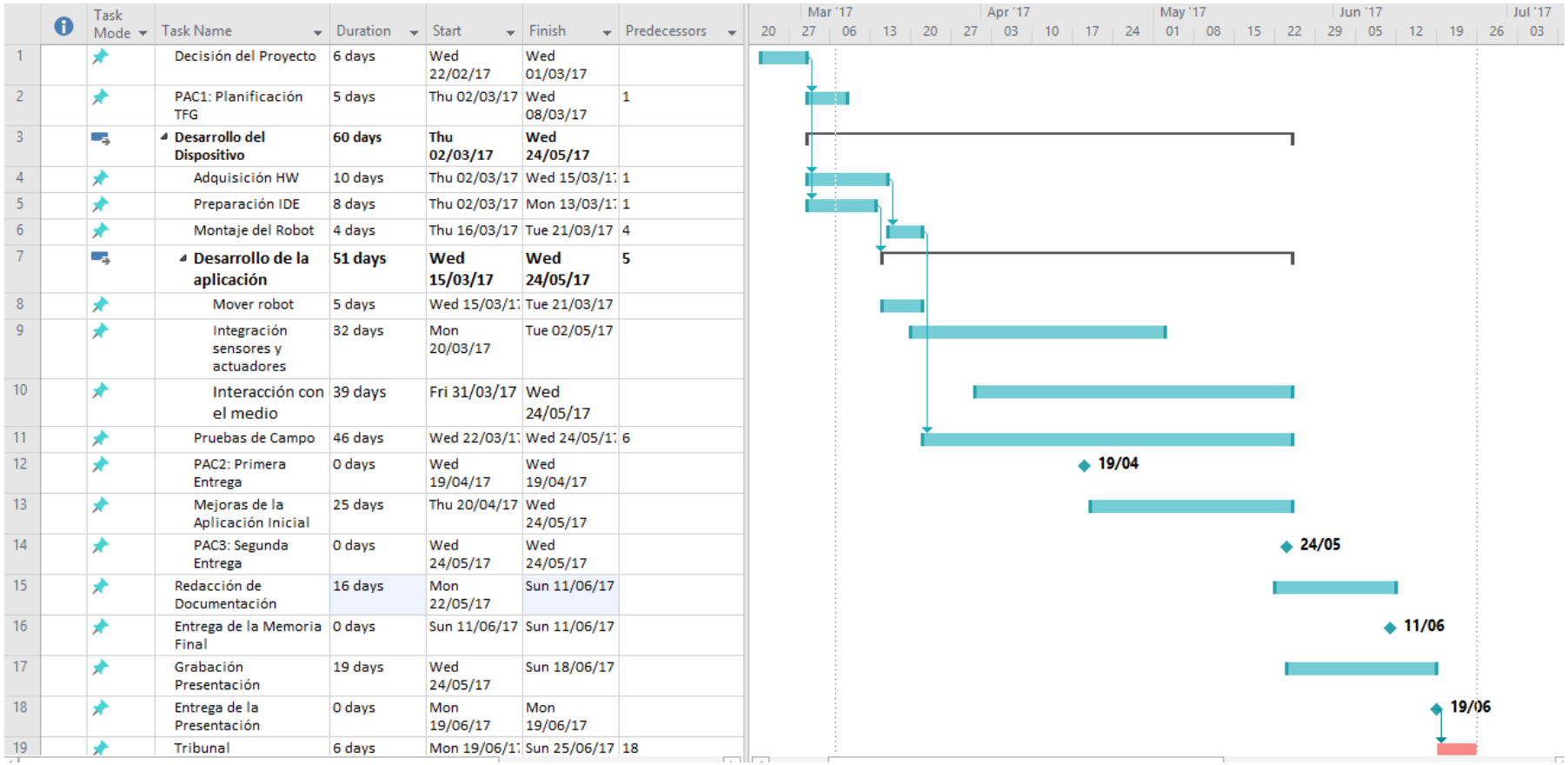
Se ha optado por adquirir el conjunto de la placa y robot en un paquete cerrado comercializado por una empresa especializada en ello. La construcción de principio a fin de un robot que se moviese hubiera conllevado complicaciones sobre todo en el diseño de la parte mecánica.

El proyecto, por consiguiente, ha consistido en montar y adaptar un producto existente y en desarrollar a partir de él varias alternativas y programas con el fin de conseguir los objetivos marcados. La cantidad de información relativa a Arduino que se encuentra en internet ha ayudado a avanzar hacia dicho propósito.

Se ha aprovechado también para explorar en sectores que tienden a sacar partido de campos que se estudian en el sector de las telecomunicaciones. Así, este trabajo ha servido al autor (y puede servir a las personas que estén interesadas en ello) para la investigación, y puede servir como elemento de apoyo para la consulta y como punto de partida para el desarrollo de sistemas y proyectos de mayor alcance.

1.4 Planificación del Trabajo

A continuación se presenta la planificación mediante diagrama de Gantt del Trabajo Fin de Grado:



	Task Mode	Task Name	Duration	Start	Finish	Predecessors
1		Decisión del Proyecto	6 days	Wed 22/02/17	Wed 01/03/17	
2		PAC1: Planificación TFG	5 days	Thu 02/03/17	Wed 08/03/17	1
3		**Desarrollo del Dispositivo**	**60 days**	**Thu 02/03/17**	**Wed 24/05/17**	
4		Adquisición HW	10 days	Thu 02/03/17	Wed 15/03/17	1
5		Preparación IDE	8 days	Thu 02/03/17	Mon 13/03/17	1
6		Montaje del Robot	4 days	Thu 16/03/17	Tue 21/03/17	4
7		**Desarrollo de la aplicación**	**51 days**	**Wed 15/03/17**	**Wed 24/05/17**	**5**
8		Mover robot	5 days	Wed 15/03/17	Tue 21/03/17	
9		Integración sensores y actuadores	32 days	Mon 20/03/17	Tue 02/05/17	
10		Interacción con el medio	39 days	Fri 31/03/17	Wed 24/05/17	
11		Pruebas de Campo	46 days	Wed 22/03/17	Wed 24/05/17	6
12		PAC2: Primera Entrega	0 days	Wed 19/04/17	Wed 19/04/17	
13		Mejoras de la Aplicación Inicial	25 days	Thu 20/04/17	Wed 24/05/17	
14		PAC3: Segunda Entrega	0 days	Wed 24/05/17	Wed 24/05/17	
15		Redacción de Documentación	16 days	Mon 22/05/17	Sun 11/06/17	
16		Entrega de la Memoria Final	0 days	Sun 11/06/17	Sun 11/06/17	
17		Grabación Presentación	19 days	Wed 24/05/17	Sun 18/06/17	
18		Entrega de la Presentación	0 days	Mon 19/06/17	Mon 19/06/17	
19		Tribunal	6 days	Mon 19/06/17	Sun 25/06/17	18

1.5 Breve sumario de productos obtenidos

Al finalizar el Proyecto se habrá desarrollado un robot controlado por un microcontrolador Arduino que será capaz de desplazarse e interactuar con el medio de una forma u otra en función de los parámetros de entrada que irá recogiendo mediante sensores conectados. Así mismo, deberá también responder con salidas luminosas o acústicas según se den las entradas.

1.6 Breve descripción de los otros capítulos de la memoria

En los siguientes capítulos de este documento se detallarán los elementos que han intervenido en la consecución de este trabajo:

En el primer apartado de introducción se hace un repaso del contexto, estado del arte, objetivos y planificación seguida.

En el segundo apartado se estudian las diferentes alternativas que existen a la hora de seleccionar el elemento central del robot: microcontroladores o microprocesadores.

El tercer apartado se centra en el propio robot. Se detalla el hardware instalado para su interconexión inalámbrica con otros dispositivos, los sensores y actuadores que se instalan o pudieran ser instalados y las capacidades de expansión. Además se estudia el puerto serie que es fundamental para la comunicación del Arduino con otros dispositivos.

En el cuarto capítulo se muestran diferentes entornos en los que podemos programar el equipo. Se repasan los propietarios del fabricante donde existe también una aplicación para dispositivos móviles, y el propio sistema de Arduino, con las diferencias que existen en cada uno de ellos.

En el quinto apartado se verá, sin entrar en detalle, la posibilidad existente de capturar y procesar los datos con software ajeno al propio Arduino desde el ordenador.

En el sexto apartado se muestra el presupuesto que se ha empleado para este proyecto.

En el séptimo apartado, se reflejarán las conclusiones que se han sacado en la puesta en marcha de este proyecto.

En los capítulos octavo, noveno y décimo están para consultar como referencia el glosario, la bibliografía y los anexos, respectivamente. En éste último, se recogen entre otras, partes de especificaciones de componentes.

2. Microcontrolador vs microprocesador

Con la expansión del fenómeno *maker* se han ampliado enormemente las posibilidades de elección de todo tipo de componentes electrónicos. Una de las primeras decisiones que se deben tomar en la mayoría de proyectos de esta naturaleza es la de gobernar el dispositivo con un microprocesador o con un microcontrolador.

En la siguiente tabla se hace un resumen de las principales características enfrentadas entre dos de los equipos de hardware (HW) libre más difundidos en la actualidad: el microprocesador Raspberry PI 3 y el microcontrolador Arduino Mega 2560:

	Microprocesador (Raspberry Pi 3)	**Microcontrolador (Arduino MEGA 2560)**
CPU	4× ARM Cortex-A53, 1.2GHz	ATmega2560. 16MHz
Memoria RAM	1GB LPDDR2 (900 MHz) en un circuito integrado aparte	8KB en el propio circuito integrado
Arquitectura	64 bits	8 bits
Velocidad de Operación	Rápida	Lenta en comparación
Propósito	Propósito general	Propósito más específico
Entradas/Salidas digitales	40 GPIO pins	E/S digitales: 54, de las cuales 15 PWM
Salidas analógicas	0	16
Video	Broadcom VideoCore IV HDMI	N/A
Sistema Operativo	Raspberry: Raspbian, Windows CE…	N/A
Consumo eléctrico	Mayor	Menor
Coste	Mayor	Menor
Interferencias	Más susceptibles a la interferencia electromagnética debido a su tamaño y a su menor nivel de integración	El alto nivel de integración reduce la interferencia electromagnética

Tabla 1: Microprocesador vs microcontrolador

Como puede comprobarse en la tabla 1, los microprocesadores son más potentes y más flexibles, y tienen mucha mayor capacidad de computación que los microcontroladores. Los microcontroladores, a su vez, están más enfocados a la electrónica y están más orientados al manejo de entradas y salidas. Así, en sistemas IoT, los equipos conectados a los sensores suelen estar gobernados por microcontroladores (recogida de datos), mientras que los equipos

encargados a procesar la cantidad de información de entrada suelen ser microprocesadores.

Por todo lo expuesto, para el propósito del presente Proyecto, se determina que un dispositivo Arduino es más óptimo que un microprocesador debido a que no se necesita un alto grado de computación ni de operaciones aritméticas con la información que se va a recoger, y sí una gran robustez a la hora de tomar los datos de las diferentes entradas y de producir los efectos requeridos en las diferentes salidas. El hecho de que no tenga un sistema operativo, hace que Arduino sea más sólido.

En cualquier caso, en otro tipo de proyectos ambos conceptos son compatibles y pueden convivir perfectamente realizando cada uno de ellos tareas diferentes. Y tanto es así que existen módulos de expansión para interconectar de manera más sencilla placas Arduino y Raspberry.

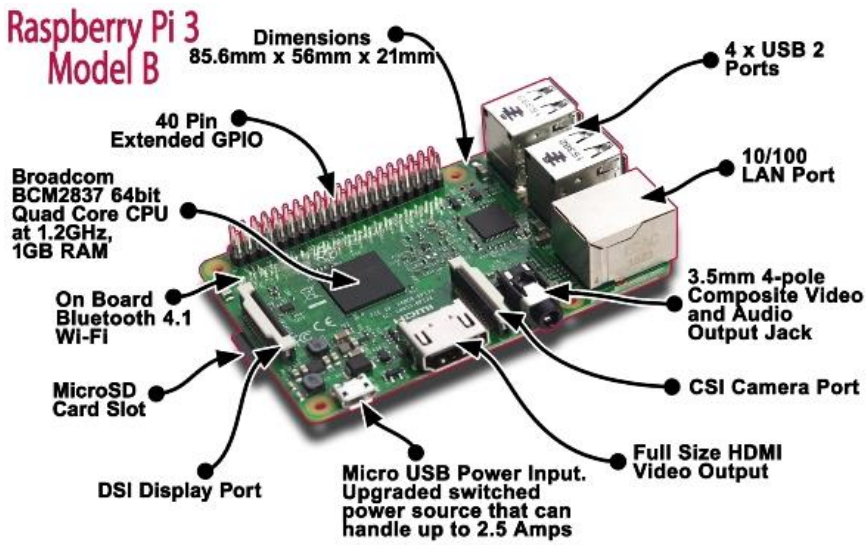


Figura 1: Raspberry Pi 3 Model B

3. Robot

Según la Real Academia Española (RAE), un robot es una “máquina o ingenio electrónico programable, capaz de manipular objetos y realizar operaciones antes reservadas solo a las personas”.

En la figura 2 se muestra el robot construido en una de sus posibles formas.

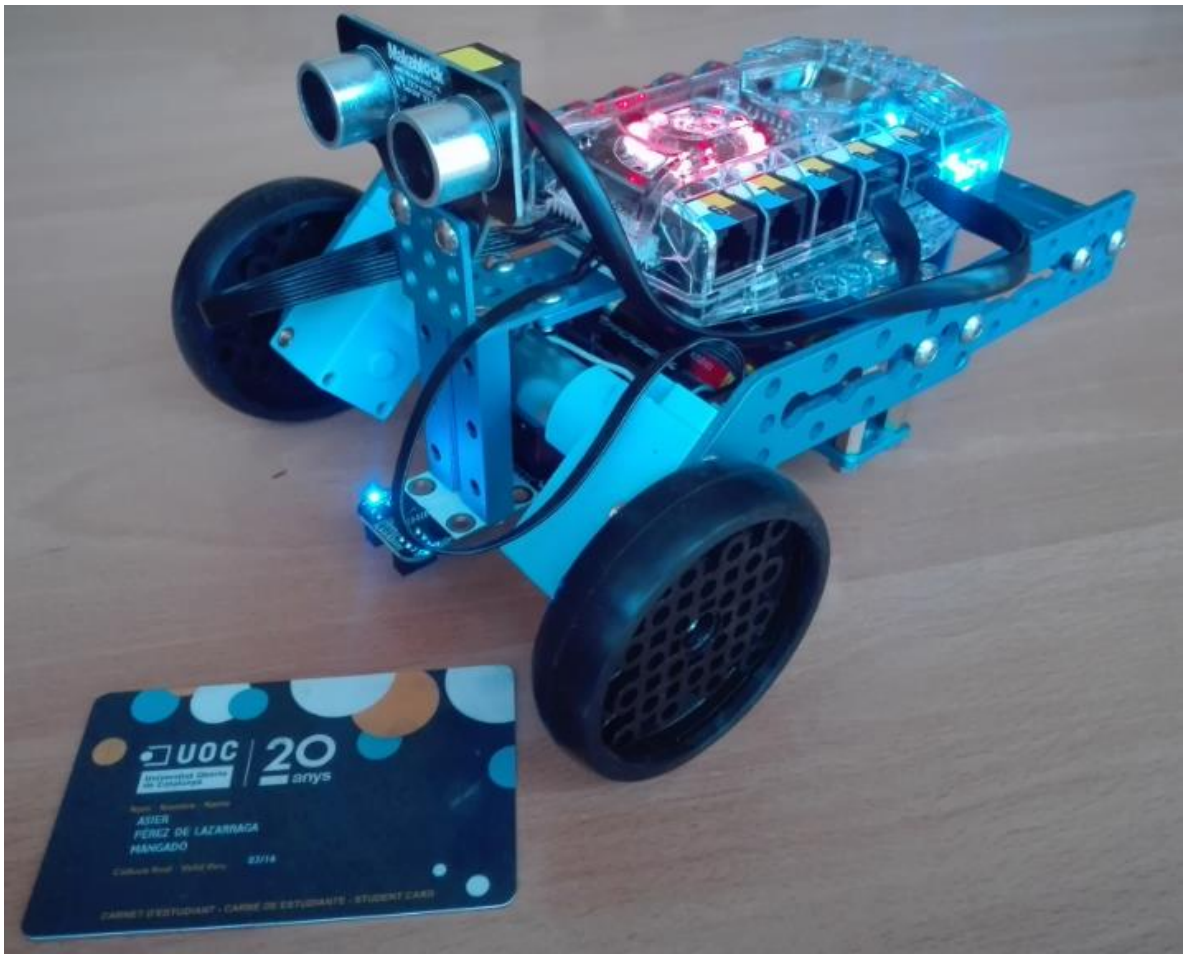


Figura 2: Robot gobernado por placa MeAuriga

Éste está gobernado por una placa base Me Auriga basada en el microcontrolador Arduino Mega 2560, con 10 puertos de expansión RJ25 y varios sensores y módulos integrados. El detalle de la misma se refleja en la siguiente figura 3:

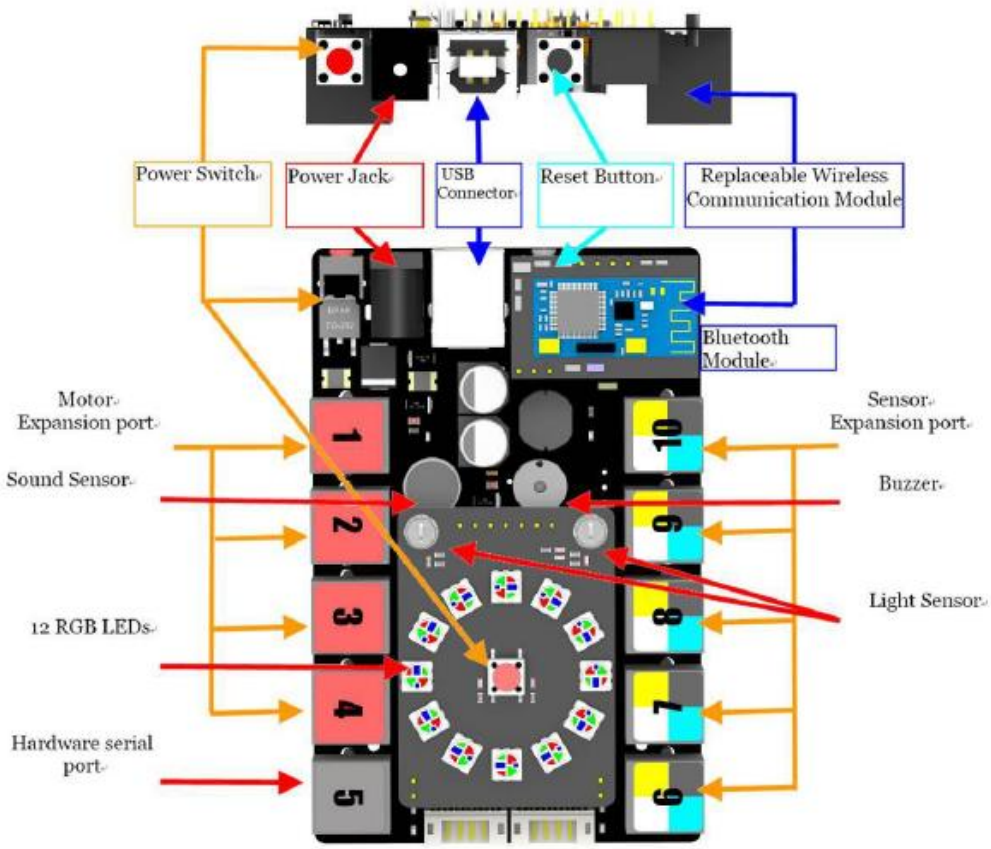


Figura 3: Placa ME Auriga [5]

3.1 Módulo Bluetooth

La placa ME Auriga cuenta con un módulo para conexión inalámbrica Bluetooth (es reemplazable) que soporta los protocolos Bluetooth 2.1, 3.0 y 4.0 y comunicación output a través del puerto serie. Su frecuencia de funcionamiento es de 2,4GHz, y hasta 115200 baudios y una distancia de operatividad de 10-15m en campo abierto.

Las características del módulo son válidas para las necesidades del proyecto. Además Bluetooth es un estándar ampliamente extendido y compatible con los otros dispositivos que se utilizan (portátil y smartphone) con lo que facilita la compatibilidad. Alternativas como Zigbee son descartadas por este motivo.

El módulo cuenta con un LED de color azul que se mantiene parpadeando cuando está a la escucha. Para conectarse vía Bluetooth lo primero que se tiene que hacer desde el PC o dispositivo móvil es activar el Bluetooth para buscar dispositivos cercanos. Si encuentra el BT del robot, el siguiente paso es enlazar los dispositivos. Al emparejarlos se solicitará una clave que se debe aceptar. Posteriormente, si se realiza la conexión, el LED del módulo pasará a azul fijo.

3.1.1 Diferentes estándares de Bluetooth

El módulo BT equipado, como se ha comentado anteriormente, soporta diferentes estándares entre los que se encuentra el BT 4.0 en cuya especificación se encuentra el BLE (*Bluetooth Low Energy*). Esta variedad difiere de las clásicas en la finalidad para la que ha sido concebida. Mientras que en las antiguas versiones principalmente se buscaba la mayor velocidad de datos, en ésta, se pretende rebajar el consumo.

Esta filosofía de bajo consumo aun a costa de reducción de velocidad está pensada para redes de sensores o IoT debido a que generalmente los datos a transmitir no deben ser de grandes volúmenes pero sí cada cierto tiempo y en equipos con baterías limitadas. De esta forma, el BLE se puede ajustar mejor a este tipo de entornos donde importa más el bajo consumo para que el dispositivo tenga más autonomía que la velocidad de transmisión.

A continuación se recogen en una tabla las características enfrentadas entre un Bluetooth clásico tipo y BLE

Especificación	BT clásico	BLE
Frecuencia de uso	2.4GHz	2.4GHz
Rango de distancia	100m	>100m
Tasa	1 – 3Mbps	125kbps – 1Mbps
Througput de aplicación	0.7 – 2.1Mbps	300kbps
Nodos esclavos activos	7	ilimitado
Latencia (desde la no conexión a envío de datos)	100+ ms	<6ms
Tiempo mínimo total para enviar datos	100ms	3ms
Voz permitida	Sí	No
Consumo de potencia	1W (normalizado)	0.01 – 0.5W (dependiendo aplicación)
Pico de consumo	<30mA	<15mA

Tabla 2: Bluetooth clásico vs BLE [8]

Por tanto, revisando las especificaciones, se podrías determinar que ambas opciones son buenas. BLE encaja mejor porque no existe necesidad de transmitir grandes volúmenes de información y haría que la autonomía del robot fuera mayor. Sin embargo, los equipos con los que se ha trabajado, no son compatibles con el BT4.0 por lo que se han conectado los equipos usando el estándar del Bluetooth clásico anterior a la versión 4.0.

3.1.2 Otras alternativas inalámbricas

- Wifi: existen módulos Wifi para Arduino. Con ellos, se consiguen velocidades mucho mayores y se pueden construir redes de dispositivos. Para el propósito de este TFG no son necesarias ninguna de las características que nos puede ofrecer. Además el consumo de Wifi es mucho mayor al del BT, con lo que se descartaría esta opción por ser menos adecuada.
- NFC: esta tecnología queda descartada por la distancia de operación (<20cm) que no es compatible con las necesidades del proyecto.
- ZigBee: se trata de una tecnología de bajo consumo y no alta velocidad de transferencia de datos muy similar a BLE y con distancias indoor de hasta 20m. Además tiene la ventaja de que puede ser configurado punto a multipunto [14]. El inconveniente reside en que la mayoría de dispositivos smartphones y ordenadores no están equipados para soportar esta tecnología, por lo que queda descartado en este proyecto, pero sí se considera buena alternativa para otro tipo de trabajos.

Tras analizar las alternativas más conocidas del mercado se determina que Bluetooth es la mejor solución de interconexión inalámbrica para este proyecto. Además, el robot ya viene equipado con un módulo por lo que no es necesario adquirir material extra que aumentase el presupuesto.

3.2 Sensórica

Según la RAE, un sensor es un "dispositivo que detecta una determinada acción externa, temperatura, presión, etc., y la transmite adecuadamente".

Se explican en este apartado los elementos de este tipo que se integran en el robot y algunos de los otros más extendidos que se pudieran integrar en mejoras o ampliaciones del robot.

3.2.1 Sensores integrados en la placa

- Sensor de sonido: se trata de un sensor que se fundamenta en un micrófono de condensador electret cuyo principal componente es un amplificador de baja potencia LM386 de Texas Instruments.
  Con él se puede encender o apagar el robot con, por ejemplo, la voz, de la misma forma que si fuera un interruptor.

- 2 sensores de luminosidad: se basan en transistores fotoeléctricos cuya resistencia disminuye a medida que aumenta la intensidad luminosa. Por tanto, propician que las variaciones de luz puedan ser convertidas en variaciones de señal eléctrica analógica.

Este tipo de componentes puede ser de gran utilidad en un amplio número de campos en donde se requiera saber el nivel de luz existente para, por ejemplo, encender o no luminaria. Así, dentro del concepto de “Smart Cities” este tipo de elementos ayudan a controlar el apagado o encendido del alumbrado público en función de la luz ambiente (y otros factores como la presencia de gente), ayudando a ahorrar energía (lo que supone también ahorro económico) y mejorando el descanso y bienestar de los vecinos.

- Sensor de temperatura: pequeño termómetro que se fundamenta en un termistor. Un termistor (*Thermally Sensitive Resistor*) es un semiconductor que varía se resistencia con la temperatura. Algunas de las funciones que se pueden implementar según su entrada pueden ser:
  - Activar o desactivar la climatización o aire acondicionado en un recinto en función de la temperatura que se esté midiendo. Esto implica un aumento de confort en la gente y control a tiempo real en la energía que gastan esos aparatos.
  - El control de la temperatura es fundamental en determinados procesos industriales para asegurar el correcto funcionamiento de los equipos, evitando por ejemplo sobrecalentamientos que puedan conllevar mal funcionamiento o acortar vida útil del equipamiento.

  El elemento que incluye la placa es un sensor de temperatura ambiente (que mide desde 0º a 50º con una precisión de ±2º), pero también los puede haber con contacto o sumergibles, que se comentarán más adelante.

- Acelerómetro de 3 ejes y sensor de giro: se fundamenta en un circuito integrado MPU-6050 [9] que combina un giroscopio de 3 ejes, un acelerómetro de 3 ejes y un procesador digital de movimientos patentado por el fabricante (InvenSense). Sus sensores internos miden las aceleraciones lineales y angulares, y su procesador es capaz de realizar cálculos en tiempo real proporcionando los ángulos de inclinación con respecto a los 3 ejes principales, que son de mayor utilidad.

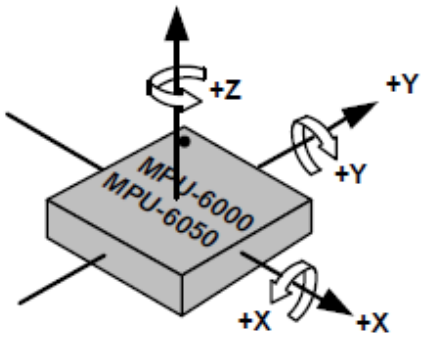


Figura 4: MPU-6050. Orientación de ejes

Un acelerómetro es un dispositivo que mide la aceleración de un elemento. Generalmente, el acelerómetro mide la aceleración mediante la medida de cuánto presiona la masa sobre algo cuando una fuerza actúa sobre ella.

Un giroscopio es un dispositivo de medida de la orientación, basado en los principios de conservación del momento angular (Coriolis).

Aplicaciones que se pueden dar según la entrada de este elemento pueden ser:

- Activar un airbag si se detecta un cambio muy brusco en la aceleración negativa de un vehículo.
- Girar una pantalla en función de en qué orientación esté posicionada.
- Suelen usarse como sistemas de posicionamiento inerciales (INS). Un INS es una ayuda a la navegación que usa un procesador y sensores de movimiento para seguir continuamente la posición, la orientación y la velocidad de un elemento sin necesidad de referencias externas. Se basa en el principio de que, si se conoce la posición inicial y se registran todos los movimientos, la posición en cada momento es conocida. Cabe comentar que en la actualidad la gran mayoría de *smartphones* incorporan estos sistemas. Como de por sí mismos no son muy precisos, suelen usarse como complemento a sistemas inalámbricos donde existen zonas de sombra en sistemas de posicionamiento interiores [10]

No entra dentro de los objetivos ni en el alcance de este proyecto el analizar sistemas de unidad de medición inercial (IMU) ni se usará este sensor de forma específica en el robot que se construye en el presente Proyecto.

3.2.2 Sensores equipados al robot conectados por RJ25

- Sensor de ultrasonidos: mide la distancia que hay hasta un objeto que está enfrente, con un rango de 3cm a 4m, en un ángulo de 30º y una precisión de 1cm. Se compone de un emisor y receptor de sonidos de

alta frecuencia no audibles por el oído humano. Su fundamenta en el tiempo que transcurre desde que se emite hasta que se recibe la señal, al ser conocida la velocidad del sonido en la atmósfera:

$$v = 343\frac{m}{s} \rightarrow 343\frac{m}{s} \cdot 100\frac{cm}{m} \cdot \frac{1s}{1000000\mu s} = \frac{1cm}{29.2\mu s}$$

Ecuación 1: Velocidad del sonido en cm/s

Por tanto, sabiendo que se tardan 29,2µs en recorrer cada centímetro, y teniendo en cuenta que la onda recorre dos veces la distancia al obstáculo (ida y vuelta):

$$v = \frac{D}{t} \rightarrow D(cm) = \frac{1}{2} \cdot \frac{t(\mu s)}{29.2\mu s}\, cm$$

Ecuación 2: Distancia sensor de ultrasonidos

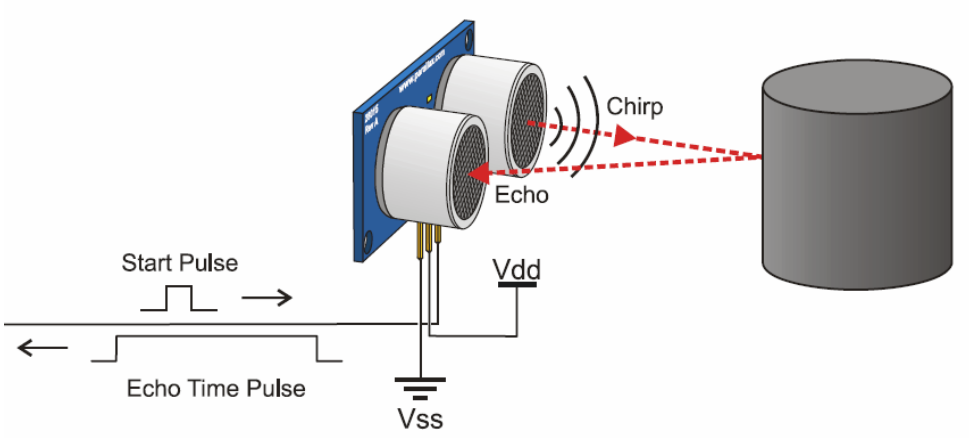


Figura 5: Sensor de ultrasonidos

Uno de los mayores inconvenientes de este tipo de dispositivos, como se muestra en la parte derecha de la Figura 6, es que no recibirán la onda reflejada si el objeto enfrentado forma un ángulo de incidencia con una desviación superior a la marcada en las especificaciones. Se podrían plantar dos soluciones para evitar este inconveniente:

- Montar dos sensores ultrasónicos, en lugar de uno, desplazados cierto ángulo sobre la dirección de la marcha del robot, para así cubrir más área.
- Montar un único sensor de ultrasonidos pero que mediante un servomotor fuera girando en abanico recorriendo el ángulo en el que pudiera haber objetos con los que pudiera chocar.

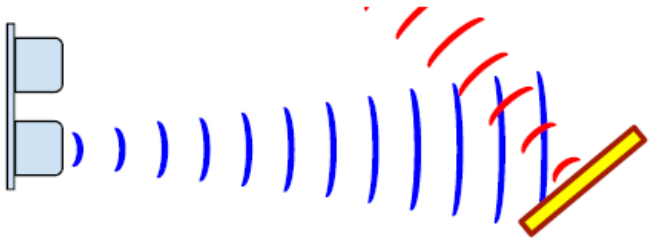


Figura 6: Limitaciones en el sensor de ultrasonidos

No es el objetivo de este proyecto que el robot evite todos los objetos, con lo que se asume que el robot pueda chocar en determinadas condiciones con determinados obstáculos.

A pesar también de que no son tan precisos en la medición de distancias, ni tienen la velocidad de respuesta de los sensores ópticos (estos por el contrario tienen un rango de medición más limitado y son más inmunes al polvo por ejemplo), su rango de exactitud de 1cm y su frecuencia de 42kHz es más que suficiente para el propósito del robot del presente proyecto.

- Sensor infrarrojo IR: en el robot construido funciona como un sensor de seguimiento de línea.
  Su funcionamiento es muy similar al del sensor ultrasónico, pero tiene un rango de detección menor, de 1-2cm. El dispositivo TCRT5000 [5] se compone por un diodo LED emisor de IR y un fototransistor sensible a los rayos IR a esa longitud de onda (λ=960nm).
  Si la superficie a la que está enfrentada es clara reflejará mayor cantidad de rayos IR que si es oscura, tal y como se muestra en la figura 7.

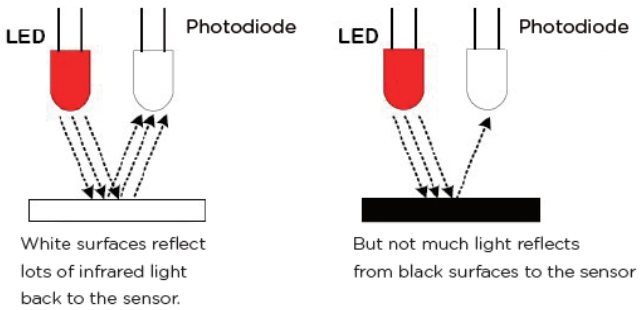


Figura 7: Sensor seguimiento de línea (LED+fotodiodo)

Colocando dos pares de estos sensores separados una determinada distancia, se puede verificar si el robot deja de seguir la línea marcada y, en consecuencia, programar para corregir ese desvío. Esta utilidad también puede servir por ejemplo para cortar una chapa, con un láser o una cuchilla, previamente marcada.

Además de las de seguimiento de línea, entre las distintas funciones que puede tener este tipo de sensores (o sensores ópticos, también pueden ser equipados con láseres) se pueden citar las de detectar

objetos enfrente de él como en los típicos detectores de los ascensores, o usarse como sensor de posición en encoders rotatorios [11].

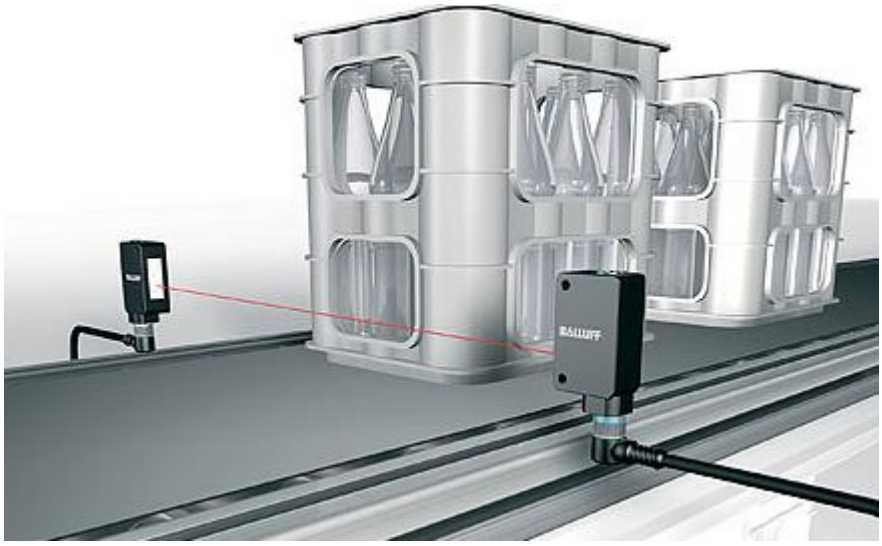


Figura 8: Sensor fotoeléctrico como detector de paso

Las barreras fotoeléctricas, muy extendidas como medidas de seguridad que pueden emplearse para parar la máquina cuando un operario entra en el área de operación de alguna máquina, están formadas por estos elementos.

3.2.3 Otros tipos de sensores en el mercado

En el mercado se pueden encontrar gran cantidad de sensores capaces de medir la gran mayoría de magnitudes físicas del entorno en el cual se instalan. Son muy frecuentes en sectores industriales, donde además de su precisión, los rangos de funcionamiento en los que pueden operar cobran vital importancia. Y es que en este tipo de escenarios se necesitan dispositivos de gran robustez frente a temperaturas extremas, productos químicos, polvos... Así pues debido a la importancia que están adquiriendo se citan algunos de los más utilizados tanto en la industria como en otros ámbitos:

- Sensor de temperatura sumergible: termómetros sumergibles en líquidos muy útiles para medir la temperatura de un líquido.
  Un dispositivo típico para Arduino o Raspberry es el basado en el termómetro digital DS18B20. Tiene una resolución ajustable de 9 a 12 bytes y solo requiere de 1 pin digital para la comunicación. Su rango de medición de temperaturas es de entre -55º y 125º con una precisión de 0,5º de -10º a 85º. Hay que prestar especial cuidado a que aunque se pueda medir ese rango, el cable que lo conecta con la placa también lo soporte.
- Sensor de humedad: detectan la humedad en el ambiente o en el entorno.

Una de las posibles aplicaciones que se le puede dar es la de la gestión de parques y jardines públicos, haciendo que el regado sea cuando realmente la tierra lo necesite y durante el tiempo justo. Cuando el sensor detecte la humedad adecuada de la tierra, los aspersores dejarán de regar el espacio. Estos sensores pueden ajustar el riego y así ahorrar un bien tan preciado como el agua.

- Sensor de llama: en determinados escenarios industriales es factible que determinadas máquinas puedan generar llamas, y por tanto, son elementos ampliamente usados en la industria como dispositivos de seguridad haciendo posible detener cualquier proceso en caso de detectar algún indico de combustión. La elección del dispositivo en la industria debe determinarse en función de la seguridad que se requiera, la sensibilidad y fiabilidad necesaria. Por ello, en entornos industriales se necesitarán sensores más profesionales que eviten falsas alarmas que puedan parar procesos (con el coste que eso conlleva) o no detecten rápidamente el fuego (con el coste y peligro que ello conlleva). Algunas de las especificaciones a tener en cuenta son (entre paréntesis valores típicos de en sensores Arduino):
    - Banda espectral que detecta (840-1200nm). Se debe asegurar que se cubren todos los tipos de fuegos que se pueden causar en el proceso.
    - Ángulo de detección (60º).
    - Tiempo de respuesta (15µs). Este tiempo de respuesta es vital en seguridad. Cuanto menor es el tiempo más rápido se detecta y mejor puede ser la respuesta.
    - Temperatura de operación (-25º a 80º y se recomienda no estar muy cerca de la llama para evitar deterioros). Este rango de operaciones puede no ser suficiente en industrias con elevadas temperaturas, como pueden ser fundiciones.
- Sensor de gas: de la misma forma que una llama, la detección de determinados gases puede ser vital para la seguridad en entornos industriales y domésticos. Existen diferentes sensores de este tipo que generalmente dan como salida una señal analógica. Su sensibilidad puede ser modificada en función de las necesidades.
  Cada dispositivo puede detectar ciertos tipos de gases. Dentro de la familia de sensores de gas MQ existen diferentes tipos dependiendo del gas al que sean sensibles: MQ-2 (metano, butano, LPG, humo), MQ-3 (alcohol, etanol, humo), MQ-4 (metano, gas CNG), MQ-5 (gas natural, LPG), MQ-6 (LPG, gas butano), MQ-7 (monóxido de carbono), MQ-8 (gas hidrogeno), MQ-9 (monóxido de carbono, gases inflamables), MQ131 (ozono), MQ135 (benceno, alcohol, humo), MQ136 (gas de sulfuro de hidrógeno), MQ137 (amoniaco), MQ138 (benceno, tolueno, alcohol, acetona, propano, gas formaldehído, gas hidrógeno), MQ214 (metano, gas natural), MQ216 (gas natural, gas de carbón), MQ303A (alcohol, etanol, humo), MQ306A (GLP, gas butano), MQ307A (monóxido de carbono), MQ309A (monóxido de carbono, gases inflamables.), MG811 (dióxido de carbono CO_2),

AQ-104 (calidad del aire), AQ-2 (gases inflamables, humo), AQ-3 (alcohol, benzina), AQ-7 (monóxido de carbono).

- Sensor de movimiento PIR: se trata de un sensor de movimiento que mide la luz infrarroja radiada por los objetos situados en su campo de visión.
  Todos los seres vivos y la mayoría de máquinas eléctricas o mecánicas que se mueven desprenden calor, y ese calor se emite a la atmósfera en forma de radiación infrarroja. La variación en el tiempo y/o espacio de esa radiación puede ser detectada por los sensores PIR dando una salida.
  Características de estos elementos que se deben tener en cuenta a la hora de diseñar un sistema dependiendo del uso que se le quiera dar y el entorno en donde se instalan, y que se podrán consultar en las especificaciones, son (con valores típicos entre paréntesis para sensores PIR Arduino):
  - Tensión nominal (5V).
  - Temperatura de trabajo (-20º a 70º)
  - Niveles de tensión de salida (5V – 0V)
  - Tiempo de respuesta (2s). Puede que en sistemas más críticos se deba rebajar este tiempo.
  - Ángulo de detección (120º). Si fuera insuficiente se podrían instalar más sensores para que de esta forma abarquen un mayor ángulo: instalando dos con un determinado ángulo haría que se doblase ese ángulo de detección.
  - Distancia de detección máxima (6m).

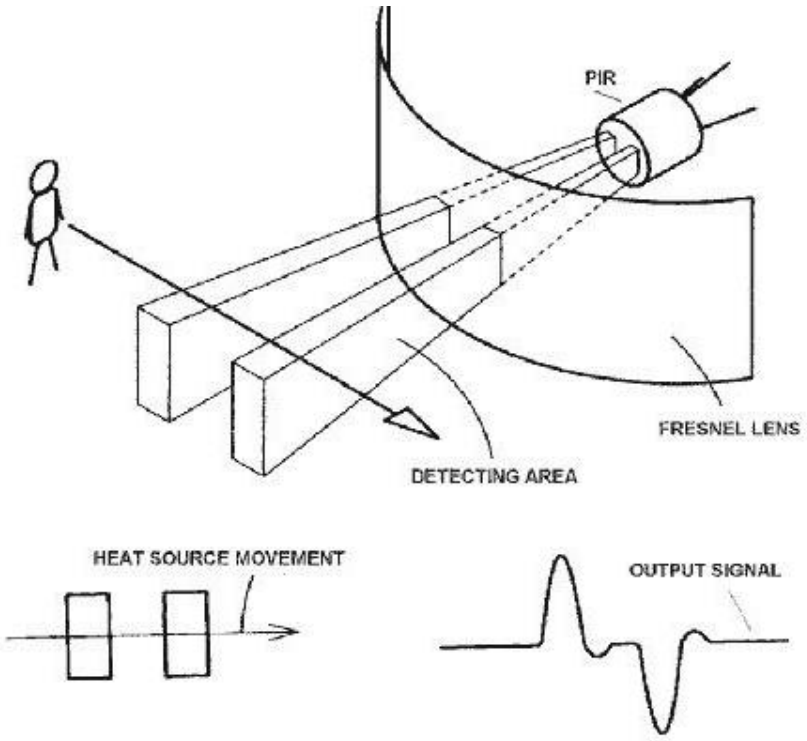


Figura 9: Sensor PIR. Principio de funcionamiento.

Aplicaciones típicas se dan en domótica, haciendo encender luces al paso de personas.

- Sensores inductivos: su funcionamiento se basa en la detección de elementos metálicos gracias al campo magnético que producen. Debido a este principio, son útiles para aplicaciones de posicionamiento como de detección de elementos formados mayoritariamente por hierro, tanto por contacto como sin contacto (aunque a distancias cortas).

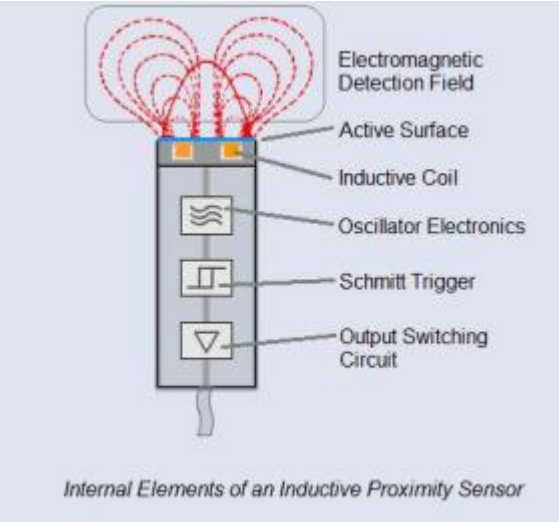


Figura 10: Sensor inductivo. Principio de funcionamiento.

- Sensores capacitivos: su principio de funcionamiento es muy similar al de los inductivos, pero usan un campo eléctrico para detectar objetos enfrentados a él. Debido a esta diferencia, estos dispositivos detectan tanto objetos metálicos como no metálicos.

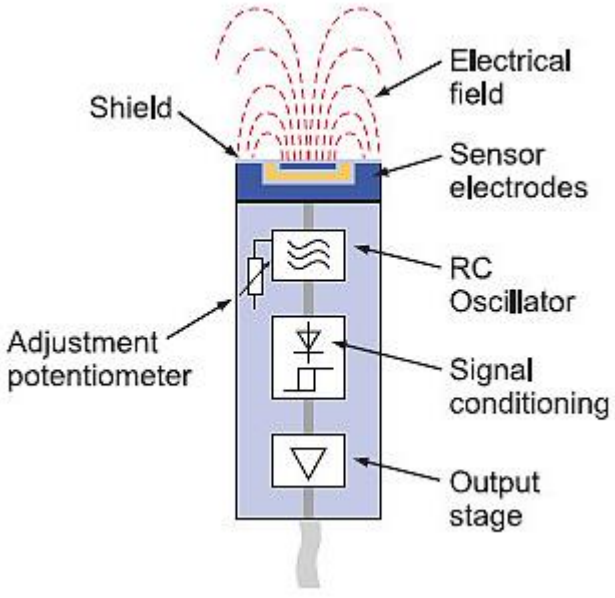


Figura 11: Sensor capacitivo. Principio de funcionamiento.

- Finales de carrera: en el mundo de la sensórica industrial tradicional, los finales de carrera son de los elementos más utilizados históricamente. Son interruptores que operan mediante accionamiento mecánico. Dan una señal dependiendo de si se producía contacto físico o no dependiendo la configuración en la que se diseñasen:
    - Normalmente abierto: cuando no se actúa sobre él (posición de reposo) el circuito está abierto.
    - Normalmente cerrado: cuando no se actúa sobre él (posición de reposo) el circuito está cerrado. Un caso típico de esta configuración son los paros de emergencia o setas de seguridad: abren el circuito si son pulsados para que se detenga todo el proceso con un reset.

  Dependiendo de la forma en que sean atacados por el objeto (puertas, compuertas...), pueden ser de diverso tipo y determinar presencia, ausencia, posicionamiento del elemento...

Figura 12: Final de carrera de palanca con rodillo

Un interruptor de contacto de palanca se podría implementar en las paredes del robot para detectar y hacer un conteo de cuántas veces se choca contra paredes u otro tipo de obstáculos. Uno de varilla podría utilizarse para detectar y contar, por ejemplo, postes (instalados como balizas físicas) que estuvieran en los costados del camino que recorre el robot.

Figura 13: Final de carrera de varilla

Quizá no sean elementos con mucha utilidad en un robot Arduino, pero se veía la necesidad de citar estos elementos tan presentes en sistemas de automatización.

3.3 Actuadores

3.3.1 Actuadores integrados en la placa

- Zumbador: es un *buzzer* pasivo que emite un sonido diferente en función de la frecuencia con la que se le haga vibrar. Que sea pasivo significa que no contiene un oscilador, con lo que la frecuencia se determina en función de la señal de salida que marque el controlador. Con este dispositivo se podrán generar diferentes tipos de señales acústicas.

- LED's RGB: la placa cuenta con un módulo de 12 LED's RGB. La particularidad de estos diodos LED es que se pueden regular cada uno de los tres colores primarios directamente y de forma independiente, dando mucha intensidad de luz y también un alto espectro de colores combinándolos.

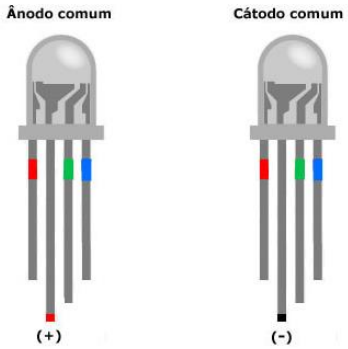


Figura 14: LED's RGB

De esta forma, sobre el mismo diodo se puede producir un código de colores que de forma visual pueda ser interpretado por un observador externo.

3.3.2 Actuadores equipados al robot

- Motores: el robot puede ser equipado con diversos tipos de motores como pueden ser motores paso a paso, motores DC, motores de encóder ópticos… El pack Ranger incluía motores de encóder ópticos de 3.7W que es con los que se ha equipado el robot.

3.3.3 Otros tipos de actuadores en el mercado

Existen gran número de actuadores y periféricos de salida que se pueden conectar al Arduino: diversos tipos de motores, pantallas LCD, displays numéricos, pantallas táctiles, pulsadores, interruptores, relés…

Al conectarse elementos, se debe tener en cuenta que la máxima corriente para cada pin de entrada/salida (I/O) es de 40mA y de 20mA de forma continua en las placas Arduino Mega 2560.

Además de a actuadores, los datos pueden enviarse a otros dispositivos para que sean procesados como pueden ser PC's, smartphones... o almacenados en ficheros, alimentar bases de datos...

3.4 Puertos de expansión

La placa ME Auriga cuenta con 10 puertos RJ25 y dos conectores 2.54mm "dupont" para la conexión de los motores.

La finalidad de los puertos RJ25 es facilitar la modularidad haciendo más cómodo el remplazo de los elementos que se conectan a la placa evitando los pines de las placas Arduino. Así, no es necesario conectar los cables hacia los pines de la placa, basta con conectar directamente el RJ25 de la misma forma que un teléfono fijo. En la figura 15 se muestra la relación entre los pinouts de las placas Arduino Mega y ME Auriga.

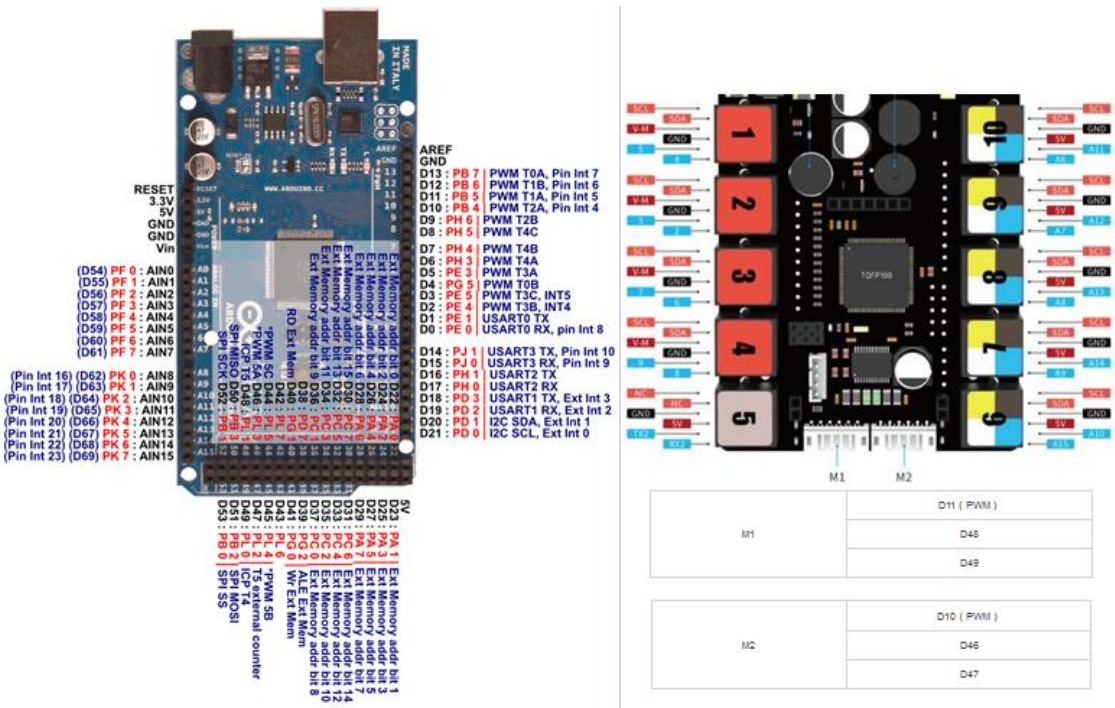


Figura 15: Arduino Mega (izq.) & ME Auriga (der.) pinout [5]

Además, a simple vista se puede conocer el uso que se puede dar a cada puerto según su color, sabiendo así de forma rápida qué dispositivo se debe conectar en qué puerto. La correspondencia se recoge en la tabla 3:

Puertos	Color	Función	Tipos de módulos compatibles
1-4	**Rojo**	Tensión de salida continua de 6-12V	Todo tipo de motores
5	**Gris**	Puerto serie	Bluetooth (BT) Módulo 2.4G Raspberry Pi. GPIO.
6-10	**Amarillo**	Interfaz digital single	Diversos sensores y actuadores analógicos y digitales
	Azul	Interfaz digital dual	
	Negro	Interfaz analógica single/dual	
	Blanco	Puerto I^2C	

Tabla 3: Puertos de expansión RJ25

- Los puertos del 1 al 4 pueden suministrar continuamente una corriente de salida de 3,5A (y picos de hasta 5A). Soportan protección a cortocircuitos y sobrecorrientes prolongadas de hasta 3,5A.
- Los puertos del 5 al 10 pueden suministrar continuamente una tensión de 5V y corriente de salida de 2,4A (y picos de hasta 3A). Soportan protección a cortocircuitos y sobrecorrientes prolongadas de hasta 3A.

3.5 Puerto serie

Existe un apartado de referencia sobre el puerto serie en Arduino que se encuentra en la web [3] para su consulta.

Un puerto es el nombre genérico con que se denomina a los interfaces físicos o virtuales que permiten la comunicación entre dos dispositivos. La comunicación serie es la que se usa generalmente para comunicar el Arduino con el ordenador u otro dispositivo mediante por ejemplo Bluetooth.

La comunicación serie en los pines TX/RX usa los niveles lógicos de referencia de TTL (*transistor-transistor logic*) que varían dependiendo de la placa desde los 3.3V a los 5V. Estos valores no son compatibles con los +/- 12V a los que opera el puerto serie RS232 con lo que no se deben conectar directamente a él para no dañar la placa.

Todos las placas Arduino tienen por lo menos un puerto serie (también conocido como UART o USART): *Serial*, que se comunica mediante los pines digitales 0 (RX) y 1 (TX) y a través del ordenador vía USB. Por tanto, hay que tener en cuenta que si se utilizan estas

funciones no se pueden usar también los pines 0 y 1 para conexiones de entradas y salidas digitales.

La placa Arduino Mega, que es en la que se basa la Me Auriga de este proyecto, cuenta con tres puertos serie adicionales: *Serial1* en los pines 19 (RX) y 18 (TX), *Serial2* en los pines 17 (RX) y 16 (TX) y *Serial3* en los pines 15 (RX) y 14 (TX). Estos puertos adicionales no están conectados al USB del Arduino.

Al emparejar el robot vía Bluetooth al ordenador, se puede observar en las opciones de Bluetooth que el sistema operativo asigna un par de puertos serie al robot para comunicarse con él:

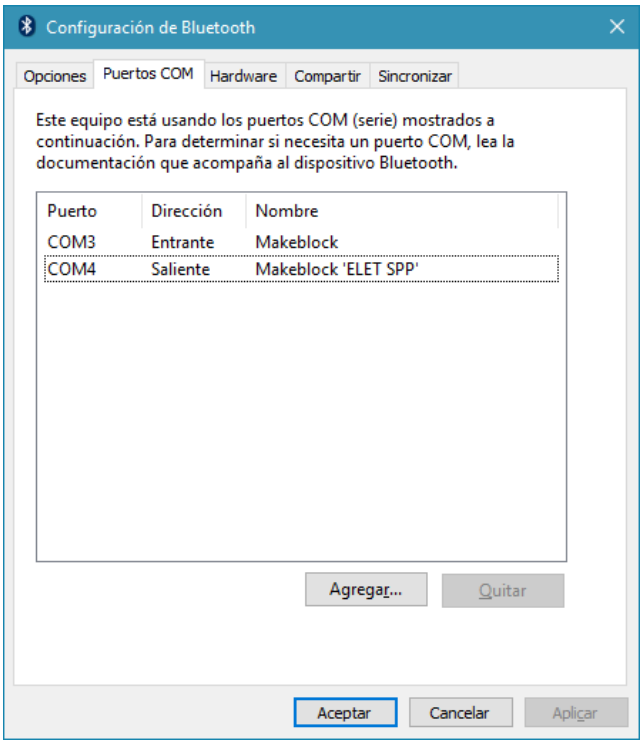


Figura 16: Puertos serie Bluetooth

En la siguiente imagen se puede observar en el IDE mBlock, que se analizará más adelante, al ordenador conectado al robot mediante el puerto COM4 que es el que nos había dado previamente el sistema operativo para esta conexión inalámbrica con el robot llamado *Makeblock*. Con el miniprograma se consigue conectar con el robot de forma inalámbrica y hacerlo mover para adelante presionando la tecla de flecha arriba del teclado. Además, como puede apreciarse, también se recogen de forma inalámbrica los datos que se envían desde un sensor inalámbrico de ultrasonidos configurado en el robot.

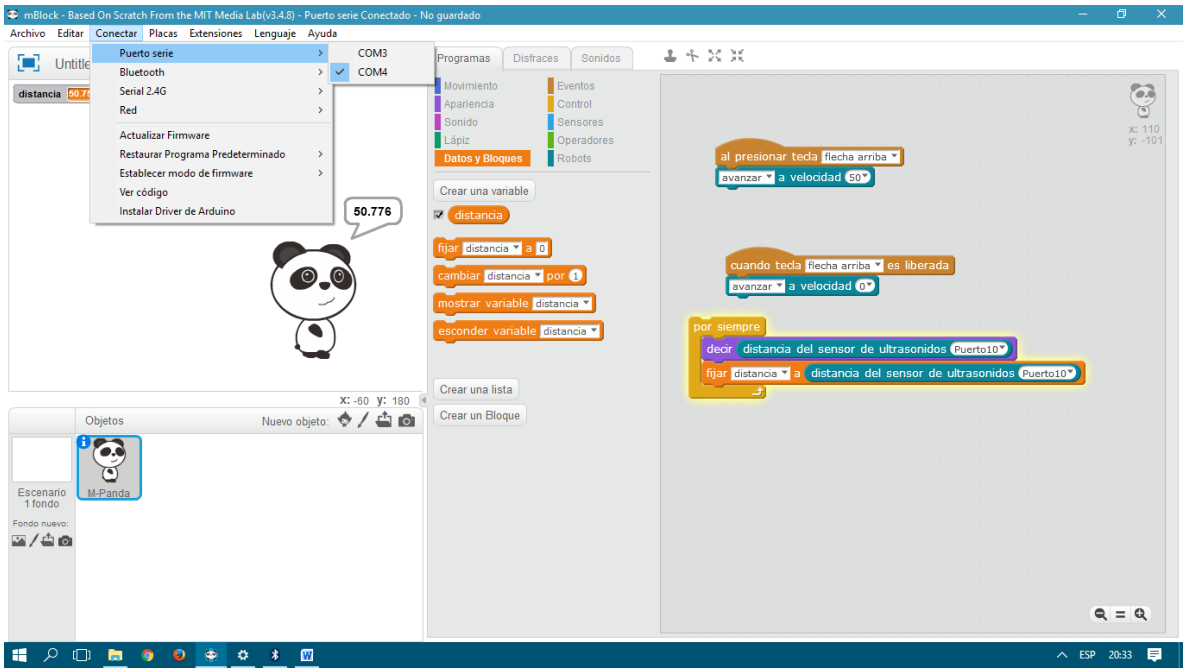


Figura 17: Conexión Bluetooth con el robot

El entono IDE de Arduino también cuenta con un monitor serie en donde se muestran los datos que se recogen desde el robot, y también se habilita una línea para enviar datos al robot. En la siguiente captura se recogen los valores del sensor de ultrasonidos en un código desarrollado durante la realización de este proyecto:

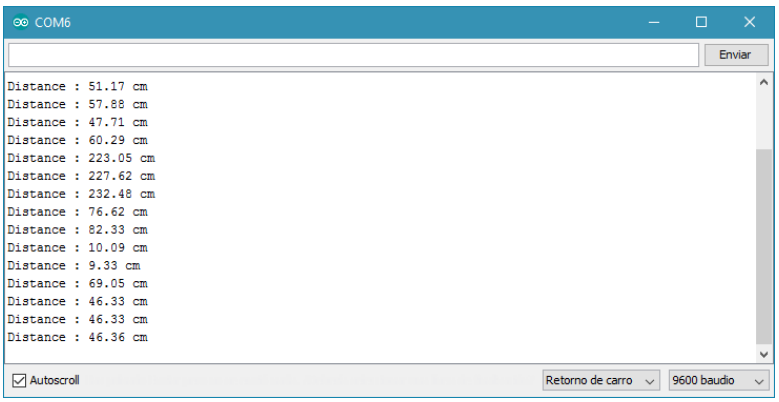


Figura 18: Monitor serie IDE Arduino

Además este IDE de Arduino también incorpora en sus últimas versiones un *Serial Plotter* donde se muestran los datos numéricos recogidos en formato de gráfica. Esto puede ser útil en determinadas ocasiones para determinar tendencias o ver evolución de las medidas registradas.

4. Entorno de programación

El fabricante dispone de un entorno de programación (IDE) propietario llamado *mBlock* basado en Scratch 2.0 y muy similar al propio Scratch donde se puede programar con este lenguaje. Además también se da la posibilidad de seleccionar un modo Arduino.

4.1 mBlock

Desde el entorno propietario se da la posibilidad de programar en Scratch, que es un lenguaje de programación visual abierto desarrollado por el *MIT (Massachusetts Institute of Technology) Media Lab*. Su codificación en bloques hace que sea muy intuitivo. Tanto es así que está principalmente orientado a la educación y enseñanza.

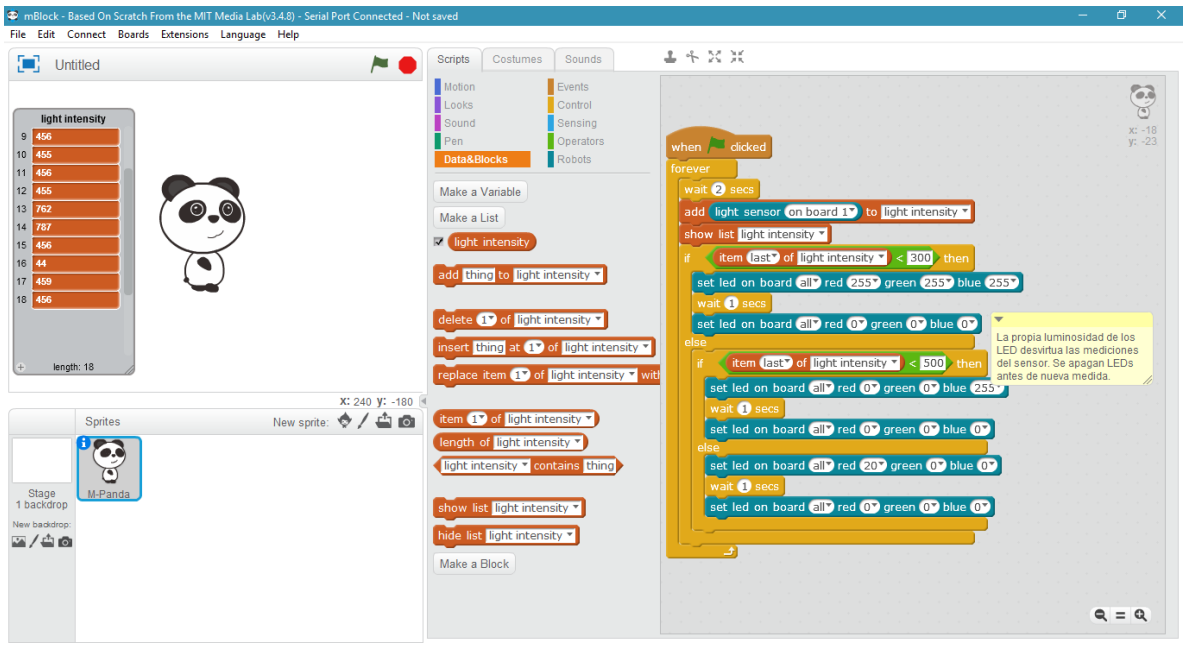


Figura 19: Programación Scratch en entorno mBlock

En la figura 19 se muestra un código en lenguaje Scratch que, en función de la intensidad luminosa recogida por el sensor de la placa, enciende todos los diodos de la matriz con máxima intensidad en color blanco (los 3 colores activos al máximo) si no hay mucha luz, enciende los LEDs en color azul con una intensidad luminosa ambiental, o los enciende con baja intensidad en rojo si se recibe mucha luz. Como comentario, se han tenido que apagar todos los diodos antes de que se realizasen las sucesivas mediciones para que su propia emisión de luz no desvirtuase la medición de la intensidad lumínica en ese momento. De no haberse realizado de esta manera, la medición del sensor siempre hubiera detectado una intensidad lumínica elevada debido a la propia luz que se emitía desde los diodos.

En el código de la figura 19 también se puede comprobar que las mediciones se han guardado en una lista donde se puede comprobar, de ser necesario, el registro de mediciones.

La programación en este entorno con Scratch únicamente ofrece la posibilidad de ejecutar el programa que se ha creado *online*, es decir, no se queda cargado en la placa Arduino. Las órdenes que se envían al robot y las entradas que se reciben de él se ejecutan directamente desde el programa. Pero también existe la opción de cargar el código en el hardware, seleccionando el "Modo Arduino". Una vez seleccionado este modo si existe código Scratch generado en IDE, el propio IDE hace una traducción de código a Arduino (lenguaje C) para poder ser subido a la placa. Para poder usarse de nuevo la opción *online*, si hubiera algo cargado en el microcontrolador, se debe actualizar el firmware del robot para que pueda entenderse con mBlock permitiendo la comunicación entre la placa Me Auriga y el ordenador dentro del software.

4.2 Makeblock

El fabricante también ha desarrollado una aplicación para dispositivos móviles smartphones compatible con Android 4.3 o superior, o el iOS 9.0 o superior de Apple, que pueden ser descargadas desde Google Play o iTunes respectivamente, ambas de forma gratuita.

Desde ella se pueden desarrollar proyectos para controlar el robot y recoger datos del mismo. En la siguiente figura se puede apreciar un proyecto implementado en el trascurso de este TFG en el que se han incluido botones para activar procesos, y elementos de visualización para printar entradas tomadas del bluetooth. Con ello se consigue desplazar al robot, hacer que se iluminasen LED's y que el altavoz emitiese una nota, y visualizar los datos recogidos por los sensores en forma numérica y de forma gráfica:

Figura 20: Proyecto en Makeblock para Android

Los diferentes tipos de bloques (visualizadores, paletas de printado, interruptores, pulsadores…) se pueden eliminar o incluir en el programa de forma muy intuitiva desde una paleta que está habilitada para ello. Para cada objeto asociado a un puerto, se puede seleccionar de forma muy visual en cuál se está conectado (en esta imagen el puerto 10):

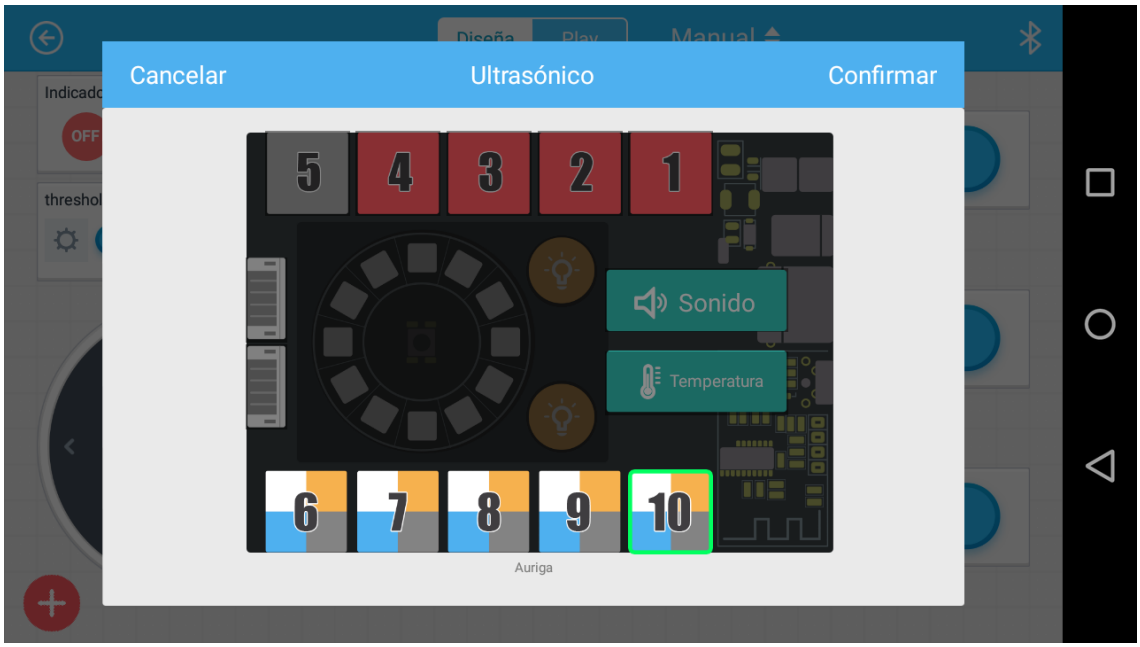


Figura 21: Selección visual de puerto en Makeblock Android

Además, seleccionando la opción de "Código" en cada objeto editable, éste se puede programar en un lenguaje basado en Scratch. Para ello, se habilita una barra de herramientas a la izquierda de la pantalla principal de donde se pueden arrastrar los módulos con los que construir el código.

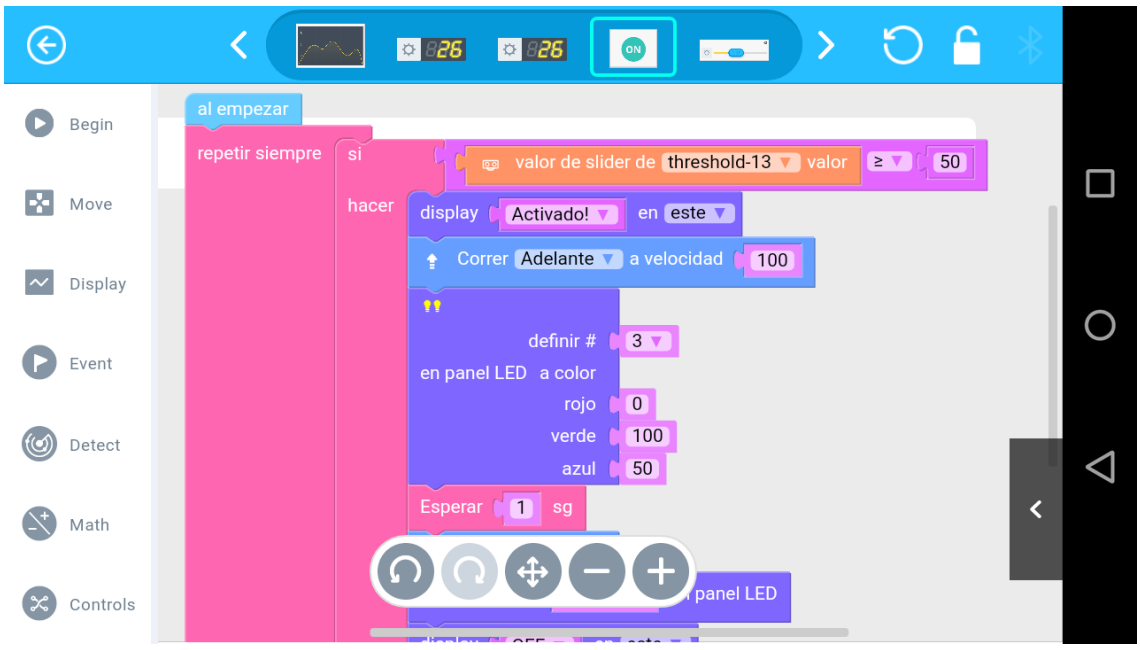


Figura 22: Código scratch asociado a un objeto

Se describen las opciones que hay en cada elemento de la barra:

- *Begin*: Se indica cuándo se ejecuta el código: Al empezar, cuando se presione una tecla, cuando se deslice la barra…
- *Move*: Acciones asociadas al movimiento del robot propio de la placa: avanzar, girar, parar… Con la velocidad a la que se indique. También es posible actuar sobre motores en otros puertos.
- *Display*: Ejecuta acciones sobre actuadores visuales o acústicos de los que se tendrá que definir el puerto en el que están y el valor que se les quiere dar.
- *Event*: Eventos que pueden generar acciones: se agita el Smartphone, el Smartphone se inclina hacia algún lado, se detecta un evento del tipo indicado en el puerto indicado, se activa interruptor o botón indicado en el proyecto en curso…
- *Detect*: Captura el valor de algún sensor del tipo que se le indique en el puerto que se le indique o el valor indicado en algún objeto que se haya incluido en el proyecto.
- *Math*: Operaciones matemáticas:
    - Sumas, restas, multiplicaciones…
    - Igual a, diferente, mayor, mayor igual…
    - AND, OR.
    - Negar
    - Números aleatorios
    - Redondear
    - Manipulación de elementos: añadir, establecer…
    - Operaciones trigonométricas
    - …
- *Controls*: Controles de flujo:
    - Condicionales: if, if else…
    - Bucles infinitos, bucles de repetición de X veces, repetir hasta, for…
    - Delays

De la misma forma que sucedía en mBlock, el código se ejecuta *online* sin ser cargado en la placa. Si se vuelve a encender el Arduino, este programa no estará grabado en él.

4.3 Arduino

Al ser un sistema basado en Arduino, también se puede programar la placa en este entorno de programación. Para preparar el entorno se deben seguir los siguientes pasos:

1. Descargar las librerías Arduino para los módulos Makeblock desde:
   https://github.com/Makeblock-official/Makeblock-Libraries
   Github es una plataforma de desarrollo colaborativo de software donde desarrolladores alojan software con el fin de su difusión y soporte a usuarios.

Además Github es un repositorio para el control de versiones con lo que periódicamente se cuelgan nuevas revisiones con mejoras o correcciones de *bugs*.

2. Copiar la carpeta “makeblock” a la ruta de librerías de Arduino. De esta forma la carpeta de librerías de Arduino debe tener la siguiente forma en Windows:
   [directorio de instalación de Arduino]\libraries\makeblock\src
3. Abrir el IDE Arduino.
4. Incluir la librería correspondiente a la placa que se vaya a utilizar en la parte superior del código. En nuestro proyecto se deberá incluir *#include <MeAuriga.h>*

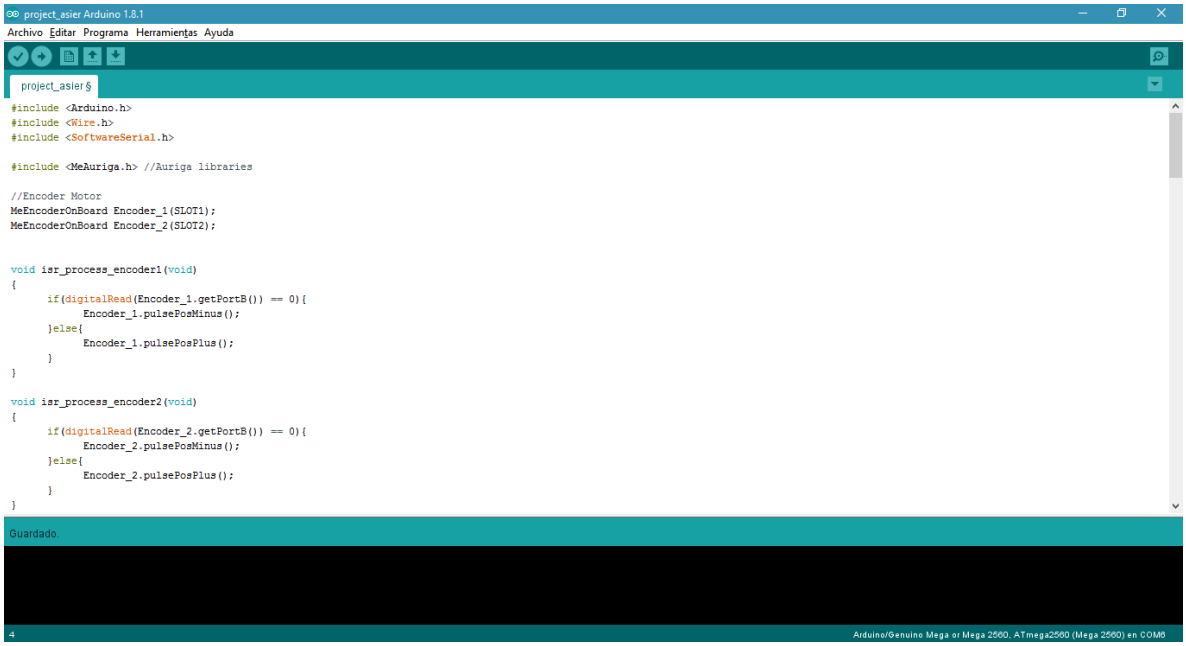


Figura 23: IDE Arduino

Arduino se programa en un lenguaje de programación derivado del lenguaje C. A continuación se pega el código cargado al robot para que trate de no chocarse con objetos, avise acústicamente cuando va marcha atrás, y encienda los LED’s de uno u otro color en función de la luz ambiente que detecte.

4.3.1 Código Arduino

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>

#include <MeAuriga.h> //Auriga libraries

//Encoder Motor
MeEncoderOnBoard Encoder_1(SLOT1);
MeEncoderOnBoard Encoder_2(SLOT2);
```

```
void isr_process_encoder1(void)
{
      if(digitalRead(Encoder_1.getPortB()) == 0){
            Encoder_1.pulsePosMinus();
      }else{
            Encoder_1.pulsePosPlus();
      }
}

void isr_process_encoder2(void)
{
      if(digitalRead(Encoder_2.getPortB()) == 0){
            Encoder_2.pulsePosMinus();
      }else{
            Encoder_2.pulsePosPlus();
      }
}

void move(int direction, int speed)
{
      int leftSpeed = 0;
      int rightSpeed = 0;
      if(direction == 1){
            leftSpeed = -speed;
            rightSpeed = speed;
      }else if(direction == 2){
            leftSpeed = speed;
            rightSpeed = -speed;
      }else if(direction == 3){
            leftSpeed = -speed;
            rightSpeed = -speed;
      }else if(direction == 4){
            leftSpeed = speed;
            rightSpeed = speed;
      }
      Encoder_1.setTarPWM(leftSpeed);
      Encoder_2.setTarPWM(rightSpeed);
}

void moveDegrees(int direction,long degrees, int speed_temp)
{
      speed_temp = abs(speed_temp);
      if(direction == 1)
      {
            Encoder_1.move(-degrees,(float)speed_temp);
            Encoder_2.move(degrees,(float)speed_temp);
      }
      else if(direction == 2)
      {
            Encoder_1.move(degrees,(float)speed_temp);
            Encoder_2.move(-degrees,(float)speed_temp);
      }
      else if(direction == 3)
```

```
      {
            Encoder_1.move(-degrees,(float)speed_temp);
            Encoder_2.move(-degrees,(float)speed_temp);
      }
      else if(direction == 4)
      {
            Encoder_1.move(degrees,(float)speed_temp);
            Encoder_2.move(degrees,(float)speed_temp);
      }
}

double angle_rad = PI/180.0;
double angle_deg = 180.0/PI;
void sensor_ultrasonidos();
MeUltrasonicSensor ultrasonic_10(10);
MeBuzzer buzzer;
void sensur_luz();
MeRGBLed rgbled_0(0, 12);
MeLightSensor lightsensor_12(12);

void sensor_ultrasonidos()
{
    if((ultrasonic_10.distanceCm()) > (50)){
        move(1,100);
    }else{
        if((ultrasonic_10.distanceCm()) < (20)){
            if(((random(1,(2)+1))==(1))){
                move(4,50);
            }else{
                move(3,50);
            }
        }else{
            move(2,50);
            buzzer.tone(262, 500);
            delay(20);
        }
    }
}

void sensur_luz()
{
    rgbled_0.setColor(0,0,0,0);
    rgbled_0.show();
    _delay(0.1);
    if((lightsensor_12.read()) < (300)){
        rgbled_0.setColor(0,255,255,255);
        rgbled_0.show();
        _delay(1);
    }else{
        if((lightsensor_12.read()) < (500)){
            rgbled_0.setColor(0,0,255,0);
            rgbled_0.show();
            _delay(1);
        }else{
```

```
            rgbled_0.setColor(0,20,0,0);
            rgbled_0.show();
            _delay(1);
        }
    }

}

void setup(){
    attachInterrupt(Encoder_1.getIntNum(), isr_process_encoder1,
RISING);
    attachInterrupt(Encoder_2.getIntNum(), isr_process_encoder2,
RISING);
    buzzer.setpin(45);
    rgbled_0.setpin(44);
    //Set Pwm 8KHz
    TCCR1A = _BV(WGM10);
    TCCR1B = _BV(CS11) | _BV(WGM12);
    TCCR2A = _BV(WGM21) | _BV(WGM20);
    TCCR2B = _BV(CS21);
    BT.begin(9600);
    Serial.begin(9600);

}

void loop(){
    sensur_luz();
    sensor_ultrasonidos();
    Serial.print("Distance : ");
    Serial.print(ultrasonic_10.distanceCm() );
    Serial.println(" cm");
    delay(100);   /*   the   minimal   measure   interval   is   100
milliseconds */
    _loop();
}

void _delay(float seconds){
    long endTime = millis() + seconds * 1000;
    while(millis() < endTime)_loop();
}

void _loop(){
    Encoder_1.loop();
    Encoder_2.loop();
}
```

A diferencia de lo que sucedía en los entornos mBlock y Makeblock, este código se carga en el microcontrolador. De esta forma, estará disponible cada vez que el Arduino sea encendido, cosa que no pasaba en los dos primeros, que ejecutaban el código *online*.

5. Obtención y procesamiento de datos

Los datos que se envían desde el Arduino pueden ser recogidos a través de un software de escaneo y captura de datos como RealTerm. Este programa permite leer los datos del puerto serie y capturarlos almacenándolos en un archivo.

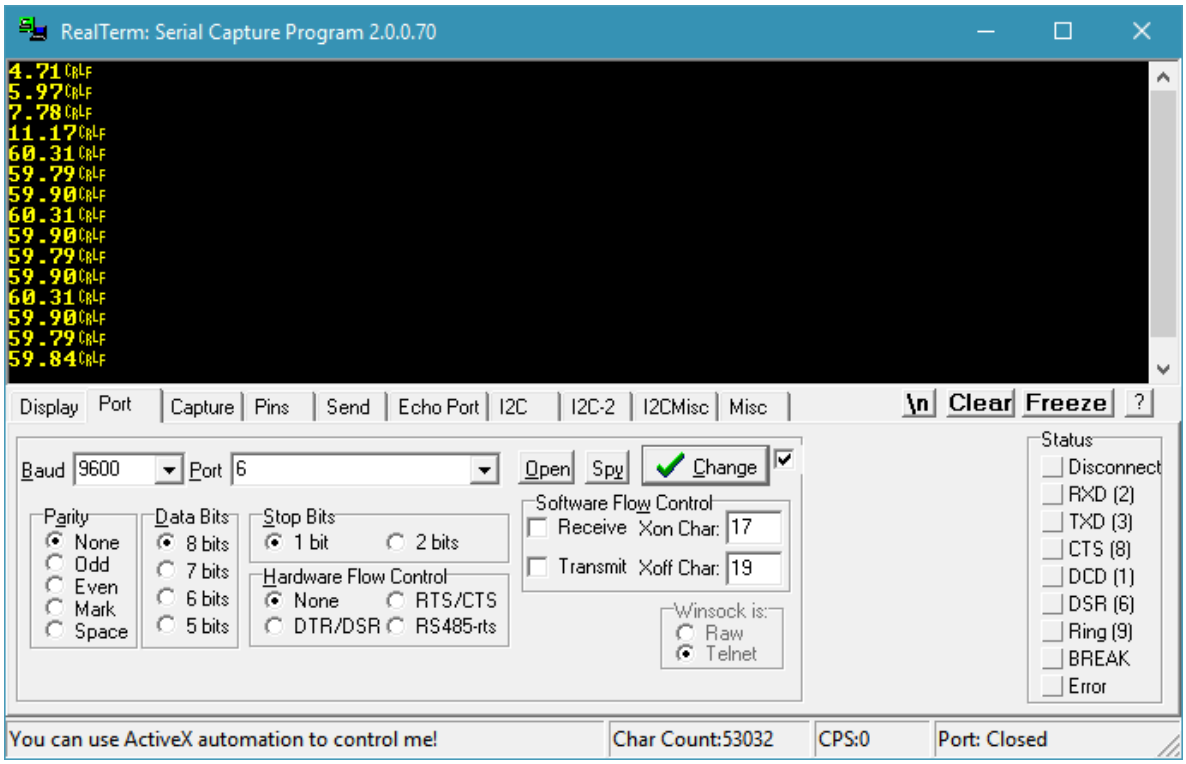


Figura 24: Programa de captura de tráfico serie RealTerm

Los datos previamente almacenados en un fichero (txt en este proyecto) son recogidos por un software de trazado y visualización de datos a tiempo real. De esta forma se consigue una monitorización instantánea de los parámetros que van reportando los sensores o el propio Arduino en sus actuadores.

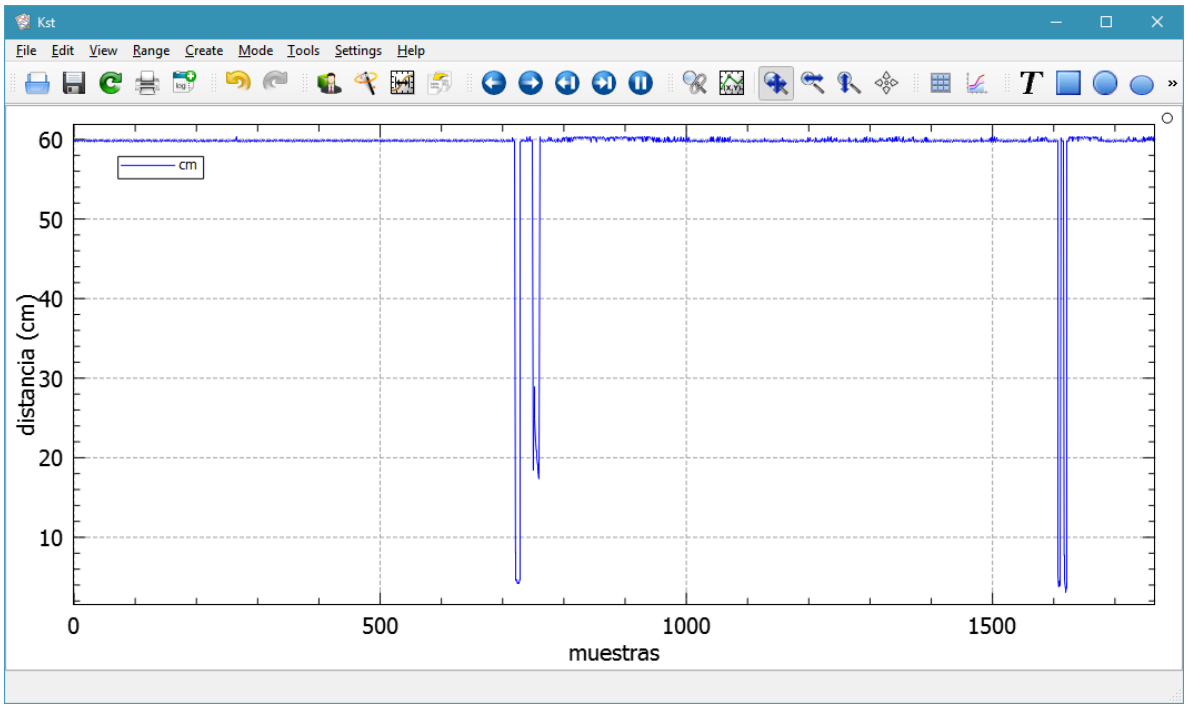


Figura 25: Software de trazado y visualización de datos Kst

Las funciones de captura y monitorización están también integradas en el IDE Arduino (o en el del propio mBlock), pero se testean otro tipo de soluciones para constatar la no dependencia del puerto serie con el IDE Arduino, y para comprobar también las posibilidades de procesamiento de los datos recogidos que proporciona esta recogida de datos.

6. Valoración económica del trabajo

La finalidad del trabajo no era económica, sino más bien de investigación y como punto de partida para el desarrollo de proyectos y sistemas de mayor alcance. En cualquier caso, se reflejan el presupuesto que se ha empleado para este TFG:

Producto	PVP
Robot Makeblock Ranger	169 €
Pilas AA (pack de 6)	4,95 €
IDE mBlock	0 €
IDE Arduino	0 €
PVP TOTAL	**173.95€**

Tabla 4: Presupuesto del Proyecto

Cabe destacar que se podrían haber conseguido los objetivos con menos inversión, pero hubiera supuesto mayor tiempo que no se hubiera dedicado al proyecto en sí.

7. Conclusiones

Con este trabajo se ha constatado las infinitas posibilidades que se presentan en el futuro dentro de los campos de la sensórica, la automatización y la robótica. Puede decirse que son los pilares fundamentales de las nuevas tecnologías con más presente y futuro de hoy en día. Existen multitud de tareas y procesos que se pueden automatizar con las herramientas que se disponen en la actualidad. Por tanto, es un mundo con mucho recorrido en los que los profesionales de las IT tienen mucho que aportar.

Por otra parte, con este proyecto se han conseguido de una manera o de otra los objetivos inicialmente marcados. En lugar de haberse implementado un código que englobase todas las funciones y depurarlo hasta dar con un programa a la medida, se ha optado por explorar más a fondo cada una de las alternativas que se disponían. En internet hay infinidad de código que se puede aprovechar para la aplicación concreta con la que se vaya a trabajar, por ello se ha decidido por conocer mejor las herramientas que se disponen para poder aprovechar la plataforma.

De haberse planeado de otra forma, se hubieran integrado en el robot nuevos módulos para completar las tareas y realizar otro tipo de pruebas y test. Hay diversas empresas que comercializan con bloques para Arduino y los pedidos llegan en pocos días. De haberse tenido más tiempo, se podrían haber presentado de mejor forma los datos recogidos del robot, desarrollando el apartado 5 más en profundidad. Se ha recogido en el documento la posibilidad de que existía, y del potencial que esto puede implicar, pero no se ha trabajado más en ello.

Como líneas de trabajo futuro se pueden mencionar:

- La presentación de los datos en algún sistema de bases de datos, procesándolos en Excel con algún tipo de macro…
- Controlar el robot mediante alguna plataforma de desarrollo de diseño de sistemas como puede ser LabView.
- Desarrollar un código más potente y depurado a cargar en el robot mediante el IDE Arduino.

Las líneas que se abren son muchas y en muchas de las ocasiones solo falta la idea de qué es lo que se pretende obtener, ya que las herramientas y el conocimiento en el mundo *maker* están al alcance de toda la comunidad.

8. Glosario

BLE: Bluetooth Low Energy
BT: Bluetooth
DIY: Do it yourself.
GPIO: General Purpose Input-Output
HW: hardware.
I/O: Input / Output
IDE: entorno de desarrollo integrado.
IMU: unidad de medición inercial, Inertial Measurement Unit en inglés.
INS: Inertial Navigation Systems
IoT: Internet de las Cosas, Internet of Things en inglés.
IR: infrarrojo, infrared en inglés.
IT: tecnologías de la información.
LED: Light-Emitting Diode en inglés.
MIT: Massachusetts Institute of Technology
NFC: Near field communication
PIR: Passive InfraRed
RAE: Real Academia Española
RFID: Radio Frequency IDentification
RX: Recepción
TTL: Transistor-Transistor logic
TX: Transmisión.
UART: Universal Asynchronous Receiver-Transmitter
USB: Universal Serial Bus

9. Bibliografía

[1] https://www.arduino.cc/ [Marzo 2017]

[2] https://create.arduino.cc/ [Marzo 2017]

[3] https://www.arduino.cc/en/Reference/Serial [Marzo 2017]

[4] http://learn.makeblock.com/en/ [Marzo 2017]

[5] http://learn.makeblock.com/en/me-auriga/ [Marzo 2017]

[6] **Datasheet**. Reflective Optical Sensor with Transistor Output TCRT5000: http://www.vishay.com/docs/83760/tcrt5000.pdf

[7] **Nieminen, J. y otros autores.** *IPv6 over BLUETOOTH(R) Low Energy*. Internet Engineering Task Force (IETF). Universitat Politecnica de Catalunya/i2CAT; Octibre 2015. Disponible en: https://tools.ietf.org/pdf/rfc7668.pdf

[8] https://es.wikipedia.org/wiki/Bluetooth [Mayo 2017]

[9] https://en.wikipedia.org/wiki/Bluetooth_Low_Energy [Mayo 2017]

[10] **Datasheet**. MPU-6050 Six-Axis (Gyro + Accelerometer) MEMS MotionTracking™ Devices. Disponible en: https://www.invensense.com/products/motion-tracking/6-axis/mpu-6050/

[11] **Rovira Jofre, J.** *Geotelemática. Posicionamiento y navegación.* Barcelona: Universitat Oberta de Catalunya; 2012.

[12] **Castaño Sánchez, M**. *Design of a Position Encoder Using Infrared Sensors.* Universitat Politècnica de Catalunya, Master Thesis; Mayo 2015.

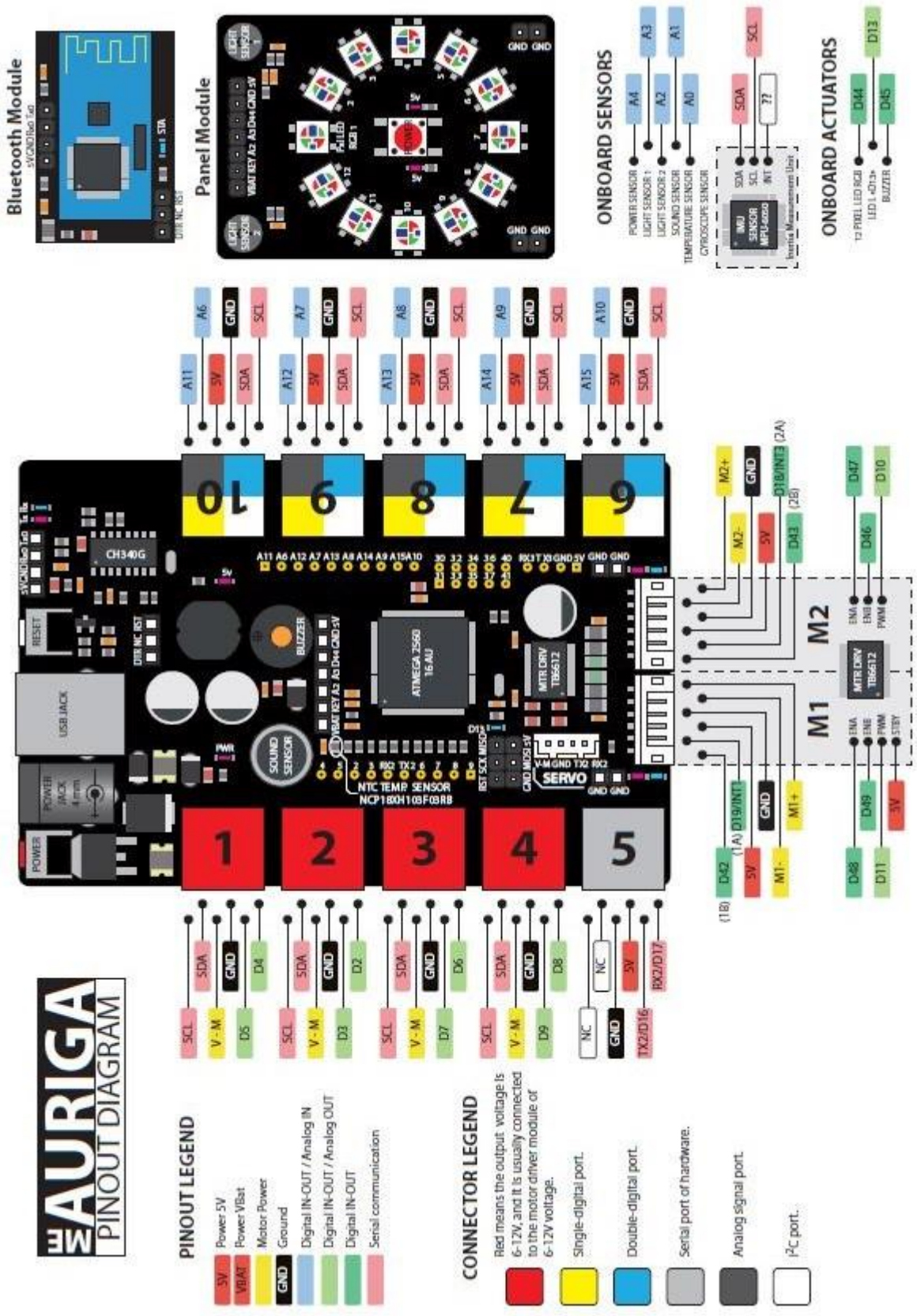
[13] **Benzi, M.** *Getting Started with Arduino*. 3ª ed. Editorial O'Reilly. Sebastopol, CA (USA); Septiembre 2012.

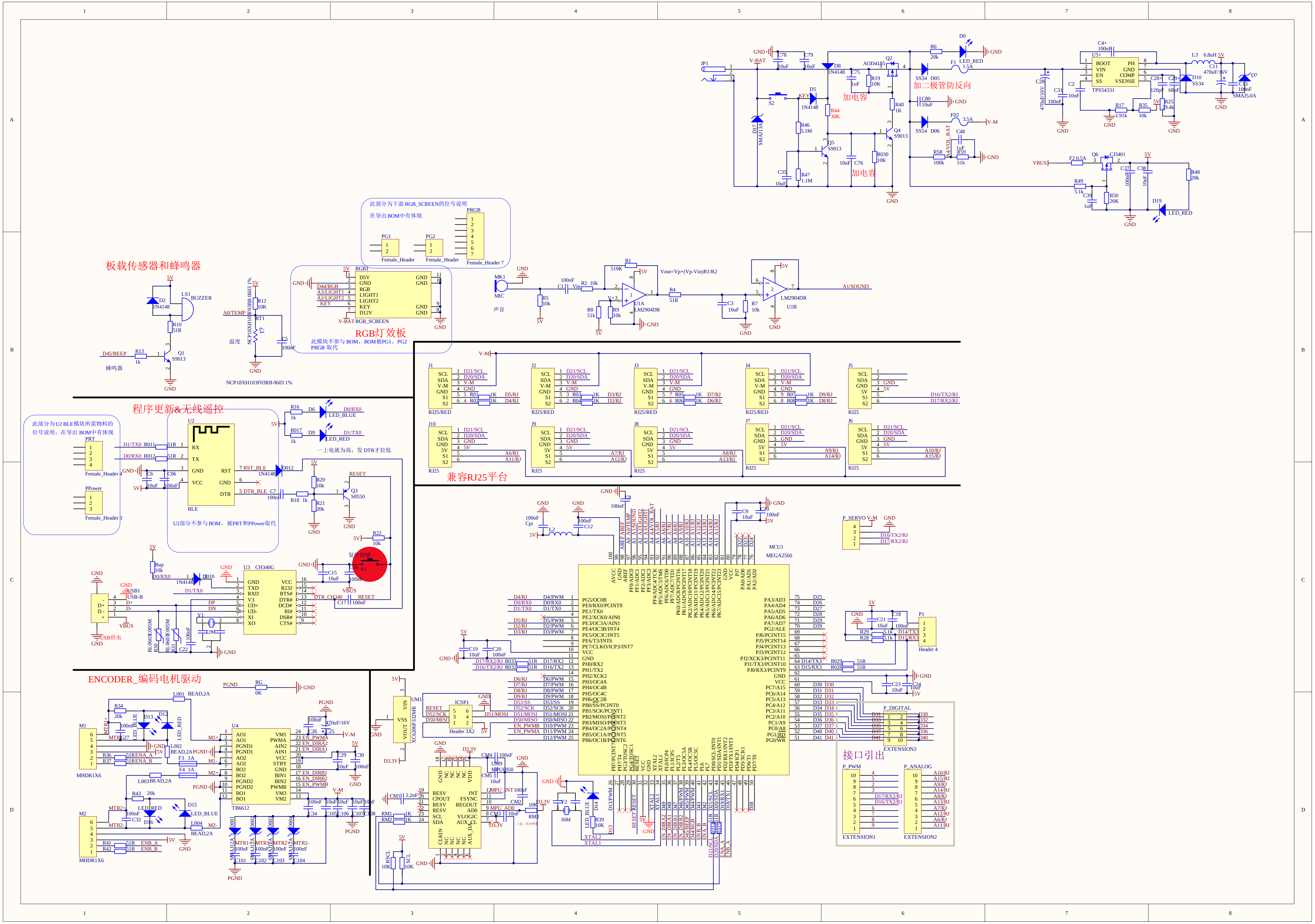
[14] **Evans, B.** *Arduino Programming Notebook*. 1ª ed. San Francisco, CA (USA); Agosto 2007.

[15] https://blogs.deusto.es/aplicaciones-tic/zigbee-vs-bluetooth-low-energy/ [Abril 2017]

[16] https://www.raspberrypi.org/ [Abril 2017]

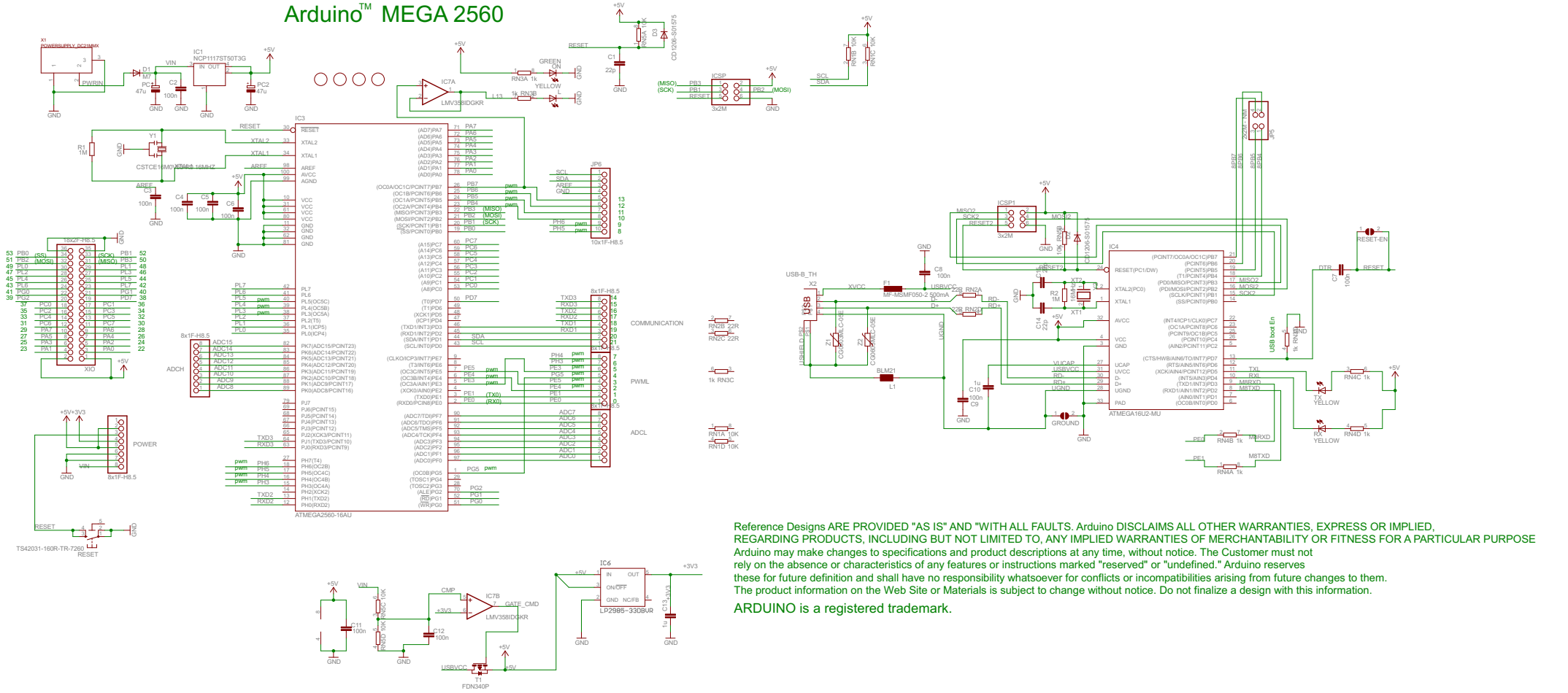
10. Anexos


Bluetooth Module
Panel Module
ONBOARD SENSORS
ONBOARD ACTUATORS
POWER SENSOR
LIGHT SENSOR 1
LIGHT SENSOR 2
SOUND SENSOR
TEMPERATURE SENSOR
GYROSCOPE SENSOR
BUZZER
M1
M2
SERVO
NTC TEMP. SENSOR
NCP18XH103F03RB
ATMEGA 2560
16 AU
MTR DRV
TB6612
CH340G
USB JACK
POWER JACK
SOUND SENSOR
BUZZER
RESET
PWR
AURIGA
PINOUT DIAGRAM
PINOUT LEGEND
Power 5V
Power VBat
Motor Power
Ground
Digital IN-OUT / Analog IN
Digital IN-OUT / Analog OUT
Digital IN-OUT
Serial communication
CONNECTOR LEGEND
Red means the output voltage is 6-12V, and it is usually connected to the motor driver module of 6-12V voltage.
Single-digital port.
Double-digital port.
Serial port of hardware.
Analog signal port.
I²C port.



板载传感器和蜂鸣器
BUZZER
蜂鸣器
温度
NCP18XH103F03RB 0603 1%
此部分为下面 RGB_SCREEN的位号说明
在导出 BOM中有体现
RGB灯效板
此模块不参与 BOM，BOM被PG1、PG2
PRGB 取代
声音
Vout=Vp+(Vp-Vin)R1/R2
LM2904DR
加二极管防反向
加电容
TPS54331
程序更新&无线遥控
此部分为U2 BLE模块所需物料的
位号说明，在导出 BOM中有体现
U2部分不参与BOM，被PRT和PPower取代
一上电就为高，发 DTR才拉低
复位按键
CH340G
USB供电
兼容RJ25平台
MCU1
MEGA2560
ENCODER_编码电机驱动
TB6612
XC6206P332MR
MPU6050
接口引出
EXTENSION1
EXTENSION2
EXTENSION3

Arduino™ MEGA 2560



Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS. Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE Arduino may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Arduino reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

ARDUINO is a registered trademark.

THE DEFINITIVE ARDUINO MEGA PINOUT DIAGRAM