



UNIVERSIDAD DE LA CORUÑA

ESCUELA POLITÉCNICA SUPERIOR. FERROL

TRABAJO FINAL DE GRADO



GRADO EN INGENIERÍA MECÁNICA

Título:

**DISEÑO Y ADAPTACIÓN DE UN SISTEMA DE
SENSORIZACIÓN INFRARROJA EN UNA PLATAFORMA
ROBÓTICA MÓVIL.**

Autor:

ALBA LEMA MARTÍNEZ.

Tutores:

D. FRANCISCO BELLAS BOUZA.

D. MARTÍN NAYA VARELA.

Fecha:

FEBRERO 2016

MEMORIA

Tabla de contenido

1. Antecedentes	1
2. Objetivos	13
3. Abreviaturas	14
4. Programas empleados	16
5. Requisitos previos	17
6. Elección del sensor infrarrojo	19
6.1. Fundamentos teóricos. Sensores infrarrojos	19
6.2. Selección de sensores	20
6.3. Sensor VCNL4040.....	21
6.4. Características del sensor infrarrojo	26
7. Diseño y fabricación del circuito integrado de los sensores infrarrojos	35
8. Ubicación de los sensores en la carcasa.....	42
8.1. Elección de número de sensores	42
8.2. Hueco en la carcasa para la salida y recepción de ondas del sensor.....	48
9. Elección del microcontrolador	52
9.1. Fundamentos teóricos. Microcontroladores.....	52
9.2. Los microcontroladores PIC	53
9.3. Microcontrolador PIC32MX534F064H	58
10. Programación del microcontrolador	60
10.1. Fundamentos teóricos. Comunicación I2C.....	60
10.2. El protocolo software para I2C.....	64
10.3. Comunicación con el sensor VCNL4040	68
11. Elección de multiplexor.....	70
11.1. Fundamentos teóricos. Multiplexores.....	70
11.2. Selección de multiplexores	71
11.3. Multiplexor	72
11.4. Diseño y fabricación del circuito integrado del multiplexor	75
11.5. Comunicación con el multiplexor.....	76
12. Resultados	79
13. Conclusiones.....	82
14. Bibliografía	83
15. ANEXO.....	86

1. Antecedentes

En este capítulo haremos una breve introducción al campo de los robots móviles autónomos utilizados en investigación. El objetivo es proporcionar una visión actualizada del mismo, y enmarcar el desarrollo realizado en este Trabajo Fin de Grado. Para ello, lo primero que debemos definir es qué es la robótica: es una rama de la tecnología que estudia el diseño y construcción de máquinas capaces de desempeñar tareas repetitivas, tareas en las que se necesita una alta precisión, tareas peligrosas para el ser humano o tareas irrealizables sin intervención de una máquina.

La palabra robot deriva de ‘robota’, que aparece por primera vez en la obra literaria R.U.R (“Rossum's Universal Robots” que traducido significa “Robots Universales de Rossum”) de 1920 escrita por el dramaturgo checo Karel Capek. La palabra checa ‘robotá’ significa ‘servidumbre’ o ‘trabajado forzado’ y cuando se tradujo al inglés se convirtió en robot. Aunque el propio inventor de ésta palabra le atribuye el mérito a su hermano Josef Capek, en una carta dirigida en 1933 a la editorial del Diccionario Oxford. Sin embargo, el acuñamiento de la palabra robótica se le debe a otro escritor, Isaac Asimov, quien la hizo famosa con su aparición en su novela Runaround en 1941. Un robot se define como una entidad hecha por el hombre con un cuerpo (anatomía) y una conexión de retroalimentación inteligente entre el sentido y la acción sin necesidad de control humano. Las acciones de los robots son generalmente llevadas a cabo por motores o actuadores que mueven extremidades o impulsan al robot, y estas acciones son seleccionadas por un sistema de control en función de unas medidas sensoriales.

De entre los diferentes ámbitos de aplicación de la robótica, este trabajo se centra en los robots autónomos, concretamente en los utilizados para investigación. Los robots autónomos son aquellos que no requieren de intervención humana para la realización de sus tareas. Esto implica que el sistema de control del robot debe poseer ciertas propiedades “inteligentes” tales como la adaptación a los cambios, el aprendizaje, etc. Este tipo de robots se desarrollan en universidades y cada vez más son adoptados por la industria. En las universidades es donde se desarrollan, fundamentalmente, los sistemas de control inteligentes que les dotan de autonomía. Los robots utilizados para investigación en este ámbito tienen una serie de características singulares: su precio debe ser asequible, deben ser de tamaño reducido para poder ser

utilizados en entornos de laboratorio, deben ser fácilmente programables y deben poseer numerosos sensores y actuadores para enriquecer así el proceso investigador. En este campo de aplicación los robots móviles son los más utilizados ya que son capaces de desplazarse por el entorno. Existen diversos tipos: rodados, con patas, aéreos, marítimos, etc.

La revolución en el campo de los robots móviles autónomos para investigación se produjo en 1999 cuando SONY introdujo en el mercado su robot-mascota *AIBO*¹ (ver Figura 1). *AIBO* fue un robot-mascota con forma de perro, y en el momento de su salida al mercado constituyó uno de los robots para investigación más vendidos, con una relación entre calidad de sensores y actuadores y precio sin competencia por aquel entonces.

Este robot ha sido utilizado para la investigación en inteligencia artificial tanto aisladamente como en conjunto con otros robots, en todo tipo de pruebas de inteligencia cooperativa. En concreto, *AIBO* ha sido presentado varias veces a las competiciones Robo Cup, en concreto la RoboCupSoccer, en la que varios robots *AIBO* colaboran entre ellos para jugar un partido de fútbol. Pese a que en sus inicios *AIBO* se vendía al público, en el año 2006, dejó de comercializarse.



Figura 1. *Robot AIBO.*

Siguiendo la estela creada por el *AIBO* en cuanto a calidad y estética, un robot autónomo para investigación que está teniendo mucho auge actualmente es el robot humanoide *NAO*² (figura 2) de 58 centímetros, interactivo, totalmente programable y en constante evolución. Nació en 2008, desarrollado por la empresa Aldebarán Robotics, y ha pasado ya por 5 versiones hasta llegar al modelo actual evolution v5.



Figura 2. *Robot NAO.*

Con su software nativo, el robot NAO es capaz de interactuar de forma natural, con todo tipo de público, escuchándolo, hablando o yendo hacia las personas. La complejidad de sus movimientos y acciones le permite jugar un partido de fútbol, hacer de profesor, o promocionar un producto en un evento interactuando con los asistentes o realizando complejas coreografías, entre otras muchas actividades.

NAO es capaz de percibir el entorno a partir de sus múltiples sensores, entre los cuáles se encuentran dos cámaras, cuatro micrófonos, nueve sensores táctiles, dos sensores de ultrasonidos, 8 sensores de presión, un acelerómetro y un giróscopo. Además, incluye otros elementos de expresión que le dan un alto grado de interactividad, como sus 53 LEDs RGB, su sintetizador de voz y sus dos altavoces.

Incluye un software gráfico de programación llamado Choreographe, compatible con Windows Linux y Mac, que permite programarlo sin tener conocimientos de un lenguaje programación. Y para usuarios avanzados incluye un conjunto completo para desarrollo de software, que permite usar distintos lenguajes como C++, Python, JAVA, .NET y MATLAB.



Figura 3. *Robot aspirador Roomba.*

No podemos dejar de mencionar el robot autónomo más vendido de la actualidad: el robot aspirador Roomba³ (figura 3), lanzado en 2002. La empresa iRobot ha lanzado ya tres generaciones distintas de este robot, y tiene una implantación real en muchos hogares, en los que empieza a convertirse en un electrodoméstico más. Está incluido en esta revisión porque existe una versión de Roomba para investigación, que incluye un software de desarrollo que permite programar al robot.



Figura 4. *Robot E-puck.*

Aparte de los robots anteriores que son un buen ejemplo de las tendencias actuales, los robots móviles que más se han utilizado en investigación han sido de pequeño tamaño y con ruedas, ya que permiten realizar tareas de forma simple en entornos más acotados y también facilitan la investigación en sistemas multi-robot. El mejor ejemplo de robot de este tipo es el e-puck⁴ diseñado por la Escuela Politécnica Federal de Lausanne. Está denominado por ellos mismos como un robot diseñado para la

educación en ingeniería, y fue presentado en la novena conferencia de sistemas robóticos autónomos (figura 4). El e-puck posee los siguientes sensores:

- Sensores infrarrojos.
- Cámara con resolución 640x480.
- Anillo de LEDs.
- Acelerómetro.
- Sensores de sonido.



Figura 5. Robot *Thymio*.

Otro robot de características similares es thymio⁵ (figura 5), desarrollado como producto de la colaboración entre la Ecole Polytechnique Fédérale de Lausanne y la Ecole Cantonale d'Art de Lausanne. El objetivo de éste robot es ofrecer un robot a un amplio público, para que muchas personas puedan explorar el mundo relacionado con la robótica y para que pueda ser integrado como herramienta pedagógica en los procesos educativos. Este robot es algo diferente al e-puck y presenta los siguientes sensores y actuadores:

- Micrófono.
- 5 sensores de proximidad frontales y 2 traseros.
- Sensores de temperatura.
- 39 LEDs para indicar el estado de los sensores.
- Altavoz.
- 2 sensores de seguimiento de líneas.

Como hemos dicho más arriba, y como puede verse en la página web del robot, está destinado principalmente a un uso didáctico, con múltiples ejemplos, entre los que

podemos destacar que el color que presentan los leds de la carcasa del robot están en función de su “estado de ánimo” (por ejemplo, rojo representa que thymio tiene miedo o verde quiere decir que está atento).

Una característica especial de éste robot es que está diseñado para que se le puedan acoplar piezas de LEGO⁶ y formar así múltiples construcciones, como puede verse en la figura 6.

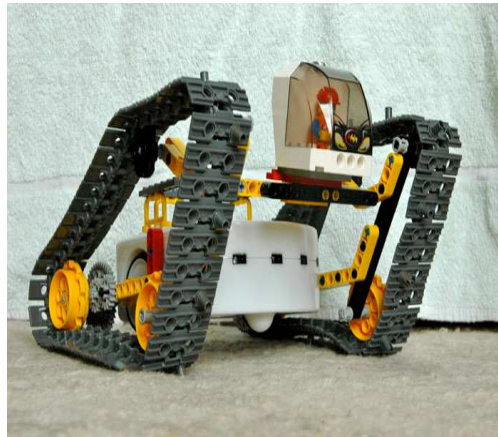


Figura 6 . *Ejemplo de construcción realizada con thymio y piezas de LEGO.*

El problema de todos estos tipos de plataformas utilizadas en investigación es que necesitan una inversión considerable para su adquisición, y posteriormente para su actualización debido a que con el tiempo, el avance y las mejoras que se producen en los campos de la robótica incrementan los requisitos técnicos de estas plataformas. Por tanto, demandan con rapidez una mayor capacidad de cómputo para ofrecer así una mejor respuesta ante estímulos, una mayor definición de datos obtenidos por los sensores, ofreciendo una mejor calidad y un mayor rango de datos, la inclusión de nuevos componentes, como pueden ser cámaras de vídeo, antenas para comunicación wifi, etc.

Como una buena solución a este problema, y dado el rápido desarrollo de la tecnología en los teléfonos inteligentes (Smartphone), hace unos años surgió la idea de utilizar una plataforma robótica móvil y acoplarle un Smartphone para dotarla de capacidad de cómputo, sensores y gran conectividad inalámbrica. La ventaja de esta solución es el constante avance de los Smartphone y la robótica, que permite ahorrarse mucho dinero en las actualizaciones de nuestro robot, ya que lo que se renovarían sería nuestro teléfono móvil. En esta línea de desarrollo está centrado el presente Trabajo Fin

de Grado, por lo que a continuación mostraremos los ejemplos más representativos de estos robots, donde el teléfono es el protagonista.



Figura 7. *Robot Romo.*

En 2012 Romotive lanzó en Kickstarer⁷ una campaña para poner en marcha su proyecto de robótica orientada a la educación infantil. El resultado es Romo⁸ (figura 7), un robot personal para niños que hace uso del iPhone como “cerebro” de esta solución. Romo es, en esencia, un juguete recomendado para niños a partir de 8 años. Las aplicaciones para iOS⁹ permiten ir aprendiendo el funcionamiento de este robot y también sirven para establecer las bases de una programación de tareas básica pero divertida. Sin embargo, no es un robot con movimiento autónomo ni está pensado para investigación, ya que carece de un software de programación de bajo nivel y está pensado para ser dirigido por el usuario. Una de las grandes desventajas del Romo son la incompatibilidad con Android¹⁰, ya que sólo es válido para iPhone y su movimiento controlado, ya que actualmente existen robots de su misma categoría que pueden ser programados.

Resolviendo el problema de ser compatible únicamente con uno de los sistemas operativos para móviles, existen tres alternativas que permiten usar iOS y Android.

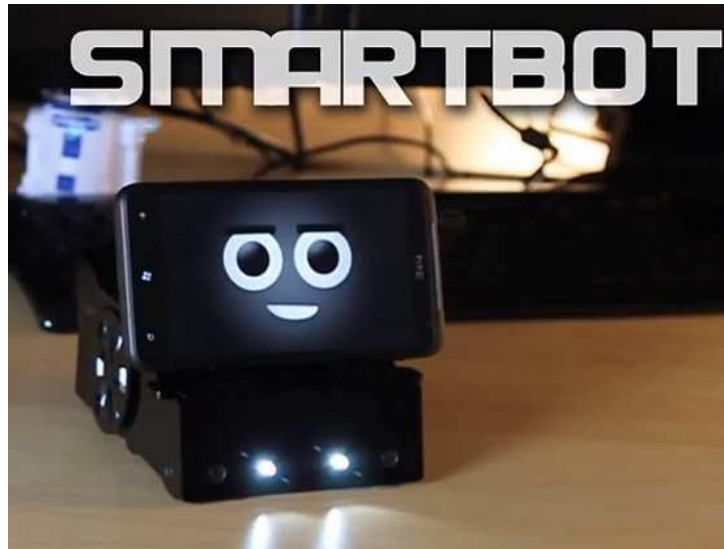


Figura 8. Robot Smartbot.

La primera, Smartbot¹¹ (figura 8), puede ser también controlada por cualquier Smartphone o incluso una placa de Arduino¹² pero no soporta el protocolo de red que utiliza ROS¹³ ni tiene ningún sensor.

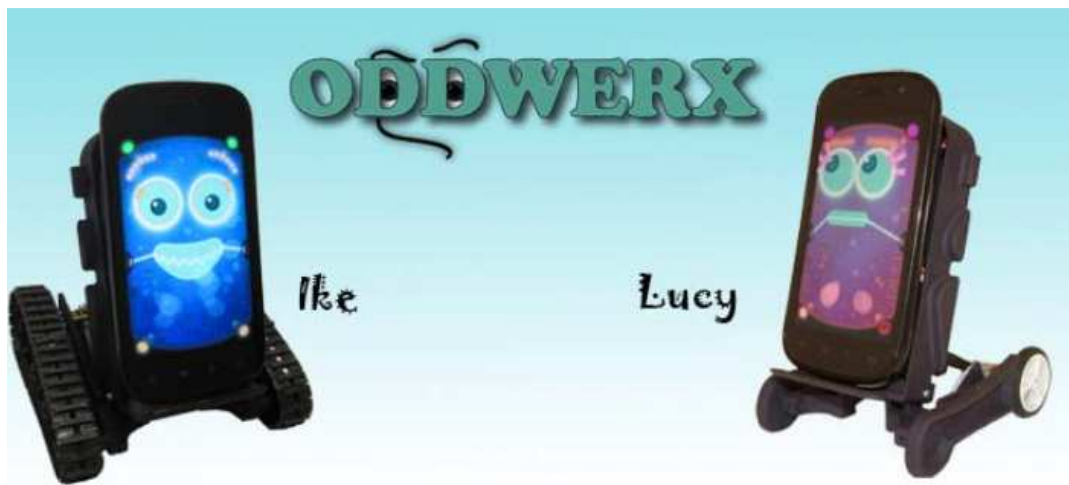


Figura 9. Robot Oddwerx

La segunda posibilidad es Oddwerx¹⁴ (figura 9), Robot que tiene reconocimiento de voz, de objetos, mapeo del entorno y lo más importante que soporta el protocolo de red que utiliza ROS, ya que esto lo hace más versátil y mayor facilidad de computo. Es utilizado como juguete, como una mascota virtual, para aplicaciones educativas y para aplicaciones de interacción con el entorno. Oddwerx no dispone de sensores en la base que le den autonomía en el movimiento.



Figura 10. *Robot Wheelphone.*

La tercera y última alternativa sería Wheelphone¹⁵ (figura 10), que fue desarrollado por GCtronic¹⁶ y es compatible con iOS y Android, posee el protocolo ROS, sensores de proximidad y un sistema de base de auto-carga, por lo que, se convierte en el sistema de robots autónomo más completo de todos los anteriores mencionados. No dispone de leds que hagan una interacción con el usuario, como por ejemplo baja batería, errores, tocar la pantalla, etc... ni tampoco dispone de una rotación independiente del móvil con respecto al resto de la plataforma. Otra desventaja que presenta esta plataforma es la comunicación entre la plataforma y el Smartphone, que es por medio de un cable, por lo que es diferente si el dispositivo opera con un iPhone o con un dispositivo Android.

Desde el año 2013, el Grupo Integrado de Ingeniería de la UDC ha venido desarrollando una plataforma robótica para Smartphone, que cubriese las carencias detectadas en las existentes. Este desarrollo se ha denominado ROBOBO, y posee, como principales características:

- Una base móvil sensorizada basada en hardware abierto y un entorno de programación integrado en el sistema operativo ROS, con la posibilidad de ser programado utilizando todo el conjunto de librerías disponibles en ROS.

- Se puede fabricar de manera totalmente artesanal (la carcasa y sus componentes no electrónicos se pueden hacer con la impresora 3D, algo totalmente revolucionario).
- Gran número de sensores, alta capacidad de cómputo y comunicación (debido al Smartphone).

El presente Trabajo Fin de Grado se centra en el desarrollo del sistema básico de sensorización de la nueva versión de la plataforma robótica ROBOBO (versión 2.0). Esta nueva versión parte del robot ROBOBO 1.0 (figura 11), pero implica un rediseño completo de la mayoría de sus elementos. El ROBOBO 1.0 constituyó una versión conceptual que permitió comprobar que el sistema plataforma + Smartphone era factible, pero durante su desarrollo se detectaron numerosas carencias y mejoras posibles, que serán el objetivo de la versión 2.0 en la que se centra el presente T.F.G.

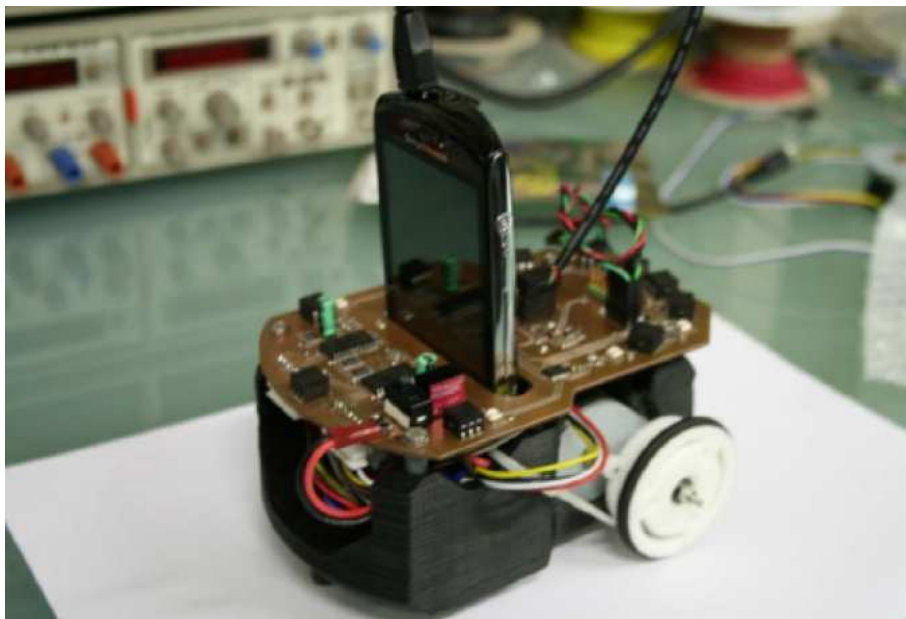


Figura 11. Robot ROBOBO

El ROBOBO 1.0 original consistía en una plataforma móvil autónoma con las siguientes características:

- Interactuación con el entorno por medio de 9 Leds RGB, que cambiaban de color con la proximidad de los objetos, hasta ponerse en rojo que se detenía e intentaba esquivar el objeto.

- Movimiento autónomo por medio de 9 Sensores IR de proximidad: 2 en la parte frontal, 2 a los laterales formando 45°, dos en laterales centrados a la parte delantera y los 3 restantes en la parte posterior. Con esta distribución se tenía una visión general del entorno.
- Movimiento por medio de 2 Motores paso a paso con un paso de 1/8 paso de rueda, teniendo una gran precisión en el giro.
- Compatible con Android, algo imprescindible.
- Capacidad de interactuar con otros dispositivos ROS
- Comunicación USB.

En esta nueva versión del robot se realizarán numerosos cambios, algunos de los cuales se aprecian en la figura 12, donde se puede apreciar que su apariencia ha variado bastante:

- El número de leds RGB será de 21, donde 9 de ellos formarán una matriz de comunicación. Estos sensores nos darán información del estado de ROBOBO por medio de colores o de formas en la matriz.
- Tipo de motores: se ha pasado dos motores paso a paso a dos motores en continua, que ocupan mucho menos espacio y consumen menos batería.
- La forma de la carcasa se ha cambiado radicalmente, para un mejor efecto visual de cara a su comercialización.
- Se ha modificado el tipo de comunicación con el Smartphone, ya que no todos los Smartphone tienen la conexión USB en el mismo lugar. Por ello se ha optado por conexión bluetooth, que tiene como desventaja el aumento de consumo de batería en el Smartphone.
- Adaptación de cualquier tipo de tamaño de móvil: el Smartphone se ha posicionado en la parte superior de la plataforma, y se le ha colocado un adaptador de sujeción de tamaño variable, ya que hay gran diversidad de dimensiones en el mercado de los teléfonos de este tipo y la tendencia es a que sean cada vez más grandes.
- Movimiento del Smartphone en la plataforma: permite al teléfono rotar y ladearse, para que la interacción con el usuario sea más realista. Este movimiento se hace por medio de otros dos motores de continua como los de las ruedas.

- La agregación de un multiplexor para la unión de los sensores al microcontrolador, que antes se hacía mediante resistencias pull-up.
- Los sensores se han modificado tanto en número, como el tipo sensor en sí mismo, su ubicación en la plataforma y su comunicación con el microcontrolador. Estos cambios constituyen el tema central de este trabajo fin de grado y se detallarán a lo largo de los apartados siguientes.

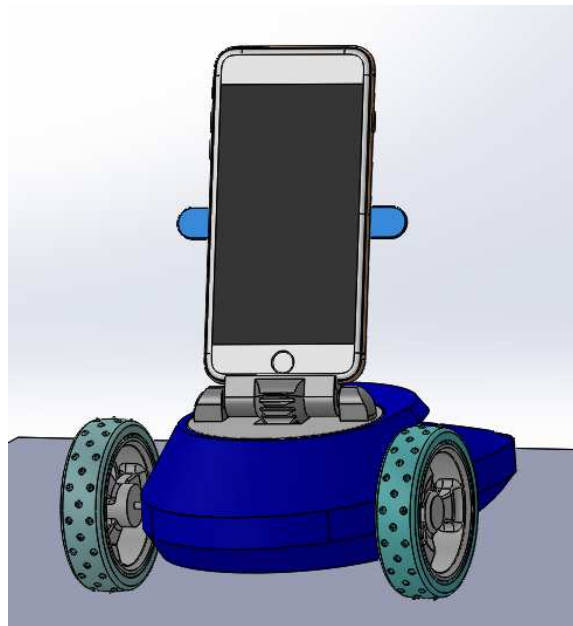


Figura 12. Robot ROBOBO 2.0.

Además de constituir una nueva opción como robot móvil autónomo para investigación, el ROBOBO 2.0 será utilizado dentro del proyecto denominado DREAM¹⁸, financiado por la Unión Europea y que comenzó el 1/1/2015 y terminará el 31/12/2018. Algunas de las entidades colaboradoras son: CNSRS¹⁹, ENSTA²⁰, UPMC²¹ y UDC. El principal uso del ROBOBO dentro del proyecto DREAM es como sistema educativo, donde los jóvenes aprenderán a programar el robot mediante un lenguaje simple (basado en bloques), y lo que es más importante, utilizando su propio teléfono móvil. El sistema ROBOBO que se pretende desarrollar tiene como principal elemento un sistema de enseñanza totalmente interactivo donde el mismo Robobo les irá indicando a los alumnos lo que están haciendo bien y mal, de modo que el aprendizaje del robot y del alumno sea concurrente.

Tras haber enmarcado el presente trabajo fin de grado, pasamos a establecer los objetivos del mismo.

2. Objetivos.

El principal objetivo de este Trabajo Fin de Grado es el siguiente:

Diseño de la circuitería electrónica y de la programación para el funcionamiento de sensores infrarrojos en la plataforma robótica ROBOBO 2.0

Para lograrlo, se establecen los siguientes sub-objetivos:

- Revisión y análisis del estado del arte del campo de los sensores infrarrojos para poder seleccionar los componentes más adecuados.
- Selección de los sensores en función de los requisitos impuestos por los investigadores del Grupo Integrado de Ingeniería.
- Análisis de posibilidades de ubicación de los sensores en la carcasa, así como definición del número de sensores que es más adecuado con un máximo control del movimiento de la plataforma.
- Diseño de la circuitería electrónica y programación de los sensores.
- Programación del circuito integrado que controla en funcionamiento de los sensores.
- Selección y programación del multiplexor que secuencia la señal de los sensores al microcontrolador de los sensores.
- Pruebas de funcionamiento de los sensores montados en la carcasa.

3. Abreviaturas y definiciones.

ADC: Analog to Digital Converter. Conversión analógica digital.

ASL: Ambient sensor light.

BLUETOOTH: es una especificación industrial para Redes Inalámbricas de Área Personal (WPAN) que posibilita la transmisión de voz y datos

CAN: Controller Area Network.

DATASHEET: como su propio nombre indica en inglés, hoja de datos. Es un documento que resume el funcionamiento y las características de un componente.

DMA: Direct Memory Access. Acceso Directo a Memoria.

DRC: Design Rule Check. Comprobacion Reglas de Diseño.

I2C: Inter-Integrated Circuit. Circuitos Inter-Integrados.

IC: Integrated Circuit. Circuito integrado.

IRLED: Infrared Light Emitting Diode.

LED: Light Emitting Diode.

PS: Proximity sensor.

PCB: Printed Circuit Board. Placa de circuito impreso.

PIC: Microcontrolador de la marca Microchip.

PWM: Pulse Width Modulation. Modulación por ancho de pulso.

RAM: Random Access Memory.

ROM: Read Only Memory. Es un medio de almacenamiento utilizado en ordenadores y dispositivos electrónicos, que permite sólo la lectura de la información y no su escritura, independientemente de la presencia o no de una fuente de energía.

RGB: Red Green Blue, esto quiere decir que pueden dar cualquier color.

SPI: Serial Peripheral Interface. Bus estándar de comunicaciones.

SRAM: Static Random Access Memory.

TILT: Inclinación, en nuestro caso inclinación del Smartphone.

UART: Universal Asynchronous Receiver/Transmitter.

USB: Universal serial bus.

USB OTG: USB On-The-Go es una extensión de la norma USB 2.0 que permite a los dispositivos USB tener mayor flexibilidad en la gestión de la interconexión. Permite que ciertos dispositivos, por ejemplo, un reproductor de audio digital o un teléfono móvil, actúen como host, por lo que se les puede conectar una memoria USB, un ratón, un teclado, un disco duro, un módem, etc.

4. Programas empleados.

Para la realización de este trabajo fin de grado se han utilizado los siguientes programas informáticos:

- **KiCad:** es un programa que se emplea para el diseño y realización de circuitos impresos.
- **MPLAB X IDE:** es el software desarrollado por Microchip para desarrollar aplicaciones para sus microcontroladores.
- **MPLAB XC32 Compiler:** es el compilador de microchip para los PIC32.
- **Circuit Cam:** programa que importa los ficheros en formato Gerber del KiCad, y los convierte en archivos legibles para el Board Master.
- **Board Master:** programa que envía los datos a la máquina LPKF ProtoMat C30s.
- **SolidWorks:** programa de diseño utilizado para la colocación de los sensores en la carcasa y la visualización de su alcance.

5. Requisitos previos.

Los requisitos impuestos por los investigadores del Grupo Integrado de Ingeniería para la realización de este trabajo fin de grado fueron los siguientes:

- Los sensores no pueden tener un coste muy elevado (máximo 4-5 euros por unidad) y tienen que tener una franja de percepción de un objeto en un rango de 10cm, que sería el valor que tomaríamos para que el robot pueda reaccionar desde que intercepta un objeto.

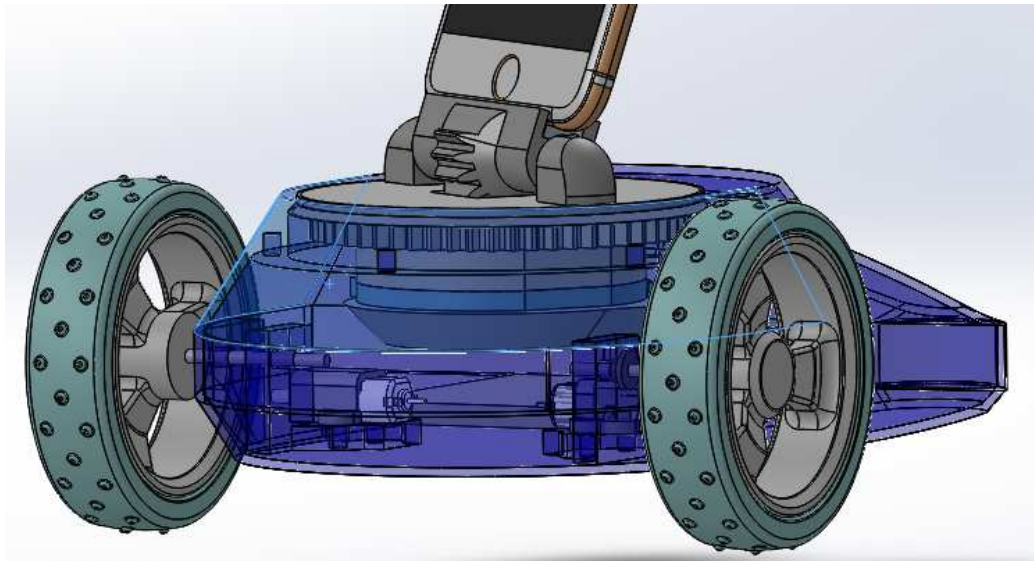


Figura 13. Interior de la carcasa del Robobo 2.0.

- Los sensores deben ir alojados en el interior de la carcasa del ROBOBO 2.0 que se muestra en la figura 13 y en la placa electrónica, con una colocación de los sensores en la carcasa de modo que sea posible sacar el led infrarrojo al exterior para su funcionamiento, nunca delante de ningún cuerpo del Robobo. Para el funcionamiento de los infrarrojos es necesario que el led del infrarrojo pueda emitir y recibir las ondas que ha emitido, por lo que sin un hueco en la carcasa no sería capaz de detectar objetos.
- El circuito impreso donde se alojan los sensores debe tener unas dimensiones nunca superiores al espacio que tendría para alojarse en el interior de la carcasa, de lo contrario no podría ser colocado dentro de él. Otro aspecto impuesto es hacerle los huecos pertinentes a la carcasa para la salida perfecta del led infrarrojo por donde se tendrá una comunicación directa con el entorno. Para ello se tiene que tener en cuenta la posición del led infrarrojo separado y en

conjunto con los demás componentes. Dentro de la placa electrónica tiene que tener una posición tal que cuando se coloque dentro de la carcasa, se sitúe en los huecos hechos para los infrarrojos.

- El microcontrolador viene escogido previamente, por lo que tiene que ser el microcontrolador PIC32MX534F064H, que tiene un protocolo de comunicación I2C que hay que programar.

En los siguientes apartados se describirá el desarrollo realizado para cumplir los objetivos del apartado 2 teniendo en cuenta estos 4 requisitos.

6. Elección de los sensores.

Para la elección de los sensores se deben tener en cuenta los requisitos establecidos con anterioridad. Por tanto, el primer paso debe ser entender con claridad el funcionamiento de un sensor infrarrojo y qué modelos existen en el mercado actual. Por este motivo, a continuación se explicará el funcionamiento básico de los sensores infrarrojos, se presentarán varios tipos de sensores de los cuales escogeremos el que más se adapta a nuestras condiciones, y finalmente pasaremos a detallar las características del sensor escogido.

6.1. Fundamentos teóricos. Sensor infrarrojo.

Un sensor es un dispositivo capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación, y transformarlas en variables eléctricas. Un sensor de proximidad infrarrojo (comúnmente llamado sensor infrarrojo) es un dispositivo que es capaz de detectar objetos que tiene en su entorno de visión, transformar la distancia a la que se encuentra el objeto en variables eléctricas y enviarla a otros dispositivos.

El sensor de infrarrojos consta, básicamente, de un diodo LED emisor, y de un diodo LED receptor: El diodo emisor, IRLED (figura 14), es un emisor de rayos infrarrojos, radiación electromagnética situada en el espectro electromagnético, en el intervalo que va desde la luz visible a las microondas. Los rayos infrarrojos se caracterizan por ser portadores de calor radiante y son producidos en mayor o menor intensidad por cualquier objeto a temperatura superior al cero absoluto. El diodo receptor es un fotodetector que trabaja como un transistor clásico, pero que no tiene conexión base (figura 15).

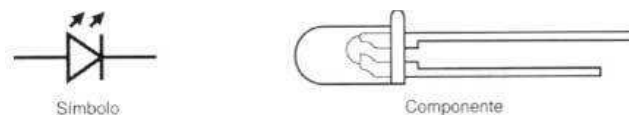


Figura 14. Diodo LED emisor.



Figura 15. *Diodo receptor.*

En estos transistores, la base está reemplazada por un cristal fotosensible que cuando recibe luz, produce una corriente y desbloquea el transistor. En el fototransistor la corriente circula sólo en un sentido y el bloqueo del transistor depende de la luz: cuanta más luz hay más conduce. El fototransistor reacciona con la luz visible y también con los rayos infrarrojos que son invisibles. Para distinguirlo del LED su cápsula es transparente.

6.2. Selección de sensores a emplear.

Tras haber realizado una búsqueda entre las distintas opciones de sensores infrarrojos presentes en el mercado, podemos destacar los siguientes modelos:

- **GP2D150AJ00F:** Sensor infrarrojo de la marca SHARP²², que presenta un rango de detección desde los 4 cm hasta los 30 cm, con un consumo medio de corriente de 33-50 mA, y siendo el de mayor tamaño y precio (el precio consultado fue de 11.99€/u por lotes de 20).
- **SI1141-A11-GMR:** Sensor de proximidad infrarrojo de la marca Silicon Labs²³. Tiene comunicación I2C, una franja de voltaje de 1.7-3.6V y un precio de 2 euros por un lote de más de 10 unidades.
- De la marca Vishay²⁴ existe la familia de sensores **VCNL**. Toda la familia tiene un rango de temperaturas igual, de -25° a +85°, una resolución de proximidad de 16 bits y un rango de distancias de 20 cm. Una de las características más importantes por las que se ha seleccionado esta familia de sensores, es que el anterior sensor de ROBOBO 1.0 era de la misma marca y con la misma comunicación i2C, lo que nos permite que su programación solo tenga que ser modificada. Los sensores tienen unas características similares que exponemos a continuación:

- **VCNL4010:** Sus dimensiones son cuadradas, 3.95mmx3.95mmx0.75mm, con un precio bastante alto de 3.62 €. Teniendo en cuenta que el Robobo anterior constaba de ocho sensores, nos pondríamos en un precio de 28.96 €, por lo que se excede el coste máximo.
- **VCNL4020 y el sensor VCNL4020X01:** tiene unas dimensiones de 4.9mmx2.4mmx0.85mm, por lo que tiene una forma rectangular pero más profunda que el anterior. Su precio es de 2.32€ y 3.386€, algo inferior al anterior mencionado, pero aun así se sale del máximo permitido.
- **VCNL4040:** es el sensor con menor superficie, con unas dimensiones de 4mmx2mmx1.1mm y con un precio de 1.71€, por lo que la gama es el

Nombre	Dimensiones (LxWxH)(mm)	Distancia (mm)	Resolución de proximidad	Temperatura (°C)	Precio (€)
VCNL4010	3.95x3.95x0.75	200	16 bit	-25 a +85	3.62
VCNL4020	4.90x2.40x0.83	200	16 bit	-25 a +85	2.32
VCNL4020X01	4.90x2.40x0.83	200	16 bit	-40 a +105	2.386
VCNL4040	4.0x2.0x1.1	200	16 bit	-40 a +85	1.71

que tiene menor precio y superficie.

Tabla 1. Resumen de características de la familia VCNL40XX.

Se analizaron más modelos de sensor pero fueron desechados ya de inicio por no cumplir con los rangos mínimos de distancia y coste requeridos. Finalmente, de entre todos los posibles sensores, se escogió el sensor óptico de Vishay VCNL4040, por ser el sensor que más se ajustaba a nuestros requerimientos en cuanto a distancia de detección, unos 20 cm, y coste no superior a los 4 euros.

6.3. Sensor infrarrojo VCNL4040

Como hemos dicho, el sensor que finalmente hemos escogido ha sido el sensor de infrarrojos VCNL4040 de VISHAY (figura 16).



Figura 16. *Sensor VCNL4040 de Vishay.*

El VCNL4040 es un sensor de proximidad (PS) con un sensor de luz ambiental integrado (ASL). Es un sensor óptico que combina un emisor de infrarrojos, fotodiodo PIN, sensor de luz ambiental y procesamiento de señales en un solo paquete con un ADC de 16 bits a un precio reducido, y se utiliza tanto para la medición de proximidad como para la medición de luz ambiental. El dispositivo dispone de un sensor de luz ambiental para apoyar la retroiluminación convencional, mostrar el ajuste del brillo y los sensores de proximidad para la detección de movimiento y del objeto. Con un alcance de hasta 20 cm, al ser un único componente simplifica en gran medida el uso y el diseño de un sensor de proximidad en las aplicaciones industriales y de consumo. El VCNL4040 cuenta con una superficie de montaje de 4,0mm x 2,0mm y una altura de 1.1mm, diseñadas específicamente para los requisitos de pequeño espesor de teléfonos móviles, cámaras digitales, y aplicaciones de tablet PC. A través de un bus I2C estándar de serie de interfaz digital, permite el acceso a una "señal de Proximidad" y a una señal que nos indica la "Intensidad luminosa", medidas sin necesidad de cálculos complejos o de programación.

El esquema eléctrico de este sensor se muestra en la figura 17.

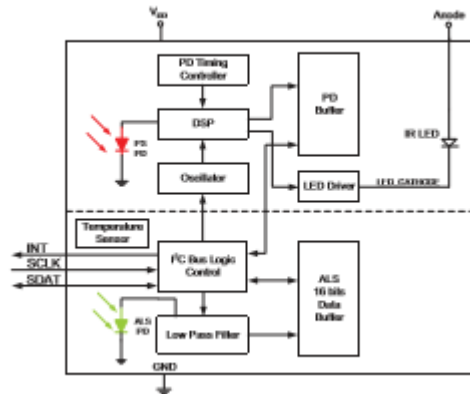


Figura 17. Esquema electrónico del sensor VCNL4040.

El emisor de infrarrojos integrado tiene una longitud de onda máxima de 940 nm. Emite una luz que se refleja en un objeto a 20 cm del sensor. El espectro emisor de infrarrojos se muestra como una línea continua. El emisor de infrarrojos tiene una unidad programable de corriente máxima a 200 mA. La luz infrarroja emitida es modulada en una de las cuatro frecuencias portadoras definidas por el usuario: 390.625 kHz, 781.25 MHz, 1.5625 MHz, 3.125 MHz.

El PIN del fotodiodo recibe la luz que se refleja en el objeto y la convierte en corriente. Tiene una sensibilidad máxima de 940 nm, igualando la longitud de onda de pico del emisor. Es insensible a la luz ambiente. El utilizar una señal modulada por la proximidad proporciona una ventaja sobre otros sensores en el mercado.

El sensor de luz ambiental recibe la luz visible y la convierte en corriente. El ojo humano puede ver la luz de longitudes de onda de 400nm a 800nm con un pico de 560nm. El sensor de luz ambiental de Vishay disminuye a este rango de sensibilidad, tiene sensibilidad máxima a 550nm.

La conexión mínima requerida para este dispositivo viene proporcionada por el fabricante en la hoja de especificaciones, la cual se muestra en la figura 18:

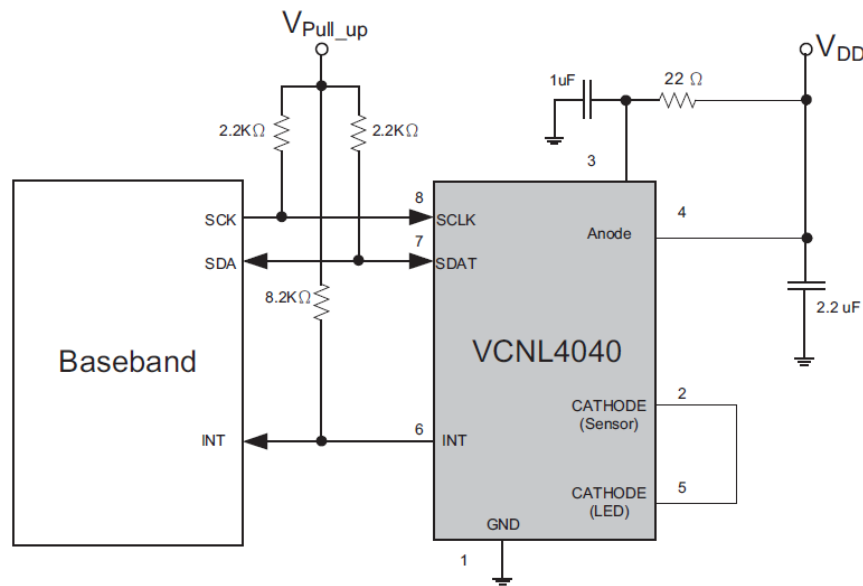


Figura 18. Conexiones requeridas para el sensor VCNL4040.

Como se puede observar, necesitamos la colocación de dos condensadores de desacoplo:

- Un condensador para la alimentación del cátodo del sensor infrarrojo de $2.2\mu F$.
- Otro condensador de $1\mu F$ para la alimentación del VCNL.

También es necesaria la colocación de cuatro resistencias de tipo pull-up (esto quiere decir que cuando no pasa corriente por ellas, su estado de salida es alto), para el correcto funcionamiento del sensor.

A continuación mostraremos las características que nos van a ser más útiles para una correcta utilización y soldadura del sensor:

a) Pico de distancia y ángulo de operación por intensidad de onda.

Es muy interesante, para favorecer la visión delante de las ruedas del Robobo, tener un sensor infrarrojo que no abra mucho su ángulo de cobertura. Lo que se pretende es que el área de detección del sensor se focalice en la parte delantera de la rueda. Se desea que así sea para que no se produzca la detección de objetos que se encuentran en una posición diferente a la orientación del sensor infrarrojo. Por ello, este sensor es muy interesante, ya que tiene un ángulo muy pequeño como se muestra en la figura 19:

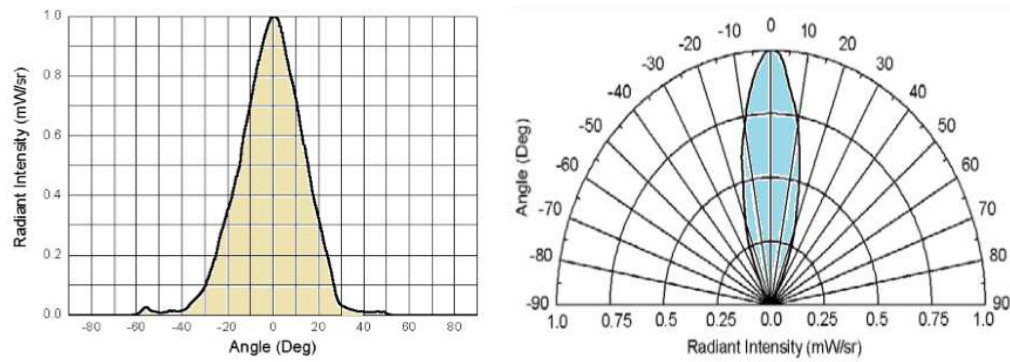


Figura 19. *Perfiles IRED.*

b) Temperatura

Como podemos ver en la figura 20, la temperatura del sensor es proporcional a la intensidad de corriente, lo que sucede en una amplia gama de productos electrónicos que sufren calentamiento con el aumento de intensidad de corriente. Como nos indica en la hoja de datos, el rango para la temperatura de funcionamiento es de -40°C a $+85^{\circ}\text{C}$, mientras que el rango para la temperatura de almacenamiento es la que se muestra en la figura 20, va desde -40°C a $+100^{\circ}\text{C}$. Además, la hoja de datos también nos indica que la máxima intensidad de corriente para su uso es de 200 mA, inferior a la que se nos muestra en la figura 20. Con todos estos datos podemos concluir que el sensor no es un buen disipador de calor, como se puede observar en esta figura, porque su temperatura aumenta en 140°C con un aumento de 15mA. Esto es debido a que la corriente en la entrada de emisiones del infrarrojo es superior que en la salida provocando un almacenamiento de temperatura.

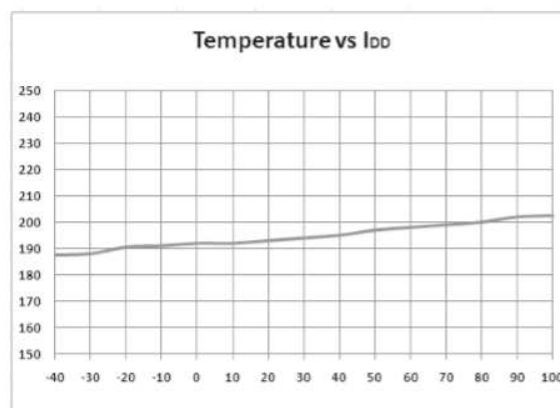


Figura 20. *Temperatura frente a corriente utilizada.*

6.4. Caracterización del sensor infrarrojo

Una vez adquirido el sensor seleccionado, se realizó una comprobación práctica de su correcto funcionamiento, con el objeto de contrastar las medidas con las especificaciones proporcionadas por el fabricante en el datasheet.

Se realizaron varias medidas con un sensor a diferentes distancias y utilizando diferentes colores y materiales. Primero nos centraremos en el análisis del sensor a diferentes distancias para comprobar cuáles son las franjas de valores en los que nos moveremos. No nos interesarán tanto los valores concretos, ya que como veremos más adelante, cambian de forma significativa con cada material. A continuación analizaremos el comportamiento del sensor con diferentes materiales, seleccionando el mejor y el peor material. Por último hablaremos del comportamiento del sensor con los diferentes colores. Después de estos apartados haremos nuestras propias conclusiones en base a los datos obtenidos.

a) Comportamiento del sensor a diferentes distancias.

Para la realización de la pruebas del funcionamiento del sensor infrarrojo, lo primero que se ha tenido que realizar, es una correcta colocación del sensor y la superficie, para realizarlo en las condiciones óptimas. Sólo se ha utilizado un sensor, el delantero, como se muestra en la figura 21, esto es así porque no estamos analizando su comportamiento en el robot, sino su comportamiento frente al ambiente, por lo que no nos interesa, la interacción que hay entre los sensores en la plataforma, aunque será un punto a analizar más adelante.

El sensor se coloca en el lugar delantero de la plataforma, en las condiciones más óptimas para el sensor, que son: una superficie de color blanca (color de mayor reflexión), perpendicular a la superficie donde se apoya y paralelo a la superficie que va a detectar. Las medidas son tomadas con el calibre desde la superficie del sensor, hasta la superficie blanca que se muestra en la figura y para cada medición recolocamos manualmente la superficie blanca y volvemos a hacer las mediciones.

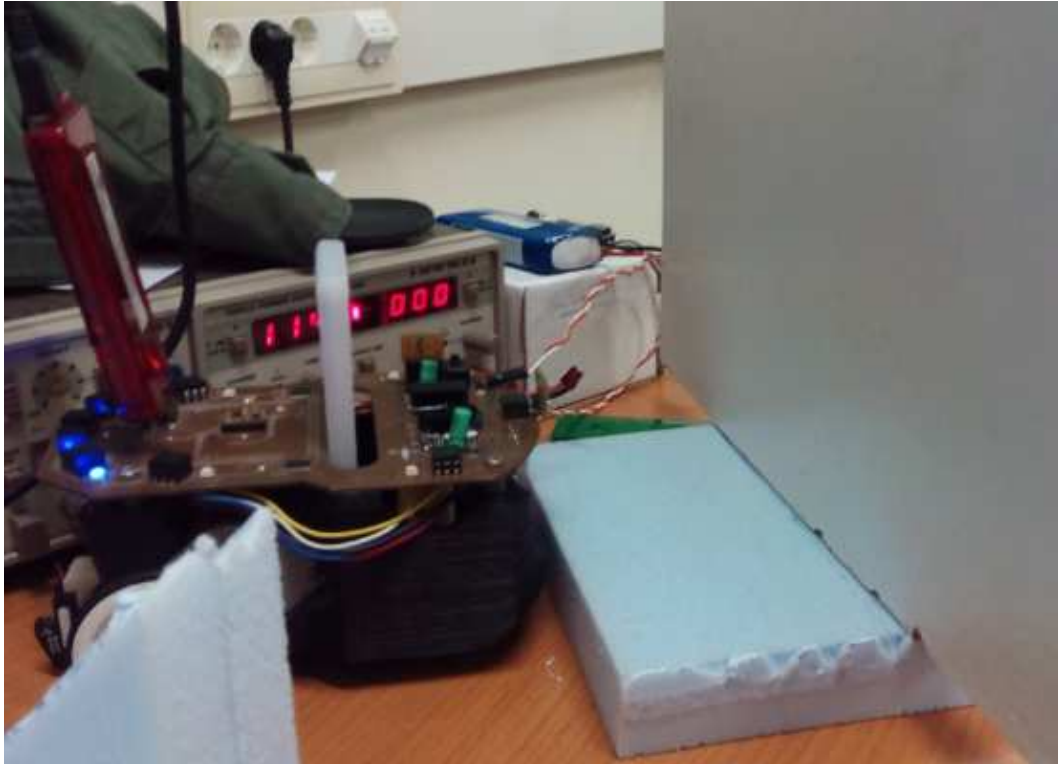


Figura 21. *Posicionamiento de la superficie a medir y el sensor VCNL4040.*

En la tabla 2 se muestran los resultados de las medidas realizadas. El sensor no devuelve una magnitud física, sino una conversión entre una señal analógica (intensidad de las ondas recibidas) a una señal digital (conversión de la señal analógica que procesa a un valor digital). Por ello es necesario realizar estas medidas, y saber así que número se corresponden con qué distancia. Se comenzó con una distancia que daba los mismos valores que sin ningún objeto delante, resultando 15 cm, y no 20 cm como pone en la hoja de datos. El valor de intensidad medido en este caso fue de 2423 como se ve en la tabla 2. Como podemos observar, las medidas a largas distancias no tienen mucha fiabilidad, ya que los valores obtenidos se intercalan en diferentes distancias. A partir de los 9 cm los valores son mucho más lineales y progresivos. Estos valores serán de los tomaremos referencia para nuestro estudio.

CALIBRACIÓN SENSORES INFRARROJOS					
DIRECCIÓN RECTA		MEDICIONES DECIMAL			
distancia (cm)	INTERVALOS	1°	2°	3°	4°
nada	2423	2455	2431	2427	2426
10	2435-2440	2442	2440	2437	2435
9	2430-2440	2435	2434	2438	2439
8	2440-2450	2447	2446	2436	2437
7	2450-2460	2450	2455	2464	2453
6	2465-2470	2467	2470	2465	2450
5	2490-2500	2489	2501	2498	2496
4	2510-2520	2518	2517	2514	2514
3	2590-2600	2599	2595	2593	2597
2	2740-2750	2741	2745	2742	2750
1	3180-3190	3187	3182	3187	3186
0,5	4015-4025	4018	4015	4013	4026
0,25	9120-9150	9102	9126	9138	9146

Tabla 2. Comprobación de la sensibilidad de los sensores.

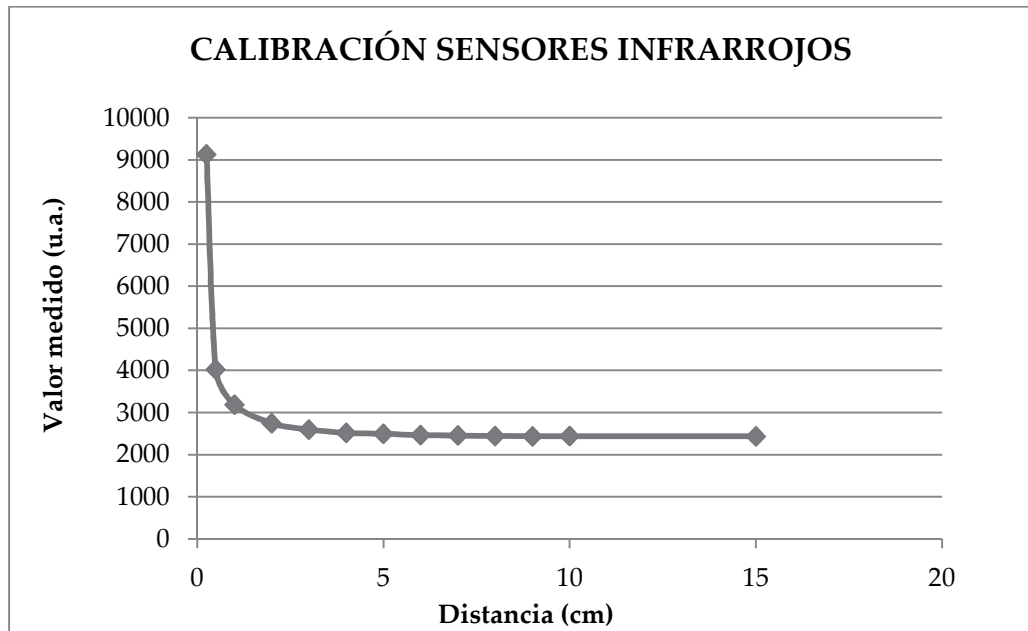


Figura 22. Señal del sensor frente a distancias a las que está el objeto.

b) Comportamiento del sensor con diferentes materiales.

A continuación se analizó la respuesta de los sensores a diferentes materiales, con una colocación como se muestra en la figura 23. Este análisis es muy interesante para este TFG, ya que el Robobo 2.0 es un dispositivo robótico polivalente que se puede usar tanto en investigación como en un entorno doméstico. El entorno el que se mueve es muy heterogéneo, por lo tanto, se tiene la necesidad de ver cómo responden los sensores ante diferentes materiales.

En este punto hay que hacer un inciso del lugar que van a ocupar los sensores en la carcasa del robot, aunque se verá con más detalle en el apartado siguiente. Así, estos deben tener dos tipos de direcciones: unos dirigidos hacia su entorno perpendicular (para no chocar) y otros dirigidos directamente al suelo (para no caerse). Los sensores de suelo tiene que estar preparados para varios tipos de suelos (parquet, plaqueta, terrazo, cristal, etc), y los sensores que van direccionados de frente tiene que estar preparados para todo tipo de paredes, rodapiés y todo tipo de patas tanto de muebles como de mesas o sillas. Por tanto, hicimos las medidas a una distancia de 8 cm donde colocamos los diferentes materiales que se muestran en la tabla 3. En este caso no nos interesa el valor de intensidad, sino la diferencia entre los valores máximo y mínimo, para determinar con que material se comporta mejor y con cual peor. En la figura 24 podemos observar este rango de sensibilidad para cada material.

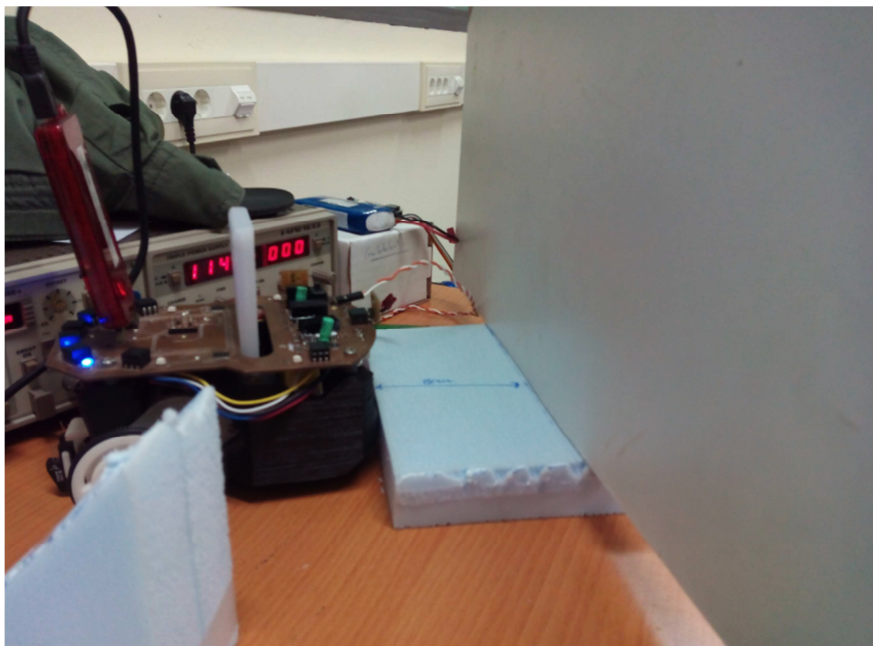


Figura 23. Colocación del material a una distancia de 8cm.

CALIBRACIÓN SEGÚN EL MATERIAL A UNA DISTANCIA DE 8 cm								
MATERIALES	MEDICIONES DECIMAL							
	1°	2°	3°	4°	media	MAX	MIN	Diferencia
MADERA	2482	2464	2467	2459	2468	2482	2459	23
PLÁSTICO	2467	2475	2465	2464	2467,75	2475	2464	11
TELA	2429	2443	2436	2437	2436,25	2443	2429	14
CRISTAL OPACO	2439	2433	2441	2431	2436	2441	2431	10
METAL	2436	2439	2438	2439	2438	2439	2436	3
METAL LACADO	2428	2434	2433	2431	2431,50	2434	2428	6
VIDRIO TRASLICUDO	2422	2418	2419	2416	2418,75	2422	2416	6
PLAQUETA CLARA	2419	2414	2412	2413	2414,50	2419	2412	7

Tabla 3. Comprobación de la sensibilidad del sensor según el material.

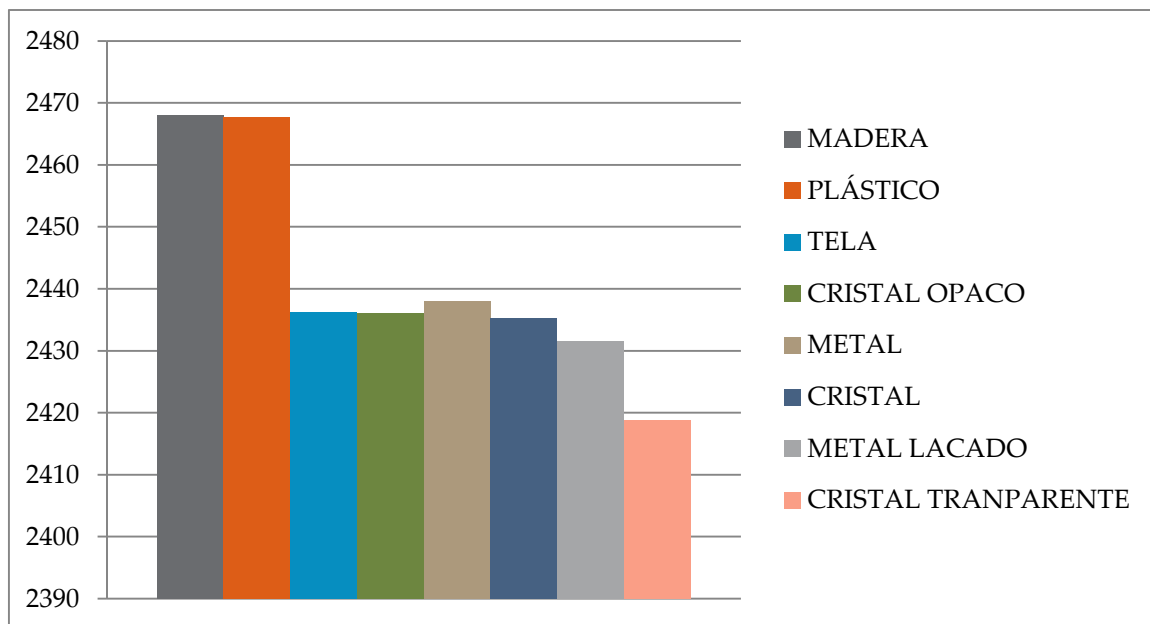


Figura 24. Sensibilidad del sensor en diferentes materiales.

En conclusión, obtenemos que los materiales que más sensibilidad tienen son la madera y el plástico, mientras que el cristal transparente es el que peor detecta el sensor. A continuación haremos un estudio más detallado del mejor material (madera) y del peor material (cristal). Mediciones de madera conglomerada (tabla 4):

distancia (cm)	MEDIDAS EN DECIMAL								MEDIA	MAX	MIN	Difer
	1°	2°	3°	4°	5°	6°	7°	8°				
20	2368	2370	2367	2363	2370	2368	2371	2376	2370,4	2377	2363	14
15	2386	2385	2382	2385	2381	2380	2383	2380	2382,6	2386	2380	6
10	2399	2402	2401	2399	2403	2406	2404	2402	2402,8	2412	2399	13
9	2406	2410	2413	2406	2408	2409	2415	2412	2409,3	2415	2406	9
8	2419	2417	2424	2416	2419	2423	2415	2419	2418,5	2424	2415	9
7	2438	2435	2435	2438	2436	2431	2434	2433	2435,1	2438	2431	7
6	2461	2452	2451	2455	2446	2455	2454	2457	2454,6	2461	2446	15
5	2490	2485	2484	2487	2492	2490	2491	2487	2488	2492	2484	8
4	2563	2561	2559	2556	2559	2557	2564	2558	2559,9	2564	2556	8
3	2696	2697	2694	2695	2692	2694	2692	2697	2694,7	2697	2692	5
2	3090	3082	3085	3080	3082	3080	3077	3083	3082	3090	3077	13
1	5415	5413	5410	5408	5411	5412	5413	5417	5412,8	5417	5408	9
0,5	12803	12793	12804	12809	13022	12810	12810	12815	12826	13022	12793	229

Tabla 4. Mediciones del sensor en la madera conglomerada.

Con estos valores nos damos cuenta de que la media es bastante fiable, ya que la diferencia no pasa nunca de 15 (excepto en 0.5 cm que ya son valores muy elevados). De esta tabla podemos asegurar que el conglomerado es un buen material, y tiene muy buen comportamiento para este sensor.

Mediciones de cristal transparente (tabla 5):

Distancia (cm)	1°	2°	3°	4°	5°	6°	7°	8°	MEDIA	MAX	MIN	Dif
20	2375	2371	2373	2372	2368	2377	2373	2374	2372,9	2377	2368	9
15	2370	2371	2377	2370	2372	2373	2368	2374	2371,5	2377	2367	10
10	2369	2374	2369	2370	2377	2373	2374	2370	2372,7	2377	2369	8
9	2373	2370	2368	2375	2376	2375	2372	2373	2372,3	2376	2368	8
8	2374	2375	2369	2373	2372	2373	2375	2379	2373,4	2379	2369	10
7	2379	2368	2377	2375	2371	2373	2373	2376	2373,2	2379	2368	11
6	2373	2377	2372	2371	2370	2373	2371	2375	2373,2	2377	2370	7
5	2373	2376	2377	2372	2376	2369	2375	2371	2374,6	2382	2369	13
4	2371	2379	2373	2376	2371	2373	2374	2375	2374	2379	2370	9
3	2381	2387	2385	2378	2382	2380	2381	2384	2382,3	2387	2378	9
2	2396	2399	2396	2393	2400	2397	2399	2395	2397,6	2401	2393	8
1	2471	2472	2477	2476	2474	2472	2471	2474	2473	2477	2471	6
0,5	2643	2645	2641	2647	2646	2645	2644	2645	2644,7	2647	2641	6

Tabla 5. Comprobación del funcionamiento del sensor en cristal.

Con estos valores nos damos cuenta que el material no es nada bueno, ya que el mínimo a 20 cm que no detecta nada es igual que a 5 cm y prácticamente igual que a 3 cm. De esta tabla podemos concluir que el vidrio es un material donde no se pueden hacer mediciones, por lo que el Robobo no va a estar preparado para superficies acristaladas. A 3 cm tiene un comportamiento creciente, es decir, detecta objetos, pero 3 cm es un valor demasiado pequeño para poder obtener una respuesta de detección de las ruedas, por lo que descartamos este material.

c) Comportamiento del sensor con diferentes colores

Ahora pasamos a analizar el comportamiento del sensor dependiendo del tipo de color de la superficie. La reflectancia media de una superficie representa la capacidad de esa superficie para reflejar la luz que proviene desde las luces instaladas (bombillas o la luz que entra por las ventanas) y desde las otras superficies del local en otros colores. La iluminación de una superficie será la suma del flujo luminoso que incide directamente sobre la misma (Componente Directa, que en nuestro caso son las ondas que manda el

sensor) y el Flujo Reflejado en las otras superficies del local (Componente Reflejada o Indirecta, que en nuestro caso es la luz del ambiente).

Como orientación, podemos esperar el siguiente comportamiento:

- Colores muy claros se deberá representar con reflectancias elevadas (mayores o iguales al 70%).
- Colores intermedios se representarán con reflectancias medias (entre 40% y 70%).
- Colores oscuros se representarán con reflectancias bajas (entre 0% y 40%).

Por lo que, antes de hacer las pruebas, ya sabemos que el color negro es el color con mayor absorción de todos, es decir, el que menos reflectividad va a tener.

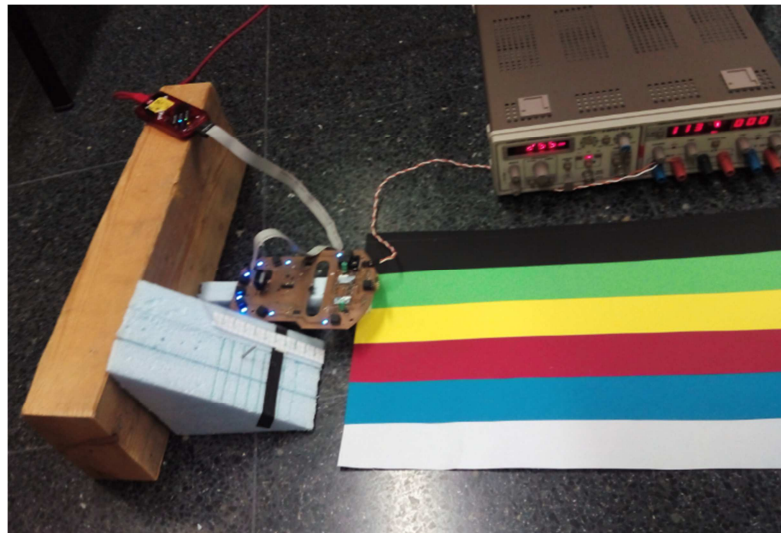


Figura 25. Pruebas con el sensor VCNL4040 para diferentes tonalidades.

Para las pruebas de reflectividad seleccionamos una cartulina con diferentes tipos de colores (figura 25) y medimos. Para hacer las mediciones, nos fijaremos en los leds que lleva la placa donde hay tres colores, azul, verde y rojo, que cada uno significa: azul si tenemos valores inferiores o iguales a 2402, verde si estamos en una franja de [2402, 2816] y rojo si estamos en la franja [216, 10000]. Colocamos los sensores a una distancia de la cartulina que nos den los leds un color azul, a continuación nos vamos aproximando, existe un punto en que es una mezcla de verde y azul igualitaria (anotamos la distancia), seguimos avanzando hasta que cambie a verde

(anotamos este valor), nos seguimos aproximando hasta que los leds se pongan de color rojo (anotamos el valor) y así con todos los colores de la cartulina.

	CALIBRACIÓN EN DECIMAL		
Color	igual a 2402	[2402,2816]	[2816,10000]
blanco	8cm	7cm	2cm
verde	9cm	8cm	2cm
amarillo	9cm	8cm	2cm
rojo	10cm	9cm	2cm
azul	9cm	8cm	2cm
negro	2,5cm	2cm	0cm

Tabla 6. Mediciones del comportamiento del sensor en diferentes colores.

Los valores medidos se muestran en la Tabla 6. Como se observa, excepto con colores oscuros, como hemos explicado anteriormente, el funcionamiento del sensor es satisfactorio. Aunque los colores oscuros están bastante presentes en nuestra vida cotidiana, el color negro intenso no es tan frecuente (como excepción tendríamos el asfalto donde tendríamos problemas). De todas formas, se ha optado por indicarle al usuario que en ese tipo de suelo no es posible su utilización en modo autónomo.

Tras la caracterización realizada en este apartado podemos definir nuestro sensor con fiabilidad, ya que la hoja de datos nos da unos datos muy generales. En concreto:

- Empieza a detectar un objeto a los 10 cm, antes es muy poco preciso.
- No puede ser utilizado en superficies de cristal, ya que su detección es prácticamente nula.
- No se comporta adecuadamente en superficies muy oscuras o negras.

Estos resultados los tenemos que tener en cuenta para el software de alto nivel, ya que el usuario tiene que tener conocimiento de estos resultados para su utilización en diferentes entornos.

7. Diseño y fabricación de los sensores.

A continuación vamos a hablar del diseño del pcb donde alojaremos el sensor infrarrojo y los componentes. Para el diseño tenemos que saber los componentes que lo integran y sus conexiones, esto viene dado en la hoja de datos, pero no su colocación en el pcb, esto lo dispondremos como más nos convenga.

Se ha decidido llevar a cabo el diseño de un pcb para cada sensor individualmente para que la realización de pruebas sea más simple, y si es necesaria alguna modificación sea mucho más económica y fácil. Posteriormente, después de la realización del primero y su correcto funcionamiento, se bajará la posibilidad de hacer un único pcb con varios sensores.

El sensor VCNL4040 viene dado sin pines de soldadura, pero tiene pads de soldadura por debajo del integrado, para las conexiones con el circuito antes mostrado en la figura 18. Para ello, se diseña un pcb que se unirá a la placa base donde irá el resto de los componentes. El pcb a diseñar parte del circuito que nos dan en la hoja de datos del sensor:

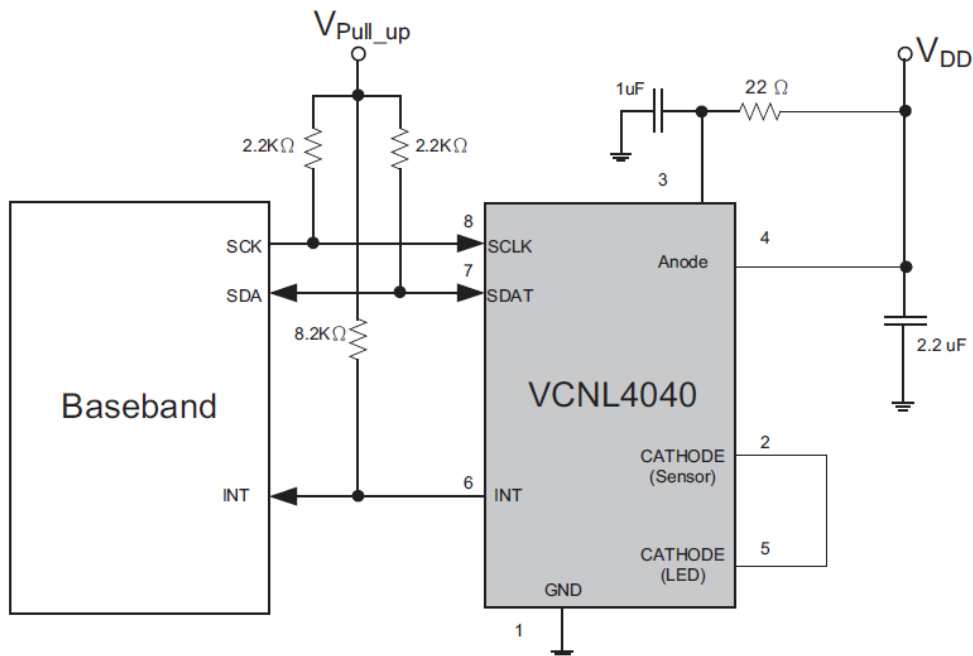


Figura 26. Conexiones requeridas para el sensor VCNL4040.

Como podemos ver en la figura 26, el circuito necesario para el sensor consta de cuatro resistencias y dos condensadores. La línea de datos y la línea de reloj llevan la misma resistencia, ya que se quiere que llegue el mismo voltaje en los dos casos por cada pulso que haya. Para el INT se pone diferente resistencia para que no llegue la misma tensión que en la línea de reloj y línea de datos, y no se puedan confundir. Los condensadores se colocarán para filtrar la señal y que no llegue con demasiado ruido. Por esa razón, los condensadores se conectan a tierra para poder filtrar la señal, de lo contrario no realizarían bien su función.

Este circuito se ha diseñado con el software de diseño KiCad, donde tenemos que realizar el modelo eléctrico del sensor VCNL4040 en el KiCad. Para realizar el modelo eléctrico, en la hoja de datos que proporciona el fabricante aparecen detalladas las dimensiones físicas del componente y de la huella (proyección de los pads de soldadura en el PCB) recomendada de dicho componente. La imagen 27 son las medidas reales del sensor VCNL4040.

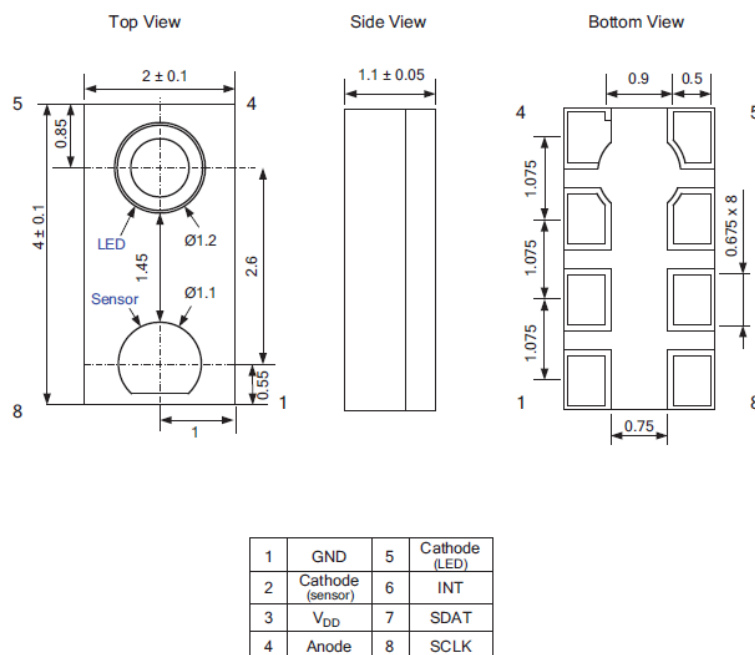


Figura 27. Medidas reales del sensor VCNL4040.

Con esto hacemos nuestro esquema en el KiCad, que nos da como resultado el circuito mostrado en la figura 28. A continuación pasaremos a hacer las huellas o encapsulados de cada componente y el ruteo del circuito, que serán “los cables” que unen los componentes entre sí.

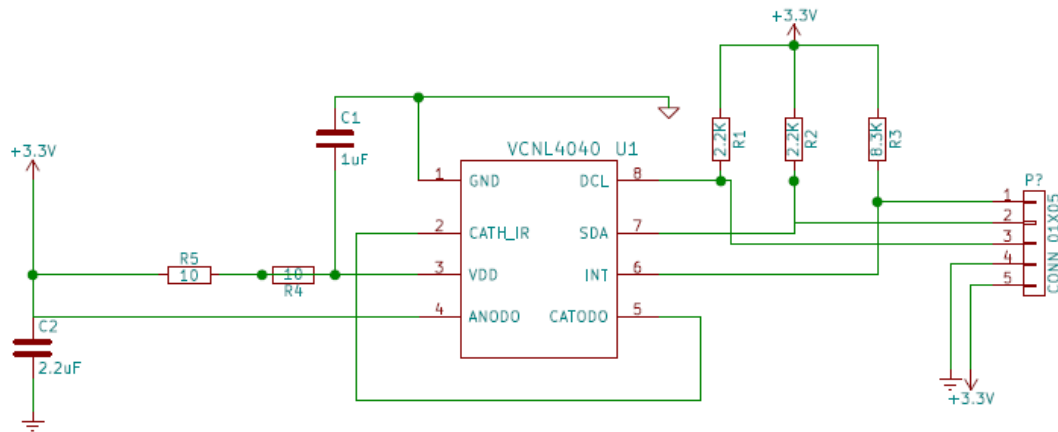


Figura 28. Circuito eléctrico del sensor VCNL4040 en el KiCad.

Para hacer una conversión entre el circuito eléctrico y el encapsulado de cada componente, tenemos que generar una lista que le asigna a cada componente del circuito eléctrico su encapsulado correspondiente, esta lista se denomina *netlist*. A continuación de la asignación de cada encapsulado a cada componente, se genera el pcb que aparecerá en otro archivo del KiCad donde aparecerán los componentes, pero en vez de como en la imagen anterior, con el encapsulado o huella asignada en la netlist. Estos componentes aparecerán desordenados y con unas líneas que simularán las conexiones que se les hicieron en el circuito electrónico. Ante esta situación, lo que tenemos que hacer es, primero ordenar los componentes en el lugar que queremos que estén en el pcb físicamente, y a continuación, hacer las pistas que relacionan cada componente con los demás. Para ello nos ayudamos de las líneas auxiliares antes mencionadas. Para la correcta realización de las pistas se requiere de gran experiencia, ya que tiene un gran número de especificaciones en cada caso, desde el grosor de la pista, el ángulo que puede formar, hasta la trayectoria más adecuada. Algunos de los requisitos utilizados para la realización de todos los pcs realizados en este trabajo son:

- Las pistas nunca pueden formar 90°, ni en su trayectoria ni con la unión con otras pistas. Esto es debido a las pistas perpendiculares no conducen bien la corriente.
- Es conveniente que las pistas sean lo más paralelas posibles entre ellas, es decir, si el número de pistas es muy amplio o salen de lugares muy próximos, es recomendable que se realicen de forma paralela para que no generen flujos magnéticos que dificulten la comunicación entre los componentes.

- Los componentes no deben estar muy próximos. Este requisito es por seguridad, ya que la huella del componente puede ser menor que el componente. Para ello dejamos un poco de holgura a su alrededor.
- El área del pin al que le asignemos el voltaje de referencia para toda la placa no debe ser muy pequeña para que este voltaje sea estable a lo largo de toda la placa.

Con todos estos requisitos en la realización de pistas y el posterior recuadro del área que va a tener nuestro sensor, obtenemos nuestro pcb que se muestra en la siguiente figura 29:

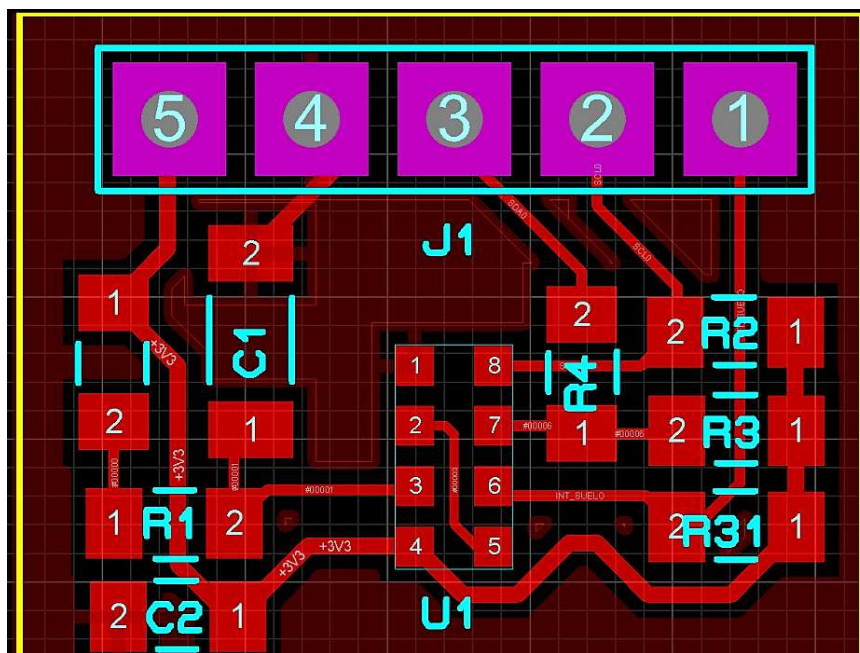


Figura 29. Circuito eléctrico para el PCB del sensor VCNL4040.

Hasta este punto se definiría el diseño del pcb, a continuación hablaremos de la fabricación del mismo.

Para la fabricación del pcb tenemos que exportar tres archivos del KiCad que posteriormente tendremos que usar en el Circuit Cam (programa para la fabricación de circuitería impresa) y a continuación en el Board Master. Estos tres archivos son de texto, y van divididos en los tres tipos de operaciones que se realizan para la fabricación del PCB. Los archivos de texto que contienen las características determinadas para:

- Las operaciones de taladrado. Se genera con el nombre del proyecto creado en el programa seguido de la palabra Drill.

- Las operaciones para la realización de las pistas. Se genera con el nombre del proyecto creado en el programa seguido de la palabra Cu.
- La operación de contorno, donde se corta nuestro pcb de la placa principal de la cual se fabrica (en esta operación no se corta todo el contorno, se dejan varios puntos fijos para que el pcb no se suelte de la placa principal ya que podría ser peligroso en los últimos movimientos por peligrar su fijación). Este archivo recibe el nombre del proyecto creado en el programa seguido de la palabra Edge-Cuts.

Se exportan estos tres archivos del KiCad y se importan en el programa Circuit Cam, en el cual, entre otras operaciones, se le asignan los lugares donde queremos que se corte el contorno para que quede fija a la placa principal, tratando de que sea fácil de quitar cuando las queramos sacar de la placa, o las fresas que se van a emplear para la fabricación.

A continuación se exportará al programa Board Master, que es el programa que controla la máquina LPKF ProtoMat C30s que se muestra en la figura 30, con unas características que se muestran en la tabla 7. En este programa se visualiza en la pantalla el lugar exacto donde queremos colocar el contorno de nuestro pcb diseñado dentro de la placa principal. Además de esto, podemos comprobar que está en el lugar que le asignamos, ya que tiene la opción de mover el cabezal de la máquina al lugar donde situamos físicamente nuestro diseño, para cerciorarnos que está exactamente donde queremos en la placa principal física.



Figura 30. Máquina LPKF ProtoMat C30s.

Área de trabajo	350x200 mm
Resolución	7,937 um
Precisión reproducible	+ / - 0,005 mm
Motor	5000 – 32000 rpm (variable)
Tamaño de pista mínimo	0,1 mm
Ancho mínimo de fresado	0,2 mm
Tamaño de perforación mínimo	0,3 mm

Tabla 7. Características técnicas de la LPKF ProtoMat C30.

Cuando ya tenemos colocado nuestro diseño del pcb en el lugar deseado, debemos ir siguiendo las diferentes operaciones de trabajo que nos indica el programa hasta obtener el pcb final.

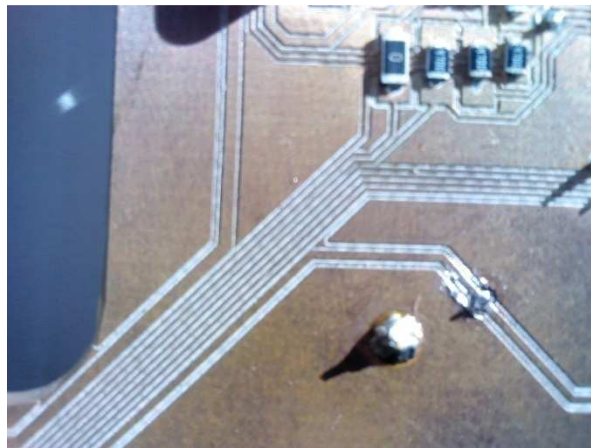


Figura 31. Detalle de las pistas realizadas por la maquina fresadora.

Después de este proceso, se continuará con el proceso de soldadura de los componentes. Este proceso se caracteriza por ser un proceso prácticamente en su totalidad manual y de mucha precisión. Para la soldadura de los componentes debemos saber previamente la temperatura máxima que es capaz de soportar cada uno. A continuación tenemos que añadir a cada huella o encapsulado una pasta que está compuesta principalmente por estaño. Esta pasta no es de fácil colocación ya que no se pega al pcb, además de que estamos hablando de unas dimensiones de menos de un milímetro en muchos casos, por lo que hay que tener mucha precisión en su colocación. Posteriormente colocaremos el componente en el lugar que le corresponde dentro del pcb diseñado (en su huella correspondiente y encima de la pasta previamente aplicada).

Finalmente, con la máquina de soldar, aplicaremos aire caliente en la zona a soldar hasta que la pasta se derrita y quede fijada tanto al componente como al pcb. En este proceso suele suceder un problema que con la experiencia se va reduciendo: el aire tiene poca potencia pero mucha temperatura y en el momento en que la pasta se funde se suele mover el componente. Esto se puede prevenir siendo muy rápidos a la hora de quitar el aire justo en el momento de fusión de la pasta. Este proceso se repetirá con cada componente, hasta tener todos los componentes soldados en sus correspondientes lugares.

Un dato a destacar del proceso de soldado, es que no es fácil ver a simple vista si el componente está bien soldado debido a que en algún caso hay pistas que van por el medio de algún componente y puede suceder que se soldase la pista del interior con el pin donde se coloca el componente.

Por último comprobaremos que todas las conexiones son correctas, que todos los componentes están bien soldados y hacen buen contacto. Estas comprobaciones se realizan con el voltímetro. Como resultado final obtenemos el PCB con sus componentes soldados, como se muestra en la figura 32.

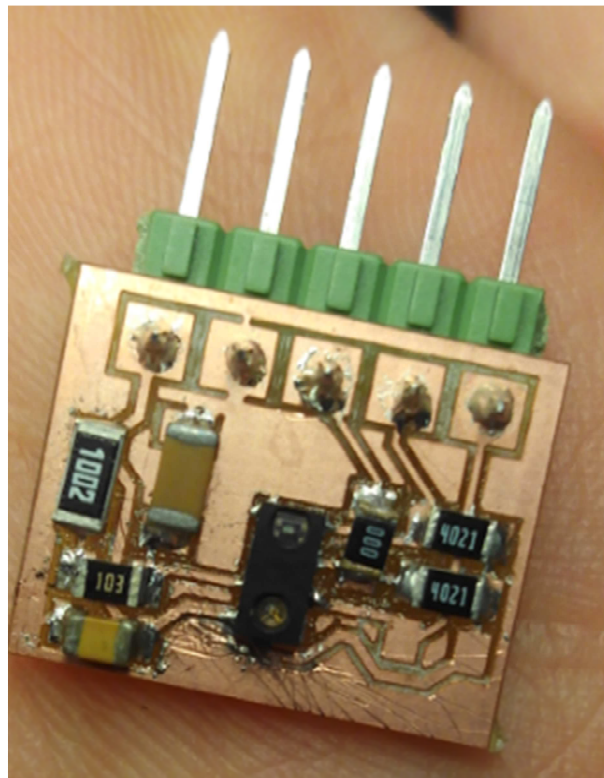


Figura 32. *Sensor VCNL4040 en su propio PCB.*

8. Ubicación de los sensores en la carcasa

Como se comentó en el apartado de requisitos previos, la carcasa de esta versión del ROBOBO (ver figura 33) ha sido modificada respecto a la del 1.0, por lo que los sensores ya no se pueden colocar en los mismos alojamientos. Además, la carcasa del Robobo 1.0 dejaba al descubierto los sensores, por lo que no había que hacerles un alojamiento a los leds para que pudieran emitir ondas hacia fuera. A continuación hablaremos del número más adecuado de sensores para el Robobo 2.0.

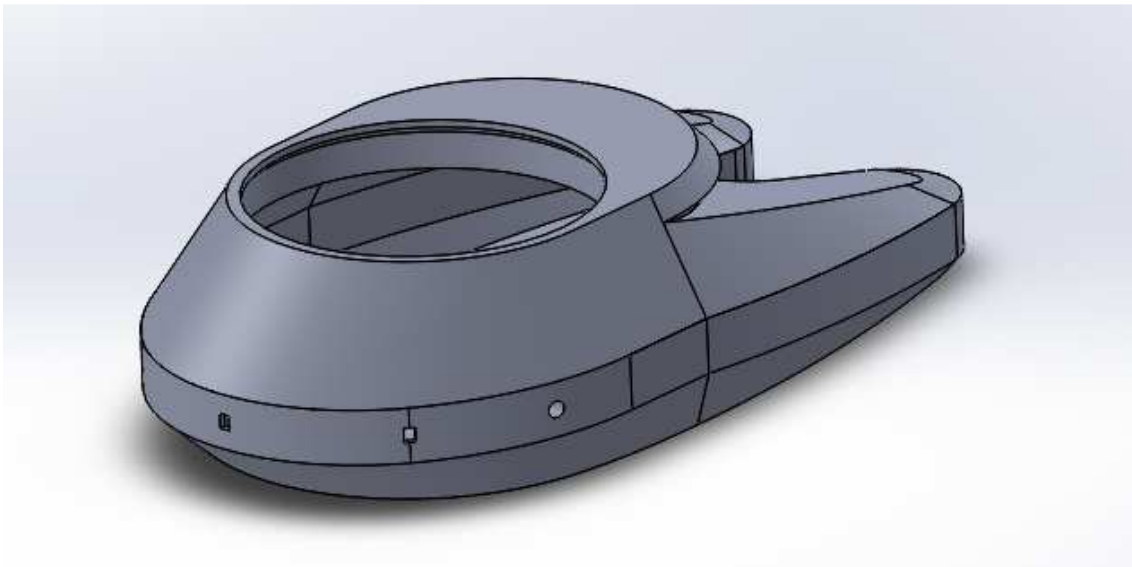


Figura 33. *Carcasa actual del ROBOBO 2.0.*

8.1. Elección de número de sensores.

Para la elección del número de sensores a utilizar, hay que tener tres variables fundamentales en total equilibrio. La primera sería el número de sensores que son capaces de conectarse al multiplexor, en nuestro caso ocho, por lo que las dos configuraciones más favorables serían 8 o 16 sensores, teniendo dos multiplexores. Por otro lado, y más importante, es la amplitud de mapeo. Esto quiere decir que en ningún momento pueda haber ningún objeto que no sea detectado, ya que esto implicaría un posible choque de consecuencias impredecibles. Por último, hay que tener en cuenta el coste total de la sensorización, ya que cuantos más sensores se utilicen más caro será el sistema.

Con el objetivo de que la detección de objetos alrededor del Robobo sea lo más amplia posible, crearemos un cono de detección como modelo representativo de la

respuesta de cada sensor. Para la creación de este cono nos guiaremos principalmente por las pruebas que hemos realizado en el apartado 6.4., donde se observaba que el sensor VCNL4040 ofrecía una distancia aceptable de detección de 10cm. El cono tiene una abertura respecto a la horizontal, información sacada de la hoja de datos, que se muestra en la figura 19 de este proyecto. Teniendo esto en cuenta hemos realizado un modelo en Solidworks (figura 34) que consiste en un cono que nos indica el espacio de detección que tiene el sensor. Posteriormente colocaremos estos conos en la carcasa, primero a grandes rasgos, y posteriormente de forma más precisa.

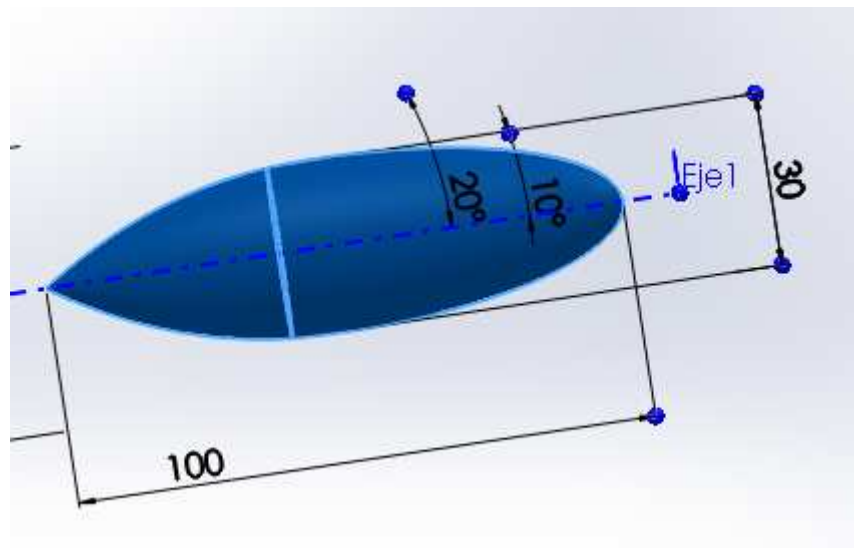


Figura 34. Cono de detección del sensor infrarrojo VCNL4040.

La colocación de estos conos en el Robobo se puede dividir en tres zonas de la carcasa:

- En la parte delantera. Donde es el mayor número de colisiones ya que el robot está diseñado para desplazarse normalmente hacia delante.
- En la parte trasera. Donde se colocarán también en los lados para cualquier choque con una pared, por una rotación.
- En el suelo. Donde podríamos hacer una subdivisión:
 - El suelo trasero aunque no es tan importante como el suelo delantero, porque el Robobo se ha diseñado para que se desplace principalmente en dirección frontal, es interesante para la rotación.

- El suelo de justo delante de las ruedas es el sitio más crucial, ya que es el único sitio de contacto con el suelo que pueda provocar la caída del Robobo, algo que hay que evitar a toda costa.

Otra tema de gran importancia, es la forma de colocación de los sensores en la carcasa, sean de realización sencilla, es decir, es importante tener en cuenta que todo lo que se puede realizar en el software no es posible realizarlo con la misma precisión en la realidad, por lo que es importante que los sensores sean de fácil colocación para nuestra carcasa. Esto se logra haciendo que los sensores queden en paralelo a las superficies de la carcasa, por lo que con un simple apoyo en su interior ya sería posible su colocación.

Considerando las singularidades de estas zonas y sus requisitos, hemos situado los sensores de modo que los conos de detección cubran la mayor área posible tratando de maximizar la zona de detección, dando como resultado el modelo 3D de la figura 35 y 36.

A continuación, explicaremos porque se ha decidido que el número de sensores en cada zona sea el asignado. Posteriormente concretaremos los huecos que se le deben realizar a la carcasa para la salida de la emisión de ondas del sensor y su posterior recepción para un correcto funcionamiento.

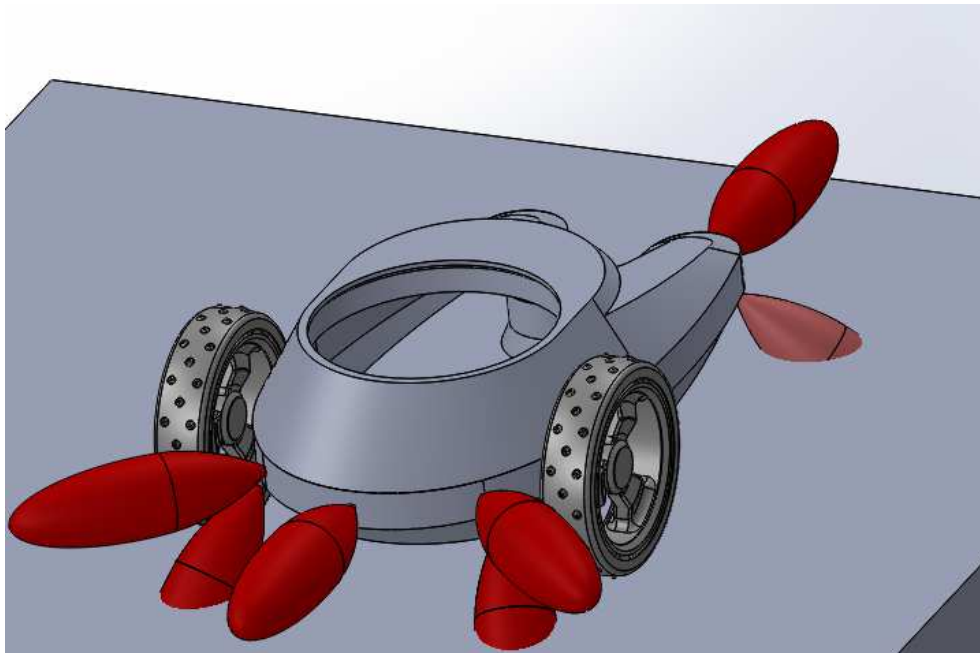


Figura 35. Colocación de los conos de detección en la carcasa del ROBOBO.

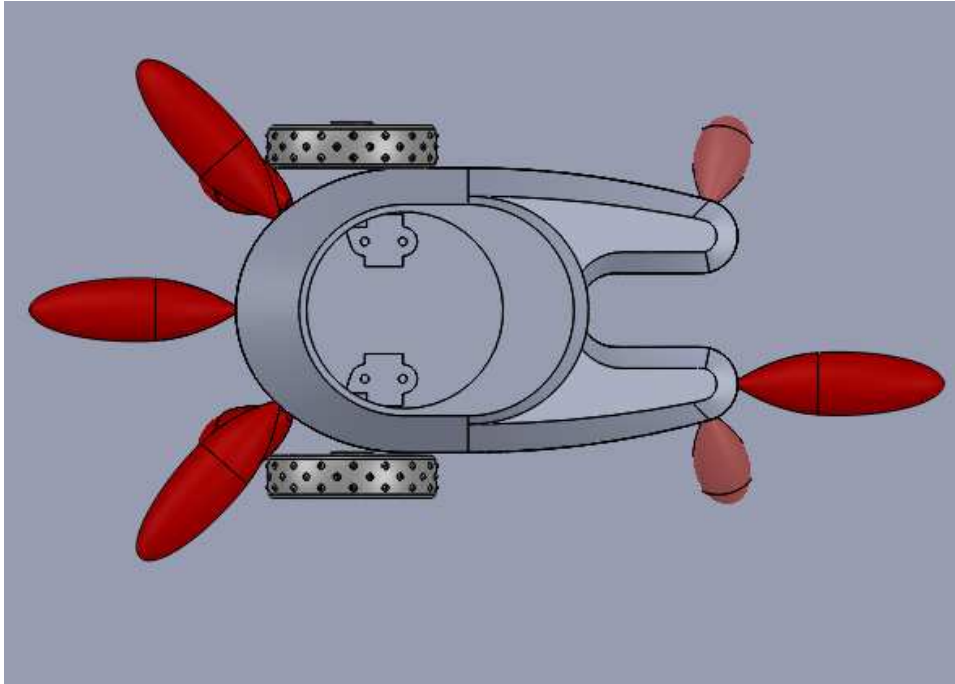


Figura 36. Vista en planta de los conos de detección de los sensores en la carcasa.

Las figuras 37, 38, 39 y 40 muestran la colocación final con mayor nivel de detalle:

- Parte delantera (ver figura 37)→ 3 sensores colocados de tal manera que no haya ninguna franja con posibilidad de choque. El Robobo se mueve principalmente hacia delante, por lo que lo más probable es que pueda chocarse contra alguna pared, mueble, rodapié o esquinas de alguna pared, por eso se han colocado de tal manera que no sea posible que pueda chocar contra ninguna superficie.

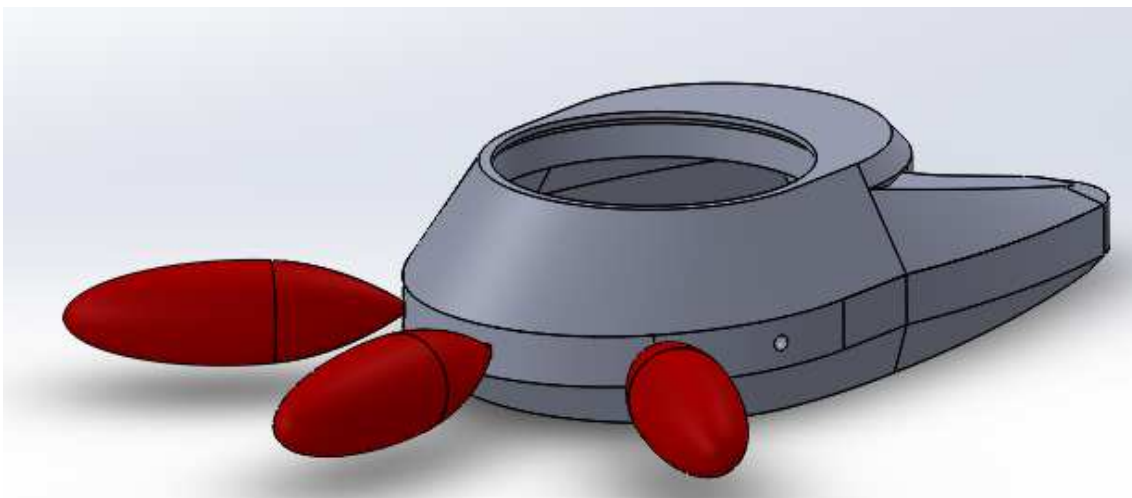


Figura 37. Conos de detección de la zona frontal en la carcasa.

- Parte trasera (ver figura 38)→ un sensor mirando hacia atrás, sólo uno. Esto es debido porque solo está pensado para un movimiento de colisión con una pared trasera cuando caminara hacia atrás, algo muy improbable como hemos mencionado previamente. Teniéndonos que ajustar el número a 8 infrarrojos, para el uso de un único multiplexor y microcontrolador, nos hemos decidido por el uso de un único sensor en la parte trasera, aunque es asimétrico y para las rotaciones sería conveniente el uso de otro sensor en el lado derecho.

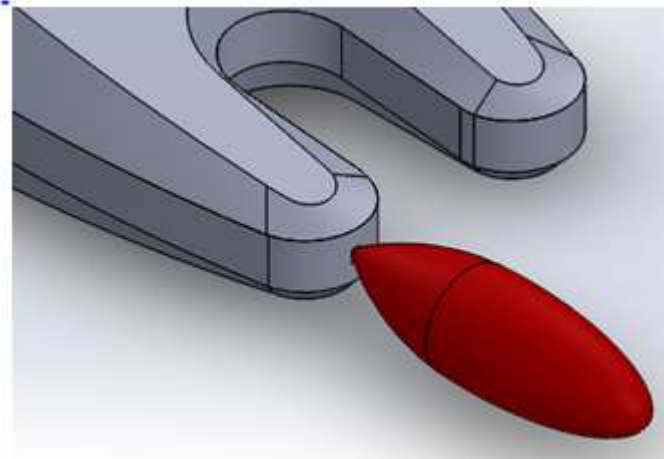


Figura 38. *Cono de detección trasera.*

- Delante de las ruedas (ver figura 39)→ estos sensores son colocados al igual que los otros, con la inclinación que tiene la carcasa para una fácil colocación. Como podemos intuir en la figura 38, la zona de visión del sensor queda un poco más hacia el interior que la rueda, aunque esto no es importante por dos motivos:
 - El Robobo no está diseñado para andar sobre lugares donde no tenga suelo por dentro de sus ruedas y sí en la superficie donde se apoyan las ruedas.
 - El motivo más importante es que aunque no hubiera suelo en medio de las ruedas, el sensor lo detectaría, ya que su percepción del suelo variaría del número fijado a esa determinada distancia e inclinación.

En este caso no son necesarios más sensores, ya que el área de detección es muy pequeña y precisa, y lo más importante es una colocación cercana al suelo.

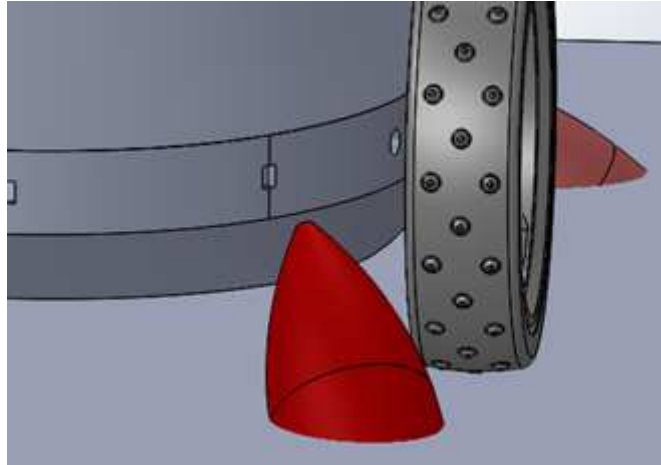


Figura 39. Cono de detección delante de las ruedas.

- Suelo trasero (conos rojo claro de la figura 40) → estos dos sensores no son tan complicados de ubicar, ya que simplemente evitan que en un giro el robot no se caiga de la superficie. Como comentamos anteriormente, cuanto más cerca se coloquen del suelo, más brusco será el cambio del valor dado por el sensor y mejor respuesta dará.

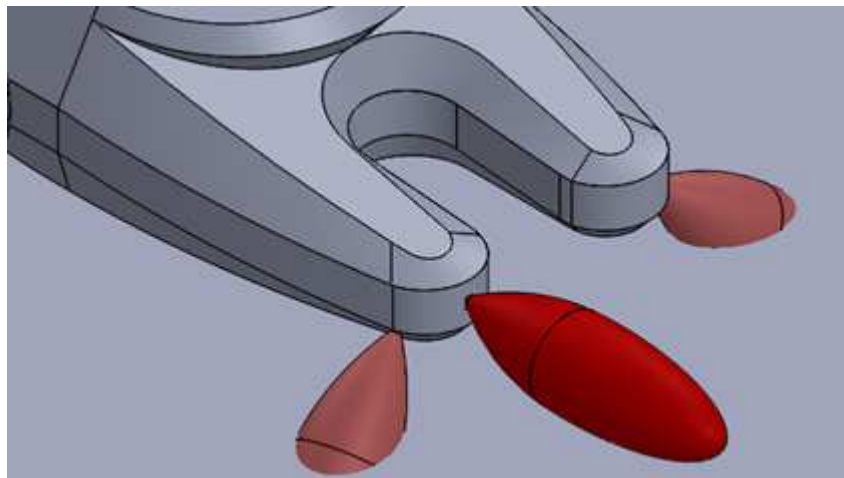


Figura 40. Conos de detección al suelo trasero.

A partir de esta primera ubicación se realizará la colocación de los sensores reales en la carcasa. Aunque no es analizado el espacio interno necesario para el sensor propiamente dicho.

8.2. Huecos en la carcasa para la salida y recepción de ondas del sensor.

A continuación hablaremos del lugar y la forma que tienen que tener es los huecos donde se alojaran los sensores propiamente dichos (ver figura 41).

En primer lugar, para que la onda que emite el sensor se propague correctamente, es adecuado que el hueco o agujero donde se van a alojar los sensores, tenga forma de cono, ya que la carcasa tiene un espesor determinado que hace que se pueda hacer un alojamiento cónico. El ángulo de inclinación que le hemos dado al cono es de 45° , ángulo suficiente para una correcta propagación de las ondas. La forma que se le ha puesto es redondeada, para poder realizarse más fácilmente en la impresora 3D, que en uno de las características del Robobo 2.0 que ya hemos mencionado anteriormente.

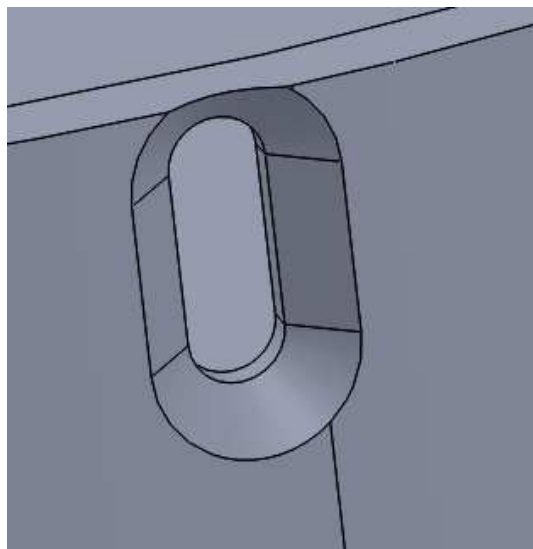


Figura 41. Forma de la salida del sensor en la carcasa cónica.

Como resultado final, tenemos la carcasa que se muestra en las figuras 42 y 43, donde los huecos están marcados por círculos rojos:

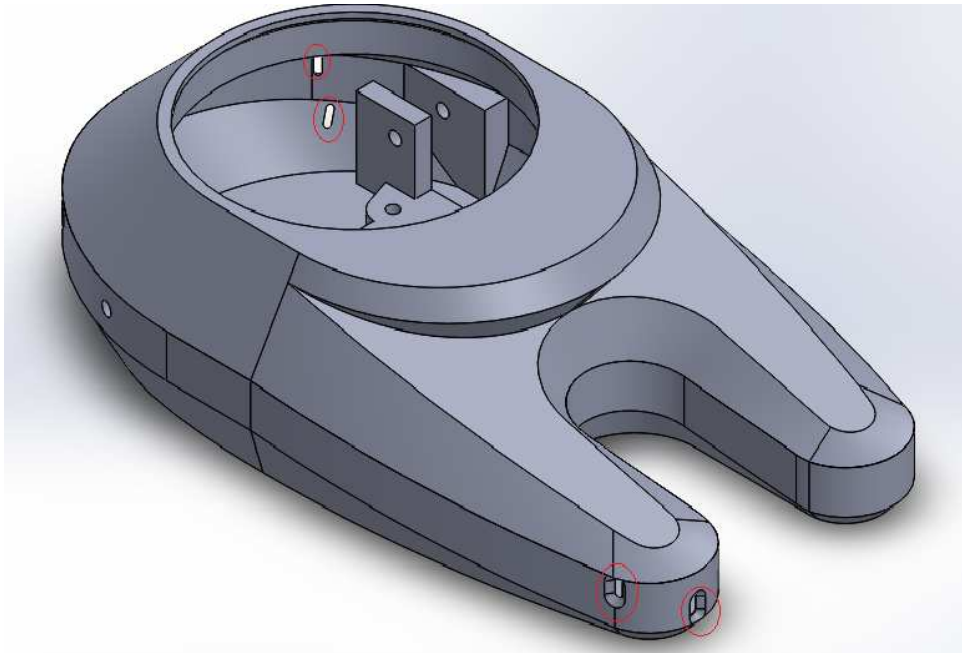


Figura 42. Alojamiento donde se colocarán los sensores.



Figura 43. Alojamiento de la parte delantera donde se colocarán los sensores.

Como se puede observar los sensores traseros laterales no están colocados en el mismo sitio, esto es debido a que durante la realización del alojamiento, nos hemos dado cuenta de que no entra en el interior el circuito integrado, por lo que lo hemos tenido que cambiar a la parte de arriba, esto quiere decir que no va a detectar el suelo en la parte trasera, sino, las paredes en las rotaciones, esto puede ser modificado posteriormente, con el cambio de la forma de la carcasa.

Otro dato curioso, es que el cono de propagación es rompe en la parte superior, como se puede observar en la figura 43, esto es debido a donde tenemos que colocar el circuito integrado en su interior, donde queda situado el sensor dentro del circuito y por

la propagación del cono. Esto también podría ser modificado para las siguiente versión, modificando la carcasa ay haciéndola más ancha, o modificando el circuito integrado y colocando el sensor más cerca de los pines en el pcb.

A continuación colocaremos los sensores en sus lugares exactos, y la el cono de detección. Esto se hace para cerciorarnos que los primeros conos y los últimos van en la misma dirección, ya que, como dijimos anteriormente, los primeros conos estaban colocados a grandes rasgos. En la figura 44 tenemos los sensores colocados, con el cono de detección colocado justo donde el sensor y todo ello colocado en el ensamblaje del Robobo.

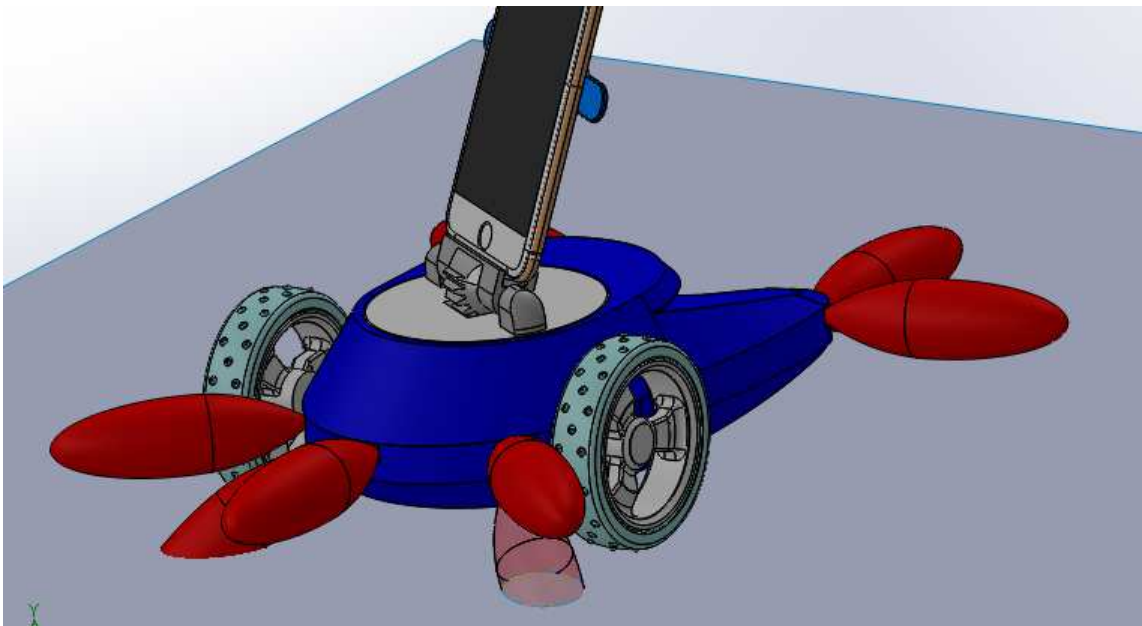


Figura 44. *Sensores colocados el sus ubicaciones correspondientes, con el cono de detección colocado.*

A modo de resumen, tal y como se muestra en las figuras 45, 46 y 47, podemos concluir:

- El cono de detección no choca en ningún momento con la carcasa.
- El área de detección del sensor delante de las ruedas este bien situado.
- Ningún cono toca con las ruedas.

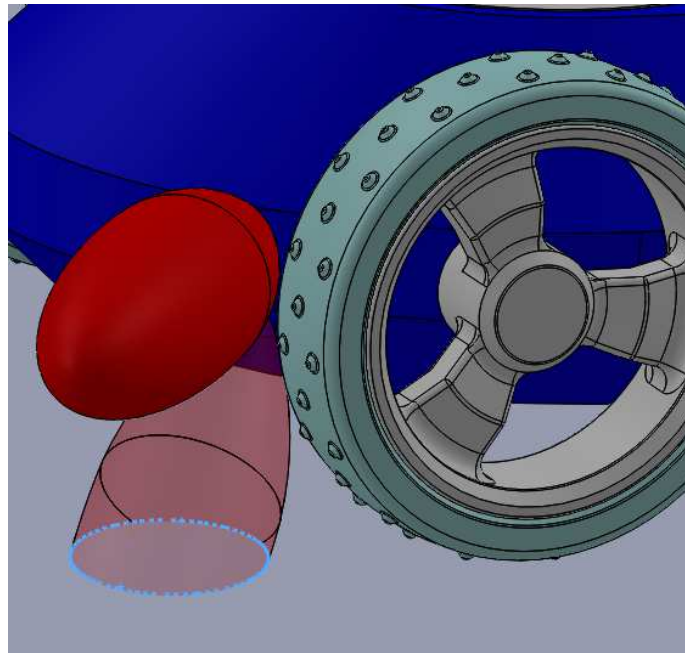


Figura 45. Área de detección del sensor en el suelo delante de las ruedas.

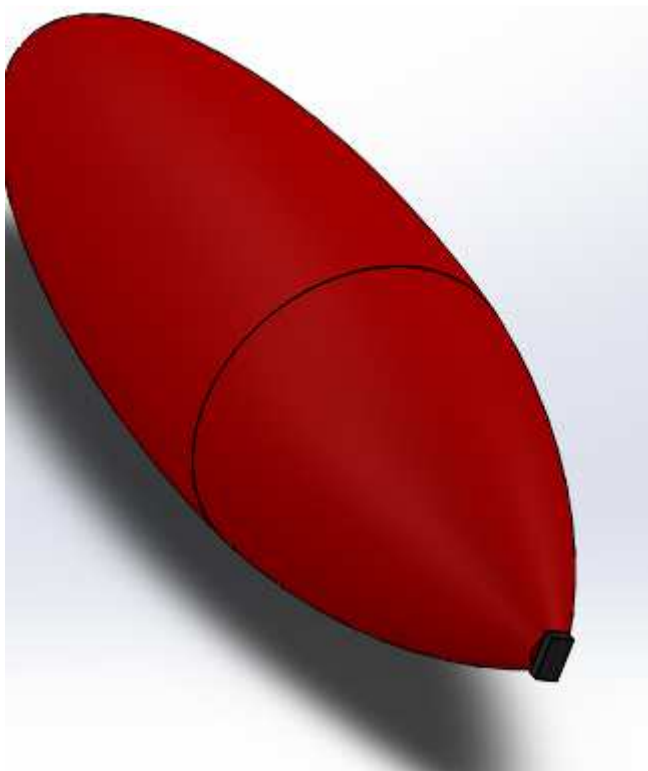


Figura 46. Cono de detección situado en el sensor

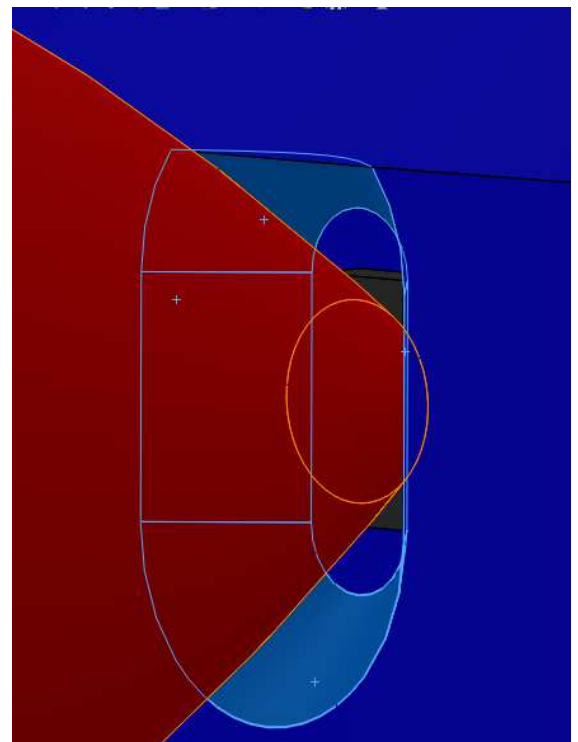


Figura 47. Sensor colocado en su hueco dentro de la carcasa, con el cono de detección.

9. El microcontrolador.

Para la realización de la comunicación entre el microcontrolador y el sensor, es necesario tener unos conocimientos previos muy amplios sobre diversos temas, como son las estructuras de los microcontroladores y su funcionamiento interno, concretamente los de la familia de microcontroladores PIC32, que es el utilizado y seguidamente de las características específicas del microcontrolador utilizado. Cuando se ha asimilado toda esta información, es necesario conocer el protocolo de comunicación entre el sensor y el microcontrolador que será el protocolo de comunicación I2C. Este último punto ya será analizado el apartado 10 extensamente.

9.1. Fundamentos teóricos. Microcontroladores.

Los microcontroladores son computadores digitales integrados en un chip que cuentan con un microprocesador o unidad de procesamiento central (CPU), una memoria para almacenar el programa, una memoria para almacenar datos y puertos de entrada salida. A diferencia de los microprocesadores de propósito general, como los que se usan en los computadores PC, los microcontroladores son unidades autosuficientes y más económicas.

El funcionamiento de los microcontroladores está determinado por el programa almacenado en su memoria. Este puede escribirse en distintos lenguajes de programación. Además, la mayoría de los microcontroladores actuales pueden reprogramarse repetidas veces.

Por las características mencionadas y su alta flexibilidad, los microcontroladores son ampliamente utilizados como el “cerebro” de una gran variedad de sistemas que controlan máquinas, con innumerables aplicaciones: industriales, robótica, domótica, equipos médicos, sistemas aeroespaciales, e incluso dispositivos de la vida diaria como automóviles, hornos de microondas, teléfonos y televisores.

Las principales características de los microcontroladores son:

- **Unidad de Procesamiento Central (CPU):** Típicamente de 8 bits, pero también las hay de 4, 32 y hasta 64 bits con *arquitectura Harvard*, con memoria/bus de datos separada de la memoria/bus de instrucciones de programa, o *arquitectura*

de von Neumann, también llamada *arquitectura Princeton*, con memoria/bus de datos y memoria/ bus de programa compartidas.

- **Memoria de Programa:** Es una memoria ROM, EPROM, EEPROM o Flash que almacena el código del programa que típicamente puede ser de 1 kilobyte a varios megabytes.
- **Memoria de Datos:** Es una memoria RAM que típicamente puede ser de 1, 2 4, 8, 16, 32 kilobytes.
- **Generador del Reloj:** Usualmente un cristal de cuarzo de frecuencias que genera una señal oscilatoria de entre 1 a 40 MHz, o también resonadores o circuitos RC.
- **Interfaz de Entrada/Salida:** Puertos paralelos, UARTs, I2C, SPIs, CAN y USB.
- **Otras opciones:**
 - Conversores Analógicos-Digitales para convertir un nivel de voltaje en un cierto pin a un valor digital manipulable por el programa del microcontrolador.
 - Moduladores por Ancho de Pulso para generar ondas cuadradas de frecuencia fija pero con ancho de pulso modificable.

9.2. Los microcontroladores PIC.

Tras haber analizado las características que presentan los microcontroladores más comunes del mercado, se nos ha dado el microcontrolador de la familia del fabricante Microchip, por ser una empresa líder mundial en la fabricación de microcontroladores y por presentar haber una cantidad de documentación de los mismos, programas de ejemplo, etc. Por lo tanto, este capítulo se centra en el estudio de los microcontroladores PIC del fabricante Microchip. Se detallan aspectos como las características generales de este tipo de microcontroladores, las distintas familias de PIC existentes y las ventajas que conlleva su uso. También se analizan las principales características del PIC32MXXX, que es el modelo de PIC que se ha utilizado para la implementación del sistema de control de motores, de dispositivos ópticos y luminosos y para la comunicación con el Smarthphone.

Antes de hablar de las características de los PIC, es necesario saber que los microcontroladores PIC son una familia de microcontroladores tipo RISC(del inglés

Reduced Instruction Set Computer, Computadora con Conjunto de Instrucciones Reducido) fabricados por Microchip Technology Inc. y derivados del PIC1650. Para una completa comprensión de los PIC necesitamos tener claro que significan que un microcontrolador sea de tipo RISC, pues bien, RISC se centra en la obtención de procesadores con las siguientes características fundamentales:

- Instrucciones de tamaño fijo.
- Pocas instrucciones.
- Sólo las instrucciones de carga y almacenamiento acceden a la memoria de datos.
- Número relativamente elevado de registros de propósito general.

Una de las características más destacables de este tipo de procesadores es que posibilitan el paralelismo en la ejecución, y reducen los accesos a memoria.

A continuación hablaremos de las características de los microcontroladores PIC:

a) La arquitectura del procesador sigue el modelo Harvard.

La arquitectura Harvard (figura 48), consiste simplemente en un esquema en el que la CPU está conectada a dos memorias por medio de dos buses separados. Una de las memorias contiene solamente las instrucciones del programa, y es llamada Memoria de Programa. La otra memoria solo almacena los datos y es llamada Memoria de Datos. Ambos buses son totalmente independientes y pueden ser de distintos anchos. Para un procesador de Set de Instrucciones Reducido, o RISC, el set de instrucciones y el bus de la memoria de programa pueden diseñarse de manera tal que todas las instrucciones tengan una sola posición de memoria de programa de longitud. Además, como los buses son independientes, la CPU puede estar accediendo a los datos para completar la ejecución de una instrucción, y al mismo tiempo estar leyendo la próxima instrucción a ejecutar. Se puede observar claramente que las principales ventajas de esta arquitectura son:

- El tamaño de las instrucciones no está relacionado con el tamaño de los datos, y por lo tanto puede ser optimizado para que cualquier instrucción ocupe una sola posición de memoria de programa, logrando así mayor velocidad y menor longitud de programa.

- El tiempo de acceso a las instrucciones puede superponerse con el de los datos, logrando una mayor velocidad de operación. Una pequeña desventaja de los procesadores con arquitectura Harvard, es que deben poseer instrucciones especiales para acceder a tablas de valores constantes que pueda ser necesario incluir en los programas, ya que estas tablas se encontraran físicamente en la memoria de programa (por ejemplo en la EPROM de un microprocesador).

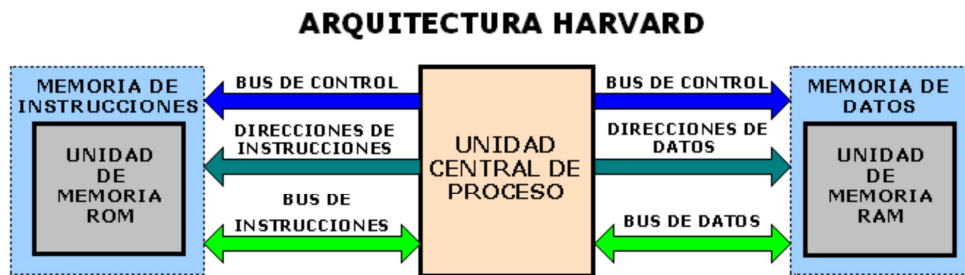


Figura 48. Esquema arquitectura Harvard.

b) Temporizadores o “Timers”.

Se emplean para controlar periodos de tiempo (temporizadores) y para llevar la cuenta de acontecimientos que suceden en el exterior (contadores).

Para la medida de tiempos se carga un registro con el valor adecuado y a continuación dicho valor se va incrementando o disminuyendo, dependiendo de si empleamos el timer como temporizador o como contador, al ritmo de los impulsos de reloj o algún múltiplo hasta que se desborde y llegue a 0, momento en el que se produce un aviso.

Cuando se desean contar acontecimientos que se materializan por cambios de nivel o flancos en alguna de las patitas del microcontrolador, el registro se va incrementando o disminuyendo al ritmo de dichos impulsos.

c) Perro guardián o “Watchdog”.

Cuando el computador personal se bloquea por un fallo del software u otra causa, se pulsa el botón del reset y se reinicializa el sistema. Pero un microcontrolador funciona sin el control de un supervisor y de forma continuada las 24 horas del día. El Perro guardián consiste en un temporizador que, cuando se desborda y pasa por 0, provoca un reset automáticamente en el sistema.

Se debe diseñar el programa de trabajo que controla la tarea de forma que refresque o inicialice al Perro guardián antes de que provoque el reset. Si falla el programa o se bloquea, no se refrescará al Perro guardián y, al completar su temporización, provocará el reset.

d) Protección ante fallo de alimentación o “Brownout”.

Se trata de un circuito que resetea al microcontrolador cuando el voltaje de alimentación (VDD) es inferior a un voltaje mínimo (“brownout”). Mientras el voltaje de alimentación sea inferior al de brownout el dispositivo se mantiene reseteado, comenzando a funcionar normalmente cuando sobrepasa dicho valor.

e) Estado de reposo ó de bajo consumo.

Son abundantes las situaciones reales de trabajo en que el microcontrolador debe esperar, sin hacer nada, a que se produzca algún acontecimiento externo que le ponga de nuevo en funcionamiento. Para ahorrar energía, (factor clave en los aparatos portátiles), los microcontroladores disponen de una instrucción especial (SLEEP en los PIC), que les pasa al estado de reposo o de bajo consumo, en el cual los requerimientos de potencia son mínimos. Al activarse una interrupción ocasionada por el acontecimiento esperado, el microcontrolador se despierta y reanuda su trabajo.

f) Conversor A/D.

Los microcontroladores que incorporan un conversor A/D pueden procesar señales analógicas. Suelen disponer de un multiplexor que permite aplicar a la entrada del conversor diversas señales analógicas desde las pines del circuito integrado.

g) Comparador analógico.

Algunos modelos de microcontroladores disponen internamente de un Amplificador Operacional que actúa como comparador entre una señal fija de referencia y otra variable que se aplica por una de las patitas de la cápsula. La salida del comparador proporciona un nivel lógico 1 ó 0 según una señal sea mayor o menor que la otra.

También hay modelos de microcontroladores con un módulo de tensión de referencia que proporciona diversas tensiones de referencia que se pueden aplicar en los comparadores.

h) Modulador de anchura de impulsos o PWM.

Son circuitos que proporcionan en su salida impulsos de anchura variable, que se ofrecen al exterior a través de los pines del encapsulado. Además de que normalmente es utilizado para el control de velocidad de los motores tanto de paso a paso como de continua.

i) Puertas de E/S digitales.

Todos los microcontroladores destinan algunas de sus pines a soportar líneas de E/S digitales. Por lo general, estas líneas se agrupan de ocho en ocho formando puertos. Las líneas digitales de los puertos pueden configurarse como entrada o como salida cargando un 1 ó un 0 en el bit correspondiente de un registro destinado a su configuración.

j) Puertos de comunicación.

Con objeto de dotar al microcontrolador de la posibilidad de comunicarse con otros dispositivos externos, otros microcontroladores, buses de sistemas... Se dispone en el microcontrolador de los puertos de comunicación

Algunos modelos disponen de recursos que permiten directamente esta tarea, entre los que destacan:

- USART, adaptador de comunicación serie síncrona y asíncrona
- Puerto paralelo esclavo para poder conectarse con los buses de otros microcontroladores.
- USB.
- Bus I2C.
- CAN, para permitir la adaptación con redes de conexión multiplexado desarrollado conjuntamente por Bosch e Intel para el cableado de dispositivos en automóviles.
- SPI es el bus estándar de comunicaciones.

9.3. Microcontrolador PIC32MX534F064H.

El microcontrolador dado es el PIC32MX534F064H en una placa de desarrollo MINI32 fabricada por mikroelektronika (figura 49), encapsulado es de tipo TQFP (Thin Quad Flat Package), perteneciente a la familia de microcontroladores de 32 bits. Se

decidió la incorporación de este microcontrolador por ser una alternativa económica de alta velocidad de procesamiento con diversas interfaces de comunicación y multitud de puertos de entrada/salida, a lo que se suma la experiencia de uso en otras aplicaciones similares. Y ya por último, decir que tanto el entorno de desarrollo como el compilador para el PIC32 son gratuitos.

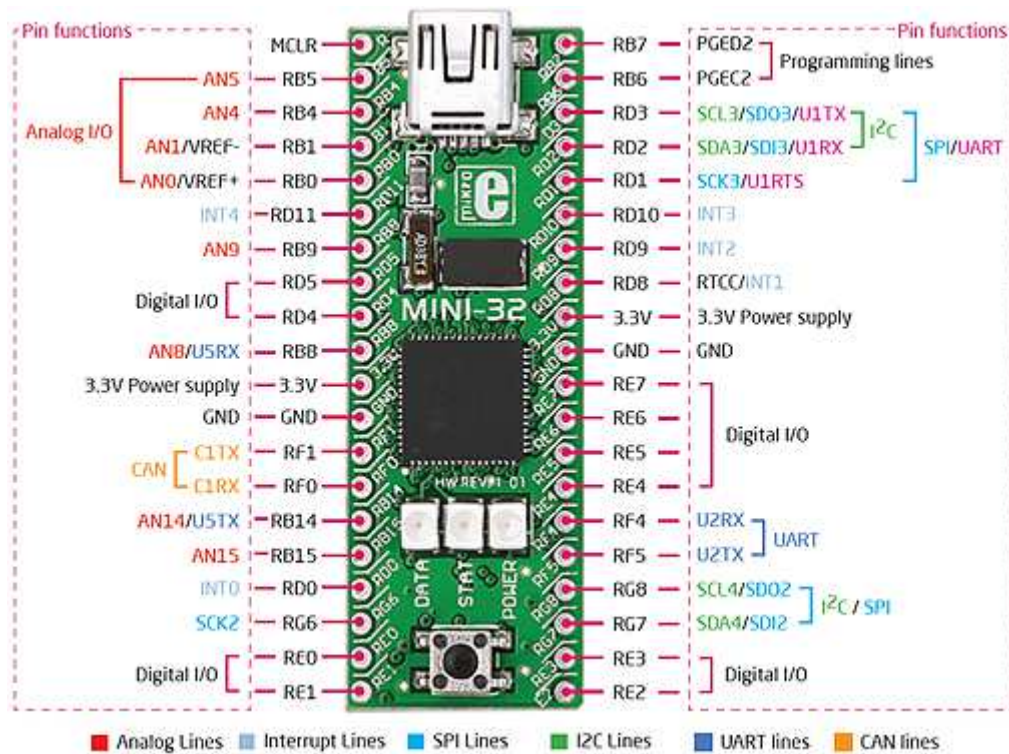


Figura 49. Placa de desarrollo MINI32 con un PIC32MX534F064H.

Este microcontrolador trabaja hasta una frecuencia de 80MHz, tiene una memoria RAM de 64 KB, 16 entradas y salidas (uno de los factores cruciales para su elección), dispone unos rangos de trabajo en voltaje de 2.3 V a 3.6 V y de temperatura -40° a 105°. Posee un controlador DMA de 4 canales para USB OTG (USB On-To-Go), memoria flash de hasta 512 kb, además de poseer conectividad USB periférica, una conexión CAN 2.0b con 1024 buffers de transmisión/recepción. Dispone de un procesador MIPS M4K Core de 32 bits. Tiene 256 bytes de cache y 64 pines de conexión.

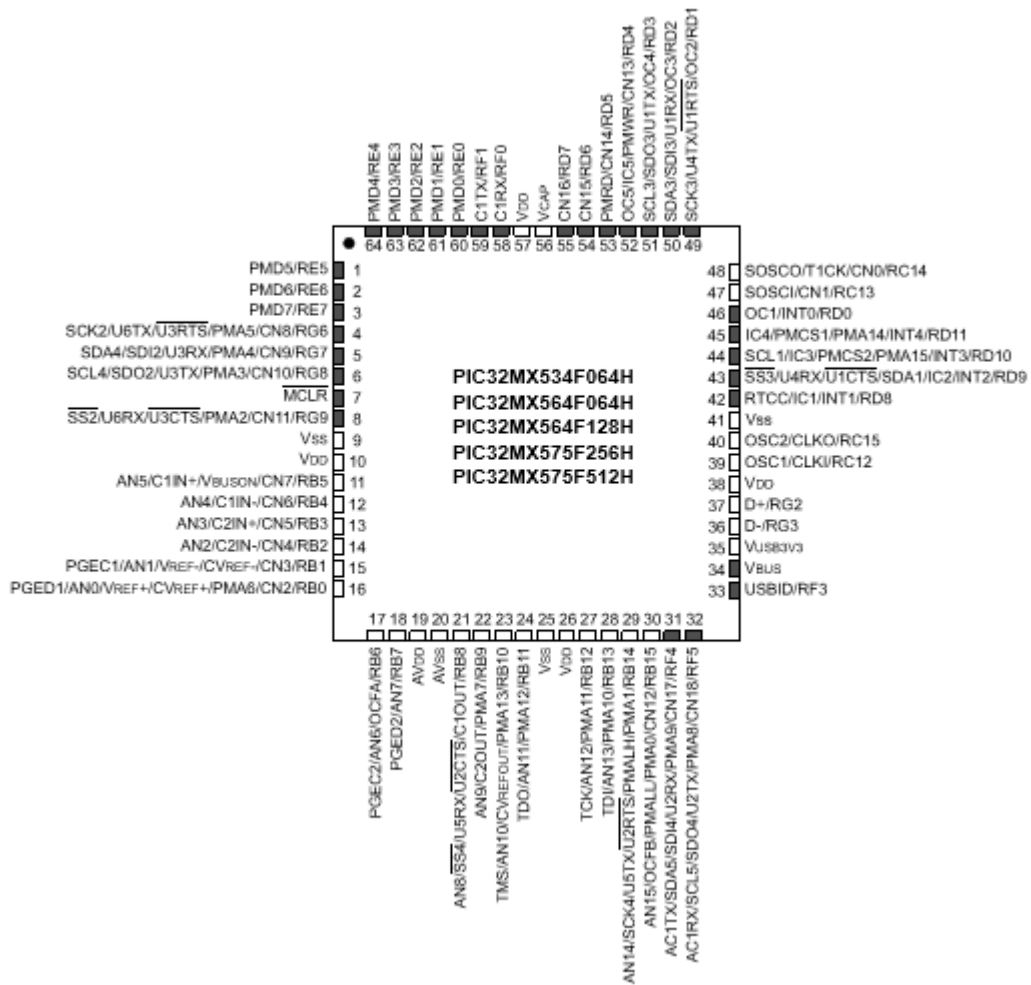


Figura 50. *Micontrolador PIC32MX534F064H.*

Las conexiones necesarias para los infrarrojos son 5, en las que se conectarán al microcontrolador por los pines:

- Línea de datos (pin RG8).
- Línea de reloj (pin RG9).
- Alimentación a 3.3 V (Pin 9).
- GND (pin 10).
- INT (pin RD0).

10. Programación del microcontrolador.

En este apartado comenzaremos explicando el protocolo de comunicación I2C (el utilizado en nuestro caso entre el microcontrolador y el sensor), que por medio de dos buses (uno línea de reloj “SCL” y otro línea de datos “SDA”) realiza toda la comunicación. A continuación se explicará el protocolo de comunicación, es decir, el orden en el que tienen que enviarse y recibirse las órdenes para una correcta comunicación entre el microcontrolador (“master”) y el sensor (“esclavo”). Y por último hablaremos del código realizado con toda esta información.

10.1. Fundamentos teóricos. I2C.

La comunicación entre el PIC32 y los dispositivos ópticos, se realiza por medio de los buses I2C de que dispone el PIC. La comunicación se realiza de este modo porque es el sistema de comunicación predeterminado de los VCNL4040. A continuación se detallan las características y el funcionamiento de éste bus.

El bus I²C es un bus de comunicaciones en serie. La versión 1.0 data del año 1992 y la versión 2.1 del año 2000, su diseñador es Philips. La velocidad es de 100Kbits por segundo en el modo estándar, aunque también permite velocidades de 3.4 Mbit/s.

La principal característica del bus I2C es que lo constituyen solo dos líneas, llamadas SCL y SDA:

- SCL (*Serial CLock*) es la señal de reloj y se usa para sincronizar todas las transferencias de datos sobre el bus I2C.
- SDA (*Serial Data*) es la línea de datos.

Ambas líneas SCL & SDA están conectadas con todos los dispositivos que haya en el bus I2C. Es necesario un tercer cable que es la línea de tierra o 0 v. También es necesaria una línea de alimentación de 3.3V.

SDA y SCL son “open drain” drivers, lo que quiere decir que el chip puede producir su salida a nivel bajo, pero no a nivel alto. Para que la línea pueda activarse a nivel alto hay que introducir resistencias pull-up conectadas a la fuente de alimentación de referencia del bus, normalmente 3.3 V, como ilustra la figura 51:

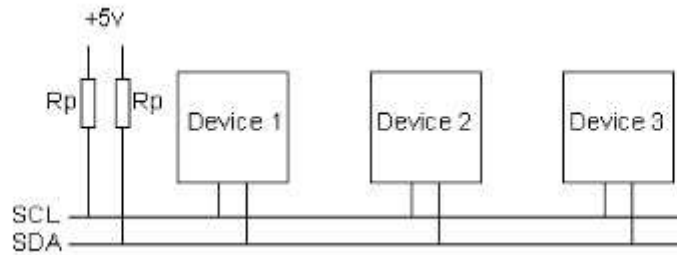


Figura 51. Esquema de funcionamiento del bus I2C.

El valor de las resistencias de pull-up no es estricto, y depende de la velocidad a la que queramos o podamos trabajar.

a) Maestros y esclavos

Los dispositivos interconectados a un bus I2C son siempre o maestros o esclavos. El maestro siempre es el dispositivo que maneja la señal de reloj y los esclavos son los dispositivos que responden al maestro. Un esclavo no puede iniciar una transferencia sobre el bus I2C, solo un maestro lo puede hacer. Puede haber, y generalmente es así, múltiples esclavos conectados al bus y un único maestro, aunque también es posible tener varios maestros (sistema multimaestro).

En un robot, el maestro será el controlador (PIC) y los esclavos serán los diferentes módulos o dispositivos. Un esclavo nunca iniciará una transferencia. Tanto el maestro como el esclavo pueden transferir datos sobre el bus, pero la transferencia siempre es controlada por el maestro.

b) Protocolo del bus I2C

El bus I2C tiene un número de “condiciones” que indican cuando una transferencia ha empezado, ha terminado, cuando se ha enviado un acknowledgement.... Estas condiciones son:

- Start condition y Stop condition:

Todas las transacciones comienza con una condición de inicio Start condition (S) y terminan con una condición de final, Stop condition (P).

Una transición en la línea de SDA de nivel alto a nivel bajo mientras SCL se mantiene alto, representa una Start condition.

Una transición en la línea de SDA de nivel bajo a nivel alto mientras SCL se mantiene alto, representa una Stop condition.

El paso de un nivel a otro en la línea de SDA y de SCL puede verse en la figura 52.

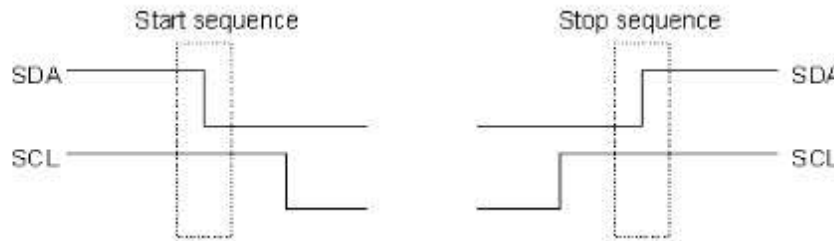


Figura 52. Start y Stop condition.

Una Start-Stop condition son siempre generadas por el maestro. Se considera que el bus está ocupado después de una condición de Start y se considera que está libre después de una Stop condition. Así mismo, el bus se mantiene ocupado si se repite una condición de Start en lugar de una Stop condition.

- Restart condition (R):

Una Restart condition ocurre cuando una fuente quiere transmitir más datos pero no quiere liberar la línea (figura 53). Esto ocurre una se ha enviado una condición de Start pero no ha ocurrido un Stop condition. Esto previene que otras líneas puedan ocupar el bus entre la transferencia de datos. También se puede enviar un Stop seguido de un Start.

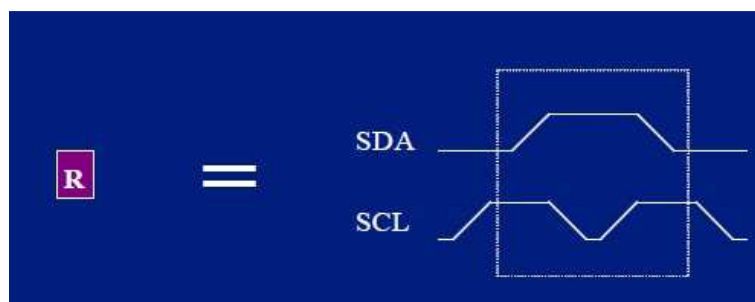


Figura 53. Reset condition.

- Acknowledge (ACK) y NotAcknowledge (NACK).

El ACK (figura 54) tiene lugar después del envío de un byte. El bit ACK permite al receptor de la señal conocer que la señal ha sido recibida correctamente y que se puede

enviar otro byte. El maestro es quien genera los pulsos de reloj, incluyendo el noveno pulso del ACK. La señal de ACK se define como sigue:

- El transmisor libera la línea de SDA durante el pulso de reloj del ACK, poniendo la línea de SDA a nivel bajo y manteniéndola así durante el pulso de nivel alto de la línea de SCL.
- Cuando la línea de SDA permanece alta durante el noveno pulso, esto se define como un NACK (figura 55). El maestro puede generar una condición de Stop para terminar la transferencia o repetir una condición de Start para empezar una nueva transferencia. Hay cinco casos que nos pueden llevar a un NACK:
 - No hay ningún receptor en el bus con la dirección que se está transmitiendo, por tanto no hay respuesta con un ACK.
 - El receptor es incapaz de recibir, porque está realizando una función en tiempo real y no está preparado para iniciar la comunicación con el maestro.
 - Durante la transferencia, el receptor recibe datos o comandos que no es capaz de entender.
 - Durante la transferencia, el receptor no puede recibir más datos.
 - Un maestro receptor debe de señalar el final de una transferencia a un esclavo transmisor.

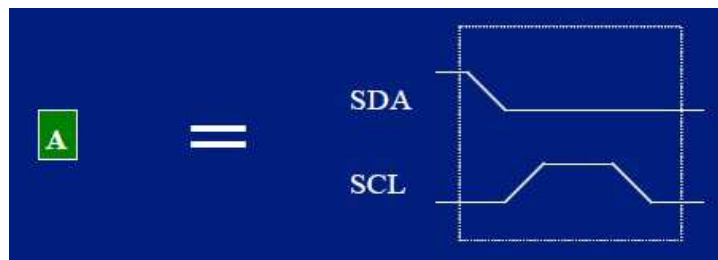


Figura 54. Señal de un Ack (Acknowledgement).

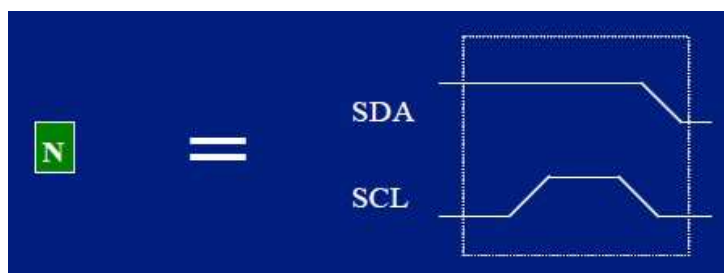


Figura 55. Señal de un Nack (no acknowledgement).

- Datos (D):

La transferencia de datos representa 8 bits de información. Los datos se envían por la línea de SDA mientras la línea SCL envía los pulsos de reloj. La sincronización de la línea de datos y la de reloj indica si cada bit es un 0 o un 1.

Los datos solo se consideran válidos cuando SCL se encuentra a nivel alto. Cuando SCL se encuentra a nivel bajo, se puede cambiar el valor de los datos, es decir, la línea de SDA puede pasar de nivel alto a nivel bajo, y viceversa, para cambiar la información que se debe de leer de nuevo cuando la línea de SCL vuelva a estar a nivel alto. Así es como funcionan.

Los bytes de datos son empleados para transmitir todo tipo de información. Cuando nos comunicamos con otro dispositivo I2C, el byte puede ser una dirección, un código de control, o simplemente un dato.

La representación de la transmisión de un dato de 8 bits puede verse en la figura 56.

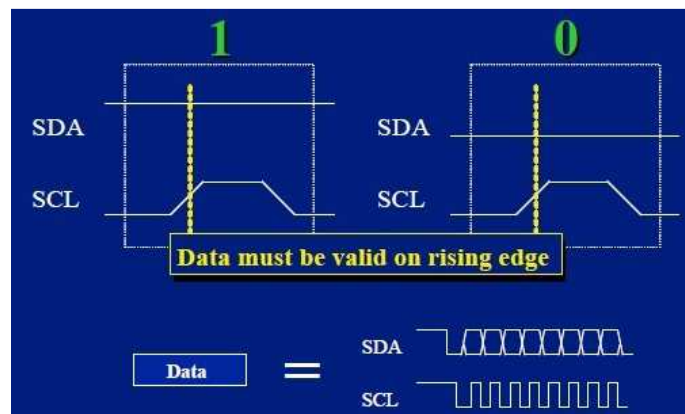


Figura 56. Representación de la transmisión de un dato.

10.2. El protocolo software para I2C.

A continuación, pasamos a explicar cómo se produce la comunicación entre el maestro y el esclavo por medio del bus I2C (figura 57).

a) Escribiendo en un dispositivo esclavo.

Lo primero que ocurrirá cuando queremos escribir a un esclavo, es que el maestro envíe una secuencia de inicio. Esto alertará a todos los dispositivos esclavos en el bus de que una transacción está empezando y que deberían escuchar en caso de que sea para ellos. A continuación el maestro enviará la dirección del dispositivo con el cual el

esclavo se quiera comunicar, y el dispositivo esclavo cuya dirección sea la emitida por el maestro continuará con la transacción, enviando un ACK al maestro, diciéndole que le ha llegado la información. Cualquier otro dispositivo ignorará la transacción y esperarán por otra. Una vez direccionado el dispositivo esclavo, el maestro deberá enviar la dirección interna o el número de registro en el que el maestro quiere leer o escribir. Este número obviamente depende del dispositivo esclavo y del número de registros de los que disponga. Una vez enviada la dirección I2C y la dirección del registro interno, esperará por el envío del ACK del esclavo; ahora el maestro puede ahora enviar el byte o bytes de datos.

El maestro puede continuar enviando bytes de datos al esclavo y estos serán normalmente ubicados en los registros siguientes ya que el esclavo incrementa automáticamente la dirección de los registros internos después de cada byte. Cuando el maestro ha terminado de escribir todos los datos en el esclavo, y después de haber esperado la respuesta del esclavo, envía una secuencia de parada que completa la transacción. Resumiendo, para escribir en un dispositivo esclavo hay que seguir estos pasos:

- Enviar la secuencia de inicio.
- Enviar la dirección I2C del esclavo con el bit lectura/escritura a 0 (el cero indica que se está escribiendo).
- Esperar a que el esclavo nos indique que ha recibido la información mediante la emisión de un ACK.
- Enviar el número de registro interno en el que se quiere escribir.
- Esperar nuevamente a que el esclavo nos indique que ha recibido la información mediante la emisión de un ACK.
- Enviar el byte de datos.
- En ocasiones, enviar otros bytes de datos y esperar al ACK del esclavo.
- Enviar la secuencia de parada.

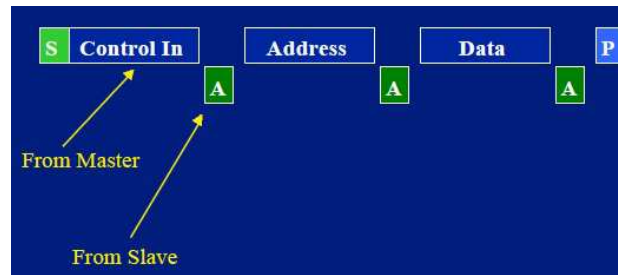


Figura 57. Leyendo de un dispositivo esclavo.

La lectura de un dispositivo esclavo es más compleja que la escritura y requiere de un mayor número de operaciones. Antes de leer datos del dispositivo esclavo, hay que indicarle al esclavo de cuál de sus direcciones internas queremos leer.

Por lo que para la lectura de un esclavo, se empieza escribiendo en él. Empezamos igual que si quisiésemos escribir en él. De forma breve, sería:

- Se envía la secuencia de inicio.
- A continuación la dirección I2C del esclavo con el bit lectura/escritura a 0.
- Se espera a que el esclavo envía un ACK.
- El maestro envía registro interno en el que se desea leer.
- Se espera al ACK del esclavo.

Hasta aquí, se ha procedido de la misma forma que si quisiésemos escribir en el esclavo, pero a partir de aquí es donde cambia:

- Se envía otra secuencia de inicio (que como hemos visto, puede ser producida por un Restart o por una Stop seguido de un Start).
- Se vuelve a enviar la dirección I2C del dispositivo, esta vez con el bit lectura escritura puesto a 1, el cual indica que queremos leer del dispositivo.
- Esperamos a que el esclavo nos envíe el ACK.
- A continuación podemos leer los bytes que queramos.
- El maestro envía un NACK al esclavo, conforme ha recibido los datos y termina la transición mediante un Stop.

La representación esquemática de cómo sería la escritura en un dispositivo esclavo puede verse en la figura 58.

NOTA: se envía un NACK en vez de un ACK para indicar que el maestro no quiere recibir más datos del esclavo.

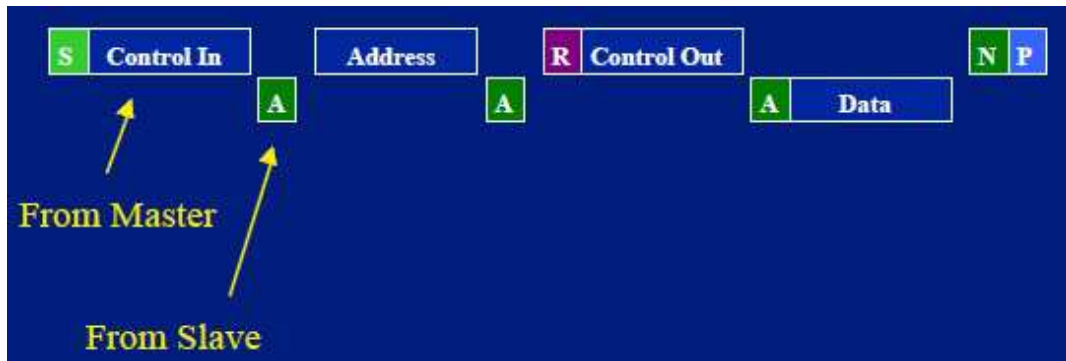


Figura 58. Representación esquemática de lectura de un dispositivo esclavo.

b) Algunos aspectos a destacar.

Esto es así para comunicaciones simples sobre I2C, pero existe una complicación a mayores. Cuando el maestro está leyendo del esclavo, es el esclavo el que pone los datos en la línea SDA, pero es el maestro el que controla el reloj. ¿Qué pasa si el esclavo no está listo para enviar los datos? Con dispositivos como EEPROMs esto no es un problema, pero cuando el esclavo es un microprocesador con otras tareas a las que atender, esto puede ser un problema. El microprocesador del dispositivo esclavo necesitará ejecutar una rutina de interrupción, guardando los registros en las que estaba trabajando, buscar la dirección que el maestro quiere leer, coger los datos y situarlos en el registro de transmisión. Esto puede llevar algunos microsegundos, lo que quiere decir que el maestro está enviando pulsos en la línea SCL a los que el esclavo no puede responder. El protocolo I2C proporciona una solución para esto: al esclavo se le permite mantener la línea SCL a nivel bajo. A esto se le llama “clock stretching”. Cuando el esclavo obtiene el comando de lectura proveniente del maestro, mantiene la señal de reloj a nivel bajo. El microprocesador entonces se encarga de conseguir los datos solicitados, situarlos en el registro de transmisión y actualizar la señal de reloj permitiendo que la resistencia pull-up la ponga a nivel alto. Desde el punto de vista del maestro esto permitirá emitir el primer pulso de la lectura mediante la activación de SCL a nivel alto y entonces chequear si realmente se ha puesto a nivel alto. Si todavía se encuentra a nivel bajo, indica que es el esclavo el que lo mantiene así, y que el maestro debe de esperar hasta que lo ponga a nivel alto para poder continuar.

Afortunadamente los puertos I2C de la mayor parte de los microprocesadores se encargan de manejar esto automáticamente.

A veces sin embargo, el maestro I2C es simplemente una colección de subrutinas y hay algunas implementaciones, que ignoran completamente el tema del “clock stretching”. Se trata de implementaciones que operan con componentes como EEPROMs, pero no con microprocesadores esclavos que usan “clock stretching”. El resultado es que se leen datos erróneos del esclavo.

10.3. Comunicación con el sensor VCNL4040.

En este punto hablaremos del código realizado, cómo lo hemos realizado y por qué ha sido realizado de esta manera, en base a los conocimientos previos adquiridos.

En primer lugar, lo que hemos hecho ha sido revisar la hoja de datos del sensor que nos proporciona el fabricante. En esta hoja de datos se nos indica el protocolo que hay que seguir para programar dicho sensor. En la hoja de datos aparece la figura 59 que nos indica el protocolo a seguir para la comunicación con el microcontrolador, tanto para configurarlo como para obtener información del mismo.



Figura 59. Protocolo de comunicación del sensor VCNL4040 con el microcontrolador.

Como en nuestro caso lo que queremos es la lectura de los datos que nos proporciona el sensor, nos interesa el segundo cuadro de instrucciones. Como se ha explicado en el apartado anterior, estos pasos deben estar dentro de una interrupción, que a su vez estará dentro de un *while* donde se van a ir dando las instrucciones seguidamente, y mientras que no se ha finalizado la anterior no se puede pasar a la siguiente.

Comenzando con el *start*, seguido de “*slave address*” (dirección del esclavo), que nos viene dada por la hoja de datos del sensor (en este caso 0x60), como podemos apreciar en la figura 59 aparecen unos números encima de los bloques que nos indican los bits que contiene esta instrucción. Un dato importante es que esta dirección de

memoria es tanto para escritura como para lectura, la diferencia es la terminación que se le añade según su finalidad, siendo de lectura un 1 y de escritura un 0. Esto hace que nuestro 0x60 se transforme en 0xC1 para lectura y 0xC0 para escritura. En el primer caso es escritura, donde vamos a escribir la configuración, que es la siguiente instrucción al Ack mandado por el esclavo al maestro.

Después de la respuesta del Ack, que también va a ser un paso en nuestro código, se encuentra el “command code” (código de comando), que es la configuración previa antes de la salida de datos del sensor, que viene predefinida en la dirección 0x0C.

Continuamos con la instrucción de lectura, donde le indicamos la dirección donde queremos que guarde el dato que obtiene de la señal, que como hemos dicho es 0xC1. Cuando ya hemos introducido todas las direcciones, solicitamos el valor que obtiene el sensor, que como vemos no entra en 8 bit, por lo que lo separa en dos partes que después tendrán que ser unidas. En cada solicitud de datos tenemos que enviar un Ack al dispositivo para que el sensor entienda que nos ha llegado la primera parte del dato y que estamos preparados para la segunda o que ya tenemos el dato entero.

El código de comunicación desarrollado ha sido añadido como un anexo a esta memoria, en el que se entenderá mucho mejor lo anteriormente comentado. Es recomendable para la comprensión del código tener en consideración estas aclaraciones.

Hasta aquí hemos hablado de la comunicación entre un sensor y el microcontrolador. Esto se complica cuando tenemos varios sensores conectados al mismo microcontrolador, ya que, como indica la comunicación I2C, cada vez que el maestro (microcontrolador) empieza una comunicación en un bus I2C, la información enviada es recibida por todos los dispositivos presentes en el bus. Cuando sólo queremos que la información la reciba un único dispositivo, una de las razones es que todos los sensores tienen las mismas direcciones de memoria, y esto es un problema, sobre todo cuando el maestro recibe la información proveniente de los esclavos, ya que colapsa.

Por eso se ha optado por la utilización de un multiplexor al que irán ocho sensores y de él saldrá una entrada/salida para el microcontrolador. Los multiplexores tienen una programación muy simple, donde simplemente se le especifican las entradas y salidas que nosotros queramos. A continuación se hablará con detalle de este componente.

11. Elección del multiplexor.

En este apartado nos centraremos en el sistema del multiplexor. Primero haremos una breve definición de su funcionamiento para comprender los siguientes apartados, seguido de la elección del multiplexor que usaremos, seguido de las características del multiplexor escogido. Por otra parte comentaremos todo lo realizado en el laboratorio para su puesta en marcha, que sería la fabricación del circuito integrado del multiplexor y su programación para un funcionamiento coordinado de los sensores, que en esencia es la finalidad del multiplexor.

11.1. Fundamentos teóricos. Multiplexores.

Los multiplexores son circuitos combinaciones de varias entradas y una única salida de datos (figura 60). Están dotados de entradas de control capaces de seleccionar una, y sólo una, de las entradas de datos para permitir su transmisión desde la entrada seleccionada hacia la salida.

En el campo de la electrónica, el multiplexor se utiliza como dispositivo que puede recibir varias entradas y transmitir las por un medio de transmisión compartido. Para ello lo que hace es dividir el medio de transmisión en múltiples canales, para que varios nodos puedan comunicarse al mismo tiempo.

Una señal que está multiplexada debe demultiplexarse en el otro extremo. Según la forma en que se realice esta división del medio de transmisión, existen varias clases de multiplexación:

- Multiplexación por división de frecuencia
- Multiplexación por división de tiempo
- Multiplexación por división de código
- Multiplexación por división de longitud de onda

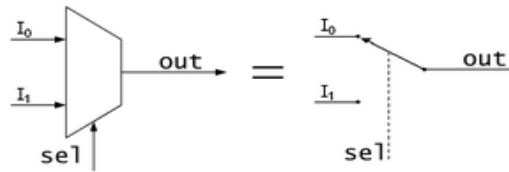


Figura 60. Esquema lógico de un multiplexor.

11.2. Selección del multiplexor

Para la selección del multiplexor, nos hemos centrado en dos distribuidores habituales con los que trabaja el Grupo Integrado de Ingeniería. Esto se ha realizado de éste modo debido a la amplia gama de productos de que disponen, de la rapidez de entrega de los productos, y de su precio competitivo. Estos dos distribuidores de productos electrónicos son Farnell ²⁵ y RS²⁶. Dentro del catálogo de estos dos distribuidores se han analizado los siguientes multiplexores:

→ Del distribuidor RS:

- **Multiplexor CD4051BEE4**, Simple 8:1, 12 V, 15 V, 18 V, 5 V, 9 V, PDIP (plastic dual in line package), 16 pines, 1 canales, marca Texas Instruments ²⁷ (0.42euros)
- **Multiplexor/Demultiplexor CD74HC4067E**, Simple 16:1, 3 V, 5 V, PDIP, 24 pines, 1 canales (1.09euros)
- **Multiplexor/Demultiplexor MC14051BDG**, Simple 8:1, 12 V, 15 V, 5 V, 9 V, SOIC²⁸, 16 pines, 1 canales (0.17euros)
- **Multiplexor/Demultiplexor HEF4051BT,013**, Simple 8:1, 5 V, 9 V, 12 V, SOIC (Small Outline Integrated Circuit), 16 pines, 1 canales(0.28euros)
- **Multiplexor/Demultiplexor PCA9547PW**, Simple 8:1, 2.3-5.5V, I2C Bus, 24 pines, 1 canales, marca NXP²⁸ (1.77euros)

→ Del distribuidor Farnell:

- **Multiplexor/Demultiplexor PCA9546APV** Simple 4:1, 2.3-5.5V, I2C, SMBus²⁹, 16 pines, 1 canales, marca NXP (1.93euros)

- **Multiplexor/Demultiplexor PCA9547PW**, Simple 8:1, 2.3-5.5V ,I2C, SMBus 24 pines, 1 canales, marca NXP (2.83euros)
- **Multiplexor/Demultiplexor PCA9540BD**, Simple 2:1, 2.3-5.5V ,I2C, SMBus, 8 pines, 1 canales, marca NXP (1.77euros)

Todos estos multiplexores presentan características similares. A la hora de seleccionar un multiplexor u otro nos hemos fijado en que: soporte el tipo de comunicación I2C, que tenga el mayor número de canales posible, que su precio no sea muy elevado y que funcione a la misma tensión del microcontrolador y del sensor infrarrojo. En consecuencia, el multiplexor escogido es el PCA9547PW con 8 canales de salida por 1 de entrada (el mayor de su familia), que pasamos a detallar a continuación.

11.3. Multiplexor/Demultiplexor PCA9547PW.

El PCA9547 (figura 61) es un multiplexor bidireccional octal controlado por el I2C-bus. Solo se puede seleccionar un canal a la vez SC_x / SD_x , determinado por el contenido del registro. Una de las funciones del dispositivo es que hasta con 0 líneas de conexión conectadas, permite la comunicación inmediata entre los dispositivos. Una activación del reset LOW, permite al PCA9547 recuperarse de una situación en la que uno de los buses I2C se ha quedado atascado en un estado bajo.

Pulsando el pin reset se restablece el estado del bus- I2C haciendo que todos los canales se anulen excepto la línea 0 para que el maestro pueda recuperar el control del bus. Las puertas de paso de los multiplexores se construyen de tal manera que el pasador de VDD se puede utilizar para limitar el alto voltaje máximo que será aprobada por el PCA9547. Esto permite el uso de diferentes tensiones de bus en cada línea de conexión, en concreto 1.8 V, 2.5 V, 3.3 V. Unas resistencias pull-up externas dan al bus-I2C el nivel de tensión deseado para cada línea de conexión. Todos los pines I / O tienen 5 V de tolerancia y una frecuencia de línea de reloj desde 0 hasta 400 KHz.

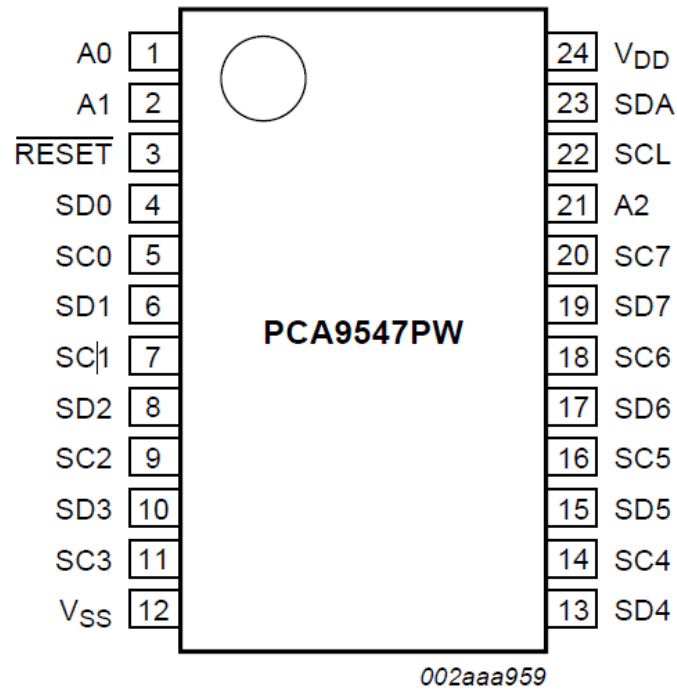


Figura 61. Multiplexor PCA9547PW

Symbol	Pin		Description
	SO24, TSSOP24	HVQFN24	
A0	1	22	address input 0
A1	2	23	address input 1
$\overline{\text{RESET}}$	3	24	active LOW reset input
SD0	4	1	serial data output 0
SC0	5	2	serial clock output 0
SD1	6	3	serial data output 1
SC1	7	4	serial clock output 1
SD2	8	5	serial data output 2
SC2	9	6	serial clock output 2
SD3	10	7	serial data output 3
SC3	11	8	serial clock output 3
V _{SS}	12	gnd	supply ground
SD4	13	10	serial data output 4
SC4	14	11	serial clock output 4
SD5	15	12	serial data output 5
SC5	16	13	serial clock output 5
SD6	17	14	serial data output 6
SC6	18	15	serial clock output 6
SD7	19	16	serial data output 7
SC7	20	17	serial clock output 7
A2	21	18	address input 2
SCL	22	19	serial clock line
SDA	23	20	serial data line
V _{DD}	24	21	supply voltage

Tabla 8. Funciones de cada pin del multiplexor PCA9547PW

Por último, mostramos el significado del multiplexor con la figura 62, donde se ven las puertas de cada canal del multiplexor seleccionado.

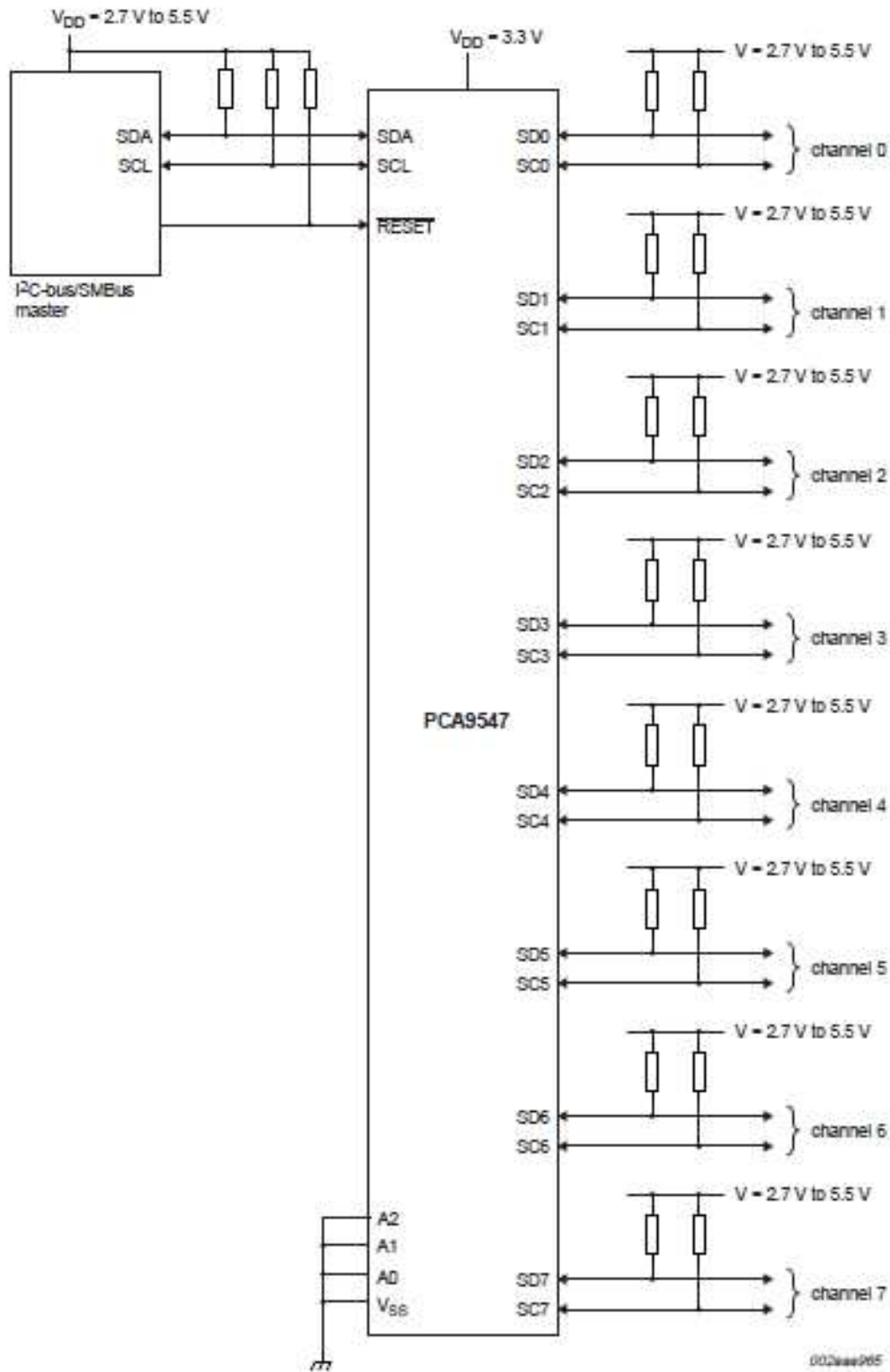


Figura 62. Funcionamiento del multiplexor PCA9547.

11.4. Diseño y fabricación del circuito integrado del multiplexor.

A continuación hablaremos del diseño del multiplexor, sin entrar en mucho detalle en su fabricación ya que es análoga a la que ha sido detallada en el apartado 7 para los sensores. Como diseño del pcb para el multiplexor obtenemos la figura 63:

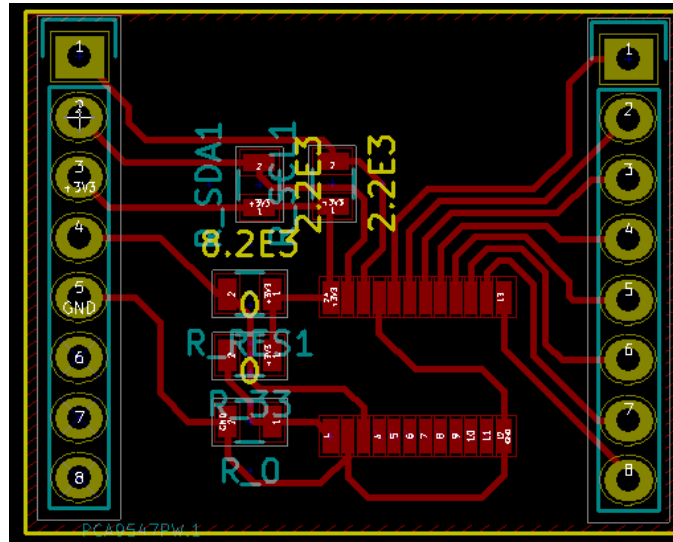


Figura 63. *Diseño del circuito integrado multiplexor PCA9547PW.*

Primero debemos hacer el circuito en el KiCad, después hacer relacionar los componentes con sus encapsulados correspondientes y hacer las pistas. Todo esto es exactamente igual que el apartado 7 para el diseño y fabricación de los sensores. Como diferencia con la realización de los sensores, ahora el proceso de soldadura es mucho más laborioso por ser un componente tan pequeño y con tantos pines. Como resultado final obtenemos:

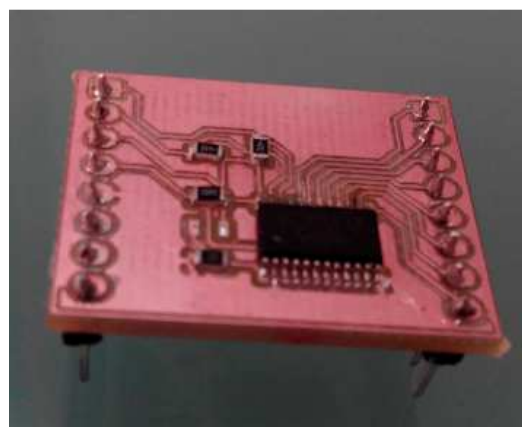


Figura 64. *Circuito integrado del Multiplexor PCA9547PW.*

Como nota, decir que para el diseño del multiplexor fueron necesarias 8 versiones, la última fue fabricada en un proceso que duró 3 horas por cada multiplexor realizado, por el correcto soldado del multiplexor.

11.5. Comunicación con el multiplexor.

La familia de los multiplexores Philips consiste en una comunicación (ver figura 65) por medio de dos canales, con la comunicación I2C, uno de los requisitos de su elección. El microcontrolador solo es capaz de mantener comunicación con un canal al mismo tiempo, por ello se utiliza un multiplexor, que hace la función de administrar las comunicaciones entre los ocho sensores y el microcontrolador, utilizando las órdenes del microcontrolador, que es el que solicita la comunicación con cada uno de los dispositivos.

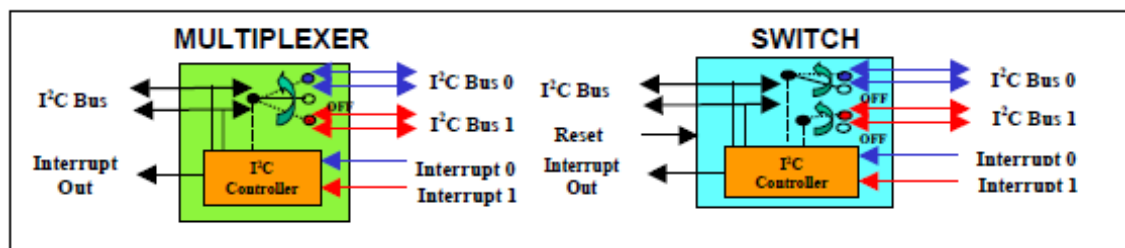


Figura 65. Comunicación del multiplexor.

A continuación hablaremos propiamente dichos de los pasos a realizar para una comunicación interna dentro del multiplexor.

Todos los dispositivos PCA954X soportan tanto el modo estándar (100 kHz) y el modo rápido (400 kHz) usado para el protocolo I2C. Una vez que los canales SDA y SCL del sensor adecuado, se han seleccionado y enviado la orden de paro, el dispositivo PCA954X utilizará hasta 400 kHz en el protocolo I2C. Una comunicación I2C estándar entre un controlador maestro y un dispositivo PCA954X contiene la siguiente secuencia:

- Condición de Start.
- Envío de una palabra de 8-bit con la siguiente información:

- PCA9547 asigna una dirección de 7-bits, que es 1 1 1 0 A2 A1 A0 (en nuestro caso las tres variables son 0, ya que están comunicadas a tierra).
- El octavo bit es la instrucción de lectura (“1”) o de escritura (“0”).
- Recepción del Ack por parte del esclavo, para la verificación de la comunicación, esto es muy habitual. También sucede en la comunicación con el microcontrolador.
- Si una instrucción de escritura es requerida la siguiente palabra de 8-bits es el registro de control.
- Si se solicita una instrucción de lectura, el master controlador se convierte en un master receptor y el esclavo se convierte en un transmisor. El estado de interrupción y el estado de selección del canal son posteriormente enviados al controlador master.
- Si la palabra previa de 8-bits es de escritura, el esclavo enviará un Ack al controlador master
- Si la palabra previa de 8-bits es de lectura, el esclavo no enviará un Ack al controlador master
- Condición de parada, cuando la condición será determinada por el multiplexor.

En la figura 66 se muestra las direcciones de control de registro para los diferentes canales que existen. La “x” significa que no importa para la dirección:

Control Register							Device Channel Selection			
6	5	4	3	2	1	0	PCA9540B/42A	PCA9543A	PCA9544A	PCA9547
x	x	x	0	0	0	0	None	None	None	None
x	x	x	0	0	0	1	None	Channel 0	None	None
x	x	x	0	0	1	0	None	Channel 1	None	None
x	x	x	0	0	1	1	None	Channel 0 & 1	None	None
x	x	x	0	1	0	0	Channel 0	None	Channel 0	None
x	x	x	0	1	0	1	Channel 1	Channel 0	Channel 1	None
x	x	x	0	1	1	0	None	Channel 1	Channel 2	None
x	x	x	0	1	1	1	None	Channel 0 & 1	Channel 3	None
x	x	x	1	0	0	0	None	None	None	Channel 0
x	x	x	1	0	0	1	None	Channel 0	None	Channel 1
x	x	x	1	0	1	0	None	Channel 1	None	Channel 2
x	x	x	1	0	1	1	None	Channel 0 & 1	None	Channel 3
x	x	x	1	1	0	0	Channel 0	None	Channel 0	Channel 4
x	x	x	1	1	0	1	Channel 1	Channel 0	Channel 1	Channel 5
x	x	x	1	1	1	0	None	Channel 1	Channel 2	Channel 6
x	x	x	1	1	1	1	None	Channel 0 & 1	Channel 3	Channel 7

Figura 66. Tabla de registros de control.

Todas estas direcciones y condiciones son introducidas dentro del código realizado anteriormente para el sensor, para que el multiplexor administre la comunicación además de pasar la comunicación necesaria para la lectura de datos del

sensor. Con la incorporación de estas direcciones para cada sensor y su protocolo de comunicación, haremos que los 8 sensores sean leídos consecutivamente por el microcontrolador. Después, el microcontrolador analizará esa información para que el comportamiento del robot sea el adecuado.

12. Resultados.

Es este último apartado nos centraremos en la integración de los elementos desarrollados con anterioridad, es decir, la colocación de los sensores en la carcasa con el multiplexor y el microcontrolador (programados y fabricados previamente). El objetivo final de esta integración es comprobar que todo funciona correctamente, es decir, que el robot esquiva los obstáculos y no se cae de ninguna superficie gracias a las distancias percibidas con los sensores infrarrojos.

Para ello es necesario que otras partes del Robobo que no son parte de este trabajo final de grado ya hayan sido realizadas, como son los motores, para que el robot pueda desplazarse, y la comunicación bluetooth, para la posibilidad de la salida de datos sin necesidad de conectar el ordenador a la plataforma, ya que se va a estar moviendo.

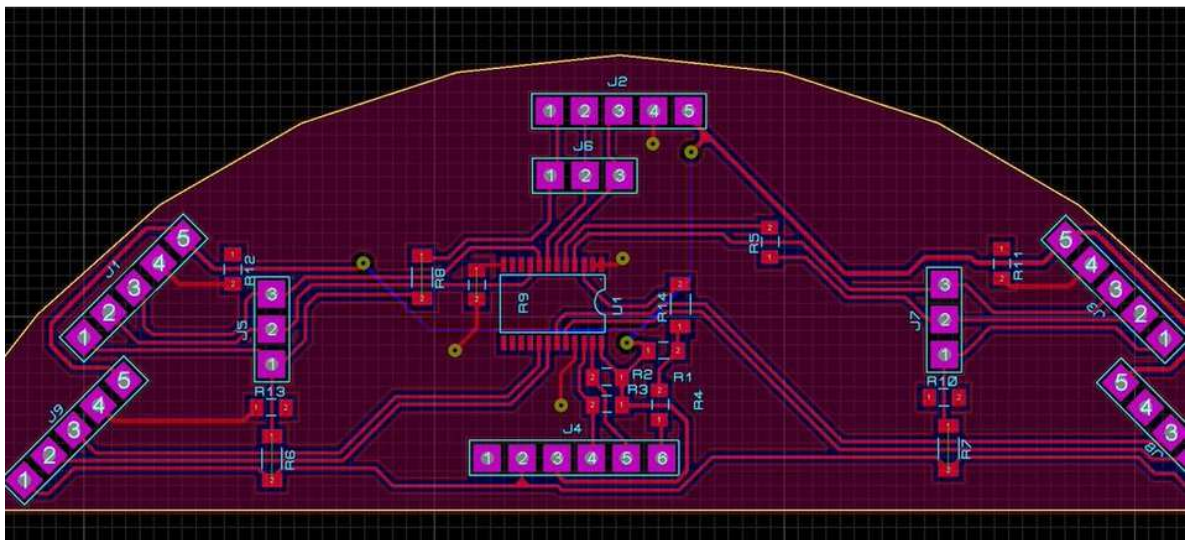


Figura 67. Pcb frontal donde van a ser conectados los sensores y el multiplexor

Un elemento fundamental que también ha tenido que ser diseñado es el pcb final que va a ir dentro de la carcasa, al que se le van a enganchar los circuitos integrados donde van a ir los sensores, donde va a ir alojado el multiplexor (figura 67). Este pcb tiene que tener la forma de la carcasa para que los sensores queden encajados en los huecos que se le harán a la misma.

Para las pruebas se colocará esta placa, con una protoboard con el microcontrolador, y con los otros dispositivos que no son tema en este trabajo, conectados como se muestra en la figura 68.

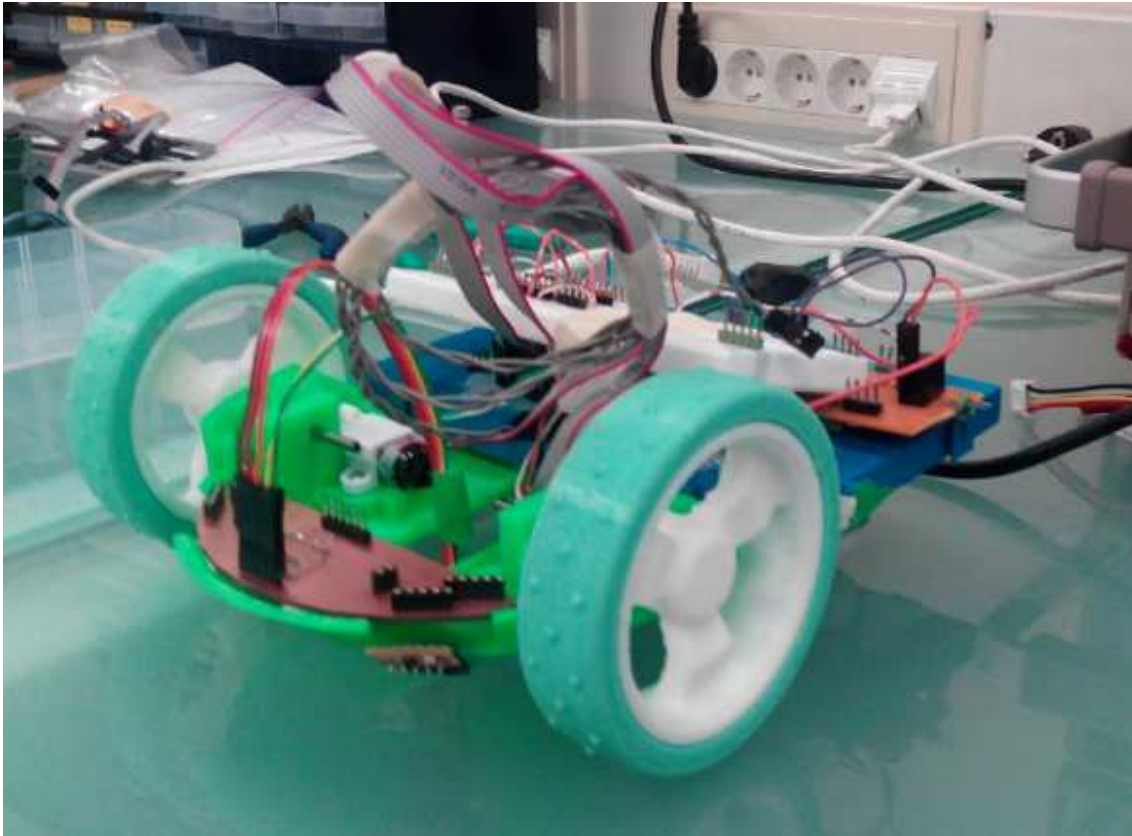


Figura 68. *Roboto de pruebas para la correcta sensorización..*

Como podemos apreciar en la imagen, la carcasa no está completa a la finalización de esta memoria. De todas formas, no es relevante para esta prueba de integración. El montaje de la figura 68 es puesto a andar en el aire poniendo la mano delante de los sensores para comprobar la velocidad de frenada de los motores, ya que si no funcionan correctamente, se chocaría contra la pared.

Se ha comprobado empíricamente que tanto el sensor como el multiplexor y el microcontrolador funcionan correctamente, no dan ninguna colisión en el bus de datos y todas las conexiones son correctas. Por lo que mostramos de forma práctica que tanto la elección de los componentes como la programación, diseño y fabricación han sido correctamente realizadas.

También se ha programado en el microcontrolador para que el led del microcontrolador se encienda cuando el sensor detecta un valor asignado. Esto ha sucedido correctamente también, y se enciende el led cuando nos colocamos a menos de

10cm del obstáculo, por lo que se verifica ahora en movimiento que el sensor tiene una distancia máxima de 10cm como era de esperar. Las pruebas realizadas muestran un correcto comportamiento de los sensores a unas velocidades de movimiento tanto bajas como altas (ver figura 69).



Figura 69. Pruebas de comprobación del correcto funcionamiento de los sensores.

13. Conclusiones.

El principal objetivo establecido en el apartado 2 para este Trabajo Fin de Grado ha sido cumplido satisfactoriamente y en estos momentos el ROBOBO ya posee un sistema de sensores infrarrojos totalmente integrado con la plataforma y totalmente funcional. Para lograrlo, se cumplieron los sub-objetivos marcados:

Teniendo en cuenta los requerimientos de diseño impuestos por los investigadores del Integrado de Ingeniería, se realizó una extensa revisión de los posibles sensores infrarrojos a aplicar en este proyecto y se seleccionó el modelo VCNL4040 de VISHAY. Dicho sensor se caracterizó para diferentes materiales, y se realizó el diseño y fabricación del pcb donde se aloja el sensor, hay que destacar que se realizaron más de una veintena de pcs, ya que hubo muchos que no funcionaron, y cómo mínimo eran necesarios ocho para completar todos los alojamientos de la carcasa.

A continuación se realizó el diseño 3D de la ubicación de los sensores en la carcasa, tanto frontales como traseros e inferiores y la modificación de la carcasa, para la salida de las ondas de los infrarrojos, estudiando la forma de este alojamiento para una correcta propagación de las ondas de los sensores.

También se realizó la selección y programación del microcontrolador que gestiona el funcionamiento de todos los sensores y del multiplexor que secuencía la señal de los sensores al microcontrolador. Además del diseño del pcb donde alojamos el multiplexor y su fabricación. Para la programación tanto del microcontrolador como del multiplexor ha sido necesario hacer un montaje en una protoboard para todos los componentes nombrados, como se puede ver en la figura 68.

Finalmente se realizaron diversas pruebas de funcionamiento de los sensores montados en una versión simplificada de la carcasa pero en movimiento, de modo que se pudo comprobar la correcta integración de todas las partes.

14. Bibliografía.

Definición de robot

<http://guiboringenieria.com/Robotica.html>

Hojas de datos

Labs, Silicon. SI1141A11GMR. [En línea] [Citado el: 11 de 12 de 2015.]

<http://www.datasheets.com/search/partdetail/SI1141A11GMR/Silicon+Labs>.

Microchip. 2013. PIC32MX5XX/6XX/7XX. [En línea] 22 de 4 de 2013. [Citado el: 2 de 12 de 2015.]

<http://www.microchip.com/wwwproducts/Devices.aspx?product=PIC32MX534F064H>.

NXP. 2014. Application note AN262. [En línea] 1 de 04 de 2014. [Citado el: 20 de 1 de 2016.] http://www.nxp.com/documents/aplication_note/AN262.pdf.

NXP. 2014. PCA9547. [En línea] 1 de 04 de 2014. [Citado el: 20 de 1 de 2016.]

www.nxp.com/documents/data_sheet/PCA9547.pdf.

SHARP. GP2D150AJ00F. [En línea] [Citado el: 11 de 12 de 15.]

www.sharpmicro.com/Page.aspx/europe/en/part/GP2D150AJ00F.

vishay. 2014. VCNL4010. *Fully Integrated Proximity and Ambient Light Sensor with.* [En línea] 13 de Agosto de 2014. [Citado el: 1 de 11 de 2015.]

www.vishay.com/docs/83462/vcnl4010.pdf.

VISHAY. 2015. VCNL4020. *Fully Integrated Proximity and Ambient Light Sensor with.* [En línea] 13 de Agosto de 2015. [Citado el: 1 de Noviembre de 2015.]

www.vishay.com/docs/83476/vcnl4020.pdf.

Vishay. 2015. VCNL4040. [En línea] 2 de marzo de 2015. [Citado el: 1 de 11 de 2015.]

www.vishay.com/docs/84274/vcnl4040.pdf.

Precios de los sensores y multiplexores

<http://es.farnell.com/webapp/wcs/stores/servlet/Search?catalogId=15001&langId=-5&storeId=10176&categoryName=Todas%20las%20categor%C3%ADas&selectedCategoryId=&gs=true&st=multiplexores>

<http://es.rs-online.com/web/c/semiconductores/circuitos-integrados-de-interfaces/circuitos-integrados-de-conmutador-multiplexor/?applied-dimensions=4294965617#>

<http://es.rs-online.com/web/p/circuitos-integrados-de-conmutador-multiplexor/0510655/>

<http://uk.farnell.com/nxp/pca9546apw-112/4-channel-i2c-bus-multiplexer/dp/1831208>

<http://es.farnell.com/nxp/pca9547pw/ic-mux-8ch-i2c-16tssop/dp/2212121>

<http://es.farnell.com/nxp/pca9540bd/ic-mux-2ch-i2c-bi-dir-8soic/dp/2212115>

REFERENCIAS:

- 1) Kaplan, Frederic. "Talking AIBO: First experimentation of verbal interactions with an autonomous four-legged robot." *Learning to behave: interacting agents CELE-TWENTE Workshop on Language Technology*. 2000.
 - . Robot AIBO. <http://www.sonyaibo.net/aibostory.htm>
 - . Robot AIBO. <http://www.sonydigital-link.com/AIBO/>
- 2) Chen, Mao, et al. "RoboCup Soccer Server for Soccer Server Tersion cefc and later." (2001).
 - . Robot NAO. <http://aliverobots.com/>
 - . Robot NAO. <http://listas.20minutos.es/lista/los-robots-reales-del-salon-de-la-fama-del-robot-345597/>
- 3) Tribelhorn, Ben, and Zachary Dodds. "Evaluating the Roomba: A low-cost, ubiquitous platform for robotics research and education." *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, 2007.
- 4) Mondada, Francesco, et al. "The e-puck, a robot designed for education in engineering." *Proceedings of the 9th conference on autonomous robot systems and competitions*. Vol. 1. No. 1. 2009.
- 5) Robot E-puck. <http://www.gctronic.com/doc/index.php/E-Puck>
- 6) Robot E-puck. <http://www.e-puck.org/>
- 7) Thymio. <https://aseba.wikidot.com/en:thymio>
- 8) LEGO Midstorm.
 - <http://el.media.mit.edu/logo-foundation/pubs/logoupdate/v7n1/v7n1-pbrick.html>
 - <http://suite101.net/article/de-logo-a-lego-con-un-enfoque-construccionista-a45690#axzz2HrJrIkqt>
- 9) Kickstare. <https://www.kickstarter.com/about?ref=nav>
- 10) Rommo.
 - <http://www.expansion.com/2013/12/10/empresas/digitech/1386697047.html>
 - <http://www.tecnoadicto.net/romo-robot-iphone.html>
- 11) iOS. <http://www.apple.com/es/ios/what-is/>; <https://es.wikipedia.org/wiki/IOS>

- 12) Android.
https://www.android.com/intl/es_es/; <https://es.wikipedia.org/wiki/Android>
- 13) Smartbot.
https://www.google.es/search?q=smartbot&safe=active&biw=1252&bih=604&tbm=isch&source=lnms&sa=X&ved=0ahUKEwj4m828mprKAhWCzxoKHeCIDbMQ_AUIBygC#imgsrc=Auwd9JLw09lfMM%3A
- 14) Arduino. <https://www.arduino.cc/>
- 15) ROS. Robotic operative system <http://www.ros.org/>
- 16) Oddberx. <http://www.oddwerx.com/>
- 17) Wheelphone. <http://www.wheelphone.com/img/wheelphone-fleet-small.jpg>
- 18) GCtronic. http://www.gctronic.com/doc/index.php/Main_Page
- 19) UDC (Universidad de la Coruña). www.udc.es
- 20) DREAM (Deferred Restructuring of Experience in Autonomous Machines).
<http://www.gii.udc.es/proyectos/detalle/342>
- 21) CNSRS (Centre national de la recherche scientifique).
<http://www.cnrs.fr/fr/organisme/presentation.htm>
- 22) ENSTA. École Nationale Supérieure de Techniques Avancées .
<https://www.ensta-paristech.fr/>
- 23) UPMC. sorbonne-universites.
http://www.upmc.fr/en/university/sorbonne_universities.html
- 24) SHARP. Electronic components. http://www.sharp-world.com/products/device/lineup/selection/opto/dust/index.html?_ga=1.92889694.214383690.1454423671
- 25) Silicon Labs.
<http://www.silabs.com/products/sensors/infraredsensors/Pages/default.aspx>
- 26) Vishay. <http://www.vishay.com/moreinfo/vcnlfamily/>
- 27) Farnell. <http://uk.farnell.com/proximity-sensors>
- 28) RS. <http://es.rs-online.com/web/>
- 29) Texas Instrument. <http://www.ti.com/>
- 30) NXP. http://www.nxp.com/about/about-nxp:COMPANY_INFO_HOME
<http://www.nxp.com/>
- 31) SMBus. System Management Bus.
https://en.wikipedia.org/wiki/System_Management_Bus

15. ANEXO

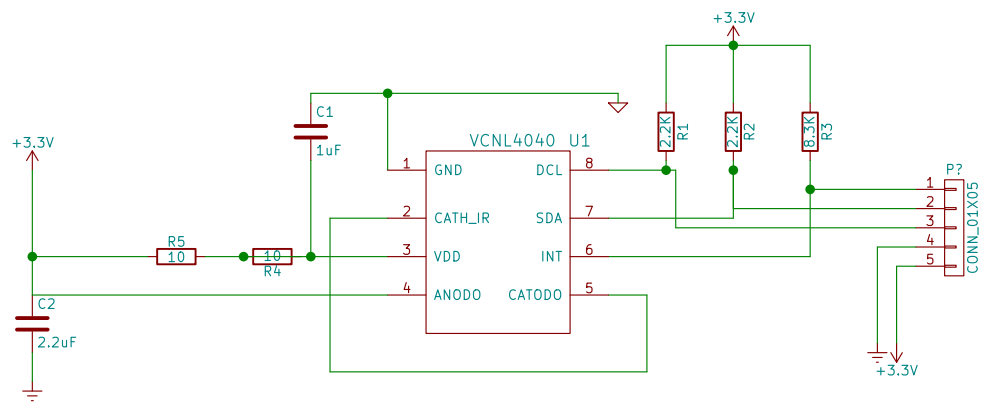
En este apéndice se muestra parte del código implementado en este TFG, ya que es demasiado extenso. En estas líneas se muestran los casos del protocolo de comunicación, que está dentro de las interrupciones generadas por el microcontrolador.

```
void __ISR(_I2C_4_VECTOR, IPL4) i2c4_interrupt_routine(void) {
    if (INTGetFlag(INT_I2C4M)) {
        if (bus_state == transmission_in_progress) {
            if (R_W == read_message) {
                switch (read_state)
                {
                    case start_sent_to_slaveWrite:
                        read_state++;
                        I2CSendByte(I2C_BUS, slaveAddress.byte);
                        break;
                    case ack_to_commandCode:
                        if (I2CByteWasAcknowledged(I2C_BUS)) {
                            read_state++;
                            I2CSendByte(I2C_BUS, 0x0C);
                        } else error = first_ack_not_received;
                        break;
                    case ack_to_start:
                        if (I2CByteWasAcknowledged(I2C_BUS)) {
                            read_state++;
                            while (I2C4CONbits.SEN || I2C4CONbits.PEN || I2C4CONbits.RSEN ||
I2C4CONbits.RCEN || I2C4CONbits.ACKEN || I2C4STATbits.TRSTAT);
                            I2C_FORMAT_7_BIT_ADDRESS(slaveAddress, VCNL_ADRR,
I2C_READ);
                            result = StartTransfer(true);
                        } else error = second_ack_not_received;
                        break;
                    case start_sent_to_slaveRead:
                        if (I2CByteWasAcknowledged(I2C_BUS)) {
                            read_state++;
                            I2CSendByte(I2C_BUS, slaveAddress.byte);
                        } else error = third_ack_not_received;
                        break;
                    case enable_data_reception:
                        if (I2CByteWasAcknowledged(I2C_BUS)) {
                            read_state++;
                            I2CReceiverEnable(I2C_BUS, true);
                        } else error = fourth_ack_not_received;
                        break;
                    case ack_to_readDBL:
                        if (I2CReceivedDataIsAvailable(I2C_BUS)) {
                            read_state++;

```

```
        I2CAcknowledgeByte(I2C_BUS, true);
        irLB = I2CGetByte(I2C_BUS);
    } else error = first_data_not_available;
    break;
case ack_to_readDBH:
    if (I2CReceivedDataIsAvailable(I2C_BUS)) {
        read_state++;
        I2CAcknowledgeByte(I2C_BUS, false);
        irHB = I2CGetByte(I2C_BUS);
    } else error = second_data_not_available;
    break;
case ack_to_stop:
    read_state++;
    I2CStop(I2C_BUS);
    break;
case end_read:
    readCompleted = true;
    bus_state = free_bus;
    read_state = ready_to_start;
    break;
    }
}
}
INTClearFlag(INT_I2C4M);
} else {
    if (INTGetFlag(INT_I2C4B)) {
        init_I2C_Module(I2C_BUS);
        bus_state = free_bus;
        read_state = ready_to_start;
        fail = true;
        INTClearFlag(INT_I2C4B);
    }
}
}
```

PLANOS



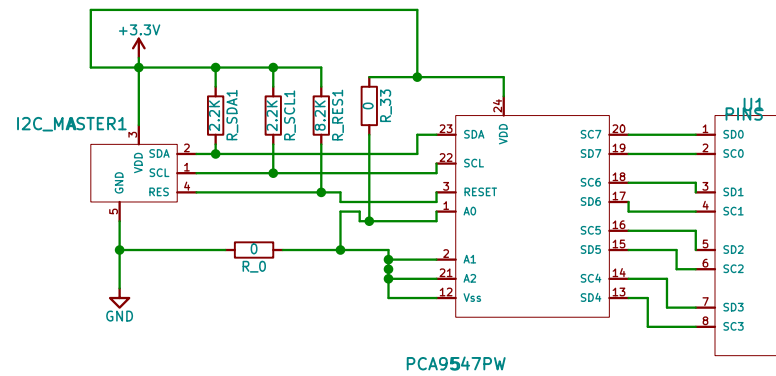
Autor: Alba Lema Martínez
Trabajo final de grado

Sheet: /
File: vcnL_sil.sch

Title: Placa VCNL4040

Size: A4 Date: 10/02/2016
KiCad E.D.A. eeschema 4.0.0-rc1-stable

Rev:
Id: 1/1



Alba Lema Martínez

Trabajo Final de Grado

Sheet: /
File: multiplexer.sch

Title: Multiplexor

Size: A4 Date: 26/01/2016
KiCad E.D.A. kicad 4.0.0-rc1-stable

Rev:
Id: 1/1

PRESUPUESTO

Escuela Politécnica Superior Ferrol**Grado
en I.M.**

Promotor: GRUPO INTEGRADO DE INGENIERÍA

EPS
Ferrol**COSTE PARA LA SENSORIZACION DE LA PLATAFORMA ROBÓTICA.**

C.I CAPÍTULO I. MATERIALES				
N/P	CONCEPTO	Uds.	P. Unit.	Importe
1.1	SENSORES	8,0	1,71 €	13,68 €
1.2	MULTIPLEXOR	1,0	2,83 €	2,83 €
1.3	MICROCONTROLADOR			- €
1.4	RESISTENCIAS DE 10 Ω	16,0	0,02 €	0,29 €
1.5	RESISTENCIA DE 2,2 K Ω	18,0	0,02 €	0,38 €
1.6	RESISTENCIA DE 0 Ω	23,0	0,03 €	0,78 €
1.7	RESISTENCIAS DE 8,3 K Ω	8,0	0,02 €	0,13 €
1.8	CONDENSADORES DE 1 uF	8,0	0,03 €	0,25 €
1.9	CONDENSADORES DE 2,2 uF	8,0	0,03 €	0,22 €
1.10	placa 1mxm	1,0	2,00 €	2,00 €
1.11	planos	2,0	0,52 €	1,04 €
1.12	acodados	1,0	0,88 €	0,88 €
	TOTAL CAPÍTULO I			22,47 €

C.II CAPÍTULO II. MANO DE OBRA				
N/P	CONCEPTO	Horas	€/hora	Importe
2.1	FABRICACIÓN	100,0	20,00	2.000,00 €
2.2	DISEÑO E INGENIERÍA	480,0	40,00	19.200,00 €
	TOTAL CAPITULO III.			21.200,00 €

IMPORTE DE EJECUCIÓN MATERIAL	21.222,47 €
13 % de Gastos Generales	2.758,92 €
6 % de Beneficio Industrial	1.273,35 €
IMPORTE DE EJECUCIÓN	25.254,74 €
21% IVA	5.303,50 €
IMPORTE DE CONTRATA	30.558,23 €

Ferrol, 10 de febrero de 2016

Fdo.: