Grado en Ingeniería Electrónica Industrial y Automática.

Trabajo de Fin de Grado

Sistema de ventilación forzada para neveras instaladas en caravanas y autocaravanas

Autor: Iván Jesús Castañeda Bethencourt
Tutor: Alejandro José Ayala Alfonso
Cotutora: Beatriz Rodríguez Mendoza

Universidad de La Laguna

ÍNDICE

Abstract

In the global pandemic that we are going through, the sale of motorhomes has exploded. There are many people who decide to buy a motorhome or a van in order to camper it and acquire greater freedom.

The objective of this project has been the design and implementation of a temperature control system based on the use of ATMega328P-Pu microcontrollers.

This project is based on the design of a prototype capable of measuring certain temperatures. With its subsequent analysis and with a programming to activate or deactivate the fans according to your requirement. A current problem such as the temperature of the compression refrigerator circuitry can be kept under control at all times. Due to the high temperatures the circuit overheats a lot and the components last very little.

Resumen

En la pandemia mundial que estamos atravesando, la venta de autocaravanas se ha disparado. Son muchas las personas que deciden adquirir una autocaravana o un furgón con el fin de camperizarlo y adquirir una mayor libertad.

El presente proyecto ha tenido como objetivo el diseño e implementación de un sistema de control de temperatura basado en la utilización de microcontroladores ATMega328P-Pu.

Dicho proyecto se basa en el diseño de un prototipo capaz de medir determinadas temperaturas. Con su posterior analizado y con una programación de activar o desactivar los ventiladores según sea su requerimiento. Se podrá mantener controlado en todo momento un problema actual como es la temperatura de la circuitería de las neveras de compresión, debido a que altas temperaturas se recalienta mucho el circuito y duran muy poco los componentes.

Capítulo I: Introducción General

I.1 Objetivos.

El presente proyecto está dirigido al mundo de las autocaravanas que, en el caso de España, gozan de una historia muy reciente. Los primeros modelos que circularon por nuestras carreteras datan de los años 70 del pasado siglo, por lo que no es necesario retroceder demasiado en el tiempo, hablamos tan solo de 50 años atrás.

El origen de la Autocaravana establece como pionero a Gordon Stables (Inglaterra, 1885) quien llevó a cabo la construcción de una caravana de seis caballos (Figura I.1). En su interior constaba de una cocina, una mesa y un sofá cama, resultando un vehículo muy bien equipado para la época. Desde 1890 las autocaravanas se construyeron sobre vehículos a motor, las cuales estaban hechas a mano y para adquirirlas era necesario disponer de un alto nivel adquisitivo.

En 1921 la empresa estadounidense Campingcar construyó lo que ya se consideraba el primer prototipo para la denominación de Autocaravana. Éstas sólo disponían de cuatro camas y una mesa, pues ya estaba introducida la idea de poder pernoctar en su interior.

En 1951, la empresa Wetsfalia (muy conocida en ámbitos campistas) construyó su primer modelo adaptándose a los requerimientos de la época y actualizada en muchas ocasiones. Westfalia hizo uso de una pequeña camioneta producida por Volkswagen, donde el interior tenía cuatro camas y desde 1961 los primeros modelos producidos en serie comenzaron a abandonar la fábrica alemana.

Figura I.1 Primera caravana 1885.

Con el paso de los años, comenzaron a surgir distintas empresas de producción de Autocaravanas hasta llegar a nuestros días donde podemos encontrar una gran variedad de modelos y marcas.

Actualmente las Autocaravanas cuentan con una equipación muy completa, con una gran autonomía permitiendo realizar viajes de larga duración con gran comodidad.

Cada vez son más las personas que deciden adquirir una autocaravana o directamente comprar un furgón y camperizarlo, potenciado por el hecho de la actual pandemia, dando lugar a un incremento importante en la venta de estos vehículos.

Entre los elementos integrantes de cualquier autocaravana, caravana o vehículo vivienda, cobra especial importancia la nevera, sobre todo si tenemos en cuenta que en la mayoría de los casos estos vehículos se suelen utilizar con mayor frecuencia en época estival. En los últimos años, el empleo de neveras con compresor, menos sensibles a la temperatura ambiente y de menor consumo, se ha incrementado de manera ostensible. Éstas, incorporan gran cantidad de electrónica necesaria para su funcionamiento, electrónica que se ve afectada por la elevada temperatura que se puede alcanzar dentro de una autocaravana (caravana o vehículo vivienda) durante los meses de verano.

Por otro lado, muchas de las empresas que fabrican estos vehículos suelen estar ubicadas en países nórdicos, con temperaturas medias muy inferiores a las que se alcanzan otros como España, Italia o Grecia, y sin tener en cuenta que éstas podrían finalizar en países como los anteriormente indicados.

A lo anterior, hay que añadir el hecho de que este tipo de neveras se suelen instalar empotradas dentro de muebles con nula conexión hacía el exterior, dando como resultado frecuentes averías en la electrónica antes mencionada. El objetivo del presente trabajo va dirigido a intentar mantener, en todo momento, la máxima refrigeración de los componentes electrónicos que forman parte de la nevera, lo que ha llevado al uso de dos microcontroladores ATMega328, tres sensores de temperatura y un enlace Bluetooth para conectar la interfaz de usuario con la etapa de electrónica de potencia encargada de activar la ventilación forzada necesaria para refrigerar la circuitería anteriormente reseñada.

I.2 Descripción general del sistema.

Como se ha comentado en el apartado anterior, el presente trabajo tiene como objetivo el diseño de un sistema electrónico capaz de disminuir la temperatura a la que trabaja la circuitería electrónica encargada de gestionar el funcionamiento de la nevera instalada en una autocaravana (caravana o vehículo vivienda).

Para tal fin, el sistema implementado consta de dos bloques denominados, respectivamente, Maestro y Esclavo (Figura I.2) y controlados, en ambos casos, por sendos microcontroladores ATMega328P. El prototipo final se ha diseñado para que cumpla con los siguientes requisitos: Bajos coste y consumo, tamaño reducido y fácil programación.

El Maestro (Figura I.2) actúa, por un lado, como interfaz de comunicación con el usuario, para lo que se emplea un display LCD de 4x20 caracteres, conjuntamente con dos pulsadores que posibilitan desplazarse por los diferentes menús y realizar la selección de distintas opciones (subir, bajar y ENTER).

Por otro lado, el Maestro dispone de un sensor de temperatura y humedad (DHT22) con el objetivo de medir dichas variables en el interior del vehículo, junto con un reloj calendario que permite visualizar la hora y fecha en el display.

En cuanto al Esclavo (Figura I.2), éste hace uso de dos sensores de temperatura (DS18b20) encargados, respectivamente, de medir dicha variable en el exterior de la autocaravana y en las proximidades del recinto que contiene los circuitos electrónicos encargados de la gestión de la nevera. Lo anterior, se complementa con dos relés de estado sólido (SIP10A05) que permiten activar sendos ventiladores para generar una ventilación forzada.

La conexión entre el Maestro y el Esclavo se ha realizado vía radio mediante un enlace Bluetooth (módulos HC-05 y HC-06, Figura I.2), lo que simplifica enormemente la instalación del sistema, pues evita el engorro que supone el unir por cable ambos módulos.

El Maestro es el encargado de gestionar todo el sistema, pues dispone del valor de la temperatura del habitáculo (proporcionada por su sensor DHT22), junto a las remitidas de manera continua (vía Bluetooth) por los dos sensores (DS18b20) conectados al Esclavo. En función de los valores recibidos por el primero, éste puede ordenar al Esclavo que active uno de los ventiladores o ambos. Uno de ellos, permite intercambiar aire desde el interior de la autocaravana con el exterior, mientras el otro lo hace desde la parte trasera de la nevera (donde se encuentra la electrónica de control de la misma) con el habitáculo.

Por último, el menú de control del Maestro dispone de una opción Manual que permite la puesta en funcionamiento de los ventiladores cuando así se desee.

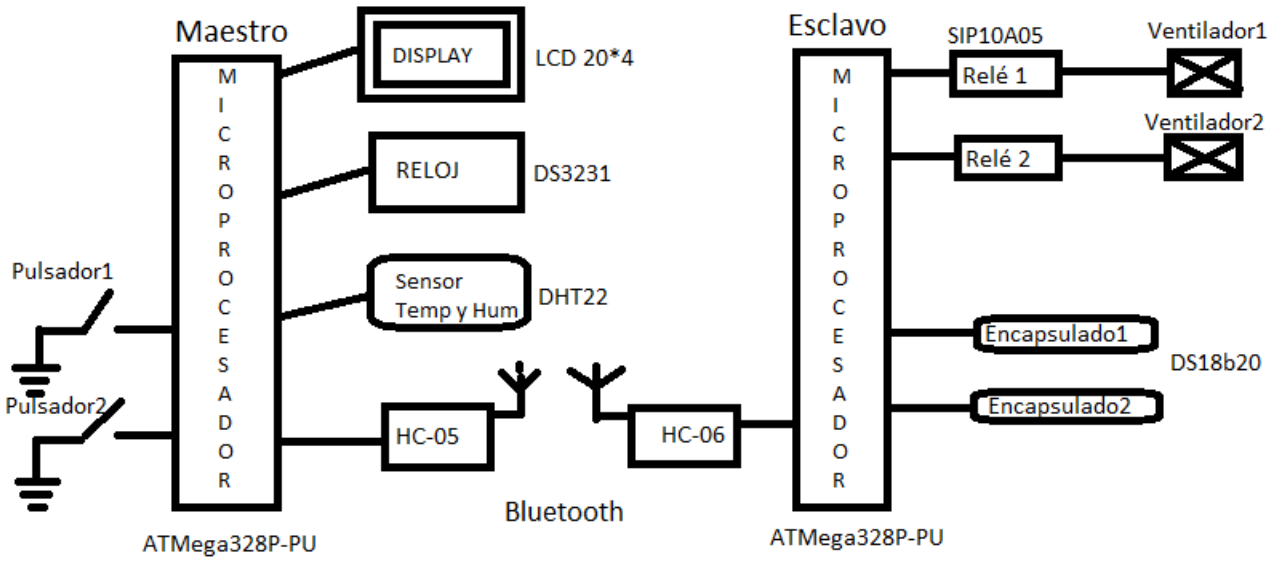


Figura I.2 Diagrama de bloques del sistema.

I.3 Estructura general del trabajo

En la presente memoria, la descripción de los diferentes módulos que conforman el sistema implementado, así como su funcionamiento se realiza mediante el desarrollo de siete capítulos.

En el primer capítulo se aborda la introducción general del prototipo, donde se introducen los objetivos de este proyecto y una descripción general del sistema.

En el segundo capítulo se describe cada componente empleado en este proyecto explicando sus características, comunicación entre bloques y conexionado, justificando su utilización.

En el tercer capítulo se indican las herramientas de diseño y desarrollo empleadas. Dichos programas fueron de vital importancia a la hora de crear el prototipo.

En el cuarto capítulo se entra en materia explicando paso a paso cómo funciona el sistema implementado y mostrando imágenes reales del mismo.

En el quinto capítulo se centra en la parte software y, haciendo uso de diagramas de flujo se explican paso a paso los programas diseñados para los microcontroladores ATMega328P-Pu.

El sexto capítulo se centra en los resultados experimentales, aprovechando en explicar el tiempo que estuvo de prueba dicho prototipo y presentando fotos demostrativas.

En el séptimo capítulo se establece el presupuesto generado con los precios de cada componente y mano de obra.

En el resto de la memoria no se consideran capítulos, pero no por ello son menos importantes. Así podemos encontrar tanto las conclusiones que se han propuesto para el proyecto, como la bibliografía fundamental donde se han asesorado para la creación de éste y un glosario con abreviaturas técnicas, entre otros.

Capítulo II: Microcontrolador y Unidades de Entrada y Salida

II.1 Microcontrolador ATMega328P-Pu

II.1.1 Definición y características.

En un primer momento se habla del ATMega328P, un microcontrolador fabricado por Atmel (Microchip), perteneciente a la familia AVR y con una arquitectura RISC de 8 bits. Cuenta con muchas instrucciones que se ejecutan en un ciclo de reloj, por lo que puede alcanzar un desempeño muy próximo a 1 MIPS por cada 1 MHz en la frecuencia de reloj.

Es la parte que procesa toda la información, donde se graba el código, en el software de Arduino se conoce como “Sketch”. En pocas palabras el Atmega328P es el cerebro de la placa, el cual tiene como función grabar, almacenar y ejecutar el código programado. El resto de los componentes se encargan de garantizar su correcto funcionamiento.

Este microcontrolador también viene montado en la tarjeta Arduino Uno R3, por lo que a la hora de programación utilizamos estas tarjetas. El inconveniente fue que se necesitaban dos, por lo que se adquirió un integrado ATMega328P-Pu, con el inconveniente de que de fábrica venían en blanco, por lo que hizo falta otro Arduino para cargar el bootloader.

Desde un primer instante, estuvo clara la elección, se iba a implementar con dos microprocesadores 328P-Pu. Cada placa se generó para cumplir su función en el cual una se caracterizaba de maestro “Master” y la otra de esclavo “Slave”. La elección de dichos microprocesadores fue con tanta claridad debido a la poca complejidad del programa en sí, no hizo falta requerir de un microprocesador mayor, ya que la memoria SRAM es de 2KB en la que el programa solo requiere el 50% y hace que el programa no se trabase ni se quede estático.

Las características de este microcontrolador se reflejan en la siguiente tabla:

PARÁMETROS	VALORES
Flash	32 Kbyte
SRAM	2 Kbyte
Cantidad de pines	28
Frecuencia Máxima de operación	20 MHz
CPU	850542-bit AVR
Pines máximos de E/S	23
Interrupciones internas	24
Canales ADC	8
Resolución ADC	10
EEPROM	1 Kbyte
Canales PWM	6
Voltaje de operación	1.8 a 5.5 VDC
Timers	3

Tabla II.1. Características del ATMega328P-Pu

II.1.2 Asociación de pines.

La definición de los pines se muestra en la Figura II.1, donde se aprecian cada uno de los pines del microcontrolador y sus correspondientes pines a los que tienen acceso en Arduino Uno. [1] Este apartado fue de vital importancia debido a que el microcontrolador se extrajo de la tarjeta de Arduino Uno, quedando al desnudo, por lo que gracias a esta asociación se supo alimentar y gestionar según el uso que iba a generar.

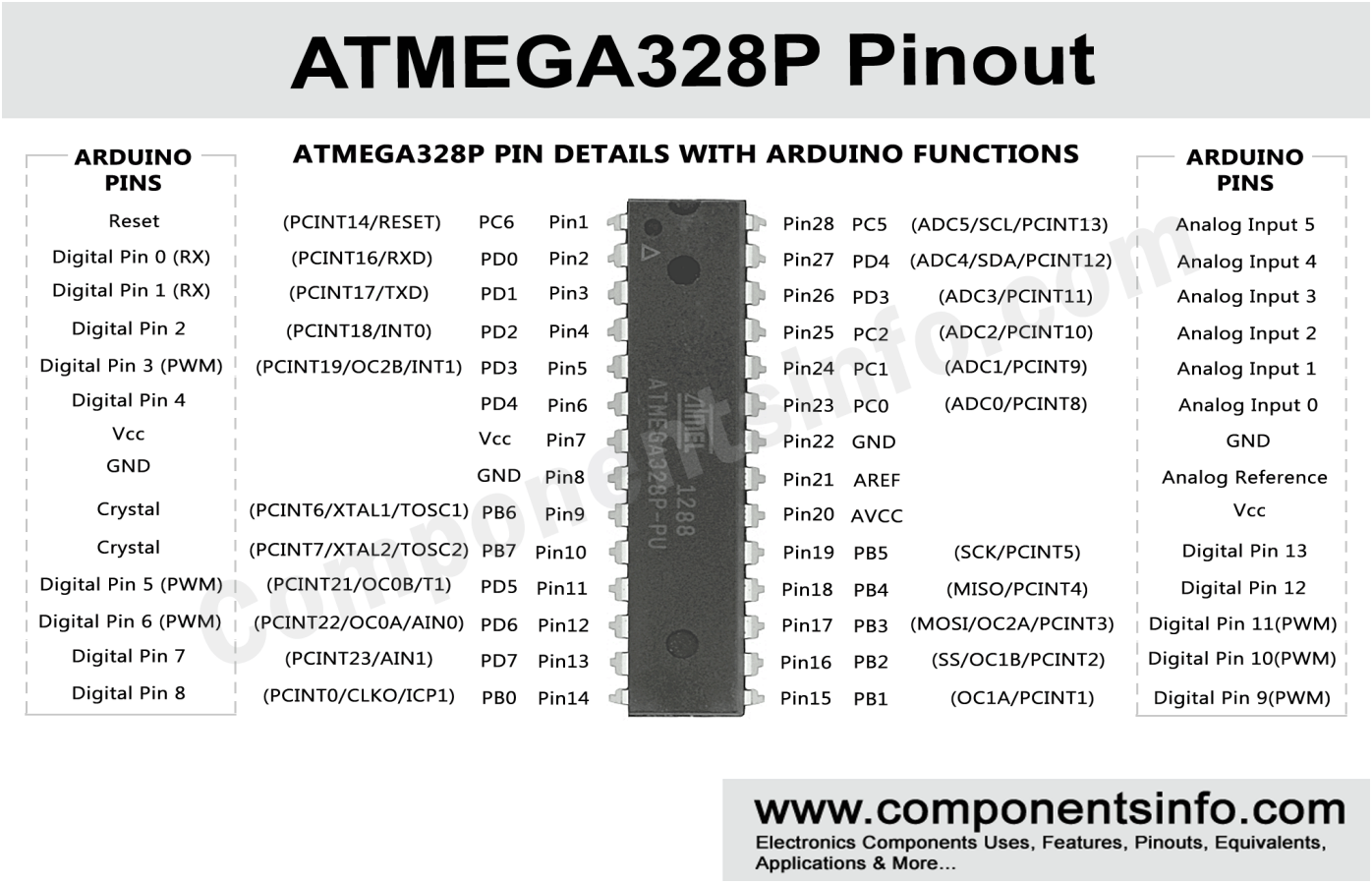


Figura II.1. Definición de pines ATMega328P-Pu

II.1.3 Uso como alternativa de la placa de desarrollo Arduino Uno.

Para todas las pruebas experimentales antes de la creación de las PCBs, se utilizaron dos placas de desarrollo Arduino Uno, los cuales tienen implementado los microprocesadores ATMega328P-Pu, y en cierto modo ahorran tiempo porque en estas tarjetas Arduino Uno viene todo integrado. A continuación (Figura II.2) se muestra la placa de forma robusta, diferenciando cada uno de sus componentes y partes que lo forman. [2]

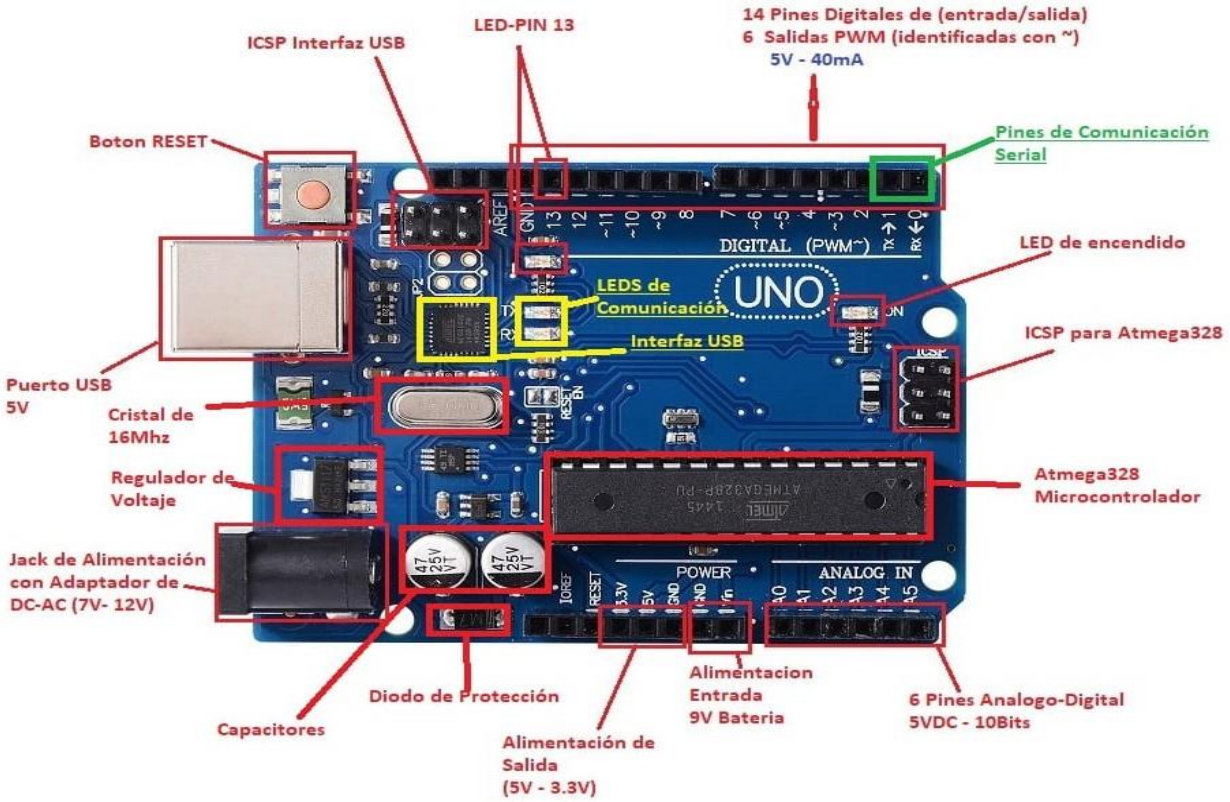


Figura II.2. Componentes de Arduino Uno

Como menciona el Apartado II.1.2, en un primer momento se trabajó y programó con la tarjeta Arduino Uno, pero según fue avanzando este proyecto, se observó una manera de disminuir espacio (algo primordial en cualquier prototipo).

De dicha placa de Arduino solo se estaban requiriendo de unos componentes electrónicos específicos y no de la placa entera y robusta, por lo que se generó unas PCBs con los componentes adecuados, es decir, se generó un Arduino en base a nuestro prototipo.

De dichos Arduino sólo se requerían el Microcontrolador ATMega328P-Pu, el cuarzo (CSTCE16M0V53-R0), la entrada de alimentación, tres condensadores (dos de 22pF y uno de 47uF) y una resistencia de 10KOhmios.

El resto de componentes de dicha placa no se iban a aprovechar por lo que se prescindieron de ellos, creando así en dos PCBs los Arduinos ajustados al requerimiento que se le iba a introducir en este prototipo, aportando también la agrupación de los componentes captando una mayor estética y ahorrando espacio. (Figura II.3).

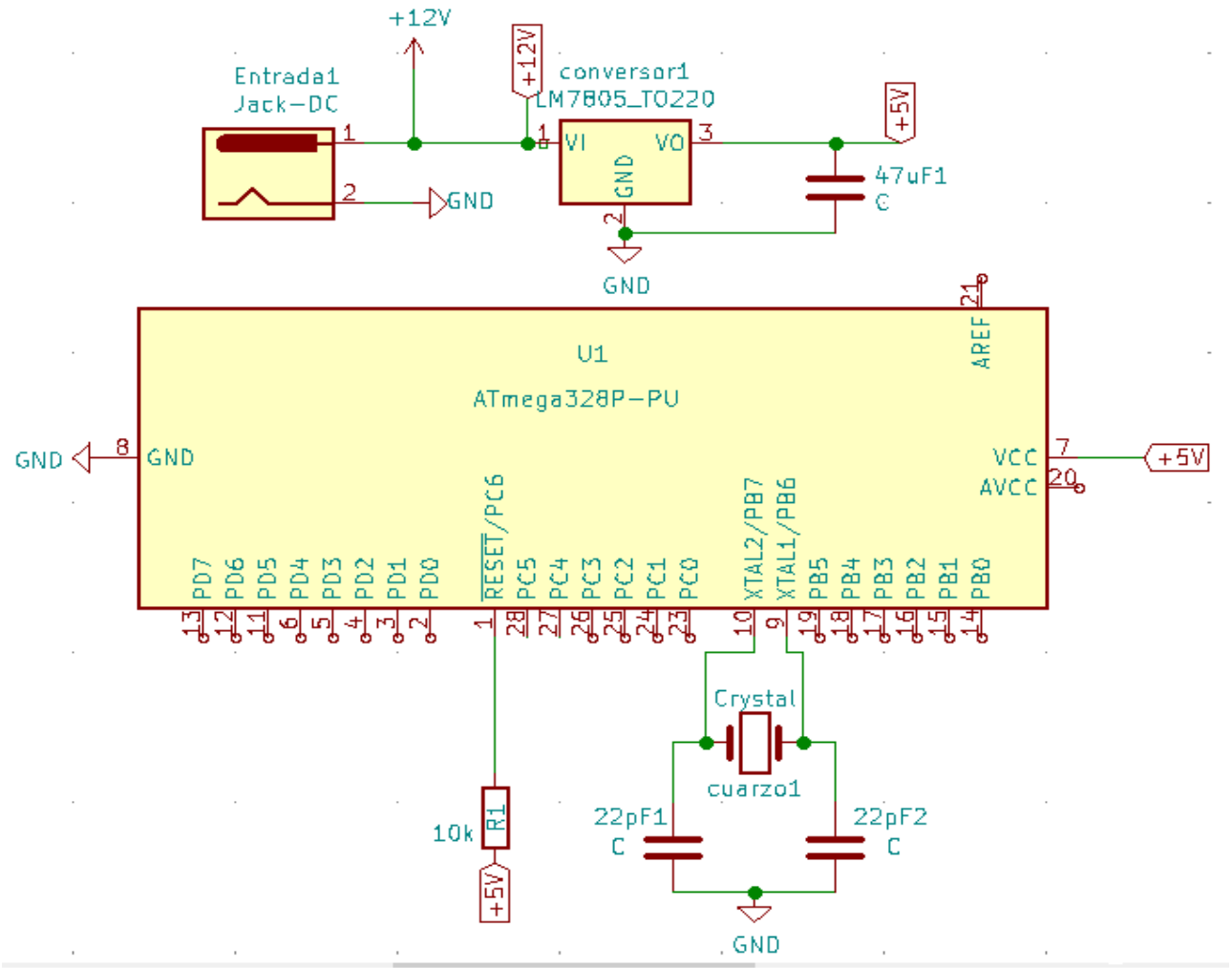


Figura: II.3 Conexionado equivalente Arduino Uno

II.2 Unidades de Entrada y Salida

El dispositivo construido dispone de unidades de entradas, unidades de salidas y algunas de ambas características (entradas/ salidas). Seguidamente vamos a ir detallándolas con mayor profundidad.

II.2.1 Display.

El display LCD 20x4 fue la interfaz que se obtuvo entre el cliente y el prototipo de forma visual. Este dispositivo concede la presentación de caracteres alfanuméricos y otras variables, en dónde, puede llegar a presentar 20 cifras en cada una de las 4 líneas que contiene.

Figura II.4 Parte anterior del LCD

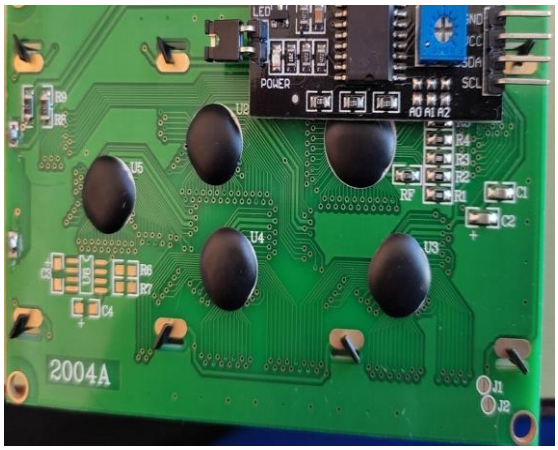


Figura II.5 Parte posterior del LCD

La selección de la pantalla Lcd se caracterizó en que se buscaba una pantalla grande y que pudiera introducir textos con extensión, de ahí se escogió el display Lcd 20*4.

Se le asignó una buena función, ya que realmente es lo que se ve del prototipo, al igual que los pulsadores (Apartado II.2.2), porque el resto se encuentra en una carcasa dónde no se ven los componentes internos.

II.2.1.1 Comunicación.

Para la comunicación del display se requirió de un controlador a través del bus I2C, el cual en reducidas palabras se basa en un dispositivo que permite controlar la pantalla con solo cuatro cables, sin necesidad de usar esa infinidad de cables que estaban acostumbrados en la antigüedad.

El controlador LCD I2C se adquirió por separado, lo que generó una pequeña dificultad. Fue necesario soldar el módulo adaptador I2C al display LED, ya que vienen separados, pero con una simple soldadura se notó una fuerte ventaja para ahorrar el conectar muchos cables a usar únicamente 4 cables, a través del bus I2c.

II.2.1.2 Conexionado.

La conexión ahora sí, se elaboró de una forma sencilla. Simplemente, se alimentó el módulo desde la salida del LM7805 con 5V y mediante GND; y se conectó el pin SDA y SCL (Figura II.1) del Microcontrolador con los pines correspondientes del controlador LCD I2C, (Figura II.6 y Figura II.12)

Figura II.6 Controlador I2C

II.2.2 Pulsadores.

Para este prototipo se necesitó, con una vital importancia, el uso de dos pulsadores (Figura II.7). Estos pulsadores tienen la función de darle una facilidad al entendimiento del prototipo y a las prestaciones que éste realiza.

Basándose en que los pulsadores, fueron a la vez que la LCD, los únicos interfaces visuales que el cliente a la hora de usar el prototipo puede tener a su alcance, pudiendo moverse por el distinto menú con la ayuda de dichos pulsadores y obteniendo una orientación y un control de lo elegido con la parte visual en la pantalla LCD.

Los pulsadores tienen la función de permitir el paso de la corriente eléctrica cuando son oprimidos y en el momento que se deje de oprimir obstaculiza el paso de la corriente. Consta del botón pulsador; una lámina conductora que establece contacto con los dos terminales al oprimir el botón, y un muelle que hace recobrar a la lámina su posición primitiva al cesar la presión sobre el botón pulsador.

Los botones son de diversas formas y tamaños y se encuentran en todo tipo de dispositivos, aunque principalmente en aparatos eléctricos y electrónicos.

Figura II.7 Pulsador

II.2.2.1 Características.

- Configuración de contacto: SPST (1 polo, 1 tiro)
- Capacidad de Voltaje máximo: 50 mA a 24 VDC
- Nota de contacto mínimo: 10 µA a 1 VDC
- Tipo de Contacto: Normalmente abierto
- Número de pines: 2 terminales
- Vida eléctrica: 500,000 ciclos
- Vida mecánica: 500,000 ciclos
- Resistencia de contacto: Max 100 mOhm
- Resistencia de aislamiento: Min 100 MOhm
- Voltaje dieléctrico: 500 VCA por 1 minuto
- Fuerza de actuación: 1.6 a 2.6 N (160 a 260 gf)
- Viaje del actuador: Promedio de 0.25 mm (0.010)
- Dimensiones: 6mm x 6mm x 4 mm
- Longitud de las terminales: 3.5mm

II.2.2.2 Conexionado

Para el conexionando de los dos pulsadores, se conectaron a los pines digitales 4 y 5 del ATMega328P-Pu (Figura II.1). Se conectaron con la configuración pull-down, para evitar los rebotes y el ruido eléctrico.

En el siguiente diagrama se aprecia reflejado el pulsador con los dos pines, uno conectado a GND y el otro terminal conectado a +5V y a su vez conectado a los pines digitales 4 y 5 correspondientes al Microprocesador.

El valor de las resistencias en este caso no fue crítico**,** pero se debe considerar que valores menores causarán que circule una corriente mayor cuando se oprime el botón, mientras que los valores mayores pueden permitir que el ruido se introduzca al pin de entrada. En este caso se usaron dos resistencias de 4,7 KOhmios que está dentro de los valores típicos(1-10KOhms)

La necesidad de utilización de las dos resistencias es para evitar el rebote y mantener en un estado lógico conocido cuando el botón se encuentra en reposo. Esto se debe a que el microcontrolador al carecer de estas resistencias, no es capaz de elegir un estado correcto por sí mismo, lo que se puede dirigir a cualquiera sin ningún orden.

De ahí se generan las dos resistencias ubicadas en los pines 4 y 5 del microcontrolador que se conectaron de forma Pull-Down, ya que es el modo que se estaba buscando, que consiste en que mientras no se presione el pulsador se mantiene en un estado bajo"0/0v", por lo consiguiente cuando se presiona pasa a un estado alto"1/+5V" con la activación, donde se produciría el paso de la corriente eléctrica. [4]

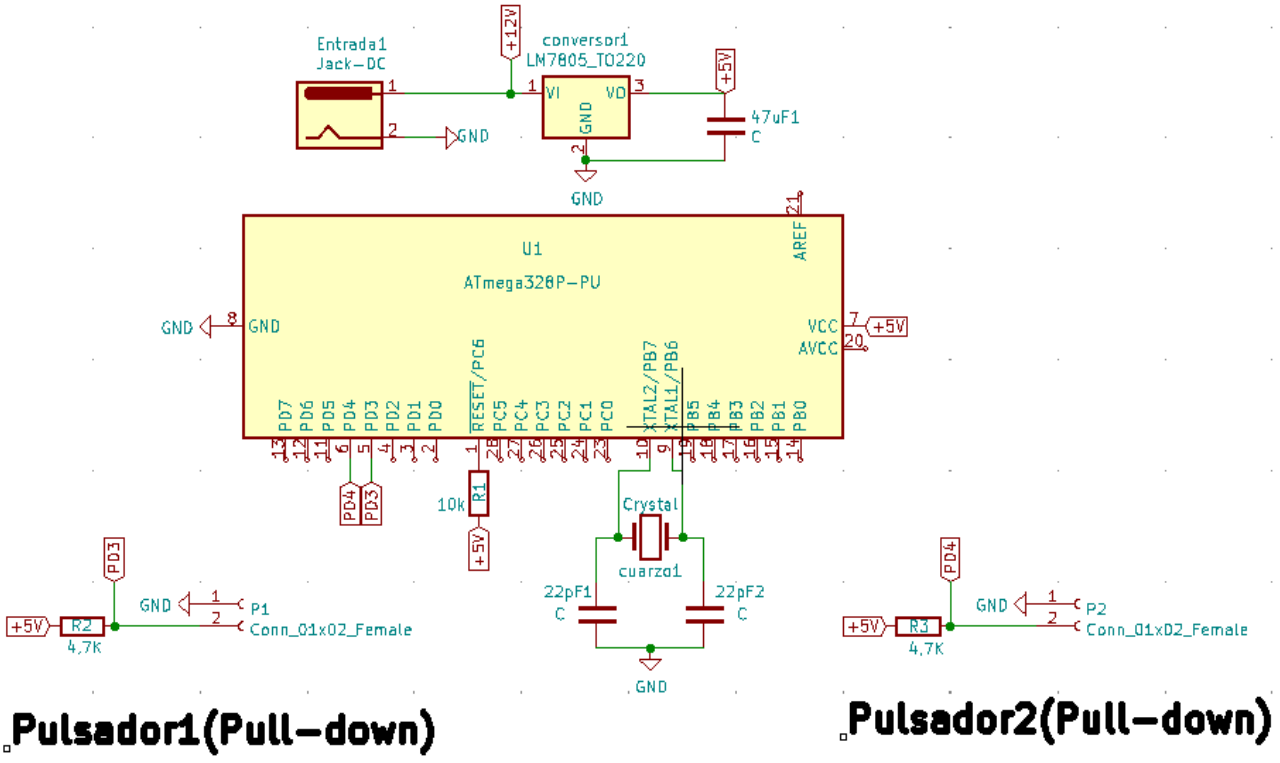


Figura II.8 Circuito con los pulsadores

II.2.3 Reloj.

En este proyecto fue importante el papel de un RTC. El DS1307 (Figura II.9 y Figura II.10) fabricado por Maxim. Fue el escogido, ya que le garantiza observar la hora y el calendario en el habitáculo a través de la pantalla LCD. [5]

Aunque el uso de este módulo puede tener una gran variedad de opciones, como, por ejemplo, se podría modificar la programación para mejorar el encendido de luces, el encender o apagar el módulo para ahorrar batería o todo lo que se nos ocurra, en este proyecto se utilizó con el fin de ceñirse en la muestra de hora y fecha en el display, las otras propuestas se dejaron en pensamientos futuros.

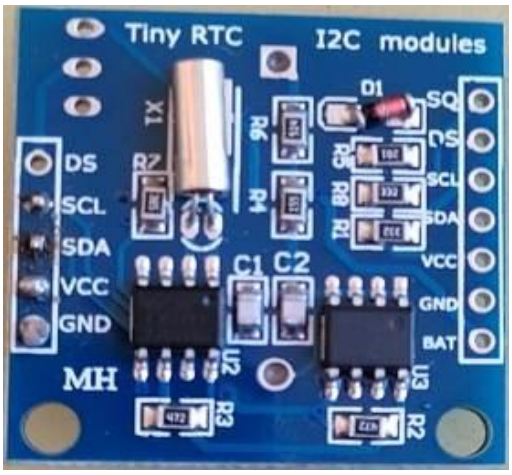


Figura II.9 Parte anterior del Reloj

Figura II.10 Parte posterior del Reloj

II.2.3.1 Características.

El término RTC se basa en un reloj de tiempo real, en el cual, se considera un dispositivo electrónico que le permite la obtención de unidades temporales que se usa en lo habitual, es decir, pueden apreciar tanto, las horas, minutos y segundos, como los días, semanas, meses y años.

Estos dispositivos son capaces de llevar el control correcto, gracias a que disponen de un cristal resonador integrado, son dispositivos tan sorprendentes, que permiten diferenciar los años bisiestos, los meses con sus correspondientes días o infinitas opciones.

Normalmente, los módulos vienen acompañados en su parte posterior de una batería que permiten mantener el valor del tiempo durante varios años, en caso de pérdida de alimentación.

En el modelo DS1307, las variaciones de temperatura que afectan a la medición del tiempo de los cristales resonadores, se traducen en errores en un desfase acumulado. Esto hace que el DS1307 sufra de un desfase temporal, que puede llegar a ser 1 o 2 minutos al día, lo que en la práctica se observan que el desfase se acumula a 15 minutos por lo que se tuvo que adaptar otro sistema (Apartado IV.1)

La comunicación se realiza a través del bus I2C, por lo que es sencillo (el mismo procedimiento que el Apartado II.2.1.1) obtener los datos medidos. La tensión de alimentación es 4.5V a 5.5V para el DS1307.

II.2.3.2 Conexionado.

La conexión es sencilla y similar a la pantalla Lcd (Apartado II.2.1.2), e incluso van conectados en paralelo, con la utilización de los mismos pines que la pantalla Lcd, dicho microcontrolador tiene la capacidad de adoptarles una dirección distinta a cada módulo.

La conexión de un módulo con DS1307 sería la siguiente, con la salida del LM7805 que ofrece 5V y mediante GND alimentamos el módulo. Los pines SCL y SDA (Figura II.1) del microcontrolador se conecta con los pines correspondientes del controlador RTC I2C

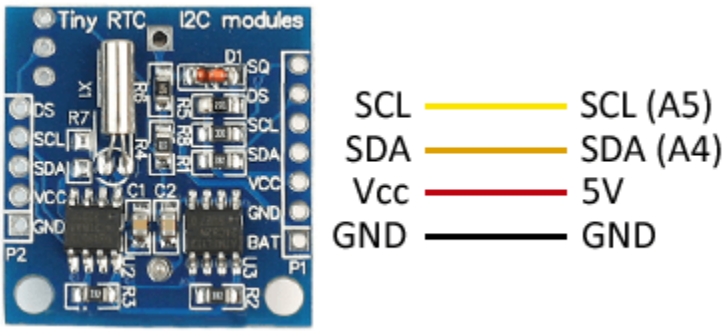


Figura II.11 Asignación de patillaje

Visto desde el microcontrolador quedaría así:

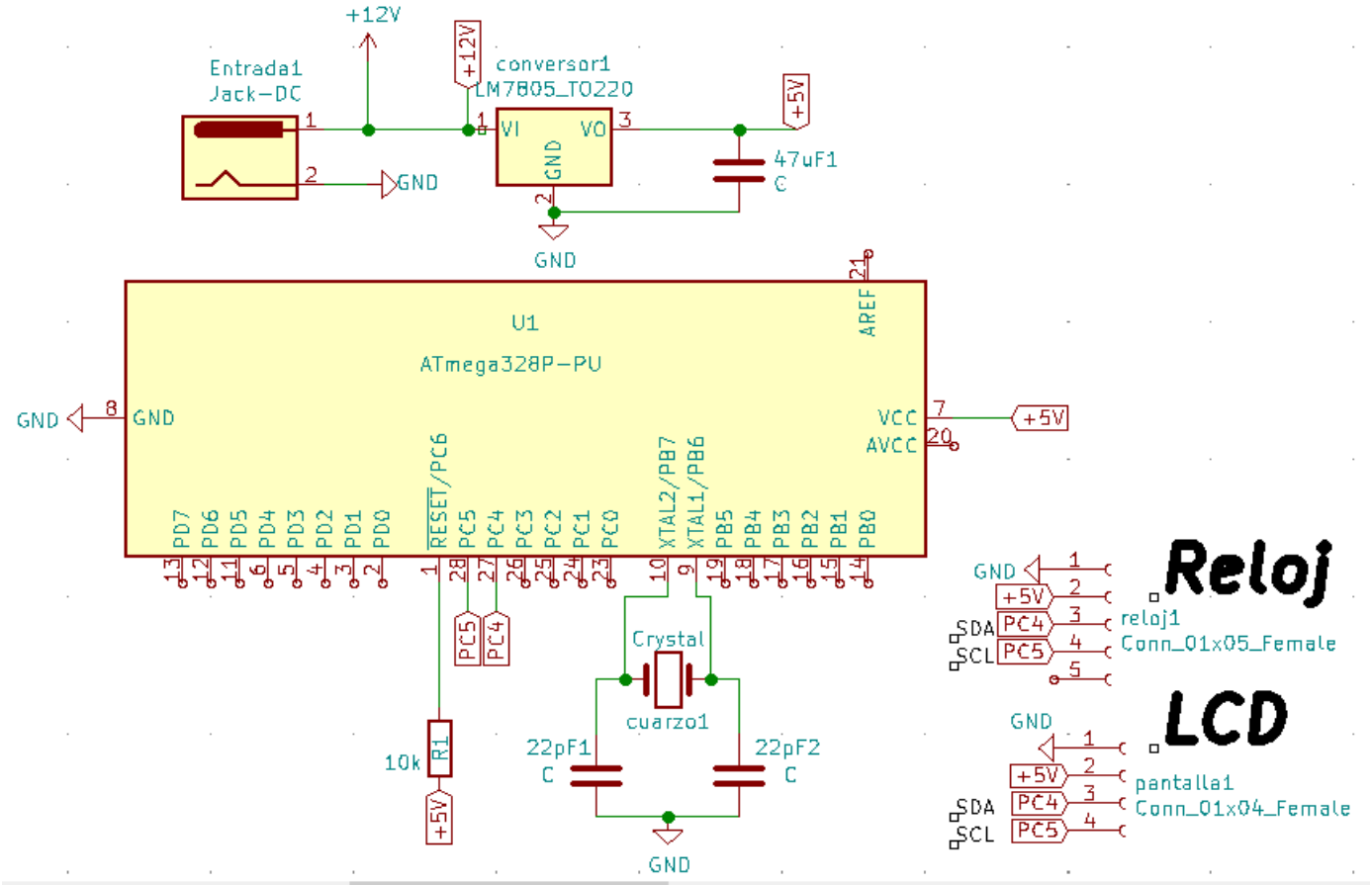


Figura II.12 Conexionado del Reloj

II.3 Sensores.

II.3.1 DHT22

A la hora de buscar un sensor que midiera de forma simultánea, tanto la temperatura como la humedad dentro del habitáculo, nos encontramos los DHT22. (Figura II.13). Presenta en su manera visual un encapsulado de plástico blanco.

Figura II.13 Sensor de temperatura y humedad DHT22

El DHT22 en un sensor que le permite la medida de temperatura y humedad relativa (RH). [6]

El sensor está compuesto por dos partes, un sensor de humedad capacitivo de polímero y un termistor para medir la temperatura. El sensor también dispone de un circuito integrado que hace la conversión analógica-digital y se encarga de enviar los datos digitales por el bus de comunicación

El sensor consta del inconveniente que solo envía los datos cada 2 segundos, que, en este proyecto, con la finalidad que va a tener (no es inconveniente alguno) la temperatura dentro del habitáculo no va incrementar un grado por cada segundo, o por lo menos no sería su uso normal.

El sensor DHT22, contiene unas muy buenas características para el precio que se puede encontrar y sobre todo para la finalidad que se puede llegar a utilizar, consta de:

- Rango de Temperatura de medición entre -40 a 125ºC, con una precisión de 0,5ºC
- Rango de Humedad de medición entre 0 a 100%, con una precisión de 2-5%
- Frecuencia de muestreo de 2 muestras por segundos lo equivalente a 2 Hercios.

Con éste DHT22 se recogen las medidas de temperatura y humedad dentro del habitáculo que con la programación se efectuará para cumplir los requisitos propuestos.

El sensor es un poco más caro comparado al DHT11(misma familia), pero garantizando una mayor fiabilidad y con la finalidad de querer una mayor precisión de temperaturas, no importó en pagar un poco más por un producto que a la larga se le va a dar más rendimiento.

II.3.1.1 Conexionado

Para la conexión del DHT22 se dispone de 3 terminales (Figura II.13), se conecta de manera muy sencilla, con la salida del LM7805 donde obtiene los +5V los administramos a la patilla VCC, la GND a tierra y con la patilla fundamental de DAT a la patilla 11 del ATMega328P-Pu (Figura II.14).

En este caso, se pidió el módulo AM2302, por lo que viene con una resistencia integrada entre los pines de DAT y VCC de 4,7KOhmios, facilitando el sólo conexionado del dispositivo.

El esquema eléctrico se observa en la Figura II.14

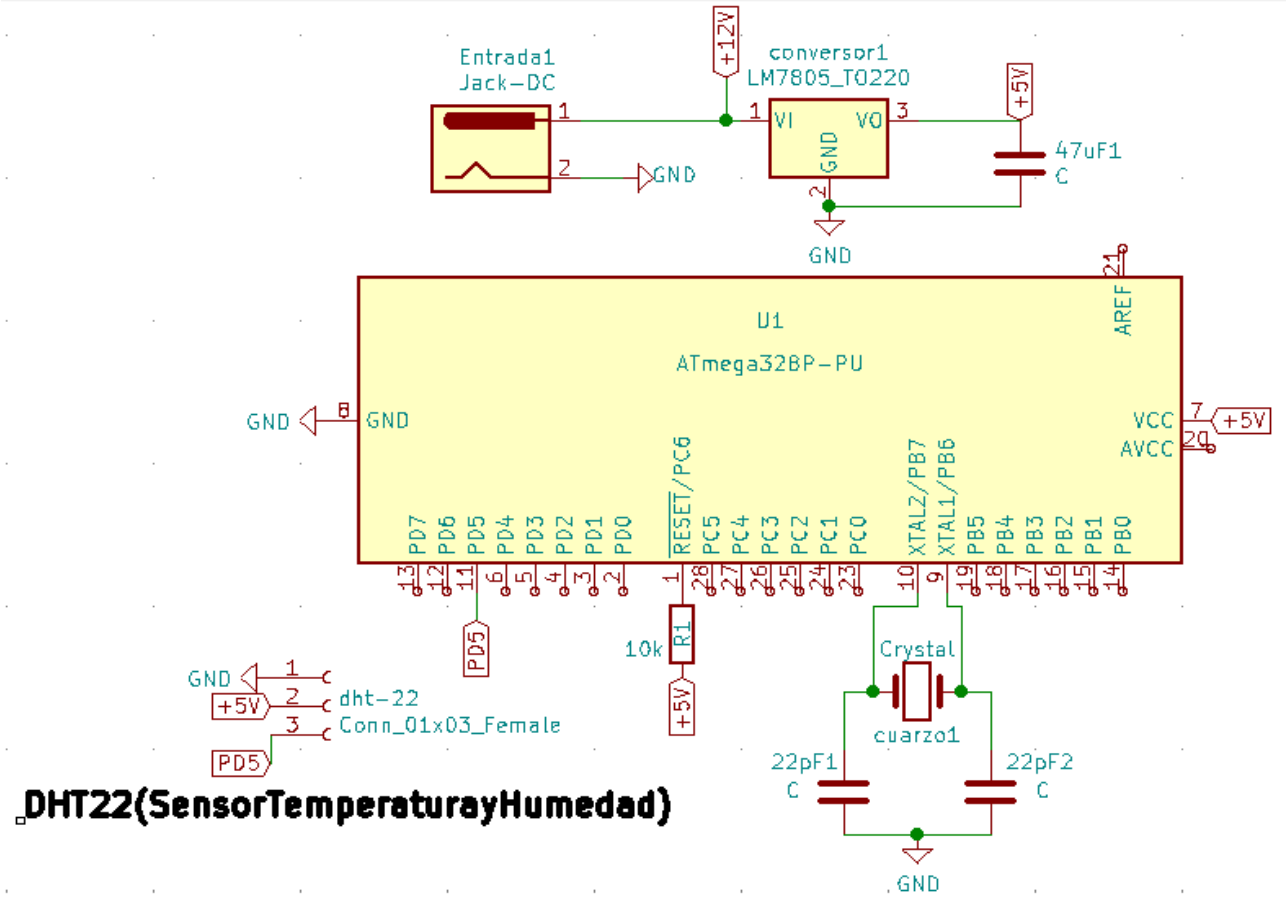


Figura II.14 Conexionado del DHT22

II.3.1.2 Comunicación

Los sensores DHT22 usan su propio sistema de comunicación bidireccional mediante un único conductor, empleando señales temporizadas.

En cada envío de medición el sensor envía un total de 40bits, en 4ms. Estos 40 bits corresponden con 2 Bytes para la medición de humedad, 2 Bytes para la medición de temperatura, más un Byte final para la comprobación de errores (8bit integral RH data + 8bit decimal RH data + 8bit integral T data + 8bit decimal T data + 8bit check sum)

Puede leer los datos del sensor directamente generando y leyendo las señales temporizadas según el protocolo del DHT22. En general, lo normal es que se emplee una librería existente para simplificar el proceso. En este caso se utilizó la librería de Adafruit (Apartado III.2)

II.3.2 DS18b20

Los DS18b20 son sensores de temperatura fabricados por Maxim Integrated. Constan de una salida de datos mediante un bus de comunicación digital que puede ser leído con las entradas digitales del microprocesador ATMega328P-Pu [7]

Disponen de un alto rango de medición, capaces de medir desde -55ºC hasta los 125ºC con una precisión de ±0.5°C.

Una de las ventajas de estos sensores, es que se pueden encontrar tanto en un integrado TO-92 como en forma de sonda impermeable (Figura II.19), la cual se utilizó en este proyecto, ya que una de las sondas se encontraría en la intemperie, pudiendo ser afectada por las temperaturas adversas.

El bus de comunicación, nombrado anteriormente, es el denominado One-Wire, teniendo como ventaja, qué al adquirir el sensor, ya viene el bus de comunicación como parte del precio del dispositivo.

Este bus solo necesita un único conductor para realizar la comunicación y dentro de este mismo bus se pueden instalar tantos sensores como deseen, que en este caso van hacer dos.

El DS18B20, también dispone de un sistema de alarma, que permite grabar en la memoria no volátil del DS18B20 los limites inferiores y superiores. El bus One-Wire permite consultar si algún dispositivo conectado ha activado una alarma.

Figura II.15 Sensor encapsulado DS18b20

La elección de estos sensores fue básicamente por su comodidad, al tener un encapsulado y con la finalidad de que uno de los sensores, se colocará en el exterior del habitáculo, donde en situaciones adversas como la lluvia, se pueda mojar y aun así no perjudicar en absoluto, ya que gracias a este recubrimiento con la sonda impermeable va a seguir mandando la información de temperatura sin variaciones ni complicaciones.

De ahí la elección con el recubrimiento, ya que dicho sensor, viene en formato integrado también pero no se podría poner a la intemperie donde si le afectarían las variaciones de tiempo.

El otro sensor estará ubicado en la circuitería de la nevera, dónde se podría haber usado el integrado, pero por su mayor manejabilidad, se decidió usar éste mismo el cual, cumple su función perfecta.

II.3.2.1 Características

Al hablar de las características del sensor, se puede pensar que es un simple sensor que mide temperatura, pero en su interior está formado por un procesador con múltiples módulos, que se encargan de controlar la comunicación, medir la temperatura, y gestionar el sistema de alarmas.

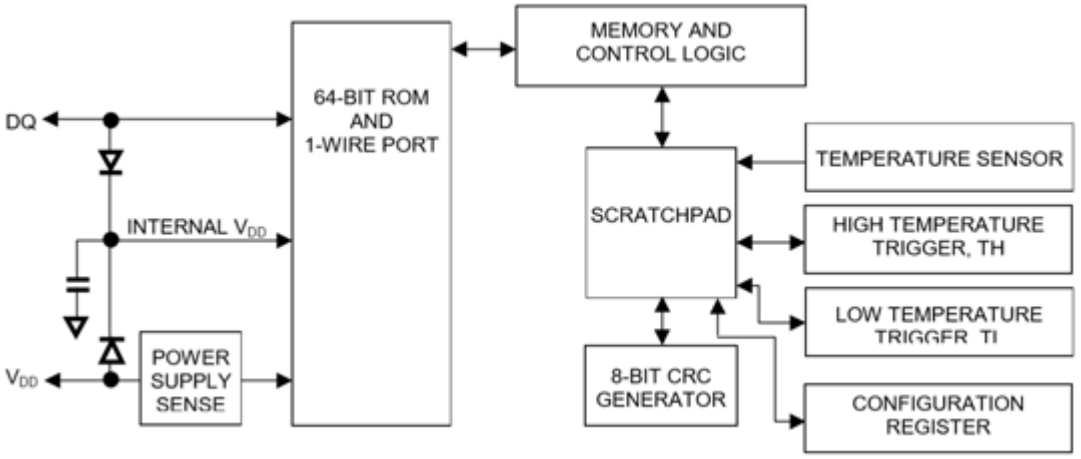


Figura II.16 Diagrama de flujo de trabajo del DS18b20

Como se comenta anteriormente, el sensor puede realizar la transmisión con un solo cable de datos. Para ello, el bus emplea un sistema de timings en la señal entre el emisor y el receptor. El tiempo de adquisición total de una medición de 750ms.

El dispositivo One-Wire permite que todos los dispositivos conectados al bus se alimenten a través de la línea de datos. Para ello, disponen de un condensador que almacena energía mientras la línea de datos está en HIGH. Este modo se denomina "modo parásito".

En caso de no usar el modo parásito, los dispositivos deberán ser alimentados a una tensión entre 3.0V y 5.5V, como es el caso, pero con el uso de una resistencia conectada de forma pull-up de 4k7Ohmios entre Vcc y Vq para que funcione correctamente.

Para poder disponer de múltiples dispositivos en un bus One-Wire cada sensor dispone de una memoria ROM que es grabada de fábrica con un número de 64 bits. Los 8 primeros bits corresponden a la familia (0x28 para el DS18B20). Los siguiente 48 bits son el número de serie único. Los últimos 8 bits son un código CRC.

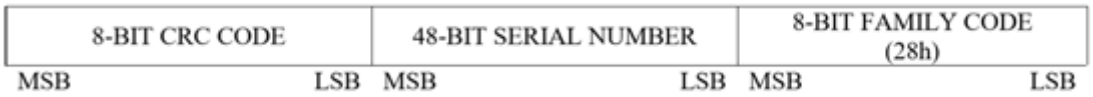


Figura II.17 Capacidad de memoria ROM

Esto significa que podrá llegar a tener 2^48 (más de 218 billones) de DS18B20 conectados en una misma red One-Wire.

La resolución del DS18B20 es configurable a 9, 10, 11 o 12 bits, siendo 12 bits el modo por defecto. Esto equivale a una resolución en temperatura de 0.5°C, 0.25°C, 0.125°C, o 0.0625°C, respectivamente

Que el DS18B20 disponga de una resolución por defecto de 0.0625ºC, no significa que esa sea su precisión**.** Se puede consultar las desviaciones medias a 3 sigma en la siguiente Figura:

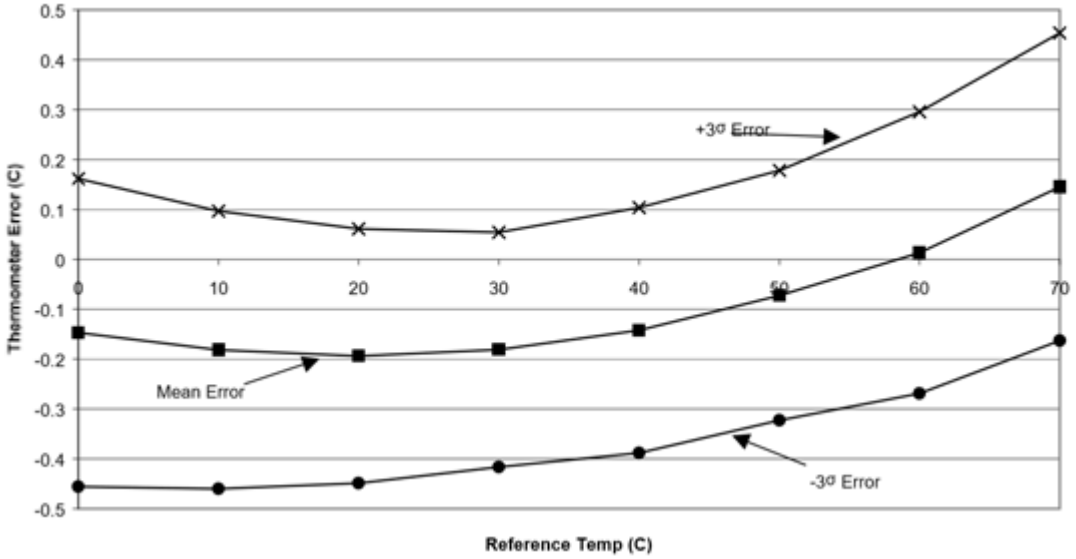


Figura II.18 Gráfica comparativa de temperatura y desviaciones medias.

II.3.2.2 Conexionado

Los dispositivos DS18b20 disponen de tres terminales

- DQ**,** la línea de datos
- VDD**,** línea de alimentación
- GND**,** línea de tierra

En la Figura II.19 se aprecia los dos tipos de formatos que vienen estos sensores, aunque en este proyecto se utilizó la sonda impermeable.

Figura II.19 Tipos de formato del DS18b20

Por tanto, el esquema de conexión (Figura II.20) se muestra dónde se puede conectar el sensor a cualquier entrada digital del ATMega328P-Pu, apreciando como se ha añadido la disposición que ocuparían los sensores adicionales, que en este caso son dos.

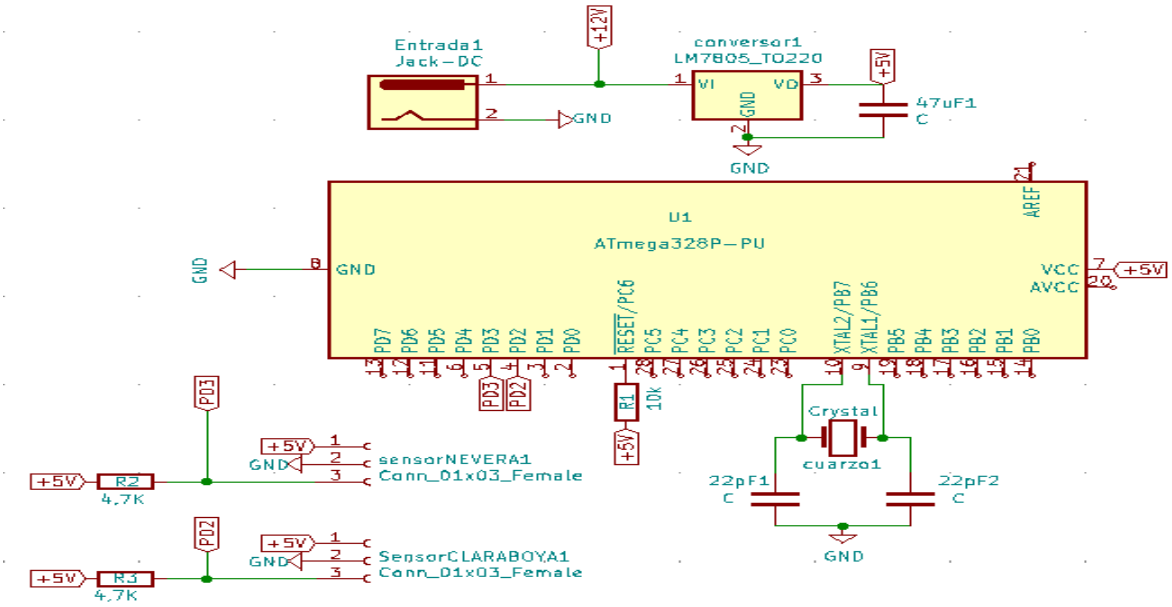


Figura II.20 Conexionado del sensor DS18b20

II.4 Relés y ventiladores.

II.4.2 Relé SIP10A05

A la hora de la selección de los relés se decidió escoger Pan Chang SIP-1A05. En cuánto, a encontrar información de este relé fue complicado, ya que es un dispositivo que proviene de china. No obstante, realiza la función que necesita, su voltaje de funcionamiento es de 5 Voltios y a su vez es un módulo pequeño que en cierta medida es recomendable a la hora de crear una buena estética en el proyecto.

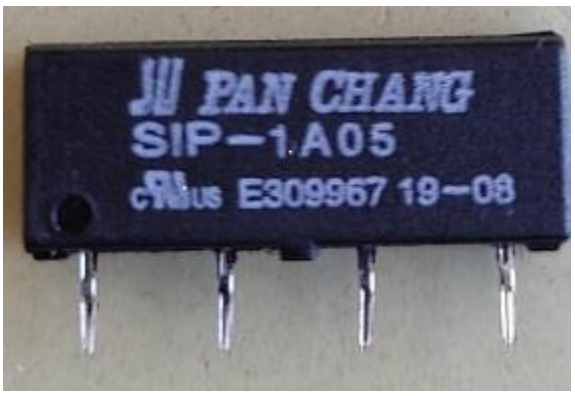


Figura II.21 Relé SIP-1A05

Como ya se comentó en el Apartado II.1, el microprocesador solamente es capaz de suministrar 5VDC y unos pocos miliamperios (40-50 mA) a través de sus salidas digitales.

Aquí es cuando surge el problema, tomando como solución el adquirir un relé de 5VDC que el microprocesador pueda controlar con sus salidas digitales, como si se tratase de un interruptor, para manejar en la salida conmutada (normalmente abierta o cerrada) una tensión de 220VAC y 5A que pueda hacer moverse al ventilador y teniendo en cuenta que la alimentación del circuito de potencia (salida conmutada del relé) no

proviene del microprocesador, puesto que no podría, sino de una fuente externa como puede ser la que un enchufe de la red eléctrica de una vivienda nos suministra, o en este caso la batería de la autocaravana.

Según la RAE (Real Academia Española), un relé es un dispositivo electromagnético que, estimulado por una corriente eléctrica muy débil, abre o cierra un circuito en el cual se disipa una potencia mayor que en el circuito estimulador, es decir, con poca corriente en su entrada, es capaz de conmutar una gran corriente en su salida.

El modelo SIP-10A05 que se utilizó se controla a 5VDC, es decir,[8] cuando le llega una señal a nivel alto proveniente del microprocesador, a su entrada de control, esta conmuta sus salidas (NC y NA), pudiendo controlar una corriente máxima de 10A con tensiones de 250VAC o de 30VDC.

En este caso se alimentará el ventilador con una fuente de tensión (la batería de la propia autocaravana) de 12VDC cuando el relé conmute.

II.4.2.1 Características

Se ha descubierto que hay distintos tipos de relés SIP-1A de la marca Pan Chang. En este caso se compró el SIP-1A05 el número de modelo que pone en letras blancas en el dispositivo. Estos números y letras indican características diferentes tal y como se muestra en la siguiente imagen:

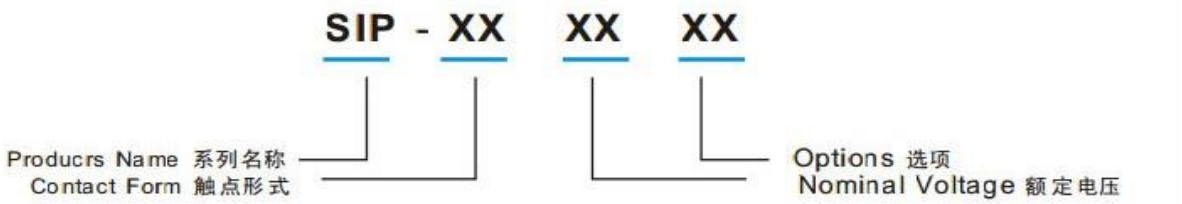


Figura II.22 Denominación de pines

Como este relé es el SIP-1A05 el 1A muestra el modelo, y tendrá de voltaje de operación (voltaje para cerrar el relé, el que se programó) 5 Voltios.

Por esta razón es idóneo para trabajar con el microprocesador, ya que el ATMega328P-Pu se alimenta con 5V.
En la Figura II.23 también aparecen las “Options” que se explican a continuación:

Nil: Std Type
B: Diode
S: Magnetic Shield
Bs:Diode and Magnetic Shield.

En el modelo que se utilizó el relé solo tiene siete caracteres (SIP-1A-05) y no tiene opciones, será el estándar. Si en cambio, en los últimos dos caracteres tiene una "B", el relé tendrá un diodo, si tiene una "S", tendrá un Shield magnético y si tiene "BS", tendrá ambas cosas.

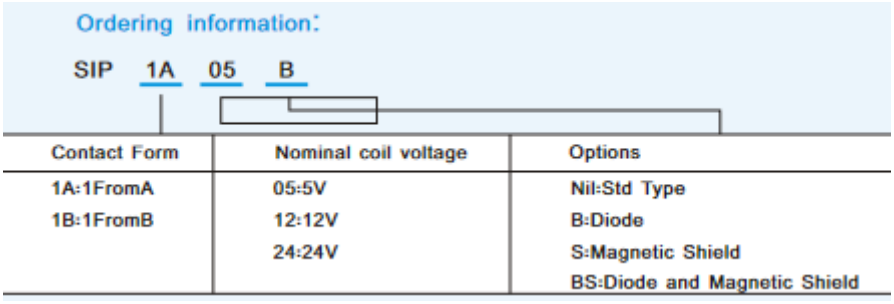


Figura II.23 Tipos de opciones.

II.4.2.2 Conexionado

La forma correcta para conectarlo al ATMega328P-Pu fue la siguiente:

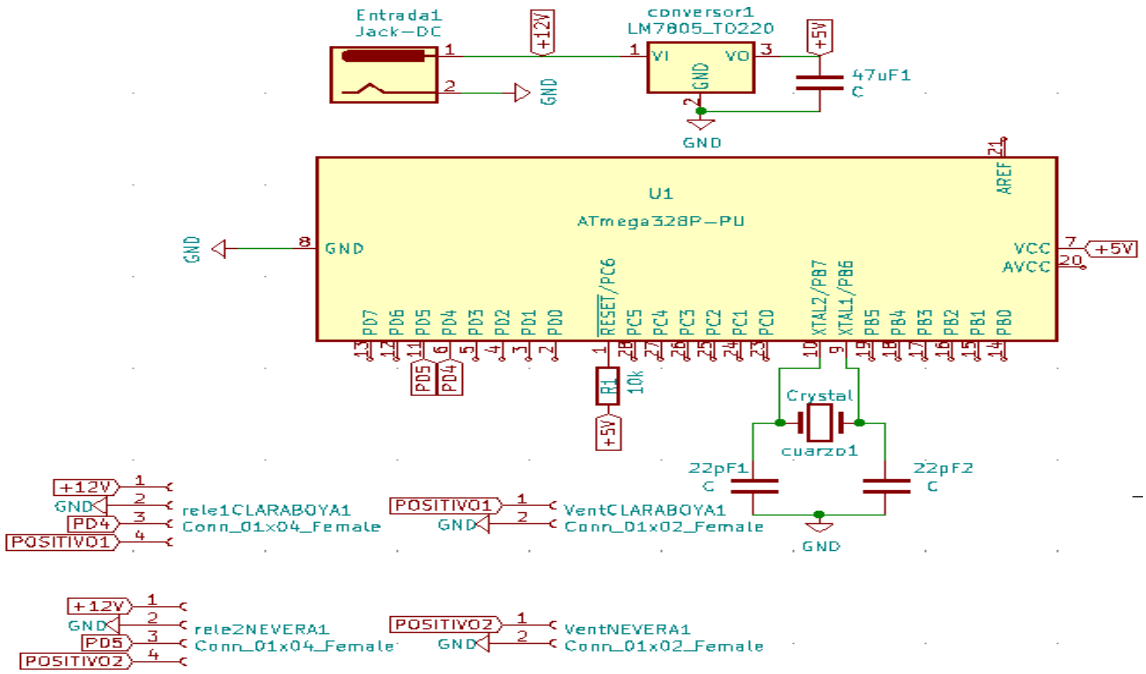


Figura II.24 Conexionado de los relés

Se han enumerado los pines comenzando por la izquierda del 1 al 4.

- En el pin número 1 se conectan los +12V que provienen de la alimentación de la batería.
- En el pin número 2 debemos conectar la tierra (GND) de la batería.
- En el pin número 3 conectaremos al pin digital 4 del microprocesador (en el caso del relé1) o el pin digital 5 (en el caso del relé2)
- En el pin número 4 se conecta al positivo del ventilador, que en el caso de activarse pueda alimentar los ventiladores.

En la Figura II.25 existe una comparación entre los tres tipos de relés existentes de este distribuidor, el de 5V, el de 12V y el de 24V. Observando este relé (SIM-1A05), se observa que el máximo voltaje que soporta en su interior (no el voltaje de operación, sino el que aguanta entre los pines 1 y 4) es 15V, por lo que no se puede conectar la corriente de un enchufe (220V) a este relé porque se quemará.

En la columna titulada "Schematic Contact Form (Bottom View)", columna 3, se ve un esquema de las conexiones un poco más complicado que el de la foto anterior, ya que se aprecia al relé por dentro. En el cuadro también se ve la resistencia que ejerce la bovina, columna 5, pero esto no es de gran importancia en nuestro caso.

Picture	Part Number	Schematic Contact Form (Bottom View)	Nominal Voltage (VDC)	Coil Resistance (ohms±10%)	Nominal Input Power (mW)	Must Release Voltage (VDC)	Must Operate Voltage (VDC)	Maximun Voltage (VCD)
	SIP-1A05	1Form A	5	500	50	3.75	0.6	15.0
	SIP-1A12		12	1000	144	8.60	1.5	30.0
	SIP-1A24	1 +3 -5 7	24	2000	288	17.50	2.5	40.0

Figura II.25 Características de los relés

Cuando se probó este relé, no hace el típico ruido que suelen hacer todos los relés al activarse, sino que este es silencioso y es uno de los motivos por lo que se escogió también dicho dispositivo.
Se comentaron alguna de las características de este relé, pero existen muchas más que se explican el Datasheets(anexo).

II.4.3 Ventiladores

Este pequeño pero potente ventilador mide 80x80x25mm y funciona a 12V.

Es muy común en los ordenadores y sirve también para ventilar todo tipo de proyectos que sean propensos a calentarse demasiado y queramos que se mantengan frescos. En este caso, son de vital importancia para garantizar la función de este proyecto. Además, son muy silenciosos, añadiendo así menor ruido dentro del habitáculo.

Figura II.26 Ventilador utilizado

Características:

- Dimensiones 80 x 80 x 25 mm
- Velocidad 400 - 1500 RPM
- Flujo de aire 47.39 CFM
- Nivel Sonoro 14 dBA
- Rango voltaje 6 - 13.8 V
- Conector 2 pines
- Rodamientos: Sleeve Bearing
- Diseño optimizado para el silencio

El conexionado es simple, pero va incluido con el esquema de los relés como se aprecia en la Figura II.24, donde al activarse los relés da paso a los 12V que son necesarios para la alimentación de estos ventiladores.

II.5 Comunicación Bluetooth

Este prototipo está configurado de un modo Master/Slave. Este funcionamiento puede ser de utilidad en la configuración de dos (o más) placas para compartir información entre sí.

En este caso, dos placas están programadas para comunicarse entre sí en una configuración de Lector Maestro / Esclavo a través del protocolo serie I2C.

El protocolo I2C implica el uso de dos líneas para enviar y recibir datos:

- un pin de reloj serie (SCL) envía los impulsos del microprocesador Maestro a intervalos regulares
- un pin de datos serie (SDA) sobre el cual se envían los datos entre los dos dispositivos.

Dichos pines se conectan de forma invertida a la hora de la conexión entre el microcontrolador y los Módulos.

A medida que cambian las líneas de reloj de LOW a HIGH, un único bit de información se transfiere desde la placa para el dispositivo I2C a través de la línea SDA.

Cuando se envía esta información, la llamada del dispositivo ejecuta la solicitud y transmite datos de nuevo es a la placa sobre la misma línea utilizando la señal de reloj que sigue siendo generada por el Maestro en SCL como la sincronización.

Se han elegido estos componentes por su fácil programación y su bajo coste, pero para su elección también se basó en que este prototipo se va a introducir en una autocaravana, en la que por muy grandes que sean las dimensiones del habitáculo, estos módulos bluetooth se pueden conectar perfectamente sin dar fallos de lectura ni de conexión hasta 15 metros, por lo que no se debe tener problema alguno dentro de la autocaravana.

Se hizo uso de dichos módulos por su facilidad de configuración, con comandos AT como se explica a continuación en los apartados II.5.1.2 y II.5.2.2

II.5.1 HC-05

Para poder transmitir datos de una placa a otra con el método master/ Slave, se ha utilizado el módulo bluetooth, concretamente Hc-05 cuyo aspecto se muestra en la Figura II.27 y Figura II.28; el hc-06 que se explica en Apartado II.5.2.

Figura II.27 Parte anterior del HC-05

Figura II.28 Parte posterior del HC-05

II.5.1.1 Características

El módulo Bluetooth HC-05 es un pequeño modulo transmisor/receptor TTL. Fue diseñado para ser controlado a través de RS232. Permite transmitir y recibir datos a través de bluetooth sin conectar cables a los dispositivos a comunicar. Es un dispositivo fácil de usar y se controla mediante comandos AT por el puerto serie. Es compatible con Arduino o cualquier microcontrolador con UART.

El módulo Bluetooth HC-05 viene configurado de fábrica como "Esclavo" (Slave), pero se puede cambiar para que trabaje como "maestro" (master), como ocurrió en este caso, donde se utiliza de maestro, además, se puede cambiar el nombre, código de vinculación, velocidad y otros parámetros más con comandos AT por el puerto serie como menciono en Apartado 1.8.2.3

Características de HC-05 Módulo Bluetooth maestro esclavo:

- Funciona como dispositivo maestro y esclavo bluetooth
- Configurable mediante comandos AT
- Bluetooth V2.0+EDR
- Frecuencia de operación: 2.4 GHz Banda ISM
- Modulación: GFSK (Gaussian Frequency Shift Keying)
- Potencia de transmisión: <=4dBm, Class 2
- Sensibilidad: <=-84dBm @ 0.1% BER
- Seguridad: Autenticación y encriptación
- Perfiles Bluetooth: Puerto serie bluetooth.
- Distancia de hasta 10 metros en condiciones óptimas
- Voltaje de Operación: 3.6 VDC a 6 VDC
- Consumo Corriente: 30 mA a 50mA
- Chip: BC417143
- Versión o firmware: 3.0-20170609
- Baudios por defecto: 38400
- Baudios soportados: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200.
- Interfaz: Serial TTL
- Antena: Integrada en el PCB
- Seguridad: Autenticación y encriptación (Contraseña por defecto: 0000 o 1234)
- Temperatura de trabajo (Max): 75°C
- Temperatura de trabajo (Min): -20°C
- Dimensiones: 4.4 x 1.6 x 0.7 cm
- Compatibilidad: Arduino UNO R3, Arduino MEGA R3, Raspberry pi 3

El funcionamiento de dicho dispositivo es muy sencillo, pero como ya se comentó puede usarse tanto como maestro o como esclavo por lo que se define primero que es un dispositivo bluetooth HC-05 como maestro y dispositivo esclavo:

- Modulo bluetooth hc-05 como esclavo:

Cuando está configurado de esta forma, espera que un dispositivo bluetooth maestro se conecte a este, generalmente se utiliza cuando se necesita comunicarse con una PC o Móvil, pero en este proyecto se creó con la necesidad de conectarse con el esclavo (II.5.2).

- Modulo bluetooth hc-05 como Maestro:

Es el modo de utilización del dispositivo en donde, el HC-05 inicia la conexión. Un dispositivo maestro solo se puede conectarse con un dispositivo esclavo. Generalmente, se utiliza este modo para comunicarse entre módulos bluetooth. Pero es necesario antes especificar con que dispositivo se tiene que comunicar.

El módulo HC-05 viene por defecto configurado de la siguiente forma:

- Modo o role: Esclavo
- Nombre por defeco: HC-05
- Código de emparejamiento por defecto: 1234
- La velocidad por defecto (Baud rate): 9600

II.5.1.2 Comunicación

Para configurar el módulo se necesita enviar los comandos AT desde un ordenador, este procedimiento se hizo mediante el siguiente esquema:

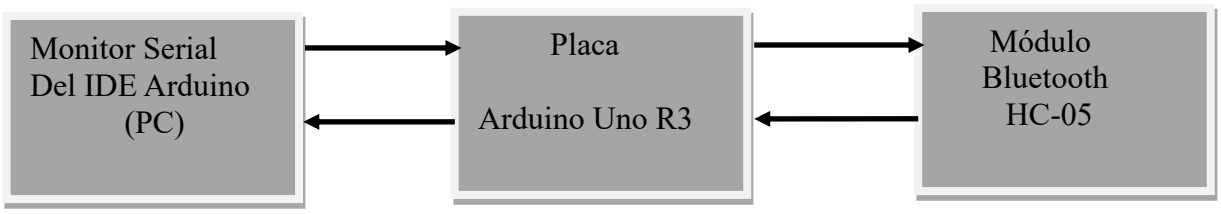


Figura II.29 Diagrama de bloques para el conexionado del HC-05

Una vez realizada la conexión se pudo empezar a enviar los comandos AT al Bluetooth:

- *Test de comunicación*

Lo primero es comprobar si el bluetooth responde a los comandos AT

Enviar: AT

Recibe: OK

Si se recibe como respuesta un OK entonces se continuará.

- *Cambiar nombre del módulo HC-05*

Por defecto el bluetooth se llama “HC-05”, lo que con este comando le cambiará el nombre, asignándole “MAESTRO”

Enviar: AT+NAME=<Nombre>

Respuesta**:** OK

- *Cambiar Código de Vinculación*

Por defecto viene con el código de vinculación (Pin) “1234”, para cambiarlo se tiene que enviar el siguiente comando AT. Es importante este código ya que para configurarlo con el esclavo tienen que tener el mismo código de vinculación que el esclavo.

Enviar**:** AT+PSWD=<"Pin">

Respuesta**:** OK

- *Configurar la velocidad de comunicación:*

La velocidad por defecto es de 9600 baudios, la cual era la que interesaba, aunque se comprobó que estuviera en su velocidad por defecto, para cambiar estos parámetros, se hace uso del siguiente comando AT:

Enviar**:** AT+UART=<Baud>

Respuesta**:** OK

- *Configurar el Role: para que trabaje como Maestro o Esclavo*

Por defecto el HC-05 viene como esclavo, lo que no interesaba. Se cambió la configuración a modo maestro, donde con el siguiente comando permite cambiar esto:

Enviar: AT+ROLE=<Role>

Respuesta: OK

Donde:<Role> 0 -> Esclavo y 1 -> Maestro

- *Configurar el modo de conexión.*

Esta configuración se aplica en el caso de que esté programado como maestro, en el que el módulo necesita saber si se va a conectar con un dispositivo en particular “0” o con cualquiera que esté disponible”1”

Enviar: AT+CMODE=<Mode>

Respuesta: OK

- *Especificar la dirección del dispositivo al cual nos vamos a conectar*

Esta configuración aplica cuando el módulo está configurado como maestro, y a la vez el modo de conexión está en 0 el cual indica que nos va a conectar al dispositivo esclavo en particular. Para especificar la dirección al cual se conectan se usa el siguiente comando AT

Enviar**:** AT+BIND=<Address>

Respuesta**:** OK

Donde: < Address > Es la dirección del dispositivo al cual se vaa conectar. La dirección se envía de la siguiente forma: 5678,56,ABCDEF la cual equivale a la dirección 56:78:56:AB:CD:EF

- *Obtener la dirección del módulo bluetooth*

Esta configuración es muy importante para saber la dirección que el dispositivo trae de fábrica, ya que para la conexión hay que introducirla en el esclavo para realizar la conexión.

Enviar: AT+ADDR?

Respuesta: +ADDR:<dirección>

- *Restablecer valores por defecto.*

Enviar**:** AT+ORGL

Respuesta: Ok

II.5.1.3 Conexionado

Para el conexionado del módulo se tuvo que habilitar dos pines del Microprocesador ATMega328P-Pu (Figura II.1) para hacer la conexión (entradas 10 y 11). Donde las conexiones serían las siguientes:

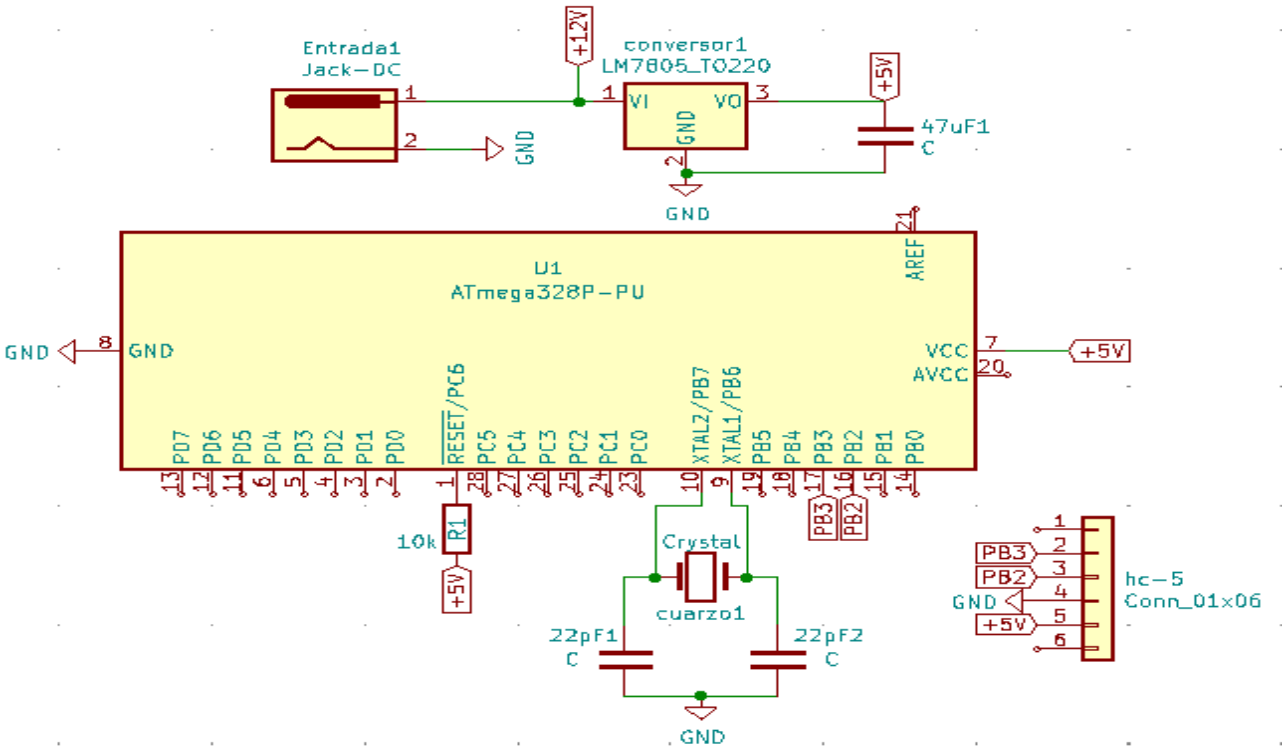


Figura II.30 Conexionado del HC-05

II.5.2 HC-06

Para poder transmitir datos de una placa a otra, se ha utilizado el método Bluetooth, concretamente Hc-06 cuyo aspecto se muestra en la Figura II.31 y el hc-05 que se explica en Apartado II.5.1

Figura II.31 Parte anterior del HC-06

Figura II.32 Parte posterior del HC-06

II.5.2.1 Características

Módulo Bluetooth HC-06 se basa en un dispositivo que soporta conexiones inalámbricas a través del “Bluetooth”. Los módulos Bluetooth se pueden comportar como esclavo o maestro, los cuales sirven para recibir peticiones de conexión y otros para ordenar peticiones de conexión.

Un módulo Bluetooth HC-06 se comporta como esclavo, esperando peticiones de conexión, Si algún dispositivo se conecta, el HC-06 transmite a este todos los datos que recibe del Arduino y viceversa.

El Módulo Bluetooth HC-06 permite la configuración de algunos de sus parámetros de funcionamiento mediante el uso de comando AT. Los comandos AT son una lista de comandos que inician siempre con las letras AT, estos comandos son enviados por medio de un puerto Serie por lo que se necesitará un Arduino o Conversor USB Serial para poder enviar los comandos desde nuestra PC.

Algunos parámetros que vienen por defecto se muestran a continuación:

- Nombre por defecto: “linvor” o “HC-06”

- Código de emparejamiento por defecto: 1234

- La velocidad por defecto (baud rate): 9600

El Módulo HC-06 viene configurado de fábrica como "Esclavo" (Slave) y no puede ser cambiado a "Maestro" como si lo permite el módulo HC-05.

Características de HC-06 Módulo Bluetooth esclavo

- Funciona como dispositivo esclavo bluetooth
- Configurable mediante comandos AT
- Bluetooth V2.0+EDR
- Frecuencia de operación: 2.4 GHz Banda ISM
- Modulación: GFSK (Gaussian Frequency Shift Keying)
- Potencia de transmisión: <=4dBm, Class 2
- Sensibilidad: <=-84dBm @ 0.1% BER
- Seguridad: Autenticación y encriptación
- Perfiles Bluetooth: Puerto serie bluetooth.
- Distancia de hasta 10 metros en condiciones óptimas
- Voltaje de Operación: 3.6 VDC a 6 VDC
- Consumo Corriente: 30 mA a 50mA
- Chip: BC417143
- Versión de firmware: 3.0-20170609
- Baudios por defecto: 9600
- Baudios soportados: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
- Interfaz: Serial TTL
- Antena: Integrada en el PCB
- Seguridad: Autenticación y encriptación (Contraseña por defecto: 1234)
- Temperatura de trabajo (Max): 75°C
- Temperatura de trabajo (Min): -20°C
- Dimensiones: 4.4 x 1.6 x 0.7 cm

Modos de trabajo del HC-06**:**

El Módulo Bluetooth HC-06 tiene dos estados de funcionamiento los cuales es importante conocer:

***** Modo AT (Desconectado)**:**

- Entra a este modo tan pronto alimentamos el módulo, y cuando no se ha establecido una conexión bluetooth con ningún otro dispositivo.

- EL LED del módulo está parpadeando (frecuencia de parpadeo del LED es de 102ms).

- En este modo es cuando se debe enviar los comandos AT en caso se quiera configurar algún parámetro, si se envían otros datos diferentes a los comandos AT, el HC-06 los ignorará.

* Modo Conectado:

- Entra a este modo cunado se establece una conexión con otro dispositivo bluetooth.

- El LED permanece encendido sin parpadear.

- Todos los datos que se ingresen al HC-06 por el Pin RX se trasmiten por bluetooth al dispositivo conectado, y los datos recibidos se devuelven por el pin TX. La comunicación es transparente para el programador.

- En este Modo el HC-06 no puede interpretar los comandos AT.

II.5.2.2 Comunicación

Para configurar el módulo necesitará enviar los comandos AT desde un ordenador, este procedimiento se hizo mediante el siguiente esquema:

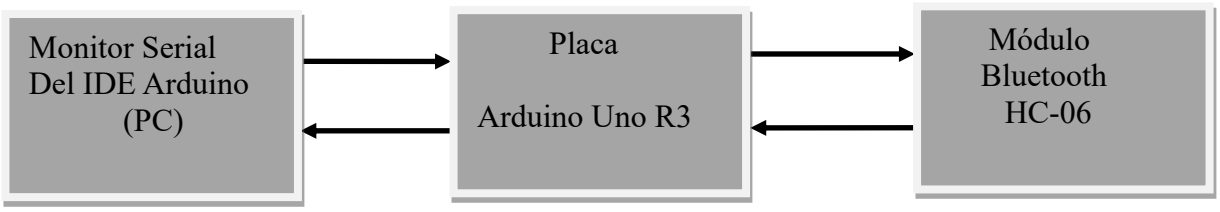


Figura II.33 Diagrama de bloques para el conexionado del HC-06

Como se observa es el mismo que con el hc-05 intercambiando solo el módulo Bluetooth

Realizado lo anterior, se enviará los comandos AT al Bluetooth:

- *Test de comunicación*:

Lo primero es comprobar si el bluetooth responde a los comandos AT

Enviar: AT

Recibe: OK

- *Cambiar nombre del módulo HC-06:*

Por defecto el módulo bluetooth se llama “HC-06” o “Linvor” esto se puede cambiar con el siguiente comando AT

Enviar**:** AT+NAME<Nombre>

Respuesta: OKsetname

- *Cambiar Código de Vinculación:*

Por defecto viene con el código de vinculación (Pin) “1234”, lo que interesa ya que tienen que tener el mismo código que el maestro HC-05, si se quisiera cambiar hay que enviar el siguiente comando AT

Enviar: AT+PIN<Pin>

Respuesta**:** OksetPIN

- *Configurar la velocidad de comunicación:*

La velocidad por defecto es de 9600 baudios, lo que interesó también ya que tienen que coincidir con el maestro, si se quisiera cambiar se hace uso del siguiente comando AT:

Enviar**:** AT+BAUD<Número>

Respuesta**:** OK<baudrate>

Donde <Número> equivale a una velocidad de <baudrate>, los valores pueden ser:

Número---baudrate

1 ---------1200; 2 ---------2400; 3 ---------4800; 4 ---------9600 ;5 ---------19200; 6 ---------38400; 7 ---------57600; 8 ---------115200

II.5.3.3 Conexionado

Para el conexionado del módulo se tuvo que habilitar dos pines del Microprocesador ATMega328P-Pu (Figura II.1) para hacer la conexión (entradas 10 y 11). Donde las conexiones serían las siguientes:

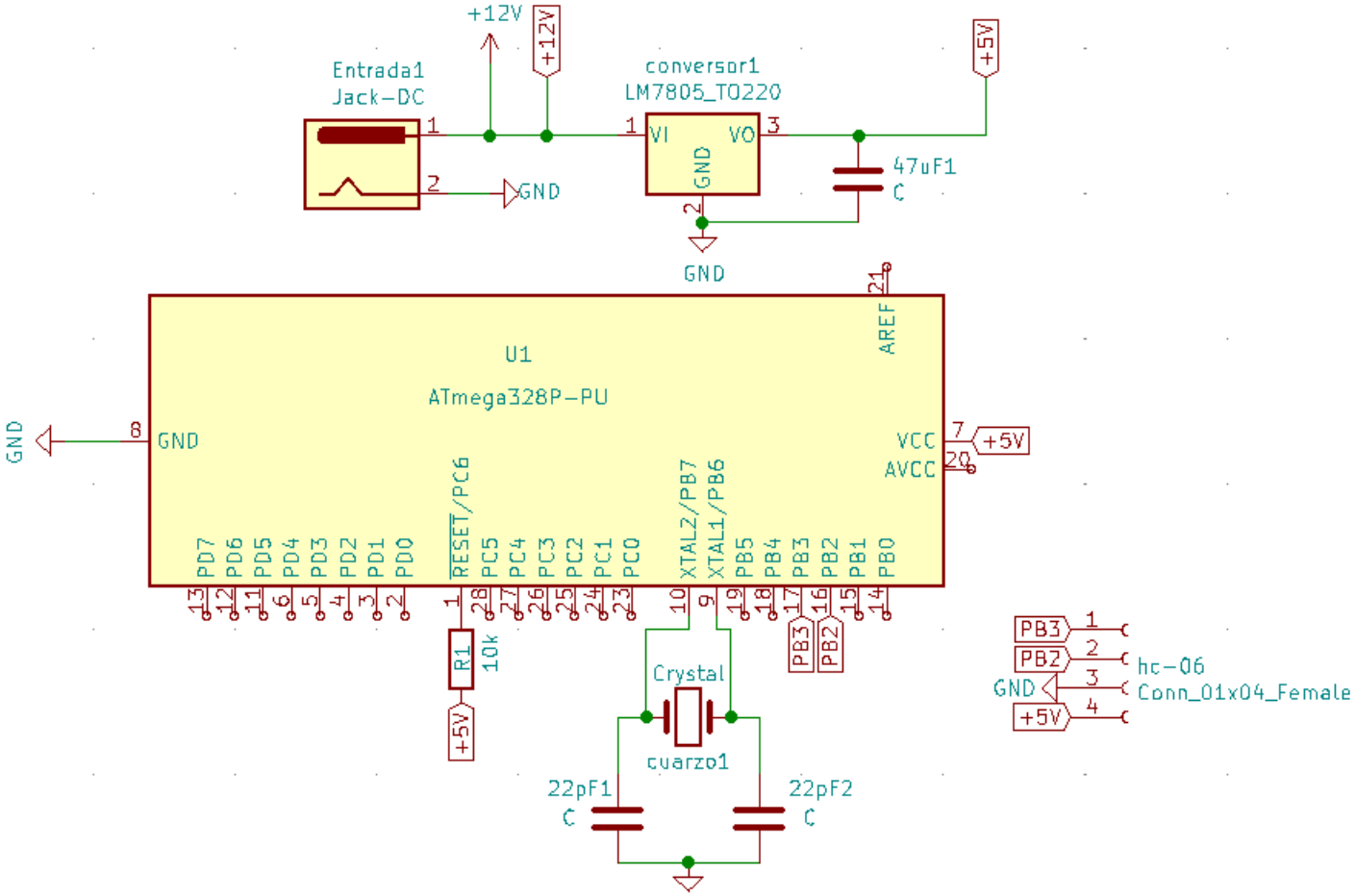


Figura II.34 Conexionado del HC-06

Capítulo III: Herramientas de diseño y desarrollo

III.1 IDE ARDUINO

El Arduino IDE es un entorno de desarrollo integrado (Integrated Development Environment) en el que se realiza la programación de cada una de las placas de Arduino.

Para la programación de Arduino se ha utilizado el Arduino IDE1.8.5, que es un entorno de desarrollo en el que se realiza la programación de estas placas.

Este entorno está basado en C++, ofreciendo unas librerías que facilitan la programación de los pines de entrada/salida y de los puertos de comunicación. Se basa en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Con este entorno de programación se podrá cargar el programa compilado en la memoria flash del Arduino [4]. En este proyecto se ha programado el microcontrolador ATmega328P. La interfaz de este entorno se puede apreciar en la siguiente figura:

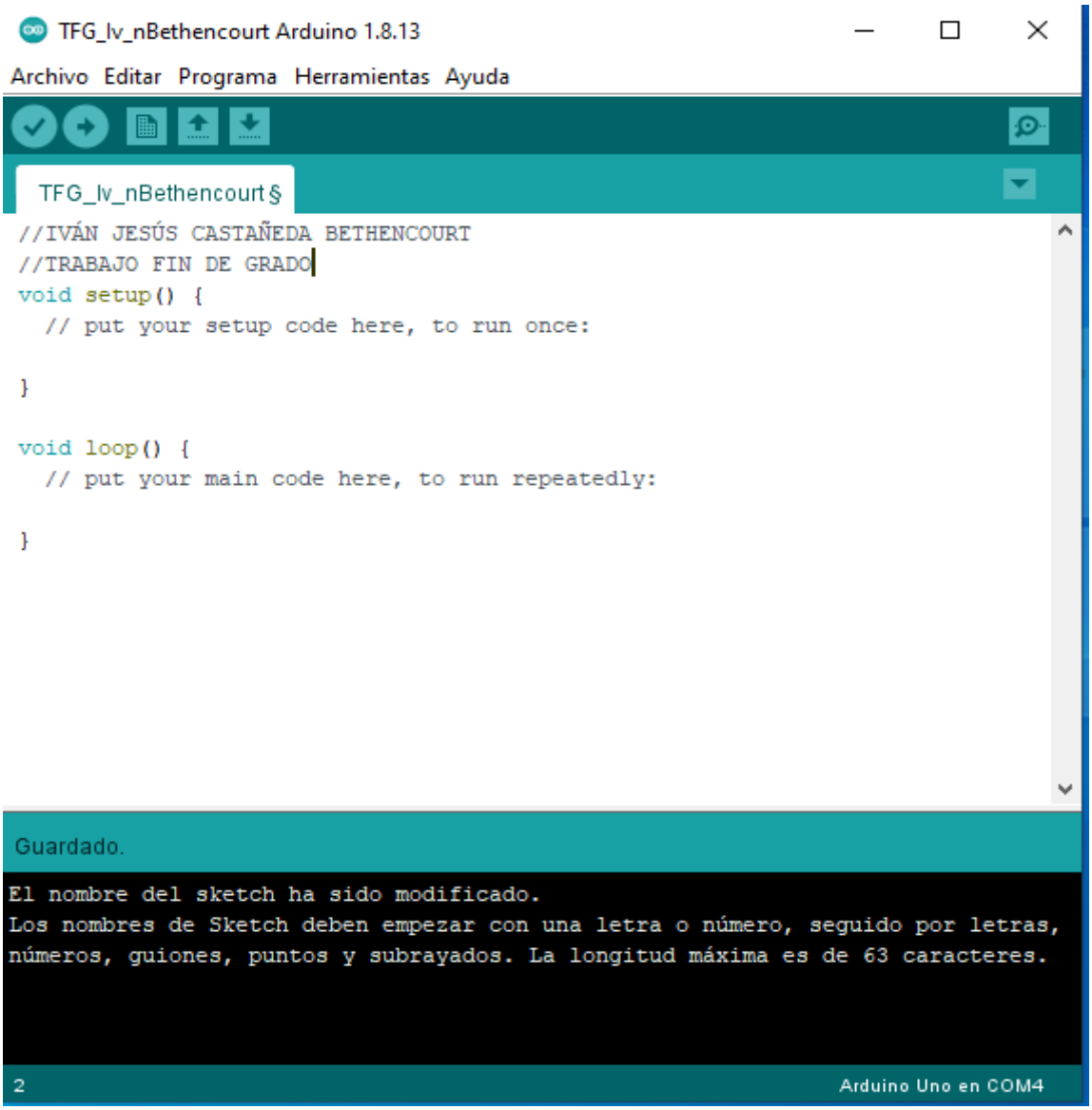


Figura III.1 Vista general del IDE Arduino

III.2 Librerías utilizadas

Para facilitar la programación se ha utilizado librerías ya creadas por terceras personas, las cuales se ubican en el IDE de Arduino. No obstante, no todas las librerías de los sensores que se han utilizado se encuentran en el IDE, pero si siguen siendo creadas por terceras personas. A continuación, se muestran las librerías usadas que no están incluidas:

- DS3232RTC es la librería utilizada para programar el RTC DS3231. Creada por Jack Christensen bajo la licencia GNU GPL v3.0. En ella se incluye la librería “Wire”, que nos permite comunicarnos a través del bus I2C, y la librería “TimeLib”.

  Con esta librería se tiene implementado el calendario hasta el año 2100 teniendo en cuenta años bisiestos y meses con diferentes números de días

- One-Wire y Dallas Temperature fue la librería utilizada para leer las temperaturas del DS18b20.Con estas conseguimos enviar y recibir datos por un único cable. Esto tiene sus ventajas e inconvenientes, pero no deja de ser una opción muy a tener en cuenta cuando van justos de conexiones de control para nuestro proyecto, ya que se ahorra un cable en cada sensor

- Para el uso del sensor DHT22 se utilizó la librería “Dht.h” de Adafruit. Con esta librería pueden realizar fácilmente la lectura de ambos sensores y no preocuparse por el protocolo de comunicación entre Arduino y los sensores. Después de descargar e importar la librería se puede empezar a programar nuestro sketch.

- Para el uso de los botones, se utilizó la librería GFbutton.h, proveniente de GeekFactory. Esta librería trae algunas funciones creadas, que se pueden utilizar directamente y facilita la programación.

- Para asignar dos pines de transmisión, se usará la librería SoftwareSerial.h, la cual ha sido desarrollada para permitir la comunicación serie en otros pines digitales del Arduino, usando el software para replicar la funcionalidad. Es posible tener múltiples puertos serie de programas con velocidades de hasta 115200 bps

- Para el uso de la pantalla LCD se utilizó la librería LiquidCrystal_I2C.h creada por Adafruit. Consta de unos parámetros ya programados, donde se pueden realizar los cambios y dejar funcionando según los requerimientos.

III.3 Kicad

Para el diseño de las PCBs se utilizó KiCad, un entorno de software libre utilizado para el diseño de circuitos impresos. En la Figura III.2 se pueden ver como se han hecho las conexiones en este caso del maestro, mostrándose en ella las pistas “forward” y “bottom” además de los drills. Se puso esta figura para mostrar como son las características del programa, ya que los circuitos en pcb y los fotolitos están en el anexo 2.

Es un programa bastante útil, ya que se pueden generar las PCB de forma gratuita y da la posibilidad de ver la placa y sus componentes en 3D, una buena estrategia para ver cómo estaba quedando la placa antes de su impresión.

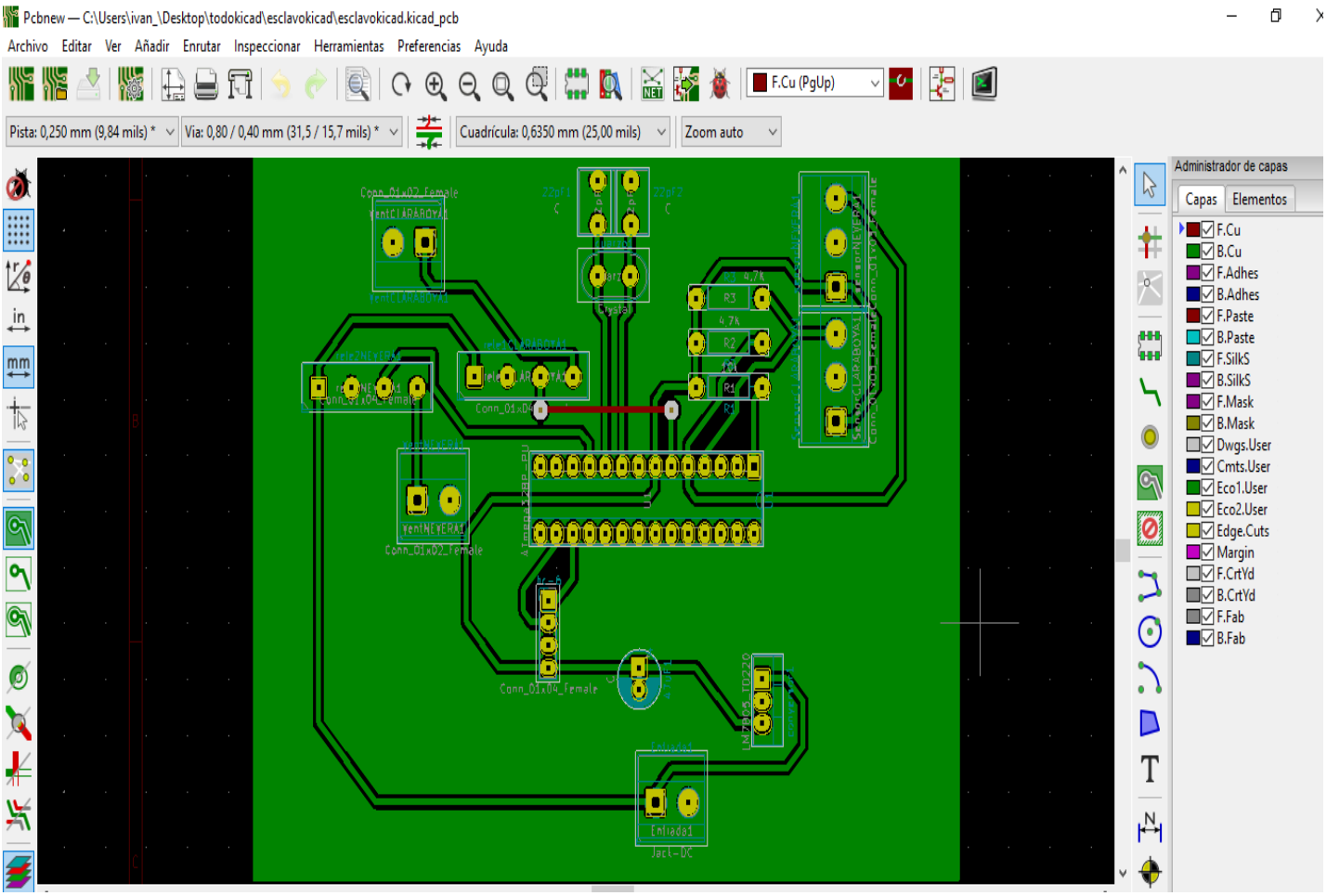


Figura III.2 Vista general del Kicad

Capitulo IV: Realización del sistema

El sistema de control de temperatura del proyecto hace uso de dos bloques, uno de control que llamaremos a partir de ahora Maestro y otro de actuación que se denominará a partir de ahora Esclavo.

El maestro es el que da las órdenes al esclavo y a su vez las recibe del usuario(cliente) que da ordenes al maestro, el las almacena y se encarga de pedir en cada caso actuaciones al esclavo para intentar lograr lo que el usuario desea ver.

La idea es controlar, en la medida de lo posible, la temperatura que hay en la circuitería de la nevera y la temperatura que hay dentro del habitáculo, para ello usamos estos dos bloques.

IV.1 Bloque maestro

El bloque del Maestro es el encargado de recibir las órdenes del usuario, lee temperatura y humedad y la pone en el display, éste a su vez le manda órdenes al esclavo.

La comunicación entre el maestro/ esclavo se realiza mediante los módulos Bluetooth, pasando información del maestro al esclavo cuando este se la pide, que a su vez, es solicitada por el usuario.

Este módulo Bluetooth, tanto recibe órdenes del cliente al accionar los botones, como ejecuta ordenes hacia el esclavo para pedir la información que requiere el cliente, o si no es por el cliente, por las condiciones ordenadas.

El funcionamiento del maestro comienza con un modo automático donde la Lcd muestra en todo momento una pantalla principal con el menú:

Figura IV.1 Menú real del prototipo

Este el menú al que el usuario va a estar en contacto en todo momento. En la imagen principal del menú se puede observar la fecha y hora en todo momento, gracias al RTC Ds1307 que nos garantiza observar la hora y el calendario en nuestro habitáculo.

Gracias a la batería con la que consta el reloj, garantiza que bajo cualquier circunstancia en la que se quede sin alimentación el prototipo, éste al volver a recobrar su alimentación es capaz de ajustar la hora y fecha actual.

Pero se encuentra, con que esta batería no era capaz de almacenarla exactamente, sino con un retardo de unos 15 minutos cuando se desconectaba el dispositivo. Por lo que se llegó a la conclusión de adaptarle una programación, que a la hora de que el prototipo se quede sin alimentación, en el momento de reinicio, se pida tanto la fecha como la hora exacta, pudiendo así dejar la hora y fecha totalmente actualizadas y pasándole la información al módulo RTC Ds1307 encargado de seguir con el proceso de manera correcta.
Este proceso se realiza sólo cuando se queda sin alimentación total el prototipo.

Figura IV.2 Selección del día

Figura IV.3 Selección del mes

Figura IV.4 Selección del año

Figura IV.5 Selección de hora Figura IV.6 Selección de minuto Figura IV.7 Selección de segundo

Con la ayuda de los botones se selecciona la fecha y hora acabando en modo de selección de “0” si hay equivocación al seleccionarla (Figura IV.8), o “1” si se ajusta la fecha y hora correcta para seguir con el código establecido (Figura IV.9).

Figura IV.8 Selección de error hora Figura IV.9 Selección de hora correcta

A su vez se observa la temperatura y humedad dentro de la autocaravana (Figura IV.1) gracias al sensor de temperatura y humedad DHT22, este sensor ofrece la temperatura ambiente, la cual juega un papel importante de comparación con las otras dos temperaturas que se nombran más adelante.

El usuario tendrá acceso a dos botones(Figura IV.2). En la carcasa principal, un botón para desplazarse por el menú y otro botón para seleccionar la opción que desee escoger.

Figura IV.2 Pulsadores reales

El usuario como se comenta tiene dos opciones de selección en el menú, en el que al desplazarse entre ellos se visualiza un icono de selección, para que este usuario sepa en todo momento en que sección está, este icono se refleja como “>”

En el momento que el usuario quiera activar la primera opción “estados”, el maestro le pregunta al esclavo las temperaturas de los sensores (DS18b20) distribuidos por la autocaravana, uno está ubicado en la claraboya y el otro está ubicado en la circuitería de la nevera.

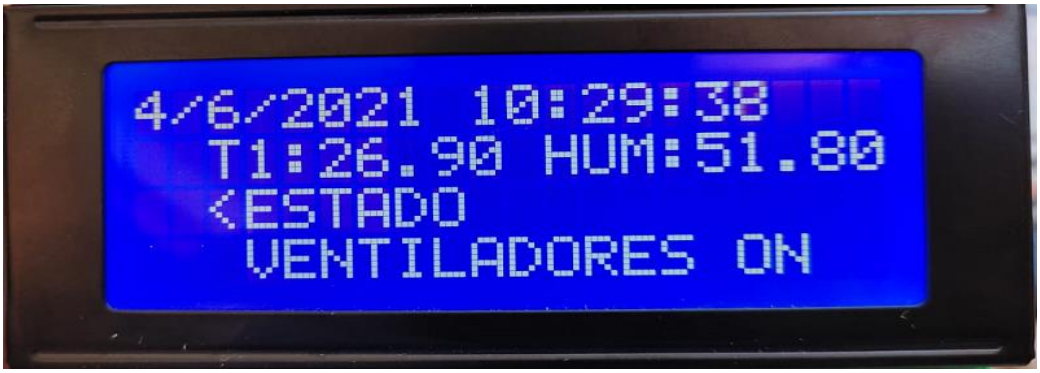


Figura IV.3 Seleccionador activado en Estado

Al usuario seleccionar esta opción entra en este submenú (Figura IV.3) al cual se llama caravana inteligente, dando los valores de las temperaturas obtenidas.

Figura IV.4 Submenú de Estado

La otra opción desde el menú principal es “ventiladores “(Figura IV.5)

Figura IV.5 Seleccionador de Ventiladores

En esta selección del menú se trata de un activado de los ventiladores en el momento que el usuario quiera, pero con ciertas condiciones explicadas en apartado control manual a continuación.

Este prototipo tiene dos modos de funcionamiento:

- MODO AUTOMÁTICO: Este es el modo que está en su funcionamiento normal. El maestro recibe cada dos segundos la temperatura y humedad dentro del habitáculo. A su vez, el maestro le pregunta al esclavo la temperatura de los dos sensores distribuidos por la autocaravana, dónde se ha programado que el sensor de temperatura de la claraboya en el momento que supere la temperatura del interior, se active el ventilador de dicha claraboya con el objetivo de extraer el aire caliente de dentro del habitáculo.

  Por otro lado, existe el otro sensor, ubicado detrás de la circuitería de la nevera. Este es el gran objetivo de este proyecto, donde el maestro le pregunta al esclavo la temperatura de dicho sensor, si la temperatura supera los 55ºC se activará el ventilador con el objetivo de enfriar la circuitería de la propia nevera y el compresor. Éste es el principal problema de la refrigeración de las neveras, que trabajan en un margen de 55-70ºC. Si superan esos rangos de temperatura, dicha circuitería sufre mucho y tienden a estropearse.

- MODO MANUAL: Este modo, el usuario al seleccionarlo activa los dos ventiladores, sea cual sea la causa, si el usuario quiere activarlos solo con accionarlo se quedan encendidos.

  El modo de desactivarlo, es más interesante, ya que el usuario puede desactivarlo también en cualquier momento que el desee con solo accionar el botón, pero con la condición de que, si una de las condiciones explicadas en modo automático está activada, el ventilador por mucho que se le seleccione OFF, no se va a apagar ya que está cumpliendo la orden correspondiente.

IV.2 Bloque esclavo.

Este bloque está enviando cada 5 segundos mediante el módulo Bluetooth HC-06, la temperatura de los sensores, ya comentados, distribuidos por la autocaravana.

Es un bloque con el objetivo de realizar las órdenes que el maestro le ejecute. Como bien se aprecia, este bloque está enviando la información cada 5 segundos, ya que, en un modo automático, se va ejecutando el programa con normalidad. A la hora de que el maestro le pide información con el menú “estado”, estos 5 segundos pasan a 1 segundo, dando la información que el usuario requiere en el mayor tiempo posible.

El esclavo consta de dos ventiladores ubicados en distintas zonas como se comentó con anterioridad, en el que por programación al cumplirse las condiciones se activan los relés dando paso a la activación de los ventiladores.

Es un bloque de espera, es decir, espera la orden del maestro, pero a su vez, va enviándole información de las temperaturas para que el programa en modo automático siga su procedimiento.

Capítulo V: Control y gestión del sistema (Software)

Para el correcto funcionamiento del sistema de refrigeración de la autocaravana, fue necesario crear la comunicación entre un Maestro y un Esclavo, donde el Maestro es el encargado de solicitar la información necesaria al Esclavo ya sea porque el cliente lo pida para asesorarse, o para poner en funcionamiento o detener los ventiladores.

A continuación, se muestran dos diagramas de flujo explicando el funcionamiento del intercambio de información en el código programado.

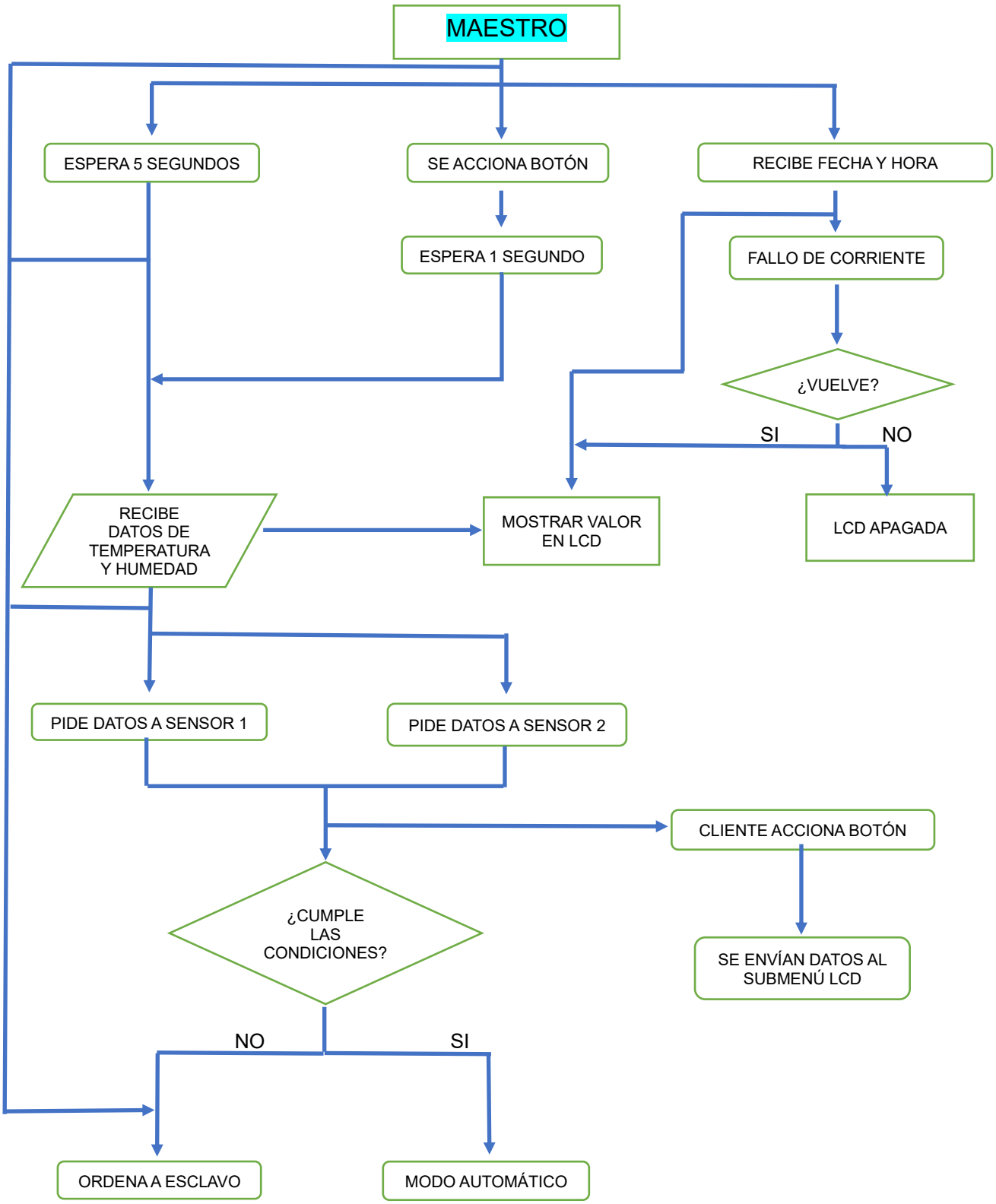


Figura V.1 Diagrama de flujo del Maestro

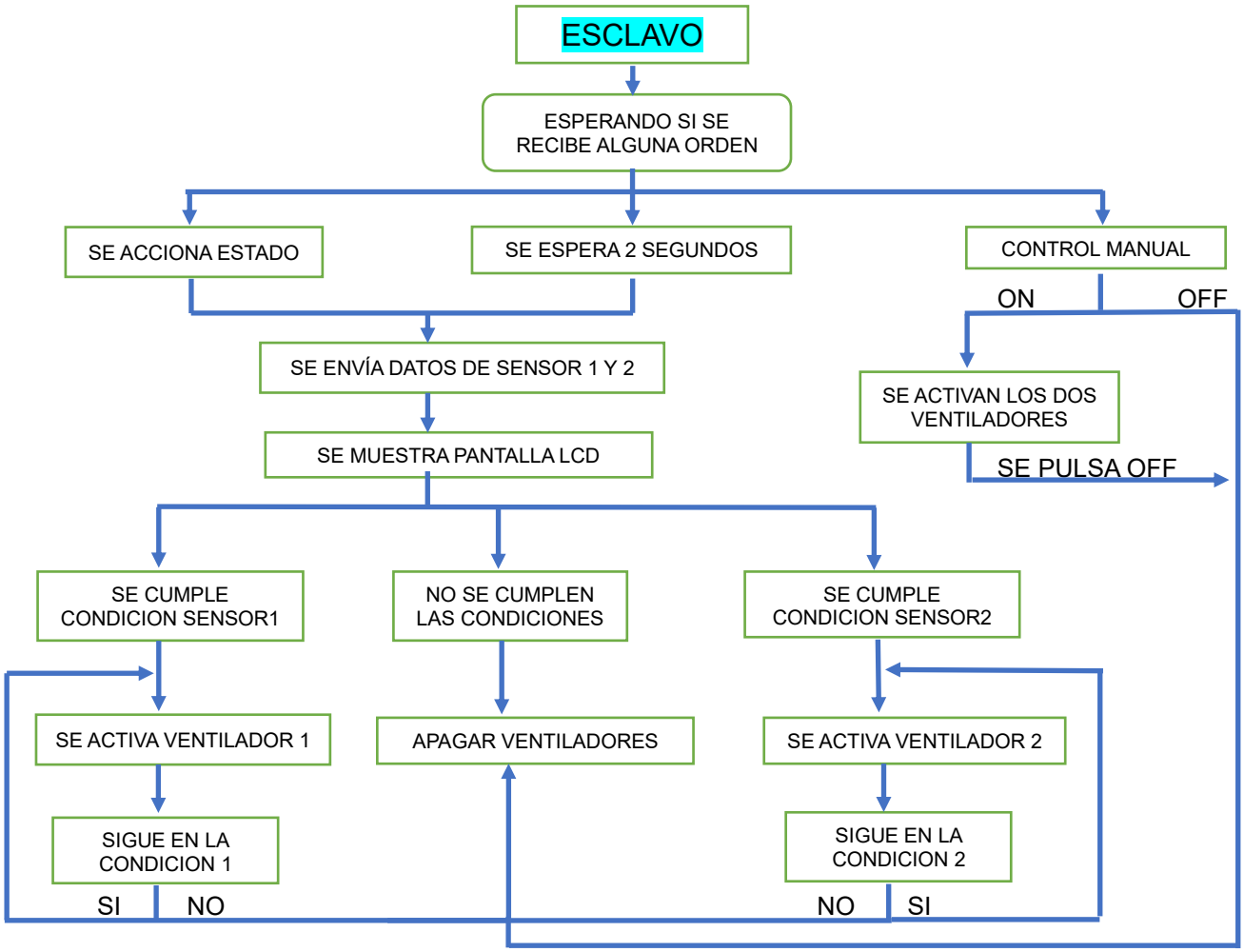
El diagrama de flujo de Maestro (Figura V.1) en todo momento está recibiendo tres datos:

En un principio el valor de fecha y hora, que, si no hay fallo de corriente, está en todo momento enviándole esa información y mostrándola en la Lcd.

En segundo lugar, cada 5 segundos recibe información de la temperatura y humedad dentro del habitáculo, a la vez pide información de los sensores, donde si cumple las condiciones programadas, sigue con el modo automático, pero si no las cumple ordena al Esclavo a realizar la función programada.

En tercer lugar, si el cliente acciona el botón los datos de la temperatura y humedad se realiza más rápido el mismo proceso que el segundo, en un segundo. También cabe la posibilidad que el cliente quiera accionar el ver la temperatura de los sensores enviándole al submenú como se explica en el diagrama de flujos.

Ninguno de los tres procesos se llevará a cabo si está el modo manual activado, donde no importa donde esté el proceso que si se activa deja de hacer todo y manda información al Esclavo de activar los ventiladores.



FiguraVI.2 Diagrama de flujo del Esclavo

El funcionamiento del Esclavo es más sencillo, ya que estará continuamente esperando si el Maestro le ha enviado un mensaje. Si el mensaje recibido “se acciona estado”, procederá a enviarle el valor de ambos sensores de temperatura, en cambio, si no recibe mensaje el Esclavo espera dos segundos y se los envía igualmente.

Por otra parte, se activa el control manual ON activa los ventiladores automáticamente y permanecen encendidos hasta que se acciones OFF, en donde si hay alguna condición activada seguirán funcionando y si no se desactivarán.

Capítulo VI: Resultados experimentales

Este proyecto en un principio se originó en unas placas Protoboard, con unos pulsadores propios para el trabajo en esta placa de pruebas, con una fuente externa de 12v para su alimentación y demás componentes que según se iba perfeccionando el prototipo se fueron intercambiando.

Lo que empezó en una caja de cartón y un cableado extenso, se fue perfeccionando, llegando a crear un prototipo de buena estética y sin esa infinidad de cables, es decir, se creó unas placas en PCB como las comentamos en el anexo 3, además de una creación de bases y soportes para que no se vea su interior, pero a su vez de fácil acceso, por si se presenta la falla de algún componente.

Como se comenta durante toda la memoria, este prototipo consta de dos módulos. En un primer momento el Maestro (Figura VI.1) y en segundo lugar el Esclavo (Figura VI.2).

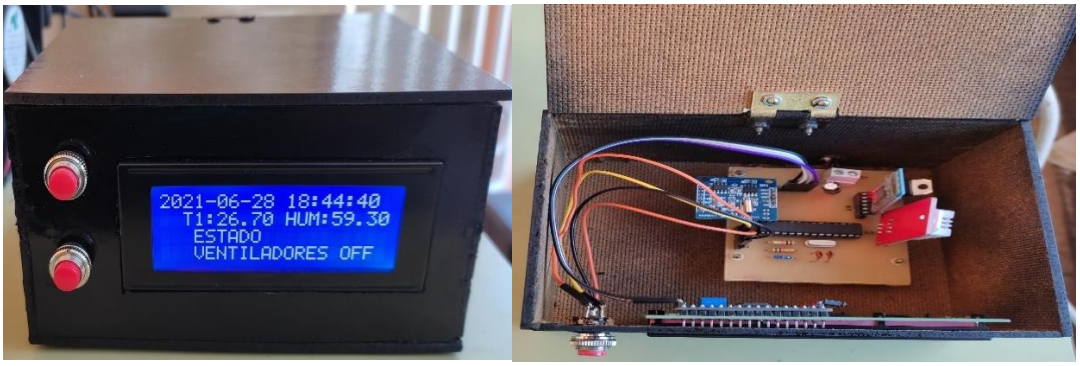


Figura VI.1 Prototipo Maestro

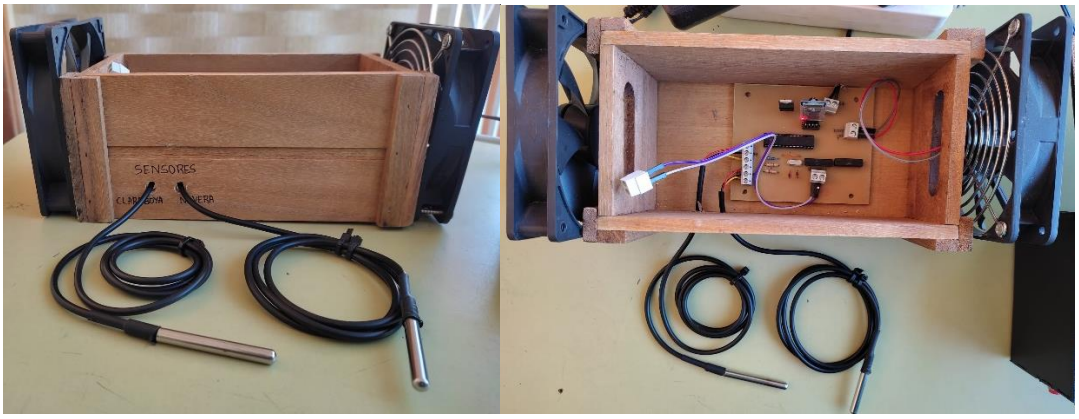


Figura VI.2 Prototipo Esclavo

El prototipo se ha estado testeando durante cuatro meses, es decir, durante la creación de este proyecto, y no han aparecido fallos. Se ha comprobado que es capaz de controlar la temperatura siendo capaz de hacer actuar los ventiladores siempre y cuando las condiciones se requieran, e incluso cabe aportar, que con la creación de estas placas en PCBs aumentó la velocidad de transferencia al estar todo mejor comunicado.

Capítulo VII: Presupuesto

El desarrollo del presente trabajo ha llevado consigo una serie de gastos, los que se van a diferenciar en gastos materiales y gastos de mano de obra como se muestran a continuación:

DESCRIPCIÓN	CANTIDAD	COSTE UNITARIO (€ / UNIDAD)	COSTE TOTAL (€)
GASTOS MATERIALES			
Placa PCB maestro y esclavo	2	7	14
Módulo Bluetooth HC-06	1	1,93	1,93
Módulo Bluetooth HC-05	1	2,62	2,62
Sensor Temperatura y humedad DHT22	1	2,62	2,62
Relé SIP-1A05	2	1,68	3,36
Sensor Encapsulado Ds18b20	2	1,63	3,26
Integrado ATMega328P-Pu	2	2,30	4,60
Reloj CRC 1307	1	1,18	1,18
Conversor LM7805	2	0,25	0,50
Resistencias de 10KOhmios	4	0,20	0,80
Resistencias de 4,7KOhmios	2	0,20	0,40
Pulsadores de dos pines	2	1,30	2,60
Pantalla LCD 20*4	1	4,32	4,32
Condensadores 47uF	2	0,77	1,54
Condensadores 22pF	4	0,05	0,20
Cristal 16Mhz	2	0,30	0,60
Ventiladores 12V	2	6,07	12,14
Bornas de dos entradas	7	1,17	8,19
Cableado de batería a ubicación prototipo (30m)	2	8	16
COSTE TOTAL			80,86 €

Tabla.VII.1 Gastos Materiales

DESCRIPCIÓN	HORAS	COSTE UNITARIO (€ / UNIDAD)	COSTE TOTAL (€)
GASTOS MANO DE OBRA			
Tiempo de programación	90	15	1350
Tiempo de implementación	240	15	3600
Tiempo de documentación	30	15	450
COSTE TOTAL			5400€

Tabla VII.2 Gastos manos de obra

GASTOS DE MATERIALES + MANO DE OBRA	TOTAL	5480,86€

Tabla VII.3 Gastos Totales.

Aportaciones y conclusiones

Conclusions

The objective of this project has been the design and implementation of an electronic device for cooling the electronic components that control the operation of a refrigerator, with the following properties:

- The designed device has been divided into two blocks: Master and Slave. This greatly facilitates its installation in motorhomes.
- The connection between the Master and the Slave has been made through a Bluetooth link, which simplifies the installation of the system because it is not necessary to use wires.
- ATMega328P-Pu microcontrollers have been used for their low price and high computing power.
- The control of the device by the user is done through a friendly interface installed in a microcontroller, designed for this project.
- The prototype is powered by the motorhome's own battery, and an external voltage source is not necessary.
- It is cheap, simple and easy to build.

Conclusiones

El objetivo de este proyecto ha sido el diseño e implementación de un dispositivo electrónico para el enfriamiento de los componentes electrónicos que controlan el funcionamiento de un frigorífico, con las siguientes propiedades:

- El dispositivo diseñado se ha dividido en dos bloques: Maestro y Esclavo. Esto facilita bastante su instalación en las autocaravanas.
- La conexión entre el Maestro y el Esclavo se ha realizado mediante un enlace Bluetooth, lo que simplifica la instalación del sistema porque no es necesario usar cables.
- Se ha hecho uso de microcontroladores ATMega328P por su bajo precio y gran potencia de cálculo.
- El control del dispositivo por parte del usuario, se realiza a través de una interfaz amigable instalada en un microcontrolador, diseñada para este proyecto.
- La alimentación en corriente del prototipo se ha realizado mediante la batería de la propia autocaravana, no siendo necesaria una fuente de tensión externa.
- Es de bajo costo, sencillo y fácil de construir.

BIBLIOGRAFÍA

ATMega328P-Pu

[1] https://www.componentsinfo.com/atmega328p-pinout-configuration-datasheet/
(disponible a fecha 25/06/2021)

Arduino Uno

[2] https://store.arduino.cc/arduino-uno-rev3 (disponible a fecha 25/06/2021)

*Pantalla LCD 20*4*

[3] https://www.luisllamas.es/arduino-lcd-i2c/ (disponible a fecha 25/06/2021)

Pulsadores

[4] https://programarfacil.com/blog/utilizar-pulsadores-en-arduino/ (disponible a fecha 25/06/2021)

Reloj

[5] https://www.geekfactory.mx/tutoriales/tutoriales-arduino/ds1307-en-tinyrtc-con-arduino/

(disponible a fecha 25/06/2021)

Sensor DHT22

[6] https://naylampmechatronics.com/blog/40_tutorial-sensor-de-temperatura-y-humedad-dht11-y-dht22.html
(disponible a fecha 25/06/2021)

Sensor Ds18b20

[7] https://programarfacil.com/blog/arduino-blog/ds18b20-sensor-temperatura-arduino/
(disponible a fecha 25/06/2021)

Relé

[8] https://www.alldatasheet.com/datasheet-pdf/pdf/1150461/ETC2/SIP-1A05.html
(disponible a fecha 25/06/2021)

Módulos de bluetooth

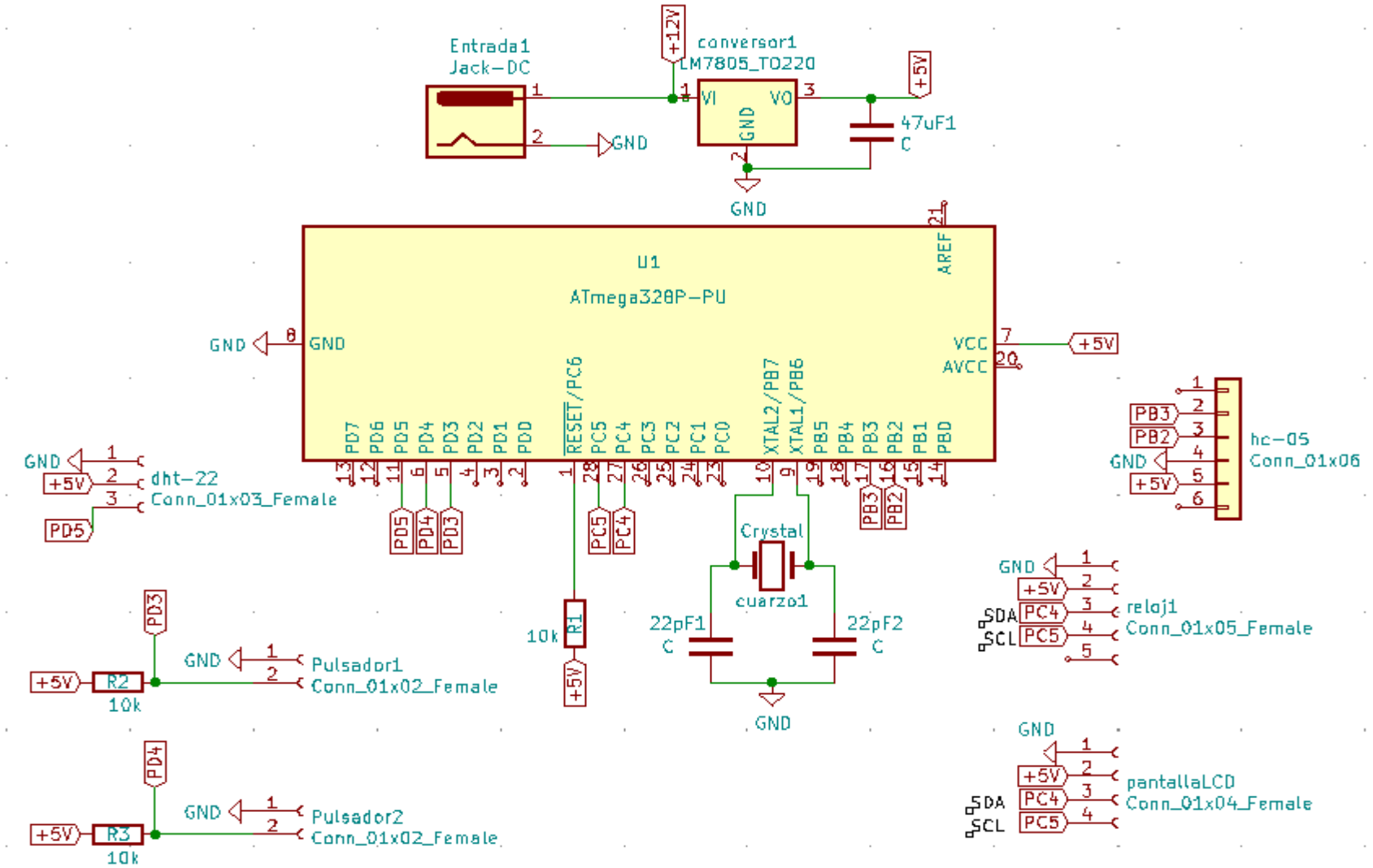
[9] https://www.luisllamas.es/conectar-arduino-por-bluetooth-con-los-modulos-hc-05-o-hc-06/
(disponible a fecha 25/06/2021)

Glosario

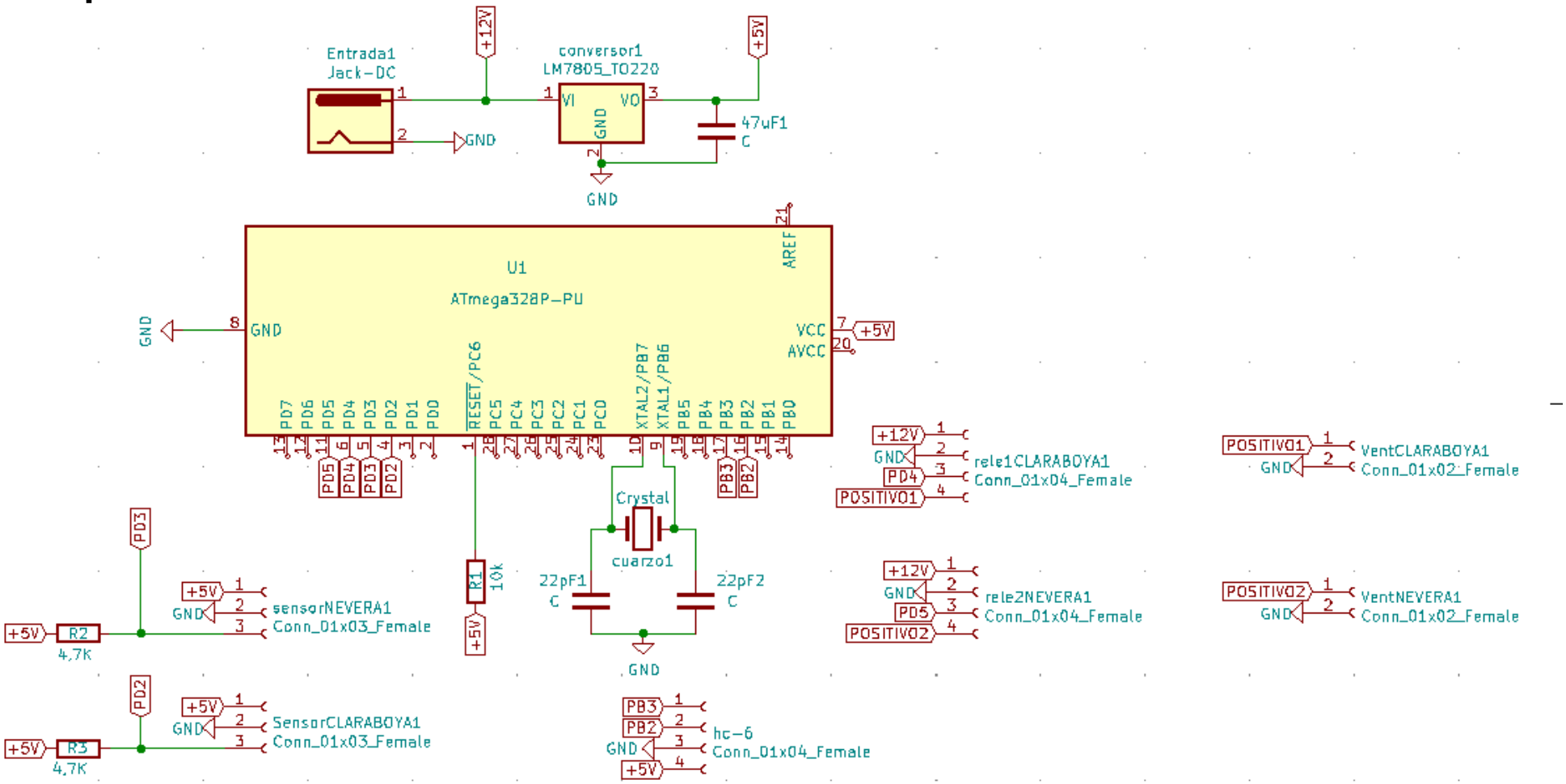
AM:	AMPLITUD MODULAR
ASK:	MODULACION POR EL DESPLAZAMIENTO DE AMPLITUD
DC:	CORRIENTE CONTINUA
MIPS:	MILLONES DE INSTRUCCIONES POR SEGUNDO
MHz:	MEGAHERCIOS
SRAM:	(STATIC RANDOM ACCESS MEMORY) MEMORIA ESTÁTICA DE ACCESO ALEATORIO
CPU:	(CENTRAL PROCESSING UNIT) UNIDAD CENTRAL DE PROCESAMIENTO
E/S:	ENTRADAS Y SALIDAS
ADC	CONVERSOR ANÁLOGICO DIGITAL
EEPROM	(ELECTRICALLY ERASABLE PROGRAMMABLE READ-ONLY MEMORY) ROM PROGRAMABLE Y BORRABLE ELÉCTRICAMENTE
PWM	(PULSE WIDTH MODULATION) MODULACIÓN DE ANCHO DE PULSO
VCC	VOLTAJE DE ALIMENTACIÓN DEL CIRCUITO
GND	(GROUND) TIERRA, NODO DE REFERENCIA QUE SE ASUME QUE TIENE 0V
I2C:	(INTER INTEGRATED CIRCUIT) PERMITE CONEXIÓN SERIE DE VARIOS DISPOSITIVOS
LCD	(LIQUID CRISTAL DISPLAY) REPRESENTACIÓN VISUAL POR CRISTAL LIQUIDO
SDA	(SYSTEM DATA) LINEA POR LA QUE SE MUEVEN LOS DATOS ENTRE LOS DISPOSITIVOS
SCL	(SYSTEM CLOCK) LINEA DE LOS PULSOS DE RELOJ QUE SINCRONIZAN EL SISTEMA
VDD	TENSIÓN POSITIVA DE ALIMENTACIÓN
RPM	REVOLUCIONES POR MINUTO
CFM	PIES CÚBICOS POR MINUTO. INDICA LA TASA DE FLUJO DE AIRE DEL COMPRESOR
PSWD	(PASSWORD) CONTRASEÑA
IDE	(INTEGRATED DEVELOPMENT ENVIRONMENT) ES DONDE SE REALIZA LA PROGRAMACIÓN DE LAS PLACAS ARDUINO

Anexos

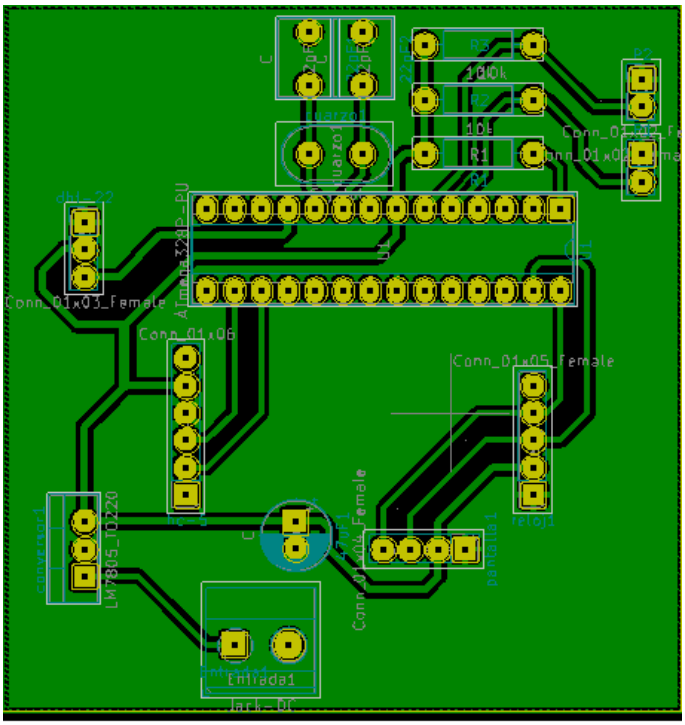
Esquema del Maestro



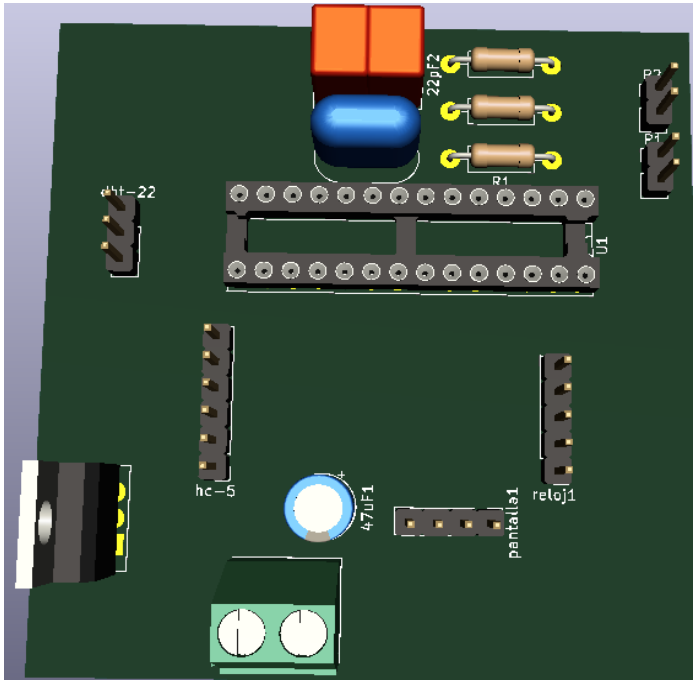
Esquema del Esclavo.



PCB y Fotolitos.

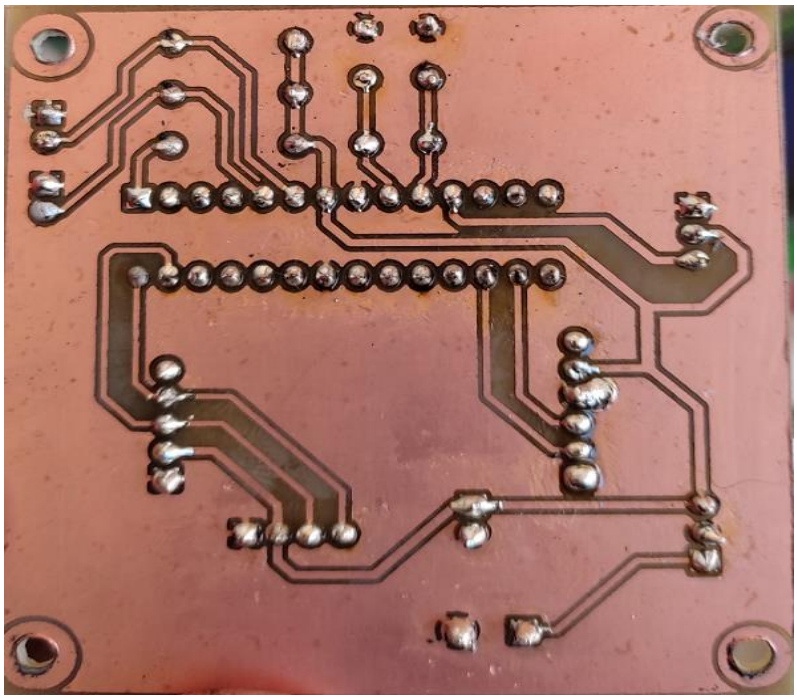


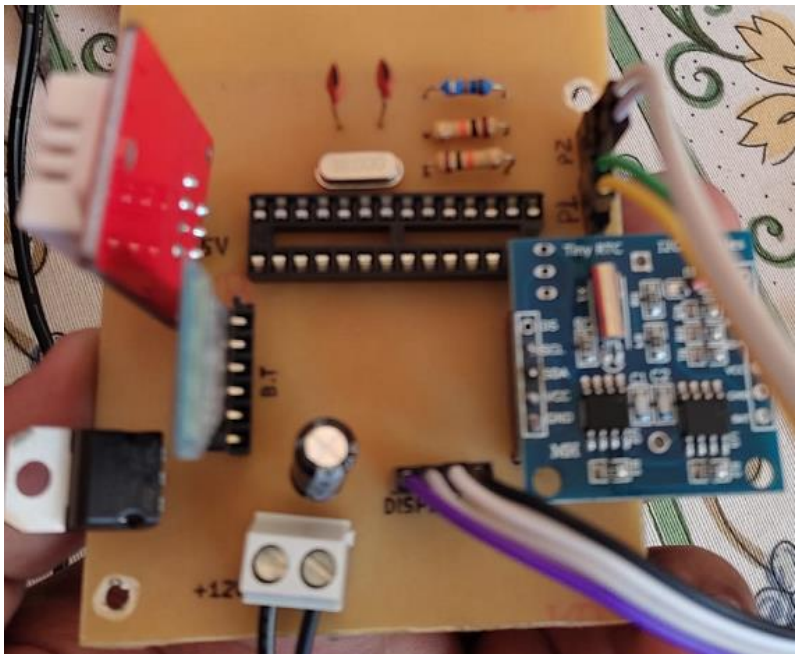
Circuito PCB del Maestro con el programa Kicad.

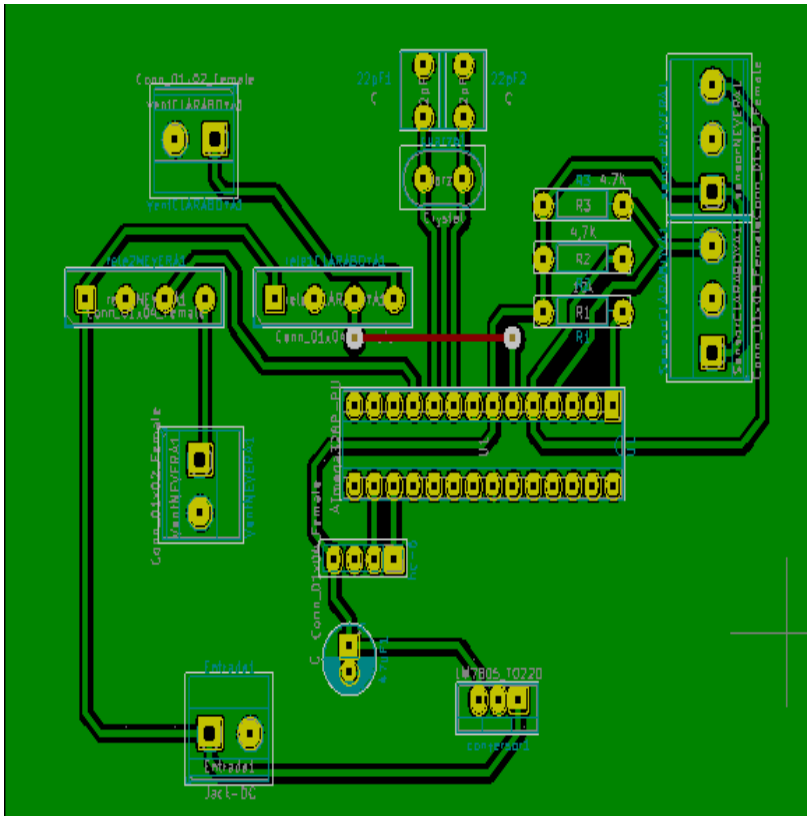


Circuito en 3D del Maestro con el programa Kicad

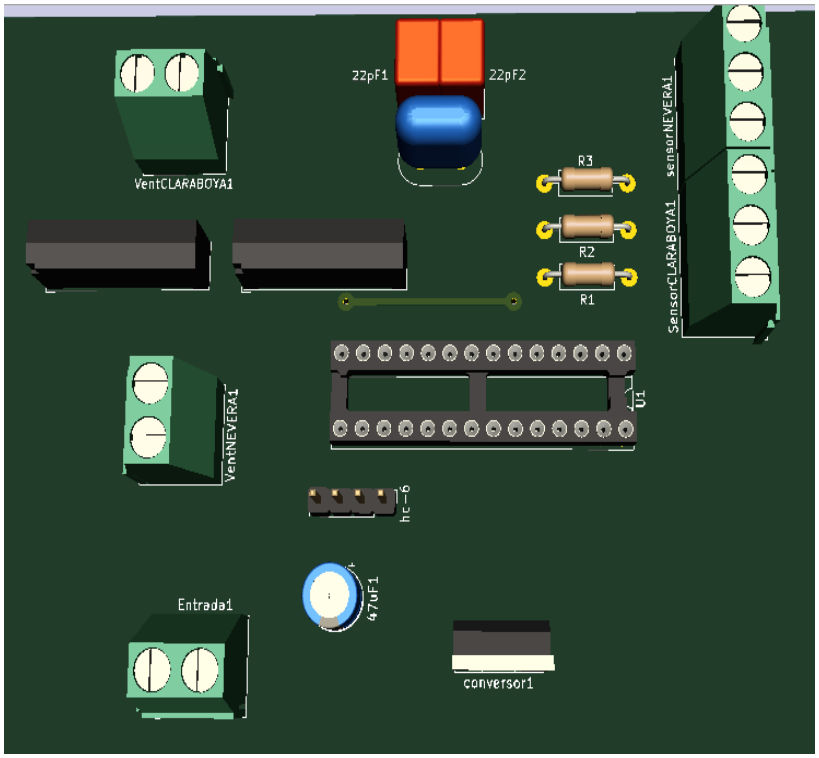
Imágenes reales Maestro





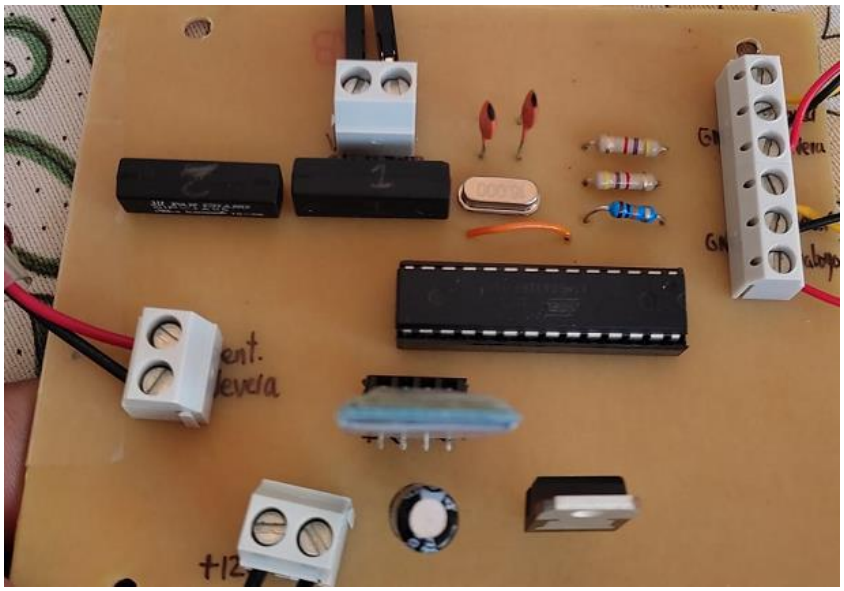


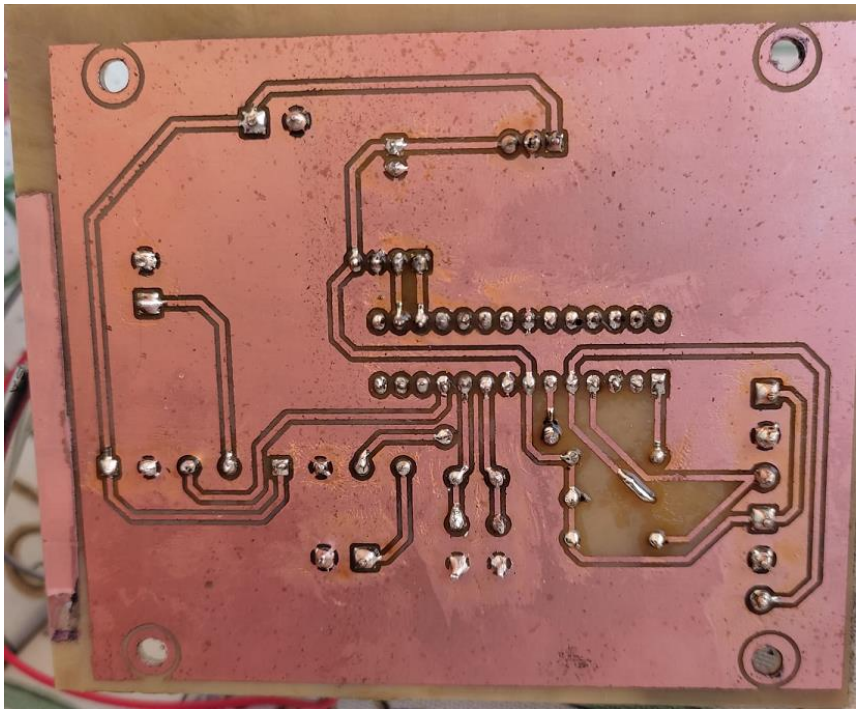
Circuito PCB del Esclavo con el programa Kicad.



Circuito en 3D del Esclavo con el programa Kicad

Imágenes reales Esclavo





Datasheets.

DS18B20 **Programmable Resolution 1-Wire Digital Thermometer**

General Description

The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems.

Applications

- Thermostatic Controls
- Industrial Systems
- Consumer Products
- Thermometers
- Thermally Sensitive Systems

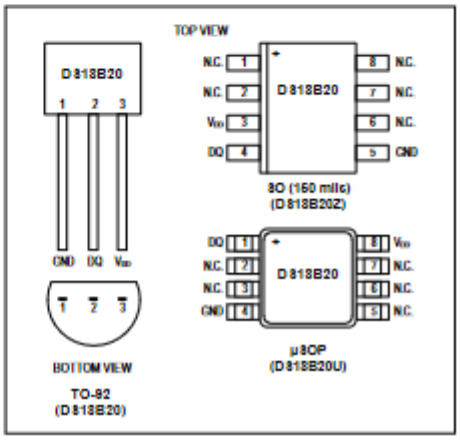
Ordering Information appears at end of data sheet.

1-Wire is a registered trademark of Maxim Integrated Products, Inc.

Benefits and Features

- Unique 1-Wire® Interface Requires Only One Port Pin for Communication
- Reduce Component Count with Integrated Temperature Sensor and EEPROM
  - Measures Temperatures from -55°C to +125°C (-67°F to +257°F)
  - ±0.5°C Accuracy from -10°C to +85°C
  - Programmable Resolution from 9 Bits to 12 Bits
  - No External Components Required
- Parasitic Power Mode Requires Only 2 Pins for Operation (DQ and GND)
- Simplifies Distributed Temperature-Sensing Applications with Multidrop Capability
  - Each Device Has a Unique 64-Bit Serial Code Stored in On-Board ROM
- Flexible User-Definable Nonvolatile (NV) Alarm Settings with Alarm Search Command Identifies Devices with Temperatures Outside Programmed Limits
- Available in 8-Pin SO (150 mils), 8-Pin µSOP, and 3-Pin TO-92 Packages

Pin Configurations



Absolute Maximum Ratings

Voltage Range on Any Pin Relative to Ground.....-0.5V to +6.0V
Operating Temperature Range..... -55°C to +125°C
Storage Temperature Range..... -55°C to +125°C
Solder Temperature.....Refer to the IPC/JEDEC J-STD-020 Specification.

These are stress ratings only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC Electrical Characteristics

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Supply Voltage	V_{DD}	Local power (Note 1)		+3.0		+5.5	V
Pullup Supply Voltage	V_{PU}	Parasite power	(Notes 1, 2)	+3.0		+5.5	V
		Local power		+3.0		V_{DD}	
Thermometer Error	t_{ERR}	-10°C to +85°C	(Note 3)			±0.5	°C
		-30°C to +100°C				±1	
		-55°C to +125°C				±2	
Input Logic-Low	V_{IL}	(Notes 1, 4, 5)		-0.3		+0.8	V
Input Logic-High	V_{IH}	Local power	(Notes 1,6)	+2.2		The lower of 5.5 or V_{DD} + 0.3	V
		Parasite power		+3.0			
Sink Current	I_L	$V_{I/O}$ = 0.4V		4.0			mA
Standby Current	I_{DDS}	(Notes 7, 8)			750	1000	nA
Active Current	I_{DD}	V_{DD} = 5V (Note 9)			1	1.5	mA
DQ Input Current	I_{DQ}	(Note 10)			5		µA
Drift		(Note 11)			±0.2		°C

Note 1: All voltages are referenced to ground.
Note 2: The Pullup Supply Voltage specification assumes that the pullup device is ideal, and therefore the high level of the pullup is equal to V_{PU}. In order to meet the V_{IH} spec of the DS18B20, the actual supply rail for the strong pullup transistor must include margin for the voltage drop across the transistor when it is turned on; thus: $V_{PU\_ACTUAL} = V_{PU\_IDEAL} + V_{TRANSISTOR}$.
Note 3: See typical performance curve in Figure 1. Thermometer Error limits are 3-sigma values.
Note 4: Logic-low voltages are specified at a sink current of 4mA.
Note 5: To guarantee a presence pulse under low voltage parasite power conditions, V_{ILMAX} may have to be reduced to as low as 0.5V.
Note 6: Logic-high voltages are specified at a source current of 1mA.
Note 7: Standby current specified up to +70°C. Standby current typically is 3µA at +125°C.
Note 8: To minimize I_{DDS}, DQ should be within the following ranges: GND ≤ DQ ≤ GND + 0.3V or V_{DD} – 0.3V ≤ DQ ≤ V_{DD}.
Note 9: Active current refers to supply current during active temperature conversions or EEPROM writes.
Note 10: DQ line is high ("high-Z" state).
Note 11: Drift data is based on a 1000-hour stress test at +125°C with V_{DD} = 5.5V.

DS18B20 Programmable Resolution 1-Wire Digital Thermometer

AC Electrical Characteristics–NV Memory

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

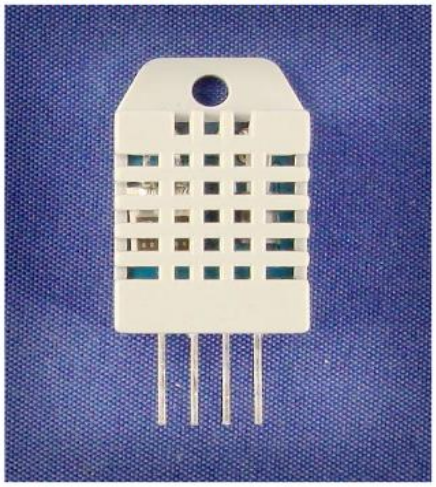
PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
NV Write Cycle Time	t_{WR}			2	10	ms
EEPROM Writes	N_{EEWR}	-55°C to +55°C	50k			writes
EEPROM Data Retention	t_{EEDR}	-55°C to +55°C	10			years

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

Digital-output relative humidity & temperature sensor/module

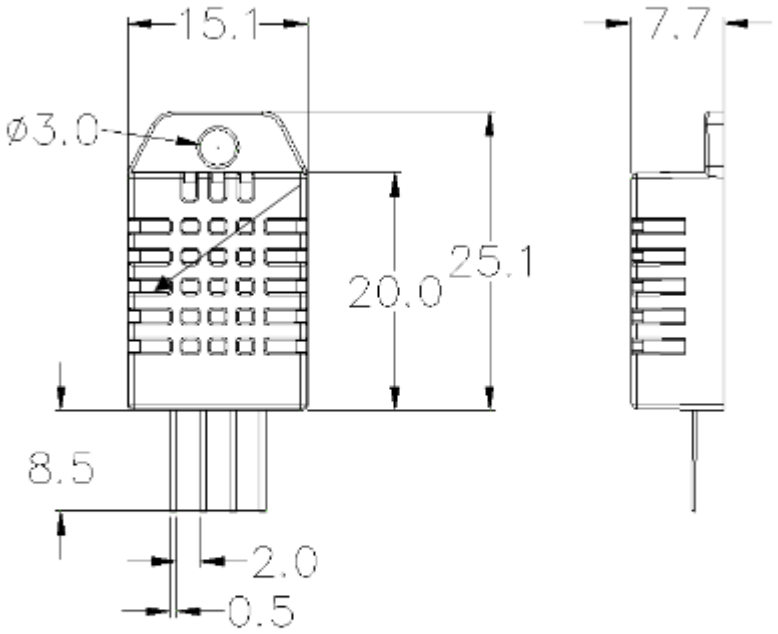
DHT22 (DHT22 also named as AM2302)



Capacitive-type humidity and temperature module/sensor

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors



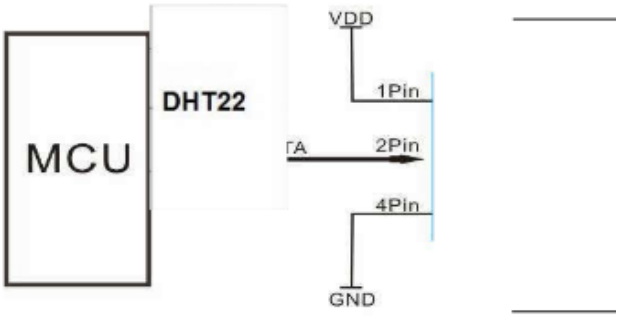
Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD----power supply
2	DATA--signal
3	NULL
4	GND

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

5. Electrical connection diagram:



3Pin---NC, AM2302 is another name for DHT22

6. Operating specifications:

(1) Power and Pins

Power's voltage should be 3.3-6V DC. When power is supplied to sensor, don't send any instruction to the sensor within one second to pass unstable status. One capacitor valued 100nF can be added between VDD and GND for wave filtering.

粤联合科技 PanChang

REED RELAYS

PART NUMBER 产品型号

SIP - XX XX XX

Producrs Name 系列名称
Contact Form 触点形式
Options 选项
Nominal Voltage 额定电压

Picture	Part Number	Schematic Contact Form (Bottom View)	Nominal Voltage (VDC)	Coil Resistance (ohms±10%)	Nominal Input Power (mW)	Must Release Voltage (VDC)	Must Operate Voltage (VDC)	Maximun Voltage (VCD)
		1Form A						
	SIP-1A05		5	500	50	3.75	0.6	15.0
	SIP-1A12		12	1000	144	8.60	1.5	30.0
	SIP-1A24	1 +3 -5 7	24	2000	288	17.50	2.5	40.0

Options:

Nil: Std Type

B: Diode

S: Magnetic Shield

BS: Diode and Magnetic Shield

Features:

- Epoxy molded,single-in-line package.
- Can be immersed during board cleaning operations
- Hing density board mounting.
- Hing isolation between input and output
- Diode and Magnetic shield are available.

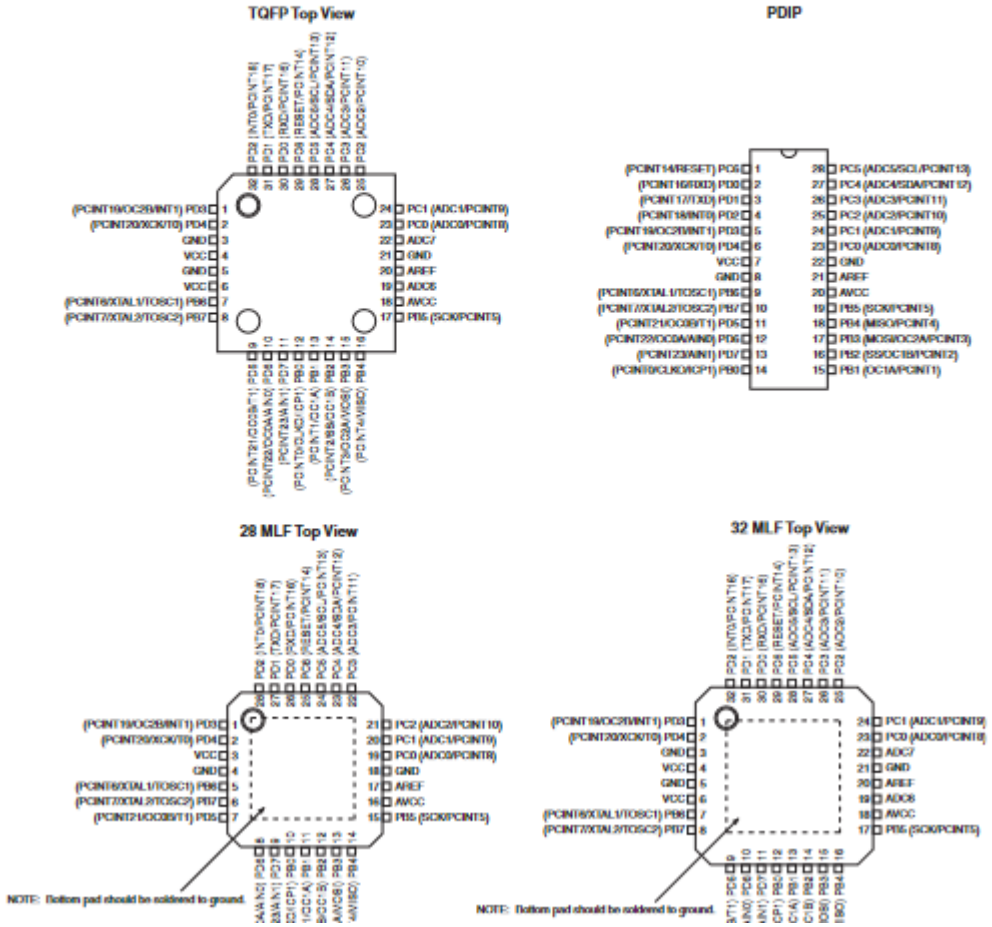
REED RELAYS

Single-In-Line Packages

Contact From	触点形式	1A
Contact Rating	触点额定值	
Maximum switching power	最大切换功率	10VA (W)
Maximum switching voltage	最大切换电压	100VDC or peak AC
Maximum switching current	最大切换电流	0.50A
Maximum carry current	最大载流	1.00A
Contact Resistance(Initial)	接触电阻	150 milliohms Max.
Life Expectancy Signal level load (Ref,12VDC,10mA)	寿命	$200X10^6$ Operations
Timing(at nominal VDC, 10HZ drive,50% duty cycle with diode suppression)		
Operate time,maximum (including Bounce)	动作时间	0.5ms
Release time,maximum	释放时间	0.5ms
Breakdown Voltage	击穿电压	
Coil to contact	线圈与触点之间	1400VDC(1000Vrms)
Across contact	触点间	250VDC(175Vrms)
Insulation Resistance	绝缘电阻	10^8 OHMS
Capacitance		
Across open contacts	开触点间	1.0 pf Max.
Open contact to coil	开触点与线圈间	2.0 pf Max.
Environmental		
Temperature	温度	
Total internal relay (storage)		-40°C to+105°C

1. Pin Configurations

Figure 1-1. Pinout ATmega48A/48PA/88A/88PA/168A/168PA/328/328P



TIGER ELECTRONIC CO.,LTD

Product specification

3-Terminal 1A Positive Voltage Regulator

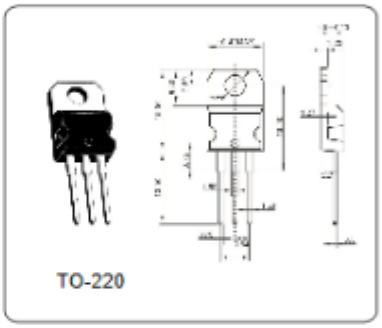
LM7805

GENERAL DESCRIPTION

The LM7805 series of three terminal positive regulators are available in the TO-220 package and with several fixed output voltages, making them useful in a wide range of applications. Each type employs internal current limiting, thermal shut down and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.

ABSOLUTE MAXIMUM RATINGS (Ta = 25 °C)

Parameter	Symbol	Typ	Unit
Input Voltage	V_I	35	V
Output Voltage	V_O	5.0	V
Peak Current	I_{PK}	2.2	A
Operating Temperature Range	T_{OPR}	0~125	°C
Storage Temperature Range	T_{STG}	-65~150	°C



TO-220

ELECTRICAL CHARACTERISTICS (Ta = 25 °C)

(Refer to test circuit, Io = 500mA, Vi = 10V, Ci= 0.33uF, Co=0.1uF unless otherwise specified)

Parameter	Symbol	Test Conditions	Min	Typ	Max	Unit
Output Voltage	V_O	V_I = 8V to 20V	4.85	5.0	5.15	V
Line Regulation (Note1)	Regline	V_O = 8V to 25V		4.0	100	mV
		V_I = 8V to 12V		1.6	50	
Load Regulation (Note1)	Regload	I_O = 5.0mA to1.5A		9	100	mV
		I_O =250mA to 750mA		4	50	
Quiescent Current	I_Q	T_J =+25 °C		5	8	mA
Ripple Rejection	RR	f = 120Hz, V_O = 8V to 18V	62	73		dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		2		V
Output Resistance	r_O	f = 1KHz		0.015		Ω
Short Circuit Current	I_{SC}	V_I = 35V, T_A =+25 °C		230		mA

Código Maestro

```
//maestro
#include "RTClib.h"
#include <OneWire.h>
#include <SoftwareSerial.h>
#include <GFButton.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>
#define DEBUG(a) Serial.println(a);
// Definimos el pin digital donde se conecta el sensor
#define DHTPIN 5
// Dependiendo del tipo de sensor
#define DHTTYPE DHT22
// Inicializamos el sensor DHT22
#define rxPin 10//receptor puerto serie virtual
#define txPin 11//transmisor puerto serie virtual
#define baudrate 9600//velocidad comunicacion
//variables para introducir manualmente la fecha y hora
int dia=0;
int mes=0;
int anio=2021;
int hora=0;
int minuto=0;
int segundo=0;
int fin=0;
int confirmar=0;
//instanciamos el objeto hc05 del puerto serie virtual
SoftwareSerial hc05(rxPin ,txPin);
//Instanciamos el objeto dht para recoger la temperatura y humedad del sensor
DHT dht(DHTPIN, DHTTYPE);
//instanciamos los botones button1 y button2
GFButton button1(3);
GFButton button2(4);
uint8_t fil;
uint8_t n;
String estado="OFF";
float t,h,t2,t3;
String cadena;
float control_remoto;
//instanciamos el objeto lcd para manejar el display
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);
//LiquidCrystal_I2C lcd(0x27,20,4);  // set the LCD address to 0x27 for a 16 chars and 2 line display
// Instanciamos el objeto rtc del módulo reloj DS3231
RTC_DS3231 rtc;
//DS3231 rtc;

void setup() {
  //asignamos los punes del puertoserie virtual
 pinMode(rxPin,INPUT);
 pinMode(txPin,OUTPUT);
  // put your setup code here, to run once:
  t2=0;
  t3=0;
//iniciamos el puerto serie virtual hc05
hc05.begin(baudrate);
Serial.begin(9600);
//inciamos el display lcd
lcd.begin(20,4);
// Print a message to the LCD.
```

```
//ilumina la parte trasera del lcd
lcd.backlight();
// limpia la pantalla
lcd.clear();
//iniciamos el sensor de temperatura y humedad
dht.begin();
fil=1;
//iniciamos el reloj
rtc.begin();
//Se establece el año
// rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
//rtc.adjust(DateTime(F(_DATE), F(TIME_)));

// rtc.adjust(DateTime(_DATE,TIME_));
//iniciamos la comunicacion I2C
Wire.begin();
//Introducimos hora y fecha de forma manual
ajuste_manual_dia();
//pasamos la fecha y hora introducidas manualmente al reloj
rtc.adjust(DateTime(anio,mes, dia,hora,minuto,segundo));
}

void loop() {
  //llamamos al modulo para visualizar la fecha y hora
visualizar_hora();
 //llamamos al módulo para leer la temperatura y humedad del sensor
leer_dht22();
//manda los datos de temperatura y humedad al esclavo
mandar_dht22();
//llama al menu
menu();

if(button1.isPressed())
{delay(70);
  cambiar_opcion();
}

if(button2.isPressed())
{ delay(70);
  switch(fil)
 {

  case 2:
  // recibe por puerto serie los datos de temperaturas encapsulado
  recibir_temperaturas_encapsulado();
 break;

  case 3:if(estado=="OFF")
  estado="ON";
  else
  estado="OFF";
  lcd.setCursor(16,3);
  lcd.print("   ");
  lcd.setCursor(16,3);
  lcd.print(estado);
  //envia la orden al esclavo para encender o apagar los ventiladores manualmente
  encender_apagar_ventiladores();
  break;
   }

}
}
```

```
//modulo del menu del programa
void menu()
{//recibir_fecha_hora();
  //lcd.setCursor(2,0);
  //lcd.print("HORA:");
  //lcd.setCursor(8,0);
  //lcd.print("12:14:15");
  lcd.setCursor(2,1);
  lcd.print("T1:");
  lcd.setCursor(5,1);
  lcd.print(t);
  lcd.setCursor(11,1);
  lcd.print("HUM:");
   lcd.setCursor(15,1);
  lcd.print(h);
  lcd.setCursor(3,2);
  lcd.print("ESTADO           ");
  lcd.setCursor(3,3);
  lcd.print("VENTILADORES");
  lcd.setCursor(16,3);
  lcd.print(estado);
}

//modulo para cambiar la opción del menú
 void cambiar_opción()
{delay(70);

   fil++;
if(fil==2)
{fil=2;
lcd.setCursor(2,fil);
lcd.print("<");
lcd.setCursor(2,fil+1);
lcd.print(" ");
lcd.setCursor(19,fil);
lcd.print(" ");
lcd.setCursor(19,fil+1);
lcd.print(" ");

}
  if(fil==4)
{fil=2;
lcd.setCursor(2,fil);
lcd.print("<");
lcd.setCursor(2,fil+1);
lcd.print(" ");
lcd.setCursor(19,fil);
lcd.print(" ");
lcd.setCursor(19,fil+1);
lcd.print(" ");

}
if(fil==3)
{
lcd.setCursor(2,fil-1);
lcd.print(" ");
lcd.setCursor(2,fil);
lcd.print("<");
lcd.setCursor(19,fil-1);
lcd.print(" ");
lcd.setCursor(19,fil);
lcd.print(" ");
```

```
}
}
//modulo para leer la temperatura y humedad del sensor
void leer_dht22()
{ h = dht.readHumidity();
  t = dht.readTemperature();
}
//modulo para enviar la orden al esclavo para encender y apagar los ventiladores
void encender_apagar_ventiladores()
{

if (estado=="ON")
{control_remoto=-50;
hc05.print(control_remoto);
hc05.print(" ");
hc05.print('\n');
delay(70);
Serial.println("control_remoto");
Serial.println(control_remoto);
}


if (estado=="OFF")
 {control_remoto=-55;
hc05.print(control_remoto);
hc05.print(" ");
hc05.print('\n');
delay(70);
Serial.println("control_remoto");
Serial.println(control_remoto);

}
Serial.println("CONTROL REMOTO= ");
Serial.println(control_remoto);
}
//modulo para visualizar fecha y hora en el display
void visualizar_hora()
{DateTime now = rtc.now();
  //char buf[20];
  //strncpy(buf,"DD.MM.YYYY hh:mm:ss\0",20);
 // Serial.println(now.format(buf));

  lcd.setCursor(0,0);
//  lcd.print(now.format(buf));
 lcd.print(now.timestamp());
  lcd.setCursor(10,0);
  lcd.print(" ");
  }
// modulo para recibir las temperaturas de los encapsulados del esclavo
void recibir_temperaturas_encapsulado()
{
   if (hc05.available()>0)
   {
   delay(70);
   t2=hc05.parseFloat();
   delay(70);
    t3=hc05.parseFloat();
    hc05.flush();
   }
```

```
   if(t2>2&&t3>20)
   {{lcd.clear();

   lcd.setCursor(0,1);
   lcd.print("Caravana_Inteligente");
   lcd.setCursor(1,2);
   lcd.print("T_Claraboya:");
    lcd.setCursor(13,2);
   lcd.print(t2);
   lcd.setCursor(1,3);
   lcd.print("T_Nevera:");
    lcd.setCursor(10,3);
   lcd.print(t3);//CREO QUE t2
    //lcd.setCursor(3,2);
   //lcd.print(t2);
   //lcd.setCursor(8,2);
  // lcd.print(" ");
   //lcd.setCursor(12,2);
   //lcd.print(t3);
   delay(2000);
   Serial.println(t2);
   Serial.println(t3);
   lcd.clear();
   }t2=0;t3=0;
   }
}
// modulo para mandar la temperatura y humedad del sensor al esclavo
void mandar_dht22()
{
  hc05.print(t);
  hc05.print("\n");
  delay(70);
 }
//modulo para ajustar la fecha y hora manualmente con dos botones
void ajuste_manual_dia()
{ while(confirmar==0)
{fin=0;
  while(fin==0)
  {lcd.setCursor(0,1);
  lcd.print("Dia:");

  if(button1.isPressed())
  {delay(180);
    dia++;
lcd.setCursor(5,1);
  lcd.print(dia);
  if(dia==32)
  {dia=0;
  lcd.setCursor(5,1);
  lcd.print("  ");
  lcd.setCursor(5,1);
  lcd.print(dia);
  }
}
if(button2.isPressed())
{delay(180);
fin=1;}
};
fin=0;
lcd.setCursor(0,1);
lcd.print("          ");
```

```
 while(fin==0)
  {lcd.setCursor(0,1);
  lcd.print("Mes:");

  if(button1.isPressed())
  {delay(180);
    mes++;
lcd.setCursor(5,1);
  lcd.print(mes);
  if(mes==13)
  {mes=0;
  lcd.setCursor(5,1);
  lcd.print("  ");
  lcd.setCursor(5,1);
  lcd.print(mes);
  }
}
if(button2.isPressed())
{delay(180);fin=1;}
};
fin=0;
lcd.setCursor(0,1);
lcd.print("          ");

while(fin==0)
  {lcd.setCursor(0,1);
  lcd.print("Anio:");

  if(button1.isPressed())
  {delay(180);
    anio++;
lcd.setCursor(5,1);
  lcd.print(anio);
  if(anio==2025)
  {anio=2020;
  lcd.setCursor(5,1);
  lcd.print("  ");
  lcd.setCursor(5,1);
  lcd.print(anio);
  }
}
if(button2.isPressed())
{delay(180);
fin=1;}
};
fin=0;
lcd.setCursor(0,1);
lcd.print("          ");
while(fin==0)
  {lcd.setCursor(0,1);
  lcd.print("Hora:");

  if(button1.isPressed())
  {delay(180);
    hora++;
lcd.setCursor(5,1);
  lcd.print(hora);
  if(hora==25)
  {hora=0;
  lcd.setCursor(5,1);
  lcd.print("  ");
  lcd.setCursor(5,1);
```

```
lcd.print(hora);
  }
}
if(button2.isPressed())
{delay(180);fin=1;}
};
fin=0;
lcd.setCursor(0,1);
lcd.print("          ");

while(fin==0)
  {lcd.setCursor(0,1);
  lcd.print("Minuto:");

  if(button1.isPressed())
  {delay(180);
    minuto++;
lcd.setCursor(8,1);
  lcd.print(minuto);
  if(minuto==61)
  {minuto=0;
  lcd.setCursor(8,1);
  lcd.print("  ");
  lcd.setCursor(8,1);
  lcd.print(minuto);
  }
}
if(button2.isPressed())
{delay(180);fin=1;}
};
fin=0;
lcd.setCursor(0,1);
lcd.print("          ");

while(fin==0)
  {lcd.setCursor(0,1);
  lcd.print("Segundo:");

  if(button1.isPressed())
  {delay(180);
    segundo++;
lcd.setCursor(8,1);
  lcd.print(segundo);
  if(segundo==61)
  {segundo=0;
  lcd.setCursor(8,1);
  lcd.print("  ");
  lcd.setCursor(8,1);
  lcd.print(segundo);
  }
}
if(button2.isPressed())
{delay(180);fin=1;}
};
fin=0;
lcd.setCursor(0,1);
lcd.print("          ");
while(fin==0)
  {lcd.setCursor(0,1);
  lcd.print("Confirmar:");

  if(button1.isPressed())
```

```
  {delay(180);
    confirmar++;
lcd.setCursor(12,1);
  lcd.print(confirmar);
  if(confirmar==2)
  {confirmar=0;
  lcd.setCursor(12,1);
  lcd.print("  ");
  lcd.setCursor(12,1);
  lcd.print(confirmar);
  }
}
if(button2.isPressed())
{delay(180);
if(confirmar==1) fin=1;
if(confirmar==0) fin=1;
}
};
lcd.setCursor(0,1);
lcd.print("                    ");
};

}
```

Código Esclavo

```
//esclavo
#include <Wire.h>
#include <SoftwareSerial.h>
#include <OneWire.h>//sensor encapsulado
#include <DallasTemperature.h>//sensor encapsulado
const int oneWirePin1 = 2;
const int oneWirePin2 = 3;
OneWire oneWireBus1(oneWirePin1);
OneWire oneWireBus2(oneWirePin2);
DallasTemperature sensor1(&oneWireBus1);
DallasTemperature sensor2(&oneWireBus2);
#define rxPin 10//receptor puerto serie virtual
#define txPin 11//transmisor puerto serie virtual
#define rele1 4// pin rele 1
#define rele2 5//pin rele 2
#define baudrate 9600//velocidad comunicacion

SoftwareSerial hc05(rxPin ,txPin);
float t1,t2,t3;//t1 dht22, t2 claraboya , t3 nevera
float control1;//t1 dht,t2 claraboya,t3 nevera
int estado_ventilador1,estado;
int estado_ventilador2;
//  Direccionamiento del hardware


void setup() {
estado_ventilador1=-1;
estado_ventilador2=-1;
estado=-1;//automático y 1 manual
control1=0;
```

```
pinMode(rxPin,INPUT);
pinMode(txPin,OUTPUT);
pinMode(rele1,OUTPUT);
pinMode(rele2,OUTPUT);

hc05.begin(baudrate);
Serial.begin(9600);
sensor1.begin();
sensor2.begin();
 digitalWrite( rele1,LOW);
 digitalWrite( rele2,LOW);

 Wire.begin();
 }
void loop() {
 recibir_control_remoto();
 recibir_temperatura_DTH22();
 leer_temperatura1();
 leer_temperatura2();
 enviar_temperaturas();
 control();
 delay(300);
}
void activar_ventilador1()
{
  Serial.println("Activando ventilador1");

  digitalWrite( rele1,HIGH);

  estado_ventilador1=1;
}

void activar_ventilador2()
{
Serial.println("Activando ventilador2");
  digitalWrite( rele2,HIGH);
  estado_ventilador2=1;
 }

void apagar_ventilador1()
{
  Serial.println("Apagando ventilador1");

  digitalWrite( rele1,LOW);
  estado_ventilador1=-1;
}

void apagar_ventilador2()
{Serial.println("Apagando ventilador2");
  digitalWrite( rele2,LOW);
  estado_ventilador2=-1;
}

void leer_temperatura1()
{
   sensor1.requestTemperatures();
   t2=sensor1.getTempCByIndex(0);
   Serial.println("temperatura2");
   Serial.println(t2);
}
```

```
void leer_temperatura2()
{
   sensor2.requestTemperatures();
   t3=sensor2.getTempCByIndex(0);
   Serial.println("temperatura3");
   Serial.println(t3);
}

void control()
{  //claraboya
if(t1>t2&&estado_ventilador1==-1&&estado==-1&&t2!=-127){ activar_ventilador1();}
if(t1<t2&&estado_ventilador1==1&&estado==-1&&t2!=-127) {apagar_ventilador1();}
 //nevera
if(t3>55&&estado_ventilador2==-1&&estado==-1&&t3!=-127){activar_ventilador2();}
if(t3<=45&&estado_ventilador2==1&&estado==-1&&t3!=-127){ apagar_ventilador2();}
}

void recibir_temperatura_DTH22()
{float s;
  if(hc05.available())
  {
   s=hc05.parseFloat();
  hc05.flush();//limpia puerto serie virtual
  }
  if(s>10&&s<45) t1=s;
  Serial.println("DHT22");
  Serial.println(t1);
}

void recibir_control_remoto()
{float c;
c=0;
int i;
i=0;
  while(c>=0&&i<12)
{if(hc05.available()>0)
  {
   delay(90);
  c=hc05.parseFloat();
   hc05.flush();
   }
   if(c<0) control1=c;
   i++;}
   if(control1<0)
  {

if(control1>-51&&control1<0) estado=1;
if(control1<-52) estado=-1;

 if(control1>-51&&control1<0)  {activar_ventilador1();activar_ventilador2();}
 if(control1<-52)  {apagar_ventilador1();apagar_ventilador2();}
 }
  control1=0;
 }
void enviar_temperaturas()
{
 hc05.print(t2);
 hc05.print(" ");
 hc05.print(t3);
 hc05.print('\n');
delay(70);
}
```