Universidad de La Laguna

ESCUELA SUPERIOR DE INGENIERÍA Y TECNOLOGÍA

SECCIÓN DE INGENIERÍA INDUSTRIAL

TRABAJO DE FIN DE GRADO

SISTEMA DE RIEGO INTELIGENTE DE BAJO COSTE

Titulación: *Grado en Ingeniería Electrónica Industrial y Automática*

Alumna: Yaiza Tejera Fumero

Tutor: Jonay Tomás Toledo Carrillo

Co-directora: Sara González Pérez

Julio, 2018

Agradecimientos

Me gustaría agradecer de manera especial y sincera a mi tutor Jonay Toledo Carrillo su paciencia, esfuerzo y dedicación infinita. Siempre estando dispuesto a explicarme cada duda planteada, ayudándome en cada obstáculo interpuesto en el desarrollo de este proyecto. Ha sido un privilegio haber podido disfrutar de tus conocimientos. Gracias por ayudarme a crecer un poco más profesionalmente.

También agradecer a cada uno de mis familiares y amigos el apoyo diario recibido en estos años de formación. Destacando a mi padre Domingo por impulsarme y animarme en este camino, y en especial a mi madre Rosi, pues allá donde estés, lo he conseguido gracias a ti, por enseñarme a luchar para conseguir cada una de mis metas.

ESCUELA SUPERIOR DE INGENIERÍA Y TECNOLOGÍA

SECCIÓN DE INGENIERÍA INDUSTRIAL

ÍNDICE GENERAL

TRABAJO DE FIN DE GRADO

Titulación: *Grado en Ingeniería Electrónica Industrial y Automática*

TÍTULO:

SISTEMA DE RIEGO INTELIGENTE DE BAJO COSTE

AUTORA:

Yaiza Tejera Fumero

Julio, 2018

ÍNDICE GENERAL

MEMORIA

CAPÍTULO 1. INTRODUCCIÓN GENERAL. 13
CAPÍTULO 2. HERRAMIENTAS DE DISEÑO Y DESARROLLO. 17
CAPÍTULO 3. DESCRIPCIÓN DEL SISTEMA. 23
CAPÍTULO 4. IMPLEMENTACIÓN. 43
CAPÍTULO 5. RESULTADOS EXPERIMENTALES. 53
CAPÍTULO 6. PROBLEMAS Y SOLUCIONES. 56
CAPÍTULO 7. PRESUPUESTO. 60
CAPÍTULO 8. CONCLUSIONES. 62
REFERENCIAS Y BIBLIOGRAFÍA 64

ANEXOS

ANEXO I: DATASHEETS 67
ANEXO II: CÓDIGO IMPLEMENTADO. 83
ANEXO III: ESQUEMAS Y CONEXIONES 91
ANEXO IV: CROQUIS DE LAS INSTALACIONES AGRÍCOLAS. 92

ESCUELA SUPERIOR DE INGENIERÍA Y TECNOLOGÍA

SECCIÓN DE INGENIERÍA INDUSTRIAL

MEMORIA

TRABAJO DE FIN DE GRADO

Titulación: *Grado en Ingeniería Electrónica Industrial y Automática*

TÍTULO:

SISTEMA DE RIEGO INTELIGENTE DE BAJO COSTE

AUTORA

Yaiza Tejera Fumero

Julio, 2018

HOJA DE IDENTIFICACIÓN

TÍTULO DEL PROYECTO:

SISTEMA DE RIEGO INTELIGENTE DE BAJO COSTE

PETICIONARIO:

ESCUELA SUPERIOR DE INGENIERÍA Y TECNOLOGÍA
SECCIÓN DE INGENIERÍA INDUSTRIAL

DIRECCIÓN: CAMINO SAN FRANCISCO DE PAULA, S/N. CAMPUS DE ANCHIETA. SAN CRISTÓBAL DE LA LAGUNA. S/C DE TENERIFE.

AUTORA:

YAIZA TEJERA FUMERO

TUTOR:

JONAY TOMÁS TOLEDO CARRILLO

CO-DIRECTORA:

SARA GONZÁLEZ PÉREZ

ÍNDICE DE LA MEMORIA

Resumen 11
Abstract 12
CAPÍTULO 1. INTRODUCCIÓN GENERAL. 13
1.1. Objetivo. 13
1.2. Alcance. 13
1.3. Antecedentes. 13
1.4. Estructura general del proyecto. 15
CAPÍTULO 2. HERRAMIENTAS DE DISEÑO Y DESARROLLO. 17
2.1. Herramientas de hardware. 17
2.1.1. Protoboard. 17
2.1.2. Osciloscopio. 17
2.1.3. Fuente de alimentación. 18
2.1.4. Multímetro. 19
2.1.5. Generador de funciones. 19
2.1.6. Arduino UNO. 20
2.2. Herramientas de software. 21
2.2.1. Arduino IDE v1.8.5. 21
2.2.2. Fritzing. 21
2.2.3. LTSpice XVII. 22
CAPÍTULO 3. DESCRIPCIÓN DEL SISTEMA. 23
3.1. Introducción. 23
3.2. Comunicación entre placas Arduino. 23
3.2.1. Comunicación alámbrica. 24
3.2.2. Comunicación inalámbrica. 28
3.2.3. Estructura general de comunicación entre Arduinos. 31
3.3. Programación del Arduino. 32
3.3.1. Librerías. 33
3.3.2. Comandos. 33
3.3.3. Descripción del código. 35
CAPÍTULO 4. IMPLEMENTACIÓN. 43
4.1. Alámbrica. 43
4.2. Inalámbrica. 51
CAPÍTULO 5. RESULTADOS EXPERIMENTALES. 53
5.1. Pruebas de alcance de los módulos RF. 53

5.1.1. Módulos RF – 433 MHZ. 53
5.1.2. Módulos NRF2401. 53
5.2. Funcionamiento del modulador – demodulador FSK. 54
CAPÍTULO 6. PROBLEMAS Y SOLUCIONES. 56
6.1. Problemas con los módulos inalámbricos. 56
6.2. Problemas con el modulador-demodulador FSK. 57
CAPÍTULO 7. PRESUPUESTO. 60
7.1. Presupuesto del proyecto. 60
7.1.1. Coste de los materiales. 60
7.1.2. Coste de la mano de obra. 61
7.2. Comparativa económica entre los distintos sistemas de riegos. 61
CAPÍTULO 8. CONCLUSIONES. 62
REFERENCIAS Y BIBLIOGRAFÍA 64

ÍNDICE DE LAS FIGURAS.

Figura 1. Croquis de una instalación agrícola convencional. 14
Figura 2. Conexión interna de la protoboard. 17
Figura 3. Osciloscopio empleado para este proyecto. 18
Figura 4. Fuente de alimentación empleada en este proyecto. 18
Figura 5. Multímetro empleado en este proyecto. 19
Figura 6. Generador de funciones empleado en este proyecto. 19
Figura 7. Placa arduino empleada en este proyecto. 20
Figura 8. Interfaz arduino IDEA v1.8.5. 21
Figura 9. Interfaz de *software* Fritzing. 22
Figura 10. Aspecto de las señales portadora, moduladora y modulada. 24
Figura 11. Representación de la modulación FSK. 25
Figura 12. Diagrama de bloques del CI CD4046. 26
Figura 13. Diagrama de bloques de un PLL como modulador. 27
Figura 14. Diagrama de bloques de un PLL como demodulador. 27
Figura 15. Estructura general del bus SPI. 28
Figura 16. Diagrama de bloques del módulo NRF2401. 29
Figura 17. Módulo NRF2401. Versión con antena integrada. 30
Figura 18. Módulo NRF2401. Versión con antena externa y amplificador. 30
Figura 19. Diagrama de flujo del maestro. 31
Figura 20. Diagrama de flujo del esclavo. 32
Figura 21. Bloques del módulo emisor. 43
Figura 22. Frecuencia de oscilación del vco en función de su tensión de entrada. 44
Figura 23.determinación de R2 y C1 a partir de FMIN. 45
Figura 24. Determinación de R1 a partir de R2 y el coeficiente FMáX/FMIN. 46
Figura 25. Bloques del módulo receptor 48
Figura 26. Amplificador operación para la explicación de un comparador. 49
Figura 27. Pines asociados al transceptor NRF2401. 51
Figura 28. Conexión del módulo NRF2401 con el arduino maestro. 51
Figura 29. Conexión del módulo NRF2401 con el arduino esclavo. 52
Figura 30. Señales obtenidas en el osciloscopio para una comunicación unidireccional tras modular y demodular la señal arduino. 54
Figura 31. Señales obtenidas en el osciloscopio para una comunicación bidireccional tras modular y demodular la señal arduino. 55
Figura 32. Módulo emisor y receptor de RF 433 – MHz. 56
Figura 33. Conexión de los módulos RF – 433 MHz en las placas arduino. 56
Figura 34. Módulo NRF2401 con condensador soldado en sus pines de alimentación. 57
Figura 35. Conexión interna del CI SN7404N. 57
Figura 36. Señales obtenidas antes de usar el Disparador de Schmitt. 58
Figura 37. Señal original y demodulada haciendo uso del Disparador de Schmitt. 59

ÍNDICE DE LAS TABLAS

Tabla 1. Presupuesto de la parte electrónica para un sistema de riego convencional.. 15
Tabla 2. Pines asociados a SPI del Arduino UNO. 29
Tabla 3. Explicación de comandos de la librería *VirtualWire*. 34
Tabla 4. Explicación de comandos de la librería RF24..... 34
Tabla 5. Explicación de comandos de la comunicación serial. 35
Tabla 6. Esquema de conexión entre Arduino y módulo RF. 51
Tabla 7. Coste de los materiales. 60
Tabla 8. Coste de la mano de obra. 61
Tabla 9. Presupuesto de la parte electrónica para un sistema de riego inteligente. 61

Resumen

El proyecto desarrollado consiste principalmente en la evaluación y comparación de dos sistemas de comunicación para llevar a cabo el riego una instalación agrícola.

El primero de ellos es de tipo alámbrico y usa la tecnología PLC (*Power Line Communication)* basándose en modulación FSK de bajo coste. Con esta tecnología se permitirá la transmisión de datos a través del voltaje de alimentación. Por el contrario, el segundo es un sistema de comunicación inalámbrica que se basa en módulos de radiofrecuencia para establecer la comunicación a través del aire.

Finalmente, se realiza una comparación económica entre un sistema de riego convencional y el inteligente desarrollado a lo largo de este proyecto, en la que se demuestra que la adopción de este último puede conducir a un importante ahorro de costos en términos de materiales empleados, especialmente se debe destacar la diferencia entre la cantidad de cable requerido, en metros, para cada uno de los sistemas desarrollados.

Abstract

The project developed consists mainly of the evaluation and comparison between two communication systems for the irrigation network of an agricultural facility.

The first one is a wired system that uses PLC (Power Line Communication) technology based on low cost FSK modulation. This technology allows the data transmission through the supply voltage. Conversely, the second one is a wireless communication system which is based on radiofrequency modules to set up communication through the air.

Finally, an economic comparison is undertaken between a conventional irrigation system and the intelligent one developed throughout this project, which shows that the adoption of the latter may lead to a significant cost saving in terms of materials used, especially being worth to remark the difference between the amount of cable required, in meters, for each of the systems shown.

CAPÍTULO 1. INTRODUCCIÓN GENERAL.

1.1. Objetivo.

El objetivo principal de este proyecto se centra en desarrollar un sistema de riego inteligente a través de sensores y actuadores de bajo coste, añadiendo funcionalidades a un sistema de riego convencional.

Para conseguirlo se hará uso de la programación de Arduino y se estudiarán distintas formas de comunicación entre diferentes puntos de riego y la cabecera. Concretamente se comparará la tecnología PLC (*Power Line Communication),* que permite la transmisión de datos por la red eléctrica consiguiendo así reducir costes de cableado, frente a la comunicación inalámbrica.

1.2. Alcance.

El presente proyecto se centrará en el sistema de comunicación de bajo nivel, lo que se denomina capa física de transporte, y se estudiarán alternativas alámbricas e inalámbricas del mismo.

Se implementará un código simple de un único maestro – esclavo para poner a prueba su funcionamiento, donde se activará o desactivará una electroválvula para abastecer de agua la zona de la instalación agrícola según las condiciones atmosféricas del momento.

Asimismo, es de especial importancia destacar que se realizará una comparativa económica entre un sistema de riego convencional con varias electroválvulas y este sistema de riego inteligente.

Este trabajo ha sido propuesto para dos alumnos, y tiene por objetivo final alcanzar un sistema de comunicación entre múltiples esclavos, sensores y actuadores, cuya programación desarrollará el compañero Samuel Estévez Rodríguez en su proyecto basándose en las comunicaciones realizadas en este trabajo.

1.3. Antecedentes.

En un sistema de riego de una instalación agrícola dispondremos de un circuito de tuberías donde cada cierto número de metros habrá una electroválvula que abastecerá el agua necesaria al terreno. Normalmente estas electroválvulas están controladas por una pareja de cables que irán desde el controlador hasta la misma.

El problema surge cuando existe bastante separación entre las electroválvulas, ya que los cables necesitarán salir de nuevo desde la cabecera donde se encuentra el

controlador hasta cada una de ellas, sumando una cantidad de metros considerable y causando un gasto costoso en cableado.

Imaginemos la instalación agrícola de la *Figura 1*. Destacar que las dimensiones de este croquis únicamente se han usado para simplificar el cálculo del cableado, pues no son aplicables a la realidad debido a que la separación entre hileras es excesiva si se desea aprovechar bien el terreno. Pero con este ejemplo se observará claramente el gasto que supone una instalación de este estilo.

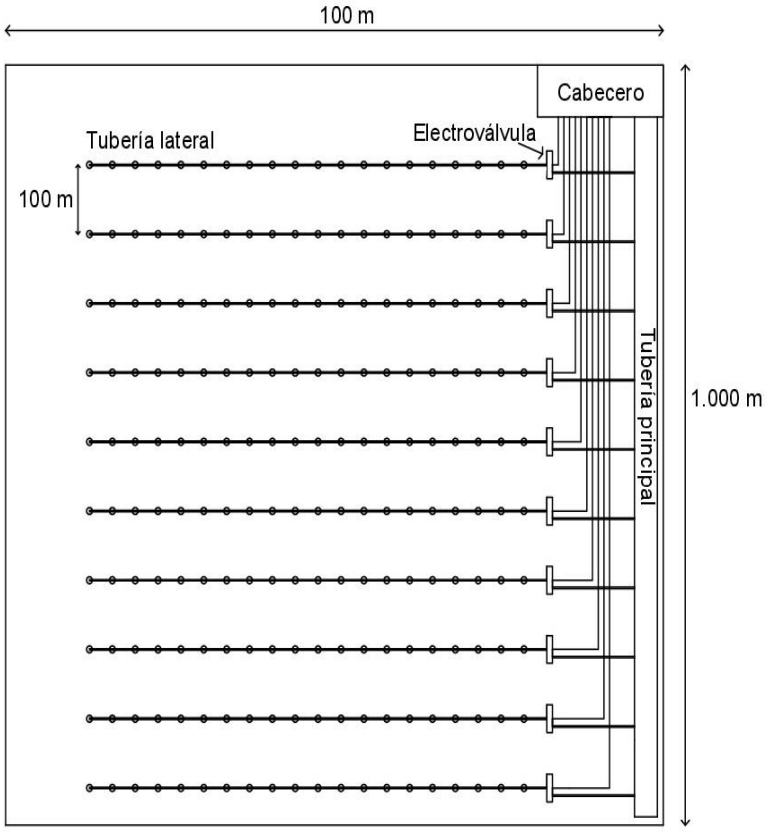


Figura 1. Croquis de una instalación agrícola convencional.

En el extremo superior derecho se encuentra el cabecero de la instalación, donde se sitúa el sistema de alimentación. Luego está la tubería principal por la que circulará el agua hasta cada una de las tuberías laterales que estarán controladas por su respectiva electroválvula. Para alimentar la primera electroválvula se han empleado 100 metros de cable, para la segunda 200 metros, para la tercera 300 metros, y así sucesivamente hasta alcanzar una suma total de cableado de 5500 metros. En cuanto a las tuberías laterales, tendrán instalado un sistema de riego por goteo que permite un mayor ahorro de agua, pero debe comentarse que el estudio relacionado con el diámetro y longitud necesario para cada tubería queda excluido en este trabajo.

A continuación, se incluye una tabla de precios reales de cada uno de los componentes electrónicos.

DESCRIPCIÓN	CANTIDAD	COSTE UNITARIO (€/UNIDAD)	COSTE TOTAL (€)
Cable manguera blanca VVF 2 x 1.5 mm	5500 metros	0.69 €/metro	3795
Electroválvula	10	16	160
COSTE TOTAL DE MATERIALES			3955

Tabla 1. Presupuesto de la parte electrónica para un sistema de riego convencional.

El coste final de la instalación eléctrica de un sistema convencional puede alcanzar 3955 euros, lo que es una cifra bastante elevada. La solución planteada es que en vez de hacer uso de diferentes cables que lleguen a cada electroválvula, pasaríamos a disponer de una única pareja de cables a la que le añadiremos frecuencia modulada con los datos, además de la alimentación. Esta pareja recorrerá toda la instalación y cada electroválvula irá conectada a la misma esperando la orden de activación o desactivación. Por lo que en la cabecera se deberá colocar un sistema de alimentación y de comunicaciones, que puede ser de tipo alámbrica o inalámbrica.

1.4. Estructura general del proyecto.

La memoria de este proyecto está dividida en diferentes capítulos en los cuales se expone detalladamente todas las características y aspectos relacionados con este proyecto. Se distribuyen de la siguiente manera:

Capítulo 1. Introducción general. Se describe brevemente el objetivo general de este proyecto. Incluye el alcance y los antecedentes donde se pondrá en situación al lector ayudándolo así a comprender mejor la finalidad de este proyecto.

Capítulo 2. Herramientas de diseño y desarrollo. Se explican las herramientas de hardware y software empleadas.

Capítulo 3. Descripción del sistema. Se centra en explicar detalladamente cada una de las partes que compone este proyecto.

Capítulo 4. Implementación. Se describe el montaje llevado a cabo para poner en funcionamiento el sistema de riego.

Capítulo 5. Resultados experimentales. Se incluyen pruebas de alcance y fotografías de los resultados obtenidos en el osciloscopio.

Capítulo 6. Problemas y soluciones. Se explican cada uno de los problemas surgidos durante la realización del proyecto.

Capítulo 7. Presupuesto. Se presenta el presupuesto necesario para llevar a cabo este proyecto, además de una comparativa económica entre dos tipos de sistemas de riego: convencional e inteligente.

Capítulo 8. Conclusiones. Se explican las conclusiones obtenidas tras el desarrollo del proyecto.

CAPÍTULO 2. HERRAMIENTAS DE DISEÑO Y DESARROLLO.

En este apartado se procederá a definir cada una de las herramientas empleadas en este proyecto, tanto de *hardware* como de *software*.

2.1. Herramientas de hardware.

2.1.1. Protoboard.

Las placas protoboard son herramientas indispensables en la electrónica. Consisten en pequeñas tablas con perforaciones en toda su superficie donde se podrán conectar diferentes componentes electrónicos. Internamente se encuentran conexionadas de la siguiente manera:

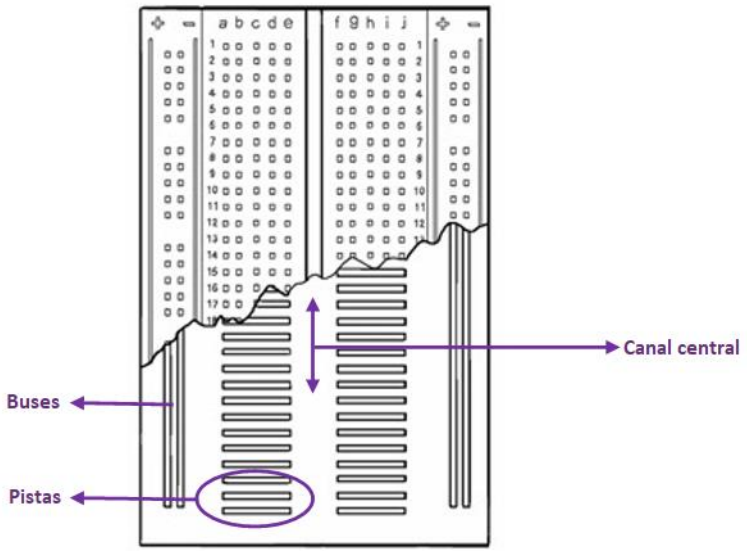


Figura 2. Conexión interna de la protoboard.

Es muy importante tener en cuenta que las conexiones horizontales, las pistas, están divididas por el canal central. Además, la placa empleada en este proyecto tiene también una separación a mitad de las conexiones verticales, los buses.

2.1.2. Osciloscopio.

El osciloscopio es un instrumento de medición que permite la visualización gráfica de aquellas señales eléctricas variables en el tiempo. El eje X representa tiempos y el eje Y representa tensiones.

En este proyecto se ha empleado el modelo HAMEG HMO2524 para observar la modulación y demodulación de las señales enviadas de una placa Arduino a otra.

Figura 3. Osciloscopio empleado para este proyecto.

2.1.3. Fuente de alimentación.

Es un dispositivo capaz de convertir la corriente alterna (CA) en corriente continua (CC), que es la empleada por los dispositivos electrónicos. Para este proyecto se ha empleado el modelo FAC-363B que "contiene tres fuentes de alimentación estabilizadas totalmente independientes. La primera suministra una tensión ajustable entre 0 y 30 V, con limitación de corriente ajustable entre 0 y 2 A. La segunda es una fuente doble fija: -15 V, 0, +15 V, con una corriente máxima de 0,5 A. La tercera, también fija, suministra 5 V, con una corriente de hasta 1 A." [1]

Figura 4. Fuente de alimentación empleada en este proyecto.

2.1.4. Multímetro.

Un multímetro es un instrumento de medidas de carácter electrónico, los parámetros más comunes que nos puede ofrecer son voltios, amperios y ohmios. En el laboratorio ha sido utilizado para medir el valor de las resistencias empleadas en el proyecto, así como para hacer comprobaciones de voltaje para garantizar un correcto funcionamiento.

Figura 5. Multímetro empleado en este proyecto.

2.1.5. Generador de funciones.

Un generador de funciones es un instrumento electrónico utilizado para simular señales de unas características concretas. Podemos variar su frecuencia, su amplitud e incluso simular una señal sinusoidal, cuadrada o triangular.

Se ha empleado en este proyecto para simular la señal enviada entre los Arduinos, comprobando así el correcto funcionamiento del modulador-demodulador.

Figura 6. Generador de funciones empleado en este proyecto.

2.1.6. Arduino UNO.

Arduino es una comunidad de usuarios que ha desarrollado el *software* y *hardware* de diferentes placas capaces de sensar y controlar objetos del mundo real [2], se trata de una plataforma de *Hardware* Libre, es decir, que existen esquemas de acceso público disponibles en internet y cualquiera puede fabricarlos. Arduino UNO es una de las diferentes placas electrónicas que forma la familia Arduino. Está controlada por el microcontrolador ATmega328P, creado por Atmel, que tiene una memoria *flash* (memoria de programa) de 32KB, 1KB de memoria EEPROM, que es un tipo de memoria ROM que puede ser programada, borrada y reprogramada eléctricamente, y 2KB de memoria SRAM, que es la zona de memoria donde se guardan las variables de los programas.

Cuenta con 14 pines que pueden ser configurados como entradas o salidas digitales, de los cuales 6 se puede utilizar como salidas PWM (Modulación por ancho de pulsos). Además, también está formada por 6 entradas analógicas que manejan valores desde 0 hasta 1023 y con ellas podremos leer, por ejemplo, sensores. Un cristal de cuarzo de 16MHz que proporciona la frecuencia de reloj al microcontrolador, una conexión USB a través de la cual se llevará a cabo su programación, un botón de *reset* y un conector de alimentación externa para cuando la placa no esté conectada vía USB. También posee pines de voltaje DC de 3.3 V y 5 V, y tres pines para la puesta a tierra (GND).

Aunque su voltaje límite de entrada está entre 6 V – 20 V, se recomienda usar voltajes comprendidos entre 7 V – 12 V para proporcionar una tensión y potencia adecuada al Arduino. A partir de 12 V es desaconsejable alimentarlo porque el regulador de tensión podría recalentarse y terminar dañado.[3]

En este proyecto se han empleado dos placas fabricadas en China, pero ofrecen las mismas especificaciones que la placa oficial de Arduino.

Figura 7. Placa Arduino empleada en este proyecto.

2.2. Herramientas de software.

2.2.1. Arduino IDE v1.8.5.

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. Este contiene 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:

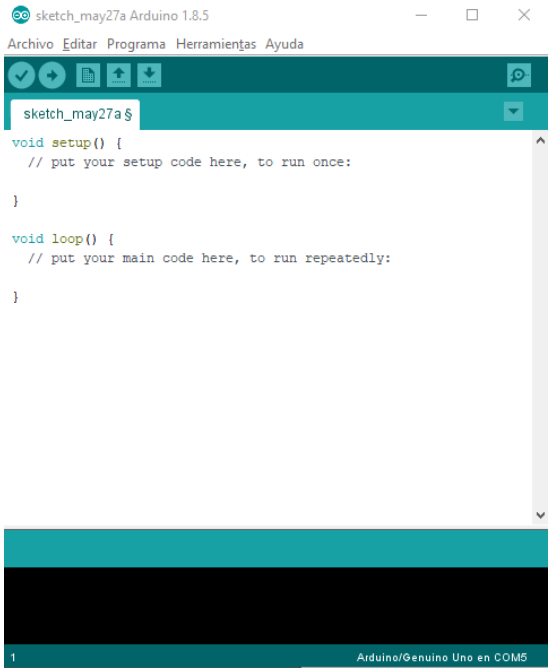


Figura 8. Interfaz Arduino IDE v1.8.5.

2.2.2. Fritzing.

Fritzing es un programa *open source* que permite la realización de esquemas eléctricos en diferentes proyectos. Incluye una amplia biblioteca donde encontraremos todo tipo de componentes como resistencias, condensadores, cables, circuitos integrados, placas de Arduino… Además, podemos crear nuestros propios componentes.

Este *software* ha sido empleado en este proyecto para la realización de los esquemas de comunicación entre Arduinos y el módulo de radiofrecuencia NRF2401.

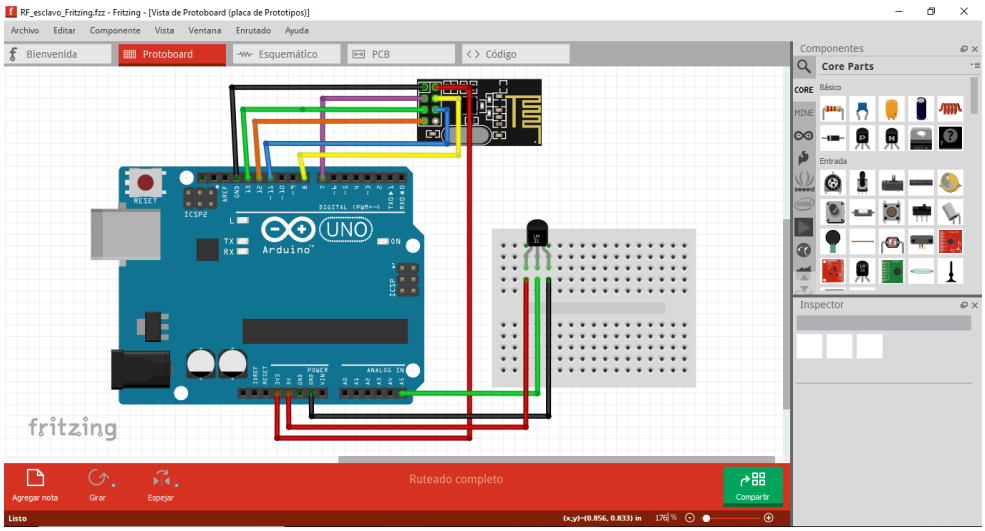


Figura 9. Interfaz de software Fritzing.

2.2.3. LTSpice XVII.

LTSpice es un software gratuito en el que se pueden diseñar esquemas eléctricos y realizar simulaciones de los mismos. Está creado por la empresa *Linear Technology*.

Este *software* ha sido empleado para la realización de los esquemas eléctricos del modulador-demodulador FSK, pero no se ha realizado ninguna simulación del mismo.

CAPÍTULO 3. DESCRIPCIÓN DEL SISTEMA.

3.1. Introducción.

En este prototipo de sistema de riego inteligente se ha desarrollado el código necesario para permitir la comunicación entre dos placas Arduino, usando una arquitectura maestro-esclavo, donde el maestro pedirá al esclavo la información recogida por sus sensores, este deberá enviarle el dato y posteriormente el maestro enviará una orden para activar o desactivar la electroválvula. Esta comunicación podrá ser alámbrica e inalámbrica, y se comparará el funcionamiento de ambas tecnologías. Además, se ha implementado un modulador-demodulador FSK, que permite transportar la información de un punto a otro mejorando la protección contra el ruido y las interferencias.

Así como se ha comentado en la introducción de esta memoria, este proyecto ha sido diseñado para dos alumnos, donde cada uno ha abarcado distintas partes del mismo. El compañero de proyecto Samuel Estévez Rodríguez, basándose en el presente TFG, se ha encargado de la programación de un protocolo de comunicación maestro-esclavo para múltiples nodos de comunicación, pudiendo así conocer las condiciones físicas y climáticas de distintas zonas de una instalación agrícola.

El prototipo final resultante de ambos estudios pretende establecer una comunicación tanto por cable como inalámbrica entre un maestro y varios esclavos, además de leer varios sensores y poner en funcionamiento los correspondientes actuadores.

3.2. Comunicación entre placas Arduino.

Para elaborar este proyecto se ha llevado a cabo la siguiente metodología:

Inicialmente se ha desarrollado la programación de un esquema cliente-servidor para una única pareja maestro-esclavo. A continuación, con este código, se han realizado pruebas de funcionamiento con unos módulos de comunicación inalámbrica de bajo coste, los RF – 433 MHz, y tras comprobar el bajo rendimiento que ofrecen, se realizaron nuevas pruebas empleando los módulos NRF2401. Finalmente, se ha comparado esta tecnología con la de tipo alámbrica, llevando a cabo la construcción de un sistema PLC basado en modulación FSK.

A continuación, en este apartado, se desarrollarán ambas tecnologías de comunicación explicando en cada apartado las técnicas y los dispositivos empleados para llevarla a cabo. También, haciendo uso de diagramas de flujo, se mostrará la estructura de comunicación necesaria para un correcto funcionamiento del sistema de riego.

3.2.1. Comunicación alámbrica.

En este proyecto interesa que tanto la comunicación alámbrica como la inalámbrica sea de tipo *half-duplex*, es decir, que sea bidireccional pero el envío y recepción de datos no sea simultánea.

Para conseguir transmitir las señales de una placa Arduino a otra por la misma pareja de cables de la alimentación, es necesario añadirle frecuencia modulada con los datos. Para lograrlo, se ha implementado un modulador-demodulador FSK.

3.2.1.1. Introducción teórica a la modulación-demodulación.

Para entender la técnica de modulación FSK, primero se deben conocer tres conceptos de señal importantes:

1) *Moduladora:* señal que contiene la información que se quiere transmitir.
2) *Portadora:* señal de alta frecuencia que se encarga de transportar la información de la moduladora.
3) *Modulada:* señal resultante de la variación en frecuencia de la portadora, que se modificará dependiendo de las variaciones de nivel de la moduladora.

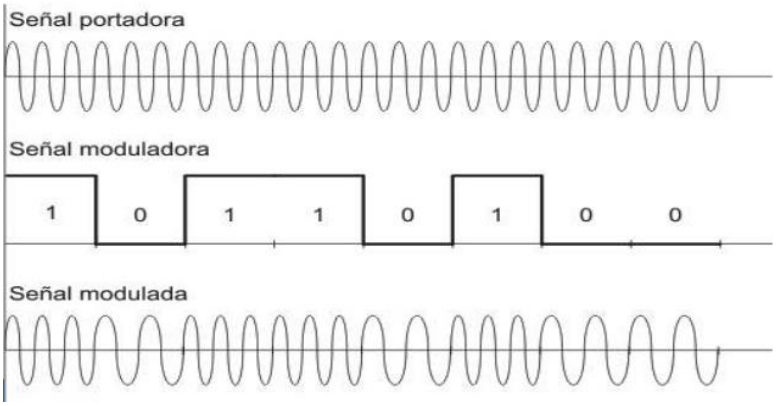


Figura 10. Aspecto de las señales portadora, moduladora y modulada.

Se denomina ***modulación*** al conjunto de técnicas que se usan para transportar información sobre una onda portadora, que generalmente es de tipo sinusoidal. Con esta técnica se aprovechará mejor el canal de comunicación permitiendo transmitir más información de forma simultánea [5]. Consiste en modificar algunos parámetros de la señal portadora (amplitud, frecuencia o fase) en función de la señal moduladora, que contiene información, para que finalmente esta última pueda ser transmitida hasta el otro extremo regenerando allí la señal original mediante la ***demodulación.***

Las principales razones por las que se modula una señal son:

- Facilita la propagación de los datos por cable o por el aire.
- Disminuye el tamaño de las antenas.

- Evita la interferencia entre canales.
- Protege la información del ruido.
- Mejora la calidad de transmisión.

Existe la modulación analógica y la digital. En cada una encontramos tres técnicas de modulación distintas:

Analógica:

- Modulación de la amplitud (AM)
- Modulación de la frecuencia (FM)
- Modulación de la fase (PM)

Digital:

- Modulación por desplazamiento de amplitud (ASK)
- Modulación por desplazamiento de frecuencia (FSK)
- Modulación por desplazamiento de fase (PSK)

La principal desventaja de la modulación analógica es la sensibilidad al ruido. Por ello en este proyecto se emplea modulación digital, concretamente FSK.

Se utiliza modulación FSK porque al conectar elementos pasivos al circuito se producen variaciones de voltaje, y si se realizase una técnica ASK, al modular respecto a la amplitud, estas variaciones podrían causar confusiones en la señal. En cambio, las variaciones de frecuencia son más estables y las de voltaje no causan problemas porque se pueden regenerar, pudiendo así recuperar la señal original transmitida.

La modulación FSK (*Frequency Shift Keying*) es una técnica de modulación para la transmisión digital de información. La frecuencia cambiará dependiendo de los valores discretos de la señal modulante, dicha señal modulante es un flujo de pulsos binarios. Es decir, el lógico 0 está representado por una onda con una frecuencia determinada y el lógico 1 con otra frecuencia diferente.

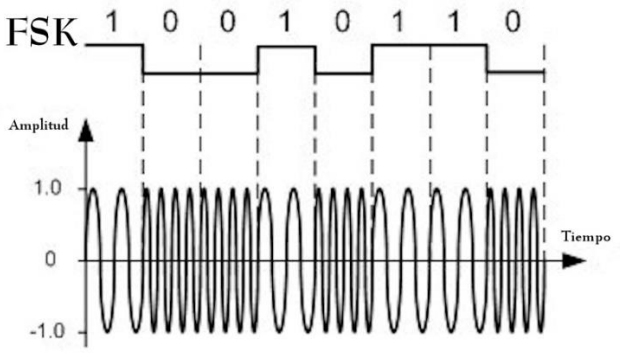


Figura 11. Representación de la modulación FSK.

Por otro lado, la ***demodulación*** consiste en el conjunto de técnicas que se usan para recuperar la información transmitida por una onda portadora, es decir, tras demodular una señal se debe obtener la señal moduladora. En cada extremo de la comunicación debe haber un modulador y un demodulador, denominado *módem*.

3.2.1.2. Circuito modulador-demodulador FSK.

Su función es convertir la señal portadora de frecuencias bajas a frecuencias altas en función de los niveles lógicos de la señal moduladora, consiguiendo una señal modulada con distintas frecuencias, una más alta y otra más baja para poder diferenciar entre los niveles lógicos 1 y 0.

Al elevar su frecuencia, la señal será transmitida hasta una distancia mayor que si no estuviese modulada, ya que en este estado viaja mejor y más rápido.

En este proyecto se ha empleado como modulador y demodulador el circuito integrado CD4046B.

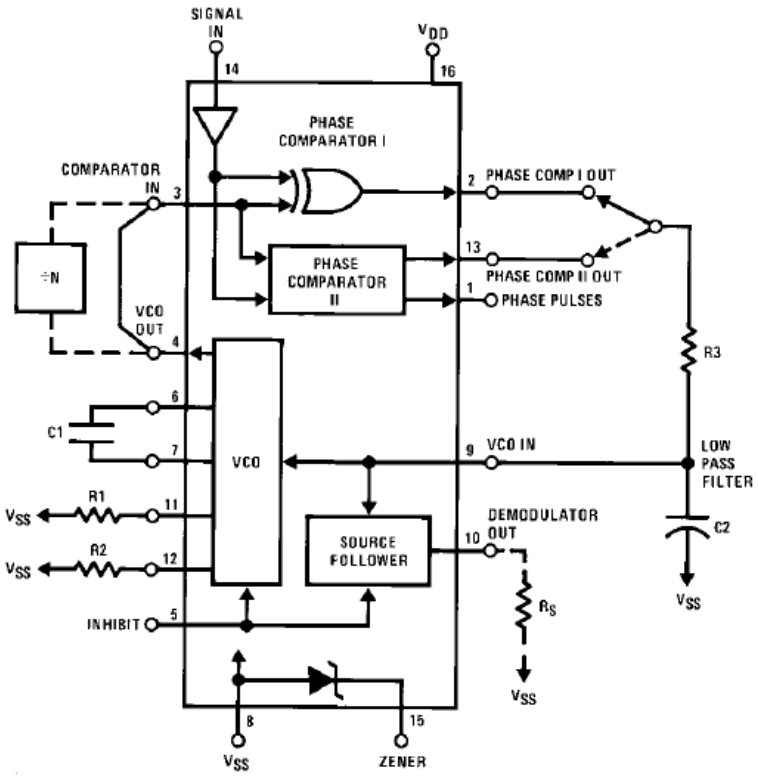


Figura 12. Diagrama de bloques del CI CD4046.

Este circuito es un PLL (*Phase Locked Loop*), que consiste en un sistema de control que genera una señal de salida cuya fase está relacionada con la fase de una señal de entrada. Está formado por un VCO (*Voltage Controlled Oscillator*) que es un oscilador electrónico cuya frecuencia puede ser seleccionada haciendo uso de resistencias y

condensadores. Su aplicación más típica es generar señales moduladas en frecuencia. Y también está formado por dos PC (*Phase Comparator*) que detecta la diferencia de fase.

Para el **modulador** se ha empleado el circuito integrado solamente como VCO, ajustando su frecuencia a 100 kHz. Para seleccionar dicha frecuencia se deben realizar una serie de cálculos que se desarrollarán en el Apartado 4.1.

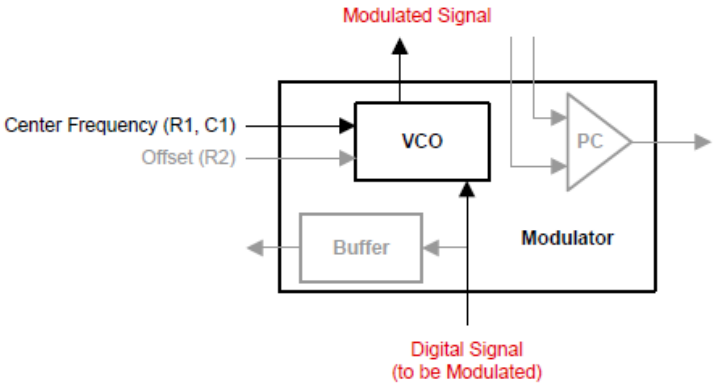


Figura 13. Diagrama de bloques de un PLL como modulador.

El **demodulador** debe extraer la señal moduladora a partir del tren de pulsos. Para ello se ha usado el CI como PLL, donde el VCO genera una señal periódica y el PC compara la fase de la señal modulada con la fase de la señal periódica de entrada. Se emplean las mismas resistencias y condensadores del modulador. Está formado por un filtro pasa-baja (LPF, *Low Pass-Filter)* y haciendo uso de un *buffer* se conseguirá amplificar la potencia de la señal recibida.

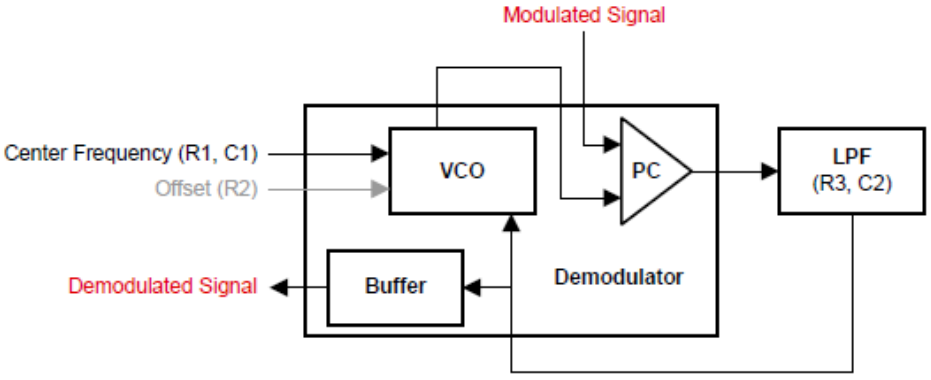


Figura 14. Diagrama de bloques de un PLL como demodulador.

3.2.2. Comunicación inalámbrica.

Para transmitir datos de manera inalámbrica se han estudiado dos módulos de radiofrecuencia existentes en el mercado:

- Módulos RF de 433 MHz
- Módulos NRF2401

Inicialmente se emplearon los módulos emisor y receptor RF – 433 MHz debido a su bajo coste y facilidad de uso, pero tras varias pruebas se comprobó que no cumplían las expectativas necesarias para este proyecto, ya que debido al ruido solamente funcionaban a medio metro de distancia de separación, este problema se explicará detalladamente en el Apartado 6.1. Debido a esto, se buscó la alternativa de los transceptores (transmisor + receptor) NRF2401. También, en el Apartado 5.1, se explicarán las pruebas de alcance realizadas con los diferentes módulos, observando así su rendimiento.

Descripción del transceptor NRF2401.

El dispositivo NRF2401 es un chip de comunicación inalámbrica fabricado por *Nordic Semiconductor* y puede ser conectado a una placa Arduino a través del bus SPI (*Serial Peripheral Interface*).

El bus SPI requiere tres líneas imprescindibles (SCK, MOSI, MISO) y una adicional (SS).

Figura 15. Estructura general del bus SPI.

SCK (*Clock*): Señal de reloj enviada por el maestro.

MOSI (*Master-out, Slave-in*): Comunicación del maestro al esclavo.

MISO (*Master-in, Slave-out*): Comunicación del esclavo al maestro.

SS (*Slave Select*): Selecciona con quién se va a realizar la comunicación.

Las placas Arduino también disponen de bus SPI vinculados a ciertos pines, por lo que en este proyecto se ha comunicado cada Arduino Maestro y Esclavo con un módulo NRF2401, permitiendo así transmitir datos de forma inalámbricas.

En Arduino UNO los pines asociados a SPI son:

MODELO	SS	MOSI	MISO	SCK
ARDUINO UNO	Pin 10	Pin 11	Pin 12	Pin 13

Tabla 2. Pines asociados a SPI del Arduino UNO.

En cuanto a las características del transceptor, integra en un único chip toda la electrónica y bloques funcionales necesarios para establecer comunicaciones RF (Radio Frecuencia) entre dos o más puntos.

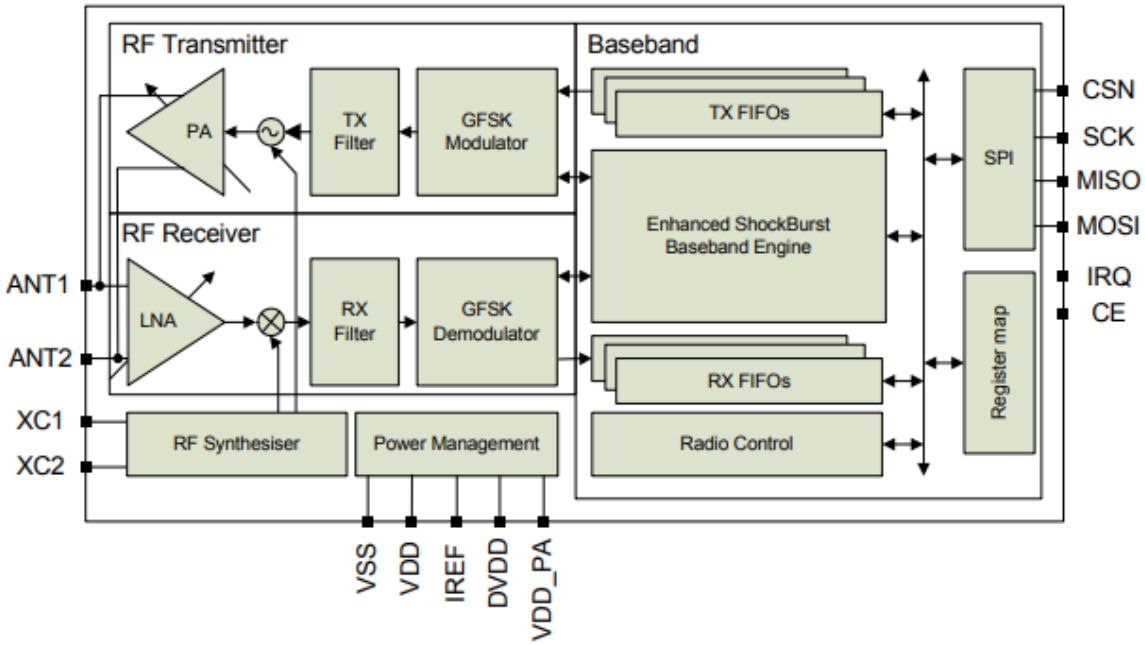


Figura 16. Diagrama de bloques del módulo NRF2401.

El propio módulo se encarga de la modulación y demodulación de la señal, por lo que únicamente es necesario configurar el envío y recepción de datos. Emplea la técnica de modulación GFSK (*Gaussian Frequency Shift Keying)*, cuya dinámica es la misma que en la modulación FSK, pero se le añade un filtro gausiano que suaviza las transiciones de fase de la señal durante su transmisión [6].

Posee corrección de errores y protocolo de reenvío cuando es necesario. Opera en la banda de 2.4 GHz, que es de libre uso a nivel mundial. Posee velocidad configurable de 250 Kbps, 1 Mbps o 2 Mbps. La tensión de alimentación está comprendida entre 1.9 V y 3.6 V. En *stand by* su consumo es muy bajo, y de unos 15 mA durante el envío y la recepción.

Teóricamente, el modelo básico de antena integrada en forma de *zig-zag* ofrece un alcance máximo sin obstáculos de 20-30 metros, y la versión con amplificador y antena

externa, un alcance de 700-1000 metros. Para conseguir el máximo alcance se recomienda alimentar el módulo con una fuente externa de 3.3 V.

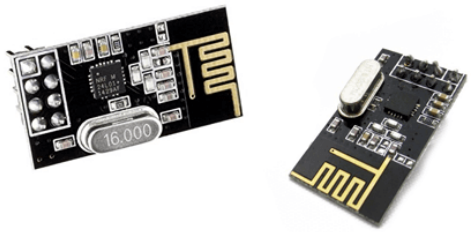


Figura 17. Módulo NRF2401. Versión con antena integrada.

Figura 18. Módulo NRF2401. Versión con antena externa y amplificador.

3.2.3. Estructura general de comunicación entre Arduinos.

Para el correcto funcionamiento del sistema de riego inteligente ha sido necesario crear la comunicación entre un maestro y, al menos, un esclavo, donde el maestro es el encargado de pedir la información necesaria al esclavo para posteriormente poner en funcionamiento o detener los actuadores. A continuación, se mostrarán dos diagramas de flujo en los que se indica cómo se ha establecido el funcionamiento del intercambio de información en el código programado. En el Apartado 3.3.3 se hará una descripción detallada de dicho código.

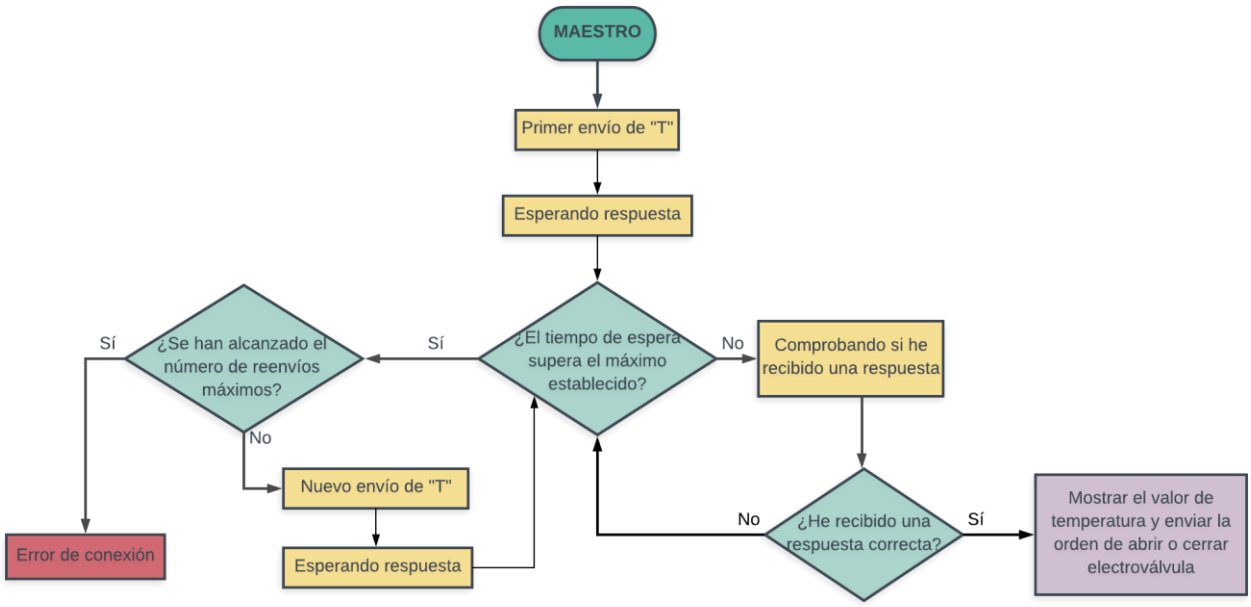


Figura 19. Diagrama de flujo del maestro.

El **maestro** enviará una "T" al esclavo como petición del valor de temperatura, luego se quedará esperando una respuesta. Se establecerá un tiempo máximo de espera durante el que estará comprobando si ha recibido una respuesta correcta. Mientras no sea la respuesta esperada y no se supere el tiempo máximo de espera, seguirá haciendo esta comprobación. En cambio, si obtiene una respuesta correcta, mostrará al usuario el valor de la temperatura y en función del mismo enviará una petición para abrir o cerrar la electroválvula. También puede ocurrir que sí se supere este tiempo de espera, por lo que habrá que definir un número de reenvíos máximo de la petición, en caso de superar estos reenvíos sin haber recibido una respuesta correcta, se informará al usuario de que hay un error para que proceda a hacer las comprobaciones necesarias.

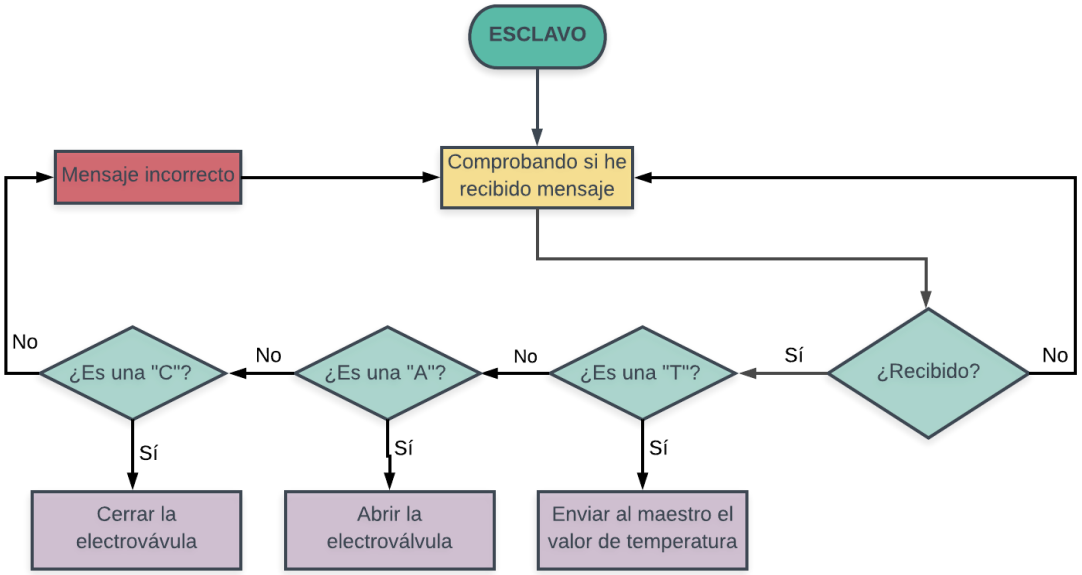


Figura 20. Diagrama de flujo del esclavo.

El funcionamiento del **esclavo** es más sencillo, ya que estará continuamente comprobando si el maestro le ha enviado un mensaje. Si el mensaje recibido es una "T", procederá a enviarle el valor del sensor de temperatura, si es una "A" abrirá la electroválvula y si es una "C" la cerrará. Si no es correcto, le mostrará al usuario que el mensaje recibido es incorrecto y volverá a realizar la comprobación de mensajes recibidos.

3.3. Programación del Arduino.

Para programar cada una de las placas de Arduino se ha empleado el entorno de desarrollo integrado Arduino IDE v1.8.5.

Para la comunicación inalámbrica, inicialmente se diseñó un código para poner en funcionamiento los módulos RF – 433 MHz empleando la librería *VirtualWire*, pero tras comprobar que la comunicación entre las placas no era la deseada, se ha adaptado para los módulos NRF2401 haciendo uso de la librería *RF24*.

Por otro lado, para la comunicación alámbrica se ha podido reutilizar el primer código desarrollado, pues la librería *VirtualWire* puede ser utilizada con cualquier sistema de comunicación, ya que únicamente es necesario declarar un pin transmisor y otro receptor, lo que adquiere especial interés de cara a la mejora de las prestaciones del sistema alámbrico. Por lo que, en lugar de realizar la conexión entre el Arduino y el módulo inalámbrico, se ha podido acoplar directamente desde el Arduino hasta el modulador o demodulador.

Las librerías empleadas para cada tipo de comunicación utilizan diferentes comandos para enviar y recibir los datos, que serán explicados en el siguiente apartado.

3.3.1. Librerías.

Existen gran cantidad de librerías desarrolladas por terceras personas que nos facilitarán mucho el trabajo a la hora de programar. Una librería consiste en un código que al añadirlo a nuestro proyecto aportará nuevas funcionalidades al mismo.

Aunque existen librerías estándar desarrolladas por Arduino y que están previamente instaladas en el IDE de Arduino, para este proyecto se han tenido que descargar e instalar dos librerías no estándar, es decir, que han sido desarrolladas por personas externas al equipo de Arduino.

Para el primer código desarrollado, ha sido necesario descargar la librería *VirtualWire* y para programar los módulos NRF2401 se ha descargado la *NRF24.* Ambas librerías se encargan de gestionar las funciones de los módulos, como por ejemplo el manejo de datos.

Una vez descargadas desde el explorador, para instalarlas en el IDE de Arduino simplemente tendremos que dirigirnos a *Programa > Incluir Librería > Añadir librería .ZIP.* Luego, para incluirla en el código deberemos seguir los mismos pasos, pero esta vez en vez de seleccionar la opción *Añadir librería .ZIP,* nos desplazaremos hasta el final del menú a *Contributed Librerías*, y allí deberemos encontrar la librería añadida.

3.3.2. Comandos.

Cada librería usa sus propios comandos para realizar las distintas funciones. En este apartado se describirá cada uno de los empleados en el código del proyecto, además de los comandos relacionados con la comunicación Serial.

Comandos de la librería *VirtualWire*.

TIPO	COMANDO	DESCRIPCIÓN
De configuración	*vw_setup()*	Inicia la comunicación con el módulo RF.
	vw_set_tx_pin(transmit_pin)	Configura el pin de transmisión de datos.
	vw_set_rx_pin(receive_pin)	Configura el pin de recepción de datos.
De transmisión	*vw_send(message, length)*	Envía el mensaje. Como primer parámetro se le pasa el mensaje a enviar, y como segundo parámetro la longitud del mismo.
	vw_wait_tx()	Se espera hasta que se transmita el mensaje.

De recepción	*vw_rx_start()*	Empieza la lectura. Permanece leyendo hasta recibir algo.
	vw_get_message(buf, &buflen)	Lee el mensaje recibido. "*buf*" es la matriz donde se copia el mensaje recibido. *"buflen"* debe tener el tamaño máximo de la matriz a la entrada, y a la salida retiene los bytes realmente copiados.

Tabla 3. Explicación de comandos de la librería VirtualWire.

Comandos de la librería *RF24*.

TIPO	COMANDO	DESCRIPCIÓN
De configuración	*RF24 radio(pinCE, pinCSN)*	Configura los pines de control que están conectados al módulo. *pinCE:* Pin del Arduino conectado al pin Chip Enable del módulo. *pinCSN:* Pin del Arduino conectado al pin *Chip Select* del módulo.
	const uint64_t pipes[2] = { 0xF0F0F0F0E1LL, 0xF0F0F0F0D2LL }	Configura los *pipes,* que se pueden entender como los canales por donde se transmiten los datos. Con dos *pipes* se consigue una comunicación bidireccional.
	radio.begin();	Inicia la radio.
	radio.openWritingPipe(pipes[0])	Abre canal para transmitir. Se debe definir el canal.
	radio.openReadingPipe(1,pipes[1])	Abre canal para recibir. Se debe definir el canal.
De transmisión	*radio.stopListening()*	Detiene la escucha para poder hablar.
	radio.write(buf, buflen)	Transmite el mensaje.
De recepción	*radio.startListening()*	Comienza a escuchar.
	radio.read(buf, buflen)	Lee el mensaje recibido.

Tabla 4. Explicación de comandos de la librería RF24.

Comandos de la comunicación Serial.

COMANDO	DESCRIPCIÓN
Serial.begin(9600)	Abre el puerto serie y fija la velocidad en baudios para la transmisión de datos en serie.
Serial.print()	Imprime los datos en el puerto serie.
Serial.println()	Tiene la misma función que el comando anterior pero facilita la lectura en el Monitor Serie del *software* porque añade un salto de línea.

Tabla 5. Explicación de comandos de la comunicación Serial.

3.3.3. Descripción del código.

Aunque en el *Anexo II* de esta memoria se incluirán el código del Maestro y del Esclavo de cada tipo de comunicación, debido a la gran similitud entre estos, en este apartado sólo se explicará de forma detallada el de la comunicación inalámbrica. Aunque se incluirá un subapartado, el 3.3.3.1., donde se comentarán algunas líneas consideradas importantes del código de comunicación alámbrica.

Código empleado para la comunicación inalámbrica.

Código Maestro:

La finalidad de este código ha sido explicada en el Apartado 3.2.3. Consiste en el envío de la letra "T" al esclavo para la petición del valor de su sensor de temperatura, y al recibir este dato el maestro se encargará de hacer la conversión necesaria para mostrar al usuario el valor en grados centígrados y según este valor enviará una orden al esclavo para abrir o cerrar la electroválvula. Se ha creado el siguiente código:

```
#include <nRF24L01.h>
#include <RF24.h>
#include <RF24_config.h>
```

En las tres primeras líneas de código se incluyen las librerías necesarias, en este caso se ha instalado la librería RF24 explicada anteriormente. Es totalmente necesaria para poder empezar a utilizar los módulos NRF2401, permitiéndonos utilizar las funciones básicas del módulo, como enviar y recibir datos.

A continuación se elegirán los pines de control del módulo. Si abrimos la librería vemos que incluye la clase RF24, y a partir de esta crearemos el objeto (instancia) de la radio con sus pines de control. En la siguiente línea se indican los *pipes* que vamos a utilizar; se pueden entender como canales de comunicación por donde se enviarán los mensajes. Serán dos porque queremos conseguir una comunicación bidireccional, por

lo que necesitaremos un *pipe* por donde uno hablará y otro escuchará, y el otro funcionará al contrario.

```
const int pinCE = 7;
const int pinCSN = 8;
RF24 radio(pinCE, pinCSN);

const uint64_t pipes[2] = { 0xF0F0F0F0E1LL, 0xF0F0F0F0D2LL };
```

Dentro del *setup*, que es la parte encargada de recoger la configuración, iniciaremos la radio y la comunicación serial a 9600 bps (baudios por segundo), que es el valor típico de velocidad para comunicarse con el ordenador. También se abrirán los canales de comunicación de escritura y lectura.

```
void setup(void) {

  Serial.begin(9600);
  radio.begin();

  radio.openWritingPipe(pipes[0]);
  radio.openReadingPipe(1,pipes[1]);

}
```

Dentro del *loop*, que es la parte que contiene el programa a ejecutar, se declararán las variables necesarias y se mostrarán por el Monitor Serial varios mensajes al usuario.

Inicialmente, se ha creado un bucle *while* que se recorrerá mientras el mensaje sea incorrecto y el número de reenvío actual sea menor que el número de reenvíos máximo declarado. Dentro de este bucle, tras parar la escucha para poder hablar, se enviará la letra “T” para pedir el valor del sensor de temperatura al esclavo. Recordemos que a la función de envío *radio.write(buf, buflen)* le debemos pasar el mensaje que queremos enviar y la longitud del mismo. Tras este envío se aumenta en 1 el número de reenvío actual y lo mostramos por pantalla.

```
void loop(){

  uint8_t Mensaje_Correcto = 0;
  const char *msg1 = "T"; //Mensaje para la petición de la temp
  const char *regar = "A"; //Ordena a Abrir la electroválvula
  const char *no_regar = "C"; //Ordena a Cerrar la electroválvula
  int Numero_reenvios = 10;
  int reenvio_actual = 0;
  uint8_t dato_temp[2]; //Dos elementos

  Serial.println("Iniciando comunicacion");
  while ((Mensaje_Correcto == 0) && (reenvio_actual <
Numero_reenvios)) {

  radio.stopListening(); // Paramos la escucha para poder hablar
  //Mensaje "T" para pedir la temp al esclavo
```

```
  Serial.print("He enviado: ");
  radio.write(msg1, sizeof msg1); //Enviamos el mensaje
  Serial.println(msg1);
  reenvio_actual = reenvio_actual +1;
  Serial.print("Reenvio numero: ");
  Serial.println(reenvio_actual);
```

Luego, se declarará un tiempo máximo de espera para recibir un mensaje y también se le asigna un valor al tiempo destinado al envío, que tendrá el valor de la función *millis(),* que es el tiempo en milisegundos que ha pasado desde que se encendió el Arduino hasta el momento actual.

Tras iniciar de nueva la escucha se crea otro bucle *while*, situado dentro del bucle anteriormente comentado, que se recorrerá mientras el mensaje siga siendo incorrecto y mientras el tiempo total transcurrido desde que se envío el mensaje menos el tiempo destinado al envío sea menor que el tiempo máximo declarado. Dentro de este bucle, si hay datos disponibles, se leerá este dato que tiene una longitud de dos bytes y se mostrarán por pantalla. No es necesario mostrarlos, pero se ha hecho para comprobar que lo que envía el esclavo es lo mismo que recibe el maestro. Se espera recibir dos bytes, si esto ocurre el mensaje será correcto (Mensaje_Correcto = 1).

```
  long Tiempo_Maximo = 500;
  long Tiempo_Envio = millis(); //Actualiza el tiempo actual
  radio.startListening(); //Volvemos a la escucha
  while ((Mensaje_Correcto == 0) && (millis() - Tiempo_Envio <
Tiempo_Maximo)) {

    if (radio.available()) { //Si hay datos disponibles

      Serial.print("He recibido ");
      radio.read(&dato_temp, sizeof(dato_temp));
      for( int i = 0; i < 2; i++)
        Serial.print(dato_temp[i], HEX);
        Serial.println();
      if (sizeof dato_temp == 2) { //Longitud de 2bytes

        Mensaje_Correcto = 1;
      }
    }
    delay(50);
  }
  }
```

Si el mensaje es correcto saldremos de los bucles *while* y entraremos en el siguiente *if* donde se mostrará el valor del sensor de temperatura, que tras una conversión nos proporciona su valor en voltios.

Se trata de un sensor analógico, por tanto, el valor que nos proporciona será de tipo analógico. Y en su *datasheet* se puede observar que trabaja con un factor de escala lineal

de 10 mV/ºC. La función *AnalogRead* que se emplea en el código Esclavo leerá el pin en el que está conectado el sensor y dará un valor entre 0 y 1023, es decir, 1024 valores. Como trabaja a 5 V, para conocer el voltaje en la entrada analógica se deberá multiplicar el valor del sensor por $\frac{5}{1024}$, y para convertir de voltios a grados sólo habrá que aplicar la equivalencia detallada en el *datasheet.* La fórmula resultante será:

$$Temperatura = \frac{ValorSensor * 5.0}{1024 * 0.01}$$

El envío del dato de temperatura realizado por el esclavo se explicará con detenimiento en su código, pero el maestro recibirá el valor del sensor dentro de un *array* de dos *bytes,* donde para obtener el valor final se deberá mover el dato de la posición 0 del *array* hacia la izquierda 8 *bits*, y luego realizar una operación OR (una suma) *bit* a *bit* con el dato de la posición 1 del *array*.

Si la temperatura supera o iguala los 28 grados centígrados, enviará la orden para regar, es decir, abrir la electroválvula. Si es menor, enviará la orden para cerrarla.

```
if (Mensaje_Correcto == 1) {
    Serial.println("Mensaje correcto");
    int ValorSensor;
        ValorSensor = dato_temp[0] << 8 | dato_temp[1];
        Serial.print("Valor sensor recibido: ");
        Serial.println(ValorSensor);
    float Temperatura;
        Temperatura = (ValorSensor * 5.0)/(1023.0 * 0.01);
        Serial.print("Valor temperatura: ");
        Serial.println(Temperatura);

    if (Temperatura >= 28.0) {
      radio.stopListening();
      Serial.println("Se debe regar");
      Serial.println("He enviado");
      radio.write(regar, sizeof regar); //Enviamos el mensaje
      Serial.println(regar);
      radio.startListening();
    }

    if (Temperatura < 28.0) {
      radio.stopListening();
      Serial.println("NO se debe regar");
      Serial.println("He enviado");
      radio.write(no_regar, sizeof no_regar); //Enviamos el mensaje
      Serial.println(no_regar);
      radio.startListening();
    }
}else
   Serial.println("Error de Conexion");

} //Cerramos void loop()
```

Por último, si no se cumplen las condiciones impuestas en los bucles *while* ni en el *if* final, se mostrará un mensaje de error de conexión por pantalla, pues esto puede

significar que el mensaje ha sido enviado un máximo de veces y no se ha recibido una respuesta (o la respuesta recibida no es correcta) durante estos intentos; por lo que sería necesario hacer una comprobación en el funcionamiento del sistema.

Código Esclavo:

Del mismo modo que en el código maestro, la finalidad del código esclavo ha sido explicada en el Apartado 3.2.3. Consiste en comprobar continuamente si se ha recibido algún mensaje. Si se recibe una "T", se enviará el valor del sensor de temperatura. Si se recibe una "C" se cerrará la electroválvula y si se recibe una "A" se abrirá.

En este código también es necesario incluir la librería RF24, que corresponde a las tres líneas de código siguientes:

```
#include <nRF24L01.h>
#include <RF24.h>
#include <RF24_config.h>
```

Será necesario declarar uno de los pines donde irá conectada la electroválvula.

```
const int electrovalvula = 9;
```

Las siguientes líneas de configuración son idénticas al código maestro, únicamente se deberán abrir los canales de comunicación adecuados, se tendrán que invertir los *pipes*, ya que por uno de los canales el maestro escribe y el esclavo escucha (*pipe[0]*) y por el otro canal el esclavo escribe y el maestro escucha (*pipe[1]*). Y también se configurará el pin destinado a la electroválvula.

```
radio.openWritingPipe(pipes[1]);
radio.openReadingPipe(1,pipes[0]);

pinMode(electrovalvula, OUTPUT);
digitalWrite(electrovalvula, false);
```

Dentro del *loop* se llevará a cabo el programa. Si hay datos disponibles procederá a leer el mensaje recibido, además de mostrar por pantallo lo que se ha recibido.

```
void loop(){

uint8_t Mensaje_Correcto = 0;
int i;

  if (radio.available()) { //Si hay datos disponibles
     Serial.println("Recibiendo mensaje");
     Serial.print("He recibido: ");
     radio.read(msj_maestro, sizeof(msj_maestro));

      for(i = 0; i < 2; i++)
         Serial.print((char)msj_maestro[i]);
        Serial.println();
```

Se debe almacenar el mensaje recibido dentro de una matriz y además pasarle el tamaño que tendrá el mismo, en este caso sabemos que recibiremos dos *bytes.* Con el bucle *for* se mostrará el mensaje por pantalla.

A continuación, si el mensaje recibido es una "T" se procederá a leer el valor del sensor de temperatura. Además, a modo de comprobación, se han hecho los cálculos de la conversión de valor analógico a valor en grados centígrados, aunque sólo es necesario conocer el valor del sensor, que es el dato que se enviará desde el esclavo al maestro.

Se declarará una matriz de dos elementos llamada *Envío,* y los datos se enviarán en hexadecimal, ya que si hay algún fallo en el envío de algunos de los *bytes*, es más fácil de detectar. En la posición 0 de la matriz se envía el último elemento hexadecimal, para ello debemos desplazarnos 8 *bits* a la derecha. En la posición 1 de la matriz se envía el primer elemento hexadecimal, para ello realizamos una operación AND (multiplicación) con el número 255 cuya representación en hexadecimal es 0xFF y en decimal 11111111. Al realizar esta multiplicación, nos quedaremos solamente con el primer *byte*.

```
 if ((char)msj_maestro[0] == 'T') {
    Serial.println("Mensaje correcto");

    unsigned int ValorSensor = 0;
    float Voltaje = 0.0;
    float Temperatura = 0.0;
    Serial.print("Valor de sensor enviado: ");
    ValorSensor = analogRead(A5); //Leemos el valor del sensor
    Serial.println(ValorSensor);

    Serial.print("Voltaje sensor: ");
    Voltaje = (ValorSensor* 5.0)/1024.0;
    Serial.println(Voltaje);

    Temperatura = Voltaje/0.01; //Fórmula matemática que calcula la
temp en función del voltaje
    Serial.print("Valor temperatura: ");
    Serial.println(Temperatura);

    radio.stopListening(); // Paramos la escucha para poder hablar
    uint8_t Envio[2]; //Dos elementos
    Envio[0] = (ValorSensor >> 8); //Mueve 8bits hacia la derecha
    Envio[1] = (ValorSensor & 0xFF); //AND (producto) con 255 (número
más pequeño que puede representar)
    radio.write((uint8_t *)Envio, 2);
    Serial.print("Valor en HEXADECIMAL: ");
    Serial.print(Envio[0],HEX);
    Serial.println(Envio[1],HEX);
    radio.startListening(); //Volvemos a la escucha
```

Y, por último, se activará o desactivará la electroválvula en función de la letra recibida.

```
        if((char)msj_maestro[0] == 'A') {
         Serial.println("Se debe regar");
         digitalWrite(electrovalvula, true);
         }

         if((char)msj_maestro[0] == 'C') {
         Serial.println("NO se debe regar");
         digitalWrite(electrovalvula, false);
         }else
           Serial.println("Mensaje no correcto");

      }
      delay(200);
}
```

3.3.3.1. Cambios realizados en el código para otros sistemas de comunicación.

Código empleado para la comunicación alámbrica.

Tal y como se observa en el Anexo II.1, que incluye el código Maestro y Esclavo para la comunicación alámbrica, el cambio más notable está relacionado con los comandos empleados para realizar el envío y recepción de datos, y aunque el código es muy similar al de la comunicación inalámbrica, a continuación se explicarán algunas líneas consideradas importantes.

Código Maestro:

El primer cambio realizado, tras incluir la librería *VirtualWire*, es la declaración y configuración de los pines de transmisión y recepción. Para ello se ha desarrollado la siguiente parte del código:

```
const int transmit_pin = 10;
const int receive_pin = 11;

vw_set_tx_pin(transmit_pin);
vw_set_rx_pin(receive_pin);
vw_rx_start();
```

Por otro lado, las líneas que cobran especial interés e importancia en este código están relacionadas con la modulación y demodulación, pues para llevar a cabo la comunicación bidireccional los Arduinos Maestro y Esclavo no pueden estar hablando a la vez, pues provocaría que ninguno de los dos recibiese correctamente los mensajes. Para evitar esto se ha usado el pin 5, pin de inhibición, del circuito integrado CD4046, que permite activar la modulación de la señal cuando recibe un lógico 0, o desactivarla cuando recibe un lógico 1.

Se ha configurado el pin 8 del Arduino como salida e inicialmente tiene una configuración alta, es decir, el pin está activo y proporcionará voltaje al pin de inhibición provocando que la modulación esté detenida.

```
const int CD4046 = 8;

pinMode (CD4046, OUTPUT);
digitalWrite (CD4046, HIGH);
```

Una vez se quiera enviar un mensaje, se pondrá el pin 8 en configuración baja poniendo a 0 V el pin de inhibición y, por tanto, activando la modulación.

```
  //Iniciamos la modulación
  digitalWrite (CD4046, LOW); //Lo ponemos a 0 para empezar a modular
  delay (320);
  Serial.print("He enviado: ");
  vw_send((uint8_t *)msg1, strlen(msg1));
  vw_wait_tx();
  Serial.println(msg1);
  //Detenemos la modulación
  digitalWrite (CD4046, HIGH); //Lo ponemos a 1 para detener la
modulación
```

Al terminar el envío, se deberá detener de nuevo la modulación, permitiendo así recibir la contestación del esclavo.

CAPÍTULO 4.IMPLEMENTACIÓN.

4.1. Alámbrica.

Para este tipo de comunicación ha sido necesaria la implementación del circuito modulador-demodulador FSK.

Este sistema está formado por un módulo emisor y otro receptor. A su vez, cada módulo está constituido por diferentes bloques que serán explicados en el presente apartado.

MÓDULO EMISOR.

Describiendo el módulo desde abajo hacia arriba nos encontramos con:

- Modulador
- Seguidor de tensión
- Transformador
- Bobina

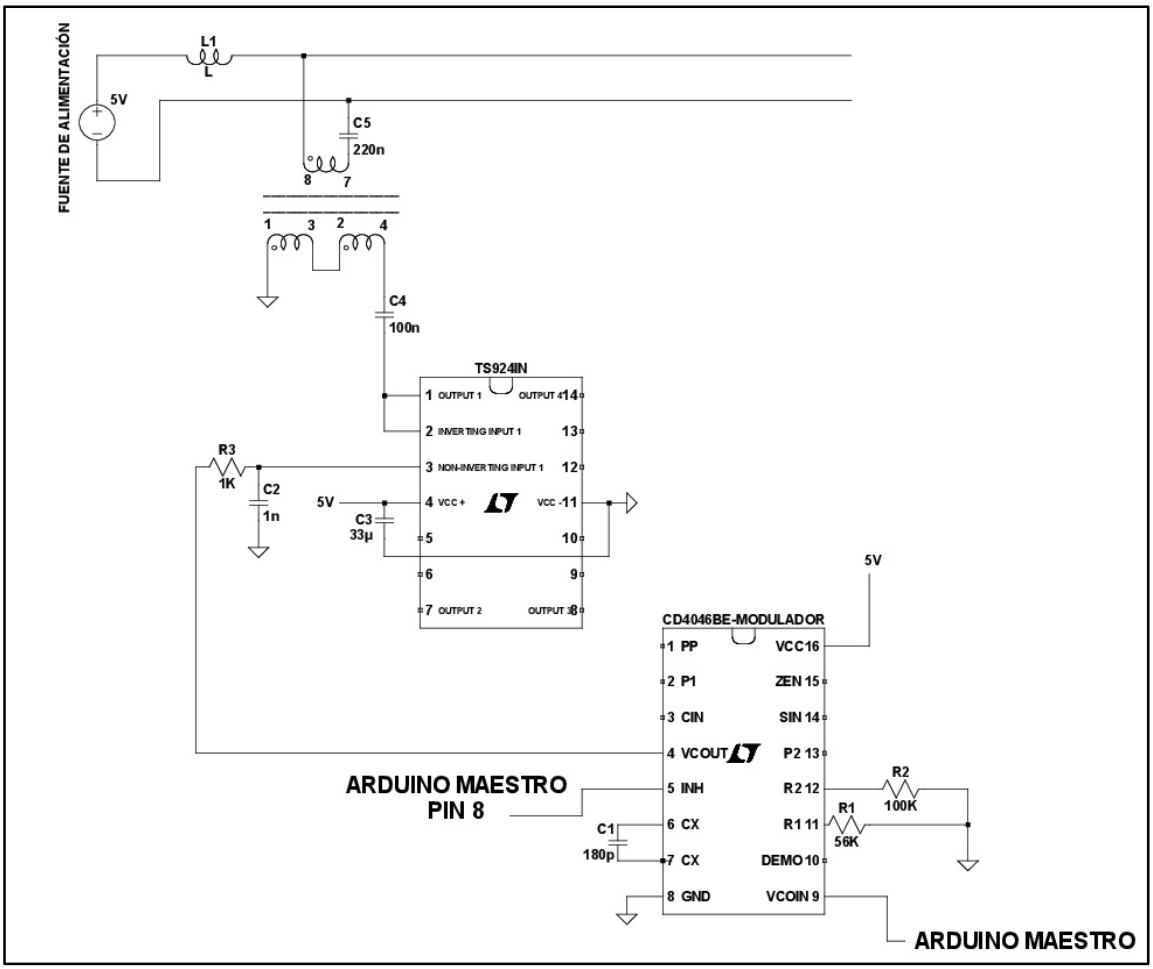


Figura 21. Bloques del módulo emisor.

Modulador:

Su función es convertir la señal de frecuencia baja de 220 Hz emitida por el Arduino Maestro a una frecuencia alta que seleccionaremos a 100 kHz. La señal enviada por el Arduino entrará al circuito integrado CD4046 por el pin 9.

La frecuencia central f_o del modulador se ajustará a 100 kHz mediante las resistencias R_1 y R_2, y el condensador C_1.

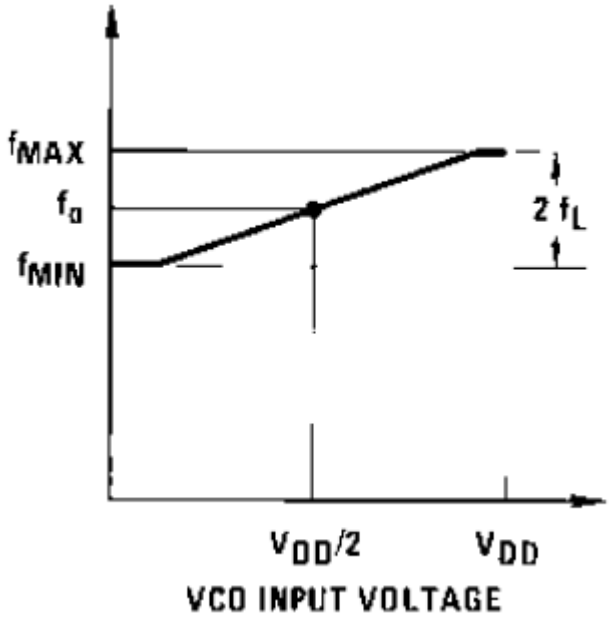


Figura 22. Frecuencia de oscilación del VCO en función de su tensión de entrada.

En la *Figura 21* se puede observar que:

$$2 \cdot f_L = f_{máx} - f_{min}$$

$$f_L = f_{máx} - f_o = f_o - f_{min}$$

Si partimos de que queremos obtener una frecuencia máxima de 140 kHz y una mínima de 60 kHz, se obtendrá que:

$$2 \cdot f_L = 140\ kHz - 60\ kHz = 80\ KHz$$

$$f_L = 40\ kHz$$

A partir de la siguiente figura, se podrán determinar los valores de R_2 y C_1 para conseguir una frecuencia mínima de 60 kHz:

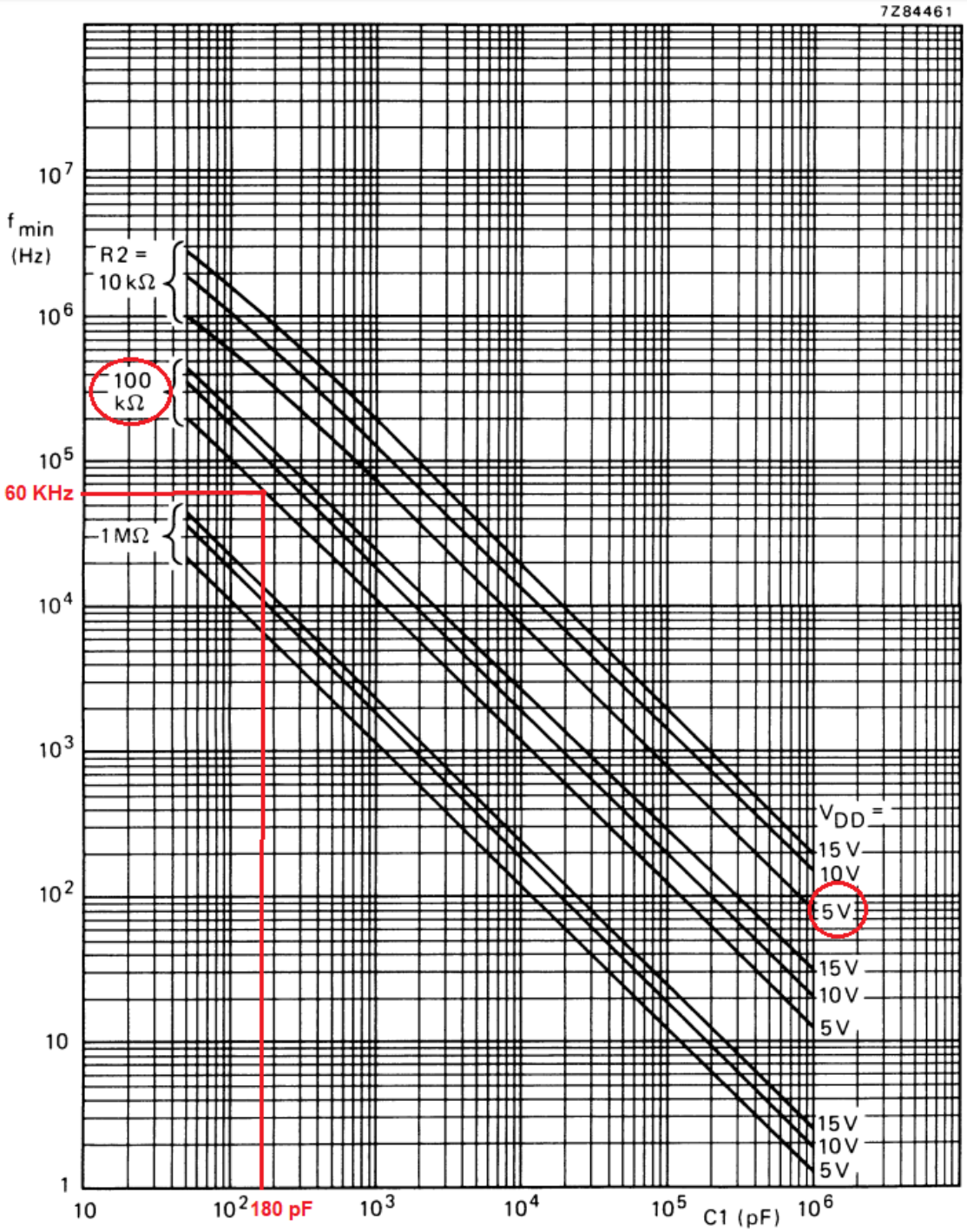


Figura 23.Determinación de R_2 y C_1 a partir de f_{min}

Para 5 V se ha obtenido que $R_2 = 100\ K\Omega$ y $C_1 = 180\ pF$.

A continuación se calculará R_1. Para ello hay que determinar la relación entre la frecuencia máxima y la mínima:

$$\frac{f_{máx}}{f_{min}} = \frac{140\ KHz}{60\ KHz} = 2'33$$

Con este valor y la siguiente figura, se determinará la relación $\frac{R_2}{R_1}$.

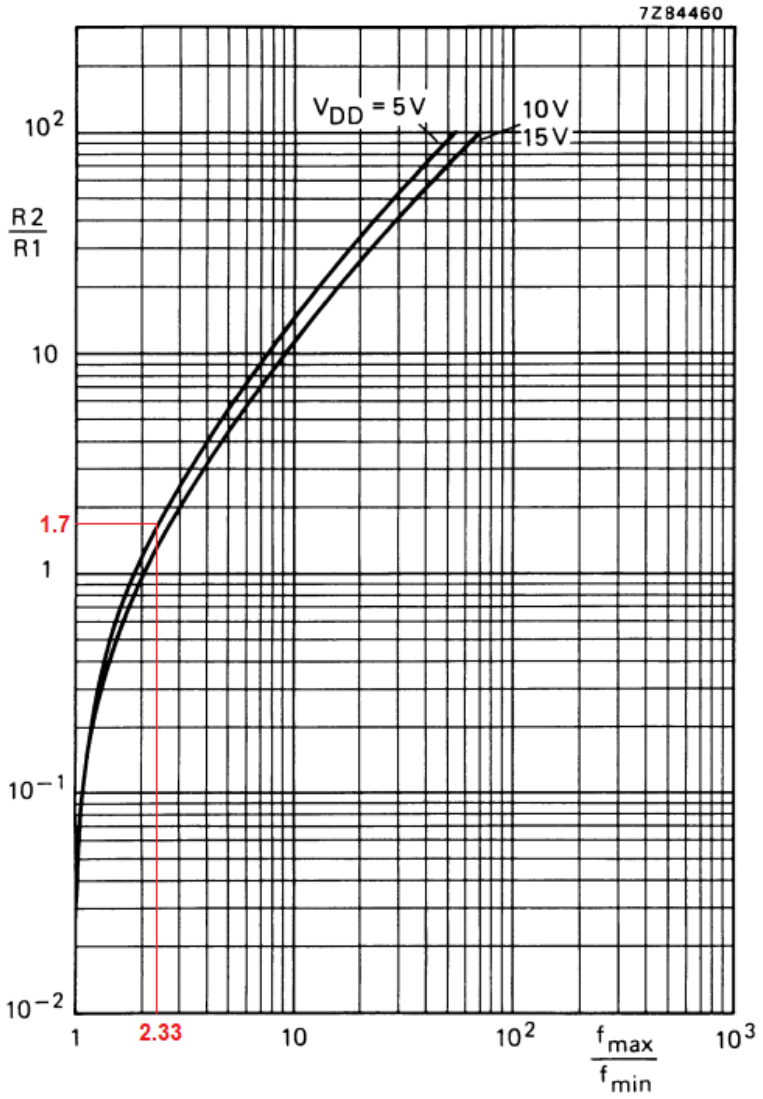


Figura 24. Determinación de R_1 a partir de R_2 y el coeficiente $\frac{f_{máx}}{f_{min}}$.

$$\frac{R_2}{R_1} = 1.7$$

$$R_1 = 58.8\ K\Omega$$

Finalmente, los valores estándar seleccionados para conseguir las frecuencias indicadas son: $R_1 = 56\ K\Omega$; $R_2 = 100\ K\Omega$ y $C_1 = 180\ pF$.

A la salida del modulador, pin 4, se ha colocado la resistencia R_3 y el condensador C_2 para tratar de filtrar la señal cuadrada consiguiendo algo más similar a una senoidal, pues al trabajar con una señal cuadrada se obtenía en el otro extremo del cable una onda distorsionada que, al sufrir cada uno de sus armónicos distintas modificaciones de fase, era difícilmente recuperable. Esto es debido a que se puede asumir que un pulso

cuadrado está formado por una serie de señales senos y cosenos de distintas frecuencias que responden de manera distinta cada una, lo que causa una variación considerable en la señal.

Seguidor de tensión:

Se ha empleado el circuito integrado TS924IN como seguidor de tensión. Este tiene la funcionalidad de adaptar impedancias, ya que cuando la señal atraviesa R_3 y C_2 tiene una impedancia de salida muy alta y no es posible conectar el transformador porque reduciría la corriente enormemente. Con este seguidor se ha conseguido copiar el voltaje y amplificar su intensidad, para así poder enviarla con la suficiente potencia al transformador y ser transmitida correctamente hasta el módulo receptor.

A la salida del seguidor se ha conectado un condensador para proteger el circuito pues, aunque no debería, por algún error se podría inyectar corriente continua y esto podría estropearlo, de esta forma proporcionamos seguridad al mismo. Además, elimina la componente de continua de la señal modulada centrándola en cero y aprovechando mejor las características del transformador.

Transformador:

Su función es acoplar la señal modulada de alta frecuencia de corriente alterna a la señal de corriente continua que circula por la pareja de cable proporcionada por la fuente de alimentación. Con esto se habría cumplido uno de los objetivos de este proyecto, pues desde el módulo emisor al receptor están viajando los datos y la alimentación a través de la misma pareja de cables.

Bobina:

Observando el circuito desde la bobina a su izquierda, se encuentra la fuente de alimentación. Para esta parte del mismo, la bobina actúa como filtro-pasa baja, por lo que se encarga de aislar las altas frecuencias y, en consecuencia, aísla la corriente alterna de la corriente continua para esta parte del circuito. De esta forma se evita que la señal de comunicación de alta frecuencia sea anulada por la corriente continua, pues para una señal de corriente alterna de alta frecuencia, la corriente continua es tierra.

Por otro lado, permite el paso de la corriente continua hasta la siguiente parte del circuito, pues se cortocircuitaría y se comportaría como un cable.

Finalmente, tendremos la señal de corriente alterna acoplada a la de continua, permitiendo transmitir, además de la alimentación, la información hasta el otro punto de la instalación agrícola.

MÓDULO RECEPTOR.

Describiendo el módulo desde arriba hacia abajo nos encontramos con:

- Bobina
- Transformador
- Comparador
- Demodulador
- Comparador

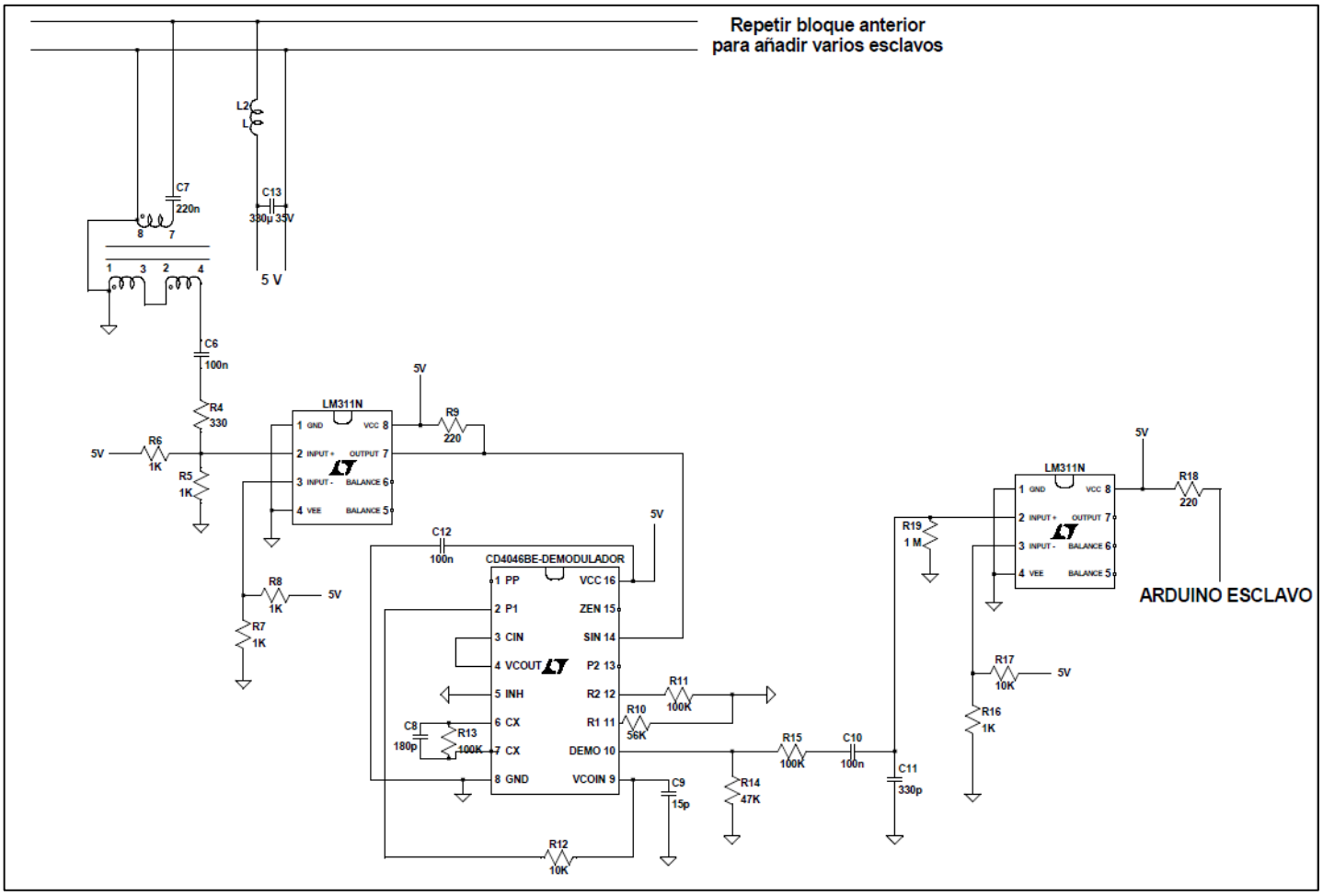


Figura 25. Bloques del módulo receptor

Bobina:

De igual manera que en el módulo emisor, la bobina separa las altas frecuencias y sólo permite el paso de las bajas frecuencias. A su salida solamente encontraremos los 5 V de continua.

Transformador:

En este extremo el transformador se encarga de desacoplar la corriente alterna de la corriente continua, consiguiendo así que los datos transmitidos se dirijan hasta el demodulador.

Comparador:

Tras atravesar los metros de cable hasta llegar al módulo receptor, la señal obtenida a la salida del segundo transformador tiene una tensión muy baja, por lo que ha sido necesario regenerarla. Para ello se ha empleado el circuito integrado LM311N como comparador.

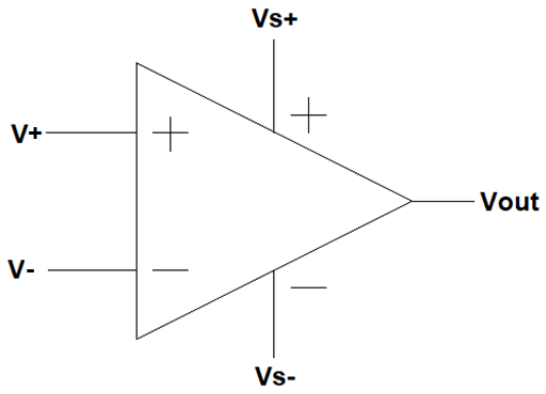


Figura 26. Amplificador operación para la explicación de un comparador.

De este modo, observando la Figura 26, si la tensión V+ es mayor que V-, la tensión de salida Vout será igual a la de la alimentación positiva, es nuestro caso serían 5 V. En cambio, si V- es mayor que V+, la tensión de salida será igual a la de la alimentación negativa, que está conectada a tierra y por tanto serían 0 V. Debido al partidor de tensión implementado en la entrada inversora, y el condensador de desacoplo y las resistencias de la entrada no inversora, se hace una comparación con 2.5 V.

Demodulador:

Tiene por objetivo recuperar el tren de pulsos enviado inicialmente, sin modular, por el Arduino Maestro a una frecuencia de 220 Hz. Tras la modulación, la señal está oscilando a una frecuencia de 100 KHz y haciendo uso del CI CD4046 como PLL se consigue disminuir de nuevo la frecuencia de la señal, pudiendo así ser recibida por el Arduino Esclavo.

El PLL está formado por un VCO, un PC y un filtro pasa-baja. El VCO se implementará con los mismos componentes que en el modulador, generando así una señal de una frecuencia idéntica. Luego, conectando el pin 4 (salida del VCO) con el 3, se dirige hacia una de las entradas del comparador de fase. En la otra entrada irá conectada la señal proveniente del modulador. Tras esta comparación, pasa por un filtro pasa-baja y el siguiente paso será llevar a cabo la demodulación, obteniendo a la salida, pin 10, una señal con frecuencia de 220 Hz. Se puede observar el diagrama de bloques del integrado en la *Figura 12*.

Debido a los elementos pasivos se producen variaciones de voltaje, por lo que a continuación se regenerará la señal mediante un comparador.

Comparador:

Tras demodular la señal se han producido variaciones de voltaje, por lo que a la salida no se obtienen los 5 V necesarios para que el Arduino Esclavo reciba el mensaje, por ello se emplea de nuevo un comparador, consiguiendo regenerar la señal cuadrada de la salida del demodulador y pudiendo así recibirla el Arduino Esclavo correctamente.

Destacar, que hasta este punto la comunicación únicamente será unidireccional, sólo el Maestro podrá enviar datos al Esclavo. Para conseguir que sea tipo bidireccional habrá que implementar otro modulador-demodulador, pero esta vez en el sentido contrario, de forma que cada bloque estará constituido por un modulador y un demodulador.

El esquema final realizado con el *software* LTSPice para una comunicación bidireccional entre un único maestro-esclavo se encuentra en el *Anexo III*.

4.2. Inalámbrica.

Para la implementación de la comunicación inalámbrica se han conectado los transceptores NRF2401 siguiendo este esquema de montaje:

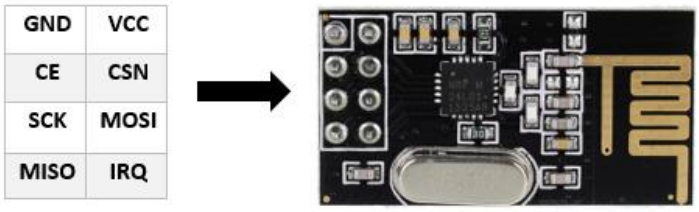


Figura 27. Pines asociados al transceptor NRF2401.

Arduino UNO	NRF2401
GND	GND
3.3 V	VCC
PIN 7	CE
PIN 8	CSN
PIN 13	SCK
PIN 11	MOSI
PIN 12	MISO

Tabla 6. Esquema de conexión entre Arduino y módulo RF.

En la siguiente figura se verá de forma clara cómo quedaría la conexión entre la placa Arduino Maestro y el transceptor NRF2401, y de igual manera para el Arduino Esclavo. Únicamente será necesario cargar a cada placa el código correspondiente y ya se establecerá la comunicación inalámbrica bidireccional.

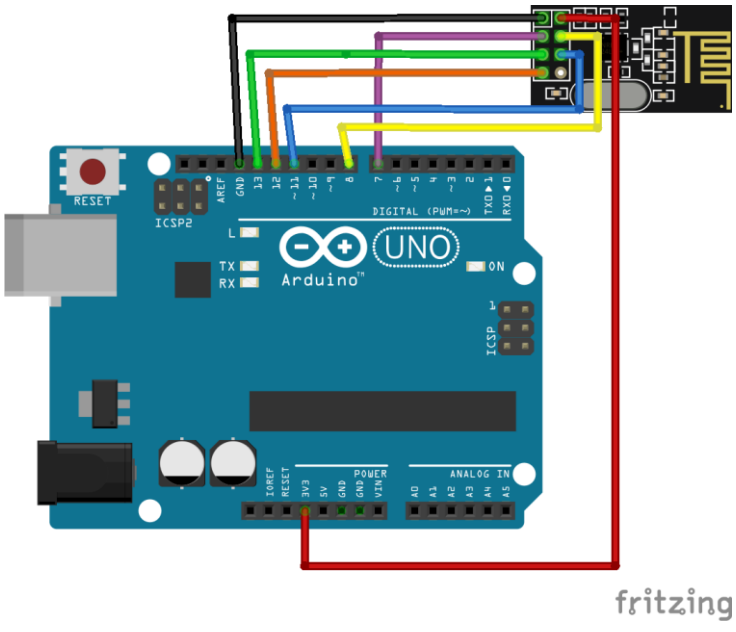


Figura 28. Conexión del módulo NRF2401 con el Arduino Maestro.

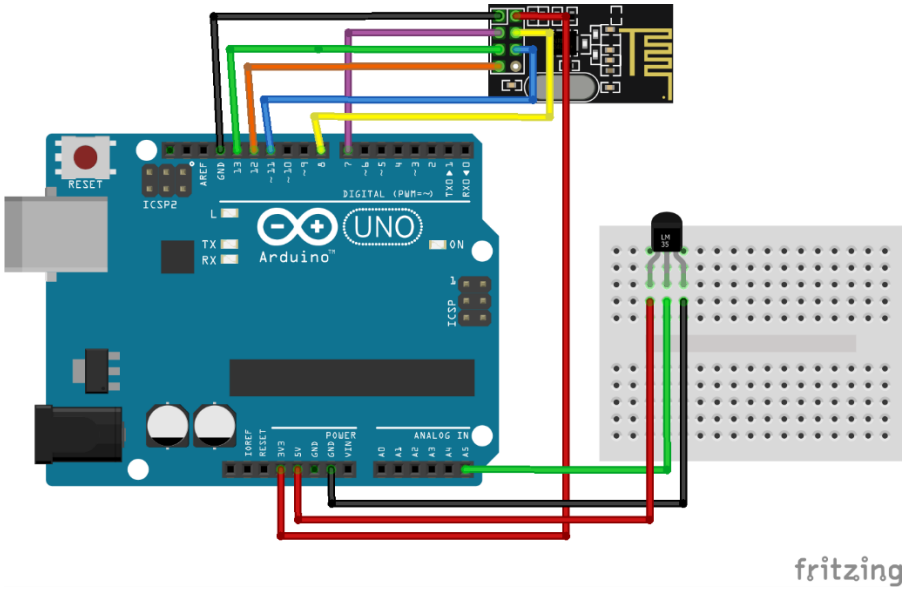


Figura 29. Conexión del módulo NRF2401 con el Arduino Esclavo.

CAPÍTULO 5.RESULTADOS EXPERIMENTALES.

5.1. Pruebas de alcance de los módulos RF.

5.1.1. Módulos RF – 433 MHZ.

Las pruebas de rendimiento realizadas con estos módulos han sido desarrolladas sin una antena externa y alimentados a 5 V. Los resultados obtenidos se han visualizado a través del Monitor Serie que ofrece el entorno de desarrollo Arduino IDE y se ha iniciado la comunicación con el módulo a una frecuencia de 2000 bps.

La primera prueba se realizó con las placas Arduino prácticamente unidas, para comprobar que el funcionamiento del código desarrollado era el esperado. Tras esta verificación se confirmó que la transmisión de datos era correcta.

La siguiente prueba se realizó con una separación de medio metro y se comprobó que no se enviaban ni recibían datos. Se varió la frecuencia de comunicación con el módulo, pero el resultado fue el mismo. Por lo que se concluyó con que, al menos sin hacer uso de una antena externa, estos módulos ofrecen un alcance muy limitado.

5.1.2. Módulos NRF2401.

Las pruebas de rendimiento se realizaron también sin una antena externa, pero debían ser alimentados 1.9 V y 3.6 V, por lo que se usó la alimentación de 3.3 V que ofrece la placa Arduino. De nuevo, los resultados obtenidos se han visualizado a través del Monitor Serie.

Inicialmente se comprobó la transmisión de datos en un medio cerrado sin obstáculos con una separación entre los módulos de 2 metros, logrando un resultado perfecto.

La siguiente prueba se realizó en un medio cerrado pero esta vez con obstáculos. Se interpusieron diferentes paredes entre los módulos, y la distancia real máximo de uso alcanzada en estas condiciones fue de aproximadamente 10 – 12 metros.

Y, por último, se realizó en un medio abierto y sin obstáculos, donde el alcance máximo real obtenido estaba comprendido entre 30 – 35 metros, un resultado bastante óptimo haciendo único uso de la antena integrada del módulo.

5.2. Funcionamiento del modulador – demodulador FSK.

Tras varios problemas relacionados con el punto de trabajo de los operacionales, que se explicarán con detenimiento en el Apartado 6.2, se emplearon los circuitos integrados LM311N y TS924IN como comparadores y seguidores de tensión para conseguir un resultado óptimo obteniendo las siguientes señales:

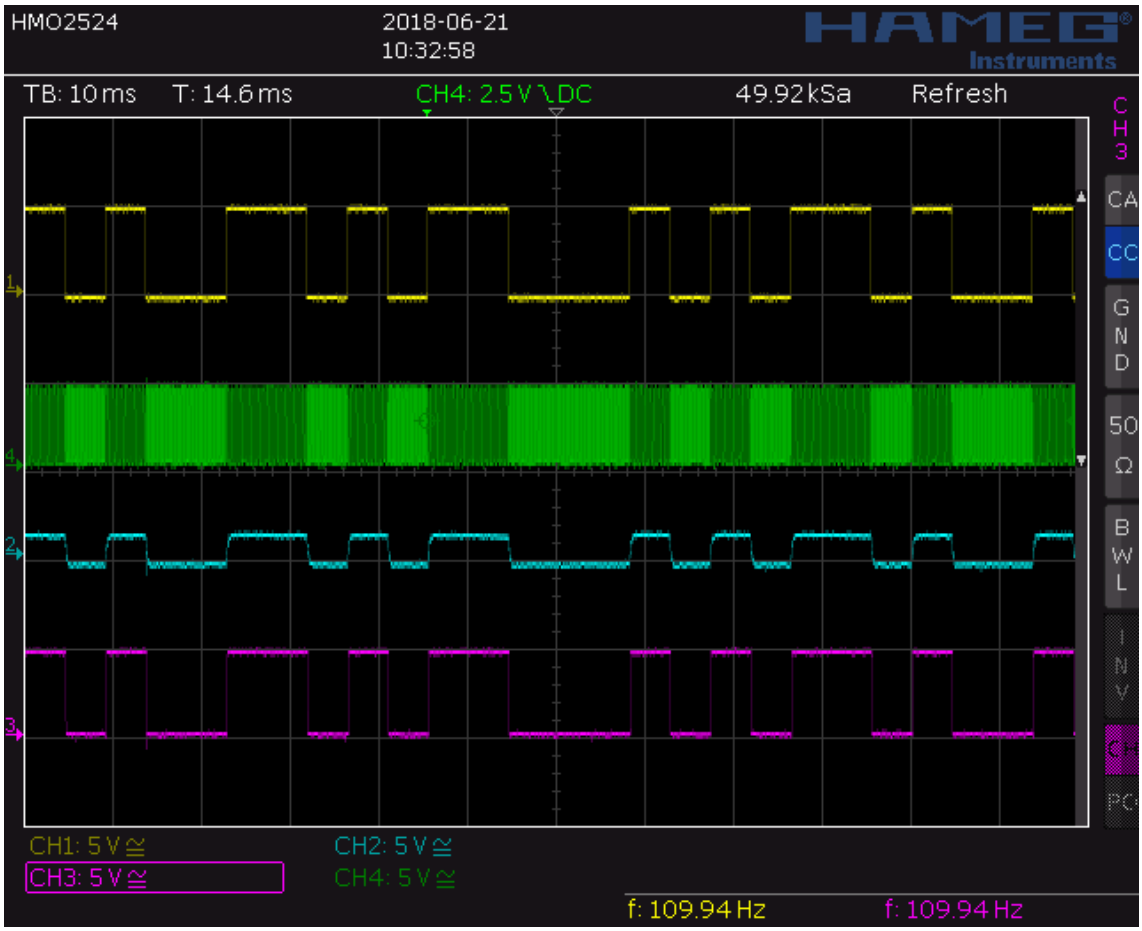


Figura 30. Señales obtenidas en el osciloscopio para una comunicación unidireccional tras modular y demodular la señal Arduino.

- La señal amarilla corresponde a la señal emitida por el Arduino Maestro.
- La señal verde corresponde a la señal modulada, que se puede observar como la frecuencia del nivel lógico 1 es ligeramente mayor que la del nivel lógico 0.
- La señal azul corresponde a la señal demodulada antes de ser amplificada.
- La señal morada corresponde a la señal demodulada amplificada, se puede observar como ahora alcanza los 5 V de amplitud.

El siguiente avance consistió en la implementación de una comunicación bidireccional, donde se recibieron las siguientes señales:

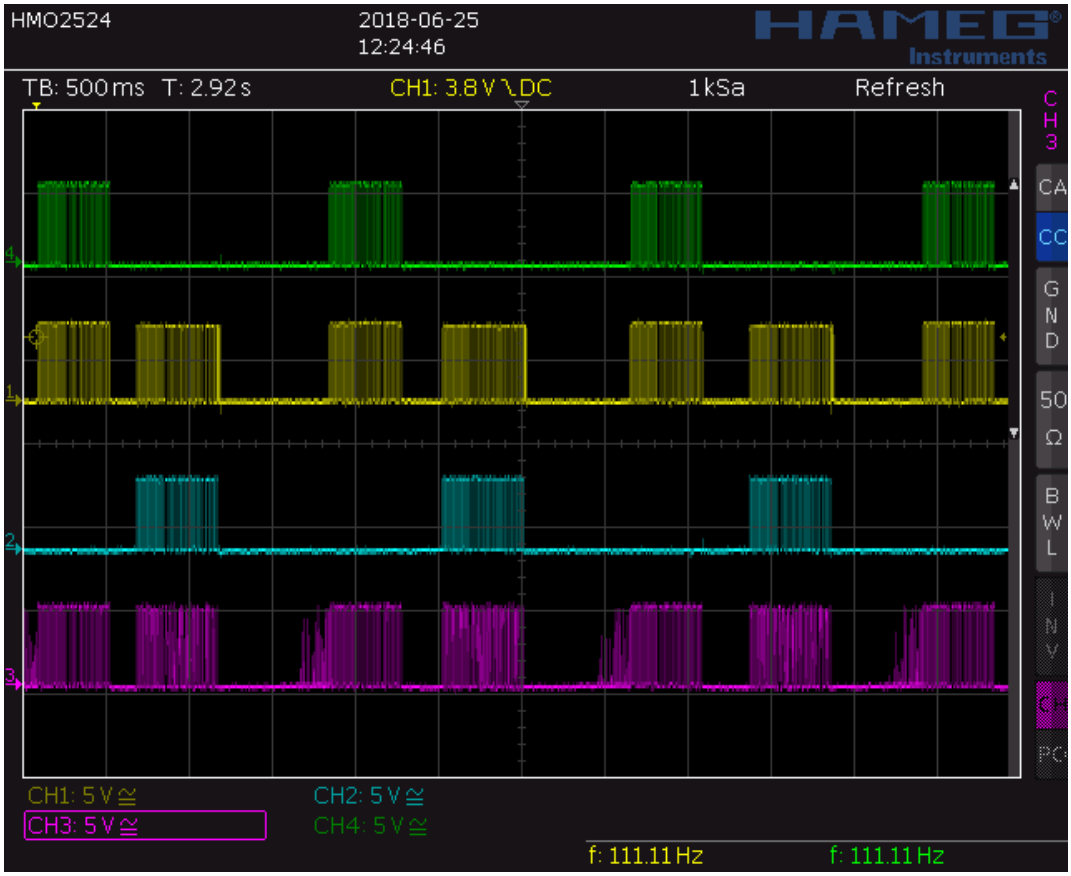


Figura 31. Señales obtenidas en el osciloscopio para una comunicación bidireccional tras modular y demodular la señal Arduino.

- La señal verde corresponde a la señal emitida por el Arduino Maestro; la petición de la temperatura al Esclavo.
- La señal amarilla corresponde a la misma señal, pero tras haber sido modulada, demodulada y amplificada.
- La señal azul corresponde a la señal emitida por el Arduino Esclavo; la contestación con los datos de temperatura al Maestro.
- La señal morada corresponde a la señal emitida por el Esclavo, pero tras haber sido modulada, demodulada y amplificada.

En la *Figura 30* se observan zonas donde no se transmite nada, esto es debido a la activación de la modulación antes de realizar el envío de datos, y a la detención al finalizar.

En las señales de color amarillo y morado se observa una aparente repetición de los mensajes. Esto se debe a que, por ejemplo, aunque sea el Maestro el que transmite el mensaje, él mismo lo recibirá; de manera análoga nos ocurre a las personas, pues cuando hablamos nos escuchamos a nosotros mismos, y a continuación la respuesta recibida.

CAPÍTULO 6. PROBLEMAS Y SOLUCIONES.

Durante el desarrollo de este proyecto han surgido algunos problemas con los módulos de radiofrecuencia y a la hora de implementar el modulador-demodulador FSK. A continuación, se explicarán cómo se han afrontado cada uno de estos.

6.1. Problemas con los módulos inalámbricos.

El primero de ellos fue referido a los módulos de radiofrecuencia. Inicialmente, para establecer una comunicación inalámbrica se eligieron los RF – 433 MHz, donde para que fuese una comunicación bidireccional, como se observa en la *Figura 32,* se tenían que emplear un emisor y un receptor en el Arduino Maestro y de igual forma en el Arduino Esclavo.

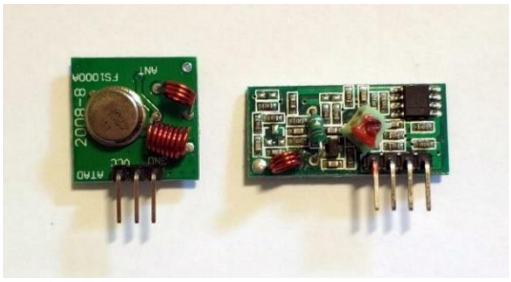


Figura 32. Módulo emisor y receptor de RF 433 – MHz.

Aquí se encontró el primer inconveniente, pues al enviar un dato a través del módulo emisor, el módulo receptor del mismo Arduino también escuchará esta información, provocando que el umbral de escucha aumente mucho y teniendo que esperar un cierto tiempo hasta volver a la normalidad y proceder a la siguiente escucha. Por esta razón, la información transmitida y recibida no se realizaba de forma eficaz, solamente funcionaban a una distancia de separación de medio metro. La solución planteada fueron los módulos de radiofrecuencia NRF2401 que mediante el uso de *pipes* pueden enviar y recibir información desde el mismo módulo, empleando así únicamente dos módulos en vez de cuatro.

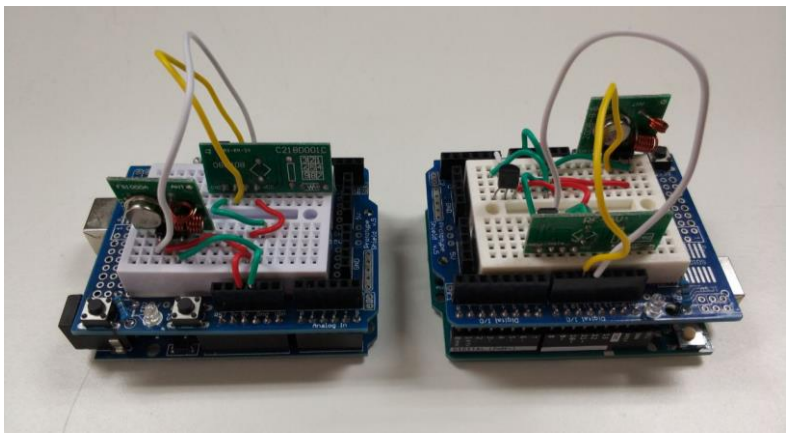


Figura 33. Conexión de los módulos RF – 433 MHz en las placas Arduino.

En segundo lugar, con los nuevos módulos de radiofrecuencia el problema surgió a la hora de alimentarlos a través de la placa Arduino. Estos módulos funcionan con un voltaje entre 1.9 V - 3.6 V, pero la alimentación de Arduino a este voltaje es muy inestable, por lo que se soldó un condensador en sus pines 4 y 5 (alimentación) para que el nivel de tensión fuese más estable.

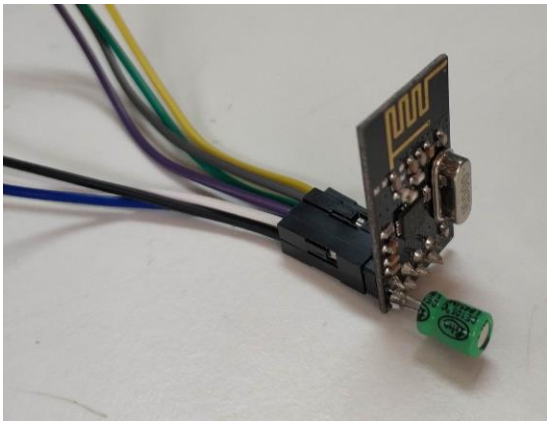


Figura 34. Módulo NRF2401 con condensador soldado en sus pines de alimentación.

6.2. Problemas con el modulador-demodulador FSK.

Primer problema.

A la hora de implementar el modulador-demodulador FSK se comprobó que, a la salida del modulador, la señal no disponía de la tensión necesaria para ser transmitida correctamente hasta la entrada del demodulador, por lo que se ha tenido que usar un *buffer* de voltaje.

Un amplificador *buffer* es un dispositivo electrónico que sirve para hacer una adaptación de impedancias entre circuitos [7]. El circuito integrado empleado ha sido el SN7404N y con él se ha conseguido transferir una señal con voltaje adecuado a la entrada del demodulador.

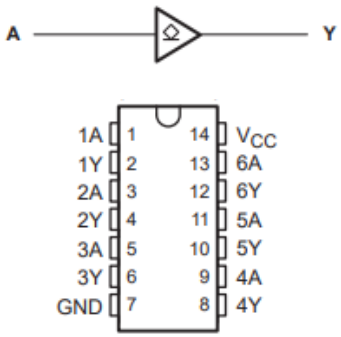


Figura 35. Conexión interna del CI SN7404N.

Segundo problema.

Por otro lado, se observó que la señal a la salida del demodulador salía invertida respecto a la señal original. Esto puede ser causado por el ruido, ya que podría provocar que aparezcan falsos cambios de estado. Además, el voltaje de salida era muy pequeño. Para evitar este problema se ha empleado un disparador de Schmitt, que usa la histéresis para prevenir el ruido.

La histéresis tiende a conservar el nivel lógico en el que se encuentra hasta que se produzca un cambio brusco, por lo que ya el ruido no afectará a nuestro circuito.

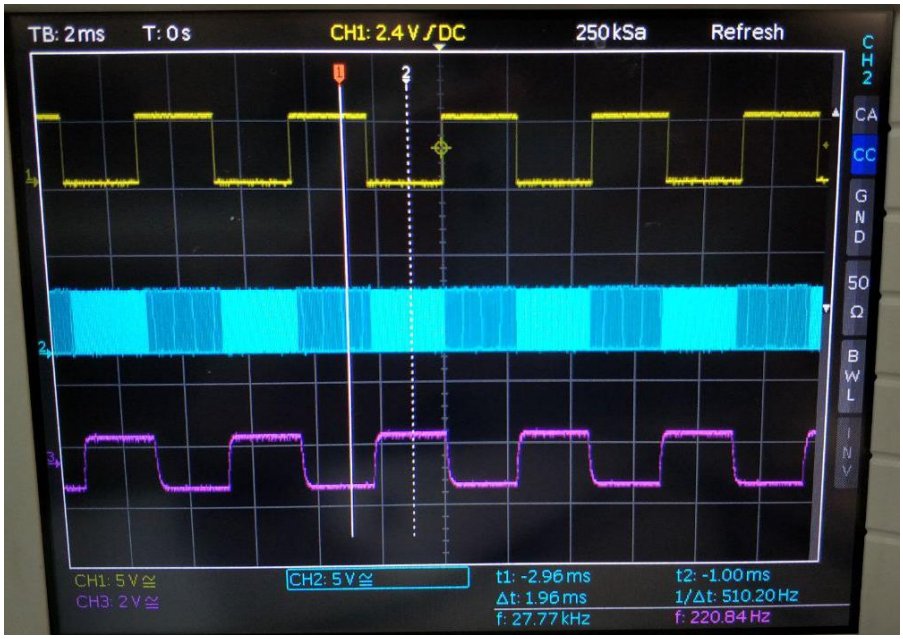


Figura 36. Señales obtenidas antes de usar el disparador de Schmitt.

En la *Figura 35*, se puede observar como la señal demodulada está invertida y ofrece un voltaje pico menor a 2V, es la correspondiente al canal 3 (señal morada). El canal 1 (señal amarilla) corresponde a la señal moduladora con un voltaje pico a pico de 5V, que ha sido simulada con el generador de funciones a 220 Hz, y el canal 2 (señal azul), corresponde a la señal modulada.

Se ha empleado el circuito integrado LM358N que está formado por dos amplificadores operacionales. Con este circuito se reconstruyó la señal y también la invirtió.

Figura 37. Señal original y demodulada haciendo uso del disparador de Schmitt.

En la Figura 36 se observan dos señales, la amarilla corresponde a la señal moduladora, y la verde a la señal demodulada. Se observa como los niveles lógicos de la señal demodulada ya corresponden con los de la señal original, además, aunque no se ha regenerado íntegramente, se ha mejorado mucho el voltaje de la misma.

Tercer problema.

A lo largo del desarrollo de la implementación del modulador-demodulador se presentaron muchos problemas con la regeneración de las señales debido a las altas frecuencias, por ello, aunque al principio se intentó trabajar con 180 kHz de frecuencia central de modulación, se terminó bajando a 100 kHz haciendo uso de otras resistencias y condensadores, porque los operacionales solamente trabajan hasta una determinada frecuencia y se sobrepasaba.

Finalmente se emplearon los circuitos integrados LM311N y TS924IN como comparadores y seguidores de tensión, que ofrecen un rango de trabajo adecuado.

CAPÍTULO 7. PRESUPUESTO.

Este apartado incluye el presupuesto de los materiales empleados en el laboratorio para el desarrollo de este proyecto. Así como el coste de la mano de obra dedicada a la programación, implementación y documentación necesaria para alcanzar el objetivo propuesto.

Por otro lado, se realizará la comparación económica entre un sistema de riego convencional y el sistema inteligente diseñado.

7.1. Presupuesto del proyecto.

7.1.1. Coste de los materiales.

DESCRIPCIÓN	CANTIDAD	COSTE UNITARIO (€/UNIDAD)	COSTE TOTAL (€)
COMUNICACIÓN ALÁMBRICA			
UNO R3 ATmega328P USB CH340G	2	2,78	5,56
Placa PCB Maestro y Esclavo	2	7	14
Circuito Integrado CD4046B	4	1,14	4,56
Circuito Integrado TS924IN	2	1,19	2,38
Circuito Integrado LM311N	3	0,71	2,13
Resistencias	37	0,1	3,7
Condensadores	20	0,2	4
Bobina	2	0,73	1,46
Transformador 750510476	2	5,35	10,7
Cable RJ/NG 2x0,75 mm2	100 metros	0,4/metro	40
Electroválvula	1	6,50	6,50
Sensor de temperatura	1	0,59	0,59
COSTE TOTAL			95,58
COMUNICACIÓN INALÁMBRICA			
Cable para alimentar Arduino *	35 metros	0,4/metro	14
UNO R3 ATmega328P USB CH340G	2	2,78	5,56
Transceptores NRF2401	2	3,57	7,14
Sensor de temperatura	1	0,59	0,59
Electroválvula	1	6,50	6,50
COSTE TOTAL			33,79

Tabla 7. Coste de los materiales [8].

* También existe la opción de alimentar nuestro Arduino y el transceptor haciendo uso de una batería. En este proyecto únicamente se ha desarrollado el sistema de comunicación, no se han hecho pruebas de consumo ni de duración de las baterías. No obstante, se plantea la idea de uso de la misma, pero empleando el modo *sleep* de Arduino, que reducirá notablemente el consumo alargando así su vida útil.

Otra opción a plantear puede ser la instalación de una pequeña placa fotovoltaica que recargue nuestra batería, pudiendo así alimentar cada uno de los Arduinos con sus electroválvulas de forma individual.

7.1.2. Coste de la mano de obra.

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

Tabla 8. Coste de la mano de obra.

7.2. Comparativa económica entre los distintos sistemas de riegos.

En el ejemplo de instalación agrícola convencional desarrollado en los antecedentes de este proyecto, se ha observado que el coste de material, únicamente entre cables y electroválvulas, es de 3955 euros. En cambio, si para esta misma parcela implementamos la idea de sistema de riego inteligente desarrollada en este proyecto, estudiando el presupuesto entre cables y electroválvulas el precio sería de 850 euros. Es decir, un ahorro de 3105 euros del que una parte se puede destinar a la electrónica del sistema.

DESCRIPCIÓN	CANTIDAD	COSTE UNITARIO (€/UNIDAD)	COSTE TOTAL (€)
Cable manguera blanca VVF 2 x 1.5 mm	1000 metros	0.69 €/metro	690
Electroválvula	10	16	160
COSTE TOTAL DE MATERIALES			850

Tabla 9. Presupuesto de la parte electrónica para un sistema de riego inteligente.

En el *Anexo IV* se incluyen los croquis de cada una de las instalaciones.

CAPÍTULO 8. CONCLUSIONES.

En este proyecto se ha llevado a cabo la evaluación de dos tipos de sistemas de comunicación para el riego de un sistema agrícola:

- Un sistema de comunicación alámbrico PLC basado en modulación FSK de bajo coste.
- Un sistema de comunicación inalámbrico haciendo uso de los módulos de radiofrecuencia NRF2401 con antena integrada.

En base a las pruebas de funcionamiento realizadas para cada tipo de comunicación se concluye con que la distancia de alcance para la de tipo alámbrica es de, al menos, 100 metros. En cambio, para la comunicación inalámbrica, sin hacer uso de antena externa en los módulos, es de máximo 35 metros. Este alcance, para terrenos extensos, puede ser insuficiente, y además se debe tener en cuenta que este tipo de comunicación necesita alimentación externa para los módulos como podría ser una pareja de cables o una batería, preferiblemente con una pequeña placa fotovoltaica instalada para su recarga.

Otro aspecto realizado en este proyecto ha sido la elaboración de los presupuestos para cada comunicación, donde la inalámbrica suma una cantidad de 33,79 euros para los 35 metros de alcance haciendo uso de cable para la alimentación de los módulos, y la alámbrica unos 92,44 euros para 100 metros. Por lo que con estos datos se espera que el consumidor valore qué opción se adapta más a sus necesidades.

Finalmente, otro punto importante realizado ha sido la elaboración de una comparativa económica entre los distintos sistemas de riego. Se ha justificado que, frente al sistema convencional, el de riego inteligente ofrece un ahorro de 3105 euros, lo que es una cifra más que considerable.

CONCLUSIONS

In this project has been undertaken the evaluation between two communication systems for the irrigation network of an agricultural facility.

- A wired communication system PLC based on low cost FSK modulation.
- A wireless communication system using the NRF2401 radiofrequency modules with antenna integrated.

Based on the functionality testing for each communication system it is concluded that the range distance for the wired system is, at least, 100 meters. Conversely, for the wireless system, without external antenna, is 35 meters maximum. For extensive land, this range distance will not be sufficient and this type of communication needs an external supply to provide power to all the modules. An external battery with a little photovoltaic panel can be used.

A budget for each communication was also estimated in this project. On the one hand, the wireless system adds an amount of 33,79 euros for 35 meters of range using a cable to supply voltage to the modules. And on the other hand, the budget for a wired system is 92,44 euros for 100 meters. With this information it is expected that the consumer values which option is better for their needs.

Finally, the economic comparison between the different irrigation systems shows that the intelligent design offers a saving of 3105 euros. Part of this amount could be used to buy the electronic devices necessaries to developed the system.

REFERENCIAS Y BIBLIOGRAFÍA

- Información técnica de la fuente de alimentación FAC-363B:

[1] http://taupres.upc.es/labseetac/wp-content/uploads/2017/02/PROMAX-FAC-363B.pdf

- Información sobre Arduino y especificaciones técnicas de Arduino UNO:

[2] https://es.wikipedia.org/wiki/Arduino

[3] https://store.arduino.cc/usa/arduino-uno-rev3

[4] https://aprendiendoarduino.wordpress.com/category/ide/

- Técnica de la modulación:

[5]https://es.wikipedia.org/wiki/Modulaci%C3%B3n_(telecomunicaci%C3%B3n)#Modulaci%C3%B3n_Anal%C3%B3gica

- GFSK:

[6]https://es.wikipedia.org/wiki/Modulaci%C3%B3n_por_desplazamiento_m%C3%ADnimo_gaussiano

- Amplificador *buffer:*

[7] https://es.wikipedia.org/wiki/Amplificador_buffer

- Presupuesto basado en el siguiente distribuidor de componentes electrónicos:

[8] https://www.digikey.es/

ESCUELA SUPERIOR DE INGENIERÍA Y TECNOLOGÍA

SECCIÓN DE INGENIERÍA INDUSTRIAL

ANEXOS

TRABAJO DE FIN DE GRADO

Titulación: *Grado en Ingeniería Electrónica Industrial y Automática*

TÍTULO:

SISTEMA DE RIEGO INTELIGENTE DE BAJO COSTE

AUTORA:

Yaiza Tejera Fumero

Julio, 2018

ÍNDICE DE LOS ANEXOS

ANEXO I: Datasheets 67
Anexo I.1: CD4046B 67
Anexo I.2: LM35 71
Anexo I.3: LM311N 74
Anexo I.4: TS924IN 76
ANEXO II: Código implementado. 83
Anexo II.1: Comunicación alámbrica. 83
II.1.1: Código Maestro. 83
II.1.2: Código Esclavo. 85
Anexo II.2: Comunicación inalámbrica. 87
II.2.1: Código Maestro. 87
II.2.2: Código Esclavo. 89
ANEXO III: Esquemas y conexiones 91
Anexo III.1: Comunicación bidireccional. 91
ANEXO IV: Croquis de las instalaciones agrícolas 92
Anexo IV.1: Instalación agrícola convencional. 92
Anexo IV.2: Instalación agrícola inteligente. 93

ANEXO I: Datasheets

Anexo I.1: CD4046B

FAIRCHILD SEMICONDUCTOR™

October 1987
Revised March 2002

CD4046BC
Micropower Phase-Locked Loop

General Description

The CD4046BC micropower phase-locked loop (PLL) consists of a low power, linear, voltage-controlled oscillator (VCO), a source follower, a zener diode, and two phase comparators. The two phase comparators have a common signal input and a common comparator input. The signal input can be directly coupled for a large voltage signal, or capacitively coupled to the self-biasing amplifier at the signal input for a small voltage signal.

Phase comparator I, an exclusive OR gate, provides a digital error signal (phase comp. I Out) and maintains 90° phase shifts at the VCO center frequency. Between signal input and comparator input (both at 50% duty cycle), it may lock onto the signal input frequencies that are close to harmonics of the VCO center frequency.

Phase comparator II is an edge-controlled digital memory network. It provides a digital error signal (phase comp. II Out) and lock-in signal (phase pulses) to indicate a locked condition and maintains a 0° phase shift between signal input and comparator input.

The linear voltage-controlled oscillator (VCO) produces an output signal (VCO Out) whose frequency is determined by the voltage at the VCO_{IN} input, and the capacitor and resistors connected to pin $C1_A$, $C1_B$, R1 and R2.

The source follower output of the VCO_{IN} (demodulator Out) is used with an external resistor of 10 kΩ or more.

The INHIBIT input, when high, disables the VCO and source follower to minimize standby power consumption. The zener diode is provided for power supply regulation, if necessary.

Features

- Wide supply voltage range: 3.0V to 18V
- Low dynamic power consumption: 70 µW (typ.) at f_o = 10 kHz, V_{DD} = 5V
- VCO frequency: 1.3 MHz (typ.) at V_{DD} = 10V
- Low frequency drift: 0.06%/°C at V_{DD} = 10V with temperature
- High VCO linearity: 1% (typ.)

Applications

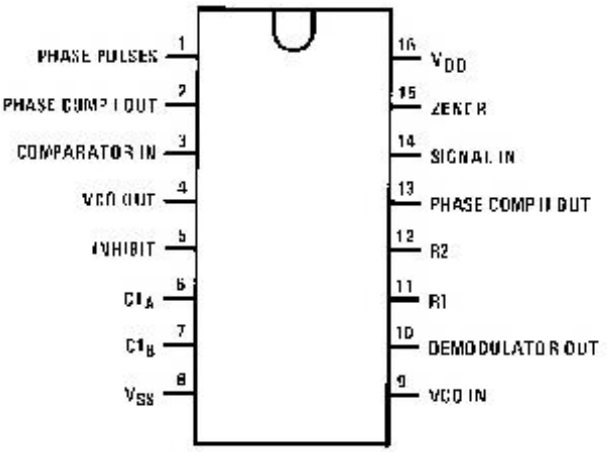
- FM demodulator and modulator
- Frequency synthesis and multiplication
- Frequency discrimination
- Data synchronization and conditioning
- Voltage-to-frequency conversion
- Tone decoding
- FSK modulation
- Motor speed control

Ordering Code:

Order Number	Package Number	Package Description
CD4046BCM	M16A	16-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-012, 0.150" Narrow
CD4046BCN	N16E	16-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300" Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Top View

Block Diagram

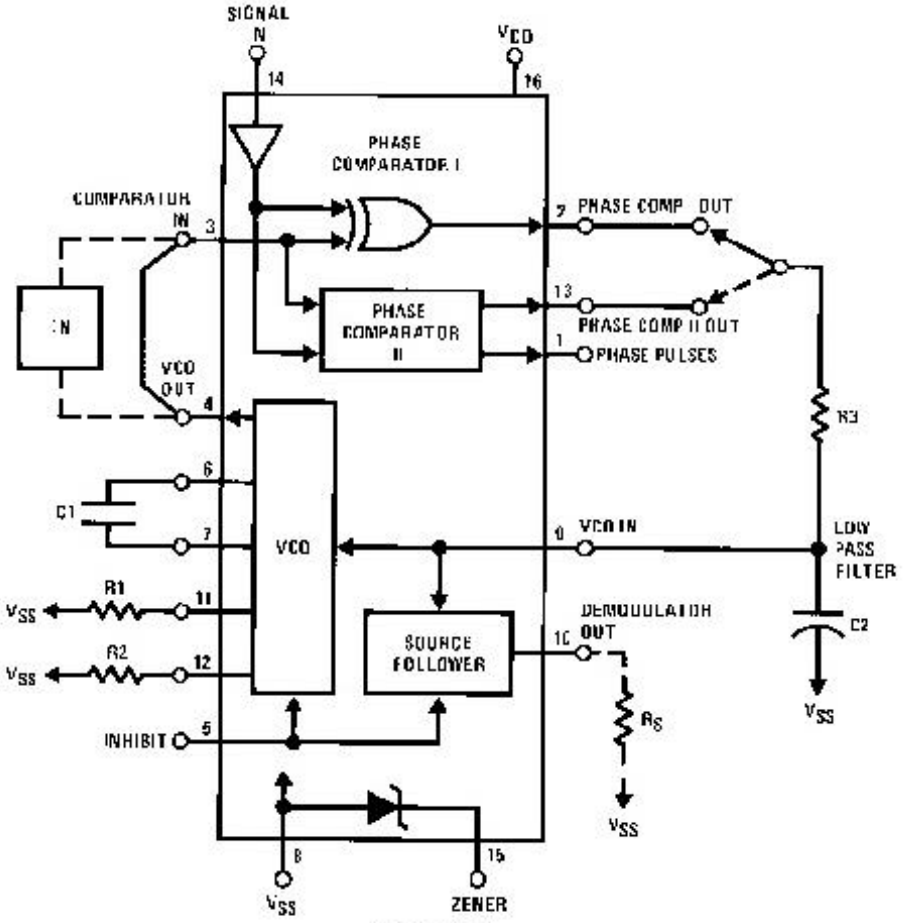


FIGURE 1.

Absolute Maximum Ratings (Note 1)

(Note 2)

DC Supply Voltage (V_{DD})	−0.5 to +18 V_{DC}
Input Voltage (V_{IN})	−0.5 to V_{DD} +0.5 V_{DC}
Storage Temperature Range (T_S)	−65°C to +150°C
Power Dissipation (P_D)	
Dual-In-Line	700 mW
Small Outline	500 mW
Lead Temperature (T_L)	
(Soldering, 10 seconds)	260°C

Recommended Operating Conditions (Note 2)

DC Supply Voltage (V_{DD})	3 to 15 V_{DC}
Input Voltage (V_{IN})	0 to V_{DD} V_{DC}
Operating Temperature Range (T_A)	−55°C to +125°C

Note 1: "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. They are not meant to imply that the devices should be operated at these limits. The table of "Recommended Operating Conditions" and "Electrical Characteristics" provides conditions for actual device operation.

Note 2: V_{SS} = 0V unless otherwise specified.

DC Electrical Characteristics (Note 2)

Symbol	Parameter	Conditions	−55°C		+25°C			+125°C		Units
			Min	Max	Min	Typ	Max	Min	Max	
I_{DD}	Quiescent Device Current	Pin 5 = V_{DD}, Pin 14 = V_{DD}, Pin 3, 9 = V_{SS}								
		V_{DD} = 5V		5		0.005	5		150	
		V_{DD} = 10V		10		0.01	10		300	µA
		V_{DD} = 15V		20		0.015	20		600	
		Pin 5 = V_{DD}, Pin 14 = Open, Pin 3, 9 = V_{SS}								
		V_{DD} = 5V		45		5	35		185	
		V_{DD} = 10V		450		20	350		650	µA
		V_{DD} = 15V		1200		50	900		1500	
V_{OL}	LOW Level Output Voltage	V_{DD} = 5V		0.05		0	0.05		0.05	
		V_{DD} = 10V		0.05		0	0.05		0.05	V
		V_{DD} = 15V		0.05		0	0.05		0.05	
V_{OH}	HIGH Level Output Voltage	V_{DD} = 5V	4.95		4.95	5		4.95		
		V_{DD} = 10V	9.95		9.95	10		9.95		V
		V_{DD} = 15V	14.95		14.95	15		14.95		
V_{IL}	LOW Level Input Voltage	V_{DD} = 5V, V_O = 0.5V or 4.5V		1.5		2.25	1.5		1.5	
	Comparator and Signal In	V_{DD} = 10V, V_O = 1V or 9V		3.0		4.5	3.0		3.0	V
		V_{DD} = 15V, V_O = 1.5V or 13.5V		4.0		6.25	4.0		4.0	
V_{IH}	HIGH Level Input Voltage	V_{DD} = 5V, V_O = 0.5V or 4.5V	3.5		3.5	2.75		3.5		
	Comparator and Signal In	V_{DD} = 10V, V_O = 1V or 9V	7.0		7.0	5.5		7.0		V
		V_{DD} = 15V, V_O = 1.5V or 13.5V	11.0		11.0	8.25		11.0		
I_{OL}	LOW Level Output Current	V_{DD} = 5V, V_O = 0.4V	0.64		0.51	0.88		0.36		
	(Note 4)	V_{DD} = 10V, V_O = 0.5V	1.6		1.3	2.25		0.9		mA
		V_{DD} = 15V, V_O = 1.5V	4.2		3.4	8.8		2.4		
I_{OH}	HIGH Level Output Current	V_{DD} = 5V, V_O = 4.6V	−0.64		−0.51	−0.88		−0.36		
	(Note 4)	V_{DD} = 10V, V_O = 9.5V	−1.6		−1.3	−2.25		−0.9		mA
		V_{DD} = 15V, V_O = 13.5V	−4.2		−3.4	−8.8		−2.4		
I_{IN}	Input Current	All Inputs Except Signal Input								
		V_{DD} = 15V, V_{IN} = 0V		−0.1		-10^{-5}	−0.1		−1.0	µA
		V_{DD} = 15V, V_{IN} = 15V		0.1		10^{-5}	0.1		1.0	
C_{IN}	Input Capacitance	Any Input (Note 3)							7.5	pF
P_T	Total Power Dissipation	f_o = 10 kHz, R1 = 1 MΩ, R2 = ∞, VCO_{IN} = V_{CC}/2								
		V_{DD} = 5V				0.07				
		V_{DD} = 10V				0.6				mW
		V_{DD} = 15V				2.4				

Note 3: Capacitance is guaranteed by periodic testing.

Note 4: I_{OH} and I_{OL} are tested one output at a time.

AC Electrical Characteristics (Note 5)

$T_A = 25°C$, $C_L = 50\,pF$

Symbol	Parameter	Conditions	Min	Typ	Max	Units
VCO SECTION						
I_{DD}	Operating Current	$f_o = 10$ kHz, R1 = 1 MΩ, R2 = ∞, $VCO_{IN} = V_{CC}/2$				
		$V_{DD} = 5V$		20		µA
		$V_{DD} = 10V$		90		
		$V_{DD} = 15V$		200		
f_{MAX}	Maximum Operating Frequency	C1 = 50 pF, R1 = 10 kΩ, R2 = ∞, $VCO_{IN} = V_{DD}$				
		$V_{DD} = 5V$	0.4	0.8		MHz
		$V_{DD} = 10V$	0.6	1.2		
		$V_{DD} = 15V$	1.0	1.6		
	Linearity	$VCO_{IN} = 2.5V \pm 0.3V$, R1 ≥ 10 kΩ, $V_{DD} = 5V$		1		%
		$VCO_{IN} = 5V \pm 2.5V$, R1 ≥ 400 kΩ, $V_{DD} = 10V$		1		
		$VCO_{IN} = 7.5V \pm 5V$, R1 ≥ 1 MΩ, $V_{DD} = 15V$		1		
	Temperature-Frequency Stability	%/°C < 5c1/f. V_{DD}				
	No Frequency Offset, $f_{MIN} = 0$	R2 = ∞				
		$V_{DD} = 5V$		0.12–0.24		%/°C
		$V_{DD} = 10V$		0.04–0.08		
		$V_{DD} = 15V$		0.015–0.03		
	Frequency Offset, $f_{MIN} \neq 0$	$V_{DD} = 5V$		0.06–0.12		%/°C
		$V_{DD} = 10V$		0.05–0.1		
		$V_{DD} = 15V$		0.03–0.06		
VCO_{IN}	Input Resistance	$V_{DD} = 5V$		10^6		MΩ
		$V_{DD} = 10V$		10^6		
		$V_{DD} = 15V$		10^6		
VCO	Output Duty Cycle	$V_{DD} = 5V$		50		%
		$V_{DD} = 10V$		50		
		$V_{DD} = 15V$		50		
t_{THL}	VCO Output Transition Time	$V_{DD} = 5V$		90	200	ns
t_{THL}		$V_{DD} = 10V$		50	100	ns
		$V_{DD} = 15V$		45	80	
PHASE COMPARATORS SECTION						
R_{IN}	Input Resistance					
	Signal Input	$V_{DD} = 5V$	1	3		MΩ
		$V_{DD} = 10V$	0.2	0.7		
		$V_{DD} = 15V$	0.1	0.3		
	Comparator Input	$V_{DD} = 5V$		10^6		
		$V_{DD} = 10V$		10^6		
		$V_{DD} = 15V$		10^6		
	AC-Coupled Signal Input Voltage Sensitivity	$C_{SERIES} = 1000$ pF f = 50 kHz				
		$V_{DD} = 5V$		200	400	mV
		$V_{DD} = 10V$		400	800	
		$V_{DD} = 15V$		700	1400	
DEMODULATOR OUTPUT						

Anexo I.2: LM35

TEXAS INSTRUMENTS

LM35

SNIS159H – AUGUST 1999 – REVISED DECEMBER 2017

LM35 Precision Centigrade Temperature Sensors

1 Features

- Calibrated Directly in Celsius (Centigrade)
- Linear + 10-mV/°C Scale Factor
- 0.5°C Ensured Accuracy (at 25°C)
- Rated for Full −55°C to 150°C Range
- Suitable for Remote Applications
- Low-Cost Due to Wafer-Level Trimming
- Operates From 4 V to 30 V
- Less Than 60-μA Current Drain
- Low Self-Heating, 0.08°C in Still Air
- Non-Linearity Only ±¼°C Typical
- Low-Impedance Output, 0.1 Ω for 1-mA Load

2 Applications

- Power Supplies
- Battery Management
- HVAC
- Appliances

3 Description

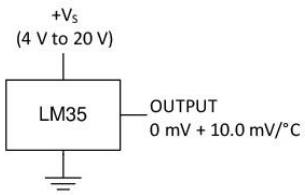
The LM35 series are precision integrated-circuit temperature devices with an output voltage linearly-proportional to the Centigrade temperature. The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 device does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55°C to 150°C temperature range. Lower cost is assured by trimming and calibration at the wafer level. The low-output impedance, linear output, and precise inherent calibration of the LM35 device makes interfacing to readout or control circuitry especially easy. The device is used with single power supplies, or with plus and minus supplies. As the LM35 device draws only 60 μA from the supply, it has very low self-heating of less than 0.1°C in still air. The LM35 device is rated to operate over a −55°C to 150°C temperature range, while the LM35C device is rated for a −40°C to 110°C range (−10° with improved accuracy). The LM35-series devices are available packaged in hermetic TO transistor packages, while the LM35C, LM35CA, and LM35D devices are available in the plastic TO-92 transistor package. The LM35D device is available in an 8-lead surface-mount small-outline package and a plastic TO-220 package.

Device Information(1)

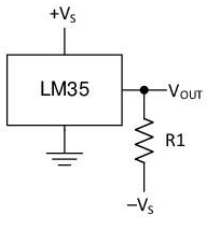
PART NUMBER	PACKAGE	BODY SIZE (NOM)
LM35	TO-CAN (3)	4.699 mm × 4.699 mm
	TO-92 (3)	4.30 mm × 4.30 mm
	SOIC (8)	4.90 mm × 3.91 mm
	TO-220 (3)	14.986 mm × 10.16 mm

(1) For all available packages, see the orderable addendum at the end of the datasheet.

Basic Centigrade Temperature Sensor (2°C to 150°C)



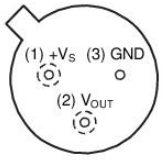
Full-Range Centigrade Temperature Sensor



Choose $R_1 = -V_S / 50\ \mu A$
$V_{OUT} = 1500$ mV at 150°C
$V_{OUT} = 250$ mV at 25°C
$V_{OUT} = -550$ mV at −55°C

5 Pin Configuration and Functions

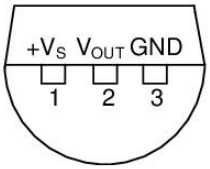
NDV Package
3-Pin TO-CAN
(Top View)



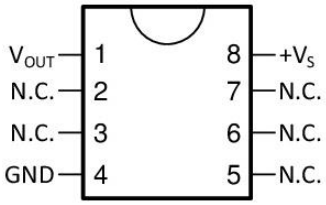
Case is connected to negative pin (GND)

Refer the second NDV0003H page for reference

LP Package
3-Pin TO-92
(Bottom View)

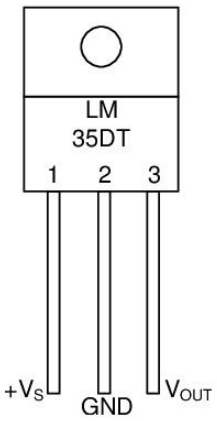


D Package
8-PIN SOIC
(Top View)



N.C. = No connection

NEB Package
3-Pin TO-220
(Top View)



Tab is connected to the negative pin (GND).

NOTE: The LM35DT pinout is different than the discontinued LM35DP

Pin Functions

PIN					TYPE	DESCRIPTION
NAME	TO46	TO92	TO220	SO8		
V_{OUT}	2	2	3	1	O	Temperature Sensor Analog Output
N.C.	—	—	—	2	—	No Connection
	—	—	—	3		
GND	3	3	2	4	GROUND	Device ground pin, connect to power supply negative terminal
N.C.	—	—	—	5	—	No Connection
	—	—	—	6		
	—	—	—	7		
$+V_S$	1	1	1	8	POWER	Positive power supply pin

6.7 Electrical Characteristics: LM35, LM35C, LM35D Limits

Unless otherwise noted, these specifications apply: −55°C ≤ T_J ≤ 150°C for the LM35 and LM35A; −40°C ≤ T_J ≤ 110°C for the LM35C and LM35CA; and 0°C ≤ T_J ≤ 100°C for the LM35D. V_S = 5 Vdc and I_{LOAD} = 50 μA, in the circuit of Full-Range Centigrade Temperature Sensor. These specifications also apply from 2°C to T_{MAX} in the circuit of Figure 14.

PARAMETER	TEST CONDITIONS	LM35			LM35C, LM35D			UNIT
		TYP	TESTED LIMIT(1)	DESIGN LIMIT(2)	TYP	TESTED LIMIT(1)	DESIGN LIMIT(2)	
Accuracy, LM35, LM35C(3)	T_A = 25°C	±0.4	±1		±0.4	±1		°C
	T_A = −10°C	±0.5			±0.5		±1.5	
	T_A = T_{MAX}	±0.8	±1.5		±0.8		±1.5	
	T_A = T_{MIN}	±0.8		±1.5	±0.8		±2	
Accuracy, LM35D(3)	T_A = 25°C				±0.6	±1.5		°C
	T_A = T_{MAX}				±0.9		±2	
	T_A = T_{MIN}				±0.9		±2	
Nonlinearity(4)	T_{MIN} ≤ T_A ≤ T_{MAX}, −40°C ≤ T_J ≤ 125°C	±0.3		±0.5	±0.2		±0.5	°C
Sensor gain (average slope)	T_{MIN} ≤ T_A ≤ T_{MAX}, −40°C ≤ T_J ≤ 125°C	10	9.8		10		9.8	mV/°C
		10	10.2		10		10.2	
Load regulation(5) 0 ≤ I_L ≤ 1 mA	T_A = 25°C	±0.4	±2		±0.4	±2		mV/mA
	T_{MIN} ≤ T_A ≤ T_{MAX}, −40°C ≤ T_J ≤ 125°C	±0.5		±5	±0.5		±5	
Line regulation(5)	T_A = 25°C	±0.01	±0.1		±0.01	±0.1		mV/V
	4 V ≤ V_S ≤ 30 V, −40°C ≤ T_J ≤ 125°C	±0.02		±0.2	±0.02		±0.2	
Quiescent current(6)	V_S = 5 V, 25°C	56	80		56	80		μA
	V_S = 5 V, −40°C ≤ T_J ≤ 125°C	105		158	91		138	
	V_S = 30 V, 25°C	56.2	82		56.2	82		
	V_S = 30 V, −40°C ≤ T_J ≤ 125°C	105.5		161	91.5		141	
Change of quiescent current(5)	4 V ≤ V_S ≤ 30 V, 25°C	0.2	2		0.2	2		μA
	4 V ≤ V_S ≤ 30 V, −40°C ≤ T_J ≤ 125°C	0.5		3	0.5		3	
Temperature coefficient of quiescent current	−40°C ≤ T_J ≤ 125°C	0.39		0.7	0.39		0.7	μA/°C
Minimum temperature for rate accuracy	In circuit of Figure 14, I_L = 0	1.5		2	1.5		2	°C
Long term stability	T_J = T_{MAX}, for 1000 hours	±0.08			±0.08			°C

(1) Tested Limits are ensured and 100% tested in production.
(2) Design Limits are ensured (but not 100% production tested) over the indicated temperature and supply voltage ranges. These limits are not used to calculate outgoing quality levels.
(3) Accuracy is defined as the error between the output voltage and 10 mv/°C times the case temperature of the device, at specified conditions of voltage, current, and temperature (expressed in °C).
(4) Non-linearity is defined as the deviation of the output-voltage-versus-temperature curve from the best-fit straight line, over the rated temperature range of the device.
(5) Regulation is measured at constant junction temperature, using pulse testing with a low duty cycle. Changes in output due to heating effects can be computed by multiplying the internal dissipation by the thermal resistance.
(6) Quiescent current is defined in the circuit of Figure 14.

Anexo I.3: LM311N

LM111-N/LM211-N/LM311-N Voltage Comparator

Check for Samples: LM111-N, LM211-N, LM311-N

FEATURES

- **Operates From Single 5V Supply**
- **Input Current: 150 nA Max. Over Temperature**
- **Offset Current: 20 nA Max. Over Temperature**
- **Differential Input Voltage Range: ±30V**
- **Power Consumption: 135 mW at ±15V**

DESCRIPTION

The LM111-N, LM211-N and LM311-N are voltage comparators that have input currents nearly a thousand times lower than devices like the LM106 or LM710. They are also designed to operate over a wider range of supply voltages: from standard ±15V op amp supplies down to the single 5V supply used for IC logic. Their output is compatible with RTL, DTL and TTL as well as MOS circuits. Further, they can drive lamps or relays, switching voltages up to 50V at currents as high as 50 mA.

Both the inputs and the outputs of the LM111-N, LM211-N or the LM311-N can be isolated from system ground, and the output can drive loads referred to ground, the positive supply or the negative supply. Offset balancing and strobe capability are provided and outputs can be wire OR'ed. Although slower than the LM106 and LM710 (200 ns response time vs 40 ns) the devices are also much less prone to spurious oscillations. The LM111-N has the same pin configuration as the LM106 and LM710.

The LM211-N is identical to the LM111-N, except that its performance is specified over a −25°C to +85°C temperature range instead of −55°C to +125°C. The LM311-N has a temperature range of 0°C to +70°C.

Typical Applications

NOTE

Pin connections shown in Schematic Diagram and Typical Applications are for the LMC TO-99 package.

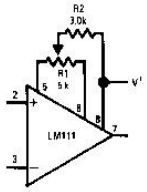
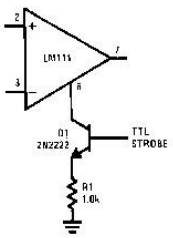


Figure 1. Offset Balancing



Do Not Ground Strobe Pin. Output is turned off when current is pulled from Strobe Pin.

Figure 2. Strobing

Absolute Maximum Ratings for the LM311-N (1) (2)

Total Supply Voltage (V_{84})			36V
Output to Negative Supply Voltage (V_{74})			40V
Ground to Negative Supply Voltage (V_{14})			30V
Differential Input Voltage			±30V
Input Voltage (3)			±15V
Power Dissipation (4)			500 mW
ESD Rating (5)			300V
Output Short Circuit Duration			10 sec
Operating Temperature Range			0° to 70°C
Storage Temperature Range			−65°C to 150°C
Lead Temperature (soldering, 10 sec)			260°C
Voltage at Strobe Pin			V^+−5V
Soldering Information	Dual-In-Line Package	Soldering (10 seconds)	260°C
	Small Outline Package	Vapor Phase (60 seconds)	215°C
		Infrared (15 seconds)	220°C

(1) "Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. Operating Ratings indicate conditions for which the device is functional, but do not ensure specific performance limits."
(2) If Military/Aerospace specified devices are required, please contact the Texas Instruments Sales Office/ Distributors for availability and specifications.
(3) This rating applies for ±15V supplies. The positive input voltage limit is 30V above the negative supply. The negative input voltage limit is equal to the negative supply voltage or 30V below the positive supply, whichever is less.
(4) The maximum junction temperature of the LM311-N is 110°C. For operating at elevated temperature, devices in the LMC package must be derated based on a thermal resistance of 165°C/W, junction to ambient, or 20°C/W, junction to case. The thermal resistance of the dual-in-line package is 100°C/W, junction to ambient.
(5) Human body model, 1.5 kΩ in series with 100 pF.

Electrical Characteristics (1) for the LM311-N

Parameter	Conditions	Min	Typ	Max	Units
Input Offset Voltage (2)	T_A=25°C, R_S≤50k		2.0	7.5	mV
Input Offset Current (2)	T_A=25°C		6.0	50	nA
Input Bias Current	T_A=25°C		100	250	nA
Voltage Gain	T_A=25°C	40	200		V/mV
Response Time (3)	T_A=25°C		200		ns
Saturation Voltage	V_{IN}≤−10 mV, I_{OUT}=50 mA , T_A=25°C		0.75	1.5	V
Strobe ON Current (4)	T_A=25°C		2.0	5.0	mA
Output Leakage Current	V_{IN}≥10 mV, V_{OUT}=35V T_A=25°C, I_{STROBE}=3 mA V^- = Pin 1 = −5V		0.2	50	nA
Input Offset Voltage (2)	R_S≤50K			10	mV
Input Offset Current (2)				70	nA
Input Bias Current				300	nA
Input Voltage Range		−14.5	13.8,−14.7	13.0	V
Saturation Voltage	V^+≥4.5V, V^-=0, V_{IN}≤−10 mV, I_{OUT}≤8 mA		0.23	0.4	V
Positive Supply Current	T_A=25°C		5.1	7.5	mA
Negative Supply Current	T_A=25°C		4.1	5.0	mA

(1) These specifications apply for V_S=±15V and Pin 1 at ground, and 0°C < T_A < +70°C, unless otherwise specified. The offset voltage, offset current and bias current specifications apply for any supply voltage from a single 5V supply up to ±15V supplies.
(2) The offset voltages and offset currents given are the maximum values required to drive the output within a volt of either supply with 1 mA load. Thus, these parameters define an error band and take into account the worst-case effects of voltage gain and R_S.
(3) The response time specified (see definitions) is for a 100 mV input step with 5 mV overdrive.
(4) This specification gives the range of current which must be drawn from the strobe pin to ensure the output is properly disabled. Do not short the strobe pin to ground; it should be current driven at 3 to 5 mA.

Anexo I.4: NRF2401

NORDIC SEMICONDUCTOR

nRF24L01
Single Chip 2.4GHz Transceiver

Product Specification

Key Features

- Worldwide 2.4GHz ISM band operation
- Up to 2Mbps on air data rate
- Ultra low power operation
- 11.3mA TX at 0dBm output power
- 12.3mA RX at 2Mbps air data rate
- 900nA in power down
- 22µA in standby-I
- On chip voltage regulator
- 1.9 to 3.6V supply range
- Enhanced ShockBurst™
- Automatic packet handling
- Auto packet transaction handling
- 6 data pipe MultiCeiver™
- Air compatible with nRF2401A, 02, E1 and E2
- Low cost BOM
- ±60ppm 16MHz crystal
- 5V tolerant inputs
- Compact 20-pin 4x4mm QFN package

Applications

- Wireless PC Peripherals
- Mouse, keyboards and remotes
- 3-in-one desktop bundles
- Advanced Media center remote controls
- VoIP headsets
- Game controllers
- Sports watches and sensors
- RF remote controls for consumer electronics
- Home and commercial automation
- Ultra low power sensor networks
- Active RFID
- Asset tracing systems
- Toys

July 2007

1.1 Features

Features of the nRF24L01 include:

- Radio
  - Worldwide 2.4GHz ISM band operation
  - 126 RF channels
  - Common RX and TX pins
  - GFSK modulation
  - 1 and 2Mbps air data rate
  - 1MHz non-overlapping channel spacing at 1Mbps
  - 2MHz non-overlapping channel spacing at 2Mbps
- Transmitter
  - Programmable output power: 0, -6, -12 or -18dBm
  - 11.3mA at 0dBm output power
- Receiver
  - Integrated channel filters
  - 12.3mA at 2Mbps
  - -82dBm sensitivity at 2Mbps
  - -85dBm sensitivity at 1Mbps
  - Programmable LNA gain
- RF Synthesizer
  - Fully integrated synthesizer
  - No external loop filer, VCO varactor diode or resonator
  - Accepts low cost ±60ppm 16MHz crystal
- Enhanced ShockBurst™
  - 1 to 32 bytes dynamic payload length
  - Automatic packet handling
  - Auto packet transaction handling
  - 6 data pipe MultiCeiver™ for 1:6 star networks
- Power Management
  - Integrated voltage regulator
  - 1.9 to 3.6V supply range
  - Idle modes with fast start-up times for advanced power management
  - 22uA Standby-I mode, 900nA power down mode
  - Max 1.5ms start-up from power down mode
  - Max 130us start-up from standby-I mode
- Host Interface
  - 4-pin hardware SPI
  - Max 8Mbps
  - 3 separate 32 bytes TX and RX FIFOs
  - 5V tolerant inputs
- Compact 20-pin 4x4mm QFN package

2 Pin Information

2.1 Pin assignment

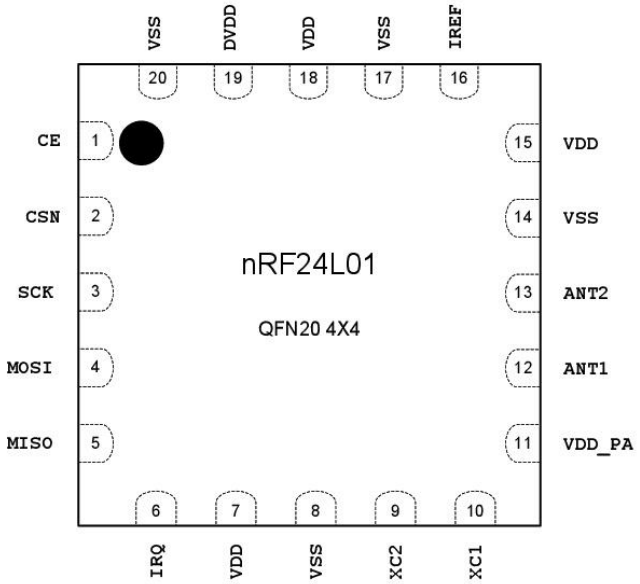


Figure 2. nRF24L01 pin assignment (top view) for the QFN20 4x4 package

2.2 Pin functions

Pin	Name	Pin function	Description
1	CE	Digital Input	Chip Enable Activates RX or TX mode
2	CSN	Digital Input	SPI Chip Select
3	SCK	Digital Input	SPI Clock
4	MOSI	Digital Input	SPI Slave Data Input
5	MISO	Digital Output	SPI Slave Data Output, with tri-state option
6	IRQ	Digital Output	Maskable interrupt pin. Active low
7	VDD	Power	Power Supply (+1.9V - +3.6V DC)
8	VSS	Power	Ground (0V)
9	XC2	Analog Output	Crystal Pin 2
10	XC1	Analog Input	Crystal Pin 1
11	VDD_PA	Power Output	Power Supply Output(+1.8V) for the internal nRF24L01 Power Amplifier. Must be connected to ANT1 and ANT2 as shown in Figure 30.
12	ANT1	RF	Antenna interface 1
13	ANT2	RF	Antenna interface 2
14	VSS	Power	Ground (0V)
15	VDD	Power	Power Supply (+1.9V - +3.6V DC)
16	IREF	Analog Input	Reference current. Connect a 22kΩ resistor to ground. See: Figure 30.
17	VSS	Power	Ground (0V)
18	VDD	Power	Power Supply (+1.9V - +3.6V DC)
19	DVDD	Power Output	Internal digital supply output for de-coupling purposes. See: Figure 30.
20	VSS	Power	Ground (0V)

Table 1. nRF24L01 pin function

4 Operating conditions

Symbol	Parameter (condition)	Notes	Min.	Typ.	Max.	Units
VDD	Supply voltage		1.9	3.0	3.6	V
VDD	Supply voltage if input signals >3.6V		2.7	3.0	3.3	V
TEMP	Operating Temperature		-40	+27	+85	°C

Table 3. Operating conditions

Anexo I.5: TS924IN

life.augmented

TS924, TS924A

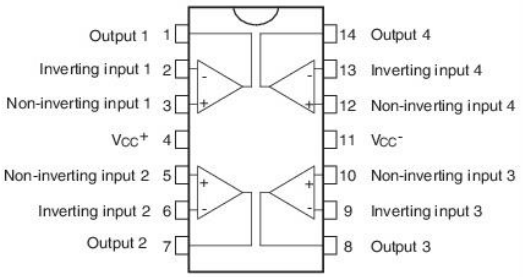
Rail-to-rail output current quad operational amplifier

Datasheet - production data

D
SO14
(plastic micropackage)

P
TSSOP14
(thin shrink small outline package)

Pin connections (top view)



Features

- Rail-to-rail input and output
- Low noise: 9 nV/√Hz
- Low distortion
- High output current: 80 mA (able to drive 32 Ω loads)
- High-speed: 4 MHz, 1.3 V/µs
- Operating range from 2.7 V to 12 V
- Low input offset voltage: 900 µV max. (TS924A)
- ESD internal protection: 3 kV
- Latch-up immunity
- Macromodel included in this specification

Related products

- See the TS921 device for the single version and the TS922 device for the dual version
- See the TSX56x series for smaller packages

Applications

- Headphone amplifiers
- Piezoelectric speaker drivers
- Sound cards
- MPEG boards, multimedia systems
- Line drivers, buffers
- Cordless telephones and portable communication equipment
- Instrumentation with low noise as key factor

Description

The TS924 and TS924A devices are rail-to-rail quad BiCMOS operational amplifiers optimized and fully specified for 3 V and 5 V operation.

High output current allows low load impedances to be driven.

The TS924 and TS924A devices exhibit a very low noise, low distortion, low offset, and high output current capability, making these devices an excellent choice for high-quality, low-voltage, and battery-operated audio systems.

The devices are stable for capacitive loads up to 500 pF.

1 Absolute maximum ratings and operating conditions

Table 1. Absolute maximum ratings

Symbol	Parameter	Value	Unit
V_{CC}	Supply voltage(1)	14	V
V_{id}	Differential input voltage(2)	±1	
V_{in}	Input voltage(3)	V_{CC-} -0.3 to V_{CC+} +0.3	
T_{stg}	Storage temperature	-65 to +150	°C
T_j	Maximum junction temperature	150	
R_{thja}	Thermal resistance junction-to-ambient(4) SO14 TSSOP14	 66 100	°C/W
ESD	HBM: human body model(5)	3	kV
	MM: machine model(6)	100	V
	CDM: charged device model(7) SO14 TSSOP14	 1.5 1	kV
	Output short-circuit duration	See footnote(8)	
	Latch-up immunity	200	mA
	Soldering temperature (10 sec.), leaded version	250	°C
	Soldering temperature (10 sec.), unleaded version	260	

1. All voltage values, except the differential voltage, are with respect to network ground terminal.
2. The differential voltage is the non-inverting input terminal with respect to the inverting input terminal. If V_{id} > ±1 V, the maximum input current must not exceed ±1 mA. In this case (V_{id} > ±1 V), an input series resistor must be added to limit input current.
3. Do not exceed 14 V.
4. Short-circuits can cause excessive heating and destructive dissipation. R_{th} are typical values.
5. Human body model: a 100 pF capacitor is charged to the specified voltage, then discharged through a 1.5 kΩ resistor between two pins of the device. This is done for all couples of connected pin combinations while the other pins are floating.
6. Machine model: a 200 pF capacitor is charged to the specified voltage, then discharged directly between two pins of the device with no external series resistor (internal resistor < 5 Ω). This is done for all couples of connected pin combinations while the other pins are floating.
7. Charged device model: all pins and the package are charged together to the specified voltage and then discharged directly to ground through only one pin. This is done for all pins.
8. There is no short-circuit protection inside the device: short-circuits from the output to V_{CC} can cause excessive heating. The maximum output current is approximately 80 mA, independent of the magnitude of V_{CC}. Destructive dissipation can result from simultaneous short-circuits on all amplifiers.

Table 2. Operating conditions

Symbol	Parameter	Value	Unit
V_{CC}	Supply voltage	2.7 to 12	V
V_{icm}	Common mode input voltage range	V_{CC-} -0.2 to V_{CC+} +0.2	
T_{oper}	Operating free air temperature range	-40 to +125	°C

2 Electrical characteristics

Table 3. Electrical characteristics at V_{CC+} = +3 V with V_{CC-} = 0 V, V_{icm} = V_{CC+}/2, T_{amb} = 25 °C, and R_L connected to V_{CC+}/2 (unless otherwise specified)

Symbol	Parameter	Min.	Typ.	Max.	Unit
DC performance					
V_{io}	Input offset voltage TS924 TS924A $T_{min} \le T_{amb} \le T_{max}$ TS924 TS924A			 3 0.9 5 1.8	mV
DV_{io}	Input offset voltage drift		2		µV/°C
I_{io}	Input offset current - $T_{min} \le T_{amb} \le T_{max}$		1	30	nA
I_{ib}	Input bias current - $T_{min} \le T_{amb} \le T_{max}$		15	100	
CMR	V_{icm} from 0 to 3 V $T_{min} \le T_{amb} \le T_{max}$	60 56	80		dB
SVR	Supply voltage rejection ratio - V_{CC+} = 2.7 to 3.3 V $T_{min} \le T_{amb} \le T_{max}$	60 60	85		
A_{vd}	Large signal voltage gain (V_{out} = 2 V_{pk-pk}) R_L= 10 kΩ, $T_{min} \le T_{amb} \le T_{max}$ R_L = 600 Ω, $T_{min} \le T_{amb} \le T_{max}$ R_L = 32 Ω	70 15	200 35 16		V/mV
V_{OH}	High level output voltage R_L= 10 kΩ, $T_{min} \le T_{amb} \le T_{max}$ R_L = 600 Ω, $T_{min} \le T_{amb} \le T_{max}$ R_L = 32 Ω	2.90 2.87	 2.63		V
V_{OL}	Low level output voltage R_L= 10 kΩ, $T_{min} \le T_{amb} \le T_{max}$ R_L = 600 Ω, $T_{min} \le T_{amb} \le T_{max}$ R_L = 32 Ω		 180	50 100	mV
I_o	Output short-circuit current	50	80		mA
I_{CC}	Supply current /operator - no load, V_{out} = V_{CC+}/2 $T_{min} \le T_{amb} \le T_{max}$		1	1.5 1.6	
AC performance					
GBP	Gain bandwidth product - R_L = 600 Ω		4		MHz
ϕm	Phase margin at unit gain - R_L = 600 Ω, C_L =100 pF		68		Degrees
G_m	Gain margin - R_L = 600 Ω, C_L =100 pF		12		dB
SR	Slew rate	0.7	1.3		V/µs
e_n	Equivalent input noise voltage - f = 1 kHz		9		$\frac{nV}{\sqrt{Hz}}$

ANEXO II: Código implementado.

Anexo II.1: Comunicación alámbrica.

II.1.1: Código Maestro.

```
#include <VirtualWire.h>
#include <VirtualWire_Config.h>

const int transmit_pin = 10;
const int receive_pin = 11;
const int CD4046 = 8;

void setup(){

    Serial.begin(9600);
    vw_setup(220);

    vw_set_tx_pin(transmit_pin);
    vw_set_rx_pin(receive_pin);
    vw_rx_start();

    pinMode(CD4046, OUTPUT);
    digitalWrite(CD4046, HIGH);

    delay(1000);
}

void loop(){

  uint8_t Mensaje_Correcto = 0;
  const char *msg1 = "T";
  const char *regar = "A";
  const char *no_regar = "C";
  int Numero_reenvios = 5;
  int reenvio_actual = 0;

  uint8_t buf[VW_MAX_MESSAGE_LEN];
  uint8_t buflen = VW_MAX_MESSAGE_LEN;

  Serial.println("Iniciando comunicacion");
  while ((Mensaje_Correcto == 0) && (reenvio_actual < Numero_reenvios)) {

  digitalWrite(CD4046, LOW);
  delay(320);

  Serial.print("He enviado: ");
  vw_send((uint8_t *)msg1, strlen(msg1));
  vw_wait_tx();
  Serial.println(msg1);
  digitalWrite(CD4046, HIGH);

  reenvio_actual = reenvio_actual +1;
  Serial.print("Reenvio numero: ");
  Serial.println(reenvio_actual);
  long Tiempo_Maximo = 1000;
  long Tiempo_Envio = millis();
```

```
  while ((Mensaje_Correcto == 0) && (millis() - Tiempo_Envio < Tiempo_Maximo)) {

    if (vw_get_message((uint8_t *)buf, &buflen)) {
      Serial.print("He recibido ");
      Serial.print(buflen);
      Serial.print(" ");
      for( int i = 0; i < buflen; i++)
        Serial.print(buf[i], HEX);
        Serial.println();
      if (buflen == 2) {

        Mensaje_Correcto = 1;
      }
    }
    delay(50);
  }
  }

  if (Mensaje_Correcto == 1) {
    Serial.println("Mensaje correcto");
    int ValorSensor;
        ValorSensor = buf[0] << 8 | buf[1];
        Serial.print("Valor sensor recibido: ");
        Serial.println(ValorSensor);
    float Temperatura;
        Temperatura = (ValorSensor * 5.0)/(1024.0 * 0.01);
        Serial.print("Valor temperatura: ");
        Serial.println(Temperatura);

    if (Temperatura >= 28.0) {
      digitalWrite(CD4046, LOW);
      delay(320);
      Serial.println("Se debe regar");
      Serial.println("He enviado");
      vw_send((uint8_t *)regar, strlen(regar));
      vw_wait_tx();
      Serial.println(regar);
      digitalWrite(CD4046, HIGH);
    }
    if (Temperatura < 28.0) {
      digitalWrite(CD4046, LOW);
      delay(320);
      Serial.println("NO se debe regar");
      Serial.println("He enviado");
      vw_send((uint8_t *)no_regar, strlen(no_regar));
      vw_wait_tx();
      Serial.println(no_regar);
      digitalWrite(CD4046, HIGH);
    }


  }else
   Serial.println("Error de Conexion");
delay(3000);

}
```

II.1.2: Código Esclavo.

```
#include <VirtualWire.h>

const int transmit_pin = 10;
const int receive_pin = 11;
const int CD4046 = 8;
const int electrovalvula = 13;

void setup(){

    Serial.begin(9600);
    vw_setup(220);


    vw_set_tx_pin(transmit_pin);


    vw_set_rx_pin(receive_pin);
    vw_rx_start();

    pinMode(electrovalvula, OUTPUT);
    digitalWrite(electrovalvula, false);

    pinMode(CD4046, OUTPUT);
    digitalWrite(CD4046, HIGH);
}

int Tiempo_mensaje = 0;
void loop(){

uint8_t buf[VW_MAX_MESSAGE_LEN];
uint8_t buflen = VW_MAX_MESSAGE_LEN;
uint8_t Mensaje_Correcto = 0;
int i;

   if (millis() - Tiempo_mensaje >= 200){
       Serial.println("Comprobando existencia mensaje");
       Tiempo_mensaje = millis();
   }

     if (vw_get_message(buf, &buflen)){
          Serial.println("Recibiendo mensaje");
          Serial.print("He recibido: ");
          for(i = 0; i < buflen; i++)
             Serial.print((char)buf[i]);
           Serial.println();

          if ((char)buf[0] == 'T') {
            Serial.println("Mensaje correcto");

            unsigned int ValorSensor = 0;
            float Voltaje = 0.0;
            float Temperatura = 0.0;
            Serial.print("Valor de sensor enviado: ");
            ValorSensor = analogRead(A5);
            Serial.println(ValorSensor);

            Serial.print("Voltaje sensor: ");
            Voltaje = (ValorSensor* 5.0)/1024.0;
```

```
            Serial.println(Voltaje);

            Temperatura = Voltaje/0.01;
            Serial.print("Valor temperatura: ");
            Serial.println(Temperatura);

            uint8_t Envio[2];
            Envio[0] = (ValorSensor >> 8);
            Envio[1] = (ValorSensor & 0xFF);
            digitalWrite(CD4046, LOW);
            delay(50);
            vw_send((uint8_t *)Envio, 2);
            vw_wait_tx();
            Serial.print("Valor en HEXADECIMAL: ");
            Serial.print(Envio[0],HEX);
            Serial.println(Envio[1],HEX);
            digitalWrite(CD4046, HIGH);
          }


          if((char)buf[0] == 'A') {
            Serial.println("Se debe regar");
            digitalWrite(electrovalvula, true);
          }
          if((char)buf[0] == 'C') {
            Serial.println("NO se debe regar");
            digitalWrite(electrovalvula, false);
          }else
            Serial.println("Mensaje no correcto");
      }
      delay(200);
}
```

Anexo II.2: Comunicación inalámbrica.

II.2.1: Código Maestro.

```
#include <nRF24L01.h>
#include <RF24.h>
#include <RF24_config.h>

const int pinCE = 7;
const int pinCSN = 8;
RF24 radio(pinCE, pinCSN);
const uint64_t pipes[2] = { 0xF0F0F0F0E1LL, 0xF0F0F0F0D2LL };

void setup(void) {

  Serial.begin(9600);
  radio.begin();
  radio.openWritingPipe(pipes[0]);
  radio.openReadingPipe(1,pipes[1]);

}

void loop(){

  uint8_t Mensaje_Correcto = 0;
  const char *msg1 = "T";
  const char *regar = "A";
  const char *no_regar = "C";
  int Numero_reenvios = 10;
  int reenvio_actual = 0;
  uint8_t dato_temp[2];

  Serial.println("Iniciando comunicacion");
  while ((Mensaje_Correcto == 0) && (reenvio_actual <
Numero_reenvios)) {

  radio.stopListening();
  Serial.print("He enviado: ");
  radio.write(msg1, sizeof msg1);
  Serial.println(msg1);
  reenvio_actual = reenvio_actual +1;
  Serial.print("Reenvio numero: ");
  Serial.println(reenvio_actual);
  long Tiempo_Maximo = 500;
  long Tiempo_Envio = millis();
  radio.startListening();
  while ((Mensaje_Correcto == 0) && (millis() - Tiempo_Envio <
Tiempo_Maximo)) {

    if (radio.available()) {

      Serial.print("He recibido ");
      radio.read(&dato_temp, sizeof(dato_temp));
      for( int i = 0; i < 2; i++)
        Serial.print(dato_temp[i], HEX);
        Serial.println();
      if (sizeof dato_temp == 2) {
```

```
        Mensaje_Correcto = 1;
      }
    }
    delay(50);
  }
  }

    if (Mensaje_Correcto == 1) {
    Serial.println("Mensaje correcto");
    int ValorSensor;
        ValorSensor = dato_temp[0] << 8 | dato_temp[1];
        Serial.print("Valor sensor recibido: ");
        Serial.println(ValorSensor);
    float Temperatura;
        Temperatura = (ValorSensor * 5.0)/(1023.0 * 0.01);
        Serial.print("Valor temperatura: ");
        Serial.println(Temperatura);

    if (Temperatura >= 28.0) {
      radio.stopListening();
      Serial.println("Se debe regar");
      Serial.println("He enviado");
      radio.write(regar, sizeof regar);
      Serial.println(regar);
      radio.startListening();
    }

    if (Temperatura < 28.0) {
      radio.stopListening();
      Serial.println("NO se debe regar");
      Serial.println("He enviado");
      radio.write(no_regar, sizeof no_regar);
      Serial.println(no_regar);
      radio.startListening();
    }
  }else
   Serial.println("Error de Conexion");
}
```

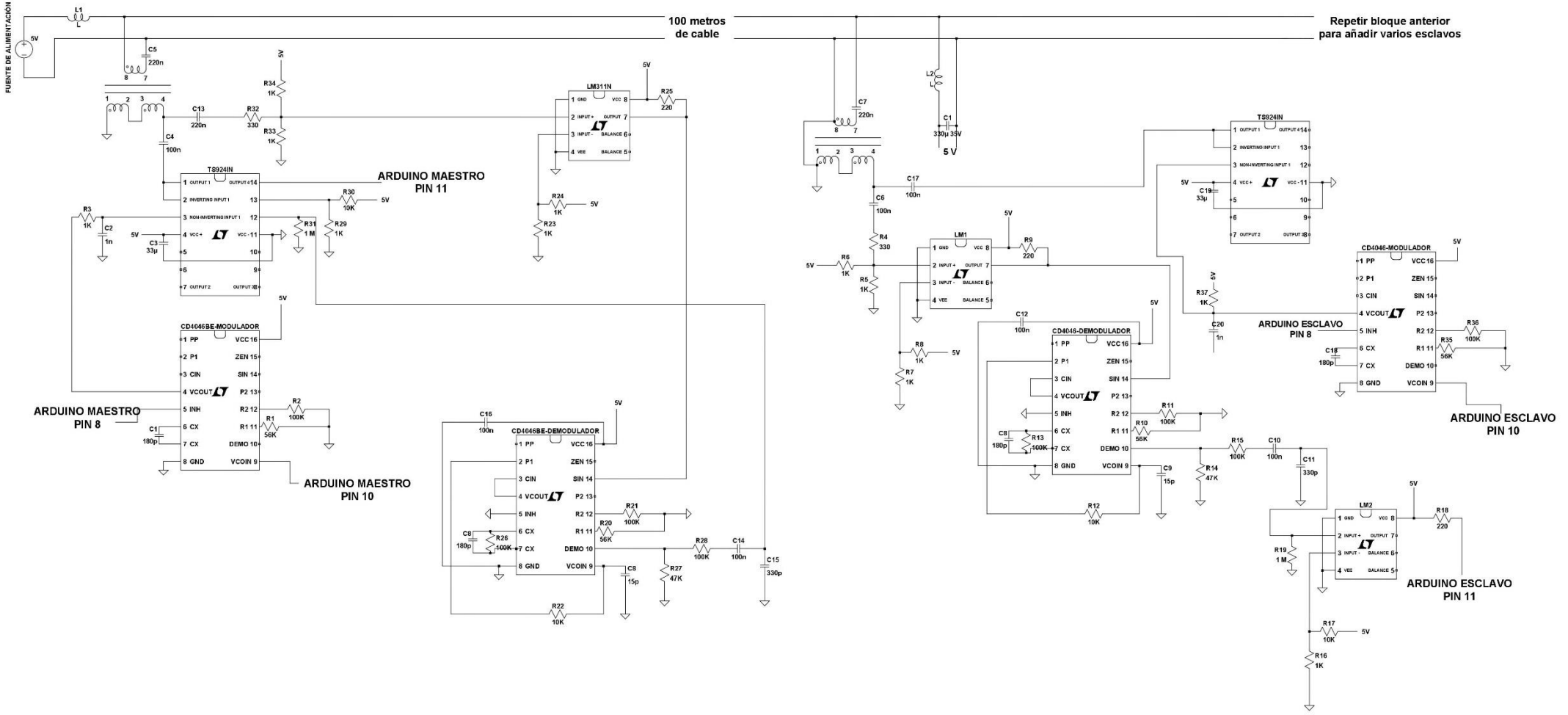
II.2.2: Código Esclavo.

```
#include <nRF24L01.h>
#include <RF24.h>
#include <RF24_config.h>

const int pinCE = 7;
const int pinCSN = 8;
RF24 radio(pinCE, pinCSN);
const uint64_t pipes[2] = { 0xF0F0F0F0E1LL, 0xF0F0F0F0D2LL };
uint8_t msj_maestro[2];
const int electrovalvula = 9;

void setup(){

    Serial.begin(9600);
    radio.begin();

    radio.startListening();
    radio.openWritingPipe(pipes[1]);
    radio.openReadingPipe(1,pipes[0]);

    pinMode(electrovalvula, OUTPUT);
    digitalWrite(electrovalvula, false);
}

int Tiempo_mensaje = 0;
void loop(){

uint8_t Mensaje_Correcto = 0;
int i;

   if (millis() - Tiempo_mensaje >= 200){
       Serial.println("Comprobando existencia mensaje");
       Tiempo_mensaje = millis();
   }

     if (radio.available()) {
        Serial.println("Recibiendo mensaje");
        Serial.print("He recibido: ");
        radio.read(msj_maestro, sizeof(msj_maestro));

        for(i = 0; i < 2; i++)
             Serial.print((char)msj_maestro[i]);
           Serial.println();
        if ((char)msj_maestro[0] == 'T') {
            Serial.println("Mensaje correcto");

            unsigned int ValorSensor = 0;
            float Voltaje = 0.0;
            float Temperatura = 0.0;
            Serial.print("Valor de sensor enviado: ");
            ValorSensor = analogRead(A5);
            Serial.println(ValorSensor);

            Serial.print("Voltaje sensor: ");
            Voltaje = (ValorSensor* 5.0)/1024.0;
            Serial.println(Voltaje);

            Temperatura = Voltaje/0.01;
```

```
            Serial.print("Valor temperatura: ");
            Serial.println(Temperatura);

            radio.stopListening();
            uint8_t Envio[2];
            Envio[0] = (ValorSensor >> 8);
            Envio[1] = (ValorSensor & 0xFF);
            radio.write((uint8_t *)Envio, 2);
            Serial.print("Valor en HEXADECIMAL: ");
            Serial.print(Envio[0],HEX);
            Serial.println(Envio[1],HEX);
            radio.startListening();
        }

        if((char)msj_maestro[0] == 'A') {
        Serial.println("Se debe regar");
        digitalWrite(electrovalvula, true);
        }

        if((char)msj_maestro[0] == 'C') {
        Serial.println("NO se debe regar");
        digitalWrite(electrovalvula, false);
        }else
          Serial.println("Mensaje no correcto");

      }
      delay(200);
}
```

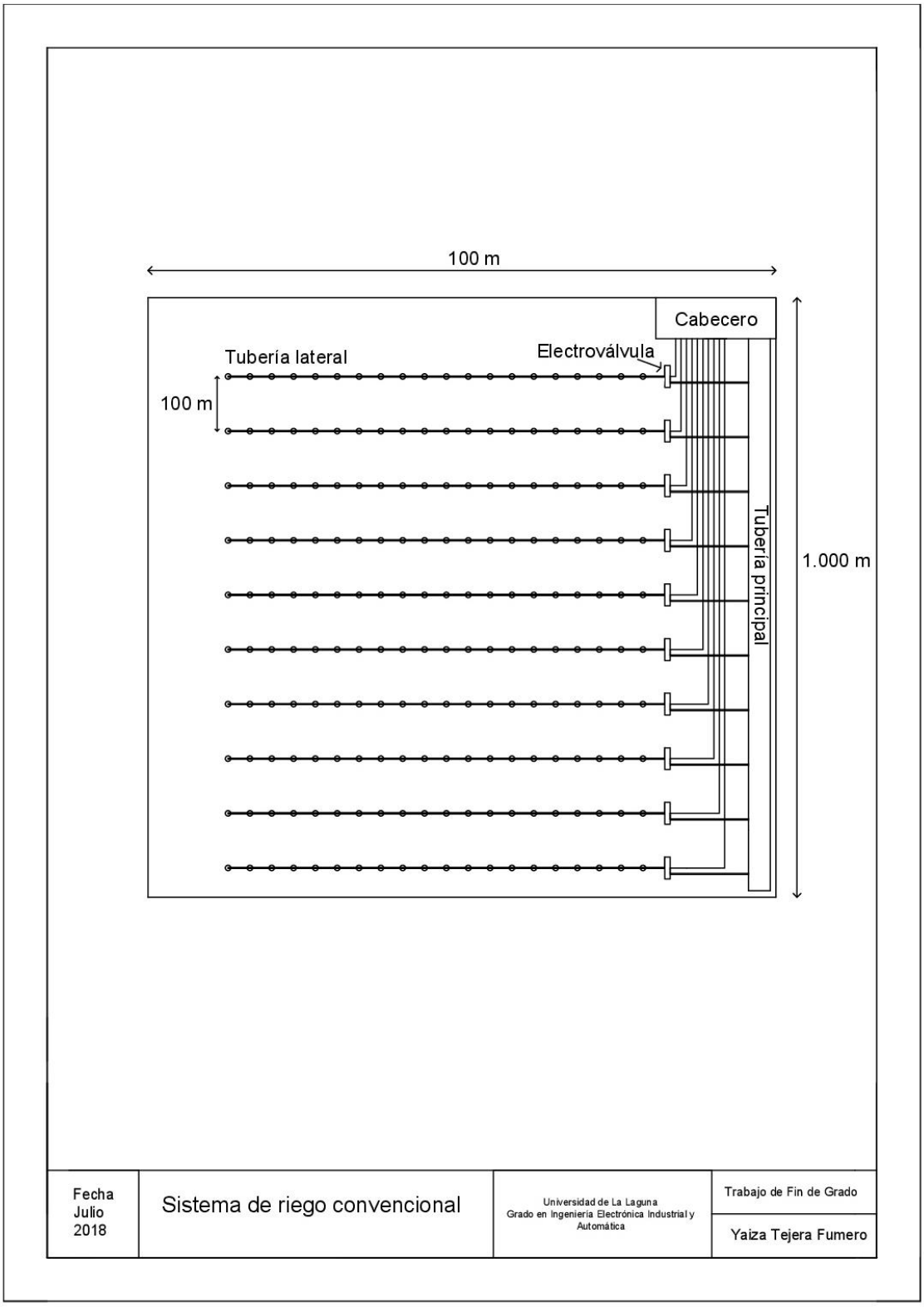
ANEXO III: Esquemas y conexiones

Anexo III.1: Comunicación bidireccional.



ANEXO IV: Croquis de las instalaciones agrícolas.

Anexo IV.1: Instalación agrícola convencional.



Anexo IV.2: Instalación agrícola inteligente.