ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

GRADO EN INGENIERÍA TELEMÁTICA

IMPLEMENTACIÓN DE UN DRIVER PARA MEDIDA DE UN SENSOR CON RADIOFRECUENCIA

Autor: Carlos Arranz Herrero

Director: Javier Matanza Domingo

Madrid

Julio 2018

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título
Implementación de un driver para medida un sensor con radiofrecuencia.
en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el
curso académico 2017/18 es de mi autoría, original e inédito y
no ha sido presentado con anterioridad a otros efectos.
El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Fdo.: Carlos Arranz Herrero Fecha: 10/ 07/ 2018

Autorizada la entrega del proyecto
EL DIRECTOR DEL PROYECTO

Fdo.: Javier Matanza Domingo Fecha: 10/ 07 / 2018

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. Carlos Arranz Herrero DECLARA ser el titular de los derechos de propiedad intelectual de la obra: Implementación de un driver para medida de un sensor con radiofrecuencia, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.
b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
e) Asignar por defecto a estos trabajos una licencia Creative Commons.
f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente).*

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

a) Que la Universidad identifique claramente su nombre como autor de la misma
b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
c) Solicitar la retirada de la obra del repositorio por causa justificada.
d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

El autor se compromete a:

a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.
d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 10 de Julio de 2018

ACEPTA

Fdo. Carlos Arranz Herrero

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

Agradecimientos

Gracias a Dios.

Las primeras personas a las que agradecer en este trabajo es a mis padres. Gracias por vuestro esfuerzo, porque sin vosotros no hubiera podido llegar a ser nada de lo que soy hoy. A vosotros os dedico este trabajo, todo lo que conseguí y todo lo que conseguiré a partir de hoy en mi vida ha sido y es gracias a vosotros. Sin vuestro amor, generosidad, gratitud y fortaleza no podría haber llegado ni a un cuarto de la mitad de lo que he llegado a ser hoy. Gracias.
Gracias también a mis hermanos, de Alberto aprendí el gusto por la ingeniería y el sacrificio, y de Javi la generosidad y paciencia del trabajo día tras días. Gracias a vosotros por todos estos 22 años.

También quiero agradecer a mi compañera en la vida. Gracias por aguantar todos estos años, la mitad de mi ingeniería es y será gracias a ti. Agradecer toda tu confianza puesta en mí es poco, ya que hubo momentos en los que tu confiaste más en mi que yo mismo. Sin tu fortaleza, voluntad y sacrificio no me hubiera sido posible llegar hasta aquí. Por todo esto y por lo que me queda agradecerte en nuestra vida futura juntos, gracias Sara.

A toda la parroquia San juan de Ávila, por todos los que han contribuido de una manera especial y esencial a mi formación como persona. Gracias a D. Juan del Rey y a D. Francisco Javier Zaaera, por todos estos años de sacrificio y de fidelidad. También agradecer a los seminaristas que han pasado por mi vida y a nuestras consagradas. Gracias todos los jóvenes pipiolos de los viernes, pilar fundamental por el que se sostenía mi vida espiritual. Gracias al coro, que me ha enseñado lo que vale la fidelidad y la entrega. Gracias al grupo de jóvenes “maduros”, vosotros sois mi mayor apuesta en la vida, no lo olvidéis nunca. Y gracias a todas esas personas que de una manera u otra han ayudado a mi vida, ya sea directamente o a través de la oración.

Gracias a mi director de proyecto, Javier Matanza. Gracias por todo el tiempo dedicado a mi. Gracias por tus rápidas respuestas y tu cercanía. Gracias por ofrecerme un trabajo de fin de proyecto en el que he podido gustar mas de la carrera que he elegido. Ser profesor es difícil, pero hacer que otros gusten esa profesión es mas todavía, y en este caso lo consigues. Para mí ha sido un ejemplo poder contar de primera mano con alguien como tu.

Gracias a mis compañeros de la universidad, sin vosotros hubiera sido muy diferente. He podido disfrutar de vuestra compañía y de vuestra generosidad. Os deseo a todos una carrera en la que consigáis todo lo que os propongáis. Gracias de una manera especial a Alex, Pilar, Irene, Conchita, Marta, Pablo, Itziar, Pío, Daniel y a los gymnasticos.

A todos vosotros os dedico este proyecto.

GRACIAS.

IMPLEMENTACION DE UN DRIVER PARA MEDIDA DE UN SENSOR CON RADIOFRECUENCIA

Autor: Arranz Herrero, Carlos.
Director: Matanza Domingo, Javier.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Se ha llevado a cabo un desarrollo de un dispositivo capaz de leer la respuesta de un sensor por medio de técnicas de radiofrecuencia. La implementación se ha basado en utilizar un microprocesador para generar una señal y una placa PCB para su proceso. Esta señal es recibida por la placa PCB debidamente para generar un pulso. El sensor generará un notch a una frecuencia específica, que variará en función del dieléctrico que se encuentre encima del sensor. La respuesta la trataremos por medio de un detector que finalmente recibiremos en el microprocesador para su almacenamiento y análisis. El microprocesador se ha desarrollado en lenguaje C (mediante el framework Kinetis [1]) y el diseño de la placa se ha desarrollado por medio del programa Kicad [2], una aplicación para diseño de placas PCB.

Palabras clave: microprocesador, Placa PCB, Notch, dieléctrico, Kinetis, Kicad.

1. Introducción

El gran inconveniente de los proyectos de electrónica donde se utilizan componentes activos [3] viene cuando tratamos de diseñar un controlador para medir un sensor. Los componentes activos requieren de la utilización de baterías integradas o poder disponer de su propia fuente de alimentación. Dichos componentes activos no son desechables, por lo que incrementa el coste de cada sensor. Este modelo de desarrollo necesita un diseño de un controlador específico para cada uno de los diferentes sensores, ya que la respuesta de cada uno de ellos es diferente.

Es por eso por lo que se requiere buscar una nueva arquitectura de desarrollo utilizando únicamente componentes pasivos [4]. De esta idea surgen diversas tecnologías que dan respuesta a estos frentes, como el RFID [5], que utiliza componentes semi-activos o pasivos. Estos componentes semi-activos también poseen una fuente de alimentación propia. La diferencia está en el uso de la fuente para alimentar el microchip, y no para transmitir una señal [6]. La tendencia actual es buscar un diseño con componentes totalmente pasivos, como resonadores, consiguiendo homogeneizar un controlador electrónico capaz de medir los diferentes sensores usando radiofrecuencia.

2. Definición del proyecto

El objetivo principal del proyecto es el desarrollo de un lector capaz de leer información de un sensor por medio de técnicas basadas en radiofrecuencia. Por lo que buscamos un dispositivo capaz de generar una señal, procesarla y posteriormente obtener la respuesta. Diseñaremos una placa PCB para todo el procesado de nuestra señal, ya que tendremos que modificar la salida del microprocesador hasta llegar al sensor, para que éste pueda darnos la respuesta en frecuencia. Este dispositivo tiene además la capacidad de ser escalable, ya que se desarrolla un software sobre un microprocesador capaz de añadir diferentes módulos con tecnologías como Ethernet, USB, etc. Aunque este aspecto no es el objeto del presente proyecto, podría servir como extensión de este en un futuro.

Además, nuestro objetivo, como hemos comentado anteriormente, será la utilización de componentes de bajo coste de fabricación y desechables, lo cual añade al proyecto un valor añadido desde el punto de vista industrial y comercial.

3. Descripción del modelo/sistema/herramienta

Para desarrollar este proyecto hemos creído conveniente dividirlo en dos partes: la primera basada en el desarrollo del software del microprocesador, y la segunda en el diseño de la placa de procesamiento de la señal de nuestro dispositivo.

- La parte del desarrollo del software del microprocesador se ha llevado a cabo utilizando el modelo de microprocesador NXP KL25Z [7]. Por ello, nos obliga a hacer uso del framework Kinetis [1] y la API de Kinetis SDK 2.0 [8] para utilizar los drivers del microprocesador. Kinetis es un framework que nos facilita desarrollar un proyecto en '*C*'. Este entorno nos ofrece la conexión y compilación con el microprocesador.
- La segunda parte consiste en el desarrollo de una placa PCB para procesar la señal. De uno de los pines del microprocesador, el DAC (Digital to Analogic Conversor [9]), saldrá una señal diente de sierra, que deberemos modular mediante un oscilador controlado por tensión (VCO [10]) para producir un pulso en frecuencia. Este pulso lo recibirá el sensor y generará un Notch a una determinada frecuencia que variará en función del dieléctrico que haya encima del sensor. Finalmente, filtraremos la respuesta del sensor y la encaminaremos por un detector que conducirá al pin del microprocesador ADC (Analogic to Digital Conversor [9]).

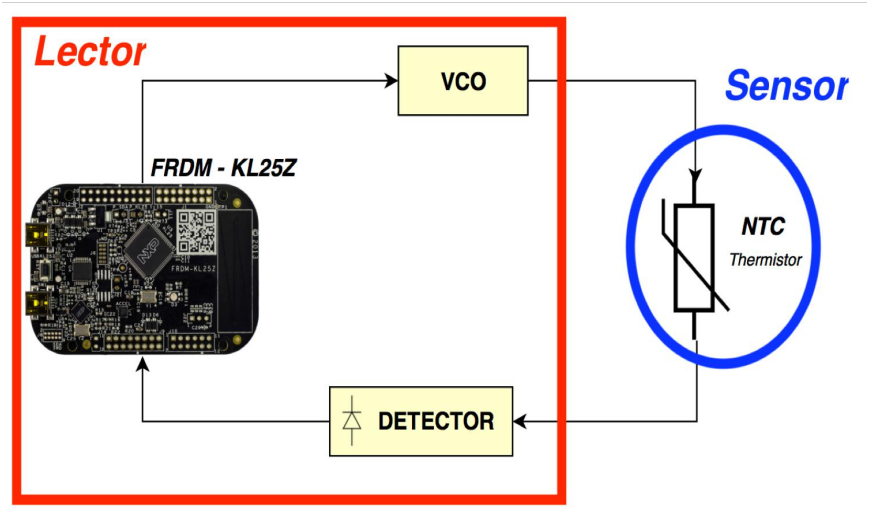


Ilustración 1 - Esquema del dispositivo

4. Resultados

- Desarrollamos una placa PCB que implementa un oscilador controlado por tensión, un filtro en las frecuencias de actuación del VCO usado y un detector que recupere la señal para pasarla por medio de un conversor analógico a digital (ADC). Lo podemos observar en la Ilustración 2.

- Desarrollamos un software en el microprocesador capaz de generar una señal en diente de sierra cada 1ms. La salida de cada etapa lo podemos observar en la Ilustración 3.
- La señal la procesamos mediante un VCO capaz de convertir la señal a un pulso en frecuencia correctamente.
- El sensor genera un Notch a una frecuencia determinada en función del dieléctrico que esté situado encima.
- Recibimos correctamente la señal en el microprocesador tras pasar por un filtro y un detector.

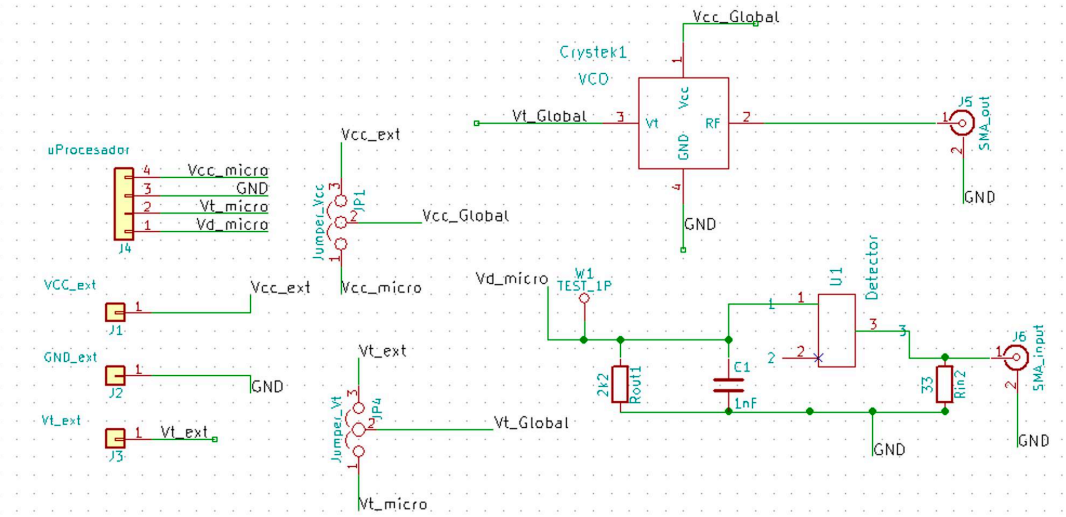


Ilustración 2 – Esquema de la placa PCB en detalle

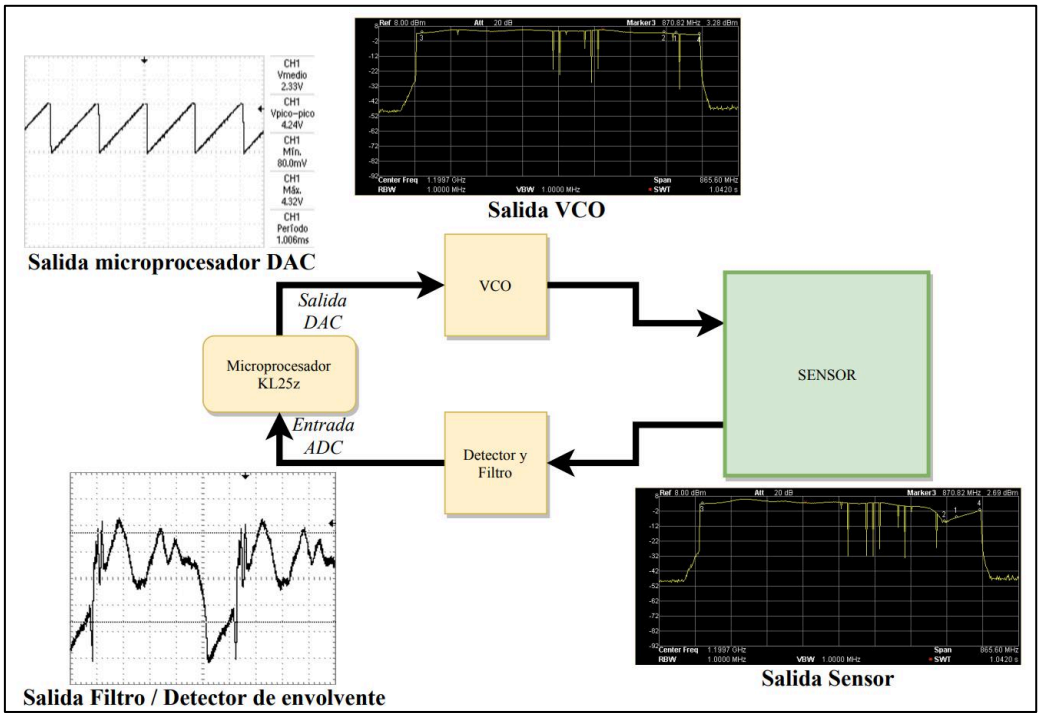


Ilustración 3 – Salida de cada etapa de izquierda a derecha y arriba a abajo: Salida del microprocesador, salida del VCO, salida del sensor, salida del filtro y el detector.

5. Conclusiones

En este proyecto se ha conseguido el objetivo de crear un lector utilizando radiofrecuencia. Se ha demostrado también la capacidad de este tipo de tecnologías para ser escalable y de bajo coste de fabricación, lo que además añade un interés comercial por el uso de componentes pasivos en los dispositivos. Este proyecto abre la puerta a un gran mundo de posibilidades de innovación utilizando técnicas que facilitan el uso de este tipo de componentes.

6. Referencias

[1] N. Freescale, «KDS_IDE: Entorno de desarrollo integrado de Kinetis® Design Studio (IDE),» [En línea]. Available: https://www.nxp.com/support/developer-resources/software-development-tools/kinetis-design-studio-integrated-development-environment-ide:KDS_IDE.

[2] Kicad, «Kicad EDA tool,» [En línea]. Available: http://kicad-pcb.org/.

[3] ECURED, «Componentes electrónicos,» [En línea]. Available: https://www.ecured.cu/Componente_electrónico.

[4] ECURED, «Componentes electrónicos pasivos,» [En línea]. Available: https://www.ecured.cu/Componentes_electrónicos_pasivos.

[5] By, «¿Qué es el RFID?,» [En línea]. Available: https://www.by.com.es/blog/que-es-rfid/.

[6] Wikipedia, «RFID,» [En línea]. Available: https://es.wikipedia.org/wiki/RFID.

[7] F. NXP, «Manual de referencia KL25z,» [En línea]. Available: https://www.nxp.com/docs/en/reference-manual/KL25P80M48SF0RM.pdf.

[8] NxP, «Kinetis SDK 2.0 API,» [En línea]. Available: https://community.nxp.com/docs/DOC-329783.

[9] Wikipedia®, «Conversor de señal digital a analógica [ADC] [DAC],» [En línea]. Available: https://es.wikipedia.org/wiki/Conversor_de_señal_digital_a_analógica.

[10] Wikipedia, «Oscilador controlado por tensión (VCO),» [En línea]. Available: https://en.wikipedia.org/wiki/Voltage-controlled_oscillator.

IMPLEMENTATION OF A DRIVER FOR MEASUREMENT OF TEMPERATURE WITH RADIO FREQUENCY

Author: Arranz Herrero, Carlos.
Supervisor: Matanza Domingo, Javier.
Collaborating Entity: ICAI – Universidad Pontificia Comillas.

ABSTRACT

A development of a device capable of reading the response of a sensor by means of radiofrequency techniques has been carried out. The implementation has been based on using a microprocessor to generate a signal and a PCB board for its process. This signal is received by the PCB board properly to generate a pulse. The sensor will generate a Noch at a specific frequency, which will vary depending on the dielectric that is on top of the sensor. We will receive the response by means of a detector in the microprocessor for its storage and analysis. The microprocessor has been developed in C language (through the Kinetis framework [1]) and the board design has been developed through the Kicad [2] program, an application for PCB board design.

Keywords: microprocessor, PCB board, Notch, dielectric, Kinetis, Kicad.

1. Introduction.

The great drawback of electronic projects where active components [3] are used comes when we try to design a controller to measure a sensor. The active components require the use of integrated batteries or be able to have their own power supply. These active components are not disposable, which increases the cost of each sensor. This development model needs a design of a specific controller for each of the different sensors, since the response of each of them is different.
That is why it is necessary to look for a new development architecture using only passive components [4]. From this idea arise various technologies that respond to these fronts, such as RFID [5], which uses semi-active or passive components. These semi-active components also have their own power source. The difference is in the use of the source to feed the microchip, and not to transmit a signal [6]. The current trend is to look for a design with totally passive components, such as resonators, getting to homogenize an electronic controller capable of measuring the different sensors using radiofrequency.

2. Definition of the project

The main objective of the project is the development of a reader capable of reading information from a sensor by means of radiofrequency-based techniques. So, we look for a device capable of generating a signal, process it and then get the answer.
We will design a PCB board for all the processing of our signal, since we will have to modify the output of the micro until reaching the sensor so that it can give us the frequency response. This device allows the ability to be scalable, since software is developed on a microprocessor capable of adding different modules with technologies such as Ethernet, USB, etc ... Although this does not fall within the scope of this project but could serve as an extension of East.

In addition, our objective, as we have commented previously, will be the use of components that are cheap to manufacture and disposable. What we get, therefore, is a commercial interest in the project.

3. Description of the model / system / tool

For the development of this project we have needed to divide it into two parts: the first based on the development of the microprocessor software, and the second on the design of the signal processing board of our device.

- The part of the development of the microprocessor software has been carried out using the NXP KL25Z microprocessor model. Therefore, it forces us to use the Kinetis framework and the Kinetis SDK 2.0 API to use the microprocessor drivers. Kinetis is a framework that facilitates us to develop a project in 'C'. This environment offers us the connection and compilation with the microprocessor.

- The second part consists in the development of a PCB board to process the signal. From one of the microprocessor pins, the DAC (Digital to Analogic converter) will output a saw tooth signal, and we have to modulate it by means of a voltage-controlled oscillator (VCO) to produce a pulse in frequency. This pulse will be received by the sensor and will generate a notch at a certain frequency that will vary depending on the dielectric on the sensor. Finally, the sensor response will be filtered and routed through a detector that will lead to the ADC (Analogic to Digital Converter) microprocessor pin.

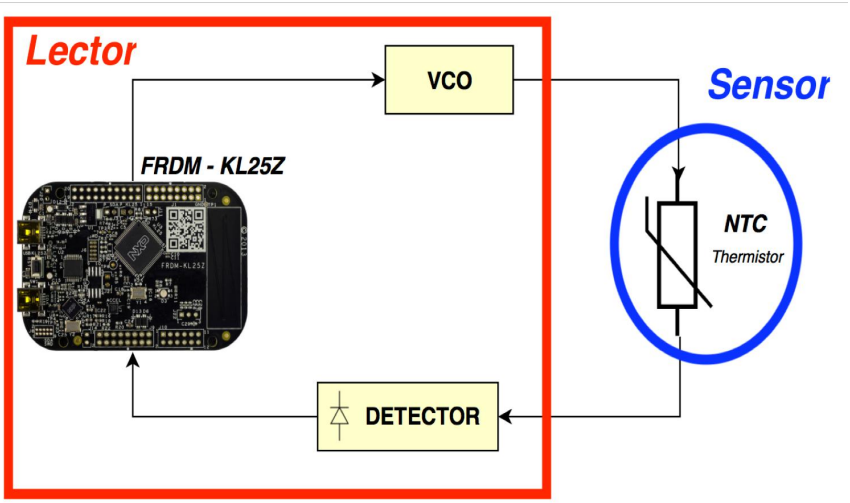


Figure 4 - Scheme of the device

4. Results

- We develop a PCB board that implements a voltage-controlled oscillator, a filter at the operating frequencies of the VCO used and a detector that recovers the signal to pass it through an analog to digital converter (ADC). We can see it in Figure 2.

- We develop a software in the microprocessor capable of generating a signal in sawtooth every 1ms. The output of each stage can be seen in Figure 3.
- The signal is processed by a VCO capable of converting the signal to a pulse in frequency correctly.
- The sensor generates a Notch at a certain frequency depending on the dielectric that is located above it.
- We correctly receive the signal in the microprocessor after passing through a filter and a detector.

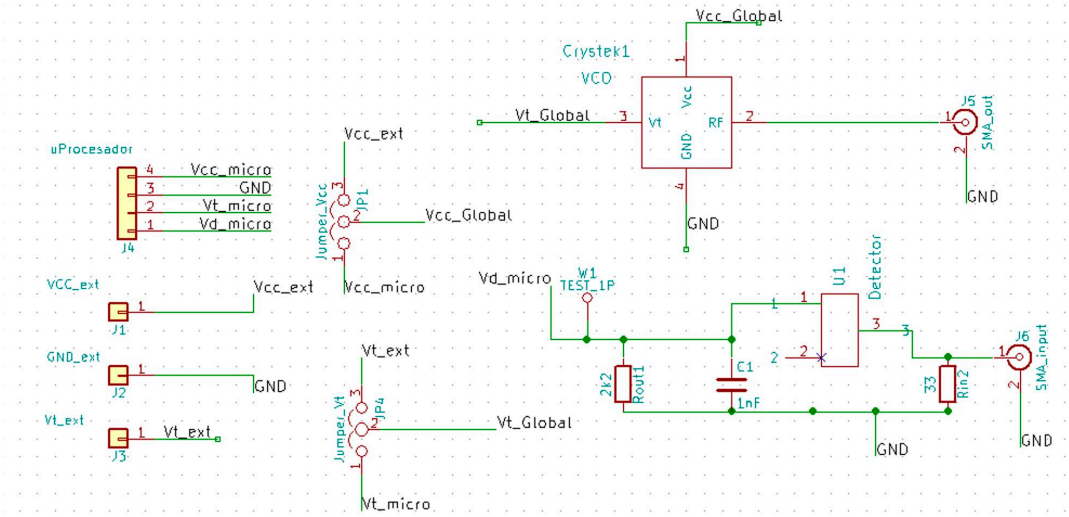


Figure 5 – Diagram of the PCB board in detail

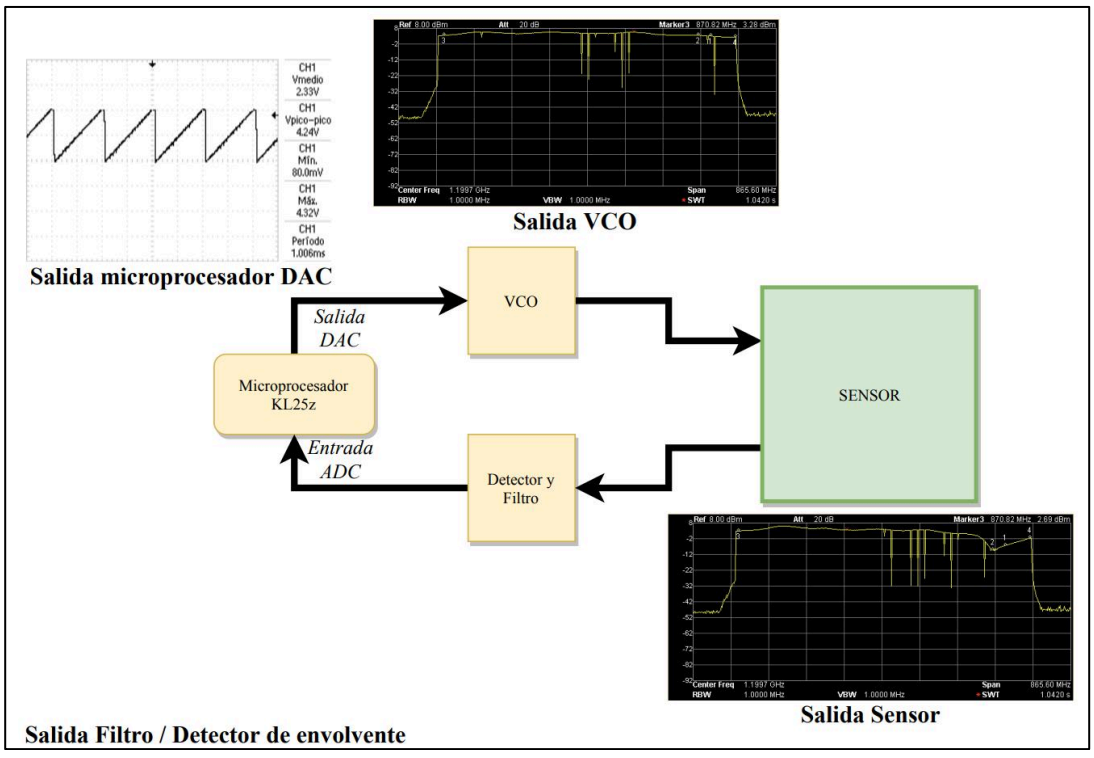


Figure 6 – Output from each stage from left to right and top to bottom: Microprocessor output, VCO output, sensor output, filter output and detector.

5. Conclusions

The objective of creating a reader using radiofrequency has been achieved. It demonstrates the ability of this type of technology to be scalable and cheap, and also, considering that we obtain a commercial interest in this type of technology by the use of passive components in the devices. This project opens the door to a great world of possibilities in which innovation stands out using techniques that facilitate the use of this type of components.

6. Referencias

[1] N. Freescale, «KDS_IDE: Entorno de desarrollo integrado de Kinetis® Design Studio (IDE),» [En línea]. Available: https://www.nxp.com/support/developer-resources/software-development-tools/kinetis-design-studio-integrated-development-environment-ide:KDS_IDE.

[2] Kicad, «Kicad EDA tool,» [En línea]. Available: http://kicad-pcb.org/.

[3] ECURED, «Componentes electrónicos,» [En línea]. Available: https://www.ecured.cu/Componente_electrónico.

[4] ECURED, «Componentes electrónicos pasivos,» [En línea]. Available: https://www.ecured.cu/Componentes_electrónicos_pasivos.

[5] By, «¿Qué es el RFID?,» [En línea]. Available: https://www.by.com.es/blog/que-es-rfid/.

[6] Wikipedia, «RFID,» [En línea]. Available: https://es.wikipedia.org/wiki/RFID.

[7] F. NXP, «Manual de referencia KL25z,» [En línea]. Available: https://www.nxp.com/docs/en/reference-manual/KL25P80M48SF0RM.pdf.

[8] NxP, «Kinetis SDK 2.0 API,» [En línea]. Available: https://community.nxp.com/docs/DOC-329783.

[9] Wikipedia®, «Conversor de señal digital a analógica [ADC] [DAC],» [En línea]. Available: https://es.wikipedia.org/wiki/Conversor_de_señal_digital_a_analógica.

[10] Wikipedia, «Oscilador controlado por tensión (VCO),» [En línea]. Available: https://en.wikipedia.org/wiki/Voltage-controlled_oscillator.

Índice de la memoria

Capítulo 1. Introducción ***9***

Capítulo 2. Descripción de las Tecnologías ***11***

2.1 Microprocesador KL25Z 11

2.1.1 Conversor digital / analógico *12*

2.1.2 Kinetis SDK 2.0 API *12*

2.2 Procesamiento digital de la señal 14

2.2.1 Oscilador controlado por tensión *14*

2.2.2 Filtro notch *14*

2.3 Diseño de placa PCB 15

2.3.1 Kicad EDA *15*

Capítulo 3. Estado de la Cuestión ***17***

3.1.1 Radio Frequency Identification *17*

3.1.2 Zigbee *19*

3.1.3 Split-ring resonator (SRR)-based sensor *20*

Capítulo 4. Definición del Trabajo ***23***

4.1 Justificación 23

4.2 Objetivos 24

4.3 Metodología 25

4.4 Planificación y Estimación Económica 26

4.4.1 Planificación *26*

Estimación económica *27*

Capítulo 5. Análisis Sistema/Modelo ***29***

5.1 Análisis del Sistema 30

5.1.1 Sensor basado en un resonador de anillo dividido *30*

5.1.2 Oscilador controlado por tensión *32*

5.1.3 Filtro y detector de envolvente *40*

Capítulo 6. Sistema/Modelo Desarrollado ***41***

6.1 Desarrollo software 41

6.1.1 Timer *41*

6.1.2 DAC *44*
6.1.3 ADC *46*
6.2 Diseño placa PCB 48
6.2.1 Esquema del circuito *48*
6.2.2 Huellas componentes *54*
6.2.3 Ensamblaje del diseño *60*
6.3 Implementación 65
6.3.1 Salida del DAC del microprocesador *66*
6.3.2 Salida del VCO *68*
6.4 Solución al sistema planteado 70

Capítulo 7. Análisis de Resultados 73
7.1 Resultados reales 73
7.1.1 Salida microprocesador amplificado *73*
7.1.2 Salida VCO *74*
7.1.3 Salida sensor cuando el dieléctrico es el aire *76*
7.1.4 Salida sensor cuando el dieléctrico es alcohol *77*
7.2 Resultados ideales 79
7.3 Respuesta del detector de envolvente 83

Capítulo 8. Conclusiones y Trabajos Futuros 85

Capítulo 9. Bibliografía 87

ANEXO A. Código 91

Índice de figuras

Ilustración 1 - Esquema del dispositivo 10
Ilustración 2 – Esquema de la placa PCB en detalle 11
Ilustración 3 – Salida de cada etapa de izquierda a derecha y arriba a abajo: Salida del microprocesador, salida del VCO, salida del sensor, salida del filtro y el detector. 11
Figure 4 - Scheme of the device 14
Figure 5 – Diagram of the PCB board in detail 15
Figure 6 – Output from each stage from left to right and top to bottom: Microprocessor output, VCO output, sensor output, filter output and detector 15
Ilustración 7 – Microprocesador KL25Z, ... 11
Ilustración 8 – Arquitectura Kinetis SDK 2.0 [8] 13
Ilustración 9 – Notch 15
Ilustración 10 – Esquema funcionamiento RFID 17
Ilustración 11 – Planificación del proyecto 26
Ilustración 12 – Esquema del modelo del trabajo propuesto 29
Ilustración 13 – Pulso en frecuencia 30
Ilustración 14 – Esquema de la salida del sensor 31
Ilustración 15 – Sensor 31
Ilustración 16 – CVCO55CW-1000-1500 32
Ilustración 17 – Curva de ajuste CVCO55CW-1000-1500-38335 32
Ilustración 18 – Respuesta VCO a una determinada tensión de entrada en el VCO 33
Ilustración 19 – Deltas del primer armónico en función de la frecuencia y la tensión de entrada 36
Ilustración 20 – Relación entre tensión de salida VCO y frecuencia fundamental de salida 37
Ilustración 21 – Variación entre tensión Vcc de referencia y la frecuencia de saldia 38
Ilustración 22 – Tren de deltas 39
Ilustración 23 – Tren de deltas anterior, pero con sweep 1 segundo 39
Ilustración 24 – Esquema del filtro y detector de envolvente implementado 40
Ilustración 25 – Configuración Timer 42

Ilustración 26 – Configuración DAC 44
Ilustración 27 – Configuración ADC 46
Ilustración 28 – Esquema circuito PCB 48
Ilustración 29 – Entradas microprocesador 49
Ilustración 30 – Entradas auxiliares 49
Ilustración 31 – Jumpers para escoger fuente de alimentación y entrada al VCO 50
Ilustración 32 – Esquemático componente VCO 50
Ilustración 33 – Entrada y salida al sensor 51
Ilustración 34 – Esquema diodo HSMS-2850 51
Ilustración 35 – Formas de utilizar el HSMS-2850 52
Ilustración 36 – Esquema filtro y detector de envolvente 52
Ilustración 37 - Huella pines macho 54
Ilustración 38 – Huella entrada auxiliar banana 2mm 55
Ilustración 39 – Huella Jumpers 55
Ilustración 40 – Medidas VCO obtenidas del datasheet 56
Ilustración 41 – Huella VCO 57
Ilustración 42 – Conector SMA para radiofrecuencia 57
Ilustración 43 – Huella conector SMA 58
Ilustración 44 – Huella del condensador y las resistencias, de izquierda a derecha. 58
Ilustración 45 – Dimensiones diodo HSMS-2850 59
Ilustración 46 – Huella diodo HSMS-2850 59
Ilustración 47 – Huella diodo y resistencia, de izquierda a derecha. 60
Ilustración 48 – Diseño PCB Final, todas las capas 62
Ilustración 49 – Capa Front-end sin Relleno 63
Ilustración 50 – Capa Front-end con relleno de Tierra 63
Ilustración 51 – Capa Back-end sin relleno 64
Ilustración 52 – Capa Back-end con relleno de Tierra 64
Ilustración 53 – Placa PCB Final Fabricada 65
Ilustración 54 – Configuración para funcionar con el microprocesador y para funcionar con entradas auxiliares, de izquierda a derecha respectivamente. 66
Ilustración 55 – Salida DAC microprocesador 67

Ilustración 56 – Analizador de espectros de la respuesta del VCO a la salida del DAC del microprocesador real 68
Ilustración 57 – Analizador de espectros de la salida del sensor, siendo el aire el dieléctrico utilizado. 69
Ilustración 58 – Analizador de espectros de la salida del sensor, siendo alcohol el dieléctrico utilizado. 69
Ilustración 59 – Esquema amplificador realimentado negativamente 70
Ilustración 60 – Esquema OP-Amp LM741 con una ganancia de 1,34 Voltios 71
Ilustración 61 – Amplificador de ganancia 1,34 Voltios 72
Ilustración 62 – Salida DAC del microprocesador tras pasar por el amplificador en tiempo 74
Ilustración 63 – Salida VCO real con sweep de 26 milisegundos en frecuencia 75
Ilustración 64 – Salida VCO real con sweep de 1 segundo en frecuencia 75
Ilustración 65 – Salida Sensor real cuando el dieléctrico es el aire con sweep de 26 milisegundos en frecuencia 76
Ilustración 66 – Salida Sensor real cuando el dieléctrico es el aire con sweep de 1 segundo en frecuencia 77
Ilustración 67 – Salida Sensor real cuando el dieléctrico es alcohol con sweep de 26 milisegundos en frecuencia 78
Ilustración 68 – Salida Sensor real cuando el dieléctrico es alcohol con sweep de 1 segundo en frecuencia 78
Ilustración 69 – Salida VCO ideal con sweep de 26 milisegundos en frecuencia 79
Ilustración 70 – Salida VCO ideal con sweep de 1 segundo en frecuencia 80
Ilustración 71 – Salida Sensor ideal cuando el dieléctrico es el aire con sweep de 26 milisegundos en frecuencia 81
Ilustración 72 – Salida Sensor ideal cuando el dieléctrico es el aire con sweep de 1 segundo en frecuencia 81
Ilustración 73 – Salida Sensor ideal cuando el dieléctrico es alcohol con sweep de 26 milisegundos en frecuencia 82
Ilustración 74 – Salida Sensor ideal cuando el dieléctrico es alcohol con sweep de 1 segundo en frecuencia 83

Ilustración 75 – Salida detector de envolvente real cuando el dieléctrico es el aire en tiempo. 84

Índice de tablas

Tabla 1 – Tabla de costes 27
Tabla 2 – Datos prueba VCO 35
Tabla 3 – Tamaño de líneas de transmisión 61
Tabla 4 – Configuraciones de cada vía 61

Capítulo 1. INTRODUCCIÓN

Desde los inicios del desarrollo en temas de innovación para dispositivos electrónicos se ha llevado a cabo el uso de componentes activos. El gran inconveniente de los proyectos donde se utilizan estos componentes viene cuando tratamos de diseñar un controlador para medir un sensor. Los componentes activos requieren de la utilización de baterías integradas, o bien disponer de su propia fuente de alimentación. Además, dichos componentes activos no son desechables, por lo que se incrementa el coste de cada sensor. Este modelo de desarrollo necesita un diseño de un controlador específico para cada uno de los diferentes sensores, ya que la respuesta de cada uno de ellos es diferente.

Algunas soluciones que han optado por el uso de componentes activos pueden ser los dispositivos wereables, ya sean relojes, pulseras inteligentes o gafas entre otros. En otros casos de uso como, por ejemplo, la domótica de una casa o empresa es cada vez más frecuente el uso de sistemas de IoT para controlar luces, termostatos, cristales y cualquier sistema que requiera de una automatización o control externo. A raíz de este caso de uso ha surgido el desarrollo de una tecnología llamada Zigbee, un conjunto de protocolos de alto nivel de comunicación. Es una tecnología de comunicación inalámbrica que presume de sencillez y de un ahorro notable en los costes de producción.

Para evitar las desventajas que presentan los componentes activos, se requiere buscar una nueva arquitectura de desarrollo utilizando únicamente componentes pasivos. De esta idea surgen diversas tecnologías que dan respuesta a estos frentes, como el RFID, que utiliza componentes semi-activos o pasivos. Estos componentes semi-activos también poseen una fuente de alimentación propia. La diferencia está en el uso de la fuente para alimentar el microchip, y no para transmitir una señal.

La tendencia actual es buscar un diseño con componentes totalmente pasivos, como los resonadores, que consiguen homogeneizar un controlador electrónico capaz de medir los diferentes sensores usando técnicas de radiofrecuencia. Este dispositivo tiene además la capacidad de ser escalable, ya que desarrolla un software sobre un microprocesador que permite añadir diferentes módulos con tecnologías como ethernet, USB, etc… Por consiguiente, conseguimos también que el proyecto tenga un interés comercial.

El presente proyecto pretende dar una solución a este problema, mediante el desarrollo de un controlador en un controlador que permita medir mediante técnicas de radiofrecuencia un sensor de temperatura. La elección de realizar el desarrollo sobre un sensor de temperatura obedece a la necesidad de crear un prototipo que en un futuro se pueda implementar para diferentes tipos de sensores por igual, mediante el solo uso de componentes totalmente pasivos. Trataremos de realizar este proyecto conectando directamente el sensor mediante cables a nuestro lector, aunque en otro proyecto futuro pudiera ser escalable a utilizarlo sin conexión directa, como por ejemplo haciendo uso de antenas.

El controlador que se va a desarrollar en este proyecto se centra en la utilización de radiofrecuencia para poder obtener la respuesta del sensor. El microprocesador se encargará de generar una señal de diente de sierra. Esta señal será la que recibirá un oscilador controlado por tensión (VCO). La misión del VCO será la de transformar esta señal en una señal senoidal, cuya frecuencia será proporcional a la tensión del diente de sierra. La salida del VCO será alimentada un circuito que contará con un resonador de radiofrecuencia que tendrá integrado un termistor. El objetivo es que este circuito tenga una frecuencia de resonancia que dependa de la temperatura. Es decir, tendrá la respuesta característica de un filtro Notch.

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

2.1 MICROPROCESADOR KL25Z

Para la elección de utilizar un determinado microprocesador nos hemos basado primero en las necesidades de nuestro proyecto. Se requiere para ello un dispositivo capaz de:

1. Generar una señal de salida, de recibirla y de poder procesar la respuesta. Esto supone que necesitamos ciertos módulos como un conversor de digital a analógico (DAC) para enviar la señal y un conversor de analógico a digital (ADC) para recibirla.
2. Tener una velocidad de proceso y eficiencia alta. Hemos de coordinar los tiempos en los que suceden a la vez ciertos eventos: el generar una señal, recibir la respuesta del pulso generado en el ciclo anterior y almacenar y/o procesar la información.

El microprocesador elegido que cumple esa tecnología es el Freedom KL25Z como podemos observar en la Ilustración 7. Es una plataforma de desarrollo de bajo costo para los MCU (Microcontroller Unit) Kinetis construidos en el procesador Arm® Cortex®-M0.

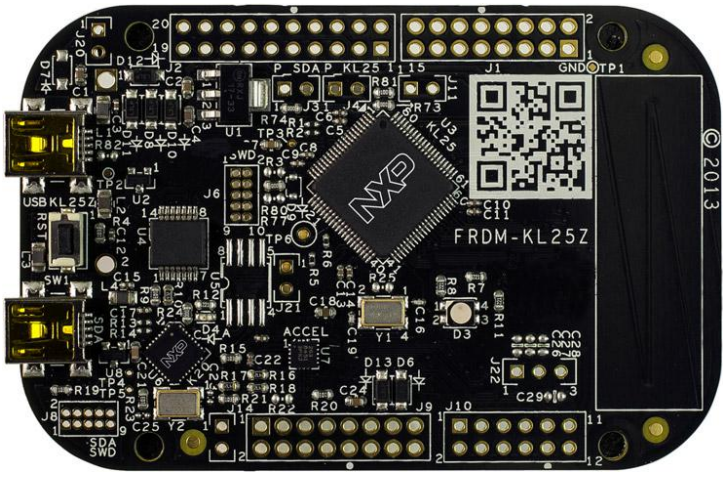


Ilustración 7 – Microprocesador KL25Z,

imagen sacada de la web oficial www.nxp.com

2.1.1 CONVERSOR DIGITAL / ANALÓGICO

Como hemos descrito en el primer punto necesitamos generar una señal desde el microprocesador. Para ello hacemos uso de un componente en el microprocesador que convierte una señal de digital a analógico, el DAC (digital-to-analog-converter). Se trata de un módulo capaz de convertir un código, generalmente binario, a una señal analógica (corriente, voltaje o carga eléctrica). En el caso del KL25z utiliza 12 bits para su conversión, que dan un máximo valor de 4.092 combinaciones, ofreciendo una salida máxima de 3'3 Voltios. Hace uso de dos registros de 8 bits (*DAC0_DATnL* y *DAC0_DATnH*), de los cuales el registro mas alto reserva 4 bits para control [7].

El segundo requisito del controlador ha de ser el de poder recibir la señal. Esto lo conseguimos mediante un módulo capaz de realizar la operación inversa que el DAC, es decir, convertir una señal analógica en una señal digital, el ADC (analogic-to-digital-converter). En el caso de nuestro microprocesador utiliza varios registros de control y configuración, pero para almacenar el resultado de la conversión utiliza un registro de 16 bits (*ADC0_Rn*) [9].

Estos dos módulos dependen de varios parámetros. El primero es su resolución, que depende del número de bits de entrada del conversor. Otro de los parámetros críticos es el tiempo de conversión que se necesita para efectuar el máximo cambio de tensión con un error mínimo en la resolución. También se requerirá analizar la tensión de referencia, puesto que está relacionada directamente con la de salida [9].

2.1.2 KINETIS SDK 2.0 API

Las características del MCU escogido le permiten: un fácil acceso a los periféricos de entrada y salida, que en nuestro caso se ve necesario por el uso del DAC y ADC, que los lleva incorporados; operaciones a baja energía, lo que permite un rápido desarrollo de aplicaciones integradas; escalabilidad con diferentes módulos estándar a utilizar, como Ethernet, USB y UART entre otros; y una depuración integrada para desarrollar mas eficientemente [11].

Se nos ofrece de un SDK (Software Development Kit) con su respectiva API (Application Programming Interface) para facilitar el uso de desarrollo del software de nuestro proyecto.

En la web oficial el SDK se define como una colección de software habilitado para microprocesadores Kinetis NXP que incluyen drivers, controladores de periféricos, paquetes y librerías para el desarrollo de software. Los elementos de configuración de cada uno de los drivers se encapsulan mediante estructuras de datos de lenguaje C. Esto se proporciona como parte del KSDK y no necesita ser modificado por el usuario [8].

De esta manera podemos utilizar los diferentes módulos del microprocesador abstrayendo el funcionamiento a alto nivel y, de esta manera, facilitar el desarrollo y las pruebas del proyecto. Podemos ver una explicación en la Ilustración siguiente, sacada de la página oficial de NxP.

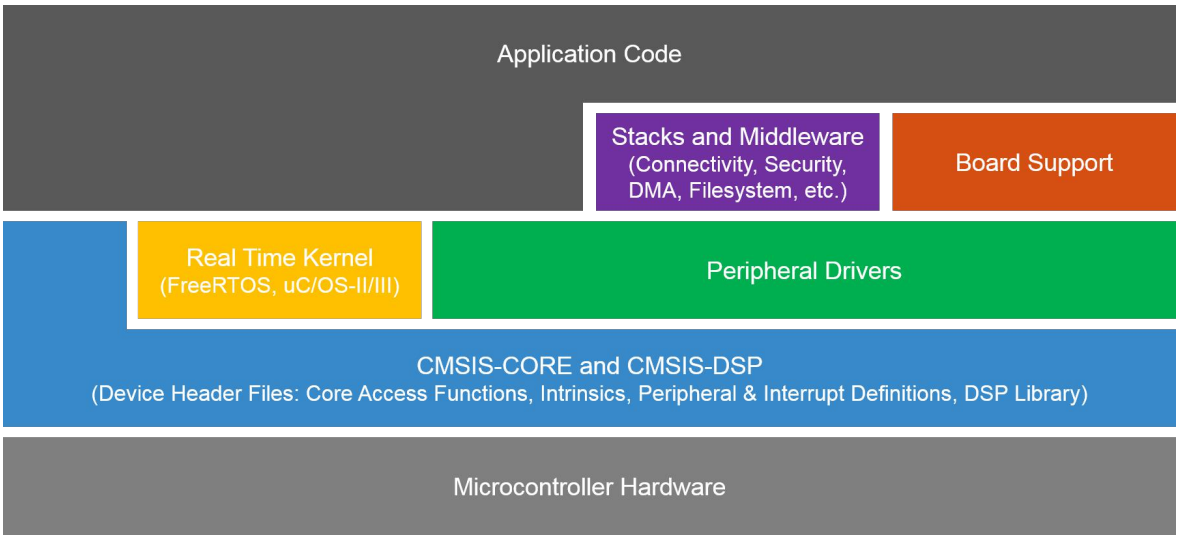


Ilustración 8 – Arquitectura Kinetis SDK 2.0 [8]

2.2 PROCESAMIENTO DIGITAL DE LA SEÑAL

2.2.1 OSCILADOR CONTROLADO POR TENSIÓN

Comúnmente conocido como VCO por sus siglas en inglés (Voltage-controlled oscillator). Es un dispositivo electrónico que hace uso de amplificación, realimentación y circuitos resonantes que da a su salida una señal eléctrica de frecuencia proporcional a una tensión de entrada tomada como referencia.

Los VCO se usan en generadores de funciones, bucles de fase sincronizada que incluyen sintetizadores de frecuencia utilizados en equipos de comunicación y la producción de música electrónica, para generar tonos variables en sintetizadores.

Los generadores de funciones son osciladores de baja frecuencia que presentan múltiples formas de onda, típicamente ondas sinusoidales, cuadradas y triangulares. Los generadores de funciones monolíticas están controlados por voltaje [10].

Uno de los factores que hemos de tener en cuenta en un VCO es el rango de frecuencias a las que actúa. La elección de un oscilador en un rango determinado es crucial, ya que depende a su vez del rango de frecuencias en las que actúe el sensor. Si el sensor elimina una frecuencia determinada que no se encuentra en el rango de frecuencias que el oscilador es capaz de proporcionar no funcionará correctamente nuestro diseño.

2.2.2 FILTRO NOTCH

Es un tipo de filtro que elimina las frecuencias que se encuentren entre la frecuencia de corte superior e inferior. Por tanto, la respuesta de este tipo de filtros es un paso banda. Es muy útil este tipo de filtros pues nos sirve para filtrar las frecuencias inferior y superior que nos limita el VCO en su rango de frecuencias de actuación. En nuestro proyecto utilizaremos esta técnica, pero únicamente para el sensor, puesto que el objetivo del sensor es tratar de eliminar una banda muy pequeña con respecto a la banda que recibe. Podemos observar este fenómeno en la siguiente ilustración.

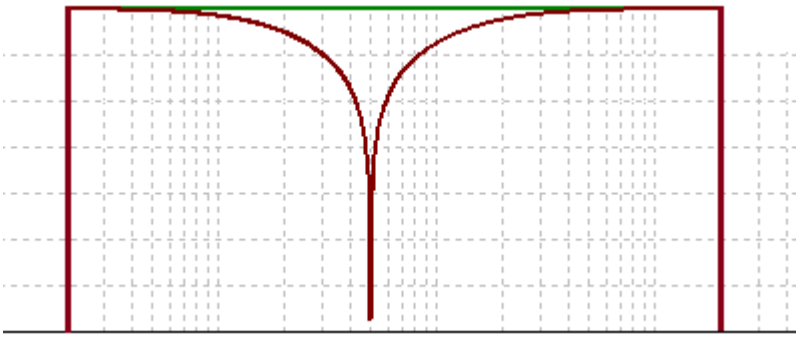


Ilustración 9 – Notch

2.3 DISEÑO DE PLACA PCB

También necesitamos construir una placa PCB para procesar todo el flujo de la señal, desde que inicia en el microprocesador hasta que lo recibe el mismo. La placa PCB contiene todos los componentes electrónicos necesarios para transformar la señal en otra que requiera el proyecto. Estos componentes pueden ser condensadores, diodos, filtros, detectores y dispositivos de entrada y salida entre otros. Existen muchas soluciones de software para realizar el diseño de la placa. En nuestro caso, hemos hecho del uso de una herramienta llamada Kicad EDA. Un software Open Source de automatización para el diseño de placas PCB [2].

2.3.1 KICAD EDA

Esta herramienta cuenta con tres principales características. La primera, es una herramienta para diseñar un esquema del circuito llamada Eeschema. De esta manera abstraemos la solución de una forma más sencilla de entender. Además, el propio software genera una serie de reglas para que sea menos complejo el diseño final de la placa PCB.

Otra herramienta que se facilita se llama Pcbnew. Sirve para realizar un diseño final desde el esquema realizado previamente. El usuario decidirá cómo y dónde se colocan los componentes que se vayan a utilizar, así como las dimensiones y las conexiones. Para ello

facilita poder realizar el diseño por capas. Esta característica es útil debido a que las conexiones entre los diferentes componentes podrán ser a través de las diferentes capas. De esta manera podemos evitar cruces entre las líneas y realizar un diseño óptimo. Veremos esta característica más adelante en cuanto a nuestro diseño.

Aunque el software contiene más herramientas, estas dos son las esenciales que necesitaremos utilizar para desarrollar la placa PCB.

Capítulo 3. ESTADO DE LA CUESTIÓN

Actualmente el mercado de los sensores está evolucionando y actualizándose continuamente, con el objetivo de buscar soluciones que no requieran de cables, sean baratos de fabricar y sean eficientes y precisos. Es por ello, que se desarrollan muchas soluciones a este gran problema que se diferencian en cuanto al objetivo final que persiguen, bien sea optimizar el coste y la utilización de los sensores, o la comunicación eficientemente entre ellos.
Para dar respuesta a este problema han surgido ciertas tecnologías, desde estudios e investigaciones en universidades hasta desarrollos por empresas o start-ups. Algunas de ellas basadas en la utilización de la radiofrecuencia y otras en cuanto a dar una solución IoT.

3.1.1 RADIO FREQUENCY IDENTIFICATION

Es un sistema de almacenamiento y recuperación de datos remotos que usa dispositivos denominados etiquetas o tarjetas RFID [6]. El objetivo de esta tecnología es transmitir a través de ondas de radio la identidad de un objeto, y para ello requieren de antenas. Las etiquetas son pequeños dispositivos que dan una respuesta por radiofrecuencia. Esta solución frente a otras como, por ejemplo, la utilización de infrarrojos es más demandada ya que no requiere de visión directa entre emisor y receptor. Podemos ver a continuación una ilustración de cómo funciona la tecnología RFID:

Ilustración 10 – Esquema funcionamiento RFID

Existen tres tipos de etiquetas en función de sus características: activas, semipasivas y pasivas.

- Las etiquetas activas hacen uso de una fuente de alimentación. Gracias a eso, son capaces de transmitir señales más potentes que las pasivas y pueden utilizarse en distancias mayores. El coste de fabricación sin embargo es más alto que el del resto de etiquetas que no son activas.
- Las etiquetas pasivas no requieren de alimentación eléctrica. Dan una respuesta al medio cuando reciben una corriente eléctrica pequeña, que es suficiente para la alimentación del sistema.
    - En el caso de esta tecnología, las antenas situadas en las etiquetas utilizan backscatter sobre la portadora. A este fenómeno se le llama retrodispersión [12], es decir, la reflexión de ondas radioeléctricas en dirección hacia el origen. De esta manera la energía recibida es suficiente como para recibir y enviar la señal. Una de las razones por las que en el mercado se solicita esta tecnología es que el coste de su fabricación es muy bajo.
- Las etiquetas semipasivas son una mezcla de los dos tipos de etiquetas anteriores. Requieren de una fuente de alimentación, pero solo para alimentar el microprocesador y no para transmitir la señal como sucede con las etiquetas pasivas.

Las bandas en las que RFID actúa son diferentes en función del tamaño que le queramos dar a la etiqueta y de la banda que queramos utilizar [5]:

- Baja frecuencia (LW): Rango de hasta 50cm y opera en las frecuencias de 125 KHz.
- Alta frecuencia (HW): Rango de hasta 8cm y opera en las frecuencias de 13,56 MHz.
- Ultra Alta frecuencia (UHF): Rango de 3m a 10m y opera en las frecuencias de 400 MHz – 1000MHz.
- Microondas: Rango de mas de 10m y opera en las frecuencias 2,45 GHz – 5,4 GHz

3.1.2 ZIGBEE

El principal objetivo a la hora de desarrollar esta tecnología fue el uso en la domótica. Por ello su desarrollo se basa en el estándar de comunicaciones inalámbricas existentes actualmente como IEEE 802.15.4. Se trata de un conjunto de protocolos de alto nivel para comunicaciones inalámbricas. Se utiliza con radiodifusión digital de bajo consumo para redes inalámbricas de área personal [13]. Sus principales características son su bajo consumo, su topología en red de malla y su fácil integración.

Con ZigBee HA (Home Automation) se ofrece a los fabricantes, integradores, desarrolladores, etc., una opción de trabajar bajo un enfoque basado en estándares a la hora de introducir los nuevos productos dedicados a la domótica o automatización del hogar, eliminando así la necesidad de hacerlo sobre tecnología patentada.

Esta nueva aplicación, definida por la propia ZiBee Alliance como el nuevo estándar global para la automatización del hogar, permite que las aplicaciones domóticas desarrolladas por los fabricantes sean completamente interoperables entre sí, garantizando así al cliente final fiabilidad, control, seguridad y comodidad [14].

Las distintas tecnologías existentes no fueron pensadas para la domótica, por lo que las características que se ofrecen en ellas no satisfacen las necesidades requeridas:

- El ancho de banda utilizado para Zigbee es de 250 Kbps, mientras que para Wi-Fi son 54 Mbps y Bluetooth son 1 Mbps.
- El consumo de potencia es de 30ma trasmitiendo y 3ma en reposo. Mientras que para Wi-Fi son de 400ma transmitiendo y 20ma en reposo y para Bluetooth son 40 ma transmitiendo y 0.2ma en reposo.
- Zigbee posee una batería de larga duración y bajo coste, mientras que Wi-Fi su gran característica es su gran ancho de banda y Bluetooth su interoperabilidad.

3.1.3 SPLIT-RING RESONATOR (SRR)-BASED SENSOR

Un resonador de anillo dividido (SRR) es una estructura producida artificialmente común a los meta-materiales [15]. Su propósito es producir la susceptibilidad magnética deseada (respuesta magnética) en varios tipos de metamateriales de hasta 200 terahercios. Estos medios crean el acoplamiento magnético fuerte necesario para un campo electromagnético aplicado, que de otro modo no estaría disponible en los materiales convencionales. Por ejemplo, se produce un efecto tal como permeabilidad negativa con una matriz periódica de resonadores de anillo dividido.

- Consisten en un par de concéntricos anillos metálicos, grabado en un dieléctrico de sustrato, con ranuras grabadas en lados opuestos.
- Los SRR pueden producir un efecto de ser eléctricamente más pequeños cuando responden a un campo electromagnético oscilante. Estos resonadores se han utilizado para la síntesis de medios de índice de refracción izquierdo y negativo, donde el valor necesario de la permeabilidad efectiva negativa se debe a la presencia de los SRR [16].
- Cuando se excita una matriz de SRR eléctricamente pequeñas por medio de un campo magnético variable en el tiempo, la estructura se comporta como un medio efectivo con una permeabilidad negativa efectiva en una banda estrecha por encima de la resonancia SRR. Los SRR también se han acoplado a líneas de transmisión plana, para la síntesis de metamateriales de líneas de transmisión [16].

En marzo de 2016 se publicó una investigación de un sensor basado en esta tecnología [17]. Se trata de un sensor basado en resonador de anillo dividido (SRR) para la detección del espesor sólido y la caracterización de permitividad relativa de los materiales sólidos y líquidos.

La estructura se compone de dos SRR alojados en una línea de transmisión microbanda. El principio de detección se basa en la detección de la muesca introducida por los resonadores en el coeficiente de transmisión. Por lo tanto, un cambio de frecuencia de la muesca está relacionado con un cambio en la permitividad efectiva de la estructura cuando el sensor está cubierto con cualquier material sólido o líquido. Una caracterización completa del sensor, para las tres aplicaciones propuestas, se realiza a través de simulaciones. Finalmente, todos los resultados simulados se corroboran con mediciones. El sensor propuesto se implementa en tecnología impresa de capa única, lo que resulta en una solución de bajo costo y baja complejidad. Presenta respuesta en tiempo real y alta sensibilidad. Además, es totalmente sumergible y reutilizable [17].

Capítulo 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

Este proyecto se centra en el desarrollo de un lector motivado por la utilización principalmente de componentes pasivos que pueda obtener la información del sensor basado en anillo divido, de acuerdo con lo explicado en el anterior apartado.

La justificación de la utilidad de este tipo de lectores, es que sus componentes no requieren fuentes de alimentación, lo cual les hace especialmente interesantes desde el punto de vista del mercado, ya que al no necesitar esas fuentes pueden ser desechables y reutilizables, disminuyendo los costes de fabricación. Por todo ello, el uso de sensores basados en esta tecnología es cada vez más común, y su desarrollo más solicitado.

Por ello nos propusimos desarrollar un dispositivo generalizado que pudiera leer las respuestas de estos sensores en frecuencia. El desarrollo de este dispositivo se realiza mediante un microprocesador. La elección de utilizar este microprocesador es por la necesidad de generar una señal, de recibirla y procesarla. Además, se le pueden añadir módulos variados en función de las necesidades que requiera el proyecto.

Este lector se diferencia con respecto al resto en que se utiliza radiofrecuencia, lo cual evita la necesidad de utilizar protocolos y tecnologías ya existentes, que limitan el uso del canal. Es por ello por lo que, al utilizar radiofrecuencia, podríamos extender el proyecto a hacer inalámbrica nuestra señal, utilizando un sensor basado en anillo dividido, así como también añadir antenas que no requieran de una batería.

4.2 OBJETIVOS

El objetivo principal de este trabajo era desarrollar un lector para leer un sensor de temperatura mediante radiofrecuencia. Debido a ciertos problemas con el sensor de temperatura, se reconfiguró posteriormente el proyecto para realizar un lector, hacia el desarrollo de un sensor de dieléctricos basado en radiofrecuencia. La base y el desarrollo del proyecto es el mismo, salvo que el sensor se redefinió. Por lo que los objetivos siguen siendo los mismos.

Podemos subdividir el objetivo principal en otros dos:

- Desarrollo del software del microcontrolador:
    - Inicialmente desarrollaremos un código basado en lenguaje C para generar las funciones necesarias para enviar y recibir nuestra señal.
    - A continuación, desarrollaremos el software necesario para generar una señal que sea un diente de sierra.
    - Posteriormente recibiremos la señal del sensor pasando por le filtro y el detector de envolvente.
    - Pruebas correspondientes a estas funciones desarrolladas.

- Desarrollo y Diseño de una placa PCB para el procesamiento de la señal. La placa por diseñar llevará el flujo de la señal desde que la enviamos hasta que la recibimos:
    - Probar el funcionamiento de los componentes por separado:
        - VCO
        - Sensor
        - Detector y Filtro
    - Diseñar el esquema del circuito.
    - Probar cada etapa por separado y comprobar el resultado.

4.3 METODOLOGÍA

Las herramientas utilizadas para este proyecto han sido diferentes en función de las necesidades de cada uno de los objetivos. Por este motivo definiremos los caminos seguidos en los dos grandes bloques de nuestro proyecto:

Para el desarrollo software en el microprocesador tenemos que primero saber que controlador vamos a utilizar. En nuestro caso se trata del KL25Z [18]:

- Como IDE (Entorno de Desarrollo Integrado) se nos facilita utilizar Kinetis Design Studio. Es un entorno de desarrollo integrado gratuito que permite la edición, compilación y depuración de sus diseños. Por otra parte, Processor Expert ® software permite a su diseño con su base de conocimientos y ayuda a crear aplicaciones potentes con unos clics del ratón [1].
- Para la compilación y la utilización de los drivers del microprocesador se hacen uso del SDK [8] facilitado por Nxp.
- Para las pruebas utilizaremos la herramienta de Debug que nos ofrece el IDE. De esta manera podemos comprobar fácilmente que sucede en cualquier punto de nuestro código.

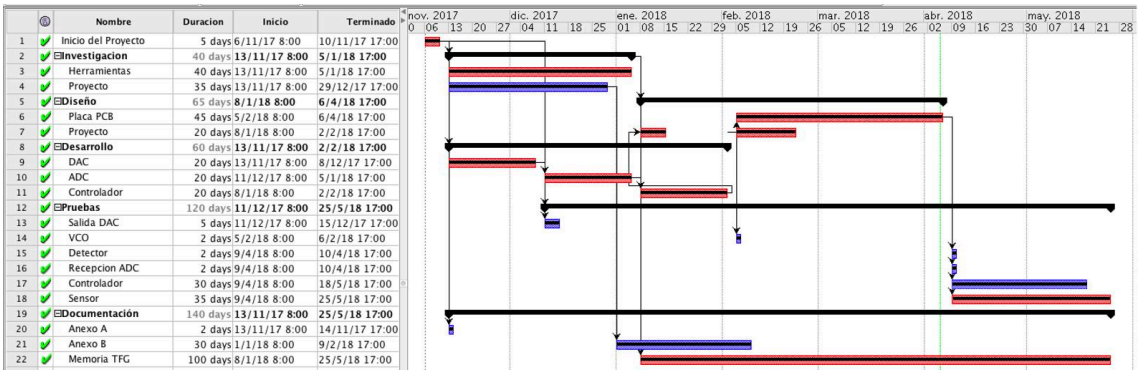
Para el desarrollo y diseño de la placa PCB que generará el flujo de nuestro circuito utilizaremos una herramienta llamada Kicad.

- Lo primero será realizar un esquema de nuestro circuito mediante la herramienta Eesquema o Schematic Capture [19].
- Para la transformación de la abstracción del esquema a un diseño real hacemos uso de la herramienta PcbNew [20].
- Para la comprobación de todo el flujo de la señal utilizaremos osciloscopios y analizadores de espectros.
- Haremos uso también de los generadores de señal, para poder comprobar también la diferencia entre la señal de entrada generada por el micro y una señal pura.

4.4 PLANIFICACIÓN Y ESTIMACIÓN ECONÓMICA

4.4.1 PLANIFICACIÓN

Al comienzo del proyecto se estableció una planificación de las tareas a realizar, que se muestra a continuación:



	Nombre	Duracion	Inicio	Terminado
1	Inicio del Proyecto	5 days	6/11/17 8:00	10/11/17 17:00
2	**Investigacion**	40 days	**13/11/17 8:00**	**5/1/18 17:00**
3	Herramientas	40 days	13/11/17 8:00	5/1/18 17:00
4	Proyecto	35 days	13/11/17 8:00	29/12/17 17:00
5	**Diseño**	65 days	**8/1/18 8:00**	**6/4/18 17:00**
6	Placa PCB	45 days	5/2/18 8:00	6/4/18 17:00
7	Proyecto	20 days	8/1/18 8:00	2/2/18 17:00
8	**Desarrollo**	60 days	**13/11/17 8:00**	**2/2/18 17:00**
9	DAC	20 days	13/11/17 8:00	8/12/17 17:00
10	ADC	20 days	11/12/17 8:00	5/1/18 17:00
11	Controlador	20 days	8/1/18 8:00	2/2/18 17:00
12	**Pruebas**	120 days	**11/12/17 8:00**	**25/5/18 17:00**
13	Salida DAC	5 days	11/12/17 8:00	15/12/17 17:00
14	VCO	2 days	5/2/18 8:00	6/2/18 17:00
15	Detector	2 days	9/4/18 8:00	10/4/18 17:00
16	Recepcion ADC	2 days	9/4/18 8:00	10/4/18 17:00
17	Controlador	30 days	9/4/18 8:00	18/5/18 17:00
18	Sensor	35 days	9/4/18 8:00	25/5/18 17:00
19	**Documentación**	140 days	**13/11/17 8:00**	**25/5/18 17:00**
20	Anexo A	2 days	13/11/17 8:00	14/11/17 17:00
21	Anexo B	30 days	1/1/18 8:00	9/2/18 17:00
22	Memoria TFG	100 days	8/1/18 8:00	25/5/18 17:00

Ilustración 11 – Planificación del proyecto

Lo primero que se realizó en el proyecto fue la investigación previa del estado del arte. Aquí se realizaron las búsquedas de artículos y de tecnologías del mercado relativos a nuestro proyecto. También se realizó una investigación relativas a las herramientas cuya utilización iba a ser necesaria a lo largo del proyecto.

La primera etapa del diseño y desarrollo se centró en el microprocesador. Las etapas de desarrollo de software en el controlador fueron tres: diseñar el funcionamiento del conversor de digital a analógico (DAC), diseñar el conversor analógico a digital (ADC) y la lógica de lectura y escritura de los pines mediante timers. Finalmente realizar las respectivas pruebas con cada componente. Una vez acabada esa etapa se propuso comenzar el diseño de la placa PCB. Una vez finalizado el diseño de la producción de la placa PCB se utilizaron los test y pruebas necesarias para comprobar si correcto funcionamiento.

ESTIMACIÓN ECONÓMICA

A continuación, presentamos una estimación económica de lo usado en este proyecto:

El coste total es de 63,693 €.

Componente	*Cantidad*	*Precio 1 ud.*	*Precio Total*	*Referencia*
VCO	1	21,29 €	21,29 €	549-CV55CW1000-1500
Detector	1	0,103 €	0,103 €	HSMS-2850-TR1G
Sensor SSR	1	15 €	15 €	
Placa PCB	1	15 €	15 €	
Microprocesador Freescale KL25z	1	12,30 €	12,30 €	841-FREEDOM-KL25Z
	PRECIO TOTAL		63,693 €	

Tabla 1 – Tabla de costes

Capítulo 5. ANÁLISIS SISTEMA/MODELO

El modelo consta de tres fases principalmente. La primera de análisis de los requerimientos, la segunda de desarrollo del software del controlador y la tercera de diseño de la placa PCB.

Podemos ver el modelo global de nuestro proyecto en la siguiente ilustración:

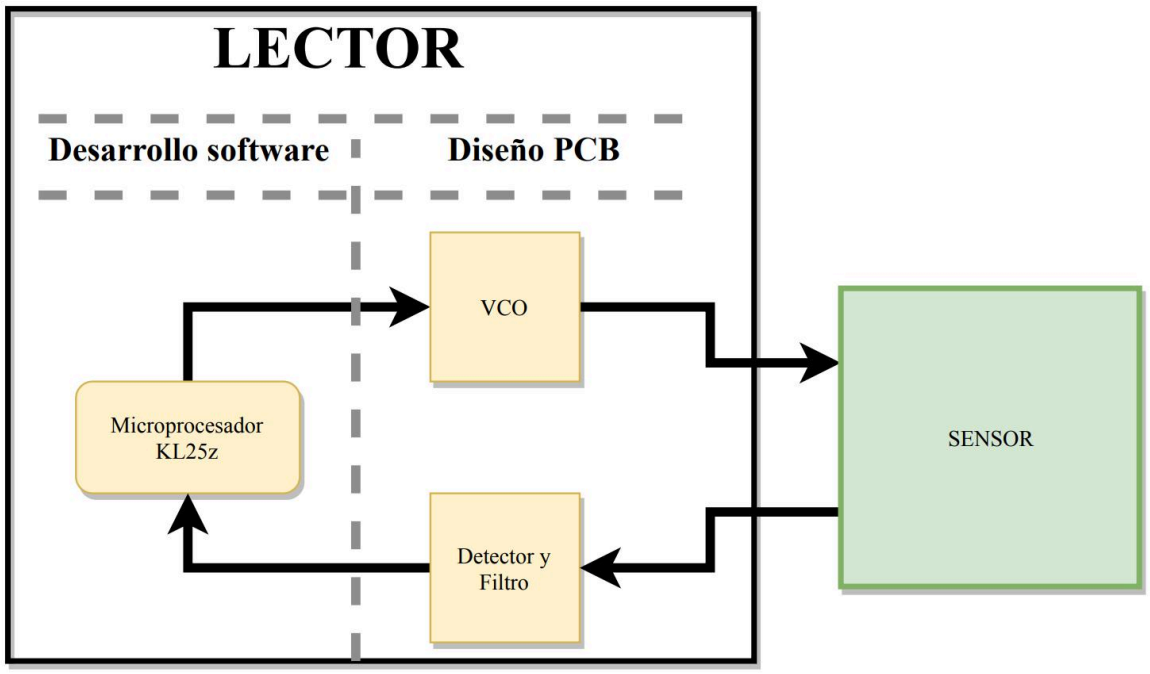


Ilustración 12 – Esquema del modelo del trabajo propuesto

A continuación, analizaremos el sistema, así como los requerimientos totales de nuestro proyecto.

5.1 ANÁLISIS DEL SISTEMA

Para el desarrollo del lector hemos primero de analizar el flujo que va a recorrer la señal por el sistema. Para ver las necesidades primero tenemos que analizar el funcionamiento del sensor. En función de su comportamiento de la respuesta dicho sensor diseñaremos el resto de componentes.

5.1.1 SENSOR BASADO EN UN RESONADOR DE ANILLO DIVIDIDO.

Como hemos explicado en el punto 3.1.3, el sensor genera una respuesta en función del dieléctrico situado cerca. La pregunta que nos deberíamos de hacer sería qué tipo de respuesta nos da este sensor. Para el funcionamiento del sensor se necesita a la entrada de un pulso con un ancho de banda fijo en frecuencia, tal y como vemos en la Ilustración 13. El rango de frecuencias del ancho de banda se situará entre 300 y 700 MHz, en función de los componentes de nuestro sistema.

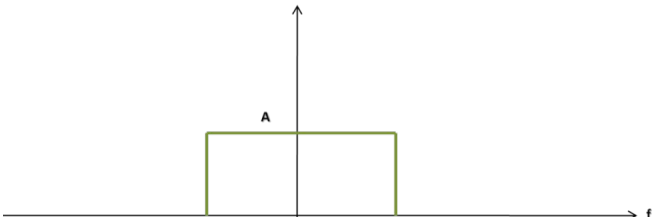


Ilustración 13 – Pulso en frecuencia

Básicamente el sensor actúa como si fuera un filtro notch o paso banda, eliminando un rango de frecuencias con un ancho de banda mucho menor al del pulso introducido. Como el diseño del sensor no entra dentro del alcance de este proyecto, tenemos que adaptar el proyecto al rango de frecuencias de actuación del sensor utilizado, por lo que hemos de conocer sus respuestas a distintas entradas. Por defecto, el dieléctrico que se sitúa en el sensor es el aire, por lo que se toma este dato como referencia para saber a qué frecuencia el sensor da una respuesta. Podemos observar un ejemplo en la siguiente ilustración:

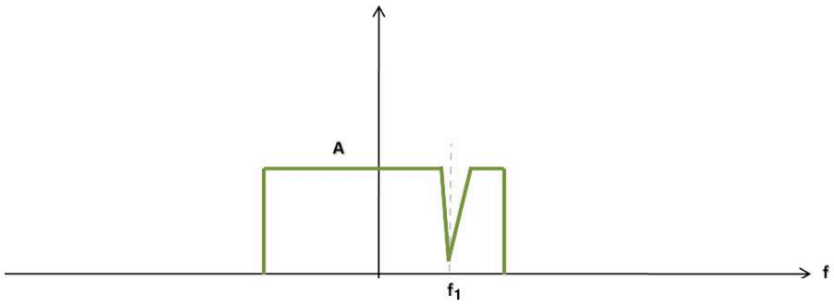


Ilustración 14 – Esquema de la salida del sensor

Aparentemente parece que el sensor elimina una frecuencia en particular. Pero en realidad elimina un ancho de banda muy pequeño si lo comparamos con el pulso que recibe. Si el ancho de banda es 600 MHz, el ancho de banda alrededor de la frecuencia f_1 será de 100 KHz. Estos datos son para comprender lo que sucede de forma abstracta. Posteriormente abordaremos con datos reales nuestro diseño.

El sensor dado, cuando el dieléctrico utilizado es el aire, tiene una frecuencia de actuación alrededor de la frecuencia de 1,48 GHz. Este dato hemos de corroborarlo con distintas pruebas que hagamos al sensor. Estos datos son necesarios para poder construir nuestro diseño del proyecto, por lo que el siguiente paso consistirá en como generar un pulso que englobe esta frecuencia. Es por ello por lo que para la generación de ese pulso utilizaremos un oscilador controlado por tensión.

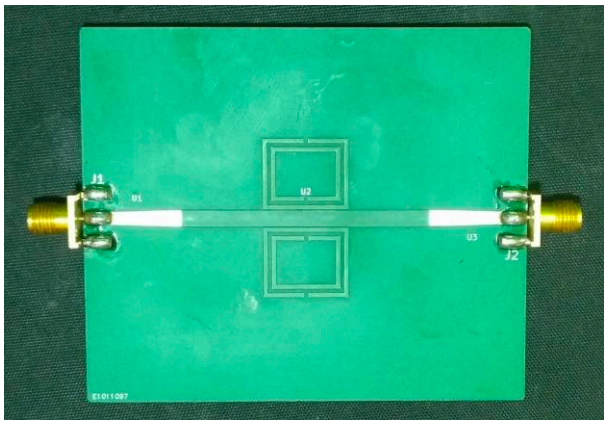


Ilustración 15 – Sensor

5.1.2 OSCILADOR CONTROLADO POR TENSIÓN

El VCO utilizado será el modelo de Crystek CVCO55CW. La elección de este dispositivo es debido a su rango de frecuencias de su curva de ajuste, que se sitúa entre los 900MHz y los 1.600MHz [21]. A continuación, observamos esta curva de ajuste del datasheet en la página oficial de Crystek.

Ilustración 16 – CVCO55CW-1000-1500

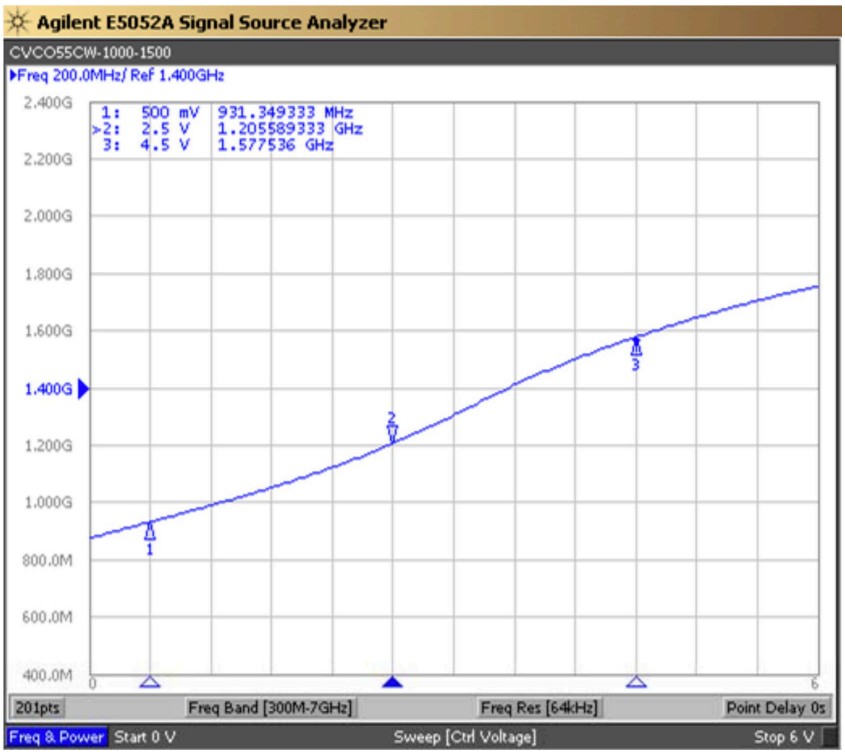


Ilustración 17 – Curva de ajuste CVCO55CW-1000-1500-38335

Estas características son muy adecuadas para nuestro sensor, ya que incorpora la frecuencia a la que nuestro sensor actúa. Primero vamos a estudiar su comportamiento. El funcionamiento de un VCO está explicado en el apartado 2.2.1. Este dispositivo tiene 4 entradas:

- Vt: La entrada de la señal principal que se desea transformar. Lo que este dispositivo tiene en cuenta de la señal de entrada es su tensión.
- RF: Es la salida del VCO. La respuesta de esta salida variará en función de la tensión de entrada.
- Vcc: Es la tensión de referencia del VCO. Según las características del componente hemos de darle una tensión de 3,3 Voltios.
- Tierra: Esta conectada a la carcasa del componente.

Realizamos las pruebas convenientes para comprobar que los datos del datasheet facilitado por el fabricante se ajustan a la realidad. Para ello hicimos uso del analizador de espectros. Comenzamos las pruebas conectando a la entrada una señal con tensión la de Tierra. A una determinada tensión encontramos lo que observamos en la Ilustración 18, una delta con el primer armónico que es el mas importante y el central, y dos segundos armónicos a los laterales con menos potencia de señal. Después de esto, anotamos los valores que recibimos a la salida que aparecen en el analizador de espectro. Los respectivos resultados se muestran en la siguiente tabla:

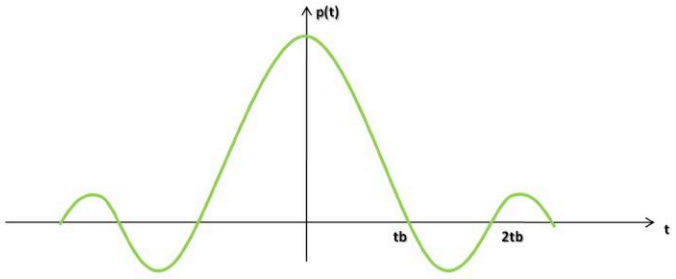


Ilustración 18 – Respuesta VCO a una determinada tensión de entrada en el VCO

(1er armónico el central, y 2do armónico los laterales)

Tensión Vcc	Tensión Vt	1º Armónico [Mhz]	Valor primer armónico [dbm]	2º armónico [Mhz]	Valor segundo armónico [dbm]
3,01	0,1252	846,0402	-1,4	850,2716	-34,24
3,01	0,1868	846,7226	-2,16	850,6811	-34,25
3	0,2411	848,9066	-2,65	853,2062	-34,25
3	0,2892	851,0905	-3,4	855,4584	-35,39
3	0,3438	854,1617	-4,61	858,7344	-33,82
3,01	0,3787	856,2092	-5,41	860,7818	-35,11
3,02	0,4226	881,0168	-4,45	885,2440	-40,44
3	0,4811	882,8204	-3,39	886,9349	-36,83
3,03	0,5532	889,5275	-2,54	893,9801	-27,15
3,04	0,5992	897,6595	-3,2	901,7735	-26,45
3,01	0,6473	904,5522	-2,85	908,9187	-26,16
3	0,7019	909,5322	-2,74	913,5562	-38,45
3	0,7389	912,1315	-3	916,3090	-35,86
2,99	0,7933	915,0291	-3,23	919,2307	-34,35
3,02	0,863	920,2690	-2,92	924,7361	-32,43
3,03	0,9088	923,4322	-3,11	927,6096	-33,44
3,03	0,9546	927,2956	-3,37	931,4489	-33,08
3,03	1,0048	932,0042	-3,84	936,1575	-32,48
3,04	1,0611	938,6929	-3,97	942,8220	-29,29
3,05	1,1263	953,3501	-3	957,6000	-18,53
3,04	1,1656	966,7033	-1,96	970,8083	-23,47
3,04	1,2081	974,2130	-2	978,3180	-26,92
3,03	1,2573	981,2880	-1,98	985,3930	-28,17
3,03	1,2951	985,3930	-2,14	989,5221	-30,63
3,03	1,3316	989,3531	-2,3	993,6030	-29,79
3,03	1,3906	995,5106	-2,77	999,6397	-32,13
3,03	1,4512	1002,1993	-3,48	1006,4008	-30,53
3,04	1,4865	1006,4250	-3,97	1010,6748	-30,61
3,04	1,5301	1013,0412	-4,03	1017,2187	-28,26
3,04	1,5778	1021,5410	-3,33	1025,7425	-25,66
3,03	1,6459	1032,8417	-3,09	1037,0433	-27,77
3,03	1,6961	1040,4963	-3,68	1044,6254	-28,39
3,03	1,7631	1051,2175	-5,07	1055,3466	-30,2
3,04	1,8072	1058,7996	-6,05	1062,9529	-31,09
3,04	1,8527	1066,8647	-6,91	1070,9938	-31,89

Tensión Vcc	Tensión Vt	1º Armónico [Mhz]	Valor primer armónico [dbm]	2º Armónico [Mhz]	Valor segundo armónico [dbm]
3,04	2,0379	1101,9985	-9,05	1106,1759	-31,99
3,04	2,0853	1111,4399	-8,4	1115,5690	-33,5
3,04	2,1403	1122,5233	-7,61	1126,7732	-28,51
3,04	2,1946	1133,2687	-7	1137,4944	-30,36
3,04	2,2447	1142,5894	-7,45	1146,7186	-31,62
3,04	2,2949	1152,4172	-8,63	1156,5946	-31,14
3,04	2,3663	1166,2293	-10,11	1170,4308	-32,53
3,04	2,4081	1174,4875	-10,68	1178,7132	-32,71
3,05	2,4605	1184,8224	-10,62	1188,9998	-32,64
3,05	2,5231	1197,1615	-8,88	1201,3389	-30,67
3,05	2,5714	1206,6753	-7,29	1210,8769	-29,71
3,04	2,6216	1216,1651	-6,43	1220,4149	-29,46
3,04	2,6674	1224,9787	-7,32	1229,1561	-30,62
3,04	2,7633	1245,0448	-11,49	1249,2946	-33,38
3,04	2,7953	1252,4579	-12,76	1256,7077	-31,68
3,05	2,8541	1266,2216	-13,71	1270,3266	-33,54
3,05	2,9262	1282,6657	-12,64	1286,7706	-32,95
3,05	2,9741	1292,9523	-11,05	1297,0089	-34,19
3,05	3,0112	1300,5349	-10,2	1304,7601	-33,45

Tabla 2 – Datos prueba VCO

De esta tabla podemos sacar ciertas gráficas que nos ayudarán a comprender mejor el comportamiento de este dispositivo. Comparamos primero los valores de la potencia de las deltas en función de la frecuencia de los primeros armónicos. Podemos observarlo en la siguiente ilustración.

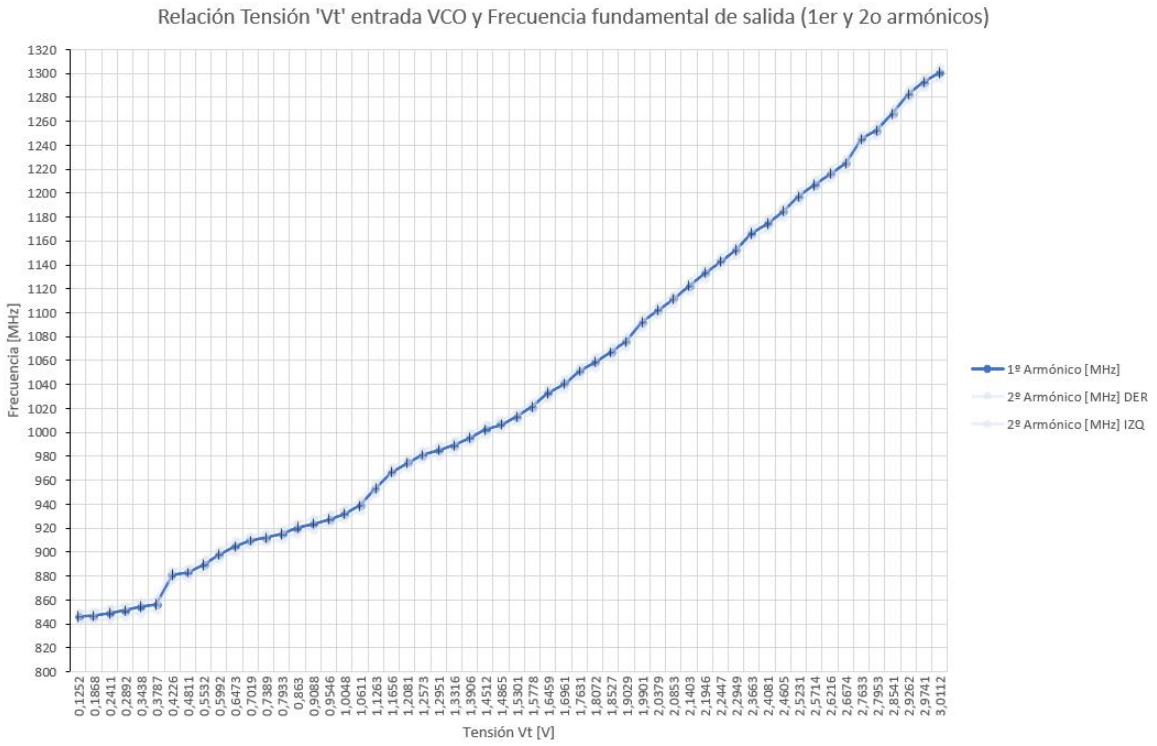


Ilustración 19 – Deltas del primer armónico en función de la frecuencia y la tensión de entrada.

La gráfica muestra una linealidad en el comportamiento del VCO, de forma muy aproximada a lo indicado en el datasheet. Este paso se hizo previamente a conocer la frecuencia de actuación del dieléctrico del aire, por lo que no se cogieron muestras que englobaran frecuencias mas altas.

En un principio se estableció una tensión de entrada de 3 voltios como máxima, aunque posteriormente veremos que es necesaria una entrada mucho mayor para que funcione correctamente el sensor.

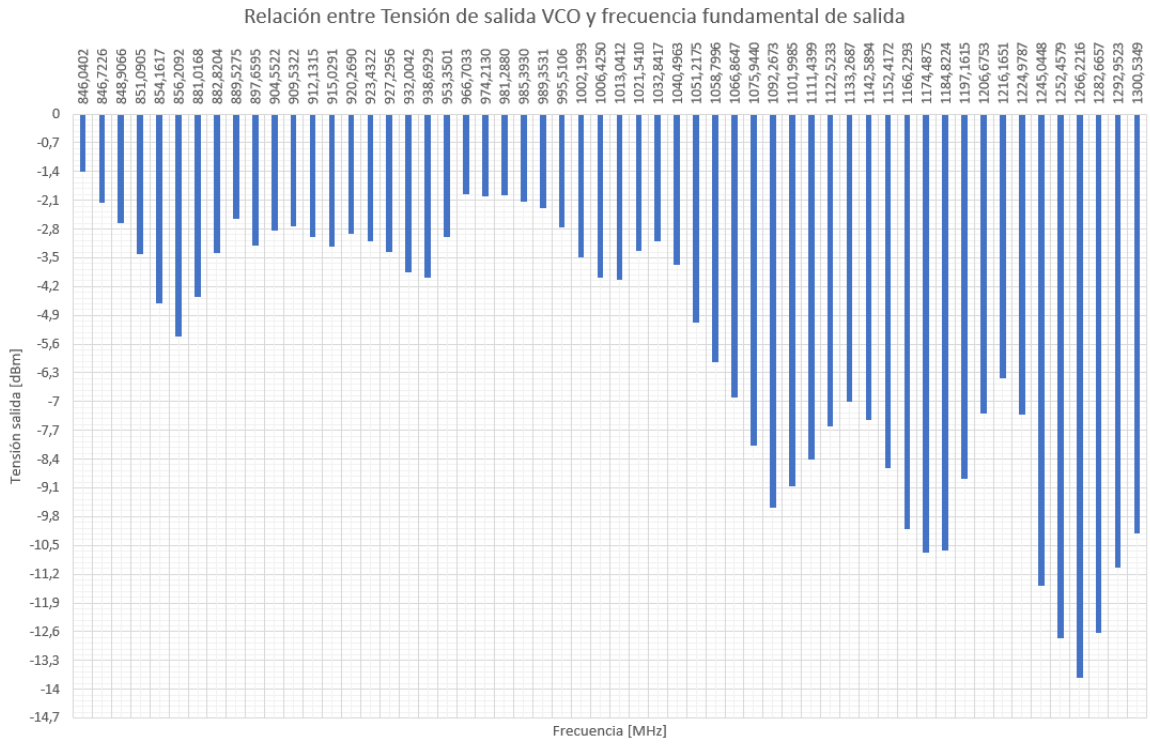


Ilustración 20 – Relación entre tensión de salida VCO y frecuencia fundamental de salida

También observamos que a medida que aumentamos la tensión de entrada disminuye la potencia de la señal de salida. Esto es debido a que el VCO se alimenta a 3,3 voltios, por lo que, si aumentamos progresivamente la tensión de entrada, la salida dispondrá de menos decibelios de amplitud.

Podemos observar que éste es un comportamiento raro. Para ello nos apoyamos de otra gráfica que nos ayuda a comprender porque existe tanta variación de potencia a la salida.

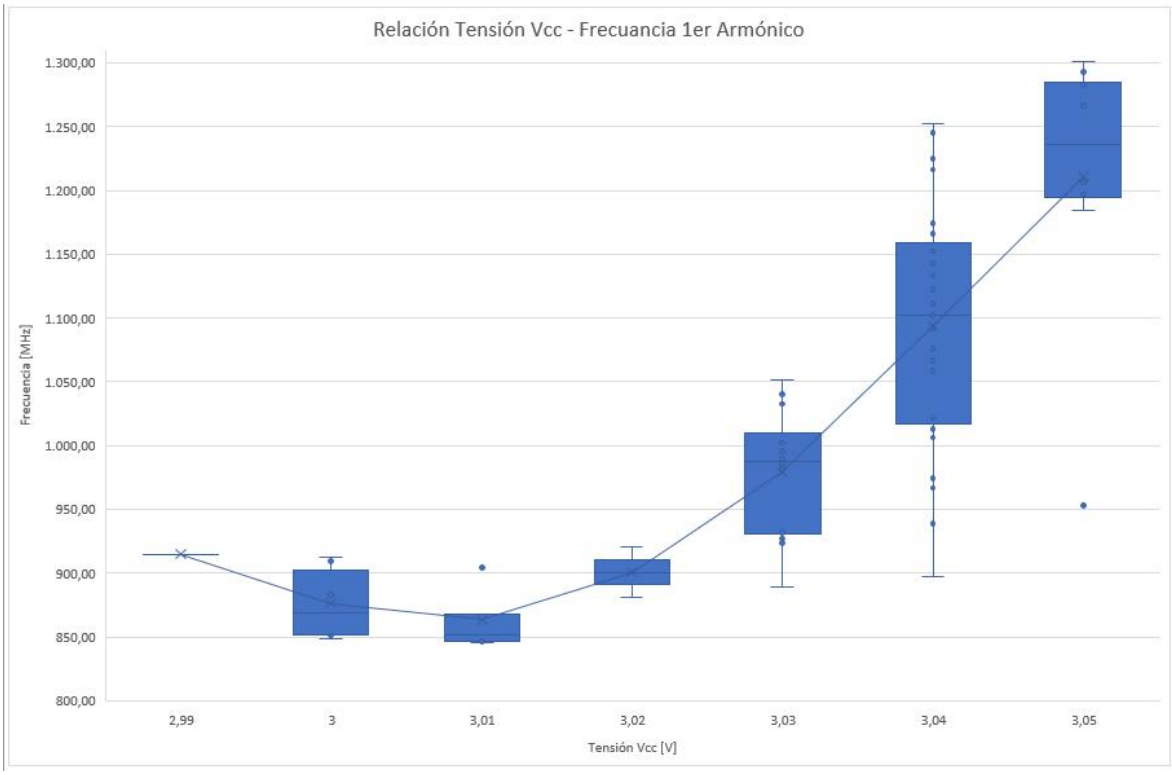


Ilustración 21 – Variación entre tensión Vcc de referencia y la frecuencia de saldia

Podemos observar que cuando el VCO ha de transformar la señal de entrada en una delta a la salida en frecuencia varía en función de si requiere mas tensión para convertir esa señal. A medida que tiene que aumentar la señal de salida del VCO, aumenta también la tensión de referencia Vcc. Suponemos que esto se debe a que requiere de más energía para convertir esas señales y hace que aumente la fuente de tensión.

Los valores de la tensión de referencia tienen poca dispersión en torno a frecuencias altas y frecuencias bajas, pero se comporta de forma diferente en torno a 3,04. No es grave ni supone un problema, puesto que son crecimientos en la tensión despreciables.

5.1.2.1 Finalidad VCO

Como hemos comentado en la introducción del punto 5.1.1 hemos de conseguir un pulso a la entrada del sensor. El VCO ya hemos explicado que dada una tensión de entrada genera una delta en frecuencia a la salida. Es por ello por lo que, si en un ciclo corto de tiempo, aumentamos progresivamente la tensión de entrada, generaremos deltas en función de ese periodo de tiempo. Podemos ver este fenómeno en la siguiente captura tomada desde el analizador de espectros:

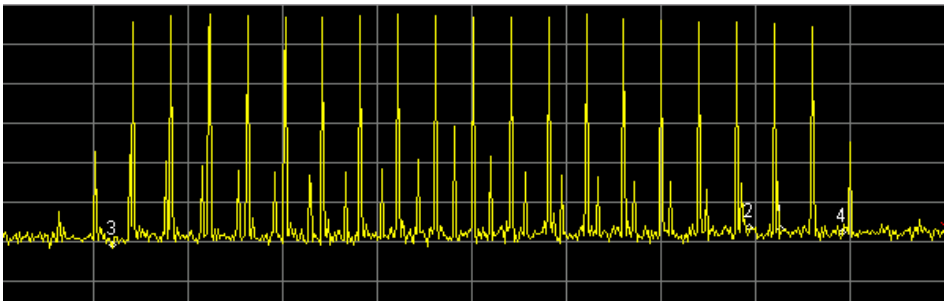


Ilustración 22 – Tren de deltas

Si aumentamos el sweep del analizador a 1 segundo por ejemplo (una función para reducir o aumentar el tiempo en el que se toma cada muestra de la señal) podemos observar los siguiente:

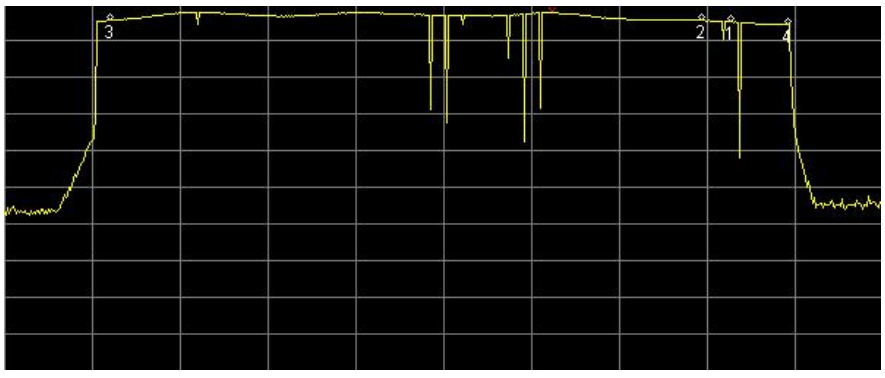


Ilustración 23 – Tren de deltas anterior, pero con sweep 1 segundo

Esto significa que podemos generar pulsos si introducimos a la entrada del VCO una señal de entrada de tensión variable ascendente con el tiempo. La señal, por tanto, que buscamos se trata de un diente de sierra. Es una señal que tiene una tensión con pendiente positiva o negativa (en nuestro caso necesitaremos que sea positiva) que aumente su tensión con el transcurso del tiempo. Mas adelante detallaremos los tiempos que se necesitan, cuando enfoquemos el apartado del microprocesador. Esto es debido a que dependemos del timer del controlador.

5.1.3 FILTRO Y DETECTOR DE ENVOLVENTE

El filtro y el detector de envolvente se encargan de la tarea de serializar la señal, es decir, de convertir la salida del sensor en frecuencia en la que actúa el notch del sensor, en tiempo. De esta manera, el notch que vemos en frecuencia podemos observarlo en tiempo, por lo que podremos categorizar en función del tiempo, que es lo que finalmente vamos a poder leer en el ADC del micro. El diodo utilizado es HSMS-2850 [22].

El filtro lo realizamos en las frecuencias de actuación del VCO para eliminar posibles ruidos e interferencias. El ancho de banda debe de ir entre 800 MHz y 1600 MHz. En la siguiente ilustración podemos observar un esquema del filtro y el detector de envolvente implementado.

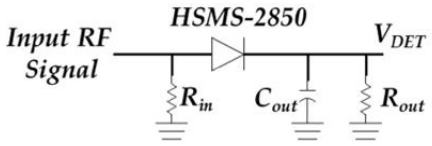


Ilustración 24 – Esquema del filtro y detector de envolvente implementado

Los valores de las resistencias son:

- R_{in}: 33 Ω
- R_{out}: 2,2 KΩ
- C_{out}: 1nF

Capítulo 6. SISTEMA/MODELO DESARROLLADO

Una vez analizado el sistema nos disponemos a explicar el diseño de nuestro sistema. Para ello hemos de dividir nuestro proyecto en dos partes. La primera fase se refiere al desarrollo del software del controlador. La segunda fase comprende el de diseño de la placa PCB por la que va a circular la señal hasta llegar al sensor, obtener su respuesta y volver para ser recibida por el microprocesador.

6.1 DESARROLLO SOFTWARE

En el punto 0 hemos explicado el porqué de la necesidad de generar un diente de sierra, por lo que del microprocesador hemos de poder enviar una señal y recibir una señal. Para ello el microprocesador KL25z hace uso de dos módulos llamados DAC y ADC, que han sido explicados en el punto 2.1.1.

Para convertir la señal de digital analógico hemos de usar el DAC. El diente de sierra que hemos de generar para producir un ancho de banda se ha estimado en 1 ms el periodo. De esta manera la frecuencia será de 1KHz. Es suficiente para el objetivo de nuestro proyecto. Primero de todo hemos de configurar el timer, será lo principal si queremos controlar todo el desarrollo del software del procesador.

Antes que todo eso hemos necesitado establecer un orden de procesos cada vez que se cumpla un ciclo: primero leemos el dato de entrada del ADC y después escribimos en la salida del DAC.

6.1.1 TIMER

Utilizamos un módulo en el microprocesador llamado pitTimer. Ese modulo nos permite establecer un periodo por el cual, mediante interrupciones, podremos controlar los ciclos que

necesitemos. El periodo establecido para cada ciclo es de 1ms, de esta manera podremos generar un diente de sierra que posteriormente genere un pulso en frecuencia.

Si hemos de generar en 1 ms un diente de sierra, hemos de tener en cuenta cuantos puntos de precisión queremos tener en nuestra señal. En nuestro caso, estimando un buen comportamiento, se ha elegido 1000 puntos de precisión para mayor seguridad. Esto significa que debemos de configurar nuestro Timer con un periodo de 1 µs. Así, en cada periodo sumaremos un valor fijo de tensión a la salida hasta llegar a 1000 µs, momento en el que reiniciaremos la cuenta. En la siguiente ilustración podemos ver una lista de configuración del timer en nuestro proyecto:

Name	Value	Details
Component name	pitTimer1	
Device	PIT	PIT
Counter	PIT_CVAL0	PIT_CVAL0
Counter type	Down counter	
Component version	1.2.0	
Configurations	Enabled	
PIT configurations	Enabled	
Configurations list	1	
Configuration 0	Enabled	
Name	pitTimer1_InitConfig0	
Type	pit_user_config_t	
Read only configuration	Enabled	
Interrupt	Enabled	
Period	1 µs	Clock cfg. 4: 1 µs
Initialization		
Auto initialization	Enabled	
Driver init. configuration	pitTimer1_InitConfig0	
Run in debug	Enabled	
Start PIT timer	yes	
Interrupts		
Interrupt	INT_PIT	INT_PIT
Interrupt priority	Enabled	
Priority value	medium priority	64
Install interrupts	Enabled	
Shared components		
fsl_clock_manager	clockMan1	
fsl_interrupt_manager	intMan1	
OS abstraction layer	Enabled	
Inherited components		

Ilustración 25 – Configuración Timer

Para configurarlo en el programa de ejecución es preciso inicializarlo primero, y después realizar la lógica de comprobaciones. Para ello utilizamos interrupciones, para comprobar si

se ha terminado de realizar la cuenta de 1 microsegundo. En el caso en el que salte la interrupción, se ejecutará las funciones de cada ciclo:

```
//Periodic Interrupt Timer (si acaba el periodo = true)
extern bool ready;

...

// INITIALISE MODULE PIT TIMER

PIT_DRV_Init(FSL_PITTIMER1,true);
PIT_DRV_StopTimer(FSL_PITTIMER1,0);

...
...


// START TIMER
PIT_DRV_StartTimer(FSL_PITTIMER1,0);

for(;;){

	if(ready){
		// READ FROM ADC
		adcReader();


		// WRITE INTO DAC (AND SUM A FIX VOLTAGE)
		dacDienteSierraAdd();

		// RESET READY TO COUNT ANOTHER MICROSECOND
		ready = false;
		PIT_DRV_StartTimer(FSL_PITTIMER1,0);
	}
	// CHECK IF IS GREATER THAN 1 MLISECOND TO RESET
	dacDienteSierraReset();

}
```

Cada vez que se genera una interrupción el microprocesador se dirige hacia el archivo events.c y ejecuta la interrupción `pitTimer1_IRQHandler(void)` para sobrescribir el valor de ready de false a true:

```
void pitTimer1_IRQHandler(void)
{
  /* Clear interrupt flag.*/
  PIT_HAL_ClearIntFlag(g_pitBase[FSL_PITTIMER1], FSL_PITTIMER1_CHANNEL);

  PIT_DRV_StopTimer(FSL_PITTIMER1,0);
  ready=true;
}
```

6.1.2 DAC

Este componente requiere una configuración más compleja. Par ello escribiremos en el DAC la señal de salida que se desee, que en nuestro caso aplica generar un diente de sierra.

Por ello, en cada ciclo hemos de sumar cierta tensión en los valores de salida. Si hemos acordado tener 1000 puntos de precisión, con 1 ms de ciclo para cada diente de sierra, para saber cuantos bits sumarle cada 1 µs hemos de conocer de cuantos bits disponemos en el registro de señal de salida del DAC [22].

Name	Value	Details
Component name	daConv1	
Device	DAC0	DAC0
Component version	1.2.0	
Configurations	Enabled	
Buffer configurations	Enabled	
Configurations list	1	
Configuration 0	Enabled	
Basic configurations	Enabled	
Configurations list	1	
Configuration 0	Enabled	
Name	daConv1_InitConfig0	
Type	dac_converter_config_t	
Read only configuration	Enabled	
Voltage reference	Reference 1	
Low power mode	Disabled	
Pins		
Output pin	Enabled	
Output pin	J10_11	DAC0_OUT/ADC0_SE23/CMP0_IN4...
Trigger pin	Disabled	
Initialization		
Auto initialization	Enabled	
Driver init. configuration	daConv1_InitConfig0	
Buffer function	Enabled	
Buffer init. configuration	daConv1_BufferInitConfig0	
Interrupts		
D/A interrupt	INT_DAC0	INT_DAC0
Configure D/A interrupt priority	Disabled	
Install interrupts	Disabled	
Shared components		
fsl_clock_manager	clockMan1	
fsl_interrupt_manager	intMan1	
OS abstraction layer	Disabled	
Inherited components		
fsl_dac_hal	KSDK 1.2.0/fsl_dac_hal	

Ilustración 26 – Configuración DAC

En la configuración podemos observar que la salida del DAC será a través de la salida J10_11. En el manual de referencia [7] observamos que el registro del DAC de salida es de 16 bits. En total podríamos obtener una salida en decimal de 2^{16}, que haría un total de 65096. Si en un principio acordamos tener una precisión de 1000 puntos, significaría que hemos de ir sumando 65 cada microsegundo. Lo primero es inicializar el DAC:

```
//Variables dac
uint16_t dacOutput=0;
int dacCount=0;

...

	//INIT DAC
	DAC_DRV_Init(FSL_DACONV1,&daConv1_InitConfig0);
	DAC_DRV_ConfigBuffer(FSL_DACONV1,&daConv1_BufferInitConfig0);
	DAC_DRV_Output(FSL_DACONV1,0);
```

En el siguiente código podemos ver las funciones comentadas en el código del Timer anterior. La función `dacDienteSierraAdd(void)` aumenta la tensión de salida en 65. La función `dacDienteSierraReset(void)` reinicia el contador si sobrepasa 1 milisegundo:

```
void dacDienteSierraAdd(void){
	dacCount++;
	dacOutput+=65;
	DAC_DRV_Output(FSL_DACONV1,dacOutput);

}

void dacDienteSierraReset(void){
	if(dacCount >= 1000){
		dacCount=0;
		dacOutput=0;

	}
```

6.1.3 ADC

Este componente va a leer la respuesta del sensor pasada por un filtro, por lo que únicamente hemos de poder calibrarla y posteriormente coger satisfactoriamente su valor. La entrada del ADC es el J1_3, la entrada por defecto de este módulo.

Su configuración es la siguiente:

Name	Value	Details
Component name	adConv1	
Device	ADC0	ADC0
Component version	1.2.0	
Configurations	Enabled	
Channel configurations	Enabled	
Configurations list	1	
Configuration 0	Enabled	
Name	adConv1_ChnConfig0	
Type	adc16_chn_config_t	
Read only configuration	Enabled	
Interrupt	Enabled	
Differential mode	Disabled	
A/D channel (pin)	VREFL	VREFL
ADC configurations	Enabled	
HW compare configurations	Enabled	
ADC PGA configurations	Disabled	
Pins		
Trigger A	Enabled	
Pin	J1_3	ADC0_SE14/TSI0_CH13/PTC0/EXT...
Trigger B	Disabled	
Pin	<Automatic>	Property is disabled
Initialization		
Auto initialization	Enabled	
Driver init. configuration	adConv1_InitConfig0	
Conversion initialization	2	
Control group 0		
Control group initialization	Enabled	
Configuration	adConv1_ChnConfig0	
Control group 1		
Control group initialization	Disabled	
Interrupts		
A/D interrupt	INT_ADC0	INT_ADC0
Configure A/D interrupt priority	Disabled	
Install interrupts	Disabled	
Shared components		
Inherited components		

Ilustración 27 – Configuración ADC

Lo primero de todo tenemos que inicializar el modulo ADC y las variables que vamos a utilizar. El código de para leer el dato del ADC lo tenemos a continuación:

```
//Variables adc
uint16_t adc_value = 0;
uint16_t adc_array[1010];
int cont_array_adc = 0;

...


void adcReader(void){
	/* Comienzo de la conversión */
	ADC16_DRV_ConfigConvChn(FSL_ADCONV1, 0U, &adConv1_ChnConfig0);
	/* Espera a fin de conversión */
	ADC16_DRV_WaitConvDone(FSL_ADCONV1,0);
	/* Lectura de la conversión */
	adc_value = ADC0_RA;
	add_to_array();
}

void add_to_array(void){
	adc_array[cont_array_adc] = adc_value; //guardo en el array el valor
	cont_array_adc++;
	if(cont_array_adc>= 1000){
		cont_arrazy_adc = 0;	//restablezco el contador del array
	}
}
```

Lo que hacemos es guardar las tensiones de salida del ADC en un array. De esta manera obtenemos la representación del espectro (en frecuencia convertido en tiempo) para poder analizarlo y utilizar esos datos, aunque no entra dentro del alcance de este proyecto.

6.2 DISEÑO PLACA PCB

El diseño de la placa de procesamiento de la señal de nuestro proyecto se ha realizado dividiéndola en tres partes. La primera consiste en diseñar un esquemático del circuito a implementar. La segunda en diseñar componente a componente la huella (diseño físico en la placa PCB a implementar). La tercera y última parte se refiere al diseño del ensamblaje de todos los componentes, atendiendo a factores de tamaños de líneas de interconexión y de la colocación en la placa.

6.2.1 ESQUEMA DEL CIRCUITO

El esquema diseñado lo podemos ver en la Ilustración 28. A continuación, hablaremos de cada uno de los distintos componentes en el esquema, explicando que papel juega cada uno en el circuito. No todos los esquemas de los componentes existen, como el VCO y el detector de envolvente. Estos esquemas hemos de generarlos nosotros a partir del datasheet. Además, en este punto solo se detallan entradas y salidas, no las medidas reales de los componentes.

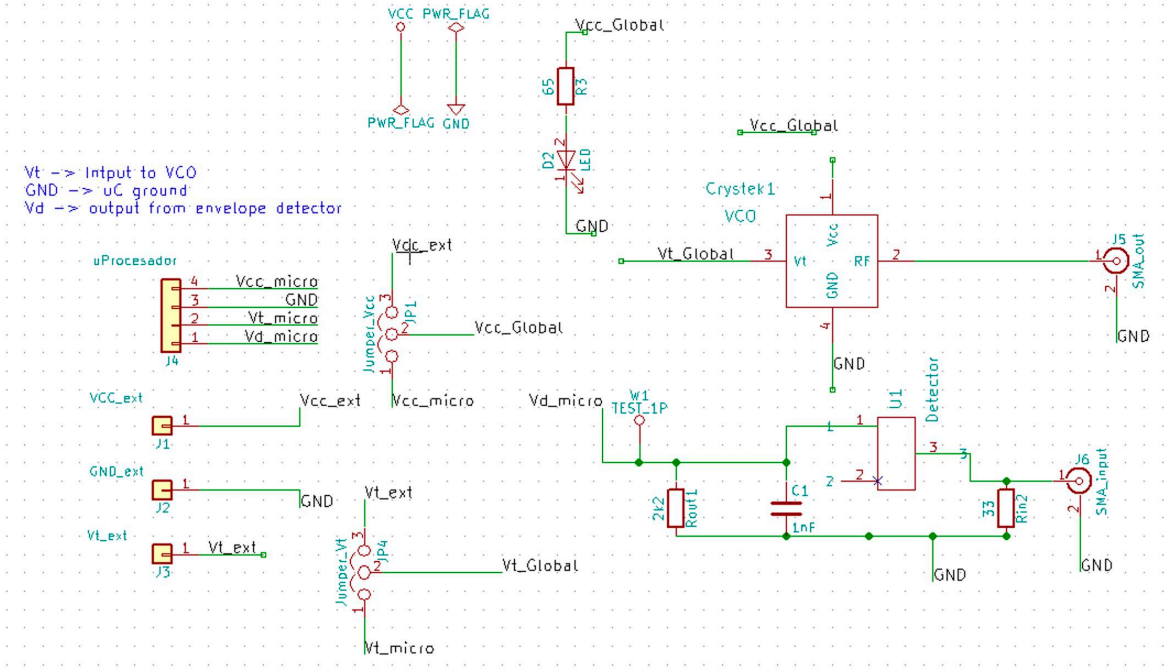


Ilustración 28 – Esquema circuito PCB

6.2.1.1 Entradas y salidas del microprocesador en la placa PCB

En nuestro circuito hemos de poder introducir las salidas del microprocesador, tales como la tensión de referencia Vcc de 3'3 voltios, la toma de tierra, la salida del DAC y la entrada al ADC. Podemos observar estas entradas en el siguiente recorte del esquema anterior:

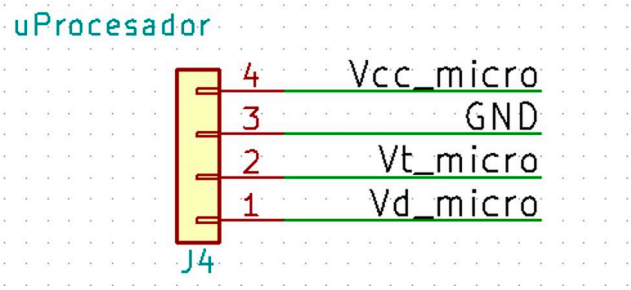


Ilustración 29 – Entradas microprocesador

6.2.1.2 Entradas auxiliares en la placa PCB

Aparte de las conexiones que tenemos entre el microprocesador y la placa PCB, que son las principales, tenemos tres entradas auxiliares más, que servirán para realizar pruebas de funcionamiento sin necesidad de utilizar el microprocesador. Estas tres entradas son la tensión de referencia Vcc auxiliar, una salida Vt auxiliar que servirá de entrada al VCO y una toma de tierra auxiliar. Podemos observar este esquema en la siguiente ilustración:

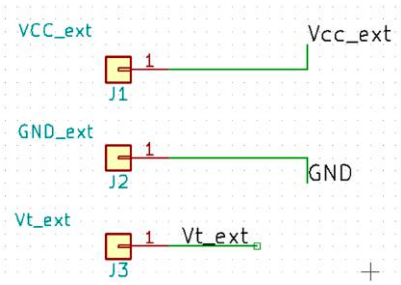


Ilustración 30 – Entradas auxiliares

Para poder elegir entre utilizar la entrada auxiliar o las entradas del microprocesador utilizamos Jumpers. Los Jumpers son componentes que interconectan dos pines machos que se sitúan cerca en una misma hilera de pines. Los Jumpers utilizados son solo para elegir de donde proviene la fuente de alimentación y de donde la tensión de entrada del VCO.

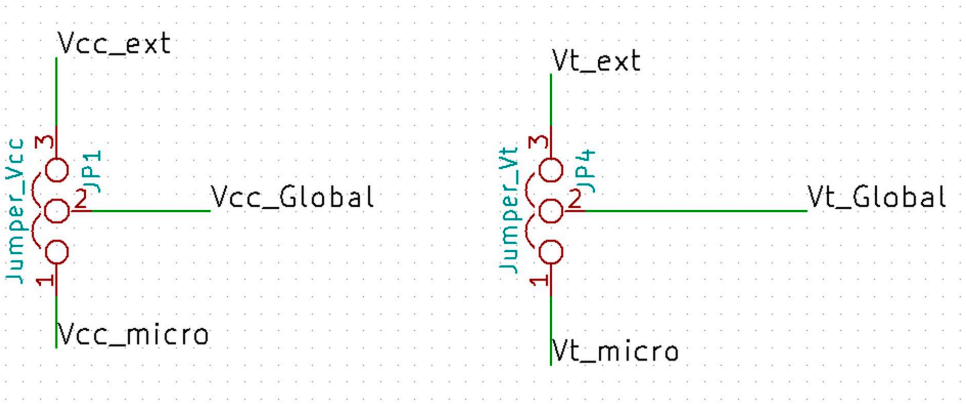


Ilustración 31 – Jumpers para escoger fuente de alimentación y entrada al VCO

6.2.1.3 VCO

El esquemático de este componente se ha generado utilizando las medidas facilitadas por el datasheet [21]. Se nos detalla las entradas y salidas del VCO, punto del que ya se ha hablado en el apartado 5.1.2. Es por eso por lo que tenemos 3 entradas (Vt, Vcc, GND) y 1 salida (RF):

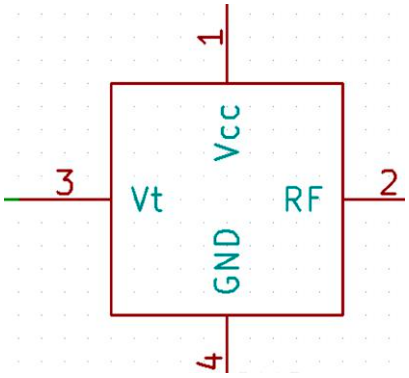


Ilustración 32 – Esquemático componente VCO

6.2.1.4 Entrada y salida del sensor en la placa PCB

La señal se procesará por el VCO generando un pulso en frecuencia. Para mandarle la señal al sensor utilizaremos conectores SMA para radiofrecuencia, ya que es un medio óptimo para enviar y proteger la señal. Los conectores tienen una entrada para la señal y una entrada para la tierra. Tendremos una salida hacia el sensor y una entrada para la respuesta del sensor:

Ilustración 33 – Entrada y salida al sensor

6.2.1.5 Detector de envolvente

Para realizar este esquema, hemos de utilizar un diodo específico mencionado en el punto 5.1.3. Utilizando el datasheet del HSMS-2850 generamos el esquema [22].

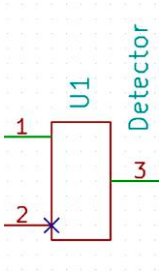


Ilustración 34 – Esquema diodo HSMS-2850

Como podemos observar en el datasheet, el HSMS-2850 tiene dos diodos. De tal manera que dependiendo de lo que necesitemos en nuestro proyecto se pueden utilizar los dos o solo uno de los dos. Lo podemos observar en la siguiente ilustración.

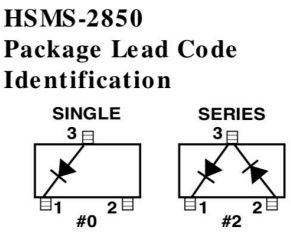


Ilustración 35 – Formas de utilizar el HSMS-2850

Es por ello por lo que únicamente para el detector de envolvente requerido en nuestro proyecto hemos de hacer uso de un solo diodo. En nuestro esquema hemos tenido que indicar que el pin 2 no esta interconectado con nada. Por lo que solo utilizamos el diodo que va desde el pin 3 al pin 1.

El esquema en el que nos hemos basado para implementar el diodo junto al filtro lo encontramos en el punto 5.1.3. El procesamiento que hemos de realizar para transformar la envolvente de frecuencia a tiempo desde que se recibe la respuesta del sensor hasta la recepción en el microprocesador, es el siguiente:

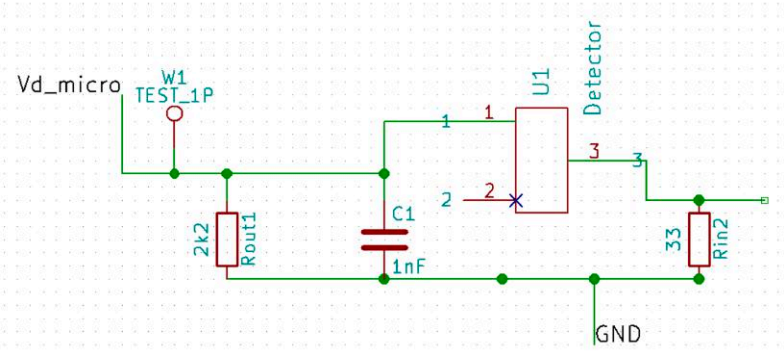
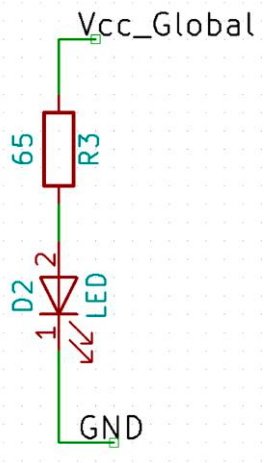


Ilustración 36 – Esquema filtro y detector de envolvente

Podemos observar cómo se le ha añadido a la salida del detector una salida para poder realizar comprobaciones y poder tomar la señal de una manera sencilla. Posteriormente cuando diseñemos físicamente la placa PCB se podrá observar con mayor facilidad su funcionalidad.

6.2.1.6 Componentes adicionales

El circuito anterior es el circuito principal del proyecto. Sin embargo, se ha añadido un circuito adicional de comprobación de funcionamiento de la placa PCB, que incluye un led junto a una resistencia para ver que la placa PCB esta correctamente alimentada. Lo podemos observar en el siguiente esquema:



6.2.2 HUELLAS COMPONENTES

Una vez comprobado que el esquema está correcto, es decir, que no se encuentra ningún componente sin conectar y que el flujo es el que debe ser, pasamos a diseñar particularmente cada una de las huellas de los componentes.

Las huellas de un componente se generan teniendo en cuenta las medidas originales y reales del componente a instalar en la placa PCB. Hemos de ser cuidadosos en esta parte, debido a que si no introducimos las medidas correctas es posible que posteriormente no podamos soldarlos. Muchos componentes ya poseen huellas predefinidas, como pueden ser algunos pines machos. Aun así, detallaremos cada una de las huellas.

6.2.2.1 Entradas y salidas del microprocesador en la placa PCB

Los pines conectados directamente con el microprocesador serán pines machos. La huella a la que posteriormente se le soldará los pines tendrá 1mm de diámetro, de la siguiente manera:

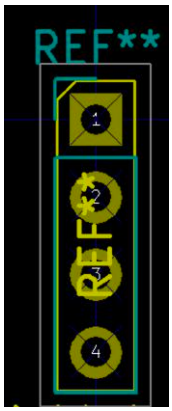


Ilustración 37 - Huella pines macho

6.2.2.2 Entradas auxiliares en la placa PCB

La huella de las tres entradas auxiliares es la misma. Debido a que se va a utilizar fuentes de tensión, tierra y entrada de una señal para el VCO, es mas sencillo usar una conector banana de 2mm de radio. La huella de las tres entradas es la siguiente:

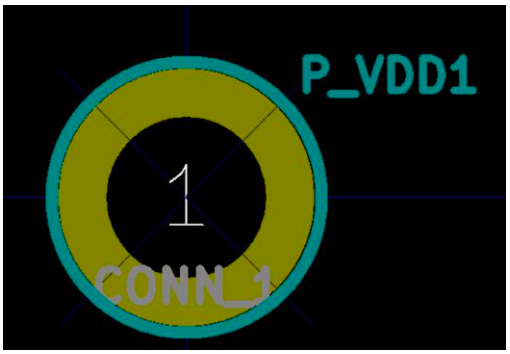


Ilustración 38 – Huella entrada auxiliar banana 2mm

La huella de los Jumpers para elegir si utilizamos la entrada auxiliar o la entrada del microprocesador es la misma. Los Jumpers tendrán las mismas medidas que los pines de entrada del microprocesador, 1mm de diámetro. La huella será la siguiente:

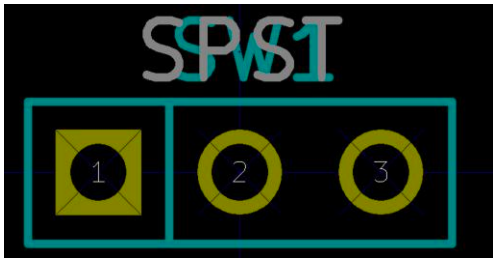


Ilustración 39 – Huella Jumpers

6.2.2.3 VCO

Para la realización de esta huella nos hemos basado en la información proporcionada por el datasheet [21]. La toma de tierra está conectada con la carcasa del VCO. Esto es debido a que cuanta más superficie de masa exista, mejor conducirá el calor, lo cual se traduce en un mejor funcionamiento del componente. Debido a que es nuestro objetivo a la hora de diseñar esta huella, colocamos las medidas facilitadas por el fabricante en el editor de huellas. Podemos observar las medidas del VCO en la siguiente ilustración:

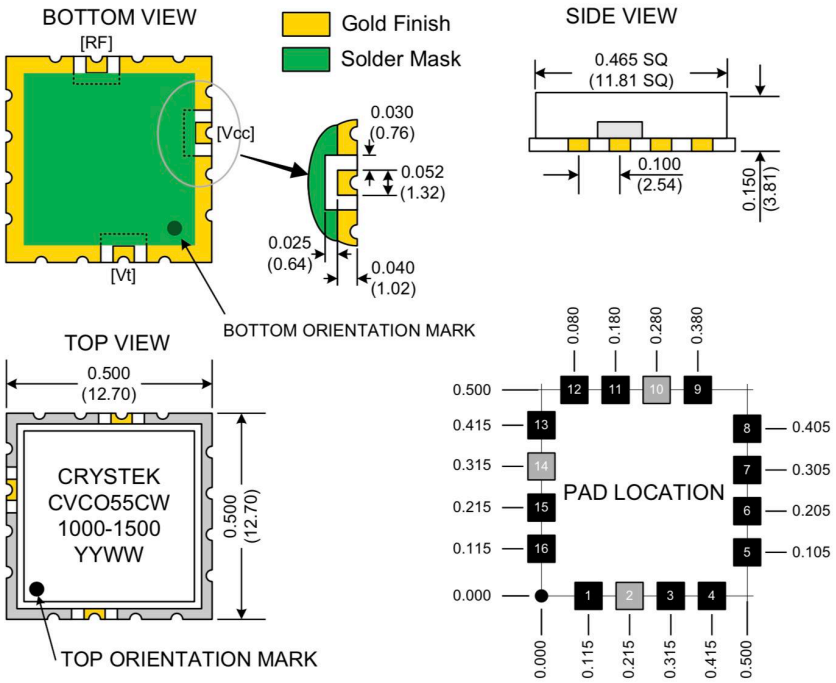


Ilustración 40 – Medidas VCO obtenidas del datasheet

Debido a que necesitamos tener una entrada de Tierra añadimos un cuarto pin, al que conectaremos a tierra como hemos indicado en el apartado 6.2.1.3. Este pin tendrá las medidas más grandes posibles, de tal manera que no interfiera con los otros pines tratando de dejar los márgenes pertinentes. La huella resultante es la siguiente:

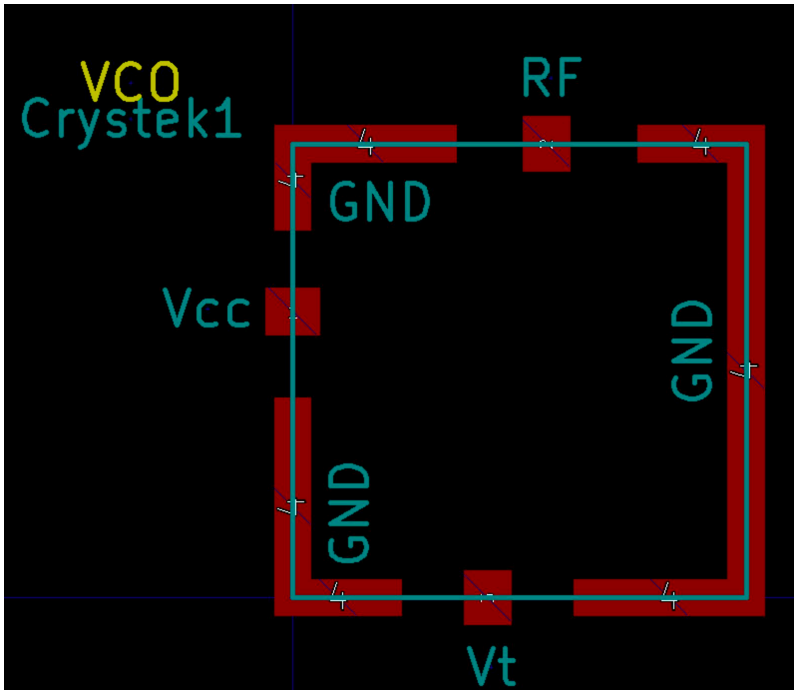


Ilustración 41 – Huella VCO

Los pines del componente en rojo indican que se tratan de conectores SMD. Esto significa que no traspasan la placa PCB, sino que se sueldan a la capa más cercana a la superficie.

6.2.2.4 Entrada y salida del sensor en la placa PCB

Estos conectores son para cable coaxial, y se llaman SMA. Se conforman de dos pines, uno conecta con la a tierra y otro es el que lleva la señal. Pueden ser fabricados de manera muy diferente, pero cuanta más superficie esté conectada a tierra mejor conducirá el calor.

Ilustración 42 – Conector SMA para radiofrecuencia

De tal manera que la huella utilizada para poder fabricar este conector será la siguiente:

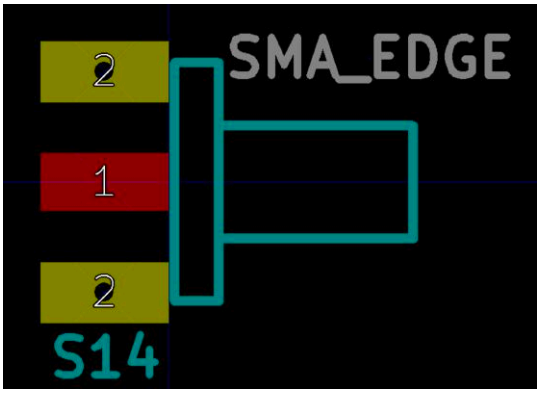


Ilustración 43 – Huella conector SMA

Podemos observar como los dos pines de los extremos son de tipo pasante, de tal manera que atravesarán toda la placa PCB. En cambio, el central, el pin 1, será de tipo SMD. Esta huella se situará en el extremo de la placa PCB, puesto que los rectángulos verdes representados en la Ilustración 43 será donde se coloque el conector SMA.

6.2.2.5 Detector de envolvente

Para generar el filtro y el detector de envolvente hemos de diseñar únicamente la huella del diodo utilizado (HSMS-2850). El resto de las huellas de los componentes, las dos resistencias y el condensador, se nos facilitan por la herramienta:

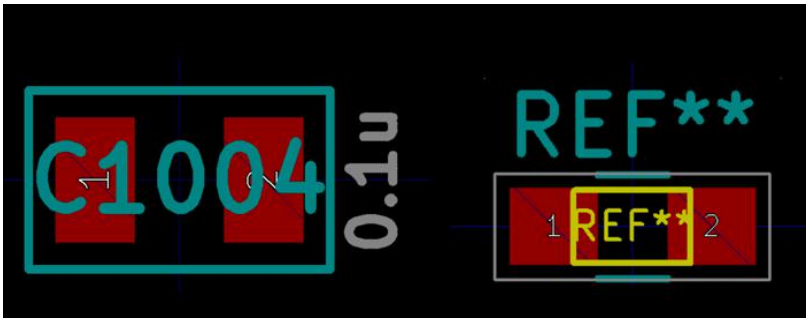


Ilustración 44 – Huella del condensador y las resistencias, de izquierda a derecha.

Para diseñar la huella del diodo HSMS-2850 se ha tenido en cuenta el datasheet facilitado por el fabricante [22]. A continuación, mostramos las medidas que hemos utilizado:

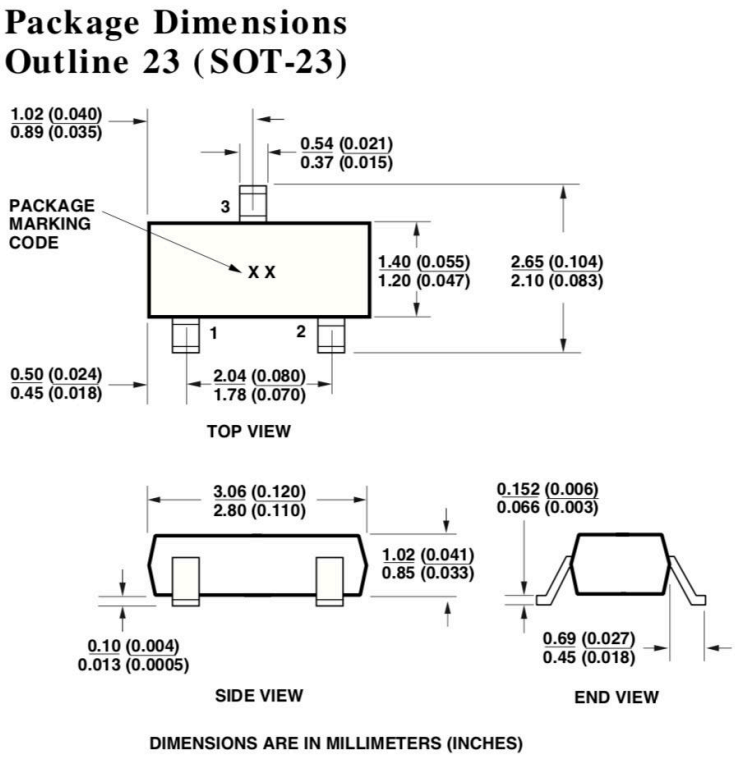


Ilustración 45 – Dimensiones diodo HSMS-2850

Como se puede observar tiene tres pines. Solo vamos a utilizar el pin 3 y el 1, dejando el pin 2 sin utilizar. Aun así, hemos de diseñar la huella con los tres pines. Podemos observar el resultado en la siguiente imagen:

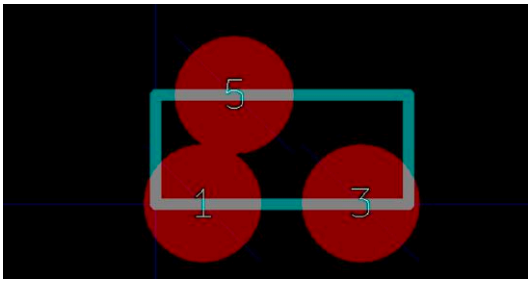


Ilustración 46 – Huella diodo HSMS-2850

6.2.2.6 Componentes adicionales

Las huellas de la resistencia y del diodo utilizado para saber si la placa está correctamente alimentada son los siguientes:

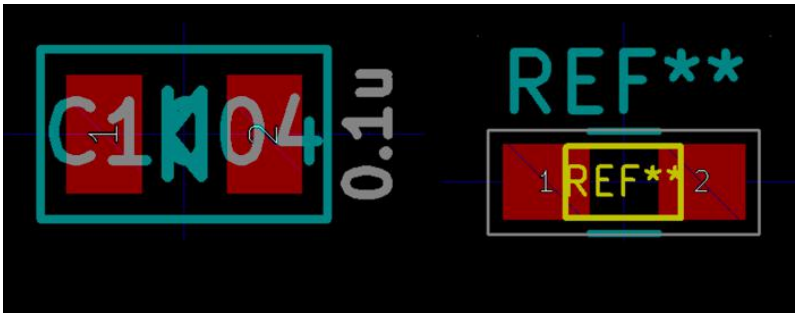


Ilustración 47 – Huella diodo y resistencia, de izquierda a derecha.

6.2.3 ENSAMBLAJE DEL DISEÑO

Una vez diseñadas las huellas de los componentes hemos de diseñar la posición en la placa de cada componente. Además, será necesario tener en cuenta el camino de las líneas de interconexión entre las huellas. Esto es debido a que al diseñar una placa PCB, podemos generar líneas de conexión en dos capas distintas llamadas Front-end y Back-end. De esta manera podemos cruzar líneas para que no interfieran entre ellas y facilitar así la conexión entre los componentes.

El tamaño de las líneas de transmisión vendrá determinado por tres factores:

- Si se trata de una línea de transmisión normal.
- Si se necesita de un ancho mínimo debido a la longitud de onda ya que trabajamos con frecuencias. Si la longitud de onda no entra en el ancho de banda perdemos señal por lo que es importante tener en cuenta qué conexiones son críticas.
- Si se trata de una línea de transmisión que proviene de una fuente de alimentación. En este caso necesitaremos tener un ancho de transmisión grande, para disipar calor y que funcione de una forma óptima.

	Margen	Ancho de pista	Diam vía	Tdro vía	Diam microvía	Tdro microvía
Default	0,007874015748	0,009842519685	0,02362204724	0,0157480315	0,01181102362	0,003937007874
Filtro_entrada	0,007874015748	0,09	0,02	0,015	0,0118	0,004
Power think	0,005905511811	0,05	0,05	0,025	0,02	0,005
RF	0,007874015748	0,1141732283	0,02362204724	0,0157480315	0,01181102362	0,003937007874

Tabla 3 – Tamaño de líneas de transmisión

A la izquierda de la tabla figura el nombre dado para cada línea de transmisión. Los valores importantes de esa tabla son el ancho de pista, el diámetro de la vía, y el margen dado para que no se crucen con la tierra o con otra vía.

Tras analizar nuestro diseño nos damos cuenta de que las vías que utilizarán el ancho de vía por defecto, que es el llamado 'Filtro_entrada', serán las que interconecten las salidas del microprocesador con las entradas de la placa PCB. Solo las entradas, porque la salida de la placa PCB al microprocesador necesitará un ancho de banda mínimo para que no se pierda información debido a la radiofrecuencia. Las entradas auxiliares Vt, Vcc y Tierra tendrán un ancho de banda grande para poder canalizar bien toda la tensión, por lo que la vía será la llamada 'Power think'. Finalmente, desde la salida del VCO hasta la entrada al microprocesador, el ancho de banda será necesario que cumpla con los tamaños de la configuración 'RF' de la Tabla 3. En la siguiente tabla podemos observar las configuraciones para los distintos enlaces o vías:

/Vd_micro	Filtro_entrada
Net-(J6-Pad1)	Filtro_entrada
/Vcc_Global	Power think
/Vcc_ext	Power think
/Vcc_micro	Power think
/Vt_Global	Power think
/Vt_ext	Power think
/Vt_micro	Power think
Net-(D2-Pad2)	Power think
Net-(Crystek1-Pad2)	RF

Tabla 4 – Configuraciones de cada vía

En el diseño de la placa PCB es necesario aclarar para qué se va a usar cada capa. La capa front-end se dejará principalmente para las conexiones entre las huellas de los componentes. Se utilizará la capa del back-end solo cuando haya que cruzar enlaces y sea necesario pasar por debajo de una vía. La capa de Back-end se deja principalmente para tensiones, como Vcc o la Tierra. En el framework Kicad se pinta de rojo los enlaces y conexiones de la capa superior (Front-end), mientras que la capa inferior utiliza el color verde (Back-end).

En la Ilustración 48 se observa el resultado de la ensamblación de las distintas huellas de los componentes. Si nos fijamos se puede observar que el tamaño de los enlaces depende de lo anteriormente mencionado. La salida RF y la entrada de la respuesta del sensor necesitan un ancho de banda mayor, pero es superior a las tensiones de referencia Vcc y la entrada Vt.

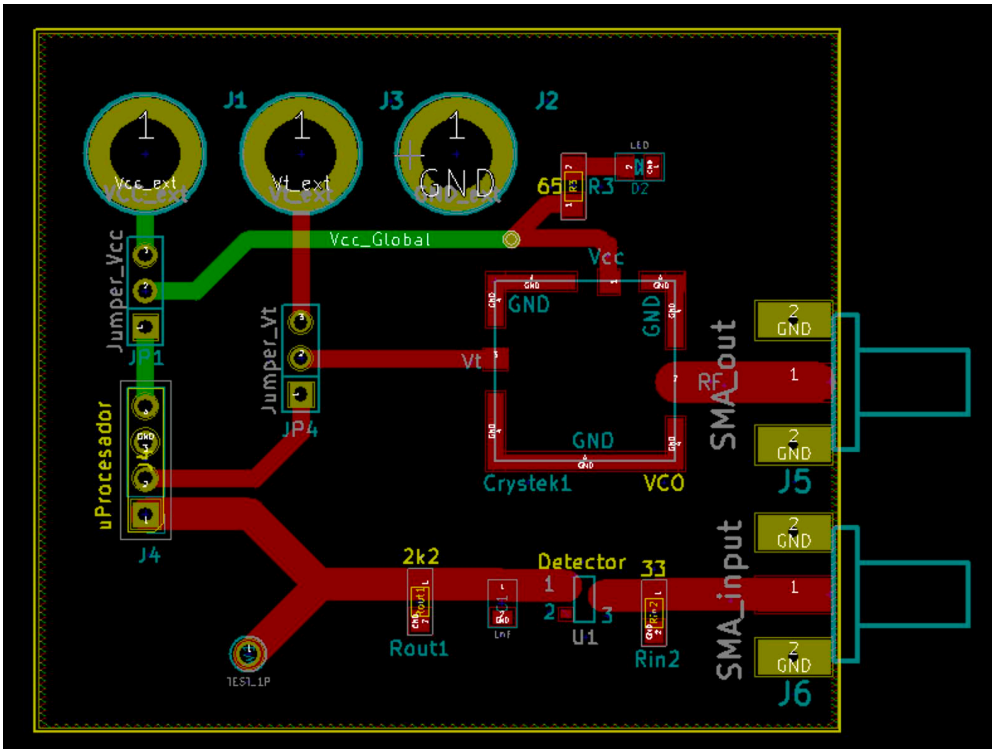


Ilustración 48 – Diseño PCB Final, todas las capas

En la Ilustración 49 se puede observar el diseño de la capa de Front-end, y que para cruzar por el enlace de Vt auxiliar necesitamos utilizar otra capa. Además, el resto del espacio de la placa no se desaprovecha, y como vemos en la Ilustración 50, lo rellenamos de masa para disipar mejor el calor.

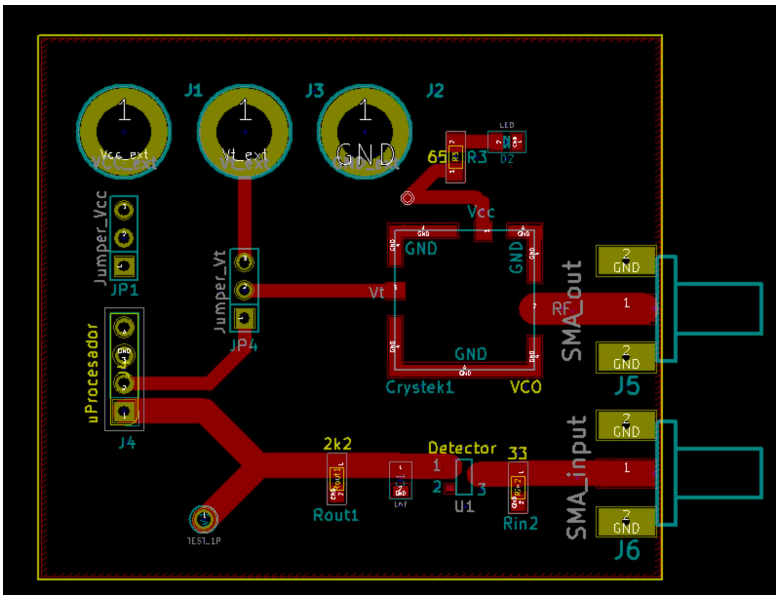


Ilustración 49 – Capa Front-end sin Relleno

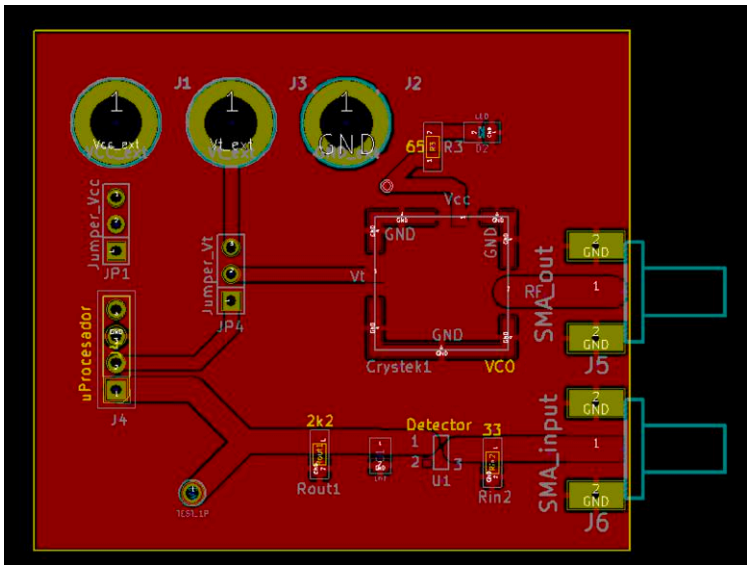


Ilustración 50 – Capa Front-end con relleno de Tierra

En la Ilustración 51 observamos que las vías utilizadas en el back-end se trataban de transmisión de tensiones, y para cruzar otro enlace. Además, también rellenamos el espacio sobrante con masa a Tierra para disipar mejor el calor.

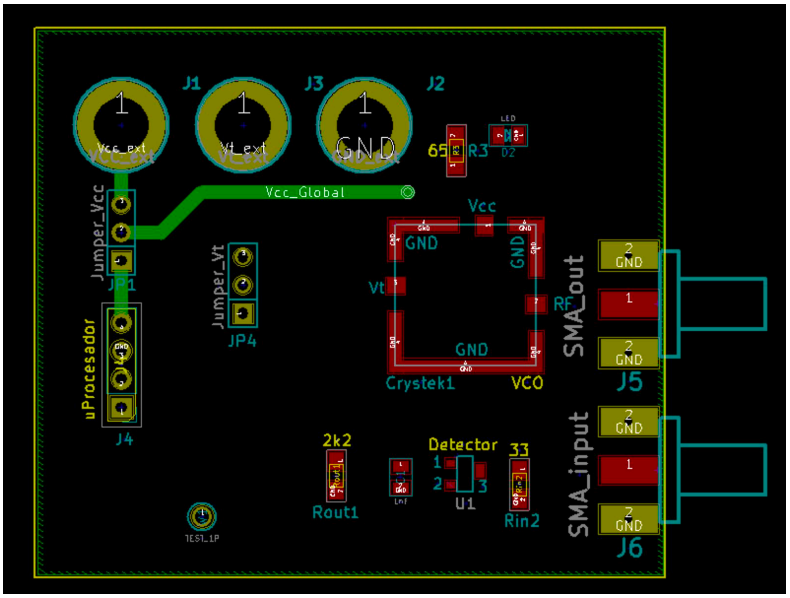


Ilustración 51 – Capa Back-end sin relleno

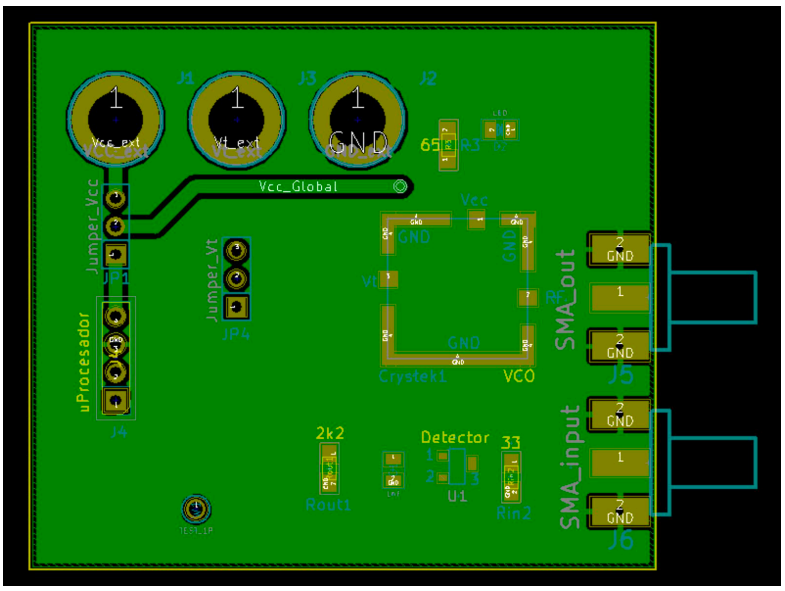


Ilustración 52 – Capa Back-end con relleno de Tierra

6.3 IMPLEMENTACIÓN

El resultado del diseño del punto 6.2.3 se fabricó por una empresa. La placa PCB se puede observar en la Ilustración 53. En la capa superior se puede percibir claramente la capa de Front-end definida en el anterior punto.

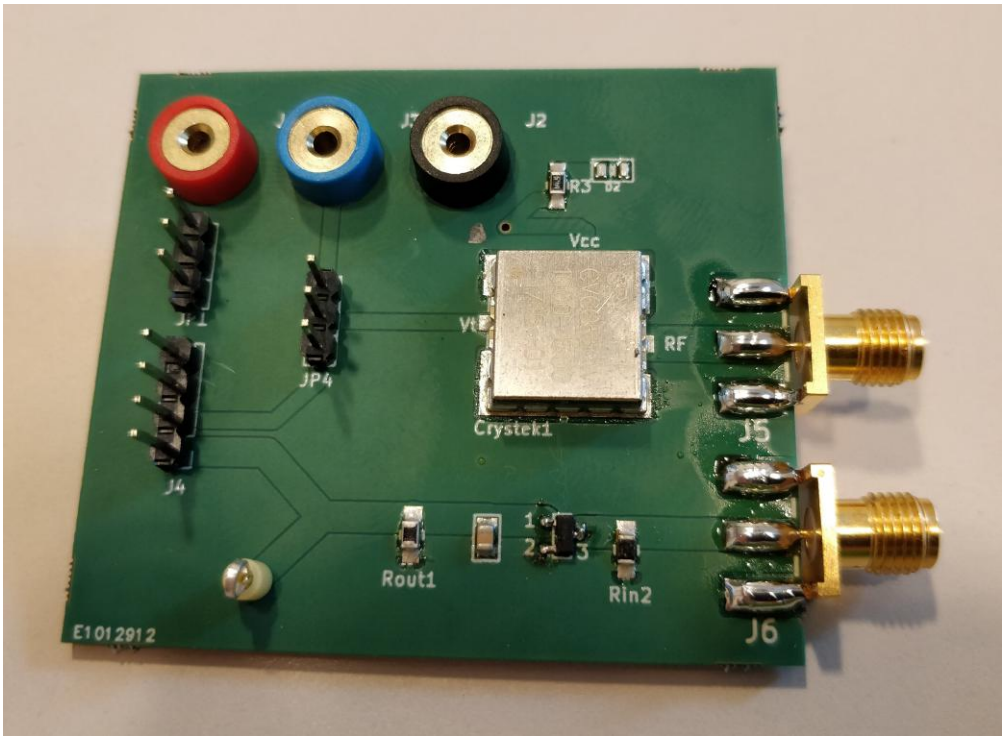


Ilustración 53 – Placa PCB Final Fabricada

En la siguiente ilustración podemos observar las dos configuraciones en función de cómo coloquemos los Jumpers. Una de ellas es para usarla con el microprocesador y otra para usarla con las entradas auxiliares.

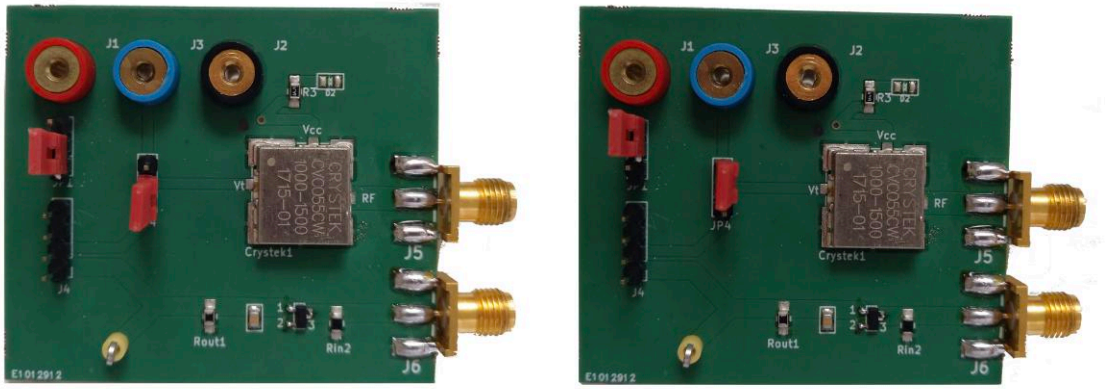


Ilustración 54 – Configuración para funcionar con el microprocesador y para funcionar con entradas auxiliares, de izquierda a derecha respectivamente.

Ahora procedemos a realizar pruebas en las distintas salidas de cada etapa y valorar su funcionamiento. Empezaremos por la salida del microprocesador, que genera según lo desarrollado un diente de sierra cada 1 milisegundo. Posteriormente analizaremos la salida del VCO, la respuesta del sensor y el detector de envolvente.

6.3.1 SALIDA DEL DAC DEL MICROPROCESADOR

Como hemos explicado en el punto 6.1.1 el microprocesador debería generar un diente de sierra, por lo que conectamos un osciloscopio a la salida para comprobar que efectivamente es lo que está sucediendo Lo podemos observar en la siguiente ilustración:

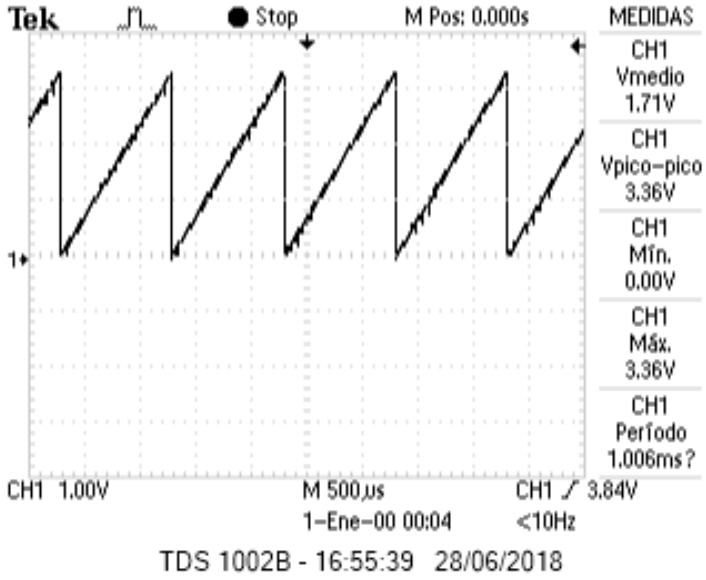


Ilustración 55 – Salida DAC microprocesador

Se puede observar que la tensión media de nuestra señal es de 3’36 Voltios. Esto significa que es la tensión máxima que el DAC puede ofrecer. Aquí tendremos un problema, puesto que el VCO necesitará llegar a más tensión de entrada para poder generar un pulso con un ancho de banda suficiente. Basicamente, si la tensión de entrada al VCO no llega a 4’2 voltios aproximadamente, el VCO no podrá generar el ancho de banda suficiente como para observar el notch del dieléctrico del aire.

6.3.2 SALIDA DEL VCO

La respuesta que el VCO ha de dar es un pulso en frecuencia de ancho de banda aproximadamente 700 MHz, en el que la frecuencia mínima sea de aproximadamente 800 MHz y la máxima sea de 1500 MHz. Para analizar la salida hacemos uso del analizador de espectros. A continuación, mostraremos la respuesta del VCO para este diente de sierra de media 3’36 Voltios.

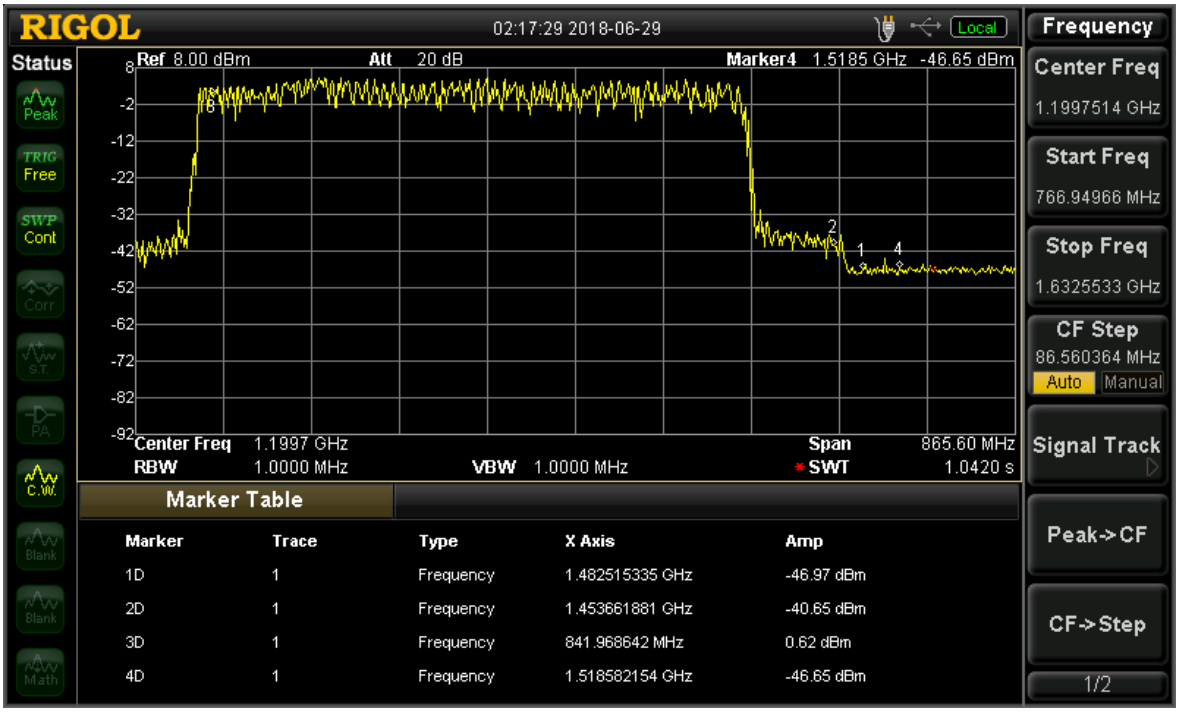


Ilustración 56 – Analizador de espectros de la respuesta del VCO a la salida del DAC del microprocesador real

Se puede observar que la salida máxima que el DAC puede ofrecer no es suficiente, ya que no genera el pulso esperado. La frecuencia mínima del pulso generado sí que es de 841 MHz, pero la máxima es de 1.372 MHz. Esto no es suficiente para ver el notch del dieléctrico del aire. Lo podemos observar en la Ilustración 57. Si por ejemplo cambiamos de dieléctrico a alcohol sucede lo mismo, no podemos observarlo, como ocurre en la Ilustración 58.

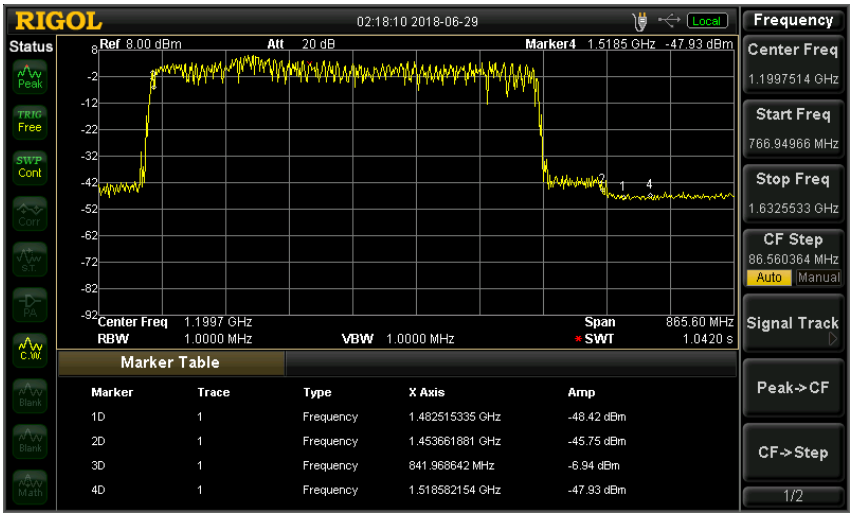


Ilustración 57 – Analizador de espectros de la salida del sensor, siendo el aire el dieléctrico utilizado.

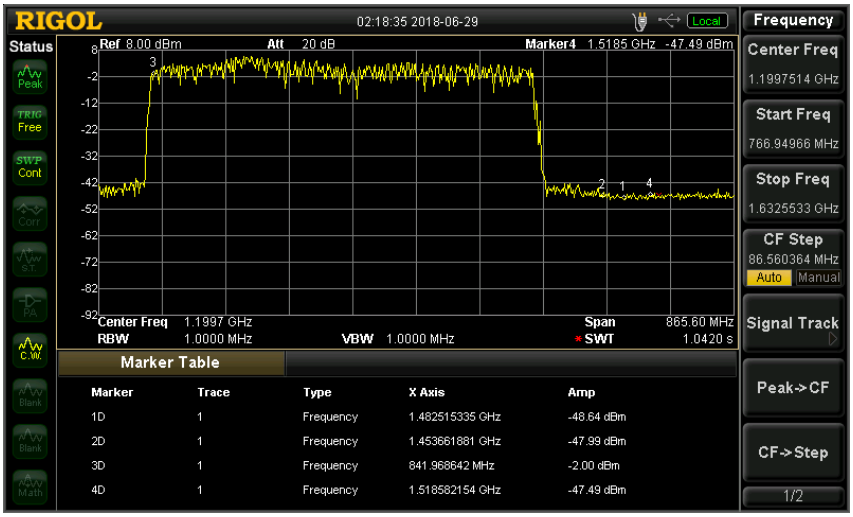


Ilustración 58 – Analizador de espectros de la salida del sensor, siendo alcohol el dieléctrico utilizado.

6.4 SOLUCIÓN AL SISTEMA PLANTEADO

El problema con el que nos hemos encontrado es fácil de solucionar. Solo hemos de aumentar la tensión de salida del DAC antes de introducirla en el VCO. Es por ello por lo que hemos de implementar un amplificador realimentado negativamente para que en vez de tener una salida de 3'36 Voltios, tengamos una salida como mínimo de 4'3 voltios.

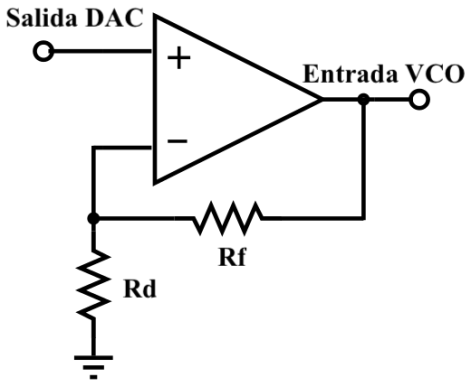


Ilustración 59 – Esquema amplificador realimentado negativamente

El amplificador escogido es el LM741 [24]. Este amplificador tiene una entrada de tensión de referencia de Vcc de +5 Voltios, una tensión Vee de -5 Voltios, la entrada de la señal principal (el DAC) y la salida (la entrada al VCO). Para obtener los valores de la resistencia hemos de analizar primero cuánto queremos amplificar.

Si queremos obtener una tensión aproximada de 4,5 voltios teniendo una tensión de 3,3 voltios de entrada, la amplificación será de 1,36 Voltios aproximadamente. Esto significa que nuestro amplificador ha de tener las resistencias necesarias para amplificar la señal con esa ganancia.

La ganancia de un amplificador realimentado negativamente es $\frac{Rf+Rd}{Rd}$. Esta relación ha de ser aproximadamente 1,36. Los valores escogidos han de ser altos del rango de algunos KΩ. Por ello los valores que coinciden aproximadamente y con valores reales de resistencias fabricadas son:

- Rf: 10 KΩ
- Rd: 29,4 KΩ

La relación entre ambas resistencias es de 1,340136054.

Este amplificador se hubiera podido añadir al sistema diseñado con anterioridad, de tal manera que hubiéramos añadido las huellas de los amplificadores para no tener que fabricar otro dispositivo. El esquema resultante del amplificador desarrollado es el siguiente:

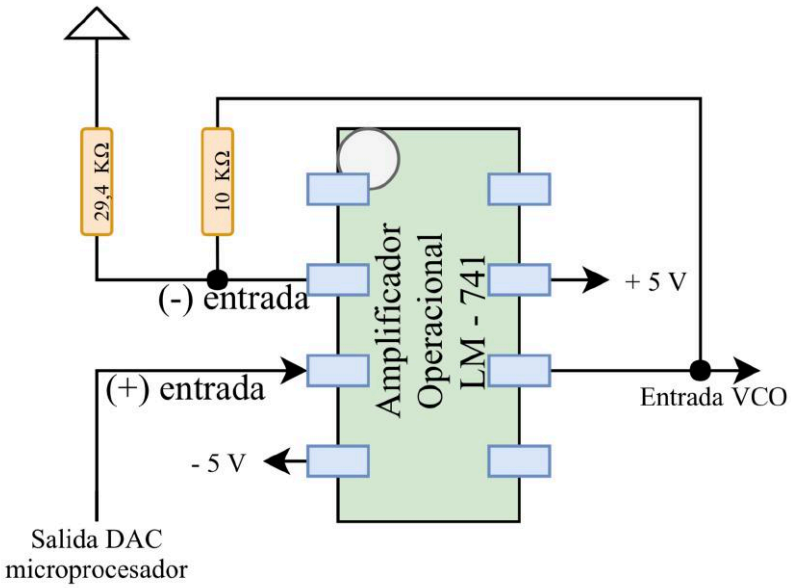


Ilustración 60 – Esquema OP-Amp LM741 con una ganancia de 1,34 Voltios

Una vez probado el esquema en el laboratorio el resultado es el siguiente:

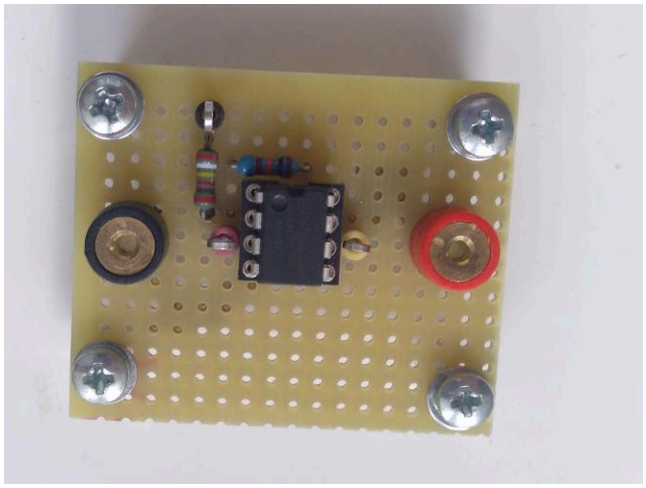


Ilustración 61 – Amplificador de ganancia 1,34 Voltios

Capítulo 7. ANÁLISIS DE RESULTADOS

Tras diseñar y desarrollar todo el proyecto vamos a proceder a analizar los resultados de todo el sistema, atendiendo a las salidas de cada etapa. Vamos a tener en cuenta la amplificación, puesto que es necesaria para que nuestro sistema funcione. Es por ello por lo que vamos a separar este capítulo en los resultados obtenidos reales y resultados obtenidos ideales. Los resultados reales son aquellos obtenidos con el funcionamiento diseñado con nuestro proyecto, mientras que los ideales no. Los resultados ideales se han generado a partir de señales no provenientes de nuestro esquema, como por ejemplo la señal diente de sierra, que será producida por un generador de señales. Subdividiremos cada uno de los apartados en el análisis de las salidas de las diferentes etapas de nuestro proyecto.

7.1 RESULTADOS REALES

7.1.1 SALIDA MICROPROCESADOR AMPLIFICADO

Hemos desarrollado nuestro sistema para que a la salida del DAC del microprocesador, pasando por el amplificador, tengamos una tensión pico-pico de 4,3 voltios aproximadamente. En la Ilustración 62 observamos la respuesta del amplificador. Podemos observar que la salida del amplificador genera un diente de sierra amplificado de 4,24 voltios de tensión pico-pico. Por lo que ahora sí que tenemos una ganancia suficiente como para poder observar el dieléctrico del aire en el sensor.

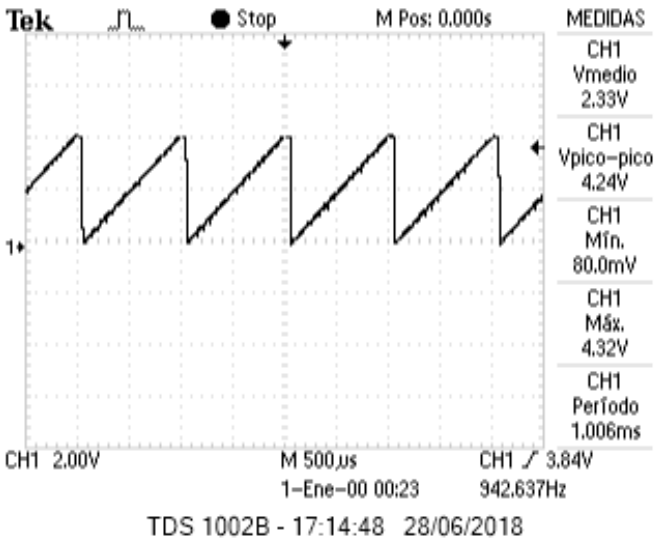


Ilustración 62 – Salida DAC del microprocesador tras pasar por el amplificador en tiempo

7.1.2 SALIDA VCO

En la Ilustración 63 observamos la respuesta en frecuencia de la salida del VCO. Podemos notar que exactamente no es un pulso, pero si aumentamos el muestreo de la señal podemos llegar a percibir el pulso. Esto lo conseguimos aumentando el parámetro sweep en el analizador de espectros. Podemos observar este fenómeno en la Ilustración 64.

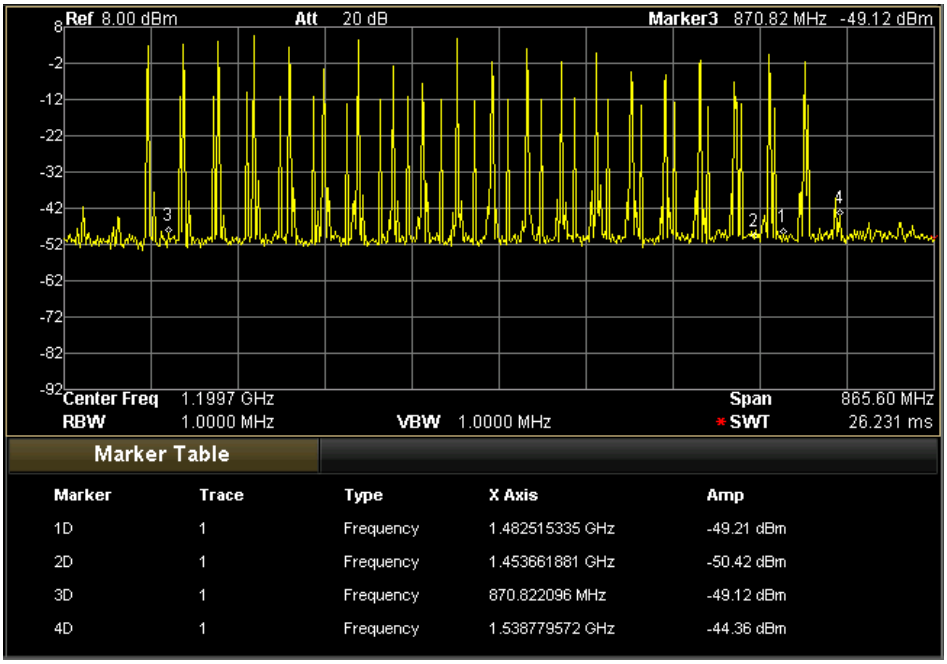


Ilustración 63 – Salida VCO real con sweep de 26 milisegundos en frecuencia

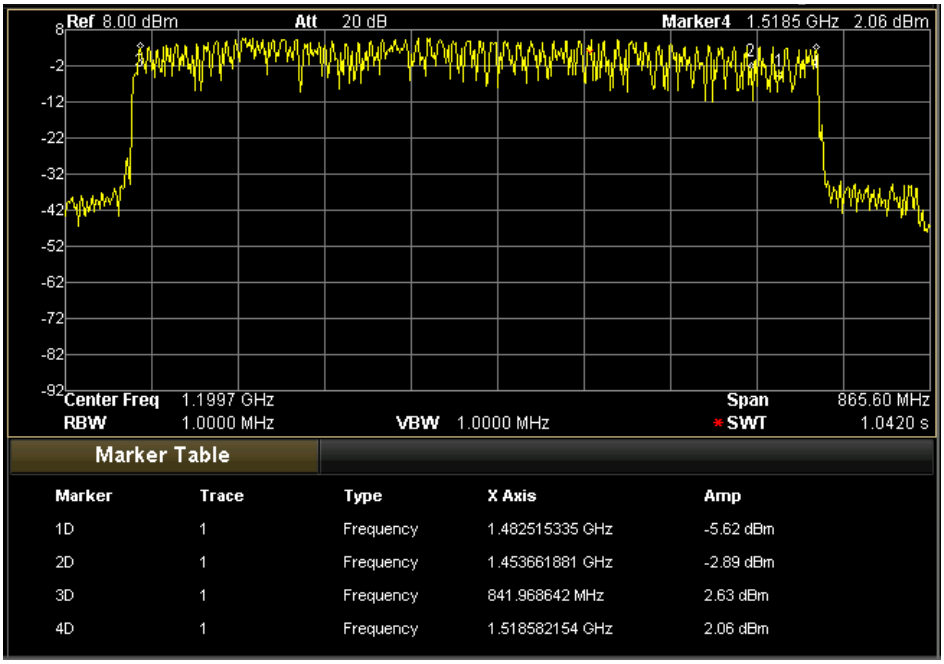


Ilustración 64 – Salida VCO real con sweep de 1 segundo en frecuencia

La diferencia entre la salida de la Ilustración 64 y la salida de la Ilustración 56 es que ahora sí que comprende las frecuencias necesarias para que el sensor perciba el dieléctrico del aire. Vamos a poder comprobar esto en el siguiente punto.

7.1.3 SALIDA SENSOR CUANDO EL DIELÉCTRICO ES EL AIRE

Si conectamos la salida de la placa PCB desarrollada con la entrada del sensor y miramos la salida del sensor en un analizador de espectros, podemos observar si realmente funciona nuestro diseño. Si conectamos el sensor y no acercamos ningún tipo de material que cambie el dieléctrico, podemos asegurar que el dieléctrico que estamos observando es el aire.

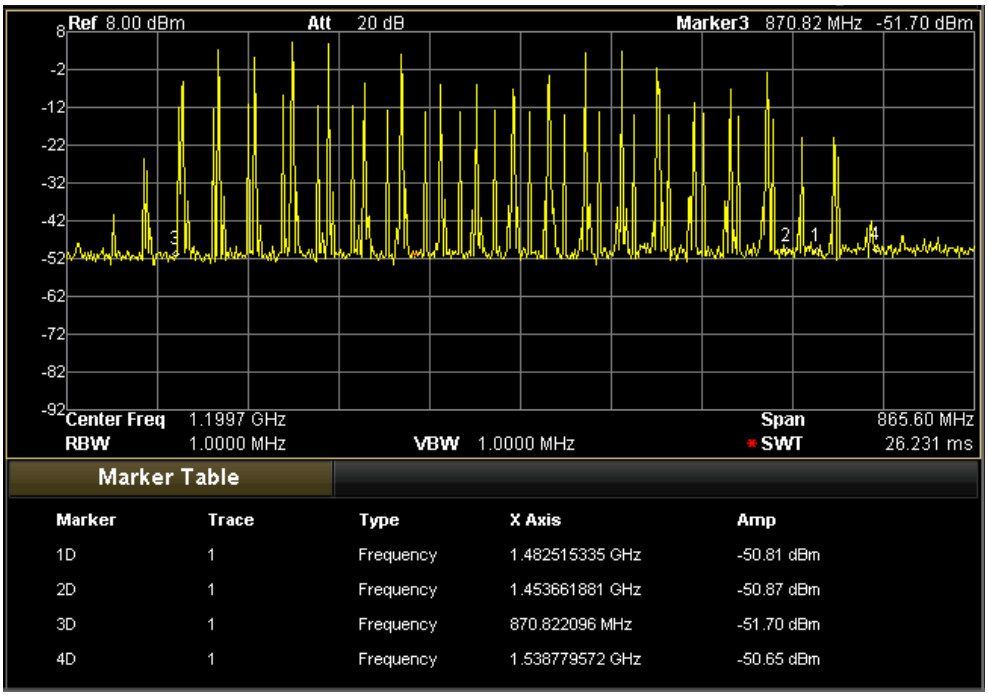


Ilustración 65 – Salida Sensor real cuando el dieléctrico es el aire con sweep de 26 milisegundos en frecuencia

Si no aumentamos el sweep no se percibe gran cambio en la respuesta. En la siguiente ilustración sí que se puede observar el notch que realiza el sensor a una frecuencia determinada cuando el dieléctrico es el aire.

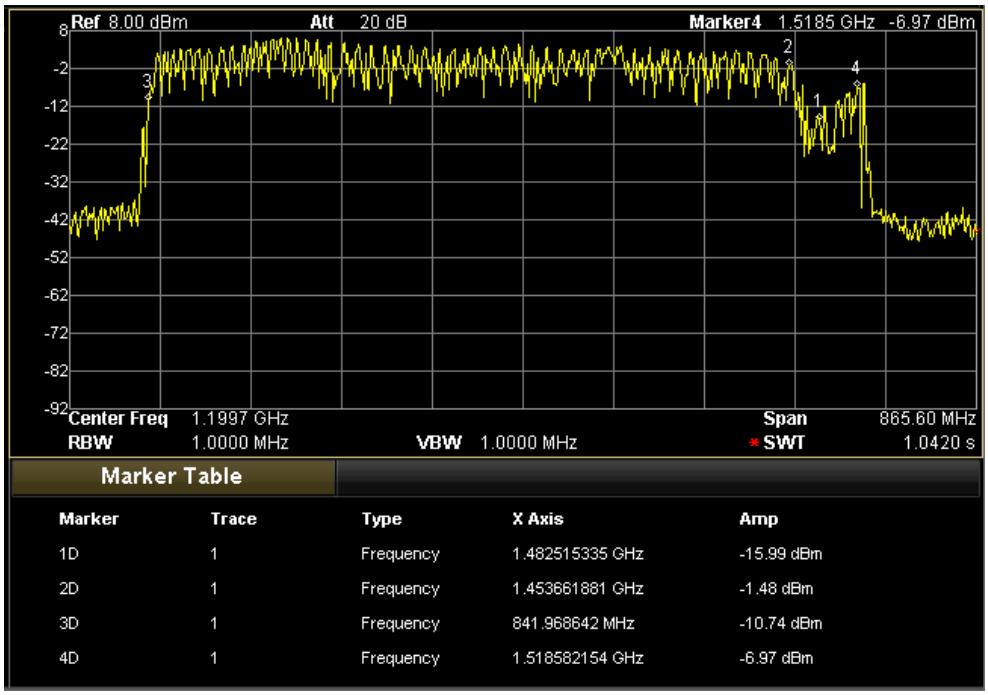


Ilustración 66 – Salida Sensor real cuando el dieléctrico es el aire con sweep de 1 segundo en frecuencia

Las marcas situadas en los extremos (3 y 4) señalan las frecuencias que limitan el principio y el final del ancho de banda, mientras que la marca 1 señala la frecuencia a la que el sensor genera el notch cuando el dieléctrico es el aire y la marca 2 señala la frecuencia a la que el sensor genera el notch cuando el dieléctrico es alcohol.

7.1.4 SALIDA SENSOR CUANDO EL DIELÉCTRICO ES ALCOHOL

Si colocamos ahora un sólido relleno de un dieléctrico distinto del aire, como por ejemplo el alcohol, el sensor debería de realizar un notch a una frecuencia completamente distinta. Es por eso por lo que en la siguiente ilustración observamos este comportamiento. Cuando intentamos ver la salida con un sweep pequeño no observamos bien la respuesta, mientras que si aumentamos el sweep podemos notar el cambio en el notch.

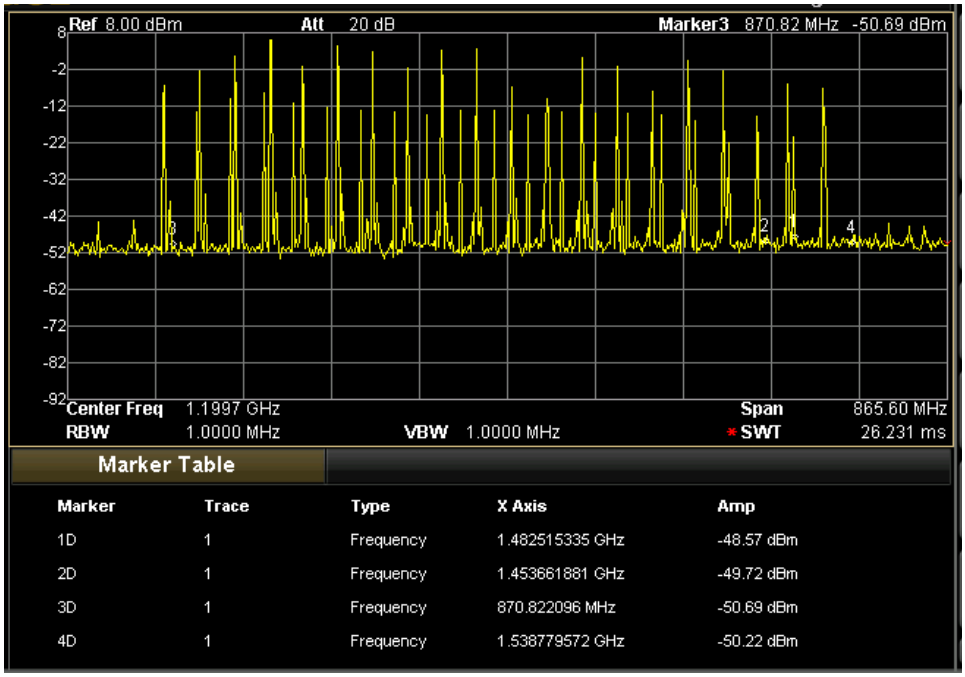


Ilustración 67 – Salida Sensor real cuando el dieléctrico es alcohol con sweep de 26 milisegundos en frecuencia

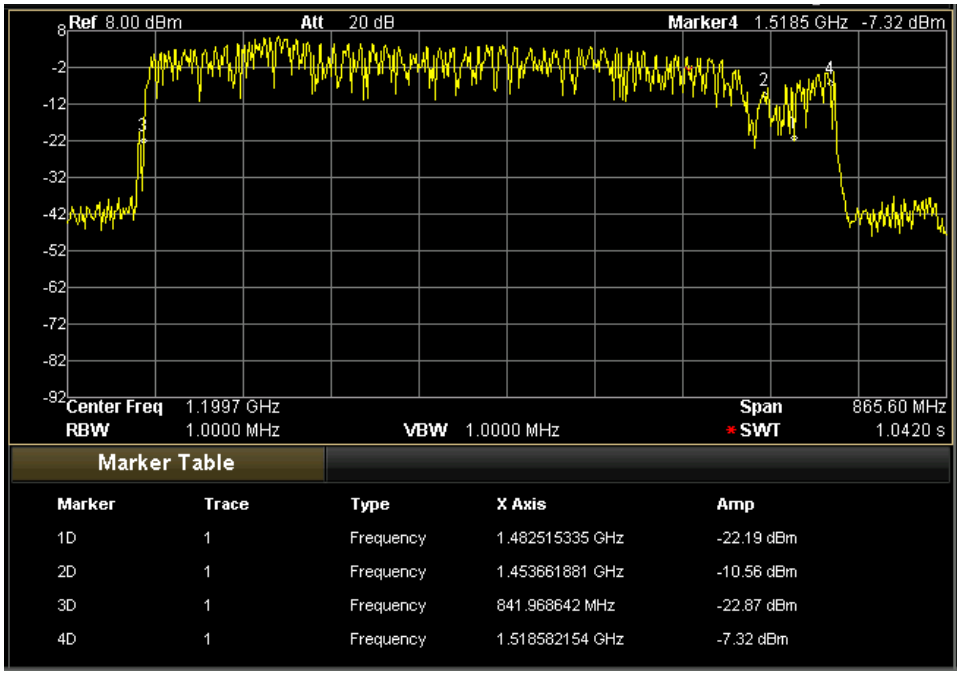


Ilustración 68 – Salida Sensor real cuando el dieléctrico es alcohol con sweep de 1 segundo en frecuencia

7.2 RESULTADOS IDEALES

Ahora procedemos a analizar los resultados frente a un generador de señales en vez de usar el diente de sierra generado por el microprocesador y amplificado posteriormente. Hemos de tener en cuenta que el generador de señales tiende a ser más preciso en las señales que genera. Es verdad que nosotros generamos un diente de sierra cada 1 milisegundo aproximadamente, y teniendo en cuenta que los microprocesadores no se diseñaron expresamente para generar señales analógicas.

A continuación, mostramos la salida del VCO cuando la entrada a este es un diente de sierra producido por un generador de señales con periodo de 1 milisegundo y 4'5 Voltios pico-pico. Al igual que en las anteriores ilustraciones, las siguientes las mostraremos también modificando el sweep, para ser capaces de observar correctamente la respuesta en frecuencia.

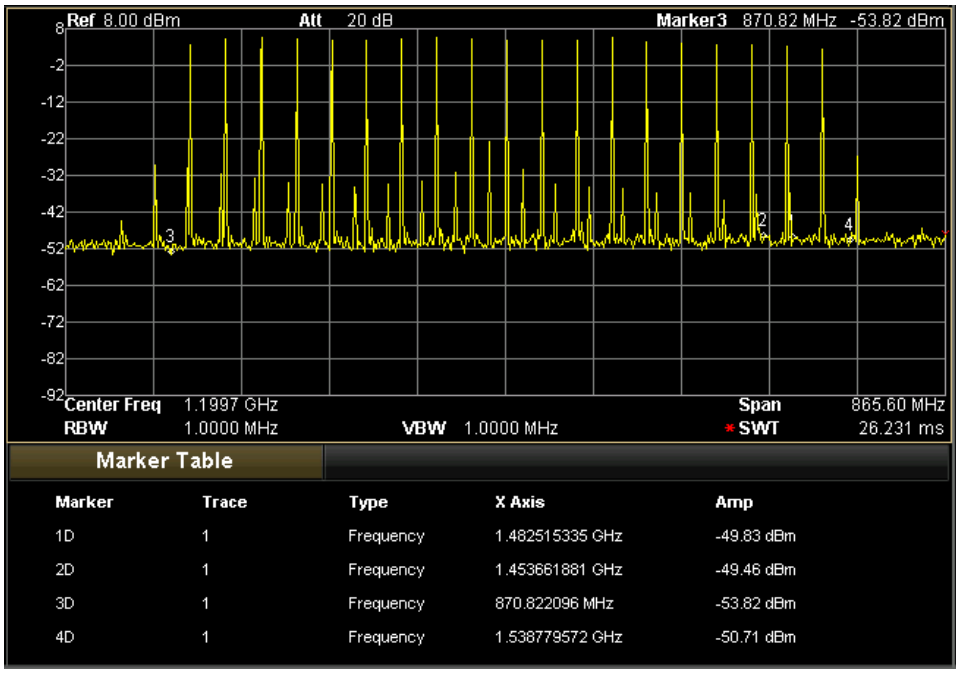


Ilustración 69 – Salida VCO ideal con sweep de 26 milisegundos en frecuencia

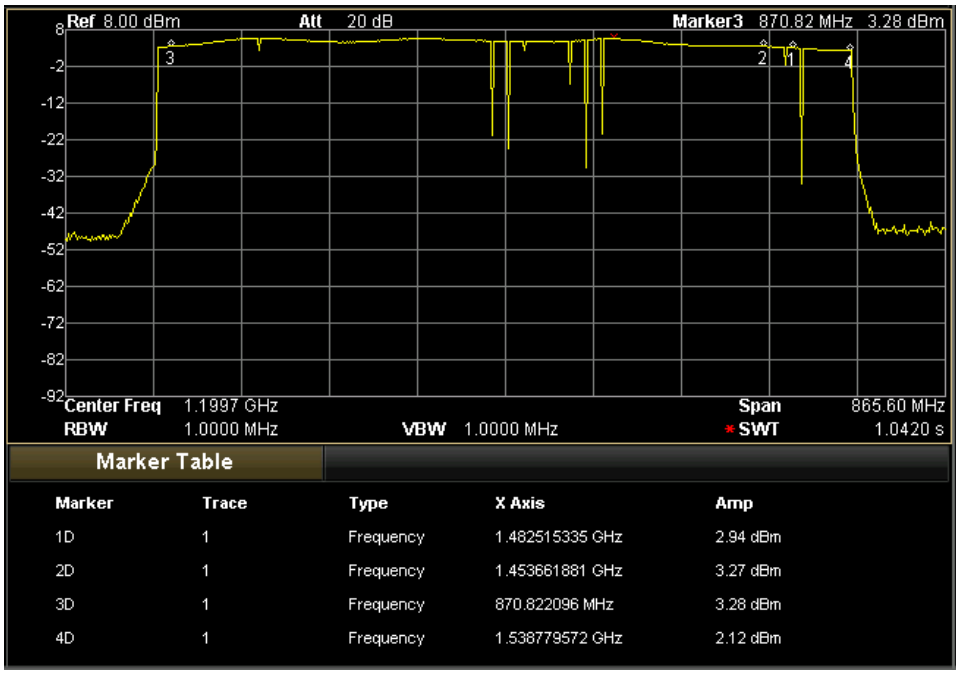


Ilustración 70 – Salida VCO ideal con sweep de 1 segundo en frecuencia

Podemos observar que la salida del VCO se ha generado de forma mas uniforme usando el generador de señales que usando la salida del DAC del microprocesador. Aun así, la diferencia entre las dos respuestas no es crítica. Podemos asegurar que el funcionamiento es correcto.

A continuación, mostramos la salida del sensor cuando el dieléctrico es el aire.

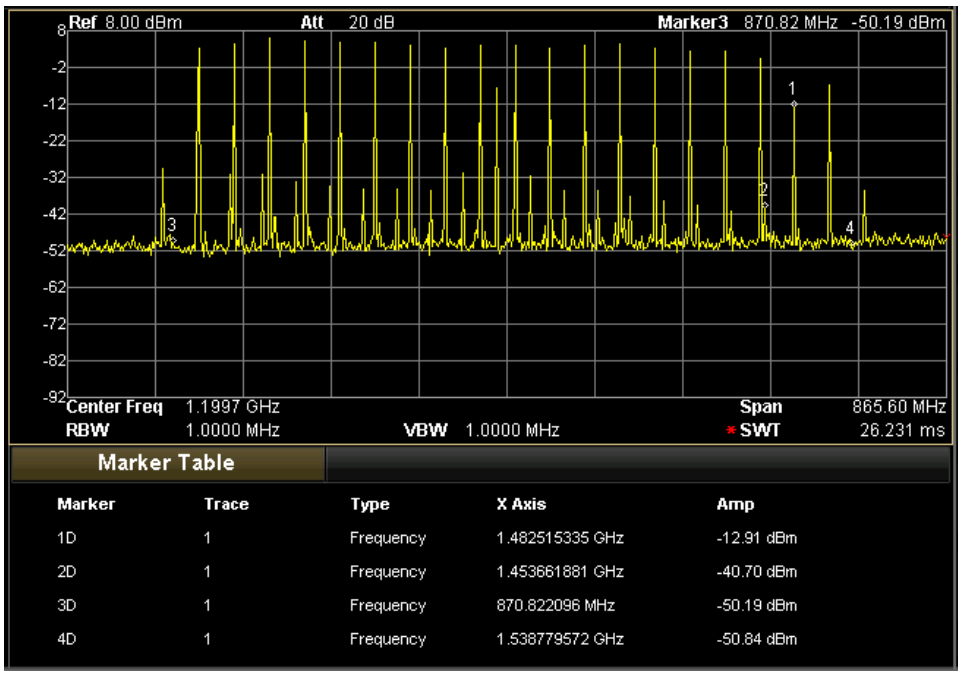


Ilustración 71 – Salida Sensor ideal cuando el dieléctrico es el aire con sweep de 26 milisegundos en frecuencia

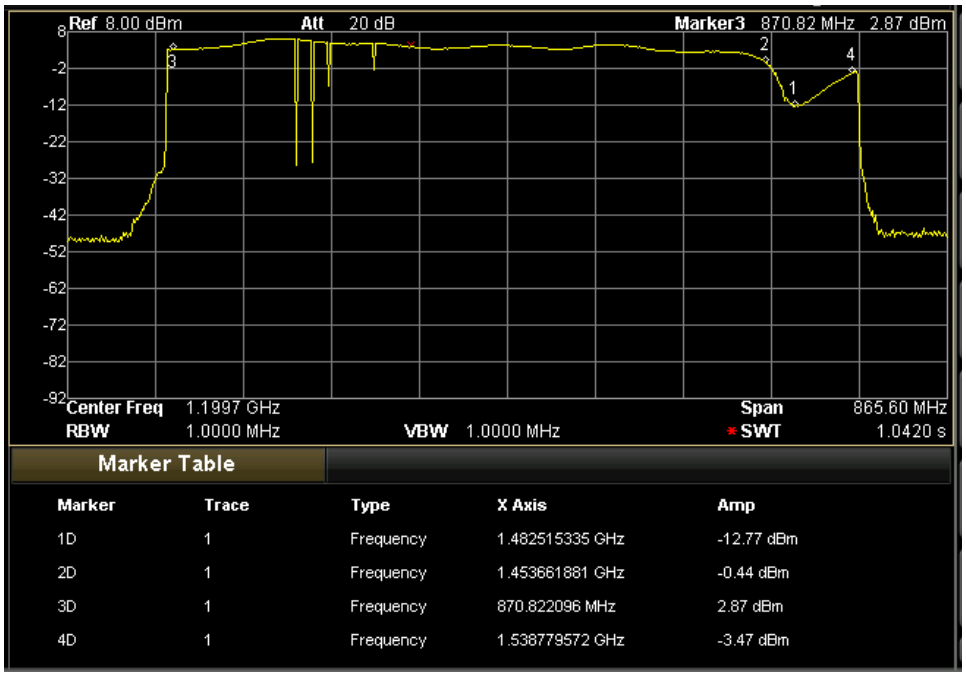


Ilustración 72 – Salida Sensor ideal cuando el dieléctrico es el aire con sweep de 1 segundo en frecuencia

En la Ilustración 72 podemos observar de una forma más clara que en la Ilustración 66 como el sensor elimina una frecuencia determinada. Sucede lo mismo cuando el dieléctrico se trata del alcohol, que observamos ese fenómeno mucho mejor. Esto es debido a que el generador de señales es mucho más preciso, ya que nuestro microprocesador únicamente genera 1000 puntos en el diente de sierra. Podríamos aumentar los puntos, disminuyendo el tiempo de interrupciones del timer, y así tener más precisión en nuestro microprocesador.

A continuación, mostramos la salida del sensor cuando el dieléctrico es alcohol. Podemos argumentar con los mismos razonamientos anteriores, pero con un cambio en la frecuencia de resonancia.

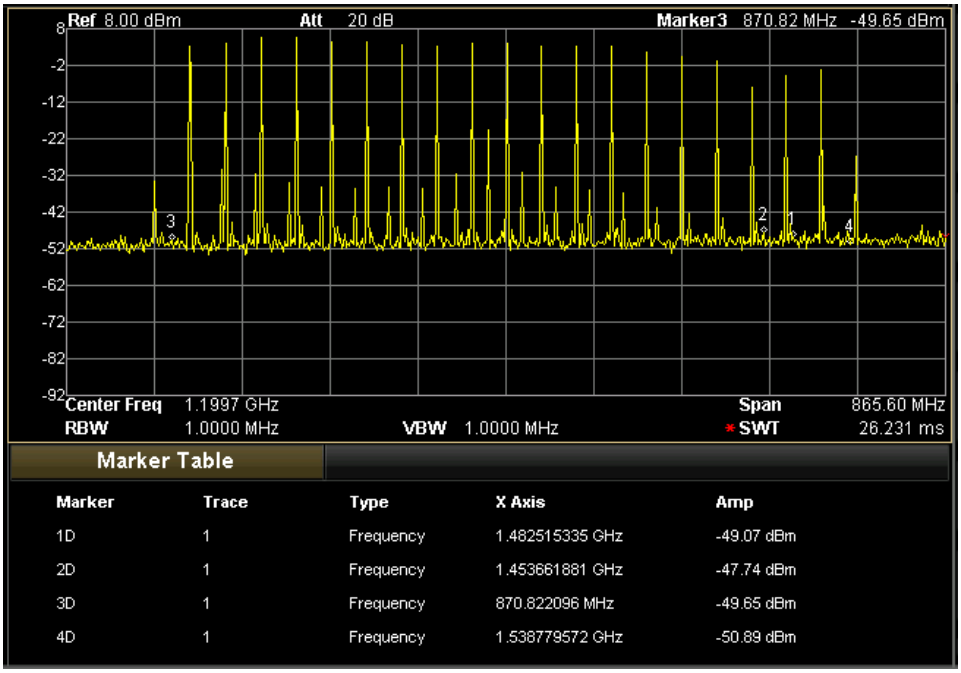


Ilustración 73 – Salida Sensor ideal cuando el dieléctrico es alcohol con sweep de 26 milisegundos en frecuencia

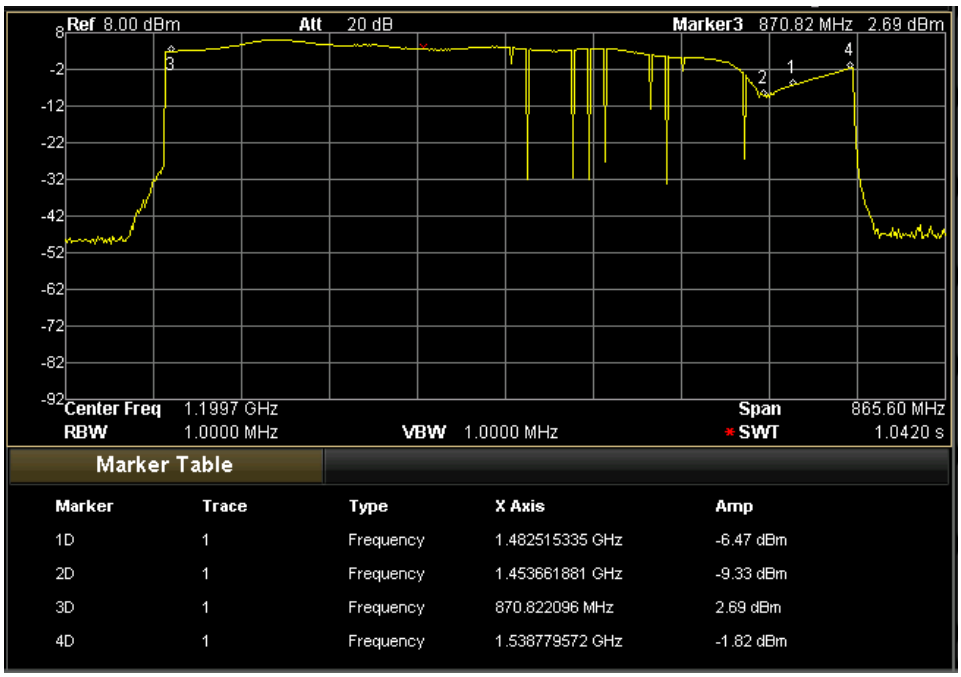


Ilustración 74 – Salida Sensor ideal cuando el dieléctrico es alcohol con sweep de 1 segundo en frecuencia

7.3 RESPUESTA DEL DETECTOR DE ENVOLVENTE

Ahora conectamos la salida del sensor a la entrada coaxial de la placa PCB para que se procese la señal por el filtro y el detector de envolvente. De esta manera podemos observar lo que vamos a obtener a la entrada del microprocesador.

Básicamente, la respuesta del detector de envolvente genera en tiempo la respuesta generada en frecuencia. Es decir, transforma la envolvente vista en el analizador de espectros de la salida del sensor de frecuencia a tiempo, para poder analizarlo en el microprocesador. A continuación, mostramos la salida del detector antes de recibirlo en el microprocesador cuando el dieléctrico es el aire.

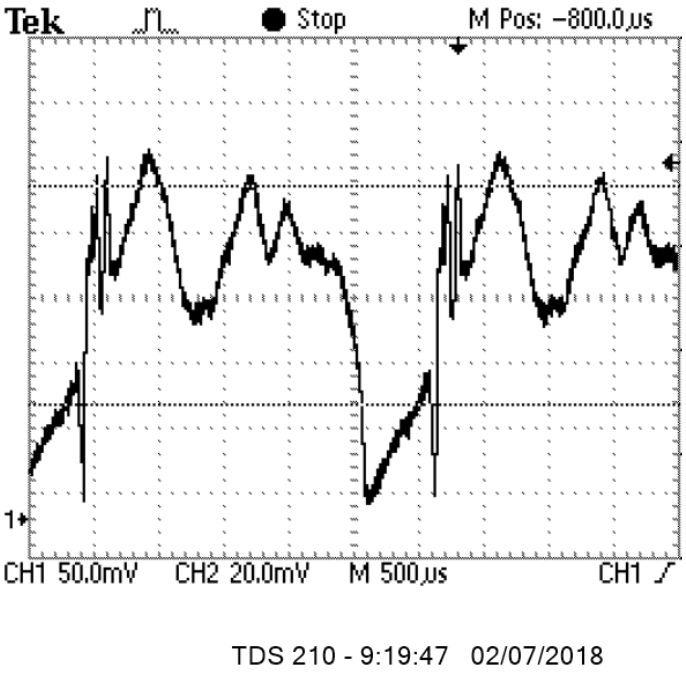


Ilustración 75 – Salida detector de envolvente real cuando el dieléctrico es el aire en tiempo.

Para analizar esta respuesta es necesario centrarse en el valle generado en la Ilustración 75. Si nos fijamos en la sexta columna podemos observar una especie de valle que se hace cero. Eso que aparece en tiempo es el notch en frecuencia generado por el sensor. En función de la colocación de ese punto respecto a los puntos cercanos podemos diferenciar el dieléctrico en el microprocesador.

Esto requiere un análisis de los datos que no entra dentro del alcance de este proyecto, pero que sí podría servir como extensión de un futuro proyecto. En el microprocesador recibimos esta señal y la almacenamos cada vez 1 milisegundo.

Capítulo 8. CONCLUSIONES Y TRABAJOS FUTUROS

El resultado final de este proyecto es un lector hecho mediante técnicas de radiofrecuencia de un sensor capaz de detectar el dieléctrico cercano. Hemos conseguido cumplir los objetivos propuestos al principio de esta memoria:

- Desarrollo del software del microcontrolador:
  - Desarrollado un código basado en lenguaje C para generar las funciones necesarias para enviar y recibir nuestra señal.
  - Desarrollo del software para generar una señal diente de sierra.
  - Recepción de la señal del sensor pasando por el filtro y el detector de envolvente.
  - Pruebas correspondientes a estas funciones desarrolladas.

- Desarrollo y Diseño de una placa PCB para el procesamiento de la señal:
  - Probar el funcionamiento de los componentes por separado:
    - VCO
    - Sensor
    - Detector y Filtro
  - Diseñar el esquema del circuito.
  - Probar cada etapa por separado y comprobar el resultado.

La conclusión es que podemos leer del sensor satisfactoriamente, atendiendo a varios factores que han afectado al funcionamiento de nuestro proyecto:

- El sensor no se diseñó con una precisión exacta, puesto que se trata de un prototipo y no de una versión preparada para el mercado.
- No ha habido un diseño unificado de los distintos componentes, ya que hemos tenido que realizar un amplificador a parte para poder alcanzar los objetivos propuestos.

El proyecto realizado tiene muchas posibles extensiones a poder realizar en un futuro, puesto que tiene un gran interés comercial:

- Utilizar tecnología wireless en los sensores. Añadir pequeñas antenas que puedan recibir y enviar información sin necesidad de cables.
- Añadir al microprocesador módulos para mejorar el rendimiento de éste.
- Ampliar el número de sensores a los que acceder, generando y recibiendo la respuesta.
- Añadir módulos, como aplicaciones web, que nos permitan generar soluciones de visualización de los datos.
- Unificar todo el lector en una sola placa PCB.

Como cometario final, cabe destacar que este proyecto es bastante completo, pero a la vez tiene muchas posibilidades de ampliación, y al tratar un tema de gran actualidad, ofrece además muchas oportunidades en el mercado. Recomendamos por ello que se continúe trabajando en la mejora de este proyecto en el futuro.

Capítulo 9. BIBLIOGRAFÍA

[1] N. Freescale, «KDS_IDE: Entorno de desarrollo integrado de Kinetis® Design Studio (IDE),» [En línea]. Available: https://www.nxp.com/support/developer-resources/software-development-tools/kinetis-design-studio-integrated-development-environment-ide:KDS_IDE.

[2] Kicad, «Kicad EDA tool,» [En línea]. Available: http://kicad-pcb.org/.

[3] ECURED, «Componentes electrónicos,» [En línea]. Available: https://www.ecured.cu/Componente_electrónico.

[4] ECURED, «Componentes electrónicos pasivos,» [En línea]. Available: https://www.ecured.cu/Componentes_electrónicos_pasivos.

[5] By, «¿Qué es el RFID?,» [En línea]. Available: https://www.by.com.es/blog/que-es-rfid/.

[6] Wikipedia, «RFID,» [En línea]. Available: https://es.wikipedia.org/wiki/RFID.

[7] F. NXP, «Manual de referencia KL25z,» [En línea]. Available: https://www.nxp.com/docs/en/reference-manual/KL25P80M48SF0RM.pdf.

[8] NxP, «Kinetis SDK 2.0 API,» [En línea]. Available: https://community.nxp.com/docs/DOC-329783.

[9] Wikipedia®, «Conversor de señal digital a analógica [ADC] [DAC],» [En línea]. Available: https://es.wikipedia.org/wiki/Conversor_de_señal_digital_a_analógica.

[10] Wikipedia, «Oscilador controlado por tensión (VCO),» [En línea]. Available: https://en.wikipedia.org/wiki/Voltage-controlled_oscillator.

[11] NXP, «Información de proveedor del MCU KL25z Kinetis,» [En línea]. Available: https://www.nxp.com/products/processors-and-microcontrollers/arm-based-processors-and-mcus/kinetis-cortex-m-mcus/l-seriesultra-low-powerm0-

plus/freedom-development-platform-for-kinetis-kl14-kl15-kl24-kl25-mcus:FRDM-KL25Z.

[12] Wikipedia, «Retrodispersión,» [En línea]. Available: https://es.wikipedia.org/wiki/Retrodispersi%C3%B3n.

[13] Wikipedia, «Zigbee,» [En línea]. Available: https://es.wikipedia.org/wiki/Zigbee.

[14] DomoDesk, «Zigbee a fondo,» [En línea]. Available: https://www.domodesk.com/216-a-fondo-zigbee.html.

[15] «Kshetrimayum, R. S. (2004). "A Brief Intro to Metamaterials". IEEE Potentials. 23 (5): 44–46. doi:10.1109/mp.2005.1368916.».

[16] «Naqui, Jordi; Durán-Sindreu, Miguel; Martín, Ferran (2011). "Novel Sensors Based on the Symmetry Properties of Split Ring Resonators (SRRs)". Sensors. 11 (12): 7545–7553. doi:10.3390/s110807545. ISSN 1424-8220. PMC 3231717 Freely accessible. PMID 22164031».

[17] G. Galindo-Romera, F. J. Herraiz-Martinez, M. Gil, J. J. Martínez Martínez y D. Segovia Vargas, «Submersible Printed Split-Ring Resonator-Based Sensor for Thin-Film Detection and Permittivity Characterization,» https://ieeexplore.ieee.org/document/7425140/authors.

[18] N. Freescale, «FRDM-KL25Z: Freedom Development Platform for Kinetis® KL14, KL15, KL24, KL25 MCUs,» [En línea]. Available: https://www.nxp.com/products/processors-and-microcontrollers/arm-based-processors-and-mcus/kinetis-cortex-m-mcus/l-seriesultra-low-powerm0-plus/freedom-development-platform-for-kinetis-kl14-kl15-kl24-kl25-mcus:FRDM-KL25Z.

[19] Kicad-pcb, «Eeschema,» [En línea]. Available: http://kicad-pcb.org/discover/eeschema/.

[20] Kicad-pcb, «PcbNew,» [En línea]. Available: http://kicad-pcb.org/discover/pcbnew/.

[21] Crystek, «CVCO55CW-1000-1500,» [En línea]. Available: https://www.mouser.es/datasheet/2/94/CVCO55CW-1000-1500-38335.pdf.

[22] HP, «HSMS-2850 datasheet,» [En línea]. Available: http://www.hp.woodshot.com/hprfhelp/4_downld/products/diodes/hsms2850.pdf.

[23] NxP, «Peripheral driver for the Digital-to-Analog Converter (DAC) module of Kinetis devices.,» [En línea]. Available: http://mcuxpresso.nxp.com/apidoc/group__dac.html.

[24] F. Semiconductor, « LM741 Datasheet (PDF),» [En línea]. Available: https://pdf1.alldatasheet.com/datasheet-pdf/view/53589/FAIRCHILD/LM741.html.

ANEXO A. CÓDIGO

```
/* ###################################################################
 **     Filename    : main.c
 **     Project     : DetectorRF
 **     Processor   : MKL25Z128VLK4
 **     Version     : Driver 01.01
 **     Compiler    : GNU C Compiler
 **     Date/Time   : 2017-09-29, 23:11, # CodeGen: 0
 **     Abstract    :
 **         Main module.
 **         This module contains user's application code.
 **     Settings    :
 **     Contents    :
 **         No public methods
 **
 ** ###################################################################*/
/*!
 ** @file main.c
 ** @version 01.01
 ** @brief
 **         Main module.
 **         This module contains user's application code.
 */
/*!
 **  @addtogroup main_module main module documentation
 **  @{
 */
/* MODULE main */


/* Including needed modules to compile this module/procedure */
#include "Cpu.h"
#include "Events.h"
#include "clockMan1.h"
#include "pin_init.h"
#include "osa1.h"
#include "daConv1.h"
#include "adConv1.h"
#include "pitTimer1.h"
#include "DbgCs1.h"


#if CPU_INIT_CONFIG
#include "Init_Config.h"
#endif
/* User includes (#include below this line is not maintained by Processor Expert)
*/
```

```
#include "stdio.h"

//funciones
void dacDienteSierraAdd(void);
void dacDienteSierraReset(void);
void adcReader(void);
void add_to_array(void);


//Variables globales

//Periodic Interrupt Timer (si acaba el periodo = true)
extern bool ready;

//Variables adc
uint16_t adc_value = 0;
uint16_t adc_array[1010];
int cont_array_adc = 0;

//Variables dac
uint16_t dacOutput=0;
int dacCount=0;

//ADC
/*adc16_config_t adc16ConfigStruct;
adc16_channel_config_t adc16ChannelConfigStruct;
ADC16_DRV_Init(FSL_ADCONV1, &adConv1_ChnConfig0);
ADC16_GetDefaultConfig(&adc16ConfigStruct);
ADC16_Configure(DEMO_ADC16_INSTANCE, &adc16ConfigStruct);
ADC16_EnableHardwareTrigger(DEMO_ADC16_INSTANCE, false);*/

/*lint -save  -e970 Disable MISRA rule (6.3) checking. */
int main(void)
/*lint -restore Enable MISRA rule (6.3) checking. */
{
      /* Write your local variable definition here */
      adc_value = 0;
      adc16_calibration_param_t userCalConfig;
      ready=false;
      /*** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!!
***/
      PE_low_level_init();
      /*** End of Processor Expert internal initialization.
***/

      /* Write your code here */

      //INIT PIT
      PIT_DRV_Init(FSL_PITTIMER1,true);
      PIT_DRV_StopTimer(FSL_PITTIMER1,0);

      //INIT DAC
```

```
      DAC_DRV_Init(FSL_DACONV1,&daConv1_InitConfig0);
      DAC_DRV_ConfigBuffer(FSL_DACONV1,&daConv1_BufferInitConfig0);
      DAC_DRV_Output(FSL_DACONV1,0);

      //INIT ADC
      ADC16_DRV_GetAutoCalibrationParam(FSL_ADCONV1, &userCalConfig);
      ADC16_DRV_SetCalibrationParam(FSL_ADCONV1, &userCalConfig);

      //START TIMER
      PIT_DRV_StartTimer(FSL_PITTIMER1,0);
      for(;;){

            if(ready){
                  // READ FROM ADC
                  adcReader();
                  // WRITE INTO DAC (AND SUM A FIX VOLTAGE)
                  dacDienteSierraAdd();
                  // RESET READY TON COUNT ANOTHER MICROSECOND
                  ready=false;
                  PIT_DRV_StartTimer(FSL_PITTIMER1,0);
            }
            dacDienteSierraReset();

      }
      /* Write your code here */
      /* For example: for(;;) { } */

      /*** Don't write any code pass this line, or it will be deleted during
code generation. ***/
  /*** RTOS startup code. Macro PEX_RTOS_START is defined by the RTOS component.
DON'T MODIFY THIS CODE!!! ***/
  #ifdef PEX_RTOS_START
    PEX_RTOS_START();                  /* Startup of the selected RTOS. Macro is
defined by the RTOS component. */
  #endif
  /*** End of RTOS startup code.  ***/
  /*** Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! ***/
  for(;;){}
  /*** Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! ***/
} /*** End of main routine. DO NOT MODIFY THIS TEXT!!! ***/

/* END main */
/*!
 ** @}
 */
/*
 ** ###################################################################
 **
 **     This file was created by Processor Expert 10.5 [05.21]
 **     for the Freescale Kinetis series of microcontrollers.
 **
 ** ###################################################################
```

```
 */



void adcReader(void){
	/* Comienzo de la conversión */
	ADC16_DRV_ConfigConvChn(FSL_ADCONV1, 0U, &adConv1_ChnConfig0);
	/* Espera a fin de conversión */
	ADC16_DRV_WaitConvDone(FSL_ADCONV1,0);
	/* Lectura de la conversión */
	adc_value = ADC0_RA;

}

void dacDienteSierraAdd(void){
	dacCount++;
	dacOutput+=65;
	DAC_DRV_Output(FSL_DACONV1,dacOutput);

}

void dacDienteSierraReset(void){
	if(dacCount >= 1000){
		dacCount=0;
		dacOutput=0;

	}
}

void add_to_array(void){
	adc_array[cont_array_adc] = adc_value; //guardo en el array el valor
	cont_array_adc++;
	if(cont_array_adc>= 1000){
		cont_arrazy_adc = 0;		//restablezco el contador del array
	}
}
```

```
/* ###################################################################
**     Filename    : Events.c
**     Project     : DetectorRF
**     Processor   : MKL25Z128VLK4
**     Component   : Events
**     Version     : Driver 01.00
**     Compiler    : GNU C Compiler
**     Date/Time   : 2017-09-29, 23:11, # CodeGen: 0
**     Abstract    :
**         This is user's event module.
**         Put your event handler code here.
**     Settings    :
**     Contents    :
**         No public methods
**
** ###################################################################*/
/*!
** @file Events.c
** @version 01.00
** @brief
**         This is user's event module.
**         Put your event handler code here.
*/
/*!
**  @addtogroup Events_module Events module documentation
**  @{
*/
/* MODULE Events */

#include "Cpu.h"
#include "Events.h"

#ifdef __cplusplus
extern "C" {
#endif


/* User includes (#include below this line is not maintained by Processor Expert)
*/
extern bool ready;

/*! adConv1 IRQ handler */
void ADC0_IRQHandler(void){
}

/*! daConv1 IRQ handler */
void DAC0_IRQHandler(void){
}

void pitTimer1_IRQHandler(void)
{
  /* Clear interrupt flag.*/
```

```
  PIT_HAL_ClearIntFlag(g_pitBase[FSL_PITTIMER1], FSL_PITTIMER1_CHANNEL);
  /* Write your code here ... */
  //PIT_DRV_ClearIntFlag(FSL_PITTIMER1,0);
  PIT_DRV_StopTimer(FSL_PITTIMER1,0);
  ready=true;
}

/* END Events */

#ifdef __cplusplus
}  /* extern "C" */
#endif
```