Universitat Oberta de Catalunya uoc.edu

Sistema de control de fatiga de vehículos de flotas

Nombre Estudiante: Sara Hoyos Alonso
Máster Universitario en Ingeniería de Telecomunicación
Área del trabajo final: Electrónica

Nombre Consultor/a: Xavier Saura Mas
Nombre Profesor/a responsable de la asignatura: Carlos Monzo Sánchez

13 de Junio de 2019

Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada 3.0 España de Creative Commons

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Sistema de control de fatiga de vehículos de flotas*
Nombre del autor:	*Sara Hoyos Alonso*
Nombre del consultor/a:	*Xavier Saura Mas*
Nombre del PRA:	*Carlos Monzo Sánchez*
Fecha de entrega (mm/aaaa):	06/2019
Titulación::	*Máster de Ingeniería de Telecomunicación*
Área del Trabajo Final:	*Electrónica*
Idioma del trabajo:	*Español*
Palabras clave	Prevención, monitorización y flota

Resumen del Trabajo:

El proyecto se ubica dentro del contexto de la industria ITS (*Intelligent Transport System*) y tiene como objeto el desarrollo e implementación de un sistema de control de fatiga de vehículos.

El sistema se orienta al control del deterioro de flotas de vehículos mediante la monitorización de parámetros tales como la tensión de las baterías o las horas de conducción. Esta información es almacenada en un entorno web accesible al usuario para la gestión del mantenimiento de los vehículos de su flota.

Además, se generan alarmas orientadas a este mantenimiento. Estas alarmas informan de eventos tales como una disminución de la tensión de la batería de un vehículo por debajo de los valores óptimos o la llegada a las horas máximas de conducción, eventos que suponen la realización de un mantenimiento preventivo del vehículo.

El objetivo de esta monitorización es mejorar la eficiencia y la productividad de la flota así como minimizar riesgos derivados de fallos en la gestión de los mantenimientos de los vehículos.

El desarrollo del sistema se basa en la plataforma KRDM-KL25Z de NXP junto con periféricos de apoyo para comunicación y localización (GPS y GPRS) y adquisición de las señales de interés, como la tensión de batería. El almacenamiento y visualización de los datos se realiza mediante una plataforma abierta de IoT (Internet of Things), Thingspeak. El envío de alarmas se hace mediante un mecanismo de alerta inmediata, que no requiera la consulta de los datos almacenados en la web, el correo electrónico.

Abstract:

The project is englobed into the context of the ITS (Intelligent Transport System) industry and it’s main goal is the development and implementation of a vehicule fatigue control system.

The system is oriented to the deteriority control of vehicle fleets throw the monitorization of some parameters such as battery voltage or conduction hours. These information is stored at a web environment which is accesible to the user to manage it's vehicle fleet maintenance.

Also, alarms oriented to maintenance are generated. These alarms inform about events such as a battery voltage lower than optimus values or the reaching of the maximus contuction time establish, events which imply preventive mantenaince work over the vehicule.

The monitorization goal is to improve the efficiency and productivity of the fleet, as much as minimize risks derivative of vehicle fleet maintenance management failures.

The development of the system is based at the NXP KRDM-KL25Z platform and peripheral support for communication and location (GPS and GPRS) and interested signals adquisition such as battery voltage. Storage and visualization of the data is done by an open IoT (Internet of Things) platform called Thingspeak. Alarms sending is implemented throw an inmediate alert system which must not require to consult the storaged data, the email.

Índice

1. Introducción 1
1.1. Contexto y justificación del Trabajo 1
1.2. Objetivos del Trabajo 2
1.3. Enfoque y método seguido 2
1.4. 1.4 Planificación del Trabajo 2
1.5. Breve sumario de productos obtenidos 3
1.6. Breve descripción de los otros capítulos de la memoria 3
2. Estado del arte 5
2.1. Sistemas de transporte inteligente 5
2.2. Sistemas de monitorización de flotas actuales 9
2.3. Sistemas de posicionamiento global por satélite 10
2.4. Tecnologías móviles 11
3. Definición de la arquitectura del sistema 14
3.1. Descripción y análisis de la plataforma embarcada 16
3.1.1. Bloque de alimentación 16
3.1.2. Bloque de acondicionamiento de señal 19
3.1.2.1. Adquisición de señal de tensión de batería 20
3.1.2.2. Adquisición de señal de contacto 24
3.1.3. Bloque de localización y comunicaciones 28
3.1.4. Bloque de control de alimentación del módulo de comunicaciones 33
3.1.5. Bloque de control 35
3.2. Análisis de componentes software 44
3.2.1. Plataformas software auxiliares 44
3.2.1.1. Servidor de correo electrónico: SMTP2GO 44
3.2.1.2. Plataforma de almacenamiento y visualización: Thingspeak 51
3.2.1.3. Entorno de desarrollo: Mbed 56
3.2.2. Visión global del funcionamiento 58
4. Pruebas realizadas 60
4.1. Test 1: generación de alarma de batería 63
4.2. Test 2: generación de alarma de tiempo de encendido superado 66
4.3. Test 3: generación de alarma de tiempo de conducción superado 69
4.4. Test 4: análisis y monitorización de consumos 72
4.4.1. Análisis y medida de la potencia suministrada por la batería del vehículo 72
4.4.2. Análisis y medida de la potencia suministrada por la batería de litio 76
5. Análisis de los objetivos y propuestas de mejora 78
5.1. Análisis de objetivos 78
5.2. Propuestas de mejora 78
6. Conclusiones 80
7. Glosario 81
8. Bibliografía 83
9. Anexo I: Código desarrollado para plataforma KRDM-KL25Z 86
10. Anexo II: Esquemático de la plataforma embarcada 105

11. Anexo III: Esquemático de la plataforma KL25Z 106

Lista de figuras

Ilustración 1 Diagrama de Gant del TFM 3
Ilustración 2 WBS de los productos obtenidos 3
Ilustración 3 Sistemas de Transporte Inteligente [1] 5
Ilustración 4 Ejemplo de sistema ITS para gestión de flotas de autobuses [6] 7
Ilustración 5 Máquinas de autoventa del Metro de Santigo (Chile) de la empresa GMV [9] 7
Ilustración 6 Ejemplo gráfico de un sistema de carsharing [11] 8
Ilustración 7 Evolución del 1G al 5G: cronología, estándares y mejoras [22] 13
Ilustración 8 Diagrama de bloques del sistema 14
Ilustración 9 Principales características LM350T [23] 17
Ilustración 10 Esquemático de aplicación típica del LM350T [23] 17
Ilustración 11 Esquemático bloque de alimentación [24] 18
Ilustración 12 Fotografías de la implementación de la etapa de alimentación 19
Ilustración 13 Esquema amplificador diferencial [24] 21
Ilustración 14 Esquema acondicionador de tensión de batería del vehículo [24] 23
Ilustración 15 Vista superior del integrado LM358 de Texas Instruments [26] 24
Ilustración 16 Esquema de un comparador básico [24] 25
Ilustración 17 Esquema de comparador básico propuesto para señal de contacto [24] 25
Ilustración 18 Esquema de comparador básico implementado para señal de contacto [24] 27
Ilustración 19 Fotografía de la implementación de la etapa de acondicionamiento de señales 28
Ilustración 20 Fotografía SIM808 EVB-3.2 29
Ilustración 21 Fotografía SIM808 EVB-3.2 Interfaces en uso 31
Ilustración 22 Esquema de conexiones del shield SIM808 evb-3.2 32
Ilustración 23 Fotografía del montaje del shield SIM808 EVB-3.2 32
Ilustración 24 *Shield* Relé e Interfaces 33
Ilustración 25 Esquema de conexiones del shield relés 34
Ilustración 26 Fotografía del montaje de los shield de control tipo relé 34
Ilustración 27 Plataforma de desarrollo KL25Z [27] 35
Ilustración 28 Extracto de las principales características del microcontrolador KL25Z [27] 36
Ilustración 29 Correo electrónico con alarma de tensión de batería inferior al umbral 39
Ilustración 30 Correo electrónico con alarma de tiempo de circulación superior al umbral 40
Ilustración 31 Correo electrónico con alarma de tiempo de encendido superior al umbral 41
Ilustración 32 Entorno de visualización de los datos de la plataforma embarcada en Thingspeak 42
Ilustración 33 Interfaz de login SMTP2GO 46
Ilustración 34 Configuración Usuarios SMTP2GO: SMTP Users 46
Ilustración 35 Configuración Usuario 47
Ilustración 36 Configuración Usuarios SMTP2GO: App Users 47

Ilustración 37 Secuencia de comandos AT para envío de correo electrónico vía SMTP2GO – Parte 1 de 2 48
Ilustración 38 Secuencia de comandos AT para envío de correo electrónico vía SMTP2GO – Parte 2 de 2 48
Ilustración 39 Registro de procesado y entrega de correo electrónico en SMTP2GO y correo asociado recibido en cuenta Gmail 51
Ilustración 40 Información API de acceso del canal Plataforma Embarcada de Thingspeak 53
Ilustración 41 Vista pública del canal Plataforma Embarcada de Thingspeak 54
Ilustración 42 Exportación de datos del canal Plataforma Embarcada de Thingspeak 55
Ilustración 43 Secuencia de comandos AT para envío de datos a Thingspeak previa configuración del módem 55
Ilustración 44 Configuración plataforma de compilación entorno online Mbed 57
Ilustración 45 Información de la plataforma KL25Z disponible en el entorno online de Mbed 57
Ilustración 46 Diagrama de flujo del software de la plataforma de desarrollo KL25Z 59
Ilustración 47 Diagrama de bloques del sistema – Inputs a simular en entorno de test marcadas en rojo 60
Ilustración 48 Diagrama de la interfaz de conexión plataforma embarcada - vehículo 61
Ilustración 49 Extracto de código a modificar para simulación de vehículo en conducción 62
Ilustración 50 Extracto de datos Thingspeak asociados a la prueba 1 con marca en las actualizaciones asociadas a la generación de la alarma de batería 63
Ilustración 51 Bandeja de entrada Gmail con los correos de alarma del 7 de mayo de 2019 64
Ilustración 52 Correo de alarma de batería del 7 de mayo de 2019 64
Ilustración 53 Registro de correos enviados por SMTP2GO el 7 de mayo de 2019 65
Ilustración 54 Extracto de datos Thingspeak asociados a la prueba 2 67
Ilustración 55 Bandeja de entrada Gmail con los correos de alarma del 8 de mayo de 2019 67
Ilustración 56 Correo de alarma de batería del 8 de mayo de 2019 67
Ilustración 57 Correo de alarma de tiempo de encendido del vehículo del 8 de mayo de 2019 68
Ilustración 58 Registro de correos enviados por SMTP2GO el 8 de mayo de 2019 68
Ilustración 59 Extracto de datos Thingspeak asociados a la prueba 3 70
Ilustración 60 Bandeja de entrada Gmail con los correos de alarma del 12 de mayo de 2019 70
Ilustración 61 Correo de alarma de tiempo de encendido del vehículo del 12 de mayo de 2019 70
Ilustración 62 Correo de alarma de tiempo de circulación del vehículo del 12 de mayo de 2019 71
Ilustración 63 Registro de correos enviados por SMTP2GO el 12 de mayo de 2019 71
Ilustración 64 Medida de consumo máximo del relé en su alimentación de 3.3V 75

Ilustración 65 Medida de consumo máximo del módulo de comunicaciones durante la obtención de la posición GPS en su línea de alimentación de 12V. 75
Ilustración 66 Medida de consumo máximo del módulo de comunicaciones durante la transmisión de datos por GPRS en su línea de alimentación de 12V. 75
Ilustración 67 Verificación de la reciente actualización de Thingspeak tras la medida con consumo del SIM808 durante la transmisión de datos por GPRS. 76
Ilustración 68 Características de fuentes de alimentación de la plataforma KL25Z [27] 76
Ilustración 69 Medida de consumo máximo de la plataforma KL25Z en su interfaz de alimentación USB de 5V. 77

1. Introducción

1.1. Contexto y justificación del Trabajo

El trabajo se ubica dentro del ámbito de los denominados ITS (*Inteligent Transport System*), sistemas orientados a proporcionar soluciones basadas en la tecnología a las problemáticas derivadas del transporte en todos sus ámbitos.

Dentro del amplio espectro abarcado por este término, el trabajo se centra en el mundo de la automoción, más de forma específica en el mantenimiento de los vehículos.

Un punto clave para la seguridad en carretera es la prevención, directamente asociada al mantenimiento preventivo para detección de deficiencias en los vehículos.

Si bien resulta sencilla la planificación de estos mantenimientos para un único vehículo particular, esto puede tornarse más complicado para la gestión de grandes flotas de vehículos.

Es importante notar que el momento de realizar un mantenimiento no tiene porqué seguir siempre una periodicidad exacta pues depende de factores tales como el número de kilómetros recorridos.

Una solución segura no basada en la tecnología consiste en establecer tiempos entre revisiones cortos, de forma que en ningún caso se sobrepase el momento necesario de revisión. Esto puede suponer un sobrecoste en el mantenimiento de las flotas, pues es posible que ciertos vehículos se estén revisando con una periodicidad excesiva.

En este contexto, este trabajo plantea el diseño y desarrollo de un sistema de monitorización de vehículos orientado a la gestión de su mantenimiento. Este sistema vigila las métricas asociadas a la necesidad de mantenimiento, almacena los datos y genera avisos cuando una de estas métricas indica la necesidad de revisar un vehículo de la flota.

Este sistema no solo facilita la gestión del estado de flotas de vehículos, sino que permite evitar el sobrecoste de establecer tiempos entre revisiones demasiado cortos a vehículos que no lo precisen.

1.2. Objetivos del Trabajo

Los objetivos principales del proyecto son los siguientes:

• Desarrollo e implementación de un sistema completo que permita la monitorización y almacenamiento en una plataforma web de código abierto de parámetros clave en el desgaste de un vehículo y la generación de alarmas en relación a estos parámetros.

• Generación de la documentación y los entregables de las distintas fases de desarrollo tanto desde la perspectiva hardware como software.

• Adquirir las capacidades relacionadas con el uso de plataformas de desarrollo embebidas basadas en el uso de un procesador ARM.

• Definir un sistema con bloques definidos abierto a ampliaciones.

1.3. Enfoque y método seguido

La estrategia planteada es el desarrollo integral del sistema apoyándose en una plataforma de desarrollo comercial.

Otras posibles estrategias pasarías por el desarrollo de la electrónica de control, a más bajo nivel.

La estrategia escogida permite abstraerse de aspectos de bajo nivel del diseño y centrarse en la funcionalidad completa del sistema, logrando la implementación de una plataforma funcional en un periodo corto de tiempo.

1.4. 1.4 Planificación del Trabajo

Se definen las tareas a realizar, las relaciones, el orden y la estimación de tiempos mediante el uso de un diagrama de Gantt en el cual los tiempos se han adaptado al calendario de la PECs.

La planificación se realiza de la siguiente manera:

- Especificación funcional del sistema: definición de las funciones a realizar y de los bloques funcionales a implementar.
- Búsqueda de información y definición de los elementos hardware y software a utilizar para la implementación del sistema definido.
- Definición del sistema a bajo nivel tanto a nivel de interconexión hardware como de interacción software.
- Montaje del entorno hardware de test.
- Desarrollo del software de control y configuración del entorno con apoyo en el entorno hardware de test.
- Definición de los modos de test y realización de pruebas.
- Análisis de objetivos y puntos a mejorar.

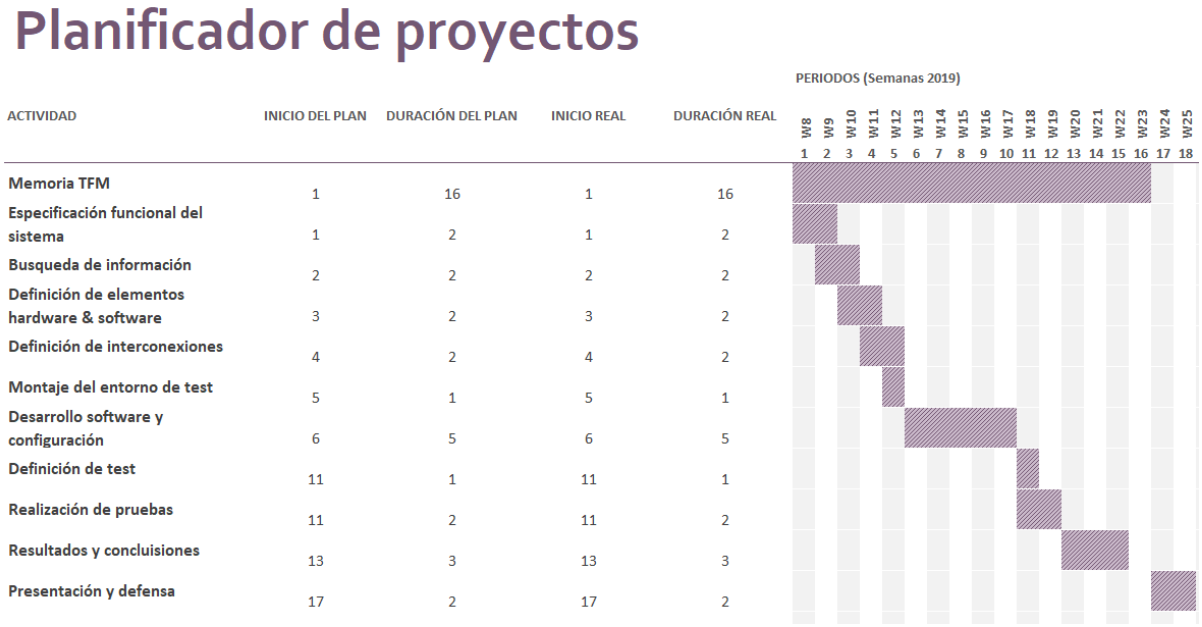


Ilustración 1 Diagrama de Gant del TFM

1.5. Breve sumario de productos obtenidos

Ilustración 2 WBS de los productos obtenidos

1.6. Breve descripción de los otros capítulos de la memoria

En el Capítulo 1 se realiza una introducción del TFM (Trabajo Fin de Máster). En ella se describen los objetivos del trabajo, el método seguido y su planificación.

En el Capítulo 2, se realiza una descripción del mundo del ITS para finalmente focalizarse en el aspecto concreto de la automoción. Además, se realiza una introducción a las distintas tecnologías a utilizar.

En el Capítulo 3, se describe y analiza el sistema de monitorización implementado. Partiendo de un esquema general de cajas negras se desglosa cada una de ellas tanto a nivel hardware como software, para finalizar como una descripción global del funcionamiento completo.

En el Capítulo 4, se exponen las pruebas realizadas sobre el sistema con objeto de verificar su funcionalidad.

En el Capítulo 5, se analiza el alcance de los objetivos propuestos y se proponen posibles mejoras para el sistema.

Finalmente, en el Capítulo 7, se exponen las conclusiones del TFM.

2. Estado del arte

En este apartado se realiza un análisis de los elementos de interés del TFM con el objeto de definir unas directrices claras que nos guíen a la hora de focalizar el proyecto.

2.1. Sistemas de transporte inteligente

El objetivo tratado en este proyecto es un ejemplo más los denominados ITS (*Inteligent Transport System*). Los sistemas de transporte inteligente surgen como un mecanismo de control del tránsito y transporte en su ámbito general, orientados a proporcionar una solución a problemas tales como la congestión de vehículos, los accidentes o la contaminación ambiental, aspecto en primera plana en el momento actual en que las ciudades comienzan a buscar modos de disminuir su contaminación debido a sus preocupantes niveles.

La Ilustración 3 presenta en una vista general multitud de ejemplos de estos sistemas, que nos muestras su extendido uso. Algunos ejemplos mostrados en la imagen son el uso de la tecnología para establecer un control adaptativo del tráfico, establecer sistemas de información al usuario o gestión de flotas.

El desarrollo a gran escala de estos sistemas se ve impulsado por el masivo aumento del transporte en las grandes ciudades y se establece como una ayuda para la gestión de estos sistemas masivos.

Ilustración 3 Sistemas de Transporte Inteligente [1]

Resulta importante destacar que en la sociedad actual el transporte constituye un factor principal tanto en el crecimiento económico como en el desarrollo de la sociedad en el ámbito cultural. Esta influencia se debe a que la disponibilidad de un transporte eficiente permite una mayor movilidad de los agentes económicos e impulsa por tanto la existencia de un mercado global dónde las distancias pierden su importancia. Además, la facilidad del flujo de personas permite a los individuos acceder a otros ambientes e ideas de un modo nunca posible en otros momentos de la historia de la humanidad. Estos aspectos en conjunto impulsan la evolución cultural de la sociedad a un espectro de conocimiento más amplio y global. [2]

La aplicación de las nuevas tecnologías al área del transporte se ha convertido durante los últimos años en una realidad cotidiana en constante y rápida evolución. En un futuro cercano la aplicación de las TIC (Tecnologías de Información y Comunicación) con el IoT (*Internet of Things*) y el *BigData* a la cabeza permitirán lograr una mayor optimización y aprovechamiento de los canales de transporte, fundamental para afrontar los futuros retos. [3] El uso conjunto de estas tecnologías permite construir el modo de recolección, procesado y diseminación de la información en tiempo real.

La monitorización del transporte resulta básica para poder trazar un mapa del estado actual de la red a gestionar. La evolución de los métodos de monitorización (que incluyen cámaras, sensores, *tags* de identificación…) para la obtención de información más específica y completa permite mejorar la toma de decisiones por parte del sistema de procesado. [4]

Por otra parte, las tecnologías de comunicación inalámbricas son el medio que habilita la recolección de los datos monitorizados y la emisión de órdenes generadas por su procesamiento hacia el sistema en tiempo real.

La normalización del uso de las nuevas tecnologías para el transporte hace que en la actualidad nadie se sorprende de conocer el tiempo de espera de un autobús urbano mediante una pantalla de led o de autenticarse a la entrada del Metro mediante un título de transporte almacenado en una tarjeta sin contacto con tecnología RFID (*Radio Frequency Identification*) y, fruto de esta continua evolución, en poco tiempo a nadie le llamará la atención el poder acceder a cualquier medio de transporte directamente con su tarjeta bancaria sin contacto (mediante la tecnología EMV (*Europay MasterCard VISA*)) sin necesidad de contar con un título de transporte específico para dicho medio [5].

Estos sistemas componen multitud de soluciones tales como redes inteligentes de semáforos, sistemas de peajes automáticos, sistemas de gestión integral de autopista y túneles, equipos para monitorización y gestión de flotas para entorno industrial, sistemas orientados a la gestión de redes de transporte público (venta y/o recarga de títulos de transporte, autenticación, gestión de vehículos…),…

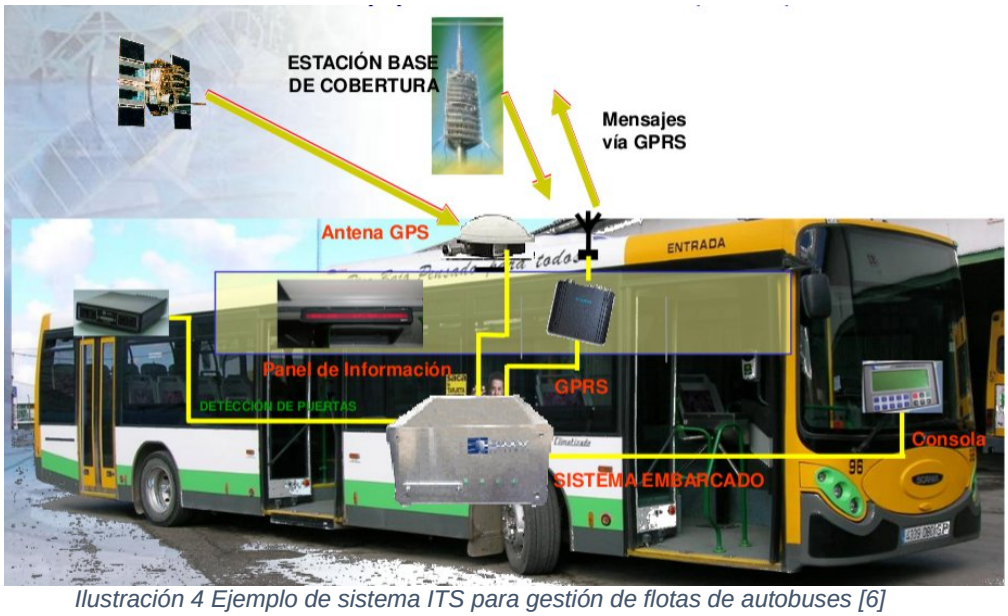


Ilustración 4 Ejemplo de sistema ITS para gestión de flotas de autobuses [6]

En España contamos con empresas de peso multinacional en la industria del Transporte Inteligente tales como Indra o GMV. Ambas copan gran parte del mercado de las ITS a nivel nacional y cuentan con proyectos internacionales de importancia en este ámbito como:

- Sistema de gestión de tráfico del tranvía de Taiwan (China), GMV. [7]
- Sistema de venta de título y control de accesos en el sistema de transporte público de Riad (Arabia Saudí), Indra. [8]
- Máquinas de autoservicio de venta y recarga de títulos de transporte de las líneas 3 y 6 del Metro de Santigo (Chile), GMV. [9]
- Sistema de *ticketing* de la línea de alta velocidad Meca-Medina, Indra. [8]
- Sistema de seguimiento y comunicación de la línea de tranvía ligero de Sydney (Austraria), GMV. [10]

Ilustración 5 Máquinas de autoventa del Metro de Santigo (Chile) de la empresa GMV [9]

Dentro del mundo del transporte inteligente nos centramos de forma más específica en el área de automoción, dónde se localiza nuestro proyecto, el cual está orientado a la monitorización de vehículos de una flota para el control de la fatiga de los vehículos con objeto de facilitar y predecir la gestión de las tareas de mantenimiento de la misma.

Dentro del sector del transporte inteligente en el área de automoción se encuentran soluciones orientadas a objetivos como: posicionamiento, monitorización y notificación del estado del vehículo, sistemas de seguridad (eCall, recuperación de vehículos robados…), sistemas de *carsharing*…

En la actualidad el área de automoción se encuentra, al igual que el global del sector, en pleno apogeo. Un ejemplo actual de sistemas de transporte inteligente en el área de automoción implementado con éxito son las plataformas para compartición de vehículos tales como emov, car2go, wible…. Estas plataformas ofrecen a los usuarios la posibilidad de utilizar coches eléctricos para la realización de desplazamientos en las grandes ciudades pagando por el tiempo de uso de los vehículos. Para dar soporte a esta aplicación se utilizan distintas soluciones ITS para la gestión de las flotas de vehículos, monitorización de los mismos e implementación de funcionalidades básicas para la compartición tales como el bloqueo y desbloqueo de puertas a través de la aplicación móvil asociada para utilizar un vehículo de la plataforma aparcado en el área urbana.

Ilustración 6 Ejemplo gráfico de un sistema de carsharing [11]

Uno de los principales motivos por los que aplicaciones como emov o car2go han gozado de tan buena implantación es el hecho de que los vehículos de sus flotas son eléctricos. Esto se debe a que en la actualidad nos encontramos en un momento clave en la lucha contra la contaminación en las ciudades y ya son muchas de ellas las que cuentan con medidas de restricción de circulación para vehículos no eléctricos (Madrid, Barcelona, Valladolid …). La posibilidad de circular sin restricción unida a la facilidad de aparcamiento (estas plataformas cuentan con acuerdos con los aparcamientos para aparcar en zonas restringidas como la zona azul sin coste adicional para el usuario) hace de estas plataformas una opción apetecible para los usuarios afincados en los entornos urbanos. [11]

A la vista del momento actual y mirando hacia el futuro se puede anticipar que este sector será clave en el futuro, no sólo por los proyectos ya implementados si no por los proyectos futuros aún en fase inicial como los vehículos autónomos.

Centrando más el foco en el objetivo de este proyecto, se puede decir que existen ya multitud de soluciones orientadas a la monitorización de flotas que brindan a las empresas de transporte por carreta la posibilidad y ventaja de conocer datos como la ubicación de los vehículos de su flota, su estado de actividad o el modo en que está transcurriendo el servicio.

Los datos obtenidos a partir de la monitorización de los vehículos de la flota pueden posteriormente orientarse a distintos objetivos tales como: [12] [13]

- Detección de incidencias en los vehículos: apertura de puertas en marcha, alarmas de accidente…
- Detección de deficiencias en el modo de conducción de los chóferes, tales como exceso de velocidad.
- Detección de incumplimiento de las normativas de descansos obligatorios tras un número determinado de horas de conducción.
- Identificación de rutas ineficientes.
- Predicción de tareas de mantenimiento preventivo
- …

Dentro de los objetivos de los distintos sistemas destacamos el lograr un aumento de la **seguridad** en el transporte por carretera ya que multitud de parámetros tales como el modo de conducción o el estado del vehículo son fundamentales para mantener un grado de protección apropiado para la realización de servicios de transporte por carretera.

La seguridad será por tanto un objetivo claro y siempre presente a la hora de orientar este proyecto.

2.2. Sistemas de monitorización de flotas actuales

Tal y como se comentaba anteriormente, existen multitud de soluciones hardware-software integradas comerciales para la monitorización de flotas centradas en proporcionar diferentes servicios en función del objetivo de la monitorización.

Estas soluciones constan de dos bloques complementarios y diferenciados: el hardware y el software.

Respecto al software podemos tomar como ejemplo las implementaciones ofertadas por empresas como MovilData [14] o Drivin [15].

Al igual que en la mayoría de las soluciones de este estilo, ambas plataformas almacenan la información monitorizada en una plataforma externa a los vehículos de la flota y ofrecen al usuario la posibilidad de consultarla a través

a través de distintas plataformas telemáticas (web, móvil, software dedicado…).

Esta facilidad en el uso es un requisito básico para estas soluciones orientadas a un público industrial amplio: desde una gran empresa con presencia internacional y flotas de gran tamaño (como Alsa, con flotas de vehículos de gran tamaño) hasta una pequeña empresa con una flota de vehículos de reparto.

A la vista de esto, un punto fundamental por tanto a tener en cuenta en nuestro proyecto será la facilidad en el acceso a los datos de cara a un usuario universal.

A nivel hardware los sistemas actuales cuentan con los siguientes elementos: [16]

- Sistema de posicionamiento global basado en GNSS (Global Navigation Satellite System), responsable de la identificación de la posición del vehículo.

- Integración con el tacógrafo digital del vehículo para extracción de datos relacionados con distancias recorridas, tiempos de conducción, paradas …

- Integración con sensores IoT para obtención de parámetros auxiliares como temperatura de componentes del vehículo, consumos, nivel de combustible …

- Sistema de comunicación inalámbrico (GPRS, 3G, 4G…) para almacenamiento datos en un servidor externo.

- Sistema de control de los componentes integrados en el sistema.

La gran mayoría de empresas del sector cuentan con un hardware dedicado propio o comercial que integra todos los elementos descritos anteriormente.

El sistema de gestión a utilizar será una plataforma de desarrollo comercial sobre la cual se integrarán el resto de elementos necesarios para conformar el sistema completo.

2.3. Sistemas de posicionamiento global por satélite

Los sistemas de posicionamiento global por satélite (GNSS) son en la actualidad el principal modo de posicionamiento utilizado. El uso de un receptor GNSS permite mediante el procesado de la señal de los satélites de su constelación obtener la posición del mismo con errores de precisión de metro o incluso centímetros, si bien la precisión en la localización se limita para uso civil. [17]

Actualmente los principales sistemas de posicionamiento global son:

- GPS (Global Positioning System). Sistema de posicionamiento que utiliza la constelación de satélites americana, logra una precisión de 7.8m para uso civil.

- GLONASS. Sistema de posicionamiento que utiliza la constelación de satélites rusa, logra una precisión de 7.4m para uso civil.

- Galileo. Sistema de posicionamiento que utiliza la constelación de satélites europea, aún incompleta. Logrará una precisión de 1m, sin restricciones para uso civil.

- BeiDou. Sistema de posicionamiento originalmente orientado a China y alrededores pero que prevé extenderse a nivel global, tal y como anunció a finales de 2018. [18]

A nivel global el sistema GPS es el sistema GNSS más utilizado, principalmente porque fue el primero en orientarse hacia el uso civil.

Cabe notar que en la actualidad existe un movimiento que apoya el uso de sistemas que utilicen la señal de los sistemas ruso y americano de forma conjunta, logrando un aumento de la velocidad y la precisión con el sistema, si bien por el momento esta solución no se ha extendido. [19]

Para este proyecto es preciso el uso de un sistema GNSS. El sistema escogido será el americano, GPS, principalmente por su uso más extendido.

2.4. Tecnologías móviles

Un elemento fundamental de un sistema de monitorización de flotas es el envío de información desde los vehículos en movimiento a un punto fijo. La tecnología utilizada entra por tanto dentro del término denominada “comunicaciones móviles” que describe cualquier radio-enlace de comunicación entre dos terminales móviles de los cuales al menos uno está en movimiento o en una ubicación indeterminada.

Las distintas generaciones de tecnologías móviles y su evolución es la siguiente: [20]

- 1G
  Basada en canales analógicos permitía el envío de voz con baja seguridad y mala transferencia en el traspaso entre celdas.

- 2G
  Da el salto a los protocolos digitales siendo GSM (Global System for Mobile Communications) el más conocido. Es la primera generación en permitir la transmisión de datos e implementa un cierto nivel de encriptación en las comunicaciones.

- 2.5G

  Ante la nueva necesidad de una mayor velocidad en el tráfico de datos para navegar por Internet el GSM evoluciona hacia el GPRS, que logra este aumento de velocidad hasta los 250 kbps.

  Esta tecnología permite la navegación por Internet, enviar y recibir correo, y descarga de datos y soporte, así como el uso de la mensajería instantánea.

- 3G

  Con la evolución de Internet la necesidad de mayor velocidad de transmisión continúa aumentando, lo que da lugar a la aparición de la tecnología UMTS.

  Esta tecnología permite alcanzar mayores velocidades de transmisión de datos (hasta 2Mbps) y mayor capacidad de voz y datos, lo que permite a los usuarios acceder a internet desde sus terminales y disfrutar de servicios derivados de este tales como la visualización de videos online.

- 4G

  La tecnología LTE establece un sistema basado totalmente en IP. Contiene importantes mejoras en seguridad y QoS, además de ofrecer velocidades mayores al 3G, de hasta 100 Mbps. El aumento de la velocidad ha propiciado el uso de aplicaciones como Whatsapp o aplicación de videollamada como Facebook Live o Skype

- .4.5G

  Previo a la aparición del 5G aparece la tecnología LTE-A-PRO, cuya principal mejora es el aumento de la capacidad de transmisión superando los 400Mbps.

- 5G

  Siguiendo la línea de evolución marcada, el 5G ofrecerá a los usuarios mayor velocidad de transferencia da datos e imágenes. Actualmente esta tecnología no está disponible para los usuarios pero ya se encuentra en proceso de pruebas. [21]

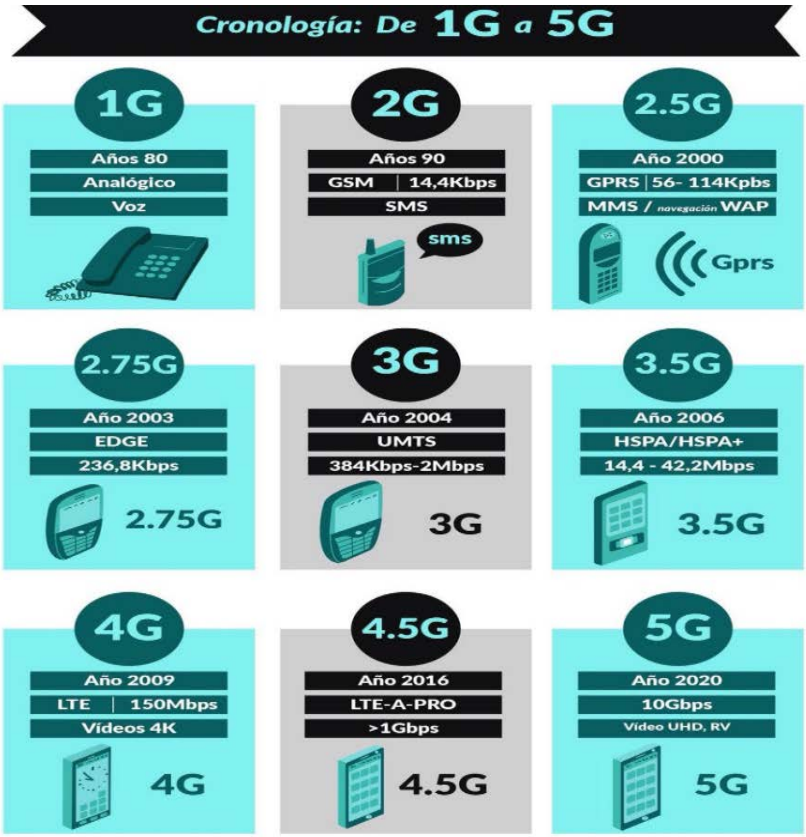


Ilustración 7 Evolución del 1G al 5G: cronología, estándares y mejoras [22]

Actualmente impera el uso de los sistemas de cuarta generación, si bien el 3G y el 2G siguen en funcionamiento.

En el proyecto utilizarán los estándares de segunda generación GSM y GPRS, pues su tasa de datos es suficiente para la funciones a implementar.

3. Definición de la arquitectura del sistema

Tal y como se adelantaba en la introducción, el objetivo del sistema es la monitorización de un conjunto de parámetros de los vehículos de una flota y su almacenamiento y procesado para la generación de alarmas orientadas al mantenimiento de los vehículos de dicha flota.

La siguiente figura muestra el diagrama de bloques del sistema a implementar, dónde las flechas en rojo simbolizan las alimentaciones, las verdes el flujo de datos y las moradas las acciones de visualización por parte del usuario final.

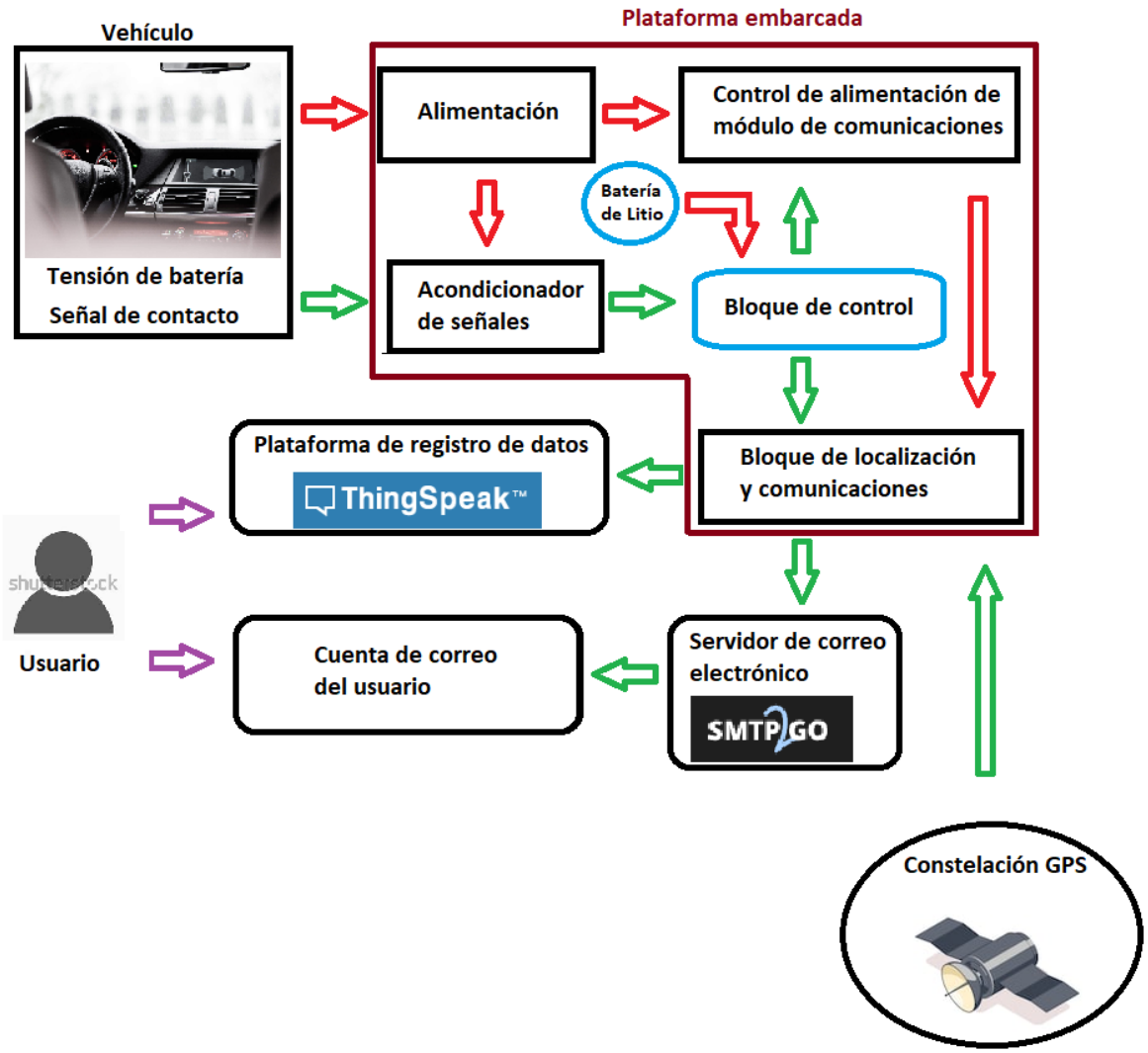


Ilustración 8 Diagrama de bloques del sistema

El análisis del sistema planteado se desglosa en dos bloques claramente diferenciados:

1. Plataforma embarcada
2. Sistema global

El primer punto, desarrollado en el apartado 3.1, se centra en la descripción de los elementos que integran los bloques de la plataforma embarcada. Es decir, la descripción de los elementos hardware integrados en el vehículo y su interconexión a nivel lógico y electrónico para lograr el funcionamiento del sistema.

En el segundo punto, que abarca el bloque 3.2 de esta memoria, se analiza la iteración de la plataforma planteado con los servidores externos utilizados para la integración del sistema global: el servidor de correo SMTP2GO y el servidor web para almacenamiento de datos Thingspeak.

Por otra parte, se describe la plataforma de desarrollo web utilizada, Mbed, y se analiza el diagrama de funcionamiento lógico del bloque de control, implementado por la plataforma de desarrollo KLM25Z.

3.1. Descripción y análisis de la plataforma embarcada

En este apartado se analizan los bloques que componen la plataforma embarcada del sistema y su implementación a nivel hardware.

Tal y como se veía en el diagrama presentado al inicio de este capítulo (*Ilustración 8*), la plataforma embarcada se compone de los siguientes bloques:

1. Bloque de alimentación
2. Bloque de acondicionamiento de señal
3. Bloque de control
4. Bloque de localización y comunicaciones
5. Bloque de control de alimentación del módulo de comunicaciones

A continuación se describen y analizan los distintos bloques definidos.

3.1.1. Bloque de alimentación

La energía necesaria para el funcionamiento del sistema es proporcionada directamente por la batería del vehículo a monitorizar con una tensión nominal de 12V.

Dada la naturaleza de los componentes utilizados, el sistema requiere de 3 tensiones de alimentación: 12, 5 y 3.3V.

Si bien la batería del vehículo proporciona una tensión nominal de 12V se ha planteado que en realidad la tensión de esta batería puede oscilar dentro de un rango que se ha considerado de 10 a 14V. Es por ello que, además de plantear una serie de esquemas que permitan obtener las tensiones de 5 y 3.3V no presentes en el sistema, se debe implementar un circuito de alimentación que permita obtener una tensión de 12V fijos.

Por otra parte, la tensión en los bornes de las baterías sufre variaciones durante procesos tales como el arranque del vehículo. Estos perfiles de comportamiento pueden llegar a dañar el sistema, de forma que se debe asegurar que la tensión de alimentación que llega a los elementos que integran la plataforma embarcada no sufre este tipo de variaciones.

Es decir, el objeto de esta etapa será obtener tensiones **estables en el tiempo y de valores específicos** de 12, 5 y 3.3V para la alimentación (y control) de los elementos que integran nuestro sistema.

El elemento elegido para lograr esta estabilidad es un regulador lineal, que por definición es un regulador basado en un componente activo trabajando en su zona lineal y un elemento pasivo trabajando en su zona de ruptura.

La limitación de este elemento en términos de uso viene dada por el hecho de que sólo permite reducir o mantener la tensión, pero nunca aumentarla.

Esta limitación no nos afecta en el caso que nos ocupa, dónde partimos de una tensión de 12V para obtener 12, 5 y 3.3V.

Una de las principales ventajas por las cuales se ha escogido este elemento es su bajo ruido a la salida, lo que permite obtener valores **muy estables**. Otro motivo es el coste, mucho menor que el de otras alternativas tales como las fuentes conmutadas. Además, permite la realización de un diseño sencillo y versátil.

El integrado escogido para el diseño es el regulador lineal ajustable LM350T. [23]

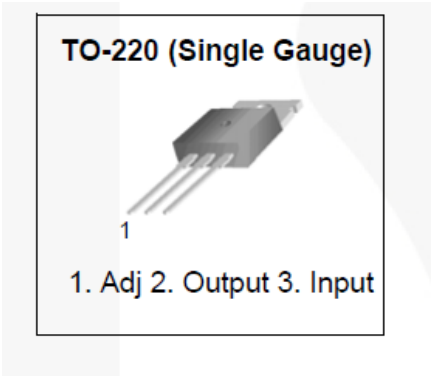


Ilustración 9 Principales características LM350T *[23]*

Este elemento permite ajustar su salida entre 1.2 y 33V con una tensión diferencial máxima entrada-salida de 35, lo cual cubre todo el rango de necesidad.

La siguiente imagen muestra el esquema típico de aplicación, utilizado en el diseño de esta etapa.

Typical Application(2)

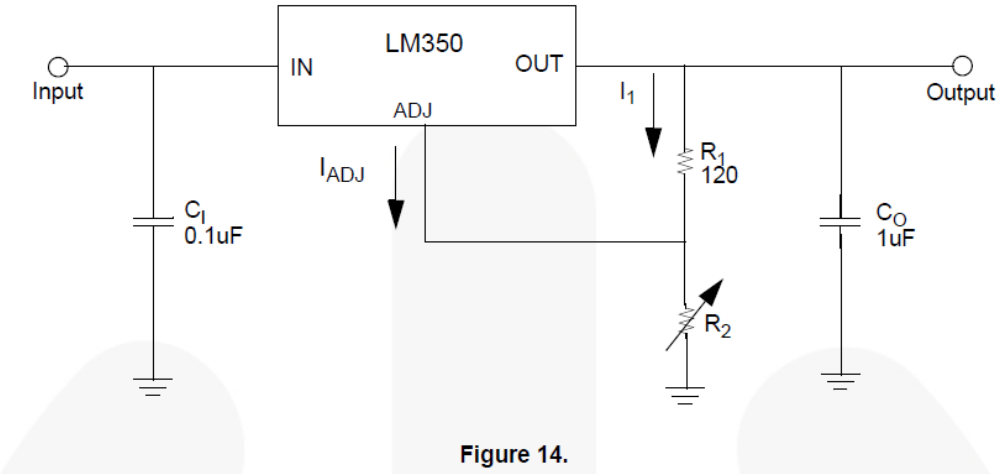


Ilustración 10 Esquemático de aplicación típica del LM350T *[23]*

A la vista del esquemático los elementos externos al regulador son los siguientes:

- C_I: este condensador es requerido cuando existe una distancia apreciable entre la fuente y el LM. Esto afectaría por tanto al diseño puesto que los equipos de transporte inteligente en vehículos son típicamente instalados en el salpicadero del vehículo dónde se dispone de puntos de conexión a la señal de batería y de contacto, por lo que existirá un cableado de una cierta longitud hasta la batería, origen de la señal de alimentación. Además, su uso es recomendado teniendo en cuenta consideraciones de EMC (Electromagnetic Compatibility).

  - C_O: este condensador, cuyo valor puede oscilar entre 1 y 100µF, es utilizado para mejorar la impedancia de salida y el rechazo de transitorios.

  - R_1 y R_2: estas resistencias permiten ajustar la tensión de salida del regulador. Dada una tensión de referencia en operación V_{REF} de 1.25V la salida del regulador VO vendrá dada por la siguiente expresión.

$$V_O = 1.25V\left(1 + {}^{R_2}/_{R_1}\right)$$

A continuación se muestra el esquema definido en el cual se implementan 3 reguladores lineales en cascada con Vbat y GND_bat obtenidos de los bornes de la batería del vehículo.

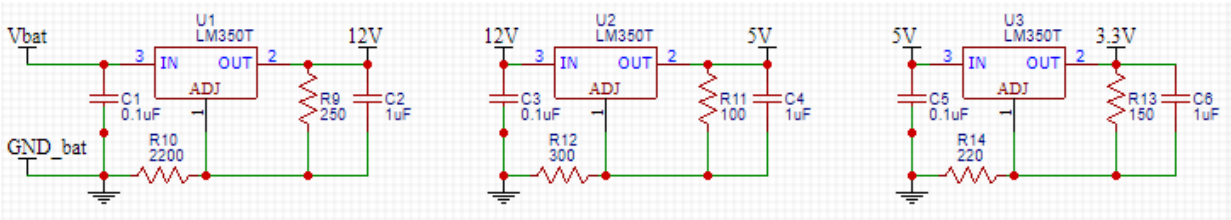


Ilustración 11 Esquemático bloque de alimentación *[24]*

La configuración en cascada permite que la diferencia VI-VO sea menor en cada caso, frente a la opción de implementar los tres reguladores en paralelo, lo que permite lograr una mayor eficiencia en la conversión.

Debido al uso para el montaje del prototipo en *proto-board* de resistencias comerciales, cuyos valores son limitados, la implementación real obtiene valores ligeramente distintos de los valores objetivos pero dentro de los rangos de funcionamiento requeridos.

- Salida de U1: Vo=1.25(1+(2200/250))=12.25V
- Salida de U2: Vo=1.25(1+(300/100))=5V
- Salida de U3: Vo=1.25(1+((100+120)/150))=3.08V

A continuación se muestran unas fotografías de la implementación de esta etapa.

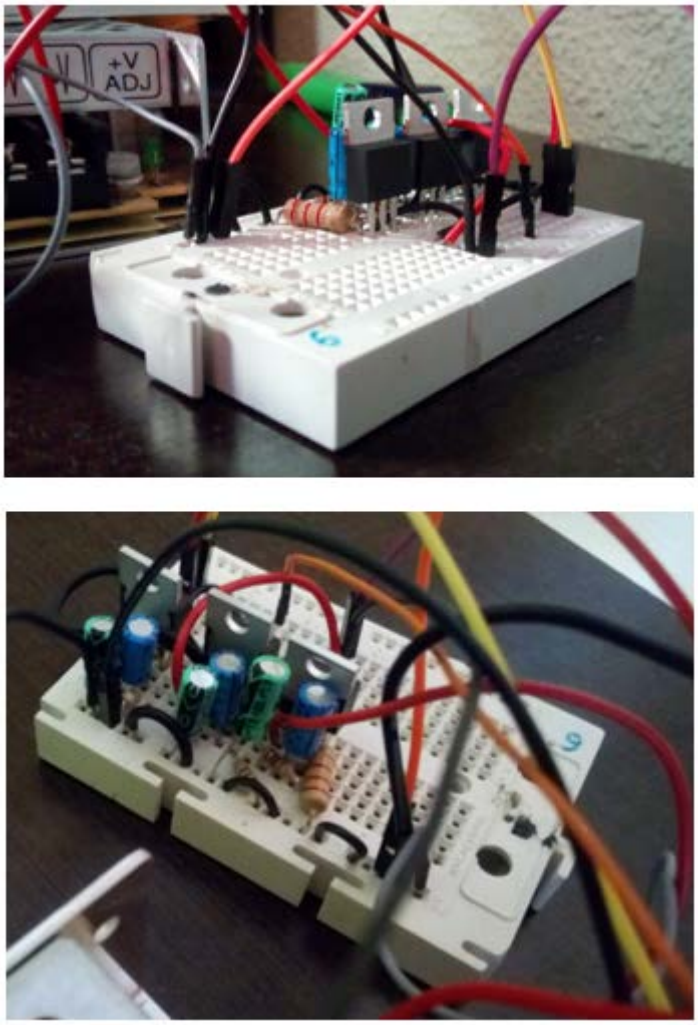


Ilustración 12 Fotografías de la implementación de la etapa de alimentación

3.1.2. Bloque de acondicionamiento de señal

El sistema basa su funcionamiento en la monitorización de 3 variables: tensión de batería del vehículo, señal de contacto del vehículo y posición.

Dos de estas variables son señales eléctricas presentes en el vehículo que la plataforma embarcada deberá leer, estas señales son:

- Tensión de batería del vehículo
- Señal de contacto del vehículo

El primer punto a la hora de determinar el modo de adquisición de estas señales es conocer su naturaleza.

La tensión de la batería del vehículo es una señal analógica presente de forma constante, la batería siempre tendrá tensión en sus bornes independientemente del estado en el cual se encuentre el vehículo.

La tensión en los bornes de la batería proporciona información del estado de salud de la misma. Es importante detectar el momento en el cual una batería comienza a presentar un comportamiento degradado con el objeto de realizarle una revisión, la realización de este preventivo evitará que continúe su proceso de degradación que puede llegar a generar que en un momento dado la batería no sea capaz de arrancar el vehículo. [25]

Por regla general, una batería de vehículo cargada al 100% de su capacidad presenta una tensión típica en reposo de 12.7V, mientras que con una carga del 80% presenta una tensión de 12.4V o inferior.

Para este diseño se considera que el rango de valores a medir de la batería oscilará entre 10 y 14V.

Respecto a la señal de contacto, es una señal de control que habilita el motor de arranque. Esta señal se activa al girar la lleve de arranque del vehículo (o al detectar la tarjeta en vehículos sin llave mecánica). Esta tensión es provista por la batería del vehículo por lo que presentará el mismo rango de tensiones.

Previo a la inyección de sendas señales en la plataforma de desarrollo KLM25Z será preciso realizar una adaptación de las mismas. A continuación se analiza el modo de adaptación propuesto para cada una de ellas.

3.1.2.1. Adquisición de señal de tensión de batería

Como se indicaba anteriormente, la señal de tensión de batería es una señal analógica con valores de tensión típicos entre 10 y 14V.

Esta señal debe ser introducida en el sistema a través de una de las entradas analógicas de la plataforma KL25Z para ser procesada por un CAD (Convertidor Analógico Digital).

El CAD de la plataforma KL25Z establece un valor digital de salida de 16 bits y tiene un rango de entrada de 0 a 3.3V. Esto se traduce en que la placa es capaz de discernir entre 2^{16}= 65536 valores con una resolución de $\frac{3,3-0}{2^{16}-1} = 5.035\ 10^{-5}\ V$ para dicho rango de tensiones.

En este caso la señal de entrada presenta un rango de entrada de 0 a 14V motivo por el cual es necesario introducir una etapa de adaptación de la señal con objeto de adaptar la señal a monitorizar al rango de entrada de la plataforma de desarrollo.

Una cuestión a tener en cuenta es si la resolución que se obtendrá para medir dicha señal será suficiente para la precisión requerida por el sistema.

Si la señal de rango 0 a 14V es mapeada directamente al rango 0 a 3.3 V la resolución en la medida será la siguiente:

$$Resolución: \frac{14-0}{2^{16}-1} = 0.000213\ V$$

Para el objetivo de esta monitorización será suficiente con una resolución de 0.2mV en la medida de la tensión de batería por lo que a priori bastaría con mapear de forma directa el rango de 0 a 14V en el rango de entrada de CAD de 0 a 3.3V.

De no haber sido así hubiera sido necesario la introducción de un bloque previo (por ejemplo con un amplificador operacional en configuración de restador) para eliminar el offset hasta 10V, lo que permitiría aumentar la resolución en la medida pero limitaría el rango tomando los valores inferiores a 10V como 10V falseando un caso crítico en que la batería dejase de proporcionar tensión.

Con este objetivo se define el esquema de modificación del rango. Para ello se utiliza un esquema basado en amplificadores operacionales, de amplia utilización en el ámbito de la sensorización.

El amplificador operacional es uno de los dispositivos más versátiles y ampliamente utilizados en cualquier esquema de adaptación de señales, permite amplificar, sumar, restar, comparar… Su versatilidad es el principal motivo por el cual se ha elegido para el planteamiento de este esquema.

El esquema funcional en el cual se basa el esquemático es el amplificador diferencial.

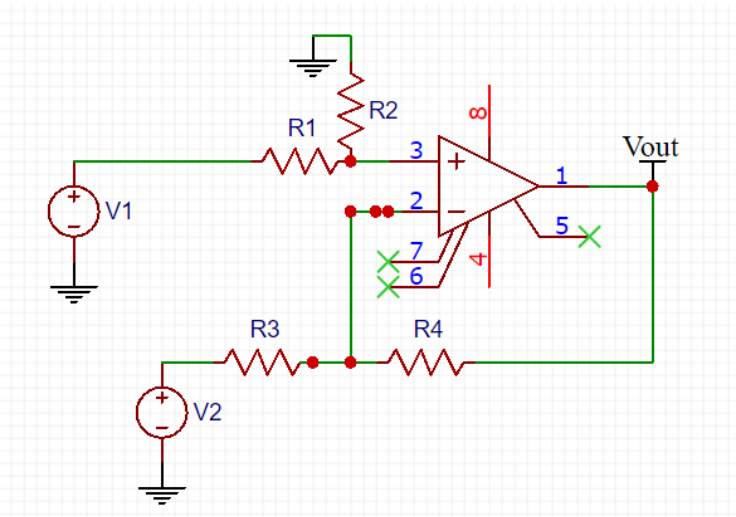


Ilustración 13 Esquema amplificador diferencial *[24]*

Este esquema permite obtener una tensión proporcional a la tensión diferencial de entrada ($V_d = V_1 - V_2$) manteniendo la relación $R_3R_2 = R_4R_1$ entre las resistencias de configuración.

Su salida quedaría como sigue:

$$V_O = K\ (V_1 - V_2), con\ K = \frac{R_4}{R_3} = \frac{R_2}{R_1}$$

En el caso analizado el valor de V_1 se corresponde a la tensión de batería y V_2 a GND. El máximo valor de tensión en batería considerado, 14V, debe corresponderse con el máximo valor admitido por el CAD, 3.3V, de forma que la ecuación a resolver para establecer el esquemático será la siguiente:

$$V_O = 3.3 = K\ (V_1 - V_2) = \frac{R_4}{R_3}\ 14, con\ \frac{R_4}{R_3} = \frac{R_2}{R_1}$$

Se obtiene que la relación a cumplir debe ser:

$$\frac{R_4}{R_3} = \frac{R_2}{R_1} = \frac{3.3}{14}$$

Con el objetivo de realizar una conversión más escalonada y añadir una protección adicional a nuestro circuito, se establece un divisor de tensión previo de forma que la tensión que recibe el esquema de amplificador operacional sea la mitad de la tensión de batería.

De este modo la relación a cumplir por las resistencias de dicho operacional será la siguiente:

$$\frac{R_4}{R_3} = \frac{R_2}{R_1} = \frac{3.3}{7}$$

Dónde 7 es la mitad de la tensión máxima considerada en los bornes de la batería.

Es importante notar que para lograr una buena precisión en la conversión las resistencias deben de tener la menor tolerancia posible, típicamente un 1%.

El amplificador operacional tomará como alimentaciones las señales de GND y 12V obtenidas de la etapa de alimentación, con 12V superior a la tensión máxima diferencial disponible.

El esquema finalmente implementado es el siguiente:

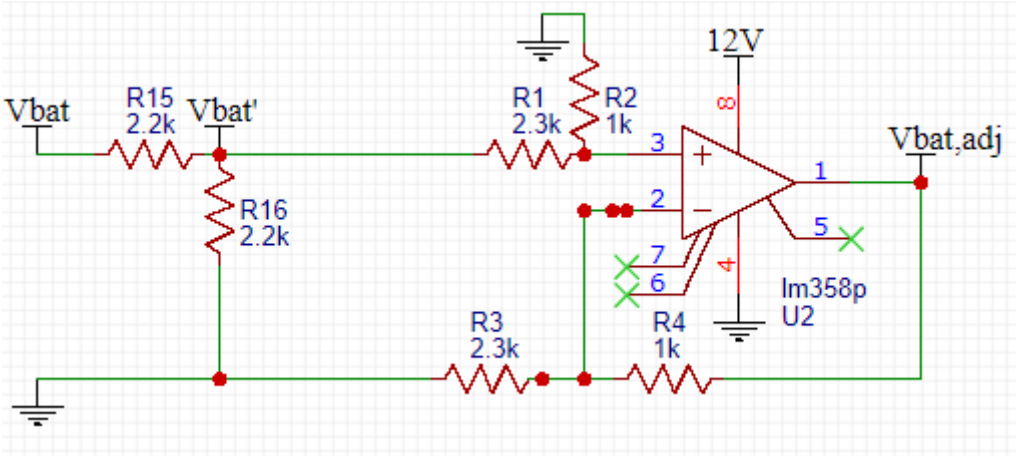


Ilustración 14 Esquema acondicionador de tensión de batería del vehículo *[24]*

Al igual que ocurrió en la etapa de alimentación, debido al uso para el montaje del prototipo en *proto-board* de resistencias comerciales, cuyos valores son limitados, la implementación real obtiene valores ligeramente distintos de los valores objetivos pero razonables.

$$R_1 = R_3 = 2.3\ k\Omega$$
$$R_2 = R_4 = 1.1\ k\Omega$$
$$R_{15} = R_{16} = 2.2\ k\Omega$$

De forma que:

$$V'_{bat} = V_{bat}\frac{R_{16}}{R_{16}+R_{15}} = \frac{V_{bat}}{2} \leq \frac{14}{2} = 7\ Vdc$$
$$V_{bat,adj} = V'_{bat}\frac{R_4}{R_3} \leq 7\frac{R_4}{R_3} = 3.34Vdc$$

El amplificador operacional utilizado es el integrado LM358P de Texas Instrument. [26]

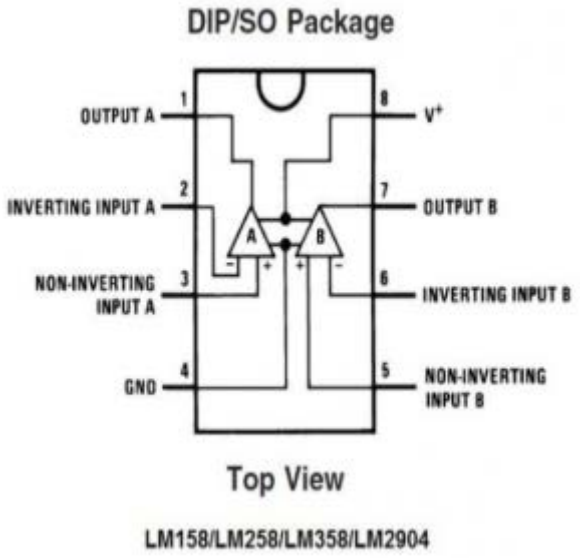


Ilustración 15 Vista superior del integrado LM358 de Texas Instruments *[26]*

Este integrado incluye dos operacionales de alta tensión, hasta 36V, orientados a aplicaciones de alta sensibilidad con bajo offset siendo estos los principales motivos por los cuales se ha escogido para nuestra aplicación.

Dado que el objetivo último de este esquema es obtener la tensión de batería es crítica la sensibilidad en la conversión con objeto de monitorizar de forma correcta esta tensión. Por otra parte, al trabajar con valores altos de tensión en términos de sensorización se precisa de un operacional de alta tensión.

Finalizada esta etapa dispondremos de la tensión de batería del vehículo adaptada al rango de entrada de las entradas analógicas de la plataforma de desarrollo KL25Z, responsable de su procesado.

3.1.2.2. Adquisición de señal de contacto

La señal de contacto es una señal que toma valores 0 o tensión de la batería en función de si está o no activa. Esta señal tiene por tanto un carácter digital y podrá ser monitorizada por una de las entradas digitales de la plataforma de desarrollo KL25Z.

Al igual que en el caso descrito anteriormente, la señal de interés se encuentra fuera del rango de entrada aceptado por la placa de desarrollo, de 0 a 3.3V. Esto supone que resulta necesario el desarrollo de un circuito de adaptación con objeto de obtener una señal adaptada a la entrada de la plataforma de desarrollo.

Al igual que para la tensión de batería, el circuito se basa en el uso de un amplificador operacional.

En este caso dada la naturaleza binaria de la señal, el esquema base utilizado es el denominado comparador básico, mostrado en la siguiente ilustración

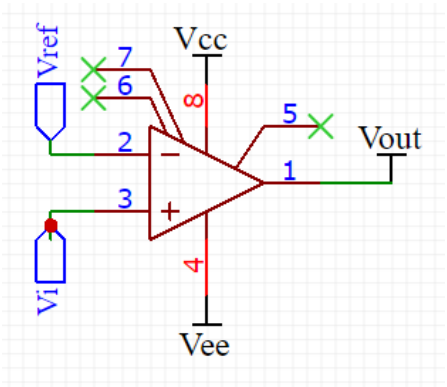


Ilustración 16 Esquema de un comparador básico *[24]*

En este esquema la salida del amplificador toma el valor V_{cc} si la entrada Vin es mayor que la señal de referencia, y toma el valor Vee si la señal de entrada el menor que la señal de referencia.

Con objeto de tener a su salida un valor digital 0/3.3V se establece el siguiente esquema.

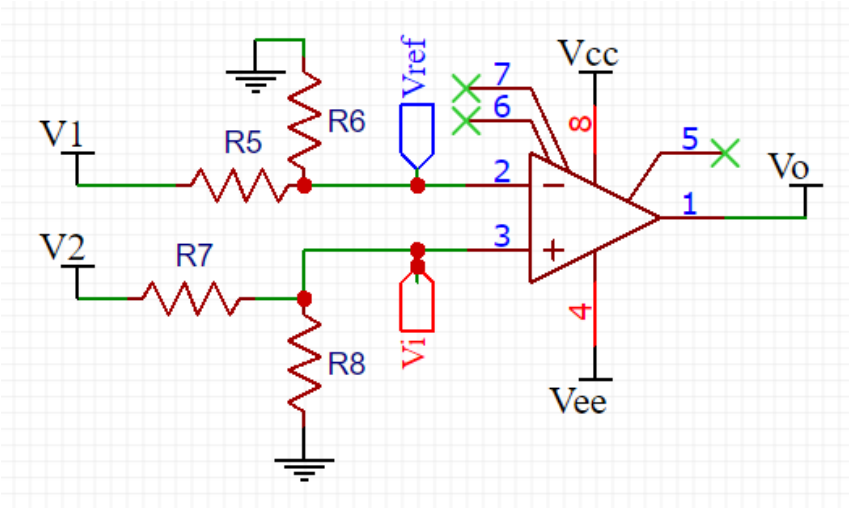


Ilustración 17 Esquema de comparador básico propuesto para señal de contacto *[24]*

Con:

- V1: la tensión de batería del vehículo.
- V2: la señal de contacto.
- Vcc = 3V
- Vee = GND.

La relación entre los valores de resistencias y tensiones de entrada al amplificador operacional son las siguientes:

$$V_{-} = V_{ref} = \frac{R_6}{R_6 + R_5} V_{bat}$$
$$V_{+} = V_i = \frac{R_8}{R_7 + R_8} V_{contacto}$$

Los valores de las resistencias para la señal de contacto se establecen de forma tal que para la tensión más alta considerada, 14V, no se supere a la entrada del amplificador operacional la señal Vcc de 3.3V.

$$V_{+} = V_i = 3.3 > \frac{R_8}{R_7 + R_8} 14$$

Se toman como ejemplo los valores $R_8 = 1k\Omega$ y $R_7 = 4k\Omega$ de modo que para la tensión de batería de 14 V a la entrada tendremos el valor siguiente:

$$V_{+} = V_i = \frac{1}{4 + 1} 14 = 2.8V$$

Los valores para las resistencia para la señal de batería se establecen de forma que la señal de referencia sea menor que la señal de contacto habilitada. Esto supone que si la relación entre la tensión de contacto y entrada se ha establecido en 1/5 la relación a establecer para la señal de referencia debe de ser menor, por ejemplo de 1/8 con $R_6 = 1k\Omega$ y $R_5 = 7k\Omega$

$$V_{+} = V_i = \frac{1}{7 + 1} 14 = 1.75V$$

La respuesta del esquema propuesto es la siguiente:

$$Vo = \begin{cases} 0V & , si\ Vcontacto = 0 \\ 3.3V & , si\ Vcontacto = Vbatería \end{cases}$$

Si bien este esquema es correcto a nivel teórico es preciso realizar algunas modificaciones sobre el mismo para su funcionamiento real.

En un amplificador real la salida obtenida a la salida del operacional no se corresponde de forma directa con la tensión de alimentación Vcc del mismo si no que existe una diferencia de tensión entre ambas.

La tensión máxima a establecer a su salida de un amplificador operacional real depende no solo de las tensiones de alimentación, como se considera en el caso ideal, sino también por el rango de tensiones precisas para la polarización de los elementos de la etapa de salida. Esto da lugar a una limitación en la tensión máxima disponible a la salida del operacional denominada *tensión de saturación de salida*, que puede estar hasta 2V por debajo de la tensión de alimentación del operacional.

Al igual que para la adquisición de la tensión de batería, el amplificador operacional escogido ha sido el LM358P de Texas Instrument. [26].

Este operacional presenta una tensión máxima a su salida 2V por debajo de la tensión de alimentación Vcc. Es por ello que se debe modificar nuestro esquema y proceder a alimentar el integrado a una tensión de 5V.

De este modo se obtiene la siguiente respuesta a la salida del operacional:

$$Vo = \begin{cases} 0V & , si\ Vcontacto = 0 \\ 3V & , si\ Vcontacto = Vbatería \end{cases}$$

Dónde 3V continúa siendo un valor dentro del rango de “1” lógico de las entradas digitales de la plataforma de desarrollo KL25Z.

El esquemático a implementar quedaría por tanto como sigue:

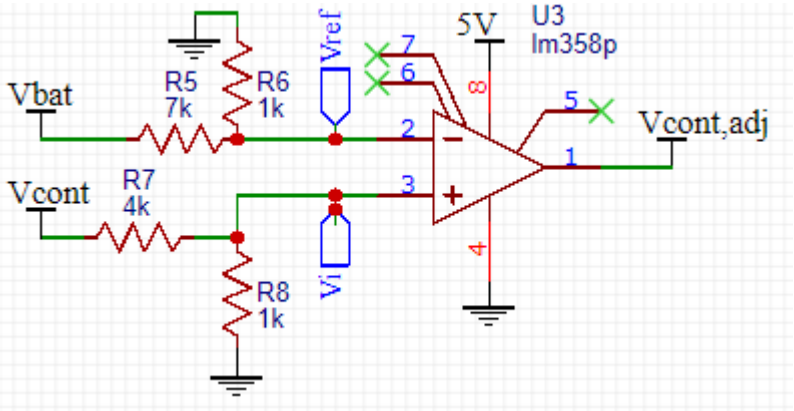


Ilustración 18 Esquema de comparador básico implementado para señal de contacto [24]

A continuación se muestran unas fotografías de la implementación del bloque de acondicionamiento de señal. Tal y como se puede ver en la imagen, para probar el contacto se ha utilizado un interruptor. Los modos de test de la plataforma serán explicados más adelante.

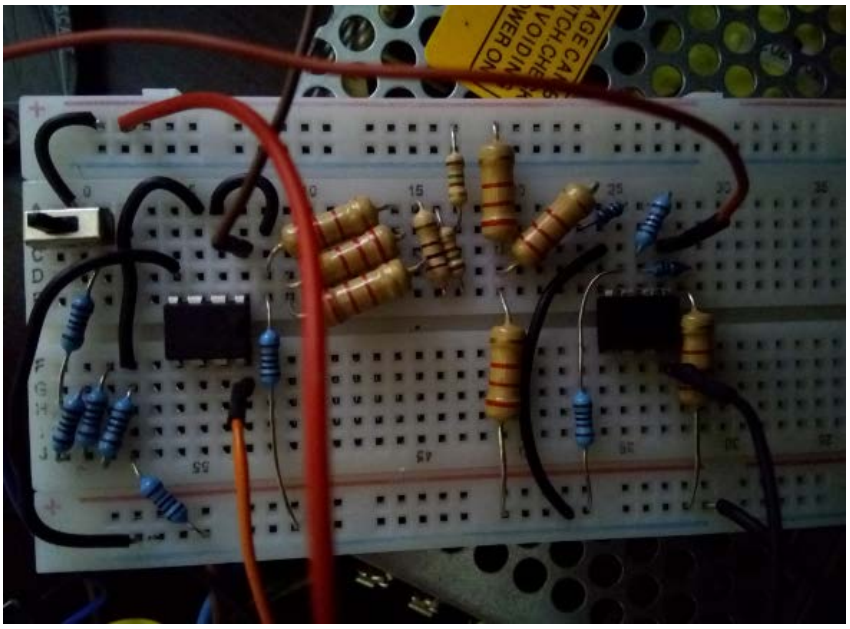


Ilustración 19 Fotografía de la implementación de la etapa de acondicionamiento de señales

3.1.3. Bloque de localización y comunicaciones

El bloque de localización y comunicaciones se compone principalmente por el módulo SIM808, un módulo que integra *Quad-Band GSM/GPRS*, GPS y *Bluetooth*.

Se trata de un módulo de bajo coste y diseño compacto que para la implementación ha sido adquirido en su PCB (*Printed Circuit Board*) de desarrollo SIM808 EVB-V3.2

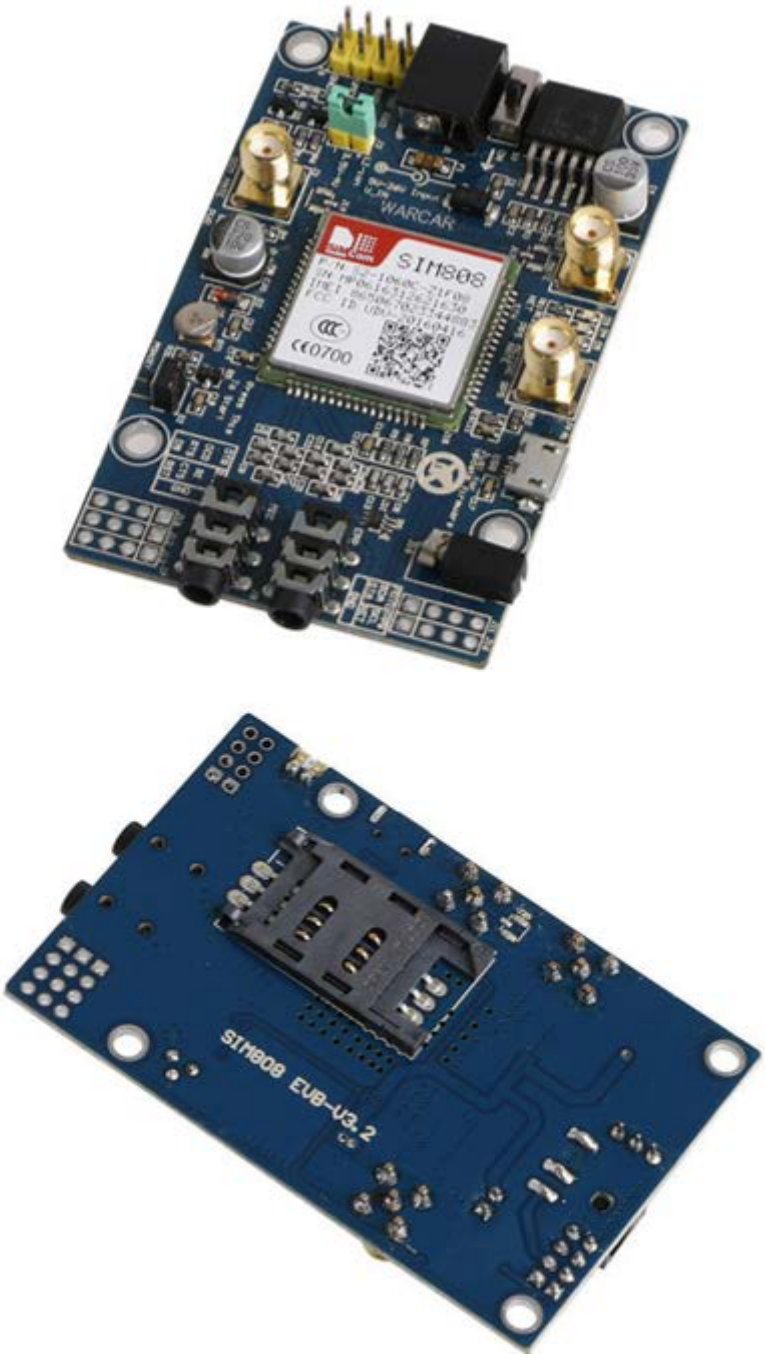


Ilustración 20 Fotografía SIM808 EVB-3.2

La decisión de utilizar un módulo de comunicaciones GPRS en lugar de un módulo con una tecnología más actual como 4G se ha tomado en base a un análisis coste-beneficio.

Dada la dimensión del proyecto se buscado un *shield* compacto que agrupase las funciones de comunicaciones y localización. Dentro de este ámbito el coste de un *shield* 4G era mucho más alto que el de un *shield* GPRS y no eran necesarias las mejoras que las nuevas tecnologías presentan frente al GPRS dado que la plataforma a desarrollar en funcionamiento enviaría una tasa muy baja de datos, de forma muy puntual

y los retrasos en los envíos de estos datos no eran un factor crítico en el funcionamiento de la misma.

Las principales características del *shield* son las siguientes:

- Alimentación principal: 5-26V
- Alimentación auxiliar: 3.5 – 4.2 V, disponible para baterías de litio
- Interfaces de debug: USB, TTL
- 1 Canal TTL puerto serie con posibilidad de conmutar entre las funciones GPS y GPRS
- Inferfaz SMA para antena GPS activa o pasiva
- Interfaz SMA para antena GSM
- Interfaz SMA para antena *Bluetooth*
- Soporte para comandos AT

En el sistema implementado el módulo recibe la alimentación de la etapa de alimentación, la cual es controlada por la plataforma de desarrollo mediante el uso de un circuito auxiliar de control. Además, este sistema debe gestionar la activación/desactivación de un *switch* de habilitación del módulo.

Por otra parte, la comunicación se establece a través del canal TTL puerto serie conmutable, de forma que el mismo puerto es utilizado para las funciones GPRS y GPS. El control se realiza mediante el envío de comandos AT desde el bloque de control.

Finalmente, se utilizan dos antenas SMA a las interfaces disponibles. La antena GPS utilizada es una antena activa con tensión de trabajo entre 3 y 5V y frecuencia 1575.42MHz.

La siguiente imagen muestra las interfaces utilizadas:

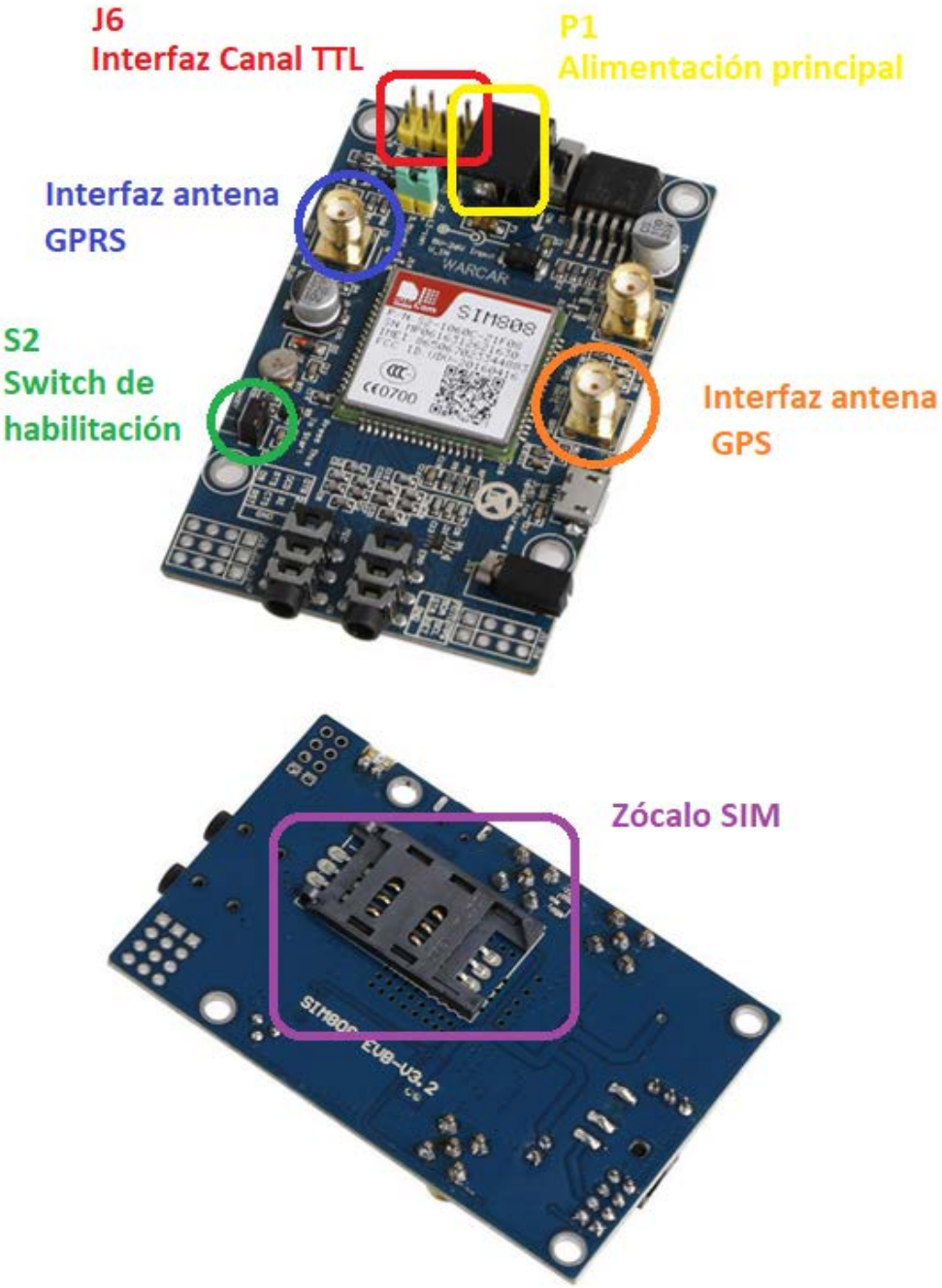


Ilustración 21 Fotografía SIM808 EVB-3.2 Interfaces en uso

El esquemático implementado es el siguiente:

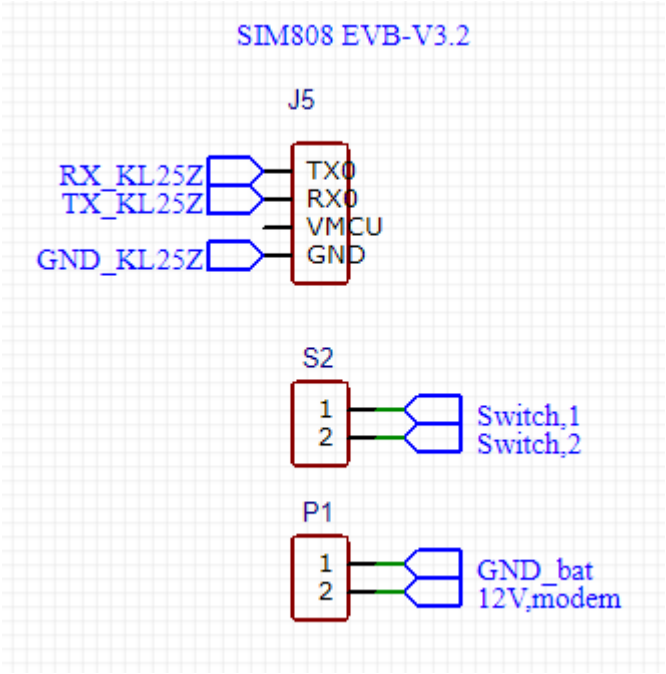


Ilustración 22 Esquema de conexiones del *shield* SIM808 evb-3.2

A continuación se muestra una fotografía del montaje del módulo.

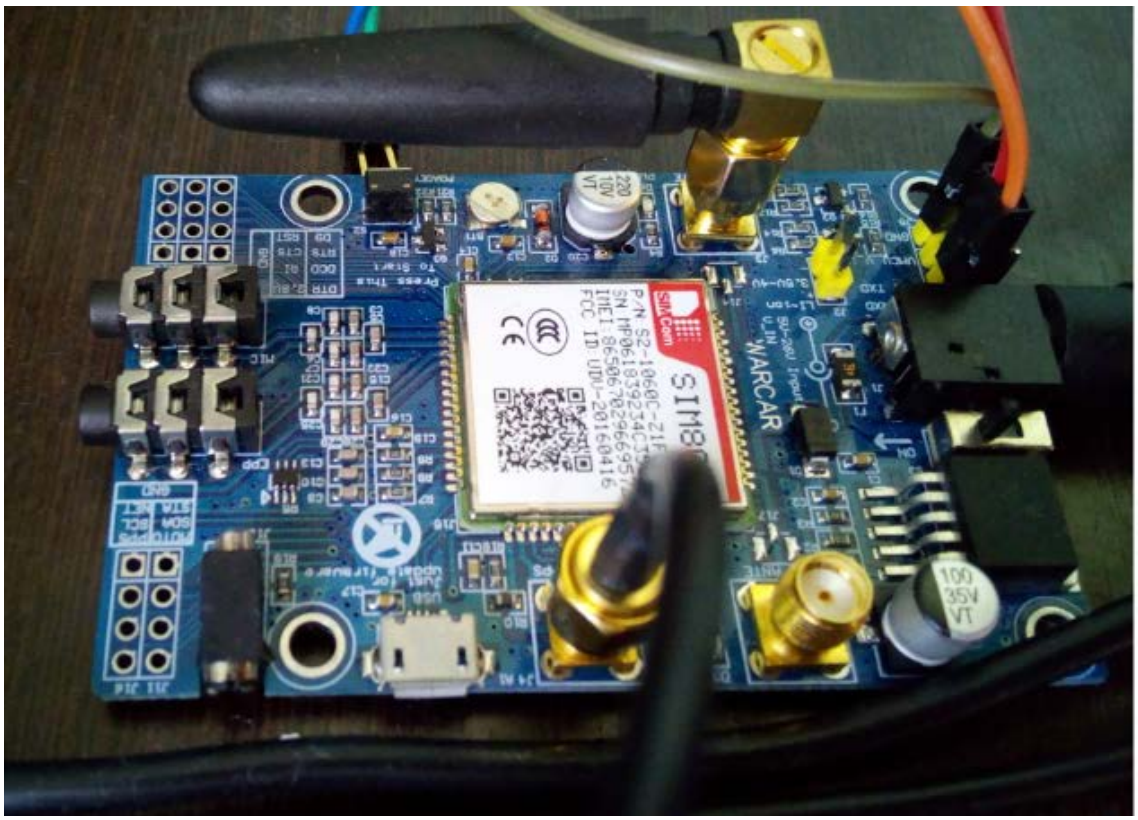


Ilustración 23 Fotografía del montaje del *shield* SIM808 EVB-3.2

3.1.4. Bloque de control de alimentación del módulo de comunicaciones

Dado el planteamiento del sistema, el módulo de control debe de ser capaz de gestionar todos los elementos que integran la plataforma embarcada, entre los cuales se encuentra el módulo de comunicaciones.

Dicha plataforma cuenta con una serie de salidas digitales de rango 0 a 3.3V que permiten establecer dicho control.

En el caso que nos ocupa el objetivo es que dicha plataforma gestione la alimentación del módulo de comunicaciones (mediante su activación/desactivación) y su habilitación/deshabilitación mediante la "pulsación" del *switch* de habilitación de dicho elemento.

Existen múltiples opciones para la implementación de este sistema tales como el diseño de interruptores mediante dispositivos MOSFEF o BJT o el uso de elementos de mayor potencia como relés.

Para este caso, y puesto que se trata de la gestión de una tensión de alimentación y la intensidad a suministrar es relevante, se han descartado los sistemas basados en MOSFET y BJT y se ha optado por el uso de un *shield* de relé, el KYES504-1G3V-2.

Este *shield* precisa de una alimentación de 3.3V y obedece a una señal lógica de la misma magnitud, de forma que puede ser controlado de forma directa por las salidas digitales de la plataforma de desarrollo KL25Z.

Por otra parte, es capaz de trabajar con un rango de tensiones de hasta 30V y 10A de corriente, suficiente para el sistema cuya máxima señal a manejar serán los 12V de alimentación con 2A máximos de consumo según los consumos facilitados por los fabricantes.

Ilustración 24 *Shield* Relé e Interfaces

El esquema implementado es el siguiente:

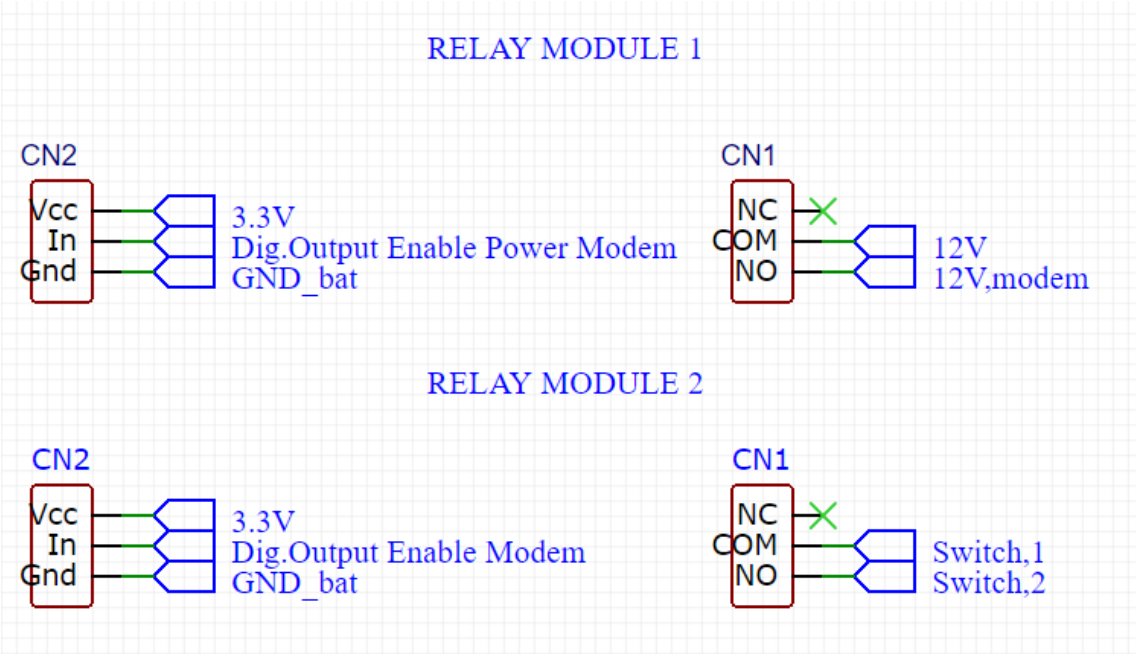


Ilustración 25 Esquema de conexiones del *shield* relés

A continuación se muestra una fotografía del montaje;

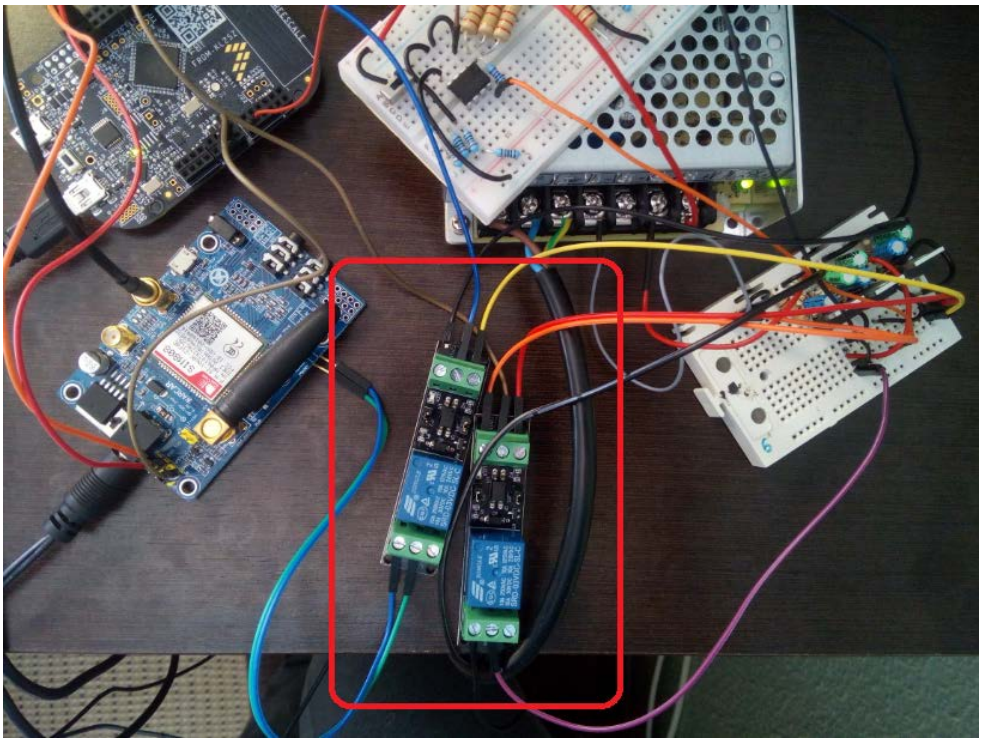


Ilustración 26 Fotografía del montaje de los *shield* de control tipo relé

3.1.5. Bloque de control

El bloque de control lo integra la plataforma de desarrollo Freescale KRDM-KL25Z, **cerebro** del sistema. Este elemento es el responsable de procesar las señales adquiridas por el sistema y gestionar las comunicaciones. [27]

Ilustración 27 Plataforma de desarrollo KL25Z *[27]*

La FRDM-KL25Z es una plataforma de desarrollo basada en un procesador ARM Cortex-M0+ de bajo coste con una frecuencia máxima de funcionamiento de 48MHz, memoria Flash de 128KB, controlador USB de máxima velocidad y numerosos periféricos analógicos y digitales.

El siguiente extracto del manual de usuario de la plataforma muestra las principales características del microcontrolador.

5.3 KL25Z Microcontroller

The target microcontroller of the FRDM-KL25Z is the KL25Z128VLK4, a Kinetis L series device in an 80 LQFP package. The KL25Z MCU features include:

- 32-bit ARM Cortex-M0+ core
  - up to 48 MHz operation
  - Single-cycle fast I/O access port
- Memories
  - 128 KB flash
  - 16 KB SRAM
- System integration
  - Power management and mode controllers
  - Low-leakage wakeup unit
  - Bit manipulation engine for read-modify-write peripheral operations
  - Direct memory access (DMA) controller
  - Computer operating properly (COP) Watchdog timer
- Clocks
  - Clock generation module with FLL and PLL for system and CPU clock generation
  - 4 MHz and 32 kHz internal reference clock
  - System oscillator supporting external crystal or resonator
  - Low-power 1kHz RC oscillator for RTC and COP watchdog
- Analog peripherals
  - 16-bit SAR ADC w/ DMA support
  - 12-bit DAC w/ DMA support
  - High speed comparator
- Communication peripherals
  - Two 8-bit Serial Peripheral Interfaces (SPI)
  - USB dual-role controller with built-in FS/LS transceiver
  - USB voltage regulator
  - Two I^2C modules
  - One low-power UART and two standard UART modules
- Timers
  - One 6-channel Timer/PWM module
  - Two 2-channel Timer/PWM modules
  - 2-channel Periodic Interrupt Timer (PIT)
  - Real time clock (RTC)
  - Low-power Timer (LPTMR)
  - System tick timer
- Human-Machine Interfaces (HMI)
  - General purpose input/output controller
  - Capacitive touch sense input interface hardware module

Ilustración 28 Extracto de las principales características del microcontrolador KL25Z *[27]*

Respecto al software de desarrollo, esta plataforma es compatibles con múltiples plataformas de desarrollo, tanto de NXP como de terceros.

Para la implementación bajo análisis se ha optado por el entorno de desarrollo online MBED, el cual es analizado más adelante.
Además, esta plataforma dispone de OpenSDA, estándar abierto para debug y programación.

Dicha plataforma presenta además un pinout compatible con Arduino R3, lo cual la convierte en una opción más atractiva para nuevos desarrollos dada la gran implantación de Arduino y la multitud de periféricos compatibles con dicha plataforma.

Dentro del entorno embarcado desarrollado la plataforma KL25Z implementa las siguientes funciones:

- Adquisición de las señales eléctricas del entorno: tensión de batería del vehículo y señal de contacto.

  La plataforma cuenta con un conjunto de GPIO (General Purpose Input/Output) que serán utilizadas para este propósito. Para el caso que nos ocupa las GPIO serán configuradas como entradas.

  Las entradas analógicas se encuentran provistas de un CAD (Convertidor Analógico Digital) de 16 bits y rango de entrada de 0 a 3.3V. Una de estas entradas analógicas, en concreto A0, será utilizada para realizar la monitorización de la tensión en los bornes de la batería del vehículo tras adaptarse su rango mediante el bloque acondicionador de señal descrito en el apartado 3.1.2.1 de esta memoria.

  Las entradas digitales interpretan las señales analógicas de 0 y 3.3V como los valores lógicos “0” y “1” respectivamente, con unos umbrales de aceptación. Una de estas entradas digitales, en concreto D2, será utilizada para realizar la monitorización del estado de la señal de contacto del vehículo previa adaptación de su rango de tensiones mediante el bloque acondicionador de señal descrito en el apartado 3.1.2.2 de esta memoria.

- Adquisición de la posición del vehículo a través de su interfaz RS232.

  La plataforma de desarrollo KL25Z establece una comunicación mediante su interfaz RS232 basada en comando AT con el módulo de localización y comunicaciones implementado mediante la placa de desarrollo SIM808 EVB-3.2 descrita en el apartado 3.1.3 de esta memoria. Es a través de dicho periférico que obtendrá la posición GPS determinada por el módulo SIM808 de dicho *shield*.

- Procesado de las señales adquiridas.

  La plataforma de desarrollo KL25Z es la responsable del procesado de los datos adquiridos por la misma, siendo estos: tensión en las bornes de la batería del vehículo, estado de la señal de contacto y posición del vehículo.

  En función de los datos medidos la plataforma obtendrá los siguientes datos derivados: tiempo de encendido del vehículo y tiempo de conducción del vehículo.

  El tiempo de encendido del vehículo se establece como el tiempo durante el cual la señal de contacto ha estado activa. Mientras que el tiempo de circulación se establece como el tiempo en que la señal de contacto ha estado activa y además la posición del

vehículo ha variado desde el momento de la habilitación del contacto hasta el momento de la deshabilitación del mismo.

- Generación de alarmas.

  En base a la información obtenida por el procesado la plataforma KL25Z será responsable de la generación de alarmas para el mantenimiento.

  Estas alarmas se generarán cuando se supera un umbral establecido para los distintos valores de interés para dicho mantenimiento: tensión en los bornes de la batería del vehículo, tiempo de encendido del vehículo y tiempo de circulación del vehículo.

  El aviso de alarma al usuario final se establece mediante el envío de un correo electrónico a través del servidor gratuito SMTP2GO, analizado en mayor profundidad en el apartado 3.2 de esta memoria.

  Para establecer la comunicación con dicho servidor la plataforma KL25Z hace uso de la interfaz GPRS del módulo de localización y comunicaciones implementado mediante la placa de desarrollo SIM808 EVB-3.2 descrita en el apartado 3.1.3 de esta memoria

  La plataforma de desarrollo KL25Z establece una comunicación mediante su interfaz RS232 basada en comando AT con el módulo de localización y comunicaciones para establecer la comunicación con el servidor de correo SMTP2GO en enviar correos electrónicos.

  Se generaran alarmas en las siguientes situaciones:

  - Tensión de batería inferior a un umbral establecido.

    Una tensión baja en los bornes de la batería de un vehículo es un indicativo de un estado degradado de la misma. La realización de un mantenimiento preventivo (o correctivo si fuera necesario) en el momento de su detección es clave para evitar una mayor degradación y limitar la aparición de situaciones problemáticas tales como que en un momento dado la batería no sea capaz de arrancar el vehículo.

    En base a los valores mencionados en la bibliografía se ha establecido un umbral para la generación de dicha alarma de 11.9V.

    En el momento en el que el sistema monitorice una tensión en bornes inferior a dicho umbral el usuario final será informado mediante el envío de un correo electrónico como

el mostrado en la siguiente captura realizada durante las pruebas del sistema.

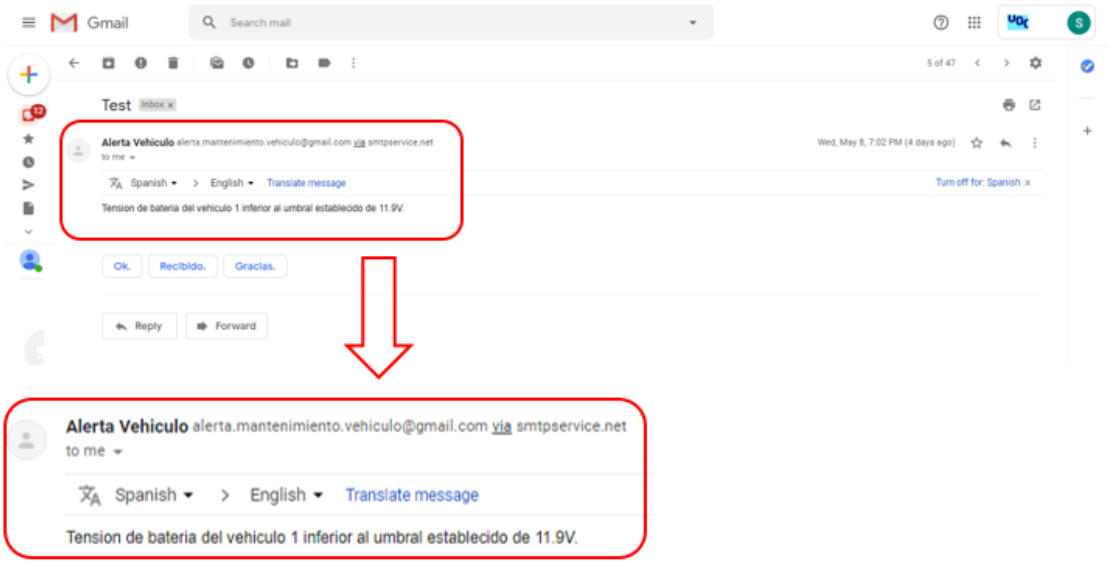


Ilustración 29 Correo electrónico con alarma de tensión de batería inferior al umbral

- Tiempo de conducción del vehículo superior a un umbral establecido.

  Con motivo de la normal degradación del vehículo los expertos recomiendan la realización de una revisión del mismo cada 150000 km. Es a partir de esta recomendación que se establece el tiempo umbral de conducción superado el cual el sistema debe de indicar que es necesario revisar el vehículo.

  Este tiempo deberá ser configurado en función del objeto de la flota en cuestión y del tipo de trayectos que realice.

  A modo de ejemplo se ha supuesto una flota orientada a largos trayectos por carretera y se ha asumido una velocidad media de conducción de 80km/h. En este caso se debería por tanto configurar un tiempo umbral de aviso de 1875 horas, superadas las cuales el sistema debe enviar una alarma.

$$Umbral, alarma\ conducción\ [horas] = \frac{150000\ [km]}{80\ [km/h]} = 1875\ [horas]$$

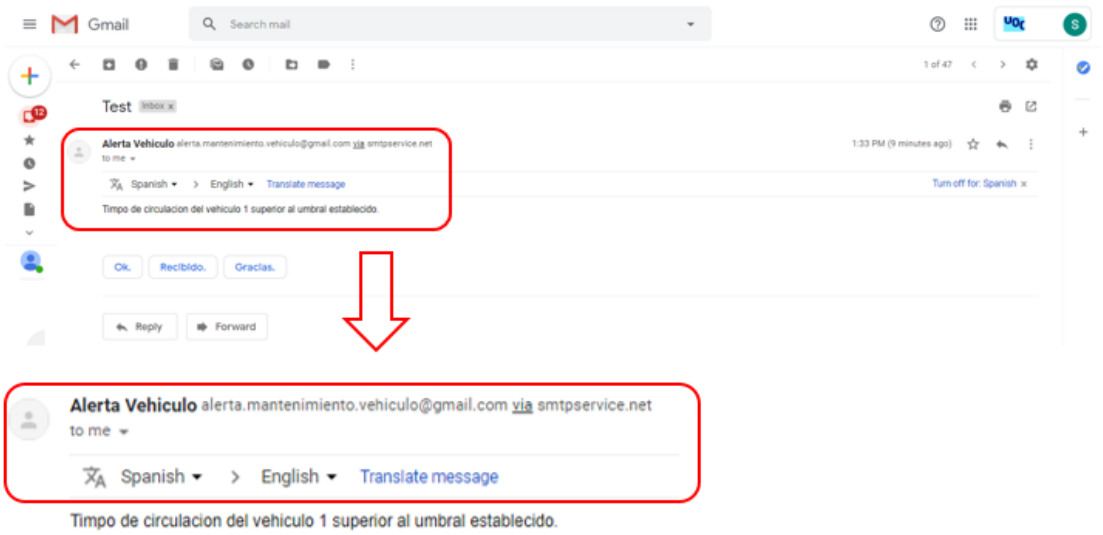


Ilustración 30 Correo electrónico con alarma de tiempo de circulación superior al umbral

- Tiempo de encendido del vehículo superior a un umbral establecido

Si bien no es habitual en vehículos particulares que el tiempo de encendido del vehículo difiera en grandes números del tiempo de circulación, tal y como se define en este documento. Se ha optado por establecer una alarma asociada como *back-up* de seguridad en términos de mantenimiento.

Un ejemplo real que han llevado a considerar el establecimiento de esta alarma serían las flotas de vehículos urbanos de reparto. Resulta habitual ver cómo los camiones de reparto permanecen encendidos mientras descargan (en muchos casos es necesario para poder utilizar el elevador trasero). Estos tiempos pueden superar con creces el tiempo de circulación de dichos vehículos y suponen también una degradación de los mismos.

Por otra parte, esta alarma se considera también un *back-up* de seguridad. Teniendo en cuenta que la plataforma embarcada desarrollada no realiza un seguimiento de la localización de los vehículos de la flota podría ocurrir que debido a un fallo en el GPS de la plataforma de un vehículo no se monitorizasen de forma correcta los tiempos de circulación y no se generase la alarma correspondiente al tiempo de circulación. Esta alarma auxiliar solventaría esta situación de riesgo un tiempo más tarde forzando a una revisión del vehículo con la consecuente reparación del sistema de monitorización.

A la vista de esto se ha considerado que el tiempo umbral de encendido del vehículo debe establecerse en base al umbral de tiempo de circulación. En concreto se ha considerado que el umbral debe ser un 25% superior al umbral de circulación.

Continuando con el ejemplo del punto anterior, para una flota interurbana estableceríamos un umbral de 2343.75 horas, superado el cual se generaría la alarma correspondiente.

$$Umbral, alarma\ encendido\ [horas] = 1875\ [horas]\ x\ 1.25 = 2343.75\ [horas]$$

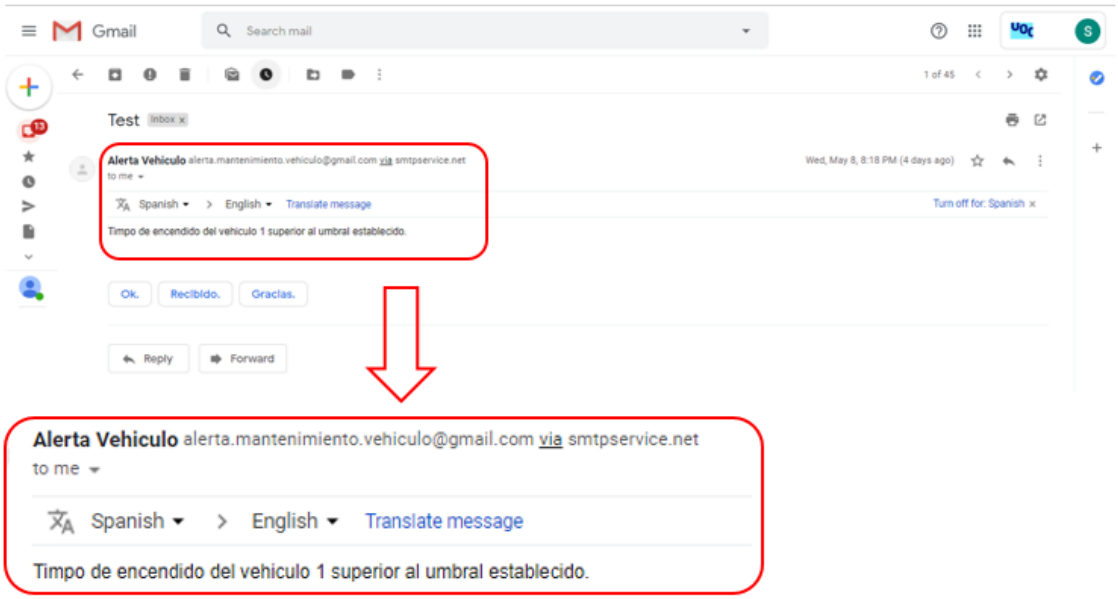


Ilustración 31 Correo electrónico con alarma de tiempo de encendido superior al umbral

La generación de alarmas es clave en el funcionamiento de la plataforma embarcada, pues su objeto es advertir al usuario final de la necesidad de la realización de un mantenimiento preventivo en el vehículo, siendo este el **principal objetivo** de este sistema.

- Gestión del almacenamiento y visualización de los datos.

  Una de las funcionalidades de esta plataforma es el almacenamiento de los datos para su visualización por parte de usuario final.

La plataforma escogida para el almacenamiento de estos datos es la plataforma IoT abierta Thingspeak que ofrece, además de capacidad de almacenamiento y una API compatible con Matlab para envío de datos, un entorno de visualización amigable al usuario, dónde nuestro usuario final visualizará los datos almacenados tal y como vemos en la siguiente ilustración.

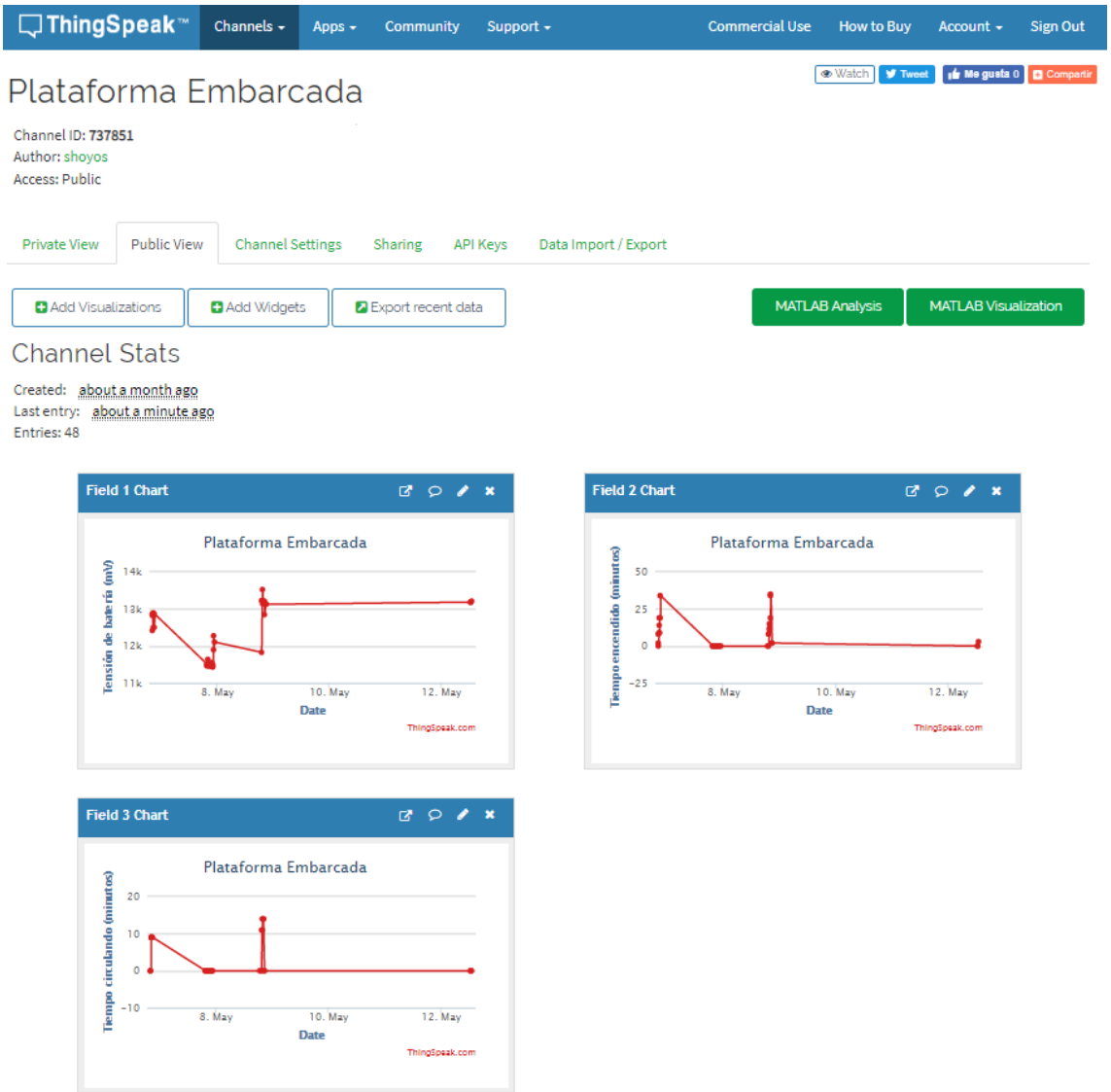


Ilustración 32 Entorno de visualización de los datos de la plataforma embarcada en Thingspeak

La plataforma KL25Z será responsable del envío de los datos procesados a Thingspeak. Los datos almacenados son aquellos responsables de la generación de las alarmas: tensión de batería, tiempo de circulación del vehículo y tiempo de encendido.

Para el envío de los datos la plataforma utilizará de nuevo el módulo auxiliar de comunicaciones SIM808 y su conexión GPRS.

La plataforma de desarrollo KL25Z establece una comunicación mediante su interfaz RS232 basada en comando AT con el módulo de localización

y comunicaciones para establecer la comunicación con el servidor de la plataforma Thingspeak y transmitirle los datos a almacenar.

Estos datos se actualizarán de forma periódica en el servidor al menos una vez cada 24 horas, además, se actualizarán de inmediato ante la ocurrencia de cualquiera de los siguientes eventos:

- Deshabilitación de la señal de contacto
- Generación de alarmas

Un análisis más en profundidad del funcionamiento de la plataforma Thingspeak se realizará en el siguiente capítulo.

Por último indicar que si bien para las pruebas dicho elemento ha recibido alimentación vía USB, en el entorno real la recibiría de una pila de litio a través de su interfaz específica para este fin.

Una vez descrita la interconexión de los distintos elementos hardware que componen la plataforma embarcada podemos ver su implementación real en el esquemático completo disponible en esta memoria en el Anexo II: Esquemático de la plataforma embarcada.

Cabe indicar que los elementos escogidos ofrecen unas temperaturas de funcionamiento con un límite superior mayor de 50ºC, siendo por tanto adecuados para las altas temperaturas que pueden alcanzarse en épocas de calor en el interior de un vehículo. La siguiente tabla indica las temperaturas de operación de los principales elementos ordenados en orden de aparición.

AQUI

Elemento	Temperatura de operación
Regulador lineal LM350T[1]	0 a 125ºC
Amplificador operacional LM358	0 a 70ºC
Módulo SIM808	-40 a 85ºC
Relé SRD-DC03V-SL-C	-25°C a +70°C
Plataforma KL25Z	-40 a 105ºC

Tabla 1 Temperatura de operación de los principales elementos hardware

[1] El regulador LM350T cuenta además con protección contra sobretemperatura.

3.2. Análisis de componentes software

Tal y como se adelantaba en el capítulo anterior, para el desarrollo del sistema ha sido precisa la utilización de una serie de componentes software: servidor de correo, entorno de desarrollo, plataforma de almacenamiento y visualización de datos…

En este capítulo se razonan los motivos por los cuales se ha optado por el uso de cada una de estas plataformas y se analiza más en detalle su uso.

Finalmente, se ofrece una visión global del funcionamiento de la plataforma en base al diagrama de flujo del código de funcionamiento, incluido en el Anexo I: Código desarrollado para plataforma KRDM-KL25Z.

3.2.1. Plataformas software auxiliares

En este apartado se aborda la justificación de la elección de cada una de las plataformas y se analiza en mayor profundidad su modo de funcionamiento e iteración con el resto de elementos que integran el sistema desarrollado.

3.2.1.1. Servidor de correo electrónico: SMTP2GO

Una de las principales funciones del sistema desarrollado es la generación de alarmas ante la detección de situaciones anómalas que causen que se requiera una revisión del vehículo de la flota bajo monitorización.

En el momento del desarrollo del sistema bajo estudio se consideraron dos modos para enviar al usuario final la información de las alertas generadas: envío de SMS o envío de correos electrónicos.

Con objeto de determinar el modo a implementar se realiza un análisis de ventajas e inconvenientes de cada una de las soluciones. La siguiente tabla refleja los pros y contras de cada elección.

Ventajas	
Envío de alertas mediante SMS	Envío de alertas mediante correo electrónico
✓ Recepción de alertas en un dispositivo portátil	✓ Consulta de alertas desde múltiples terminales (móvil, PC, Tablet …) ✓ Posibilidad de configurar la recepción de alertas de forma simultánea en múltiples terminales ✓ Posibilidad de usar un gestor de correo para organizar los correos de alerta, facilitando su gestión
Inconvenientes	
Envío de alertas mediante SMS	Envío de alertas mediante correo electrónico
✗ Recepción de las alertas en un único terminal ✗ Recepción de alertas sujeta a la disponibilidad de cobertura móvil	✗ Recepción de las alertas sujeta a la disponibilidad de acceso a internet

Tabla 2 Comparativa recepción de alertas vía SMS vs Correo Electrónico

A la vista del análisis realizado se concluye que es más ventajosa la implementación de la notificación de alertas mediante el envío de correos electrónicos.

Entre sus ventajas frente al uso del SMS cabe destacar que las alertas pueden visualizarse desde más de un terminal, lo que posibilita que puedan gestionarse entre varias personas en paralelo.

Por otra parte, su única limitación vendría dada por precisar de acceso a internet. Asumiendo como práctica habitual que este tipo de actividades de gestión se realizan desde un entorno fijo (una oficina) es muy improbable que se limite el acceso a internet. Además, gracias a su acceso multiplataforma en caso del fallo de la conexión a internet por cable siempre podrían consultarse dichas alertas vía red móvil.

Una vez definido el modo de envío de alarmas procedemos a analizar su modo de implementación.

El servidor SMTP escogido para el envío de correos electrónicos es el servidor gratuito SMTP2GO[2]. Este servidor presenta una interfaz web de configuración sencilla y sus correos son admitidos por servidores de correos de uso ampliamente extendido como Gmail.

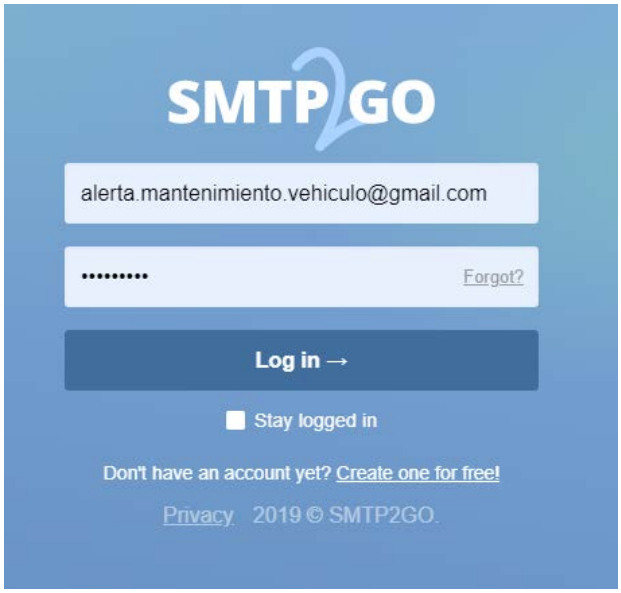


Ilustración 33 Interfaz de login SMTP2GO

Las siguientes imágenes muestran la configuración establecida en la cuenta SMTP2GO para el envío de correos desde una cuenta de Gmail, “alerta.mantenimiento.vehiculo@gmail.com”, configurada como usuario, a otra cuenta de Gmail, que en este caso es la cuenta de usuario de la UOC “sha@uoc.edu”.

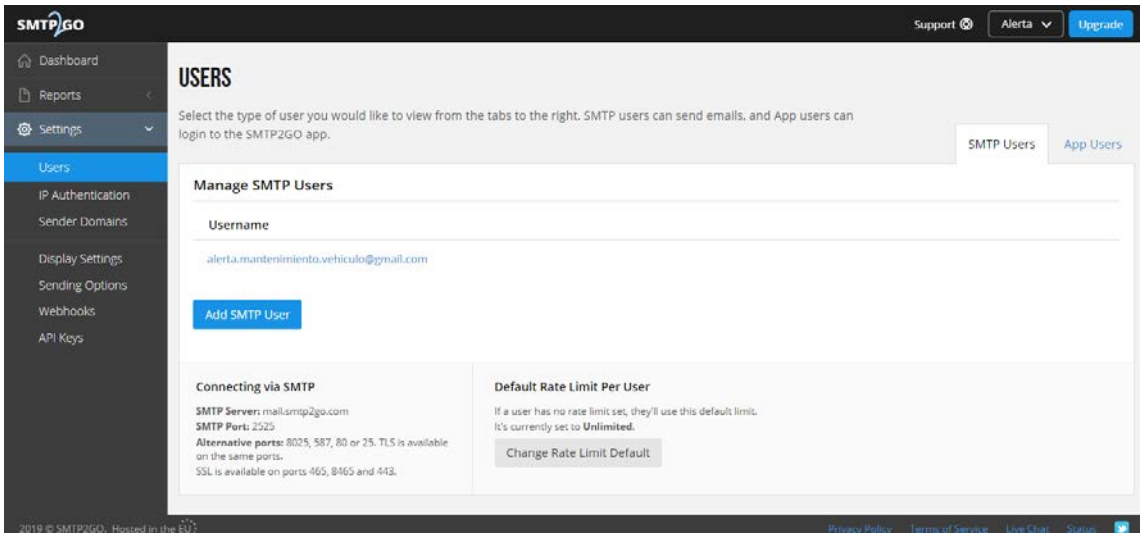


Ilustración 34 Configuración Usuarios SMTP2GO: SMTP Users

[2] El uso gratuito de SMTP2GO queda restringido a un número máximo de correos diarios. Para uso masivo se precisa de una cuenta de pago.

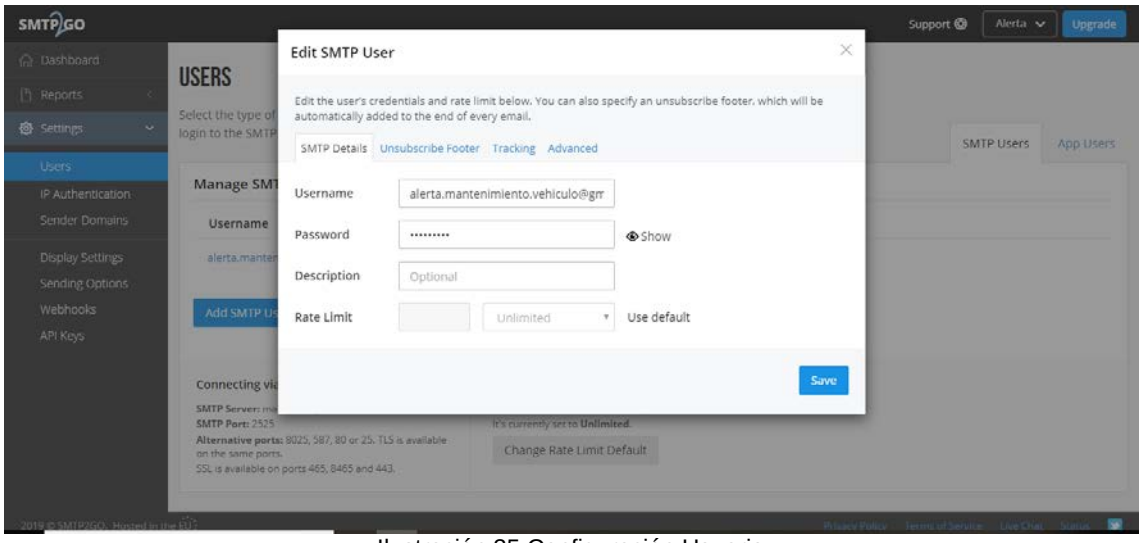


Ilustración 35 Configuración Usuario

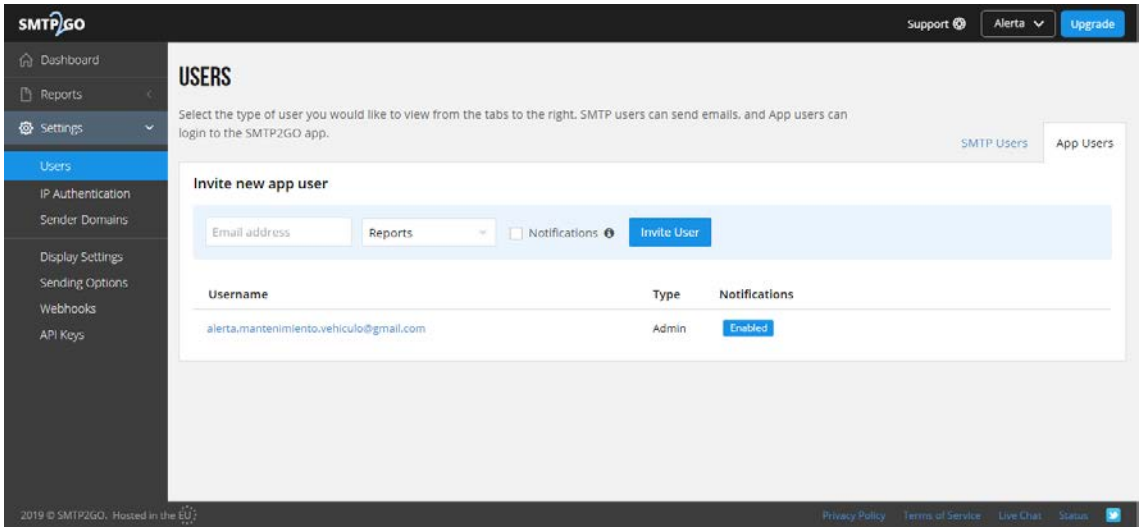


Ilustración 36 Configuración Usuarios SMTP2GO: App Users

Tal y como se observa en la *Ilustración 34*, la conexión vía SMTP al servidor mail.smtp2go.com puede realizarse a través de varios puertos. Para el caso que nos ocupa se utiliza el puerto por defecto 2525.

La conexión con el servidor SMTP se establece mediante TCP a través del módulo de comunicaciones SIM808. Este módulo es a su vez controlado por la plataforma de desarrollo KL25Z mediante el envío de comando AT a través de un puerto serie. La definición de los comandos utilizados viene dada por los documentos de desarrollo facilitados por el módulo de comunicaciones y sus notas de aplicación. [28] [29]

Las siguientes imágenes muestran a modo descriptivo la secuencia de comandos AT necesarios para el envío de un correo electrónico desde el módulo SIM808, mientras que la tabla describe el significado de cada uno de ellos. Para esta prueba, previa al volcado de los

comandos en el código definitivo de la plataforma, se usó un conversor de niveles TTL a USB para la conexión directa del *shield* de comunicaciones al ordenador.

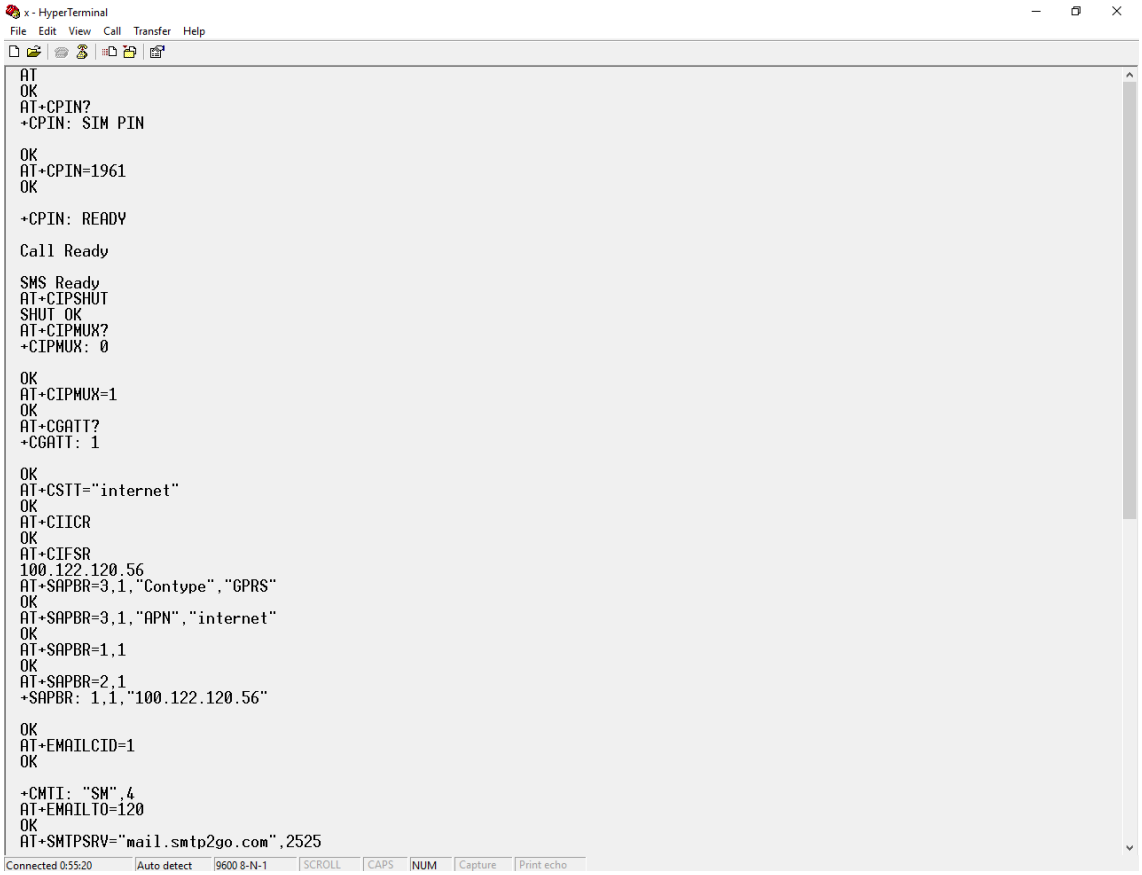


Ilustración 37 Secuencia de comandos AT para envío de correo electrónico vía SMTP2GO – Parte 1 de 2

Ilustración 38 Secuencia de comandos AT para envío de correo electrónico vía SMTP2GO – Parte 2 de 2

Comando	Descripción
AT OK	Comprobación del estado del módulo. El módulo responde con un “OK”.
AT+CPIN? +CPIN: SIM PIN OK	Consulta al módulo acerca de su estado al respecto del PIN. El módulo envía como respuesta una cadena de texto con información acerca de si está esperando o no una contraseña y el tipo de contraseña esperada (código pin, puk …). En este caso el módulo responde indicando que está a la espera de un código PIN.
AT+CPIN=1961 OK +CPIN: READY Call Ready SMS Ready	Comando para introducción del código PIN de la SIM. Puesto que es correcto el módulo responde actualizando su estado a “READY”, indicando que ya no está pendiente de recibir ninguna contraseña. Informa también de que está habilitado.
AT+CIPSHUT SHUT OK	Comando para cerrar el contexto PDP del GPRS.
AT+CIPMUX? +CIPMUX: 0 OK	Comando de consulta del tipo de conexión IP configurada. El módulo responde con un “0” indicando que está configurada una conexión tipo *single*.
AT+CIPMUX=1 OK	Comando de configuración del tipo de conexión IP. Se establece la configuración el 1 para permitir con conexión IP tipo múltiple.
AT+CGATT? +CIPMUX: 1 OK	Comando para consultar al módulo si la conexión a la red GPRS está establecida. Responde con un “1” lo que indica que está establecida. Si devolviera un “0” se ordenaría la conexión mediante el comando AT+CGATT=1.
AT+CSTT=”internet” OK	Comando para definición del APN (Access Point Number) de la red.
AT+CIICR OK	Comando para activación del perfil de datos inalámbrico.
AT+CIFSR 100.122.120.56	Comando para consulta de la IP del módulo. Nos permite verificar que el módulo dispone ya de IP.
AT+SAPBR=3,1,"Contype","GPRS" OK	Comando de configuración de la portadora de la conexión IP. Los parámetros indicados activan la conexión GPRS.
AT+SAPBR=3,1,"APN","internet" OK	Comando de configuración de la portadora de la conexión IP. Los parámetros indicados establecen el APN utilizado: “internet”.

AT+SAPBR=1,1 OK	Comando de apertura de portadora.
AT+SAPBR=2,1 +SAPBR: 1,1,” 100.122.120.56” OK	Comando de consulta de la portadora. Si la conexión se ha establecido con éxito responde con la IP.
AT+EMAILCID=1 OK +CMTI: “SM”,4	Comando de configuración para el envío de un email. Establece el identificador de perfil de portadora.
AT+EMAILTO=120 OK	Comando de configuración para el envío de un email. Establece el *timeout* de espera de respuesta del servidor de correo en 120 segundos.
AT+SMTPSRV="mail.smtp2go.com",2525 OK	Comando de configuración de la dirección y el puerto del servidor SMTP a utilizar para el envío del correo.
AT+SMTPAUTH=1,"alerta.mantenimiento.vehiculo@gmail.com","alerta123" OK	Comando de configuración del usuario y contraseña a utilizar para la autenticación en el servidor SMTP.
AT+SMTPFROM="alerta.mantenimiento.vehiculo@gmail.com","Alerta Vehiculo" OK	Comando de configuración de la dirección y nombre del originador del mensaje.
AT+SMTPRCPT=0,0,"sha@uoc.edu","Sara" OK	Comando de configuración de la dirección y nombre del destinatario del mensaje.
AT+SMTPSUB="Test" OK	Comando de configuración de título del mensaje de correo.
AT+SMTPBODY=34 DOWNLOAD …. OK +SMTPSEND: 1	Comando de configuración de la longitud del mensaje a enviar. Tras esto el módulo queda a la espera. Una vez recibe la cadena de la longitud recibida responde con un “OK”. Tras enviar el correo electrónico responde con la cadena “SMTPSEND:1”.

Tabla 3 Secuencia de comandos AT para envío de correo electrónico vía SMTP2GO

Las siguientes imágenes muestran a modo de ejemplo el registro en el servidor del envío de un correo y el correo recibido.

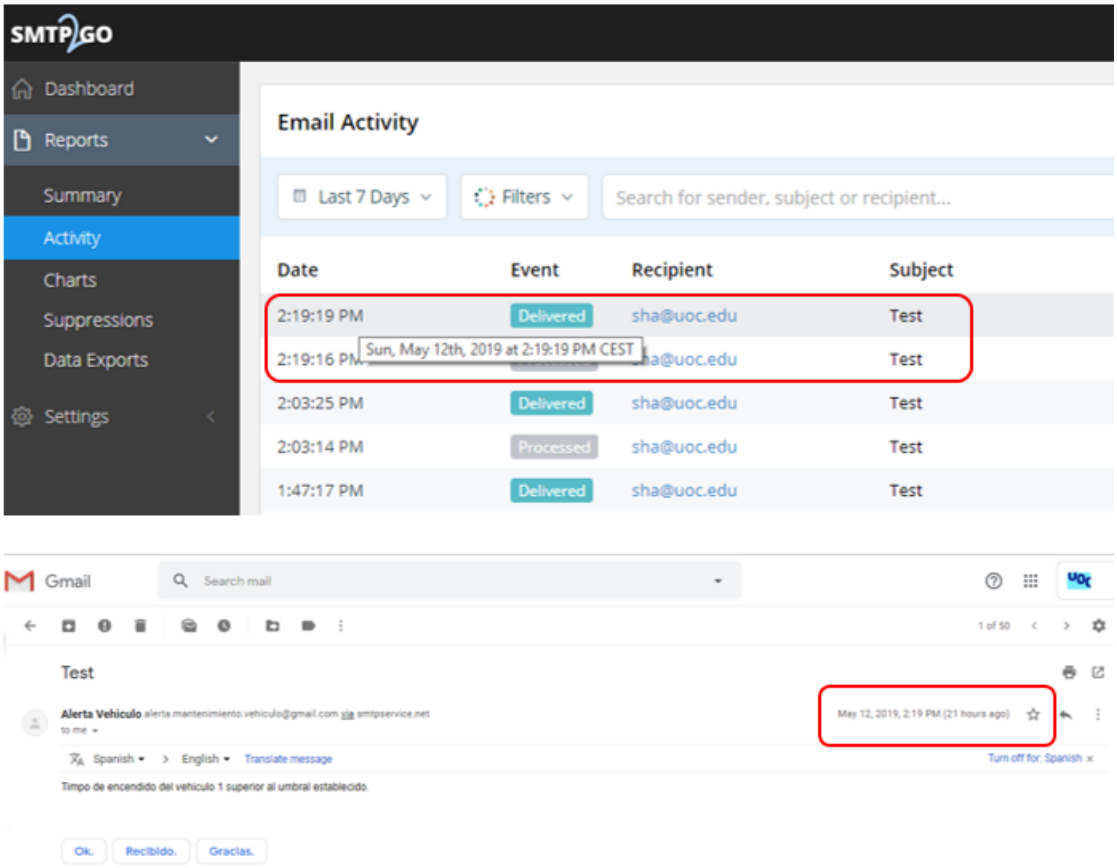


Ilustración 39 Registro de procesado y entrega de correo electrónico en SMTP2GO y correo asociado recibido en cuenta Gmail

3.2.1.2. Plataforma de almacenamiento y visualización: Thingspeak

La segunda función del sistema de control de fatiga es el almacenamiento de las variables monitorizadas, en base a las cual se general las alarmas. Además, estos datos deben de ser accesibles al usuario final de forma que pueda visualizarlos y extraerlo para su procesado, en caso de que así lo desee.

La plataforma escogida para el almacenamiento de dichos datos es Thingspeak. [30]

Thingspeak es una plataforma orientada al IoT que permite al usuario almacenar datos monitorizados en la nube, visualizarlos y extraerlos.

Una de las virtudes de dicha plataforma es su compatibilidad con MATLAB, software popular de programación y análisis matemático de

gran implantación y capacidad, lo que hace de Thingspeak una opción muy atractiva para proyectos de distinta orientación y magnitud.

Las capacidades básicas de la plataforma con actualizaciones del canal separadas un mínimo de 15 segundos son de uso gratuito mediante el uso de una cuenta de MathWorks, suficiente para esta plataforma que no precisa de capacidades de procesado en nube adicionales.

La estructura de Thingspeaks se basa en canales. El canal es la unidad de almacenamiento de datos. Cada canal puede tener uno o más campos en los que almacenar los distintos parámetros a guardar en el canal.

Dentro de cada canal es posible añadir visualizaciones, así como importa y exportar datos bien del canal completo bien de un parámetro en particular.

Los canales pueden ser públicos o privados, de forma que puedan o no visualizarse por usuarios externos.

Por otra parte cada canal cuenta con unas credenciales (API Keys) que permiten la iteración de las APIs de la plataforma para el acceso y gestión de los datos desde el dispositivo externo que la utilice.

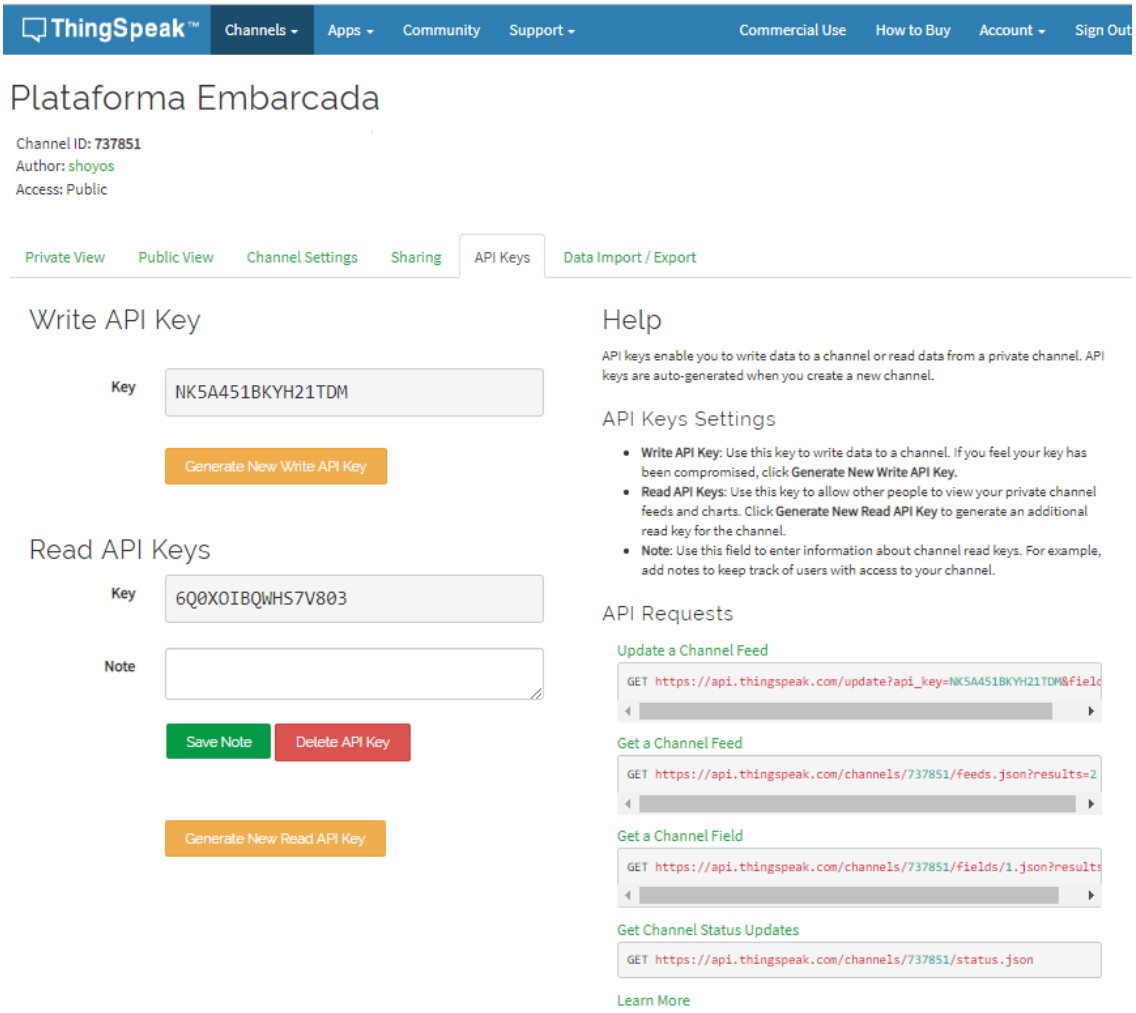


Ilustración 40 Información API de acceso del canal Plataforma Embarcada de Thingspeak

Para el desarrollo de esta plataforma se ha generado un canal público con 3 campos:

- Tensión de batería en mV.
- Tiempo de encendido en minutos.
- Tiempo de circulación en minutos.

Este canal es accesible a través del siguiente enlace: https://thingspeak.com/channels/737851

La siguiente imagen muestra la vista que obtendría un usuario externo al acceder al canal.

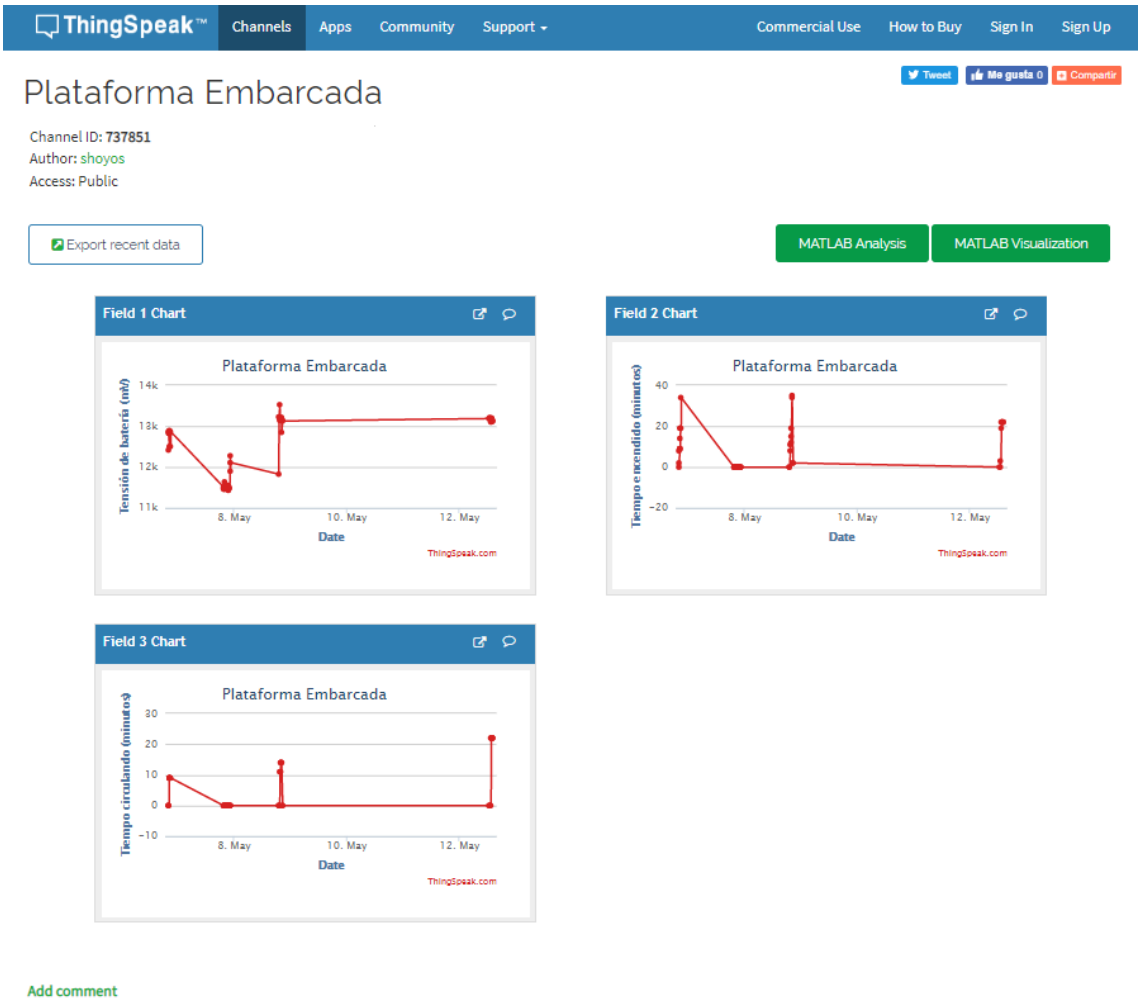


Ilustración 41 Vista pública del canal Plataforma Embarcada de Thingspeak

Las variables de tiempo actualmente definidas como minutos vienen dadas por las pruebas funcionales ejecutadas sobre el sistema, descritas en el capítulo 4.

Si bien en el caso real esta variable se establecería como horas para testear el sistema se ha establecido en minutos con objeto de poder probar los distintos modos de funcionamiento en un tiempo acotado Nótese que de usarse los tiempos reales las actualizaciones periódicas se realizarían cada 24 horas, y para que, por ejemplo, se generase la alarma de tiempo de conducción superado serían necesarias más de 1000 horas de funcionamiento del sistema.

De cara a la implementación real se dispondría de un canal por cada vehículo de la flota.

Tal y como se indicaba anteriormente, es posible exportar los datos del canal completo o de alguno de sus campos mediante la opción *Export Recent Data* visible en la *Ilustración 41*. Esta exportación puede realizarse en distintos formatos: JSON, XML o CSV.

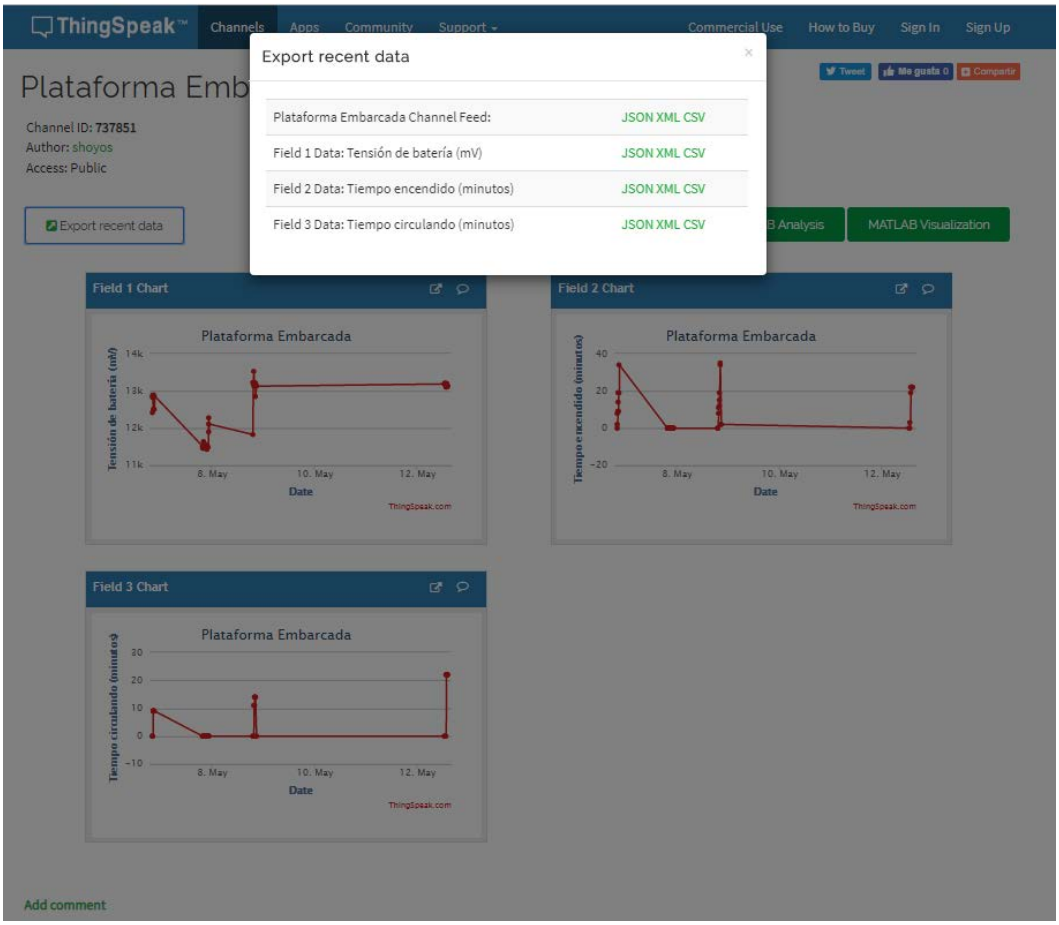


Ilustración 42 Exportación de datos del canal Plataforma Embarcada de Thingspeak

La conexión con el servidor de Thingspeak se establece mediante TCP a través del módulo de comunicaciones SIM808. Este módulo es a su vez controlado por la plataforma de desarrollo KL25Z mediante el envío de comando AT a través de un puerto serie. La definición de los comandos utilizados viene dada por los documentos de desarrollo facilitados por el módulo de comunicaciones y la información disponible de la API de Thingspeak. [28] [30]

A continuación se indican los comandos AT específicos utilizados para dicha comunicación, implementados en el código desarrollado en Mbed disponible en el Anexo I: Código desarrollado para plataforma KRDM-KL25Z. La siguiente tabla describe el significado de cada uno de ellos.

```
AT+CIPSTART="TCP","184.106.153.149","80"
Respuesta esperada
OK
CONNECT OK

AT+CIPSEND= longitud de cadena + 4

GET https://api.thingspeak.com/update?api_key=NK5A451BKYH21TDM&field1=dato1&field2=dato2&field3=dato3
```

Ilustración 43 Secuencia de comandos AT para envío de datos a Thingspeak previa configuración del módem

Comando	Descripción
AT+CIPSTART=”TCP”,”184.106.153.149”,”80” OK	Comando para establecimiento de una conexión TCP a la dirección y puerto suministrados como parámetro, siendo estos la IP ser servidor de Thingspeak y el puerto en el cual escucha.
AT+CIPSEND= longitud + 4	Comando que indica la longitud de la cadena de datos a enviar a través de la conexión TCP. La longitud debe de ser la de la cadena a enviar más 4, dónde estos cuatro incluyen el espacio y el retorno de carro (\r y \n).
GET https://api.thingspeak.com/update?api_key=NK5A451BKYH21TDM&field1=dato1&field2=dato2&field3=dato3	Comando GET de actualización de Thingspeak según el formato definido por la plataforma. Incluye la *api_key* asociada al canal a escribir, y los distintos campos de dicho canal, en nuestro caso 3.

Tabla 4 Secuencia de comandos AT para envío de datos a Thingspeak previa configuración del módem

3.2.1.3. Entorno de desarrollo: Mbed

Tal y como se indicó en apartados anteriores, la plataforma de desarrollo KL25Z resulta compatible con múltiples plataformas de desarrollo software tanto de NXP como de otros fabricantes.

Dos plataformas gratuitas ampliamente utilizadas son: KEIL y Mbed.

Ambas de acceso gratuito se orientan a distintos sectores.

KEIL es un entorno de desarrollo complejo que requiere de instalación y permite acceder a la programación del propio microcontrolador. Es un entorno más versátil y de mayor complejidad, orientado a usuario expertos.

Por otra parte, Mbed es un entorno de desarrollo amigable e intuitivo con conjuntos de librerías disponibles y optimizadas para su uso directo. Dispone de un entorno online que no requiere instalación local y dispone además de librerías y códigos de ejemplo. Este entorno orientado a usuarios iniciados no permite acceder a la programación del microcontrolador a bajo nivel.

Dada la naturaleza del proyecto y la facilidad de uso del entorno se opta por el desarrollo del código de nuestra plataforma KL25Z mediante el entorno web de Mbed en el lenguaje C.

Las configuraciones previas a comenzar el desarrollo son dos:

1. Configuración de la plataforma de compilación en el entorno web: KL25Z

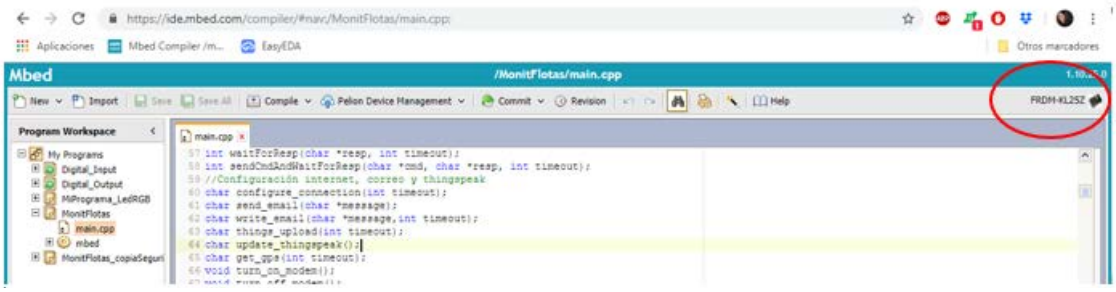


Ilustración 44 Configuración plataforma de compilación entorno online Mbed

2. Descargar e instalar el driver de soporte OpenSDA USB para el sistema operativo en uso (http://www.pemicro.com/opensda).

Hecho esto es posible comenzar a desarrollar.

El propio entorno contiene información de la plataforma de desarrollo, tal y como muestra la siguiente imagen, y dispone de test programas de test específicos para dicha plataforma que permiten familiarizarse con el entorno.

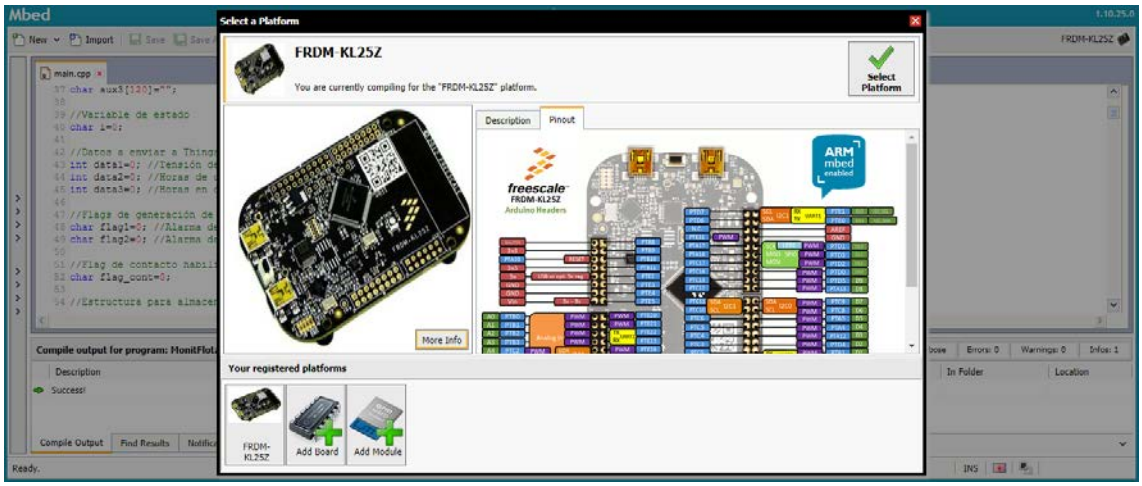


Ilustración 45 Información de la plataforma KL25Z disponible en el entorno online de Mbed

Para programar el dispositivo basta con compilar y copiar el fichero descargado en la KL25Z vía la interfaz OpenSDA, que en el PC se visualizará como un disco externo de almacenamiento.

3.2.2. Visión global del funcionamiento

Analizados los distintos bloques que componen el sistema tanto a nivel hardware como software se pasa a dar una visión global de funcionamiento del mismo.

Tal y como podemos observar en el diagrama de flujo mostrad en la Ilustración 46 el software presenta un comportamiento en bucle.

El software monitoriza de forma continua los parámetros de interés y verifica la necesidad de la generación de alarmas y envío de datos a Thingspeak en base a los datos monitorizados y una serie de temporizadores.

Observamos que independientemente de que un vehículo no genere alarmas durante un tiempo, su estado se actualizará al menos una vez cada cada X tiempo; siendo X el valor umbral fijado para el temporizador global.

Este tiempo en un entorno real sería de 1 día, mientras que los umbrales para las distintas alarmas serían los siguientes:

- Umbral de tiempo máximo de conducción: 1875 horas
- Umbral de tiempo máximo de vehículo encendido: 1875 x 1.25 horas
- Umbral de batería del vehículo: 11.9V

El software establece unos controles de forma que una vez se ha generado una alarma el sistema envíe un único correo de alarma y este se repita de forma periódica cada 24 horas, con el temporizador global.

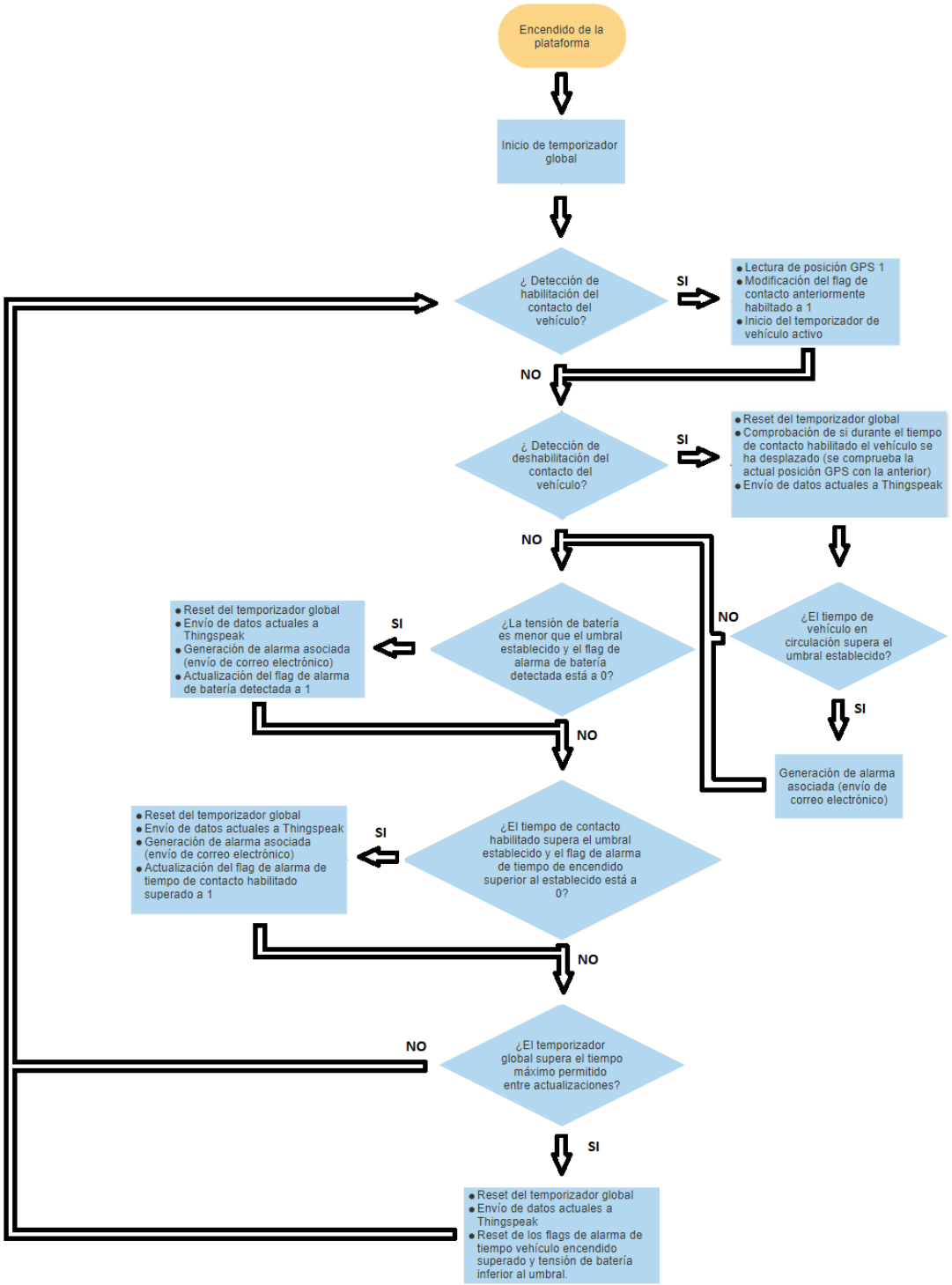


Ilustración 46 Diagrama de flujo del software de la plataforma de desarrollo KL25Z

4. Pruebas realizadas

Una vez desarrollado el sistema se plantean una serie de pruebas para verificar el correcto funcionamiento del mismo.

Dada la imposibilidad de realizar las pruebas en un entorno real, embarcando la solución planteada en un vehículo, es preciso la definición de un entorno de test simulado.

Las *inputs* del vehículo hacia la plataforma embarcada a simular son 2:

- Tensión de batería del vehículo.
- Señal de contacto del vehículo.

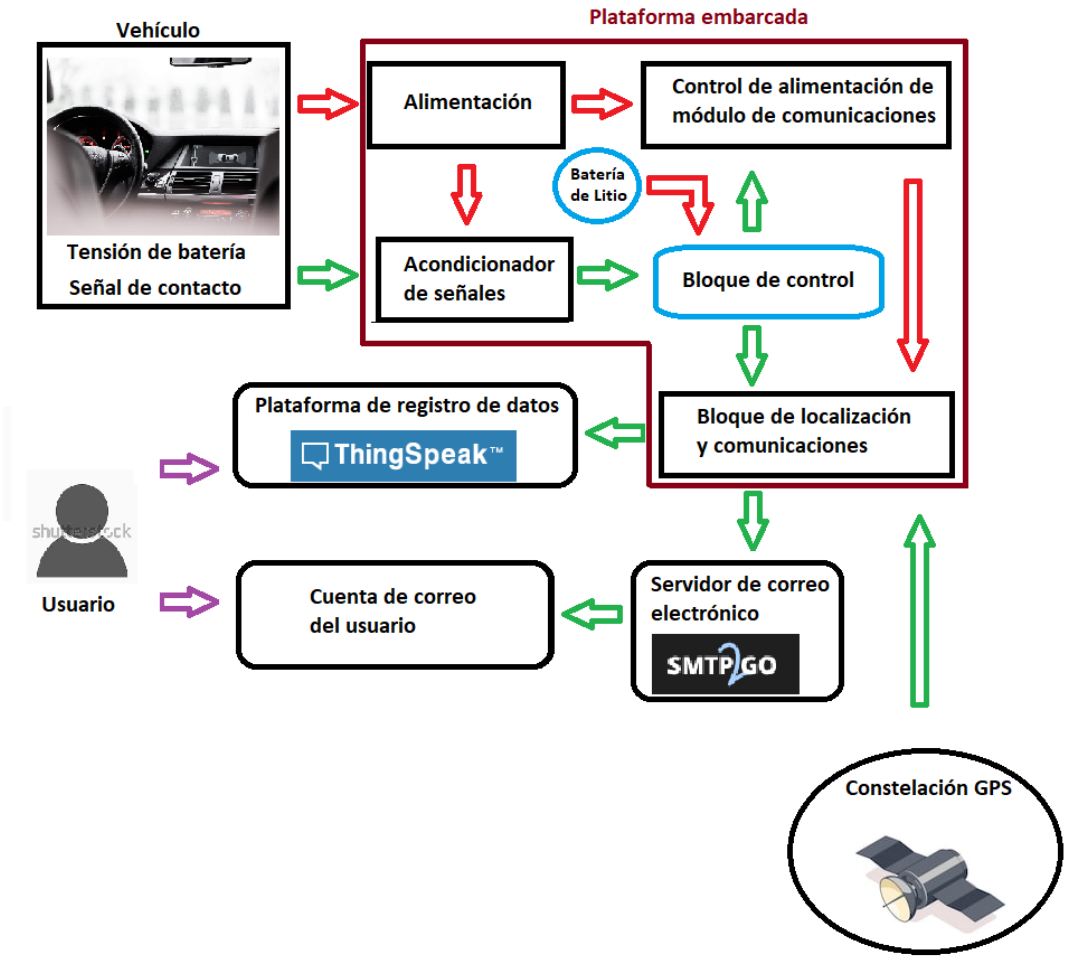


Ilustración 47 Diagrama de bloques del sistema – Inputs a simular en entorno de test marcadas en rojo

Es esquema lógico en el vehículo sería el siguiente:

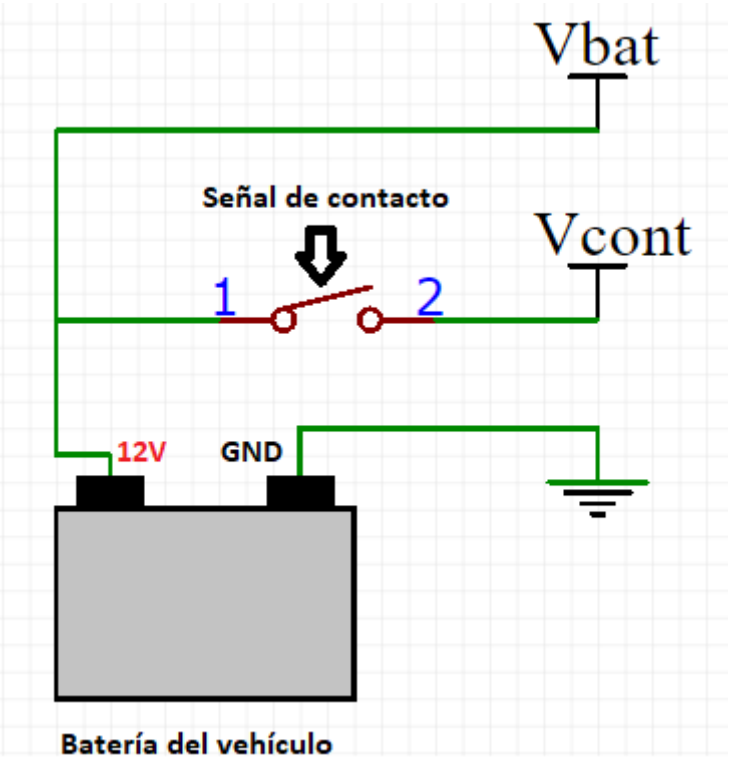


Ilustración 48 Diagrama de la interfaz de conexión plataforma embarcada - vehículo

Para simular la entrada Vbat en el entorno definido se utiliza una fuente externa de 12V regulable entre 10 y 13V, de forma que se puede simular la variación de tensión en batería.

Para simular Vcont se utiliza esta misma tensión de la fuente de alimentación con un interruptor, de forma que pueda activarse y desactivarse la señal de contacto.

Por otra parte, para simular el movimiento del vehículo es preciso forzarlo mediante el código. Se debe tener en cuenta que en las pruebas en estático la antena GPS nunca obtendrá dos posiciones distintas, de forma que nunca se considerará que el vehículo ha estado en movimiento.

Para simular esta casuística se establece que la posición inicial, obtenida al detectar la activación del contacto del vehículo, no sea la leída por el GPS si no que sea una posición fija diferente a la de la ubicación en la cual se esté realizando el test.

```
//Obtengo la posición GPS inicial, queda almacenada en location1
get_gps(120);
//Apago el módulo SIM808
turn_off_modem();

//Guardo la posición actual en la variable auxiliar location2
location2=location1;

//Para simular movimiento del vehíuclo en las pruebas en estático se sustituye
//la línea anterior por estas dos siguientes
//que fuerzan a que la posición inicial obtenida sea diferente a la final
//location2.longitude=08.000000;
//location2.latitude=24.003579;
```

Ilustración 49 Extracto de código a modificar para simulación de vehículo en conducción

En el código desarrollado, disponible en el Anexo I: Código desarrollado para plataforma KRDM-KL25Z, se han dejado comentadas las líneas de código utilizadas para esta simulación.

Finalmente, como ya se indicó anteriormente, cabe notar que los tiempos utilizados para la simulación no son los que se implantarían en la vida real. Los tiempos utilizados para las pruebas son:

- Temporizador global, tiempo tras el cual se realizan actualizaciones automáticas: 15 minutos.
- Tiempo máximo de conducción, tiempo de conducción tras el cual se genera la alarma asociada: 15 minutos.
- Tiempo de encendido, tiempo de vehículo encendido tras el cual se genera la alarma asociada: 18 minutos.

Recordar que la tensión umbral de batería para la cual se genera la alarma asociada es de 11.9V.

A continuación se detallan las pruebas realizadas y los resultados obtenidos en este entorno.

4.1. Test 1: generación de alarma de batería

El objetivo de esta prueba es verificar la correcta generación de la alarma asociada a la degradación de la tensión de batería así como comprobar que una vez generada una alarma esta se vuelve a generar de forma periódica según el temporizador global.

La siguiente imagen muestra los datos exportados de la plataforma Thingspeak en formato Excel para el día 7 de mayo, momento de realización del test siendo el *field*1 la tensión de batería del vehículo en mV, el *field* 2 el tiempo de encendido en minutos y el *field* 3 el tiempo de circulación en minutos.

1	created_at,entry_id,field1,field2,field3
2	2019-05-07 19:33:02 +0200,11,11513,0,0
3	2019-05-07 19:48:02 +0200,12,11486,0,0
4	2019-05-07 19:49:03 +0200,13,11463,0,0
5	2019-05-07 20:04:04 +0200,14,11640,0,0
6	2019-05-07 20:05:06 +0200,15,11480,0,0
7	2019-05-07 20:20:06 +0200,16,11457,0,0
8	2019-05-07 20:21:07 +0200,17,11545,0,0
9	2019-05-07 20:36:07 +0200,18,11482,0,0
10	2019-05-07 20:37:09 +0200,19,11526,0,0
11	2019-05-07 20:52:10 +0200,20,11514,0,0
12	2019-05-07 20:53:11 +0200,21,11469,0,0
13	2019-05-07 21:08:11 +0200,22,11511,0,0
14	2019-05-07 21:09:13 +0200,23,11521,0,0
15	2019-05-07 21:24:13 +0200,24,11465,0,0
16	2019-05-07 21:25:14 +0200,25,11474,0,0
17	2019-05-07 21:40:15 +0200,26,11556,0,0
18	2019-05-07 21:41:16 +0200,27,11498,0,0
19	2019-05-07 21:56:17 +0200,28,11432,0,0

Ilustración 50 Extracto de datos Thingspeak asociados a la prueba 1 con marca en las actualizaciones asociadas a la generación de la alarma de batería

Observamos que las actualizaciones se realizan de forma periódica cada 15 o 16 minutos (debemos tener en cuenta que dada la estructura serie del código la comprobación periódica de batería puede darte unos segundos después lo que se traduce en que la actualización aparezca reflejada un minuto más tarde, esto es aplicable a todas las temporizaciones). Además, dado que la generación de una alarma de batería da lugar a una actualización de Thingspeak se observa una actualización 1 minutos después de cada actualización periódica (por ejemplo en los puntos 4 y 6 de la imagen anterior).

Junto a esta actualización se genera la alarma que da lugar al envío del correo electrónico de aviso. En la Ilustración 51 y la Ilustración 52 se pueden ver los correos generados en nuestra bandeja de entrada y el contenido de uno de ellos, mientras que en la Ilustración 53 se observan los tiempos de envío de los distintos correos de alarma asociados al servidor SMTP2GO el día de la realización de la prueba, 7 de mayo de 2019.

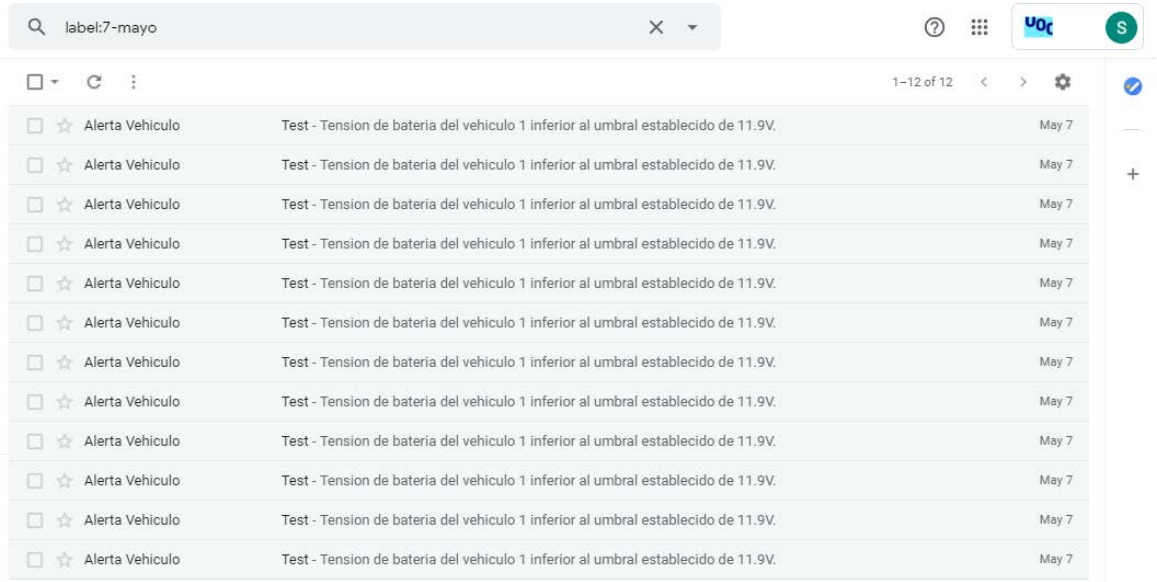


Ilustración 51 Bandeja de entrada Gmail con los correos de alarma del 7 de mayo de 2019

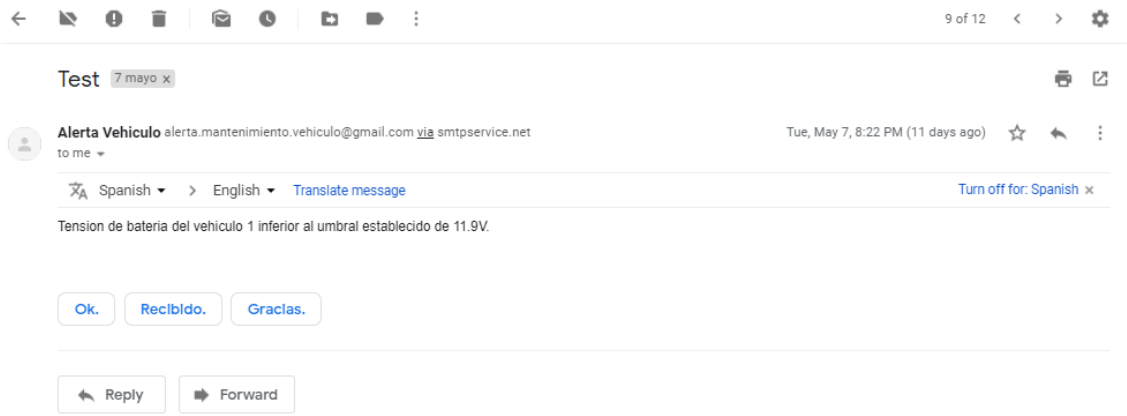


Ilustración 52 Correo de alarma de batería del 7 de mayo de 2019

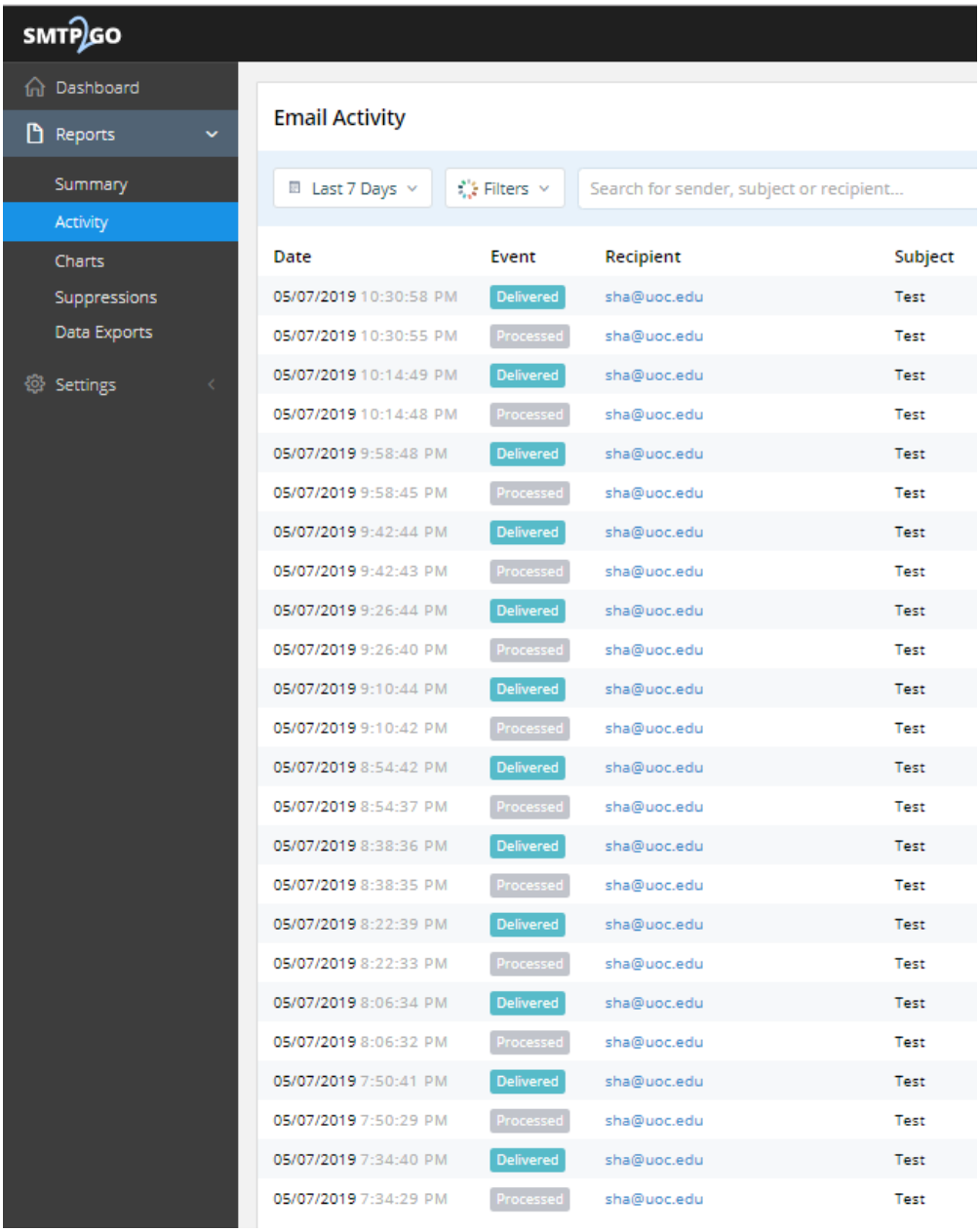


Ilustración 53 Registro de correos enviados por SMTP2GO el 7 de mayo de 2019

4.2. Test 2: generación de alarma de tiempo de encendido superado

El objetivo de esta prueba es verificar la correcta generación de la alarma asociada a la superación del tiempo de encendido del vehículo.

Durante la realización de este test se utiliza el código modificado para simular el movimiento del vehículo.

La secuencia de realización del test es la siguiente (reflejada en la Ilustración 54 con los datos exportados de Thingspeak).

1. Entrada 26: a las 18.45 de enciende la plataforma con una tensión inferior a 11.9 Vdc. Esto genera el envío de una alarma de batería. Posteriormente se modifica la tensión por encima del umbral de forma que esta alarma no vuelve a generarse. (Ilustración 56, Ilustración 55 e Ilustración 58)

2. Entrada 27: se genera la actualización periódica.

3. A las 19.27 se habilita el contacto de la plataforma.

4. En la entrada 28 se registra un tiempo de contacto habilitado de 8 minutos.

5. A las 19.37 se deshabilita el contacto lo que fuerza una actualización del tiempo de contacto habilitado y del tiempo de conducción, dada que se está simulando el movimiento de la plataforma.

6. Esta secuencia vuelve a repetirse en la entrada 31 lo que aumenta de nuevo el tiempo de conducción.[3]

7. En la entrada 32 a las 20:00 se supera el umbral de tiempo de conducción de 18 minutos lo que fuerza a una actualización de Thingspeak y al envío del primer correo de alarma. (Ilustración 55, Ilustración 57 e Ilustración 58)

8. En la entrada 33, 15 minutos después que la entrada 32 de acuerdo al temporizador global, se actualiza de nuevo Thingspeak. Al igual que en el test 1 el *reset* del *flag* de las distintas alarmas fuerza 1 minutos más tarde a una nueva actualización y el envío de un nuevo aviso de alarma (Ilustración 58 e Ilustración 55),

[3] El importante notar que el tiempo de conducción únicamente se actualiza cuando se deshabilita el contacto, momento en el cual se comparan las posiciones. Es por ello que aunque el tiempo de encendido aumenta de forma progresiva esto no es así con el tiempo de conducción.

1	created_at,entry_id,field1,field2,field3
26	2019-05-08 19:01:10 +0200,35,11828,0,0
27	2019-05-08 19:16:12 +0200,36,13227,0,0
28	2019-05-08 19:32:13 +0200,37,13525,8,0
29	2019-05-08 19:37:18 +0200,38,13145,11,11
30	2019-05-08 19:50:24 +0200,39,13212,12,11
31	2019-05-08 19:53:25 +0200,40,13144,15,14
32	2019-05-08 20:00:58 +0200,41,13183,19,14
33	2019-05-08 20:16:00 +0200,42,13212,34,14
34	2019-05-08 20:17:03 +0200,43,12848,35,14

Ilustración 54 Extracto de datos Thingspeak asociados a la prueba 2

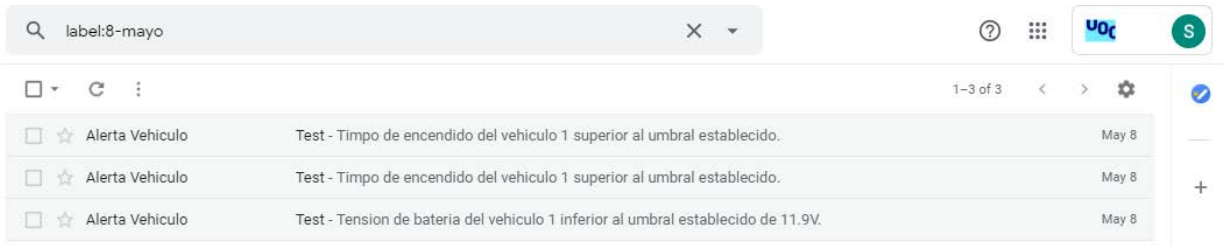


Ilustración 55 Bandeja de entrada Gmail con los correos de alarma del 8 de mayo de 2019

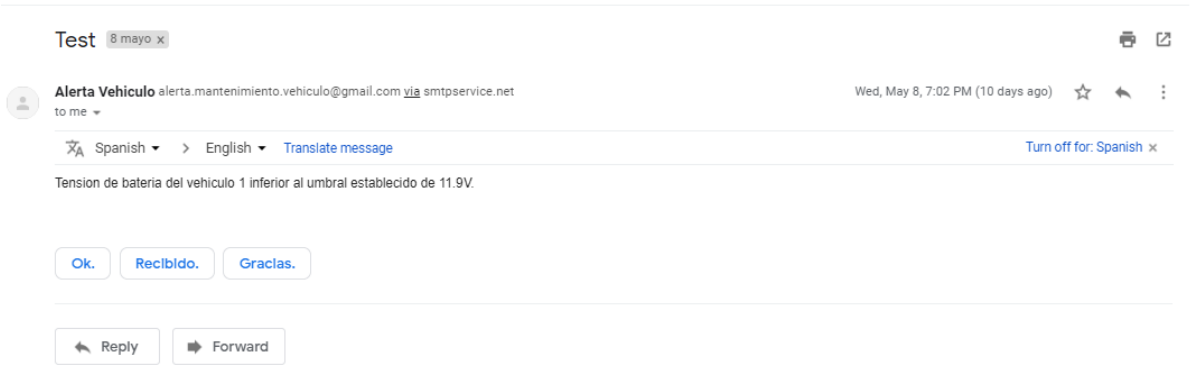


Ilustración 56 Correo de alarma de batería del 8 de mayo de 2019

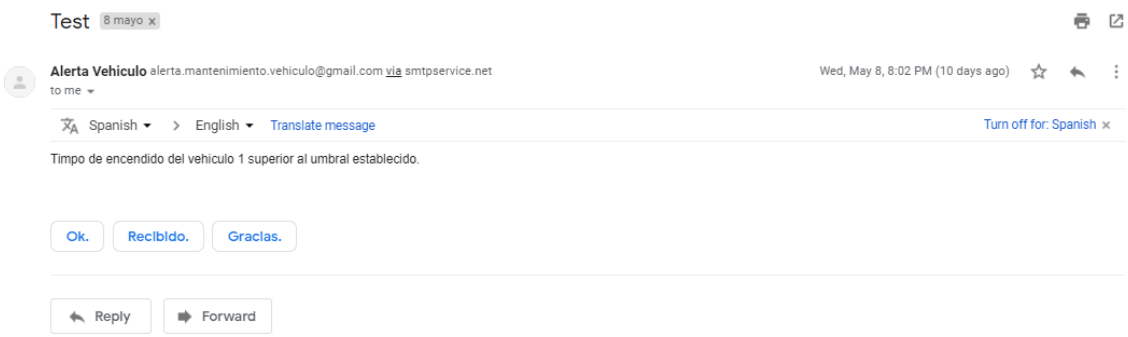


Ilustración 57 Correo de alarma de tiempo de encendido del vehículo del 8 de mayo de 2019

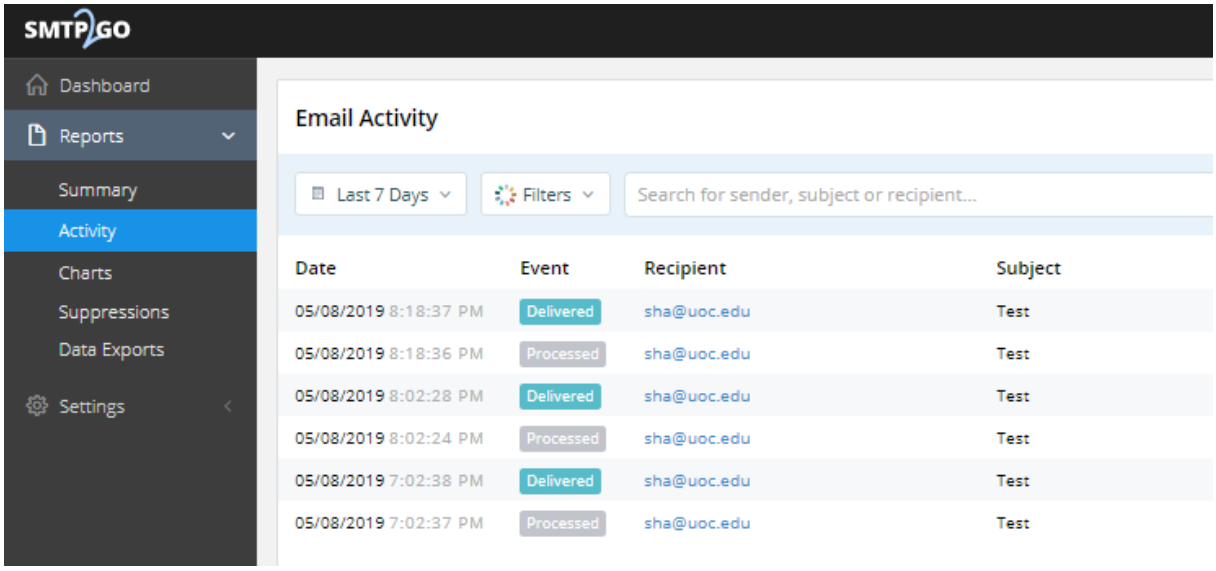


Ilustración 58 Registro de correos enviados por SMTP2GO el 8 de mayo de 2019

4.3. Test 3: generación de alarma de tiempo de conducción superado

El objetivo de esta prueba es verificar la correcta generación de la alarma asociada a la superación de conducción del vehículo.

Durante la realización de este test se utiliza el código modificado para simular el movimiento del vehículo.

La secuencia de realización del test es la siguiente (reflejada en la con los datos exportados de Thingspeak).

1. Encendido de la plataforma a las 20:30 sin contacto habilitado ni tensión de batería inferior al umbral.

2. Entrada 37: primera actualización periódica de Thingspeak.

3. Entrada 38: segunda actualización periódica de Thingspeak, 15 minutos más tarde según el temporizador global.

4. Habilitación del contacto del vehículo a las 13.08.

5. Entrada 39: primer registro con incremento de tiempo de contacto habilitado, 3 minutos.

6. Entrada 40: actualización periódica en la cual se llega al umbral de tiempo de encendido superado. Se genera el correo de alarma de asociado (Ilustración 60, Ilustración 61 e Ilustración 63).

7. En la entrada 28 se registra un tiempo de contacto habilitado de 8 minutos.

8. A las 19.37 se deshabilita el contacto lo que fuerza una actualización del tiempo de contacto habilitado y del tiempo de conducción, dada que se está simulando el movimiento de la plataforma.

9. Entrada 41: se deshabilita el contacto. Esto fuerza a una actualización de Thingspeak y a una actualización del tiempo de circulación. Puesto que este tiempo supera el umbral de 15 minutos se genera la alarma asociada (Ilustración 60, Ilustración 62 e Ilustración 63).

10. Posteriormente de forma periódica tras las actualizaciones cada 15 minutos del Thingspeak se fuerza al nuevo envío de una alarma de tiempo de encendido del vehículo superado.

1	created_at,entry_id,field1,field2,field3
37	2019-05-12 12:39:02 +0200,46,13186,0,0
38	2019-05-12 12:55:03 +0200,47,13182,0,0
39	2019-05-12 13:11:04 +0200,48,13213,3,0
40	2019-05-12 13:26:10 +0200,49,13173,19,0
41	2019-05-12 13:31:55 +0200,50,13102,22,22
42	2019-05-12 13:44:44 +0200,51,13158,22,22
43	2019-05-12 13:45:46 +0200,52,13133,22,22
44	2019-05-12 14:00:46 +0200,53,13108,22,22
45	2019-05-12 14:01:48 +0200,54,13196,22,22

Ilustración 59 Extracto de datos Thingspeak asociados a la prueba 3

Ilustración 60 Bandeja de entrada Gmail con los correos de alarma del 12 de mayo de 2019

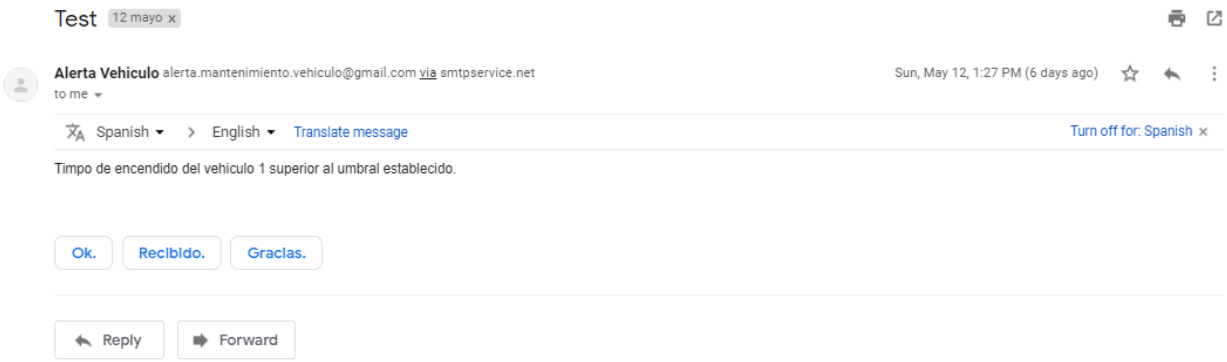


Ilustración 61 Correo de alarma de tiempo de encendido del vehículo del 12 de mayo de 2019

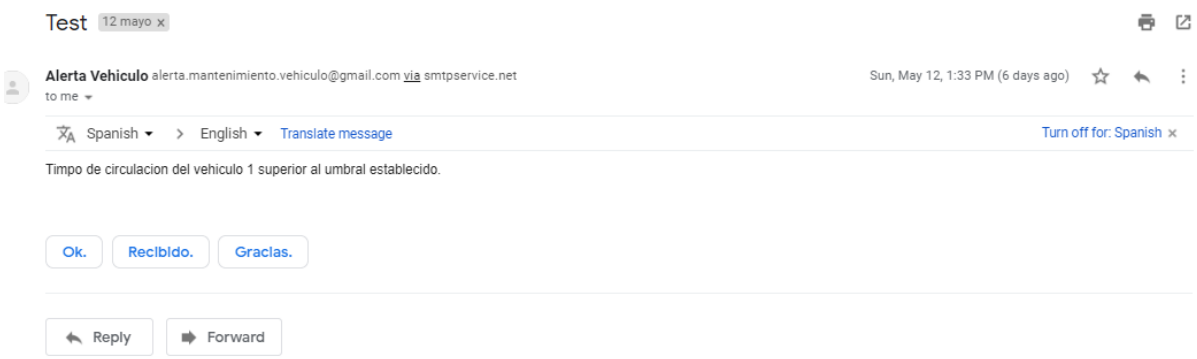


Ilustración 62 Correo de alarma de tiempo de circulación del vehículo del 12 de mayo de 2019

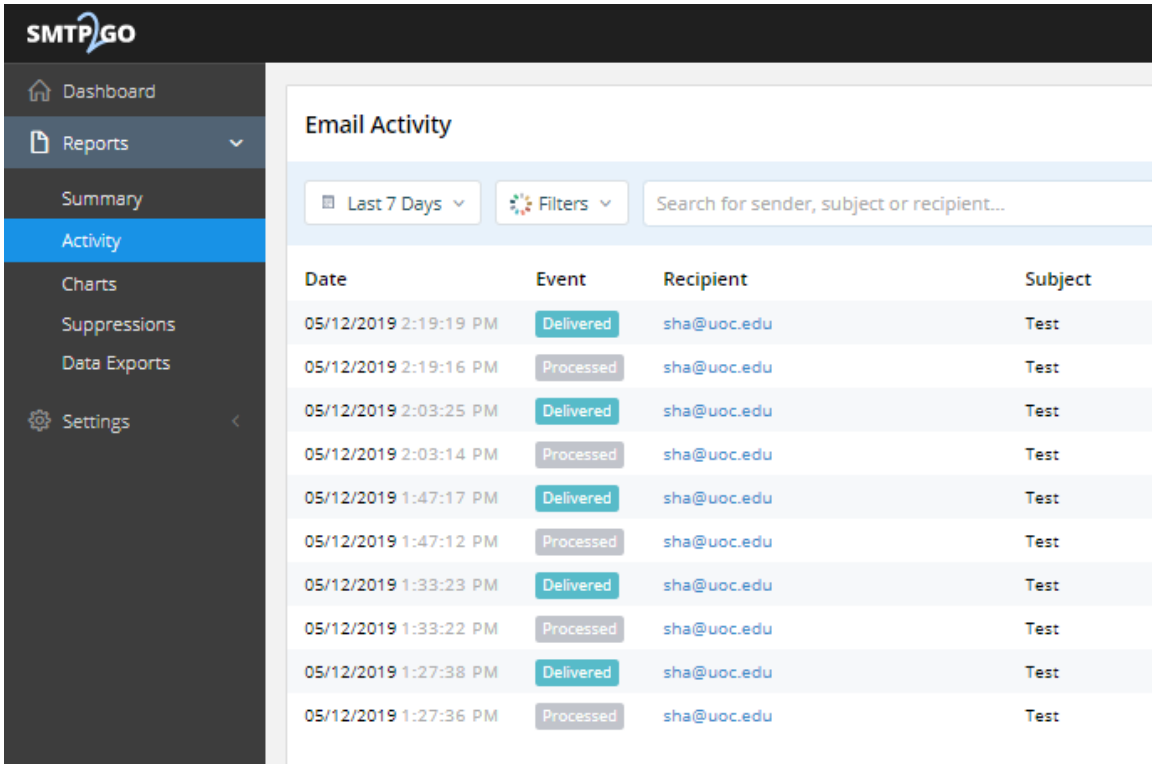


Ilustración 63 Registro de correos enviados por SMTP2GO el 12 de mayo de 2019

4.4. Test 4: análisis y monitorización de consumos

Un elemento de vital importancia en el diseño realizado en este proyecto es el consumo del sistema implementado. La plataforma utiliza dos fuentes de energía: la batería del vehículo y una batería de litio auxiliar utilizada para alimentar la plataforma de desarrollo KL25Z. Esto supone dos cosas fundamentales:

1. Se debe asegurar que la plataforma no descargue la batería del vehículo de forma que gaste dicha batería.
2. Se debe asegurar que la batería de litio que alimente la placa de desarrollo KL25Z proporciona al sistema una autonomía suficiente.

A continuación se analiza el consumo para cada uno de los bloques de la plataforma (bloque alimentado por la batería del vehículo y plataforma de desarrollo alimentada por la batería de litio auxiliar).

4.4.1. Análisis y medida de la potencia suministrada por la batería del vehículo

En base al primer punto se realiza el siguiente análisis para asegurar el correcto dimensionamiento de la plataforma en términos de consumo, así como su posterior medida real.

Relativo a la batería, el principal parámetro a tener en cuenta en este análisis es su capacidad nominal. Este parámetro medido en Ah (Amperios hora) indica la cantidad de energía que una batería es capaz de suministrar en una hora, estando totalmente cargada. Es decir, define la capacidad de energía de la batería.

Para determinar un valor típico de capacidad nominal se han considerado los datos típicos publicados la revista web *Autofacil* en su sección de mecánica [31]. En ella se indica que los valores de capacidad nominal de la batería de un vehículo pueden oscilar entre 50Ah en un vehículo urbano de gasolina hasta 100Ah en una berlina de lujo. Con objeto de comprobar el correcto funcionamiento de la plataforma en el entorno más desfavorable se realizan los cálculos asumiendo una capacidad nominal de 50Ah.

Definida la capacidad nominal se procede a analizar el consumo de los principales módulos de la plataforma alimentados por la batería del vehículo siendo estos el módulo de comunicaciones y los relés de control de dicho módulo.

El módulo de comunicaciones SIM808 es el componente de mayor consumo dentro del sistema. El fabricante de la placa de evaluación indica un consumo máximo de **2W**. [32]

Esta placa recibe alimentación de 12V procedente de la batería del vehículo previamente ajustada por un regulador lineal, cuya máxima

eficiencia es del 80%. De este modo, la corriente a suministrar por la batería del vehículo para la alimentación del módulo es de $\frac{2/12}{0.8} = \mathbf{0.208\ A}$.

Por otra parte, los *shield* de relé presentan según *datasheet* un consumo máximo de **0.36W**. Este *shield* recibe alimentación de 3.3V procedente de la batería del vehículo previamente ajustada por tres reguladores lineales, cuya máxima eficiencia es del 80%. De este modo, la corriente a suministrar por la batería del vehículo para la alimentación del módulo es de $\frac{0{,}36/3.3}{0.8\ 0.8\ 0.8} = \mathbf{0.213\ A}$. La plataforma integra dos *shield* de relés.

A la vista de los cálculos la corriente a suministrar por la batería del vehículo a la plataforma nunca superará los **0.634A**. En base a esto se calcula el tiempo que tardaría la plataforma en descargar una batería de vehículo de capacidad nominal 50Ah.

$$Tiempo\ de\ descarga = \frac{50\ [Ah]}{0.634A} = 78{,}864\ [h]$$

El sistema descargaría la batería del vehículo en 78,864 horas. Ahora bien, se debe tener en cuenta que en el caso crítico en que el vehículo permanezca deshabilitado durante mucho tiempo, de forma que no su batería no se cargue, el sistema encenderá los módulos dependientes de la batería un máximo de 5 minutos diarios para realizar la actualización diaria de los datos en Thingspeak. En base a esto calculamos el número de días de autonomía del sistema con el vehículo apagado.

$$Tiempo\ de\ descarga = \frac{78{,}864[horas]x\ 60[minutos]}{5\ \left[\frac{minutos}{día}\right] 1\ [hora]} = 946{,}36\ días$$

El sistema realizando una actualización diaria de la plataforma Thingspeak tardaría 946 días en descargar la batería del vehículo, más de dos años, un tiempo más que razonable.

La siguiente tabla recoge las medidas máximas de corriente obtenidas en la plataforma real de test junto con el consumo máximo teórico, observamos que en ningún caso se supera el máximo teórico por lo que la autonomía real del sistema sería mayor de lo calculado. Para la obtención de la corriente se utiliza un multímetro digital en serie en el punto a medir. Las medidas obtenidas son: corriente en la línea de alimentación de un relé a 3.3V, corriente en la línea de alimentación de la placa del módulo de comunicaciones SIM808 durante su uso para la obtención de una localización GPS alimentado a 12V y durante el uso de GPRS durante el envío de una actualización de la plataforma Thingspeak.

Elemento	Consumo máximo teórico	Consumo máximo medido
***Shield* de relé**	**0.36W**	0.031A@3.3V, **0.1023W**
SIM808 durante obtención de posición GPS	**2W**	0.072A@12V, **0.864W**
SIM808 durante envío de datos por GPRS	**2W**	0.083@12V, **0.996W**

Tabla 5 Comparativa de consumos teóricos y medidos de los principales elementos alimentados por la batería del vehículo

La secuencia utilizada para la medida es la siguiente:

1. Activación del contacto del vehículo. Esta acción causa la habilitación del relé y permite medir su consumo. Además, el módulo de comunicaciones obtiene la posición inicial del vehículo por lo que una repetición de esta misma secuencia permite medir el consumo del módulo de comunicaciones durante la obtención de posición GPS.

2. Desactivación del contacto del vehículo. Esta acción desencadena lo siguiente: primero se activa el módulo de comunicaciones para obtención de posición GPS, momento en el cual se vuelve a mediar su consumo, posteriormente se desactiva y vuelve a activarse para enviar datos por GPRS a *Thingspeak*, momento en el cual se mide el consumo del módulo de comunicaciones para transmisión GPRS.

3. Finamente se verifica que el envío de datos GPRS ha tenido lugar mediante la comprobación de que *Thingspeak* se ha actualizado.

Las siguientes fotografías muestras los momentos de realización de las medidas.

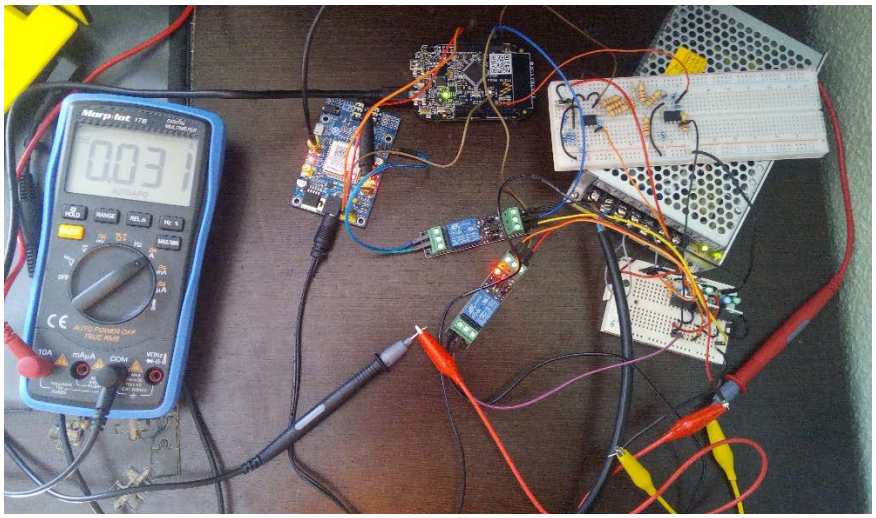


Ilustración 64 Medida de consumo máximo del relé en su alimentación de 3.3V

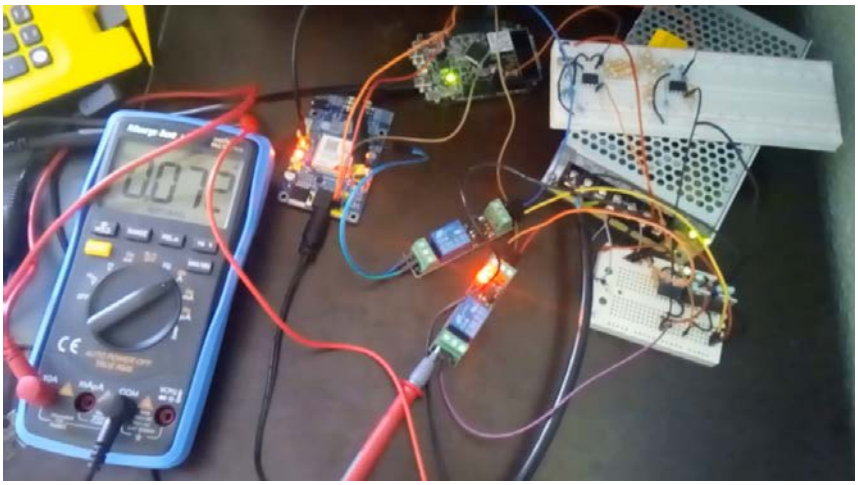


Ilustración 65 Medida de consumo máximo del módulo de comunicaciones durante la obtención de la posición GPS en su línea de alimentación de 12V.

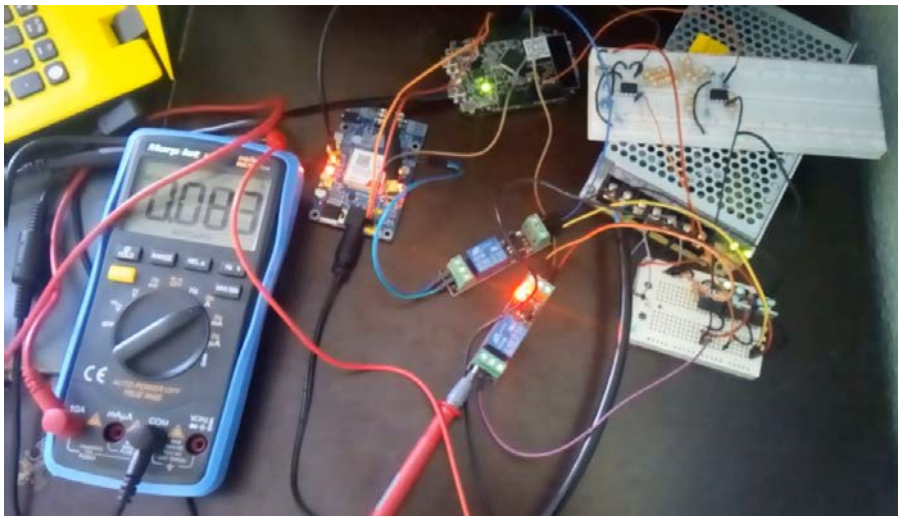


Ilustración 66 Medida de consumo máximo del módulo de comunicaciones durante la transmisión de datos por GPRS en su línea de alimentación de 12V.

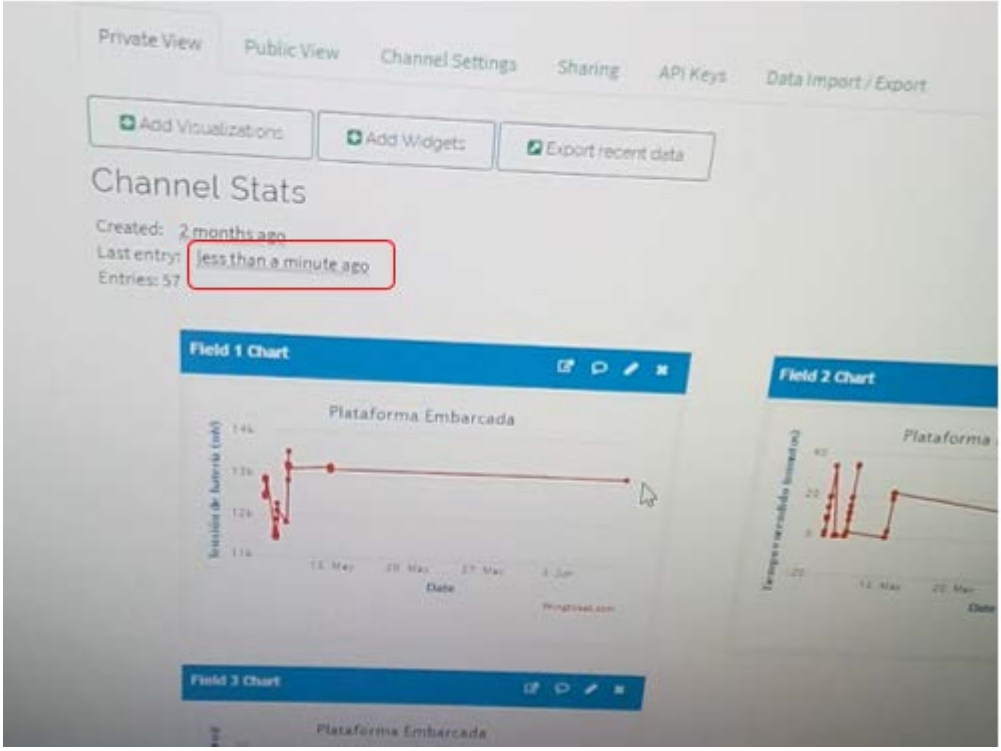


Ilustración 67 Verificación de la reciente actualización de Thingspeak tras la medida con consumo del SIM808 durante la transmisión de datos por GPRS.

4.4.2. Análisis y medida de la potencia suministrada por la batería de litio

En base al segundo punto analizamos la autonomía proporcionada al sistema por la batería de litio utilizada para alimentar la placa de desarrollo KL25Z. La placa de desarrollo KL25Z admite varias fuentes de alimentación (USB, fuente de alimentación, batería…) las características de las distintas fuentes admitidas se recogen en la siguiente ilustración extraída del manual de usuario de la plataforma.

Supply Source	Valid Range	OpenSDA Operational?	Regulated on-board?
OpenSDA USB (J7)	5V	Yes	Yes
KL25Z USB (J5)	5V	No	Yes
V_{IN} Pin	4.3-9V	No	Yes
3.3V Pin	1.71-3.6V	No	No
Coin Cell Battery	1.71-3.6V	No	No

Ilustración 68 Características de fuentes de alimentación de la plataforma KL25Z [27]

El principal motivo para utilizar una batería de litio para la alimentación de la placa de desarrollo es proteger al microcontrolador de las posibles fluctuaciones en la tensión de batería.

Para la medida del consumo se alimenta la placa de desarrollo mediante su interfaz de 5V procedentes del USB. Se monitoriza una secuencia de funcionamiento completo durante la cual es máximo consumo medido se da en el momento en el cual la plataforma activa sus dos salidas digitales utilizadas para el control de los *shield* de reles. Esto consumo máximo medido es de 0.024A@5V, **0.12W**.

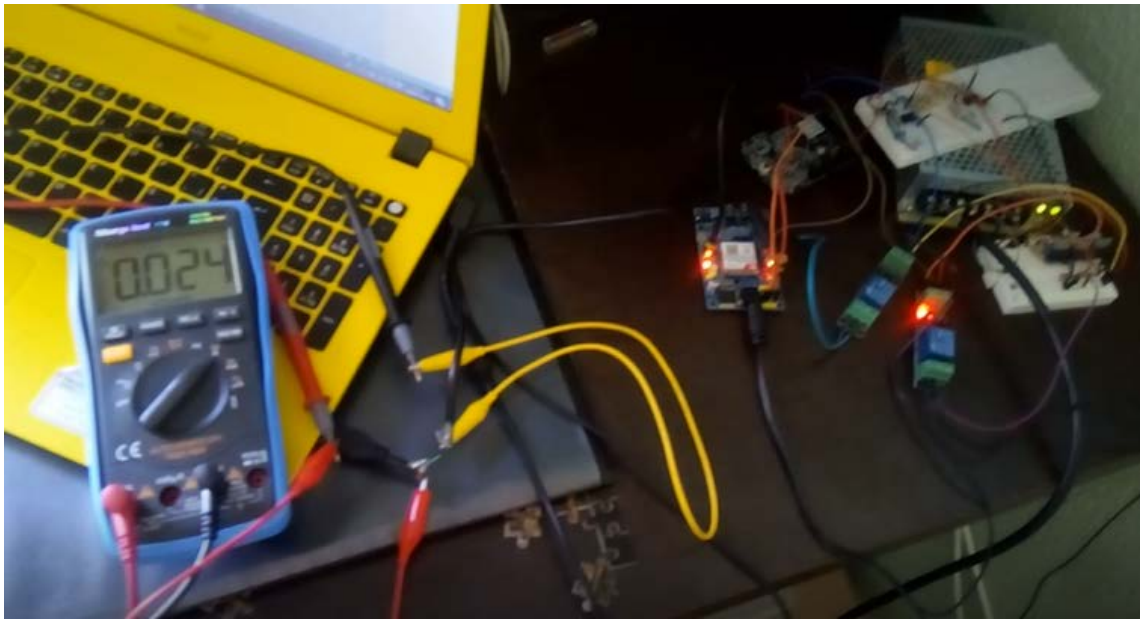


Ilustración 69 Medida de consumo máximo de la plataforma KL25Z en su interfaz de alimentación USB de 5V.

Establecemos el uso de una batería de litio de 3.6V. A modo de ejemplo establecemos la batería de litio de LG C2601036SZ, de capacidad nominal de 78000mAh. [33]

$$Duración = \frac{78\ [Ah]}{0.12[W] / 3.6[V]} = 2340\ [horas]$$

La duración de la batería sería de 2440 horas, es decir, 3 meses. Para alargar la autonomía se utilizaría un banco de 4 pilas de litio en paralelo con objeto de hacer coincidir la sustitución de la pila con las tereas de mantenimiento del vehículo.

5. Análisis de los objetivos y propuestas de mejora

En este capítulo se analiza el cumplimiento de los objetivos propuestos para este trabajo en el apartado 1.2 de esta memoria. Además, se define una línea de trabajo futuro orientada a la mejora del sistema implementado.

5.1. Análisis de objetivos

- Objetivo 1. Desarrollo e implementación de un sistema completo que permita la monitorización y almacenamiento en una plataforma web de código abierto de parámetros clave en el desgaste de un vehículo y la generación de alarmas en relación a estos parámetros.

  El sistema implementado proporciona las funcionalidades objetivo y permite al usuario final la consulta de los datos almacenados en un entorno amigable, además de su exportación para un posterior procesado.

- Objetivo 2. Generación de la documentación y los entregables de las distintas fases de desarrollo tanto desde la perspectiva hardware como software.

  Tanto el diagrama de flujo como el esquemático han sido desarrollados en paralelo a la implementación, reflejando de forma fiel el sistema implementado.

- Objetivo 3. Adquisición de las capacidades relacionadas con el uso de plataformas de desarrollo embebidas basadas en el uso de un procesador ARM.

  El uso del entorno de desarrollo Mbed ha permitido adquirir las capacidades necesarias para el uso de dichas plataformas con un procesador ARM.

- Objetivo 4. Definición de un sistema con bloques definidos abierto a ampliaciones.

  El sistema se ha definido de forma que permite ampliaciones hardware y software de forma sencilla, además, los elementos software utilizados (SMTP2, Thingspeak) son ampliables en su capacidad.

5.2. Propuestas de mejora

El uso de la plataforma de desarrollo KL25Z ha permitido la definición de un sistema de monitorización versátil y ampliable. Además, la lógica del código se ha establecido de modo que sea escalable y ampliable.

En base a esto una primera propuesta de mejora sería implementar nuevas funcionalidades dentro del sistema, destacando las siguientes:

- Utilización del potencial del módulo de comunicaciones para implementar la localización de flotas mediante el envío continuo de la posición de cada vehículo y su visualización en un entorno web.

- Incorporación a la plataforma embarcada de un emisor acústico que permita en base al tiempo de encendido del vehículo, ya monitorizado, el envío de alarmas acústicas al conductor cuando supere el tiempo máximo de conducción continua recomendado. Dicha funcionalidad de control de tiempos de conducción y descanso de los conductores de las flotas es de crucial importancia para una conducción segura.

Una segunda propuesta sería la optimización del hardware mediante la omisión del uso de la plataforma de desarrollo KL25Z y su sustitución por una placa propia. Se debe tener en cuenta que la plataforma de desarrollo dispone de gran cantidad de funcionalidades no utilizadas, por lo que de cara a una implementación real resultaría más rentable contar con una placa de desarrollo propio con las funciones necesarias. En base a los esquemas proporcionados por NXP de la plataforma de desarrollo KL25Z incluido en el Anexo III: Esquemático de la plataforma KL25Z de esta memoria, los elementos a omitir serían los siguientes:

- Sensor inercial
- Interfaz capacitiva
- LED RGB

Dentro de la mejora del hardware se incluiría el diseño de una etapa de alimentación protegida para alimentar el KL25Z desde batería, evitando el uso de baterías de litio, lo que permitiría aumentar la autonomía de la plataforma. Si bien sería de interés mantener la batería de litio como *back-up* de energía.

6. Conclusiones

El presenta trabajo ha mostrado el desarrollo completo de un sistema ITS, ampliamente utilizados y en auge. Se ha logrado un sistema claro y funcional pensado para hacer del mundo real un lugar más seguro.

Al inicio de este trabajo se barajaron múltiples funciones a implementar, materializándose finalmente las más fundamentales. Conocida la necesidad futura de ampliación se ha desarrollado un sistema ampliable y versátil, preparado para el futuro.

El desarrollo del proyecto se ha realizado en tiempo y forma cumpliéndose con los objetivos temporales estimados, si bien hubiera resultado de interés el contar con más tiempo de test con objeto de realizar pruebas reales en vehículo se ha logrado testear el sistema completo sin este entorno.

7. Glosario

ITS (*Inteligent Transport Systems* o Sistemas de Transporte Inteligente): conjunto de soluciones tecnológicas de telecomunicaciones y telemática diseñadas para mejorar la operación y seguridad del trasporte terrestre. [34]

IoT (*Internet of Things*): concepto referido a la interconexión de elementos de la vida diaria con Internet.

TFM (Trabajo Fin de Máster)

TIC (Tecnologías de la Información y la Comunicación): agrupación de redes, servicios, software y equipos integrados en un sistema de información.

RFID (*Radio Frequency Identification*): tecnología para almacenamiento y recuperación de datos basada en *tags.*

EMV (*Europay MasterCard VISA*): estándar de interoperabilidad de tarjetas bancarias para autenticación de pagos.

GNSS (*Global Navigation Satellite System*): sistema global de navegación por satélite utilizado para posicionamiento global.

GPS (*Global Positioning System*): sistema GNSS implementado por el Departamento de Defensa de Estados Unidos.

GPIO (*General Purpose Input/Output*): denominación para un pin genérico de un integrado cuyo comportamiento es configurable mediante código.

GSM (*Global System for Mobile Communications*): estándar de comunicaciones móviles digital, perteneciente a la segunda generación.

GPRS (*General Packet Radio Service):* estándar de comunicaciones móviles digitales extensión del GSM.

UMTS (*Universal Mobile Telecommunications System*): sistema de comunicaciones móviles de tercera generación desarrollado dentro del marco de trabajo de la ITU (International Telecommunications Union).

LTE (*Long Term Evolution*): estándar de comunicaciones móviles para transmisión de datos a alta velocidad, ubicado entre la tercera y la cuarta generación.

EMC (*Electromagnetic Compatibility)*: rama de la ingeniería orientada al estudio de la generación, propagación y recepción no intencionada de energía electromagnética y sus efectos asociados.

CAD (Convertidor Analógico Digital): dispositivo electrónico para codificación digital de una señal analógica mediante muestreo, cuantificación y codificación.

PCB (Printed Circuit Board): superficie que integra pistas conductoras que conecta a diferentes componentes electrónicos en base a un diseño previo.

SMTP (Simple Mail Transfer Protocol) protocolo de comunicación para el envío de correos electrónicos en internet.

8. Bibliografía

[1] Ministerio de Fomento de España, «Web Ministerio de Fomento de España,» [En línea]. Available: https://www.fomento.gob.es/transporte-terrestre/sistemas-inteligentes-de-transporte/sistemas-inteligentes-de-transporte-its. [Último acceso: 16 03 2019].

[2] J. R. y. L. Fernando, «Sistemas Inteligentes de Transporte y nuevas tecnologías en el conttrol y administración del transporte,» *Puente Revista Científica - Universidad Pontificia Bolivariana,* pp. 53-62, 2016.

[3] «Tecnología para los negocios - Cámara de Valencia,» [En línea]. Available: https://ticnegocios.camaravalencia.com/industria/tendencias/la-transformacion-digital-sector-del-transporte-la-logistica/ . [Último acceso: 10 03 2019].

[4] GTZ - Proyecto sectorial Servicio de Asesoría en Política de Transporte, «SUTP - Sustainable Urban Transport Projects,» 2004. [En línea]. Available: https://www.sutp.org/files/contents/documents/resources/A_Sourcebook/SB4_Vehicles-and-Fuels/GIZ_SUTP_SB4e_Intelligent-Transport-Systems_ES.pdf. [Último acceso: 16 03 2019].

[5] «Revista Viajeros,» 17 09 2018. [En línea]. Available: https://www.revistaviajeros.com/noticia/11489/el-sistema-de-billetaje-emv-del-consorcio-de-transportes-de-mallorca-sera-de-gmv. [Último acceso: 09 03 2019].

[6] «Smart City Jaén,» 24 07 2012. [En línea]. Available: http://www.smartcityjaen.com/2012/07/24/smart-public-bus-management-system/?lang=en. [Último acceso: 09 03 2019].

[7] «EuropaPress,» 23 04 2014. [En línea]. Available: https://www.europapress.es/economia/noticia-economia-empresas-espanola-gmv-implantara-sistema-gestion-trafico-tranvia-taiwan-20140423140029.html. [Último acceso: 16 03 2019].

[8] M. Jiménez, «El país - Tecnología,» 28 01 2018. [En línea]. Available: https://cincodias.elpais.com/cincodias/2018/01/26/companias/1516991336_724453.html. [Último acceso: 16 03 2019].

[9] «Empresa Exterior,» 12 03 2018. [En línea]. Available: https://empresaexterior.com/art/65483/puesta-en-marcha-de-las-maquinas-autoservicio-suministradas-por-gmv-en-el-metro-de-santiago-de-chile . [Último acceso: 06 03 2019].

[10] «EFE: Empresas,» 13 04 2016. [En línea]. Available: https://www.efeempresas.com/noticia/gmv-suministrara-su-tecnologia-al-tranvia-ligero-de-sidney/. [Último acceso: 16 03 2019].

[11] «TP Emprender, innovar, triunfar,» 16 03 2019. [En línea]. Available: http://www.ticpymes.es/tecnologia/noticias/1104902049504/carsharing-futuro-de-movilidad-sostenible.1.html. [Último acceso: 16 03 2019].

[12] C. Ibáñez, «Innovadores: La Razón,» 09 08 2018. [En línea]. Available: https://innovadores.larazon.es/es/not/taller-40-conectividad-y-monitorizacion-de-las-flotas-de-vehiculos. [Último acceso: 09 03 2019].

[13] «GMV Blog -¿Sabe usted qué está haciendo en este momento cada uno de los conductores de su flota de camiones?,» 15 01 2019. [En línea]. Available: https://www.gmv.com/blog_gmv/language/es/sabe-usted-que-esta-haciendo-en-este-momento-cada-uno-de-los-conductores-de-su-flota-de-camiones/. [Último acceso: 09 03 2019].

[14] «Movil Data,» [En línea]. Available: https://movildata.com/localizador-de-vehiculos-por-gps-fleetmanager/. [Último acceso: 09 03 2019].

[15] «Drivin,» [En línea]. Available: https://www.driv.in/es/home. [Último acceso: 09 03 2019].

[16] «GestInFlot - Técnicas de Ingeniería,» 18 06 2018. [En línea]. Available: https://gesinflot.com/cuales-los-componentes-sistema-gestion-flotas/. [Último acceso: 09 03 2019].

[17] M. B. Capilla, *Desarrollo de un software para la gestión de correcciones RTCM de la red andaluza de posicionamiento para navegadores de código (GEORAP),* ETSI de la Universidad de Sevilla.

[18] S. Fernández, «Xataca Móvil : Beidou se hace internacional: así es el competidor chino del GPS,» 27 12 2018. [En línea]. Available: https://www.xatakamovil.com/conectividad/beidou-se-hace-internacional-asi-competidor-chino-gps. [Último acceso: 10 03 2019].

[19] R. B. Mundo, «BBC: Cómo funciona GLONASS y por qué el sistema de navegación ruso no tiene el éxito del GPS estadounidense,» 12 10 2017. [En línea]. Available: https://www.bbc.com/mundo/noticias-41596292. [Último acceso: 10 03 2019].

[20] «Universidad Internacional de Valencia,» 12 10 2016. [En línea]. Available: https://www.universidadviu.es/evolucion-la-red-comunicacion-movil-del-1g-al-5g/. [Último acceso: 16 03 2019].

[21] J. Catá, «El País,» 16 12 2018. [En línea]. Available: https://elpais.com/ccaa/2018/12/15/catalunya/1544903130_397095.html. [Último acceso: 16 03 2019].

[22] A. Vela, «Cualquier Cosa de Tecnología,» 12 02 2018. [En línea]. Available: https://cualquiercosadetecnologia.wordpress.com/2018/02/12/5g-la-nueva-generacion-de-telefonia-movil/. [Último acceso: 16 03 2019].

[23] Fairchild, «Datasheet LM350,» 09 210. [En línea]. Available: https://www.mouser.es/datasheet/2/308/LM350-1125023.pdf. [Último acceso: 03 05 2019].

[24] «Easy Eda - Online PCB Design Tool,» [En línea]. Available: https://easyeda.com. [Último acceso: 10 05 2019].

[25] Midtronics, «Midtronics: Advancing Battery Managment,» 02 03 2017. [En línea]. Available: https://midtronicseurope.com/assets/el-por-qu%C3%A9-y-el-c%C3%B3mo-del-mantenimiento-de-las-bater%C3%ADas-es-2mar2017.pdf. [Último acceso: 06 04 2019].

[26] T. Instruments, «Datasheet LM358P,» 09 2018. [En línea]. Available: http://www.ti.com/lit/ds/symlink/lm358.pdf. [Último acceso: 11 05 2019].

[27] I. Freescale Semiconductor, *FRDM-KL25Z User's Manual Rev 2.0,* 2013.

[28] SIMCom, «SIM800 Series_AT Command Manual_V1.09,» 2015.

[29] SIMCom, «SIM800 Series_Email_Application Nota V1.00,» 2013.

[30] MathWorks, «Documentation - ThingSpeak,» [En línea]. Available: https://es.mathworks.com/help/thingspeak/. [Último acceso: 13 05 2019].

[31] Autofácil, «Autofácil,» 24 03 2017. [En línea]. Available: https://www.autofacil.es/mecanica/2017/03/24/elegir-bateria-coche/37325.html. [Último acceso: 07 06 2019].

[32] «Ebay: Shield SIM808,» 2019. [En línea]. Available: https://www.ebay.co.uk/itm/SIM808-EVB-V3-2-Module-GSM-GPRS-GPS-Development-Board-SMA-GPS-Antenna-Arduino-/232633989231. [Último acceso: 11 05 2019].

[33] LG, «Datasheet LGC2601036SZ,» 10 03 2014. [En línea]. Available: https://docs-emea.rs-online.com/webdocs/1319/0900766b81319c5b.pdf. [Último acceso: 08 06 19].

[34] «Wikipedia,» 14 05 2018. [En línea]. Available: https://es.wikipedia.org/wiki/Sistemas_inteligentes_de_transporte. [Último acceso: 09 03 2019].

[35] T. Instrument, «Datasheet LM358P,» 09 2018. [En línea]. Available: http://www.ti.com/lit/ds/symlink/lm358.pdf. [Último acceso: 11 05 2019].

[36] SIMCom, «SIM808_Hardware Disign_V1.03,» 2016.

[37] I. Freescale Semiconductor, «Kinetis KL25 Sub-Family rev.5,» 2014.

9. Anexo I: Código desarrollado para plataforma KRDM-KL25Z

```
//Librerías
#include "mbed.h"

//Creación de puerto serie para comunicación con módulo SIM808

//Velocidad de transmisión, default = 9600
//Nº de bits por palabra, default = 8
//Paridad, default = Serial::None
//Nº de bits de parada, default = 1
//Pin tx: PTD3
//Pin rx: PTD2

Serial SIM808(PTD3,PTD2);

//Temporizadores principales

Timer t_main; //Temporizador global
Timer t_cont; //Temporizador de contacto contacto encendido
Timer t_cond; //Temporizador de tiempo de conducción

//Temporizadores auxiliares para control de funciones auxiliares
Timer t;
Timer t_aux;

//Entradas y salidas

DigitalIn contacto(D2); //J1 6 de la PCI - Entrada digital - señal de con
tacto
AnalogIn analog_value(A0);//J10, pin2 - Entrada analógica - tensión de ba
tería
DigitalOut power_modem(D4,0); //J1 10 de la PCI - Salida digital, estado
inicia               l apagado - habilitación de alimentación de SIM808
DigitalOut enable_modem(D5,0); //J1 12 de la PCI- Salida digital, estado
inicial apagado - habilitación de SIM808

//Variables globales

//Variables auxiliares para gestión de cadenas de texto
char aux1[40]="";
char aux2[120]="";
char aux3[120]="";

//Variable de estado
char i=0;

//Datos a enviar a Thingspeak
int data1=0; //Tensión de batería en mV
int data2=0; //Horas de contacto activo en minutos
int data3=0; //Horas en circuilación en minutos

//Flags de generación de alarmas
char flag1=0; //Alarma de tensión de batería
char flag2=0; //Alarma de tiempo de encendido

//Flag de contacto habilitado
```

```
char flag_cont=0;

//Estructura para almacenamiento de posición GPS
struct GPS{
  float latitude; //Latitud formato ddmm.mmmmmm
  float longitude; //Longitud formato ddmm.mmmmmm
};

//Definición de variables de posición tipo struct GPS
struct GPS location1; //Almacena el valor devuelto por get_gps
struct GPS location2; //Variable auxiliar para almacenamiento de posición

//Definición de funciones auxiliares

//Funciones para envío de comandos AT
void sendCmd(char *cmd);
int waitForResp(char *resp, int timeout);
int sendCmdAndWaitForResp(char *cmd, char *resp, int timeout);

//Funciones para gestión del módulo de comunicaciones y localización

char configure_connection(int timeout);

char send_email(char *message);
char write_email(char *message,int timeout);

char things_upload(int timeout);
char update_thingspeak();

char get_gps(int timeout);

void turn_on_modem();
void turn_off_modem();

//Definiciones globales

//Estructura de comando GET para actualización de Thingspeak
//API KEY del canal: NK5A451BKYH21TDM
#define get "GET https://api.thingspeak.com/update?api_key=NK5A451BKYH21T
DM&field1=\0"

//Parámetros de configuración de la SIM
#define pin "1961\r\n\0" //Código PIN
#define apn "internet" //Nombre del APN

//Parámetro de conexión al servidor SMTP2GO y envío de correos de alerta
#define mail_server "mail.smtp2go.com" //Servidor
#define mail_port "2525" //Puerto
#define user_smtp "alerta.mantenimiento.vehiculo@gmail.com" //Correo orig
en
#define password_smtp "alerta123" //Contraseña del correo
#define name_user_smtp "Alerta Vehiculo" //Usuario origen
#define mail_dst "sha@uoc.edu" //Correo destino
#define user_dst "Sara" //Usuario destino

/*-----------------------------------------------------------------------
--*/
/*------------------------INICIO DEL MAIN--------------------------------
--*/
/*-----------------------------------------------------------------------
--*/
int main()
```

```
{

    //Inicio del temporizador global
    t_main.start();

    while(1)
    {
        //Lectura del valor de tensión de batería
        // El valor leido está en el rango 0.0 a 1.0, conviento al rango
0 a 3.3 V y después al rango 0 a 14V y almaceno el valor en mV
        data1 = (int)(analog_value.read() * 3300 * 5.874);

        //Lectura del valor de tiempo de activación del contacto almacena
do en minutos
        //NOTA: en el mundo real el valor debería estar en horas, basta c
on dividir de nuevo por 60
        data2=(int)(t_cont.read()/(60));

        //Comprobación de la activación del contacto
        //Se detecta la activación del contacto si el contacto está activ
o y el flag de contacto anteriormente habilitado está a 0
        if(contacto == 1 && flag_cont==0)
        {
            //Inicio el temporizador de tiempo de contacto activo
            t_cont.start();
            //Inicio el temporizador de tiempo de circulación
            t_cond.start();
            //Actualizo el flag de activación de contacto a 1
            flag_cont=1;
            //Enciendo y habilito el módulo SIM808
            turn_on_modem();
            //Obtengo la posición GPS inicial, queda almacenada en locati
on1
            get_gps(120);
            //Apago el módulo SIM808
            turn_off_modem();

            //Guardo la posición actual en la variable auxiliar location2
            location2=location1;

            //Para simular movimiento del vehíuclo en las pruebas en está
tico se sustituye la línea anterior por estas dos siguientes
            //que fuerzan a que la posición inicial obtenida sea diferent
e a la final
            //location2.longitude=08.000000;
            //location2.latitude=24.003579;
        }

        //Comprobación de la desactivación del contacto
        //Se detecta la desactivación del contacto si el contacto está
desactivado y el flag de contacto anteriormente habilitado está a 1
        if(contacto == 0 && flag_cont==1)
        {
            //Reset del temporizador global
            t_main.reset();
            //Pausa de los temporizadores de tiempo de conducción y tiemp
o de  contacto activo
            t_cont.stop();
            t_cond.stop();

            //Encendido y hablitación de módulo SIM808
            turn_on_modem();
```

```
            //Obtención de la posición GPS final, queda almacenada en loc
ation1
            get_gps(120);

            //Apagado del módulo SIM808
            turn_off_modem();

            //Actualización del tiempo de contacto encendido en minutos
            //En la aplicación real el tiempo debería ser en horas, basta
con  dividir de nuevo entre 60
            data2=(int)(t_cont.read()/(60));

            //Desactivación del flag de contacto anteriormente habilitado
            flag_cont=0;

            //Se compara la posición final almacenada en location2 con la
     inicial almacenada en location1.
            //El formato de longitud y latitud es ddmm.mmmmmm
            //Si la variación es de más de 0.0001 considero que me he
desplazado y que he estado circulando
            if(((abs(location1.longitude-
location2.longitude))>0.001)||((abs(location1.latitude-
location2.latitude))>0.001))
            {
                //Actualizo el tiempo de circulación en minutos
                //En la aplicación real el tiempo debería ser en horas, b
asta  con dividir de nuevo entre 60
                data3=data3+((int)(t_cond.read()/(60)));
            }

            //Reset del temporizador de tiempo de conducción
            t_cond.reset();

            //Encendido y habilitación del módulo SIM808
            turn_on_modem();

            //Actualización de thingspeak con los datos almacenados en da
ta1, data2 y data3
            update_thingspeak();

            //Apagado del módulo SIM808
            turn_off_modem();

            //Comprobación de si el tiempo de vehículo circulando supera
el    umbral establecido.
            //En la realidad el umbral debería ser de 1875 horas, como se
     indica en la memoria.
            //Para los test se establece un valor de 15 minutos.
            if(data3>15)
            {
                    //Encendido y habilitación del módulo SIM808
                    turn_on_modem();
                    //Envío de correo electrónico de alerta
                    send_email("Timpo de circulacion del vehiculo 1 super
ior al umbral establecido.");
                    //Apagado del módulo SIM808
                    turn_off_modem();
            }
        }

        //Comprobación de si la tensión de batería medida es inferior al
```

```
umbral establecido de 11.9V
        if(data1<11900)
        {
            //Genero la alerta y actualizo los datos sólo si el flag de a
lerta enviada no está activo, es decir, si la alerta no se ha enviado dur
ante el     último ciclo del
            //temporizador global
            if(flag1==0)
            {
                //Reset del temporizador global
                t_main.reset();
                //Encendido y habilitación del módulo SIM808
                turn_on_modem();
                //Actualización de datos de Thingspeak
                update_thingspeak();
                //Apagado del módulo SIM808
                turn_off_modem();

                //Encendido y habilitación del módulo SIM808
                turn_on_modem();
                //Envío de correo electrónico de alerta
                send_email("Tension de bateria del vehiculo 1 inferior al
umbral establecido de 11.9V.");
                //Apagado del módulo SIM808
                turn_off_modem();

                //Actualizo del flag a 1, esto evita que se envíen avisos
a    cada lectura una vez se ha detectado el fallo
                flag1=1;
            }
        }

        //Comprobación de si el tiempo de contacto encendido supera en un
25%  el máximo de horas de circulación mando aviso
        //En la realidad el umbral debería ser de 1.25 x 1875 horas, como
se   indica en la memoria.
        //Para los test se establece un valor de 18 minutos.
        if(data2>18)
        {
            //Genero la alerta y actualizo los datos sólo si el flag de a
lerta enviada no está activo, es decir, si la alerta no se ha enviado dur
ante el     último ciclo del temporizador global
            if(flag2==0)
            {
                //Reset del temporizador global
                t_main.reset();
                //Encendido y habilitación del módulo SIM808
                turn_on_modem();
                //Actualización de los datos de Thingspeak
                update_thingspeak();
                //Apagado del módulo SIM808
                turn_off_modem();

                //Encendido y habilitación del módulo SIM808
                turn_on_modem();
                //Envío de correo de alerta
                send_email("Timpo de encendido del vehiculo 1 superior al
umbral establecido.");
                //Apagado del módulo SIM808
                turn_off_modem();

                //Actualizo del flag a 1, esto evita que se envíen avisos
```

```
a    cada lectura una vez se ha detectado el fallo
                flag2=1; //Evito que se envíen avisos a cada lectura una
vez se ha detectado el fallo
            }
        }

        //Comporobación de que el temporizador global no ha superado el t
iempo máximo establecido entre actualizaciones del Thingspeak
        //En la aplicación real el tiempo mínimo entre actualizaciones es
de 24 horas
        //Para el modo test se establece un tiempo mínimo entre actualiza
ciones de 15 minutos (900 segundos)
        if(t_main.read() > 900)
        {
            //Encendido y hablitación del módulo SIM808
            turn_on_modem();
            //Actualización de datos de Thingspeak
            update_thingspeak();
            //Apagado del módulo SIM808
            turn_off_modem();

            //Reset del temporizador global
            t_main.reset();

            //Reseteo los flags para que vuelvan a enviarse las alertas
            flag1=0;
            flag2=0;

        }

        wait(1);
    }

}

/*------------------------ FIN DEL MAIN ---------------------------------
----*/
/*-----------------------------------------------------------------------
----*/

/*-----------------------------------------------------------------------
----*/
/*--------------------FUNCIONES AUXILIARES------------------------------
----*/

/*------------------Funciones para envío de comandos AT -----------------
----*/

//Envío de comando parametrizado como cadena
void sendCmd(char *cmd)
{
    SIM808.puts(cmd);
}

//Función que espera la respuesta de un comando. Esta debe ser identica a
la cadena "resp" y ser recibida ante de un tiempo timeout
//si todo sale bien retorna un cero si algo sale mal (no se parece o se d
emora mucho )retorna -1
int waitForResp(char *resp, int timeout)
{
    int len = strlen(resp);
    int sum=0;
```

```
    //Inicio un temporizador auxiliar
    t.start();

    while(1) {
        //Comprueba que la entrada es "leible"
        if(SIM808.readable()) {
            char c = SIM808.getc();
            sum = (c==resp[sum]) ? sum+1 : 0;//Sum se incrementa o se hac
e cero segun c
            if(sum == len)break;  //Obtiene la cadena completa a buscar,
acaba y sale
        }
        if(t.read() > timeout) {  //Si he sobrepasado mi timeout devuelvo
error

            //Pauso y reseteo el temporizador auxiliar
            t.stop();
            t.reset();

            return -1;
        }
    }

    //Pauso y resetao el temporizador auxiliar
    t.stop();
    t.reset();

    while(SIM808.readable())
    {
        char c = SIM808.getc();
    }

    return 0;
}

//Funcion que envia un comando y espera la respuesta
//Si todo sale bien retorna un cero
//Si hay algún error o se sobrepasa el timeout establecido retorna un -1
int sendCmdAndWaitForResp(char *cmd, char *resp, int timeout)
{
    //Llamada a función de envío de comando
    sendCmd(cmd);
    //Llamada a función de espera de respuesta
    return waitForResp(resp,timeout);
}


/*-------------------------Obtención de posición GPS --------------------
--*/
//Función para obtención de posición gps
//Si no obtiene la posición y se sobrepasa el timeout devuelve error 1
//Si todo va bien devuelve 0 y la posición leida queda almacenada en loca
tion1
char get_gps(int timeout)
{
    //Inicio de temporizador auxiliar
    t_aux.start();
    //índice a 0
    i=0;

    //Comprobación del estado del módulo SIM808
```

```
    while (0!=sendCmdAndWaitForResp("AT\r\n","OK", 2))//Si va bien devuel
ve 0, si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Activación de alimentación del GPS
    while (0!=sendCmdAndWaitForResp("AT+CGPSPWR?\r\n","+CGPSPWR: 1", 2))/
/Si va bien devuelve 0, si no -1
    {
        sendCmdAndWaitForResp("AT+CGPSPWR=1\r\n","OK", 2);
        wait(1);

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Comprobación de que la localización ha sido fijada
    while (0!=sendCmdAndWaitForResp("AT+CGPSSTATUS?\r\n","+CGPSSTATUS: Lo
cation 3D Fix", 2)) //Si va bien devuelve 0, si no -1
    {
        wait(10);

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Reset de variable global aux3
    strcpy(aux3,"\0");

    //Solicitud de posición
    sendCmd("AT+CGPSINF=0\r\n");

    //Lectura de la respuesta y almacenaje en aux3
    i=0;

    while(1)
    {
        if(SIM808.readable())
        {
```

```
            aux3[i] = SIM808.getc();

            if(i>>0)
            {
                if(aux3[i-
1] == 'O' && aux3[i] == 'K')break;//Leido el OK final de la respuesta se
sale del bucle
            }

            i++;

        }
        if(t_aux.read()>timeout){// Si he sobrepasado mi timeout devuelvo
error
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    //Contateno caracter de fin de cadena
    strcat(aux3,"\0");

    wait(2);

    //La respuesta dispone de varios campos: hora, fecha, longitud, latit
ud, número de satélites...
    //Separo los campos que me interesan: longitud y latitud
    //Los distintos campos se separan por comas "," por lo que busco este
caracter para acotar los capmos

    //Separo el inicio de la respuesta AT+CGPSINF=0 +CGPSINF: 0
    i=0;
    char *ptr;
    ptr = strtok(aux3, ",");    // Primera llamada

    //Estraigo el campo latitud
    ptr = strtok( NULL, "," );
    strcpy(aux1,ptr);
    strcat(aux1,"\0");
    location1.latitude=atof(aux1);

    //Extraigo el campo longitud
    ptr = strtok( NULL, "," );
    strcpy(aux1,ptr);
    strcat(aux1,"\0");
    location1.longitude=atof(aux1);

    //Apago el GPS
    while (0!=sendCmdAndWaitForResp("AT+CGPSPWR?\r\n","+CGPSPWR: 0", 2))/
/Si va bien devuelve 0, si no -1
    {
        sendCmdAndWaitForResp("AT+CGPSPWR=0\r\n","OK", 2);
        wait(1);

    }

    //Pauso y reseteo el temporizador auxiliar
    t_aux.stop();
    t_aux.reset();

    return 0;
```

```
}

/*----------------Configuración de conexión-----------------------------
----*/
//Función de configuración de la conexión a internet mediante el módulo S
IM808
//Si todo va bien devuelve 0
//Si hay algún error o finalizar el timeout devuelve 1
char configure_connection(int timeout)
{
    //Inicio de temporizador auxiliar
    t_aux.start();

    //Comprobación del estado del módulo SIM808
    while(0!=sendCmdAndWaitForResp("AT\r\n","OK",2))//Si va bien devuelve
0, si no -1
    {

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }

    }

    wait(2);

    //Introducción del PIN de la tarjeta SIM
    strcpy(aux1,pin);
    strcpy(aux2,"AT+CPIN=");
    strcat(aux2,aux1);

    while(0!=sendCmdAndWaitForResp("AT+CPIN?\r\n","+CPIN: READY",5))//Si
va    bien devuelve 0, si no -1
    {
        sendCmdAndWaitForResp(aux2,"OK",5);

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }

    }

    wait(2);

    //Desactivación del contesto GPRS PDP
    while(0!=sendCmdAndWaitForResp("AT+CIPSHUT\r\n","SHUT OK",65)) //Si v
a bien devuelve 0, si no -1
    {
        wait(1);
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
```

```
            return 1;
        }
    }

    wait(2);

    //Configuración de conexión IP única
    while(0!=sendCmdAndWaitForResp("AT+CIPMUX?\r\n","+CIPMUX: 0",5))//Si
va    bien devuelve 0, si no -1
    {
        sendCmdAndWaitForResp("AT+CIPMUX=0\r\n","OK",5);
        wait(1);

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Agregación al servicio GPRS
    while(0!=sendCmdAndWaitForResp("AT+CGATT?\r\n","+CGATT: 1",10))//Si v
a bien devuelve 0, si no -1
    {
        sendCmdAndWaitForResp("AT+CGATT=1\r\n","OK",10);
        wait(1);

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Inicio de taras y establecimiento del APN
    strcpy(aux1,apn);
    strcpy(aux2,"AT+CSTT=");
    strcat(aux2,aux1);
    strcat(aux2,"\r\n\0");

    while(0!=sendCmdAndWaitForResp(aux2,"OK",5))//Si va bien devuelve 0,
si no -1
    {

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }

    }
```

```
    wait(2);

    //Levantamiento de la conexión inalámbrica con GPRS
    while(0!=sendCmdAndWaitForResp("AT+CIICR\r\n","OK",85))//Si va bien
devuelve 0, si no -1
    {

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }

    }

    wait(2);

    //Verificación de que el módulo tiene IP asignada
    //Este comando responde con la IP local o con ERROR
    //La verificación para por comprobar que no devuelve error pues desco
nozco la IP que me van a asignar
    while(0==sendCmdAndWaitForResp("AT+CIFSR\r\n","ERROR",5)) //Si va bie
n     devuelve 0, si no -1
    {

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }

    }

    wait(2);

    //Pausa y reset del temporizador auxiliar
    t_aux.stop();
    t_aux.reset();

    return 0;
}

/*-------------------Correo electrónico---------------------------------
----*/
//Función que configura y envía un correo electrónico con el contenido re
cibido
//El origen y destinatario viene predeterminado en la programación
//Si todo va bien devuelve 0, si hay algún error o se acaba el timeout de
vuelve 1
char send_email(char *message)
{
    //Variable auxiliar contador
    char count=0;

    //Configuración de conexión GPRS
    //Se realizan un máximo de 3 intentos
    while(configure_connection(120)!=0) //Devuelve 0 si va bien y 1 si va
mal
```

```
    {
        count++;
        wait(1);
        if(count>=3)
        {
          return 1;
        }
    }

    count=0;

    //Escritura y envío del correo
    //Se realizan un máximo de 3 intentos
    while(write_email(message,240)!=0) //Devuelve 0 si va bien y 1 si va
mal
    {
        count++;
        wait(1);
        if(count>=3)
        {
          return 1;
        }

    }

    return 0;

}

/*--------------Función de escritura y envío del email ------------------
--*/
//Función que envía los comandos necesarios para el envío del correo
electrónico, previa configuración de la conexión GPRS
//Devuelve 0 si va bien y 1 si hay algún error o finaliza el timeout
char write_email(char *message,int timeout)
{
    //Inicio de temporizador auxiliar
    t_aux.start();

    //Configuración de tipo de conexión GPRS
    while(0!=sendCmdAndWaitForResp("AT+SAPBR=3,1,\"Contype\",\"GPRS\"\r\n
","OK",2)) //Si va bien devuelve 0, si no -1
    {

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }

    }

    wait(2);

    //Configuration of APN
    strcpy(aux2,"AT+SAPBR=3,1,\"APN\",\"");
    strcpy(aux1,apn);
    strcat(aux2,aux1);
    strcat(aux2,"\"\r\n\0");
```

```
    while(0!=sendCmdAndWaitForResp(aux2,"OK",2)) //Si va bien devuelve 0,
si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Apertura del portador
    while(0!=sendCmdAndWaitForResp("AT+SAPBR=1,1\r\n","OK",2)) //Si va bi
en    devuelve 0, si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Petición de portador
    while(0!=sendCmdAndWaitForResp("AT+SAPBR=2,1\r\n","+SAPBR: 1,1,",2))/
/Si va bien devuelve 0, si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Establecimiento de identificador de perfil de portador Email a 1
    while(0!=sendCmdAndWaitForResp("AT+EMAILCID=1\r\n","OK",2)) //Si va b
ien devuelve 0, si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Establecimiento del timeout para la respuesta del servidor SMTP a 2
     minutos
    while(0!=sendCmdAndWaitForResp("AT+EMAILTO=120\r\n","OK",2)) //Si va
```

```
bien  devuelve 0, si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Configuración del mensaje

    //Establecimiento del servidor SMTP y su puesto
    //Servidor: mail.smtp2go.com
    //Puerto: 2525
    strcpy(aux2,"AT+SMTPSRV=\"");
    strcpy(aux1,mail_server);
    strcat(aux2,aux1);
    strcat(aux2,"\",");
    strcpy(aux1,mail_port);
    strcat(aux2,aux1);
    strcat(aux2,"\r\n\0");

    while(0!=sendCmdAndWaitForResp(aux2,"OK",10))//Si va bien devuelve 0,
si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Establecimiento de usuario y contraseña para identificación SMTP
    //Usuario: alerta.mantenimiento.vehiculo@gmail.com
    //Contraseña: alerta123

    strcpy(aux2,"AT+SMTPAUTH=1,\"");
    strcpy(aux1,user_smtp);
    strcat(aux2,aux1);
    strcat(aux2,"\",\"");
    strcpy(aux1,password_smtp);
    strcat(aux2,aux1);
    strcat(aux2,"\"\r\n\0");

    while(0!=sendCmdAndWaitForResp(aux2,"OK",10))//Si va bien devuelve 0,
si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
```

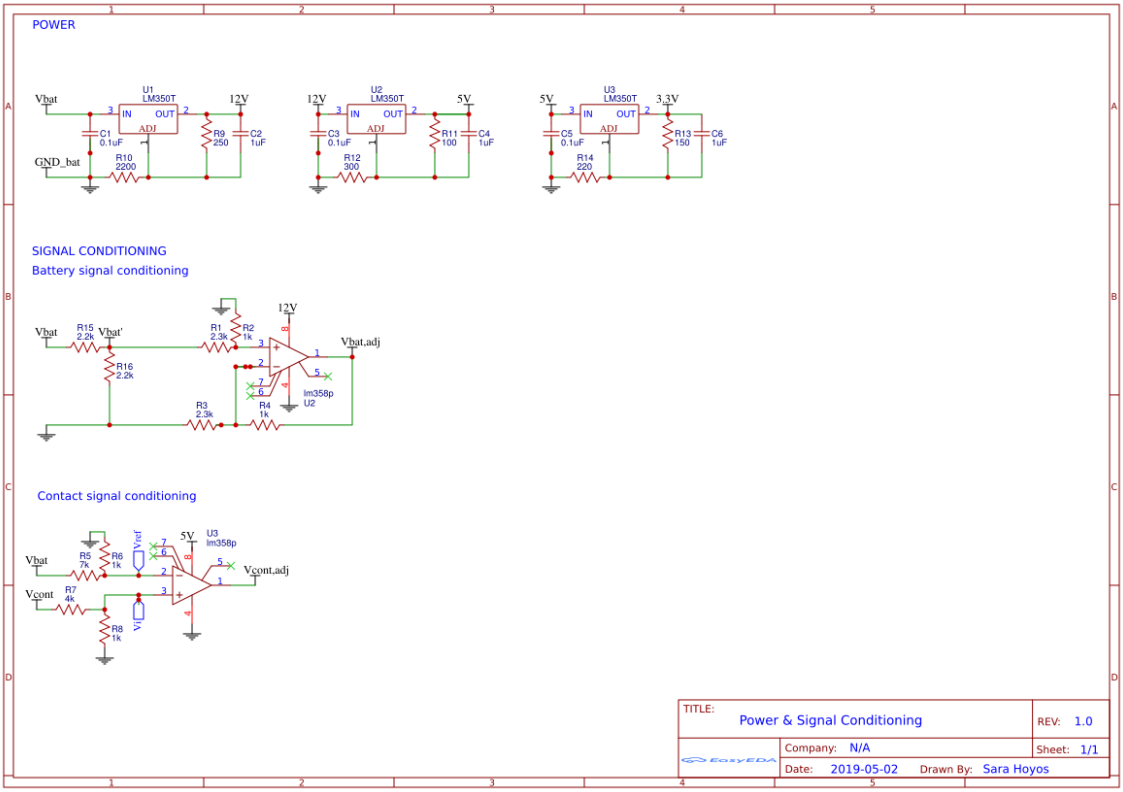
```
    }

    wait(2);

    //Establecimiento de dirección y nombre del originador del mensaje
    //Dirección origen: alerta.mantenimiento.vehiculo@gmail.com
    //Nombre: Alerta Vehiculo
    strcpy(aux2,"AT+SMTPFROM=\"");
    strcpy(aux1,user_smtp);
    strcat(aux2,aux1);
    strcat(aux2,"\",\"");
    strcpy(aux1,name_user_smtp);
    strcat(aux2,aux1);
    strcat(aux2,"\"\r\n\0");

    while(0!=sendCmdAndWaitForResp(aux2,"OK",10))//Si va bien devuelve 0,
si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Establecimiento de la dirección y nombre del destinatario en el cam
po    "TO"
    strcpy(aux2,"AT+SMTPRCPT=0,0,\"");
    strcpy(aux1,mail_dst);
    strcat(aux2,aux1);
    strcat(aux2,"\",\"");
    strcpy(aux1,user_dst);
    strcat(aux2,aux1);
    strcat(aux2,"\"\r\n\0");

    while(0!=sendCmdAndWaitForResp(aux2,"OK",10))//Si va bien devuelve 0,
si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Establecimiento del asunto del mensaje
    while(0!=sendCmdAndWaitForResp("AT+SMTPSUB=\"Test\"\r\n","OK",10)) //
Si va bien devuelve 0, si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
```

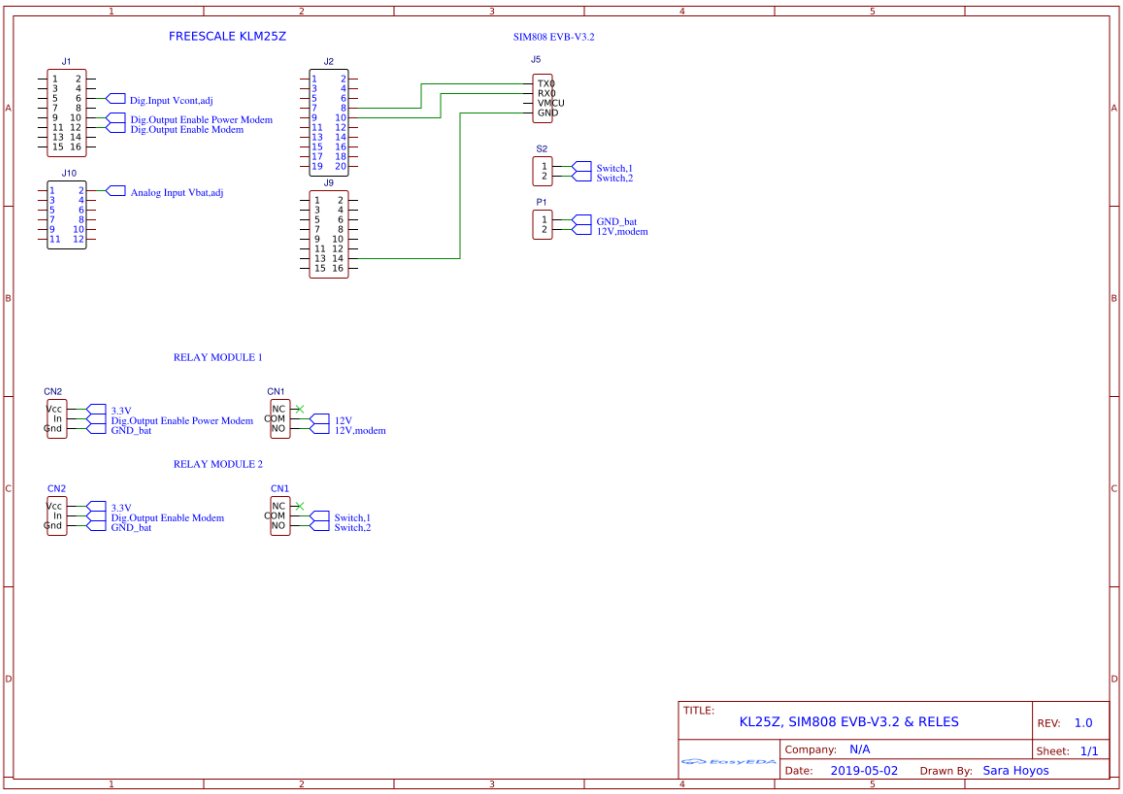
```
            return 1;
        }
    }

    wait(2);

    //Establecimiento de la longitud del cuerpo del mensaje
    strcpy(aux2,"\0");//Borro el contenido de aux2
    sprintf(aux2, "AT+SMTPBODY=%d\r\n\0",strlen(message));

    while(0!=sendCmdAndWaitForResp(aux2,"DOWNLOAD",10))//Si va bien devue
lve 0, si no -1
    {
        wait(1);

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }

    }

    wait(2);

    //Establecimiento del cuerpo del mensaje
    while(0!=sendCmdAndWaitForResp(message,"OK",10))//Si va bien devuelve
0, si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(2);

    //Envío del mensaje
    while(0!=sendCmdAndWaitForResp("AT+SMTPSEND\r\n","OK\r\n\r\n+SMTPSEND
: 1",10)) //Si va bien devuelve 0, si no -1
    {
        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    //Pausa y reset del temporizador auxiliar
    t_aux.stop();
    t_aux.reset();

    return 0;
}
/*--------Función de envío de dato a thingspeak-------------------------
```

```
*/
//Función para actualización de datos en la plataforma Thingspeak con los
datos almacenados en data1, data2 y data3
//Si todo va bien devuelve 0, si hay algún error o finaliza el timeout de
vuelve 1
char things_upload(int timeout)
{
    //Inicio del temporizador auxiliar
    t_aux.start();

    //Conexión a thingspeak
    while(0!=sendCmdAndWaitForResp("AT+CIPSTART=\"TCP\",\"api.thingspeak.
com\",80\r\n","OK",5)) //Si va bien devuelve 0, si no -1
    {
        wait(1);

        if(t_aux.read() > timeout)//Si he sobrepasado mi timeout devuelvo
error
        {
            t_aux.stop();
            t_aux.reset();
            return 1;
        }
    }

    wait(3);

    //Establezco el comando a enviar para determinar su longitud
    strcpy(aux2,get);

    strcpy(aux1,"\0"); //Borro el contenido de aux1
    sprintf(aux1,"%d",data1);
    strcat(aux2,aux1);

    strcpy(aux1,"\0"); //Borro el contenido de aux1
    sprintf(aux1,"%d",data2);
    strcat(aux2,"&field2=");
    strcat(aux2,aux1);

    strcpy(aux1,"\0"); //Borro el contenido de aux1
    sprintf(aux1,"%d\0",data3);
    strcat(aux2,"&field3=");
    strcat(aux2,aux1);

    strcat(aux2,"\r\n\0"); //El cmd queda en aux2

    strcpy(aux1,"\0"); //Borro el contenido de aux1

    //Indico la longitud del comando a enviar
   sprintf(aux1,"AT+CIPSEND=%d\r\n\0",strlen(aux2));
   sendCmdAndWaitForResp(aux1,"",5);

    wait(2);

    //Envío comando con datos
    sendCmdAndWaitForResp(aux2,"",5);

    wait(3);

    //Pausa y reset del temporizador auxiliar
    t_aux.stop();
    t_aux.reset();
```

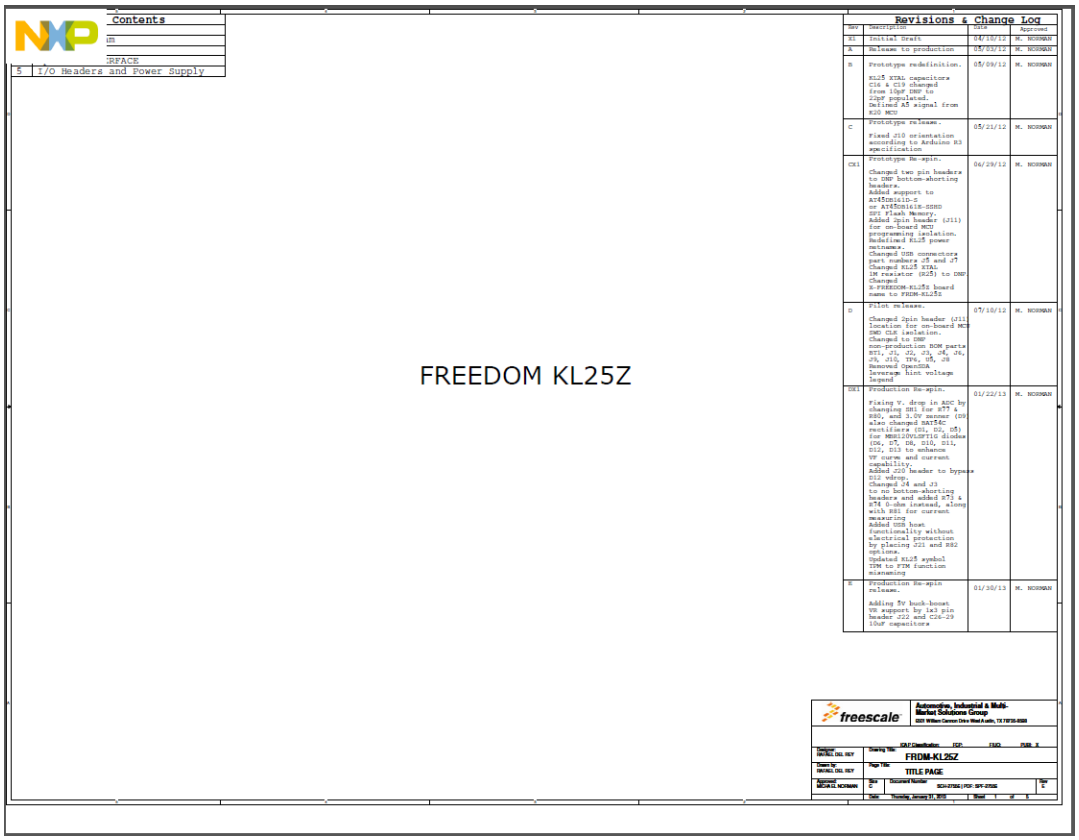
```
    return 0;

}
//Actualiza thingspeak con los datos guardados en las variables globales
tipo  data1, 2...
char update_thingspeak()
{
    char count=0;

    //Configuración de a conexión GPRS
    while(configure_connection(120)!=0) //Devuelve 0 si va bien y 1 si va
mal
    {
        count++;
        wait(1);
        if(count>=3)
        {
          return 1;
        }
    }
    count=0;

    //Actualización de datos
    while(things_upload(120)!=0)//Devuelve 0 si va bien y 1 si va mal
    {
        count++;
        wait(1);
        if(count>=3)
        {
          return 1;
        }
    }

    return 0;

}

/*-------------Enciende y habilita el modem-----------------------------
----*/
//Función de encendido y habilitación del módulo SIM808
void turn_on_modem()
{
    //Enciendo alimentación del módem
    power_modem=1;
    wait(1);
    //Envío unp pulso para habilitar el módem
    enable_modem=1;
    wait(2);
    enable_modem=0;

}
/*-------------Apaga el modem ------------------------------------------
-*/
//Función para apagado del modem
void turn_off_modem()
{
    //Apago alimentación del módem
    power_modem=0;
    wait(1);

}
```

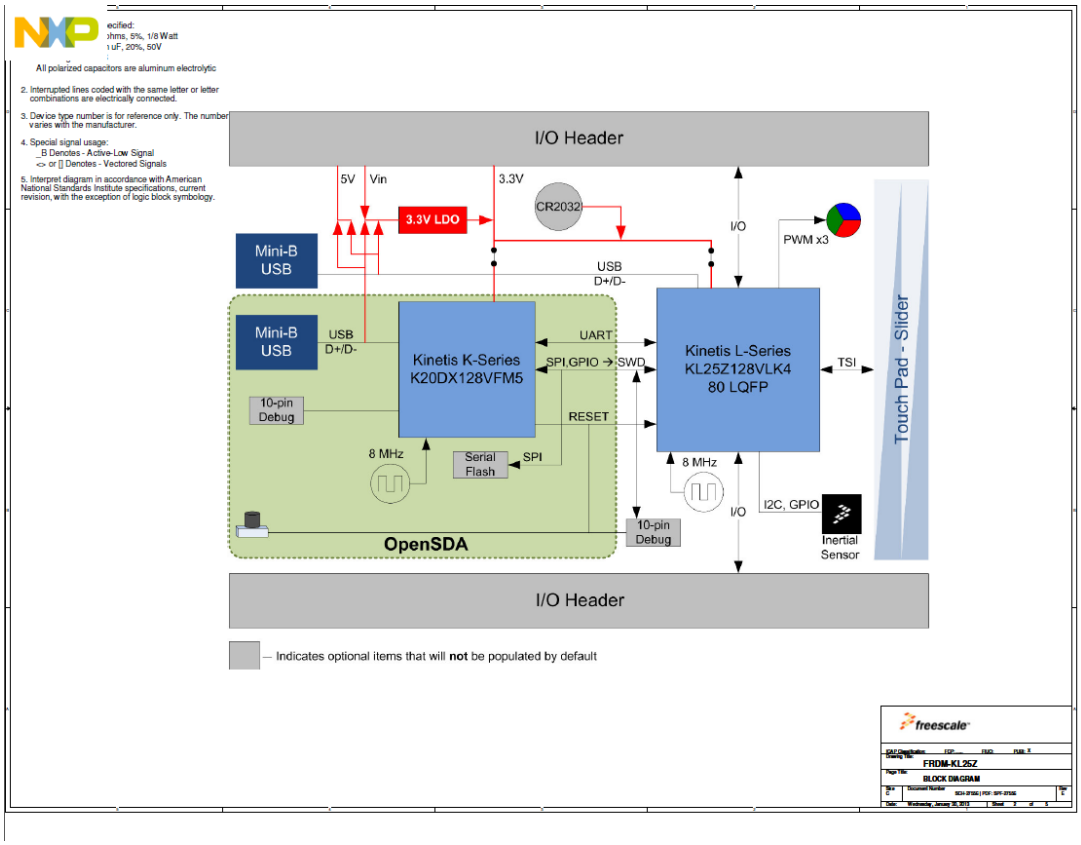
10. Anexo II: Esquemático de la plataforma embarcada

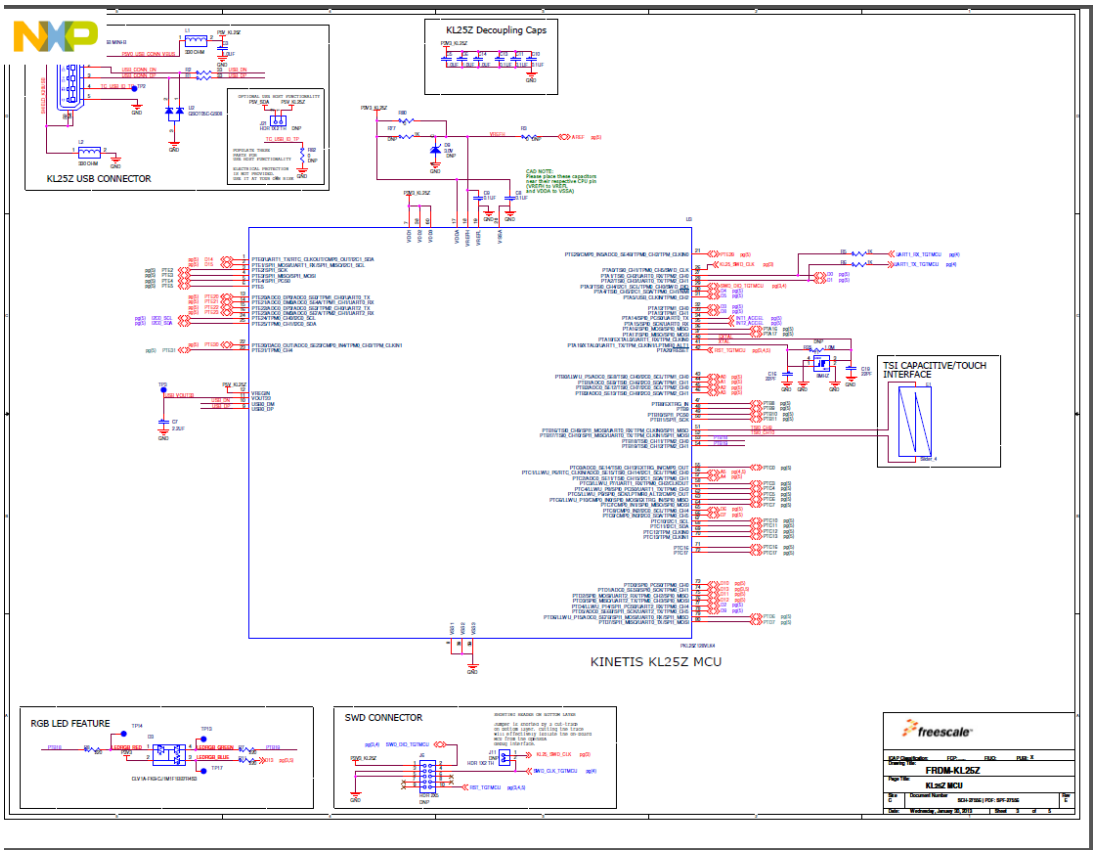


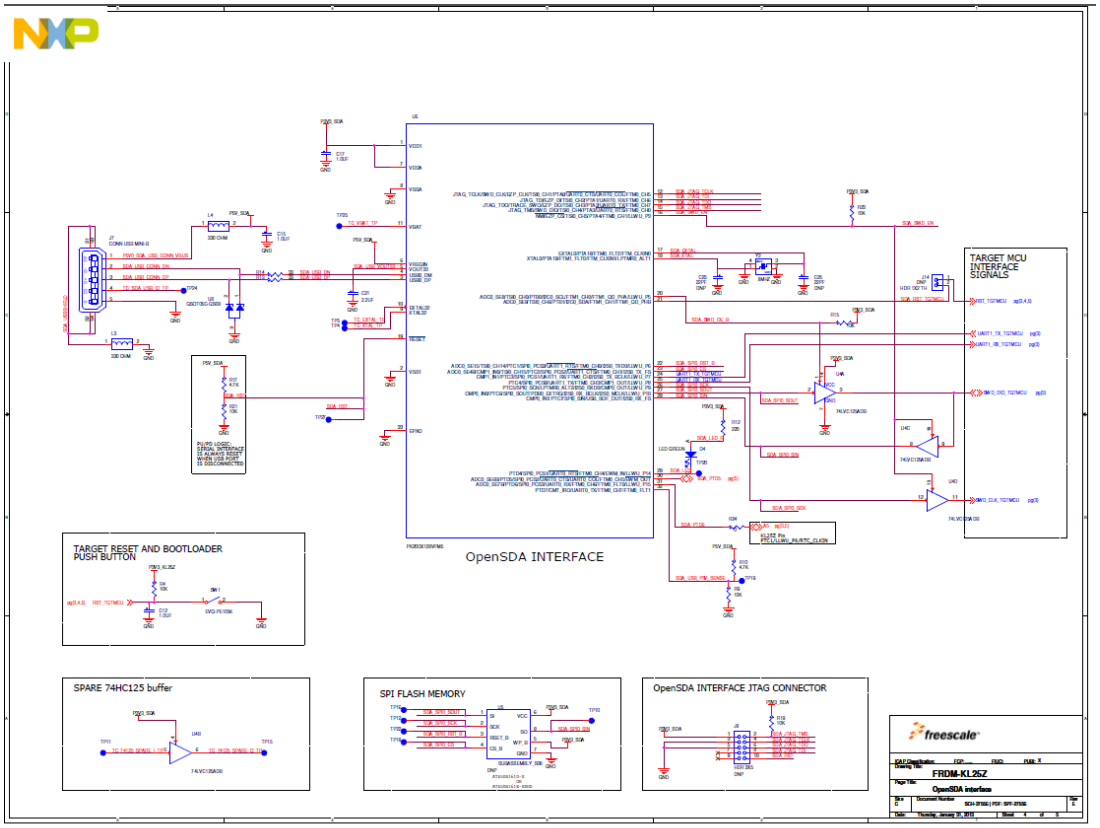


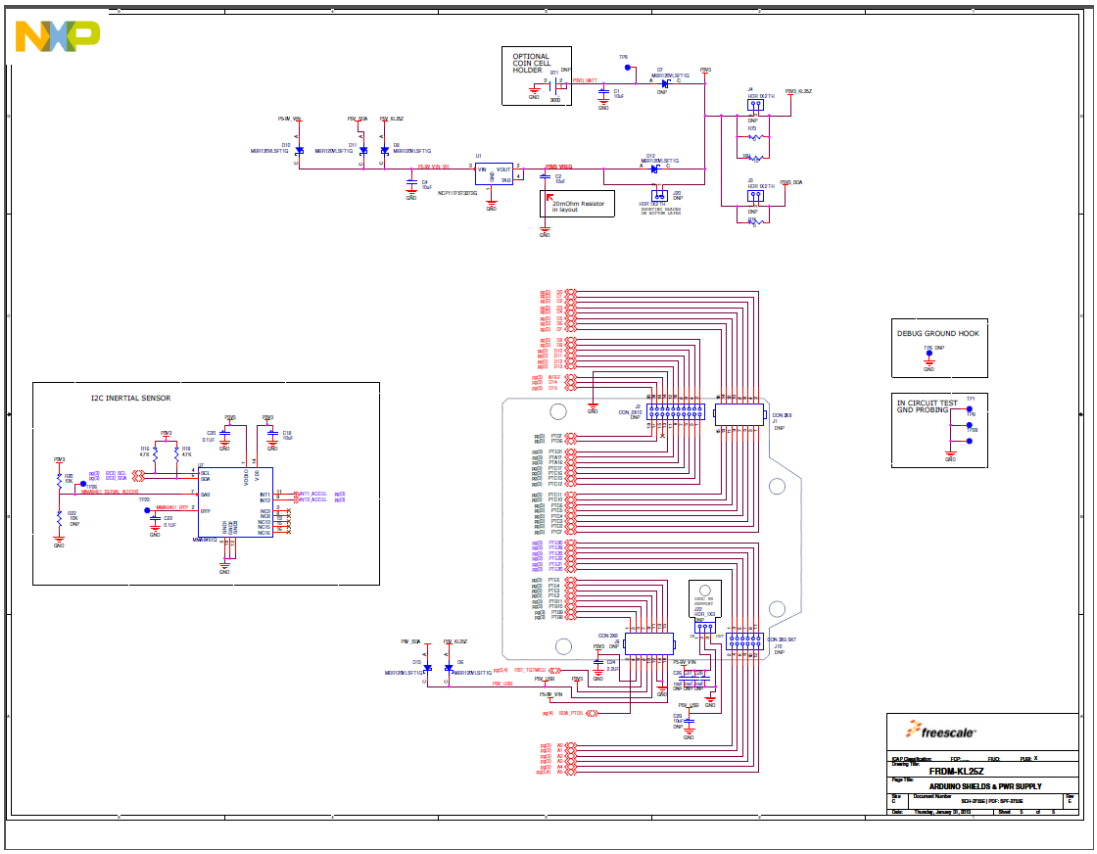
11. Anexo III: Esquemático de la plataforma KL25Z[i]






KL25Z USB CONNECTOR
KL25Z Decoupling Caps
TSI CAPACITIVE/TOUCH INTERFACE
KINETIS KL25Z MCU
RGB LED FEATURE
SWD CONNECTOR
freescale
FRDM-KL25Z
KL25Z MCU


TARGET MCU INTERFACE SIGNALS
OpenSDA INTERFACE
TARGET RESET AND BOOTLOADER PUSH BUTTON
SPARE 74HC125 buffer
SPI FLASH MEMORY
OpenSDA INTERFACE JTAG CONNECTOR
freescale
FRDM-KL25Z
OpenSDA interface



[i] Esquemáticos oficiales extraídos de la web de NXP:
https://www.nxp.com/downloads/en/schematics/FRDM-KL25Z_SCH_REV_E.pdf