UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE MASTER EN INGENIERÍA INDUSTRIAL

Desarrollo de un sistema de monitorización de un motor de corriente continua con sensores inteligentes

AUTOR: SANTIAGO RODRÍGUEZ DÍAZ

TUTOR: FRANCISCO JAVIER GARCIA CASADO

COTUTORA: YIYAO YE LIN

Curso Académico: 2016-17

RESUMEN

El presente TFM fue desarrollado en el Departamento de Ingeniería Electrónica de la UPV. Nace de la necesidad de introducir los sensores inteligentes a los alumnos del Máster de Ingeniería Industrial dado el creciente uso de estos sensores en el área industrial. Para ello, se dispone de un motor de corriente continua, cuya velocidad es controlada por un sistema electrónico. La carga se simula mediante resistencias eléctricas conectadas al generador acoplado al motor. El sistema motor/generador incorpora salidas analógicas que informan sobre velocidad de giro, corriente y tensión en borne del motor. Además, se quieren obtener lecturas de temperatura y vibraciones mediante sensores inteligentes con salida digital.

El objetivo es el desarrollo de un sistema de monitorización inalámbrica multivariable del sistema motor/generador utilizando sensores inteligentes. El sistema debe transmitir los datos a un PC vía cableado o de forma inalámbrica, y enviar los datos a una plataforma de Nube mediante GPRS y alertas SMS vía GSM.

Primero se han seleccionado los componentes hardware del sistema: sensores de temperatura y vibraciones con salida digital, microcontrolador, módulo de comunicación GSM/GPRS. Subsecuentemente se ha diseñado un circuito hardware para adaptar los niveles de tensión de las señales analógicas del sistema motor/generador a la entrada del microcontrolador. Posteriormente, se ha desarrollado una placa de circuito impreso que incluye el circuito adaptador y el bluetooth.

Asimismo, un programa desarrollado en el microcontrolador recoge la información sobre las señales analógicas y permite leer la temperatura y las vibraciones por puerto SPI. El microcontrolador transmite estos datos al PC vía USB o de forma inalámbrica, el cual ejecuta un programa de Labview para tratamiento de los datos. También se ha dotado al sistema de envío de datos a la Nube vía GPRS y SMS de alarma. Se han realizado los ensayos para validar este sistema de monitorización.

Palabras Clave: Monitorización industrial, motor, sensores inteligentes, programación microcontrolador, Transmisión inalámbrica, comunicación industrial.

RESUM

El present TFM va ser desenvolupat al Departament d'Enginyeria Electrònica de la UPV. Naix de la necessitat d'introduir el sensors intel·ligents als alumnes del Màster d'Enginyeria Industrial a causa del creixent ús d'estos sensors a l'àrea industrial. Per a això, es disposa d'un motor de corrent continu, velocitat del qual, és controlada per un sistema electrònic. La càrrega es simula mitjançant resistències elèctriques connectades al generador acoblat al motor. El sistema motor / generador incorpora eixides analògiques les quals informen sobre velocitat de gir, corrent i tensió en born del motor. A més, es volen obtindre lectures de temperatura i vibracions mitjançant sensors intel·ligents amb eixida digital.

L'objectiu és el desenvolupament d'un sistema de monitorització sense fils multivariable del sistema motor / generador utilitzant sensors intel·ligents. El sistema ha de transmetre les dades a un PC via cablejat o sense fils, i enviar les dades a una plataforma de Núvol mitjançant GPRS i alertes SMS via GSM.

Primer s'han seleccionat els components maquinari del sistema: sensors de temperatura i vibracions amb eixida digital, microcontrolador, mòdul de comunicació GSM / GPRS. Subsequentment, s'ha dissenyat un circuit maquinari per adaptar els nivells de tensió dels senyals analògiques del sistema motor / generador a l'entrada del microcontrolador. Posteriorment, s'ha desenvolupat una placa de circuit imprès que inclou el circuit adaptador i el bluetooth.

Així mateix, un programa desenvolupat en el microcontrolador recull la informació sobre els senyals analògiques i permet llegir la temperatura i les vibracions per port SPI. El microcontrolador transmet estes dades al PC via USB o sense fils, el qual executa un programa de Labview per a tractament de les dades. També s'ha dotat el sistema d'enviament de dades al Núvol via GPRS i SMS d'alarma. S'han realitzat els assajos per validar este sistema de monitorització.

Paraules Clau: Monitorització industrial, motor, sensors intel·ligents, programació microcontrolador, Transmissió sense fil, comunicació industrial.

ABSTRACT

The present TFM was developed in the Department of Electronic Engineering of the UPV. It is born of the need to introduce intelligent sensors to the students of the Master of Industrial Engineering given the growing use of these sensors in the industrial area. To make this tangible, it has a DC motor, whose speed is controlled by an electronic system. The load is simulated by electrical resistances connected to the generator coupled to the motor. The motor / generator system incorporates analogue outputs that report speed, current and voltage at the motor terminal. Furthermore, we want to obtain temperature and vibration readings through intelligent sensors with digital output.

The aim is the development of a multivariable wireless monitoring system of the motor / generator system using smart sensors. The system must transmit the data to a PC via wiring or wirelessly and send the data to a Cloud platform through GPRS and SMS alerts via GSM.

First the hardware components of the system have been selected: temperature and vibration sensors with digital output, microcontroller, GSM / GPRS communication module. Subsequently, a hardware circuit has been designed to adapt the voltage levels of the analogue signals of the motor / generator system to the input of the microcontroller. Afterwards, a printed circuit board has been developed that includes the adapter circuit and the Bluetooth.

Also, a program developed in the microcontroller collects the information about the analogue signals and allows to read the temperature and the vibrations by SPI port. The microcontroller transmits this data to the PC via USB or wirelessly, which runs a LabVIEW program for data processing. The system for sending data to the Cloud via GPRS and alarm SMS has also been provided. Tests have been carried out to validate this monitoring system.

Keywords: Industrial monitoring, motor, smart sensors, microcontroller programming, wireless transmission, industrial communication.

ÍNDICE

DOCUMENTOS CONTENIDOS EN EL TFM

- Memoria
- Presupuesto
- Anejos

I. ÍNDICE DE LA MEMORIA

II. ÍNDICE DEL PRESUPUESTO

III. ÍNDICE DE LOS ANEJOS

ÍNDICE DE FIGURAS

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE MÁSTER DE INGENIERÍA INDUSTRIAL

DESARROLLO DE UN SISTEMA DE MONITORIZACIÓN DE UN MOTOR DE CORRIENTE CONTINUA CON SENSORES INTELIGENTES

MEMORIA

CAPÍTULO 1. INTRODUCCIÓN

1.1. INTRODUCCIÓN AL MANTENIMIENTO INDUSTRIAL

El desarrollo en la tecnología de equipo de medida y control ha provocado una evolución considerable en el campo del mantenimiento industrial. Se presenta, a continuación, la evolución en el mantenimiento industrial y se dan a conocer unos criterios básicos para la implantación del mantenimiento predictivo.

Mantenimiento correctivo: El mantenimiento consiste en intervenir cuando se producen averías, lo que conlleva unos costes de reparación y costes por paradas en la producción.

Mantenimiento preventivo: Revisiones periódicas por parte de los técnicos encargados del mantenimiento con el fin de reducir los costes productivos asociados a la disponibilidad reducida de la máquina y de sus paradas de producción, manteniendo las máquinas en un mejor estado y reduciendo la probabilidad de fallo. La eficiencia del mantenimiento está limitada por la incertidumbre derivada del coste que este método genera.

Mantenimiento predictivo: Fuente de la incertidumbre del mantenimiento preventivo y con ayuda del desarrollo tecnológico, este método de mantenimiento se basa en monitorización de la condición y/o estado de la máquina. Este método de mantenimiento se basa en la anticipación a una posible avería en la máquina conociendo su comportamiento y cómo debería comportarse, pudiendo conocer que elemento puede dar fallo y cuándo se produciría este fallo. Da la posibilidad de intervenir sin afectar en gran medida al proceso de producción, logrando de este modo un descenso tanto del coste de parada de la línea de producción, como de la mano de obra y los repuestos requeridos.

Mantenimiento proactivo: Se basa en el análisis de las causas que producen fallos, intentando evitar y reducir las averías repetitivas.

En la Figura 1 se puede ver gráficamente la actuación temporal de cada tipo de mantenimiento analizado anteriormente.

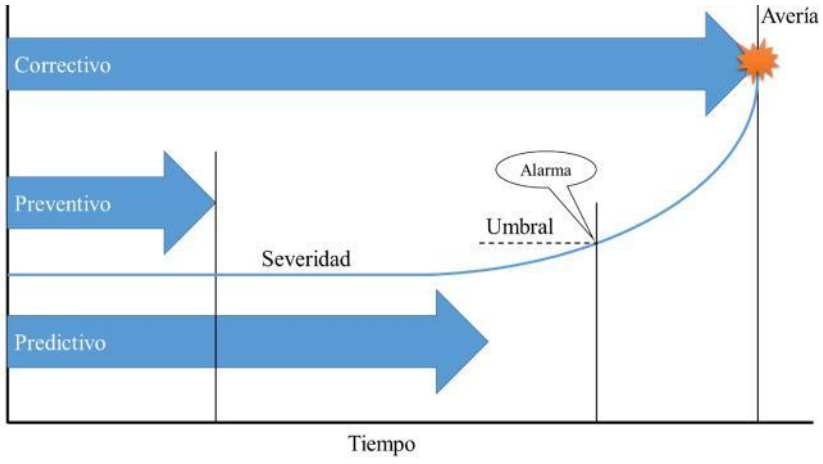


Figura 1. Tipos de mantenimiento

1.1.1. MANTENIMIENTO CORRECTIVO

Las intervenciones correctivas son inevitables pues la finalidad del mantenimiento es alargar la vida de una máquina por medio de la reparación o sustitución de sus componentes dañados. Sin embargo, cuando se habla de realizar únicamente un mantenimiento correctivo, se está hablando de una reparación.

Este tipo de mantenimiento sólo presenta la ventaja de que el coste inicial de inversión se puede considerar nulo. En contraposición, este tipo de mantenimiento presenta una serie de desventajas:

- Paradas de producción inesperadas.
- Paradas producto de roturas catastróficas.
- Costes elevados de reparación.
- Stock elevado de repuestos ante el desconocimiento del componente que puede fallar.
- Elevado riesgo de accidentes.
- Desconocimiento de las condiciones y estado de la máquina.
- Desconocimiento del origen de los fallos.

1.1.2. MANTENIMIENTO PREVENTIVO

El mantenimiento preventivo consiste en sustituir los componentes cuando se supone que se ha agotado su vida útil. El estudio teórico sobre la vida útil de los componentes por lo general es suministrado por el fabricante del equipo, que suele incluir un mantenimiento preventivo con indicaciones de sustitución de componentes y cambios en la lubricación.

Las desventajas que presenta este tipo de mantenimiento son las que siguen:

- Modificación de constantes de equilibrio de funcionamiento de la máquina, a causa de la intervención en correcto funcionamiento cuando le toca revisión.
- Desconocimiento del estado real de los componentes hasta su desmontaje.
- Descontrol sobre daños ocultos.
- Posible elevación de los costes causados por las frecuentes intervenciones en la máquina, aunque no sean necesarias.

1.1.3. MANTENIMIENTO PREDICTIVO

Este sistema de mantenimiento emplea técnicas de seguimiento y análisis para evaluar el estado de los componentes y permitir programar los mantenimientos requeridos cuando sean necesarios. Consiste en estudiar ciertos parámetros como la vibración, temperatura, aislamientos, aceites, etc., que tienen una relación con la condición o estado de la máquina y nos dan información sobre el estado de sus componentes y del modo de funcionamiento del equipo, lo cual permite detectar tanto problemas en los componentes como en el diseño e instalación. El sistema de mantenimiento predictivo tiene como objetivo la reducción de costes tanto de operación como de mantenimiento, con el consecuente aumento de la fiabilidad de la máquina.

En funcionamiento normal de la instalación se deben evaluar los parámetros necesarios para el control, por lo que se deben monitorizar los equipos para llevar a cabo el mantenimiento predictivo. Con esto, el estado de las máquinas es conocido durante el trabajo, no siendo necesaria una parada para la evaluación de las condiciones.

Dado que con las técnicas disponibles es posible evaluar los fallos en los componentes y el seguimiento de su evolución, posibilita la coordinación del momento más adecuado para realizar una intervención de mantenimiento.

Nos encontramos con unas desventajas principales que son el elevado coste de inversión inicial en tecnología y formación cuya rentabilidad se obtiene a medio y largo plazo. Sin embargo, las ventajas que este tipo de mantenimiento presenta la convierten en la mejor de entre las descritas:

- Conocimiento actualizado de las condiciones de funcionamiento y estado de la máquina.
- Supresión de la mayor parte de las posibles averías.
- Intervención cuando es necesario.
- Conocimiento del daño en los componentes, con la posible planificación para la sustitución en el momento conveniente.
- Conocimiento del problema con la consecuente reducción del tiempo de reparación.
- Identificación de fallos ocultos y crónicos.
- Stock de repuestos reducido.
- Incremento de la seguridad en la planta.

No siempre es conveniente aplicar el mantenimiento predictivo, existiendo dos razones que no sea aplicable:

- En la planta no todas las causas de fallo podrán ser detectadas con antelación.
- Costoso en equipo, en mano de obra o en una combinación de ambos.

Sólo en el caso en el de que el coste de monitorización preventiva sea inferior a la reducción esperada en costes de indisponibilidad o mano de obra, o se trata de que la seguridad de las personas es relevante, es conveniente la aplicación del mantenimiento predictivo.

1.1.4. MANTENIMIENTO PROACTIVO

Tiene como base los métodos predictivos, con la necesidad de la implicación del personal de mantenimiento para la identificación y corrección de las causas de las averías en la máquina.

La viabilidad de este sistema se encuentra en una adecuada organización de los recursos disponibles, una motivación de los recursos humanos que se le destinan, un control del funcionamiento de los equipos permitiendo acotar sus paradas y el coste que conlleva y una exhaustiva planificación de las tareas que se deben realizar durante un intervalo de tiempo. Por tanto, con el mantenimiento proactivo se pretende lograr llevar al equipo a las condiciones que se establecen para que desempeñe adecuadamente su trabajo por más tiempo sin un exceso de paradas para mantenimiento, es decir, las pérdidas de tiempo justas.

Se va a ver a continuación cuál de los siguientes tipos de mantenimiento es el más empleado en la industria tradicionalmente, y hacia dónde está evolucionando actualmente y en el futuro próximo.

Como se puede observar en la Figura 2 el mantenimiento correctivo sigue siendo muy importante en el mantenimiento industrial. Seguido de éste, se encuentra el mantenimiento preventivo, siendo éste el 25% de la práctica de mantenimiento. De la misma manera, el personal de predictivo tiene que estar altamente cualificado en estas tecnologías y requiere de un coste mayor en la adquisición de equipos, por lo que su implantación es menor. En general, esto muestra que la implicación general de una organización funcionando así (proactivo) es baja.

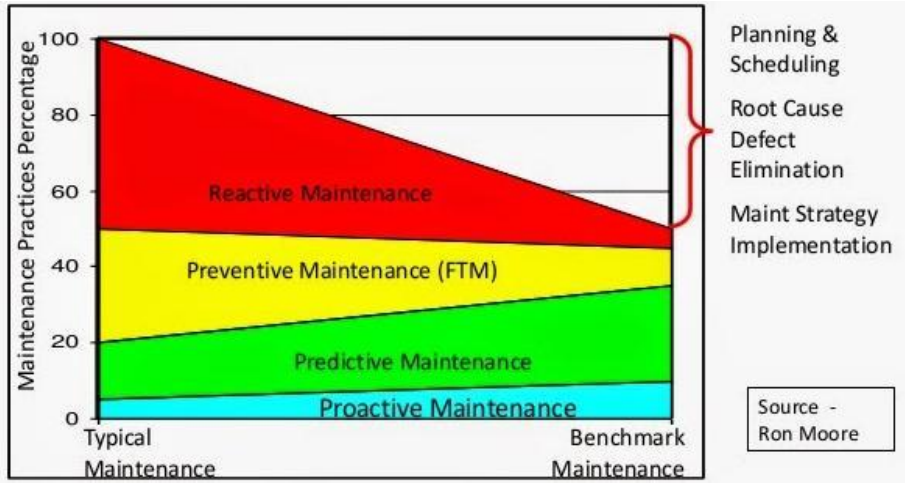


Figura 2. Evolución de las estrategias de mantenimiento[1].

Sin embargo, actualmente el mantenimiento predictivo está adquiriendo una mayor importancia. Por ello, se está realizando una importante inversión en el mantenimiento predictivo, moviendo algunos de los recursos de preventivo hacia esta estrategia. Con ello se consigue detectar a tiempo un mayor número de averías, sustituyendo los componentes dañados o arreglando la causa raíz del problema. Consecuentemente, el personal de correctivo tiene menos carga de trabajo, por lo que al final se consigue un ahorro en horas de trabajo, tiempo de paro y costes de las reparaciones planificadas.

1.2. TÉCNICAS DE MANTENIEMINTO PREDICTIVO

Este mantenimiento predictivo tiene como objetivo evitar averías mediante el empleo de una serie de técnicas que evalúan el estado de la máquina. Se basa en la medición de un conjunto de variables que pueden ser capaces de determinar la aparición de problemas de funcionamiento en el equipo al variar sus mediciones fuera de unos valores preestablecidos. Este tipo de mantenimiento requiere unos conocimientos tecnológicos superior al resto de mantenimientos, así como unos conocimientos científicos también mayores.

Dentro de las técnicas de mantenimiento predictivo destacan:

- Inspección visual y revisión de indicadores. Consiste en buscar problemas que se puedan observar a simple vista, en el equipo a controlar, que puedan generar averías.
- Análisis de vibraciones. Se trata del estudio del funcionamiento de máquinas rotativas en base a las vibraciones que en ella se producen a causa de su funcionamiento. Esta técnica es explicada con mayor detalle en el apartado 1.3 al tratarse una de las técnicas más empleadas en el mantenimiento predictivo.
- Análisis de temperatura. Esta técnica toma el comportamiento de la temperatura en el equipo como factor que puede influir en un fallo. Este calor es producido tanto por rozamiento como en la parte eléctrica, por efecto Joule. EL calor puede ser medido mediante cámaras termográficas, que capturan la radiación infrarroja emitida.
- Análisis de ultrasonidos. Esta técnica consiste en analizar los sonidos que produce la máquina, imperceptibles por el oído humano, mediante sensores ultrasónicos que convierten un sonido de onda corta en una señal audible por el oído humano, o la muestran en una pantalla. La utilización de sensores ultrasonidos permite reconocer las condiciones y estado de la máquina, como por ejemplo para la inspección de rodamientos o para la detección de defectos internos en las piezas.
- Análisis de aceite. En el caso de máquinas lubricadas mediante fluido, las espectrometrías, ferrografías y contenidos de partículas proporcionan información sobre el estado de los componentes lubricados en la máquina, pudiéndose detectar con mayor antelación los defectos.
- Análisis de corriente. Se basa en el análisis de la corriente consumida por cada parte que compone la máquina eléctrica. Un consumo muy por encima de los valores nominales, en alguna de sus partes, puede indicar un deterioro importante de la máquina. La medición de la corriente se realiza mediante amperímetros, colocados en serie con la línea que se desea medir.

1.3. MANTENIMIENTO PREDICTIVO MEDIANTE EL ANÁLISIS DE VIBRACIONES

El análisis de vibraciones es una de las técnicas más empleadas en el mantenimiento predictivo. Los elementos que constituyen la máquina poseen unas tolerancias inherentes que harán vibrar a la máquina, proporcionándole una vibración característica básica de referencia para comparar con las vibraciones registradas durante el funcionamiento de la máquina. Debido a estas tolerancias constructivas, todas las máquinas similares funcionando en buenas condiciones poseerán vibraciones características similares. Si en una máquina cambia su vibración básica de funcionamiento en condiciones normales, esto será indicativo de un incipiente defecto en alguno de sus componentes. Los diferentes tipos de fallos producirán diferentes cambios en la vibración básica de la máquina, ayudando a identificar la fuente del fallo.

Para identificar la vibración característica de los componentes o defectos del equipo se suele realizar un análisis espectral de vibraciones, consistente en la realización de una transformación de una señal en el tiempo al dominio de la frecuencia. Un ensayo de vibraciones habitual para máquinas rotativas en el que se realiza cuando se procede a la reducción de la velocidad que antecede a la

parada de la máquina y que aplica un efecto de amplificación de las vibraciones cuando el sistema entra en resonancia.

La medición de las vibraciones en el entorno de los cojinetes de la máquina podría detectar y diferenciar entre desalineamiento del eje, desequilibrio, fallo de cojinetes, fallo en elementos de transmisión como engranajes, desgaste y numerosos fallos más. Mediante la monitorización de las vibraciones en la máquina se pueden detectar las causas más comunes de fallo:

- Desequilibrio.
- Desalineamiento.
- Holguras estructurales.
- Desgaste mecánico.
- Desequilibrio.

1.3.1. DESEQUILIBRIO

El desequilibrio en un equipo mecánico es, probablemente el fallo más común, estando asociado a la distribución no uniforme de masas sometidas a rotación. El desequilibrio mecánico no es la única causa de un desequilibrio en la máquina, las inestabilidades hidráulicas o aerodinámicas también pueden generar un desequilibrio. Es decir, el desequilibrio, de una u otra forma, será generado por todos los tipos de fallo.

En la mayoría de las ocasiones, cuando se trata de la señal de vibración en desequilibrio, la componente de la velocidad de giro será excitada y con una amplitud dominante, aunque puede tomar diferentes formas. Por otra parte, se debe tener en cuenta que esta condición de desequilibrio podrá excitar múltiples armónicos (múltiplos de la velocidad de giro), tanto el número de armónicos como su amplitud son directamente proporcionales con el número de planos de desequilibrio y su relación de fases.

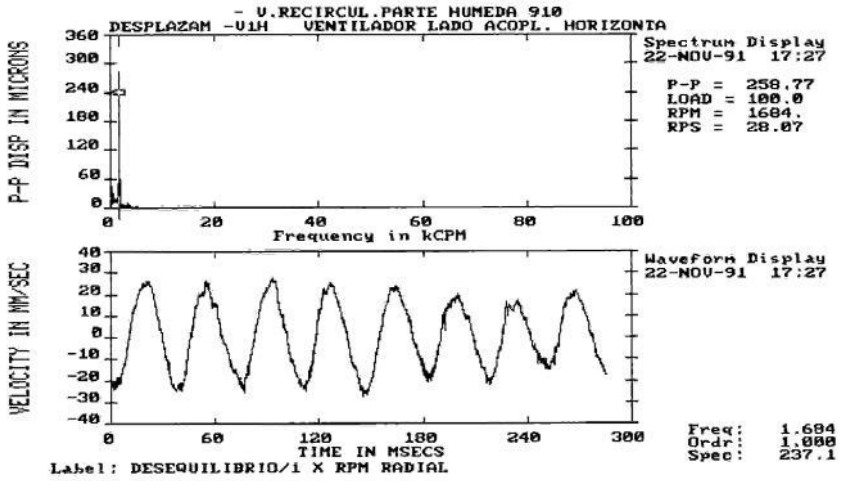


Figura 3. Desequilibrio ventilador horizontal.[2]

En la Figura 3 se puede ver un ejemplo de un ventilador horizontal en el que se monitoriza las revoluciones del mismo en función del tiempo, y la amplitud de la vibración en función de la frecuencia, de donde podemos obtener el número de armónicos que posee el ventilado. En este caso, un único armónico.

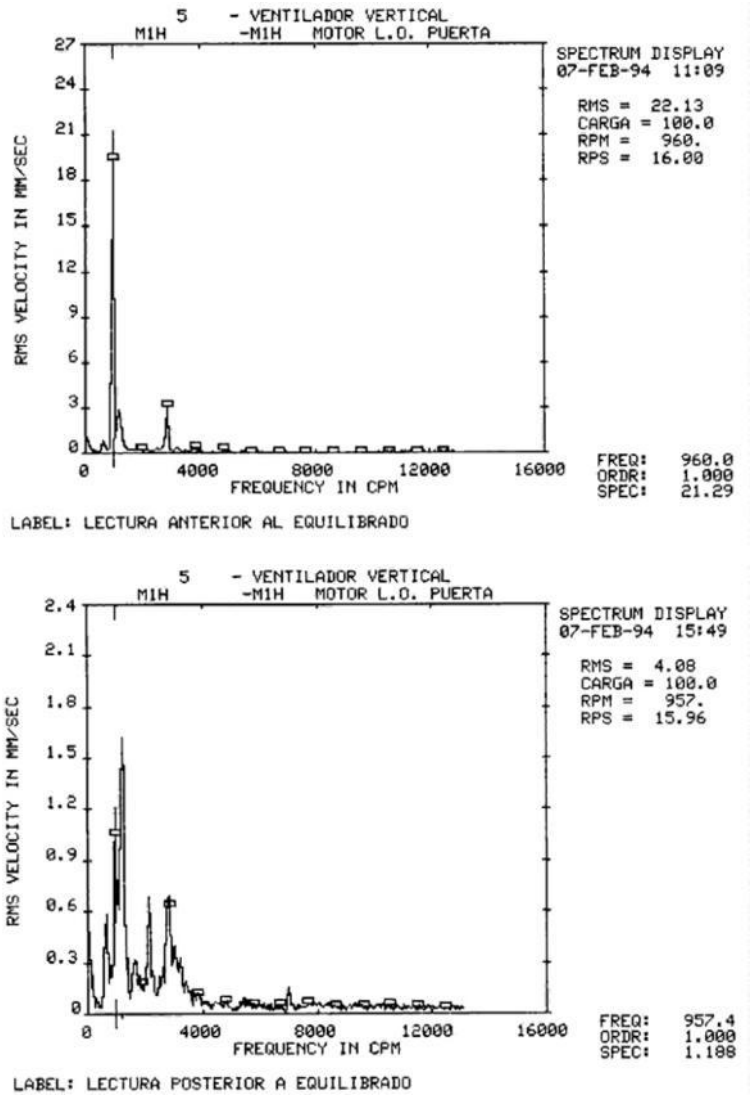


Figura 4. Desequilibrio en un ventilador vertical.[3]

En la Figura 4 se muestra la gráfica de la frecuencia de vibración en función de la velocidad de giro de un ventilador vertical antes y después del equilibrado. Aquí se ve que, tras el equilibrado, las frecuencias de vibración son mayores con una menor velocidad de giro.

1.3.2. DESALINEAMIENTO

Cuando existe un grupo de máquinas, este tipo de fallo suele estar presente. Por lo general, cuando dos ejes están conectados por un acoplamiento podría dar lugar al desalineamiento. Además,

también se puede dar entre los cojinetes de un eje sólido o entre otro par de puntos. Éste último hace referencia al desalineamiento producido en la unión entre máquinas por otro tipo de elemento que no sea acoplamiento, i.e, engranajes.

La morfología de la señal de vibración por desalineamiento dependerá del tipo de este:

- Desalineación paralela: Se da entre dos ejes paralelos entre sí, pero que no están en el mimo plano. Este tipo de desalineamiento producirá una vibración radial de dos veces la velocidad de giro real del eje.

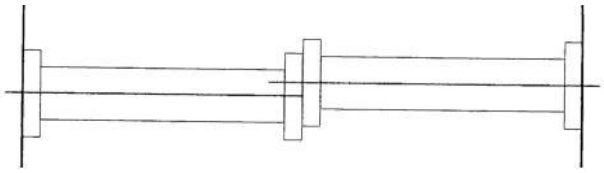


Figura 5. Desalineación paralela.

Fuente: "Vibraciones en máquinas. Mantenimiento predictivo". Departamento de Ingeniería Mecánica, Energética y de Materiales, Universidad de Navarra.

- Desalineación angular: Se da entre dos ejes que no están paralelos entre sí. Como consecuencia se producirán vibraciones axiales (como son las paralelas al eje). En este caso, la frecuencia de vibración puede llegar a ser dos o tres veces la velocidad de rotación.

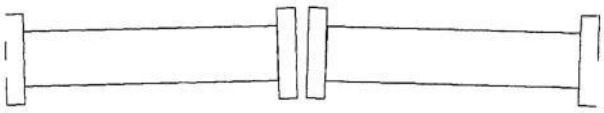


Figura 6. Desalineación angular.[4]

1.3.3. HOLGURAS ESTRUCTURALES

La insuficiente sujeción de la máquina a la bancada provocará múltiples armónicos con unas amplitudes prácticamente idénticas y una amplia variedad de morfologías de señales de vibración. La señal más frecuente es aquella originada con un componente de frecuencia primaria la mitad de la velocidad de rotación, y generará varios armónicos de este componente primario.

A continuación, se muestra un ejemplo de fallos de anclaje con la bancada de una bomba horizontal que soporta a la máquina junto con su espectro de frecuencias (Figura 7).

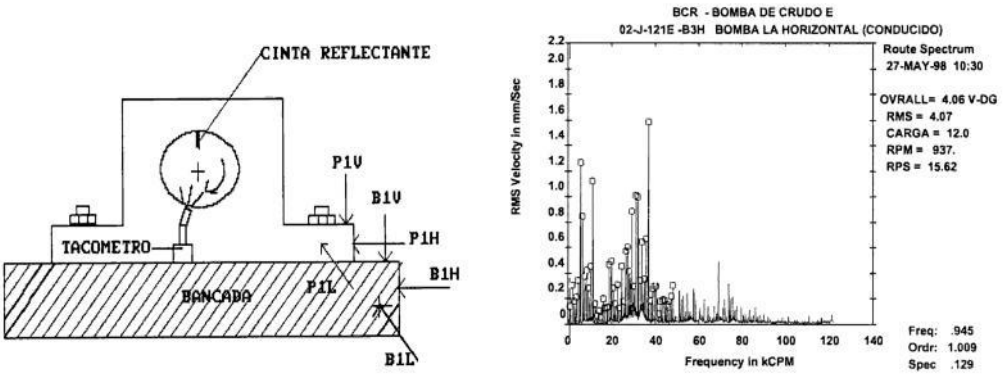


Figura 7. Bomba horizontal sobre bancada.[5]

1.3.4. DESGASTE MECÁNICO

Este defecto se dará por el rozamiento incorrecto de los elementos de la máquina, lo que provocará vibraciones de baja amplitud (picos entre 1 y 10 Hz normalmente) difícilmente medibles por muchos sistemas de monitorización. Los picos de baja frecuencia irán acompañados por un pico menor, entre el 25 y 40% de la velocidad de rotación del eje.

1.3.5. MEDIDA DE LAS VIBRACIONES

Con el objeto de determinar las condiciones de operación de la máquina la programación del mantenimiento predictivo en base al análisis de vibraciones debe proporcionar datos precisos y repetitivos.

Para medir las vibraciones en el exterior y estructura de las máquinas se emplean transductores, que transforman una magnitud física que están midiendo en una señal eléctrica proporcional. Para este fin, son los acelerómetros los transductores más empleados. Estos tienen la ventaja, respecto a otros transductores como los velocímetros, de tener un mayor rango de frecuencia, poder integrar la señal para obtener velocidad o desplazamiento vibratorio y ser más pequeños.

Estos acelerómetros pueden estar basados en diferentes tecnologías como son:

- Piezo-eléctricos, en los que una deformación física del material causa un cambio en la estructura cristalina, cambiando así las características eléctricas.
- Piezo-resistivos, en los que una deformación física del material cambia el valor de la resistencia del puente.
- Capacitivos, en los cuales el movimiento paralelo de una de las placas del condensador, de los que está compuesto, hace variar su capacidad.

De este ellos, son los sensores capacitivos los más utilizados para el monitoreo de máquinas de rotación para el análisis de sus características vibratorias, producidas por grietas o fatigas de sus componentes, monitorizando continuamente las vibraciones de la máquina.

No sólo el tipo de transductor afecta a la calidad de los datos, sino que hay tres factores que también son de importancia:

- El punto de medición.
- La orientación del transductor.
- La carga de compresión.

Es importante que cada medición se consiga exactamente en el mismo punto con la misma orientación, durante todo el programa, puesto que la desviación de uno de estos dos factores afectará a la exactitud de los datos afectando a la detección de los posibles problemas incipientes.

La carga de compresión que se aplica al transductor deberá ser la misma en cada medición, puesto que las desviaciones de esta carga provocarán errores en la amplitud de las vibraciones pudiendo crear unas componentes de falsas frecuencias que no se deben asociar con la máquina.

Con el fin de garantizar la seguridad y repetitividad de los datos recogidos es mediante transductores fijos. Estos transductores se instalan rígidamente en los puntos de medición que se han seleccionado para el control. Su inconveniente es el coste inicial que tiene el programa establecido para el control.

Una solución para este elevado coste inicial es mediante el diseño de un conector rápido que permita el acople de un acelerómetro mediante un acople a una clavija de desconexión rápida.

El montaje magnético también se puede emplear como solución, con la ventaja de poder situarse el transductor en cualquier lugar de la máquina, pero presenta el inconveniente de que no puede garantizarse la situación y orientación exacta en cada medición. Su uso con base magnética debe limitarse por debajo de los 1000 Hz, aunque el conjunto del transductor y la base magnética tendrá una frecuencia de resonancia que puede ocasionar distorsiones en el dato registrado.

Una última forma de conseguir los datos es mediante transductores manuales, aunque este procedimiento no es recomendable, pues no proporcionan la seguridad y repetición requeridas.

El transductor empleado se sitúa sobre la parte del equipo que se quiere controlar o cerca de este punto a controlar. Por ejemplo, si lo que se requiere es comprobar el estado del rodamiento el transductor se situaría sobre el soporte del mismo; si se pretende detectar la cavitación de la bomba que el motor está moviendo el transductor se situaría sobre la carcasa de esta. Puesto que lo habitual es que las vibraciones estén asociadas a partes móviles, el soporte del rodamiento o la carcasa de la bomba serán los lugares más adecuados para la mayor parte de las mediciones.

Los transductores deben situarse en función del efecto que se quiera controlar, localizándolos en aquellos elementos de la máquina que transmitan apropiadamente la vibración.

Cada componente tendrá una frecuencia natural a la que vibrará con mayor o menor magnitud en respuesta a una excitación, y esta frecuencia natural será función de la masa y elasticidad del componente. Ante excitaciones muy por encima de su frecuencia natural los componentes no responderán de forma significativa.

Por ello, para que la vibración sea captada por el transductor este se debe instalar sobre un componente con una frecuencia natural elevada, siendo los más adecuados los componentes rígidos.

1.4. MANTENIMIENTO PREDICTIVO MEDIANTE EL ANÁLISIS DE TEMPERATURA

El control de temperatura en los componentes de la máquina persigue los siguientes propósitos:

- Control de la temperatura de un proceso o que se está llevando a cabo adecuadamente.
- Detección de incremento de generación de calor en un elemento.
- Detección de cambios de la transmisión de calor en una máquina exterior, con motivo de cambios en alguno de sus componentes.

Se pueden detectar los siguientes fallos mediante la monitorización de la temperatura:

- Rodamientos dañados. Los daños en rodamientos y cojinetes lubricados producirán un incremento en la generación de calor que se traducirá en un aumento de temperatura en la superficie del soporte del elemento en cuestión. Este incremento de temperatura es detectable mediante un sensor colocado en la superficie, como puede ser un termopar, o por la diferencia de temperatura entre dos sensores cuya posición de uno sea la superficie y el otro posicionado a una pequeña distancia de la superficie.
- Daños en el aislamiento. Mediante cámaras de infrarrojo se puede detectar los daños en los aislamientos de elementos que se deben encontrar aislados térmicamente.
- Fallos en componentes eléctricos. Al igual que los daños en aislamiento, mediante cámaras infrarrojas también son detectables los puntos calientes generados por una mala conexión eléctrica en la que se genera calor por la resistencia en contacto entre componentes.

Se puede controlar la temperatura en un punto interior del equipo, como la temperatura de la superficie de un componente (por ejemplo, de un rodamiento). Las medidas superficiales proporcionan una información más generalizada sobre la generación de calor en el equipo, así como de las vías de intercambio de calor.

Sin embargo, la adquisición de datos de temperaturas superficiales es más compleja, pues en el perfil de temperaturas habitualmente se da una fuerte discontinuidad presente en la superficie, y que con la instalación de sensores de temperatura es fácilmente modificado. En consecuencia, los pequeños instrumentos de medida como son los termopares, o los sensores sin contacto, como son los medidores por radiación, son los que más se adecuan para este fin.

- Sensores de contacto

Son los instrumentos más ampliamente utilizados, tomando la temperatura del objeto con el que están en contacto y transmitiendo dicha información.

La precisión y el tiempo de respuesta se ven afectados por el sistema de sujeción que se emplee para el sensor. El mejor método para la medición de la temperatura superficial es incrustando o soldando el sensor al objeto. Los sensores más pequeños serán los que detecten más rápidamente los cambios de temperatura, pues el tiempo de respuesta está relacionado con su volumen.

- Sensores sin contacto

De acuerdo con la ley de Stefan-Boltzmann:

$$E = \sigma \epsilon \mathrm{T}^4$$

Donde:

- σ, es la contante de Stefan-Boltzmann.
- ϵ, es la emisividad de la superficie.
- T, es la temperatura absoluta de la superficie.

La radiación de energía de un cuerpo varía con su temperatura absoluta y la emisividad de la superficie radiante. De este modo, la energía radiante del cuerpo a medir se puede deducir sin necesidad de estar en contacto. Sin embargo, la variabilidad de la emisividad es la principal fuente de imprecisión de este método.

1.5. SISTEMA MOTOR/GENERADOR

El objetivo de este trabajo fin de máster es el desarrollo de un sistema de monitorización multivariable del sistema motor/generador utilizando en la medida de lo posible los sensores inteligentes, específicamente se pretende monitorizar la tensión en borne del motor, la corriente que atraviesa el motor, velocidad de giro del motor, temperatura en la resistencia de recarga y las vibraciones. El sistema debe enviar los datos recopilados a un PC local mediante cables y con posibilidad de enviar inalámbricamente. Asimismo, el sistema debe poder enviar los datos a una plataforma de Nube mediante GPRS al mismo tiempo, y mensajes de alerta vía GSM ante cualquier situación indeseada.

1.5.1. DESCRIPCIÓN DEL MOTOR

En el laboratorio del Departamento de Ingeniería Electrónica está equipado con un motor de corriente continua que está unido mediante un acoplamiento mecánico a un generador de las mismas características, cuya velocidad de funcionamiento se puede alterar mediante la variación de tensión en borne del motor con un puente H cuyo esquema es el que se muestra a continuación (Figura 8):

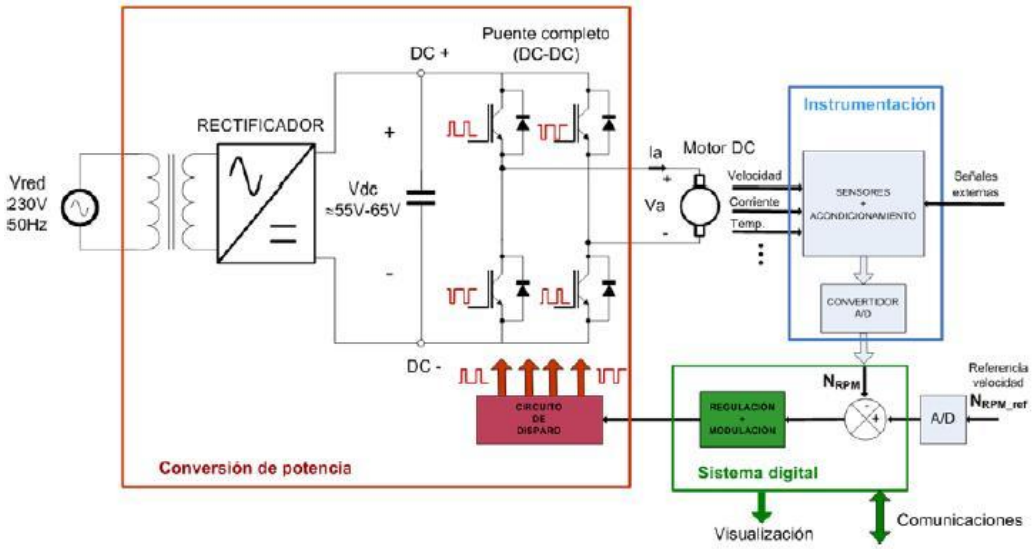


Figura 8. Diagrama de bloques de un variador de velocidad para un motor de corriente continua.[6]

El conjunto se encuentra aislado de la red eléctrica mediante un transformador reductor de tensión (Figura 9). Tras esta reducción de tensión nos encontramos con un puente de diodos que convierte la corriente alterna que entrega la red eléctrica en una tensión continua (tensión rectificada, Vdc), que es la que alimenta la placa del variador de velocidad.

Para que la tensión rectificada se aproxime a un valor continuo se realiza un filtrado capacitivo con un condensador de un valor suficientemente elevado. Sin embargo, como en el arranque el condensador se encuentra descargado, la corriente inicial puede llegar a ser muy elevada, por lo que se limita esta corriente de arranque con una resistencia que es cortocircuitada cuando el condensador se encuentra cargado.

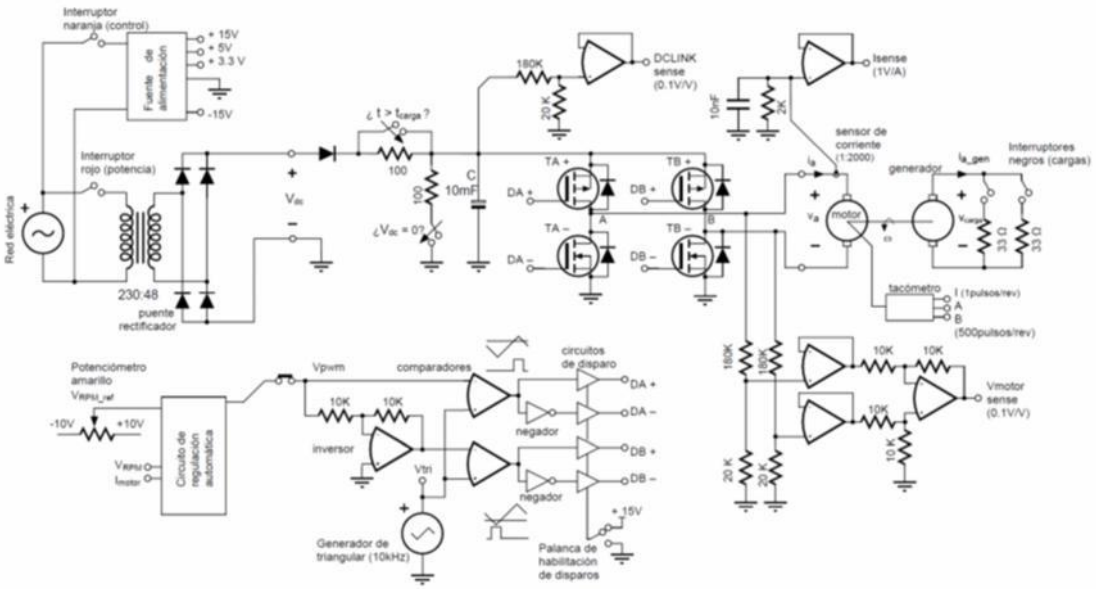


Figura 9. Esquema simplificado de un variador de velocidad para un motor de corriente continua.[7]

El puente en H (Figura 9) consiste en un par de transistores. TA+ y TA- son los encargados de conectar el terminal A con Vdc o GND según esté activado TA+ o TA-. Igualmente, pero con el terminal B, los transistores TB+ y TB- actúan del mismo modo. Esta etapa tiene como objetivo entregar una tensión positiva (Vdc) en bornes del motor si está activada la pareja de transistores TA+ y TB-, y una tensión negativa (-Vdc) si está activada la pareja de transistores TA- y TB+. La tensión en bornes valdrá 0 si están activos los dos transistores superiores o los dos inferiores, ya que en este caso habría un cortocircuito de la tensión de entrada, destruyéndose los transistores correspondientes por sobre corriente.

Los disparos de los transistores (TA+, TA-, TB+, TB-) son producidos por las señales de control en las puertas de los mismos (DA+, DA-, DB+, DB-). Estas tensiones son pulsos de duración variable obtenidos por la comparación entre una tensión triangular (generada por un circuito basado en amplificadores operacionales) y un nivel de tensión (Vpwm) que es el obtenido de una tensión de un circuito de regulación automática que considera la velocidad deseada, impuesta por el usuario mediante un potenciómetro (potenciómetro amarillo en recuadro rojo de la Figura 10), y una tensión que es proporcional a la velocidad de giro (Vrpm que se construye a partir de la señal entregada por el tacómetro del motor y un circuito no representado).

Figura 10. Interruptor para la habilitación de disparos (recuadro amarillo) y el potenciómetro para variar la velocidad de giro del motor (recuadro rojo).[8]

El motor se encuentra conectado a un generador (máquina eléctrica de idénticas características) mediante una unión mecánica. Dicho generador puede "rodar en vacío" si no se conecta a ninguna carga. Asimismo, se han instalado dos resistencias de 33 Ω que pueden conectarse (solo una o, en paralelo, las dos) a la salida del generador. El objetivo de este generador y las cargas es simular un trabajo mecánico (de magnitud fija y conocida) ejercido por el motor, de manera que se puede variar el par electromecánico, T_e , producido por el motor en el punto de operación. La conexión de una o de las dos resistencias se realiza actuando sobre los interruptores negros en el panel de mandos (recuadro amarillo en Figura 11), al lado de los interruptores de marcha de la etapa de potencia y de control (círculos rojos en Figura 11).

En la Figura 11 se muestra el bloque contenedor de los circuitos de alimentación y accionamiento de las cargas con sus respectivos interruptores. A la derecha, con dos círculos rojos, nos encontramos con los interruptores de alimentación de potencia del sistema y alimentación de control de la placa reguladora del sistema. A la izquierda, remarcados con un cuadro amarillo, se encuentran los interruptores de accionamiento de las resistencias eléctricas simuladoras de las cargas en el motor.

Figura 11. Conjunto cerrado para la manipulación de la alimentación y las cargas del sistema.[9]

Por otra parte, la placa proporciona una serie de señales de importancia para el proyecto. Estas señales son extraídas de esta placa y tratadas posteriormente (como se describirá en el **¡Error! No se encuentra el origen de la referencia.**):

- V_{m_s} : Es la tensión que aparece en el motor. La tensión en borne del motor es una señal cuadrada de 10 Hz, por lo que posteriormente se debe de implementar un filtro paso bajo. La manera de obtener esta tensión es mediante la resta de las tensiones en sus dos terminales, que se realiza mediante un amplificador diferencial. Para no superar la tensión de alimentación del operacional, se atenúan previamente ambas tensiones de bornes a la décima parte mediante un divisor resistivo en cada terminal. De este modo, V_m devuelve la tensión en bornes dividida por 10.
- I_{s} : Proporciona la corriente que atraviesa el motor con una ganancia de 1 V/A. Esta corriente se obtiene mediante un sensor de corriente que tiene un factor de conversión de valor de 1:2000 de modo que intercalando una resistencia de 2000 Ω obtenemos esta tensión proporcional a la corriente. Con el fin de evitar el problema de adaptación de impedancia, se instaló un seguidor de tensión mediante amplificador operacional.
- V_{R} : Devuelve una señal de tensión proporcional a la velocidad de giro del motor. Se emplea un encóder incremental en el eje del motor, el cual proporciona tres señales de salida: A, B, I. Los terminales A y B devuelven unas señales del tipo TTL de 500 pulsos/revolución. Estas señales tienen un desfase de 90º una respecto a la otra, de modo que es posible la determinación del sentido de giro del motor: si la señal B está desfasada respecto a la señal A el sentido de giro es antihorario, y si la señal A está desfasada respecto a la B el sentido de giro es horario. La señal I es el pulso índice, un pulso que ocurre una vez por rotación. Su duración es nominalmente un ciclo eléctrico del terminal A o B. Así mismo, se ha realizado un procesado de señal proveniente del tacómetro mediante un convertidor de frecuencia-tensión de

manera que la velocidad de giro del motor puede medirse mediante la tensión continua del terminal $V_{r1} = 0.003 \cdot R$.

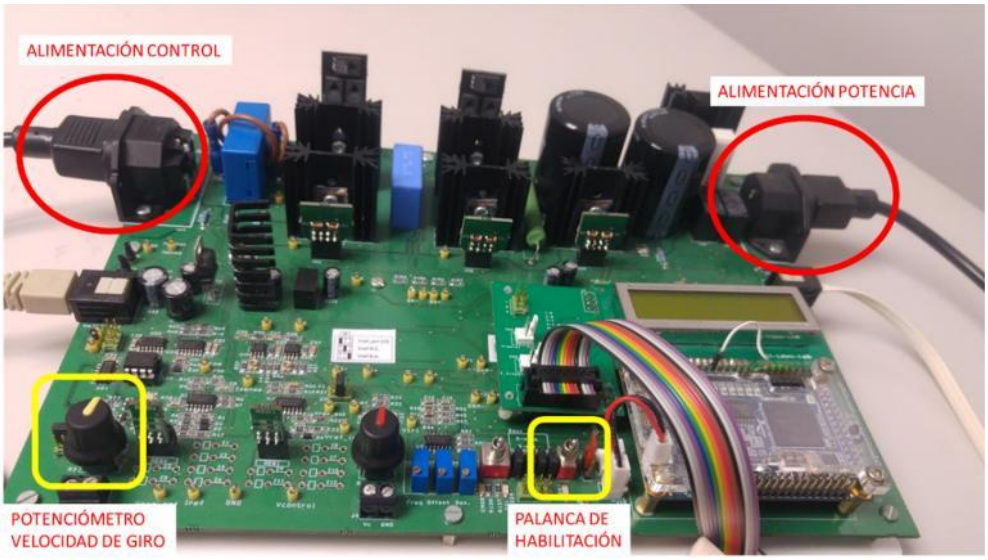


Figura 12. Placa de control de potencia.[10]

La Figura 12 muestra la placa de control de potencia empleada para la regulación de velocidad del motor y la toma de datos. Con dos círculos rojos se destacan los cables de alimentación de control (a la izquierda) y alimentación de potencia (a la derecha). Con rectángulos amarillos se destaca la palanca de habilitación de la energía al motor (a la derecha) y el potenciómetro empleado para la variación de la velocidad de giro (izquierda).

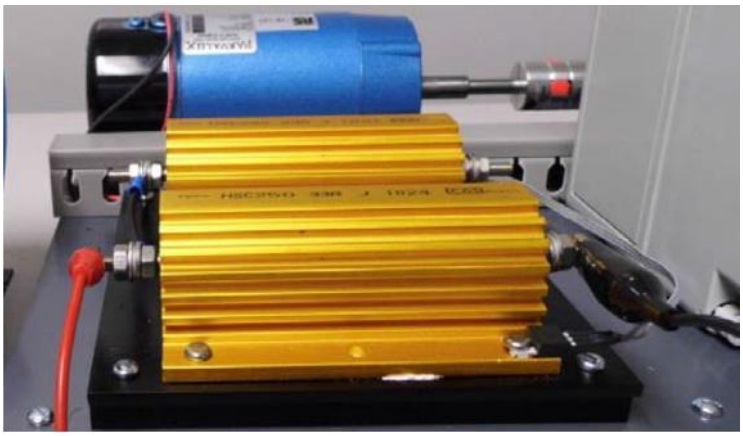


Figura 13. Resistencias para la simulación de cargas mecánicas.[11]

En último lugar, en la Figura 13 se muestran las resistencias empleadas para la simulación de la carga en el motor, que son accionadas mediante los interruptores negros mostrados en la Figura 11.

1.6. TENDENCIA ACTUAL DEL ÁREA DE INSTRUMENTACIÓN

Los sensores o transductores son dispositivos capaces de convertir la magnitud física a medir en una señal eléctrica. No obstante, la señal de salida del sensor suele ser de muy baja amplitud, por lo que es necesario el circuito acondicionamiento de señal que es el responsable de realizar la adaptación de impedancia si procede, la amplificación y filtrado de la señal, proporcionando así una señal analógica de mayor amplitud más robusta. A continuación, se suele digitalizar la señal analógica para su posterior visualización y procesado.

En este contexto, el término de sensor inteligente se ha introducido para definir aquellos sensores cuya funcionalidad va más allá que la simple transducción de señales en el mismo dispositivo, sino incluye el circuito acondicionamiento y digitalización de la señal. Además, poseen características tales como la capacidad de comunicación y diagnósticos que proporcionan información a un sistema de monitorización para aumentar la eficiencia operativa y reducir los costes de mantenimiento.

Las principales ventajas de estos sensores inteligentes son:

- Reducen el tiempo de diseño y desarrollo.
- Reducen el número y longitud de los cables, eliminando con ello la posibilidad de errores.
- Su consumo es más reducido, al igual que su tamaño.
- Evitan la transmisión de señales analógicas y, con ello, la reducción del efecto de ruidos e interferencias.
- Poseen un autodiagnóstico (detección de mal funcionamiento).
- Facilitan el control distribuido.
- Aportan una mayor escalabilidad-flexibilidad

La utilización de los sensores inteligentes en los sistemas de producción tiene tendencia a crecer en la actualidad debido a que resultan decisivos para poder conseguir procesos más eficientes, competitivos, optimizados y rentables. Es por ello que, se debe concienciar y formar a las nuevas generaciones de estudiantes y futuros trabajadores en el uso y aplicación de estas nuevas tecnologías.

En la Figura 14 se observa una gráfica representativa del creciente uso de los sensores inteligentes para aplicaciones domésticas e industriales.

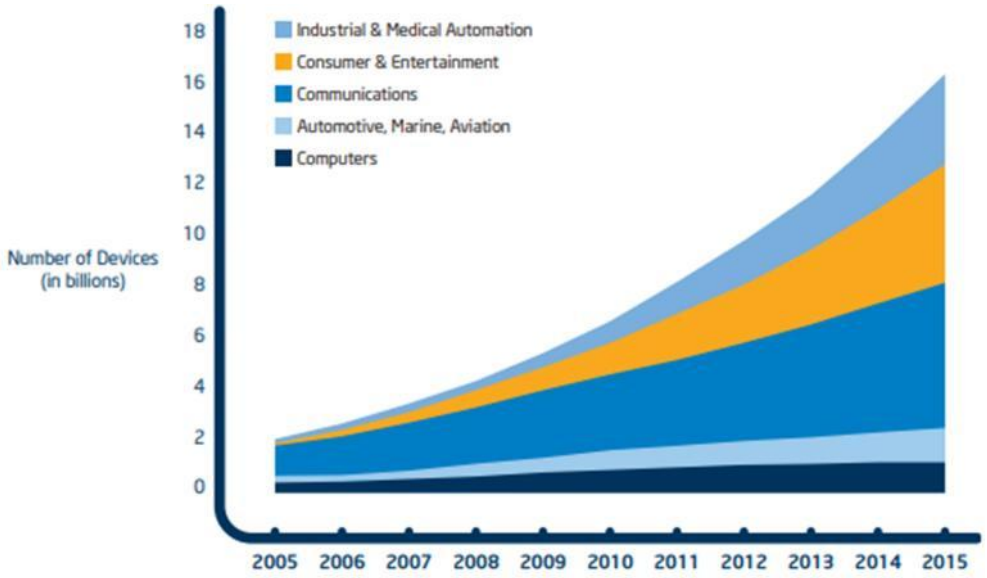


Figura 14. Evolución de IoT.[12]

Los sensores inteligentes con salida digital se comunican mediante buses de serie. Los más típicos son: Inter-IC bus (I2C) y Serial Peripheral Interface (SPI). En el ANEJO I se han descrito las principales características de estos modos de comunicación.

Por otro lado, resulta interesante que al instrumentar una máquina para su control no sea necesaria la presencia de un operario constantemente junto a ella, sino posee un sistema de alarma a distancia que permita informar de errores en el funcionamiento de la máquina. Por ello, es posible emplear la tecnología GSM para el envío de mensajes ante cambios no deseados en el funcionamiento. Es por ello, que se hace cada vez más necesaria la formación de los alumnos en estos nuevos métodos de control para adaptarlos y formarlos para su futuro laboral.

Asimismo, en la actualidad, debido a los nuevos sistemas de trabajo basados en las nuevas tecnologías, resulta necesaria una instrumentación que permita la recolección de la información de los sensores en una base de datos accesible desde diversos puntos de trabajo. Esta base de datos podría ser la Nube, que permite almacenar información en un servidor y ser accesible por varios usuarios desde diferentes partes de la geografía. Con el objetivo de mejorar la planificación, reducción de costes y optimización de procesos mediante un mantenimiento eficaz y controlado se hace cada vez más necesario la aplicación de las tecnologías inteligentes y la comunicación instantánea mediante información obtenida en tiempo real para mantener informados a toda una cadena de un proceso industrial (proveedor, productor y cliente). Serán, por tanto, los futuros alumnos y posteriormente trabajadores los encargados de desarrollar todos estos procesos e implementar y mejorar procesos de mantenimiento industrial que hagan posible la mejora continua.

CAPÍTULO 2. JUSTIFICACIÓN Y OBJETIVOS

En la industria, además de perseguir la optimización de los recursos para obtener un mayor rendimiento en la producción y por tanto la reducción del coste de producción, también se pretende aumentar la vida útil de las máquinas minimizando los costes de mantenimiento de las máquinas. El mantenimiento predictivo consiste en la monitorización continua de algunas magnitudes física, i.e, corriente, temperatura, vibraciones, sonidos producidos por las máquinas, con el fin de determinar la 'salud' de las máquinas comparando éstas con las magnitudes de referencia que deberían proporcionar las máquinas en su correcto funcionamiento. De esta manera es posible conocer de forma anticipada si se producirá algún error en la máquina, pudiendo actuar antes de perder eficiencia antes del paro de la máquina para su reparación. Todo ello con el fin de minimizar las paradas de las líneas de producción imprevistas por averías en las máquinas y reducir por tanto el coste de mantenimiento. Igualmente, es posible mejorar los paros por mantenimiento de las máquinas, puesto que no siempre que una máquina se detiene por mantenimiento, es necesaria la reposición de piezas.

Por otro lado, gracias a la integración del circuito de acondicionamiento y digitalización de la señal, asimismo se suele dotarle capacidad de comunicación y diagnóstico, existe actualmente una creciente tendencia del uso de sensores inteligentes en el entorno industrial que permiten reducir el tiempo de desarrollo y consumo del dispositivo, minimizar las interferencias que se acoplan al sistema y facilitar el control distribuido. El empleo de sensores inteligentes permite detectar con antelación los posibles fallos del sistema minimizando el coste adicional que suponen las paradas de la línea de producción inesperadas, aumentando así la eficiencia de la producción. Asimismo, ante la tendencia creciente del desarrollo de dispositivos IoT y su implantación en el entorno industrial, nace la necesidad de introducir estas tecnologías a los futuros alumnos del Máster Universitario de Ingeniería Industrial.

Por tanto, el objetivo de este trabajo fin de máster es el desarrollo de un sistema de monitorización multivariable del sistema motor/generador utilizando en la medida de lo posible los sensores inteligentes, específicamente se pretende monitorizar la tensión en borne del motor, la corriente que atraviesa el motor, velocidad de giro del motor, temperatura en la resistencia de recarga y las vibraciones. El sistema debe enviar los datos recopilados a un PC local mediante cables y con posibilidad de enviar inalámbricamente. Asimismo, el sistema debe poder enviar los datos a una plataforma de Nube mediante GPRS al mismo tiempo, y mensajes de alerta vía GSM ante cualquier situación indeseada.

CAPÍTULO 3. DESCRIPCIÓN DE LA SOLUCIÓN

Atendiendo a las especificaciones del proyecto, se ha propuesto un sistema de monitorización continua multivariable basado en microcontrolador.

A continuación, se adjunta el diagrama de bloque de la solución propuesta:

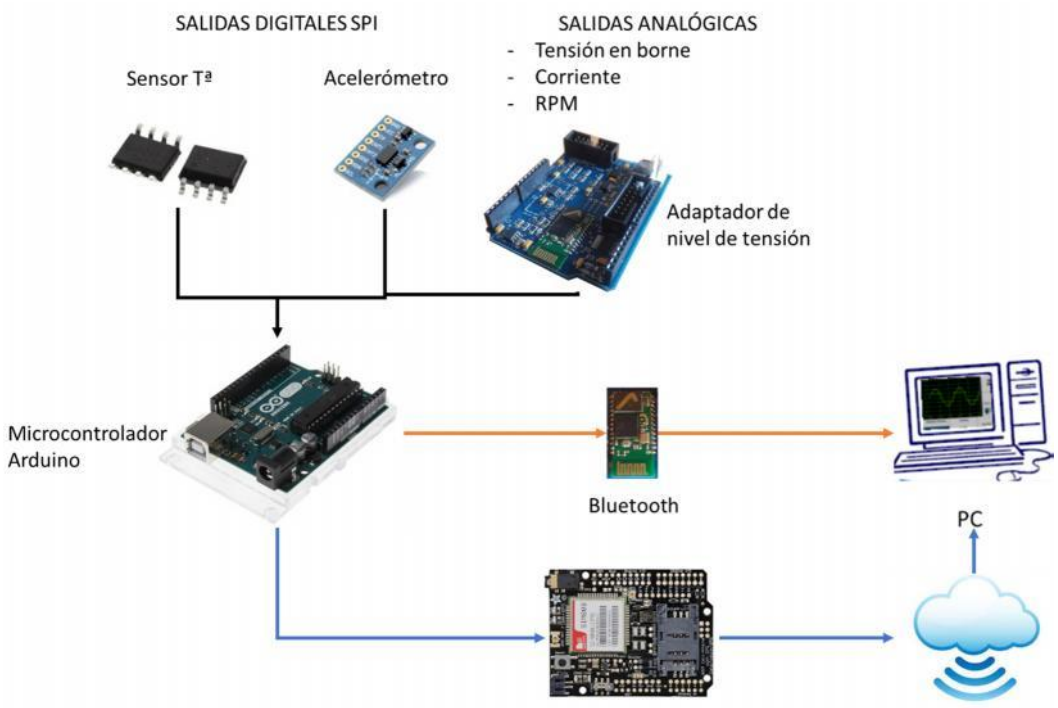


Figura 15. Solución de monitorización propuesta.

Primero en este proyecto sólo se pretende la monitorización de la temperatura y las vibraciones en la carga del sistema motor/generador mediante los sensores inteligentes ya que se dispone la instrumentación necesaria para la monitorización de las demás variables. Por lo que sólo es necesario instalar un sensor de temperatura y un acelerómetro. Tal como se mencionó en el apartado 1.6, los principales buses de comunicaciones de los sensores inteligentes son I2C y SPI. En principio no hay una gran diferencia en cuanto el número de líneas de comunicación entre el microcontrolador y los sensores en función del protocolo de comunicación a utilizar: SPI o I2C. Dado que el software del protocolo SPI es más sencillo y permite alcanzar una velocidad de tasa de transferencia de datos, se ha optado por SPI por sus mayores prestaciones y facilidad de programación. Por tanto, se ha utilizado un sensor de temperatura y un acelerómetro de 3 ejes con salida SPI para la monitorización de la temperatura y las vibraciones en la carga del sistema motor/generador.

Tal como se ha mencionado en el apartado 1.5, las placas de control del sistema motor/generador proporciona señales analógicas cuadradas de la tensión en borne del motor (entre 4 y -4 V), corriente que atraviesa por el motor (entre 4 y -4 V) y la velocidad de giro del motor (entre 10 y -10 V), por lo que se corrige el nivel de tensión a la entrada del microcontrolador mediante un circuito adaptador cuya función se expone en el apartado 4.5 y resulta:

$$V_{in\_mr} = \frac{R_2 R_4}{R_1 R_3} V_{in\_a} - \frac{R_5}{R_3} V_{ref\_a}$$

El microcontrolador, específicamente el kit de evaluación del Arduino Uno, no sólo es el responsable de leer la información del sensor de temperatura y de vibraciones por el puerto SPI, sino también debe adquirir las señales analógicas asociadas a la tensión en borne del motor, corriente que atraviesa por el motor y la velocidad de giro del motor mediante el convertidor analógico-digital integrado en el propio microcontrolador. Asimismo, el microcontrolador se encarga de transmitir al PC vía USB o de forma inalámbrica mediante Bluetooth. En el PC se ejecuta un programa de Labview que se encarga de leer los datos enviados por el microcontrolador o bien mediante USB o bien mediante Bluetooth, convertirlos en su correspondiente magnitud física para mostrarlos visualmente mediante gráficas.

Por otro lado, el microcontrolador también es el responsable de subir los datos a una plataforma de Nube, en la que se desarrolla una interfaz gráfica de visualización de los datos subidos, mediante GPRS y de enviar mensajes de alerta vía SMS ante cualquier situación inesperada.

CAPÍTULO 4. DESARROLLO DEL HARDWARE

4.1. SENSOR DE VIBRACIÓN

Un acelerómetro es un dispositivo electromecánico capaz de detectar las fuerzas de aceleración, ya sean fuerzas estáticas, que incluyen la gravedad, o fuerzas dinámicas, que pueden incluir el movimiento y las vibraciones.

Estos dispositivos son capaces de medir la aceleración en uno, dos o tres ejes, siendo cada vez más común la medición en tres ejes puesto que los costes de producción de este último tipo de dispositivos son cada vez menores.

Generalmente son dispositivos de baja potencia cuyo consumo es de orden de μA o mA, y tensiones iguales o inferiores a los 5V, lo que los hace dispositivos ideales para funcionamiento con baterías.

Para la detección de las vibraciones del sistema motor/generador se han definido las siguientes especificaciones:

- **Tipo de comunicación:** Tal como se ha comentado en la introducción, se prefiere trabajar con el protocolo SPI dado que éste permite alcanzar una mayor velocidad de transferencia que I2C, el cual podría ser importante para la adquisición de la señal de vibración del sistema motor.
- **Rango:** Se debe seleccionar un dispositivo que disponga de un rango de medición un tanto superior a las medidas esperadas. En el sistema motor/generador, cabe esperar una vibración máxima de unos 16 g.
- **Acelerómetro de 3 ejes:** Se debe conocer la respuesta del sistema en el espacio, es decir, su movimiento en las tres direcciones del espacio, por lo que el acelerómetro buscado debe de ser capaz de proporcionar tres tensiones de salida correspondientes a cada eje X, Y , Z.
- **Ancho de banda:** El nivel de ruido depende del ancho de banda elegido, 400 Hz para esta aplicación, influyendo en la calidad de la señal que se obtiene.
- **Resolución:** Esta característica, cuanto mayor sea menor será la variación que será capaz de detectar el sensor. Por ello, se pretende un acelerómetro con la mayor resolución posible.
- **Interrupción de dato disponible:** Las interrupciones son un mecanismo de alertar el sensor al microcontrolador de ciertos eventos (i.e, datos disponibles), el cual facilitaría la realización de una función asociado a dichos eventos en el software. Por ello, el acelerómetro seleccionado deberá poseer al menos un pin de interrupción para asociarlo con una interrupción en el programa cuando éste cambie de estado.
- **Consumo:** En principio el sistema de monitorización se alimentará con la fuente de alimentación conectada a la red, el consumo de los dispositivos no es un factor primordial para su elección. Aun así, siempre es preferible utilizar dispositivos de bajo consumo para ser respetuoso con el medio ambiente.

- **Bajo coste:** Puesto que el objetivo es académico y se trata de mostrar la creciente utilización de los sensores inteligentes, se busca el sensor más económico que logre cumplir en el resto de funcionalidades por motivos de presupuesto.

Atendiendo a estas especificaciones se ha realizado una selección y comparación de varios acelerómetros comerciales de 3 ejes comúnmente utilizados. Esta comparativa se detalla en la siguiente tabla:

Dispositivo	**Rango**	**Interfaz**	**Ancho banda (HZ)**	**Resol. (bits)**	**Consumo**	**Pines interr.**	Precio (€)
ADXL345	$\pm 2g/\pm 4g/\pm 8g/\pm 16g$	(I2C y SPI)	1600	13	3.3V a 5V, 40µA	2	1.9
LIS331	$\pm 6g/\pm 12g/\pm 24g$	(I2C y SPI)	1000	16	2.16V a 3.6V, 10µA	2	8.7
MMA7455	$\pm 2g/\pm 4g/\pm 8g$	(I2C y SPI)	125	10	2.4V a 3.6V, 400µA	2	3.4

Tabla 1. Comparativa de acelerómetros.

Dado que cualquier de los 3 acelerómetros cumplen los requisitos básicos de esta aplicación (rango de medida, disponibilidad de pines de interrupción y salida SPI), se atiende principalmente a dos factores: el coste y el consumo. El acelerómetro MMA7455 tiene un consumo un orden de magnitud mayor que los otros acelerómetros posibles, menor resolución y un ancho de banda menor. Además, no es el más económico de los tres, por lo que queda descartado. En cuanto al LIS331 y el ADXL345, no decantamos por el acelerómetro ADXL345 por su bajo y su cumplimiento de los demás requisitos.

Figura 16. Acelerómetro ADXL345.

4.2. SENSOR DE TEMPERATURA

En el mercado existen distintos tipos de sensores para la medida de temperatura: termopar, RTD, termistores, piroeléctricos y los sensores basados en semiconductores. Asimismo, existen sensores de temperatura con salida analógica, digital y mediante una interfaz de conexión modulada por ancho de impulso, conocida como PWM. Atendiendo a la necesidad del proyecto, se han establecido las siguientes especificaciones para su elección.

- **Interfaz de comunicación.** Tal como se ha comentado anteriormente, se precisa un sensor de vibración con salida SPI para la adquisición de la señal de vibración del sistema motor, por lo que se prefiere emplear el sensor de temperatura con salida SPI con el fin de simplificar la programación del microcontrolador para comunicarse con los sensores de temperatura y vibración sólo por el puerto SPI.
- **Rango de medida:** El rango de medida debe estar lo más ajustado posible a los valores que se esperan medir, pues un menor rango de medida se asocia a una mayor sensibilidad. Como se espera una máxima temperatura de unos 100ºC cuando se activan las resistencias para simular la carga mecánica, y el sensor está expuesto a la temperatura ambiente cuando no las activa, por lo que se espera medir temperaturas entre 0 °C y 100 °C.
- **Precisión:** Dado que no se trata de una aplicación crítica, no se precisa sensor de temperatura de mucha precisión. En principio se considera aceptable una precisión de $\pm$ 1°C, lo que supone un error máximo del 1%.
- **Resolución:** Dado que no se trata de una aplicación crítica, no se precisa sensor de temperatura de mucha resolución, aunque es preferible utilizar un sensor de mayor resolución.
- **Bajo coste:** Puesto que el objetivo es académico y se trata de mostrar la creciente utilización de los sensores inteligentes, se busca el sensor más económico que logre cumplir en el resto de funcionalidades por motivos de presupuesto.
- **Bajo consumo:** A pesar de que el sistema de monitorización se alimentará con una fuente de alimentación, siempre es conveniente seleccionar los componentes con menor consumo, para así respetar el medio ambiente.

En la Tabla 2 se muestra el estudio comparativo de dos sensores que cumplen los requisitos establecidos.

Dispositivo	Rango (°C)	Interfaz	Precisión	Resol. (bits)	Consumo	Precio (€)
ADT7310TRZ	-55 – 150	SPI	$\pm$ 0.5 ºC	16	2.7V – 3.6V 0.7 mW	3.03
ADT7301ARTZ-500RL7	-40 – 150	SPI	$\pm$ 0.5 ºC	13	2.7V – 5.25V 0.6 – 1.4 mW	2.96

Tabla 2. Comparativa sensores de temperatura.

Primero ambos sensores presentan rangos de medidas similares y la misma precisión. Aunque el ADT7301 es ligeramente más económico, su consumo en funcionamiento normal es superior al ADT7310. En este proyecto se ha optado por el sensor ADT7310 por su bajo consumo y mayor capacidad de resolución.

Figura 17. Sensor de temperatura ADT7310TRZ

4.3. MICROCONTROLADOR

El microcontrolador es el responsable de leer los datos del sensor de temperatura y acelerómetro por el puerto SPI y las señales analógicas provenientes del sistema motor/generador, y transmitirlos al PC vía USB o vía Bluetooth. Asimismo, se encarga de subir los datos a la plataforma de Nube mediante GPRS y mensajes de alerta ante cualquier situación inesperada vía GSM.

Atendiendo a la necesidad del proyecto, se ha establecido las siguientes especificaciones:

- **Frecuencia:** La velocidad de procesamiento del microcontrolador. Cuando mayor sea, mayor velocidad de cómputo poseerá el microcontrolador. Se le exigirá el mínimo de 8MHz, que es la máxima velocidad de la comunicación SPI.
- **Flash:** La capacidad de memoria disponible en el microcontrolador para programar las acciones a realizar. Como mínimo serán necesarios en torno a los 26 kB.
- **Entradas y salidas digitales:** Necesarias para la interacción con los sensores. Debe contar con los pines de comunicación suficientes para los sensores instalados, además de ser útil que contara con pines adicionales para posibles ampliaciones. Como mínimo son necesarios 9 pines digitales para poder conectar el módulo GPRS y el Bluetooth, no ambos a la vez, y 13 pines digitales para poder conectar ambos dispositivos simultáneamente (además de los sensores de aceleración y temperatura).
- **Entradas analógicas:** Debe de poseer al menos 3 entradas analógicas para las señales analógicas del sistema motor/generador (tensión en bornes del motor, corriente que circula por el motor y la velocidad de giro del motor).
- **Comunicación:** Deben disponer al menos 1 puerto de SPI para interaccionar con el acelerómetro y el sensor de temperatura instalados.

- **Precio:** Debemos de tener en cuenta el precio de estos componentes. No se desea que el precio sea muy elevado, puesto que en principio el objetivo es académico, y las unidades de las que se debe disponer pueden ser relativamente elevadas.

Con los criterios expuestos en el punto anterior, se procede a realizar una comparativa entre los 4 microcontroladores más comunes en el mercado actual (Tabla 3 y Tabla 4):

Dispositivo	Frec.	RAM	Flash	I/O Digital	I/O Analog	SPI
Arduino UNO	16 MHz	2 kB	32kB	14	6 (10 bit)	1
BeagleBone	700 MHz	256 MB	4GB (microSD)	66	7 (12 bit)	1
Raspberry Pi	700 MHz	256 MB	SD Card	8	N/A	1
MSP-EXP430G2	16 MHz	512 bytes	16kB	16	8 (10 bit)	2

Tabla 3. Comparativa de microcontroladores

Dispositivo	Consumo mín. (W)	€	Facilidad de programación	Necesidad de licencia
Arduino UNO	0.3	21.2	Simple	No – Libre
BeagleBone	0.85	59.3	Compleja	No – Libre
Raspberry Pi	3.5	32.9	Compleja	No – Libre
MSP-EXP430G2	0.01	13,2	Simple	Sí

Tabla 4. Comparativa de microcontroladores

En principio, cualquier de estos cuatro microcontroladores/microprocesadores cumple los requisitos básicos para la aplicación específica del presente TFM. La velocidad de reloj es muy similar en la Raspberry Pi y en la BeagleBone, sin embargo, en pruebas realizadas por analistas mostraron que la BeagleBone funcionaba en torno a dos veces más rápida que la Raspberry Pi. Ambos son pequeños ordenadores que pueden ejecutar su propio sistema operativo y, con ello, ejecutar varios programas al mismo tiempo. Además, pueden ser programados en varios lenguajes. Una de las características más interesantes de la Raspberry y la BeagleBone es que su memoria Flash está constituida por una tarjeta SD, lo que les atribuye una mayor capacidad y ampliación, frente a los 32 kB de serie de Arduino. Así mismo, podemos tener diferentes configuraciones en diferentes tarjetas SD. En comparación con Arduino y el MSP-EXP430G2, la velocidad de reloj y la frecuencia del Raspeberry Pi y BeagleBone es 44 veces mayor, y la memoria RAM es 128000 veces menor. No obstante, la Raspberry y la BeagleBone funcionan con sistema operativo Linux, lo que hace más compleja su configuración.

En cambio, las características técnicas del Arduino y MSP-EXP430G2 son muy similares. Sin embargo, a pesar de que el MSP-EXP430G2 es más económico, su memoria podría no ser suficiente para contener el código del programa a desarrollar. Además, es necesaria una licencia para la configuración del MSP-EXP430G2, por lo que queda descartado fuera del estudio. En cambio, Arduino es una plataforma abierta, es decir, no requiere licencia de software y de bajo coste. Además, existen una gran cantidad de códigos de ejemplo en el internet que facilita el desarrollo de aplicaciones. Más aún, se puede conectar módulos de comunicación, i.e, Wifi, GPS, Bluetooth mediante las Shields. Dada la aplicación específica del presente TFM es relativamente sencilla en la que no requiere una gran capacidad de cómputo, se ha decantado por el Arduino. En la Figura 18 se muestra el kit de evaluaciones del Arduino Uno.

Figura 18. Arduino Uno

En la Figura 80 del ANEJO II se muestra el un esquema del kit de evaluación Arduino Uno.

4.4. SUBSISTEMA DE COMUNICACIÓN

Tal como se comentó anteriormente, los datos recogidos del sistema motor/generador se envían al PC mediante cables o de forma inalámbrica y a su vez a la plataforma de Nube mediante GPRS. Asimismo, debe poder enviar mensajes de alerta vía GSM ante cualquier situación inesperada. En esta sección, se seleccionará los dispositivos necesarios que conforman el subsistema de comunicación.

4.4.1. COMUNICACIÓN ENTRE EL SISTEMA MOTOR Y EL PC LOCAL

De acuerdo con las especificaciones del proyecto, el sistema debe contemplar la posibilidad de transmitir inalámbricamente los datos recopilados a un PC local. El subsistema de comunicación debe cumplir los siguientes requisitos:

- Distancia de comunicación: Dado que en general el PC local está situado físicamente cercano al sistema motor/generador (menos de 3 m), nos centramos en las tecnologías de comunicación inalámbrica de área personal que se utiliza para la conexión de dispositivos portátiles muy próximos entre sí, sin necesitar para esta conexión cables.
- Tasa de transferencia: La tasa de transferencia de la comunicación inalámbrica debe ser superior a 31.2 kbps, siendo esta la cantidad de datos mínima en cada envío de datos.

- Topología de red: En principio se prevé la necesidad de implementar un sistema de comunicación punto a punto, es decir, se trata de establecer comunicación entre el PC local y el nodo emisor.
- Bajo consumo.
- Bajo coste.

Los protocolos de comunicación de las redes inalámbricas de área personal más utilizados en el área de instrumentación son:

- Bluetooth: Se utiliza para eliminar cables de conexión entre dispositivos de uso personal. Esta es una tecnología inalámbrica de bajo consumo y bajo coste. En función de la clase de dispositivo, los consumos rondan desde los 100 mW a 1 mW, y el coste está en torno a los 5 €. La tasa de transferencia también es dependiente de la versión de módulo bluetooth y nos encontramos en el rango de 700 kbps hasta 3 Mbps.
- ZigBee: Se emplea en la comunicación de radiodifusión digital con un reducido consumo mediante un protocolo de alto nivel. Esta tecnología tiene un ultra-bajo consumo, generalmente inferior al bluetooth (versión 2 y versión 3). Es posible encontrar módulos entorno al precio del bluetooth, pero la tasa de transferencia de datos es más baja, entre los 10 kbps y los 115.2 kbps.

Dispositivo	Consumo en reposo	Alcance	Tasa de transferencia	Coste €
Bluetooth	~40 mA	1 m – 100 m	700 kbps – 1 Mbps	~5
ZigBee	~30 mA	10 m – 75 m	10 kbps – 115.2 kbps	~6.5

Tabla 5. Comparativa Bluetooth vs ZigBee

En este proyecto, se ha escogido la tecnología Bluetooth dada la tasa de transferencia requerida.

El módulo Bluetooth empleado es el HC-05 (ver Figura 19), por su disponibilidad en el laboratorio y su fácil integración en el sistema. Además, el manejo del protocolo Bluetooth es totalmente transparente para el programar, es decir, desde el microcontrolador sólo se necesita gestionar el puerto UART (Rx y Tx) para enviar y recibir datos de forma inalámbrica.

Las especificaciones de este módulo se muestran en la siguiente la Tabla 6:

Figura 19. Bluetooth HC-05

- Radio Chip: CSR BC417
- Memoria: Flash externa 8 Mbit
- Potencia de salida: -4 a 6 dbm Clase 2
- Sensibilidad: -80 dbm
- Velocidad de bit: EDR, hasta 3 Mbps
- Interface: UART
- Antena: embebida
- Dimensiones: 27 mm x 13 mm
- Voltaje: 3.1 a 4.2 Vdc
- Corriente máx: 40 mA

Tabla 6. Especificaciones del Módulo Bluetooth HC-05.

4.4.2. COMUNICACIÓN ENTRE EL SISTEMA MOTOR Y LA NUBE

De acuerdo con las especificaciones del proyecto, el sistema debe poder enviar los datos recopilados a una plataforma de Nube mediante GPRS/GSM al mismo tiempo y mensajes de alerta vía GSM ante cualquier situación indeseada.

GPRS (General Package Radio Service, “Sistema general de paquetes vía radio” en castellano) se trata de un sistema de comunicación para la transmisión de datos de forma remota. Su funcionamiento se basa en la conmutación de paquetes, por lo que diferentes usuarios pueden estar compartiendo el mismo canal de transmisión, es decir, el ancho de banda es ocupado con usuarios que envían datos en un mismo instante.

GSM (Global System for Mobile communications, “Sistema Global para las comunicaciones Móviles” en castellano) se trata de uno de los estándares inalámbricos 2G más importantes para que una red digital de teléfono móvil sea capaz de soportar voz, datos, mensajes de texto y roaming.

Las especificaciones requeridas del módulo de comunicación serán:

- Comunicación de datos vía GPRS/GSM
- Bajo consumo.
- Bajo coste.

En la siguiente tabla (Tabla 7) se muestran las características de cada uno de los módulos seleccionados, y a partir de estas se determinará cuál se escogerá para la aplicación.

	GSM/GPRS	Ancho de banda (MHz)	Alimentación (V)	Consumo	€
SIM800L	85.6 kbps (↓/↑)	Quad-band 850/900/1800/1900	3.4 – 4.4	450 mA	7.6
SIM808	85.6 kbps (↓/↑)	Quad-band 850/900/1800/1900	3.4 – 4.4	540 mA	42.9
SIM900	85.6 kbps (↓/↑)	Quad-band 850/900/1800/1900	3.2 – 4.8	540 mA	52.5
SIM5320E	85.6 kbps (↓), 42.8 kbps (↑)	Dual-Band UMTS/HSDPA 900/2100 Quad-band 850/900/1800/1900	3.3 – 4.2	540 mA	68.7

Tabla 7. Comparativa módulos comunicación GSM/GPRS. (↓ d /↑ c)

Las cuatro opciones propuestas disponen de lo fundamental, capacidad de envío y recepción de datos y mensajes mediante GPRS. Entre ellos, cabe destacar que el módulo SIM 800L, presenta el inconveniente de que no dispone de ranura para la tarjeta SIM, por lo que sería necesario un montaje nuevo para su implementación y adaptación a Arduino Uno. Por tanto, se ha descartado el módulo SIM800L a pesar de su menor precio. Tanto en el SIM808, SIM900 y SIM5320E disponemos de la capacidad de enviar SMS y una ranura para la tarjeta SIM. Por ello, los tres cumplen nuestro requisito fundamental.

En cuanto el consumo, se observa que para las mismas condiciones las tres opciones tienen consumos similares. Por tanto, la selección del módulo GPRS/GSM se basó en el criterio económico optando finalmente por el módulo SIM808 (ver Figura 20).

Figura 20. Módulo SIM808.

4.5. ACONDICIONAMIENTO DE LAS SEÑALES ANALÓGICAS PROCEDENTES DEL SISTEMA MOTOR

Además de las señales de temperatura y vibración, se pretende monitorizar la tensión en borne del motor, la corriente que circula por el motor y la velocidad de giro del mismo. Tal como se ha mencionado en el apartado *SISTEMA MOTOR/GENERADOR*, la tensión en borne del motor varía entre $\pm 55\ V$ y en función del sentido de giro del motor, y ya se dispone de una instrumentación en

el sistema motor que convierte la señal diferencial en una señal unipolar referida a masa y presenta una ganancia de 0.1 V/V. De igual modo, para la obtención de la corriente que atraviesa el motor se tiene una instrumentación que presenta una ganancia de 1 V/A. Asimismo se tiene una instrumentación de procesado de señal de manera que la velocidad de giro del motor puede medirse mediante una tensión continua con ganancia 0.003 V/RPM. En la Tabla 8 pueden encontrar los rangos de medida de estas señales. Más aun, estas señales presentan un serio problema de ruido debido al control electrónico de la velocidad de giro del motor mediante el puente H.

Por otro lado, las entradas analógicas del Arduino presentan un rango de medida entre 0 y 5 V ya que el microcontrolador se alimenta a 5 V. No obstante, la gran mayoría de los microcontroladores comerciales se suele alimentar con 3.3 V. Además, se ha determinado experimentalmente que los rangos de las señales analógicas difieren entre los distintos puestos de trabajo. Por todo ello, es preferible diseñar el circuito adaptador de los niveles de tensión para que sea compatible con la entrada de cualquier microcontrolador ([0, 3.3]V) y que la señal salida del circuito adaptador siempre se encuentra en el rango de 0 a 3.3 V. Por lo que se precisa un circuito para adaptar los niveles de tensión y filtrar los componentes no deseados embebidos en la señal.

En la Tabla 8 se puede ver un resumen de las tensiones máximas y mínimas obtenidas de cada señal analógica.

Variable	Señal tensión mínima	Señal tensión máxima
Intensidad	-4.33 V	4.12 V
Tensión en bornes	-2.94 V	2.73 V
Velocidad de giro	-9.90 V	9.93 V

Tabla 8. Señales de tensión máxima y mínimas en el sistema motor/generador.

Para ello, se ha diseñado un circuito de acondicionamiento (Figura 21) que consiste en dos etapas, donde la primera etapa es un amplificador inversor con ganancia unitaria y filtrado paso bajo de orden 1 con frecuencia de corte de 10 Hz (Figura 23) y la segunda etapa se encarga de adaptar los niveles de tensión, es decir, atenuación de señal y desplazamiento de nivel.

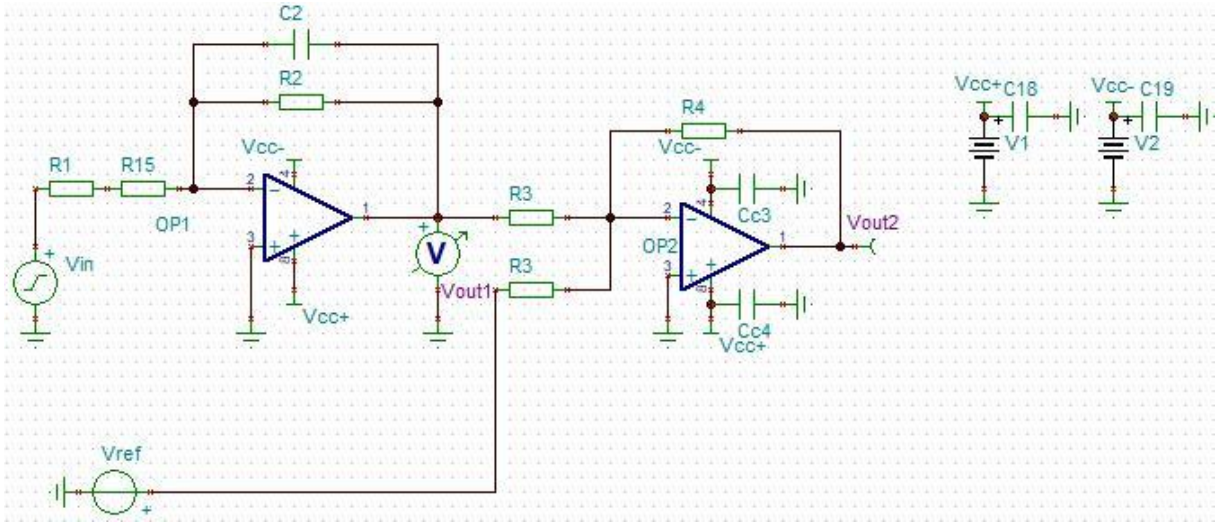


Figura 21. Esquema electrónico acondicionador de señal.

La función de transferencia asociada al filtro paso bajo es: este circuito electrónico es:

$$\frac{V_{o\ \ 1}(s)}{V_{i1}\ (s)} = \frac{\frac{1}{C_2 \cdot (R_1 + R_1\ )}}{s + \frac{1}{C_2 \cdot (R_1 + R_1\ )}}$$

A continuación, se muestran los valores de componentes para cada señal analógica del sistema (Figura 22, Figura 25, Figura 28) junto con los resultados de simulación: diagrama de bode (Figura 23, Figura 26, Figura 29) y tensión de salida (Figura 24, Figura 27, Figura 30) ante una señal senoidal simulando la tensión de salida de las señales analógicas del sistema motor.

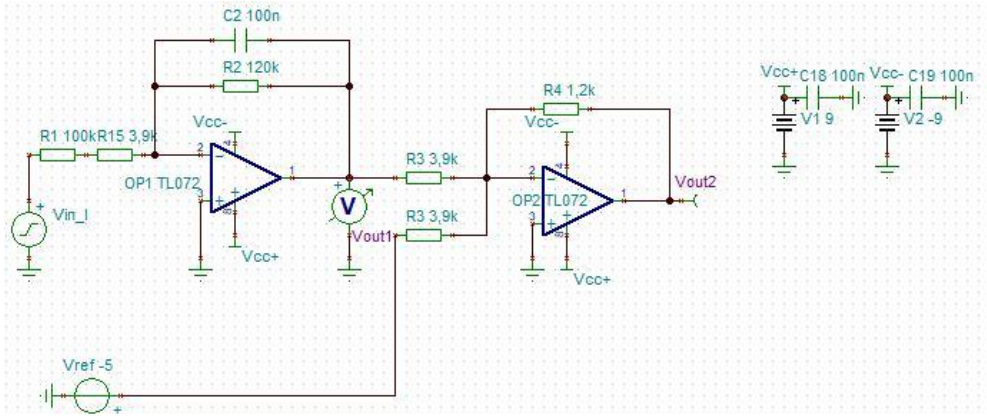


Figura 22. Circuito de acondicionamiento de la señal de intensidad.

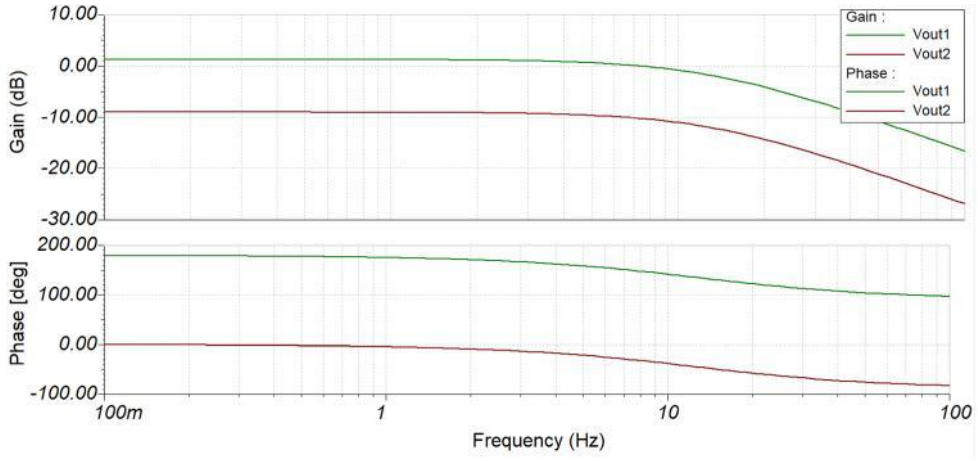


Figura 23. Diagrama de bode del acondicionamiento de la señal de intensidad.

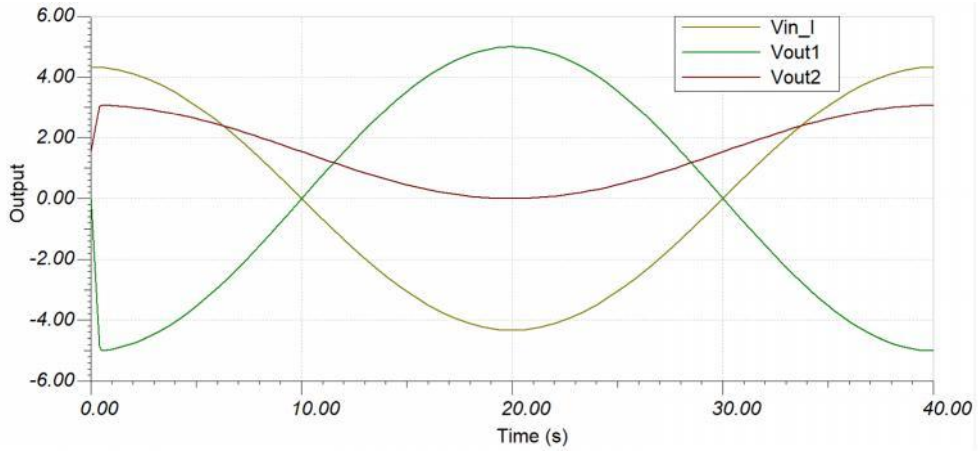


Figura 24. Salida de la señal de intensidad acondicionada en función de la entrada.

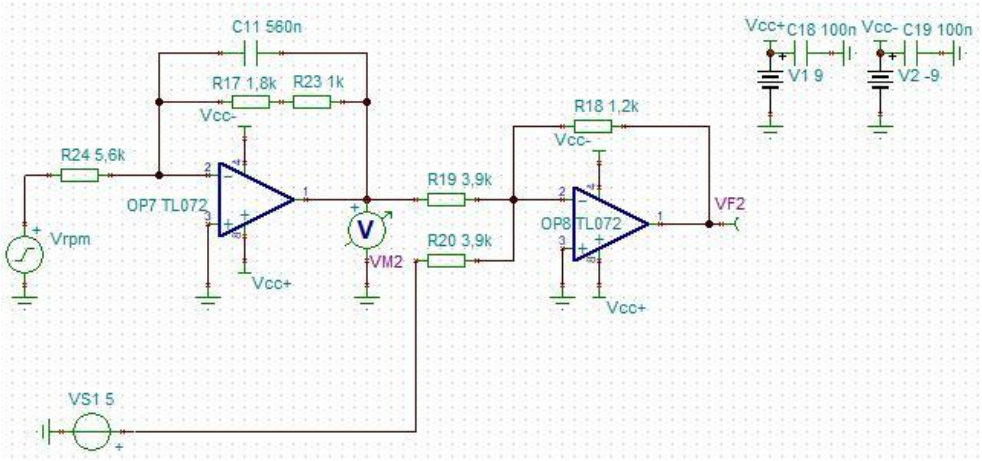


Figura 25.Circuito de acondicionamiento de la velocidad de giro.

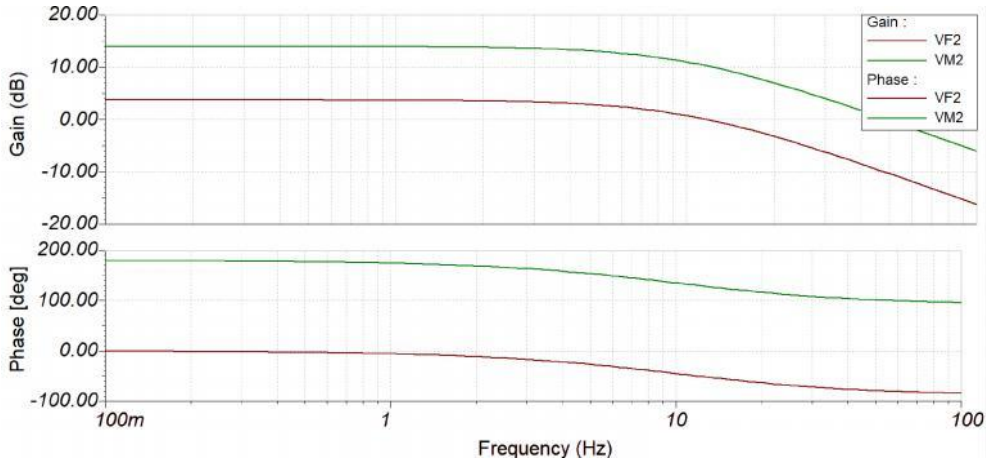


Figura 26. Diagrama de bode del acondicionamiento de la señal de rpm.

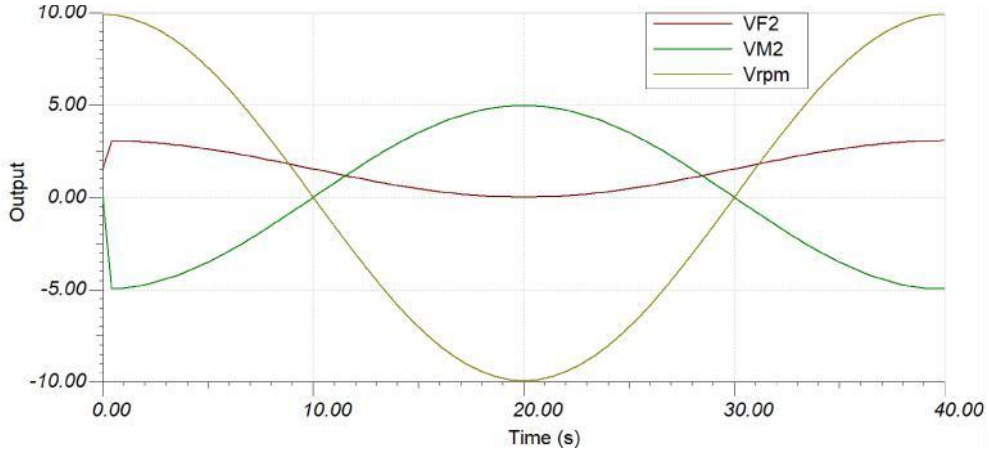


Figura 27. Salida de la señal de rpm acondicionada en función de la entrada.

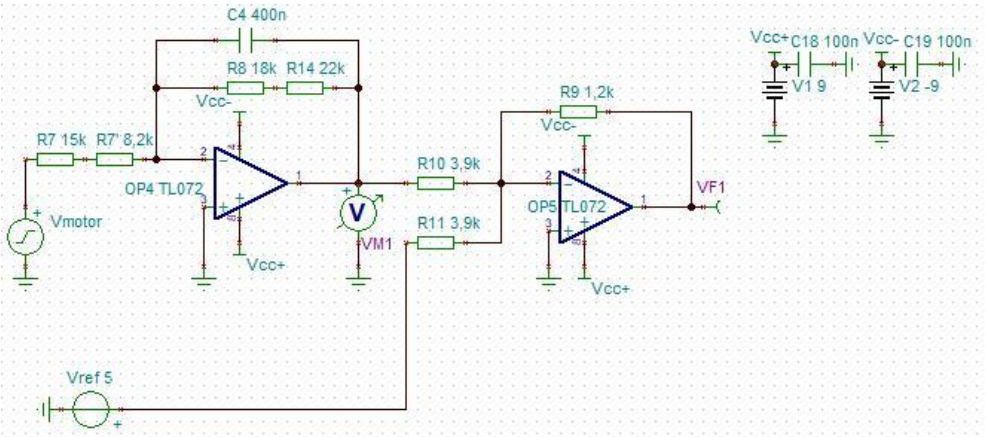


Figura 28. Circuito de acondicionamiento de la tensión en bornes del motor.

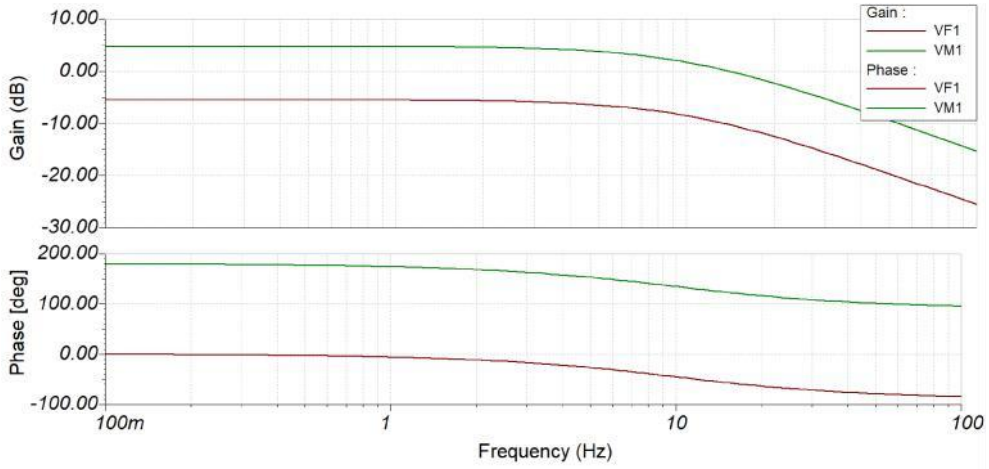


Figura 29. Diagrama de bode del acondicionamiento de la señal de tensión en bornes.

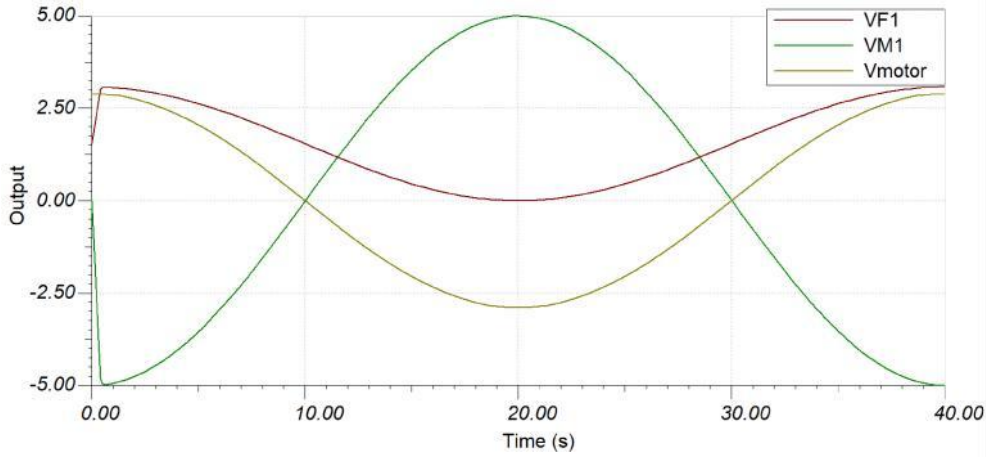


Figura 30. Salida de la señal de tensión en bornes acondicionada en función de la entrada.

Se atienden a una serie de especificaciones para la selección del operacional empleado en el montaje de este circuito de adaptación:

- **Corriente consumida:** que sea la mínima posible
- **Rechazo a modo común (CMRR):** rechazar las señales presentes simultáneamente en ambas entradas. De forma ideal, las señales comunes no influyen sobre la salida, sin embargo, para un amplificador operacional se define este factor como CMRR = ganancia diferencial/ganancia en modo común.
- **Separación de canales:** al encontrarse dos amplificadores en un encapsulado, suelen presentarse interferencias entre ellos. Se mide en dB este factor que representa cuán bien son rechazadas o atenuadas las señales presentes en la entrada de un amplificador operacional.

Puesto que la señal es de continua, el nivel de ruido no es muy importante. Asimismo, no se requieren grandes presiones ya que la señal ya está acondicionada y en esta aplicación

principalmente hay que atenuar la señal. De este modo, principalmente interesa un amplificador operacional de bajo coste, de propósito general, y que un mismo chip contenga dos operacionales, ya que cada circuito de acondicionamiento requiera dos amplificadores operacionales. Con esto, se seleccionan amplificadores operacionales que contienen dos de ellos por cada chip. En la Tabla 9 se encuentran un resumen de las características que se analizan:

	Corriente consumida	**Rechazo a modo común**	**Separación de canales**	**€**
LM358-N	1 mA	85 dB	120 dB	0.708
TL072	1.4 mA	100 dB	120 dB	0.623
TL082	1.4 mA	86 dB	120 dB	0.484
LF353-N	3.6 mA	100 dB	120 dB	0.579
AD8002	5.0 mA	54 dB	60 dB	5.81

Tabla 9. Amplificadores operacionales para el circuito de acondicionamiento.

Se elige el amplificador operacional TL072 para esta aplicación, por su mejor comportamiento y relativo bajo consumo a uno de los menores precios.

Subsecuentemente se procede a un montaje en protoboard para comprobar su correcto funcionamiento.

Para ello se realiza el montaje en una protoboard de Frizing, un programa que permite realizar montajes eléctricos, pero no su funcionamiento de forma gratuita, por lo que simplemente sirve de guía para su posterior montaje (Figura 31):

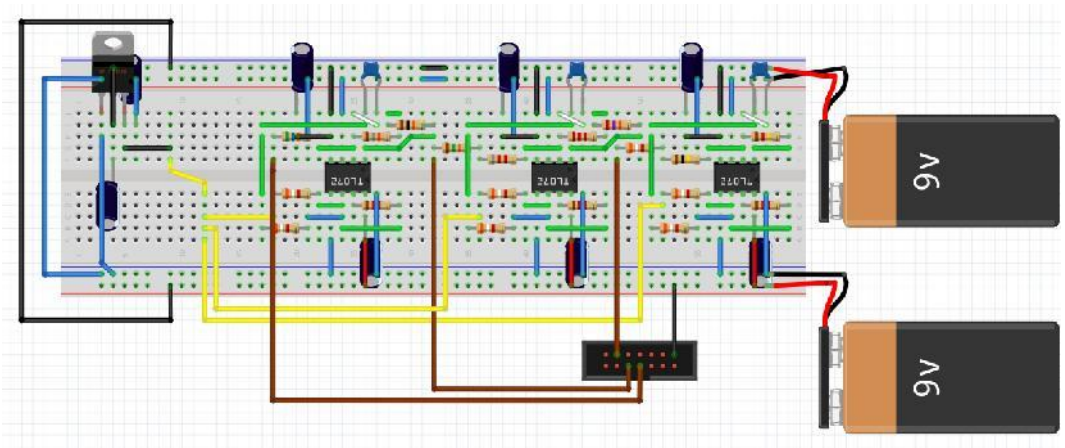


Figura 31. Montaje en Flipboard

Donde se pueden observar las distintas partes en la Figura 32. Partes del montaje en Flipboard.:

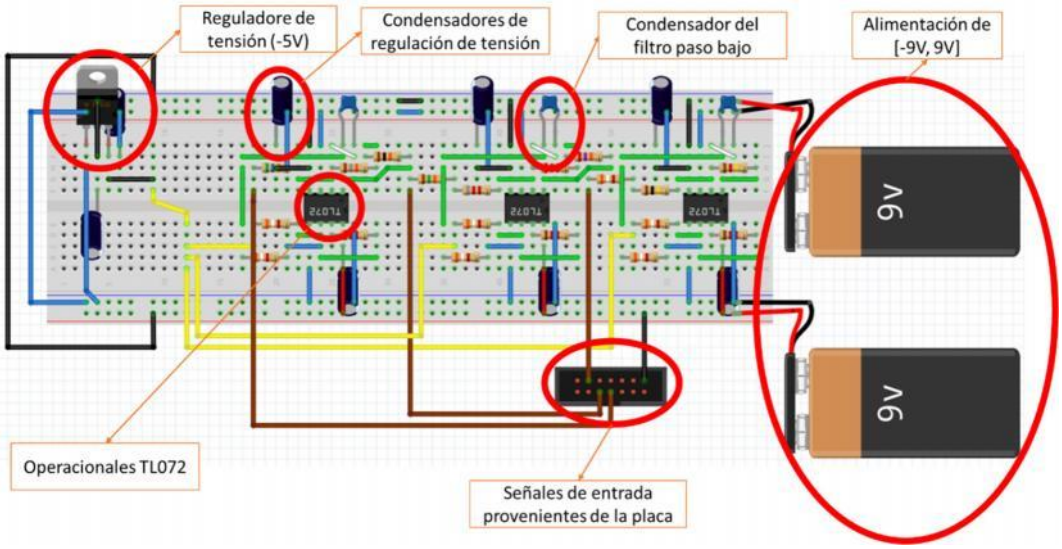


Figura 32. Partes del montaje en Flipboard.

En las Figura 33 y Figura 34 se muestra el montaje físico del circuito de acondicionamiento de las tres señales analógicas procedentes del sistema motor.

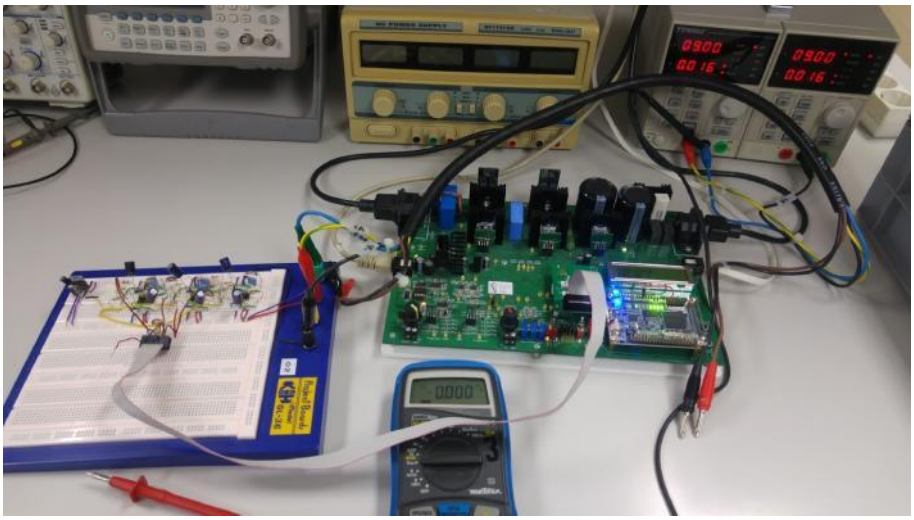


Figura 33. Montaje completo de acondicionamiento de las señales, conectado al sistema motor.

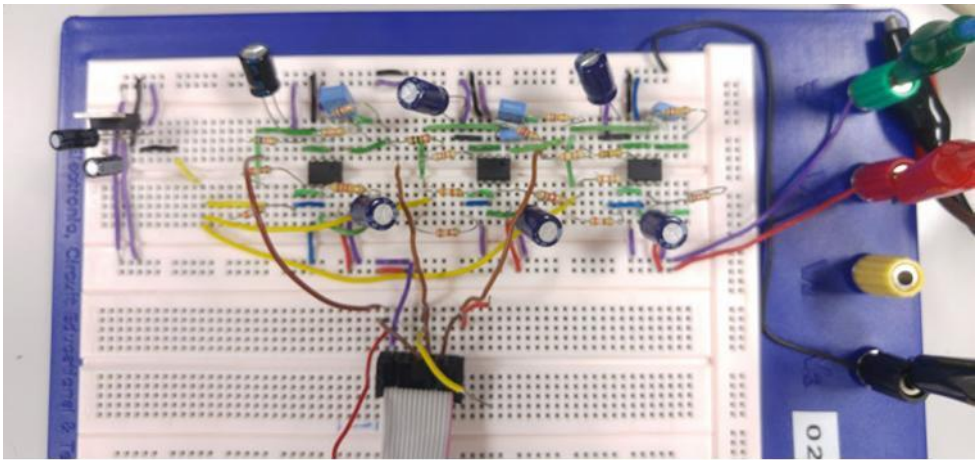


Figura 34. Detalle circuitos concionadores de las señales.

Con este montaje se verifica que lo diseñado desde el inicio funciona correctamente, obteniéndose las tres señales en un rango de [0V, +3V].

4.6. IMPLEMENTACIÓN FÍSICA DEL ARDUINO SHIELD DE DISEÑO PROPIO

Tanto el sensor de temperatura como el acelerómetro son instalados en el sistema motor (Figura 35 y Figura 36) y sus señales SPI se llevan mediante un cable plano al puerto SPI del Arduino Uno (Figura 37).

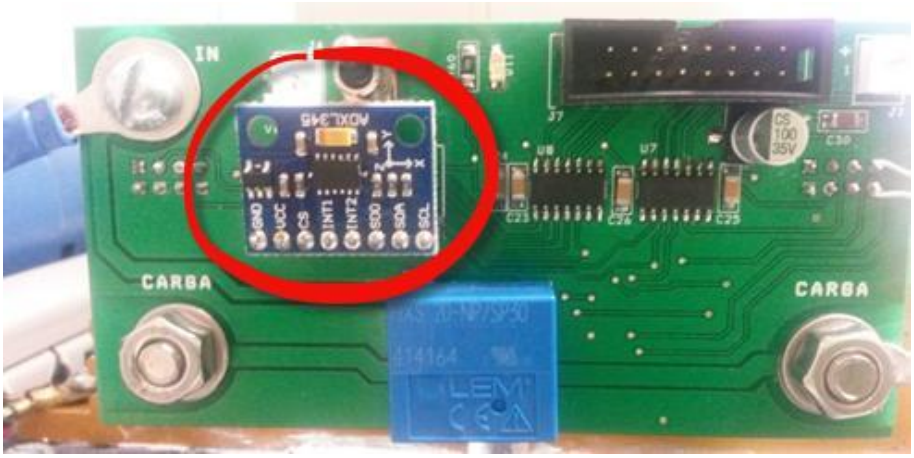


Figura 35. Acelerómetro instalado en el sistema motor.

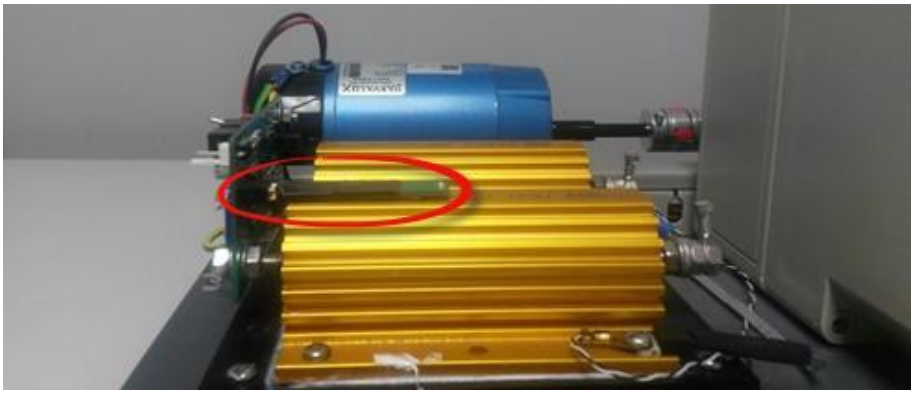


Figura 36. Sensor de temperatura instalado en el sistema motor.

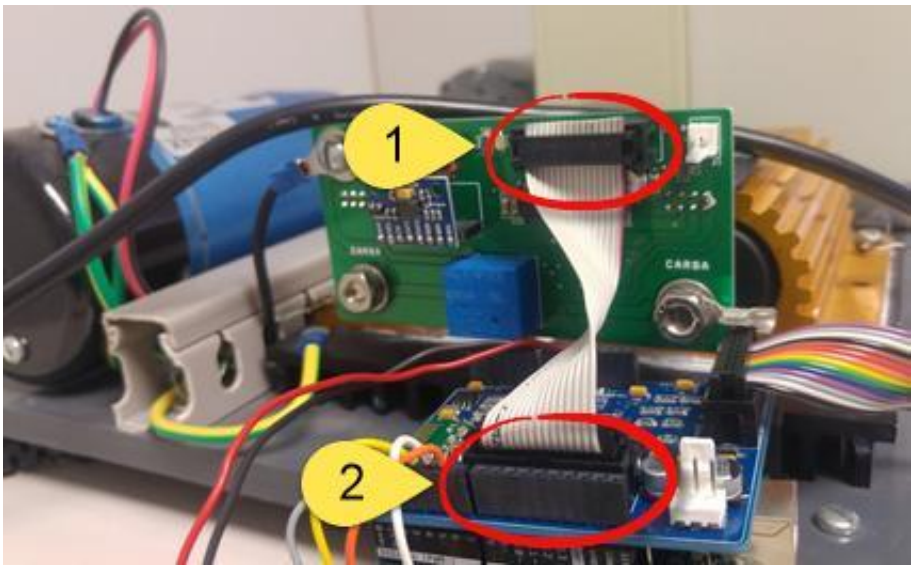


Figura 37. Señales SPI de los sensores (1) se comunican con los puertos SPI de Arduino (2) mediante cable plano.

Por otro lado, se precisa el acondicionamiento de las tres señales analógicas procedentes del sistema motor y su señal de salida se rutea a las entradas analógicas del Arduino Uno (Figura 38). El Arduino Uno es el responsable de enviar los datos recopilados a un PC local vía cableado o por Bluetooth (módulo HC05), y a una plataforma de Nube mediante GPRS (módulo SIM808). Asimismo, el microcontrolador se encarga de enviar el mensaje de alerta vía SMS ante cualquier situación indeseada.

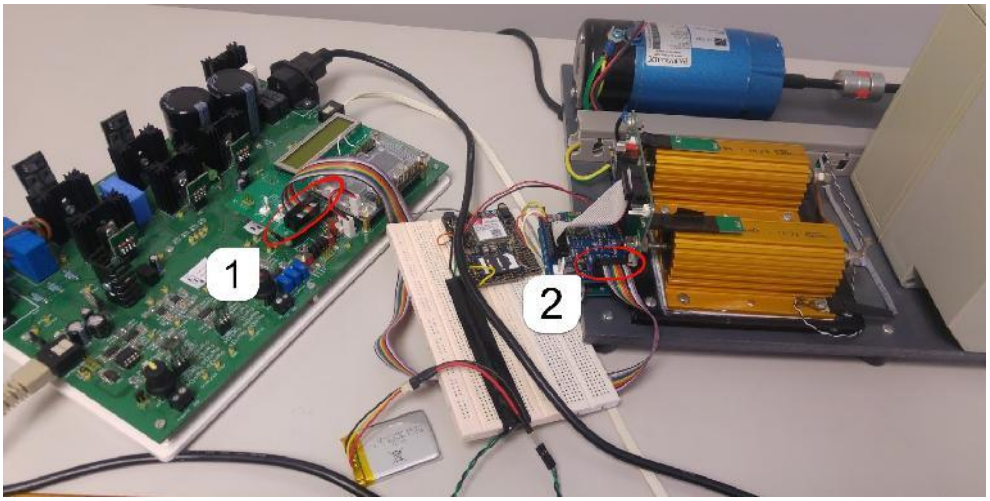


Figura 38. Señales analógicas del sistema motor (1) se comunican con las Shiel de Arduino (2) para su acondicionamiento y comunicación con las entradas analógicas de Arduino Uno.

Para cumplir todos los requisitos del proyecto, se ha optado por diseñar un Arduino Shield de diseño propio que incluye el circuito adaptador de niveles de tensión de las señales analógicas procedentes del sistema motor (ver apartado4.5) y el chip Bluetooth. Asimismo, debe poder comunicar con el sensor de temperatura y el acelerómetro instalado en la carga. En principio, la conexión con el módulo SIM808 que integra el GSM/GPRS se realiza mediante cables ya que sólo requiere la conexión de cuatro líneas (ver Figura 38). Los Arduino Shileds son placas de circuitos modulares que se apilan unas encima de otras. Estas placas se apilan cobre el propio Arduino o sobre otro Shield. Los Shield se comunican con el Arduino por los pines y normalmente se alimentan a través de Arduino mediante los pines de 5 V y masa (GND), es por ello que cada Shield debe temer el factor de forma de Arduino con el espaciado entre pines idéntico para poder encajarlo.

Para el diseño de la PCB de la Shield de diseño propio se emplea el programa de diseño de diagramas electrónicos y PCBs EAGLE (Easily Applicable Graphical Layout Editor). Este programa posee un editor de diagramas electrónicos, en el cual, los componentes pueden ser añadidos de manera simple al diagrama y enrutar los componentes entre sí. Posteriormente, es posible acceder a partir de este editor de circuitos al editor de PCBs, que dispone de un sistema de autoenrutación o de una enrutación manual. A partir de este editor se pueden obtener archivos GERBER que se han enviado a una empresa externa para su fabricación.

La PCB de la Shield debe contener la entrada de datos procedente de los sensores y de las entradas analógicas. Por ello, se incluyen en el esquemático dos entradas de cable plano para los datos, como se muestra en la Figura 39.

En la parte superior de dicha imagen nos encontramos con el conector de cable plano que comunica los sensores de temperatura y aceleración con Arduino mediante las conexiones SPI y la interrupción *INT1_ADXL345*. El conector inferior de la Figura 39 es el correspondiente a las entradas analógicas recogidas de la placa de control mediante un cable plano. De este conector se toman las conexiones referentes a la tensión en bornes del motor, intensidad y velocidad de giro (*V_MOTOR*, *V_RPM*, *V_I*, respectivamente).

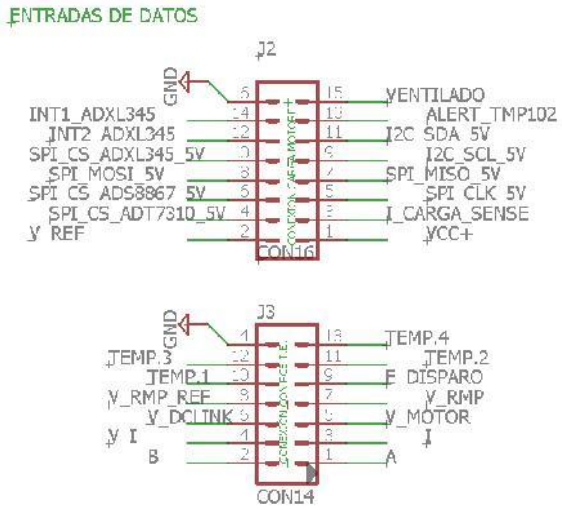


Figura 39. Esquemático: entrada de datos a la shield.

Asimismo, el Arduino Shield incluye el conector de alimentación simétrica para el circuito de acondicionamiento de señal y un led para indicar que el circuito está alimentado (Figura 40). Este conector de alimentación presenta dos condensadores de desacoplo, para estabilizar la tensión de alimentación proveniente de la fuente de alimentación conmutada del laboratorio de electrónica, reduciendo el nivel de rizado presente en la tensión de alimentación.

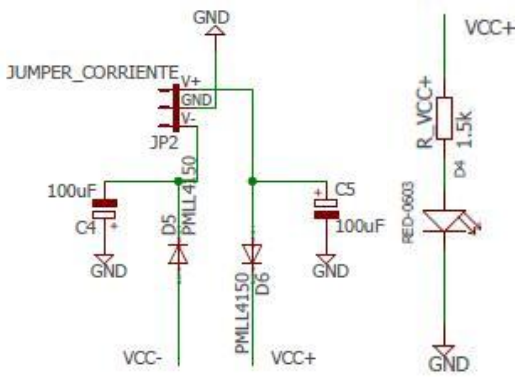


Figura 40. Conector de alimentación para el circuito de acondicionamiento junto con led de comprobación.

Asimismo se han instalado los *test points* (*CTRL_**) en el circuito de acondicionamiento para la adaptación de los niveles de tensión de la intensidad, velocidad de giro y tensión en bornes del motor descritos en el apartado4.5, (Figura 22, Figura 25 y Figura 28, respectivamente) para la comprobación de su correcto funcionamiento y la detección de posibles errores durante la fase de implementación, i.e, fallo de soldadura. Asimismo, se ha instalado dos condensadores de desacoplo (uno para la alimentación positiva y otro para la alimentación negativa) por cada integrado TL072 con el fin de minimizar el rizado de la tensión de alimentación.

En la Figura 41 se pueden ver estos circuitos de acondicionamiento para las tres señales.

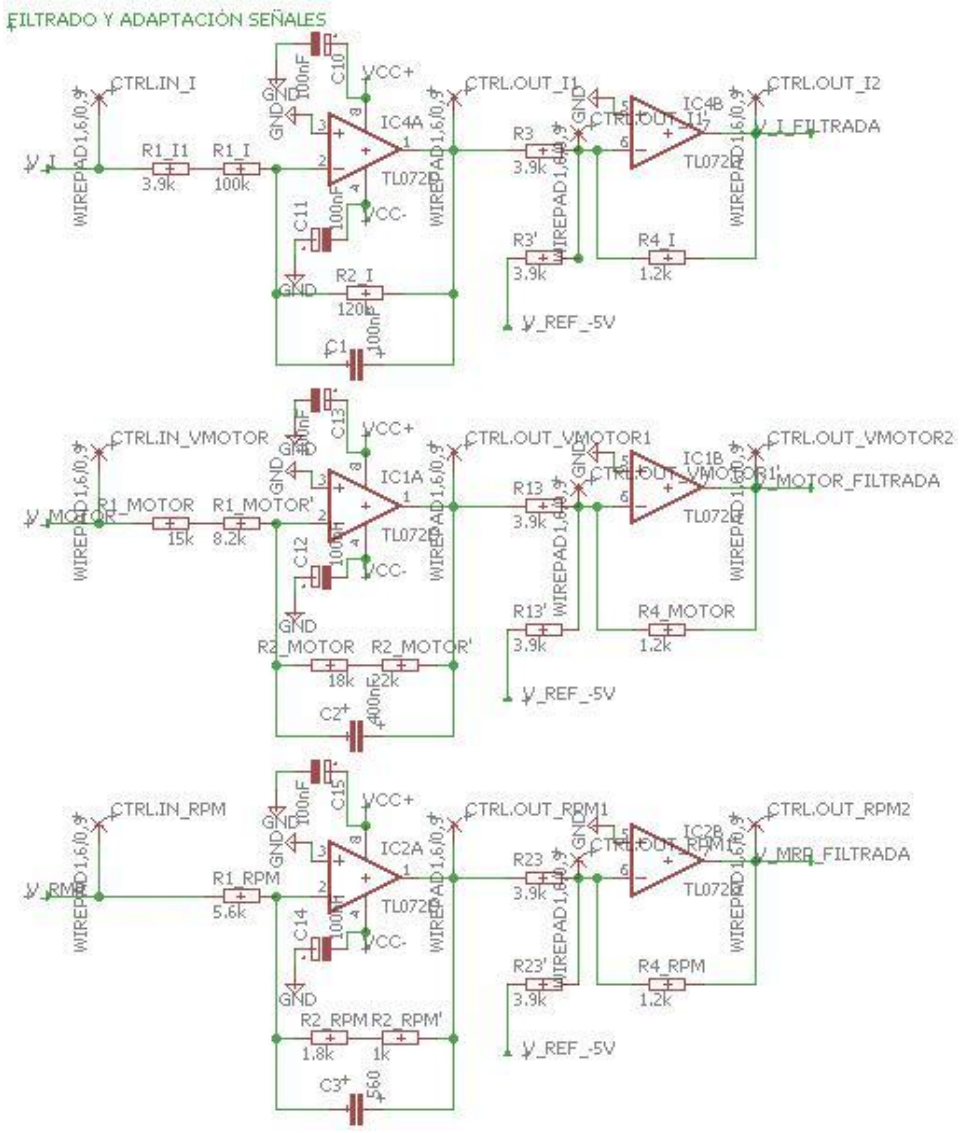


Figura 41. Esquemático circuitos de acondicionamiento de las señales analógicas procedentes del sistema motor.

Tal como se puede apreciar en la figura 41, los circuitos de acondicionamiento de la señal requieren una tensión de referencia de -5 V a la entrada de la segunda etapa para el correcto funcionamiento de los mismos. Para ello, se ha utilizado un regulador de tensión LM79L05 (Figura 42) que permite obtener una tensión continua estable de -5 V a partir de la tensión de alimentación negativa de la PCB (-9 V). Del mismo modo, se ha instalado dos condensadores de desacoplo tanto en la entrada como en la salida de este integrado para obtener una tensión de referencia estable de -5 V

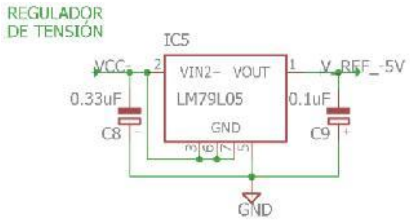


Figura 42. Esquemático regulador de tensión.

Asimismo, se incluye el módulo Bluetooth alimentado por el pin de Arduino Uno de 3.3 V en este diseño del Shield. De nuevo, se ha colocado un condensador de desacoplo para estabilizar la tensión de alimentación. Además, se incluye un diodo led que indicará si el módulo está en funcionamiento. En la Figura 43 se muestra el esquemático del módulo Bluetooth incorporado en el diseño.

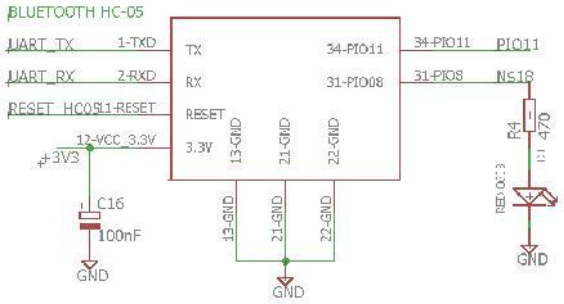


Figura 43. Esquemático bluetooth HC05

Por otro lado, tal como se ha mencionado anteriormente, el sensor de temperatura admite un rango de tensión de alimentación entre 2.7 y 5.5 V y el acelerómetro ADXL345 trabaja en el rango de 2 a 3.6 V, mientras que el Arduino Uno se alimenta a 5 V, por lo que es necesario insertar un adaptador de nivel de tensión de las señales digitales – para permitir la transferencia de datos entre el Arduino y los sensores con salida SPI. Para ello, se ha utilizado el MAX3000 (Figura 44). El MAX3000 es un traductor de nivel de 8 canales que proporciona el cambio de nivel necesario para permitir la transferencia de datos en un sistema con diferentes voltajes. Las tensiones aplicadas externamente, VCC y VL, establecen los niveles lógicos en cualquier lado del dispositivo. Las señales presentes en el lado de VL del dispositivo aparecen como una lógica de mayor tensión en el lado de VCC del dispositivo, y viceversa. A continuación, se adjunta las principales características del MAX3000:

- Emplea una arquitectura bidireccional sin la utilización de un pin direccional.
- Cuenta con una entrada (EN) que, cuando es baja, reduce las corrientes de suministro de VCC y VL a menos de 2 μA.
- Funciona con una velocidad de datos garantizada de 230 kbps en todo el rango de tensión.
- Acepta tensiones en VL de 1.2 V a 5.5 V y tensiones en VCC de 1.65 V a 5.5 V.

De nuevo, se ha colocado un condensador de desacoplo para la tensión de alimentación de 5 V y de 3.3 V.

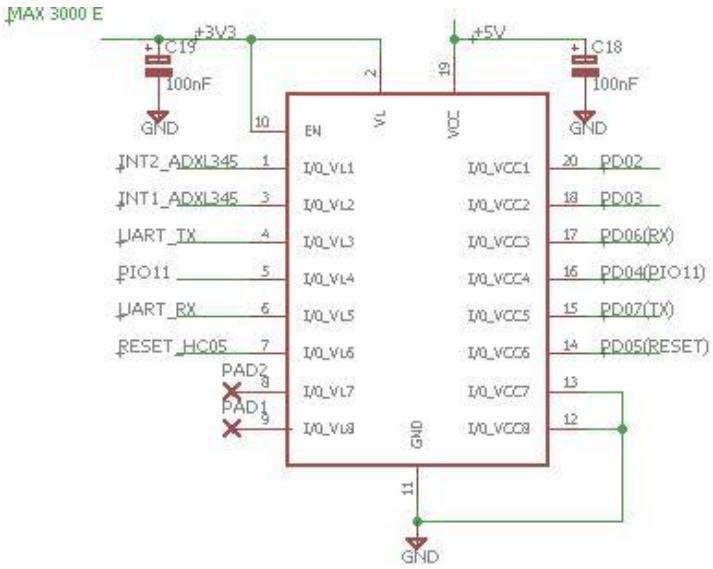


Figura 44. Esquemático MAX3000.

Como se ha mencionado anteriormente, se ha optado por una solución apilable, es necesario incluir en el esquemático los mismos pinout de la placa de evaluaciones del Arduino Uno (Figura 45) y con la misma distribución de los conectores. Las salidas de 3.3 V y 5 V están conectadas cada una a un condensador de desacoplo, y los pines *TX* y *RX* están conectados a dos diodos para conocer si durante la comunicación de datos SPI se están produciendo las interrupciones (Figura 46):

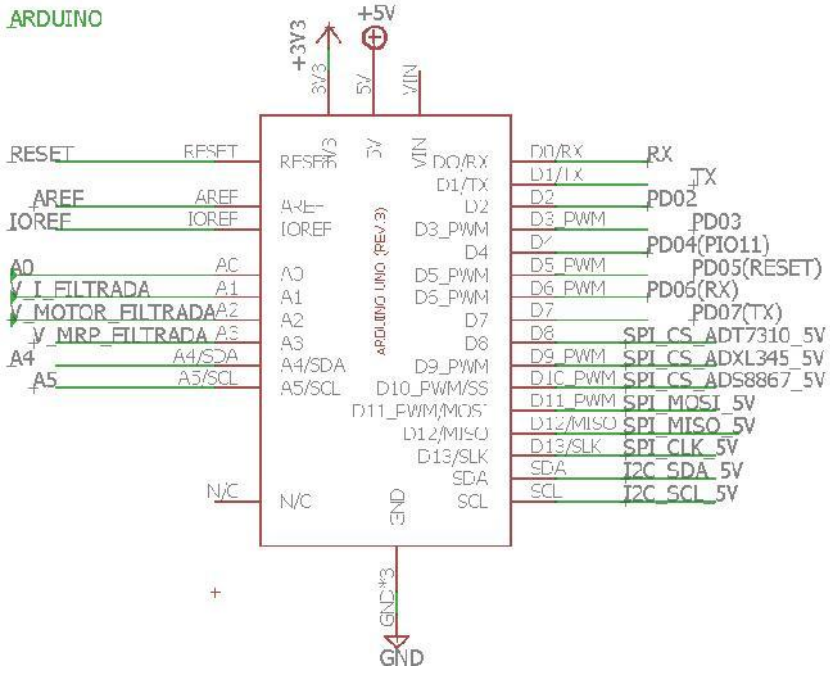


Figura 45. Esquemático Arduino.

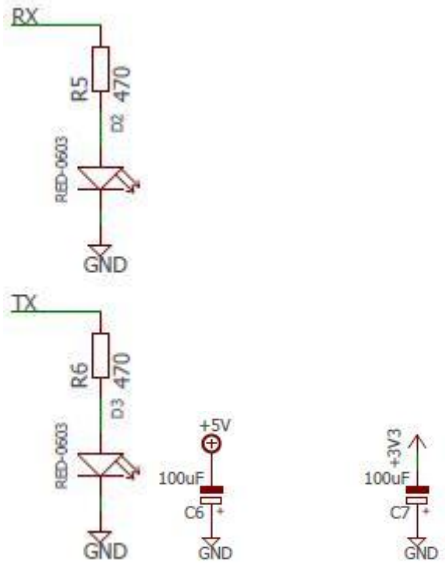


Figura 46. Leds de interrupción y condensadores de desacoplo de la Shield.

La instalación del sensor de temperatura y el acelerómetro en el sistema motor/generador se encargó el técnico de laboratorio, por lo que no se contempla el esquemático y montaje en este trabajo. En la Figura 81 del ARDUINO SHIELD se muestra el esquemático completo del Arduino Shield.

Con el esquemático realizado, se procede al diseño de la PCB. Eagle dispone los elementos colocados en el esquemático en un plano. Estos elementos deben de ser ordenados y concentrados en un plano del tamaño de la placa Arduino para que esta Shield sea apilable en él.

Los componentes se sitúan todos en la capa top (capa superior) del Shield (Figura 48), evitando de esta manera que alguno de los componentes del Shield entre en contacto con algún componente de la placa Arduino. La alimentación y conexiones externas (conectores de los cables planos) se sitúan en los bordes, para evitar distractores (cables por medio del circuito que distraigan en la búsqueda de posibles fallos). El Bluetooth también es colocado en un borde de la PCB para que no existan interferencias eléctricas durante la comunicación. Asimismo, en los laterales se sitúan los conectores apilables con la placa de evaluaciones del Arduino Uno. El resto de componentes como son resistencias, leds, condensadores y operacionales se distribuyen por el centro de la placa agrupando los componentes por la función a realizar en el circuito, i. e., todos los componentes que conforman el acondicionamiento de señal de la tensión en borne del motor están colocados físicamente uno cerca del otro.

El ancho de las pistas de enrutado viene en función de los siguientes parámetros,

- Circulación de corriente.
- Espesor de cobre de la placa.
- Temperatura ambiente.
- Temperatura de sobreelevación permitida.
- Rigidez eléctrica de la placa.

Teniendo en cuenta estos factores y la gráfica de la Figura 47 obtenida de las normas de diseño de PCBs del laboraorio, se opta por un ancho de pistas de 0.3 mm lo que no producirá un sobrecalentamiento, ya que la corriente que circulará por la misma es reducida (menor a 1 A).

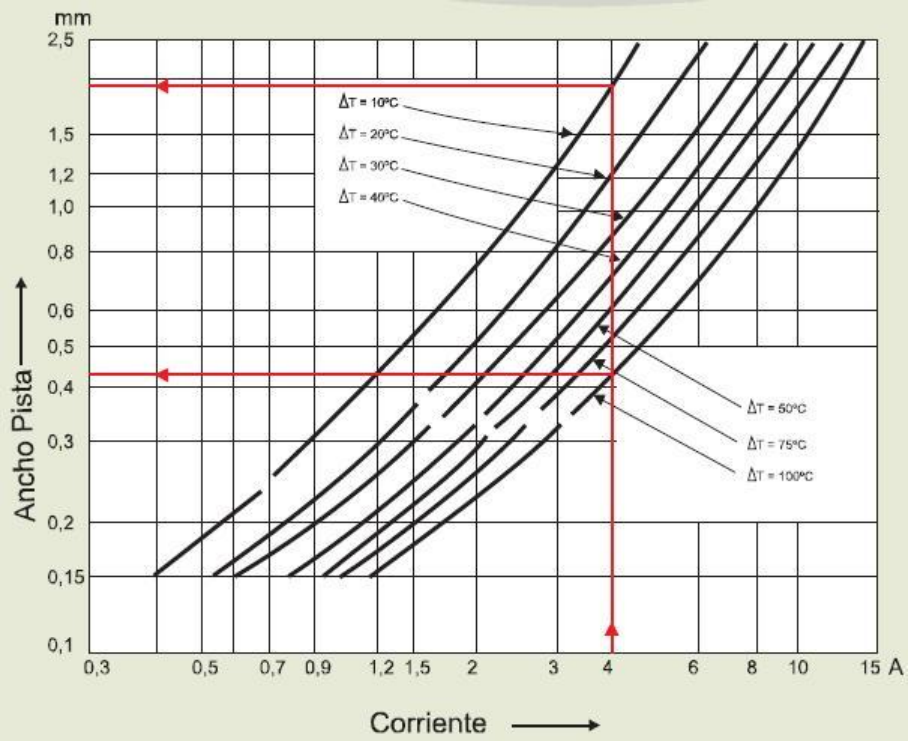


Figura 47. Ancho de pistas en PCB.

Los elementos del Shield, como son las pistas de enrutado, taladros y huellas conductoras donde se sueldan componentes, no pueden estar excesivamente juntas, pues se podrían producir interferencias y ruido en la comunicación, por lo que se deja una distancia mínima entre conductores de 0.2 milímetros.

El enrutado que no es posible realizar en la capa top se realiza en la bottom (capa inferior), para lo que son necesarios talados para comunicar ambas capas.

Tanto en la capa top como en la bottom se incluye un plano de masa para unir a él todas las conexiones de masa de los componentes, facilitando así el enrutado. El plano de masa (GND) no se puede incluir en torno a la antena del módulo Bluetooth, como se puede ver en las figuras correspondientes de los planos top y bottom (Figura 48), puesto que produciría ruido en la comunicación.

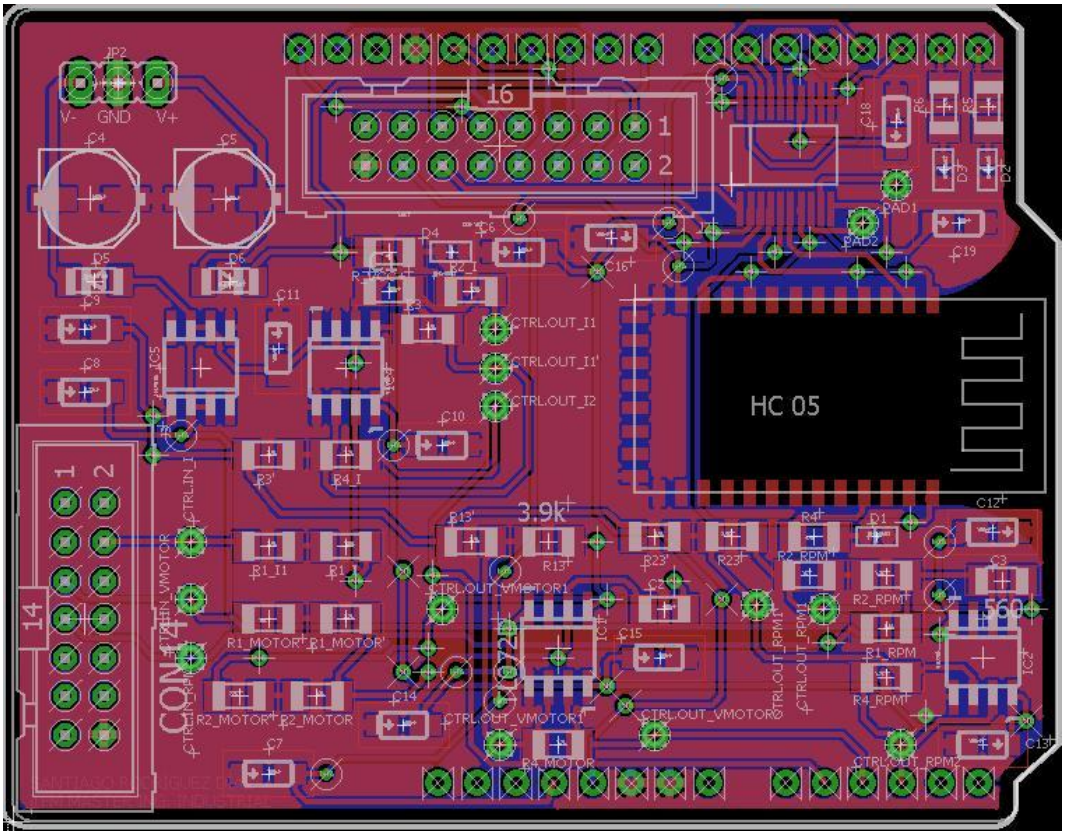


Figura 48. Layer Top y Bottom Shield de diseño propio.

A partir de este diseño podemos obtener los ficheros Gerber necesarios para enviarlos a una empresa externa para su fabricación. En la Figura 49 se muestra la implementación física del Arduino Shield de diseño propio:

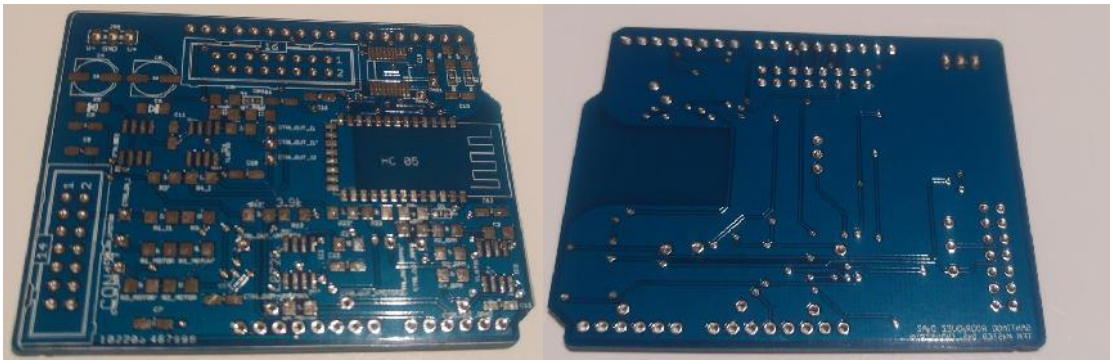


Figura 49. Implementación física del Arduino Shield de diseño propio.

En la y del ANEJO III se muestran la capa top y la capa bottom, respectivamente, de los ficheros Gerber que se han enviado para la fabricación de la Arduino Shield.

A continuación, se procede a soldar los distintos componentes de la PCB. En la Figura 50 y Figura 51 se muestra el Arduino Shield de diseño propio con los componentes soldados apilado al kit de evaluación Arduino Uno:

Figura 50. Kit de evaluación del Arduino Uno (izquierda) junto con el Arduino Shield de diseño propio (derecha) con todos los componentes soldados.

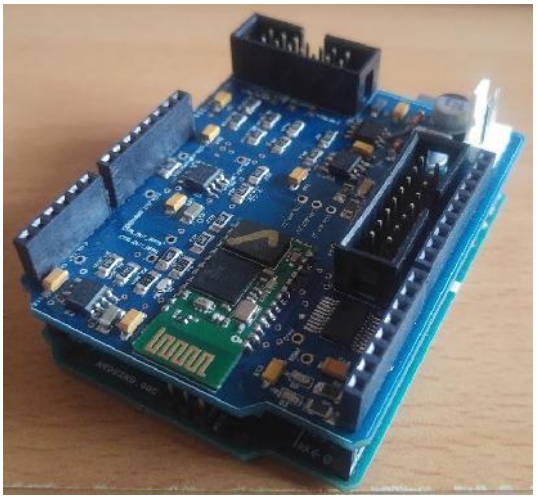


Figura 51. Kit de evaluación del Arduino Uno junto con el Arduino Shield de diseño propio con todos los componentes soldados apilado al kit de evaluación del mismo.

CAPÍTULO 5. DESARROLLO DEL SOFTWARE

En esta apartado, se describe detalladamente el desarrollo del software en el Arduino uno para la adquisición y transmisión de datos al PC y a la plataforma de Nube, y el programa de Labview para la adquisición, análisis, visualización y almacenamiento de los datos.

5.1. PROGRAMA EN EL ARDUINO

Como se ha dicho con anterioridad, Arduino UNO es como un pequeño ordenador capaz de ejecutar una serie de instrucciones escritas en forma de código introducido previamente. Es por esto que es necesario un programa en el que se puedan escribir estos códigos y capaz de introducirlos en la placa de Arduino. Este programa es el conocido como IDE (Integrated Development Environment) (Figura 52). El IDE de Arduino consiste en un editor de código, un compilador, un depurador, un constructor de interfaz gráfica y las herramientas para cargar el programa en la placa Arduino, en su memoria flash.

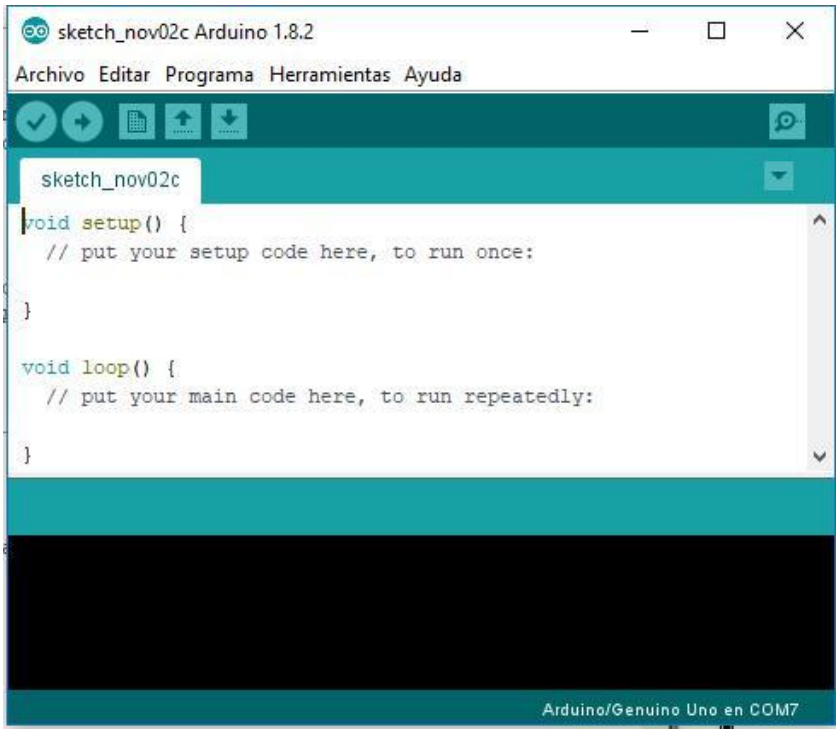


Figura 52. IDE Arduino.

La plataforma en la Nube empleada es "*io.adafruit.com*". Adafruit.IO es un plataforma IoT utilizadas para tratar los datos recogidos por los sensores y señales y luego enviados a los microcontroladores, almacenarlos y ofrecer otras funcionalidades más potentes como es la sencilla creación de dashboards.

El microcontrolador Arduino Uno es el encargado, mediante el código implementado, de recoger los datos de los sensores de vibración y temperatura por medio del puerto y configuración SPI y leer las 3 entradas analógica de tensión en borne del motor, corriente que circula por el motor y velocidad de giro. Estos datos recogidos los envía a un PC configurando una conexión mediante

cableado físico por USB o mediante la configuración de un módulo Bluetooth y mediante la configuración de un módulo GPRS los envía a una plataforma en la Nube, además de tener la capacidad de enviar mensajes de alerta vía SMS ante cualquier situación que se le programe.

Atendiendo las especificaciones del proyecto, La programación de nuestro microcontrolador Arduino uno consiste en incluir las librerías necesarias y declarar las variables que se van utilizar posteriormente en el programa y los registros del sensor de temperatura y del acelerómetro. Específicamente, se han declarado las variables globales que se emplearán en la programación para la lectura y almacenamiento de los datos recogidos, para así operar con valores. Por ello, es necesario declarar las variables para el acelerómetro ADXL345, para el sensor de temperatura ADT7310, para las señales analógicas procedentes del sistema motor, y una variable necesaria para enviar el conjunto de datos como String. Respecto a las librerías necesarias para el desarrollo de esta aplicación, cabe mencionar la librería "*SoftwareSerial.h*" para la creación de un puerto serie virtual para la comunicación entre el microcontrolador y el PC. Otra librería empleada es la SPI ("*SPI.h*"), empleada para la comunicación SPI con los sensores de vibración y temperatura instalados las cargas del sistema motor/generador. Para la comunicación entre el Arduino y la Nube, por medio del módulo GPRS/GSM, se emplean cuatro librerías diferentes. Una de ellas para resetear automáticamente el módulo SIM808, "*Adafruit_SleepDog.h*", cuando no es capaz de transmitir los datos por falta de conexión. Otra de las librerías empleadas para el control del módulo de comunicación GPRS/GSM, "*Adafruit_FONA.h*". Por otra parte, tenemos la librería para comunicarse con un cliente MQTT, "*Adafruit_MQTT.h*", esto es un protocolo de comunicación dentro del Internet de las Cosas enfocado al envío de datos a un servidor. Por último, se emplea una librería para acceder a la página web del servidor Adafruit, "*Adafruit_MQTT_FONA.h*".

A continuación, se procede a la configuración de los pines de entradas y salidas. Primero se ha configurado los pines analógicos físicamente conectados a la tensión en borne del motor, intensidad que circula por el motor y su velocidad de giro como pines de entrada. También se declaran los pines digitales de entrada y salida de datos para los sensores de temperatura y vibración, así como los pines GPIO utilizados para definir un UART virtual para comunicar inalámbricamente el microcontrolador con el PC. Asimismo, se declaran los pines digitales de entrada y salida de datos para el control del módulo FONA SIM808. La configuración de estos pines se realiza mediante la instrucción "*#define*". Esta instrucción es empleada para dar un nombre a un valor constante, en este caso, dar un nombre a los pines que utilizará cada componente. Al compilar el programa, las referencias a estas constantes son reemplazadas con el valor definido. Esta instrucción también es empleada para la configuración APN de la red móvil y para el acceso al servidor de datos Adafruit. Con la instrucción "*#define*" asignamos los datos que se deben de compilar para acceder a la red de la compañía móvil y conectarse a la red: el nombre de la APN de la red móvil, el usuario de la compañía, su contraseña y el pin de la SIM móvil empleada y el número asociado. Por otra parte, se definen los datos que en la inicialización de la conexión a la Nube requerirá el programa como: nombre del servidor al que debe de acceder, el puerto del servidor, el nombre de usuario y clave (Figura 53) que se tiene en el servidor y los nombre de los feeds a enviar. En la Tabla 10 se muestran los pines configurados en la placa Arduino Uno para el control de los componentes y la lectura de los sensores y señales.

Comunicación SPI	ADXL345	9
	ADT7310	8
	Interrupción	2
Entradas analógicas	Intensidad	1
	Tensión en bornes	2
	Velocidad de giro	3
Módulo Bluetooth	Permitir programación (KEY)	4
	Reset	5
	RX	6
	TX	7
Módulo FONA SIM808	TX	3
	Reset	4
	RX	5
	Interrupción SMS	6

Tabla 10. Pines configurados en el Arduino Uno.

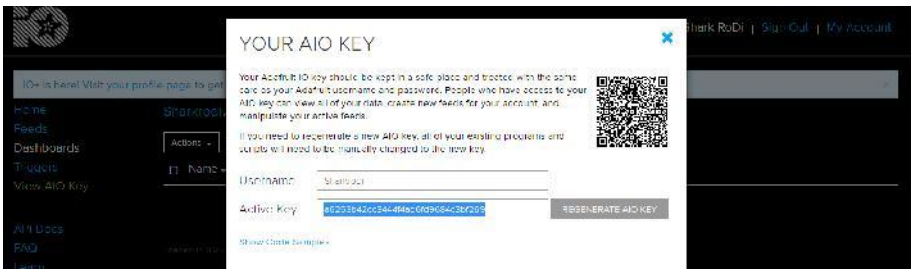


Figura 53. Nombre de usuario y clave tras el registro.

Subsecuentemente, se declara la interrupción asociada a la función "*dato_listo*" que indicará en la ejecución del programa cuando hay un nuevo dato de aceleración disponible. Asimismo, se debe configurar los parámetros del bus SPI (velocidad, modo de SPI, etc), la comunicación física por el puerto serie (comunicación USB con e PC), y la comunicación inalámbrica con el PC y la nube mediante el módulo Bluetooth HC05 y el módulo de comunicación GPRS/GSM SIM808, respectivametne. Subsecuentemente se configuran los sensores de vibración (ADXL345) y

temperatura (ADT7310) especificando la frecuencia de muestreo, rango de medida, la resolución, pin de interrupción a utilizar en caso de que hay un dato disponible.

En la Tabla 11 se muestra la configuración del acelerómetro ADXL345 y del sensor de temperatura ADT7310 utilizada en este TFM. En la Tabla 12 se muestra la configuración del bus SPI y del puerto UART entre el Arduino y el HC05.

ADXL345	Modo SPI	4-wire
	Rango de medida	±8g
	Frecuencia de muestreo	400 Hz
	Resolución	10 bits
	Power_saving register	Modo medida
	Pin de interrupción	2
ADT7310	Resolución	16-bits
	Frecuencia de muestreo	1 Hz (1 SPS)

Tabla 11. Configuración del acelerómetro y del sensor de temperatura.

Bus SPI	Bit Order	MSBFIRST
	Datamode	SPI_mode3 (reloj normalmente HIGH, muestreo en flanco de bajada)
	Frecuencia del bus	400 Hz
Puerto UART entre μC-HC05	Velocidad	115200 bits/s
	Bit de paridad	none
	Bit de stop	1

Tabla 12. Configuración del bus SPI y del puerto UART.

La configuración del módulo Bluetooth HC-05 requiere que el módulo entre en modo de comandos AT. Una vez puesto en modo AT, se reinicia el módulo para que se actualice el modo de comandos y se comienza con la configuración que se muestra en la Tabla 13. Establecidos los valores de configuración del módulo, se fuerza nuevamente a nivel bajo el modo AT y se habilita el modo

serie. Por ello, es preciso volver a resetear el módulo para que identifique el nuevo como de comandos.

Descripción	Instrucción	Comando y valores configurados
Baudrate prefijada para este modo	BT1.begin()	38400 bps
Valores iniciales de fábrica	BT1.println()	AT+ORGL
Baudrate para el modo puerto serie	BT1.println()	AT+UART=115200, 0, 0
Nombre del dispositivo	BT1.println()	AT+NAME=SRD
Clase de dispositivo, clase de servicio “Captura”, clase de dispositivo principal “Periférico” y clase de dispositivo menor “no es un teclado + dispositivo de detección”	BT1.println()	AT+CLASS=80510

Tabla 13. Configuración del módulo Bluetooth.

Por otro lado, se configura el módulo de comunicación GPRS/GSM FONA SIM808. Para ello son empleadas funciones de la librería de control del módulo a partir de las cuales se configuran los valores para su funcionamiento. En la Tabla 14 se describe la configuración del módulo junto con las instrucciones de la librería empleadas y los valores configurados.

Descripción	Instrucción	Comando y valores configurados
Velocidad de transmisión	fonaSS.begin()	4800 bps
Desbloqueo de la SIM	flushSerial(); Serial.println(PIN)	(Constante definida inicialmente)
Interrupción por SMS	pinMode()	FONA_RI_INTERRUPT (constante definida inicialemnte), INPUT
	digitalWrite()	FONA_RI_INTERRUPT, HIGH
	fona.setSMSInterrupt()	1
Conexión GPRS	fona.setGPRSNetworkSettings()	APN, nombre, contraseña (definidos)

Tabla 14. Configuración del módulo FONA SIM808.

Entre cada instrucción es común colocar una función adicional que provoca el reseteo automático del FONA SIM808 en caso de que el tiempo que tarda en intentar configurarse supere un tiempo establecido por esta función de reseteo.

Tras la configuración de los distintos componentes del sistema, se programa la función loop(), que es la función que contiene el programa que se ejecuta cíclicamente. En esta función, cada vez que detecta una interrupción, habilita la lectura del acelerómetro. Este es leído y se almacenan los valores en las variables previamente declaradas. Con la misma frecuencia que el acelerómetro se leen las señales analógicas que provienen de la placa de control de potencia del motor, es decir, las señales de intensidad, tensión y rpm del motor.

El sensor de temperatura no es capaz de adquirir datos con tanta frecuencia como el acelerómetro, su frecuencia de muestreo es de 1 Hz, frente a los 400Hz que se configuran en el acelerómetro. Por ello, cada 400 lecturas del acelerómetro, se comprueba si hay un nuevo dato de temperatura, y de ser así, se guarda en la variable declarada.

Asimismo, conforme se van obteniendo los valores de todas las variables analizadas en el sistema estos van siendo acumulados para calcular la aceleración total eficaz, así como los valores eficaces de temperatura, intensidad que circula por el motor, tensión en borne del motor y velocidad de giro, en un determinado tiempo (el tiempo programado es de 1 segundo, pero este se puede modificar).

Tras la lectura de las variables se crea una cadena de caracteres con los datos leídos, ADXL345 + señales analógicas + ADT7310 (cada 400 muestreos del acelerómetro), que es enviada por Bluetooth al PC y por puerto serie a modo de verificación del funcionamiento. La creación del *string* se realiza mediante la concatenación de los datos adquiridos separados por un una coma y un espacio (“, “). El orden de concatenación es: primero los datos adquiridos de los tres ejes de acelerómetro, luego los datos analógicos y luego la temperatura:

“x, y, z, V_I, V_MOTOR, V_MRP, T\r\n”

Los datos del acelerómetro y del sensor de temperatura son enviados en complemento a dos en formato decimal, es decir, la conversión a su correspondiente magnitud física se realiza en el programa de LabView. Del mismo modo, los datos de las señales analógicas son enviados tal y como son recogidos, es decir, con el valor de las tensiones adquiridas de cada señal en un formato de 0 a 1024, siendo posteriormente tratados y convertidos a su magnitud física correspondiente en LabView..

Del mismo modo, se publican los datos (valor eficaz) en el servidor web mediante una función específica a la que se le pasan los datos a publicar y el feed del servidor. No obstante, los datos enviados y publicados en el servidor se encuentran en la correspondiente magnitud física de la señal adquirida. El envío y publicación de estos datos se hace mediante una función que llama el programa principal, “*void publicar()*”. A esta función se le pasan dos parámetros, el valor que se quiere transmitir y el feed en el que se quiere registrar. Internamente esta función crea un buffer de tipo string que contendrá los datos que se quieren publicar en la web. Dado que los datos son numéricos, mediante una función se transforman los valores numéricos en datos del tipo caracteres y se almacenan en el buffer. Este buffer es publicado mediante la instrucción “*publishFeed()*” a la que

se le introduce el buffer a enviar. SI la publicación es exitosa o errónea, nos informará mediante un mensaje en la pantalla del PC.

Por último, en la programación nos encontramos con tres subfunciones más, que son llamadas en el programa principal.

La función "*void halt()*" es la función que se llama cuando ocurre un error, parándose la ejecución hasta que se resuelva. Si no se resuelve, al estar el watchdog (biblioteca para el reseteo automático del microcontrolador en caso de errores) activado, se reseteará el microcontrolador a los 8 segundos.

La función "*void flushSerial()*" es la encargada de que el módulo FONA SIM808 lea los valores que se le introducen. Por ello, internamente posee la instrucción "*Serial.read()*".

La función "*void dato_listo()*" es la función a la que salta el microcontrolador cuando se active la interrupción conforme hay un dato nuevo en la comunicación SPI del acelerómetro. En ella se introducen unos contadores para realizar posteriormente el cálculo de los valores eficaces de las medidas tomadas.

En la Figura 54 se muestra el flujograma del programa que se implementa en el Arduino Uno.

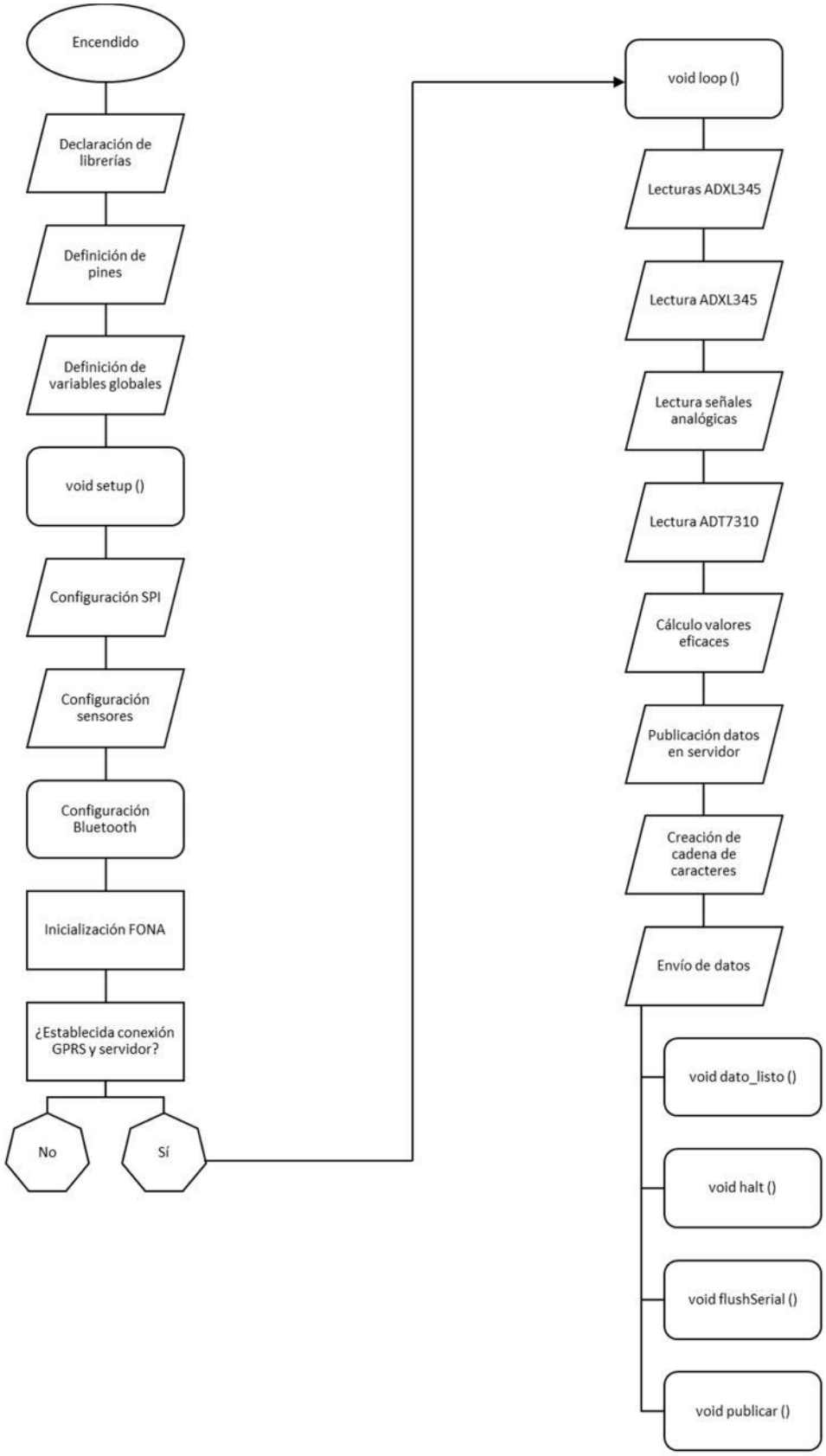


Figura 54. Flujograma del programa

5.2. PROGRAMACIÓN DE LabVIEW

Acrónimo de *Laboratory Virtual Instrument Engineering Workbench*, se trata de un programa para el desarrollo de sistemas mediante un lenguaje gráfico. Empleado en la ingeniería e investigación para el desarrollo de aplicaciones de medición, control y pruebas, puesto que su programación es muy intuitiva y lo hacen una herramienta ideal para estas aplicaciones.

Sus principales ventajas son su simplicidad, lo que lo convierte en un programa fácil de aprender, gran funcionalidad y la capacidad de entradas y salidas integradas.

El entorno de programación es muy simple, como se viene diciendo. Consiste en dos pantallas:

- Un panel frontal, en el que se muestra la ejecución del programa (mediciones, controles, avisos…).
- Diagrama de bloques, en el que se programan de forma gráfica las funciones que se quieren realizar.

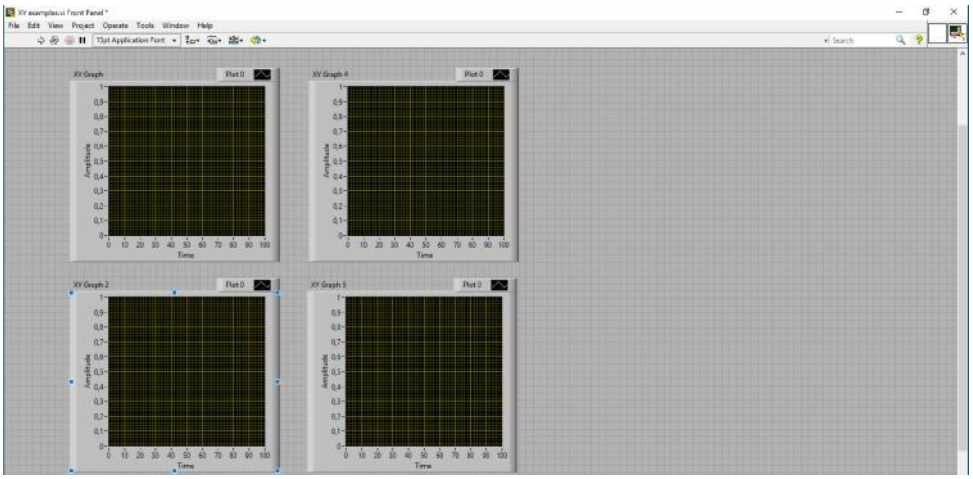


Figura 55. Panel frontal de LabVIEW.

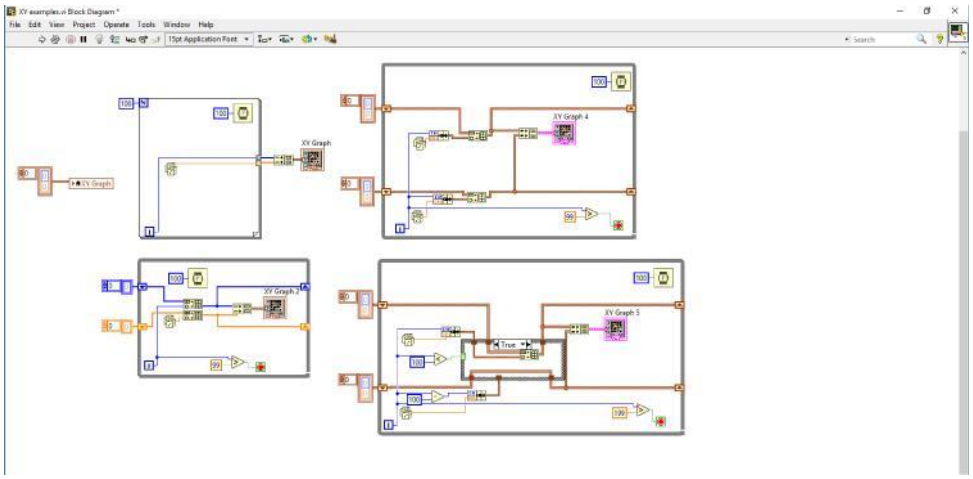


Figura 56. Diagrama de bloques de LabVIEW.

5.2.1. FUNCIÓN DEL PROGRAMA LabVIEW

LabVIEW es el responsable de leer los datos enviados por el microcontrolador en formato string, o cadena de caracteres, convertir cada señal en su correspondiente magnitud física, representar gráficamente, analizar y almacenar el los datos. Recordar que, los datos enviados por el microcontrolador corresponden a los datos crudos (sin convertir en magnitud física) separados en coma y un espacio, además los datos de cada medida están separados por '\r\n'. Además, la frecuencia de muestreo del acelerómetro y las señales analógicas (tensión en bornes del motor, corriente que circula por el motor y velocidad de giro del motor) es de 400 Hz, y la frecuencia de muestreo para la medida de temperatura es mucho inferior (1 Hz).

"x, y, z, V_I, V_MOTOR, V_MRP, T\r\n"

El programa realizado en LabVIEW se basa en el flujograma de la Figura 57.

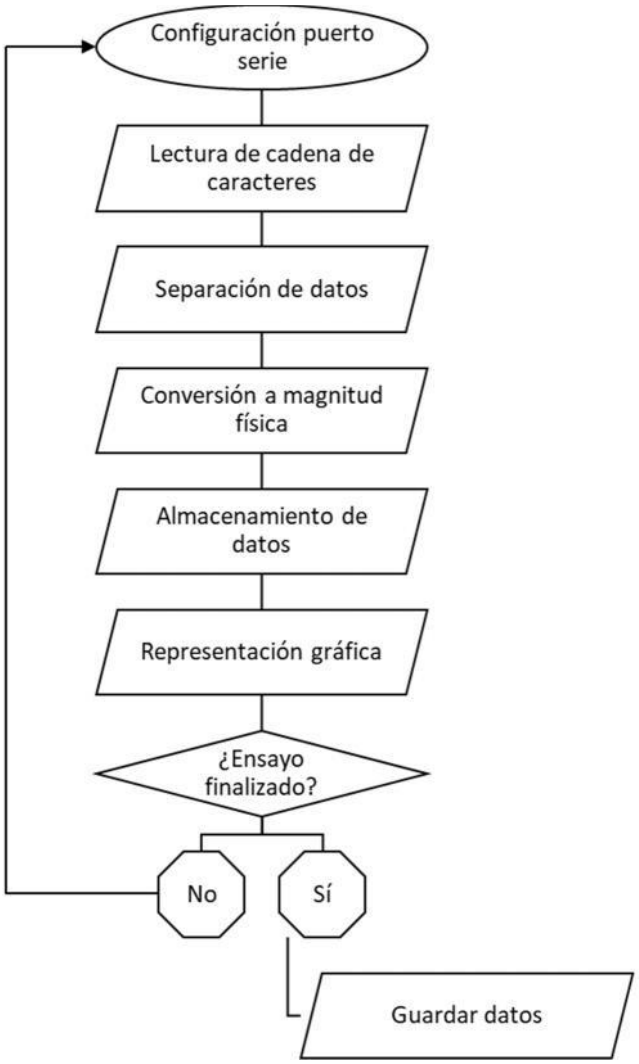


Figura 57. Grafcet programación LabVIEW.

Independientemente de la comunicación entre el Arduino y el PC se realiza mediante USB o Bluetooth HC05, se emplea el paquete VISA de LabVIEW el cual permite la lectura de cadenas de caracteres recibidos por un puerto serie. En caso de transmisión inalámbrica, a nivel hardware se requiere el uso de un adaptador de Bluetooth-USB en el lado de receptor (PC), y con ello la comunicación Bluetooth es totalmente transparente. La única diferencia con un puerto serie es que hay que emparejar el receptor con el emisor antes de la ejecución del programa para que así exista comunicación de datos. Los parámetros que se deben de tener en cuenta para la comunicación entre Arduino y LabVIEW es el puerto COM que emplea el Bluetooth para enviar los datos o el puerto COM correspondiente al USB.

LabVIEW lee una cadena de caracteres, proveniente del microcontrolador, que contiene todos los datos de una medición separados por "\r\n". Esta cadena de caracteres sigue siempre el mismo patrón, como se ha mencionado anteriormente, y cada paquete de datos contiene seis o siete datos, dependiendo si se ha medido o no la temperatura. El programa busca en esta cadena de caracteres el punto y el espacio y separa los datos, enviándolos cada uno por una línea de información distinta.

Una vez separada la información del paquete de datos, se procede a identificar la magnitud física correspondiente a cada dato contenidos en la medida. A continuación, se decodifican los datos en complemento a dos en formato decimal de las señales del acelerómetro, teniendo en cuenta su resolución y el rango de medida. Del mismo modo, las señales analógicas, que son enviadas en forma de tensión medida por el microcontrolador en un formato de 0 a 1024 niveles de resolución, se convierten a magnitudes físicas reales, mediante operaciones aritméticas.

En cambio, la frecuencia de muestreo de temperatura es de 1 Hz, el cual es mucho inferior a la de otros datos. Esto implica que en algún paquete de datos hay medida de temperatura y no en otros paquetes. Para ello, se procede a comprobar primero si hay medida de temperatura o no en el paquete de datos bajo análisis. En caso afirmativo, se procede a la conversión a su magnitud física decodificando el complemento a dos y teniendo la resolución del sensor de temperatura.

Para cada señal leída (aceleraciones, temperatura, corriente, tensión en borne y velocidad de giro del motor) se genera un vector de datos en el que se van acumulando los datos ya obtenidos y el nuevo dato, con el fin de estudiar la evolución temporal de todas las señales recibidas, representándose gráficamente en tiempo real cada una de las señales recibidas. Asimismo, se puede estudiar la relación de la vibración (valor eficaz), tensión en borne del motor, intensidad y potencia en función de la velocidad de giro.

Cuando se finaliza la prueba se guardan automáticamente los vectores de datos en tres ficheros Excel (uno para las aceleraciones, otro para las medidas analógicas y otro para las temepraturas), cuya localización es previamente definida en el panel de control, de igual manera que se puede definir una ubicación para guardar las gráficas obtenidas.

5.2.2. INTERFAZ GRÁFICA DEL USUARIO

De la Figura 58 a la Figura 60 se ilustra el panel frontal del programa LabView desarrollado.

Figura 58. Panel frontal, pestaña de "Control"

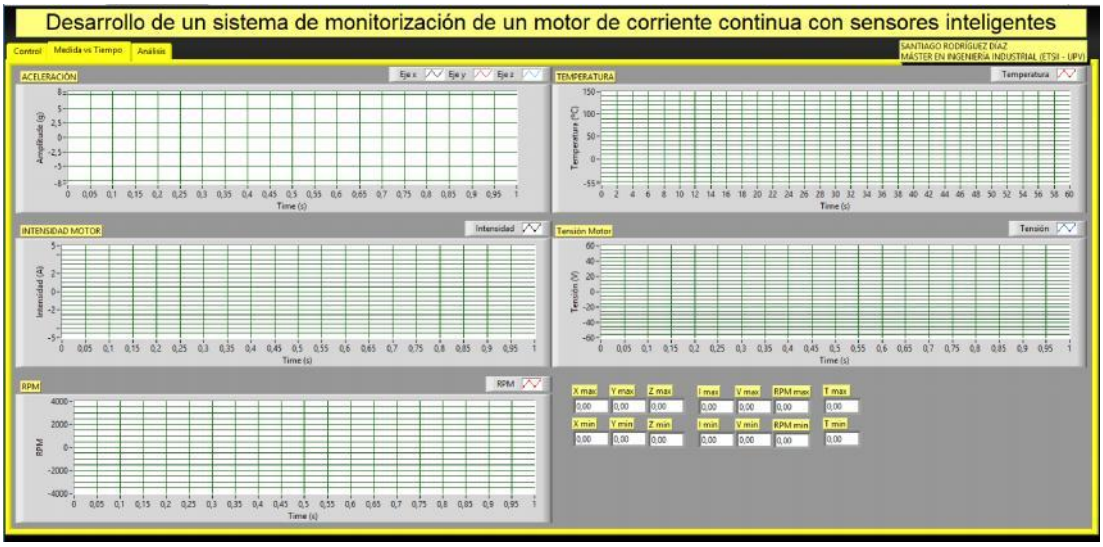


Figura 59. Panel frontal, pestaña de "Medidas vs tiempo"

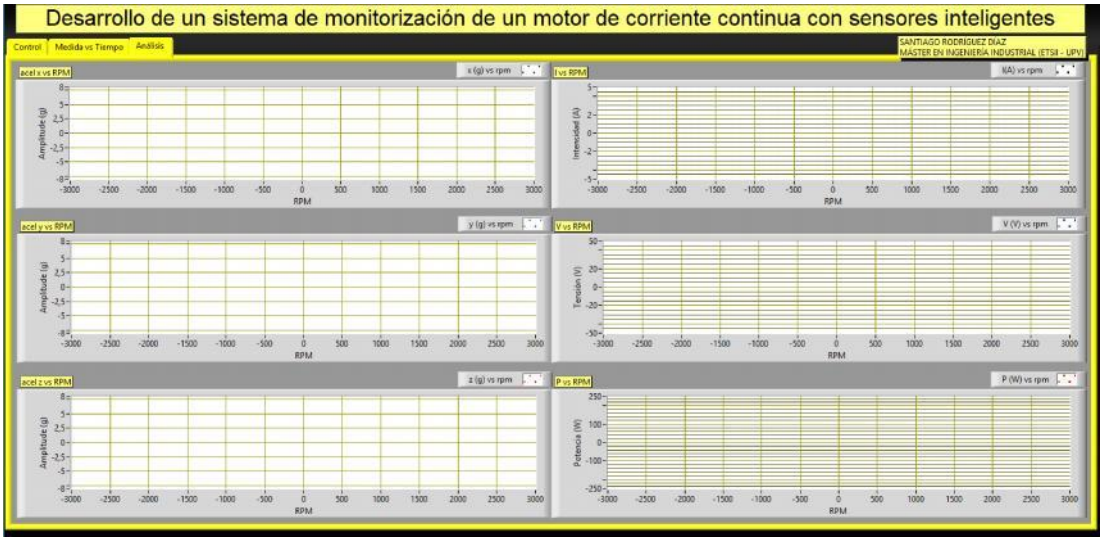


Figura 60. Panel frontal, pestaña de "Análisis"

Como se muestra en la Figura 58, se configura el panel frontal con tres pestañas (Figura 58, 1). La primera para la configuración inicial, con los elementos de control, donde se selecciona el puerto serie del que se recibirán los datos (Figura 58, 2), se seleccionan las direcciones de ficheros donde se guardarán los datos (Figura 58, 3 y 4), se mostrarán los errores si suceden (Figura 58, 5) y se detiene el programa (Figura 58, 6).

En la pestaña “Medida vs. Tiempo” (Figura 59) se encuentran las representaciones gráficas de las distintas señales adquiridas (tensión en borne del motor, corriente que circula por el motor, velocidad de giro, aceleración en cada eje y temperatura) en función del tiempo.

Por último, en la tercera pestaña se muestra el análisis de las vibraciones, la tensión en borne del motor y la corriente que circula por el motor (Figura 60) en función de la velocidad de giro del mismo. Respecto a la vibración, se representan la aceleración adquirida en cada eje en función de la velocidad de giro, para ello se calcula la media cuadrática de cada eje de aceleración tras eliminar el nivel de continua para un intervalo de 5 segundos en función de la velocidad de giro. De este modo se consigue obtener un estudio significativo de la aceleración en cada régimen de giro. De igual modo, internamente el programa calcula la potencia consumida, y muestra su evolución en función de la velocidad de giro.

CAPÍTULO 6. RESULTADOS EXPERIMENTALES

6.1. MONTAJE

En este apartado, se describe el montaje del sistema de monitorización multivariable del sistema motor para realizar los ensayos pertinentes. Tal como se ha mencionado anteriormente, los sensores de temperatura y vibración se colocan junto con la resistencia de carga, y envía la señal salida de SPI al Arduino mediante un cable plano. Se ha de tener especial precaución respecto a la longitud del cable plano el cual no debe exceder los 30 cm, en caso contrario podría aumentar la tasa de datos erróneos de la transmisión de la señal SPI, es decir, podría dar lugar a error en la interpretación de los datos.

De la Figura 61 a la Figura 64 se muestran las imágenes del montaje físico realizado para los ensayos.

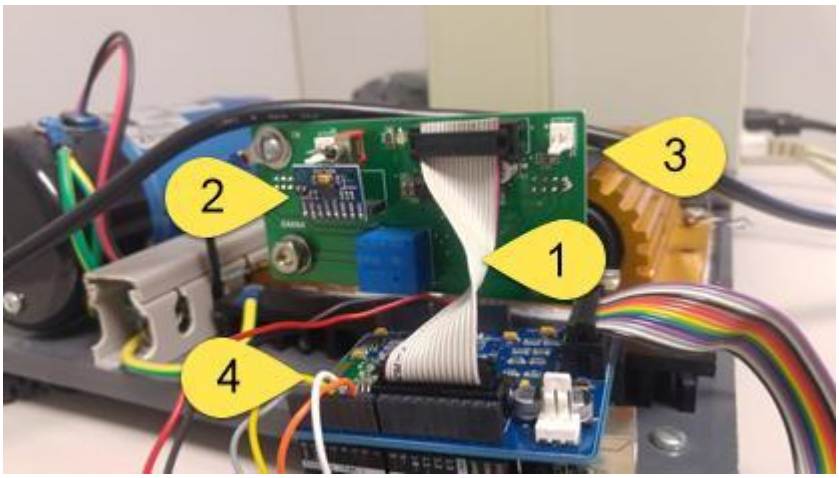


Figura 61. Conexión mediante cable plano (1) de los sensores de aceleración (2) y temperatura (3) a Shield de diseño propio conectada a conectada a Arduino Uno (4).

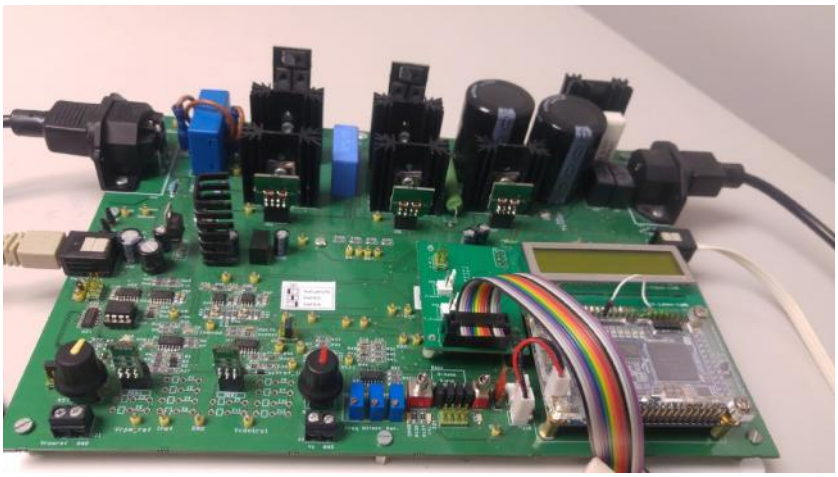


Figura 62. Control electrónico del sistema motor del cual proceden las tres señales analógicas (tensión en borne del motor, corriente que circula por el motor y velocidad de giro del motor).

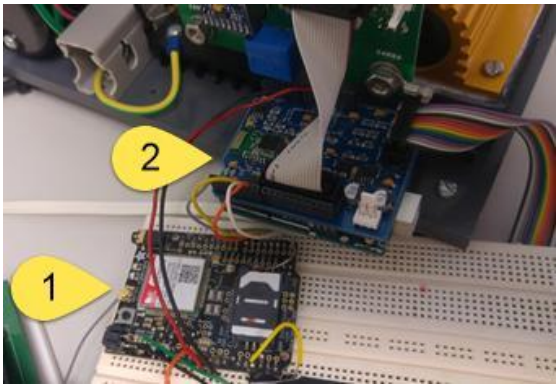


Figura 63. Conexión de Arduino (1) con FONA SIM808 (2).

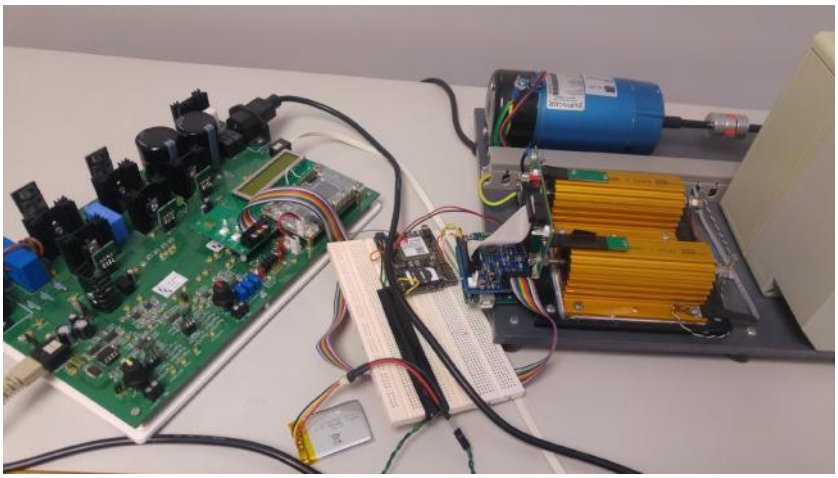


Figura 64. Visión general del montaje.

6.2. RESULTADOS

6.2.1. ADQUISICIÓN DE DATOS EN EL PC

Se realizan 9 ensayos bajo unas determinadas condiciones de funcionamiento: motor sin carga y con carga. Cada ensayo se realiza durante un tiempo de 10 minutos, dejando enfriar el motor entre cada ensayo hasta recuperar su temperatura ambiente, para así poder ver claramente la tendencia de la temperatura en el transcurso del funcionamiento. También se ha testeado el envío de datos y publicación de los mismos en el servidor web.

En la Figura 65 se muestran la representación gráfica de los datos adquiridos en tiempo real cuando el motor gire a -3000 rpm y con las dos cargas conectadas. En ella se puede observar unas componentes de altas frecuencias de vibración en todos los ejes, en las que las mayores aceleraciones se producen en el eje “x” y en el eje “y”, no superando en ninguno de los casos los $\pm 0.75g$. Las señales de intensidad, tensión en borne del motor y velocidad de giro, alcanzan todas sus valores absolutos máximos (negativo) sin sufrir variaciones a lo largo del ensayo, es decir, prácticamente es una señal continua. En cambio, la temperatura de la carga muestra una tendencia creciente y es prácticamente lineal durante los 10 minutos de ensayo.

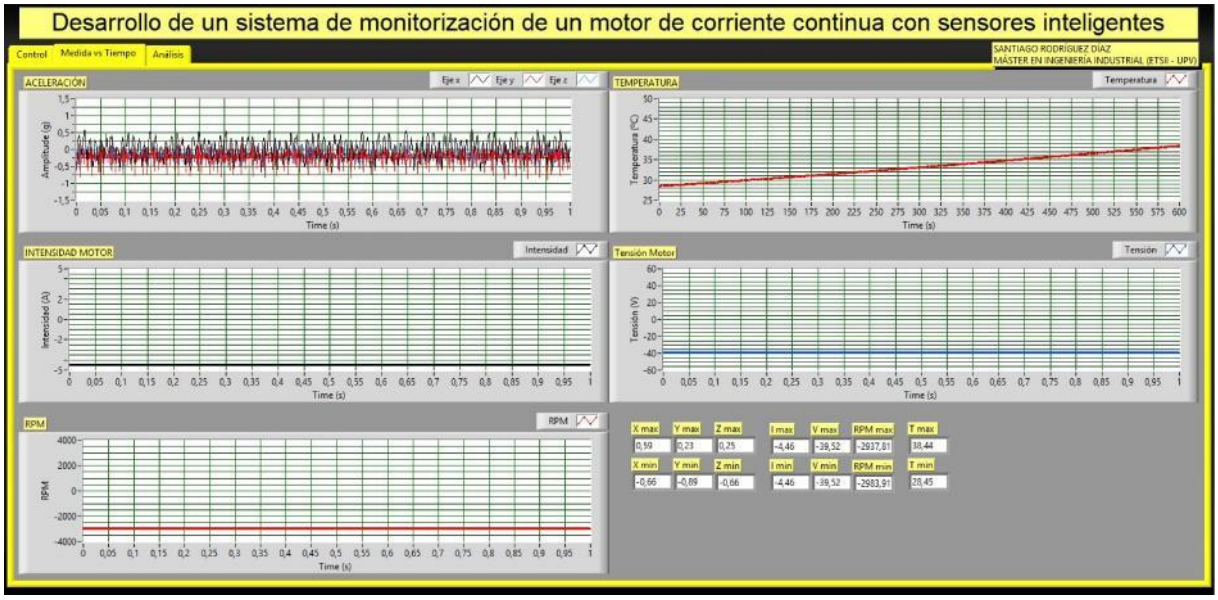


Figura 65. Señales adquiridas con el motor a -3000 rpm y las dos resistencias conectadas..

En la Figura 66 se muestra los datos adquiridos en tiempo real cuando el motor gire a 3000 rpm y con las dos cargas conectadas. En ellas se puede apreciar que las vibraciones son ligeramente menores que en el ensayo con el motor a -3000 rpm., sobre todo en el eje "y", y que las vibraciones en el eje "x" tienden a ser positivas. Sin embargo, los valores absolutos de las señales analógicas son superiores, debido a que el ajuste y cálculo de los circuitos acondicionadores de la señal se realizó en un puesto diferente al de los ensayos finales, obteniéndose así pequeñas variaciones. En cuanto a la temperatura, se observa una tendencia similar al caso anterior (Figura 65) aunque en este caso se alcanza una temperatura un poco inferior, pero sin diferir en exceso el comportamiento.

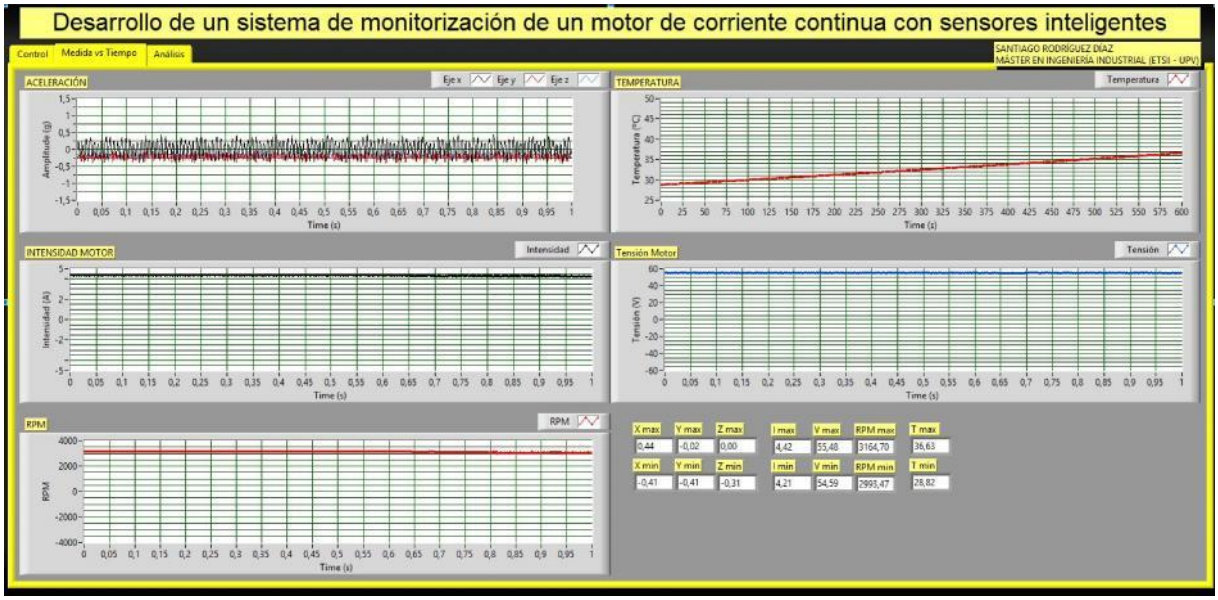


Figura 66. Señales adquiridas con el motor a 3000 rpm y las dos resistencias conectadas.

El siguiente ensayo consiste en tener las dos cargas activas, y llevar al motor a aproximadamente cero revoluciones (Figura 67). La amplitud de la señal de vibración es prácticamente nula, es decir, se observa una señal continua correspondiente a la aceleración estática debido a la gravedad. Del mismo modo, tanto la tensión en borne del motor, la intensidad que circula por el motor como la velocidad de giro del motor son prácticamente nulas. La única magnitud física no-nula (28ºC) es la temperatura al estar expuesto el sensor de temperatura a la temperatura ambiente.

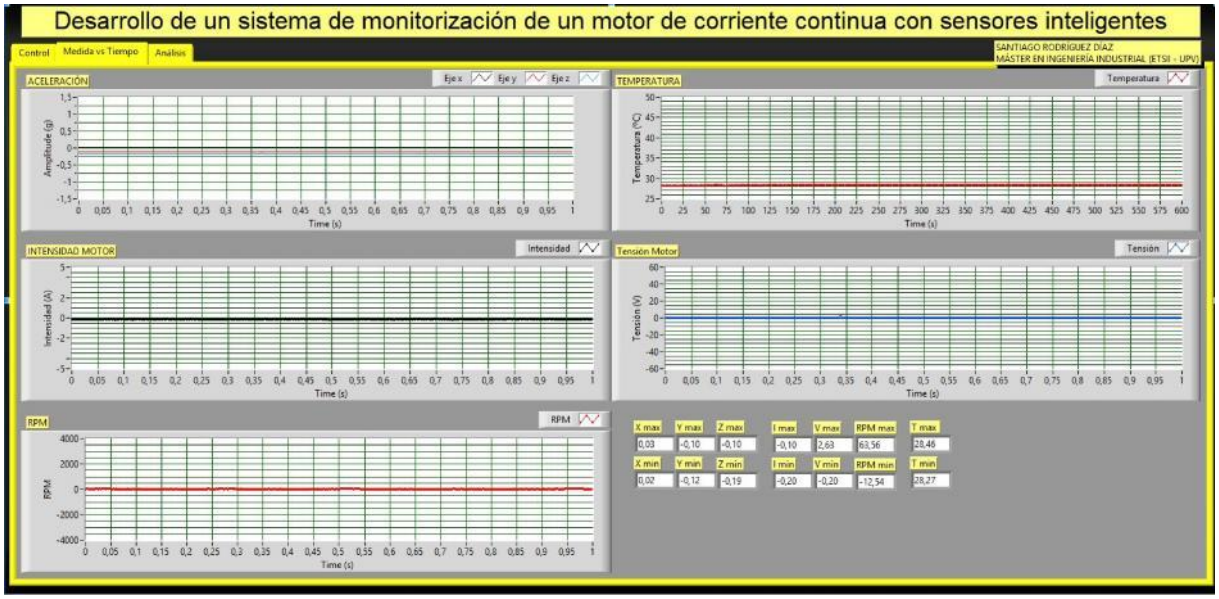


Figura 67. Señales adquiridas con el motor a 0 rpm y las dos resistencias conectadas.

La Figura 68 muestra las señales adquiridas después de descentrar el acoplo entre el motor y el generador, para provocar irregularidades en el giro, y llevar el motor a un régimen de giro de 3000 rpm a máxima carga. Se puede observar las vibraciones obtenidas son superiores al mismo caso sin desalineamiento (3000 rpm, dos cargas Figura 66), donde destaca la mayor amplitud de la aceleración en el eje “x”. Para estas condiciones de máxima carga y máxima velocidad de giro positiva se observa como la intensidad y tensión en borne del motor son constantes durante todo el ensayo, como sucedía en el caso de no estar el eje desalineado, y los valores son igualmente máximos. En cambio, la temperatura experimenta una variación más rápida que en el caso sin desalineamiento (Figura 66), siendo la temperatura final alcanzada superior.

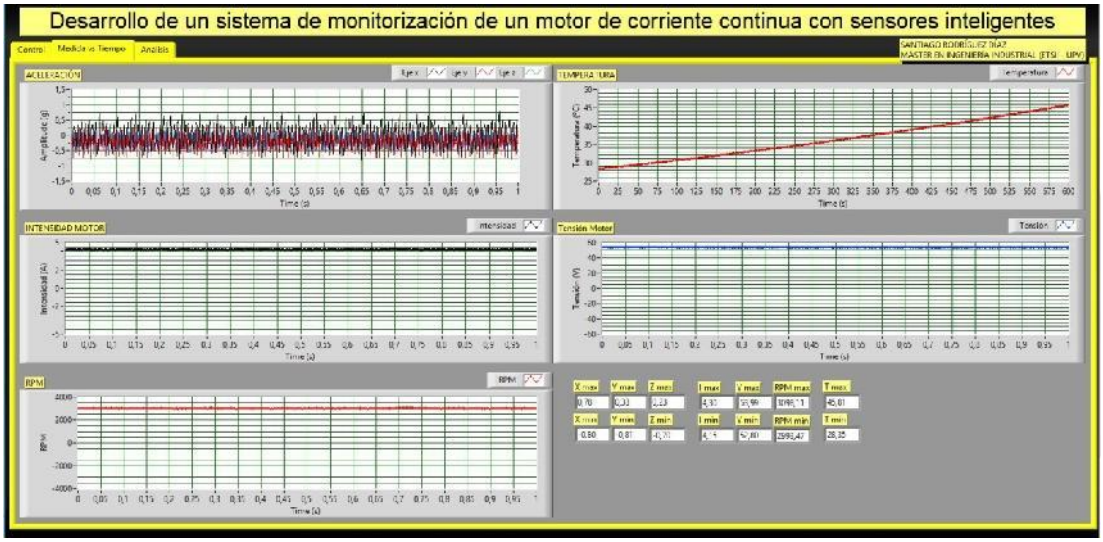


Figura 68. Señales adquiridas después de descentrar el acoplo entre el motor y el generador, para provocar irregularidades en el giro, y llevar el motor a un régimen de giro de 3000 rpm a máxima carga.

A continuación, se muestra el estudio de las distintas variables en función de la velocidad de giro del motor, sin cargas (Figura 69). Para ello se varía lentamente la velocidad de giro del motor desde -3000 rpm hasta las 3000 rpm. Recordar que se ha computado la vibración como la media cuadrática de la aceleración tras eliminar el nivel de continua.

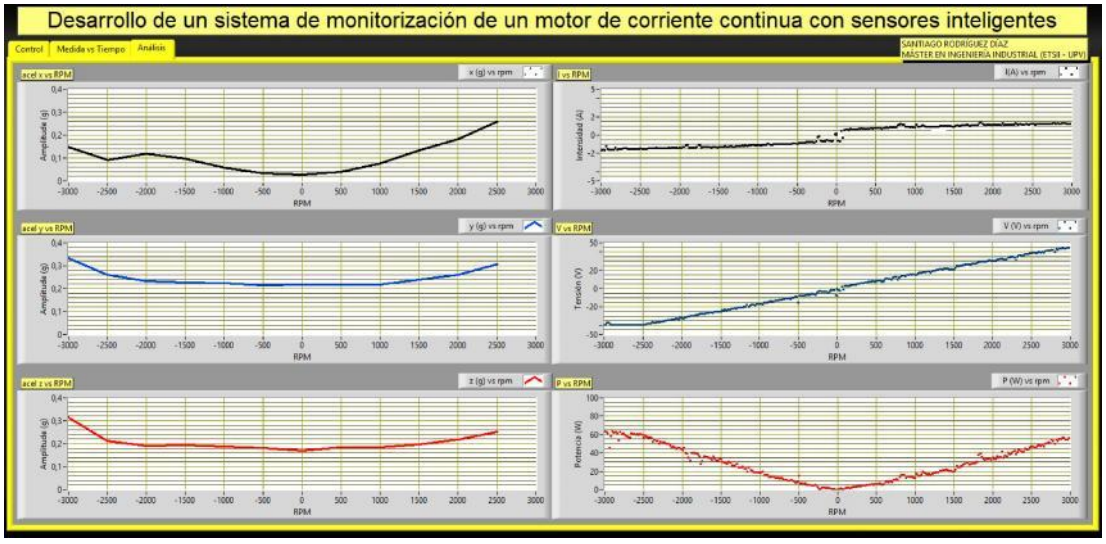


Figura 69. Estudio de las distintas magnitudes físicas en función de la velocidad de giro del motor sin carga.

En primer lugar, el comportamiento de la aceleración en el eje “x” ante los diferentes regímenes de giro es un comportamiento parabólico. EN cambio, la vibración en el eje “y” y “z” son prácticamente iguales para revoluciones bajas del motor, sólo aumenta la vibración en estos ejes a partir de 1500 rpm.

La intensidad para el funcionamiento sin carga varía entre los $-2\,A$ y los $2\,A$, aproximadamente, obteniendo los máximos absolutos en los extremos y teniendo un comportamiento casi lineal. Asimismo, la tensión en borne del motor tiene un comportamiento lineal, con una pendiente superior a la de la curva de intensidad, y teniendo su máximo absoluto para el funcionamiento sin carga en $40\,V$, aproximadamente. La potencia consumida por el motor presenta un comportamiento parabólico con máximos en los extremos de en torno 60 W.

En la Figura 70 se muestra el estudio de las distintas variables en función de la velocidad de giro del motor a máxima carga.

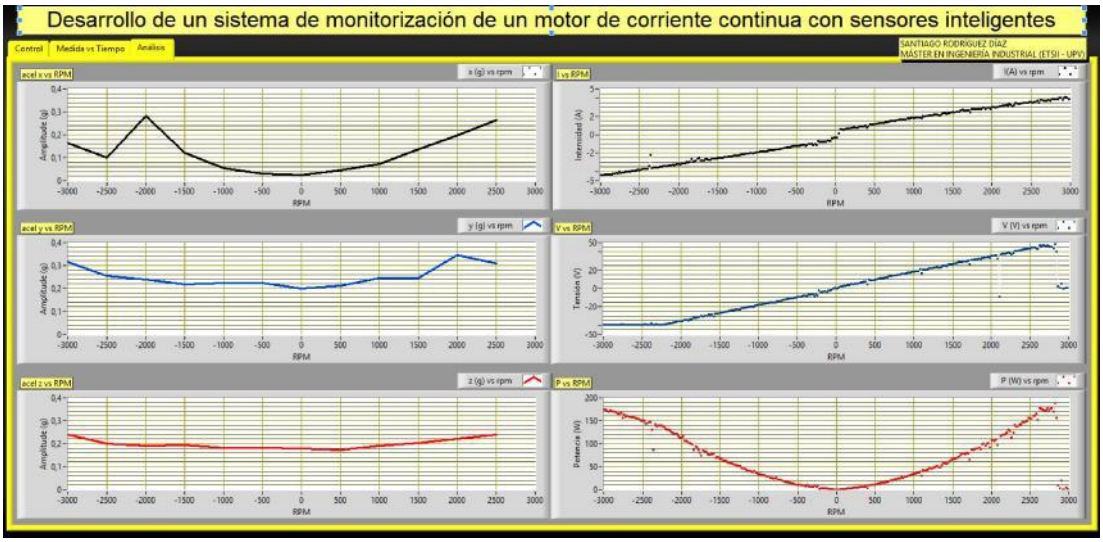


Figura 70. Estudio de las distintas magnitudes físicas en función de la velocidad de giro del motor a máxima carga.

Bajos estas condiciones, las aceleraciones en los tres ejes son del mismo orden de magnitud y presenta respuestas muy similares que cuando el motor trabaja sin cargas. Es decir, la carga a la que está sometido el motor no afecta significativamente a las vibraciones que se producen con los distintos regímenes de giro. Del mismo modo, la tensión en borne del motor presenta una respuesta similar que cuando el motor trabaja sin cargas, variando entre los mismos máximos, $\pm 40\,V$, y teniendo la misma pendiente. En cambio, la intensidad que atraviesa por el motor presenta la misma respuesta lineal, pero con un mayor rango de variación ($\pm 4\,A$). Como consecuencia de la intensidad, la potencia se ve notoriamente afectada por los cambios de carga en el motor, llegando ahora el motor a un consumo de $175\,W$, aproximadamente.

6.2.2. FUNCIONAMIENTO GPRS/GSM.

El último ensayo realizado es la comprobación del sistema de comunicación GPRS/GSM y el envío y publicación de los datos enviados en el servidor configurado. Cada segundo, el programa calcula los valores eficaces de los datos recogidos en ese tiempo y los envía y publica en el servidor configurado.

Este ensayo se realiza con el sistema sometido a las dos cargas y a un régimen de giro de $-3000\,rt$. De esta forma se obtienen resultados similares a los de la Figura 65. En la Figura 71 se

muestra la página del servidor a donde se envían y publican los datos, con las variables que son enviadas.

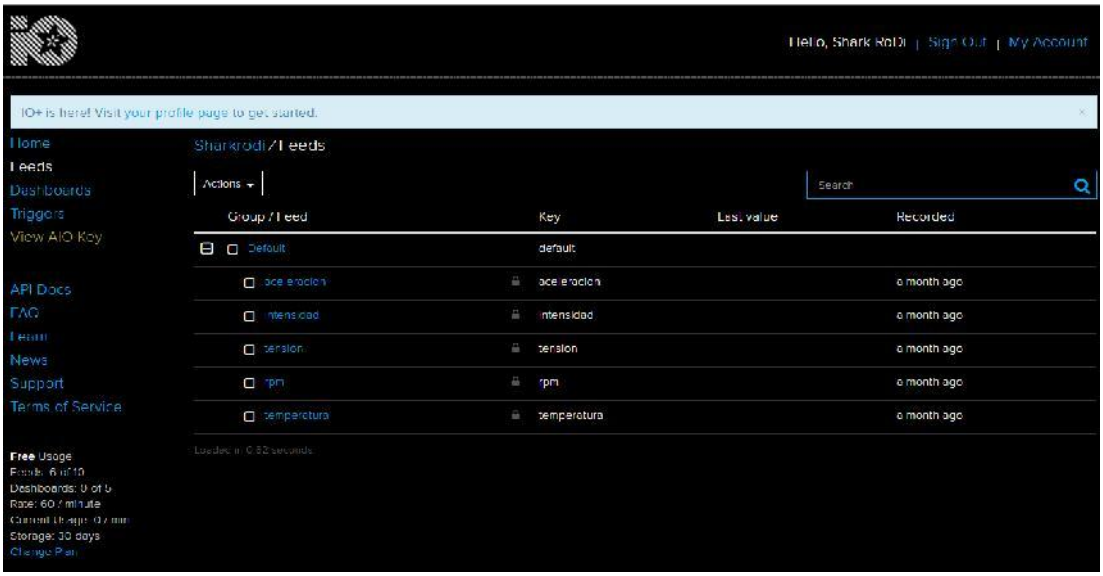


Figura 71. Datos publicados en el servidor.

El servidor permite la representación gráfica de los datos enviados, por lo que entrando en el feed de cada variables se puede ver su representación gráfica del valor eficaz de la aceleración total (Figura 72), el valor eficaz de la intensidad que circula por el motor (Figura 73), y el de la tensión en bornes del motor(Figura 74). Además, se disponen los datos recopilados para poder descargarlo durante el funcionamiento del sistema motor/generador. Asimismo, el sistema permite enviar un mensaje de alerta ante situaciones inesperada. En la figura 75 se muestra el mensaje SMS enviado al número previamente configurado en el programa cuando el sistema alcanza una temperatura de 39 grados (temperatura fijada en el programa).

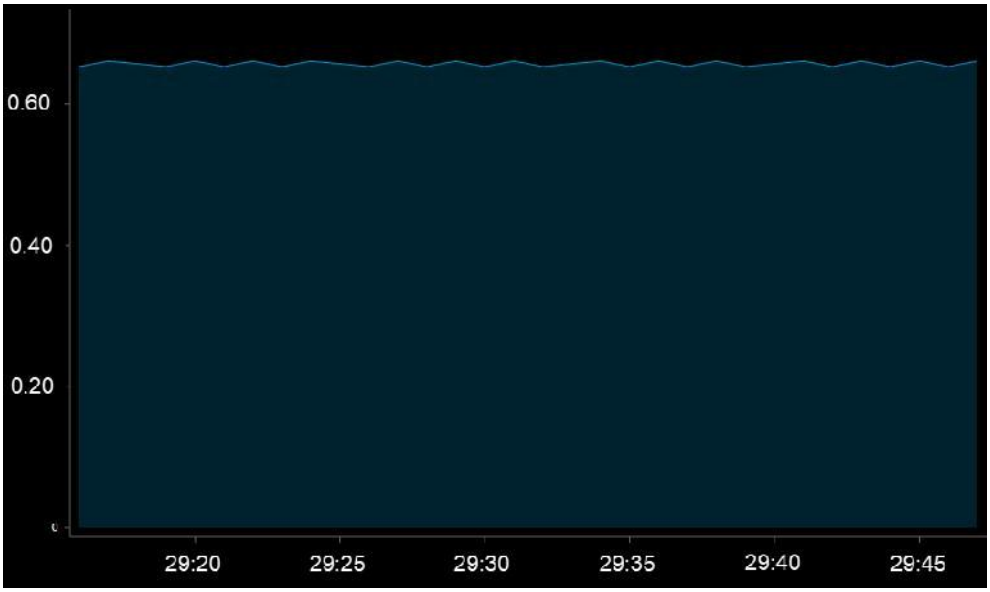


Figura 72. Gráfica de valor eficaz de los tres ejes de aceleración motor publicado en el servidor web cuando el motor gire a -3000 rpm.

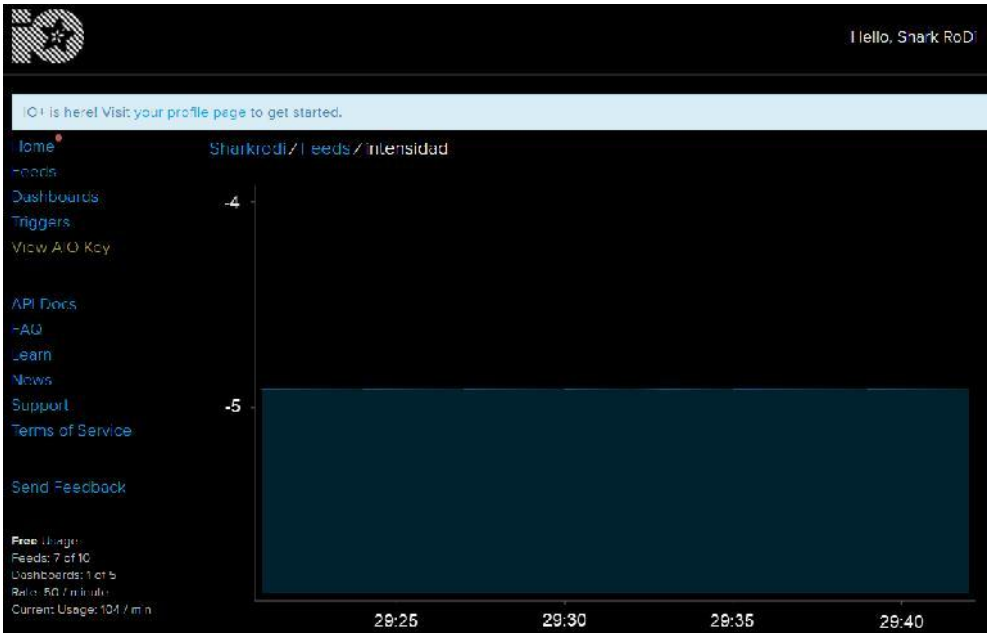


Figura 73. Gráfica de la intensidad que circula por el motor publicado en el servidor web cuando el motor gire a -3000 rpm.

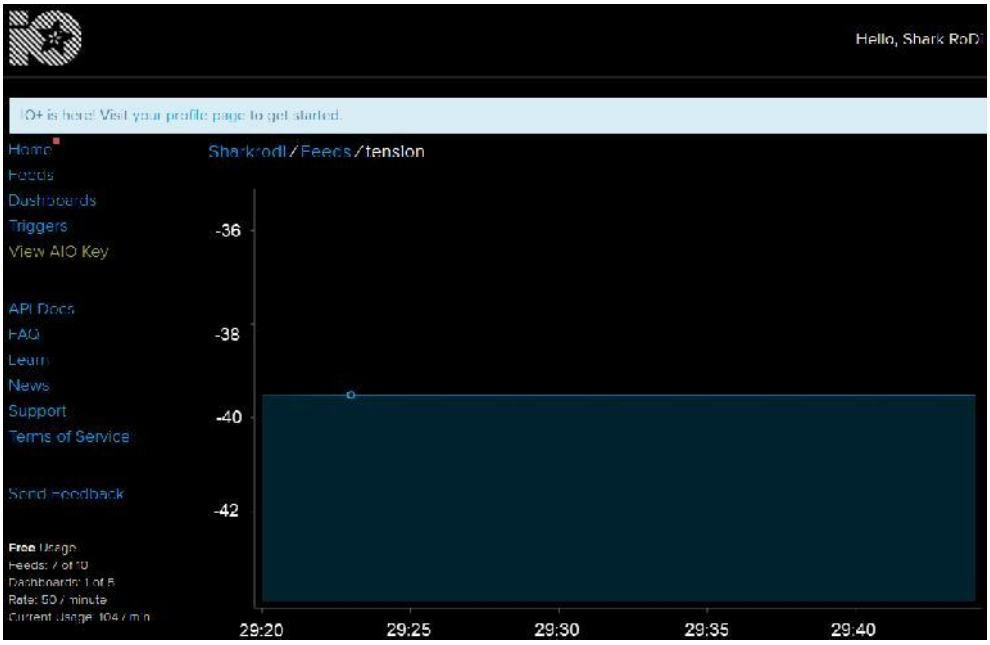


Figura 74. Gráfica de la tensión en bornes del motor publicada en el servidor web cuando el motor gire a -3000 rpm.

Figura 75. Mensaje SMS de alerta ante un exceso de temperatura.

CAPÍTULO 7. CONCLUSIÓN

Se ha desarrollado un sistema de monitorización continua de las múltiples magnitudes físicas del sistema motor/generador basado en sensores inteligentes. El sistema de monitorización se basa en un microcontrolador que es el responsable de leer la información de los sensores de temperatura y vibración por el puerto SPI, y 3 señales analógicas procedentes del sistema motor, y los envía a un PC local vía cableado o mediante transmisión inalámbrica. Asimismo, el sistema permite enviar los datos a una plataforma de Nube mediante GPRS y mensajes de alerta vía GSM ante cualquier situación indeseada. El sistema desarrollado podría utilizarse para introducir los sensores inteligentes a los alumnos de nuevo ingreso del Máster Universitario de Ingeniería Industrial y también como un prototipo de un sistema de monitorización continua para el mantenimiento predictivo.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE MÁSTER DE INGENIERÍA INDUSTRIAL

DESARROLLO DE UN SISTEMA DE MONITORIZACIÓN DE UN MOTOR DE CORRIENTE CONTINUA CON SENSORES INTELIGENTES

PRESUPUESTO

1. CONTENIDO DEL PRESUPUESTO

Título:	Desarrollo de un sistema de monitorización de un motor de corriente continua con sensores inteligentes
Situación:	Universitat Politècnica de València

CAPÍTULO I. SENSORES

N/P	*CONCEPTO*	*Uds*	*P. Unit.*	*Importe*
1.1	Módulo GY-291 acelerómetro 3 ejes ADXL345 para Arduino	1	1.5700	1.57 €
1.2	ADT7310TRZ - IC de Detección de Temperatura, Digital	1	2.5371	2.54 €
	TOTAL CAPÍTULO I			**4.11 €**

CAPÍTULO II. SOPORTE HARDWARE

N/P	*CONCEPTO*	*Uds*	*P. Unit.*	*Importe*
2.1	Arduino Uno Rev 3, MCU ATmega328, 6 Entradas Analógicas, 6 Salidas PWM	1	19.5900	19.59 €
	TOTAL CAPÍTULO II			**19.59 €**

CAPÍTULO III. SUBSISTEMA DE COMUNICACIÓN

N/P	*CONCEPTO*	*Uds*	*P. Unit.*	*Importe*
3.1	Bluetooth HC-05 sin pines	1.00	4.9180	4.92 €
3.2	Adafruit FONA 808 Shield (GSM + GPS) para Arduino	1	42.9000	42.90 €
	TOTAL CAPÍTULO III			47.82 €

CAPÍTULO IV. ADAPTADOR DE NIVEL DE SEÑALES ANALÓGICAS AL MCU

N/P	*CONCEPTO*	*Uds*	*P. Unit.*	*Importe*
4.1	PCB 2 caras menor de 5 x 5 cm	1	2.1490	2.15 €
4.2	Condensador de Cerámica Multicapa, 1206 [Métrica 3216], 0.1 µF	2	0.0704	0.14 €
4.3	Condensador de Cerámica Multicapa, 1206 [Métrica 3216], 0.39 µF	5	0.1580	0.79 €
4.4	Condensador de Cerámica Multicapa, 1206 [Métrica 3216], 0.56 µF	2	0.1460	0.29 €
4.5	Condensador Electrolítico Aluminio SMD, Radial - SMD, 100 µF	4	0.2490	1.00 €

4.6	Condensador de Tantalio SMD, 1206 [Métrica 3216],100 µF	4	0.6130	2.45 €
4.7	Condensador de Tantalio SMD, 1206 [Métrica 3216], 0.33 µF	2	0.2880	0.58 €
4.8	Condensador de Tantalio SMD, 1206 [Métrica 3216], 0.1 µF	20	0.2570	5.14 €
4.9	LED, Rojo, Montaje Superficial, 1.6mm x 0.8mm, 2 mA	4	0.2870	1.15 €
4.10	Diodo Schottky de Pequeña Señal, 30 V, 200 mA, 800 mV, 5 A	2	0.6330	1.27 €
4.11	Amplificador Operacional, Doble, 2 Amplificadores, SOIC	3	0.6230	1.87 €
4.12	Regulador Lineal de Tensión Fija Negativa, 79L05, Vreg -5V, SOIC-8	1	0.9110	0.91 €
4.13	MAX3000EEUP+ Traductor de Nivel de Tensión, 8 Entradas, 1 µs, 1.65 V a 5.5 V, TSSOP-20	1	3.5000	3.50 €
4.14	Resistencia SMD, Cerámica, 100 kohm, 1206 [Métrica 3216]	1	0.0198	0.02 €
4.15	Resistencia SMD, Cerámica, 3.9 kohm, 1206 [Métrica 3216]	7	0.0333	0.23 €
4.16	Resistencia SMD, Cerámica, 15 kohm, 1206 [Métrica 3216]	1	0.0454	0.05 €
4.17	Resistencia SMD, Cerámica, 8.2 kohm, 1206 [Métrica 3216]	1	0.0454	0.05 €
4.18	Resistencia SMD, Cerámica, 5.6 kohm, 1206 [Métrica 3216]	1	0.0333	0.03 €
4.19	Resistencia SMD, Cerámica, 120 kohm, 1206 [Métrica 3216]	1	0.0454	0.05 €
4.20	Resistencia SMD, Cerámica, 18 kohm, 1206 [Métrica 3216]	1	0.0454	0.05 €
4.21	Resistencia SMD, Cerámica, 22 kohm, 1206 [Métrica 3216]	1	0.0333	0.03 €
4.22	Resistencia SMD, Cerámica, 1.8 kohm, 1206 [Métrica 3216]	1	0.0272	0.03 €
4.23	Resistencia SMD, Cerámica, 1 kohm, 1206 [Métrica 3216]	1	0.0275	0.03 €
4.24	Resistencia SMD, Cerámica, 470 ohm, 1206 [Métrica 3216]	3	0.0479	0.14 €
4.25	Resistencia SMD, Cerámica, 1.5 kohm, 1206 [Métrica 3216]	1	0.0333	0.03 €
4.26	Resistencia SMD, Cerámica, 1.2 kohm, 1206 [Métrica 3216]	3	0.0464	0.14 €
4.27	Conector de Cable a Placa, Vertical, 2.54 mm, 16 Contactos, Header, Agujero Pasante	1	0.4900	0.49 €
4.28	Conector de Cable a Placa, Vertical, 2.54 mm, 14 Contactos, Header, Agujero Pasante	1	0.6000	0.60 €

4.29	Conector de Placa a Placa, Terminación Cuadrada, 2.54 mm, 10 Contactos, Receptáculo	1	1.9500	1.95 €
4.30	Conector de Placa a Placa, Terminación Cuadrada, 2.54 mm, 8 Contactos, Receptáculo	2	2.4900	4.98 €
4.31	Conector de Placa a Placa, Terminación Cuadrada, 2.54 mm, 6 Contactos, Receptáculo	1	1.2200	1.22 €
	TOTAL CAPÍTULO IV			**31.34 €**

CAPÍTULO V. SOFTWARE

N/P	*CONCEPTO*	*Uds*	*P. Unit.*	*Importe*
5.1	Tina-TI	1	0.0000	- €
5.2	Flipboard	1	0.0000	- €
5.3	Eagle	1	0.0000	133.10 €
5.4	Arduino IDE	1	0.0000	- €
5.5	LabView	1	0	3 451.00 €
	TOTAL CAPÍTULO V			**3 584.10 € €**

CAPÍTULO VI. AMORTIZACIÓN DE EQUIPOS

N/P	*CONCEPTO*	*h*	*P. Unit.*	*Importe*
6.1	Toshiba Satellite S70-A-10G	394	0.0794	31.30 €
	TOTAL CAPÍTULO VI			**31.30 €**

CAPÍTULO VII. MANO DE OBRA

N/P	*CONCEPTO*	*h*	*P. Unit.*	*Importe*
7.1	Diseño	144	9.5	1 368.00 €
7.2	Documentación	250	9.5	2 375.00 €
	TOTAL CAPÍTULO VI			**3 743.00 €**

TOTAL. IMPORTE EJECUCION MATERIAL	**7 461.25 €**

Resumen por capítulos

C.I	SENSORES	4.11 €
C.II	SOPORTE HARDWARE	19.59 €
C.III	SUBSISTEMA DE COMUNICACIÓN	47.82 €
C.IV	COMPONENTES ACONDICIONAMIENTO DE SEÑALES	31.34 €
C.V	SOFTWARE (* LICENCIAS ACADÉMICAS)	3 584.10 €
C.VI	AMORITZACIÓN DE LOS EQUIPO	31.30 €
C.VII	MANO DE OBRA	4 693.00 €

IMPORTE EJECUCIÓN MATERIAL	
13% gastos materiales	969.96 €
6% beneficio industrial	447.68 €
IMPORTE DE EJECUCION	
21% IVA	1 864.57 €
IMPORTE CONTRATA	
	10 743.46 €

Importe en letra: DIEZ MIL SETECIENTOS CUARENTA Y TRES EUROS Y CUARENTA Y SEIS CÉNTIMOS

Valencia, MARZO de 2018 Fdo

Santiago Rodríguez Díaz

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE MÁSTER DE INGENIERÍA INDUSTRIAL

DESARROLLO DE UN SISTEMA DE MONITORIZACIÓN DE UN MOTOR DE CORRIENTE CONTINUA CON SENSORES INTELIGENTES

ANEJOS

ANEJO I. MODOS DE COMUNICACIÓN

1.2. INTER INTEGRATED CIRCUIT (I2C)

El protocolo de comunicación I2C (Inter-Integrated Circuit) es de tipo síncrono, es decir, existe una señal de reloj para la sincronizar su funcionamiento, igual que en el protocolo SPI. Se trata de un bus serie de 2 hilos: SDA (Serial DAta) para la transmisión de los datos y SCL (Serial Clock Line) para la señal de reloj. Al disponer sólo una línea de datos, entonces puede enviar y recibir datos, pero no al mismo tiempo, por tanto, es una comunicación half-duplex, con 100 kHz de transmisión estándar, y un modo de alta velocidad de 400 kHz.

La topología básica del bus I2C es el que se muestra en la Figura 76, donde hay un maestro y n esclavos. Para el correcto funcionamiento del bus I2C, es necesario colocar 2 resistencias de pull-up (una resistencia para la línea SDA, y otra para la línea SCL) las cuales sirven para establecer los niveles lógicos del bus. En caso de que existen múltiples maestros, se aplica el '0' dominante para resolver los posibles conflictos. Esto es, si hay varios maestros empiezan la comunicación al mismo tiempo, el maestro cuya trama tiene más ceros (dirección) gana el arbitraje. Durante la transmisión de datos, cada maestro compara sus datos con los que hay en la línea. Si son diferentes, entonces entiende que la línea está ocupada por tanto interrumpirá su comunicación.

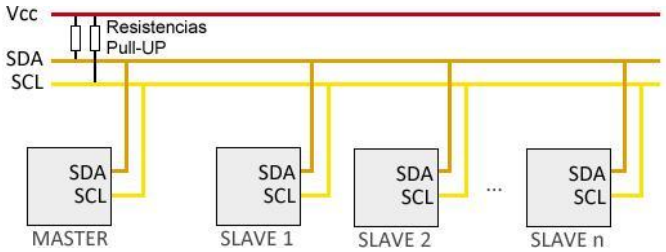


Figura 76. Protocolo de comunicación I2C.[13]

En este tipo de comunicación, el maestro inicia la transferencia de datos y sincronización mediante el direccionamiento de esclavo. Asimismo, para garantizar el envío de mensaje, hay un mecanismo de confirmación de recepción de datos.

El bus I2C emplea un formato amplio de datos enviados para así poder realizar correctamente la comunicación con un solo cable de datos. La comunicación, **¡Error! No se encuentra el origen de la referencia.**, consta de 7 bits de dirección del dispositivo esclavo, un bit para indicar si queremos enviar o recibir información, un bit de validación, uno (o más) bits correspondientes a los datos enviados o recibidos del esclavo y un bit de validación.

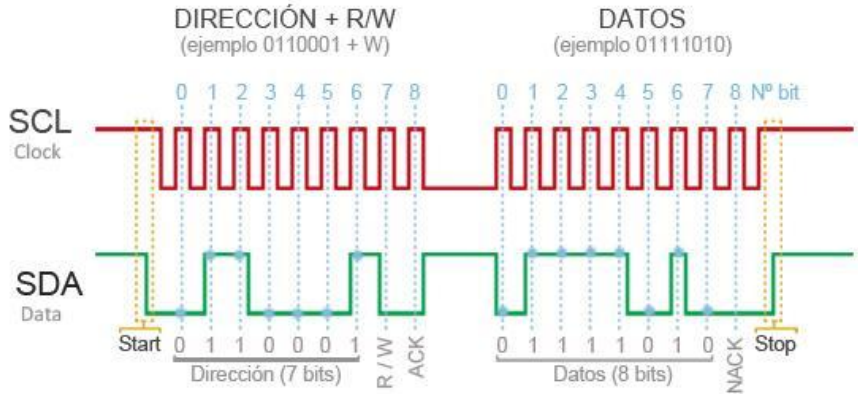


Figura 77. Funcionamiento de la comunicación I2C.[14]

En términos generales, podemos presentar una serie de ventajas e inconvenientes de este protocolo:

VENTAJAS	DESVENTAJAS
Requiere pocos cables. Esquema de direccionamiento permite la interconexión de múltiples dispositivos sin cables adicionales. Se utiliza cuando la comunicación es ocasional. Mecanismo de verificación de llegada de señal.	Velocidad transmisión relativamente baja. Half dúplex. Complejidad de implementación hardware y software.

Tabla 15. Ventajas e inconvenientes del protocolo de comunicación I2C

Es un protocolo más lento que el SPI, pero es de utilidad, dado su bajo coste, cuando los periféricos con los que permite la comunicación no deben ser necesariamente rápidos, siendo utilizado comúnmente para la transmisión de datos de control y configuraciones.

I.2. SERIAL PERIPHERAL INTERFACE(SPI)

Igual que el I2C, la comunicación SPI (Serial Peripheral Interface) es también de tipo síncrono. Este tipo de protocolo es utilizado para la comunicación de un microcontrolador con otros o con periféricos externos (microSD, sensores, convertidor analógico-digital, relojes de tiempo real, etc) que se conectan a él, con una velcidad de transmisión de hasta 8 MHz.

Consiste en un bus serie de 4 líneas (Figura 78):

- Dos líneas de datos:
  - MOSI, SIMO (Master Output Slave Input), salida de datos del maestro y entrada de datos al esclavo.
  - MISO, SOMI (Master Input Slave Output), entrada de datos en el maestro y salida de datos del esclavo.
- Dos líneas de control:

- SCLK, CLK (Clock), señal de reloj. Se trata del pulso que marca la sincronización, y con el cual se envía o lee un bit.
- SS, CS (Slave Select), utilizada para la selección del esclavo.

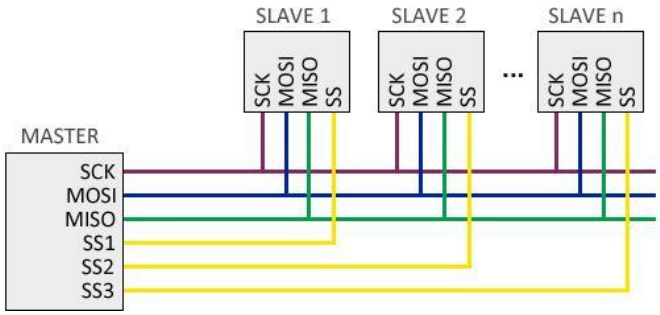


Figura 78. Topología típica del bus SPI.[15]

En este protocolo de comunicación no existe un control de flujo específico, ni un mecanismo de confirmación de recepción de datos, como ocurría en I2C. Además, no utiliza el direccionamiento de esclavos, por tanto, es necesaria una línea SS para cada dispositivo esclavo conectado, es así que si se disponen de muchos esclavos conectados este protocolo no resulta ser práctico. Para iniciar la comunicación con el esclavo se realiza a nivel de hardware mediante la línea de chip select (SS).

En el flujograma de la Figura 79 se puede ver el funcionamiento de la comunicación SPI en el presente proyecto. Iniciada la comunicación entre el microcontrolador y los sensores, se comprueba si hay un dato nuevo para enviar. En el caso de que no lo haya se sigue comprobando hasta que lo haya, y en el caso de que sí exista un dato nuevo el chip select del acelerómetro se pone a nivel bajo y el esclavo (acelerómetro) le envía los datos al maestro (microcontrolador). Tras esto, se comprueba si el sensor de temperatura tiene un dato nuevo que comunicar al maestro. SI no lo tiene, se vuelve al primer punto. Si el sensor de temperatura tiene un nuevo dato que comunicar, al igual que con el acelerómetro, su chip select se pone a nivel bajo y le comunica la nueva información al microcontrolardor. Tras ello, se vuelve al paso inicial.

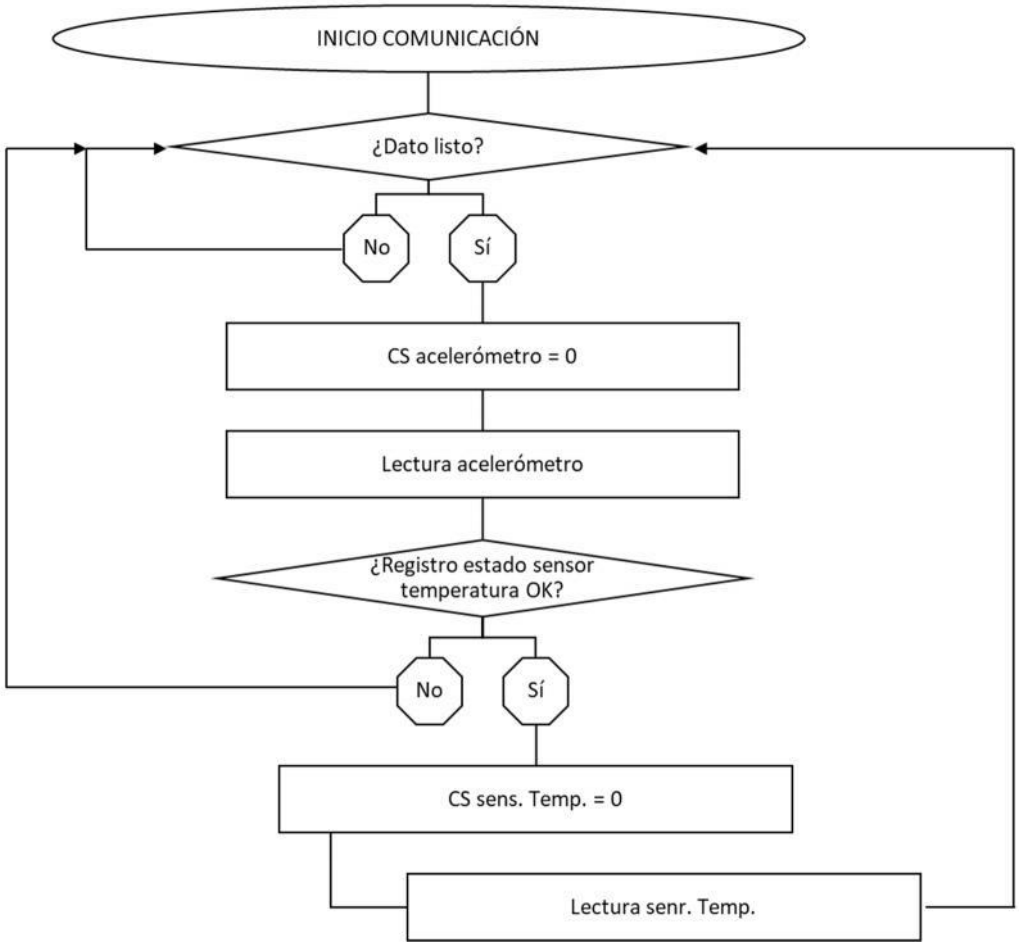


Figura 79. Flujograma de la comunicación SPI del proyecto.

Al igual que en el caso de I2C, podemos mencionar las ventajas y desventajas más destacables en este tipo de comunicación:

VENTAJAS	DESVENTAJAS
Mayor velocidad de transmisión (hasta 10 Mbits/s) Full dúplex. Puede mandar secuencias de bit de cualquier tamaño. No requiere R_p $_U$. Menos complejo al no existir direccionamiento ni mecanismos de confirmación. Menor data error.	3 líneas + 1 línea adicional por periférico.

Tabla 16. Ventajas e inconvenientes del protocolo de comunicación SPI

ANEJO II. KIT DE EVALUACIÓN ARDUINO

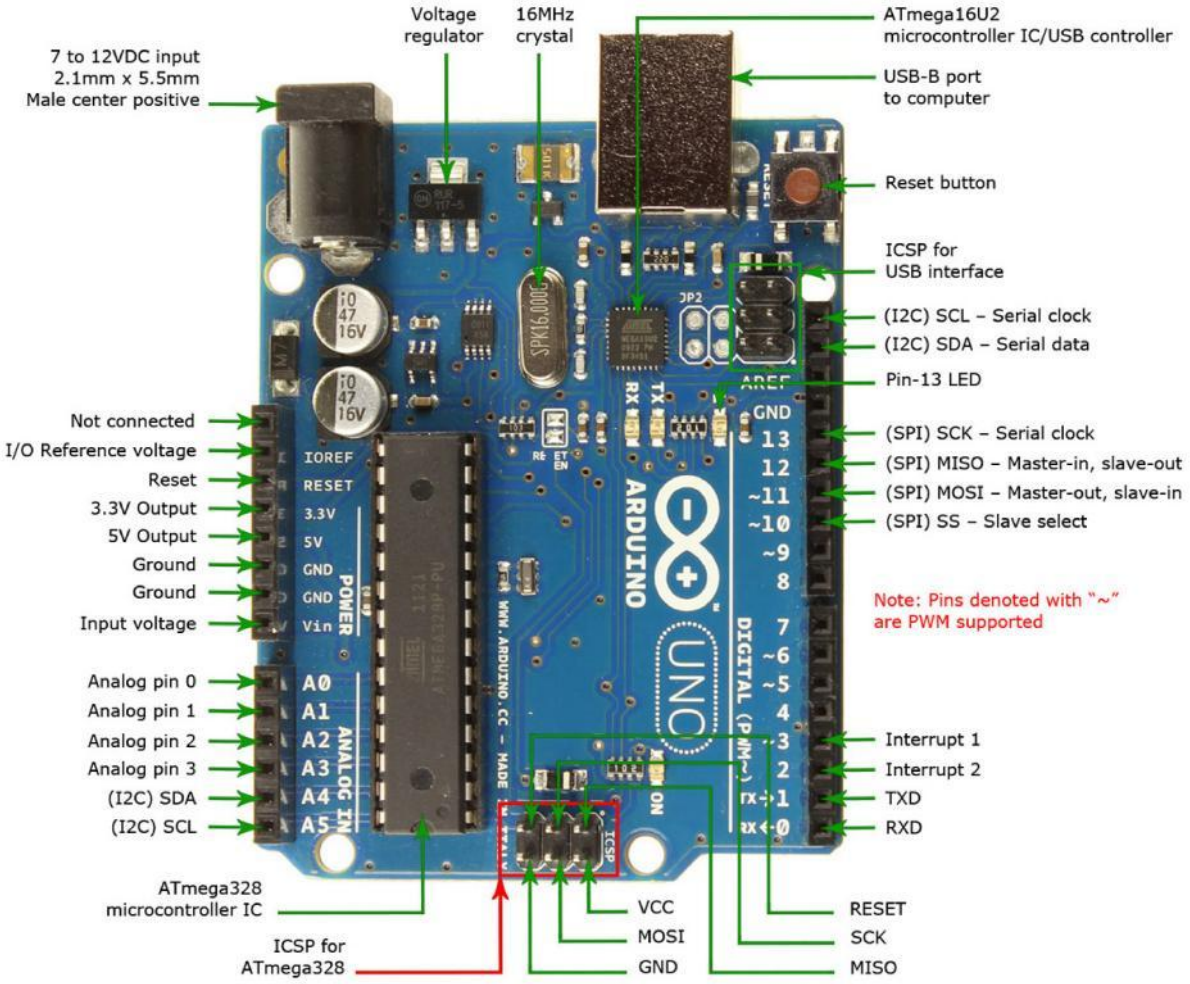


Figura 80. Esquema del kit de evaluación Arduino Uno.[16]

ANEJO III. ARDUINO SHIELD

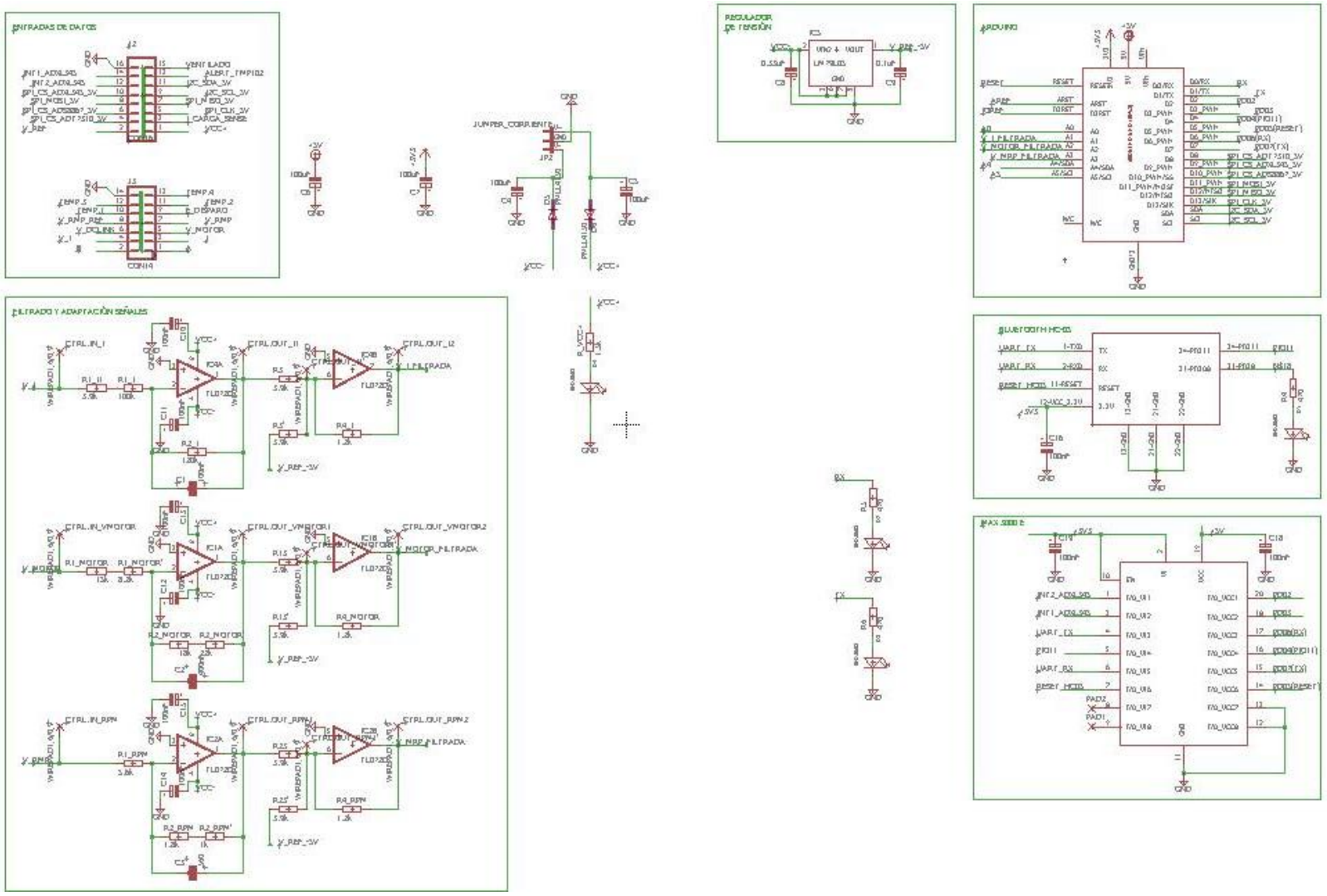


Figura 81. Esquemático del Arduino Shield de diseño propio.

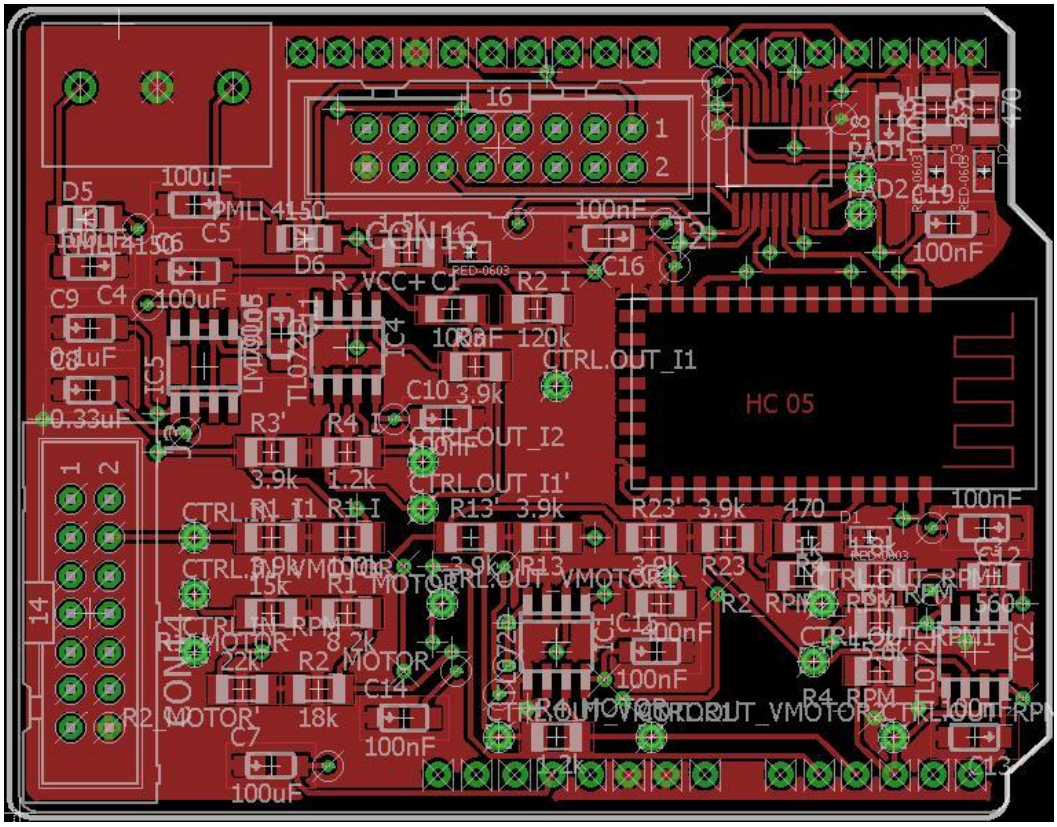


Figura 82. Capa top de la Shield Arduino de diseño propio.

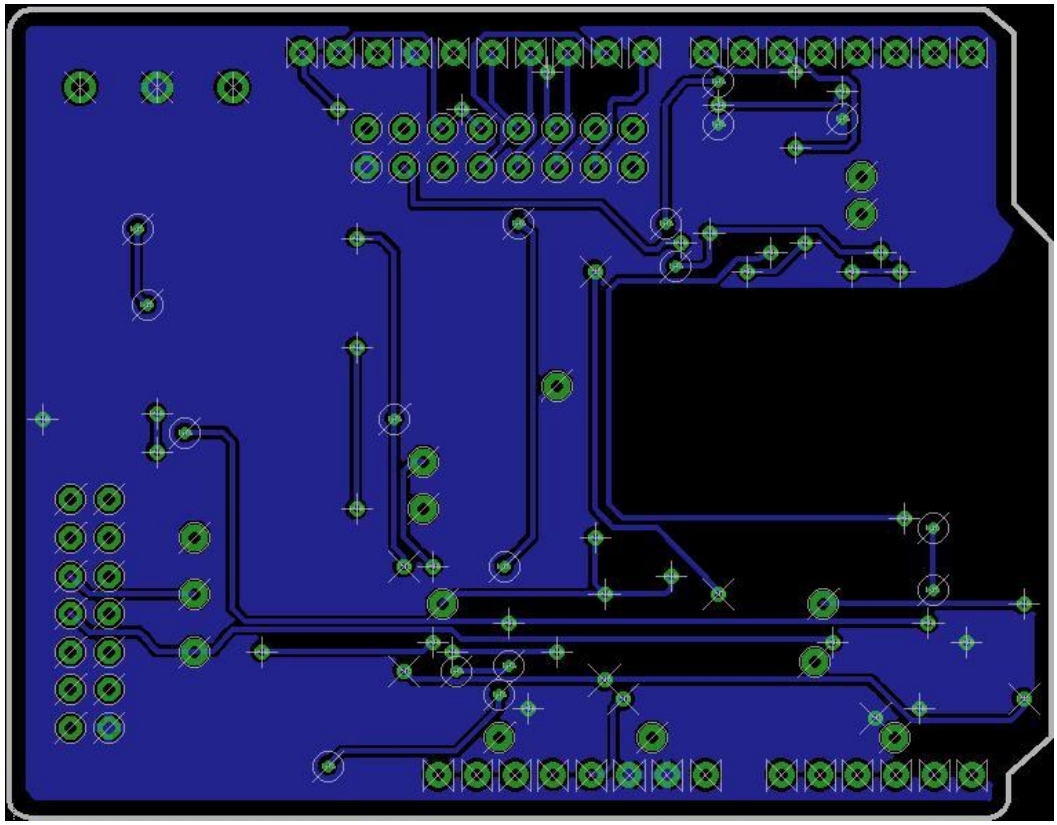


Figura 83. Capa bottom de la Shield Arduino de diseño propio.

ANEJO IV. CÓDIGO ARDUINO

```
/*Desarrollo de un sistema de monitorización de un motor de corriente continua con sensores inteligentes*/
                /*Santiago Rodríguez Díaz*/
                /*Máster Ingeniería Industrial*/
/*-------------------------------------------------------------------------------*/
/*LIBRERÍAS*/
#include <SoftwareSerial.h> //Librería para implementar un puerto serie virtual
#include <SPI.h>          //Librería para comunicación con sensores ADXL345 y ADT7310

#include <ADXL345_SPI.h>
#include <ADT7310_SPI.h>

#include "Adafruit_SleepyDog.h"        //Para resetear el microcontrolador automáticamente
#include "Adafruit_FONA.h"             //Para control de la placa Adafruit FONA sim808
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_FONA.h"        //Para acceder a Adafruit IO (acceder a la nube)
/*-------------------------------------------------------------------------------*/
/*DEFINICIÓN DE PINES*/

#define LED_PIN                        7       //Pin conectado a un LED que parpadea con el estado

//Pines lecturas analógicas
#define V_I_FILTRADO           A1
#define V_MOTOR_FILTRADO       A2
#define V_MRP_FILTRADO                 A3

//Pines del FONA
#define FONA_RX                        5       //Pin RX serie del FONA (pin 7 de la shield)
#define FONA_TX                        3       //Pin TX serie del FONA (pin 3 de la shield)
#define FONA_RST               4       //Pin de reset del FONA (pin 4 de la shield)
```

```
#define FONA_RI_INTERRUPT    6          //FONA detectará una entrada de SMS por 100 ms
// Comunicación SPI
#define CS_ADXL345  9
#define CS_ADT7310  8
#define INT0_PIN    2

ADXL345 adxl345(CS_ADXL345);
ADT7310 adt7310(CS_ADT7310);

//Comunicación Bluetooth
//#define TX_PIN       7
//#define RX_PIN       6
//#define BT_RESET_PIN  5
//#define KEY_PIN      4

//SoftwareSerial BT1(RX_PIN, TX_PIN); // RX | TX
*-----------------------------------------------------------------------------*/
/*CONFIGURACIÓN APN DE LA RED MÓVIL*/
//No todas las tarjetas SIM funcionan. Es posible que de error de conexión con la red.
//Ha sido probado con tarjeta SIM de YOIGO (sin éxito) y de VODAFONE (con éxito).

//En el siguiente enlace se pueden consultar los parámetros requeridos:

//https://wiki.bandaancha.st/APN_de_las_operadoras_para_configurar_el_m%C3%B3dem_de_Internet_m%C3%B3vil_3G

#define FONA_APN            "ac.vodafone.es"      // APN de la red móvil
#define FONA_USERNAME       "vodafone"                    // Usuario de la red móvil
#define FONA_PASSWORD       "vodafone"            // Contraseña de la red móvil
#define SIM_PIN             "9306"                // Pin de la SIM
#define NUM                 "0034610958139"       // Número correspondiente a la SIM
/*-----------------------------------------------------------------------------*/
```

```
/*SERVIDOR PARA ENVIAR LOS DATOS - ADAFRUIT IO*/

#define AIO_SERVER         "io.adafruit.com"  // Nombre del servido de Adafruit IO.
#define AIO_SERVERPORT     1883                    // Puerto de Adafruit IO.
#define AIO_USERNAME       "Sharkrodi"             // Nombre de usuario de Adafruit IO

          //(http://accounts.adafruit.com/).
#define AIO_KEY            "a6253b42cc3444f4ad6fd9684c3bf269"  // clave Adafruit IO
                                                   //(see settings page at:
https://io.adafruit.com/settings).

// Nombre de los feeds a enviar
#define ACELERACION_FEED_NAME "aceleracion" // Nombre del feed de aceleración
#define INTENSIDAD_FEED_NAME "intensidad"          // Nombre del feed de intensidad
#define TENSION_FEED_NAME "tension"                // Nombre del feed de tension
#define RPM_FEED_NAME "rpm"                             // Nombre del feed de rpm
#define TEMPERATURA_FEED_NAME "temperatura" // Nombre del feed de temperatura

#define MAX_TX_FAILURES       3  // Máximo número de fallos publicados en un rengón antes de
resetear.
/*------------------------------------------------------------------------------*/
// Global state (you don't need to change this):
SoftwareSerial fonaSS = SoftwareSerial(FONA_TX, FONA_RX);    // FONA software serial connection.
Adafruit_FONA fona = Adafruit_FONA(FONA_RST);              // FONA library connection.
const char MQTT_SERVER[] PROGMEM    = AIO_SERVER;              // MQTT server, client, username,
password.
const char MQTT_CLIENTID[] PROGMEM  = __TIME__ AIO_USERNAME;  // (stored in flash memory)
const char MQTT_USERNAME[] PROGMEM  = AIO_USERNAME;
const char MQTT_PASSWORD[] PROGMEM  = AIO_KEY;

Adafruit_MQTT_FONA mqtt(&fona, MQTT_SERVER, AIO_SERVERPORT,   // MQTT connection.
                  MQTT_CLIENTID, MQTT_USERNAME,
                  MQTT_PASSWORD);
```

```
/*------------------------------------------------------------------------------*/
/*CONFIGURACIÓN DE FEEDS PARA PUBLICAR*/

// Note that the path ends in '/csv', this means a comma separated set of values
// can be pushed to the feed

const char ACELERACION_FEED[] PROGMEM = AIO_USERNAME "/feeds/" ACELERACION_FEED_NAME "/csv";
Adafruit_MQTT_Publish aceleracion = Adafruit_MQTT_Publish(&mqtt, ACELERACION_FEED);

const char INTENSIDAD_FEED[] PROGMEM = AIO_USERNAME "/feeds/" INTENSIDAD_FEED_NAME "/csv";
Adafruit_MQTT_Publish intensidad = Adafruit_MQTT_Publish(&mqtt, INTENSIDAD_FEED);

const char TENSION_FEED[] PROGMEM = AIO_USERNAME "/feeds/" TENSION_FEED_NAME "/csv";
Adafruit_MQTT_Publish tension = Adafruit_MQTT_Publish(&mqtt, TENSION_FEED);

const char RPM_FEED[] PROGMEM = AIO_USERNAME "/feeds/" RPM_FEED_NAME "/csv";
Adafruit_MQTT_Publish rpm = Adafruit_MQTT_Publish(&mqtt, RPM_FEED);

const char TEMPERATURA_FEED[] PROGMEM = AIO_USERNAME "/feeds/" TEMPERATURA_FEED_NAME "/csv";
Adafruit_MQTT_Publish temperatura = Adafruit_MQTT_Publish(&mqtt, TEMPERATURA_FEED);
/*------------------------------------------------------------------------------*/
//Cuántos fallos de transmisión en una fila estamos dispuestos a aceptar antes de restablecer (conteo).
uint8_t txFailures = 0;
/*------------------------------------------------------------------------------*/
/*DEFINICIÓN DE VARIABLES*/

//Acelerómetro

char values[6];        // Buffer que almacena los valores leídos del registro del ADXL345
```

```
int x, y, z;          // Variables que almacenarán los valores de los ejes xyz del acelerómetro

float x_r, y_r, z_r;
float acel, acel_ef;

char dato_nuevo = 1;    // Indica si el acelerómetro tiene nuevos datos

//Contadores
int contador = 0;
int contador_analog = 0;        // Cuenta el número de veces que se lee el acelerómetro antes de leer los
                        //datos analógicos y el sensor de tempertura, pues la frecuencia de muestreo
                        //de este último es menor
int contador_GPRS = 0;


//Sensor de temperatura
unsigned int read_value_temp = 0; //Lee la salida del sensor de temperautra
float temp = 0;          // Almacena la temperatura en grados centrígrados
float temp_ef = 0;

//Variables analógicas
int V_I, V_MOTOR, V_MRP;                  // variable que almacena el valor raw (0 a 1023)
float value_V_I, value_V_MOTOR, value_V_MRP;  // variable que almacena el voltaje (0.0 a 3.0)
float I_r, V_r, RPM_r;
float I_ef, V_ef, RPM_ef;

// Envío de datos
String stringToSend = "";

//Segundos entre envío GPRS - no superior a 8 segundos
```

```
 //si seg>8 segundos, Watchdog reseteará el proyecto
float seg = 1;
/*-------------------------------------------------------------------------------*/
void setup()
{
 /*CONFIGURACIÓN COMUNICACIÓN SPI*/

  /*Configura la Inrerrupcion*/
 attachInterrupt(0, dato_listo, RISING);    // IRQ 0 = pin 2 , saltarí a la función
                                 //" dato_listo "  , cuando la señal " suba "
                                 //( pase de 0 a 1 ) sagun manual de ADXL345
 pinMode(INT0_PIN, INPUT);

  /*Inicio comunicación SPI*/
 SPI.begin();
 SPI.setBitOrder(MSBFIRST);

  /*Condigura la conexión SPI pra los sensores*/
 SPI.setDataMode(SPI_MODE3);

 //Configuro la conexion USB del Arduino
 Serial.begin(115200, SERIAL_8N1);

 while (!Serial);

  /*Reseteo de los sensores - Condiciones de funcionamiento de serie*/
 adxl345.reset();
 adt7310.reset();

/*CONFIGURACIÓN SENSORES*/

 /*ADXL345*/
```

```
  //Configuro en el registro de configuración (0x01) el modo de lectura 1SPS y 16bits de resolución (0xC0)
 adt7310.write(0x01, 0xC0);

//

 //Put the ADXL345 into +/- 4G range by writing the value 0x01 to the DATA_FORMAT register.
 //adxl345.write(0x31, 0x01);

 //Pongo el ADXL345 en el rango +/- 8G escribiendo el valor 0x02 en el registro DATA_FORMAT.
 adxl345.write(0x31, 0x02);

 //Put the ADXL345 into +/- 16G range by writing the value 0x03 to the DATA_FORMAT register.
 //adxl345.write(0x31, 0x03);

//

 //Put the ADXL345 into 100Hz range by writing the value 0x0A to the BW_RATE register.
    //- - Necesita SerialBaudRateMinimo de 115200
 //adxl345.write(0x2C, 0x0A);
 //Put the ADXL345 into 200Hz range by writing the value 0x0B to the BW_RATE register.
    // - - Necesita SerialBaudRateMinimo de 115200
 //adxl345.write(0x2C, 0x0B);

 //Pongo el ADXL345 en el rango de 400Hz escribiendo el valor 0x0C en el registro BW_RATE.
    //- - Necesita SerialBaudRateMinimo de 115200
 adxl345.write(0x2C, 0x0C);

 //Put the ADXL345 into 800Hz range by writing the value 0x0C to the BW_RATE register.
    //- - Necesita SerialBaudRateMinimo de 230400
 //adxl345.write(0x2C,0x0D);
```

```
//

 //Configuración del DATA_READY INT del ADXL345 escribiendo el valor 0x80 en el registro INT_MAP
  //para enviear su respectiva INT por el pin INT2.
 adxl345.write(0x2F, 0x80);

//

 //Enable the ADXL345 DATA_READY INT by writing the value 0x80 to the INT_ENABLE register.
 adxl345.write(0x2E, 0x80);

//

 //Put the ADXL345 into Measurement Mode by writing 0x08 to the POWER_CTL register.
 adxl345.write(0x2D, 0x08); //Measurement mode

//

 /*ADT7310*/
  //Contador de lectura del adt7310
 contador = 0;

 /*Lecturas analógicas*/
  //Contador de lecturas analógicas
 contador_analog = 0;

 /*Valoles eficaces*/

 acel = 0;

 acel_ef = 0;
```

```
 I_ef = 0;
 V_ef = 0;
 RPM_ef = 0;
//------------------------------------------------------------------------
//
// /*MÓDULO BT*/
//
//   /*          MODO COMANDOS AT    */
//
//     //Primero lo ponemos en modo de comandos AT para cambiar los parámetros
// digitalWrite(KEY_PIN, HIGH); // Al poner en HIGH forzaremos el modo AT
//
//      //Hacemos un reset del módulo BT, para que identifique que estamos en modo de comandos AT
// digitalWrite(BT_RESET_PIN, LOW);
//
// delay (100);//Espero 100ms (>5ms) para que el RESET tenga efecto
//
// digitalWrite(BT_RESET_PIN, HIGH);
//
//     //Configuramos el puerto serie virtual con un Baudrate de 38400bps, que es la velocidad
//     //de transmisión prefijada por el fabricante para este modo de funcionamiento.
//     //Nota: el usuario no puede cambiar el baudrate del modo comandos AT.
// BT1.begin(38400);
//
// while (!BT1);
//
// delay (1000);  //Para que el módulo BT pueda inicializarse
//
//     //ponemos el módulo BT con valores iniciales de fábrica
//   BT1.println("AT+ORGL");//Slave mode, pin code :1234, device name: H-C-2010-06-01 ,Baud 38400bits/s.
```

```
//
//  delay (100); //Hay que poner Delay's entre instrucciones de los comandos AT
//
//     //fijamos el baudrate del módulo BT para cuando este funcione
//     //en modo puerto serie.
//  BT1.println("AT+UART=115200,0,0");
//
//  delay (100); //Hay que poner Delay's entre instrucciones de los comandos AT
//
//     //Comando para cambiar el nombre del dispositivo
//  BT1.println("AT+NAME=SRD");
//
//  delay (100);
//
//     //Comando para cambiar la clase de dispositivo (CoD), según aplicación
//     //de la web http://www.ampedrftech.com/cod.htm, con los parámetros
//     //Service Class=Capturing (Scanner, Microphone)
//     //Major Device Class=Peripheral
//     //Minor Device Class=Not Keyboard / Not Pointing Device + Sensing device
//  BT1.println("AT+CLASS=80510");//Esta es la clase de dispo
//
//  delay(100);
//
//   /*          MODO COMANDOS SerialPort    */
//     //Tras configurar los parametros necesarios, pasamos el módulo BT a modo puerto serie
//     //Cambiamos la velocidad del puerto serie virtual del Arduino a la velocidad que
//     //va a utilicar el módulo BT en modo SerialPort
//
//  BT1.begin(115200);
//
//  while (!BT1);
//
```

```
//  digitalWrite(KEY_PIN, LOW); // Al poner en LOW forzaremos el modo SerialPort
//
//      //Hacemos un reset del módulo BT, para que identifique que estamos en modo de comandos AT
//  digitalWrite(BT_RESET_PIN, LOW);
//
//  delay (100);//Espero 10ms (>5ms) para que el RESET tenga efecto
//
//  digitalWrite(BT_RESET_PIN, HIGH);
//------------------------------------------------------------------------
        delay (100);  //hacemos un pequeño delay para que el módulo BT pueda inicializarse
/*-------------------------------------------------------------------------------*/
        Serial.println(F("Seguimiento - FONA 808 y Adafruit IO"));

/*      //Configura el pin del LED y lo pone a nivel bajo
        pinMode (LED_PIN, OUTPUT);
        digitalWrite (LED_PIN, LOW); */

        //Inicio del módulo FONA
        Serial.println(F("Inicializando FONA....(tandará 10 s)"));
        fonaSS.begin(4800);
        if (!fona.begin(fonaSS))
        {
                halt(F("No se ha podido encontrar FONA"));
        }
        Serial.println(F("[U] Desbloquea SIM con código PIN"));
        char PIN[5]=SIM_PIN;
        flushSerial();
        Serial.println(PIN);
        Serial.print(F("Desbloqueando SIM card: "));
        if (! fona.unlockSIM(PIN))
        {
```

```
        Serial.println(F("FALLO"));
}
else
{
        Serial.println(F("OK!"));
}

//Establecimiento de interrupción por SMS
pinMode (FONA_RI_INTERRUPT, INPUT);
digitalWrite (FONA_RI_INTERRUPT, HIGH);
fona.setSMSInterrupt(1);

//Se utiliza watchdog para simplificar la lógica de reinicio.
//Se habilita despues de que FONA se inicia porque tarda sobre 8-9 segundos.
Watchdog.enable(16000);
Watchdog.reset();

//Espera a que FONA se conecte a la red móvil (8 segundos, despues watchdog reinicia)
Serial.println (F("Comprobando network..."));
while (fona.getNetworkStatus() != 1)
{
        delay (500);
}

Serial.println (F("Comprobado"));

//Inicio de los datos de conexión GPRS
Watchdog.reset();
fona.setGPRSNetworkSettings (F(FONA_APN), F(FONA_USERNAME), F(FONA_PASSWORD));
delay (2000); //espera 2 s a que se inicie

Watchdog.reset();
```

```
        Serial.println (F("GPRS inhabilitado"));
        fona.enableGPRS(false);
        delay (2000);

        Watchdog.reset();
        Serial.println (F("Permitir GPRS"));
        if (!fona.enableGPRS(true))
        {
                halt (F("Fallo al activar GPRS, reseteando..."));
        }
        Serial.println (F("Conectado a la red móvil"));

        //Espera un momento a que se estabilice la conexión
        Watchdog.reset();
        delay (3000);

        //Conexión MQTT
        int8_t ret = mqtt.connect();

        while (ret = mqtt.connect ())
        {
                Serial.println (mqtt.connectErrorString(ret));
  ret = 0;
        }
        Serial.println (F("¡MQTT conectado!"));


}
*-----------------------------------------------------------------------------*/
void loop()
{
```

```
 /*LECTURA DE DATOS*/

 /*Cuando el aceleróemtro dispone de un dato nuevo se entra en este condición*/
 if (dato_nuevo >= 1)
 {
  //Leemos 6 Bytes a partir del registro DATAX0
  adxl345.read(0x32, 6, values);

  //El ADXL345 proporciona valores de aceleración de 10 bits, pero son almacenados como bytes (8 bits)
    //Para obtener el valor completo, se deben convinar dos bytes para cada eje

  //El valor de X está almacenado en values[0] y values[1]
  x = ((int)values[1] << 8) | (int)values[0];
  x_r = x * 8 / 512;

  //El valor de Y está almacenado en values[2] y values[3]
  y = ((int)values[3] << 8) | (int)values[2];
  y_r = y * 8 / 512;

  //El valor de Z está almacenado en values[4] y values[5]
  z = ((int)values[5] << 8) | (int)values[4];
  z_r = z * 8 / 512;

  //Actualizo Variable para esperar la nueva interrupción
  dato_nuevo = 0;

  /*Leo las variables analógicas cada muestreo del ADXL345*/

  if (contador_analog >= 1)
  {
    /*Lectura entradas analógicas*/
```

```
  V_I = analogRead(V_I_FILTRADO);      // realizar la lectura analógica
  V_MOTOR = analogRead(V_MOTOR_FILTRADO);
  V_MRP = analogRead(V_MRP_FILTRADO);

   /*Conversión de la entrada analógica (que va de 0 - 1023) a un voltage (0 - 5V):*/
  value_V_I = V_I*5/1023;
  value_V_MOTOR = V_MOTOR*5/1023;
  value_V_MRP = V_MRP*5/1023;

   /*Conversión a magnitudes reales:*/
  I_r = value_V_I * 2.885 - 4.457;
  V_r = (value_V_MOTOR * 2.608 - 3.952) * 10;
  RPM_r = (value_V_MRP * 6.494 - 9.935) * 300;

  contador_analog = 0;

}

if (contador_GPRS <= (400*seg))
{
  //Sumatorio para cálculo de valores eficaces
  acel = acel + sqrt (x_r*x_r + y_r*y_r + z_r*z_r);

  I_ef = I_ef + I_r*I_r;
  V_ef = V_ef + V_r*V_r;
  RPM_ef = RPM_ef + RPM_r*RPM_r;
}

/* y el sensor de temperatura cuando se han realizado
400 muestreos del acelerómetro*/
if (contador >= 400)
{
```

```
  /*Lectura de la temperatura*/
    //Primero se comprueba que hay dato listo revisando el registro de estado (0x00)
 read_value_temp = adt7310.read(0x00, 8);

    //Ahora comprobamos si el bit 7 de ese registro está a 1 o a 0.
      //  Si está a 1 se procede a la lectura
 if (!(read_value_temp && 0x80))
 {
  read_value_temp = 0;

     // Lee el valor de los 16 bit de temperatura del registro 0x02
  read_value_temp = adt7310.read(0x02, 16);

     // Convierte a temperatura en punto flotante en grados centrígrados
  temp = adt7310.temperature(read_value_temp,16);

  temp_ef = temp_ef + temp*temp;

  contador = 0;
 }

}

if (contador_GPRS >= (400*seg))
{
                /*Cálculo de valores medios y eficaces*/

                acel_ef = sqrt(acel/(400*seg));

                I_ef = sqrt(I_ef/(400*seg));
                V_ef = sqrt(V_ef/(400*seg));
                RPM_ef = sqrt(RPM_ef/(400*seg));
```

```
                temp_ef = sqrt(temp_ef/((400*seg)/400));

                contador_GPRS = 0;
    //-----------------------------------------------------------------
    //SMS

    if (temp_ef >= 39 && mensaje == 0)
    {
     char callerIDbuffer[32] = "+34680395036";  //we'll store the SMS sender number in here

     //Send back an automatic response
     Serial.println("Sending reponse...");
     if (!fona.sendSMS(callerIDbuffer, "La temperatura de tu motor es excesiva! Cambie las condiciones de funcionamiento de sus sistema para reducir la temperatura y evitar daños mayores."))
     {
      Serial.println(F("Failed"));
     }
     else
     {
      Serial.println(F("Sent!"));
     }
     mensaje = 1;
    }
    //-----------------------------------------------------------------

/*      //Crea una cadena de caracteres para mostrar todos los datos
    stringToSend = String(acel, DEC);

    stringToSend = stringToSend + "\n";
```

```
        stringToSend = stringToSend + String(I_ef, DEC);
    stringToSend = stringToSend + ", ";
    stringToSend = stringToSend + String(V_ef, DEC);
    stringToSend = stringToSend + ", ";
    stringToSend = stringToSend + String(RPM_ef, DEC);

    stringToSend = stringToSend + "\n";

    stringToSend = stringToSend + String(temp_ef, DEC);

    stringToSend = stringToSend + "\n";

        stringToSend = stringToSend + "-------------------------------------------------";

    Serial.println(stringToSend);
*/

                // Watchdog reiniciado al inicio del bucle loop --> debemos asegurar que todo lo que sigue se ejecute en menos de 8 s
                Watchdog.reset();

                // Resetear todo si se desconecta o se detectan demasiados errores en la línea.
                if (!fona.TCPconnected() || (txFailures >= MAX_TX_FAILURES))
                {
                        halt(F("Connection lost, resetting..."));
                }

                 //Watchdog.reset();

                publicar (acel_ef, aceleracion);
                publicar (I_ef, intensidad);
                publicar (V_ef, tension);
```

```
                publicar (RPM_ef, rpm);
                publicar (temp_ef, temperatura);

                // Resetea el contador para esperar otras 400 mediciones del acelerómetro

                I_ef = 0;
                V_ef = 0;
                RPM_ef = 0;

                temp = 0;
                temp_ef = 0;

                delay (100);

}

        //Crea una cadena de caracteres para mostrar todos los datos

/*ACELERÓMETRO*/
stringToSend = String(x, DEC);
stringToSend = stringToSend + ", ";
stringToSend = stringToSend + String(y, DEC);
stringToSend = stringToSend + ", ";
stringToSend = stringToSend + String(z, DEC);

  // Si el contador_analog está a 0 quiere decir que se midió datos analógicos
  /*VARIABLES ANALÓGICAS*/
if (contador_analog == 0)
{
  stringToSend = stringToSend + ", ";
  stringToSend = stringToSend + String(V_I, DEC);
  stringToSend = stringToSend + ", ";
```

```
    stringToSend = stringToSend + String(V_MOTOR, DEC);
    stringToSend = stringToSend + ", ";
    stringToSend = stringToSend + String(V_MRP, DEC);

  }

  // Si el contador está a 0 quiere decir que se midió dato de temperatura
  if (contador ==0)
  {
   /*SENSOR DE TEMPERATURA*/
    stringToSend = stringToSend + ", ";
    stringToSend = stringToSend + String(read_value_temp, DEC);
  }

        /*IMPRESIÓN DATOS LEÍDOS*/

  //La siguiente instrucción envía el String generado por BT
  //BT1.println(stringToSend);

  //La siguiente instrucción envía el String generado por el USB
  Serial.println(stringToSend);

 }

  else
 {
  delayMicroseconds(100);
 }
}
/*-------------------------------------------------------------------------------*/
/*FUNCIONES A LLAMAR EN LA EJECUCIÓN DEL PROYECTO*/
```

```
// Función a la que saltaría nuestro Arduino cuando se active (cuando la señal pase de 1 a 0 )
  //en el pin 2 de nuestro Arduino
void dato_listo()
{
  dato_nuevo = 1;
  contador++;
  contador_analog++;
  contador_GPRS++;
}
/*--------------------------------------------------------------------------------*/
```

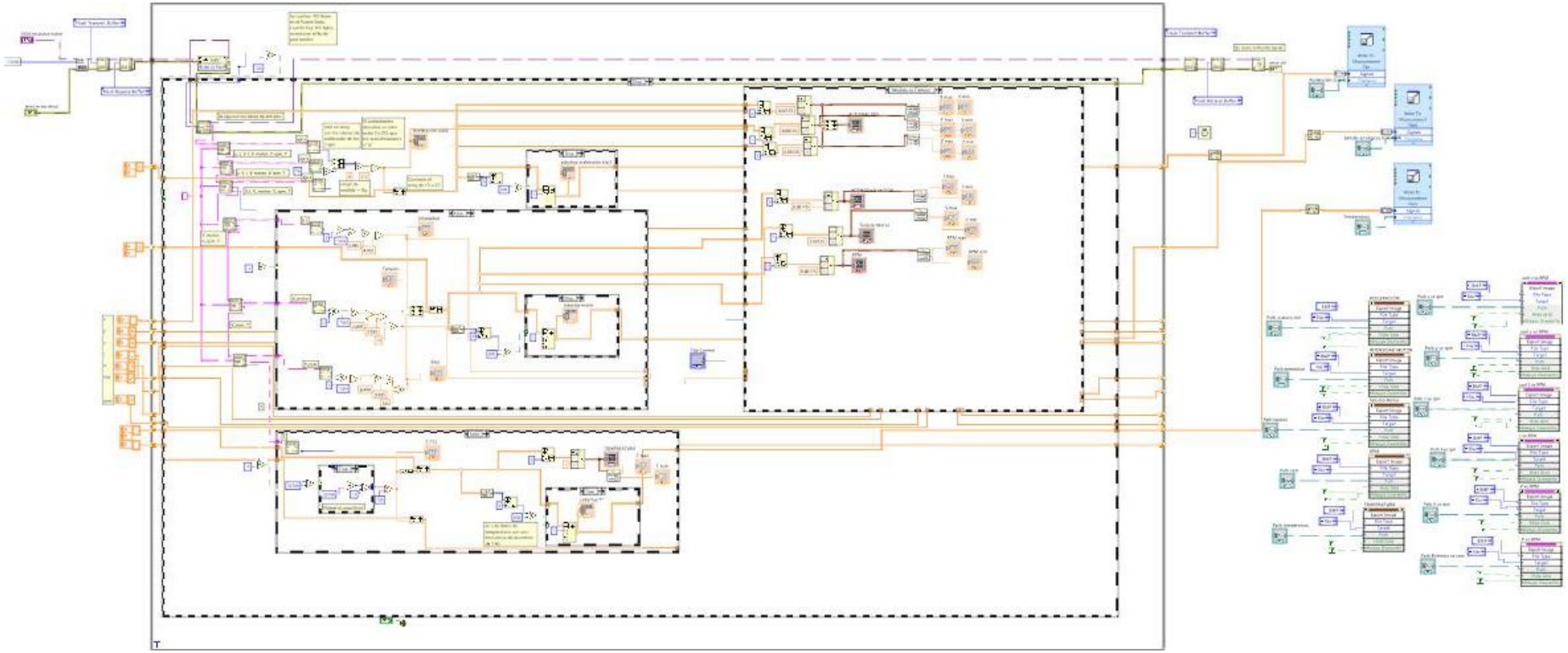
//La función "Halt" se llamará cuando ocurra un error. Imprimirá un error y parará la ejecución mientras se hace un parpadeo rápido del LED. Si el watchdog está habilitado reseteará después de 8 segundos.

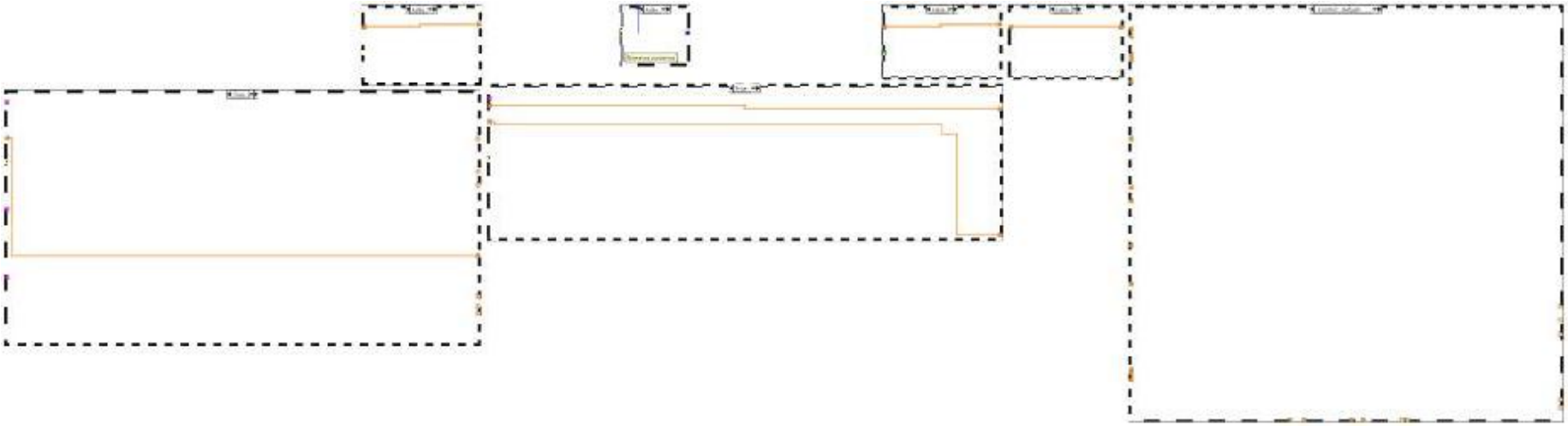
```
void halt (const __FlashStringHelper *error)
{
        Serial.println(error);
        while(1)
        {
/*              digitalWrite (LED_PIN, LOW);
                delay (100);
                digitalWrite (LED_PIN, HIGH);
                delay (100); */
        }
}

void flushSerial()
{
  while (Serial.available())
  Serial.read();
}
```

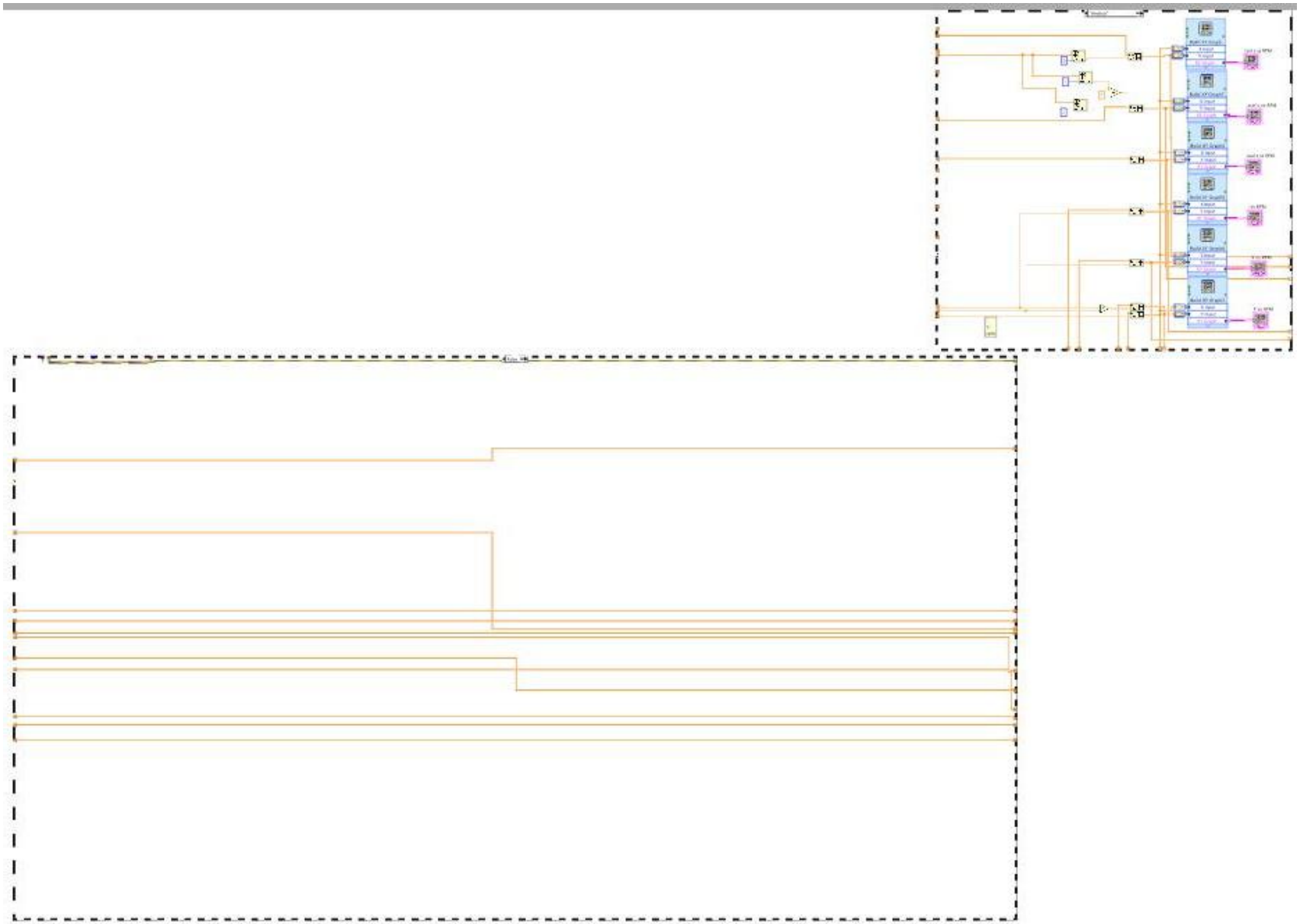
```
// La función "publicar" publica los datos en la web.

void publicar (float feed, Adafruit_MQTT_Publish & publishFeed)
{
        //Crea un buffer tipo string que contine los datos que se publicarán en la web.
        char enviarBuffer [120];
        memset(enviarBuffer, 0, sizeof (enviarBuffer));
        int index = 0;

        //"dtostrf" convertir un número en cadena
        dtostrf (feed, 1, 2, &enviarBuffer[index]);
/*      |                   | |       \_ buffer donde almacenaremos la cadena
        |               |   \_ Precisión (nº decimales)
                                  |               \_Tamaño del número en carácteres
        \_ Número a convertir
*/
        //"strlen" longitud del string
//      index += strlen (&enviarBuffer[index]);
//      enviarBuffer[index++] = ', ';

        // Se publica el string.
        Serial.print (F("Publicando: "));
        Serial.println (enviarBuffer);
        if (!publishFeed.publish(enviarBuffer))
        {
                Serial.println (F("¡Publicación fallida!"));
                txFailures++;
        }
        else
        {
                Serial.println(F("¡Publicación exitosa!"));
                txFailures = 0;
```

```
        }
}
```

ANEJO V. PROGRAMACIÓN LABVIEW







BIBLIOGRAFÍA

[1] «SIM800L Datasheet - 4-band GSM/GPRS Module - SIMCom», 2017. [En línea]. Disponible en: http://www.datasheetcafe.com/sim800l-datasheet-simcom/. [Accedido: 20-feb-2018].

[2] «Preditécnico: Instalación de sensores de vibración en motores eléctricos». [En línea]. Disponible en: http://www.preditecnico.com/2012/12/instalacion-de-sensores-de-vibracion-en.html. [Accedido: 19-feb-2018].

[3] «Guía de selección de puntos de medición de vibraciones | pruftechnik». [En línea]. Disponible en: https://pruftechnik.wordpress.com/2013/04/24/guia-de-seleccion-de-puntos-de-medicion-de-vibraciones/. [Accedido: 19-feb-2018].

[4] P. Yiyao, Y. Lin, F. Javier, y G. Casado, «Instrumentación en Red y Comunicaciones».

[5] P. Yiyao, Y. Lin, F. Javier, y G. Casado, «TECNOLOGIA ELECTRÓNICA (GITI). PRÁCTICA 2 : Estudio y medidas sobre un convertidor de potencia en puente completo con modulación PWM unipolar para el control de velocidad de un motor de corriente continua . motor de corriente continua . En la práctica 1».

[6] 'Texas Instruments Incorporated', «MSP-EXP430G2 MSP430 LaunchPad Value Line Development kit | TI.com». [En línea]. Disponible en: http://www.ti.com/tool/msp-exp430g2. [Accedido: 20-feb-2018].

[7] 'Texas Instruments Incorporated', «MSP430FR2633 MSP430 MCU with CapTIvate technology - the lowest power, most noise-immune capacitive touch | TI.com». [En línea]. Disponible en: http://www.ti.com/product/msp430fr2633/description. [Accedido: 20-feb-2018].

[8] Steven Cogswell, «The inability to follow simple instructions», 2013. [En línea]. Disponible en: https://awtfy.com/tag/arduino/. [Accedido: 20-feb-2018].

[9] Steven Cogswell, «An Arduino Library for the ADT7310 SPI Temperature Sensor – The inability to follow simple instructions», 2010. [En línea]. Disponible en: https://awtfy.com/2010/11/09/an-arduino-library-for-the-adt7310-spi-temperature-sensor/. [Accedido: 20-feb-2018].

[10] 'SparkFun Electronics', «ADXL345 Hookup Guide - learn.sparkfun.com», 2016. [En línea]. Disponible en: https://learn.sparkfun.com/tutorials/adxl345-hookup-guide#introduction-. [Accedido: 20-feb-2018].

[11] Sinais Ingeniería, «Sinais ingenieria». [En línea]. Disponible en: http://www.sinais.es/Recursos/Curso-vibraciones/motores/frecuencias_motores.html. [Accedido: 19-feb-2018].

[12] J. R. Santa Anna Zamudio, «Bluetooth HC-05 y HC-06 Tutorial de Configuración - Geek Factory», 2014. [En línea]. Disponible en: https://www.geekfactory.mx/tutoriales/bluetooth-hc-05-y-hc-06-tutorial-de-configuracion/. [Accedido: 20-feb-2018].

[13] M. Pérez Esteso, «INTERNET DE LAS COSAS [PARTE 2] – SUBIR LOS DATOS A UNA BASE DE DATOS». [En línea]. Disponible en: https://geekytheory.com/internet-de-las-cosas-parte-2-subir-los-datos-a-una-base-de-datos. [Accedido: 20-feb-2018].

[14] M. Pérez Esteso, «LabVIEW + Arduino: Voltímetro». [En línea]. Disponible en: https://geekytheory.com/labview-arduino-voltimetro. [Accedido: 20-feb-2018].

[15] M. Pérez Esteso, «Weblet Importer». [En línea]. Disponible en: https://geekytheory.com/category/tutoriales-arduino?page=6. [Accedido: 20-feb-2018].

[16] P. Nelson Saavedra, «LA MEDICION Y ANALISIS DE LAS VIBRACIONES COMO TECNICA DE INSPECCION DE EQUIPOS Y COMPONENTES, APLICACIONES, NORMATIVAS Y CERTIFICACION».

[17] 'Naylamp Mechatronics SAC', «Tutorial Básico de Uso del Módulo Bluetooth HC-06 y HC-05». [En línea]. Disponible en: http://www.naylampmechatronics.com/blog/12_Tutorial-Básico-de-Uso-del-Módulo-Bluetooth-H.html. [Accedido: 20-feb-2018].

[18] National-Instruments-Corporation, «Comunicación Serial Utilizando LabVIEW con un Microcontrolador - National Instruments», 2008. [En línea]. Disponible en: http://www.ni.com/white-paper/7907/es/. [Accedido: 20-feb-2018].

[19] 'Metas & Metrólogos Asociados', «Criterios Básicos para la Selección de Sensores de Temperatura Somos su Relevo a la Calidad». 2006.

[20] J. Mayné, «Sensores, acondicionadores y procesadores de señal», Silica an Avnet Div. Los pasos a seguir para …, vol. 1, pp. 1-65, 2003.

[21] 'Maxim Integrated', «MAX3004 +1.2V to +5.5V, ±15kV ESD-Protected, 0.1µA, 35Mbps, 8-Channel Level Translators - Maxim». [En línea]. Disponible en: https://www.maximintegrated.com/en/products/interface/level-translators/MAX3004.html. [Accedido: 20-feb-2018].

[22] Marco Schwartz, «Monitor Data From Anywhere With Arduino & the Adafruit FONA», 2016. [En línea]. Disponible en: https://openhomeautomation.net/monitor-data-arduino-fona. [Accedido: 20-feb-2018].

[23] Luis LLamas, «El bus I2C en Arduino», 2016. [En línea]. Disponible en: https://www.luisllamas.es/arduino-i2c/. [Accedido: 19-feb-2018].

[24] Luis LLamas, «El bus SPI en Arduino», 2016. [En línea]. Disponible en: https://www.luisllamas.es/arduino-spi/. [Accedido: 19-feb-2018].

[25] L. Llmas, «Usar un acelerómetro ADXL345 con Arduino», 2016. [En línea]. Disponible en: https://www.luisllamas.es/arduino-acelerometro-adxl345/. [Accedido: 20-feb-2018].

[26] L. LLamas, «Conectar Arduino por Bluetooth con los módulos HC-05 ó HC-06», 2015. [En línea]. Disponible en: https://www.luisllamas.es/conectar-arduino-por-bluetooth-con-los-modulos-hc-05-o-hc-06/. [Accedido: 20-feb-2018].

[27] L. Llamas, «¿Qué es Arduino? ¿Qué modelo comprar?», 2013. [En línea]. Disponible en: https://www.luisllamas.es/que-es-arduino-que-modelo-comprar/. [Accedido: 20-feb-2018].

[28] L. Llamas, «Cómo usar un acelerómetro en nuestros proyectos de Arduino». [En línea]. Disponible en: https://www.luisllamas.es/como-usar-un-acelerometro-arduino/. [Accedido: 20-feb-2018].

[29] 'Libelium Comunicaciones Distribuidas S.L.', «GPRS+GPS Quadband Module for Arduino, Raspberry Pi and Intel Galileo (SIM908)». [En línea]. Disponible en: https://www.cooking-hacks.com/gprs-gps-quadband-module-for-arduino-sim908. [Accedido: 20-feb-2018].

[30] Javier Brathwaite, «Conectar Arduino a Base de Datos Mysql | Panama Hitek», 2015. [En línea]. Disponible en: http://panamahitek.com/conectar-arduino-base-datos-mysql/. [Accedido: 20-feb-2018].

[31] Iván Lozano, «Cuatro alternativas a Arduino: BeagleBone, Raspberry Pi, Nanode y Waspmote», 2013. [En línea]. Disponible en: https://blogthinkbig.com/4-alternativas-arduino-beaglebone-raspberrypi-nanode-waspmote. [Accedido: 20-feb-2018].

[32] HETPRO/TUTORIALES, «Comunicacion Arduino y Labview por Bluetooth», 2014. [En línea]. Disponible en: https://hetpro-store.com/TUTORIALES/comunicacion-arduino-y-labview-por-bluetooth/. [Accedido: 20-feb-2018].

[33] HETPRO/TUTORIALES, «Arduino LabVIEW Sensor de temperatura NTC SNS TMP10K HETPRO TUTORIALES», 2017. [En línea]. Disponible en: https://hetpro-store.com/TUTORIALES/arduino-labview-sensor-temperatura/. [Accedido: 20-feb-2018].

[34] Gustavo Circelli, «Acelerómetros de 3 ejes, lo que necesitas saber | Panama Hitek», 2015. [En línea]. Disponible en: http://panamahitek.com/acelerometros-de-3-ejes-lo-que-necesitas-saber/. [Accedido: 20-feb-2018].

[35] R. Flores y T. I. Asiaín, «Diagnóstico de Fallas en Máquinas Eléctricas Rotatorias Utilizando la Técnica de Espectros de Frecuencia de Bandas Laterales», Inf. tecnológica, vol. 22, n.o 4, pp. 73-84, 2011.

[36] FERNANDO DOUTEL, «Raspberry Pi frente a Arduino: ¿quién se adapta mejor a mi proyecto maker?», 2015. [En línea]. Disponible en: https://www.xataka.com/makers/raspberry-pi-frente-a-arduino-quien-se-adapta-mejor-a-mi-proyecto-maker. [Accedido: 20-feb-2018].

[37] Enrique Crespo, «Bluetooth en Arduino | Aprendiendo Arduino». [En línea]. Disponible en: https://aprendiendoarduino.wordpress.com/2016/11/13/bluetooth-en-arduino/. [Accedido: 20-feb-2018].

[38] Enrique Crespo, «Arduino vs Raspberry Pi. | Aprendiendo Arduino», 2016. [En línea]. Disponible en: https://aprendiendoarduino.wordpress.com/2016/03/28/arduino-vs-raspberry-pi/. [Accedido: 20-feb-2018].

[39] Enrique Crespo, «Adafruit IO | Aprendiendo Arduino». [En línea]. Disponible en: https://aprendiendoarduino.wordpress.com/tag/adafruit-io/. [Accedido: 19-feb-2018].

[40] Enrique Crespo, «Bus I2C/TWI | Aprendiendo Arduino». [En línea]. Disponible en: https://aprendiendoarduino.wordpress.com/2016/11/14/bus-i2ctwi/. [Accedido: 19-feb-2018].

[41] Enrique Crespo, «Tema 6 – Comunicaciones con Arduino (4) | Aprendiendo Arduino». [En línea]. Disponible en: https://aprendiendoarduino.wordpress.com/2014/11/18/tema-6-comunicaciones-con-arduino-4/. [Accedido: 19-feb-2018].

[42] 'Elecrow Technology', «SIM808 GPRS/GSM+GPS Shield v1.1 - Elecrow». [En línea]. Disponible en: https://www.elecrow.com/wiki/index.php?title=SIM808_GPRS/GSM%2BGPS_Shield_v1.1. [Accedido: 20-feb-2018].

[43] DX.com, «SIM800L IPEX quad-banda GPRS GSM módulo de ruptura + antena 3G - sin Gastos de Envío - DealExtreme». [En línea]. Disponible en: http://www.dx.com/es/p/sim800l-quad-band-network-gprs-gsm-breakout-module-3g-3dbi-pcb-antenna-with-ipex-interface-406905?tc=EUR&gclid=Cj0KCQjw3MPNBRDjARIsAOYU6x8zTjYMRRw4yG-u8TDoccdczFUqc3U8UERxBiNWHeJW3NWJ0gHNIjUaArDWEALw_wcB#.Woye36jOWCj. [Accedido: 20-feb-2018].

[44] A. Durán Rocha, «Bluetooth HC-06 y HC-05 Android - Arduino TUTORIALES HETPRO», 2015. [En línea]. Disponible en: https://hetpro-store.com/TUTORIALES/bluetooth-hc-06-app-arduino/. [Accedido: 20-feb-2018].

[45] 'Digi-Key Electronics', «113030009 Seeed Technology Co., Ltd | Programadores, sistemas de desarrollo | DigiKey». [En línea]. Disponible en: https://www.digikey.es/products/es?keywords=sim900. [Accedido: 20-feb-2018].

[46] 'Digi-Key Electronics', «2636 Adafruit Industries LLC | Programadores, sistemas de desarrollo | DigiKey». [En línea]. Disponible en: https://www.digikey.es/product-detail/es/adafruit-industries-llc/2636/1528-1707-ND/5761269. [Accedido: 20-feb-2018].

[47] 'Digi-Key Electronics', «2691 Adafruit Industries LLC | RF/IF y RFID | DigiKey». [En línea]. Disponible en: https://www.digikey.es/product-detail/es/adafruit-industries-llc/2691/1528-1749-ND/5761268?WT.srch=1&mkwid=s&pcrid=218510851654&pkw=&pmt=&pdv=c&gclid=Cj0KCQjw3MPNBRDjARIsAOYU6x_WG9nbSpPdM4ZQXWWo7bkHcgpUJMIl_RZ2mQZYvs8Ry-DHvImjtgAaAuEgEALw_wcB. [Accedido: 20-feb-2018].

[48] 'DFRobot ©', «SIM808 GPS/GPRS/GSM Shield SKU: TEL0097 - DFRobot Electronic Product Wiki and Tutorial: Arduino and Robot Wiki-DFRobot.com». [En línea]. Disponible en:

https://www.dfrobot.com/wiki/index.php/SIM808_GPS/GPRS/GSM_Shield_SKU:_TEL0097. [Accedido: 20-feb-2018].

[49] 'DFRobot ©', «SIM808 GPS/GPRS/GSM Shield SKU: TEL0097 - DFRobot Electronic Product Wiki and Tutorial: Arduino and Robot Wiki-DFRobot.com». [En línea]. Disponible en: https://www.dfrobot.com/wiki/index.php/SIM808_GPS/GPRS/GSM_Shield_SKU:_TEL0097#Connect_TCP_and_Send_GET_Requests. [Accedido: 20-feb-2018].

[50] E. y de M. (Universidad de N. Departamento de Ingeniería Mecánica, «Normativa sobre Vibraciones».

[51] Departamento de Ingeniería Electrónica, «Descripcion general del medio Maylas».

[52] L. del Valle Hernández, «Sensor de temperatura, escoge el mejor para tus proyectos con Arduino». [En línea]. Disponible en: https://programarfacil.com/podcast/82-escoger-mejor-sensor-temperatura-arduino/. [Accedido: 20-feb-2018].

[53] 'Cuantex', «GSM GPRS Shield SIM900 Arduino UNO - Tutorial». [En línea]. Disponible en: https://www.cuantex.com/2016/04/01/gsm-sim900-shield/. [Accedido: 20-feb-2018].

[54] E. Crespo, «Shields para Arduino | Aprendiendo Arduino», 2017. [En línea]. Disponible en: https://aprendiendoarduino.wordpress.com/2015/03/23/shields-para-arduino/. [Accedido: 20-feb-2018].

[55] A. Castro Domínguez, «PROYECTO FIN DE CARRERA».

[56] blogElectronica.com, «Las comunicaciones GPRS | blogElectronica.com». [En línea]. Disponible en: http://www.blogelectronica.com/que-es-la-tenologia-gprs/. [Accedido: 20-feb-2018].

[57] Arturo Vargas Mercado, «Análisis de Vibración para el Mantenimiento Predictivo de Maquinaria».

[58] Alasdair Allan, «Arduino Uno vs BeagleBone vs Raspberry Pi», 2013. [En línea]. Disponible en: https://makezine.com/2013/04/15/arduino-uno-vs-beaglebone-vs-raspberry-pi/. [Accedido: 20-feb-2018].

[59] Adafruir(R), «Adafruit Learning System». [En línea]. Disponible en: https://learn.adafruit.com/category/adafruit-io. [Accedido: 20-feb-2018].

[60] '© Prometec', «Tutoriales Arduino | Prometec». [En línea]. Disponible en: https://www.prometec.net/. [Accedido: 20-feb-2018].

[61] '© Amidata S.A. ', «RS Components | Componentes Electrónicos y Eléctricos». [En línea]. Disponible en: https://es.rs-online.com/web/. [Accedido: 20-feb-2018].

FUENTE DE FIGURAS

[1] Fuente: http://solomantenimiento.blogspot.com.es/2015/03/que-es-el-mantenimiento-reactivo.html

2 Fuente: “Vibraciones en máquinas. Mantenimiento predictivo”. Departamento de Ingeniería Mecánica, Energética y de Materiales, Universidad de Navarra.

3 Fuente: “Vibraciones en máquinas. Mantenimiento predictivo”. Departamento de Ingeniería Mecánica, Energética y de Materiales, Universidad de Navarra.

4 Fuente: “Vibraciones en máquinas. Mantenimiento predictivo”. Departamento de Ingeniería Mecánica, Energética y de Materiales, Universidad de Navarra.

5 Fuente: “Vibraciones en máquinas. Mantenimiento predictivo”. Departamento de Ingeniería Mecánica, Energética y de Materiales, Universidad de Navarra.

6 Fuente: “Descripción completa del motor”, Tecnología Electrónica (GITI)

7 Fuente: “Descripción completa del motor”, Tecnología Electrónica (GITI)

8 Fuente: “Descripción completa del motor”, Tecnología Electrónica (GITI)

9 Fuente: “Descripción completa del motor”, Tecnología Electrónica (GITI)

10 Fuente: “Descripción completa del motor”, Tecnología Electrónica (GITI)

11 Fuente: “Descripción completa del motor”, Tecnología Electrónica (GITI)

12 Fuente: “Rise of the Embedded Internet”, White Paper Intel®, Embedded Processors, 2009.

13 Fuente: https://www.luisllamas.es/arduino-i2c/

14 Fuente: https://www.luisllamas.es/arduino-i2c/

15 Fuente: https://www.luisllamas.es/arduino-spi/

[16] https://aprendiendoarduino.wordpress.com/tag/esquemas-electricos/