PROYECTO FIN DE GRADO

TÍTULO: IMPLEMENTACIÓN Y ANÁLISIS DE UN DISPOSITIVO DE POSICIONAMIENTO INERCIAL SOBRE RASPBERRY PI

AUTOR/A: FUENTETAJA DE PRADO, PABLO

TITULACIÓN: INGENIERÍA ELECTRÓNICA DE COMUNICACIONES

DIRECTOR/A: GONZÁLEZ CRESPO, AMADOR MIGUEL

TUTOR/A: RUIZ GONZÁLEZ, MARIANO

DEPARTAMENTO: DEPARTAMENTE DE INGENIERÍA TELEMÁTICA Y ELECTRÓNICA (DTE)

VºBº TUTOR/A

Miembros del Tribunal Calificador:

PRESIDENTE/A:

TUTOR/A:

SECRETARIO/A:

Fecha de lectura:

Calificación:

El Secretario/La Secretaria,

RESUMEN

IMPLEMENTACIÓN Y ANÁLISIS DE UN DISPOSITIVO DE MEDIDA INERCIAL SOBRE *RASPBERRY PI*.

El propósito de este proyecto es el diseño y desarrollo completo de un sistema empotrado con la funcionalidad de unidad de medida inercial. El microcontrolador usado para este fin es una *Raspberry Pi 3B+*, cargada con sistema operativo embebido *Linux*.

El resto de los elementos del diseño, se seleccionan con el condicionante de no ser directamente compatibles con *Raspberry Pi,* de modo que sea necesario adaptar el *hardware* para posibilitar el acople de todos los periféricos, y desarrollar *software* propio para su gestión y control.

Adicionalmente, se establece que el *software* de la aplicación se realice en lenguaje de programación C, usando librerías de alto nivel. De esta forma, se busca analizar el desempeño de los pines de entradas salidas cuando se gestionan a este nivel, siendo éste el segundo propósito del presente proyecto.

Los dispositivos de medida inercial se caracterizan por generar información acerca del posicionamiento de un objeto, como puede ser su orientación, su velocidad o su inclinación. Estos dispositivos se constituyen normalmente por un sensor acelerómetro y por un giróscopo, aunque en este proyecto se ha decidido completarlo añadiendo un magnetómetro.

Tras un minucioso proceso de análisis de las diferentes opciones del mercado, se decide incorporar el integrado *BMI160*, formado por un acelerómetro y un giróscopo, y el magnetómetro *BMM150*. Entre los motivos de la elección de estos componentes está la posibilidad de acoplar el magnetómetro al integrado *BMI160*, de modo que se trabaja como si se tratara de una única unidad.

Para mostrar los datos extraídos de los sensores, se incorpora al diseño una pequeña pantalla táctil de tamaño 240x320. De este modo también se posibilita que el usuario interactúe con el dispositivo, seleccionando el sensor del que se quiere leer los datos o configurándolos a su gusto. A parte de las medidas leídas de los sensores, también se muestran los ángulos de inclinación de los ejes y se implementa una brújula electrónica.

El proyecto se desarrolla de forma gradual, es decir, se trabaja con cada elemento por separado, creando las librerías particulares, para terminar integrándolo todo en el diseño completo. Se comienza trabajando con el bloque de los sensores. En este bloque es necesario implementar el protocolo de comunicaciones *I2C* para posibilitar la configuración de los 3 sensores y la lectura de las medidas tomadas, para su posterior procesado.

El siguiente bloque para tratar es la pantalla táctil, la cual se divide en su apartado táctil y su apartado visual. El primero, hace uso del mismo canal *I2C* de los sensores para comunicar a la *Raspberry Pi* las coordenadas del toque en la pantalla. El apartado visual necesita de dos protocolos distintos, uno de tipo serie para el control y configuración, y otro para la transmisión de las imágenes. Ambos protocolos se codifican completamente ya que no existen librerías asociadas, debido a la restricción de no estar preparados para su montaje con *Raspberry Pi*.

El último paso es la integración de ambos elementos con la *Raspberry Pi,* mediante una máquina de estados. Es también necesario el diseño de una placa que permita el acople de los elementos en un único dispositivo, implementando las distintas interconexiones y el reparto de alimentaciones.

Tras analizar los resultados obtenidos se llega la conclusión de que el diseño alcanza los propósitos marcados. Se verifica que los sensores se calibran correctamente y que las medidas realizadas son de calidad. Se comprueba que la pantalla muestras las imágenes correctamente y reacciona como es de esperar a las interacciones con la capa táctil.

Sin embargo, el análisis del desempeño de los pines de entrada salida, revela la necesidad de limitar la funcionalidad del diseño para poder ser soportado. Esto se debe al protocolo de la pantalla táctil, que exige una frecuencia de señal excesiva. Por este motivo, se toma la decisión de reducir el valor de la frecuencia de los 6,4 MHz que marca el fabricante a 1,7 MHz, y el número de señales de datos de 24 a 1, lo que implica que únicamente se representarán imágenes en blanco y negro. Para llegar a determinar estos valores se realiza un detallado análisis de los diferentes retardos implicados en el ciclo de la aplicación, llegando a la conclusión de que la gestión a alto nivel de los pines de entrada-salida queda limitado para frecuencias inferiores a 2 MHz.

ABSTRACT

IMPLEMENTATION AND ANALYSIS OF AN INERTIAL MEASUREMENT DEVICE OVER RASPBERRY PI.

The purpose of this project is the design and complete development of an embed system functioning as an inertial measurement unit. The microcontroller used is a *Raspberry Pi 3B+,* loaded with *Linux* Operating System.

The rest of the elements of the design are selected with the condition of not being directly compatible with *Raspberry Pi*, so that it is necessary to adapt the hardware in order to make it possible to couple all the peripherals, and develop own software for its control and management.

Additionally, it is stablished that the application software is going to be implemented in C programming language, using high level libraries. This way, the performance of the general input-output pins is going to be analyzed when they are managed using this high level language, which is the second purpose of this project

The inertial measurement devices are characterized for generating information about an object positioning, such as its orientation, velocity or inclination. These devices are usually constituted by an accelerometer and gyroscope sensor, even though in this project it is completed by adding a magnetometer to the design.

After a thorough process of analyzing the different market options, it is selected the integrated circuit BMI*160*, conformed by an accelerometer and a gyroscope, and the magnetometer *BMI150*. One of the reasons for choosing these two components is the possibility of connecting the magnetometer to the *BMI160.* This way it is possible to work as if there was only one integrated circuit.

In order to display the data extracted from the sensors, a small 240x320 touchscreen is incorporated to the design. This way the user is allowed to interact with the device, selecting the sensor which data is going to be read or configuring the sensor as pleased. Additionally to the measurements read from the sensors, the tilt angles of the axis are also displayed and an electronic compass is implemented.

The project is developed gradually, meaning that each element is managed separately, creating specialized libraries, to finally being able of integrating everything in the complete design. The sensors block is the first one created. In this block it is necessary to implement the I2C communication protocol to enable the configuration of the 3 sensors and the reading of the measurements done, for its processing.

The next block created is the touchscreen, which divided in its touchscreen section and its visual display section. The first one, uses the same *I2C* channel as the sensors to communicate the *Raspberry Pi* the coordinates of a touch in the screen. The visual section needs two different protocols, one serial protocol for the control and configuration of the display and another one for the transmission of the images. Both of them are completely codified due to the inexistence of libraries, since it was established the condition of not being prepared for its coupling with Raspberry Pi.

The last step is the integration of these two elements with the Raspberry Pi, using a state machine. It is also necessary to design a board for the mounting of these elements into one single device, implementing the different interfaces and the distribution of the different power levels.

After analyzing the obtained results, it is concluded that the design reaches the marked purposes. It is verified that the sensors are calibrated correctly and that the obtained measurements are of good quality. It is checked that the images shown in the display are exactly as they should and that the display reacts as expected to the interaction with the touch layer.

However, the analysis of the performance of the input-output pins reveals the need of limiting the design's functionality in order to be supported. This is due to the touchscreen protocol, which demands and excessive signal frequency. For this reason, the value of this frequency is reduced from 6.4 MHz that indicates the manufacturer to 1.7 MHz, and the number of data signal from 24 to 1, meaning that only black and white images can be displayed. For determining these values a detailed study of the different delays involved in the application cycle is made, reaching the conclusion that the high level management of the input-output pins is limited to frequencies lower than 2 MHz.

ÍNDICE

LISTA DE ACRÓNIMOS

ACRÓNIMO	*SIGNIFICADO*
GPIO	*General Purpose Input Output*
MEMS	*MicroElectroMechanical Systems*
UART	*Universal Asynchronous Receiver / Transmitter*
EEPROM	*Electrically Erasable Programmable Read-Only Memory*
I2C	*Inter-integrated Circuit*
ISR	*Interrupt Service Routine*
ODS	Objetivos de Desarrollo Sostenible
IMU	*Inertial Movement Unit*
SPI	*Serial Peripheral Interface*
ADC	*Analog to Digital Converter*
ODR	*Output Data Rate*
LCD	*Liquid Crystal Display*
TFT	*Thin Film Transistor*
FFC	*Flat Flexible Cable*
ACK	*Acknowledge*
API	*Application Programming Interface*
C.D.	Código decimal
SSH	*Secure Shell*
BIOS	*Basic Input Output System*
PWM	*Pulse Width Modulation*

1 INTRODUCCIÓN

El objetivo de este proyecto es el diseño e implementación de un sistema empotrado sobre *Raspberry Pi*, con el fin de realizar las funciones de una unidad de medida inercial (*IMU*, por sus siglas en inglés). Igualmente, se analizará el desempeño de los pines de entrada y salida de propósito general (*GPIO*, por sus siglas en inglés) determinando sus limitaciones frente a aplicaciones que requieran un rendimiento exigente.

Se comienza con un primer apartado en el que se detalla el marco tecnológico del proyecto. Aquí se explica en qué consiste una *IMU* y sus aplicaciones, así como los sensores que la conforman: acelerómetro, giróscopo y magnetómetro. Tras dar una pequeña introducción sobre su funcionamiento, se introduce el siguiente elemento del diseño, la pantalla táctil LCD *LCD* (*Liquid Crystal Display,* en sus siglas en inglés). Para finalizar este apartado, se indican las configuración y aspectos relevantes de la *Raspberry Pi*.

A continuación, y tras un apartado de especificaciones técnicas que limitarán el diseño, comienza el apartado con la descripción del diseño. Primero se describe el diseño *hardware* para después describir el diseño *software*. En este segundo apartado del diseño, más extenso que el anterior, se describe el funcionamiento general de la aplicación y su implementación, detallando aspectos como los protocolos implementados, el control de los periféricos o la integración de todos los elementos.

Después, se analizan los resultados obtenidos tras la realización de numerosas pruebas que abarcan todos los aspectos de relevantes del diseño, aportando todas las evidencias necesarias.

En el siguiente apartado, en base a los resultados de expuestos en el apartado anterior, se concluye si el diseño cumple los objetivos marcados en su inicio. También se da una visión sobre mejoras que podrían ser de aplicación en futuros diseños.

Se continúa con un apartado dedicado al impacto del proyecto, exponiendo sus implicaciones en distintos aspectos de nuestro entorno y su posicionamiento dentro de los Objetivos de Desarrollo Sostenible (*ODS*).

Para finalizar, se incluye un anexo en el que se encuentra el desglose presupuestario del proyecto, las herramientas usadas durante el desarrollo del mismo, y un manual de usuario, para facilitar la correcta inicialización y manipulación por cualquier persona ajeno al proyecto.

2 MARCO TECNOLÓGICO

Las unidades de medida inercial se caracterizan por la capacidad de medir su entorno para ofrecer información acerca del posicionamiento del dispositivo al que van acoplados, es decir, la orientación del dispositivo, el ángulo en cada uno de sus tres ejes o la velocidad de su trayectoria, entre otras.

Estos dispositivos se encuentran en infinidad de aplicaciones de distinto ámbito. Por ejemplo, son pieza fundamental de los sistemas de navegación, siendo especialmente necesarios en el ámbito aeronáutico, donde no hay un punto de referencia exterior. Las *IMUs* son usadas para monitorizar aspectos del rumbo de la aeronave como son la inclinación, posición o trayectoria. Al ser capaces de detectar variaciones mínimas e instantáneas, si se detecta algún desvío respecto al dato esperado se corrige inmediatamente, posibilitando un rumbo seguro y controlado.

Fuera del ámbito de la navegación, estos dispositivos se usan más comúnmente en lo teléfonos móviles o los relojes y pulseras inteligentes. La posición del móvil (vertical, horizontal, tumbado, etc…), la cantidad de pasos realizados, la estabilización de imágenes o el desplazamiento en interiores son algunas de las muchas aplicaciones que puede ofrecer.

Para todo esto, los dispositivos de medida inercial cuentan con tres sensores diferentes. Estos serían: un acelerómetro, un giroscopio y un magnetómetro. Aunque en ocasiones se puede prescindir de alguno de ellos.

En función de la forma de obtener y transformar la información del medio para ofrecer los datos pertinentes, se pueden clasificar en diferentes tipos de sensores. Los más comunes, son los de tipo *MEMS* (*MicroElectroMechanical Systems,* en sus siglas en inglés). Éstos se caracterizan por transformar las variaciones mecánicas sufridas en su diseño, en datos discretos fácilmente interpretables, y por poder trabajar bajo un rango amplio de temperaturas, así como su bajo coste y tamaño. Tanto el acelerómetro como el giróscopo de este diseño pertenecen a este tipo de sensores.

Los acelerómetros son sensores que proporcionan información sobre la aceleración lineal experimentada en uno o más de sus ejes.

La estructura básica del sensor consiste en dos electrodos fijos y alineados con el eje del sensor entre los cuales se desplaza una masa conductora, como se observa en la *Figura 1*. Tomando esta placa intermedia como la masa común y haciendo pasar una corriente eléctrica por cada uno de los dos electrodos fijos, se tiene un sistema formado por dos condensadores con capacidad variable.

Al aplicar una fuerza sobre el eje, la masa intermedia se desplaza hacia uno de los electrodos, disminuyendo así la distancia al mismo y aumentando la distancia al otro electrodo. La variación de la distancia entre electrodos de un condensador provoca una variación del dieléctrico, de manera que aumentará su valor si aumenta la separación y viceversa. Un aumento en el dieléctrico implica un aumento en la capacidad del condensador.

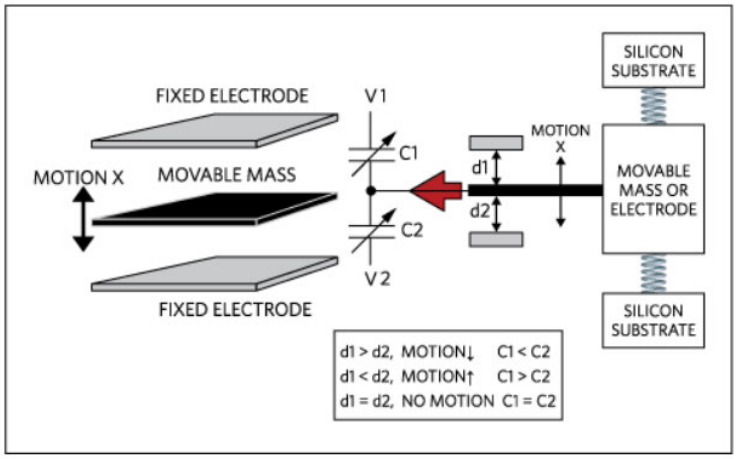

Figura 1: Mecanismo acelerómetro de tipo MEMS (Fuente: www.analog.com)

De esta manera, se mide la capacitancia de ambos condensadores y se comparan para poder así determinar el sentido de la aceleración sufrida. Mediante un circuito amplificador y un convertidor analógico digital se traduce la diferencia de tensión en los capacitores a un valor discreto con el que ya se podrá trabajar para determinar la aceleración sufrida en el eje. Este mecanismo se reproduce las veces necesarias hasta tener suficientes medidas simultáneas y así obtener mediciones precisas de las variaciones de potencial. Por último, dependiendo de la completitud del sensor, este mecanismo se implementa en uno, dos o tres de sus ejes.

Debido a este diseño, el sensor detecta variaciones en la aceleración al cambiar la inclinación del mismo. Sin embargo, se comportará de manera indiferente frente a rotaciones del eje de medida ya que las distancias d1 y d2 no variarán. Para esto es necesario el giroscopio.

El giroscopio, también denominado giróscopo, proporciona información acerca de la velocidad angular en cada uno de sus ejes. La forma de obtener dicho dato es muy similar a la descrita para el acelerómetro. Su estructura también la componen dos laminas conductoras fijas y una móvil entre ellas. La diferencia radica en que en este caso el movimiento de la lámina intermedia se debe a la fuerza producida por el efecto Coriolis. Este efecto hace referencia a la aparente fuerza que experimenta los cuerpos dentro de un sistema con velocidad angular, que tiende a alejarlos del centro de rotación.

De esta manera, al rotar el sensor sobre uno de sus ejes, la distancia entre electrodos variará y será posible usar las capacidades para determinar la velocidad angular, mediante un circuito similar al del acelerómetro.

Debido a este diseño, mientras el dispositivo esté en reposo, las lecturas del giróscopo serán nulas. Esto implica que será posible determinar la variación instantánea respecto a una posición dada, pero no conocer la posición en sí misma. Es por este motivo que se suele trabajar conjuntamente con ambos sensores. Mientras que el acelerómetro determina la posición inmediata, el giróscopo detecta la mínima variación respecto a esta posición. De esta manera, es posible disponer de un conocimiento absoluto del posicionamiento de un objeto, y todas las variables que derivan de este dato (velocidad, aceleración, trayectoria, etc).

El magnetómetro proporciona información acerca de la intensidad del campo magnético en cada uno de los tres ejes.

El magnetómetro usado en este diseño es un sensor de tipo efecto Hall. Esta denominación hace referencia a la desviación que sufre la dirección del flujo de una corriente eléctrica a través de un elemento conductor cuando se encuentra dentro de un campo magnético con dirección perpendicular al flujo, descubierto por el científico homónimo Edwin Hall.

De esta forma, si se hace circular corriente por un elemento conductor ésta lo hará describiendo una línea recta, si no existen perturbaciones magnéticas externas. Al situar dicho elemento conductor dentro de un flujo magnético que lo atraviese en sentido perpendicular, la carga se verá desplazada de la trayectoria lineal, creando así una diferencia de potencial en sus extremos, como se observa en la *Figura 2*. Esta desviación respecto a la trayectoria lineal será mayor cuanto mayor sea la intensidad del campo magnético, obteniéndose así una relación de proporcionalidad entre este último y la diferencia de potencial.

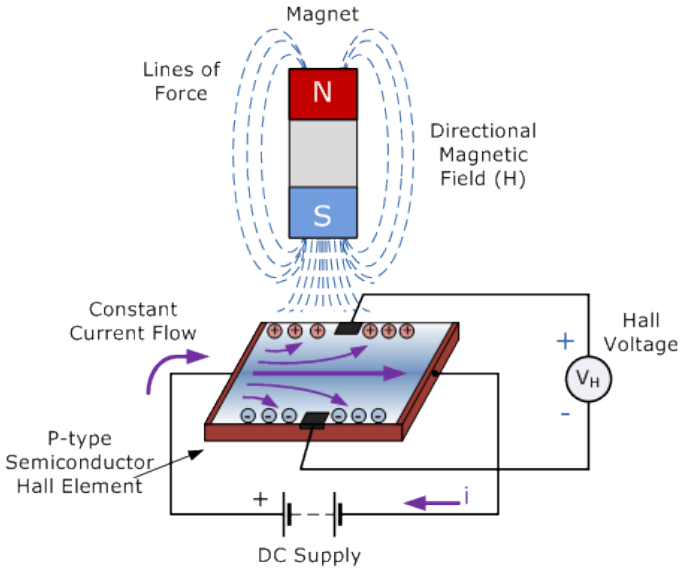

Figura 2: Representación efecto Hall (Fuente: www.ti.com)

Debido a que esta tensión es del orden de µV, es necesario un sistema amplificador previo al conversor analógico digital que proporcionará la medida discreta final.

Estos sensores se usan ampliamente, especialmente en la industria automovilística, donde se implementan para la detección de proximidad, el control de la velocidad de rotación de un eje o la medición de los niveles de los distintos fluidos del vehículo.

La principal desventaja de estos sensores son las elevadas temperaturas que alcanza el material semiconductor debido a la naturaleza del diseño, lo que provoca que aparezca un elemento resistivo que introduce un error en la medida, como se detalla en el apartado 4.2.2.

Para la documentación acerca del funcionamiento de estos sensores, así como para las figuras, se han consultado las referencias [1] y [2].

Para el control de estos sensores y el procesado de las medidas se ha incluido al diseño una *Raspberry Pi,* modelo 3 B+ [3]. Este producto de la compañía *Raspberry Pi Foundation*, nació como una herramienta educativa para usar en institutos y universidades. Sin embargo, rápidamente alcanzó popularidad y ahora es usada extensamente en distintos ámbitos tecnológicos.

Esto se debe a su bajo coste y su gran versatilidad para distintos proyectos, ya que en definitiva se trata de un ordenador de tamaño reducido. Muchos de sus modelos incluyen periféricos y distintos puertos para ampliar más aún su posible uso. Adicionalmente, todos los modelos comparten un conector de 40 pines para la integración de elementos y señales externas.

Una de sus características más valoradas es la posibilidad de cargarla con un sistema operativo. En este caso se ha hecho uso de la herramienta *buildroot* [4] para montar un sistema embebido de tipo Linux. Debido a esto, es necesario trabajar sobre un equipo con sistema operativo Linux, o en su defecto, una máquina virtual. Esta última opción ha sido la llevada a cabo en este proyecto.

La construcción del sistema operativo embebido se ha llevado a cabo de acuerdo a los estándares establecidos en el documento de referencia [5]. Entre las características más relevantes de su configuración están el acceso remoto a la *Raspberry Pi* a través de una red inalámbrica para el volcado de archivos y la instalación de controladores para cuestiones como la implementación del protocolo *I2C* (*Inter-integrated Circuit,* por sus siglas en inglés) o el control de pines de entrada y salida. También se ha instalado el paquete de la librería de software de alto nivel *pigpio* [6], sobre la que se ha desarrollado el código.

Para permitir la comunicación entre la máquina virtual y la *Raspberry Pi* se ha implementado un puerto serie. Para el uso de comandos del terminal de Linux, así como para las consultas general en el manejo y trabajo con el sistema operativo Linux, se han consultado extensamente las referencias [7] y [8].

El último elemento con el que cuenta el diseño es una pequeña pantalla táctil (también denominado *display*), con el fin de mostrar los datos de los sensores y permitir la interacción hombre máquina. Esta pantalla es de tipo TFT (*Thin Film Transistor*, en sus siglas en inglés), también llamado de matriz activa. Esto indica que la polarización de los puntos de color de la pantalla no se realiza directamente, si no a través de transistores, consiguiendo así una mayor velocidad de refresco, a la vez que reducir notablemente el consumo, lo que la hace idónea para este diseño.

La capa táctil de la pantalla es de tipo capacitivo. Este tipo de tecnología consta de una matriz de tantas filas y columnas como alto y ancho tenga la pantalla, capaces de determinar una variación de tensión en un punto concreto de la matriz y generar unas coordenadas para el toque. Las pantallas de tipo capacitivo se caracterizan por tener mayor nitidez de imagen y detectar toques simultáneos, entre otras características. También tienen mayor sensibilidad en la detección, sin necesidad de ejercer presión como en las pantallas de tipo resistivo, siendo suficiente la electricidad estática del dedo humano para provocar la diferencia de tensión detectable por la matriz.

3 ESPECIFICACIONES TÉCNICAS

General*:*

- Alimentación: 5V
- Rango temperatura de funcionamiento: 0 a 70 ºC
- Dimensiones: 10,5 cm x 6,5 cm x 4,5 cm

Raspberry Pi

- Modelo: 3B+
- Alimentación: 5V
- Rango de temperatura: 0, 70 ºC
- Sistema operativo embebido tipo Unix
- Acceso remoto tipo *ssh (115200 Bd)*

Sensores

- Modelo: Placa "*BOOSTER SENSORS"* (*Texas Instruments)* [9]. Incluye los siguientes integrados:
    - IMU BMI160 (*Bosch),* con acelerómetro y giróscopo.
        - Alimentación: 3,3V
        - Rango de temperatura de operación: -40ºC a +85ºC
        - Frecuencia máxima de muestreo acelerómetro: 1600 Hz
        - Rango acelerómetro: ±2g, ±4g, ±8g, ±16g
        - Frecuencia máxima de muestro giróscopo: 3200 Hz
        - Rango de giróscopo: ±125º/s, ±250º/s, ±500º/s, ±1000º/s, ±2000º/s
    - Magnetómetro BMM150 (*Bosch).*
        - Alimentación: 3,3 V
        - Rango de temperatura de operación: -40ºC a +85ºC
        - Frecuencia de muestreo: 100 Hz
        - Rango campo magnético: ±1300µT (eje x/y), ±2500µT (eje z)

Pantalla LCD

- Modelo: DT035BTFT-PTS1 (*Displaytech*) [10]
- Tamaño: 320x240
- Alimentación: 3,3V
- Alimentación luz trasera: 19 V
- *Driver display*: NV3035
- *Driver* pantalla táctil: FT5346
- Rango de temperatura de operación: -20 a +70ºC

4 DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA

4.1 DISEÑO HARDWARE

Debido a que ni la pantalla táctil ni la placa de sensores son directamente compatibles con *Raspberry Pi*, se diseña una placa intermedia para permitir la integración de estos tres elementos. Además de facilitar el ensamblado del dispositivo, esta placa también implementa las interfaces que interconectan los tres elementos, y genera y distribuye los distintos niveles de tensión a cada elemento del dispositivo. Para ello se monta un pequeño convertidor DC/DC elevador que proporciona el nivel de tensión adecuado a la matriz trasera de leds del *display* (con valor de 19 V), a partir de los 5 V de alimentación, previo ajuste del potenciómetro que incorpora.

El modo en que se interconectan el *display* y los sensores con la *Raspberry Pi* se muestra en el diagrama de bloques de la *Figura 3,* donde se reflejan las interfaces más relevantes del diseño en negro y las distintas alimentaciones en rojo.

Para facilitar la comunicación con un equipo exterior, se implementa también un pequeño puerto serie *UART (Universal Asynchronous Receiver/Transmitter,* por sus siglas en inglés).

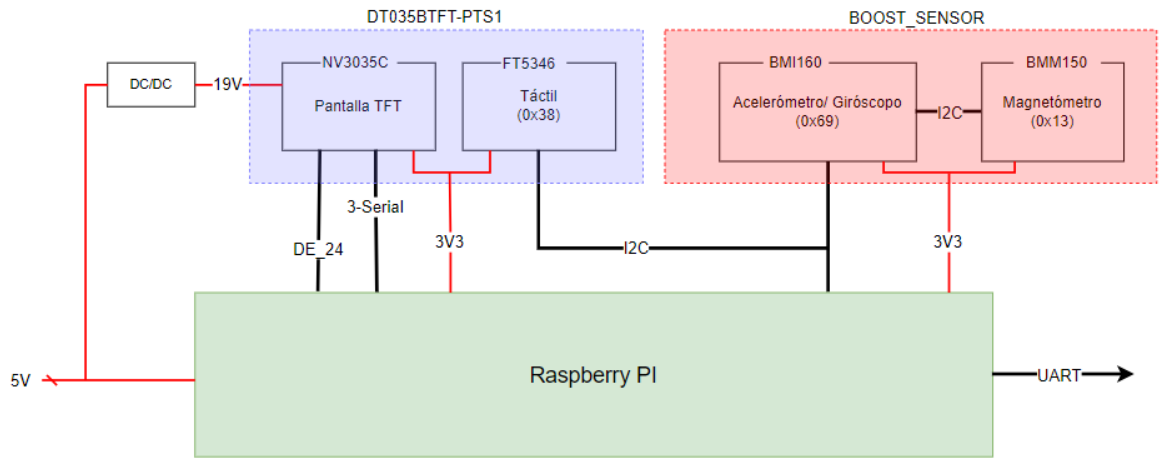

Figura 3: Diagrama arquitectura hardware del dispositivo.

Cabe señalar que la comunicación con los sensores se realiza únicamente a través del sensor *BMI160*, ya que éste se puede configurar para manejar el sensor *BMM150* de manera autónoma, almacenando los datos en sus propios registros. Así es posible tratar ambos integrados como si se tratara de un único integrado.

Por otro lado, debido a que la pantalla usada dispone de dos conectores tipo *FFC* (*Flexible Flat Cable*, por sus siglas en inglés), uno para el manejo del *display* y otro para el manejo de la pantalla táctil, es necesario un adaptador que aúne las señales de estos dos conectores en un único conector de tipo pines de inserción, para su ensamblado en la placa auxiliar. Para tal fin se ha hecho uso del adaptador comercial mostrado en la *Figura 35*.

Por último, para posibilitar la depuración del dispositivo, se han derivado las señales relevantes para estudio hacia los conectores CM1 y CM2 (ver Figura 33 y Figura 34), ocupando pines libres de la placa de sensores.

4.2 DISEÑO SOFTWARE

En sintonía con el diseño hardware y el diagrama de arquitectura de la *Figura 3*, el dispositivo se divide en cuatro bloques diferenciados, dedicados a un aspecto específico de su funcionalidad. El bloque del *display* controla y actualiza la información que se muestra en la pantalla; el bloque sensores configura y recoge información del acelerómetro, magnetómetro y giroscopio, para su posterior procesamiento; el bloque táctil permite la interactuación hombre-máquina, detectando los gestos y toques en la pantalla táctil y desencadenando la acción pertinente; por último, el bloque de control gestiona el funcionamiento general del dispositivo, comandando al resto de bloques para permitir su coordinación e interactuación.

Para la implementación software de estos bloques, se desarrolla una aplicación multihilo en lenguaje de programación C, conformada por un total de cinco hilos. El hilo principal, tras inicializar el resto de los hilos, arranca una máquina de estados, la cual mediante el recibo y envío de señales controla al resto de hilos y así la funcionalidad global del dispositivo. El primero de estos hilos secundarios, está dedicado exclusivamente al proceso de envío de imágenes a la pantalla. Un segundo hilo gestiona el contenido de estas imágenes, alternando entre entornos y actualizando los valores de los sensores a una frecuencia constante (independiente de la frecuencia variable de toma de datos) acorde a lo que comanda el hilo principal. El tercer hilo se dedica al control de la pantalla táctil, verificando si las coordenadas de toque que se reciben se corresponden con algún botón táctil, y en caso afirmativo, se envía una señal al hilo principal para que éste desencadene la acción pertinente. El último hilo, controla los tres sensores, configurándolos adecuadamente y extrayendo las medidas que toman para su posterior procesado.

El proyecto se estructura de manera modular, siendo constituido por un módulo principal y cinco módulos auxiliares, donde se recogen funciones y variables que conforman librerías dedicadas.

En los módulos *bmi160* y *bmm150* se recogen las funciones para el control de ambos integrados, de acuerdo a las indicaciones de sus hojas de características. Estas librerías son las que suministra *Bosch,* con algún cambio menor.

En el módulo *I2C* se recogen las funciones que conforman el protocolo de comunicaciones con el controlador táctil de la pantalla y los sensores.

El módulo *lcd* recoge las funciones relativas a la pantalla táctil. En él se codifican los dos protocolos necesarios: el protocolo de control y configuración de la pantalla, y el protocolo de envío de imágenes. En este módulo también se encuentran funciones para la manipulación de las imágenes, tales como escritura de caracteres, inversión de píxeles o notificaciones visuales. Para esto el módulo cuenta con el fichero de definiciones *nums_font* en el que traducen los números naturales y los caracteres usados a un mapa de píxeles para su representación visual. Un segundo fichero de definiciones, denominado *coords,* almacena las coordenadas de la

pantalla táctil que se corresponden con los botones con los que el usuario puede interactuar en cada pantalla.

Por último, el módulo *sensors* recoge las funciones necesarias para tratar los datos extraídos de los sensores.

4.2.1 BLOQUE CONTROL

El bloque de control cumple el objetivo de coordinar el funcionamiento global del dispositivo.

Esto se realiza mediante la implementación de la máquina de estados de tipo Moore, que se muestra en la *Figura 4*. Esto significa que las salidas únicamente dependen del estado en el que se encuentra y no de las entradas (como ocurriría en una máquina de estados de tipo Mealey). En este caso, las salidas del estado serían la habilitación o deshabilitación de los sensores, los datos mostrados en pantalla o la distribución de los puntos de interactuación con el usuario en la pantalla.

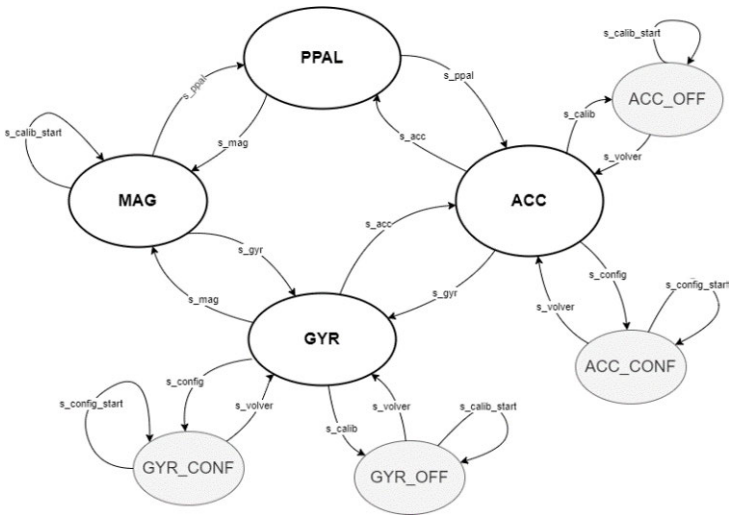

Figura 4: Mapa de estados del bloque de control

Es esta interactuación del usuario con la pantalla táctil la que produce, dentro del bloque táctil, las señales que desencadenan los cambios entre los distintos estados que se describen a continuación.

PPAL es el estado inicial, en el que se encuentra el dispositivo al arrancar la aplicación. En él se inicializan los periféricos y se realiza su configuración inicial. Una vez realizada la puesta a punto, el dispositivo se mantiene en un estado de reposo en la que los sensores están inactivos a la espera de recibir alguna señal válida. Desde este estado es posible avanzar al estado ACC o MAG.

Dentro del estado ACC, se trabaja únicamente con el acelerómetro, manteniendo los otros tres sensores deshabilitados. Por un lado, se le indica al bloque de sensores que debe obtener datos

de este sensor y procesarlos para, a través del bloque del *display*, mostrarlos en la pantalla. Desde este punto es posible avanzar al estado ACC_CONF o al estado ACC_OFF.

En el estado ACC_CONF se comanda al bloque de sensores para que modifique los parámetros de configuración del sensor y al bloque de *display* para que los actualice en la pantalla, indicando si la operación se ha realizado correctamente. Estos parámetros son la frecuencia de toma de datos, *ODR* (*Output Data Rate* por sus siglas en inglés), y el rango del acelerómetro. Es el bloque táctil el que transmite la configuración concreta que el usuario introduce.

En el estado ACC_OFF se comanda al bloque de sensores para realizar una calibración del sensor, indicándose por pantalla los valores de offset actualizados y el éxito de la operación.

Los estados GYR y MAG, son equivalentes al sensor ACC pero en relación a los sensores giroscopio y magnetómetro respectivamente. De igual manera, los estados GYR_CONF y GYR_OFF son equivalentes a los estados ACC_CONF y ACC_OFF, pero aplicados al giróscopo en este caso.

Es posible trabajar con el magnetómetro desde un único estado, sin necesidad de incluir uno de configuración y otro de calibración, debido a la particularidad de su disposición en el diseño y su manejo, como se detalla en el apartado del bloque de sensores 4.2.2.

4.2.2 BLOQUE SENSORES

El bloque de los sensores tiene dos funciones principales: implementar la interfaz de comunicación entre la *Raspberry Pi* y los sensores, e interpretar los datos obtenidos de estos.

Tanto el integrado BMI160 como el BMM150 permiten escoger entre los protocolos *I2C* o *SPI* (*Serial Protocol Interface*, por sus siglas en inglés) para la comunicación con los sensores, en función de la configuración en la que se encuentren ciertos puentes. Debido a que ambos integrados forman parte de la tarjeta comercial *BOOST_SENSORS*, la elección del protocolo queda definido por el diseño de la misma, que en este caso fija el protocolo de comunicaciones como *I2C*.

La interfaz de este protocolo consta de dos únicas señales. Por un lado, la señal de datos (SDA) es la vía por la que se transmite el mensaje o palabra y, por otro lado, la señal de reloj (SCL) sincroniza la transmisión de los datos entre maestro y esclavo.

Esta terminología, de uso común en protocolos de comunicación, hace referencia a los roles de los integrados implicados. El maestro es el que controla la comunicación, escribiendo y leyendo de los registros de los integrados pasivos o esclavos. En este diseño la *Raspberry Pi* ejerce de maestro y los sensores de esclavos.

Esto implica que es un protocolo multiesclavo, es decir, que en un mismo canal pueden coexistir distintos esclavos, cada uno de ellos con una dirección única que los identifica. El maestro al inicio de la transacción indica esa dirección para que únicamente el esclavo correspondiente responda.

La *Raspberry Pi* cuenta con dos parejas de pines configurables para formar dos interfaces *I2C*, una específica para memorias *EEPROM* (Electrically Erasable Programmable Read-Only Memory, por sus siglas en inglés) y otra de uso general que es la usada en este diseño.

Para poder implementar la interfaz *I2C* a través de estos pines es necesario, por un lado, configurarlos como las señales SDA y SCL, y, por otro lado, cargar el controlador *I2C*. Para lo primero, se debe añadir la orden en el fichero config.txt, como se muestra en la línea 30 de la *Figura 5*. Con esto los pines 2 y 3 quedan configurados a su segunda funcionalidad, SDA y SCL, y de esta manera no es necesario indicarlo por código.

Adicionalmente, se configura que la frecuencia de reloj *I2C* sea de 400 kHz ya que por defecto es de 100 kHz. Este valor es el máximo que soporta la *Raspberry Pi*, estando por debajo de la frecuencia máxima de 1 MHz que soportan los sensores.

```
25 gpu_mem_1024=100
26
27 # fixes rpi (3B, 3B+, 3A+, 4B and Zero W) ttyAMA0 serial console
28 dtoverlay=miniuart-bt
29
30 dtparam=i2c_arm=on,i2c_arm_baudrate=400000
```

Figura 5: Detalle fichero config.txt

El fichero config.txt hace las funciones de *BIOS* (*Basic Input Output System* por sus siglas en inglés) dentro del sistema operativo empotrado Linux en *Raspberry Pi* y en él se indican los módulos que se desea cargar, así como los aspectos configurables del procesador. Por eso motivo también se ha incluido la línea 28, con el propósito de activar el puerto serie y permitir la comunicación entre el ordenador y la *Raspberry Pi*.

Para cargar el módulo *I2C* es necesario hacer uso de la herramienta de instalación de paquetes de Linux *modprobe* incluyendo la orden dentro de un script en la carpeta *init.d*, donde se encuentran el resto de módulos que se cargarán al iniciar el núcleo del sistema operativo.

La necesidad de instalarlo en cada arranque del sistema se debe a que la compilación de *buildroot* almacena los módulos que se han seleccionado, pero no gestiona su instalación. Por eso es necesario instanciarlo manualmente.

Una vez arrancado el módulo, y configurados los pines, automáticamente se crea el fichero /dev/i2c-x, donde x es un número dinámico que identifica al canal *I2C* establecido. Al tratarse de un fichero de texto, la comunicación con los esclavos se convierte en meras operaciones de escritura y lectura.

Para la implementación del protocolo se ha elaborado una librería propia en base a las librerías de Linux *linux/i2c-dev.h* y *sys/ioctl.h.*

La librería se ha implementado de manera que la longitud de los datos a escribir o leer sea configurable (a diferencia de lo que ocurre con las librerías de Linux), posibilitando la lectura o escritura de múltiples registros en una sola transacción.

Una vez acondicionado el hardware e implementado el software *I2C* ya es posible trabajar con los sensores.

Al arrancar la aplicación, y tras abrir el manejador para el fichero i2c-0 del canal *I2C* implementado, se inicia la configuración de los sensores, con los valores más relevantes indicados en la *Tabla 1, Tabla 2 y Tabla 3*.

Tabla 1: Valores iniciales de configuración del acelerómetro.

Parámetro	Valor	Comentario
ODR	25 Hz	Frecuencia con la que se realizarán las mediciones.
Rango	±2 g	Rango de valores medibles.
Modo de operación	*NORMAL_MODE*	Todas las funciones del acelerómetro están activas, al contrario de los modos *SUSPEND_MODE* o *LOW_POWER_MODE* donde se ven restringidas.

Tabla 2: Valores iniciales de configuración del giróscopo.

Parámetro	Valor	Descripción
ODR	50 Hz	Frecuencia con la que se realizarán las mediciones.
Rango	±500 °/s	Rango de valores medibles.
Modo de operación	*NORMAL_MODE*	Todas las funciones del giróscopo están activas, al contrario de los modos *SUSPEND_MODE* o *LOW_POWER_MODE* donde se ven restringidas.

Tabla 3: Valores iniciales de configuración del magnetómetro.

Parámetro	Valor	Descripción
ODR	100 Hz	Frecuencia con la que se realizarán las mediciones.
Rango	±1300 µT (ejes X/Y) ±2500 µT (eje Z)	Único rango disponible para el magnetómetro.
Modo de operación	*FORCED_MODE*	En este modo se toman las medidas de manera periódica y autónoma, a una frecuencia fija de 100 Hz, coincidente con la frecuencia a la que el integrado *BMI160* leerá de los registros del magnetómetro.

Después, se configura el integrado *BMI160* para ejercer de maestro sobre el magnetómetro *BMM150*, indicando también los métodos de lectura y escritura que debe seguir (siendo estos los de la librería creada) así como la dirección del magnetómetro, 0x13.

Por último, se habilita la emisión de interrupciones cada vez que se realiza una nueva lectura por parte de alguno de los tres sensores.

A partir de este punto, la rutina seguida a la hora de leer los datos de los sensores se repite de la siguiente manera:

- Se recibe una interrupción indicando que alguno de los tres sensores ha realizado una medida nueva.
- Se lee el registro de estado de los sensores y se comprueba si el sensor que se desea leer (atendiendo al estado en el que se encuentre el software: ACC, GYR o MAG) tiene su *flag* activado.

- Si es el caso, se realiza una lectura de los seis registros que conforman la medición en los tres ejes.
- Se espera a una nueva interrupción.

A continuación, se expone esta rutina de atención a la interrupción para el caso del estado ACC, a través de capturas realizadas con un analizador lógico. De este modo, se reflejan también la correcta implementación del protocolo *I2C* en sus operaciones de escritura y lectura.

En la *Figura 6* se observa cómo tras recibir un nivel alto en la señal de interrupción (INT) se inicia el proceso de lectura del registro de estado, con dirección 0x1B.

Como en toda transacción del protocolo *I2C* se comienza indicando la dirección del esclavo, seguido por un '0', indicando que se trata de una operación de lectura, para terminar con la dirección del registro sobre la que se va a leer o escribir antes de cerrar la transmisión.

Tras esto, comienza la transacción de escritura, indicado mediante el '1' tras la dirección de esclavo, recibiendo en su segunda parte el contenido del registro de estado.

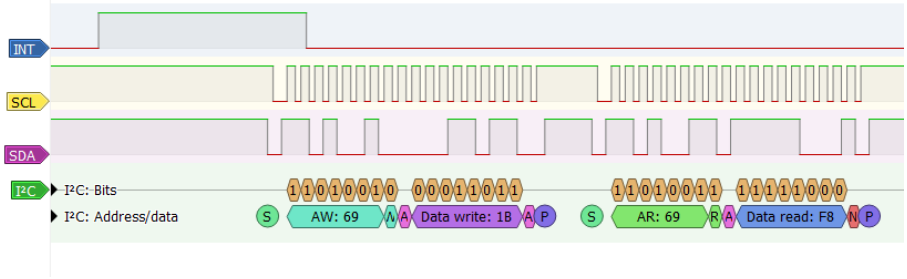

Figura 6: Operación de lectura del registro 0x1B STATUS.

En la hoja de características del integrado se indica que el *flag* correspondiente al acelerómetro es el más significativo. Al comprobar que está activo, se continúa con la lectura de los registros de medidas del sensor.

La medida en cada eje tiene un tamaño de 16 bits, es decir, dos registros de un byte. Por lo tanto, para conocer la medida de los tres ejes se debe realizar una lectura de 6 registros. Debido a que son registros contiguos, mediante la librería creada es posible indicar el registro inicial y el tamaño total que se quiere leer a partir de ese punto. Así, no es necesario realizar seis operaciones de lectura independientes, liberando antes el canal *I2C*.

En la *Figura 7* y *Figura 8*, se observa cómo, tras apuntar al registro de medida más bajo (0x12) e indicar que se trata de una operación de lectura, se reciben los 6 bytes encadenadamente. Como el primer byte recibido es el menos significativo y el último el más significativo, es necesario reordenar los datos además de agruparlos en parejas para interpretar las mediciones correctamente:

- Eje x: 0xFEEF
- Eje y: 0xFF18
- Eje z: 0x3FB9

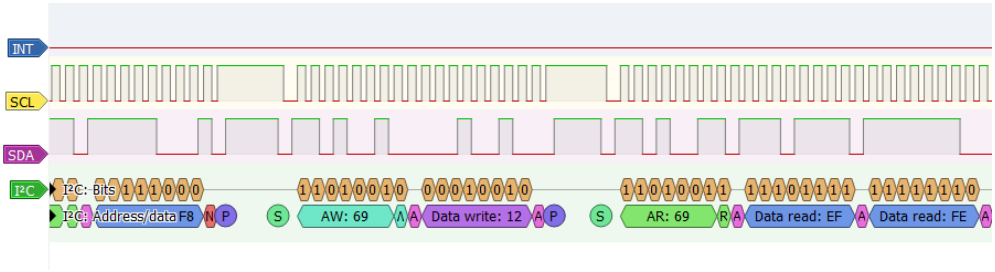

Figura 7: Operación lectura de los registros de datos del acelerómetro (parte 1).

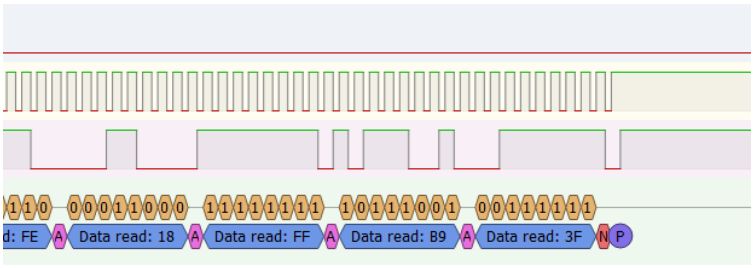

Figura 8: Operación de lectura de los registros de datos del acelerómetro (parte 2).

Tras esto se finalizaría la transacción, a la espera de recibir una nueva interrupción.

Tras mostrar el proceso de lectura, en la *Figura 9 y Figura 10* , se indica el proceso de escritura del protocolo *I2C* para un cambio en el valor del rango y *ODR* del acelerómetro de manera simultánea.

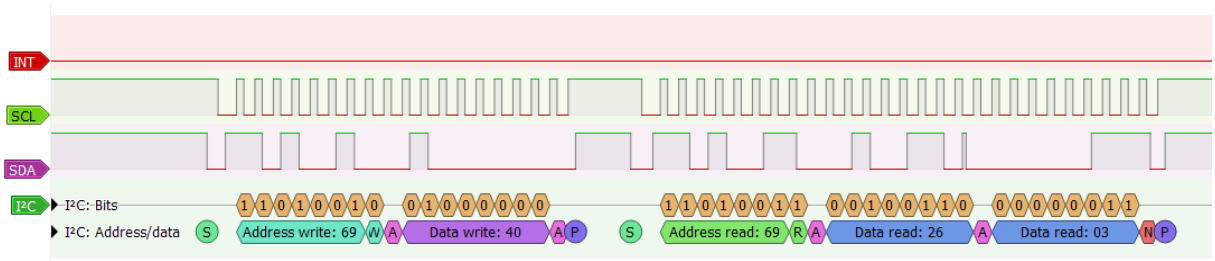

Figura 9: Operación de escritura en los registros de configuración del acelerómetro (parte 1).

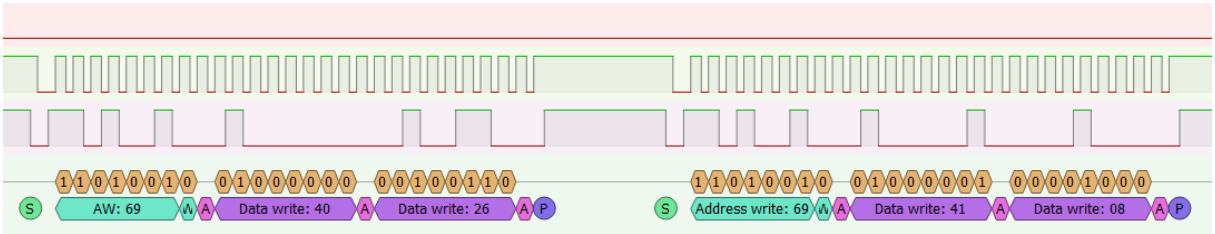

Figura 10: Operación de escritura en los registros de configuración del acelerómetro (parte 2).

En la hoja de característica se indica que los registros donde se almacena el *ODR* y el rango son los registros consecutivos 0x40 y 0x41, en sus 4 bits menos significativos. Para no alterar el contenido del resto de bits, se realiza primero una operación de lectura y se modifica únicamente el valor de las posiciones objetivo.

De esta manera, en el registro 0x40 se modifican los 4 bits menos significativos, siendo la correspondencia 01100 = 25Hz, y de igual manera en el registro 0x41, siendo la equivalencia en este caso 1000 = ±8g.

Para la implementación de la librería que gestiona la funcionalidad de los sensores, se ha trabajado sobre la librería que ofrece *Bosch* para los dos integrados, tras haber comprobado su fiabilidad en el seguimiento de los pasos indicados en la hoja de características para la modificación de los registros.

Del acelerómetro se extrae una medida de tamaño 16 bits por cada eje. Se tiene, por lo tanto, una resolución de 2^{16}, lo que se traduce en un rango de valores decimales desde -32.768 a 32.767. Estos números son adimensionales por ello es necesario traducirlos a una unidad conocida para su correcta interpretación. Como el dato que se extrae del acelerómetro es la aceleración sufrida en proporción a la aceleración de la gravedad, se usan unidades "g". Así, una medida 1g indicará una aceleración de valor 9,8 m/s^2.

Para realizar esta conversión, es necesario dividir el valor leído entre la sensibilidad, que es dependiente del rango que se selecciona, Δg, según la ecuación (1):

$$Sensibilidad = \frac{2^{16}}{\Delta g} \ [LSB/g] \qquad (1)$$

El sensor *BMI160* permite trabajar en los rangos ±2g, ±4g, ±8g y ±16g. Haciendo uso de la ecuación (1), se obtienen unos valores de sensibilidad de 16384 LSB/g, 8192 LSB/g, 4096 LSB/g y 2048 LSB/g respectivamente. Se observa que a mayor rango hay menor sensibilidad. Esto supone que cuanto menor sea el rango, el dispositivo será capaz de captar variaciones más pequeñas en las mediciones.

El rango del acelerómetro irá en sintonía con la finalidad que se le vaya a dar. En experimentos y pruebas de medidas de seguridad de vehículos es necesario medir la fuerza sufrida en un choque, a partir de la aceleración medida. Esto supone ser capaz de medir cambios muy bruscos y de valor elevado en aceleración del vehículo. Por eso se configura el acelerómetro para un rango amplio, ya que las variaciones pequeñas no son relevantes en este caso.

Por el contrario, en un podómetro se configura el acelerómetro en rangos bajos ya que la aceleración sufrida debida al desplazamiento de una persona no va a ser muy elevada respecto a la aceleración de la gravedad.

Una vez se tiene la sensibilidad, se multiplica por el entero leído y se obtiene la medida, que indica la aceleración experimentada en los ejes, de manera proporcional a la aceleración de la gravedad.

Para evitar errores en la medida es necesario calibrar el dispositivo, y establecer el *offset* adecuado. Este *offset* es el error de base común a todas las medidas que debe eliminarse de la medida final.

Para ello, se coloca el dispositivo en una posición de reposo como puede ser sobre una superficie horizontal. Como la gravedad tiene dirección vertical respecto al plano horizonte, los ejes X e

Y deben medir 0 ya que no se ven afectados por ella. Sin embargo, el eje Z debe experimentar una medida de 1g ya que se encuentra en la misma vertical que la aceleración de la gravedad y es la única fuerza que experimenta el dispositivo.

El procedimiento consiste en tomar un número significativo de muestras para obtener su valor medio y obtener la diferencia con el valor esperado. Dicha diferencia se almacena invirtiendo su signo en los registros del integrado. El integrado *BMI160* sumará ese valor negado a las mediciones que realice antes de almacenarlas en sus registros.

La sensibilidad para el cálculo de *offsets* es fija y de valor 3,9 mg para cualquier rango. Debido a que el offset en cada eje ocupa un registro de un byte, el mayor offset que se podrá corregir será de ±499,2 mg.

Adicionalmente a las medidas del acelerómetro, en el *display* se muestra la posición angular del dispositivo.

En la *Figura 11* se representa la relación entre el ángulo de inclinación y la aceleración de la gravedad para el caso de un sensor biaxial.

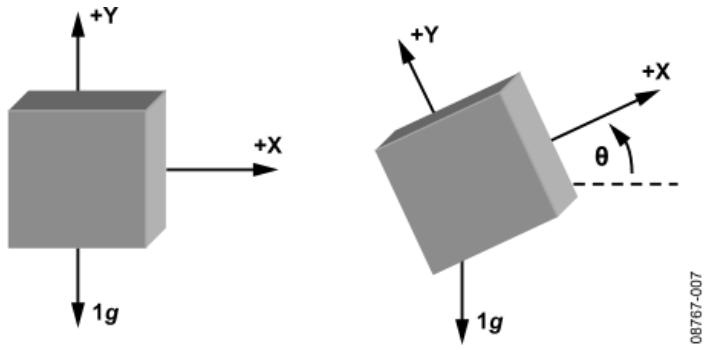

Figura 11: Uso de un sensor de dos ejes para el cálculo de su ángulo (Fuente: www.analog.com)

Es directo obtener el ángulo de inclinación mediante las relaciones trigonométricas sencillas (2) y (3).

$$Ax = 1g * \sin(\theta) \quad (2)$$
$$Ay = 1g * cos(\theta) \quad (3)$$

Resultando finalmente en el cálculo del arcotangente de la división de las dos medidas tomadas.

$$\theta = \arctan\left(\frac{Ax}{Ay}\right) \quad (4)$$

Partiendo de este desarrollo se obtienen las ecuaciones (5), (6) y (7) que determinan los ángulos en un sistema triaxial como el de la *Figura 12*.

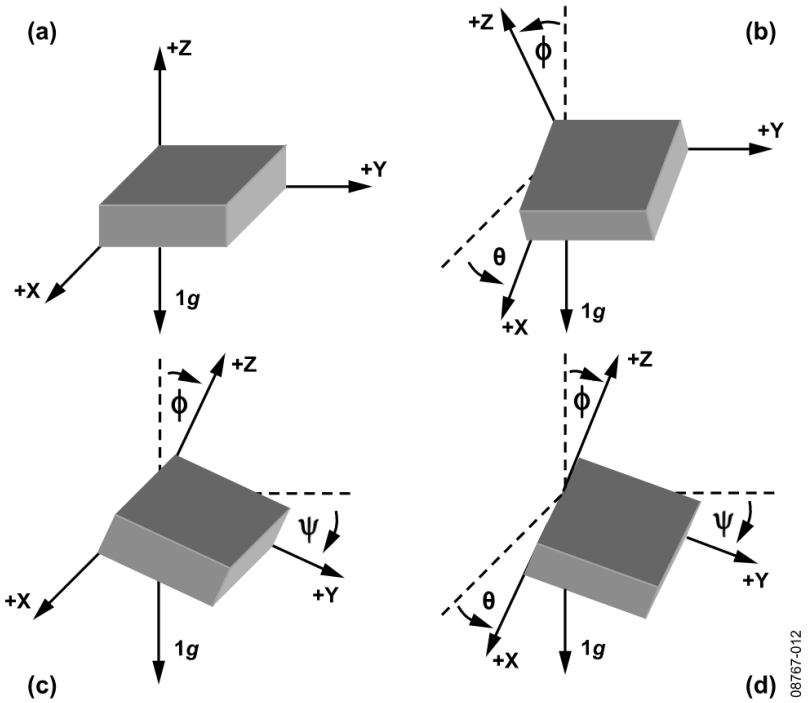

Figura 12: Ángulos en sensor de tres ejes (Fuente: ww.analog.com)

$$\theta = \arctan\left(\frac{Ax}{\sqrt{A_y^2 + A_z^2}}\right) \tag{5}$$

$$\psi = \arctan\left(\frac{A_y}{\sqrt{A_x^2 + A_z^2}}\right) \tag{6}$$

$$\phi = \arctan\left(\frac{\sqrt{A_x^2 + A_y^2}}{A_z}\right) \tag{7}$$

Para el desarrollo descrito, así como para las imágenes explicativas, se ha consultado la nota de aplicación de la referencia [11].

Al igual que el acelerómetro, el giróscopo genera mediciones de 16 bits, que deben ser transformadas para poder interpretarse en unidades de velocidad angular: grados partido de segundos [º/s].

En este caso, el giróscopo ofrece los cinco rangos siguientes: ±125º/s, ±250º/s, ±500º/s, ±1000º/s y ±2000º/s. Sin embargo, las medidas de sensibilidad no se obtendrán a través de la fórmula (1), sino que se usarán las indicadas en la hoja de características del sensor, como recomienda el fabricante, debido al mayor error en su cálculo.

Para la calibración del sensor y el cálculo de los *offsets*, se coloca el dispositivo en una posición de reposo, sin importar la inclinación. Debido a que el giróscopo mide la velocidad angular en sus ejes, en reposo las medidas tomadas deben ser nulas en los tres ejes.

En el caso del giróscopo, el offset ocupa 10 bits por eje, con una sensibilidad fija de 0,061º/s, obteniéndose así un offset máximo de ±31,25º/s.

El magnetómetro proporciona junto con las medidas de la intensidad del campo magnético en los tres ejes, el valor de la resistencia del efecto Hall. Esta resistencia es consecuencia de las temperaturas elevadas que se alcanzan debido al flujo de corriente en el semiconductor. En la hoja de características del integrado se indica que, para la correcta interpretación de las medidas, se debe realizar una compensación del valor fuera del hardware, usando para ello la *API* (*Application Programming Interface*, por sus siglas en inglés) que ofrece Bosch.

Una vez hecho esto, se obtienen medidas compensadas con rango ±1300 μT para los ejes X e Y, y ±2500 μT para el eje Z. Tras esta primera corrección es necesario continuar con la calibración compensando las perturbaciones ajenas.

En un entorno sin efectos magnéticos externos, en el que sólo influye el campo magnético terrestre, si se gira sobre sí mismo el magnetómetro sobre una superficie plana, las medidas tomadas en los ejes X e Y formarán una circunferencia centrada en el origen. En una situación no ideal, el campo magnético terrestre no es el único presente, sino que se encuentran las perturbaciones conocidas como *hard iron* y *soft iron.*

La perturbación *hard iron* se debe al campo magnético generado por dispositivos en el entorno, como pueden ser equipos electrónicos, altavoces o motores. Este campo magnético es constante siempre que el dispositivo que lo genere no cambie su posición o características. De esta manera el error que genera también es constante, y, se reconoce porque la circunferencia que forma el magnetómetro está desplazada del origen.

Para corregirlo basta con encontrar el offset en cada eje, según se indica en las ecuaciones (8) y (9).

$$offset_x = \frac{X_{max} - X_{min}}{2} \tag{8}$$

$$offset_y = \frac{Y_{max} - Y_{min}}{2} \tag{9}$$

La perturbación *soft iron* es más compleja de corregir, ya que no es constante y depende de la orientación del magnetómetro. No se debe a campos electromagnéticos adicionales, si no a materiales porosos magnéticamente que perturban el entorno, como puede ser el cobre o el níquel. En este caso la perturbación provoca un cambio en la forma de la circunferencia, que pasa a ser un elipsoide.

Una forma sencilla de corregirlo es obteniendo la ratio entre el radio del eje X y el radio del eje Y, según la ecuación (10), el cual, en una situación ideal, debería tener valor unidad. En caso de que sea menor que 1, se multiplica este factor de escala por el eje X, y, en caso contrario, se divide.

$$r = \frac{X_{max} + X_{min}}{Y_{max} + Y_{min}} \tag{10}$$

Esta calibración es válida únicamente para el entorno en el que se realiza. Si el dispositivo se cambia de lugar, sería necesario realizar una nueva calibración.

Además de las medidas del sensor, el dispositivo implementa y muestra por pantalla una brújula electrónica.

Para ello, hay que tener en cuenta que las medidas del magnetómetro en un eje serán nulas si está perpendicular al flujo magnético. En caso contrario, la medida será positiva si el eje apunta en la posición del sentido del flujo magnético y negativo si el sentido es contrario.

El flujo magnético terrestre fluye del polo norte al polo sur magnético, coincidentes con el polo sur y polo norte geográfico respectivamente. De esta manera, un valor positivo en un eje del sensor indica que éste apunta en sentido al polo norte geográfico.

Teniendo esto en cuenta, se obtiene el ángulo de desviación respecto al norte geográfico mediante la siguiente operación trigonométrica (11).

$$\alpha = \frac{\pi}{2} - \arctan(y/x) \tag{11}$$

Para obtener unos valores comprendidos entre 0 y 2π radianes es necesario tener en cuenta el cuadrante en el que se encuentra el ángulo, atendiendo a los signos de las magnitudes en los ejes.

Ya que la brújula implementada se representa en dos dimensiones, los valores del eje Z no es necesario tenerlos en cuenta para su correcto funcionamiento.

Para consultar información acerca de la toma de medidas del magnetómetro y de las perturbaciones que las alteran se ha consultado el artículo de referencia [12].

4.2.3 BLOQUE TÁCTIL

El bloque táctil tiene la función de interpretar las interacciones del usuario en la pantalla para su traducción en señales que provoquen las acciones oportunas.

El *display* instalado en este dispositivo cuenta con el *driver* FT5346 para el apartado táctil, que implementa un protocolo de comunicaciones *I2C*. Para su integración se ha usado el mismo canal que para bloque de los sensores, con dirección 0x38. Este driver indica las coordenadas de la pantalla en las que se ha producido un toque, así como el tipo de toque, es decir, si ha sido un desplazamiento, un toque doble, un toque simple, etc.

Además de las dos señales del protocolo *I2C*, la interfaz con el sensor incluye una señal de interrupción, la cual usa el integrado para notificar que se ha realizado una pulsación. El integrado se ha configurado de manera que esta señal de interrupción oscile, a una frecuencia de 100Hz, mientras se está pulsando la pantalla, siendo posible también configurarlo de manera que se mantenga constante a nivel bajo, para después volver al estado de reposo a nivel alto una vez finaliza el toque.

La rutina de atención a la interrupción consiste en los siguientes pasos:

- Se espera a recibir la interrupción (flanco de bajada en este caso).
- Mientras la señal de interrupción oscila, se lee de manera periódica el registro que indica el tipo de toque para comprobar si se trata de un toque puntual o un desplazamiento. En

el caso de ser un desplazamiento, tras leer el sentido del gesto, no sería necesario continuar leyendo, ya que queda definida completamente la acción.

- Una vez la señal de interrupción cesa su oscilación y vuelve a su estado de reposo, si ha sido un toque puntual se leen los registros de datos para obtener sus coordenadas.
- Una vez definido el gesto, éste se traduce a una señal interpretable por el resto de bloques para desencadenar la acción correspondiente. Esto se realiza atendiendo a la pantalla concreta en la que se encuentra el dispositivo, pudiendo ser toques inocuos si no se ha interactuado en la zona adecuada.

En la *Figura 13* se observa el comportamiento cuando se trata de un desplazamiento. Se lee periódicamente hasta leer un valor de distinto de cero o uno. En este caso, el valor 0x14 se corresponde con un desplazamiento hacia la derecha. Como ya ha quedado definido el toque se cesa la lectura. Se observa también como una vez liberado el canal, aparecen varias transacciones. Esto se debe a que el desplazamiento a provocado un cambio en la pantalla, pasando de la pantalla principal, en la que no hay actividad ninguna, a la pantalla del acelerómetro, donde se realizan lecturas periódicas del sensor a través del canal *I2C compartido*.

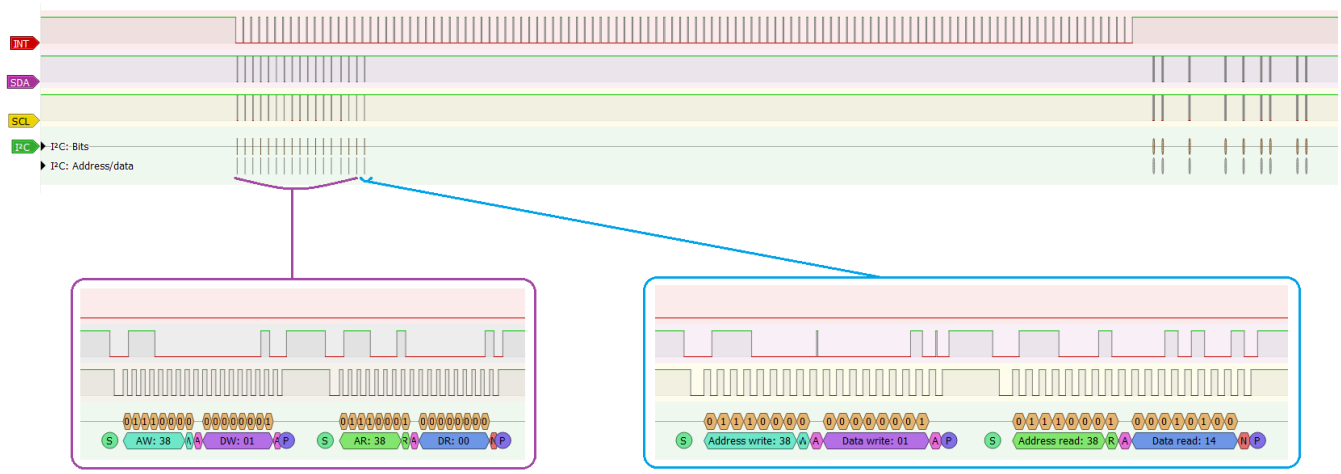

Figura 13: Comunicación con driver FT5346. Acción desplazamiento

En la *Figura 14*Figura 14: Comunicación con driver FT5346. Acción toque puntual por el contrario, se observa el comportamiento de esta rutina para un toque puntual. Se comprueba en cada lectura que se trata de un toque puntual (valor 0x00 en el registro 0x01) y una vez liberado el canal se extraen los datos de las coordenadas.

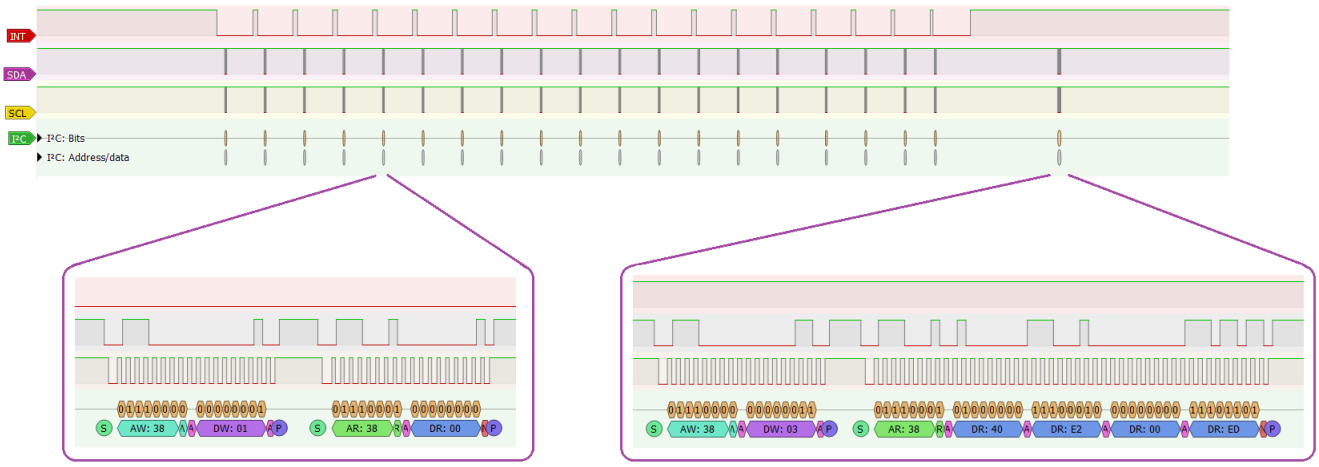

Figura 14: Comunicación con driver FT5346. Acción toque puntual

Los datos de las coordenadas leídos se forman de manera que de los dos primeros bytes, los 12 bits menos significativos se corresponden con la coordenada horizontal y de los dos últimos bytes, los 12 bits menos significativos se corresponden con la coordenada vertical.

En este caso la coordenada X sería 0x0E2, 226 en decimal, y la coordenada Y sería 0x0ED, 237, formando el punto (226, 237).

4.2.4 BLOQUE DISPLAY

Este bloque tiene la función de configurar la pantalla *lcd* y gestionar lo que se muestra en cada momento en sintonía con el funcionamiento del dispositivo.

Implementa dos interfaces distintas, una para la configuración del *driver* y los modos de funcionamiento, y otra para la transmisión de las imágenes. Los protocolos de estas interfaces no están estandarizos, por lo que es necesario su diseño completo en base a las pautas que marca el fabricante en la hoja de características.

Para la lectura y escritura de los registros del *driver* NV3035C , es necesario una interfaz serie bidireccional. Esta interfaz se compone de tres señales: una señal de habilitación (SPENB), una señal de reloj (SPCLK) y otra señal de datos (SPDA).

Para la implementación de este protocolo, denominado *3-W*, se ha usado la técnica de *bit-banging*. Ésta consiste en actuar directamente sobre los pines de entrada/salida de las tres señales sin ninguna configuración previa. Es decir, la señal SPCLK no se trata como una señal *pwm (pulse width modulation,* por sus siglas en inglés*)* o de reloj, si no que su nivel alto o bajo se varía haciendo una operación de escritura de uno o cero en el pin explícitamente.

De esta manera, se van poniendo a nivel alto o bajo cada una de las tres señales de manera secuencial, en base a los diagramas que proporciona el fabricante, poniendo especial atención a los tiempos límite marcados.

Una vez creada esta librería ya es posible aplicar la configuración inicial al *display*, cuyos elementos más relevantes son el tipo de protocolo para el envío de las imágenes, el formato de las imágenes y la cantidad de bits que se envían en paralelo.

El formato de imagen configurado en el *driver* para representarse por pantalla es *RGB*. Esta denominación viene de *Red Green Blue* (rojo, verde y azul en castellano), los colores primarios que en sus distintas combinaciones forman el resto de colores. De esta manera, para almacenar o proyectar una imagen basta con definir la proporción de color rojo, verde y azul en cada punto de la misma. Esta unidad de información se denomina píxel.

El *display* tiene un tamaño de 320x240 píxeles lo que hacen un total de 76800 píxeles, que se deben extraer de las imágenes a representar. Se ha considerado oportuno para esto usar el formato de imagen *bmp* (*bit map*, mapa de bits en castellano), que no es más que una ristra ordenada de píxeles. El archivo lo conforman una cabecera, con información autodescriptiva, seguida por el conjunto de píxeles, de tres bytes (uno por color) cada uno y termina con dos bytes de valor cero que indican el fin del documento [13].

Los píxeles se ordenan recorriendo las filas de manera ascendente y las columnas de izquierda a derecha. Es decir, el primer píxel que se almacenará en el fichero será el que se encuentre en la esquina inferior izquierda de la imagen, y el último píxel almacenado será el que se encuentre en la esquina superior derecha de la imagen. También se ha de tener en cuenta el orden de los colores dentro del píxel ya que a diferencia de lo que resulta intuitivo, estos no se almacenan siguiendo el orden RGB, si no BGR: azul, verde y por último rojo.

Otra característica relevante de este formato es que obliga a que el número de columnas sea múltiplo de 4, y en caso de no serlo rellena con información trivial columnas hasta alcanzar el siguiente múltiplo de cuatro. Esto ocurre en imágenes pequeñas (como figuras de selección o iconos) con un ancho de 15 píxeles, lo que supone un total de 45 bytes por fila. El formato rellenará las filas con tres bytes más para alcanzar los 48, múltiplo de cuatro más cercano por arriba. Es importante tener en cuenta cuantos bytes extra contienen las filas del fichero para obviarlos a la hora de leer el contenido de éste.

La cabecera cuenta con un tamaño fijo de 54 bytes, con información relevante para la correcta interpretación de los datos. En la *Figura 15* se muestra el comienzo de un fichero con extensión *bmp*, correspondiente a una imagen completamente verde, con la información más relevante comentada en la *Tabla 4*.

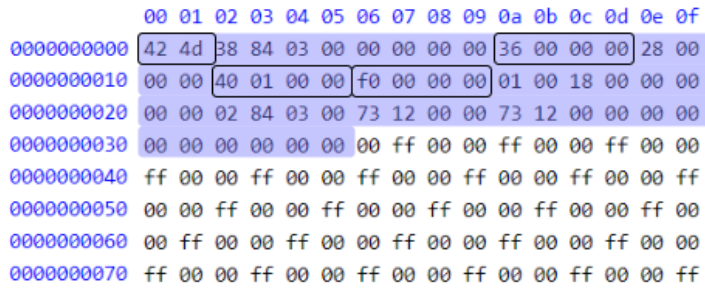

Figura 15: Detalle del formato de archivo bmp

Tabla 4: Especificación cabecera

Posición (extensión)	Valor	Comentario
0x00 (2 bytes)	0x42, 0x4D	Todos los ficheros en formato bmp empiezan con estos dos bytes, que en formato ASCII se corresponden con los caracteres B y M, haciendo referencia a las iniciales de Bit Map.
0x0A (4 bytes)	0x36	Offset a partir del cual finaliza la cabecera y comienza la información de los píxeles de la imagen.
0x12 (4 bytes)	0x00000140	Tamaño del ancho del mapa de bits en decimal: 320(1)
0x16 (4 bytes)	0x000000F0	Tamaño del largo del mapa de bits en decimal: 240(1)
0x36 (hasta fin)	-	A partir de este punto ya comienza el mapa de bits de la imagen. Como se trata de una imagen completamente verde, se repiten los valores 0x00 0xFF y 0x00, siguiendo el orden BGR. Es decir que todos los píxeles tienen únicamente componente verde.

El envío de la imagen al controlador del *display* se ha de hacer a través de un *buffer* que contenga únicamente el mapa de píxeles, ordenados de arriba a abajo y de izquierda a derecha, con los colores ordenados de la forma rojo, verde y azul. Por eso es necesario conocer todo lo documentado acerca del formato *bmp*, para poder transformar el archivo de imagen a un *buffer* de la forma adecuada.

Una vez listo el *buffer*, se enviará al *display* siguiendo el protocolo que se describe a continuación.

El controlador NV3035C no dispone de memoria interna y por este motivo es necesario un envío constante de imágenes, ya sean fotogramas de un vídeo o imágenes estáticas, ya que en caso contrario se quedará la pantalla en blanco.

Para el envío de las imágenes el fabricante indica que el controlador soporta dos protocolos distintos, denominados: “H/VSYNC” y “DE”. Ambos están conformados por un bus de datos una señal de reloj, y señales de control. La diferencia radica en que el primero implementa dos señales diferenciadas para el sincronismo entre filas y columnas, y el segundo las aglutina en una única señal.

Como se describe en el apartado de análisis de resultados, el mayor reto en el diseño es alcanzar la sincronía entre la señal de reloj y el resto de señales a altas frecuencias. Debido a que el protocolo DE cuenta con una señal menos, se ha considerado más adecuado para este diseño.

En la *Figura 16* se muestra el resultado de la implementación de este protocolo, escalado para facilitar su comprensión y análisis, simulando un *display* de tamaño 4x10 en lugar de 240x320 y siendo el dato a transmitir 0x2AA.

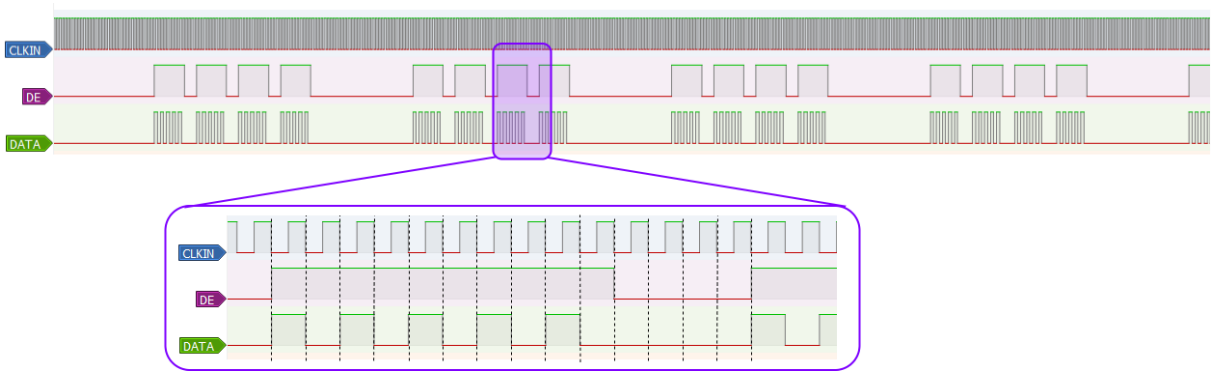

Figura 16: Protocolo display escalado

La señal DE organiza los datos de la transmisión de manera que genera un pulso por cada fila del *display* (4 para el caso simulado). Dentro de cada pulso de la señal DE, la señal de datos transmite la información de la fila, que coincidirá con el número de columnas del *display* (10 para el caso simulado). Una vez finaliza la transmisión de la fila, el canal entra en estado de reposo durante un tiempo marcado, para repetir el proceso tantas veces como filas tenga el *display*. Una vez terminado el envío de la imagen, el canal entra de nuevo en un estado de reposo previo a continuar con el bucle de envío de imágenes. Todos estos cambios en las señales DE y DATA se realizan coincidentes con el flanco de bajada de la señal de reloj.

A pesar de que en la captura se observe que DATA es una única señal, el fabricante indica que debe ser un bus de datos, pudiendo estar conformado por 8 o 24 líneas que se transmitan de manera paralela, con las siguientes características:

- DE_24 bits: La comunicación consta de 24 líneas de datos (ocho para cada color primario), lo que significa que cada ciclo de reloj se puede mandar un píxel completo (24 bits) de la imagen. La frecuencia de reloj ha de ser de 6,4MHz.
- DE_8 bits: La comunicación consta de 8 líneas de datos. En este caso, se necesitan tres ciclos de reloj para enviar la información de un píxel. Para mantener un adecuado refresco de la imagen en el *display*, la frecuencia de reloj se debe ver aumentada a 27 MHz.

Al no disponer de 24 pines de salida libres en la *Raspberry Pi*, parece que la opción obligada es la transmisión en paralelo de 8 bits, y 27 MHz de frecuencia de reloj.

Sin embargo, esta opción resulta inviable ya que la frecuencia de reloj requerida resulta inmanejable debido a que supera con creces la frecuencia de la *Raspberry Pi* de escaneo de los pines de entrada y salida. Al detectar un flanco de bajada ya han ocurrido múltiples más y la imagen transmitida es irreconocible.

Además, la frecuencia de escaneo de los pines afecta tanto a la lectura como escritura, lo que implica que el tiempo que se transcurre en la escritura de los pines de datos es igual de elevado. Más teniendo en cuenta que la escritura en los ocho pines se hace de manera secuencial con lo que el retardo se multiplica por 8 veces.

La opción alternativa implementada para paliar esta imposibilidad del diseño consiste en configurar el *driver* en modo 24 bits pero reduciendo el bus de datos a tres líneas, una por cada uno de los tres colores primarios. Es decir, las ocho señales que forman los ochos bits de cada color se empalman a una única salida de la *Raspberry Pi*. De cara al *display*, éste sigue

recibiendo los 24 bits de manera simultánea, mientras que la Raspberry Pi únicamente está enviando 3 bits simultáneamente. De esta manera, es posible usar la frecuencia del reloj más baja, a costa de perder profundidad de color, pasando de 16777216 (2^{24}) colores distintos a únicamente 8 (2^3).

A pesar de mejorar la calidad del protocolo, en esta configuración continúa siendo un problema el tiempo que transcurre desde que se da un flanco de bajada en la señal de reloj, se detecta y genera la interrupción y se modifican las tres salidas secuencialmente. Por este motivo, es necesario hacer una segunda modificación a la implementación del protocolo.

Para eliminar el retardo generado en la escritura de las tres salidas, se decide reducirlas a una única salida. Las 24 entradas del *driver* son manejadas por un único pin de salida. De esta manera las acciones que se realizan en cada interrupción por flanco de bajada de la señal de reloj quedan reducidas a modificar un solo pin, con lo que mejora notablemente la calidad de las imágenes mostradas en el *display*. La contrapartida de esta solución es que el número de colores de la imagen queda reducido a 2: negro o blanco.

Para no eliminar del diseño la posibilidad de emitir imágenes en color se han incluido los jumpers JP1 y JP2, de manera que si están puestos en la posición 3-2 las tres salidas RGB serán independientes y cada una gestionará sus ocho entradas desde el software. En caso de estar puestos en la posición 1-2, el software sólo trabajará con una salida que estará unida por hardware a las 24 entradas y la imagen será en blanco y negro.

En la *Figura 17* se ve reflejado este detalle del diseño hardware.

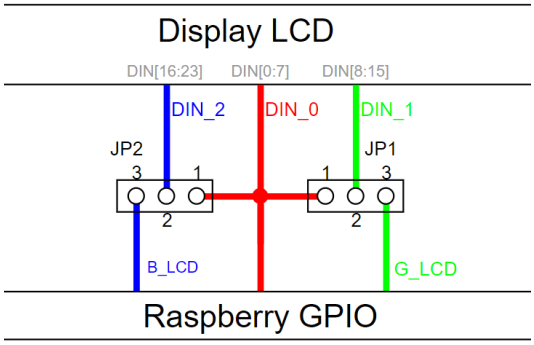

Figura 17: Detalle configuración hardware de color para display.

Con este diseño el retardo sigue siendo excesivo así que finalmente se fija la frecuencia de reloj a 1,7MHz en vez de los 6 MHz que marca el fabricante. Este valor se ha seleccionado por suponer un buen equilibrio entre velocidad de refresco y calidad de imagen.

En el apartado de análisis de resultados se comenta en profundidad el rendimiento del protocolo y los distintos retardos sufridos.

5 ANÁLISIS DE LOS RESULTADOS

Tras haber finalizado la implementación del diseño, se realiza un amplio análisis de todas sus funcionalidades y apartados. Esto es, el correcto funcionamiento de los protocolos y configuraciones usadas, la calidad de las medidas realizadas, la correcta representación de las imágenes en el *display* y la interactuación hombre-máquina.

Para facilitar el estudio a posteriori de los datos recabados, al iniciar la aplicación se crean tres ficheros de texto, uno por sensor. En ellos se almacena cada nueva medición leída del sensor correspondiente, junto con otros datos como la sensibilidad, el calibrado o una marca de tiempo. Esta marca de tiempo se recoge de los propios registros del sensor y se obtiene mediante un contador de 24 bits y una resolución de 39 µs, lo que supone un tiempo total de aproximadamente 11 minutos, momento a partir del cual reinicia su cuenta.

5.1 ACELERÓMETRO

En primer lugar, se verifica que la configuración del sensor se realiza correctamente. Para esto se puede realizar una lectura de los propios registros de configuración y comprobar si son correctos. Paralelamente se puede analizar el comportamiento del dispositivo externamente y observar si cumple con los valores de la *Tabla 1*: tasa de muestreo de 25 Hz y rango de 2 g.

Para la verificación visual de la tasa de muestreo, hay que tener en cuenta que la interrupción de nueva medida de los sensores tiene una frecuencia igual a la mayor tasa de muestreo de los tres sensores. En el caso de los valores iniciales, la frecuencia más alta es la del magnetómetro, con valor de 100 Hz.

Este comportamiento se verifica en la captura de la *Figura 18,* donde se observa cómo se recibe una interrupción cada 10 ms, pero únicamente se leen los registros de datos cada 4 interrupciones, momento en el que el *flag* del acelerómetro está activado en el registro de estado.

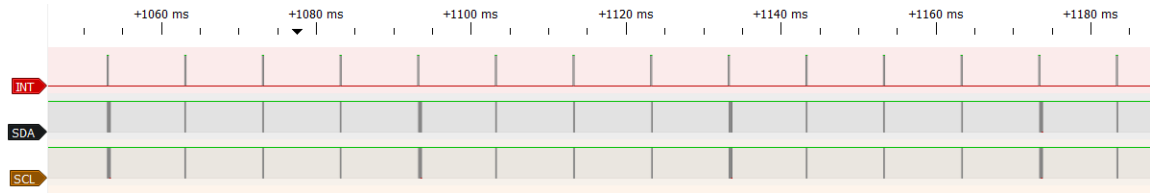

Figura 18: Muestreo del acelerómetro a 25 Hz.

Para verificar que el rango es el marcado en los valores iniciales, se posiciona el dispositivo en horizontal sobre una superficie plana y se lee el dato de aceleración en el eje Z, que debe ser de 1g.

Teniendo en cuenta que el dato puro que proporciona el sensor está en formato complemento a dos, la relación con la medida en unidades g se puede modelar como la gráfica bipolar de la *Figura 19.* En el eje vertical se representan todos los valores de aceleración posible, que serán aquellos que entren dentro del rango seleccionado, ±2g en el caso del valor inicial.

En el eje horizontal se reparten equitativamente todos los valores decimales que puede generar el sensor en función del tamaño de dato, 16 bits en este caso.

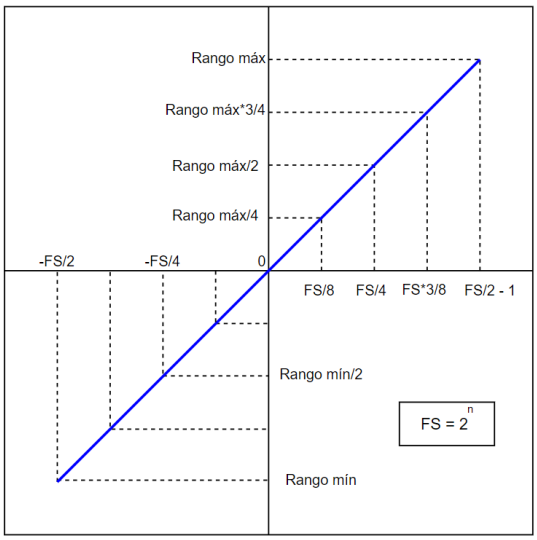

Figura 19: Relación rango-código numérico

De esta manera, para un valor de +2g se deben obtener mediciones de alrededor de 32768, y por lo tanto para un valor de +1g se deben obtener mediciones igualmente de valor medio del total positivo, es decir, 16384, como se observa en la *Figura 22*.

El segundo aspecto por verificar es la calidad de los datos medidos. Para esto, continuando en la posición de reposo horizontal, se toman suficientes medidas, antes y después de calibrar el dispositivo. Como se ha indicado anteriormente, éstas deben ser de valor nulo en los ejes x e y, y de valor 1 en el eje z.

A continuación, se expone el resultado de la calibración en los tres ejes para mil muestras recogidas, tanto en código decimal como en unidades g. Se observa como en la *Figura 20 y Figura 21* tras calcular y aplicar el offset, las medidas tomadas se aproximan considerablemente al valor real de 0, y en la *Figura 22*, el valor tras la calibración se acerca a 1g (16384 LSB).

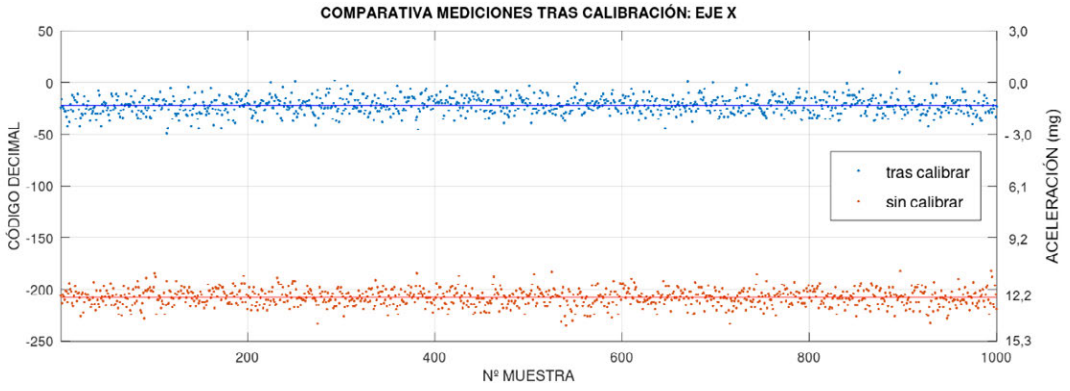

Figura 20: Calibración acelerómetro eje X

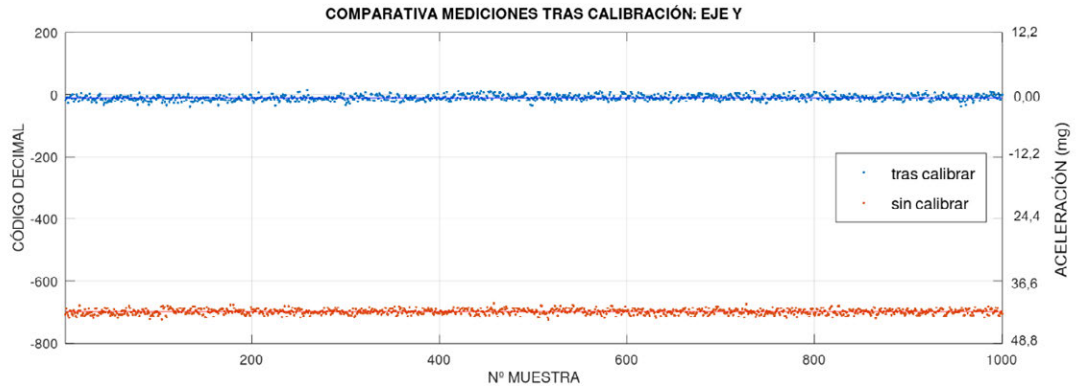

Figura 21: Calibración acelerómetro eje Y

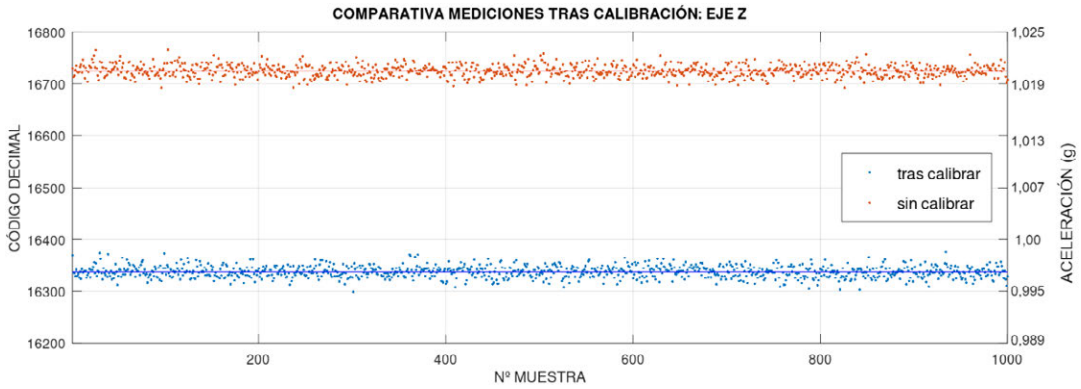

Figura 22: Calibración acelerómetro eje Z

El proceso para el cálculo del valor de offset comienza obteniendo el valor medio de la muestra en unidades *LSB* y según el rango y sensibilidad transformarlo en unidades g. A continuación, es necesario restárselo al valor ideal que debería medirse (12). Por último, este valor de offset se pasa a unidades *LSB*, usando la sensibilidad fija de 3,9 mg/*LSB* que marca el fabricante, para almacenarse en la memoria del integrado con el signo invertido(13).

A partir de este momento, las medidas que se leen del sensor ya tendrán el offset aplicado internamente.

$$offset\ [g] = Valor\ medio\ real\ [LSB]/sensibilidad[LSB/g]\ -\ valor\ ideal[g] \quad (12)$$

$$offset\ [LSB] = offset[g]/3{,}9[mg/LSB] \quad (13)$$

En la *Tabla 5* se muestra el resultado del uso de estas ecuaciones para el cálculo de los offsets en los tres ejes, así como los valores medios previos y posteriores a la calibración para las mil muestras representadas anteriormente.

Tabla 5: Resumen valores calibración acelerómetro

	Media sin offset	Valor ideal	Valor de offset	Media tras offset
Eje X	-208 c.d. / -12,70 mg	0 c.d. / 0,00 g	+3 c.d. / 11,7 mg	-22 c.d. / 1,34 mg
Eje Y	-698 c.d. / 42,60 mg	0 c.d. / 0,00 g	+11 c.d. / 42,9 mg	-11 c.d. / 0,67 mg
Eje Z	16725 c.d. / 1,021 g	16384 c.d. / 1,00 g	-6 c.d. / 23,4 mg	16338 c.d. / 0,997 mg

Para el análisis de la calidad de los ángulos medidos, se posiciona el dispositivo sobre una superficie plana con una inclinación conocida y se comprueba que el ángulo medido coincide con el ángulo de la superficie plana.

Para esto se ha montado la plataforma de la *Figura 23*, la cual, haciendo uso de un transportador de ángulos (también denominado goniómetro), se coloca a la inclinación deseada.

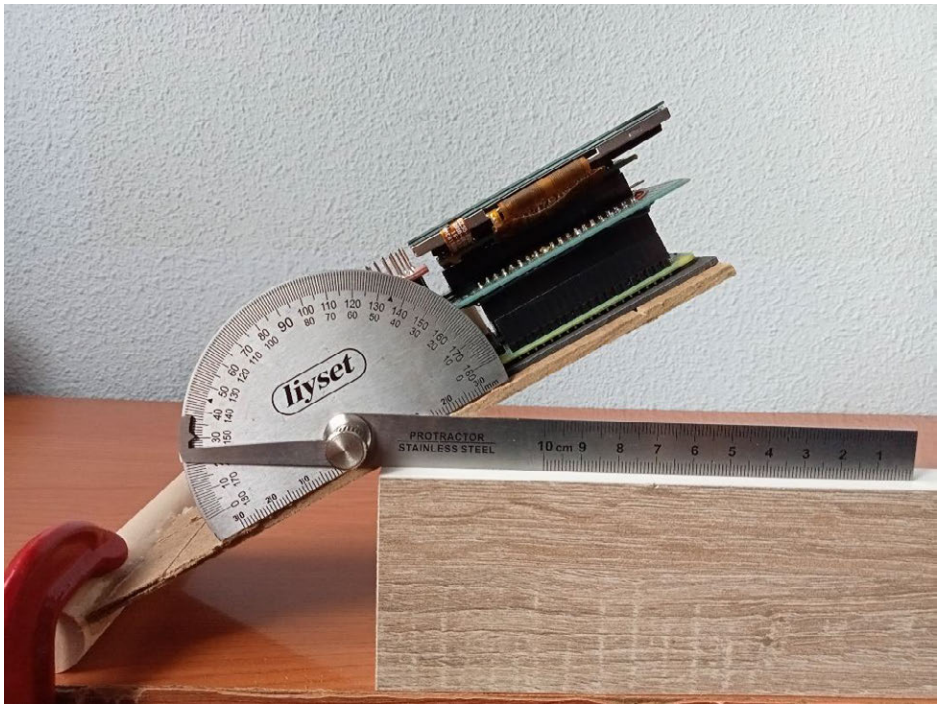

Figura 23: Montaje para medición precisa de ángulos (caso 30º)

Se han realizado mediciones a inclinaciones de 30, 45, 60 y 90 grados en cada uno de los tres ejes, obteniéndose los resultados mostrados en la *Tabla 6.*

Tabla 6: Medidas ángulos acelerómetro

	30º	45º	60º	90º
Eje X	29,781	45,404	60,284	89,675
Eje Y	30,595	45,075	60,264	89,139
Eje Z	29,867	45,142	60,366	89,191

5.2 GIRÓSCOPO

De igual manera que con el acelerómetro, se comienza verificando que la configuración inicial se realiza correctamente.

En la *Figura 24* se comprueba que de nuevo las interrupciones de nueva medida saltan a una frecuencia de 100 Hz debido al magnetómetro y únicamente cada dos interrupciones se verifica que hay una nueva medida del giróscopo y se realiza la lectura de los registros de datos. Este comportamiento es coherente con el valor inicial de frecuencia de toma de medidas de 50 Hz.

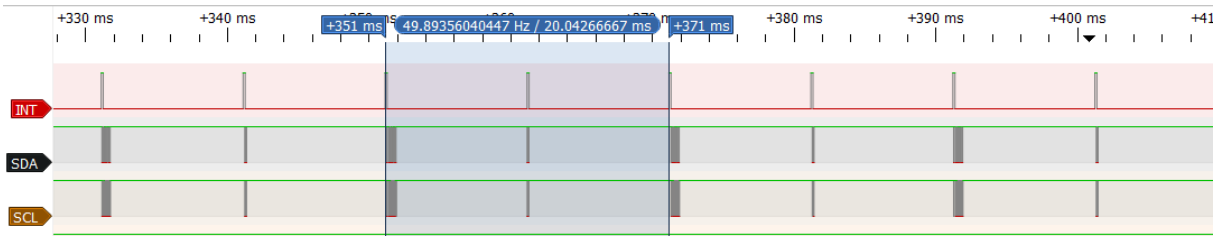

Figura 24: Muestreo giróscopo 50Hz

A falta de un dispositivo de evaluación externo capaz de rotar el dispositivo a una velocidad conocida, se hace imposible comprobar el rango del giróscopo.

Por este mismo motivo, tampoco es posible comprobar la exactitud de las medidas de velocidad angular tomadas, de igual manera que se hizo con los ángulos para el caso del acelerómetro.

Por lo tanto, únicamente se evalúa la calibración del dispositivo. Para ello, se coloca el dispositivo en una posición de reposo y se toman medidas antes y después de la calibración. En esta situación, debido a la falta de movimiento, las lecturas deben ser nulas en cada uno de los tres ejes.

En la *Figura 25, Figura 26* y *Figura 27* se enfrenta una muestra de 1000 medidas previa la realización de la calibración con otras 1000 muestras tras el proceso de calibración, para los ejes X, Y y Z respectivamente.

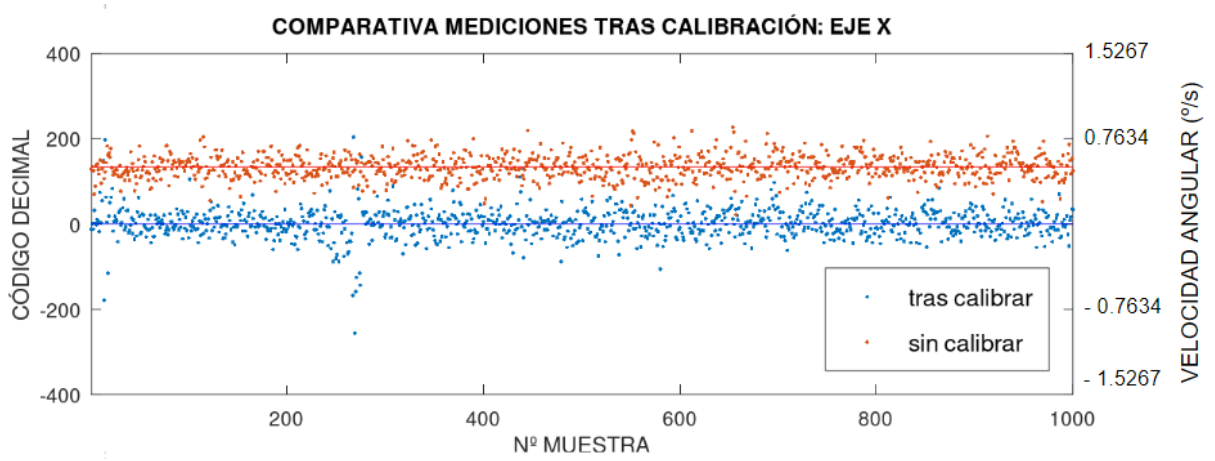

Figura 25: Calibración giróscopo eje X

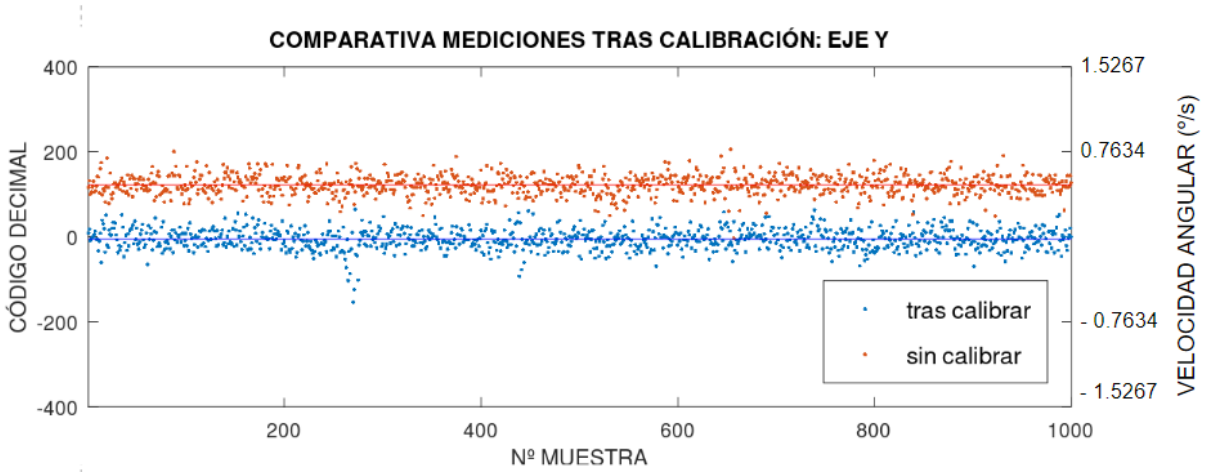

Figura 26: Calibración giróscopo eje Y

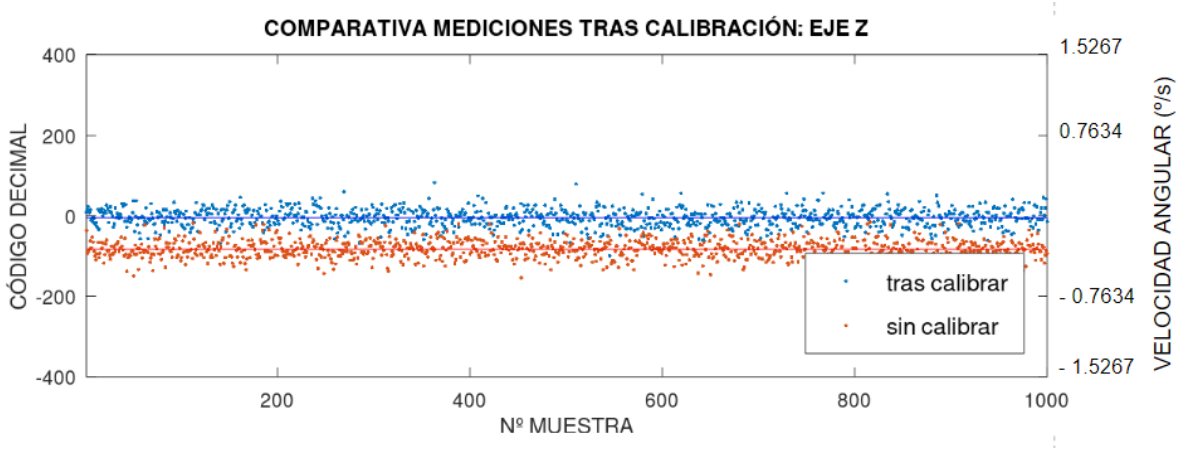

Figura 27: Calibración giróscopo eje Z

En la *Tabla 7* se exponen los datos más relevantes del proceso de calibración y la aplicación de los *offsets*. El proceso de cálculo de estos valores de *offsets* es parejo al indicado para el acelerómetro, con la excepción del tamaño, que en este caso es de 10 bits en vez de 8, y la sensibilidad, que el fabricante fija a 0,061°/s/LSB.

Bajo estas condiciones el valor de offset estará comprendido entre -31,23 °/s/*LSB* y +31,23 °/s/*LSB*.

Tabla 7: Resumen valores calibración giróscopo

	Media sin offset	Valor ideal	Valor de offset	Media tras offset
Eje X	133,51 c.d. / 0,5096 °/s	0 c.d. / 0,00 °/s	-8 c.d. / -0.0305 °/s	0,318 c.d. / 0,0012 °/s
Eje Y	122,65 c.d. / 0,4681 °/s	0 c.d. / 0,00 °/s	-8 c.d. / -0.0305 °/s	0,318 c.d. / 0,0012 °/s
Eje Z	-83,36 c.d. / 0,3182 °/s	0 c.d. / 0,00 °/s	+5 c.d. / 0.3050 °/s	-4,724 c.d. / 0,0180 °/s

5.3 MAGNETÓMETRO

Para el análisis de las medidas tomadas por el magnetómetro, éste se hace girar sobre su eje Z sobre una superficie horizontal. Al representar en una gráfica las medidas tomadas debe observarse una circunferencia centrada en el origen.

Debido a las perturbaciones magnéticas del entorno esto no suele ser así y por eso es necesario realizar una calibración para corregir los errores de *soft iron* y *hard iron*, como se comentó en el apartado 4.2.2.

Tras realizar la calibración y determinar los valores de *offset*, se vuelve a rotar el dispositivo en la misma posición y se analizan las medidas tomadas. El resultado de este análisis se refleja en la *Figura 28,* donde se representa en rojo las medidas en bruto y en azul las medidas tras aplicar los valores de offsets y se verifica una correcta calibración.

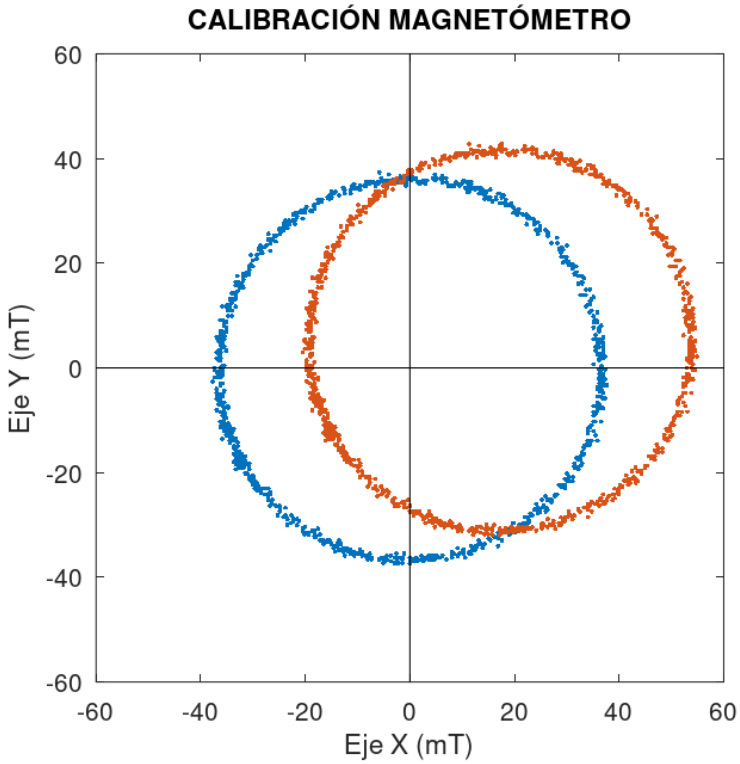

Figura 28: Calibración magnetómetro

Los valores concretos de ajuste de offset obtenidos en el proceso de calibración se indican en la *Tabla 8*. Es reseñable el valor de sigma de 1, ya que indica que la figura en rojo se acerca a una circunferencia perfecta y no requiere del ajuste *soft iron*, en el entorno de pruebas.

Este valor de sigma no es único del proceso de calibración concreto reflejado en esta memoria, si no que se han obtenido valores similares o idénticos en diversas calibraciones, indicando la buena calidad del integrado, así como su fuerte inmunidad a perturbaciones de tipo *soft iron*.

Tabla 8: Valores de calibración del magnetómetro

Hard iron		Soft iron
Alfa	**beta**	**sigma**
- 17.1964	- 5.4889	1.0000

Este proceso se podría ampliar al eje Z y calibrar el dispositivo tridimensionalmente, de manera que en vez de una circunferencia se observara una esfera perfecta. Sin embargo, debido a que la brújula implementada es plana, se considera irrelevante el valor medido en el eje Z.

Para terminar, se compara la posición de la maneta y el ángulo mostrado, con los mismos elementos de una brújula adicional.

5.4 DISPLAY

Para comprobar la correcta gestión de la pantalla es necesario verificar primero el apartado visual, segundo el apartado táctil, y, por último, la interrelación de ambos.

Se comienza verificando el protocolo de control para la gestión del controlador NV3035 a través de su interfaz serie 3-W.

En la hoja de características del controlador, el fabricante indica que el periodo de la señal de reloj tiene un valor mínimo de 320 ns, es decir, una frecuencia máxima de 3,125 MHz. Esta interfaz se usa únicamente al inicio de la aplicación, momento en el que se configura el *display*, realizando un total de 4 transacciones en toda la vida de la aplicación. Además, tras la configuración, se muestra por pantalla durante un segundo una imagen a color, a modo de test, por lo que tampoco es necesario una extrema rapidez en las transacciones.

Debido a esto, se considera innecesario implementar el protocolo maximizando la velocidad de las transacciones y acercando la señal de reloj a su frecuencia máxima. Por ello se establece la frecuencia a 45 kHz, obteniéndose un holgado margen en todos los tiempos indicados en la hoja de características.

Para verificar la correcta implementación del protocolo, se analiza la captura de la *Figura 29,* correspondiente al proceso de configuración inicial del controlador, consistente en:

- Una operación de lectura del registro R03, para conocer su contenido y modificar únicamente los bits pertinentes.
- Una operación de escritura del registro R03 cambiando sus 4 bits menos significativos a “1101”. De esta manera se establece el protocolo de envío de imágenes como DE_24.
- Una operación de lectura del registro R03 para comprobar que se actualiza correctamente su contenido.

Figura 29: Detalle protocolo 3-W

Los paquetes enviados constan de 16 bits, cuyo contenido y significado se indica en la *Tabla 9*.

Tabla 9: Contenido mensajes protocolo 3-W

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Contenido	Dirección registro						R/W	X	Datos							
Lectura1	0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0
Eescritura1	0	0	0	0	1	1	1	0	1	1	0	0	1	1	0	1
Lectura 2	0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	1

La segunda lectura confirma que se ha actualizado correctamente el registro deseado, de manera que queda validado el protocolo 3-W.

Tras verificar el analizar el protocolo 3-W, se continúa analizando el protocolo de envío de imágenes DE_24.

En el apartado de diseño 4.2.4 se indica que la implementación de este protocolo no es exacta a la indicada por el fabricante, sino que se ha adaptado de acuerdo a las limitaciones de la *Raspberry Pi* y el manejo del *gpio*.

En la *Figura 30* se representa el comportamiento real del protocolo en el instante del flanco de bajada de la señal de reloj, momento en el que se modifica el valor del resto de señales. Este análisis se ha realizado para el caso de tener un bus de datos de tres canales.

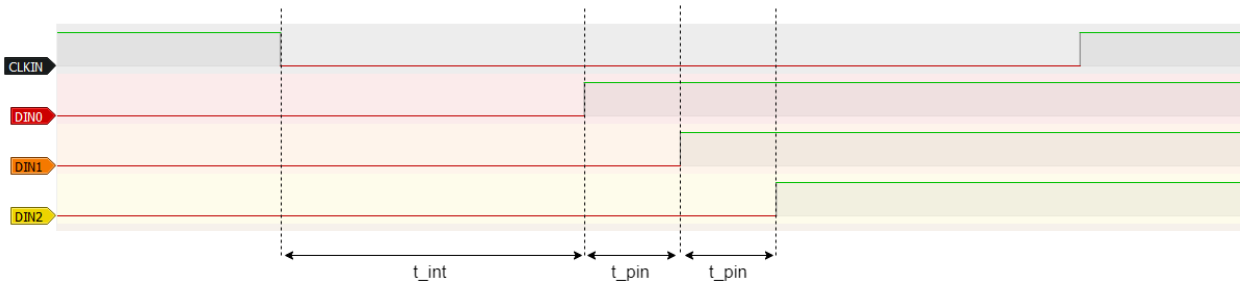

Figura 30: Representación comportamiento real protocolo DE_24

Se observa como la actualización de las señales no es exactamente coincidente con el flanco de bajada, sino que sufren una serie de retardos. Por un lado, se tiene el retardo debido al desfase en la detección del flanco de bajada (t_{int}), y, por otro lado, el retardo intrínseco de la operación de escritura en los pines (t_{pin}). Así, el retardo total se puede modelar como la ecuación (14), donde "nº pines" se corresponde con el número de salidas sobre las que se realiza una operación de escritura de manera secuencial.

$$Retardo\ total = t_{int} + t_{pin} * n^{\underline{o}}\ pines \tag{14}$$

Para determinar el valor del retardo en la operación de escritura, se realiza un bucle en el que únicamente se modifica el valor de 3 salidas de manera secuencial, alternando su valor. Tras realizar varias mediciones se concluye que el retardo de la operación de escritura es fijo y de del orden de los 230 ns, tanto para escritura de "1" como de "0".

Para determinar el valor del retardo en la detección, se debe medir el tiempo desde que ocurre el blanco de bajada hasta que se modifica el valor de la salida, y a ese valor restarle los 230 ns del tiempo de escritura.

Este retardo en la detección del flanco de bajada varía en función del método usado, habiendo sido los métodos analizados los siguientes:

- Rutina de atención a la interrupción (*ISR* por sus siglas en inglés):
  Este método consiste en lanzar una interrupción cada vez que se detecta un evento determinado en uno o varios pines objetivos (en este caso: flanco de bajada y pin de reloj). La interrupción corta el flujo del programa para ser la siguiente orden a ser ejecutada por el procesador. Es en ese momento en el que se activa el *flag* que indica que ha habido un flanco de reloj e inmediatamente da comienzo la actualización de los pines de salida.
  El análisis se ha realizado tanto a alto nivel, mediante las funciones de la librería *pigpio,* como a un nivel más bajo mediante implementando de la técnica de *polling*.
  En ambos casos el retardo oscila entorno a los 45 us.

- Alerta:
  Este método es similar al caso de la interrupción, pero con dos diferencias notables. Por un lado, en vez de saltar con la detección de un flanco indicado, salta cuando detecta un cambio en el valor del pin objetivo, sea cual sea este cambio. Esto implica que una de cada dos interrupciones se deberá obviar, y se habrá sufrido un retardo innecesario en la comprobación del flanco de subida. Por otro lado, al no tratarse estrictamente de una interrupción, el flujo de trabajo del software no se detiene de manera tan inmediata debido que no se trata con tanta prioridad.
  En este caso el retardo alcanza un valor de 1ms.

- Hilo dedicado:
  Este tercer método consiste en crear un hilo cuya única tarea es detectar el flanco de bajada. Dentro de un bucle se comparan constantemente dos lecturas consecutivas del pin del reloj. Si la primera lectura indica que está a nivel alto y la segunda a nivel bajo, se activará un *flag* hasta la siguiente iteración del hilo.
  De esta forma se obtiene un retardo del orden de 350 ns

- Bucle *while* interno:
  En este último caso, el propósito no es activar un flag para que, desde otro hilo o apartado del programa, se desencadene la actualización de los pines de salida. Sino que la detección del flanco de bajada se realiza desde la propia función que genera el protocolo, haciendo uso de dos sentencias *while*. En la primera, se bloquea la ejecución de la función hasta dejar de leer un nivel bajo (ha ocurrido un flanco de subida), para en la siguiente línea bloquear la ejecución hasta que se deje de leer un nivel alto, lo que indica la existencia de un flanco de bajada. Tras esto, la siguiente línea del código ya es la operación de escritura del pin de datos.
  De esta forma se obtienen retardos del orden de 240 ns.

Tras el análisis se determina que los dos últimos métodos son los que ofrecen mayor rapidez. En el diseño final se opta por el método 4, no sólo por ser el que menos retardo aporta, si no porque el método 3, tiene la desventaja de que ralentiza el resto de procesos. Esto es debido a que el hilo que detecta el flanco de bajada no puede tener esperas, lo que supone acaparar el

procesador la mayor parte del tiempo, negando el acceso al resto de hilos del programa. Los 110 ns de diferencia entre ambos métodos se explica por el retardo de las operaciones de comparación que hace el primero.

De esta manera, sumando el retardo en la detección del flanco de bajada, el retardo de la operación de escritura, y reduciendo el número de salidas a una, según la ecuación (14) se obtiene un retardo mínimo del orden de 470 ns.

Para el cálculo de la frecuencia máxima de reloj, a parte del valor del retardo, es necesario tener en cuenta el valor del tiempo de estabilización. Esto es el tiempo mínimo que debe pasar desde que se varía el valor de la salida de datos hasta que ocurre el flanco de subida. De esta manera se asegura que el *display* lee un valor correcto. El fabricante indica que este tiempo de estabilización mínimo es de valor 12 ns. Para un único pin de salida, la suma de todos los retardos es como se muestra en la captura de la *Figura 31*.

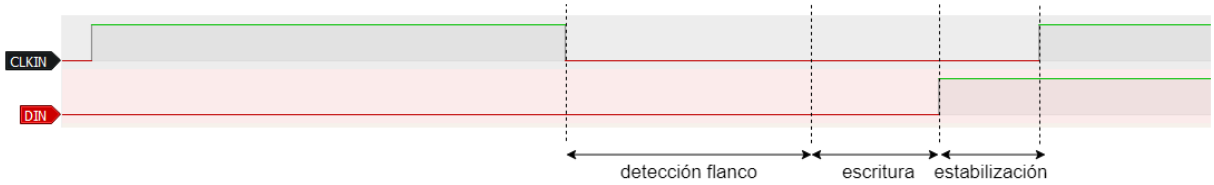

Figura 31: suma de retardos en protocolo DE

Dando un margen al tiempo de estabilización hasta llegar a los 20 ns, se obtiene un tiempo mínimo a nivel bajo de la señal de reloj de 490 ns, lo que implica una frecuencia máxima de 1 MHz para un ciclo del 50%.

Este valor dista del valor ideal, por eso se hace una última modificación para elevar la frecuencia de reloj. En lugar de detectar el flanco de bajada se busca detectar el flanco de subida. De esta manera se desplaza el retardo total, iniciándose en el flanco de subida, haciendo que la salida se actualice casi inmediatamente después del flanco de bajada. También se elimina la primera orden *while,* porque el retardo siempre va a ser mayor que medio ciclo de reloj. Así, el tiempo de estabilización será siempre suficiente, el valor de la salida siempre se leerá correctamente y la imagen se verá más nítida.

Tras esto, se establece una frecuencia de reloj de 1,7 MHz, pudiendo elevarse hasta los 2 MHz, obteniendo una forma de protocolo como la captura de la *Figura 32* .

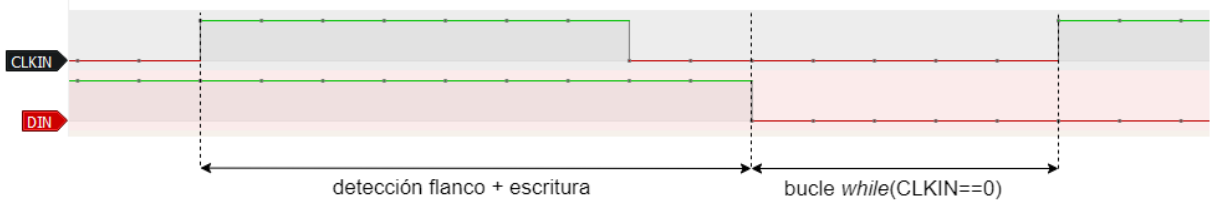

Figura 32: Tiempos en protocolo DE_24 a 1,7 MHz

Como nota aclaratoria, los tiempos se dan dentro de un orden porque al tratarse de una aplicación multihilo, la forma en que cada uno de ellos acapara el procesador e interrumpe al hilo previo es arbitraria e influye en la medida de los retardos.

Finalmente, es necesario indicar que otro factor que provoca malfuncionamiento del *display* es el ruido que se introduce en las señales a través de los cables, más al aumentar la frecuencia. Este ruido es difícil de eliminar, si no imposible, en un diseño sobre una tarjeta de prototipado. La forma de reducir este ruido sería mediante la implementación de la placa auxiliar en una *pcb*, apantallando las señales con planos de tierra y una correcta distribución de las mismas a lo largo de las capas.

Para terminar el análisis del diseño, resta comprobar el funcionamiento del apartado táctil del *display*.

Para esto, primero se verifica que se detectan correctamente los tres tipos de toques usados en el diseño: desplazamiento hacia la izquierda, desplazamiento hacia la derecha y toque único. En el apartado de diseño ya se demostró este correcto funcionamiento, ilustrado en la *Figura 13 y Figura 14.*

Una vez hecho esto, se verifica que se detectan y generan correctamente las coordenadas del toque. Para ello se realizan toques en distintos puntos de la pantalla hasta barrer todo el rango de valores x e y. Esto es, el rango de valores de la coordenada x de ir desde 0 hasta 320; y el rango de valores de la coordenada y desde 0 hasta 240, siendo la coordenada (0,0) la esquina superior izquierda y la coordenada (320,240) la esquina inferior derecha.

Tras esta prueba se observa que el funcionamiento del apartado táctil no es correcto, ya que únicamente detecta toques en la parte inferior de la pantalla. Se llega a la conclusión de que es un error en la fabricación de la pantalla táctil debido a que no se lanzan interrupciones para indicar un toque. Por este motivo se decide modificar para la interfaz gráfica de manera que la parte táctil de la pantalla se localice en la zona inferior. Será ahí donde se localicen los botones con los que el usuario interactuará.

Por último, se realiza un uso general del dispositivo para verificar la correcta integración de todos los bloques del diseño. Comprobando de esta manera que se pasa de manera correcta de pantalla en pantalla, que las coordenadas de los botones son las indicadas y que las acciones llevadas a cabo por cada uno de ellos son las esperadas.

6 IMPACTO DEL PROYECTO

Al tratarse de un proyecto sobre el análisis del rendimiento de un microcontrolador, el impacto más allá del ámbito académico o tecnológico no es notable. Dentro de este ámbito sí es de relevancia, sirviendo para conocer el límite en las prestaciones de una *Raspberry Pi,* y así tener un conocimiento más amplio, que será de utilidad a la hora de encontrar un dispositivo dentro del mercado actual que se ajuste a las necesidades de tu diseño.

En cuanto a la funcionalidad de unidad de medida inercial, en sí misma no supone un gran impacto, actualmente existen infinidad de dispositivo similares. Lo que es de relevancia es el uso futuro que se le pueda dar. Como sistema capaz de monitorizar la posición y velocidad, posibilita una conducción extremadamente eficiente, lo que permite, entre otros beneficios, una reducción en el consumo de combustible. Este punto entronca directamente con el objetivo 13 de los Objetivos de Desarrollo Sostenible (ODS) [14]. Este objetivo, denominado "Acción por el clima", en su meta número 3 indica la importancia de "mitigar el cambio climático, reduciendo sus efectos". Es necesario para ello reducir la dependencia y consumo de combustibles fósiles, con vistas a un futuro en el que las energías limpias abarquen todo el consumo energético, siendo incluso en ese punto necesario un uso eficiente de la energía para evitar derroches innecesarios.

Por otro lado, este proyecto también está en sintonía, con el objetivo 4 "Educación de Calidad". Es necesario que todo el mundo tenga acceso a una educación de calidad que le de las herramientas suficientes para tener igualdad de oportunidades en el mercado laboral. Del formato de este proyecto se puede extraer una lectura pedagógica, al detallar todo su diseño, desarrollo y posterior análisis, posibilitando así su replicabilidad. Además, los elementos usados, tantos los periféricos como la *Raspberry Pi,* son relativamente económicos. De esta manera, no es necesario disponer de un gran capital para poder alcanzar resultados similares. No sólo para una *Raspberry Pi*, si no para cualquier microcontrolador con similares características.

7 CONCLUSIONES

Este proyecto tenía el objetivo de diseñar una unidad de medida inercial sobre una *Raspberry Pi*, con el condicionante de realizar la gestionar los pines de entrada salida de uso general con un leguaje de alto nivel. Así se planteaba el problema de la idoneidad de este tipo de programación para aplicaciones exigentes, como la implementada.

El segundo condicionante era que los periféricos usados no estuviesen prediseñados para su uso con *Raspberry Pi,* sino que fuera necesario realizar un desarrollo hardware y software para su incorporación al proyecto.

A nivel hardware, se implementa mediante una tarjeta de prototipado una placa auxiliar que permite el acople y ensamblado de todos los elementos. En este punto cabe señalar la necesidad de usar un adaptador para convertir el conector de la pantalla de tipo *ffc* a tira de pines de 2,54 mm, como de los que disponen tanto los sensores como la *Raspberry Pi*.

En esta placa también se realiza el reparto de los correctos niveles de las alimentaciones, se implementan las interconexiones entre los elementos y las interfaces de control y comunicaciones. Las interfaces más relevantes son el canal *I2C* multiesclavo, que posibilita el control de los sensores y el apartado táctil del display sobre un único canal, la interfaz serie que configura la pantalla *LCD*, y la interfaz de transmisión de datos a la pantalla. Esta última, no se puede implementar en su totalidad debido a que no existen pines de salida suficientes en la Raspberry Pi para completar el *buffer* de 24 señales de datos. La solución llevada a cabo es reducir el número de señales totales a 3, limitando así el número de colores que se pueden generar a 8.

A nivel software, se desarrolla una aplicación multihilo en lenguaje C, haciendo uso de la librería de gestión de pines de alto nivel *pigpio*. Mediante una máquina de estados se controlan los sensores, se interpretan las medidas de estos, y se trasladan los datos a la pantalla táctil. Para el control de los sensores y la capa táctil se implementa el protocolo *I2C*. Para el control del *display* se implementa el protocolo serie (mediante la técnica de *bit banging*) y el protocolo de transmisión de datos. Este último sufre una adaptación adicional debido a que la frecuencia que indica el fabricante en los requisitos del protocolo es inmanejable por el *gpio*. La solución llevada a cabo consiste en reducir de nuevo el número de pines de salida a 1, y establecer la frecuencia de reloj en 1,7 MHz, en lugar de los 6,4 MHz que marca el fabricante. Esto supone la eliminación del color en las imágenes y un refresco de pantalla notable visualmente. Se llega a estos valores tras observar que el retardo sufrido en la escritura de los pines y en la detección de los flancos de la señal de reloj para alcanzar la sincronía son de relevancia cuando la frecuencia de reloj alcanza valores superiores a 1 kHz. Para encontrar la frecuencia de reloj máxima posible se realiza un análisis exhaustivo de los distintos métodos para la detección de eventos en los pines de entrada, y se determina cuál es el retardo en la operación de escritura. Se llega a la conclusión de que la frecuencia máxima de detección de eventos es de 2 MHz, más allá de ese valor pasarán algunos desapercibidos.

Tras finalizar el diseño, se someten a diversas pruebas específicas cada uno de los bloques del dispositivo para analiza su desempeño. Se comprueba que el control y configuración de los sensores se realiza correctamente y que los datos que de ellos se extraen, tras el proceso de

calibración, son los esperados. También se enfrentan estos datos a instrumentos de medida externos para verificar su calidad, observando un mínimo error en las medidas.

Del *display* se verifica que las imágenes mostradas se corresponden con las imágenes cargadas en la memoria. En relación con su apartado táctil, se analiza si se detecta correctamente el tipo de toque y sus coordenadas. Se observa que se detecta correctamente pero únicamente en la parte inferior de la pantalla, debido a un fallo de fabricación. Esto obliga a modificar la interfaz de usuario para que la zona táctil se limite a esa zona. Adicionalmente, se verifica que cada uno de los gestos y botones táctiles desencadenan la acción asociada correctamente.

Por último, se hace un uso general del dispositivo para comprobar la correcta integración de todos los elementos y sus funcionalidades.

Tras este exhaustivo proceso de análisis, se llega a la conclusión de que se cumplen los objetivos marcados. El dispositivo funciona como una unidad de movimiento inercial en su completitud. Incluso, el *display*, a pesar de la imposibilidad de implementar un protocolo acorde a sus requisitos, emite imágenes nítidas a una frecuencia inferior a la ideal.

El análisis del rendimiento de los pines de entrada y salida determina que su manejo a alto nivel provoca una serie de retardos que son significativos para diseños con tiempos de ejecución elevados. En aplicaciones que requieran detección de eventos en las entradas de manera inmediata, actualizar el valor de las salidas a gran velocidad, o una sincronía perfecta entre estas operaciones, no es posible su uso.

Este proyecto ha servido de aprendizaje para exponer la importancia de hacer una buena gestión del tiempo. Es imprescindible conocer y respetar los tiempos marcados, así como analizar los retardos que genera el procesador en las distintas operaciones, para poder implementar un código eficiente. En aplicaciones multihilo, la gestión del tiempo requiere de especial cuidado, no solo para la implementación de protocolos si no para asegurar un uso del procesador equilibrado entre los distintos hilos y evitar así bloqueos en la ejecución del programa.

Para finalizar, se plantean dos alternativas mejorar en un futuro este diseño. Por un lado, se recomienda el uso de una *fpga* haciendo las veces de intermediario entre el *display* y la *Raspberry Pi*. Ésta enviaría una única vez la imagen que se quisiera mostrar en la pantalla a la *fpga*, para que la almacenara en sus registros y la enviara al *display* de manera continua. Las altas frecuencias que caracterizan a los relojes de las *fpgas* y su capacidad para actualizar las salidas de manera paralela permitirían la implementación del protocolo en su totalidad, tanto en número de pines como frecuencia de señal.

La segunda alternativa sería el desarrollo una *pcb* de la placa auxiliar diseñada, en lugar de su montaje como tarjeta de prototipado. Realizando un diseño centrado en el apantallamiento y aislamiento de las señales de alta frecuencia se reduciría considerablemente el ruido introducido en el sistema.

8 REFERENCIAS BIBLIOGRÁFICAS

[1] Majid Dadafshar, «analog,» 2014. [En línea]. Available: https://www.analog.com/media/en/technical-documentation/tech-articles/accelerometer-and-gyroscopes-sensors-operation-sensing-and-applications.pdf. [Último acceso: 2022].

[2] «electronics-tutorial,» [En línea]. Available: https://www.electronics-tutorials.ws/electromagnetism/hall-effect.html. [Último acceso: 2022].

[3] «raspberrypi,» [En línea]. Available: https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/. [Último acceso: 2022].

[4] «buildroot,» [En línea]. Available: https://buildroot.org/. [Último acceso: 2022].

[5] M. Ruiz y S. Esquembri, «Embedded Linux Systems,» Madrid, 2021.

[6] «abyz,» [En línea]. Available: https://abyz.me.uk/rpi/pigpio/. [Último acceso: 2023].

[7] «raspberrypi,» [En línea]. Available: https://www.raspberrypi.com/documentation/computers/linux_kernel.html. [Último acceso: 2023].

[8] «kernel,» [En línea]. Available: https://www.kernel.org/doc/html/latest/index.html#. [Último acceso: 2022].

[9] «texasinstruments,» [En línea]. Available: https://www.ti.com/tool/BOOSTXL-SENSORS. [Último acceso: 2022].

[10] «rs-online,» [En línea]. Available: https://es.rs-online.com/web/p/displays-en-color-lcd/9156458. [Último acceso: 2022].

[11] C. J. Fisher, «analog,» 2010. [En línea]. Available: https://www.analog.com/media/en/technical-documentation/app-notes/an-1057.pdf. [Último acceso: 2022].

[12] M. Christopher Konvalin, «fierceelectronics,» 2009. [En línea]. Available: https://www.fierceelectronics.com/components/compensating-for-tilt-hard-iron-and-soft-iron-effects. [Último acceso: 2022].

[13] N. Liesch, «THE BMP FILE FORMAT,» [En línea]. Available: https://www.ece.ualberta.ca/~elliott/ee552/studentAppNotes/2003_w/misc/bmp_file_format/bmp_file_format.htm. [Último acceso: 2022].

[14] ONU, «un,» [En línea]. Available: https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/. [Último acceso: 2023].

ANEXO 1. ESQUEMÁTICOS Y DETALLES DE MONTAJE

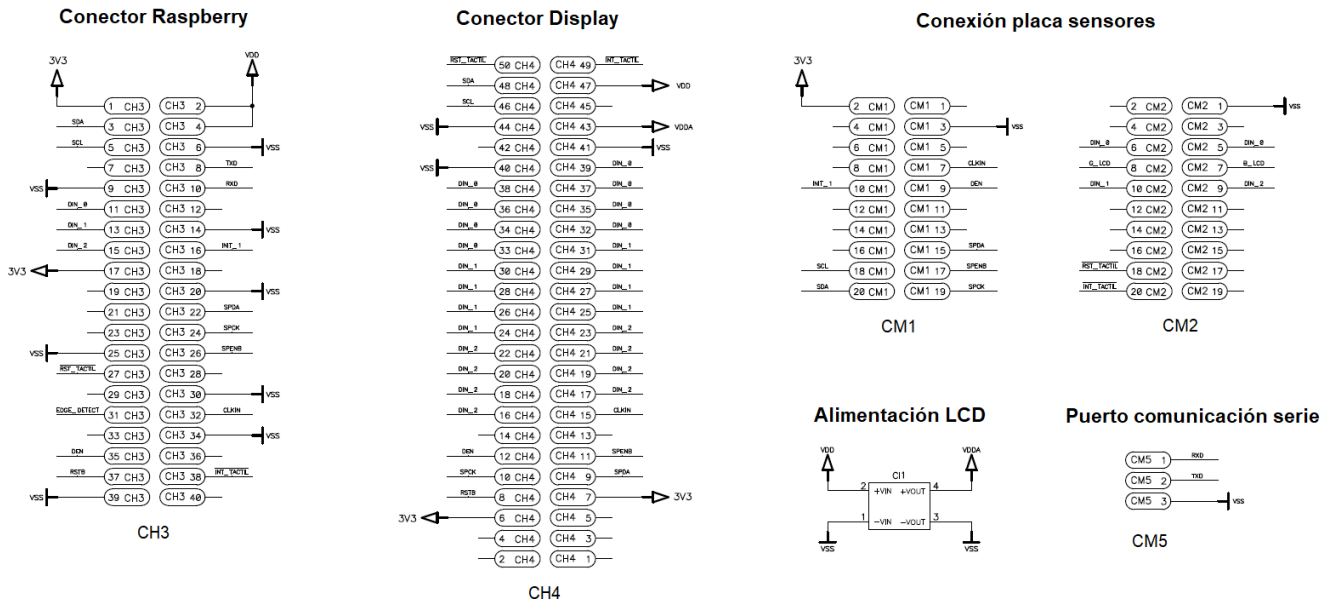

Figura 33: Esquemáticos placa auxiliar

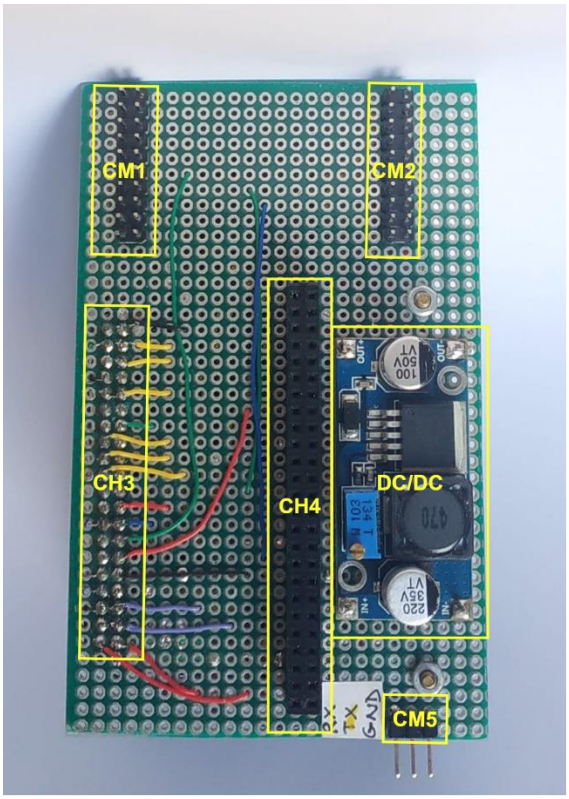

Figura 34: Detalle montaje placa auxiliar

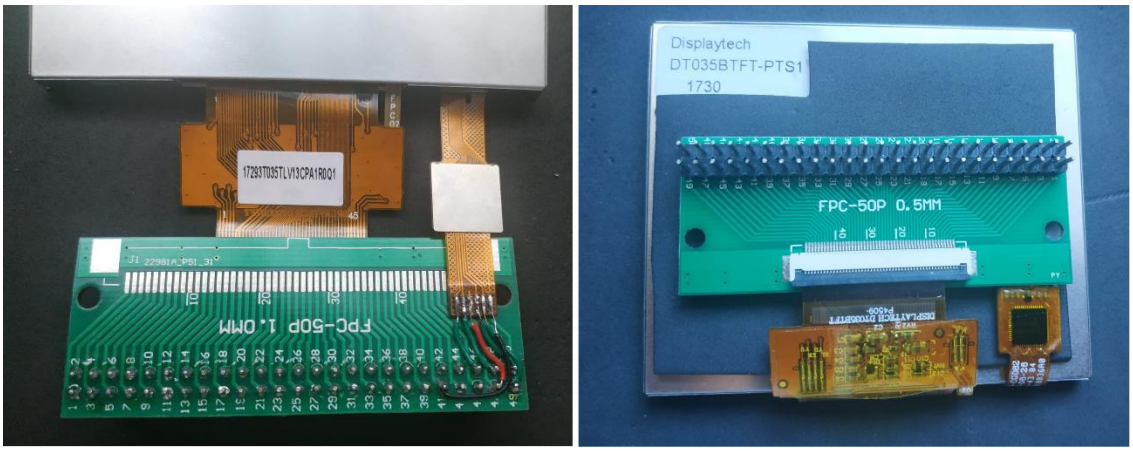

Figura 35: Detalle adaptador conector ffc a tira 24 pines

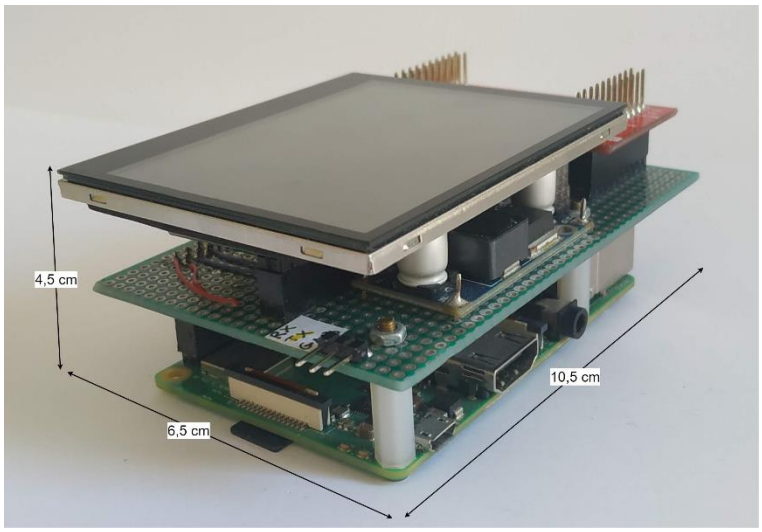

Figura 36: Detalle dimensiones montaje completo

ANEXO 2. PRESUPUESTO

En la *Tabla 10* se refleja el desglose del presupuesto necesario para este proyecto, con duración de un año.

Tabla 10: Desglose del presupuesto total

CONCEPTO	PRECIO (€)
Placa Sensor BoosterPack (TI)	31,23
Raspberry Pi 3B+	78,00
Display DT035BTFT-PTS1	28,48
Regulador ajustable	3,00
Placa de prototipado	2,27
Mano de obra	32.000 (ingeniero/año)
TOTAL	32.142,98

ANEXO 3. RECURSOS USADOS

En la *Tabla 11* se indican las herramientas usadas en el desarrollo de este proyecto y el objetivo de su uso.

Tabla 11: Recursos usados en el diseño del proyecto

Herramienta	Necesidad cubierta
Buildroot	Configuración y generación del sistema operativo empotrado Linux dentro de la *Raspberry Pi.*
Putty	Comunicación serie entre la *Raspberry Pi* y el ordenador.
Eclipse	Programación en lenguaje C del software de la aplicación.
Octave	Análisis a posteriori de los datos extraídos de los sensores.
PulseView y analizador lógico	Visualización y análisis de los distintos protocolos y señales.
Photoshop	Generación de imágenes píxel a píxel con tamaño fijo de 240x320 en formato bmp.
PCAD	Diseño de los esquemáticos de la placa auxiliar.
VMware	Generar y cargar la máquina virtual con sistema operativo Linux sobre la que se ha desarrollado el proceso de diseño.
Github	Alojar código del proyecto
Herramientas propias de laboratorio de electrónica: multímetro, soldador, etc	Montaje, ajustes, medidas, comprobaciones…

ANEXO 4. MANUAL DE USUARIO

Para iniciar la aplicación se debe entrar al directorio en el que se encuentra el archivo *imu* (junto a las carpetas *registro* y *media*) y ejecutar el comando: *./imu.*

Mientras se realiza la configuración e inicialización, el *display* mostrará una serie de colores tras lo cual se mostrará la pantalla inicial (*Figura 37*).

Figura 37: Pantalla inicial

La zona inferior marcada en azul es el apartado táctil del *display*, donde se realizan todas las interacciones. Desplazando el dedo en esa zona hacia la izquierda o derecha se avanzará entre las distintas pantallas de manera cíclica de la forma: pantalla principal ↔ acelerómetro ↔ giróscopo ↔ magnetómetro ↔ pantalla principal, como se indicó en el apartado 4.2.

En la *Figura 38* se muestra la pantalla correspondiente al acelerómetro, donde los puntos numerados indican:

1. Datos de la aceleración del dispositivo en cada eje.
2. Datos del ángulo de inclinación del dispositivo en cada eje.
3. Botón táctil para alternar el formato en que se muestran los datos de aceleración: unidades “g” o el dato numérico sin tratar que se extrae del sensor.
4. Botón táctil para alternar el formato en que se muestran los datos de ángulo de inclinación: radianes o grados.
5. En esta zona se muestra la tasa de muestreo y el rango de medida.
6. Botón táctil para entrar al menú de configuración.
7. Botón táctil para pasar al menú de calibración.
8. Representación del sentido y dirección de los ejes del dispositivo.

Los puntos 1, 3, 5, 6, 7 y 8 son equivalentes para la pantalla del giróscopo.

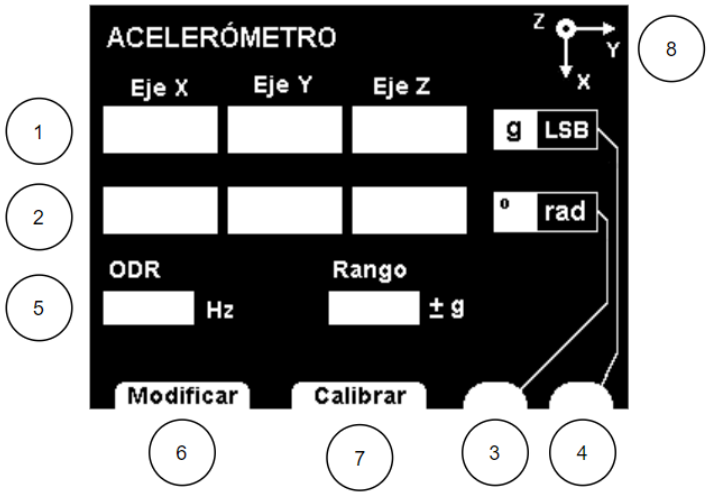

Figura 38: Pantalla acelerómetro

En la *Figura 39* se muestran las pantallas de calibración y configuración del acelerómetro, donde los puntos numerados indican:

1. Datos de la aceleración del dispositivo en cada eje.
2. Valores de offset tras la última calibración.
3. Botón táctil para volver a la pantalla del acelerómetro
4. Botón táctil para comenzar el proceso de calibración.
5. Botón táctil para alternar el formato en que se muestran los datos de aceleración: unidades “g” o el dato numérico sin tratar que se extrae del sensor.
6. Botón táctil para alternar el formato en que se muestran los datos de offset: unidades “mg” o el dato numérico sin tratar que se extrae del sensor.
7. Botones táctil para desplazarse entre los distintos valores de tasa de muestreo.
8. Botón táctil para guardar en los registros del sensor los valores seleccionados y hacer efectiva la configuración.
9. Botones táctil para desplazarse entre los distintos valores de rango.

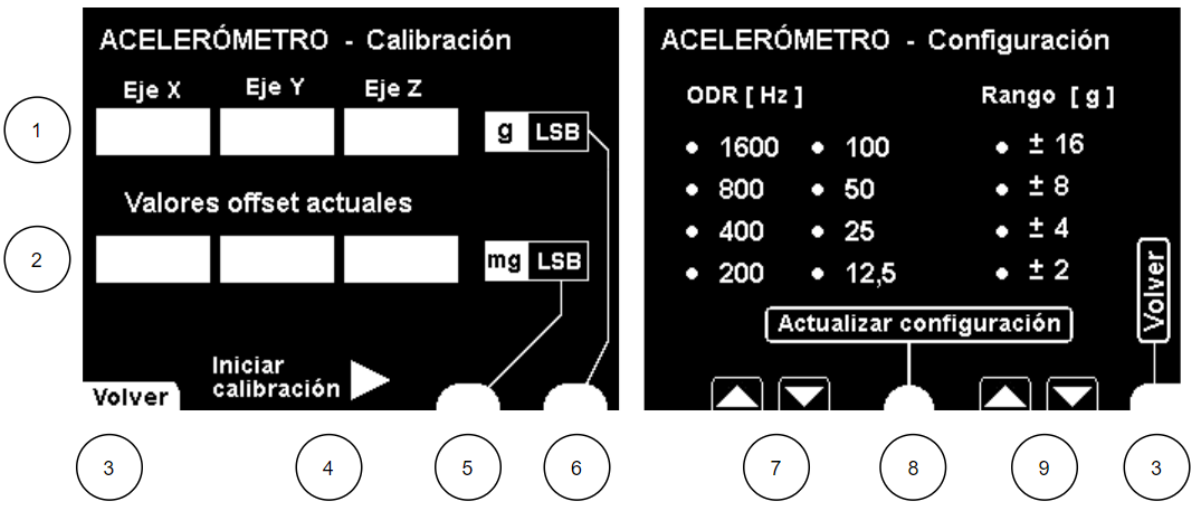

Figura 39: Pantallas de calibración y configuración

Todos los puntos anteriores son equivalentes para las pantallas de configuración y calibración del giróscopo, en sus respectivas unidades.

Por último, en la Figura 40 se muestra la pantalla del magnetómetro, donde los puntos numerados indican:

1. Datos de intensidad de campo magnético en los tres ejes del dispositivo
2. Botón táctil para comenzar el proceso de calibración. El usuario debe seguir los pasos que se indicarán en la pantalla.
3. Representación del sentido y dirección de los ejes del dispositivo.
4. Brújula digital donde el punto negro indica la dirección del norte magnético terrestre.

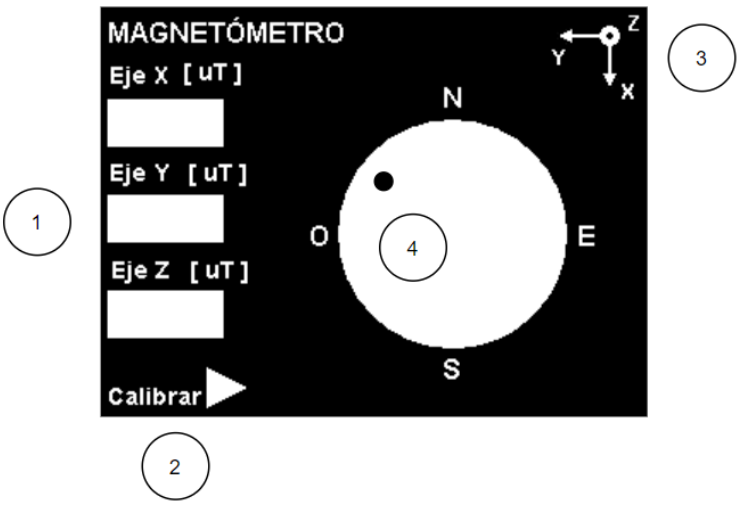

Figura 40: Pantalla del magnetómetro

Para finalizar la aplicación ejecutar el comando *ctrl+c*.

Por último, los ficheros en los que se ha almacenado las medidas tomadas de cada sensor se encuentran dentro de la carpeta *registros*, en los archivos *registro_acc.txt*, *registro_gyr.txt* y *registro_mag.txt*.

ANEXO 5. CÓDIGO DE APLICACIÓN

En el siguiente enlace, se aloja el proyecto completo con el código fuente que se ha desarrollado para este proyecto.

https://github.com/PFuentetaja/PFG_PFP/tree/main