Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

MASTER EN INGENIERÍA INDUSTRIAL

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

TRABAJO FIN DE MÁSTER

Sistema de monitorización de bajo coste para el registro de desplazamientos, fuerzas y aceleraciones en maquetas de estructuras

Autor: D. David Tuñón Cabeza
Tutor: D. Antolín Lorenzana Ibán

Valladolid, junio, 2016

Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

MASTER EN INGENIERÍA INDUSTRIAL

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

TRABAJO FIN DE MÁSTER

Sistema de monitorización de bajo coste para el registro de desplazamientos, fuerzas y aceleraciones en maquetas de estructuras

Autor: D. David Tuñón Cabeza
Tutor: D. Antolín Lorenzana Ibán

Valladolid, junio, 2016

Resumen

En el presente TFM se tratan los siguientes puntos:

1- Viabilidad de un sistema de bajo coste para el monitoreo de parámetros en estructuras, como desplazamientos, fuerzas y aceleraciones.
2- Diseño y desarrollo hardware para la medición de estos parámetros, donde se ha seleccionado la instrumentación necesaria para la correcta puesta a punto de los sensores. En cuanto al registro de estos datos se ha utilizado un ordenador de placa reducida como es la Raspberry Pi.
3- Diseño y desarrollo software para el registro de las variables en una base de datos y su posterior visualización en una aplicación de escritorio desarrollada con javaFX, la cual tiene portabilidad a cualquier tipo de dispositivo.

El prototipo resultante se ha probado en maquetas estructurales, disponibles en el laboratorio de esta escuela. Asimismo, se han comparado ciertos registros con los obtenidos mediante equipos profesionales. Tras los trabajos realizados se evalúa la posible implantación en estructuras reales.

Palabras clave

Raspberry pi, acelerómetro, célula de carga, calibre digital, JavaFX

Summary

In this TFM are treated the following points:

1- Viability of a low cost system to monitor structural parameters as displacement, strength and acceleration.
2- Hardware design and development to measure these parameters, the required instrumentation has been selected for the correct tuning of these sensors. For data recording has been used a single-board computer, as the Raspberry Pi.
3- Software design and development to record these parameters into a database and then display in a desktop application, this application could be port to new devices.

The resulting prototype has been tested on structural models, available in the laboratory of structures of this school. Likewise has been compared these registers with obtained using profesional equipment. After the work done, the possible implementation in real structures is evaluated.

Keywords

Raspberry Pi, accelerometer, load cell, digital caliper, JavaFX

Agradecimientos

Agradecer el apoyo de mi familia que siempre me han ayudado para estudiar el Master y seguir adelante.

Agradecer también a mi tutor D. Antolín Lorenzana Ibán y a José Pereda Llamas por brindarme la oportunidad de realizar este proyecto con el que he aprendido áreas nuevas.

Y por último agradecer a mis amigos que siempre han estado apoyándome para realizar trabajos de este tipo.

Contenido

1. INTRODUCCIÓN

1.1. Justificación y motivaciones

Este proyecto ha surgido como continuación de un proyecto fin de carrera previo desarrollado en el departamento, cuyo objetivo era registrar las medidas de fuerza y desplazamiento que se adquirían mediante sensores comerciales de bajo coste, concretamente calibres y pesamaletas portátiles digitales. Tras este antecedente se ha planteado un objetivo mucho más ambicioso, extendiendo el número y tipo de sensores, incrementando la frecuencia de muestreo y dotando de capacidad de comunicación entre los distintos centros de registro.

En la actualidad ya existen equipos que realicen esta tarea, pero el principal inconveniente es el precio. Estos equipos se componen de unidades de adquisición de datos y sensores. Para el presente proyecto lo que se analiza es si existen sistemas de bajo coste para monitorizar estructuras en prácticas de laboratorio.

Tras la puesta en marcha del sistema se podría evaluar su rendimiento y su viabilidad para su instalación en estructuras reales. Si el coste no fuera un impedimento, muchas estructuras de cierta responsabilidad (como puentes, aerogeneradores, centrales de producción de energía, etc.) podrían incorporar este tipo de sistemas que permitieran detectar posibles necesidades de mantenimiento antes de fallos catastróficos.

1.2. Objetivos

El objetivo principal de este proyecto es desarrollar un sistema de bajo coste para monitorizar ciertos parámetros físicos en una maqueta estructural, realizar prácticas con los alumnos para que visualicen como responde la estructura y poder realizar mediciones dinámicas de problemas de vibraciones.

Lo que se plantea con este proyecto es medir parámetros tanto en tiempo real como el almacenamiento de estos para un posterior análisis. Se busca tener un sistema con conectividad a internet para el fácil acceso a estos datos.

Otro objetivo, más a futuro de este proyecto, es la posibilidad de introducir este sistema en estructuras reales con una conexión a internet mediante tarjetas 3G o 4G, ofreciendo así sistemas de bajo coste para un análisis predictivo de estructuras.

1.3. Monitoreo de estructuras

Existen diversos parámetros que se pueden monitorizar en estructuras como aceleración (vibraciones), desplazamientos, fuerzas y variables ambientales. Los parámetros se consiguen de una red sensorial, desde la cual se obtienen indicadores que permiten detectar anomalías (daños o degradación) en la estructura. Esta información puede ser periódica o en tiempo real y estar asociada a cambios diversos como deterioro, corrosión, fatiga, reacciones químicas, humedad, cambios en las variables ambientales, así como las propiedades físicas relativas a la carga, esfuerzos, desplazamientos, deformaciones, aceleraciones, agrietamientos, vibraciones y otros.

En función del tipo de estructura se puede seleccionar un equipo con registro en tiempo real y visualización remota o simplemente un equipo que registre los datos para realizar un análisis posterior.

Estos equipos disponen de la instrumentación necesaria para el acondicionamiento de las señales de los sensores, sistemas de adquisición de datos, almacenamiento de estos y conexión a red si el equipo es de visualización remota.

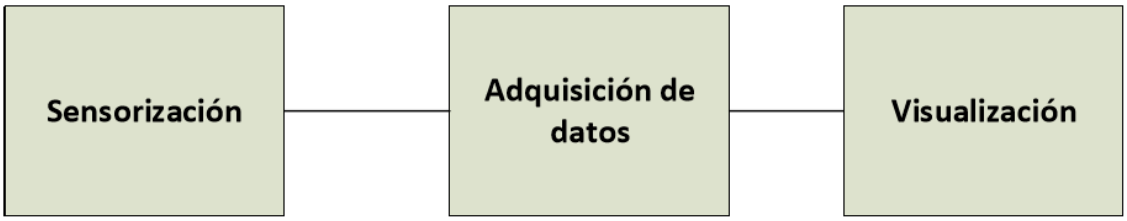

Figura 1.1

1.4. Estructura del documento

Este documento contiene 7 capítulos, de los cuales éste presenta una introducción de lo que se ha realizado en el trabajo fin de máster y unos objetivos fijados para el mismo.

El capítulo dos describe la situación o estado del arte de los sistemas de medición de parámetros en estructuras que existen en la actualidad.

En el tercer capítulo se abordará todo lo relacionado con el diseño hardware para el desarrollo de este proyecto, con lo que describirá el sistema completo a nivel físico.

El capítulo cuatro trata de dar una idea general de la arquitectura software diseñada para este sistema, así como la descripción y manual de uso de la aplicación de escritorio para la visualización de los registros de los sensores.

El capítulo número cinco es una discusión de los resultados obtenidos entre un equipo de alto coste frente a el equipo de bajo coste diseñado en este trabajo fin de máster.

En el capítulo seis se muestra el estudio económico relativo al proyecto.

Y en último lugar, en el capítulo siete, se expondrá una serie de conclusiones obtenidas con el desarrollo del proyecto, así como la descripción de líneas futuras para continuar el trabajo fin de máster.

2. ESTADO DEL ARTE

En este capítulo se van a exponer los diferentes sistemas que existen en la actualidad para la monitorización de estructuras, así como los sensores empleados en estos, este análisis se realiza principalmente para estructuras reales, aunque también se exponen sistemas utilizados en maquetas de estructuras.

2.1. Monitorización de estructuras reales

El monitoreo estructural o la monitorización de salud estructural (SHM) es una técnica para controlar la seguridad, integridad y rendimiento de una estructura. Esta técnica pretende dar en cada momento de la vida de la estructura el estado de los materiales constitutivos de diferentes partes, y del conjunto completo de estas partes que constituyen la estructura en su conjunto.

El estado de la estructura puede verse alterado por el envejecimiento debido al uso, por las acciones medioambientales y por eventos accidentales. Gracias a la monitorización de estas somos capaces de tener una base de datos de la historia de la estructura completa, esta nos puede dar un pronóstico de la evolución del daño, vida residual, etc.

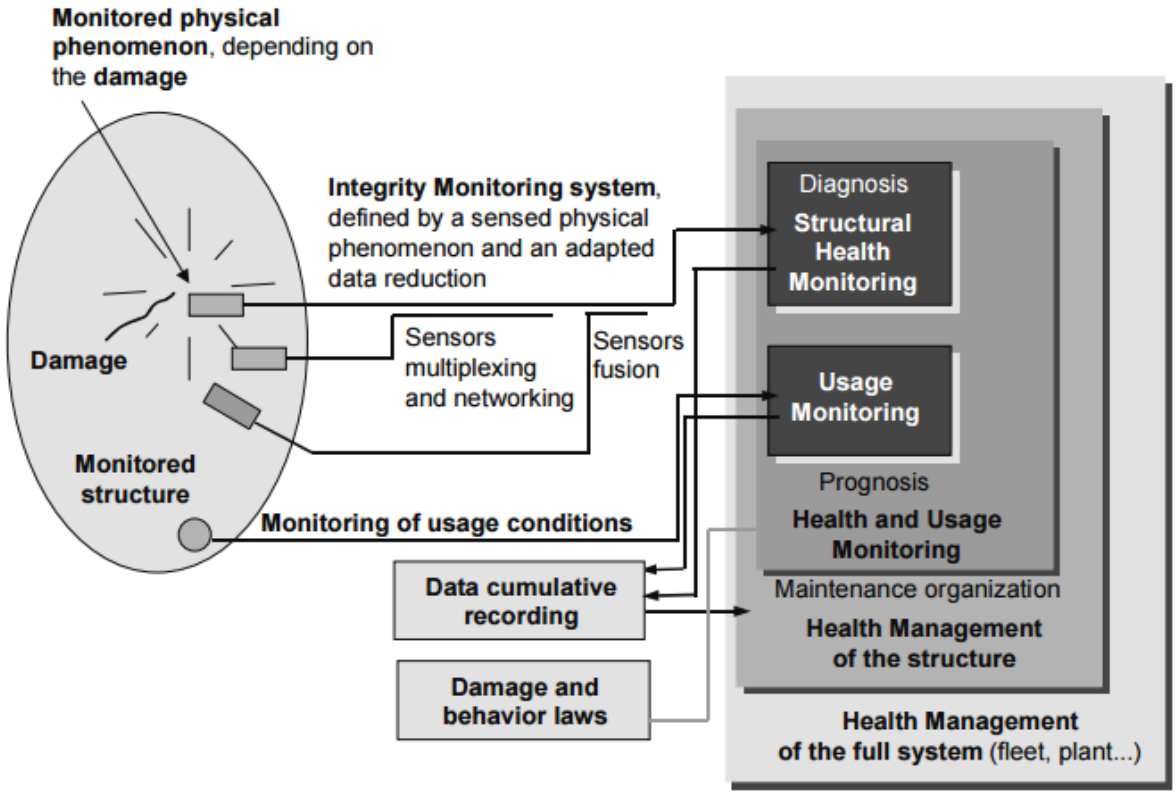

Figura 2.1. Esquema de un sistema de monitorización de salud estructural

En la figura 2.1, se muestra la organización de un sistema SHM típico en detalle. En la sección izquierda se ve lo que corresponde a la monitorización de la estructura definido por el tipo de fenómeno físico que daña la estructura (medido con un sensor), el tipo de fenómeno físico que es usado por el sensor para producir la señal enviado al sistema de adquisición de datos. Pueden existir varios sensores del mismo tipo, por lo que muy a menudo se tenga que multiplexar la señal.

También se pueden introducir otros sensores monitorizando las condiciones ambientales mejorando así la función de monitoreo del sistema. Las señales

generadas por el sensor junto con las registradas en el histórico son usadas por el microcontrolador para crear un diagnóstico.

Juntando la información del subsistema de monitoreo con el conocimiento basado en los daños mecánicos y las leyes de comportamiento se puede realizar un pronóstico y una gestión de vida de la estructura. (Balageas, Fritzen, & Güemes, 2006)

En definitiva, un sistema de monitoreo de salud de estructuras consta de cinco aspectos como:

- Sensores
- Generación de una señal
- Procesamiento y transmisión de la señal
- Interpretación y identificación del evento generado
- Integración del sistema para la gestión de la vida de la estructura

Un SHM no solo tiene que ser capaz de almacenar y mostrar la respuesta de los sistemas, sino de caracterizar la respuesta y compararla con un modelo apropiado. (KARBHARI, 2009)

Personas	**Información**
Ingenieros y diseñadores	Detalles de diseño
Especialista de materiales	Vida-servicio y respuesta esperada
Analista mecánico	Umbrales
Especialista IT	Detalles de los sensores
Especialista de sensores	
Modeladores	
Ciencia y tecnología	**Implementación**
Sensores	Redes
Conectividad a la red	Selección de datos y almacenamiento
Adquisición de datos	Evaluación
Algoritmos de daño en la estructura	Metodologías de diseño

Tabla 2.1. Elementos críticos de un sistema SHM

El impacto económico de colocar sistemas de medida en estructuras no es tan alto como parece ya que el ahorro en tiempo de mantenimiento o en construcciones similares posteriores es importante.

Las motivaciones para crear sistemas de este tipo son:

- Permitir un uso óptimo de la estructura, minimizando tiempos de inactividad y evitando fallos catastróficos.
- Dar al constructor una mejora de sus productos. Incluso durante la ejecución de una obra se pueden usar estos sistemas para detectar posibles desviaciones del diseño proyectado
- Reducir los tiempos de mantenimiento.

2.2. Sensores utilizados en monitoreo de estructuras

En este apartado se van a describir los diferentes tipos de sensores más usados en monitoreo de estructuras.

2.2.1. Sensores de desplazamiento

En cuanto a sensores de desplazamiento, los más utilizados son los medidores laser de desplazamiento por triangulación. Estos sensores entran dentro de la categoría de sensores sin contacto para medida de distancias y desplazamientos. El principio de funcionamiento de estos sensores es el que se muestra en la figura 2.2. A continuación se explica el funcionamiento del sensor.

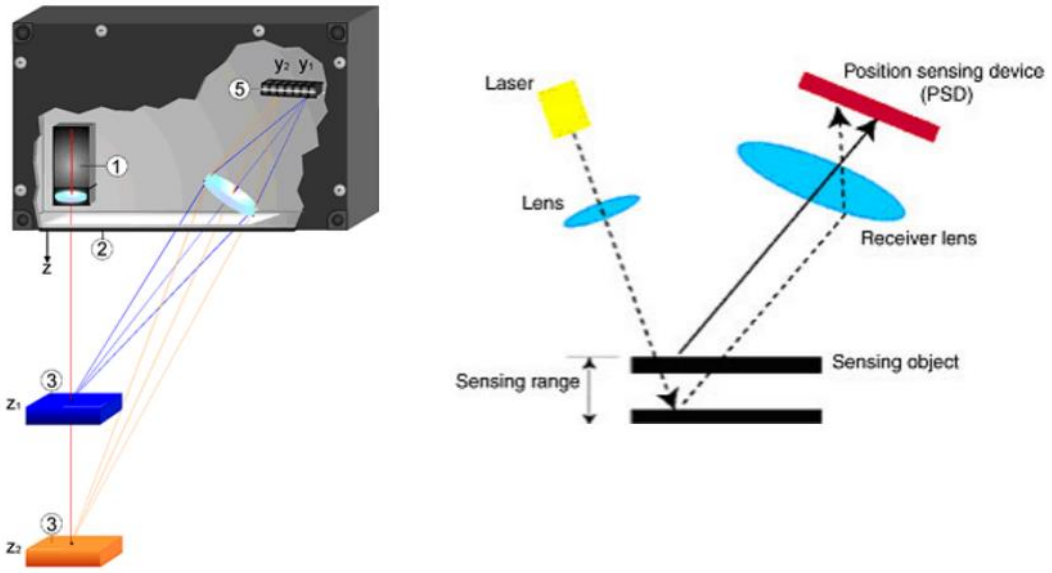

Figura 2.2. Sensor de desplazamiento laser por triangulación.

Se emite una luz con el láser mostrado a la izquierda del objeto de la figura, este se refleja en el objeto a medir y se proyecta sobre una cámara de matriz lineal (array CMOS), en la cual se puede conocer la posición donde ha incidido, con lo que se obtiene el desplazamiento del objeto deseado.

La posición del punto de la imagen proyectada en los pixeles de la cámara es transformada a un valor de distancia al objeto. Los dispositivos de triangulación son interesantes para medir distancias con alta precisión.

Los sensores de triangulación pueden ser difusos o especulares. La necesidad de dos tipos de sensores surge de las diferentes características de reflectancia de los materiales. Superficies suaves como los espejos son especulares, otras como el aluminio son difusas.

Superficies suaves o brillantes generalmente necesitan un sensor especular, mientras que superficies con una alta dispersión de la luz se miden mejor con sensores difusos. Muchas superficies tienen ambas componentes difusas y especulares por lo que es más difícil la elección del sensor adecuado, en estos casos se realiza mediante experimentos.

Los subsistemas del sensor son:

- Transmisor: Este suele ser un diodo laser con óptica de haz, proyecta un haz

que ilumina el objeto. El transmisor más usado en la actualidad es de baja potencia (670 nm).

- Receptor: este es el que recoge la luz reflejada en el objeto a medir, el detector envía la posición del punto al procesador. Existen varios tipos de detectores ópticos, los dos más usados son: detector de posición sensible (PSDs) y arrays.
- Procesador: la electrónica varía en función del tipo de sensor usado. Los receptores suelen dar dos corrientes de salida que deben ser convertidas a tensión y posteriormente a una señal digital con un conversor A/D. (Sensors Mag, 2016)

2.2.2. Sensores de fuerza

Los sensores de fuerza más utilizados en monitoreo de estructuras son las galgas extensométricas, estos sensores miden la deformación, se basan en el efecto piezorresistivo, propiedad que poseen ciertos materiales para cambiar su resistencia en función del esfuerzo al que es sometido. En la siguiente tabla se muestran los tipos más usados en SHM.

Dispositivo	Propósito	Observaciones
Galga extensométrica	Medir tensiones las cuales son formadas por deformaciones resultado de flexión, torsión, cizalladura y alargamiento/contracción.	La galga debe ser estable con respecto al tiempo y temperatura, debe dar una salida lineal con respecto al rango de tensiones a medir.
Cable vibrante (VWSG)	Es un sensor de tipo inductivo, está posicionado por encima de un alambre de acero inoxidable pretensado que cuando esta tensado mide la frecuencia de las vibraciones resultantes. La teoría de las vibraciones dice que las aceleraciones están relacionadas con la tensión.	Solo son adecuados para medidas estáticas, se ven afectados por las vibraciones ambientales. Si son instalados en el exterior necesitan de protección especial.
Galga extensométrica eléctrica (ESG)	Están basadas en el principio de que bajo una tensión mecánica varia la resistencia de un conductor. Esto se coloca en un puente Wheatstone con otras tres resistencias conocidas como se muestra en la figura 2.4. Como la resistencia	Se ven afectadas por el ruido eléctrico.

	cambia, la corriente del puente Wheatstone también.	
Fibra óptica de Bragg (FBG)	Fibras ópticas, las cuales son fibras finas (de unos pocos µm) de cristal o sílice. Se utiliza las propiedades de la fibra para generar señales optoelectrónicas indicando los parámetros externos a medir.	Las fibras son muy frágiles, por eso limita instalarlas en estructuras civiles. Esta técnica es relativamente cara con respecto a los sistemas convencionales.

Tabla 2.2. Tabla descripción de sensores de fuerza. (Sethi, 2016)

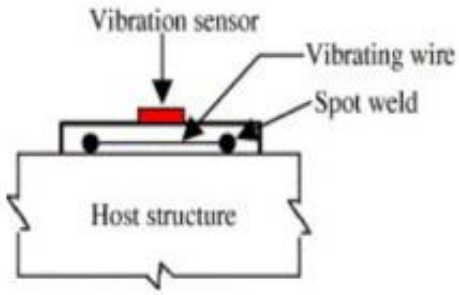

Figura 2.3. Cable vibrante. (Sethi, 2016)

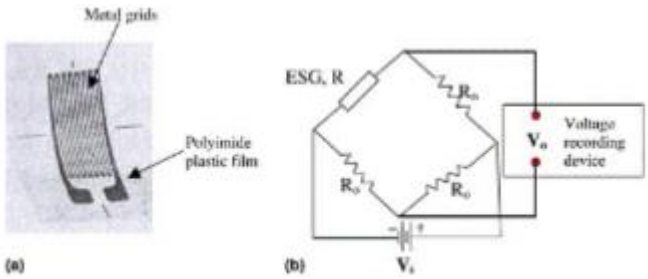

Figura 2.4. Galga extensométrica eléctrica. (Sethi, 2016)

En cuanto a la técnica de la detección óptica FBG, es una excelente solución ante limitaciones inherentes como perdida de transmisión y susceptibilidad a interferencias (ruido) que tienen los sensores eléctricos.

Este sistema refleja una longitud de onda de luz que se transmite en respuesta a variaciones en temperatura y/o tensión. Los FBGs están hechos usando interferencia holográfica o una máscara de fase para exponer un tramo corto de fibra fotosensible a una distribución periódica de intensidad de luz.

Una fuente ilumina el FBG (o múltiples si existen), la onda de luz reflejada pasa a través de un elemento dispersivo que distribuye los diferentes componentes de longitud de onda de la reflexión a distintas ubicaciones en un sensor CCD lineal como se muestra en la figura 2.6

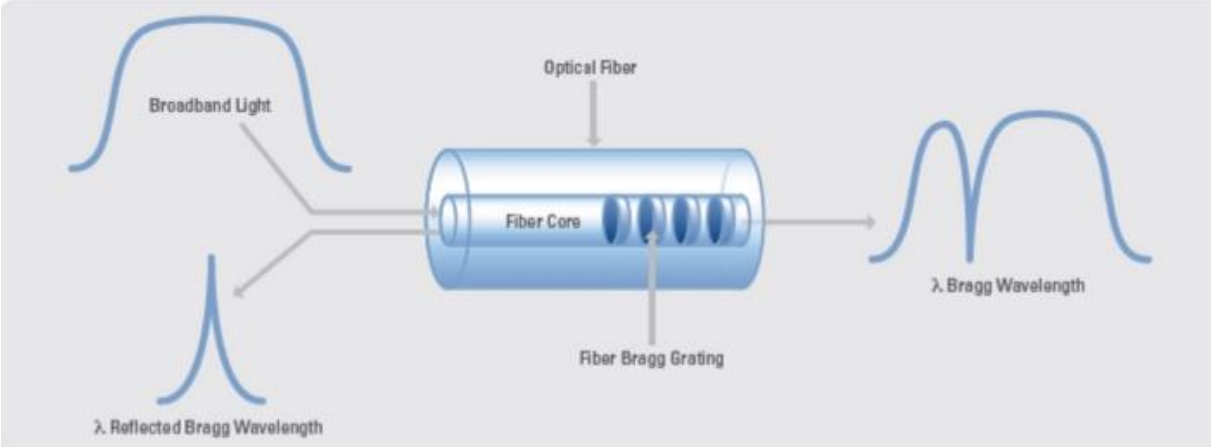

Figura 2.5. Sensor óptico FBG. (Instruments N. , Fundamentos de la deteccion optica FBG, 2016)

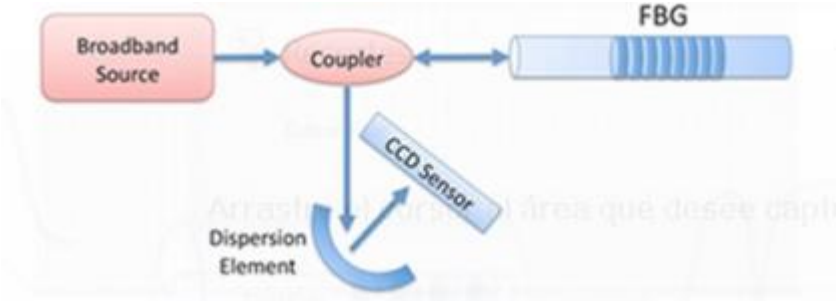

Figura 2.6. Método de conversión de posición de longitud de onda en FBGs.

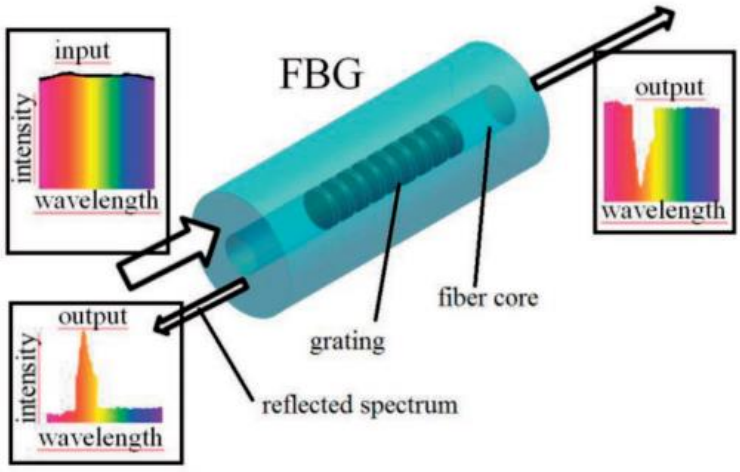

Figura 2.7. Principio de funcionamiento de FBGs. (MAREK BARSKI, 2014)

El método consiste en introducir fibras de Bragg en el núcleo de fibra óptica con las cuales se filtran determinadas longitudes de onda, si la fibra varia su longitud, también lo hace la longitud de onda con la que se filtran, con lo que se puede relacionar la longitud de onda con la deformación de la estructura.

Por último, también se usan como sensores las galgas extensométricas o células de carga también llamadas, que pueden ser de varios tipos:

- Piezoeléctricas: estas trabajan con el mismo principio que una galga extensométrica normal, pero el voltaje de salida es generado por materiales piezoeléctricos proporcional a la deformación de la célula de carga.

- Hidráulica: la célula utiliza un pistón y un cilindro, está llena de aceite. Cuando se aplica una carga sobre el pistón este realiza un movimiento, el cual se traduce en un aumento de la presión del aceite que es el que se puede transformar en una fuerza determinada. Este tipo de células es ideal para ambientes peligrosos ya que no utiliza dispositivos eléctricos.
- Neumáticas: se mide la presión del aire dentro de la célula.

Las células convencionales que se colocan en estructuras llevan integrado el puente Wheatstone, con lo que obtienes una señal diferencial del directamente además de la alimentación que tienes que proporcionar a la célula.

Figura 2.8. Células de carga.

2.2.3. Sensores de vibraciones

Los sensores más usados para medir vibraciones en una estructura son acelerómetros.

Los acelerómetros miden la aceleración de una ubicación particular de una estructura ya sea debido a la gravedad o a cargas aplicadas. Los valores de aceleración pueden proporcionar información valiosa acerca de las características dinámicas de la estructura. Estos dispositivos emiten una gran cantidad de datos si mides en modo dinámico que requieren un procesamiento intensivo.

Estos suelen ser acelerómetros piezoeléctricos (acelerómetros convencionales) o MEMS (sistemas microelectromecánicos).

Acelerómetros piezoeléctricos

Este es el acelerómetro más popular, estos sensores poseen varias ventajas, tamaño pequeño, alto rendimiento, alta durabilidad, amplia gama de frecuencias. Los componentes básicos de este son:

- Base
- Elemento piezoeléctrico
- Masa sísmica

La configuración básica de este tipo de sensores se puede ver en la figura 2.9.

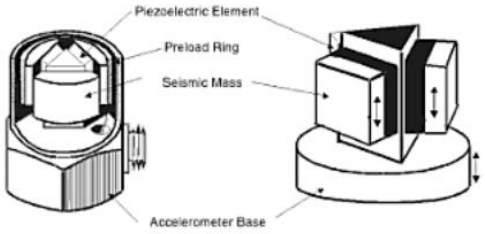

Figura 2.9. Configuración básica de un acelerómetro piezoeléctrico. (Plankis, 2012)

Los componentes piezoeléctricos tienen la propiedad única de que producen una salida eléctrica proporcional a la tensión aplicada. Cuando se aplica una aceleración a la base del sensor, los elementos piezoeléctricos experimentan una fuerza proporcional a la masa que están conectados y por lo tanto a la aceleración que sufren.

Esta fuerza genera una salida eléctrica que es enviada a través de un cable a un sistema de adquisición de datos. Una ventaja de este tipo de sensores es que son auto-generativos, el material piezoeléctrico emite una señal eléctrica sin la necesidad de una fuente de energía.

Cuando fueron inventados, se requerían cables especiales para que no les afectara el ruido, hoy en día ya llevan la electrónica necesaria para reducir este ruido.

Existen dos tipos de materiales que pueden ser utilizados en este tipo de sensores, el cuarzo y la cerámica cristalina. Ambos materiales tienen su utilidad, simplemente depende de la aplicación en la que se quieran utilizar. También existen tres estructuras diferentes para estos acelerómetros que dependen del tipo de esfuerzo aplicado al sensor.

Las opciones son en modo cizalladura, flexión y compresión. La primera opción tiene un tamaño muy pequeño y es muy útil para la respuesta de alta frecuencia, los del modo flexión son los mejores para bajas frecuencias, por último, los de compresión son muy resistentes y pueden soportar altos niveles de choque, pero son más sensibles a los efectos térmicos. (Plankis, 2012)

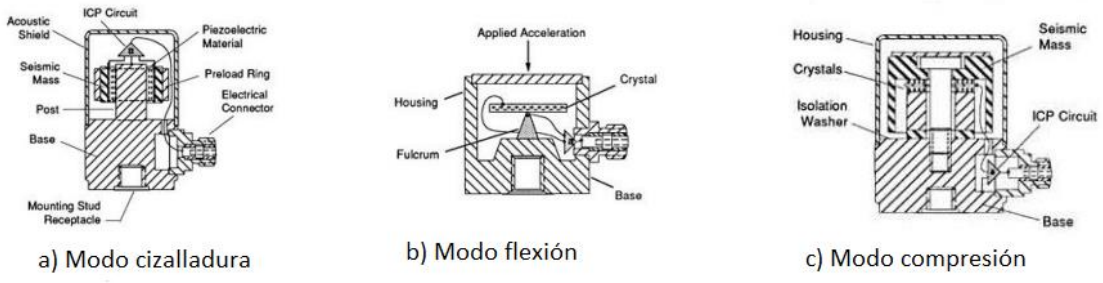

Figura 2.10. Tipos de estructura para acelerómetros piezoeléctricos. (Plankis, 2012)

Acelerómetros MEMS

El principio de funcionamiento de estos sensores es el desplazamiento de una pequeña masa grabada en una superficie de silicio y suspendido por unas pequeñas vigas. De acuerdo con la segunda ley de Newton (F = ma) se puede obtener la aceleración deseada. Las vigas actúan como un resorte, el aire actúa como

amortiguador con lo que se puede modelarlo como un sistema de segundo orden.

En la siguiente tabla se muestran dos tipos de acelerómetros MEMS, simplemente para ver las características de los mismos.

	ADXL 202E	Silicon Designs 1221L
Tipo	MEMS	MEMS
Rango	-2G a 2G	-0.1G a 0.1G
Ruido	200 (μG/$\sqrt{Hz}$)	30 (μG/$\sqrt{Hz}$)
Precio	10 $	150 $

Tabla 2.3. Comparación de dos acelerómetros MEMS. (Sukun Kim S. P., 2006)

2.3. Ejemplos de monitorización en estructuras reales

En este apartado se van a describir ejemplos de medición de parámetros en estructuras reales.

2.3.1. Monitorización de dos estructuras independientes en la línea de alta velocidad Madrid – Levante

Con el fin de obtener una comparativa practica entre sensores tradicionales (bandas extensométricas) y sensores de fibra óptica en estructuras reales INECO-TIFSA y ÁLAVA INGENIEROS llevaron a cabo en el Aranjuez (Madrid) una prueba en dos estructuras independientes de la LAV Madrid-Levante.

La primera estructura es una jácena metálica hiperestática con tres apoyos sin articulaciones y la segunda un puente de dos vanos isostático de vigas “in-situ” de hormigón.

Para hacer una comparativa coherente se instrumentan dos zonas sometidas a las mismas acciones.

Puente metálico:

Se instrumentaron dos montantes contiguos trabajando ambos a tracción-compresión al paso del tren a dos alturas distintas. El montante de la izquierda se controló con dos bandas extensométricas y el de la derecha con dos bandas ópticas como se observa en la figura 2.11, con sus correspondientes compensaciones térmicas.

Se adquirieron datos a 500 HZ durante el paso de 9 trenes distintos, mostrándose los resultados de dos ejemplos en las figuras 2.12 y 2.13 de la comparativa entre las dos tecnologías:

- Media distancia
- Cercanías

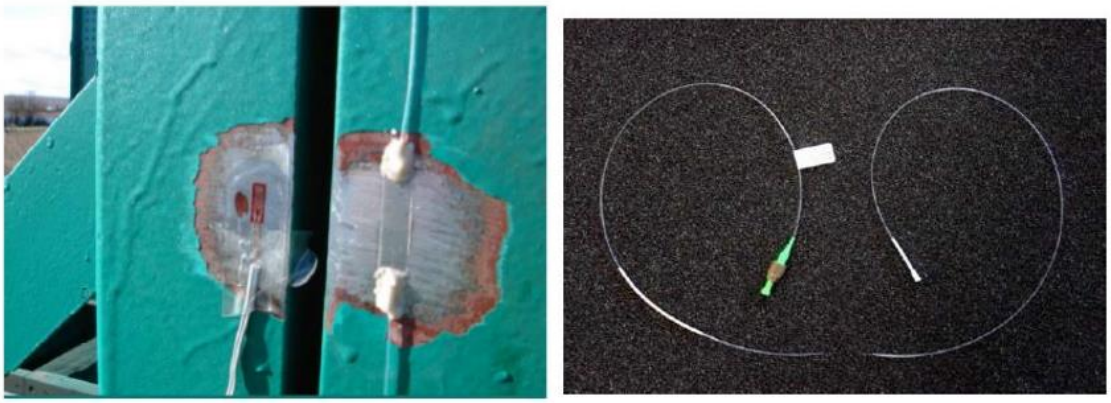

Figura 2.11.Bandas extensométricas y ópticas. (Ingenieros, 2016)

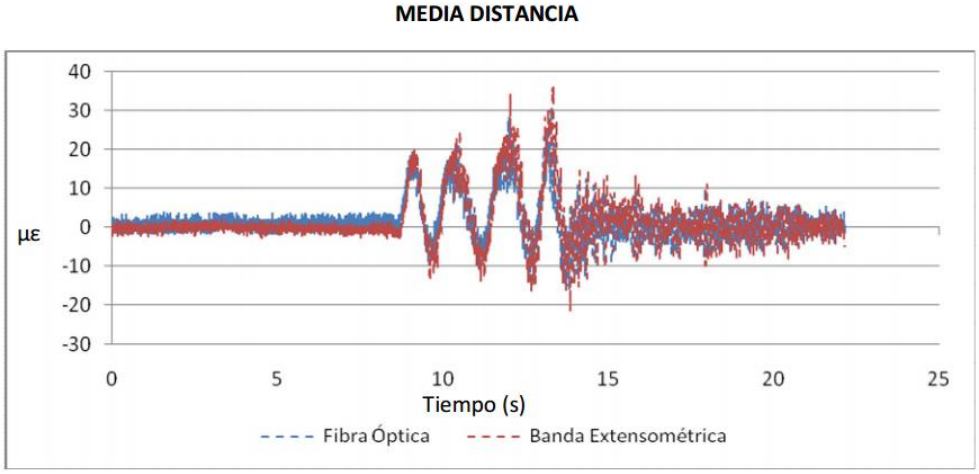

Figura 2.12. Datos para el tren Media distancia. (Ingenieros, 2016)

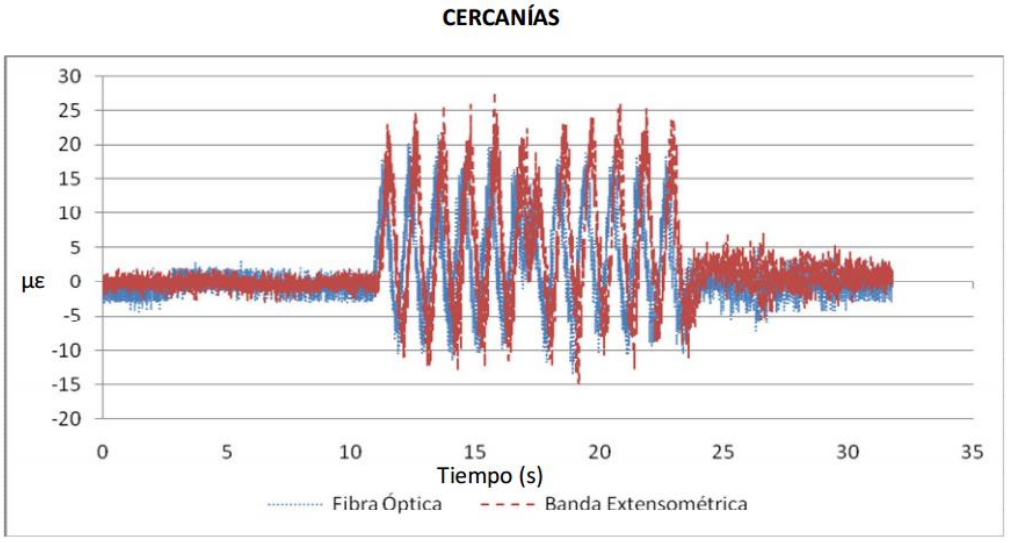

Figura 2.13. Datos para el tren Cercanías. (Ingenieros, 2016)

Puente de hormigón:

Aun sabiendo que el comportamiento tensional de un elemento portante de hormigón fisurado sometido a esfuerzos no sigue un patrón definido, se ha instrumentado una fisura generada por efectos cortantes.

Para ello se han instalado tres tipos de sensores:

- Banda extensométrica
- Banda óptica para anclar (mide deformaciones entre los dos puntos de anclaje, en este caso 60 mm) (MS-01)
- Cable de esfuerzo óptico de 1 m (SC-01)

En la figura 2.14 se ven los resultados obtenidos al paso de un tren cercanías, y en la figura 2.15 al paso de un tren media distancia.

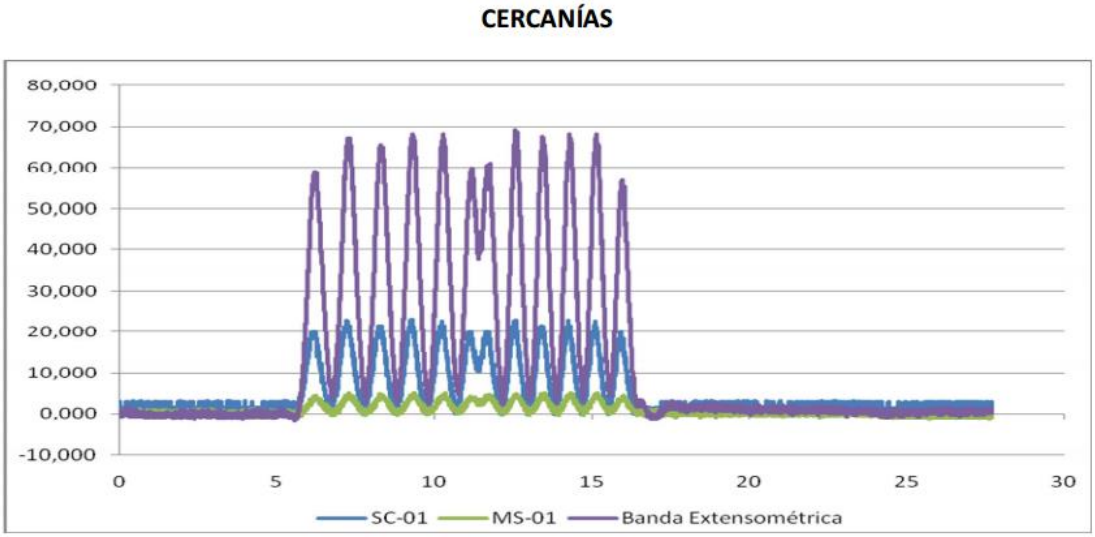

Figura 2.14. Resultados para el paso de un tren cercanías. (Ingenieros, 2016)

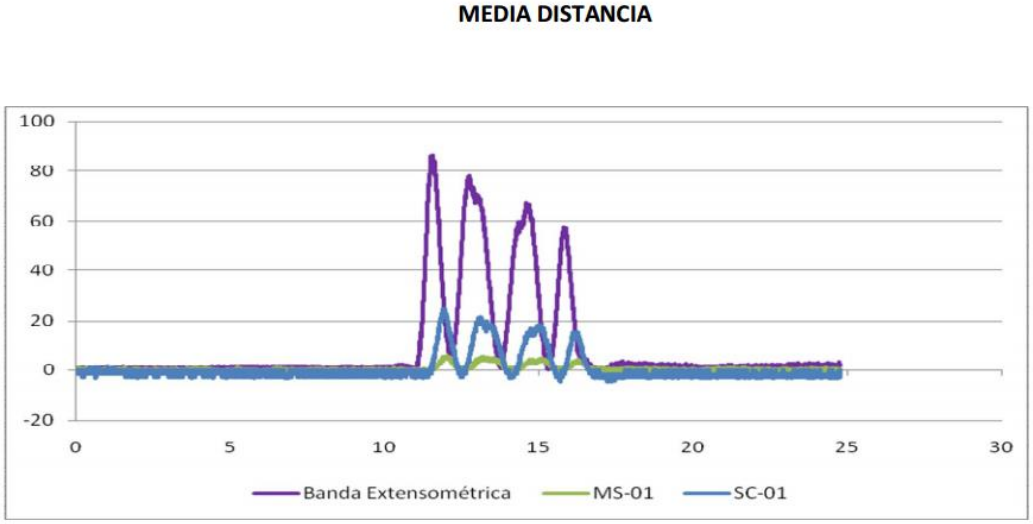

Figura 2.15. Resultados al paso de un tren media distancia. (Ingenieros, 2016)

Se observan claramente diferencias en cuanto a la amplitud en la señal de los 3 sensores, esto es debido a la longitud de medida y a la disposición de ellos en la viga, esta última se puede observar en la figura 2.16.

Figura 2.16. Disposición de los sensores en la viga. (Ingenieros, 2016)

Como se puede ver en la siguiente figura los tres sensores se han instalado en una fisura de la viga.

Figura 2.17. Instalación de los sensores en la fisura. (Ingenieros, 2016)

Con los resultados obtenidos se puede concluir que para el puente metálico existe un pequeño desfase en el tiempo debido a que no se toman las medidas sincronizadas.

En cuanto al puente de hormigón, como bien se sabe el hormigón siempre ha tenido su problemática con la medida e interpretación de los datos de extensometría debido a la falta de homogeneidad del material. En acero sin embargo la extensometría es puntual en toda la superficie y profundidad, pero en el hormigón puede aparecer fisuras o grietas superficiales, debido a esto se utilizan sensores de mayor longitud y o que se pretende es poder promediar las deformaciones existentes y obtener un valor de deformación lo más realista posible.

El caso que se analizó era un puente que se iba a rehabilitar posteriormente, por lo tanto, las fisuras superficiales como internas eran más que aparente.

2.3.2. Aplicaciones actuales con fibra óptica

En la actualidad son varias las estructuras que se han monitorizado con sistemas basados en fibra óptica, pero a modo de ejemplo se van a mostrar las siguientes:

Monitorización de un puente en Dresden (Alemania)

Puente de hormigón pre-tensado en el cual se instalan 4 sensores de fibra óptica en el vano, son FBGs pegados sobre el acero de refuerzo.

Figura 2.18. Puente de hormigón pre-tensado en Dresden (Alemania). (Meissner, 2016)

Monitorización de puente de Horsetail Falls (Oregón):

Se reforzaron dos vigas del puente y se instalaron 28 sensores de fibra óptica, para observar su comportamiento a lo largo del tiempo. Bajo las láminas de refuerzo de fibra de carbono se instalan 14 sensores y los demás se instalan por la parte exterior de las vigas protegidos con una capa de epoxy.

Figura 2.19.Instalacion de sensores en el puente de Horsetail Falls (Oregón). *(Llorente, 2014)*

Monitorización de un puente de ensayo de la universidad Braunschweig

Típicamente los sensores que se colocan en las estructuras son numerosos y conectados mediante cables largos que aumentan mucho los costes del sistema. En este ensayo se instaló una red de sensores MEMS conectados inalámbricamente, los MEMS son componentes que pueden ser producidos por unos 50 €.

El esquema de una instalación de este tipo es el mostrado en la figura 2.20, el construir una red de este tipo puede reducir los costes de los sistemas convencionales.

Como primera prueba se instaló el equipo para mediciones inalámbricas de tensión y emisiones acústicas en una instalación de ensayo grande de la universidad técnica de Braunschweig, Alemania (Figura 2.21).

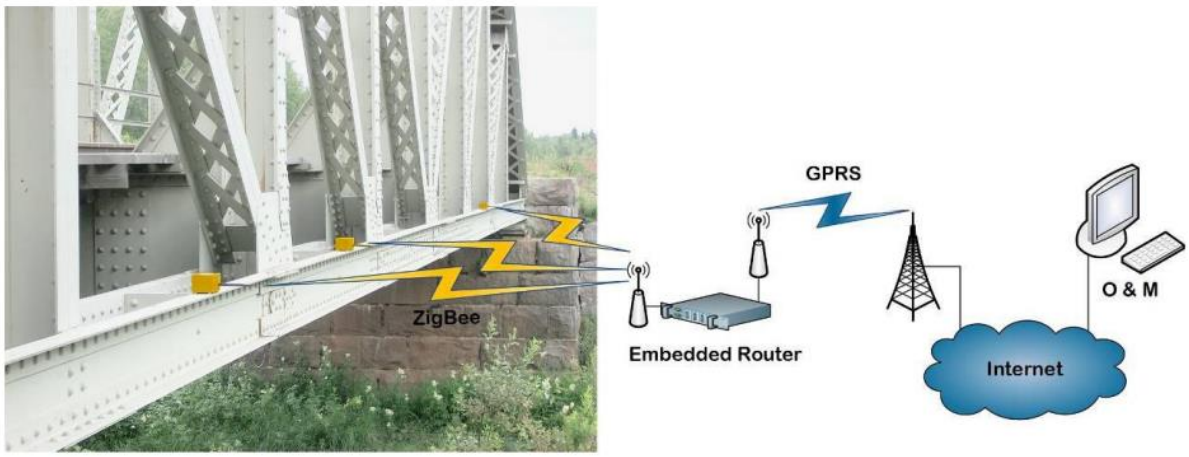

Figura 2.20.Esquema de una red de sensores inalámbrica en una estructura real. (Christian Grosse, 2010)

Figura 2.21. "Concerto Bridge" equipado con sensores inalámbricos acústicos (izquierda) y de tensión (derecha). (Christian Grosse, 2010)

Monitorización del puente Golden Gate

Una red de sensores para SHM es diseñada, desarrollada y testeada en el puente Golden Gate en San Francisco, EEUU. Vibraciones estructurales y variables medioambientales se miden de forma fiable y a un bajo coste sin interferir en el funcionamiento del puente.

64 nodos se distribuyen en el tramo principal y la torre. La recogida de vibraciones se hace de forma síncrona a una tasa de muestreo de 1 KHz.

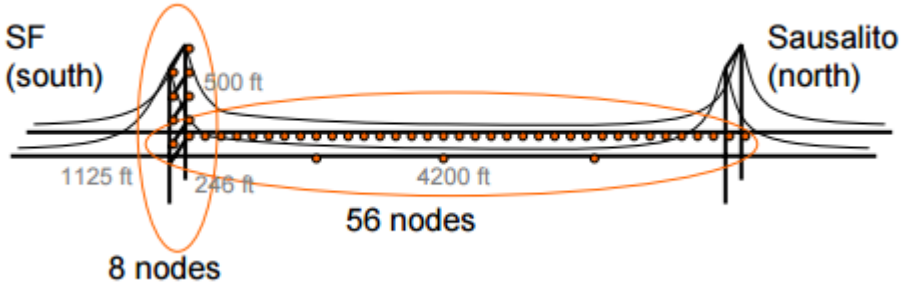

Figura 2.22. Despliegue de nodos en el "Golden Gate Bridge". (Sukun Kim S. P., 2016)

Figura 2.23. Instalación del nodo en el puente Golden Gate.

2.3.3. Aplicación en Pasarela Pedro Gómez Bosque (Valladolid)

En 2011 se construyó la “Pasarela Pedro Gómez Bosque” en la capital de Valladolid, esta constituye una ligera estructura peatonal sobre el río Pisuerga que crea una conexión entre un complejo deportivo y el centro de la ciudad (Figura 2.24). En 2012, el centro tecnológico CARTIF junto con la Universidad de Valladolid desarrollaron e implementaron un sistema de monitoreo de vibraciones continúas controlado de forma remota.

La singularidad de la estructura en cuanto a esbeltez y estética conduce a la decisión de utilizar acelerómetros MEMS de bajo coste incrustados dentro de la barandilla. Para el desarrollo de este sistema de monitorización se lleva a cabo un plan de monitorización específico.

Figura 2.24. Pasarela Pedro Gómez Bosque. (a) Vista de la entrada a la pasarela. (b) Vista paisaje. (Jesús de Sebastián, 2013)

Esta pasarela mide 85 m de largo, consiste en una placa de acero de 3.6 m de ancho y 30 mm de espesor. Este acero esta pretensado y anclado a dos pilares de acero por lo que es un único elemento estructural. La estructura se completa con suelo de goma y una barandilla de cristal.

Un análisis modal anterior se llevó a cabo utilizando acelerómetros convencionales, con lo que se obtuvo las frecuencias naturales y las formas de onda. Se estimaron 20 modos de vibración, incluyendo modos verticales, laterales, de torsión y acoplados entre 0.9 y 10 Hz. Estos parámetros son importantes para el diseño del sistema de monitoreo y elegir las especificaciones de los sensores más adecuados.

Un sistema de monitoreo de vibración estructural fue ideado con el fin de estimar de forma continua los parámetros modales de la estructura y ser capaz de conocer sus cambios bajo diversas condiciones ambientales, la variación de estos parámetros se hace frente al tiempo. Además de los acelerómetros necesarios también se instalaron

sensores para el viento y temperatura.

El sistema de monitoreo tiene dos limitaciones claras:

- Tiene que ser lo más barato posible (existía poco presupuesto)
- No debe influir en la estética de la pasarela.

El sistema consiste en 18 acelerómetros triaxiales, 9 en cada lado de la cubierta, un sensor de temperatura y un anemómetro (véase figura 2.25). Los acelerómetros se colocan equidistantes a lo largo del tramo, dentro del pasamanos.

Figura 2.25. Distribución de los sensores en la pasarela Pedro Gómez Bosque. (Jesús de Sebastián, 2013)

El sensor de vibración elegido fue el acelerómetro ADXL327 desarrollado por Analog Devices. Este sensor es muy pequeño, dispone de 3 ejes, de bajo consumo y bajo coste. Permite medir vibraciones estáticas para la inclinación y muestreos altos para vibraciones dinámicas. Se puede ver la disposición de los acelerómetros en la figura 2.26.

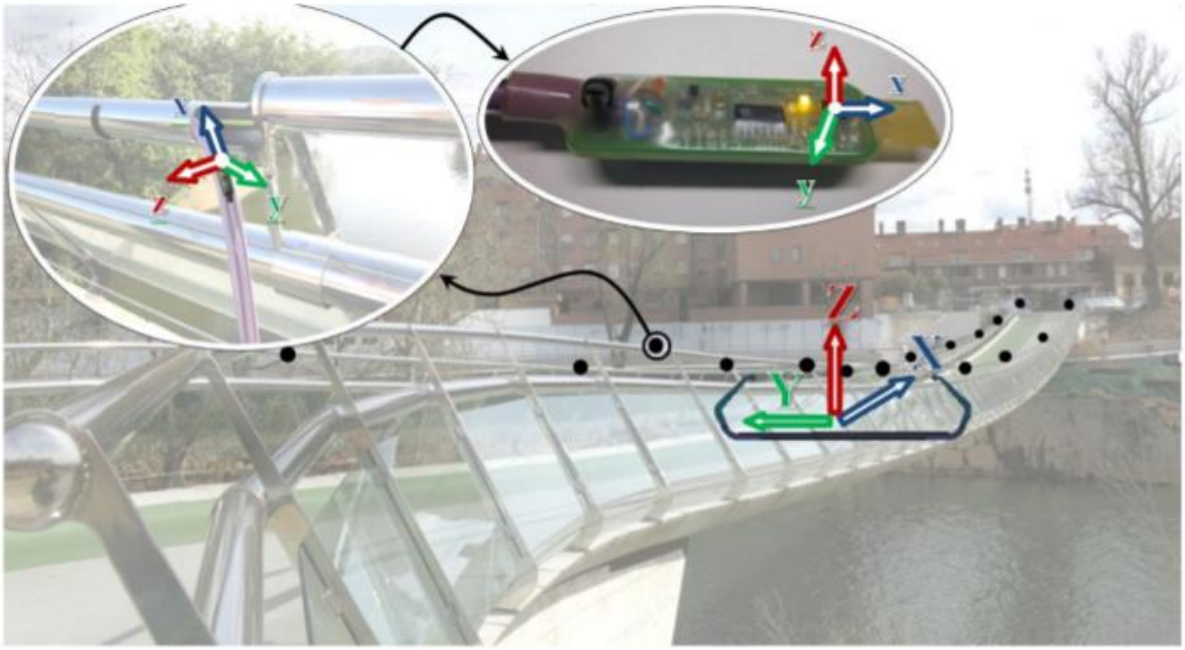

Figura 2.26. Disposición de los acelerómetros en la pasarela Pedro Gómez Bosque. (Llorente, 2014)

El sistema de adquisición de datos se instaló en una caja eléctrica situada cerca del puente, este sistema contiene un “data logger” con dos tarjetas de 32 canales de señales analógicas cada una puesto que se necesitaban 57 entradas. También dispone de un router para el envío de los datos cada hora.

Figura 2.27. Sistema de adquisición de datos en la pasarela Pedro Gómez Bosque. (Jesús de Sebastián, 2013)

El sistema se validó mediante la comparación de los datos medidos por los acelerómetros piezoeléctricos convencionales y los datos obtenidos por los acelerómetros MEMS. Estas pruebas demostraron que estos sensores son una alternativa competitiva a las tradicionales.

Los datos obtenidos para los dos tipos de sensores fueron los que se pueden ver en la figura 2.28.

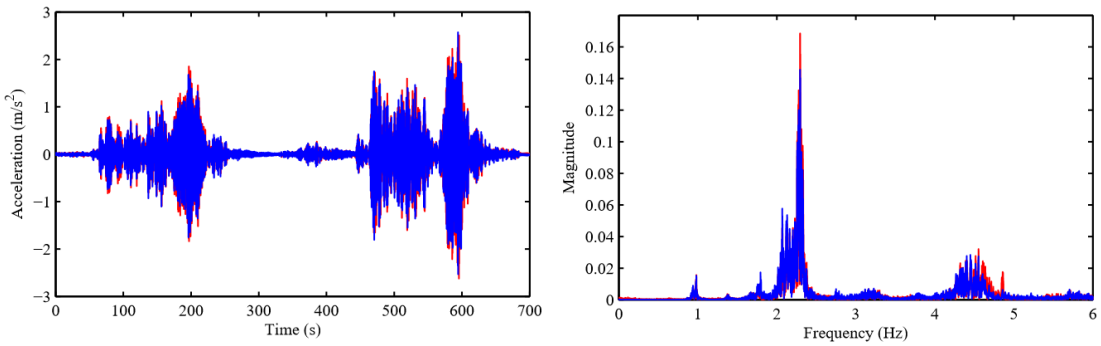

Figura 2.28. Datos en el dominio del tiempo y la frecuencia de acelerómetros MEMS (rojo) y acelerómetros convencionales (azul). (Jesús de Sebastián, 2013)

3. DISEÑO HARDWARE

En este capítulo se va a hacer una descripción de todo el sistema a nivel hardware, con el detalle del sistema de adquisición de datos y la selección de sensores adecuada.

Se empezará con una breve introducción al sistema diseñado y desarrollado, a continuación, se explicará cada sistema por separado y más adelante la instrumentación requerida para cada sensor seleccionado.

3.1. Introducción

En cuanto al sistema de adquisición de datos, este se compone de un ordenador de placa reducida como es la Raspberry pi y una tarjeta diseñada para leer entradas analógicas y leer a su vez los sensores necesarios para el desarrollo de este proyecto.

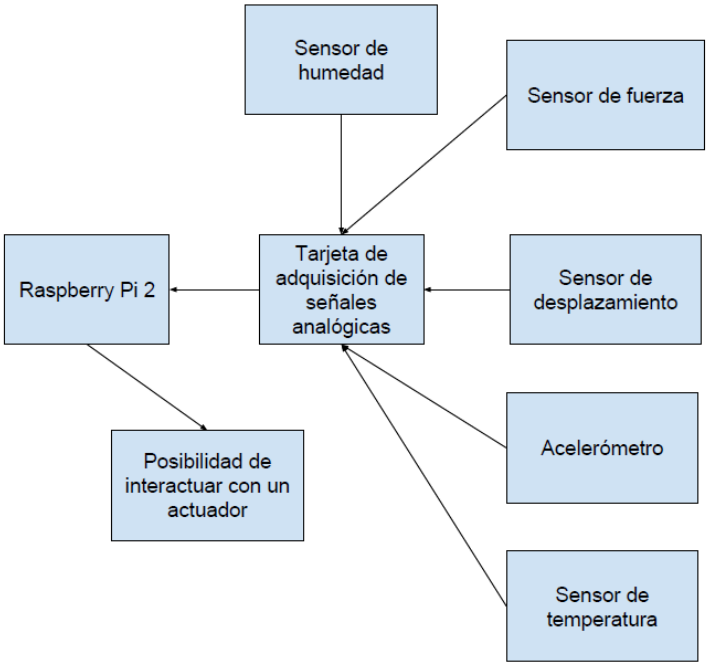

Figura 3.1. Esquema introducción al hardware.

En la figura 3.1 se ven las capacidades del conjunto del sistema diseñado para este proyecto. Este sistema las posibilidades que nos ofrece es leer cualquier tipo de sensor con la instrumentación requerida por este, y además la posibilidad de interactuar con un actuador y poder transmitir perturbaciones.

La Raspberry pi 2 por si sola puede leer sensores digitales, pero esto es una limitación a la hora de crear un sistema de monitoreo, ya que la mayoría de sensores ofrecen señales analógicas, con lo que se decidió diseñar y fabricar una placa que supere esta limitación.

3.2. Ordenador de placa reducida. Raspberry pi 2

El ordenador de placa reducida elegido es una Raspberry pi 2, la cual cuenta con un procesador central como es el BCM2836 ARM Cortex-A7 de 4 núcleos a 900 MHz y memoria RAM de 1 GB. El diseño de la misma no incluye un disco duro ni una unidad de estado sólido, ya que utiliza una tarjeta SD para el almacenamiento permanente, pero dispone de puertos USB 2.0 por si en un futuro se necesitase mayor

almacenamiento.

Esta placa está alimentada a 5V vía Micro USB y tiene un consumo de 800 mA aproximadamente. La Raspberry pi usa mayoritariamente sistemas operativos basados en Linux. Raspbian es una distribución basada en Debian que esta optimizada para el hardware de la Raspberry pi, se lanzó en julio de 2012 y es la distribución recomendada para iniciarse con la Raspberry pi.

Existen una gran variedad de ordenadores de bajo coste como son:

- BeagleBoard, un ordenador embebido con hardware libre, también dispone de un procesador ARM a 1GHz, 1 puerto USB, µHDMI, Ethernet y alimentación a 5V.
- OLinuXino, es un ordenador Linux de bajo coste con hardware y software abierto, disponen de varias versiones con procesadores de hasta 1GHz y memoria RAM de 512 MB, Wifi integrado. El precio de una placa de estas características es más alto que el de una Raspberry pi.
- Banana Pi, este ordenador embebido posee prácticamente las mismas características que una Raspberry pi, pero la gran limitación es el soporte y el precio más alto.
- CubiBoard5, este mini ordenador tiene un procesador de 2GHz y RAM de 2GB, puerto Ethernet Gigabit, pero su precio supera los 100 €.

En la actualidad ya se comercializa el modelo 3 de la Raspberry pi, que trae como mejoras Wifi y Bluetooth Low energy integrados. Esta nueva placa se puede integrar en líneas futuras del presente proyecto ya que dispone de 2 conectividades integradas como puede ser el Bluetooth que tiene aplicaciones interesantes en este tipo de sistemas.

En el presente proyecto observando la cantidad de alternativas que tenemos disponibles para el hardware de adquisición de datos, se ha elegido la Raspberry Pi debido a la comunidad que existe desarrollando para este ordenador de placa reducida, con lo que nos va a facilitar el desarrollo de este sistema.

Figura 3.2. Raspberry pi 2. (Wordnice, 2016)

Para leer determinados sensores se necesitan puertos como son:

- I2C, necesario para sensores de temperatura y de humedad, también necesario para poder integrar en la tarjeta de expansión conversores analógico digitales.
- SPI, necesario para determinados conversores analógico digitales.
- Entradas analógicas, aquí está la limitación de la Raspberry pi, pero este tema se soluciona como se ha comentado anteriormente diseñando una placa que facilite la lectura de este tipo de sensores.

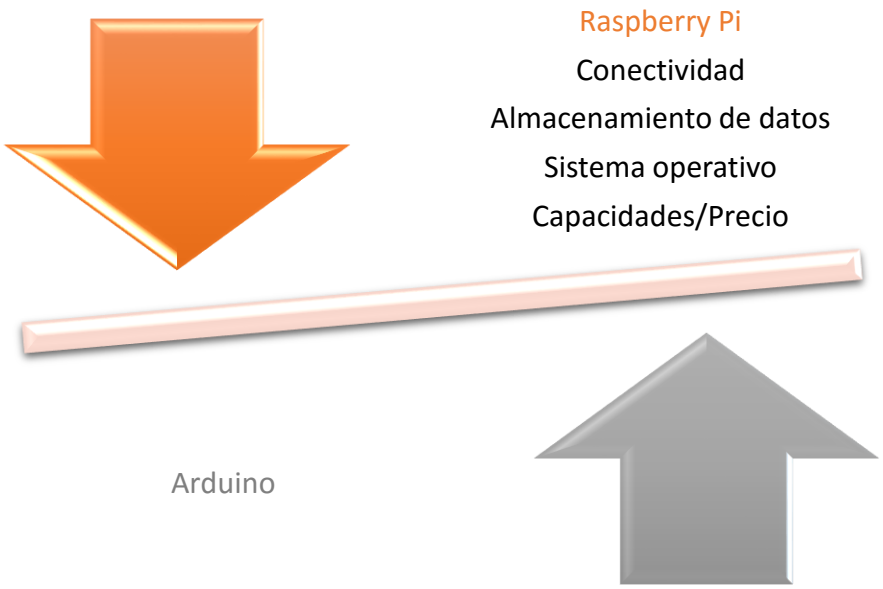

Figura 3.3. Comparación Raspberry pi con Arduino.

Los elementos clave que interesan para seleccionar la Raspberry pi como elemento de adquisición de datos son los que se ven en la figura 3.3.

Para el presente proyecto se ha elegido la Raspberry pi por varias características:

- Conectividad a la red más fácil y barato que con un microcontrolador como puede ser un Arduino. La Raspberry pi dispone de puerto ethernet, puertos USB para conectar Wifi o un modem 3g.
- Almacenamiento de datos, con la Raspberry pi se dispone de una ranura para tarjeta SD de hasta 32 GB, también de puertos USB para conectar cualquier tipo de almacenamiento externo flash con lo que almacenar datos de las medidas deseadas.
- Sistema operativo, en el cual se pueden programar tareas, instalar un servidor con el cual se puede gestionar una base de datos.
- Precio, el precio de la Raspberry pi es muy competitivo en comparación con microcontroladores como Arduino y además ofrece todas las características comentadas anteriormente.

Los puertos de comunicación que se utilizarán con la Raspberry pi, son los que se ven en la figura 3.4.

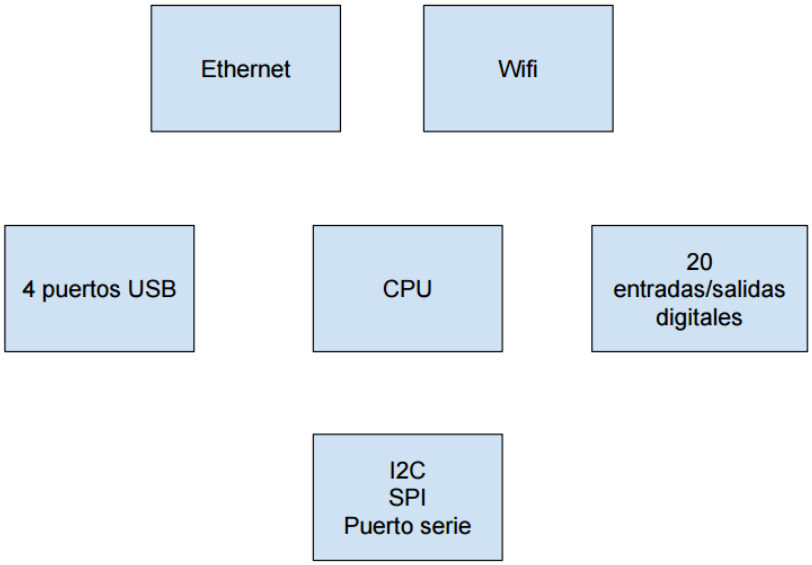

Figura 3.4. Descripción elementos Raspberry Pi 2.

En la figura 3.5 se puede observar el conector de pines del que dispone la Raspberry pi, se ve que dispone de alimentación de 3.3V y 5V, 8 pines de GND, de casi 20 entradas/salidas digitales (se muestran en verde en la figura), puerto I2C (para conectarse a sensores digitales, en azul en la figura), puerto SPI (para determinados sensores o electrónica para medir estos, en morado en la figura), puerto serie por si se tuviese que conectar con otro microcontrolador o directamente con otro ordenador (se muestra en naranja en la figura).

Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I2C)	DC Power 5v	04
05	GPIO03 (SCL1 , I2C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I2C ID EEPROM)	(I2C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Figura 3.5. Entradas/Salidas Raspberry pi 2. (Raspberry pi Geek, 2016)

3.3. Tarjeta de adquisición de señales analógicas y otro tipo de sensores

En este apartado se van a describir todos los módulos realizados dentro de la tarjeta para la adquisición de datos de los sensores seleccionados para el monitoreo de estructuras, viendo así las dos versiones utilizadas en este proyecto. La primera con ciertas limitaciones en medir determinados sensores, pero con la segunda se obtienen medidas de todos los sensores requeridos. En la siguiente figura se pueden ver las capacidades que se obtienen con la versión definitiva.

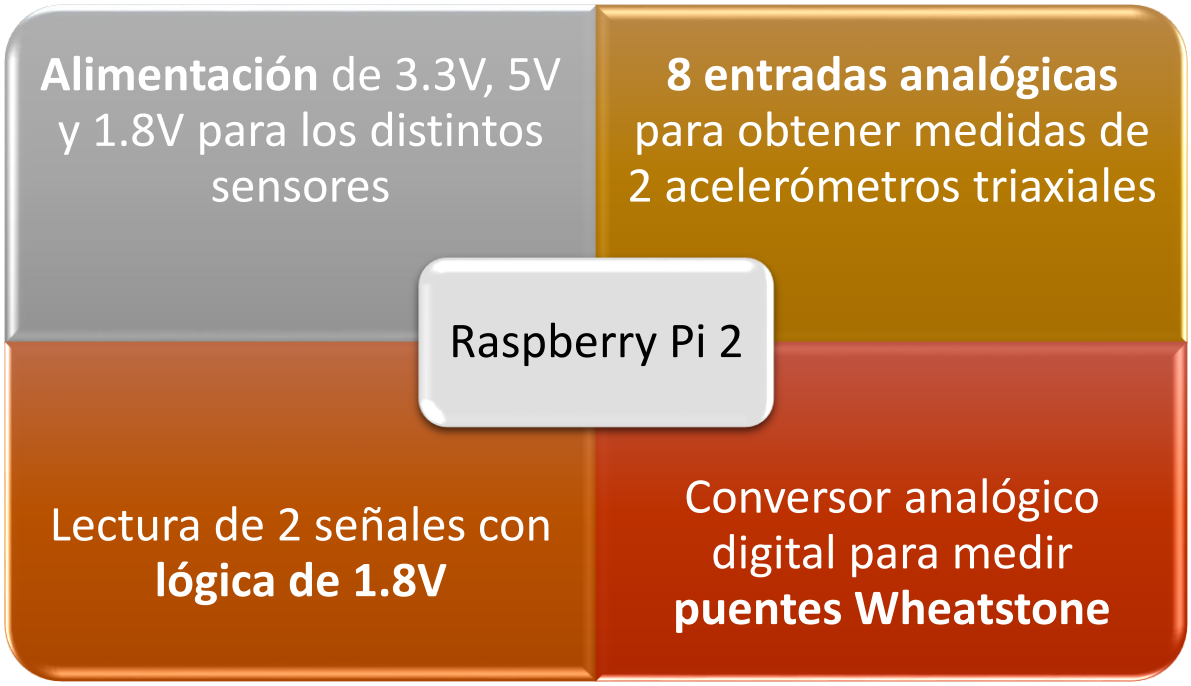

Figura 3.6. Capacidades extras con la tarjeta de adquisición de datos.

Con esta tarjeta diseñada se obtienen una serie de características que no se tienen con la Raspberry Pi por sí sola, con lo que se consiguen las siguientes capacidades extras:

- 8 entradas analógicas, a través de dos conversores analógicos/digitales, con estas señales se pueden medir dos acelerómetros triaxiales y uno biaxial, en esta aplicación solo se dispone de un acelerómetro, pero en el presente capítulo se muestran todas las capacidades del sistema.
- Conversor analógico digital para medir puentes Wheatstone, este es un conversor analógico digital especial para medir diferencias de tensión en puentes Wheatstone, es muy útil en el ámbito de medir sensores de fuerza ya que suelen ser una composición de galgas extensiométricas en puentes de Wheatstone.
- Lectura de señales con lógica de 1.8V, este módulo es útil para la medición de sensores de desplazamiento como son calibres digitales que más adelante se describirán.
- Alimentación de los distintos sensores, aquí la principal capacidad extra que añadimos es la alimentación a 1.8V para el sensor de desplazamiento.

Se pueden distinguir tres grandes módulos en esta tarjeta, que son la electrónica necesaria para los acelerómetros, la electrónica necesaria para el sensor de fuerza y

la necesaria para el sensor de desplazamiento.

3.3.1. Módulo para acelerómetros

Este módulo se compone de dos conversores analógico/digitales con los que se transformaran las señales analógicas a digitales para que las pueda leer la Raspberry Pi.

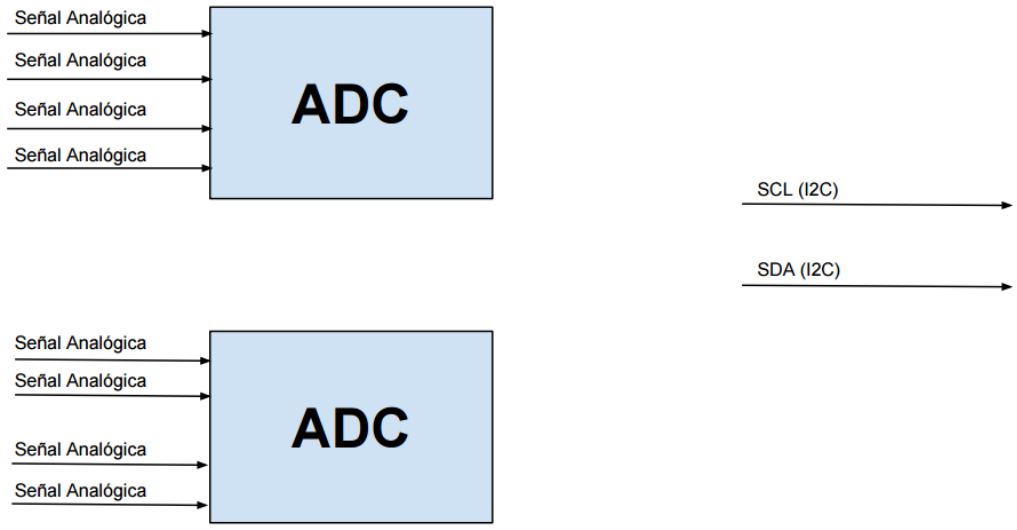

Figura 3.7. Esquema módulo para acelerómetros.

A este módulo se le hace una petición a través del puerto I2C de que señales quiero leer, con estas se obtiene el nivel de tensión de estos canales. Esta comunicación se realiza a través de tres líneas: SDA (datos), SCL (reloj) y GND (tierra). La información sólo se transmite a través de SDA y SCL, mientras que GND se utiliza como referencia. SDA y SCL son líneas en drenador abierto, por lo que necesitan resistencias de pull-up, la función de estas es mantener a nivel lógico alto los pines SDA y SCL, de modo que el bus está libre cuando estas están a este nivel.

La comunicación se puede efectuar cuando la línea SCL este a nivel lógico alto, ya que esto significa que el bus está libre y se pueden comunicar dispositivos. El dispositivo maestro (en nuestro caso la Raspberry Pi) envía una cadena de bits que representa la dirección del dispositivo con el que se desea comunicar, seguido de un bit que representa si se quiere escribir o leer de ese dispositivo.

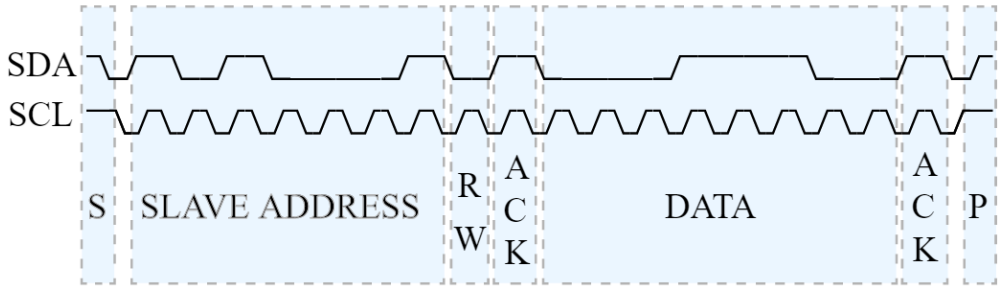

Figura 3.8. Protocolo de comunicación I2C. (DLNWare, 2016)

Si el dispositivo está conectado al bus y se puede comunicar envía un ACK

(confirmación) al maestro con lo que empieza la transmisión. En la figura 3.8 se muestra la forma de comunicación entre dos dispositivos I2C, donde se ve que el primer bit es un bit de arranque (SDA y SCL pasan a nivel bajo). En este momento se le pueden enviar comandos para escribir en determinados registros o leerlos. Cada pulso de la línea de SCL nos permite intercambiar un bit.

El convertidor analógico a digital seleccionado es un ADS1115 de Texas Instruments con 16 bits de resolución, el diagrama funcional del mismo es el que se muestra en la figura 3.9.

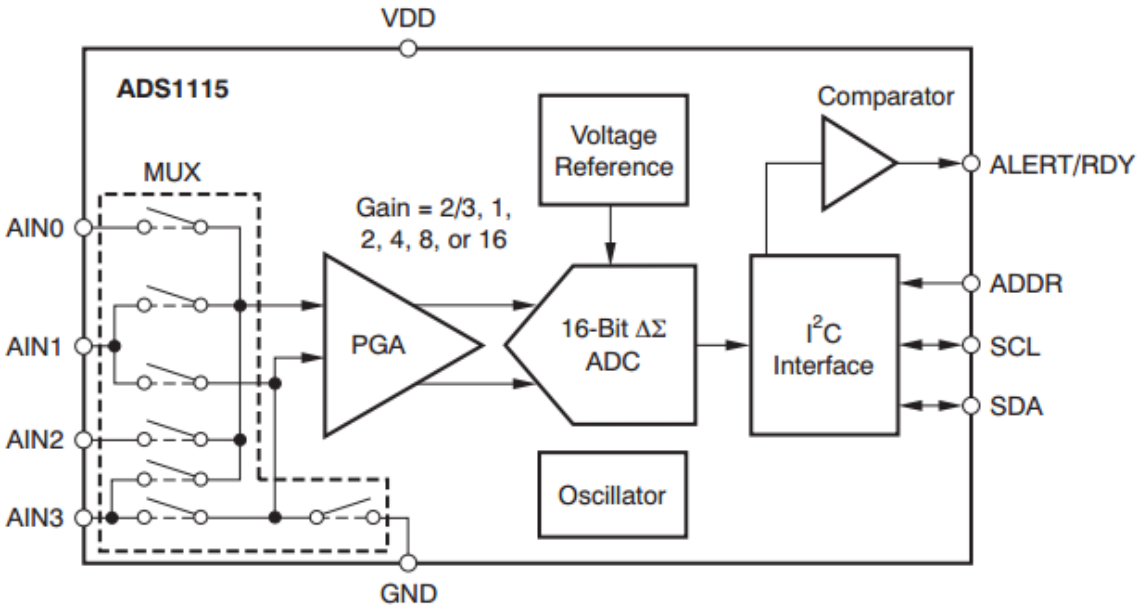

Figura 3.9. Diagrama funcional del ADS1115. (Instruments T. , ti.com, 2016)

Las principales características de este convertidor son las siguientes:

- Tensión de alimentación de 2V a 5V
- Resolución de 16 bits
- Velocidad de muestreo de hasta 860 muestras por segundo
- 4 canales (entradas)
- Programación de ganancia interna
- Puerto de comunicaciones I2C para la comunicación con la Raspberry Pi
- Posee dos modos de entrada (Directa o diferencial)
- Encapsulado 10VSSOP

Se ha seleccionado este convertidor ya que ha sido probado en diversos proyectos anteriores y da buenos resultados, también con él se obtiene una buena resolución en los acelerómetros y velocidades de hasta 860 muestras por segundo si se desease.

En la figura 3.10 se puede observar una trama de petición de datos de la Raspberry Pi al convertidor, en la que se ve como la comunicación empieza con bit de arranque a continuación le sigue la dirección única del esclavo y por último se le envía un comando con el cual se le indica que queremos obtener datos de un determinado canal. Todos los bytes recibidos por el esclavo o maestro son respondidos con un bit de confirmación indicado por una “A” en la figura 3.10.

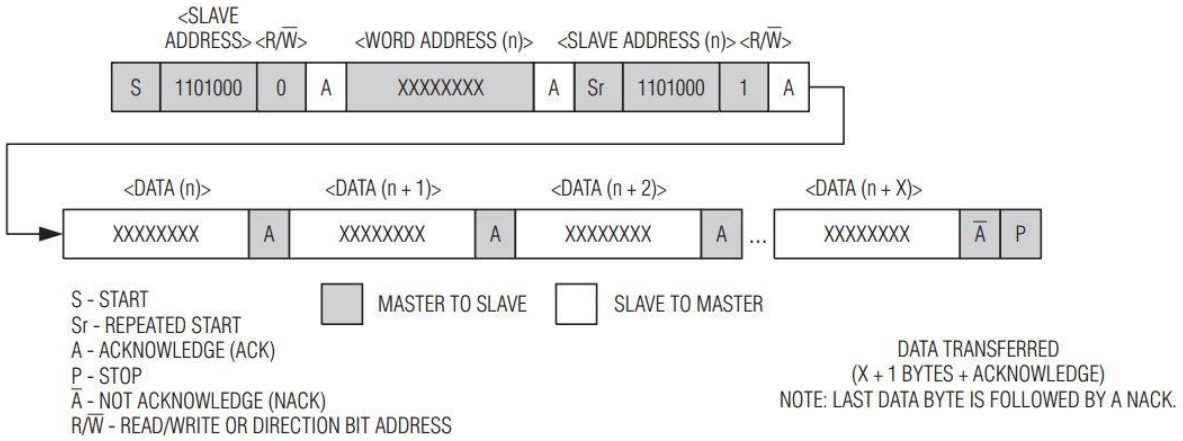

Figura 3.10. Trama de petición de datos al convertidor.

En la figura 3.11 se puede ver el esquema de conexión que nos da el fabricante para que el convertidor se comunique con cualquier microcontrolador o procesador que tenga puerto de comunicaciones I2C. Como se puede observar se necesitan integrar resistencias de pull-up para mantener a nivel alto los dos canales cuando no se estén comunicando los dispositivos.

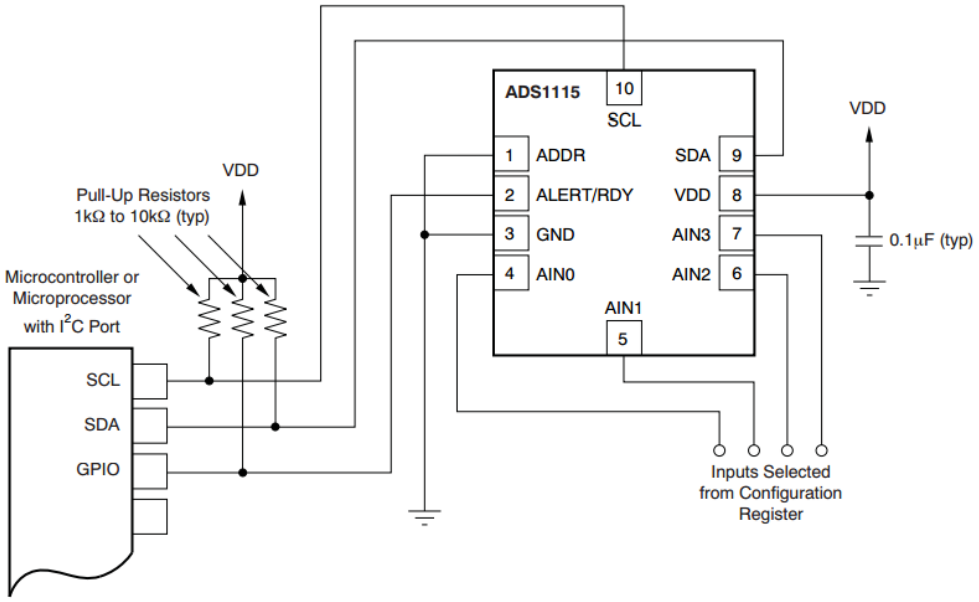

Figura 3.11. Esquema de conexión ADS1115. (Instruments T. , ti.com, 2016)

También destacar que a la entrada de las 4 señales de un convertidor analógico a digital se han colocado amplificadores operacionales en modo seguidor de tensión (o buffer de voltaje), con esto lo que se pretende conseguir es transferir una tensión de un primer circuito (acelerómetro), que tiene un nivel de salida de alta impedancia, a un segundo circuito (convertidor analógico a digital) con un nivel de entrada de baja impedancia.

Este buffer impide que el circuito del convertidor analógico a digital cargue demasiado al acelerómetro, provocando un funcionamiento incorrecto.

El circuito final para este módulo es el que se muestra en la figura 3.12.

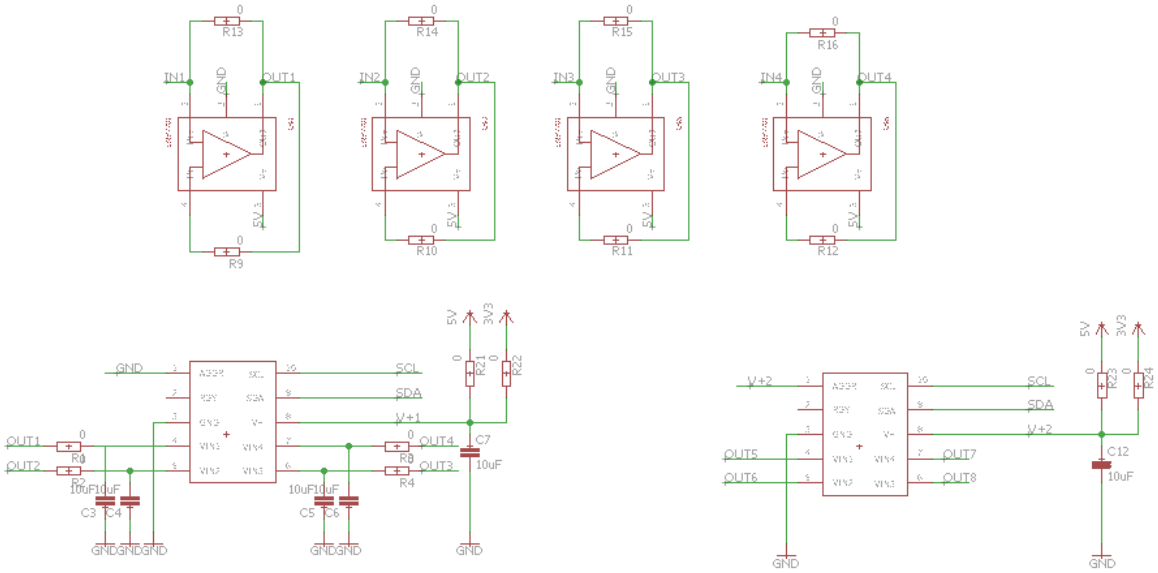

Figura 3.12. Esquema final módulo para acelerómetros

3.3.2. Módulo para sensores de fuerza

En este apartado se va a describir la parte dedicada a medir sensores de fuerza, este módulo se compone de un convertidor analógico digital de 32 bits de resolución dedicado a medir señales diferenciales que se obtienen de un puente Wheatstone con una interfaz de comunicaciones SPI.

El SPI (Serial Peripheral Interface) es un estándar de comunicaciones principalmente usado para la transferencia de información entre circuitos electrónicos. Este protocolo es síncrono y permite al maestro iniciar la comunicación con un dispositivo esclavo. Los datos se envían con una señal de reloj, este reloj controla cuando los datos son intercambiados y cuando deberían ser leídos, la señal de reloj la maneja el maestro.

Los datos son “intercambiados”, por lo que ningún dispositivo solo lee o solo escribe en el bus, realizan las dos tareas, por lo consiguiente tienen dos líneas de datos una de entrada y otra de salida. Una señal llamada CS (Chip select) es la que selecciona a que dispositivo habilitamos para intercambiar datos. Frecuentemente esta señal se activa a nivel bajo.

En la figura 3.13 se puede ver un ejemplo de comunicación SPI para entender cómo se interpretan las señales. Las flechas indican si la señal está experimentando un flanco descendiente o ascendiente. SDI muestra cuando los datos son muestreados y con qué valor.

Como se puede ver claramente, el muestreo se realiza en el flanco de reloj contrario al cual se realizan los cambios. El término “Modo 1,1” se refiere al modo en el cual está trabajando el SPI que se explicará más adelante.

SPI usa 4 señales:

- SS, selección de chip
- SCK, señal de reloj
- SDO, datos de salida

- SDI, datos de entrada

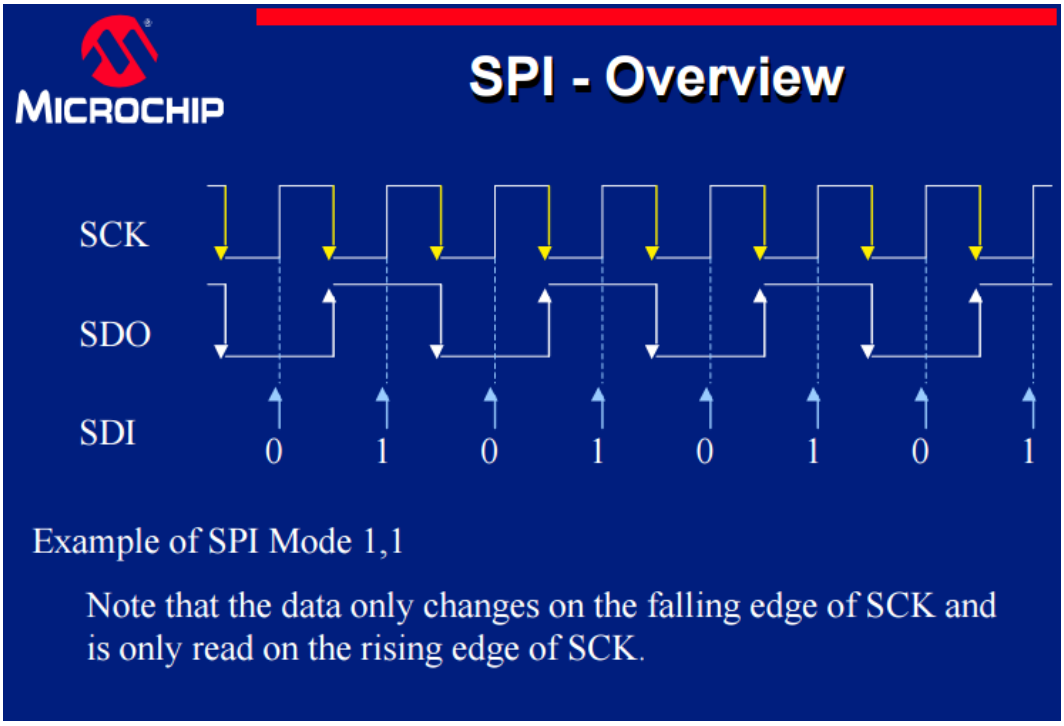

Figura 3.13. Ejemplo comunicación SPI. (Microchip, 2016)

Para programar el modo en el que se quiere trabajar, existen dos bits:

- CKP, selección de la polaridad del reloj, este determina si el reloj está inactivo a nivel alto o a nivel bajo. CKP = 1, SCK se inactiva a nivel alto y CKP = 0, SCK se inactiva a nivel bajo.
- CKE, selección del flanco del reloj, este bit controla cuando se transmiten datos en relación con el reloj.

En la siguiente tabla se pueden observar los modos SPI a través de las combinaciones de los dos bits comentados anteriormente.

Modo SPI	CKP	CKE
0,0	0	1
0,1	0	0
1,0	1	1
1,1	1	0

Tabla 3.1. Modos SPI

Después de haber explicado el protocolo de comunicaciones que va a usar el convertidor seleccionado, se van a describir las características del mismo por las cuales se ha elegido este convertidor:

- Precisión de 32 bits
- Velocidad de muestreo de 2.5 muestras por segundo hasta 38400 muestras por segundo

- Entrada diferencial, CMOS PGA
- 11 entradas analógicas multifunción
- 2,5V de tensión de referencia interna
- Test de señal ADC internos
- 8 entradas/salidas de propósito general

El fabricante recomienda este convertidor para balanzas y galgas extensiométricas ya que posee un bajo ruido, en la figura 3.14 se puede observar un ejemplo de aplicación que nos da el fabricante para medir con este convertidor células de carga.

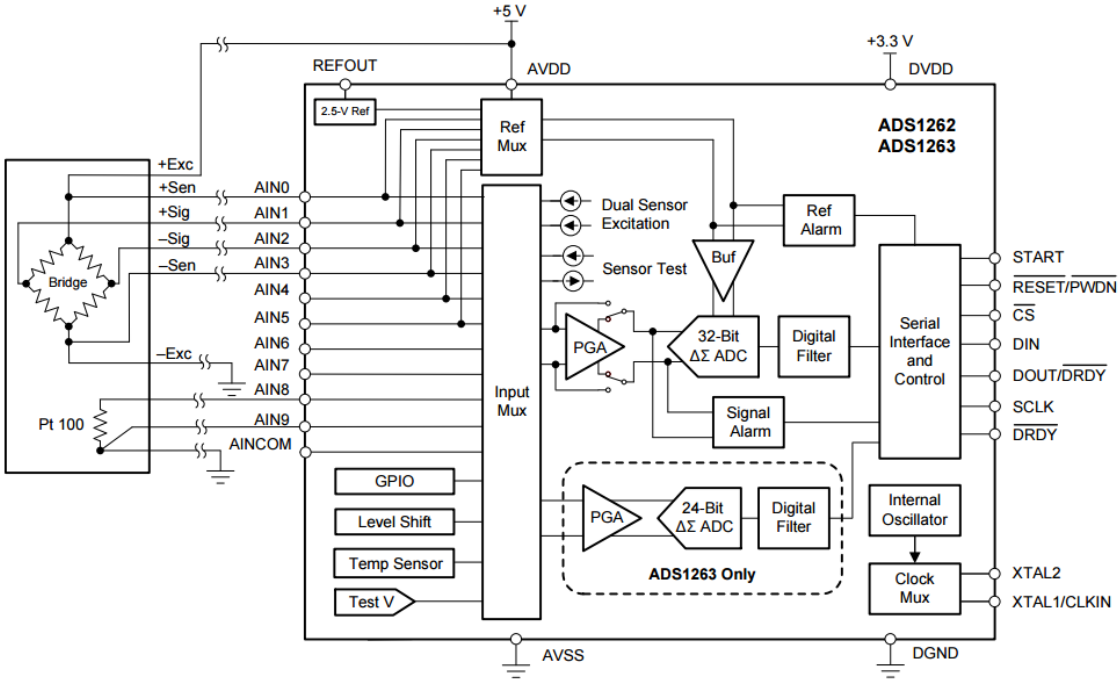

Figura 3.14. Ejemplo de aplicación ADS1262. (Instruments T. , ADS1262 ti.com, 2016)

Se puede observar cómo se necesitan 6 entradas para medir el puente de Wheatstone de la galga extensiométrica, aunque estas galgas posean 4 cables ya que +Exc y +Sen van puenteados, así como sus inversos.

El modo SPI en el que trabaja este convertidor es el “0,1”, con lo que ya se verá en el apartado software como se configura este modo para poder comunicarnos con el conversor.

El circuito definitivo integrado en la tarjeta (solo en la versión 2) es el que se muestra en la figura 3.15.

En la versión 1 se utiliza una tarjeta externa para medir galgas extensiométricas, incluso en la versión 2 también se sigue usando esta tarjeta porque no da malos resultados, el único inconveniente que tiene es la velocidad de muestreo que la limita a 90 muestras por segundo.

Esta tarjeta es una interfaz para puentes de Wheatstone basada en el chip HX711 que tiene las siguientes características:

- Dos entradas diferenciales seleccionables
- Chip de bajo ruido con ganancia programable de 32, 64 y 128

- Regulador para alimentar el ADC y las células de carga
- Velocidad de muestreo de 10 muestras por segundo hasta 90 muestras por segundo
- Voltaje de alimentación de 2.6-5V

En la figura 3.16 se puede observar una foto de la tarjeta.

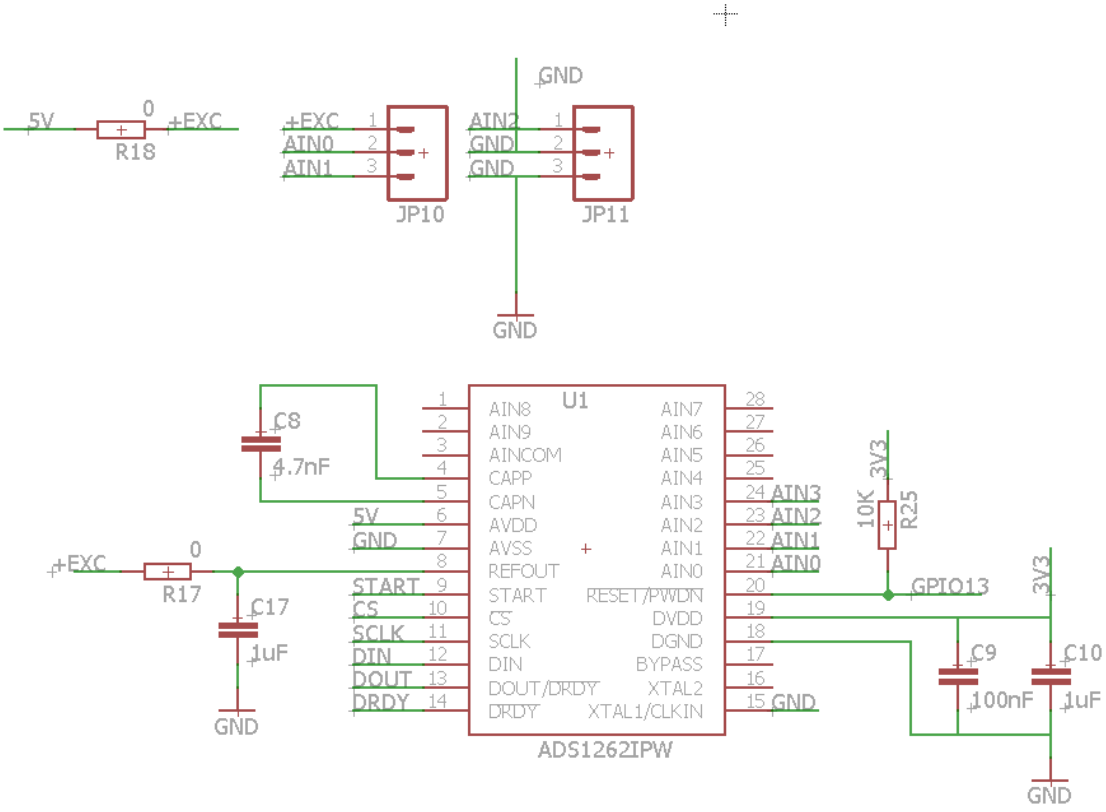

Figura 3.15. Esquema final modulo para sensores de fuerza

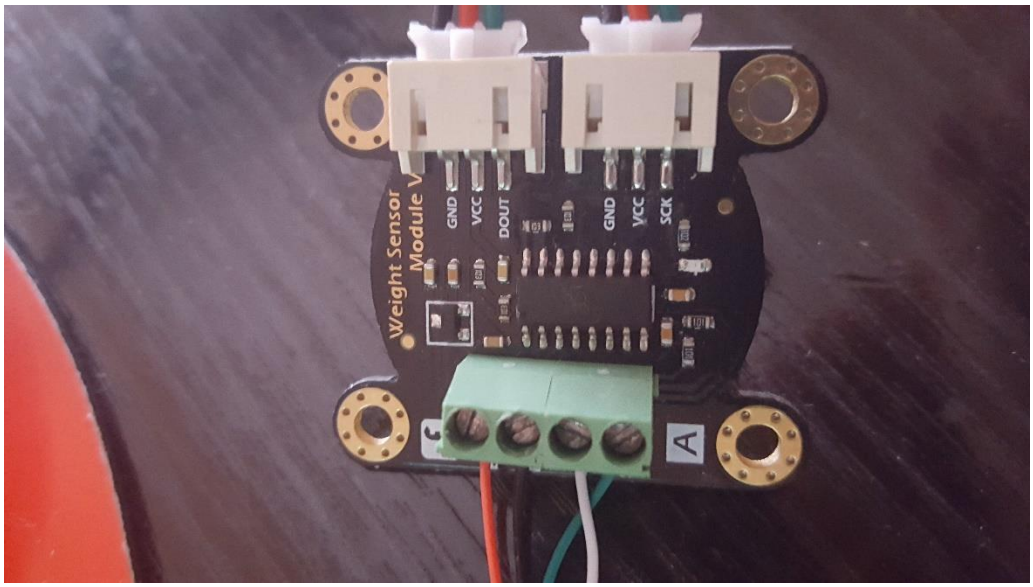

Figura 3.16. Foto tarjeta para galgas extensiométricas

El diagrama funcional de este circuito es el que se muestra en la figura 3.17. Se puede ver como se alimenta al puente con la misma tensión de referencia para tomar la diferencia de tensión del puente. Y vemos como la interfaz de comunicaciones de salida es diferente al SPI.

Esta interfaz de comunicación es propia del HX711 y se explicará en el apartado software detalladamente.

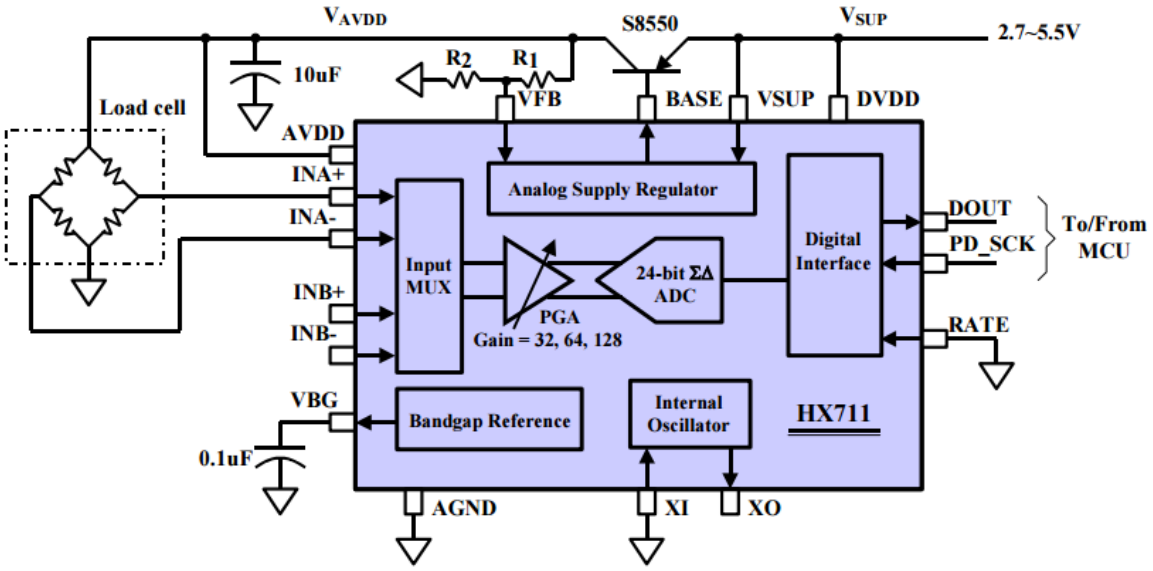

Figura 3.17. Diagrama de aplicación circuito HX711. (Sparkun, 2016)

Las conexiones de este circuito con una célula de carga son las que se muestran en la figura 3.18.

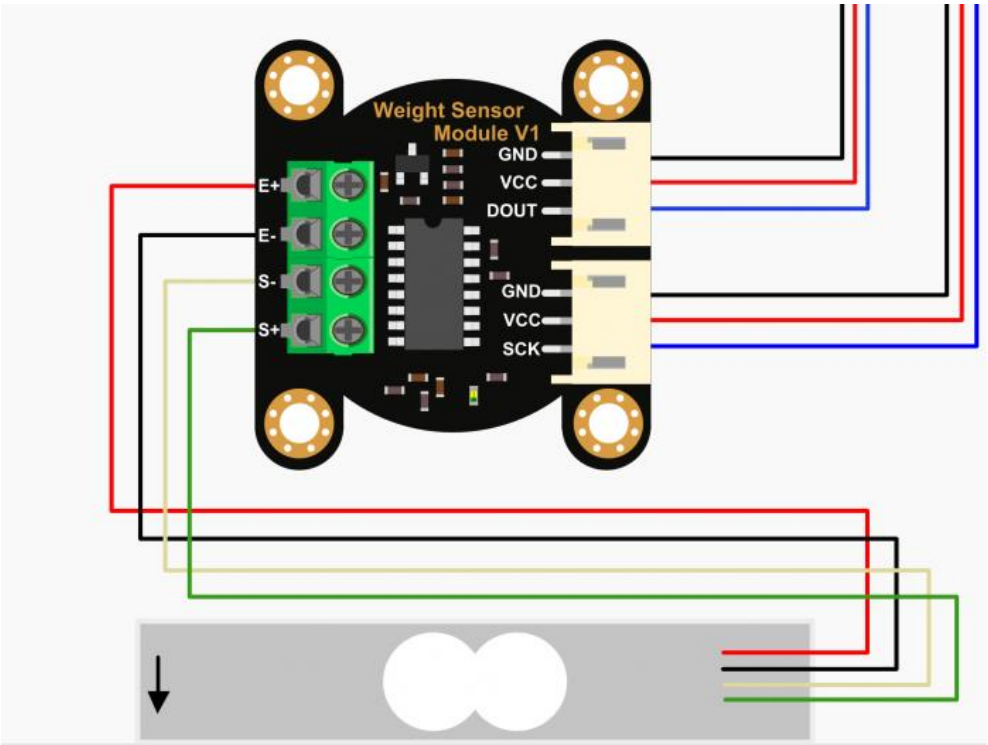

Figura 3.18. Conexiones del HX711 con una célula de carga. (DFRobot, 2016)

3.3.3. Módulo para el sensor de desplazamiento

En este apartado se va a explicar la parte diseñada para poder medir sensores de desplazamiento. Para estos sensores se exploran varias opciones como se verá más adelante en este capítulo 3. Para la versión 2 se escogió medir con calibres digitales, ya que en la versión 1 no se medía el desplazamiento, sino solo fuerza y aceleración.

Existen antecedentes de medición de calibres digitales con un microcontrolador, pero no existían con una Raspberry Pi, debido a la gran información que existe y la facilidad para este proyecto se ha diseñado un circuito de acoplamiento para este tipo de sensores.

Este calibre digital funciona a una tensión entre 1.5-1.8V, y la Raspberry Pi funciona a una tensión de 3.3V, aunque se alimente a 5V (su lógica interna funciona a 3.3V) por lo que hay que adaptar las señales que devuelve el calibre digital a 3.3V y se debe tener una alimentación de 1.5-1.8V para alimentar dicho calibre.

La primera opción que se plantea para adaptar las señales de lógica más baja con la lógica de la Raspberry Pi mediante un transistor NPN, y invirtiendo en software los bits que se leen por la entrada digital de la Raspberry Pi, ya que el transistor NPN cuando deja pasar un “1” lógico es porque existe un “0” en su puerta.

Mediante este circuito, cuando no hay señal en la línea de datos del calibre (0V) la tensión colector- emisor es nula por lo que a la Raspberry Pi llegarán 3.3V = 1 lógico. Sin embargo, cuando hay señal en la línea de datos (1,8V), sí que se tendrá tensión colector-emisor por lo que a la Raspberry Pi llegará muy poca tensión aproximadamente 0V = 0 lógico. Con la señal de reloj sucederá lo mismo. En la figura 3.18 se puede observar el circuito como opción 1.

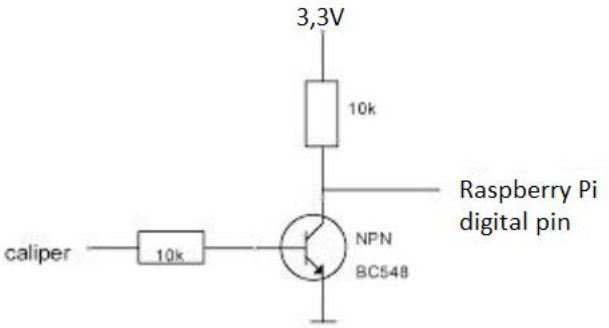

Figura 3.19. Circuito de acoplamiento para el sensor de desplazamiento con un transistor.

Como se observa mediante este circuito, se consigue acoplar la tensión del calibre y de la Raspberry Pi, pero como resultado se obtiene la señal tanto de datos como de reloj invertida; es decir, que si el calibre estuviera enviando: 1 000, 0000, 0000, 0000 = 0.00 mm. Raspberry Pi interpretaría: 0 111, 1111, 1111, 1111. Si se utiliza esta opción hay que tener en cuenta esto a la hora de realizar el diseño software.

Para este proyecto es más fácil escoger un conversor de niveles lógicos y mantener las señales sin invertir cuando llegue un “1” lógico entender que es un “1” y no un “0”.

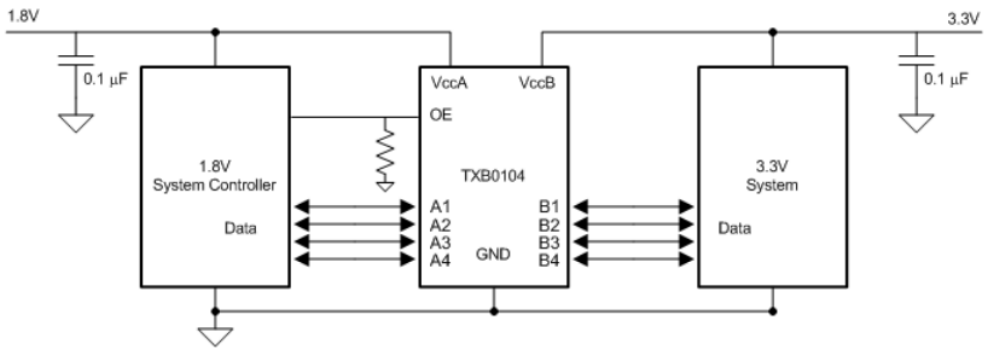

Figura 3.20. Acoplamiento de dos lógicas diferentes. (Instruments T. , ti.com TXB0104, 2016)

Para adaptar estas 2 lógicas de señales se va a usar un TXB0104 (traductor de voltaje de 4 bits bidireccional de Texas Instruments) con las siguientes características:

- Alimentación en el puerto A de 1.2V a 3.6V
- Alimentación en el puerto B de 1.65V a 5.5V
- Aislamiento de dichas alimentaciones
- Bajo consumo de 5µA
- 4 señales de entrada y salida
- Bidireccional

Un ejemplo de aplicación para 4 señales se puede ver en la figura 3.19.

Se usa un traductor de 4 bits, aunque se necesiten dos señales nada más, pero es debido a facilidades de diseño.

Para la alimentación del calibre digital se necesita una señal estable de 1.5 a 1.8V, por lo que la primera opción es mantener la pila en el calibre, pero como debe existir una tensión de 1.5-1.8V en la tarjeta diseñada debido a que el conversor de niveles lógicos anterior necesita esta alimentación en el puerto A, es más fácil elegir un regulador lineal para obtener esta tensión. Por lo que se ha elegido un TPS70918, regulador lineal de Texas Instruments. Este tiene las siguientes características:

- Baja corriente de consumo, 1µA
- Rango de voltaje de entrada, 2.7-30V, por lo que nos vale con una señal de 3.3V de la Raspberry Pi para obtener 1.8V
- Soporta 200mA de pico de corriente, nuestro calibre no tiene esos consumos
- Encapsulado SOT-23-5
- Pin de habilitado

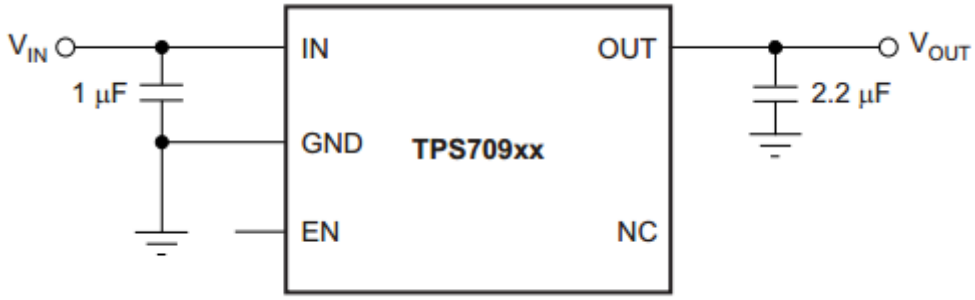

Figura 3.21. Ejemplo de aplicación TPS709. (Instruments T. , ti.com TPS709, 2016)

El circuito definitivo integrado en la tarjeta diseñada es el que se ve en la figura 3.21.

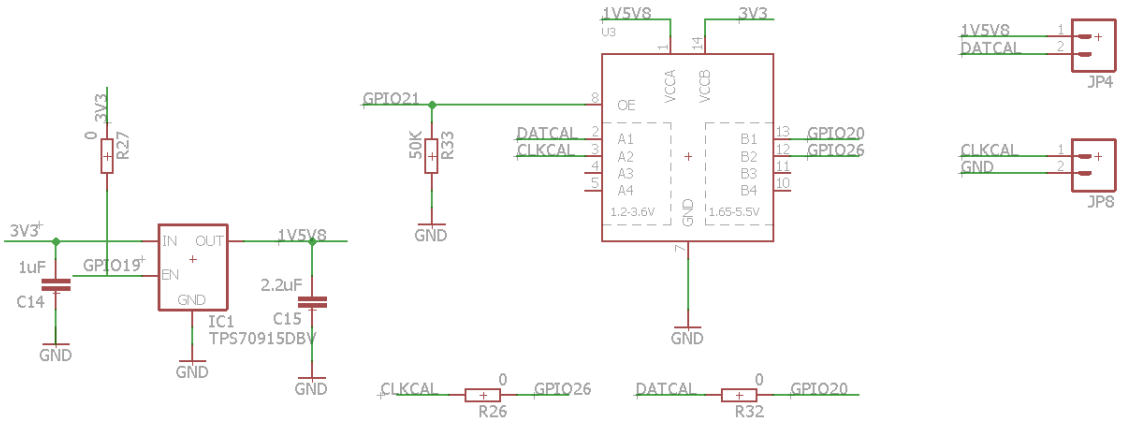

Figura 3.22. Esquema final módulo del sensor de desplazamiento.

3.4. Sensores seleccionados para el proyecto

En este apartado se va a realizar una discusión de porque se han seleccionado unos determinados sensores para la realización de este proyecto, así como sus principales características y como se van a acoplar con los módulos diseñados en la tarjeta de adquisición de datos.

Se va a dividir el apartado en 3 subapartados:

- Acelerómetros
- Sensores de Fuerza
- Sensores de desplazamiento

3.4.1. Acelerómetros

Queda claro que la elección para medir vibraciones son acelerómetros, ahora bien, ¿qué tipo de acelerómetros son los más correctos para el proyecto?, esta pregunta se va a resolver en este subapartado.

Para el presente proyecto se ha decidido utilizar acelerómetros MEMs en lugar de algún tipo de acelerómetro piezoeléctrico o piezorresistivo, ya que estos son más baratos y más pequeños para integrar en maquetas de estructuras, además se ha visto a través de ensayos que dan muy buenos resultados con respecto al precio que tienen.

En principio se optó por seleccionar unos acelerómetros MEMs fabricados ya para un proyecto antecedente en la pasarela Pedro Gómez Bosque, como se ha hablado en el capítulo 2 del presente documento. Este sensor es el ADXL327 desarrollado por Analog Devices, las características son las siguientes:

- Es muy pequeño
- Bajo consumo
- Acelerómetro de 3 ejes con las salidas acondicionadas en tensión
- Puede medir aceleraciones estáticas de gravedad o aceleraciones dinámicas
- Rango de medición de hasta ± 2.5g
- Sensibilidad de hasta 500 mV/g

- Ancho de banda de hasta 550 Hz, se deben implantar condensadores a la salida de cada eje para seleccionar el ancho de banda

Aunque tiene un gran inconveniente, las grandes distancias, ya que este proyecto está orientado principalmente a monitorizar maquetas estructurales de laboratorio, pues este no es un inconveniente. Pero la solución a este problema sería introducir un amplificador operacional a la salida de cada señal con el fin de reducir la impedancia del canal y así reducir el ruido también.

La figura 3.22 muestra un diagrama del circuito acondicionador para un acelerómetro el cual consiste en un acelerómetro MEMS ADXL327 (1), 3 condensadores seleccionados para un ancho de banda de 100Hz (2), un regulador de voltaje (3), un amplificador operacional a cada salida (4) y un led de test (5).

Como se puede observar el amplificador operacional a cada salida ya se ha colocado en la entrada de la tarjeta de adquisición de datos.

En esta aplicación se ha decidido usar unas tarjetas diseñadas por Adafruit, ya que así no se tienen que diseñar tarjetas de nuevo. Existen multitud de tarjetas con acelerómetros y su correspondiente acondicionamiento ya diseñadas y a la venta, por lo que se ha optado por esta opción.

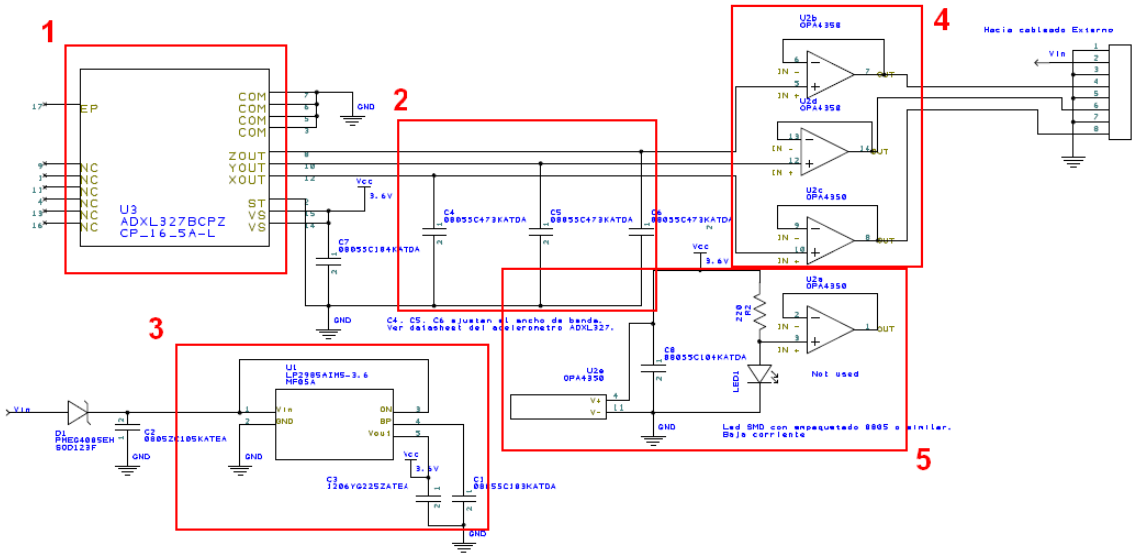

Figura 3.23. Circuito acondicionador para un acelerómetro ADXL327

El acelerómetro MEMs seleccionado ha sido un ADXL335 de Analog Devices que tiene las siguientes características:

- Es muy pequeño, también como el ADXL327
- Bajo consumo
- Acelerómetro de 3 ejes con las salidas acondicionadas en tensión
- Puede medir aceleraciones estáticas de gravedad o aceleraciones dinámicas
- Rango de medición de hasta ± 3g, un poco más amplio que el ADXL327
- Ancho de banda de hasta 500 Hz, la tarjeta inicialmente posee amplificadores de 0.1µF con lo que se obtiene un ancho de banda de 50 Hz, por lo que si se

cambian estos condensadores a 0.01µF se obtiene el ancho de banda máximo

Tener en cuenta, que los parámetros que ofrece la hoja de datos del fabricante están testeados para una alimentación de 3V, pero se va a alimentar la placa a 3.3V por lo que habrá que calcular los nuevos parámetros de sensibilidad para este acelerómetro.

La salida del ADXL335 es radiométrica, por lo que la sensibilidad de salida varia proporcionalmente a la alimentación. A 3.6V de alimentación la sensibilidad es de 360 mV/g, a 2V de alimentación la sensibilidad es de 195 mV/g. El señal de "cero g" también varía con la alimentación por lo que suele ser $\frac{V_S}{2}$, siendo V_S la tensión de alimentación.

El ruido de salida no es radiométrico, pero es absoluto en voltios, por lo que la densidad de ruido decrece al aumentar la tensión de alimentación. Esto es debido a que el factor de escala (mV/g) aumenta mientras que el ruido se mantiene constante. En $V_S = 3.6V$, la densidad del ruido del eje X y el eje Y es típicamente 120 $\mu g/\sqrt{Hz}$ cuando $V_S = 2V$, la densidad del ruido del eje X y el eje Y es normalmente 270 $\mu g/\sqrt{Hz}$.

La respuesta en g del auto test que se le puede realizar al acelerómetro es aproximadamente proporcional al cuadrado de la tensión de alimentación, sin embargo cuando se tiene en cuenta la radiometría de sensibilidad con respecto a la tensión de alimentación, la respuesta del test en voltios es aproximadamente proporcional al cubo de la tensión de alimentación, por ejemplo a $V_S = 3.6V$, la respuesta del test para el eje X es -560mV, para el eje Y 560mV y para el eje Z 950mV. A $V_S = 2V$, la respuesta para el test en el eje X es de -96mV, en el eje Y es de +96mV y en el eje Z es de -63mV. También tener en cuenta que la corriente de consumo disminuye al disminuir la tensión de alimentación. (Devices, 2016)

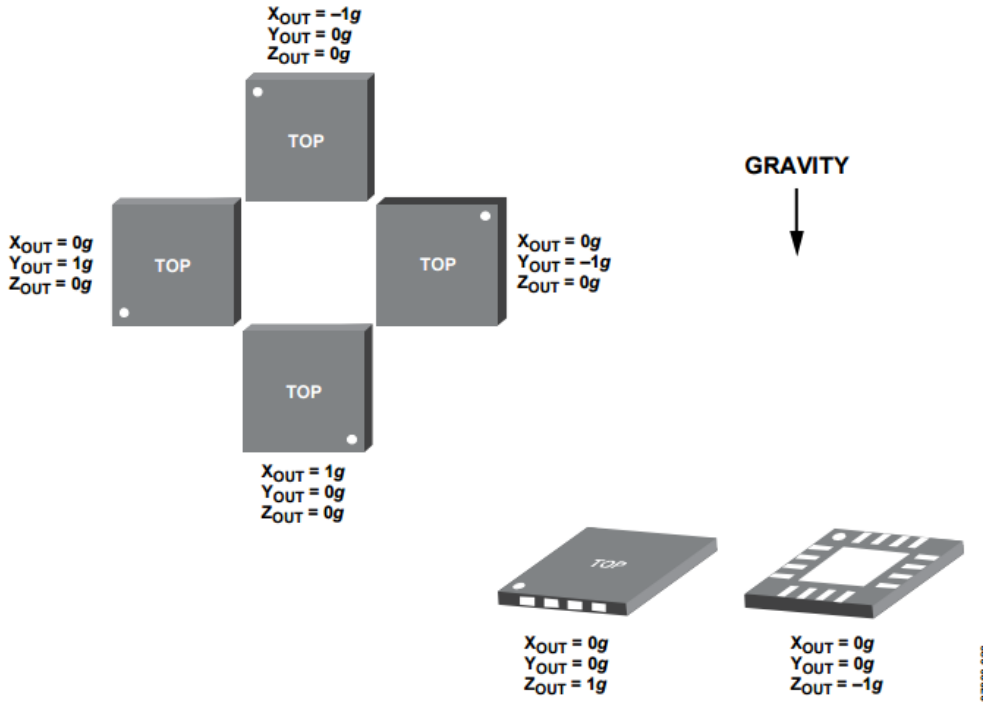

Figura 3.24. Respuesta de salida frente a la orientación a la gravedad. (Devices, 2016)

Para calcular nuestra sensibilidad aproximada del acelerómetro y poder así calibrarle y obtener los valores en "g" transformando la tensión se ha obtenido una recta a partir

de los datos del fabricante.

Con estos datos se obtiene una recta cuya ecuación es:

$$y = 103{,}13x - 11{,}25$$

$$Sensibilidad = 103{,}13 \cdot 3{,}3 - 11{,}25 = 329{,}079\ mV/g \approx 329\ mV/g$$

Luego con esta sensibilidad ya se puede calibrar nuestro acelerómetro en función de la salida en tensión que obtengamos a partir de los conversores analógico digitales.

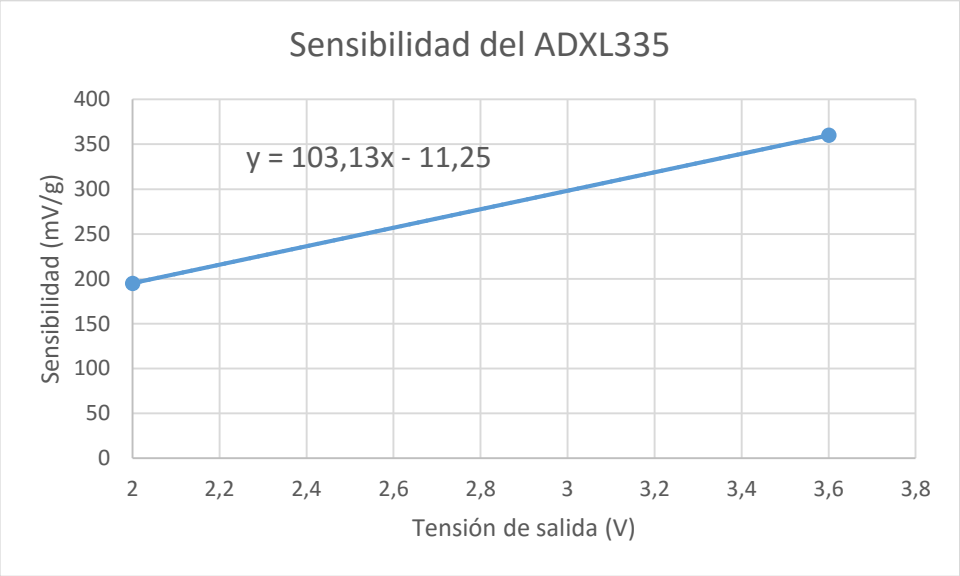

Figura 3.25. Grafica de sensibilidad ADXL335

En la siguiente figura se puede ver una foto de la tarjeta para el acelerómetro ADXL335.

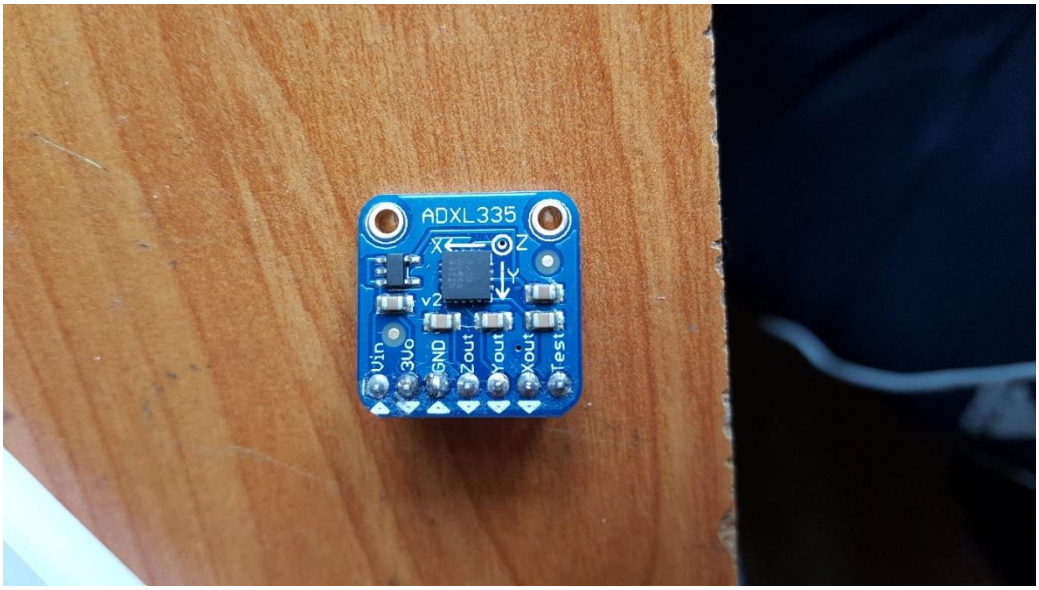

Figura 3.26. Foto de la tarjeta del acelerómetro ADXL335

Se puede ver en la tarjeta que hay 7 pines, de los cuales se van a usar 5, puesto que el pin de “Test” y “Vin” carecen de interés. Se tienen los dos pines de alimentación y la salida de tres señales analógicas las cuales se van a introducir a un convertidor analógico digital, y como se comentó anteriormente estas señales de tensión se tienen

que traducir a “g” por lo que se conoce que al estar alimentado el acelerómetro a 3.3V exactos (regulador de tensión en la placa) la salida de 0g seria 1.65V, 3g serían 3.3V y -3g serían 0V.

3.4.2. Sensor de fuerza

Para el sensor de fuerza se pensaron varias opciones, entre las primeras estaba un dinamómetro que se usó en anteriores proyectos, pero los resultados no eran demasiado buenos.

El funcionamiento del dinamómetro se basa en un puente Wheatstone, este se utiliza para medir resistencias desconocidas mediante el equilibrio de los brazos del puente. Estos están constituidos por cuatro resistencias que forman un circuito cerrado, siendo una de ellas la resistencia a medir.

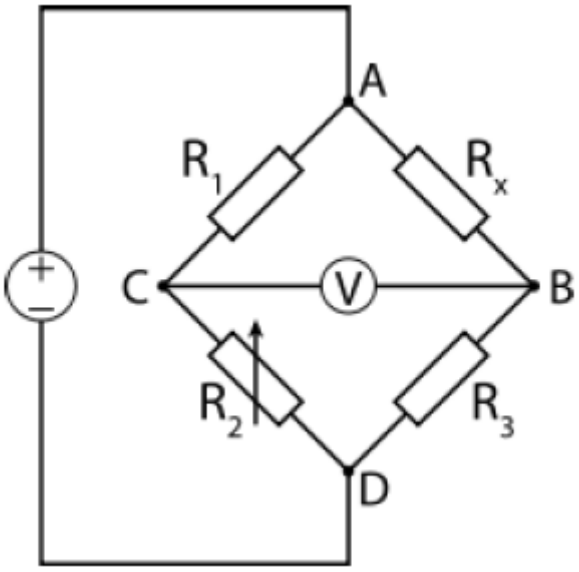

Figura 3.27. Disposición del puente de Wheatstone.

La figura 3.26 muestra la disposición eléctrica del circuito. En ella vemos que, Rx es la resistencia cuyo valor queremos determinar, R1, R2 y R3 son resistencias de valores conocidos, además la resistencia R2 es ajustable. Si la relación de las dos resistencias del brazo conocido (R1/R2) es igual a la relación de las dos del brazo desconocido (Rx/R3), el voltaje entre los dos puntos medios será nulo y por tanto no circulará corriente alguna entre esos dos puntos C y B.

Para efectuar la medida lo que se hace es variar la resistencia R2 hasta alcanzar el punto de equilibrio. La detección de corriente nula se puede hacer con gran precisión mediante el voltímetro V.

La disposición en puente es ampliamente utilizada en instrumentación electrónica. Para ello, se sustituyen una o más resistencias por sensores, que al variar su resistencia dan lugar a una salida proporcional a la variación de la variable a medir (lineal). A la salida del puente suele colocarse un amplificador para reducir la impedancia de la señal, así como el ruido.

A mayor fuerza ejercida sobre el sensor, menor será la resistencia y por tanto mayor será la salida en tensión del puente.

En un dinamómetro entonces se pueden observar cuatro señales:

- Alimentación +
- Alimentación –
- Señal + del puente de Wheatstone
- Señal - del puente de Wheatstone

La segunda opción que se planteó era utilizar sensores de fuerza resistivos (FSR), estos están formados de una película gruesa de polímero la cual presenta una disminución de resistencia en función de la carga aplicada en su superficie.

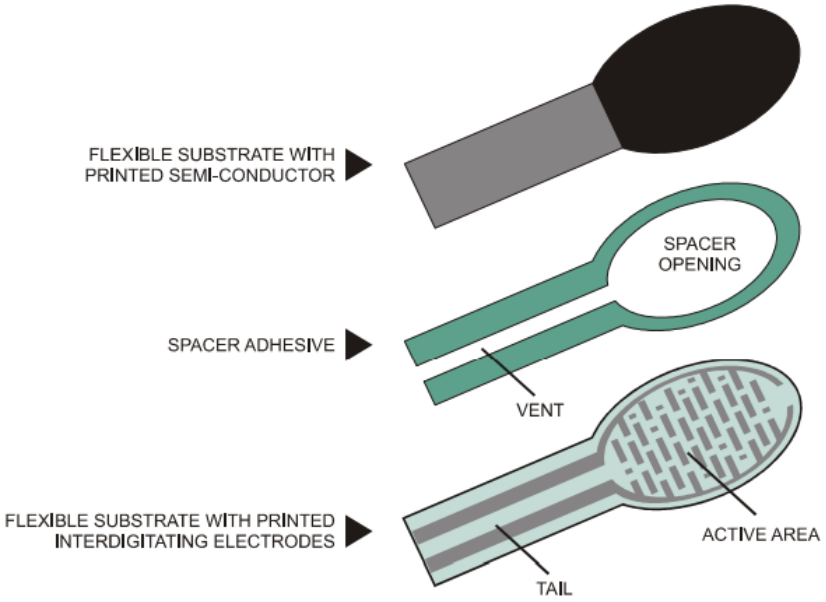

Figura 3.28. Construcción de un FSR. (Adafruit, 2016)

Las FSRs no son como las células de carga o galgas extensiométricas, aunque tienen propiedades similares, el problema de las FSRs es que no son adecuadas para mediciones de precisión.

La característica de resistencia frente a fuerza se muestra en la Figura 3.29, esta proporciona una visión general del comportamiento de la respuesta de este tipo de sensores. Para mayor comodidad la fuerza frente al desplazamiento se representa es escala logarítmica.

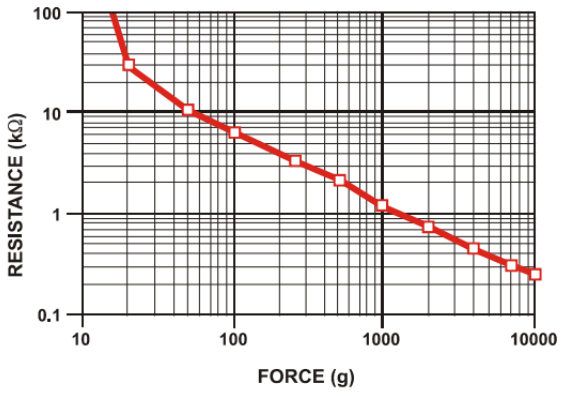

Figura 3.29. Resistencia vs Fuerza en un sensor FSR. (Adafruit, 2016)

Haciendo referencia a la Figura 3.29. Resistencia vs Fuerza en un sensor FSR. , se ve que la respuesta entre 100KΩ y 10KΩ no sigue un comportamiento lineal con respecto a la parte de la gráfica por debajo de los 10KΩ.

Por lo general, el tamaño y la forma de los sensores FSR son los parámetros que limitan la integración de los mismos.

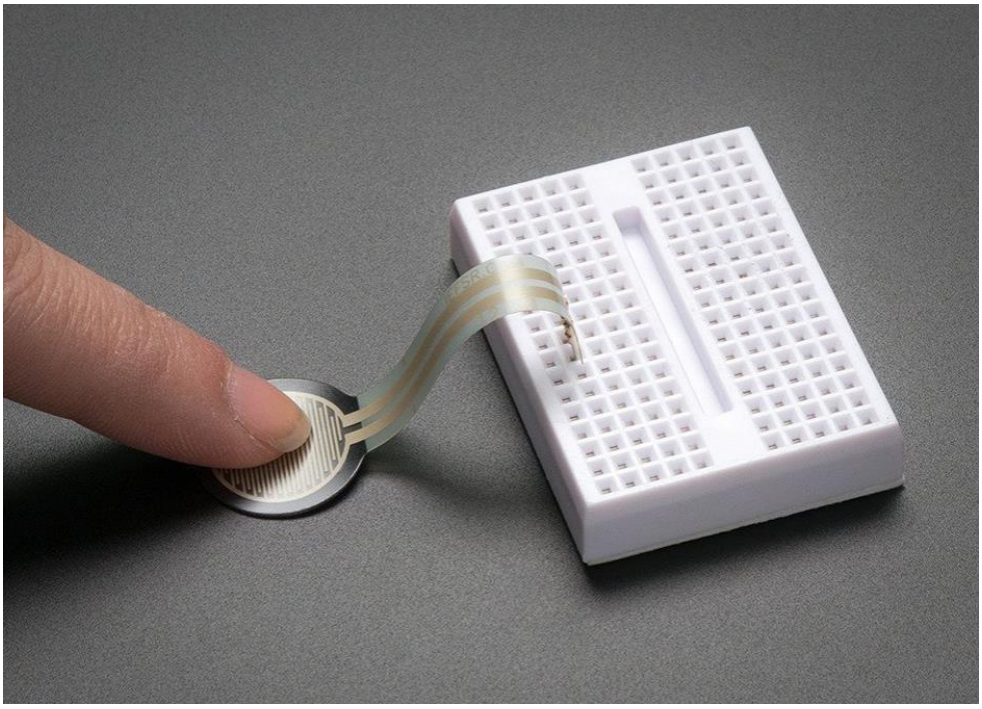

Figura 3.30. Imagen de un FSR.

Los rangos de fuerza de este tipo de sensor que se ven en la figura 3.28 aproximadamente son de 0 a 100N, y los rangos de resistencias se mueven de 100KΩ a 200Ω. Este tipo de sensores miden en la zona inestable que se veía en la Figura 3.29. Resistencia vs Fuerza en un sensor FSR. de 20g a 100g, luego que para este rango de pesos no es adecuado. (Adafruit, 2016)

Por último, la opción escogida para nuestro proyecto son células de carga (galgas extensiométricas), las cuales se pueden encontrar con una gran variedad de rango de fuerza y dan muy buenos resultados. Esta elección surge de la investigación en sensores instalados en estructuras reales, con lo que se decide elegir esta opción puesto que es la más usada en este tipo de monitorizaciones.

En este apartado se va a explicar este tipo de células con una de 0.78Kg de capacidad, se han probado para el proyecto células de 1 Kg y de 5Kg. En la figura 3.30 se puede ver una foto de una célula de carga para ver el aspecto que tienen este tipo de sensores.

Una célula de carga es un módulo de detección fuerza, es una estructura de metal diseñada cuidadosamente con pequeños elementos llamados galgas extensiométricas o medidores de deformación que están montados en lugares específicos del conjunto de metal.

Las células de carga están diseñadas para medir la fuerza en una dirección determinada e ignorar las fuerzas en otras direcciones. La señal de salida en tensión de estas células es muy pequeña por lo que requieren una amplificación

especializada.

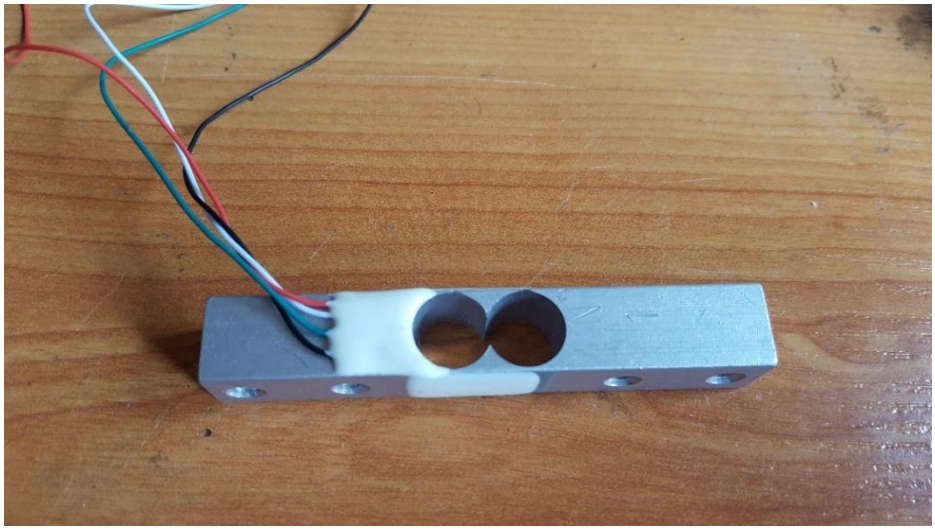

Figura 3.31. Foto de una célula de carga.

Las células de carga convierten la carga que actúan sobre ellas en señales eléctricas, la medida se realiza con pequeños medidores de deformación o galgas como se comentaba anteriormente. Estos medidores se deforman y varia su resistencia en función de la deformación a la que son sometidos.

El efecto de la temperatura es muy importante en este tipo de sensores, ya que a veces influye más que la propia deformación. El método más importante de compensación de temperatura implica el uso de múltiples medidores de deformación, en el que todos responden al cambio de temperatura con un cambio de resistencia.

Algunos diseños incluso utilizan medidores que no están sometidos a ninguna fuerza con lo que contrarrestan los efectos de la temperatura. La mayoría de los diseños utilizan cuatro medidores de deformación, unos en compresión y otros en tensión lo que maximiza la sensibilidad de la célula de carga y se cancela el efecto de la temperatura.

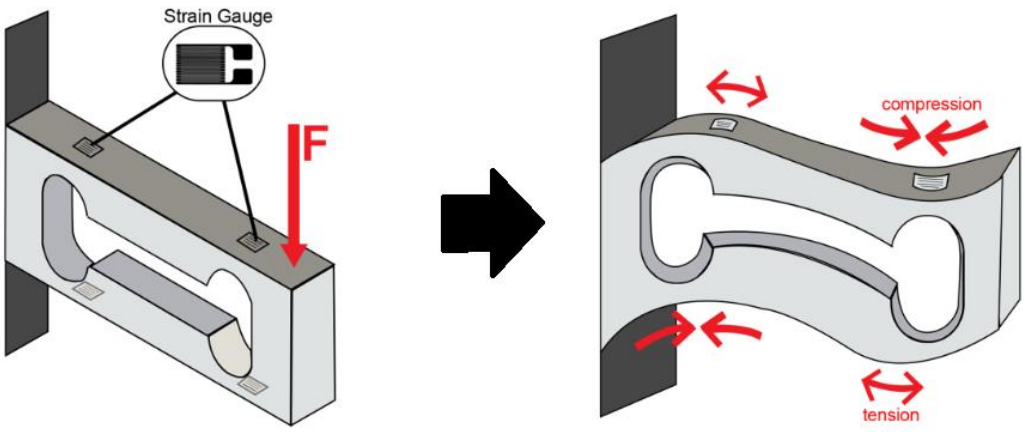

Figura 3.32. Medidores de deformación en una célula de carga. (Sensors, 2016)

Las células de carga pueden ser resistivas o capacitivas, las resistivas que son las que se van a usar en el presente proyecto se basan en el efecto piezorresistivo con lo que varía su resistencia en función de la carga aplicada, esto es lo que se comentaba anteriormente, que se colocan varios medidores de deformación (piezorresistivos) a lo largo de la estructura de acero del sensor.

En la célula de carga que se muestra en la figura 3.31 existen un total de cuatro medidores de deformación que están unidos a las superficies inferior y superior de la célula de carga.

Cuando se aplica una carga al cuerpo de la célula de carga resistiva como se ve en la figura 3.31, el elemento elástico se deforma y crea una tensión (esfuerzo) en esos lugares debido a la carga aplicada. Como resultado dos de los medidores de tensión están a compresión, mientras que los otros dos están a tracción.

Las cuatro galgas extensiométricas son configuradas en un puente de Wheatstone como cuatro resistencias (véase figura 3.32), un voltaje de excitación es aplicado en las esquinas del puente (A y C). La diferencia de tensión se mide entre las otras dos esquinas (B y D). En el equilibrio sin carga aplicada la salida de voltaje es cero o muy cercana a cero ya que el valor de las cuatro resistencias es muy parecido. Es por eso que se conoce como circuito de puente equilibrado.

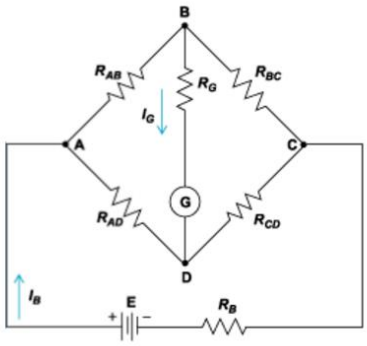

Figura 3.33. Puente Wheatstone en una célula de carga. (Sensors, 2016)

Cuando el cuerpo metálico al que los medidores de deformación están unidos, se deforma por la aplicación de una fuerza, la tensión resultante conduce a un cambio en la resistencia en una o más de las galgas. Este cambio en la resistencia resulta en un cambio en la tensión de salida, estos cambios suelen ser del orden de milivoltios con lo que se puede amplificar y digitalizar esta medida para poder tratarla. (Sensors, 2016)

3.4.3. Sensor de desplazamiento

En este subapartado se va a exponer las diferentes opciones planteadas para medir desplazamiento y la opción seleccionada para ello.

Como primera opción se plantea un encoder lineal, con el cual se puede conocer en qué posición nos encontramos en cada momento y así monitorizar el desplazamiento de una viga.

Se estudia un encoder lineal magnético del fabricante AMS, este fabricante se dedica en su mayoría a fabricar encoders magnéticos tanto lineales como rotatorios, se elige este fabricante por la facilidad de desarrollar ya que facilita muestras gratuitas y se le

conoce de anteriores proyectos.

El encoder lineal es el NSE-5310 con su correspondiente banda magnética de 30 cm. Este encoder tiene las siguientes características:

- Sensor de posición incremental
- Un array de elementos hall son usados para determinar la posición incremental
- Detecta los extremos de la banda magnética para fijar un cero de referencia
- Señal digital de salida I2C
- 0.488 µm
- ±10 µm de error absoluto
- Sensor de temperatura integrado
- Encapsulado TSSOP-20

Un ejemplo de aplicación de este encoder lineal con su correspondiente banda magnética es el que se ve en la figura 3.34. Esta imagen muestra el NSE-5310 montado sobre una placa PCB, donde una banda magnética multipolo se coloca por encima del sensor.

Figura 3.34. Ejemplo de aplicación del NSE-5310. (AMS, 2016)

Esta opción se descarta por la dificultad de calibrar y controlar este tipo de encoders, con lo que se debería hacer una calibración exacta con equipos de alto coste (AMS, 2016). Con lo que se opta por una opción más barata y simple de controlar como es un calibre digital.

Lo primero es desmontar el calibre por completo para intentar establecer qué principio se utiliza para detectar los cambios de posición. Una vez desmontado se demuestra que el calibre utiliza la capacitancia para detectar el movimiento; es decir, la parte fija y móvil del calibre forman una serie de condensadores. Estos calibres digitales son sensores capacitivos.

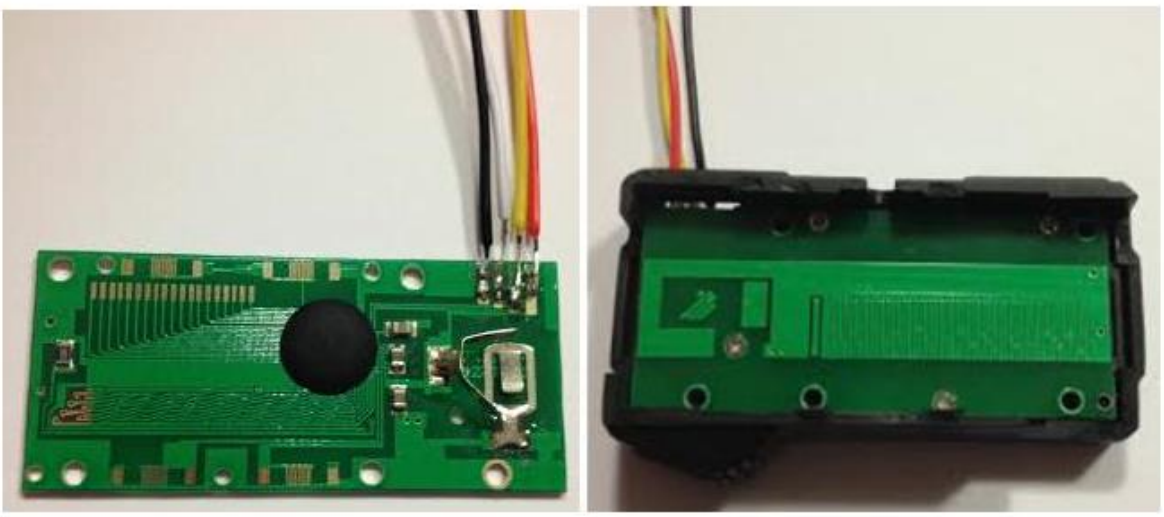

Figura 3.35. Interior de un calibre digital.

Una vez desmontado el calibre se es consciente de que se puede utilizar desmontado para utilizarlo en las estructuras que se quieren monitorizar, por lo que se deberá encontrar la manera de utilizarlo completo. Se puede observar que consta de dos tapas en su estructura de plástico, una que alberga la pila de 1,5 V que lo alimenta y otra en la que asoman cuatro pines como se muestra en la figura siguiente.

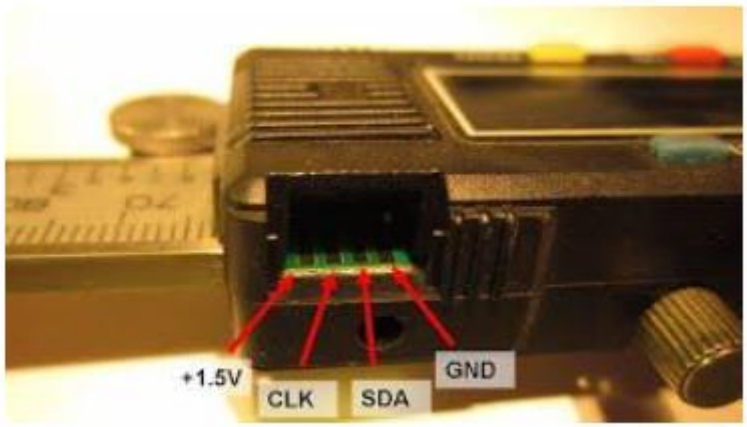

Figura 3.36. Pines digitales de un calibre.

Al documentarse sobre este tema se descubre que estos pines resultan ser salidas digitales del calibre que se pueden utilizar para conectarlo a nuestra Raspberry Pi. Como se ve en la figura 3.36 cada uno de ellos corresponde de izquierda a derecha a: 1,5V alimentación, pin de reloj, pin de datos y tierra. El orden de estos pines debe ser el mismo para la mayoría de calibres, aunque lo mejor es comprobarlo con un osciloscopio o un multímetro para asegurarse.

Por lo tanto, ésta será la manera en que se utilizará el calibre para este proyecto, aprovechando estas salidas digitales de las que se dispone. Para poder utilizar estos pines digitales lo primero que se debe hacer es soldar unos cables a cada uno de ellos, como se muestra en la figura 3.37.

Figura 3.37. Cables soldados al calibre digital.

Si se conectan los pines de datos y de reloj a un osciloscopio para ver qué datos se transfieren para 0.0mm (invertido) se obtienen las señales que se observan en la figura 3.38.

Como se puede ver, el calibre envía sus datos a ráfagas de 100ms por lo que se tendrán 10 mediciones por segundo. La señal de reloj envía una serie de 24 pulsos y luego permanece a nivel alto (1,5V) durante un tiempo. Durante esta ráfaga de 24 pulsos la señal de datos envía pulsos cuando la señal de reloj pasa de nivel alto a nivel bajo por lo que el calibre envía 24 bits de datos en las bajadas del pulso de reloj de la siguiente manera (del bit menos significativo LSB al más significativo MSB):

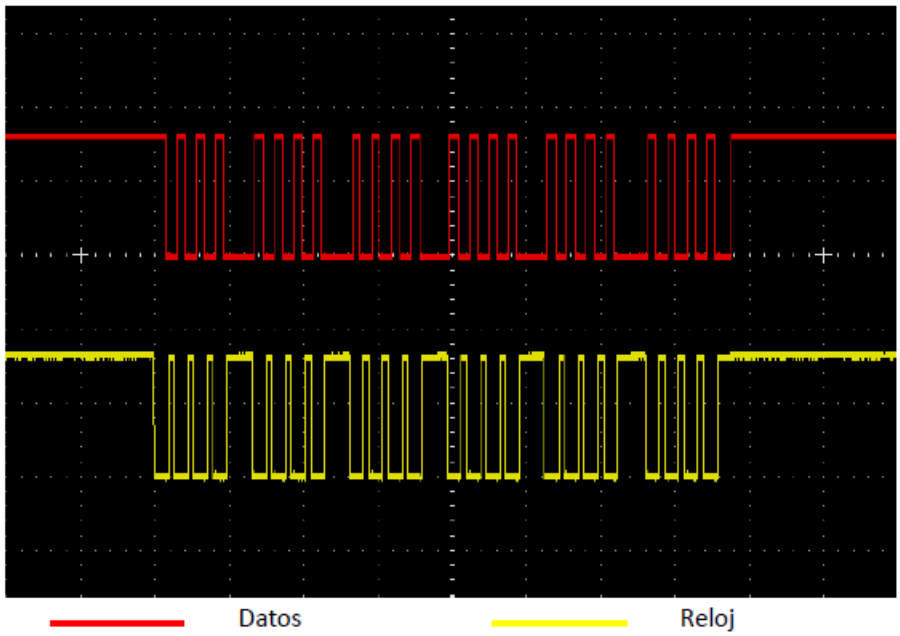

Figura 3.38. Señales de datos y reloj en un osciloscopio.

De la serie de 24 bits enviada el primer bit se omiten los tres primeros bits, y el bit 21 se corresponde con el bit de signo para el que si éste es 1 se tratará de un número negativo y si es 0 de un número positivo.

3.4.4. Sensor de temperatura y humedad

Por ultimo está el sensor de temperatura y humedad, para tal propósito se va a emplear un HDC1008 de Texas Instruments.

Este sensor ofrece una interfaz de comunicación I2C, con la cual se comunicará la Raspberry Pi para solicitar los registros de temperatura y humedad. Las principales características de este sensor son las siguientes:

- Tensión de alimentación de 3V a 5V
- 14 bits de resolución
- Exactitud en la humedad relativa de ±4%
- Exactitud en la temperatura de ±0.2ºC
- Consumo de corriente de 1.2µA de media
- Interfaz I2C

El sensor final utilizado con su tarjeta correspondiente es el que se muestra en la figura 3.39.

Para la medida de temperatura y humedad se puede usar cualquier sensor con una interfaz digital o incluso analógica con su correspondiente acondicionamiento ya que existen multitud de sensores de este tipo en el mercado, pero para el presente proyecto se ha seleccionado este debido a que ha sido utilizado en proyectos previos y se tienen librerías para la programación del mismo en la Raspberry Pi.

Para la lectura de los registros, tanto de temperatura como de humedad, pueden consultarse las direcciones de estos en la hoja de datos del fabricante. En la estructura se pueden colocar varios sensores ya que dispone de una posibilidad para cambiar la dirección I2C que posee en el bus el sensor, con lo que con esta opción se pueden

tener hasta cuatro sensores de este tipo en la estructura.

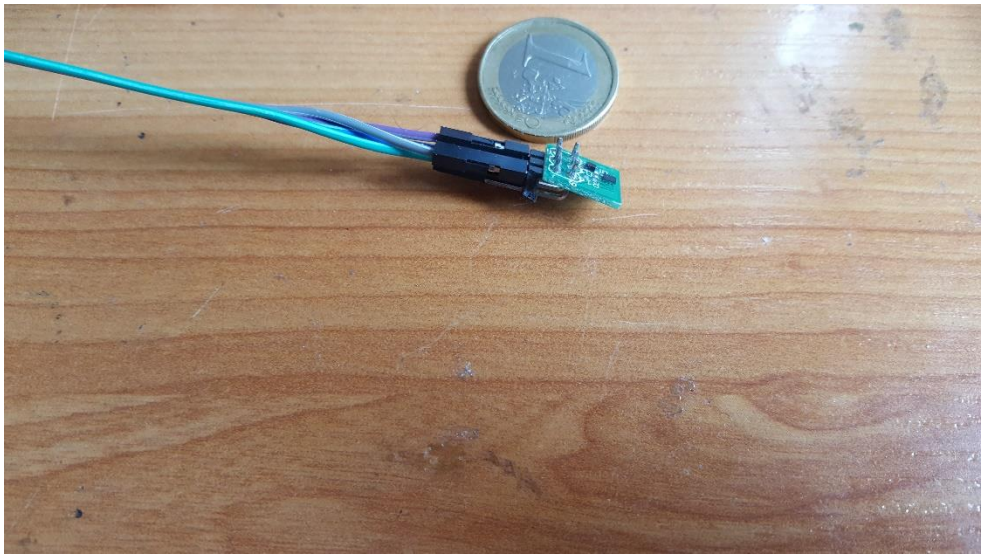

Figura 3.39. Sensor de temperatura y humedad HDC1008

3.5. Imágenes del sistema hardware finalizado

En la figura 3.40 se puede observar la imagen de la versión de la tarjeta de adquisición de datos diseñada para la Raspberry Pi.

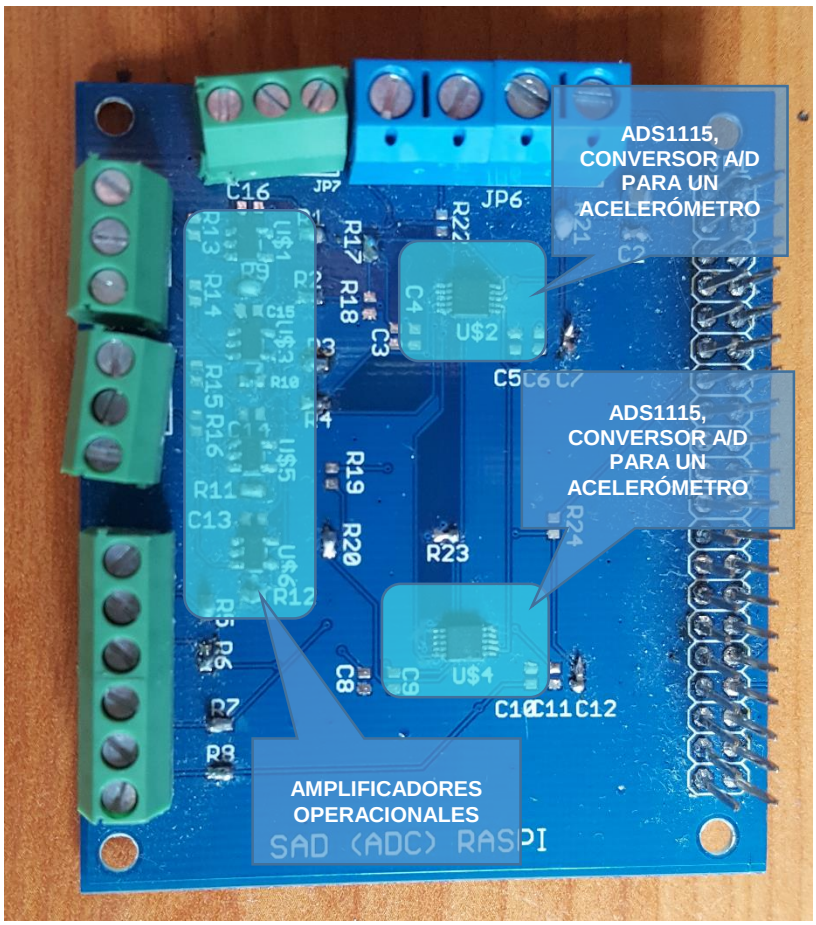

Figura 3.40. Tarjeta de adquisición de datos versión 1

Como se puede en esta versión solo existen dos conversores analógico digitales para

los acelerómetros, con esta versión solo se monitoriza la aceleración y la fuerza mediante una placa ya fabricada con el conversor HX711 que se ha comentado anteriormente en el presente capitulo. En esta versión se puede observar que sobran pines ya que por un fallo de diseño las separaciones de los pines de los conectores eran incorrectas y se tuvo que soldar conectores de 3 pines donde eran de dos.

En la siguiente figura se puede observar una imagen de la versión dos de la tarjeta de adquisición de datos.

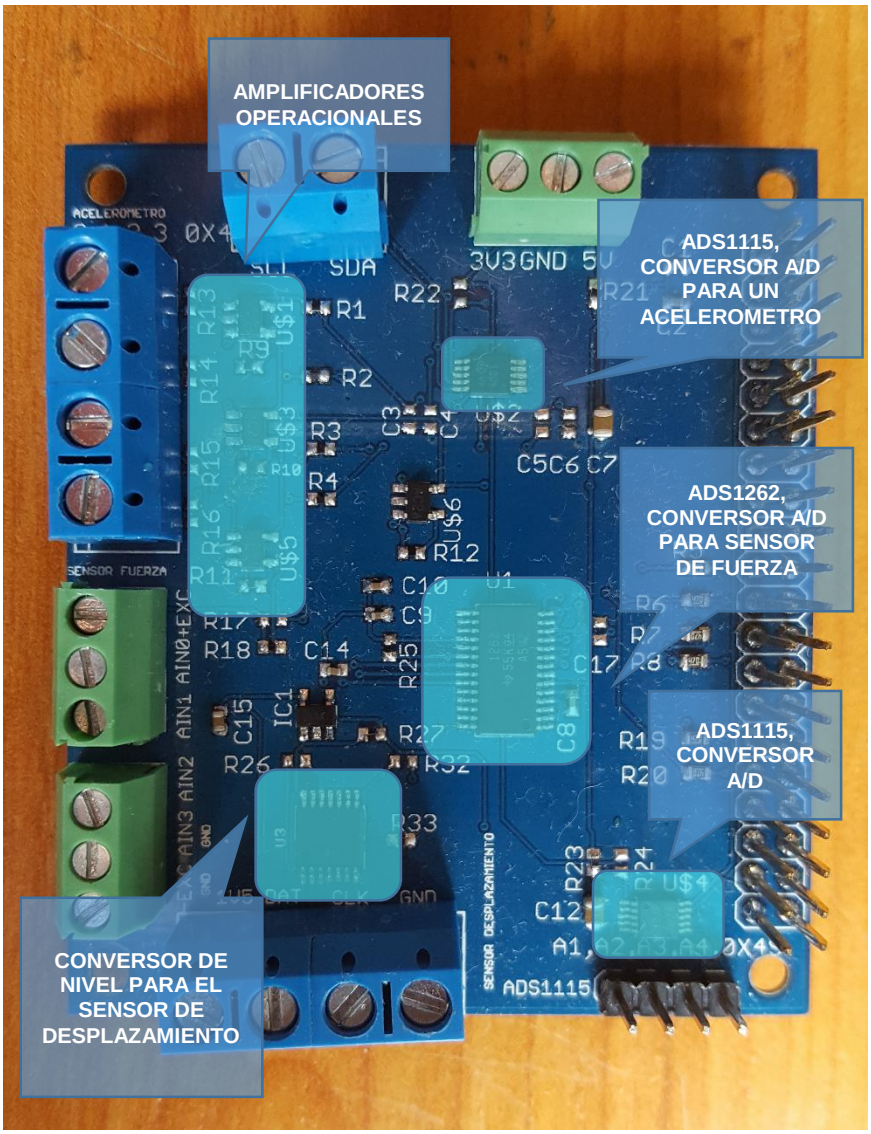

Figura 3.41.Tarjeta de adquisición de datos versión 2

En esta versión ya se puede ver que si se puede medir desplazamiento y fuerza incluso con el conversor analógico digital integrado en la tarjeta. Se puede ver también como se ha corregido el error del espaciado de pines de los conectores, con lo que ahora están bien espaciados los conectores.

Por último, se ve en la figura 3.42 como queda la tarjeta montada encima de la Raspberry Pi.

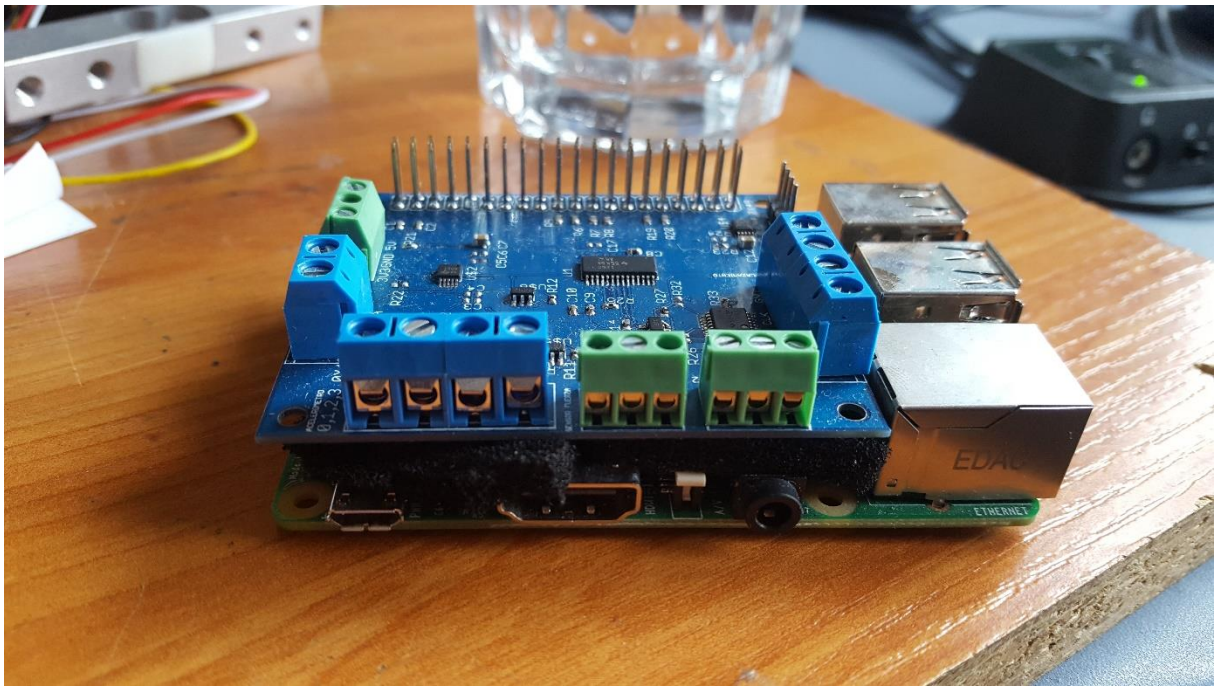

Figura 3.42. Imagen Raspberry Pi + Tarjeta de adquisición de datos

3.6. Herramientas y métodos para diseñar circuitos

Lo primero es describir la herramienta utilizada para diseñar los circuitos electrónicos utilizados en el presente proyecto. El software seleccionado para este propósito es Eagle.

Eagle son las siglas de easily applicable graphical layout editor. Se trata de un programa de diseño de diagramas y PCB con ordenador. Se trata de un programa que dispone de licencias 'freeware', además de una gran cantidad de librerías de componentes por la red. Básicamente, está compuesto de dos módulos.

- Editor de esquemas
- Diseño de circuitos impresos

Aunque estos módulos se basan en librerías de componentes, los cuales tienen asociados un package (para la PCB) y un dibujo que se utiliza en los esquemas electrónicos.

A continuación, en la figura 3.43 se puede ver una captura de pantalla de la interfaz de desarrollo de esquemáticos con Eagle.

También en la figura 3.44 se ve la interfaz de desarrollo de PCB con la herramienta mencionada.

Esta herramienta de diseño hardware CAD/CAM tiene las siguientes características:

- Fácil edición de las librerías de los componentes
- Posibilidad de rutado automático
- Soporte completo en smd

En cuanto a la posibilidad de rutado automático, en este proyecto se ha decidido rutar manualmente ya que hay que ser preciso con la longitud de las pista.

Básicamente por la fácil edición de componentes y su gran cantidad de librerías en

internet se decide utilizar este programa para la creación del hardware del proyecto.

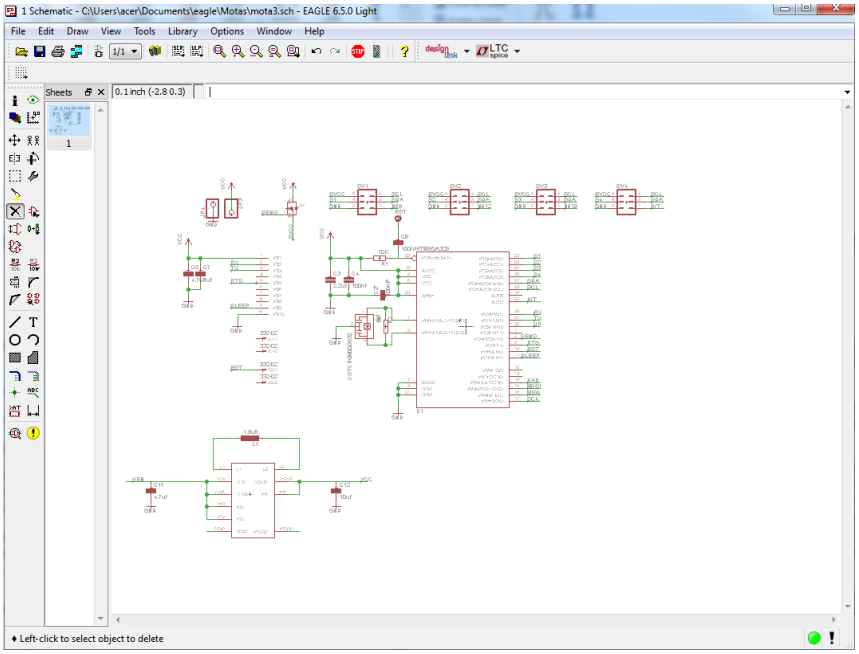

Figura 3.43. Interfaz para el desarrollo de esquemas Eagle.

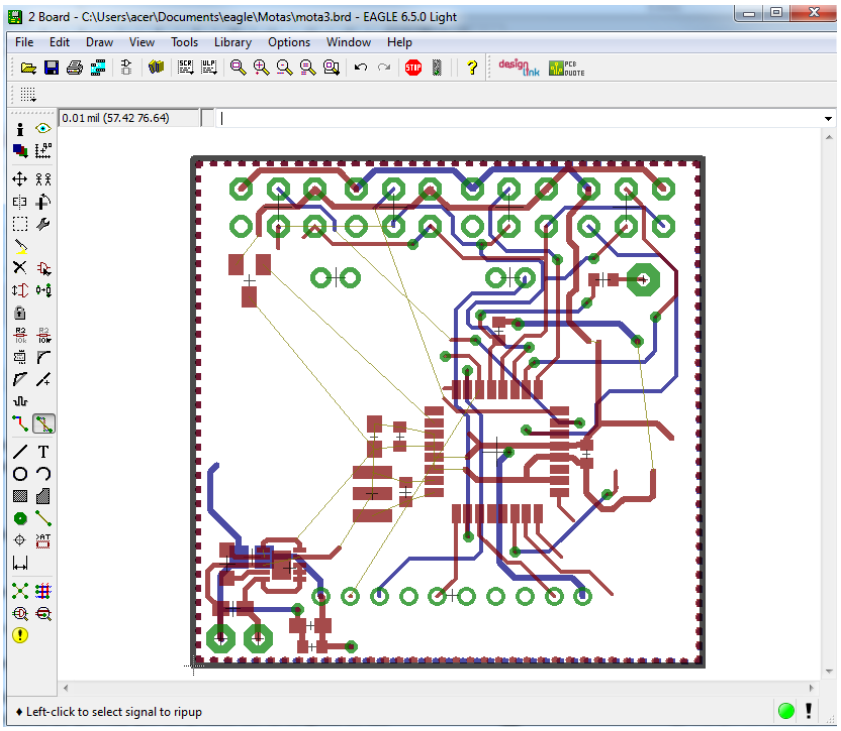

Figura 3.44. Interfaz para el desarrollo de PCBs de Eagle.

Una vez que se han creado los esquemas y se han rutado los circuitos impresos, hay que fabricarlos.

Para ello se puede optar por la fabricación manual, es decir, se realiza el revelado de

las placas de circuito impreso. Pero ello implica que hay que ser muy rigurosos y tener un cuidado muy especial. Esto se debe a que alguna de las pistas es muy fina, pudiendo desaparecer si no se revela de manera correcta la placa de circuito impreso.

La otra opción posible es mandarlas fabricar a una empresa externa, donde disponen de los medios necesarios para fabricarlas sin que pueda desaparecer alguna pista. Además de esto, la presentación y el resultado final es más atractivo a la vista que las placas realizadas de manera manual.

Para que las empresas fabriquen las PCB, lo que hay que hacer es mandar todos los archivos 'gerbers'. Estos archivos contienen todo lo necesario para la fabricación del circuito impreso. Contienen todos los rutados creados con el Eagle, la ubicación de los componentes, nombres y valores de los mismos, 'pads', etc.

En el presente proyecto se ha tomado esta última opción y se ha decidido fabricarlas en una empresa externa.

Debido a que se intenta que el sistema creado tenga el menor tamaño posible, se utilizan componentes en tamaño SMD, es decir, con tamaño mucho menor al habitual, de dimensiones milimétricas.

Por ello, para soldarlos hay que tener un especial cuidado para que queden en la posición correcta. Para la sujeción de los mismos se utilizan unas pinzas cuya punta es minúscula, para obtener mayor precisión.

Para soldarlo se utiliza una estación de soldadura, la cual es una pistola de aire caliente que calienta el estaño y lo funde. Hace la misma función que un estañador, pero con aire por lo que no necesitas tener contacto con el componente a la hora de soldar. A continuación, en la figura 3.45 se muestra las estaciones utilizadas para la soldadura de las PCB.

Figura 3.45. Estación de soldadura

El estaño es reemplazado por pasta para soldar, la cual funciona muy bien con las pistolas de aire y los resultados obtenidos son al menos iguales que con el estaño. La pasta está en un estado líquido pero denso, de forma que se puede aplicar con unas pinzas metálicas. Esto hace que se pueda poner el componente encima de la pasta. Una vez que está encima, se calienta la pasta hasta que se vuelve líquida. Entonces se retira la pistola de aire y se deja enfriar por unos segundos hasta que solidifica.

El uso de la pistola de aire y la pasta para soldar, hace que los resultados obtenidos sean más preciso, además de tener un aspecto más profesional y comercial.

Para la fabricación de los circuitos impresos, se han tenido en cuenta ciertas reglas de diseño a la hora de trabajar con la herramienta Eagle.

4. DISEÑO SOFTWARE

En este capítulo se va a describir el sistema a nivel de software. Para ello se va a realizar una breve introducción a Java y JavaFX, lenguaje con el cual se ha desarrollado la aplicación de visualización de datos y a Python otro lenguaje que se ha utilizado en este proyecto.

En segundo lugar, se describirán todas las capas a nivel software, la primera capa es la capa de obtención de datos con la descripción de todos los programas Python realizados para la medición de sensores.

La segunda capa es la capa de persistencia de datos con la instalación de un servidor MySQL en la Raspberry Pi y un script realizado en Python para el almacenamiento de datos en la misma.

La tercera capa, es la capa dedicada a la publicación de datos con la instalación de un servidor Glassfish en la Raspberry Pi y un proyecto java para la obtención de los registros de la base de datos.

Y la cuarta capa es la capa de visualización de datos con un cliente creado en JavaFX.

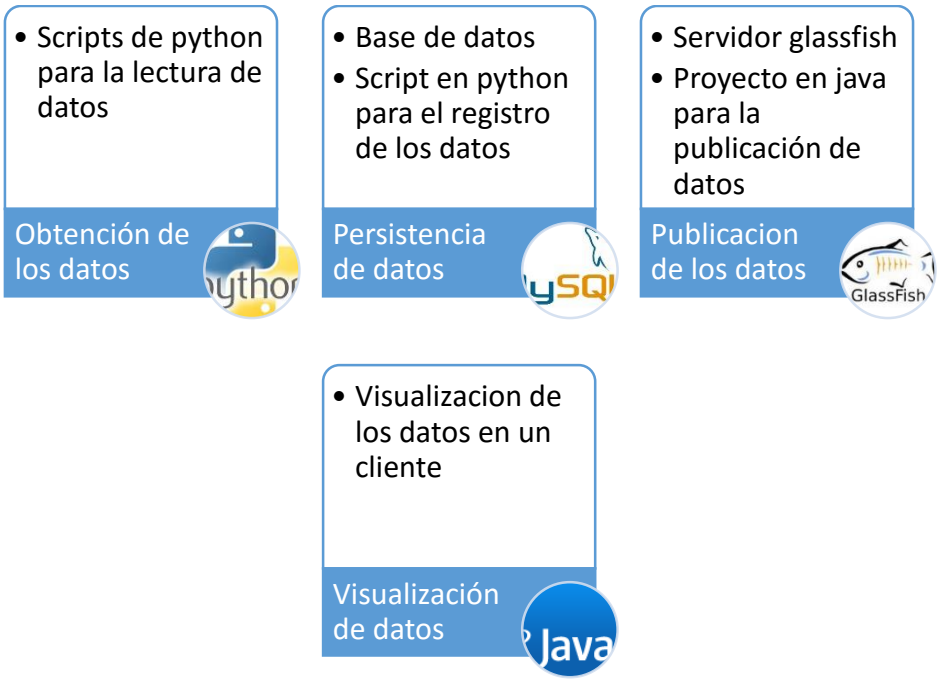

Figura 4.1. Esquema capas de Software

4.1. Lenguaje Java

4.1.1. Breve historia de Java

El lenguaje de programación Java se diseñó para hacer frente al desafío que suponía la heterogeneidad existente en el ambiente de Internet. Siendo un lenguaje que puede extenderse dinámicamente, se puede ejecutar desde cualquier tipo de plataforma, tanto hardware como software.

El lenguaje de programación Java se originó como resultado de la búsqueda para desarrollar software que se pudiera utilizar en una amplia variedad de dispositivos en redes de trabajo. La meta era desarrollar una plataforma, operativa en tiempo real que fuese fiable, y a la vez, portátil y distribuida.

Las computadoras personales han tenido un gran impacto a nivel social, comercial y

tecnológico. Todo ello ha sido posible, gracias al desarrollo de los microprocesadores. Se cree que la siguiente área que éstos van a revolucionar es el área de los dispositivos electrónicos de consumo inteligentes. Con vistas a esto, Sun Microsystems financió un proyecto de investigación, a nivel corporativo interno, que recibió el nombre de “Proyecto Green”. Uno de los resultados de este proyecto, fue la creación del lenguaje Java. Pero el proyecto Green se vio en serios apuros al no crecer, con la rapidez con la que Sun había previsto, el mercado de los dispositivos electrónicos de consumo inteligentes. A la par, la explosión de popularidad de la World Wide Web en 1993, hizo que Sun detectase el potencial de este lenguaje para la creación de páginas de Web con contenido dinámico. Esto hizo que se consolidara el proyecto Green, gracias al gran interés comercial que despertó la World Wide Web. De esta forma, Java se convirtió en un lenguaje eminentemente de uso comercial, en contraste son otros lenguajes académicos como Pascal, y de interés para la comunidad de los negocios; no sólo para el uso local de un grupo reducido como el C o el C++.

El C++ era el lenguaje en auge pues es un lenguaje híbrido que permite programar al estilo del C, con orientación a objetos o con una mezcla de ambos. Pero debido que las dificultades que éste planteaba fueron creciendo se llegó a un punto en el que los problemas que surgían, tan sólo podrían encauzarse a una solución creando una plataforma de lenguaje totalmente nueva. Así, se diseñó una arquitectura de decisiones distinta desde multitud de lenguajes, como: Eiffel, SmallTalk, Objective C, y Cedar/Mesa. El resultado fue una plataforma de lenguaje que, con el tiempo, se ha convertido en seguro, distribuido y con muy diversas aplicaciones y ambientes; desde usuarios de redes de trabajo, hasta la World Wide Web.

4.1.2. Ventajas del lenguaje Java

Aunque Java es un lenguaje potente y con orientación a objetos, su aprendizaje es relativamente rápido especialmente para los programadores que están familiarizados con el C o el C++.

Un programa escrito en Java puede llegar a tener hasta cuatro veces menos código que el mismo escrito en C++.

A la vez que el código es menor, también el tiempo de desarrollo de un programa decrece. Este rápido desarrollo de programas puede llevarnos a ahorrar el doble de tiempo para un programa escrito en Java, respecto al mismo programa escrito en C++. Esto es así porque, además de tener que escribir menos líneas de código, el Java es un lenguaje mucho más simple que el C++.

El lenguaje de programación Java anima a tener buenas prácticas de programación y, por ello, a escribir mejor código. Además, el colector de basura, nos ayuda a evitar fugas de memoria.

La dependencia que tenemos en otros lenguajes con la plataforma, desaparece en el Java gracias a la portabilidad. Podemos contar con una recopilación de manuales de procesos históricos, para Java, y así evitar el uso de librerías escritas en otros lenguajes de programación; no perdiendo, de esta manera, la independencia con las plataformas.

Ya que los programas en Java son compilados en bytecode, tienen la cualidad de ejecutarse consistentemente en cualquier plataforma Java, por ello se pueden ejecutar una vez que ya estén escritos.

4.1.3. Características generales de java

En este capítulo, enumeraremos las principales cualidades de Java, haciendo hincapié en aquellas que nos han resultado especialmente interesantes para la realización del presente Proyecto.

Simple, orientado a objetos y familiar

La principal característica de Java es la simplicidad. Este lenguaje puede ser utilizado sin amplios conocimientos de programación, al tiempo que se ciñe a las actuales prácticas de software. Los conceptos fundamentales de la tecnología Java pueden ser adquiridos con rapidez y los programadores pueden ser realmente productivos desde el comienzo.

Java se engloba en los lenguajes de programación orientados a objetos, resultando una clara y eficiente plataforma, desarrollada en la orientación a objetos. Después de un periodo de gestación de más de treinta años, la tecnología de los objetos ha conseguido ocupar un puesto importante dentro de las principales corrientes de programación.

Robusto y seguro

El lenguaje Java se diseñó para crear un software de alta fiabilidad. Provee de un amplio tiempo de comprobación en la compilación, seguido de un segundo nivel de comprobación cuando se corre el programa. Las características del lenguaje guían al programador a través de unos hábitos de programación fiables.

Arquitectura neutral y portátil

La tecnología Java se diseñó para dar soporte a aplicaciones desplegadas en ambientes heterogéneos de redes de trabajo. En muchos casos, la aplicación ha de poderse ejecutar en multitud de arquitecturas hardware. A través de esta variedad de plataformas hardware, las aplicaciones deben ejecutarse en variedad de sistemas operativos e interoperativos con múltiples interfaces de lenguajes de programación. Para acomodarse a la diversidad de ambientes operativos, el compilador de Java genera bytecodes; una arquitectura neutral diseñada en formato intermedio para transportar el código, eficientemente, a múltiples plataformas de software y hardware. El intérprete de la tecnología Java, resuelve el problema de distribución binaria y el problema de versión. El mismo código en octetos correrá en cualquier otra plataforma.

Alto comportamiento

La plataforma Java consigue un alto funcionamiento adoptando un esquema mediante el cual, el intérprete puede correr a total velocidad sin necesidad de comprobar el entorno runcheck. El recolector de basura automático corre como un hilo con baja prioridad; asegurando de esta manera, que haya una alta probabilidad de que la memoria esté disponible cuando sea requerida. Todo ello desemboca en un mejor funcionamiento. Las aplicaciones que requieren gran cantidad de potencia computacional, pueden ser diseñadas de tal forma que, secciones de computación intensiva se pueden reescribir en código máquina nativo, según puedan ser requeridas o interpretadas por la plataforma Java. En general, los usuarios perciben que las aplicaciones interactivas responden rápidamente, aunque sean interpretadas.

4.1.4. La tecnología de Java

La tecnología Java está compuesta de un lenguaje de programación y una plataforma.

El lenguaje de programación Java es un lenguaje de alto nivel que se caracteriza por ser simple, orientado a objetos, distribuido, interpretado, robusto, seguro, de arquitectura neutral, portátil, de alto comportamiento, con multihilos y dinámico.

Una plataforma es el ambiente de hardware o software en el que un programa corre. La mayoría de plataformas pueden ser descritas como una combinación de sistemas operativos y hardware. La plataforma de Java difiere del resto, en el sentido de que sólo es una plataforma software que corre en la base hardware de otras plataformas.

La plataforma de Java tiene dos componentes: la Java Virtual Machina (Java VM) o máquina virtual de Java y la Java Application Programming Interface (Java API).

La máquina virtual de Java es la base para la plataforma Java y tiene soporte en varias bases de plataformas hardware.

La Java API, es una amplia colección de componentes de software que dan capacidad útil, como interfaces graficas de usuario (GUI). La Java API está agrupada en librerías de clases e interfaces emparentadas, conocidas como paquetes.

El código nativo resulta de la compilación, corriendo el compilador en un hardware específico. Al ser un ambiente independiente de la plataforma, la plataforma Java puede ser un poco más lenta que el código nativo. De todas formas, los buenos compiladores son capaces de actuar cerca del código nativo, en cuanto a velocidad, sin perder su portabilidad.

En la mayoría de los lenguajes de programación, se compila o interpreta un programa para que pueda ser corrido en la propia computadora. El lenguaje de programación Java es inusual en este sentido, ya que un programa es, a la vez, compilado e interpretado. Con el compilador, primero, se traduce el programa a un lenguaje intermedio llamado Java bytecodes; que es un código independiente del intérprete en la plataforma de Java. El intérprete analiza y corre cada instrucción bytecode de Java en la computadora. La figura 4.2 muestra un esquema de esto.

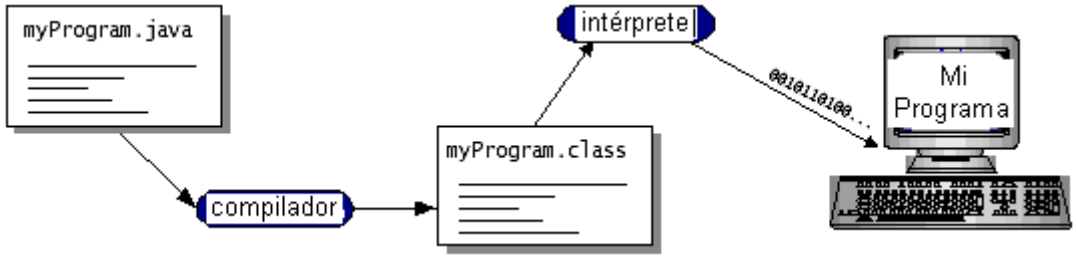

Figura 4.2. Proceso de compilación

Se puede pensar en los bytecodes de Java como en un código de instrucciones para la máquina virtual de Java. Cada intérprete de Java es una implementación de la máquina virtual de Java. Es posible compilar un programa en bytecodes en cualquier plataforma que tenga compilador de Java.

La mayoría de los tipos de programas escritos en lenguaje Java, son applets y aplicaciones de escritorio.

Una aplicación es un programa que corre directamente en la plataforma de Java. Un tipo especial de aplicación son por ejemplo los servlets. Un servlet puede definirse

como un applet que corre en el lado del servidor. Los *servlets* de Java son populares para construir aplicaciones Web interactivas, reemplazando el uso de los CGI scripts. Los servlets son similares a los applets en el sentido de que son extensiones runtime de aplicaciones.

Las implementaciones de las plataformas Java tiene muchas características, destacando la seguridad e internacionalización. La seguridad es tanto a bajo, como a alto nivel e incluye certificados, controles de acceso y manejo público y privado de clases. La internacionalización implica la escritura de programas que pueden ser localizados por usuarios de todo el planeta.

La tecnología Java contiene elementos esenciales como números, cadenas, hilos, objetos y estructuras de datos; pero además otros elementos que la enriquecen considerablemente como el trabajo en redes o los applets. Además de esto, componentes de software como el JavaBeans, que ha sido utilizado para la realización de este proyecto, son capaces de unirse a arquitecturas de componentes ya existentes.

Además, podemos obtener uniformidad de acceso para un amplio rango de bases de datos emparentadas gracias a la conectividad de las bases de datos de Java o JDBC.

4.1.5. La interfaz gráfica JavaFX

JavaFX es una familia de productos y tecnologías de Sun Microsystems, adquirida por Oracle Corporation, para la creación de Rich Internet Applications (RIAs), esto es, aplicaciones web que tienen las características y capacidades de aplicaciones de escritorio, incluyendo aplicaciones multimedia interactivas. Las tecnologías incluidas bajo la denominación JavaFX son JavaFX Script y JavaFX Mobile, aunque hay más productos JavaFX planeados.

Aunque como se ha dicho el objetivo con el que se creó esta tecnología es el desarrollo de las conocidas como RIAs, esta tecnología nos ha permitido la creación de una aplicación de escritorio aprovechando sus numerosas funcionalidades en cuanto a la distribución en pantalla de los distintos elementos, las opciones de animaciones gráficas que el API nos ofrece, y cuantiosas opciones en relación al apartado gráfico.

Cabe hacer mención al API de JavaFx, el cual nos permite integrar totalmente una interfaz de usuario con un código escrito en Java.

Otra de las características de JavaFx, y que ha sido bastante usada en esta aplicación, es la funcionalidad del Data Binding. En los siguientes párrafos se va a detallar un poco más esta funcionalidad.

Esta funcionalidad nos ofrece la posibilidad de relacionar dos variables dentro de nuestro programa. Esto se consigue gracias a la observabilidad de las variables.

Comenzamos con la asignación de unas propiedades observables a las clases de nuestra aplicación. A estas propiedades, se les asignan unos “listeners”. Éstos, no son más que clases con un método que en el momento en el que la propiedad es modificada es ejecutado. La observabilidad de estas propiedades, nos permite añadir relaciones entre clases, de manera que cuando la propiedad se modifica en una de ellas, en la otra se ejecuta una determinada operación.

El Data Binding consiste en relacionar de manera uni o bidireccional dos propiedades observables.

Estructura de una aplicación JavaFX

Para dar unas ideas generales del proceso de creación de una interfaz usando esta tecnología, se van a exponer dos de los procesos que se han usado para el desarrollo de este proyecto, y la estructura que implican los mismos.

En primer lugar, es posible desarrollar toda una interfaz gráfica a través de la creación de fichero xml, que, usando un lenguaje de etiquetas, permiten la definición de todos los elementos de un componente de la interfaz.

En segundo lugar, el API de JavaFx ofrece la posibilidad de crear una interfaz gráfica, totalmente desde cero, a través de código Java y totalmente integrado con el resto de la aplicación. Esto es sumamente importante, por ejemplo, a la hora de crear pequeños diálogos de error, de manera que no es necesario crear siempre un fichero fxml, ahorrando en número de ficheros creados y mejorando la comprensión del programa.

Por último, decir que la estructura básica de una aplicación en JavaFx, comienza por la definición de un "stage" el cual puede definirse como una ventana, capaz de mostrar "scene" o vistas, una o varias diferentes a lo largo del uso de la aplicación. Ya dentro de cada "scene", se introducen los "nodos" o elementos gráficos de la aplicación, procurando siempre tener un "nodo padre" que recoja todos los elementos que se pretenden introducir en la pantalla.

Por último, decir que cada vez se está extendiendo más el uso de esta tecnología en el ámbito de las aplicaciones de escritorio, y en los últimos meses, la posibilidad de portación a las plataformas Android e IOs, ha supuesto la apertura de un nuevo horizonte de posibilidades para el desarrollo de esta tecnología.

4.2. Lenguaje Python

Python es un lenguaje de programación interpretado, funcional, orientado a objetos e interactivo. Su sintaxis favorece el código legible, posee licencia de código libre.

4.2.1. Breve historia de Python

El lenguaje fue creado a finales de los años 80 como sucesor del lenguaje de programación ABC, capaz de manejar excepciones. El nombre proviene de la afición de su creador por los humoristas británicos Monty Python.

Guido Van Rossum es el principal autor de Python que creó este lenguaje en un instituto nacional de investigación informática en Amsterdam. Desde entonces, Python se ha convertido en un lenguaje muy popular entre los desarrolladores, a quienes atrae su sintaxis limpia y su fama de productividad. (Python, 2016)

4.2.2. Ventajas del lenguaje Python

Desde una primera perspectiva se puede decir que Python es un lenguaje de 'scripting' a alto nivel. Pero desde otra se puede decir que es un lenguaje de alto nivel que pasa a ser implementado de una manera que hace hincapié en la interactividad.

Python comparte algunas características con los lenguajes de script, pero también comparte algunas con los lenguajes más tradicionales. Python es un lenguaje que puede ser aplicado a muchos tipos diferentes de problemas.

El lenguaje viene con una gran biblioteca estándar que cubre áreas tales como el procesamiento de cadenas (expresiones regulares, Unicode, cálculo de diferencias

entre archivos), protocolos de internet (HTTP, FTP, SMTP, XML-RPC, POP, IMAP), ingeniería de software ('testing' unitario, 'logging', 'profiling' (optimización de rendimiento), 'parsing' del código Python) y interfaces de sistema operativo (llamadas al sistema, sistemas de ficheros, sockets TCP/IP). Además, una amplia variedad de extensiones de terceros está disponibles.

Python es un lenguaje para programadores principiantes, todavía es común iniciar a los estudiantes con un lenguaje de procedimientos y tipos estáticos, tales como Pascal, C, o C++ y Java. Los estudiantes pueden ser mejor enseñados con un lenguaje como Python como su primer idioma.

Python tiene una sintaxis muy simple y consistente con una gran biblioteca estándar, y lo más importante, el uso de Python en un curso de programación para principiantes permite que los estudiantes se concentren más en habilidades de programación más importantes como descomposición de un problema y el tipo de datos de diseño.

Con Python los estudiantes se pueden introducir de forma rápida en los conceptos básicos, tales como bucles y procedimientos, es probable que puedan trabajar con objetos definidos por el usuario en su primer curso.

Para un estudiante que nunca ha programado antes, le resulta más difícil empezar con un lenguaje estático, se añade una complejidad adicional que el estudiante debe dominar y ralentiza el curso.

El uso de la biblioteca estándar también enseña a los estudiantes sobre la reutilización de código o los módulos de tercero también son útiles para extender el conocimiento de los estudiantes.

El intérprete interactivo ayuda a los programadores a testear características del lenguaje mientras están programando. Ellos se apoyan en la ventana del interprete mientras en otra están desarrollando. (Python, 2016)

La Raspberry Pi apuesta por Python como principal lenguaje de programación para desarrollar aplicaciones para este ordenador de placa reducida, viene preinstalado y con mucha documentación al respecto.

4.3. Capa de obtención de datos

Esta capa define la manera de obtener los datos de los sensores y tenerlos disponibles para su posterior almacenamiento en una base de datos. A continuación, se va a describir el software desarrollado para la Raspberry Pi en cuanto a la lectura de los sensores, es decir los scripts programados en Python para leer estos.

En la Raspberry Pi se han desarrollado varias librerías para cada sensor a medir, así como un proceso para capturar los datos medidos en una base de datos (explicado en el apartado 4.4), todo ello desarrollado en Python.

Lo primero que se realiza en la Raspberry Pi es leer las entradas analógicas de los conversores analógico a digitales mediante una librería para el ADS1115, en la cual se tiene métodos para obtener valores de las entradas en modo simple o en modo diferencial.

La estructura de la librería es la siguiente:

1- Adafruit_I2C.py: esta librería que importa a su vez 'smbus' (librería de Python para poder comunicarse con un dispositivo I2C) permite leer una serie de bytes, escribir valores en un registro del dispositivo I2C etc.

2- Adafruit_ADS1x15.py: librería que importa a ‘1’, en esta librería se definen todos los registros y configuraciones del ADS1115. Con esta se puede leer las entradas analógicas, ya bien sea de manera simple (readADCSingleEnded) o diferencial (readADCDifferential), luego se explicarán más a fondo. De este tipo de librería existen dos, la misma y otra con el sufijo 0x49 que lo que indica es que tiene esa dirección el dispositivo I2C (conversor analógico digital secundario para otro acelerómetro)

Se puede implementar un simple ejemplo para medir las tres señales analógicas de un acelerómetro de la siguiente forma:

- **Importar** librerías necesarias
- Seleccionar el **chip**
- Seleccionar la **ganancia**
- Seleccionar las **muestras** por segundo
- **Instanciar** la clase
- **Leer** las tres señales en modo simple
- **Imprimir** por pantalla los valores

Lo segundo que se realiza en el proyecto es una librería para leer los sensores de fuerza, pero con la versión 1 de la placa de adquisición de datos, lo que se quiere decir con esto es que se va a usar la tarjeta HX711 comprada en ‘dfRobot.com’.

Esta librería tiene un único fichero que ‘.py’, existen dos versiones:

- 1ª versión, solo lee valores positivos en la dirección de la fuerza, esta funcionalidad es como la que ofrece el fabricante de la placa para leer este tipo de sensores con Arduino, lo único que se ha portado a Raspberry Pi.
- 2ª versión, se añade una nueva funcionalidad que es medir la fuerza en ambos sentidos.

En esta librería se tiene una clase llamada ‘HX711’, en la cual se tienen métodos como:

- Init: con este se construye la clase en un programa principal, a este método se le pasan los pines de datos y reloj que se van a usar con la placa, en el caso de ejemplo se usan los pines 17 (como línea de datos) y 27 (como línea de reloj), se puede ver en el main (el main en las librerías se usa para ejecutarlas y realizar un test de tu librería) de esta librería.
- Read: con este método se mide el valor devuelto del convertidor HX711, se miden 24 bits del más significativo al menos significativo, esto se convierte a entero para ser devuelto.
- Read_average: se toma una media de 25 valores por defecto, este número se puede pasar por parámetro y hacer medias de más muestras. Este método también descarta valores muy dispersos, puesto que debido a ruido y otros inconvenientes, a veces se toman muestras erróneas.
- Get_grams: se obtiene el valor en gramos, tanto negativos como positivos ya que se aplica el complemento a dos.
- Existen otras dos funciones que fijan los valores de calibración tanto el offset como la escala.

También se ha realizado una librería para medir el desplazamiento con la Raspberry Pi, procedente de un calibre digital. Como se vio en el subapartado hardware

correspondiente (3.4.4), este calibre posee dos señales de comunicación y dos de alimentación.

Para este sensor se ha creado una librería con una clase llamada “DISPLACEMENT”, la cual tiene métodos como:

- Init, constructor de la clase al cual le tienes que pasar los 3 pines a utilizar para medir este sensor además de los dos de alimentación, se usan 3 pines y no dos como se comentaba anteriormente ya que el conversor de nivel necesita un pin de habilitación. Los pines son un pin de datos, un pin de reloj y un pin de habilitación del conversor de nivel (1.5V a 3.3V)
- Read: método que lee el sensor de desplazamiento, este método posee timeouts por si el sensor no responde en 300ms es que no está conectado y no devuelva una señal incorrecta. En la librería esta explicado los pasos a seguir para medir este sensor como se explicó en el apartado de hardware correspondiente hay que recibir una señal a alto de 500µs en la línea del reloj para poder medir el sensor, a partir de ahí tomar 24 bits descartando los 3 primeros, en el orden de LSB to MSB. Saber también que el bit 21 es el signo.

Este sensor tiene una limitación y es que se tarda en tomar una medida de él 300ms por lo que este sensor es usado para medidas estáticas, y cuando se quieran realizar problemas dinámicos se deberá deshabilitar este sensor (como se verá más adelante se integra una funcionalidad en la aplicación para deshabilitar este sensor).

También se ha creado una librería para el sensor de temperatura y humedad (HDC1008), con la cual se puede medir estas dos variables a partir de la comunicación I2C. El fichero se llama ‘hdc1008.py’. Tiene métodos como:

1- Init, constructor de la clase.
2- readTemperature, para medir la temperatura, devuelve un float.
3- readHumidity, para medir humedad, devuelve un float también.

4.4. Capa de persistencia de datos

En esta capa lo que se va a definir es el servidor instalado para el manejo de una base de datos, así como el programa creado en Python para la captura de los datos y el registro en la tabla correspondiente a la base de datos. Lo primero que se va a explicar es como se ha instalado un servidor para gestionar una base de datos. La primera opción que se manejo fue instalar Java DB (derby) pero es más complicado y existe menos documentación para manejar esta base de datos con Python por lo que se optó por una opción más fácil como es instalar MySQL.

Figura 4.3. MySQL + Python

Para tener bien configurado el servidor MySQL en la Raspberry Pi y poder interactuar con él en este proyecto es necesario seguir los siguientes pasos:

1- Lo primero será crear una base de datos, que se debe hacer entrando en la linea de comandos de MySQL con el siguiente comando

   Mysql –u root –p

   Enter password:

   mysql> CREATE DATABASE temps

   mysql> USE temps

   Luego se pedirá la contraseña y a continuación se introducen los comandos para crear la base de datos y posicionarse en ella para usarla.

2- Lo siguiente es dar permisos a los usuarios para acceder a esta base de datos, para ello se ejecuta el siguiente comando:

   GRANT ALL ON temps.* TO root@'192.168.%.%' IDENTIFIED BY 'raspi';

   'raspi' es la contraseña y '192.168.%.%' es la IP desde donde se da permiso para acceder a la base de datos, en este caso se dan permisos para acceder a la base de datos desde todos los ordenadores locales de la red.

3- En el archivo '/etc/mysql/my.conf' hay que comentar la bind address para poder acceder desde remoto y local.
4- Lógicamente esto más adelante habrá que regularlo por temas de seguridad para que solo se pueda acceder a la base de datos desde 'localhost'.

Para crear tablas, con sus correspondientes filas y columnas se va a usar Netbeans, IDE de programación en el cual se pueden gestionar bases de datos. Este IDE es el que se utiliza también para programar la aplicación JavaFX para la visualización de los datos de los sensores.

Las tablas que se van a tener en nuestra aplicación son las siguientes:

1- Tabla para el registro de los datos obtenidos de los sensores, su nombre es 'measurements'.
2- Tabla para el registro de usuarios, su nombre es 'credentials'.

El contenido de la primera tabla es el siguiente:

Nombre columna	**Tipo de dato**	**Descripción**
var_ax	FLOAT	Medida de la aceleración en el eje X
var_ay	FLOAT	Medida de la aceleración en el eje Y
var_az	FLOAT	Medida de la aceleración en el eje Z
var_fx	FLOAT	Medida de la fuerza en el eje X
var_fy	FLOAT	Medida de la fuerza en el eje Y
var_dy	FLOAT	Medida del desplazamiento en el eje Y
var_ta	FLOAT	Medida de la temperatura

var_hr	FLOAT	Medida de la humedad relativa
date	BIGINT	Fecha y hora en milisegundos desde 1970, esta fila es la 'primary key' de la tabla

Tabla 4.1. Descripción de la tabla de medidas alojada en la base de datos

La descripción del contenido de la tabla de credenciales de usuarios alojada la base de datos es la siguiente:

Nombre columna	Tipo de dato	Descripción
user	VARCHAR	El nombre del usuario, esta es la 'primary key' de la tabla
pass	LONGVARBINARY	Contraseña del usuario, esta encriptada con AES

Tabla 4.2. Descripción de la tabla de usuarios alojada en la base de datos

Se puede observar que el campo 'pass' va encriptado con AES con una contraseña especifica. Esto es debido a seguridad, si alguien accedería a la base de datos las contraseñas no estarían en texto plano. Debido a esta seguridad en la aplicación JavaFX para la visualización de datos cuando se entre se deben pedir a esta tabla los credentials para un usuario y esta contraseña hay que desencriptarla con la misma 'parphase'.

El segundo elemento que compone esta capa es un proceso en Python para poder capturar muestras y guardarlas en una base de datos. El código esta comentado en ingles en el correspondiente archivo 'capturar.py'.

La librería que se usa para integrar MySQL con Python es 'MySQLdb'. Con esta librería se pueden realizar querys a la base de datos.

El seudocódigo para este script realizado en Python es el siguiente:

- **Importar** librerías
- Declarar variables **globales** (como la conexión a la base de datos)
- Clase llamada **App()**:
  - Método init (constructor de la clase, al cual le pasas como parámetro una flag para ver si se quiere habilitar el desplazamiento o no). En este se configurar todos los paths como el de el pid del proceso, etc. También se configuran las muestras a tomar en un minuto
  - Método handler: es por seguridad si dejas la aplicación leyendo datos más de dos horas que la apague para que no se llene la base de datos
  - Método run, bucle del proceso en el cual empieza a tomar muestras y grabarlas en la base de datos.
  - Método stop, se le llama cuando se desea para el proceso. Aquí viene una parte de los procesos en Python que se ha modificado para poder pararles con un usuario externo. Creo el proceso con DAEMON pero hay que modificar el procedimiento _stop en '/usr/lib/pymodules/python2.7/daemon/runner.py', ese procedimiento debe llamar a mi método creado stop así se cierra la conexión a la base de datos.

- Por ultimo existe un **main**, donde se llama a la estructura de DAEMON para crear el proceso:
  - Se crea el objeto app, pasando como argumento si se desea medir desplazamiento o no.
  - Luego se crea el logger para guardar cualquier tipo de error del proceso.
  - Y por último se llama a la clase runner con nuestra app para que se ejecute como un proceso con su pid y así tener controlado este proceso.

Por último, decir que se ha creado un script en lenguaje Shell para la ejecución de este proceso con tan solo ejecutar el comando 'capturar start' y parar el proceso con tan solo ejecutar el comando 'capturar stop'. Este archivo se llama 'capturar' y se encuentra en la carpeta /usr/bin de la Raspberry Pi para poderlo ejecutar desde cualquier ruta de la Raspberry Pi. A este archivo hay que darle permisos para todos los usuarios con el comando 'chmod'.

4.5. Capa de publicación de datos

En esta capa se define como se publican los registros de la base de datos, lo que se quiere decir es que hay que tener acceso a esos datos de alguna forma y la forma seleccionada para este proyecto es instalar un servidor Glassfish y continuación crear en java servicios web para hacer peticiones de estos registros.

4.5.1. Instalacion de un servidor Glassfish

Lo primero, es instalar el servidor Glassfish en la Raspberry Pi, puesto que en este servidor se van a desplegar servicios web, que son los que van interactuar con la base de datos y la Raspberry Pi.

Se debe comprobar la versión de java instalada en la Raspberry Pi con el comando "java –version", seguramente este instalada la versión 1.7, entonces se debe instalar la versión 1.8 de la siguiente manera:

1- Descargar la versión 1.8 de java de la siguiente url: http://www.oracle.com/technetwork/java/embedded/embedded-se/downloads/index.html
2- Descomprimir en /usr/bin
3- Seleccionas la versión 1.8 con el comando "sudo update-alternatives --config java"

La instalación de Glassfish en la Raspberry Pi es muy sencilla y se van a seguir los siguientes pasos:

1- Descargar de la página web 'https://glassfish.java.net/download.html' la versión Full Platform en zip.
2- Transferir el zip por SFTP a través del programa Filezilla o uno similar a la carpteta /home/pi/Downloads.
3- Crear la carpeta en /app/glassfish.
4- Dar permisos "https://glassfish.java.net/download.html" en la cual se descomprime el zip descargado.
5- Cambiar de directorio. " cd /app/glassfish/glassfish4/bin".
6- Ejecutar "./asadmin change-admin-password --user admin" para asignar la contraseña al usuario admin. La password por defecto es en blanco.
7- Ejecuta "./asadmin start-domain" para arrancar el servidor.
8- Ejecuta " ./asadmin enable-secure-admin" para habilitar la gestión remota

usando la consola web.

9- Ejecuta “./asadmin stop-domain” para parar el servidor.
10- Añade en el archivo “/etc/rc.local” la siguiente linea:

    /app/glassfish/glassfish4/bin/asadmin start-domain

    Para que arranque el servidor al arrancar la Raspberry Pi

11- Resetear la Raspberry Pi
12- Ir desde tu ordenador a la URL “192.168.x.x:4848”, en las ‘x’ poner tu dirección ip local de la Raspberry Pi
13- Introducir tus credenciales “admin” y “raspi” y listo.

Con estos pasos ya se tiene instalado un servidor web en la Raspberry Pi, en el cual se pueden desplegar paquetes war (Web Application Archive).

4.5.2. Servicios web

Ahora se necesita tener servicios web corriendo en nuestra Raspberry Pi, los cuales van a escuchar peticiones desde un cliente para devolver registros de la base de datos.

A continuación, se va a explicar el proyecto creado en Java, el cual contiene los servicios web que van a devolver registros al cliente para poder visualizarlos.

El proyecto tiene el nombre de DataServerVal y se compone de los siguientes paquetes:

- ‘com.conversion’, este paquete contiene una clase llamada ‘MeasuementsConversor’ que es la encargada de pasar los valores de tensión si es que se tienen en tensión (por ejemplo la fuerza es en gramos) a su correspondiente unidad. En definitiva, aquí se calibran las medidas guardadas en la base de datos a la hora de enviarlas al cliente, no se calibran en el propio script de Python debido a que sería sobrecargar ese proceso.

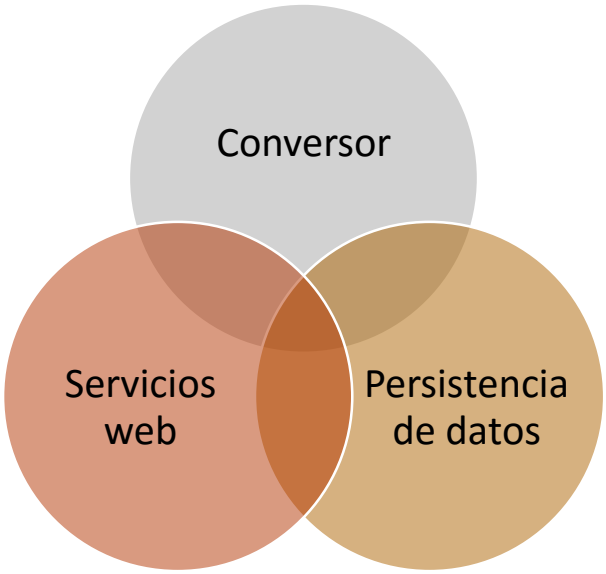

Figura 4.4. Resumen paquetes en el proyecto de servidor Netbeans

- ‘com.dataserver’, en este paquete se encuentra la clase principal llamada ‘DataServer.java’, en esta clase se implementan varios servicios de consulta a

los registros de la base de datos. Esta clase constituye la interfase entre el cliente y nuestra base de datos. También en este paquete está la clase que contiene las medidas ya calibradas, esta se llama 'MeasurementsCalibrated.java'.

- 'com.persistence', este contiene clases java que representan las entidades de nuestra base de datos, haciendo referencia con entidades a la estructura de los registros de nuestra base de datos (estos es estructura de las tablas: columnas, nombre, tipo de dato, etc.).

4.6. Capa de visualización de los datos

Esta capa es la encargada de mostrar los datos al usuario y también permite la introducción de parámetros para modificar esta visualización, etc.

4.6.1. Introducción a la capa de visualización

En esta capa se han creado dos proyectos separados uno que es el preloader de carga para que un determinado usuario inicie la aplicación con su contraseña y elija a que servidor quiere conectarse y una aplicación UI para la visualización de datos e interactuación con los sensores.

Los proyectos han sido creados con JavaFX y se pueden distinguir 3 grandes elementos a alto nivel:

1- Preloader, carga de una pantalla de 'login'
2- Comunicaciones con el servidor de datos, este pertenece a el proyecto de la aplicación UI
3- Presentación de los datos, también pertenece a la aplicación UI.

Figura 4.5. elementos principales de la capa de visualización de datos.

Lo primero que se crea es un preloader (pantalla cargada antes de iniciar la aplicación), este es un inicio de sesión para los usuarios. Solo contiene dos paquetes clave además de las comunicaciones con el servidor:

1- Paquete de imágenes, aquí se guardan las imágenes usadas para el estilo de la pantalla de preloader
2- Paquete de UI, en este paquete se encuentra la clase principal del preloader, que es la encargada de presentar la vista al usuario de la pantalla de 'login'.
3- Referencia al WDSL publicado en la dirección local de la Raspberry Pi, con este archivo se pueden llamar a los servicios publicados en esa dirección y pedir en este caso la tabla de usuarios y contraseñas.

En este preloader están definidos todas las direcciones de las Raspberry Pis conectadas a nuestra red, con lo que se puede elegir a que Raspberry Pi conectarnos junto con el usuario y contraseña correspondientes.

En el preloader es donde se debe tener la clave AES con la que desencriptar la contraseña del usuario correspondiente que quiera entrar en la aplicación.

Este elemento se debe comunicar con nuestra UI principal para poder transmitirla en que servidor de datos se encuentra el usuario conectado, además de si ese usuario es administrador o no, para ofrecerle funcionalidades extra.

El siguiente gran elemento, comunicaciones con el servidor, que está alojado en el proyecto principal de presentación de datos, se compone de un paquete llamado 'com.server' el cual contiene dos clases:

- 'DataManager.java', la cual se encarga de manejar una lista observable, que a su vez es la que tiene un 'listener' que se encarga de modificar la gráfica de los registros si esta cambia.
- 'Server.java', clase que se construye con una URL, esta apunta al WSDL alojado en la Raspberry Pi de la cual se desean obtener los registros. Esta clase maneja todas las llamadas a los servicios web publicados en la Raspberry Pi.

Y por último se tiene la UI, esta se divide en cinco paquetes:

- 'ui', paquete en el cual se tiene la clase principal que carga un FXML (modelo de datos), que cargará la vista al usuario, también incluye un controlador que es el que maneja la vista.
- 'ui.button', paquete que contiene una clase que extiende a la clase 'Button', esta clase se ha creado en especial para este proyecto debido a que se necesita que cuando se presione este tipo de botones se cargue una pantalla u otra en función de los sensores que se quieran visualizar.
- 'ui.dialogos.customizequery', este contiene una clase con su correspondiente FXML (modelo de datos para visualizar) que representa un diálogo de la aplicación para pedir datos de unas determinadas fechas y horas.
- 'ui.images', contiene las imágenes que se van a cargar en la UI.
- 'ui.sensors', esta es una clase enumerable creada para este proyecto que define los sensores conectados a la Raspberry Pi. Si se quieren añadir nuevos sensores hay que añadirlos a esta clase como un nuevo enumerable.

Dejar claro que existen dos listas observables en el controlador de la aplicación, las cuales tienen un 'listener'', que es el mismo, con el que si alguna de estas dos listas se modifican la gráfica se modifica.

- Lista observable para los sensores, si presionamos otro botón la gráfica

cambiará de sensores y en ella se representan los nuevos sensores.
- Lista observable de datos, si los datos cambian (se ha hecho una nueva petición al servidor) esta lista cambia por lo que la gráfica cambia.

4.6.2. Descripción de la aplicación

En este apartado se explicará el funcionamiento de la interfaz gráfica de usuario para la visualización de datos, detallando las opciones que se ofrecen en la misma. Mediante la interfaz se puede acceder a la Raspberry Pi requerida, una vez se está dentro de la aplicación se puede seleccionar los sensores deseados para representarlos en una gráfica.

Se tienen dos funcionalidades claras para la representación de los datos, o en tiempo real o representar datos en un rango temporal seleccionado. En tiempo real los datos se muestran cada segundo, con lo que sería útil para problemas estáticos, o simplemente para comprobar el correcto funcionamiento de alguno de ellos.

Pero también existe la opción de mostrar las ultimas 'x' muestras recogidas con lo que se pueden representar problemas dinámicos de aceleraciones en una maqueta estructural.

También destacar que se puede arrancar el proceso de capturar sin visualizar datos y luego visualizar un rango temporal determinado. Con esto se quiere decir que se puede arrancar la aplicación en tiempo real y más adelante visualizar datos en un segundo, por ejemplo. Este método es más lento que guardar datos en el servidor de manera normal y luego visualizarlos.

Cuando el usuario ejecuta el programa lo primero que aparecerá será la pantalla inicial ('preloader') con la que el usuario debe elegir el servidor al que conectarse e introducir el usuario y contraseña.

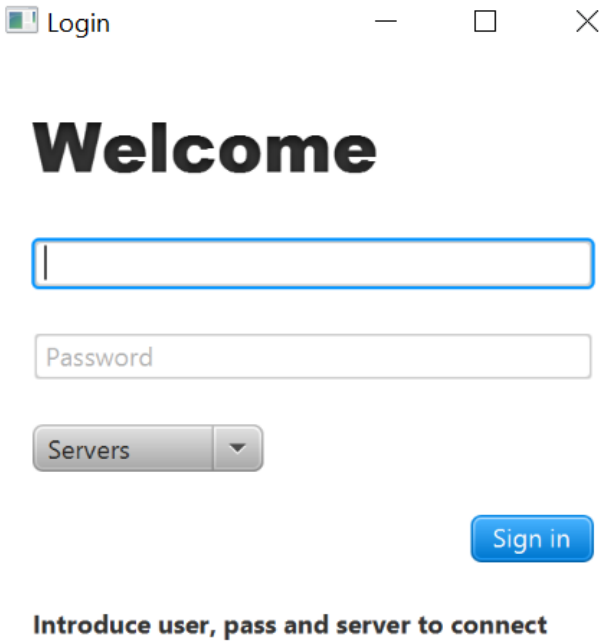

Figura 4.6. Pantalla de 'login' de la aplicación de visualización de datos

Si se introduce unas credenciales erróneas, se le comunicará al usuario mediante la etiqueta de abajo, al igual que si se introduce un servidor que no está disponible en la red se le informa mediante la etiqueta.

Sin embargo, si las credenciales han sido correctas la aplicación nos lanzará la pantalla principal de la aplicación que es la que se muestra en la figura 4.7.

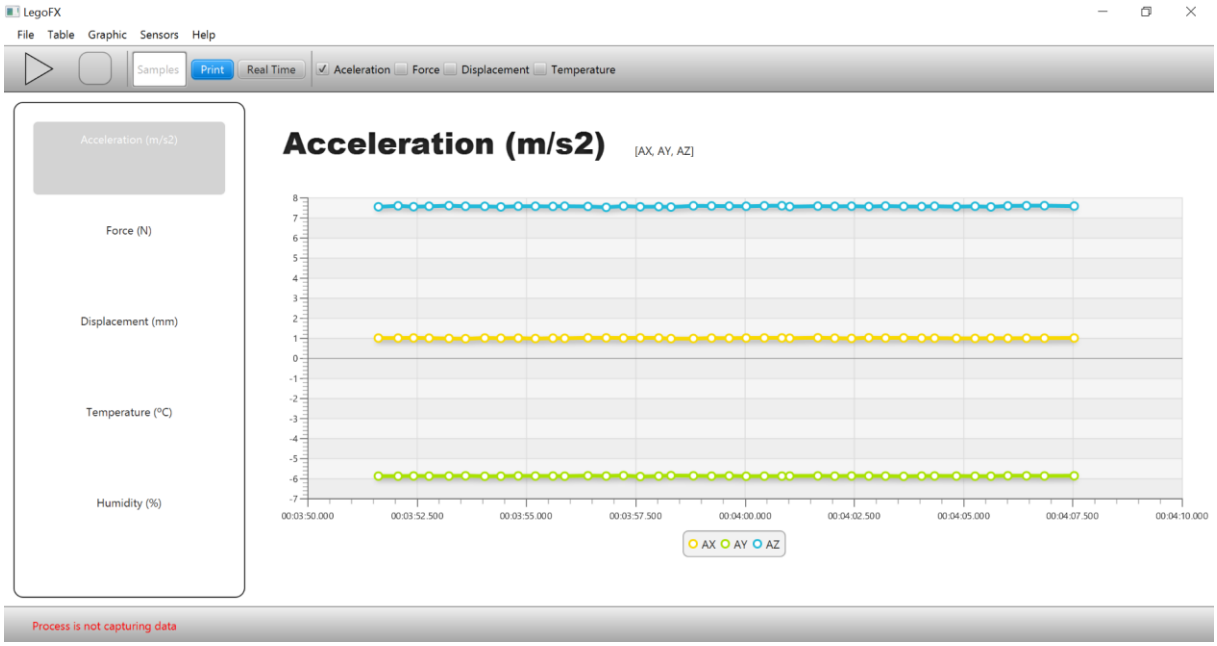

Figura 4.7. Pantalla principal de la aplicación

Se puede ver como en la barra de herramientas superior se tienen los botones para capturar datos (Botón de 'play'), para detener la captura de datos (Botón de 'stop'), un cuadro de entrada para elegir las muestras que representar en la gráfica si presionamos el botón 'print'.

También se tiene un botón de 'Real time' para arrancar la captura de datos en modo 'Tiempo real' y por último la selección de sensores con los que iniciar una captura de datos, separando así por sensores, hay que tener en cuenta que con esta selección se inicia la captura, pero luego se puede cambiar de visualización seleccionando el sensor en el panel izquierdo.

En la barra lateral izquierda se pueden ver los botones de selección de sensores, con lo que seleccionaremos los sensores a representar en la gráfica. Y en el cuerpo principal de la aplicación se encuentra la gráfica en la que se verán los datos a representar.

En la parte inferior de la aplicación, se puede observar una etiqueta que informa de si el proceso de captura de datos está activo o no.

Existe una opción de representar en la gráfica datos en un rango temporal determinado, esta opción se escoge en File – Customize query. Con esta opción se realiza una query a la base de datos customizada de manera que se representaran datos en un rango temporal determinado. En la figura 4.8 se pueden ver las opciones a introducir.

También existe opción de calibrar los acelerómetros de una manera sencilla, ya que son de una tecnología MEMs y permiten medir la gravedad, con lo que es muy sencillo

calibrarles con una pantalla a nivel software siguiendo unos pasos.

Figura 4.8. Dialogo para realizar una query customizada.

Por último, destacar una serie de opciones que se tienen en el menú 'Table' para descargar los datos en formato CSV y otra opción para borrar la tabla de registros, por supuesto esta opción solo estará permitida si el usuario es administrador. También existe la opción de añadir usuarios al equipo de adquisición de datos.

5. RESULTADOS

En este capítulo se pretende exponer los resultados obtenidos con el sistema desarrollado, la comparación se va a realizar frente a un equipo de alto coste en el laboratorio de la escuela.

Este capítulo se va a dividir en apartados según el análisis realizado a cada sensor.

5.1. Análisis del acelerómetro

En primer lugar, se ha hecho un análisis al acelerómetro, este análisis se ha realizado en dos grandes partes, una primera que ha sido un análisis dinámico de aceleración y una segunda un análisis estático de aceleración.

En cuanto al primer análisis, se ha colocado el acelerómetro con una pinza en una maqueta de estructura metálica del laboratorio de la escuela, como se ve en la Figura 5.1.

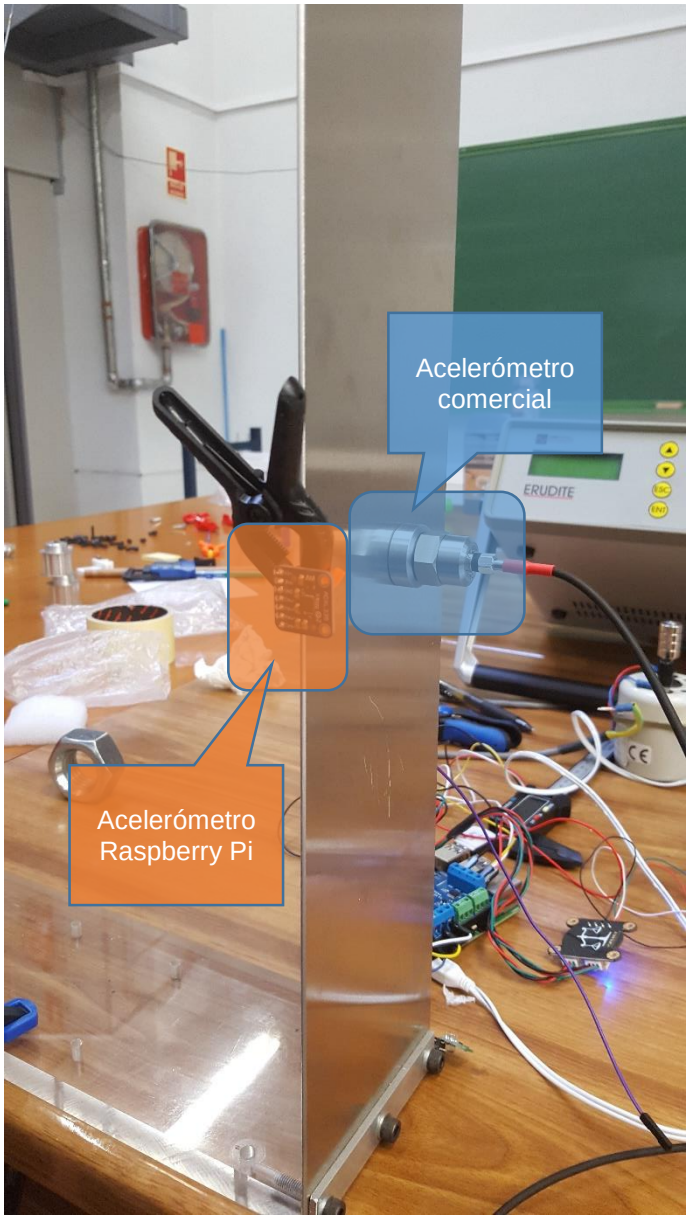

Figura 5.1. Acelerómetro en maqueta estructural.

Y las ondas obtenidas en el ‘eje Y’ son las siguientes, mostradas en la Figura 5.2

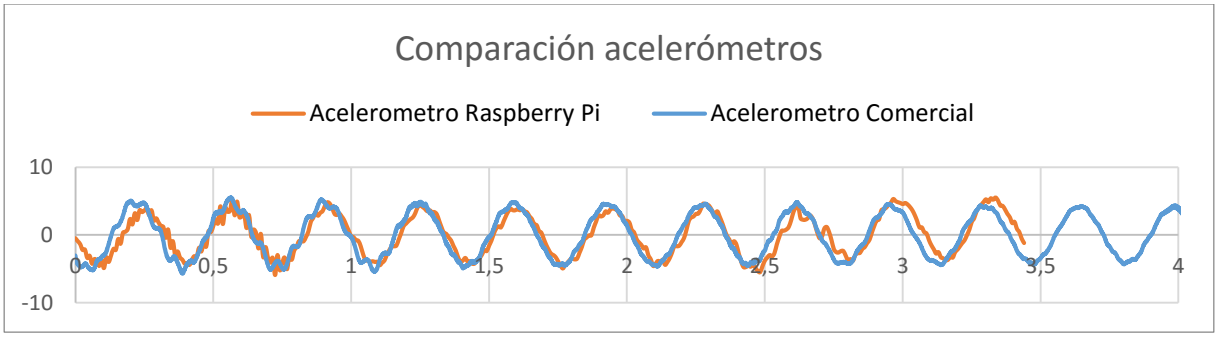

Figura 5.2. Gráficas ambos acelerómetros.

Como se puede ver en la anterior grafica las ondas son muy parecidas y se puede dar por bueno el resultado, viendo que los máximos y mínimos dan en torno a ±5,5 m/s2.

En cuanto al análisis estático de las aceleraciones, los acelerómetros son muy fáciles de calibrar puesto que se coloca el acelerómetro en uno de sus ejes apuntando hacia el suelo y en este caso debe dar 1g = 9,8m/s2.

A continuación, se coloca en el mismo eje sin aceleración estática y debe dar 0 m/s2, y por último se coloca el acelerómetro apuntando el eje medido hacia arriba y el acelerómetro debe marcar -9,8m/s2.

Esto es una ventaja, poder calibrar los acelerómetros de esta manera, debido a que son de tecnología MEMs miden gravedad.

El equipo de medición comercial que posee el laboratorio de la escuela es el que se muestra en la Figura 5.3, el modelo es:

- SIRIUS-HD 16xSTGS
- 16 entradas analógicas con puerto serie.

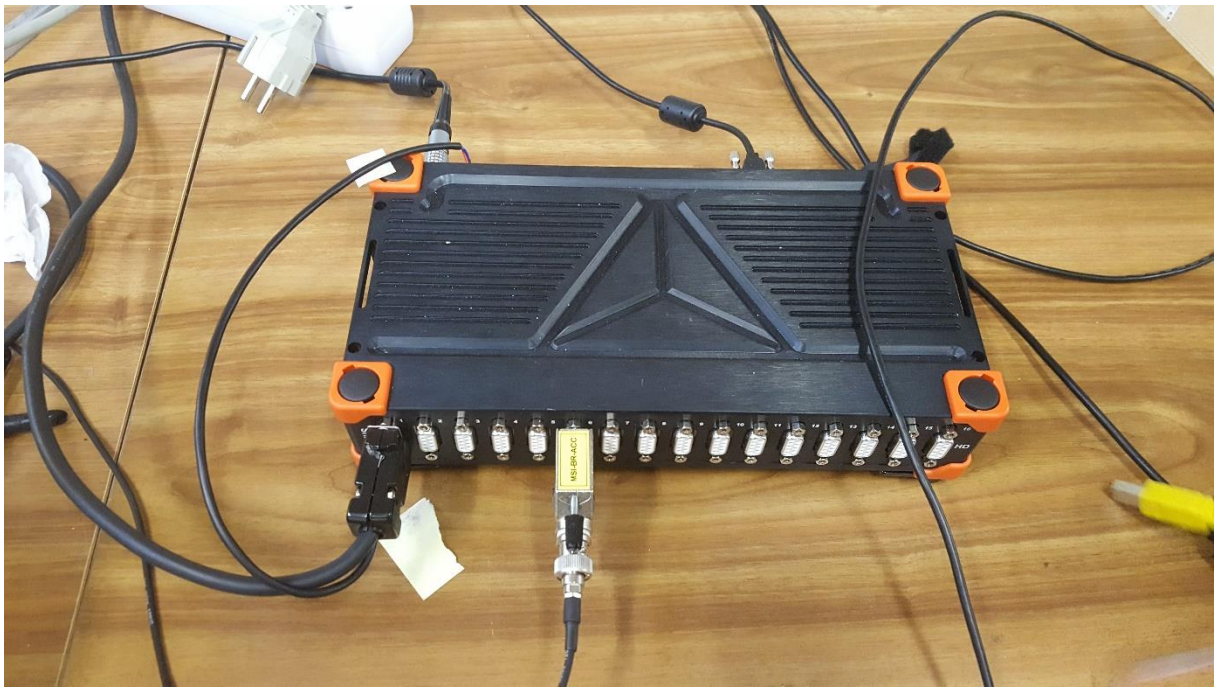

Figura 5.3. Equipo de medición comercial de la escuela.

5.2. Análisis del sensor de fuerza

En este apartado se va a seguir la misma dinámica que para realizar las pruebas del acelerómetro, primero se realizará una prueba dinámica del sensor de fuerza, aplicándole fuerza en ambos sentidos y por último se le realizará una prueba con carga estática en ambos sentidos.

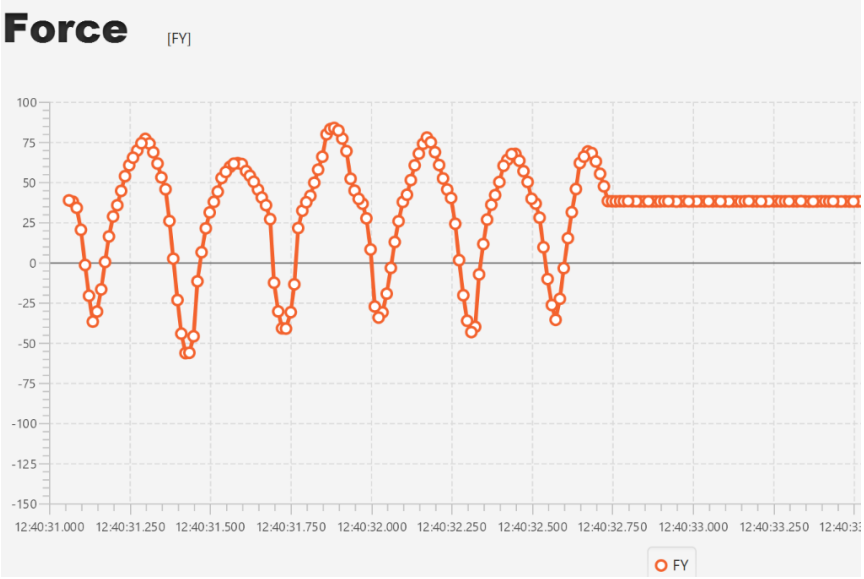

Figura 5.4. Medida dinámica del sensor de fuerza.

En esta prueba lo que se pretende es analizar es si el sensor responde ante cambios de fuerza en ambas direcciones sin necesidad de obtener valores perfectamente calibrados para el sensor.

Para analizar el correcto funcionamiento en cuanto a una calibración exacta lo que se hace es aplicar carga estática en un sentido, luego carga nula y continuación la misma carga anterior, pero en el sentido inverso. Esto comentado se puede apreciar en la Figura 5.4.

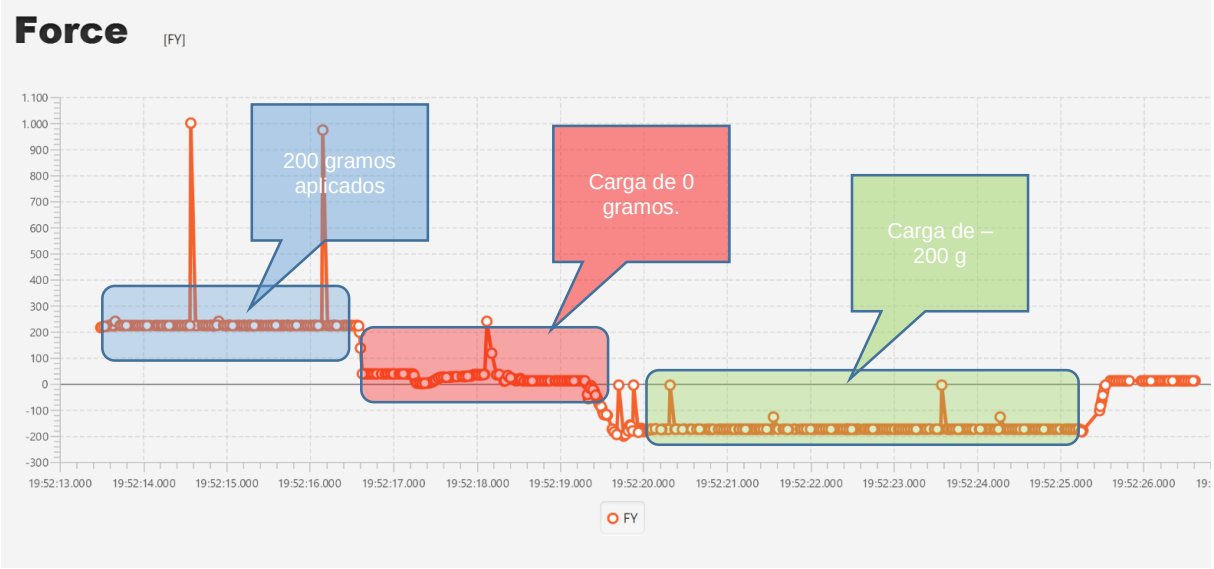

Figura 5.5. Fuerza estática aplicada en ambos sentidos

Esta carga estática se ha aplicado con una tuerca del laboratorio, de la cual se conocía

el peso, con lo que se puede obtener la fuerza, en la figura anterior se pueden observar los 3 intervalos aplicados al sensor.

5.3. Análisis de los sensores de desplazamiento y temperatura

Este análisis ha sido muy breve ya que estos sensores son para problemas estáticos, y solo se han hecho pruebas de este tipo.

En cuanto al sensor de desplazamiento, comentar que tiene un tiempo de respuesta de unos 100ms con lo que se pueden obtener 10 medidas por segundo ya a la hora de analizar datos en un segundo se ve una gráfica discreta como se ve en la Figura 5.6.

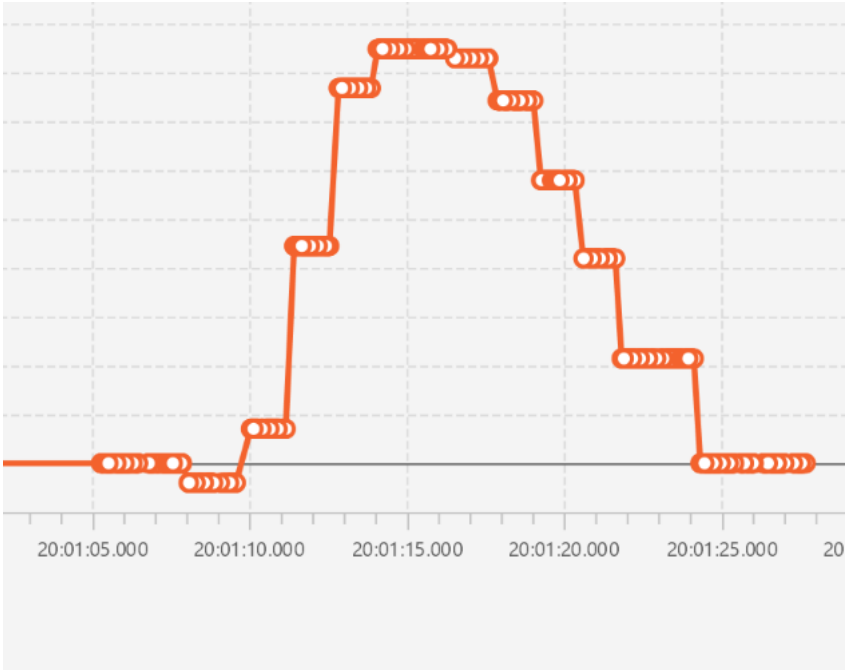

Figura 5.6. Medidas discretas de desplazamiento.

Se puede ver como el tiempo de respuesta es muy bajo y se acumulan muchas muestras con el mismo valor.

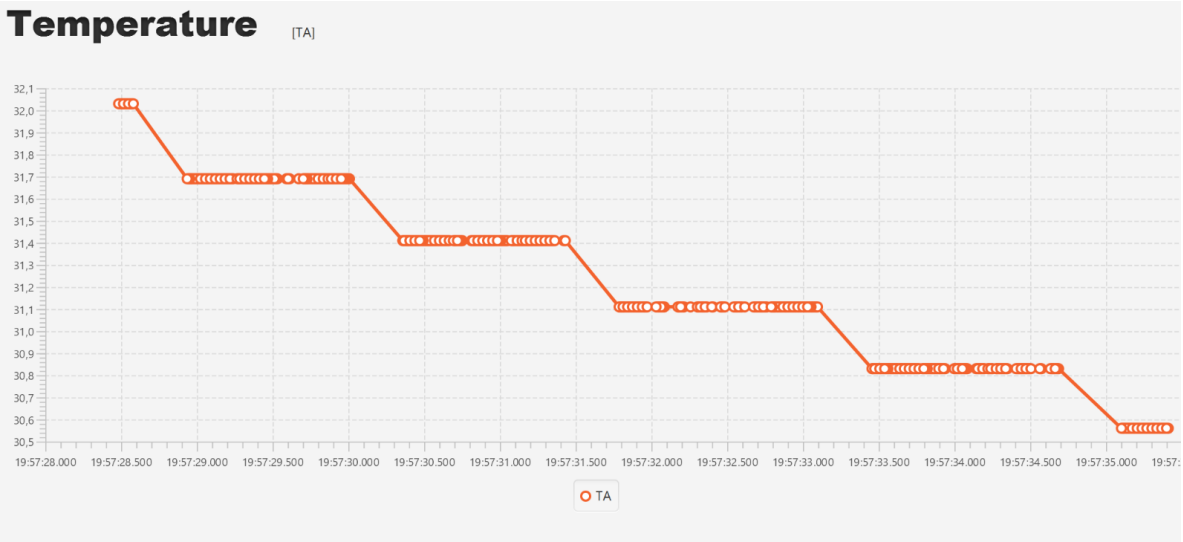

Figura 5.7. Medida de temperatura en el laboratorio

En cuanto a la temperatura, posee el mismo comportamiento este sensor, pero en este importa menos ya que la inercia térmica en una estructura a la intemperie o en un laboratorio es muy baja, y no va a haber grandes cambios de temperatura.

Lo que se hace es mantener el sensor con la mano para subir la temperatura y a continuación dejarla bajar como se aprecia en la Figura 5.7.

6. ESTUDIO ECONÓMICO

En este capítulo se va a llevar a cabo una evaluación económica del proyecto realizado. La realización del proyecto se ha fundamentado en tres grandes etapas: el desarrollo hardware para la medida de parámetros, el desarrollo de un programa informático para la visualización de los datos, así como de los programas para la medición de sensores y por último la redacción de la documentación necesaria.

Concretamente en este Proyecto, la aportación humana ha sido decisiva; además de los medios materiales que se han necesitado, como el equipo informático y el software pertinente. También deberemos tener en cuenta, la documentación que se ha consultado.

Dentro de los gastos, nos encontraremos con dos grupos de costes: los directos y los indirectos. Los directos afectarán de forma inmediata a la realización del proyecto, como por ejemplo los gastos de personal, la amortización del equipo y el coste del material. Los indirectos no revertirán de forma directa para la realización del proyecto, pero, aun no siendo importantes, serán necesarios para llevar a cabo dicho proyecto, y, por tanto, han de ser reflejados.

6.1. Costes directos

Como ya comentamos anteriormente, los costes de personal, han de incluirse como costes directos. Éstos, se componen de los gastos ocasionados por el pago a las personas que han llevado a cabo el proyecto.

Además, deberemos incluir dentro de los costes directos, el coste de amortización de los equipos. Deberemos considerar la amortización de los equipos, ya que verán decrementado su valor con el paso del tiempo, una vez realizada la importante inversión inicial.

Además, consideraremos en este apartado los gastos de material, entre los que se incluyen las materias primas, el material fungible y en general, cualquier tipo de material que esté relacionado de forma directa con la realización del proyecto.

6.1.1. Los costes de personal

Los costes de personal son directamente proporcionales al número de horas empleadas en el Proyecto y a la cualificación de las personas que intervienen en él.

Los costes de personal, son el producto del coste por hora de la persona o personas que participan en el Proyecto, por el número de horas empleado por dicha persona o personas.

Los días laborables, serán el total de días del año, menos aquellos en los que no se ha trabajado.

Concepto	Días
Días totales 2015-2016	365
Fines de semana	-104
Festivos reconocidos legalmente	-12
Vacaciones	-20

Perdidos por enfermad	-10
Perdidos por formación	-4
Perdidos por otros motivos	-1
Total días laborables	213

Tabla 6.1. Calculo de días laborables

Teniendo esto en cuenta, el número de horas trabajadas son el resultado de multiplicar los 213 días trabajados, por las ocho horas diarias que dura la jornada laboral. El número total de horas al año es de 1704 horas trabajadas.

Seguidamente, calcularemos el coste horario de la persona dedicada al Proyecto.

Personal	Sueldo bruto (€/año)	Cotización S.S (35%) (€/año)	Total anual	Coste/hora
Ingeniero	24120	8442	32562	19,11

Tabla 6.2. Cálculo de la hora de un Ingeniero

El coste/ hora se calcula de dividir el total anual entre 1704 horas laborables que posee un año natural.

Especificaremos en una lista las tareas realizadas, con la correspondiente cantidad de tiempo empleada en cada una.

Actividad	Horas
Estudio de los sensores disponibles en el mercado	15
Diseño Hardware	40
Desarrollo Hardware	5
Estudio de la documentación Java	120
Desarrollo de software en Python	100
Desarrollo e instalación de todo el sistema software en la Raspberry Pi	50
Desarrollo del programa JavaFx y software Java para el servidor	160
Elaboración de memoria	100
Número total de horas empleadas	590

Tabla 6.3. Horas empleadas en el proyecto

Con todos estos datos, se calcularán los costes directos de personal.

Personal	Coste horario (€/hora)	Horas empleadas	Total (€)
Ingeniero	19,11	590	11275
Coste total de personal			11275

Tabla 6.4. Costes directos de personal

6.1.2. Los costes de amortización

Todo el material susceptible de ser amortizado, quedará reflejado dentro de este capítulo. Consideraremos, que la amortización es de tipo lineal, con un coeficiente de finalidad, dependiente del número de años considerados para su amortización.

En el proyecto, se ha empleado un ordenador personal para todo el desarrollo software como para la investigación y diseño hardware del proyecto. Además, se ha utilizado una máquina de soldar para el desarrollo de las tarjetas hardware.

El ordenador en el que se desarrolló el software es un ordenador personal Intel i5, 2.2MHz y 8 GB de RAM, monitor de 13.3", teclado, ratón y Windows 10. Este se amortizará a 3 años y la máquina de soldar circuitos a dos años.

Mostraremos a continuación, el coste de los equipos utilizados.

Equipo	Coste bruto (€)	Amortización anual (€/año)	Horas de trabajo al año
Ordenador de desarrollo	820	273,33	1704
Máquina para soldar circuitos	150	75	1704

Tabla 6.5. Amortización equipos

Las horas de trabajo de este tipo de equipos son las mismas que las horas laborables del ingeniero al año.

Equipo	Coste horario (€/hora)	Horas empleadas	Coste total (€)
Ordenador de desarrollo	0,18	585	105,30
Máquina para soldar circuitos	0,044	5	0.22
Coste total de los equipos			105,52

Tabla 6.6. Coste total de los equipos

Respecto al software, durante este proyecto se han empleado programas comerciales como Microsoft Office, Netbeans 8.0.1, Eagle, y PyCharm. Todo este software ha sido utilizado con licencias de estudiante, luego que no tiene coste sobre el proyecto.

6.1.3. Los costes del material

En este apartado se incluyen los gastos del material tanto de componentes Hardware como de material para la redacción de este tomo.

Concepto	Coste (€)
Raspberry Pi	30,05
Tarjeta SD 8GB Kingston	8,06
Cargador 2A Raspberry Pi	6,58
Tarjeta de adquisición de datos + Componentes	16,88
Sensor de fuerza + Acondicionamiento	20,21
Acelerómetro ADXL335	13,16
Calibre digital	12,99
Conexiones	0,50
Total material hardware	108,43

Tabla 6.7. Coste material Hardware

A este coste del material Hardware se debe añadir el coste de la redacción del tomo.

Concepto	Coste (€)
Total material hardware	108,43
Redacción tomo	35,25
Coste total material	143,68

Tabla 6.8. Coste total material

6.2. Costes indirectos

Dentro de este tipo de costes, se engloban todos aquéllos que derivan de las actividades de la empresa en su conjunto. Se formarán por los costes administrativos y los de explotación.

6.2.1. Costes administrativos

Los costes que derivan de las actividades administrativas, se han estimado en

375,00€.

6.2.2. Costes de explotación

En la siguiente tabla, se detallan este tipo de costes, resultado del consumo de energía requerido y manuales consultados.

Concepto	Coste (€)
Consumo eléctrico	315,00
Documentación	65,00
Material de oficina	25,00
Costes totales de explotación	405,00

Tabla 6.9. Costes totales de explotación

6.3. Costes totales

Los costes totales, serán la suma de los costes directos más los costes indirectos, se puede ver en la siguiente tabla.

Concepto	Coste
Costes directos de personal	11275,00
Costes directos de amortización	105,52
Costes directos de material	143,68
Costes indirectos administrativos	375,00
Costes indirectos de explotación	405,00
Coste total del proyecto	12304,2

Tabla 6.10. Coste total del proyecto

Así pues, el coste total del proyecto asciende a doce mil trescientos cuatro con veinte euros.

De este total, el 93,66% de los costes corresponde a costes directos y el restante son costes indirectos.

Por otra parte, el coste de personal es el 91,63% del total de costes, valor muy elevado como se ha comentado al principio del capítulo.

7. CONCLUSIONES Y LÍNEAS FUTURAS

En este capítulo se pretende exponer una serie de conclusiones que se han obtenido con la realización de este proyecto, así como las líneas futuras que se proponen para la continuación del mismo.

7.1. Conclusiones

Cambiando la línea de proyectos similares que se habían realizado anteriormente, en este caso se ha seleccionado una placa de adquisición de datos ya fabricada como es la Raspberry Pi, apostando así por una conectividad a internet de esta y una disposición de los datos en la nube. En proyectos anteriores se seleccionó un Arduino que no tiene la velocidad de procesamiento y la facilidad de conectividad de una Raspberry Pi.

Además, posee un sistema operativo en el cual se puede instalar un servidor que gestione una base de datos en la cual se puede gestionar un gran volumen de datos con una fácil accesibilidad a ellos.

En cuanto a los motivos económicos de la selección de una Raspberry Pi, claramente se ha optado por la compra, ya que la fabricación de una placa de este tipo consta de un gran coste de desarrollo, y no merece la pena ya que es fácil comprarla y por un precio asequible.

Un inconveniente común en una placa de este tipo utilizada para este tipo de mediciones es que posee microprocesador y no microcontrolador, con lo que las muestras obtenidas por los sensores no son equidistantes en el tiempo, pero no es mayor inconveniente a la hora de analizar estos datos.

Y por último el otro gran inconveniente es la integración del sistema con baterías, una Raspberry Pi puede llegar a consumir 700mA, con lo que este sistema se podría usar durante horas con alimentación autónoma. Pero este inconveniente no es tan grave ya que se puede disponer de alimentación en determinadas estructuras reales.

Se ha optado por fabricar una placa para leer entradas analógicas, así como otros sensores como pueden ser fuerza y desplazamiento con la misma. La opción de fabricar esta placa ha sido claramente por motivos económicos (Ahorro de un 300% respecto a placas comerciales).

En cuanto a los sensores seleccionados para el proyecto, el acelerómetro MEMs ha sido una gran elección por su precio y los resultados obtenidos frente a los acelerómetros piezoeléctricos comerciales.

Para el sensor de fuerza, la opción seleccionada ha sido la utilizada en métodos comerciales, pero con un sistema de instrumentación más barato con lo que se han notado las diferencias de resultados.

Los sensores de desplazamiento han sido un inconveniente a la hora de realizar problemas dinámicos, pero ya que el proyecto conlleva un sistema de bajo coste se han tenido que seleccionar de este tipo, los sensores comerciales suelen ser de tecnología láser, con los cuales se obtienen grandes resultados.

En cuanto al sensor de temperatura ha sido una gran elección por su coste y su bajo tiempo en desarrollo software, ya que este dispone de una gran cantidad de librerías en internet. Se puede observar que el precio de este sensor no se ha incluido en el coste de material del proyecto debido a su bajo coste, casi despreciable.

Para finalizar en cuanto al bloque de desarrollo software, se ha optado por una

aplicación de escritorio que permite una independencia mayor que las hasta ahora desarrolladas aplicaciones cliente-servidor.

Además, se ha optado por un lenguaje orientado a objetos como es Java para el desarrollo de servicios web (parte del servidor), así como JavaFX para la aplicación de escritorio, con lo que se ha obtenido una interfaz estéticamente buena y rápida.

En cuanto a las conclusiones a nivel técnico-económico, se ha realizado una comparación entre los sistemas de medición que posee la escuela, como pueden ser el CompactRIO y el SIRIUS-HD 16xSTGS, con el sistema desarrollado en este proyecto.

En primer lugar, CompactRIO es un controlador que incluye procesador y un FPGA reconfigurable. El procesador tiene la posibilidad de ser usado para una comunicación en red, registro de datos, control y procesamiento con el Sistema Operativo deseado. La FPGA es usada por el usuario para implementar hardware personalizado para control de alta velocidad, procesamiento de datos en línea o temporización y disparo complejos. (Instruments N. , ni.com, 2016)

El precio de este sistema puede variar 6000-7000€, incluyendo controlador más interfaz más chasis, luego se le deben añadir los módulos de acelerómetros, fuerza, etc. Un módulo de E/S de aceleración puede oscilar entre los 700-1800€.

En segundo lugar, se tiene un Sirius-HD modelo 16xSTGS, el cual tiene 16 entradas analógicas con conector DB9, posee configuraciones internas de puentes Wheastone, convertidores analógico a digitales Delta-Sigma de 24 bits.

El precio exacto de un sistema de esta marca sin incluir sensores es de 10.196€, un acelerómetro piezoeléctrico tiene un precio de 300€ aproximadamente.

Por último, el sistema creado en este proyecto tiene un coste de 108,43€ con lo que se puede ver que los precios tienen una diferencia importante, esto se basa en la frecuencia de muestreo, este sistema no muestrea a las velocidades proporcionadas por los sistemas de alto coste, se puede muestrear hasta 120Hz la aceleración, con un sistema de alto coste se puede muestrear hasta 20KHz.

También destacar el soporte que proporcionan estas marcas frente a posibles fallos y mejoras también. Pero destacar que el sistema de bajo coste creado puede ser una opción viable para ensayos en laboratorios con su posible futuro desarrollo para estructuras reales, invirtiendo más dinero en él con lo que obtener un sistema más caro, pero a su vez más funcional sin sobrepasar la barrera de los 1000€.

7.2. Líneas futuras

En cuanto a las líneas futuras principales para continuar este proyecto, la primera sería desarrollar un nuevo sensor de desplazamiento con los encoders magnéticos descritos en el apartado de hardware 3.4.3.

Integrar sensores de temperatura PT100 con las galgas extensiométricas o células de carga para corregir la deriva de temperatura de estos sensores.

Portar la aplicación de escritorio a una aplicación móvil para la visualización de cualquier tipo de sensor sin necesidad de disponer de un ordenador personal.

Conexión con otros microcontroladores como Arduino, para la transmisión de datos de otro tipo de sensores que se midan con estas placas, esta conexión puede ser mediante cable y puerto serie o inalámbrica mediante radiofrecuencia, por ejemplo,

teniendo el sistema creado para este proyecto como un nodo central que además recibe datos de nodos externos.

Posibilidad de lectura de sensores mediante tarjetas fabricadas por Texas Instruments, llamadas LaunchPad como la que se muestra en la Figura 7.1, están placas tienen características que responden perfectamente a los requisitos del proyecto.

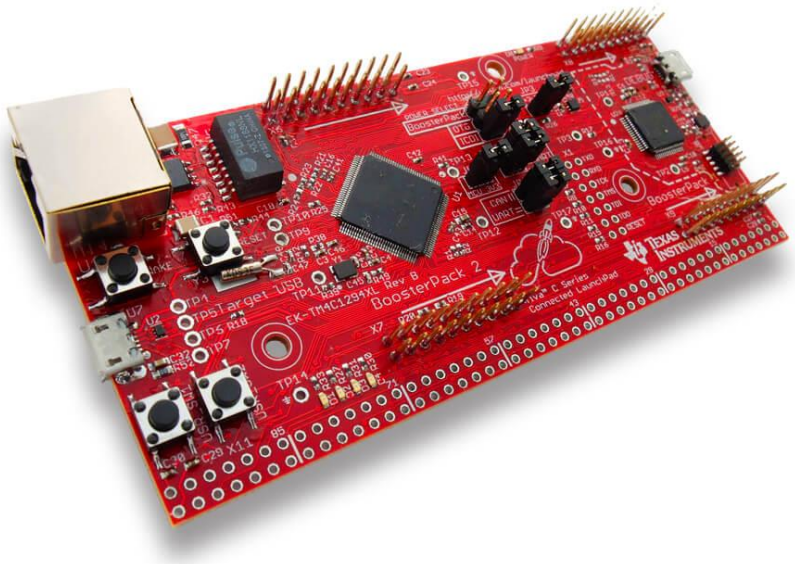

Figura 7.1. Launchpad de Texas Instruments

Las características de las Launchpad como la versión mostrada en la Figura 7.1 son:

- Tiva Series C, EK-TM4C1294XL
- Conectividad a la nube mediante un software llamado Exoxite
- 16 entradas analógicas de 12 bit de resolución
- Frecuencia de reloj 120 MHz, lógicamente no se puede muestrear a esta frecuencia, pero se debería hacer un análisis de tiempo de muestreo de sensores
- Facilidad de programación con un IDE parecido a Arduino
- Precio de 20$ más gastos de envió desde EEUU

El único inconveniente encontrado para esta solución sería el almacenamiento interno, ya que posee una memoria Flash de 1MB, pero en el caso de que no se tenga internet y no se puedan subir los datos a la nube, no se podrían borrar datos del dispositivo, con lo que el espacio sería insuficiente. La línea futura seria encontrar una solución a este inconveniente.

Por último, la implementación del sistema en estructuras reales seria otra línea futura, ya que solo ha sido testeado en maquetas estructurales, pero para este propósito se debe desarrollar su correspondiente envoltura para proteger la electrónica a la

intemperie.

BIBLIOGRAFÍA

Adafruit. (28 de Mayo de 2016). *adafruit.com.* Obtenido de https://cdn-learn.adafruit.com/assets/assets/000/010/126/original/fsrguide.pdf

AMS. (28 de Mayo de 2016). *www.ams.com*. Obtenido de http://ams.com/eng/Products/Magnetic-Position-Sensors/Linear-Position/NSE-5310

Balageas, D., Fritzen, C.-P., & Güemes, A. (2006). *Structural Health Monitoring.* London: ISTE Ltd.

Christian Grosse, G. M. (2010). A hybrid wireless sensor network for acoustic emission testing in SHM . *SPIE Digital Library*, 9.

Devices, A. (25 de Mayo de 2016). *analog.com ADXL335.* Obtenido de http://www.analog.com/media/en/technical-documentation/data-sheets/ADXL335.pdf

DFRobot. (28 de Mayo de 2016). *www.dfrobot.com*. Obtenido de http://www.dfrobot.com/wiki/index.php?title=Weight_Sensor_Module_SKU:SEN0160

DLNWare. (22 de Mayo de 2016). Obtenido de http://dlnware.com/i2c

Ingenieros, A. (13 de Mayo de 2016). *Alava Ingenieros.* Obtenido de http://www.alava-ing.es: http://www.alava-ing.es/repositorio/4d5a/pdf/3350/2/comparativa-extensometria-con-fibra-optica.pdf

Instruments, N. (02 de Mayo de 2016). *Fundamentos de la deteccion optica FBG.* Obtenido de http://www.ni.com/white-paper/11821/es/

Instruments, N. (18 de Junio de 2016). *ni.com*. Obtenido de http://www.ni.com/compactrio/whatis/esa/

Instruments, T. (23 de Mayo de 2016). *ADS1262 ti.com.* Obtenido de http://www.ti.com/lit/ds/symlink/ads1262.pdf

Instruments, T. (22 de Mayo de 2016). *ti.com.* Obtenido de http://www.ti.com/lit/ds/symlink/ads1115.pdf

Instruments, T. (24 de Mayo de 2016). *ti.com TPS709.* Obtenido de http://www.ti.com/lit/ds/symlink/tps709.pdf

Instruments, T. (24 de Mayo de 2016). *ti.com TXB0104.* Obtenido de http://www.ti.com/lit/ds/symlink/txb0104.pdf

Jesús de Sebastián, A. E. (2013). A LOW-COST VIBRATION MONITORING SYSTEM FOR A STRESS-RIBBON FOOTBRIDGE. *ResearchGate*, 15.

KARBHARI, V. M. (2009). Introduction: structural health monitoring –a means to optimal design in the future. En V. M. KARBHARI, *Structural Health Monitoring of Civil Infrastructure Systems* (págs. 517-528).

Llorente, A. M. (1 de Noviembre de 2014). SISTEMA DE MONITORIZACIÓN ESTRUCTURAL MEDIANTE SENSORES Y ELECTRÓNICA DE BAJO COSTE . Valladolid, Valladolid, España.

MAREK BARSKI, P. K. (2014). STRUCTURAL HEALTH MONITORING (SHM) METHODS IN MACHINE DESIGN AND OPERATION. *archive the mechanical engineering*, 25.

Meissner, J. (13 de Mayo de 2016). *Aos-fiber.* Obtenido de http://www.monico-eu.org/documents/Workshop/4.%20Meisner%20-%20Fiber%20Bragg%20Grating%20Technologies%20for%20Structural%20Health%20Monitoring.pdf

Microchip. (23 de Mayo de 2016). *Microchip Web.* Obtenido de http://ww1.microchip.com/downloads/en/devicedoc/spi.pdf

Plankis, A. (1 de Julio de 2012). STRUCTURAL HEALTH MONITORING MEMS SENSORS USING ELASTICITY-BASED BEAM VIBRATIONS.

Python. (05 de Junio de 2016). Obtenido de https://docs.python.org/3/license.html

Raspberry pi Geek. (15 de Mayo de 2016). Obtenido de http://www.raspberry-pi-geek.com/howto/GPIO-Pinout-Rasp-Pi-1-Model-B-Rasp-Pi-2-Model-B

Sensors Mag. (02 de Mayo de 2016). Obtenido de http://archives.sensorsmag.com/articles/0598/tri0598/

Sensors, L. S. (28 de Mayo de 2016). *www.loadstarsensors.com.* Obtenido de http://www.loadstarsensors.com/what-is-a-load-cell.html

Sethi, A. (02 de Mayo de 2016). *Structural health monitoring of steel structures using electrical strain gauges.* Obtenido de http://web.iitd.ac.in/~sbhalla/thesispdf/sethi.pdf

Sparkun. (23 de Mayo de 2016). *sparkfun.com.* Obtenido de https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf

Sukun Kim, S. P. (2006). Wireless Sensor Networks for Structural Health Monitoring. 2.

Sukun Kim, S. P. (13 de Mayo de 2016). *cs.berkeley.edu.* Obtenido de cs.berkeley.edu: http://www.cs.berkeley.edu/~binetude/ggb/

Wordnice. (15 de Mayo de 2016). Obtenido de http://wordnice.net/info/raspberry-pi-2-model-b-caracteristicas-t1044.html

ANEXOS