



industriales

etsii

Escuela Técnica
Superior
de Ingeniería
Industrial

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Escuela Técnica Superior de Ingeniería
Industrial

Sistema de adquisición y control de lisímetro de pesada en maceta con Arduino

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA EN TECNOLOGÍAS
INDUSTRIALES

| | |
|---------------|-------------------------------|
| Autor: | Adrián Antolino Merino |
| Director: | Manuel Jiménez Buendía |
| Codirector: | José Alfonso Vera Repullo |

Cartagena. Septiembre de 2016



Universidad
Politécnica
de Cartagena

Agradecimientos

A José Alfonso Vera y Manuel Jiménez, por permitirme participar en este proyecto y trabajar a su lado.

A Amanda, que me acompañó de cerca durante esta última etapa y vivió conmigo tanto los buenos como los malos momentos.

A mi madre y hermanos, por su apoyo incondicional durante estos duros años de carrera. Sin mi familia nunca habría llegado a donde estoy.

Y en especial a mi padre, por ayudarme a dar los primeros pasos en el mundo de la ingeniería y contagiarme su entusiasmo.

*"Vive como si fueras a morir mañana.
Aprende como si fueras a vivir para siempre"*
(Gandhi)

ÍNDICE DE CONTENIDOS

| | |
|--|----|
| Capítulo 1. Introducción y objetivos | 1 |
| 1.1 Introducción..... | 2 |
| 1.2 Planteamiento inicial del proyecto | 2 |
| 1.3 Objetivos del proyecto..... | 3 |
| Capítulo 2. Estado del Arte..... | 5 |
| 2.1 Agricultura de precisión | 6 |
| 2.2 Evapotranspiración. Balance hídrico | 9 |
| 2.3 Introducción a la lisimetría. | 12 |
| 2.4 Plataforma Open-source. Arduino board | 16 |
| Capítulo 3. Descripción del Hardware | 20 |
| 3.1 Sensores..... | 21 |
| 3.1.1 Células de carga..... | 21 |
| 3.1.2 Caudalímetro de precisión..... | 29 |
| 3.1.3 Convertidor Analógico - Digital..... | 30 |
| 3.1.4 Optoacoplador | 36 |
| 3.1.5 Real Time Clock..... | 36 |
| 3.1.6 Módulo HY – 1.8 SPI..... | 37 |
| 3.2 Elementos actuadores..... | 38 |
| 3.2.1 Electroválvulas de 230V..... | 38 |
| 3.2.2 Relé de 230V..... | 39 |
| 3.3 Dispositivos de Control..... | 40 |
| 3.3.1 Arduino Uno | 40 |
| 3.3.2 Programador de riego..... | 41 |
| 3.4 Shields para Arduino | 42 |
| 3.4.1 Shield para el convertidor CS5534-ASZ..... | 42 |
| 3.4.2 Shield con módulo HY-1.8 SPI y DS1302..... | 45 |
| 3.4.3 Shield Ethernet | 47 |

| | |
|---|-----|
| Capítulo 4. Sistema de Instrumentación | 49 |
| 4.1 Descripción general | 50 |
| 4.2 Filtrado de datos | 53 |
| 4.3 Configuraciones del sistema | 56 |
| Capítulo 5. Software desarrollado..... | 68 |
| 5.1 Configuración CS5534-ASZ | 69 |
| 5.2 Flujograma del Arduino de Adquisición..... | 73 |
| 5.3 Modelo A: Ensayo en el laboratorio. LabView..... | 74 |
| 5.4 Modelo B: Ensayo en finca (I). Shield HY-1.8 SPI con DS1302..... | 82 |
| 5.5 Modelo C: Ensayo en la finca (II). Shield Ethernet..... | 87 |
| 5.5.1 <i>Lisimetro-upct</i> | 88 |
| 5.5.2 <i>Ubidots</i> | 102 |
| Capítulo 6. Conclusiones y líneas futuras | 110 |
| Capítulo 7. Bibliografía | 115 |
| Capítulo 8. Anexos | 119 |
| Anexo I. Funciones del Arduino de Adquisición | 120 |
| Anexo II. Manejo del Arduino de Adquisición..... | 122 |
| Anexo III. Pruebas de los convertidores AD | 126 |
| <i>Prueba 1. Calibradores de precisión</i> | 126 |
| <i>Prueba 2 (I). Células de carga. Alimentación desde calibrador.</i> <i>Subida y bajada de pesos</i> | 132 |
| <i>Prueba 2 (II). Células de carga. Alimentación desde calibrador.</i> <i>Subida y bajada de voltaje</i> | 135 |
| <i>Prueba 3. Calibración de células de carga. Alimentación desde Arduino</i> | 137 |
| <i>Prueba 4. Calibración de células de carga. Alimentación desde Arduino.</i> <i>Guarda Activa</i> | 138 |

ÍNDICE DE FIGURAS

| | |
|---|----|
| Figura 2.1: Las tres etapas de la agricultura de precisión. | 6 |
| Figura 2.2: Sistema de posicionamiento global (GPS) | 7 |
| Figura 2.3: Sistema de información geográfica (GIS) | 8 |
| Figura 2.4: Evapotranspiración durante el crecimiento del cultivo | 10 |
| Figura 2.5: Balance hídrico del cultivo | 11 |
| Figura 2.6: Lisímetro de drenaje..... | 13 |
| Figura 2.7: Lisímetro de pesada | 13 |
| Figura 2.8: Lisímetro de pesada en maceta: Prototipo, Lisímetro utilizado | 15 |
| Figura 2.9: Placa PCB de Arduino..... | 18 |
| Figura 2.10: Entorno de desarrollo Integrado (IDE) | 20 |
| Figura 3.1: Células de carga..... | 21 |
| Figura 3.2: Comportamiento de las Galgas extensiométricas..... | 22 |
| Figura 3.3: Diferentes tipos de puente Wheatstone..... | 23 |
| Figura 2.4: Célula de carga Vetek 108TA | 24 |
| Figura 3.5: Certificado de calibración | 26 |
| Figura 3.6: Recta de calibración de la célula 4. Bola de drenaje | 27 |
| Figura 3.7: Rectas de calibración de las células de la estructura..... | 28 |
| Figura 3.8: Caudalímetro de precisión. Tabla de características | 29 |
| Figura 3.9: Fases de conversión Analógica Digital | 30 |
| Figura 3.10: Esquema interno del CS5534 | 31 |
| Figura 3.11: esquemático CS5532 para alimentación simple de 5V..... | 32 |
| Figura 3.12: Convertidor HX711 | 33 |
| Figura 3.13: Diagrama del ACSL - 6410..... | 36 |
| Figura 3.14: RTC modelo DS1302..... | 36 |
| Figura 3.15: Módulo HY -1.8 SPI..... | 37 |
| Figura 3.16: Electroválvula para drenaje | 38 |
| Figura 3.17: Electroválvula para riego | 39 |
| Figura 3.18: Módulo relé para electroválvulas..... | 39 |
| Figura 3.19: Placa Arduino Uno | 40 |
| Figura 3.20: Programador digital Coati 13201 | 41 |
| Figura 3.21: Shield para el convertidor CS5534-ASZ..... | 43 |
| Figura 3.22: Esquemático del shield CS5534-ASZ. Aplicación Eagle..... | 44 |
| Figura 3.23: Shield con HY-1.8 SPI y DS1302 | 45 |
| Figura 3.24: Esquemático del shield HY-1.8 SPI y DS1302. Aplicación Eagle | 46 |
| Figura 3.25: Shield Ethernet | 47 |
| Figura 4.1: Gráfica de vaciado de bola. Bola de drenaje..... | 50 |
| Figura 4.2: Programador de riego | 51 |
| Figura 4.3: Elementos del lisímetro. Nodo de Adquisición | 52 |
| Figura 4.4: Estación Experimental Agroalimentaria Tomás Ferro | 53 |
| Figura 4.5: Lisímetro en el Laboratorio y en la Finca | 56 |
| Figura 4.6: Estructura general del sistema (Modelo A)..... | 57 |
| Figura 4.7: Programadores | 57 |

| | |
|--|-----|
| Figura 4.8: Conexión del Arduino de Adquisición en el Modelo A..... | 58 |
| Figura 4.9: Estructura general del sistema (Modelo B) | 59 |
| Figura 4.10: Caja estanca para la finca | 60 |
| Figura 4.11: Arduino de Registro, Arduino de Adquisición y relé | 60 |
| Figura 4.12: Conexión en el Modelo B: Finca (I) | 63 |
| Figura 4.13: Estructura general del sistema (Modelo C) | 64 |
| Figura 4.14: Conexión en el Modelo C: Finca (II) | 66 |
| Figura 5.1: Diagrama de registros del CS5534-ASZ..... | 69 |
| Figura 5.2: Ciclo de escritura, CS5534..... | 70 |
| Figura 5.3: Ciclo de lectura, CS5534 | 71 |
| Figura 5.4: Menú principal. Navegación por el CS5534-ASZ | 71 |
| Figura 5.5: Arduino de Adquisición. Flujograma..... | 73 |
| Figura 5.6: Ejemplo de Panel frontal y trasero | 75 |
| Figura 5.7: Panel frontal de LabView. Inicio | 76 |
| Figura 5.8: Panel frontal de LabView. Gráfica | 77 |
| Figura 5.9: Diagrama de bloques (I)..... | 78 |
| Figura 5.10: Diagrama de bloques (II) | 78 |
| Figura 5.11: Diagrama de bloques (III) | 79 |
| Figura 5.12: Aplicación de LabView para calibración de células | 80 |
| Figura 5.13: Aplicación de LabView con riego programable..... | 81 |
| Figura 5.14: Módulo HY- 1.8 SPI, DS1302 y Arduino de Registro..... | 82 |
| Figura 5.15: Arduino de Registro. Flujograma..... | 85 |
| Figura 5.16: Arduino de Adquisición para finca (I). Modo normal y error..... | 86 |
| Figura 5.17: Menú de acceso..... | 88 |
| Figura 5.18: Menú principal | 89 |
| Figura 5.19: Menú "Mostrar datos" | 89 |
| Figura 5.20: Menú "Configuración" | 90 |
| Figura 5.21: Hostinger. Acceso FTP a nuestra página | 93 |
| Figura 5.22: Hostinger. Base de datos MySQL..... | 93 |
| Figura 5.23: Filezilla. Contenido de página web y Gestor de sitios..... | 94 |
| Figura 5.24: Ubidots permite implementar nuestros proyectos en la nube | 102 |
| Figura 5.25: Ejemplo de mesa de trabajo con diferentes widgets | 103 |
| Figura 5.26: Diferentes tipos de widgets que podemos generar..... | 103 |
| Figura 5.27: Respuestas programadas | 104 |
| Figura 5.28: Evolución del peso del lisímetro | 105 |
| Figura 5.29: Indicador bola drenaje. Variables booleanas..... | 105 |
| Figura 5.30: Compartir un Widget. Configurar Widget..... | 106 |
| Figura 5.31: Ubidots. Variables y sus IDs correspondientes | 107 |
| Figura 5.32: Ubidots. Código Token de la cuenta..... | 107 |
| Figura 6.1: Resultados obtenidos en el Laboratorio | 112 |
| Figura 6.2: Resultados obtenidos en la finca Tomás Ferro..... | 112 |
| Figura II.1: Prueba 1. CS5534 Unipolar. Resultados | 128 |
| Figura II.2: Prueba 1. CS5534 Bipolar. Resultados..... | 129 |
| Figura II.3: Prueba1. HX711. Error cometido | 131 |
| Figura II.4: Prueba1. HX711. Resultados | 131 |

ÍNDICE DE TABLAS

| | |
|---|-----|
| Tabla 1: Comparativa entre modelos de Arduino | 19 |
| Tabla 2: Hoja de características, Célula de carga Vetek 108TA | 25 |
| Tabla 3: Límites aplicados al filtrado de datos..... | 54 |
| Tabla 4: Base de datos, Arduino..... | 91 |
| Tabla 5: Base de datos, Configuración..... | 92 |
| Tabla 6: Base de datos, Usuarios | 92 |
| Tabla 7: Prueba 1. CS5534 Unipolar. Resultados | 127 |
| Tabla 8: Prueba 1. CS5534 Bipolar. Resultados | 129 |
| Tabla 9: Prueba 1. HX711. Resultados | 130 |
| Tabla 10: Pesas calibradas disponibles | 132 |
| Tabla 11: Prueba 2 (I). CS5534 Unipolar. Resultados..... | 133 |
| Tabla 12: Prueba 2 (I). CS5534 Bipolar. Resultados | 133 |
| Tabla 13: Prueba 2 (I). HX711. Resultados..... | 134 |
| Tabla 14: Prueba 2 (II). CS5534 Unipolar. Resultados | 135 |
| Tabla 15: Prueba 2 (II). CS5534 Bipolar. Resultados | 136 |
| Tabla 16: Prueba 2 (II). HX711. Resultados | 136 |
| Tabla 17: Prueba 3. CS5534 Unipolar. Resultados | 137 |
| Tabla 18: Prueba 3. HX711. Resultados | 137 |
| Tabla 19: Prueba 4. CS5534 Unipolar. Resultados | 138 |



Capítulo 1.

Introducción y objetivos

En este primer capítulo se expone la introducción al trabajo, así como los objetivos establecidos para su realización.

1.1 Introducción

Se espera que la mayor parte del incremento de población global tendrá lugar en un tercio de los países que actualmente sufren problemas de abastecimiento hídrico y sanitario. Este crecimiento de población, junto con los altos estándares de vida actuales, causa el aumento de demanda de agua de calidad para uso municipal e industrial, así como el destinado a irrigación para cubrir las necesidades alimentarias. Por esta razón, podemos esperar el aumento progresivo de la competencia por estos recursos hídricos, lo cual requiere de una gestión intensiva y de una cooperación a nivel internacional (Bouwer 2000).

La gestión del riego tiene como objetivo la optimización y eficiencia del uso del agua, no solo para una mejora de la calidad y rendimiento de los cultivos, sino para evitar además su derroche evitando el lixiviado de la misma. Aunque se han hecho grandes avances en la investigación de sistemas tecnológicos de irrigación, su uso sigue siendo una excepción frente al uso de los viejos sistemas mucho más ineficaces. Estos avances en los sistemas de riego de precisión están limitados debido a los costes de los sistemas de control específicos propios de cada localización. No obstante, debido a los crecientes avances en la tecnología electrónica, combinada con internet y las tecnologías de la información, permiten vencer estas limitaciones en el campo del riego de precisión (Chávez et al. 2010).

La presencia de métodos de adquisición de datos de bajo coste, sistemas de posicionamiento y el desarrollo de programación computacional, ayudará a determinar y regular la cantidad de agua requerida dentro de un sector del campo. Los sistemas de riego de precisión tienen como objetivo aplicar en el momento adecuado la cantidad necesaria de agua en el lugar donde es necesitada. Por lo que, los nuevos esquemas de riego actuales buscan responder a cuando, donde y que cantidad de agua regar (Al-Karadsheh, Sourell, and Sommer 2002).

Uno de los métodos más fiables para determinar las necesidades hídricas de un cultivo es la utilización de lisímetros, puesto que es el único equipo de medida que mide directamente el balance hídrico de un cultivo para lo que utiliza una serie de sensores y electroválvulas. Permite realizar experimentos bajo diferentes condiciones (variedad de planta, composición de la tierra, condiciones atmosféricas, horas de riego y duración) y extrapolar el comportamiento a grandes extensiones de terreno, permitiendo de esta manera optimizar los recursos hídricos según sea el caso.

1.2 Planteamiento inicial del proyecto

Con este TFG se pretende diseñar el sistema de instrumentación de bajo coste para lisímetro de pesada en maceta, para lo que haremos uso de la plataforma open-source

Arduino. El lisímetro de pesada en maceta es el resultado de la tesis doctoral de Leandro Ruiz Peñalver (Ruiz-Peñalver et al. 2015).

En primer lugar, se utilizará el lisímetro en un laboratorio, un entorno cerrado para realizar ensayos de control. Después, se instalará el lisímetro al aire libre para estudiar su comportamiento bajo condiciones climatológicas reales.

En nuestro diseño, la placa de Arduino hará las veces de Datalogger. Se encargará de leer las diferentes señales de entrada (hora /fecha, agua regada y lecturas de peso) y activar los actuadores (permitir el riego y vaciar los depósitos). Para poder almacenar los datos, se hará uso de otro dispositivo externo con el cual la placa Arduino tendrá que comunicarse.

Durante el desarrollo del TFG se han diseñado tres modelos diferentes de estructura y conexionado, en función de a que dispositivo se conecte el Arduino:

- Modelo A: Ensayos en laboratorio. Se conectará a un ordenador.
- Modelo B: Ensayo en finca I. Se conectará a otro Arduino, el cual dispondrá de ranura SD.
- Modelo C: Ensayo en finca II (Mejora). Se conectará a otro Arduino, el cual dispondrá de un shield Ethernet

1.3 Objetivos del proyecto

En este apartado se van a definir las metas a alcanzar durante el desarrollo del proyecto:

- a) Seleccionar el convertidor AD de 24 bits más apropiado para la adquisición de las señales de la instrumentación utilizada en el lisímetro. Se estudiarán las alternativas HX711 y CS5534.
- b) Calibración de la instrumentación. AD, células de carga y caudalímetro de precisión.
- c) Diseño de la interfaz PC (Modelo A) para realizar los ensayos de control en el laboratorio. Se utilizará LabView.
- d) Incorporar un shield con pantalla TFT, ranura SD y Real Time Clock (Modelo B) para almacenaje de lecturas en la finca.
- e) Estudiar la alternativa de un shield Ethernet (Modelo C).



Capítulo 2.

Estado del Arte

En este segundo capítulo se describe la situación actual de los sistemas tecnológicos sobre los que se ha basado este trabajo.

2.1 Agricultura de precisión

Introducción

Agricultura de precisión es un concepto que surgió en nuestro país durante el año 1995, en la cual se utilizan tecnologías de la información y la electrónica para recoger datos e información en tiempo real sobre un cultivo o suelo. El objetivo, por tanto, es suministrar al cultivo el insumo adecuado, en el lugar, hora y cantidad precisa. De esta manera se permite el control y manejo óptimo de zonas de cultivo que varían desde pequeños cultivos a grandes extensiones de terreno.

La agricultura de precisión buscar recoger la mayor cantidad de datos significativos y elaborar una estrategia de gestión para tomar una serie de decisiones en los procesos de producción que se apoyen en la estadística.

En la figura 2.1 se puede observar las etapas de la agricultura de precisión y sus tecnologías involucradas:

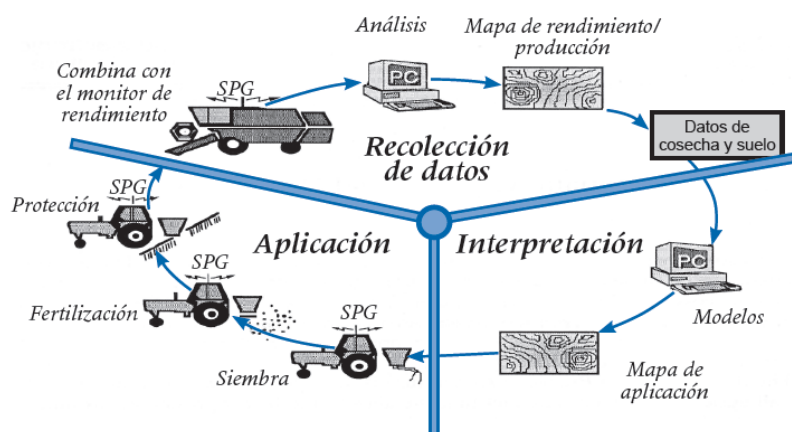


Figura 2.1: Las tres etapas de la agricultura de precisión (Ortega et al. 1999a).

La etapa de recolección de datos supone la medición de variables como topografía y propiedades de suelo, detección de plagas, monitoreo de rendimiento. Incluye el uso de sistemas de posicionamiento global (GPS), instrumentos de topografía, así como todo tipo de sensores directos y remotos.

La etapa de análisis conlleva el procesamiento y la interpretación de información anteriormente recogida, para determinar si hay dependencia entre variables y establecer modelos y patrones. Se utilizan toda clase de programas estadísticos así como Sistemas de información geográfica (SIG) que permite confeccionar mapas de información en niveles o capas.

La etapa final supone la aplicación la aplicación variable de insumos: agua, fertilizantes, plaguicidas,.. Esta tecnología de dosis variables está automatizada mediante programas y sistemas GPS.

Los beneficios directos de la agricultura de precisión son:

- Se incrementa el rendimiento de los insumos utilizados
- Por tanto, se reduce el coste de insumos utilizados
- Mayor productividad y calidad de los cultivos (debido a combinación de necesidades agrícolas y los insumos aplicados de manera óptima.

Resumiendo, la agricultura de precisión ha sido posible debido a los avances tecnológicos en (Ortega et al. 1999a):

1. Sistemas de posicionamiento global (GPS).
2. Sistemas de información geográfica (GIS).
3. Adquisición remota.
4. Tecnologías de dosis variable. Georreferenciación.

Sistemas de posicionamiento global (GPS)

Actualmente, encontramos diferentes tecnologías creadas para el posicionamiento en el campo, aunque los sistemas globales de navegación por satélites (GNSS) son de lejos los más usados. Su concepción inicial fue para uso militar en la década de 1990, pero más tarde se introdujo en la comunidad civil y permitió la medición y navegación de grandes áreas y distancias. En un principio, se utilizó en aviación y náutica, aunque más tarde revolucionó el mundo rural.

Su tecnología se basa en la triangulación de la posición mediante satélites destinados para este propósito, los cuales poseen orbitas que son conocidas; esto permite que, sabiendo la posición relativa de estos satélites, por comparación se puede determinar la posición del usuario en la superficie de la tierra con una precisión que varía entre los 5 y los 20 metros. Un ejemplo de los sistemas GNSS es el famoso GPS o sistema de posicionamiento global, siendo el más utilizado en todo el mundo. (Fernández and others 2012)



Figura 2.2: Sistema de posicionamiento global (GPS)

Sistema de información geográfica (GIS)

Después de los sistemas de navegación global por satélites (GNSS), la herramienta más utilizada en la agricultura de precisión son los Sistemas de información geográfica

(GIS). El GPS por sí solo no concibe la agricultura de precisión. Para poder interpretar los datos y realizar una toma de decisión, se necesita de un sistema que acceda a todos estos datos recopilados de una manera organizada, permita interactuar con ellos y obtener análisis de ellos. De aquí nace el Sistema de información geográfica.

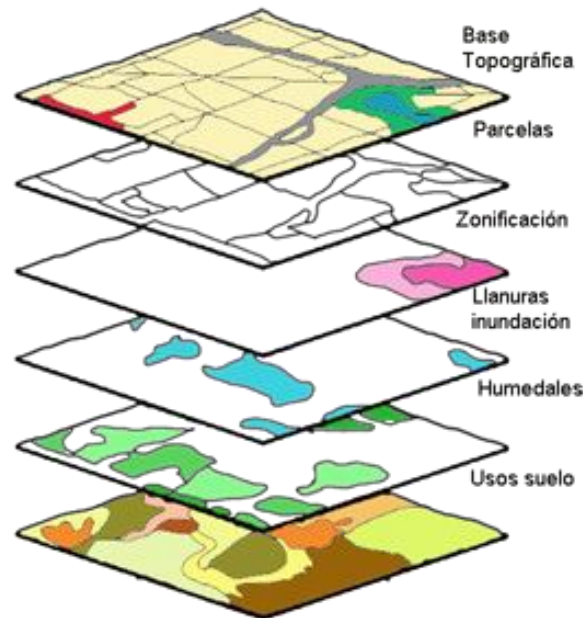


Figura 2.3: Sistema de información geográfica (GIS)

Un GIS es un software cuyo objetivo es almacenar, analizar y representar datos de carácter cartográfico. Permite el análisis de patrones, modelos y simulaciones. Mediante un GIS no se representa la superficie terrestre como un mapa o dibujo, si no como datos e información (Figura 2.3).

Todos estos datos espaciales se almacenan de forma estructurada. Encontraremos una capa para ríos, vegetaciones, asentamientos humanos, etc. Es una herramienta muy útil dentro de la AP puesto que integra datos espaciales de diferentes naturalezas, permitiendo relacionar gran cantidad de datos. (Fernández and others 2012)

Este tipo de tecnología permite por tanto la representación espacial de datos obtenidos en una determinada área productiva, contribuyendo de manera significativa a la toma de decisiones y la gestión óptima de insumos, mitigando o solucionando cuestiones no solo de tipo económico, sino también ambiental y social.

Adquisición remota

Para poder llevar a cabo la agricultura de precisión es necesario un mapeo de las variables de tipo espacial y temporal en las unidades de producción. Una de las maneras más eficientes de obtener esta información es de manera remota, permitiendo obtener

información sobre un proceso a través de sensores (dependiendo del sistema y tipo de información, puede estar a varios centímetros o kilómetros de la fuente). Dentro de la adquisición remota podemos diferenciar la adquisición de datos y el análisis de datos. (Fernández and others 2012)

Tecnologías de dosis variable. Georreferenciación

Típicamente, en la gestión de un cultivo se han considerado siempre valores promediados de insumos requeridos, rendimiento y fertilidad, extrapolando los valores y considerando a un mismo lote de manera homogénea.

No obstante, actualmente dentro del sector agrícola, se conoce perfectamente que dentro del mismo lote de cultivo, la cantidad de insumos requeridos, así como propiedades del suelo y rendimiento del cultivo es heterogéneo. Actualmente, encontramos en el mercado todo tipo de controladores que monitorean y regulan las dosis requeridas de insumos en cada metro cuadrado. (Ortega et al. 1999b)

Los avances en la tecnología DGPS (Differential Global Positioning System) permite elaborar el mapeado de datos con una precisión de hasta 10 cm. Una vez obtenidos los datos de las variables georreferenciadas, se procede a su análisis y elaboración de respuestas.

2.2 Evapotranspiración. Balance hídrico

Evapotranspiración

La evapotranspiración (ET) es el proceso de pérdida de agua en un cultivo, combinación de evaporización del agua a través del suelo, y transpiración a través de los tejidos de las plantas. (Allen and ONU para la Agricultura y la Alimentación 2006).

En una planta, la mayor parte del agua absorbida por las raíces se transforma en vapor de agua dentro de la propia planta, y se regula su salida a la atmosfera a través de los estomas (pequeñas aberturas en las hojas). Esto es lo que denominamos transpiración (T). Sólo una pequeña porción del agua absorbida pasará a formar parte de los tejidos de la planta, la mayoría se liberará a la atmosfera mediante la transpiración.

En el siguiente esquema, se puede apreciar claramente cómo evoluciona la evapotranspiración durante la vida del cultivo (Figura 2.4). La componente transpiración, al comienzo es prácticamente inexistente, contándose más tarde en la principal causante de la ET. Esto es debido a que, a medida que aumenta la cosecha, el mayor número de hojas y el tamaño de los cultivos cubre de manera más sustancial el suelo, protegiéndola de la radiación solar y por tanto se reduce de manera muy importante la evaporación debido a la componente del suelo.

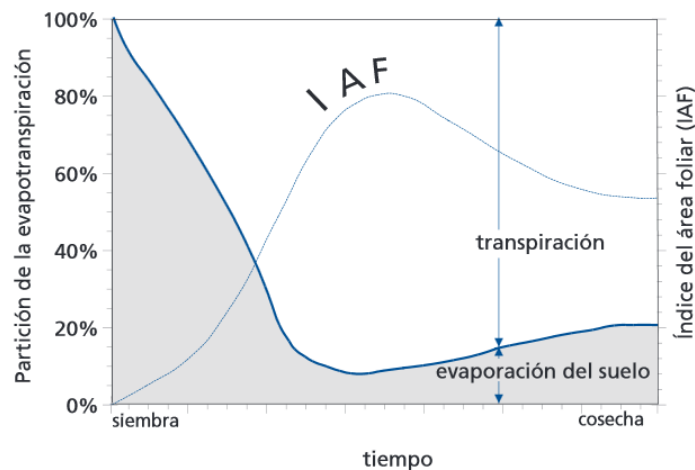


Figura 2.4: Evapotranspiración durante el crecimiento del cultivo (Allen and ONU para la Agricultura y la Alimentación 2006)

Los principales factores que afectan a la evapotranspiración por tanto, podemos clasificarlos en:

- a) Climáticos: presión atmosférica, intensidad y duración de la radiación solar, condiciones de viento, humedad.
- b) Suelo: textura, espacio entre poros, estructura.
- c) Vegetales: tipo de planta, forma, color, estado de crecimiento.

Balance hídrico

La evapotranspiración se puede estimar a partir de los elementos que componen el balance hídrico. El balance hídrico proviene del concepto de balance de materia, es decir, la suma de todos los elementos que entran a un sistema serán igual a la suma de lo que sale de él y permanece. Basándose en esto, para estudiar el balance hídrico en un cultivo se procede de la misma manera.

Todo cultivo de regadío requiere de un riego para su correcto crecimiento; la cantidad de agua aportada puede determinarse fácilmente mediante un caudalímetro o mediante volúmenes de agua preestablecidos.

La lluvia, junto con el riego, conforman los sistemas de entrada al sistema. La lluvia al contrario que el riego es una variable no controlable, y cuya medición se lleva a cabo mediante estaciones meteorológicas locales.

El agua que permanece dentro del sistema será aquella que pase a formar parte de la planta en sí, ya sea en sus tejidos, cuerpo u hojas (aunque muy poca agua se queda en la planta, pues casi toda el agua absorbida por la planta es transferida a la atmosfera mediante la transpiración) así como la contenida en la tierra.

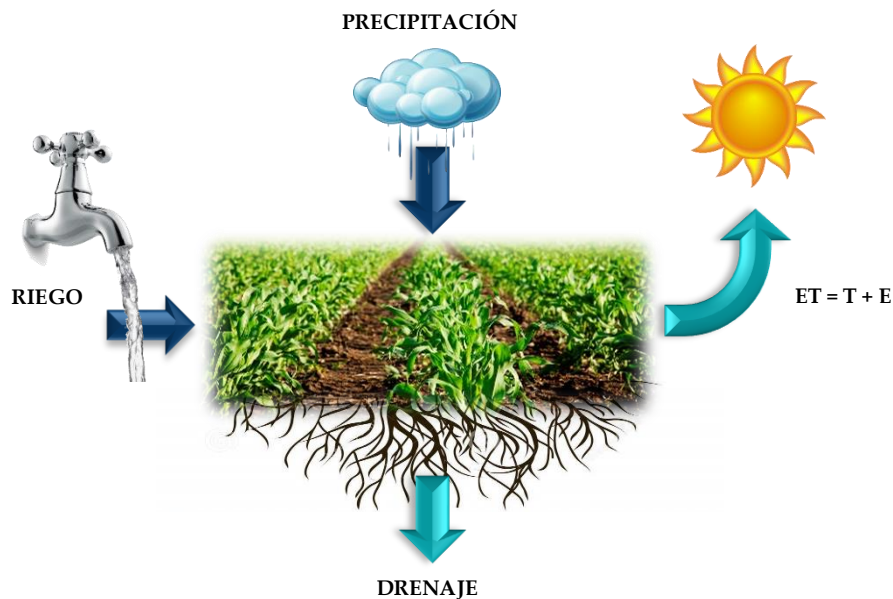


Figura 2.5: Balance hídrico del cultivo

Como salidas del sistema tenemos el agua de drenaje y la evapotranspiración. El concepto de agua de drenaje es aquella que, tras empapar la tierra por completo y superar la capacidad de campo, lixivias y discurre por la tierra, perdiéndose.

La ecuación del balance hídrico del suelo, por tanto, es:

$$P + R = ET + D \pm \Delta w$$

Donde P representa la cantidad de agua precipitada, R la regada, ET la evapotranspiración, D el agua drenada y Δw la variación de cantidad de agua contenida en la tierra durante un determinado tiempo.

La evapotranspiración es la pérdida de agua mediante evaporización y transpiración. Por tanto, la evapotranspiración supone un elemento importante de nuestro balance hídrico, siendo ésta difícil de estimar mediante métodos tradicionales. Este balance hídrico en el suelo se suele utilizar para estimar la evapotranspiración en periodos semanales o superiores (Allen and Organización de las Naciones Unidas para la Agricultura y la Alimentación 2006).

2.3 Introducción a la lisimetría.

Introducción

El concepto de lisímetro se atribuye a aquel dispositivo cuyo objetivo es determinar la cantidad, velocidad y composición del agua percolada. La lisimetría comenzó a utilizarse hace aproximadamente 300 años. Un matemático francés en el año 1688 comenzó lo que serían las primeras investigaciones en el campo de la lisimetría. Utilizando recipientes de plomo, concluyó que se evaporaba menos agua en los recipientes con arena solo que aquellos cubiertos de césped (Santa Olalla Mañas, López Fuster and Calera Belmonte, 2000).

Ya en 1937 se instalaron en Estados Unidos los primeros lisímetros equipados con sistemas electrónicos para registrar las precipitaciones, agua de drenaje y evapotranspiración, permitiendo realizar balances hídricos. A partir del siglo XX los lisímetros se perfeccionan y comienzan a instalarse en centros de investigación a lo largo de todo el mundo, cuya función principal se convirtió en estudiar la evapotranspiración de los cultivos, y determinar modelos que permitiesen estimar y ajustar las ecuaciones empíricas y semiempíricas (Santa Olalla Mañas, López Fuster and Calera Belmonte, 2000).

Además, los lisímetros se han utilizado para determinar otros conceptos como la evapotranspiración de referencia (ET_o) evapotranspiración de cultivo (ET_c), lixiviación mínima de sales, y la precipitación efectiva.

Podríamos definir lisímetro es aquel dispositivo que contiene un volumen de suelo con una muestra del cultivo a estudiar (por tanto aislado del resto de terreno) que recibe agua de riego y/o de lluvia pretendiendo representar las condiciones naturales y permite evaluar mediante sensores el flujo de agua y, por tanto, aquellos parámetros que intervienen en el balance hídrico.

La base de la lisimetría es el cálculo o medida volumétrica de los flujos de agua entrante y saliente de una masa de suelo aislada en la que se encuentra nuestro cultivo muestra, por lo que utiliza la ecuación del balance hídrico anteriormente mencionada (Santa Olalla Mañas, López Fuster and Calera Belmonte, 2000).

Tipos de lisímetro

A la hora de clasificar los lisímetros encontramos dos grandes criterios (Santa Olalla Mañas, López Fuster and Calera Belmonte, 2000):

a) En función del sistema que utilice para determinar los flujos de agua:

- **Lisímetros no pesantes o de drenaje.**



Figura 2.6: Lisímetro de drenaje

En este tipo de lisímetros, la cantidad de evapotranspiración resulta de la diferencia entre el agua aplicada y el agua drenada. La lluvia y riegos se determinan con pluviómetros, y el agua drenada se recoge al fondo del lisímetro y se mide volumétricamente.

- **Lisímetros de pesada.**



Figura 2.7: Lisímetro de pesada

En esta categoría, el suelo muestra se encuentra en un recipiente que tiene libertad de movimiento vertical, y reposa sobre una balanza mecánica o electrónica que registra la variación de peso en el lisímetro debido a evapotranspiración, riego o precipitación.

Normalmente, los lisímetros de pesada constan de una arqueta subterránea donde se encuentra un equipo de pesada (suelen tener capacidad de 25 toneladas y precisión de hasta 250 gramos), y sobre la cual reposa un gran recipiente con una muestra de suelo. La parte superior del suelo se encuentra a ras de tierra y tiene muestras del cultivo a evaluar.

El lisímetro se instala en el centro de una parcela cuadrada de al menos 1 hectárea, de manera que la tierra que rodea al lisímetro se encuentra sembrada con el mismo cultivo que el lisímetro, permitiendo simular de la manera más exacta posible las condiciones del cultivo.

b) En función del método utilizado para aislar el volumen de tierra a estudiar con el resto de suelo:

- **Lisímetros monolíticos** (suelo sin perturbar)

En este tipo de lisímetros, se limita el contenido de suelo ya existente con unas paredes, encerrándolo y aislándolo del resto de suelo, y sobre su base se instala el sistema de medición de agua drenada o percolada. Debido a su elevado coste, suelen utilizarse para determinar el agua de escorrentía, infiltración y drenaje.

- **Lisímetros de relleno** (suelo perturbado)

Al alterarse el perfil de suelo, se debe intentar en la mayoría de lo posible conservar las condiciones y características del suelo, ya que se producen cambios físicos como flujo de calor y movimiento de agua irregular. Esto provoca que muchas veces se generen crecimientos de cultivo desigual entre las plantas situadas dentro del lisímetro y las del exterior.

Este tipo de lisímetro suele utilizarse para determinar la necesidad hídrica de agua en un cultivo (Santa Olalla Mañas, López Fuster and Calera Belmonte, 2000).

- **Lisímetros tipo Ebermayer** (caso intermedio)

Este tipo de lisímetro no tiene paredes laterales, lo que permite el flujo libre lateral de agua. Se considera un tipo especial de lisímetro monolítico, y consta de un embudo y un colector situados bajo la muestra de suelo a estudiar, en este caso sin alterar.

Lisímetro de pesada en maceta

Una vez vistos los tipos más comunes de lisímetro existentes, procedemos a mostrar el lisímetro utilizado en nuestro trabajo fin de grado. El lisímetro de pesada en maceta (Figura 2.8), como se comentó en la introducción, es el resultado de la tesis doctoral de Leandro Ruiz Peñalver (Ruiz-Peñalver et al. 2015). Al no necesitar de obra civil, este lisímetro reduce enormemente la inversión necesaria.



Figura 2.8: Lisímetro de pesada en maceta: Prototipo (Izq), Lisímetro utilizado (Der)

En este caso, la tierra de cultivo está contenida dentro de una maceta, eliminando los movimientos laterales de agua. A su vez, al no estar rodeado de cultivo ni suelo de características similares, es necesario intentar emular lo mejor posible estas condiciones. Ésta, descansa sobre una superficie triangular que cuenta con 3 células de carga de 30 Kg cada una, permitiendo utilizar muestras de cultivo de hasta 90 Kg de rango teórico. Estas células de carga serán las responsables de determinar la evolución del peso de la maceta y su contenido en agua.

Una vez realizado el riego y empapado la tierra (capacidad de campo) se recoge el agua percolada en su parte inferior mediante una bola de drenaje. Esta bola cuenta con su propia célula de carga de rango 10 Kg, y su capacidad es de aproximadamente 4 litros.

El lisímetro de pesada en maceta es un prototipo de bajo coste, que ya ha sido validado en trabajos anteriores por grupos de investigación de la Universidad Politécnica de Cartagena (Ruiz-Peñalver et al. 2015).

2.4 Plataforma Open-source. Arduino board

Introducción

En este apartado se hablará de la nueva tendencia Open-source y una de sus plataformas más populares, Arduino, ya que es la que se ha decidido utilizar durante el desarrollo del proyecto (debido a su reducido coste y gran versatilidad).

El término “open source” se atribuye a aquello que puede modificarse y compartirse debido a que su diseño es de uso público, en el que podemos diferenciar el hardware y software libre.

El software libre es aquel software cuyo código fuente es accesible, y puede modificarse y mejorarse. Algunos programas software tienen un código fuente al que solo las personas con privilegios pueden acceder y modificarlo, como la persona, grupo u organización que lo creó. Se llama a este tipo de software “cerrado” o “closed source”. Solo los propietarios o creadores pueden legalmente acceder, copiar y alterar este software. Al tener permisos de utilizar este tipo de software, los usuarios normalmente tienen que firmar una licencia en la que aseguran que los usuarios no harán nada con el software que los autores no hayan permitido expresamente. Como ejemplo tenemos el famoso Microsoft Office o Adobe Photoshop.

El software libre es distinto, los autores hacen accesible el código fuente para aquellos que quieran ver el código, copiarlo, alterarlo, compartirlo y en general aprender de él. Como ejemplos tenemos el LibreOffice. Al igual que con el “closed source”, los usuarios también deben aceptar los términos de la licencia cuando quieren hacer uso de este software “open source”, aunque estos términos difieren bastante de los anteriores. Por lo tanto, el “open source” software promueven compartir y colaborar ya que permiten a la gente modificar el código fuente e incorporarlo a sus propios proyectos. (Opensource.com)

El hardware libre u “open hardware” se refiere a aquellos diseños de hardware que tienen permiso de ser estudiados, modificados y distribuidos por cualquiera. El término se puede aplicar a todo tipo de objetos, desde coches, ordenadores y robots hasta casas. El código fuente debería ser accesible, y los componentes utilizados suelen ser comunes y fáciles de obtener para cualquier persona.

Al igual que en el software libre, el código fuente” del “open hardware” (esquemáticos, diseños de CAD,...) está disponibles para su modificación y mejora siempre que se encuentre bajo una licencia que así lo permita. Por tanto, también está sujeto a copyright y patentes.

Entre las ventajas del software y hardware “open source” se pueden encontrar:

- a) Permite el control absoluto sobre código fuente y, en definitiva, examinar el código y diseño para cerciorarse de que éste va a hacer exactamente lo que ellos quieren que haga, y en caso contrario modificarlo.
- b) Hay gente que lo considera más seguro, ya que cualquier persona puede verificar y modificar este código y/o diseño, y por tanto corregir los errores que el autor original haya cometido.
- c) Estabilidad a largo plazo, ya que el código abierto está en constante cambio y mejora. El software tradicional puede acabar quedando obsoleto y con errores sin solventar si sus creadores dejan de trabajar en ellos.
- d) En general, sirve de entrenamiento para convertirse en mejor programador. Al tener un código fuente público, los estudiantes y aquellas personas en proceso de aprendizaje pueden fácilmente acceder a él para utilizarlo. Además, se crea un ciber-comunidad en la que se ayuda, invitando a la crítica constructiva, y se fomenta al resto de internautas a seguir trabajando y a compartir los progresos. Muchas veces nos encontramos un problema al que posiblemente otra persona ya se haya enfrentado y solventado.

A pesar de todo esto, las plataformas “open source” no implican que sean gratis. Los programadores de este tipo de plataformas pueden cobrar dinero por sus programas, códigos o diseños que han creado o en los que hayan participado, o al brindar un soporte técnico. (Opensource.com)

Arduino board

El proyecto de Arduino surgió en Italia. Un estudiante colombiano llamado Hernando Barragán creó la plataforma de desarrollo Wiring en 2004 como resultado de su tesis. El objetivo era crear una herramienta de bajo coste para permitir a aquellas personas sin conocimientos técnicos, desarrollar sus propios proyectos digitales. La plataforma Wiring consistía en una placa PCB con un microcontrolador ATmega 128, un entorno de desarrollo integrado (o IDE) basado en Processing con librerías para facilitar el uso del microcontrolador.

Actualmente Arduino es una empresa de software y hardware, proyecto, y una comunidad de usuarios, que diseñan y manufacturan hardware y software “open source” basados en microcontroladores, así como kits para construir dispositivos que monitorizan y controlan procesos u otros dispositivos físicos.



Figura 2.9: Placa PCB de Arduino

La placa Arduino hasta hace poco (ahora se usan más microcontroladores) utilizaba un microcontrolador AVR Atmel de 8, 16 o 32 bits, con una serie de componentes complementarios para hacer más accesible y fácil su programación, así como su integración e incorporación en otros circuitos. Lo bueno de Arduino, es que utiliza un sistema estandarizado de pines y conexionado que permite utilizar de manera muy sencilla un amplio rango de módulos intercambiables o shields. Tecnología más reciente permitió comunicar algunos de estos shields mediante el bus serial I2C, lo que permite utilizar paralelamente varios de estos módulos.

El proyecto está basado en varios diseños de placas, producidos por varios fabricantes, utilizando diferentes microcontroladores. Antes de 2015, los Arduinos oficiales utilizaban los encapsulados megaAVR de Atmel, pero actualmente se utiliza un rango mucho más amplio.

La placa (Figura 2.9) cuenta con 14 pines digitales que pueden programarse como entrada / salida y 6 entradas analógicas, que permiten al usuario interactuar con sensores, actuadores, otros circuitos u otras placas de expansión (shields). Varios de estos pines permiten generar pulsos de ancho modulado o PWM. La mayoría de las placas incluyen un regulador de voltaje de 5V así como un oscilador de 16MHz. Incluye un puerto serial para su comunicación con periféricos, además de conexión USB (permite volcar de manera muy sencilla códigos desde nuestro ordenador).

Actualmente en el mercado tenemos una amplia variedad de modelos, pero las placas más comúnmente utilizadas son:

- Arduino Uno
- Arduino Due
- Arduino Mega2560
- Arduino Leonardo.

En la Tabla 1 podemos observar una comparativa de sus características principales entre los modelos más populares:

| Modelo | Arduino UNO | Arduino Leonardo | Arduino Mega 2560 | Arduino DUE |
|--|-----------------------------|----------------------------------|-----------------------------------|-----------------------------|
| Microcontroller | ATmega328 | ATmega32u4 | ATmega2560 | AT91SAM3X8E |
| Operating Voltage | 5V | 5V | 5V | 3.3V |
| Input Voltage | 7-12V | 7-12V | 7-12V | 7-12V |
| Input Voltage (limits) | 6-20V | 6-20V | 6-20V | 6-20V |
| Digital I/O Pins | 14 | 20 | 54 | 54 |
| Digital I/O Pins PWM output | 6 | 7 | 15 | 12 |
| Analog Input Pins | 6 | 12 | 16 | 12 |
| Analog Outputs Pins | | | | 2 (DAC) |
| Total DC Output Current on all I/O lines | 40 mA | 40 mA | 40 mA | 130 mA |
| DC Current for 3.3V Pin | 50 mA | 50 mA | 50 mA | 800 mA |

| Modelo | Arduino UNO | Arduino Leonardo | Arduino Mega 2560 | Arduino DUE |
|-------------------------------|------------------------------------|----------------------------------|-----------------------------------|------------------------------------|
| DC Current for 5V Pin | | | | 800 mA |
| Flash Memory | 32 KB 0.5 KB used by bootloader | 32 KB 4 KB used by bootloader | 256 KB 8 KB used by bootloader | 512 KB available |
| SRAM | 2 KB (ATmega328) | 2.5 KB | 8 KB | 96 KB two banks: 64KB y 32KB |
| EEPROM | 1 KB (ATmega328) | 1 KB | 4 KB | |
| Clock Speed | 16 MHz | 16 MHz | 16 MHz | 84 MHz |
| Tipo de USB | Estandar | Mini | Estándar | Mini |
| ~Precio Local US\$ (Sin imp.) | \$33.00 | \$33.00 | \$64.00 | \$64.00 |

Tabla 1: Comparativa entre modelos de Arduino

Voltaje: Las placas Arduino más comunes trabajan todas con 5V; no obstante, los pines del Arduino DUE trabajan con 3.3V, lo que hace más difícil encontrar sensores y actuadores compatibles.

Memoria: El Arduino Uno y el Arduino Leonardo sólo disponen de 32k de memoria disponible, frente a los 256k del Arduino Mega2560 y los 512k del Arduino DUE. Muchas veces, debido a la complejidad y peso de algunas librerías, es más importante una buena capacidad de memoria que la velocidad de procesamiento.

Velocidad: El Arduino DUE trabaja con un oscilador 5 veces más rápido que sus compañeros, 84MHz frente a 16MHz.

Pines: Los modelos Arduino Mega2560 y DUE cuentan con hasta 54 pines digitales, frente a los 14 y 20 de los otros dos. En función de la complejidad del proyecto, este punto será determinante si por ejemplo trabajamos con servos y/o motores y necesitamos muchos pines modulados PWM.

Salidas analógicas: El Arduino DUE permite reproducir sonidos de buena calidad mediante dos salidas analógicas o DAC.

Para programar el microcontrolador, Arduino tiene su propio entorno de desarrollo integrado (IDE) que puede descargarse de forma gratuita desde la página oficial. Este software libre se escribe normalmente en Wiring, que está basado en Processing, y soporta C y C++.

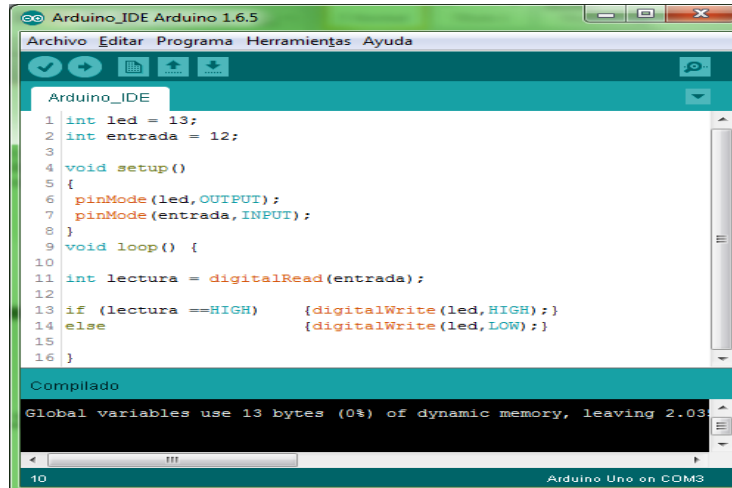


Figura 2.10: Entorno de desarrollo Integrado (IDE)

Este software permite escribir el código en un entorno muy sencillo de usar, que trae herramientas para dicha función, compilando y volcando los códigos de manera cómoda. Cualquier código escrito en este IDE recibe el nombre de “sketch”. Los programas son cíclicos y constan de dos grandes partes:

- El “setup()”, que se utiliza para inicializar variables y, en general, para configurar los parámetros al iniciar el programa, ya que éste solo correrá una vez al inicio del programa
- El “loop()”, que es el cuerpo del programa, aquella parte de código que se repite en bucle hasta que el dispositivo se desconecta de la fuente de alimentación.

Arduino puede programarse de igual forma en otros lenguajes de programación mediante compiladores que generen código máquina. Aunque los diseños de hardware software son libres, los desarrolladores han pedido que el nombre “Arduino” sea exclusivo del producto oficial y que no sea usado para otros diseños sin permiso.

El futuro de Arduino es claro: va a seguir extendiéndose su uso, ya que es un método muy sencillo de hacer llegar la tecnología a las aulas y a aquellas personas que quieran asomarse al mundo de la programación y la electrónica. Además, Windows 10 será el primer sistema operativo compatible con Arduino. El fabricante del microcontrolador expone “Windows 10 permite que se puedan crear objetos inteligentes combinando las capacidades hardware de Arduino con las capacidades software de Windows”. (González 2015)



Capítulo 3.

Descripción del Hardware

En este tercer capítulo se detalla el hardware utilizado para la realización del trabajo, así como sus especificaciones técnicas. El capítulo se ha dividido en cuatro apartados en función de la naturaleza del dispositivo: Sensores, Actuadores, Dispositivos de control y Shields.

3.1 Sensores

Un sensor es aquel dispositivo que detecta un tipo de señal física de su entorno y reacciona ante ésta. Esta entrada de información puede ser de muchos tipos, como humedad, temperatura, velocidad, presión, y en general cualquier fenómeno físico que deseemos cuantificar. La salida del sensor suele ser generalmente una señal analógica que necesita ser transformada a un lenguaje entendible para nosotros, o bien por el mismo dispositivo a bien es transmitido y almacenado para su posterior procesamiento.

3.1.1 Células de carga

Una célula de carga (Figura 3.1) es un transductor que transforma una fuerza aplicada sobre ella (deformación física) en una señal eléctrica que puede medirse fácilmente y es proporcional a esa fuerza. Las células de carga tienen un uso muy extendido en el sector industrial, ya que su amplio rango de trabajo permite obtener medidas de peso / presión con gran precisión bajo cargas de diferente naturaleza.

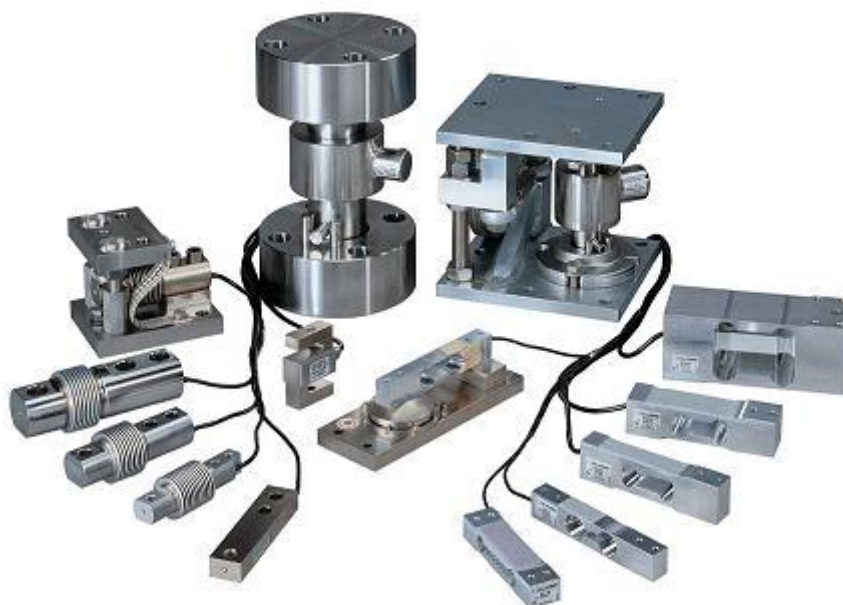


Figura 3.1: Células de carga

Existe un amplio rango de células de carga que, en función de su aplicación industrial, tienen desde una capacidad de 1 gr (química, sanitaria) hasta 5.000 toneladas.

Los tipos de célula de carga se pueden clasificar principalmente en:

- Tipo de señal de salida: Si la célula genera una señal de tipo neumático, eléctrico o hidráulico, de manera que sea cuantificable
- El tipo de entrada: Hay muchos tipos de célula en el mercado, y cada una está estudiada para ser sometida a un tipo de fuerza, flexión, tensión, compresión o cizalladura.

Las más comunes son las de tipo eléctrico. Está formado por unas **galgas extensiométricas**, cuyo principio de funcionamiento se basa en la variación de resistividad con la deformación. Se componen de un fino cable (normalmente aleación cobre-níquel) de pequeño diámetro en forma de zigzag, formando de esta manera un plano.

Esta galga se sitúa sobre la superficie de un objeto en el que pretendemos medir un esfuerzo, en una o varias de sus caras. Al aplicar una fuerza sobre el cuerpo, la cara donde se encuentra la galga varía de tamaño; se encoge si esa cara se encuentra a compresión (Galga 1, Figura 3.2), y se estira si se encuentra a tracción (Galga 2, Figura 3.2).

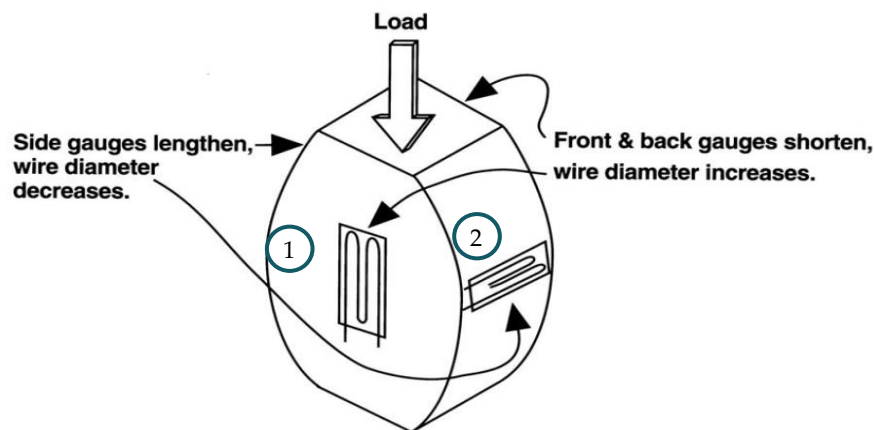


Figura 3.2: Comportamiento de las Galgas extensiométricas. *Load Cell Handbook 2007, Rice Lake Weighing Systems*

Cuando la galga aumenta su longitud, disminuye su sección y por tanto su resistencia. Donde la resistividad ρ del material es constante. De la misma manera ocurre cuando disminuye su longitud, aumenta su sección y disminuye su resistencia.

La estructura interna de las células de carga está compuesta por una o varias galgas extensiométricas en forma de **punto de Wheatstone**. Hay diferentes configuraciones, pueden verse en la Figura 3.3. En la primera imagen tenemos una sola galga sometida a tracción en una cara. En la segunda, 2 galgas, una a tracción y otra a compresión, en una cara. Y en la tercera, 2 galgas, tracción y compresión, en ambas caras.

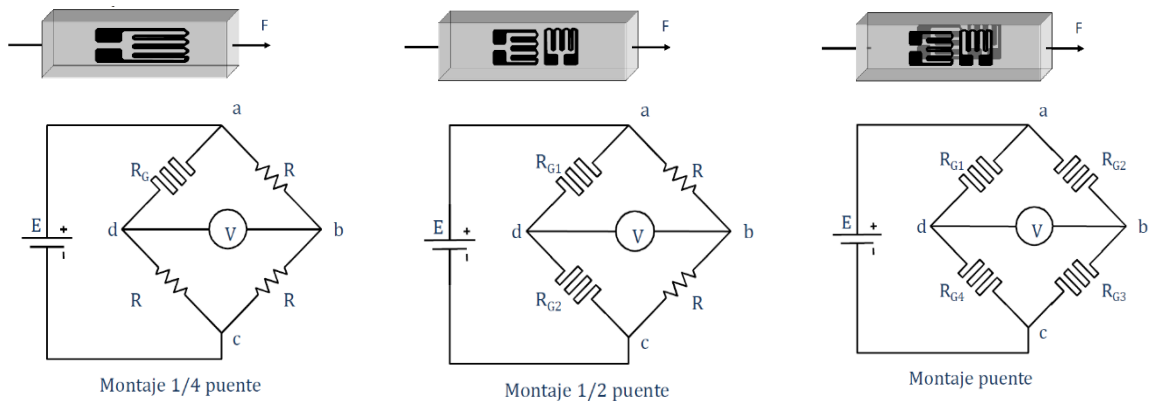


Figura 3.3: Diferentes tipos de puente Wheatstone. Apuntes de Sistemas de Instrumentación Electrónica (Upct). Joaquín Roca González

Conociendo la tensión de la fuente de alimentación (E), podemos cuantificar el valor de la fuerza (F) sabiendo que la tensión de salida (V) es proporcional al valor resistivo de las resistencias (R y R_{Gx}).

Antes de su descubrimiento, para el pesaje industrial se utilizaban las básculas mecánicas, que permiten pesar con gran precisión desde fármacos (pequeñas pastillas) hasta un autobús, siempre que esté debidamente calibrada.

Células de carga Vetek 108TA

En nuestro proyecto, las células de carga que vamos a utilizar son de puente completo, modelo Vetek 108TA. Usaremos tres células para medir el peso del lisímetro, que se encontraran bajo la estructura, y una de rango menor para la bola de drenaje, encargada de monitorizar el agua lixiviada.

Las especificaciones técnicas de este modelo, para los dos rangos de peso son los siguientes:

- Estructura (3 x 30Kg):
 - Rango máximo de 30 Kg
 - Precisión de 6 gr.
 - Sensibilidad de $2\pm 10\%$ mV/V
 - Resistencia de salida de $350\ \Omega$
 - Alimentación de 0.5 ~ 12V
 - Carga máxima de seguridad de 45 Kg
 - Carga de rotura de 90 Kg

- Bola de drenaje (1 x 10Kg):
 - Rango máximo de 10 Kg
 - Precisión de 2 gr.
 - Sensibilidad de $2\pm 10\%$ mV/V
 - Resistencia de salida de $350\ \Omega$
 - Alimentación de 0.5 ~ 12V
 - Carga máxima de seguridad de 15 Kg
 - Carga de rotura de 30 Kg



Figura 1.4: Célula de carga Vetek 108TA

A partir de la sensibilidad y el voltaje de alimentación, podemos obtener la salida máxima de la célula. Suponiendo sensibilidad de 2 mV/V y alimentando a 5 V, obtendremos una salida máxima de 10 mV para la célula de rango 30 Kg, y salida máxima de 3.33 mV para la de rango 10 Kg.

La normativa OIML R60 es la que se encarga de regular y certificar las células de carga en Europa, es el marco normativo. Y en función de su precisión, se pueden clasificar en cuatro clases: A, B, C y D. En nuestro caso, C3 supone que el rango de medición de la celda de carga puede dividirse en máximo 5000 divisiones de verificación. Por lo tanto, nuestra célula de 30 Kg tendrá una precisión de $30 / 5000 = 6$ gramos, y la célula de 10 Kg una precisión de $10 / 5000 = 2$ gramos.

Aquí se observa la tabla con los aspectos técnicos generales del modelo:

| Type | | 108TA |
|---|------------------------|--|
| Accuracy class according to OIML R60 | | C3 |
| Maximal numbers of load cell verification intervals (n _v) | | 5 000 |
| Maximal capacity (E _{max}) | kg | 5;7;10;15;20;30;50;75;100;150 |
| Minimum load cell verification interval (V _{min}) | % of C _n | 0.02 |
| Reference Max. platform size | mm | 400x450 |
| Weight (G), approx. | kg | 0.33 |
| Sensitivity (C _n) | mV/V | 2±10% |
| Zero balance | mV/V | ±0.06 |
| Temperature effect on sensitivity (TK _c) * | % of C _n /K | < ±0.0012 |
| Temperature effect on zero balance (TK _o) | % of C _n /K | < ±0.0040 |
| Non-linearity (d _{lin}) * | % | < ±0.017 |
| Repeatability (d _{rep}) * | % | < ±0.017 |
| Hysteresis error (d _{hy}) * | % | < ±0.017 |
| Creep (d _{DR}) in 30 min. | % | < ±0.023 |
| Input resistance (RLC) [Red(+)-black(-)] | Ω | 415 ±15 |
| Output resistance (RO) [Green(+)-white(-)] | Ω | 350±3 |
| Reference excitation voltage (U _{ref}) | V(DC/AC) | 0.5...12 |
| Maximal excitation voltage | V(DC/AC) | 18 |
| Insulation resistance (R _{is}) | GΩ | >1 050 VDC □ |
| Nominal temperature range | □□°F□ | -10...+40□15...+105□ |
| Service temperature range | □□°F□ | -10...+50□15...+122□ |
| Storage temperature range | □□°F□ | -25...+70□-13...+158□ |
| Safe load limit (EL) | % of C _n | 150 |
| Breaking load (Ed) | % of C _n | 300 |
| Protection class (IP) acc. to IEC529 | | IP66 |
| Material | | Aluminum |
| Optionally | | Explosion proof version (Ex ib II CT4) |

Tabla 2: Hoja de características, Célula de carga Vetek 108TA

Como se puede ver en la hoja de características, la manguera contiene 4 cables:

- Rojo: Alimentación (Positiva).
- Negro: Alimentación (Negativa).
- Verde: Señal de salida (Positiva).
- Blanco: Señal de salida (Negativa).

Para obtener los valores de peso de manera precisa, algunos parámetros de la célula de carga vienen calibrados, y provistos de un certificado individual para cada célula de carga. En la Figura 3.5 podemos ver un ejemplo:

LOAD CELL CALIBRATION CERTIFICATE

Model: 108TA Max.Cap. (Emax): 30kg
 QA: 004 S/N : 1211015855

| NTEP 1:5000 Class III Multiple Cell | | Certificate Number: 07-024 | |
|-------------------------------------|--------|----------------------------|----------|
| Vmin | kg | 0.003 | |
| Output sensitivity | mV/V | 2.02 | |
| Combined Error | %FS | <0.03 | |
| Zero balance | μV | 36 | |
| Input resistance | Ω | 403 | |
| Output resistance | Ω | 350 | |
| Insulation resistance | MΩ | ≥5000[50VDC] | |
| Safe overload | %FS | 150 | |
| Compensated Temp.Range | °C | -10~+40 | |
| Recommended supply voltage | V | 5~12(DC/AC) | |
| 4-Core shielded cable <u>2</u> m | Input | +: Red | -: Black |
| | Output | +: Green | -: White |
| | Shield | Bare | |

Date: 2012-5-22



Figura 3.5: Certificado de calibración

Rectas de calibración

Las células de carga de la estructura (rango 30Kg) soportarán como máximo estimado, 20 Kg cada una, y la de la bola de drenaje (rango 10Kg) aproximadamente 4 Kg (es lo que pesa la bola con máximo volumen de agua admisible). Por tanto para calibrar las células, se ha dividido el rango de trabajo de la misma en 7 puntos.

30 Kg: Sin plato (0 Kg), Con plato de pesada (240,05 gr), 5, 10, 15, 20 y 25 Kg.

10 Kg: Sin plato (0 Kg), Con plato de pesada (240,05 gr), 0.5, 1.5, 2.5, 3 y 5 Kg.

Para poder calibrar las células de manera precisa, se diseñó un programa de **LabView** (Apartado 5.3 - Otras aplicaciones). Las rectas de calibración se han obtenido a partir de hasta 1000 puntos, de manera que aumenta su precisión y nuestro R^2 es 1 un todos los casos.

Durante el proyecto, para facilitar su manejo, se ha nombrado de la siguiente manera a las células de carga:

- Célula 1 (30 Kg). Número de serie: 121101-5836
- Célula 2 (30 Kg). Número de serie: 121101-6578
- Célula 3 (30 Kg). Número de serie: 121101-5868
- Célula 4 (10 Kg). Número de serie: 10605075

A continuación se muestran las 4 calibraciones resultantes:

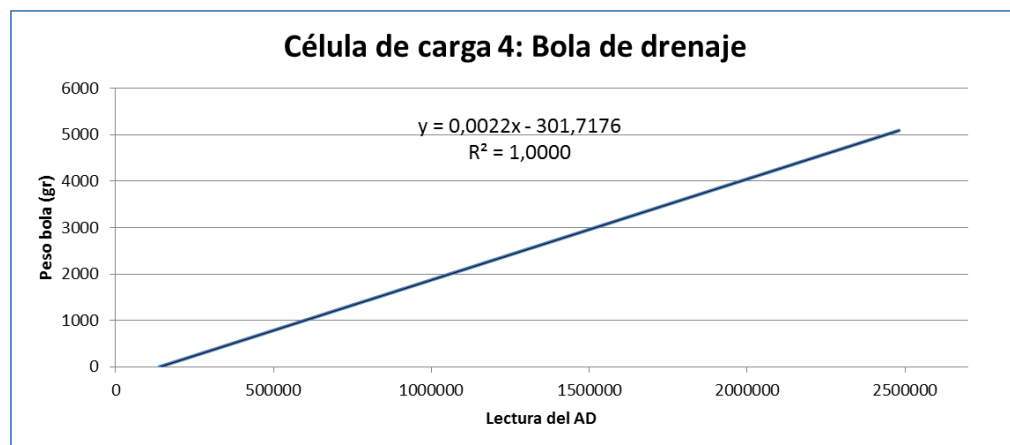


Figura 3.6: Recta de calibración de la célula 4. Bola de drenaje

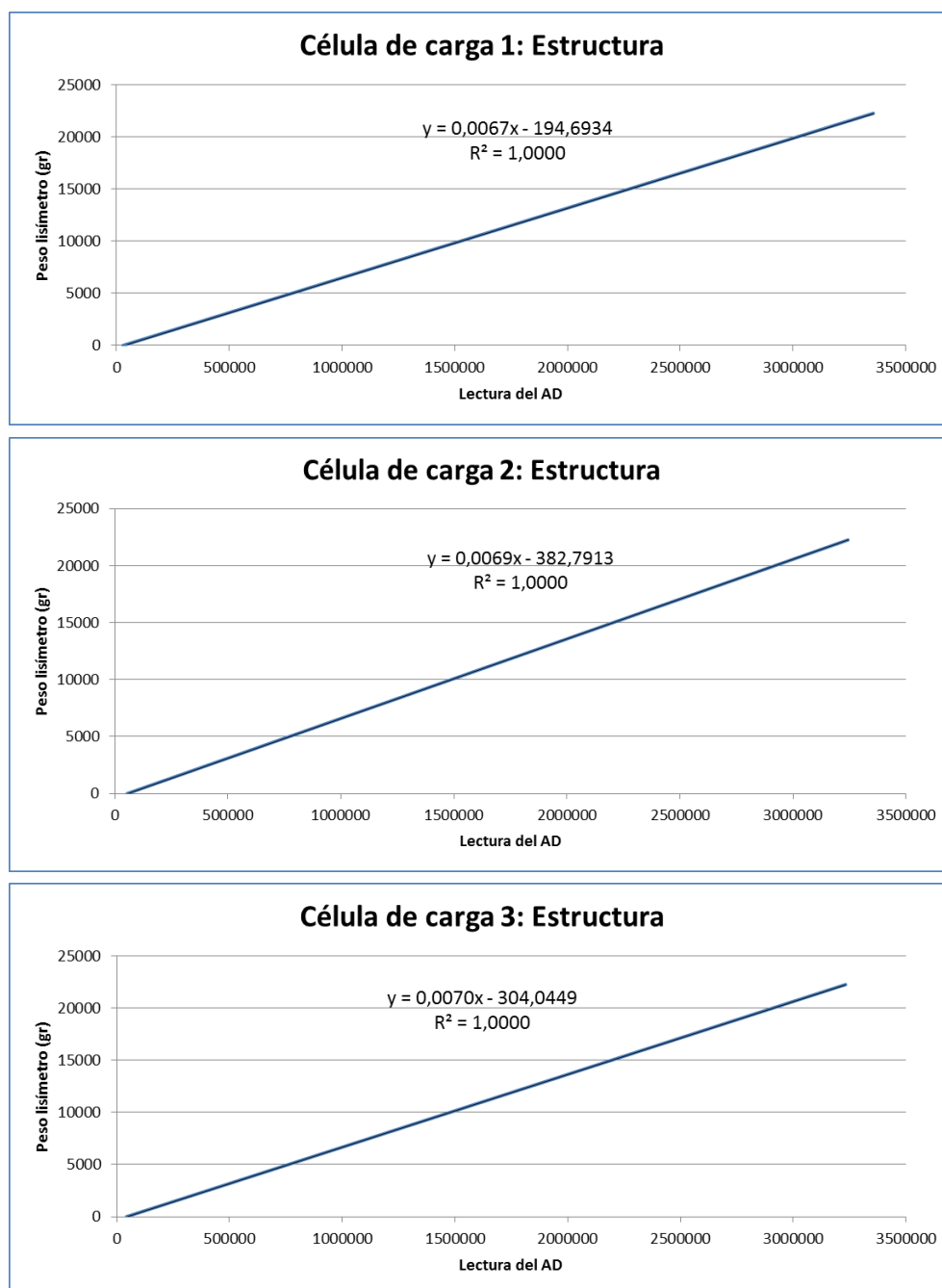


Figura 3.7: Rectas de calibración de las células 1, 2 y 3 que conforman la estructura del lisímetro

3.1.2 Caudalímetro de precisión

Para medir y registrar la cantidad de agua de riego, se necesita disponer de un caudalímetro. Es importante saber con precisión qué cantidad de agua estamos regando para determinar con mayor exactitud la capacidad de campo y evapotranspiración del cultivo. Un caudalímetro es un sensor usado para determinar la cantidad de líquido o gas circulante por un conducto. Los caudalímetros de precisión se utilizan para controlar y monitorear la cantidad de caudal exacto.

Se ha usado el caudalímetro de precisión **PVDF Flowmeter 0045** de la empresa Equflow sensors (Figura 3.8). Es de un rango tan bajo puesto que el riego se realiza por goteo; el rango de caudal es de 0,03–2 Litros/minuto. En el lisímetro se han usado goteros de riego autocompensados, de caudal 2 Litros/ hora, que equivale a 0,033 litros/minuto, por lo que nos encontramos dentro del rango. Al realizarse el riego por goteo, se necesita de gran precisión por lo que se ha usado este modelo.



| | |
|---------------------------------|----------------------|
| Model | 0045 |
| Inner diameter in mm | 4.7 |
| Flow range | 0.03 – 2 L/min |
| Accuracy | 1% of reading |
| Repeatability | < 0.15 % |
| Wetted parts | PVDF / Ruby bearing |
| Tube connection | 7 mm hose barb |
| Tube length in mm | 53 |
| Liquid temperature in °C | -20 tot +80 |
| Max. pressure at 20°C in MPa | 2.5 (25 Bar) |
| Viscosity in cSt. | 0.8 - 10 |
| K factor (water) in pulse/Litre | 100.000 |
| Power supply | 5 - 30 Vdc |
| Output signal | 5 - 30 V square wave |
| Power consumption | 34 mA at 5 V |
| Electrical cable length | PVC 1 meter |

Figura 3.8: Caudalímetro de precisión (Izq). Tabla de características (Der)

El caudalímetro se alimenta con 5 Vdc, y su señal de salida consiste en una serie de pulsos de onda cuadrada. Para verificar el correcto funcionamiento del caudalímetro, se utilizó el osciloscopio del laboratorio junto con varios valores de caudal. El número de pulsos por minuto es directamente proporcional al caudal que circula por el interior.

De esta manera, se puede establecer una relación entre el número de pulsos y los litros de caudal, a lo que se llama factor K; en este modelo tiene un valor 100.000, aunque el fabricante proporciona una factor de conversión real para cada caudalímetro que difiere un poco. Esto nos proporciona una resolución de 10 µl/ pulso, con una precisión del 1%.

Para confirmar la fiabilidad del este factor K, se hicieron experimentos en el laboratorio. Usando la balanza de precisión, se pesó un recipiente. Con el caudalímetro,

y un sencillo programa de Arduino para registrar el caudal, se vierte un determinado volumen. A posteriori, volvemos a pesar el recipiente y por diferencia obtenemos el volumen de agua. Al compararlo con el registrado por el Arduino, el error cometido era prácticamente despreciable.

Para poder registrar el número de pulsos (y por tanto el caudal regado) de manera ininterrumpida dentro del programa de Arduino, se ha hecho uso de su pin 2 para programar una interrupción.

Las interrupciones en Arduino consisten básicamente, en una sentencia que se ejecutará con una señal de disparo, para la cual el microcontrolador abandonará momentáneamente el punto de programa donde se encuentre en ese momento, atenderá a esa tarea, y una vez finalizada proseguirá por donde se encontraba.

Para la condición de disparo se ha utilizado la transición de nivel bajo a alto, flanco de subida "Rising".

3.1.3 Convertidor Analógico - Digital

En nuestro entorno están presentes de manera continua toda serie de señales analógicas que cambian constantemente, ya sea de manera lenta (como el caso de la temperatura ambiente) o brusca (como una señal de audio).

Si buscamos almacenar estas señales o utilizarlas nos será difícil debido a su naturaleza, por lo que tendremos que utilizar un convertidor analógico-digital. Este dispositivo convierte la señal analógica en un valor binario, de fácil manipulación.

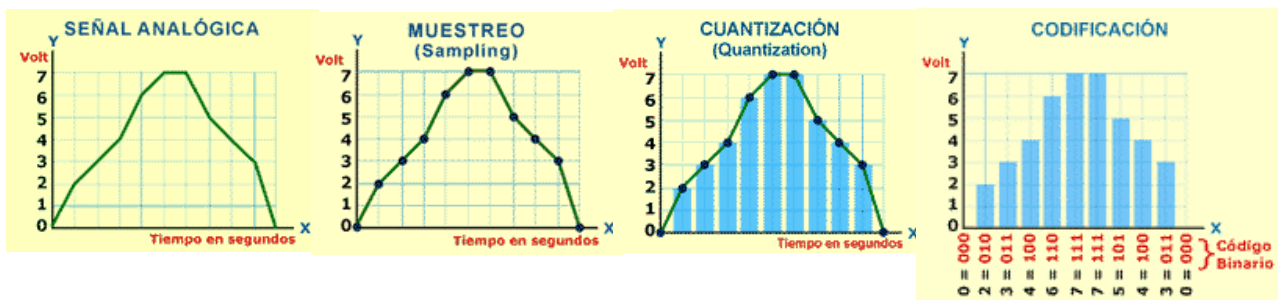


Figura 3.9: Fases de conversión Analógica Digital

De hecho, posteriormente podemos volver a recuperar nuestra señal analógica con el dispositivo opuesto, un convertidor digital- analógico.

La característica más importante a la hora de elegir un convertidor AD es posiblemente su resolución de bits, ya que determina la precisión con que transforma un valor de tipo analógico a digital, siendo el rango del convertidor la diferencia entre el valor DC máximo y mínimo que puede registrar en la señal de entrada.

A continuación voy a describir las principales características de los convertidores analógico-digitales que se propusieron para el trabajo: el CS5534-ASZ y el HX711.

Convertidor CS5534-ASZ de Cirrus Logic

El CS5534-ASZ es un convertidor Analógico – Digital de alto rendimiento, que mediante técnicas de balanceado de cargas consigue obtener 24 bits de precisión. Está optimizado para medir señales de bajo nivel de sistemas de pesada, procesos de control, y en general aplicaciones médicas y científicas. El AD encaja bien en nuestro proyecto puesto que tiene 4 canales de entrada que coinciden con las 4 células de carga.

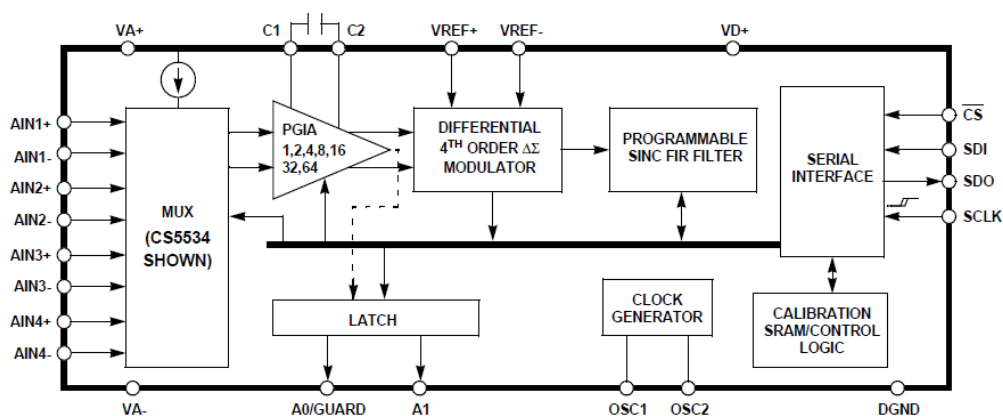


Figura 3.10: Esquema interno del CS5534 (Datasheet)

El resumen de sus características técnicas es el siguiente:

- Convertidor de tipo Delta-sigma con resolución de hasta 23 bits sin ruido (noise-free)
- Selector de ganancias: 1x hasta 64x
- 4 canales de entrada multiplexados
- Interfaz simple de 3 hilos: SPI y Microwire
- Velocidad de muestreo variable: desde 6.25 hasta 3840 Sps
- Configurable a 50 o 60 Hz
- Diferentes configuraciones de alimentación:
 - Unipolar (+5V)
 - Bipolar ($\pm 2.5V$) y ($\pm 3V$)

El CS5534 por tanto tiene varias configuraciones de alimentación: dos bipolares ($\pm 2.5V$ y $\pm 3V$) y una unipolar ($+5V$). Después de realizar varios experimentos con estas configuraciones, y tras determinar que no afecta de manera significativa a la precisión de las medidas, se ha optado por utilizar la configuración Unipolar ($+5V$), debido a su menor complejidad de montaje.

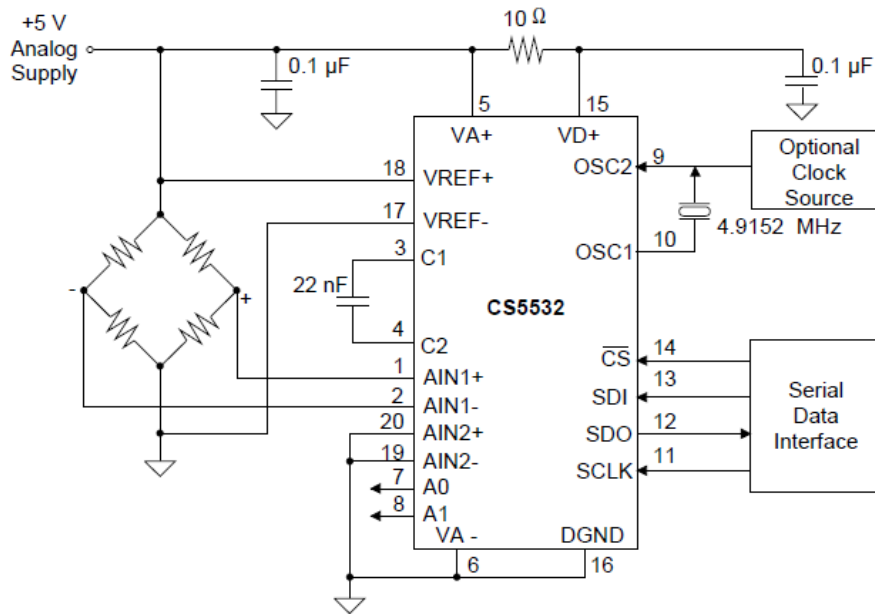


Figura 3.11: esquemático CS5532 para alimentación simple de 5V (Datasheet)

En este caso se dispone de un convertidor AD de 24 bits, por lo que tendrá una resolución de 16.777.216 divisiones. Con este número tan alto, se pueden registrar variaciones muy pequeñas en la tensión de entrada. La resolución es la cantidad de bits que se utilizan para representar una señal analógica. A mayor resolución más número de subdivisiones para dividir el rango y por tanto más preciso será.

La resolución del convertidor usado de 24 bits es:

$$Resolución = \frac{V_D}{V_S \cdot 2^n} \cdot E = \frac{5}{2^{24}} = 0,000000298 V/bit = 0,3 \mu V/bit$$

Para la mayoría de aplicaciones que no requieren de mucha precisión, una resolución de 10 bits es más que suficiente. No obstante, las variaciones del orden de los gramos son tan pequeñas en la salida de las células de carga que necesitamos un AD con una gran resolución. Por tanto, nuestro CS5534 será capaz de registrar variaciones de peso de la salida de las células de hasta 2 gr en la de 10 Kg y de 6 gr en la de 30 Kg.

Convertidor HX711 de Avia Semiconductor

El HX711 es un convertidor analógico digital de 24 bits de resolución, diseñado para sensores de pesada y en general aplicaciones para control industrial.

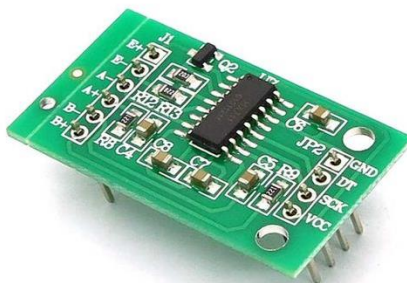


Figura 3.12: Convertidor HX711

Sus características técnicas principales las podríamos recoger en:

- 2 canales de entrada diferenciales A y B
- Selector de ganancias: 32x, 64x y 128x
- No requiere de cristal oscilador externo
- Interfaz simple de 3 hilos SPI
- Velocidad de muestreo variable: 10 u 80 Sps
- Configurable a 50 o 60 Hz
- Alimentación de 2.6 ~ 5.5 V

Es un módulo bastante sencillo de utilizar ya que se pueden encontrar librerías para tal efecto. El canal A puede programarse para ganancia de 64x y 128x mientras que el canal B tiene ganancia fija de 32x.

Comparativa de pruebas realizadas

Para poder elegir de manera justificada entre los dos convertidores AD, se han realizado experimentos para evaluar su precisión, repetitividad, y en general su comportamiento bajo las mismas condiciones. De esta manera, a su vez, determinaremos las condiciones óptimas de trabajo del convertidor finalmente seleccionado. Los experimentos se han resumido a continuación, y los valores se pueden encontrar en el Anexo III:

- Prueba 1. Alimentación: Calibrador 1. Entrada: Calibrador 2

En este experimento se ha hecho uso de dos calibradores de precisión del laboratorio. Con el primero de ellos, se alimentó externamente el convertidor AD con diferentes tensiones y configuraciones (bipolar / unipolar). Se ha dividido el rango de alimentación en los siguientes puntos:

Unipolar: +4.5, +4.75, +5, +5.25, +5.5V

Bipolar: ±2.5, ±2.75, ±3V

Con el segundo calibrador simultáneamente, se ha simulado las diferentes tensiones que podría generar una célula de carga en la entrada. Se dividió el rango teórico en el que va a trabajar la célula. Las células generan una salida máxima de 10mV, siendo su sensibilidad de 2mV/V. Por tanto, se ha diseñado el experimento con las siguientes señales de entrada: 0, 2, 4, 6, 8 y 10mV.

El objetivo del experimento es determinar cómo varía la precisión de las lecturas de los convertidores AD en función de la tensión de alimentación.

Para convertir las lecturas digitales a analógicas, se han usado la ecuación:

$$mV = \frac{V_{dc} \cdot \text{lectura AD} \cdot 1000}{(2^{24} - 1) * \text{Ganancia}}$$

Donde nuestra ganancia vale 128 para ambos convertidores.

- Prueba 2. Alimentación: Calibrador. Entrada: Célula de carga

Para esta prueba, se ha alimentado el convertidor AD desde el calibrador del laboratorio. A la entrada del AD hemos utilizado una Célula de carga real. El objetivo de la misma es observar el comportamiento frente a incrementos y decrementos de peso y tensión de alimentación. Para ello, se ha hecho uso de unas pesas calibradas.

Las tensiones de alimentación serán las mismas mencionadas anteriormente:

Unipolar: +4.5, +4.75, +5, +5.25, +5.5V

Bipolar: ±2.5, ±2.75, ±3V

Se ha dividido el rango de la célula (30 kg) en 6 puntos: 0, 5, 10, 15, 20 y 25 Kg. Por tanto, se han combinado las pesas del laboratorio para obtener de la manera más aproximada posible, y siempre con los mismos pesos, los 6 puntos necesarios para el experimento.

A estas pesadas, además, ha habido que sumarles los 240,05 gramos del plato de pesada, de ahí que estén separadas en columnas (siendo la que incluye el plato la usada para los datos finales).

Esta prueba se ha dividido en dos apartados:

- Prueba 2 (I). Para tensión de alimentación constante (5V), variamos el peso de la célula de carga.
 - Prueba 2 (II). Para un mismo peso de la célula, variamos la tensión de alimentación del AD
- Prueba 3. Alimentación: Arduino. Entrada: Célula de carga

En este experimento queremos comparar el comportamiento de los convertidores AD cuando la tensión de alimentación proviene del Arduino en lugar del Calibrador de precisión, siendo la del Arduino de naturaleza más inestable y menos depurada. En este apartado hemos repetido la Prueba 2 (I) bajo las condiciones mencionadas.

- Prueba 4. Repetición de la Prueba 3 con la Guarda activa

Para este apartado se ha activado la “guarda” del CS5534-ASZ. Básicamente, consiste en conectar la malla del cable de las células de carga al convertidor AD, de manera que evita interferencias y ruido. Por tanto, se ha repetido la prueba 3 activando la guarda.

- Conclusión

Analizando los resultados obtenidos, se puede concluir que el **CS5534 presenta mayor repetitividad** que el HX711. Por tanto, para nuestro proyecto hemos utilizado finalmente el convertidor CS5534-ASZ.

Además, observando las pruebas, podemos concluir que el convertidor CS5534-ASZ tiene una **precisión excelente** bajo todas las configuraciones en general. No obstante, presenta un ligero aumento de precisión con la configuración bipolar. Esto era de esperar puesto que la configuración bipolar proporciona mayor estabilidad, pero al no ser un incremento significativo (prácticamente despreciable) se ha optado por la **configuración unipolar** por su sencillez.

De manera que, dentro de esta configuración, la tensión de alimentación no afecta de manera significativa a las lecturas, por tanto se ha utilizado finalmente una **tensión de alimentación de +5V** por ser la más común.

3.1.4 Optoacoplador

Para aislar eléctricamente la comunicación SPI entre el CS5534-ASZ y el Arduino, se ha utilizado un optoacoplador modelo ACSL – 6410. Se ha usado este modelo puesto que tiene una velocidad de transmisión de datos muy alta, desde 10 hasta 15 Mbd, por lo que garantizará una buena transferencia de datos. Las especificaciones técnicas del ACSL – 6410 son:

- Comunicación dúplex bidireccional
- 4 canales en configuración 3 / 1
- Tensión de alimentación de 3 ~ 5.5 V
- Alta velocidad de transferencia: 10 ~ 15 Mbd
- Compatible con LSTTL / TTL
- Configurable a 50 o 60 Hz
- Alimentación de 2.6 ~ 5.5 V
- Aislamiento eléctrico completo mediante fotodiodos

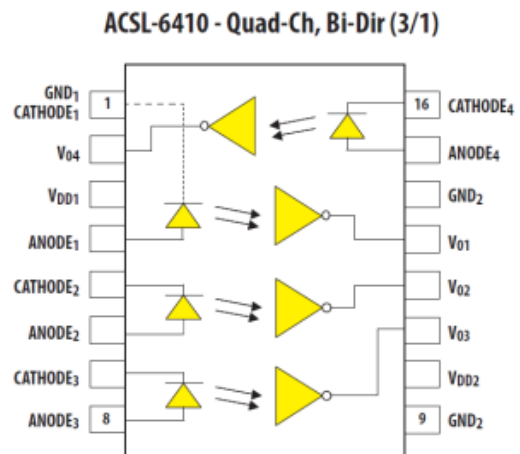


Figura 3.13: Diagrama del ACSL - 6410

Estas características hacen este modelo de optoacoplador perfecto para aplicaciones SPI, I2C, comunicaciones dúplex, interfaces para sistemas de microprocesadores, y aislamiento digital para conversiones A/D y D/A.

3.1.5 Real Time Clock

Para poder tener un registro de datos es necesario que los mismos dispongan de su correspondiente fecha/hora que sea fiable y que no se vea afectada por caídas de tensión de alimentación. Para ellos se ha utilizado un reloj a tiempo real o RTC, modelo DS1302.



Figura 3.14: RTC modelo DS1302

El DS1302 es un circuito integrado que registra la hora y fecha: segundos, minutos, horas, día de la semana, día del mes, mes y año (coherencia temporal hasta el 2100). El final de mes es ajustado automáticamente para aquellos meses con menos de 31 días, incluyendo el ajuste de salto de año. El reloj permite operar en modo 24 horas y 12 horas, con su correspondiente indicador AM/PM.

El RTC cuenta con interfaz de 3 hilos (Clock, Input/Output, Chip Enable). Este integrado consume muy poco (300 nA a 2V) por lo que su vida “útil” es muy alta. El DS1302 tiene un rango de alimentación de 2 ~ 5.5V y necesita de una pila, en este caso hemos usado una de 3V de tipo CR2032. Para generar la señal de reloj utiliza un oscilador de cuarzo de 32.768kHz.

3.1.6 Módulo HY – 1.8 SPI

Para poder monitorizar los experimentos del lisímetro en la finca, se utilizó el módulo HY-1.8 SPI para visualizar y registrar las lecturas. El módulo HY-1.8 SPI incluye una pantalla TFT a color de 1.8 pulgadas de alta resolución, y una ranura para tarjeta SD. La ranura y la pantalla funcionan de manera independiente mediante interfaz SPI. El módulo dispone de un regulador de tensión para la pantalla de 3.3V (estamos conectando directamente a los 5V del Arduino) con 30mA de corriente.

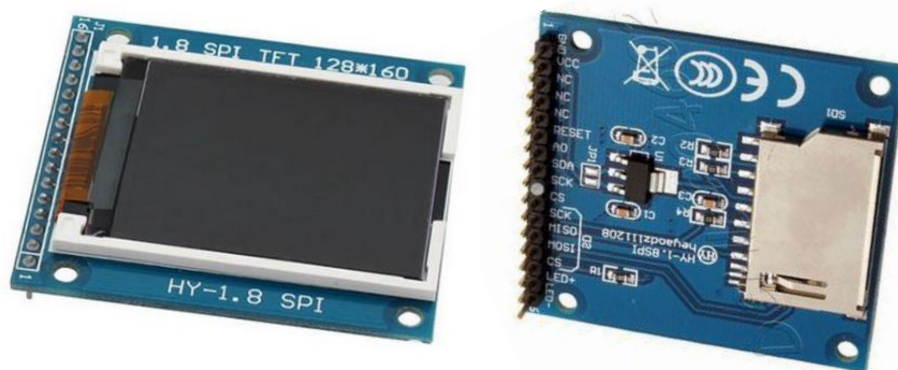


Figura 3.15: Módulo HY -1.8 SPI

La pantalla TFT cuenta con resolución de 128x160 píxeles, 262K colores por píxel, retro-iluminación LED, y cuenta con una RAM interna de capacidad 132x162x18 bits. Para poder comunicarse utiliza el chip controlador ST7735R para pantallas gráficas de tipo TFT-LCD. Este controlador acepta interfaz de 3 hilos, 4 hilos y MCU, y tensión de trabajo 3 ~ 5V.

3.2 Elementos actuadores

Un actuador es aquel dispositivo que genera un movimiento físico de tipo lineal o rotacional, a partir de una fuente de alimentación y bajo las órdenes de una señal de control. La señal de control suele ser poco energética y de naturaleza eléctrica, aunque la solemos encontrar de tipo hidráulica y neumática, o incluso el propio ser humano. Cuando el actuador recibe la señal de control, convierte la energía de alimentación en movimiento mecánico.

3.2.1 Electroválvulas de 230V

Electroválvulas de drenaje

El control del vaciado de la bola de drenaje está compuesto por 2 electroválvulas de 230V, una de tipo NO y otra NC.



Figura 3.16: Electroválvula para drenaje

La normalmente abierta se encuentra en la parte de arriba de la bola de drenaje, y permite el paso de agua de manera habitual. La electroválvula normalmente cerrada se ha instalado debajo de la bola, gestionando el vaciado de la bola.

La electroválvula de arriba se utilizó para evitar falsear las medidas, de manera que durante el tiempo que está la salida de la bola abierta no siga entrando agua a la bola que no vaya a registrarse. Ambas electroválvulas se activan con un relé conectado a Arduino y a 230V (Apartado 3.2.2).

Electroválvula de riego

El paso de agua de irrigación se controló gracias a una electroválvula reciclada de lavadora de 230V.



Figura 3.17: Electroválvula para riego

Se conectó la entrada de la electroválvula al agua corriente mediante una manguera, y su salida al sistema de riego por goteo del lisímetro, pasando por el caudalímetro. Su apertura se reguló con un programador externo (Apartado 3.3.2)

3.2.2 Relé de 230V

Para poder abrir y cerrar las dos electroválvulas de 230V, se ha hecho uso de este módulo con relé que permite manejar cualquier tipo de electroválvula comercial. Usa un acoplamiento óptico permitiendo aislar la zona de control de la salida.



Figura 3.18: Módulo relé para electroválvulas

Permite conectar a su salida una carga de hasta 250V 10A en AC, y 30V 10A en DC. El voltaje de alimentación en la zona de control es de 5 ~ 24 V, con corriente de disparo de 5 mA. Esto permite conectarlo directamente a los pines del Arduino. El módulo incorpora dos Leds para verificar su funcionamiento (verde para verificar la alimentación, y otro rojo para monitorear el estado del relé). Además, mediante un jumper podemos configurar la señal de disparo de nivel alto o bajo.

3.3 Dispositivos de Control

Son aquellos elementos del sistema cuyo objetivo es el recibir información por parte de los sensores y/o regular la actividad de los actuadores. En este apartado describiremos el corazón de nuestro sistema, una placa de Arduino Uno, y el otro dispositivo de control, el Programador digital.

3.3.1 Arduino Uno

Arduino es una plataforma de hardware abierto basada en una placa con un microcontrolador de la marca Atmel de bajo coste, y una sencilla circuitería: puerto USB, una serie de entradas y salidas de tipo analógico y digital, reguladores de tensión, etc.

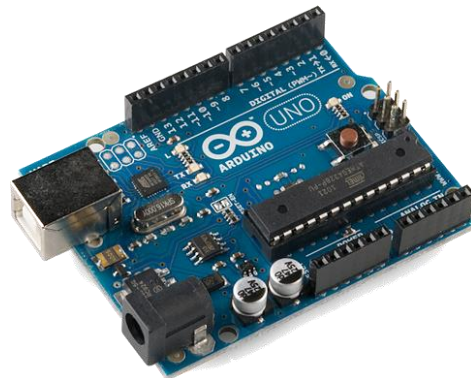


Figura 3.19: Placa Arduino Uno

A continuación se resumen las características técnicas:

- Microcontrolador: Atmega328
- Voltaje de operación: 5V
- Voltaje de entrada: 6 ~ 12V
- Interfaz: Serial, SPI e I2C
- Pines digitales: 14 (6 programables para salida PW)
- Pines analógicos: 6
- Corriente continua: 40 mA
- Memoria flash: 32 KB (0.5 ocupados por el bootloader)
- SRAM: 2 KB
- EEPROM: 1 KB
- Frecuencia reloj: 16 MHz

El Arduino tiene 14 pines digitales configurables para salida o entrada, y pueden dar hasta 5V, y la corriente de entrada/salida máxima es de 40mA. Cada uno de estos pines

cuenta con una resistencia interna entre $20k\Omega$ y $50k\Omega$ de tipo pull-up. 6 de estos pines pueden configurarse como salidas para generar señales de tipo PWM (modulación por ancho de pulso) de hasta 8 bits.

Dispone también de 6 pines analógicos de entrada, cuyas señales son procesadas por un convertidor analógico digital de 10 bits, por lo que devuelve valores entre 0 y 1023.

Arduino dispone de varios tipos de comunicación; la Serial en los pines 0 y 1, también llamados Rx y Tx; comunicación SPI de los pines 10 al 13, permitiendo controlar paralelamente diferentes dispositivos, y también el I2C, permitiendo establecer comunicaciones con hasta 127 dispositivos. La alimentación de la placa está entre los 5V y los 12V, siendo el límite inferior limitante para sacar del regulador de tensión los 5V, y si superamos los 12V hay posibilidad de daño.

3.3.2 Programador de riego

Para programar las horas y duración de los riegos, se ha utilizado el programador digital Coati modelo 13201 (Figura 3.20). A su salida se ha conectado la electroválvula para riego (Figura 3.17) para permitir la irrigación.



Figura 3.20: Programador digital Coati

Se programaron los riegos diariamente, a las 13:00 horas (se decidió que era una buena hora para poder medir la evapotranspiración), y durante una duración de 15 minutos, suficiente tiempo como para conseguir empapar la tierra de la maceta (capacidad de campo) y que se produzca un poco de lixiviado.

El programador digital de Coati es muy cómodo, ya que cuenta con una pantalla LCD, función manual, programable y aleatoria, hasta 16 programas diferentes diarios, horario de verano y cuenta atrás. Solo necesita dos pilas de 1.5V y te permite controlar cargas de hasta 3.500W a 230V/50Hz.

3.4 Shields para Arduino

Los Shields son placas de expansión que pueden conectarse encima de la placa Arduino para extender en algún aspecto su capacidad y características. Son sencillas de montar y su coste suele ser bastante asequible. Durante la realización del proyecto, se han hecho uso de 3 shields diferentes para Arduino:

1. Shield para el convertidor CS5534-ASZ, para el Arduino de Adquisición
2. Shield para el módulo HY-1.8 SPI y DS1302, para el Arduino de Registro, en la finca (I)
3. Shield comercial Ethernet para el Arduino de Registro, en la finca (II)

3.4.1 Shield para el convertidor CS5534-ASZ

Este shield fue diseñado con el objetivo de utilizar de manera compacta y cómoda el CS5534-ASZ. Para su montaje se utilizó el laboratorio Desi ya que disponía de una estación de soldadura por flujo de aire caliente.

Como se puede observar en la Figura 3.21, el shield tiene 4 conectores que corresponden con los 4 canales físicos del AD. A cada uno de estos conectores se ha conectado una de las células de carga. Cada conector posee 5 cables que corresponden con:

- La tensión diferencial positiva del puente correspondiente a la medida de peso: **AIN+**
- La tensión diferencial negativa del puente correspondiente a la medida de peso: **AIN-**
- La tensión de referencia positiva que alimenta al puente de la célula: **Vref+**
- La tensión de referencia negativa que alimenta al puente de la célula: **Vref-**
- La malla del cable de la célula de carga: **Guarda**

Para evitar el ruido en el CS5534-ASZ procedente del Arduino, la comunicación con el AD está optoacoplada, por lo que se ha utilizado una fuente de alimentación independiente para alimentar el CS5534-ASZ (Figura 3.21, +5V). Este conector tiene V+, V- y Gnd, de manera que permite cambiar de manera casi inmediata entre alimentación unipolar y bipolar. En nuestro caso hemos utilizado Unipolar a 5V, por lo que V- y Gnd están puenteados.

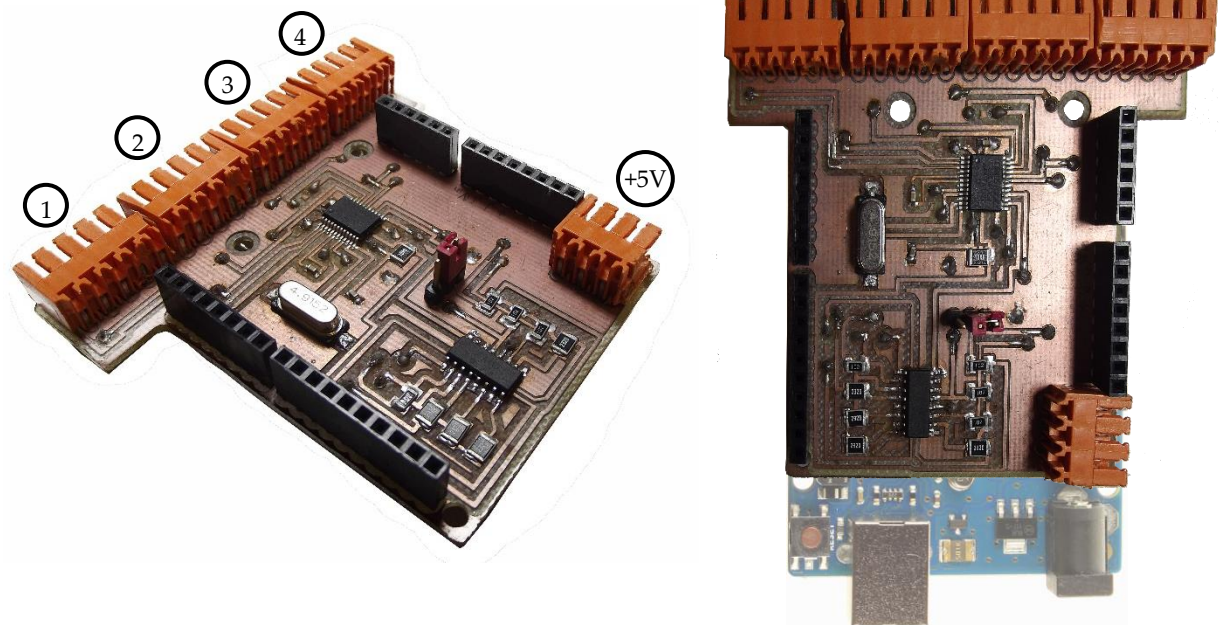


Figura 3.21: Shield para el convertidor CS5534-ASZ

Los componentes utilizados para el montaje de la placa son:

- Placa PCB
- Convertidor CS5534-ASZ
- Un optoacoplador ACSL – 6410
- Cristal de cuarzo de 4.9152 MHz
- Tiras de pines y zócalos para los cables
- Resistencias y condensadores SMD

En la Figura 3.22 se puede observar el diagrama esquemático de funcionamiento. En la parte izquierda del diagrama nos encontramos las salidas digitales del Arduino.

Se comunica con el AD mediante interfaz SPI, por lo que Arduino utiliza los pines 13, 12, 11 y 10 para tal efecto. Estos cuatro cables (SDO_Ard, SDI_Ard, CLK_Ard, CS_Ard) procedentes de Arduino entran a las patillas 1-8 del optoacoplador ACSL – 6410, y a su salida, sus homólogos (SDO_AD, SDI_AD, CLK_AD, CS_AD) van al CS5534-ASZ que tenemos a la derecha del diagrama (patillas 13 – 16).

En la parte inferior vemos los 3 cables de alimentación independiente del CS5534-ASZ, y en la superior los 4 zócalos de las 4 células de carga. Como explicamos antes, cada célula tiene 5 cables, siendo la Guarda, Vref+ y Vref- común para todas ellas.

Para completar el resto del circuito, se han utilizado componentes Smd:

- 4x resistencias de 230Ω
- 4x resistencias de 1kΩ
- 1x resistencia de 10Ω
- 1x condensador de 22nF
- 2x condensadores de 0.1uF

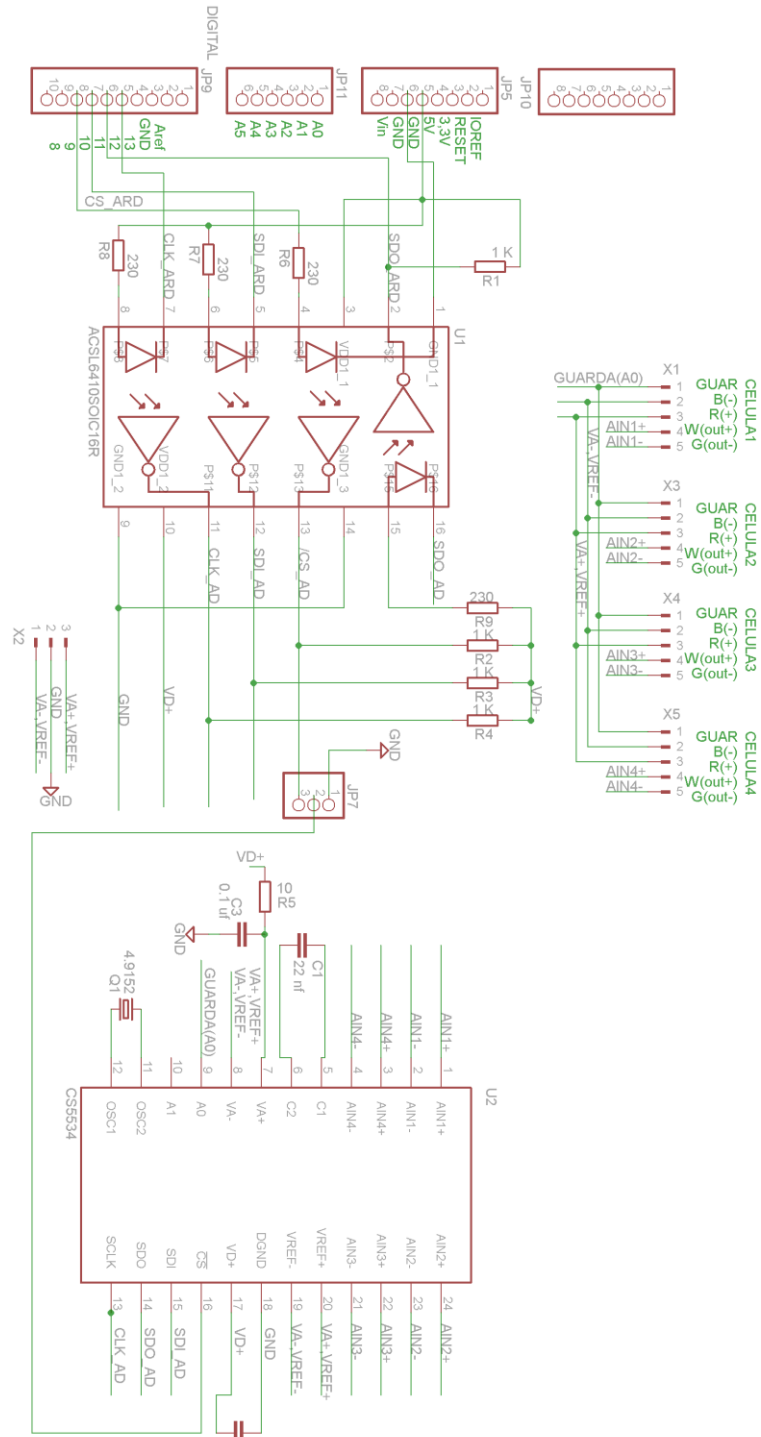


Figura 3.22: Esquemático del shield CS5534-ASZ. Aplicación Eagle

3.4.2 Shield con módulo HY-1.8 SPI y DS1302

Para poder realizar los experimentos en la finca, se necesitaba de un dispositivo que recibiese las medidas del Nodo de Adquisición y las almacenase. En el Modelo B: Finca (I) este dispositivo es un Arduino Uno junto con un Shield.

Este shield (Figura 3.23) está compuesto por una pantalla TFT para representar las lecturas y verificar el correcto funcionamiento del lisímetro, una ranura SD para almacenar las medidas, y además de un DS1302 para configurar la fecha y hora. La pantalla TFT y la ranura SD componen el módulo HY-1.8 SPI.

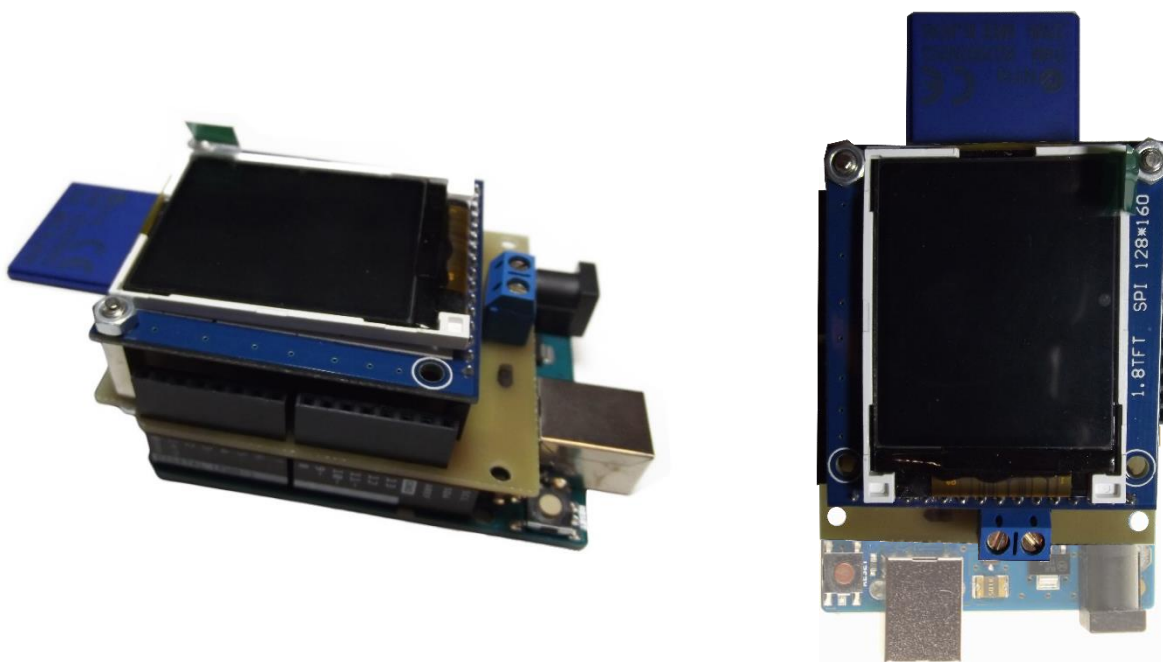


Figura 3.23: Shield con HY-1.8 SPI y DS1302

Los componentes utilizados para el montaje de la placa son:

- Placa PCB
- Módulo HY-1.8 SPI
- DS1302
- Cristal de cuarzo de 32.768kHz
- Pila de 3V con zócalo
- Tiras de pines y zócalos para los cables
- Tarjeta SD

La pila que se ha usado para el DS1302 es de 3V tipo CR2032. Para poder utilizar la tarjeta SD, requiere un formateo a FAT16 o FAT32.

En la Figura 3.24 se puede ver el diagrama esquemático de funcionamiento. En la parte inferior izquierda del diagrama nos encontramos la placa de Arduino Uno con sus salidas digitales en la parte superior. Arriba está el módulo HY-1.8 SPI y a la derecha, el DS1302 con su pila CR2032.

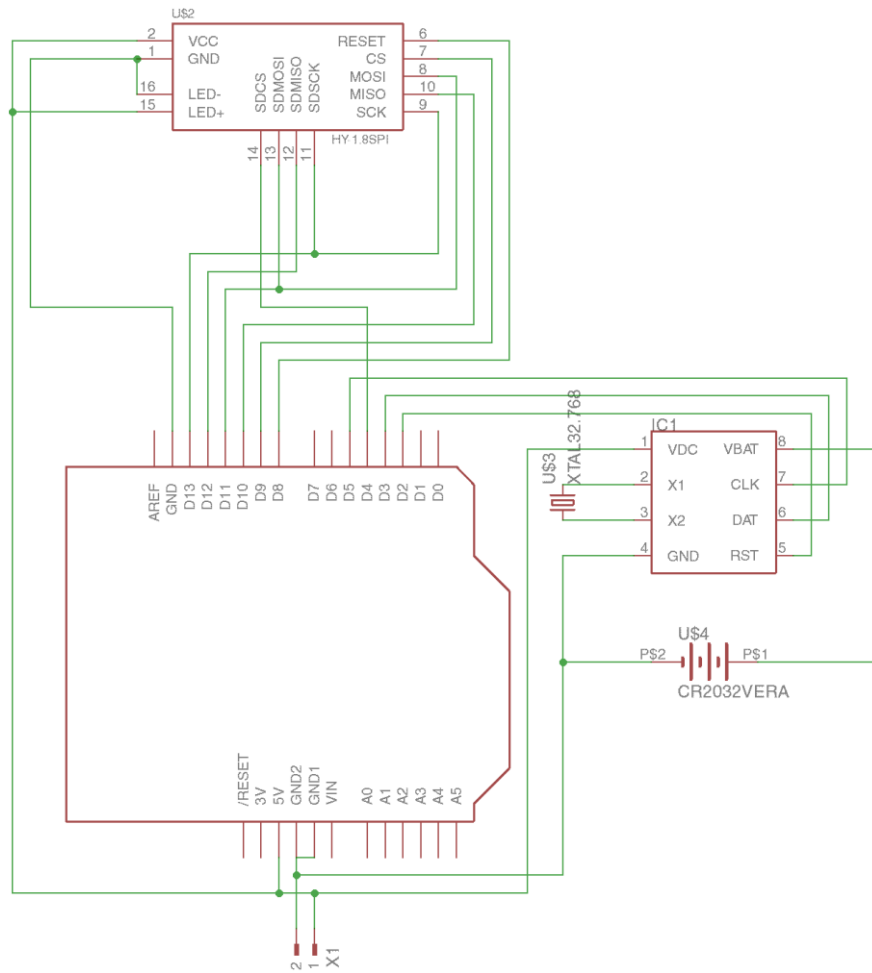


Figura 3.24: Esquemático del shield HY-1.8 SPI y DS1302. Aplicación Eagle

El Arduino utiliza interfaz SPI para la pantalla TFT y la tarjeta SD (pines 13, 12, 11, 10, 9, 8 y 4) e interfaz de 3 hilos para el DS1302 (pines 5, 3 y 2).

3.4.3 Shield Ethernet

Para poder subir las medidas del lisímetro a la nube, se ha utilizado el Shield comercial Ethernet ya que permite al Arduino conectarse a la nube. Tiene un precio que ronda los 8€ y sólo necesita instalar una librería y un router con conexión para comenzar a utilizarlo.

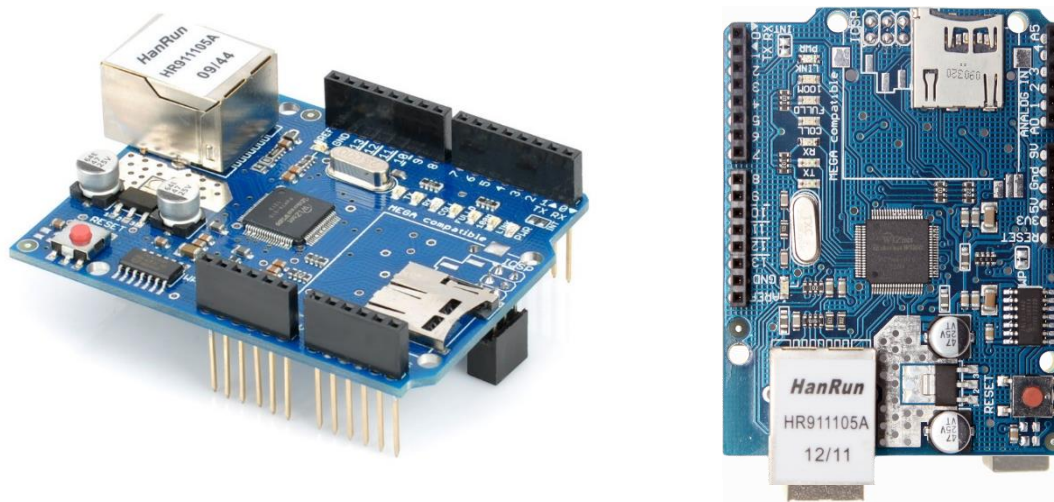


Figura 3.25: Shield Ethernet

Sus características técnicas principales son:

- Ranura para microSD
- Conexión a la red con clavija RJ45
- Voltaje de operación a 5V (del Arduino al que se conecta)
- Basado en el chip WIZnet W5100 (buffer interno de 16K)
- Velocidad de conexión de 10 ~ 100 Mb
- Comunicación SPI con el Arduino

El corazón del shield es el chip WIZnet W5100, que proporciona a nuestro sistema una red (IP) para utilizar TCP y UDP. Como se puede apreciar, el Shield Ethernet incluye una ranura para una tarjeta microSD; esta característica es especialmente útil ya que permite almacenar los datos en un dispositivo externo (microSD) en caso de fallo de conexión al servidor. Para usar la microSD basta con utilizar la conocida librería SD.

Para poder conectarlo a la red, se ha utilizado un router con conexión a internet, y se ha hecho llevar un cable RJ45 hasta el shield.

IV

Capítulo 4.

Sistema de Instrumentación

En este cuarto capítulo se describe el sistema de instrumentación utilizado con sus diferentes configuraciones y nodos, todo ello desarrollado a partir del hardware anteriormente descrito.

4.1 Descripción general

El lisímetro consta de una serie de células de carga ensambladas a una estructura metálica, sobre la cual descansa una maceta con una muestra del cultivo a estudiar (utilizamos 3 células de carga para determinar el peso total de la planta, y otra célula para la bola que contiene el agua que drena). La planta utilizada es una *Euonymus fortunei* de tres años.

Diariamente, la planta es regada, y una vez terminado el riego, una determinada cantidad de agua empapará la tierra; una vez llegado este punto, el resto del agua que echemos discurrirá por la tierra hasta la bola de drenaje. Una vez finalizado el riego, vaciará la bola y esperará hasta el siguiente riego. Para realizar el riego se utiliza un caudalímetro de precisión y dos goteros de riego autocompensados.

Para el vaciado de la bola se usó un relé para activar una electroválvula NC. Para evitar falsear las medidas, se utiliza otra electroválvula NO para la entrada a la bola, de manera que durante el tiempo que está la salida de la bola abierta no siga entrando agua a la bola. Al finalizar el vaciado de la bola, y al volver a abrir la entrada a la bola, podemos observar un pequeño incremento en el peso debido al agua acumulada a la entrada de la bola durante el vaciado en la Figura 4.1.



Figura 4.1: Gráfica de vaciado de bola (Izq). Bola de drenaje (Der).

El programa de Arduino vacía la bola cuando detecta el inicio de riego, evitando desbordar la bola (por si ésta se encontraba llena debido a lluvias) y vuelve a vaciarla pasada una hora después del fin de riego, dándole a la tierra tiempo más que suficiente para lixiviar toda el agua sobrante.

Se ha tomado pertinente almacenar medidas minutalmente, puesto que hablando en términos de evapotranspiración vegetal, es tiempo más que de sobra para registrar cambios en el peso de la maceta. Pero como las variaciones más significativas se concentran durante el drenaje de agua y el riego (se produce la mayor cantidad de evapotranspiración e incremento de peso), durante estos intervalos de tiempo se almacenarán cada nueve segundos.

Nuestro sistema muestrea el peso cada tres segundos, y cada minuto realiza la media aritmética de los datos, y la almacena. Junto al peso se guardan la fecha, hora, el peso de la bola de drenaje, volumen de agua regado, y diversos estados booleanos del programa (regando, pre-vaciando, llenando, vaciando).

Para realizar los riegos, se utilizó un programador externo (Figura 4.2) y se configuró como hora de riego al medio día (periodo en el que supuestamente se genera la máxima evapotranspiración).

Se determinó como duración de regado 20 minutos, tiempo suficiente como para conseguir empapar la tierra contenida en la maceta y lixiviar un mínimo de agua. De esta manera nos es difícil determinar la capacidad de campo.



Figura 4.2: Programador de riego

Nodo de Adquisición y Nodo de Registro

El sistema de instrumentación se ha dividido en dos Nodos en función de su función: Nodo de Adquisición y Nodo de Registro.

El Nodo de Adquisición (Figura 4.3) está compuesto por todos aquellos actuadores y sensores que participan en la adquisición de datos y en general del control del lisímetro. Este Nodo no varía a lo largo de las diferentes configuraciones (Modelos A, B y C), tan solo cambia el modo en que se comunica con el Nodo de Registro.

Los elementos que los componen son:

- Arduino de Adquisición (Arduino Uno con Shield CS5534-ASZ)
- 4 Células de carga
- 1 Caudalímetro
- 1 Relé
- 2 Electroválvulas

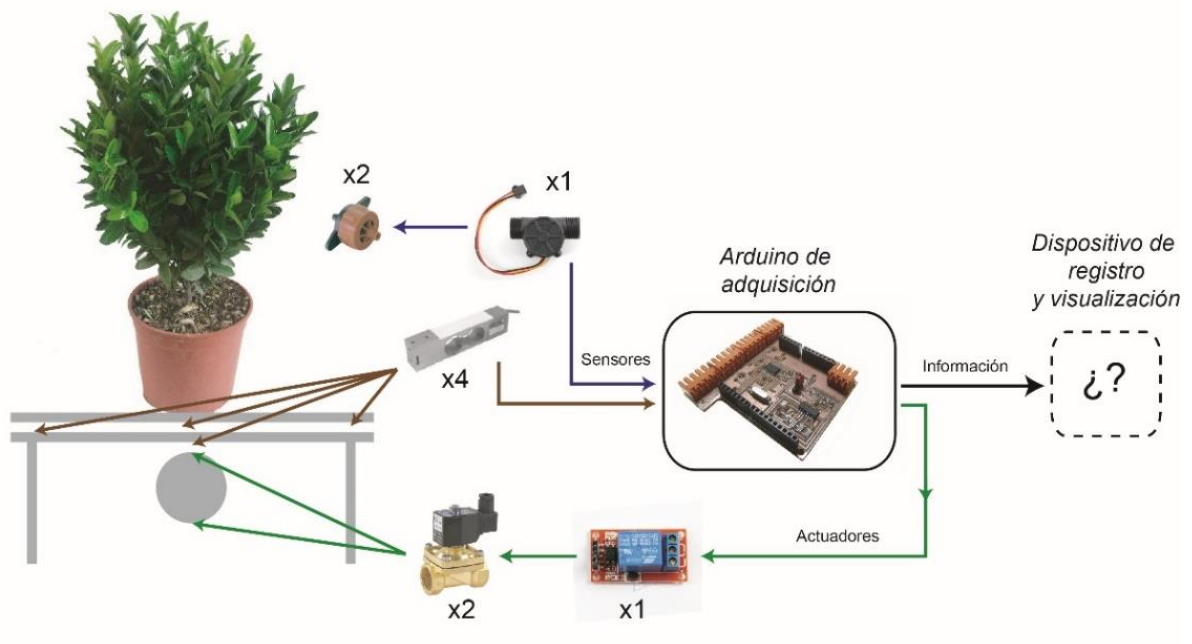


Figura 4.3: Elementos del lisímetro. Nodo de Adquisición

El Nodo de Registro es el encargado de almacenar y registrar las medidas, además de permitir la visualización a tiempo de las lecturas. Se han utilizado tres Nodos de Registro diferentes durante desarrollo del TFG, en función del dispositivo encargado de almacenar los datos:

- Modelo A: un ordenador, mediante una aplicación de LabView
- Modelo B: Arduino de Registro (Arduino con Shield HY-1.8 SPI + RTC)
- Modelo C: Arduino de Registro (Arduino con Shield Ethernet)

En la primera fase, se han realizado en el Laboratorio las calibraciones y ensayos para la puesta en marcha del lisímetro (Modelo A). Más tarde, se trasladó el lisímetro a la Estación Experimental Agroalimentaria Tomás Ferro localizada en La Palma (Murcia), permitiendo evaluar su comportamiento frente a condiciones de trabajo reales (Modelo B y C).



Figura 4.4: Estación Experimental Agroalimentaria Tomás Ferro. Localización del lisímetro

4.2 Filtrado de datos

A medida que se iban obteniendo datos de “largo plazo” (dejar el sistema tomando medidas durante al menos 24 horas) se pudo observar que algunos datos almacenados eran erróneos y no tenían coherencia con el resto.

Estos datos aislados, aparecen por alguna razón: un fallo momentáneo en algún sensor, un error de comunicación entre Arduino y su dispositivo de almacenamiento (Pc, SD, web) y por tanto almacena un cero, o la presencia de alguna persona ajena que accidentalmente se apoya en la estructura.

Debido a todo esto, se implementó en el sistema una serie de filtros para descartar los datos inválidos. Para los procesos de adquisición de datos de variables meteorológicas, el AENOR UNE 500520 standard establece una serie de criterios de adquisición. En este caso, usaremos un algoritmo de filtrado de datos con diferentes niveles: nivel cero, nivel uno y nivel dos (Molina-Martinez et al. 2012).

- El nivel cero es el más básico, en el cual un dato será descartado automáticamente cuando alguna de sus variables de un error debido a una mala comunicación o conexión con el sensor o dispositivo.
- El nivel uno es la validación de datos en función de los límites del sistema, divididos en rígidos y flexibles. Primeramente tenemos los límites rígidos, que son los delimitados por los límites físicos del sensor de medida. Si tenemos una célula de carga de 100 kg, será imposible que pueda medir (de manera coherente)

por debajo de 0 kg y por encima de 100 kg. Por tanto, estos serán sus límites rígidos.

Por otra parte, los límites flexibles los delimitan las condiciones de trabajo del sistema. Si resulta que este mismo sensor de antes, siempre tendrá una carga variable entre 40 y 60 kg, entonces estos serán sus límites flexibles. Para obtener el límite de nivel uno, se usa el más restrictivo obtenido de la suma de rígido y flexible.

- El nivel dos valida los datos en función de su coherencia temporal. Si dos datos consecutivos en el tiempo difieren más de un tanto por ciento (estudiado con anterioridad) se deberá descartar. Si por ejemplo, nuestra estructura del lisímetro sufre un decremento de 5 kg de un minuto a otro, se deberá de considerar erróneo ya que es imposible que la planta haya perdido tal cantidad de agua en ese espacio de tiempo.

A continuación se muestra la tabla que resume los límites utilizados durante el desarrollo del proyecto:

| | NIVEL 1: Límite rígido | | NIVEL 1: Límite flexible | | NIVEL 2: Coherencia temporal |
|---|---------------------------|--------|-----------------------------|-----------------------|------------------------------------|
| | Mínimo | Máximo | Mínimo | Máximo | Variación por segundo |
| Caudalímetro - Caudal agua (pulsos / litro) | 50 ⁽¹⁾ | 3.333 | 0 | 150 ⁽²⁾ | ± 150 |
| Célula carga - Peso lisímetro (gr) | 0 | 30.000 | 8.500 ⁽³⁾ | 27.000 ⁽⁴⁾ | ± 500 |
| Célula carga - Peso drenaje (gr) | 0 | 10.000 | 550 ⁽⁵⁾ | 4.500 ⁽⁶⁾ | ± 500 |

Tabla 3: Límites aplicados al filtrado de datos

- (1) Nuestro caudalímetro tiene un umbral mínimo de 50 pulsos / litro.
- (2) Al utilizar dos goteros autocompensados de 2 litros/hora, se determina por tanto que el número máximo de pulsos por litro que el caudalímetro registrará en condiciones normales será de 150. Se ha implementado además que, en caso de fuga de agua (al subir este valor por encima de 150) se registre y se notifique.

- (3) Se ha determinado este valor restándole el 50% del valor de peso inicial (estructura con maceta y planta) sin regar.
- (4) Se ha determinado este valor sumándole el 50% del valor de peso inicial (estructura con maceta y planta) sin regar.
- (5) Es el peso de la bola de drenaje vacía.
- (6) Es el peso de la bola de drenaje llena.

Para todos los tipos de configuración (Modelo A, B y C) se ha implementado en el código este filtrado de datos a tiempo real. De esta manera, cuando el sistema se encuentra ante un dato erróneo, no lo tiene en cuenta numéricamente hablando, y a su vez genera una alarma y especifica qué tipo de error se ha encontrado.

En el código del lisímetro, por tanto, se han implementado 4 tipos de errores:

1. Error en la célula N° X: Cuando una célula en concreto está trabajando fuera de su rango normal, se notifica para su revisión y/o sustitución.
2. Fuera de rango global: cuando el peso del lisímetro, suma de las tres células, no está trabajando dentro de su rango de funcionamiento por alguna razón.
3. Pérdida de agua: cuando el caudalímetro registra un valor superior al de los goteros (se ha tenido que implementar la excepción del transitorio inicial, puesto que al abrir la válvula de riego inicialmente se registran unos picos en el caudalímetro que alarmaban falsamente de una fuga de agua)
4. Error en la tarjeta SD, para la finca (I). cuando el Arduino no puede comunicarse correctamente con la SD.
5. Error en de conexión a internet, para la finca (II). cuando el Arduino no puede comunicarse correctamente con la base de datos alojada en la Nube.

4.3 Configuraciones del sistema

Primeramente, se utilizó el lisímetro en un laboratorio, un entorno cerrado para realizar ensayos de control. Después, se instaló el lisímetro al aire libre para estudiar su comportamiento bajo condiciones climatológicas reales.



Figura 4.5: Lisímetro en el Laboratorio (Izq) y en la Finca (Der)

En nuestro diseño, la placa de Arduino hará las veces de Datalogger. Se encargará de leer las diferentes señales de entrada (hora /fecha, agua regada y lecturas de peso) y activar los actuadores (permitir el riego y vaciar los depósitos). Para poder almacenar los datos, se hará uso de otro dispositivo externo con el cual la placa Arduino tendrá que comunicarse.

Modelo A. Ensayo en el Laboratorio

En esta configuración sólo se utiliza una placa de Arduino Uno. Este Arduino se comunica primeramente con el AD incorporado en el Shield para que éste obtenga las medidas de pesada. A éste conjunto lo llamaremos Arduino de Adquisición.

A su vez, el Arduino toma las lecturas del caudalímetro, y envía toda esta información por el puerto Serial COM al ordenador PC, donde tenemos instalada nuestra aplicación de LabView. La aplicación muestra y almacena los datos a tiempo real en un documento lo que permite obtener las tablas Excel de manera cómoda y rápida. Es una cómoda interfaz que permite interactuar con el lisímetro.

En la figura 4.6 se puede observar la estructura general del sistema (Modelo A).

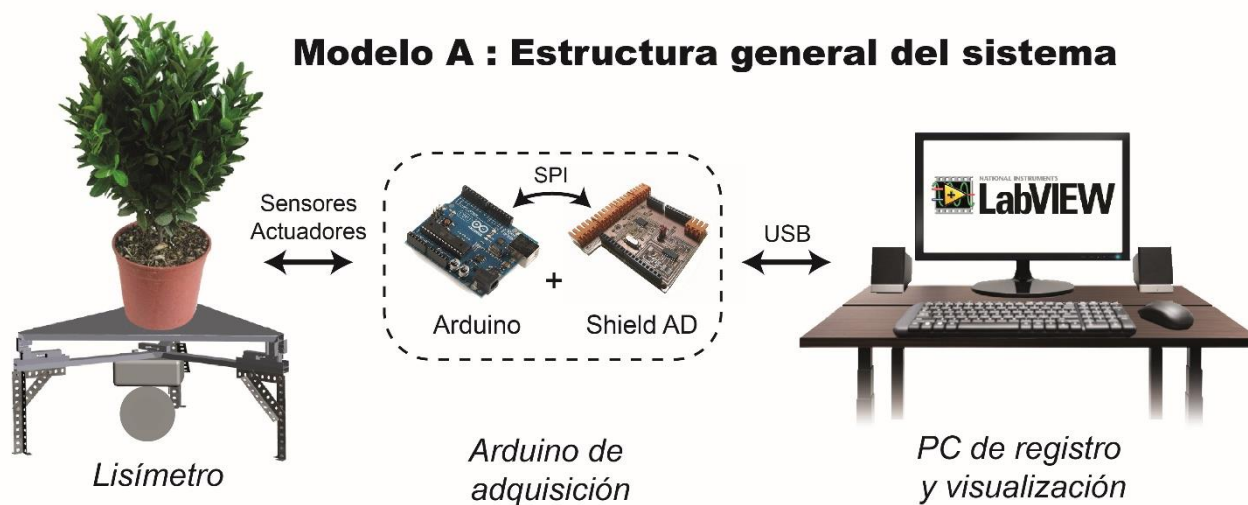


Figura 4.6: Estructura general del sistema (Modelo A)

Dentro del laboratorio, para poder suministrar luz de manera a la planta se ha usado una lámpara LED para cultivos de interior, lo que permite estimular los procesos fotosintéticos.

En la figura 4.7 a la derecha se encuentra el programador digital que se utilizó para tal efecto; a la izquierda, el programador digital Coati cuya función es la de programar los riegos (apartado 3.3.2).



Figura 4.7: Programadores

Conexión de pines

A continuación, en la Figura 4.8 se puede ver el conexionado de nuestro Arduino de Adquisición, formado por la placa Arduino Uno junto con el Shield para comunicación con el convertidor CS5534-ASZ. Como se puede observar, la comunicación del Arduino con los periféricos los hemos dividido en 4 categorías:

Con color Rojo tenemos, mediante el USB, la conexión entre el Arduino y la aplicación de LabView en el ordenador. Permite el flujo bidireccional de información.

El color Azul representa los pines destinados a interactuar con el Shield, y por tanto, con el convertidor analógico CS5534-ASZ. Utiliza protocolo SPI:

- Pin 13: SCK. Señal de reloj para generar los pulsos de tiempo
- Pin 12: MISO. Master In Slave Out, es el pin donde Arduino recibe los bytes provenientes del AD.
- Pin 11: MOSI. Master Out Slave In, corresponden al cable donde Arduino envía las ordenes al AD.
- Pin 10: CS. Chip Select, para habilitar la comunicación con el AD.

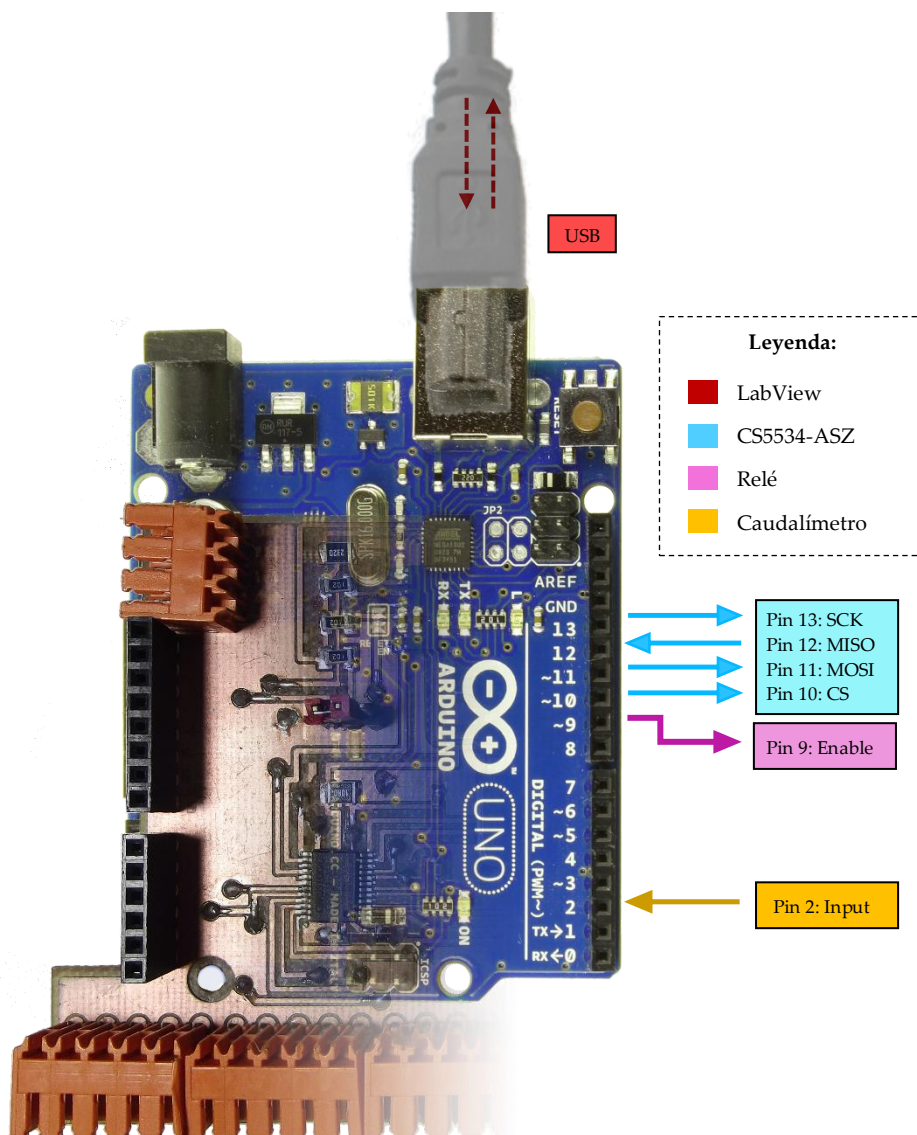


Figura 4.8: Conexión del Arduino de Adquisición en el Modelo A

Con color Morado el pin destinado al control del relé de las electroválvulas:

- Pin 9: Enable. Señal de disparo para habilitar o deshabilitar el relé

Con color Amarillo encontramos el cable proveniente del caudalímetro. Lo hemos conectado a uno de los dos puertos de Arduino que dispone de la categoría de "Interrupt".

- Pin 2: Input. Arduino recibe a través de este pin los pulsos del caudalímetro; a mayor número de pulsos por segundo, mayor caudal circulante.

Modelo B. Ensayo en finca (I)

Para poder evaluar el comportamiento del lisímetro al aire libre en la finca Tomas Ferro, se necesita sustituir la aplicación de LabView del PC como medio para almacenar y representar las medidas.

Para el Modelo B se ha utilizado un segundo Arduino, que recibe las medidas del Arduino de Adquisición y las almacena. Para dicho propósito, se ha utilizado para el segundo Arduino un Shield que incorpora el módulo HY-1.8 SPI que dispone de una pantalla TFT de 1.8 pulgadas de alta resolución, y una ranura para tarjeta SD. A este Shield se le ha añadido, además, un Real Time Clock modelo DS1302 para evitar desconfiguraciones de fecha/hora por caídas de tensión. Al conjunto de este segundo Arduino con el Shield TFT, SD y RTC lo llamaremos ahora Arduino de Registro.

En la figura 18 podemos observar la estructura general del sistema (Modelo B).

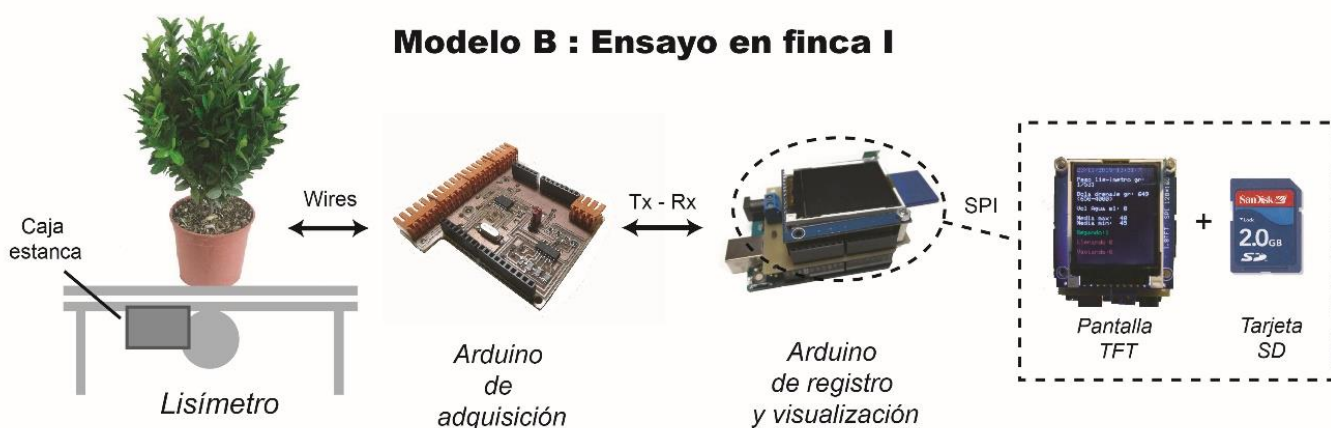


Figura 4.9: Estructura general del sistema (Modelo B)

Para organizar los componentes electrónicos y protegerlos de las condiciones externas (lluvia, polvo), se ha utilizado una caja estanca con protección IP65 (Figura 4.10).

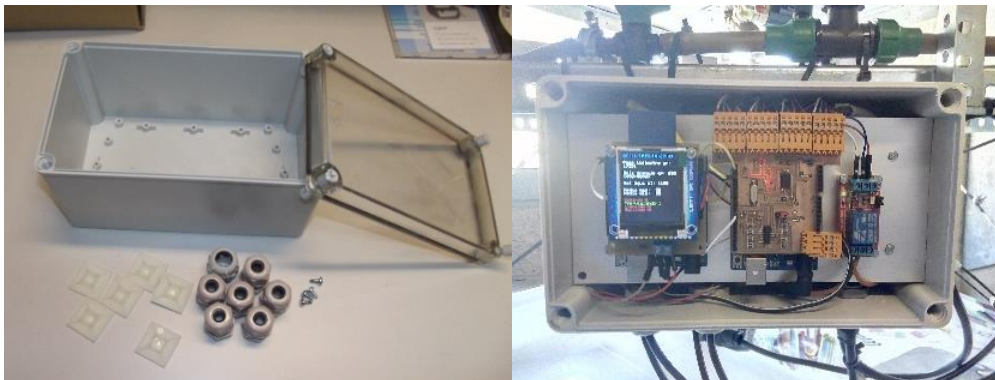


Figura 4.10: Caja estanca para la finca

Como se puede observar en la Figura 4.11, la caja se organizó en dos capas, superior e inferior. En la inferior se encuentran las entradas y salidas de los cables a la caja utilizando los prensas, además de las dos fuentes de alimentación (una los dos Arduino y demás sensores y actuadores, y otra independiente para el Shield CS5534-ASZ del Arduino de Adquisición).

En la capa superior como se observa, todos aquellos componentes electrónicos organizados para permitir su manipulación.

Esta caja, una vez montada, se acopló a la estructura del lisímetro mediante bridas, y lo mismo se hizo con el programador de riego (Figura 4.4).

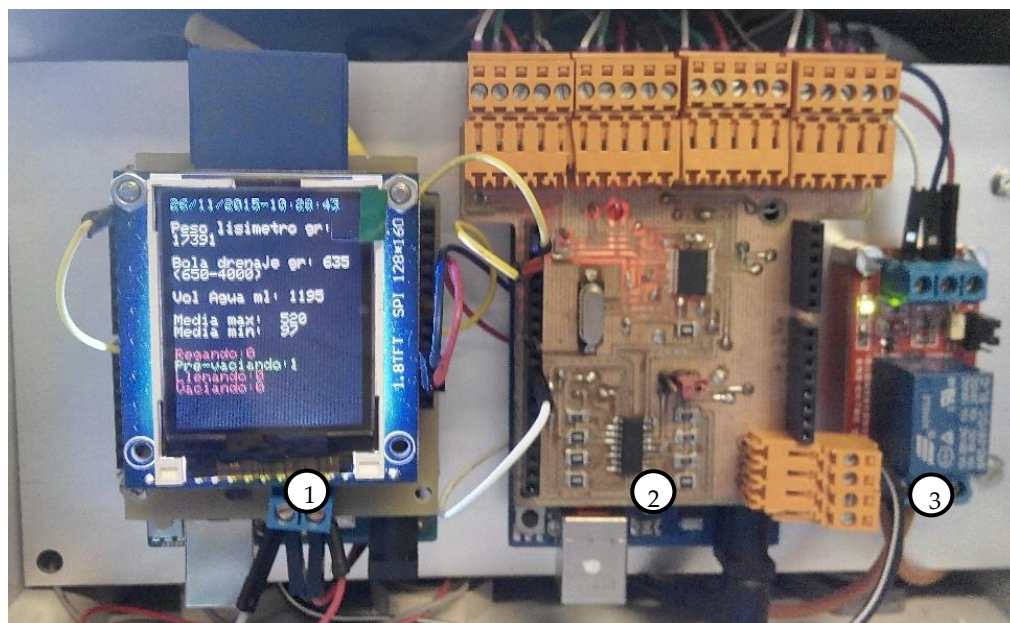


Figura 4.11: Arduino de Registro (1), Arduino de Adquisición (2) y relé de electroválvulas (3)

En la figura anterior se puede ver la distribución de los componentes dentro de la caja estanca; de izquierda a derecha se encuentran, el Arduino de Registro con su shield TFT + SD + RTC, después el Arduino de Adquisición con el shield del AD, y finalmente el relé para el control de las electroválvulas.

En la siguiente página, Figura 4.12, se puede ver el conexionado de nuestro Arduino de Adquisición (Arduino Uno junto con el Shield para el convertidor CS5534-ASZ) y el Arduino de Registro (Arduino Uno más el Shield con módulo HY-1.8 SPI y RTC).

Conexión de pines

Los pines del Arduino de Adquisición se describieron en el apartado anterior por lo que no se nombrarán, ya que no varía salvo que en lugar de cable USB para comunicarlo con un ordenador, se han usado los pines 0 y 1 para comunicarlo con el Arduino de Registro.

Como se puede observar, la comunicación del Arduino de Registro con los periféricos los hemos dividido en 3 categorías, verde, azul y rojo en función del dispositivo al que se conecta.

El color Verde representa los pines destinados a interactuar con módulo HY-1.8 SPI integrado en el shield. Utiliza protocolo SPI:

- Pin 13: SCK. Señal de reloj para generar los pulsos de tiempo. Para la pantalla TFT y para la ranura SD.
- Pin 12: MISO. Master In Slave Out, es el pin donde Arduino recibe los bytes provenientes de la SD.
- Pin 11: MOSI. Master Out Slave In, corresponden al cable donde Arduino envía las ordenes a la pantalla TFT y para la ranura SD.
- Pin 10: CS. Chip Select, para habilitar la comunicación de la pantalla TFT.
- Pin 9: DC. También llamado A0 o “Command data”, es el pin por el cual Arduino lee información que manda la pantalla TFT.
- Pin 8: Reset. Cable de reseteo de la pantalla TFT.
- Pin 4: CS. Chip Select, para habilitar la comunicación con la SD

Con el color Azul se encuentran los pines destinados al DS1302, mediante interfaz de tres hilos:

- Pin 5: Clock. Señal de reloj para generar los pulsos de tiempo
- Pin 3: Input/Output. Pin bidireccional para recibir datos o enviar instrucciones
- Pin 2: Chip Enable. Para habilitar la comunicación con el DS1302

El color Rojo se ha utilizado para la comunicación Serial entre los dos Arduinos:

- Pin 1: Tx. Pin de escritura o “text”, es un cable de salida de información.
- Pin 0: Rx. Pin de lectura o “read”, es el cable de entrada de información.

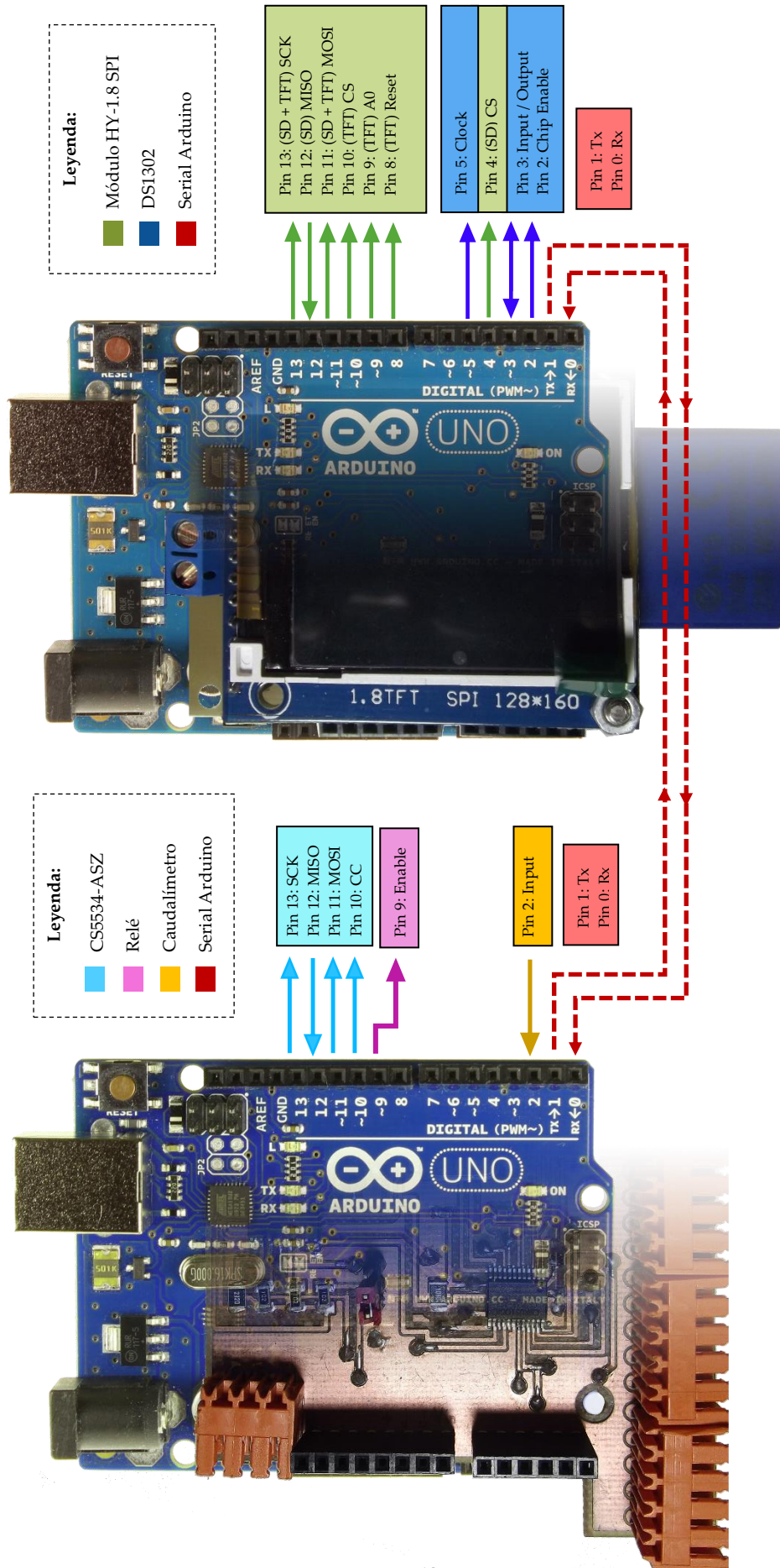


Figura 4.12: Conexión de Arduino de Adquisición (Izq) y Arduino de Registro (Der) en el Modelo B: Finca (I)

Modelo C. Ensayo en finca (II)

Básicamente, con este modelo se pretende mejorar el diseño anterior, ya que guardar las lecturas en la nube es una solución mucho más práctica que guardarlas en una tarjeta SD. Se ha sustituido el Shield al Arduino de Registro: en lugar del anterior Shield TFT + SD + RTC, se utilizará un Shield Ethernet con ranura microSD. De manera que a la vez que se suben las medidas a una base de datos en la nube, permite almacenarlas también en la microSD para evitar perder datos en caso de fallo de conexión. Ya no se dispone por tanto de la RTC para configurar la hora, pero al tener acceso a internet, se ha implementado en el código que cada aproximadamente 10 minutos actualice la hora utilizando internet.

El departamento de Tecnología Electrónica dispuso un router T-Link y una tarjeta SIM para obtener acceso a Internet en la finca de manera remota. El router se instaló en una caja estanca dentro de un invernadero cercano. Se hizo llevar un cable RJ45 para la conexión del Shield Ethernet a internet.

A continuación podemos ver la Estructura general del sistema para nuestro nuevo Modelo C:

Modelo C : Estructura general del sistema

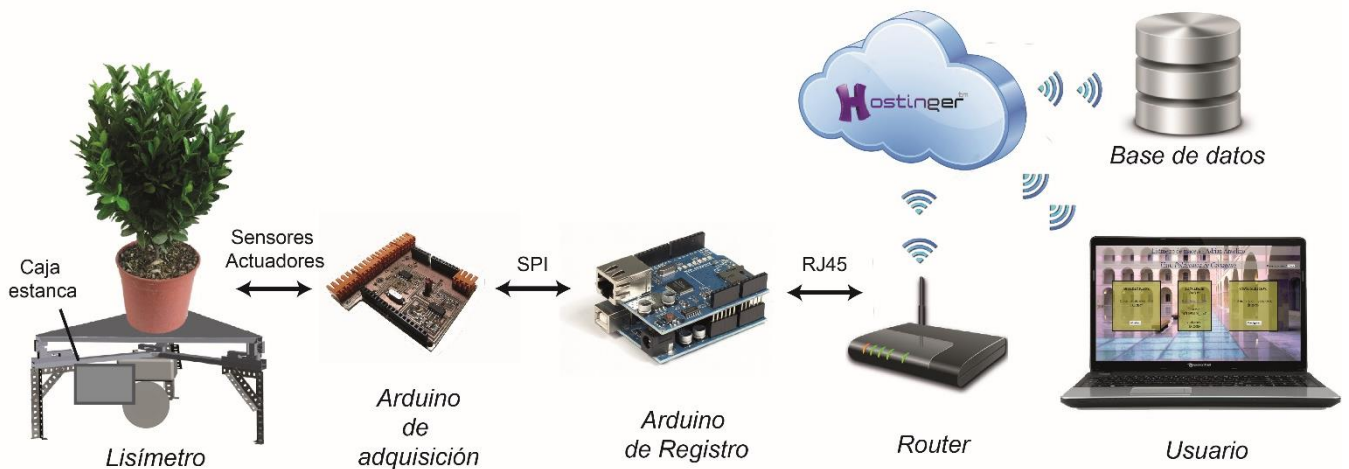


Figura 4.13: Estructura general del sistema (Modelo C)

Por tanto, ahora el Arduino de Registro será el segundo Arduino con Shield Ethernet. Este recibe las lecturas del Nodo de Registro, y las almacena en una base de datos alojada en la Nube. Para cumplir este objetivo, se ha diseñado una página web llamada **www.lisimetro-upct.hol.es** que se ha alojado en Hostinger, el cual nos brinda de manera gratuita un espaciado virtual de hosting cediéndonos un subdominio gratuito. En

nuestra página web se han utilizado scripts basados en PHP y creado una base de datos en MySQL para almacenar las lecturas del lisímetro. Esta página web será accesible desde cualquier dispositivo remoto con conexión a internet, ya sean ordenadores, tablets o smartphones.

Conexión de pines

En la siguiente página, Figura 4.14, se puede ver el conexionado de nuestro Arduino de Adquisición y el Arduino de Registro (formado por una placa Arduino Uno más el Shield Ethernet). Los pines del Arduino de Adquisición no se volverán a nombrar, ya que se profundizó en el apartado anterior.

La comunicación del Arduino de Registro con los periféricos se ha dividido en 3 categorías, morado, verde y rojo, en función del dispositivo al que se conecta.

Con color Morado tenemos, mediante el un cable RJ45, la conexión a Internet del Shield Ethernet, y por tanto del Arduino. El cable RJ45 está conectado a un Router con conexión a internet.

El color Verde representa los pines destinados a interactuar con Shield Ethernet y la ranura microSD. Utiliza protocolo SPI:

- Pin 13: SCK. Señal de reloj para generar los pulsos de tiempo. Ethernet y ranura SD.
- Pin 12: MISO. Master In Slave Out, es el pin donde Arduino recibe los bytes provenientes del Ethernet y ranura SD.
- Pin 11: MOSI. Master Out Slave In, corresponden al cable donde Arduino envía las ordenes al Ethernet y ranura SD.
- Pin 10: CS. Chip Select, para habilitar la comunicación con el Ethernet.
- Pin 4: CS. Chip Select, para habilitar la comunicación con la SD

El color Rojo se ha utilizado para la comunicación Serial entre los dos Arduinos:

- Pin 1: Tx. Pin de escritura o "text", es un cable de salida de información.
- Pin 0: Rx. Pin de lectura o "read", es el cable de entrada de información.

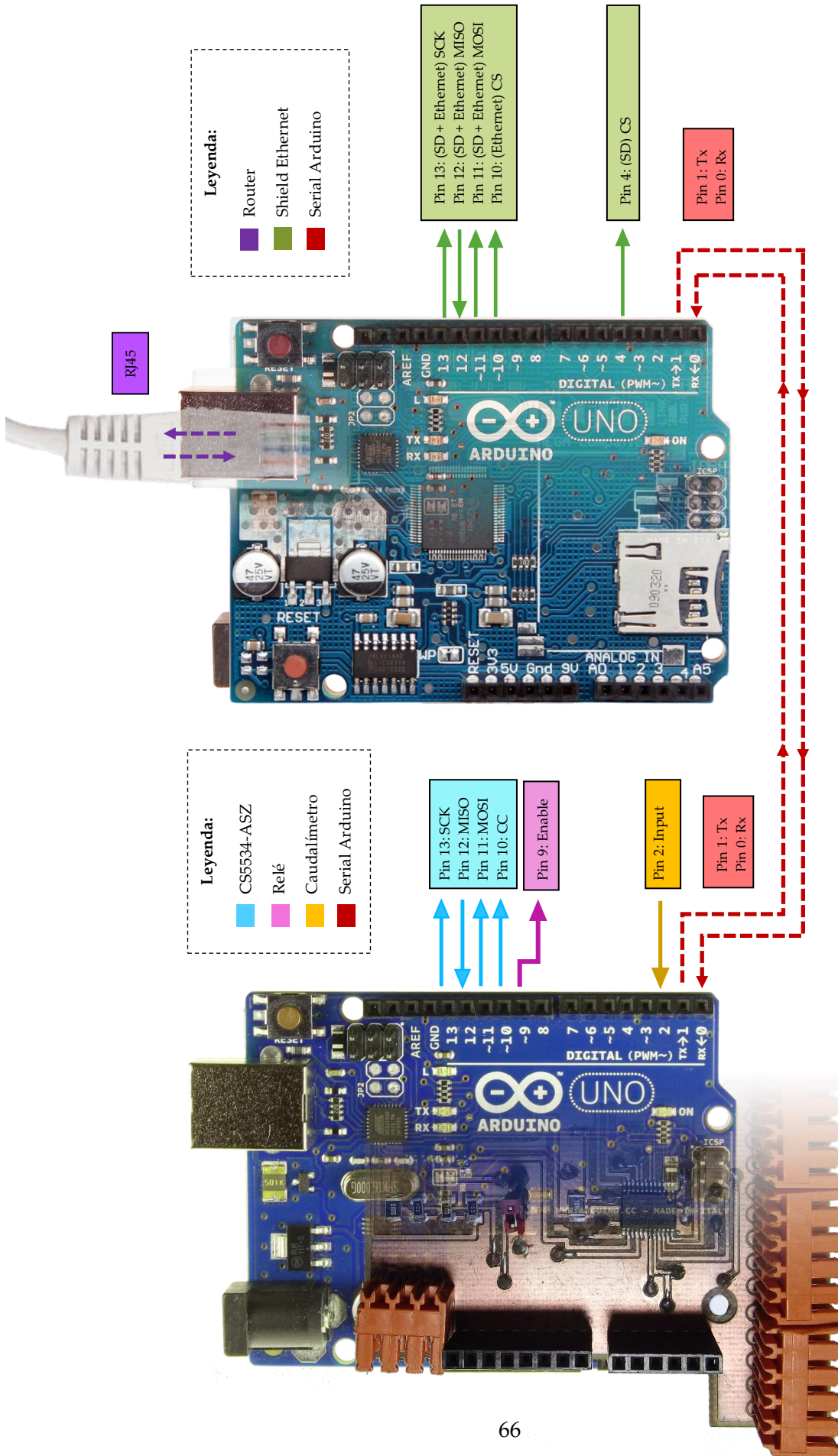


Figura 4.14: Conexión de Arduino de Adquisición (Izq) y Arduino de Registro (Der) en el Modelo C: Finca (II)



Capítulo 5.

Software desarrollado

En este quinto capítulo se expone el software desarrollado para las diferentes configuraciones del sistema, tanto para laboratorio como para finca.

5.1 Configuración CS5534-ASZ

Durante el desarrollo del TFT, el Arduino de Adquisición no varía a lo largo de las diferentes configuraciones (Modelos A, B y C), tan solo cambia el modo en que se comunica con el Nodo de Registro. Por tanto, su código fuente apenas cambia. En este apartado, se pretende mostrar cómo se ha comunicado el Arduino con el convertidor CS5534-ASZ. En el ANEXO I se pueden observar las funciones creadas en el código del Arduino para comunicarse con el CS5534-ASZ.

Introducción al convertidor CS5534-ASZ

La estructura de registros interna del CS5534 se encuentra en la Figura 5.1. En ella se muestra el diagrama de bloques de como el CS5534 incluye una serie de registros internos a los que el usuario puede acceder:

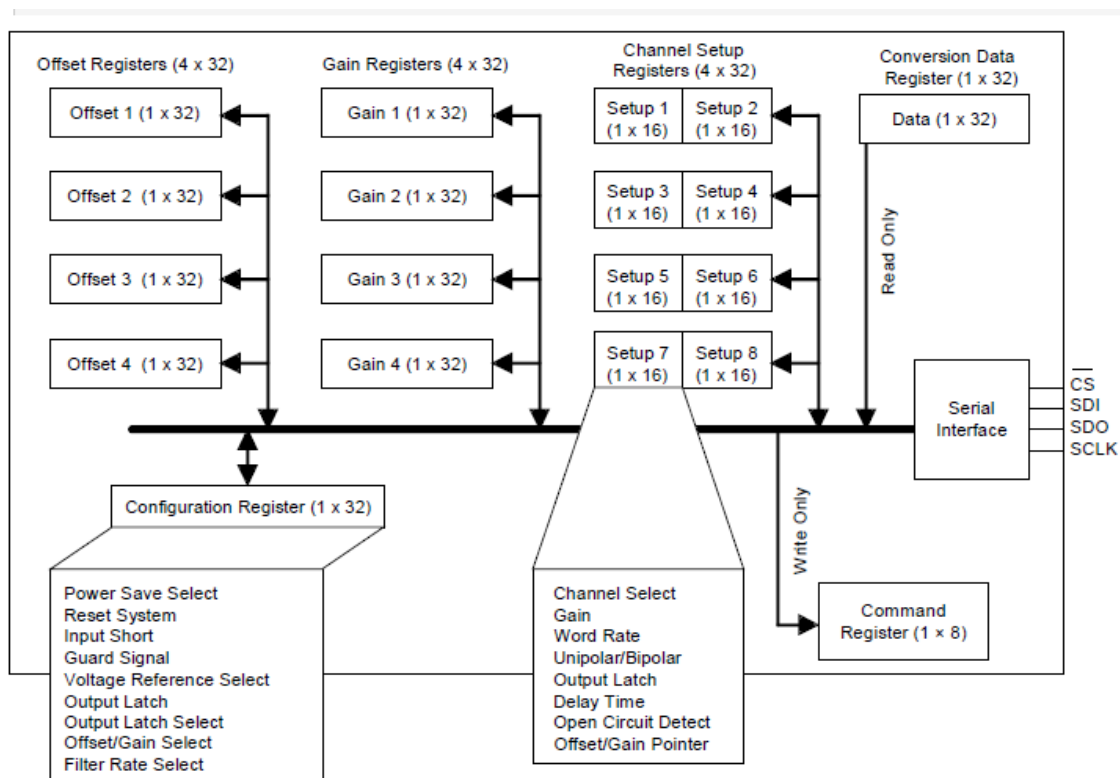


Figura 5.1: Diagrama de registros del CS5534-ASZ (Datasheet)

Estos registros se utilizan para guardar los resultados de las calibraciones (offset y ganancia), configurar diferentes modos de operación, así como los diferentes parámetros e instrucciones. El CS5534 cuenta con:

- 4 registros (de 32 bits cada uno) de calibración de Offset
- 4 registros (de 32 bits cada uno) de calibración de Ganancia
- 8 registros de Canal (16 bits cada uno) para almacenar instrucciones de conversión de diferentes canales para diferentes parámetros (ganancia, velocidad de trabajo, modo Unipolar / Bipolar, ...)
- 1 registros de configuración (32 bits) para definir las variables generales, como modo de alimentación, apagado y encendido, modo de reseteo y habilitar la señal de guarda.
- Registro de conversión de datos (32 bits). Sirve para tomar lecturas
- Registro de comandos (8 bits). Para mandar una instrucción

Para poder interactuar con el CS5534 se necesita mandar un comando de 8 bits. Después, si estamos en modo escritura (Figura 5.2) mandaremos 32 bits con la información que queremos registrar:

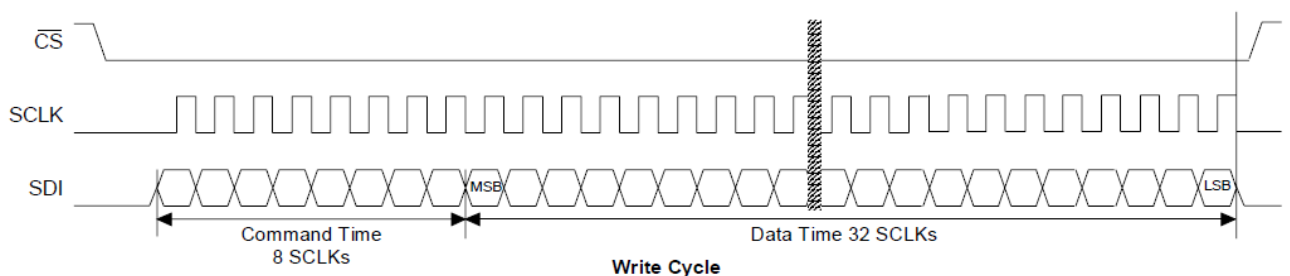


Figura 5.2: Ciclo de escritura, CS5534 (Datasheet)

Si lo que queremos es leer, mandamos el comando pertinente y a continuación recibiremos una cadena de 32 bits con la información requerida:

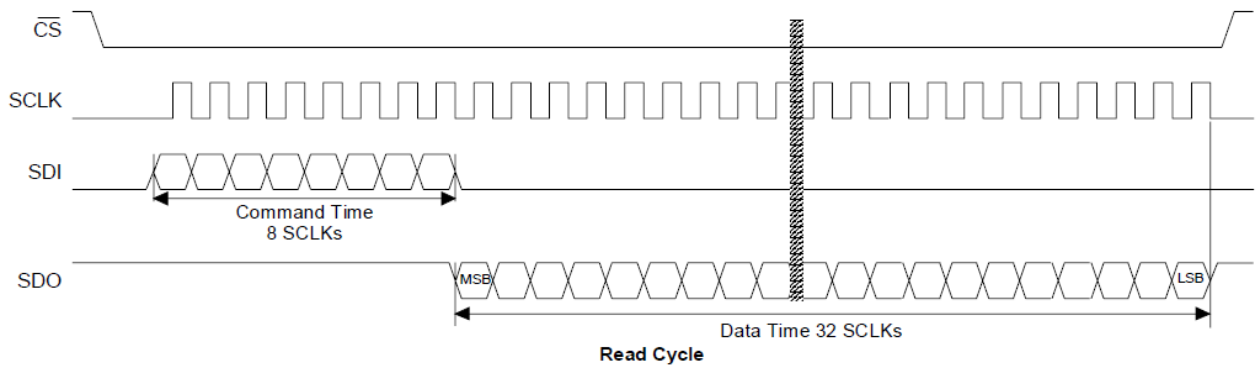


Figura 5.3: Ciclo de lectura, CS5534 (Datasheet)

Para empezar a trabajar con el convertidor CS5534, se comenzó realizando pequeñas pruebas en el laboratorio para comprender su funcionamiento: tipo de comunicación, funciones principales, configuraciones.

Utilizando la hoja de características del AD, se diseñó con Arduino un programa que permitiese interactuar con el AD y explorar sus diferentes características. Se muestra con detalle en el [ANEXO II](#). El menú inicial para navegar es el mostrado a continuación:

```

|MAIN MENU|
1) Leer AD_Setup           5) Initialize AD
2) Read/Write Registers    6) Reset AD
3) Perform Calibration     7) COMMAND + Read_DATA
4) Read Global water volume 8) COMMAND + Write_DATA

o) Open Valve
c) Close Valve
    
```

Figura 5.4: Menú principal. Navegación por el CS5534-ASZ

Configuración de los canales

Después de aprender a utilizar correctamente el convertidor AD mediante el programa anterior, se configuraron de manera permanente los parámetros del CS5534-ASZ. Cada una de las células de carga, utiliza un canal físico de los cuatro posibles. Cada canal físico utiliza su propia configuración:

- Canal físico 1. Channel Setup 1.
- Canal físico 2. Channel Setup 2
- Canal físico 3. Channel Setup 3
- Canal físico 4. Channel Setup 4

Para los cuatro canales, se ha utilizado finalmente una Ganancia de 64x, alimentación en configuración Unipolar a 50Hz, señal de guarda activa (de las células de carga) y velocidad de muestro de 6.25 muestras por segundo.

Cadena de bytes

El Arduino de Adquisición, durante todas las configuraciones, se ha comunicado con el Nodo de Registro enviado a través del puerto Serial una cadena en formato binario en lugar de utilizar caracteres. Esto es debido a que, al utilizar *Serial.print()* se envía en formato ASCII, lo que a la larga puede generar problemas de malinterpretación por parte del dispositivo que lo recibe, ya sea porque el valor recibido coincide con un retorno de carro o con un elemento separador de variables. De manera que se ha hecho uso de *Serial.write()*.

Para variables pequeñas de 1 solo byte no hay problema (como las booleanas), se utiliza directamente:

```
Serial.write(regando);
```

El resto de variables de mayor tamaño, 32 bits (unsigned long) se ha transmitido a través de 4 bytes. Para éstas, se ha diseñado la siguiente función para enviarlas haciendo uso del *Serial.write()*:

```
void enviar(unsigned long num)
{ byte resultado[ ]={0,0,0,0};

    for(int i = 3; i!=-1; i--)
        {resultado[i]= num;
          num = num >>8; }

    for(int i = 0; i<4; i++) { Serial.write(resultado[i]); }
}
```

De esta manera mediante *Serial.write* se ha podido enviar variables de más de 1 byte. Para su recepción en el Arduino de Registro para la finca (I) se ha utilizado un procedimiento inverso.

5.2 Flujoograma del Arduino de Adquisición

A continuación se muestra el algoritmo interno del lisímetro (Figura 5.5) perteneciente al Arduino de Adquisición, encargado de tomar las medidas de peso, manejar las electroválvulas y enviar la información para su guardado y procesado. Éste es el resultado final, resultado de múltiples ajustes y modificaciones.

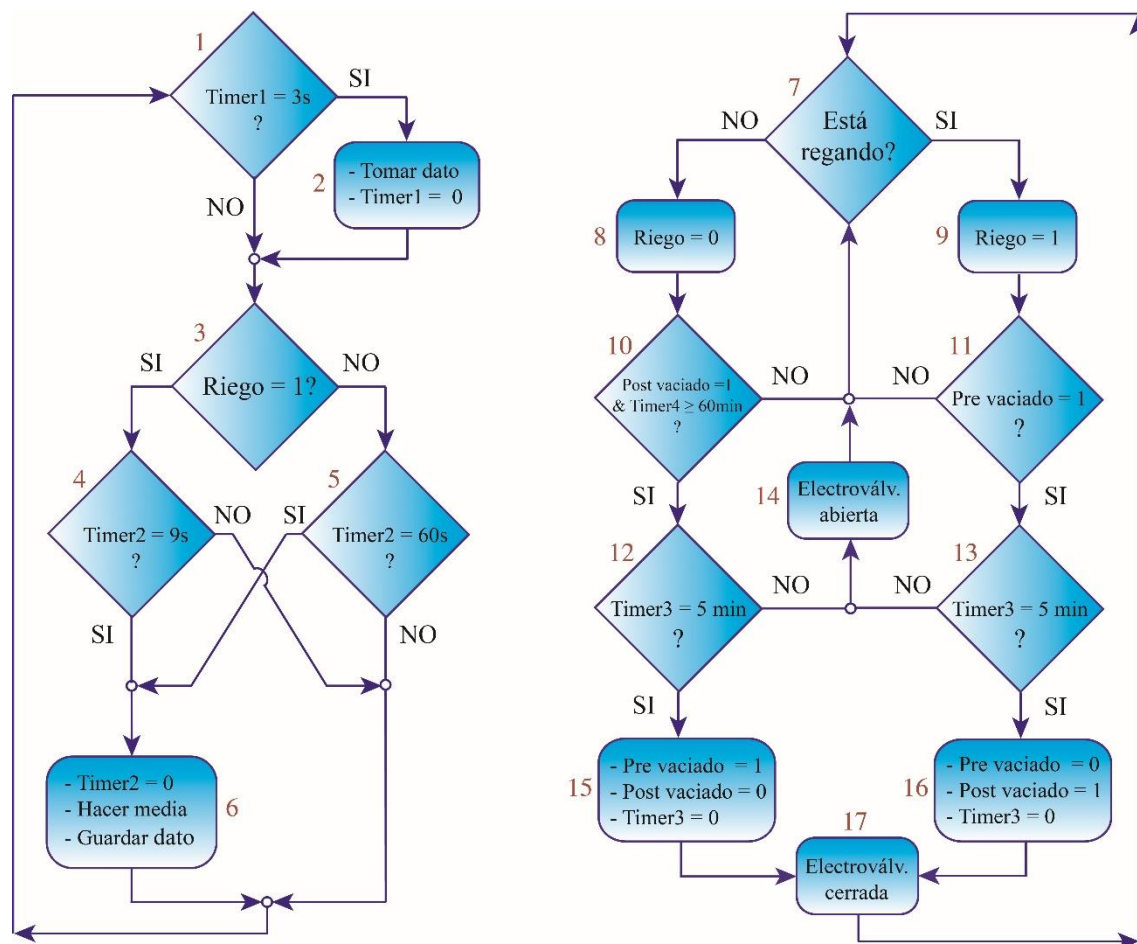


Figura 5.5: Arduino de Adquisición. Flujoograma de toma de datos (Izq) y electroválvulas (Der)

Toma de datos

A la izquierda, se encuentra el algoritmo de toma de datos. (1)(2) El AD toma una medida de peso de la bola y del peso total cada 3 segundos con el Timer1 (esto incluye el filtrado de datos, si un resultado es erróneo no se registra).

(3)(4)(6) Si el lisímetro se encuentra en Modo Riego (Riego = 1), entonces los datos se almacenarán cada 9 segundos para poder registrar con mayor precisión la evolución

durante el regado (Timer2 = 6), realizando la media aritmética de los datos registrados hasta el momento.

(5) Si por el contrario, se encuentra en Modo Normal (Riego = 0), pasará a almacenar los datos minutalmente (Timer2 = 60).

Electroválvulas

A la derecha, se encuentra el algoritmo de las electroválvulas y el caudalímetro. (7)(9) Si el caudalímetro registra un riego, se configurará el lisímetro en Modo Riego (Riego = 1). (11)(13) Si el riego acaba de iniciarse, pre vaciará la bola de drenaje (Pre vaciado = 1) por si se encontrase llena. (15)(17) Tras 5 minutos (tiempo determinado en el laboratorio para su completo vaciado), las electroválvulas se cerrarán de nuevo (Timer3 = 5).

(8)(10)(12) Una vez terminado el riego, pasará a Modo Normal (Riego = 0), y pasada una hora (Timer4 = 60), cuando ya se haya producido todo el lixiviado, entonces vaciará la bola de drenaje (12)(15)(17).

5.3 Modelo A: Ensayo en el laboratorio. LabView

Introducción

Para desarrollar la interfaz en PC, se ha optado por utilizar NI LabView, que es un lenguaje de programación gráfico cuyo principal objetivo es desarrollar aplicaciones de monitorización y control, pruebas y simulaciones (siglas de Laboratory Virtual Instrumentation Engineering Workbench). Salió al mercado en 1986 por National Instruments, para la plataforma MAC, aunque actualmente también funciona en Linux, UNIX o Windows.

El lenguaje que utiliza LabView es gráfico (más conocido como lenguaje G), lo que permite realizar programas de relativa dificultad con unos conocimientos de manejo básicos, siendo accesible para cualquier programador (profesional o amateur). Lo que significa, que es un lenguaje basado en diagramas de flujo de datos, usando símbolos para construir aplicaciones en lugar de los típicos lenguajes.

Dentro del programa, tenemos el panel frontal donde crear nuestros controles e indicadores gráficos (este está el panel del usuario, la interfaz con la que el usuario trabaja) y después el panel trasero donde está el diagrama de bloques (contiene el código fuente del programa). Todos aquellos objetos que situemos en el panel frontal tendrán su correspondiente en el panel trasero, con sus terminales de conexión correspondientes:

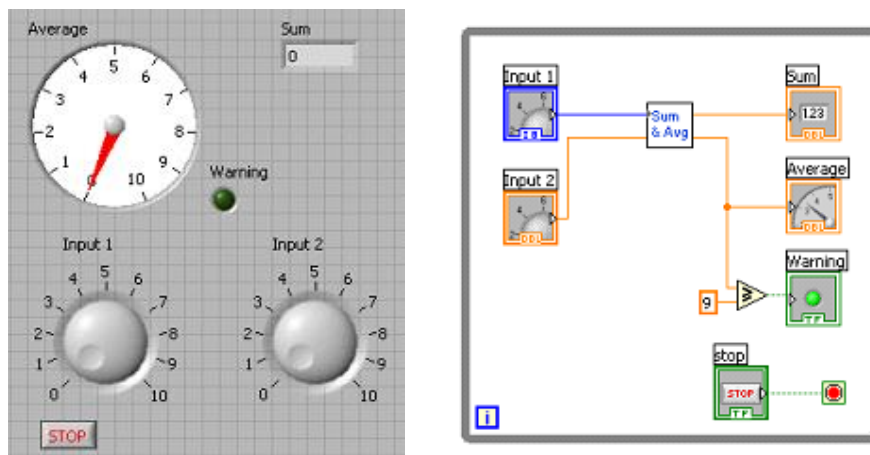


Figura 5.6: Ejemplo de Panel frontal (Izq) y trasero (Der)

Mediante los diagramas de bloques, se pueden sustituir cientos de líneas de código en el lenguaje convencional, ahorrándose muchísimo trabajo y esfuerzo. Los programas generados por LabView se llaman Instrumentos Virtuales o “VIs”, y permiten combinarse con otros nuevos Vis o unos ya creados anteriormente.

Resumiendo, las ventajas de usar LabView son:

- Es muy sencillo e intuitivo, reduce de 4 a 10 veces el tiempo requerido para desarrollar aplicaciones
- Es un sistema flexible, se pueden realizar actualizaciones y cambios de hardware y software
- Los usuarios pueden crear soluciones complejas y completas
- El sistema de desarrollo integra todas las funciones: adquisición, análisis y representación.
- Utiliza un compilador gráfico para alcanzar mayor velocidad de ejecución.
- Se pueden incorporar aplicaciones realizadas en otros lenguajes

Front-end

Es la interfaz gráfica del usuario, en la que se muestran los controles para ingresar los parámetros (variables de entrada), e indicadores para representar las salidas de información (variables de salida), la información generada por el programa. Simula el panel de instrumentación físico, donde podemos interactuar e introducir datos e información mediante el teclado o ratón.

Aquí podemos ver el panel frontal inicial del programa (Figura 5.7).

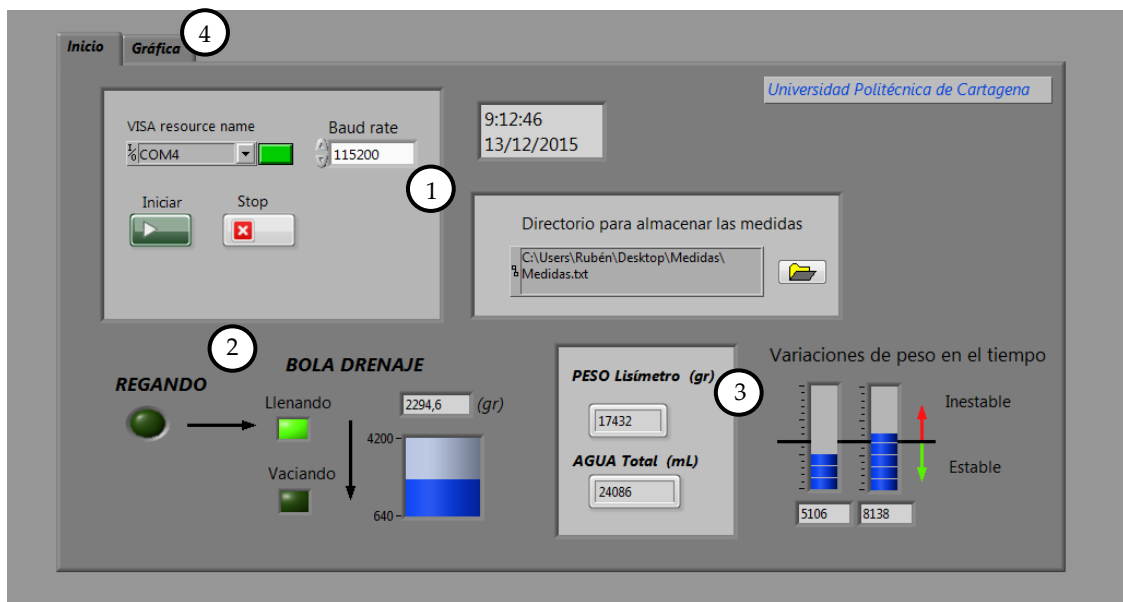


Figura 5.7: Panel frontal de LabView. Inicio

Cómo podemos observar, el panel frontal se ha dividido en varios sectores:

- 1) Configuración:

El usuario puede seleccionar el puerto COM, variar el Baud-Rate, iniciar y detener el programa, así como el directorio seleccionar el destino de guardado de medidas. También se muestra la fecha y hora.
- 2) Etapa del programa y Bola de drenaje:

Muestra mediante luces Led el momento del programa en que se encuentra el lisímetro. A su vez, muestra el valor numérico mediante indicadores
- 3) Peso del lisímetro y agua:

El valor en gramos del peso total, así como el volumen de agua regada registrada. También se incluyeron unas variables de variación de peso temporal que se utilizó en el comienzo, para detectar incrementos de peso en la bola de drenaje debidos al lixiviado y discriminarlos de variaciones debido a razones no definidas (viento, acción humana).
- 4) Gráfica (Figura 5.8):

Para poder evaluar y monitorizar el comportamiento a tiempo real de:

 - Peso del lisímetro (Color Blanco)
 - Peso de la bola de drenaje (Color Rojo)

Permite la navegación a través de la gráfica, así como configurar algunos parámetros de representación.

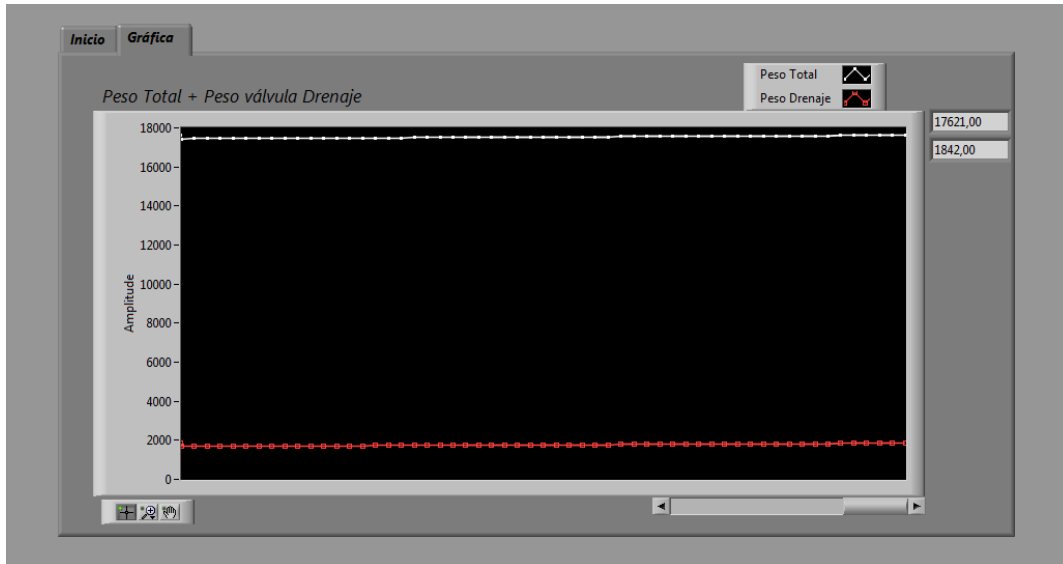


Figura 5.8: Panel frontal de LabView. Gráfica

Back-end

En el panel trasero de LabView es donde se encuentra el código fuente, el diagrama de bloques. Aquí se gestiona el flujo de entradas y salidas creadas en el panel frontal o interfaz. El diagrama de bloques se crea conectando, como en un circuito, los objetos entre sí mediante cableado de datos, definiendo los terminales de entrada y salida. Los cables tienen colores diferentes y estilos de representación en función de su contenido.

LabView permite hacer subprogramas o integrar otros programas ya que su estructura es modular y de tipo hereditaria, creando o utilizando diferentes "subVI" dentro de un mismo "VI". Mediante esta programación modular, una aplicación compleja puede estar dividida en una serie de subtareas sencillas. Una de las ventajas de LabView es que tiene una biblioteca con todo tipo de funciones; para realizar operaciones aritméticas, comparaciones, análisis,...

A continuación podemos ver el diagrama de bloques del programa. Está dividido en varios módulos:

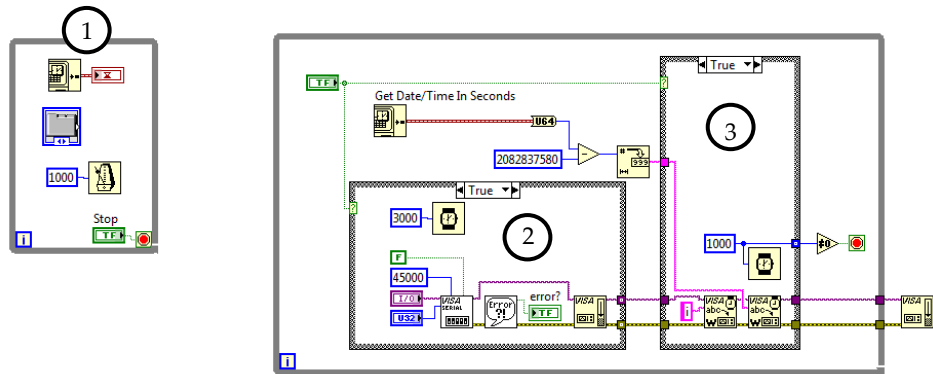


Figura 5.9: Diagrama de bloques (I)

- Nodo 1: Bucle loop encargado de que, independientemente de que el programa esté corriendo o no, se pueda visualizar la hora y fecha así como navegar en las ventanas Inicio / Gráfica.
- Nodo 2: Bucle loop cuya función es la de, una vez pulsado el botón de inicio, inicia una sesión VISA para poder acceder al puerto serial mediante LabView.
- Nodo 3: Una vez pulsado inicio, su función es comunicar al Arduino el inicio del programa (letra "i") para arrancar el programa.

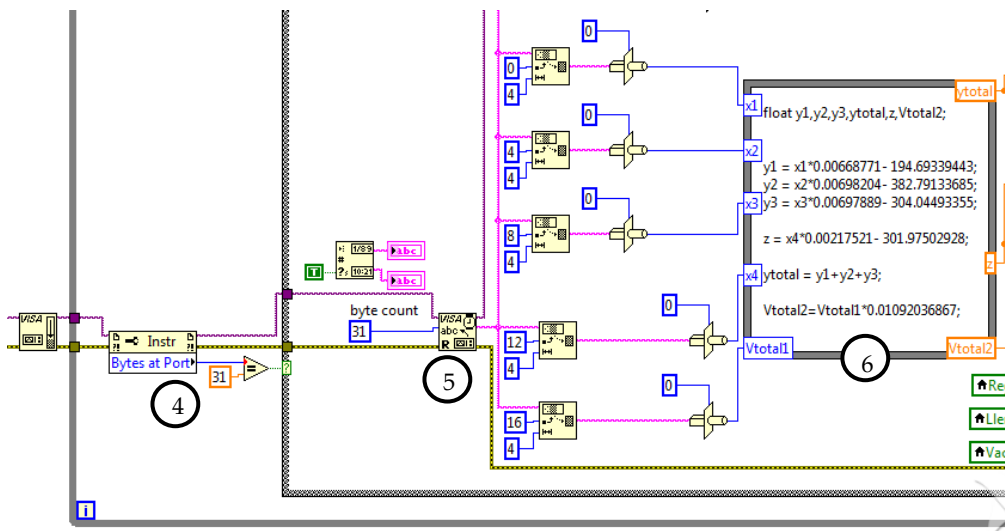


Figura 5.10: Diagrama de bloques (II)

- Nodo 4: Encargado de monitorizar el número de bytes en el puerto serie. Cada lectura del lisímetro implica 31 bytes: 6 variables de 4 bytes (unsigned long) para las medidas de peso/agua y 3 de 1 bytes para las booleanas.
- Nodo 5: Este SubVI se encarga de leer la trama de 31 bytes que envía Arduino, y dividir la cadena de letras en módulos de bytes para identificar cada variable, convirtiéndolas en variables numéricas o booleanas. LabView, por tanto, recibe una cadena de bytes. LabView permite dividir la cadena de ceros y unos en módulos de bytes y separar todas las variables de manera independiente en Substrings, y estas tramas convertirlas a variables de tipo numérico.
- Nodo 6: Nodo cuyo objetivo es el de obtener los valores de peso de las 4 galgas (x1, x2, x3, x4) y la lectura del caudalímetro (Vtotal2) y mediante las rectas de calibración, obtener el peso total del lisímetro (ytotal) en gramos, peso de la bola de drenaje (z) en gramos y agua regada (Vtotal2) en litros.

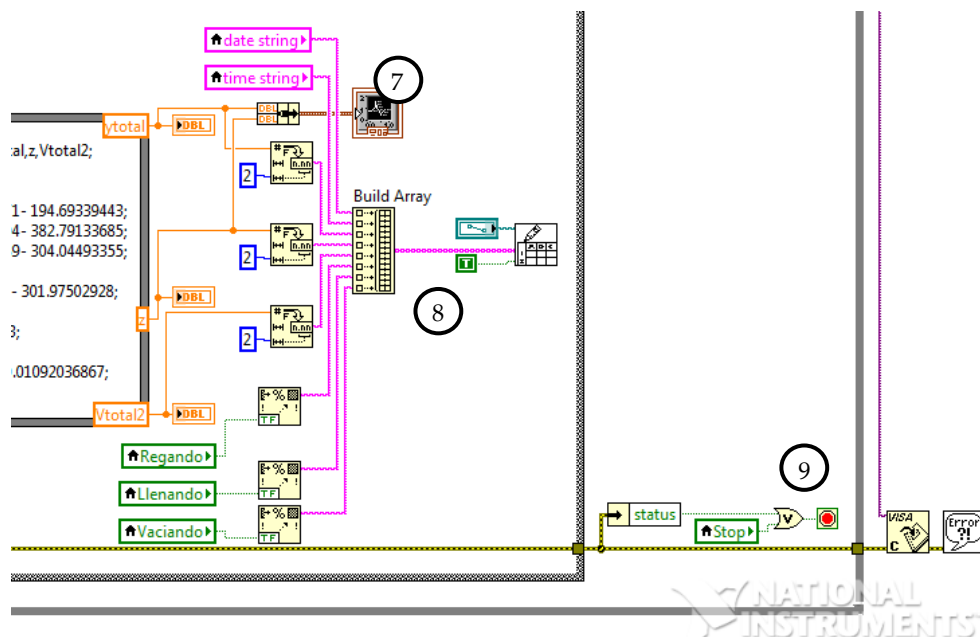


Figura 5.11: Diagrama de bloques (III)

- Nodo 7: Función encargada de graficar las variables peso del lisímetro (ytotal) y bola de drenaje (z).
- Nodo 8: Nodo de guardado. La función Build array genera un array de datos con todas las variables, y después se almacenan los datos en una hoja de datos en formato tabla.

Nodo 9: Nodo de finalización. Cierra la sesión VISA para liberar el puerto. Permite el cierre del programa de manera manual o en caso de error en la comunicación.

El apartado que pone “variaciones de peso” fue eliminado posteriormente. Era una variable interna que determinaba la variación del peso de la bola en el tiempo. De esta manera, cuando la bola comenzaba a llenarse de agua, Arduino lo reconocía, y una vez que se estabilizaba en peso en el tiempo (dejaba de caer agua) entonces vaciaba la bola. Finalmente, por sencillez, se determinó que se vaciaría la bola pasada una hora después del riego, tiempo más que suficiente para lixiviar toda el agua sobrante.

Otras aplicaciones

En este apartado, para poder calibrar las células de manera precisa, se diseñó un programa de **LabView** específicamente para ello (Figura 5.12). Las rectas de calibración se han obtenido a partir de hasta 1000 puntos, de manera que aumenta su precisión y nuestro R^2 es 1 en todos los casos.

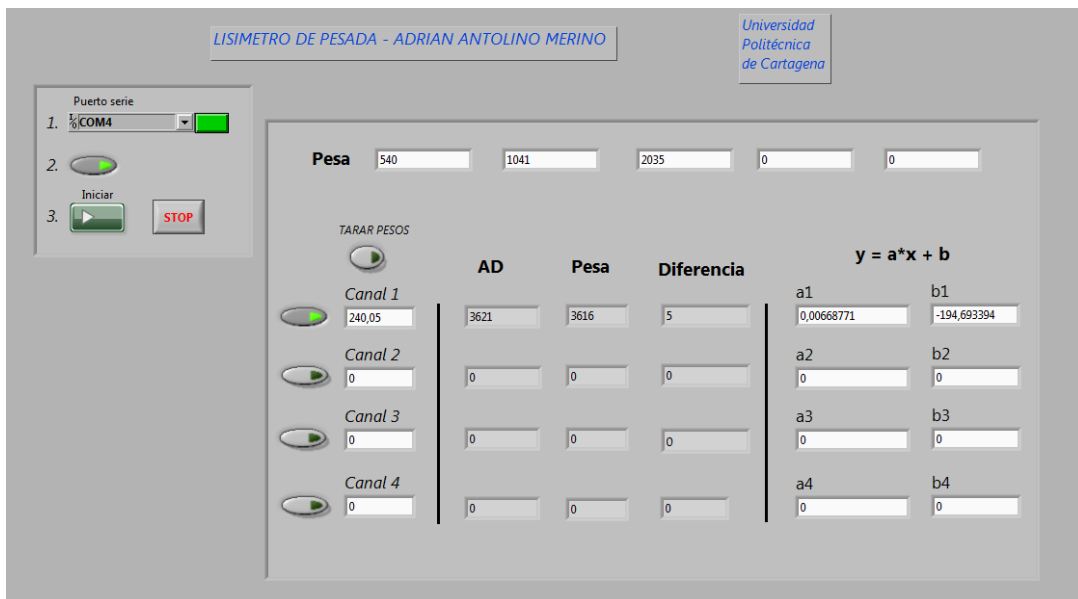


Figura 5.12: Aplicación de LabView para calibración de células

Además, durante los ensayos de laboratorio, se diseñó otra aplicación para realizar el riego programado a través de Arduino (Figura 5.13). Este permitía programar de manera externa (programador de riego) o interna (mediante riegos programables, hasta 3 diferentes). Además de todo lo anterior, incluía el control manual del lisímetro para apertura de electroválvulas.

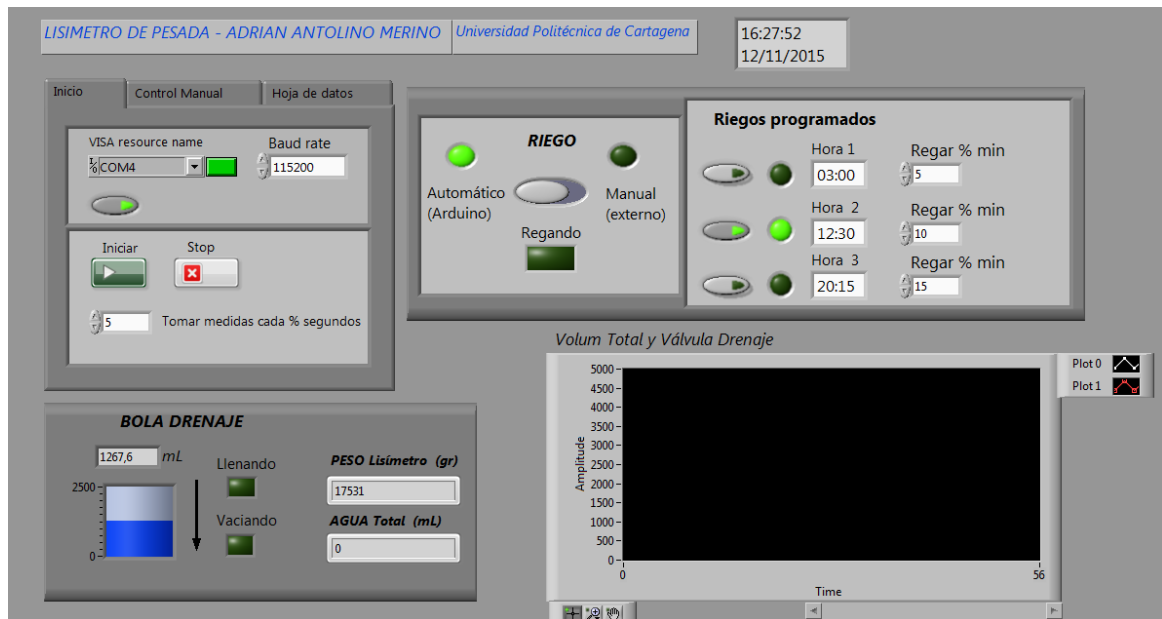


Figura 5.13: Aplicación de LabView con riego programable

Más tarde se utilizó un programador de riego externo por lo finalmente se utilizó la aplicación mencionada al comienzo del capítulo.

5.4 Modelo B: Ensayo en finca (I). Shield HY-1.8 SPI con DS1302

Software

Una vez hecha la primera prueba en el laboratorio, se comenzó a desarrollar el sistema de adquisición para la finca. El Arduino que se encargará del registro de las lecturas utiliza para ello el shield que incluye el módulo HY-1.8 SPI y un Real Time Clock.

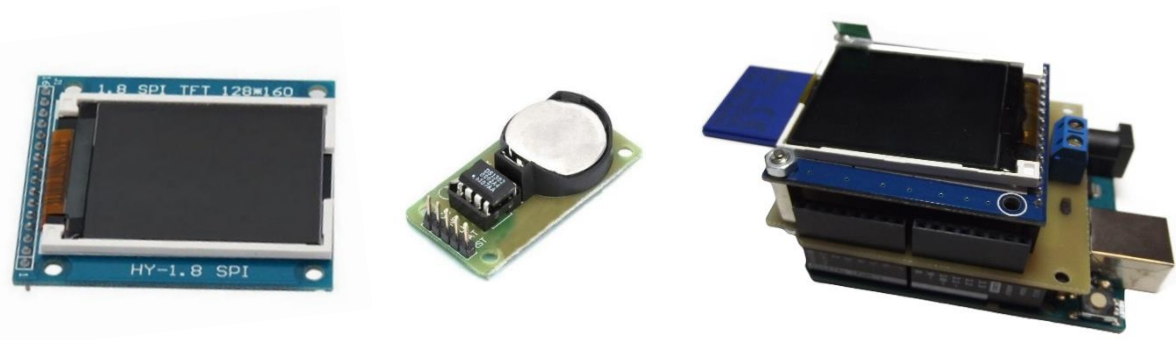


Figura 5.14: Módulo HY- 1.8 SPI (Izq), DS1302 (Med) y Arduino de Registro y Visualización (Der)

El módulo HY-1.8 SPI tiene una pantalla TFT a color de 1.8 pulgadas (128x160 píxeles). La ranura y la pantalla funcionan de manera independiente, y se comunican con el Arduino mediante protocolo SPI. Utiliza por tanto 2 librerías, que se han descargado de la página web Github:

- La librería gráfica `Adafruit_GFX.h`, que ahorra mucho trabajo ya que además de imprimir textos, proporciona comandos básicos para representar cualquier punto, líneas, círculos,... en un display.
- La librería `Adafruit_ST7735.h` utilizar el ST7735, el chip controlador para pantallas gráficas de tipo TFT-LCD

Para poder utilizar una tarjeta SD, hay que formatear la tarjeta en formato FAT16 o FAT32. Para usarlo usamos:

- La librería `SD.h` incorpora una serie de funciones de escritura y lectura de ficheros básicas, además de algunas otras funciones para el control de la tarjeta como el espacio disponible, tipo de formato, y creación y eliminación de ficheros.

El RTC modelo DS1302 también utiliza su propia librería para utilizarla:

- Librería DS1302.h junto con Time.h, incluyen funciones para determinar la hora y fecha con segundos, minutos, horas, días, mes y año, además de proporcionarlo en formato estándar C `time_t` que permite calcular transcurros de tiempo de manera muy sencilla en diferentes plataformas.

Para las dos primeras, necesitamos también la librería SPI.h que por defecto configura los pines 13, 12 y 11 de Arduino como Clock, MISO y MOSI respectivamente.

Los datos entre Arduinos se han enviado en formato binario mediante "`Serial.write()`", menos un retorno de carro enviado al comienzo del mensaje para comenzar la comunicación.

- Para utilizar el RTC, necesitamos configurar sus pines.

```
#include <DS1302.h>

const int kCePin = 2; // Chip Enable
const int kloPin = 3; // Input/Output
const int kSclkPin = 5; // Serial Clock

DS1302 rtc(kCePin, kloPin, kSclkPin);
```

Seguidamente, para obtener el tiempo utilizamos la siguiente instancia.

```
Time t = rtc.time();
```

- Para utilizar la tarjeta SD, creamos un nuevo objeto tipo File.

```
#include <SD.h>
File myFile;
```

Y para inicializarla necesitamos el comando `Begin` con el pin configurado para `chip select` de la SD.

```
SD.begin(4);
```

Después, para guardar los datos en la tarjeta SD usamos `escritura_sd(t)`, una función que hemos creado para tal efecto. Primeramente, abre el archivo de nombre "MEDIDAS.txt" contenido en la tarjeta SD para proceder a su escritura, o bien crea uno nuevo si no existe el archivo.

```
myFile = SD.open("MEDIDAS.txt", FILE_WRITE);
```

El proceso de guardado es sencillo, tan solo hay que usar la siguiente instrucción con la variable deseada, en este caso hemos mostrado el peso total del lisímetro.

```
myFile.print(peso_total);
```

Para registrar los errores de funcionamiento se ha incorporado la siguiente instancia. El primero para determinar si recibe lecturas fuera de rango de manera global, el segundo registra la célula que está estropeada si hubiese, y la tercera si hay una fuga de agua.

```
if(fuera_rango)
{myFile.print("Peso-fuera-rango");myFile.print(" "); }

if(fuera_rango_celula>0)
{myFile.print("Error-celula-nº");myFile.print(fuera_rango_celula);myFile.print(" ");}

if(perdida_agua)
{myFile.print("Perdida-agua");myFile.print(" ");}
```

Una vez finalizado el proceso de escritura, se cierra la comunicación:

```
myFile.close();
```

- Para configurar la pantalla TFT:

```
#include <Adafruit_GFX.h> // Adafruit Biblioteca gráfica
#include <Adafruit_ST7735.h> // Adafruit Biblioteca ST7735

Adafruit_ST7735 tft = Adafruit_ST7735(10, 9, 8);
```

Para representar los datos en la pantalla TFT, usamos la función *dibujar(t)*, creada para tal efecto.

```
tft.fillScreen(ST7735_BLACK);
tft.setCursor(0,0);
tft.setTextSize(1);
tft.setTextColor(ST7735_BLUE);
tft.println(peso_total);
```

La primera función, *tft.fillScreen* limpia la pantalla (inicializa) y la devuelve a su color negro original. *Tft.setCursor* determina dónde se sitúa el cursor para escribir (coordenadas x e y, unidades en píxeles). *Tft.setTextSize* define el tamaño de escritura, en número de píxeles. Después, *tft.setTextColor* establece el color del texto, y finalmente *tft.print()* o *tft.println()* junto con variable que queremos guardar (*peso_total*).

Flujograma del Arduino de Registro en finca (I)

A continuación se muestra el flujograma del Arduino de Registro, el cual recibe las lecturas del Arduino de Adquisición para almacenarlas y mostrarlas en pantalla:

- 1) El Arduino de Registro se encuentra a la escucha y solo reacciona cuando recibe la instrucción de recibir datos mediante el retorno de carro \n.
- 2) Una vez para recibir las lecturas, el código de Arduino está preparado para recibir 12 variables. La variable *Cont* se inicializa.
- 3) Las booleanas de 1 byte usará sencillamente *Serial.read()*, pero aquellas variables grandes de 4 bytes como las medidas de peso o los pulsos de agua utilizarán la función *recibir_4bytes()* destinada para tal efecto.
- 4) 5) 6) Al final de cada variable recibida, la variable *Cont* incrementa en uno, hasta el valor de 12 donde termina nuestro proceso de lectura.
- 7) 8) Una vez terminada la comunicación entre el Arduino de Adquisición y el Arduino de Registro, este último almacena las variables actualizadas en la tarjeta SD, y a continuación refresca la pantalla TFT con los nuevos valores. Una vez finalizado este proceso, vuelve al estado inicial de escucha en el puerto.

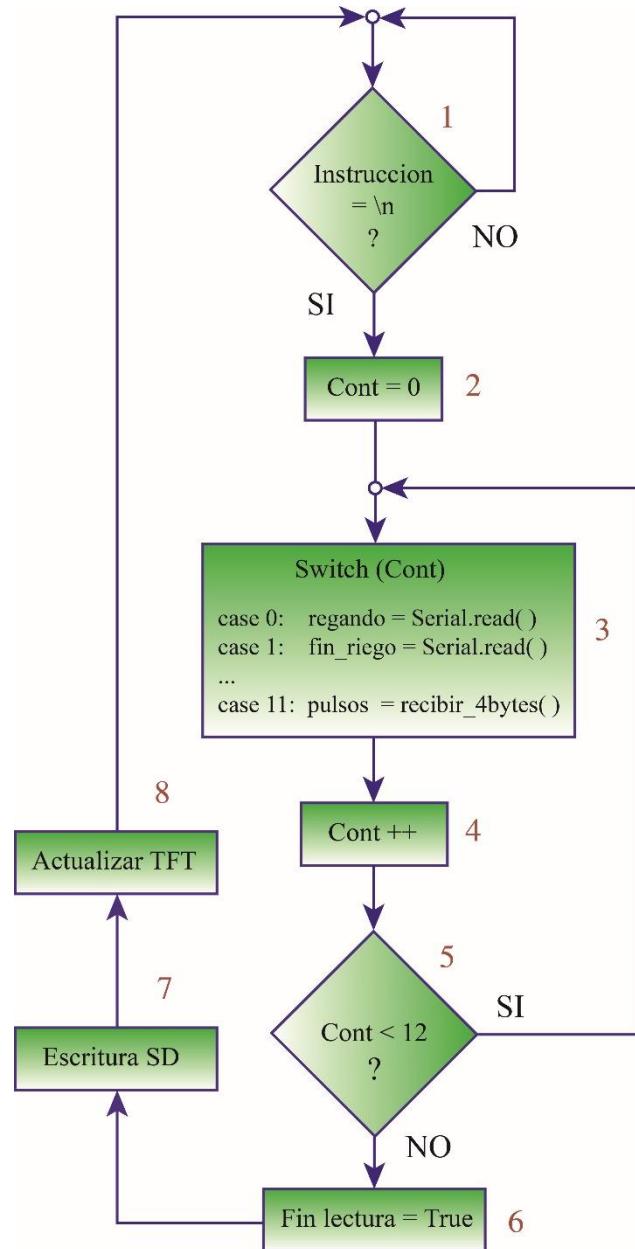


Figura 5.15: Arduino de Registro. Flujograma

Resultado

En la figura 5.16 se observa el Arduino de Registro en funcionamiento. A la izquierda el modo de funcionamiento normal, y la derecha el modo error:

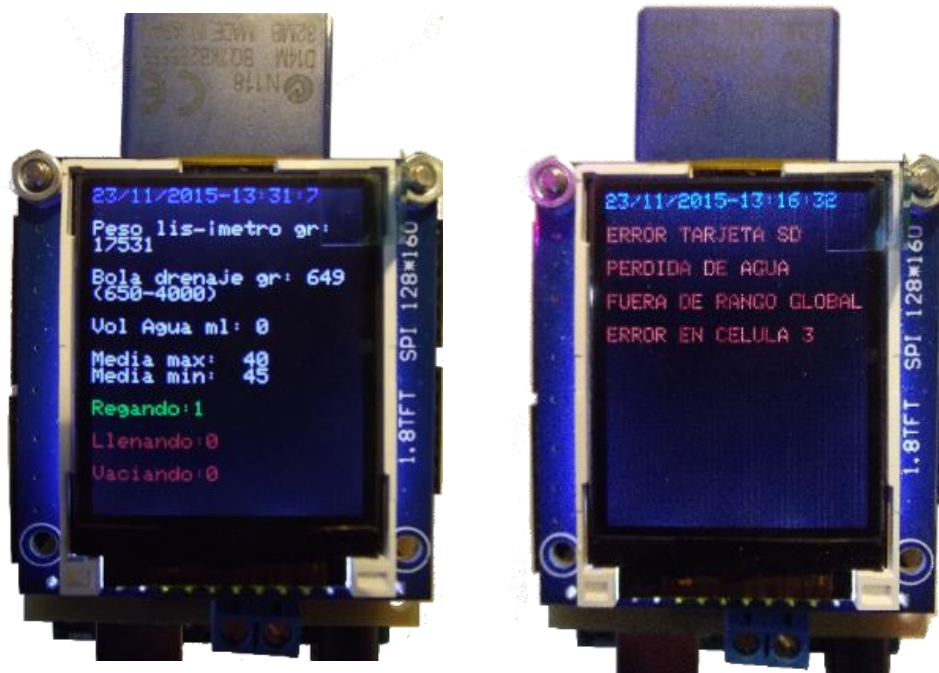


Figura 5.16: Arduino de Adquisición para finca (I). Modo normal (Izq), y modo error (Der)

Como se puede observar, se han utilizado el color azul para la fecha y hora, blanco para lecturas normales, verde para los booleanos con valor alto, y rojo para los de valor bajo y errores (mayúsculas).

Para el modo error carga una pantalla diferente para hacer más notorio su estado. Podemos ver los 4 tipos de errores, explicados anteriormente en el apartado 4.2:

- Error en la conexión con la tarjeta SD
- Perdida o fuga de agua en el sistema de alimentación (registrado debido a un incremento de caudal)
- Fuera de rango global cuando el lisímetro trabaja por encima o debajo de su rango de funcionamiento debido a alguna razón
- Cuando la célula nº X trabaja fuera de su rango de trabajo

5.5 Modelo C: Ensayo en la finca (II). Shield Ethernet

Introducción

Este capítulo se desarrolló como una mejora de la configuración anterior (Modelo B), ya que recoger los datos en un dispositivo físico como es una tarjeta SD supone un método engorroso y que limita los experimentos. El conocido como “Cloud storage” o almacenaje en la nube es un tipo de almacenamiento para datos en espacios virtuales.

Las ventajas de utilizar la nube como medio para almacenar y representar los datos son grandes:

- Es flexible: el espacio donde se almacenan los datos varía en función de la cantidad de datos. En el caso de la tarjeta SD, estamos sujetos a la capacidad de la tarjeta.
- Es rápido: es la ventaja más inmediata. Nos permite acceder a los datos desde prácticamente cualquier parte del mundo de manera cómoda y rápida.
- Barato: los precios para almacenar los datos en los servidores son muy asequibles y en la mayoría de los casos, si requerimos de poco espacio suele ser gratuito.
- Es fiable: un medio físico como una SD siempre puede sufrir daños más fácilmente que los servidores de la nube, ya que estos están preparados para ello. Además en caso de fallo se puede recurrir de manera “fácil” al data backup o recuperado de archivos.

Este capítulo se ha dividido en dos partes:

- a) Lisímetro-upct
- b) Ubidots

Primeramente, se empezó a trabajar con la página web *www.lisímetro-upct.hol.es*, pero al implementar el sistema en la finca aparecieron algunos problemas de conexión con la página web por lo que finalmente se optó por presentar una solución alternativa: *www.ubidots.com*.

5.5.1 Lisímetro-upct

Front-end

Para comenzar, decir que para el desarrollo de esta página web, no se ha hecho uso en ningún momento de herramientas de diseño de páginas web.

Todo se ha hecho mediante modificación del código (se ha usado la herramienta Notepad++), lo que le otorga un aspecto no tan atractivo a la página web, pero que me ha permitido desarrollar mi conocimiento en HTML y PHP de manera muy sustancial.

Al acceder a “*Lisímetro-upct.hol.es*” nos encontramos un menú de acceso (Figura 5.17) en el cual debemos autenticarnos para poder acceder al lisímetro.



Figura 5.17: Menú de acceso

Para iniciar sesión, basta con introducir el nombre de usuario y la contraseña. Al confirmar nuestro acceso, nos da paso al menú principal (Figura 5.18). Tenemos 3 opciones principales: Mostrar los datos, Acceder a la base de datos, y Configuración.



Figura 5.18: Menú principal

Si se accede al submenú “Mostrar datos” (Figura 5.19) nos encontramos una tabla con los datos guardados hasta el momento: ID del dato, fecha de la medida, volumen de agua registrado, peso del lisímetro, peso de la bola de drenaje, y varias variables de estado booleanas.

Ha accedido usted a la base de datos...

Desea volver al menú?

| ID | Fecha | Vol agua | Peso total | Peso drenaje | B_regando | B_fin_riego | B_drenaje | B_vaciando_peso |
|----|---------------------|----------|------------|--------------|-----------|-------------|-----------|-----------------|
| 4 | 2015-02-03 18:13:05 | 0 | 17852 | 653 | 0 | 0 | 0 | 0 |
| 5 | 2015-02-03 18:14:05 | 0 | 17851 | 653 | 0 | 0 | 0 | 0 |
| 6 | 2015-02-03 18:15:05 | 0 | 17855 | 653 | 0 | 0 | 0 | 0 |
| 7 | 2015-02-03 18:16:05 | 0 | 17853 | 653 | 0 | 0 | 0 | 0 |
| 8 | 2015-02-03 18:17:05 | 0 | 17851 | 653 | 0 | 0 | 0 | 0 |
| 9 | 2015-02-03 18:18:05 | 0 | 17857 | 652 | 0 | 0 | 0 | 0 |
| 10 | 2015-02-03 18:19:05 | 0 | 17856 | 653 | 0 | 0 | 0 | 0 |
| 11 | 2015-02-03 18:20:06 | 0 | 17852 | 653 | 0 | 0 | 0 | 0 |
| 12 | 2015-02-03 18:21:06 | 0 | 17855 | 653 | 0 | 0 | 0 | 0 |
| 13 | 2015-02-03 18:22:06 | 0 | 17853 | 653 | 0 | 0 | 0 | 0 |
| 14 | 2015-02-03 18:23:06 | 0 | 17856 | 653 | 0 | 0 | 0 | 0 |
| 15 | 2015-02-03 18:24:06 | 0 | 17854 | 652 | 0 | 0 | 0 | 0 |
| 16 | 2015-02-03 18:25:06 | 0 | 17857 | 653 | 0 | 0 | 0 | 0 |

Figura 5.19: Menú "Mostrar datos"

En el submenú “*Configuración*” (Figura 5.20), nos encontramos dos variables, Var1 y Var2. Como ejemplo, se ha nombrado a la primera Capacidad de campo y la segunda número de riegos. Ambas variables no se han llegado a utilizar realmente en el lisímetro, puesto que éste tiene programado la hora de riego y duración mediante el programador externo. Este apartado se creó para implementar en un futuro el riego por lisimetría, en el cual se podría configurar diferentes parámetros.

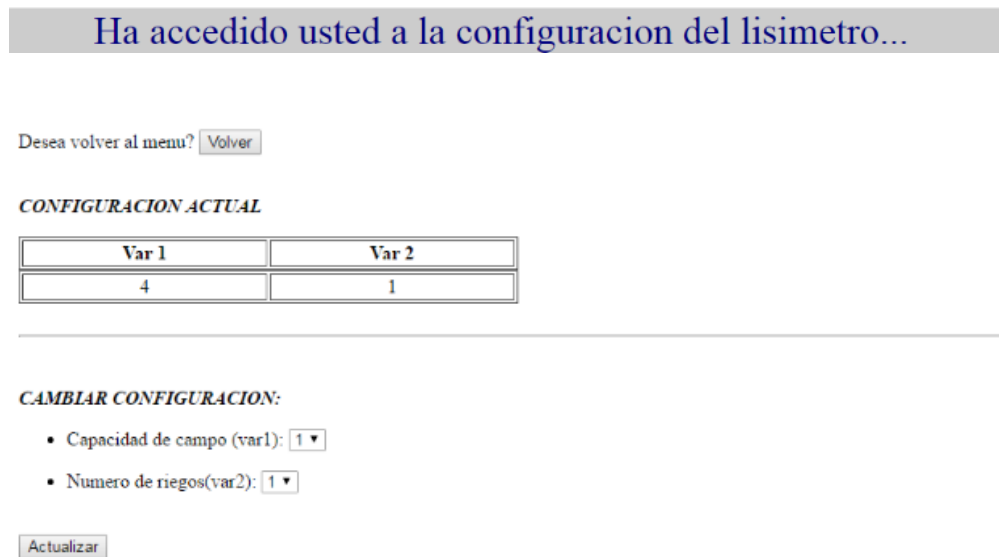


Figura 5.20: Menú "Configuración"

Base de datos

En el submenú “*Descargar datos*” nos encontramos un hiperlink donde se encuentra la base de datos; nos pedirá ingresar usuario (u514696440_upct) y contraseña (lísímetro).

La base de datos (Figura 33) MySQL se gestiona mediante phpMyAdmin; ésta última es una herramienta escrita en lenguaje PHP cuyo objetivo es el control y administración de MySQL a través de páginas web. Permite crear, editar y eliminar bases de datos así como sus tablas incluidas en la misma; en definitiva, phpMyAdmin permite ejecutar cualquier sentencia SQL.

Mostrando registros 0 - 18 (19 total. La consulta tardó 0.0002 seg)

```
SELECT *
FROM `arduino`
LIMIT 0, 30
```

Mostrar : Fila de inicio: 0 Número de filas: 30 Cabeceras cada 100 filas

Ordenar según la clave: Ninguna

| ID | tiempo | vol_agua | peso_total | peso_drenaje | b_regando | b_fin_riego | b_drenaje | b_vaciando_peso |
|----|---------------------|----------|------------|--------------|-----------|-------------|-----------|-----------------|
| 4 | 2015-02-03 18:13:05 | 0 | 17852 | 653 | 0 | 0 | 0 | 0 |
| 5 | 2015-02-03 18:14:05 | 0 | 17851 | 653 | 0 | 0 | 0 | 0 |
| 6 | 2015-02-03 18:15:05 | 0 | 17855 | 653 | 0 | 0 | 0 | 0 |
| 7 | 2015-02-03 18:16:05 | 0 | 17853 | 653 | 0 | 0 | 0 | 0 |
| 8 | 2015-02-03 18:17:05 | 0 | 17851 | 653 | 0 | 0 | 0 | 0 |
| 9 | 2015-02-03 18:18:05 | 0 | 17857 | 652 | 0 | 0 | 0 | 0 |
| 10 | 2015-02-03 18:19:05 | 0 | 17856 | 653 | 0 | 0 | 0 | 0 |
| 11 | 2015-02-03 18:20:06 | 0 | 17852 | 653 | 0 | 0 | 0 | 0 |
| 12 | 2015-02-03 18:21:06 | 0 | 17855 | 653 | 0 | 0 | 0 | 0 |
| 13 | 2015-02-03 18:22:06 | 0 | 17853 | 653 | 0 | 0 | 0 | 0 |

Tabla 4: Base de datos, Arduino

Hemos creado 3 tablas diferentes:

- Arduino (Tabla 4) donde almacena las variables del lisímetro:
 - ID. Único de cada valor
 - Tiempo. Hora y fecha actuales, obtenidos de la configuración del ordenador
 - Vol_agua. El volumen en ml regado que se ha registrado hasta ese momento
 - Peso_Total. El peso total en gramos del lisímetro
 - B_regando. Variable booleana: vale 1 si el caudalímetro detecta que se está regando
 - B_fin_riego. Su valor es 1 al finalizar el riego. Se utiliza para el timer4 (Figura 5.5)
 - B_drenaje. Su valor es 1 cuando se abren las electroválvulas para vaciar la bola de drenaje. Se utiliza para el timer3 (Figura 5.5)
 - B_vaciando_peso. Variable cuyo valor cambia a 1 cuando no se encuentra en ninguna otra parte del programa, y debido a lluvias el lisímetro necesita vaciar la bola drenaje

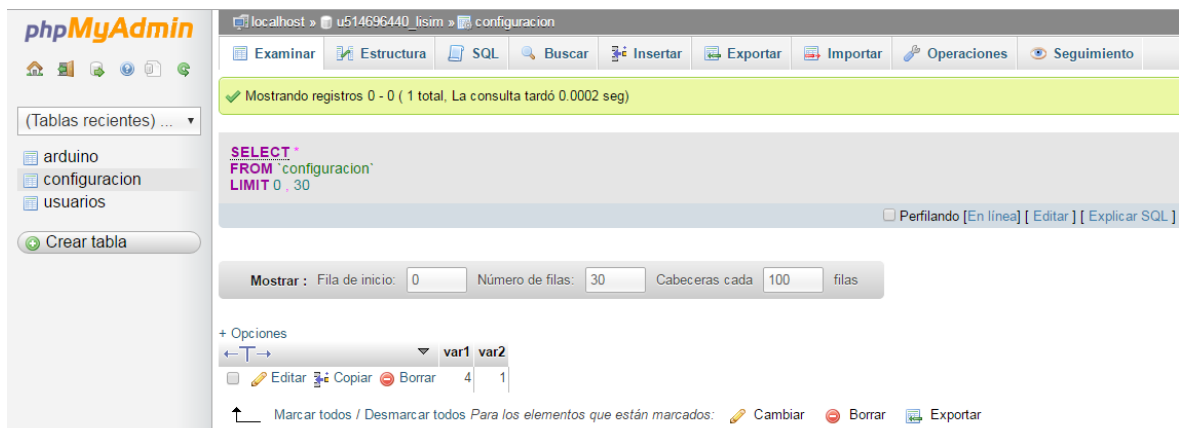


Tabla 5: Base de datos, Configuración

- Configuración (Tabla 5) se encuentran nuestras variables configurables del lisímetro:
 - Variable a configurar numero 1 (Var1)
 - Variable a configurar numero 2 (Var2)

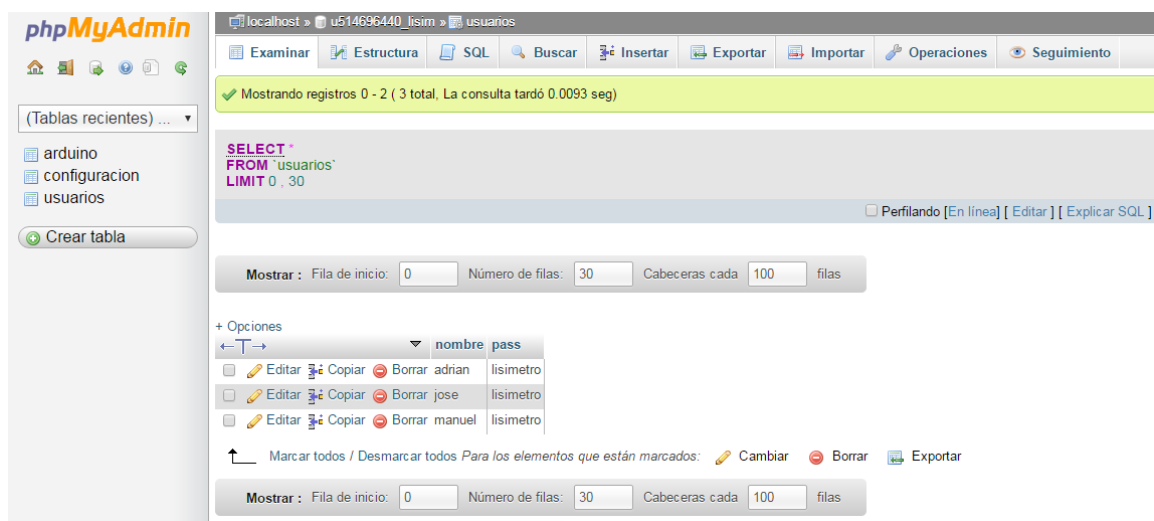


Tabla 6: Base de datos, Usuarios

- Usuarios (Tabla 6):
 - Nombre. Usuario con acceso a la página web (adrian, jose, manuel)
 - Pass. Contraseña de acceso (lisímetro)

Hosting

Nuestra página web se encuentra alojada en Hostinger, el cual nos brinda de manera gratuita un espaciado virtual de Hosting de hasta 2GB de almacenamiento y 100GB de tráfico mensual, cediéndonos un subdominio gratuito de tipo *.hol.es* en este caso. También permite usar scripts basados en PHP, así como crear un par de bases de datos en MySQL.

Nuestro usuario FTP es “u514696440” y su contraseña “lísímetro”.

| Acceso FTP | |
|-------------------------------|---|
| Host FTP | ftp.lisimetro-upct.hol.es |
| IP FTP | 31.170.164.228 |
| Puerto FTP | 21 |
| Usuario FTP | u514696440 |
| Contraseña FTP | |
| Carpeta dónde Cargar Archivos | public_html |
| ¿Olvidaste tu contraseña FTP? | Cambiar contraseña de la cuenta |
| Cientes FTP recomendados | SmartFTP FileZilla |

Figura 5.21: Hostinger. Acceso FTP a nuestra página

Como ya hemos dicho, en Hostinger hemos creado una base de datos MySQL donde almacenamos las lecturas del Arduino. A continuación podemos ver la base de datos creada, donde se puede acceder, eliminar o modificar.

| Base de Datos MySQL | Usuario MySQL | Host MySQL | Uso de Disco, Mb |
|---------------------|-----------------|--------------------|------------------|
| u514696440_lisim | u514696440_upct | mysql.hostinger.es | 0.04 |

Figura 5.22: Hostinger. Base de datos MySQL

Archivos de la página web

Para la realización de la página web se ha hecho uso de 4 tipos de código. Los formularios para interactuar con los datos mediante **PHP**, la presentación de menús, datos e imágenes se han programado con **HTML**, la modificación estética con **CSS**, y algo de **MySQL** para la gestión de la base de datos. Los códigos completos se encuentran en el CD adjuntado con el TFE, en el apartado CD/Códigos/Lisimetro-upct/.

Para poder interactuar y modificar el contenido de la página web, se ha utilizado una herramienta muy útil llamada Filezilla (Figura 5.23), el cual es un cliente de tipo FTP de código abierto y libre, un programa que te permite bajar y subir archivos de manera sencilla a un servidor remoto.

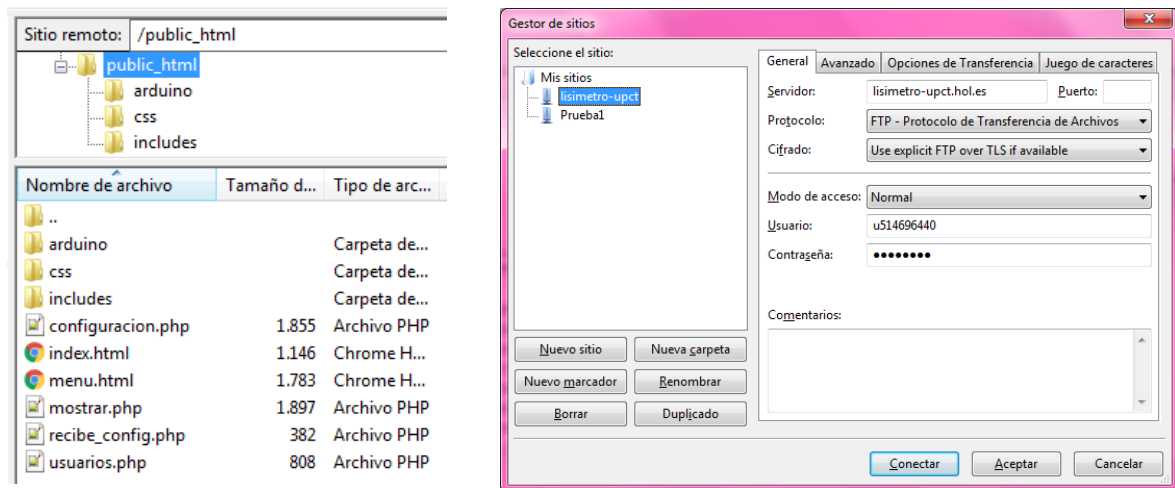


Figura 5.23: Filezilla. Contenido de página web (Izq) y Gestor de sitios (Der)

Como podemos observar en la figura anterior, el contenido de la página web se puede clasificar en varias carpetas y archivos:

1. ARDUINO: Carpeta donde se encuentran los formularios PHP que permiten interactuar a nuestro Arduino con la página web.
 - Connection.php
 - Enviar_datos.php
 - Leer_configuracion.php
2. CSS: Aquí se guardan los códigos CSS para crear estilos y modificar la interfaz gráfica de la página web.
 - Estilos.css
 - Estilos.css

3. INCLUDES: Carpeta donde se configuran los parámetros de conexión entre la página web y la base de datos.
 - Connection.php
4. HTML: Se pueden ver 2 archivos diferentes:
 - Index.html es el archivo inicial, el que solicita que introduzcamos usuario y contraseña (Figura 5.17)
 - Menu.html es nuestro menú principal (Figura 5.18).
5. PHP: Se observan 4 archivos.
 - Usuarios.php forma parte de Index.html para introducir y comprobar el usuario y contraseña.
 - Mostrar.php solicita a la base de datos los valores registrados (opción Mostrar Datos).
 - Configuracion.php y recibe_configuracion.php permiten mostrar y modificar los parámetros del lisímetro (opción Configuración)

Programación página web

Ahora procede a mostrar y explicar el funcionamiento de las partes más importantes de los archivos creados para la página web www.lisímetro.upct.hol.es.

Uno de los primeros archivos es Connection.php ya que define los parámetros hosting, usuario, contraseña y nombre de la base de datos, para realizar la conexión de nombre "conn" con la base de datos.

```
$dbhost = "mysql.hostinger.es";  
$dbuser = "u514696440_upct";  
$dbpass = "lisímetro";  
$db = "u514696440_lisim";  
  
$conn = mysqli_connect($dbhost, $dbuser, $dbpass, $db);
```

Nada más ingresar a la página web, el primer archivo al que accederá será aquel cuyo nombre sea Index.html. Por tanto, se ha programado este archivo como un menú de acceso en el que hay que ingresar usuario y contraseña.

```

<p class="leyenda">INICIAR SESIÓN:</p>
<form name="usuariocontra" type="submit" action="usuarios.php" method="post">
Nombre:
<input type="text" id="usuario" name="usuario" placeholder="Nombre" size="15">
<br /><br />
Contraseña :
<input type="password" id="pass" name="pass" placeholder="Password"size="15">
<br /><br />
<input class="inicio" type="submit" id="enviar" name="enviar" value="Entrar">
</form>

```

Introducimos usuario y contraseña, y al darle a “Aceptar”, se lleva el usuario introducido (id= usuario) y contraseña (id= pass”) al archivo Usuarios.php.

```

$query = "SELECT * FROM usuarios WHERE nombre = '$user'";
$q = mysqli_query($conn, $query);

if ($f2=mysqli_fetch_array($q))
{
    if($pass==$f2['pass']){header('Location: menu.html');} // si la contraseña es correcta,
//da paso al menu principal

    else{header('Location: index.html'); // si la contraseña es erronea, vuelve a cargar
    }
}
else{header('Location: index.html'); // si no existe este usuario, vuelve a cargar
}

```

Usuarios.php será el encargado de comprobar si los datos introducidos coinciden con los existentes en la base de datos. Si no coinciden, vuelve a cargar la página. Si coinciden, da paso a Menu.html que es el archivo que contiene el menú principal. Dentro de este, tenemos 4 opciones.

1. La primera y más sencilla, es cerrar la sesión, donde nos redirecciona al menú de acceso anteriormente mencionado Index.html.

```

<div class="cinco"> <!-- al hacer click nos redirecciona al index.html,
vuelta al menu de ingreso a la página-->
Desea cerrar sesión?
<input type="button" onClick="parent.location='index.html'" value='Cerrar'>
</div>

```

2. La segunda, es mostrar los datos del lisímetro almacenados hasta ese momento. Al pinchar en el botón, nos lleva al archivo Mostrar.php.

```

<p class="leyenda">MOSTRAR DATOS:</p><br />
<form name="mostrardatos" type="submit" action="mostrar.php">
Quiere visualizar datos del lisímetro?
<br /><br />
<input class="boton1" type="submit" id="mostrar" name="mostrar" value="Mostrar">
</form>

```

Dentro de Mostrar.php, se crea una consulta `mysqli_query` a la base de datos, y se obtienen todas las variables pertenecientes a una lectura (ID, tiempo, vol_agua,...) y las muestra en una tabla.

```
$result = mysqli_query($conn, "SELECT * FROM arduino");

if ( $a==$b ) { // Este bucle obtiene los variables de cada lectura
                // de la base de datos y los muestra en la tabla
?>
    <table border="1">
    <tr>
    <td width="50"><?php echo $person['ID'] ?></td>
    <td width="200"><?php echo $person['tiempo'] ?></td>
    <td width="100"><?php echo $person['vol_agua'] ?></td>
    <td width="100"><?php echo $person['peso_total'] ?></td>
    <td width="100"><?php echo $person['peso_drenaje'] ?></td>
    <td width="100"><?php echo $person['b_regando'] ?></td>
    <td width="100"><?php echo $person['b_fin_riego'] ?></td>
    <td width="100"><?php echo $person['b_drenaje'] ?></td>
    <td width="120"><?php echo $person['b_vaciando_peso'] ?></td>
    </tr>
    </table>
<?php }
```

3. La tercera opción, es acceder a la base de datos para modificarla, crear nuevos usuarios, o descargar los datos en formato Excel. Al pinchar en acceder, nos lleva a la URL de la BD. Tan solo necesitamos ingresar el usuario y la contraseña proporcionados (usuario = u514696440_upct, contraseña = lisímetro).

```
<div id="dos">    <!-- al hacer click nos redirecciona
                  a la url de la base de datos -->

    <p class="leyenda">DESCARGAR DATOS:</p>
    <a target="_blank" href= "http://sql5.hostinger.es/phpmyadmin/
                            index.php?db=u514696440_lisim">
                            Ir a la base de datos</a>

    <br /><br />
    usuario: "u514696440_upct"
    <br /><br />
    contraseña:
    <br />
    "lisímetro"
</div>
```

4. La cuarta opción es acceder al submenú de configuración del lisímetro. Da paso al archivo Configuracion.php

```

<p class="leyenda">CONFIGURACIÓN:</p>
<br />
<form name="config" type="submit" action="configuracion.php">
Ir al menu de configuración del lisímetro
<br /><br />
<input class="boton2" type="submit" id="mostrar" name="mostrar"
value="Configurar">
</form>

```

Dentro de Configuracion.php, primeramente se mostrará la configuración actual:

```

while($person = mysqli_fetch_array($result)){
if ( $a==$b ) { ?>
<table style="text-align:center;" border="1">
<tr>
<td width="200"><?php echo $person['var1'] ?></td>
<td width="200"><?php echo $person['var2'] ?></td>
</tr>
</table>

```

Y después, permite modifícalas, utilizando para esto Recibe_config.php.

```

<p>CAMBIAR CONFIGURACION:</p>
<form name="asdf" type="submit" action="recibe_config.php" method="post">
<ul><li>
Capacidad de campo (var1):
<select name="var1" id="var1">
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
</select></li>
<p> </p><li>
Numero de riegos(var2):
<select name="var2" id="var2">
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
</select></li>
</ul><br />
<input type="submit" onclick="alert('Enviando datos...!')" id="enviar2"
name="enviar2" value="Actualizar">

```

Dentro de Recibe_config.php, nos llevamos los valores nuevos de las dos variables, y los actualiza "Update" en la base de datos.

```

$num1 = $_POST['var1'];
$num2 = $_POST['var2'];

mysqli_query($conn, "UPDATE `configuracion` SET `var1`=' $num1', `var2`=' $num2'");

```

Interacción entre la página web y Arduino

Dentro de los archivos alojados en la página web, hay una carpeta de nombre “Arduino” en donde están almacenados aquellos archivos encargados de trabajar con Arduino.

El primero y más importante, es el que define la conexión del Arduino con la base de datos, llamado Connection.php. Este archivo, aunque tiene mismo nombre que el definido en el apartado anterior, son diferentes.

```
$dbhost = "mysql.hostinger.es";
$dbuser = "u514696440_upct";
$dbpass = "lisimetro";
$db      = "u514696440_lisim";
$con     = mysqli_connect($dbhost,$dbuser,$dbpass,$db);
```

El siguiente archivo se llama Enviar datos.php, y es el encargado de recibir los datos procedentes del Arduino, y guardar cada variable en la base de datos.

```
$num1 = mysqli_real_escape_string($con, $_GET['vol_agua']);
$num2 = mysqli_real_escape_string($con, $_GET['peso_total']);
$num3 = mysqli_real_escape_string($con, $_GET['peso_drenaje']);
$num4 = mysqli_real_escape_string($con, $_GET['b_regando']);
$num5 = mysqli_real_escape_string($con, $_GET['b_fin_riego']);
$num6 = mysqli_real_escape_string($con, $_GET['b_drenaje']);
$num7 = mysqli_real_escape_string($con, $_GET['b_vaciando_peso']);

$query = "INSERT INTO arduino(`vol_agua`,`peso_total`,`peso_drenaje`,`b_regando`,`b_fin_riego`,`b_drenaje`,`b_vaciando_peso`) VALUES('".$num1."', '".$num2."', '".$num3."', '".$num4."', '".$num5."', '".$num6."', '".$num7."')";
```

El siguiente archivo incluido en la carpeta es Leer configuracion.php. Este archivo obtiene los valores de las variables de configuración del lisímetro, y las muestra en dos cadenas diferentes incluyendo el elemento separador “@”, de manera que el Arduino se encargará de descifrarlas en su propio código para obtener las substrings y por tanto los valores numéricos.

```
$result1 = mysqli_query($con, "SELECT var1 FROM configuracion");
while($lectura1 = mysqli_fetch_array($result1))
{ echo "var1=".$lectura1['var1']."@"; } // esta es la cadena 1

$result2 = mysqli_query($con, "SELECT var2 FROM configuracion");
while($lectura2 = mysqli_fetch_array($result2))
{ echo "var2=".$lectura2['var2']."@"; } // esta es la cadena 2
```

Código de Arduino

- Para utilizar el shield Ethernet en Arduino necesitamos usar la librería “Ethernet.h”, y para establecer la conexión con nuestra página web configuraremos como servidor:

```
char server[] = "www.lisimetro-upct.hol.es";
```

y para conectarse a la misma:

```
if(client.connect(server, 80))
```

- Para que el Arduino **envíe** los valores registrados a la página web, necesitamos que los envíe a “enviar_datos.php” ya que este es el encargado de recibir los datos de Arduino. Por tanto, usamos la instancia:

```
client.print("GET /arduino/enviar_datos.php?");
```

A continuación enviamos todas las variables concatenadas en formato string, separadas mediante el símbolo “&” para discriminar una variable de su anterior.

```
client.print("vol_agua=");
```

```
client.print(vol_agua);
```

```
client.print("&peso_total=");
```

```
client.print(peso_total);
```

```
client.print("&peso_drenaje=");
```

```
client.print(peso_drenaje);
```

```
client.print("&b_regando=");
```

```
client.print(regando);
```

```
client.print("&b_fin_riego=");
```

```
client.print(fin_riego);
```

```
client.print("&b_drenaje=");
```

```
client.print(drenaje);
```

```
client.print("&b_vaciando_peso=");
```

```
client.print(vaciando_peso);
```

Después una cadena de caracteres requeridos para establecer la conexión:

```
client.println(" HTTP/1.1");
```

```
client.print("Host: ");
```

```
client.println(server);
```

```
client.println("User-Agent: Arduino-Ethernet");
```

```
client.println("Connection: close");
client.println();
```

Para cerrar la comunicación, finalmente utilizamos

```
client.stop();
client.flush();
```

- Para que el Arduino **lea** los valores de configuración, utilizamos:

```
client.print("GET /arduino/leer_configuracion.php");
```

tras lo que mediante `client.read()` leemos la respuesta del servidor.

```
while (client.available())
    { char c = client.read();
      cadena.concat(c); }
```

Igual que anteriormente, nuestra página está programada para enviar a Arduino dos parámetros configurables; los valores van concatenados en formato string y los discriminamos mediante el símbolo "@" entre un valor y otro. Para hacer esta discriminación en Arduino, utilizamos `cadena.indexOf` y después `cadena.substring`:

```
int posicion1=cadena.indexOf("var1=");
int posicion2=cadena.indexOf("var2=");
int posicion3=cadena.indexOf("@");
int posicion4=cadena.indexOf("@",posicion2 + 1);
```

Por lo que los dos parámetros, una vez leídos de la página, corresponderán con cadena 1 y cadena2:

```
String cadena1 = cadena.substring(posicion1,posicion3);
String cadena2 = cadena.substring(posicion2,posicion4);
```


5.5.2 Ubidots

Introducción

Ubidots nació en América Latina a partir de una firma de ingeniería, especializada en el desarrollo de software y hardware para proyectos dentro del “internet de las cosas” (IoT). Desde que fue lanzada en 2014, la nube de Ubidots se ha convertido en una de las mayores plataformas de IoT del mercado, dando soporte a miles de proyectos de IoT en más de 40 países.

Ubidots trae una serie de librerías que permiten la implementación de microcontroladores como Arduino, Adafruit o Raspberry Pi.

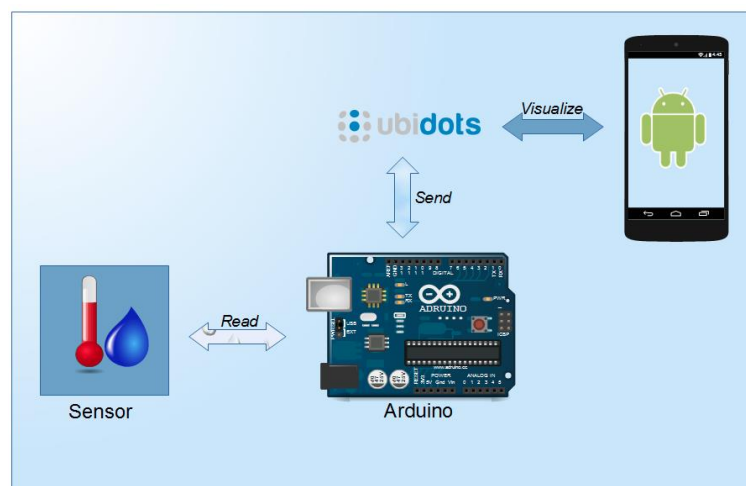


Figura 5.24: Ubidots permite implementar nuestros proyectos en la nube de manera sencilla

Para comenzar a usar Ubidots necesitamos conectar nuestro dispositivo a la nube, utilizando el “firmware” y librerías propias de Ubidots. A continuación, podremos crear diferentes “mesas de trabajo” o Dashboard (Figura 5.25) que se ajuste a nuestras necesidades, en el cual podemos representar, de manera sencilla y cómoda mediante Widgets, las diferentes variables en formato de gráfica, indicadores, geolocalización y muchas más.



Figura 5.25: Ejemplo de mesa de trabajo con diferentes widgets

En la Figura 5.26 se puede observar las diferentes opciones que tenemos para visualizar una variable: tablas, gráficas, indicadores...

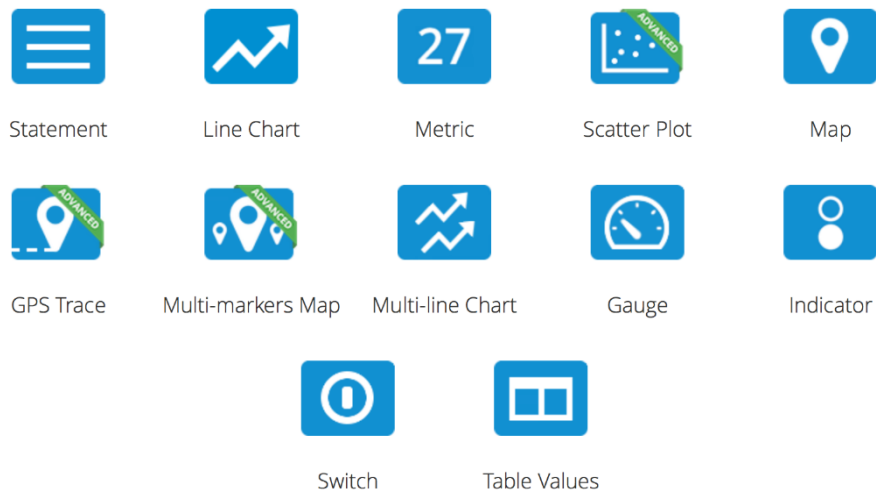


Figura 5.26: Diferentes tipos de widgets que podemos generar

Entre sus herramientas, se encuentra la programación de respuestas programadas o triggers a determinados sucesos (Figura 5.27), desde enviar avisos en forma de SMS a nuestro móvil, o un email a nuestro correo, hasta interactuar con las variables y otras páginas web (mediante métodos GET, POST,...). A su vez, permite la descarga de datos en formato Excel de manera directa.

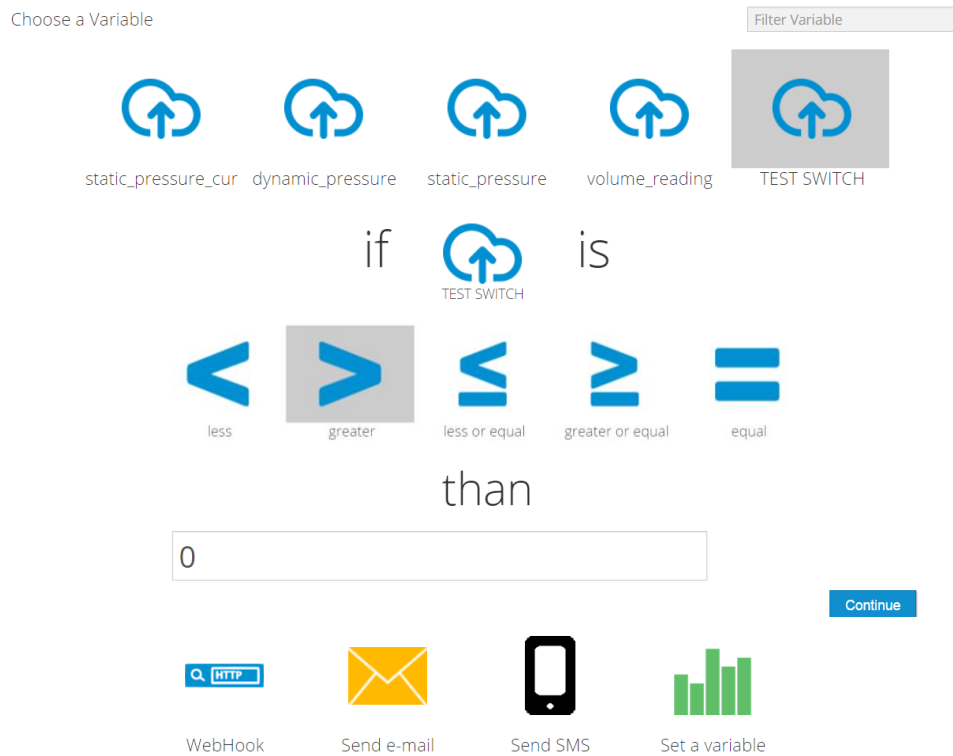


Figura 5.27: Respuestas programadas

Front-end

Para nuestro caso, se ha considerado que para monitorizar nuestro proyecto de manera representativa, bastaría con una gráfica general del peso del lisímetro, un indicador de aguja para peso de la bola de drenaje e indicadores on-off para los booleanos. Las figuras 5.28 y 5.29 componen entonces nuestra “dashboard” o mesa de trabajo. La figura 5.28 corresponde a una gráfica a tiempo real de la evolución del peso del lisímetro, en la que fácilmente podemos deducir un incremento de peso debido al riego.

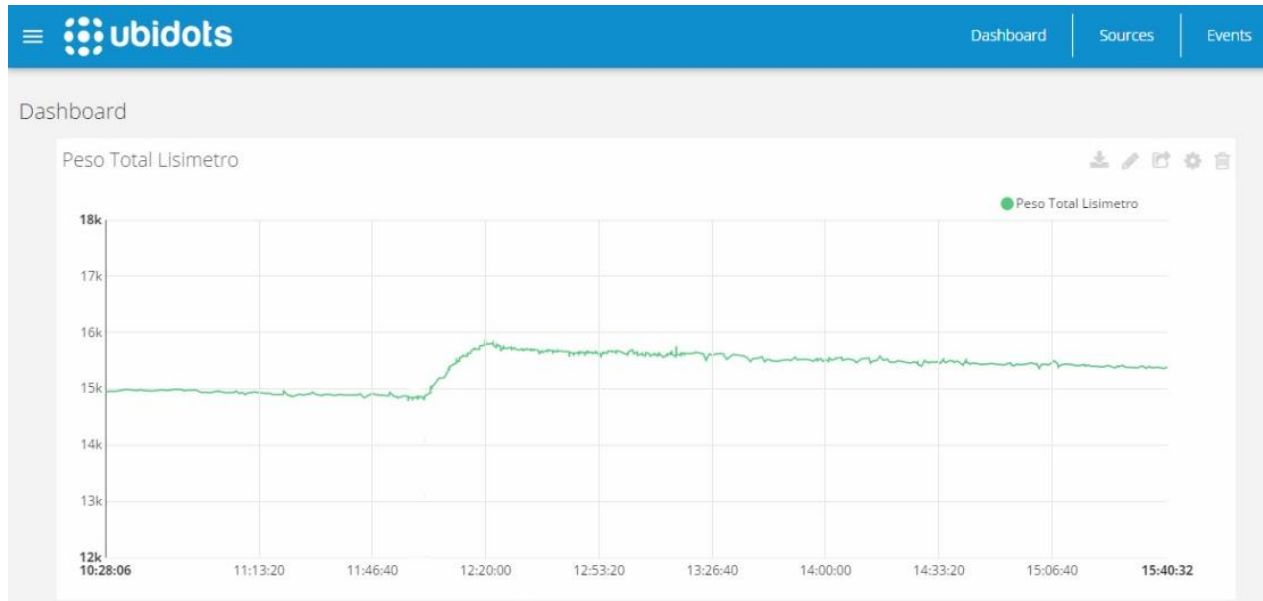







Figura 5.28: Evolución del peso del lisímetro

Abajo a la izquierda tenemos un indicador de aguja que representa a tiempo real el peso de la bola de drenaje. A la derecha, los diferentes estados booleanos del programa en formato ON – OFF (Regando, Fin de riego, Drenando, Vaciando).



Figura 5.29: Indicador bola drenaje (Izq). Variables booleanas (Der)

En la esquina superior derecha de los Widgets encontramos una serie de opciones:

-  Descargar los datos al ordenador, en una tabla de Excel.
-  Editar el formato de representación de la tabla
-  Compartir el Widget (Figura 5.30, Izq.)
-  Configurar los parámetros de la gráfica (Figura 5.30, Der)
-  Eliminar Widget

La opción de Compartir el Widget es especialmente útil puesto que genera un link público del Widget, permitiendo a otro usuario acceder a él externamente sin necesidad de entrar a la cuenta de Ubidots.

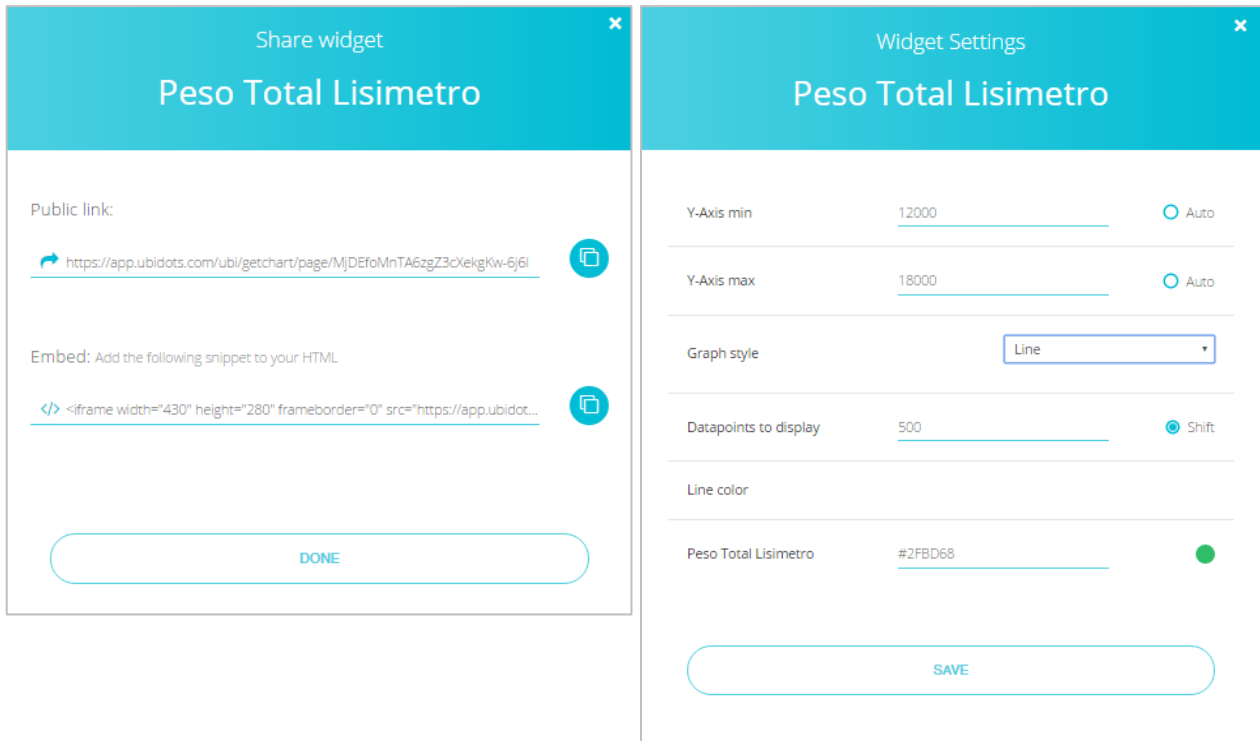


Figura 5.30: Compartir un Widget (Izq). Configurar Widget (De)

Back-end

En el apartado “Sources” dentro de Ubidots, hemos creado una nueva fuente de datos con el nombre de Arduino Ethernet. A continuación, aparece el contenido de la misma, todas las variables que se almacenan en la nube (Figura 5.31):

- B_Vaciando peso
- B_Fin de riego
- B_Drenaje
- B_Regando
- Peso Total Lisímetro
- Volumen Agua Total
- Peso bola drenaje

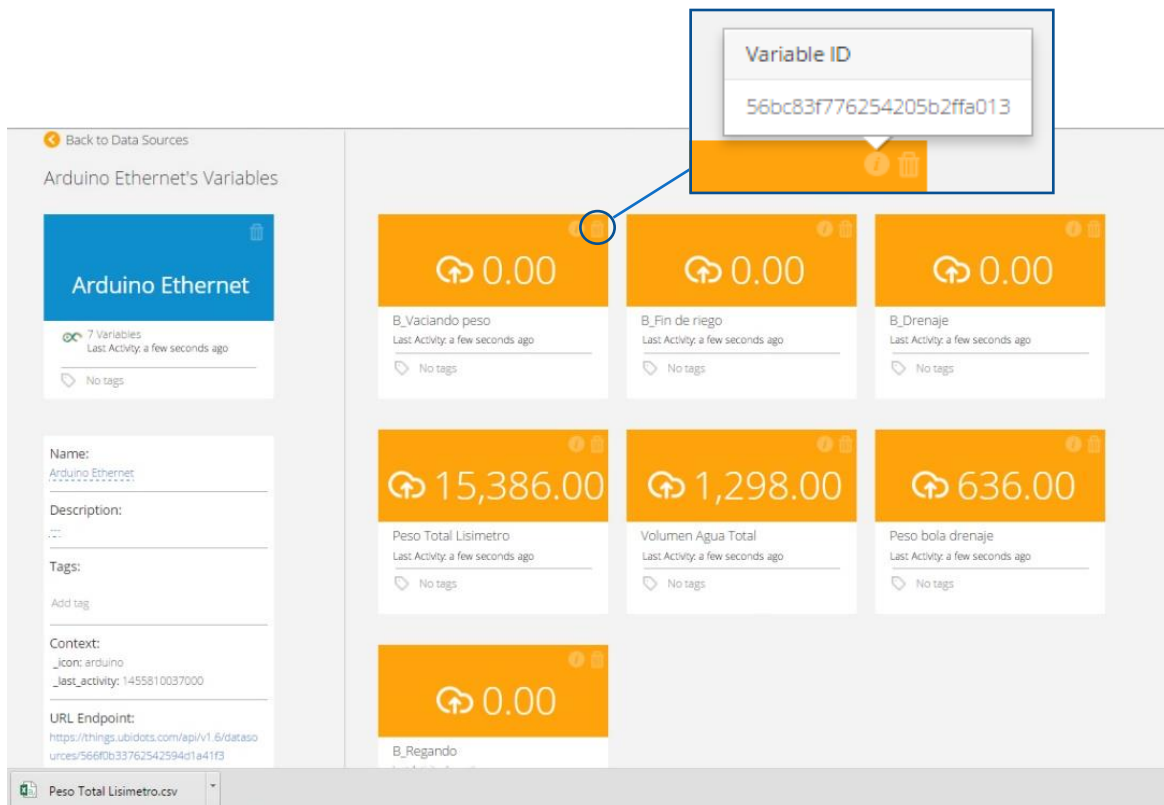


Figura 5.31: Ubidots. Variables y sus IDs correspondientes

Como se muestra en la figura anterior, cada variable creada en Ubidots tiene un código ID único. Este código nos permitirá darle una dirección a los datos que Arduino suba a la nube.

A su vez, nuestra cuenta tiene su propio código “Token” que será también necesario incluir en nuestro código de Arduino para vincularlo con nuestras variables en Ubidots.

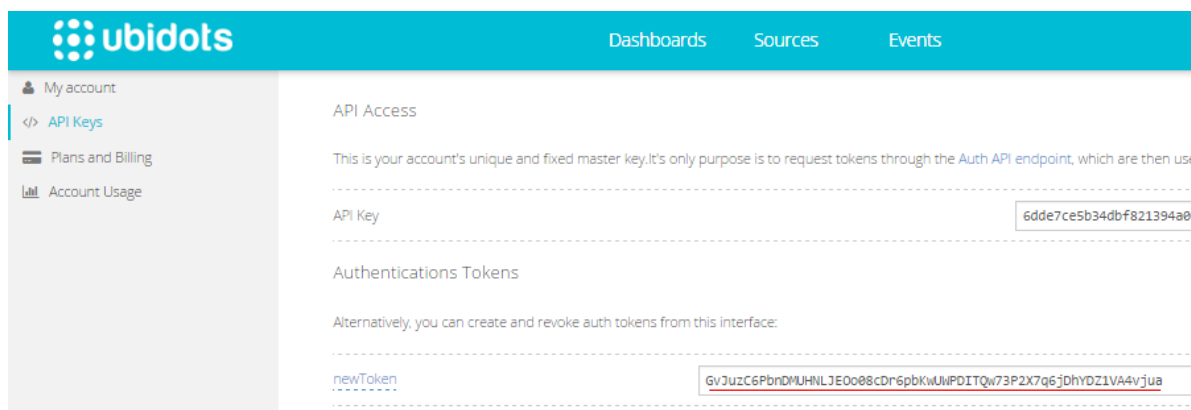


Figura 5.32: Ubidots. Código Token de la cuenta

En el código de Arduino, necesitaremos introducir los IDs de las variables anteriormente mencionadas, así como el código Token identificador de nuestra cuenta.

```
String peso_total_id = "56bc83f776254205b2ffa013";
String vol_agua_id = "56bc8413762542060758a92a";
String peso_drenaje_id = "56bc905c7625424f1f95b14f";

String regando_id = "56bc8996762542269f594f81";
String fin_riego_id = "56bc899c76254227aea15979";
String drenaje_id = "56bc89ba7625422774df59d4";
String vaciando_peso_id = "56bc89c3762542281f8f2852";

String token =
"GvJuzC6PbnDMUHNLEOo08cDr6pbKwUWPDITQw73P2X7q6jDhYDZ1VA4vjua";
```

Para enviar cada variable desde Arduino utiliza una función del tipo:

```
void save_value(String value, String id)
```

Donde la cadena *value* es el valor numérico de la variable, y su ID es el código propio de la variable que vimos anteriormente. Para conectarse a Ubidots, usamos:

```
if(client.connect("things.ubidots.com", 80))
```

A continuación una instancia de tipo:

```
client.print("POST /api/v1.6/variables/");
```

Donde seguidamente enviaremos el ID de la variable (id), una la cadena de caracteres y finalmente el valor numérico de la variable (var).

```
client.print(id);
client.println("/values HTTP/1.1");
client.println("Content-Type: application/json");
client.println("Content-Length: "+String(num));
client.println("X-Auth-Token: "+token);
client.println("Host: things.ubidots.com\n");
client.print(var);
```

Para cerrar la comunicación, finalmente utilizamos

```
if (!client.connected()) {client.stop();}
if (client.available()) {char c = client.read();}

client.flush();
client.stop();
```


VI

Capítulo 6.

Conclusiones y Líneas futuras

En este capítulo se detallan las conclusiones obtenidas tras la finalización del trabajo, así como las líneas para una futura mejora y ampliación del mismo.

Resultados obtenidos

En la Figura 6.1 y 6.2 se muestran dos ejemplos de gráficas comparativas entre los resultados obtenidos en el laboratorio (izquierda) y la finca Tomás Ferro (derecha).

La gráfica superior representa el peso del lisímetro (de la maceta) frente al tiempo. Como se puede observar, se realizaba un riego al día, que provoca un incremento instantáneo del peso de la maceta. El agua cala la tierra hasta que supera su capacidad de campo, tras el cual se produce un lixiviado hasta estabilizarse el peso (todo el agua que sobra a la tierra discurre por el suelo hasta caer en la bola de drenaje).

En la gráfica inferior muestra la evolución del volumen de agua regada, cuyos escalones se corresponden con los picos de incremento de peso en el lisímetro.

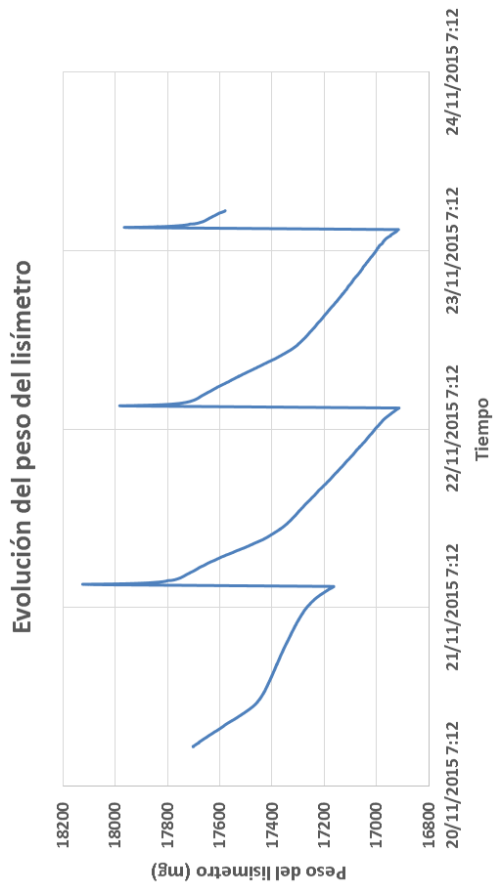
Como se puede apreciar, la gráfica obtenida en la finca presenta más irregularidades que la del laboratorio. La presencia de fenómenos naturales como el viento y lluvia, y animales (pájaros), generan un “ruido” en las medidas. En la gráfica del peso podemos observar un crecimiento de peso “inusual” hasta las 00:00 horas del día 22-12 que puede corresponderse posiblemente con un riego inesperado debido a la lluvia. Habría que contrastar los datos obtenidos con alguna estación meteorológica cercana.

Conclusiones

Los resultados obtenidos han sido ampliamente satisfactorios, ya que se han obtenido datos coherentes de la evolución del peso y riegos, tanto en el laboratorio como en la finca. Se han cumplido todos los objetivos puesto que se han diseñado varios prototipos que han funcionado correctamente y han sido validados en campo, en tres configuraciones diferentes.

El sistema ha demostrado poder obtener las lecturas de las variables que intervienen directamente en los balances hídricos, permitiendo en un futuro determinar la evapotranspiración y cumpliendo así con su objetivo. Utilizando componentes de bajo coste económico y una plataforma open-source como es Arduino, se puede crear un sistemas de instrumentación de gran precisión para monitorizar variables agrícolas. Con estos datos se puede elaborar una estrategia de gestión y una toma de decisiones para suministrar a los cultivos la cantidad de agua adecuada.

LABORATORIO



Evolución de riego (Caudalímetro)

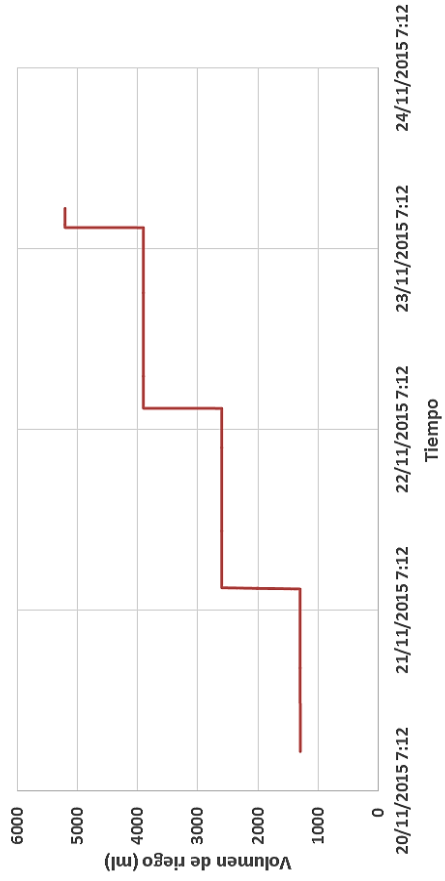
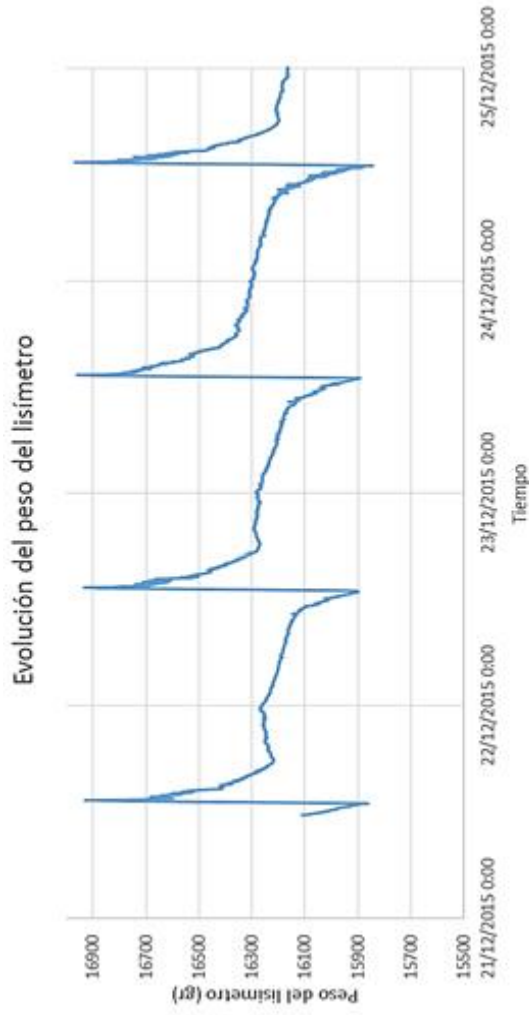


Figura 6.1: Resultados obtenidos en el Laboratorio

FINCA TOMÁS FERRO



Evolución del riego - Caudalímetro

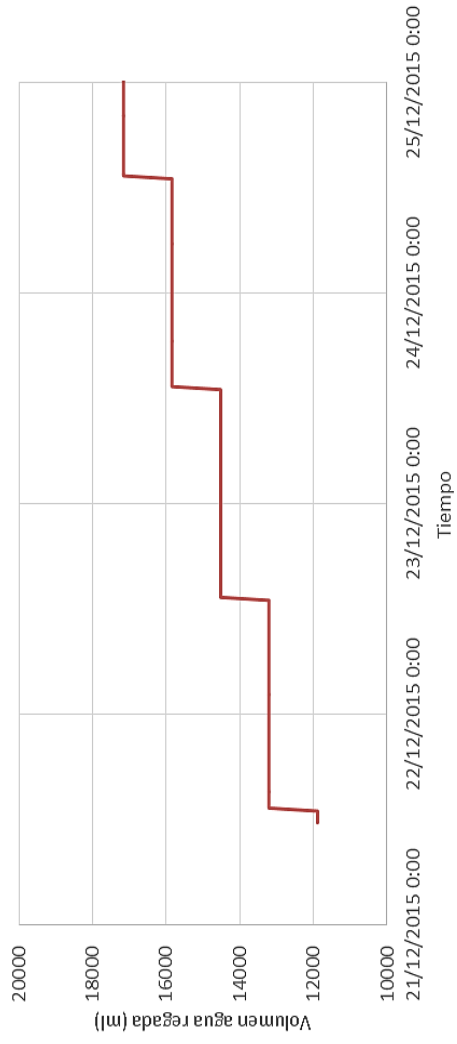


Figura 6.2: Resultados obtenidos en la finca Tomás Ferro

Líneas futuras

A continuación se detallan los puntos que se ha considerado de interés para desarrollar una futura mejora o ampliación del proyecto:

- El primer aspecto a mejorar sería dejar de utilizar el programador de riego externo para los riegos e incorporar al programa un riego por lisimetría. Nuestro sistema iría actualizando nuestra capacidad de campo conforme crece la planta, y solo realizaría riegos precisos cuando la maceta perdiese una determinada cantidad de peso en agua. Este punto sería especialmente interesante puesto que estaríamos incluyendo en sistema unos cálculos para determinar el balance hídrico y permitiese de esta manera regar de manera óptima.
- Para un futuro, sería interesante eliminar el segundo Arduino Uno utilizado para los Modelos B y C y utilizar solamente el del Arduino de Adquisición, creando un único datalogger capaz de hacer las lecturas y almacenarlas en el mismo dispositivo.

VII

Capítulo 7. **Bibliografía**

En este séptimo capítulo se muestran las referencias bibliográficas que han sido consultadas para la realización del trabajo.

Al-Karadsheh, E., H. Sourell, and C. Sommer

2002 Potential of precision irrigation: First results. *In* Conference: Agricultural Engineering 2002 Pp. 151–156. Dusseldorf: V D I-V D E - Verlag Gmbh.

Allen, Richard G, and Organización de las Naciones Unidas para la Agricultura y la Alimentación

2006 Evapotranspiración del cultivo: guías para la determinación de los requerimientos de agua de los cultivos. Roma: Organización de las Naciones Unidas para la Agricultura y la Alimentación.

Bouwer, H.

2000 Integrated Water Management: Emerging Issues and Challenges. *Agricultural Water Management* 45(3): 217–228.

Chávez, José L., Francis J. Pierce, Todd V. Elliott, and Robert G. Evans

2010 A Remote Irrigation Monitoring and Control System for Continuous Move Systems. Part A: Description and Development. *Precision Agriculture* 11(1): 1–10.

Fernández, Darío E., and others

2012 *Cydia Pomonella* (L.)(LEPIDOPTERA: TORTRICIDAE). Aspectos de Su Taxonomía, Comportamiento Y Monitoreo Aplicados a Programas de Control En Grandes áreas.

<http://www.tdx.cat/bitstream/handle/10803/94521/Tdf1de1.pdf?sequence=1>, accessed August 17, 2016.

González, María

2015 Xataka.com. Xataka. <http://www.xataka.com/aplicaciones/windows-10-y-arduino-se-llevaran-muy-bien-para-hacerse-fuertes-en-el-futuro-del-internet-de-las-cosas>, accessed August 19, 2016.

Molina-Martinez, J. M., P. J. Navarro, M. Jimenez, et al.

2012 VIPMET: New Real-Time Data Filtering–Based Automatic Agricultural Weather Station. *Journal of Irrigation and Drainage Engineering* 138(9): 823–829.

Opensource.com

N.d. Opensource.com. <https://opensource.com/>, accessed August 19, 2016.

Ortega, Rodrigo, Luis Flores, CRI Quilamapu INIA, Departamenyo de Recursos Naturales, and Medio Ambiente

1999a Agricultura de Precisión: Introducción Al Manejo Sitio-Específico. Ministerio de Agricultura, Instituto de Investigaciones Agropecuarias. CRI Quilamapu.(Chile): 13–46.

1999b Agricultura de Precisión: Introducción Al Manejo Sitio-Específico. Ministerio de Agricultura, Instituto de Investigaciones Agropecuarias. CRI Quilamapu.(Chile): 13–46.

Ruiz-Peñalver, L., J.A. Vera-Repullo, M. Jiménez-Buendía, I. Guzmán, and J.M. Molina-Martínez

2015 Development of an Innovative Low Cost Weighing Lysimeter for Potted Plants: Application in Lysimetric Stations. *Agricultural Water Management* 151: 103–113.

Santa Olalla Mañas, F., López Fuster, P. and Calera Belmonte, A. (2000). *Agua y agronomía*. Mundi-Prensa.

VIII

Capítulo 8. **Anexos**

En este último capítulo se detallarán aquellos aspectos relevantes del trabajo que debido a su naturaleza, necesitan una mención aparte.

Anexo I

Funciones del Arduino de Adquisición

Este apartado pretende describir las funciones principales utilizadas por el Arduino de Adquisición para configurar y en general, comunicarse con el convertidor CS5534-ASZ, así como para controlar los demás elementos del lisímetro. Se han mostrado por orden de aparición en el código. El código completo se puede encontrar en el CD adjuntado con el TFE, en el apartado CD/Códigos/Arduino_Adquisición /.

- **ADreset**

Se utiliza reiniciar el AD y resetear la comunicación. Para resetear la comunicación tenemos que enviar 15 veces el comando FF hexadecimal para resetear el puerto serie, tras el cual enviamos el comando SYNC0 (FE).

Para resetear el AD primero escribimos el bit RS = 1 en el Registro de Configuración (Configuration Register), y a continuación lo volvemos a escribir el bit a RS = 0.

- **ADConfig**

Sirve para configurar los parámetros del CS5534-ASZ. Primeramente ponemos el bit FSR = 1 del Registro de Configuración para activar el filtro a 50Hz.

A continuación asignamos los diferentes Channel_Setup con el canal físico correspondiente, además de configurarle la ganancia, velocidad de muestreo, tipo de alimentación (bipolar / unipolar); asignamos el Channel_Setup1 con el canal físico 1, Channel_Setup2 con el 2, Channel_Setup3 con el 3, y Channel_Setup4 con el 4.

- **Read_AD_Setup1**

Función que se utiliza para realizar una lectura al canal físico asignado al Channel_Setup1 (en este caso al canal 1 por tanto a la célula de carga 1). Para ello, tras enviar el comando correspondiente al AD, éste nos envía un byte para confirmar la instrucción y a continuación 4 bytes, los 3 primeros con el resultado del peso de la célula, y el último es el llamado “cola de conversión” que contiene información acerca de posibles errores durante la medición.

Es aplicable a Read_AD_Setup1, Read_AD_Setup2, Read_AD_Setup3 y Read_AD_Setup4.

- **Volumen_agua**

Esta función es a la que está vinculada la interrupción del pin2, encargado de leer los pulsos del caudalímetro. A medida que recibe pulsos de caudal, los almacena y más tarde de ahí se obtendrá la transformación a mililitros por segundo.

- **Toma_datos**

Encargada de tomar lecturas de peso cada 3 segundos. Si se encuentra dentro de nuestros límites rígidos, se almacena.

- **Guardado_60**

Esta función se repite cada 60 segundos. Realiza la media de las lecturas de peso tomadas hasta este momento, realiza su media aritmética y las envía por el puerto serial junto con las demás variables de interés al dispositivo de recepción (LabView o Arduino de Adquisición).

Esta función se activa en el modo normal, cuando no está regando ni drenando agua. Cuando la variación de peso en el tiempo es pequeña.

- **Guardado_9**

Función cuyo objetivo es el mismo que el anterior, salvo que se repite cada 9 segundos. Esta función se activa en modo riego, cuando el lisímetro está siendo regado o está drenando, de manera que se pueden registrar variaciones de peso en pequeños espacios de tiempo y así obtener datos más significativos.

- **Vaciado_bola**

Se encarga de gestionar el vaciado de la bola de drenaje mediante la activación de las electroválvulas. Para ello usa sus correspondientes temporizadores.

- **Espera_drenaje**

Función que incluye un temporizador para determinar el tiempo que hay que esperar tras el riego para permitir a la maceta drenar agua de manera adecuada.

- **Control_agua**

Sirve para registrar el caudal circulante para detectar posibles fugas en el sistema de alimentación

- **Enviar**

Esta función se utiliza para enviar de manera cómoda variables de 4 bytes (unsigned long) mediante la instrucción *Serial.write* que solo envía 1.

Anexo II

Manejo del Arduino de Adquisición

En este apartado se pretende mostrar la serie de menús y submenús creados dentro de un Sketch de Arduino para facilitar enormemente el aprendizaje del CS5534-ASZ, así como la apertura de las electroválvulas y lecturas del caudalímetro. Este primer código se utilizó primeramente para comenzar a usar el convertidor y así explorar sus múltiples características, configuraciones y funciones. A continuación, podemos ver el menú principal que se muestra nada más comenzar el programa.

```

|-----|
|                |MAIN MENU|                |
|-----|
| 1) Leer AD_Setup           | 5) Initialize AD          |
| 2) Read/Write Registers    | 6) Reset AD              |
| 3) Perform Calibration     | 7) COMMAND + Read_DATA  |
| 4) Read Global water volume| 8) COMMAND + Write_DATA |
|
| o) Open Valve              |
| c) Close Valve              |
|-----|

```

En el menú principal encontramos los apartados más importantes del programa que se explicarán a continuación:

- 1) Leer AD Setup. Realiza una lectura de los 4 canales físicos del CS5534-ASZ. Cada uno de los canales está configurado para un Setup diferente.

```

Lectura en gramos
Kg aprox: 109255.765 , Valor recibido: 16763592 , Cola conversion: 0
Kg aprox: 114150.929 , Valor recibido: 16763600 , Cola conversion: 0
Kg aprox: 121547.382 , Valor recibido: 16763382 , Cola conversion: 0
Kg aprox: 36462.054 , Valor recibido: 16763600 , Cola conversion: 0

          Canal 1 = 109255.77      , Canal 2 = 114150.93
          Canal 3 = 121547.39     , Canal 4 = 36462.05

peso total = 381416.12

```

De cada canal nos muestra primeramente el peso aproximado en gramos, el valor digital obtenido del AD y la cola de conversión (cuando es diferente a 0 indica un error de lectura). A su vez, el peso total suma de las 3 células de la estructura.

- 2) Read/Write Registers. Es el submenú donde se muestran los valores de configuración. Primeramente nos muestra el valor actual en binario y hexadecimal (facilita su comprensión) del "Configuration Register", "Channel Setup 1 register", "Channel Setup 2 register", "Gain Registers" y "Offset Register 1".

```

1- Write Conf.Register      3- Write Channel-Setup Registers      5- Read All Offset Register
2- Write Gain Registers    4- Write Offset Register 1            6- Write All Offset Register
.- EXIT

Conf. Register:           HEX = 0:0:0:0                BIN = 0:0:0:0
Channel-Setup Reg1.:     HEX = 0:0:0:0                BIN = 0:0:0:0
Channel-Setup Reg2.:     HEX = 0:0:0:0                BIN = 0:0:0:0
Gain Registers:          HEX = 1:0:0:0                BIN = 1:0:0:0
Offset Register 1:       HEX = 0:0:0:0                BIN = 0:0:0:0
    
```

Una vez mostrados los registros, permite modificar estos registros e ingresar nuevas configuraciones, así como volver al menú principal. Los valores obtenidos en la imagen superior corresponden al AD desconfigurado. A continuación podemos ver el mismo submenú con el AD debidamente configurado:

```

1- Write Conf.Register      3- Write Channel-Setup Registers      5- Read All Offset Register
2- Write Gain Registers    4- Write Offset Register 1            6- Write All Offset Register
.- EXIT

Conf. Register:           HEX = 0:8:0:0                BIN = 0:1000:0:0
Channel-Setup Reg1.:     HEX = 32:40:72:40           BIN = 110010:1000000:1110010:1000000
Channel-Setup Reg2.:     HEX = B2:40:F2:40           BIN = 10110010:1000000:11110010:1000000
Gain Registers:          HEX = 1:0:0:0                BIN = 1:0:0:0
Offset Register 1:       HEX = 0:0:0:0                BIN = 0:0:0:0
    
```

- 3) Perform Calibration. Entre las cualidades del CS5534-ASZ está que permite realizar calibraciones de offset y ganancia de manera automática o mediante medio externo. En este caso, solo hemos incluido las calibraciones en el Canal 1:

```

1- Setup 1 Self-Offset Calibration      3- Setup 1 System-Offset Calibration
2- Setup 1 Self-Gain Calibration        4- Setup 1 System-Gain Calibration
.- EXIT
    
```

Si realizamos consecutivamente las 4 opciones encontraremos una respuesta parecida a esta:

```

Offset Register 1:    HEX = 0:2:A5:0          BIN = 0:10:10100101:0
Done...

Gain Registers:     HEX = 0:B6:E0:61        BIN = 0:10110110:11100000:1100001
Done...

Offset Register 1:  HEX = FF:E2:B8:0          BIN = 11111111:11100010:10111000:0
Done...

Gain Registers:     HEX = 0:45:C1:25        BIN = 0:1000101:11000001:100101
Done...
    
```

- 4) Read global water volume. Realiza una lectura de la cantidad de agua regada hasta ese momento. Es especialmente útil para la calibración del caudalímetro y comprobar el funcionamiento del mismo. La señal del caudalímetro, al ser una interrupción por pulsos, la variable contador (cont) muestra el número de pulsos, y a la izquierda la transformación a milímetros mediante el factor de conversión.

```

Volumen global agua = 0.00 mL ,      cont= 0
Volumen global agua = 26.54 mL ,     cont= 2573
Volumen global agua = 34.24 mL ,     cont= 3320
    
```

- 5) Initialize AD. Nada más iniciar el programa, y después de un reseteo, el convertidor AD se encuentra desconfigurado. Esta opción sirve para inicializar el convertidor apropiadamente. Esta transición se puede observar en el punto número 2 de este anexo. Una vez configurado nos lo confirma con el mensaje siguiente:

```

Done...
    
```

- 6) Reset AD. Esta opción realiza un reseteo completo del CS5534-ASZ a un estado inicial. Para confirmarlo, mostramos el *Configuration Register* antes y después del reseteo.

```

Configuration Register despues de reset: 10:0:0:0
Configuration Register: 0:0:0:0
Done...
    
```

- 7) Command + Read Data. Para poder comunicarse con el CS5534-ASZ, se le necesita enviar primeramente un comando de 8 bits que hemos denominado "Command". Este byte contiene las instrucciones para el AD, y en el caso de que sea un comando de lectura a continuación el Arduino realizará una lectura de los 32 bits que envía el AD. En la siguiente imagen observamos el comando de lectura introducido, y la respuesta del AD en formato hexadecimal y binario.

```
IN: COMMAND 8 BITS

Comando introducido= 11101
Respuesta AD    HEX = B2:40:F2:40    BIN = 10110010:10000000:11110010:1000000
```

- 8) Command + Write Data. Si por el contrario, el comando es de escritura, después de enviar el byte de comando necesitamos enviar 32 bits con la información que queremos registrar. Para facilitarlos, se ha dividido el ingreso de 32 bits en 4 bytes independientes. Además, se interpretará el carácter "." como 4 bits con valor 0 consecutivos, ahorrándonos bastante tiempo para tareas repetitivas.

```
IN: COMMAND 8 BITS

Comando introducido= 10000000

IN: DATA 32 BITS      . = 4 x empty bits

Introduzca byte numero 1: 00010001
Introduzca byte numero 2: 00000000
Introduzca byte numero 3: 00010001
Introduzca byte numero 4: 00000000
data final= 10001 : 0 : 10001 : 0
```

- 9) Valve Open / Closed. A la hora de realizar las medidas de riego y caudal, este apartado permite abrir y cerrar la electroválvula de manera manual. Además, durante el tiempo en que está abierta, muestra el caudal circulante en milímetros por segundo, y el volumen regado total entre otros:

```
Valve OPEN...
caudal: 0.33 ml/s , volumen global: 2.09 , volumen ahora: 2.09 , volumen2: 0.00 , tiempo_ahora: 6356
caudal: 0.38 ml/s , volumen global: 2.48 , volumen ahora: 0.38 , volumen2: 2.09 , tiempo_ahora: 1004
caudal: 0.24 ml/s , volumen global: 2.71 , volumen ahora: 0.24 , volumen2: 2.48 , tiempo_ahora: 1004
caudal: 0.22 ml/s , volumen global: 2.93 , volumen ahora: 0.22 , volumen2: 2.71 , tiempo_ahora: 1003
caudal: 0.22 ml/s , volumen global: 3.15 , volumen ahora: 0.22 , volumen2: 2.93 , tiempo_ahora: 1004
caudal: 0.29 ml/s , volumen global: 3.43 , volumen ahora: 0.29 , volumen2: 3.15 , tiempo_ahora: 1002
caudal: 0.20 ml/s , volumen global: 3.63 , volumen ahora: 0.20 , volumen2: 3.43 , tiempo_ahora: 1004

Valve CLOSED...
```


Anexo III

Pruebas de los convertidores AD

En este anexo se incluyen aquellas pruebas realizadas a los convertidores analógico digitales CS5534-ASZ y HX711 para seleccionar de manera justificada el convertidor más conveniente así como su configuración y tensión de trabajo.

Prueba 1. Calibradores de precisión

En este experimento se hizo uso de dos calibradores de precisión en el laboratorio. Con el primero de ellos, se alimenta externamente el convertidor AD con diferentes tensiones y configuraciones (bipolar / unipolar).

a) Unipolar:

(0 – 4.5 V) (0 – 4.75 V) (0 – 5 V) (0 – 5.25 V) (0 – 5.5 V)

b) Bipolar:

(-2.5 / +2.5 V) (-2.75 / +2.75 V) (-3 / +3 V)

Con el segundo calibrador, se simula las diferentes tensiones que podría generar una célula de carga en la entrada. Se divide el rango teórico en el que va a trabajar la célula. Las células generan una salida máxima de 10mV, siendo su sensibilidad de 2mV/ V. Por tanto, se ha diseñado el experimento con las siguientes señales de entrada: 0, 2, 4, 6, 8 y 10mV.

El objetivo del experimento es determinar cómo varía la precisión de las lecturas del AD en función de la tensión de alimentación.

Se han hecho pruebas tanto con el HX711 como con el CS5534.

Para convertir las lecturas digitales a analógicas, se han usado la ecuación (Ganancia = 128):

$$mV = \frac{V_{dc} \cdot lectura \cdot 1000}{(2^{24} - 1) * Ganancia}$$

CS5534-ASZ Configuración Unipolar

Tensión de alimentación del AD: Teórica y Real Tensión para simular la célula de carga Lecturas del AD: Valor digital Lecturas del AD: Valor analógico Error de lectura

| 4,5 | Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
|--------|---------|-----------|-----------|-----------|-----------|-----------|--------|--------|---------|------------|
| 4,4929 | 0 | 4.782 | 4.707 | 4.481 | 4.628 | 4.650 | 0,010 | - | | |
| 4,4929 | 2 | 956.755 | 955.883 | 956.340 | 956.899 | 956.469 | 2,001 | 1,991 | - | 0,009 |
| 4,4929 | 4 | 1.924.233 | 1.922.215 | 1.922.512 | 1.921.826 | 1.922.697 | 4,023 | 4,013 | - | 0,013 |
| 4,4929 | 6 | 2.886.320 | 2.890.284 | 2.889.716 | 2.890.351 | 2.889.168 | 6,045 | 6,035 | - | 0,035 |
| 4,4929 | 8 | 3.857.180 | 3.855.807 | 3.861.837 | 3.857.521 | 3.857.521 | 8,071 | 8,061 | - | 0,061 |
| 4,4929 | 10 | 4.822.048 | 4.821.715 | 4.821.836 | 4.822.155 | 4.821.939 | 10,088 | 10,079 | - | 0,079 |

| 4,75 | Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
|--------|---------|-----------|-----------|-----------|-----------|-----------|--------|--------|---------|------------|
| 4,7426 | 0 | 3.998 | 3.951 | 4.018 | 3.720 | 3.922 | 0,009 | - | | |
| 4,7426 | 2 | 912.319 | 912.708 | 913.173 | 912.573 | 912.693 | 2,016 | 2,007 | - | 0,007 |
| 4,7426 | 4 | 1.827.674 | 1.827.896 | 1.828.002 | 1.827.749 | 1.827.830 | 4,037 | 4,028 | - | 0,028 |
| 4,7426 | 6 | 2.742.299 | 2.742.824 | 2.742.980 | 2.743.062 | 2.742.791 | 6,057 | 6,049 | - | 0,049 |
| 4,7426 | 8 | 3.657.258 | 3.658.239 | 3.657.467 | 3.657.592 | 3.657.639 | 8,078 | 8,069 | - | 0,069 |
| 4,7426 | 10 | 4.572.915 | 4.573.000 | 4.573.074 | 4.572.627 | 4.572.904 | 10,099 | 10,090 | - | 0,090 |

| 5 | Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
|--------|---------|-----------|-----------|-----------|-----------|-----------|--------|--------|---------|------------|
| 4,9923 | 0 | 3.551 | 3.523 | 3.433 | 3.433 | 3.485 | 0,008 | - | | |
| 4,9923 | 2 | 866.294 | 868.830 | 867.360 | 867.223 | 867.427 | 2,017 | 2,008 | - | 0,008 |
| 4,9923 | 4 | 1.734.995 | 1.735.467 | 1.736.156 | 1.736.392 | 1.735.753 | 4,035 | 4,027 | - | 0,027 |
| 4,9923 | 6 | 2.605.371 | 2.606.017 | 2.606.013 | 2.609.105 | 2.606.627 | 6,060 | 6,052 | - | 0,052 |
| 4,9923 | 8 | 3.473.893 | 3.474.651 | 3.474.549 | 3.474.794 | 3.474.472 | 8,077 | 8,069 | - | 0,069 |
| 4,9923 | 10 | 4.344.231 | 4.343.969 | 4.343.817 | 4.343.760 | 4.343.944 | 10,098 | 10,090 | - | 0,090 |

| 5,25 | Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
|--------|---------|-----------|-----------|-----------|-----------|-----------|--------|--------|---------|------------|
| 5,2421 | 0 | 4.225 | 3.929 | 3.807 | 3.742 | 3.926 | 0,010 | - | | |
| 5,2421 | 2 | 825.367 | 824.606 | 825.179 | 825.693 | 825.211 | 2,014 | 2,005 | - | 0,005 |
| 5,2421 | 4 | 1.653.507 | 1.652.366 | 1.651.998 | 1.654.842 | 1.653.178 | 4,035 | 4,026 | - | 0,026 |
| 5,2421 | 6 | 2.481.533 | 2.479.917 | 2.480.337 | 2.479.904 | 2.480.423 | 6,055 | 6,045 | - | 0,045 |
| 5,2421 | 8 | 3.301.639 | 3.306.935 | 3.308.126 | 3.308.743 | 3.306.361 | 8,071 | 8,061 | - | 0,061 |
| 5,2421 | 10 | 4.135.599 | 4.137.109 | 4.135.391 | 4.135.977 | 4.136.019 | 10,096 | 10,087 | - | 0,087 |

| 5,5 | Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
|--------|---------|-----------|-----------|-----------|-----------|-----------|--------|--------|---------|------------|
| 5,4919 | 0 | 3.768 | 3.968 | 3.528 | 3.460 | 3.681 | 0,009 | - | | |
| 5,4919 | 2 | 789.230 | 789.116 | 789.216 | 789.222 | 789.196 | 2,018 | 2,009 | - | 0,009 |
| 5,4919 | 4 | 1.578.000 | 1.578.643 | 1.577.762 | 1.578.118 | 1.578.131 | 4,036 | 4,026 | - | 0,026 |
| 5,4919 | 6 | 2.368.225 | 2.369.124 | 2.369.163 | 2.365.440 | 2.367.988 | 6,056 | 6,046 | - | 0,046 |
| 5,4919 | 8 | 3.157.444 | 3.157.881 | 3.158.081 | 3.161.359 | 3.158.691 | 8,078 | 8,069 | - | 0,069 |
| 5,4919 | 10 | 3.947.888 | 3.948.604 | 3.948.575 | 3.948.705 | 3.948.443 | 10,098 | 10,088 | - | 0,088 |

Tabla 7: Prueba 1. CS5534 Unipolar. Resultados

A partir de los datos anteriores, se ha representado en forma de gráfica y podemos observar como todas las configuraciones tienen una precisión excelente. De manera que dentro de esta configuración, la tensión de alimentación no afecta de manera significativa a las lecturas.

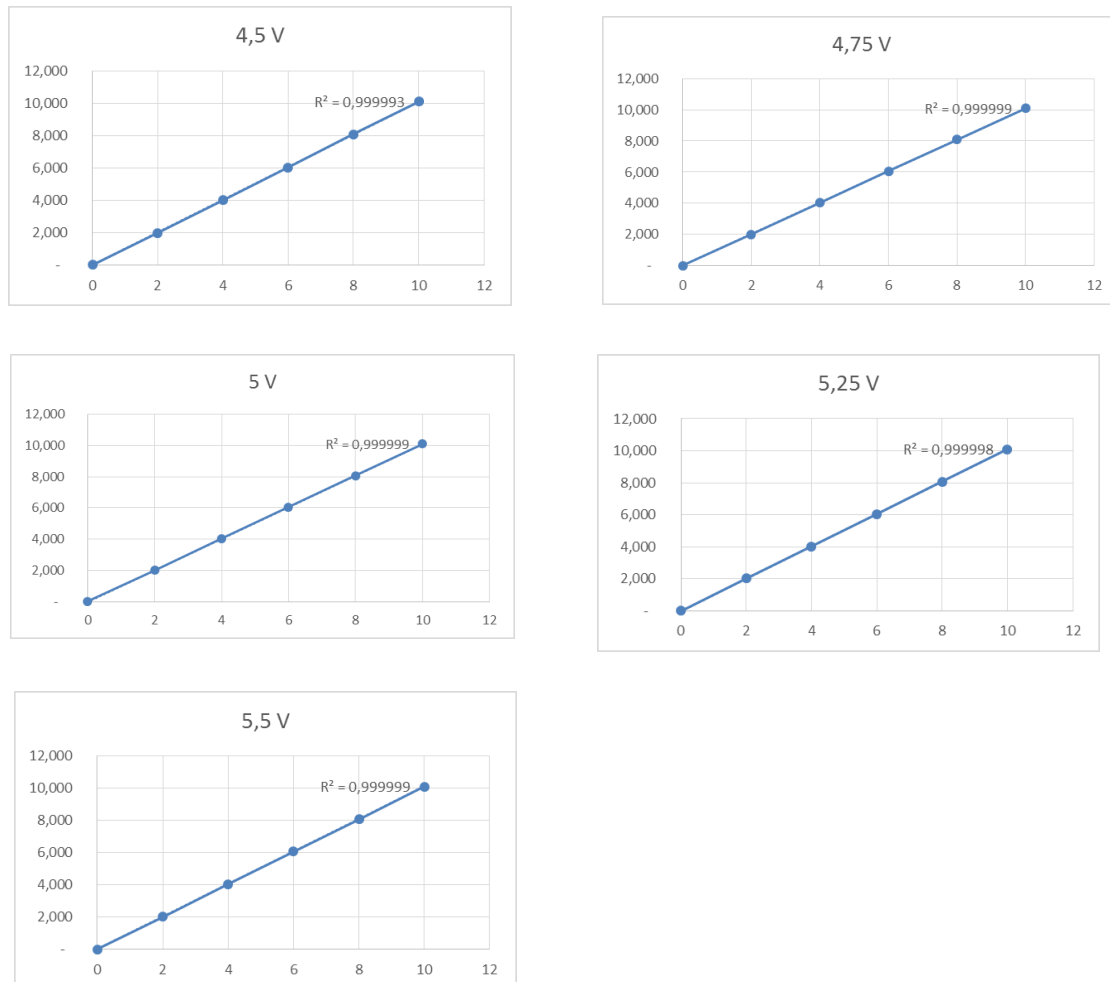


Figura II.1: Prueba 1. CS5534 Unipolar. Resultados

CS5534-ASZ Configuración bipolar

Ahora, igual que en el apartado anterior, realizamos el mismo experimento pero configurando el CS5534 para alimentación bipolar.

| +/- 2,5 V | | | | | | | | | |
|-----------|----------|-----------|-----------|-----------|-----------|-----------|--------|---------|------------|
| Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
| 5,0129 | 0 | 2.101 | 2.095 | 2.185 | 2.465 | 2.212 | 0,010 | - | |
| 5,0129 | 2 | 434.806 | 434.677 | 434.851 | 434.938 | 434.818 | 2,030 | 2,020 | 0,020 |
| 5,0129 | 4 | 867.850 | 867.676 | 867.856 | 867.811 | 867.798 | 4,051 | 4,041 | 0,041 |
| 5,0129 | 6 | 1.301.158 | 1.300.848 | 1.301.220 | 1.300.861 | 1.301.022 | 6,074 | 6,064 | 0,064 |
| 5,0129 | 8 | 1.734.383 | 1.734.140 | 1.734.140 | 1.734.476 | 1.734.285 | 8,097 | 8,086 | 0,086 |
| 5,0129 | 10 | 2.168.005 | 2.167.873 | 2.168.165 | 2.168.181 | 2.168.056 | 10,122 | 10,112 | 0,112 |

| +/- 2,75 V | | | | | | | | | |
|------------|----------|-----------|-----------|-----------|-----------|-----------|--------|---------|------------|
| Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
| 5,5066 | 0 | 1.741 | 1.534 | 1.728 | 1.871 | 1.719 | 0,009 | - | |
| 5,5066 | 2 | 394.946 | 395.769 | 395.013 | 395.212 | 395.235 | 2,027 | 2,018 | 0,018 |
| 5,5066 | 4 | 789.066 | 789.061 | 789.041 | 789.044 | 789.053 | 4,047 | 4,038 | 0,038 |
| 5,5066 | 6 | 1.183.074 | 1.183.185 | 1.184.575 | 1.183.199 | 1.183.508 | 6,070 | 6,061 | 0,061 |
| 5,5066 | 8 | 1.578.292 | 1.578.038 | 1.577.992 | 1.577.975 | 1.578.074 | 8,093 | 8,084 | 0,084 |
| 5,5066 | 10 | 1.973.222 | 1.972.842 | 1.972.843 | 1.972.829 | 1.972.934 | 10,118 | 10,109 | 0,109 |

| +/- 3 V | | | | | | | | | |
|---------|----------|-----------|-----------|-----------|-----------|-----------|--------|---------|------------|
| Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
| 6,0083 | 0 | 2.642 | 3.040 | 3.038 | 3.034 | 2.939 | 0,016 | - | |
| 6,0083 | 2 | 362.472 | 362.608 | 361.390 | 361.475 | 361.986 | 2,026 | 2,009 | 0,009 |
| 6,0083 | 4 | 722.808 | 722.926 | 722.974 | 723.042 | 722.938 | 4,045 | 4,029 | 0,029 |
| 6,0083 | 6 | 1.085.314 | 1.084.994 | 1.084.432 | 1.084.433 | 1.084.793 | 6,070 | 6,054 | 0,054 |
| 6,0083 | 8 | 1.445.553 | 1.445.460 | 1.445.460 | 1.445.559 | 1.445.508 | 8,089 | 8,072 | 0,072 |
| 6,0083 | 10 | 1.807.375 | 1.807.271 | 1.807.166 | 1.807.259 | 1.807.268 | 10,113 | 10,096 | 0,096 |

Tabla 8: Prueba 1. CS5534 Bipolar. Resultados

A partir de los datos anteriores, se ha representado en forma de gráfica y se puede observar como todas las configuraciones tienen una precisión excelente. De manera que dentro de esta configuración, la tensión de alimentación no afecta de manera significativa a las lecturas.

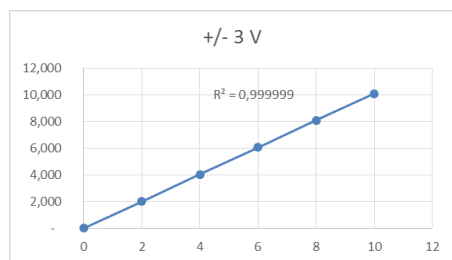
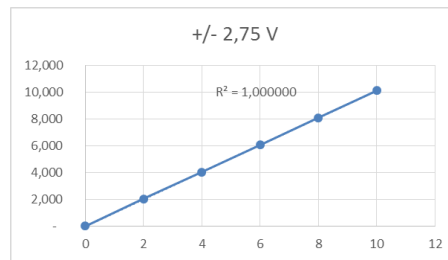
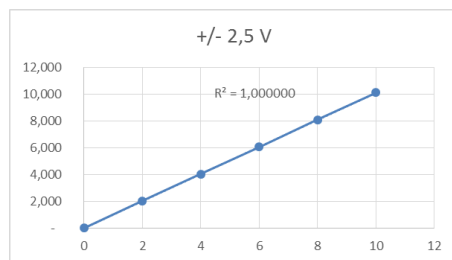


Figura II.2: Prueba 1. CS5534 Bipolar. Resultados

HX711

Ahora, igual que en el apartado anterior, realizamos el mismo experimento pero configurando con el HX711, ganancia 128.

| 4,5 | | | | | | | | | |
|---------|----------|------------|------------|------------|------------|-----------|------|---------|------------|
| Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
| 4,4972 | 0 | 8.384.432 | 8.384.633 | 8.384.577 | 8.384.594 | 4.049 | 0,0 | - | |
| 4,4972 | 2 | 9.590.422 | 9.590.329 | 9.590.538 | 9.590.434 | 1.201.823 | 2,5 | 2,5 | 0,5 |
| 4,4972 | 4 | 10.794.737 | 10.794.911 | 10.794.661 | 10.794.453 | 2.406.083 | 5,0 | 5,0 | 1,0 |
| 4,4972 | 6 | 11.999.951 | 12.000.097 | 11.999.917 | 12.000.152 | 3.611.421 | 7,6 | 7,6 | 1,6 |
| 4,4972 | 8 | 13.204.292 | 13.204.722 | 13.204.762 | 13.204.819 | 4.816.041 | 10,1 | 10,1 | 2,1 |
| 4,4972 | 10 | 14.409.872 | 14.409.803 | 14.409.608 | 14.409.507 | 6.021.090 | 12,6 | 12,6 | 2,6 |

| 4,75 | | | | | | | | | |
|---------|----------|----------|----------|----------|----------|-----------|------|---------|------------|
| Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
| 4,7468 | 0 | 8385139 | 8385057 | 8385064 | 8384864 | 3.577 | 0,0 | - | |
| 4,7468 | 2 | 9517158 | 9517037 | 9517261 | 9517303 | 1.128.582 | 2,5 | 2,5 | 0,5 |
| 4,7468 | 4 | 10648120 | 10648036 | 10648070 | 10647921 | 2.259.429 | 5,0 | 5,0 | 1,0 |
| 4,7468 | 6 | 11779851 | 11780008 | 11779708 | 11779515 | 3.391.163 | 7,5 | 7,5 | 1,5 |
| 4,7468 | 8 | 12910517 | 12910810 | 12910623 | 12910896 | 4.522.104 | 10,0 | 10,0 | 2,0 |
| 4,7468 | 10 | 14042991 | 14042978 | 14042801 | 14042872 | 5.654.303 | 12,5 | 12,5 | 2,5 |

| 5 | | | | | | | | | |
|---------|----------|----------|----------|----------|----------|-----------|------|---------|------------|
| Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
| 4,9964 | 0 | 8385509 | 8385541 | 8385300 | 8385274 | 3.202 | 0,0 | - | |
| 4,9964 | 2 | 9452612 | 9452293 | 9452796 | 9452718 | 1.063.997 | 2,5 | 2,5 | 0,5 |
| 4,9964 | 4 | 10518656 | 10518487 | 10518008 | 10518359 | 2.129.770 | 5,0 | 5,0 | 1,0 |
| 4,9964 | 6 | 11584711 | 11584773 | 11584989 | 11584774 | 3.196.204 | 7,4 | 7,4 | 1,4 |
| 4,9964 | 8 | 12651023 | 12651077 | 12650871 | 12650502 | 4.262.260 | 9,9 | 9,9 | 1,9 |
| 4,9964 | 10 | 13717489 | 13717707 | 13717528 | 13717485 | 5.328.944 | 12,4 | 12,4 | 2,4 |

| 5,25 | | | | | | | | | |
|---------|----------|----------|----------|----------|----------|-----------|------|---------|------------|
| Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
| 5,2496 | 0 | 8380090 | 8379823 | 8380368 | 8380546 | 8.401 | 0,0 | - | |
| 5,2496 | 2 | 9401147 | 9400886 | 9401035 | 9400995 | 1.012.408 | 2,5 | 2,5 | 0,5 |
| 5,2496 | 4 | 10419584 | 10419495 | 10419814 | 10420049 | 2.031.128 | 5,0 | 5,0 | 1,0 |
| 5,2496 | 6 | 11439533 | 11439386 | 11439457 | 11439382 | 3.050.832 | 7,5 | 7,5 | 1,5 |
| 5,2496 | 8 | 12459036 | 12459179 | 12459135 | 12459091 | 4.070.502 | 10,0 | 10,0 | 2,0 |
| 5,2496 | 10 | 13478938 | 13478993 | 13479031 | 13478913 | 5.090.361 | 12,4 | 12,5 | 2,5 |

| 5,5 | | | | | | | | | |
|---------|----------|----------|----------|----------|----------|-----------|------|---------|------------|
| Vdc (V) | Vin (mV) | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA | mV | mV - Vo | error (mV) |
| 5,4996 | 0 | 8381249 | 8381178 | 8381026 | 8381341 | 7.410 | 0,0 | - | |
| 5,4996 | 2 | 9401267 | 9401653 | 9401570 | 9401563 | 1.012.905 | 2,6 | 2,6 | 0,6 |
| 5,4996 | 4 | 10420749 | 10420675 | 10420839 | 10420431 | 2.032.066 | 5,2 | 5,2 | 1,2 |
| 5,4996 | 6 | 11440144 | 11440460 | 11440464 | 11440776 | 3.051.853 | 7,8 | 7,8 | 1,8 |
| 5,4996 | 8 | 12460268 | 12460268 | 12460518 | 12460237 | 4.071.715 | 10,4 | 10,4 | 2,4 |
| 5,4996 | 10 | 13479248 | 13479371 | 13479811 | 13479985 | 5.090.996 | 13,0 | 13,1 | 3,1 |

Tabla 9: Prueba 1. HX711. Resultados

A primera vista se puede observar el error cometido a priori es mayor que con el CS5534.

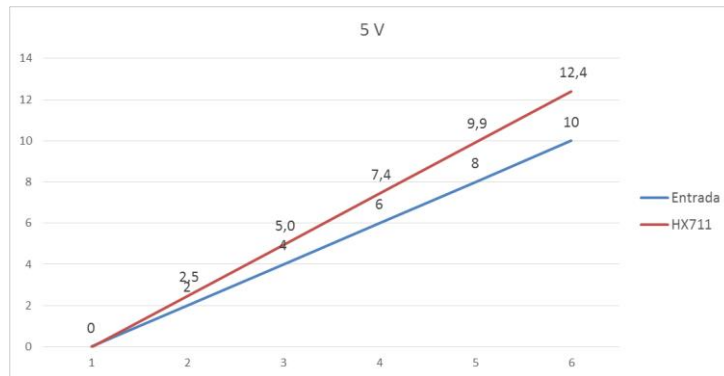


Figura II.3: Prueba1. HX711. Error cometido

No obstante, es un error que al ser proporcional es fácilmente corregible. A continuación se muestran las ecuaciones lineales de los distintos voltajes:

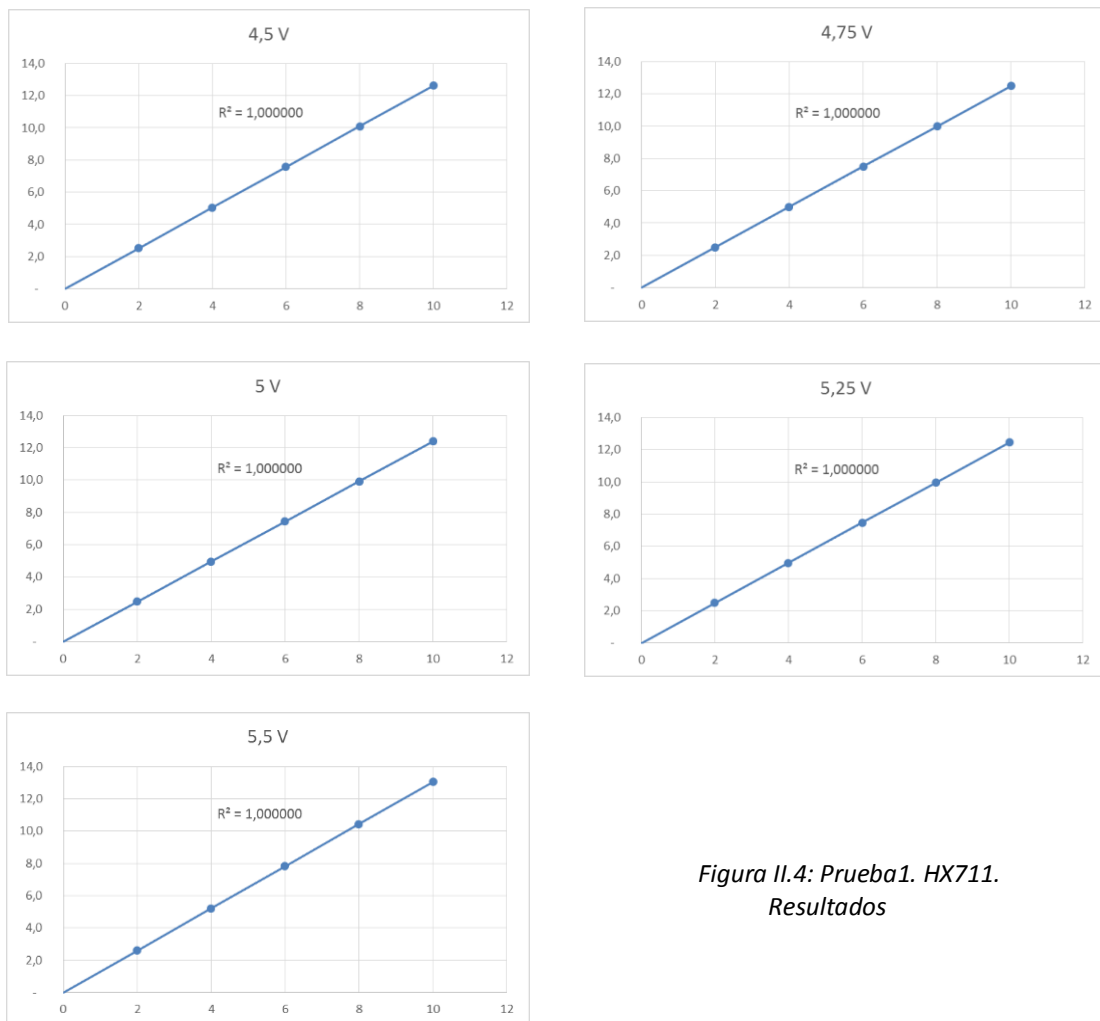


Figura II.4: Prueba1. HX711. Resultados

Prueba 2 (I). Células de carga. Alimentación desde calibrador. Subida y bajada de pesos

Para esta prueba, se ha alimentado el convertidor AD con 5 Vdc desde el calibrador del laboratorio. El objetivo de la misma es observar el comportamiento de la célula de carga frente a incrementos y decrementos de peso.

Para ello, se ha hecho uso de unas pesas calibradas. La lista de pesas disponibles en el laboratorio es la siguiente:

| Teórico (gr) | Real (gr) |
|--------------|-----------|
| | |
| 500 | 504,3 |
| 500 | 504,6 |
| 1.000 | 996,4 |
| 1.000 | 1.041,1 |
| 1.000 | 1.053,9 |
| 2.000 | 2.030,5 |
| 2.000 | 2.035,2 |
| 2.000 | 2.047,4 |
| 2.000 | 2.067,2 |
| 5.000 | 4.924,8 |
| 5.000 | 5.031,0 |
| 5.000 | 5.057,8 |

Tabla 10: Pesas calibradas disponibles

Se ha dividido el rango de la célula (30 kg) en 6 puntos: 0, 5, 10, 15, 20, 25.

Por tanto, se han combinado las pesas del laboratorio para obtener de la manera más aproximada posible, y siempre con los mismos pesos, los 6 puntos necesarios para el experimento.

A estas pesadas, además, ha habido que sumarles los 240,05 gramos del plato de pesada, de ahí que estén separadas en columnas (siendo la que incluye el plato la usada para los datos finales).

CS5534-ASZ Configuración unipolar

CS5534 subida

| Vdc = 4,9896 V | | | | | | | | |
|----------------|-----------|-----------------|------------------------------------|-----------|-----------|-----------|-----------|-----------|
| Peso teorico | | Peso real pesas | Combinacion de pesas | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
| Sin plato | - | - | sin plato | 15.244 | 15.747 | 15.142 | 14.791 | 15.231 |
| - | 240,05 | - | solo el plato | 14.791 | 49.510 | 49.112 | 48.849 | 40.566 |
| 5.000 | 5.271,05 | 5.031,00 | 5031 | 778.141 | 778.464 | 777.907 | 778.308 | 778.205 |
| 10.000 | 10.328,85 | 10.088,80 | 5057,8 + 5031,0 | 1.512.006 | 1.510.781 | 1.511.751 | 1.511.660 | 1.511.550 |
| 15.000 | 15.253,65 | 15.013,60 | 5,057,8 + 5031 + 4,924,8 | 2.223.955 | 2.223.829 | 2.223.400 | 2.223.750 | 2.223.734 |
| 20.000 | 20.392,45 | 20.152,40 | 15013,6 + 2067,2 + 2030,5 + 1041,1 | 2.967.625 | 2.967.803 | 2.967.803 | 2.967.892 | 2.967.781 |
| 25.000 | 25.528,95 | 25.288,90 | todos menos los 2 primeros | 3.856.858 | 3.855.845 | 3.856.288 | 3.856.184 | 3.856.294 |

CS5534 bajada

| Vdc = 4,9896 V | | | | | |
|----------------|-----------|-----------|-----------|-----------|-----------|
| Peso | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
| 25.288,9 | 3.856.550 | 3.856.492 | 3.856.547 | 3.856.729 | 3.856.580 |
| 20.152,4 | 2.970.071 | 2.969.918 | 2.969.890 | 2.970.264 | 2.970.036 |
| 15.013,6 | 2.225.897 | 2.225.934 | 2.225.695 | 2.226.024 | 2.225.888 |
| 10.088,8 | 1.512.639 | 1.512.298 | 1.512.551 | 1.512.479 | 1.512.492 |
| 5.031,0 | 780.333 | 780.171 | 780.218 | 780.033 | 780.189 |
| - | 51.165 | 50.781 | 50.948 | 50.867 | 50.940 |

| | | | | | |
|---|--------|--------|--------|--------|--------|
| - | 51.128 | 51.301 | 51.309 | 50.909 | 51.162 |
|---|--------|--------|--------|--------|--------|

hemos dejado reposar sin peso 5 minutos mas para ver si varia el valor de pesada

Tabla 11: Prueba 2 (I). CS5534 Unipolar. Resultados

CS5534-ASZ Configuración bipolar

CS5534 subida

| Vdc = 4,9896 V | | | | | | | | |
|----------------|-----------|-----------------|------------------------------------|-----------|-----------|-----------|-----------|-----------|
| Peso teorico | | Peso real pesas | Combinacion de pesas | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
| Sin plato | - | - | sin plato | 13.129 | 13.332 | 13.041 | 13.356 | 13.215 |
| Con plato | 240,05 | - | solo el plato | 30.541 | 30.625 | 30.776 | 30.710 | 30.663 |
| 5.000 | 5.271,05 | 5.031,00 | 5031 | 395.470 | 395.732 | 395.545 | 395.396 | 395.536 |
| 10.000 | 10.328,85 | 10.088,80 | 5057,8 + 5031,0 | 762.278 | 762.798 | 762.193 | 762.292 | 762.390 |
| 15.000 | 15.253,65 | 15.013,60 | 5,057,8 + 5031 + 4,924,8 | 1.119.225 | 1.119.137 | 1.119.257 | 1.119.499 | 1.119.280 |
| 20.000 | 20.392,45 | 20.152,40 | 15013,6 + 2067,2 + 2030,5 + 1041,1 | 1.492.959 | 1.492.845 | 1.492.925 | 1.493.006 | 1.492.934 |
| 25.000 | 25.528,95 | 25.288,90 | todos menos los 2 primeros | 1.937.334 | 1.937.334 | 1.937.633 | 1.937.334 | 1.937.409 |

CS5534 bajada

| Vdc = 4,9896 V | | | | | |
|----------------|-----------|-----------|-----------|-----------|-----------|
| Peso | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
| 25.288,9 | 1.937.638 | 1.937.724 | 1.937.764 | 1.937.714 | 1.937.710 |
| 20.152,4 | 1.492.835 | 1.492.541 | 1.492.701 | 1.492.561 | 1.492.660 |
| 15.013,6 | 1.119.794 | 1.119.262 | 1.119.619 | 1.119.005 | 1.119.420 |
| 10.088,8 | 761.945 | 762.105 | 762.223 | 762.079 | 762.088 |
| 5.031,0 | 395.186 | 395.090 | 395.011 | 395.213 | 395.125 |
| - | 29.768 | 29.651 | 29.992 | 29.722 | 29.783 |

| | | | | | |
|---|--------|--------|--------|--------|--------|
| - | 29.813 | 29.847 | 30.058 | 29.720 | 29.860 |
|---|--------|--------|--------|--------|--------|

hemos dejado reposar sin peso 5 minutos mas para ver si varia el valor de pesada

Tabla 12: Prueba 2 (I). CS5534 Bipolar. Resultados

HX711

HX711 subida

| Vdc = 4,9913 V | | Peso real | Combinacion de pesas | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|----------------|-----------|-----------|------------------------------------|------------|------------|------------|------------|---------------|
| Peso teorico | | - | sin plato | 8.409.607 | 8.409.370 | 8.409.348 | 8.409.406 | 8.409.432,75 |
| Sin plato | | - | solo el plato | 8.444.521 | 8.444.416 | 8.444.528 | 8.444.483 | 8.444.487,00 |
| 5.000 | 5.271,05 | 5.031,00 | 5031 | 9.185.718 | 9.185.836 | 9.185.721 | 9.185.602 | 9.185.719,25 |
| 10.000 | 10.328,85 | 10.088,80 | 5057,8 + 5031,0 | 9.930.246 | 9.930.111 | 9.930.258 | 9.930.186 | 9.930.200,25 |
| 15.000 | 15.253,65 | 15.013,60 | 5.057,8 + 5031 + 4.924,8 | 10.655.146 | 10.655.135 | 10.655.148 | 10.655.204 | 10.655.158,25 |
| 20.000 | 20.392,45 | 20.152,40 | 15013,6 + 2067,2 + 2030,5 + 1041,1 | 11.411.950 | 11.412.049 | 11.412.022 | 11.412.153 | 11.412.043,50 |
| 25.000 | 25.528,95 | 25.288,90 | todos menos los 2 primeros | 12.314.077 | 12.314.099 | 12.314.102 | 12.314.094 | 12.314.093,00 |

HX711 bajada

| Vdc = 4,9913 V | | | | | |
|----------------|------------|------------|------------|------------|---------------|
| Peso | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
| 25.000 | 12.314.107 | 12.314.184 | 12.314.147 | 12.314.386 | 12.314.206,00 |
| 20.000 | 11.411.974 | 11.411.972 | 11.411.919 | 11.412.117 | 11.411.995,50 |
| 15.000 | 10.655.678 | 10.655.621 | 10.655.484 | 10.655.402 | 10.655.546,25 |
| 10.000 | 9.930.559 | 9.930.538 | 9.930.676 | 9.930.576 | 9.930.587,25 |
| 5.000 | 9.186.014 | 9.185.964 | 9.185.950 | 9.185.963 | 9.185.972,75 |
| - | 8.445.051 | 8.445.232 | 8.444.941 | 8.445.037 | 8.445.065,25 |
| - | 8.444.620 | 8.444.500 | 8.444.553 | 8.444.453 | 8.444.531,50 |

hemos dejado reposar sin peso 5 minutos mas para ver si varia el valor de pesada

Tabla 13: Prueba 2 (I). HX711. Resultados

Prueba 2 (II). Células de carga. Alimentación desde calibrador. Subida y bajada de voltaje

Para esta prueba, hemos variado el voltaje de alimentación del AD para un mismo peso.

Los voltajes que se han utilizado han sido los mismos que en la prueba 1: 4.5 , 4.75 , 5 , 5.25 , 5.5V

CS5534-ASZ Configuración Unipolar

Plato sólo →

| V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|-----------|--------|----------|----------|----------|----------|--------|
| 4,5 V | 4,4904 | 54.049 | 54.538 | 55.074 | 55.246 | 54.727 |
| 4,75 V | 4,7398 | 53.751 | 53.935 | 53.827 | 54.157 | 53.918 |
| 5 V | 4,9894 | 54.316 | 53.811 | 54.183 | 54.268 | 54.145 |
| 5,25 V | 5,2398 | 58.927 | 58.844 | 59.039 | 58.971 | 58.945 |
| 5,5 V | 5,5012 | 58.889 | 58.842 | 59.109 | 59.109 | 58.987 |

8 Kg →

| V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|-----------|--------|-----------|-----------|-----------|-----------|-----------|
| 4,5 V | 4,4987 | 1.228.762 | 1.228.762 | 1.228.762 | 1.228.758 | 1.228.761 |
| 4,75 V | 4,7515 | 1.228.621 | 1.228.741 | 1.228.816 | 1.228.754 | 1.228.733 |
| 5 V | 5,0082 | 1.228.663 | 1.228.996 | 1.228.636 | 1.228.643 | 1.228.735 |
| 5,25 V | 5,2574 | 1.229.138 | 1.229.032 | 1.228.954 | 1.228.839 | 1.228.991 |
| 5,5 V | 5,5092 | 1.229.366 | 1.229.398 | 1.229.216 | 1.228.938 | 1.229.230 |

Combinacion pesas: 8.089,4 Kg

16 Kg →

| V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|-----------|--------|-----------|-----------|-----------|-----------|-----------|
| 4,5 V | 4,5089 | 2.376.495 | 2.376.743 | 2.377.138 | 2.376.774 | 2.376.788 |
| 4,75 V | 4,7496 | 2.377.035 | 2.377.221 | 2.376.860 | 2.376.694 | 2.376.953 |
| 5 V | 5,0041 | 2.377.032 | 2.376.796 | 2.376.857 | 2.376.779 | 2.376.866 |
| 5,25 V | 5,2546 | 2.377.273 | 2.377.608 | 2.377.387 | 2.376.643 | 2.377.228 |
| 5,5 V | 5,5028 | 2.377.075 | 2.376.873 | 2.376.943 | 2.377.245 | 2.377.034 |

Combinacion pesas: 16.010 Kg

24 Kg →

| V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|-----------|--------|-----------|-----------|-----------|-----------|-----------|
| 4,5 V | 4,5018 | 3.560.127 | 3.560.235 | 3.560.139 | 3.560.120 | 3.560.155 |
| 4,75 V | 4,7512 | 3.561.013 | 3.561.010 | 3.560.732 | 3.560.267 | 3.560.756 |
| 5 V | 5,0024 | 3.560.488 | 3.560.031 | 3.560.302 | 3.560.550 | 3.560.343 |
| 5,25 V | 5,2491 | 3.560.485 | 3.560.700 | 3.560.676 | 3.560.282 | 3.560.536 |
| 5,5 V | 5,501 | 3.560.755 | 3.560.353 | 3.560.133 | 3.560.755 | 3.560.499 |

Combinacion pesas: 24.190,3 Kg

Tabla 14: Prueba 2 (II). CS5534 Unipolar. Resultados

CS5534-ASZ Configuración bipolar

| Plato sólo | V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|------------|------------|--------|----------|----------|----------|----------|--------|
| | +/- 2,5 V | 5,0116 | 31.809 | 32.090 | 32.003 | 32.011 | 31.978 |
| | +/- 2,75 V | 5,5003 | 31.610 | 31.670 | 31.675 | 31.666 | 31.655 |
| | +/- 3 V | 6,0189 | 31.593 | 31.556 | 31.557 | 31.694 | 31.600 |

| 8 Kg | V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|------|------------|--------|----------|----------|----------|----------|---------|
| | +/- 2,5 V | 5,0028 | 616.719 | 616.695 | 616.937 | 616.870 | 616.805 |
| | +/- 2,75 V | 5,505 | 616.958 | 616.823 | 616.837 | 616.846 | 616.866 |
| | +/- 3 V | 6,0048 | 616.955 | 616.626 | 616.731 | 616.751 | 616.766 |

Combinacion pesas: 8.089,4 Kg

| 16 Kg | V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|-------|------------|--------|-----------|-----------|-----------|-----------|-----------|
| | +/- 2,5 V | 5,0177 | 1.190.330 | 1.190.414 | 1.190.372 | 1.190.390 | 1.190.377 |
| | +/- 2,75 V | 5,5219 | 1.190.667 | 1.190.600 | 1.190.330 | 1.190.538 | 1.190.534 |
| | +/- 3 V | 6,0295 | 1.190.224 | 1.190.240 | 1.190.011 | 1.190.112 | 1.190.147 |

Combinacion pesas: 16.010 Kg

| 24 Kg | V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|-------|------------|--------|-----------|-----------|-----------|-----------|-----------|
| | +/- 2,5 V | 5,0221 | 1.782.084 | 1.782.257 | 1.782.152 | 1.782.006 | 1.782.125 |
| | +/- 2,75 V | 5,5324 | 1.782.382 | 1.782.046 | 1.782.178 | 1.782.337 | 1.782.236 |
| | +/- 3 V | 6,024 | 1.782.165 | 1.782.158 | 1.782.202 | 1.782.263 | 1.782.197 |

Combinacion pesas: 24.190,3 Kg

Tabla 15: Prueba 2 (II). CS5534 Bipolar. Resultados

HX711

| Plato sólo | V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|------------|-----------|--------|-----------|-----------|-----------|-----------|-----------|
| | 4,5 V | 4,4904 | 8.441.945 | 8.442.110 | 8.442.173 | 8.442.087 | 8.442.079 |
| | 4,75 V | 4,7398 | 8.442.866 | 8.442.909 | 8.442.864 | 8.442.871 | 8.442.878 |
| | 5 V | 4,9894 | 8.443.453 | 8.443.403 | 8.443.333 | 8.443.436 | 8.443.406 |
| | 5,25 V | 5,2398 | 8.444.134 | 8.444.068 | 8.444.128 | 8.444.079 | 8.444.102 |
| | 5,5 V | 5,5012 | 8.435.707 | 8.433.776 | 8.435.161 | 8.433.818 | 8.434.616 |

| 8 Kg | V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|------|-----------|--------|-----------|-----------|-----------|-----------|-----------|
| | 4,5 V | 4,4987 | 9.630.289 | 9.630.314 | 9.630.418 | 9.630.460 | 9.630.370 |
| | 4,75 V | 4,7515 | 9.631.198 | 9.631.242 | 9.631.271 | 9.631.228 | 9.631.235 |
| | 5 V | 5,0082 | 9.632.351 | 9.632.213 | 9.632.204 | 9.632.135 | 9.632.226 |
| | 5,25 V | 5,2574 | 9.633.023 | 9.633.070 | 9.633.029 | 9.633.091 | 9.633.053 |
| | 5,5 V | 5,5092 | 9.631.200 | 9.629.969 | 9.630.335 | 9.631.991 | 9.630.874 |

Combinacion pesas: 8.089,4 Kg

| 24 Kg | V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|-------|-----------|--------|------------|------------|------------|------------|------------|
| | 4,5 V | 4,5018 | 11.993.080 | 11.993.140 | 11.993.100 | 11.993.222 | 11.993.136 |
| | 4,75 V | 4,7512 | 11.995.435 | 11.995.516 | 11.995.520 | 11.995.467 | 11.995.485 |
| | 5 V | 5,0024 | 11.997.521 | 11.997.540 | 11.997.484 | 11.997.535 | 11.997.520 |
| | 5,25 V | 5,2491 | 11.997.535 | 11.999.821 | 11.999.730 | 11.999.842 | 11.999.232 |
| | 5,5 V | 5,501 | 11.982.523 | 11.982.961 | 11.982.082 | 11.982.394 | 11.982.490 |

Combinacion pesas: 24.190,3 Kg

| 16 Kg | V teórico | V real | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|-------|-----------|--------|------------|------------|------------|------------|------------|
| | 4,5 V | 4,5089 | 10.794.793 | 10.794.793 | 10.794.811 | 10.794.821 | 10.794.805 |
| | 4,75 V | 4,7496 | 10.796.394 | 10.796.516 | 10.796.535 | 10.796.583 | 10.796.507 |
| | 5 V | 5,0041 | 10.797.703 | 10.797.821 | 10.797.846 | 10.797.884 | 10.797.814 |
| | 5,25 V | 5,2546 | 10.798.697 | 10.798.673 | 10.798.762 | 10.798.558 | 10.798.673 |
| | 5,5 V | 5,5028 | 10.782.980 | 10.781.618 | 10.783.142 | 10.784.067 | 10.782.952 |

Combinacion pesas: 16.010 Kg

Tabla 16: Prueba 2 (II). HX711. Resultados

Prueba 3. Calibración de células de carga. Alimentación desde Arduino

CS5534-ASZ Configuración Unipolar

CS5534 subida

| Vdc = 4,8736V | | Peso real pesas | Combinacion de pesas | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|---------------|-----------|-----------------|------------------------------------|-----------|-----------|-----------|-----------|-----------|
| Sin plato | - | - | sin plato | 22.422 | 22.422 | 23.062 | 22.857 | 22.691 |
| Con plato | 240,05 | - | solo el plato | 58.172 | 58.169 | 57.792 | 57.969 | 58.026 |
| 5.000 | 5.271,05 | 5.031,00 | 5031 | 785.526 | 785.858 | 785.770 | 785.248 | 785.601 |
| 10.000 | 10.328,85 | 10.088,80 | 5057,8 + 5031,0 | 1.517.193 | 1.518.100 | 1.517.568 | 1.517.969 | 1.517.708 |
| 15.000 | 15.253,65 | 15.013,60 | 5,057,8 + 5031 + 4,924,8 | 2.231.668 | 2.231.591 | 2.231.142 | 2.231.791 | 2.231.548 |
| 20.000 | 20.392,45 | 20.152,40 | 15013,6 + 2067,2 + 2030,5 + 1041,1 | 2.975.166 | 2.975.139 | 2.974.791 | 2.974.958 | 2.975.014 |
| 25.000 | 25.528,95 | 25.288,90 | todos menos los 2 primeros | 3.862.174 | 3.862.379 | 3.862.514 | 3.862.336 | 3.862.351 |

CS5534 bajada

| Vdc = 4,8736V | | | | | |
|---------------|-----------|-----------|-----------|-----------|-----------|
| Peso | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
| 25.288,9 | 3.862.474 | 3.863.002 | 3.862.685 | 3.862.989 | 3.862.788 |
| 20.152,4 | 2.973.540 | 2.974.164 | 2.974.280 | 2.974.280 | 2.974.066 |
| 15.013,6 | 2.229.786 | 2.230.151 | 2.230.992 | 2.230.838 | 2.230.442 |
| 10.088,8 | 1.517.984 | 1.517.899 | 1.517.693 | 1.517.902 | 1.517.870 |
| 5.031,0 | 784.212 | 784.249 | 783.833 | 784.321 | 784.154 |
| - | 55.769 | 55.018 | 55.450 | 55.491 | 55.432 |

hemos dejado reposar sin peso 5 minutos mas para ver si varia el valor de pesada

Tabla 17: Prueba 3. CS5534 Unipolar. Resultados

HX711

HX711 subida

| Vdc = 4,938 V | | Peso real | Combinacion de pesas | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
|---------------|-----------|-----------|------------------------------------|------------|------------|------------|------------|---------------|
| Sin plato | - | - | sin plato | 8.407.453 | 8.407.499 | 8.407.482 | 8.407.403 | 8.407.459,25 |
| - | 240,05 | - | solo el plato | 8.442.590 | 8.442.514 | 8.442.598 | 8.442.450 | 8.442.538,00 |
| 5.000 | 5.271,05 | 5.031,00 | 5031 | 9.183.191 | 9.183.315 | 9.183.203 | 9.183.148 | 9.183.214,25 |
| 10.000 | 10.328,85 | 10.088,80 | 5057,8 + 5031,0 | 9.927.794 | 9.927.794 | 9.927.920 | 9.927.863 | 9.927.842,75 |
| 15.000 | 15.253,65 | 15.013,60 | 5,057,8 + 5031 + 4,924,8 | 10.652.629 | 10.652.634 | 10.652.562 | 10.652.702 | 10.652.631,75 |
| 20.000 | 20.392,45 | 20.152,40 | 15013,6 + 2067,2 + 2030,5 + 1041,1 | 11.408.920 | 11.408.841 | 11.408.949 | 11.408.915 | 11.408.906,25 |
| 25.000 | 25.528,95 | 25.288,90 | todos menos los 2 primeros | 12.311.070 | 12.310.929 | 12.311.018 | 12.311.100 | 12.311.029,25 |

HX711 bajada

| Vdc = 4,938 V | | | | | |
|---------------|------------|------------|------------|------------|---------------|
| Peso | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
| 25.000 | 12.311.102 | 12.311.156 | 12.311.138 | 12.311.172 | 12.311.142,00 |
| 20.000 | 11.408.975 | 11.409.077 | 11.409.065 | 11.409.059 | 11.409.044,00 |
| 15.000 | 10.652.856 | 10.652.936 | 10.652.884 | 10.652.960 | 10.652.909,00 |
| 10.000 | 9.928.240 | 9.928.277 | 9.928.112 | 9.928.102 | 9.928.182,75 |
| 5.000 | 9.183.639 | 9.183.656 | 9.183.539 | 9.183.678 | 9.183.628,00 |
| - | 8.442.923 | 8.442.895 | 8.442.844 | 8.442.861 | 8.442.880,75 |

Tabla 18: Prueba 3. HX711. Resultados

Prueba 4. Calibración de células de carga. Alimentación desde Arduino. Guarda Activa

Para este apartado se ha activado la “guarda” del convertidor AD. Básicamente, consiste en conectar la malla del cable de las células de carga al convertidor AD, de manera que evita interferencias y ruido. Se ha repetido la prueba 3 activando la guarda.

CS5534 subida

| Vdc = 4,9899V | | | | | | | | |
|---------------|-----------|-----------------|-----------------------------------|-----------|-----------|-----------|-----------|-----------|
| Peso teorico | | Peso real pesas | Combinacion de pesas | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
| Sin plato | - | - | sin plato | 18.688 | 18.711 | 18.875 | 18.917 | 18.798 |
| Con plato | 240,05 | - | solo el plato | 53.331 | 53.547 | 53.618 | 53.512 | 53.502 |
| 5.000 | 5.271,05 | 5.031,00 | 5031 | 781.532 | 781.672 | 781.624 | 781.767 | 781.649 |
| 10.000 | 10.328,85 | 10.088,80 | 5057,8 + 5031,0 | 1.513.584 | 1.513.663 | 1.513.683 | 1.513.743 | 1.513.668 |
| 15.000 | 15.253,65 | 15.013,60 | 5,057,8 + 5031 + 4,924,8 | 2.225.945 | 2.226.292 | 2.226.370 | 2.226.472 | 2.226.270 |
| 20.000 | 20.392,45 | 20.152,40 | 15013,6 + 2067,2 + 2030,5 +1041,1 | 2.969.806 | 2.969.770 | 2.969.833 | 2.970.110 | 2.969.880 |
| 25.000 | 25.528,95 | 25.288,90 | todos menos los 2 primeros | 3.857.202 | 3.857.285 | 3.857.327 | 3.857.368 | 3.857.296 |

CS5534 bajada

| Vdc = 4,9899 V | | | | | |
|----------------|-----------|-----------|-----------|-----------|-----------|
| Peso | Medida_1 | Medida_2 | Medida_3 | Medida_4 | MEDIA |
| 25.288,9 | 3.857.303 | 3.857.261 | 3.857.386 | 3.857.539 | 3.857.372 |
| 20.152,4 | 2.970.458 | 2.970.548 | 2.970.546 | 2.970.532 | 2.970.521 |
| 15.013,6 | 2.226.631 | 2.226.737 | 2.226.841 | 2.226.936 | 2.226.786 |
| 10.088,8 | 1.514.121 | 1.514.174 | 1.514.057 | 1.514.162 | 1.514.129 |
| 5.031,0 | 782.357 | 782.393 | 782.294 | 782.348 | 782.348 |
| - | 54.090 | 54.095 | 54.007 | 53.881 | 54.018 |
| - | 53.499 | 53.543 | 53.522 | 53.522 | 53.522 |

hemos dejado reposar sin peso 5 minutos mas para ver si varia el valor de pesada

Tabla 19: Prueba 4. CS5534 Unipolar. Resultados