Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Titulación: Grado en Ingeniería Electrónica Industrial y Automática

"Aplicación de la tecnología WEB en el desarrollo de un sistema SCADA"

Autora: Cristina Romera Belmonte

Tutor: Alberto Francisco Hamilton Castro

La Laguna, a 4 de Julio de 2018

Agradecimientos

La realización de este proyecto no hubiera sido posible sin la ayuda de mi tutor Alberto Francisco Hamilton Castro, quien me ha ayudado en todo momento cuando he solicitado su apoyo, aportando sus conocimientos y nuevas formas de afrontar los problemas. Así mismo quisiera agradecer a mi profesor de fundamentos de Ingeniería Electrónica, Alejandro José Ayala Alfonso, por su interés en prestarme su ayuda a la hora de realizar los circuitos necesarios.

También quisiera dar las gracias a Endesa, empresa donde realicé las prácticas, porque fue donde pude tener un primer acercamiento con los procesos industriales y fue el principal motivo que me llevó a escoger este proyecto, además quisiera tener en cuenta al Técnico Jefe de Mantenimiento de la empresa Dipicell, Antonio Castro, por su contribución y aporte de ideas para mi proyecto.

Por último, agradecer a mi familia y mis amigos que me han apoyado en todo momento, sobre todo a mis padres y mi hermana que me han aportado sus mejores consejos y que han estado siempre que los he necesitado, dándome las fuerzas necesarias.

Gracias a todos.

Resumen

En la actualidad se está viviendo un proceso de transformación donde todas las tendencias llevan hacia la digitalización, los entornos web están mucho más desarrollados y son más atractivos para el usuario que hace unos años atrás, las páginas web permiten tener el mundo en tus manos, en el momento y lugar que se desee, por eso este trabajo ha llevado esta nueva manera de llegar al usuario al área industrial, se ha trabajado con todas estas tecnologías para conseguir crear un sistema SCADA con el formato de una página web.

La creación de este sistema no sería factible si no permitiera una interacción con el mundo físico y real, es decir, un cambio de estado en los componentes que se desean controlar, para ello se ha seleccionado la placa de desarrollo BeagleBone Black la cual actúa como servidor de la página web, y es quien se comunicará con aquellos elementos sobre los que se quiere cambiar su estado.

Abstract

Nowadays, a process of transformation is taking place and all tendencies lead towards digitalization. Web environments are far more developed and are more attractive for the user than a few years ago. Furthermore, web pages allow you to have the world in your hands at the desired time and place; for this reason, this work has taken this new way to reach the user to the industrial area. All these technologies have been employed to create a SCADA system with the format of a web page.

The creation of this system would not be feasible if it did not allow an interaction with the physical and real world, that is, a change of state in the components that we wish to control. For this, we have selected the BeagleBone Black development board, which acts as a server for the web page, and which will communicate with those elements whose status is desired to be changed.

Índice

Capítulo 0: Tablas

0.1 Índice de tablas

0.2 Índice de figuras

0.3 Nomenclatura y abreviaturas

- ADC: "Analogic Digital Convertor", Convertidor analógico digital.
- AO: Amplificador Operacional.
- BBB: BeagleBone Black.
- BCD: "Binary-Coded Decimal", Decimal Codificado en Binario.
- CSS: "Cascading Style Sheets", Hojas de estilo en cascada.
- DCS: "Distributed Control System", Sistema de Control Distribuido.
- DOM: "Document Object Model", Modelo de Objetos del Documento.
- GIF: "Graphics Interchange Format", Formato de Intercambio Gráfico.
- HMI: "Human Machine-Interface", Interfaz Humano Máquina.
- HTML: "HyperText Markup Language".
- HTTP: "Hypertext Transfer Protocol".
- IDE: "Integrated Development Environment", Entorno de Desarrollo Integrado.
- IoT: "Internet of Things", Internet de las cosas.
- JPG-JPEG: "Joint Photographic Experts Group", Grupo de Expertos Fotográficos Unidos.

- LAN: "Local Area Network", Red de Área Local.
- MAP: "Manufacturing Automation Protocol", Protocolo de Fabricación Automatizada.
- MTU: "Maximum Transmission Unit", Unidad Terminal Maestra.
- PLC: "Programmable Logic Controller", Controlador Lógico Programable.
- PNG: "Portable Network Graphic", Gráfico portable para la red.
- PV: "Process value", Valor de proceso.
- PWM: "Pulse Width Modulation", Modulación por Ancho de Pulso.
- RTU: "Remote Terminal Unit", Unidad Terminal Remota.
- SCADA: "Supervisory Control And Data Acquisition", Supervisión, Control y Adquisición de Datos.
- SISO: "Single Input Single Output", una sola salida y entrada.
- SO: Sistema operativo.
- SP: "Setpoint", Valor de consigna.
- SVG: "Scalable Vector Graphics", Gráfico Vectorial Escalable.
- W3C: "World Wide Web Consortium", El Consorcio WWW.
- WAN: "Wide Area Network", Red de Área Amplia.

Capítulo 1: Introducción

1.1 Motivación y Antecedentes

Este Trabajo de Fin de Grado nace con la necesidad investigar acerca de las tecnologías en el desarrollo web y trasladar su uso al ámbito industrial, para ello se va a explorar este campo creando un sistema de visualización y control a determinados sistemas, se comenzará con un sistema algo más sencillo y se incrementará el grado de dificultad hasta la realización de un sistema de monitorización y visualización para la maqueta de un proceso industrial, caracterizado por un lazo cerrado.

Como bien se sabe, en el ámbito industrial es de vital importancia la monitorización de todos los cambios que puedan producirse durante el transcurso del proceso, además de poder operar sobre los mismos, para ello es necesaria una interfaz humano-máquina que permita al usuario interaccionar sobre dicho procedimiento.

Este proyecto aborda dicha propuesta, la realización y desarrollo de una interfaz que permite a un operador interactuar con una planta industrial y que pueda ser controlada por el mismo, en concreto se habla de la materialización de un sistema de supervisión, control y adquisición de datos, es decir, un SCADA.

La principal y novedosa característica de este proyecto es la realización de un software con los lenguajes fundamentales de la programación web, por tanto, se habla de los siguientes lenguajes:

- HTML5, la última versión del lenguaje de marcas HTML.
- JavaScript, permitirá que la página web sea dinámica e interactiva
- CSS3, actualización de CSS que ofrece la posibilidad de añadir estilo a una página web.

Se estudiará la viabilidad de dichas tecnologías para la implantación de sistemas SCADA.

A lo largo de este proyecto se explicarán con más detalles estos lenguajes y las diversas herramientas utilizadas.

Para situar al lector y que comprenda las características del proyecto, se debe de aclarar que éste se diseña desde cero, y que por tanto no cuenta con ningún antecedente que se deba destacar.

Sin embargo, si hay que comentar en qué estado se encuentra el mercado en cuanto a este producto.

Actualmente existen empresas que han desarrollado software para la implementación de sistemas SCADA, ejemplo de ello sería SIMATIC WinCC de la empresa Siemens, un software HMI flexible y con un gran potencial en el ámbito industrial.

En definitiva, es un sistema de visualización de procesos escalable y dotado de potentes funciones para la supervisión de procesos automatizados. Si se trabaja con el sistema operativo Microsoft Windows se obtiene el máximo rendimiento de WinCC y aumenta con la combinación de controladores fabricados por la propia empresa. [1]

Al igual que Siemens existen muchas otras empresas como Logitek o ABB, que desarrollan sistemas SCADA acorde a sus máquinas industriales, como motores o turbinas que existen en el mercado, para procesos de fabricación de gran envergadura estas soluciones son bastante eficientes, aunque también son costosas y privadas pues es necesario tener el producto de la propia marca para poder incorporar su propio sistema de visualización.

A continuación, se muestra la tabla 1 con los principales softwares SCADA y sus fabricantes.

Tabla 1: Software SCADA

Software	Fabricante
Aimax	Desin Instruments S.A.
CUBE	Orsi España S.A.
FIX	Intellution
Lookout,	National Instruments.
Monitor Pro	Schneider Electric.
SCADA InTouch	Logitek
SYSMAC SCS,	Omron
Scatt Graph 5000	Scatt Graph 5000
WinCC	Siemens

Este proyecto quiere dar cabida y experimentar con las nuevas tecnologías web, altamente competentes y sofisticadas que cualquier ingeniero con una base de conocimientos informáticos puede desarrollar y personalizar acorde a las exigencias de su proyecto, con un coste mucho menos elevado que lo que proponen las grandes compañías, incluso llegando al coste cero.

Además, el SCADA debe de tener una arquitectura abierta, es decir, que pueda adaptare a las nuevas especificaciones y cambios que surjan en la empresa, esto es posible con las tecnologías aplicadas en este trabajo, asimismo al ser una tecnología en pleno desarrollo se cuenta con un amplio contenido de información en la red.

Hoy en día se puede decir que se tiene el mundo en un solo *click*, pues internet está al alcance de cualquiera, por eso en este proyecto se saca partido a este hecho. Con este proyecto se puede demostrar que las tendencias en el mundo industrial, en cuanto a los sistemas SCADA, deberían de ir hacia una completa conexión a internet lo que permite que no necesariamente hay que estar cercano al proceso industrial, incluso ni en la misma zona, pues las páginas web están presentes en cualquier momento y a cualquier distancia, por eso se hace hincapié en este concepto, porque ayudan a ampliar la disponibilidad de la planta.

1.2 Objetivos

Este trabajo tiene como objetivo final dar cabida a las tecnologías utilizadas en el desarrollo web dentro del mundo industrial, a día de hoy existen páginas web con usos totalmente diferentes, de carácter informativo, algunas divulgan conocimientos sobre cualquier área, las hay dedicadas al ocio, o interacción entre las personas, entre otras. Por ello se ha pensado que entre toda la variedad existente se puede hacer un hueco a las páginas web de uso industrial.

Debido a que el fin de este proyecto no es el control de una planta, sino comprobar que se pueden utilizar páginas web como sistemas SCADA. Se comenzará con maquetas más sencillas para el diseño de los mismos, donde se harán pruebas sobre circuitos básicos compuestos por leds, un potenciómetro y un botón. Conforme las pruebas vayan dando resultados satisfactorios se incrementará la dificultad gradualmente hasta llegar a crear un SCADA más complejo y dedicado a un ejemplo de planta industrial.

Para concretar acerca de la maqueta sobre la que se incorporará y probará la utilidad del diseño del SCADA, se ha seleccionado un módulo educativo, "38-001 SYSTEM Level & Flow control", que permite el aprendizaje sobre diferentes sensores y la aplicación de varios tipos de control.

Es por ello que se desarrollará un sistema SCADA para esta planta industrial, donde se utilizará un microordenador, BeagleBone Black, como hardware. Para mejorar el trabajo y darle una funcionalidad al mismo se implementará un control On-Off sobre la bomba de corriente continua que forma parte de este panel educativo, y se realizará así mismo un control PID que actuará sobre la servo válvula.

Se necesitará para llevar a cabo este último tipo de control, un circuito electrónico que acondicione la señal de salida del microordenador para su correcta utilización como señal de control.

1.3 Estructura de la memoria

Este apartado pretende explicar la estructura de este trabajo y la finalidad de cada capítulo y subapartados.

El capítulo cero alberga la información sobre tablas y las figuras incluidas a lo largo del trabajo, además se incorpora un apartado para nomenclatura y abreviaturas utilizadas a lo largo del trabajo con el propósito de conseguir una fácil lectura del mismo.

El capítulo uno pone énfasis a los motivos que han originado el desarrollo de este trabajo y los cuáles son los objetivos que se desean cumplir con el mismo.

Seguidamente el capítulo dos contiene toda la información que ha sido necesaria para el desarrollo de este proyecto y que aporta al lector unos conocimientos previos.

En el capítulo tres se explica con detalle la solución adoptada por la autora para la materialización del sistema SCADA, esta sección es la más extensa y explicativa sobre el trabajo en sí mismo.

El capítulo cuatro contiene las posibles líneas de mejora del trabajo y las conclusiones del mismo.

El capítulo cinco contiene el presupuesto, concretando el precio de los materiales utilizados, con el valor añadido de su ejecución.

El capítulo seis incluye la información extra de los dispositivos, Datasheets, y la bibliografía.

1.4 Metodología y programación del proyecto

La metodología seguida a lo largo de este trabajo ha sido la que lógicamente se debe de aplicar cuando se desconoce el entorno sobre el que se va a trabajar.

Primero se comenzó con un estudio de la placa de desarrollo adquiriendo información de la misma y sus principales características, para poder sacar provecho de ella posteriormente en la ejecución del trabajo. Seguidamente se tuvo un primer contacto con los lenguajes utilizados para la creación de páginas web. Una vez obtenido una base en la que poder fundamentar el trabajo, se comenzó con pequeñas pruebas primero localmente en la placa de desarrollo y después de manera remota, hasta llegar a conseguir un SCADA que se pueda incorporar a un proceso industrial.

A continuación, se detalla la planificación del proyecto.

Planificación del Proyecto

	Enero				Febrero				Marzo				Abril				Mayo				Junio				Julio			
Actividades	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª	1ª	2ª	3ª	4ª
Adquisición de Documentación acerca de BeagleBone																												
Recopilación de información de los principales lenguajes para el desarrollo web																												
Primeras pruebas en el desarrollo web																												
Pruebas en el desarrollo web																												
Pruebas con el dispositivo BeagleBone																												
Realización de un SCADA a pequeña escala																												
Realización del SCADA para la planta																												
Aplicación del control on-off sobre la bomba																												
Mejora del SCADA planteado																												
Diseño del circuito de salida para el acondicionamiento de la señal																												
Prueba del circuito acondicionador																												
Redacción del proyecto																												
Diseño de sistema SCADA para una maqueta de planta industrial																												

Planificación del proyecto

Capítulo 2: Marco teórico

Este capítulo concentra toda la información que ha sido necesaria para la elaboración de este proyecto, y que aporta al lector los conceptos previos para el correcto entendimiento y lectura del trabajo.

2.1 La automatización

2.1.1 Introducción a la automatización.

Para llegar a entender el desarrollo de la industria hay que mirar unos cuantos siglos atrás para darnos cuenta de lo mucho que se ha evolucionado y cómo todavía hoy en día, se sigue haciendo.

Cabe destacar que La Revolución Industrial, también conocida como Primera Revolución Industrial supuso el inicio de la industria, recordada como un proceso de transformación en el ámbito económico, social y tecnológico que comenzó a mediados del siglo XVIII en Gran Bretaña y que se terminaría extendiendo al resto de Europa. Con anterioridad a ella, todos aquellos productos que eran manufacturados se elaboraban en pequeños talleres artesanos, y debido a una serie de descubrimientos tecnológicos se propició el proceso de industrialización.

Los talleres artesanales fueron sustituidos por fábricas, grandes establecimientos que contrataban a gran número de obreros para trabajar junto con máquinas. El uso de nuevos materiales como el acero o el carbón significó un avance en el campo de las fuentes de energía, sobre todo en las máquinas motrices existentes de la época, ejemplo de ello: la máquina de vapor.

La idea de división del trabajo, incorporada en La Segunda Revolución Industrial (mediados del siglo XIX), en los procesos de fabricación posibilitó un incremento en la producción que trajo consigo la reducción del nivel de especialización de los obreros.

El concepto de mecanización nacido durante el proceso de Revolución Industrial implicó el uso de maquinaria en aquellas tareas que requerían trabajos físicos, aportando de esta manera ayuda al operario, en ocasiones las máquinas remplazaron de forma parcial o incluso en su totalidad el trabajo del ser humano o animales.

La mecanización fue el paso previo a la automatización ya que con el paso de los años surge la necesidad de ahorrar energía humana tanto en las actividades que generan un beneficio económico como en las cotidianas.

Conforme la tecnología de transferencia de energía evolucionaba, las máquinas especializadas se monitorizaron aumentando así su eficacia productiva. El desarrollo de la tecnología energética también dio lugar al surgimiento del sistema industrial de producción, ya que todos los trabajadores y máquinas debían estar situados junto a la fuente de energía, proceso que se conoce como Tercera Revolución Industrial.

Todos estos acontecimientos y el desarrollo de las nuevas tecnologías han propiciado el aún crecimiento de la industria, ya que aprovechando todos estos recursos se puede llegar a la eficiencia en cualquier cadena de fabricación, un ejemplo de ello es el control y monitorización de los procesos que se desempeñan de manera cíclica y secuencialmente, y que por tanto es esencial llevar un chequeo de cada una de las tareas desempeñadas.

Por último, se habla de la Cuarta Revolución Industrial y que origina término de "Industria 4.0" que proviene de un proyecto alemán que incorpora la alta tecnología en el sector industrial.

El principal propósito de este movimiento es crear una fábrica que por sí sola sepa actuar ante cualquier cambio, es decir, que sea inteligente, para ello las maquinas deben estar automatizadas y conectadas entre sí a través de una red, en este caso, Internet, lo que trae consigo el concepto de Internet de las cosas (abreviado como IoT) que se basa precisamente en lo comentado anteriormente. A continuación, la figura 1 muestra la evolución de la industria.

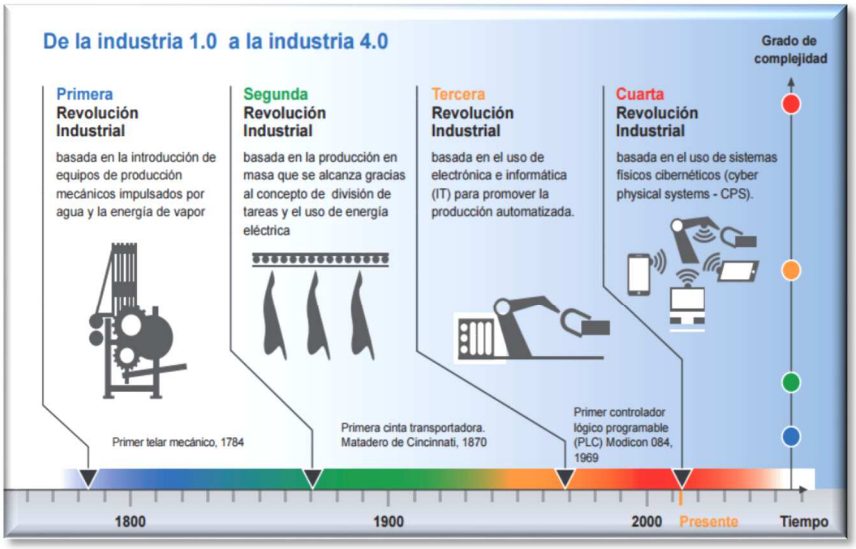


Figura 1: Evolución de la industria [2]

2.1.2 Sistemas de control distribuido y centralizado.

La implementación de un sistema de control de los equipos es muy importante para poder garantizar la seguridad y la eficiencia de cualquier planta industrial. Gracias a dicha implantación, se controlan todas las variables que intervienen en el proceso y que podrían afectar a este y, por lo tanto, se asegura la calidad, la viabilidad y la seguridad.

Una de las primeras cuestiones que hay que resolver para incluir un sistema de control, es la elección entre un sistema de control distribuido o centralizado, esta decisión marcará la arquitectura y organización de los elementos que la componen.

A continuación, se van a comentar las características, las diferencias y ventajas de cada uno.

Se comienza con la explicación de la arquitectura de control centralizada que se basa en la concentración de información y de procesado de señal en un único controlador centralizado. Asimismo, opta por la utilización de redes analógicas (4-20mA) tanto para la conexión de sensores dedicados a la captación de señales de entrada como para la activación de indicadores.

Esta arquitectura fue bastante útil y efectiva durante años, pero se caracterizaba por la gran cantidad de cable utilizado para las conexiones, por lo que si se necesitaba incorporar un nuevo dispositivo, ya fuera de captación de información o de actuación, implicaba tener que ampliar la instalación, en algunos caso era posible pues aún había capacidad pero en otros se daba la situación de no tener disponibilidad para esta nueva incorporación, por lo que se intentó optar por otro tipo de arquitectura.

Sin embargo, sí que es útil para ciertas aplicaciones, por ejemplo, para procesos que sean simples o que sólo requieran de una operación, se puede optar entonces por esta solución, este proyecto es un ejemplo de ello ya que sólo se dispone de una sola variable controlada y una variable manipulada.

Por otro lado, un sistema de control distribuido se fundamenta en la comunicación de los dispositivos que se encuentran en la planta, mediante una red de comunicaciones que se complementa con múltiples ordenadores, cada uno encargado de un cierto número equipos y de su lazo de control, estos nodos son físicos y están dotados de una capacidad de proceso y además están enlazados con sensores y actuadores. Su principal propósito es el de compartir información entre ellos y con los ordenadores de operación.

La principal característica y ventaja, es la descentralización de la supervisión de los procesos, lo que permite que ante cualquier fallo en uno de los lazos de control el resto de lazos puedan seguir efectuando sus tareas, debido a que los nodos se coordinan entre ellos, este hecho permite la reducción del cableado entre dispositivos.

Asimismo, debido a la importancia de ciertos dispositivos, por ejemplo, aquellos encargados de la adquisición de señales y las unidades de procesamiento, se aplica la técnica de redundancia de los mismos, por lo que se requiere de un repuesto para el caso en el que el principal o primario falle, éste segundo ejerce de principal. [3]

La figura 2 muestra una arquitectura típica de un sistema de control distribuido.

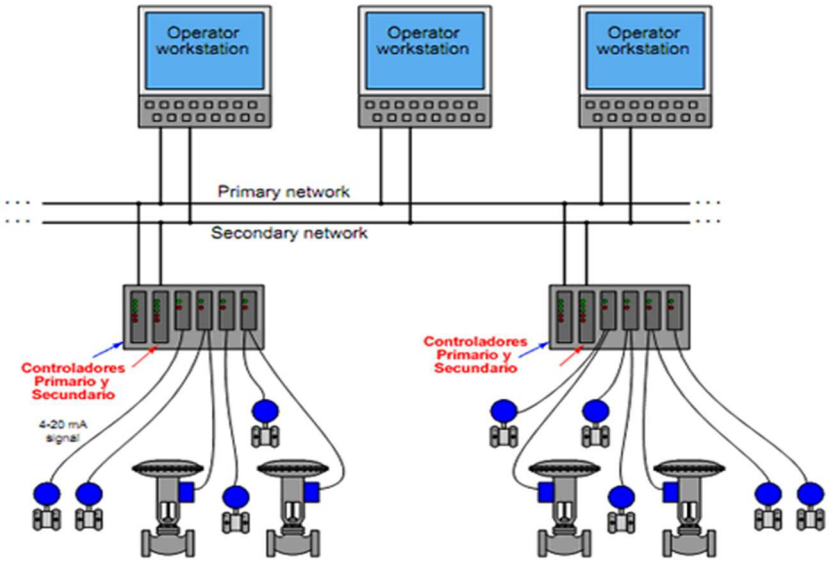


Figura 1: Arquitectura de un DCS[4]

2.1.3 Niveles de automatización.

Dada las exigencias de los procesos industriales, éstos se han estructurado buscando su efectividad, para ello se dividen en distintos niveles, conformando la pirámide de la automatización. Estos niveles son:

- **Nivel de proceso:** es el nivel inferior donde se encuentran los dispositivos de campo como actuadores y sensores, están en contacto e interactúan con el proceso industrial.

- **Nivel de control:** en este nivel están los distintos dispositivos de control tales como por ejemplo los PLC. Este nivel es el encargado de controlar todos los dispositivos del campo.

- **Nivel de supervisión y visualización:** es el encargado de la monitorización de los procesos y de controlar la interacción entre los distintos dispositivos ubicados en el nivel de control. A este nivel son imprescindibles los sistemas SCADA o HMI.

- **Nivel de información y manufactura:** este nivel proporciona la información necesaria para optimizar el sistema de producción desde el lanzamiento de la orden de fabricación hasta el producto acabado.

- **Nivel de administración:** es el nivel que integra la tecnología para organizar la producción y todos los procesos relacionados que se llevan a cabo en toda la empresa.

Como es de esperar entre los diferentes niveles de la pirámide existe un flujo de información, que se rige bajo los protocolos establecidos de comunicación. En función del tipo de nivel que se trate y de acuerdo con la cantidad de datos a enviar y la velocidad con la que se envían se selecciona un protocolo que agrupe las exigencias.

Algunos de los protocolos más utilizados en la industria son: Profibus DP, Devicenet, Modbus, Can Open, AS-i, Ethernet/IP, Modbus, entre otros. Con esta organización estructural se pueden controlar y monitorear desde pequeños automatismos hasta procesos de grandes envergaduras, consiguiendo reducir los tiempos de mantenimiento y optimizar los niveles de productividad.

La figura 3 muestra la pirámide de la automatización con cada uno de los niveles explicados.

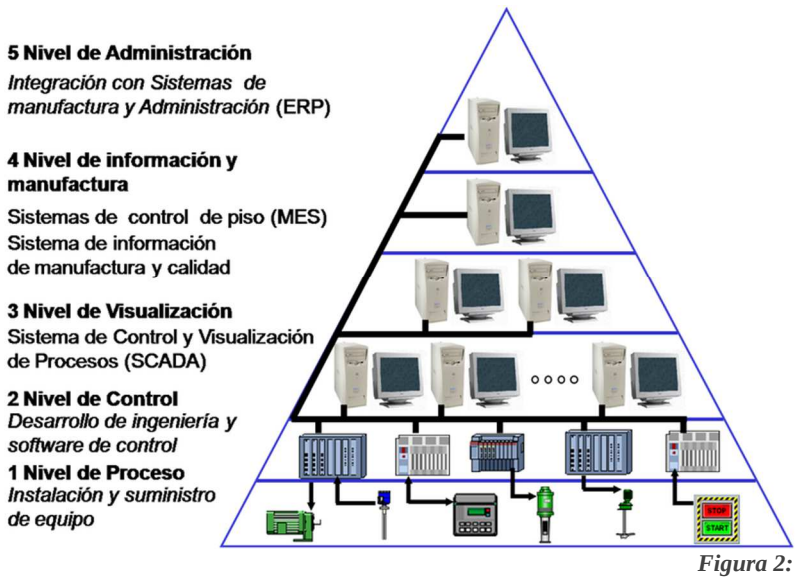


Figura 2: Niveles de automatización [5]

2.1.4 Redes de comunicación industrial.

La mejora de los procesos automatizados se produjo con el desarrollo de las redes de comunicación, numerosos sistemas están compuestos por equipos de diferentes fabricantes y trabajan en diferentes niveles de automatización. Fundamentalmente, se desea que, a pesar de estar distanciados, en ocasiones varios kilómetros, trabajen de forma coordinada.

En la actualidad, las comunicaciones industriales conforman un pilar esencial, para el proceso en sí y para su automatización, y por consiguiente para el sistema SCADA que se encuentre implementado.

Para que la información enviada por el terminal emisor llegue de manera correcta y segura a su receptor, las instrucciones deben de adecuarse al estándar utilizado en el medio de comunicación.

Para responder a las necesidades de la intercomunicación en tiempo real, la comunicación debe de cumplir algunos requisitos en particular. Asimismo, debe de ajustarse a las adversidades que pueda presentar la zona dónde está implantada, por ejemplo, las altas temperaturas, por encontrase cerca de una caldera de combustión, el ruido electromagnético, si existen cables u otro equipo que emita señal.

Según la utilizad y hacia dónde se emita la información, se puede distinguir entre una comunicación a nivel de campo, donde se localizan los actuadores, sensores y controladores del sistema, o una comunicación hacia el sistema SCADA.

Ahora bien, según el marco de los niveles de automatización industrial, donde se van a establecer, se diferencian varios tipos de redes,

- **Red de factoría:** esta red es la encargada de la contabilización y administración de las ventas que se llevan a cabo en la empresa y de la gestión de los pedidos, productos y almacén. En cuanto al volumen de información que se transporta, cabe decir que es bastante elevado pero el tiempo de transporte no resulta crítico. Se emplea una red de tipo LAN, "*Local Area Network*", o WAN, "*Wide Area Network*".[6][13]

- **Red de planta:** esta red permite la interconexión de los módulos y las células de fabricación entre sí y con los departamentos. Se obtiene un sistema óptimo sistematizado que administra de manera secuencial todas las operaciones. Está constituido por un servidor que dirige cada departamento de la empresa, enviando y recibiendo archivos y en este caso enlazándolos para que al trabajar conjuntamente se obtenga resultados muy productivos. El tráfico de datos es muy variable, desde mensajes cortos de órdenes de ejecución hasta mensajes interactivos de terminales de operarios. Se suele emplear una red de tipo LAN. [6][13]

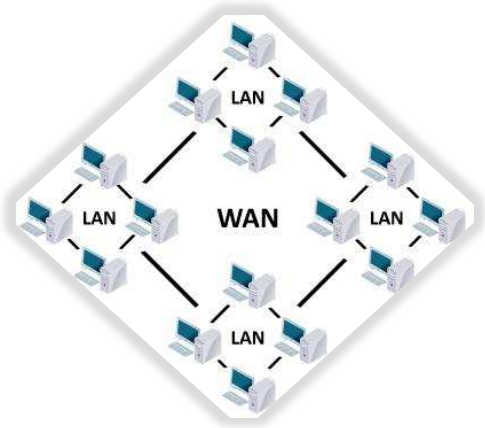


Figura 3: Ejemplo de red WAN y LAN.[7]

- **Red de célula:** este nivel se dedica a la integración de los dispositivos como los PLC's, máquinas de control numérico, robots, es decir, pequeños automatismos, éstos se pueden implantar en sub-redes. Los requisitos que deben cumplir son la capacidad de gestión de mensajes breves, el manejo de tráfico de eventos discretos, disponer de mecanismos de control de error, la posibilidad de enviar mensajes con prioridad, como por ejemplo,

alarmas, su coste de instalación y mantenimiento debe ser bajo, restablecimiento del mensaje ante situaciones de anomalía y por supuesto una alta fiabilidad. En este nivel se emplean las redes MAP ("*Manufacturing Automation Protocol"*), del mismo modo se puede instalar redes superiores como Ethernet Industrial, siempre que cumpla con las premisas y asegurando el establecimiento en la red. [[6][13]

- **Bus de campo:** este sería el nivel más próximo al proceso, por lo que es dónde se encuentran los sensores y actuadores, en este nivel se pretende sustituir los sistemas de cableado tradicional por buses de campo con prestaciones que superan a los tradicionales, como su fiabilidad y robustez en las transmisiones a tiempo real y serial, su bajo coste, la capacidad de interconectar controladores con todo el espectro de dispositivos de entrada-salida. Éstos incluyen los sensores, transmisores, interruptores y actuadores sencillos, además de los controladores esclavos inteligentes. Deben de tener un tiempo de respuesta breve ante el transporte de mensajes de tamaño pequeño y con frecuencia de tráfico periódica, con una alta fiabilidad. La implantación de buses de campo mejora la calidad y la cantidad en el flujo de datos, además de ahorrar material en la instalación(ver figura 5), y con vistas de futuro, facilita la ampliación, o si fuera necesario la reducción del número de elementos del sistema, y lo más importante reduce errores en la instalación. [6][13]

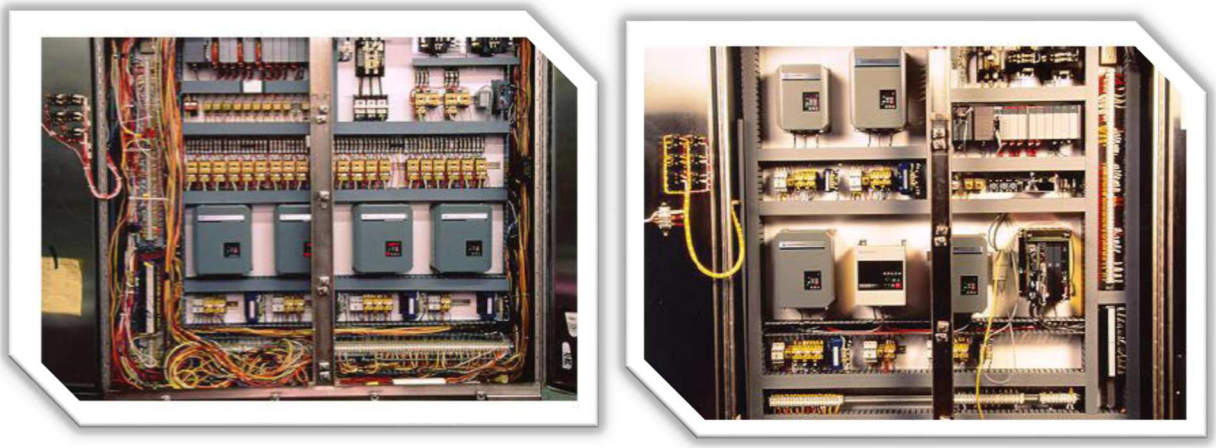


Figura 4: Instalación industrial sin bus de campo y con bus de campo [8]

La siguiente tabla 2 cuenta con los principales buses industriales del mercado y sus características.

Tabla 2: Características de los buses campo [6]

Bus de campo	Características
Interbus.	Comunicación maestro-esclavo, emplea la topología anillo.
Profibus FieldBus	• DP: utilizado con sensores o actuadores enlazados a procesadores. • PA: cumple con normas especiales de seguridad en ambientes peligrosos y con riesgo de explosión. • FMS: permite la comunicación entre células de proceso o equipos de automatización.
CAN.	Es un bus multimaestro en el que se puede conectar dispositivos inteligentes de todo tipo y está estandarizado como ISO 11898-1.
CANOpen	Es utilizado por fabricantes de maquinaria e integradores de célula de proceso.
DeviceNet.	Es posible la conexión de hasta 64 nodos con velocidades de 125 Kbps a 500 Kbps en distancias de 100 a 500 m, maestro-esclavo, interrogación cíclica, strobing o lanzamiento de interrogación general de dispositivos. Adecuada para conectar dispositivos simples, por tratarse de una red de bajo nivel.
Fieldbus.	Para la interconexión de dispositivos en industrias de proceso continuo
Modbus.	Puede implementarse sobre cualquier línea de comunicación serie y permite la comunicación por medio de tramas binarias o ASCII con un proceso interrogación-respuesta simple. En concreto es un protocolo de transmisión para sistemas de control y supervisión de procesos (SCADA)
Industrial Ethernet.	Método más extendido para interconexión de computadores personales en redes de proceso de datos.
Actuator Sensor Interface, ASI.	Empleado para la interconexión de actuadores y sensores binarios puede adoptar cualquier tipo de topología: estructura en bus, en árbol, en estrella o en anillo. Permite la interconexión de un máximo de 31 esclavos
Hart	Su campo de aplicación básico es la comunicación digital sobre las líneas analógicas clásicas de los sistemas de instrumentación

Para concluir con este apartado y con la explicación de los sistemas de comunicación, se requiere añadir la figura 6 que ejemplifica el flujo de información tanto de manera vertical, de un nivel a otro, como horizontalmente, entre los dispositivos del mismo nivel.

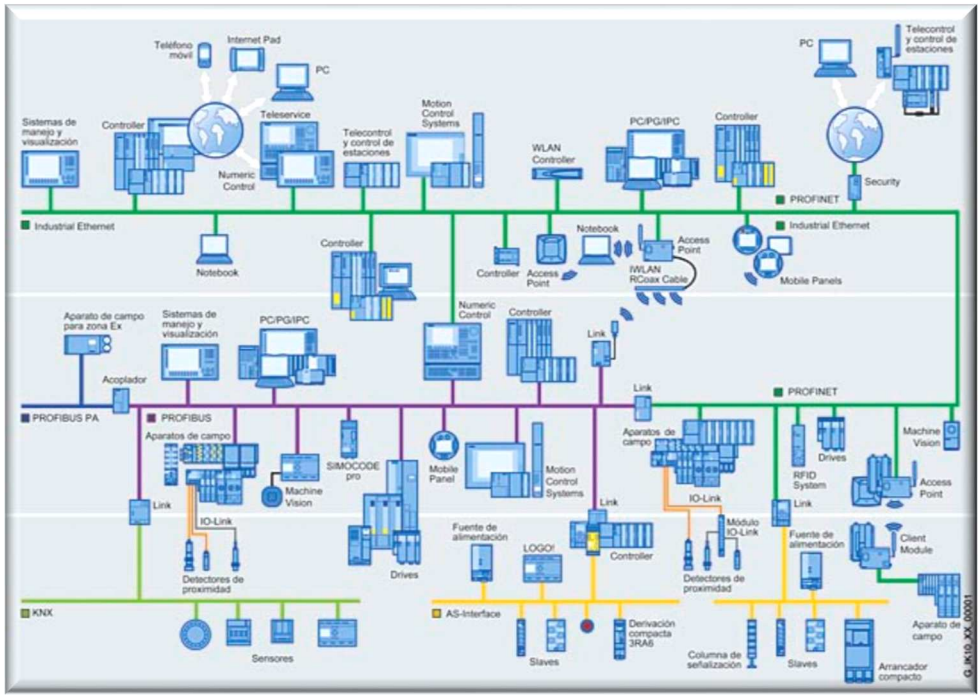


Figura 5: Esquema de la comunicación industrial [9]

2.2 Sistema SCADA.

Debido a la importancia del SCADA para este proyecto se ha decidido dedicar un apartado al mismo.

2.2.1 Interfaz Humano-Máquina.

Una interfaz de usuario asistida por ordenador, actualmente una interfaz de uso, también conocida como interfaz hombre-máquina (HMI), forma parte del programa informático que se comunica con el usuario. En ISO 9241-110, el término interfaz de usuario se define como "*todas las partes de un sistema interactivo (software o hardware) que proporcionan la información y el control necesarios para que el usuario lleve a cabo una tarea con el sistema interactivo*". [14]

Una Interfaz Humano-Máquina favorece a la compresión de los procesos y ofrece información sobre el estado en el que se encuentra, se define como un panel de instrumentos y es una herramienta clave tanto para operarios como supervisores de línea que sirve para coordinar y controlar procesos industriales y de fabricación. [15]

Es casi obligatorio conocer el valor de las variables de cualquier proceso, el HMI simplifica esta laboriosa tarea, de forma que muestra al usuario lo que necesita conocer de manera rápida, se debe de hacer hincapié en esto, ya que demasiada información puede saturar y provocar errores en la actuación y de igual forma la escasez de la misma.

Para que la información aportada sea de utilidad al usuario tiene que ser proporcionada en tiempo real. Es bastante práctico generar gráficos de procesos visuales que aportan significado y contexto al estado de cualquier parámetro del proceso, en el caso particular de este proyecto será vital para el nivel de agua del depósito, por tanto, debe quedar claro que para una optimización acertada del proceso es necesaria una información apropiada.

2.2.2 ¿Qué es un sistema SCADA?

Previamente a la existencia de la tecnología digital lo sistemas de interfaz entre el usuario y la planta se fundamentaban en grandes paneles físicos de control que incluían numerosos indicadores luminosos, instrumentos de medida y pulsadores, que además de ser poco práctico generaba equivocaciones y dificultaba la lectura de la información relevante.

Con la digitalización de los sistemas, se trasladaron estos paneles de control a la pantalla de los ordenadores añadiendo nuevas funciones, nace por tanto el concepto sistema SCADA.

SCADA es el acrónimo de *Supervisory Control And Data Acquisition*, que en español sería Supervisión, Control y Adquisición de Datos. Realmente se trata de un software que permite controlar y supervisar cualquier proceso industrial con la ventaja de que puede existir cierta distancia entre el proceso y el ordenador donde se encuentra el programa instalado, admitiendo además un control a tiempo real gracias a la adquisición de información de los sensores y transmisores, que permiten la retroalimentación de información, y cuyo resultado es la intervención sobre los actuadores para alcanzar la consigna deseada.

Asimismo, un SCADA permite la conectividad y transformación de información con otros ordenadores situados en la planta.

Es necesario señalar que el control se efectúa a través de los controladores autónomos digitales y/o autómatas programables (PLC) y que mediante su conexión a un ordenador y el SCADA permite la manipulación de información y el control de la producción.

La parte visual del SCADA es bastante importante ya que ofrece la posibilidad a un operador, el cual no debe por qué tener conocimientos teóricos sobre el proceso, actúe y controle dicho proceso. De igual forma es imprescindible que cuente con un sistema de seguridad, el cual supervise qué operación tiene permitido realizar cada operario o supervisor de la planta.

2.2.3 Funciones de un sistema SCADA.

A continuación, se exponen las principales funciones básicas que realiza un sistema SCADA.

Una de las principales prestaciones es la adquisición de datos, es bastante importante porque en base al valor recogido se tomarán las diferentes decisiones, para ello se debe de recoger, procesar y guardar la información recibida. Recordemos que en este apartado los sensores o transductores tienen un papel crucial, ya que traducen las variaciones del fenómeno físico que se desea registrar en variaciones proporcionales de una variable con magnitud eléctrica (voltaje, corriente, resistencia, capacidad, inductancia, etc.). [10]

Después de procesar la información se debe de chequear que ésta es correcta, es decir, realizar la supervisión de la evolución de las variables de control, esta funcionalidad permite valorar el estado de los sensores, ya que con los datos recogidos se puede observar si están descalibrados o defectuosos, asimismo permite una estimación de la situación de la planta. En definitiva, ayuda a encaminar las tareas de mantenimiento y estadística de fallas. [10]

Por último, pero no por ello menos importante, la función de control. Si se requiere modificar la evolución del proceso, se pueden llevar a cabo actuando en los reguladores autónomos básicos como puede ser el valor de consigna, alarma, parámetros, algoritmos de control, o de forma directa en el proceso mediante las salidas conectadas, por ejemplo, abriendo o cerrando válvulas, encendiendo motores o desactivando interruptores, estos cambios se pueden realizar de manera manual o automáticamente. [10]

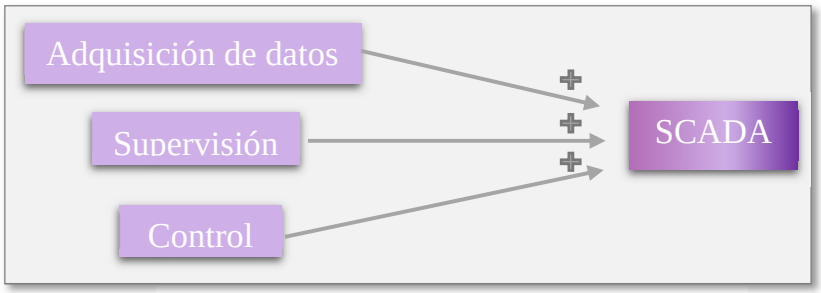


Figura 6: Funciones básicas del sistema SCADA

Además, existen otras funciones más específicas, como puede ser la transmisión de información o comunicación con los equipos situados en campo y otros ordenadores.

Toda la información recogida es de gran interés para investigaciones futuras de la planta, para ello se debe de gestionar los datos con bajo tiempo de acceso, y crear bases de datos. De este modo se considera la opción de generar reportes con los datos estadísticos del proceso, en un tiempo determinado por el operador e informes históricos.

Es primordial la forma en la que se presentan los datos y la representación gráfica de los mismos, se debe de contar con una HMI eficiente y potente. Esta interfaz mejora si se incorpora una visualización dinámica y a tiempo real que representen con imágenes el comportamiento del proceso, aunque parezca de poca utilidad ayuda al usuario. Se debe tener también en cuenta la representación de señales de alarma ya sea de forma visual, con mensajes de alerta o cambios de color en el SCADA, o sonora.

No es sólo importante la adquisición de los datos sino también la explotación de los mismos, para la gestión de calidad, control estadístico, gestión de la producción y gestión administrativa y financiera.

2.2.4 Componentes de un SCADA.

Los principales componentes de un sistema SCADA son los siguientes:

- **La interfaz HMI**: previamente comentado, es un espacio visual que ofrece el software para que el usuario interactúe con la planta.
- **Unidad Terminal Maestra (MTU) o Estación Maestra:** es el elemento central de control del sistema de adquisición de datos y control, este término se refiere al servidor y al software encargado de la comunicación con los elementos de campo, pudiendo estos ser RTU o PLC. En la unidad maestra se halla el software HMI corriendo para las estaciones de trabajo.[11]
- **Unidad Terminal Remota (RTU):** son dispositivos de adquisición de datos y control situados remotamente en campo, cuya misión principal es hacer de interfaz entre los equipos de instrumentación y la unidad terminal maestra, es decir una vez tomado y descifrado los datos por los sensores estos son enviados a MTU. [12]

La RTU se encuentra conectada al equipo físicamente y puede leer el estado en el que se encuentran las salidas, asimismo la RTU puede enviar señales que pueden controlar los dispositivos, por ejemplo, para abrirlos o cerrarlos, por lo que puede manipular las señales de entrada.

- **Sistemas de Comunicaciones:** el sistema de comunicaciones asume la labor de la transferencia de información, entre la RTU y MTU. Forman parte de la comunicación, los transmisores, los receptores y el medio de comunicación.
- **Transductores o Sensores:** permiten la conversión de una señal física en una señal eléctrica (y viceversa). Su calibración es muy importante para mantener y verificar el buen funcionamiento de los equipos, de ello dependerá la exactitud y precisión del valor de la medida.

La figura 8 muestra la representación de los componentes.

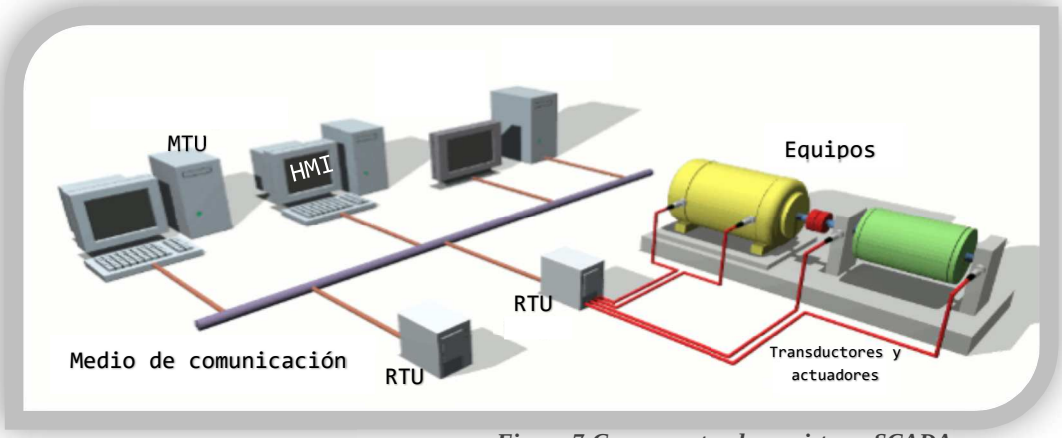


Figura 7:Componentes de un sistema SCADA

2.3 BeagleBone Black

El elemento fundamental de este proyecto es la tarjeta de desarrollo, BeagleBone Black, por lo que es necesario realizar una explicación acerca del mismo.

BeagleBone Black es un producto lanzado por la compañía Texas Instruments en colaboración con DigiKey y Newark Electronics en abril de 2013 y forma parte de la familia BeagleBoard, cuyo coste es de $49 ≈ 41,87€. [16]

BeagleBoard nace con la intención de incentivar el uso de software y hardware *open source*, es decir, manejar el modelo de desarrollo de software basado en la colaboración abierta y a la cual todo el mundo tiene acceso para editar, adquirir y compartir, lo que permite la adquisición de conocimientos gracias al intercambio de ideas entre desarrolladores.

2.3.1 Características principales.

Todos los lanzamientos de BeagleBoard cumplen con la característica de ser pequeños ordenadores, cuyo nombre técnico es placa computadora u ordenador de placa reducida, en inglés: "*Single Board Computer*" o SBC, lo que significa que son ordenadores integrados en un sólo circuito. Para lograr esta peculiaridad el diseño se basa en tener un sólo microprocesador con la memoria RAM, E/S y en conseguir la integración de los elementos fundamentales de una placa madre, además se debe prescindir de componentes y conectores.

Finalmente, se obtiene una tarjeta cuyo tamaño es mucho más reducido si se compara con el de un ordenador, ligera y también logra la eficacia en el manejo de la potencia eléctrica. Sin embargo, si se estropea algún componente electrónico dando lugar a un fallo, o si se requiere actualizar el sistema, se trataría de una tarea complicada de realizar, donde la compra de una nueva tarjeta sería la mejor solución.

Poniendo el foco de atención de nuevo en BeagleBone Black, este dispositivo dispone de las principales características de la original BeagleBone y cuyo punto fuerte es su sistema de desarrollo que está basado en el procesador XAM3359AZCZ100 Cortex A8 ARM de Texas Instruments. [16]

Lo que convierte a BBB en un rival competente es la ventaja de disponer un servidor privado desarrollado en Node.js que se puede ejecutar desde el entorno Cloud9 IDE, debido a que dispone de una réplica local del mismo, esta característica se puede utilizar para diferentes fines y en concreto en este proyecto permitirá establecer la conexión entre la planta que se desea monitorizar y el interfaz web. En la memoria interna del dispositivo se incluye, actualmente por defecto la distribución Debian, lista para comenzar a trabajar, aunque si fuera necesario también es compatible con las diferentes distribuciones de Linux y otros sistemas operativos como por ejemplo Android, sobre estos sistemas se pueden ejecutar programas, lo que hace que sea aún más peculiar. [16]

Con el paso de los años han surgido mejoras que se han manifestado en revisiones de la placa, de las versiones lanzadas hasta el momento, se va a trabajar con la última de ellas, conocida como revisión C , en la cual se aumenta el eMMC (*embedded MultiMediaCard*) de 2 GB a 4 GB, por último cabe destacar que está diseñado para para trabajar a alto nivel. [17]

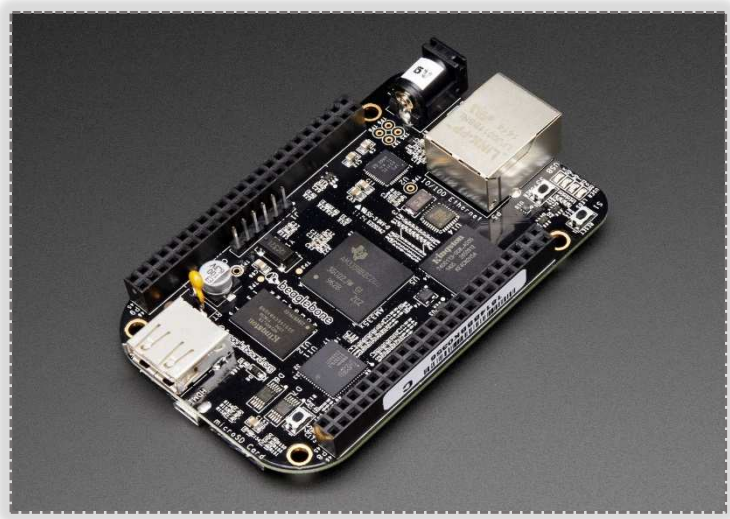


Figura 8: BeagleBone Black [17]

Para concluir se muestra la tabla 3 que resume las principales características de la plataforma.

Tabla 3: Características de BeagleBone Black

Características		
Procesador	1Gz Sitara AM3359AZCZ100	
Motor Gráfico	SGX530 3D, 20M Polygons/S	
Memoria SDRAM	512 MB DDR3L 400Mhz	
Flash	2GB,8bit Embedded MMC	
PMIC	TPS65217C PMIC regulator and one additional LDO.	
Alimentación	miniUSB o Jack DC	5VC Via header de expasión
PCB	3.4" x 2.1", 6 capas	
Indicadores	1 Power, 2 Ethernet, 4 Leds para control del usuario	
Puerto Serial	UART0 via pin 6 3.3V TTL.	
Ethernet	10/100, RJ45	
SD/Conector MMC	microSD, 3.3V	
Entradas Usuario	Botón de reset y poder	
Salida de Video	16b HDMI, 1280×1024 (MAX)1024×768, 1280×720, 1440x900w/Soporte EDID	
Audio	Via HDMI, Stereo	
Conectores de Expansión	Poder 5V, 3.3V VDD_ADC(1.8V)3.3V I/OMcASP0, SPI1, I2C, GPIO(69), LCD, GPMC, MMC1, MMC2, 7 AIN(1.8 MAX) , 4 Timers, 4 Puertos Seriales, CAN0, EHRPWM(0.2) , Interrupciones XDMA, Botón de Poder	
Consumo	210-460mA 5V Dependiendo de la actividad y velocidad del procesador.	
Peso	39.68 gramos	

2.3.2 Comparación con otros microrordenadores.

A día de hoy, BeagleBone Black no es el único microrordenador que podemos encontrar en el mercado hay más plataformas, de hecho, BeagleBone Black es un competidor reciente en conjunto con Raspberry Pi.

A continuación, se va a realizar una comparación entre varias plataformas con características similares, como son Arduino, Raspberry Pi y Dragon Board 410c. Esta comparación tiene el propósito de encontrar las ventajas y desventajas de estos microcontroladores, pero no se va destacar a ninguno, porque cada uno de ellos es útil para diferentes fines. Antes de comenzar se hará una breve descripción de cada uno.

- **Arduino:** es una placa de desarrollo que incorpora un microcontrolador, generalmente de la gama Atmel AVR, fabricados por la empresa estadounidense Atmel. El lenguaje de programación utilizado es C++, pero tiene la peculiaridad de contar con una librería (avr-libc) que adapta este lenguaje al microcontrolador AVR. Arduino incorpora en su software un IDE, un entorno de programación donde el usuario puede escribir el código, compilarlo y depurarlo. El uso de esta placa no está limitado sólo a personas con alto conocimiento de programación, ya que está al alcance de cualquiera, ofreciendo incluso la posibilidad de crear pequeños proyectos.

Figura 9: Arduino Uno [22]

Fundamentado en la vertiente de desarrollo *open source,* software y hardware libre, es bastante popular entre los usuarios porque existen infinidad de archivos que están disponibles para uso personal o comercial y que son gratuitos, además consume poca energía. [23]

- **Raspberry Pi:** esta plataforma, al igual que BBB, es una microcomputadora o microordenador. Se puede utilizar lenguajes de alto nivel como Python, C ++ o Java. Fue desarrollada en Reino Unido por la Fundación Raspberry Pi, con el objetivo de estimular el aprendizaje sobre conceptos relacionados con la informática. El primer lanzamiento de esta placa fue en febrero de 2012, a partir de ahí sacaron más versiones, la última de ellas Raspberry Pi 3 Modelo B+, cuenta con un nuevo procesador y mejor conectividad que los modelos anteriores.

Figura 10:Raspberry Pi 3 Modelo B+ [24]

Al tratarse de un ordenador necesita un sistema operativo, esta plataforma está pensada para correr distribuciones de Linux específicas para Raspberry Pi, por este hecho es perfecta para proyectos Linux, aunque permite la compatibilidad con los escudos Arduino, asimismo el coste de Raspberry es muy bajo.[25]

- **Dragon Board 410c:** fue desarrollado por Qualcomm, y aunque fue creado con la intención de promover el SO Linux, también es compatible con Windows, por lo que nuevamente se trata de un microordenador. Su soporte podría ser de 32 bits o 64 bits. Este microordenador tiene un precio elevado, se debe a la incorporación de extras como Wi-Fi, Bluetooth y GPS.

Figura 11: Dragon Board 410c[26]

- **Original BeagleBone:** BeagleBone fue la primera placa de la familia BeagleBoard, tiene el tamaño de una tarjeta de crédito y funciona con distribuciones Linux, además puede ejecutar versiones de otros sistemas operativos como como Android 4.0 (Android) o Ubuntu (Linux), cuenta con la característica de ser un ordenador que puede conectarse a Internet. Tiene numerosas conexiones de entrada y salida y potencia de procesamiento para análisis en tiempo real proporcionada con un procesador AM335x 720MHz ARM®.

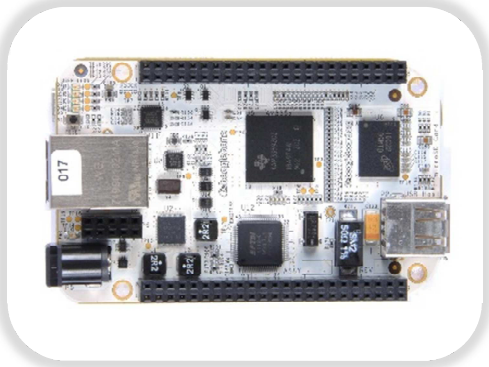


Figura 12: Original BeagleBone [27]

La principal diferencia entre ellos es que Arduino es un microcontrolador, sin embargo, el resto de ellos son microordenadores, es decir, funcionan con un sistema operativo que permite trabajar con programas desarrollados para estos sistemas.

Tabla 4: Sistemas operativos para las diferentes plataformas

	Beagle Bone Black	Raspberry Pi	Dragon Board 410 c
SO	Linux, Android	Linux	Linux, Android, Window

Centrándonos en el BBB, debemos enfatizar que es una combinación del poder de Raspberry al agregar las opciones de la interfaz externa de Arduino.

La ventaja que presenta BBB frente a sus competidores es que cuenta con una memoria flash que permite el almacenamiento de cantidades de datos en un espacio reducido. Entonces, como ya hemos mencionado, tiene un sistema operativo ya instalado en eMMC para que pueda operar desde el principio, desde que se conecta a través del puerto USB al ordenador se puede comenzar a programar en su entorno de desarrollo, Cloud9. En cambio, Raspberry no puede hacerlo porque necesita tener una tarjeta SD para ejecutar el sistema operativo Linux.

Si se desea visualizar gráficos es bastante fácil, y no se necesita un hardware adicional, en otros microcontroladores sí sería necesario, por ejemplo, Raspberry Pi necesitaría un monitor, teclado, ratón, cable HDMI, una fuente de alimentación, cable de red y la tarjeta microSD para conseguir el mismo resultado. En comparación con el resto de dispositivos, BeagleBone se destaca por la gran cantidad de entradas y salidas disponibles para propósitos generales (GPIO), tiene 92 puntos de conexión.

Tabla 5: Comparación de cantidad de GPIO

	BeagleBone Black	Arduino	Raspberry Pi	Dragon Board 410c
GPIO	**92**	14	8	65

Si comparamos sus procesadores, se observa que BeagleBone Black y Dragon Board son bastante similares, tienen la velocidad de reloj más rápida.

Tabla 6: Comparación de la velocidad de sus procesadores

	BeagleBone Black	Arduino	Raspberry Pi	Dragon Board 410c
Velocidad del reloj (Mhz)	**1000**	16	700	1200

Tabla 7: Comparación de la memoria RAM

	BeagleBone Black	Arduino	Raspberry Pi	Dragon Board 410c
RAM	512MB	2KB	512MB	**1GB**

En cuanto a las desventajas, la principal es que este microordenador es uno de los menos conocidos, Raspberry y Arduino están más extendidos, lo que hace que sea más difícil encontrar ayuda o tutoriales en la web, por lo que está muy por detrás de sus competidores en este aspecto. Como se comentó con anterioridad, Raspberry y Arduino están destinados a usuarios con un grado de conocimiento en esta área no demasiado elevado, sin embargo para trabajar con BeagleBone Black es recomendable tener conocimientos más avanzados.

Tabla 8: Comparativa de los costes de cada plataforma

	BeagleBone Black	Arduino	Raspberry Pi	Dragon Board 410c
Coste ($)	**50**	26,56	34	75

Para concluir, se muestra una tabla con la evolución de las principales características del Original BeagleBone al BeagleBone Black.

Tabla 9: Evolución de las principales características

Características	Original BeagleBone	⟹	BeagleBone Black
Procesador	**720MHz**	⟹	**1GHz**
Memoria RAM	**256MB**	⟹	**512MB**
memoria flash eMMC	**2GB**	⟹	**4GB**

2.3.3 BeagleBone como servidor Web.

Como se comentó en el apartado de "*Motivación y Antecedentes*", este proyecto aprovechará las tecnologías web para la creación del SCADA, finalmente se obtendrá una página web cuyo objeto es la visualización de una planta. Para este fin es necesario un espacio donde queden alojados los archivos de la página y que responda a las peticiones del operario, éstas funciones son realizadas por el servidor.

Un servidor web almacena los archivos que constituyen una página web, que serían el documento principal HTML, el archivo JavaScript, la hoja de estilo CSS y las posibles imágenes que contenga, asimismo transmite los datos según los solicite el navegador del usuario. En este caso BeagleBone realizará la función de servidor.

Los servidores son el motor de Internet, son los encargados de generar el tráfico de información. Para establecer correctamente la transferencia de información, todo dispositivo conectado a la red debe de estar

identificado, esta identificación se hace por medio de una dirección IP que es única tanto para el servidor web, como para el ordenador del usuario o cualquier dispositivo que esté conectado.

El método típico de trabajo es conectar la placa al puerto USB del equipo, después de un tiempo de espera, se creará una red virtual con la dirección IP.

A continuación, se adjunta la tabla 10 que especifica el número de IP según el sistema operativo que resida en el equipo.

Tabla 10: Dirección IP según SO[18]

Dirección IP	Tipo de conexión	SO
192.168.7.2	USB	Windows
192.168.6.2	USB	Mac OS X, Linux
192.168.8.1	WiFi	Todos

Sabiendo la dirección IP y la localización de los archivos HTML dentro de BeagleBone, se pueden acceder a ellos desde el navegador del equipo al que se encuentra conectado la placa.

El protocolo en el que se fundamenta la comunicación es el HTTP, este protocolo es el encargado de permitir la realización de una petición de datos y recursos, en este caso de la página web que contiene el SCADA, cabe recordar que este protocolo forma parte del conjunto de protocolos del modelo TCP/IP que es la descripción de protocolos de red.

2.3.4 Cloud9 IDE como interfaz de desarrollo.

En cuanto al entorno de trabajo utilizado se ha seleccionado la plataforma Cloud 9 IDE, que se abre de forma automática y se accede a ella a través del servidor de BeagleBone, por lo tanto, se ha fomentado su uso.

Clou9 IDE es un entorno de desarrollo pensado para que sus usuarios trabajen en línea, además promueve el pensamiento de código abierto. Este entorno proporciona la posibilidad de aplicar multitud de lenguajes de programación donde están incluidos C, C ++, PHP, Python, HTML, JavaScript con Node.js y un largo etcétera. [28][29]

Para acceder a Clou9 desde la placa de desarrollo, primero se debe de conectar al ordenador a través del cable Mini USB, una vez inicializada puede accederse a través del puerto 3000 de la dirección IP de BeagleBone Black. Además de Cloud9 se puede acceder a varias páginas que residen en el servidor y que ofrecen ejemplos de códigos simples.

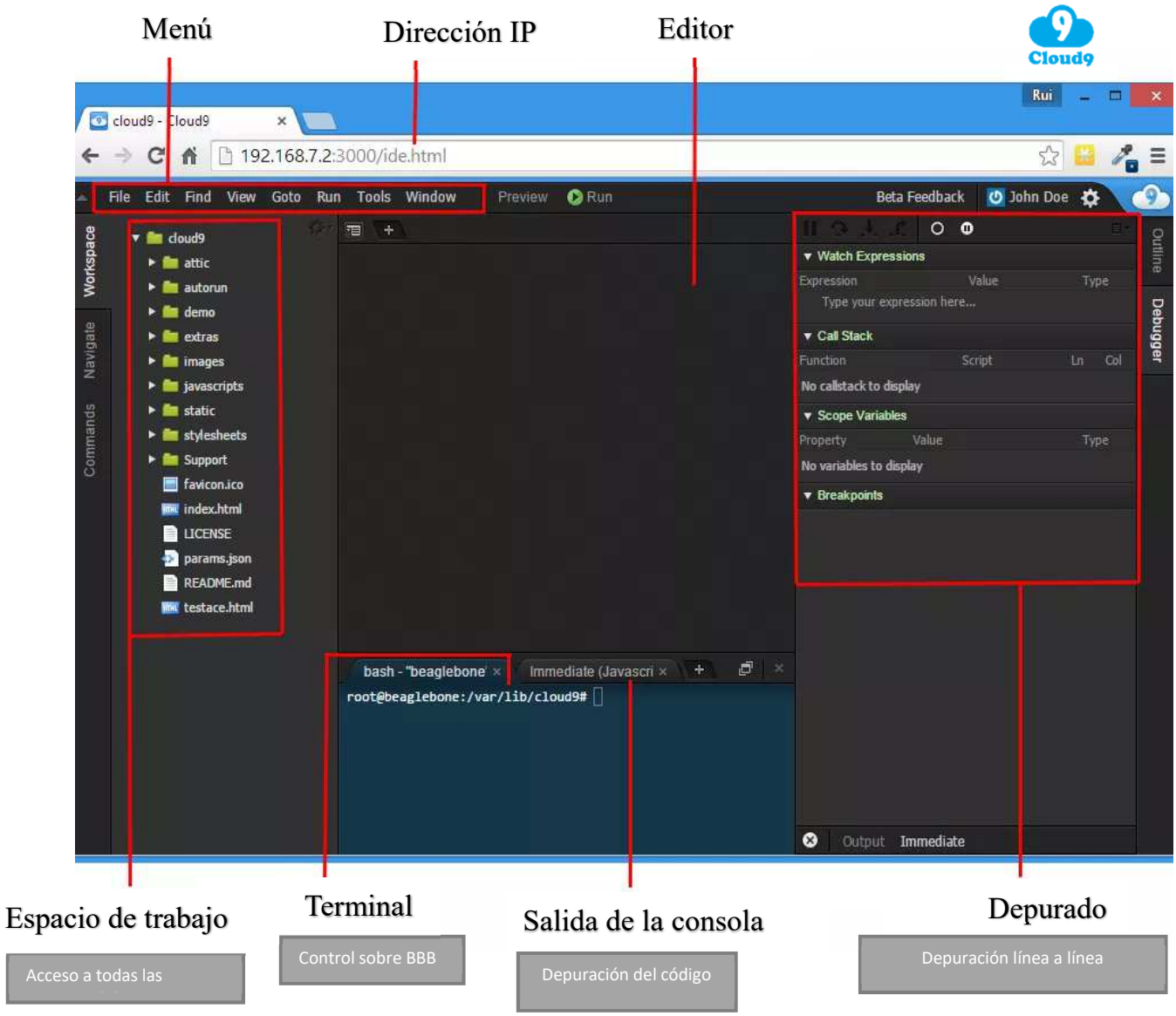


Figura 14: Distribución de la pantalla de Clou9 IDE [19]

2.3.5 Funciones de la librería BoneScript

BoneScript es una librería desarrollada en Node.js, está compuesta por funciones que interactúan con el hardware de la plataforma. Las llamadas a estas funciones pueden hacerse de forma local en la placa, pero también pueden ser ejecutadas desde el navegador, si se añade en el fichero HTML la librería bonescript.js, esta característica es la que permite darle la funcionalidad de sistema SCADA.

Por último se da acceso a las funciones de esta librería después de llamar a la función "require ('bonescript')", que devuelve un objeto "b"(var b = require ('bonescript')) con todas las funciones de la librería.

La tabla 11 incluye un listado de alguna de las funciones más relevantes que se han usado en este proyecto, además se incluye la descripción dada en el enlace [20] .

Tabla 11: Funciones de la librería BoneScript [20]

Funciones	Descripción
pinMode ()	Configurar el modo de un pin de E / S digital.
digitalWrite ()	Poner a alto o bajo un pin de E / S digital.
digitalRead ()	Leer el estado de un pin de E / S digital.
analogWrite ()	Mostrar una señal modulada por ancho de pulso en el pin.
analogRead ()	Leer el voltaje en un pin de entrada analógica.
attachInterrupt ()	Detectar cambios en una línea de entrada digital.
detachInterrupt ()	Eliminar el controlador de interrupción.

2.3.6 Gestión de las E/S.

Para la gestión de las entradas y salidas de BBB se debe de tener el claro que los pines que proporciona se pueden diferenciar principalmente aquellos que sean de uso analógico del digital (hay más usos).

Estos vienen agrupados bajo el nombre genérico GPIO, *General Purpose Input Output,* que hacen referencia a las entradas y salidas digitales, entradas analógicas y salidas PWM. [21]

En la figura 15 se muestra todos los pines de propósito general y en la figura 16 el uso que específico de éstos.

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	GPIO_113
GPIO_111	29	30	GPIO_112
GPIO_110	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71

Figura 15: Pines GPIO [20]

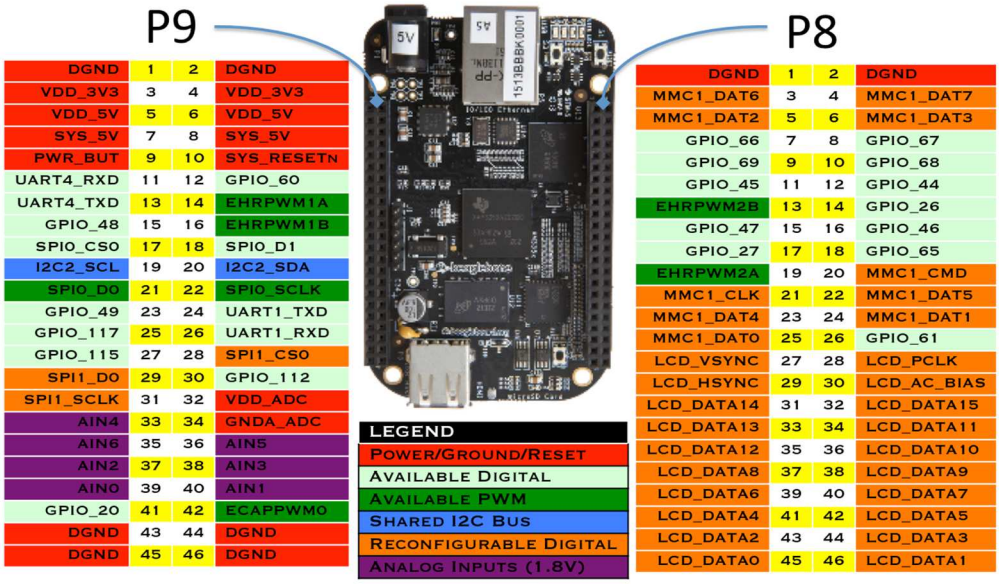


Figura 16: Pines de uso específico [20]

Como se observa hay dos conjuntos de pines de expansión en los conectores P8 y P9, en cada uno de los pines de estos conectores se exponen múltiples señales de la CPU ARM, por lo que existe un multiplexor que permite seleccionar cuál de las señales es accesible en cada momento. [21]

Además, esta diferenciación permite al usuario distinguir entre dos pines con igual enumeración, pues podría tratarse tanto del módulo de expansión P8 como del P9.

E/S digitales.

Se comienza a explicar aquellos pines de uso digital y que por tanto pueden resultar más fáciles. Para la configuración de los mismos se puede seleccionar cualquiera de los múltiples pines para E/S digital a través de los conectores P8 y P9 son todos aquellos indicados con GPIOx_xx.

Se deben de tomar ciertas precauciones y es que estos pines trabajan a 3.3 voltios cuando se configuran como entrada, si se utilizan con una tensión mayor se puede dañar la placa. Asimismo, cuando están configurados como salida, la máxima corriente que se debe solicitar es de 6 miliamperios. [21]

Para la configuración de los pines se aprovecha la librería BoneScript.

En código 1 muestra la configuración y un ejemplo del estado en el que se pueden encontrar.

```
var b = require('bonescript'); //Librería BoneScript
var led = 'USR0'; //Asignación del pin
b.pinMode(led, b.OUTPUT); //Modo de actuación
b.digitalWrite(led, b.HIGH); //Estado del pin
```

Código 1: Configuración de los pines digitales [20]

Si fuera necesario, se puede leer el estado de los pines digitales utilizando el método `digitalRead(pin,[callback])` que tiene como argumento de entrada el pin que se desea explorar y su *callback* especifica el estado del pin y el mensaje de estado de error.

Entradas analógicas.

Si se requiere de una entrada analógica se puede seleccionar cualquiera de los pines de uso analógico, existen 7 entradas analógicas, disponibles en el conector P9 con la referencia AINxx, con 12 bits de resolución (valores 0 a 4095), y una frecuencia de muestreo máxima de 100Ks/s. Esta lectura de valores analógicos es posible gracias los conversores analógico-digital (ADC) disponibles en estos pines. [21]

Al igual que con los pines digitales se debe de tener especial cuidado con la tensión que soportan estas entradas, cuyo rango es de 0 a 1.8 voltios.

Se elige el pin con el número 33: P9_33, y se muestra un ejemplo (ver código 2) de cómo inicializar una entrada para leer valores analógicos:

```
var b = require('bonescript');
b.analogRead('P9_36', printStatus);
function printStatus(x) {
      console.log('x.value = ' + x.value);
      console.log('x.err = ' + x.err);
}
```

Código 2: Lectura del valor analógico de un pin [20]

Salidas PWM.

La señal PWM es una técnica conocida como modulación de ancho de pulso que logra producir el efecto de una señal analógica sobre una carga, a partir de la variación de la frecuencia y ciclo de trabajo de una señal digital. BeagleBone cuenta con pines adaptados para reproducir este tipo de señal, concretamente se cuenta con 3 salidas PWM que se activan cuando se selecciona el modo adecuado en los pines correspondientes [21]:

- EHRPWM0A: pin P9_22 (modo 3) ó pin P9_31 (modo 1).
- EHRPWM0B: pin P9_21 (modo 3) ó pin P9_29 (modo 1).
- EHRPWM1A: pin P8_36 (modo 2) ó pin P9_14 (modo 6).
- EHRPWM1B: pin P8_34 (modo 2) ó pin P9_16 (modo 6).
- EHRPWM2A: pin P8_19 (modo 4) ó pin P8_45 (modo 3).

- EHRPWM2B: pin P8_13 (modo 4) ó pin P8_46 (modo 3).

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	EHRPWM1A
GPIO_48	15	16	EHRPWM1B
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
EHRPWM0B	21	22	EHRPWM0A
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	ECAPPWM2
EHRPWM0B	29	30	GPIO_112
EHRPWM0A	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	ECAPPWM0
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
TIMER4	7	8	TIMER7
TIMER5	9	10	TIMER6
GPIO_45	11	12	GPIO_44
EHRPWM2B	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
EHRPWM2A	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	EHRPWM1B
GPIO_8	35	36	EHRPWM1A
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
EHRPWM2A	45	46	EHRPWM2B

Figura 17: Salidas PWM y Timer [21]

Además, es necesario previamente activar el reloj de las señales PWM para lograr el efecto correcto.

Por último, recordar que la máxima corriente que se puede solicitar a estos pines es de 6 miliamperios y valores por encima de este podrían dañar la placa.

```
var b = require('bonescript');
b.pinMode('P9_14', b.OUTPUT);
//Pin, ciclo de trabajo y frecuencia
b.analogWrite('P9_14', 0.7, 2000.0, printStatus);
function printStatus(x) {
    console.log('x.err = ' + x.err);
}
```

Código 3: Salida PWM [20]

2.4 Adquisición de señales físicas

La adquisición de datos del mundo exterior es una tarea que requiere determinados procesos para acondicionar dicha señal para que ésta pueda ser empleada.

Cabe recordar que las señales físicas son analógicas, sin embargo, éstas deben de ser tratadas como digitales, para que puedan ser procesadas (es imposible tomar muestras con un periodo de muestreo tan pequeño que se aproxime a cero), dicho esto, se necesita acondicionar la señal, para adecuarla al dispositivo encargado de realizar la transformación digital.

El esquema que se adjunta a continuación muestra todas las etapas por las que debe pasar una señal antes de ser utilizada y procesada.

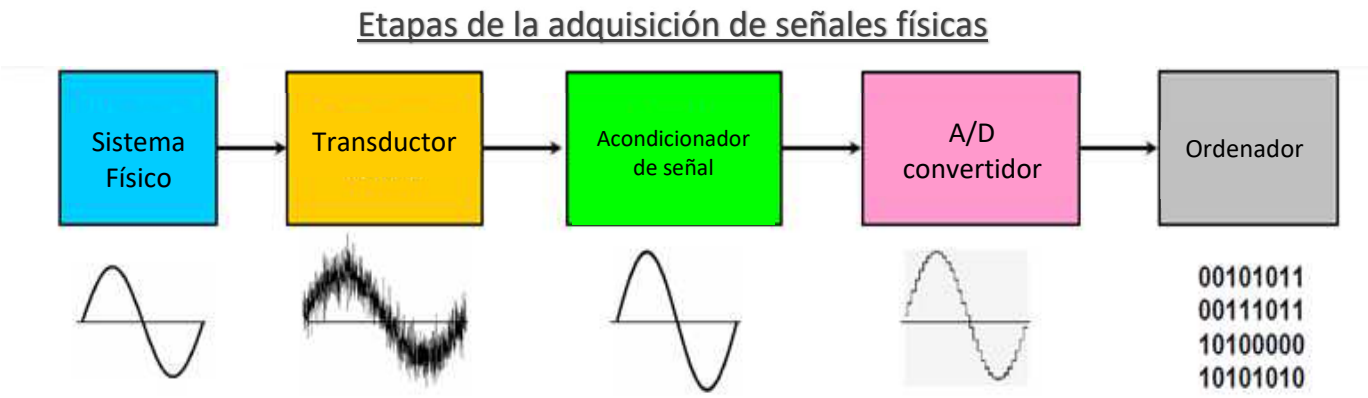


Figura 18: Etapas de la adquisición de señales físicas [30]

Primeramente, se debe de seleccionar la variable que se desea conocer y su correspondiente señal, como objeto para realizar el proceso de adquisición de datos y su posterior acondicionamiento.

En la primera etapa se puede ver como la señal generada es equivalente al fenómeno físico del que se desea conocer su magnitud, como se comentó anteriormente, lo usual es que ésta sea analógica, pero puede ser digital, por ejemplo, si se tratara de un pulsador o interruptor, aquellas señales denominadas analógicas pueden ser valores de temperatura, presión, vibración, nivel o de cualquier otra naturaleza. [36]

A continuación, estos cambios físicos deben ser traducidos a cambios de voltaje o corriente, trabajo que realizan los sensores, también conocidos por transductores, los sensores cuya salida es analógica emplean estándares como son de 0 a 10 voltios, de 0 a 20 miliamperios y de 4 a 20 miliamperios. [36]

El siguiente paso sería el acondicionamiento de la señal, si fuera necesario, ya que puede darse la situación que la señal a su salida no sea adecuada, ya sea porque requiere de un filtrado para la eliminación de ruido, una amplificación o por el contrario atenuación, todos estos tratamientos pueden ser realizados por un módulo acoplado al transductor, y siempre pensando en el tipo de señal que es capaz de procesar el conversor analógico-digital, ya que éste es el fin de la adaptación. [36]

Después de la adaptación de la señal al conversor, el siguiente paso sería la propia conversión analógica-digital, la figura 19 muestra cada uno de los procesos por los que debe pasar para obtener la señal digital deseada. [36]

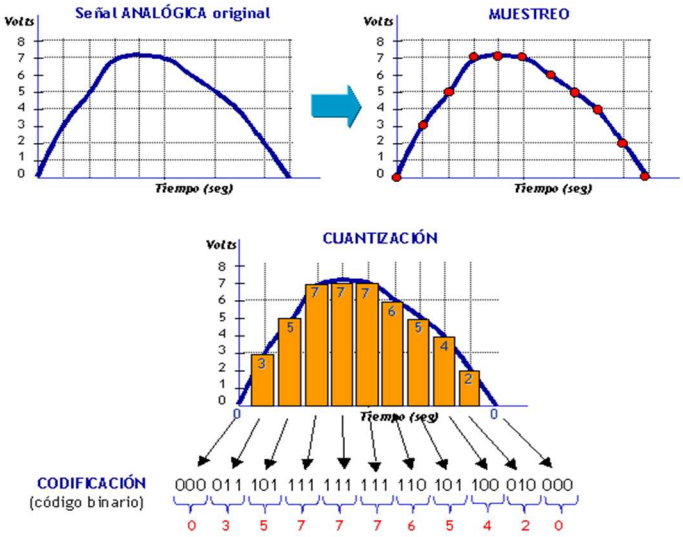


Figura 19:Proceso de conversión analógica digital [37]

El proceso de conversión viene descrito por tres pasos: el muestreo, cuantificación y codificación. El proceso de muestreo toma muestras del contenido de la señal a una determinada frecuencia, conocida como frecuencia de muestreo, esta frecuencia debe de cumplir con el teorema de muestreo de Nyquist-Shannon que enuncia que "la reconstrucción exacta de una señal periódica continua en banda base a partir de sus muestras, es matemáticamente posible si la señal está limitada en banda y la tasa de muestreo es superior al doble de su ancho de banda." [38]

Matemáticamente se representa como:

$$F_s > 2F_{max} \equiv 2B$$

Ecuación 1: Teorema del muestreo [38]

Donde F_s es la frecuencia de muestreo, F_{max} la frecuencia máxima de la tasa de cambio de la señal y B es el ancho de banda.

El proceso de cuantificación relaciona cada valor continuo a un valor discreto, es decir, el valor de cada muestra de la señal se representa como un valor elegido de entre un conjunto finito de posibles valores. Aquí entra en juego la resolución del ADC, una mayor resolución (16 bits = 65535 niveles) permite un error de cuantificación menor (valor de la señal continua – valor de la señal discreta).

Por último, dentro del proceso de conversión, estaría la parte de codificación simplemente se representa el valor de la cuantificación empleando códigos establecidos y estándares como la codificación binaria. [39]

Ahora es posible procesar la información y tratarla para el control de la planta.

2.5 Señal de control 4-20mA

Este apartado es meramente electrónico, se incluye debido a la importancia que supuso para este proyecto generar dicha señal.

La señal 4 a 20 miliamperios es un estándar a nivel de instrumentación industrial, se caracteriza por hacer corresponder el valor de 4 miliamperios de corriente al 0% de la medida, y el valor de 20 miliamperios representa un 100% de la medida, por lo que cualquier otro valor entre 4 y 20 miliamperios representa un porcentaje entre 0% y 100%, la representación queda escalada de forma lineal (ver figura 20). [40] [41]

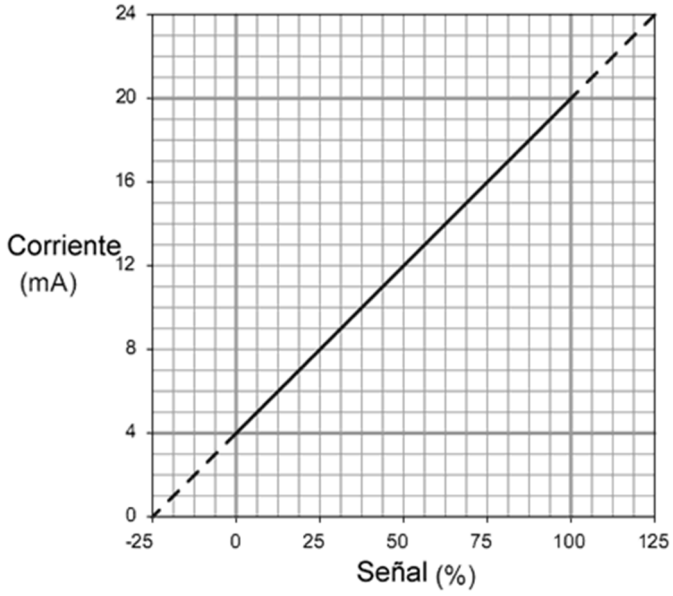


Figura 20: Relación lineal entre la corriente y el valor de la señal [41]

El hecho de que esta señal se haya estandarizado es debido a la robustez que presenta en la transmisión de la señal, si se implementara cualquier señal de tensión, por ejemplo, de 0 a 10 voltios en continua, ésta no sería inmune al ruido eléctricos o efectos electromagnéticos del entorno, de igual forma podría no soportar las caídas de tensión debido a la resistencia que ofrece el cable. Sin embargo, la señal de corriente ofrece una mayor resistencia contra efectos electromagnéticos, ya que las perturbaciones electromagnéticas se manifiestan en variaciones de tensión y provocan pocas variaciones de corriente, así mismo la transmisión en intensidad es independiente de la resistencia de los cables.

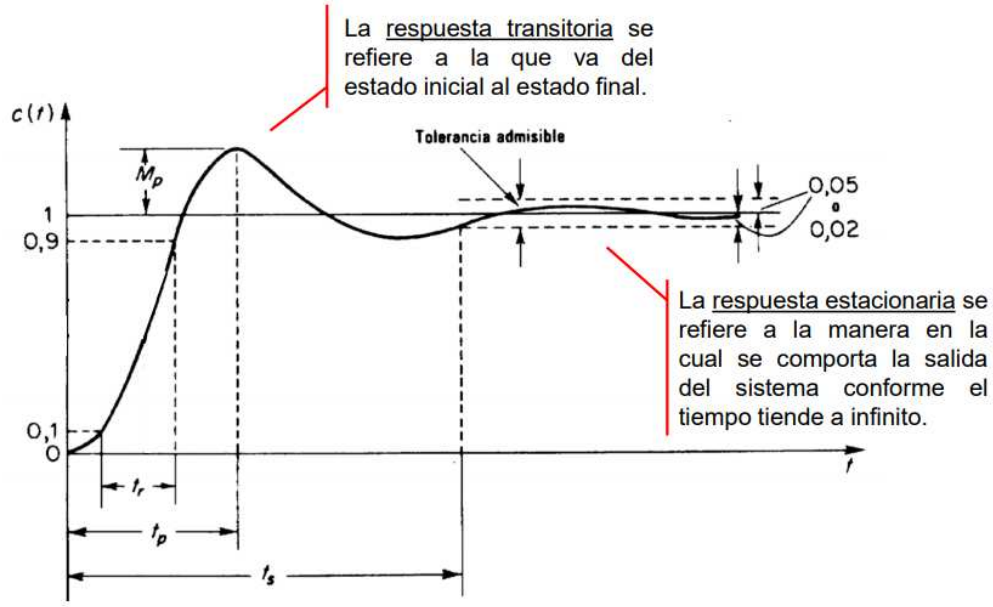
La principal ventaja de este método es la capacidad de diagnóstico del cableado, valores por debajo de 4 miliamperios pueden significar roturas en el cableado, así como los que se encuentran por encima de 20 miliamperios un cortocircuito en la línea, son traducidos como roturas, cabe destacar que esta capacidad no sería posible si la señal fuera de 0 a 20 miliamperios es la ventaja de haber trasladado el punto cero a 4 miliamperios, lo que se denomina "cero vivo". [40] [41]

Es ampliamente utilizado cuando se tienen transmisores de campo de 2 hilos o para transmitir datos digitales del protocolo de comunicación HART, para aplicaciones con señales neumáticas (3-15 psi). [40] [41]

Por último, es una señal que no supone un peligro para el ser humano ya con una tensión nominal de 24 voltios (tensión de seguridad), el valor de corriente que se considera de riesgo debe de ser igual o por encima de 30 miliamperios.

2.6 Teoría del control PID

Antes de comenzar se deben de aclarar conceptos acerca de los parámetros de la respuesta temporal, para ello se incorpora la siguiente imagen, que los ejemplifica.



Gráfica PID

El control PID es un controlador sofisticado que está compuesto por una acción proporcional (P), acción integral (I) y por último la derivativa (D), cada una de ellas tiene una finalidad durante el control. [43][44][45]

- La acción proporcional P, ayuda a alcanzar el valor deseado, pero no elimina el error por completo en el estado estacionario, genera una salida que es proporcional al error. Matemáticamente se puede describir como:

$$u(t) = K_p \cdot e(t)$$

Ecuación 2: Acción proporcional

donde K_p es la ganancia y es un parámetro ajustable, también se puede hablar de la banda proporcional PB, pero ambos son términos equivalentes. [43][44][45]

- La acción integral I, ayuda en la corrección del error en el estacionario disminuyéndolo, provocado por la acción proporcional, asegura la estabilidad de la respuesta pues si se manifestaran perturbaciones éstas serían corregidas, origina una salida teniendo en cuenta la magnitud del error y el tiempo que éste ha permanecido, pero se considera una acción lenta. Su ecuación matemática se describe de la siguiente forma:

$$u(t) = K_i \cdot \int_0^t e(\tau)d\tau$$

Ecuación 3: Acción Integral

- La acción derivativa D, tiene en cuenta la velocidad a la que varía el error o lo que los mismo, su tasa de cambio, es una manera de anticiparse antes de que la magnitud del error sea demasiado grande y es bastante útil para sistemas con una dinámica rápida. Su estructura es la siguiente. [43][44][45]

$$u(t) = K_d \cdot \frac{de(t)}{dt}$$

Ecuación 4: Acción Derivativa

Finalmente, la salida de un controlador PID viene descrita por la siguiente fórmula

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau)d\tau + K_d \cdot \frac{de(t)}{dt}$$

Ecuación 5: Salida controlador PID

Otra versión de la ecuación puede ser la siguiente donde en la accion integral y derivativa aparecen los términos T_i y T_d que representan el tiempo integral y derivativo respectivamente. [43][44][45]

$$u(t) = K_p \cdot e(t) + \frac{K_p}{T_i} \cdot \int_0^t e(\tau)d\tau + K_p \cdot T_d \cdot \frac{de(t)}{dt}$$

Ecuación 6: Salida controlador PID en función de Kp

Aunque sólo se haya comentado el controlador PID, el lector debe saber que se pueden realizar múltiples combinaciones de las acciones: P, I, PI, PD y PID

2.7 Tecnología WEB aplicada.

Este proyecto se ha focalizado en la aplicación de los lenguajes web para el desarrollo del sistema SCADA, como se comentó al comienzo de este proyecto, lo que se pretende es diseñar un SCADA con unas herramientas que sean cómodas de utilizar, que el coste de su uso sea bajo, en este caso gratuito, y en un futuro si desea modificar que pueda hacerse.

Figura 21:Tecnologías de frontend [88]

En el desarrollo web existen dos tecnicismos anglosajones conocidos como *frontend* y *backend*, cada uno con funciones diferentes. [46][47][48]

El término "*frontend*", es aquel que hace referencia a todas las tecnologías que corren del lado del cliente y por lo tanto lo que ve el usuario. Es imprescindible conocer los siguientes lenguajes: HTML, JavaScript y la hoja de estilos CSS, para la creación de la página web, este proyecto se centra en el *frontend* del sistema SCADA.

Por otro lado, el término *backend*, especifica aquellos trabajos que se ejecutan en el lado del servidor. Mejora la experiencia del usuario al encargarse de que todo lo que está detrás del sitio web, recoge los datos y los procesa para enviarlos de nuevo al usuario, asimismo se encarga de peticiones de acceso a la base de datos. Algunas de las tecnologías que se pueden aplicar son PHP, Ruby, Python, JavaScript, SQL, MongoDb, MySQL, en el caso particular de este trabajo se aplicará Node.js [46][47][48]

2.7.1 HTML y HTML5.

La base de cualquier página web es su archivo HTML, este es un lenguaje de marcado, no se debe de considerar como lenguaje de programación, pues dentro de cualquier código de este segundo tipo se pueden encontrar estructuras complejas o con una lógica, sin embargo, el lenguaje de marcado inicialmente surgió con el objetivo de añadir un formato al texto escrito, a día de hoy han evolucionado hasta llegar el punto de ser el pilar de las páginas web.

Este lenguaje se fundamenta en la utilización de etiquetas que adquieren su significado cuando son leídas e interpretadas por el navegador, las etiquetas indican como está estructurada la página web según ha sido diseñada, es decir, donde se mostrarán exactamente los diversos elementos que constituyen la página. [49][50][51]

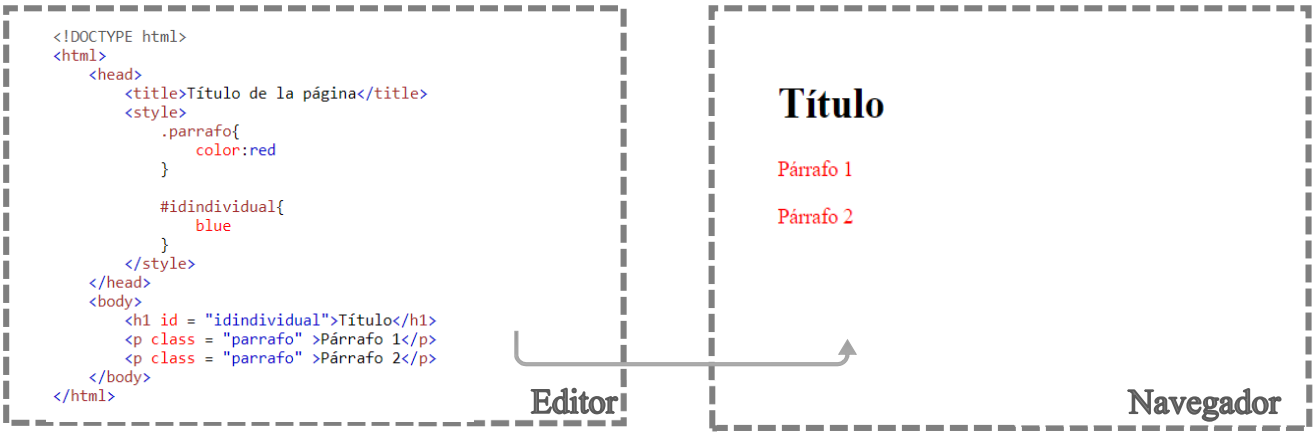
De manera general las etiquetas incluyen en su interior un fragmento de texto, se pueden reconocer porque están encerrados por los signos de "menor" (<) y "mayor" (>), asimismo deben de tener una marca de inicio y otra de final.

Para personalizar cada etiqueta a gusto del diseñador web HTML proporciona atributos que añaden una información adicional, por ejemplo, la etiqueta `<a></a>` se emplea para definir enlaces a cualquier otra página o incluso secciones dentro de una misma página, pero para ello hay que indicar la dirección por medio del atributo `href=""`

Una de las características de este lenguaje es el uso del texto plano, por lo que no se necesita de un programa intermediario para la interpretación del mismo, cualquier editor de texto puede trabajar con este lenguaje.

El modo de trabajo es bastante sencillo, según el elemento que se desee incluir se escoge una u otra etiqueta que sea adecuada al contenido. Primeramente, se abre la etiqueta, se incluye la información necesaria: texto, imágenes, tablas de datos, o vínculos entre otros, y por último se cierra esta misma etiqueta. Como se ha visto a las etiquetas se les puede añadir atributos, éstos pueden ser específicos de la etiqueta, aunque también los hay que se pueden aplicar a diversas de ellas.

El código 4 es un ejemplo de estructura básica del cuerpo del HTML.



Código 4: Estructura básica HTML

Dentro de un código HTML se pueden diferencias dos secciones, la primera marcada con la etiqueta <head> </head>, es la cabecera de la página, en la cual se adjunta información general acerca del documento, incluyendo su título y enlaces a scripts y hojas de estilos, esta información no es visualizada por el usuario, pero es útil para el navegador. La otra etiqueta sería <body></body>, esta etiqueta engloba todo el contenido que, sí es visible para el usuario, en él se pueden incluir párrafos, tablas, botones, imágenes y de más objetos.

Como cualquier otro lenguaje, HTML a lo largo de los años de su uso ha ido evolucionando, actualmente se encuentra la versión número cinco conocida como HTML5 que cuenta con nuevos elementos, atributos y comportamientos.

En la figura 22 se puede observar las etiquetas básicas que cuenta HTML5 (dentro del cuerpo

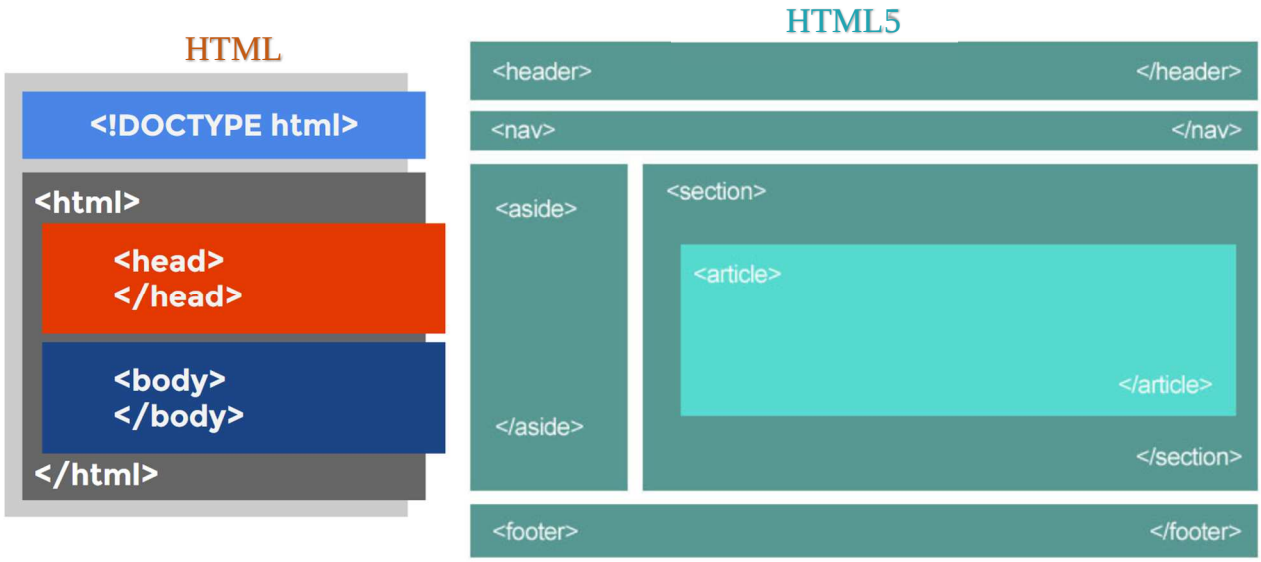


Figura 22: Comparativa entre la semántica de HTML y HTML5

2.7.2 Árbol DOM

DOM es un conjunto de herramientas utilizadas en la manipulación de los documentos escritos con lenguajes de marcas. *"Técnicamente, DOM es una API de funciones que se pueden utilizar para manipular las páginas XHTML de forma rápida y eficiente."* [52]

DOM examina la estructura interna del archivo en lenguaje de marcas y lo analiza, posteriormente éste es transformado y representado en forma de jerarquía de nodos. De esta forma queda definida la estructura lógica del documento y los diferentes nodos donde figura el contenido del archivo original y sus relaciones.

Es un error pensar que estas técnicas solo están asociadas al desarrollo web, pues son aplicables a cualquier tipo de lenguaje de programación.

El árbol DOM se crea una vez la página web se carga por completo, ya definidos los nodos y las relaciones se pueden aplicar de forma correcta las funciones DOM. Gracias a esta técnica se puede llevar un control exhaustivo del contenido de la página, y entonces poder manipular con él, añadiendo, eliminando y modificando algunos de sus elementos.

Para que el concepto quede más claro la figura 23 detalla el árbol DOM de una tabla.

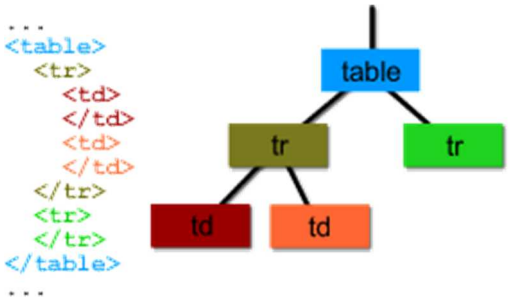


Figura 23: Árbol DOM de una tabla [53]

Navegadores como Firefox permiten ver esta representación que ayuda al programador a comprender la estructura del documento y su comportamiento, así como la detección de fallos.

El objeto principal o nodo raíz queda definido como `HTMLDocument` y a partir de éste se definen el resto de elementos, para acceder a ellos existen varios métodos, una forma directa de acceso a los atributos es con las funciones `getAttribute(nombre)` `setAttribute(nombre,valor)`, los cuales permiten el acceso y su modificación. De igual forma para acceder a los nodos, en principio sería necesario acceder primero al elemento padre e ir descendiendo en la jerarquía, este método de trabajo puede ser tedioso si se tienen en cuenta la cantidad de antecesores que puede tener un elemento, por ello DOM soluciona este problema con funciones de acceso más directo como `getElementsByTagName()`, `getElementsByName()` y `getElementById()`.[54]

2.7.3 JavaScript.

Con la evolución de las páginas web, nació la necesidad de permitir a los diseñadores crear páginas que fueran susceptibles a los cambios con las que poder interactuar, a diferencia de las páginas estáticas, con estilo, que existían hasta el momento. Se necesitaban diseños más complejos. Es por ello que aparece JavaScript, en el año 1995 de la mano de su creador Brendan Eich y con el nombre de LiveScrip, más adelante pasaría a ser JavaScript. [55-59]

Actualmente, es el lenguaje de *frontend* por excelencia en la actualidad, permite crear contenido dinámico, controlar archivos multimedia, crear animaciones, sin necesidad de usar demasiadas líneas de código.

Resulta cómodo de utilizar gracias a su sintaxis, la cual es bastante similar al lenguaje de programación C. Uno de sus puntos fuertes en la amplia librería que posee, que ayuda a disminuir el código, para que éste sea más eficiente, como la biblioteca multiplataforma jQuery creada inicialmente por John Resig.

Los scripts o programas pueden ser insertados en el documento HTML dos formas diferentes, la primera es directamente en el interior del documento entre las etiquetas <body></body> con su propia etiqueta `<script></script>` o pueden estar ubicados en ficheros externos usando:

```
<script type="text/javascript" src="codigo.js"></script>
```

Aunque se ha comentado que es potencialmente usado en el *frontend*, cabe decir que también existe un JavaScript que se ejecuta en lado del servidor, por ejemplo, en el entorno de Node.js

Este lenguaje es interpretado, que no es lo mismo que compilado por el navegador, y aunque la mayoría de los navegadores como Firefox, Google Chrome, Safari, Mozilla, entre otros, soportan el uso de JavaScript, los usuarios siempre tienen la opción de activarlo o desactivarlo en los mismos.

A continuación, se muestran las principales características:

- Ligero.
- Multiplataforma.
- Es Imperativo y estructurado.
- Prototipado.
- Orientado a objetos y eventos.
- Es Interpretado, no se compila para ejecutarse.

Por último, destacar que existe un estándar cuya última versión es ECMAScript6 que determina como trabajar con JavaScript.

Al igual que en cualquier otro lenguaje de programación la forma de proceder que se sigue JavaScript no difiere del resto de ellos. Aunque algo característico de este lenguaje son los eventos que se pueden incluir a elementos de la página, un ejemplo de evento sería cuando el usuario pulsa un botón.

2.7.4 CSS y CSS3.

Las siglas CSS es un lenguaje que permite definir la apariencia de un documento previamente escrito en un lenguaje de marcas, en este caso HTML. Es importante hacer hincapié en esto, pues con el primer lenguaje simplemente se definen los elementos de la página web y con este segundo se le agrega un aspecto visual para que éste sea más agradable, incluso para los casos en los que la información es pronunciada a través de un dispositivo de lectura o para dispositivos táctiles basados en Braille, este lenguaje es esencial. [60-62].

El Consorcio WWW, abreviatura de las palabras en inglés, World Wide Web Consortium, más conocido como W3C, es un es un consorcio internacional que genera recomendaciones y estándares que aseguran el crecimiento de la World Wide Web a largo plazo[X], precisamente CSS es una de estas especificaciones que se lanzó con el objetivo de separar los contenidos de los documentos escritos con lenguajes de marcado. [63]

Por un lado, se crea el elemento y por otro se especifica su apariencia, con CSS resulta bastante fácil de añadir:

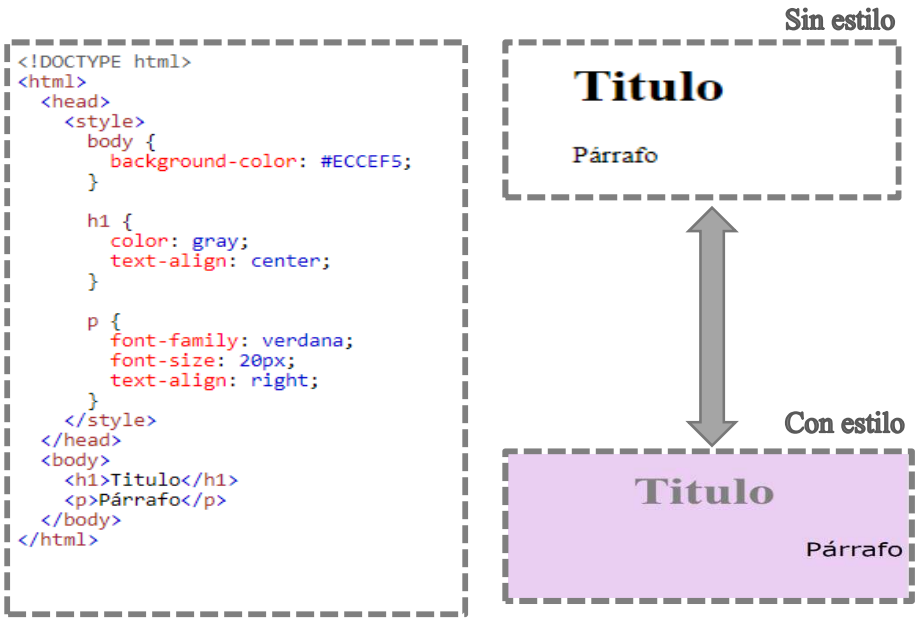
- Color.
- Fondo.
- Fuente y tamaño de letra.
- Ajustar tamaño y posición de los elementos.
- Espacio entre elementos.
- Márgen.

Para añadir estilo a un determinado componente, es necesario que este esté identificado, para ello en el documento HTML se debe de añadir un atributo global `id = ""` que define un identificador único, el cual no debe repetirse en todo el documento, o mediante un selector de clase, `class = ""` que si se le puede atribuir a varios elementos.

Seguidamente, se definen unas propiedades a dichos elementos, por ejemplo, la fuente y tamaño de la letra.

Por último, se hace referenciar la hoja de estilos a un documento en concreto, al igual que los códigos de JavaScript, existen dos formas de efectuar: la primera, incluyendo la hoja de estilos de forma externa a través de un enlace, entre las etiquetas `<head></head>` añadiendo el elemento HTML `<link>`, que especifica la relación entre el documento actual y el recurso externo, muy utilizado por CSS, `<link rel="stylesheet" href="estilo.css">`, y la segunda, adjuntando la hoja de estilos de manera directa en el documento nuevamente entre las etiquetas `<head></head>` con una etiqueta que hace referencia al estilo `<style></style>`. A continuación, se muestra un ejemplo, donde el método utilizado sería segundo.

Como se puede observar en este simple ejemplo, la diferencia es clara, esta herramienta ha resultado muy útil para del diseño del sistema SCADA.




```
<!DOCTYPE html>
<html>
  <head>
    <style>
      body {
        background-color: #ECCEF5;
      }

      h1 {
        color: gray;
        text-align: center;
      }

      p {
        font-family: verdana;
        font-size: 20px;
        text-align: right;
      }
    </style>
  </head>
  <body>
    <h1>Titulo</h1>
    <p>Párrafo</p>
  </body>
</html>
```

Código 5: Comparativa entre el resultado con y sin estilo

Al igual que el resto de lenguajes, CSS cuenta con una evolución, CSS3 es la última versión del mismo, que incorpora mejoras, como puede ser transiciones o animaciones (sin uso de JavaScript), elementos div (sección de contenido) flexibles, sombras, gradientes y esquinas redondeadas, aunque pueden cambiar para cada tipo de navegador, no sólo en CSS3 si no también en CSS

2.7.5 SVG.

En las páginas web además de la apariencia, la información que se presenta también es importante, así como su calidad, en muchas ocasiones se requiere incluir imágenes o gráficos, de forma usual se opta por incluirlos en formato JPG-JPGEG, PNG o GIF, para las animaciones. [64] Sin embargo, cuando éstos son ampliados llegan a perder mucha información, y hacen que empeore su visualización, se pixelan y no se llegan a ver con nitidez.

En 2001 W3C lanza como recomendación SVG, no obstante, este estándar abierto fue impulsado por W3C en 1999, es un formato de gráficos vectoriales bidimensionales, tanto estáticos como animados, en formato XML, que soluciona la problemática comentada anteriormente. [67][68]

La solución propuesta es la incorporación de Gráficos Vectoriales Escalables, de ahí las siglas SVG, tienen una base fundamentada en la matemática de vectores, es decir, que se puede escalar (imagen más grande o pequeña) y tener la misma resolución inicial, lo que ayuda a construir páginas web adaptables, sin embargo, con los mapas de bits no sucede.

Figura 13: Ejemplo de gráfico adaptativo [65]

En cuanto a su implementación, navegadores como Firefox, Google Chrome, Safari, son capaces de mostrar imágenes en formato SVG sin necesidad de complementos externos. Aunque otros navegadores web, necesitan un conector o plug-in. [70]

Queda expuesto que el formato SVG presenta numerosas ventajas para el diseño gráfico, en particular, en los entornos digitales.

Existen editores de gráficos vectoriales que facilitan el trabajo del diseñador web, un ejemplo sería Inckscape, editor de código abierto, emplea el SVG como formato nativo, permite crear el gráfico además de ver su código en XML, este editor ha sido utilizado para el desarrollo del trabajo.

En la figura 25 se muestra la pantalla del editor.

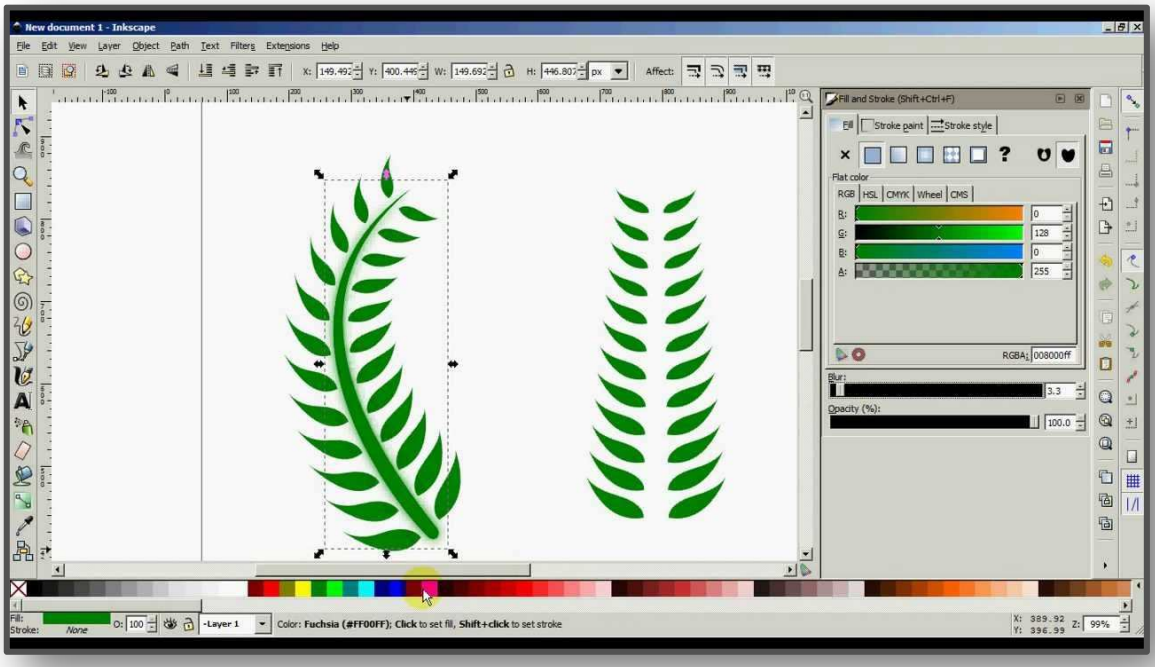


Figura 14: Pantalla de edición del programa Inckscape [69]

2.7.6 Node.js.

En apartados anteriores se han comentado tecnologías aplicables al fro*ntend*, o el lado del cliente. Sin embargo, también se debe de prestar atención al *backend* o el lado del servidor.

En el apartado 2.5.2 en el cual se explica JavaScript, se comentó que este lenguaje se puede aplicar tanto en el lado del cliente como en el servidor, aunque es más común en el primero.

Node.js es precisamente un entorno JavaScript aplicable al lado del servidor, con E/S de datos una arquitectura orientada a eventos y que permite una programación asíncrona, ya que ofrece la posibilidad de devolver el control al programa llamante sin haber terminado las operaciones llamadas, mientras éstas se ejecutan en segundo plano, además está basado en el motor V8 de Google, asegurando una velocidad muy alta de compilación. [71][72]

El servidor proporcionado por BeagleBone está basado en Node.js, en el código 6 muestra un ejemplo de un servidor HTTP escrito en Node.js:

```
// Cargar el modulo HTTP HolaMundo con Node
var http = require('http');

// Configurar una respuesta HTTP para todas las peticiones
function onRequest(request, response) {
  console.log("Peticion Recibida.");
  response.writeHead(200, {"Content-Type": "text/html"});
  response.write("Hola Mundo");
  response.end();
}

var server = http.createServer(onRequest);

// Escuchar al puerto 8080
server.listen(8080);

// Poner un mensaje en la consola
console.log("Servidor funcionando en
http://localhost:8080/");
```

Código 6: Creación de servidor con Node.js [73]

Capítulo 3: Desarrollo del proyecto

3.1 Alcance del proyecto

En este capítulo se describen los procesos y decisiones tomadas en el desarrollo del sistema SCADA aplicando la tecnología web, se harán diseños tanto a sistemas más simples como para la maqueta de una planta industrial, de esta forma se podrá ver que es fácilmente aplicable a cualquier tipo de sistema.

Antes de comenzar con el desarrollo hay que destacar que la planta cuenta con su propio sistema de control, sin embargo, a raíz de la implantación del sistema SCADA se decide realizar un control (tanto On-Off como PID) de forma externa y utilizando la plataforma BeagleBone. Igualmente, el fabricante también proporciona un sistema de visualización del estado de la planta, pero se opta por el desarrollo de uno nuevo, usando la nueva tecnología y comprobando su amplio potencial.

3.2 Puesta en marcha de BeagleBone Black.

La placa BeagleBone Black es el pilar más importante de este proyecto, pues da soporte a la página web donde reside el sistema SCADA, además realiza la función de interacción con la planta industrial, y gestiona el control de la misma, por ello se ha cuidado que todo lo relacionado con ella se efectúe de la manera correcta.

Empezando por la parte más básica, la alimentación del dispositivo, cabe decir que como cualquier otro dispositivo la requiere, pero con la ventaja de que no necesariamente tiene que ser de una fuente de alimentación externa, en este caso se provee mediante el propio sistema de comunicación con el ordenador.

Otro aspecto a tener en cuenta es el establecimiento de la comunicación entre los componentes, en este caso aparece además de la placa, un ordenador con acceso a internet para poder al sistema SCADA, y la propia planta.

3.2.1 Sistemas de comunicaciones.

Este apartado se dividirá en dos secciones, la primera sección dedicada a la comunicación entre la placa de desarrollo y el ordenador que le da soporte, y el segundo apartado destinado a la comunicación entre el servidor proporcionado por BeagleBone y el sistema SCADA.

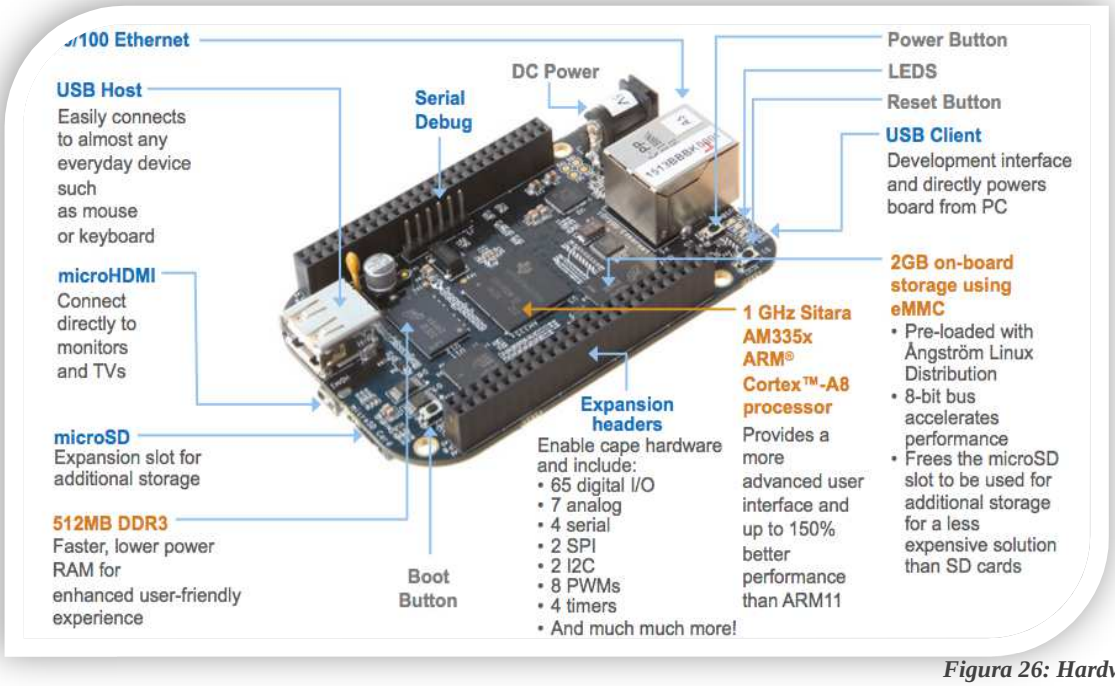


Figura 26: Hardware BBB [74]

BeagleBone cuenta con las siguientes interfaces de salida y entrada, que se muestran en la imagen siguiente.

- USB "*host*": " USB On-The-Go " u OTG por sus siglas en inglés, permite conectar cualquier dispositivo a BBB, en este caso la plataforma pasa del rol esclavo a maestro. [75]
- USB "*client*": este puerto permite a un ordenador controlar y enviar órdenes a la placa.
- microHDMI: esta interfaz está diseñada para conectar a BBB un dispositivo que pueda crear y reproducir audio y vídeo de alta definición. [76]
- MicroSD: formato que da soporte a tarjetas de memorias flash que almacenan datos y archivos.
- Puerto serie: interfaz de comunicaciones de datos digitales.
- Conexiones de expansión: estos pines han sido comentados con anterioridad.

3.2.1.1 Bus de comunicación USB.

En concreto para esta aplicación se ha tomado el puerto USB "client" , las siglas USB provienen de Bus Universal en Serie (en inglés: Universal Serial Bus), es un bus de comunicaciones que sigue un estándar que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica entre ordenadores, periféricos y dispositivos electrónicos. [77]

Gracias a la conexión USB, no se requiere de ninguna fuente de alimentación externa para nuestro dispositivo, ya que USB distribuye alimentación a todos los dispositivos conectados al mismo.

El hecho de que se haya seleccionado este puerto es debido a que BBB ejecuta un servidor DHCP cuando se conecta a través de USB y proporciona al ordenador conectado una dirección IP de 192.168.7.1 o 192.168.6.1, dependiendo del tipo de adaptador de red USB compatible con el sistema operativo del mismo,

reservándose para él la dirección IP 192.168.7.2 o 192.168.6.2 (Windows o Mac OS X/Linux respectivamente).

Aunque no es necesario preparar la placa para la conexión, por el contrario, sí que lo es para el ordenador, hay que instalar determinados drivers proporcionados por la página oficial de beagleboard.org [20]

3.2.1.2 Comunicación entre servidor y cliente.

Al conectarse mediante USB BBB, actúa como servidor DHCP, que es una extensión del protocolo Bootstrap (BOOTP) desarrollado en 1985, se trata de un protocolo de configuración dinámica del servidor que permite a cualquier máquina obtener su configuración red de forma automática.

La asignación de las direcciones IP con DHCP se basa en un modelo cliente-servidor: el terminal que quiere conectarse solicita la configuración IP al servidor DHCP que, por su parte, recurre a una base de datos que contiene los parámetros de red asignables. Este servidor, puede asignar los siguientes parámetros al cliente con ayuda de la información de su base de datos:

- Dirección IP única
- Máscara de subred
- Puerta de enlace estándar
- Servidores DNS
- Configuración proxy por WPAD (Web Proxy Auto-Discovery Protocol)

La comunicación entre el cliente DHCP y el servidor DHCP se realiza a través de peticiones, el cliente envía un paquete, el servidor escucha la petición, si hay varios servidores el cliente selecciona uno y el resto conoce esta decisión, por último, el servidor envía de nuevo un paquete con los datos solicitados hacia el cliente. [78]

3.2.1.3 Creación del servidor.

Ahora bien, una vez conocida la dirección IP del dispositivo conectado a la red, lo que interesa es iniciar la navegación y buscar el sistema SCADA que se aloja en el servidor web BeagleBone, esto quiere decir que guarda todo lo relacionado con la página web, y esperará que sea solicitado mediante el navegador del usuario.

Para ello se puede crear un servidor mediante Node.js por ejemplo, o simplemente usar el servidor web que corre directamente desde BeagleBone y que está basado en Node.js, éste se ejecuta cuando se apunta en el navegador hacia la dirección 192.168.7.2, las páginas residen en la dirección /var/lib/cloud9, como se puede observar la plataforma Cloud 9 ayuda en el desarrollo de creación, seguidamente se debe de crear una carpeta donde queden guardados todos los archivos del sitio web (documentos HTML, JavaScript, CSS, SVG e imágenes) y después apuntar hacia esta dirección en el navegador.

Cuando no se especifica el nombre del archivo HTML, se abre de forma predeterminada aquel con el nombre index.html.

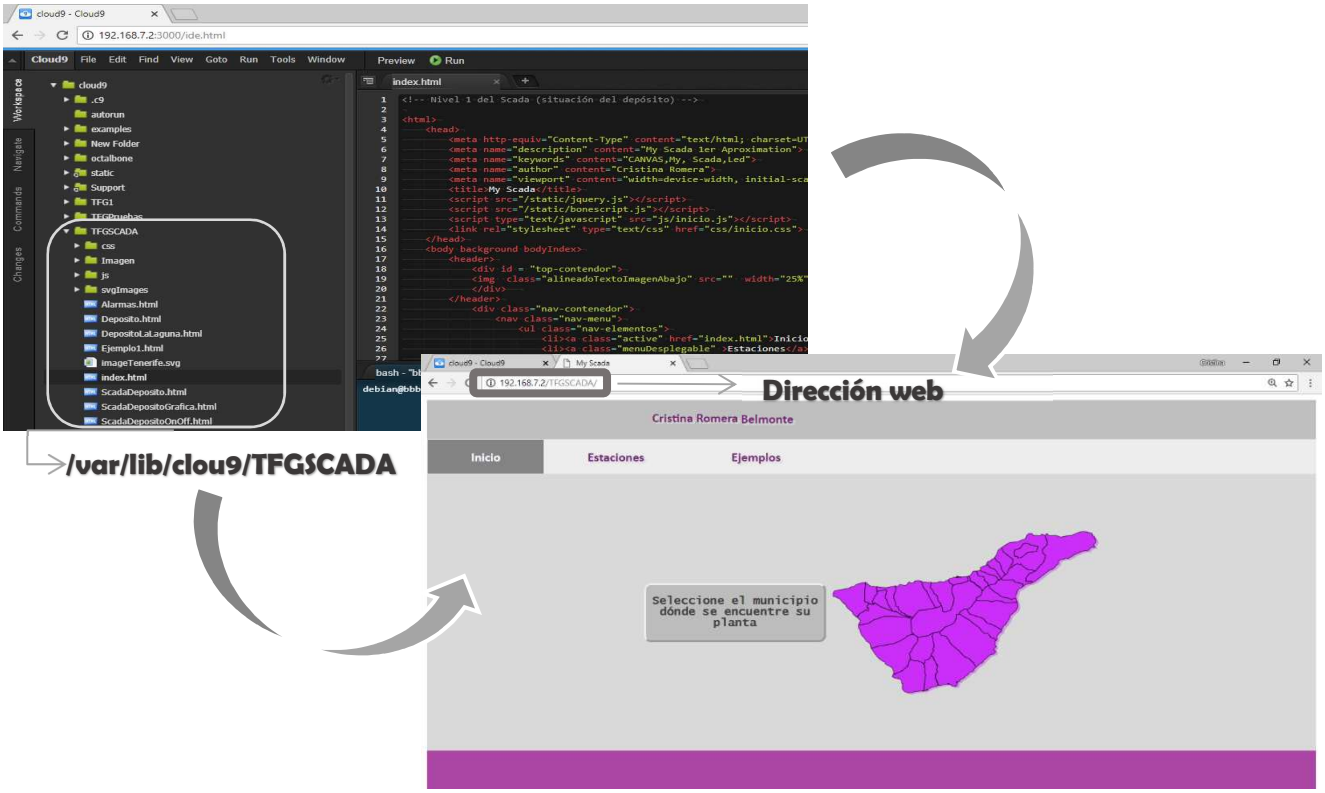


Figura 27: Creación de la carpeta del sitio web y vista en el navegador web[l]

Para aclarar el concepto del envío y recepción de los datos se ha realizado un esquema (ver figura 28), donde se puede observar que BeagleBone realizará tanto el trabajo de control de la planta como el del servidor del sistema SCADA, por consiguiente, adquiere el papel de cliente el ordenador desde el que se realiza la petición para acceder al sistema.

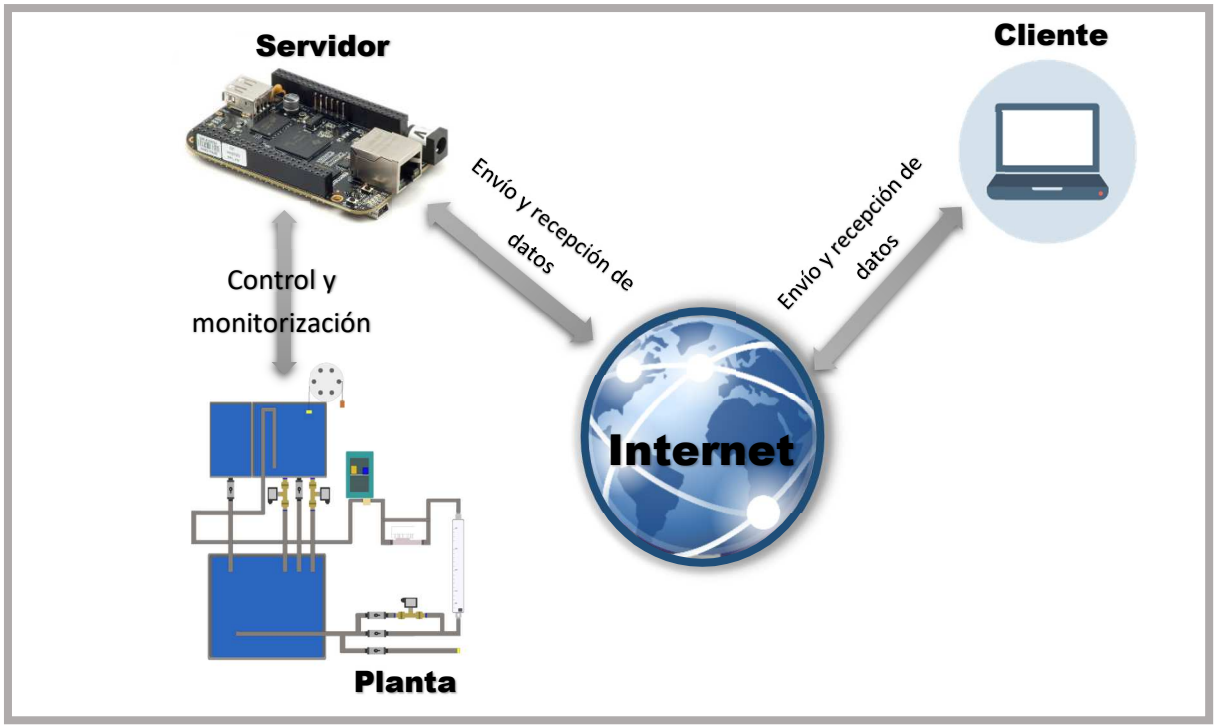


Figura 28: Esquema de comunicación entre el servidor y cliente

3.3 Primeras aproximaciones

Una vez explicado los primeros pasos de la instalación y puesta en marcha del servidor, se puede comenzar con el desarrollo del proyecto, debido a que el fin de éste no es el control de la planta, sino una maquetación de un sistema SCADA con novedosas herramientas, se decide empezar haciendo prototipos más sencillos, que a continuación se irán explicando, como fue su desarrollo y cuáles eran sus objetivos.

3.3.1 Circuito con entrada y salida digital.

De forma bastante usual en las plantas industriales de gran envergadura, aparecen dispositivos con carácter digital, sensores que indican cuándo la variable detectada rebasa un cierto umbral, por ejemplo, los termostatos todo o nada, o cuando la señal cambia de un estado a otro, interruptores o sensores de posición y nivel, esta señal es codificada mediante pulsos o sistemas como binario o BCD. De igual forma existen actuadores todo o nada que pueden ser eléctricos, neumáticos, hidráulicos y térmicos.

Pues bien, dada la importancia de la señal de entrada y salida digital se decide realizar un circuito sencillo que posteriormente permita trasladar su potencial a otro ejemplo más complejo.

El circuito pensado para trabajar con estos términos a pequeña escala está formado por un led (aunque se pueden añadir tantos como se desee) y un interruptor, la idea es poder encender el led ya sea desde el SCADA o con el botón físico.

Primero se monta el siguiente circuito (ver figura 29, esquema generado con la aplicación Fritzing), donde se puede seleccionar cualquiera de los pines de BBB dedicados a fines digitales (ver figura 15). En este caso se selecciona el número 10 del conector P8 para el dispositivo led y el número 16 para el interruptor físico.

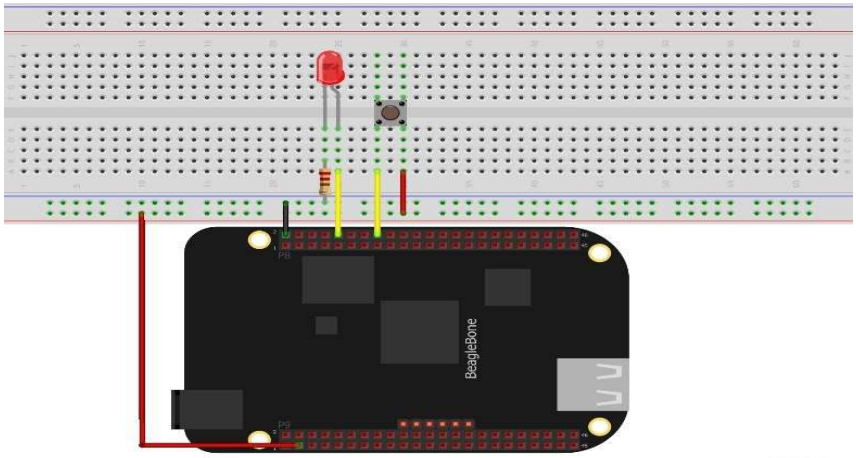


Figura 29: Ejemplo1

Una vez montado el circuito, se crea el HTML básico del SCADA, en este caso lo esencial es poder disponer de dos botones, uno para el encendido y otro para el apagado del led, y un panel de indicadores dónde se visualice el estado del mismo y del botón físico.

El SCADA tiene varias partes diferenciadas y esenciales que se deben de comentar y explicar, se comienza con los elementos incorporados en el archivo HTML.

- **El esquemático**

En este caso el esquema que representa el circuito y estado de la planta se ha creado con la aplicación de escritorio Fritzing, se buscan los elementos necesarios como BBB, un led, el interruptor y la placa protoboard, en el buscador de la aplicación, finalmente se hace el conexionado de los pines. Una vez terminado se exporta como archivo ". svg".

Dado a que se guarda en el formato comentado anteriormente, se puede manipular cada uno de los elementos que forman el esquemático, ya sea el color de una sombra o incluso un elemento completo, para ello se debe de conocer el identificativo `id = ""` correspondiente, por eso se debe de abrir la imagen generada en una aplicación que la muestre en el lenguaje de marcas, en este caso se utiliza la aplicación Inkscape.

Se buscan los identificativos que puedan resultar útiles para la manipulación de la imagen, como se desea cambiar el color del led y del botón, una vez estos cambien de estado, parece lógico localizar dichos id, simplemente seleccionando el elemento y detectando en el editor XML su identificativo, si es necesario se pueden cambiar los nombres que fueron generados automáticamente.

Tabla 12: Identificadores

Elemento	ID
Dibujo encapsulado	"led1"
Dibujo encapsulado	"led2"
Círculo del botón	"boton"

Para incorporar el esquema al archivo HTML se puede hacer de dos formas totalmente diferentes, la primera es poner la imagen en lenguaje de marcas directamente en el archivo, lo que si se tiene un HTML complejo, puede dificultar su edición por lo extenso que pueda quedar, la segunda opción, en este caso más acertada, es incluirlo como un archivo externo entre las etiquetas `<object></object>`, esta etiqueta es soportada por todos los navegadores, en su interior se debe de especificar el *id="svgLed"* , el tipo de imagen (svg+xml) y la ruta relativa donde se haya la imagen guardada, en este caso en la carpeta svgImages.

```
<object id="svgLed" type="image/svg+xml" data="svgImages/BBLed_bb.svg" ></object>
```

Código 7: Incluir una imagen formato SVG

- **Botones**

Los botones en este SCADA cumplen con la función de encender y apagar el led, ya sean los virtuales como el físico, en el HTML se declaran con las etiquetas `<button></button>`, el atributo más importante es el del evento `onclick`, donde se define el evento que se lanza cuando se pulsa el botón, `funcionLed(x)`, tiene un parámetro de entrada, en el caso de querer encender el led éste será uno y cero cuando se pulse el botón de apagado.

```
<button class="boton" value="Boton2" onclick="funcionLed(1)">Led ON</button>
```

Código 8: Etiqueta botón en HTML

- **Indicadores**

Los indicadores siempre deben de aparecer en cualquier SCADA, con ellos se puede saber el valor de proceso, en este caso sólo hay dos, uno y cero.

Para definirlos se utiliza `<p></p>` donde en su interior se incluye esta otra etiqueta `<span>-</span>` que es un contenedor de línea dónde se muestra la magnitud de la variable y que debe de ir identificada, la segunda etiqueta permite que sólo lo que está dentro de ella se pueda modificar, es decir, "Led:" aparecerá siempre y "-" no, ya que variará en función del estado del led, "ON" y "OFF".

```
<p>Led: <span id="LedEstado">-</span></p>
```

Código 9: Indicador del estado del Led

Creado ya el archivo HTML se debe de crear el correspondiente archivo JavaScript, aunque cabe destacar que para comenzar con el segundo se debe de incluir la librería JQuery, que facilita la escritura del código en ciertas ocasiones y la correspondiente Bonescript, y por supuesto el propio archivo con extensión ".js", entre las etiquetas `<head></head>` del HTML.

```
<script src="/static/jquery.js"></script>
<script src="/static/bonescript.js"></script>
<script type="text/javascript" src="js/Ejemplo1.js"></script>
```

Código 10: Librerías

Primero se debe indicar la espera de la carga del archivo (DOM), para ello en el código JavaScript se agrega un evento al objeto SVG, con el método `document.addEventListener(tipo, listener[])`

Donde el `tipo` es una cadena que representa el tipo de evento a escuchar, y `listener` recibe una notificación cuando un evento especificado ocurre. [79]

En el código x se puede ver cómo hay que esperar a la carga del contenido del árbol DOM, una vez finalizado se agrega nuevamente otro evento al contenido del archivo con extensión ".svg" identificado con el `id="svgLed"`, para saber que ya está listo para su manipulación, se añade un mensaje de depuración que indica el fin de la acción, por último las funciones de la librería Bonescript se ejecutan de manera local en BBB, para que el navegador tenga conocimiento de dónde deben ser efectuadas se incluye la función `setTargetAddress()` que indica al mismo en qué dirección se encuentra BBB, una vez la conexión se establece se inicializa la función run, en este caso la función principal que en su interior contiene todo lo relacionado con el SCADA.

```
document.addEventListener("DOMContentLoaded", () => {
      svg = document.getElementById("svgLed"); //Se carga el svg
      console.log('svg:', svg);
      svg.addEventListener('load', () => {
          console.log('Se cargó el SVG');
          setTargetAddress('192.168.7.2', {
              initialized: run
          });

      });
});
```

Código 11: Carga del árbol DOM

Para operar con las E/S, se declaran variables con la palabra reservada `var`, se les da un nombre más cómodo como, Led y Botón, y se inicializan con sus correspondientes pines, así mismo se adjunta el módulo BoneScript utilizando `require`.

```
var b =require('bonescript');
var Boton = 'P8_16';
var LED = 'P9_10';
```

Código 12: Declaración de variables

De igual forma se declaran las variables que están asociadas a los id del archivo SVG, y a los indicadores, de la siguiente forma, se muestra un ejemplo donde se aplica el jQuery con el selector id del mensaje para el estado del led.

```
var LedEstado = $('#LedEstado'); //jQuery
var ledElemnto = svg.contentDocument.getElementById("led1");
var ledElemnto1 = svg.contentDocument.getElementById("led2");
var boton1 = svg.contentDocument.getElementById("boton");
```

Código 13: Declaración de variables

Después de indicar si los pines actúan como entrada (botón) o salida (led), empieza la definición y declaración de las funciones.

Es necesario una función para la lectura del estado del botón, y otra dónde se muestra el valor leído.

```
estadoBoton(); //Llamada a la función
    function estadoBoton() {
        b.digitalRead(Boton, mostrarLecturaBoton); //Se lee el valor y se llama a la
función
    }

    function mostrarLecturaBoton(x) {
        if (!x.err) {
            $('#mensajeBoton').html(x.value); //Se muestra el valor en web
        }
        estadoBoton(); //Llamada función
    }
```

Código 14: función estadoBoton

Por otro lado, se define las funciones asociadas a los botones del SCADA, que a su vez llaman a otras funciones en función del botón que se pulse.

```
    function funcionLed(estado){
            console.log("En led 1: Estado = ", estado); //Mensaje depuración
            if(estado == 1){
                Rojo(); //Llamada a función rojo
            }else{
                Blanco();
            }
      }
```

Código 15: función funcionLed

Como se comentó al principio de este apartado además de los botones del SCADA, el botón físico también tiene una funcionalidad, que se debe de configurar como interrupción, porque de no ser así el botón se encuentra normalmente apagado hasta que se pulse, y no permitiría el cambio de estado del led por medio de los botones del SCADA.

```
//Libreria BoneScript
      b.attachInterrupt(BUTTON, true, b.CHANGE, interrupcionBoton);        function
      interrupcionBoton (x){
            if(x.value == 1){
                boton1.style.fill = "#8A0808"; //Cambio de color del botón
                Rojo();
            }
            else{
                boton1.style.fill = "#483737"; //Color en  hexadecimal
                Blanco();
            }
       }
```

Código 16: Interrupción

Por último las funciones Rojo() y Blanco() son las encargadas de los cambios tanto en el dibujo como en el led y en los indicadores, sólo se incluye la función Rojo() ya que ambas tienen la misma estructura.

```
function Rojo() {
    var LedEstado = document.getElementById("LedEstado");
    var ledElemnto = svg.contentDocument.getElementById("color_path32");
    var ledElemnto1 = svg.contentDocument.getElementById("color_path14");
    ledElemnto.style.fill =  "#FF0000"; //Rojo en hexadecimal
    ledElemnto1.style.fill =  "#FF0000"; //Rojo en hexadecimal
    b.digitalWrite(LED, b.HIGH); //Libreria Bonescript
    LedEstado.innerHTML = "ON"; //Indicaddor
}
```

Código 17: función Rojo()

Se agrega el estilo adecuado para que sea vistoso y agradable a la vista. (ver figura 30)

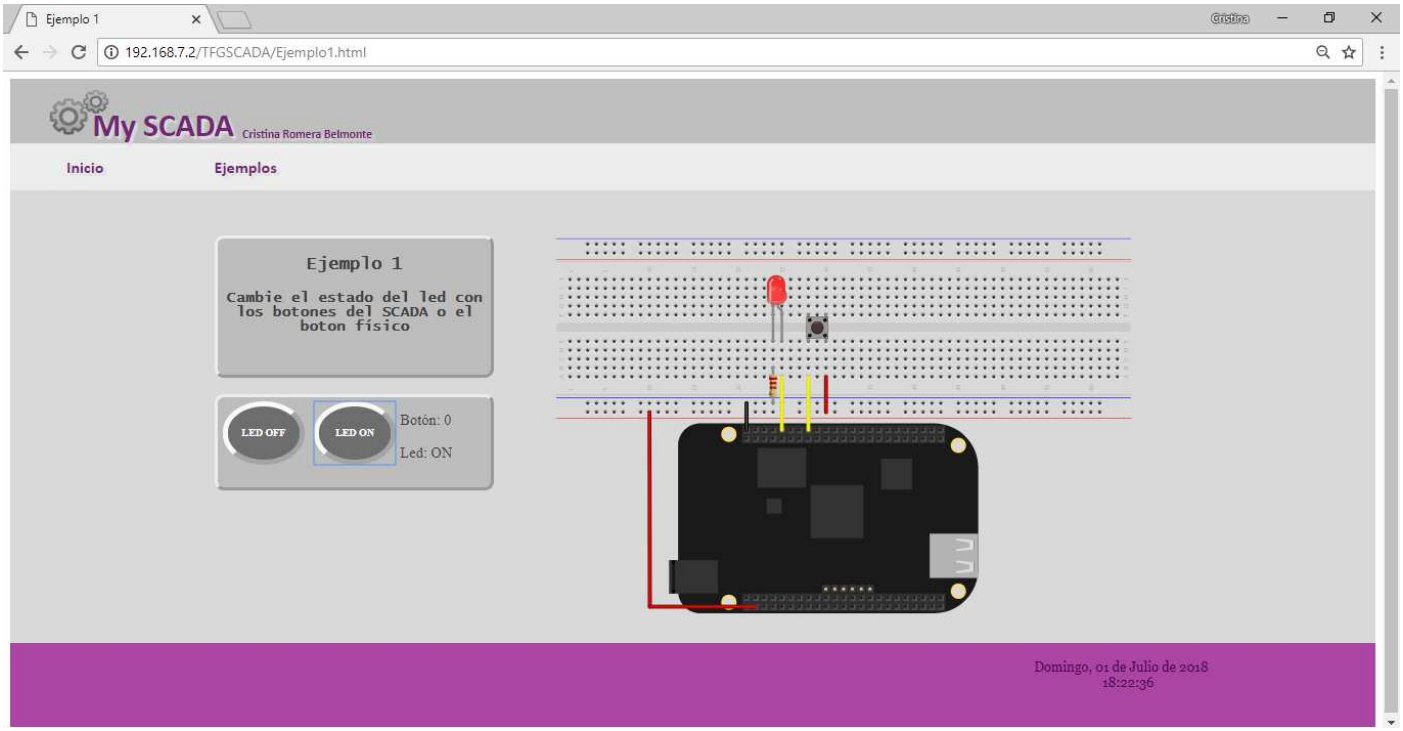


Figura 30: SCADA ejemplo 1

No sólo se puede encender y apagar el led mediante botones, también se puede hacer que éste lo haga de forma intermitente aplicando los métodos adecuados.

- Método **setInterval ()**, llama a una función o evalúa una expresión a intervalos específicos (en milisegundos).
- Método **clearInterval (),** borra un temporizador establecido con el método setInterval ().
- Método **setTimeout ()**, llama a una función o evalúa una expresión después de un número específico de milisegundos.
- Método **clearTimeout ()**, borra un temporizador establecido con el método setTimeout ().

En este caso se emplea un botón para empezar el parpadeo y otro para pararlo.

```
<button class="boton"value="Boton1" onclick = "clearTimeout(Parpadeo)">Led
Off</button>
<button class="boton" value="Boton2" onclick = "Parpadeo =
setInterval(funcionParpadeo, 600)">Parpadeo</button>
```

Código 18: Definición botones en HTML para el parpadeo:

Dentro de la función parpadeo se debe de cambiar tanto el color del led, como el valor del indicador de estado, y el estado real, para ellos se utiliza las siguientes expresiones.

```
ledElemnto1.style.fill = ledElemnto1.style.fill == "white" ? "red" : "white";
LedEstado.innerHTML =  LedEstado.innerHTML == "OFF" ? "ON" : "OFF";
estado = (estado == 0)? 1: 0; //Si el estado = 0 se pone a 1 y si no a 0
```

Código 19: Interior de la función parpadeo

3.3.2 Circuito con entrada analógica y salida digital.

Al igual que las señales digitales, las analógicas también juegan un papel fundamental en la industria, y a diferencia de los sensores digitales (todo o nada) los analógicos pueden detectar cualquier variación de la señal física, teniendo en cuenta la resolución del mismo, por ejemplo, los potenciómetros detectan variaciones de desplazamientos lineales o angulares en valores de resistencia, o los termopares que detectan la temperatura ambiente como variaciones en milivoltios.

Para este ejemplo se realiza un circuito nuevamente a pequeña escala, formado por un potenciómetro y tres leds, que indicará el de resistencia del potenciómetro.

Se tomará el pin número 32 del conector P9 como la tensión de referencia en el divisor de tensión, el pin número 34 como la tierra de referencia, y el pin 33 para la lectura de los valores analógicos.

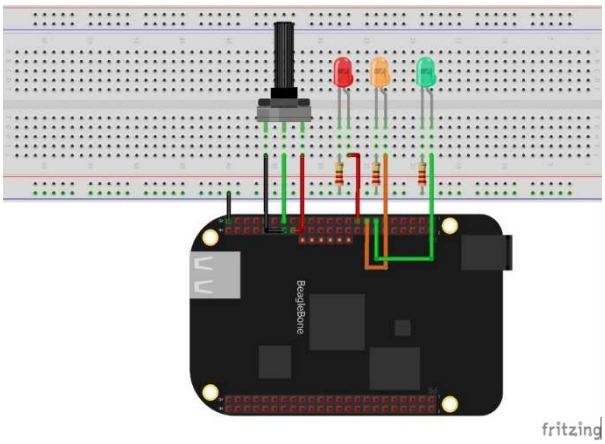


Figura 31: Ejemplo con entrada analógica

En este caso el cuerpo del HTML es más básico que el anterior, sólo se incluye la imagen en formato SVG de forma externa y un indicador para el valor del potenciómetro.

```
<p>Potenciómetro = <span id="estadoPotenciómetro">-</span>
```

Código 20: Indicador para el valor leído del potenciómetro:

Para el archivo ".js" la estructura es la misma, en este caso el potenciómetro actúa como entrada y los leds como salida, para la lectura del valor se hace la siguiente función sencilla, que llama a otra encargada de efectuar los cambios en función del valor leído.

```
function  estadoPotenciomentro() {
        b.analogRead(potenciometro, LecturaPotenciometro); //Funcion Bonescript
    }

function LecturaPotenciometro(x) {
        if (!x.err) {
            //toFixed ajusta el número de decimales
            $('#estadoPotenciometro').html(x.value.toFixed(3));
            var ledRojo1 = svg.contentDocument.getElementById("ledrojo");
            var ledNaranja1 = svg.contentDocument.getElementById("lednan");
            var ledVerde1 = svg.contentDocument.getElementById("ledverde");
            var valor = x.value;
            if(x.value>=0.9){
                b.digitalWrite ( led, b.HIGH);
                b.digitalWrite ( led1, b.LOW);
                b.digitalWrite ( led2, b.LOW);
                ledRojo1.style.fill = "#ffffff";
                ledNaranja1.style.fill = "#ffffff";
                ledVerde1.style.fill = "#00FF00";
            }
            if((x.value<=0.1)){
               ...
            }
             if((x.value<0.9)&&(x.value>0.1)){
               ...
            }
        }
         setTimeout( estadoPotenciomentro, 200); //Lectura cada 200 milisegundos
        }
```

Código 21: Lectura del valor del potenciómetro y animación del archivo SVG

```
}
```

Para este caso se tiene que tener en cuenta el tiempo de muestreo de la señal, pues el potenciómetro puede estar en constante cambio, si se escoge un tiempo demasiado pequeño será imposible conocer el valor de resistencia, si éste se encuentra oscilando entre dos valores con una diferencia entre ellos insignificante, por otro lado si se toma un periodo de muestreo de la señal muy por encima de la tasa de cambio que ésta puede presentar habrán valores de por medio que no se tendrán en cuenta.

Este sería el resultado una vez aplicado un estilo adecuado.

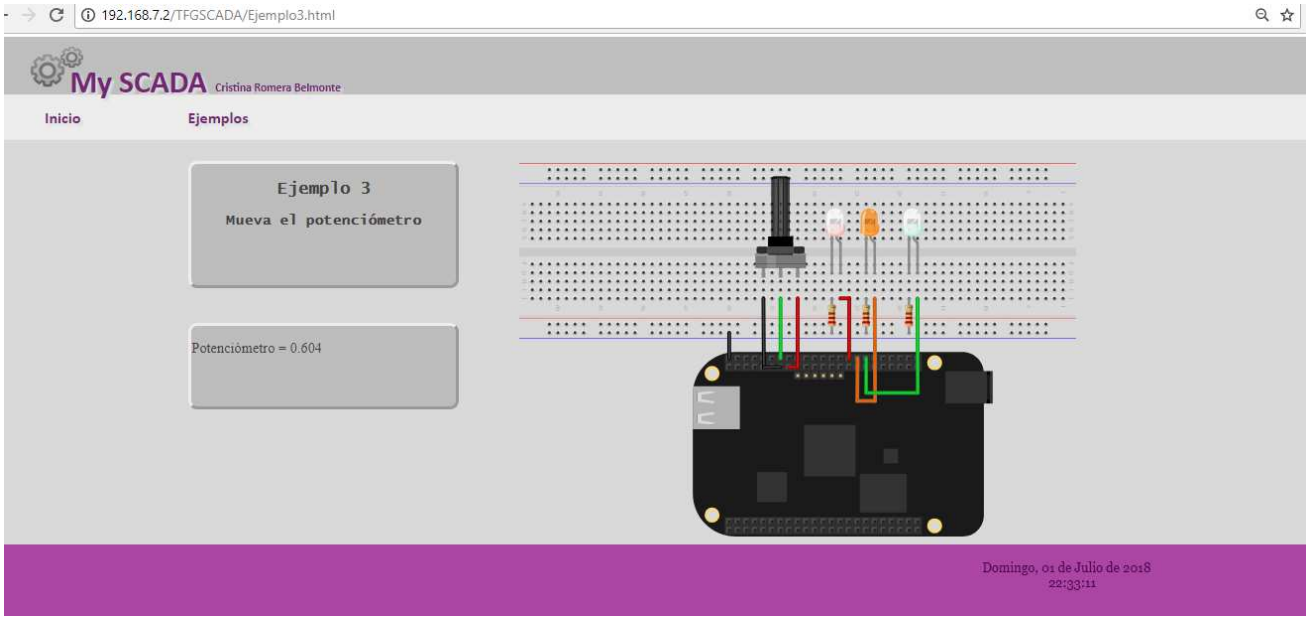


Figura 32: SCADA del ejemplo con entrada analógica

3.4 SCADA para la maqueta de una planta Industrial

En los apartados anteriores se hicieron pruebas de sistemas SCADA con circuitos bastantes sencillos, pero que trabajan igual que los equipos que se pueden encontrar en cualquier proceso industrial, aunque como el lector puede observar, faltaría por efectuar algún sistema donde se desempeñen pruebas con salida de señal analógica, esta prueba se realiza en los siguientes apartados, donde se trabaja con un sistema algo más complejo.

3.4.1 Características de la planta

La planta seleccionada para efectuar un control y monitorización de la misma aplicando la tecnología web, es "*38-001 System Level & Flow control*", de la empresa Feedback, es un panel de aprendizaje educativo, se trata de un sistema en lazo cerrado, cuyas variables de proceso vienen caracterizadas por el nivel del líquido y las tasas de flujo.

El fluido empleado es el agua, el circuito está diseñado para soportar una baja presión del líquido.

Este panel está compuesto por un tanque de proceso de doble compartimiento, conectado a un tanque sumidero mediante válvulas manuales y solenoides. Se emplea una bomba (situada en el tanque inferior) para hacer fluir el agua por el circuito, se hace pasar por un medidor de flujo de área variable y una válvula de control motorizada, si se quisiera saber el nivel del tanque, éste se mide a través de un sensor de nivel con resistencia variable (potenciómetro). [80]

Según su fabricante cuenta con las siguientes características:

- Contiene una selección de indicadores y sensores de nivel y flujo.
- Flujo controlado por válvula de control motorizada lineal.
- Control ON/OFF y PID.
- Autoajuste para el control P, PI y PID.
- Accesorios de empuje modernos.

- ESPIAL software.

Asimismo, se emplea el estándar de 4 a 20 miliamperios tanto como para el control de elementos, como en el transductor de 4 a 20 miliamperios y el visualizador de transmisión, también cuenta con una fuente de corriente y convertidores de corriente a voltaje y un comparador de voltaje con histéresis ajustable, además todo el panel está protegido mediante un interruptor automático de corriente residual. [80]

En la figura 33 se muestran los elementos de la planta enumerados:

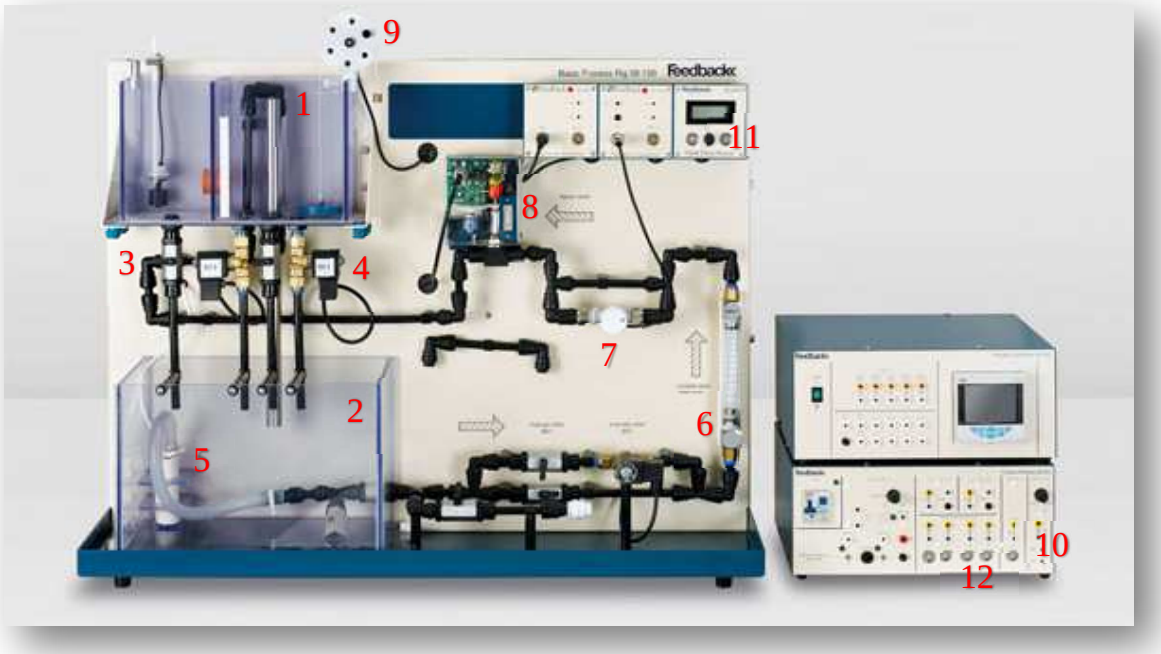


Figura33: 38-001 System Level & Flow control[80]

Tabla 13: Elementos del panel

Nº	Elemento
1	Tanque de proceso con doble compartimento
2	Tanque sumidero
3	Válvula manual
4	Válvula solenoide
5	Bomba
6	Caudalímetro
7	Sensor de flujo
8	Válvula motorizada
9	Sensor de nivel
10	Fuente de corriente
11	Visualizador de transmisión
12	Salida de transmisores

3.4.2 Situación de la planta

La planta con la que se desea comprobar el potencial de las tecnologías comentadas en apartados anteriores se encuentra localizada en el municipio y ciudad de La Laguna concretamente en la Facultad de Física y Matemáticas con la siguiente dirección: Av. Astrofísico Francisco Sánchez, S/N, 38206 San Cristóbal de La Laguna, Santa Cruz de Tenerife. Debido a su uso educativo el panel se encuentra en el laboratorio del Departamento de Ingeniería Informática y de Sistemas. Se especifica la localización de la misma porque se utilizará en el sistema SCADA.

Figura 34: Situación de la planta

3.4.3 Instrumentos de medición y actuación.

En este apartado se hace una breve descripción de aquellos dispositivos con los que se interactúa, ya sea recibiendo o enviándole una señal de control.

Sensor de nivel mediante resistencia variable: el principio básico de funcionamiento de este instrumento de medida es la variación de la resistencia en función de desplazamiento angular realizado, es decir, se efectúa una transformación de la magnitud física a un valor de resistencia eléctrica, donde un valor de 0Ω corresponde a un valor de nivel del cero por ciento y el máximo valor significa que el tanque está lleno.

Bomba Centrífuga: tiene carácter de actuador y su objetivo principal es el de mover una cantidad de agua para que ésta sea absorbida por la tubería. Opera con una señal de 0 a 5 voltios, lo que quiere decir que cuando recibe una señal de 0 voltios, la bomba se encuentra apagada, por el contrario, cuando ésta recibe 5 voltios se enciende. Este dispositivo es el principal del para la realización del control ON-OFF

Válvula solenoide de escape: tiene un carácter de actuador, es un dispositivo encargado del flujo de agua, puede encontrarse en dos estados diferentes normalmente abierta o cerrada, al igual que la bomba opera con una señal de 0 a 5 voltios.

Servo válvula: este actuador se caracteriza por la capacidad que tiene de controlar la cantidad de caudal del fluido, la señal de control, en este caso, de 4 a 20 miliamperios permite el desplazamiento de un vástago, que hace variar el área de paso y con ello el caudal, gracias a la posición obturador con respecto el asiento de la válvula. Este elemento es el principal para realizar un control PID.

3.4.4 Adquisición de la señal de entrada.

Para la realización del sistema SCADA de esta maqueta, el cual es un proceso industrial a pequeña escala, hay que trabajar y efectuar los cambios en la planta según los valores de entrada, señal que describe el nivel de agua del tanque.

El sensor que se encuentra en contacto directo con la magnitud física que se pretende evaluar es el flotador, éste se halla en el tanque flotando y está conectado a un peso mediante una cuerda que se hace pasar a través de una polea, a medida que incrementa el nivel del agua este se traduce a un ángulo de giro del cuerpo de la polea, que gracias al potenciómetro se traduce a un valor de resistencia, esta es la forma en la que se describe el nivel de agua que hay en el interior del tanque. Esta es una señal analógica (nivel de agua) y gracias al transductor se convierte en una señal mucho más manejable (resistencia eléctrica).

En serie al transductor se encuentra un transmisor cuyo objetivo es la amplificación de dicha señal y convertirla en una señal estandarizada, en este caso de 4 a 20 miliamperios, estándar habitual en la industria.

Ahora bien, esta señal debe de ser tratada como una señal digital, para ello se debe de realizar la conversión analógica-digital. En este proyecto, el dispositivo encargado de la conversión es el microrordenador BeagleBone, como se comentó en apartados anteriores soporta a su entrada señales con un máximo de voltaje de 1.8 voltios (E/S Digital), por lo que se necesita una adaptación, para hacer equivaler el máximo de 20 miliamperios a un valor de 1.8 voltios.

La solución que se ha optado es por la directa aplicación de la ley de Ohm.

$$V = I \cdot R \rightarrow 1.8(voltios) = 20(mA) \cdot R(\Omega) \rightarrow R = \frac{1.8}{0.02} = 90\Omega$$

Ecuación 7: Ley de Ohm

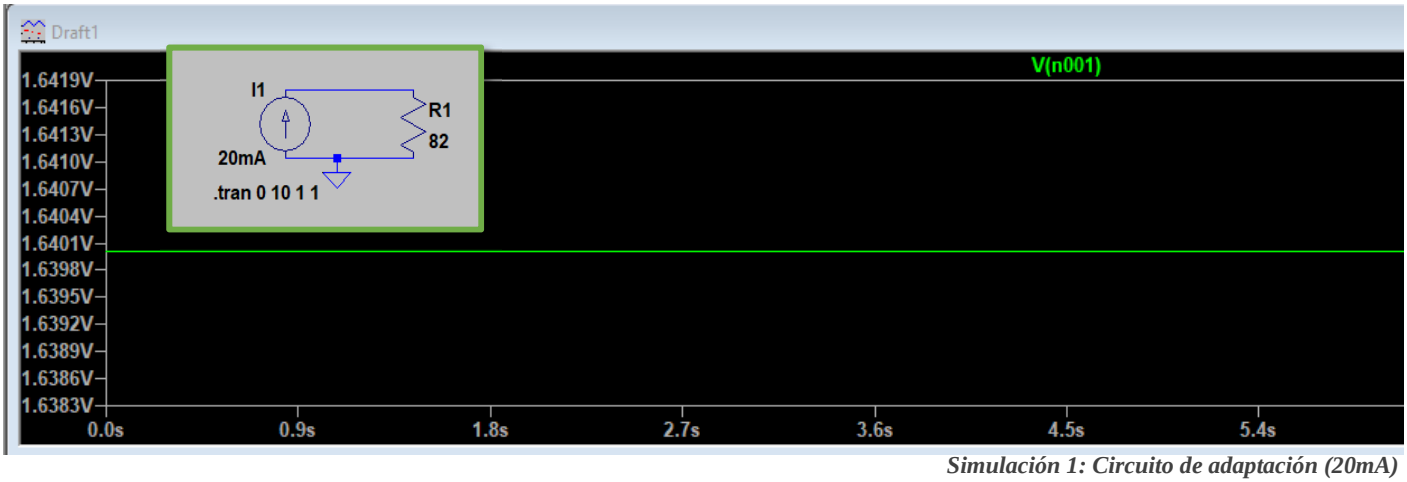
En la práctica esta resistencia no tiene un valor comercializable por lo tanto se debe de escoger una que si lo sea y que tenga un valor próximo o inferior a ella para que no supere el máximo voltaje admisible.

El El valor seleccionado es de 82 ohmios, y que por tanto el máximo voltaje generado es de 1.64 voltios inferior al admisible (1.8 voltios).

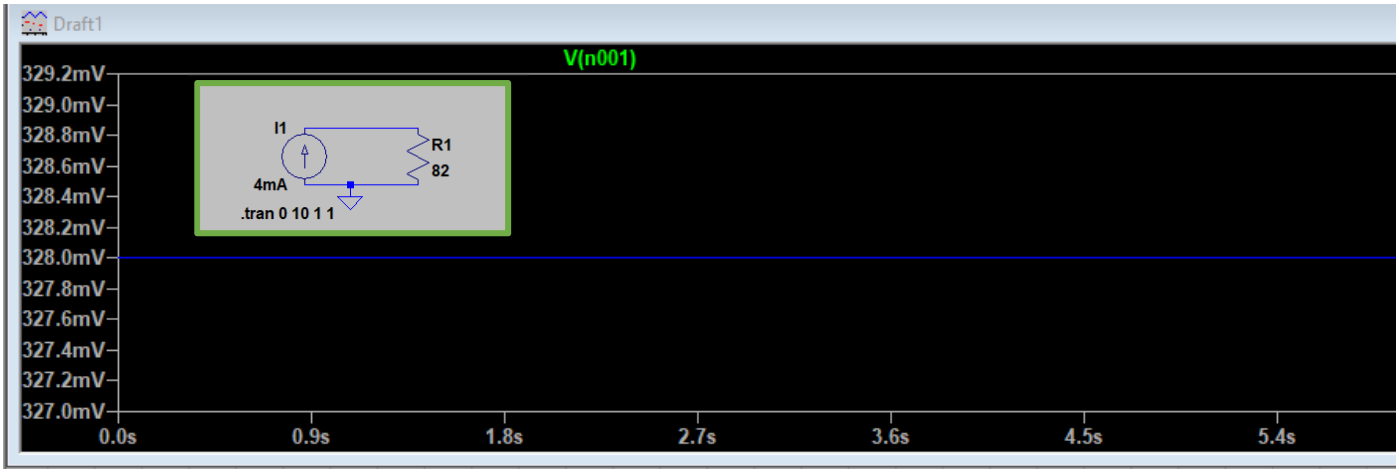
Tabla 14: Resistencias de valores comerciales [81]

x 1	x 10	x 100	x 1.000 (K)	x 10.000 (10K)	x 100.000 (100K)	x 1.000.000 (M)
1 Ω	10 Ω	100 Ω	1 KΩ	10 KΩ	100 KΩ	1 M Ω
1,2 Ω	12 Ω	120 Ω	1K2 Ω	12 KΩ	120 KΩ	1M2 Ω
1,5 Ω	15 Ω	150 Ω	1K5 Ω	15 KΩ	150 KΩ	1M5 Ω
1,8 Ω	18 Ω	180 Ω	1K8 Ω	18 KΩ	180 KΩ	1M8 Ω
2,2 Ω	22 Ω	220 Ω	2K2 Ω	22 KΩ	220 KΩ	2M2 Ω
2,7 Ω	27 Ω	270 Ω	2K7 Ω	27 KΩ	270 KΩ	2M7 Ω
3,3 Ω	33 Ω	330 Ω	3K3 Ω	33 KΩ	330 KΩ	3M3 Ω
3,9 Ω	39 Ω	390 Ω	3K9 Ω	39 KΩ	390 KΩ	3M9 Ω
4,7 Ω	47 Ω	470 Ω	4K7 Ω	47 KΩ	470 KΩ	4M7 Ω
5,1 Ω	51 Ω	510 Ω	5K1 Ω	51 KΩ	510 KΩ	5M1 Ω
5,6 Ω	56 Ω	560 Ω	5K6 Ω	56 KΩ	560 KΩ	5M6 Ω
6,8 Ω	68 Ω	680 Ω	6K8 Ω	68 KΩ	680 KΩ	6M8 Ω
8,2 Ω	82 Ω	820 Ω	8K2 Ω	82 KΩ	820 KΩ	8M2 Ω
						10M Ω

Con el software LTspice se realiza una simulación de este sencillo circuito.



Simulación 1: Circuito de adaptación (20mA)



Simulación 2: Circuito de adaptación (4mA)

Tabla 15: Resultados de la simulación

Valor de corriente (mA)	Valor de voltaje (V)
20	1.6401
4	0.328

Con la señal ya adaptada se puede realizar el proceso de conversión realizado por el dispositivo BeagleBone, aunque como se manifestó en el apartado de "Adquisición de señales", la frecuencia de muestreo es clave para una conversión acertada.

Para ello se tiene que estudiar el ancho de banda de la señal y toma un valor de frecuencia de muestreo que sea el doble de la banda de la señal, o que su frecuencia máxima, en este enunciado se fundamenta el teorema de muestreo de Nyquist-Shannon.

Otro concepto importante para conversión analógica-digital es la resolución de la conversión, BeagleBone que es capaz de realizar 100,000 muestras por segundo y cuenta con una resolución de 12 bits lo que equivale a 4095 niveles, estas características son suficientes para un correcto muestreo de la señal puesto que la planta tiene una dinámica lenta.

Aunque BBB nos da la posibilidad de seleccionar un valor de muestreo bastante pequeño, éste no resulta útil para el sistema que se plantea, pues al tratarse de un sistema con una dinámica muy lenta no es necesaria tanta precisión, al contrario, porque podría producir un pequeño rizado en la señal. Por ello se toma un valor de muestreo que puede oscilar entre 1 o 3 segundos.

Con todo lo comentado hasta ahora es posible procesar la información y tratarla para el control de la planta.

3.4.5. Acondicionamiento de la señal de salida.

Los actuadores mencionados con anterioridad, bomba y válvula solenoide, trabajan con señales de control digitales, dónde 0 voltios equivale a un estado de apagado o cerrado, y 5 voltios un estado de encendido y abierto.

Estas señales de salidas son fácilmente generables con BeagleBone, simplemente en la lógica implementada se indica cual debe de ser el estado de los componentes con la función digitalWrite().

Si se recuerda la tensión de salida máxima a la que trabajan los pines de BBB, esta es de 3.3 voltios en continua, sin embargo, la bomba y la válvula solenoide trabajan con 5 voltios, por lo que se necesita un circuito adaptador de nivel que permita convertir la señal lógica de un nivel a otro, este ya se encuentra en el interior del panel educativo, por eso no es necesario realizar ningún montaje.

A diferencia de estos equipos la válvula motorizada o servo válvula, utiliza como señal de control, el estándar de 4 a 20 miliamperios, la gran dificultad se debe a que para este caso la placa no es capaz de generar esta señal ni tampoco una señal analógica.

A pesar de esto, BeagleBone ofrece la posibilidad de emplear las salidas de señales PWM comentadas en el apartado (Salidas PWM), esta opción hubiera sido válida si la señal de control hubiera requerido ser de 0 a 5 voltios, porque con la incorporación de un filtro pasa bajo (resistencia y condensador en serie), se puede lograr que a la salida se observe una señal que aparentemente parezca ser analógica (la realidad es que todavía no lo es) con un poco de rizado que dependerá de los valores del filtro, este filtro tiene como propósito permitir el paso de aquellas frecuencias que estén por debajo de la frecuencia de corte que debe de ser elegida teniendo en cuenta la frecuencia del *"timer"* que genera la señal PWM, para que esta no se refleje en el resultado.

El esquema eléctrico se muestra en la figura 35:

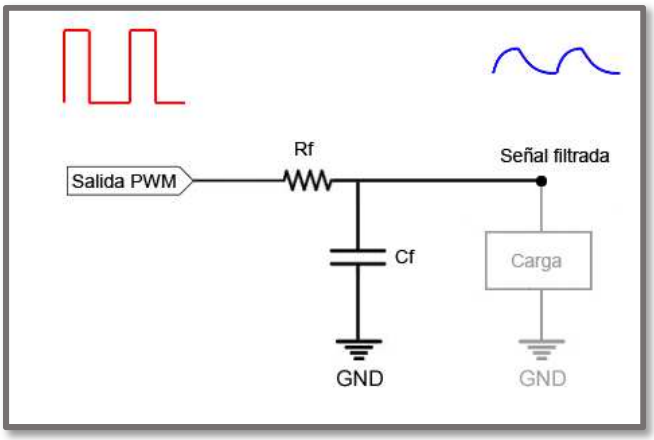
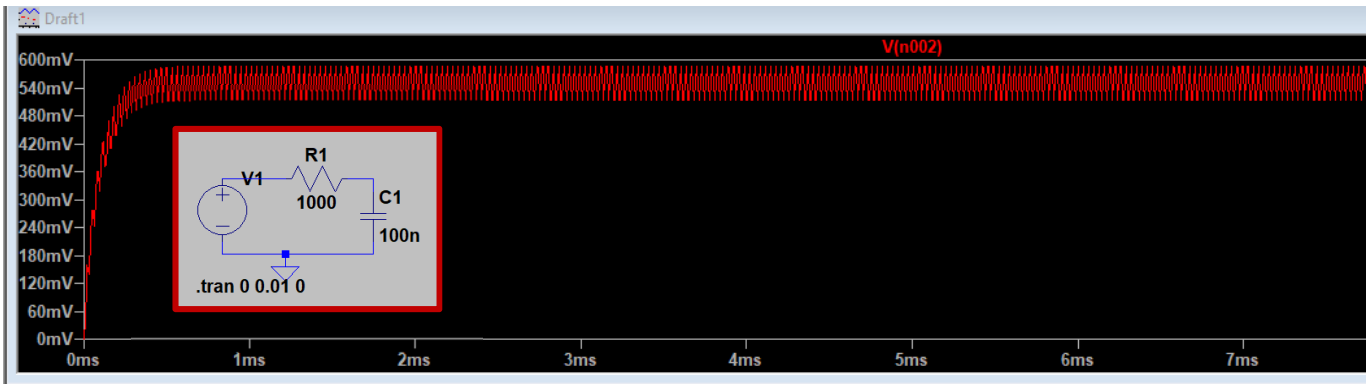


Figura 35: Conversión de una señal PWM mediante un filtro paso bajo [82]

También se adjunta la simulación correspondiente, donde se aprecia el rizado de la señal de salida. Sin embargo, esta opción no es válida para el problema que se propone.

La solución que finalmente se ha llevado a cabo, puede describirse como fácilmente implantable y de bajo costo, cuyos componentes principales son un conversor digital-analógico, simplificado como DAC y un amplificador de corriente.



Simulación 3: Filtro paso bajo

3.4.5.1 Justificación de la solución

La solución anterior no era la adecuada porque lo que se necesita a la salida es una señal de corriente que varíe en función del valor de la señal de control, esta señal de control será generada por BBB y ya que la salida PWM no es del todo adecuada, se busca otra alternativa donde las salidas de los pines digitales tengan cabida. Aquí entra en juego la conversión decimal a binaria, donde se convierte el valor de la salida, resultado dado en decimal a binario, aquellos pines que se encuentren en estado alto proporcionarán una tensión de salida igual a 3.3 voltios en continua, ahora hay que saber gestionarlos para obtener el resultado que se busca, hasta aquí se parece bastante con la solución anterior, pero hay un elemento fundamental el DAC (siglas en inglés), conversor digital-analógico.

El DAC ofrece un nivel analógico de tensión de salida a partir de la palabra digital proporcionada por BBB, esta tensión de salida viene dada como una suma ponderada de componentes asociados a cada dígito, cada una con un peso que en potencia de dos.

Gracias a la amplia variedad de pines que ofrece la placa de desarrollo ha sido posible la implementación de esta opción, pues el DAC cuenta con una resolución de 8 bits que equivalen a 255 niveles, de no utilizarlos todos se estaría perdiendo esta característica, se ha optado por el DAC808 debido a su disponibilidad y sobre todo a la salida que genera, un máximo de 10 voltios y 2 miliamperios, cuando todos sus bits están a uno, por tanto parece lógico tomar la corriente y amplificarla para obtener los 20 miliamperios elementales para el control.

3.4.5.2 Esquema electrónico

El esquema que se detalla a continuación, como se observa está formado por un DAC, con la referencia DAC0808LCN, cuenta con 16 pines, de los cuales 8 están destinados a la conversión, por lo que se cuenta con 8 bits de resolución, es el encargado de realizar la conversión digital a una corriente que como máximo puede ser de 2 miliamperios.

La conversión se rige por la ecuación:

$$V_0 = V_{ref} \cdot \left(\frac{A_1}{2} + \frac{A_2}{4} + \frac{A_3}{8} + \frac{A_4}{16} + \frac{A_5}{32} + \frac{A_6}{64} + \frac{A_7}{128} + \frac{A_8}{256}\right)$$

Ecuación 8: Salida en voltaje del conversor

$$I_0 = \frac{V_{ref}}{R_{14}} \cdot \left(\frac{A_1}{2} + \frac{A_2}{4} + \frac{A_3}{8} + \frac{A_4}{16} + \frac{A_5}{32} + \frac{A_6}{64} + \frac{A_7}{128} + \frac{A_8}{256}\right)$$

Ecuación 8: Salida en corriente del conversor

Donde V_o es la tensión de salida, I_o la corriente de salida respectiva, A_n los bits cuyo estado puede ser igual a uno si se encuentra en *High* o cero si están en *Low*, donde el bit más significativo es el A_1 que corresponde al pin número 5 y el menos significativo A_8, pin número 12.

El paso entre valores de voltaje discretos, también conocido como LSB, tiene un valor de 39,0625 milivoltios que corresponden a un valor en corriente de 7,8125 µA.

Cálculo de LSB

$$LSB = \frac{V_{ref}}{2^n} \rightarrow Donde\ V_{ref} = 10\ voltios\ y\ n = número\ de\ bits = 8$$

$$LSB = \frac{10}{2^8} = \frac{10}{256} = 0{,}0390625\ V$$

$$LSB = \frac{10}{R_{14} \cdot 2^8} = \frac{10}{5000 \cdot 256} = 7{,}8125 \cdot 10^{-6}\ A$$

Tabla 16: Conversión de ciertos valores decimales

Decimal	A1	A2	A3	A4	A5	A6	A7	A8	Valor teórico de corriente (mA)
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1	0,0078125
2	0	0	0	0	0	0	1	0	0,015625
3	0	0	0	0	0	0	1	1	0,0234375
4	0	0	0	0	0	1	0	0	0,03125
…..	…..	…..	…..	…..	…..	…..	…..	…..	…..
51	0	0	1	1	0	0	1	1	0,3984375
52	0	0	1	1	0	1	0	0	0,40625
53	0	0	1	1	0	1	0	1	0,4140625
…..	…..	…..	…..	…..	…..	…..	…..	…..	…..
253	1	1	1	1	1	1	0	1	1,9765625
254	1	1	1	1	1	1	1	0	1,984375
255	1	1	1	1	1	1	1	1	2

El siguiente componente esencial, es un integrado encargado de la amplificación de la corriente, es un amplificador, con referencia LM741CN, que gracias a la disposición de las resistencias en el lazo se puede conseguir una ganancia en corriente de 10 veces el valor de entrada.

El amplificador tiene el siguiente desarrollo matemático:

$$V_- = V_+ = V_o \quad [1]$$

$$I_e = I_2 \quad [2]$$

$$I_e = I_2 = \frac{V_- - V_A}{R_2}\ [2] \text{ sabiendo que } V_- = V_o\ [1] \text{ entonces } I_e = \frac{V_o - V_A}{R_2}$$

Despejando se obtiene que $V_A = V_o - I_e \cdot R_2$ [3]

Por otro lado $I_0 = I_1 = \frac{V_A - V_o}{R_1} = \frac{V_o - I_e \cdot R_2\ [3] - V_o}{R_1} = -I_e \frac{R_2}{R_1}$ [4]

Finalmente, la ganancia en corriente $\frac{I_0}{I_e} = -\frac{R_2}{R_1}$

Ecuación 8: Ganancia del amplificador

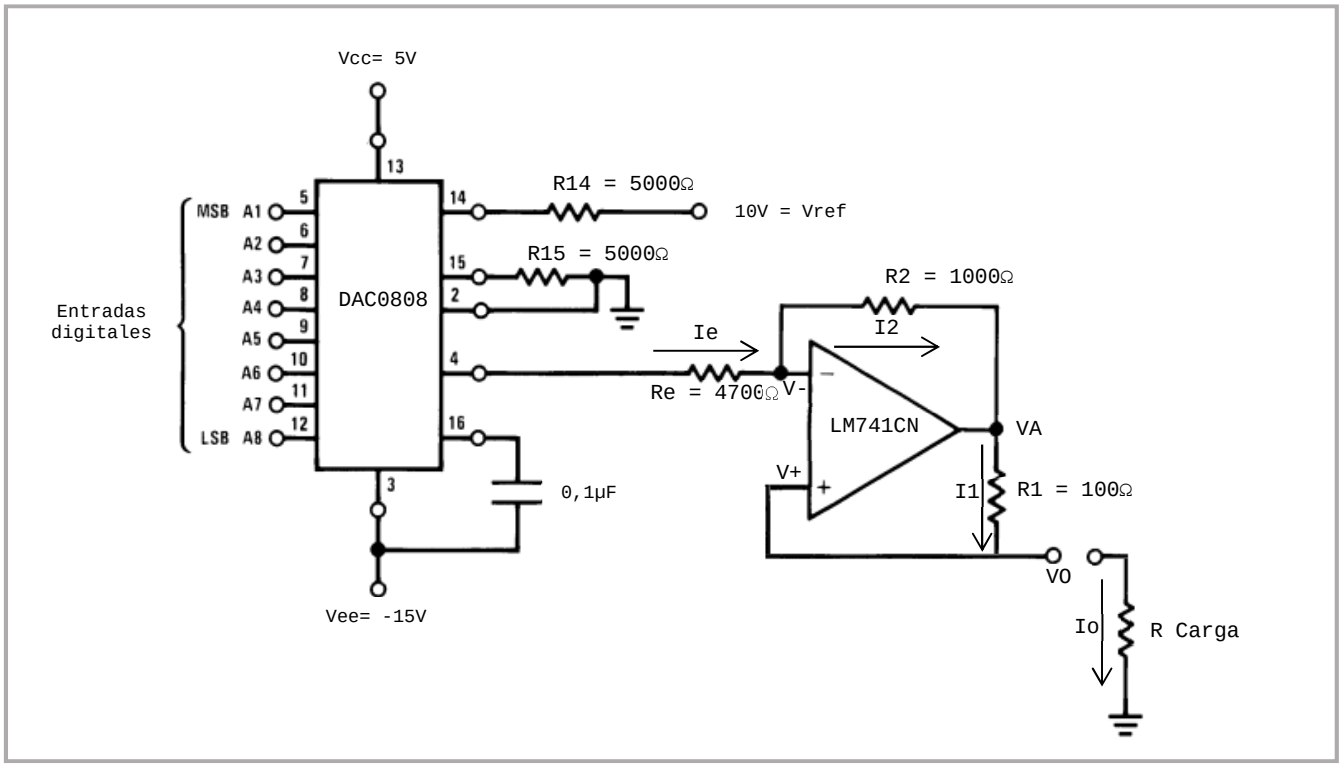
Debido a que se necesita unos 20 miliamperios cuando el DAC tiene como salida 2 miliamperios, la ganancia del amplificador debe de ser de 10, por tanto, se escoge como valores para las resistencias de 1000Ω y 100Ω para R_2 y R_1 respectivamente.

Tabla 17: Valores teóricos de amplificación

Decimal	Corriente (mA)	Corriente amplificada (mA)
0	0	0
1	0,0078125	0,078125
2	0,015625	0,15625
3	0,0234375	0,234375
4	0,03125	0,3125
…..	…..	…..
51	0,3984375	3,984375
52	0,40625	4,0625
53	0,4140625	4,140625
…..	…..	…..
253	1,9765625	19,765625
254	1,984375	19,84375
255	2	20

Si se presta atención a la configuración y a la tabla de valores se aprecia que se alcanza un valor de 4 miliamperios con en el nivel 51 (código en binario: 00110011) esto quiere decir que los niveles por debajo de este valor no son necesarios o lo que es lo mismo, no se van a emplear, por lo un 20% del DAC queda en desuso, pero se asume este inconveniente.

A continuación, se adjunta el esquema empleado, la información adicional de los integrados se adjuntan en el apartado de Anexos.



Esquema Electrónico 1

El voltaje de alimentación positiva en el DAC es de 5 voltios y por consiguiente el voltaje de tensión negativa de -15 voltios, el condensador que se encuentra entre las patas 3 y 16 del convertidor está destinado a filtrar las posibles variaciones de la fuente de tensión, los voltajes de referencia son 10 (pata número 14) y 0 voltios (pata número 15), esto da como resultado que valores de tensión por encima de 2 voltios se asume que se encuentra en estado alto lo que indica un uno a la entrada del pin, y valor por debajo de 0.8 un estado bajo. Por último, a través de la pata número 4 fluye la corriente hacia el amplificador operacional.

La resistencia de entrada del AO teóricamente no afecta al resultado, pero se coloca como protección, se aprovechan las alimentaciones que genera la fuente de alimentación utilizada para el DAC, para teniendo una alimentación positiva de 15 voltios y una negativa de -15 voltios.

La figura 36 muestra un esquema del montaje hecho con la aplicación Fritzing.

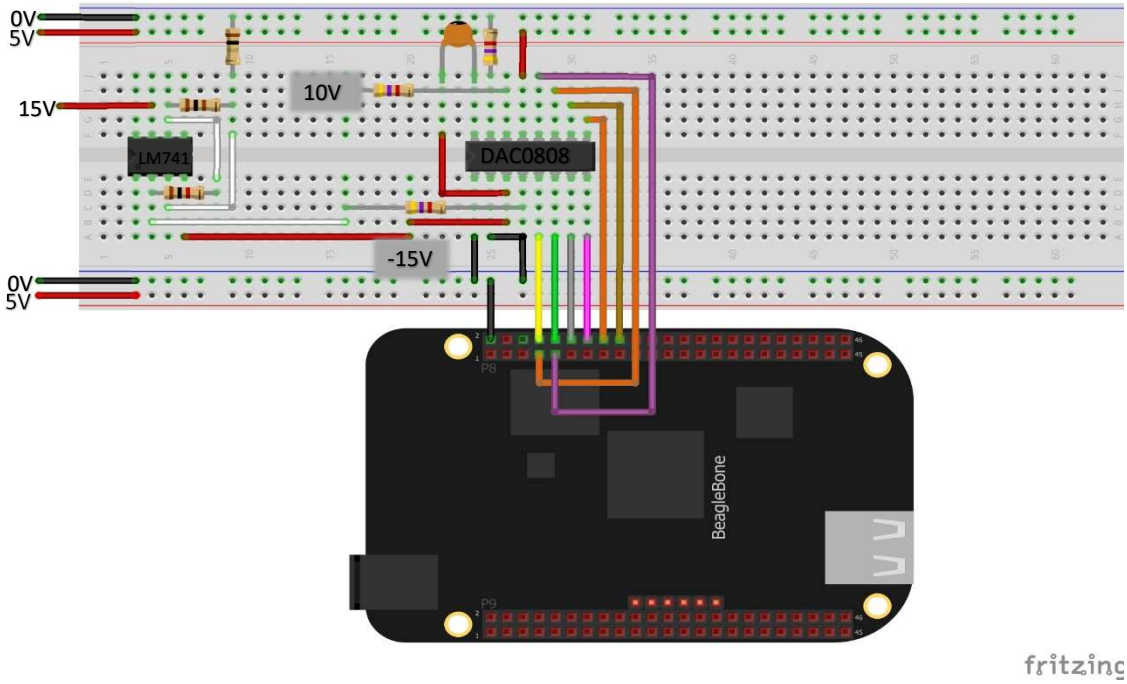


Figura 36: Esquema para realizar un convertidor con entrada digital con salida en corriente

Haciendo pruebas sobre el mismo y haciendo un barrido de los resultados, se observa como de forma general el valor real de corriente es ligeramente superior al esperado, este hecho se atribuye a la tensión de offset del AO, ya que no es ideal y cuando se tiene un valor digital de 0 y que por tanto a la salida debe haber 0 miliamperios esto no sucede.

3.4.6 Arquitectura de control.

Puesto que un sistema SCADA lleva intrínseco en su definición la opción de control de la planta que se monitoriza, se decide crear tanto un sistema de visualización como de control. Para ello se deben de definir tanto el sistema al que se desea incorporar el control como el propio control.

3.4.6.1 Modelado de la planta industrial.

Se comienza con la definición donde se tratará el tanque superior como un sistema de control accionado por válvulas y una bomba en corriente continua, dónde el objetivo es mantener el nivel de agua a pesar de las variaciones de la demanda.

El sistema está formado por una entrada de caudal, este caudal puede variarse gracias a la servo válvula, y una salida de caudal constante accionado por una válvula solenoide, asimismo se puede considerar como perturbación la salida de flujo a través de la válvula con accionamiento manual.

Debido a que sólo cuenta con una entra o variable manipulada, caudal de entrada de agua, y una salida o variable controlada, nivel de agua en el tanque, se considera un sistema SISO.

Para aplicar el control del nivel de agua, se necesita una modelación del sistema de forma matemática, para ello se busca una relación entre la salida y la entrada del sistema.[83]

La relación entre la salida y la entrada se denomina función de transferencia y viene dada por la ecuación 9, en la transformada de Laplace, donde C es la superficie del tanque y R la resistencia que opone el paso de fluido en la válvula este sistema se caracteriza por ser de primer orden

$$\frac{h(s)}{q_i(s)} = \frac{R}{R \cdot C \cdot s + 1}$$

Ecuación 9: Función de transferencia de primer orden

3.4.6.2 Componentes de un sistema de control

Cualquier sistema puede describirse normalmente como el esquema de la figura 37:

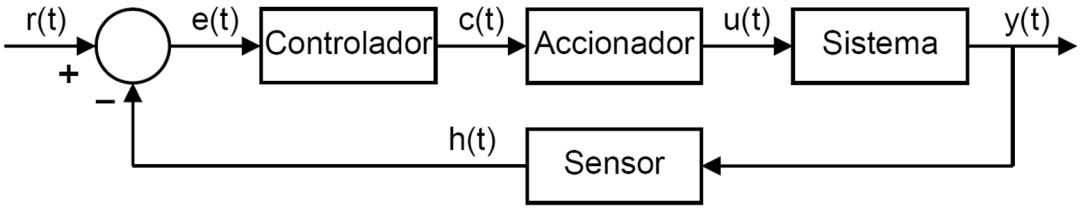


Figura 37: Típico lazo de control de un sistema [84]

Para concretar aún más, se debe de tener en cuenta el tipo de control que se va a realizar, ya que hay determinados elementos que difieren en función del control, por ello se van a explicar cada uno de los controles por separado y como se ha llevado a cabo su implementación.

Antes de comenzar se recuerda el significado de determinados parámetros:

- El valor de consigna o *setpoint* abreviado como SP, es el valor que se desea alcanzar, en este caso un valor de nivel de agua.
- El valor de proceso o PV, indica el valor real de la variable que se desea controlar.

3.4.6.3 Control On-Off

Este es el primer control que se ha puesto en práctica en la planta sobre la bomba que se encuentra en el depósito inferior, y se caracteriza por contar con dos estados, como su nombre indica, el de encendido y el de apagado.

El comportamiento es bastante lógico, pues cuando el valor de la variable controlable (nivel de agua) es menor que el nivel de referencia o de consigna definido con anterioridad, se envía una señal de activación (se enciende la bomba), sin embargo, cuando ésta se encuentra por encima del *setpoint* se desactiva la señal de control (se apaga la bomba).

Ahora bien, si no se toma ninguna otra especificación de diseño, este control puede dañar el dispositivo que controla, pues si la señal varía con facilidad o bien hay ruido en la medida, la frecuencia con la que se cambia el estado de la bomba es elevada.

3.4.6.4 Control On-Off con histéresis

Con el control On-Off se pueden llegar a obtener resultados medianamente aceptables, pero como se planteó, se producen cambios muy rápidos en la salida del controlador, este control se puede mejorar si se añade una histéresis. [86]

La histéresis define el tiempo de espera antes de que se produzca el cambio de estado, además para un mismo nivel existe la posibilidad de estar en los dos estados.

La explicación teórica es la siguiente, según la figura X se puede apreciar como a medida que la variable controlada disminuye su valor (PV>SP) y se va acercando al valor de referencia, el error va disminuyendo hasta alcanzar un valor nulo, momento en el que se produce el cambio de signo en el error, sin embargo, hasta que no se alcance el límite superior (+δ) no se conmuta al nivel alto. Este mismo proceso se lleva al otro caso, cuando la variable aumenta su valor (PV<SP), efectuando el cambio en cuando se llega al límite inferior.

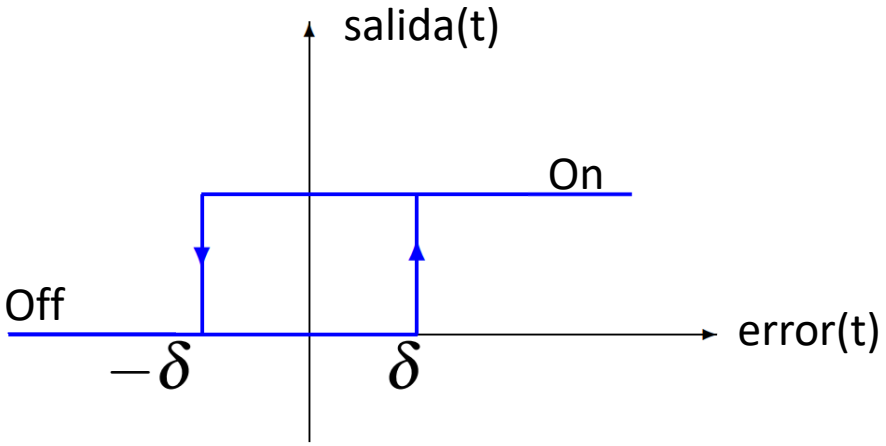


Figura38: Control On-Off con histéresis [85]

Esta solución permite que no se produzcan cambios de forma instantánea, pero se manifiesta una oscilación en la salida que aumenta directamente con un aumento en el valor de los límites y es inversamente proporcional a la tasa de cambio en la variable.

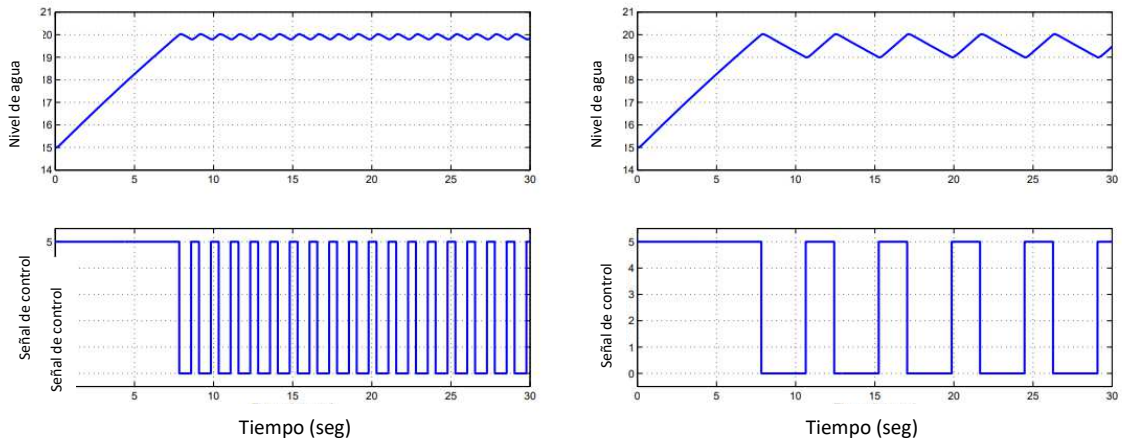


Figura 39: Comparación entre un δ inferior (izquierda) con uno mayor (derecha)[85]

Una vez aclarado los conceptos se incluye el lazo de control empleado, caracterizado con los elementos del sistema real.

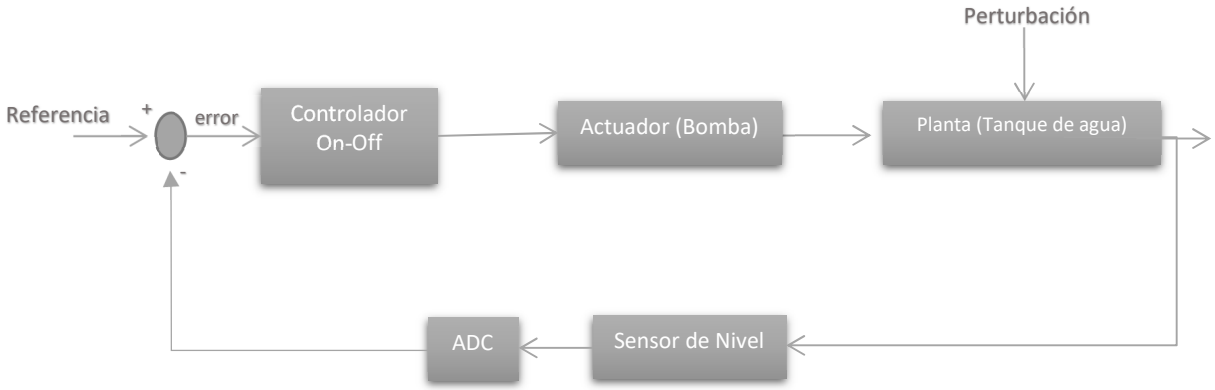
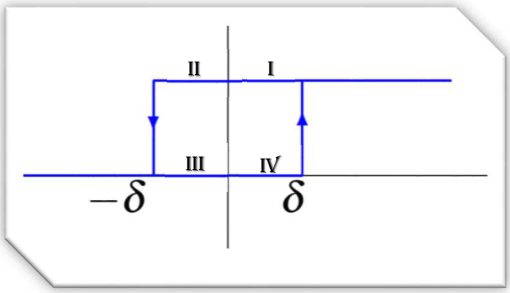


Figura 40: Lazo de control con controlador ON-OFF con histéresis

3.4.6.5 Implementación del Control On-Off con histéresis

A continuación, se incorpora el código empleado.

```
function ControlOnOff(x) {
        const SP = 0.500; //Setpoint, en SCADA se puede variar
        const Delta = 0.200; //Valor de hist, en SCADA se puede variar
        const Uminf = -Delta; //Límite inferior
        const Umsup = +Delta; //Límite superior
        var bool = 0; //variable para distinguir entre los cuadrantes

        if (!x.err) {

           //Si no hay error en la medida

            var Pvalue = x.value;
            var error = SP - Pvalue;
            console.log(Delta);

            //Cuadrante I
            if ((error<Umsup) && (error>0)&&(bool==1)){
             b.digitalWrite(Bomba, b.HIGH);
            }

            //Cuadrante II
            else if  ((error > Uminf)&& (error<0)&&(bool==1)){
                b.digitalWrite(Bomba, b.HIGH);
            }

            //Cuadrante III
            else if  ((error > Uminf)&& (error<0)&&(bool=0)){
                b.digitalWrite(Bomba, b.LOW);
            }

            //Cuadrante IV
            if ((error < Umsup) && (error>0)&&(bool==0)){
                b.digitalWrite(Bomba, b.LOW);
            }

            //Fuera del límite inferior
            else if  (error <= Uminf){
                b.digitalWrite(Bomba, b.LOW);
                bool=0;
            }

            //Fuera del límite superior
            else if (error >= Umsup){
                b.digitalWrite(Bomba, b.HIGH);
                bool=1;
            }
        }
        }
```



Código 22: Control On-Off con histéresis

3.4.6.6 Control PID

Para este proyecto se opta por incluir un control meramente proporcional-integral (PI), pues la dinámica de la planta no es demasiado rápida y es por ello que no se requiere de la acción derivativa.

Ahora bien, una vez conocida la ecuación faltaría calcular los parámetros, K_p, T_i, que se ajusten a la planta y que permitan su control. Existen varios métodos de sintonización teórica del controlador, como por ejemplo el Método de Ziegler-Nichols, pero la realidad es que, aunque es de utilidad, lo mejor es ajustar los parámetros mediante pruebas sobre la planta, de igual forma el sistema SCADA permite al usuario variar estos parámetros

Por último, destacar que estas técnicas no se pueden aplicar de forma directa cuando se trabaja con componentes digitales, pues nuevamente se debe de hablar de un muestreo, ya que ningún dispositivo es capaz de tomar muestras con un tiempo tan pequeño que casi sea cero, por lo tanto, el controlador, en este caso PI, debe de igual forma ajustarse a las circunstancias, se habla por tanto de un controlador PI discreto.

Existen varias técnicas para discretizar el controlador que son:

- Aproximación rectangular hacia delante.
- Aproximación rectangular hacia atrás.
- Aproximación bilineal (trapezoidal o Tustin).

De igual forma existen tres técnicas para convertir un sistema analógico en discreto:

- Mantenedor de orden zero (ZOH).
- Correspondencia polos-ceros.
- Transformación bilineal.

Por último, se incluye el lazo de control empleado, caracterizado con los elementos del sistema real.

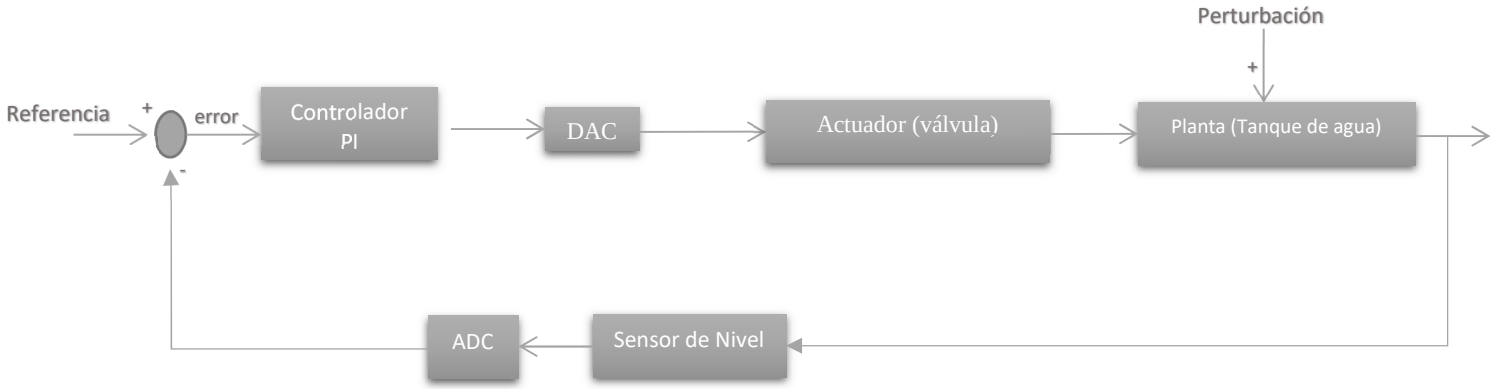


Figura 41: Lazo de control con controlador PI

3.4.6.7 Implementación del Control PI

A continuación, se adjunta el código empleado para el diseño del control PID, como se observa está definida la acción derivativa por si fuera necesaria su implementación, se aplica la discretización mediante la aproximación rectangular [44]

$$u(n) = K_p \cdot e(n) + K_i T \sum_i e_i + K_d \cdot \frac{[e(n) - e(n-1)]}{T}$$

Ecuación 10: PID discreto

Donde n es el instante de tiempo en el que se evalúa la expresión y T el periodo de muestreo.

Para la implementación hay que tener en cuenta que la acción sobre el sistema está limitada físicamente, pues la válvula no se puede abrir y cerrar cuanto se quiera, hay unos topes. Por ello es tarea del diseñador acotar la salida del controlador a los límites físicos del actuador.

De igual forma hay que incorporar una estrategia para proporcionar una desaturación del término integral, para ello se especifica al controlador que cuando la salida se encuentre saturada, tanto en el límite superior como en el inferior el valor del término integrador permanezca constante.

En este caso se emplea una clase para definir el PID, en su método constructor aparecen los atributos propio del controlador, y el método `salida()` calcula el valor de la salida del controlador.

Si se quiere hacer uso del mismo simplemente se debe de instanciar un objeto

```
var pid1 = new PID;
```

Código 23: Nuevo objeto de la clase PID

```
class PID {
        constructor (){

            this.SP = 80; //Nivel del agua al que s quiere llegar (se puede cambiar)
            this.salida = undefined;

            //Valores que se pueden cambiar
            this.KP = 5; //Ganancia proporcional
            this.KI = 0.7; // Ganancia integral
            this.KD = 0; // Ganancia Derivativa

            this.sumError = 0; //acumulador de error
            this.preError = 0; // Error en el instante anterior
            this.tiempoMuestreo = 2; // la funcion de muestreo se hace cada 2 seg

            //Saturación de la salida
            this.min = 0;
            this.max = 100;
        }
        salidaPID (PV){

            let terProporcional; //Término Proporcional
            let terIntegral; //Término Integral
            let terDerivativo = 0; //Término derivativo

            //Cálculo del error
            let error = this.SP - PV;
            console.log ("error",error);

            //Cálculo termino proporcional
            terProporcional = error * this.KP;
            console.log ("proporcional", terProporcional);


            //Cálculo término integral
            terIntegral = (this.sumError) * this.KI *this.tiempoMuestreo;
            console.log ("integral",terIntegral);

            //Cálculo término derivativo
            terDerivativo = (((error - this.preError)/this.tiempoMuestreo))*this.KD;
            console.log ("derivativa",terDerivativo);
            console.log ("error previo",this.preError);

            //Salida
            let salida = terProporcional + terIntegral + terDerivativo;
            console.log ("salida1",salida);


             //Saturación de la salida
            if (salida < this.min ) {
                salida = this.min;
                this.sumError = this.sumError;
                console.log ("sumerror", this.sumError);
            } else if ( salida > this.max) {
                salida = this.max;
                this.sumError = this.sumError;
                console.log ("sumerror", this.sumError);
            }else{
                this.sumError += error;
                console.log ("sumerror", this.sumError);
            }
            pid1.preError = error;
            console.log ("salida",salida);
            return salida;
        }
    }
```

Código 24: Control PID

3.4 Solución propuesta del sistema SCADA.

La solución propuesta ha sido diseñada con la intención de proveer a la planta un sistema de visualización, lo más parecido que se puede encontrar en el mercado industrial, teniendo en cuenta el entorno educativo y aprovechando las mismas tecnologías que se han empleado a lo largo de todo este trabajo, las tecnologías web. El resultado dispone de elementos como panel de alarmas, indicadores, gráficos del sistema y sinóptico del proceso. Aunque se conocen las limitaciones de esta solución se cree en su amplio potencial y en las mejoras que se le puedan añadir posteriormente.

3.4.1 Arquitectura.

La solución propuesta cuenta con tres niveles de jerarquía y su esquema se detalla en la figura 42

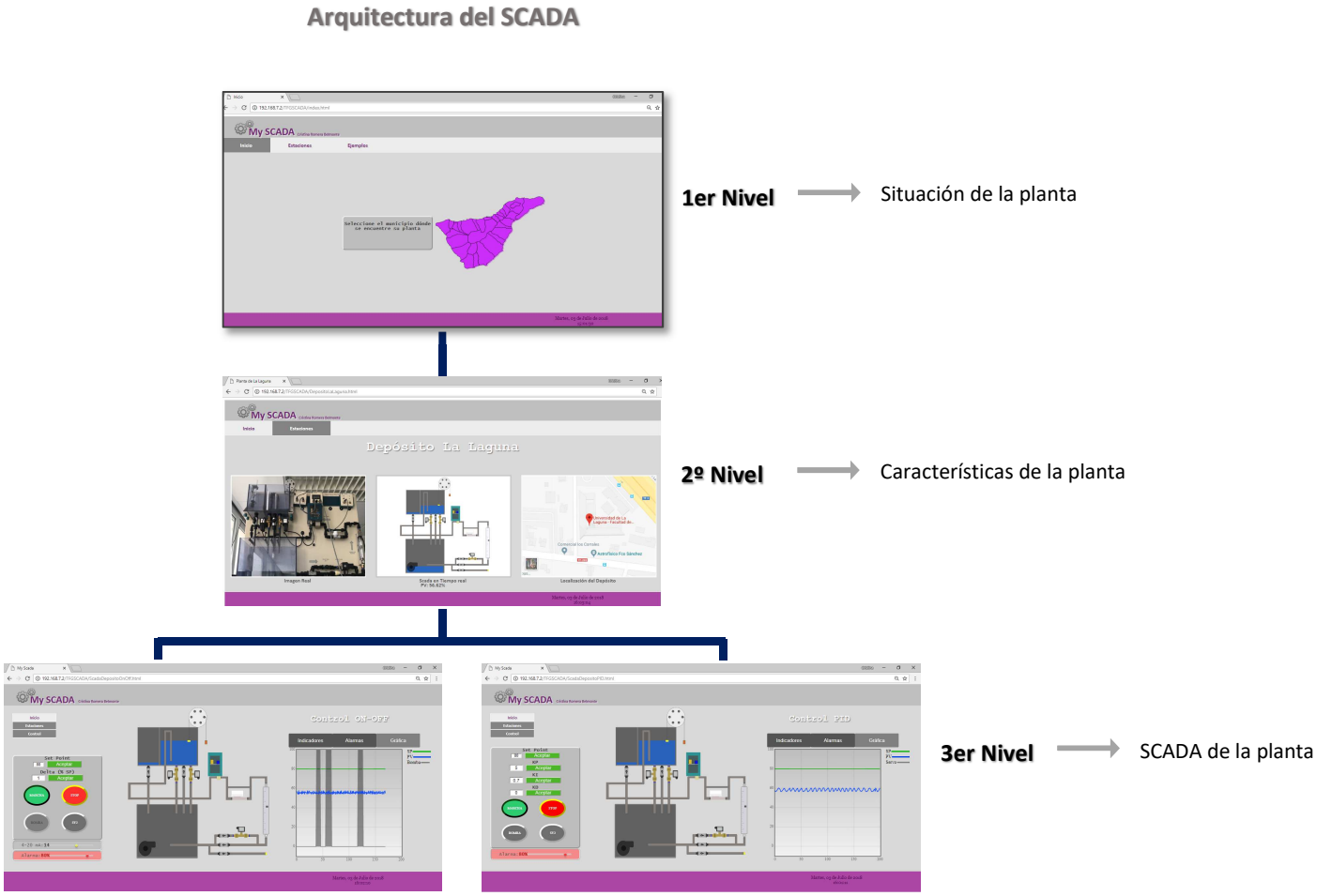


Figura 42: Arquitectura del SCADA

Como se observa en la figura 42, el proyecto mira hacia un futuro donde se incorporen más plantas.

En los siguientes apartados se explica cada uno de los diseños y su elaboración mediante los lenguajes web respectivos.

3.4.2 Nivel 1: Situación de la planta

Al sistema SCADA se accede a través del navegador de uso preferente, buscando la dirección IP de BeagleBone y añadiendo la ruta donde se aloja el sistema, "192.168.2/TFGSCADA", se toma esta IP porque en este caso se utilizó un ordenador con el sistema operativo Windows, no es necesario incluir el nombre del archivo HTML ya que de forma predeterminada se abre aquel con el nombre "*index.html*", es por ello la pantalla de inicio tiene este nombre.

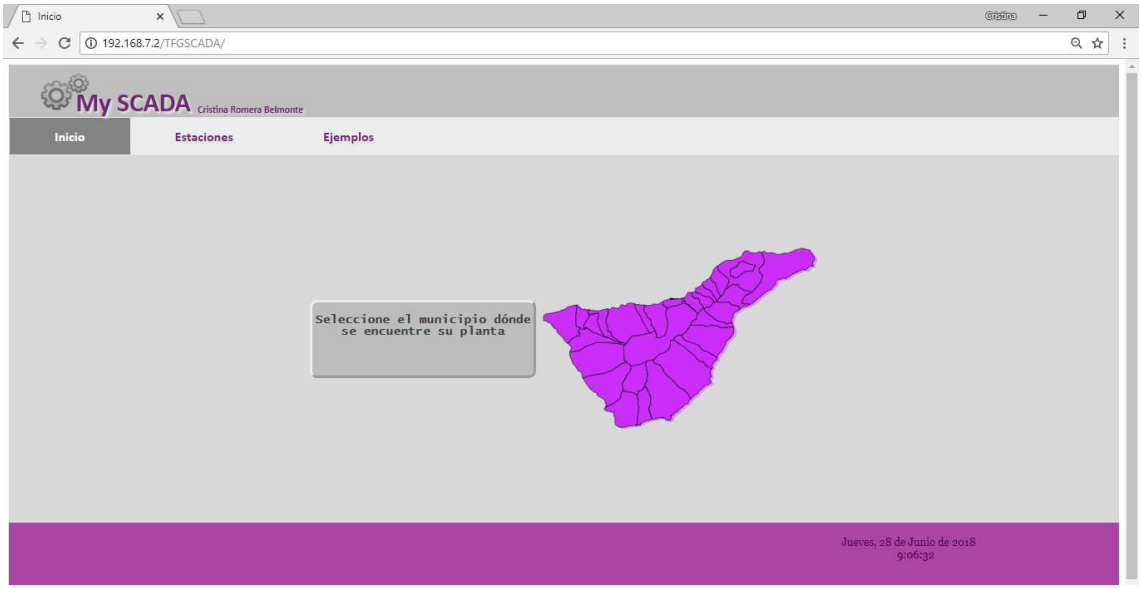


Figura43: Nivel 1 Inicio

Esta pantalla puede dividirse visualmente en cuatro secciones.

- La sección `header`, o parte superior, donde se puede ver el nombre del proyecto y el logo, "*MY SCADA*", y el nombre de la autora y diseñadora del proyecto.
- La siguiente sección, `nav`, se dedica al panel de navegación, donde por defecto se marca como página activa de "Inicio", pues es donde actualmente se encuentra el usuario. Hay un apartado denominado "Estaciones" que, si se pasa el selector por encima, se despliega un submenú con los diferentes municipios de la isla que disponen de una planta, ahora mismo sólo se dispone de la planta situada en la Laguna. Por último, si nuevamente se pasa el ratón sobre "Ejemplos" se despliega un submenú con ejemplos prácticos sencillos para familiarizarse con el entorno.
- La siguiente sección o div, se le ha denominado el "cuerpo" de la pantalla, es bastante sencillo, se deben de seguir las instrucciones, y seleccionar el municipio donde se encuentra la planta que se desea controlar, es una alternativa al menú, visualmente se ayuda al usuario a la selección añadiendo una etiqueta con el nombre del municipio.
- La parte inferior de esta pantalla `footer`, se denomina pie de página, añade un contraste de color a la pantalla. Tiene como función incluir la hora y fecha a tiempo real, dato que siempre debe de estar presente.

Algunas de estas secciones están presenten en el resto de pantalla para dar un aire de unificación al entorno.

3.4.2.1 Código

Cada una de las pantallas incorpora internamente tres lenguajes web que se deben de diferenciar claramente, HTML, JavaScript y CSS.

Se comienza explicando, a modo de ejemplo, el código que estructura esta pantalla, es decir, el código HTML, en los siguientes apartados se hará hincapié en etiquetas del lenguaje que sean específicas de esa pantalla y nuevas funcionalidades.

Primeramente, se debe de agregar toda la información que puede ser de interés al navegador, como puede ser el título de la página, `<title></title>`, el nombre del autor, palabras claves, descripción de la página web. Seguidamente se incorporan los enlaces a archivos externos, como las librerías usadas en ejemplos anteriores, la hoja de estilos y el código ".js"

```
<!-- Nivel 1 del Scada (situación del depósito) -->
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
       <title>My Scada</title>
        <meta name="description" content="My Scada Nivel 1">
        <meta name="keywords" content="My,Scada, Level, Control">
        <meta name="author" content="Cristina Romera">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <script src="/static/jquery.js"></script>
        <script src="/static/bonescript.js"></script>
        <script type="text/javascript" src="js/inicio.js"></script>
        <link rel="stylesheet" type="text/css" href="css/inicio.css">
    </head>
```

Código 25: Inicio HTML <head></head>

Para realizar las diferentes secciones de la página web, el contenido de cada sección se encierra dentro de las etiquetas `<div> </div>`, cada uno de ellos tiene una identificación, según su acometido, en este ejemplo no hay demasiado texto, pero sí estructuras que merecen ser explicadas.

La barra de navegación se realiza a partir de las etiquetas , especifica que en su interior se haya una lista de elementos no ordenada, el contenido se incluye gracias a las etiquetas y por tratarse de enlaces a otras páginas se requiere de la incorporación de enlaces a través de <a>, el atributo href = "" contiene la URL al cual apunta el enlace.

```
<div class="nav-contenedor">
    <nav class="nav-menu">
        <ul class="nav-elementos">
            <li><a class="active" href="index.html">Inicio</a></li>
            <li><a class="menuDesplegable" >Estaciones</a>
                <ul class ="sub-menu">
                    <li><a href="DepositoLaLaguna.html">La Laguna</a></li>
                </ul>
            </li>
                <li><a class="menuDesplegable" >Ejemplos</a>
                <ul class ="sub-menu">
                    <li><a href="Ejemplo1.html">Salida y entrada Digital 1</a>
                   </li>

                </ul>
            </li>
        </ul>
    </nav>
</div>
```

Código 26: Menú de navegación

Nuevamente la ilustración de la isla de Tenerife tiene un formato ".svg" por ello se adhiere a través de <object></object>, como un objeto incrustado, de esta forma se minimiza el código HTML, además esta etiqueta no restringe a la imagen y permite realizar operaciones con ella.

En cuanto al código JavaScript, para este caso en particular, es bastante sencillo, sólo cuenta con las indicaciones que se le da al navegador para que encuentre la dirección IP de BBB, al código 11 además aparecen las funciones para la hora y la fecha, en tiempo real, para la hora se utiliza una función ya establecida en este lenguaje, Date(). Para la fecha también se toman métodos que ayudan a realizar dicha composición, el código x muestra la función empleada.

```
function fecha(){

    var fecha=new Date();
    var año=fecha.getYear();
    var dia=fecha.getDay(); //Numero de la semana Lunes = 1, Martes = 2...
    var mes=fecha.getMonth();
    var daym=fecha.getDate();
    if (año < 6000){
      año+=1900; //Efecto 2000 (si no saldría 118)
    }
    if (daym<10){
      daym="0"+daym ; //Se añade un cero del delante del numero, estética 07 en vez 7.
    }
    var vectorDia=new
Array("Domingo,","Lunes,","Martes,","Miércoles,","Jueves,","Viernes,","Sábado,");
    var vectorMes=new
Array("Enero","Febrero","Marzo","Abril","Mayo","Junio","Julio","Agosto","Septiembre","Octu
bre","Noviembre","Diciembre")
    document.getElementById('dia').innerHTML=vectorDia[dia]+" "+daym+" de
"+vectorMes[mes]+" de "+año;

}
```

Código 27 Fecha JavaScript

Por último, se requiere agregar el estilo para crear un entorno más vistoso e intuitivo, el código CSS ayudará en esto. Todos los elementos quedan identificados con un `id` que es individual, pero si varios elementos exigen una misma apariencia, entonces se agrega el atributo `class`. Hay infinidad de propiedades que pueden ser agregadas, aquellas que no pueden faltar son: el ancho (`width`), el alto (`height`), su disposición (`display`) y su posición (`position`).

A modo de ejemplo se incluye un fragmento de código en CSS empleado en la composición de la parte superior (dónde se encuentra el logo y nombre del proyecto y autora) (ver código 28).

```css
 /*Bloque superior*/
#top-contendor {
    width:100%;
    height:10%;
    position: relative;
    display: block;
    background-color: #bfbfbf;
    color: #732874;
    font:bold 18px "Calibri";
    text-shadow: 3px 3px 2px #e0dbe0;
}
#svgLogo{
    width:10%;
    height:100%;
    position:absolute;
    left:0%;
}
 #Nombre{
    position:absolute;
    left:6%;
    top:-9%;
    font-size:35px;
}
  #Autora{
    position:absolute;
    left:17%;
    top:55%;
    font-size:13px;
}
```

Código 28 : Estructura CSS parte superior

3.4.3 Nivel 2: Características de la planta

Una vez seleccionada la planta, se entra en el segundo nivel del SCADA, este nivel ayuda al usuario a situar la planta.

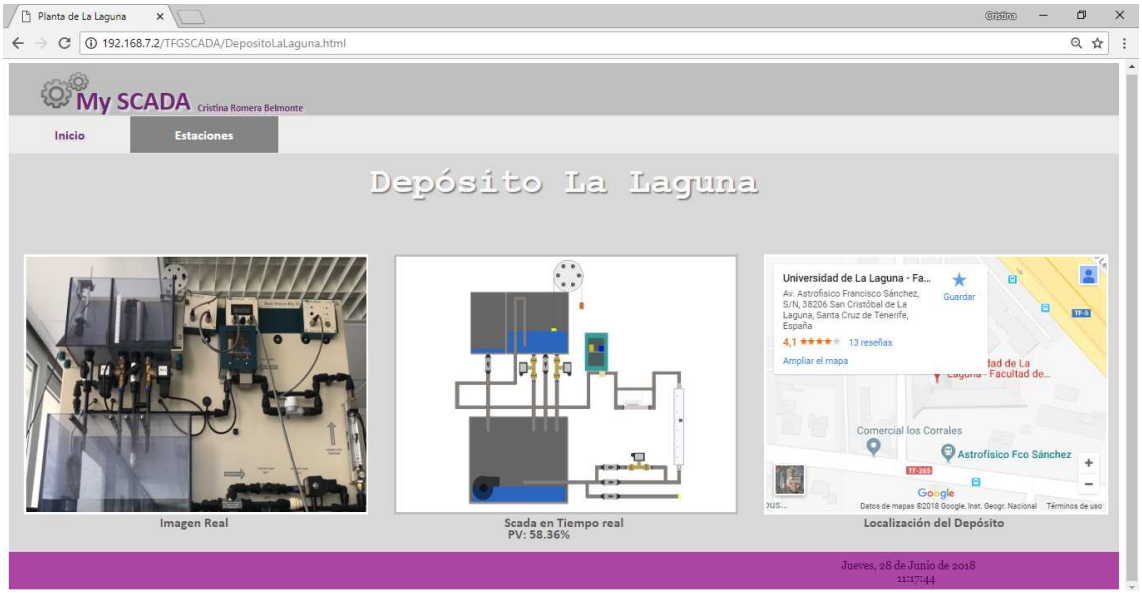


Figura 44: Nivel 2 del SCADA

Cuenta con las mismas secciones que el nivel anterior, y con la misma funcionalidad, sin embargo, es el cuerpo que ahora en su interior cuenta con un sinóptico de la planta a tiempo real, y con la información más relevante de ésta, el nivel de agua del tanque superior. Asimismo, a la derecha del sinóptico se añade la ubicación del depósito para asegurar en todo momento al usuario donde se encuentra la planta, y se agrega una foto real del depósito.

En cuanto al código, si se comienza de nuevo con el HTML, las etiquetas empleadas son las mismas, se crean `<div></div>` por cada uno de los elementos rectangulares, imagen real, sinóptico y localización. Los pies de fotos son nuevamente etiquetas de enlace `<a></a>`, salvo la imagen.

El pie del esquema de la planta llevaría al usuario al siguiente nivel, y el del mapa abre una nueva pestaña, en el navegador, con la geolocalización de la misma.

Para incrustar la imagen si incorpora el siguiente código:

```
<div class= "elemento">
      <img id="EstadoReal" class="ElementosCuerpo" src="Imagen/ImagenReal.jpeg"
      alt="Estado de la planta" title ="Imagen Real"/>
      <div class= "PieFoto">Imagen Real</div>
</div>
```

Código 29: Cómo insertar una imagen

El atributo que especifica dónde se encuentra guardada la imagen es `src`, seguidamente, `alt`, agrega información, para los casos en los que el usuario por cualquier motivo no pueda ver dicha imagen, por último, cuando se pasa el ratón por encima se muestra el título, `title`.

Otra etiqueta a nueva en esta página es la utilizada en el mapa, para este caso se ha buscado con la ayuda de la herramienta Google Maps en dónde se encuentra el depósito, se genera un enlace y se incorpora de la siguiente manera.

```
  <iframe class="ElementosCuerpo" id= "Google" src="" width="270" height="270"
  frameborder="0" allowfullscreen>
</iframe>
 <div class= "PieFoto"><a target ="blank" href ="">Localización del Depósito</a>
 </div>
```

Código 30: Incluir un mapa

En `src=""` se añade la URL generada con Google Maps, para el enlace a una nueva pestaña se incluye el atributo `target ="blank"`, especifica dónde se abre el enlace.

En este caso el documento JavaScript es igual al del "Inicio", pero se añade una función de `Animacion()` encargada, como su nombre indica de la animación del SCADA, esta se explicará más adelante.

El CSS enlazado a este archivo consta de una parte dedicada al menú de navegación, y la parte superior igual que en "Inicio", pero también una parte dedicada al "cuerpo" de la página (zona gris), donde se indica la posición de cada elemento. Este código no ha generado conflicto en su creación, pero para que el lector se familiarice con el lenguaje se añade una parte del mismo donde se detalla el comportamiento de los pies de foto, seleccionado con el atributo `class = "PieFoto"`

```
.PieFoto{
  text-align: center;
  text-decoration: none;
  color: #5b5959
  font:bold 18px "Calibri";
  }
 .PieFoto a {
  text-decoration: none;
   color: #5b5959;
   font:bold 18px "Calibri";
 }

.PieFoto:hover{
  text-shadow: 2px 2px 2px #7a797a;
  font-weight : bold;
  }
```

Código 31: Estilo CSS para el pie de foto

3.4.4 Nivel 3: SCADA de la planta.

Este tercer nivel es realmente el que se considera el SCADA de la planta, esta vez, está divido en tres secciones, se ha decidido minimizar el menú de navegación para darle más importancia a la planta. En la figura X se muestra su aspecto.

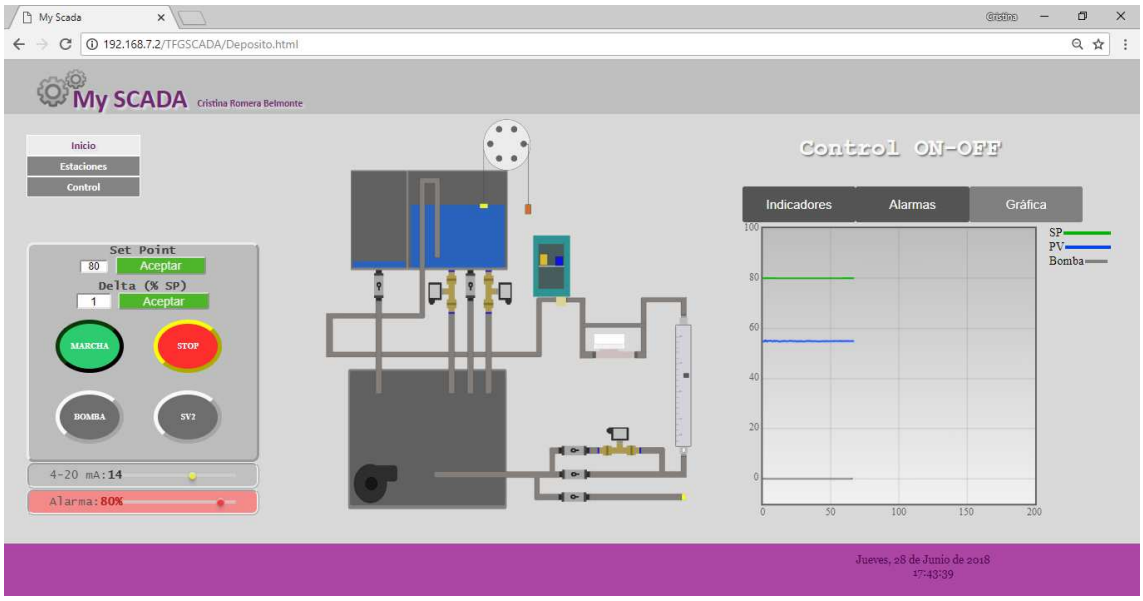


Figura 15:Nivel 3 My SCADA

En el interior del cuerpo, se pueden diferenciar cuatro secciones cada una destinada a realizar una función. En los siguientes apartados serán explicadas.

3.4.4.1 Menú de navegación.

Comenzando por la izquierda el primer elemento encontrado es el menú de navegación que consta de tres pestañas, la primera de ellas: "Inicio", permite al usuario regresar al Nivel 1 de la navegación. La siguiente pestaña, "Estaciones", facilita llegar de nuevo al Nivel 2, por último, de forma predeterminada se abre la estación con un control On-Off, por ello se ha habilitado una pestaña, para que el usuario decida si desea realizar un control más sofisticado como es el PID.

Al igual que el menú de "Inicio" se cuenta con las mismas etiquetas que en HTML, pero en este caso se hace uso de CSS para lograr el efecto de menú vertical. La diferencia reside en el valor de la propiedad `display`, para el caso del menú horizontal, existen dos alternativas, la primera con `display:inline` o la segunda forma con `float:left,` en cambio para el menú vertical se debe de insertar `display:block,` para que los elementos aparezcan uno debajo del otro.

3.4.4.2 Botonera.

La botonera ha sido pensada con la intención de que tenga un aspecto agradable y familiar, que ayude al operario, que no provoque errores en la operación y que cuente con todas las funcionalidades requeridas.

Inputs.

Comenzado desde la parte superior, se distinguen dos campos de entrada dónde el usuario puede especificar tanto el *Set Point*, el cual está limitado a un nivel del 99% porque aunque el depósito cuenta con un sumidero que impide que no rebose el agua, se decide no llegar a este límite.

Para el caso del control On-Off, se puede cambiar el valor de la histéresis (Delta) introduciendo un valor como porcentaje de la consigna, es decir si se quiere alcanza un valor del 80% de nivel de agua con un Delta de 1, quiere decir que la conmutación del estado en el límite superior se realiza cuando alcance un valor de 80,8% de agua y por consiguiente el otro cambio se efectúa en un valor de 79,2%. De forma predeterminada se sugiere un valor de consigna del 80% y un Delta de 1% de la consigna.

Por otro lado, en el control PID se incorporan los parámetros del regulador, los cuales según se han explicado en la parte de control, son la ganancia proporcional, integral y derivativa, K_p, K_i, K_d, de forma programada los valores de las ganancias serían 5, 0.7 y 0 respectivamente ya que son los valores que se han considerado óptimos según las pruebas prácticas realizadas.

Al tratase de parámetros de importancia, a modo de seguridad se incorpora un botón de aceptación del valor, donde se pregunta al operario si da el consentimiento o no para concretar la operación.

Para el botón de confirmación se incorpora el código 32 en JavaScript.

```
function confirmacionSetPoint(){
        var respuesta = confirm("Desea cambiar el Set Point a " +valorSP.value);
        if (respuesta == true) {
            mensajeSP.innerHTML = valorSP.value;
            SP = (valorSP.value);
            mensajeDelta.innerHTML =(valorDelta.value*SP)/100;
            //Cuando cambia el SP se debe de cambiar el parámetro Delta
            Delta = (valorDelta.value*SP)/100;
            console.log("valor corregido");

        } else {
            console.log("cancelar");
            valorSP.value = mensajeSP.innerHTML;
            SP = (valorSP.value);
        }
        console.log(valorSP.value);
        return valorSP.value:
```

Código 32: Función confirmación

Sección de Botones.

Debajo de los campos de entradas existen cuatro botones que corresponden a:

"Marcha": Si este botón es pulsado comienza el control de la planta y las respectivas funciones de animación del esquemático de la planta. Debido a que el circuito de agua está en continuo movimiento, cuando se activa este botón, también se abre la válvula de escape, si la planta se encuentra en "Marcha", existen operaciones que quedan restringidas como el encendido y apagado de la bomba y la apertura y cierre de la válvula de escape. Este botón se puede reconocer por su color verde.

"STOP": botón de parada del control, se deja de tomar muestras sobre la planta, los valores recogidos en el instante del apagado quedan guardados, se apaga la bomba y la válvula de escape permanece cerrada.

Las funciones de estos botones se controlan con una misma función se muestra en el código 33.

```
function ComienzaControl(x){
        estadoControl = x;
        if (estadoControl==0){
            console.log("stop");
            b.digitalWrite(Bomba, b.LOW);
            b.digitalWrite(SV2, b.LOW);
            dibujoBomba.style.fill = "#000000"; //Cambio de color de bomba en SVG
            dibujoSV2.style.fill = "#a0892c";
            dibujoSV2.style.stroke= "none";
            mensajeBomba.innerHTML = "OFF";
            mensajeSV2.innerHTML = "OFF";
            estadoMensaje.innerHTML = "STOP"; //Mensaje en el panel de indicadores
            estadoMensaje.style.color = "#FFFFFF";

        }else{
            console.log("marcha");
            b.digitalWrite(SV2, b.HIGH);
            dibujoSV2.style.fill = "#81F781";
            dibujoSV2.style.stroke= "#000000";
            mensajeSV2.innerHTML = "ON";
            estadoMensaje.innerHTML = "Marcha";
            estadoMensaje.style.color = "#298A08";
        }
        console.log(x);
    }
```

Código 33: botones "STOP" y "Marcha"

Cuando la función `ComienzaControl(X)`, recibe un 1 se hace referencia a la "Marcha" y por consiguiente 0 hace referencia a "STOP", se incluyen operaciones de animación como el cambio de color de los objetos (el color se indica en hexadecimal).

"Bomba" y "SV2": son botones que su uso está delimitado cuando la planta se encuentra en paro, es decir, no se está realizando la función de control, operan con la bomba (encendido y apagado de la misma) y la válvula de escape (abriéndola y cerrándola). Estos botones se pueden distinguir por su color gris.

A continuación, se muestra únicamente la función para el control de la bomba, pero para la servo válvula sería igual.

```
function ControlBomba(){

    if ((estadoBomba==1)&&(estadoControl==0)){
        estadoBomba=0;
        console.log("se apaga bomba");
        b.digitalWrite(Bomba, b.LOW);
        dibujoBomba.style.fill = "#000000";
        mensajeBomba.innerHTML = "OFF";
    }else if  ((estadoBomba==0)&&(estadoControl==0)){
        console.log("se enciende bomba");
        estadoBomba=1;
        b.digitalWrite(Bomba, b.HIGH);
        dibujoBomba.style.fill = "#A9F5A9";
        mensajeBomba.innerHTML = "ON";
    }
    console.log(estadoBomba);
}
```

Código 34: Control de la Bomba.

Control deslizante.

Al final de la botonera se encuentran dos controles deslizantes, cuya etiqueta de HTML, sería `<input></input>`, misma etiqueta utilizada para introducir el valor de consigna y el parámetro delta, con la peculiaridad de contar con el atributo `type="range"`, `type="text"` para el otro caso, Estos controles son sencillos de entender y de operar con ellos.

El primer deslizador, está dedicado al control de la servo válvula, éste sólo aparece en el control On-Off, por si se desea aumentar o disminuir el caudal, sin embargo en el control PID no aparece pues es el propio controlador quien realiza el control. Simula una fuente de corriente de (4 a 20mA).

El segundo deslizador desempeña una función de control de nivel, por si fuera necesario saber cuándo se alcanza cierto valor de nivel, que no tiene por qué ser el de consigna, normalmente se selecciona un valor superior a este.

```
        //Funcion para el valor de corriente

    valorCorriente.oninput = function() {
    indicadorCorriente.innerHTML = this.value; //Se muestra el valor al lado del slider
    elementoFlotante.setAttribute('y',(-7.0625*this.value + 242.14));//Ecuación de la recta,
parte de Animación
    CorrienteCaudal(this.value); //Llamada a otra funcion donde se convierte este valor de
corriente en caudal


   //Se incorpora una alerta tanto si se supera un valor máx como min
   //Porque se puede dañar la válvula
   if (this.value>18){
        alarmas[i].innerHTML = "Caudal Max de " + caudal.toFixed(2).css +" l/min";
        horas[i].innerHTML =today.toLocaleTimeString();
        i++;
   }

    if (this.value<5){
        alarmas[i].innerHTML = "Caudal Min de " + caudal.toFixed(2) +" l/min";
        horas[i].innerHTML =today.toLocaleTimeString();
        i++;
   }
};
```

Código 35: Función dedicada al deslizador de corriente

3.4.4.3 Sinóptico de la planta.

El esquemático de la planta actúa como una imagen a tiempo real del estado de la planta, en este experimento la distancia entre la planta y el SCADA es mínima, pero si no fuera así se debe de poder hacerse una idea de la misma.

Este esquema presenta grandes similitudes con la planta, porque en caso de avería o reposición de algún elemento, cuanto más parecido tenga la imagen a la realidad, la ejecución de dichas tareas se hará de una manera más diáfana.

El dibujo está escrito en su totalidad con el lenguaje de marcas XML, y tiene un formato ".svg " que ayuda a su visualización y resolución, se empleó en este caso la aplicación de escritorio Inkscape.

Cada elemento, los tanques, válvulas manuales, solenoides, servo válvula, caudalímetro, son objetos individuales que a su vez están formados por figuras más simples.

Se deben de crear de manera jerárquica cada elemento, empezando por figuras geométricas, o con la combinación de alguna de ellas, finalmente todos se unen para formar el esquemático. Al igual que en el lenguaje HTML, se puede añadir etiquetas identificativas a los elementos, aquí sucede los mismo. El programa de edición utilizado permite dibujar y crear objetos, él mismo incorpora esta etiqueta, pero es una buena práctica cambiar los nombres porque facilita el proceso de animación del esquema.

Como ejemplo se añade la figura 46 con la elaboración de la válvula solenoide en formato SVG.

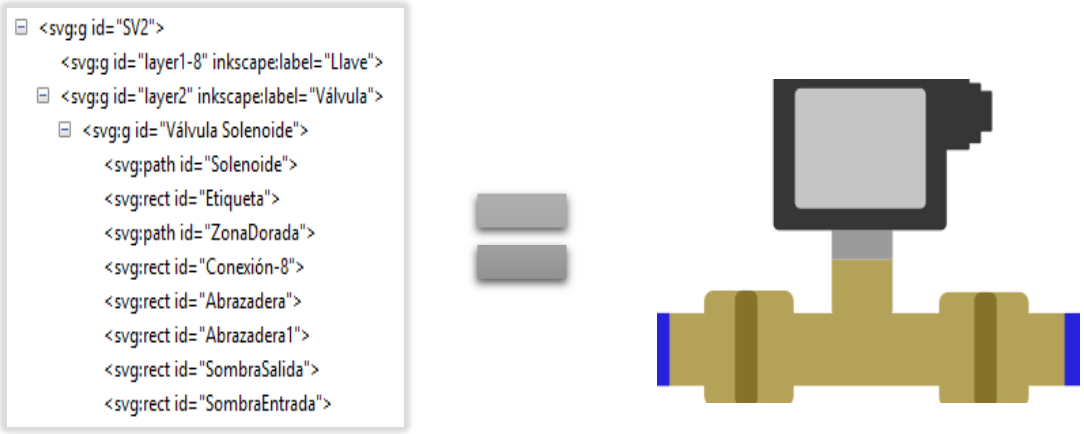


Figura46: Dibujo en formato SVG de solenoide

3.4.4.4 Indicadores, alarmas y gráfica.

Para facilidad del operario se ha decidido agrupar, los indicadores, el panel de alarmas y gráficas, en un solo panel. Que sea el operario quien decida, que prefiere evaluar.

Se puede dividir en dos partes, una primera dedicada a los botones, "Indicadores", "Alarmas" y "Gráficos", situada en la parte superior y la parte inferior con el contenido de cada sección.

"Indicadores": se deben de tener presente el valor de aquellas variables que se consideren importantes, para este caso el valor de la variable controlable, el nivel de agua (PV) y la manipulable, el estado de la Bomba o servo válvula. Además se muestra el valor de consigna (SP), el error, la conversión del parámetro Delta (sólo en control On-Off), el estado de la válvula de escape (SV2), el caudal y un apartado para ciertos mensajes que avisan en qué estado se encuentra la planta: "Marcha", mientras se realiza el control, "STOP", cuando no se realice el control , "Alarma", cuando el nivel sobrepase el valor preestablecido, con el indicador deslizante.

Los indicadores tienen todos el mismo color para no despistar al operario, en cambio los mensajes sí que cuentan con cambio de color, incluso el de "Alarma" tiene asignado la función de parpadeo, si se prefiere también se puede hacer uso de la librería jQuery.

```
var estadoMensaje; //Declaración
//Se referencia al elemento que tiene ese id en HTML
estadoMensaje = document.getElementById("mensaje");
estadoMensaje.innerHTML = "Marcha"; //Valor
estadoMensaje.style.color = "#298A08"; //Color
```

Código 36: Mensajes

"Alarmas": es un cuadro donde quedan registrados todos los acontecimientos de interés de la planta, por ejemplo, cuando se sobrepasa el nivel de alarma, y cuando el nivel de agua se repone. También se considera mensaje de interés cuando el operario decide poner al máximo (20mA) o al mínimo (4mA) la servo válvula. Los mensajes de alarma se acompañan del valor requerido, el nivel de agua y de corriente, y la hora a la que sucede el acontecimiento. La tabla cuenta con una capacidad de 40 alarmas y una vez sobrepasado este valor se comienza a rellenar de nuevo la tabla.

```
var i = 0; //Se declara fuera por esta función se llama cada cierta periodicidad

//De no ser así se reiniciaría a cero el valor de i, cada vez que fuera llamada
function Alarma(x){
    const valor = x.value;
    console.log(parametroAlarma);
    console.log(valor);

    //Sturacion del cuadro de información
    if(i==100){
     i=0;
    }
    if (!x.err){

        //Añade alarma en el cuadro Sobrepaso del nivel
        if((valor>parametroAlarma)&&(bool1==0)){
            bool1=1;
            b.digitalWrite(Bomba, b.LOW);
          //Conversión del valor leido al real
            alarmas[i].innerHTML = "Sobrepaso Nivel"+ (131.41*valor -
23.522).toFixed(2)
             +"%";
            horas[i].innerHTML =today.toLocaleTimeString();
            estadoMensaje.innerHTML = "Alarma"; //Mensje
            estadoMensaje.style.color = estadoMensaje.style.color == "white" ? "red" :
"white"; //Parpadeo
            aguaEle2.style.fill =  "#FF0000"; //Cambio de color del agua
            i++;
        }

        //Mantiene el color de alarma, pero no añade otro mensaje
        // (para no saturar el cuadro)
        if((valor>parametroAlarma)&&(bool1==1)){
            bool1=1;
            b.digitalWrite(Bomba, b.LOW);
            aguaEle2.style.fill =  "#FF0000";
            estadoMensaje.innerHTML = "Alarma";
            estadoMensaje.style.color = estadoMensaje.style.color == "white" ? "red" :
"white";
        }

        //No hay alarma
        if((valor<parametroAlarma)&&(bool1==1)){
            bool1=0;
            aguaEle2.style.fill =  "#0065ff";
            estadoMensaje.innerHTML = "Marcha";
            estadoMensaje.style.color = "#298A08";
            alarmas[i].innerHTML = "Reposición de Nivel";
            horas[i].innerHTML =today.toLocaleTimeString();
            i++;
        }
    }
}
```

Código 37: Gestión de alarmas

Asimismo en el código 35 se especifica que cuando se alcanza un valor de corriente límite en con el indicador de 4 a 20 miliamperios, se añade nuevamente una alarma.

"Gráfica": se incorpora una sección de gráficos, que de forma predeterminada es el contenido que al abrir el SCADA por primera vez se observa, ayuda generar una idea sobre las variables, se puede visualizar, tanto el valor de consigna como el de la variable manipulada y controlada (en ambos controles). A su derecha se muestra una leyenda, que coopera en el entendimiento del gráfico.

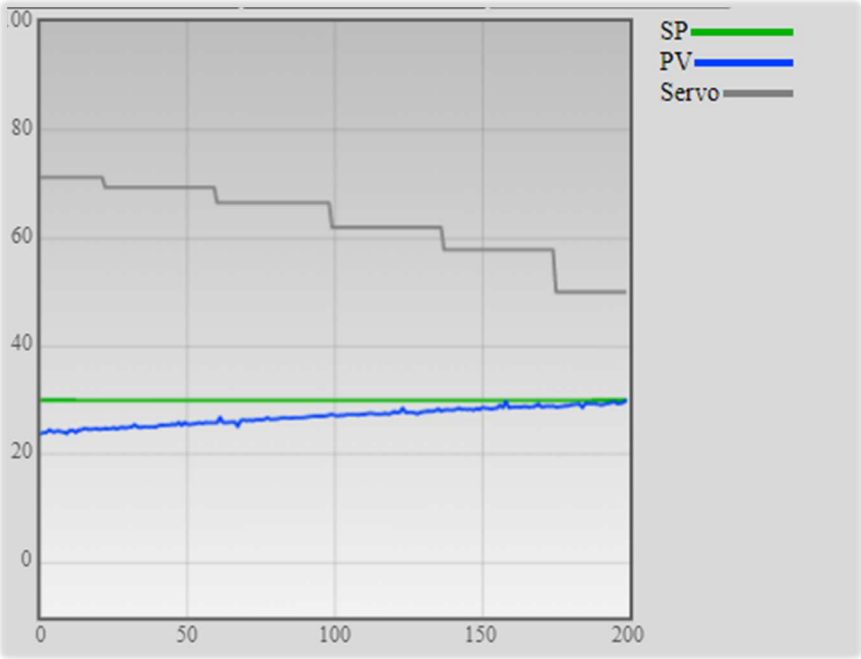


Figura 47: Gráfico del SCADA

3.4.4.5 Código

A pesar de que en las secciones anteriores se han incorporado partes del código, al ser éste muy complejo se necesita explicar de forma esquematizada, se recuerda que para tanto el control On-Off como para el PID, la estructura es la misma salvo adecuada al control.

En la parte superior se declaran todas las variables de uso global, mediante la palabra reservada `var`, de forma general se ha preferido incorporarles un valor indefinido hasta que sean utilizadas.

La siguiente parte está dedicada a la carga de todos los elementos del árbol DOM, sobre todo se espera a que se cargue el archivo ".svg" que contiene el dibujo del SCADA y conocida la dirección de BBB, se inicializa la función `run()`.

Las funciones que implican el uso de botón en el SCADA, se declaran fuera de la función principal, en este caso, la comentada anteriormente, `run()`.

Dentro de la función principal, se declaran aquellas variables necesarias, por ejemplo, para la animación, gestión de las alarmas y mensajes.

A continuación se llama a una de las funciones más importantes de este código: `muestreo()`, sólo en caso de que la planta se encuentre en "Marcha" se ejecuta las llamadas al resto de funciones, se utiliza la función `analogRead()`de la librería Bonescript después de que el valor sea leído se invocan el resto de funciones.

La función muestro actúa de igual forma tanto para el caso del control On-Off como para el PID, sólo cambia la llamada de la propia función de control, además que para el caso del control On-Off no es necesaria añadir aquí la llamada a la función `corrienteCaudal()`, pues no guarda relación con el valor muestreado

```
muestreo();
        function muestreo() {
            if (estadoControl==1){

                b.analogRead(PV,(lectura) => {
                console.log("lectura analog fue:", lectura);
                Animacion(lectura);
                Alarma(lectura);
                salidacontrol = pid1.salidaPID(lectura.value); //Método de la clase PID
            });
             corrienteCaudal(salidacontrol);
            }
            setTimeout(muestreo, 2000); //Muestreo cada 2 seg
        }
```

Código 38: Muestreo de la señal

Seguidamente se incorporan las declaraciones de las funciones, las funciones de control se comentaron en el apartado de control, a continuación, se explican el resto de ellas.

Animacion()

`Animacion()` se dedica como su nombre indica a la animación del SCADA, se ha añadido movimiento al agua, al flotador y sensor de nivel, la bomba y servo válvula, y al elemento flotante del caudalímetro.

Para ello se hicieron ecuaciones de rectas que relacionaban la posición del objeto con el valor medido, por lo tanto, son movimientos lineales.

La forma de proceder es la siguiente, tomaremos como ejemplo el movimiento lineal que realiza el agua del tanque superior.

1. Se declara el elemento que se quieren animar.

```
const aguaEle2 = svg.contentDocument.getElementById("Agua-6");
```

Código 39: Declaración del elemento que se animar

2. Se guarda en una variable el atributo que se necesita manipular para la animación.

```
const aguaMaxHeight2 = aguaEle2.getAttribute('height'); //Valor max de Altura
```

Código 40: Declaración de la altura máxima del agua

3. Se encuentra la ecuación de la recta que relaciona el movimiento del agua del tanque superior con el valor muestreado.

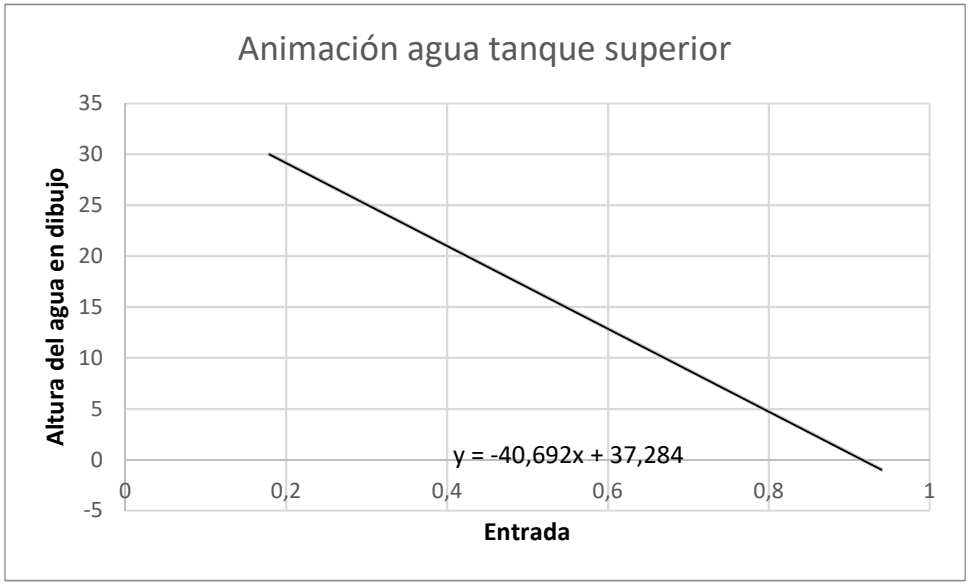


Figura 48: Relación matemática entre la señal de entrada y la animación

4. Se realiza la manipulación del valor del atributo con el método `SetAttribute()` en su interior se incluye el atributo y el valor del mismo, en este caso como el atributo ya está creado `'height'` simplemente se cambia su valor, de no ser así se crearía el atributo correspondiente. En este caso la función viene dada por su valor máximo declarado anteriormente.

```
aguaEle3.setAttribute('height',(((valor*aguaMaxHeight3*-2999)/0.76100)
+37.504));
```

Código 41: Ecuación de la recta para el movimiento

Por último, se incluye la función Animación, para el resto de elementos la manera de proceder es la misma.

```
function Animacion(x) {
      if (!x.err){
          const valor = x.value;
          //Indicador del nivel Tanque sup
          $('#TanqueNivelS').html((137.36*valor - 24.588).toFixed(2));
          //Animación
          aguaEle2.setAttribute('height', (((valor*aguaMaxHeight2)/0.76100) –
          11.748)); //Animación para el Tanque Sup
        Cuerda.setAttribute('height',(((valor*cuerdaMaxHeight*0.57)/0.76100) +
         20.756));//Animación cuerda 1
          Elemento.setAttribute('y',((-2*valor*elementoMaxY)/0.76100) +
69.485);//Animación del  flotador
          Cuerda2.setAttribute('height',(((  -
1.176258223*valor*cuerdaMaxHeight2)/0.761) + 135.55));//Animación cuerda 2
          Elemento2.setAttribute('y',((-84.421*valor) + 45.734));//Animación del
peso
      }
}
```

Código 42: Función animación completa

Corriente()

Esta función se dedica a la transformación del valor dado ya sea por el indicador de corriente, (SCADA con control on-off) o por la propia salida del control PID, en una salida analógica, para el segundo (PID) se hace una conversión de porcentaje de apertura de la válvula a corriente, se emplea una conversión lineal, dónde 100% corresponde a 20 miliamperios y 0% a 4 miliamperios.

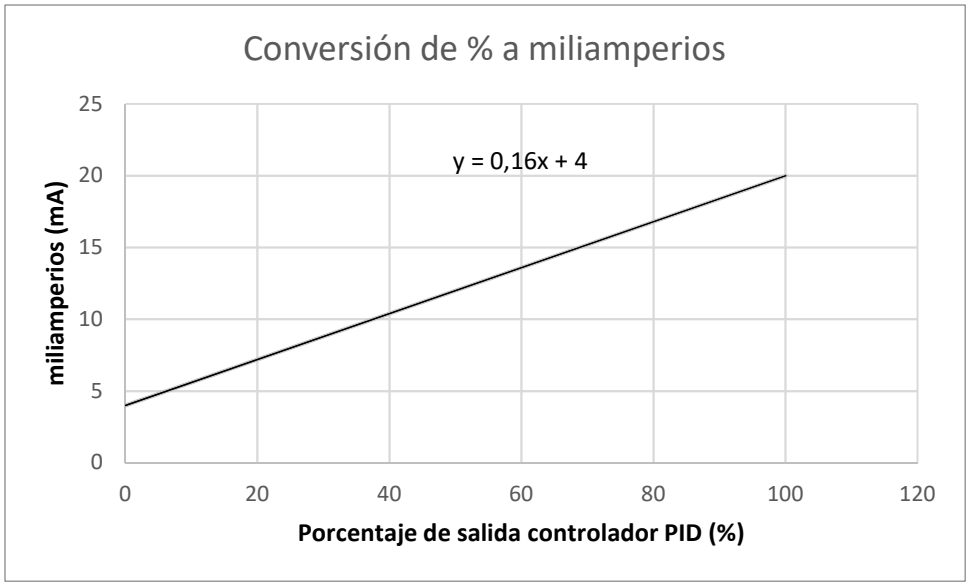


Figura 49: Relación matemática entre la salida del controlador y miliamperios

Primero se divide el valor en corriente (4 a 20 miliamperios) entre el mínimo valor de corriente a la salida del circuito electrónico, este resultado indica cuantos niveles, refiriéndose a los niveles del conversor digital analógico, son necesarios para obtener este valor, éste se guarda en la variable nivel (hay 256 niveles posibles).

```
const dosn = 256; //constante
const resistencia = 5000;
const mili = 1000;
const amplificacion = 10;
const resolucion = (voltajeRef/(dosn*resistencia))*mili*amplificacion ;
var conversionporama;
conversionporama = 0.16 *valor + 4;
var nivel = conversionporama/resolucion;
```

Código 43:Declaración de constantes y variables

Cabe destacar que es difícil que este valor sea exacto por tanto se debe de decidir entre si se sube de nivel o se mantiene el nivel, pues los posibles niveles del conversor son números enteros para ello se diferencia de parte entera y decimal, se redonda al alza con valores mayores o iguales a 0,5.

Una vez se obtiene el nivel, se debe de realizar su conversión a binario, el resultado se vuelca a los 8 bits de resolución que ofrece el conversor.

```
var partentera = Math.trunc(nivel);
var partedecimal = nivel - partentera;
var numcomversion;

if (partedecimal>=0.5){
   numcomversion = partentera + 1;
}else{
    numcomversion = partentera;
}
 if (nivel>255){
    numcomversion = 255;
}
```

Código 44:Decisión del nivel

Para la conversión decimal a binario se aprovecha el método `toString(2)`, el valor devuelto por este método es una cadena de caracteres, en este caso en la base que se indica, es decir, en binario, el problema es que este método sólo utiliza los dígitos necesarios para la conversión, por ejemplo el número "2" sería "10" y el 15 "1111" , y la salida buscada tiene que tener un valor para cada uno de los bits, aquellos que no fueron utilizados tienen que ser igual a cero.

Para ello se recorre la cadena de caracteres empezando por la posición número ocho y se guarda el valor en un vector, en algunos casos este puede ser cero o uno, y en otros puede no estar definido debido a que en la conversión no fue necesario.

```
var n = numcomversion.toString(2); //Salida del código anterior
var bin = []; //Resultado binario con 8 dígitos
//Bin[0] = MSB Bin[7]= LSB

//Se recorre el string de manera inversa
for (var i = 8; i>0; i--){
    //Comprueba si el valor es nulo o indefinido
    if((n[n.length-i]==0)||(n[n.length-i]== undefined)){
        bin[8-i]=0; //la primera posición del vector es 0 y la última 7
    }else{
     bin[8-i]=1; //
    }
}
```

Código 45:Se completa el resto de los bitsl

Por último, se indica el estado de los pines del DAC según la conversión hecha anteriormente.

```
for( var i in pins) {
    b.pinMode(pins[i], b.OUTPUT);
    b.digitalWrite(pins[i], bin[i]);
}
```

Código 46:Salida l

Para concluir este apartado destinado a la explicación del código más significativo de este trabajo se incluye el último apartado destinado a la generación del gráfico que incluye la evolución temporal en el tiempo tanto de las variables controladas como las manipulables, tanto para el control On-Off como para el control PID.

Generación del gráfico

El código que se adjunta a continuación no es de propia autoría, por ello se agrega el enlace de donde se obtuvo en la bibliografía de este proyecto [87].

Este código requiere de la implementación de la librería Flot, la cual está desarrollada en su totalidad en el lenguaje JavaScript para jQuery, esta librería trae consigo un enfoque de uso sencillo y atractivo. Para poder utilizar este recurso hay que agregarlo entre las etiquetas `<head></head>`.

```
<script src="/static/flot/jquery.min.js"></script>
<script src="/static/flot/jquery.flot.min.js"></script>
```

Código 47: Librería gráfica l

Se han ajustado los parámetros y creado nuevas funciones para cumplir con los requisititos de este trabajo. En esta primera parte (código 47) se declaran las características de la gráfica y los vectores necesarios para guardar los datos.

```
    var container = $("#Nivels"); //id del div HTML
    var totalPoints = container.outerWidth() / 2 || 250; //Eje de
abscisas
    var data = []; //Vector de datos PV
    var data1 = []; //Vector de datos SP
    var data2 = []; //Salida control

     //Características de la gráfica
    var plotOptions = {
        series: {
            shadowSize: 0
        },
        yaxis: {
            min: -10,
            max: 100
        },
        xaxis: {
            min: 0,
            max: totalPoints,
            show: true
        },

        grid: {
                    hoverable: true,
                    borderWidth: 2,
                    backgroundColor: { colors: ["#BDBDBD", "#F2F2F2"]
}
        }

    };
```

Código 48: Características de la gráfica

En la segunda parte del código se crea una variable `plot` donde se vuelvan todos los datos.

```
var plot = $.plot(container, getData(), plotOptions); //función de Flot

drawGraph();

function drawGraph() {
    plot.setData(getData());
    plot.draw();
    b.analogRead(PV,LecturaPV);
    pushData1(pid1.SP);
    pushData2(salidacontrol);

   //b.digitalRead(Bomba, LecturaBomba); //Uso en el control On-Off

    setTimeout(drawGraph, 100);

}

// Se le el valor y se transforma a un valor de 0 a 100
//Esta función ejectuta lo que hay en su interior
//Cuando el valor de la medida es correcta y tipo = número
function LecturaPV(x) {
    if (!x.err && typeof x.value == 'number') {
        pushData((137.36*x.value - 24.588).toFixed(2));
    }
}

 //Devuelve una copia del vector en el caso que
//supera el tamaño del contenedor de la gráfica

  function pushData(y) {
    if (data.length && (data.length + 1) > totalPoints) {
        data = data.slice(1);
    }
```

Código 49: Se recogen los datos y se guardan en vector

Por último, se muestran las series de datos.

```
//Se recorren las posiciones de los vectores y se crean las series de
datos
    function getData() {
        var res = [];
        var resSP = [];
        var reServo = [];
        for (var i = 0; i < data.length; ++i) {
            res.push([i, data[i]]);
        }

        for (var i = 0; i < data2.length; ++i) {
            reServo.push([i, data2[i]]);
        }
        for (var i = 0; i < data1.length; ++i) {
            resSP.push([i, data1[i]]);
        }
        var series = [
             {data: resSP,
               color:"#04B404",
                 lines: {
                    fill: false,
                    }

                 },

                {data: reServo,

                 color:"gray",
                 lines: {
                    fill:false,
                    }

                 },

                 {data: res,
                 color:"#0040FF",
                 lines: {
                    fill: false,
                    }

        }];
        return series;
    }
```

Código 50: Se muestran los datos en la gráfica

Todo el código desarrollado se encuentra en [89].

Capítulo 4: Conclusiones

4.1 Conclusiones

Después de este largo aprendizaje sobre el uso e incorporación de las tecnologías que hoy dominan la vida diaria, en referencia nuevamente a las tecnologías para el desarrollo web, se puede decir que han cumplido con las expectativas propuestas, poder crear un sistema SCADA desarrollado al completo con ellas.

No han limitado en ningún momento las ideas que se propusieron en un comienzo, al contrario, estas herramientas han ayudado a que éstas pudieran ser materializarlas. Este proyecto nació desde cero, su autora desconocía el uso de estas herramientas, por ello se ha podido ver una larga evolución, porque conforme el proyecto y las aspiraciones de crear un entorno web apto para el control y monitorización de cualquier sistema crecieron, también lo hicieron sus conocimientos.

Durante el desarrollo de este trabajo se ha creído siempre en el poder de estar herramientas, y de sobre todo de su alcance, pues no han supuesto un costo elevado, simplemente dedicación y tiempo, por lo que cualquier otra persona con una base de conocimiento en informática puede alcanzar. Sobre todo, este proyecto mira por aquellas personas que deseen controlar un pequeño sistema y que por su sencillez no se necesite invertir una gran cantidad de dinero.

Además, la comodidad de poder crear un sistema a tu gusto y tus propias necesidades añade un valor a este trabajo, que si en un futuro fuera necesario ampliar o mejorar, es fácilmente realizable.

Por último, una de las muchas comodidades que añade este tipo de herramientas es su acceso desde cualquier plataforma digital, lo que lo hace aún más manejable. Asimismo, al tratarse de herramientas en constante evolución, hay infinidad de información en la web.

En conclusión, las herramientas utilizadas han satisfecho con creces los objetivos planteados y aspiran a ser una herramienta que el día de mañana se quiera introducir en cualquier planta industrial.

4.2 In conclusion

After this long learning about the use and incorporation of the technologies that today dominate the daily life, in reference again to the technologies for the web development, it can be said that they have fulfilled the proposed expectations, to be able to create a SCADA system developed in full with them.

They have not limited at any time the ideas that were proposed at the beginning, these tools have helped these could be materialized. This project was born from scratch, its author was unaware of the use of these tools, so it has been able to see a long evolution, because according to the project and the aspirations of creating a web site suitable for the control and monitoring of any system grew with them her knowledge.

During the development of this work, we have always believed in the power of being tools, and above all of its scope, since they have not involved a high cost, simply dedication and time, so any other person with a knowledge base in IT can reach. Above all, this project looks for those people who want to control a small system and that for its simplicity does not need to invest a large amount of money.

In addition, the convenience of being able to create a system to your liking and your own needs adds value to this work, which if in the future it were necessary to expand or improve, is easily achievable.

Finally, one of the many amenities that this type of tool adds is its access from any digital platform, which makes it even more manageable. Also, as they are tools in constant evolution, there is an infinity of information on the web.

In conclusion, the tools used have amply met the objectives set and aspire to be a tool that tomorrow you want to introduce in any industrial plant.

4.3 Líneas futuras

En este capítulo se incorporan las posibles mejorar para completar aún más el sistema creado.

Empezando por el propio sistema y su desarrollo, se propone un sistema de seguridad que debería limitar ciertas operaciones a los usuarios del SCADA en función del rol que tengan en la planta industrial, pues bien, a un supervisor se le debe de permitir el acceso a cualquiera de los parámetros de control, sin embargo, a un operario o cualquier persona que no posea los conocimientos necesarios sobre la misma no debería poder cambiar a su antojo parámetros que estén relacionado con el control de la planta, pues podrían provocar errores en la línea de producción, De esta forma se asegura un buen desarrollo de la actividad industrial.

También se debería de incorporar una base de datos dónde queden almacenados todos los valores recogidos ya que hasta ahora éstos se quedan en el navegador y una vez se cierra la pantalla son eliminados, si se consiguiera esto se podrían crear gráficas que muestren el histórico de la planta y una forma de acceder a ellos es a través de botones de acceso directo.

Incorporar un panel más completo de la configuración del sistema, en este proyecto no se dejó a elección del usuario el tiempo de muestreo de la señal, pero puede ser un parámetro que dé juego en el futuro en función de la planta que se desea monitorizar.

En cuanto a posibles mejoras teniendo como base este mismo proyecto, se debería de perfeccionar el sistema de alarmas dónde conforme surgen nuevas alarmas la tabla vaya incrementando.

Para el gráfico que se incorpora, conseguir que a medida que la recogida de los valores avanza en el tiempo también lo haga los valores del eje, para saber el instante de tiempo correcto, además de incorporar un botón que amplíe su tamaño.

Capítulo 5: Presupuesto

Para la realización del proyecto se ha utilizado los siguientes medios materiales y humanos detallados en el presupuesto que se muestra a continuación cuyo valor total asciende a 2.054,53€

Tabla 18: Presupuesto

Unidades	Concepto	Precio
1	BeagleBone Black	42,59 €
1	Dac0808lcn	1,89 €
1	Lm741cn	0,59 €
2	Resistencias de 5000 ohmio (5k)	0,05 €
1	Condensador de 0.1 microfaradio	3,51 €
1	Resistencia de 1000 ohms	0,05 €
1	Resistencia de 100 ohms	0,05 €
1	Resistencia de 82 ohms	0,05 €
1	Resistencia de 4700 ohms	0,05 €
1	Placa Protoboard	5,70€
1	Ingeniero Técnico	2000€
Total		2.054,53€

Capítulo 6: Anexos y Bibliografía

6.1 Anexos

Paro no incrementar el número de páginas de este trabajo se incorporan los Datasheets de los componentes electrónicos, DCA0808 y LM741 a través de un enlace web.

- DAC0808: http://www.ti.com/lit/ds/symlink/dac0808.pdf
- LMT741: http://www.ti.com/lit/ds/symlink/lm741.pdf

6.2 Bibliografía

[1] https://w3.siemens.com/mcms/human-machine-interface/en/visualization-software/wincc-flexible/pages/default.aspx
[2] Figura 1: http://grupofranja.com/index.php/oftalmica/item/1763-de-la-industria-1-0-a-la-4-0
[3] https://instrumentacionycontrol.net/introduccion-a-los-dcs-sistemas-de-control-distribuido/
[4] Figura 2: https://instrumentacionycontrol.net/introduccion-a-los-dcs-sistemas-de-control-distribuido/
[5] Figura 3: https://ingenieriabjb.wordpress.com/2014/04/15/procesos-de-automatizacion-y-control/
[6] https://www.uv.es/rosado/courses/sid/Capitulo3_rev0.pdf.
[7] Figura 4: https://www.panamcom.com/blogs/soluciones/20854593-redes-lan-nbsp-span-y-wan-span
[8] Figura 5: http://homepage.cem.itesm.mx/vlopez/buses_de_campo.html
[9] Figura 6: http://www.propysa.net/
[10] http://www.uco.es/grupos/eatco/automatica/ihm/descargar/scada.pdf
[11] http://www.oocities.org/gabrielordonez_ve/MTU.htm
[12] http://www.oocities.org/gabrielordonez_ve/Unidades_Remotas_SCADA.htm
[13] http://www.infoplc.net/documentacion/7-comunicaciones-industriales/2474-manual-redes-comunicacion-industrial
[14] https://www.copadata.com/es/soluciones-hmi-scada/interfaz-hombre-maquina-hmi/
[15] https://es.wikipedia.org/wiki/HMI
[16] https://elinux.org/Beagleboard:BeagleBoneBlack
[17] Figura 9: https://www.adafruit.com/product/1876
[18] Tabla 10: https://beagleboard.org/getting-started
[19] Figura 14: http://randomnerdtutorials.com/cloud9-ide-on-the-beaglebone-black/
[20] http://beagleboard.org
[21] http://isa.uniovi.es/~ialvarez/Curso/descargas/BeagleBone/InstalacionYManejoBeaglebone.pdf
[22] Figura 10: https://www.robotshop.com/en/arduino-uno-r3-usb-microcontroller.html
[23] https://aprendiendoarduino.wordpress.com/tag/avr/
[24] Figura 11: http://www.techmake.com/00571.html
[25] https://es.wikipedia.org/wiki/Raspberry_Pi#Pre-lanzamiento
[26] Figura 12 https://www.lankatronics.com/dragon-board/dragon-board-410c.html
[27] Figura 13: https://ca.wikipedia.org/wiki/Fitxer:BeagleBone.jpg

[28] https://www.adictosaltrabajo.com/tutoriales/un-entorno-de-programacion-en-la-nube-un-vistazo-a-cloud-9-ide/
[29] https://en.wikipedia.org/wiki/Cloud9_IDE
[30] Figura 18https://wiki.genexus.com/commwiki/servlet//hwikiprinterfriendly?16895,20
[31] Tabla 4: http://beagleboard.org/static/beaglebone/latest/README.htm
[32] Figura 20: http://randomnerdtutorials.com/cloud9-ide-on-the-beaglebone-black/
[33] Figura 24: https://graffica.info/formato-svg-ventajas/n
[34] Figura 25: https://www.youtube.com/watch?v=3Bhg727wYMc
[35] Figura 16: http://www.feedback-instruments.com/pdf/brochures/38-001_datasheet_level_flow_control_ESPIAL_10_2013.pdf
[36] https://es.wikipedia.org/wiki/Adquisición_de_datos
[37] Figura 19: http://panamahitek.com/escalando-unidades-de-conversion-analoga-digitales/
[38] https://es.wikipedia.org/wiki/Teorema_de_muestreo_de_Nyquist-Shannon
[39] http://ceres.ugr.es/~alumnos/luis/mycuan.htm
[40] http://plcdesign.xyz/por-que-4-20-ma/
[41] Figura 20: https://instrumentacionycontrol.net/la-senal-4-20ma-y-su-proporcion-a-variables-fisicas-nunca-esta-de-mas-repasarlo/
[42] http://mapir.isa.uma.es/varevalo/teaching/automatica/pdfs/Tema%2004%20-%20Respuesta%20Temporal%20con%20Routh%20v2_vicente.pdf
[43] http://www.eng.newcastle.edu.au/~jhb519/teaching/caut1/Apuntes/PID.pdf
[44] https://upcommons.upc.edu/bitstream/handle/2117/6123/TEMA6.pdf
[45] https://es.wikipedia.org/wiki/Controlador_PID
[46] https://devcode.la/blog/frontend-y-backend/
[47] https://www.campusmvp.es/recursos/post/Desarrollador-web-Front-end-back-end-y-full-stack-Quien-es-quien.aspx
[48] https://serprogramador.es/que-es-frontend-y-backend-en-la-programacion-web/
[49] https://developer.mozilla.org/es/docs/HTML/HTML5
[50] https://tecnologia-facil.com/que-es/que-es-html/
[51] https://devcode.la/blog/que-es-html/
[52] http://librosweb.es/libro/ajax/capitulo_4.html.
[53] Figura 23: http://www.hipertexto.info/documentos/dom.htm
[54] http://librosweb.es/libro/ajax/capitulo_4/html_y_dom.html
[55] https://es.wikipedia.org/wiki/JQuery
[56] http://nereida.deioc.ull.es/~pcgull/hli03/html/node2.html
[57] https://devcode.la/blog/que-es-javascript/
[58] https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/Qu%C3%A9_es_JavaScript
[59] https://www.campusmvp.es/recursos/post/JavaScript-ECMAScript-ES6-Existe-ES7-Aclarando-las-diferentes-versiones-del-lenguaje.aspx
[60] http://www.masadelante.com/faqs/css
[61] https://developer.mozilla.org/es/docs/Learn/CSS/Introduction_to_CSS/Como_funciona_CSS
[62] http://www.arumeinformatica.es/dudas/css/
[63] https://es.wikipedia.org/wiki/World_Wide_Web_Consortium
[64] http://www.ite.educacion.es/formacion/materiales/107/cd/imagen/imagen0105.html
[65] https://graffica.info/formato-svg-ventajas/
[66] http://www.ite.educacion.es/formacion/materiales/107/cd/imagen/imagen0105.html
[67] http://fp.uoc.edu/blog/imagenes-formato-svg/

[68] https://es.wikipedia.org/wiki/Gr%C3%A1ficos_vectoriales_escalables
[69] Figura 25: https://www.pinterest.com.mx/pin/103019910204665320/
[70] https://es.wikipedia.org/wiki/Gr%C3%A1ficos_vectoriales_escalables
[71] https://nodejs.org/es/
[72] https://es.wikipedia.org/wiki/Node.js
[73] https://www.luisllamas.es/tutorial-nodejs-montar-servidor/
[74] Figura 26: https://github.com/szmoore/MCTX3420/wiki/Hardware:-BeagleBone
[75] https://elandroidelibre.elespanol.com/2013/03/todo-sobre-el-usb-otg-que-es-como-se-usa-es-compatible-mi-smartphone.html
[76] https://medium.com/@Tabletchina/los-usos-para-cable-usb-mini-y-micro-hdmi-e14bc3a92bce
[77] https://es.wikipedia.org/wiki/Universal_Serial_Bus
[78] https://www.1and1.es/digitalguide/servidores/configuracion/que-es-el-dhcp-y-como-funciona/
[79] https://developer.mozilla.org/es/docs/Web/API/EventTarget/addEventListener
[80] http://www.feedback-instruments.com/pdf/brochures/38-001_datasheet_level_flow_control_ESPIAL_10_2013.pdf
[81] http://www.electrontools.com/Home/WP/2016/04/14/valores-comerciales-de-resistencias/
[82] https://www.luisllamas.es/calculadora-filtro-paso-bajo-pwm/
[83] http://materias.fi.uba.ar/6722/Nivel.pdf
[84] https://www.google.es/search?q=lazo+de+control&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiWpaHjjYbcAhXIL1AKHVZSB3sQ_AUICigB&biw=1517&bih=735#imgrc=seOiVGqXkN_o_M:
[85] http://csd.newcastle.edu.au/SpanishPages/clase_slides_download/on_off.pdf
[86] https://zulaco64.updog.co/Material_Didactico/Control-OnOff_Sistemas_Control_Controladores.pdf
[87] http://jsfiddle.net/jkridner/rC4AD/
[88] https://desarrollowp.com/blog/tutoriales/como-cargar-correctamente-los-archivos-js-y-css-si-estas-desarrollando-un-tema-o-plugin-a-medida/
[89] https://github.com/ULL-InformaticaIndustrial-Empotrados/TFG_Cristina_SCADA