UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Sistema basado en microcontrolador para la automatización de un acuario

Jesús Montañez Ruiz

Director: Francisco José Ballester Merelo

Trabajo de Fin de Master para optar por
el título de Máster Universitario en
Ingeniería de Sistemas Electrónicos

Universitat Politécnica de Valencia

Valencia, 30 de Abril de 2017

Versión DIGITAL

Índice de contenidos

Anexos de la Memoria

Índice de ilustraciones

1. Introducción

1.1 Motivación

La idea de este proyecto surgió tras las continuas quejas de un familiar, mi hermana, por la dificultad que conllevaba el mantenimiento de su acuario. El hecho de tener que introducir la mano en el acuario para regular la potencia del calentador o estar encendiendo y apagando el filtro, la luz y el oxigenador a diario le obligaban a estar pendiente continuamente.

Estudiando cómo solucionar el problema pude ver que aplicando conocimientos obtenidos en diferentes asignaturas del master, Sistemas embebidos, Sensores y Adquisición de Datos y Diseño Electrónico orientado a producto podía resolver el problema dando rienda suelta a mi imaginación, partiendo de cero para obtener un producto que se adapte a los requerimientos establecidos.

El propósito del proyecto consiste en desarrollar un sistema basado en un microcontrolador para automatizar el control de un acuario. Este sistema permitiría ajustar diferentes parámetros como tiempo de filtrado, temperatura ideal, modo de oxigenación y controlar el encendido y apagado de todos los aparatos conectados de forma sencilla, todo ello a través de una interfaz gráfica implementada en una pantalla LCD táctil, la cual mostrará el estado del acuario en todo momento. Un pequeño sistema al que se denominará "Akuatización" y que permitirá al usuario una total independencia de su acuario.

Dada la actual importancia de la automatización eléctrica, uno de los sistemas de automatización más empleados y extendidos en la actualidad tanto en la industria como en la vida diaria, consideré que el desarrollo de este sistema sería una buena propuesta para el TFM, opinión compartida por mi profesor de Sistemas Embebidos, Francisco José Ballester.

1.2 Objetivos

Generales:

- Crear un sistema mediante un microcontrolador que permita al usuario controlar el estado de su acuario en todo momento, haciendo invisible toda la programación o dispositivos utilizados y que con un mínimo de conocimientos eléctricos se pueda implementar en un acuario genérico.

- Se pretende automatizar un proceso manual y mejorar el control del proceso, la seguridad y el confort de la persona.

- Lograr que el diseño de este sistema se realice considerando en todo momento el coste de cada uno de los componentes necesarios consiguiendo abaratar el precio del producto final, obteniendo un producto que pueda ser utilizado en la mayoría de los acuarios sin suponer una gran inversión para el usuario.

Específicos:

- Establecer las conexiones necesarias entre los distintos dispositivos utilizando diferentes tipos de comunicación como por ejemplo SPI, I2C o 1-Wire, asegurando la seguridad de los mismos.

- Diseñar un programa para que el microcontrolador sea capaz de controlar el encendido y el apagado de todos los dispositivos útiles para el mantenimiento del acuario.

- Diseñar una interfaz gráfica sencilla con la que el usuario pueda monitorizar el estado de los dispositivos conectados, los parámetros de control y las variables a medir pudiendo cambiar la configuración existente en cualquier momento a través de una pantalla de ajustes.

1.3 Estructura de la memoria

El presente documento se encuentra dividido en cuatro capítulos: Introducción, Estado del arte, Desarrollo y Conclusiones; más los capítulos correspondientes a los anexos, presupuesto y a la bibliografía, ubicados al final del mismo.

En primer lugar, se encuentra el capítulo de Estado del Arte, en él se explica brevemente en que consiste la automatización eléctrica y como ha ido ganando terreno con el paso de los años. Se introducen los conceptos vinculados a: bus I2C, comunicación One Wire y el bus SPI, todos ellos utilizados para el control de los diferentes dispositivos. Además, se comentará el marco histórico de los microcontroladores, la base de este proyecto, detallando como han ido evolucionando con el paso del tiempo y como nos los podemos encontrar en la actualidad.

Seguidamente, el capítulo de desarrollo detalla paso a paso cómo se diseñó el sistema de automatización para el control del acuario, desde la idea principal hasta el sistema final obtenido, comentando como fue su avance y como ha quedado estructurado el control, explicando este último apartado a través de diversos diagramas de flujo. Comenzaremos explicando brevemente todo el hardware al que se tiene acceso, continuaremos con el desarrollo del proyecto y se finalizará con las pruebas y resultados.

En los resultados se presentará la descripción de la implantación de nuestro sistema "Akuatización" en un acuario de 60L, donde se podrá comprobar cómo se ha pasado a automatizar por completo un proceso que anteriormente se realizaba de forma manual., obteniéndose siempre resultados satisfactorios.

Ilustración 1 Acuario 60L

Por último, en el capítulo de conclusiones se establecen los objetivos alcanzados y se presentan las principales conclusiones extraídas del análisis de resultados. También, de este análisis surgen ideas y propuestas de trabajos futuros a desarrollar, las cuales son introducidas.

2. Estado del Arte

2.1 La automatización eléctrica [1][3]

La automatización eléctrica es uno de los sistemas de automatización más empleados y extendidos en la actualidad tanto en la industria como en la vida diaria. Ésta ha permitido un aumento en la producción industrial, un mejor control de los procesos, y una mayor seguridad y confort de las personas en la sociedad actual.

La automatización eléctrica es un sistema diseñado con el fin de aprovechar la capacidad de las máquinas en la realización de determinadas tareas realizadas por seres humanos, así como para controlar la secuencia de dichas operaciones sin su intervención. El empleo de la automatización eléctrica no se limita solamente a la producción industrial, sino que ésta se utiliza en cualquier otro sector en que se requiera el funcionamiento independiente o semi-independiente de algún dispositivo.

Los sistemas de actuación eléctrica se basan en motores, actuadores electromagnéticos y otros, y el mando eléctrico es generalmente mediante relés.

La automatización eléctrica conjuntamente con la electrónica es una de las más empleadas actualmente dentro de las diferentes variantes de automatizaciones existentes (sistemas de automatización mecánica, neumática e hidráulica). Es aplicada en las máquinas, herramientas y sistemas de producción en series industriales, en las comunicaciones, en los sistemas de transporte, en la astronáutica, en algunos sistemas de seguridad, incluso en el hogar, como por ejemplo, en muchos equipos electrodomésticos.

Las ventajas que estos sistemas de automatización aportan a nuestro día a día son más que evidentes, entre ellas se encuentran:

- Humanización del trabajo: libera al operario de la carga de ciertos trabajos forzados y tediosos.
- Mayor aprovechamiento laboral: las máquinas y sistemas automatizados son generalmente más productivas y eficientes que el ser humano.
- Mayor seguridad laboral: realizan el trabajo más engorroso y perjudicial.
- Mayor efectividad y exactitud: los sistemas de automatización eléctrica son de gran ayuda, ya que permiten una mayor eficiencia, rapidez y confiabilidad de los mismos.

- Mayor confort: la cualidad de estos sistemas de funcionar independiente o semi-independientemente tiene un efecto positivo sobre la vida de la sociedad, incluso en la vida privada o del hogar, mediante el empleo de estos sistemas automatizados en equipos electrodomésticos y otros.

Es evidente que la automatización eléctrica trae grandes beneficios y es extremadamente útil en el mundo moderno, cualquier aspecto negativo que pueda señalarse no es culpa de la automatización en sí, sino del proceder humano y del uso que pueda hacer de ésta.

2.2 Microcontrolador[2][4][5]

Ilustración 2. Microcontrolador ATMEGA2560

Un microcontrolador es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento, memoria y periféricos de entrada/salida.

Podemos encontrar desde microcontroladores que utilizan palabras de cuatro bits y funcionan a velocidad de reloj con frecuencias tan bajas como 4 kHz que tienen con un consumo de potencia del orden de mW o microvatios, hasta microcontroladores que sirven para roles de rendimiento crítico, donde sea necesario actuar más como un procesador digital de señal (DSP), con velocidades de reloj y consumo de energía muy altos.

Los microcontroladores están diseñados para reducir el coste económico y el consumo de energía de un sistema en particular. Por eso el tamaño de la unidad central de procesamiento, la cantidad de memoria y los periféricos incluidos dependerán de la aplicación. El control de un electrodoméstico sencillo como una batidora utilizará un procesador muy pequeño (4 u 8 bits) porque sustituirá a un autómata finito. En cambio, un reproductor de música o vídeo digital requerirá de un procesador de 32 bits o de 64 bits y de uno o más códecs de señal digital. El control de un sistema de frenos ABS se basa normalmente en un microcontrolador de 16 bits, al igual que el sistema de control electrónico del motor en un automóvil.

Los microcontroladores representan la inmensa mayoría de los chips de computadoras vendidos, sobre un 50% son controladores "simples" y el restante corresponde a DSP más especializados. Mientras se pueden tener uno o dos microprocesadores de propósito general en casa, se puede llegar a tener entre los electrodomésticos del hogar una o dos docenas de microcontroladores. Pueden encontrarse en casi cualquier dispositivo electrónico como automóviles, lavadoras, hornos microondas, teléfonos, etc.

2.2.1 Historia

El primer microprocesador fue el Intel 4004 de 4 bits, lanzado en 1971, seguido por el Intel 8008. Ambos procesadores requerían circuitos adicionales para implementar un sistema de trabajo y esto elevaba el coste del sistema total.

Esto dio lugar al primer microcontrolador, el TMS 1000, destinado a los sistemas embebidos, fue desarrollado en 1971 y comercializado en 1974. Combinaba memoria ROM, memoria RAM, microprocesador y reloj en un chip.

Poco más tarde Intel desarrolló el Intel 8048, un sistema de ordenador en un chip optimizado para aplicaciones de control, que comenzó a comercializarse en 1977. Combinaba memoria RAM y ROM en el mismo chip y aun puede encontrarse en más de mil millones de teclados de compatible IBM PC, y otras numerosas aplicaciones. Este microcontrolador fue uno los productos más exitosos en la historia de la compañía.

La mayoría de los microcontroladores en aquel momento tenían dos variantes. Unos tenían una memoria EPROM, reprogramable tras exponer a luz ultravioleta la tapa de cuarzo transparente, y otros menos caros que tenían la variante PROM, la cual sólo se programa una vez.

En 1993, el lanzamiento de la EEPROM en los microcontroladores permitió borrarla eléctrica y rápidamente, lo que permitió la creación rápida de prototipos y la programación en el sistema. El mismo año, Atmel lanzó el primer microcontrolador con memoria flash. Otras compañías siguieron el ejemplo rápidamente utilizando los dos tipos de memoria.

El coste se ha desplomado con el tiempo, encontrando microcontroladores de 8 bits por menos de 0,25 euros (comprados por miles) y de 32 bits con un precio cercano al euro/unidad. En la actualidad cualquier aficionado puede adquirir un microcontrolador fácilmente sin que le suponga un gran coste, y obtener infinita información en internet sobre él.

2.2.2 Arquitecturas de computadora

Básicamente existen dos arquitecturas de computadoras, y por supuesto, están presentes en el mundo de los microcontroladores: Von Neumann y Harvard. Ambas se diferencian en la forma de conexión de la memoria al procesador y en los buses que cada una necesita.

2.2.2.1 Arquitectura Von Neumann

La arquitectura Von Neumann utiliza el mismo dispositivo de almacenamiento tanto para las instrucciones como para los datos, siendo la que se utiliza en un ordenador personal porque permite ahorrar una buena cantidad de líneas de E/S, que son bastante costosas.

En un ordenador personal, cuando se carga un programa en memoria, a éste se le asigna un espacio de direcciones de la memoria que se divide en segmentos, de los cuales típicamente tendremos los siguientes: código (programa), datos y pila. Es por ello que podemos hablar de la memoria como un todo, aunque existan distintos dispositivos físicos en el sistema (disco duro, memoria RAM, memoria flash, unidad de disco óptico...).

En el caso de los microcontroladores, existen dos tipos de memoria bien definidas: memoria de datos (típicamente algún tipo de SRAM) y memoria de programas (ROM, PROM, EEPROM, flash u de otro tipo no volátil). En este caso la organización es distinta a las del ordenador personal, porque hay circuitos distintos para cada memoria y normalmente no se utilizan los registros de segmentos, sino que la memoria está segregada y el acceso a cada tipo de memoria depende de las instrucciones del procesador.

A pesar de que en los sistemas integrados con arquitectura Von Neumann la memoria esté segregada, y existan diferencias con respecto a la definición tradicional de esta arquitectura; los buses para acceder a ambos tipos de memoria son los mismos, del procesador solamente salen el bus de datos, el de direcciones, y el de control. Como conclusión, la arquitectura no ha sido alterada, porque la forma en que se conecta la memoria al procesador sigue el mismo principio definido en la arquitectura básica.

2.2.2.2 Arquitectura Harvard

La otra variante es la arquitectura Harvard, y por excelencia la utilizada en supercomputadoras, en los microcontroladores, y sistemas integrados en general. En este caso, además de la memoria, el procesador tiene los buses segregados, de modo que cada tipo de memoria tiene un bus de datos, uno de direcciones y uno de control.

La ventaja fundamental de esta arquitectura es que permite adecuar el tamaño de los buses a las características de cada tipo de memoria; además, el procesador puede acceder a cada una de ellas de forma simultánea, lo que se traduce en un aumento significativo de la velocidad de procesamiento. Típicamente los sistemas con esta arquitectura pueden ser dos veces más rápidos que sistemas similares con arquitectura Von Neumann.

2.2.3 Procesador

Ahora comenzaremos a ver cómo está hecho un procesador, no será una explicación detallada porque desde su invención ha tenido importantes revoluciones propias, pero hay aspectos básicos que no han cambiado y que constituyen la base de cualquier microprocesador.

2.2.3.1 Registros

Son un espacio de memoria muy reducido pero necesario para cualquier microprocesador, de aquí se toman los datos para varias operaciones que debe realizar el resto de los circuitos del procesador. Los registros sirven para almacenar los resultados de la ejecución de instrucciones, cargar datos desde la memoria externa o almacenarlos en ella.

Una parte de los registros, la destinada a los datos, es la que determina uno de los parámetros más importantes de cualquier microprocesador. Cuando escuchamos que un procesador es de 4, 8, 16, 32 o 64 bits, nos estamos refiriendo a procesadores que realizan sus operaciones con registros de datos de ese tamaño, y por supuesto, esto determina muchas de las potencialidades de estas máquinas.

Mientras mayor sea el número de bits de los registros de datos del procesador, mayores serán sus prestaciones, en cuanto a poder de cómputo y velocidad de ejecución, ya que este parámetro determina la potencia que se puede incorporar al resto de los componentes del sistema, por ejemplo, no tiene sentido tener una ALU de 16 bits en un procesador de 8 bits.

Por otro lado un procesador de 16 bits, puede que haga una suma de 16 bits en un solo ciclo de máquina, mientras que uno de 8 bits deberá ejecutar varias instrucciones antes de tener el resultado, aun cuando ambos procesadores tengan la misma velocidad de ejecución para sus instrucciones. El procesador de 16 bits será más rápido porque puede hacer el mismo tipo de tareas que uno de 8 bits, en menos tiempo.

2.2.3.2 Unidad de control

Esta unidad es de las más importantes en el procesador, en ella recae la lógica necesaria para la decodificación y ejecución de las instrucciones, el control de los registros, la ALU, los buses y cualquier cosa más que se quiera meter en el procesador.

La unidad de control es uno de los elementos fundamentales que determinan las prestaciones del procesador, ya que su tipo y estructura determina parámetros tales como el tipo de conjunto de instrucciones, velocidad de ejecución, tiempo del ciclo de máquina, tipo de buses que puede tener el sistema, manejo de interrupciones y un buen número de cosas más que en cualquier procesador van a parar a este bloque.

Por supuesto, las unidades de control son el elemento más complejo de un procesador y normalmente están divididas en unidades más pequeñas trabajando de conjunto. La unidad de control agrupa componentes tales como la unidad de decodificación, unidad de ejecución, controladores de memoria caché, controladores de buses, controlador de interrupciones, pipelines, entre otros elementos, dependiendo siempre del tipo de procesador.

2.2.3.3 Unidad aritmético-lógica (ALU)

Como los procesadores son circuitos que hacen básicamente operaciones lógicas y matemáticas, se le dedica a este proceso una unidad completa, con cierta independencia. Aquí es donde se realizan las sumas, restas, y operaciones lógicas típicas del álgebra de Boole.

Actualmente este tipo de unidades ha evolucionado mucho y los procesadores más modernos tienen varias ALU, especializadas en la realización de operaciones complejas como las operaciones en coma flotante. De hecho en muchos casos le han cambiado su nombre por el de "coprocesador matemático", aunque este es un término que surgió para dar nombre a un tipo especial de procesador que se conecta directamente al procesador más tradicional.

Su impacto en las prestaciones del procesador es también importante porque, dependiendo de su potencia, tareas más o menos complejas, pueden hacerse en tiempos muy cortos, como por ejemplo, los cálculos en coma flotante.

2.2.3.4 Buses

Son el medio de comunicación que utilizan los diferentes componentes del procesador para intercambiar información entre sí, eventualmente los buses o una parte de ellos estarán reflejados en los pines del encapsulado del procesador.

En el caso de los microcontroladores, no es común que los buses estén reflejados en el encapsulado del circuito, ya que estos se destinan básicamente a las E/S de propósito general y periféricos del sistema.

Existen tres tipos de buses:

- Dirección: Se utiliza para seleccionar al dispositivo con el cual se quiere trabajar o en el caso de las memorias, seleccionar el dato que se desea leer o escribir.
- Datos: Se utiliza para mover los datos entre los dispositivos de hardware (entrada y salida).
- Control: Se utiliza para gestionar los distintos procesos de escritura lectura y controlar la operación de los dispositivos del sistema.

2.2.3.5 Conjunto de instrucciones

Define las operaciones básicas que puede realizar el procesador, que conjugadas y organizadas forman lo que conocemos como software. El conjunto de instrucciones vienen siendo como las letras del alfabeto, el elemento básico del lenguaje, que organizadas adecuadamente permiten escribir palabras, oraciones y cuanto programa se le ocurra.

Existen dos tipos básicos de repertorios de instrucciones, que determinan la arquitectura del procesador: CISC y RISC.

CISC, del inglés Complex instruction set computing, Computadora de Conjunto de Instrucciones Complejo. Los microprocesadores CISC tienen un conjunto de instrucciones que se caracteriza por ser muy amplio y que permiten realizar operaciones complejas entre operandos situados en la memoria o en los registros internos. Este tipo de repertorio dificulta el paralelismo entre instrucciones, por lo que en la actualidad, la mayoría de los sistemas CISC de alto rendimiento convierten las instrucciones complejas en varias instrucciones simples del tipo RISC, llamadas generalmente microinstrucciones.

RISC, del inglés Reduced Instruction Set Computer, Computadora con Conjunto de Instrucciones Reducido. Se centra en la obtención de procesadores con las siguientes características fundamentales:

- Instrucciones de tamaño fijo.
- Pocas instrucciones.
- Sólo las instrucciones de carga y almacenamiento acceden a la memoria de datos.
- Número relativamente elevado de registros de propósito general.

Una de las características más destacables de este tipo de procesadores es que posibilitan el paralelismo en la ejecución, y reducen los accesos a memoria. Es por eso que los procesadores más modernos, tradicionalmente basados en arquitecturas CISC, implementan mecanismos de traducción de instrucciones CISC a RISC, para aprovechar las ventajas de este tipo de procesadores.

2.2.4 Memoria

Anteriormente se ha visto que la memoria en los microcontroladores debe estar ubicada dentro del mismo encapsulado, esto es así la mayoría de las veces, porque la idea fundamental es mantener el grueso de los circuitos del sistema dentro de un solo integrado.

En los microcontroladores la memoria no es abundante, aquí no encontrará Gigabytes de memoria como en las computadoras personales. Típicamente la memoria de programas no excederá de 16 K-localizaciones de memoria no volátil (flash o eprom) para contener los programas.

La memoria RAM está destinada al almacenamiento de información temporal que será utilizada por el procesador para realizar cálculos u otro tipo de operaciones lógicas. En el espacio de direcciones de memoria RAM se ubican además los registros de trabajo del procesador y los de configuración y trabajo de los distintos periféricos del microcontrolador. Es por ello que en la mayoría de los casos, aunque se tenga un espacio de direcciones de un tamaño determinado, la cantidad de memoria RAM de que dispone el programador para almacenar sus datos es menor que la que puede direccionar el procesador.

El tipo de memoria utilizada en las memorias RAM de los microcontroladores es SRAM, lo que evita tener que implementar sistemas de refresco. A pesar de que la memoria SRAM es más costosa que la DRAM, es el tipo adecuado para los microcontroladores porque éstos poseen pequeñas cantidades de memoria RAM.
En el caso de la memoria de programas se utilizan diferentes tecnologías, y el uso de una u otra depende de las características de la aplicación a desarrollar, a continuación se describen las cinco tecnologías existentes, que mayor utilización tienen o han tenido:

- Mascara ROM. En este caso no se "graba" el programa en memoria sino que el microcontrolador se fabrica con el programa, es un proceso similar al de producción de los CD comerciales mediante masterización. El costo inicial de producir un circuito de este tipo es alto, porque el diseño y producción de la máscara es un proceso costoso, sin embargo, cuando se necesitan varios miles o incluso cientos de miles de microcontroladores para una aplicación determinada, como por ejemplo, algún electrodoméstico, el costo inicial de producción de la máscara y el de fabricación del circuito se distribuye entre todos los circuitos de la serie, y el costo final de ésta es bastante menor que el de sus semejantes con otro tipo de memoria.

- Memoria PROM (Programmable Read-Only Memory) también conocida como OTP (One Time Programmable). Este tipo de memoria también es conocida como PROM o simplemente ROM. Los microcontroladores con memoria OTP se pueden programar una sola vez, con algún tipo de programador. Se utilizan en sistemas donde el programa no requiera futuras actualizaciones y para series relativamente pequeñas, donde la variante de máscara sea muy costosa.

- Memoria EPROM (Erasable Programmable Read Only Memory). Los microcontroladores con este tipo de memoria son muy fáciles de identificar porque su encapsulado es de cerámica y llevan encima una ventanita de vidrio desde la cual puede verse la oblea de silicio del microcontrolador. Se fabrican así porque la memoria EPROM es reprogramable, pero antes debe borrarse, y para ello hay que exponerla a una fuente de luz ultravioleta, el proceso de grabación es similar al empleado para las memorias OTP. Al aparecer tecnologías menos costosas y más flexibles, como las memorias EEPROM y FLASH, este tipo de memoria han caído en desuso, se utilizaban en sistemas que requieren actualizaciones del programa y para los procesos de desarrollo y puesta a punto.

- EEPROM (Electrical Erasable Programmable Read Only Memory). Fueron el sustituto natural de las memorias EPROM, la diferencia fundamental es que pueden ser borradas eléctricamente, por lo que la ventanilla de cristal de cuarzo y los encapsulados cerámicos no son necesarios. Al disminuir los costos de los encapsulados, los microcontroladores con este tipo de memoria se hicieron más baratos y cómodos para trabajar que sus equivalentes con memoria EPROM. Otra característica destacable de este tipo de microcontrolador es que fue en ellos donde comenzaron a utilizarse los sistemas de programación en el sistema que evitan tener que sacar el microcontrolador de la tarjeta que lo aloja para hacer actualizaciones al programa.

- Memoria flash. En el campo de las memorias reprogramables para microcontroladores, son el último avance tecnológico en uso a gran escala, y han sustituido a los microcontroladores con memoria EEPROM. A las ventajas de las memorias flash se le adicionan su gran densidad respecto a sus predecesoras lo que permite incrementar la cantidad de memoria de programas a un costo muy bajo. Pueden además ser programadas con las mismas tensiones de alimentación del microcontrolador, el acceso en lectura y la velocidad de programación es superior, disminución de los costos de producción, entre otras.

Lo más habitual es encontrar que la memoria de programas y datos está ubicada toda dentro del microcontrolador, de hecho, actualmente son pocos los microcontroladores que permiten conectar memoria de programas en el exterior del encapsulado. Las razones para estas "limitaciones" están dadas porque el objetivo fundamental es obtener la mayor integración posible y conectar memorias externas consume líneas de E/S que son uno de los recursos más preciados de los microcontroladores.

Cuando se requiere aumentar la cantidad de memoria de datos, lo más frecuente es colocar dispositivos de memoria externa en forma de periféricos, de esta forma se pueden utilizar memorias RAM, FLASH o incluso discos duros como los de los ordenadores personales, mientras que para los cálculos y demás operaciones que requieran almacenamiento temporal de datos se utiliza la memoria RAM interna del

microcontrolador. Esta forma de expandir la memoria de datos está determinada, en la mayoría de los casos, por el tipo de repertorio de instrucciones del procesador y porque permite un elevado número de configuraciones distintas, además del consiguiente ahorro de líneas de E/S que se logra con el uso de memorias con buses de comunicación serie.

2.2.5 Interrupciones

Las interrupciones son esencialmente llamadas a subrutina generadas por los dispositivos físicos, al contrario de las subrutinas normales de un programa en ejecución. Como el salto de subrutina no es parte del hilo o secuencia de ejecución programada, el controlador guarda el estado del procesador en la pila de memoria y entra a ejecutar un código especial llamado "manejador de interrupciones" que atiende al periférico específico que generó la interrupción. Al terminar la rutina, una instrucción especial le indica al procesador el fin de la atención de la interrupción. En ese momento el controlador restablece el estado anterior, y el programa que se estaba ejecutando antes de la interrupción sigue como si nada hubiese pasado. Las rutinas de atención de interrupciones deben ser lo más breves posibles para que el rendimiento del sistema sea satisfactorio, porque normalmente cuando una interrupción es atendida, todas las demás interrupciones están en espera.

Los procesos de atención a interrupciones tienen la ventaja de que se implementan por hardware ubicado en el procesador, así que es un método rápido de hacer que el procesador se dedique a ejecutar un programa especial para atender eventos que no pueden esperar por mecanismos lentos.

En términos generales, un proceso de interrupción y su atención por parte del procesador, tiene la siguiente secuencia de acciones:

1. En el mundo real se produce el evento para el cual queremos que el procesador ejecute un programa especial, este proceso tiene la característica de que no puede esperar mucho tiempo antes de ser atendido o no sabemos en qué momento debe ser atendido.
2. El circuito encargado de detectar la ocurrencia del evento se activa, y como consecuencia, activa la entrada de interrupción del procesador.
3. La unidad de control detecta que se ha producido una interrupción y "levanta" una bandera para registrar esta situación; de esta forma si las condiciones que provocaron el evento desaparecen y el circuito encargado de detectarlo desactiva la entrada de interrupción del procesador, ésta se producirá de cualquier modo, porque ha sido registrada.
4. La unidad de ejecución termina con la instrucción en curso y justo antes de comenzar a ejecutar la siguiente comprueba que se ha registrado una interrupción

5. Se desencadena un proceso que permite guardar el estado actual del programa en ejecución y saltar a una dirección especial de memoria de programas, donde está la primera instrucción de la subrutina de atención a interrupción.
6. Se ejecuta el código de atención a interrupción, esta es la parte "consciente" de todo el proceso porque es donde se realizan las acciones propias de la atención a la interrupción y el programador juega su papel.
7. Cuando en la subrutina de atención a interrupción se ejecuta la instrucción de retorno, se desencadena el proceso de restauración del procesador al estado en que estaba antes de la atención a la interrupción.

Como podemos observar, el mecanismo de interrupción es bastante complicado, sin embargo tiene dos ventajas que obligan a su implementación: la velocidad y su capacidad de ser asíncrono.

Los mecanismos de interrupción no solo se utilizan para atender eventos ligados a procesos que requieren atención inmediata sino que se utilizan además para atender eventos de procesos asíncronos.

Las interrupciones son tan eficaces que permiten que el procesador actúe como si estuviese haciendo varias cosas a la vez cuando en realidad se dedica a la misma rutina de siempre, ejecutar instrucciones una detrás de la otra.

2.2.6 Periféricos

A continuación describiremos algunos de los periféricos que con mayor frecuencia encontraremos en los microcontroladores.

2.2.6.1 Entradas y salidas de propósito general

También conocidos como puertos de E/S, generalmente agrupadas en puertos de 8 bits de longitud, permiten leer datos del exterior o escribir en ellos desde el interior del microcontrolador, el destino habitual es el trabajo con dispositivos simples como relés, LED, o cualquier otra cosa que se le ocurra al programador.

Algunos puertos de E/S tienen características especiales que le permiten manejar salidas con determinados requerimientos de corriente, o incorporan mecanismos especiales de interrupción para el procesador.

Típicamente cualquier pin de E/S puede ser considerada E/S de propósito general, pero como los microcontroladores no pueden tener infinitos pines, ni siquiera todos los pines que queramos, las E/S de propósito general comparten los pines con otros periféricos. Para usar un pin con cualquiera de las características a él asignadas debemos configurarlo mediante los registros destinados a ellos.

2.2.6.2 Temporizadores y contadores

Son circuitos sincrónicos para el conteo de pulsos que permite obtener medidas de tiempo muy precisas. Si la fuente de pulsos es el oscilador interno del microcontrolador es común que no tengan un pin asociado, y en este caso trabaja como temporizador. Por otra parte, cuando la fuente es externa, entonces tienen asociado un pin configurado como entrada, este es el modo contador.

Los temporizadores son uno de los periféricos más habituales en los microcontroladores y se utilizan para muchas tareas, como por ejemplo, la medición de frecuencia, implementación de relojes, para el trabajo de conjunto con otros periféricos que requieren una base estable de tiempo entre otras funcionalidades. Es frecuente que un microcontrolador típico incorpore más de un temporizador/contador. Este periférico es un elemento casi imprescindible y es habitual que tengan asociada alguna interrupción.

Los tamaños típicos de los registros de conteo son 8 y 16 bits, pudiendo encontrar dispositivos que solo tienen temporizadores de un tamaño o con ambos tipos de registro de conteo.

2.2.6.3 Conversor analógico/digital

Como es muy habitual en el trabajo con señales analógicas, éstas deben ser convertidas a digital y por ello muchos microcontroladores incorporan un conversor analógico-digital, el cual se utiliza para tomar datos de varias entradas diferentes que se seleccionan mediante un multiplexor.

Las resoluciones más frecuentes son 8 y 10 bits, que son suficientes para aplicaciones sencillas. Para aplicaciones en control e instrumentación están disponibles resoluciones de 12bit, 16bit y 24bit. También es posible conectar un convertidor externo.

2.2.6.4 Puertos de comunicación

- Puerto serie: Este periférico está presente en casi cualquier microcontrolador, normalmente en forma de UART (Universal Asynchronous Receiver Transmitter) o USART (Universal Synchronous Asynchronous Receiver Transmitter) dependiendo de si permiten o no el modo sincrónico de comunicación. La finalidad de este periférico es la comunicación con otro microcontrolador o con un PC y en la mayoría de los casos hay que agregar circuitos externos para completar la interfaz de comunicación.

- SPI: Este tipo de periférico se utiliza para comunicar al microcontrolador con otros microcontroladores o con periféricos externos conectados a él, por medio de una interfaz muy sencilla. Hay solo un nodo controlador que permite iniciar cualquier transacción, lo cual es una desventaja en sistemas complejos, pero su

sencillez permite el aislamiento galvánico de forma directa por medio de optoacopladores.

- I2C: Cumple las mismas funciones que el SPI, pero requiere menos señales de comunicación y cualquier nodo puede iniciar una transacción. Es muy utilizado para conectar las tarjetas gráficas de las computadoras personales con los monitores, para que estos últimos informen de sus prestaciones y permitir la autoconfiguración del sistema de vídeo.

- USB: Los microcontroladores son los que han permitido la existencia de este sistema de comunicación. Es un sistema que trabaja por polling (monitorización) de un conjunto de periféricos inteligentes por parte de un amo, que es normalmente un computador personal. Cada modo inteligente está gobernado inevitablemente por un microcontrolador.

- Ethernet: Es el sistema más extendido en el mundo para redes de área local cableadas. Muchos de los enrutadores caseros de pequeñas empresas están construidos sobre la base de un microcontrolador que hace del cerebro del sistema.

- Can: Este protocolo es del tipo CSMA/CD con tolerancia a elevados niveles de tensión de modo común y orientado al tiempo real. Este protocolo es el estándar más importante en la industria automotriz (OBD). También se usa como capa física del "field bus" para el control industrial.

2.2.6.5 Comparadores

Son circuitos analógicos basados en amplificadores operacionales que tienen la característica de comparar dos señales analógicas y dar como salida los niveles lógicos '0' o '1' en dependencia del resultado de la comparación. Es un periférico muy útil para detectar cambios en señales de entrada de las que solamente nos interesa conocer cuando está en un rango determinado de tensión.

2.2.6.6 Modulador de ancho de pulsos

Los PWM (Pulse Width Modulator) son periféricos muy útiles sobre todo para el control de motores, sin embargo hay un grupo de aplicaciones que pueden realizarse con este periférico, dentro de las cuales podemos citar: inversión DC/AC para UPS, conversión digital analógica D/A, control regulado de luz (dimming) entre otras.

2.2.6.7 Memoria de datos no volátil

Muchos microcontroladores han incorporado estos tipos de memoria como un periférico más, para el almacenamiento de datos de configuración o de los procesos que se

controlan. Esta memoria es independiente de la memoria de datos tipo RAM o la memoria de programas, en la que se almacena el código del programa a ejecutar por el procesador del microcontrolador.

Muchos de los microcontroladores PIC incluyen este tipo de memoria, típicamente en forma de memoria EEPROM, incluso algunos de ellos permiten utilizar parte de la memoria de programas como memoria de datos no volátil, por lo que el procesador tiene la capacidad de escribir en la memoria de programas como si ésta fuese un periférico más.

2.3 El bus I2C [6][7]

El I²C está diseñado como un bus maestro-esclavo. La transferencia de datos es siempre inicializada por un maestro y el esclavo reacciona.

2.3.1 Definición eléctrica

En el diagrama inferior se encuentran representados tres dispositivos. El I²C precisa de dos líneas de señal: reloj (CLK, Serial Clock) y la línea de datos (SDA, Serial Data). Ambas líneas precisan resistencias de pull-up hacia VDD. El nivel alto debe ser de al menos 0,7 x VDD y el nivel bajo no debe ser más de 0,3 x VDD. El Bus I²C trabaja con lógica positiva, esto quiere decir que un nivel alto en la línea de datos corresponde a un 1 lógico, el nivel bajo a un 0.

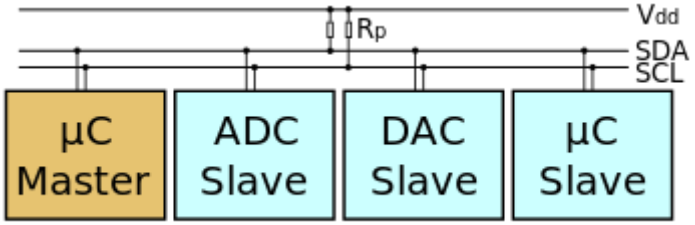


Ilustración 3 Ejemplo esquema bus I2C

Un ejemplo esquemático con un maestro (un microcontrolador) y tres nodos esclavos (un ADC, un DAC, y otro microcontrolador) con resistencias pull-up R_p

2.3.2 Pulso y estado del bus

La señal de reloj siempre es generada por el maestro. Para cada modo especificado, está predeterminado un pulso de reloj máximo permitido. En general, también pueden ser utilizadas señales de reloj más lentas, siempre y cuando sean compatibles con la interfaz del maestro. Sin embargo, algunos circuitos integrados requieren una frecuencia mínima con el fin de funcionar correctamente. En la tabla siguiente se muestran los porcentajes máximos permisibles de reloj.

Modo	Velocidad de transmisión máxima	Dirección
Standard Mode (Sm)	0,1 Mbit/s	Bidireccional
Fast Mode (Fm)	0,4 Mbit/s	Bidireccional
Fast Mode Plus (Fm+)	1,0 Mbit/s	Bidireccional
High Speed Mode (Hs-mode)	3,4 Mbit/s	Bidireccional
Ultra Fast-mode (UFm)	5,0 Mbit/s	Unidireccional

Si el esclavo necesita más tiempo que el dictado por el reloj del maestro, puede mantener, entre la transferencia de bytes individuales, la señal de reloj en nivel bajo para frenar de este modo al maestro.

Los datos (bits individuales) sólo son válidos si su nivel lógico no cambia durante una fase de reloj alta. Las excepciones son el inicio, la parada, y la señal de inicio repetida o reset. La señal de arranque es un flanco descendente en SDA mientras SCL se encuentra en nivel alto. La señal de parada es un flanco ascendente en SDA mientras SCL está en nivel alto. La señal de reset se comporta de igual manera que la señal de inicio.

Una unidad de datos consta de 8 bits, que pueden ser interpretados como un valor o como una dirección, y un bit de confirmación ACK (Acknowledge).

2.3.3 Direccionamiento

La dirección de I²C estándar es el primer byte enviado por el maestro, los primeros 7 bits representan la dirección y el octavo bit (R/W-Bit) es el que comunica al esclavo si debe recibir datos del maestro o enviar datos al maestro. Por lo tanto, I²C utiliza un espacio de direccionamiento de 7 bits, lo cual permite hasta 112 nodos en un bus (16 de las 128 direcciones posibles están reservadas para fines especiales).

Cada uno de los circuitos integrados con capacidad de soportar un I²C tiene una dirección predeterminada por el fabricante, de la cual los últimos tres bits (subdirección) pueden ser fijados por tres pines de control. En este caso, pueden funcionar en un I²C hasta 8 circuitos integrados.

Debido a la escasez de direcciones, se introdujo más tarde un direccionamiento de 10 bits. Es compatible con el estándar de 7 bits mediante el uso de 4 de las 16 direcciones reservadas. Ambos modos de direccionamiento pueden utilizarse simultáneamente.

2.3.4 Protocolo de transferencia

El inicio de una transmisión es indicado por la señal de inicio del maestro, seguido de la dirección. Ésta es confirmada por el ACK-Bit del esclavo correspondiente. En función del R/W-Bit se escriben bytes de datos (datos al esclavo) o se leen (datos al maestro). El ACK es enviado desde el esclavo al escribir, y desde el maestro al leer. Una transmisión es finalizada por la señal de parada. Como alternativa, puede ser enviada una señal de reset al arranque de una nueva transmisión, sin necesidad de parar la transmisión anterior con una señal de parada.

Todos los bytes son transferidos de esta manera como "Most Significant Bit First" (bit más significativo primero).

2.3.5 Uso

Una de las propiedades del I²C es el hecho de que un microcontrolador puede controlar toda una red de circuitos integrados con sólo dos I/O-Pins (Input/Output) y un software muy simple. Esto se traduce en una reducción significativa del precio ya que el coste de un circuito integrado y la placa de circuito dependen del tamaño de la carcasa y del número de pines. Una carcasa grande tiene más pines, necesita más espacio en la placa de circuito y tiene más conexiones que podrían fallar. Todo esto aumenta los costes de desarrollo, producción y pruebas.

Aunque es más lento que los sistemas de bus más nuevos, I²C es beneficioso (debido al bajo coste) para los sistemas periféricos que no necesitan ser rápidos. A menudo es usado para la transmisión de datos de control y configuración, por ejemplo para control de volumen, conversor de señal analógica-digital o digital-analógica con baja tasa de frecuencia de muestreo, relojes a tiempo real, pequeños espacios de memoria o conmutadores bidireccionales y multiplexores. Incluso los sensores electrónicos integran con frecuencia un convertidor analógico-digital con un I²C.

2.3.6 Estabilidad

El protocolo del I²C es muy susceptible a las interferencias. Este hecho limita su uso únicamente a entornos de poca interferencia, en los cuales no se ha de esperar ningún tipo de ruido, problemas de compatibilidad electromagnética o diafonías.

2.4 Comunicación One Wire [8][9]

1-Wire es un protocolo de comunicaciones en serie basado en un bus, un maestro y varios esclavos de una sola línea de datos en la que se alimentan. Por supuesto, necesita una referencia a tierra común a todos los dispositivos.

Los dispositivos de red 1-Wire poseen un único pin de datos al que se conecta una resistencia "Pull Up" anclada a +5V DC (nominal). Una de las características de la tecnología 1-Wire es que cada dispositivo esclavo tiene una identificación única grabada en su memoria ROM al momento de su fabricación

El BUS 1-Wire, permite realizar una comunicación serial asincrónica entre un dispositivo maestro y uno o varios dispositivos esclavos, utilizando un único pin de E/S del microcontrolador.

2.4.1 Características

- Utiliza niveles de alimentación CMOS/TTL con un rango de operación que abarca desde 2.8V hasta 6V.
- Tanto el maestro como los esclavos transmiten información de forma bidireccional pero sólo en una dirección a la vez.
- Toda la información es leída o escrita comenzando por el bit menos significativo (LSB).
- No se requiere del uso de una señal de reloj, ya que, cada dispositivo posee un oscilador interno que se sincroniza con el del maestro cada vez que en la línea de datos aparezca un flanco de bajada.
- La alimentación de los esclavos se puede hacer utilizando el voltaje propio del BUS. Para ello, cada circuito esclavo posee un rectificador de media onda y un condensador, durante los períodos en los cuales no se efectúa ninguna comunicación, la línea de datos se encuentra en estado alto debido a la resistencia de "Pull Up"; en esa condición, el diodo entra en conducción y carga al condensador. Cuando el voltaje de la red cae por debajo de la tensión del condensador, el diodo se polariza en inverso evitando que el condensador se descargue. La carga almacenada en el condensador alimentará al circuito esclavo.
- Todas las tensiones mayores que 2,2 voltios son consideradas un 1 lógico mientras que un 0 lógico son aquellas menores o iguales a 0,8.
- La transferencia de información es a 16.3Kbps en modo "Standard" y hasta 142Kbps en modo "overdrive".

2.4.2 Envío y recepción de datos

Para enviar un bit a "1" el maestro se lleva a 0 voltios la línea de datos durante 1-15 microsegundos. Para enviar un bit a "0" el maestro se lleva a 0 voltios la línea de datos durante 60 microsegundos.

Los dispositivos esclavos leen el bit aproximadamente a los 30 microsegundos después del flanco de bajada de cada bit.

Cuando el maestro lee los datos del dispositivo esclavo pone 0 voltios durante 1-15 microsegundos en la línea de datos y a partir de ese momento el esclavo no hace nada (la señal se mantiene en 5 voltios) si quiere enviar un 1 lógico o mantiene la señal en 0 voltios hasta los 60 microsegundos si quiere enviar un 0 lógico.

Los datos se envían o reciben en grupos de 8 bits. Para iniciar una comunicación se reinicia el bus. El protocolo puede incluir detección de errores transmitiendo códigos de detección de errores (CRC).

Como en el bus puede haber muchos dispositivos el protocolo incluye el direccionamiento de los mismos empleando un código único de 64 bits de los cuales el byte más significativo indica el tipo de dispositivo, y el último es un código de detección de errores (CRC) de 8 bits.

Los comandos que pueden interpretar los dispositivos esclavos dependerán de estos.
Para encontrar los dispositivos presentes en el bus el maestro puede enviar un comando de enumeración que responderán todos los dispositivos.

2.5 Bus SPI [11][12]

El Bus SPI (Serial Peripheral Interface) es un estándar de comunicaciones, usado principalmente para la transferencia de información entre circuitos integrados en equipos electrónicos. El bus de interfaz de periféricos serie o bus SPI es un estándar para controlar casi cualquier dispositivo electrónico digital que acepte un flujo de bits serie regulado por un reloj (comunicación sincrónica).

Incluye una línea de reloj, dato entrante, dato saliente y un pin de chip select, que conecta o desconecta la operación del dispositivo con el que uno desea comunicarse. De esta forma, este estándar permite multiplexar las líneas de reloj.

Muchos sistemas digitales tienen periféricos que necesitan existir pero no ser rápidos. La ventajas de un bus serie es que minimiza el número de conductores, pines y el tamaño del circuito integrado. Esto reduce el coste de fabricar, montar y probar la electrónica.

2.5.1 Operación

El SPI es un protocolo síncrono. La sincronización y la transmisión de datos se realizan por medio de 4 señales:

- SCLK (Clock): Es el pulso que marca la sincronización. Con cada pulso de este reloj, se lee o se envía un bit.
- MOSI (Master Output Slave Input): Salida de datos del Master y entrada de datos al Slave.
- MISO (Master Input Slave Output): Salida de datos del Slave y entrada al Master.
- SS/Select: Para seleccionar un Slave, o para que el Master le diga al Slave que se active.

La Cadena de bits es enviada de manera síncrona con los pulsos del reloj, es decir con cada pulso, el Master envía un bit. Para que empiece la transmisión el Master baja la señal SSTE o SS/Select a cero, con esto el Slave se activa y empieza la transmisión, con un pulso de reloj al mismo tiempo que el primer bit es leído. Cabe destacar que los pulsos de reloj pueden estar programados de manera que la transmisión del bit se realice en 4 modos diferentes, a esto se llama polaridad y fase de la transmisión:

- 1. Con el flanco de subida sin retraso.
- 2. Con el flanco de subida con retraso.
- 3. Con el flanco de bajada sin retraso.
- 4. Con el flanco de bajada con retraso.

2.5.2 Pros y contras del bus SPI

Ventajas

- Comunicación Full Dúplex
- Mayor velocidad de transmisión que con I²C o SMBus
- Protocolo flexible en que se puede tener un control absoluto sobre los bits transmitidos
    - No está limitado a la transferencia de bloques de 8 bits
    - Elección del tamaño de la trama de bits, de su significado y propósito
- Su implementación en hardware es extremadamente simple
    - Consume menos energía que I²C o que SMBus debido que posee menos circuitos (incluyendo las resistencias pull-up) y estos son más simples
    - No es necesario arbitraje o mecanismo de respuesta ante fallos
    - Los dispositivos clientes usan el reloj que envía el servidor, no necesitan por tanto su propio reloj

  - No es obligatorio implementar un transceptor (emisor y receptor), un dispositivo conectado puede configurarse para que solo envíe, sólo reciba o ambas cosas a la vez

- Usa mucho menos terminales en cada chip/conector que una interfaz paralelo equivalente
- Como mucho una única señal específica para cada cliente (señal SS), las demás señales pueden ser compartidas

Desventajas

- Consume más pines de cada chip que I²C, incluso en la variante de 3 hilos
- El direccionamiento se hace mediante líneas específicas (señalización fuera de banda) a diferencia de lo que ocurre en I²C que se selecciona cada chip mediante una dirección de 7 bits que se envía por las mismas líneas del bus
- No hay control de flujo por hardware
- No hay señal de asentimiento. El servidor podría estar enviando información sin que estuviese conectado ningún cliente y no se daría cuenta de nada
- No permite fácilmente tener varios servidores conectados al bus
- Sólo funciona en las distancias cortas a diferencia de, por ejemplo, RS-232, RS-485, o Bus CAN

2.6 Elección del microcontrolador

Para este proyecto se utilizará un microcontrolador de la línea Atmel, presente en la plataforma de desarrollo Arduino. Dentro de esta plataforma podemos encontrar distintas placas con las que trabajar, de las cuales solo elegiremos una de ellas.

En la siguiente tabla se observan las características de las placas Arduino más populares:

Característica de Arduino	**UNO**	**Mega 2560**	**Leonardo**	**DUE**
Tipo de microcontrolador	Atmega 328	Atmega 2560	Atmega 32U4	AT91SAM3X8E
Velocidad de reloj	16 MHz	16 MHz	16 MHz	84 MHz
Pines digitales de E/S	14	54	20	54
Entradas analógicas	6	16	12	12

Salidas analógicas	0	0	0	2 (DAC)
Memoria de programa (Flash)	32 Kb	256 Kb	32 Kb	512 Kb
Memoria de datos (SRAM)	2 Kb	8 Kb	2.5 Kb	96 Kb
Memoria auxiliar (EEPROM)	1 Kb	4 Kb	1 Kb	0 Kb
Característica de Arduino	**UNO**	**Mega 2560**	**Leonardo**	**DUE**

En nuestro caso el microcontrolador utilizado será es el Atmega2560, el cual podemos encontrar en la placa Arduino Mega2560, placa superior a la utilizada en prácticas de Sistemas embebidos, Arduino Uno, y con la que obtuvimos muy buenos resultados. La elección de esta plataforma se basa principalmente en que Arduino es potente, simple, y conlleva poco tiempo de desarrollo.

La potencia de desarrollo de Arduino podemos explicarla en 3 características fundamentales: [14]

- El Hardware prearmado alrededor del Microcontrolador proporciona una interface de puertos y periféricos ligados al microcontrolador Atmega.

- Las librerías de Arduino cuya potencia permite el desarrollo de múltiples aplicaciones de manera simple y rápida.

- Los Shields de Arduino que simplifican aún más la tarea del desarrollo de las aplicaciones puesto que existe un Shield (módulo de hardware especial) a la medida de las aplicaciones. Estos Shields prevén la adaptabilidad de señales para acondicionamiento de los sensores, Drivers de corriente, regulación de alimentación, y sobre todo la funcionalidad para lo que fue construido.

3. Desarrollo

A continuación se describirá como se diseñó el sistema paso a paso, desde la primera idea hasta el sistema final obtenido al que se le ha denominado como "Akuatización".

3.1 Microcontrolador y RTC

Como la base del proyecto es el control de tiempos para el encendido y apagado de ciertos dispositivos se ha utilizado un RTC con el que poder sincronizar el microcontrolador y que se mantenga en hora en caso de un corte en la alimentación, lo que se consigue con la batería externa de 3V para el RTC. De esta forma evitamos tener que volver a programar el microcontrolador estableciendo la hora correspondiente para retomar el control del acuario.

La sincronización se lleva a cabo mediante comunicación I2C utilizando la librería Wire, primero establecemos la hora y seguidamente le indicamos al microcontrolador que adquiera la hora del RTC. Aquí se muestran dos de los comandos principales, disponibles en la librería DS1307RTC, para realizar la sincronización.

RTC.set (t);
SetSyncProvider (RTC.get);

Los dispositivos utilizados son la placa de Funduino Mega2560, copia económica del Arduino Mega 2560, placa que se comentó en un principio, y el RTC.

3.1.1 Funduino Mega2560

Ilustración 4 Funduino Mega 2560

3.1.1.1 Especificaciones técnicas [13]

Microcontrolador	**Atmega2560**
Tensión de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límite)	6-20V
Digital pines I / O	54 (de los cuales 15 proporcionan salida PWM)
Pines de entrada analógica	16
Corriente DC por Pin I / O	20 mA
Corriente CC para Pin 3.3V	50 mA
Memoria flash	256 KB de los cuales 8 KB utilizado por cargador de arranque
SRAM	8 KB
EEPROM	4 KB
Velocidad de reloj	16 MHz
LED_BUILTIN	13
Longitud	101.52 mm
Anchura	53,3 mm
Peso	37 g

3.1.1.2 Resumen entradas y salidas

Cada uno de los 54 pines digitales pueden usarse como una entrada o como una salida utilizando las funciones pinMode(), digitalWrite(), y digitalRead(). Operan a 5 voltios, tienen una resistencia pull-up interna (desconectado por defecto) de 20-50 kohm y pueden proporcionar o recibir 20 mA. Siempre se debe tener cuidado de no superar el máximo de 40 mA para evitar daños permanentes al microcontrolador.

Además, algunos pines tienen funciones especializadas que serán de utilidad en este proyecto o en posibles ampliaciones:

I2C: SDA y SCL. Comunicación soportada utilizando la librería Wire.

Serie: Se utiliza para recibir (RX) y transmitir datos en serie (TX). Los pines 0 y 1 están conectados al chip de serie ATmega16U2 USB-a-TTL que actúa como un puente entre el puerto USB del ordenador y el puerto serie del procesador principal.

PWM: Proporciona una salida de PWM de 8 bits mediante la función analogWrite ().

SPI: MISO, MOSI, SCK y SS. Soportan la comunicación SPI utilizando la librería SPI.

Interrupciones externas: Estos pines se pueden configurar para desencadenar una interrupción en un nivel bajo, un flanco o un cambio de nivel.

El Mega 2560 tiene un extra de 16 entradas analógicas con 10 bits de resolución (es decir, 1024 valores diferentes). Por defecto se miden desde "tierra" a 5 voltios, aunque es posible cambiar el límite superior usando el pin AREF, voltaje de referencia para las entradas analógicas, y la función analogReference ().

Para más información acerca de las entradas se puede consultar el pinout de Funduino en anexos.

3.1.2 RTC I2C Tiny DS3231 AT24C32

Ilustración 5 Módulo RTC I2C DS3231

Este reloj de tiempo real (RTC) es un dispositivo electrónico que permite obtener mediciones de tiempo en las unidades temporales que empleamos de forma cotidiana y acceder a ellas a través de la interfaz I2C, pines 20 (SDA) y 21 (SCL) de Funduino.

Su gran ventaja es que dispone de un circuito de control de la fuente de alimentación que permite detectar automáticamente la fuente y gestionar la alimentación principal y de reserva (es decir, la batería de bajo voltaje) para cambiar entre ambas, de forma que si falla la alimentación principal, el dispositivo puede continuar proporcionando la hora exacta y la temperatura en el caso de desearlo, el rendimiento no se ve afectado.

3.1.2.1 Características

- Reloj de tiempo real I2C muy preciso, con un oscilador integrado con compensación de temperatura del cristal para evitar desfases acumulados.
- Interfaz de bus IIC, con velocidad máxima de transmisión de 400 KHz
- El dispositivo incorpora una entrada de batería para que en caso de desconectar la fuente de alimentación principal se mantenga el cronometraje exacto.
- Oscilador integrado para mejorar la precisión a largo plazo del dispositivo.
- Disponible en rangos de temperaturas comerciales e industriales.
- RTC mantiene segundos, minutos, horas, días, fecha, mes, año.
- Chip de memoria: AT24C32 (capacidad de almacenamiento de 32 K)
- Voltaje de alimentación entre 2.5V y 5.5V, se alimentará a 5V.

3.2 Relés y RTC

Una vez sincronizado el microcontrolador con el RTC para la obtención de la hora se pasó a comprobar el funcionamiento de los relés con diferentes alarmas de forma que conmutasen periódicamente con un intervalo de tiempo establecido o bien que se apagasen o encendiesen a unas horas concretas. Para realizar esta tarea se utilizaron diferentes librerías entre ellas TimeAlarms, una librería que nos permite configurar múltiples alarmas de forma sencilla y que se activan o desactivan por polling.

3.2.1 Módulo de relés de 4 canales con aislamiento por optoacoplador

3.2.1.1 Características

- Puerto de control: 5V
- Punto de contacto máximo de salida de relé: 250V / 10A
- Las entradas IN1 / IN2 / IN3 / IN4 activas a nivel bajo.
- Puerto de entrada de alimentación individual JD-VCC para relé.
- Aplicación: Área industrial, control del PLC, control inteligente casero

Las salidas de la placa Funduino son perfectamente útiles para controlar cargas que no consuman demasiada corriente, como un Led, pero son insuficientes para cargas mayores como una lámpara o un motor que se alimentan de 220 voltios. Para solucionar esto se emplea el módulo de relés visualizado en la imagen inferior. Este módulo de 4 relés que funcionan a 5 Voltios es capaz de manejar cargas de hasta 10 Amperios en 250 Voltios y están convenientemente aislados mediante optoacopladores en las entradas, las cuales cuentan con leds individuales que sirven como indicadores de estado.

Los distintos componentes del módulo pueden verse en la siguiente imagen

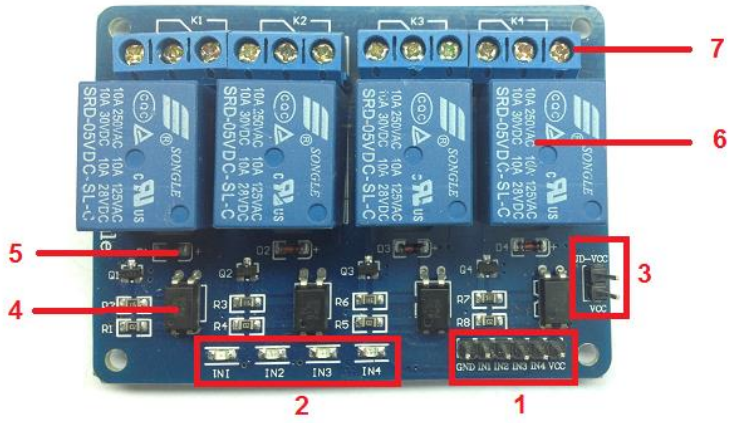


Ilustración 6 Módulo 4 relés 10A/250V

Como se puede apreciar, la placa tiene un conector de entradas (IN1 a IN4) y alimentación (GND es masa o negativo y Vcc es el positivo) (1), cuatro leds que indican el estado de la entradas (2), un jumper selector para la alimentación de los relés (3), cuatro optoacopladores (4), cuatro diodos de protección (5), cuatro relés con bobinas de 5V y contactos capaces de controlar hasta 10 Amperes en una tensión de 250V (6) y cuatro borneras, con tres contactos cada una (Común, Normal abierto y Normal cerrado), para las salidas de los relés (7).

3.2.1.2 Esquemático

En la imagen de más abajo se puede apreciar el circuito esquemático de un canal, el resto de los canales repite la misma configuración.

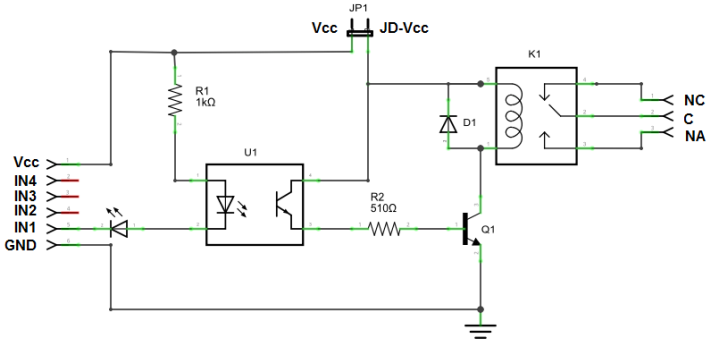


Ilustración 7 Circuito esquemático

3.2.1.3 Funcionamiento [15]

A partir del circuito de la imagen anterior se analizará el funcionamiento del circuito.

La entrada IN1 está conectada al cátodo del diodo del optoacoplador a través del led indicador. El ánodo del diodo del optoacoplador se conecta a Vcc (positivo) por medio de R1. Estos tres componentes, el diodo indicador, el diodo del opto y la R1 forman un circuito serie por el cual circula la corriente cuando la entrada está a un nivel BAJO (conectada a GND) y no circula si la entrada está a un nivel ALTO (conectada a Vcc).

El transistor del opto tiene su colector a JD-Vcc y su emisor conectado a Q1 a través de una resistencia de 510 ohm. Este es otro circuito serie por el cual circula corriente cuando el transistor del opto conduce al ser "iluminado" por su diodo, con lo que se introduce corriente en la base de Q1 a través de R2.

Finalmente, Q1 está conectado en una típica configuración emisor común, con su emisor a masa (GND) y la bobina del relé como carga en el colector. Cuando circula corriente por la base desde el opto, Q1 se satura permitiendo el paso de la corriente a través de la bobina del relé, lo que produce que se cierren los contactos del mismo (común con normal abierto). El diodo D1 protege al transistor de la tensión que aparece en la bobina del relé cuando deja de circular corriente por la misma.

En resumen, al ponerse la entrada a nivel bajo se pone a saturación el transistor Q1 a través del optoacoplador con lo que se cierra el contacto normal abierto del relé.

3.2.1.4 Alimentación y consumo

La forma más sencilla de alimentar este módulo es desde Vcc y GND de la placa Funduino, manteniendo el Jumper en su lugar, con lo que JD-Vcc = Vcc. Esta conexión tiene dos limitaciones importantes:

Se pierde la aislación eléctrica que brindan los optoacopladores, lo que aumenta la posibilidad de daño al Funduino si hay algún problema con las cargas de los relés.

La corriente consumida por las bobinas de los relés debe ser provista por la placa Funduino. Cada bobina consume unos 90 mA y las cuatro juntas suman 360 mA. Si a esto le sumamos el consumo del microcontrolador, unos 100 mA, y el de otros dispositivos y salidas, estamos superando los 500 mA que puede suministrar un puerto USB. En nuestro caso se decidió alimentar la placa con una fuente externa 9V – 1000 mA, lo que aumenta el límite de corriente a 1 A y nos evita problemas durante las pruebas.

3.2.2 Librería TimeAlarms [16]

Existe una librería más, relacionada con la librería Time, llamada TimeAlarms que nos permite definir alarmas en nuestro programa, muy al estilo de la que podemos fijar en un reloj despertador. Cuando una alarma dispara, llama a una función específica que se ejecuta en el momento preestablecido.

Es una librería interesante si se tiene que repetir tareas periódicamente y además nos ofrece uno timers cómodos que se ejecutan de forma programada, sin necesidad de definir interrupciones que podrían interferir con otras funciones.

- La librería TimeAlarm no emplea interrupciones, sino que se apoya en la gestión interna de Time.

- TimeAlarm no necesita ningún hardware o RTC especifico, solo requiere que la librería Time esté cargada.

- Se pueden definir múltiples alarmas, cambiando el header de TiemeAlarm.h

- No se pueden definir intervalos menores de 1 segundo. Para periodos inferiores se requiere usar una interrupción programada o Timer

Utilizando la función Alarm.alarmRepeat(hours, minutes, seconds, function) se hará una llamada a la función especificada en sus argumentos de entrada para que se encargue del encendido o apagado de los dispositivos poniendo en estado bajo o alto las entradas de los relés indicados.

Para las funciones que se deban ejecutar en un cierto intervalo de tiempo, como es el caso de la medición de temperatura, se utilizará Alarm.timerRepeat(seconds,function).

Por último, se tiene que mencionar que las alarmas y los temporizadores son sólo chequeados y sus funciones llamadas cuando se utiliza la función de retardo Alarm.delay(milisegundos). Se puede poner a 0 para un retardo mínimo. Este retardo se debe utilizar en lugar del retardo normal para el procesamiento oportuno de alarmas y temporizadores.

3.2.3 Conexionado

Después de configurar las salidas digitales de Funduino, donde el relé 1 será controlado por la salida 12 y se conectará al filtro, el relé 2 tendrá asociada la salida digital 11 y controlará la luz y por último el relé 3 con la salida 10 y el calentador, quedaba verificar que todo funcionaba correctamente, pues los relés conmutaban pero no se veía si eran

capaces de apagar y encender los dispositivos. Para ello se conectó la luz del acuario a uno de los relés, cortando su cable principal y sacando los dos cables que se encuentran en su interior:

- Cable marrón: Cable positivo o de fase
- Cable azul: Cable negativo o neutro.

El cable marrón es el que se conectó al relé, se cortó en dos, se le añadieron dos punteras de 0,75 mm^2 fijadas con una crimpadora y se conectó a los terminales C y NA, tal y como se explicó en los apartados anteriores.

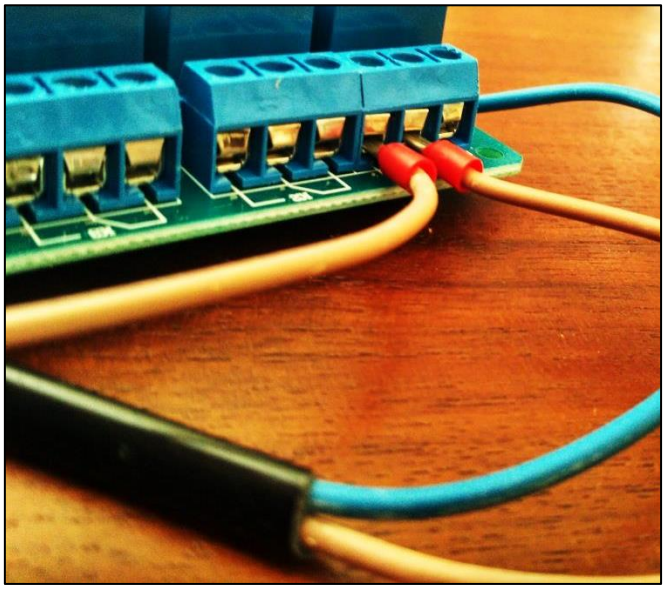


Ilustración 8 Conexión luz-relé

Tras realizar la prueba de una conmutación periódica se pudo verificar la capacidad del módulo de relés para encender y apagar los dispositivos conectados.

3.3 Sensado de temperatura

Con los conocimientos adquiridos anteriormente acerca de la librería Timelarms, gestionaremos una alarma que se ejecute cada 20 segundos y que llame a una función que se encargue de hacer una lectura de la temperatura a la que se encuentra el agua, en caso de estar por debajo de la temperatura ideal llamaremos a otra función que ponga en un nivel bajo la entrada correspondiente al relé que controla el calentador si fuese necesario, pues de darse el caso de encontrarse encendido no realizaríamos ninguna acción.

Los temporizadores activan las tareas que se producen después de transcurrirse un intervalo de tiempo especificado. El intervalo del temporizador se puede especificar en segundos, o en horas, minutos y segundos.

Alarm.timerRepeat (15, Repeticiones); // tarea del temporizador cada 15 segundos

3.3.1 Sensor de temperatura DS18B20 y Circuito de acondicionamiento

El kit contiene un módulo adaptador DS18B20 y un sensor de temperatura digital a prueba de agua DS18B20, por lo que puede medir la temperatura directamente en el agua. Utiliza un interfaz de bus único que permite múltiples sensores DS18B20 montados en el mismo bus, con el fin de lograr una gran área de medición multipunto de la temperatura en caso de ser necesario. Con la resistencia pull-up, el módulo adaptador se puede conectar directamente a la mayoría de los microcontroladores y es ampliamente utilizado en la monitorización de la temperatura de los tanques de peces, equipos, maquinaria e invernaderos.

3.3.1.1 Características

- Comunicación de un solo bus, múltiples sensores pueden compartir una única línea de datos
- Biblioteca Arduino: OneWire
- Voltaje de trabajo: 3.2 ~ 5.25VDC, se alimentará con 5V.
- Corriente de trabajo: 2mA (máx.)
- Resolución: programable de 9-12 bits
- Rango de medición: -55 ~ 110 ℃
- Precisión de la medición: ± 0, 5 'C @ 10 ~ 80' C;
- Conductores de salida: amarillo (DATA), rojo (VCC), negro (GND)

El DS18B20 es un sensor de temperaturas fabricado por la compañía Maxim Integrated. Proporciona la salida mediante un bus de comunicación digital que puede ser leído con las entradas digitales de Funduino.

El sensor DS18B20 es un sensor barato y, sin embargo, bastante avanzado. Dispone de un rango amplio de medición de -55ºC a +125ºC y una precisión superior a ±0.5°C en el rango –10°C de +85°C.

Una de las ventajas del DS18B20 es que se comercializa tanto en un integrado TO-92 como en forma de sonda impermeable, lo que permite realizar mediciones de temperatura en líquidos y gases.

Ilustración 9 Sensor de temperatura DS18B20

El DS18B20 emplea un bus de comunicación denominado 1-Wire, propiedad de la empresa Maxim Integrated, aunque podemos usarlo sin tener que pagar por ninguna tasa (es parte del precio del dispositivo).

La principal ventaja del bus 1-Wire es que necesita un único conductor para realizar la comunicación (sin contar el conductor de tierra). Los dispositivos pueden ser alimentados directamente por la línea de datos, o mediante una línea adicional con una tensión de 3.0 a 5.5V.

Dentro del mismo bus 1-Wire podemos instalar tantos sensores como deseemos. Además, el bus 1-Wire permite emplear cables más largos que otros sistemas antes de que se deteriore la comunicación.

El DS18B20 es un gran sensor para la medición de temperatura, tanto en ambientes domésticos como industriales. Sus características permiten crear redes con gran número de sensores a controlar.

3.3.1.2 Funcionamiento [17]

Una de las principales ventajas de DS18B20 es su bus de comunicación 1-Wire que le permite realizar la transmisión empleando únicamente un cable de datos. Para ello, 1-Wire está basado en un complejo sistema de timings en la señal entre el dispositivo emisor y el receptor, sistema que se explicó en el capítulo dos.

La mayor desventaja del sistema 1-Wire es que requiere un código complejo, lo que a su vez supone una alta carga del procesador para consultar el estado de los sensores. El tiempo de adquisición total de una medición de 750ms.

El dispositivo 1-Wire permite que todos los dispositivos conectados al bus se alimenten a través de la línea de datos. Para ello, disponen de un condensador que almacena energía mientras la línea de datos está en HIGH. Este modo se denomina "modo parásito". En caso de no usar el modo parásito, los dispositivos deberán ser alimentados a una tensión entre 3.0V y 5.5V.

En cualquier caso, el bus 1-Wire requiere una resistencia de pull-up de 4k7 entre Vcc y Vq para que funcione correctamente.

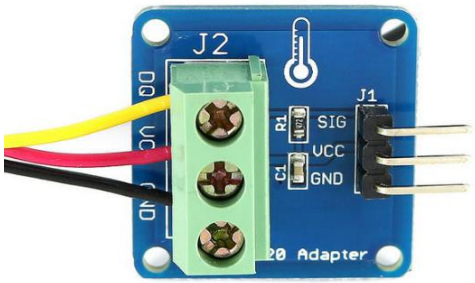


Ilustración 10 Módulo adaptador DS18B20

Para poder utilizar múltiples dispositivos en un bus 1-Wire cada sensor dispone de una memoria ROM que es grabada de fábrica con un número de 64 bits. Los 8 primeros bits corresponden a la familia (0x28 para el DS18B20). Los siguientes 48 bits son el número de serie único. Los últimos 8 bits son un código CRC.

La resolución del DS18B20 es configurable a 9, 10, 11 o 12 bits, siendo 12 bits el modo por defecto. Esto equivale a una resolución en temperatura de 0.5°C, 0.25°C, 0.125°C, o 0.0625°C, respectivamente.

Que el DS18B20 disponga de una resolución por defecto de 0.0625ºC, no significa que esa sea su precisión. Sin embargo, el DS18B20 es considerablemente preciso en todo el rango –10°C de +85°C. Se pueden consultar las desviaciones medias en la siguiente gráfica.

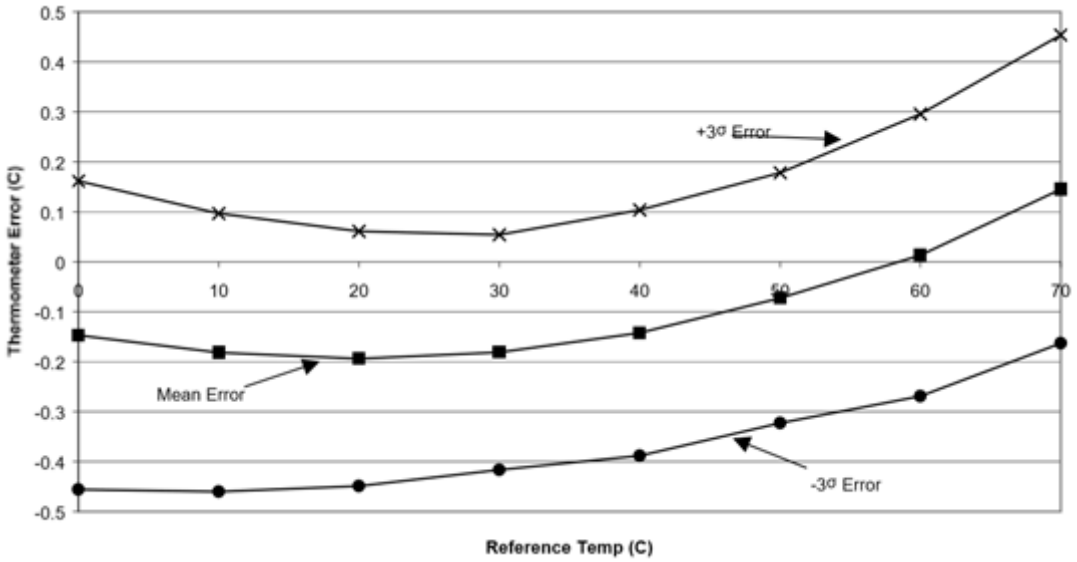


Ilustración 11 Desviaciones medición DS18B20

3.3.1.3 Montaje

Los dispositivos 1-Wire disponen de tres terminales

- **DQ**, la línea de datos, conectada al pin digital 2 de Funduino.
- **VDD,** línea de alimentación, proporcionada por los pines 5V.
- **GND**, línea de tierra

En la siguiente imagen figura la posición de estos pines en ambos formatos del sensor, como integrado y como sonda impermeable.

Ilustración 12 Terminales DS18B20

Hemos comentado que el bus 1-Wire necesita una resistencia de pull-up de 4K7, y que podemos alimentar el sensor directamente a través del pin Vdd o usar el modo "parásito" y alimentarlo con la propia línea de datos.

Por tanto, el esquema más simple de conexión se muestra en la siguiente imagen, donde podemos conectar el sensor a cualquier entrada digital de Funduino.

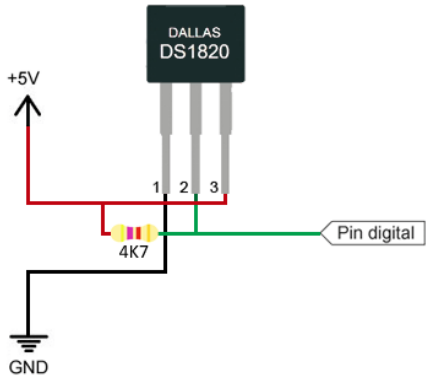


Ilustración 13 Conexión simple DS1820

Finalmente, si quisiéramos emplear el modo parásito, que no es nuestro caso, simplemente se tendría que conectar Vdd de todos los dispositivos a GND. De esta forma, los sensores tomarán su alimentación de la línea de datos DQ.

3.4 Interfaz Gráfica

Tras probar a sincronizar el microcontrolador y el RTC, activar o desactivar los relés con alarmas configuradas a las horas de encendido y apagado de los dispositivos, y realizar mediciones de temperatura en un intervalo de tiempo establecido tocaba crear una interfaz gráfica que nos permitiese controlar todo lo implementado hasta este punto. La idea se basa en poder visualizar el estado de los relés así como las mediciones de temperatura en una pantalla principal y a su vez tener una segunda pantalla de ajustes donde poder configurar los tiempos de encendido y apagado a través de la edición de alarmas, los modos de funcionamiento de los dispositivos (formato 12h o 24h, habilitado o deshabilitado) y la temperatura a la que se desea mantener el agua.

3.4.1 Pantalla 2.8" TFT LCD Táctil

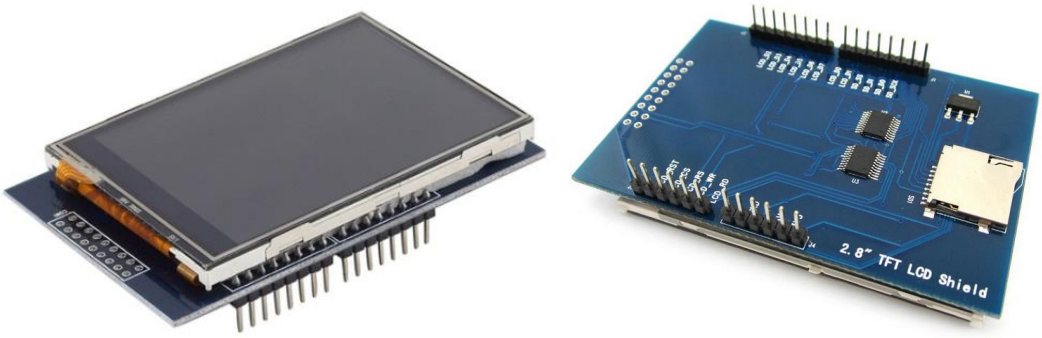


Ilustración 14 Pantalla 2,8" LCD Táctil

3.4.1.1 Características

Esta shield para Funduino consiste en una pantalla TFT táctil de 2.8" con zócalo para tarjetas Micro SD, con una resolución de 240x320 pixeles.

- Pantalla TFT táctil resistiva de 2.8"
- Resolución 240x320
- Controlador HX8347G
- Interfaz SPI para el control de tarjeta micro SD
- Librerías Adafruit_GFX Adafruit_TFTLCD y TouchScreen

    - *Librería Adafruit_TFTLCD:* Esta es la librería Adafruit_TFTLCD original, que incluye drivers de varias pantallas con diferentes chips (ILI9325, ILI9341, HX8347G, HX8357), por lo que se debe seleccionar el chip correcto en el código del sketch para poder visualizar texto, gráficos e imágenes en nuestra pantalla LCD.
    - *Librería Adafruit_GFX*: Esta librería de Adafruit es la que nos proporcionará el código necesario para la realización de gráficos en la pantalla (puntos, círculos, líneas, etc)
    - *Librería TouchScreen*: Librería que se encarga de traducir e interpretar las coordenadas, para poder trabajar con el panel táctil resistivo de 4 hilos que incluye nuestro shield.

- El display y el control táctil usan los pines analógicos A0-A4 y los digitales 22-29 por lo que quedan muchos libres para el resto de dispositivos.

3.4.1.2 Funcionamiento

Lo primero que se llevó a cabo fue la configuración de los pines para el control de la pantalla táctil y la TFT LCD mediante dos sencillas funciones y la identificación del controlador de pantalla.

```
TouchScreen ts = TouchScreen(XP, YP, XM, YM, 300);
Adafruit_TFTLCD tft(LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);
tft.begin(identifier);
```

Una vez realizada la configuración se realizó una prueba donde mostrásemos por pantalla las coordenadas de la pulsación y tras verificar el funcionamiento se pasó al desarrollo de la interfaz.

Aquí se muestra unas imágenes de cómo fue la primera interfaz, mejorada posteriormente, y cuál fue la idea base para su desarrollo.

Ilustración 15 Primera interfaz desarrollada

La pantalla principal consta de 3 Apartados:

- El primero donde vemos la hora a la que nos encontramos y la última lectura de temperatura.

- El estado de los dispositivos conectados, en verde aquellos que están encendidos y en rojo los apagados.

- Un botón de ajustes que nos llevará a la segunda pantalla donde se realizarán los cambios que se deseen en el control de nuestro sistema.

En la pantalla de ajustes veremos el control de encendido y apagado del filtro, pudiendo configurar la hora exacta, el modo de luz, si seguimos horario de invierno o de verano, y por último la temperatura idónea para el acuario. Si pulsamos OK se configurará el sistema con los nuevos ajustes y si presionamos X volveremos a la pantalla principal sin realizar ninguna modificación.

3.5 Sistema Akuatización

Una vez introducidos y probados los diferentes apartados del proyecto de forma individual se juntaron todos en un mismo sketch para testearlo y comprobar su funcionamiento, de esta forma surgirían nuevas ideas y se podrían introducir mejoras posteriormente para dar lugar a un sistema sencillo, robusto y que cumpla todos los objetivos propuestos, proporcionando un control automático de los dispositivos conectados al acuario.

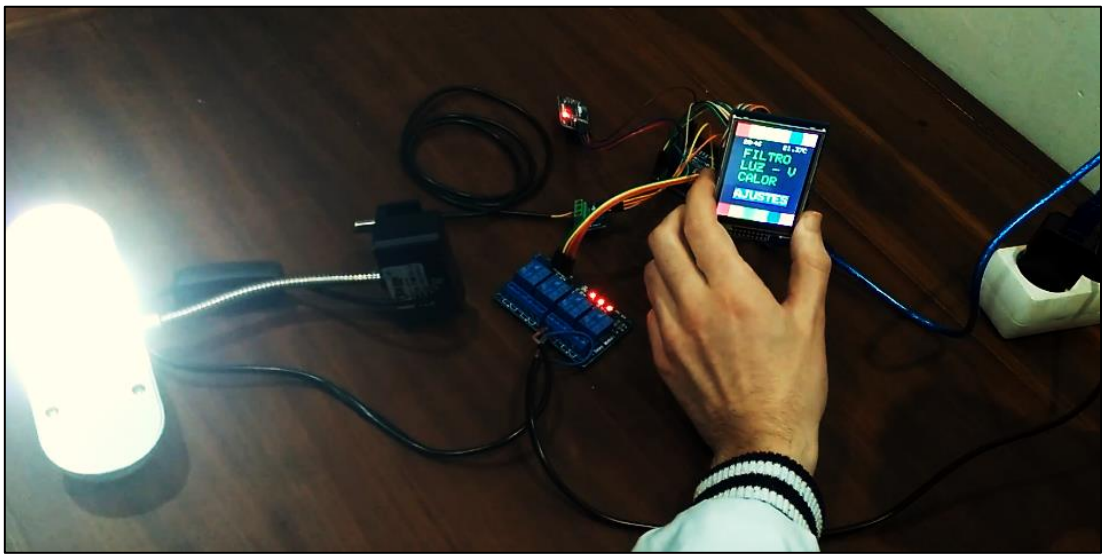


Ilustración 16 Pruebas Sistema Akuatización

El funcionamiento del primer programa era sencillo:

Se configuró la pantalla táctil de forma que se pudiese detectar donde se estaba pulsando exactamente y así actuar en función de las coordenadas leídas, pudiendo pasar de la pantalla principal (el monitor donde se ve el estado de los dispositivos) a la pantalla de ajustes, para así poder cambiar la configuración a nuestra conveniencia.

Se configuraron unas alarmas para que encendiesen o apagasen los relés en función del valor de los parámetros de control, y otra alarma para hacer una lectura periódica de la hora y temperatura y conectar el calentador si fuese necesario.

Los ajustes permitían configurar el encendido y el apagado del filtro, el horario utilizado para el encendido de la luz, verano para un horario de 22:00 a 10:00 o invierno para iluminar el acuario de 21:00 a 9:00, y por último la temperatura ideal.

En el momento de aceptar estas modificaciones, se sustituye el valor de las antiguas variables de control fijadas en el setup del programa, se configuran las nuevas alarmas y se chequea si los dispositivos deben estar encendidos con los nuevos horarios fijados o por el contrario deben estar en off.

Después de testear el sistema surgieron muchas ideas para mejorarlo o bien para arreglar algunas partes que no acababan de convencer. Estas mejoras, que se llevaron a cabo tras la primera prueba, se detallaran más adelante, a continuación se describen brevemente para una mejor comprensión del capítulo:

1. Lo primero que se realizó fue el diseño de una pantalla de arranque que mostrase el nombre del sistema "Akuatización" mientras se ejecuta el setup del código, de esta forma no vemos una pantalla en blanco hasta que se ejecuta el último paso del setup, donde se muestra el monitor.

2. La segunda mejora fue implementar la habilitación y deshabilitación de los diferentes dispositivos que se encargan del mantenimiento del acuario desde la pantalla principal, pudiendo apagarlos en cualquier instante sin tener que esperar a que finalice su tiempo de encendido.

3. Detección automática del horario de invierno y de verano con su correspondiente ajuste de hora para que el usuario no tenga que preocuparse de ello.

   Se configura una alarma diaria a las 2:30 de la madrugada que chequee si se debe hacer un cambio de horario, estas fechas están introducidas en el sistema hasta 2020, de forma que se realizarán estos cambios durante los próximos 3 años sin ningún problema. Llegada la última fecha tocará introducir los nuevos cambios horarios.

   - 31/3/2017, 29/10/2017
   - 25/3/2018, 28/10/2018
   - 31/3/2019, 27/10/2019
   - 29/3/2020, 25/10/2020

   Cuando se produzca un cambio de horario se redibujara la pantalla principal, donde se podrá ver junto al apartado luz el horario en el que nos encontramos.
   Se eligió mostrarlo en ingles por tema de espacio en el monitor, por tanto veremos "Summer" en verano y "Winter" en invierno.

En horario de verano la luz tendrá un horario de encendido de 22:00 – 10:00 y en invierno de 21:00 – 09:00

4. Introducción del control de oxigenación, el cual se podrá poner en modo 12H o 24H dependiendo de las necesidades del acuario, ya que para aquellos que tengan plantas solo es necesario tener el oxigenador activo durante la noche. El conexionado de los relés quedó configurado de la siguiente forma:

   - Relé 1: Salida Digital 12 - Filtro
   - Relé 2: Salida Digital 11 - Luz
   - Relé 3: Salida Digital 10 - Calentador
   - Relé 4: Salida Digital 09 - Oxigenación
   -

   En caso de encontrarse en modo 12H seguirá el mismo horario que la luz, de 22:00 – 10:00 en verano y de 21:00 – 09:00 en invierno. Si el modo elegido es 24H entonces permanecerá siempre encendido

5. Almacenamiento de las variables de control en la memoria EEPROM para que en caso de apagado se conserven las modificaciones realizadas sobre los valores de control iniciales que traerá por defecto el sistema.

3.5.1 Inicio del sistema

Ilustración 17 Pantalla de arranque

Nada más arrancar el sistema pasa por diferentes fases

1. Pantalla inicial Akuatización.
2. Configuración de los pines de Funduino.
3. Sincronización del microcontrolador con el RTC.
4. Se establece una alarma que llame a la función encargada de medir la temperatura cada 20 segundos y consultar la hora.
5. Se establece una alarma diaria a las 2:30 de la madrugada que compruebe el horario en el que nos debemos encontrar, verano o invierno.
6. Llamada a la función Setcontrol. En ella leeremos de la memoria EEPROM los parámetros de control establecidos.
7. Ejecución de la función SetAlarms. Aquí se configuraran las alarmas necesarias para el encendido y apagado de los dispositivos con los parámetros de configuración leídos anteriormente.
8. Llamada a la función SetReles. Se chequeará si los relés deben estar encendidos o no, para ello comprobaremos con los parámetros de control obtenidos en el paso 6 si un dispositivo está dentro de la franja horaria de encendido, o en el caso del calentador, si el acuario se encuentra a una temperatura por debajo de la ideal. Por ejemplo, en el caso del filtro chequearemos cuál es su hora de encendido y de apagado, si está configurado para encenderse a las 8:00 y apagarse a las 23:00, se comprobaría que estamos dentro de la franja horaria 8:00 – 23:00 y en caso afirmativo pondríamos la entrada correspondiente del relé en estado bajo, dejando paso a la circulación de corriente.
9. Se dibuja la pantalla principal, mostrando la primera lectura de temperatura, la hora en la que nos encontramos y el estado de todos los dispositivos una vez chequeado su estado y sus parámetros de control.

3.5.2 Pantalla principal

Ilustración 18 Pantalla Principal - Monitor

Esta pantalla se puede dividir en tres partes:

- La cabecera: Es donde vemos la hora y la temperatura medida en todo momento, esta se actualiza cada 20 segundos y se redibuja sin machacar el resto de la pantalla para tener un sistema más eficiente.

- La monitorización: Se trata de la parte central de la pantalla, en ella vemos todos los dispositivos conectados a los relés, El filtro conectado al relé 1, la luz al 2, el calentador al tercero y el oxigenador al cuarto.

    - Se puede ver cómo está configurado el control del sistema sin tener que acceder a los ajustes:

        - Franja horaria de encendido para el filtro
        - Si estamos en horario de verano o de invierno, lo que determinara la franja de encendido para la luz y el oxígeno (en caso de encontrarse en modo 12H), siendo de 22:00 a 10:00 en horario de verano y de 21:00 a 9:00 en caso de invierno
        - La temperatura ideal a la que queremos mantener el acuario.
        - El modo de oxigenación, 12H en caso de tener plantas y 24H en caso contrario.

    - Se puede habilitar o deshabilitar cualquier dispositivo con tan solo pulsar sobre su nombre. En caso de estar habilitado mostrará si se encuentra encendido o apagado mediante el color del texto.

        - Verde en caso de encendido y habilitado
        - Rojo si está apagado y habilitado
        - Blanco si se encuentra deshabilitado

- La tercera parte de la pantalla principal es el acceso a la pantalla de ajustes.

3.5.2.1 Deshabilitación

La explicación del proceso que se lleva a cabo en la deshabilitación de un dispositivo es muy sencilla. Cuando se activa esta funcionalidad para uno de los dispositivos conectados se pasa a ignorar todas las alarmas asociadas a este y se pone en estado alto la entrada de su relé correspondiente para que lo apague de inmediato hasta una posterior habilitación.

3.5.2.2 Habilitación

Se vuelve a tener en cuenta las alarmas asociadas al dispositivo habilitado y se realiza un chequeo de su franja horaria, o de la temperatura en caso del calentador, para ver si se debe activar o no. Después de este chequeo se redibuja la parte de monitorización de la pantalla principal para asignar los colores correspondientes.

3.5.3 Pantalla de Ajustes

Ilustración 19 Pantalla de Ajustes

Aquí podremos modificar todos los parámetros de control

- Hora de encendido del filtro, ajustando hora y minutos.
- Hora de apagado del filtro, ajustando hora y minutos.
- Modo de oxigenación, activo 12H o 24H.
- Temperatura ideal, entre 20ºC y 32ºC.

En la parte inferior de la pantalla podremos cancelar esta modificación pulsando X, lo que nos llevará a la pantalla principal directamente, o bien aceptar la modificación pulsando OK, que también nos llevará de vuelta a la pantalla principal, pero en este caso tras la ejecución de diferentes funciones encargadas de hacer definitivo el cambio en los parámetros de control.

Cabe destacar que para el redibujado de la pantalla de ajustes utilizamos unas variables auxiliares que adquieren el valor de las variables de control utilizadas en el paso de la pantalla principal a la de ajustes, y en caso de aceptar las modificaciones utilizaremos el valor que contengan las variables auxiliares para sustituir el valor almacenado en la EEPROM.

3.5.3.1 Modificación parámetros de control

Las llamadas a las distintas funciones que se realizan entre la aceptación de la modificación de los parámetros y el dibujo de la pantalla principal se pueden resumir en tres etapas.

Escritura de los nuevos parámetros en la memoria EEPROM, sobrescribiendo los que se encuentran en ese momento.

Liberación en memoria de las alarmas antiguas y configuración de las nuevas con los parámetros establecidos por el usuario.

Control del estado de los relés, se comprueba si con los parámetros introducidos se debe conmutar alguna entrada de los relés. Con los valores antiguos un dispositivo se podía encontrar fuera de la franja de encendido y con los nuevos puede pasar a estar dentro, o viceversa. En el caso del calentador se podría pasar a tener una temperatura por debajo de la nueva temperatura ideal introducida, lo que nos obligaría a conmutar el estado del relé que controla el calentador para encenderlo, o bien encontrarnos con el caso contrario, una temperatura superior a la establecida, y obligarnos a apagarlo.

3.5.4 Funcionamiento – Diagrama de flujo

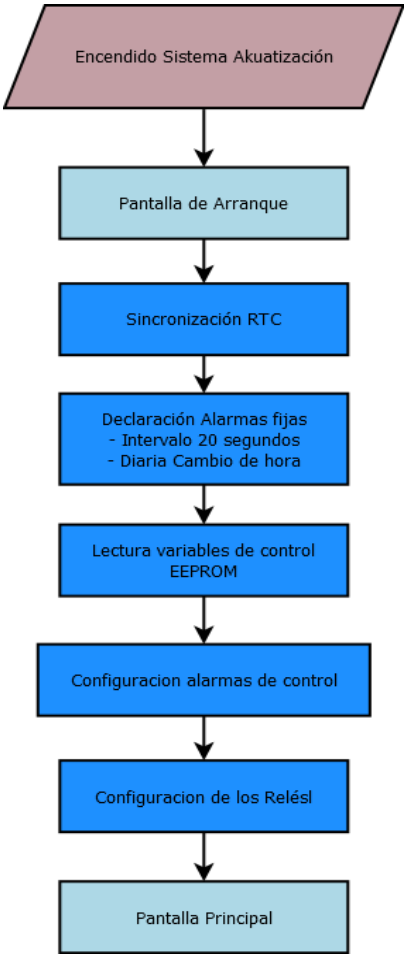


Ilustración 20 Setup del Sistema

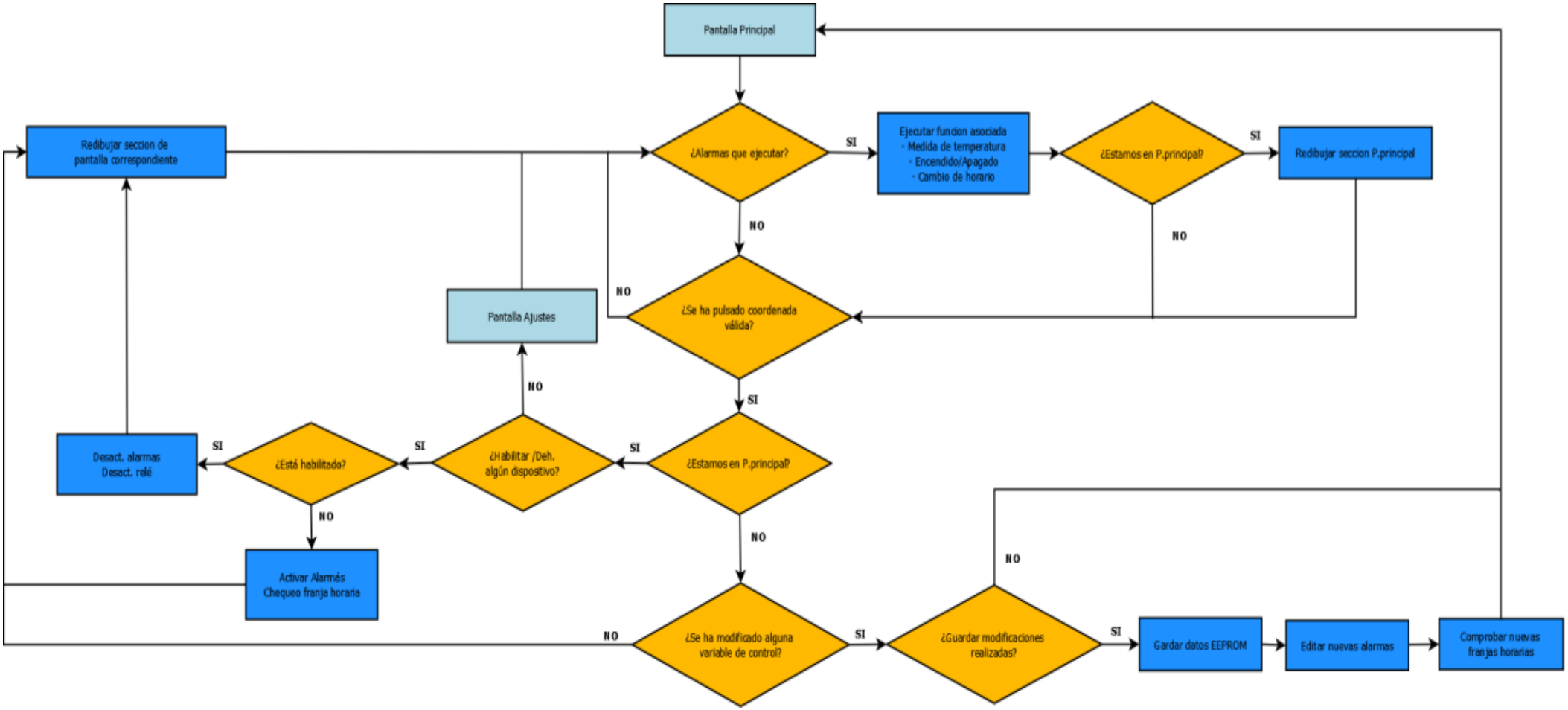


Ilustración 21 Loop del Sistema

3.5.5 Mecanizado

Llegados a este punto del proyecto el objetivo es mantener todos los dispositivos bien organizados, sujetos, sin problemas para conectarlos entre sí y que nos permitan sacar los cables externos necesarios para el funcionamiento del sistema a través de pasadores que se instalaran en la caja. El cableado externo se trata de la alimentación para Funduino, el cable del sensor de temperatura que se debe introducir en el acuario y todos los cables a conectar con los relés.

Para proteger el conjunto de dispositivos que dan lugar al sistema encargado de automatizar el acuario se reutilizó una caja cuadrada de plástico (ABS) con medidas 13 x 13 x 7.5 cm empleada en otro proyecto anterior. Estas medidas son perfectas para el proyecto ya que ninguna medida de los dispositivos utilizados excede las de la caja.

- Funduino Mega 2560: 7.0 x 5.4 x 1.4 cm
- 2'8 TFT-LCD: 7.0 x 5.4 x 1.4 cm
- Módulo Relés: 7.6 x 5.6 x 1.9 cm
- Circuito acondicionador Temp: 2.8 x 2.4 x 1.5 cm
- RTC: 3.7 x 2.1 x 0.6 cm

El mecanizado a llevar a cabo consiste en:

- Perforación de la caja para realizar tres orificios traseros por los que pasar los cables externos de alimentación, sensado y control.

- Corte de un rectángulo de 7 cm x 5 cm centrado en la tapa superior que permita sacar y fijar (con termocola) la pantalla táctil para la interactuación del usuario con el sistema.

- Dividir la caja en dos niveles utilizando una tabla cuadrada de madera de 9 cm x 9 cm, la cual será previamente perforada para la sujeción de los dispositivos, y unos tornillos de 3 cm de altura, dejando en la parte inferior el RTC y el circuito acondicionador y así tener en el nivel superior los relés y Funduino para facilitar el posterior conexionado.

El resultado del mecanizado se puede ver en las siguientes imágenes, todo salió tal y como estaba previsto, obteniendo una caja de protección en la que encajaban a la perfección todos los dispositivos que componen nuestro sistema.

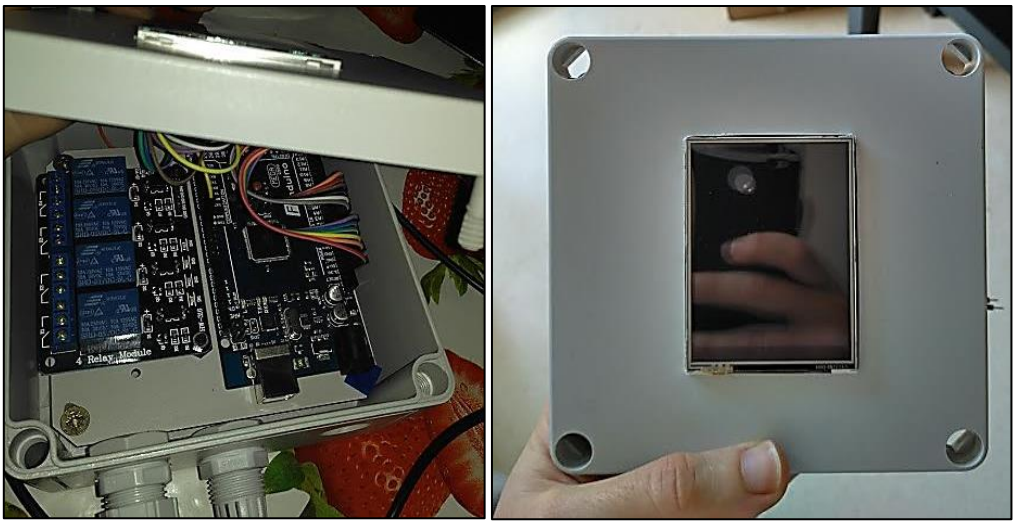


Ilustración 22 Resultado del mecanizado

3.5.6 Logotipo

Como era de esperar todo producto diseñado debe tener una imagen y un nombre. Aquí podemos ver el logotipo creado para el sistema Akuatización.

Ilustración 23 Logotipo del sistema

3.6 Implantación y Resultados

La primera prueba para comprobar que todo funcionaba correctamente se realizó en un pequeño acuario que solo contaba con filtro, luz, y calentador, por lo que uno de los relés quedaba libre y por tanto se procedió a su deshabilitación.

Se realizaron todas las conexiones, cortando los cables de los dispositivos tal y como se explicó en el apartado conexionado de los relés y se instaló en el acuario tantos los dispositivos como el sensor de temperatura una vez fijadas todas las entradas y salidas del sistema Akuatización.

- Alimentación de 1A-5V para Funduino por el primer orificio
- Salida del sensor de temperatura por el segundo
- Dispositivos conectados a los relés por el tercero
    - Relé 1: Filtro
    - Relé 2: Luz
    - Relé 3: Temperatura
    - Relé 4: Sin conexión – Deshabilitado

Ilustración 24 Prueba 1 Akuatización

Las pruebas en este caso fueron todo un éxito, no hubo ningún tipo de problema durante los 4 días que estuvo funcionando el sistema.

- Encendido y apagado conforme se configuraron los horarios inicialmente.
- Cambio en la configuración conforme lo previsto, se liberaron alarmas antiguas evitando sus correspondientes llamadas a funciones y se impusieron las nuevas, actuando en el nuevo horario establecido.
- Correcto almacenamiento en EEPROM de las variables de control, se apagó el sistema momentáneamente para comprobar que la nueva configuración había quedado guardada en la memoria no volátil.
- Cambio de horario realizado con éxito el 26 de Marzo, adelantando una hora a las 2:30 de la madrugada. Se configuro la hora actual más una hora al RTC y se sincronizo con Funduino como se esperaba.
- Perfecta deshabilitación y habilitación de los relés. El cuarto relé se mantuvo deshabilitado durante los 4 días, se ignoraron las alarmas previstas para la oxigenación tanto en horario de verano como de invierno. Se probó a deshabilitar el filtro durante su franja horaria de encendido, esto conllevó un apagado instantáneo, las alarmas se ignoraron durante su deshabilitación y se probó a su habilitación fuera de la franja de encendido, el chequeo de la franja horaria en la que se encontraba se realizó correctamente y por tanto se mantuvo apagado, mostrándose en inactivo (rojo) en el monitor hasta su posterior encendido a la hora indicada en el control.

Tras múltiples cambios en la configuración el sistema respondió a la perfección por lo que podemos concluir que en esta primera prueba ha demostrado ser fiable y robusto, se buscó el fallo de manera constante para futuras mejoras pero no se encontró nada relevante.

3.6.1 Implantación acuario 60L

Por último se probó a implantarlo en el lugar para el cual ha sido diseñado, una pecera de 60L con filtro, calentador, luz y oxigenación.

Ilustración 25 Akuatización Final

En este caso todo funcionó tal y como se esperaba, lo más complejo fue la instalación del sistema, conectando cada uno de los elementos que mantienen el acuario con los cuatro relés que se encargan de su encendido y apagado. Una vez realizada la instalación, el control del sistema resultó de lo más sencillo para el usuario, uno de los objetivos de este proyecto.

Las pruebas realizadas fueron las mismas que en el primer caso, añadiendo un par más para asegurarnos que la oxigenación también trabajaba como estaba previsto. En el modo 24H permanecía siempre encendido y configurado con modo 12H seguía el mismo horario que la luz, de 22:00 a 10:00 al encontrarnos en horario de verano.

4. Conclusiones

En primer lugar hay que comentar que el sistema ha cumplido por completo las expectativas creadas. Una pequeña idea que se tuvo para solventar un problema ha dado lugar a un proyecto en él se ha podido dar rienda suelta a la imaginación ya que no había unas directrices que seguir, buscando información en internet, leyendo libros y recorriendo de arriba abajo distintas tiendas de electrónica se ha obtenido un sistema sobre el que se podrá trabajar en un futuro.

Se ha podido comprobar que hemos alcanzado los objetivos propuestos en este proyecto, logrando crear un sistema mediante un microcontrolador que permite controlar el estado de un acuario en todo momento, haciendo invisible toda la programación o dispositivos utilizados y que con un mínimo de conocimientos eléctricos se pueda implementar en un acuario genérico.

El proceso manual ha pasado a ser completamente automático y se ha mejorado su control, la seguridad y el confort de la persona.

Por ultimo cabe destacar que el coste de desarrollo de este sistema ha sido menor de lo esperado, por debajo de los 40 euros, obteniéndose un producto que puede utilizarse en la mayoría de los acuarios y que no supone una gran inversión para el usuario, además el precio final puede reducirse comprando los componentes en grandes cantidades.

A pesar de haber desarrollado un buen sistema siempre se pueden implantar mejoras que faciliten su uso o nuevas funcionalidades, tal y como aprendimos en la asignatura de diseño electrónico orientado a producto necesitamos tener un par de mejoras a nuestra disposición para futuras versiones. A continuación comentaremos un par de ideas que aportaran un extra de calidad y funcionalidad en el sistema para su posterior desarrollo.

4.1 Trabajo futuro

4.1.1 Facilitar conexionado

Una de los inconvenientes de la caja que protege el sistema es la necesidad de abrirla para conectar o desconectar los dispositivos del acuario a los relés, esto dificulta la tarea de realizar las conexiones, ya que primero tenemos que introducir todos los cables a través del pasador y posteriormente atornillarlos a las entradas de los relés. Esta manipulación del sistema puede provocar que se suelte alguna otra conexión y que por tanto requiera de una persona que entienda de electrónica para que solucione el problema.

Una forma de evitar esto es realizar un mecanizado diferente en la caja ABS, realizando una apertura en la tapa de la caja, justo por encima de la pantalla y otra apertura rectangular en la parte trasera.

La apertura superior tiene como objetivo poder introducir un destornillador para apretar los tornillos que fijan las conexiones de los cables a los relés, mientras que la apertura trasera tiene la finalidad de poder realizar las conexiones dispositivos-relés sin que sea necesario abrir la caja, con lo que nos ahorraremos futuros problemas.

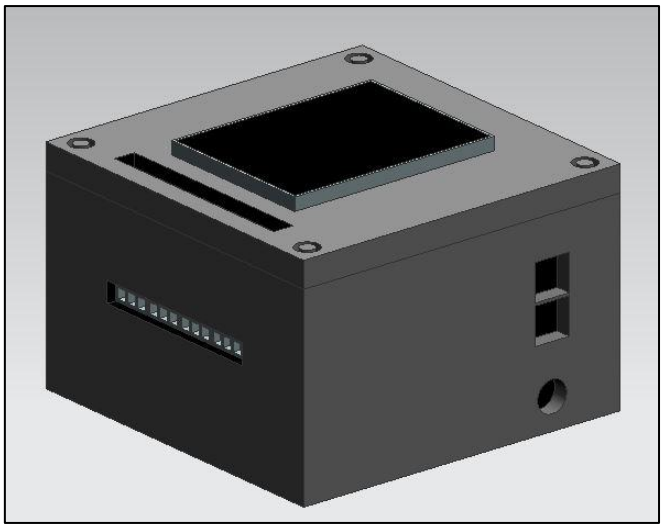


Ilustración 26 Modelo nuevo mecanizado

Por ultimo comentar que en esta apertura trasera se especificaría donde tiene que ir conectado cada dispositivo y así nos ahorraríamos tener que especificar a través de la pantalla táctil a que relé se encuentra conectado cada dispositivo.

Ilustración 27 Pantalla ajustes conexionado relés

4.1.2 Control vía Ethernet

En la actualidad parece que necesitamos un control remoto para cada uno de los dispositivos electrónicos que se encuentran en el hogar, la idea es aumentar la calidad de vida del usuario facilitándole el control de estos elementos como bien puede ser un televisor, un aire acondicionado o un equipo de música.

Cada día crece más esta dependencia y con ella nuevas ideas como el control vía internet pues la comunicación bluetooth o infrarroja limita el alcance del control a escasos metros, lo que muchas veces hace un sinsentido utilizar este tipo de comunicación, como es nuestro caso.

No merece la pena invertir en una comunicación inalámbrica vía bluetooth o infrarroja si tenemos un panel táctil con el que configurar y controlar todo y al cual accederemos con mucha menos frecuencia en comparación con el resto de dispositivos del hogar. Lo interesante para futuras aplicaciones o proyectos es establecer un control a través de una página web a la que podamos acceder mediante múltiples dispositivos, móvil, ordenador, tablet...

Para alcanzar este objetivo la idea es utilizar otro Funduino Mega con una shield Ethernet y que este módulo se comunique con nuestro sistema conectando sus puertos serie o bien empleando una comunicación inalámbrica para reducir el conexionado, lo que nos permitiría tratarlo como una expansión independiente cuya única finalidad seria el control vía Ethernet. Para ello se podría dotar de comunicación inalámbrica a nuestro sistema utilizando los pines desocupados y que así pueda recibir desde la expansión las modificaciones en el control que se realicen por internet.

Dicho todo esto, las opciones para la comunicación entre el sistema principal y la expansión serian:

- Cableado puerto serie
- Radiofrecuencia
- Bluetooth

La comunicación serie sería inmediatamente descartada en caso de no querer unificar el sistema con el módulo Ethernet. La comunicación bluetooth sería una buena idea pero se vería limitada por su alcance, nos obligaría tener el módulo Ethernet cerca de módulo principal y esto supondría un problema en caso de no tener cerca un puerto Ethernet.

Por ultimo está la opción de la radiofrecuencia que solucionaría el problema del alcance y podríamos utilizar módulos bidireccionales que conectaríamos a una de las dos UART disponibles. En el caso de la expansión tampoco tendríamos problemas porque la shield

Ethernet no utiliza todas las UART disponibles. Solo haría falta definir una trama para que no interfieran otras comunicaciones en la misma frecuencia.

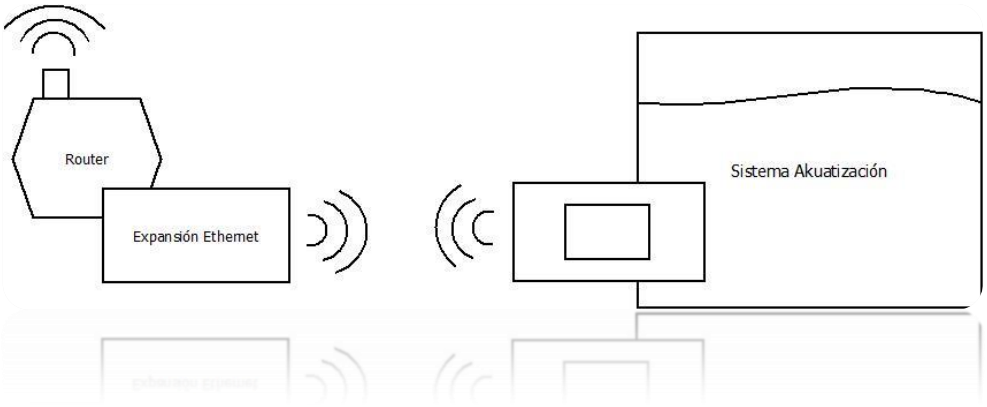


Ilustración 28 Esquema adición expansión Ethernet

Por otro lado con Ethernet Shield y Funduino se crearía un web server para poder controlar el sistema Akuatización a través de Internet. Para ello tendríamos que seguir cuatro pasos: [18]

1. Conseguir que conexión IP entre Funduino Ethernet Shield y el Router (área local) no cambie, para lograrlo hay que introducirse en la configuración del router y decirle que siempre le de la misma IP a la Ethernet Shield, ya que esta vendrá especificada por hardcode en el sketch y si cambia perderemos la conexión.

2. Abrir los puertos de tu Router para permitir que entre y salga información.

   Con el paso anterior se ha creado una red propia de área local, por tanto, si el proyecto únicamente necesitara conexión entre los elementos de casa (del Router hacia dentro) no se necesitaría configurar nada más. Ahora bien, como la idea es controlar el sistema desde cualquier parte del mundo se debe permitir el acceso a este.

3. Conseguir que la conexión entre tu Router e Internet no cambie (para que se pueda acceder siempre desde la misma dirección).

   Utilizar Un Servicio de DDNS, el sistema de las DDNS consiste básicamente en un dominio (www.akuatizacion.es) que está asociado en cada momento a tu IP actual, es decir, un nombre fijo elegido por ti (y por lo tanto más fácil de recordar que una dirección IP) que esté siempre actualizado y dirija la conexión hacia la Ethernet Shield.

4. Como último paso quedaría el código, que no será mayor problema si se tienen ciertos conocimientos de HTML y la librería Ethernet de Arduino.

ANEXO I – BIBLIOGRAFÍA

Revistas:

[1] Automatización eléctrica. Crogan, Patrick La automatización y digitalización de la vida cotidiana. Revista Científica de Estrategias, Tendencias e Innovación en Comunicación (2016).

Libros:

[2] Microcontroladores. Gunther Gridling, Bettina Weiss, Introduction to Microcontrollers, Vienna University of Technology (2007)

Páginas Web:

[3] Automatización eléctrica. [Consulta: 10 de Febrero del 2017] Disponible en <http://isbelg.over-blog.com>

[4] Microcontroladores. [Consulta: 10 de Febrero del 2017] Disponible en <http://www.microchip.com>

[5] Microcontroladores. [Consulta: 10 de Febrero del 2017] Disponible en <http://www.analog.com>

[6] I2C. [Consulta: 17 de Febrero del 2017] Disponible en <http://www.interfacebus.com>

[7] I2C. [Consulta: 17 de Febrero del 2017] Disponible en <http://www.i2c-bus.org>

[8] OneWire. [Consulta: 17 de Febrero del 2017] Disponible en <https://www.maximintegrated.com>

[9] OneWire. [Consulta: 17 de Febrero del 2017] Disponible en <http://www.1wire.info>

[10] SPI. [Consulta: 17 de Febrero del 2017] Disponible en <http://www.embedded.com>

[11] SPI. [Consulta: 17 de Febrero del 2017] Disponible en <http://www.epanorama.net/links/serialbus.html>

[12] SPI. [Consulta: 18 de Febrero del 2017] Disponible en <http://www.mct.net/faq/spi.html>

[13] Funduino. [Consulta: 28 de Enero del 2017] Disponible en <https://www.arduino.cc>

[14] Funduino. [Consulta: 28 de Enero del 2017] Disponible en <http://sabetecnologia.blogspot.com.es>

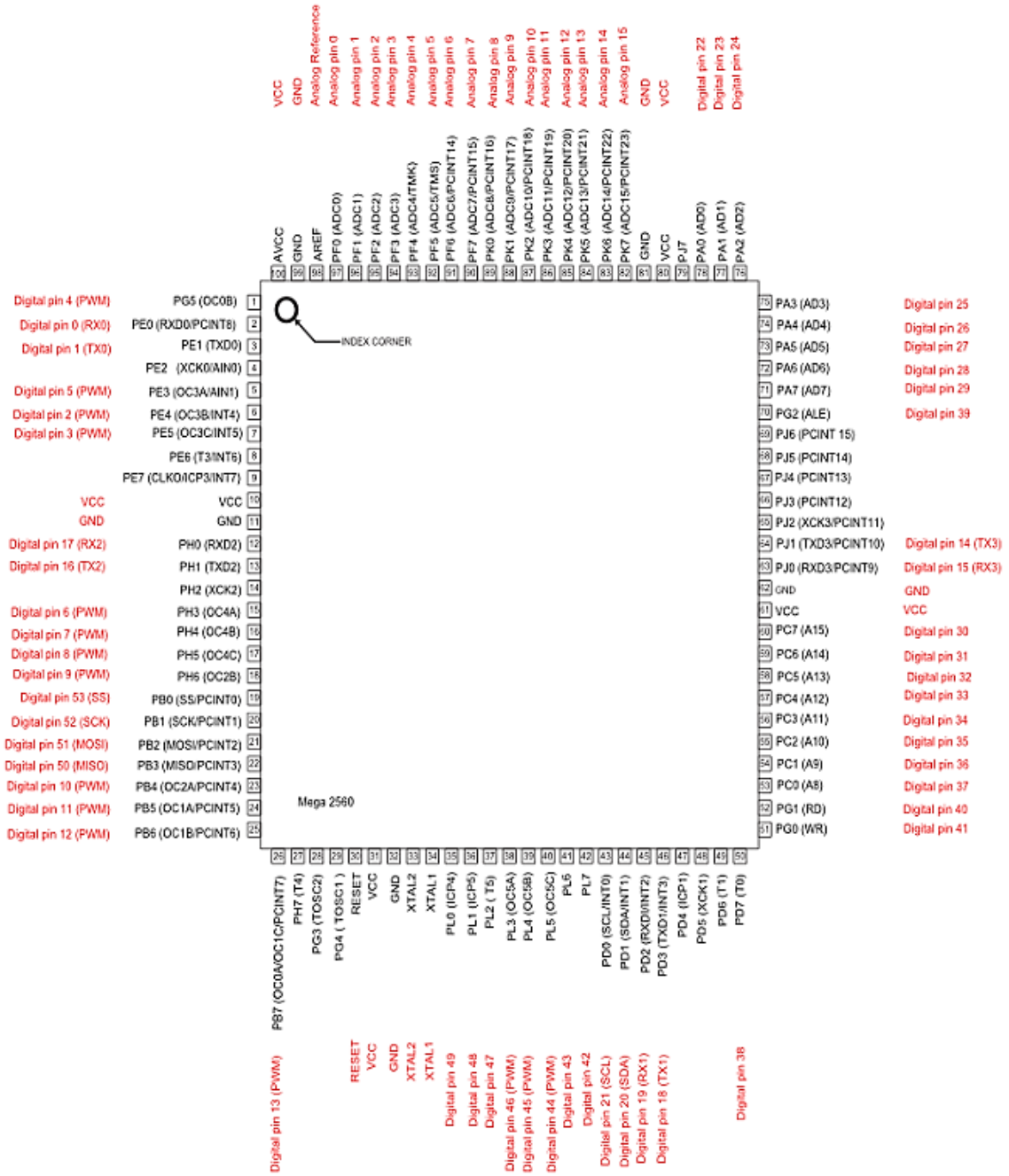
[15] Relés. [Consulta: 20 de Febrero del 2017] Disponible en <http://www.profetolocka.com.ar>

[16] Relés. [Consulta: 28 de Enero del 2017] Disponible en <http://www.prometec.net>

[17] DS18B20. [Consulta: 28 de Enero del 2017] Disponible en <https://www.luisllamas.es>

[18] Ethernet. [Consulta: 15 de Abril del 2017] Disponible en <http://www.educachip.com/arduino-ethernet-shield/>

ANEXO II – PINOUT ATMEGA2560



PRESUSPUESTO

Componente	**Precio**	**Precio +10**
Funduino Mega2560	9.37 €	7.24 €
Alimentación 9V – 1A	3.16 €	2.78 €
Cable jumper Macho - Hembra	2.64 €	2.32 €
RTC DS3231	1.66 €	1.46 €
Sensor Temperatura DS18B20	3.50 €	3.04 €
2.8" TFT LCD	10.06 €	7.85 €
Relés 4 canales	4.79 €	4.21 €
8 punteras 0.75 mm^2	0.15 €	0.12 €
Caja ABS	4.27 €	3.69 €
	39.60 €	**32.67 €**