

PROYECTO FIN DE GRADO

TÍTULO: Desarrollo y diseño de un seguidor solar

AUTOR/A: Ángel García Cortés

TITULACIÓN: Electrónica de Comunicaciones

TUTOR/A: Francisco Javier Jiménez Martínez

DEPARTAMENTO: Electrónica Física, Ingeniería Eléctrica y Física Aplicada

VºBº TUTOR/A

Miembros del Tribunal Calificador:

PRESIDENTE/A: Ignacio Antón Hernández

TUTOR/A: Francisco Javier Jiménez Martínez

SECRETARIO/A: Manuel Vázquez López

Fecha de lectura: 10 Junio 2024

Calificación:

El Secretario/La Secretaria,

Resumen

El presente Proyecto de Fin de Grado se centra precisamente en el **desarrollo y diseño de un seguidor solar**, con el objetivo de contribuir al avance de las energías renovables y ofrecer una solución práctica y efectiva para la captación de energía solar. La tecnología empleada en este proyecto se basa en el uso de microcontroladores Arduino, específicamente el Arduino UNO, junto con servomotores y sensores de luz para lograr un seguimiento preciso del movimiento del sol a lo largo del día.

En la actualidad, las energías renovables experimentan un notable auge y desempeñan un papel crucial en una revolución social en constante crecimiento, integrándose cada vez más en nuestra vida cotidiana. Dentro de estas fuentes de energía, la solar destaca como una de las más utilizadas, a pesar de presentar un rendimiento relativamente limitado en comparación con otras alternativas como puede ser la hidroeléctrica. Esto es debido a que la energía solar depende de diferentes factores climáticos, como pueden ser las estaciones del año, el clima del lugar y por supuesto, la posición del sol. La dependencia de combustibles fósiles y la creciente preocupación por el cambio climático han generado una demanda sin precedentes de soluciones energéticas más limpias y sostenibles entre las que se encuentra la energía solar.

Dentro de este panorama, el diseño y la implementación de seguidores solares emergen como soluciones clave para maximizar la captación de energía solar en sistemas fotovoltaicos. Estos dispositivos automatizados tienen la capacidad de orientar los paneles solares de manera dinámica, siguiendo el movimiento del sol a lo largo del día y optimizando así su exposición a la radiación solar incidente. Esta capacidad de seguimiento mejora significativamente la eficiencia de los sistemas fotovoltaicos al asegurar que los paneles estén siempre orientados de forma óptima para captar la máxima cantidad de energía solar disponible.

La necesidad de optimizar la orientación de los paneles solares para aumentar su eficiencia energética y su adaptación a las cambiantes condiciones ambientales motiva la investigación y desarrollo de esta tecnología.

El objetivo principal de este proyecto es abordar los desafíos asociados con el seguimiento automático de placas solares, con el propósito de optimizar la captación de radiación solar y mejorar la eficiencia energética del sistema fotovoltaico. Se busca desarrollar un algoritmo de control eficiente y confiable que permita al seguidor solar ajustar dinámicamente la posición de los paneles en función de la posición del sol maximizando la eficiencia del sistema. Esto se acompañará del desarrollo y estudio del hardware requerido para materializar un seguidor solar completamente funcional.

Se espera que como resultado de este proyecto se obtenga un seguidor solar capaz de mejorar significativamente la eficiencia de los sistemas fotovoltaicos, promoviendo así el uso de energía solar como una alternativa sostenible y limpia. Este proyecto no solo busca ofrecer una solución tecnológica innovadora, sino también sentar las bases para futuras investigaciones y aplicaciones en el campo de las energías renovables.

Abstract

The present Bachelor's Thesis project focuses precisely on the **development and design of a solar tracker**, aimed at advancing renewable energies and providing a practical and effective solution for capturing solar energy. The technology used in this project is based on the use of Arduino microcontrollers, specifically the Arduino UNO, along with servo motors and light sensors to achieve precise tracking of the sun's movement throughout the day.

Currently, renewable energies are experiencing a significant increase and playing a crucial role in a constantly growing social revolution, integrating more and more into our daily lives. Among these energy sources, solar energy stands out as one of the most used, despite presenting a relatively limited performance compared to other alternatives such as hydroelectric power. This is because solar energy depends on various climatic factors, such as the seasons, the climate of the location, and of course, the position of the sun. The dependence on fossil fuels and growing concerns about climate change have generated an unprecedented demand for cleaner and more sustainable energy solutions, including solar energy.

Within this context, the design and implementation of solar trackers emerge as key solutions to maximize the capture of solar energy in photovoltaic systems. These automated devices can orient solar panels dynamically, following the movement of the sun throughout the day and thus optimizing their exposure to incident solar radiation. This tracking capability significantly improves the efficiency of photovoltaic systems by ensuring that the panels are always optimally oriented to capture the maximum amount of available solar energy.

The need to optimize the orientation of solar panels to increase their energy efficiency and adapt to changing environmental conditions motivates the research and development of this technology.

The main objective of this project is to address the challenges associated with the automatic tracking of solar panels, aiming to optimize the capture of solar radiation and enhance the energy efficiency of the photovoltaic system. We aim to develop an efficient and reliable control algorithm that allows the solar tracker to dynamically adjust the position of the panels based on the sun's position, maximizing the system's efficiency. This will be accompanied by the development and study of the necessary hardware to implement a fully functional solar tracker.

It is expected that as a result of this project, a solar tracker capable of significantly improving the efficiency of photovoltaic systems will be obtained, promoting the use of solar energy as a sustainable and clean alternative. This project not only aims to offer an innovative technological solution but also to lay the groundwork for future research and applications in the field of renewable energies.

Índice de figuras

Figura 3-1: Potencia fotovoltaica en España [1].....	19
Figura 3-2: Satélite Vanguard en la actualidad [2]	20
Figura 3-3: Efecto fotoeléctrico ante radiación electromagnética [3].....	21
Figura 3-4: Efecto fotovoltaico [3]	21
Figura 3-5: Instalación fotovoltaica conectada a red [4].....	22
Figura 3-6: Instalación fotovoltaica aislada de la red [5]	23
Figura 3-7: Movimiento solar [6].....	25
Figura 3-8: Comparativa de eficiencia seguidor y sistema fijo [7]	26
Figura 3-9: Seguidores solares de uno y ambos ejes [8]	27
Figura 3-10: Ejemplo de algoritmo de seguimiento solar [9]	28
Figura 4-1: Esquema principal de montaje	29
Figura 4-2: Panel fotovoltaico instalado	31
Figura 4-3: V_F del diodo 1N5817 [10]	32
Figura 4-4: Montaje y conexión del módulo TP4056	33
Figura 4-5: Alimentación del módulo TP4056 [11]	34
Figura 4-6: Corriente de carga en función de resistencia programable [11].....	34
Figura 4-7: Resistencia programable y efecto en la corriente de carga [12]	34
Figura 4-8: Batería ion-litio 18650 de 4800mAh [13].....	38
Figura 4-9: Regulador de voltaje elevador [14]	39
Figura 4-10: Conexión del transistor 2N3904.....	40
Figura 4-11: Encapsulado indicado por el fabricante [16]	41
Figura 4-12: Voltaje de dropout para 200mA	42
Figura 4-13: Configuración y conexión del LM3940 [17].....	42
Figura 4-14: Situación de pines del encapsulado [17]	43
Figura 4-15: Pines del módulo ESP8266-01 [19].....	44
Figura 4-16: Conexión del ESP8266-01 con el adaptador [20].....	45
Figura 4-17: Divisor de tensión para la conexión de Tx-Rx.....	45
Figura 4-18: Servomotor SG90 [22]	49
Figura 4-19: Módulo LDR 106[23].....	50
Figura 4-20: Montaje módulos LDRs	50
Figura 4-21: Parte posterior del LCD 1602 [25]	51
Figura 4-22: LCD en su inicialización en modo TEST	52
Figura 4-23: LCD en modo Normal	52
Figura 4-24: Arduino UNO R3 pinout [26]	53
Figura 4-25: Sensor Shield V5.0 pinout [27]	55
Figura 4-26: Montaje del micropulsador.....	56
Figura 5-1: Máquina de estados	57
Figura 5-2: Estado elección de modo funcionamiento	58
Figura 5-4: Estado de inicialización del sistema	59
Figura 5-7: Estado para la realización de medidas de luz.....	60
Figura 5-10: Estado de activación del WiFi.....	62
Figura 5-11: Comando AT para la configuración de velocidad de transmisión	62

Figura 5-14: Estado de consulta al servidor NTP.....	64
Figura 5-19: Estado de realización de medidas T y H	66
Figura 5-21: Estado para el envío de datos por WiFi.....	67
Figura 5-24: Identificador y api key de ThingSpeak	68
Figura 5-27: Resultados mostrados en ThingSpeak	69
Figura 5-28: Estado de representación.....	70
Figura 5-31: Estado encargado del apagado del WiFi	71
Figura 5-32: Ángulos de inclinación y declinación módulo FV	72
Figura 5-34: Estado para el ajuste de ambos ejes.....	73
Figura 5-38: Estado para la ejecución del eje vertical.....	76
Figura 5-40: Estado de entrada al modo bajo consumo	77
Figura 6-1: Ejemplo movimiento motor	81
Figura 6-2: Código de bloques de ejemplo	82
Figura 6-3: Esquema analógico diseñado en TinkerCad	83
Figura 6-4: Elección de modo programado en bloques	83
Figura 6-5: Estado de elección de modo	84
Figura 6-6: Estado de inicialización del sistema	84
Figura 6-7: Inicialización de variables con código de bloques	85
Figura 6-8: Estado para la realización de medidas de luz.....	85
Figura 6-9: Calculo de medias de luz.....	86
Figura 6-10: Estados para activación WiFi y consulta NTP	86
Figura 6-11: Activación módulo WiFi, consulta NTP y medidas.....	87
Figura 6-12: Estados de medida, envío de datos, representación y apagado de WiFi	87
Figura 6-13: Realización de medidas, envío y representación.....	88
Figura 6-14: Estado para el ajuste de ambos ejes.....	88
Figura 6-15: Lógica de ajuste seguida para el ajuste de ambos servomotores	89
Figura 6-16: Lógica seguida para el ajuste un solo eje.....	89
Figura 6-17: Ajuste vertical y horizontal de los servomotores	90
Figura 7-1: Orientación panel solar con luz en su perpendicular	91
Figura 7-2: Orientación del panel solar desde los extremos	91
Figura 7-3: Medidas mostrada en el LCD, actualizada cada 90 segundos	92
Figura 7-4: Medidas vistas desde ThingSpeak	92
Figura 7-5: Demostración entrada bajo consumo	93
Figura 7-6: Ejemplo control de carga	94
Figura 9-1: Objetivos de desarrollo sostenible [41]	97
Figura 9-2: Emisiones de CO2 de diferentes energías	98
Figura 9-3: Panel solar en zonas rurales [42]	99
Figura 12-1: Acceso al gestor de librerías del Arduino IDE.....	111
Figura 12-2: Instalación manual de librería de Arduino.....	111
Figura 12-3: Configuración IDE Arduino para el uso de ESP8266.....	114
Figura 12-4: Selección de placa a codificar	114
Figura 12-5: Situación del micropulsador en la placa de prototipado.....	116
Figura 12-6: Búsqueda del canal por id de usuario.....	116

Índice de tablas

Tabla 1: Tipos de paneles solares	22
Tabla 2: Valores de corriente soportados por 1N5817 [10].....	32
Tabla 3: Tensión de reversa máxima soportada por el diodo 1N5817 [10].....	33
Tabla 4: Indicador de carga mediante Leds [11]	35
Tabla 5: Consumo en modo normal.....	37
Tabla 6: Corriente y tensión de salida en función de la entrada [15]	39
Tabla 7: Corriente de colector del 2N3904 [16].....	40
Tabla 8: Caída de tensión colector emisor [16].....	41
Tabla 9: Tensión a la salida del regulador [17]	42
Tabla 10: Voltaje y corriente en el sensor BME680 [21]	47
Tabla 11: Precisión de humedad según hoja de datos [21]	48
Tabla 12: Precisión de temperatura según hoja de datos [21]	48
Tabla 13: Materiales adquiridos y precios.....	96
Tabla 14: Presupuesto inicial	96

Lista de acrónimos

La lista de acrónimos incluida en este capítulo debe estar ordenada alfabéticamente.

ADC (Analog Digital Converter)

API (Application Programming Interface)

AT (Attention)

BOD (Brownout Detector)

ETSIT (Escuela Técnica Superior de Ingenieros de Telecomunicación)

IDE (Integrated Development Environment)

I2C (Inter-Integrated Circuit)

IF (Forward Current Average)

IFAV (Forward Current Average)

IP (Internet Protocol)

LCD (Liquid-Crystal Display)

LDR (Light Dependant Resistor)

NTP (Network Time Protocol)

ODS (Objetivos de Desarrollo Sostenible)

PCB (Printed Circuit Board)

PFG (Proyecto Fin de Grado)

PWM (Pulse Width Modulation)

SPI (Serial Peripheral Interface)

TBD (To Be Defined)

UTC (Coordinated Universal Time)

USB (Universal Serial Bus)

VF (Voltage Forward)

VRRM (Reverse Repetitive Voltage)

WiFi (Wireless Fidelity)

Índice de contenidos

Resumen	2
Abstract	3
Índice de figuras	4
Índice de tablas	7
Lista de acrónimos.....	8
1 INTRODUCCIÓN.....	13
1.1 Marco y motivación del proyecto	13
1.2 Objetivos técnicos y académicos	14
1.3 Estructura del resto de la memoria	15
2 OBJETIVOS Y ESPECIFICACIONES DEL PROYECTO	17
2.1 OBJETIVOS DEL PROYECTO	17
2.2 ESPECIFICACIONES DEL PROYECTO.....	17
3 ESTADO DEL ARTE	19
3.1 Historia y funcionamiento placas fotovoltaicas.....	19
3.1.1 Historia	20
3.1.2 Funcionamiento.....	20
3.2 Tipo de instalaciones fotovoltaicas.....	22
3.2.1 Instalaciones fotovoltaicas conectadas a red.....	22
3.2.2 Instalaciones fotovoltaicas aisladas de la red	23
3.3 Movimiento solar.....	23
3.3.1 Inclinação solar.....	24
3.3.2 Declinación solar	24
3.3.3 Ángulo azimutal.....	24
3.4 Rendimientos placas fotovoltaicas	25
3.5 Seguidores solares	26
3.5.1 Seguidores de un solo eje.....	26
3.5.2 Seguidores de ambos ejes	26
3.5.3 Seguidores de concentración	27
4 DESCRIPCIÓN DEL EQUIPAMIENTO UTILIZADO	29
4.1 Hardware	30
4.1.1 Placa fotovoltaica	30
4.1.2 Diodo protección 1N5817	31
4.1.3 Módulo carga TP4056.....	33

4.1.4	Consumo del sistema	35
4.1.5	Batería recargable	37
4.1.6	Regulador elevador 5V	39
4.1.7	Transistor bipolar 2N3904.....	40
4.1.8	Regulador 3.3V (LM3940).....	41
4.1.9	Módulo WiFi ESP8266-01	43
4.1.10	Sensor BME680	46
4.1.11	Servomotores.....	48
4.1.12	Módulo de LDRs	49
4.1.13	LCD 1602	50
4.1.14	Arduino UNO y Sensor Shield.....	53
4.1.15	MICROPULSADOR.....	55
4.2	Lenguaje de Programación	56
5	APLICACIÓN SOFTWARE DESARROLLADA	57
5.1	Elección de modo.....	58
5.2	Init	59
5.3	Medidas de luz	60
5.4	Activar WiFi.....	61
5.4.1	Configuración de comunicación serial.....	62
5.4.2	Activación 2N3904.....	62
5.4.3	Conexión WiFi.....	63
5.5	Consulta NTP.....	63
5.5.1	Envío y procesamiento de datos al Arduino UNO	64
5.6	Medida temperatura y humedad	65
5.7	Enviar datos WiFi	66
5.8	Representación	69
5.9	Apagar WiFi.....	71
5.10	Condición ejecución diaria.....	72
5.10.1	¿Por qué este método?.....	72
5.10.2	Código implementado.....	72
5.11	Ajuste ambos ejes α y β	73
5.11.1	Función general.....	73
5.11.2	Función para determinar cantidad de luz suficiente	74
5.11.3	Función para limitar movimiento horizontal y vertical.....	74

5.11.4	Función para movimiento de ambos ejes	75
5.12	Ajuste vertical β	76
5.13	Bajo consumo	77
5.14	Especificaciones extra	78
5.14.1	Evitar demanda de corriente excesiva	78
5.14.2	Evitar interferencia en la comunicación serial	78
5.14.3	Asegurar obtención del día actual	79
6	Aplicación software por bloques en TinkerCad	81
6.1	Elección de modo en código de bloques	83
6.2	Estado Init en código de bloques	84
6.3	Cálculo de medias de luz con bloques	85
6.4	Activación WiFi y consulta NTP	86
6.5	Medidas, envío de datos WiFi y representación	87
6.6	Lógica de ajuste y modo bajo consumo	88
6.7	Ajuste vertical/horizontal con bloques	89
7	RESULTADOS	91
7.1	Seguimiento de luz	91
7.2	Realización de medidas y representación	92
7.3	Entrada bajo consumo	93
7.4	Control de carga de la batería	93
7.5	Conclusión	94
8	PRESUPUESTO	95
9	IMPACTO DEL PROYECTO	97
10	CONCLUSIONES, MEJORAS Y TRABAJOS FUTUROS	101
10.1	Conclusiones	101
10.2	Mejoras y trabajos futuros	101
11	REFERENCIAS	105
12	BIBLIOGRAFÍA	109
Anexos		111
A.1	Manual de instalación	111
A.1.1.	Librerías usadas por el Arduino UNO	112
A.1.2.	Librerías usadas por el ESP8266-01	113
A.2	Manual de usuario	115
A.2.1.	Ejecución modo TEST	115

A.2.2.	Ejecución modo normal.....	116
--------	----------------------------	-----

1 INTRODUCCIÓN

1.1 Marco y motivación del proyecto

En la actualidad, el uso de las energías renovables es cada vez más fuerte, y la energía solar juega un papel fundamental en la transición hacia un futuro más sostenible por diferentes razones entre las que se encuentran:

- Es considerada una fuente virtualmente inagotable de energía ya que proviene del sol, esto la hace mucho más atractiva que los combustibles fósiles.
- Tiene una baja huella ambiental ya que no produce emisiones de efecto invernadero.
- Tecnología asequible y accesible debido a la reducción del costo de instalación de los últimos tiempos y a diferentes subvenciones proporcionadas por el gobierno y las eléctricas.

Sin embargo, a pesar de sus beneficios ambientales y su potencial para reducir la dependencia de combustibles fósiles, la eficiencia de la energía solar sigue siendo un desafío debido a su dependencia de factores ambientales cambiantes, entre los que se encuentran:

- Condiciones meteorológicas como la presencia de nubes, lluvias, nevadas que pueden obstaculizar la cantidad de luz que reciba el panel solar.
- Ángulo y orientación del sol, la eficiencia de los paneles depende en gran medida de su orientación y ángulo con respecto al sol el cuál varía su ángulo de incidencia según las estaciones.
- Contaminación atmosférica como las partículas en suspensión pueden reducir la claridad del cielo y por tanto la radiación recibida por el panel.

Debido a las desventajas e inconvenientes comentados anteriormente, surge la necesidad de desarrollar tecnologías que permitan superar en parte estas limitaciones para maximizar el aprovechamiento de la energía solar. Una de estas tecnologías es el seguidor solar, ya que permite la orientación de paneles solares maximizando su exposición a la radiación solar a lo largo del día y las estaciones del año.

El objetivo principal de este proyecto es precisamente abordar la creación y desarrollo de un sistema de seguimiento solar eficiente y automatizado a una pequeña escala inicialmente, con la característica de que sea escalable a futuro. El seguidor desarrollado mejorará significativamente la captación de radiación solar y, por tanto, optimizará la generación de energía. Al implementar un sistema de seguimiento solar, se busca contribuir al avance de la tecnología fotovoltaica y fomentar la adopción de energía solar como una fuente de energía limpia y sostenible.

Es importante señalar que el seguidor solar que se desarrollará estará diseñado para una placa solar de tamaño reducido, lo que facilitará la realización de diversas pruebas de manera más ágil y efectiva. Además, se considerará que el algoritmo diseñado tendrá la capacidad de adaptarse y ser aplicado en diferentes tipos de seguidores solares, independientemente de su escala o aplicación específica.

Este proyecto reconoce la existencia de una amplia gama de códigos y sistemas de seguimiento solar disponibles en internet. Sin embargo, se enfocará en desarrollar una solución que destaque por su sencillez, eficiencia energética, seguridad en la gestión de la carga y descarga de baterías, y su facilidad de uso, especialmente en el contexto académico. La prioridad radicará en crear un sistema accesible y robusto, que pueda servir como una herramienta educativa efectiva para estudiantes y profesionales interesados en el campo de las energías renovables y la automatización.

Por lo tanto, este proyecto utiliza la innovación tecnológica y la necesidad urgente de encontrar soluciones energéticas más eficientes y respetuosas con el medio ambiente.

1.2 Objetivos técnicos y académicos

Los objetivos de este proyecto fin de carrera son, desde el punto de vista técnico:

- Diseño y desarrollo del seguidor solar funcional que utilice tecnología Arduino y algoritmos optimizados de tal manera que sea capaz de seguir la posición del sol de manera precisa y eficiente.
- Eficiencia energética, se priorizará asegurando que la orientación dinámica de los paneles solares permita una captación óptima de la radiación solar a lo largo del día y estaciones del año.
- Simplicidad y usabilidad, se procurará diseñar un sistema que sea fácil de usar y mantener para un futuro.

Desde el punto de vista académico, el proyectista adquiere una serie de competencias y habilidades fundamentales, entre las que se incluyen:

- Conocimientos prácticos sobre el desarrollo de un proyecto de esta envergadura, afianzando así conocimientos teóricos adquiridos durante la formación académica, especialmente en el área electrónica de control y energías renovables.
- Capacidad de investigación y desarrollo, debido a que previo al desarrollo del seguidor solar se tendrá que realizar una labor de investigación, análisis y resolución de diversas dudas sobre el funcionamiento básico de estos sistemas.
- Habilidad de resolución de errores que surjan a lo largo del desarrollo del proyecto, así como su análisis.
- Capacidad de innovación y creatividad, debido a que se tendrán que buscar soluciones eficientes y seguras para los desafíos técnicos planteados en el proyecto, explorando nuevas ideas y enfoques en el campo de la energía solar.
- Competencias en comunicación y presentación por la realización de los diferentes informes como el anteproyecto y la memoria final, así como la lectura del proyecto, estas deben tener una comunicación escrita y oral clara y convincente.

Este proyecto fin de carrera no solo tiene como objetivo desarrollar un seguidor solar funcional y eficiente, sino también proporcionar una experiencia enriquecedora y formativa que contribuya al crecimiento profesional y académico en el campo de la ingeniería y las energías renovables.

1.3 Estructura del resto de la memoria

En los próximos capítulos de la memoria se ofrecen información sobre los siguientes puntos:

1. Especificaciones, se detallarán las especificaciones técnicas y funcionales del sistema a desarrollar.
2. Objetivos del proyecto, en este punto se tratarán los objetivos específicos que se pretenden alcanzar con el desarrollo del proyecto.
3. Estado del arte, se realizará un análisis del estado actual de la tecnología utilizada en el proyecto, identificando los avances más relevantes y estableciendo el contexto para el desarrollo de la solución propuesta.
4. Descripción del equipamiento utilizado, en este apartado se tratarán los diferentes materiales hardware utilizados en la implementación del proyecto, así como una explicación de su funcionamiento y características principales.
5. Aplicación software desarrollada, se describirá la aplicación software creada para el control y gestión del sistema, explicando su diseño, funcionamiento y mostrando el código más clave e interesante para el proyecto.
6. Aplicación software por bloques en TinkerCad, proporciona una descripción software desarrollada utilizando la codificación por bloques aportada por la plataforma TinkerCad la cual aporta una interfaz sencilla y comprensiva para su posterior uso en la docencia.
7. Resultados, se presentarán los resultados obtenidos durante la ejecución del proyecto, evaluando el correcto cumplimiento de los objetivos fijados inicialmente.
8. Presupuesto, en este punto se detallará el presupuesto estimado para la realización del proyecto, incluyendo materiales, equipos y recursos necesarios.
9. Impacto del proyecto, en este apartado se discuten las implicaciones sociales, tecnológicas y ambientales del proyecto.
10. Conclusiones, mejoras y trabajos futuros, se presentarán las conclusiones del trabajo realizado y se proponen posibles mejoras para el sistema desarrollado.
11. Referencias y bibliografía, en este punto se incluyen las fuentes consultadas durante la elaboración de la memoria y el proyecto.
12. Anexos, este apartado contiene el manual de instalación para posibles modificaciones futuras así como el manual de usuario para el correcto funcionamiento del sistema.

Esta página se ha dejado en blanco intencionadamente

2 OBJETIVOS Y ESPECIFICACIONES DEL PROYECTO

2.1 OBJETIVOS DEL PROYECTO

El objetivo principal de este proyecto es diseñar e implementar un seguidor solar funcional utilizando tecnología Arduino con un algoritmo optimizado. Este dispositivo estará diseñado para seguir automáticamente la posición del sol a lo largo del día y durante las estaciones del año, ajustando la orientación de los paneles solares para maximizar la captación de energía solar. La implementación de este seguidor solar permitirá aumentar y maximizar la eficiencia energética del panel solar instalado en su estructura, lo que resultará en una mayor producción de energía eléctrica a partir de la energía solar disponible.

Además de la función de seguimiento solar, el sistema también estará equipado con sensores de temperatura y humedad ambiental, que se integrarán para proporcionar mediciones ambientales en tiempo real. Estos datos se podrán visualizar tanto en un LCD (Liquid-Crystal Display o pantalla de cristal líquido) local como a través de una interfaz web, lo que permitirá un monitoreo remoto y conveniente del entorno donde se encuentra el sistema de seguimiento solar.

La finalidad de este proyecto es abordar la necesidad de optimizar la orientación de los paneles solares, especialmente en sistemas fotovoltaicos de menor escala donde la eficiencia energética es crucial. La energía fotovoltaica es una de las fuentes de energía renovable con menor rendimiento debido a las variaciones en la radiación solar y la orientación subóptima de los paneles solares. Por lo tanto, el modelo diseñado proporcionará una solución eficiente para mejorar la captación de energía solar y, en última instancia, contribuirá a la creación de un sistema de medida autónomo y más eficiente desde el punto de vista energético.

2.2 ESPECIFICACIONES DEL PROYECTO

El sistema de seguimiento solar que se desarrollará en este PFG cumplirá con una serie de especificaciones detalladas para garantizar su funcionalidad y eficiencia. A continuación, se detallan estas especificaciones de manera más extensa:

1. El sistema utilizará la tecnología Arduino y se programará en el lenguaje de programación C++ para el control del seguimiento solar. Esta elección se basa en la versatilidad y la amplia comunidad de desarrollo que ofrece Arduino, lo que facilitará la implementación y el mantenimiento del sistema.
2. El sistema estará diseñado para seguir automáticamente la posición del sol de manera óptima, lo que permitirá aumentar la captación de energía solar. Esto se logrará mediante algoritmos de seguimiento solar que ajustarán la orientación de los paneles solares en función de la posición del sol a lo largo del día y las estaciones del año.
3. Se emplearán servomotores para realizar los movimientos necesarios de los paneles solares, lo que permitirá ajustes precisos y rápidos en la orientación de los paneles en respuesta a los cambios en la posición del sol.

4. El código desarrollado para el sistema será eficiente, compacto, mantenible y ampliable a futuro. Se seguirán las mejores prácticas de programación para garantizar un código limpio y fácil de entender, lo que facilitará su mantenimiento y futuras actualizaciones.
5. Se desarrollará código de bloques para el sistema, lo que permitirá su posterior uso en la docencia. Esta aproximación facilitará la comprensión y el aprendizaje de los conceptos relacionados con el seguimiento solar y la programación en Arduino.
6. Se utilizará una batería de ion-litio de 4800mAh para alimentar el circuito de medida y representación del sistema. Esta batería será capaz de recargarse mediante la energía fotovoltaica, lo que garantizará la autonomía y la sostenibilidad del sistema. Se espera que el consumo del sistema sea menor de 100mAh para una autonomía significativa.
7. Se integrará un sensor de temperatura y humedad ambiental con la suficiente fiabilidad para proporcionar mediciones precisas del entorno donde se encuentra el sistema de seguimiento solar.
8. Se implementará un sistema de lectura y procesamiento de datos de los sensores que permitirá la verificación y el análisis de los datos recopilados. La resolución de las medidas será del orden de las centésimas, y se buscará reducir el consumo a menos de 50mA en el modo bajo consumo para que la recarga de la batería sea eficiente.

3 ESTADO DEL ARTE

Previo a la realización de un proyecto basado en un seguidor solar, se debe realizar una investigación sobre las diferentes tecnologías que lo componen, de cómo han ido avanzando en el tiempo y de cuál es su estado actual, así como de los efectos físicos que afectan al rendimiento de los seguidores.

Los temas sobre los que se ha investigado para tener la suficiente información son los siguientes:

- La historia y funcionamiento de las placas fotovoltaicas, para entender cómo funcionan a nivel físico y eléctrico para convertir luz solar en electricidad.
- Tipo de instalaciones fotovoltaicas para entender las diferentes aplicaciones y configuraciones de los sistemas solares.
- Movimiento solar para comprender el cambio de la posición solar a lo largo del día y año.
- Rendimientos de instalación de placas fotovoltaicas.
- Necesidad y tipo de seguidores solares.

3.1 Historia y funcionamiento placas fotovoltaicas

Las placas fotovoltaicas, aunque es una tecnología en auge que en los últimos años ha evolucionado y reinventado mucho como se muestra en la Figura 3-1, realmente es una tecnología que se lleva utilizando desde hace mucho tiempo, sobre todo en España.

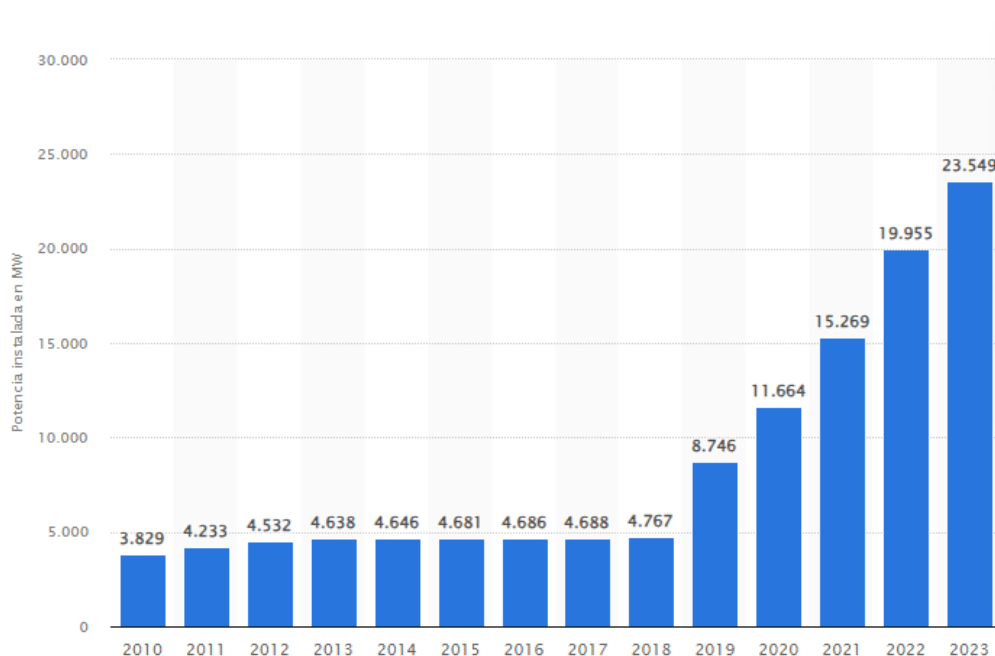


Figura 3-1: Potencia fotovoltaica en España [1]

3.1.1 Historia

En sus inicios la energía fotovoltaica fue pensada para el espacio, ya que era una manera muy inteligente de alimentar los diferentes satélites de hecho se realizó el lanzamiento de los satélites Sputnik y Vanguard en 1958 con 6 células de 1W como se muestra en la Figura 3-2.



Figura 3-2: Satélite Vanguard en la actualidad [2]

Sin embargo, con el paso del tiempo se consideró llevar esta tecnología a zonas rurales con poco acceso a la red eléctrica.

Los primeros paneles fotovoltaicos se crearon en 1883 por Charles Fritts, utilizando selenio con una capa de oro consiguió una eficiencia del panel del 1%, pese a su pobre rendimiento causó una revolución en el ámbito científico. Con el paso del tiempo, en 1954 los investigadores de Bell Labs crearon lo que hoy conocemos como una célula solar, con una eficiencia del 6%. Como se puede ver es una eficiencia mínima, por lo que por esa época su uso quedo relegado por completo y no fue hasta la década de los 70 que se pudo empezar a hablar de una revolución energética.

La primera instalación española fue en 1985, aunque el uso doméstico de paneles solares se remonta incluso más en el tiempo hasta 1973 cuando se creó la primera célula solar de silicio en el laboratorio de semiconductores de la E.T.S.I.Telecomunicación (Escuela Técnica Superior de Ingenieros).

3.1.2 Funcionamiento

La energía solar fotovoltaica es la energía que se llega a obtener al convertir luz solar en electricidad empleado el efecto fotoeléctrico que es el fenómeno físico por el cual algunos materiales son capaces de absorber fotones y liberar electrones para generar una corriente eléctrica, como se muestra en la Figura 3-3.

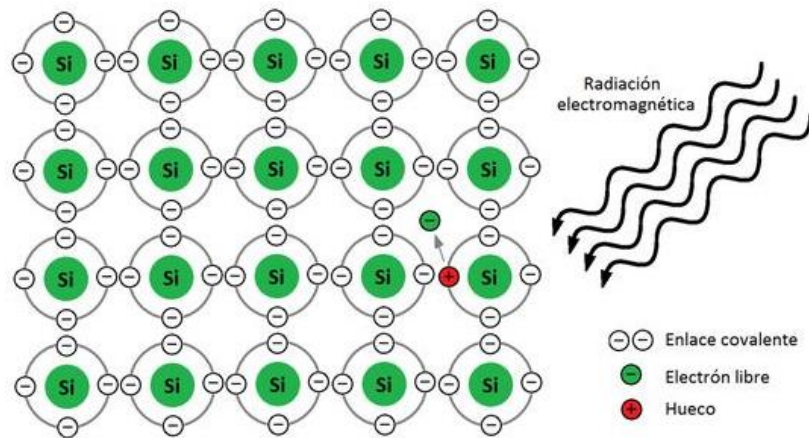


Figura 3-3: Efecto fotoeléctrico ante radiación electromagnética [3]

Aparte del efecto fotoeléctrico otro fenómeno clave es el efecto fotovoltaico, el cual se produce cuando dos piezas de material semiconductor, una de tipo N y otra de tipo P, están en contacto y expuestas a la luz o a cualquier radiación electromagnética. Estos dos materiales se combinan para formar una unión PN, donde la zona N está dopada con electrones adicionales y la zona P está dopada con huecos (falta de electrones). Los electrones se desplazarán hacia la zona P y los huecos hacia la zona N, creando una diferencia de potencial eléctrico a través de la unión como se muestra en la Figura 3-4.

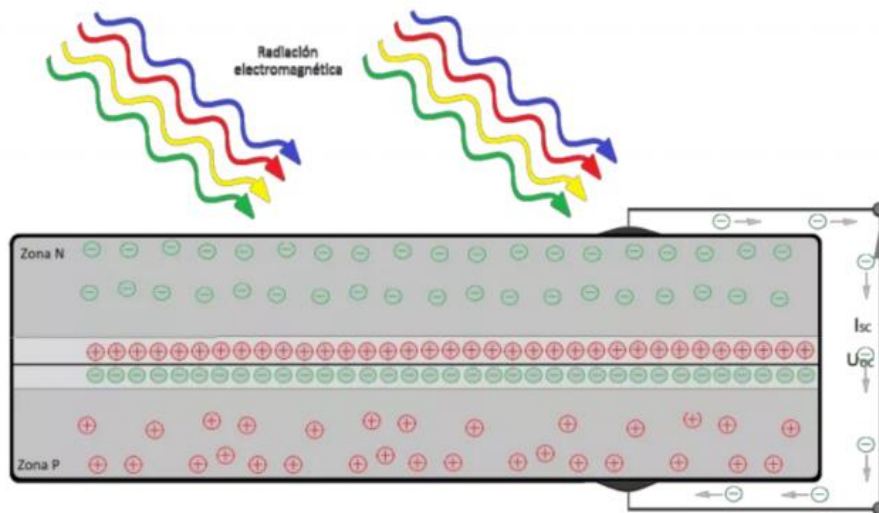


Figura 3-4: Efecto fotovoltaico [3]

Los materiales más usados para la fabricación de paneles solares es el Silicio, debido a su gran abundancia y estabilidad, aunque se podrían utilizar otros materiales que aporten una mayor eficiencia, el precio y facilidad de fabricación del Silicio lo hacen la mejor opción sin duda. Dentro de la fabricación de placas fotovoltaicas de silicio, las hay de diferentes tipos, cada uno con sus ventajas y desventajas mostradas en la Tabla 1:

Tipo	Coste	Rendimiento	Estabilidad
Monocristalinas	Alto	$\leq 27\%$	Mucha
Policristalinas	Alto	$\leq 24\%$	Mucha
Silicio amorfo	Bajo	$\leq 15\%$	Poca
Orgánicas	Bajo	$\leq 9\%$	Poca

Tabla 1: Tipos de paneles solares

3.2 Tipo de instalaciones fotovoltaicas

Las instalaciones fotovoltaicas se pueden dividir en dos grandes tipos en función de su conexión a red:

3.2.1 Instalaciones fotovoltaicas conectadas a red

En este tipo de instalaciones, los paneles solares están conectados directamente a la red eléctrica convencional, inicialmente la energía generada se utiliza primero para satisfacer las necesidades de la vivienda o edificio dónde están instalados, y el excedente irá a la red como se muestra en la Figura 3-5:

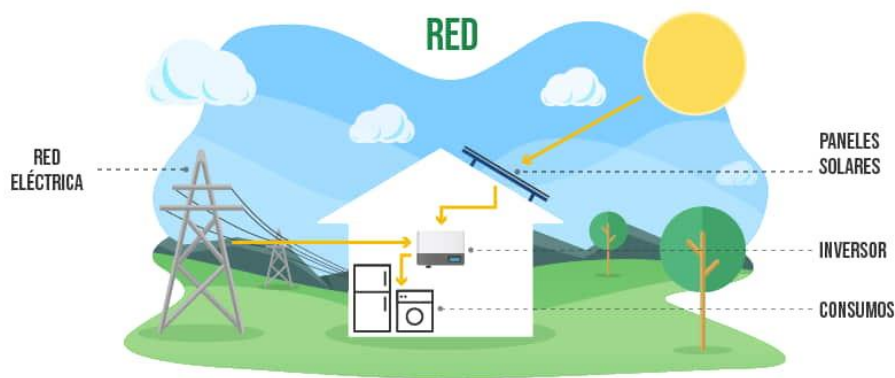


Figura 3-5: Instalación fotovoltaica conectada a red [4]

Se caracterizan por ser sistemas muy fiables ya que en caso de que la generación de electricidad obtenida no sea suficiente para el abastecimiento de la vivienda o lugar de instalación se podrá consumir de la red eléctrica. De esta manera, en ningún momento se quedará sin suministro eléctrico y no es necesario el uso de baterías lo que simplifica y abarata este tipo de instalaciones.

Este tipo de instalación requiere de un inversor, el cual será el encargado de convertir la corriente continua a corriente alterna, que es la forma de electricidad que maneja la red eléctrica. Además, el inversor es capaz de maximizar la eficiencia (mediante seguidores del punto de máxima potencia) y adaptarse a las condiciones cambiantes de luz solar. Sin embargo, este módulo es bastante costoso lo que hace que las instalaciones fotovoltaicas conectadas a red disparen su precio pese a las subvenciones proporcionadas en los últimos tiempos.

Las grandes plantaciones fotovoltaicas suelen ser en su inmensa mayoría instalaciones conectadas a red, ya que lo que interesa en estos casos es la venta de la energía generada. Es

en estas instalaciones en las que se realizan instalaciones de seguidores solares para maximizar la producción de energía y por tanto los beneficios.

3.2.2 Instalaciones fotovoltaicas aisladas de la red

Este tipo de instalaciones están pensadas para dar soporte a lugares a los que la red eléctrica no puede llegar y por tanto son sitios aislados. En estos casos la energía producida por los paneles solares se usa íntegramente en la vivienda o edificio instalado, almacenando el excedente en baterías como se muestra en la Figura 3-6. Sin embargo, este tipo de instalaciones son menos fiables, debido a que en el caso de no producir suficiente energía para el consumo no se puede utilizar la red eléctrica y se podría quedar sin suministro eléctrico.



Figura 3-6: Instalación fotovoltaica aislada de la red [5]

La clave de las instalaciones fotovoltaicas aisladas es el sistema de almacenamiento de energía, que puede consistir en baterías de plomo-ácido, baterías de litio u otros sistemas de almacenamiento de energía renovable. Estos sistemas almacenan la energía generada durante el día para su uso durante la noche o en días nublados, garantizando un suministro continuo de electricidad.

Aunque las instalaciones fotovoltaicas aisladas no requieren inversores, ya que la electricidad se utiliza directamente en corriente continua, todavía necesitan componentes de control de carga y descarga para garantizar un funcionamiento seguro y eficiente del sistema. Estos sistemas pueden ser más simples en términos de componentes, pero deben ser diseñados cuidadosamente para satisfacer las necesidades específicas de cada aplicación y maximizar la eficiencia energética.

3.3 Movimiento solar

El movimiento solar es fundamental para comprender la orientación óptima de los paneles solares en un seguidor solar. Este movimiento se refiere al desplazamiento del sol a lo largo del día y durante las estaciones del año debido a la rotación de la Tierra sobre su eje y su traslación alrededor del sol.

Durante el día, el sol se eleva desde el horizonte en el este, alcanza su punto más alto en el cielo al mediodía solar y luego desciende hacia el horizonte en el oeste. Además, el sol se mueve de este a oeste a lo largo del ecuador celeste a medida que cambian las estaciones del año, lo que resulta en variaciones en la duración del día y la altura del sol en el cielo.

Para maximizar la captación de energía solar, es crucial orientar correctamente los paneles solares para que estén perpendiculares a los rayos solares incidentes en todo momento, esto se logra comprendiendo varios conceptos fundamentales como son la inclinación, la declinación solar y el ángulo azimutal.

3.3.1 Inclinación solar

La inclinación solar es el ángulo que forma la superficie de una placa solar y la línea perpendicular al suelo β . Es este ángulo el que determina cómo los rayos solares inciden sobre la superficie de la placa, la inclinación óptima va variando en función de la situación geográfica, estación del año, hemisferio y hora del día.

3.3.2 Declinación solar

La declinación solar es la medida del ángulo entre el plano del ecuador terrestre y la dirección de los rayos solares y que inciden sobre la Tierra en un momento dado. La declinación solar varía a lo largo del año debido a la inclinación del eje de rotación de la Tierra en relación con su órbita alrededor del sol y alcanza sus valores máximos y mínimos en el solsticio de verano e invierno respectivamente.

3.3.3 Ángulo azimutal

Es el ángulo que indica la posición del seguidor solar en relación con la dirección sur y se refiere a la medida del ángulo horizontal desde el punto cardinal sur hasta la dirección en la que están orientados los paneles solares.

Un ejemplo de dichos ángulos y del movimiento solar se muestra en la Figura 3-7

Inclinación y Orientación de los Paneles Solares

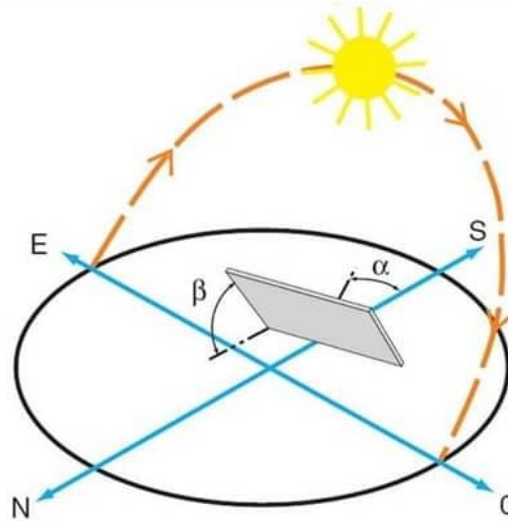


Figura 3-7: Movimiento solar [6]

3.4 Rendimientos placas fotovoltaicas

Como se ha comentado previamente el rendimiento de las placas fotovoltaicas es realmente bajo, sobre todo si las comparamos con otras fuentes de energía como la hidráulica. Además, el rendimiento de los módulos fotovoltaicos se ve afectado por diferentes motivos entre los que se encuentran:

1. Sombreamiento:

La presencia de sombras parciales puede reducir significativamente su rendimiento. Si una sola parte del panel está sombreada puede afectar negativamente al rendimiento de todo el conjunto.

2. Temperatura:

Un aumento de temperatura ambiente puede reducir la eficiencia de las células fotovoltaicas, cuanto mayor sea la temperatura menor será la eficiencia de estas.

3. Orientación fija:

A medida que el sol se mueve a lo largo del día y durante las estaciones del año, los paneles fijos pueden no estar siempre en la mejor posición para aprovechar la luz solar incidente.

Por tanto, introducir un seguidor solar puede ayudar a abordar algunos de estos problemas y mejorar el rendimiento de las placas fotovoltaicas como se ha demostrado en numerosos estudios [7]. La Figura 3-8 muestra una comparativa de eficiencia entre un sistema fijo y uno con seguidor solar, siendo la eficiencia del seguidor solar superior en todos los casos.

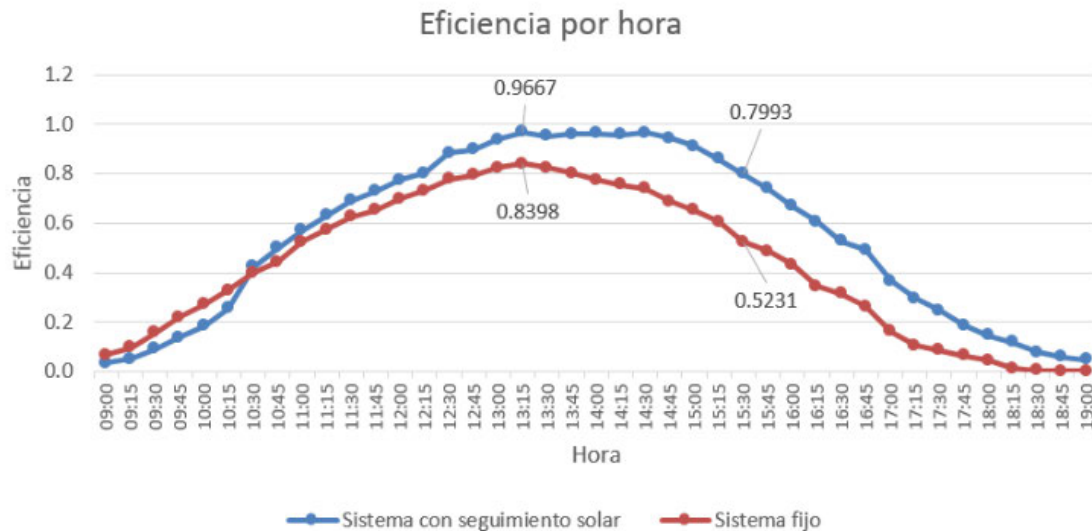


Figura 3-8: Comparativa de eficiencia seguidor y sistema fijo [7]

3.5 Seguidores solares

Los seguidores solares son estructuras móviles cuyo objetivo es aumentar la eficiencia y producción de electricidad en una instalación fotovoltaica.

Existen tres grandes tipos de seguidores solares, aquellos que realizan ajustes en un solo eje, los que realizan ajustes en ambos ejes y de concentración los cuales usan espejos o lentes.

3.5.1 Seguidores de un solo eje

Son los seguidores más sencillos y económicos, realizan el ajuste en un único eje, el cual suele ser el eje β , ya que es el que aporta una mayor eficiencia en la mayoría de los casos. Estos seguidores tienen la ventaja de ser menos aparatosos y voluminosos, por tanto permiten la instalación de más paneles fotovoltaicos aprovechando el espacio de instalación y haciéndolos más beneficiosos.

3.5.2 Seguidores de ambos ejes

Son dispositivos complejos que ajustan la orientación de los paneles tanto en el eje β como α . Son más eficientes que los seguidores de un solo eje pero en cambio requieren una inversión económica más grande, así como un área de instalación mayor.

Tanto el seguidor solar de un eje como el de ambos ejes mostrados en la Figura 3-9 son los más utilizados en la industria, sobre todo en instalaciones fotovoltaicas.

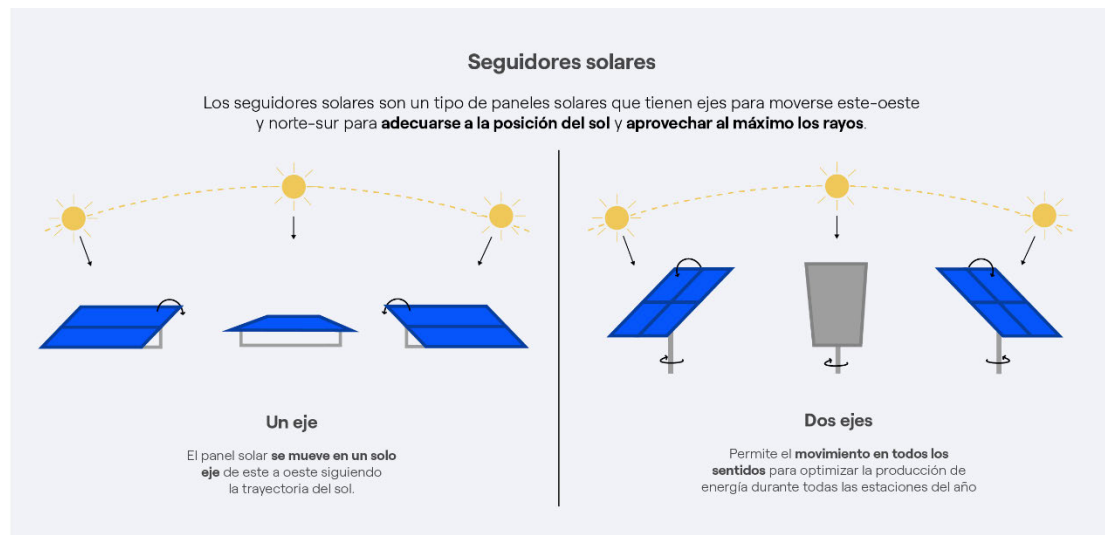


Figura 3-9: Seguidores solares de uno y ambos ejes [8]

3.5.3 Seguidores de concentración

Estos sistemas utilizan espejos o lentes para concentrar la luz solar en un punto llamado foco más pequeño, concentrando de esta manera la radiación y aumentando bastante la eficiencia. Estos pueden ser de uno o dos ejes, el inconveniente es que son muy aparatosos de instalar y el aumento de eficiencia que aportan no merece la pena.

Con respecto al software hay una gran cantidad de información sobre diferentes métodos y algoritmos para la realización de un seguidor solar. El algoritmo general utilizado tanto para un seguidor solar grande como uno pequeño es aproximadamente igual, la única diferencia reside en el monitoreo de los grandes seguidores para tener un control más exhaustivo de su rendimiento y funcionamiento. Un enfoque general e interesante es el mostrado en la Figura 3-10, donde se puede observar un algoritmo centrado en el ajuste de un panel solar sencillo.

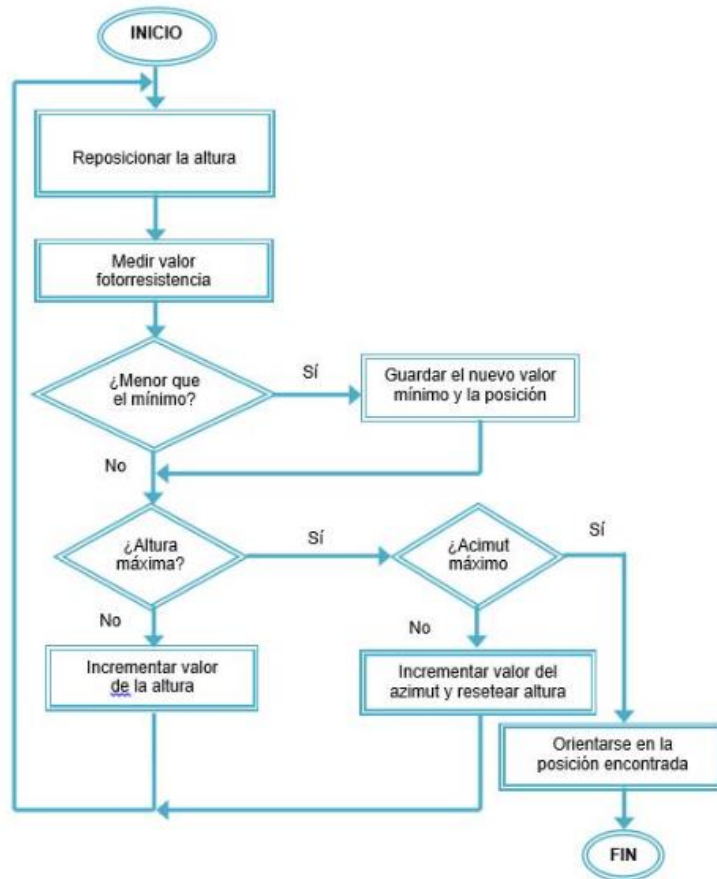


Figura 3-10: Ejemplo de algoritmo de seguimiento solar [9]

Sin embargo, la mayoría están centrados en la obtención de la mejor orientación sin tener en cuenta el consumo que podría llevar a cabo el movimiento de los motores, sobre todo a gran escala.

4 DESCRIPCIÓN DEL EQUIPAMIENTO UTILIZADO

En este apartado, se proporcionará una descripción detallada del equipamiento utilizado en el proyecto de seguimiento solar. Desde los componentes electrónicos básicos hasta los dispositivos de adquisición de datos y control, se examinará el conjunto de herramientas y tecnologías necesarias para el diseño, implementación y operación del sistema. Esta descripción proporcionará una visión general completa del hardware empleado, destacando sus características, funciones y roles dentro del proyecto. En la Figura 4-1 se muestran los bloques del sistema implementado, así como sus conexiones:

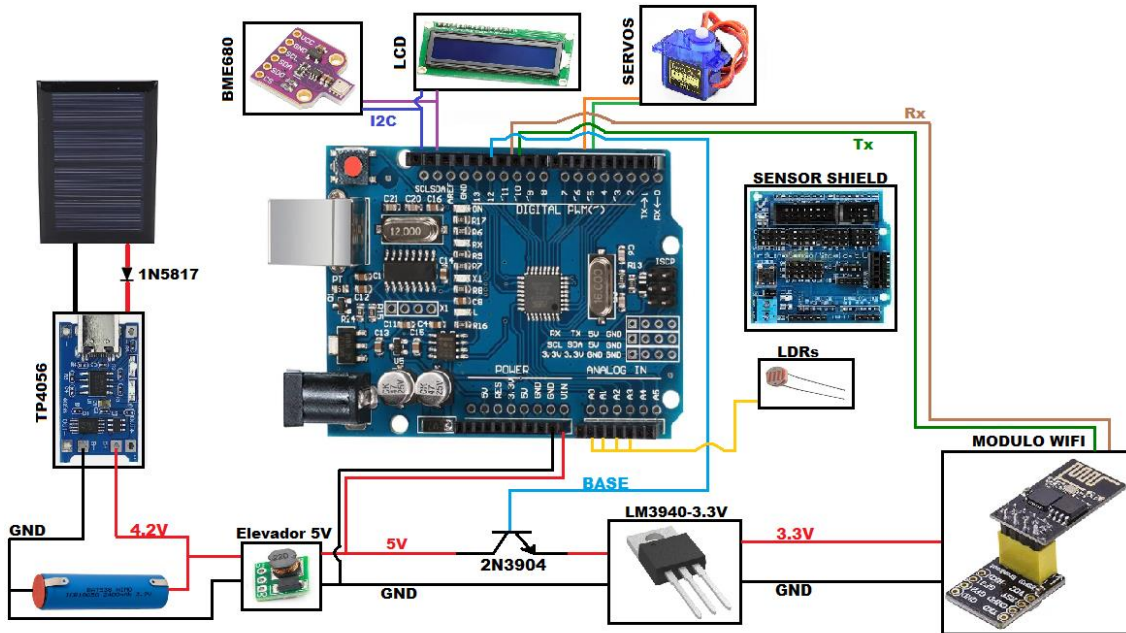


Figura 4-1: Esquema principal de montaje

El sistema está dividido en dos grandes bloques, el encargado de la alimentación y el formado por los sensores y actuadores del proyecto. El sistema de alimentación consta de una Placa fotovoltaica, la cual estará en la posición óptima para captar la radiación solar gracias al software y será la encargada de cargar la batería de ion-litio mediante el regulador de carga TP4056. Se ha colocado un diodo de protección 1N5817 entre el módulo fotovoltaico y el regulador TP4056, para evitar posibles corrientes de retorno que puedan dañar nuestra placa solar.

Las salidas B+ y B- proporcionadas por el TP4056 tendrán la función de cargar la batería cuando esta lo necesite. Además, las salidas Vout+ y Vout- de este mismo modulo aportarán la tensión actual de la batería. Sin embargo, la corriente que son capaces de aportar estas salidas es limitada por lo que será la propia batería la que irá dirigida al elevador de 5V que será el encargado de alimentar el Arduino UNO a partir de los pines Vin y GND.

En este proyecto se utiliza la placa del sensor shield, la cual va montada encima del Arduino UNO, para que las conexiones sean más limpias y fáciles, sobre todo con lo que respecta a los motores y LDRs (Light Dependant Resistors o Resistencia Dependiente de la luz). Debido a esto en vez de alimentar directamente a la placa Arduino desde el pin Vin y GND, se utilizará la

alimentación externa de 5V del sensor shield para alimentarlo tanto a él como a nuestro Arduino, con una configuración que se explicará más adelante.

Una vez está solucionado el tema de la alimentación del Arduino UNO, se pasa a alimentar el módulo WiFi el cual requiere una tensión de 3.3V. Para conseguir esto, se ha utilizado el regulador de voltaje LM3940-3.3V para proporcionar el voltaje requerido. Además, para controlar el encendido y apagado del módulo WiFi desde el Arduino UNO es necesario la instalación de un interruptor y se ha decidido optar por el transistor bipolar 2N3904 cuya base será controlada desde el pin digital 12.

Refiriéndose al bloque de los sensores y actuadores, el sensor BME680 será el encargado de realizar medidas de temperatura y humedad y está configurado por I2C (Inter-Integrated Circuit o circuito inter-integrado) conectado directamente en la placa sensor shield. Además, los módulos de las LDRs irán conectados a los pines analógicos de la placa sensor shield, mientras que los servomotores estarán conectados a cualquier pin que admita una señal PWM (Pulse Width Modulation o Modulación de ancho de pulso), en este caso serán los pines 5 y 6. Por último el módulo LCD 1602 que se comunicará también mediante I2C, será el encargado de mostrar los valores medidos por el BME680 así como el día y la hora actual para que el usuario pueda ver de un solo vistazo la información deseada.

4.1 Hardware

4.1.1 Placa fotovoltaica

El mini panel solar seleccionado para el proyecto debía cumplir las siguientes características, tener la suficiente tensión y corriente para ser capaz de alimentar correctamente el módulo TP4056 así como unas dimensiones pequeñas para su instalación.

En este caso hemos optado por un panel de 6V de tensión nominal, 200 mA de corriente en sus casos ideales y 1.2W de potencia, son unos valores más que suficientes para el proyecto. Además de esto, las dimensiones del panel (112x84 [mm]) son lo suficientemente pequeñas para su instalación en la estructura de madera mostrada en Figura 4-2 la cuál será la encargada de posicionar a nuestro módulo en la posición más optima de radiación solar gracias a los motores.

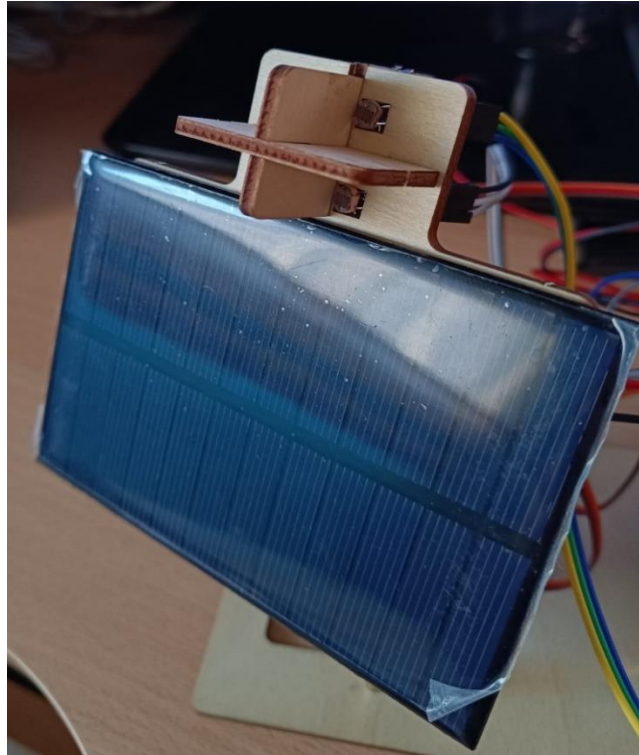


Figura 4-2: Panel fotovoltaico instalado

Realizando pruebas y medidas posicionando al panel solar en su orientación óptima, se ha observado que la tensión proporcionada por el panel fotovoltaico podía llegar hasta los 7V y la corriente emitida hasta los 250mA, resultando así en un panel con una potencia de 1.75W. Aún con este aumento el sistema es capaz de funcionar sin problema.

4.1.2 Diodo protección 1N5817

Este diodo Schottky es el encargado de proteger a la placa fotovoltaica de posibles corrientes inversas que puedan darse. Dichas corrientes pueden ser capaces de empeorar el funcionamiento del sistema además de su eficiencia y en casos extremos de incluso dañar al sistema fotovoltaico.

Se ha elegido este tipo de diodo por las siguientes razones y características extraídas de la hoja de datos aportada por el fabricante [10]:

1. Baja caída de voltaje directo V_F (Voltage Forward o Voltaje Directo):

Esta pérdida de tensión se da cuando la corriente está fluyendo desde la placa solar al TP4056. Poniéndose en el peor de los casos y teniendo en cuenta que la corriente como máximo será de 1A, podremos ver que la V_F es de 0.58V aproximadamente como se muestra en la Figura 4-3.

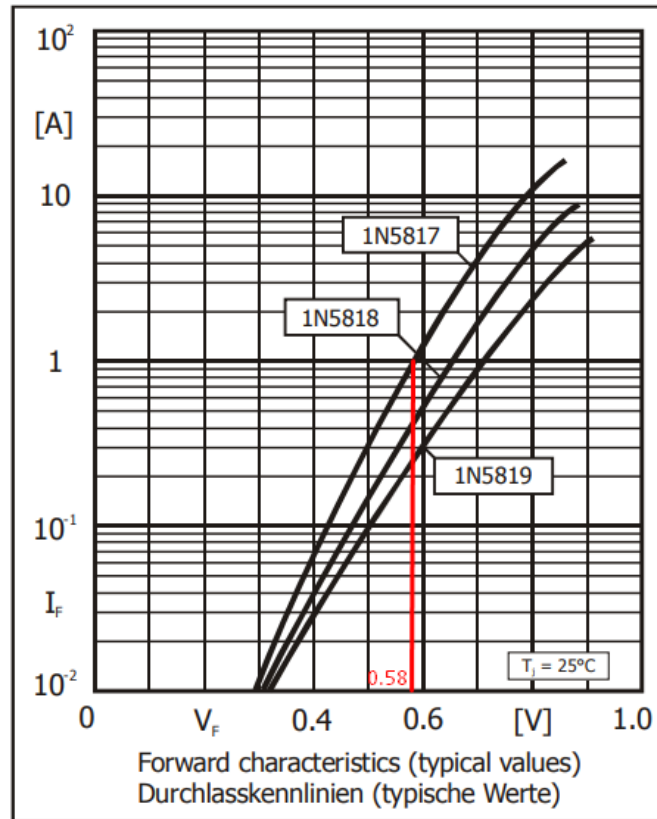


Figura 4-3: V_F del diodo 1N5817 [10]

Teniendo en cuenta que en condiciones óptimas el módulo fotovoltaico proporcionará en torno a los 7V, esta caída de tensión provocada por el diodo no supone ningún problema para el TP4056 ya que admite este rango de alimentación sin problema.

2. Manejo de alta corriente I_{FAV} (Forward Current Average o Corriente directa promedio):

Este parámetro indica la corriente máxima que puede fluir a través del diodo cuando está polarizado en directa, por tanto esta característica limitará la corriente que irá de la placa solar al TP4056. Según la información mostrada en la Tabla 2 este valor será de 1A, este valor es más que aceptable ya que la corriente máxima que proporcionará el módulo fotovoltaico será de 250mA como se ha comentado en el apartado de Placa fotovoltaica.

Max. average forward rectified current, R-load Dauergrenzstrom in Einwegschtaltung mit R-Last	$T_A = 75^\circ\text{C}$	I_{FAV}	1 A)
Repetitive peak forward current Periodischer Spitzenstrom	$f > 15 \text{ Hz}$	I_{FRM}	10 A ²⁾

Tabla 2: Valores de corriente soportados por 1N5817 [10]

3. Tensión de reversa V_{RRM} (Reverse Repetitive Voltage o Voltaje Repetitivo Inverso Máximo)

Este parámetro se refiere a la tensión inversa que es capaz de soportar el diodo, en este caso no es un parámetro realmente importante ya que es un valor que no va a ser superado en

ninguna circunstancia ya que no se utilizan tensiones altas para el proyecto y este límite se sitúa en torno a los 20V como se puede ver en la Tabla 3.

Maximum ratings and Characteristics

Type	Repetitive peak reverse voltage V_{RRM} [V]	Surge peak reverse voltage V_{RSM} [V]
1N5817	20	20
1N5818	30	30

Tabla 3: Tensión de reversa máxima soportada por el diodo 1N5817 [10]

En resumen, este diodo Schottky cumple todas las especificaciones necesarias para su uso en el proyecto, además es muy económico y fácil de encontrar por lo que es la elección ideal.

4.1.3 Módulo carga TP4056

El módulo TP4056 es un elemento clave para el sistema ya que será el encargado de cargar la batería de ion-litio de una manera inteligente y segura a partir de la corriente proporcionada por la placa fotovoltaica. Dispone de una entrada a la que irá conectado el módulo fotovoltaico y de dos salidas, B+/B- a la que estará conectada la batería para su carga, también dispone de Vout+/Vout- las cuales están diseñadas con una limitación de corriente integrada. Esta característica es esencial para proteger baterías de menor capacidad contra posibles daños causados por una extracción de corriente que exceda sus especificaciones de seguridad. En este caso disponemos de una batería con la suficiente capacidad para saltarnos esta protección por lo que estas salidas no se utilizarán para este proyecto.

Por tanto, será la propia batería la encargada de alimentar el proyecto tal y como se muestra en la Figura 4-4:



Figura 4-4: Montaje y conexión del módulo TP4056

Este módulo tiene las siguientes funcionalidades y características por las cuales ha sido elegido:

1. Alimentación de entrada:

Debido a que la entrada del regulador de carga de baterías será en este caso una placa fotovoltaica la cuál puede variar su tensión y corriente frecuentemente, es necesario que la

alimentación de entrada del TP4056 sea lo suficientemente flexible en este sentido. Esto se cumple si se observa la hoja de especificaciones de este [11], concretamente lo mostrado en la Figura 4-5:

ELECTRICAL CHARACTERISTICS

The ● denotes specifications which apply over the full operating temperature range, otherwise specifications are at $T_A=25^{\circ}\text{C}$, $V_{CC}=5\text{V}$, unless otherwise noted.

SYMBOL	PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
V_{CC}	Input Supply Voltage	●	4.0	5	8.0	V

Figura 4-5: Alimentación del módulo TP4056 [11]

La tensión de entrada que es capaz de soportar está en el rango de entre 4V y 8V. Teniendo en cuenta que la placa fotovoltaica aportará como máximo en torno a los 7V y que el diodo de protección tiene una pérdida máxima aproximada de 0.58V, se tiene un margen de alimentación que hará que el TP4056 funcione perfectamente.

2. Corriente de entrada y salida:

Tanto la corriente de entrada como la de salida estarán limitadas por la resistencia programable R_{PROG} . Por defecto la resistencia valdrá 1.2k Ω , lo que configurará al módulo para aportar 1A como máximo a la batería, esto se puede ver en la Figura 4-6 y Figura 4-7 que ha sido extraída de la hoja de datos [11]. En este caso, y como se ha comentado antes, bajo ningún concepto se hará circular por el TP4056 más de 250mA que es lo aportado por la placa fotovoltaica, por lo que la batería no será cargada a más corriente que esta.

I_{BAT}	BAT Pin Current	$R_{\text{PROG}} = 2.4\text{k}$, Current Mode	●	450	500	550	mA
	Text condition: $V_{\text{BAT}}=4.0\text{V}$	$R_{\text{PROG}} = 1.2\text{k}$, Current Mode	●	950	1000	1050	mA
		Standby Mode, $V_{\text{BAT}} = 4.2\text{V}$	●	0	-2.5	-6	μA

Figura 4-6: Corriente de carga en función de resistencia programable [11]

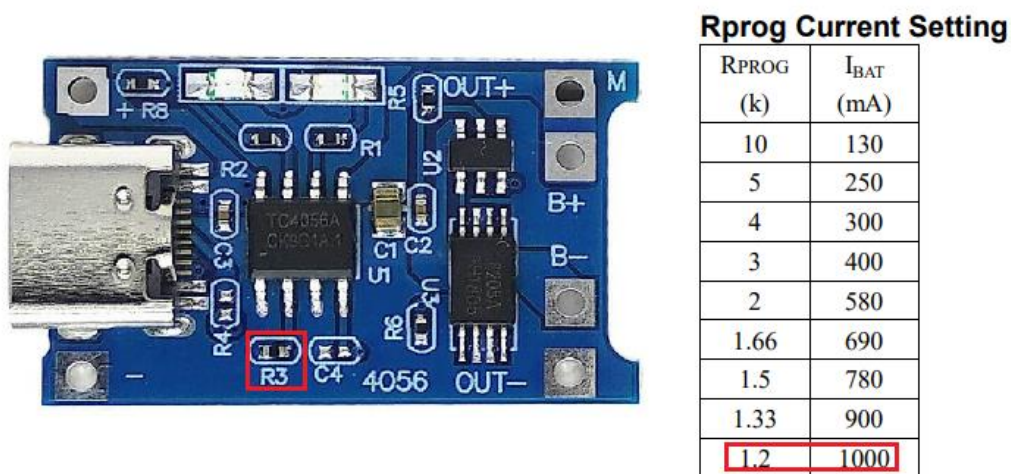


Figura 4-7: Resistencia programable y efecto en la corriente de carga [12]

3. Control de carga de batería:

Este es el motivo principal por el cual se ha seleccionado este módulo, ya que al cargar una batería de ion-litio, es fundamental protegerla contra posibles sobrecargas y sobre descargas que podrían dañarla, esto permitirá un aumento en la seguridad, durabilidad y rendimiento de esta.

Es importante destacar que, para que este módulo funcione correctamente, la batería que se esté cargando debe tener una tensión nominal de 3.7V, ya que estas baterías pueden cargarse de forma segura hasta 4.2V. El módulo detectará cuando la tensión de la batería llegue al valor máximo de 4.2V y cortará el paso de corriente por los pines B+ y B- protegiendo así la batería.

4. Control del estado de carga:

El TP4056 tiene incorporados dos leds los cuales indicarán el estado de carga de la batería, el led rojo estará encendido mientras se esté cargando y el led azul cuando la carga esté completa, esta información la facilita el fabricante en su hoja de datos [11], concretamente en la Tabla 4.

indicator light state

Charge state	Red LED $\overline{\text{CHRG}}$	Green LED $\overline{\text{STDBY}}$
charging	bright	extinguish
Charge Termination	extinguish	bright

Tabla 4: Indicador de carga mediante Leds [11]

5. Método de pruebas:

El módulo TP4056 se puede alimentar mediante un micro USB (Universal Serial Bus), el cuál aportará 5V y 200mA de corriente, de esta manera se podrán realizar pruebas de una manera sencilla y comprobar que la batería se carga y descarga correctamente. Además, se podrá ver si realmente se está protegiendo contra sobrecargas y sobre descargas eficientemente fijándonos en los leds comentados anteriormente.

Debido a todas las características comentadas anteriormente, su gran disponibilidad y el costo asequible que tiene es la mejor elección para el proyecto y su correcto desempeño.

4.1.4 Consumo del sistema

Antes de proceder a la elección de una batería es interesante realizar un estudio sobre el consumo del sistema, para así saber cuál sería la autonomía del proyecto y si sería conveniente el uso de otra o más baterías.

En este caso, se va a realizar el estudio de consumo dividido en dos partes, cuando se está ejecutando el programa en modo Test y por tanto, no se arranca la parte del módulo WiFi (Wireless Fidelity o Fidelidad Inalámbrica) y otra en el comportamiento normal del sistema, con el correspondiente módulo ESP8266-01 así como la entrada en bajo consumo.

1. Modo Test:

La particularidad de este modo es que se centra en el seguimiento de luz, dejando a un lado la conexión WiFi, el envío de datos a ThingSpeak y la consulta al servidor NTP (Network Time Protocol o Protocolo de Tiempo de Red). Por lo que es un modo en el que el consumo será menor por la omisión del ESP8266-01. Sin embargo, el consumo total será variable en función de cuánto tiempo queramos realizar pruebas.

Cuando se realizan movimientos de ambos servomotores el consumo puede llegar a ser de hasta 250mA en el peor de los casos, dependiendo de la posición y resistencia encontrada para mover estos. En cambio, cuando no se están realizando movimientos de los servomotores y únicamente se están realizando medidas de temperatura/humedad así como su representación en el LCD se llega a consumir 61mA.

Se supone un caso en el que se moverán los motores un total de 10 minutos a la hora, y el resto del tiempo estará realizando las medidas pertinentes, realizando unos sencillos cálculos con la Ecuación 4-1 se obtendrá el consumo por hora:

$$\text{Consumo de energía (mAh)} = I(\text{mA}) \times t(\text{seg})/3600 \quad \text{Ecuación 4-1}$$

$$61\text{mA} * \frac{3000 \text{ segundos}}{3600 \frac{\text{segundos}}{\text{hora}}} = 50.83 \text{ mAh} \quad 250\text{mA} * \frac{600 \text{ segundos}}{3600 \frac{\text{segundos}}{\text{hora}}} = 41.6 \text{ mAh}$$

Dándonos como resultado un consumo de **92.49mAh**.

2. Modo normal:

En este modo, aparte de orientar al módulo fotovoltaico en su posición óptima y realizar medidas de temperatura/humedad con su respectiva representación, también se activa el módulo ESP8266-01 lo que hace que se dispare el consumo. Sin embargo, este modo realiza medidas cada 5 minutos, cuenta con un modo bajo consumo y solo realiza movimiento en ambos ejes una vez al día.

Tras la realización de medidas utilizando el multímetro se observan los siguientes consumos:

- El movimiento de ambos motores es de 250mA.
- Un solo servomotor consume 120mA.
- Realización de medidas de temperatura/humedad y su representación consume 61mA.
- En modo bajo consumo baja hasta los 42mA.
- El módulo WiFi consume cuando está arrancado alrededor de los 70mA.

Para realizar el cálculo del consumo por hora se necesita saber durante cuánto tiempo se dan estos consumos, para esto se ha elegido la siguiente estimación:

- Los servomotores se moverán durante 2 segundos, aunque en la mayoría de los casos será mucho menos tiempo.
- El movimiento de ambos ejes se realizará una sola vez, es prácticamente despreciable (2 segundos).
- El movimiento de un solo eje se realizará 11 veces en la primera hora y 12 veces el resto del tiempo (11*2=22 segundos).
- El tiempo que se tarda en realizar las medidas es ínfimo y hemos supuesto 1 segundo cada 5 minutos (12 segundos).
- El módulo ESP8266-01 estará arrancado entre 10 y 15 segundos, dependiendo de la rapidez de la conexión, suponiendo el peor caso se toman 15 segundos cada 5 minutos (12*15=180 segundos).
- El resto del tiempo el sistema se encontrará en modo bajo consumo.

Teniendo todo esto en cuenta, se muestran los cálculos realizados para obtener el consumo real utilizando la Ecuación 4-2:

$$\begin{aligned}
 250mA * \frac{2 \text{ segundos}}{3600 \frac{\text{segundos}}{\text{hora}}} &= \mathbf{0.14mAh} & 120mA * \frac{22 \text{ segundos}}{3600 \frac{\text{segundos}}{\text{hora}}} &= \mathbf{0.73mAh} \\
 61mA * \frac{12 \text{ segundos}}{3600 \frac{\text{segundos}}{\text{hora}}} &= \mathbf{0.2mAh} & 70mA * \frac{180 \text{ segundos}}{3600 \frac{\text{segundos}}{\text{hora}}} &= \mathbf{3.5mAh} \\
 42mA * \frac{3384 \text{ segundos}}{3600 \frac{\text{segundos}}{\text{hora}}} &= \mathbf{39.48mAh}
 \end{aligned}$$

Si se realiza la suma de todos los consumos calculados, se obtiene un consumo total de **44.06mAh** al que se añadirá un 20% de margen de seguridad ante posibles incrementos como el del módulo WiFi, resultando en un consumo de **52.87mAh**. Toda esta información se presenta de una manera más intuitiva en la Tabla 5.

Variables	Ambos servos	Un servo	Medidas	Bajo consumo	ESP8266-01
Consumo (mA)	250	120	61	42	70
Tiempo activo (seg)	2	22	12	3384	180
Consumo (mAh)	0,14	0,73	0,20	39,48	3,50
Consumo Total (mAh)	44,06				
Consumo Margen (mAh)	52,87				

Tabla 5: Consumo en modo normal

Por tanto, se puede concluir que el consumo de corriente será mayor en el modo Test debido al uso de los servomotores que son los principales consumidores de este sistema.

4.1.5 Batería recargable

La batería es una parte fundamental del sistema, ya que será la encargada de alimentar al Arduino UNO y por tanto a los motores, sensores y demás módulos del proyecto. La elección de esta viene determinada por tres parámetros:

- La tensión nominal, ya que el módulo TP4056 obliga a que sea de 3.7V nominales.
- La capacidad, entre mayor capacidad mayor será la autonomía del sistema, por lo que se ha maximizado en la medida de lo posible teniendo en cuenta lo visto en el apartado Consumo del sistema.
- Precio, hay que tener en cuenta que entre mayor sea la capacidad mayor será el precio, por lo que se ha intentado buscar una relación calidad-precio aceptable.

Teniendo en cuenta los puntos anteriores se ha elegido una batería de ion-litio del tipo 18650, las cuales son las que tienen mayor capacidad, en este caso se ha optado por la batería SDNMY de 4800mAh mostrado en la Figura 4-8 que tiene una capacidad bastante alta para una sola batería.



Figura 4-8: Batería ion-litio 18650 de 4800mAh [13]

Como se sabe que el consumo en el modo normal de ejecución es de 52.87mAh y que la batería utilizada tiene 4800mAh, se puede calcular la autonomía de la misma utilizando la Ecuación 4-2

$$\text{Autonomía (horas)} = \frac{\text{CapacidadBateria (mAh)}}{\text{ConsumoSistema (mA)}} \quad \text{Ecuación 4-2}$$

$$A = \frac{4800\text{mAh}}{52.87\text{mA}} = 90.79\text{h} \rightarrow \mathbf{3.78 \text{ días}}$$

El sistema será capaz de funcionar durante 90.79 horas sin ningún tipo de recarga por parte de la placa solar lo que da al proyecto una autonomía suficiente para su uso.

Un detalle clave a tener en cuenta es que la batería se carga con 250mAh gracias a la placa fotovoltaica, siempre y cuando esté incidiendo radiación solar, no obstante, se consume 52.87mAh, lo que significa que se recarga aproximadamente a 200mA. Con estas condiciones, se espera que la batería se cargue por completo en aproximadamente 24 horas con la radiación solar adecuada. Suponiendo una duración de 8 horas de sol diarias, se lograría una recarga completa en 3 días.

$$Recarga = \frac{4800mAh}{200mA} = 24 \text{ horas}$$

4.1.6 Regulador elevador 5V

Para este proyecto se quiere alimentar al Arduino UNO a través de la placa sensor shield de una manera segura y eficiente, para esto es necesario un regulador el cual aporte una tensión constante en torno a los 5V y una corriente suficiente para el sistema, en este caso se usa el regulador Youmile DC-DC mostrado en la Figura 4-9.

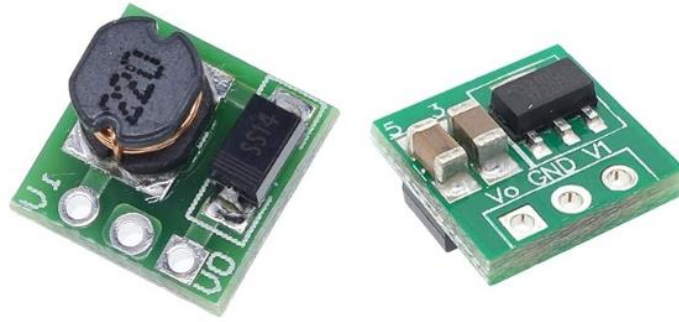


Figura 4-9: Regulador de voltaje elevador [14]

Como se ha comentado, se utiliza una batería de ion-litio de valor nominal 3.7V las cuales pueden variar de 3.2V a 4.2V, la salida de esta batería será la que se conecte al regulador de 5V, el cual según lo mostrado en la Tabla 6 aportará un margen de corriente de entre 380-480mA.

Voltaje entrada	Corriente salida (mA)	Voltaje salida(V)
1-1,5 [V]	40-100	5
1,5-2 [V]	100-150	5
2-3 [V]	150-380	5
>3 [V]	380-480	5

Tabla 6: Corriente y tensión de salida en función de la entrada [15]

En este caso, la tensión de entrada al regulador dependerá de la tensión que aporte la batería:

- En el caso de que la batería esté completamente cargada, aportará en torno a los 4.2V y será capaz de extraer sin problema aproximadamente 400mA de corriente.
- Si la batería se descargase por una serie de días en los que no ha habido radiación solar, esta seguiría aportando 3.2V, por lo que seremos capaces de extraer al menos 380mA en cualquiera de los casos.

Hay que tener en cuenta también la eficiencia de dicho módulo que según el comerciante [15] es de en torno al 85%, de hecho, realizando pruebas se ha observado que la tensión obtenida a la salida es de 5.12V, esto no supone ningún problema ya que los pines de alimentación de la placa sensor shield admiten hasta 12V.

En el mercado hay una gran cantidad de reguladores de este tipo, pero se ha decidido utilizar este ya que tiene un costo muy reducido, un hardware muy pequeño, así como compacto para simplificar el montaje del sistema y su facilidad de uso.

4.1.7 Transistor bipolar 2N3904

Para controlar la activación del módulo WiFi y por tanto su consumo de corriente, se ha optado por diseñar un interruptor que se pudiera controlar directamente desde el Arduino UNO con la configuración mostrada en la Figura 4-10.

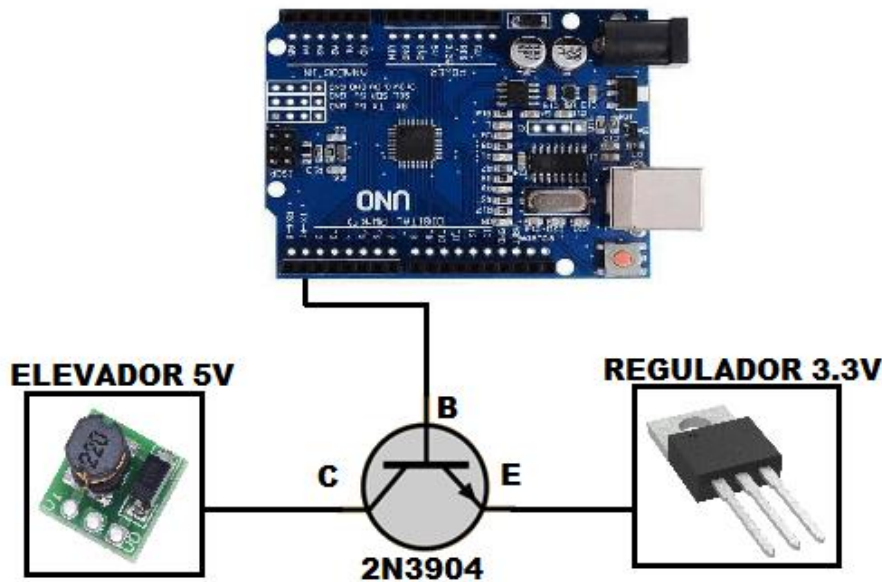


Figura 4-10: Conexión del transistor 2N3904

Se ha decidido utilizar el transistor bipolar 2N3904 que tiene las siguientes características extraídas de la hoja de datos del fabricante [16]:

1. Corriente de colector adecuada:

El módulo WiFi ESP8266-01 consumirá un máximo de 200mA en su arranque aunque una vez realizado el estudio del Consumo del sistema se ha observado que está en torno a los 70mA, por lo que es necesario que el transistor sea capaz de conducir al menos esta corriente como se muestra en la Tabla 7:

Collector Current – Continuous	I_C	200	mAdc
--------------------------------	-------	-----	------

Tabla 7: Corriente de colector del 2N3904 [16]

Aunque sería recomendable tener un poco más de margen en este parámetro por si se supera dicho valor en algún caso, en ningún momento ha habido problemas de corriente, incluso se ha podido alimentar con los 50mA del Arduino UNO por lo que la corriente de colector del transistor es más que suficiente.

2. Caída de tensión colector emisor

La tensión de colector-emisor es relativamente baja, esto quiere decir que cuando el transistor esté encendido, la caída de voltaje será mínima como se muestra en la Tabla 8:

Collector-Emitter Saturation Voltage (Note 2.) ($I_C = 10 \text{ mAdc}$, $I_B = 1.0 \text{ mAdc}$) ($I_C = 50 \text{ mAdc}$, $I_B = 5.0 \text{ mAdc}$)	$V_{CE(sat)}$	-	0.2 0.3	V_{dc}
--	---------------	---	------------	----------

Tabla 8: Caída de tensión colector emisor [16]

La pérdida de tensión debido a la conducción es prácticamente insignificante en torno a 0.2-0.3V. Se ha elegido este transistor ya que es muy sencillo tanto de utilizar como de obtener, además de su bajo coste, teniendo en cuenta que cumple todas las expectativas. Este transistor viene en un encapsulado TO-92 que el fabricante en su hoja de datos [16] indica como conectar y su asociación de pines:

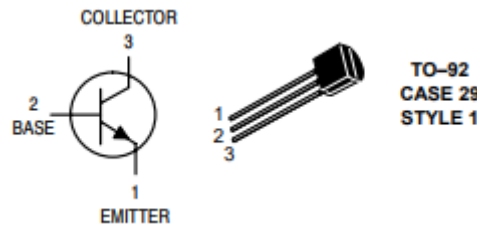


Figura 4-11: Encapsulado indicado por el fabricante [16]

4.1.8 Regulador 3.3V (LM3940)

Debido a que el módulo WiFi necesita de una alimentación de 3.3V y no debe sobrepasar nunca los 3.6V es necesario utilizar un regulador lo suficientemente fiable para que estas condiciones se cumplan. La elección para este proyecto ha sido usar el LM3940 el cuál es un regulador de bajo dropout que proporciona una conversión muy estable y con una corriente suficiente. Las características principales son las siguientes que han sido extraídas de la hoja de datos facilitada por el fabricante [17]:

1. Corriente soportada:

Este regulador es capaz de soportar corrientes de hasta 1A, teniendo en cuenta que la entrada de este regulador viene del Transistor bipolar 2N3904 anterior que como máximo aportará 200mA, la corriente de entrada del LM3940 es más que suficiente para nuestras especificaciones.

2. Bajo dropout:

El dropout es un término importante a tener en cuenta al trabajar con reguladores, ya que es la mínima diferencia de voltaje entre la entrada y la salida de dicho regulador. La entrada del LM3940 será siempre en torno a los 5V, por lo que el dropout en este caso será de $5-3.3=1.7V$ más que adecuada para el regulador, el cuál admite hasta 0.15V cuando se consuman los 200mA que es el valor máximo que se consumirá en este módulo según lo mostrado en la Figura 4-12:

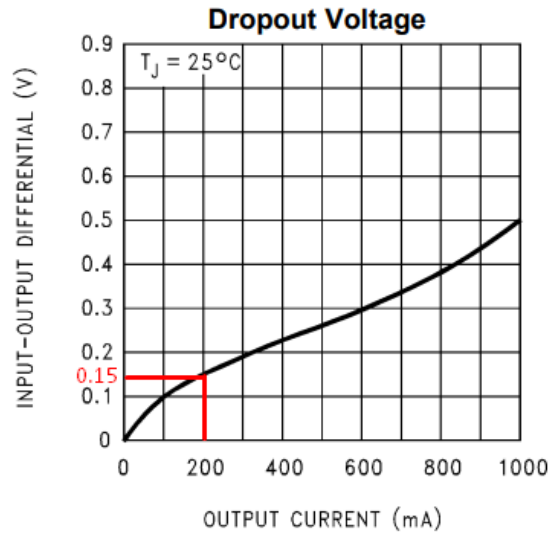


Figura 4-12: Voltaje de dropout para 200mA

3. Tensión de salida estable:

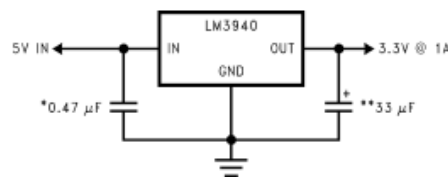
La tensión de salida será de 3.3V, siempre y cuando se cumpla el dropout, una corriente de entrada aceptable así como un montaje correcto del mismo, esto se muestra en la Tabla 9:

Symbol	Parameter	Conditions	Typical	LM3940 ⁽¹⁾		Units
				min	max	
V_O	Output Voltage	$5\text{ mA} \leq I_L \leq 1\text{ A}$	3.3	3.20 3.13	3.40 3.47	V

Tabla 9: Tensión a la salida del regulador [17]

4. Montaje:

A diferencia del regulador anterior, el LM3940 requiere de una configuración específica para que funcione correctamente, el fabricante indica dicha configuración en la hoja de datos [17] más concretamente en la Figura 4-13:



*Required if regulator is located more than 1" from the power supply filter capacitor or if battery power is used.

Figura 4-13: Configuración y conexión del LM3940 [17]

Los condensadores marcados en el esquema son utilizados para mantener una estabilidad del voltaje a su salida así como la reducción de su rizado.

Con respecto al primero de los condensadores, se indica que es necesario instalarlo si nuestro sistema está alimentado por una batería, ya que así ayudará a suavizar cualquier fluctuación. Sin embargo, no se comenta ninguna limitación sobre el valor de este, por lo que se ha optado por poner un valor de 33 μF para tener un mayor margen de seguridad en caso de variaciones.

Sobre el segundo condensador el fabricante da más datos e indica que como mínimo debe ser de 33uF para que funcione como se espera.

El encapsulado utilizado por el fabricante en este caso es el TO-220, y en la Figura 4-14 se muestra a qué corresponden cada uno de sus pines:

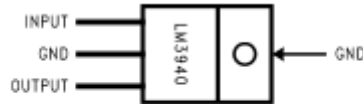


Figure 1. 3-Lead TO-220 Package (Front View)

Figura 4-14: Situación de pines del encapsulado [17]

5. Necesidad de disipador térmico:

Para averiguar si es necesario un disipador térmico primero se deben realizar los cálculos mostrados en la hoja de datos facilitada por el fabricante con la Ecuación 4-3:

$$\Theta_{(H-A)} = \Theta_{(JA)} - \Theta_{(C-H)} - \Theta_{(J-C)}$$

Ecuación 4-3: Disipador ambiente

- $\Theta_{(J-C)}$ es la resistencia térmica desde la unión hasta la superficie de la carcasa del regulador, como el encapsulado es el TO-220 el fabricante supone 4°C/W.
- $\Theta_{(C-H)}$ es la resistencia térmica entre la carcasa y la superficie del disipador de calor, como se desconoce el valor el fabricante supone un valor de 2°C/W.
- $\Theta_{(JA)}$ es la resistencia térmica unión ambiente y que también proporciona el fabricante con un valor de 32°C/W.

Realizando los cálculos pertinentes se obtiene que la resistencia térmica disipador ambiente ($\Theta_{(H-A)}$) es de 26°C/W. Una vez se tiene este valor y teniendo en cuenta la T_j , que es la temperatura de unión, es de 125°C se calculará si realmente es necesario un disipador térmico utilizando la Ecuación 4-4

$$T_j = P_D \times (\Theta_{(J-C)} + \Theta_{(C-H)} + \Theta_{(H-A)}) + T_a$$

Ecuación 4-4: Temperatura de unión resultante

Se sabe que por el regulador pasarán como máximo 200mA y 5V a su entrada lo que da una P_D de 1W y una temperatura ambiente (T_a) de 25°C, con estos valores el cálculo queda:

$$T_j = 1W * (4 + (2 + 26)) + 25 = 57^\circ C$$

Como esta T_j calculada es bastante menor que la T_j máxima de 125°C, no será necesario ningún tipo de regulador en este sistema.

Por tanto, se ha decidido optar por utilizar el regulador LM3940 ya que es fácil de encontrar, cumple todas las características que se desean, sobre todo la del dropout y por su facilidad de montaje.

4.1.9 Módulo WiFi ESP8266-01

El módulo ESP8266-01 es un dispositivo compacto que integra una unidad de comunicación WiFi, diseñado por Espressif Systems el cual facilita la hoja de datos [18]. Este módulo es el

encargado de establecer una conexión con el servidor NTP y también con la web ThingSpeak para la transmisión de datos. Este dispositivo está compuesto por 8 pines que se deberán de conectar de manera distinta si queremos cargar un programa o si queremos correrlo.

A continuación se explican los diferentes pines y sus funcionalidades:

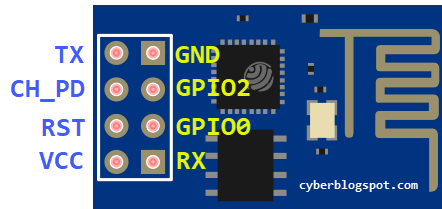


Figura 4-15: Pines del módulo ESP8266-01 [19]

1. Tx es el pin encargado de la transmisión de datos del puerto serie.
2. GND es la tierra del módulo e irá cortocircuitada con la masa del Arduino.
3. CH_PD será la habilitación del módulo, a nivel alto se habilitará y a nivel bajo estará deshabilitado.
4. GPIO0 y GPIO2, son pines entrada-salida.
5. RST, pin encargado de resetear el módulo, activo a nivel bajo.
6. VCC será la alimentación del ESP8266-01.
7. Rx es el pin encargado de recibir los datos del puerto serie.

Consideraciones a tener en cuenta sobre la alimentación:

El módulo funciona con 3,3V y un máximo de 3,6V por lo que es muy importante que la conexión esté preparada para esta alimentación, por eso se ha usado el regulador LM3940.

Este sistema puede llegar a consumir hasta 200mA por lo que es imprescindible que al menos le llegue esta cantidad de corriente, esta limitación obliga a alimentar el módulo con la batería y ciertos reguladores ya que el Arduino UNO en su pin de 3,3V solo puede proporcionar 50mA, esta es una corriente baja y que puede funcionar en la mayoría de los casos, pero para una mayor seguridad se ha optado por la alimentación externa.

Adaptador para la conexión:

El módulo WiFi tiene un formato compacto y solo cuenta con un conector de 2x4 pines para la conexión, por tanto, para que el montaje sea más limpio se ha decidido utilizar un adaptador aportado por AZDelivery [20] , así la conexión a la protoboard será más fácil y sencilla tal y como se muestra en la Figura 4-16.

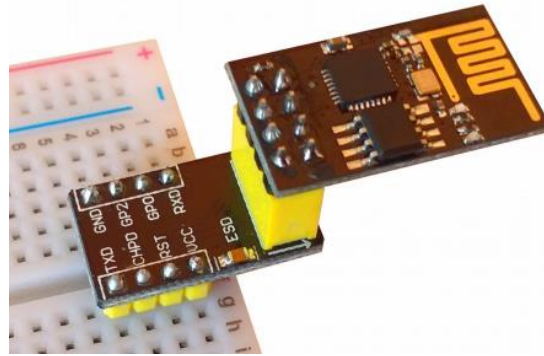


Figura 4-16: Conexión del ESP8266-01 con el adaptador [20]

Como se ha comentado antes, la conexión del ESP8266-01 será distinta cuando se quiera cargar el software a cuando se desee correr el código cargado. En un principio el adaptador solo se utilizará para lanzar el código y la implementación final mientras que para la carga del software se utilizará la siguiente conexión:

Conexión para la carga del software:

1. El pin de VCC deberá estar conectado a los 3.3V que da el Arduino UNO, en este caso no hay problema con la limitación de corriente ya que solo se cargará software.
2. El GND deberá ir al pin de tierra del Arduino UNO para establecer una referencia común.
3. El pin Tx del módulo ESP8266-01 deberá ir al pin Rx del Arduino UNO y el pin Rx del módulo WiFi irá al pin Tx del Arduino UNO.

Hay que tener en cuenta que estos dos módulos trabajan con niveles lógicos distintos, por tanto, es necesario un divisor de tensión entre el Tx del Arduino UNO que funciona a 5V y el Rx del ESP8266-01 que funciona a 3.3V, de esta manera se protege al módulo WiFi frente a posibles picos de tensión superiores a lo indicado:

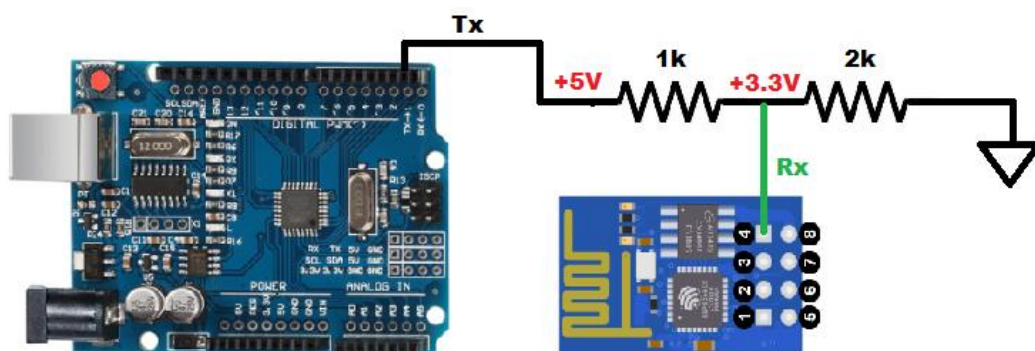


Figura 4-17: Divisor de tensión para la conexión de Tx-Rx

4. El pin de habilitación EN o CH_PD tendrá que ir a los 3.3V para que el módulo permita su configuración.
5. El GPIO2 deberá estar conectado también a 3.3V, pero como tiene internamente un pull-up se tiene la opción de dejarlo sin conectar.

6. El pin de RST estará conectado a 3.3V ya que no se desea que se reinicie durante este proceso, en caso de que se quede bloqueado bastaría con conectarlo a GND para realizar un reset del sistema.
7. Cortocircuitar el GND y el pin de RST del Arduino UNO, esto deshabilitará la comunicación serial interna del Arduino y evitará conflictos de comunicación con el ESP8266-01.
8. El pin GPIO0 deberá estar conectado a GND cuando demos corriente al sistema, activando así el modo programación y permitiendo la carga de programas a través del Arduino UNO.

Conexión para ejecutar el programa cargado:

Una vez se ha realizado la carga del programa se retirará la alimentación del sistema y habrá que realizar la siguiente conexión, en este caso sí que utilizaremos el adaptador [20]:

1. Los pines VCC, RST y CH_PD están conectados a 3.3V, tanto para alimentar, evitar un reset y habilitar el módulo respectivamente.
2. El GND deberá ir al pin de tierra del Arduino UNO para establecer una referencia común.
3. El GPIO2 deberá estar conectado también a 3.3V, pero como tiene internamente un pull-up se da la opción de dejarlo sin conectar.
4. El GPIO0 estará conectado también a 3.3V, al igual que el GPIO2 tiene un pull-up interno que da la opción de conectarlo.
5. El pin Tx irá conectado directamente al Arduino UNO en el pin 10 y estará configurado mediante software para recibir la información del módulo WiFi.
6. El pin Rx una vez más contará con el divisor de tensión, para proteger al ESP8266-01 frente a niveles lógicos superiores como se observaba en la Figura 4-17. Esta recepción estará conectada al pin 11 del Arduino que estará configurado mediante software para transmitir la información al módulo WiFi.

Para ejecutar el programa, como se ha comentado anteriormente los 3.3V deben venir del regulador de voltaje LM3940 para proteger al módulo frente a posibles picos de tensión.

4.1.10 Sensor BME680

El sensor BME680 será el encargado de realizar medidas de temperatura y humedad las cuales se mandarán posteriormente a la web de ThingSpeak. Es un dispositivo altamente avanzado el cuál es capaz de realizar medidas de temperatura, humedad, presión atmosférica y calidad del aire. Además, admite la comunicación mediante I2C y SPI (Serial Peripheral Interface o Interfaz Periférica Serial), ya que la placa del sensor shield tiene conexión directa con I2C se utilizará esta.

Existen una gran variedad de sensores de temperatura y humedad en el mercado, como puede ser por ejemplo el AM2320, pero se ha decidido utilizar el BME680 por las siguientes características extraídas de la hoja de datos del fabricante [21]:

Alimentación y consumo:

Como se puede observar en la Tabla 10 para que el módulo funcione correctamente este debe estar alimentado entre 1.71V y 3.6V.

OPERATING CONDITIONS BME680						
Parameter	Symbol	Condition	Min	Typ	Max	Unit
Supply Voltage Internal Domains ¹	V _{DD}	ripple max. 50 mVpp	1.71	1.8	3.6	V
Supply Voltage I/O Domain	V _{DDIO}		1.2	1.6	3.6	V
Sleep current	I _{DDSL}			0.15	1	μA
Standby current (inactive period of normal mode)	I _{DDSB}			0.29	0.8	μA
Current during humidity measurement	I _{DDH}	Max value at 85 °C		340	450	μA
Current during pressure measurement	I _{DDP}	Max value at -40 °C		714	849	μA
Current during temperature measurement	I _{DDT}	Max value at 85 °C		350		μA

Tabla 10: Voltaje y corriente en el sensor BME680 [21]

Debido a esta restricción de alimentación se utilizará el pin de 3.3V de la placa sensor shield. La hoja de datos [21] también especifica la alimentación de los pines entrada/salida que es menor que la anteriormente comentada, sin embargo, en este caso se refiere a la tensión necesaria para alimentar la comunicación con otros dispositivos mediante I2C o SPI.

También se puede ver el consumo de corriente cuando se realiza una medida de humedad y de corriente, en ambos casos es menor a 1mA, por lo que se puede concluir que el consumo de este módulo es ínfimo.

Precisión en las medidas:

Este módulo tiene una precisión muy elevada en las medidas, tanto en humedad como en temperatura que son las que conciernen al proyecto.

Como se puede observar en la Tabla 11 la humedad medirá entre 20-80 % con un posible error en la precisión de $\pm 3\%$ y aporta una resolución de 0.008% por humedad relativa. Mientras que la temperatura, se puede ver en la Tabla 12 que hace medidas de -40°C a 85°C, con un posible error de $\pm 0.5^\circ\text{C}$ y una resolución de temperatura de 0.01°C.

Estos valores se van a mostrar en la web de ThingSpeak la cual tiene una resolución de centésimas, por lo que las resoluciones que aporta este sensor son suficientes.

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Operating Range ¹¹			-40	25	85	°C
			0		100	% r.H.
Full accuracy range			0		65	°C
			10		90	% r.H.
Supply Current	I _{DD,H}	1 Hz forced mode, temperature and humidity measurement		2.1	2.8	μA
Absolute Accuracy	A _H	20–80 % r.H., 25 °C, including hysteresis		±3		% r.H.
Hysteresis ¹²	H _H	10→90→10 % r.H., 25°C		±1.5		% r.H.
Nonlinearity ¹³	NL _H	10→90 % r.H., 25°C		1.7		% r.H.
Response time to complete 63% of step ¹⁴	τ _{0-63%}	N ₂ (dry) → 90 % r.H., 25°C		8		s
Resolution	R _H			0.008		% r.H.

Tabla 11: Precisión de humedad según hoja de datos [21]

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Operating temperature range	T _A	operational	-40	25	85	°C
Supply current	I _{DD,T}	1 Hz forced mode, temperature measurement only		1.0		μA
Absolute accuracy temperature ¹⁷	A _{T,25}	25 °C		±0.5		°C
	A _{T,full}	0–65 °C		±1.0		°C
Output resolution	R _T	API output resolution		0.01		°C
RMS noise	N _T	Lowest oversampling		0.005		°C

Tabla 12: Precisión de temperatura según hoja de datos [21]

El BME680 es un sensor el cuál para realizar medidas correctas debe llevar un tiempo arrancado a diferencia de otros sensores y por tanto no se le debe quitar la alimentación para que los valores medidos sean adecuados.

Pese a esto se ha decidido utilizar este sensor por su gran precisión en las medidas así como su resolución, además permite dar una posible escalabilidad al sistema gracias a que se podrían realizar otras medidas como la presión atmosférica o la calidad del aire.

4.1.11 Servomotores

En este proyecto se utilizan dos servomotores que serán los encargados de mover en ambos ejes la estructura de madera instalada y que soporta el panel fotovoltaico. Se ha decidido optar por servomotores de 9 gramos y 180° del modelo SG90 como el mostrado en la Figura 4-18: Servomotor SG90 Figura 4-18, estos son pequeños y altamente utilizados en proyectos de este calibre y tienen las siguientes características:

Dimensión y peso:

Tienen un tamaño pequeño, (23mm x 12mm x 29mm) y un peso de 9 gramos, lo que hace que sean fácilmente manejables e instalables.

Voltaje de operación:

Necesitan un voltaje pequeño para operar, entre 4V y 7.2V, lo que los hace ideales para trabajar con Arduino el cual trabaja con un nivel lógico de 5V.

Velocidad de giro:

Aunque en este proyecto no es necesario realizar giros rápidos o bruscos, estos servomotores son capaces de realizar giros de 60º en 0.12 segundos, una velocidad incluso superior a lo necesitado.

Peso soportado:

Para este diseño, no es necesario que los componentes soporten un gran peso. Sus características les permiten soportar entre 1.2 kg y 1.6 kg, considerando que alimentados a 4.8V podrán soportar 1.2 kg/cm. Esto resulta más que adecuado, dado que el peso de la placa solar es de 28 g y el de la estructura de madera no supera los 10 g.

Consumo y precisión:

El consumo de un servomotor depende de la carga, la tensión aplicada y la velocidad de giro pero suelen consumir entre 100mA y 500mA. Además tienen una precisión razonable para proyectos de este tipo.



Figura 4-18: Servomotor SG90 [22]

4.1.12 Módulo de LDRs

Los módulos LDRs son utilizados para realizar medidas de luz en los 4 ejes y ajustar el movimiento de los servomotores en consecuencia. Estos módulos suelen tener tres pines que desempeñan funciones específicas como los mostrados en la Figura 4-19.

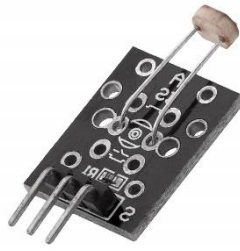


Figura 4-19: Módulo LDR 106[23]

El primer pin, denominado GND, se conecta a la masa del sistema, proporcionando una referencia común para el circuito. El segundo pin, llamado Vcc, suministra una tensión al LDR, lo que permite que funcione correctamente. Por último, el tercer pin, conocido como pin de señal, muestra la tensión en función de la resistencia interna del LDR. Esta resistencia interna varía en función de la cantidad de luz que incide sobre el LDR a menor luz mayor será la resistencia, y la señal de salida del pin refleja esos cambios de resistencia, lo que proporciona una indicación de la intensidad luminosa detectada por el LDR.

Se ha optado por utilizar módulos LDRs para que el montaje sea más sencillo en la estructura de madera, ya que se necesita que la fotorresistencia sobresalga por los agujeros marcados para ello para poder realizar medidas correctamente como se puede ver en la Figura 4-20.

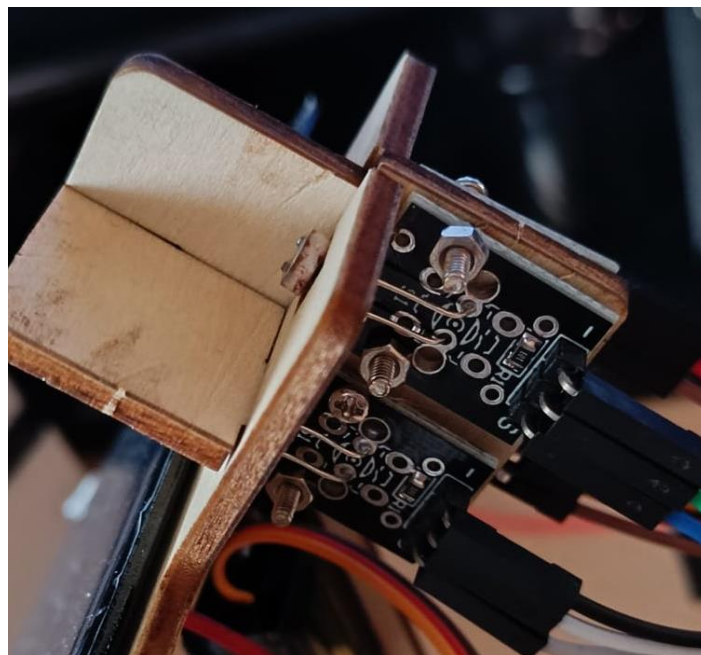


Figura 4-20: Montaje módulos LDRs

4.1.13 LCD 1602

En este proyecto el LCD será el encargado de informar al usuario de la información que necesita, como puede ser la temperatura, humedad, timestamp, así como de en qué estado se encuentra el programa. Se utiliza el LCD 1602 ya que admite configuración mediante comunicación I2C, además de tener las siguientes características extraídas de diferentes foros de Arduino [24]:

1. Alimentación y conexión sencilla:

Este módulo LCD funciona con 5V de alimentación, esto facilita su conexión ya que como se ha comentado, el nivel lógico del sistema es de 5V y se podrá alimentar directamente desde los pines de I2C facilitados por el sensor shield.

2. Manejo de contraste y luz de fondo:

Debido a que este sistema estará situado idealmente en un lugar en el que reciba luz directa del sol, es necesario que el LCD se pueda ver con facilidad. El LCD 1602 da la capacidad de manejar el contraste mediante un potenciómetro situado en su parte posterior, además se puede activar o desactivar la luz de fondo introduciendo o extrayendo el jumper respectivamente para que la lectura sea lo más clara posible como se puede observar en la Figura 4-21.



Figura 4-21: Parte posterior del LCD 1602 [25]

3. Bajo consumo:

Se han realizado medidas utilizando únicamente el LCD y se ha observado que el consumo de corriente, aún con la luz de fondo establecida, es de aproximadamente 20mA, pudiendo reducirlo si se elimina la luz de fondo extrayendo el jumper, aunque esto empeorará significativamente la visualización de los valores.

4. Conexión I2C configurable:

Para realizar una conexión I2C exitosa se debe saber cuál es la dirección I2C del LCD, en este caso la dirección es configurable, permitiendo así la conexión de diferentes LCDs del mismo tipo. La dirección por defecto es la 0x27 según la hoja de datos del fabricante, además se ha realizado la ejecución de un código el cuál escanea todas las direcciones I2C posibles del Arduino UNO hasta encontrar la que está asignada al LCD.

Cabe destacar que la dirección I2C será configurable únicamente si disponemos del LCD 1602 con el chip PCF8574AT, en este caso se dispondrán de 3 pines bajo el potenciómetro que se podrán soldar para configurar la dirección I2C del módulo.

Debido a estos motivos comentados y a su gran disponibilidad en tiendas, precio e información se ha decidido utilizar este LCD para el proyecto.

El LCD representará diferentes datos en función del estado en el que se encuentre el programa, en caso de que se acabe de iniciar el sistema se visualizará lo mostrado en la Figura 4-22:



Figura 4-22: LCD en su inicialización en modo TEST

Se mostrará el modo en el que se encuentra, si es modo TEST o Normal, y se indicará que los valores de temperatura y humedad faltan por definir TBD (To Be Defined o Por Definir). Si se continua en modo TEST, se acabarán realizando medidas de temperatura y humedad y se mostrarán en el LCD. En cambio, si se inicia el sistema en modo Normal, se mostrará un timestamp, la fecha actual así como los valores de temperatura y humedad como se muestra en la Figura 4-23:



Figura 4-23: LCD en modo Normal

Durante el ajuste de los servomotores, el sistema pausará el LCD, esto se realiza debido a que los servomotores son los módulos que más consumen, y pueden llegar a hacer que la

alimentación del LCD se corté y que por tanto la comunicación I2C sea errónea. Para evitar esto, se pausará la representación y se reanudará una vez ajustado el sistema.

4.1.14 Arduino UNO y Sensor Shield

El Arduino UNO es el cerebro de todo el proyecto, ya que es el encargado de manejar todas las señales, medidas, movimientos, envíos y resultados del sistema. Además se ha optado por utilizar la placa sensor shield para facilitar las conexiones de los motores, fotorresistores, sensores y alimentación, de esta manera el sistema es más presentable.

Se ha utilizado un Arduino UNO R3 el cual tiene un pinout como el mostrado en la Figura 4-24

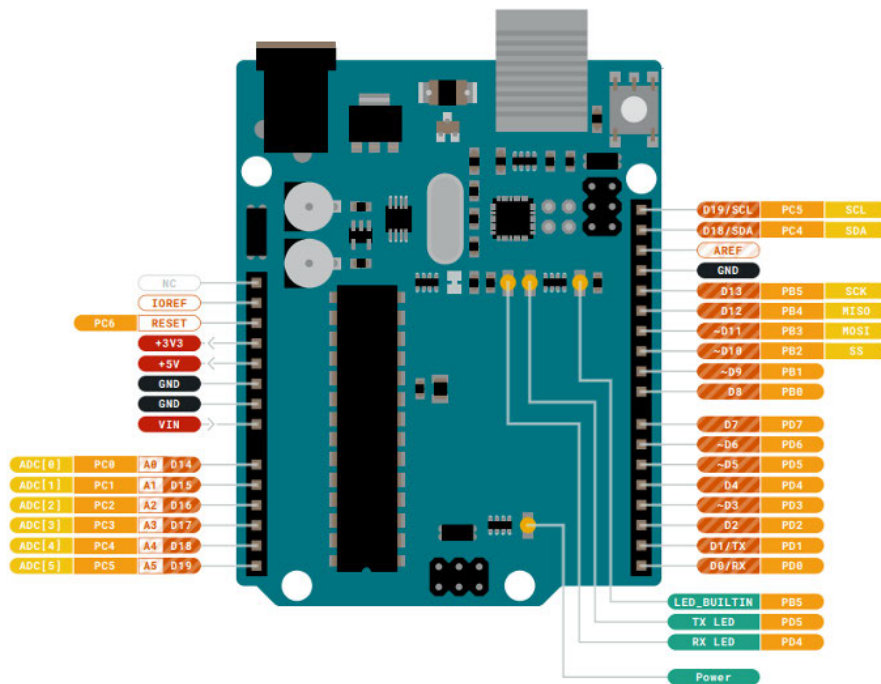


Figura 4-24: Arduino UNO R3 pinout [26]

El Arduino UNO R3 tiene las siguientes características:

1. Alimentación:

Este sistema tiene varias formas de alimentarse, en la hoja de datos [26] se establece que puede ser alimentado desde 2.7V a 5.5V a partir del pin Vin aunque se recomienda utilizar una tensión de entre 7 y 12V. También se le puede alimentar a partir del USB el cual se utiliza para cargar los programas al mismo. Por último, se puede alimentar mediante el Jack externo que admite entre 7 y 12 V.

En esta configuración, se optó por alimentar el Arduino Uno a través de la placa Sensor Shield, utilizando su conector externo. Dado que el Sensor Shield se monta encima del Arduino Uno, al proporcionarle energía, se alimenta directamente el pin Vin del Arduino. Esto garantiza que tanto el Arduino Uno como el Sensor Shield reciban la potencia necesaria para su funcionamiento.

2. Pines:

El Arduino UNO dispone de 6 pines analógicos a través de los cuales se configurará un ADC (Analog Digital Converter o Convertidor Analógico Digital) para realizar las lecturas de los valores de luz, estos se marcan en la Figura 4-24 como Ax.

Dispone de 14 pines digitales, de los cuales 6 de ellos están marcados con el símbolo ~ y pueden configurarse como salidas PWM. Todos estos pines se utilizarán para detectar, mandar señales y también para controlar los servomotores con PWM.

También dispone de pines para la comunicación I2C con una velocidad predeterminada de 100kHz, los cuales se utilizarán para la configuración del sensor BME680 y la representación de los datos en el LCD.

Por último, dispone de pines como 3V3 el cuál aporta 3.3V y 50mA constantes gracias a un regulador interno del Arduino UNO, y el pin 5V que puede servir para alimentar diferentes sensores los cuales necesiten poca corriente y 5V para su funcionamiento.

3. Nivel lógico y ADC:

El nivel lógico con el que funciona un Arduino UNO R3 es de 5V, esto hay que tenerlo en cuenta si se quiere comunicar con otros sistemas con diferentes niveles lógicos como pasa con el ESP8266-01 o si se quiere alimentar algún sistema a partir de un pin digital. Además dispone de un ADC de 10 bits, lo que supone una entrada analógica con 1024 niveles distintos lo que da una resolución bastante buena de aproximadamente 4.88mV para este proyecto.

Todas estas conexiones también se encuentran en el sensor shield para Arduino [27], simplificando así sobre todo la alimentación de los servomotores y LDRs como se puede ver en el esquema mostrado en la Figura 4-25:

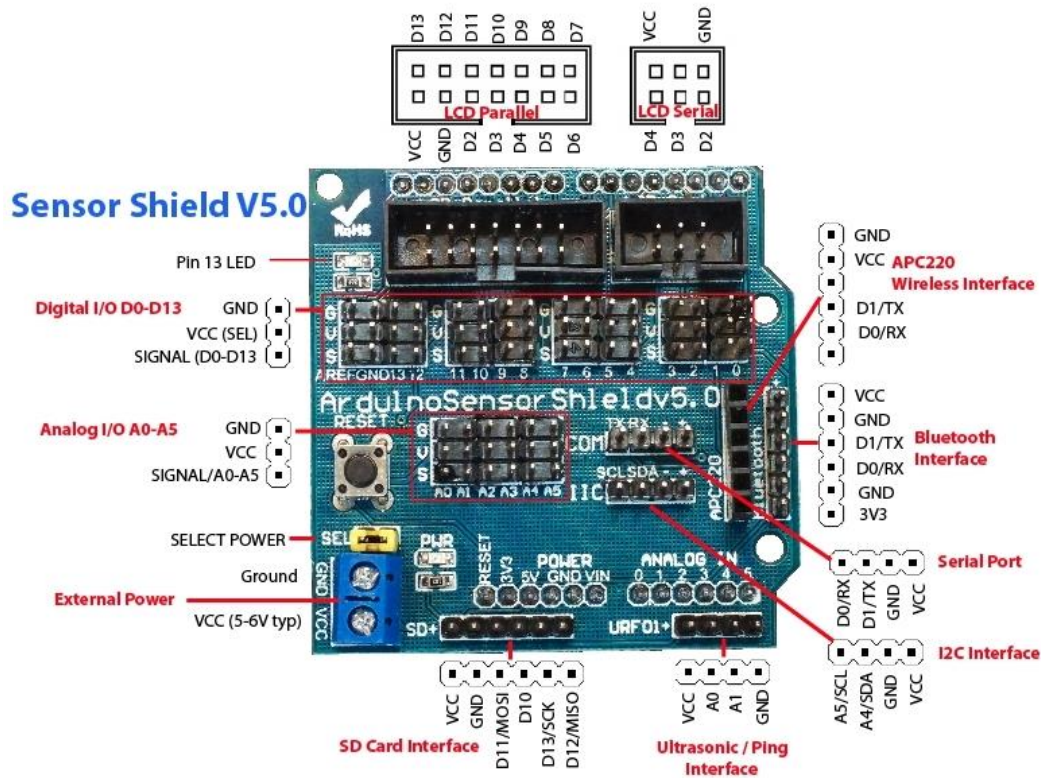


Figura 4-25: Sensor Shield V5.0 pinout [27]

La utilización del sensor shield tiene diferentes ventajas entre las que destacan la facilitación de conexiones, el orden y la organización del sistema, protección contra el Arduino frente a posibles conexiones erróneas y su compatibilidad con múltiples sensores. Aparte de tener todos los pines comentados anteriormente en el Arduino UNO, da la opción de alimentarlo externamente con otra fuente de alimentación en función del jumper “SEL” para seleccionar la alimentación que se desee en los pines digitales:

1. Jumper ON:

Esta ha sido la opción elegida para el proyecto ya que permite que los pines digitales estén alimentados a 5V y por tanto con el mismo nivel lógico que el sistema. Sin embargo, esta opción puede conllevar que cuando los motores estén en movimiento se genere ruido eléctrico en el sistema y ha tenido que ser manejado por software.

2. Jumper OFF:

Esto implica que los pines digitales estarán alimentados directamente desde la batería externa conectada y no desde los 5V del propio Arduino, este tipo de configuración ayuda a reducir el ruido eléctrico generado por los motores. Sin embargo, esto hace que se estén manejando diferentes niveles lógicos en el mismo sistema y no es algo deseable.

4.1.15 MICROPULSADOR

El sistema podrá funcionar con dos modos, el modo TEST y el modo normal, para la selección de este modo se ha optado por utilizar un micropulsador con el montaje mostrado en la Figura 4-26:

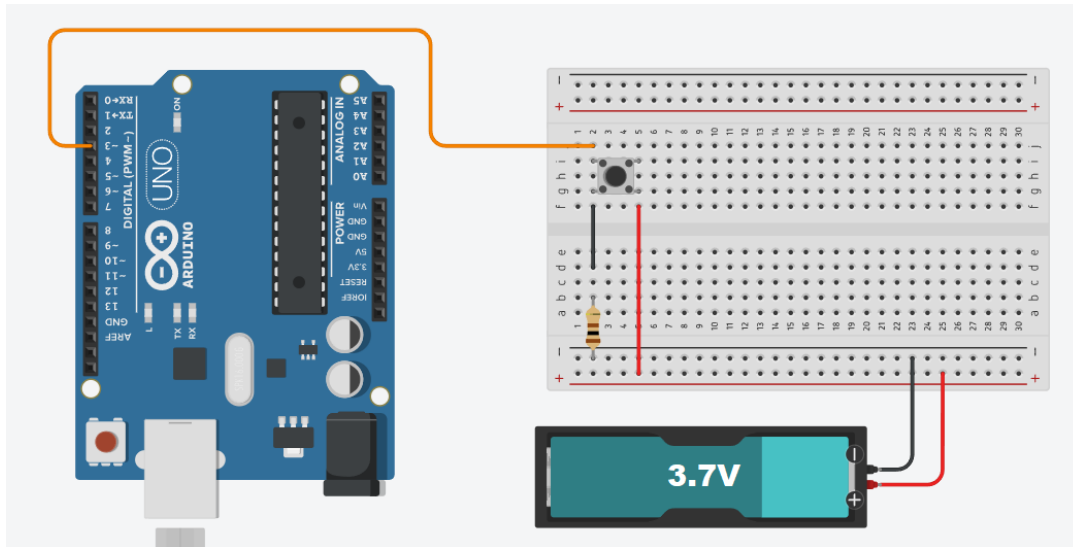


Figura 4-26: Montaje del micropulsador

El funcionamiento de este montaje se basa en que si no se realiza ninguna pulsación, el pin digital 3 del Arduino UNO leerá un nivel bajo gracias a que el pulsador está cortocircuitado a masa con una resistencia. Sin embargo, si se realiza una pulsación el nivel lógico que leerá el Arduino cambiará inmediatamente a nivel alto, ya que cualquier tensión superior a 3.5V se considerará un nivel alto en el sistema.

4.2 Lenguaje de Programación

El lenguaje de programación utilizado para programar en Arduino es una variante simplificada del lenguaje de programación C++. Este lenguaje simplificado se conoce como "Arduino C++". Aunque comparte muchas similitudes con el lenguaje C++ estándar, hay algunas diferencias que hacen que sea más accesible para proyectos de hardware específicos como los que se realizan con placas Arduino.

Además, se ha utilizado un método de programación en bloques facilitado por la herramienta TinkerCad [28] orientado a un objetivo docente, la cual permite simular diferentes circuitos en Arduino UNO aunque con menos profundidad que la que se puede conseguir con la programación en C++.

5 APLICACIÓN SOFTWARE DESARROLLADA

Se ha realizado el diseño de una máquina de estados para la correcta organización del código, cada estado representa una situación específica y un conjunto de acciones enfocadas a un objetivo.

En este apartado se explicará estado a estado para su correcto entendimiento, guiándose del diagrama mostrado en la Figura 5-1, teniendo en cuenta que el color verde marca aquellos estados por los que pasará el sistema en ambos modos, mientras que el color morado marcará los estados que son usados únicamente en modo normal.

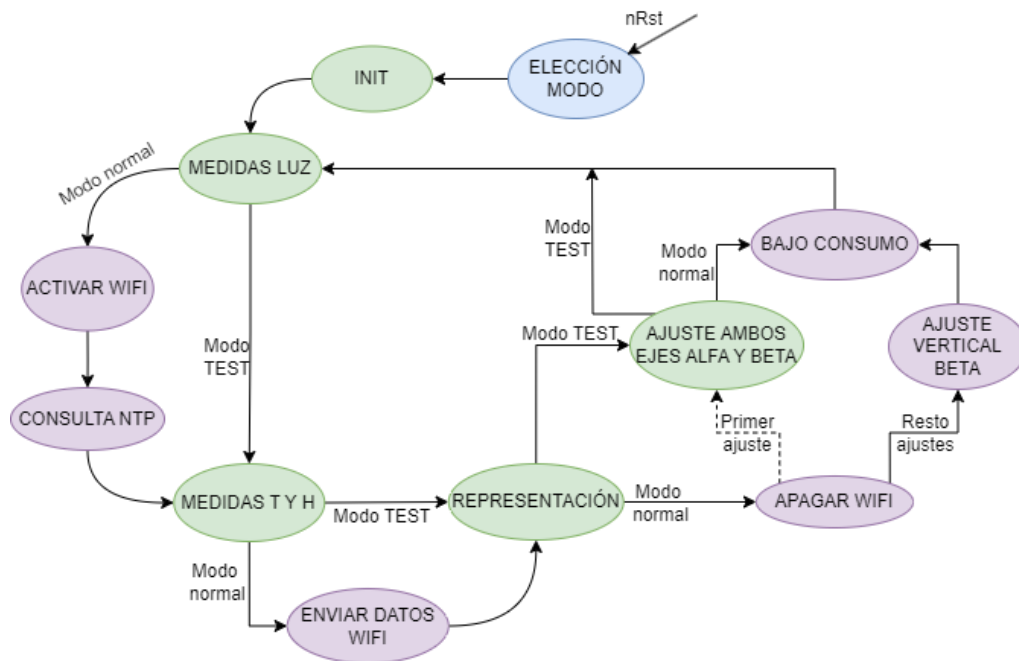


Figura 5-1: Máquina de estados

Durante los primeros cinco segundos de ejecución se elegirá el modo de funcionamiento del sistema, si se realiza una pulsación en el micropulsador se activará el modo normal, el cual está caracterizado por el uso del módulo ESP8266-01 con su correspondiente envío a ThingSpeak y consultas al NTP, además de su entrada en modo bajo consumo. Sin embargo, sino se realiza una pulsación, automáticamente el sistema entrará en modo TEST, este modo está diseñado para mostrar el correcto funcionamiento del seguidor solar, y por tanto, no utilizará el módulo ESP8266-01 ni el modo bajo consumo.

Una vez se ha elegido el modo de funcionamiento se realizará la inicialización de todas las variables a utilizar, seguido de las medidas de luz iniciales que indicarán si se debe realizar o no un movimiento más adelante, además se inicializará el LCD representando el modo de funcionamiento elegido. Si se encuentra en modo normal, se activará el WiFi y se consultará al servidor NTP para obtener la fecha y hora actuales, en cambio, si se encuentra en modo TEST se pasará directamente a realizar medidas de temperatura y humedad por parte del BME680. Una vez se han realizado las medidas, se mandarán los datos a ThingSpeak para su representación online si se encuentra en modo normal, para después pasar esta información al LCD, el cuál representará los valores independientemente del modo en el que se encuentre el

sistema. Una vez se han subido los datos a la plataforma ThingSpeak encontrándose en modo normal, se desactivará el módulo WiFi mediante el transistor LM3940, el cual está controlado por el Arduino uno, de esta manera la autonomía del sistema no se resentirá.

Posteriormente, y teniendo en cuenta que se esté ejecutando el modo normal, se realizará una comprobación para saber si es la primera ejecución del día o no, en el caso de que sí lo sea se realizará un ajuste utilizando ambos servomotores y situando al sistema en el punto de máxima luz. Sino fuese la primera ejecución del día se realizaría únicamente un ajuste vertical del sistema, moviendo por tanto un único servomotor y aumentando la eficiencia del sistema. Ante ambos ajustes, la representación del LCD se pausará para aumentar así la seguridad de su representación. Una vez ajustado, y teniendo en cuenta que se encuentre en modo normal, se reanudará la representación del LCD y entrará en modo bajo consumo durante 5 minutos, reduciendo el consumo total del sistema.

En cambio, si se encuentra en modo TEST, no se hará ninguna comprobación y se ejecutará el ajuste en ambos servomotores, omitiendo la entrada en el bajo consumo pero reanudando la representación del LCD como en el modo normal.

5.1 Elección de modo

Nada más iniciar el sistema, se ejecutará el estado marcado en la Figura 5-2:

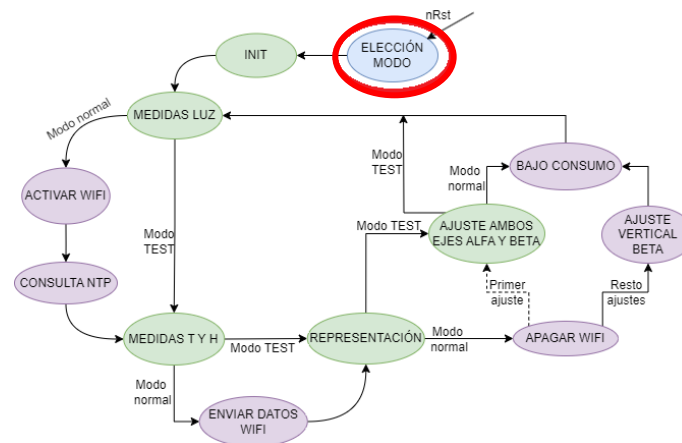


Figura 5-2: Estado elección de modo funcionamiento

Este estado ejecutará la función mostrada más adelante, la cual realizará un bucle que verificará durante los primeros cinco segundos si se realiza una pulsación del micropulsador, este micropulsador estará conectado al pin digital 3. Se guardarán el número de milisegundos que han transcurrido desde el inicio del programa, permitiendo así su ejecución durante los primeros cinco segundos y estableciendo un booleano a true en el caso de que se realice una pulsación. En caso de que no se realice ninguna pulsación el sistema mantendrá a false el booleano, de esta manera se sabrá si se quiere iniciar el programa en modo TEST (booleano a false) o modo normal (booleano a true).

```
bool verificarModoTest(int pinTension) {
    bool tensionDetectada = true; // Booleano para indicar si se ha realizado una pulsacion
    bool primerosCincoSegundos = true; // Variable para controlar los primeros 5 segundos

    unsigned long tiempoInicio = millis(); // Guarda el tiempo de inicio

    while (millis() - tiempoInicio < 5000) { // Permanece en el bucle durante 5 segundos
        if (digitalRead(pinTension) == HIGH) { // Si detecta tension
            tensionDetectada = false; // Establece la variable a true
            break; // Cerramos el bucle
        }
    }
    return tensionDetectada;
}
```

5.2 Init

Una vez se ha elegido el modo de funcionamiento del sistema se ejecutará el estado de inicialización "INIT" mostrado en la Figura 5-3:

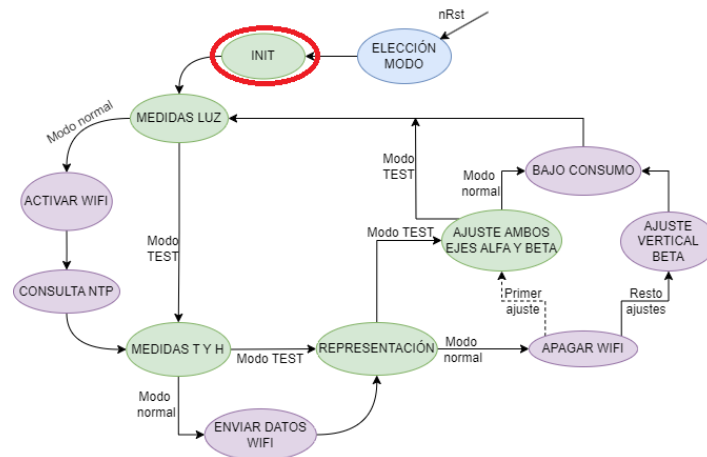


Figura 5-3: Estado de inicialización del sistema

Como en cualquier logaritmo es necesario realizar una inicialización de variables y constantes del sistema que serán usadas más adelante. Al ser un programa extenso el uso de variables y constantes se va a dividir en sectores, inicialmente todas aquellas relacionadas con las medidas de luz, destacando las constantes marcadas en el siguiente código:

```
// Constantes para medidas de luz
const int luz_top_izqd = A0;
const int luz_top_dcha = A1;
const int luz_bot_izqd = A2;
const int luz_bot_dcha = A3;
const int LIGHT_THRESHOLD = 5; // Cambio de luz significativo
const int NUMERO_MEDIDAS = 5; // Numero de medidas de luz realizadas en cada medida
const int UMBRAL_LUZ = 100; // Umbral de luz a partir del cual se realizarán movimientos
```

La constante que indica un cambio de luz significativo es LIGHT_THRESHOLD y será la encargada de indicar al sistema si se debe realizar un movimiento de los servomotores o si en cambio la diferencia no es suficiente para realizarlo. En este caso tiene un valor de 5, teniendo en cuenta que los valores de luz van de 0 a 1023, se detectará un cambio significativo de luz con muy poca variación, en torno al 0.5%, este valor se ha decidido tras realizar pruebas con luz directa del sol.

También se observa la constante que indica el número de medidas que se realizan por cada LDR, entre mayor sea esta, mayor será la precisión de la medida. Por último tenemos el umbral de luz, el cuál cuando esté nublado o de noche indicará al sistema que no se deben mover los motores.

Después del bloque de las medidas de luz, se instanciarán también las constantes y variables para el control de servomotores:

```
const int pinMotor1 = 5; // Ajusta el pin del motor1 según sea necesario
const int pinMotor2 = 6; // Ajusta el pin del segundo motor según sea necesario
const int LIMITE_SERVO_HORIZONTAL_SUP = 170; // Limite del servo horizontal va de 20 a 170
const int LIMITE_SERVO_HORIZONTAL_INF = 20; // Limite del servo horizontal va de 20 a 170
const int LIMITE_SERVO_VERTICAL_SUP = 100; // Limite del servo vertical va de 20 a 100
const int LIMITE_SERVO_VERTICAL_INF = 20; // Limite del servo vertical va de 20 a 100

int pos_sh = 0;
int pos_sv = 0;
Servo servol;
Servo servo2;
```

Los límites de movimiento de los servomotores se han fijado realizando un código de prueba que los mueva en el rango de 0 a 180 grados, el límite se puede identificar fácilmente gracias al sonido mecánico que realizan los motores cuando se llega a este. Para crear las variables de los motores utilizaremos la Librería para los servomotores la cual permite el control de los servomotores con señales PWM de una manera sencilla, por último, se muestran las variables pos_sh (posición servo horizontal) y pos_sv (posición servo vertical) con las que se controlará el ángulo de los motores en todo momento.

Además, se instanciarán las variables para el control del servidor NTP, simplemente serán variables de tipo entero las cuales almacenarán la respuesta en formato, año, mes y día, así como, hora, minutos y segundos. Por último, se inicializarán variables y constantes para el control del LCD (dirección I2C, columnas y filas) el cual mostrará el modo elegido de funcionamiento, la comunicación serial (marcando los pines para la comunicación) y el BME680 (inicialización y control).

5.3 Medidas de luz

Se tratarán en este apartado las sentencias ejecutadas en el estado marcado en la Figura 5-4:

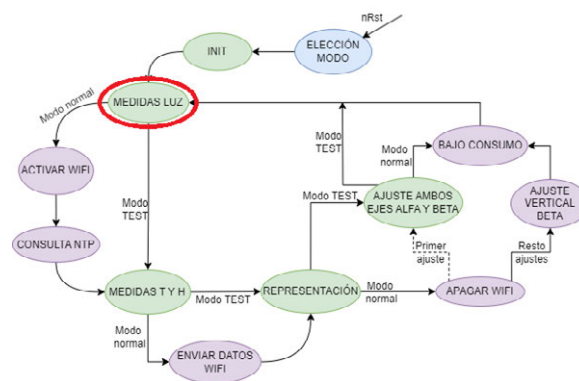


Figura 5-4: Estado para la realización de medidas de luz

Inicialmente será necesario realizar medidas de luz ya que se tienen que ir actualizando los valores para poder realizar los ajustes correctamente, por tanto, esta función es bastante clave ya que es la encargada de realizar dichas medidas y calcular la media de ellas. Esta función a su vez está dividida en dos ya que se desea que el código sea lo más legible posible. Inicialmente,

en la función principal mostrada en el código siguiente, se recibe por parámetro los valores actuales de medias de luz (inicialmente a 0) en cada uno de los ejes, y estos se modificarán siguiendo el siguiente orden:

1. Eje superior con la luz de los fotorresistores superiores tanto izquierdo como derecho.
2. Eje inferior con la luz de los fotorresistores inferiores tanto izquierdo como derecho.
3. Eje izquierdo con la luz de los fotorresistores izquierdos superiores e inferiores.
4. Eje derecho con la luz de los fotorresistores derechos superiores e inferiores.

```
void calcularMedias(int smediaTop, int smediaBottom, int smediaIzquierda, int smediaDerecha) {  
    mediaTop = (calcularMedia(luz_top_izqd, NUMERO_MEDIDAS) + calcularMedia(luz_top_dcha, NUMERO_MEDIDAS)) / 2;  
    mediaBottom = (calcularMedia(luz_bot_izqd, NUMERO_MEDIDAS) + calcularMedia(luz_bot_dcha, NUMERO_MEDIDAS)) / 2;  
    mediaIzquierda = (calcularMedia(luz_top_izqd, NUMERO_MEDIDAS) + calcularMedia(luz_bot_izqd, NUMERO_MEDIDAS)) / 2;  
    mediaDerecha = (calcularMedia(luz_top_dcha, NUMERO_MEDIDAS) + calcularMedia(luz_bot_dcha, NUMERO_MEDIDAS)) / 2;  
}
```

Como se puede observar la función anterior hace una llamada a la función “CalcularMedia” la cual se puede observar en el siguiente fragmento de código, pasándola por parámetro el pin analógico del cuál realizar la medida y la constante “NUMERO_MEDIDAS” que indica el número de veces que se realiza una medida de luz en el pin indicado, aumentando así la precisión de la medida.

```
int calcularMedia(int pinSensor , int muestras)  
{  
    int totalLectura = 0;  
  
    for (int i = 0; i < muestras; i++)  
    {  
        totalLectura += analogRead(pinSensor);  
    }  
    delay(50);  
    return totalLectura / muestras;  
}
```

5.4 Activar WiFi

Esta parte del software está centrada en el ESP8266-01, concretamente en el estado marcado en la Figura 5-5 que es el encargado de realizar la conexión WiFi si el sistema se encuentra en modo normal de ejecución y sigue un orden específico.

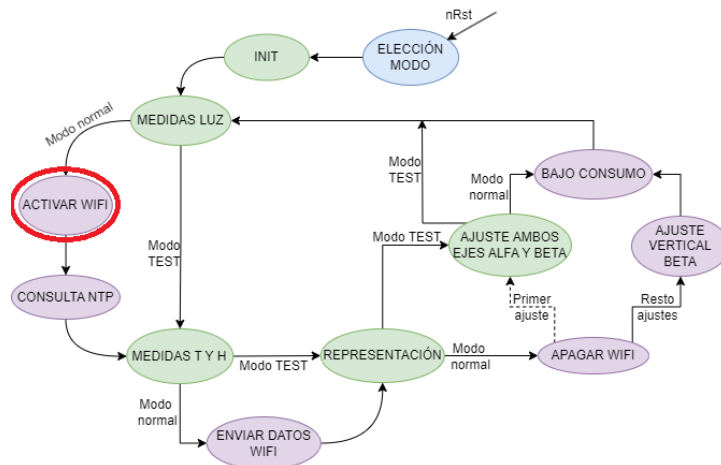


Figura 5-5: Estado de activación del WiFi

5.4.1 Configuración de comunicación serial

Previamente a la conexión WiFi se debe establecer la comunicación serial junto con la velocidad de esta. Para elegir esta velocidad se debe tener en cuenta que, inicialmente, el ESP8266-01 está configurado con 115200 bauds, mientras que el Arduino uno, por defecto, utiliza 9600 bauds. Los bauds indican la cantidad de cambios de estado que es capaz de realizar una comunicación por segundo, para que los dos sistemas sean capaces de entenderse el uno al otro y pasarse información es necesario que esta velocidad sea igual, para ello configuraremos el ESP8266-01 con la velocidad de 9600 bauds mediante comandos AT (Attention o Atención). Los comandos AT [29] son una lista de comandos de Arduino, los cuáles permiten la configuración de diferentes aspectos, como puede ser la velocidad de transmisión, indicar el modo de operación... En este caso se ha utilizado la comunicación serial del Arduino IDE (Integrated Development Environment o Entorno de Desarrollo Integrado) como se puede ver en la Figura 5-6 para dicha configuración, aunque se podría utilizar cualquier programa con acceso a los puertos seriales de nuestro ordenador.

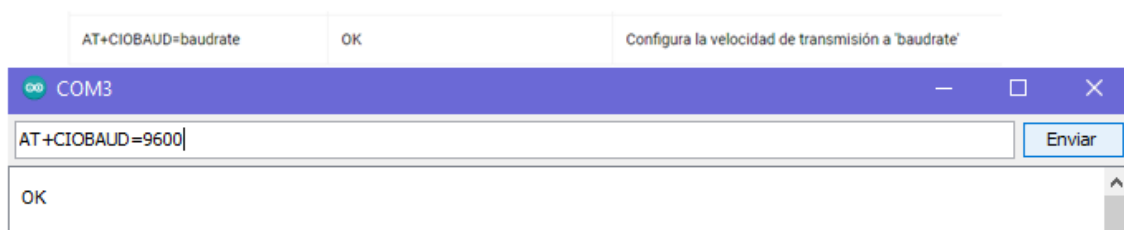


Figura 5-6: Comando AT para la configuración de velocidad de transmisión

Una vez configurada la velocidad de transmisión podemos declarar en el setup() o configuración inicial del código la comunicación serial simplemente como "Serial.begin(9600)".

5.4.2 Activación 2N3904

El módulo WiFi es un sistema que consume bastante en el proyecto, pudiendo llegar a picos de corriente de hasta 200mA, para alargar la autonomía del proyecto se controlará el encendido mediante el transistor bipolar 2N3904. La manera de controlar esto por software es sencilla utilizando las funciones de control de los pines digitales, en este caso la base del bipolar estará

conectada al pin 3 del Arduino UNO, por lo que bastará con indicar que este PIN tiene un nivel alto o bajo para encender o apagar respectivamente el módulo WiFi.

5.4.3 Conexión WiFi

Una vez se ha configurado la comunicación serial y se ha activado el transistor 2N3904, el sistema se conectará al WiFi, para esto se necesitará la librería “ESP8266WiFi.h” [30] la cual facilitará la codificación de dicha conexión, se definirán dos constantes para guardar el nombre (SECRET_SSID) y contraseña (SECRET_PASS) de la red a la que se conectará:

```
#define SECRET_SSID " "
#define SECRET_PASS " "
```

Se ha creado la función “conectarWiFi()” que se puede ver el fragmento de código siguiente, la cuál es llamada desde la configuración inicial del sistema para que el código sea más claro y conciso. Esta función intenta iniciar la conexión a la red WiFi utilizando las credenciales especificadas en las constantes SECRET_SSID (nombre de la red) y SECRET_PASS (contraseña). Posteriormente se iniciará el bucle while, el cual se ejecutará continuamente hasta que la conexión sea exitosa.

```
void conectarWiFi() {
  // Conectar a la red Wi-Fi
  WiFi.begin(SECRET_SSID, SECRET_PASS);

  Serial.print("Conectando a la red Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("\nConexión exitosa");
}
```

5.5 Consulta NTP

Una vez se ha activado y conectado el WiFi encontrándose el sistema en modo normal de ejecución, se realizará el estado mostrado en la Figura 5-7:

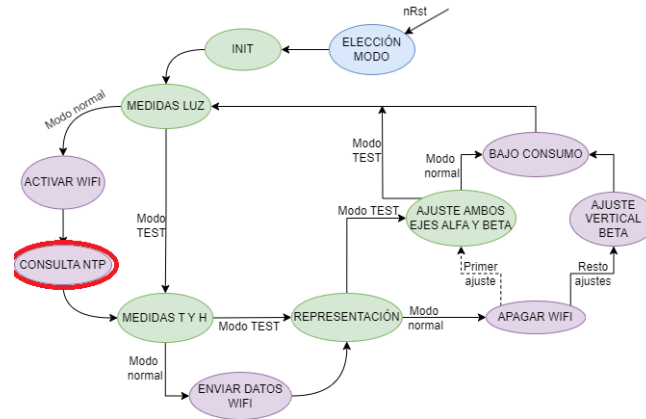


Figura 5-7: Estado de consulta al servidor NTP

Se realizará una consulta a un servidor NTP para lo que se han utilizado las siguientes librerías de libre acceso de Arduino: “NTPClient.h” [31] y “WiFiUDP.h” [32]. Inicialmente se ha creado una constante y objetos para la consulta, ya que se desea obtener la hora local, se han sumado dos horas (7200 segundos) a la respuesta que dé el servidor NTP, ya que esta estará en UTC (Coordinated Universal Time o Tiempo Universal Coordinado), y la hora en España para el horario de verano es de UTC + 2. Esto implica que el sistema siempre se encuentre en horario de verano incluso cuando se realice el cambio horario.

```
const long utcOffsetInSeconds = 7200; // UTC+2
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org", utcOffsetInSeconds);
```

Se crea una instancia de la clase WiFiUDP llamada ntpUDP, esta clase se utiliza para establecer una conexión UDP, el cual es un protocolo de comunicación sin conexión que se utiliza para la transmisión de datos en redes, es necesario su creación para la creación del cliente NTP.

Por último se crea un cliente NTP, el cual se configura con el protocolo UDP, se le marca el servidor NTP al que hacer la consulta, en este caso “pool.ntp.org” aunque se podría haber utilizado otro como “time.google.com” sin problema. Además, se marca el offset del UTC como se ha comentado previamente.

Por tanto, para iniciar la comunicación con el servidor NTP se utilizará en la configuración inicial del sistema(setup) la llamada “timeClient.begin()”.

5.5.1 Envío y procesamiento de datos al Arduino UNO

Para el envío de información al Arduino UNO se ha creado una función “getAndSendTime()” que se puede ver en el siguiente fragmento de código la cuál será llamada en el bucle loop del programa, de esta manera conseguiremos que se haga una consulta para obtener el tiempo epoch del servidor cada vez que se arranque el módulo WiFi. Esta función actualizará la hora almacenada en el objeto timeClient con la hora actual del servidor NTP, almacenando el tiempo epoch en la variable que se mandará por la comunicación serial.

```
void getAndSendTime() {  
    timeClient.update();  
    long epochTime = timeClient.getEpochTime();  
  
    Serial.println(epochTime); // Segundos desde 1970  
  
}
```

Una vez se ha realizado la consulta al servidor NTP, el módulo ESP8266-01 se dormirá mediante un delay de 5 segundos, para dar tiempo al Arduino UNO a que procese la información obtenida.

Para que el Arduino UNO pueda recibir dicha información se necesitará la librería “SoftwareSerial.h” [33] la cual configurará los pines 10 y 11 como se puede ver en el siguiente código para admitir la comunicación serial con el ESP8266-01 y la siguiente configuración inicial:

```
SoftwareSerial ESPSerial(10, 11); // RX, TX
```

Después de esto solo quedará procesar la información utilizando la librería “Time.h” [34], la cuál será la encargada de facilitar el desglose de los segundos en el formato dd/mm/yyyy y hh:mm:ss como se ve en el siguiente código a continuación:

```
void processResponse() {  
    if (ESPSerial.available() > 0) {  
        String response = ESPSerial.readStringUntil('\n');  
        long epochTime = response.toInt(); // Convertir el epoch time de String a long  
  
        // Convertir epoch time a una estructura de fecha y hora  
        struct tm *timeinfo;  
        time_t epoch = (time_t)epochTime;  
        timeinfo = localtime(&epoch);  
  
        // Desglosamos lo obtenido del epoch  
        anio = timeinfo->tm_year + 1970; // Año  
        mes = timeinfo->tm_mon + 1; // Mes  
        dia = timeinfo->tm_mday; // Día del mes  
        hora = timeinfo->tm_hour; // Hora  
        minutos = timeinfo->tm_min; // Minutos  
        segundos = timeinfo->tm_sec; // Segundos  
    }  
}
```

Primero se verificará si hay datos en el puerto serial para ser leídos, en el caso de que los haya leerá los datos hasta el salto de línea, para así no perder información. Por último, utilizamos las funciones facilitadas por la librería para desglosar correctamente cada uno de los valores.

Una vez se ha procesado la fecha y la hora actual devuelta por el servidor NTP, se verificará si se ha cambiado de día y por tanto, si se tiene que realizar un ajuste de ambos ejes o no.

5.6 Medida temperatura y humedad

El estado mostrado en la Figura 5-8 se realizará independientemente del modo de funcionamiento en el que se encuentre el sistema, para las medidas de temperatura y humedad utilizaremos la librería “Adafruit_Sensor.h” [35] y “Adafruit_BME680.h” [36], así como la librería “Wire.h” [37] la cual facilitará la comunicación I2C con el sensor.

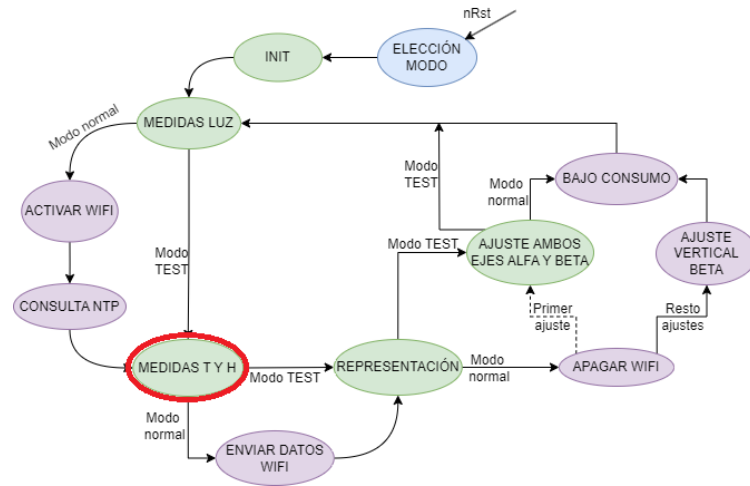


Figura 5-8: Estado de realización de medidas T y H

Inicialmente se tendrá que crear un objeto bme e inicializarlo en el setup() del código. Una vez hecho, las librerías aportan dos funciones muy sencillas para obtener el valor de temperatura y humedad, estas dos funciones devolverán un float, por lo que se crearán dos variables para almacenar dichos valores tal y como se muestra a continuación:

```
// Leer temperatura y humedad del sensor
float temp = bme.readTemperature();
float humidity = bme.readHumidity();
```

El sistema realizará una medida de temperatura y humedad por parte del sensor BME680 controlado desde el Arduino UNO por comunicación I2C. Una vez se ha estabilizado la medida se manda por el puerto serial al módulo ESP8266-01 que será el encargado de comunicarse con la web ThingSpeak para que represente los valores.

Además, esta información estará representada en el LCD 1602, el cual también se configurará con comunicación I2C, el objetivo de este módulo es realizar una comprobación de que los datos visualizados en la web sean los mismos que los mostrados en el LCD.

5.7 Enviar datos WiFi

Una vez realizadas las medidas de temperatura y humedad, y teniendo en cuenta que se encuentre en modo normal de ejecución se ejecutará el estado mostrado en la Figura 5-9:

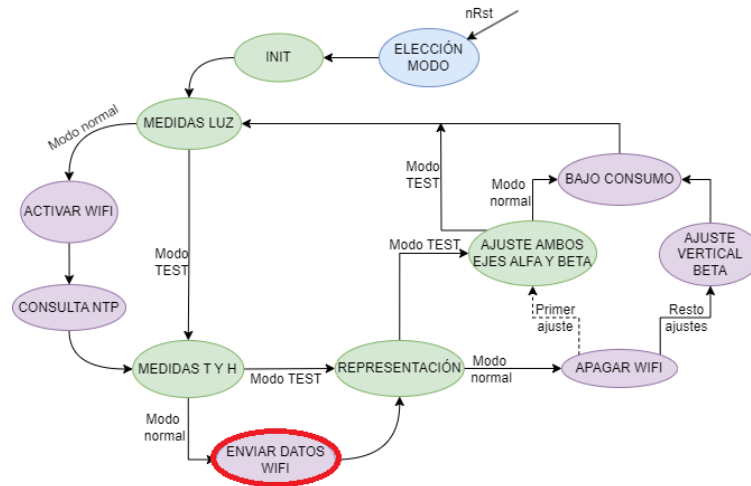


Figura 5-9: Estado para el envío de datos por WiFi

El Arduino UNO mandará los datos al módulo ESP8266-01, el cuál será el encargado de subir estos valores a ThingSpeak que es una web con soporte de MathWorks la cual permite procesar estos valores y representarlos en una gráfica. El código realizado para este envío es sencillo y consiste en escribir en la comunicación serial los valores separados en el formato “Dato1: Temperatura” y “Dato2: Humedad” tal y como se muestra en el siguiente código para que el módulo ESP8266-01 sea capaz de distinguirlos y procesarlos correctamente.

```

// Envío de temperatura y humedad al ESP y la web
enviarTemperaturaHumedad(temp, humidity);

void enviarTemperaturaHumedad(float temp, float humidity) {
    // Enviar temperatura y humedad por el puerto serie
    Serial.print("Dato1:");
    Serial.print(temp);
    Serial.print(" Dato2:");
    Serial.println(humidity);

    // Envía la temperatura y la humedad actual al ESP8266-01 por el puerto serie del ESP
    ESPSerial.print("Dato1:");
    ESPSerial.print(temp);
    ESPSerial.print(" Dato2:");
    ESPSerial.println(humidity);
}

```

En el código del ESP8266-01 y tras la espera de 5 segundos a que el Arduino UNO procese la información del servidor NTP, estará leyendo activamente de la comunicación serial como se muestra en el código siguiente a la espera de recibir la información de temperatura y humedad que se mandará a ThingSpeak, para esto se usará la librería “ThingSpeak.h” [38], la cual facilitará el envío de datos a la web y su representación. Así como la librería “WiFiUdp.h” [32] para crear el cliente que se conectará a la WiFi y a la web.

```
// Leer el valor de la comunicación serial
if (Serial.available() > 0) {
    String dato = leerDato();

    // Enviar el valor a ThingSpeak
    enviarDatosThingSpeak(dato);
}
```

Para utilizar la librería de ThingSpeak es necesario declarar el identificador del canal y la clave de su API (Application Programming Interface o Interfaz de Programación de Aplicaciones) la cual se puede encontrar en la web como muestra la Figura 5-10 y en el código a continuación una vez se ha creado un perfil.

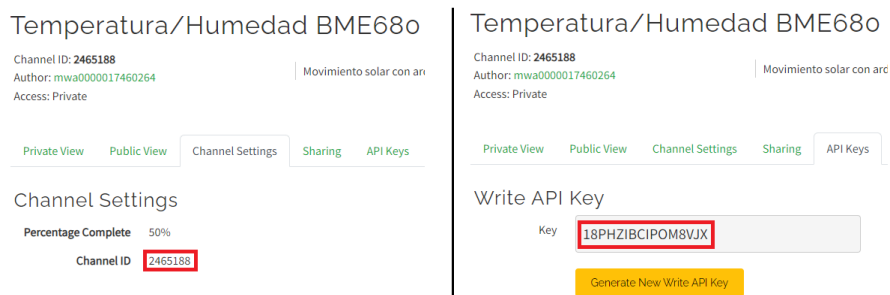


Figura 5-10: Identificador y api key de ThingSpeak

```
#define SECRET_CH_ID 2465188
#define SECRET_WRITE_APIKEY "18PHZIBCIPOM8VJX"
```

Primero se verificará si hay datos en el puerto serial y en caso de que los haya se realiza la llamada a “leerDato()”, esta función leerá hasta el próximo salto de línea, así se asegurará de que no se pierda información.

Después de esto se enviarán los datos a ThingSpeak mediante la función “enviarDatosThingSpeak” que se muestra a continuación, esta función busca la posición de los caracteres “Dato1:” o “Dato2:”, en caso de que no los encuentre devolverá un -1 y por tanto no se mandará nada a ThingSpeak. Cuando encuentre cualquiera de estas cadenas extraerá el valor numérico del valor que le llega por parámetro y los guardará en dos floats creados que se transmitirán a la web utilizando “setField” y “writeFields” de la librería ThingSpeak. En esta función se marcará inicialmente el gráfico en el que queremos escribir en la web (1 para temperatura y 2 humedad) y posteriormente el valor a escribir.

```
void enviarDatosThingSpeak(String dato) {
  // Buscar el prefijo "Dato1:" y "Dato2:"
  int prefijoIndice1 = dato.indexOf("Dato1:");
  int prefijoIndice2 = dato.indexOf("Dato2:");

  if (prefijoIndice1 != -1 && prefijoIndice2 != -1) {
    // Extraer la parte numérica después del prefijo para Dato1
    float valorFloat1 = extraerValor(dato, "Dato1:");

    // Extraer la parte numérica después del prefijo para Dato2
    float valorFloat2 = extraerValor(dato, "Dato2:");

    // Configurar los campos en ThingSpeak
    ThingSpeak.setField(1, valorFloat1);
    ThingSpeak.setField(2, valorFloat2);

    // Enviar datos a ThingSpeak
    ThingSpeak.writeFields(SECRET_CH_ID, SECRET_WRITE_APIKEY);
  }
}
```

A continuación, y en la Figura 5-11 se muestran los datos guardados en las bases de datos de ThingSpeak:

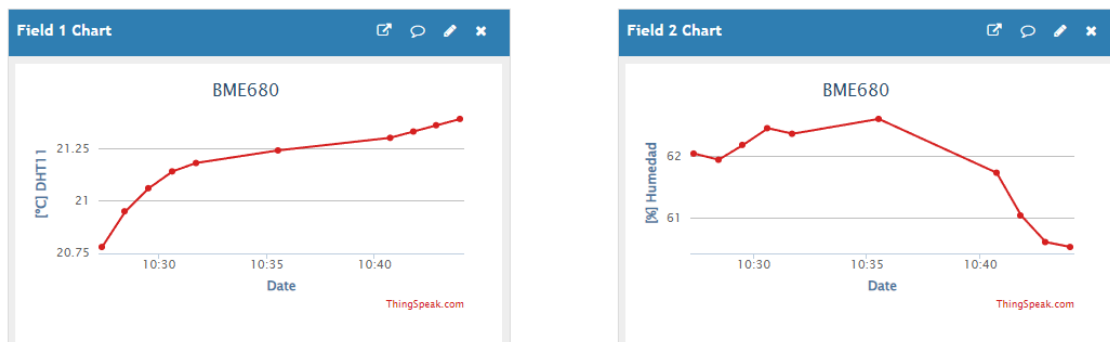


Figura 5-11: Resultados mostrados en ThingSpeak

5.8 Representación

En el estado mostrado en la Figura 5-12 se realizará la representación de los valores de humedad y temperatura extraídos del BME680 independientemente del modo en el que se encuentre el sistema.

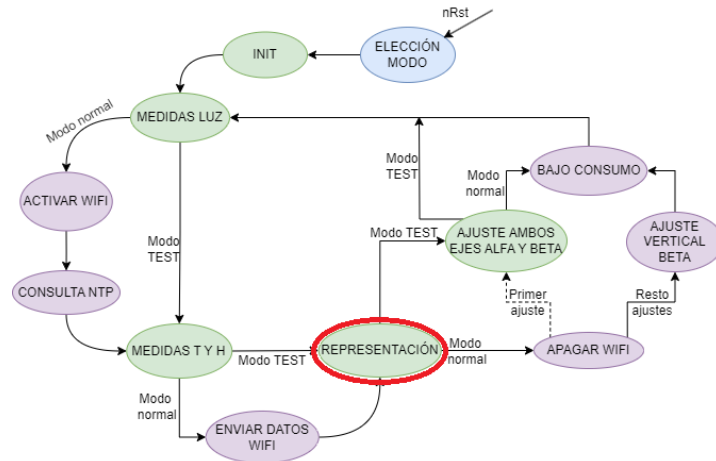


Figura 5-12: Estado de representación

En este caso se han creado dos funciones fundamentales, la de inicialización del LCD y la de la representación de los valores. Utilizando la librería proporcionada por Arduino, se podrá elegir en qué línea del LCD se quiere escribir mediante el cursor, realizar la inicialización del mismo e incluso configurar el encendido o apagado de la luz trasera.

Se configurará al sistema encendiendo la luz de fondo y escribiendo dónde irán los valores de temperatura y humedad, así como el modo de funcionamiento elegido en función de la variable modoTest como se puede ver a continuación:

```
// Inicializacion del LCD
void inicializarLCD() {
    lcd.init();           // Inicializar el LCD
    lcd.backlight();      // Encender el backlight
    lcd.setCursor(0, 0);  // Establecer el cursor en la posición (0, 0)
    lcd.print("T: TBD");  // Imprimir el texto inicial en la primera fila
    lcd.print("  H: TBD"); // Imprimir el texto inicial en la segunda fila
    lcd.setCursor(0, 1);
    if(modosTest){
        lcd.print("Modo TEST");
    }else{
        lcd.print("Modo normal");
    }
}
```

Por último, se realizará la escritura en el LCD como en el siguiente código una vez se han recibido los valores de temperatura y humedad, se pasarán por parámetro para que se puedan representar.

Inicialmente se limpiará la pantalla utilizando la función "lcd.clear()", de esta manera se evitan sobreescrituras en el LCD. En el caso de que se encuentre en modo normal, aparte de la representación de los valores otorgados por el BME680, también se podrá ver la hora actual o timestamp de la medida, así como el día y mes actual. El código utilizado solo verifica el valor de la variable modoTest y en función de esta representa una opción o la otra.

```
// Funcion escritura del LCD
void escribirLCD(float temp, float humidity, int dia, int mes,
                int hora, int minutos, int segundos) {
    lcd.clear(); // Limpiar la pantalla LCD
    // Escribir temperatura y humedad en el LCD
    lcd.setCursor(0, 0);
    lcd.print(temp);
    lcd.print(" C ");
    lcd.print(humidity);
    lcd.print(" %");
    // Escribir el timestamp y el dia/mes
    lcd.setCursor(0, 1);
    if(modosTest == false){
        lcd.print(hora);
        lcd.print(":");
        lcd.print(minutos);
        lcd.print(":");
        lcd.print(segundos);
        lcd.print(" ");
        lcd.print(dia);
        lcd.print("/");
        lcd.print(mes);
    }
    else{
        lcd.print("Modo TEST");
    }
}
```

Durante el proceso del movimiento de los motores, no se representará ningún valor en el LCD debido a que el consumo de corriente de los servomotores podría consumir demasiado y llegar a afectar a la alimentación del LCD, provocando así una mala comunicación I2C y por tanto, el mal funcionamiento del sistema. El proceso para dejar de mostrar y volver a mostrar los valores será mediante la función noDisplay y Display que proporciona la librería de Arduino.

5.9 Apagar WiFi

Una vez se han representado los valores medidos se ejecutará el estado mostrado en la Figura 5-13:

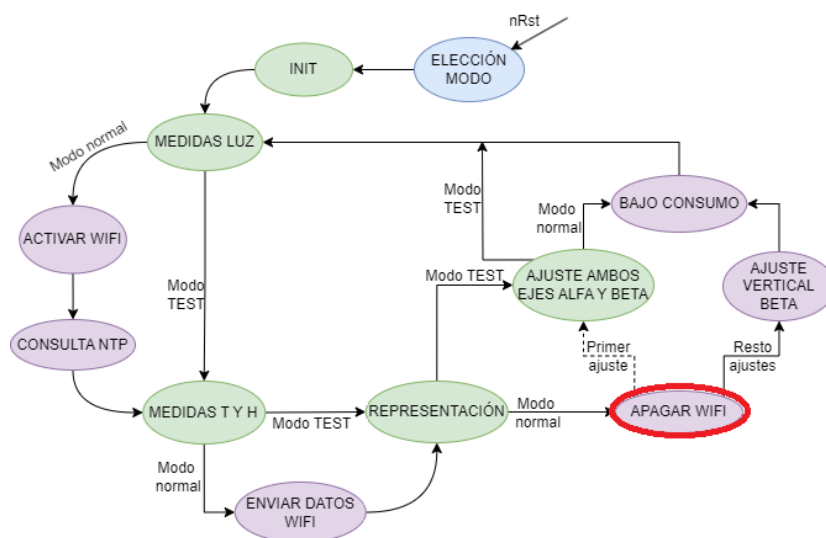


Figura 5-13: Estado encargado del apagado del WiFi

El módulo WiFi que es el ESP8266-01 es un sistema bastante exigente en lo que se refiere a corriente. Para evitar que este consumo se dispare a lo largo del tiempo se desactivará la alimentación de este módulo mediante un transistor bipolar 2N3904. El sistema se apagará en dos ocasiones.

1. Cuando ha obtenido respuesta del NTP al inicio del programa y se ha procesado.
2. Cada vez que reciba los datos de temperatura y humedad y se hayan mandado correctamente a ThingSpeak.

El Arduino UNO simplemente actuará sobre la base del bipolar (conectada al pin 3), permitiendo así el paso de corriente de colector a emisor, la sentencia de encendido y apagado del WiFi es realmente sencilla como se ha comentado anteriormente en Activar WiFi únicamente se establecerá un nivel alto o bajo en el pin digital.

5.10 Condición ejecución diaria

El sistema cuenta con dos servomotores, uno para realizar un ajuste horizontal del panel solar, es decir, el movimiento en el ángulo α , y el otro para realizar el ajuste vertical, en el ángulo β , tal y como se muestra en la Figura 5-14. Para buscar la máxima eficiencia entre la relación de orientación óptima y de consumo de nuestro sistema, se ha decidido realizar un algoritmo el cuál verifique si es la primera ejecución del día o no.

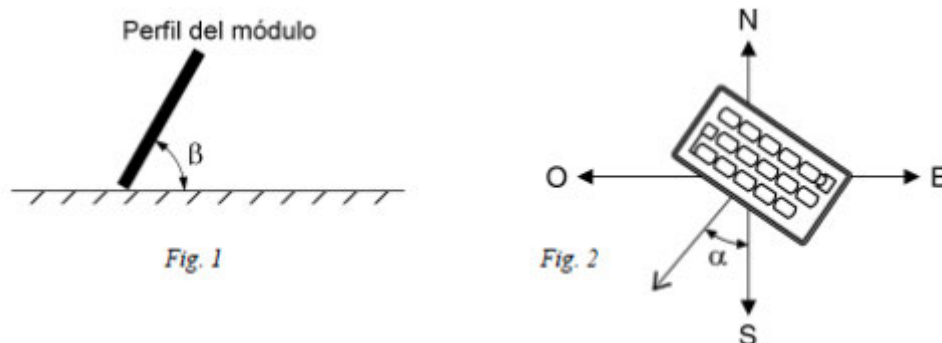


Figura 5-14: Ángulos de inclinación y declinación módulo FV

5.10.1 ¿Por qué este método?

Durante el transcurso del día el sol se mueve de Este a Oeste en el cielo, por lo que el ajuste horizontal o α es solo necesario al inicio del día para que apunte hacia el Este, durante el resto del día será suficiente con variar el eje vertical o β . Esto permite que el sistema sea mucho más eficiente energéticamente además de simple y fiable.

5.10.2 Código implementado

En términos de código es algo sencillo de implementar, inicialmente se obtendrá el día actual en el que se encuentra el sistema como se describe en el apartado de Consulta NTP, se comprobará en el bucle loop si ese día ha cambiado y se asignará un booleano en función del resultado:

```
while(anio == 1970){    // Espera una respuesta correcta del NTP
    processResponse();
    delay(50);
}

// Comprobar si el día ha cambiado
if (dia != dia_anterior) {
    dia_anterior = dia;
    ajusteInicial = true;    // Hay que hacer el ajuste inicial
}
```

Posteriormente y con el booleano ajustado, se realizará un ajuste u otro en función de lo que corresponda.

5.11 Ajuste ambos ejes α y β

Una vez se ha determinado si el ajuste se debe hacer en uno o ambos ejes, en caso de ser la primera ejecución del día se ejecutará el estado mostrado en la Figura 5-15

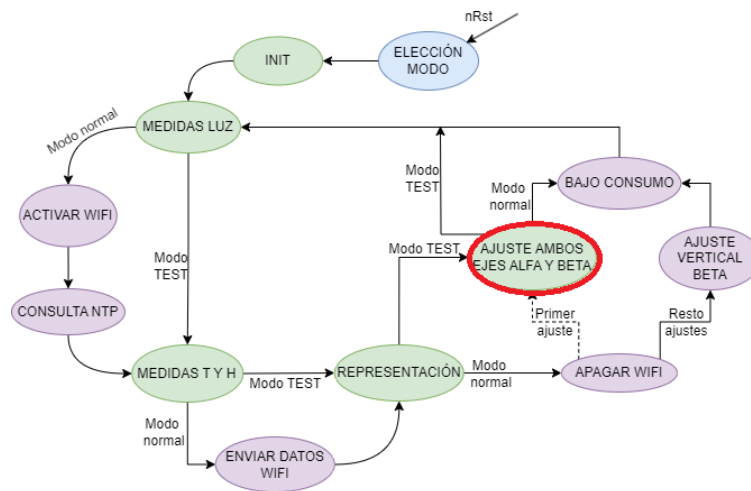


Figura 5-15: Estado para el ajuste de ambos ejes

Como se ha especificado en el punto anterior, en caso de que tengamos que realizar el ajuste inicial del sistema o se encuentre en modo TEST, se tendrá que realizar el ajuste de ambos motores, el horizontal para orientarlo idealmente al Este y el vertical para obtener la máxima luz posible. El código principal que es llamado desde el bucle loop del Arduino UNO contiene llamadas a múltiples funciones de tal manera que el código fuese lo más limpio posible.

A continuación, se explicará la función general para posteriormente entrar en detalles para cada una de ellas:

5.11.1 Función general

Para realizar un ajuste de ambos ejes, aparte de tener que ser la primera ejecución del día, tendrá que cumplir que haya suficiente luz, ya que en caso de que sea de noche o se nuble no tiene sentido que el sistema se ajuste, aunque sí que se tomaran medidas de temperatura y humedad.

En este sistema se ha establecido un umbral de luz de 100 (valor obtenido cubriendo los fotorresistores), teniendo en cuenta que el valor máximo de luz que se puede obtener es de

1023 ya que las lecturas analógicas del Arduino UNO son de 10 bits debido a su convertidor analógico digital.

Si las dos condiciones se cumplen se entrará al bucle del ajuste tanto vertical como horizontal, este bucle se ejecutará mientras:

1. La diferencia de luz entre los sensores superiores e inferiores supere un umbral de luz de 5 y se encuentre entre los límites verticales.
2. La diferencia de luz entre los sensores izquierdos y derechos supere un umbral de luz de 5 y se encuentre entre los límites horizontales.

Mientras se encuentre en este bucle realizará actualizaciones de las medidas de luz mediante la función “CalcularMedias” y el seguidor solar se moverá en función de los resultados obtenidos, desactivando previamente la muestra de valores del LCD para proteger frente a una posible mala comunicación I2C por falta de corriente. Además, se añade un pequeño delay para que el movimiento de los servos sea más fluido.

Una vez se ha terminado el ajuste pondremos la variable booleana “ajusteInicial” a false, para que la próxima vez no se ajusten ambos servomotores a no ser que se detecte un cambio de día por el servidor NTP.

5.11.2 Función para determinar cantidad de luz suficiente

La primera comprobación que se hace antes de iniciar un movimiento de los servomotores es comprobar la cantidad de luz que hay, en caso de que haya suficiente se realizará un ajuste, sino no se hará nada. La función devolverá true si cualquiera de los fotorresistores lee un valor de luz lo suficientemente alto marcado por la constante UMBRAL_LUZ que vale 100 tal y como se ve a continuación:

```
// Sera true si hay luz suficiente para mover los motores
bool luzSuficiente(void){
    lectA0 = analogRead(luz_top_izqd);
    lectA1 = analogRead(luz_top_dcha);
    lectA2 = analogRead(luz_bot_izqd);
    lectA3 = analogRead(luz_bot_dcha);

    return (lectA0 > UMBRAL_LUZ || lectA1 > UMBRAL_LUZ || lectA2 > UMBRAL_LUZ || lectA3 > UMBRAL_LUZ);
}
```

Se utiliza la función analogRead() la cuál realiza una lectura en el pin analógico que se le pase por parámetro en este caso A0 se corresponde con el LDR conectado arriba a la izquierda(luz_top_izqd), A1 arriba a la derecha(luz_top_dcha), A2 abajo izquierda(luz_bot_izqd) y por último A3 abajo derecha(luz_bot_dcha).

5.11.3 Función para limitar movimiento horizontal y vertical

Se ha desarrollado una función para averiguar si el sistema se encuentra entre los límites verticales y otra para los horizontales con funcionamiento prácticamente idéntico, a este tipo de funciones les llega una posición, pos_sv si es en el movimiento vertical y pos_sh si es en el movimiento horizontal, y en el caso de que dicha posición no se encuentre en el límite superior

o inferior del eje que corresponde devolverá un true y el sistema se seguirá ajustando con normalidad.

Pero si esta posición coincide con cualquiera de los límites devolverá un false, indicando así que no se debe realizar ningún ajuste más. Además, en caso de que llegue al límite en cualquiera de las posiciones y para que el servomotor no se quede bloqueado en un límite que pueda dañarlo se hará lo siguiente:

1. Si se llega al límite superior se le restarán 2 posiciones.
2. Si se llega al límite inferior se le sumarán 2 posiciones.

Este funcionamiento se puede observar en el siguiente código para el límite horizontal.

```
// Funcion que indica si nos encontramos entre los limites horizontales
// True -> Te encuentras entre los limites
// False -> Te encuentras en algún limite
bool entreLimitesHorizontales(int posicion){
    bool limites = true;
    if((pos_sh != LIMITE_SERVO_HORIZONTAL_SUP) && (pos_sh != LIMITE_SERVO_HORIZONTAL_INF)){
        limites = true;
    }else if(pos_sh == LIMITE_SERVO_HORIZONTAL_SUP){
        pos_sh = LIMITE_SERVO_HORIZONTAL_SUP - 2; // Le restamos para que no esté justo en el limite y pueda ajustarse posteriormente
        limites = false;
    }else if(pos_sh == LIMITE_SERVO_HORIZONTAL_INF){
        pos_sh = LIMITE_SERVO_HORIZONTAL_INF + 2; // Le sumamos para que no esté justo en el limite y pueda ajustarse posteriormente
        limites = false;
    }
    return limites;
}
```

5.11.4 Función para movimiento de ambos ejes

Esta función mostrada a continuación es la encargada de realizar y controlar el movimiento de los servomotores en función de los niveles medios de luz que se han calculado previamente.

```
void moverSeguidorSolar(int average_top, int average_bottom, int average_left, int average_right) {
    // Movemos el solar tracker hacia arriba o hacia abajo
    if ((average_top - average_bottom > LIGHT_THRESHOLD) && (pos_sv < LIMITE_SERVO_VERTICAL_SUP)) {
        pos_sv = pos_sv + 2;
        servo2.write(pos_sv);
    } else if ((average_bottom - average_top > LIGHT_THRESHOLD) && (pos_sv > LIMITE_SERVO_VERTICAL_INF)) {
        pos_sv = pos_sv - 2;
        servo2.write(pos_sv);
    }

    // Movemos el solar tracker hacia la derecha o hacia la izquierda
    if ((average_left - average_right > LIGHT_THRESHOLD) && (pos_sh > LIMITE_SERVO_HORIZONTAL_INF)) {
        pos_sh = pos_sh - 2;
        servo1.write(pos_sh);
    } else if ((average_right - average_left > LIGHT_THRESHOLD) && (pos_sh < LIMITE_SERVO_HORIZONTAL_SUP)) {
        pos_sh = pos_sh + 2;
        servo1.write(pos_sh);
    }
}
```

En el código anterior inicialmente tendremos que comprobar si el motor se debe mover en un sentido u en otro, para esto realizaremos la siguiente lógica:

1. Si la diferencia entre la media de luz superior e inferior es mayor al umbral de luz, el motor vertical se tendrá que mover en sentido positivo, siempre y cuando no se supere el límite superior.

2. Si la diferencia entre la media de luz inferior y superior es mayor al umbral de luz, el motor vertical se moverá en sentido negativo, siempre y cuando no se supere el límite inferior.
3. Si la diferencia entre la media de luz izquierda y derecha es mayor al umbral de luz, el motor horizontal se moverá en sentido negativo, siempre y cuando no se supere el límite izquierdo.
4. Si la diferencia entre la media de luz derecha e izquierda es mayor al umbral de luz, el motor horizontal se moverá en sentido positivo, siempre y cuando no se supere el límite derecho.

Para actualizar el valor de la posición de los servomotores se ha decidido moverlos en saltos de 2, de esta manera llegará antes al punto de máxima luz, aumentando su eficiencia energética sin perjudicar su posición ideal de luz que sí que se vería comprometida si los saltos fuesen mayores.

Para entender este concepto, supongamos que la luz ideal se encuentra en la posición 55 del servo, si realizásemos saltos de 10 en 10, podríamos ajustar el sistema a la posición 60 o 50, teniendo así un error con la posición ideal de 5. Sin embargo, si se realiza movimiento de 2 en 2, se podrá configurar el sistema en 56 o 54, obteniendo así un margen de error mucho menor.

5.12 Ajuste vertical β

Si en cambio ya se ha realizado el ajuste inicial se pasará al estado mostrado en la Figura 5-16

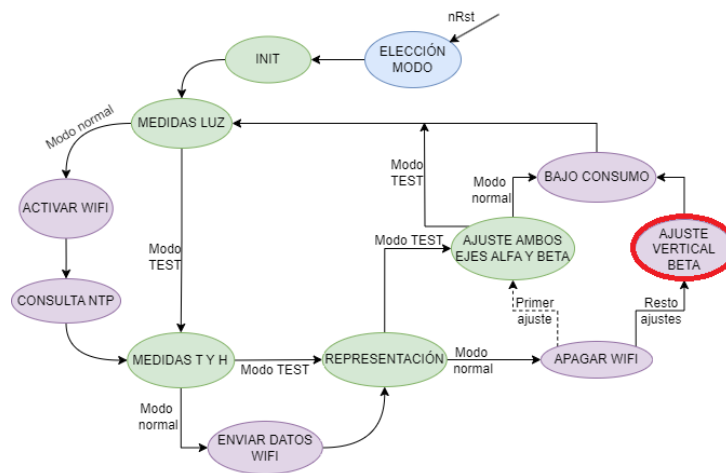


Figura 5-16: Estado para la ejecución del eje vertical

Se realizará el ajuste únicamente del eje vertical o β en el caso de que ya se haya realizado el ajuste inicial y siempre y cuando se encuentre en modo normal. Las funciones que desempeña son las mismas que en la Función para movimiento de ambos ejes aunque se realiza la comprobación del estado de funcionamiento del sistema y en caso de que se encuentre en modo TEST realizará el ajuste de ambos ejes de nuevo como se puede ver en el siguiente código, en este caso también se desactivará la representación del LCD mientras está en movimiento el sistema:

```

}else if(luzSuficiente() && !ajusteInicial){
  while (((abs(mediaTOP - mediaBOTTOM) > LIGHT_THRESHOLD) && (entreLimitesVerticales(pos_sv)))) {
    lcd.noDisplay();
    calcularMedias(mediaTOP, mediaBOTTOM, mediaIZQDA, mediaDCHA);
    if(modosTest){ // Si se encuentra en modo test realiza el ajuste en ambos ejes igualmente
      moverSeguidorSolar(mediaTOP, mediaBOTTOM, mediaIZQDA, mediaDCHA);
    }else{
      moverSeguidorSolar(mediaTOP, mediaBOTTOM, 0, 0);
    }
    delay(50); // Tiempo para que el motor se ajuste a la posición deseada
  }
}
}

```

Los pasos que realiza el sistema en este caso son los siguientes:

1. Verificación de cantidad de luz suficiente y que no sea el primer ajuste del día.
2. Mientras que la diferencia de medias de luz superior e inferior sea significativa y se encuentre entre los límites verticales se realizará el ajuste.
3. Desactivación de la presentación en el LCD mediante `lcd.noDisplay()`.
4. Se actualizan los valores medidos de luz.
5. Por último se verificará en qué estado de funcionamiento se encuentra el sistema. Si es en modo TEST realizará un ajuste de ambos ejes, mientras que si es en estado normal ajustará uno solo marcando que las medias izquierda y derecha sean 0, de esta manera el eje horizontal no se ajustará.

5.13 Bajo consumo

Una vez se han realizado los ajustes pertinentes se pasará al estado de bajo consumo mostrado en la Figura 5-17

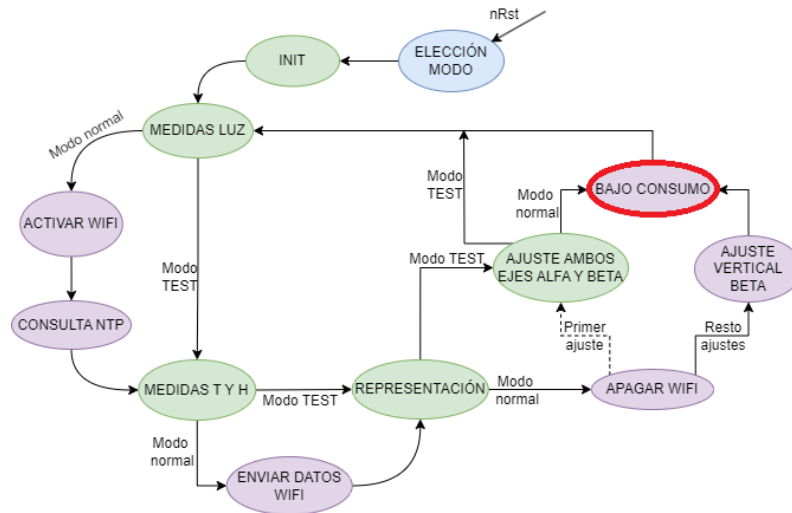


Figura 5-17: Estado de entrada al modo bajo consumo

El sistema estará sin realizar operaciones significativas la mayoría del tiempo, ya que al ser un seguidor solar, la cantidad de luz detectada por los fotorresistores variará cada bastante tiempo. Por tanto, es muy importante la introducción de un sistema de bajo consumo que permita que el sistema baje su consumo drásticamente para que así la autonomía del proyecto sea mayor.

Para entrar en modo bajo consumo se utilizará la librería “LowPower.h” [39], hay que destacar que esta función solo da la opción de dormir al Arduino durante 8 segundos, que es un tiempo bastante pequeño para lo que requiere este proyecto. Teniendo en cuenta que se quiere dormir al Arduino durante 5 minutos que son 300 segundos, se opta por un bucle for con 38 iteraciones de 8 segundos cada una, resultando así en 304 segundos, o lo que es lo mismo 5 minutos y 4 segundos de bajo consumo como se puede ver en a continuación:

```
for (int i = 0; i < 38; i++) { // Modo bajo consumo durante i_max*8 segundos = 304 segundos
  LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
}
```

Además como se puede ver en la entrada de bajo consumo, desactiva el ADC así como el BOD (Brownout Detector o detector baja tensión) para que el consumo de corriente en este modo sea el más bajo posible.

Por último y una vez el Arduino despierta se volverá al estado de Medidas de luz para que en la siguiente iteración dichos valores ya estén actualizados.

5.14 Especificaciones extra

5.14.1 Evitar demanda de corriente excesiva

El sistema utiliza tanto servomotores como un módulo WiFi, ambos consumen una gran cantidad de corriente si están en funcionamiento. Por lo que para evitar demandar una corriente excesiva al sistema, se ha diseñado el software de tal manera que servomotores y WiFi nunca trabajen simultáneamente.

Por eso nada más arrancar el sistema arrancamos el módulo WiFi y no es hasta que este se ha apagado por completo que se realizarán las comprobaciones para ejecutar el movimiento de los servomotores.

5.14.2 Evitar interferencia en la comunicación serial

Las interferencias en la comunicación serial se dan cuando controlamos el bipolar desde el pin 12 del Arduino UNO y los servomotores que están conectados al pin 5 y 6, provocando así que un encendido del módulo WiFi mueva los motores debido a las interferencias y viceversa. Esto se debe al ruido eléctrico generado por los motores debido a la forma de alimentación del sensor shield y que se ha comentado en el apartado de Arduino UNO y Sensor Shield.

El surgimiento de las interferencias en la comunicación serial suele darse por básicamente tres razones en nuestro caso:

1. Picos de corriente proporcionados a los motores que pueden provocar fluctuaciones en la línea de alimentación y tierra y provocar el encendido del módulo WiFi.
2. Picos de corriente al conmutar el transistor bipolar, además esto es un cambio brusco de corriente que puede inducir ruido eléctrico en el sistema.
3. Proximidad entre los pines y poca calidad del sensor shield de Arduino.

Para evitar movimientos innecesarios de los motores lo que haremos será desconectar los motores mediante la sentencia detach() de la librería “Servo.h” [40], esto permite que los

servomotores no reciban ninguna señal PWM hasta que se vuelvan a configurar evitando así que se muevan. Una vez se han desconfigurado se podrá encender el módulo WiFi sin problema tal y como se muestra en el siguiente código:

```
servo1.detach();    // Desactivamos para evitar interferencias en el ESPSerial
servo2.detach();

// ARRANCAR MODULO WIFI INICIALMENTE
digitalWrite(base_transistor, HIGH);    // Aplicar voltaje a la base
```

En este caso no es necesario controlar un encendido del módulo WiFi involuntario, ya que este módulo consume bastante corriente, y la interferencia provocada en la comunicación serial no es suficiente para provocar un encendido.

5.14.3 Asegurar obtención del día actual

La obtención y procesamiento correcto de la respuesta del servidor NTP es esencial para el programa ya que dependiendo de ello se realizará el ajuste en un eje o en ambos. Para asegurarse de que inicialmente el sistema obtenga un día válido del servidor NTP se utilizará el código mostrado a continuación:

```
// Espera activa de respuesta del NTP
while(anio == 1970){
    processResponse();
    delay(50);
}
```

Esta página se ha dejado en blanco intencionadamente

6 Aplicación software por bloques en TinkerCad

Durante el desarrollo de este proyecto se han tenido que realizar diferentes pruebas con lo que respecta al código y diseño de este para obtener el funcionamiento deseado. Para realizar dichas pruebas de una manera más eficiente, cómoda y protegida se ha decidido utilizar TinkerCad [28] que es una plataforma en línea que permite la codificación, tanto en bloques como en texto, de diferentes montajes en el entorno de Arduino.

En este caso se ha decidido optar por la codificación en bloques, ya que esto permite mostrar el pseudocódigo del sistema desarrollado de una manera sencilla y comprensiva, proporcionando así un entendimiento para un enfoque académico.

Un ejemplo sencillo de funcionamiento y herramientas que proporciona TinkerCad es el mostrado en la Figura 6-1, en el que se realizará un movimiento del motor en función de únicamente dos fotorresistores.

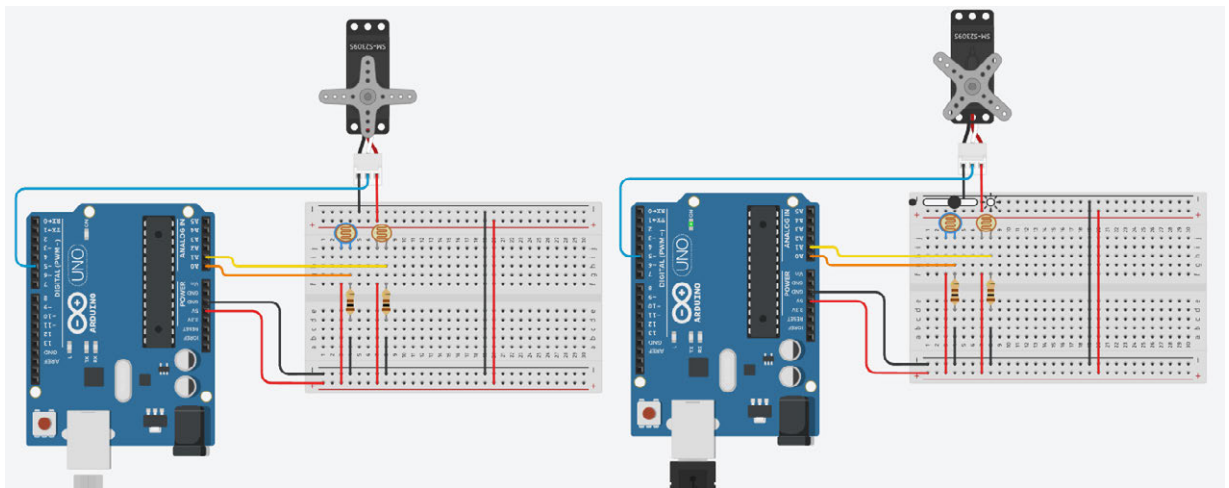


Figura 6-1: Ejemplo movimiento motor

Esta herramienta online permite actuar sobre los fotorresistores para indicar la luz que incide sobre ellos y que el motor se mueva en función de estos valores. Se ha realizado un código de bloques sencillo para simular el movimiento utilizando las herramientas aportadas por la plataforma, tal y como se muestra en la Figura 6-2.

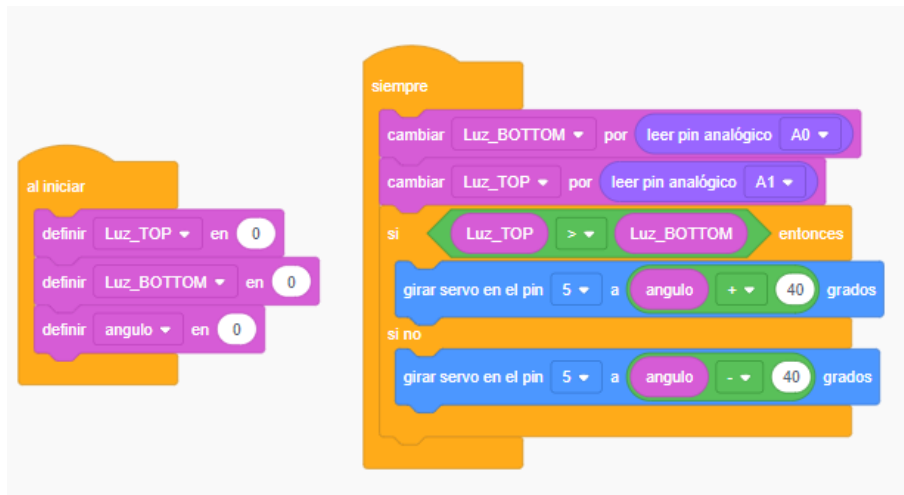


Figura 6-2: Código de bloques de ejemplo

La codificación por bloques tiene ciertas restricciones, ya que no se puede configurar al sistema con tanta precisión y profundidad como si se utilizase un código en texto. El código por bloques mostrado en los apartados siguientes está editado y Tinkercad no permite su ejecución debido a la limitación de bucles que tiene, pese a esto, es la mejor herramienta para proporcionar una visión general del funcionamiento del sistema más sencilla.

Cabe destacar que Tinkercad es una plataforma limitada también en lo que respecta a sensores y diferentes elementos hardware, por lo que el diseño realizado no contiene algunos elementos clave utilizados en el proyecto final, como puede ser el sensor BME680, el LCD o el TP4056 que no se han podido simular. Tampoco se encuentran baterías de ion-litio recargables que han sido simuladas con pilas AA de 1.5V cada una. Sin embargo, esto no es un problema para el diseño que se ha realizado ya que se busca un diseño de código por bloques sencillo y fácil de entender para un enfoque académico.

Por tanto, el diseño hardware realizado ha sido el mostrado en la Figura 6-3.

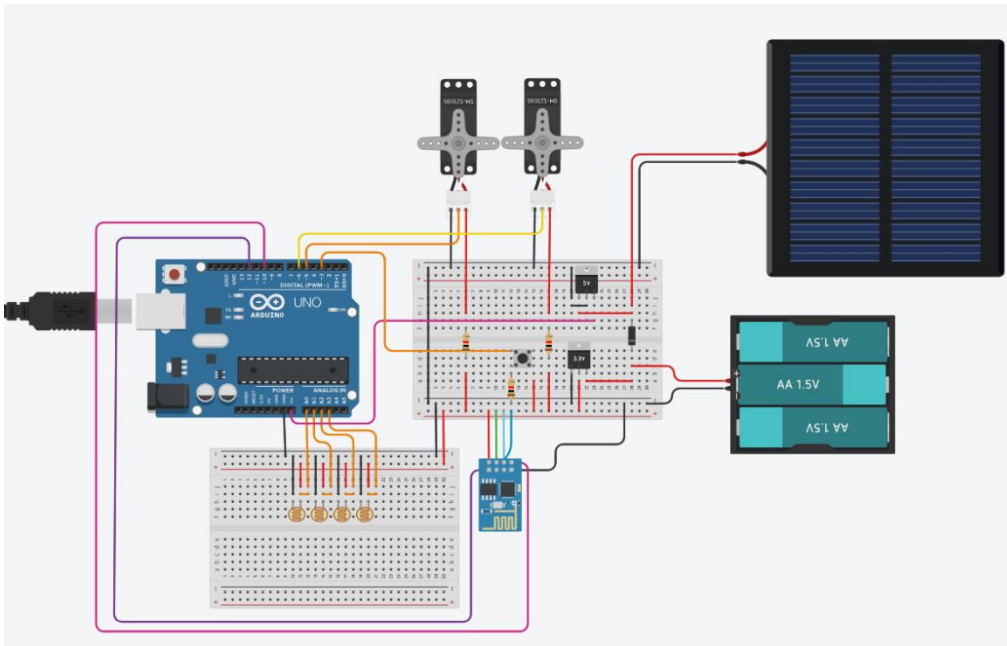


Figura 6-3: Esquema analógico diseñado en TinkerCad

6.1 Elección de modo en código de bloques

En el código de bloques mostrado en la Figura 6-4 se puede ver el código de bloques diseñado para la elección del modo de funcionamiento del sistema y que representa el estado marcado en la Figura 6-5 que como se ha comentado anteriormente detecta una pulsación en el pin del micropulsador.



Figura 6-4: Elección de modo programado en bloques

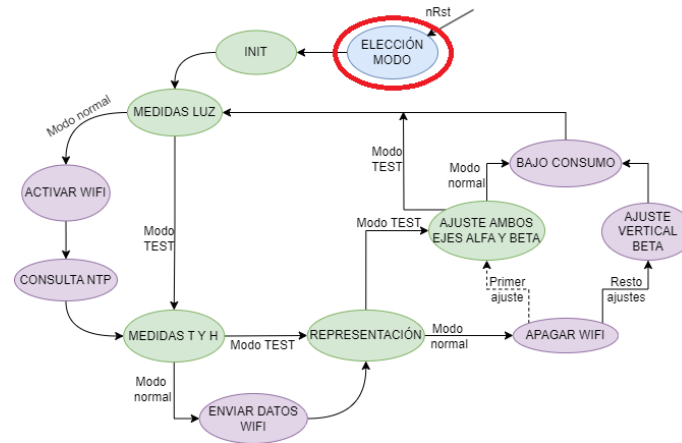


Figura 6-5: Estado de elección de modo

6.2 Estado Init en código de bloques

En el estado mostrado en la Figura 6-6 el sistema inicializa todas las variables a utilizar tal y como se muestra en la Figura 6-7, también se realiza la inicialización del LCD para mostrar el modo de funcionamiento elegido. Es un código sumamente parecido al mostrado en el apartado 5.2.

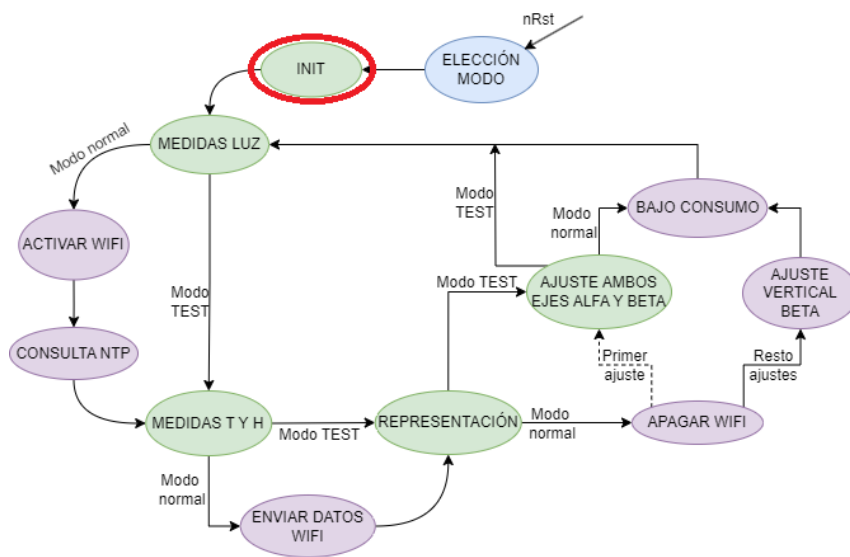


Figura 6-6: Estado de inicialización del sistema

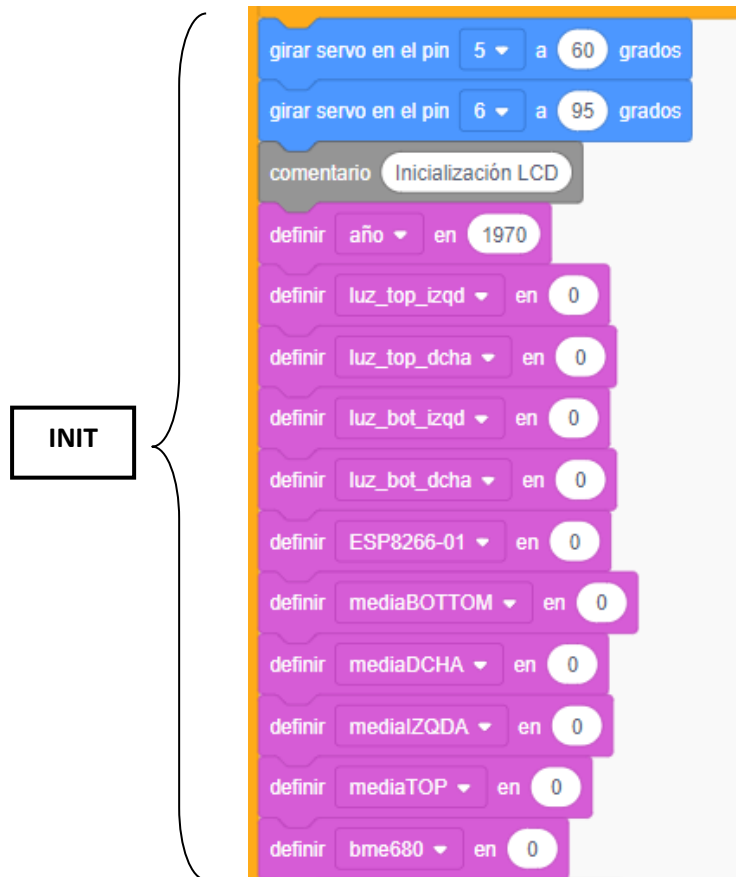


Figura 6-7: Inicialización de variables con código de bloques

6.3 Cálculo de medias de luz con bloques

En este apartado se mostrará el código de bloques para el estado marcado en la Figura 6-8.

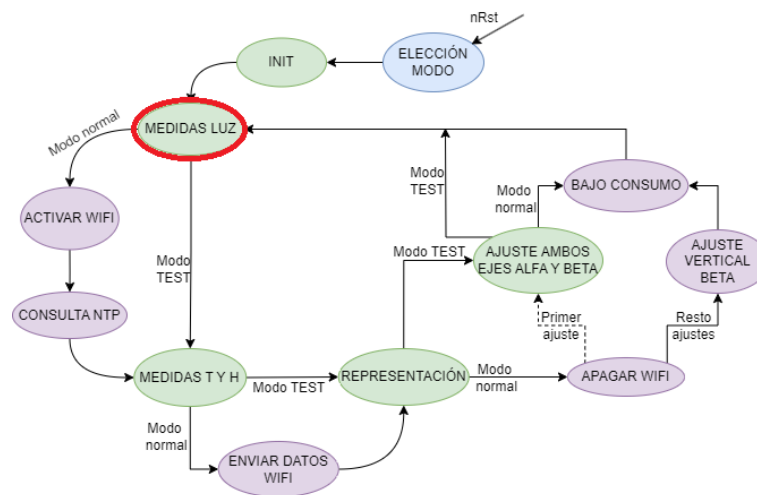


Figura 6-8: Estado para la realización de medidas de luz

En la Figura 6-9 se observa el código de bloques que realiza el cálculo de las medias de luz de cada uno de los ejes. Una vez realizadas 5 medidas de los pines analógicos se calcula la media entre ellas para posteriormente realizar la media de cada uno de los ejes.

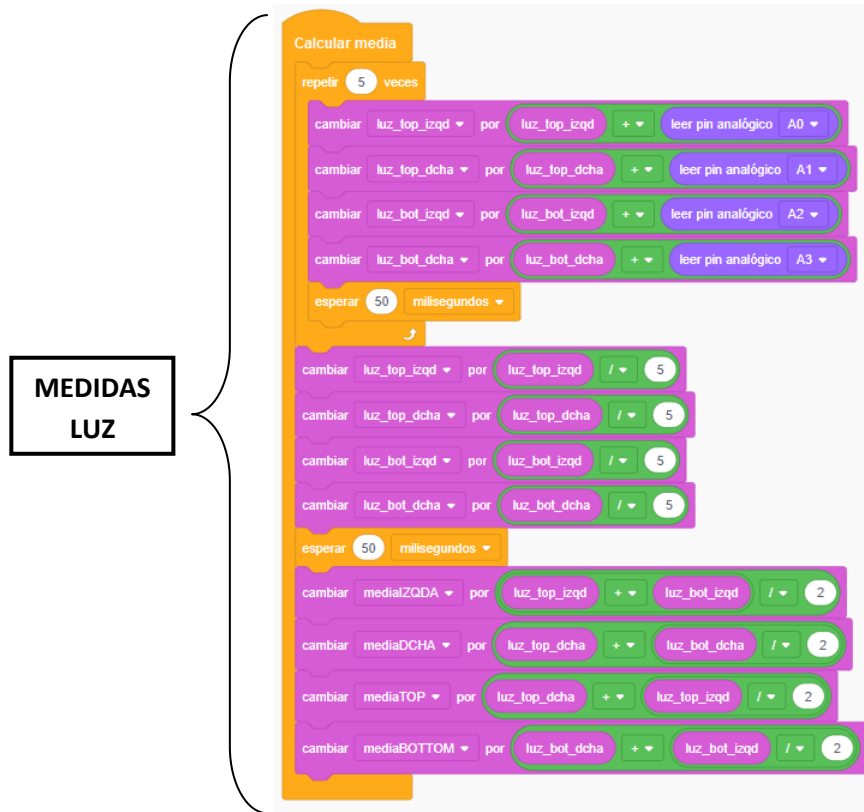


Figura 6-9: Calculo de medias de luz

6.4 Activación WiFi y consulta NTP

En este punto se tratarán los estados marcados en la Figura 6-10 que se realizarán únicamente cuando el sistema se encuentre en modo normal.

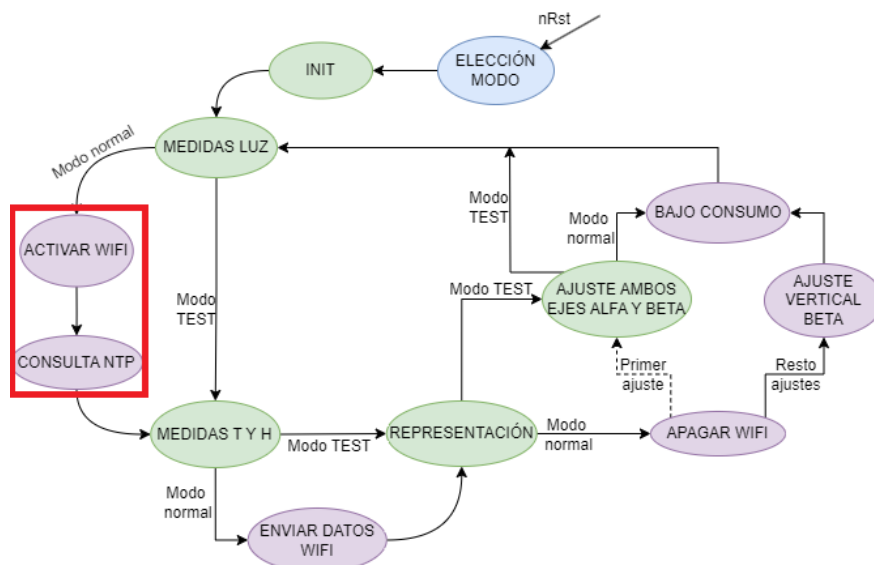


Figura 6-10: Estados para activación WiFi y consulta NTP

En la Figura 6-11 se realiza la comprobación del modo de funcionamiento del sistema y en caso de que se encuentre en modo normal se activará el módulo WiFi y se realizará la consulta al servidor NTP.

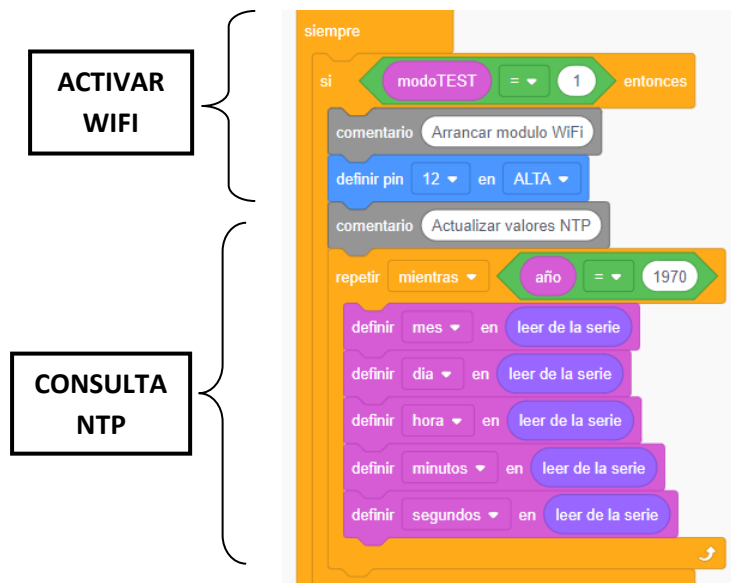


Figura 6-11: Activación módulo WiFi, consulta NTP y medidas

6.5 Medidas, envío de datos WiFi y representación

En este apartado se tratarán todos los estados marcados en la Figura 6-12 ya que en la programación por bloques y debido a las limitaciones de TinkerCad son muy sencillos.

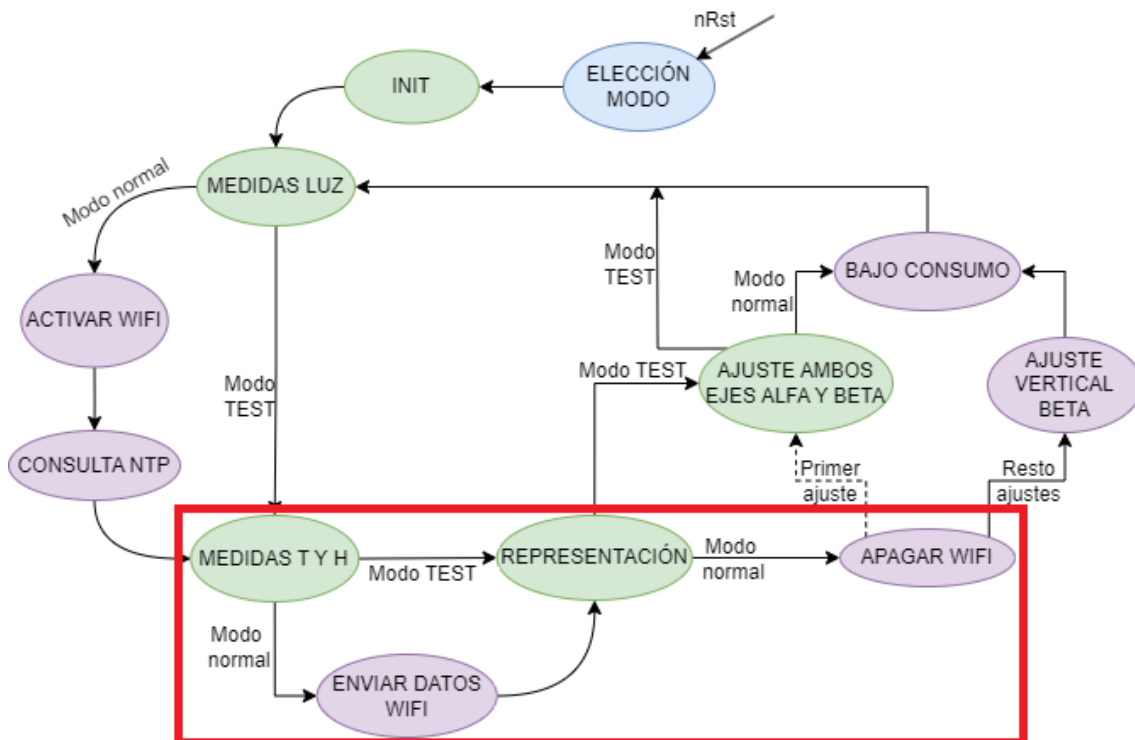


Figura 6-12: Estados de medida, envío de datos, representación y apagado de WiFi

En el código de bloques mostrado en la Figura 6-13 se realizan las medidas de temperatura y humedad por parte del sensor BME680 y se enviarán al módulo WiFi en caso de que se encuentre en modo normal para que estas se representen en ThingSpeak. Por último, se

imprimirán los valores en el LCD independientemente del modo que se esté ejecutando y se apagará el módulo WiFi para evitar el consumo de corriente.



Figura 6-13: Realización de medidas, envío y representación

6.6 Lógica de ajuste y modo bajo consumo

Una vez se ha desactivado el módulo WiFi se realizará el ajuste de los servomotores según corresponda en función de si es el primer ajuste o si la ejecución es en modo TEST o no. Además, se inducirá al sistema en modo bajo consumo si es un ajuste en modo normal, reflejando los estados marcados en la Figura 6-14:

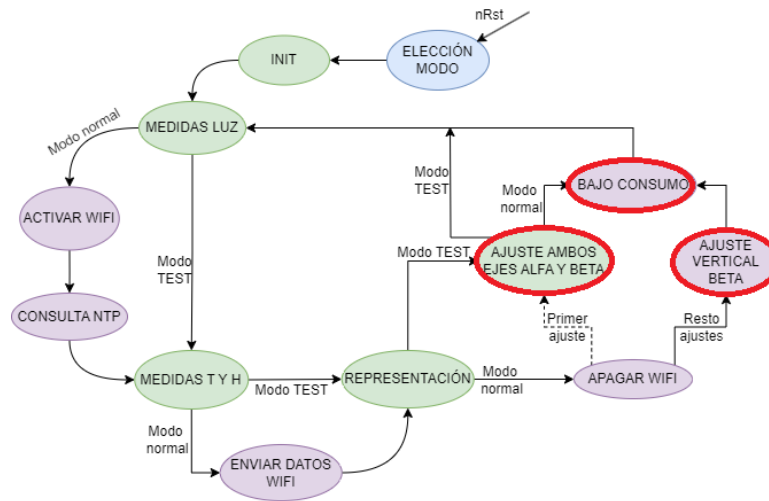


Figura 6-14: Estado para el ajuste de ambos ejes

Se seguirá la lógica marcada en los bloques de la Figura 6-15, en este caso se realizan las comprobaciones de que haya luz suficiente y de sí se tiene que realizar un ajuste inicial o no. Posteriormente se realiza la comprobación para verificar en qué sentido se deben realizar los movimientos de los motores y se realiza la llamada a la función de ajuste vertical y horizontal,

realizando medidas de luz para actualizar estos valores correctamente. Una vez se haya hecho el ajuste se marcará el ajuste inicial como realizado.

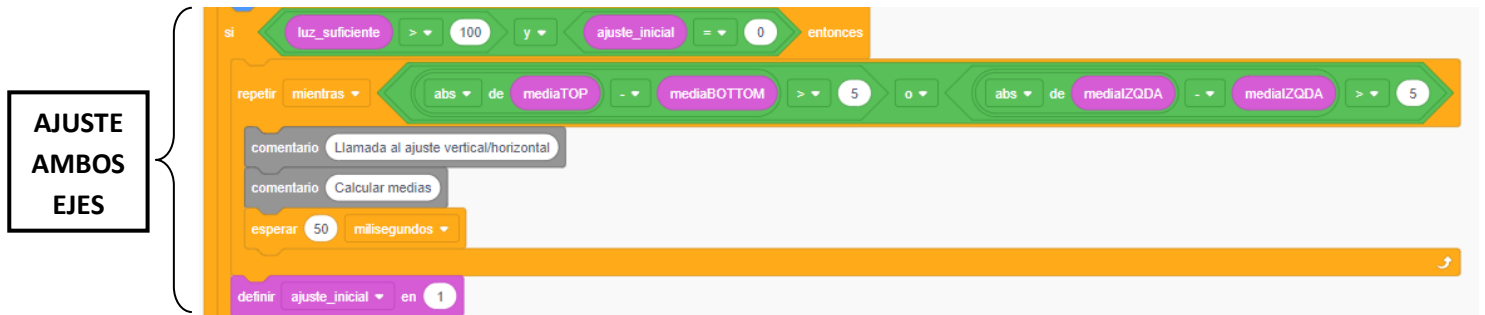


Figura 6-15: Lógica de ajuste seguida para el ajuste de ambos servomotores

Por otra parte, en la Figura 6-16 se puede ver que el ajuste se realiza únicamente en un solo eje gracias a la variable ajuste inicial. En caso de que se encuentre en modo TEST se realizará el ajuste de ambos ejes independientemente del valor de la variable. Además se actualizarán los valores de luz cada 50ms para realizar un ajuste exacto en todo momento.

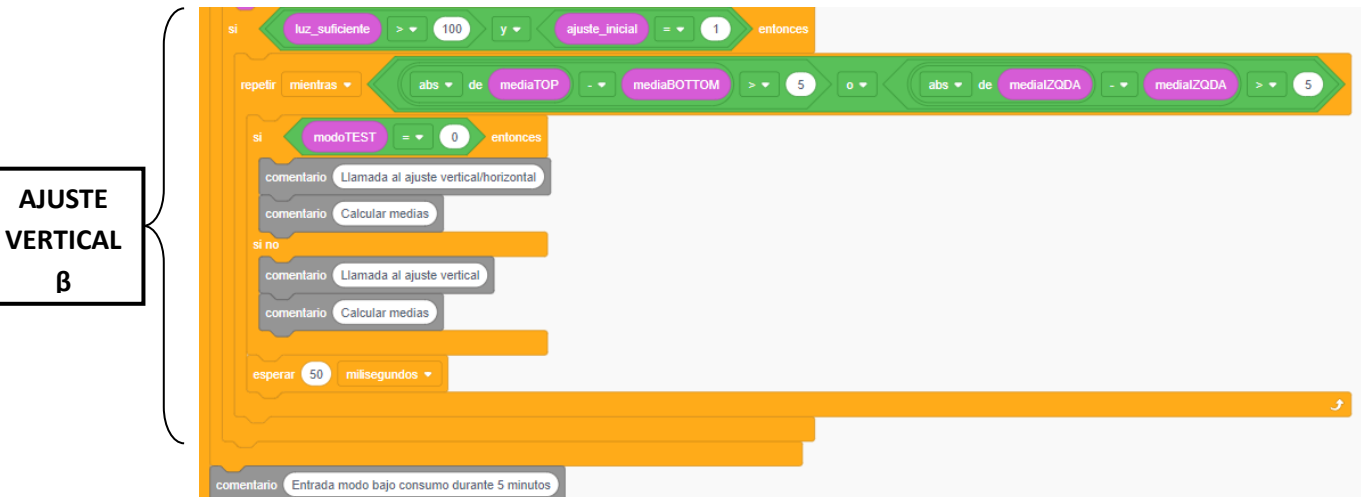


Figura 6-16: Lógica seguida para el ajuste un solo eje

Por último se realizará la entrada en el modo bajo consumo por 5 minutos, durante los cuales el sistema consumirá menos y permitirá la carga de la batería de una manera más eficiente.

6.7 Ajuste vertical/horizontal con bloques

En la Figura 6-17 se observa el código de bloques con la lógica seguida para realizar el ajuste tanto vertical como horizontal en los servomotores, siempre y cuando la diferencia de luz sea significativa y no se sobrepasen los límites de los servos.

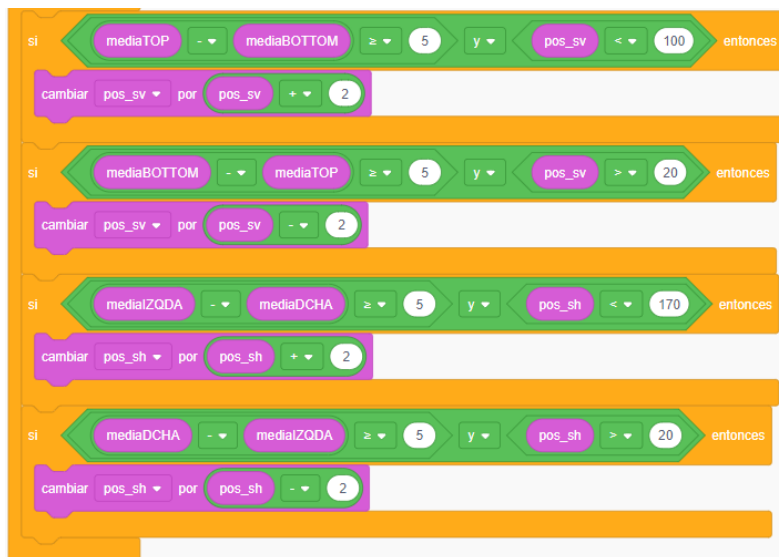


Figura 6-17: Ajuste vertical y horizontal de los servomotores

7 RESULTADOS

Una vez se han detallado las funciones del proyecto se pretende mostrar los resultados obtenidos con diferentes ejemplos. Debido a que el proyecto realiza numerosas acciones entre las que se encuentran la realización de medidas, el movimiento de motores, la conexión WiFi... La demostración del correcto funcionamiento de estas se va a dividir en diferentes puntos.

7.1 Seguimiento de luz

La parte fundamental de este trabajo se basa en el seguimiento solar o de luz por parte de los servomotores y de los LDRs controlados a partir del Arduino UNO. Para mostrar el correcto funcionamiento de este sistema, se utilizará una linterna la cual se enfocará a los LDRs para que siga el movimiento de esta. Inicialmente se situará la luz justo encima del montaje, haciendo así que el módulo fotovoltaico se oriente como se muestra en la Figura 7-1

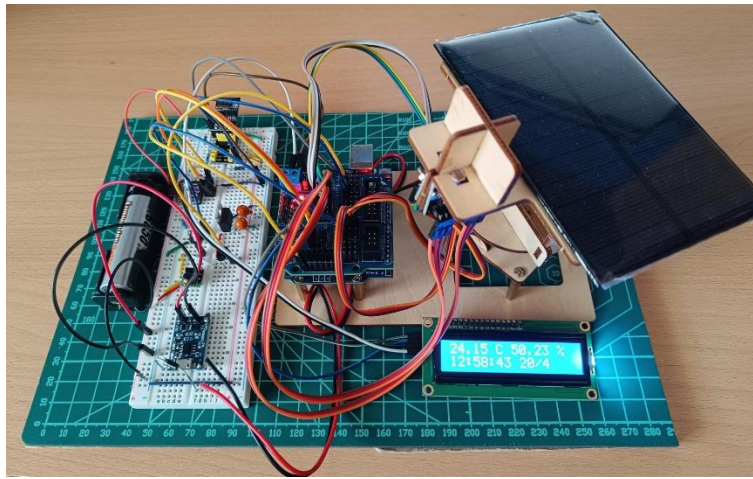


Figura 7-1: Orientación panel solar con luz en su perpendicular

Para demostrar que realmente se está ajustando según la luz recibida, si se enfoca la luz desde un extremo del sistema se puede observar como la orientación del panel cambia como en la Figura 7-2

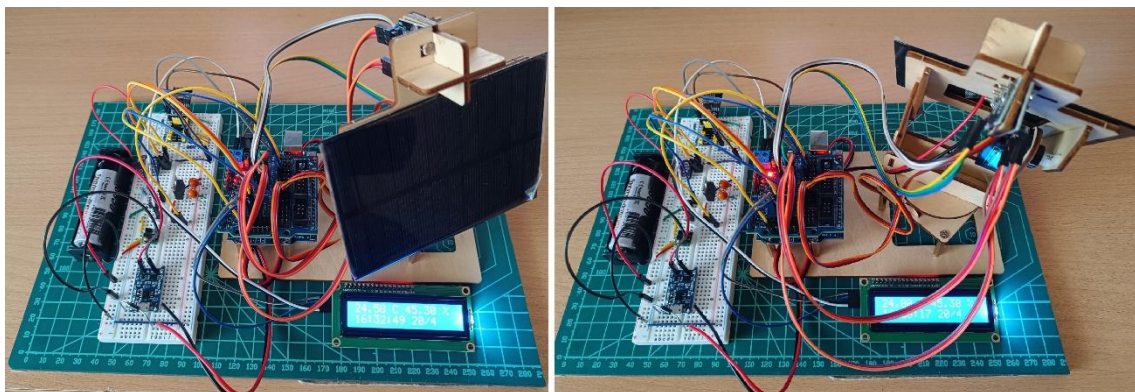


Figura 7-2: Orientación del panel solar desde los extremos

Para que el resultado sea el obtenido es importante que la batería esté cargada y tenga la suficiente capacidad para que se pueda extraer la corriente necesaria para que el movimiento de los motores sea lo más efectivo posible.

7.2 Realización de medidas y representación

Otra parte fundamental del proyecto se basa en las medidas de temperatura y humedad que realiza el BME680 así como su representación en el LCD y en la web ThingSpeak.

Para poder visualizar estas medidas y comprobar su correcto funcionamiento se dispone del LCD, el cuál cuando se encuentre en modo normal de funcionamiento mostrará un timestamp con la hora exacta en la que se tomó la medida. Esta misma medida mostrada en el LCD se podrá visualizar en la web de ThingSpeak con un pequeño delay, de esta manera se comprobará que las medidas son correctas y que la comunicación establecida con el servidor ha sido exitosa.

Como demostración del correcto funcionamiento se ha dejado ejecutar el sistema durante 10 minutos, configurándolo para que realice medidas y ajustes cada minuto y medio, dando como resultado lo observado en la Figura 7-3 y Figura 7-4:



Figura 7-3: Medidas mostrada en el LCD, actualizada cada 90 segundos

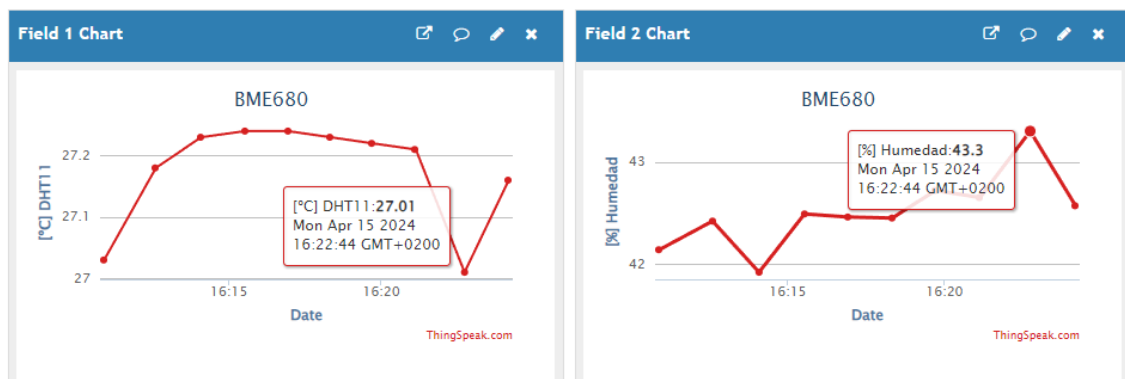


Figura 7-4: Medidas vistas desde ThingSpeak

Como se ha podido observar coinciden los valores mostrados en el LCD y los de ThingSpeak, con una diferencia de 5 segundos entre unos y otros debido a lo que tarda el servidor en procesar los datos mandados.

7.3 Entrada bajo consumo

El modo bajo consumo es importante para el sistema ya que permitirá que la autonomía aumente y permitirá que la batería se cargue a una mayor velocidad. La comprobación de la entrada al modo bajo consumo se puede probar una vez más con la linterna, en este caso cuando el sistema entre en ese modo de bajo consumo y se apunte la linterna a las LDRs para que se vuelvan a ajustar los servomotores no realizará nada.

Sin embargo, la mejor manera de comprobar la entrada a este modo es realizando una medida de corriente, en este caso se ha realizado utilizando un multímetro de mano, dando como resultado lo mostrado en la Figura 7-5, donde se puede ver que el consumo se reduce de 135mA a 41mA una vez se entra al modo de bajo consumo.

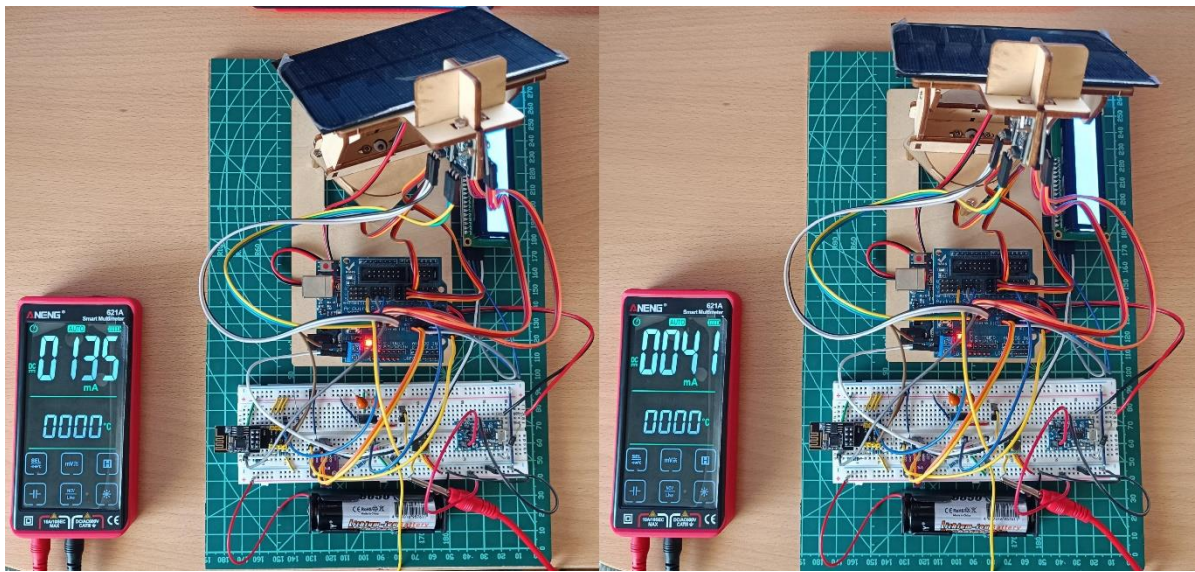


Figura 7-5: Demostración entrada bajo consumo

7.4 Control de carga de la batería

A través del módulo TP4056 se controla la carga y descarga de la batería como se ha comentado en el apartado Módulo carga TP4056. Cuando la batería tenga menos de 4.2V de tensión el TP4056 alimentará la batería cargándola y encendiendo el led rojo como se muestra en la Figura 7-6.

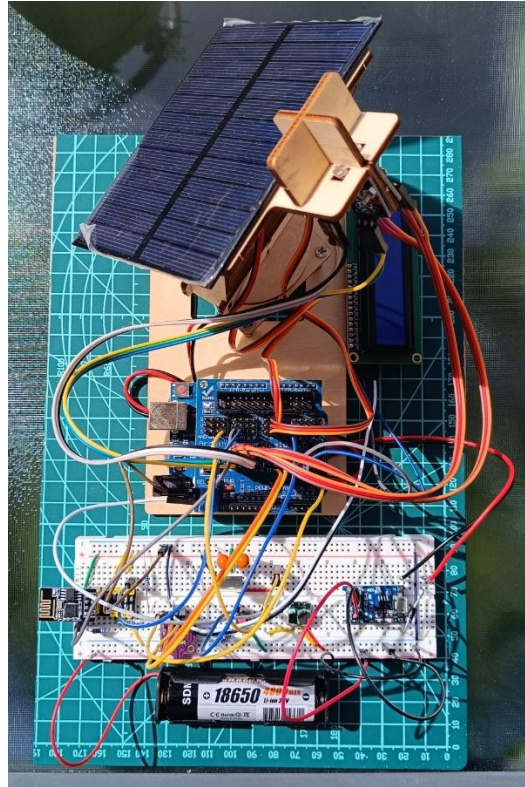


Figura 7-6: Ejemplo control de carga

Sin embargo, si la batería ya está completamente cargada, el TP4056 cortará la corriente de los pines B+ y B- y no dejará que se cargue protegiéndola así frente a sobrecargas.

7.5 Conclusión

Todas las pruebas comentadas anteriormente se han realizado en diferentes entornos, con luz directa del sol y sin ella para ver como afectaba a su rendimiento. Además se han realizado pruebas durante varios días consecutivos desde el diseño dónde los únicos problemas observados han sido debido a desconexiones o malas conexiones del cableado del sistema que se han solventado inmediatamente.

8 PRESUPUESTO

El objetivo principal del presupuesto de un proyecto fin de grado es planificar y controlar el uso de los recursos financieros disponibles para asegurar que el proyecto se complete dentro de los límites presupuestarios establecidos.

En el caso específico de este proyecto, se ha optado por utilizar exclusivamente herramientas de software de código abierto, como puede ser ArduinoIDE o TinkerCad, lo que significa que no ha sido necesario incurrir en gastos relacionados con la adquisición de licencias de software.

Sin embargo, en cuanto al hardware necesario para llevar a cabo el proyecto, se han identificado ciertos materiales que han requerido una inversión financiera entre los que se encuentran:

1. Kit de Arduino que incluye mini panel solar de 6V y 200mA, 4 módulos de LDRs, 2 servomotores de 9g, un Arduino UNO R3 y una placa sensor shield v5.0 por 21.06€.
2. Kit solar que incluye, batería de 600mAh, cargador de batería TP4056 y un elevador de tensión por 18.06€.
3. Batería ión-litio 18650 recargable de 3.7V y 4800mAh por 6.50€.
4. ESP8266-01 con adaptador para protoboard por 6.49€.
5. LCD 1602 por 3.48€.
6. Regulador 3.3V LM3940 por 2.30€.
7. Diodo Schottky 1N5817 por 0.11€.
8. Transistor 2N3904 por 0.05€.

El resto de los materiales, entre los que se encuentran los cables, placas de prototipado, sensores, condensadores y resistencias han sido obtenidos previamente en otras asignaturas, por lo que no se tendrán en cuenta para el presupuesto. Además, a lo largo del proyecto se han utilizado los diferentes materiales del laboratorio para la realización de pruebas y verificación del correcto funcionamiento del sistema, entre las que se encuentran los multímetros, fuentes de alimentación y placas de prototipado.

Un desglose del presupuesto se puede ver en la Tabla 13:

PRESUPUESTO FINAL		
MATERIAL	PRECIO	Tienda
Kit Arduino	21,06 €	Turibot
Kit Solar	18,06 €	Turibot
Batería ion-litio	6,50 €	Turibot
ESP8266-01 y adaptador	6,49 €	Amazon
LCD 1602	3,48 €	Turibot
LM3940	2,30 €	Conectrol

1N5817	0,11 €	Conectrol
2N3904	0,05 €	Conectrol
TOTAL	58,05 €	

Tabla 13: Materiales adquiridos y precios

Comparando este presupuesto con el estimado en el anteproyecto, el cual se puede observar en la Tabla 14, hay un aumento en el precio, debido a que en un principio no se tuvo en cuenta la realización de medidas de temperatura/humedad, ni por tanto su representación en el LCD y en la web de ThingSpeak gracias al ESP8266-01.

PRESUPUESTO INICIAL

MATERIAL	PRECIO	Tienda
Kit Arduino	21,06 €	Turibot
Kit Solar	18,06 €	Turibot

Tabla 14: Presupuesto inicial

Por tanto, el aumento en el presupuesto ha sido de 18.93€ lo que es un incremento aceptable para la cantidad de módulos distintos que se han introducido.

9 IMPACTO DEL PROYECTO

El desarrollo del presente proyecto tiene implicaciones significativas en diversos ámbitos, desde el tecnológico hasta el medioambiental y el social, alineándose con varios ODS (Objetivos de Desarrollo Sostenible) [41] establecidos por las Naciones Unidas como se puede observar en la Figura 9-1.



Figura 9-1: Objetivos de desarrollo sostenible [41]

A continuación, se detallan las principales áreas de impacto:

1. Implicaciones Tecnológicas:

El diseño e implementación de un seguidor solar controlado por servomotores, junto con la integración de sensores de luz, temperatura y humedad, representan un avance tecnológico significativo en el ámbito de las energías renovables y la automatización. Este proyecto no solo se limita a la captación pasiva de energía solar a través de paneles fotovoltaicos, sino que busca optimizar activamente la eficiencia de esta captación mediante un sistema dinámico de seguimiento solar. Esta innovación contribuye directamente al Objetivo de Desarrollo Sostenible (ODS) 7 (Energía asequible y no contaminante), siendo este el principal objetivo que aborda el proyecto. Al promover el uso de energía solar como fuente renovable y limpia, se impulsa la transición hacia sistemas energéticos más sostenibles y respetuosos con el medio ambiente.

Además, al mejorar la eficiencia en la captación de energía solar, el proyecto se alinea estrechamente con el ODS 9 (Industria, innovación e infraestructura). La implementación de un seguidor solar no solo implica el desarrollo de tecnologías avanzadas para el control de servomotores y la integración de sensores, sino también la optimización de sistemas de generación y gestión de energía. Al aumentar la eficiencia energética de un sistema alimentado por placas fotovoltaicas, se contribuye a la expansión de infraestructuras innovadoras y sostenibles en el ámbito energético, fomentando así el desarrollo industrial y tecnológico.

Por último, este proyecto también tiene un impacto positivo en el ODS 13 (Acción por el clima). Al abordar la eficiencia energética y la mitigación del cambio climático, se promueve

activamente la reducción de las emisiones de gases de efecto invernadero y la adopción de prácticas energéticas más sostenibles. La preferencia por el uso de una energía limpia y renovable frente a otras fuentes más contaminantes contribuye a la lucha global contra el cambio climático.

2. Impacto Ambiental:

El enfoque de este proyecto hacia el aprovechamiento de la energía solar no solo representa un avance tecnológico como el comentado en el apartado anterior, sino que también tiene un impacto significativo en la disminución de los efectos del cambio climático y la preservación del medio ambiente. La dependencia global de fuentes de energía no renovables, como el petróleo, el gas y el carbón, ha llevado a la emisión masiva de gases de efecto invernadero, contribuyendo al calentamiento global y sus consecuencias devastadoras.

Al centrarse en el aprovechamiento de una fuente de energía abundante y limpia como es el sol, este proyecto contribuye directamente a la reducción de la dependencia de combustibles fósiles y, por tanto, a la disminución de las emisiones de gases de efecto invernadero. Sin embargo, una de las grandes desventajas de la energía solar es su poca eficiencia, por eso, el desarrollo de un seguidor solar solventa en parte la mayor desventaja de este método de energía. Las emisiones de gases de efecto invernadero son una de las principales causas del calentamiento global, que incluyen fenómenos climáticos extremos, aumento del nivel del mar y pérdida de biodiversidad.

Para demostrar que realmente la energía fotovoltaica es una energía limpia se puede ver en la gráfica mostrada en la Figura 9-2 que las emisiones de efecto invernadero por parte de la energía solar son prácticamente nulas en comparación con otro tipo de energías no renovables.

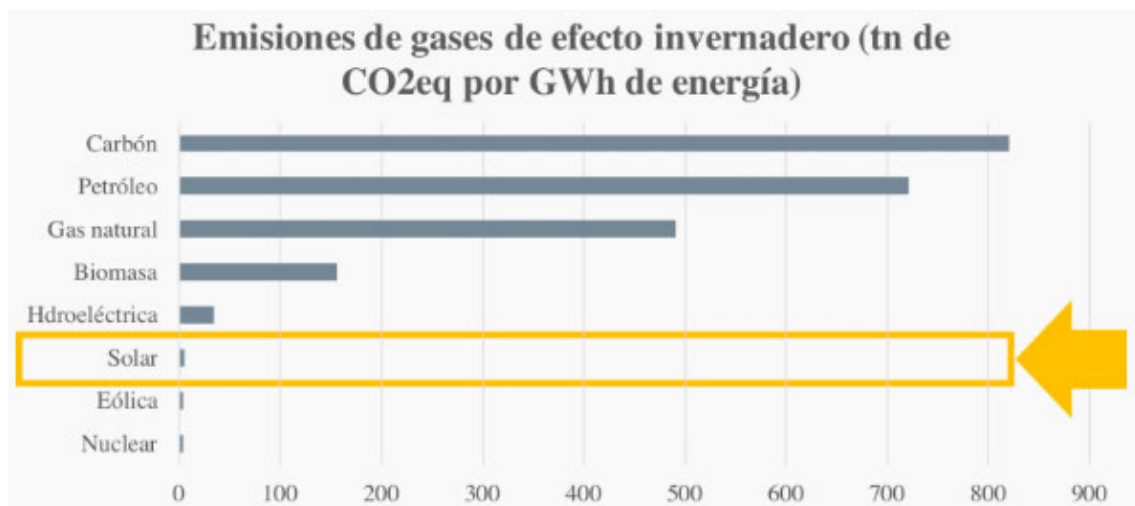


Figura 9-2: Emisiones de CO₂ de diferentes energías

3. Implicaciones Sociales:

El acceso a energía limpia y sostenible es un factor crucial para el desarrollo y el bienestar de las comunidades, especialmente en áreas rurales o remotas que carecen de infraestructura eléctrica adecuada, un ejemplo es el mostrado en la Figura 9-3.



Figura 9-3: Panel solar en zonas rurales [42]

La implementación de sistemas de seguimiento solar, como el desarrollado en este proyecto, ofrece una solución viable para abordar esta brecha energética. En áreas donde el suministro eléctrico convencional es limitado o inexistente, el seguimiento solar puede ser una alternativa rentable y sostenible para satisfacer las necesidades básicas de energía.

Además de proporcionar alimentación para dispositivos electrónicos, como teléfonos móviles y radios, el acceso a la energía solar también puede tener un impacto transformador en otros aspectos de la vida comunitaria. Por ejemplo, en una mayor escala, este proyecto puede permitir el funcionamiento de centros de salud con equipos médicos, facilitar la educación mediante el uso de ordenadores y acceso a Internet, y apoyar actividades económicas como la agricultura y la producción de alimentos.

Esta página se ha dejado en blanco intencionadamente

10 CONCLUSIONES, MEJORAS Y TRABAJOS FUTUROS

En este apartado se tratarán diferentes aspectos del proyecto, entre los que se encuentran unas conclusiones de funcionamiento así como posibles mejoras y trabajos futuros para la optimización del proyecto.

10.1 Conclusiones

1. Desarrollo y diseño de un seguidor solar:

Se ha logrado diseñar e implementar un sistema de seguimiento solar completamente funcional que utiliza servomotores para realizar los ajustes pertinentes de la manera más eficiente posible, siguiendo la posición de sol a lo largo del día y estaciones.

2. Implementación de sensor y representación:

Se ha conseguido la realización de lecturas de luz mediante módulos LDRs para orientar de manera precisa la placa solar. Además, se ha integrado correctamente el sensor BME680 para la realización de medidas de temperatura y humedad, las cuales se envían mediante el ESP8266-01 a la plataforma web y también se muestran en el LCD de una manera simple e intuitiva.

3. Conexión WiFi local:

Se ha logrado conectar al sistema a un WiFi predeterminado, consiguiendo realizar consultas a un servidor NTP y mandando los datos medidos por el sensor BME680 a la web de ThingSpeak.

4. Control de carga/descarga batería:

Por último, se ha conseguido controlar y proteger la carga/descarga de la batería utilizada mediante el TP4056, proporcionando así al proyecto una seguridad y eficiencia muy alta.

En resumen, el proyecto ha alcanzado con éxito sus objetivos principales y ha demostrado el potencial de las tecnologías renovables y la automatización para abordar desafíos energéticos y ambientales. Sin embargo, se reconocen áreas de mejora y oportunidades para futuras investigaciones que se tratarán en el punto siguiente.

10.2 Mejoras y trabajos futuros

Aunque los objetivos marcados inicialmente se han cumplido con el desarrollo de este proyecto, todavía queda margen de mejora y optimización para proyectos futuros. Entre las principales mejoras destacan:

1. Diseño y fabricación de PCB:

Aunque una parte del cableado existente en el proyecto es inevitable, debido a que las conexiones con los motores y LDRs deben ser externas a una protoboard por la estructura de

movimiento utilizada, hay una buena parte de las conexiones que se podrían optimizar mediante una PCB.

La mayoría de los errores y funcionalidades incorrectas observadas, como pueden ser una mala representación en el LCD, o una conexión tardía al WiFi se han debido a la desconexión de cables, todo esto podría arreglarse fácilmente en una PCB. Esto mejoraría la fiabilidad y la eficiencia del sistema, reduciría el espacio ocupado y simplificaría el proceso de montaje

2. Integración de sensores adicionales:

Actualmente en el proyecto únicamente se realizan medidas de temperatura y humedad, para poder representarlas tanto en el LCD como en ThingSpeak. Sin embargo, se podrían añadir numerosos sensores hasta crear una estación meteorológica portátil y autónoma.

Entre los sensores que se podrían utilizar están sensores de calidad del aire, radiación UV vibraciones, distancia... De esta manera el proyecto podría evolucionar a diferentes objetivos como pueden ser detección de incendios en zonas aisladas o incluso detectores de terremotos mediante otro tipo de sensores. Todo esto de una manera autónoma gracias a la recarga de las baterías mediante la radiación solar.

3. Creación de aplicación web:

En este proyecto se utiliza la web de ThingSpeak para poder visualizar los diferentes datos obtenidos. Sin embargo, se podría realizar el diseño de una web o aplicación móvil que permita acceder y controlar el sistema de manera remota y desde cualquier dispositivo con conexión a internet. Por ejemplo, se podría implementar un botón en la web que dispare medidas de los sensores, otro que reinicie el sistema o incluso uno que permita ver el estado de la batería, en este ámbito las posibilidades son casi infinitas.

Ligado al diseño web, la adición de una base de datos en la que almacenar las diferentes medidas para analizar comportamientos y patrones que se puedan consultar en cualquier momento es un aliciente muy interesante.

4. Mejora de autonomía:

Debido a que este sistema se puede instalar en lugares aislados para monitorizar diferentes situaciones, una de las principales características en las que fijarse es la autonomía. Actualmente tiene una autonomía limitada debido a la batería instalada, sin embargo, se podrían realizar diferentes modificaciones para aumentarla aún más, entre ellas estaría:

- Optimización del consumo de energía, ya sea cambiando el hardware o introduciendo un modo de bajo consumo más profundo.
- Implementación de sistemas de almacenamiento de energía basados en almacenamiento térmico para que el sistema se encuentre alimentado incluso en condiciones de baja radiación solar.
- Uso de energía renovable complementaria para diversificar el suministro energético del sistema.
- Aumento de tamaño y calidad de la placa fotovoltaica.

5. Implementación del sistema en LabVIEW:

La implementación y traspaso de este proyecto a un entorno de desarrollo en LabVIEW podría traer diferentes ventajas entre las que destacan:

- Creación de interfaces de usuario personalizadas y altamente interactivas. Esto permitirá mejorar la experiencia del usuario.
- Análisis de datos más sencillo e intuitivo.
- Mayor flexibilidad y escalabilidad, permite adaptar el sistema a diferentes requisitos de una manera sencilla.

6. Cambio de microcontrolador:

En este proyecto se utiliza Arduino UNO R3, el cuál es uno de los módulos de Arduino más utilizados a la vez que simples y sencillos. Existen otro tipo de Arduinos, más compactos y con más funcionalidades, entre los que destaca el Arduino R4 el cuál viene implementado con un módulo WiFi.

El cambio de microcontrolador permitiría ahorrar una gran cantidad de hardware. Además, aportaría una mayor potencia de procesamiento, más recursos de memoria y menor consumo de energía entre otros.

7. Exploración de tecnologías emergentes:

Actualmente la tecnología más utilizada es la inteligencia artificial, por lo que la implementación de esta en el proyecto podría suponer un salto de calidad interesante. Por ejemplo para optimizar al máximo la orientación del panel solar se podría configurar una inteligencia artificial la cual sea capaz de procesar los datos de una base de datos y observar diferentes patrones y comportamientos para ajustar la posición según un criterio.

Otra de las tecnologías más usadas es el blockchain, esto puede ayudar a proteger los datos que se tomen en el sistema en caso de que se quieran realizar confidencialmente, además de la transparencia y rastreabilidad de las medidas.

8. Configuración conexión WiFi dinámica:

En este proyecto para probar su funcionamiento normal es necesario una conexión WiFi que en este caso está configurada internamente en el módulo ESP8266-01.

Por lo que para que el sistema arranque se deberá de conectar al WiFi configurado, y no se conectará a ningún otro, sería interesante realizar ciertos cambios en el código que permitan la configuración dinámica del WiFi o incluso la adición de un teclado con el que configurar inicialmente la conexión WiFi a la que se conectará el módulo ESP8266-01. Esto conseguirá que sea más flexible y dinámico en su uso, ya que no habrá que reconfigurarlo continuamente.

Esta página se ha dejado en blanco intencionadamente

11 REFERENCIAS

- [1] Statista. (2024). "Potencia Solar Fotovoltaica Instalada en España." [Online]. Disponible en: <https://es.statista.com/estadisticas/1004387/potencia-solar-fotovoltaica-instalada-en-espana/>
- [2] El Periódico de la Energía. (2018). "Vanguard 1: la energía solar en el espacio cumple 60 años." [Online]. Disponible en: <https://elperiodicodelaenergia.com/vanguard-1-la-energia-solar-en-el-espacio-cumple-60-anos/>
- [3] Helioesfera. (2016). "Diferencia entre Efecto Fotoeléctrico y Efecto Fotovoltaico." [Online]. Disponible en: <https://www.helioesfera.com/diferencia-entre-efecto-fotoelectrico-y-efecto-fotovoltaico/>
- [4] AutoSolar. (s.f.). "¿Qué significa una instalación fotovoltaica con conexión a red?" [Online]. Disponible en: <https://autosolar.es/aspectos-tecnicos/que-significa-una-instalacion-fotovoltaica-con-conexion-a-red>
- [5] Villa Flores Martín. (s.f.). "Servicios de Energías Renovables." [Online]. Disponible en: <https://www.villafloresmartin.com/es/servicios/energias-renovables/>
- [6] Área Tecnología. (s.f.). "Irradiancia e Irradiación." [Online]. Disponible en: <https://www.areatecnologia.com/electricidad/irradiancia-irradiacion.html>
- [7] J. A. Pelayo López, A. Luna Soto F. Bernabe Ramos, y B. Guzmán Flores, (2018) "Comparativa de la eficiencia entre un sistema fotovoltaico con seguimiento solar y un sistema fotovoltaico fijo" Disponible en: <https://ciba.org.mx/index.php/CIBA/article/view/76>
- [8] Endesa. (2023). "Seguidores Solares." [Online]. Disponible en: <https://www.endesa.com/es/la-cara-e/energias-renovables/seguidores-solares>
- [9] MACHADO TORANZO, Noel et al. Solar tracker, optimizing of improvement of the solar energy. (2015) [online]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59012015000200008&lng=es&nrm=iso
- [10] Motorola. (1996) "1N5817 Datasheet." [Online]. Disponible en: <https://pdf1.alldatasheet.com/datasheet-pdf/view/2816/MOTOROLA/1N5817.html>
- [11] ETC2. (2006). "TP4056 Datasheet." [Online]. Disponible en: <https://pdf1.alldatasheet.es/datasheet-pdf/view/1487471/ETC2/TP4056.html>
- [12] TiendaTec. (2018). "Cargador de litio 18650 USB-C con protección TP4056." [Online]. Disponible en: <https://www.tiendatec.es/electronica/componentes/conversor-dc-dc/1163-cargador-litio18650-usb-c-con-proteccion-tp4056-03962a-8472496016391.html>
- [13] Turibot. (2023). "Batería de 3.7V 4800mAh 18650 Ion de Litio Recargable." [Online]. Disponible en: <https://www.turibot.es/bateria-de-37v-4800mah-18650-ion-de-litio-recargable>
- [14] Amazon. (2021). "Youmile 10 piezas DC-DC 1.5V 1.8V 2.5V 3V 3.3V 3.7V 4.2V a 5V módulo convertidor de refuerzo" [Online]. Disponible en: https://www.amazon.es/dp/B09KMQBWL4?psc=1&ref=ppx_yo2ov_dt_b_product_details
- [15] Turibot. (2023). "Módulo Elevador de Tensión 0.9-5V a 5V DC-DC." [Online]. Disponible en: <https://www.turibot.es/modulo-elevador-de-tension-09-5v-to-5v-dc-dc>

- [16]ON Semiconductor. (2000). "2N3904 Datasheet." [Online]. Disponible en: <https://pdf1.alldatasheet.com/datasheet-pdf/view/11470/ONSEMI/2N3904.html>
- [17]Texas Instruments. (2013). "LM3940 Datasheet." [Online]. Disponible en: <https://pdf1.alldatasheet.com/datasheet-pdf/view/517666/TI1/LM3940.html>
- [18]Espressif. (2015). "ESP8266 Datasheet." [Online]. Disponible en: <https://pdf1.alldatasheet.es/datasheet-pdf/view/1148031/ESPRESSIF/ESP8266.html>
- [19]CyberBlogSpot. (2023). "ESP-01 and ESP-01S Pinout and Configuration." [Online]. Disponible en: <https://cyberblogspot.com/esp-01-and-esp-01s-pinout-and-configuration/>
- [20]Amazon. (2018). "ESP8266 01 ESP de 01 Adaptador para Raspberry Pi Breadboardadapter" [Online]. Disponible en: https://www.amazon.es/dp/B079TBS88H?ref=ppx_yo2ov_dt_b_product_details&th=1
- [21]Bosch. (2017). "BME680 Datasheet." [Online]. Disponible en: <https://pdf1.alldatasheet.com/datasheet-pdf/view/1132061/BOSCH/BME680.html>
- [22]SonRobots. (2010). "SG90 Servomotor TowerPro 9g." [Online]. Disponible en: <https://sonrobots.com/producto/sg90-servomotor-towerpro-9g/>
- [23]AZ-Delivery. (2019). "Módulo sensor de luz." [Online]. Disponible en: <https://www.az-delivery.de/es/products/licht-sensor-modul>
- [24]SunFounder. (2022). "Componente LCD I2C." [Online]. Disponible en: https://docs.sunfounder.com/projects/vincent-kit/en/latest/components/component_i2c_lcd.html
- [25]Handsontec. (2012). "I2C Serial Interface 1602 LCD Module". [Online]. Disponible en: https://www.handsontec.com/dataspecs/module/I2C_1602_LCD.pdf
- [26]Arduino. (2024). "Arduino Uno Rev3." [Online]. Disponible en: <https://docs.arduino.cc/hardware/uno-rev3/>
- [27]ProtoSupplies. (2017). "Sensor Shield V5.0." [Online]. Disponible en: <https://protosupplies.com/product/sensor-shield-v5-0/>
- [28]Tinkercad. (2011). "Tinkercad: Modelos 3D para imprimir". [Online]. Disponible en: <https://www.tinkercad.com/>
- [29]Naylampmechatronics. (2016) "Tutorial ESP8266 - Parte I," Naylampmechatronics.com,[Online]. Available: https://naylampmechatronics.com/blog/21_tutorial-esp8266-parte-i.html.
- [30]Esp8266/Arduino. (2022). "Librería ESP8266WiFi.h." [Online]. Disponible en: <https://github.com/esp8266/Arduino/blob/master/libraries/ESP8266WiFi/src/ESP8266WiFi.h>
- [31]Arduino Libraries. (2019). "Librería NTPClient.h." [Online]. Disponible en: <https://github.com/arduino-libraries/NTPClient>
- [32]esp8266/Arduino. (2019). "Librería WiFiUdp.h." [Online]. Disponible en: <https://github.com/esp8266/Arduino/blob/master/libraries/ESP8266WiFi/src/WiFiUdp.h>
- [33]Arduino. (2017). "Librería SoftwareSerial.h." [Online]. Disponible en: <https://www.arduino.cc/en/Reference/SoftwareSerial>
- [34]Arduino. (2020). "Librería Time.h." [Online]. Disponible en: <https://www.arduino.cc/reference/en/libraries/time/>

- [35]Adafruit. (2019). "Librería Adafruit Sensor.h." [Online]. Disponible en: https://github.com/adafruit/Adafruit_Sensor
- [36]Adafruit. (2018). "Librería Adafruit BME680.h." [Online]. Disponible en: https://github.com/adafruit/Adafruit_BME680
- [37]Arduino. (2018). "Librería Wire.h." [Online]. Disponible en: <https://www.arduino.cc/en/Reference/Wire>
- [38]MathWorks. (2020). "Librería ThingSpeak.h." [Online]. Disponible en: <https://github.com/mathworks/thingspeak-arduino>
- [39]Arduino. (2018). "Librería LowPower.h." [Online]. Disponible en: <https://github.com/rockscream/Low-Power>
- [40]Arduino. (2019). "Librería Servo.h." [Online]. Disponible en: <https://www.arduino.cc/en/Reference/Servo>
- [41]Naciones Unidas. (2022). "Objetivos de Desarrollo Sostenible." [Online]. Disponible en: <https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/>
- [42]Solmic. (2021). "Paneles Solares en Zonas Rurales." [Online]. Disponible en: <https://www.solmic.co/paneles-solares-en-zonas-rurales>
- [43]Desenchúfate. (2017). "5 Beneficios Clave de los Paneles Solares al Medio Ambiente." [Online]. Disponible en: <https://desenchufate.es/5-beneficios-clave-de-los-paneles-solares-al-medio-ambiente/>

Esta página se ha dejado en blanco intencionadamente

12 BIBLIOGRAFÍA

- [1] Programar Fácil. (2018). "Cómo configurar ESP-01 (WiFi ESP8266)." [Online]. Disponible en: <https://programarfácil.com/podcast/como-configurar-esp01-wifi-esp8266/>
- [2] Instructables. (2019). "Getting Started With the ESP8266 ESP-01." [Online]. Disponible en: <https://www.instructables.com/Getting-Started-With-the-ESP8266-ESP-01/>
- [3] Rantec Tutoriales. (2016). "___Trashed-23." [Online]. Disponible en: https://tutoriales.rantec.mx/_trashed-23/
- [4] Arduino France. (2023). "Módulo ESP8266." [Online]. Disponible en: <https://sp.arduino-france.site/modulo-esp8266/>
- [5] HWlibre. (2015). "Servo SG90." [Online]. Disponible en: <https://www.hwlibre.com/servo-sg90/>
- [6] Repsol. (2023). "Seguidores Solares." [Online]. Disponible en: <https://www.repsol.com/es/energia-futuro/futuro-planeta/seguidores-solares/index.cshtml>
- [7] Arduino. (2015). "Librería LiquidCrystal_I2C.h." [Online]. Disponible en: <https://www.arduino.cc/reference/en/libraries/liquidcrystal-i2c/>
- [8] Esparcia, J. Á. (2021). "Estudio comparativo de una instalación fotovoltaica aislada y una conectada a red." [Tesis de grado, Universitat Politècnica de València]. Repositorio Institucional de la Universitat Politècnica de València. Disponible en: <http://hdl.handle.net/10251/160887>

Esta página se ha dejado en blanco intencionadamente

Anexos

A.1 Manual de instalación

Pese a que se utiliza en este proyecto el ArduinoIDE para configurar los microprocesadores y codificar el sistema, para que esto se pueda realizar en todos los sistemas operativos, se deberá realizar una serie de instalaciones de librerías previamente.

La instalación de librerías es fundamental para el funcionamiento de cualquier código de Arduino, en este caso y para instalar la mayoría de ellas utilizamos el IDE de Arduino el cual tiene incorporado un instalador como el mostrado en la Figura 12-1:

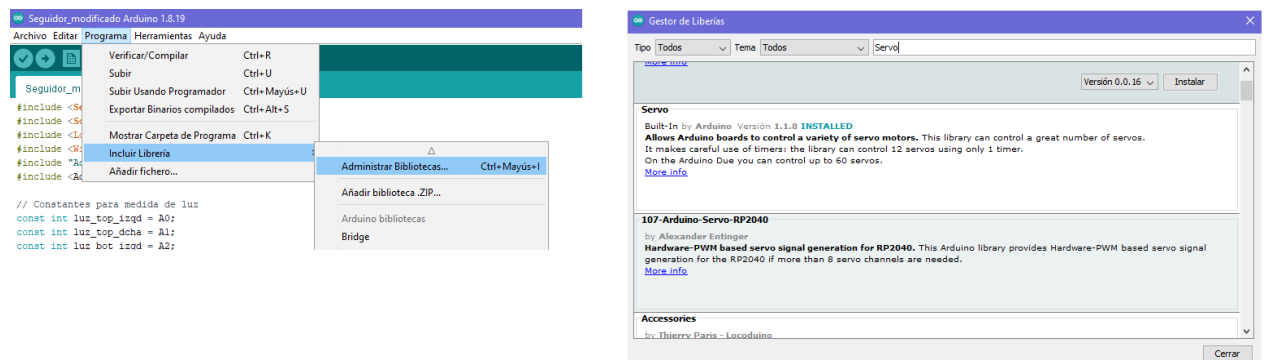


Figura 12-1: Acceso al gestor de librerías del Arduino IDE

En caso de no encontrarlas directamente, se podrá acceder a los repositorios de GitHub de Arduino, descargarlas en un zip e importarlas directamente a nuestro sistema de la manera mostrada en la Figura 12-2:

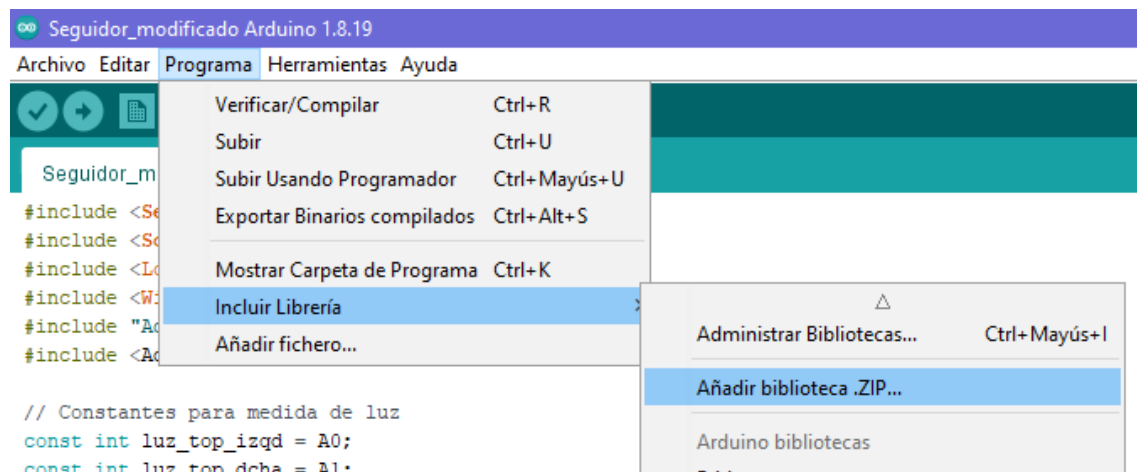


Figura 12-2: Instalación manual de librería de Arduino

Si se realiza la última opción se abrirá el explorador de archivos, pudiendo así seleccionar el zip que se quiera marcar como librería.

A.1.1. Librerías usadas por el Arduino UNO

Arduino es una plataforma de hardware y software de código abierto, esto implica que su diseño y documentación están disponibles públicamente y pueden ser utilizados por cualquier persona. Por tanto y para hacer nuestro proyecto más legible utilizaremos las librerías de libre acceso para facilitar el desarrollo.

- Librería para los servomotores:

La librería "Servo.h" [40] es una de las librerías estándar incluidas en el entorno de desarrollo de Arduino. Esta librería proporciona una interfaz simple para controlar servomotores mediante señales de ancho de pulso modulado (PWM), utilizándola se ahorrará el control de dichas señales.

Además de simplificar el control de los servos, ofrece una gama de funciones adicionales. Estas funciones permiten detener el movimiento del motor en cualquier momento, liberar el pin asociado al servo para otros usos, y obtener información valiosa como el ángulo actual del servo, entre otras posibilidades.

- Librería para la comunicación serial:

Para este proyecto, se empleará la librería "SoftwareSerial.h" [33], la cual fue inicialmente desarrollada por David A.Mellis, aunque ha sido actualizada y mantenida por la comunidad de Arduino.

Su utilización resulta fundamental, ya que habilita la configuración de la comunicación serial empleando cualquier conjunto de pines digitales disponibles en la placa Arduino. Esto implica que no haya una restricción a los pines Tx y Rx del Arduino UNO, sino que da la libertad de utilizar cualquier otro par de pines disponibles. Además, esta librería ofrece una interfaz intuitiva tanto para transmitir como para recibir datos, sin requerir la incorporación de hardware adicional.

- Librería para bajo consumo

La introducción de un sistema de bajo consumo es esencial para el programa. En este caso se va a utilizar la librería "LowPower.h" [39] creada por Richard Omar González y posteriormente actualizada por la comunidad de Arduino.

Esta librería ofrece una variedad de funciones que permiten poner el microcontrolador en diferentes estados de bajo consumo energético, como el modo de suspensión (sleep mode), el modo de reposo (standby mode) o el modo de ahorro de energía (power down mode). Estos modos ayudan a reducir significativamente el consumo de energía del dispositivo cuando no está activamente realizando tareas, prolongando así la vida útil de la batería o permitiendo que el dispositivo funcione de manera más eficiente en términos de energía.

- Librería para la comunicación I2C

El proyecto utilizará el sensor BME680 para realizar medidas de temperatura y humedad, así como el LCD 1602 para la representación de los valores, para realizar una buena comunicación

con el Arduino necesitaremos una conexión I2C, para esto se utilizará la librería “Wire.h” [37], la cual, al igual que la librería de los servomotores, ha sido creada por el entorno de desarrollo de Arduino.

Esta librería proporciona funciones y métodos para configurar y utilizar el bus I2C en proyectos de Arduino. Permite a los desarrolladores enviar y recibir datos entre el microcontrolador Arduino y otros dispositivos I2C, como sensores, actuadores, pantallas LCD, módulos de memoria, entre otros.

- Librerías para el sensor BME680

En este caso se utilizarán dos librerías esenciales para que el BME680 funcione correctamente, “Adafruit_Sensor.h” [35] y “Adafruit_BME680.h” [36]. Ambas librerías han sido desarrolladas por Adafruit Industries, una empresa que proporciona tutoriales y productos relacionados con la electrónica y la programación.

Las librerías facilitan la integración del sensor BME680 de Bosch en proyectos de Arduino proporcionando una interfaz simple y conveniente para inicializar el sensor y obtener lecturas precisas de temperatura y humedad.

- Librería para el manejo del LCD 1602

Para la representación de valores en el LCD se utilizará la librería “LiquidCrystal_I2C.h” [7] creada por Frank de Brabander y mantenida posteriormente por la comunidad Arduino, especialmente por Marco Schwartz, esta facilitará la comunicación por I2C así como la escritura, borrado y actualización de valores de la pantalla LCD.

- Librería para el manejo del tiempo Epoch

Al hacer una consulta a un servidor NTP, este responde con el tiempo en segundos que ha pasado desde el 1 de Enero de 1970, para procesar esta información y poder dividirla correctamente en diferentes variables, es necesario el uso de la librería “Time.h” [34], la cuál ha sido creada por el propio entorno de desarrollo de Arduino.

A.1.2. Librerías usadas por el ESP8266-01

En este caso se ha configurado el IDE para que sea capaz de codificar nuestro ESP8266-01 como si fuese un sistema de Arduino, dando así las herramientas necesarias como las librerías. Para poder utilizar este módulo mediante el Arduino IDE hemos hecho los siguientes cambios reflejados en la Figura 12-3:

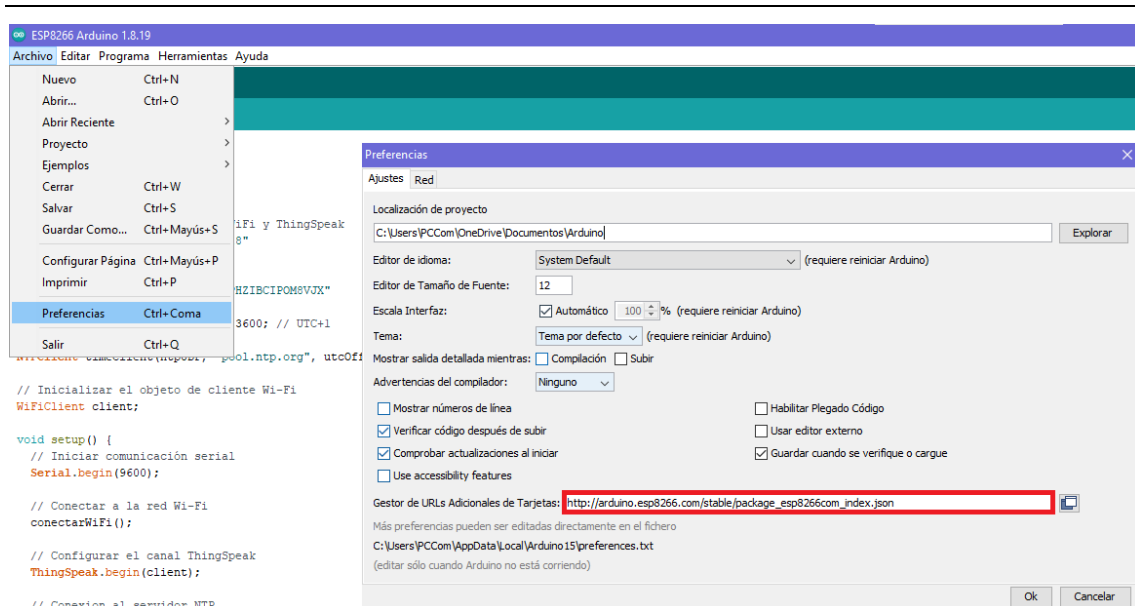


Figura 12-3: Configuración IDE Arduino para el uso de ESP8266

Cuando se agrega esta URL a la configuración en el Arduino IDE, se indica al IDE que busque y descargue el paquete de soporte para el ESP8266. Esto permite programar y cargar código en el ESP8266 utilizando el Arduino IDE, lo que facilita el desarrollo de proyectos. Una vez el IDE se haya actualizado con el paquete de soporte indicado, podremos simplemente indicar la placa que queremos codificar como se indica en la Figura 12-4:

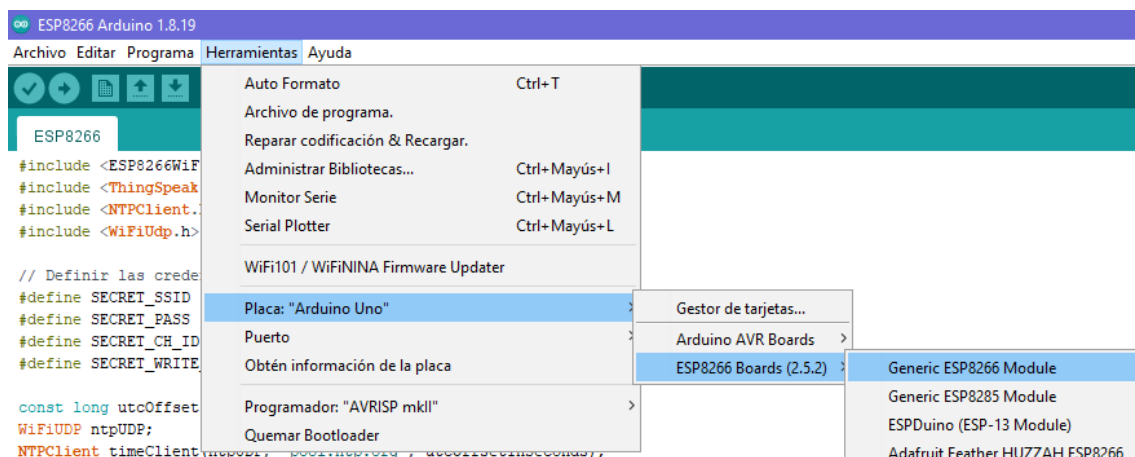


Figura 12-4: Selección de placa a codificar

- Librería para conexión WiFi

Debido a que se está utilizando el ESP8266-01, se va a utilizar la librería “ESP8266WiFi.h” [30] la cuál ha sido creada por el equipo de desarrollo de Espressif Systems, una empresa con sede en china que también creó el propio ESP8266-01. Para instalar esta librería se debe acceder a directorios de GitHub que la publiquen ya que no se puede instalar directamente desde el IDE de Arduino.

Esta librería proporciona todas las funciones necesarias para configurar y gestionar la conectividad WiFi en un dispositivo basado en ESP8266, como puede ser la conexión a redes

WiFi, configuración de modo de operación, gestión de IP (Internet Protocol o Protocolo de Internet)...

- Librería para conexión a ThingSpeak

A la hora de mandar información del ESP8266-01 a la web de ThingSpeak es necesario realizar una conexión utilizando la librería "ThingSpeak.h" [38] la cuál fue desarrollada por MathWorks, la misma empresa que creó MATLAB.

La librería proporciona una interfaz para integrar fácilmente la plataforma ThingSpeak en proyectos de Arduino, permitiendo enviar y recibir datos de forma sencilla a través de la API de ThingSpeak.

- Librería para sincronismo con NTP

En este sistema es necesario realizar una sincronización con un servidor NTP para obtener así el 'epoch time', que es el tiempo transcurrido en segundos desde el 1 de Enero de 1970. En este caso se utiliza la librería "NTPClient.h" [31], la cual fue desarrollada por Farbice Weinberg. También será necesario el uso de la librería "WiFiUdp.h" [32], para el manejo de protocolos UDP y soporte a la librería del servidor NTP, esta última librería fue creada por el equipo de Arduino.

El NTPClient.h proporciona una interfaz para la sincronización utilizando el protocolo NTP (Network Time Protocol) en proyectos de Arduino, simplificando así la obtención y gestión precisa del tiempo en aplicaciones IoT, permitiendo a los desarrolladores sincronizar sus dispositivos con servidores de tiempo públicos.

A.2 Manual de usuario

Una vez se ha explicado en el apartado de APLICACIÓN SOFTWARE DESARROLLADA el funcionamiento general del sistema, en esta sección se procederá a la explicación de su uso por parte de un usuario paso a paso, diferenciando claramente la ejecución del modo TEST y el modo normal. Se tendrá en consideración que el sistema inicialmente no está alimentado, por lo que el paso inicial será el siguiente:

Introducir la batería de ion-litio en el porta baterías habilitado para ello, teniendo en consideración la parte positiva y negativa, NUNCA se deberá de polarizar inversamente ya que el sistema no está protegido frente a esto y provocará un mal funcionamiento.

A.2.1. Ejecución modo TEST

En caso de no realizar nada más que la introducción de la batería, el sistema se iniciará en modo TEST y el usuario únicamente tendrá que visualizar los valores medidos en el LCD, así como el modo de funcionamiento elegido y observar cómo los servomotores orientan el panel solar a la posición de máxima luz.

A.2.2. Ejecución modo normal

Una vez introducida la batería se dispondrá de 5 segundos para realizar una pulsación en el micropulsador situado en el centro de la placa, mostrado en la Figura 12-5 si se quiere seleccionar el modo normal de ejecución.

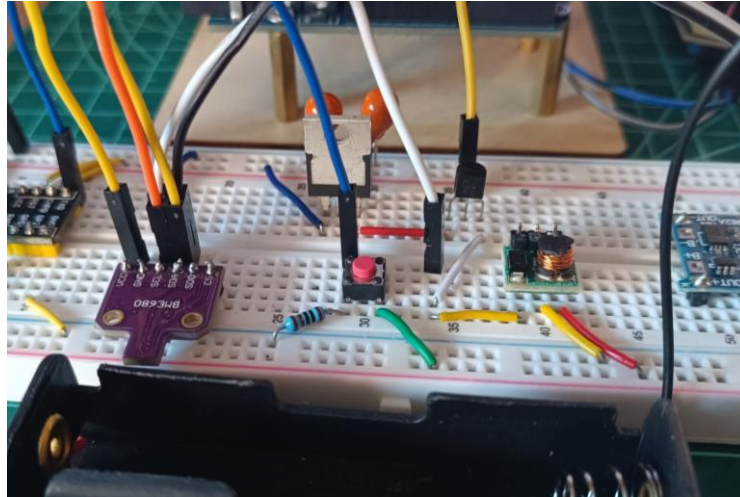


Figura 12-5: Situación del micropulsador en la placa de prototipado

Una vez iniciado el modo normal de ejecución, se debe disponer de conexión a internet para poder acceder a la web de ThingSpeak y poder comprobar así el correcto funcionamiento del sistema.

Inicialmente el usuario observará en el LCD que el modo de funcionamiento es “Normal” y tras esperar aproximadamente 5 segundos, el LCD se actualizará para mostrar la hora actual, así como los valores de temperatura y humedad, este tiempo de espera dependerá de la calidad de conexión WiFi que disponga.

En este punto el usuario podrá acceder a la web de ThingSpeak con sus credenciales y buscar el canal mediante el nombre de usuario: “mwa0000017460264” tal y como se muestra en la Figura 12-6, de esta manera podrá visualizar los datos y comprobar que son los mismos que los mostrados por el LCD.

Public Channels for User: mwa0000017460264

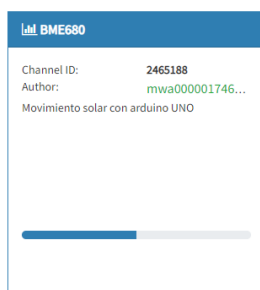


Figura 12-6: Búsqueda del canal por id de usuario

Para cambiar de modo será necesario reiniciar el sistema, esto se hace simplemente quitando la batería de ion-litio de su porta baterías y volviendo a conectarla.