.UBAfiuba
FACULTAD DE INGENIERÍA

Desarrollo de un sistema embebido para la gestión y monitoreo de cultivos verticales

Ing. José Luis Krüger

Carrera de Especialización en Sistemas Embebidos

Director: Esp. Ing. Mariano Campos (FIUBA)

Jurados:

Jurado 1 (pertenencia)
Jurado 2 (pertenencia)
Jurado 3 (pertenencia)

Ciudad de [lugar], [mes] de [año]

Resumen

Este trabajo presenta el desarrollo, construcción, instalación y prueba de un dispositivo semiautomático de monitoreo y gestión para un cultivo hidropónico vertical. Su implementación requirió la aplicación de conocimientos en electrónica, programación, sistemas operativos de tiempo real, testing de software, diseño y ensamblaje de PCB, integración de sensores y protocolos de comunicación.

El sistema mejora la eficiencia del cultivo al minimizar los desperdicios y aumentar la productividad en comparación con los métodos tradicionales. Este avance representa un paso clave hacia la sostenibilidad y la optimización de la producción de alimentos en el futuro.

Agradecimientos

Esta sección es para agradecimientos personales y es totalmente **OPCIONAL**.

Índice general

Índice de figuras

Índice de tablas

Dedicado a... [OPCIONAL]

Capítulo 1

Introducción general

En este capítulo se expone la necesidad que dio origen al desarrollo del trabajo. Se presenta el concepto de cultivo vertical y el estado del arte de su integración con la tecnología. Asimismo, se explican el objetivo y los alcances del trabajo.

1.1. Agricultura vertical como solución sostenible

Con una población mundial que supera los 8000 millones de personas y continúa en crecimiento [1], la agricultura enfrenta el desafío de volverse más eficiente y sostenible [2] [3] [4]. En este contexto, los cultivos verticales emergen como una solución innovadora que optimiza el uso del espacio y los recursos, especialmente en entornos urbanos, donde el suelo es limitado y costoso.

Este tipo de agricultura utiliza técnicas como la hidroponía, que permite un uso preciso del agua, y la aeroponía, que maximiza el oxígeno disponible para las raíces. Además, las granjas verticales aprovechan áreas infrautilizadas, como edificios abandonados o naves industriales, y permiten cultivar alimentos en zonas donde la agricultura tradicional resulta dificil de desarrollar. Así, se logra una mayor densidad de cultivo y se contribuye a la sostenibilidad al reducir el uso de pesticidas y fertilizantes.

Aunque los cultivos verticales requieren un alto nivel de tecnología y tienen costes energéticos asociados, su capacidad para ahorrar recursos, disminuir la huella de carbono y fomentar la producción de alimentos en espacios reducidos y con menor dependencia de factores climáticos, justifica la inversión en su desarrollo y expansión [5].

Este método, también conocido como agricultura urbana, permite el crecimiento de plantas en interiores sin necesidad de luz solar directa, gracias a la aplicación de tecnologías agrícolas avanzadas. Estas optimizan las condiciones de cultivo, lo que facilita la propagación de plantas jóvenes y la producción de alimentos más saludables sin el uso de pesticidas.

Además, maximizan la producción al emplear sistemas de iluminación especializados de nueva generación, diseñados para estructuras multicapa. Gracias a esta tecnología, es posible obtener mayores rendimientos en un espacio reducido, consolidando a la agricultura vertical como una solución sostenible y eficiente para la producción de alimentos en entornos urbanos y con recursos limitados [4].

1.1.1. Tipos de cultivos verticales

Existen distintos tipos de agricultura vertical, que optimizan el crecimiento de las plantas sin grandes extensiones de suelo y emplean estructuras especializadas para maximizar la producción. A continuación, se describen los principales tipos de cultivos verticales y sus características.

Hidroponía

La hidroponía es un método de cultivo sin suelo que suministra nutrientes a las plantas a través de una solución acuosa, con raíces en un medio inerte como lana de roca o perlita. Es un sistema eficiente en el uso de agua, permite un control preciso de nutrientes y favorece una mayor densidad de cultivo y un crecimiento más rápido que los métodos tradicionales [4] [6].

Aeroponía

La aeroponía es un método de cultivo en el que las raíces de las plantas quedan suspendidas en el aire y se rocían con una solución nutritiva. Maximiza la oxigenación, acelera el crecimiento y optimiza el uso de nutrientes, aunque requiere un control preciso para evitar la desecación de las raíces [4] [6].

Acuaponía

La acuaponía combina la cría de peces con la hidroponía, utilizando los desechos de los peces como fertilizante para las plantas, mientras estas purifican el agua. Es un sistema sostenible que minimiza el desperdicio, promueve la biodiversidad y es viable en entornos urbanos y rurales [6].

1.1.2. Sistemas de agricultura vertical más conocidos

Los sistemas de agricultura vertical varían según el tipo de cultivo y la técnica más adecuada para cada necesidad. A continuación, se describen los principales sistemas de cultivos verticales y sus características.

Sistemas de torre

Las torres verticales permiten el cultivo de plantas en estructuras cilíndricas apiladas, con el fin de maximizar el uso del espacio y facilitar tanto la recolección como el mantenimiento. Este sistema es ideal para cultivos de hoja verde y hierbas, y se integra fácilmente en espacios urbanos como balcones y terrazas [6].

Sistemas en rack o estanterías

Estos sistemas emplean estanterías apiladas donde las plantas crecen en bandejas o contenedores, son ideales para invernaderos y ambientes controlados. Facilitan el riego, la recolección y el monitoreo, al mismo tiempo que se optimiza el uso del espacio. Los sistemas en rack o estanterías son especialmente útiles para cultivos que requieren diferentes niveles de luz y humedad [6].

Sistemas montados en pared

En estos sistemas, las plantas crecen en módulos montados en paredes. Su integración en la arquitectura no solo optimiza el espacio, sino que también aporta beneficios estéticos y funcionales. Los jardines verticales pueden mejorar la calidad del aire y contribuir al aislamiento térmico de los edificios [6].

Sistemas de bastidor en A

El bastidor en forma de "A" proporciona una estructura estable y accesible para el cultivo vertical, lo que a su vez facilita el acceso a la luz y el riego. Este diseño es particularmente útil para cultivos que necesitan un soporte adicional, como tomates y pepinos, y puede ser utilizado tanto en interiores como en exteriores [6].

Sistemas de contenedores

Estos sistemas utilizan contenedores apilables que pueden moverse y configurarse según las necesidades, adaptándose a diversos entornos y tipos de cultivo. Los contenedores son ideales para cultivos modulares y pueden ser personalizados para optimizar el uso del espacio y los recursos [6].

1.2. Motivación

Este trabajo surge como un desarrollo de interés personal, impulsado por el deseo de contribuir positivamente al desarrollo sustentable y la optimización de los recursos naturales en la producción de alimentos. En la actualidad, el crecimiento de la población mundial y el aumento en la demanda de productos agrícolas presentan grandes desafíos ambientales y económicos.

La producción tradicional de alimentos enfrenta dificultades debido a la explotación intensiva de suelos, el uso excesivo de fertilizantes y pesticidas y el desperdicio de recursos esenciales como el agua. Con el fin de maximizar sus ganancias, muchas empresas recurren a métodos poco sostenibles que generan un impacto ambiental negativo, y afectan la biodiversidad, la calidad del suelo y el bienestar de las comunidades locales que dependen de estas fuentes de producción.

Uno de los problemas más críticos en la agricultura tradicional es el significativo desperdicio de agua en el riego. Se estima que se desperdicia hasta el 84 % del agua utilizada para riego [7]. Esta pérdida ocurre principalmente por evaporación, drenaje o absorción ineficiente. En un contexto donde la crisis hídrica es una amenaza global, desarrollar sistemas más sostenibles se vuelve una necesidad urgente.

Por otro lado, existe una creciente preocupación entre los consumidores sobre la calidad y procedencia de los alimentos. Cada vez más personas optan por un estilo de vida saludable, adoptando hábitos de consumo responsables, como el vegetarianismo, el veganismo y el autocultivo. Sin embargo, muchos de estos consumidores no cuentan con espacios adecuados ni conocimientos técnicos para producir sus propios alimentos, lo que genera una gran oportunidad para soluciones tecnológicas que faciliten la agricultura doméstica.

A partir de estas problemáticas, nació la motivación para desarrollar un sistema hidropónico inteligente, que no solo optimice el uso del agua, sino que también permita una gestión eficiente del cultivo con mínima intervención del usuario. Este trabajo ofrece una alternativa viable y escalable para la producción de alimentos frescos, reduce la dependencia de sistemas agrícolas tradicionales y promueve un modelo más sustentable para el futuro.

1.3. Estado del arte

Durante la etapa de investigación, se llevó a cabo una búsqueda de productos comerciales tanto en el mercado local como en el internacional. Se identificaron diversas soluciones con características similares al sistema desarrollado.

A continuación, se describen los productos identificados.

NIDO PRO

El NIDO PRO, figura 1.1, es un sistema hidropónico inteligente para cultivos verticales que permite controlar la solución nutritiva a través de una aplicación móvil. Facilita la gestión del pH y la conductividad eléctrica (CE), con algoritmos que realizan comprobaciones diarias para garantizar la estabilidad de la solución nutritiva. Además, admite hasta cuatro ranuras para fertilizantes líquidos y ajustes de pH, lo que optimiza el proceso de nutrición de las plantas [8].

FIGURA 1.1. Ilustración del módulo NIDO PRO conectado a un sistema hidropónico horizontal[1].

Xiaomi Mi Flower Care Plant Sensor

El Xiaomi Mi Flower Care Plant Sensor, figura 1.2, monitorea variables ambientales como luz, humedad, nutrientes y temperatura del sustrato. Los sensores se conectan a aplicaciones móviles, lo que permite realizar un seguimiento detallado y controlar las condiciones del cultivo de manera precisa [9].

[1]Imagen tomada de https://acortar.link/uEmExI

FIGURA 1.2. Ilustración del dispositivo Xiaomi Mi Flower Care Plant Sensor y sus prestaciones[2].

Kit de Riego Hydro de Konyks

El sistema de riego inteligente Konyks, figura 1.3, permite controlar la llave de paso de forma remota mediante una aplicación móvil y asistentes de voz como Alexa, Google Home o Siri. Incorpora un medidor de caudal para monitorear el consumo de agua y una toma con relé RF que amplía la conectividad hasta 60 m y permite gestionar hasta cuatro grifos. Además, ofrece automatización programable según el clima y los horarios, lo que optimiza el uso del agua y facilita la gestión del riego desde cualquier lugar [10].

FIGURA 1.3. Sistema de riego conectado Konyks[3].

[2]Imagen tomada de https://acortar.link/bVNwHC

[3]Imagen tomada de https://konyks.com/es/producto/hydro-kit/

Smart 9 Pro de Click & Grow

El Smart Garden 9 PRO, figura 1.4, es un sistema de cultivo doméstico automatizado con control a través de una aplicación. Ofrece riego automático, iluminación ajustable tanto por control táctil como desde la app, y un suministro equilibrado de nutrientes y oxígeno en las raíces. Permite cultivar hierbas, frutas, ensaladas y flores durante todo el año, con la opción de utilizar cápsulas presembradas (más de 50 variedades disponibles) o semillas propias. Incluye un set inicial con cápsulas de tomate, albahaca y lechuga [11].

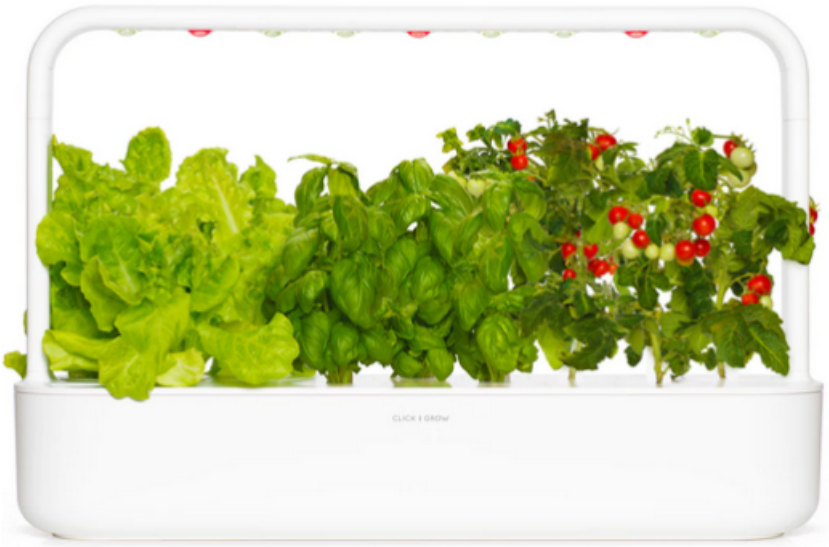


FIGURA 1.4. Módulo Smart Garden 9 PRO[4].

1.4. Objetivos y alcances

A continuación, se presentan los objetivos del trabajo.

1.4.1. Objetivos

- Desarrollar un prototipo de sistema hidropónico automatizado para monitoreo y gestión eficiente del cultivo.
- Implementar la conexión de sensores para medir temperatura, humedad, pH, conductividad eléctrica, nivel de agua e intensidad lumínica.
- Diseñar una interfaz web y móvil que permita la supervisión y configuración remota del sistema.
- Evaluar el rendimiento del sistema en comparación con métodos tradicionales, midiendo eficiencia y productividad.
- Poner en práctica los conocimientos adquiridos a lo largo de la carrera de especialización en sistemas embebidos.

[4]Imagen tomada de https://www.clickandgrow.com/products/the-smart-garden-9-pro

1.4.2. Alcances

A continuación, se presentan los alcances del trabajo.

- El sistema permite el monitoreo en tiempo real y la gestión automatizada del riego, iluminación y ventilación.
- El dispositivo se enfocó en cultivos hidropónicos verticales de pequeña y mediana escala.
- Se realizaron pruebas de funcionamiento y validación en un entorno controlado, pero no se contempló una implementación comercial en esta etapa.

Capítulo 2

Introducción específica

En el presente capítulo se describen los componentes de hardware, software, protocolos de comunicación y plataformas utilizados para realizar el trabajo.

2.1. Componentes principales del hardware

En esta sección se describen los módulos de hardware de terceros utilizados en el trabajo.

2.1.1. ESP32-DevKitC

La placa ESP32-DevKitC V4, figura 2.1, es una placa de desarrollo basada en el SoC (*System On Chip*) ESP32 de Espressif Systems. Es una placa popular y versátil ampliamente utilizada por desarrolladores, ingenieros y aficionados para la creación de prototipos y el desarrollo de proyectos de internet de las cosas o IoT (del inglés *Internet of Things*), sistemas embebidos y otras aplicaciones [12].

Cacterísticas generales:

- Microcontrolador con arquitectura de uno o dos núcleos de 32 bits con velocidades de reloj de hasta 240 MHz.
- Memoria SRAM integrada.
- Conectividad Wi-Fi 802.11 b/g/n (2.4 GHz) integrada.
- Conectividad bluetooth (classic y low energy) integrada.

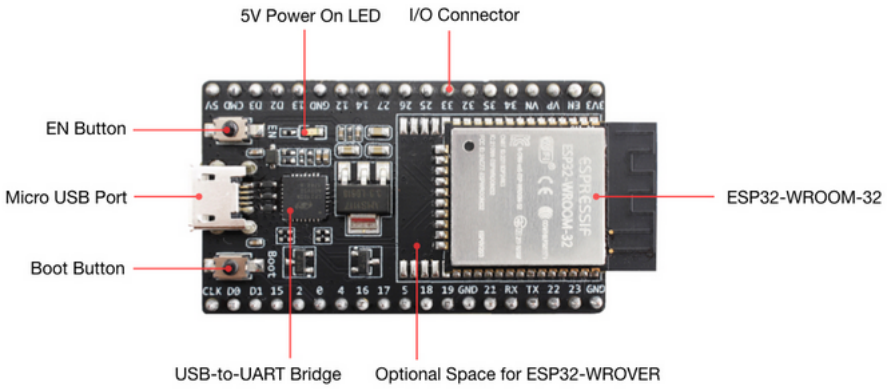


FIGURA 2.1. ESP32-DevKitC V4 con el módulo ESP32-WROOM-32 integrado[1].

2.1.2. Módulo sensor PH-4502C

El sensor de pH analógico PH-4502C, figura 2.2, es un módulo electrónico diseñado para medir el grado de acidez de soluciones líquidas, típicamente el modelo E201-BNC [13]. Este sensor proporciona una salida de tensión analógica que es proporcional al nivel de pH detectado por el electrodo.

Características:

- Rango de detección de pH de 0 a 14.
- Salida analógica que varía, entre 0 y 5 VDC, con el pH del liquido.
- Temperatura de operación del módulo generalmente entre -10 °C y 50 °C.

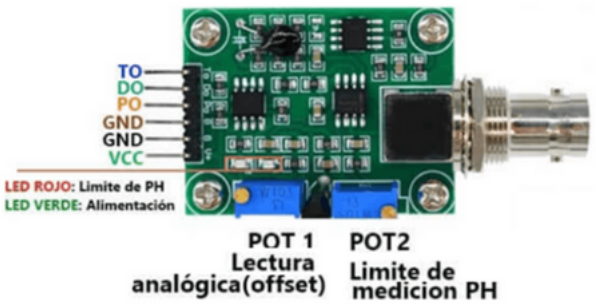


FIGURA 2.2. Distribución de pines del módulo PH-4502C[2].

2.1.3. Módulo medidor de sólidos disueltos totales

El medidor de sólidos disueltos totales o TDS (del inglés *Total Dissolved Solids*) Meter 1.0, figura 2.3, es un sensor o módulo electrónico diseñado para medir la cantidad total de sustancias orgánicas e inorgánicas disueltas en un líquido, expresada típicamente en partes por millón (ppm) o miligramos por litro (mg/l) [14]. Este tipo de sensor se utiliza para evaluar la calidad del agua en diversas aplicaciones como hidroponía, acuicultura, tratamiento de aguas y monitoreo ambiental.

Características:

- Salida analógica de 0 a 2,3 VDC (proporcional a TDS).
- Rango de medición típico de 0 a 1000 ppm.
- Interfaz de 3 pines (VCC, GND, output).
- Conector para la sonda (2 pines).

2.1.4. Sensor de humedad capacitivo V2.0

El sensor de humedad capacitivo V2.0, figura 2.4, mide los niveles de humedad mediante detección capacitiva en lugar de resistiva, como otros sensores disponibles. Fabricado con material resistente a la corrosión, ofrece una vida útil prolongada al insertarse en el sustrato alrededor de las plantas [15].

[1]Imagen tomada de https://docs.espressif.com

[2]Imagen tomada de https://uelectronics.com/producto/sensor-de-ph-liquido/

[3]Imagen tomada de https://www.digikey.be/htmldatasheets

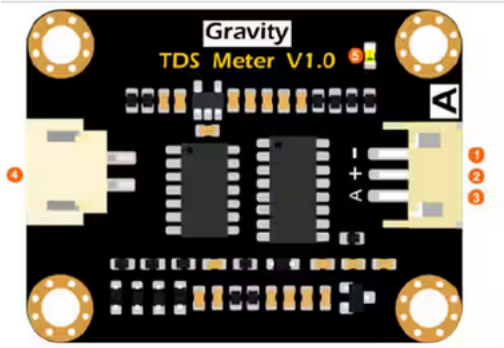


FIGURA 2.3. Ilustración del módulo sensor TDS meter v1.0[3].

- Requiere alimentación de 3,3 a 5,5 VDC.
- Corriente de operación de 5 mA.
- Salida analógica.

FIGURA 2.4. Módulo sensor de humedad capacitivo[4].

2.1.5. Sensor de temperatura digital DS18B20

El DS18B20, figura 2.5, es un sensor de temperatura digital que proporciona mediciones en grados Celsius con una resolución configurable de 9 a 12 bits [16].

Características:

- Rango de medición de temperatura: -55 °C a +125 °C.
- Precisión: ±0,5 °C en el rango de -10 °C a +85 °C.
- Interfaz de comunicación: 1-Wire (requiere un solo pin digital).

[4]Imagen tomada de https://probots.co.in/

[5]Imagen tomada de https://tienda.ityt.com.ar/sensor-temp-hum-ic/

FIGURA 2.5. Pinout del sensor DS18B20 y presentación del chip con su vaina protectora característica[5].

2.1.6. Sensor de luz ambiental digital BH1750

El circuito integrado BH1750, figura 2.6, es un sensor de luz ambiental digital con interfaz de bus I2C (*Inter-Integrated Circuit*) [17]. Este sensor es capaz de detectar un amplio rango de intensidad luminosa, desde 1 hasta 65535 lux, con una resolución de 16 bits. El chip proporciona una salida digital directa, lo que elimina la necesidad de cálculos complejos.

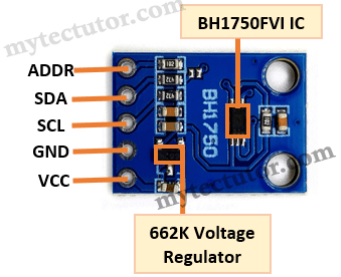


FIGURA 2.6. Pinout del módulo de adaptación del sensor BH1750[6].

2.2. Componentes principales del software

En esta sección se describen las herramientas de software de terceros utilizados en el trabajo.

2.2.1. ESP-IDF

ESP-IDF (*Espressif IoT Development Framework*) es un conjunto de herramientas de desarrollo integral provisto por Espressif Systems. Este framework facilita la creación de firmware para su línea de SoCs ESP32 y ESP8266. Incluye un sistema operativo en tiempo real (FreeRTOS), bibliotecas con APIs para diversos periféricos y protocolos (Wi-Fi, bluetooth, TCP/IP), compilador (basado en GCC), depurador (GDB) y utilidades para la construcción, flasheo y monitoreo de proyectos. ESP-IDF permite a los desarrolladores escribir aplicaciones en C o C++, quienes aprovechan así la potencia y la conectividad de los chips de Espressif [18].

[6]Imagen tomada de https://mytectutor.com/

2.2.2. FreeRTOS

FreeRTOS es un sistema operativo en tiempo real (RTOS) popular y de código abierto. Ofrece un núcleo pequeño y eficiente, apropiado para microcontroladores y sistemas con recursos limitados. Proporciona mecanismos de multitarea, como hilos (tareas), gestión de memoria, sincronización y comunicación entre tareas (semáforos, mutexes, colas). Facilita la organización y la gestión de la ejecución de múltiples funciones de manera concurrente y determinista, crucial para aplicaciones de tiempo real. Su portabilidad permite su uso en una amplia variedad de arquitecturas de procesadores [19].

2.2.3. Ceedling

Ceedling es un framework de construcción y prueba para proyectos de software embebido en C. Automatiza tareas como la compilación, el enlazado y la ejecución de pruebas unitarias. Integra herramientas como Unity, CMock y Ruby. Facilita la adopción de prácticas de desarrollo basadas en pruebas (TDD) y asegura la calidad del código mediante la verificación automatizada de unidades de software individuales [20].

2.2.4. React Native

React Native es un framework de código abierto desarrollado por Meta. Permite la creación de aplicaciones móviles para plataformas iOS y Android desde una única base de código JavaScript. Utiliza los mismos bloques de construcción de la interfaz de usuario que las aplicaciones nativas. Esto resulta en aplicaciones con apariencia y rendimiento nativos [21].

2.3. Protocolos de comunicación empleados

A continuación, se detallan los protocolos de comunicación empleados en la realización del trabajo.

2.3.1. HTTP

HTTP (*Hypertext Transfer Protocol*) es un protocolo de aplicación que define cómo los clientes (navegadores web) solicitan recursos (páginas web, imágenes, etc.) a los servidores y cómo estos responden. Utiliza un modelo de petición-respuesta. Las peticiones HTTP incluyen un método (GET, POST, PUT, DELETE, etc.) que indica la acción que el cliente desea realizar. Las respuestas HTTP contienen un código de estado que informa sobre el resultado de la petición [22].

2.3.2. I2C

I2C es un protocolo de comunicación serial síncrono, multi-maestro/esclavo, de baja velocidad y corta distancia. Utiliza solo dos líneas bidireccionales: SDA (datos seriales) y SCL (reloj serial), ambas conectadas a través de resistencias *pull-up*. Permite que múltiples dispositivos se comuniquen entre sí en el mismo bus. Los maestros inician la comunicación y controlan el reloj, mientras que los esclavos responden a las peticiones de los maestros [23].

2.3.3. 1-Wire

1-Wire es un protocolo de comunicación serial semidúplex. Utiliza un único conductor para la comunicación de datos y, en algunos casos, para la alimentación. Un maestro controla la comunicación con uno o varios dispositivos esclavos en el mismo bus. El protocolo es relativamente lento pero resulta económico para conectar sensores, memorias y otros dispositivos de baja velocidad, especialmente en aplicaciones donde el bajo número de pines es limitado [24].

Capítulo 3

Diseño e implementación

En este capítulo se abordará la descripción de la arquitectura general del sistema, la arquitectura del software, los módulos componentes del software, el desarrollo del software, el diseño del hardware, la selección y la calibración de sensores y el desarrollo de la aplicación movil.

3.1. Diagrama de bloques

En la figura 3.1 se muestra el diagrama en bloques general del sistema donde se describe la arquitectura aplicada al trabajo.

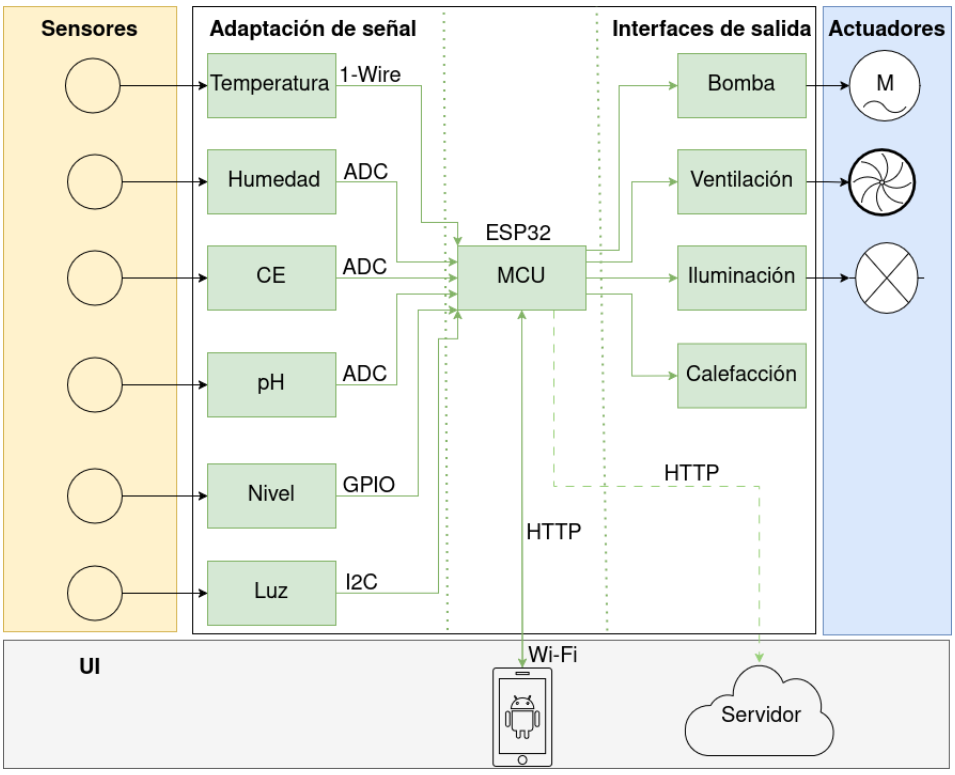


FIGURA 3.1. Diagrama de bloques del sistema.

El sistema embebido implementado en este trabajo consta de una PCB centralizadora, diseñada para integrar y gestionar todos los módulos de hardware. Esta arquitectura asegura la alimentación, adaptación y protección de todos sus componentes. El sistema completo abarca desde la adquisición de datos mediante sensores hasta las acciones sobre el entorno a través de actuadores. Además, se

complementa con una interfaz de usuario móvil para el monitoreo y control remoto.

A continuación, se describen brevemente los bloques y su función.

- Sensores: este bloque se encarga de la transducción de magnitudes físicas en señales eléctricas, lo que permite la digitalización de variables ambientales de interés para el control del cultivo.
- Adaptación de señal: los módulos de adaptación acondicionan las señales provenientes de los sensores. Para garantizar la correcta interpretación por parte de la unidad de microcontrolador o MCU (del inglés *MicroController Unit*), ajustan los niveles de tensión y la relación señal-ruido a valores apropiados.
- MCU: el núcleo del sistema, basado en el chip ESP32, orquesta la comunicación y el control de todos los módulos. Provee la capacidad de procesamiento y la conectividad inalámbrica necesarias para la automatización del cultivo y la interacción con la aplicación móvil.
- Interfaces de salida: proporciona aislamiento galvánico y acondicionamiento de potencia para la activación de los actuadores, lo que asegura la protección del MCU y la correcta operación de los componentes de mayor potencia.
- Actuadores: los actuadores (ventiladores, bomba de irrigación, luces, resistencia calefactora) ejecutan las acciones de control y modifican las condiciones ambientales del cultivo según las necesidades.
- Aplicación móvil de usuario: desarrollada para facilitar la interacción con el sistema, la aplicación móvil permite el monitoreo en tiempo real de las condiciones del cultivo y el control remoto de los actuadores.
- Interfaz de servicio web: se implementó una interfaz para la transmisión de datos a un servicio web externo que permite el almacenamiento y análisis de información del cultivo. Esta funcionalidad se encuentra fuera del alcance principal de este trabajo.

3.2. Arquitectura del firmware

En la presente sección se aborda la arquitectura del firmware del microcontrolador.

3.2.1. Patrones

A continuación, se detallan los patrones de diseño arquitectónico utilizados.

Arquitectura en capas

Se adoptó un patrón de arquitectura en capas para estructurar el software desarrollado, lo que permitió una separación de funcionalidades clara mediante niveles de abstracción. Dicha metodología divide el sistema en niveles horizontales, cada uno con responsabilidades específicas y bien definidas, que facilita el desarrollo, la mantenibilidad y la escalabilidad del código.

A continuación, se enumeran las capas de abstracción que constituyen el firmware.

- Capa de aplicación.
- Capa de sistema operativo.
- Capa de abstracción de hardware (HAL).

Capa de abstracción de hardware

Para facilitar la interacción con los diversos componentes de hardware y garantizar la portabilidad del código, se implementó una capa de abstracción basada en Espressif HAL (*Hardware Abstraction Layer*). Integrada dentro del SDK de ESP-IDF, esta capa proporciona una interfaz uniforme para el control de los periféricos del ESP32, independientemente de las particularidades del hardware subyacente.

Control ambiental

El patrón de control ambiental se adoptó como estrategia arquitectónica para la capa de aplicación. El sistema embebido requirió la monitorización y modificación del entorno mediante sensores y actuadores. Este patrón permitió la estructuración de la lógica de control y facilitó la gestión de las interacciones entre los componentes de hardware y la implementación de los algoritmos de control.

3.2.2. Componentes

La capa de aplicación, figura 3.2, está constituida por los componentes de software encargados de gestionar cada una de las funcionalidades del sistema. A continuación, se describe brevemente la funcionalidad de cada uno de estos componentes.

- Monitor de temperatura: mide la temperatura de la solución nutritiva para garantizar que las plantas crezcan en un entorno térmicamente adecuado.
- Monitor de conductividad eléctrica (CE): evalúa la concentración de nutrientes presentes en la solución hidropónica.
- Monitor de nivel: monitorea que el nivel de solución nutritiva en el sistema no caiga por debajo de un valor determinado.
- Monitor de pH: mide el nivel de acidez de la solución nutritiva, lo que ayuda a mantener el pH dentro del rango óptimo para el cultivo.
- Monitor de humedad del sustrato: mide el contenido de humedad en el sustrato del cultivo.
- Monitor de luz: mide la cantidad de luz disponible en el entorno a través de un sensor.
- Servidor web embebido: el servidor web embebido permite la comunicación con el sistema de control del cultivo a través de la red. Este servidor permite monitorear, configurar y controlar el sistema desde cualquier dispositivo con acceso a la red.

- Control de ciclo de luz: este módulo se encarga de gestionar el ciclo de iluminación del cultivo. La configuración de dicho ciclo se realiza por medio de la interfaz de usuario.
- Control de ciclo de oxigenación/ventilación: este componente activa y desactiva el flujo de aire en el sistema de cultivo según la configuración del usuario.
- Control de hidratación: gestiona el riego, activa la bomba de agua y ajusta los niveles de humedad del sustrato según las mediciones del sensor de humedad. Esto asegura la cantidad adecuada de agua para el desarrollo de las plantas.
- Administrador: es el componente encargado de la centralización y validación global de los datos suministrados por los demás módulos de software.

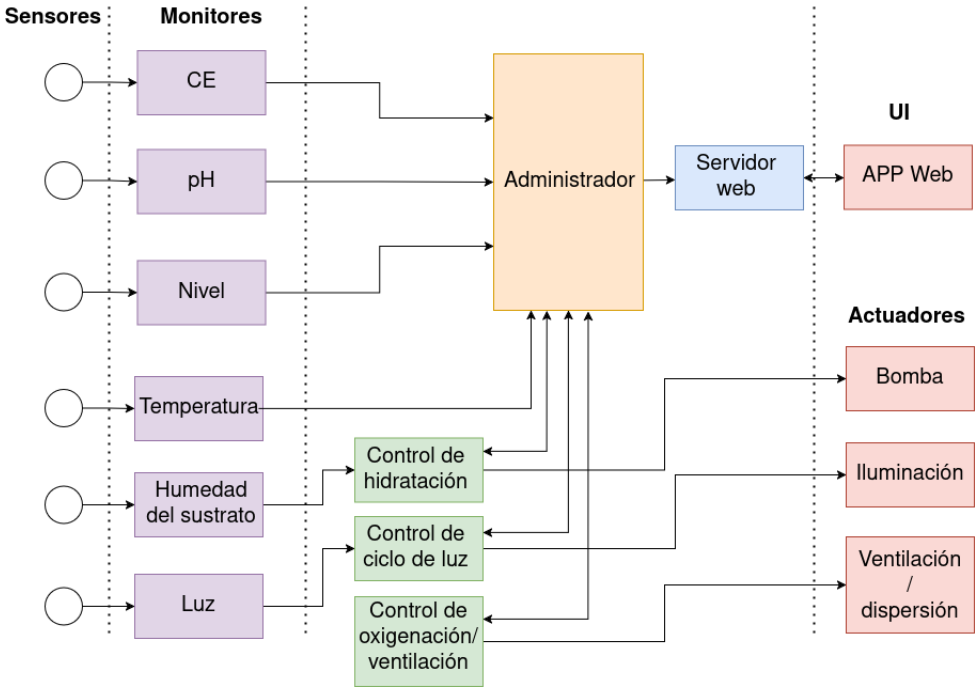


FIGURA 3.2. Diagrama de módulos funcionales y sus interacciones.

3.3. Desarrollo del software

Para facilitar la escalabilidad de la aplicación, se implementó una tarea de FreeRTOS para cada componente del firmware. Esta arquitectura modular permite modificar cada componente de forma independiente, sin afectar al resto del sistema.

3.3.1. Gestión de la comunicación entre tareas

La comunicación entre tareas se gestiona mediante un patrón de publicación/-suscripción con múltiples canales, que utiliza colas de FreeRTOS como buffers para el intercambio de datos. Esta arquitectura desacopla los productores de información (ej., monitores) de los consumidores (ej., controlador, administrador), lo que garantiza la integridad de los datos y la estabilidad del sistema al evitar condiciones de carrera.

3.3.2. Gestión de la prioridad

Durante el desarrollo del firmware, se abordó la organización de las tareas considerando el clásico problema de productores-consumidores. Esta estrategia permitió estructurar el sistema en función del flujo de información entre tareas.

Se definieron los monitores como tareas productoras, encargadas de obtener datos del entorno mediante sensores y publicarlos en estructuras compartidas .

Se implementaron los controladores como tareas consumidoras, cuya función es tomar decisiones en base a los datos recibidos y accionar los actuadores correspondientes.

Se desarrolló una tarea de administrador, también consumidora, responsable de coordinar el funcionamiento general del sistema y gestionar configuraciones.

El servidor embebido fue programado con un doble rol: responde a eventos generados por el usuario (como solicitudes de información o configuraciones) y también produce eventos que otras tareas consumen, como actualizaciones de parámetros.

Se asignaron diferentes niveles de prioridad a las tareas según su disponibilidad requerida. Las tareas de monitoreo se crearon con prioridad intermedia, mientras que las tareas de control y administración se configuraron con mayor prioridad para asegurar tiempos de respuesta adecuados ante eventos prioritarios. El servidor, al depender de la interacción del usuario, fue asignado con prioridad alta.

3.3.3. Controladores de dispositivos

Se desarrollaron controladores de dispositivos (*device drivers*) dedicados a gestionar directamente las unidades de hardware involucradas en el sistema.

Estos controladores permiten abstraer las operaciones de bajo nivel necesarias para interactuar con sensores y actuadores.

3.3.4. Gestión de fallos

El sistema incorpora mecanismos de detección y manejo de fallos tanto a nivel de hardware como de software. Cada tarea cuenta con rutinas de supervisión que permiten identificar comportamientos anómalos o pérdida de respuesta de los dispositivos. En caso de error, se activan procedimientos de contingencia que buscan restablecer el funcionamiento sin comprometer la estabilidad del sistema general.

Por ejemplo, si un sensor deja de enviar datos válidos durante un tiempo determinado, se genera una alerta que es registrada y comunicada al usuario. De igual modo, si un actuador no responde a una orden tras múltiples intentos, se desactiva su uso y se notifica al sistema mediante una bandera de estado.

Además, se implementó un mecanismo de *watchdog* que reinicia automáticamente el microcontrolador en caso de bloqueo o error crítico no recuperable. La configuración base del usuario se almacena en memoria no volátil (NVS), lo que permite que el sistema se recupere de cortes de energía sin perder los parámetros definidos.

3.3.5. Servidor embebido

Para facilitar la interacción con el usuario, el sistema cuenta con un servidor embebido basado en HTTP alojado en la propia ESP32. Este servidor permite exponer una API REST, a través de la cual se puede monitorear el estado del cultivo, consultar los valores de los sensores en tiempo real y ejecutar acciones como el encendido manual de actuadores o la configuración de alertas.

El servidor opera de forma concurrente con el resto de las tareas del sistema gracias a la arquitectura multitarea de FreeRTOS. Para garantizar la seguridad de acceso, se implementó un sistema básico de autenticación por usuario y contraseña que protege las rutas de la API.

El servidor responde a peticiones HTTP provenientes tanto de navegadores como de la aplicación móvil desarrollada en React Native, que primero transmite las credenciales de red por Bluetooth para iniciar la conexión Wi-Fi y luego se comunica mediante la API expuesta.

Este servidor embebido representa un componente clave para el monitoreo remoto y la escalabilidad del sistema, ya que permite una integración sencilla con otras plataformas IoT o bases de datos externas si se desea en el futuro.

3.3.6. Lógica de negocio e interacción con la aplicación móvil

La interacción entre la aplicación móvil y el dispositivo se desarrolla en dos etapas principales: configuración inicial por Bluetooth y comunicación operativa vía HTTP.

Fase de configuración

Al iniciar el sistema por primera vez o tras un reinicio de red, la aplicación móvil se conecta a la ESP32 mediante *Bluetooth Low Energy* (BLE). A través de esta conexión, el usuario ingresa las credenciales de su red Wi-Fi doméstica (SSID y contraseña), que son transmitidas de forma segura al microcontrolador. Una vez recibidos estos datos, el dispositivo intenta conectarse a la red y, de ser exitoso, inicia el servidor embebido.

Fase operativa

Luego de establecer conexión Wi-Fi, la ESP32 expone una API REST que permite a la aplicación móvil interactuar con el sistema. Entre las operaciones más importantes se incluyen:

- Lectura de datos ambientales (temperatura, humedad, pH, EC, etc.).
- Consulta del estado de los actuadores (iluminación, bomba, ventilación).
- Configuración de parámetros personalizados y umbrales de alerta.
- Activación manual o automática de funciones del sistema.
- Acceso a registros de eventos o errores del sistema.

Manejo de estados y sincronización

El firmware mantiene una estructura centralizada de datos en memoria que refleja el estado actual del sistema. Esta información se sincroniza con la app móvil cada vez que se establece una sesión HTTP válida. De este modo, el usuario siempre visualiza información actualizada del estado del cultivo.

Además, se contempló la posibilidad de incorporar una capa adicional de persistencia remota, como sincronización con bases de datos externas o servicios en la nube, lo que permitirá futuras extensiones del sistema hacia modelos más complejos de análisis de datos o inteligencia artificial.

Capítulo 4

Ensayos y resultados

Si en el texto se hace alusión a diferentes partes del trabajo referirse a ellas como capítulo, sección o subsección según corresponda. Por ejemplo: “En el capítulo 1 se explica tal cosa”, o “En la sección **??** se presenta lo que sea”, o “En la subsección 4.0.1 se discute otra cosa”.

Cuando se quiere poner una lista tabulada, se hace así:

- Este es el primer elemento de la lista.
- Este es el segundo elemento de la lista.

Notar el uso de las mayúsculas y el punto al final de cada elemento.

Si se desea poner una lista numerada el formato es este:

1. Este es el primer elemento de la lista.
2. Este es el segundo elemento de la lista.

Notar el uso de las mayúsculas y el punto al final de cada elemento.

4.0.1. Este es el título de una subsección

Se recomienda no utilizar **texto en negritas** en ningún párrafo, ni tampoco texto subrayado. En cambio sí se debe utilizar *texto en itálicas* para palabras en un idioma extranjero, al menos la primera vez que aparecen en el texto. En el caso de palabras que estamos inventando se deben utilizar “comillas”, así como también para citas textuales. Por ejemplo, un *digital filter* es una especie de “selector” que permite separar ciertos componentes armónicos en particular.

La escritura debe ser impersonal. Por ejemplo, no utilizar “el diseño del firmware lo hice de acuerdo con tal principio”, sino “el firmware fue diseñado utilizando tal principio”.

El trabajo es algo que al momento de escribir la memoria se supone que ya está concluido, entonces todo lo que se refiera a hacer el trabajo se narra en tiempo pasado, porque es algo que ya ocurrió. Por ejemplo, "se diseñó el firmware empleando la técnica de test driven development".

En cambio, la memoria es algo que está vivo cada vez que el lector la lee. Por eso transcurre siempre en tiempo presente, como por ejemplo:

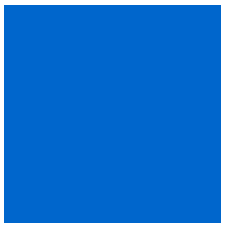
“En el presente capítulo se da una visión global sobre las distintas pruebas realizadas y los resultados obtenidos. Se explica el modo en que fueron llevados a cabo los test unitarios y las pruebas del sistema”.

Se recomienda no utilizar una sección de glosario sino colocar la descripción de las abreviaturas como parte del mismo cuerpo del texto. Por ejemplo, RTOS (*Real Time Operating System*, Sistema Operativo de Tiempo Real) o en caso de considerarlo apropiado mediante notas a pie de página.

Si se desea indicar alguna página web utilizar el siguiente formato de referencias bibliográficas, dónde las referencias se detallan en la sección de bibliografía de la memoria, utilizado el formato establecido por IEEE en [25]. Por ejemplo, "el presente trabajo se basa en la plataforma EDU-CIAA-NXP [26], la cual...".

4.0.2. Figuras

Al insertar figuras en la memoria se deben considerar determinadas pautas. Para empezar, usar siempre tipografía claramente legible. Luego, tener claro que **es incorrecto** escribir por ejemplo esto: "El diseño elegido es un cuadrado, como se ve en la siguiente figura:"



La forma correcta de utilizar una figura es con referencias cruzadas, por ejemplo: "Se eligió utilizar un cuadrado azul para el logo, como puede observarse en la figura 4.1".

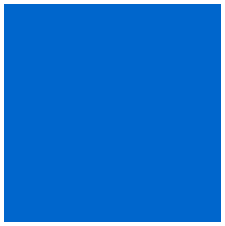


FIGURA 4.1. Ilustración del cuadrado azul que se eligió para el diseño del logo.

El texto de las figuras debe estar siempre en español, excepto que se decida reproducir una figura original tomada de alguna referencia. En ese caso la referencia de la cual se tomó la figura debe ser indicada en el epígrafe de la figura e incluida como una nota al pie, como se ilustra en la figura 4.2.

FIGURA 4.2. Imagen tomada de la página oficial del procesador[1].

La figura y el epígrafe deben conformar una unidad cuyo significado principal pueda ser comprendido por el lector sin necesidad de leer el cuerpo central de la

[1]Imagen tomada de https://goo.gl/images/i7C70w

memoria. Para eso es necesario que el epígrafe sea todo lo detallado que corresponda y si en la figura se utilizan abreviaturas entonces aclarar su significado en el epígrafe o en la misma figura.

FIGURA 4.3. ¿Por qué de pronto aparece esta figura?

Nunca colocar una figura en el documento antes de hacer la primera referencia a ella, como se ilustra con la figura 4.3, porque sino el lector no comprenderá por qué de pronto aparece la figura en el documento, lo que distraerá su atención.

Otra posibilidad es utilizar el entorno *subfigure* para incluir más de una figura, como se puede ver en la figura 4.4. Notar que se pueden referenciar también las figuras internas individualmente de esta manera: 4.4a, 4.4b y 4.4c.

(A) Un caption. (B) Otro. (C) Y otro más.

FIGURA 4.4. Tres gráficos simples.

El código para generar las imágenes se encuentra disponible para su reutilización en el archivo **`Chapter2.tex`**.

4.0.3. Tablas

Para las tablas utilizar el mismo formato que para las figuras, sólo que el epígrafe se debe colocar arriba de la tabla, como se ilustra en la tabla 4.1. Observar que sólo algunas filas van con líneas visibles y notar el uso de las negritas para los encabezados. La referencia se logra utilizando el comando `\ref{<label>}` donde label debe estar definida dentro del entorno de la tabla.

```
\begin{table}[h]
\centering
\caption[caption corto]{caption largo más descriptivo}
\begin{tabular}{l c c}
\toprule
\textbf{Especie}     & \textbf{Tamaño} & \textbf{Valor}\\
\midrule
Amphiprion Ocellaris & 10 cm           & \$ 6.000 \\
Hepatus Blue Tang    & 15 cm           & \$ 7.000 \\
Zebrasoma Xanthurus  & 12 cm           & \$ 6.800 \\
\bottomrule
```

```
\hline
\end{tabular}
\label{tab:peces}
\end{table}
```

TABLA 4.1. caption largo más descriptivo.

Especie	Tamaño	Valor
Amphiprion Ocellaris	10 cm	$ 6.000
Hepatus Blue Tang	15 cm	$ 7.000
Zebrasoma Xanthurus	12 cm	$ 6.800

En cada capítulo se debe reiniciar el número de conteo de las figuras y las tablas, por ejemplo, figura 2.1 o tabla 2.1, pero no se debe reiniciar el conteo en cada sección. Por suerte la plantilla se encarga de esto por nosotros.

4.0.4. Ecuaciones

Al insertar ecuaciones en la memoria dentro de un entorno *equation*, éstas se numeran en forma automática y se pueden referir al igual que como se hace con las figuras y tablas, por ejemplo ver la ecuación 4.1.

$$ds^2 = c^2 dt^2 \left(\frac{d\sigma^2}{1 - k\sigma^2} + \sigma^2 \left[d\theta^2 + \sin^2 \theta d\phi^2 \right] \right) \tag{4.1}$$

Es importante tener presente que si bien las ecuaciones pueden ser referidas por su número, también es correcto utilizar los dos puntos, como por ejemplo "la expresión matemática que describe este comportamiento es la siguiente:"

$$\frac{\hbar^2}{2m}\nabla^2\Psi + V(\mathbf{r})\Psi = -i\hbar\frac{\partial\Psi}{\partial t} \tag{4.2}$$

Para generar la ecuación 4.1 se utilizó el siguiente código:

```
\begin{equation}
\label{eq:metric}
ds^2 = c^2 dt^2 \left( \frac{d\sigma^2}{1-k\sigma^2} +
\sigma^2\left[ d\theta^2 +
\sin^2\theta d\phi^2 \right] \right)
\end{equation}
```

Y para la ecuación 4.2:

```
\begin{equation}
\label{eq:schrodinger}
\frac{\hbar^2}{2m}\nabla^2\Psi + V(\mathbf{r})\Psi =
-i\hbar \frac{\partial\Psi}{\partial t}
\end{equation}
```

4.1. Análisis del software

La idea de esta sección es resaltar los problemas encontrados, los criterios utilizados y la justificación de las decisiones que se hayan tomado.

Se puede agregar código o pseudocódigo dentro de un entorno lstlisting con el siguiente código:

```
\begin{lstlisting}[caption= "un epígrafe descriptivo"]
las líneas de código irían aquí...
\end{lstlisting}
```

A modo de ejemplo:

```
#define MAX_SENSOR_NUMBER 3
#define MAX_ALARM_NUMBER  6
#define MAX_ACTUATOR_NUMBER 6

uint32_t sensorValue[MAX_SENSOR_NUMBER];
FunctionalState alarmControl[MAX_ALARM_NUMBER]; //ENABLE or DISABLE
state_t alarmState[MAX_ALARM_NUMBER];           //ON or OFF
state_t actuatorState[MAX_ACTUATOR_NUMBER];     //ON or OFF

void vControl() {

  initGlobalVariables();

  period = 500 ms;

  while(1) {

    ticks = xTaskGetTickCount();

    updateSensors();

    updateAlarms();

    controlActuators();

    vTaskDelayUntil(&ticks, period);
  }
}
```

CÓDIGO 4.1. Pseudocódigo del lazo principal de control.

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

4.2. Pruebas funcionales del hardware

La idea de esta sección es explicar cómo se hicieron los ensayos, qué resultados se obtuvieron y analizarlos.

Capítulo 5

Conclusiones

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

5.1. Conclusiones generales

La idea de esta sección es resaltar cuáles son los principales aportes del trabajo realizado y cómo se podría continuar. Debe ser especialmente breve y concisa. Es buena idea usar un listado para enumerar los logros obtenidos.

En esta sección no se deben incluir ni tablas ni gráficos.

Algunas preguntas que pueden servir para completar este capítulo:

- ¿Cuál es el grado de cumplimiento de los requerimientos?
- ¿Cuán fielmente se puedo seguir la planificación original (cronograma incluido)?
- ¿Se manifestó algunos de los riesgos identificados en la planificación? ¿Fue efectivo el plan de mitigación? ¿Se debió aplicar alguna otra acción no contemplada previamente?
- Si se debieron hacer modificaciones a lo planificado ¿Cuáles fueron las causas y los efectos?
- ¿Qué técnicas resultaron útiles para el desarrollo del proyecto y cuáles no tanto?

5.2. Próximos pasos

Acá se indica cómo se podría continuar el trabajo más adelante.

Este es un ejemplo de cita de un artículo [27]. Aquí citamos un libro [28] y un capítulo de libro [29]. También podemos citar una página web [30]. O una referencia de un manual técnico [31].

Bibliografía

[1] UN. «Población». En: *www.un.org* (2023). URL: https://www.un.org/es/global-issues/population.

[2] Miguel A. Gómez Carlos Miguel Marschoff Marisa Arienza Andrés E. Carsen Pittaluga. «AGUA». En: *www.greencross.org.ar* (). URL: https://www.greencross.org.ar/download/LibroAguaaimprirweb.pdf.

[3] FAO. «El estado de la seguridad alimentaria y la nutrición en el mundo 2022». En: *fao.org* (2022). URL: https://openknowledge.fao.org/items/ec2794d0-7265-47d8-a9c1-439c7d836a50.

[4] Banco Santander. *Cultivos Verticales: Ventajas y Desventajas*. 2023. URL: https://www.bancosantander.es/blog/pymes-negocios/cultivos-verticales-ventajas-desventajas#:~:text=Los%20cultivos%20verticales%20son%20una,del%20espacio%20y%20la%20luz..

[5] Debajyoti Saha Aditi Saha Roy Saptashree Das y Subhajit Barat. «Vertical farming: A sustainable agriculture format of the future». En: *International Journal of Research in Agronomy* (2024). URL: https://doi.org/10.33545/2618060X.2024.v7.i4Sd.641.

[6] Agroquivir. *Agricultura vertical: Cómo funciona y tipos*. 2023. URL: https://agroquivir.com/agricultura-vertical-como-funciona-y-tipos/.

[7] Clarín. *Se desperdicia el 84 por ciento del agua para riego*. 2024. URL: https://www.clarin.com/sociedad/desperdicia-84-ciento-agua-riego_0_rJxuMJZCtg.html.

[8] Agrointec. *Nidopro - Sistema Hidropónico Inteligente de Cultivo Vertical*. 2025. URL: https://www.agrointec.com/producto/nidopro-sistema-hidroponico-inteligente-cultivo-vertical/.

[9] Techpunt. *Xiaomi Mi Flower Care Plant Sensor*. 2025. URL: https://www.techpunt.nl/en/xiaomi-mi-flower-care-plant-sensor.html.

[10] Konyks. *Hydro Kit*. 2025. URL: https://konyks.com/es/producto/hydro-kit/.

[11] Click and Grow. *The Smart Garden 9 Pro*. 2025. URL: https://www.clickandgrow.com/products/the-smart-garden-9-pro.

[12] Espressif Systems. *ESP32 DevKitC User Guide*. 2025. URL: https://docs.espressif.com/projects/esp-dev-kits/en/latest/esp32/esp32-devkitc/user_guide.html#functional-description.

[13] Think Robotics. *PH4502C PH Meter*. 2025. URL: https://thinkrobotics.com/products/ph4502c-ph-meter?srsltid=AfmBOopTC1VLNuFBrGYsfEB4faApwjy9sQJns4zOBfhpiQRU1.

[14] Ichibot. *TDS Meter V1 Module - Water Meter Filter Measuring Water Quality Sensor*. 2025. URL: https://store.ichibot.id/product/tds-meter-v1-module-water-meter-filter-measuring-water-quality-sensor/.

[15] ProBots. *Soil Moisture Sensor Capacitive V1.2*. 2025. URL: https://probots.co.in/soil-moisture-sensor-capacitive-v1-2.html.

[16] Analog Devices. *DS18B20 Programmable Resolution 1-Wire Digital Thermometer Datasheet*. 2025. URL: https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf.
[17] Mouser Electronics. *BH1750FVI Digital Light Sensor Datasheet*. 2025. URL: https://www.mouser.com/datasheet/2/348/bh1750fvi-e-186247.pdf?srsltid=AfmBOorrJyGddy77cYJODidiGAIv0xVFPwBJiKPiwQfXF0zuK2BGv7UM.
[18] Espressif Systems. *ESP-IDF: Espressif IoT Development Framework*. 2025. URL: https://www.espressif.com/en/products/sdks/esp-idf.
[19] FreeRTOS. *FreeRTOS: Real-Time Operating System*. 2025. URL: https://www.freertos.org/.
[20] Throw The Switch. *Ceedling: A Build System and Test Framework for C*. 2025. URL: https://www.throwtheswitch.org/ceedling.
[21] React Native. *React Native: Learn Once, Write Anywhere*. 2025. URL: https://reactnative.dev/.
[22] Mozilla Developer Network (MDN). *Protocolo de Transferencia de Hipertexto (HTTP)*. 2025. URL: https://developer.mozilla.org/es/docs/Web/HTTP.
[23] Wikipedia. *I²C*. 2025. URL: https://es.wikipedia.org/wiki/I%C2%B2C.
[24] Wikipedia. *1-Wire*. 2025. URL: https://es.wikipedia.org/wiki/1-Wire.
[25] IEEE. *IEEE Citation Reference*. 1.ª ed. IEEE Publications, 2016. URL: http://www.ieee.org/documents/ieeecitationref.pdf (visitado 26-09-2016).
[26] Proyecto CIAA. *Computadora Industrial Abierta Argentina*. Visitado el 2016-06-25. 2014. URL: http://proyecto-ciaa.com.ar/devwiki/doku.php?id=start.
[27] John Doe. «Title». En: *Journal* (2017).
[28] John Doe. *The Book without Title*. Dummy Publisher, 2100.
[29] John Doe. «The Book without Title». En: Dummy Publisher, 2100, págs. 100-200.
[30] Intel. *Example Website*. http://example.com. Dic. de 1988. (Visitado 26-11-2012).
[31] *DS18B20 Programmable Resolution 1-Wire Digital Thermometer*. 050400. Rev. 3. Dallas Semiconductor. 2015.