CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

MEMORIA DEL TRABAJO FINAL

Robot de exploración ambiental

Autor:
Ing. Gonzalo Carreño

Director:
Esp. Ing. Sergio Alberino (UTN.BA)

Jurados:
Nombre del jurado 1 (pertenencia)
Nombre del jurado 2 (pertenencia)
Nombre del jurado 3 (pertenencia)

Este trabajo fue realizado en la Ciudad Autónoma de Buenos Aires, entre marzo de 2023 y noviembre de 2023.

Resumen

El presente trabajo describe un emprendimiento personal en el que se desarrolla un dispositivo robótico de exploración ambiental, controlable a distancia con las funciones básicas de desplazamiento, medición y reporte de parámetros ambientales tales como presión, temperatura, humedad y luminosidad.

El sistema cuenta con una arquitectura base robusta y flexible sobre la cual otros sensores y actuadores pueden ser adaptados para poder interacturar con el medio ambiente, y logra por tanto brindar una solución que ayuda a incrementar la oferta de robots exploradores en la industria argentina pudiendo ser utilizados en casos de uso de IoT, como por ejemplo en aplicaciones de explotación de petróleo, Smart Mining, Smart Farming, etc.

Para su implementación se utilizaron conceptos y herramientas tales como buenas prácticas en el diseño y desarrollo de firmware, la utilización de sistemas operativos de tiempo real como plataforma de ejecución base, protocolos de comunicaciones para sistemas embebidos, y técnicas y frameworks de testing para asegurar la calidad del producto final.

Agradecimientos

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

Índice general

Índice de figuras

Índice de tablas

Dedicado a mi familia y mis amigos.

Capítulo 1

Introducción general

Esta sección presenta la motivación, alcance, objetivos y requerimientos del producto en el marco del estado del arte y su importancia en la industria.

1.1. Motivación

La motivación del presente trabajo fue primeramente volcar y unificar en un emprendimiento personal los conceptos aprendidos en la especialización de Sistemas Embebidos, con una arquitectura robusta que pueda ser extrapolada a otros casos de uso de valor en la industria como por ejemplo la exploración de suelos en el agro, la exploración submarina para la perforación de pozos de petróleo, o los mencionados más adelante en el estado del arte. Por otra parte, se buscó desarrollar un producto que pueda contribuir a aumentar la oferta de dispositivos robóticos exploradores en Argentina.

1.2. Estado del arte

Los robots exploradores son dispositivos robotizados capaces de moverse de forma autónoma, y/o controlados a distancia, que han sido creados con el fin de reconocer y explorar un lugar o entorno donde una persona no pueda o deba acceder ya sea por motivos de capacidad, practicidad o seguridad. Por este motivo, en función de las necesidades de desplazamiento, existen diferentes sistemas de motricidad, como son por ejemplo, los bípedos, cuadrúpedos, con ruedas, tracción oruga, acuáticos/sumergibles, aéreos, etc. En cuanto a la forma de control, los hay manejados por control remoto cableado o inalámbrico, habiendo equipos más sofisticados, que gracias a aplicaciones de Inteligencia Artificial, están preparados para desplazarse y tomar decisiones de forma autónoma. Algunos de los tipos de robots exploradores más conocidos son los espaciales, de minas, de rescate en catástrofes, de tuberías, submarinos, y de suelos.

Tanto en el ámbito académico como en la industria existen trabajos, proyectos, e implementaciones comerciales similares al presente trabajo, como por ejemplo:

- El prototipo robótico de exploración minera publicado en varios artículos [1], [2], e impulsado por el Instituto de Automática de la Facultad de Ingeniería de la Universidad Nacional de San Juan en el marco de un convenio con la Comisión Nacional de Energía Atómica y el Gobierno argentino [3].
- El robot de exploración terrestre denominado Geobot [4] desarrollado por los ingenieros Nelson Dario García Hurtado y Melvin Andrés González

Pino, de la universidad de Pamplona, capaz de realizar reconocimiento de zonas y manipulación de muestras de manera autónoma o asistida.

- El robot minero MIN-SIS 1.0 SDG-STR [5] desarrollado por los ingenieros Hernán L. Helguero Velásquez y Rubén Medinaceli Tórrez de la Universidad Técnica de Oruro, capaz de detectar gases, almacenar datos locales y enviar video e imágenes al puesto de mando.
- Spot [6], desarrollado por Boston Dynamics, un robot explorador cuadrupedo de propósito general capaz de explorar, almacenar y enviar información en tiempo real.
- BIKE [7], desarrollado por Waygate Technologies, un robot con ruedas magnéticas, muy utilizado en la industria de petróleo y gas entre otras, capaz de desplazarse por el interior de tuberías para poder realizar inspecciones y comunicar hallazgos.

1.3. Alcance y objetivos

A continuación se detallan las funcionalidades incluidas en el alcance del trabajo.

- Sistema de desplazamiento terrestre.
- Operaciones de exploración
  - Medición de humedad ambiental.
  - Medición de temperatura ambiental.
  - Medición de presión ambiental.
  - Medición de luminosidad ambiental.
- Visualización de estado de exploración (lecturas de los sensores).
- Sistema de control por medio de un joystick cableado.

Queda fuera del alcance:

- Locomoción por cualquier otro medio que no sea terrestre.
- Cualquier otra función no contemplada en este alcance.

1.4. Requerimientos

A continuación se listan los requerimientos del producto:

1. Requerimientos funcionales
   a) El sistema debe contar con funciones de desplazamiento para poder moverse hacia adelante y atrás, y poder girar radialmente un ángulo de 360 grados.
   b) El sistema debe ser capaz de realizar las siguientes operaciones de exploración:
      - Medición de humedad ambiental.
      - Medición de temperatura ambiental.

- Medición de luminosidad ambiental.
- Medición de presión ambiental.

c) El sistema debe poder ser controlado a distancia mediante un joystick para que el dispositivo pueda realizar sus movimientos. En caso de que alguna de sus operaciones de exploración requiera algún otro mecanismo de control, el mismo también será integrado en el joystick.

d) El sistema debe proveer un mecanismo de visualización de las operaciones de exploración al usuario que controla el dispositivo para poder ver el estado y lectura de las operaciones de exploración.

- La interfaz de usuario debe permitir visualizar las lecturas de cada uno de los sensores.
- Debe haber una pequeña leyenda de la magnitud que se está midiendo y la unidad utilizada junto con el valor.

2. Requerimientos no funcionales

a) La arquitectura del producto debe ser robusta y tolerante a fallas.

b) A fin de maximizar la mantenibilidad, la arquitectura del producto debe estar modularizada para permitir que los diferentes módulos puedan ser integrados y orquestados separadamente.

Capítulo 2

Introducción específica

Esta sección presenta una breve introducción técnica a las herramientas hardware y software utilizadas en el trabajo.

2.1. Tecnologías de hardware utilizadas

2.1.1. Espressif ESP32

ESP32 [8] es una serie de microcontroladores embebidos en un chip con Wi-Fi y Bluetooth integrados, de bajo costo y consumo, desarrollado por *Espressif Systems*. Emplea dos cores Xtensa® 32-bit LX6 CPU, incluye interruptores de antena, amplificador de potencia, amplificador de recepción de bajo ruido, un co-procesador ULP (Ultra Low Power), módulos de administración de energía y varios periféricos. En la siguiente imagen (2.1) se puede apreciar la placa ESP32-WROOM-32D [9] utilizada para el desarrollo del presente trabajo.

FIGURA 2.1. Microcontrolador ESP32-WROOM-32D.

2.1.2. Sensor de temperatura y humedad DHT11

El DHT11 [10] es un sensor digital de temperatura y humedad relativa de bajo costo y fácil uso. Integra un sensor capacitivo de humedad y un termistor para medir el aire circundante, y muestra los datos mediante una señal digital en el pin de datos (no posee salida analógica). Entre otras aplicaciones se lo suele utilizar principalmente en aplicaciones relacionadas al control automático de temperatura, aire acondicionado y monitoreo ambiental en agricultura. En la figura 2.2 se puede apreciar una imagen del componente.

FIGURA 2.2. Sensor DHT11.

2.1.3. Sensor de presión BMP280

El BMP280 [11] es un sensor de presión barométrica absoluta, especialmente factible para aplicaciones móviles que puede ser utilizado con I2C o SPI. Permite alta precisión y linealidad, estabilidad a largo plazo, alta robustez a un muy bajo consumo. En la figura 2.3 se puede apreciar una imagen del componente.

FIGURA 2.3. Sensor BMP280.

2.1.4. Fotoresistor como sensor de luminosidad

El fotoresistor es una resistencia eléctrica que varía su valor en función de la cantidad de luz que incide sobre su superficie. Cuando el fotoresistor no está expuesto a radiaciones luminosas, los electrones están firmemente unidos en los átomos que lo conforman, por lo que alcanza su máxima resistencia eléctrica, y cuando sobre él inciden radiaciones luminosas, esta energía libera los electrones con lo cual el material se vuelve más conductor, y se disminuye su resistencia. En la figura 2.4 se puede apreciar una imagen del componente.

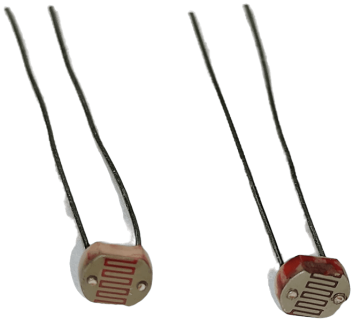

FIGURA 2.4. Fotoresistor.

2.1.5. Joystick analógico

El módulo de joystick analógico [12] está construido sobre el montaje de dos potenciómetros en un ángulo de 90 grados. Los potenciómetros están conectados a una palanca corta centrada por resortes. Este módulo produce una salida de alrededor de 2,5 Volts cuando la palanca se encuentra en reposo (en el centro), mientras que al desplazarse hará que la salida varíe de 0 a 5 Volts dependiendo de su posición. La obtención de los valores en Volts se obtiene tras convertir las lecturas en niveles lógicos mediante el módulo ADC (conversor analógico digital) [13] del microcontrolador ESP32. En la figura 2.5 se puede apreciar una imagen del componente.

FIGURA 2.5. Joystick analógico.

2.1.6. Display LCM1602A

El display LCM1602A [14] consta de una pantalla de cristal líquido que permite representar dos filas con hasta 16 caracteres alfanuméricos en cada una y dado que se encuentra integrada a una interfaz adaptadora I2C puede ser controlada por este protocolo. En la figura 2.6 se puede apreciar una imagen del componente.

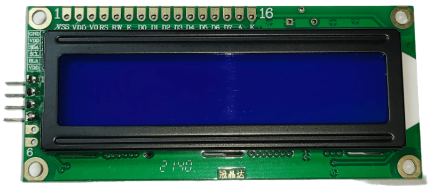

FIGURA 2.6. Display LCM1602A.

2.1.7. Motores de corriente continua

El motor DC (corriente continua) [15] es un electromotor que transforma energía eléctrica en energía mecánica. Estos motores operan con un voltaje entre 3 y 6 Volts, corriente de 150 mA, permiten una velocidad de entre 90 y 200 RPM y un torque de entre 0,15 Nm y 0,60 Nm. En la figura 2.7 se puede apreciar una imagen del componente.

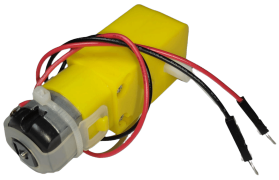

FIGURA 2.7. Motor de corriente continua.

2.1.8. Modulos L298N

...

2.1.9. Baterias de Li-Ion 3.7 V

...

2.1.10. Baterias AA 1.5 V

...

2.1.11. Plaquetas genericas

...

2.1.12. Cables de conexion DuPont

...

2.1.13. Board Pin Headers para montaje de componentes y cables

...

2.1.14. Interruptores de On-Off

...

2.1.15. Porta-pilas de On-Off

...

2.1.16. Ruedas

...

2.2. Tecnologías de software utilizadas

2.2.1. Marco de trabajo ESP-IDF

Espressif Systems proporciona recursos básicos de hardware y software para ayudar a los desarrolladores de aplicaciones a realizar sus ideas utilizando el hardware de la serie ESP32. El framework de software de Espressif está destinado al desarrollo de aplicaciones de IoT (Internet de las cosas) con Wi-Fi, Bluetooth, administración de energía y varias otras características del sistema. Sus componentes son:

1. Toolchain, utilizado para compilar el código para ESP32.
2. Build tools, que provee utilidades como CMake [16] y Ninja [17] para construir la aplicación completa para ESP32.
3. ESP-IDF [18], que brinda la API de desarrollo para ESP32 y scripts para ejecutar Toolchain.

Además de las herramientas mencionadas se utilizó el conjunto de bibliotecas y drivers provistos por el proyecto ESP-IDF-Lib [19] basados en el framework ESP-IDF.

En la figura 2.8 se puede apreciar una imagen del proceso de desarrollo y despliegue usando el framework ESP-IDF.

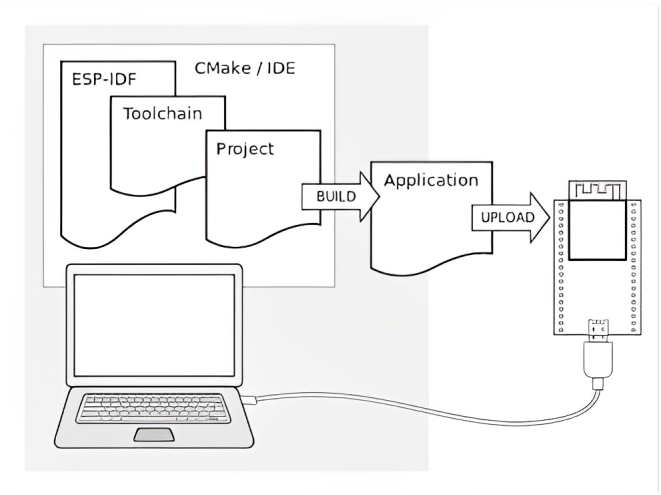

FIGURA 2.8. Proceso de desarrollo utilizando ESP-IDF[1].

2.2.2. Plataforma Docker

Docker [21] es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos. Docker utiliza características de aislamiento de recursos del kernel Linux, tales como cgroups y espacios de nombres (namespaces) para permitir que contenedores livianos independientes se ejecuten en paralelo de manera aislada evitando la sobrecarga de iniciar y mantener máquinas virtuales.

2.2.3. Visual Studio Code

Visual Studio Code [22] es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

2.2.4. Sistema operativo Ubuntu

Ubuntu [23] es una distribución Linux basada en Debian GNU/Linux y patrocinado por Canonical, que incluye principalmente software libre y de código abierto. Puede utilizarse en ordenadores y servidores, está orientado al usuario promedio, con un fuerte enfoque en la facilidad de uso y en mejorar la experiencia del usuario.

[1]Imagen tomada de [20]

Capítulo 3

Diseño e implementación

Esta sección presenta los detalles técnicos del diseño e implementación de las diferentes funcionalidades del producto, la arquitectura hardware y software, y finalmente la interfaz de usuario para el control y reporte de las operaciones del robot.

El proyecto fue realizado siguiendo una metodologia basada en crear un prototipo funcional con una arquitectura escalable que cumpla con alcance basico especificado en el plan de proyecto, y una vez conseguido agregar en lo posible y de acuerdo a la capacidad, funcionalidades adicionales. De esta forma, una vez que se logro el alcance basico, considerado como la version v1.0 del producto, se continuo con el desarrollo y se agrego la funcionalidad adicional de control inalambrico por medio de un joystick, en lo que se considera la version v2.0 del mismo.

En las siguientes sub secciones se detallan los detalles del disenio e implementacion realizados para ambas versiones.

3.1. Arquitectura de software del sistema

Con el fin de poder lograr una arquitectura de software modular, se implementaron diferentes componentes software y servicios que abstraen el acceso a los modulos de hardware (explicados en la siguiente seccion) y permitieron un escalamiento facil de funcionalidades en la expancion de las funcionalidades del robot entre su version v1.0 y la v2.0.

3.1.1. Arquitectura y disenio de componentes en la version v1.0

A continuacion podemos apreciar la arquitectura de software del sistema en su version v1.0 y el detalle de sus componentes y servicios.

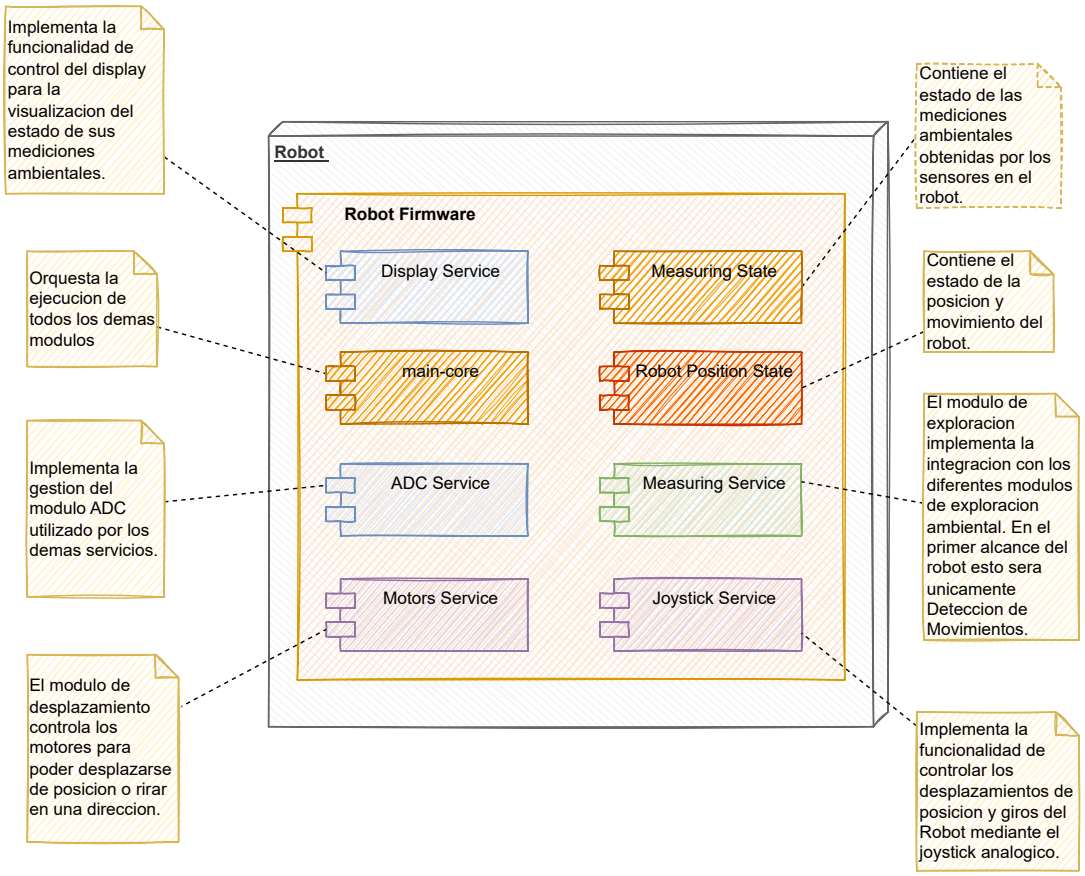

FIGURA 3.1. Arquitectura global.

Los componentes y servicios de software del Robot en la version v1 son los siguientes:

1. Componentes de software
   a) main-core: orquesta las diferentes tareas desde las que se invocan los demas componentes de software.
   b) Measuring Service: abstrae el acceso a los modulos de medicion de temperatura, humedad, presion y luminosidad.
   c) Measuring State: mantiene el estado de cada uno de los parametros ambientales medidos.
   d) Robot Position State: mantiene el estado del movimiento actual del robot.
   e) ADC Service: abstrae el acceso a los modulos que hacen uso de los canales que requieren conversiones ADC (Analogico/Digital), como el joystick analogico y el fotoresistor.
   f) Motors Service: abstrae el acceso al modulo de control de motores.
   g) Display Service: abstrae el acceso al modulo de control del display mediante Ic2.
   h) Joystick Service: abstrae el acceso al modulo de control del joystick.
2. Servicios (tareas)

a) Display Task

b) Measuring Task

c) Joystick Task

d) Motors Task

3.1.2. Arquitectura y disenio de componentes en la version v2.0

Luego de expandir su funcionalidad, los componentes y servicios de software se separaron fisicamente en el hardware del robot y el del joystick, agregandose ademas los necesarios para la comunicacion inalambrica. A continuacion podemos apreciar la arquitectura de software del sistema en su version v2.0 y el detalle de sus componentes y servicios.

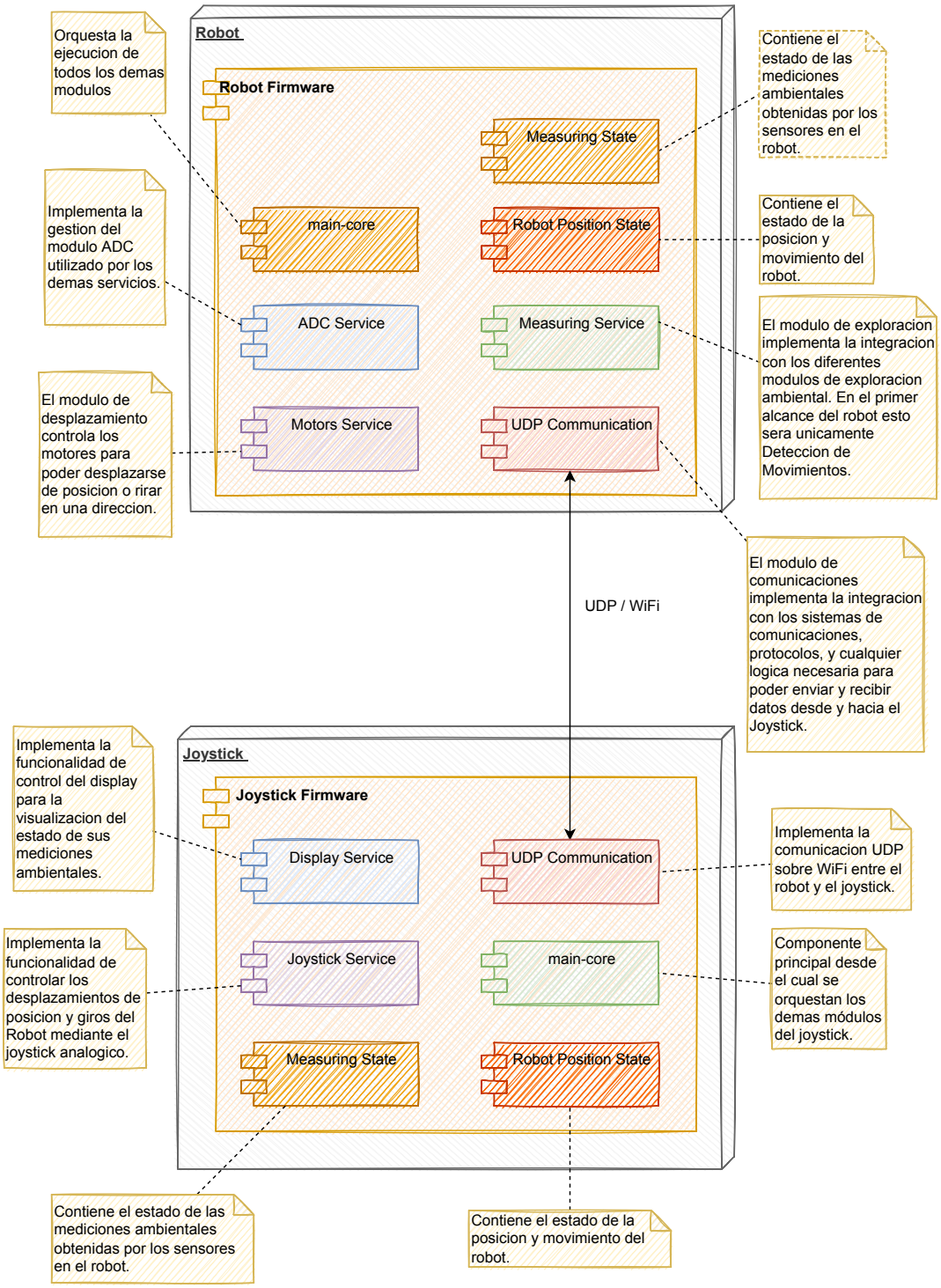

FIGURA 3.2. Arquitectura global.

Los componentes de software y servicios del robot son:

1. Componentes de software
   a) main-core
   b) Measuring Service
   c) Measuring State

   d) Robot Position State
   e) ADC Service
   f) Motors Service
   g) WIFI Service
   h) UDP Service
2. Servicios (tareas)
   a) Measuring Task
   b) Motors Task
   c) UDP Server Task

Los componentes de software y servicios del joystick son:

1. Componentes de software
   a) main-core
   b) Measuring State
   c) Robot Position State
   d) UDP Service
2. Servicios (tareas)
   a) Display Task
   b) Joystick Task
   c) UDP Server Task

3.2. Implementacion de los modulos

Los macro componentes de hardware presentes en la arquitectura de la version v2.0 son el robot y el joystick, los cuales constituyen dos sistemas embebidos independientes implementados con dos microcontroladores ESP32, integrados entre si por medio de una red WiFi y una comunicacion UDP. Los modulos hardware desarrollados son los siguientes.La arquitectura de hardware del sistema esta compuesta por los siguientes modulos:

- En el robot:
  - Control de los motores DC.
  - Control de los sensores de medicion (DHT11, BMP280 y fotoresistor).
  - Gestion de la comunicacion inalambrica via WIFI (en la version v2.0).
- En el joystick:
  - Control del display.
  - Control del joystick.
  - Gestion de la comunicacion inalambrica via WIFI (en la version v2.0).

La integracion de los mismos se realizo mediante el disenio y contruccion de una placa integradora central, que conecta los dispositivos hardware con el microcontrolador ESP-32.

3.2.1. Control de la red WIFI

El modulo de red WiFi esta integrado en el chip ESP32 en el cual se soportan multiples features [24] por lo cual a nivel hardware no fue necesario realizar ningun conexionado. A nivel software, la gestion del modulo WiFi se encuentra incluida en el SDK ESP-IDF, y el acceso al mismo se realiza desde el modulo ADC 2 [13]. Por este motivo, cuando el sistema embebido utiliza el modulo WiFi, el uso del ADC2 queda restringido a esta funcionalidad por lo cual cualquier otro dispositivo que deba hacer uso del ADC debe ser configurado para utilizar el ADC1, como por ejemplo los modulos de joystick y deteccion de luminosidad, explicados en las siguientes secciones.

La configuracion del *soft access point WiFi* implementado se realizo en base a los parametros de red detallados en la siguiente tabla 3.1:

TABLA 3.1. Configuracion de AP WiFi

Parametro	**Valor**
SSID	Robot
password	Robot

El desarrollo de este modulo se baso en el ejemplo provisto por Espressif [25]. El codigo fuente del prototipo realizado puede apreciarse en el siguiente enlace [26].

3.2.2. Control del joystick analogico

Para el desarrollo de este prototipo se utilizó el SDK ESP-IDF y se configuro el modulo ADC1 de acuerdo a los siguientes detalles en la tabla 3.3:

TABLA 3.2. Conexionado joystick

Channel	**Unit**	**Pin GPIO**
6	1	34
7	1	35

A continuación se puede apreciar el conexionado fisico en la figura 3.3.

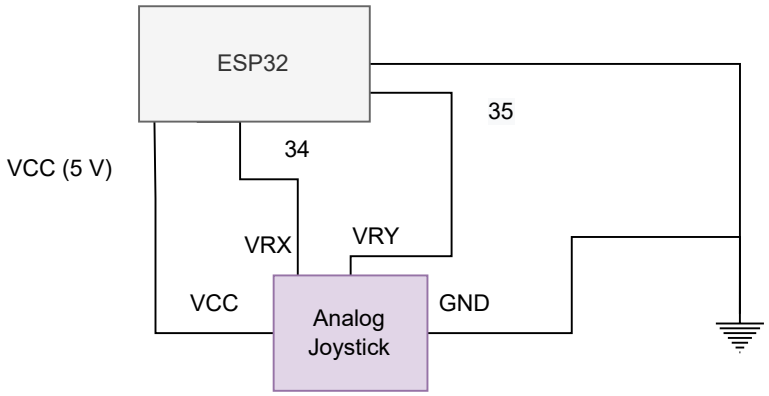

FIGURA 3.3. Conexionado joystick.

El desarrollo de este modulo se baso en el ejemplo provisto por Espressif [27]. El codigo fuente del prototipo realizado puede apreciarse en el siguiente enlace [28].

3.2.3. Medición de valor de luminosidad

Para el desarrollo de este prototipo se utilizó el SDK de ESP-IDF y se configuro el modulo ADC1 de acuerdo a los detalles provistos en la siguiente tabla 3.3

TABLA 3.3. Conexionado fotoresistor

Channel	Unit	Pin GPIO
0	1	36

A continuacion se puede apreciar un diagrama de su conexionado fisico en la figura 3.4.

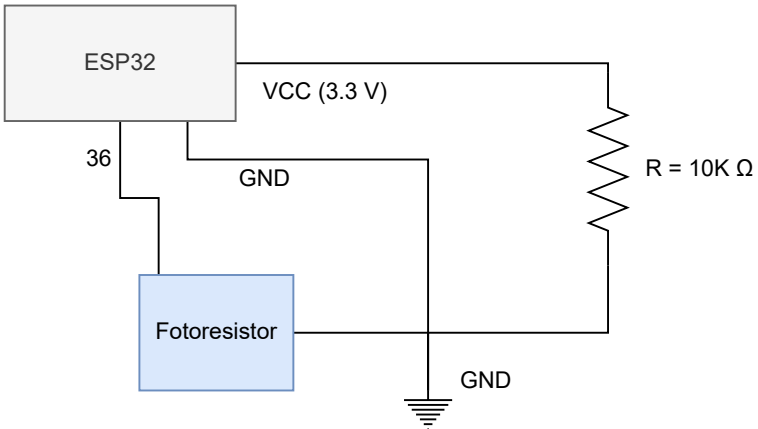

FIGURA 3.4. Conexionado fotorresistor.

El desarrollo de este modulo se baso en el ejemplo provisto por Espressif [27]. El codigo fuente del prototipo realizado puede apreciarse en el siguiente enlace [29].

3.2.4. Medición de temperatura y humedad

Para el desarrollo de este modulo se utilizó la biblioteca de código ESP-IDF-Lib Components Library [19] que provee el soporte para gestionar el DHT11. Para acceder a las lecturas del dispositivo se abstrajo el mismo mediante el componente

Measuring Service, el cual es invocado por la tarea Measuring Task y el estado de la lectura es almacenado en el componente Measuring State. A continuación se puede apreciar el conexionado del prototipo en la figura 3.5.

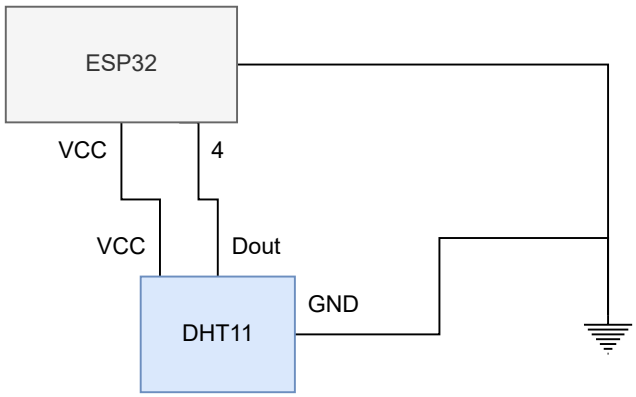

FIGURA 3.5. Circuito del conexionado DHT11.

El desarrollo de este modulo se baso en el ejemplo provisto por la libreria ESP-IDF-Lib [30]. El codigo fuente del prototipo realizado puede apreciarse en el siguiente enlace [31].

3.2.5. Medición de presión

Para el desarrollo de este modulo se utilizó el framework ESP-IDF y la biblioteca de código ESP-IDF Components que provee el soporte para gestionar el dispositivo BMP280 por medio del protocolo I2C. El driver es inicializado en el componente main-core para ser posteriormente invocado desde el Measuring Service en la tarea Measuring Task. Sus lecturas son guardadas en el Measuring State. A continuación se puede apreciar el conexionado del prototipo en la figura 3.6.

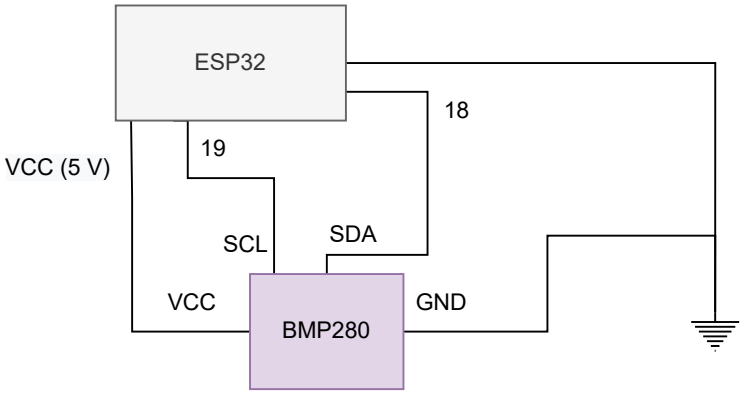

FIGURA 3.6. Conexionado BMP280.

El desarrollo de este modulo se baso en el ejemplo provisto por la libreria ESP-IDF-Lib [32]. El codigo fuente del prototipo realizado puede apreciarse en el siguiente enlace [33].

3.2.6. Control de motores DC

Para la implementacion del modulo de control de los motores de corriente continua se utilizaron a nivel hardware dos modulos puentes L298N [34] que permiten integrar y controlar dos motores cada uno. Con el fin de alimentar los modulos

con una fuente de poder de amperaje y voltaje consistente se utilizaron dos baterias de Li-Ion de 3.7 V y 2000 mA/h conectadas en serie activadas mediante un interruptor. A nivel driver en el ESP32 se utilizo el modulo de control de motores por modulacion de pulsos (MCPWM) [35] que por medio de la configuracion de sus unidades y del duty cycle [36] se puede controlar el sentido y velocidad de rotacion de los motores. Los puentes L298N brindan ademas una tension de salida de 5 V, y esto fue utilizado para la alimentacion del ESP32 conectandolo su pin Vin.

En el siguiente diagrama puede apreciarse el conexionado logico para el control de los motores con los componentes mencionados.

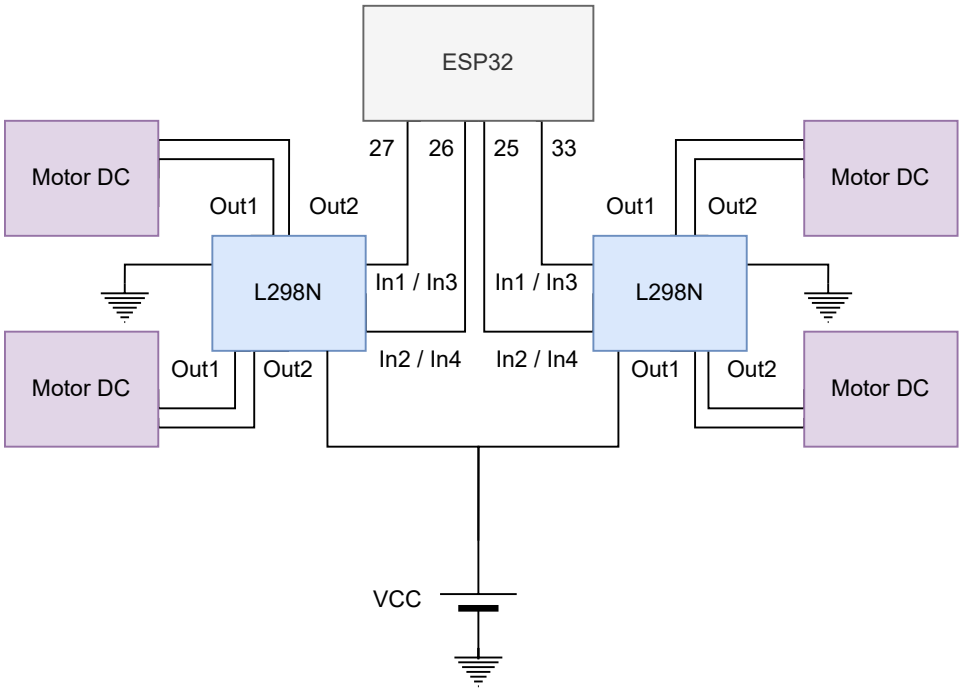

FIGURA 3.7. Conexionado motores.

El desarrollo de este modulo se baso en el ejemplo provisto por Espressif [37]. El codigo fuente del prototipo realizado puede apreciarse en el siguiente enlace [38].

3.2.7. Control del display

Para la implementacion del modulo de visualizacion de valores observados se integro un display de dos lineas y diesiseis caracteres LCM1602A por medio de un driver I2C que facilita su control. Al basarse en el protocolo I2C el display comparte las mismas lineas SCL y SDA que el sensor BMP280.

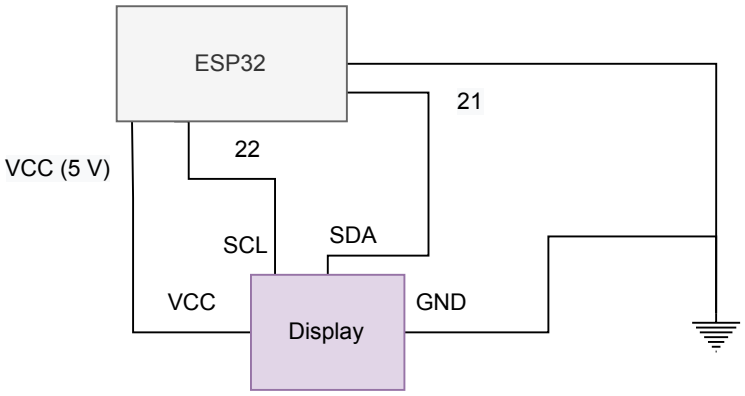

FIGURA 3.8. Conexionado display.

El desarrollo de este modulo se baso en el ejemplo provisto en el enlace [39]. El codigo fuente del prototipo realizado puede apreciarse en el siguiente enlace [40].

3.3. Arquitectura de hardware

3.3.1. Conexionado logico

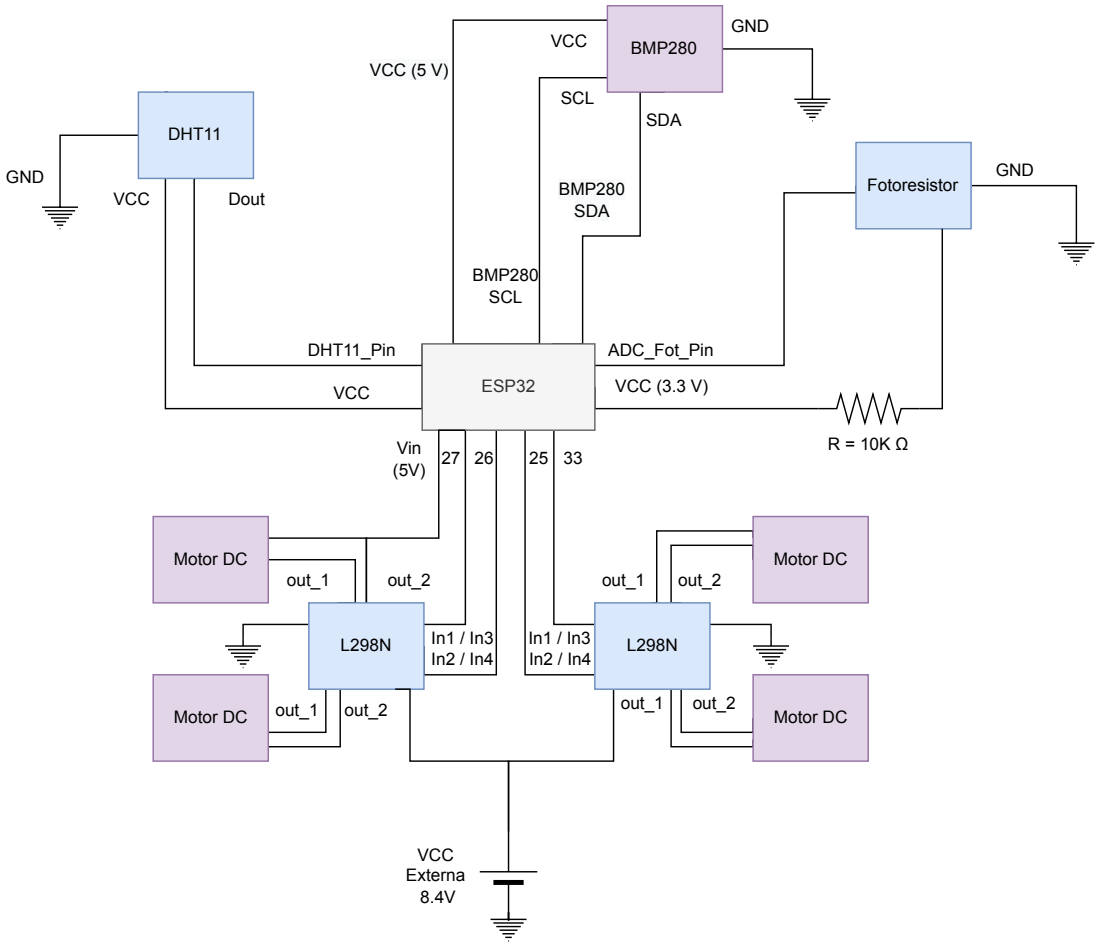

FIGURA 3.9. Conexionado del robot.

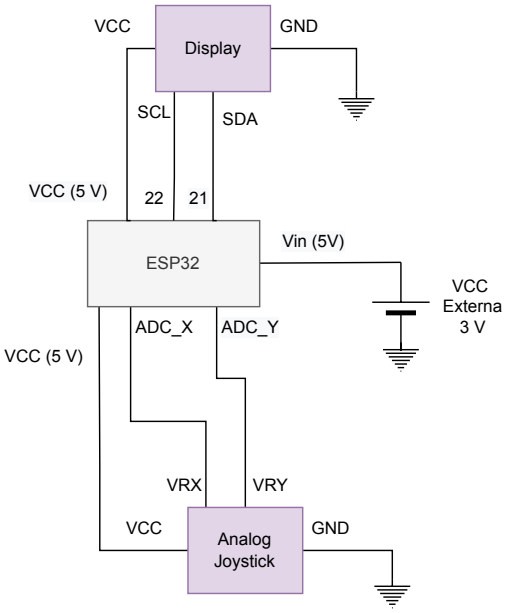

FIGURA 3.10. Conexionado del joystick.

3.3.2. Conexionado fisico

Capítulo 4

Ensayos y resultados

Esta sección presenta los diferentes prototipos realizados para determinar la viabilidad de cada una de las funcionalidades provistas, la metodología de desarrollo, testing, y finalmente los entregables finales del trabajo.

4.1. Proceso de desarrollo y aseguramiento de calidad

Para el proceso de desarrollo se utilizó TDD (Test Driven Development) con CMock, Ceedling y Unity como frameworks para abordar esto. Para la construcción de los casos de pruebas se utilizó Control Flow Test como técnica de caja blanca a fin de diseñar los tests unitarios y de integración. Adicionalmente, se configuró el entorno de desarrollo basado en Docker, por lo que se extiende la imagen de Espressif y se agregó el conjunto de utilidades ESP-IDF Components como parte de la misma. Luego de versionar esta imagen en el *Docker Registry* utilizado se configuró el servidor de integración continua para ejecutar las compilaciones usando esta imagen por cada push realizado. En la siguiente imagen se puede apreciar la infraestructura implementada para abordar el ciclo de desarrollo y despliegue.

[pendiente imagen]

4.2. Prototipos de los diferentes modulos

Se suigio una metodologia basada en lograr prototipar las funcionalidades de los diferentes modulos independientes para luego integrarlas todas juntas en un prototipo integrador. En las siguientes secciones se explican los diferentes prototipos realizados.

- Prototipo de funcionalidad de medición de temperatura y humedad.
- Prototipo de funcionalidad de medición de presión.
- Prototipo de funcionalidad de medición de valor de luminosidad.
- Prototipo de funcionalidad de obtencion de valores analogicos del joystick.
- Prototipo de funcionalidad de presentación de display.
- Prototipo de funcionalidad de control de motores DC.

4.3. Tests de los diferentes módulos

...

4.4. Tests del producto final

...

4.5. Reportes de testing

...

4.6. Verificacion y validacion del producto

...

4.7. Documentacion del producto

...

Capítulo 5

Conclusiones

Conclusiones del trabajo...

- ¿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?

5.1. Próximos pasos

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

Bibliografía

[1] Latam Mining. *Robots y minería: Gobierno argentino quiere implementarlos*. URL: https://www.latam-mining.com/robots-y-mineria-gobierno-argentino-quiere-implementarlos/.

[2] Diario de Cuyo. *Gobierno pone la mira en el desarrollo de robots para la actividad minera*. URL: https://www.diariodecuyo.com.ar/politica/Gobierno-pone-la-mira-en-el-desarrollo-de-robots-para-la-actividad-minera-20200202-0052.html.

[3] Universidad Nacional de San Juan. *Robots en la minería*. URL: http://www.unsj.edu.ar/home/noticias_detalles/4810/1.

[4] Ing. Nelson Dario García Hurtado e Ing. Melvin Andrés González Pino. *Robot de exploración terrestre Geobot*. URL: https://www.unipamplona.edu.co/unipamplona/portalIG/home_40/recursos/01_general/revista_1/09102011/v01_09.pdf.

[5] Ing. Hernán L. Helguero Velásquez1 e Ing. Rubén Medinaceli Tórrez. *Robot Minero: Sistema Detector de Gases utilizando Sensores en Tiempo Real MIN – SIS 1.0 SDG-STR*. URL: http://www.scielo.org.bo/scielo.php?script=sci_arttext&pid=S2519-53522020000100003.

[6] Boston Dynamics. *Spot*. URL: https://www.bostondynamics.com/products/spot.

[7] Waygate Technologies. *BIKE - An advanced crawler robot for remote visual inspection*. URL: https://www.bakerhughes.com/waygate-technologies/robotic-inspection/bike.

[8] Espressif. *ESP32*. URL: https://www.espressif.com/en/products/socs/esp32.

[9] Espressif. *ESP32-WROOM-32D Datasheet*. URL: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf.

[10] Mouser. *DHT11 datasheet*. URL: https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf.

[11] Bosch. *BMP280 datasheet*. URL: https://cdn-shop.adafruit.com/datasheets/BST-BMP280-DS001-11.pdf.

[12] Handson Technology. *PS2 Joy Stick for Arduino/Raspberry*. URL: http://www.handsontec.com/dataspecs/accessory/PS2-Joystick.pdf.

[13] Espressif. *Analog to Digital Converter (ADC)*. URL: https://docs.espressif.com/projects/esp-idf/en/v4.4/esp32/api-reference/peripherals/adc.html.

[14] Handson Technology. *I2C Serial Interface 1602 LCD Module*. URL: http://www.handsontec.com/dataspecs/module/I2C_1602_LCD.pdf.

[15] Adafruit. *DC Gearbox Motor - TT Motor -200RPM - 3 to 6VDC*. URL: https://media.digikey.com/pdf/Data%20Sheets/Adafruit%20PDFs/3777_Web.pdf.

[16] CMake. *CMake*. URL: https://cmake.org/.
[17] CMake. *Ninja*. URL: https://cmake.org/cmake/help/latest/generator/Ninja.html.
[18] Espressif. *ESP-IDF Programming Guide | Get Started*. URL: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/.
[19] Readthedocs by Ruslan V. Uss. *ESP-IDF Components library*. URL: https://esp-idf-lib.readthedocs.io/en/latest/.
[20] Espressif Programing Guide. *ESP-IDF Get Started*. URL: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html.
[21] Docker. *Docker*. URL: https://docker.com/.
[22] Visualstudio. *Visualstudio Code*. URL: https://code.visualstudio.com/.
[23] Ubuntu. *Ubuntu*. URL: https://ubuntu.com/.
[24] Espressif. *ESP-IDF WiFi*. URL: https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-guides/wifi.html.
[25] Espressif. *ESP-IDF - Wi-Fi SoftAP Example*. URL: https://github.com/espressif/esp-idf/tree/v4.4/examples/wifi/getting_started/softAP.
[26] Gonzalo Carreno. *POC ESP32-WiFi v4.4*. URL: https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-wifi-ap-v4.4.
[27] Espressif. *ESP-IDF ADC1 Example*. URL: https://github.com/espressif/esp-idf/tree/v4.0.3/examples/peripherals/adc.
[28] Gonzalo Carreno. *POC ESP32-joystick*. URL: https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-joystick.
[29] Gonzalo Carreno. *POC ESP32-photoresistor*. URL: https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-photoresistor.
[30] UncleRus. *ESP32 - Example for dht driver*. URL: https://github.com/UncleRus/esp-idf-lib/tree/master/examples/dht/default.
[31] Gonzalo Carreno. *POC ESP32-DHT11*. URL: https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-dht11.
[32] UncleRus. *ESP32 - Example for bmp280 driver*. URL: https://github.com/UncleRus/esp-idf-lib/tree/master/examples/bmp280/default.
[33] Gonzalo Carreno. *POC ESP32-BMP280*. URL: https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-bmp280.
[34] Espressif. *L298N Dual H-Bridge Motor Driver*. URL: https://www.handsontec.com/dataspecs/L298N%20Motor%20Driver.pdf.
[35] Espressif. *ESP-IDF Motor Control Pulse Width Modulator (MCPWM)*. URL: https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/peripherals/mcpwm.html.
[36] Espressif. *MCPWM*. URL: https://docs.espressif.com/projects/esp-idf/en/v4.2/esp32/api-reference/peripherals/mcpwm.html.
[37] Espressif. *MCPWM brushed dc motor control Example*. URL: https://github.com/espressif/esp-idf/tree/v4.2/examples/peripherals/mcpwm/mcpwm_brushed_dc_control.

[38] Gonzalo Carreno. *POC ESP32-motor-pwm*. URL: https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-motor-pwm.

[39] ESP32 Tutoriales. *ESP32 I2C LCD with ESP-IDF*. URL: https://esp32tutorials.com/i2c-lcd-esp32-esp-idf/.

[40] Gonzalo Carreno. *POC ESP32-DHT11*. URL: https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-display.