Índice general

Resumen I

1. Introducción general 1
1.1. Motivación . . . 1
1.2. Estado del arte . . . 1
1.3. Alcance y objetivos . . . 1
1.4. Requerimientos . . . 2

2. Introducción específica 3
2.1. Tecnologías de hardware utilizadas . . . 3
2.1.1. Espressif ESP32 . . . 3
2.1.2. DHT11 . . . 3
2.1.3. BMP280 . . . 4
2.1.4. Fotoresistor . . . 4
2.1.5. Joystick analógico . . . 4
2.1.6. Display . . . 5
2.1.7. Motores . . . 5
2.2. Tecnologias de software utilizadas . . . 6
2.2.1. ESP-IDF . . . 6
2.2.2. Docker . . . 6
2.2.3. Visual Studio Code . . . 7
2.2.4. Ubuntu . . . 7

3. Diseño e implementación 9
3.1. Arquitectura de software del sistema . . . 9
3.2. Arquitectura de hardware del sistema . . . 9
3.3. Interfaz de usuario . . . 9

4. Ensayos y resultados 11
4.1. Proceso de desarrollo y aseguramiento de calidad . . . 11
4.2. Prototipos de los diferentes modulos . . . 11
4.2.1. Prototipo de funcionalidad de medición de temperatura y humedad . . . 11
4.2.2. Prototipo de funcionalidad de medicion de presion . . . 12
4.2.3. Prototipo de funcionalidad de medición de valor de luminosidad . . . 13
4.2.4. Prototipo de funcionalidad de obtencion de valores analogicos del joystick . . . 13
4.2.5. Prototipo de funcionalidad de presentación de display . . . 14
4.2.6. Prototipo de funcionalidad de control de motores DC . . . 14
4.3. Tests de los diferentes modulos . . . 14
4.4. Tests del producto final . . . 14
4.5. Reportes de testing . . . 14

v

Índice general

Resumen I

1. Introducción general 1
1.1. Motivación . . . 1
1.2. Estado del arte . . . 1
1.3. Alcance y objetivos . . . 1
1.4. Requerimientos . . . 2

2. Introducción específica 3
2.1. Tecnologías de hardware utilizadas . . . 3
2.1.1. Espressif ESP32 . . . 3
2.1.2. Sensor de temperatura y humedad DHT11 . . . 3
2.1.3. Sensor de presión BMP280 . . . 4
2.1.4. Fotoresistor como sensor de luminosidad . . . 4
2.1.5. Joystick analógico . . . 4
2.1.6. Display LCM1602A . . . 5
2.1.7. Motores de corriente continua . . . 5
2.2. Tecnologías de software utilizadas . . . 6
2.2.1. Marco de trabajo ESP-IDF . . . 6
2.2.2. Plataforma Docker . . . 6
2.2.3. Visual Studio Code . . . 7
2.2.4. Sistema operativo Ubuntu . . . 7

3. Diseño e implementación 9
3.1. Arquitectura de software del sistema . . . 9
3.2. Arquitectura de hardware del sistema . . . 9
3.3. Interfaz de usuario . . . 9

4. Ensayos y resultados 11
4.1. Proceso de desarrollo y aseguramiento de calidad . . . 11
4.2. Prototipos de los diferentes modulos . . . 11
4.2.1. Prototipo de funcionalidad de medición de temperatura y humedad . . . 11
4.2.2. Prototipo de funcionalidad de medición de presión . . . 12
4.2.3. Prototipo de funcionalidad de medición de valor de luminosidad . . . 12
4.2.4. Prototipo de funcionalidad de obtencion de valores analogicos del joystick . . . 13
4.2.5. Prototipo de funcionalidad de presentación de display . . . 14
4.2.6. Prototipo de funcionalidad de control de motores DC . . . 14
4.3. Tests de los diferentes módulos . . . 14
4.4. Tests del producto final . . . 14
4.5. Reportes de testing . . . 14

Índice de tablas

4.1. Conexionado DHT11. 12
4.2. Conexionado BMP280. 12
4.3. Conexionado fotorresistor. 13
4.4. Conexionado joystick. 14

Índice de tablas

4.1. Conexionado DHT11 11
4.2. Conexionado BMP280 12
4.3. Conexionado fotorresistor 13
4.4. Conexionado joystick 13

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 es una serie de microcontroladores de bajo costo y bajo consumo creado y desarrollado por *Espressif Systems* embebido en un chip con Wi-Fi integrado (2.4 GHz band) y Bluetooth de modo dual. Emplea dos cores Xtensa® 32-bit LX6 CPU. Incluye interruptores de antena integrados, balun de RF, amplificador de potencia, amplificador de recepción de bajo ruido, filtros, 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 utilizada para el desarrollo del presente trabajo.

FIGURA 2.1. Microcontrolador ESP32-WROOM-32D.

2.1.2. DHT11

El DHT11 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). Utilizado en aplicaciones académicas relacionadas al control automático de temperatura, aire acondicionado, monitoreo ambiental en agricultura y más. En la figura 2.2 se puede apreciar una imagen del mismo.

3

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 es una serie de microcontroladores de bajo costo y bajo consumo creado y desarrollado por *Espressif Systems* embebido en un chip con Wi-Fi integrado (2.4 GHz band) y Bluetooth de modo dual. Emplea dos cores Xtensa® 32-bit LX6 CPU. Incluye interruptores de antena integrados, balun de RF, amplificador de potencia, amplificador de recepción de bajo ruido, filtros, 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 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 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). Utilizado en aplicaciones académicas relacionadas al control automático de temperatura, aire acondicionado, monitoreo ambiental en agricultura y más. En la figura 2.2 se puede apreciar una imagen del mismo.

FIGURA 2.2. Sensor DHT11.

2.1.3. BMP280

El BMP280 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 de EMC a un muy bajo consumo. En la figura 2.3 se puede apreciar una imagen del mismo.

FIGURA 2.3. Sensor BMP280.

2.1.4. Fotoresistor

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 electrones con lo cual el material se hace más conductor, y se disminuye su resistencia. En la figura 2.4 se puede apreciar una imagen del mismo.

FIGURA 2.4. Fotoresistor.

2.1.5. Joystick analógico

El módulo de joystick analógico 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 V cuando inicialmente se encuentra en posición de reposo (en el centro), mientras que en el trayecto del desplazamiento de la palanca hará que la salida varíe de 0V a 5V dependiendo de su dirección X e Y. Al conectar este módulo a un

4 Capítulo 2. Introducción específica

FIGURA 2.2. Sensor DHT11.

2.1.3. Sensor de presión BMP280

El BMP280 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 mismo.

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 electrones con lo cual el material se hace más conductor, y se disminuye su resistencia. En la figura 2.4 se puede apreciar una imagen del mismo.

FIGURA 2.4. Fotoresistor.

2.1.5. Joystick analógico

El módulo de joystick analógico 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 V cuando inicialmente se encuentra en posición de reposo (en el centro), mientras que en el trayecto del desplazamiento de la palanca hará que la salida varíe de 0V a 5V dependiendo de su dirección X e Y. Al conectar este módulo a un

microcontrolador se puede leer un valor de alrededor de 512 en su posición de reposo mientras que al moverlo cambia entre 0 y 1023 dependiendo de su posición. En la figura 2.5 se puede apreciar una imagen del mismo.

FIGURA 2.5. Joystick analógico.

2.1.6. Display

El display LCM1602A consta de una pantalla de cristal líquido de 1602 caracteres, en un módulo de matriz de puntos para mostrar letras, números y caracteres, etc. Permite representar dos filas con hasta 16 caracteres en cada una y dado que se encuentra integrado a una interfaz adaptadroa I2C puede ser controlado por este protocolo. En la figura 2.6 se puede apreciar una imagen del mismo.

FIGURA 2.6. Display LCM1602A.

2.1.7. Motores

El motor DC (o corriente continua), pertenece a la clase de los electromotores y sirve principalmente para transformar la 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 0.15Nm y 0.60Nm. En la figura 2.7 se puede apreciar una imagen del mismo.

microcontrolador se puede leer un valor de alrededor de 512 en su posición de reposo mientras que al moverlo cambia entre 0 y 1023 dependiendo de su posición. En la figura 2.5 se puede apreciar una imagen del mismo.

FIGURA 2.5. Joystick analógico.

2.1.6. Display LCM1602A

El display LCM1602A consta de una pantalla de cristal líquido de 1602 caracteres, en un módulo de matriz de puntos para mostrar letras, números y caracteres, etc. Permite representar dos filas con hasta 16 caracteres en cada una y dado que se encuentra integrado a una interfaz adaptadroa I2C puede ser controlado por este protocolo. En la figura 2.6 se puede apreciar una imagen del mismo.

FIGURA 2.6. Display LCM1602A.

2.1.7. Motores de corriente continua

El motor DC (o corriente continua), pertenece a la clase de los electromotores y sirve principalmente para transformar la 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 mismo.

FIGURA 2.7. Motor de corriente continua.

2.2. Tecnologias de software utilizadas

2.2.1. ESP-IDF

Espressif 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 Internet de las cosas (IoT) con Wi-Fi, Bluetooth, administración de energía y varias otras características del sistema. Sus componentes son:

1. Toolchain para compilar el codigo para ESP32,
2. Build tools - con utilidades como CMake y Ninja para construir la aplicación completa para ESP32,
3. ESP-IDF que esencialmente contiene la API de desarrollo (software base y librerias complementarias) para ESP32 y scripts para ejecutar Toolchain.

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

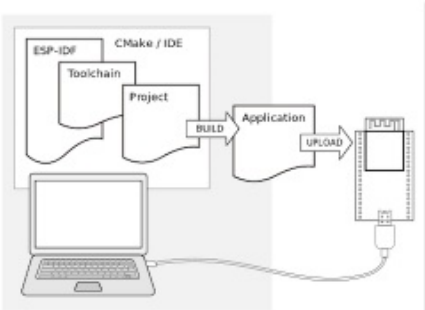

FIGURA 2.8. Proceso de desarrollo utilizando ESP-IDF.

2.2.2. Docker

Docker 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.

6 Capítulo 2. Introducción específica

FIGURA 2.7. Motor de corriente continua.

2.2. Tecnologías de software utilizadas

2.2.1. Marco de trabajo ESP-IDF

Espressif 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 Internet de las cosas (IoT) con Wi-Fi, Bluetooth, administración de energía y varias otras características del sistema. Sus componentes son:

1. Toolchain para compilar el codigo para ESP32,
2. Build tools - con utilidades como CMake y Ninja para construir la aplicación completa para ESP32,
3. ESP-IDF que esencialmente contiene la API de desarrollo (software base y bibliotecas complementarias) para ESP32 y scripts para ejecutar Toolchain.

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

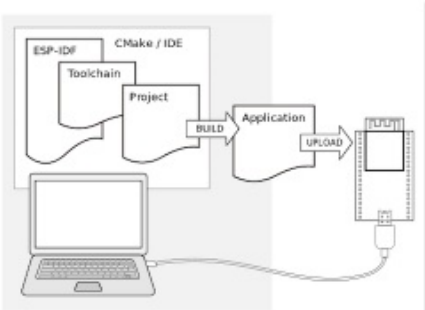

FIGURA 2.8. Proceso de desarrollo utilizando ESP-IDF.

2.2.2. Plataforma Docker

Docker 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 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. Ubuntu

Ubuntu 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.

2.2. Tecnologías de software utilizadas 7

2.2.3. Visual Studio Code

Visual Studio Code 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 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.

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

4.2.1. Prototipo de funcionalidad de medición de temperatura y humedad

Para el desarrollo de este prototipo se utilizó el framework ESP-IDF y la librería de código ESP-IDF Components que provee el soporte para gestionar el DHT11. A continuación se puede apreciar el conexionado del prototipo en la figura 4.1 y los detalles del conexionado de sus pines en la tabla 4.1.

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

4.2.1. Prototipo de funcionalidad de medición de temperatura y humedad

Para el desarrollo de este prototipo se utilizó el framework ESP-IDF y la biblioteca de código ESP-IDF Components que provee el soporte para gestionar el DHT11. A continuación se puede apreciar el conexionado del prototipo en la figura 4.1 y los detalles del conexionado de sus pines en la tabla 4.1.

TABLA 4.1. Conexionado DHT11

Nombre lógico	Pin GPIO
Pin DHT11	2

Logic Name	Pin GPIO
Pin DHT11	2

TABLA 4.1. Conexionado DHT11.

Logic Name	Pin GPIO
BMP SDA	18
BMP280 SCL	19

TABLA 4.2. Conexionado BMP280.

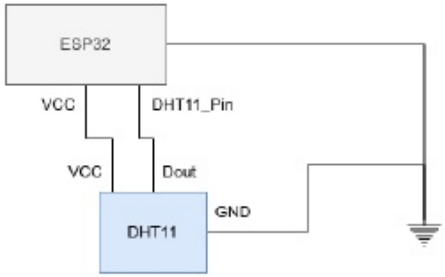

FIGURA 4.1. Circuito del conexionado DHT11.

4.2.2. Prototipo de funcionalidad de medicion de presion

Para el desarrollo de este prototipo se utilizó el framework ESP-IDF y la librería de código ESP-IDF Components que provee el soporte para gestionar el BMP280. A continuación se puede apreciar el conexionado del prototipo en la figura 4.2 y los detalles de sus pines en la tabla 4.2.

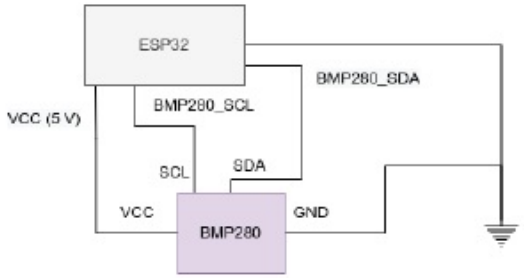

FIGURA 4.2. Conexionado BMP280.

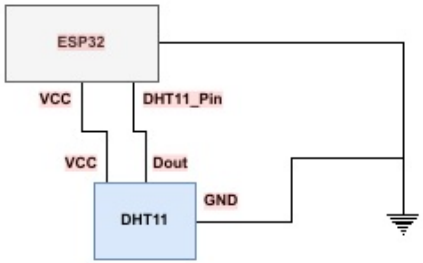

FIGURA 4.1. Circuito del conexionado DHT11.

4.2.2. Prototipo de funcionalidad de medición de presión

Para el desarrollo de este prototipo se utilizó el framework ESP-IDF y la biblioteca de código ESP-IDF Components que provee el soporte para gestionar el BMP280. A continuación se puede apreciar el conexionado del prototipo en la figura 4.2 y los detalles de sus pines en la tabla 4.2.

TABLA 4.2. Conexionado BMP280

Nombre lógico	Pin GPIO
BMP SDA	18
BMP280 SCL	19

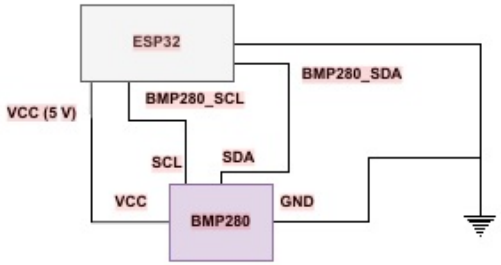

FIGURA 4.2. Conexionado BMP280.

4.2.3. Prototipo de funcionalidad de medición de valor de luminosidad

Para el desarrollo de este prototipo se utilizó el siguiente conexionado y la biblioteca de código ADC provista por ESP-IDF. A continuación se puede apreciar el conexionado del prototipo en la figura 4.3 y los detalles de sus pines en la tabla 4.3.

Logic Name	Pin ADC	Pin GPIO
ADC Fot pin	channel 0 - unit 2	4

TABLA 4.3. Conexionado fotorresistor.

4.2.3. Prototipo de funcionalidad de medición de valor de luminosidad

Para el desarrollo de este prototipo se utilizó el siguiente conexionado y la librería de código ADC provista por ESP-IDF. A continuación se puede apreciar el conexionado del prototipo en la figura 4.3 y los detalles de sus pines en la tabla 4.3.

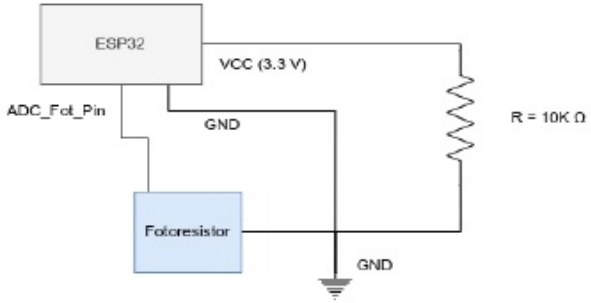

FIGURA 4.3. Conexionado fotorresistor.

4.2.4. Prototipo de funcionalidad de obtencion de valores analogicos del joystick

Para el desarrollo de este prototipo se utilizó el siguiente conexionado y la librería de código ADC provista por ESP-IDF. A continuación se puede apreciar el conexionado del prototipo en la figura 4.4 y los detalles de sus pines en la tabla 4.4.

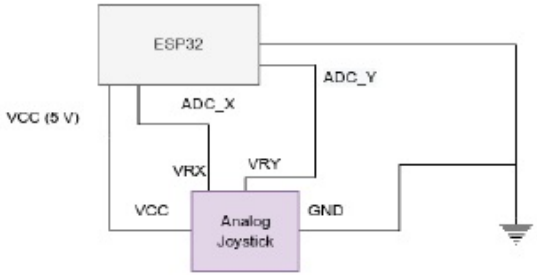

FIGURA 4.4. Conexionado joystick.

TABLA 4.3. Conexionado fotorresistor

Nombre lógico	Pin ADC	Pin GPIO
ADC Fot pin	channel 0 - unit 2	4

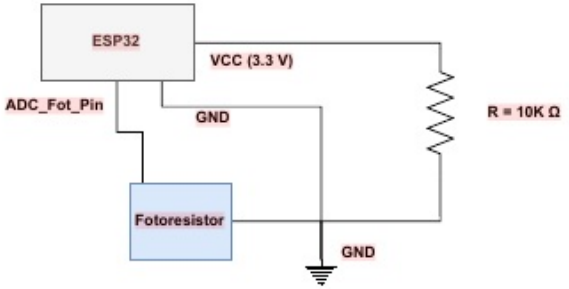

FIGURA 4.3. Conexionado fotorresistor.

4.2.4. Prototipo de funcionalidad de obtencion de valores analogicos del joystick

Para el desarrollo de este prototipo se utilizó el siguiente conexionado y la biblioteca de código ADC provista por ESP-IDF. A continuación se puede apreciar el conexionado del prototipo en la figura 4.4 y los detalles de sus pines en la tabla 4.4.

TABLA 4.4. Conexionado joystick

Nombre lógico	Pin ADC	Pin GPIO
ADC X	channel 1 - unit 2	0
ADC Y	channel 7 - unit 1	35

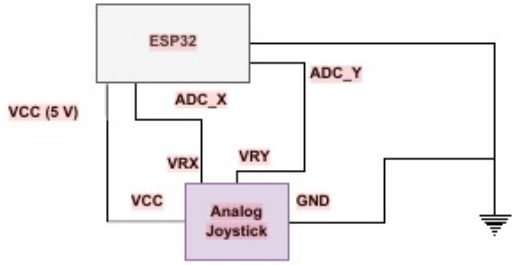

FIGURA 4.4. Conexionado joystick.

Logic Name	Pin ADC	Pin GPIO
ADC X	channel 1 - unit 2	0
ADC Y	channel 7 - unit 1	35

TABLA 4.4. Conexionado joystick.

4.2.5. Prototipo de funcionalidad de presentación de display

4.2.6. Prototipo de funcionalidad de control de motores DC

...

4.3. Tests de los diferentes modulos

...

4.4. Tests del producto final

...

4.5. Reportes de testing

...

4.6. Verificacion y validacion del producto

...

4.7. Documentacion del producto

...

4.2.5. Prototipo de funcionalidad de presentación de display

4.2.6. 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

...