Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computadores

CE1105 – Principios de modelado en Ingeniería

Documentación de código

Trabajo elaborado por:

2022016016 – Dylan Guerrero González

2022321746 – Javier Hernández Castillo

2023138210 – José Fabio Ruiz Morales

2023058736 – José Luis Vargas Vargas

Profesor:

Diego Mora Rojas

I Semestre 2025

Contenido

1. ESP-32

Esta sección detallará el código que se encarga de manejar los sensores y actuadores del sistema.

1.1 Bibliotecas

Para lograr un buen funcionamiento del dispositivo se incluyeron las siguientes bibliotecas:

1. `Wifi.h` : permite la conexión y gestión de la red Wi-Fi de la ESP-32.
2. `WebServer.h` : otorga las funcionalidades básicas para crear un servidor web.
3. `DHT.h`: sirve para controlar el sensor DHT responsable de medir humedad y temperatura ambiental.

1.2 Constantes y objetos importantes

- `DHTPIN`: pin conectado al sensor DHT11 para medir la humedad y temperatura ambiental.
- `ssid` y `password`: credenciales de la red Wi-Fi a la que se conecta la ESP-32.
- `pinBomba`, `pinVentilador` y `pinLuzUV`: pines para controlar el encendido y apagado del ventilador, bomba de agua y luz LED
- `IN1, IN2, IN3, IN4`: pines para controlar la dirección de giro de los motores.
- `pinSensorDrenaje`, `pinSensorRiego`, `pinSensorLuz`, `pinSensorHumedad`: pines de lectura para los demás sensores conectados al dispositivo.

1.3 Variables

Se definen las siguientes variables de tipo booleano para almacenar el estado de algunos dispositivos:

- `bombaEncendida`
- `ventiladorEncendido`
- `luzEncendida`

1.4 Lectura de sensores

Las funciones que se detallan a continuación se encargan de leer los valores de cada sensor conectado al equipo

- `leerNivel(pin)`: lee los valores del sistema de riego y drenaje y los muestra en una escala de 1 a10.

- `leerIntensidadLuz()`: lee el valor de intensidad lumínica del ambiente y lo muestra en una escala de 1 a 10.
- `leerHumedadTierra()`: lee el valore de la humedad del suelo y lo muestra en una escala de 1 a 10.
- `dht.readTemperature()`: método propio de la biblioteca `DHT.h` que devuelve el valor en grados Celsius de la temperatura del ambiente
- `dht.readHumidity()`: método propio de la biblioteca `DHT.h` que devuelve el valor de la humedad relativa del ambiente

1.5 Página web

La función `generarPagina()` crea una pagina web que permite al usuario controlar los diversos sensores y actuadores además de visualizar los distintos datos. Esta pagina se actualiza cada segundo para mostrar en tiempo real los datos.

1.6 Inicialización

`setup()` inicializa la comunicación serial, los pines, la conexión Wi-Fi y energiza los sensores. Una vez realizados los pasos anteriores configura el servidor y las rutas para poder manejar las solicitudes HTTP. Finalmente inicia el servidor para que el equipo pueda manejar las peticiones.

1.7 Control por medio de HTTP

Se manejan diferentes rutas para controlar los dispositivos y obtener los datos recolectados por los sensores

- `/bomba/encender` y `/bomba/apagar` : encienden y apagan la bomba de agua respectivamente.
- `/ventilador/encender` y `/ventilador/apagar` : apaga y enciende el dispositivo
- `/uv/encender` y `/uv/apagar` : controlan el encendido y apagado de las luces led.
- `/motores/adelante`, `motores/atrás` y `/motores/stop`: manejan la dirección de giro de los motores o los detienen.

1.8 Lectura de datos

Al acceder a la ruta `/niveles` el servidor envía los valores de los sensores que se han descrito anteriormente. En caso de encontrar un error se muestra un mensaje de advertencia.

1.9 Manejo de solicitudes

`loop()` se encarga de manejar de forma persistente las solicitudes del cliente permitiendo que el servidor responda a las interacciones con la página web.

2. ESP-32 CAM

Esta sección se encarga de explicar el código que se encarga del funcionamiento de la cámara integrada en sistema. Esta cámara se basa en una ESP-32S con un módulo de cámara OV2604 de 2 megapíxeles.

2.1 Bibliotecas

Se incluyeron las siguientes bibliotecas para asegurar un correcto funcionamiento del dispositivo

- `esp_camera.h`: otorga las funciones esenciales para la configuración y operación de la unidad.
- `WiFi.h`: permite conectar la ESP32S a una red Wi-Fi y gestionar la comunicación

2.2 Definiciones importantes

`CAMERA_MODEL_AI_THINKER` especifica el modelo de cámara a emplear y `camera.pins.h` es un archivo que contiene la configura de pines para el modelo a utilizar.

2.3 Configuración de red

`ssid` y `password` ayudan a establecer la conexión con la red Wi-Fi a utilizar. local_IP, gateway y subnet le asignan una IP fija necesaria para la comunicación.

2.4 Servidor web

Es necesario crear un servidor web para manejar las solicitudes de los clientes. Para eso `WiFiServer.server()` crea uno en el puerto 80

2.4.1 Arranque

`startCameraServer()` inicia el servidor al llamar a `server.begin()` y `Serial.println()` muestra un mensaje de confirmación junto con la IP para acceder al servidor

2.5 Captura de imagen

`handleCapture()` llama a la función `esp_camera_fb_get()` la cual es la que verdaderamente se encarga de capturar la imagen y enviarla al cliente por medio de HTTP.

2.6 Configuración

`setup()` inicializa la conexión de red, configura los pines y arranca el servidor

- `WiFi.config()` configura la red tal como se comentó anteriormente
- `WiFi.begin()` intenta conectar la ESP a la red proporcionada
- `esp_camera_init()` inicializa la cámara con la configuración establecida.

2.7 Manejo de solicitudes

`loop()` maneja las solicitudes del cliente mediante el protocolo HTTP comprobando con `server.available()` si algún cliente esta conectado y llamando a `handleCapture()` en caso de que la comprobación sea exitosa, en caso contrario se devuelve error 404.