

Bomba de riego automática con sensor de humedad

Joana Artetxe
Patricia Rodríguez
Giovanna Lani
Sara Bolaños

Introducción

La funcionalidad principal del sistema es automatizar el riego de las plantas ajustando parámetros configurables como el tiempo de riego y el nivel mínimo de humedad en el suelo. Además, tiene un mecanismo de seguridad que asegura un riego periodico en caso de que las condiciones de humedad no se cumplan, para que la planta no sufra de estrés hídrico. El microcontrolador controla el riego y la monitoriza la humedad mediante un sensor permitiendo ajustes remotos a través de un programa en Java que interactúa mediante comunicación serial.



Justificación

El sistema optimiza el riego, asegurando un uso eficiente del agua y adaptándose a distintas necesidades mediante parámetros configurables. Su función de seguridad protege las plantas, garantizando su cuidado incluso en condiciones adversas.

Periféricos

GPIO

Controla la activación de la bomba de riego

TIMERS

Gestionan la periodicidad del riego y el monitoreo del sensor

UART

Permite la comunicación entre el microcontrolador y el programa de Java

ADC

Lee los valores del sensor de humedad.

Recursos



STM32

PC

Bomba de
riego

Sensor de
humedad

IRQ (interrupciones)

ADC1_2_IRQHandler

Procesa los datos del sensor de humedad y activa flag para enviar información.

EXTI4_IRQHandler

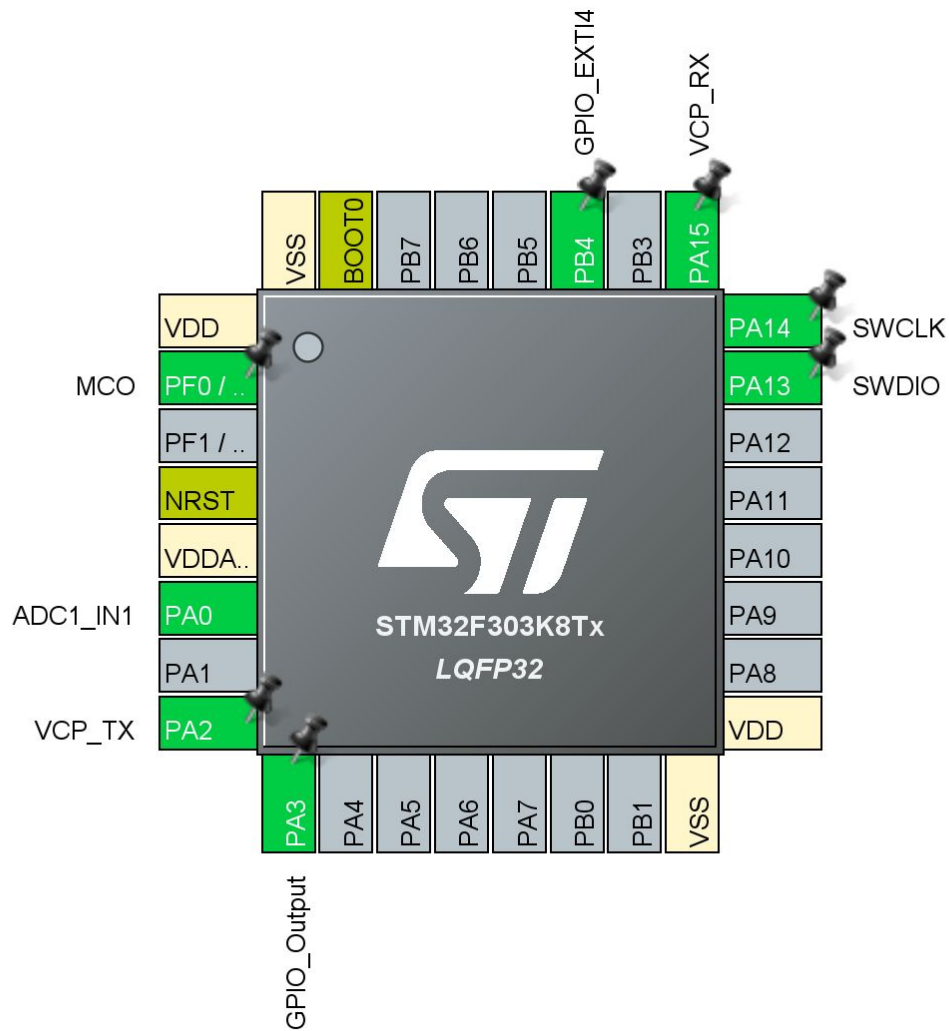
Permite activar o desactivar la bomba mediante un pulsador.

TIM3_IRQHandler

Sincroniza la lógica del riego automático y asegura que se revisen las condiciones de riego periódicamente

USART2_IRQHandler

Procesa comandos recibidos desde la aplicación



Líneas de código (MAIN)

```
while (1) {
    if (estadoAnterior != estado) {
        if (estado) {
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 1);
        } else {
            HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 0);
        }
        estadoAnterior = estado;
    }
    if (enviarInfo) {
        char cadena[10];
        sprintf(cadena, "%d\n\r", valor);
        HAL_UART_Transmit_IT(&huart2, (uint8_t*) cadena, strlen(cadena));
        enviarInfo = 0;
    }
    if (empezarRegar && bombaActivadaManualmente == 0) {
        if (valor > humedad_minima && !estado) {
            estado = 1;
        }
        if (estado && valor <= humedad_minima) {
            estado = 0;
            empezarRegar = 0;
            TIM3->CNT = 0;
        }
    }
}
```


Funciones adicionales I

```
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart) {
    if (huart->Instance == USART2) {
        if (rx_data != '\n') {
            rx_buffer[rx_index++] = rx_data;
            if (rx_index >= sizeof(rx_buffer) - 1) {
                rx_index = sizeof(rx_buffer) - 1;
            }
        } else {
            rx_buffer[rx_index] = '\0';
            rx_index = 0;
            if (rx_buffer[0] == 'T') {
                cambiarTemporizador(atoi((char*) &rx_buffer[1]));
            } else if (rx_buffer[0] == 'H') {
                humedad_minima = atoi((char*) &rx_buffer[1]);
            }
        }
        HAL_UART_Receive_IT(&huart2, &rx_data, sizeof(uint8_t));
    }
}
```

```
void cambiarTemporizador(uint32_t tiempo_riego)
{
    HAL_TIM_Base_Stop(&htim3);
    uint32_t periodo = tiempo_riego * 1000 - 1;
    TIM3->ARR = periodo;
    TIM3->CNT = 0;
    HAL_TIM_Base_Start(&htim3);
}
```

STM32f3xx_it

```
void EXTI4_IRQHandler(void) {
```

```
    if (estado) {  
        if (bombaActivadaManualmente) {  
            estado = 0;  
            bombaActivadaManualmente = 0;  
        }  
    } else {  
        estado = 1;  
        bombaActivadaManualmente = 1;  
    }  
    HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_4);  
}
```

```
void ADC1_2_IRQHandler(void) {
```

```
    valor = HAL_ADC_GetValue(&hadc1);  
    enviarInfo = 1;
```


```
    HAL_ADC_IRQHandler(&hadc1);  
}
```

```
void TIM3_IRQHandler(void)
```

```
{  
    empezarRegar = 1;  
}
```

Aplicación de Java

```
private static void configurarPuertoSerial() {  
    puertoSerial = SerialPort.getCommPort(PUERTO);  
    puertoSerial.setBaudRate(BAUD_RATE);  
    if (puertoSerial.openPort()) {  
        System.out.println("Puerto serial abierto correctamente.");  
        Thread hiloLectura = new Thread(() -> leerHumedad(puertoSerial));  
        hiloLectura.setDaemon(true);  
        hiloLectura.start();  
        Timer timer = new Timer(500, e -> actualizarDatos());  
        timer.start();  
    } else {  
        System.out.println("Error al abrir el puerto serial.");  
    }  
}
```



The screenshot shows a Java application window titled "Control de Riego". The window has a light gray background and contains the following elements:

- Tiempo de Riego (segundos):** A label followed by the value "3".
- Valor Tiempo Riego:** A text input field.
- Establecer Tiempo:** A blue button.
- Humedad mínima (%):** A label followed by the value "2000".
- Valor Humedad Mínima:** A text input field.
- Establecer Humedad:** A blue button.
- Última Humedad:** A label followed by the value "2067".

DAFO

Debilidades

- Sin respaldo ante fallos del sensor o cortes de energía.
- Posible dificultad de instalación para usuarios no técnicos.
- Comunicación limitada a conexión serial.

Amenazas

- Competencia con sistemas comerciales avanzados.
- Dependencia de componentes específicos.
- Problemas climáticos extremos pueden afectar efectividad.



Fortalezas

- Automatización basada en mediciones a tiempo real.
- Ahorro de agua y cuidado sostenible
- Económico y accesible

Oportunidades

- Integración con IoT para monitoreo remoto.
- Expansión a usos agrícolas.
- Incorporación de más sensores.
- Añadir conectividad Wi-Fi o Bluetooth.