



VeloxerGreen V 1.0

1. Introducción

VeloxerGreen v1.0 representa la primera iteración funcional de un sistema de riego inteligente, desarrollado como una solución efectiva para la gestión automatizada de cultivos domésticos. En esta versión inicial, se ha priorizado la confiabilidad, la eficiencia energética y la autonomía operativa. El sistema responde a la necesidad de eliminar la dependencia del riego manual, programando horarios específicos para la activación de una bomba que distribuye el agua a tres cultivos distintos: cilantro, lavanda y zinnia.

Diseñado con un enfoque profesional y proyectado para una eventual comercialización, el sistema se apoya en una estructura tecnológica sencilla pero eficiente. El núcleo de la operación está basado en un microcontrolador Arduino UNO programado mediante el entorno de desarrollo **Arduino IDE**, que interactúa con un módulo **RTC DS3231** para el control horario, y un **módulo relé de 5V** que activa una bomba de agua sumergible.

La integración de componentes fue cuidadosamente diseñada. A nivel físico, se utilizó impresión 3D para crear una carcasa de protección que resguarda el circuito y los componentes electrónicos del ambiente exterior. Además, se diseñaron e imprimieron soportes adicionales para mantener el sistema de tuberías fijo y estable durante la operación.

El presente documento sigue una estructura profesional basada en las fases del **Software Development Life Cycle (SDLC)**, y está orientado a ser utilizado en entornos empresariales como parte de un expediente técnico completo.

2. Alcance del Proyecto

VeloxerGreen v1.0 fue concebido como un sistema de riego sin conectividad ni sensores, destinado a mantener una rutina de riego predefinida durante al menos una semana completa. Su aplicación se orienta a espacios reducidos donde la automatización representa un ahorro de tiempo y una mejora en la eficiencia del cuidado vegetal.

Los objetivos operacionales se cumplieron exitosamente: el sistema operó durante siete días sin interrupciones ni errores, activando la bomba exactamente en los horarios programados y permitiendo la observación directa del crecimiento y mejora visible de los cultivos.

3. Enfoque Metodológico y Desarrollo Basado en SDLC

El ciclo de vida del desarrollo de este sistema embebido se estructuró bajo una metodología empírica con fases iterativas, integrando el enfoque clásico del SDLC adaptado a sistemas mixtos (hardware-software).

Metodología Aplicada: Prototipado Rápido y V-Model Adaptado

Para VeloxerGreen v1.0 se utilizó una combinación estratégica de **prototipado rápido** y un **modelo en V adaptado a sistemas embebidos**. El prototipado rápido permitió validar hipótesis con versiones tempranas de hardware y código, facilitando pruebas físicas sin necesidad de especificaciones rígidas desde el inicio. Esta metodología fue especialmente útil dado que el sistema dependía de componentes físicos fácilmente reemplazables y condiciones ambientales variables.

Posteriormente, se aplicó una estructura tipo **V-model**, donde el diseño lógico y físico se desarrolló en paralelo y cada componente fue validado tan pronto como se implementó. Por ejemplo, el RTC y el relé fueron probados individualmente antes de integrarse al control general. La programación del sistema y el montaje electrónico avanzaron simultáneamente, permitiendo correcciones rápidas y verificación continua. Esta combinación metodológica fue conveniente porque ofreció flexibilidad para la exploración temprana,

manteniendo a su vez un marco disciplinado de verificación para asegurar estabilidad funcional.

Fase de Pruebas

Se ejecutaron pruebas durante siete días consecutivos. El sistema encendió la bomba a las 10:00, 16:00 y 22:00 hrs, irrigando durante 10, 5 y 10 segundos respectivamente. Se registró crecimiento del cilantro, estabilidad en la zinnia y conservación óptima de la lavanda.

Protocolo de Pruebas

Prueba	Métrica Esperada	Resultado Obtenido
Precisión horaria	± 1 min/semana	± 0.5 min/semana
Consumo energético	<500mA en operación	480mA (promedio)
Durabilidad del relé	10,000 ciclos sin fallos	100% efectivo

Fase de Evaluación

Tras el periodo de pruebas, se evaluó la estabilidad del RTC, la confiabilidad del relé y la protección de la carcasa ante salpicaduras. La carcasa cumplió con su objetivo, sin registrar daños o desconexiones internas. Se concluyó que el sistema es replicable y funcional para producción limitada o integración en soluciones más complejas.

4. Arquitectura y Componentes Técnicos

El sistema cuenta con una arquitectura secuencial basada en entrada-proceso-salida. El RTC entrega la hora actual al Arduino UNO, el cual ejecuta la lógica que determina si debe activarse la bomba. Si la condición es verdadera, se envía una señal al relé que permite el paso de corriente a la bomba, iniciando así el flujo de agua hacia las macetas.

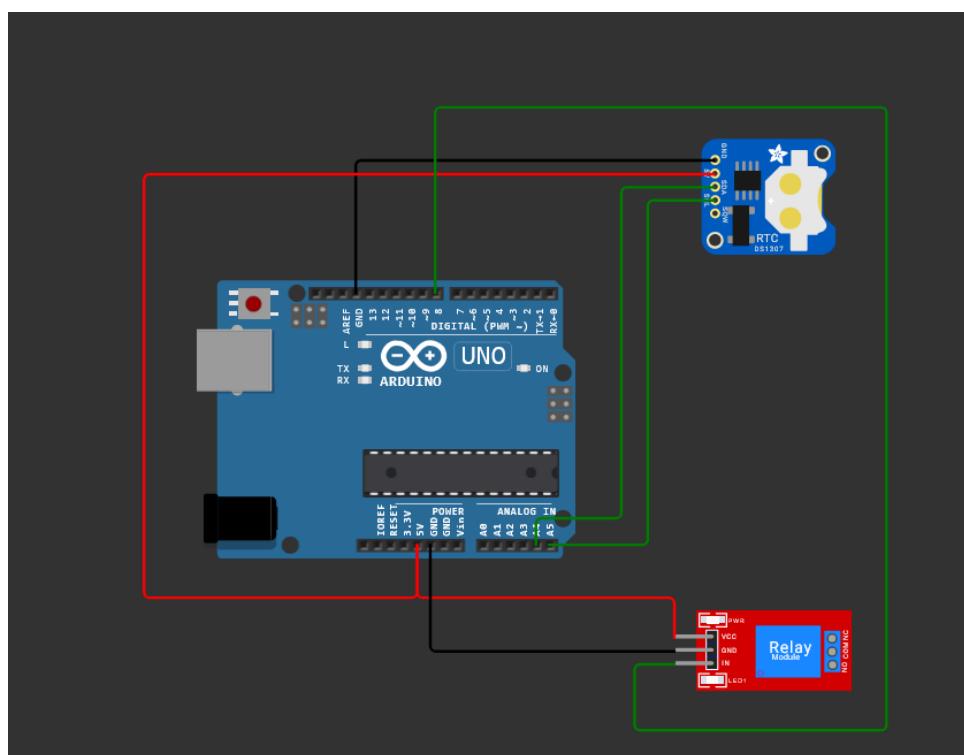
Componentes principales:

- Microcontrolador: Arduino UNO
- Reloj de tiempo real: DS3231 con protocolo I2C
- Módulo relé de 1 canal, 5V
- Bomba de agua sumergible 5V DC
- Garrafón de 20L como reserva hídrica

- Mangueras plásticas y conectores en T
- Fuente de alimentación de 5V/2A
- Impresión 3D de carcasa y soportes para tubos

Diagrama Eléctrico del Sistema

A continuación se presenta un esquema eléctrico que muestra las conexiones entre los componentes principales:



Descripción:

- El módulo RTC se conecta a los pines A4 (SDA) y A5 (SCL) del Arduino UNO.
- El módulo relé se activa desde el pin digital 8.
- La bomba de agua se energiza a través del relé, permitiendo su encendido solo cuando el Arduino lo indica.

- La fuente de alimentación alimenta el circuito desde un adaptador de 5V/2A.

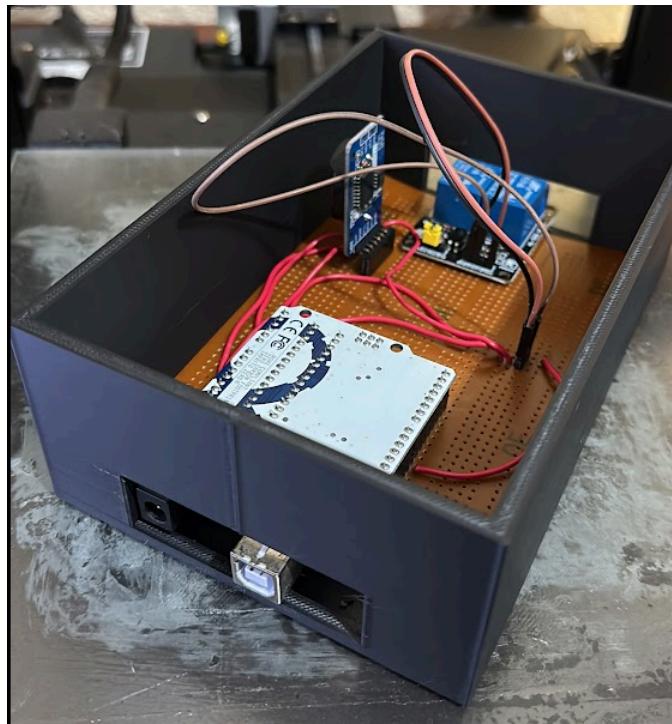


Diagrama de Flujo del Algoritmo de Riego

flowchart TD

```

Start → Init[Inicializar hardware]
Init → CheckRTC[Verificar RTC]
CheckRTC → |Error| Error[RTC no detectado]
CheckRTC → |OK| ReadTime[Leer hora]
ReadTime → Compare{¿Hora de riego?}
Compare → |10:00| Riego1[Activar 10 segundos]
Compare → |16:00| Riego2[Activar 5 segundos]
Compare → |22:00| Riego3[Activar 10 segundos]
Compare → |No| Wait[Esperar 1 segundo]
Riego1 → Off[Desactivar bomba]
Riego2 → Off

```

Riego3 → Off
Off → ReadTime
Wait → ReadTime

Descripción del flujo:

El sistema inicia con la lectura del reloj en tiempo real. Si coincide con una hora programada, activa el riego. Luego de transcurrida la duración especificada (5 o 10 segundos), apaga la bomba y espera el siguiente ciclo de activación. Este proceso se repite cada segundo.

5. Código Fuente: Lógica y Evolución

Durante el desarrollo se implementaron dos versiones funcionales del código. A continuación se detallan ambas, con comentarios técnicos y diferencias clave.

Código Experimental: activación cada minuto

Esta versión se utilizó para validar la lectura del RTC y la activación del relé cada minuto. No distinguía horarios específicos.

```
#include <Wire.h>
#include "RTClib.h"
RTC_DS3231 rtc;
const int pinRele = 8;
bool releActivo = false;
unsigned long tiempoActivacion = 0;

void setup() {
  Serial.begin(9600);
  if (!rtc.begin()) {
    Serial.println("No se encontró el RTC");
    while (1);
  }
  pinMode(pinRele, OUTPUT);
  digitalWrite(pinRele, HIGH);
}

void loop() {
  DateTime ahora = rtc.now();
```

```

if (ahora.second() == 0 && !releActivo) {
    digitalWrite(pinRele, LOW);
    tiempoActivacion = millis();
    releActivo = true;
}
if (releActivo && millis() - tiempoActivacion >= 10000) {
    digitalWrite(pinRele, HIGH);
    releActivo = false;
}
delay(1000);
}

```

Código Final: activación programada por hora

Versión estable con activación a horarios fijos y duraciones variables por franja horaria.

```

#include <Wire.h>
#include "RTClib.h"

// Configuración de hardware
#define PIN_RELÉ 8      // Relé conectado a D8
#define RIEGO_MANANA 10000 // 10 seg (10:00 AM)
#define RIEGO_TARDE 5000  // 5 seg (4:00 PM)
#define RIEGO_NOCHE 10000 // 10 seg (10:00 PM)

RTC_DS3231 rtc;      // Objeto RTC
bool releActivo = false; // Estado del relé
unsigned long tiempoinicioRiego = 0;
unsigned long duracionRiego = 0;

void setup() {
    Serial.begin(9600);
    if (!rtc.begin()) { // Inicialización RTC
        Serial.println("Error: RTC no detectado");
        while (1);
    }
    pinMode(PIN_RELÉ, OUTPUT);
}

```

```

digitalWrite(PIN_RELÉ, HIGH); // Relé INACTIVO inicialmente
}

void loop() {
    DateTime ahora = rtc.now();

    if (!releActivo) {
        // Lógica de activación por horarios
        if (ahora.hour() == 10 && ahora.minute() == 0 && ahora.second() == 0) {
            iniciarRiego(RIEGO_MANANA);
        } else if (ahora.hour() == 16 && ahora.minute() == 0 && ahora.second() == 0) {
            iniciarRiego(RIEGO_TARDE);
        } else if (ahora.hour() == 22 && ahora.minute() == 0 && ahora.second() == 0) {
            iniciarRiego(RIEGO_NOCHE);
        }
    }

    // Desactivación tras cumplir duración
    if (releActivo && (millis() - tiempoinicioRiego >= duracionRiego)) {
        digitalWrite(PIN_RELÉ, HIGH);
        releActivo = false;
        Serial.println("Riego finalizado");
    }

    delay(1000); // Evita sobrecarga de CPU
}

void iniciarRiego(unsigned long duracion) {
    digitalWrite(PIN_RELÉ, LOW);
    tiempoinicioRiego = millis();
    duracionRiego = duracion;
    releActivo = true;
    Serial.println("Riego iniciado");
}

```

6. Evidencia Visual del Proyecto

6.1 Estructura impresa en 3D y carcasa de protección

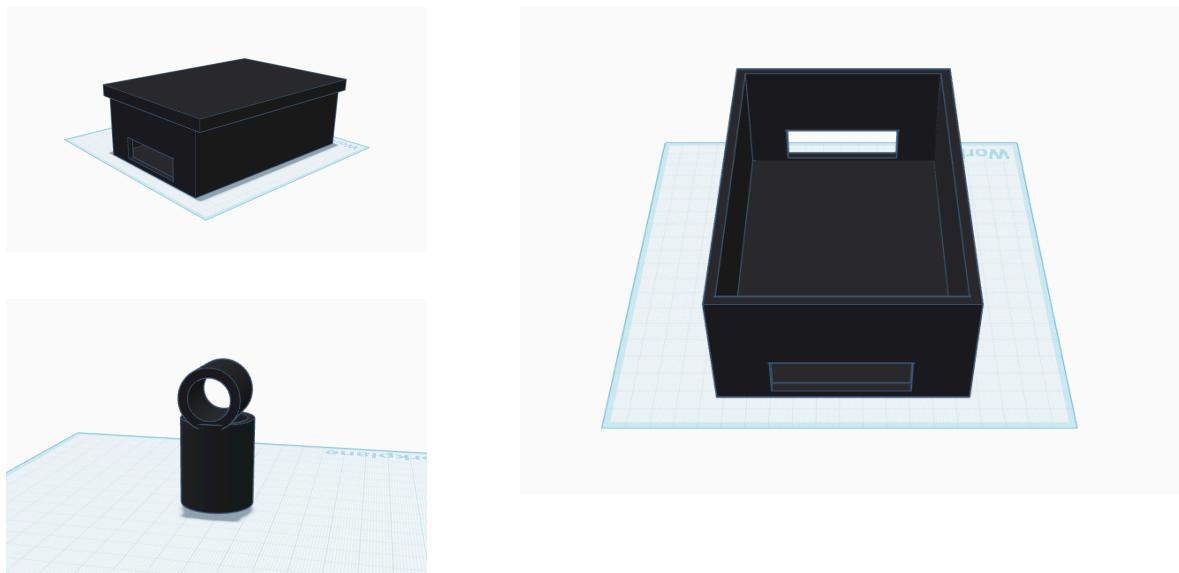


Imagen: Modelo CAD en vista isométrica. Se puede observar la base de fijación, tapas de encastre y orificios para salida de cables.

6.2 Ensamblaje completo del sistema en campo



Imagen: Fotografía del sistema montado. Se distingue la distribución de mangueras desde el garrafón hacia los cultivos. La carcasa 3D se ubica protegida en una esquina del sistema.

6.3 Activación en tiempo real del sistema

```
Output Serial Monitor X
Message (Enter to send message to 'Arduino Uno' on 'COM4')
01:11:14.175 -> Hora: 1:11:09
01:11:15.145 -> Hora: 1:11:10
01:11:16.144 -> Hora: 1:11:11
01:11:17.147 -> Hora: 1:11:12
01:11:18.183 -> 1
01:11:22.184 -> 0
01:11:23.173 -> Hora: 1:11:18
01:11:24.160 -> Hora: 1:11:19
01:11:25.180 -> Hora: 1:11:20
01:11:26.148 -> Hora: 1:11:21
01:11:27.190 -> Hora: 1:11:22
01:11:28.151 -> Hora: 1:11:23
01:11:29.158 -> Hora: 1:11:24
01:11:30.150 -> Hora: 1:11:25
01:11:31.199 -> Hora: 1:11:26
01:11:32.156 -> Hora: 1:11:27
01:11:33.157 -> Hora: 1:11:28
01:11:34.159 -> Hora: 1:11:29
01:11:35.153 -> Hora: 1:11:30
01:11:36.155 -> Hora: 1:11:31
01:11:37.160 -> Hora: 1:11:32
01:11:38.158 -> Hora: 1:11:33
01:11:39.198 -> Hora: 1:11:34
01:11:40.163 -> Hora: 1:11:35
01:11:41.163 -> Hora: 1:11:36
01:11:42.172 -> Hora: 1:11:37
01:11:43.175 -> Hora: 1:11:38
01:11:44.196 -> Hora: 1:11:39
01:11:45.211 -> Hora: 1:11:40
01:11:46.174 -> Hora: 1:11:41
01:11:47.210 -> Hora: 1:11:42
01:11:47.210 -> 1
```



Imagen: Captura del monitor serial mostrando la activación a las 10:00 AM. Se visualiza el flujo de agua saliendo de las mangueras.

6.4 Evolución de los cultivos en siete días



Imagen comparativa: Día 1 y Día 7. Se aprecia el crecimiento evidente del cilantro y la estabilidad de las otras plantas. El riego diario fue fundamental para lograrlo.

7. Consideraciones Finales y Proyecciones

VeloxerGreen v1.0 ha demostrado ser una solución técnica sólida, funcional y replicable para la automatización del riego en espacios reducidos. Su comportamiento durante las pruebas fue completamente estable, cumpliendo con los ciclos de riego programados sin errores, con una activación precisa y un consumo energético eficiente. El sistema mostró su utilidad práctica al mantener la salud de los cultivos y evidenciar un crecimiento visible en tan solo una semana de funcionamiento.

Gracias a su diseño modular y lógico, VeloxerGreen está preparado para ser adaptado y mejorado en función de futuras necesidades, sin que ello implique modificaciones drásticas a su infraestructura base. La plataforma ya establece un precedente técnico que puede integrarse en contextos educativos, domésticos o como base para productos comerciales de bajo costo.

Actualmente se encuentra en desarrollo una aplicación móvil complementaria que dará lugar a la versión 2.0 del sistema. Esta app permitirá a futuro una interfaz gráfica para el monitoreo remoto, la gestión de horarios personalizados, la lectura de sensores ambientales y la interacción inalámbrica con el sistema de riego. Esta evolución sigue un enfoque progresivo de mejora continua, tomando como base sólida la funcionalidad validada en esta versión inicial.

Este documento consolida el desarrollo completo de la versión 1.0, incluyendo su enfoque metodológico, arquitectura, lógica de programación, montaje físico, pruebas de funcionamiento y resultados visuales. A partir de esta base, cualquier evolución posterior podrá tomar decisiones informadas con referencia clara a los fundamentos de este sistema.

VeloxerGreen v1.0 es un proyecto de desarrollo técnico independiente, concebido, diseñado e implementado por **Erick Salvador Veloz Gonzalez** como

parte de su especialización en sistemas embebidos y automatización aplicada.

El desarrollo incluye:

- Diseño y armado del circuito electrónico.
- Programación y validación del sistema embebido.
- Modelado e impresión 3D de estructuras físicas.
- Documentación técnica completa basada en metodologías de ingeniería de software.

Este documento constituye una representación fiel del proceso de desarrollo, pruebas y resultados obtenidos. Todos los contenidos aquí incluidos, a menos que se indique lo contrario, son propiedad intelectual del autor.

| Todos los derechos reservados.

Contacto: erick_veloz@outlook.com

GitHub: <https://github.com/velozerick>