HO CHI MINH CITY, VIETNAMESE-GERMAN UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

AGRICULTURAL IRRIGATION SYSTEM

RTOS ASSIGNMENT PROJECT

Group: Đệ Nhất Củ Lạc Giòn Tan

Student:	Vuong Thieu Luan
ID:	10421035
Student:	Do Duc Toan
ID:	10421060
Student:	Tran Duy Khang
ID:	10421025
Student:	Nguyen Ngoc Hoang Nam
ID:	10421042
Student:	Mai Nguyen Viet Phu
ID:	10421047
Student:	Pham Tran Truong An
ID:	10421001
Student:	Tu Trung Tin
ID:	10421106

HỒ CHÍ MINH CITY

Content

1 Introduction **2**

1.1 Project Overview 2

1.2 Development Tools 2

2 System Analytic and Design **3**

2.1 System Architecture 3

2.2 Finite State Machine 5

3 System development and implementation **6**

3.1 Proteus Simulation 6

3.2 System initialization 6

3.3 Module 1 - Task 1 7

3.4 Module 2 - Task 2 7

3.5 Module 3 - Task 3 8

3.6 Module 4 - Task 4 8

3.7 Module 5 - Task 5 8

3.8 Module 6 - IDLE 9

4 Source Code Deployment **9**

5 Conclusion **9**

1 Introduction

This report details the implementation of an automated agricultural irrigation system designed to optimize fertilizer distribution and water usage. The system sequentially activates and deactivates three fertilizer mixing valves and two pumps, with LED indicators representing each device's status. The process ensures efficient irrigation cycles, repeating every 60 seconds to maintain optimal soil moisture and nutrient levels.

1.1 Project Overview

An agricultural irrigation system needs to perform the following 5 tasks, for each irrigation session:

- Open fertilizer mixing Valve 1 for 5 seconds
- Close Valve 1, open fertilizer mixing Valve 2 for 5 seconds
- Close Valve 2, open fertilizer mixing Valve 3 for 5 seconds
- Close valve 3, open pump 1 for 10 seconds
- Turn off pump 1, turn on pump 2 for 30 seconds.
- Finally, turn off pump 2.

At the end of this process, pump 2 will turn off, and after a 60-second interval, the cycle will repeat. Each device (valve or pump) is indicated by an LED light that illuminates when the valve is opened or the pump is turned on, and turns off otherwise.

1.2 Development Tools

- **STM32Cube:** STM32Cube is a comprehensive development platform for STM32 microcontrollers, including STM32CubeMX for configuration and code generation. It provides embedded software libraries and middleware components. In this project, STM32Cube facilitated efficient microcontroller setup and integration of various components, streamlining the development process.
- **Proteus:** Proteus is a powerful simulation and design tool for electronic circuits, supporting a wide range of microcontrollers, including STM32. It offers features for schematic capture, PCB layout design, and real-time simulation. Proteus was used to simulate the irrigation system, allowing thorough testing and validation of the design before physical implementation.

2 System Analytic and Design

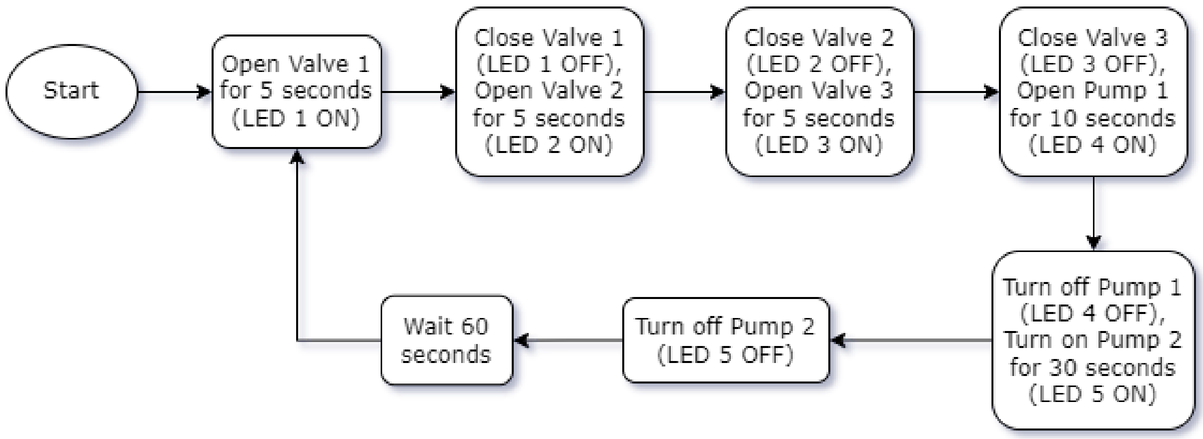
2.1 System Architecture

The system architecture of the automated agricultural irrigation system is designed to efficiently manage the sequence of operations for valves and pumps. It consists of the following key components:

- **Microcontroller:** The STM32 microcontroller acts as the central control unit, managing the timing and sequence of valve and pump operations.
- **Valves:** Three fertilizer mixing valves are used, each controlled to open and close for specific intervals to ensure precise fertilizer mixing.
- **Pumps:** Two pumps are employed, where Pump 1 operates for a shorter duration to initiate irrigation and Pump 2 runs for a longer duration to complete the irrigation cycle.
- **LED Indicators:** LED lights are connected to each valve and pump to visually indicate their operational status, turning on when a valve is opened or a pump is running, and off otherwise.
- **Power Supply:** A stable power supply is essential to ensure uninterrupted operation of the microcontroller, valves, pumps, and LEDs.

The integration of these components ensures a reliable and efficient irrigation system, capable of maintaining optimal soil moisture and nutrient levels through automated cycles

The following flowchart illustrates the sequence of operations for the irrigation system:


Start
Open Valve 1 for 5 seconds (LED 1 ON)
Close Valve 1 (LED 1 OFF), Open Valve 2 for 5 seconds (LED 2 ON)
Close Valve 2 (LED 2 OFF), Open Valve 3 for 5 seconds (LED 3 ON)
Close Valve 3 (LED 3 OFF), Open Pump 1 for 10 seconds (LED 4 ON)
Turn off Pump 1 (LED 4 OFF), Turn on Pump 2 for 30 seconds (LED 5 ON)
Turn off Pump 2 (LED 5 OFF)
Wait 60 seconds

Hình 1: Flowchart of the Irrigation System

Flowchart Explanation: The flowchart is designed to manage the sequence of operations for valves and pumps efficiently. Below is a detailed step-by-step explanation of the flowchart:

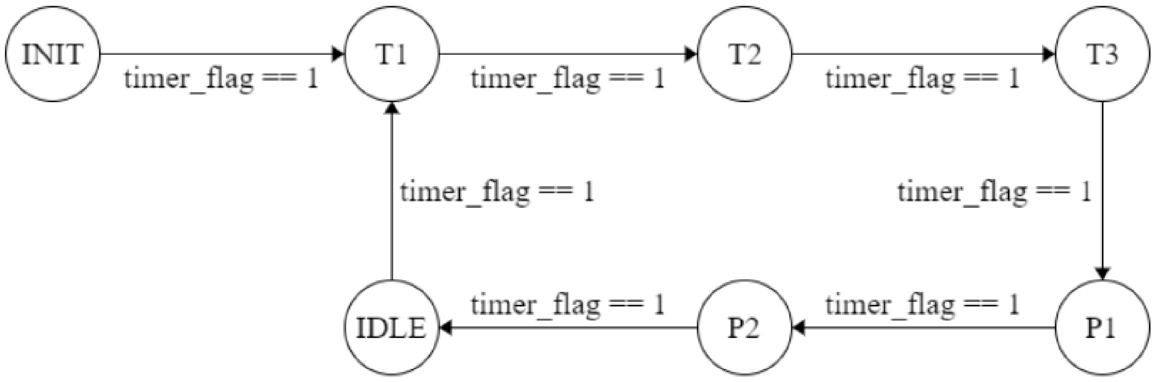
1. **Start:** The process begins with the initial state where all valves and pumps are off.
2. **Open Valve 1 for 5 seconds:**
    - Valve 1 is opened.
    - LED 1 turns on to indicate Valve 1 is active.
    - Wait for 5 seconds.
3. **Open Valve 2 for 5 seconds:**
    - Valve 1 is closed.
    - LED 1 turns off.
    - Valve 2 is opened.
    - LED 2 turns on to indicate Valve 2 is active.
    - Wait for 5 seconds.
4. **Open Valve 3 for 5 seconds:**
    - Valve 2 is closed.
    - LED 2 turns off.
    - Valve 3 is opened.
    - LED 3 turns on to indicate Valve 3 is active.
    - Wait for 5 seconds.
5. **Open Pump 1 for 10 seconds:**
    - Valve 3 is closed.
    - LED 3 turns off.
    - Pump 1 is turned on.
    - LED 4 turns on to indicate Pump 1 is active.
    - Wait for 10 seconds.
6. **Turn on Pump 2 for 30 seconds:**
    - Pump 1 is turned off.
    - LED 4 turns off.
    - Pump 2 is turned on.
    - LED 5 turns on to indicate Pump 2 is active.
    - Wait for 30 seconds.
7. **Turn off Pump 2**
    - Pump 2 is turned off.

  - LED 5 turns off.

8. **Wait for 60 seconds:**
   - The system pauses for 60 seconds before repeating the cycle.

9. **Repeat:** The entire process repeats from Step 2, ensuring continuous and efficient irrigation.

2.2 Finite State Machine



Hình 2: State Machine Diagram

- T1: fertilizer mixing Valve 1
- T2: fertilizer mixing Valve 2
- T3: fertilizer mixing Valve 3
- P1: Pump 1
- P2: Pump 2

State Machine Diagram Explanation:

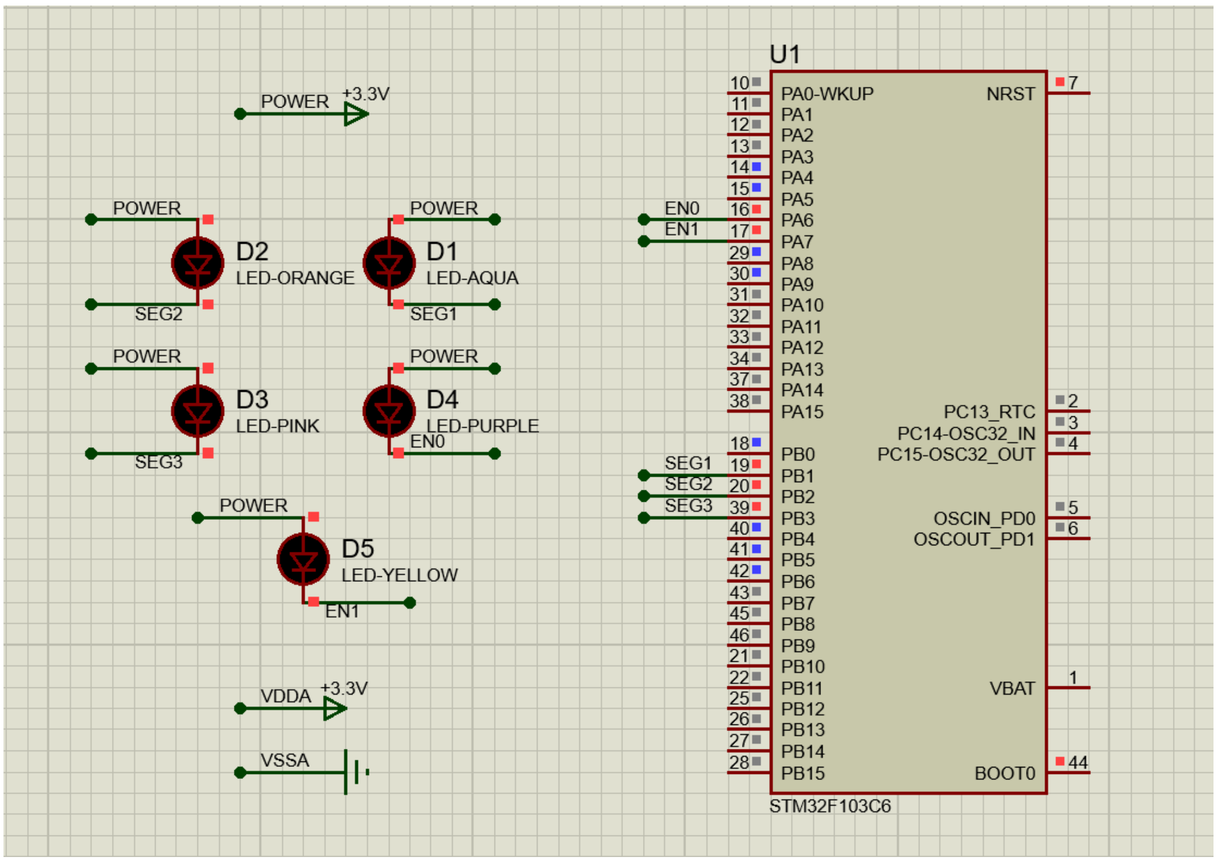
- INIT State: Turn off all T1, T2, T3, T4, T5.
- T1 State: Turn on T1 for 5 seconds.
- T2 State: Turn off T1, turn on T2 for 5 seconds.
- T3 State: Turn off T2, turn on T3 for 5 seconds.
- P1 State: Turn off T3, turn on P1 for 10 seconds.
- P2 State: Turn off P1, turn on P2 for 30 seconds.
- IDLE State: Turn off P2, stay for 60 seconds and then change back to State T1.

The use of a cooperative scheduler in this project provides several significant advantages:

- **Predictable Task Execution:** Each task runs to completion before the next task starts, making the system behavior predictable and easier to debug.
- **Simplicity:** Cooperative scheduling is simpler to implement compared to preemptive multitasking, reducing the complexity and potential for bugs.
- **Resource Efficiency:** It avoids the overhead associated with context switching, which is beneficial for systems with limited resources like microcontrollers.
- **Deterministic Timing:** Tasks execute in a known order, which is crucial for time-sensitive applications like irrigation where precise timing of valve and pump operations is required.

3 System development and implementation

3.1 Proteus Simulation



3.2 System initialization

In the INIT State, all the Valves and Pumps are set to non-active state. The ‘setTimer(0, 5000)‘ function sets a delay of 5 seconds for the Valve 1.

```
void fsm_irrigation_init(){
    HAL_GPIO_WritePin(SEG1_GPIO_Port, SEG1_Pin, SET);
    HAL_GPIO_WritePin(SEG2_GPIO_Port, SEG2_Pin, SET);
    HAL_GPIO_WritePin(SEG3_GPIO_Port, SEG3_Pin, SET);
    HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, SET);
    HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, SET);
    status = T1;
    setTimer(0, 5000);
}
```

3.3 Module 1 - Task 1

In Task 1, Valve 1 is activated by setting SEG1 to the active state. After 5 seconds (checked by ‘isTimerExpired(0)‘), the system transitions to Task 2 and deactivates Valve 1. The ‘setTimer(0, 5000)‘ function sets a delay of 5 seconds for the Valve 2.

```
case T1:
    HAL_GPIO_WritePin(SEG1_GPIO_Port, SEG1_Pin, RESET);
    if(isTimerExpired(0) == 1){
       status = T2;
        setTimer(0, 5000);
        HAL_GPIO_WritePin(SEG1_GPIO_Port, SEG1_Pin, SET);
    }
    break;
```

3.4 Module 2 - Task 2

In Task 2, Valve 2 is activated by setting SEG2 to the active state. After 5 seconds, the system transitions to Task 3 and deactivates Valve 2. The ‘setTimer(0, 5000)‘ function sets a delay of 5 seconds for Valve 3:

```
case T2:
    HAL_GPIO_WritePin(SEG2_GPIO_Port, SEG2_Pin, RESET);
    if(isTimerExpired(0) == 1){
        status = T3;
        setTimer(0, 5000);
        HAL_GPIO_WritePin(SEG2_GPIO_Port, SEG2_Pin, SET);
    }
    break;
```

3.5 Module 3 - Task 3

In Task 3, Valve 3 is activated by setting SEG3 to the active state. After 5 seconds, the system transitions to Task 4 and deactivates Valve 3. The 'setTimer(0, 10000)' function sets a delay of 10 seconds for Pump 1:

```
case T3:
    HAL_GPIO_WritePin(SEG3_GPIO_Port, SEG3_Pin, RESET);
    if(isTimerExpired(0) == 1){
        status = P1;
        setTimer(0, 10000);
    HAL_GPIO_WritePin(SEG3_GPIO_Port, SEG3_Pin, SET);
    }
    break;
```

3.6 Module 4 - Task 4

In Task 4, Pump 1 is turned on by setting EN0 to the active state. After 10 seconds, the system transitions to Task 5 and turns off Pump 1. The 'setTimer(0, 30000)' function sets a delay of 30 seconds for Pump 2:

```
case P1:
    HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, RESET);
    if(isTimerExpired(0) == 1){
        status = P2;
        setTimer(0, 30000);
        HAL_GPIO_WritePin(EN0_GPIO_Port, EN0_Pin, SET);
    }
    break;
```

3.7 Module 5 - Task 5

In Task 5, Pump 2 is turned on by setting EN1 to the active state. After 30 seconds, the system transitions to the IDLE state and turns off Pump 2. The 'setTimer(0, 60000)' function sets a delay of 60 seconds for IDLE State:

```
case P2:
    HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, RESET);
    if(isTimerExpired(0) == 1){
        status = IDLE;
        setTimer(0, 60000);
        HAL_GPIO_WritePin(EN1_GPIO_Port, EN1_Pin, SET);
    }
    break;
```

3.8 Module 6 - IDLE

In the IDLE state, the system waits for 60 seconds before restarting the sequence from Task 1. The ‘setTimer(0, 5000)‘ function sets a delay of 5 seconds for Valve 1.

```
case IDLE:
    if(isTimerExpired(0) == 1){
       status = T1;
       setTimer(0, 5000);
    }
    break;
```

4 Source Code Deployment

The completed source code can be found at **GitHub repository**.

5 Conclusion

This project successfully developed an automated irrigation system that performs a sequence of tasks, including the timed operation of valves and pumps. By visualizing each device with LED indicators, the system ensures clear and intuitive monitoring of its operational status. The cooperative scheduler implemented in the system allows for seamless and efficient task management, ensuring that the irrigation process runs smoothly. The combination of advanced development tools and a well-structured approach resulted in a robust and effective irrigation solution, demonstrating the potential for further advancements in agricultural automation.