Smart River Monitoring.

Leonardo Tassinari

January 29, 2024

Contents

1 Circuit

1.1 Level detector

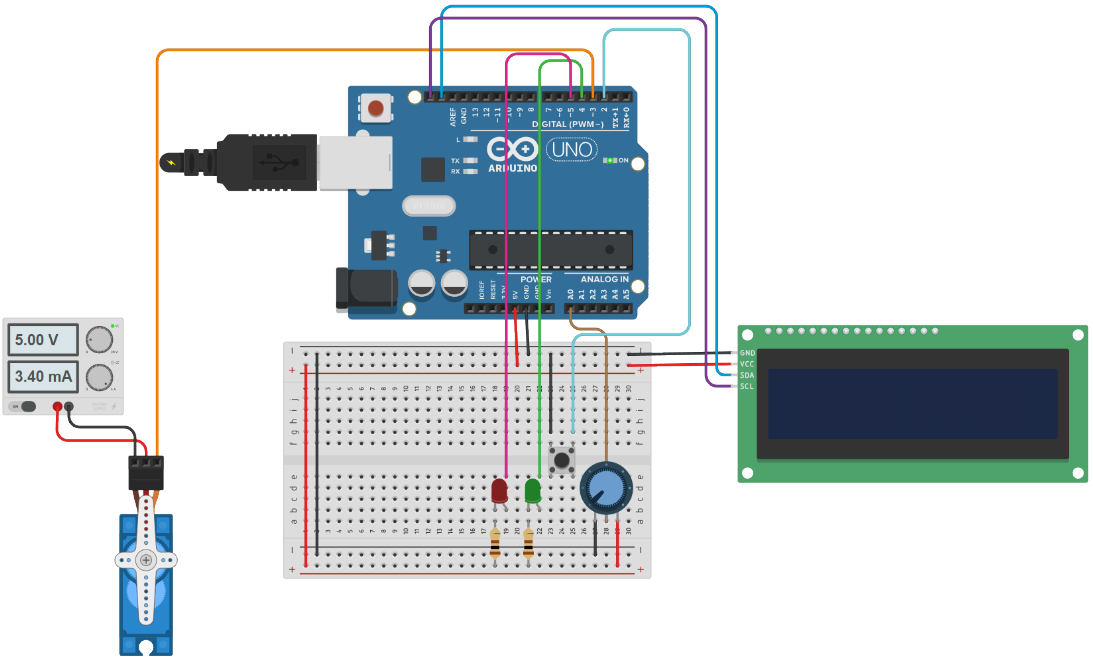
Here the main board is an esp8266 not the arduino uno.
The circuit schematic is as follows:

Device	pinout
Sonar vcc	Vin
Echo	D1
Trig	D0
Green Anode	D2
Red Anode	D3
Red cathode	220ohm to ground
Green cathode	220ohm to ground

Since the sonar works at 5 volt its vcc pin has to be fed by the vin of the esp8266, it is directly liked to the 5v line of the usb, otherwise it would get only 3.3 that is the working tension of the microcontroller.

1.2 Water channel controller

The circuit schematic is as follows:



Device	pinout
Servo vcc/gnd	ext power supply
Servo signal	3
Button	2
Green Anode	4
Red Anode	5
Red cathode	220ohm to ground
Green cathode	220ohm to ground
Potentiometer	A0
LCD SDA	SDA
LCD SCL	SCL

The button is configured as input pullup, so it does not require an external resistor. The servo is fed by an external power supply since it needs peaks of current that the arduino is not capable of offering without some additional capacitors.

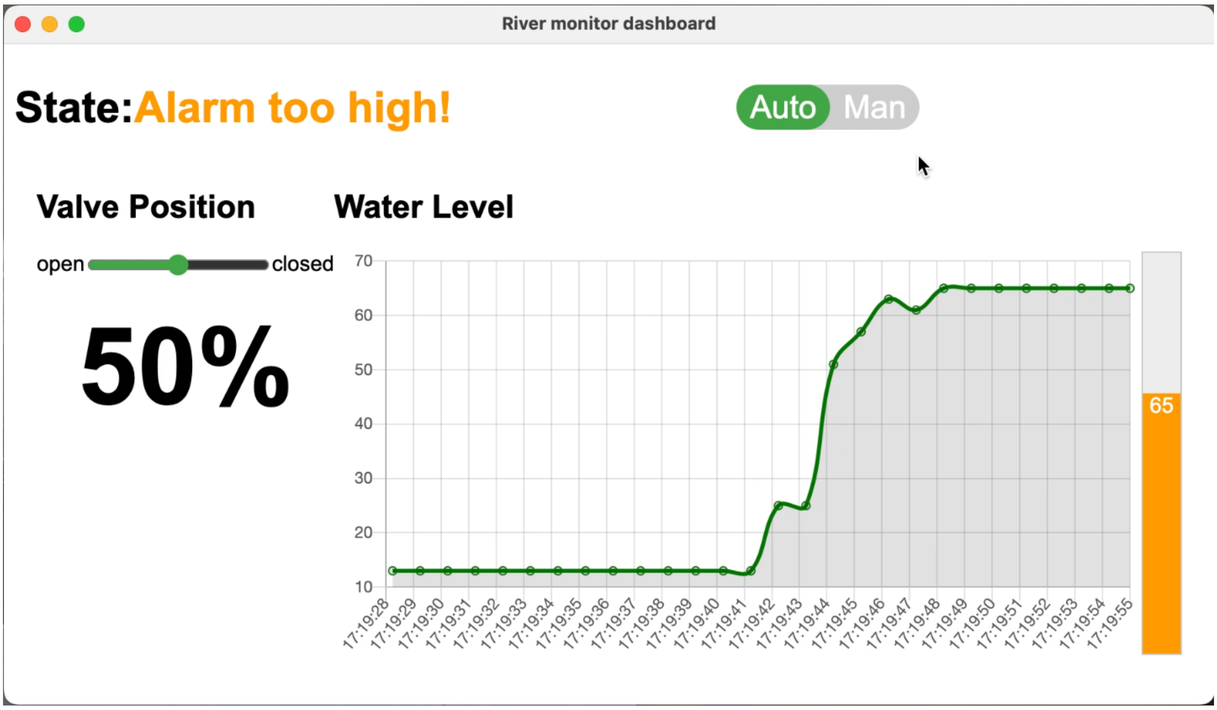
2 Software Architecture

2.1 Water Level Monitoring subsystem

This segment of the system is composed by a simple arduino sketch that sends the readings of the sonar to a mqtt topic.
It also reads the required frequency of the recordings in another topic, this frequency is set by the main service (see below).

2.2 Dashboard

The dashboard is written in html css an js, it is packaged in a runnable executable exploiting the electron framework.
It fetches the data via http with a the get method, the main service sends the parameter in a json format.
It sets the parameter with a post request sending data in json format as well.
From the dashboard it is possible to select the mode and the position of the valve.
It also shows the actual level of the water and a chart of the past values.



River monitor dashboard

State:Alarm too high!

Valve Position 50%

2.3 River Monitoring Service

This is the main service that manages the logic of the system, it follows the specs given by the assignment.
It is written in js and run by node, it communicates with arduino exchanging data on the serial line. It fetches the level of the water via mqtt.
Via express.js it exposes the API with get and post methods to set and get the data.

2.4 Water Channel Controller

The program running on arduino is written in C++ using the wiring framework and the platformIo utility.
It is managed by a syncronous scheduler, it is triggered by a timer and launches non cooperative tasks.
Each task period has been chosen accordingly to its operative requirements, and is a multiple of the base period of the scheduler. Some empiric tests have been made to ensure that no overrun happens in any scenario.
The work tree of the program is divided in:

- Sensors, containing interfaces (abstract classes) and implementations for the devices.
- actuators, containing interfaces (abstract classes) and implementations for the devices.
- System, containing the scheduler and the task interface.
- Task, containing all the tasks.

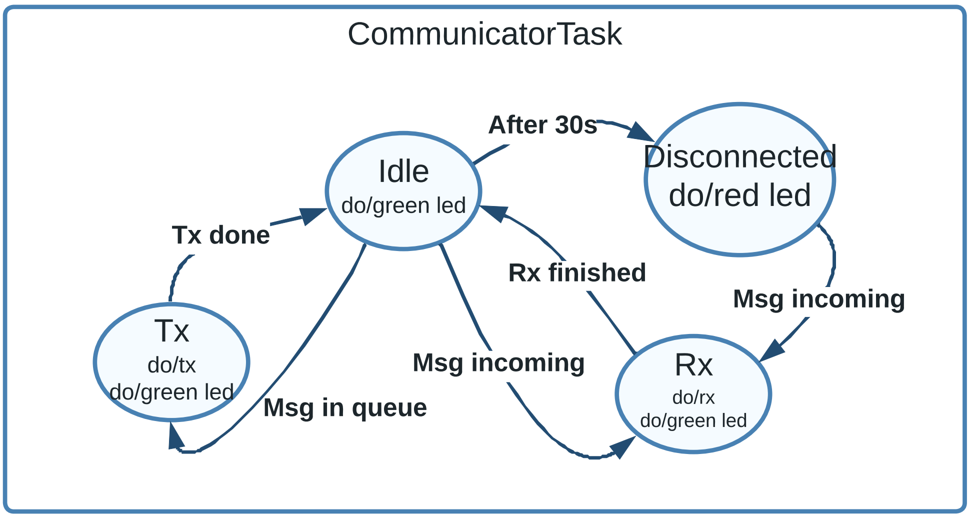
On the root are present:

- The main file, entry point of the program, where the scheduler and the tasks are instantiated.
- Smart river monitoring class, representing the domain of the system with all the shared variables.
- Config, configuration file to tune all the parameters of the program.

3 Tasks

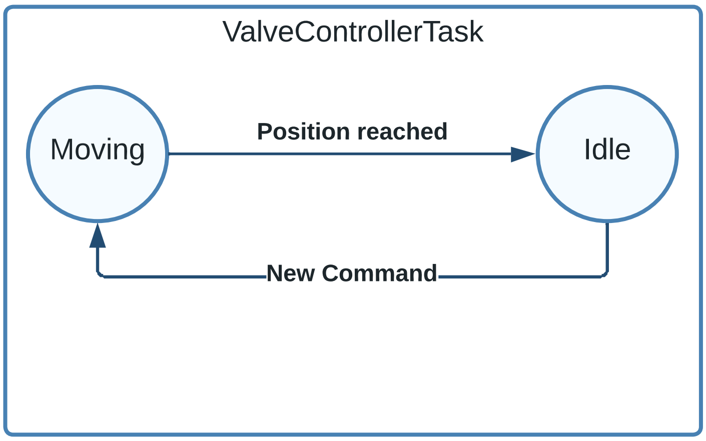
3.1 communicator Task

This task is responsible of communicating via serial with the dashboard encoding in json the relative data.
It also collects the request incoming via serial in the same format.



3.2 valveController Task

This task is responsible of change the position of the valve according to the instructions fetched from the other tasks.



3.3 inputChecker Task

This task is responsible of sensing the changes in the potentiometer and on the button, it issues commands accordingly. It automatically switches to manual if the potentiometer is moved, while for the button it alternates between the two modes.

3.4 displayFeedback Task

This task displays the active status of the machine (auto/manual) and the position of the valve message on the LCD display.