Report on CM10194 Computer Systems Architecture 1 - Coursework 2

Plant care project

Introduction

My project is about monitoring plants and providing an automatic care.

Motivation: I have flowers I need to look after at home and sometimes I worry that I can not notice if they are in uncomfortable condition (the light can be too bright for a plant that enjoys a half-shadow or vice versa). Also, I thought it might be a good idea to ease the process of watering by providing a pump watering system, which would start just by clicking a button that is next to me. Since the amount of water is always the same, the user would not worry if they overwater or underwater the plant, because now the hose will always transport the same amount of liquid.

Since the piezo can alarm and draw my attention, just as the LED RGB, I can immediately check the LCD for information and change whatever is causing the problem.

It also was an interesting project to start with, because I wanted to use my LCD screen in a real-life application. Here I was able to lookup data and possible problems just by looking and listing through LCD.

Technologies used:

<u>1) Arduino-sender</u>

I use such sensors as:

- temperature sensor: reads room temperature. The rounded part of the sensor is facing away from the Arduino, left pin goes to power and right pin leads to ground. Centre pin is connected to analog input pin 2 (A2). Calibrated using the tutorial on https://bc-robotics.com/tutorials/using-a-tmp36-temperature-sensor-with-arduino.

- soil humidity sensor: read the humidity of a plant's soil. The data is read by connecting the blue cable of the sensor to the analog input pin 1 (A1). To calibrate the value, I found the data read from water and then the value of the dry air. The soil humidity will always fall between those. After that I calculated the percentage of moisture we are looking for using map function.

(Soil humidity sensor)

- phototransistor: reads brightness. Connected to analog pin A0. I found that the range of values I can get is between 0 and 1008, which I directly send to Arduino-receiver and after that analyse if they meet conditions for the plant (fall between 200 and 600)

- 5V water pump: transports water from the source to a plant using a common plastic hose. In order to safely connect to the breadboard, I used a transistor. The power cable of the pump is connected to the emitter. Transistor's base leads directly to 5V through a cable and the collector goes to the digital pin 4 through the Resistor of 220 Om.

2) Arduino-receiver

- common cathode RGB LED: depending on the value received from the problem counter, changes colour. 3 legs are connected to digital pins 7, 8, 9 and the longest leg is connected to ground.

- piezo: plays alarm if any problem is encountered. Connected to digital pin 13. I also added a 10kOM Resistor that leads to ground instead of a cable to reduce the volume of the sound it creates.

- LCD: displays the conditions of the plant. I also placed the potentiometer on the board, the centre pin of which is connected to V0 on the LCD, which allows me to change the contrast of the screen.

3) Serial communication

I decided to use the serial communication between Arduino-sender and Arduino-receiver, because it worked well as a way to receive and process data. I needed to connect GND on Arduino-sender to the GND of Arduino-receiver, as well as connecting them by UART(Universal Asynchronous Receiver Transmitter) pins: TX to RX and RX to TX. In order for the data transmission to be successful, the speed of transferring data should be the same, in my case – 9600 baud rates.

Development notes

In order to save memory from using constants to initialize pins, I declared them using #define in the beginning of the code file.

The LCD screen, connected to the Arduino-receiver displays current data received from the serial port, when the user clicks the button.

About watering: although, in my case, a person must physically press the button, the code can be simply changed to enable an automatic watering every 4 hours or as much a user sees as best.

In the beginning I thought about indicating each problem with a light color, but in the process, I found out that is a very bad choice for the user interface, because there would be too much colors and it would be confusing. I decided to change it that the common cathode RGB LED is an indicator if there are any (or no) issues with the conditions for the plant were found. Every time the problem is found, the problem counter is incremented, and, based on that I execute the switch condition, where cases correspond to colour, the LED will take. If more than 2 problems were found, piezo will play an alarm.

To ease the troubleshooting, LCD screen also displays where and what type of problem was found.

Final product

The Arduino-sender stands closer to the plant, because it has sensors which monitor plants conditions, and which should display data as closely to reality as possible.

The Arduino-receiver is closer to user. It has a display, which initially, just shows “The plant care project” title. When user clicks the button, attachInterrupt is called, the function inside lets user list the condition of the plant (humidity, soil, temperature). In order to change the displayed text, were implemented switch-case conditions, where each increments the counter and the last one sets it to 0, in order to loop again through the values when the button is pressed.

In order to distinguish information from the serial correctly, I implemented a function that assigns variables of temperature, humidity and light by analysing the given string.

To detect the problems encountered through the program (whether the humidity is low and the soil is dry or the light value is too high and the brightness should be reduced) I initialized a problem counter variable that looks for the possible issues. In the end of the loop the LED RGB is used to indicate the state of the system by it’s colour. Green is used to indicate the state without problems, yellow (I mixed red and green value in my LED RGB to create it) is used for 1 problem and red is for 2 and more problems. If the light is red, then piezo calls an alarm, in my project I simplified it to the state of two sharp sounds, but I have also commented away an array of notes from the Tetris theme which, so if anyone who uses the program decides it would be better, then they can just uncomment the line and it will work.

As well as piezo and LED RGB implementation for problem detection, user can list through the LCD. Inside each case there are conditions which evaluate if variables are outside norm, and, if so, LCD prints a message on the second line (the first line is always used for the value:

ex. Humidity: 50%) inside exclamation points. So that, in instance, if the soil is dry, “!! too dry !!” is printed under the general value of moisture.

To make LCD more comfortable for the user, a person can use a potentiometer to change the brightness of the text.

Another interesting implementation is the water pump system. When the soil humidity is low, user is allowed to press the button placed on the Arduino-sender. After that the attachInterrupt function is called. The second condition for the attachInterrupt to actually work is if there is a period of a certain time(for example in my program: 4 hours) is completed, which for the first time does not count, bit does for the following ones. It is necessary, so that soil can soak in the water. That is why I set the moisture value immediately to the minimum normal (30%) when the pump has transferred water. Also, when it is 30%, the second Arduino does not detect it as a problem. When the soil’s humidity starts becoming higher than 30, it will also change, but not before.

Conclusion

I created a user-friendly way to monitor and analyse their houseplants’ state. A person can be in the other place of the room, having an Arduino-receiver by their side and still look after their plants with ease. If there are any problems, their attention will be drawn by LED RGB light change and even a piezo, the loudness of which can be regulated by a resistor that connects buzzer to ground, which make user check the LCD and find out the specific problem. If watering is need for a plant, user does not need to measure how much of liquid they need, because it is automatically decided in the program.

If a user does not have a way to press a watering button, they can simply change a few lines of code to make the decision automatic and time-based.

Appendix