HERIOT - WATT UNIVERSITY

School of Engineering & Physical Sciences

Mechanical Engineering Laboratory

Arduino Experiments

Alarms

Problem:

A typical application of a microcontroller is the handling of level alarms. Figure 1 shows a typical level alarm system. The tank has an inflow and an outflow; if the inflow is higher than the outflow the level will rise, and if the outflow is higher than the inflow the level will fall. Alarm sensors have been installed to detect when the level becomes too high (causing an overflow) or too low (causing the tank to become empty, and gas to enter the pipework).

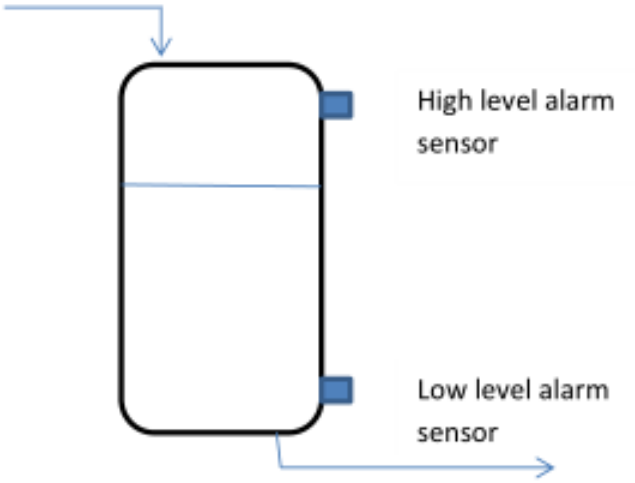


Figure 1: A tank with level alarms

Required Components:

- Arduino UNO
- 3* LED
- 3* 330Ω **resistor for LED**
- 3* 10k Ω **resistor for button**
- 3* Push buttons
- Breadboard
- Jumper cables

Set up:

We will not use actual liquid level sensors but will simulate the output of the sensors using push buttons.

The way a push button works with Arduino is that when the button is pushed, the voltage goes LOW. The Arduino reads this input and reacts accordingly. In this circuit, we will use additional resistors to prevent false readings from the button.

**In Arduino input pins are marked with ~ in front of it*

***Don't forget the negative and positive legs of the LED*

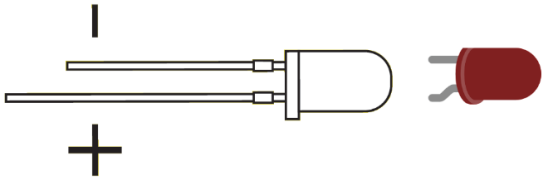
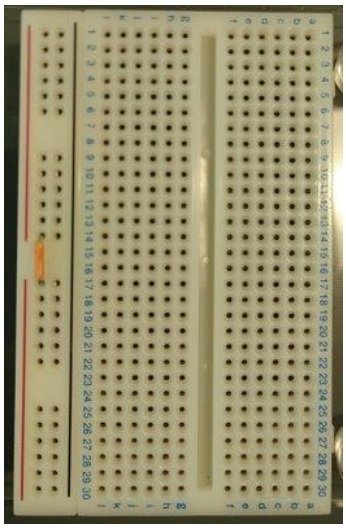


Figure 2: Light Emitting Diode

BEFORE building the rest of the circuit – connect the two parts of the + line of the breadboard together using a small section of jumper wire.



This connects the two isolated sections of the **power bus** (+) on the left-hand side and will allow you to connect the rest of the components correctly.

Task 1:

Use the code from "Blink" (or download the "Alarm" example code) and add two additional inputs for the buttons and an additional LED. Imagine that the buttons in figure 3 are sensors and if pressed they are reading the presence of fluid. When a button is pressed it will give digital LOW.

If pin 11 is at GND (the electronic "low" or digital "0") then the fluid level is close to the top of the tank, and we run the risk of the tank overflowing. If, on the other hand, input pin 10 is at 5V (the electronic "high" or digital "1") then the level is too low, and we run the risk of the tank becoming empty. The upper LED should indicate if the tank is overflowing and the bottom one should indicate if it is empty.

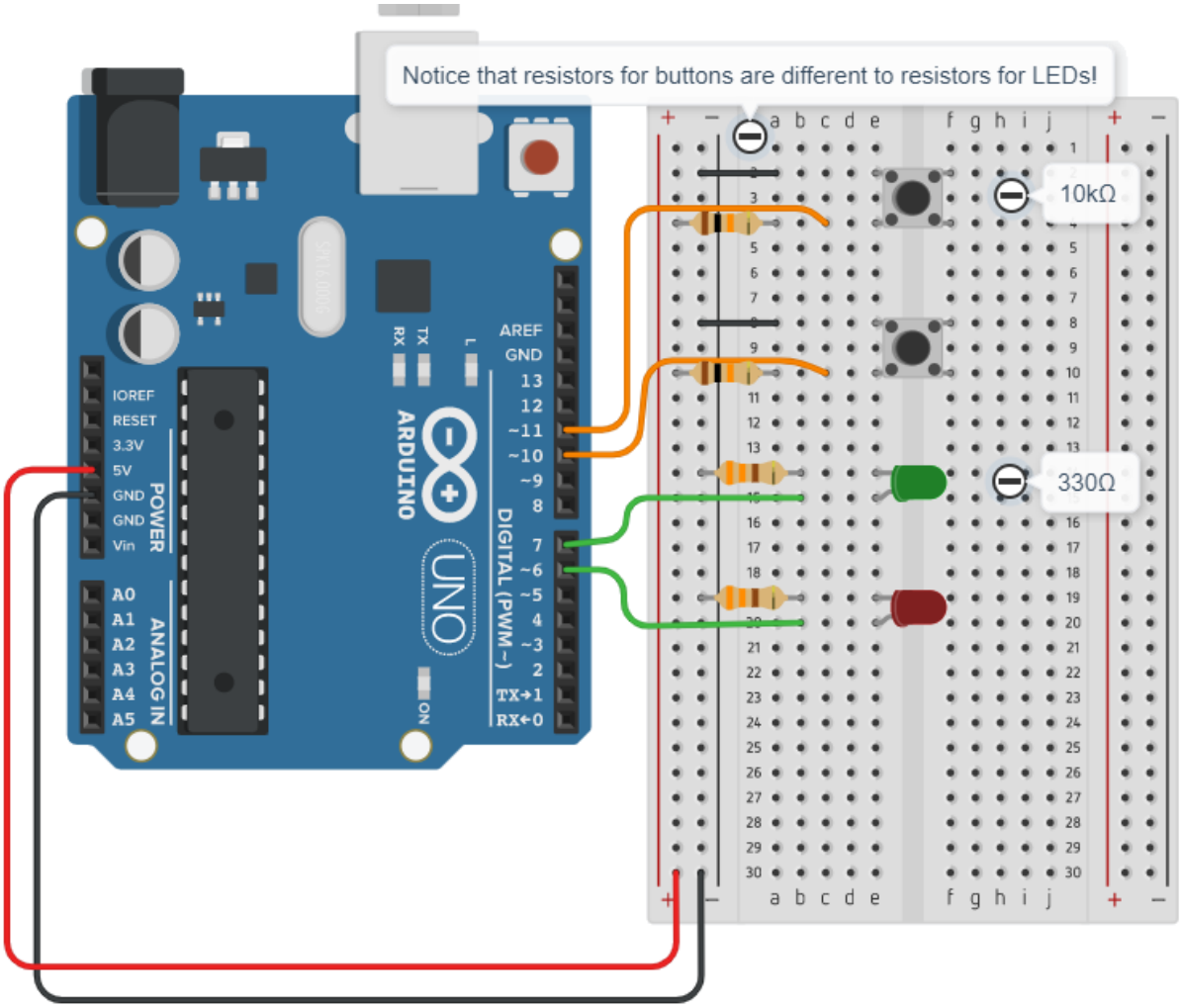


Figure 3: Alarm Circuit Task 1

Task 2:

In the previous exercise we used a very simple alarm system. It is, however, possible to give more information about the level in the tank if we use more LEDs. Figure 4 shows four different ways in which a tank can be filled.

Figure 4 - Possible filling situations in a tank

Using your experience with writing code for the Arduino and constructing circuits, design a system in which:

A. A green LED lights up when the tank is partly filled

B. A yellow LED lights up when the tank is empty
C. A red LED lights up when the tank is overfilled
D. Give an ERROR warning when the top sensor indicates there is liquid in the top of the tank, but the other sensor indicates the bottom part of the tank is empty (this is an impossible situation most likely caused by an error in the alarm system)

Put `Serial.begin(9600);` in `void setup(){…}` and then `Serial.println("ERROR!");` in your code at the appropriate place. This is a fast and visual way to debug your code. Click on the Serial Monitor icon on the top right of the Arduino IDE window to display the Serial Monitor window.

Figure 5: Serial Monitor

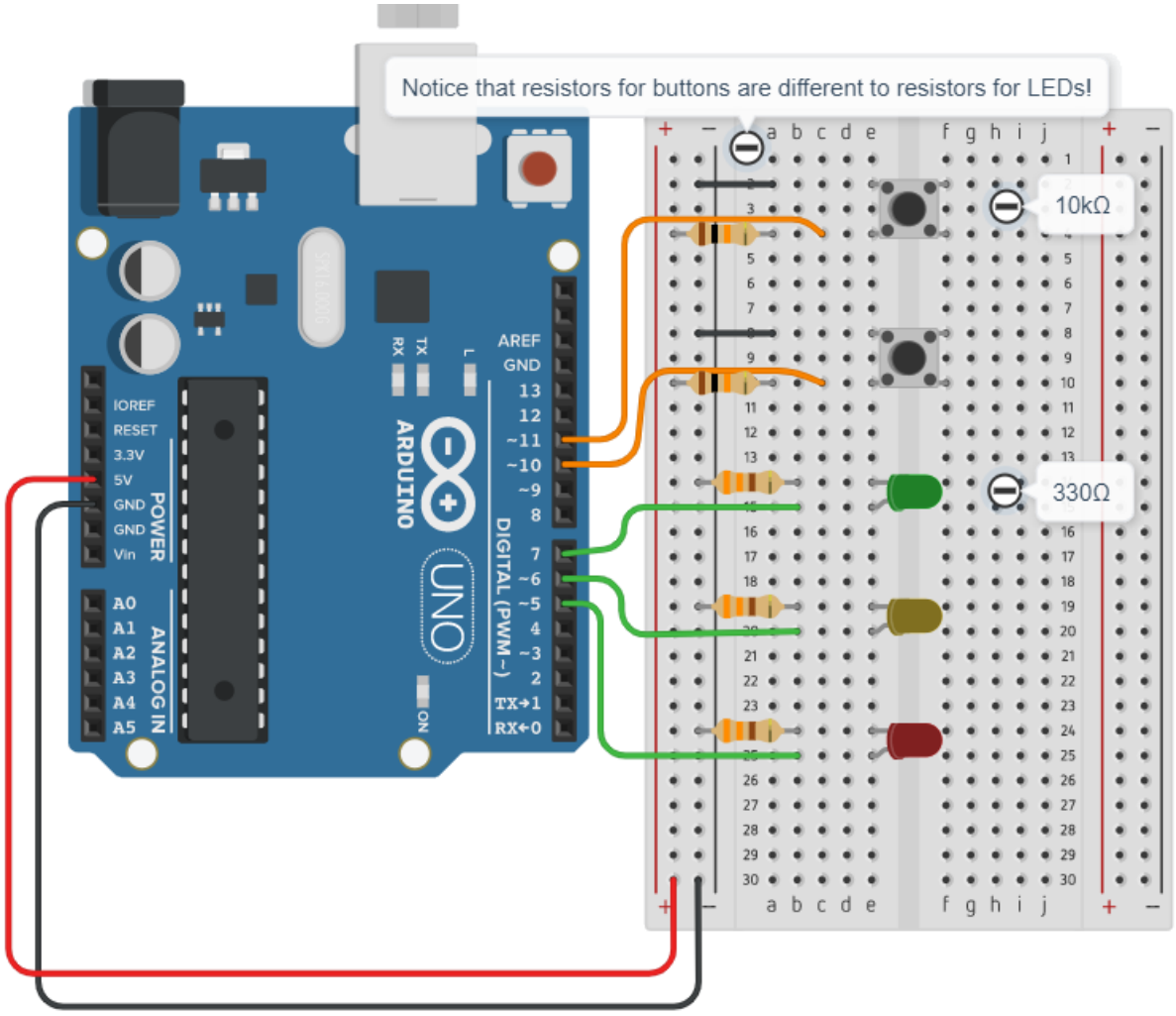


Figure 6: Alarm Circuit Task 2