Alexandria University
Faculty of Engineering
Computer and Systems Engineering Dept.
CSE233: Computer Organization

Lab #7A Report

Remote Control Blink Rate

Names:

1. Mohamed Abdalla Yassen Mohamed (23010765)
2. Ahmed Mohamed Saied Mohamed (23011684)

1. Problem Statement

Implementation of Remote Control Blink Rateusing Arduino.

2. Code “GitHub Rebo”

GitHub:

https://github.com/Mohamed-Abdalla-Yassen/Computer-Organization-Projects.git

Arduino web editor:

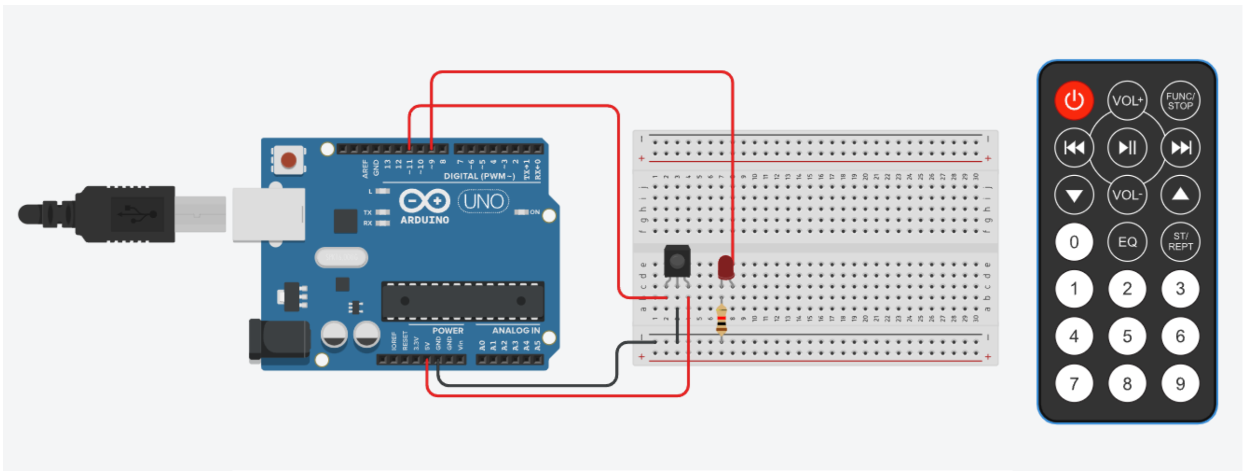
First code: to detect the values of our remote
https://app.arduino.cc/sketches/e5f47836-0056-453b-a830-da8cae528f93?view-mode=preview

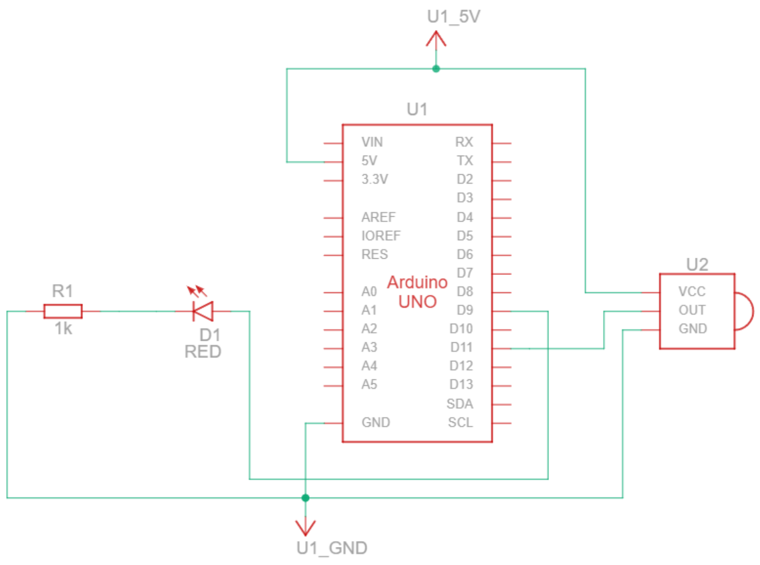
Second code: to implement the remote control blink rate
https://app.arduino.cc/sketches/532e61f1-ba6b-4cfd-98c2-03be0fadd5a6?view-mode=preview

3. Video “YouTube”

https://youtu.be/7N_u0WuayX0?feature=shared

4. Schematic diagram





5. Description & Challenges

Description:

The objective of this lab was to use an **IR remote control** together with an **IR receiver (TSOP1838)** to control the blink rate of an LED using the Arduino. The IR receiver detects the encoded infrared signals transmitted by the remote buttons and converts them into digital data that can be read by the Arduino through the **IRremote library**.
In this experiment, each numeric button on the remote (0–9) was mapped to a specific delay value that determined the blinking speed of the LED. For example, pressing button **1** set the blink rate to **100 ms**, pressing **5** set it to **500 ms**, and pressing **9** set it to **900 ms**. The decoded IR command values were printed on the Serial Monitor to verify correct reception, and the LED blinked at different speeds accordingly.

Implementation –Signal Processing then Mapping
This part involved writing the Arduino program to receive and decode the IR signals using the **IRremote (v4.x)** library. The library was initialized using IrReceiver.begin() to activate the receiver on pin 9. Inside the main loop, the Arduino continuously checked whether a new IR command had arrived using IrReceiver.decode().
When a remote button was pressed, the command value was extracted using IrReceiver.decodedIRData.command and displayed on the Serial Monitor for verification. Because different remotes produce different command values, each numeric button (0–9) was manually mapped to a specific LED blink delay.
This allowed real-time control of the LED's blink speed based on the remote input. Finally, the LED was toggled ON and OFF with the calculated delay, completing the functional behavior of the system.

Table of Remote Values and Corresponding Blink Delay

Remote Button	Received Command Value	Blink Delay (ms)
1	4	100 ms
2	5	200 ms
3	6	300 ms
4	8	400 ms
5	9	500 ms
6	10	600 ms
7	12	700 ms
8	13	800 ms
9	14	900 ms
0	17	100 ms (treated as 1)

Challenges:

During the lab, several challenges were encountered:

1. Different Remote Controls Produce Different Command Codes

Not all remotes use the same IR protocol or command values.
When reading the button values, the numeric buttons on the remote produced unexpected codes such as 4, 5, 6, 8, 9 instead of standard 1–9.
This required manually printing all received command values in the Serial Monitor and mapping each one to a blink delay. Without this debugging step, the LED would not respond correctly.

2. IR Receiver Sensitivity

During the experiment, an unexpected challenge appeared while recording the demonstration using a mobile phone. The IR receiver (TSOP1838) is highly sensitive to infrared light, and the phone's camera was emitting **constant IR pulses** that the sensor interpreted as valid signals. As a result, the receiver was continuously detecting unwanted IR activity, causing false readings in the Serial Monitor even when no remote button was pressed.

This interference made it difficult to observe the actual remote commands, since the sensor was receiving both the intended signals from the remote and the unintended IR noise from the phone. To overcome this issue, the recording angle had to be adjusted so that the phone was not directly facing the IR receiver, and the experiment was repeated while keeping the phone farther from the line-of-sight of the sensor.

5. Adapting to the New IRremote Library Syntax

The syntax for versions 4.x of the IRremote library is different from the older versions used in most online tutorials.
Examples using IRrecv and decode_results no longer work, so the new methods (IrReceiver.begin(), decodedIRData.command) needed to be learned and applied.