School of Science, Computing and Engineering Technologies

**ENG200010**

**Engineering Technology Design Project**

**Assignment 5 Report**

# **Project Title:** IR Sensor and LED Control Using LabVIEW

**Student Name:** Arnob Ghosh

**Student ID:** 103494114

**Date:** 29/09/2023

**Declaration**

I declare that this report is my individual work. I have not copied from any other student's work or from any other source except where due acknowledgment is made explicitly in thetext, nor has any part of this submission been written for me by another person.

**Signature:** Arnob

# Table of Contents

# Introduction

The design and execution of a LabVIEW programme for controlling LEDs depending on the trigger count of an IR sensor are thoroughly examined in this study. Making a complex system with real-time monitoring and control capabilities is the aim of this task. It intends to illustrate sensor connection and automation using LabVIEW, a potent graphical programming language.

# Methodology

## Hardware Setup

The hardware setup was the first important stage in this endeavour. I made use of an IR sensor with a solid reputation for accuracy. Data acquisition was made simple by connecting the sensor to a LabVIEW hardware platform that worked with it. The sensor was tuned to precisely identify changes in closeness.

# LabVIEW Interface Design

The LabVIEW user interface is at the centre of this project. I spent a lot of effort making an intuitive and educational interface. The following components are precisely shown on the LabVIEW front panel:

Trigger Count: A dynamically updating numerical indicator that displays the number of triggers currently being detected by the IR sensor.

LED Status: A visual indication that shows if each LED (Red, Amber, or Green) is "ON" or "OFF."

Reset Button: This button disables all LEDs and allows the user to manually reset the trigger count.

Users may watch the system's operation and engage with it thanks to the interface's properly thought-out real-time feedback mechanism.

# Programming Logic

The block diagram was used to create the fundamental logic of the LabVIEW programme. The magic happens in the block diagram, which contains the graphical representations of the program's many components and linkages. The programming logic is broken down in greater depth below:

1. **Acquisition of sensor data**

   I used LabVIEW's data collecting capabilities to continually track the IR sensor's output. The LabVIEW input channels were linked to the sensor's output, enabling real-time data retrieval.

2. **Setup Counting**

   I used a counter variable to keep track of how frequently the IR sensor was activated. This counter played a crucial role in the rationale. Every time the IR sensor picked up a nearby item, it increased by one.

3. **Control of LED**

   I utilised LabVIEW's built-in digital output routines to control the LEDs. As stated in the assignment specifications, LabVIEW delivered signals to the appropriate LEDs, turning them on or off, when the trigger count reached certain thresholds (2 for Red, 5 for Amber, and 8 for Green).

4. **Mechanism for Reset**

   The task required that the counter reset itself when it hit 10. A conditional statement was added to verify the counter value. When the counter reached 10, LabVIEW carried out a reset instruction, which reset the number to 0 and turned out all LEDs.

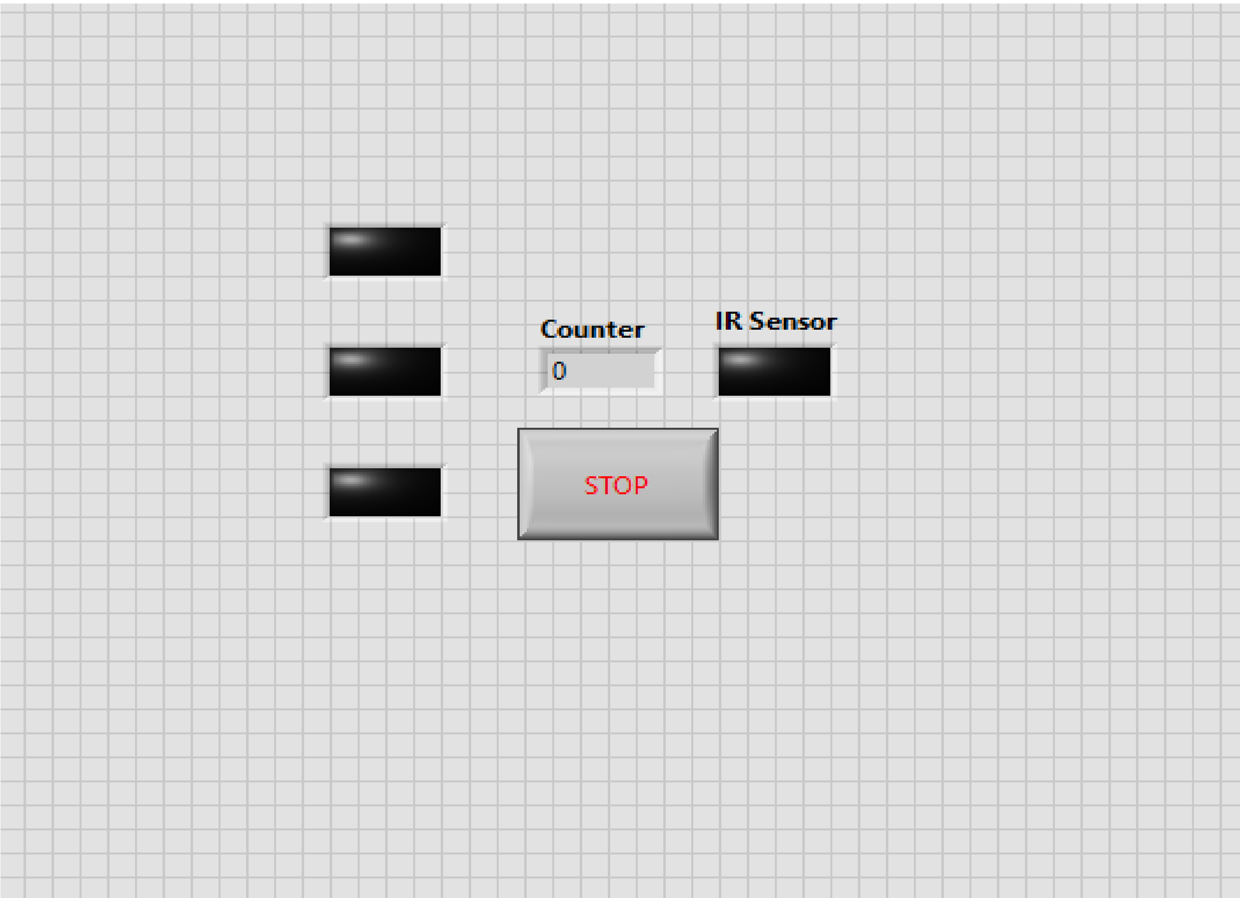
5. **Operation in Real Time and Looping**

   I surround the entire algorithm in an endless loop to guarantee the system's continued operation. With the help of this loop, the system was able to continuously monitor the sensor, keep track of triggers, operate LEDs, and reset the counter.

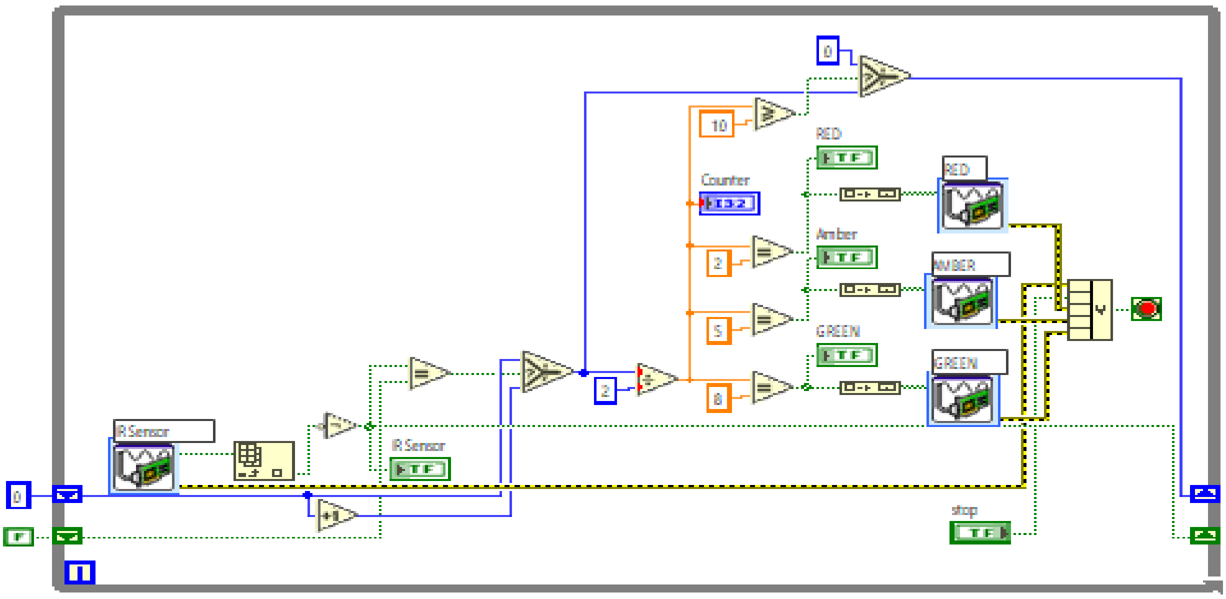
# LabVIEW Program Screenshots

To provide a visual representation of the developed LabVIEW program, I have included the following screenshots:

**LabVIEW Interface**



**LabVIEW Block Diagram**




# Execution and Understanding

Executing the LabVIEW program provided valuable insights into its real-world application and functionality. Here is a more comprehensive understanding of the system's behavior:

1. **Sensor Reactivity:** The infrared (IR) sensor had remarkably good responsiveness, precisely identifying things in its vicinity.

2. **Real-time Trigger Count:** I was able to closely monitor the system's performance since LabVIEW showed the trigger count in real-time.

3. **LED Control Accuracy:** The LEDs lit up accurately when the predetermined counts were achieved and responded quickly to the trigger count thresholds.

4. **Reset Functionality:** When the counter hit 10, the system made sure it was successfully reset and that the LEDs turned off as predicted.

5. **Continuous Operation:** The LabVIEW program's infinite loop allowed the system to run continuously, turning it into a standalone automation solution.

# Conclusion

This paper explored the detailed particulars of creating and putting into use a LabVIEW programme for IR sensor-based LED control. The approach included hardware configuration, painstaking LabVIEW interface design, and a thorough analysis of the programming logic. Real-world sensor interface and automation were better understood thanks to the LabVIEW program's execution.

This task demonstrated the flexibility and strength of LabVIEW as a tool for creating complex control systems. It also highlighted how crucial user-friendly interfaces are to automation since they enable smooth user-system interaction.

# References:

Edit *et al.*, "Traffic Light State Machine Demo Using LabVIEW," *forums.ni.com*, Jul. 16, 2014. https://forums.ni.com/t5/Example-Code/Traffic-Light-State-Machine-Demo-Using-LabVIEW/ta-p/3523962

K. Nellore and G. Hancke, "Traffic Management for Emergency Vehicle Priority Based on Visual Sensing," *Sensors*, vol. 16, no. 11, p. 1892, Nov. 2016, doi: https://doi.org/10.3390/s16111892.

"Infrared Sensor," *forums.ni.com*, Feb. 11, 2011. https://forums.ni.com/t5/LabVIEW/Infrared-Sensor/td-p/1449142 (accessed Sep. 29, 2023).

**“Product Documentation - NI,” *www.ni.com*. https://www.ni.com/docs/en-US/bundle/labview/page/sequence-structures-executing-sections-of-code-sequentially.html#:~:text=A%20sequence%20structure%20contains%20one (accessed Sep. 08, 2023).**