Traffic Light

Firmware Documentation

CONTENTS:

System Description

This traffic light system is mainly implemented using (ATmega 32) microcontroller, the firmware is written from scratch using C programming language without the aid of any pre-built libraries. The program is divided into smaller drivers to control the different protocols and connected hardware.

The function of this traffic light system is to simulate the real-world traffic control systems using six LEDs, three of them for cars, and the rest for pedestrians. The system waits 5 seconds only for each colored LED, green, yellow, and red. For pedestrians, there is a button to be pressed for individuals to use before crossing the street. Pressing the button will inform the system that a pedestrian is waiting to cross the street. The individual should cross the street only when the green pedestrian LED is powered on, at the same time, the car's red led will be on.

System flow is mainly divided into two modes, the normal mode, in which the cars LEDs will continue to change in the normal order, green, blinking yellow, and finally red, each for five seconds only, the other mode is the pedestrian mode, which allows pedestrians to cross the street. When an individual presses the pedestrian button while cars red LED is on, the pedestrian green led will be powered on to allow him crossing the street while red cars led is still on, then a smooth transition from the pedestrian mode to normal mode, if the pedestrian presses the button while the cars LED is green or yellow, a smooth transition is made to power on the pedestrian green led to allow the individual to pass.

System Design

Hardware

The system is built using ATmega-32 microcontroller, six colored LEDs, 1 button, and wires for connections.

Software

System software design is implemented using SOLID principles and dividing the system into small drivers to control the different protocols and connections.

- Hardware Abstraction Layer (HAL)
    - Microcontroller Abstraction Layer (MCAL)
        - DIO Driver
        - Interrupt Driver
        - Timer Driver
    - Electronic Units Abstraction Layer (ECUAL)
        - LEDs Driver
        - Button Driver
- Software App
- Main Driving Software

Following is a description for each designing block:

HAL: Contains MCAL and ECUAL, and represents a blind interface to the hardware, facilitating the application development.

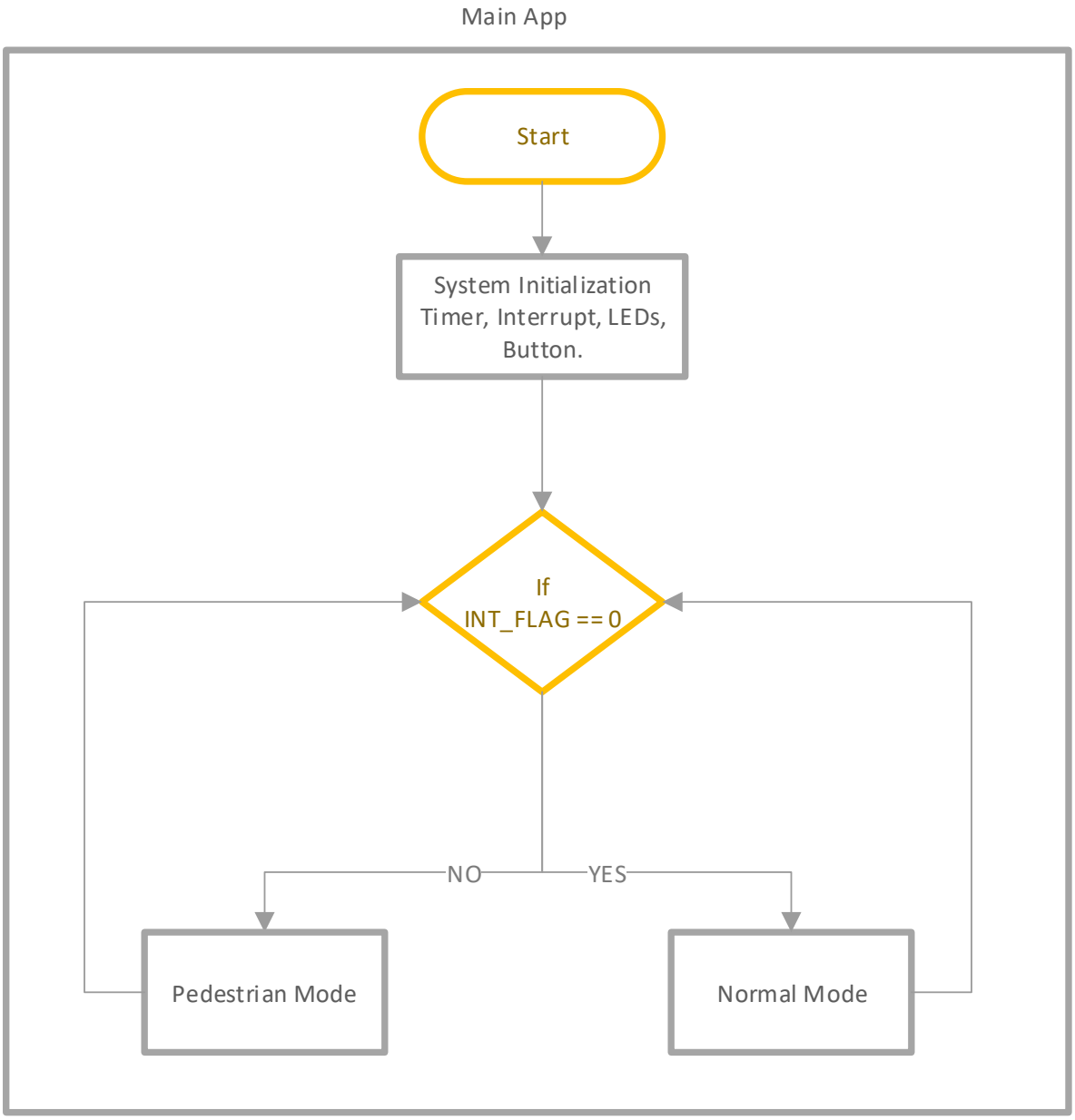
MCAL: Contains internal drivers to the microcontroller, DIO, Interrupt, and Timer drivers, each with ".c" and ".h" files.

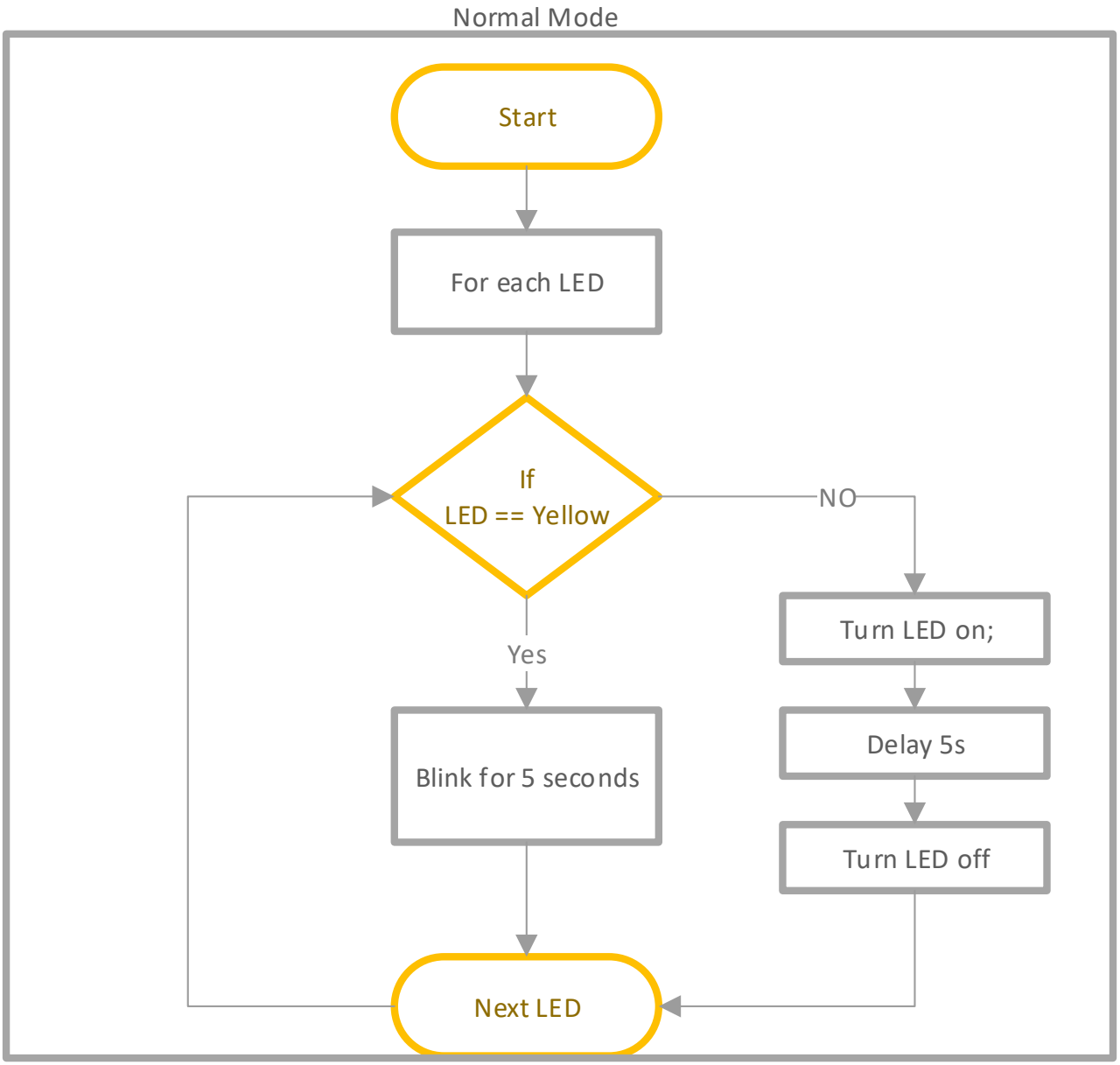
ECUAL: Contains external electronic component drivers, namely, LEDs, and button drivers, each with ".c" and ".h" files.

Software App: Contains the function which describes the program flow, also contains two files, namely ".c" and ".h" files.

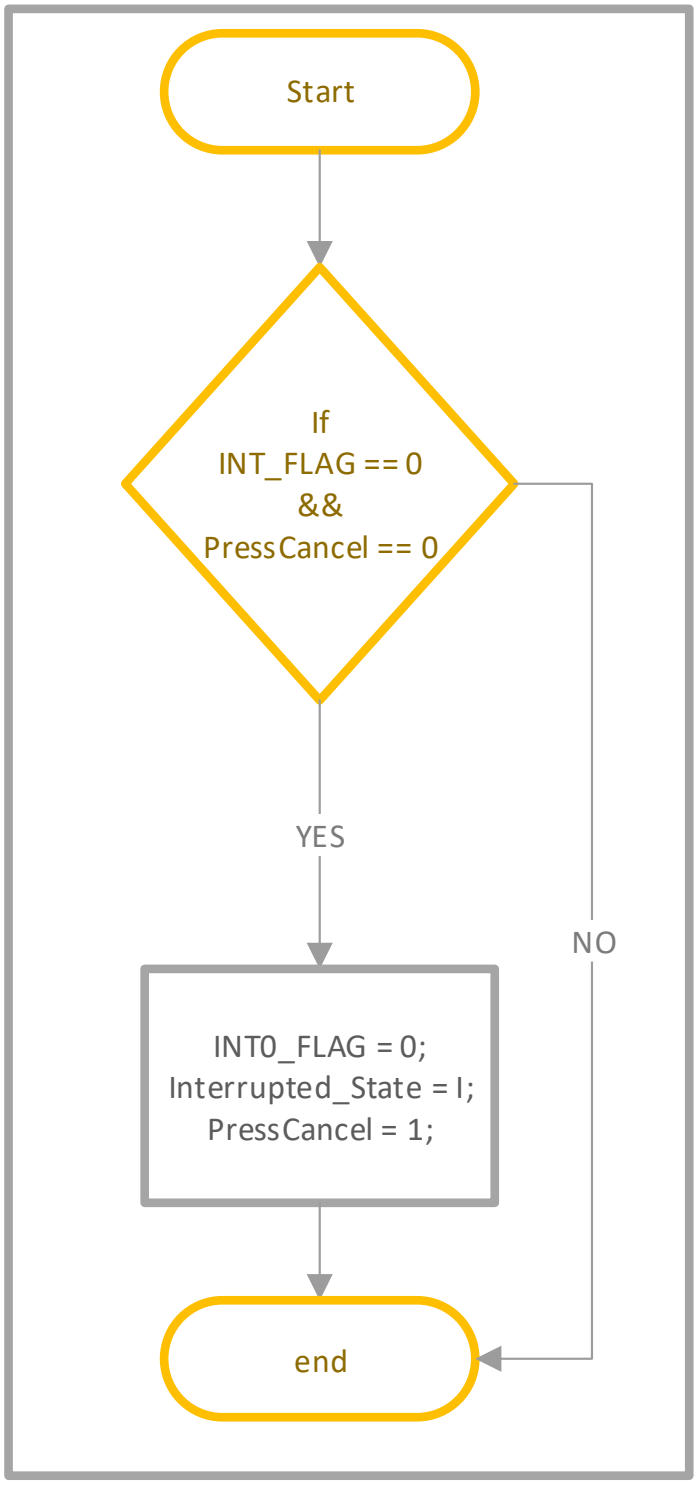
Main Driving Software: the main.c function which calls the software app function.

System Flow Charts

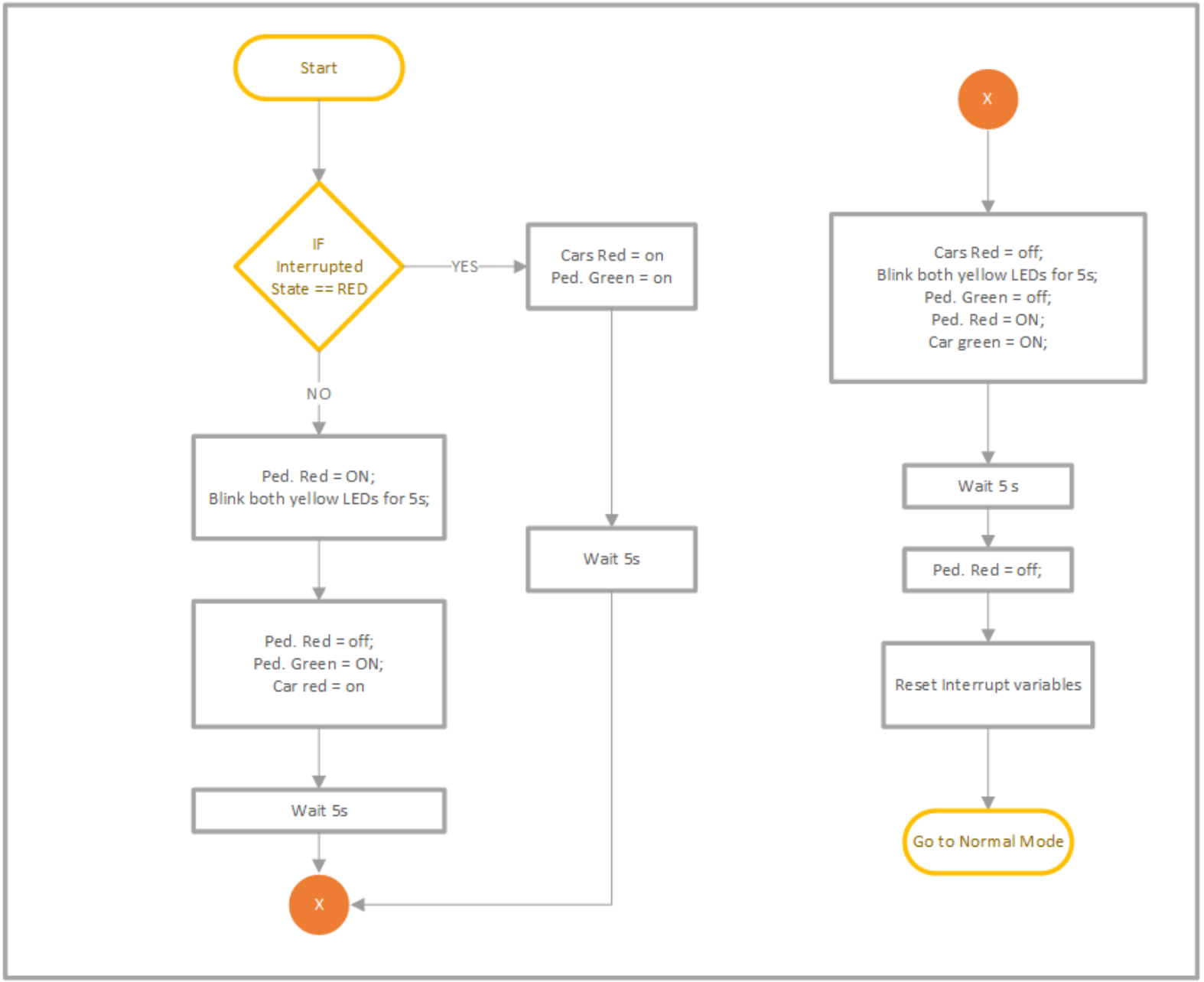



Normal Mode
Start
For each LED
If
LED == Yellow
NO
Yes
Turn LED on;
Delay 5s
Turn LED off
Blink for 5 seconds
Next LED

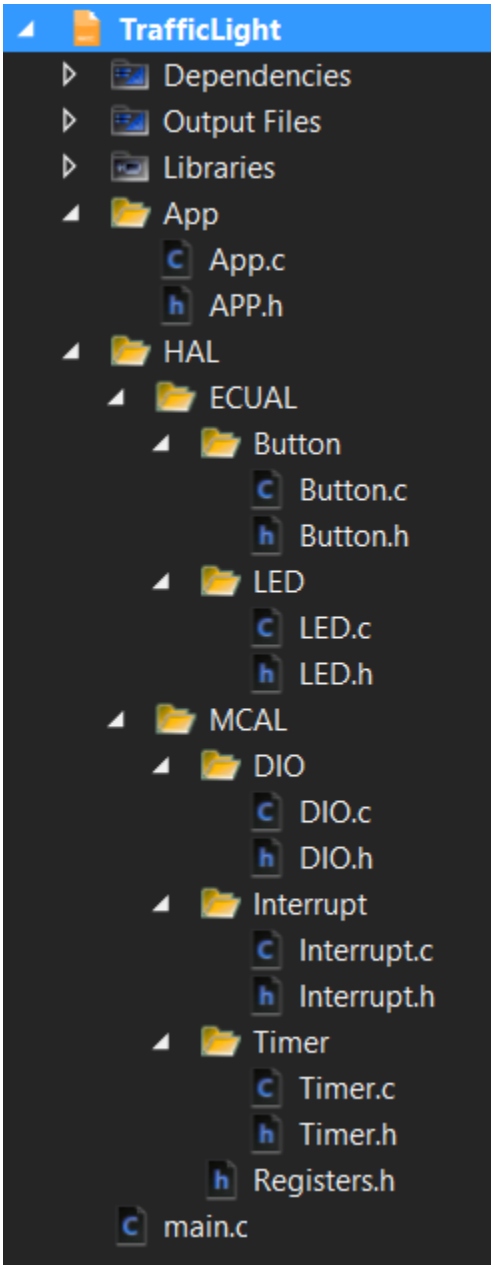
Interrupt Service Routine



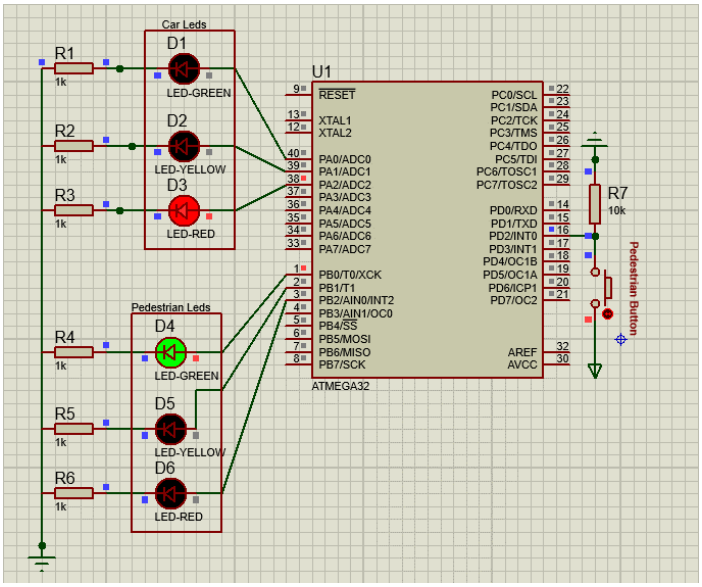
Pedestrian Mode

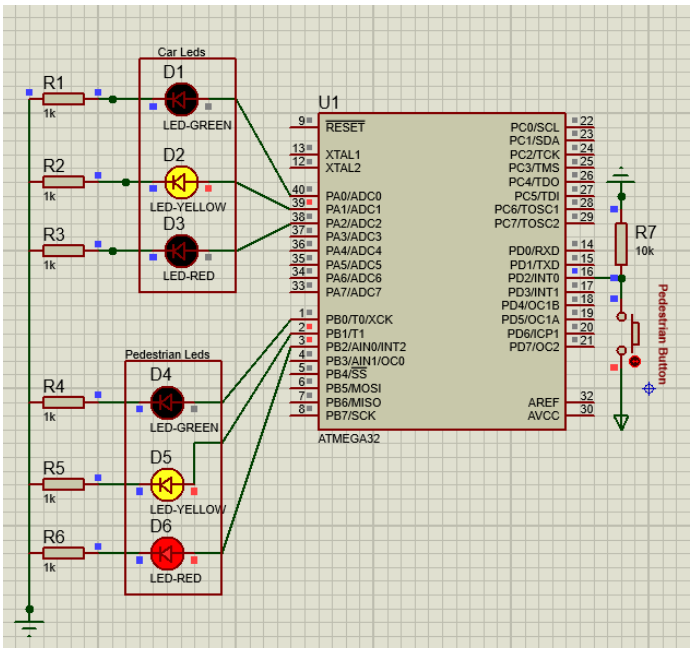


Screenshots



Solution Explorer





System Future Updates

In this section, we will introduce some future development work ideas to update the system behavior and get the best use of the applied hardware and broaden the usage of the system by increasing its features.

- **Feature 1**
  7-Segment Displays to display counting stages on them for both cars and pedestrians, it could be with multiple colors (RGB) to indicate the state of the LED, or a classical 7-seg.
- **Feature 2**
  Wired / Remote control station to control the state of the traffic light system, know which state is on, change it depending on the traffic statistics and jam problems, or connecting the traffic system to the internet through a connected network, which introduces a wide range of new features.
- **Feature 3**
  Connecting a camera with or without movement detection protocols, to detect any traffic accidents or issues. Features 2, and 3 will allow using AI algorithms, to collect more accurate data, and conclude better judgmental decisions.
- **Feature 4**
  Connecting the area's traffic light systems together to adjust green/red timings automatically in different places according to the gathered statistics and the smart algorithms to be applied.

These 4 features are examples to develop classical traffic light systems, however, the real-life problems for each certain area/time induces new ideas and new solutions for those certain needs and dependencies.

AbdulRahman AlSindiony

Electronics and Communications Engineering

Undergraduate Student

Faculty of Engineering – Alexandria University

es-abdulrahman.hasan2024@alexu.edu.eg