# Udacity On Demand Traffic Project

GitHub: https://github.com/melgmry0101b/udacity-on-demand-traffic

By: Mohammed Elghamry c_elghamry96@outlook.com

## Rubric Videos

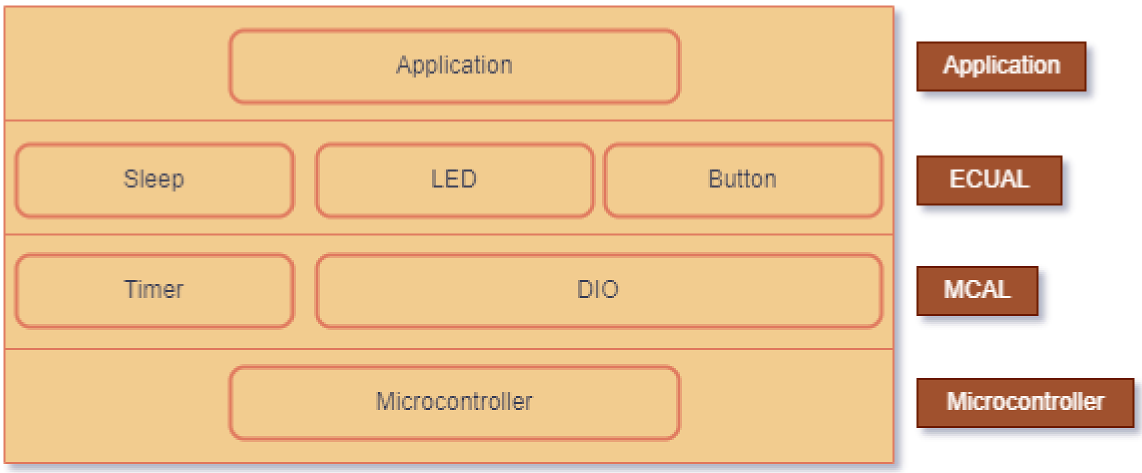
- System Design (6:35): https://github.com/melgmry0101b/udacity-on-demand-traffic/blob/main/docs/vids/1_vid_system_design.mp4
- Development Environment (4:39): https://github.com/melgmry0101b/udacity-on-demand-traffic/blob/main/docs/vids/2_vid_dev_env.mp4
- Application Implementation (19:11, Drivers testing starts @ 14:25): https://github.com/melgmry0101b/udacity-on-demand-traffic/blob/main/docs/vids/3_vid_implement_app.mp4
- Extended Driver Testing (9:14): https://github.com/melgmry0101b/udacity-on-demand-traffic/blob/main/docs/vids/3_vid_extended_testing.mp4
- Application User Stories (4:28): https://github.com/melgmry0101b/udacity-on-demand-traffic/blob/main/docs/vids/4_vid_test_user_cases.mp4

## System Description

This system is a traffic control system that prioritizes pedestrians. At designated road crossings with traffic lights, pedestrians can request green light for crossing on-demand. Our system, in normal mode, moves car traffic lights from rad to yellow then green and vice versa, with the opposite happening for the pedestrians' lights. Each stage lasts for five seconds, noting in yellow stage the light blinks. When the pedestrian presses a request button for crossing, the system enters pedestrian mode. In pedestrian mode, if the car lights were red or yellow moving to red, the system resets the five seconds interval. If the car lights were yellow moving to green, the system resets the interval and then moves to red for cars. If the car lights were green, the system moves immediately to yellow with five seconds blink then to red. The system returns to normal operation afterwards.

## System Design

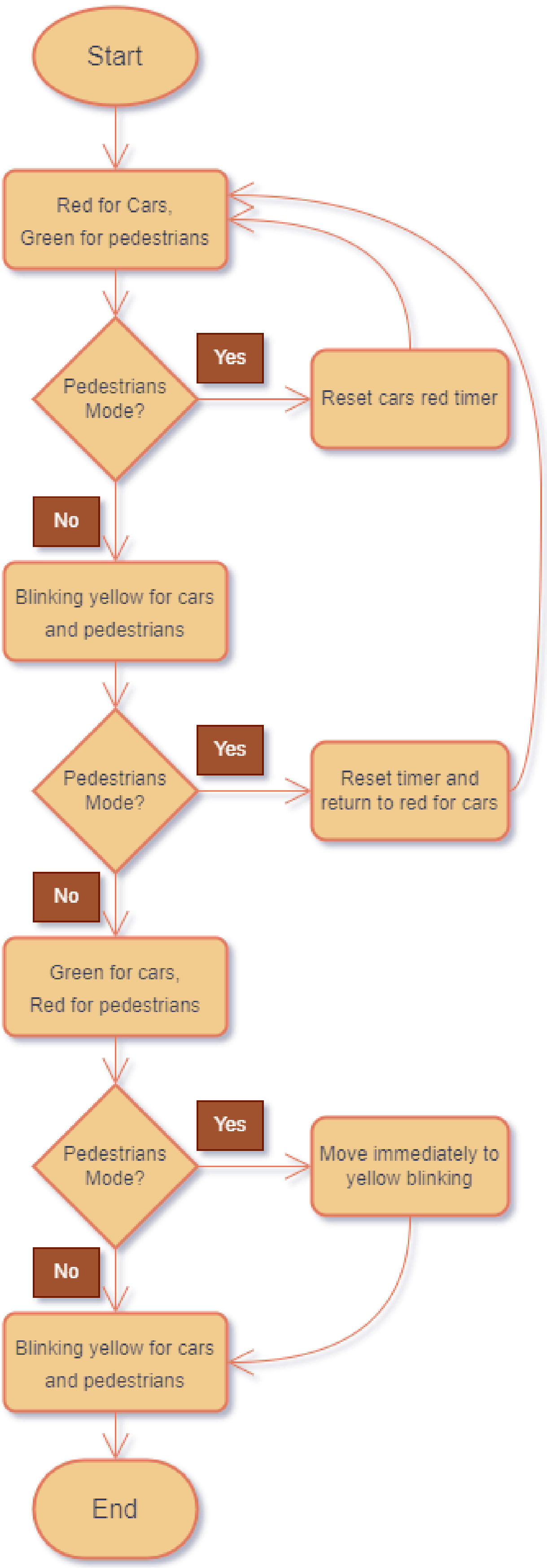
The system uses ATmega32 microcontroller as the embedded brain. We have three major layers on-top of the microcontroller: the microcontroller abstraction layer (MCAL), the electronic unit abstraction layer (ECUAL), and the application. Our MCAL has two drivers: the digital input/output driver (DIO), and the timer diver. The ECUAL has three higher-level drivers: the LED driver, the sleep driver, and the button driver. Our system embraces SOLID principles preventing higher levels from direct communication with any lower level other than the one directly below.



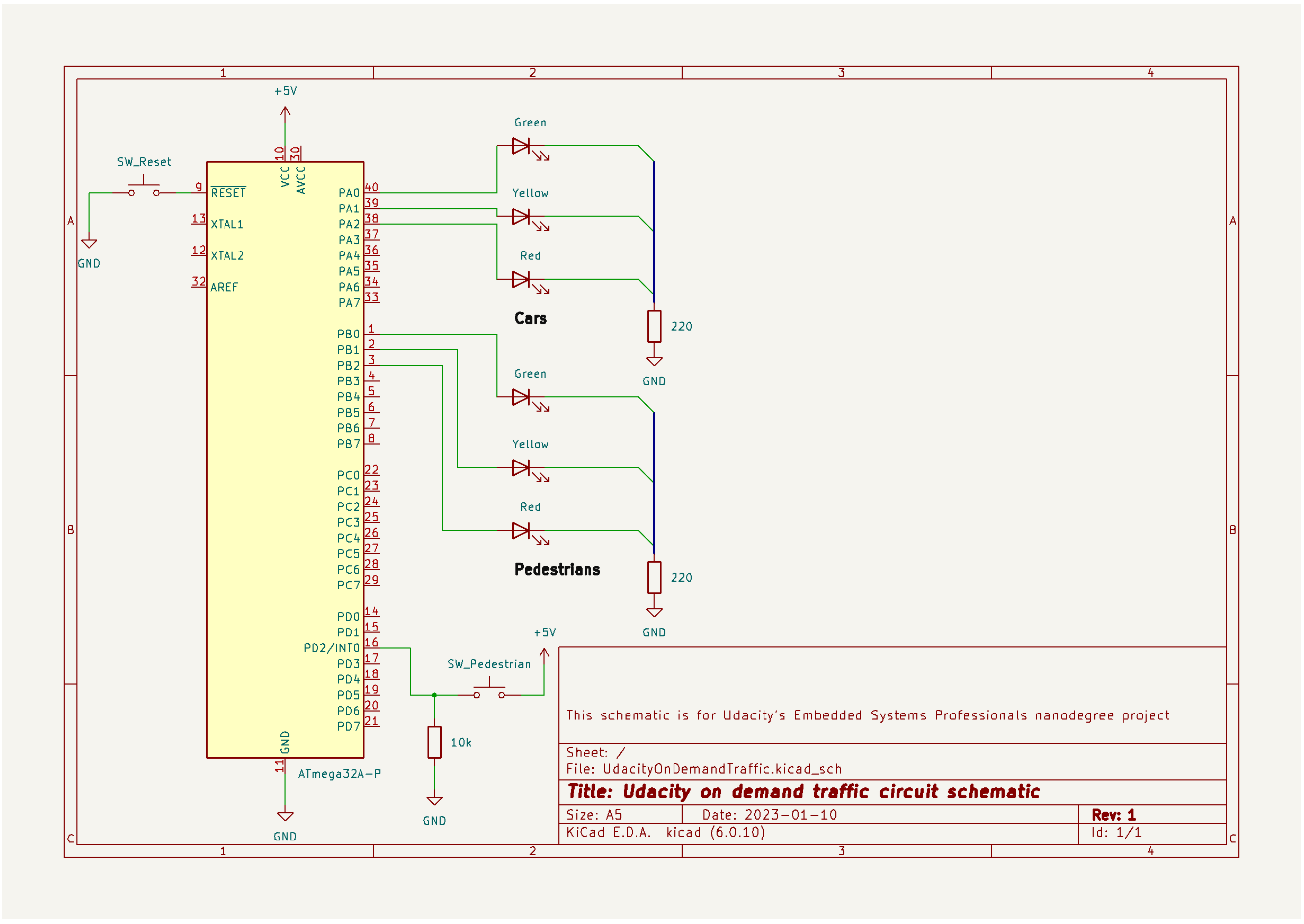
## System Constraints

One major constraint of the current implementation is that the user can put the system into a loop between red light for cars and blinking yellow. If the user pressed the pedestrian button while the system is blinking yellow going into the green light for cars, the system will change course and return to the red light for cars. The user can do this each time the lights are yellow. This "flaw" can be mitigated by extending the pedestrian mode to cover two processing cycles so that the system doesn't register another press in this period. Other constraints include: if the user pressed the button for very short time, the system may not be able to register it by being into an ISR or couldn't sense the change in voltage. The system doesn't have specific constraints regarding long or double presses.

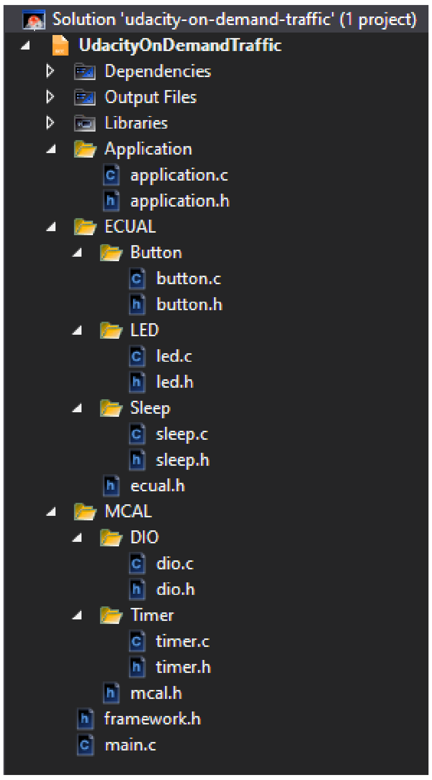
## System Flowchart



## Demo Circuit


+5V
SW_Reset
RESET
XTAL1
XTAL2
AREF
GND
VCC
AVCC
PA0 PA1 PA2 PA3 PA4 PA5 PA6 PA7
PB0 PB1 PB2 PB3 PB4 PB5 PB6 PB7
PC0 PC1 PC2 PC3 PC4 PC5 PC6 PC7
PD0 PD1 PD2/INT0 PD3 PD4 PD5 PD6 PD7
ATmega32A-P
Green
Yellow
Red
Cars
220
GND
Green
Yellow
Red
Pedestrians
220
GND
+5V
SW_Pedestrian
10k
GND
GND
This schematic is for Udacity's Embedded Systems Professionals nanodegree project
Sheet: /
File: UdacityOnDemandTraffic.kicad_sch
Title: Udacity on demand traffic circuit schematic
Size: A5
Date: 2023-01-10
Rev: 1
KiCad E.D.A. kicad (6.0.10)
Id: 1/1

## Folder Structure



```
Solution 'udacity-on-demand-traffic' (1 project)
  UdacityOnDemandTraffic
    Dependencies
    Output Files
    Libraries
    Application
      application.c
      application.h
    ECUAL
      Button
        button.c
        button.h
      LED
        led.c
        led.h
      Sleep
        sleep.c
        sleep.h
      ecual.h
    MCAL
      DIO
        dio.c
        dio.h
      Timer
        timer.c
        timer.h
      mcal.h
    framework.h
    main.c
```