**On-demand Traffic light control.**

1. *System description.*

An On-demand Traffic light control system normally operates on normal mode in which the cars LEDs will illuminate for 5 seconds straight except the yellow LED which will blink for the same amount of time unless the pedestrian clicks on the button which will switch the mode to pedestrian mode in which both the pedestrian and car LEDS will illuminate in parallel but different colors except for yellow which will remain the same for both sides.

1. *System design****.***

|  |
| --- |
| *Application* |
| *ECUAL* |
| *MCAL* |
| *Atmega32* |

In this system we are using the micro-controller Atmega32 in the microcontroller layer.

In the MCAL layer we implement our drivers which are DIO ,Timer ,interrupt , and registers where we define our registers.

In the DIO driver we implement four functions.

void dio\_init(uint8\_t portNum,uint8\_t pinNum, uint8\_t direction); // function to initialize DIO pin.

void dio\_write(uint8\_t portNum,uint8\_t pinNum,uint8\_t value); // function to set the pin's value.

void dio\_read(uint8\_t portNum,uint8\_t pinNum, uint8\_t \*value); // function to read the pin's value.

In the timer driver we only need a delay function in milliseconds. In this project we use timer0 in CTC mode with a CLK/8 prescaler in order to increase the accuracy of our timer.

void delayms(); // to delay the required time in ms.

In interrupt driver we need a ISR to handle the pedestrian mode when the pedestrian clicks the push button.

void init\_interrupts(); //function to initialize interrupts.

In the ECUAL layer is where we define our external components like the

Push button and the LEDs and there function.

So the driver for the LEDs will contain the following functions.

void LED\_init(uint8\_t port\_number,uint8\_t pin\_number); // to initialize.

void LED\_on(uint8\_t port\_number,uint8\_t pin\_number); // to turn on led.

void LEDs\_off(uint8\_t port\_number); // to turn off led.

And as for button driver we won’t need to implement any new

Functions as the interrupt module will be enough.

In the APP layer we will initialize the required components for the project.

void APP\_init(); // to initialize our app.

void APP\_start(); // function to start our application.

1. *System flow chart****.***

Diagram

Description automatically generated

1. *System constraints****.***
2. Change from normal mode to pedestrian mode when the pedestrian button is pressed.
3. If pressed when the cars' Red LED is on, the pedestrian's Green LED and the cars' Red LEDs will be on for five seconds, this means that pedestrians can cross the street while the pedestrian's Green LED is on.
4. If pressed when the cars' Green LED is on or the cars' Yellow LED is blinking, the pedestrian Red LED will be on then both Yellow LEDs start to blink for five seconds, then the cars' Red LED and pedestrian Green LEDs are on for five seconds, this means that pedestrian must wait until the Green LED is on.
5. At the end of the two states, the cars' Red LED will be off and both Yellow LEDs start blinking for 5 seconds and the pedestrian's Green LED is still on.
6. After the five seconds the pedestrian Green LED will be off and both the pedestrian Red LED and the cars' Green LED will be on.
7. Traffic lights signals are going to the normal mode again.