**On demand traffic light system control**

Project Description

The project aim is to make traffic light system for cars and pedestrian, So we have two modes normal mode and pedestrian mode. First the normal mode it works as follows the system consist of three LED (Green, Red, Yellow) for cars and another three for pedestrian in normal mode the green light for the cars will be on for 5 seconds then both yellow LED lights will blink for 5 seconds then the green led for the pedestrian will go on and the red LED light for car will also turns on. For the pedestrian mode there will be a push button if this push button is pressed the system will change to both yellow LED will blink and after 5 seconds the green led of the pedestrian will go on. But if someone pressed the button two times it will be ignored.

**System Constrains**

One of the main constrain is that this is my first embedded project I ever made so it was hard in the beginning to built the drivers and test them but after watching the tutorials it made it easier after that the second constrain is dealing with the interrupts it was a challenge to make the driver and understand it.

System Design

First, I had to make some drivers before writing the application like:

Register: I/O registration for PORTS (A, B, C, D).

Timer0 registration.

External interrupts registrations.

**MCAL Layer:**

1.DIO driver:

void DIO\_init(uint8\_t pinNumber, uint8\_t portNumber , uint8\_t direction)

(To initialize the Digital input output).

void DIO\_write(uint8\_t pinNumber, uint8\_t portNumber , uint8\_t value);

(This to write in the pin and set if ON or OFF).

void DIO\_toggle(uint8\_t pinNumber, uint8\_t portNumber);

(Toggle the state of pin).

void DIO\_read(uint8\_t pinNumber, uint8\_t portNumber , uint8\_t \*value);

(This function used to read the pin).

2.Interupt driver:

This library defines the external interrupt vectors, macros to set OR clear global register and ISR macro o Make function to initialize the interrupt

3. Timer driver:

void Timer0\_init();

(This function is used to initialize timer 0).

void TIMER0\_start(uint16\_t preScalar, uint32\_t number\_of\_overflow);

(This function is used to set the prescaler and get number of overflow).

void Timer\_stop();

(This function is used to stop the timer)

void delay\_5s(void);

(This function is used to set the timer delay for 5 seconds).

void Timer\_delay(uint16\_t delay);

(This function is used to give the timer the number of microseconds you want to delay).

**ECUAL Layer**

LED Driver:

void LED\_init(uint8\_t ledPort ,uint8\_t ledPin);

(This function is to initialize LED port and pin).

void LED\_on(uint8\_t ledPort ,uint8\_t ledPin);

(This function is used to turn on the LED sends 1 the pin).

void LED\_off(uint8\_t ledPort ,uint8\_t ledPin);

(This function is used to turn off the LED sends 1 the pin).

void LED\_toggle(uint8\_t ledPort ,uint8\_t ledPin);

(This function is used to toggle the pin ).

void toggle\_LED\_5s(uint8\_t ledPort, uint8\_t ledPin);

(This function is used to set a timer delay for 5 seconds and toggle the pin).

void toggle\_2\_LEDs\_5s(uint8\_t ledPort1, uint8\_t ledPin1, uint8\_t ledPort2, uint8\_t ledPin2);

(This function is used to set a timer delay for 5 seconds and toggle the pin for two leds).

Button Driver:

void BUTTON\_init(uint8\_t buttonPort, uint8\_t buttonPin);

(This function is used to initialize the button).

void BUTTON\_read(uint8\_t buttonPort, uint8\_t buttonPin, uint8\_t \*value);

(This function is used to read the state of the button High or LOW.)