# On-demand Traffic light control

## 1. System Description

- Traffic lights are signaling devices positioned at road intersections, pedestrian crossings, and other locations to control the flow of traffic.
- Traffic lights normally consist of three signals, transmitting meaning to drivers and riders through colors and symbols including arrows and bicycles.
- The regular traffic light colors are red, yellow, and green arranged vertically or horizontally in that order.
- Crosswalk buttons let the signal operations know that someone is planning to cross the street, so the light adjusts, giving the pedestrian enough time to get across.

## On-demand Traffic light control

### 1. System Description

- ***<u>Hardware Requirements</u>*** :

  1. ATmega32 microcontroller
  2. One push button for pedestrian
  3. Three LEDs for cars - Green, Yellow, and Red
  4. Three LEDs for pedestrians - Green, Yellow, and Red

# On-demand Traffic light control

## 1. System Description

- ***<u>Software Requirements</u>*** :

  There are two modes of operation :

  **I. Normal mode** :

  1. Cars' LEDs will be changed every five seconds starting from Green then yellow then red then yellow then Green.
  2. The Yellow LED will blink for five seconds before moving to Green or Red LEDs.

  **II. Pedestrian mode** :

  1. Change from normal mode to pedestrian mode when the pedestrian button is pressed.

  2.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.

  3.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.

  4.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.

  5.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.

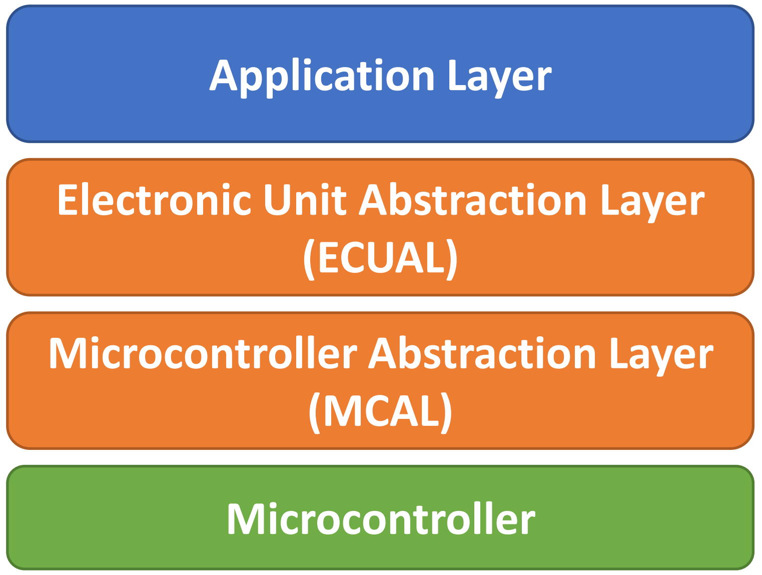
  6.Traffic lights signals are going to the normal mode again.

# On-demand Traffic light control

## 2. System Design

- ***System Layers*** :

1. **Microcontroller** : physical controller of the system.
2. **Microcontroller Abstraction Layer (MCAL)** : microcontroller dependent drivers.
3. **Electronic Unit Abstraction Layer (ECUAL)** : input / output units' drivers.
4. **Application Layer** : software logic that control the system.


Application Layer
Electronic Unit Abstraction Layer (ECUAL)
Microcontroller Abstraction Layer (MCAL)
Microcontroller

## On-demand Traffic light control

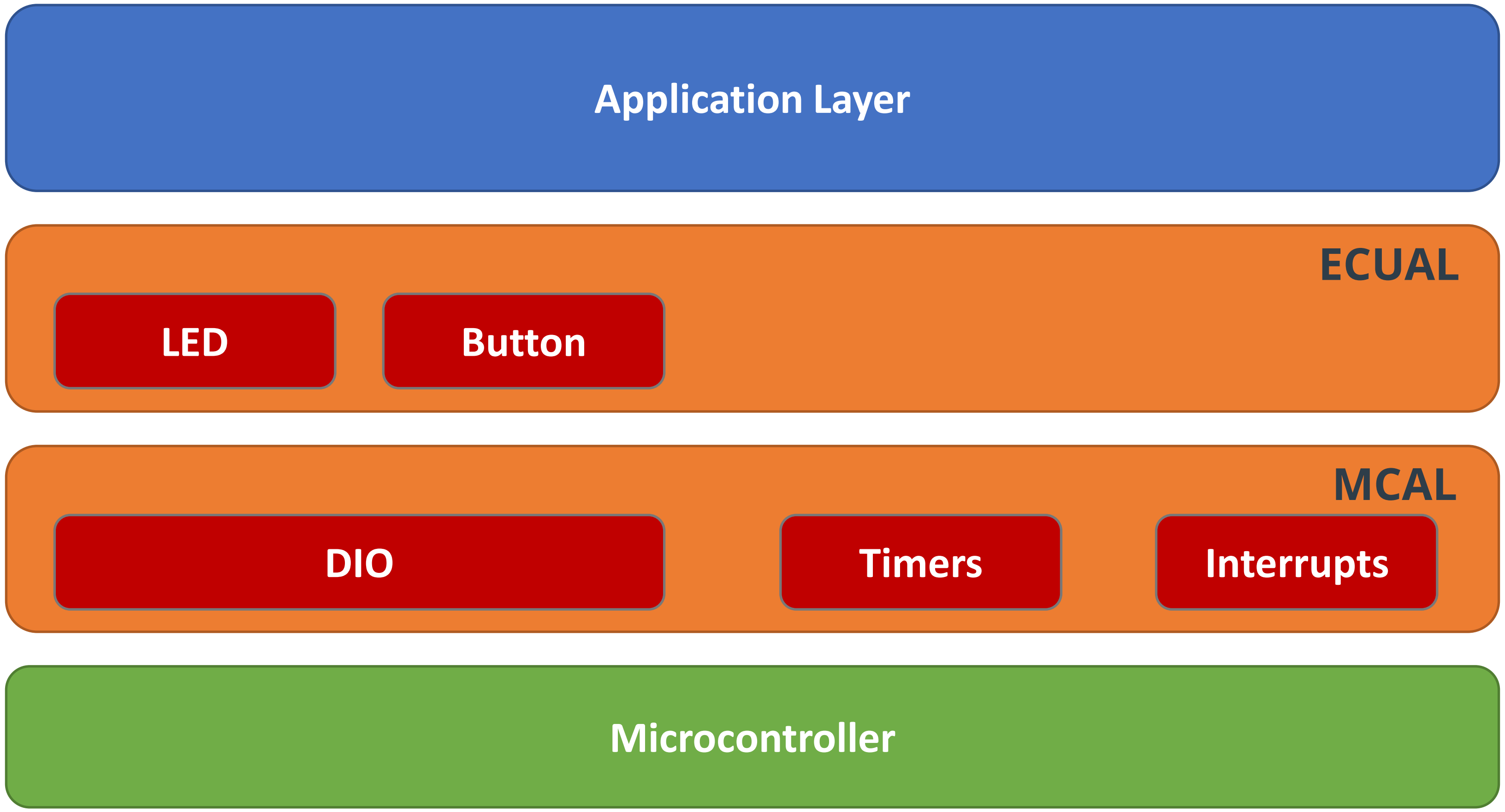
## 2. System Design

- ***<u>System Drivers</u>*** :
  - **Digital Input/Output Driver (DIO Driver)** :
    Initializes any pin in each port to be INPUT or OUTPUT pin.
    Controls any pin in each port to be HIGH , LOW or TOGGLE that pin.
    Get the status of any pin.
  - **Interrupts Driver** :
    Defines the External Interrupt vectors.
    Defines the Interrupt Service Routine (ISR) function prototype.
    Sets/Clears the global interrupt flag.
    Initializes interrupts according to the interrupt action event.
  - **Timers Driver** :
    Initializes the timer.
    Defines delay function.
  - **LED Driver** :
    Initializes LED for a pin within a port.
    Controls the LED ( ON,OFF and TOGGLING ).
  - **Button Driver** :
    Initializes BUTTON for a specific pin within a specific port.
    Get the button status.

# On-demand Traffic light control

## 2. System Design

- ***System Layers*** :



# On-demand Traffic light control

## 2. System Design

- ***System Drivers APIs*** :

DIO

```
EN_dioError_t DIO_init(uint8_t portNumber,uint8_t pinNumber,uint8_t direction); // Initialize direction of pins
EN_dioError_t DIO_write(uint8_t portNumber,uint8_t pinNumber,uint8_t value); // Write data to DIO pin
EN_dioError_t DIO_toggle(uint8_t portNumber,uint8_t pinNumber); // Toggle a DIO Pin
EN_dioError_t DIO_read(uint8_t portNumber,uint8_t pinNumber,uint8_t* value); // Read DIO Data
```

Interrupts

```
EN_interruptError_t INT0_init(uint8_t value); // Initialize INTERRUPT 0
```

Timers

```
EN_timerError_t TIMER_init(EN_Timers_t timer,EN_timerModes_t mode); // Initialize Timer to work in the selected mode
EN_timerError_t delay(EN_Timers_t timer,unsigned int value); // Calculate Timer parameters and enable/start timer to
delay in milliseconds
```

LED

```
EN_ledError_t LED_init(uint8_t ledPort,uint8_t ledPin); // Initialize the LED pin to be output
EN_ledError_t LED_on(uint8_t ledPort,uint8_t ledPin); // Write HIGH to LED pin
EN_ledError_t LED_off(uint8_t ledPort,uint8_t ledPin); // Write Low to LED pin
EN_ledError_t LED_toggle(uint8_t ledPort,uint8_t ledPin); // Toggle LED pin
```

Button

```
EN_buttonError_t BUTTON_init(uint8_t buttonPort,uint8_t buttonPin); // Initialize button pin to be INPUT
EN_buttonError_t BUTTON_read(uint8_t buttonPort,uint8_t buttonPin,uint8_t* value); // Get status of the button
```

## On-demand Traffic light control

## 2. System Design

- ***System Drivers Data Types*** :

DIO

```
typedef enum EN_dioError
{
DIO_OK,WRONG_DIO_PORT_NUMBER,WRONG_DIO_PIN_NUMBER,WRONG_DIO_DIRECTION,WRONG_DIO_PIN_VALUE
}EN_dioError_t;
```

Interrupts

```
typedef enum EN_interruptError
{
INTERRPUT_OK,WRONG_INTERRPUT_TRIGGER
}EN_interruptError_t;
```

Timers

```
typedef enum EN_Timers{TIMER_0,TIMER_1,TIMER_2}EN_Timers_t;
typedef enum EN_timerModes{NORMAL,CTC,PWM_FAST,PWM_PHASE_CORRECT,COUNTER}EN_timerModes_t;
typedef enum EN_timerError{TIMER_OK,WRONG_TIMER_NAME,WRONG_TIMER_MODE,WRONG_DELAY_VALUE}EN_timerError_t;
```

LED

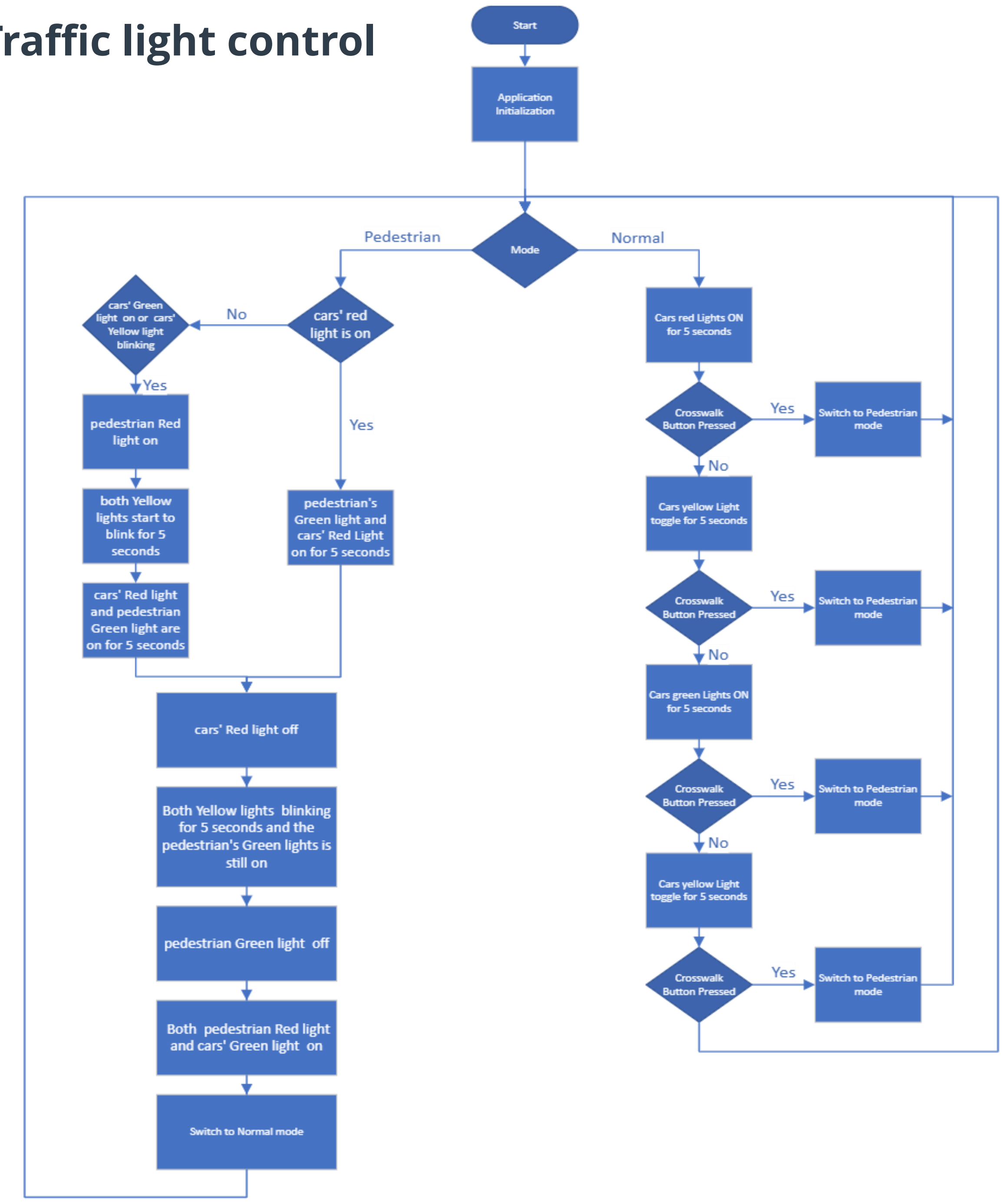
```
typedef enum EN_ledError
{
LED_OK,WRONG_LED_PORT_NUMBER,WRONG_LED_PIN_NUMBER
}EN_ledError_t;
```

Button

```
typedef enum EN_buttonError
{
BUTTON_OK,WRONG_BUTTON_PORT_NUMBER,WRONG_BUTTON_PIN_NUMBER
}EN_buttonError_t;
```

## On-demand Traffic light control

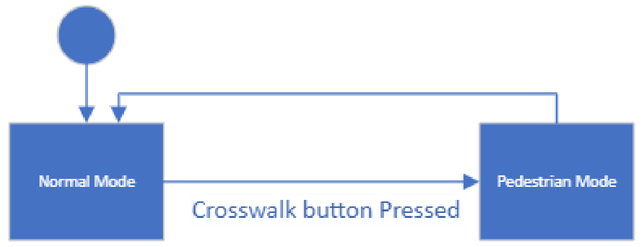
### 3. System Flowchart


Start
Application Initialization
Mode
Pedestrian
Normal
cars' red light is on
No
cars' Green light on or cars' Yellow light blinking
Yes
pedestrian Red light on
both Yellow lights start to blink for 5 seconds
cars' Red light and pedestrian Green light are on for 5 seconds
Yes
pedestrian's Green light and cars' Red Light on for 5 seconds
cars' Red light off
Both Yellow lights blinking for 5 seconds and the pedestrian's Green lights is still on
pedestrian Green light off
Both pedestrian Red light and cars' Green light on
Switch to Normal mode
Cars red Lights ON for 5 seconds
Crosswalk Button Pressed
Yes
Switch to Pedestrian mode
No
Cars yellow Light toggle for 5 seconds
Crosswalk Button Pressed
Yes
Switch to Pedestrian mode
No
Cars green Lights ON for 5 seconds
Crosswalk Button Pressed
Yes
Switch to Pedestrian mode
No
Cars yellow Light toggle for 5 seconds
Crosswalk Button Pressed
Yes
Switch to Pedestrian mode

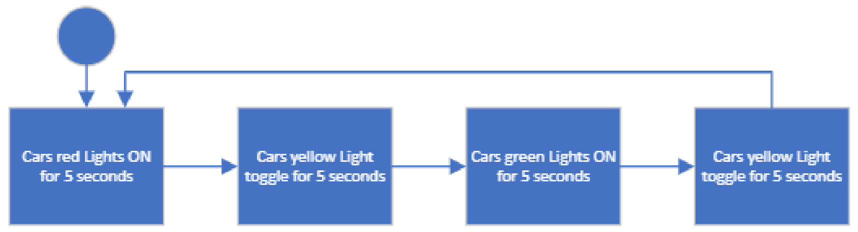
# On-demand Traffic light control

## 4. System State Machine

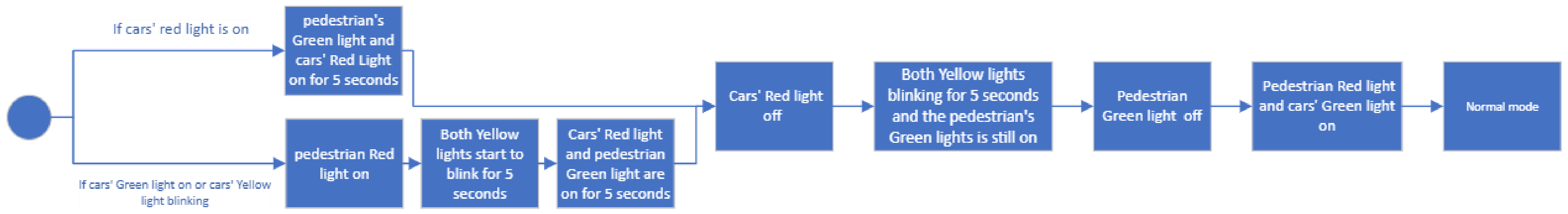
Overall Program State Diagram



Normal Mode State Diagram


Cars red Lights ON for 5 seconds
Cars yellow Light toggle for 5 seconds
Cars green Lights ON for 5 seconds
Cars yellow Light toggle for 5 seconds

Pedestrian Mode State Diagram


If cars' red light is on
pedestrian's Green light and cars' Red Light on for 5 seconds
If cars' Green light on or cars' Yellow light blinking
pedestrian Red light on
Both Yellow lights start to blink for 5 seconds
Cars' Red light and pedestrian Green light are on for 5 seconds
Cars' Red light off
Both Yellow lights blinking for 5 seconds and the pedestrian's Green lights is still on
Pedestrian Green light off
Pedestrian Red light and cars' Green light on
Normal mode