On-Demand Traffic Light Control System

Table of Contents

1- Introduction
2- System description
3- System design
4- System flow chart
5- System constraints

Introduction

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.

Although this is internationally standardized, variations exist on national and local scales as to traffic light sequences and laws.

System description

On-demand traffic light control system contains and on-demand cross walk button 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.

Full system consists of the following...

1- Three leds for car signals (Green, Yellow, Red)
2- Three leds for Pedestrian signals (Green, Yellow, Red)
3- On-demand cross walk button

On-demand traffic light control system has two modes of operation they are normal mode and pedestrian mode, lets discuss them in the following lines…

First normal mode, in this mode car leds changed each five seconds as following order…

1- Green led will be on for five seconds mean car can cross the road
2- Yellow led will start blinking for five seconds
3- Red led will turn on for five seconds
4- Yellow led will start blinking for five second again
5- repeat the above four steps

second is pedestrian mode, switch to this mode when the pedestrian button is pressed, in this mode we have 2 state of operations depends on when the pedestrian pressed the on-demand cross walk button.

1- If the pedestrian pressed the button when the car red led is on
    a. The pedestrian green led and car red led will be on for five seconds
    b. The pedestrian yellow led and car yellow led will start blinking for five seconds while the pedestrian green led still on and car red led turned off
    c. After five second blinking the pedestrian yellow and green led and car yellow led will turned off and the pedestrian red led and car green led will be turned on and back to normal mode operation

2- If the pedestrian pressed the button while the car green led is on or yellow led is blinking
    a. Pedestrian red led will be on and pedestrian yellow led and car yellow led will start blinking for five seconds
    b. Then the pedestrian green led and car red led will be on for five seconds
    c. After that pedestrian yellow led and car yellow led will start blinking for five seconds while pedestrian green led still on
    d. Finally pedestrian red led and car green led will be on and system back to normal mode

The above is a brief description about how on-demand traffic light control system works which each mode and state explained.

System design

In this section will discuss system static design starting with system drivers to the system sequential diagram.

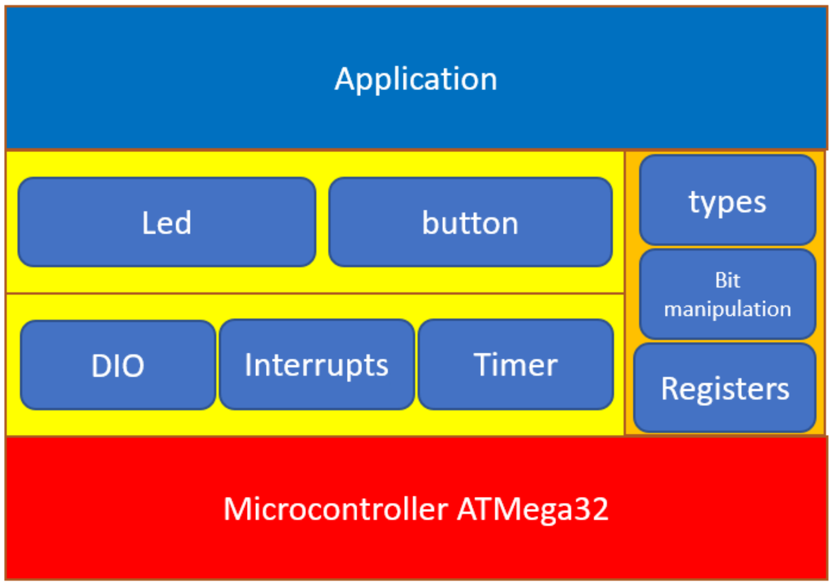
The system layers will be as following…

1- Microcontroller (ATMega32)
2- MCAL
3- ECUAL
4- Application
5- Utilities

The system required drivers will be as following…

1- DIO driver
2- Interrupts
3- Timer driver
4- Led driver
5- Push button driver

Following sketch defines drivers relations with layers.

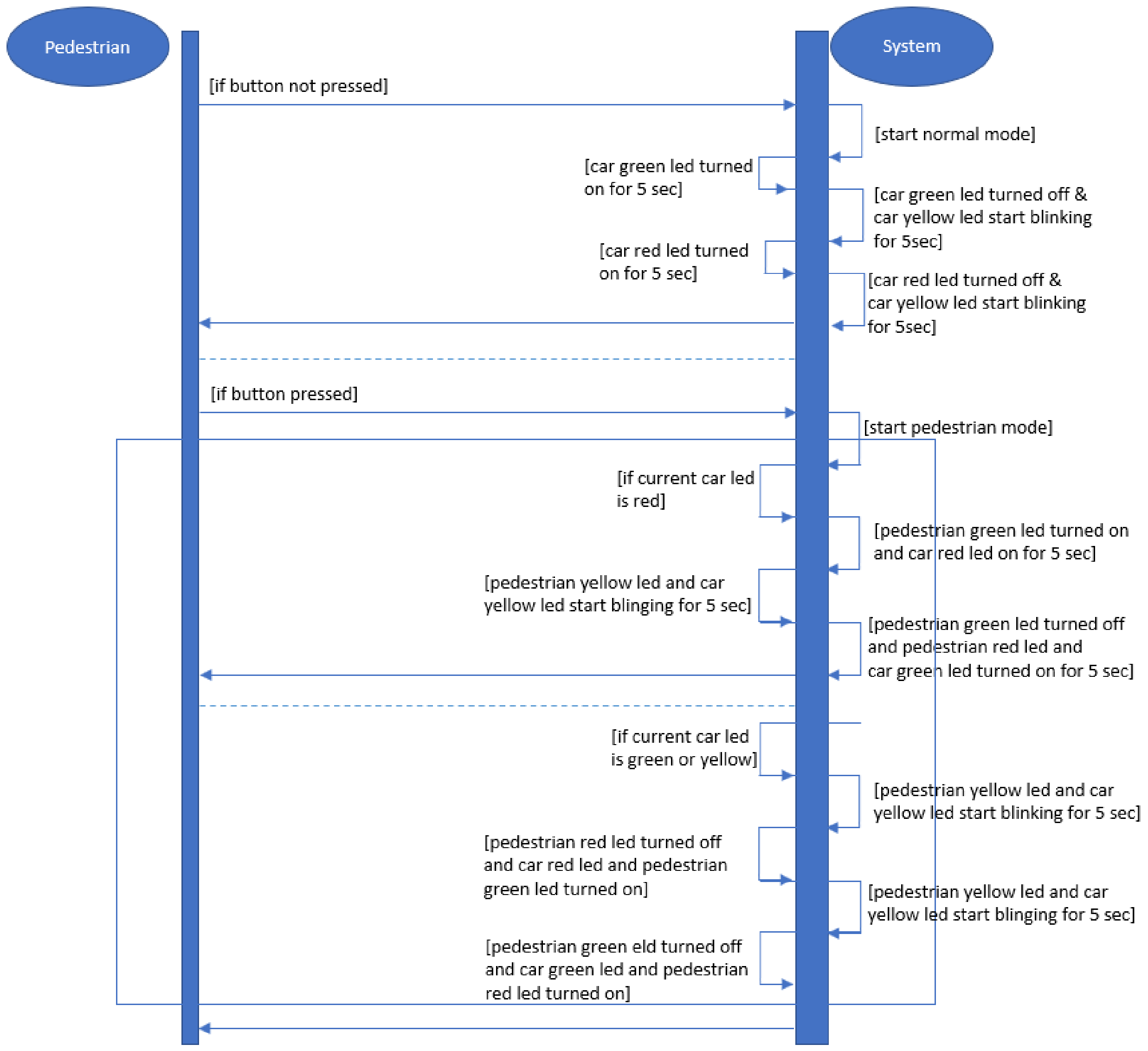


In the following section we will discuss APIs used in each driver…

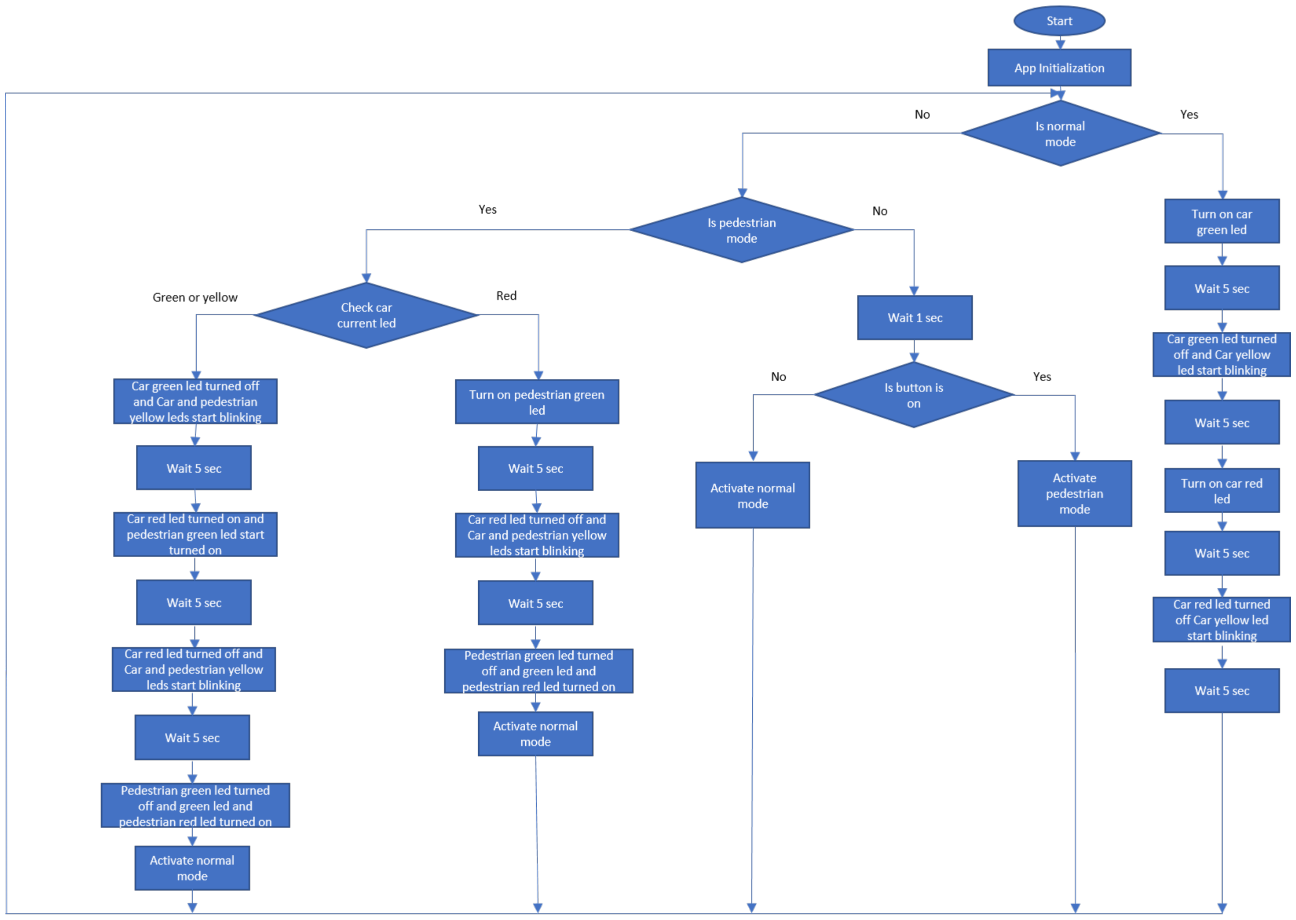
1- DIO driver contains the following APIs
    a. void DIO_init(uint8_t port, uint8_t pin, uint8_t dir);
        i. this API used to initialize MCU pin as input or output it takes the PORT name, pin number and direction (IN or OUT) as inputs.
    b. void DIO_write(uint8_t port, uint8_t pin, uint8_t value);
        i. this API writes logic zero (LOW) or logic one (HIGH) to the specified pin number in a port as per its arguments.
    c. void DIO_toggle(uint8_t port, uint8_t pin);
        i. this API toggle the pin number in port as pass to it from HIGH to LOW or LOW to HIGH depends on the current state on the pin
    d. void DIO_read(uint8_t port, uint8_t pin, volatile uint8_t *value);
        i. this API gets the current value on pin number in specified port and save the it in the value argument as LOW or HIGH (0 or 1)
2- interrupts has the following definitions
    a. #define EXT_INT_0 __vector_1
        i. INT0 external interrupt vector
    b. #define TIMER0_OVF __vector_11
        i. Timer 0 over flow interrupt vector
    c. #define TIMER0_COMP __vector_10
        i. Timer 0 compare match interrupt vector
    d. #define sei() __asm__ __volatile__ ("sei" ::: "memory")
        i. Used to enable global interrupts
    e. #define cli() __asm__ __volatile__ ("cli" ::: "memory")
        i. Used to clear global interrupts
    f. #define ISR(INT_VECTOR) void INT_VECTOR(void) __attribute__ ((signal used));\
    void INT_VECTOR(void)
        i. Used to define the routine which should be executed if the INT_VECTOR activated
3- Timer driver
    a. typedef enum EN_timer0Mode_t {
    NORMAL, CTC, FAST_PWM, PHASE_CORRECT_PWM
    }EN_timer0Mode_t;
        i. this new type define used as input to timer0_start API to define the required mode
    b. void timer0_start(EN_timer0Mode_t mode);
        i. this API configure timer 0 registers to start the passed mode as it's input
    c. uint32_t timer0_millis(void);
        i. this API returns the number of milli seconds passed from the MCU power on the minimum resolution is 2 milli seconds used compare match mode and enabled interrupts to get accurate timing

4- led driver
    a. typedef struct ST_led_t{
       uint8_t port;
       uint8_t pin;
       }ST_led_t;
        i. new defined type structure contains port and pin used for the led
    b. void LED_init(ST_led_t led);
        i. used to initialize passed led as argument as output pin
    c. void LED_on(ST_led_t led);
        i. used to turn the led on
    d. void LED_off(ST_led_t led);
        i. used to turn the led off
    e. void LED_toggle(ST_led_t led);
        i. used to toggle the led state
5- button driver
    a. typedef struct ST_button_t{
       uint8_t port;
       uint8_t pin;
       volatile uint8_t value;
       }ST_button_t;
        i. new defined type structure contains port, pin number and value to store the current button state
    b. void button_init(ST_button_t button);
        i. used to initialize the button defined pin as input DIO
    c. void button_read(ST_button_t *button);
        i. used to get the current state of the button if pressed or not

On-demand traffic light control system sequential diagram


Pedestrian
System
[if button not pressed]
[start normal mode]
[car green led turned on for 5 sec]
[car green led turned off & car yellow led start blinking for 5sec]
[car red led turned on for 5 sec]
[car red led turned off & car yellow led start blinking for 5sec]
[if button pressed]
[start pedestrian mode]
[if current car led is red]
[pedestrian green led turned on and car red led on for 5 sec]
[pedestrian yellow led and car yellow led start blinging for 5 sec]
[pedestrian green led turned off and pedestrian red led and car green led turned on for 5 sec]
[if current car led is green or yellow]
[pedestrian yellow led and car yellow led start blinking for 5 sec]
[pedestrian red led turned off and car red led and pedestrian green led turned on]
[pedestrian yellow led and car yellow led start blinging for 5 sec]
[pedestrian green eld turned off and car green led and pedestrian red led turned on]

System Flow Chart


Start
App Initialization
Is normal mode
No
Yes
Turn on car green led
Wait 5 sec
Car green led turned off and Car yellow led start blinking
Wait 5 sec
Turn on car red led
Wait 5 sec
Car red led turned off Car yellow led start blinking
Wait 5 sec
Is pedestrian mode
Yes
No
Wait 1 sec
Is button is on
No
Yes
Activate normal mode
Activate pedestrian mode
Check car current led
Green or yellow
Red
Car green led turned off and Car and pedestrian yellow leds start blinking
Wait 5 sec
Car red led turned on and pedestrian green led start turned on
Wait 5 sec
Car red led turned off and Car and pedestrian yellow leds start blinking
Wait 5 sec
Pedestrian green led turned off and green led and pedestrian red led turned on
Activate normal mode
Turn on pedestrian green led
Wait 5 sec
Car red led turned off and Car and pedestrian yellow leds start blinking
Wait 5 sec
Pedestrian green led turned off and green led and pedestrian red led turned on
Activate normal mode

System Constrains

1- long press on the button should has not effect and this handled by condition to check the state of button after the interrupt executed by one second if it still HIGH then don’t activate the pedestrian mode if back to zero then activate pedestrian mode
2- multiple short presses should it will only execute the first press because if the pedestrian mode is activated system will ignore any press until normal mode activated again