Software Design

1) System Description

In this project we need to make a traffic light control for both cars and pedestrians to be able to manage the road flow for both.

To deliver this project, we should have:

- Atmega32 MCU
- One Push Button
- 6 LEDs (2 Red – 2 Green – 2 Yellow)

The project goes as follows → we have two traffic lights, one for cars and another for pedestrians.

- ➢ When the cars' traffic lights became red, it means that they should stop.
- ➢ When the cars' traffic lights became yellow, it means that they will wait for a few seconds and then move.
- ➢ When the cars' traffic lights became green, it means that they will move immediately.

And for the pedestrians:-

- ❖ When the pedestrians' traffic lights became red, it means that the pedestrians should stop.
- ❖ When the pedestrians' traffic lights became yellow, it means that the pedestrians will wait for a few seconds and then move.
- ❖ When the pedestrians' traffic lights became green, it means that the pedestrians will move immediately.

In addition, if the pedestrian is on hurry, he/she will press a push button to be able to stop the car and cross the road.

2) System Design

In this section, we will make a full static architecture for the system.

- System Layers

  They will be divided into 4 layers:

  - ✓ Application → Contains the main file.
  - ✓ ECUAL → Contains the electronics we need in the project.
  - ✓ MCAL → Contains the internal peripherals of the MCU.
  - ✓ Microcontroller → Contains the hardware (MCU).

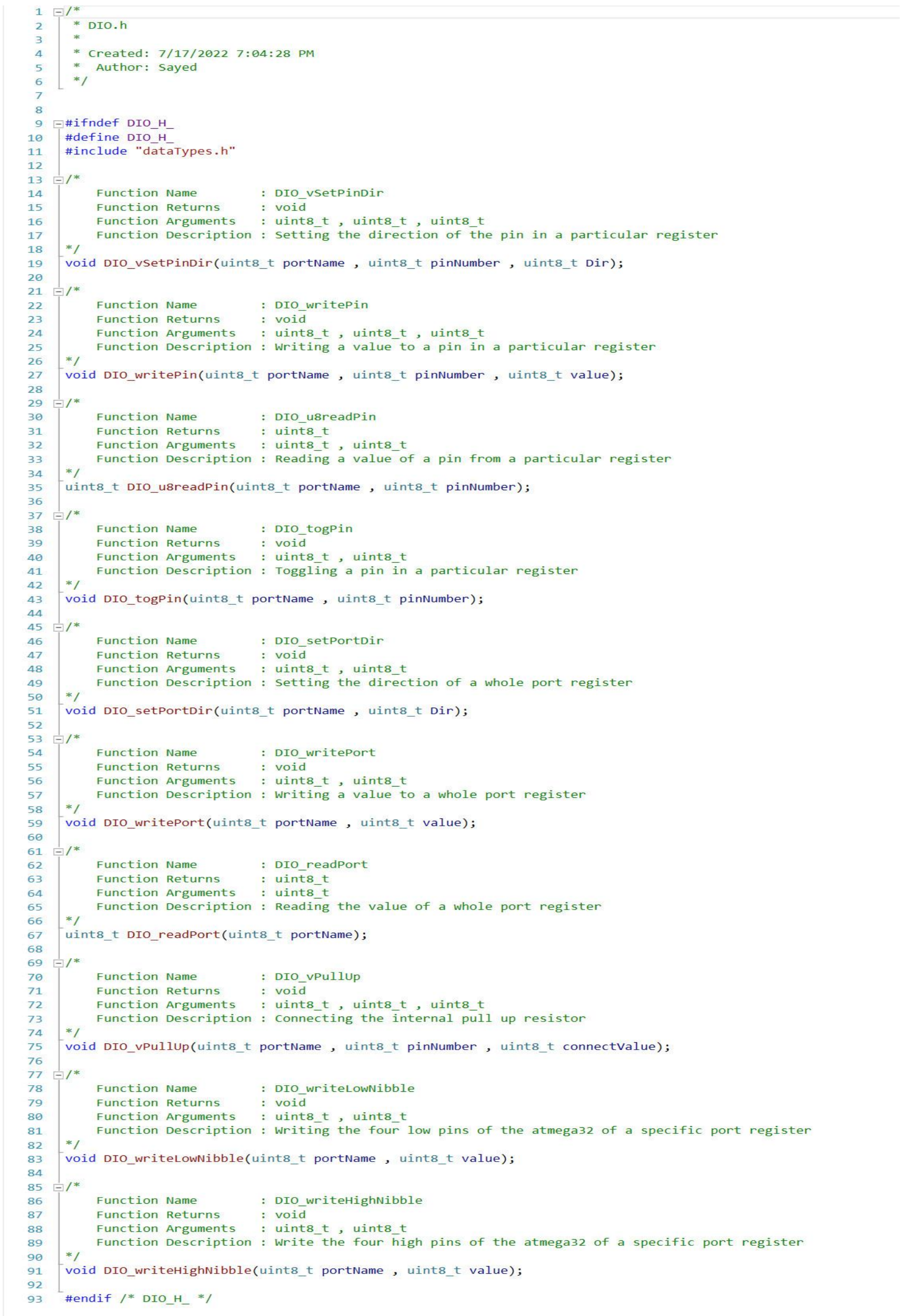
Application
ECUAL
MCAL
Microcontroller

- System Drivers

  Every module we use in the MCAL layer will have a driver which splits to file.c and file.h .

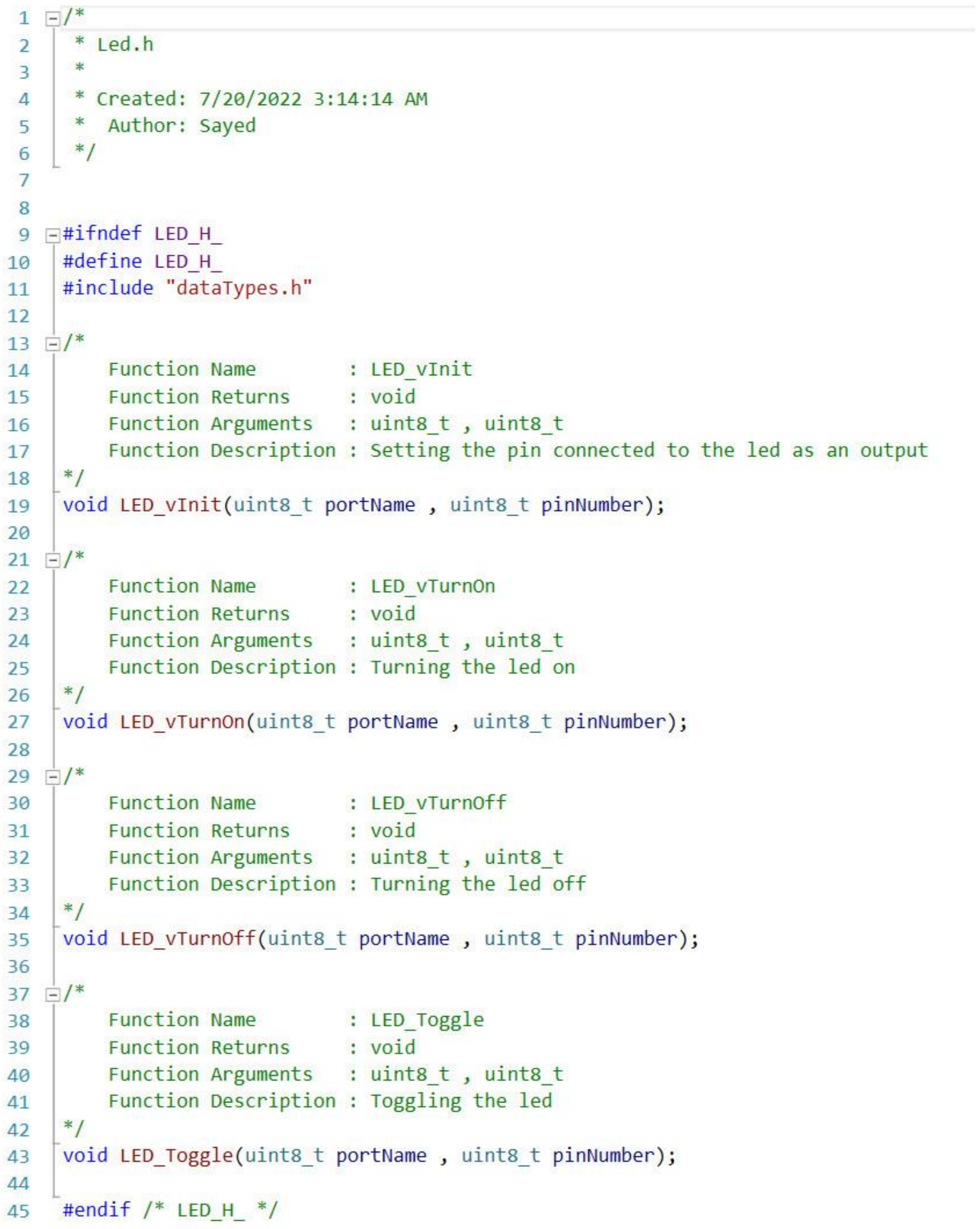
Application
LED – Button
DIO – Timer
Microcontroller

✓ DIO APIs



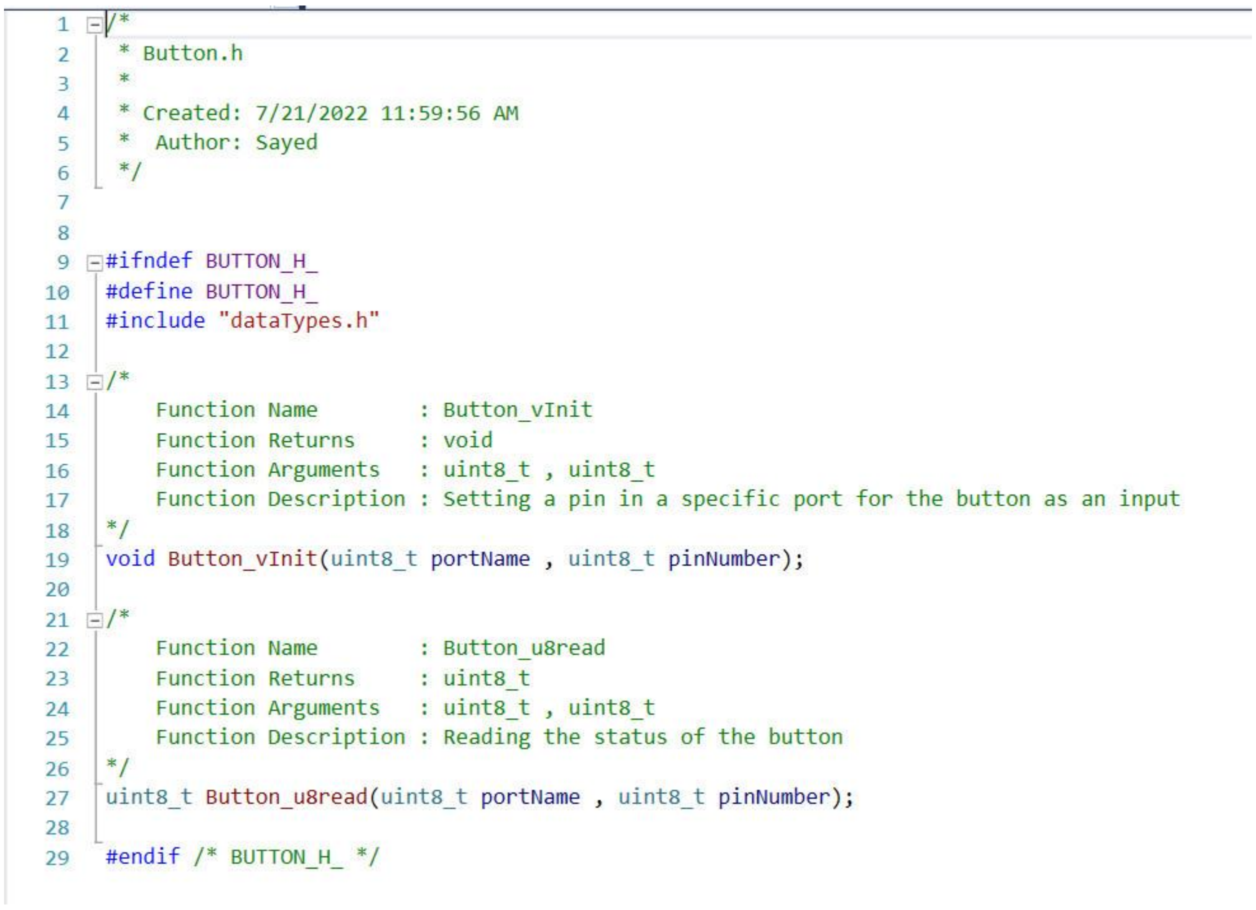
```c
/*
 * DIO.h
 *
 * Created: 7/17/2022 7:04:28 PM
 *  Author: Sayed
 */


#ifndef DIO_H_
#define DIO_H_
#include "dataTypes.h"

/*
    Function Name        : DIO_vSetPinDir
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t , uint8_t
    Function Description : Setting the direction of the pin in a particular register
*/
void DIO_vSetPinDir(uint8_t portName , uint8_t pinNumber , uint8_t Dir);

/*
    Function Name        : DIO_writePin
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t , uint8_t
    Function Description : Writing a value to a pin in a particular register
*/
void DIO_writePin(uint8_t portName , uint8_t pinNumber , uint8_t value);

/*
    Function Name        : DIO_u8readPin
    Function Returns     : uint8_t
    Function Arguments   : uint8_t , uint8_t
    Function Description : Reading a value of a pin from a particular register
*/
uint8_t DIO_u8readPin(uint8_t portName , uint8_t pinNumber);

/*
    Function Name        : DIO_togPin
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Toggling a pin in a particular register
*/
void DIO_togPin(uint8_t portName , uint8_t pinNumber);

/*
    Function Name        : DIO_setPortDir
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Setting the direction of a whole port register
*/
void DIO_setPortDir(uint8_t portName , uint8_t Dir);

/*
    Function Name        : DIO_writePort
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Writing a value to a whole port register
*/
void DIO_writePort(uint8_t portName , uint8_t value);

/*
    Function Name        : DIO_readPort
    Function Returns     : uint8_t
    Function Arguments   : uint8_t
    Function Description : Reading the value of a whole port register
*/
uint8_t DIO_readPort(uint8_t portName);

/*
    Function Name        : DIO_vPullUp
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t , uint8_t
    Function Description : Connecting the internal pull up resistor
*/
void DIO_vPullUp(uint8_t portName , uint8_t pinNumber , uint8_t connectValue);

/*
    Function Name        : DIO_writeLowNibble
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Writing the four low pins of the atmega32 of a specific port register
*/
void DIO_writeLowNibble(uint8_t portName , uint8_t value);

/*
    Function Name        : DIO_writeHighNibble
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Write the four high pins of the atmega32 of a specific port register
*/
void DIO_writeHighNibble(uint8_t portName , uint8_t value);

#endif /* DIO_H_ */
```

- ✓ LED APIs



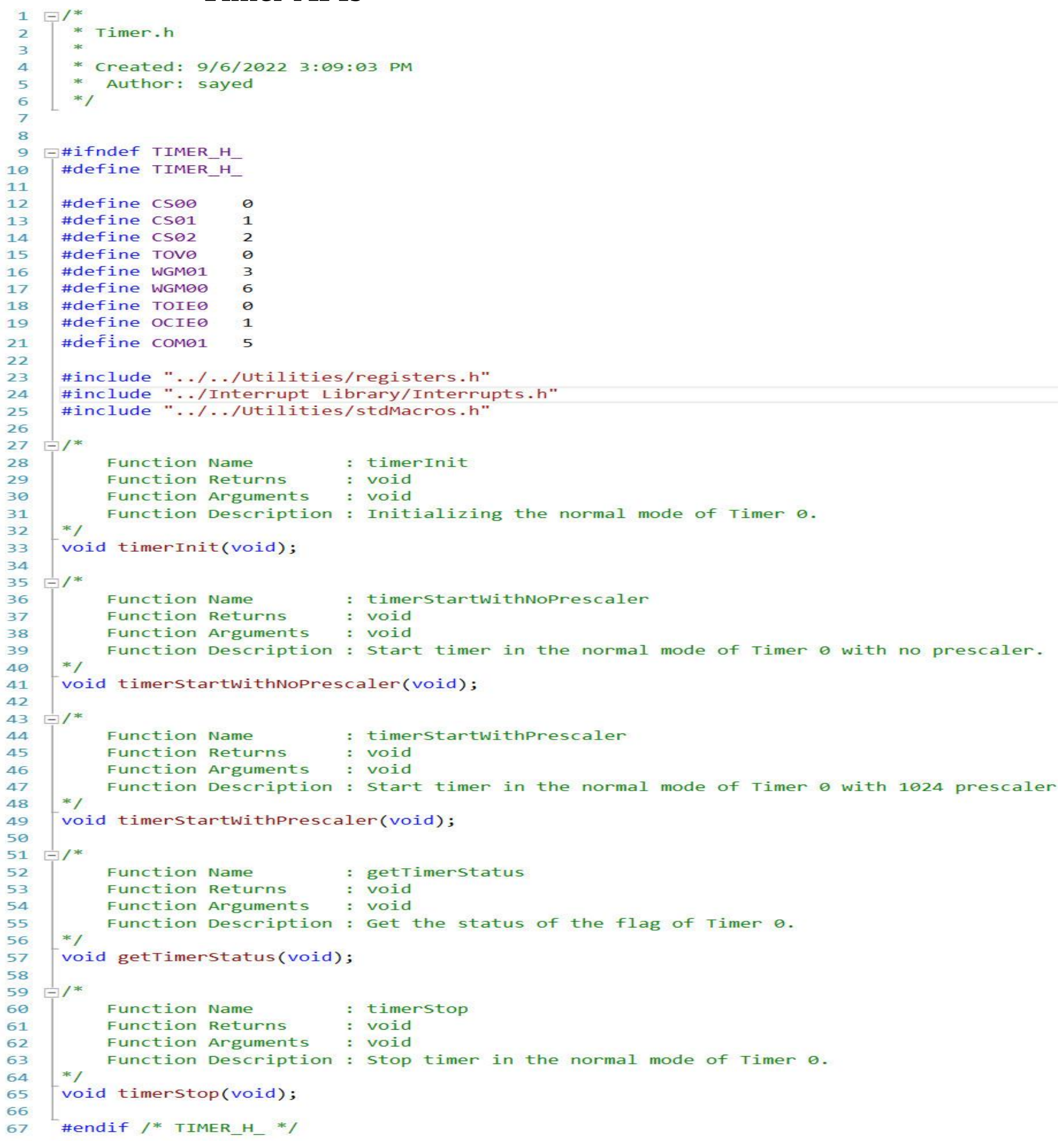
```c
/*
 * Led.h
 *
 * Created: 7/20/2022 3:14:14 AM
 *  Author: Sayed
 */


#ifndef LED_H_
#define LED_H_
#include "dataTypes.h"

/*
    Function Name        : LED_vInit
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Setting the pin connected to the led as an output
*/
void LED_vInit(uint8_t portName , uint8_t pinNumber);

/*
    Function Name        : LED_vTurnOn
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Turning the led on
*/
void LED_vTurnOn(uint8_t portName , uint8_t pinNumber);

/*
    Function Name        : LED_vTurnOff
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Turning the led off
*/
void LED_vTurnOff(uint8_t portName , uint8_t pinNumber);

/*
    Function Name        : LED_Toggle
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Toggling the led
*/
void LED_Toggle(uint8_t portName , uint8_t pinNumber);

#endif /* LED_H_ */
```

✓ Button APIs



```
/*
 * Button.h
 *
 * Created: 7/21/2022 11:59:56 AM
 *  Author: Sayed
 */


#ifndef BUTTON_H_
#define BUTTON_H_
#include "dataTypes.h"

/*
    Function Name        : Button_vInit
    Function Returns     : void
    Function Arguments   : uint8_t , uint8_t
    Function Description : Setting a pin in a specific port for the button as an input
*/
void Button_vInit(uint8_t portName , uint8_t pinNumber);

/*
    Function Name        : Button_u8read
    Function Returns     : uint8_t
    Function Arguments   : uint8_t , uint8_t
    Function Description : Reading the status of the button
*/
uint8_t Button_u8read(uint8_t portName , uint8_t pinNumber);

#endif /* BUTTON_H_ */
```

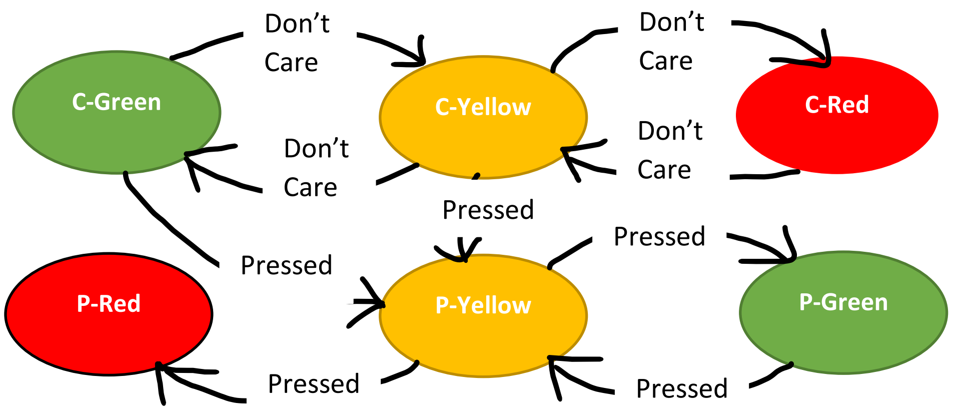
✓ Timer APIs



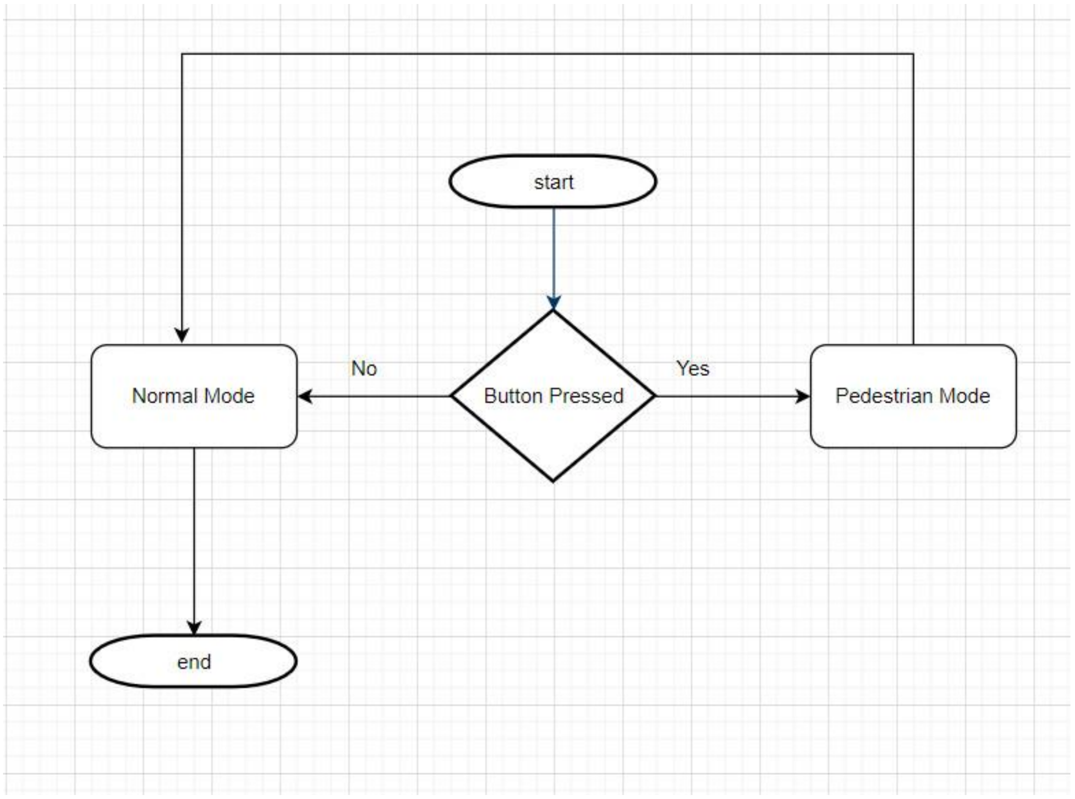
```
/*
 * Timer.h
 *
 * Created: 9/6/2022 3:09:03 PM
 *  Author: sayed
 */


#ifndef TIMER_H_
#define TIMER_H_

#define CS00     0
#define CS01     1
#define CS02     2
#define TOV0     0
#define WGM01    3
#define WGM00    6
#define TOIE0    0
#define OCIE0    1
#define COM01    5

#include "../../Utilities/registers.h"
#include "../Interrupt Library/Interrupts.h"
#include "../../Utilities/stdMacros.h"

/*
    Function Name        : timerInit
    Function Returns     : void
    Function Arguments   : void
    Function Description : Initializing the normal mode of Timer 0.
*/
void timerInit(void);

/*
    Function Name        : timerStartWithNoPrescaler
    Function Returns     : void
    Function Arguments   : void
    Function Description : Start timer in the normal mode of Timer 0 with no prescaler.
*/
void timerStartWithNoPrescaler(void);

/*
    Function Name        : timerStartWithPrescaler
    Function Returns     : void
    Function Arguments   : void
    Function Description : Start timer in the normal mode of Timer 0 with 1024 prescaler
*/
void timerStartWithPrescaler(void);

/*
    Function Name        : getTimerStatus
    Function Returns     : void
    Function Arguments   : void
    Function Description : Get the status of the flag of Timer 0.
*/
void getTimerStatus(void);

/*
    Function Name        : timerStop
    Function Returns     : void
    Function Arguments   : void
    Function Description : Stop timer in the normal mode of Timer 0.
*/
void timerStop(void);

#endif /* TIMER_H_ */
```

3) <u>System State Machine</u>

C → Car || P → Pedestrian || Don't care means if button pressed or not



Simple Flowchart



4) System Constrains

- Long or double button press has no effect for the system.
- Delays are prohibited to be used and we should use timer instead.