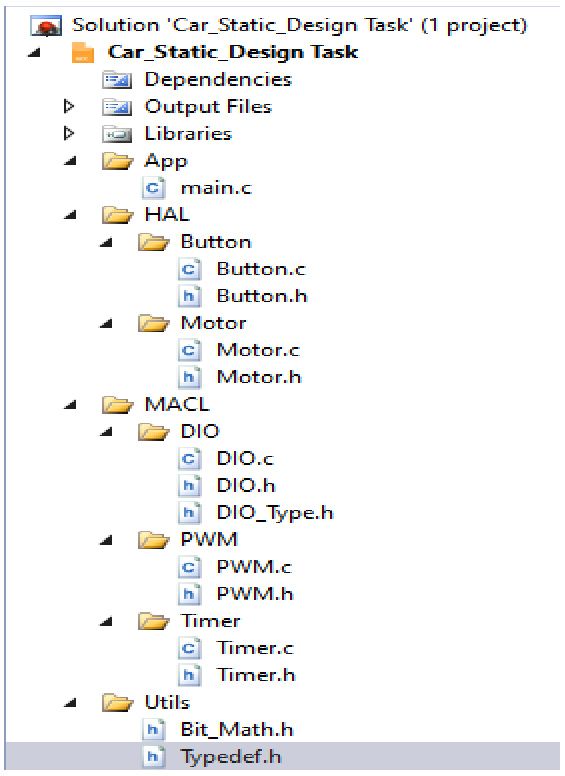
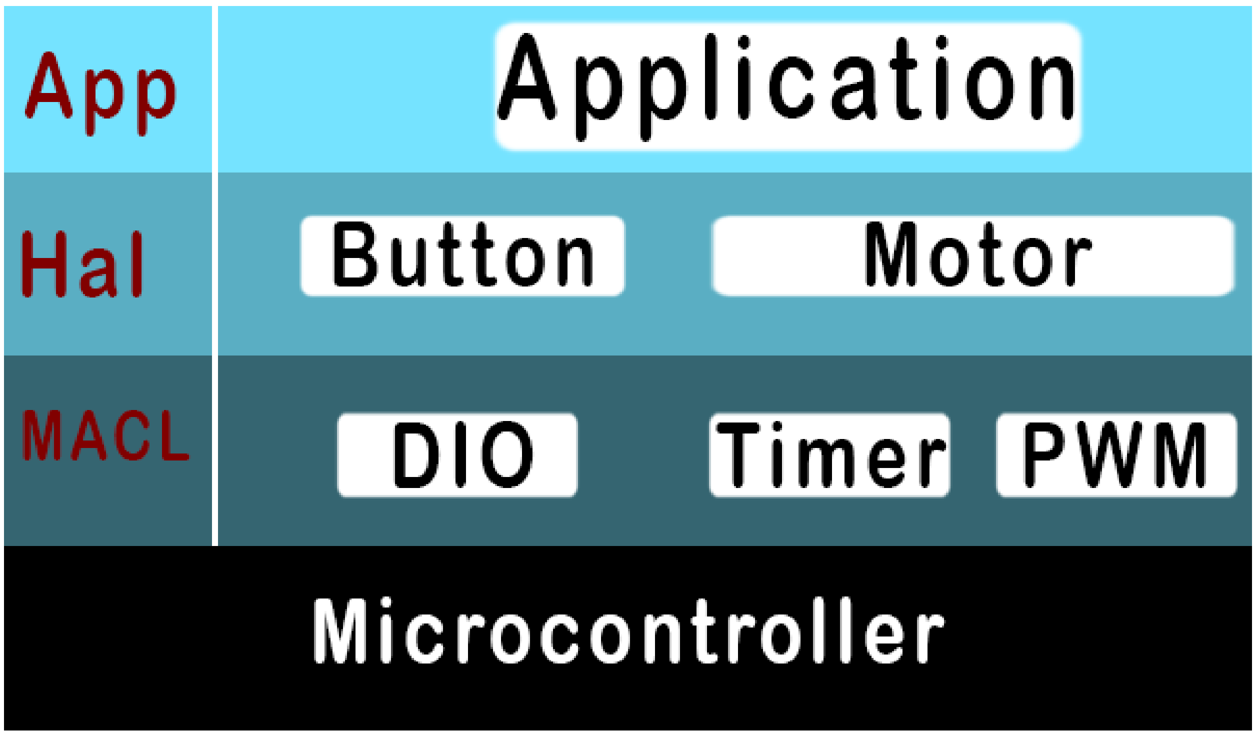
- **Project Files**



```
Solution 'Car_Static_Design Task' (1 project)
◢ Car_Static_Design Task
    Dependencies
  ▷ Output Files
  ▷ Libraries
  ◢ App
      main.c
  ◢ HAL
    ◢ Button
        Button.c
        Button.h
    ◢ Motor
        Motor.c
        Motor.h
  ◢ MACL
    ◢ DIO
        DIO.c
        DIO.h
        DIO_Type.h
    ◢ PWM
        PWM.c
        PWM.h
    ◢ Timer
        Timer.c
        Timer.h
  ◢ Utils
      Bit_Math.h
      Typedef.h
```

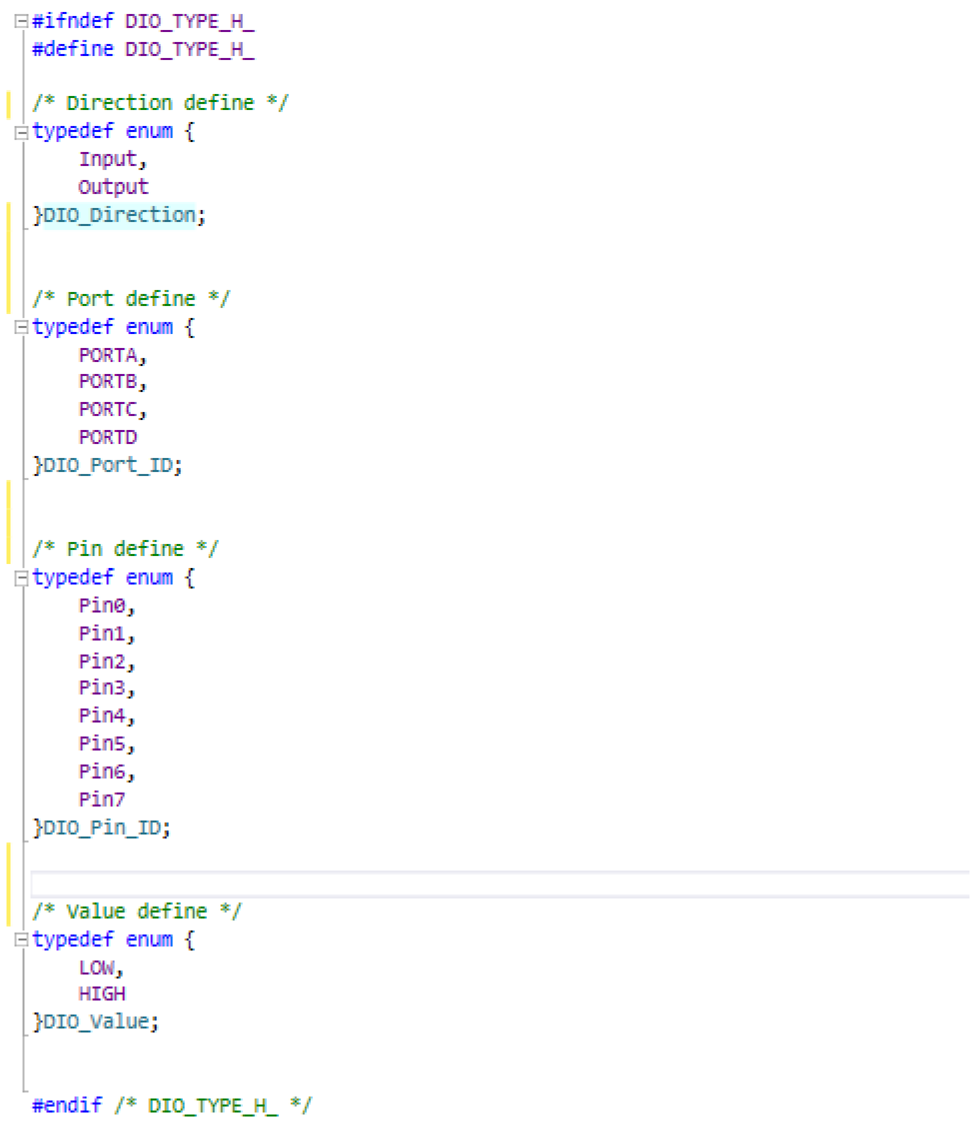
- **Layered Architecture**



## 1. MACL Layer

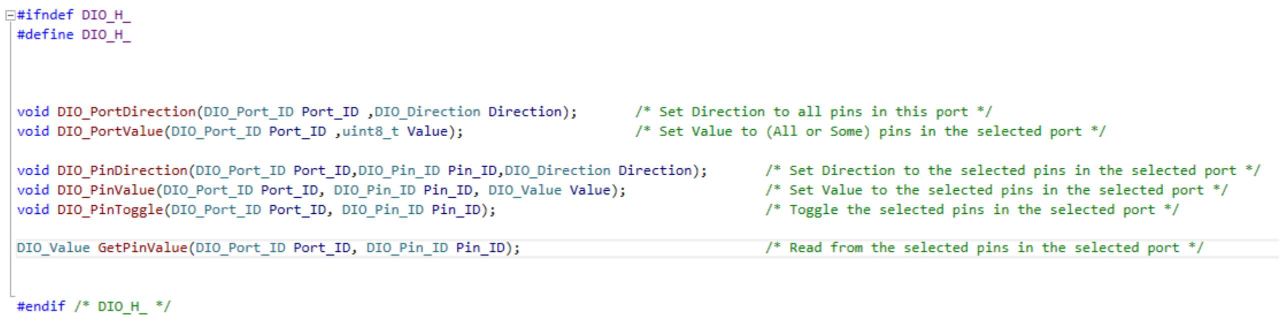
### ✓ DIO

### ❖ DIO_type.h



```c
#ifndef DIO_TYPE_H_
#define DIO_TYPE_H_

/* Direction define */
typedef enum {
    Input,
    Output
}DIO_Direction;


/* Port define */
typedef enum {
    PORTA,
    PORTB,
    PORTC,
    PORTD
}DIO_Port_ID;


/* Pin define */
typedef enum {
    Pin0,
    Pin1,
    Pin2,
    Pin3,
    Pin4,
    Pin5,
    Pin6,
    Pin7
}DIO_Pin_ID;


/* Value define */
typedef enum {
    LOW,
    HIGH
}DIO_Value;


#endif /* DIO_TYPE_H_ */
```

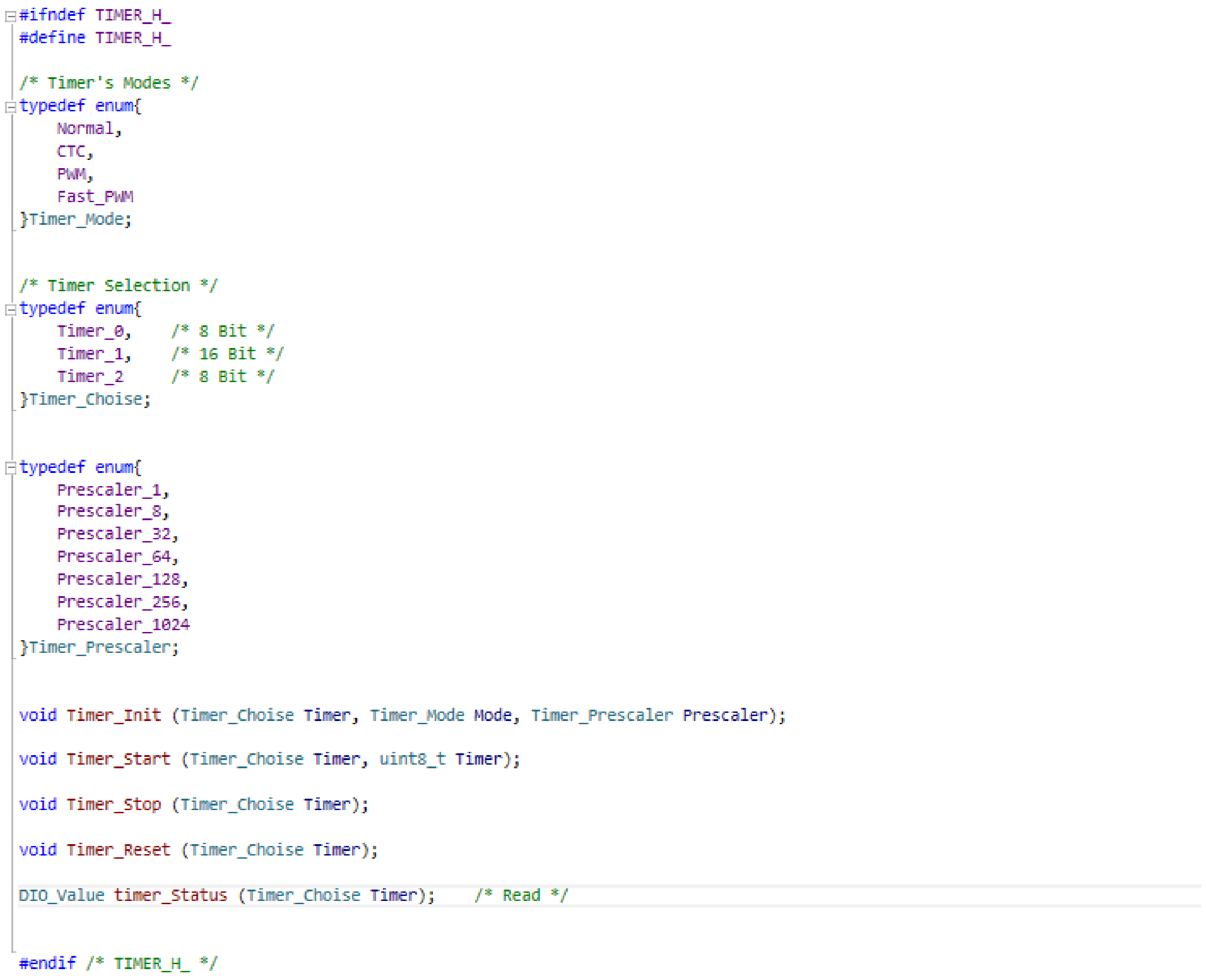
## ❖DIO.h



```
#ifndef DIO_H_
#define DIO_H_



void DIO_PortDirection(DIO_Port_ID Port_ID ,DIO_Direction Direction);       /* Set Direction to all pins in this port */
void DIO_PortValue(DIO_Port_ID Port_ID ,uint8_t Value);                     /* Set Value to (All or Some) pins in the selected port */

void DIO_PinDirection(DIO_Port_ID Port_ID,DIO_Pin_ID Pin_ID,DIO_Direction Direction);       /* Set Direction to the selected pins in the selected port */
void DIO_PinValue(DIO_Port_ID Port_ID, DIO_Pin_ID Pin_ID, DIO_Value Value);                 /* Set Value to the selected pins in the selected port */
void DIO_PinToggle(DIO_Port_ID Port_ID, DIO_Pin_ID Pin_ID);                                 /* Toggle the selected pins in the selected port */

DIO_Value GetPinValue(DIO_Port_ID Port_ID, DIO_Pin_ID Pin_ID);                              /* Read from the selected pins in the selected port */


#endif /* DIO_H_ */
```

## ✓Timer

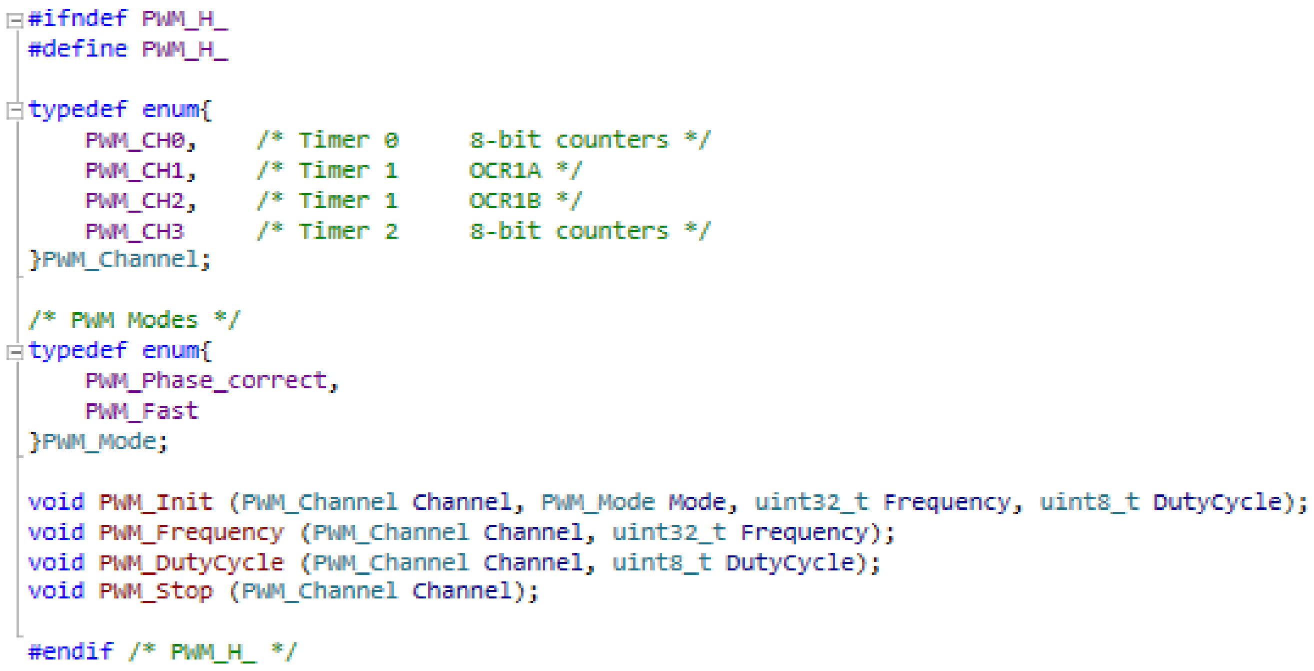
## ❖Timer.h



```
#ifndef TIMER_H_
#define TIMER_H_

/* Timer's Modes */
typedef enum{
    Normal,
    CTC,
    PWM,
    Fast_PWM
}Timer_Mode;


/* Timer Selection */
typedef enum{
    Timer_0,    /* 8 Bit */
    Timer_1,    /* 16 Bit */
    Timer_2     /* 8 Bit */
}Timer_Choise;


typedef enum{
    Prescaler_1,
    Prescaler_8,
    Prescaler_32,
    Prescaler_64,
    Prescaler_128,
    Prescaler_256,
    Prescaler_1024
}Timer_Prescaler;


void Timer_Init (Timer_Choise Timer, Timer_Mode Mode, Timer_Prescaler Prescaler);

void Timer_Start (Timer_Choise Timer, uint8_t Timer);

void Timer_Stop (Timer_Choise Timer);

void Timer_Reset (Timer_Choise Timer);

DIO_Value timer_Status (Timer_Choise Timer);    /* Read */


#endif /* TIMER_H_ */
```

## ✓ PWM

## ❖ PWM.h

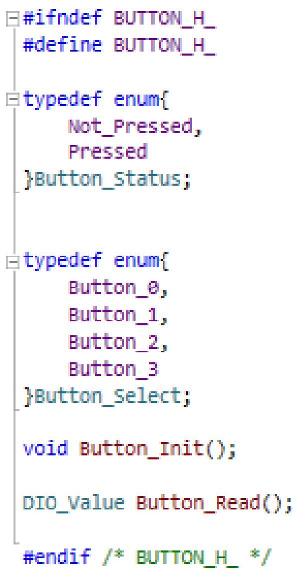


```c
#ifndef PWM_H_
#define PWM_H_

typedef enum{
    PWM_CH0,    /* Timer 0     8-bit counters */
    PWM_CH1,    /* Timer 1     OCR1A */
    PWM_CH2,    /* Timer 1     OCR1B */
    PWM_CH3     /* Timer 2     8-bit counters */
}PWM_Channel;

/* PWM Modes */
typedef enum{
    PWM_Phase_correct,
    PWM_Fast
}PWM_Mode;

void PWM_Init (PWM_Channel Channel, PWM_Mode Mode, uint32_t Frequency, uint8_t DutyCycle);
void PWM_Frequency (PWM_Channel Channel, uint32_t Frequency);
void PWM_DutyCycle (PWM_Channel Channel, uint8_t DutyCycle);
void PWM_Stop (PWM_Channel Channel);

#endif /* PWM_H_ */
```

## 2. HAL Layer

## ✓ Button

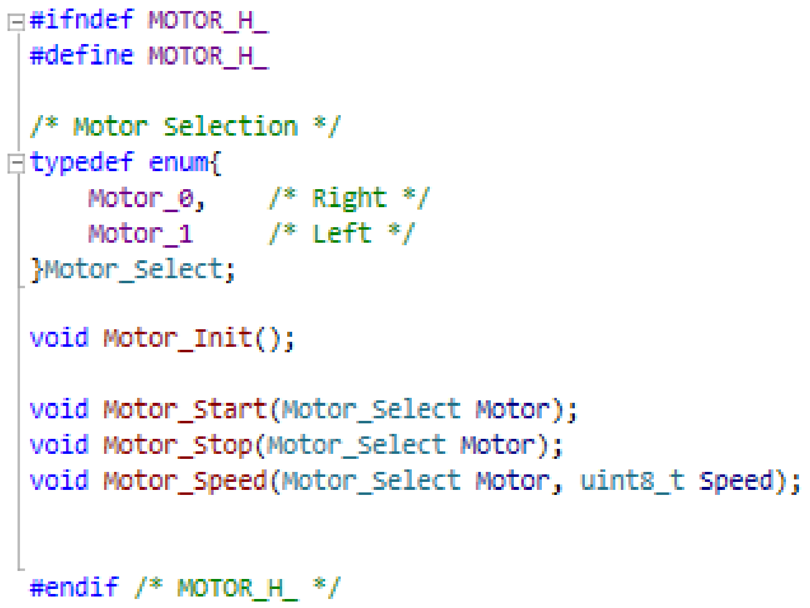
## ❖ Button.h



```c
#ifndef BUTTON_H_
#define BUTTON_H_

typedef enum{
    Not_Pressed,
    Pressed
}Button_Status;


typedef enum{
    Button_0,
    Button_1,
    Button_2,
    Button_3
}Button_Select;

void Button_Init();

DIO_Value Button_Read();

#endif /* BUTTON_H_ */
```

✓ **Motor**

❖ **Motor.h**

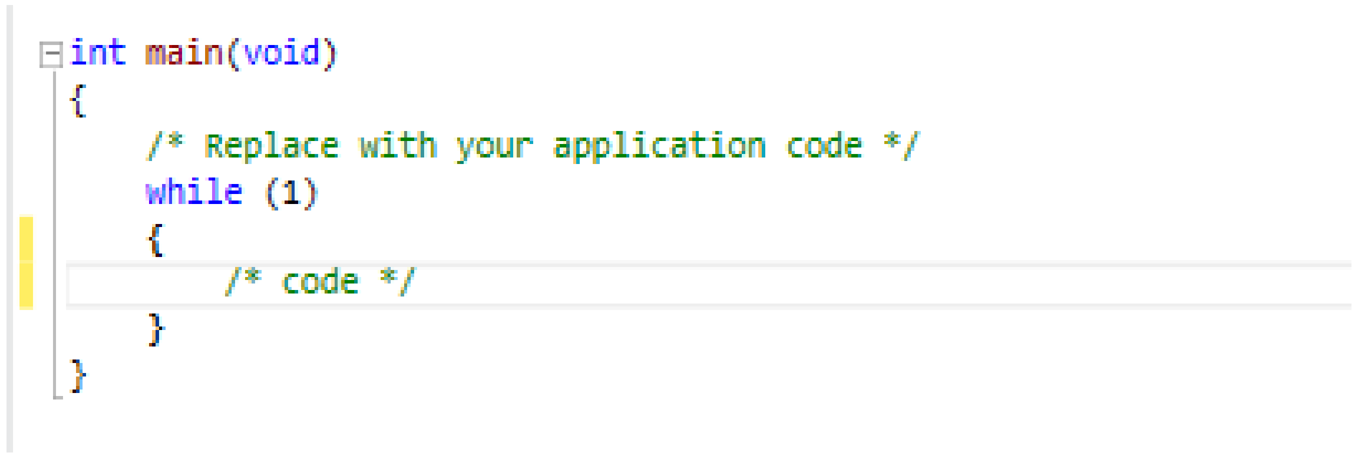


```c
#ifndef MOTOR_H_
#define MOTOR_H_

/* Motor Selection */
typedef enum{
    Motor_0,     /* Right */
    Motor_1      /* Left */
}Motor_Select;

void Motor_Init();

void Motor_Start(Motor_Select Motor);
void Motor_Stop(Motor_Select Motor);
void Motor_Speed(Motor_Select Motor, uint8_t Speed);


#endif /* MOTOR_H_ */
```

**3. Application layer**

✓ **Main**

❖ **Main.c**



```c
int main(void)
{
    /* Replace with your application code */
    while (1)
    {
        /* code */
    }
}
```