Obstacle Avoidance Car

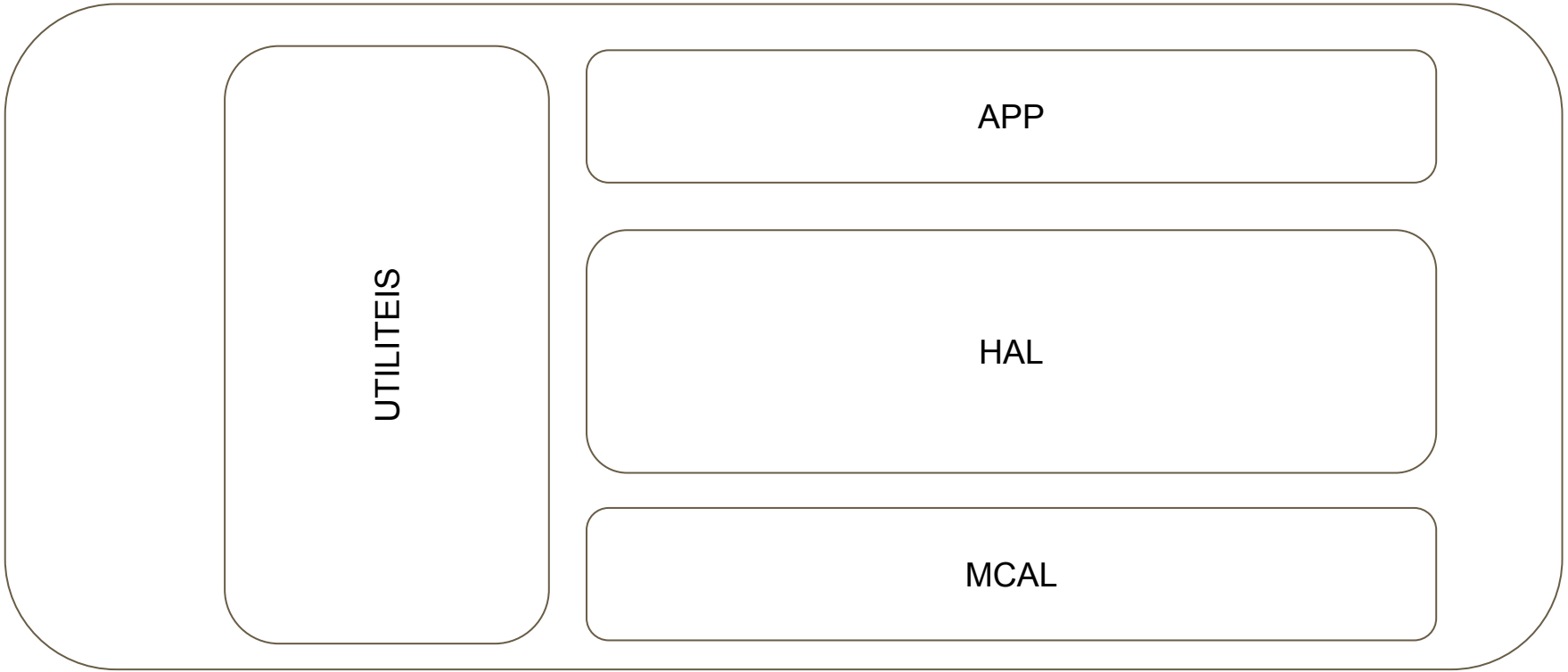
Prepared by:Mohab Ahmed

Table of Contents

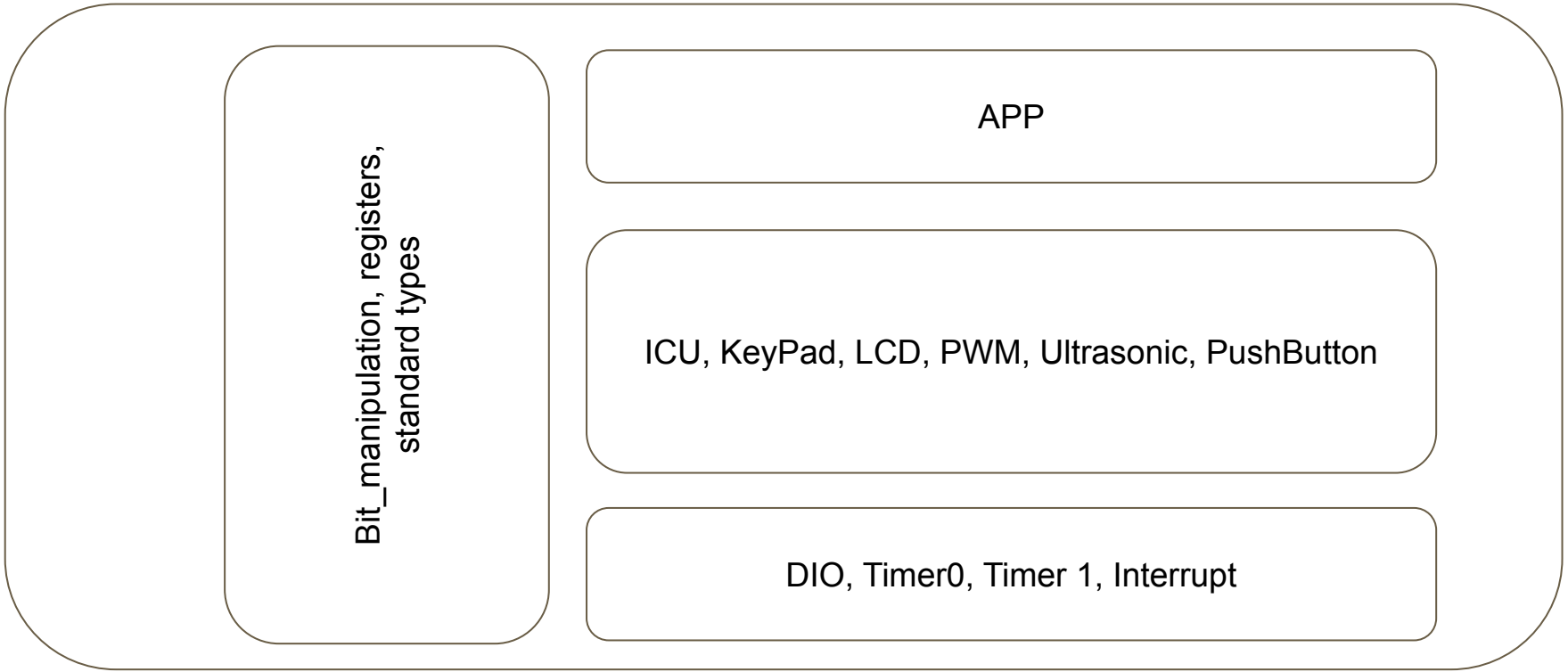
Project Description

The car starts from zero speed. The default rotation direction is to the right. Press PB2 to start or stop the robot. After pressing start, the LCD will display default rotation direction and wait for 5 seconds to choose the rotation direction by Pressing PB1. The robot will move after 2 seconds from setting the default direction. If there is no obstacle for more than 70 cm, the robot will move forward. If there is an obstacle the robot will rotate to the chosen direction of rotation.

Layered architecture



Modules/drivers



Drivers documentation

- DIO: It is Digital Input/Output module and it is responsible for dealing with the input and output digital pins.
- Timer_0: It is the timer module which deals with timers in the microcontroller and enables us to make delays and it will be used for PWM.
- Timer_1: It is the timer module which deals with timers in the microcontroller and enables us to make delays and it will be used for ultrasonic driver.
- Interrupt: It is the module that deals with the interrupts in the microcontroller in enables us to use ISR.
- ICU: Input capture unit module enables us to capture time for the use in ultrasonic sensor
- KeyPad: this driver allows us to use the push buttons as keypad.

Drivers documentation

- LCD: This driver allows us to use the LCD in the 4-bit mode.
- PWM: This driver allows us to control motor speed with the help of timers.
- Ultrasonic: This module allows us to use the ultrasonic sensor to know the distance infront of it.
- Pushbutton: It is a driver that allows us to know the state of a push button.
- APP: It is the module containing the application itself.

Drivers Documentation

- **DIO**

1. PinDirection_t DIO_setpindir(uint8_t u8_a_portid , uint8_t u8_a_pinid , uint8_t u8_a_pindir);

   Description: sets the direction of the GPIO to either Input or Output.

   Arguments: takes the port and pin numbers and pin direction either Input or output.

   Return: Return Error statement from VALID_DIRECTION or NOT_VALID_DIRECTION.

Continue Drivers Documentation

2. PinValue_t DIO_setpinvalue(uint8_t u8_a_portid , uint8_t u8_a_pinid , uint8_t u8_a_pinval);

   Description: This function sets a specified pin to high or low.

   Arguments: takes the port and pin numbers and the value of the Pin.

   Return: returns the error statement from VALID_VALUE or NOT_VALID_VALUE.

Continue Drivers Documentation

3. PinRead_t DIO_readpin(uint8_t u8_a_portid , uint8_t u8_a_pinid , uint8_t* u8_a_val);

   Description: This function reads the state of the pin in case of input either high or low.

   Arguments: takes the port and pin numbers and a variable to put the value inside.

   Return: returns the error statement from VALID_READ or NOT_VALID_READ.

Continue Drivers Documentation

- **Timer_0**

1. TMR0_init_error TMR0_init(void);

   Description: sets the direction of the GPIO to either Input or Output.

   Arguments: nothing to take.

   Return: Return Error statement from VALID_INIT or NOT_VALID_INIT.

Continue Drivers Documentation

2. TMR0_start_error TMR0_start(void);

   Description: starts the timer to count.

   Arguments: nothing to take.

   Return: Return Error statement from VALID_START or NOT_VALID_START.

Continue Drivers Documentation

3. TMR0_stop_error TMR0_stop(void);

   Description: stops the timer to count.

   Arguments: nothing to take.

   Return: Return Error statement from VALID_STOP or NOT_VALID_STOP.

Continue Drivers Documentation

4. TMR0_delay_error TMR0_delayms(uint32_t u32_a_delayms);

   Description: delays the system for a period of time.

   Arguments: takes the time to be delayed for in milliseconds.

   Return: Return Error statement from VALID_DELAY or NOT_VALID_DELAY.

Continue Drivers Documentation

- **Interrupt_0**

1. uint8_t SET_GLOBALINTERRUPT(void);

   Description: enables the global interrupt

   Arguments: nothing to take.

   Return: Return Error statement.

Continue Drivers Documentation

2. uint8_t INT0_init(void);

   Description: enables interrupt 0 through its registers

   Arguments: takes nothing.

   Return: Return Error statement.

Continue Drivers Documentation

- **ICU**

1. en_a_icuerrorstatus ICU_init(void);

   Description: specifies the external interrupt to work with.

   Arguments: nothing to take.

   Return: Return Error statement.

Continue Drivers Documentation

2. uint16t ICU_getvalue(void);

   Description: get the value of the timer when the event happens

   Arguments: takes nothing.

   Return: Return time stamp.

Continue Drivers Documentation

- **KeyPad**

1. uint8_t KEYPAD_init(void);

   Description: set the pin direction and the initial pin values

   Arguments: nothing to take.

   Return: Return Error statement.

Continue Drivers Documentation

2. uint8_t KEYPAD_getpressedkey(void);

Description: get the value of the pressed key

Arguments: takes nothing.

Return: Return the pressed key or “0” if no key is pressed.

Continue Drivers Documentation

- **LCD**

1. LCD_status LCD_init(void);

   Description: initializes the LCD through its Pins.

   Arguments: nothing to take.

   Return: Return Error statement.

Continue Drivers Documentation

2. LCD_status LCD_sendcmd(uint8_t u8_a_cmd);

   Description: sends commands to the LCD throughout its pins

   Arguments: command to be done.

   Return: Return error status.

Continue Drivers Documentation

3. LCD_status LCD_writechar(uint8_t u8_a_chr);

   Description: writes a spicific character on the LCD throughout its pins

   Arguments: character to write.

   Return: Return error status.

Continue Drivers Documentation

4. LCD_status LCD_writestr(uint8_t* u8_s_str);

   Description: writes a group of characters on the LCD throughout its pins.

   Arguments: pointer to the string.

   Return: Return error status.

Continue Drivers Documentation

- **PWM**

1. err_state MOTOR_control(uint8_t u8_a_mask, uint8_t u8_a_portNumber,float f_a_speedPercentage);

   Description: controls the speed of the motor.

   Arguments: the masked pins to be high or low, the port that uses the motor and the speed percentage.

   Return: Return Error statement.

Continue Drivers Documentation

- **Ultrasonic**

1. uint8_t USONIC_init(void);

    Description: set the direction of the pins and initializes the timer.

    Arguments: takes no arguments.

    Return: Return Error statement.

Continue Drivers Documentation

2. EN_USONIC_STATUS USONIC_getdistance(uint8_t *u8_a_distance);

   Description: gets the distance calculated through the ultrasonic sensor.

   Arguments: a pointer inwhich the distance will be put.

   Return: Return error status.

Continue Drivers Documentation

- **Pushbutton**

1. err_state BUTTON_init(uint8_t u8_a_pinNumber, uint8_t u8_a_portNumber);

   Description: set the direction of the pin.

   Arguments: takes the pin number and port number.

   Return: Return Error statement.

Continue Drivers Documentation

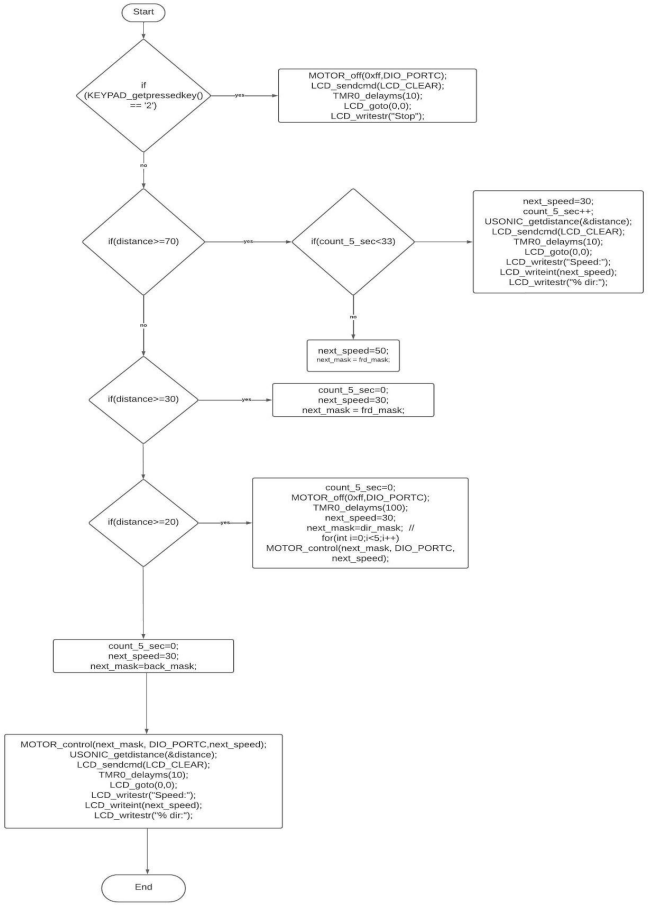
2. err_state BUTTON_read(uint8_t u8_a_pinNumber, uint8_t u8_a_portNumber, pin_state *en_a_value);

Description: gets the state of the Pushbutton

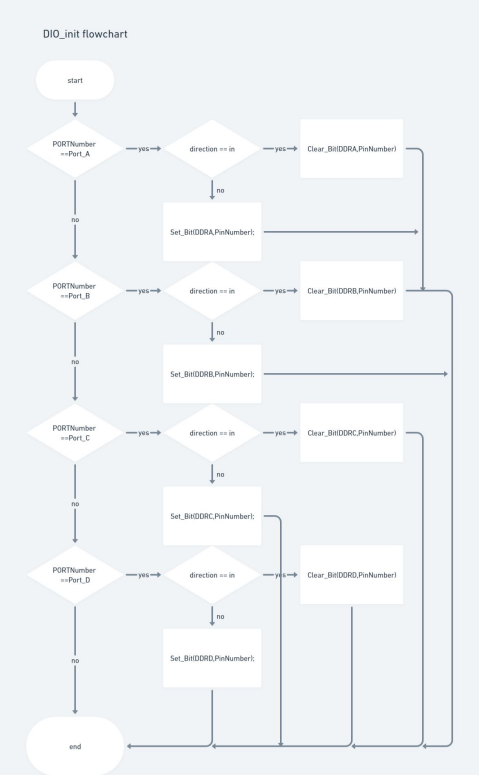
Arguments: takes the pin and port number and a pointer to put the state value in.

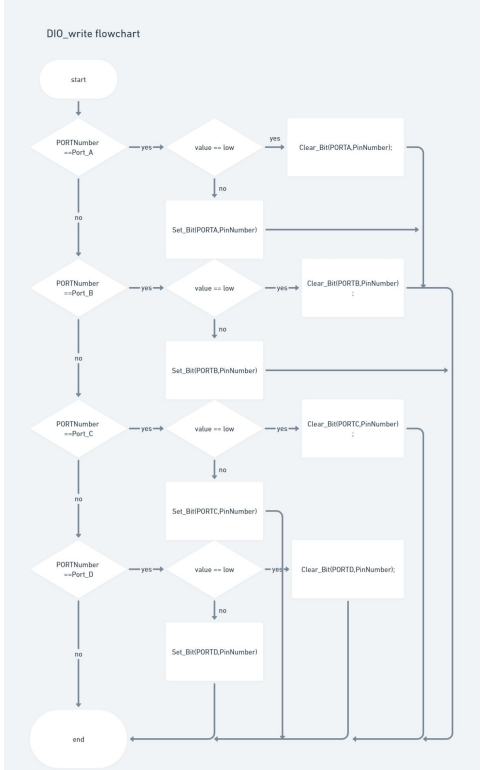
Return: Return error status.

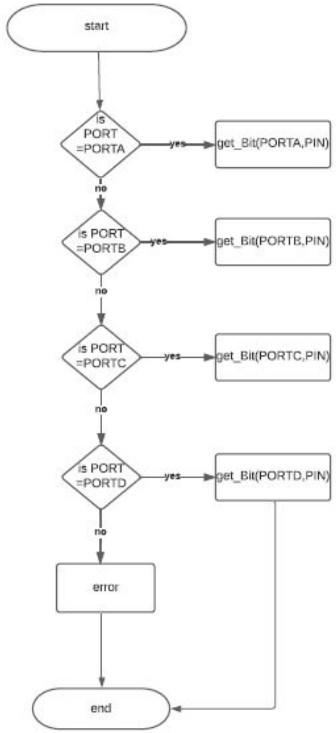
Project flowchart



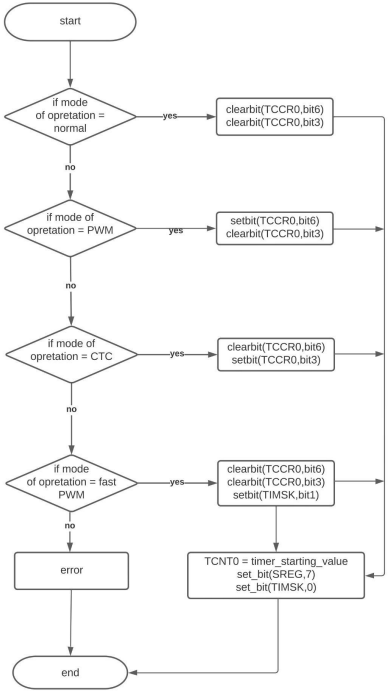
Driver flowcharts

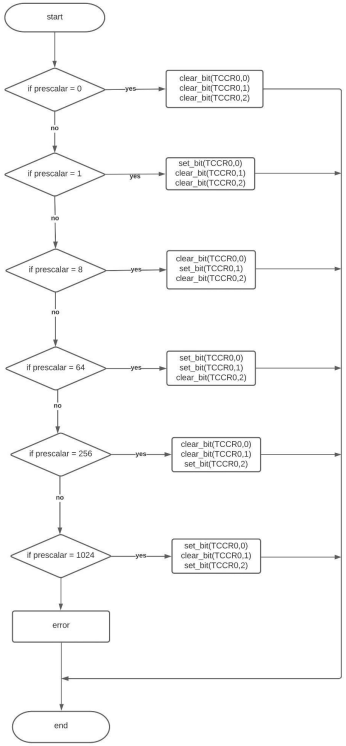


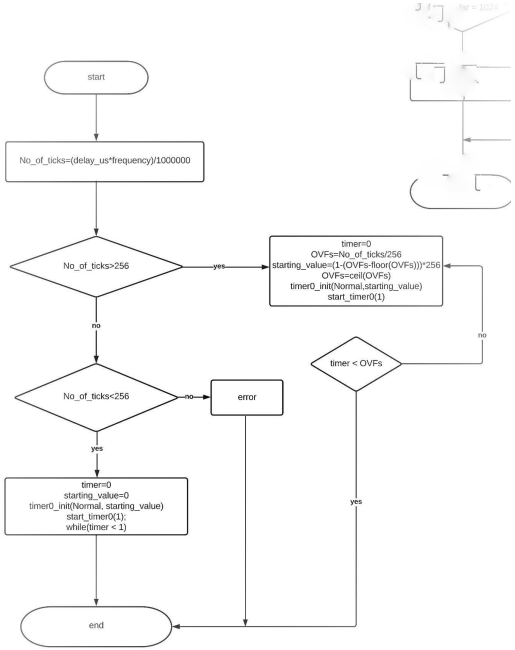




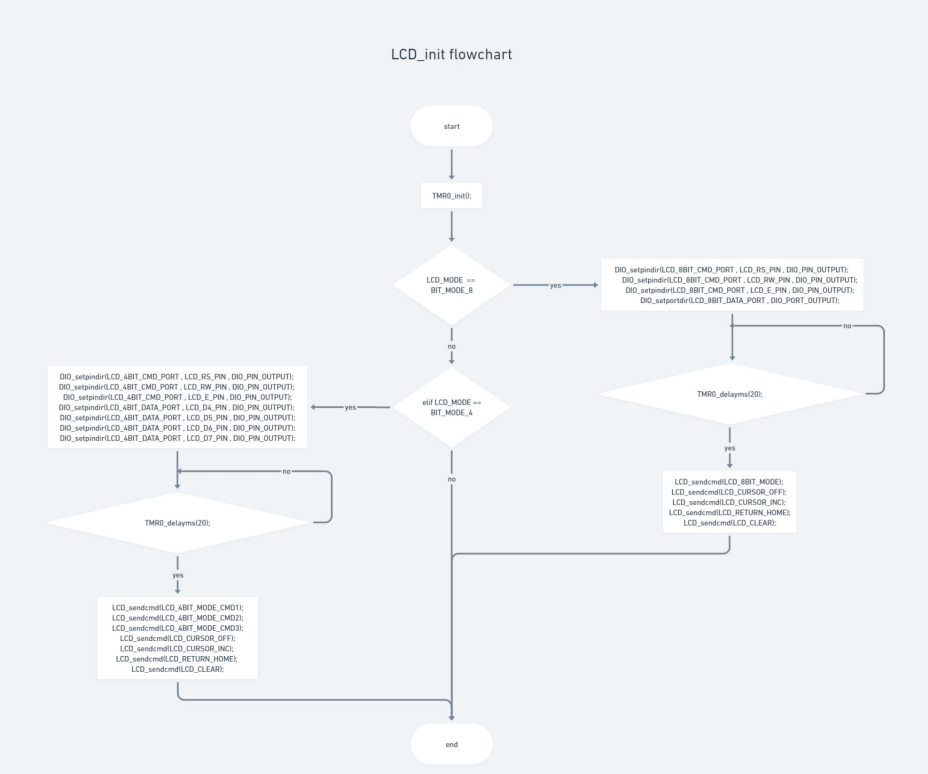
Continue Driver flowcharts

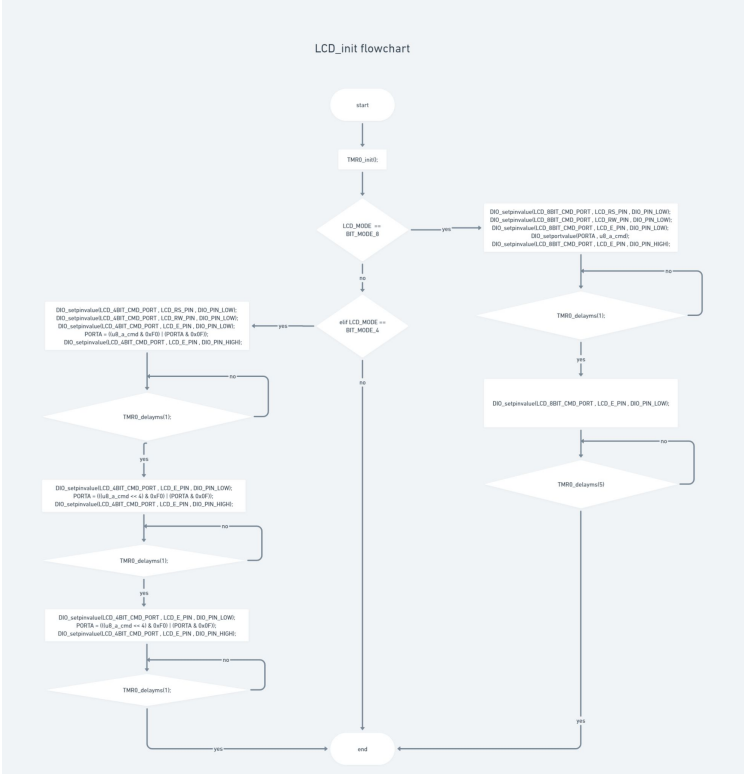




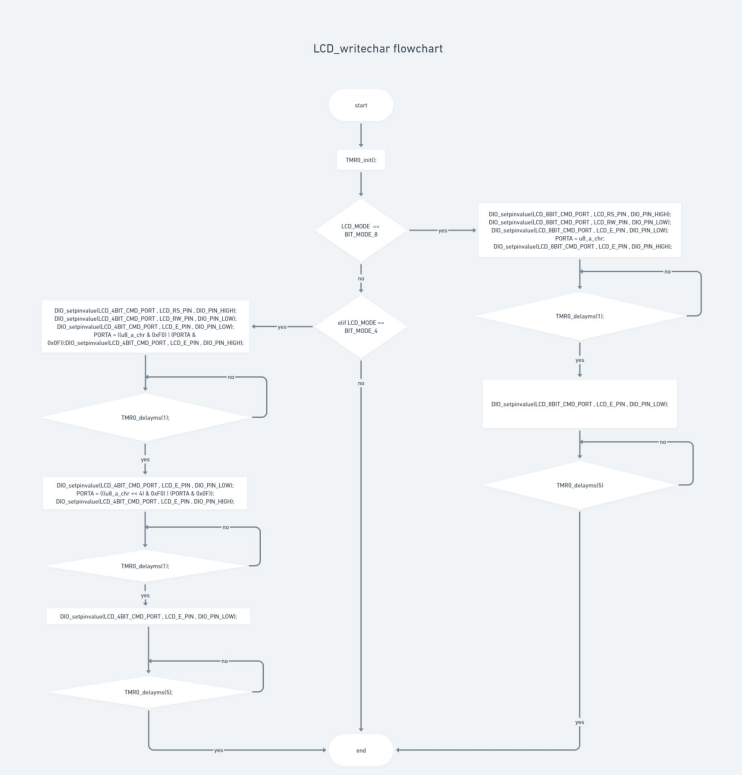


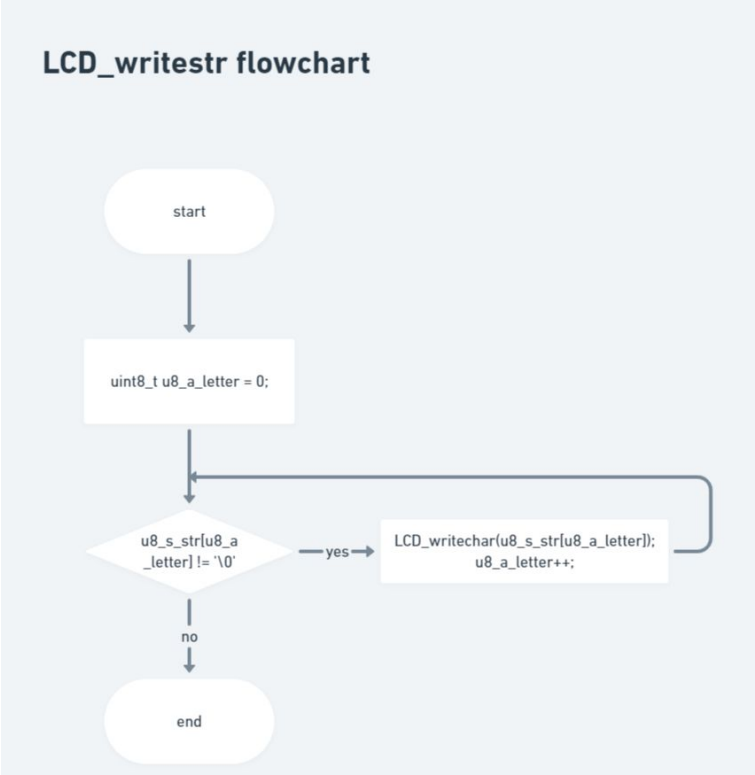
Continue Driver flowcharts



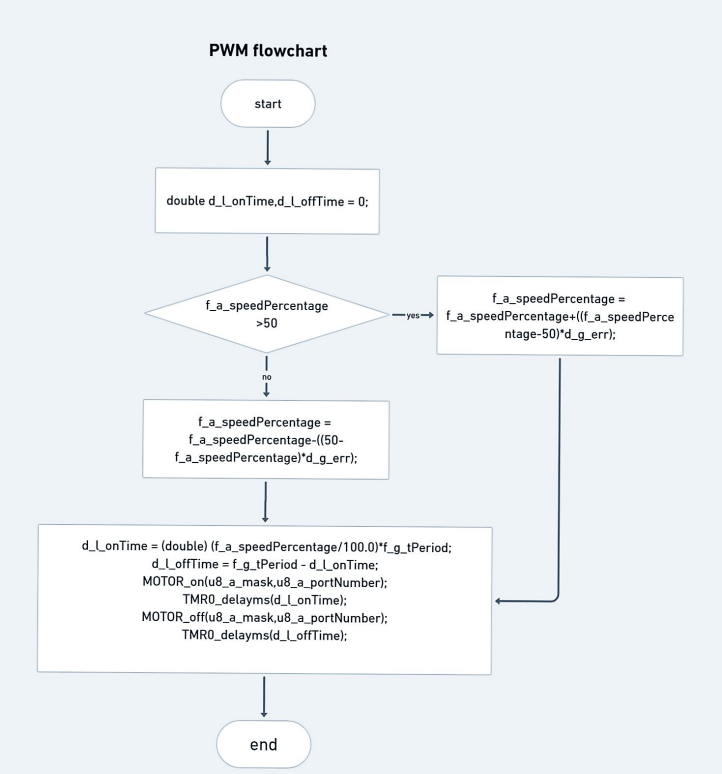


Continue Driver flowcharts





Continue Driver flowcharts



Continue Driver flowcharts