OBSTACLE AVOIDANCE ROBOT DESIGN V1

KAREEM MAGDY ALBOLAQI

SPRINT
9

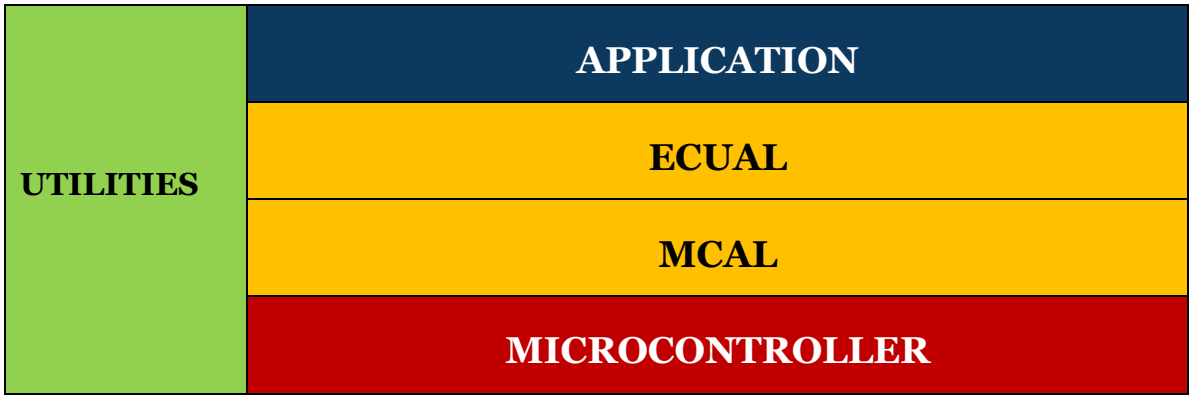
Table of Contents

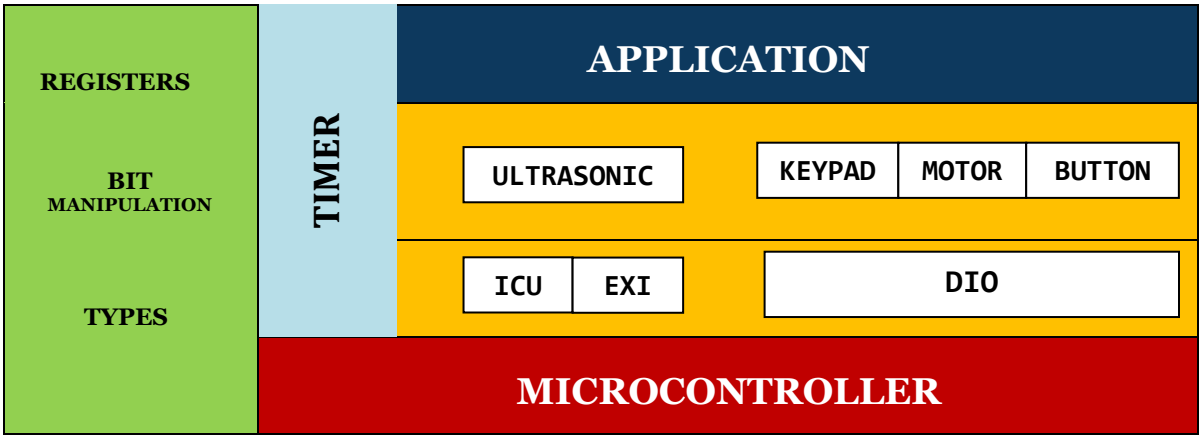
Project description 2
LAYERD ARCHTICTURE 3
DRIVERS 5
MCAL 5
DIO DRIVER 5
EXTERNAL INTERRUPT 7
TIMER DRIVER 9
ICU DRIVER 11
HAL 12
BUTTON DRIVER 12
KEYPAD DRIVER 13
LCD DRIVER 15
MOTOR DRIVER 18
ULTRASONIC DRIVER 19

Project description

- **The car starts initially from 0 speed**
- **The default rotation direction is to the right**
- **Press (Keypad Btn 1), (Keypad Btn 2) to start or stop the robot respectively**
- **After Pressing Start:**
  1. The LCD will display a centered message in line 1 “Set Def. Rot.”
  2. The LCD will display the selected option in line 2 “Right”
  3. The robot will wait for 5 seconds to choose between Right and Left
- **When PBUTTON0 is pressed once, the default rotation will be Left and the LCD line 2 will be updated**
- **When PBUTTON0 is pressed again, the default rotation will be Right and the LCD line 2 will be updated**
- **For each press the default rotation will changed and the LCD line 2 is updated**
- **After the 5 seconds the default value of rotation is set**
- **The robot will move after 2 seconds from setting the default direction of rotation.**
- **For No obstacles or object is far than 70 centimeters:**
  1. The robot will move forward with 30% speed for 5 seconds
  2. After 5 seconds it will move with 50% speed as long as there was no object or objects are located at more than 70 centimeters distance
  3. The LCD will display the speed and moving direction in line 1: “Speed:00% Dir: F/B/R/S”, F: forward, B: Backwards, R: Rotating, and S: Stopped
  4. The LCD will display Object distance in line 2 “Dist.: 000 Cm”
- **For Obstacles located between 30 and 70 centimeters**
  1. The robot will decrease its speed to 30%
  2. LCD data is updated
- **For Obstacles located between 20 and 30 centimeters**
  1. The robot will stop and rotates 90 degrees to right/left according to the chosen configuration
  2. The LCD data is updated
- **For Obstacles located less than 20 centimeters**
  1. The robot will stop, move backwards with 30% speed until distance is greater than 20 and less than 30
  2. The LCD data is updated
  3. Then preform point 8

LAYERD ARCHTICTURE



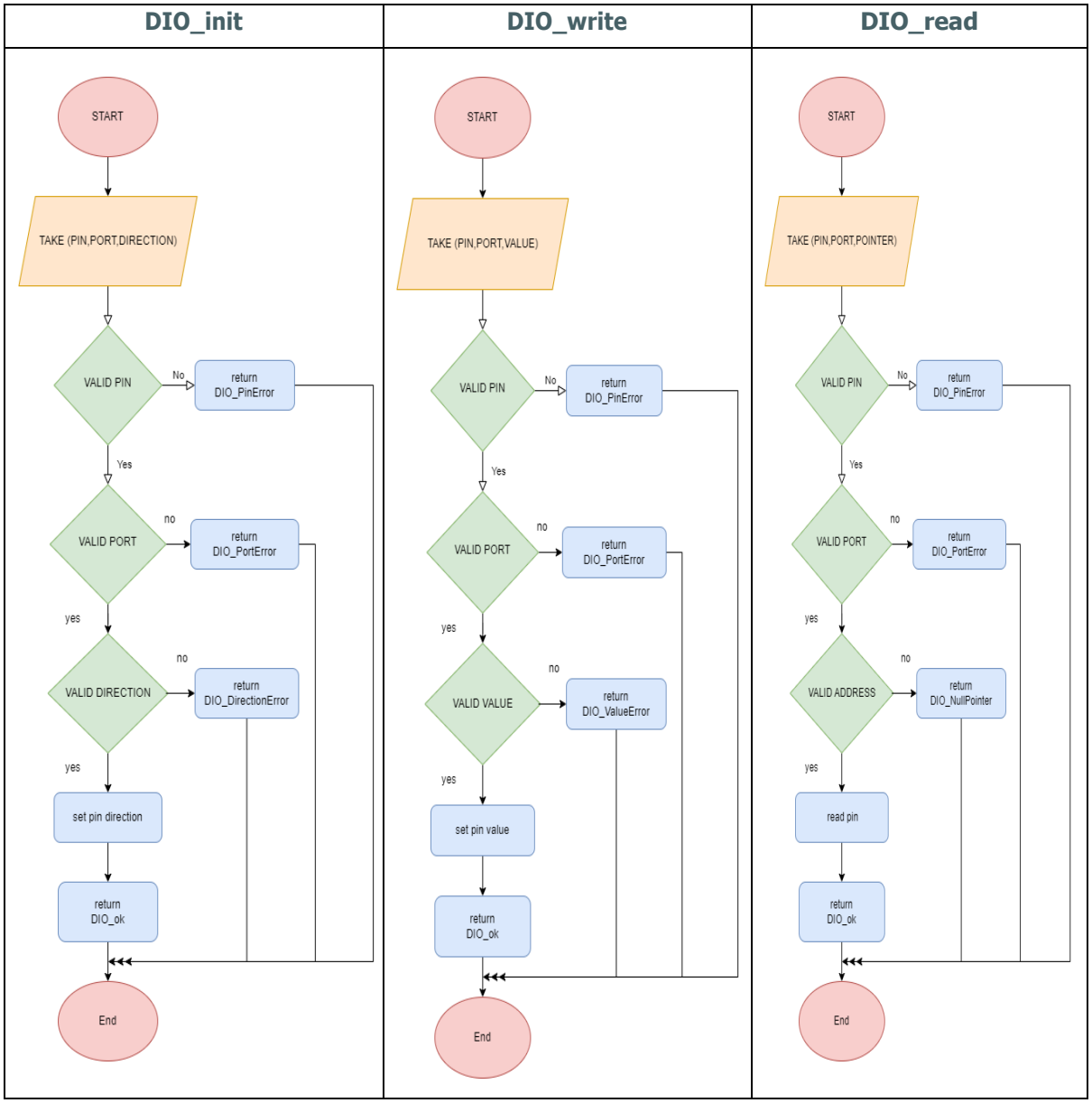


DRIVERS

MCAL

DIO DRIVER

```
Dio_ErrorStatus Dio_init(void);
Dio_ErrorStatus Dio_WriteChannel(Dio_Channel channel , Dio_status state);
Dio_ErrorStatus Dio_ReadChannel(Dio_Channel channel);
```



Configurations :

```c
typedef enum {
    portA_0,
    portA_1,
    portA_2,
    portA_3,
    portA_4,
    portA_5,
    portA_6,
    portA_7,

    portB_0,
    portB_1,
    portB_2,
    portB_3,
    portB_4,
    portB_5,
    portB_6,
    portB_7,

    portC_0,
    portC_1,
    portC_2,
    portC_3,
    portC_4,
    portC_5,
    portC_6,
    portC_7,

    portD_0,
    portD_1,
    portD_2,
    portD_3,
    portD_4,
    portD_5,
    portD_6,
    portD_7,

}Dio_Channel;
```

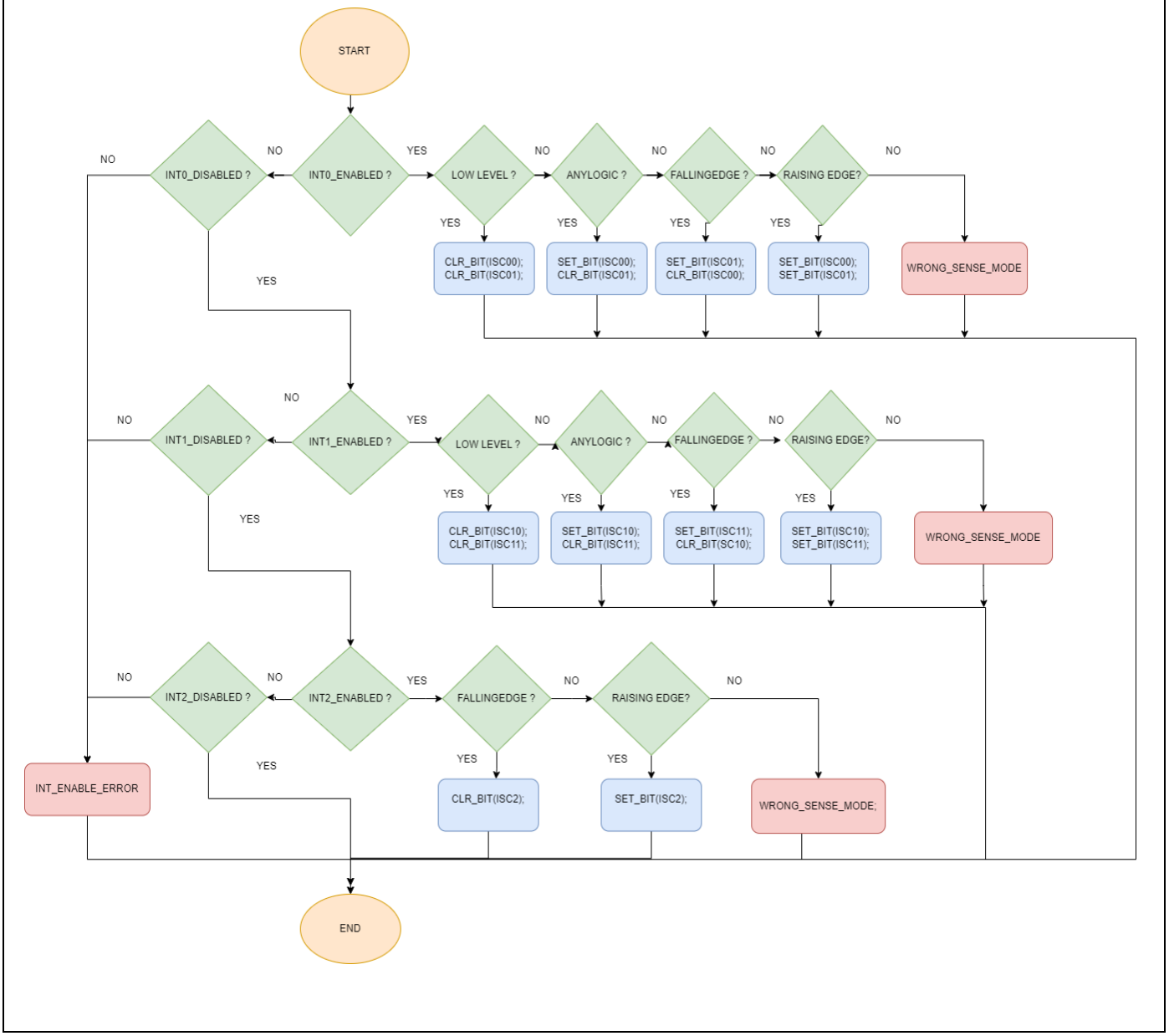
```c
typedef enum {
    LOW,
    HIGH

}Dio_status;

typedef enum {
    PIN_0,
    PIN_1,
    PIN_2,
    PIN_3,
    PIN_4,
    PIN_5,
    PIN_6,
    PIN_7,

}Dio_PIN;

typedef enum {
    PORT_A,
    PORT_B,
    PORT_C,
    PORT_D

}Dio_PORT;


typedef enum {
    INPUT,
    OUTPUT

}Dio_DIR;

typedef enum {
    PULLUP_OFF,
    PULLUP_ON

}Dio_PULLUP;
```

```
Dio_PinCfg Dio_PINS_Cfg[PIN_COUNT]= {

    //LCD
    {   PORT_A,PIN_4,OUTPUT,PULLUP_OFF},
    {   PORT_A,PIN_5,OUTPUT,PULLUP_OFF},
    {   PORT_A,PIN_6,OUTPUT,PULLUP_OFF},
    {   PORT_A,PIN_7,OUTPUT,PULLUP_OFF},
    {   PORT_B,PIN_0,OUTPUT,PULLUP_OFF},
    {   PORT_B,PIN_1;OUTPUT,PULLUP_OFF},

    //KEYPAD
    {   PORT_C,PIN_0,OUTPUT,PULLUP_OFF},
    {   PORT_C,PIN_1,OUTPUT,PULLUP_OFF},
    {   PORT_C,PIN_2,OUTPUT,PULLUP_OFF},
    {   PORT_C,PIN_3,OUTPUT,PULLUP_OFF},
    {   PORT_C,PIN_4,INPUT,PULLUP_ON},
    {   PORT_C,PIN_5,INPUT,PULLUP_ON},
    {   PORT_C,PIN_6,INPUT,PULLUP_ON},
    {   PORT_C,PIN_7,INPUT,PULLUP_ON},


    // BUTTON
    {   PORT_D,PIN_2,INPUT,PULLUP_ON},

};
```

EXTERNAL INTERRUPT

```
Ext_intErrorStatus ExtInt_init();
Ext_intErrorStatus INT0_SetCallback(void(*callback)(void));
Ext_intErrorStatus INT1_SetCallback(void(*callback)(void));
Ext_intErrorStatus INT2_SetCallback(void(*callback)(void));
```

ExtInt_init();



Configurations :

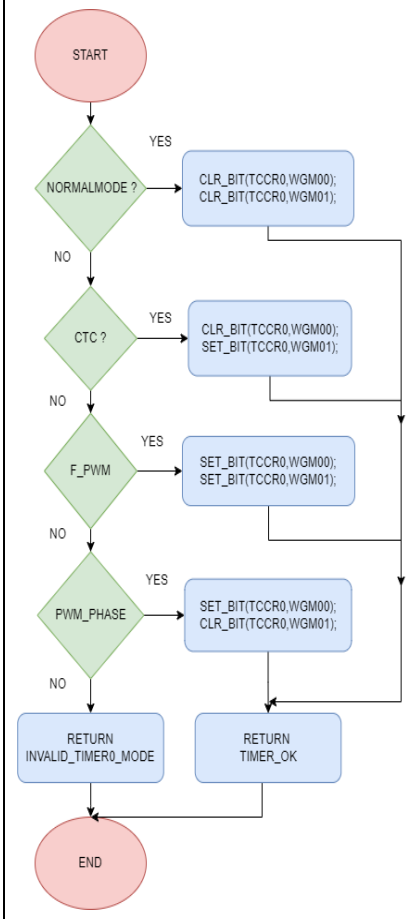
```
/************* CONFIGURATIONS **************/

/* NOTE : INT0,INT1 ----> CAN SENSE       ( LOW_LEVEL    , FALLING_EDGE , RAI
          INT2 ---------> CAN SENSE ONLY  ( FALLING_EDGE , RAISING_EDGE )
*/

#define INT0_ENABLE         ENABLE
#define INT1_ENABLE         ENABLE
#define INT2_ENABLE         DISABLE

#define INT0_SENSE_MODE     LOW_LEVEL
#define INT1_SENSE_MODE     FALLING_EDGE
#define INT2_SENSE_MODE     FALLING_EDGE
```

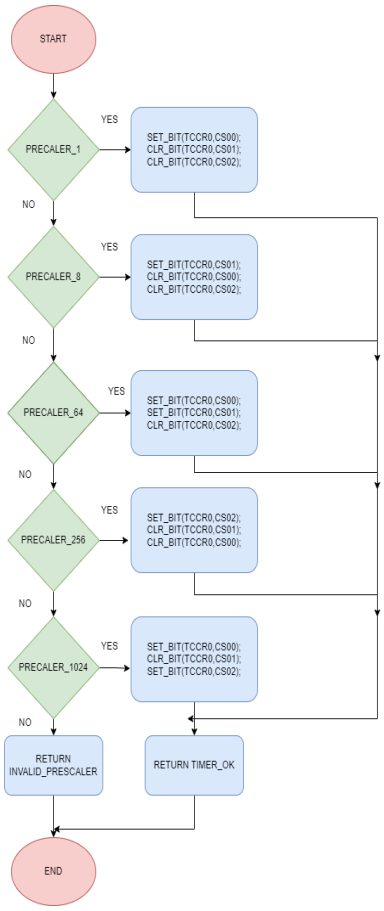
```
// ENABLE / DISABLE INTER
#define DISABLE
#define ENABLE


// INTERRUPT SENSE
#define LOW_LEVEL
#define ANY_LOGIC
#define FALLING_EDGE
#define RAISING_EDGE

//ERROR  TYPES
typedef enum {
    WRONG_SENSE_MODE,
    INT_ENABLE_ERROR,
    NULL_POINTE,
    EXT_INT_OK
}Ext_intErrorStatus;
```

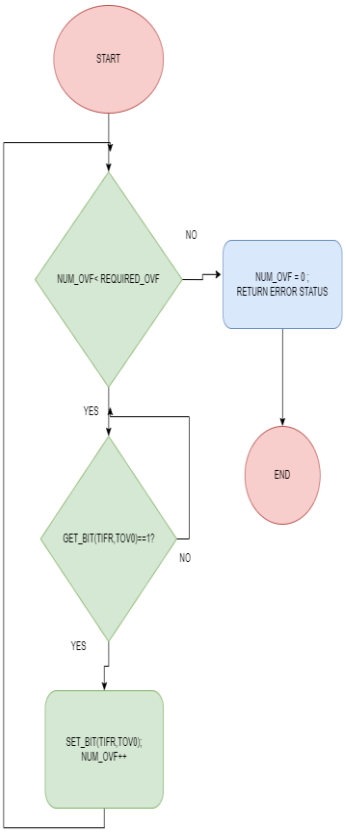
TIMER DRIVER

```
Timer_ErrorStatus TIMER_init(Timer_Mode mode);
Timer_ErrorStatus TIMER_start(Timer_Prescaler prescaler);
void TIMER_stop(void);
Timer_ErrorStatus TIMER_setIntialValue(uint8_t value);
Timer_ErrorStatus TIMER_OvfNum(uint32 overflow);
```

TIMER_0_init	TIMER_0_start	TIMER_OvfNum







Configurations :

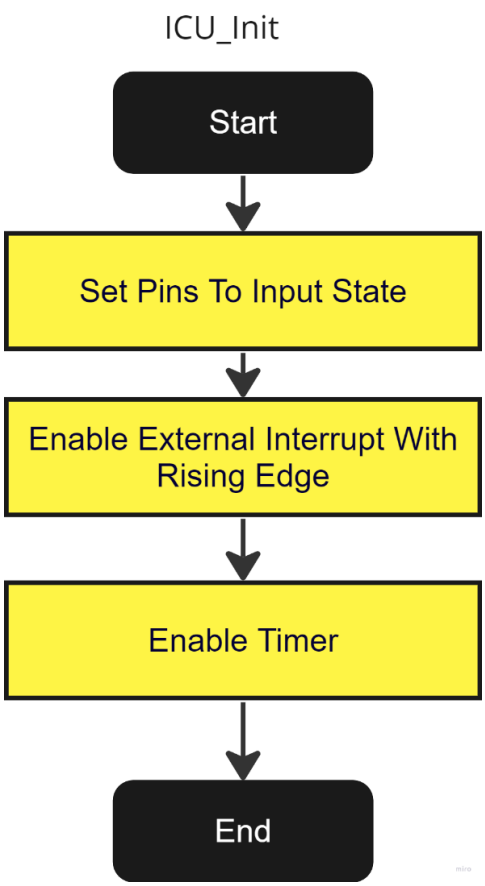
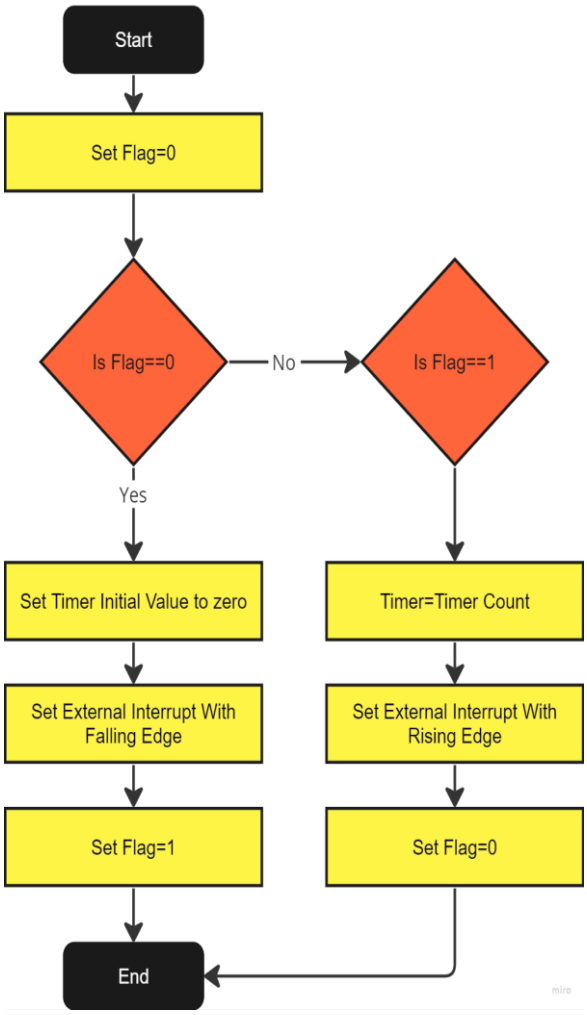
```c
typedef struct {

    Timer_Mode       mode ;
    Timer_Prescaler prescaler ;


}TIMER_cfg;

extern const TIMER_cfg Timer_cfgArray[TIMER_NUM] ;
```

```c
#include "timer.h"



const TIMER_cfg Timer_cfgArray[TIMER_NUM]={

    // TIMER_0
    {
        NORMAL_MODE , PRECALER_1
    },


};
```

```c
typedef enum {
    TIMER_0,
    TIMER_1,
    TIMER_2

    }TIMERS;

typedef enum {
    INVALID_PRESCALER,
    INVALID_MODE,
    INVALID_OVF,
    INVALID_VALUE,
    TIMER_OK
    }Timer_ErrorStatus;

typedef enum {
    NORMAL_MODE,
    FAST_PWM,
    CTC,
    PWM_PHASE_CORRECT

    }Timer_Mode;

typedef enum{
    PRECALER_1,
    PRECALER_8,
    PRECALER_64,
    PRECALER_32, // ONLY FOR TIMER_2
    PRECALER_128, // ONLY FOR TIMER_2
    PRECALER_256,
    PRECALER_1024,
    }Timer_Prescaler;
```

ICU DRIVER

```
Icu_ErrorStatus ICU_init();
Ext_intErrorStatus INT0_SetCallback(void(*callback)(void));
```

ICU_init()	CALLBACK FINCTION
	

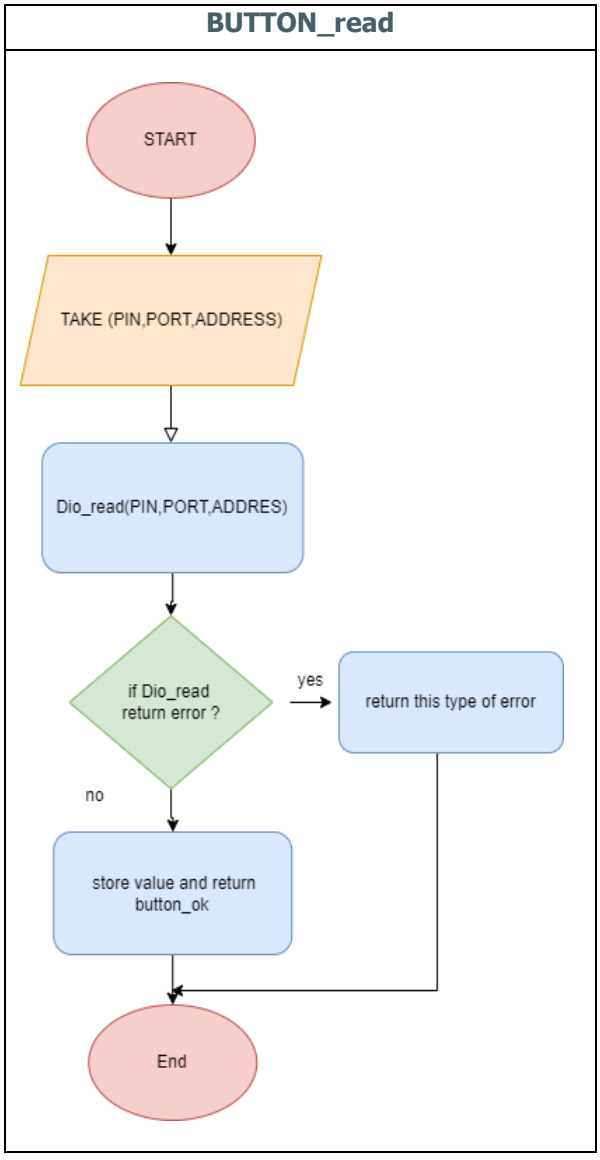
Configurations :

WE CAN USE DIO AND EXI CONFIGURATION TO SET OUR ICU DRIVER CONFIGURATION

HAL

BUTTON DRIVER

```
Dio_ErrorStatus BUTTON_read(PORT_NUM portnum ,PIN_NUM pinnum, uint8_t *value);
```

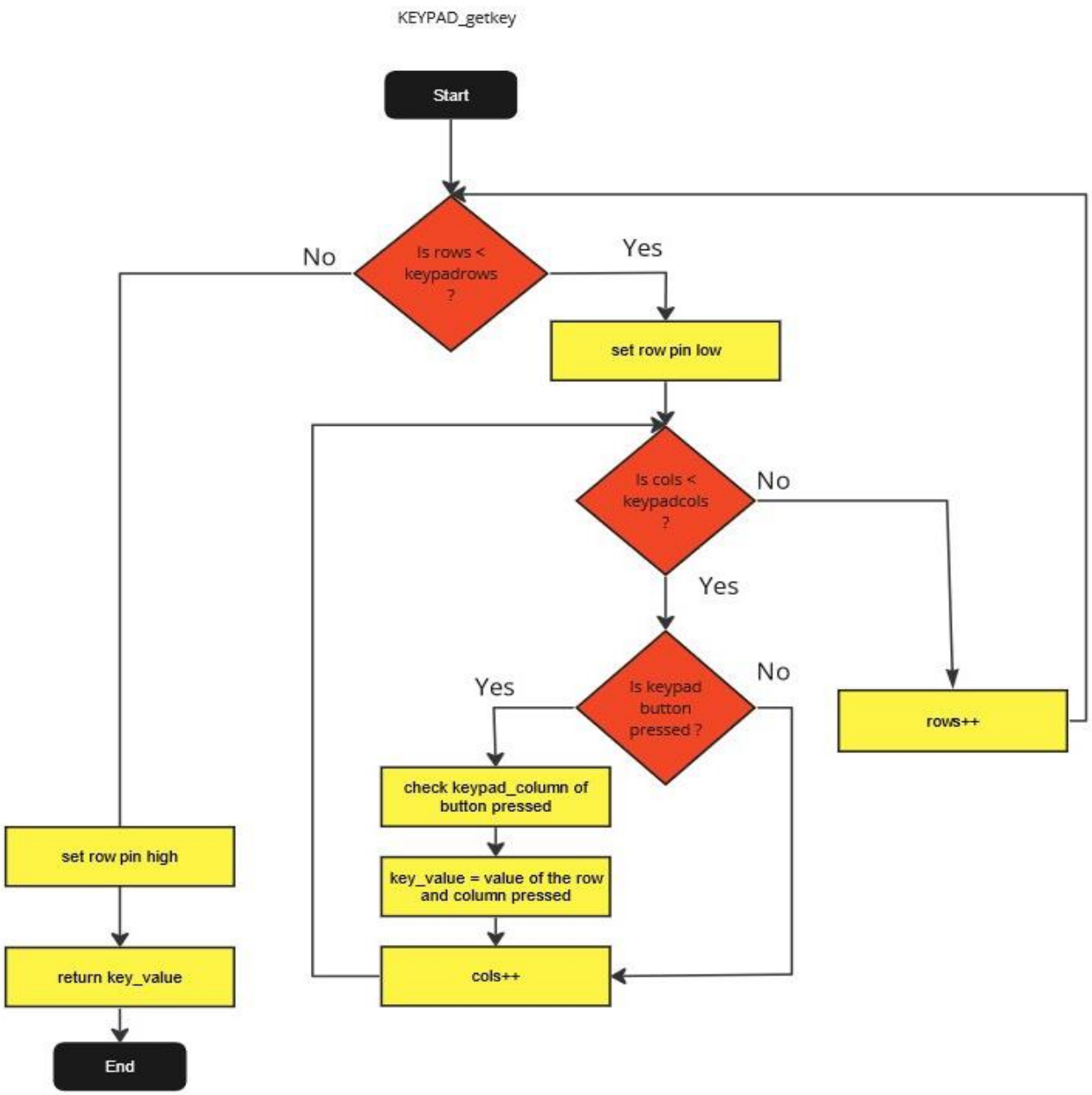


Configurations :

WE CAN USE DIO AND EXI CONFIGURATION TO SET OUR BUTTON DRIVER CONFIGURATION

KEYPAD DRIVER

```
uint8_t KEYPAD_getKey(void);
```



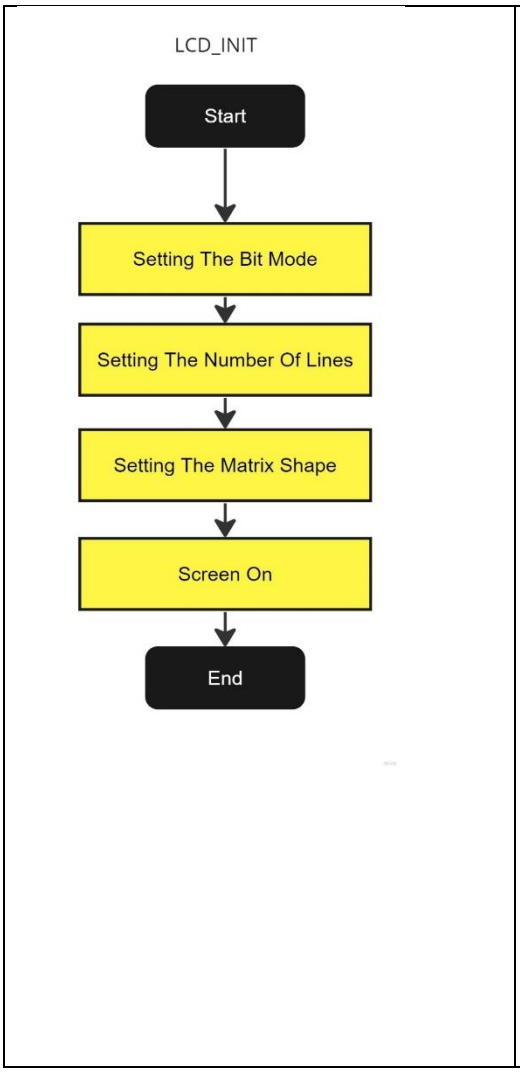
Configurations :

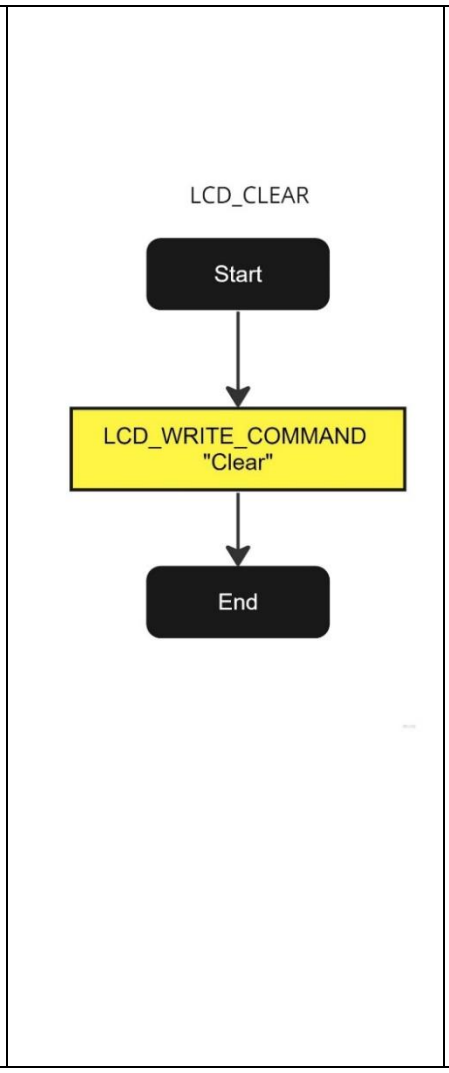
WE CAN USE DIO TO SET OUR KEYPAD DRIVER CONFIGURATION

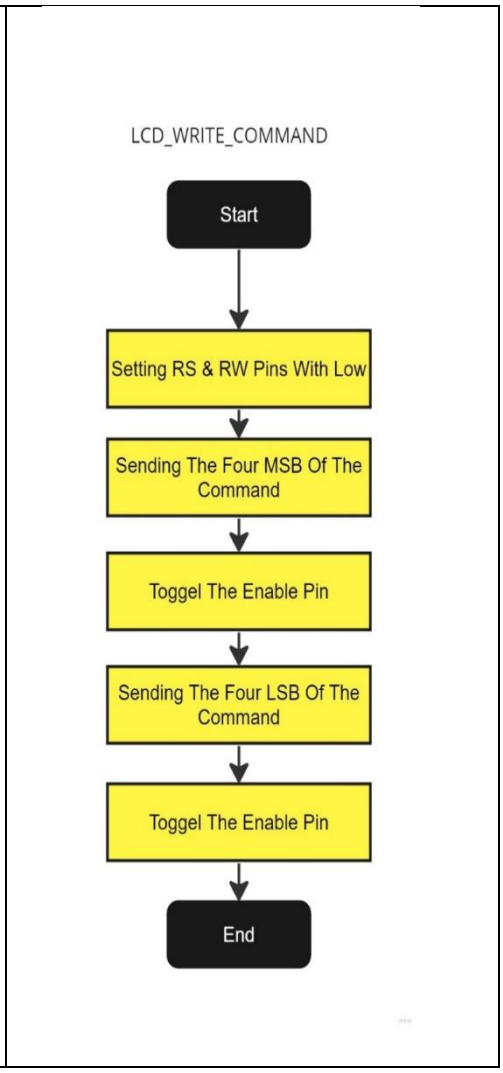
```
// PINS
#define FIRST_ROW_PIN               pinc2
#define LAST_ROW_PIN                pinc4
#define FIRST_COL_PIN               pinc5
#define LAST_COL_PIN                pinc7


typedef enum {
    COL_0 = FIRST_COL_PIN,
    COL_1,
    COL_2,
    COL_3,
}COLUMN;

typedef enum {
    ROW_0 = FIRST_ROW_PIN,
    ROW_1,
    ROW_2,
    ROW_3,
}ROW;
```

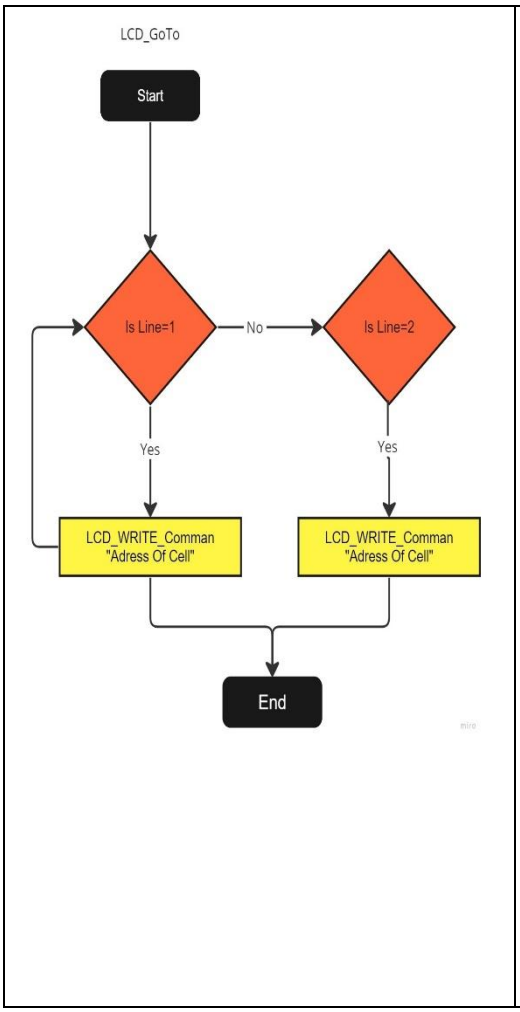
LCD DRIVER

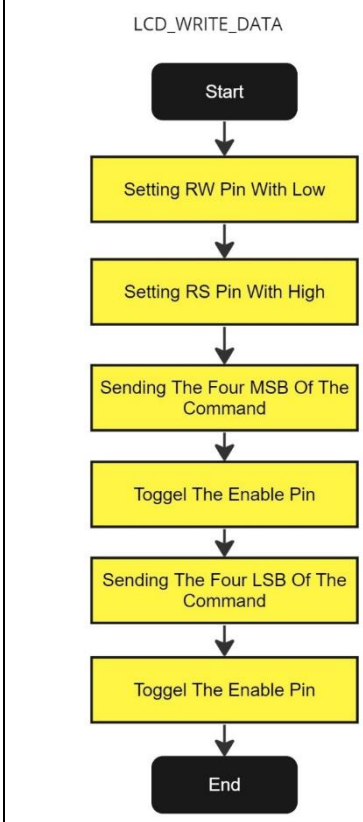
```
void LCD_WRITE_COMMAND(uint8_t a_COMMAND);
void LCD_WRITE_DATA(uint8_t a_DATA);
void LCD_INIT(void);
void LCD_Clear(void);
void LCD_GoTo(uint8_t a_line,uint8_t a_cell);
void LCD_Write_Charecter(uint8_t a_char);
```

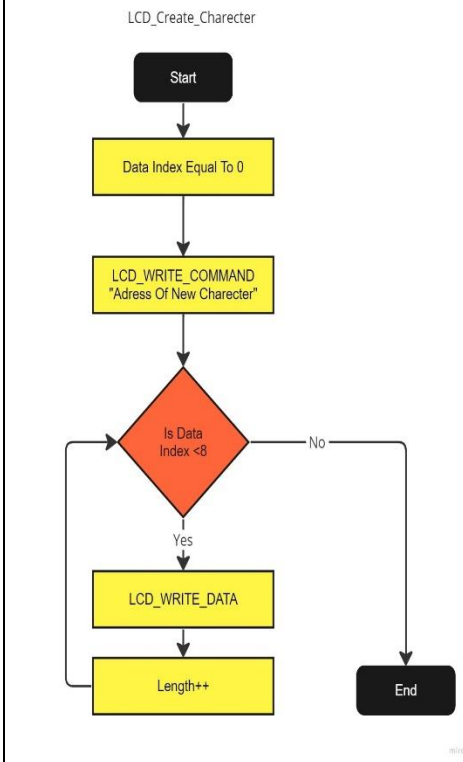







LCD_GoTo
Start
Is Line=1
No
Is Line=2
Yes
Yes
LCD_WRITE_Comman
"Adress Of Cell"
LCD_WRITE_Comman
"Adress Of Cell"
End


LCD_WRITE_DATA
Start
Setting RW Pin With Low
Setting RS Pin With High
Sending The Four MSB Of The Command
Toggel The Enable Pin
Sending The Four LSB Of The Command
Toggel The Enable Pin
End


LCD_Create_Charecter
Start
Data Index Equal To 0
LCD_WRITE_COMMAND
"Adress Of New Charecter"
Is Data
Index <8
No
Yes
LCD_WRITE_DATA
Length++
End

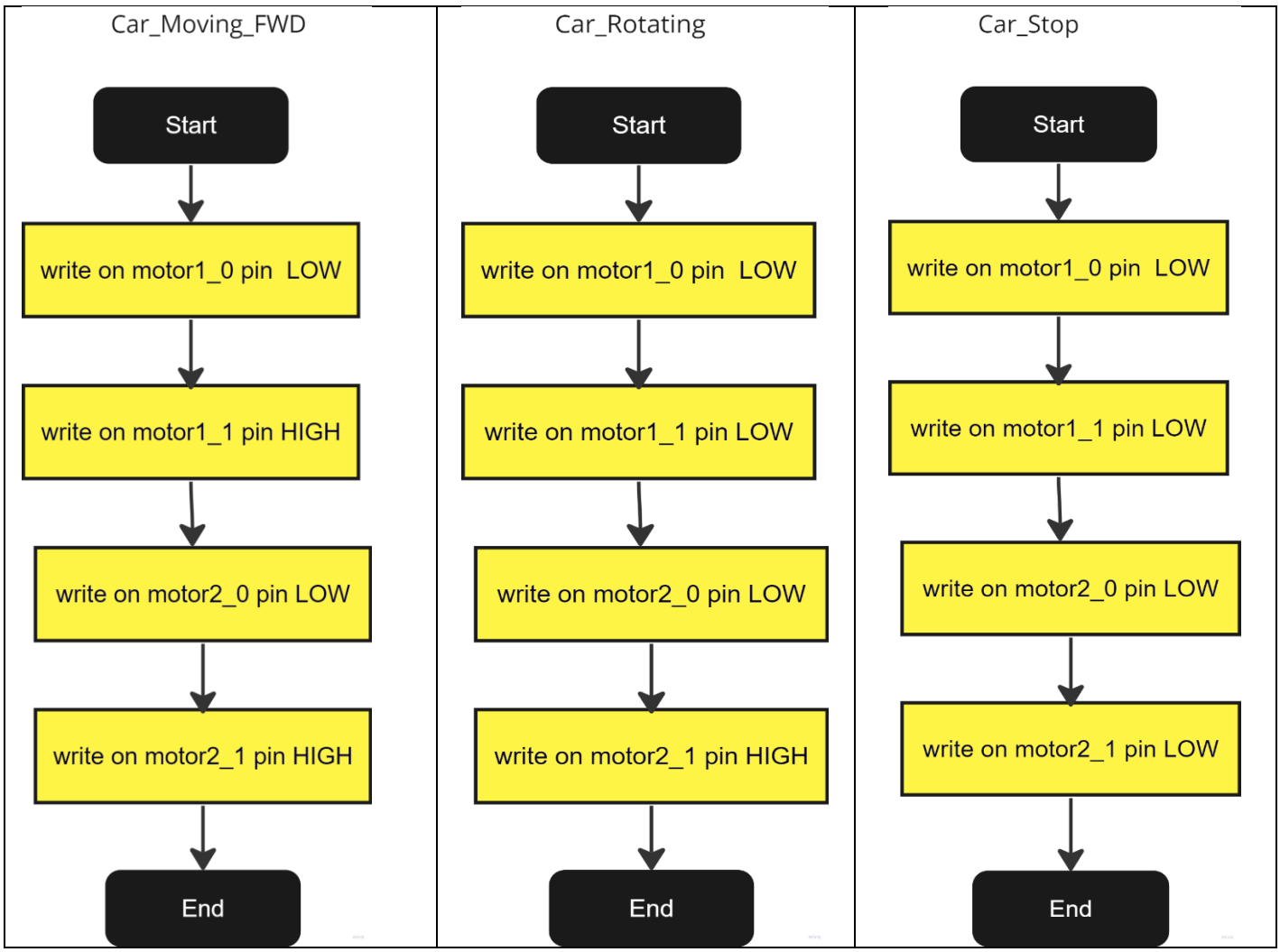
Configurations :

WE CAN USE DIO TO SET OUR KEYPAD DRIVER CONFIGURATION

```
#define _4_bit_mode     0
#define _8_bit_mode     1
/************************ config*************************************************/
#define LCD_Mode       _4_bit_mode

#define  LCD_PORT      PA

#define   D4           pina4
#define   D5           pina5
#define   D6           pina6
#define   D7           pina7

#define   RS           pina1
#define   RW           pina2
#define   EN           pina3

/********************************************************************************/
```

MOTOR DRIVER

```
void Car_Moving_FWD(void);
void Car_Rotating(void);
void Car_Stop(void);
```



Configurations :

WE CAN USE DIO TO SET OUR KEYPAD DRIVER CONFIGURATION

ULTRASONIC DRIVER

```
void get_Distance(uint_8 * a_distance);
```