Air Condition Design

TEAM 8 (RETURN BRAIN)

ASSINED BY:

MOHAMED SAYAD

MOHAMED ABDELSALAM

EBRAHEM MOSTAFA

Table of contents

Contents

Sprints

Firstly: Project Description:

Description

Hardware Requirements

LCD

KEYPAD

TEMPERATURE SENSOR

BUZZER

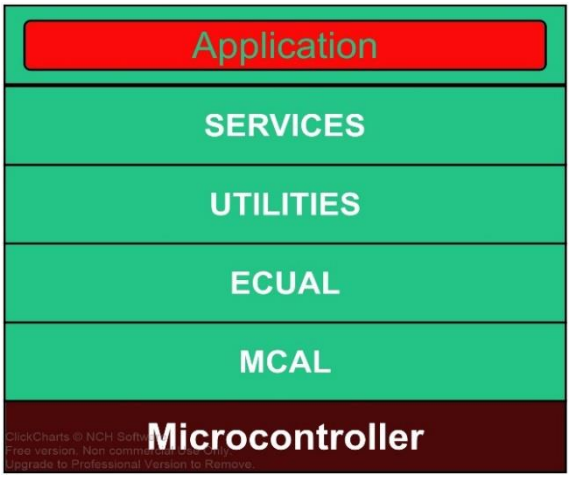
Software Requirements

1. The user set temperature (20 ~ 35 Celsius)
2. KEYPAD 1 INCREAMENT BUTTON for incrementing input range
3. KEYPAD 2 DECCREAMENT BUTTON for decrementing input range
4. KEYPAD 3 SET BUZZER BUTTON for choosing the setting temperature
5. When temperature exceed set temperature, BUZZER ON
6. Display the current reading of temperature sensor
7. KEYPAD 4 STOP BUZZER BUTTON for stop BUZZER
8. KEYPAD 5 RESET BUTTON for resetting temperature to 20 Celsius
9. This will be repeated forever
10. The sequence is described below
    1. LCD display “Welcome Message” for one second
    2. LCD clear
    3. LCD display “Default Temp is 20” for one second
    4. LCD clear
    5. LCD display “Please Choose The Required Temp” for half second
    6. Press KEYPAD 1 OR 2 for in/decrement temperature
    7. Press KEYPAD 3 for set temperature
    8. After step 7, any Press for KEYPAD 1,2 OR 3:
        1. display “This Operation Is Not Allowed” for half second time out
    9. LCD display current temperature
    10. LCD display bell-shape if the current temperature exceed setting temperature

Sprints

Secondly: Layered architecture:

1- Microcontroller
2- MCAL
3- ECUAL
4- UTILITIES
5- SERVICES
6- Application



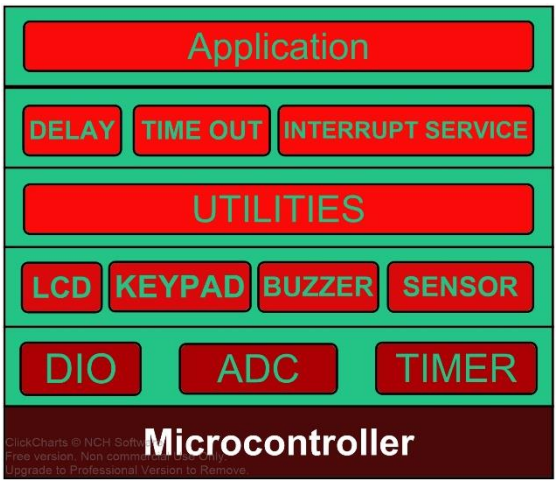
Thirdly : System modules:

1- Specify system modules/drivers:

- DIO, TIMER, ADC
- LCD, KEYPAD,SENSOR,BUZZER,
- DELAY, TIME OUT, INTERRUPT SERVICE
- APPLICATION

2- Assign each module to its related layer:

- By drawing



Fourthly: APIs:

DIO APIs:

```
void DIO_InitPin (PIn_name pin ,PIN_Status status );
void DIO_init (void);
void DIO_WRitePin (PIn_name pin ,Voltage_type s);
Voltage_type DIO_ReadPin(PIn_name pin);
void DIO_WritePort(PORT_Type Port,u8 data);
```

Sprints

TIMER APIs:

```
void TIMER_init (uint8_t Mode,uint8_t intial_value);
    void TIMER_start (uint8_t prescaler_value);
       void TIMER_set(uint8_t intial_value);
       void TIMER_getStatus(uint8_t *value);
              void TIMER_Stop (void);
    void TIMER2_init (u8 Mode,u8 intial_value);
      void TIMER2_start (u8 prescaler_value);
         void TIMER2_set(u8 intial_value);
         void TIMER2_getStatus(u8 *value);
             void TIMER2_Stop (void);
```

ADC APIs:

```
void ADC_init(PIn_name channel, uint8_t V_ref_type ,
  uint8_t Diff_OR_Single , uint8_t ADCH_OR_ADCL ,
 uint8t    uint8_t  prescaler , uint8_t INT_init);

        uint8_t ADC_Read(PIn_name channel);
```

Sprints

LCD APIs:

```
void LCD_init (void);
void LCD _sendcommand (uint8_t cmnd);
void LCD_sendchar (uint8_t char_data);
void LCD _sendstring(uint8_t *str);
void LCD_setcursor (uint8_t row, uint8_t column);
void LCD_clear (void);
void LCD_customchar(uint8_t *pattern, uint8_t location);
LCD_floattostring (f32_t float_value);
```

KEYPAD APIs:

```
Keypad_Status_en KEYPAD_Init(PIn_name First_Output,PIn_name  Firs_Input);

unit8_t KEYPAD_GetNum_time(unit8_t timeout);

static unit8_t KEYPAD_GetKey(void);
```

Sprints

SENSOR APIs:

```
void Temp_init(PIn_name channel);
 uint8_t Temp_Read(PIn_name channel);
```

BUZZER APIs:

```
void buzz_init(PIn_name pin_num);
void buzz_ON();
void buzz_OFF();
```

INTERRUPT SERVICE APIs:

```
#  define ISR(vector,...)            \
void vector (void)    __attribute__
((signal,used))__VA_ARGS__ ; \
void vector (void)
```

Sprints

DELAY APIs:

```
void Delay (uint8_t milliseconds);
```

TIMEOUT APIs:

```
Static void TIMER_out_Stop (void);
Static void TIME_out_init (void);
void TIMER_out (uint8_t milliseconds);
static void TIMER_ISR(void);
static void Timer0_Ovf_CALLBACK (void (*copyFuncptr)
(void));
```

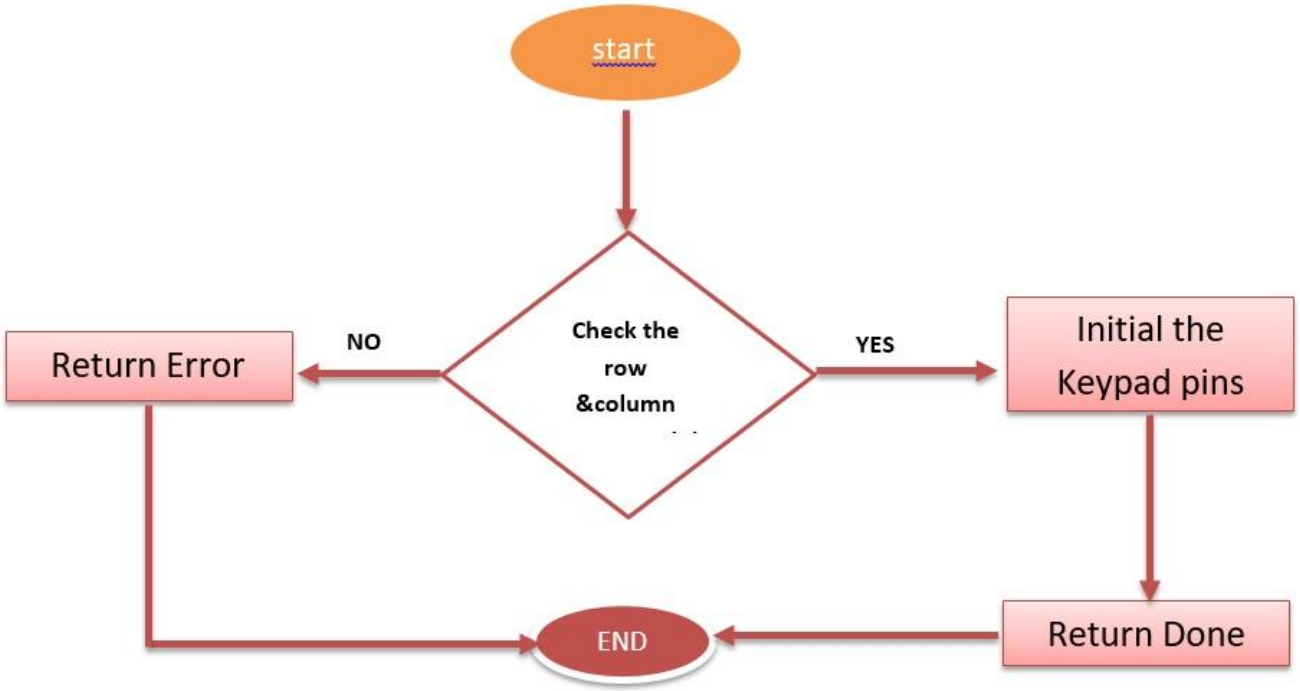
APPLICATION APIs:

```
void APP_Init(void);
void APP_Start(void);
```

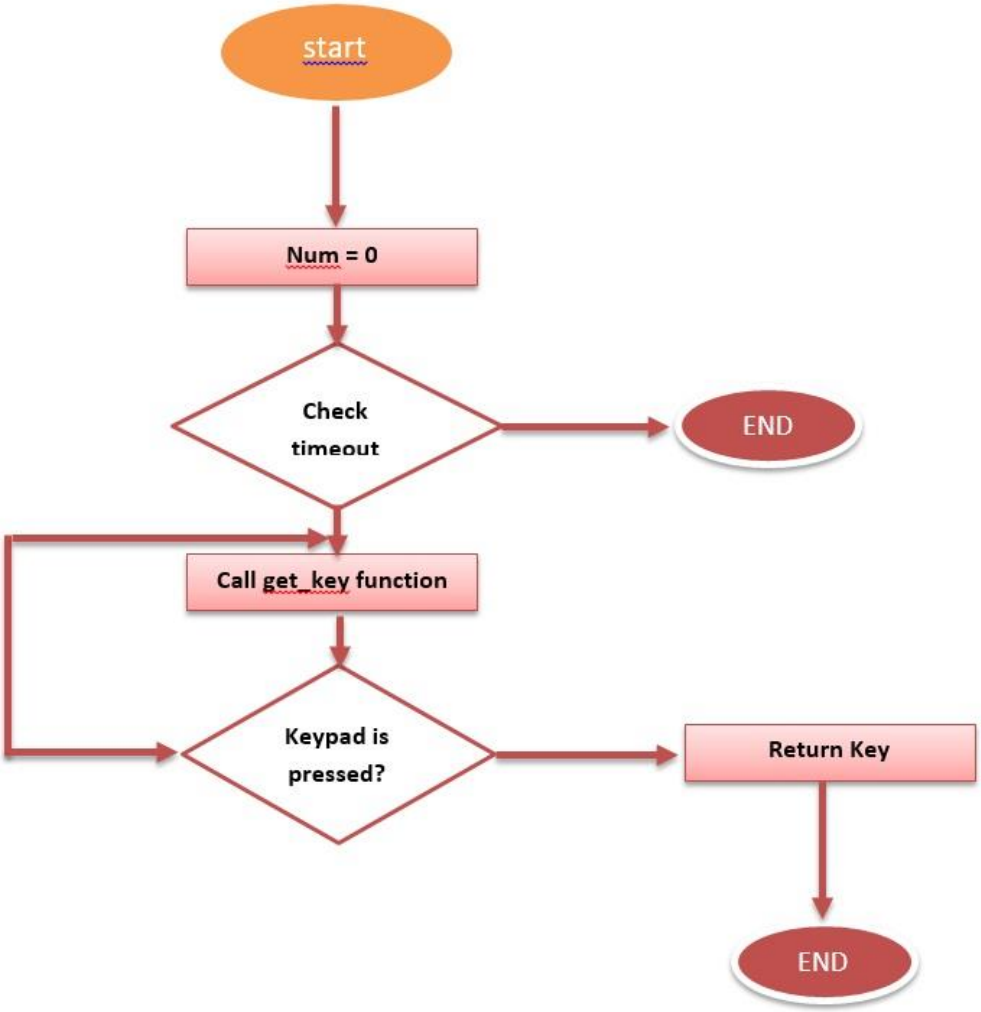
Fifthly: Flowcharts APIs:

ECUAL FLOWCHARTS:

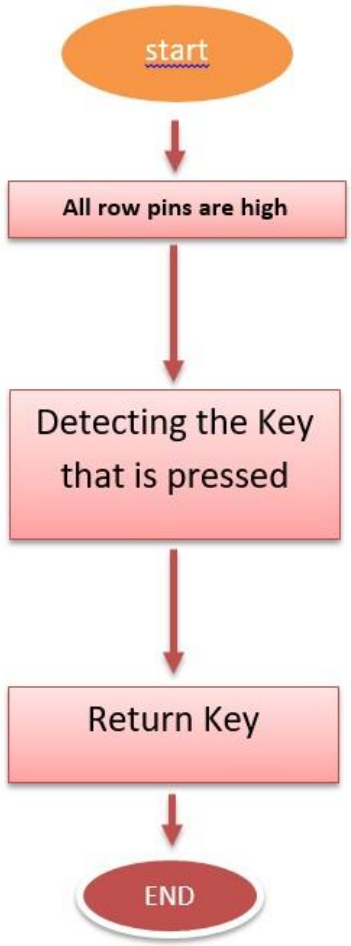
Keypad APIs	Keypad_Status_en KEYPAD_Init(PIn_name First_Output,PIn_name Firs_Input);



Keypad APIs	unit8_t KEYPAD_GetNum_time(unit8_t timeout);



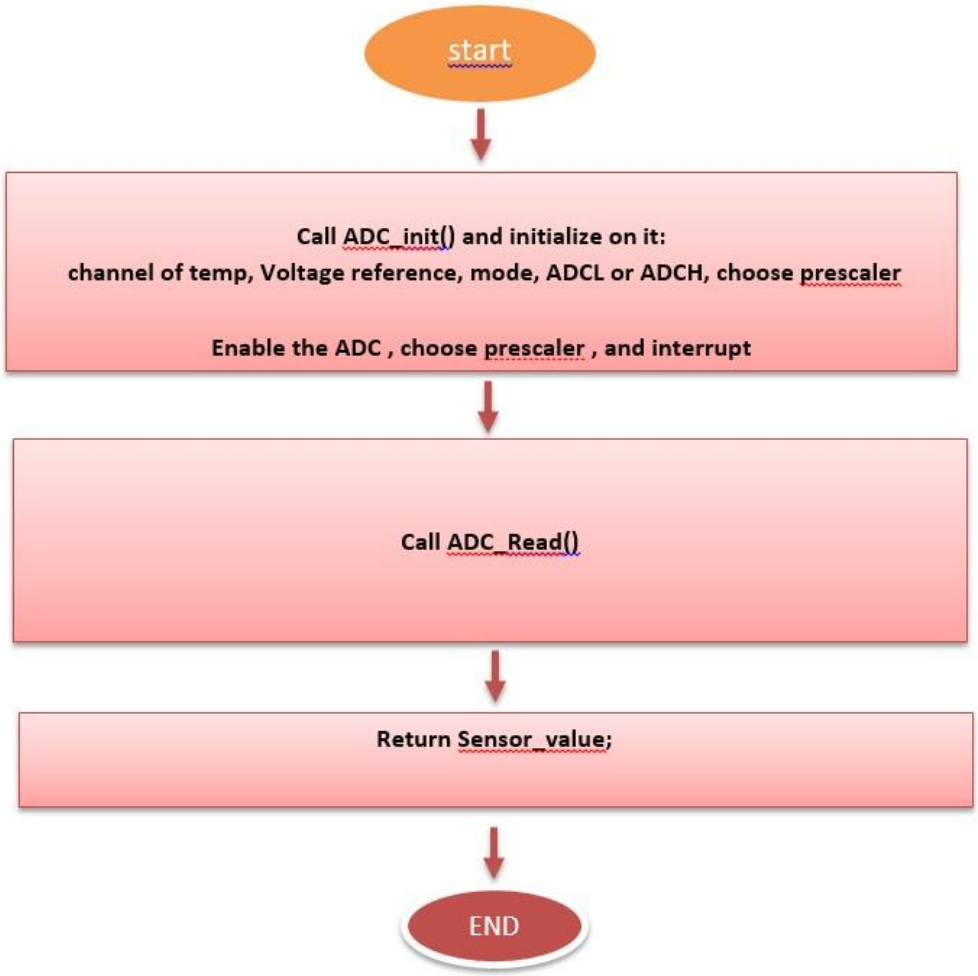
Keypad APIs	static unit8_t KEYPAD_GetKey(void);



Sensor APIs

void Temp_init(PIn_name channel);

uint8_t Temp_Read(PIn_name channel);



Buzzer APIs

```
void bazz_init(PIn_name pin_num);
void bazz_ON();
void bazz_OFF();
```

LCD

APIs

void LCD_init (void);

void LCD _sendcommand (uint8_t cmnd);

void LCD _sendstring(uint8_t *str);

void LCD_sendchar (uint8_t char_data);

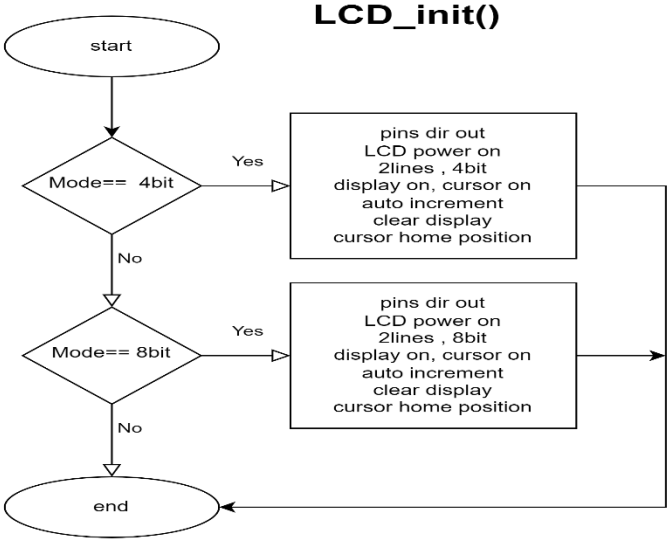
void LCD_setcursor (uint8_t row, uint8_t column);

void LCD_clear (void);

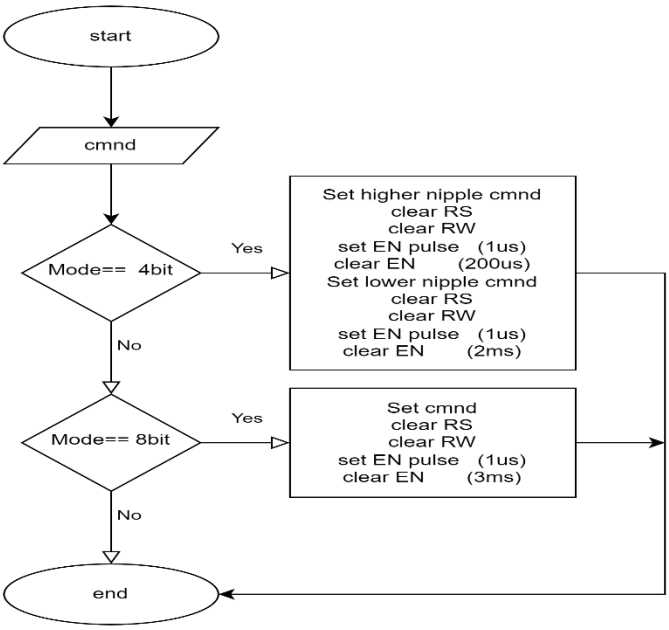
void LCD_customchar(uint8_t *pattern, uint8_t location);

void LCD_floattostring (f32_t float_value)

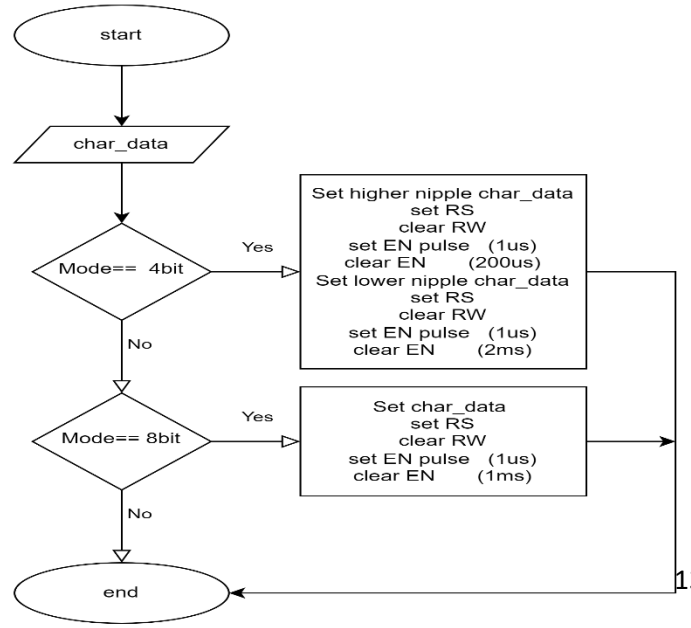
LCD_init()

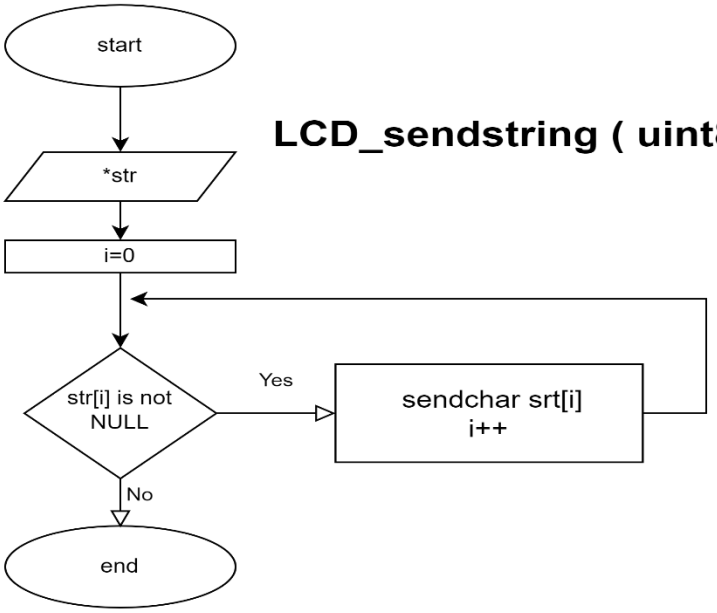


LCD_sendcommand(uint8_t cmnd)



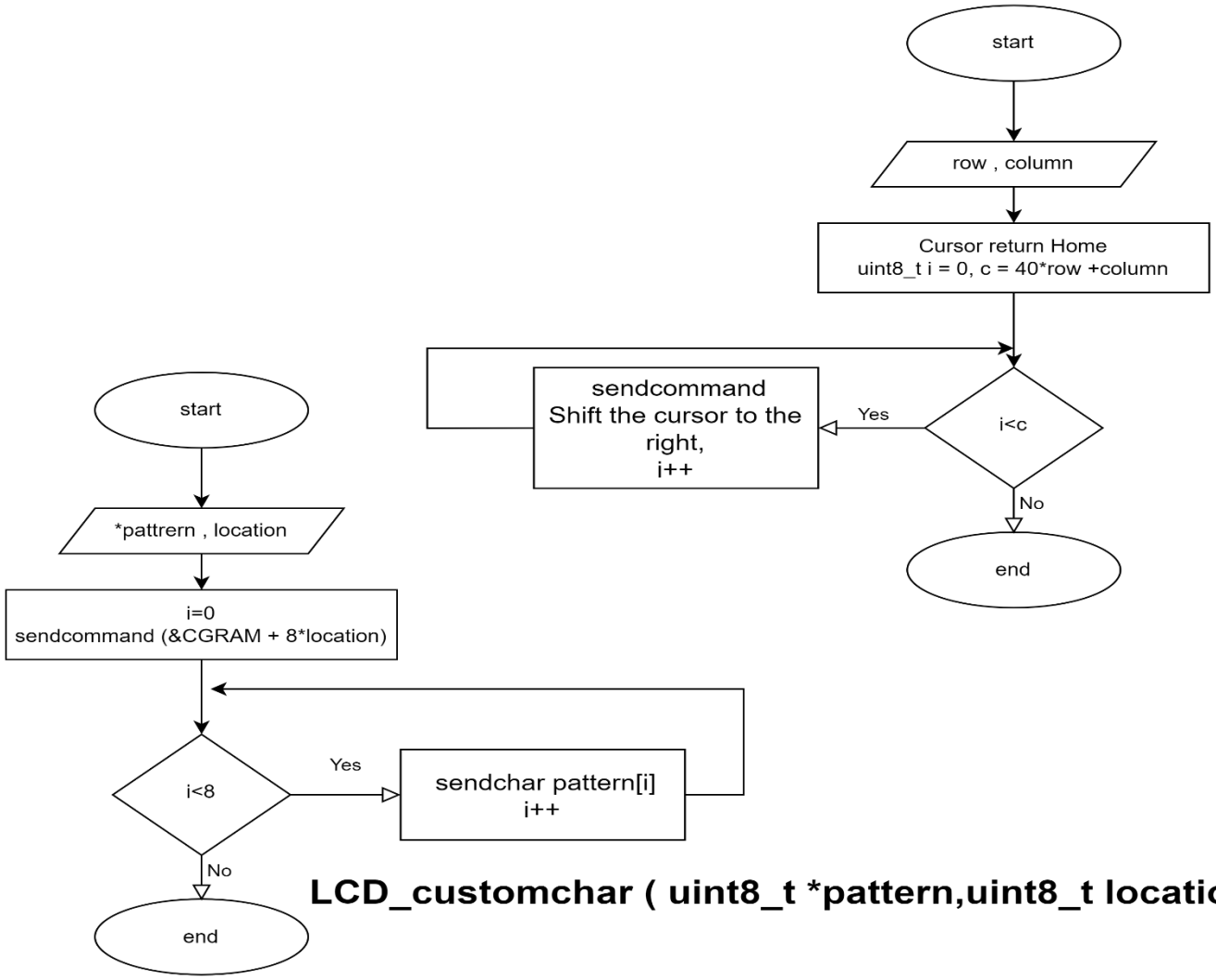
LCD_sendchar (uint8_t char_data)





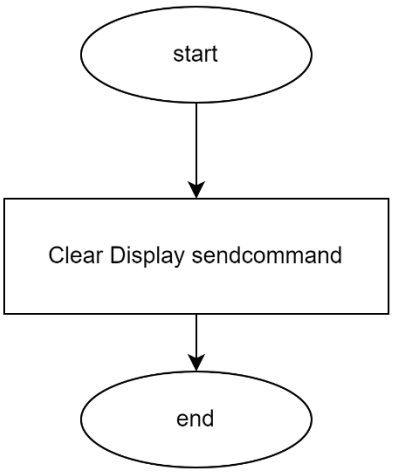
LCD_sendstring (uint8_t *str)

LCD_setcursor (uint8_t row,uint8_t column)

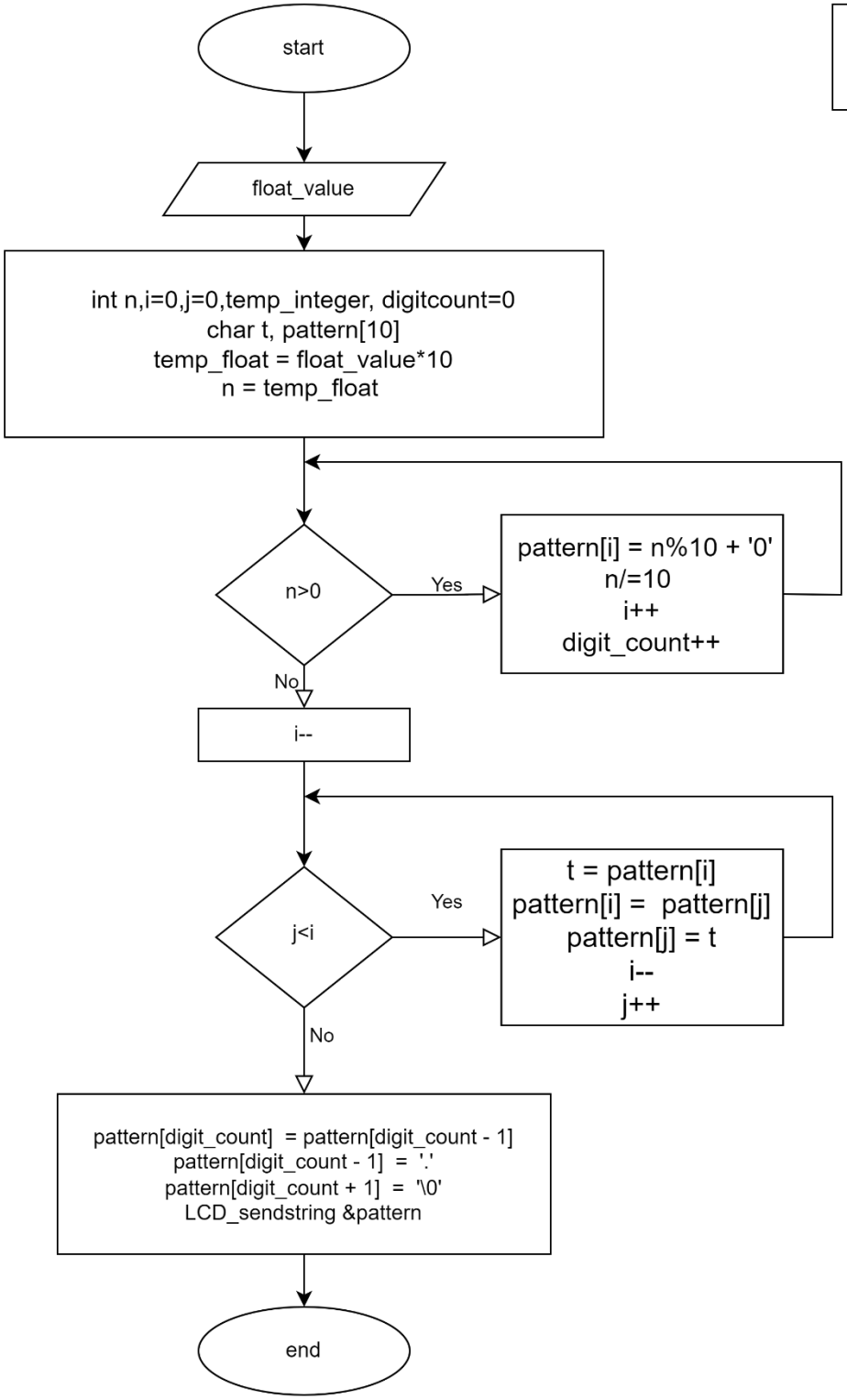


LCD_customchar (uint8_t *pattern,uint8_t location)

LCD_clear ()

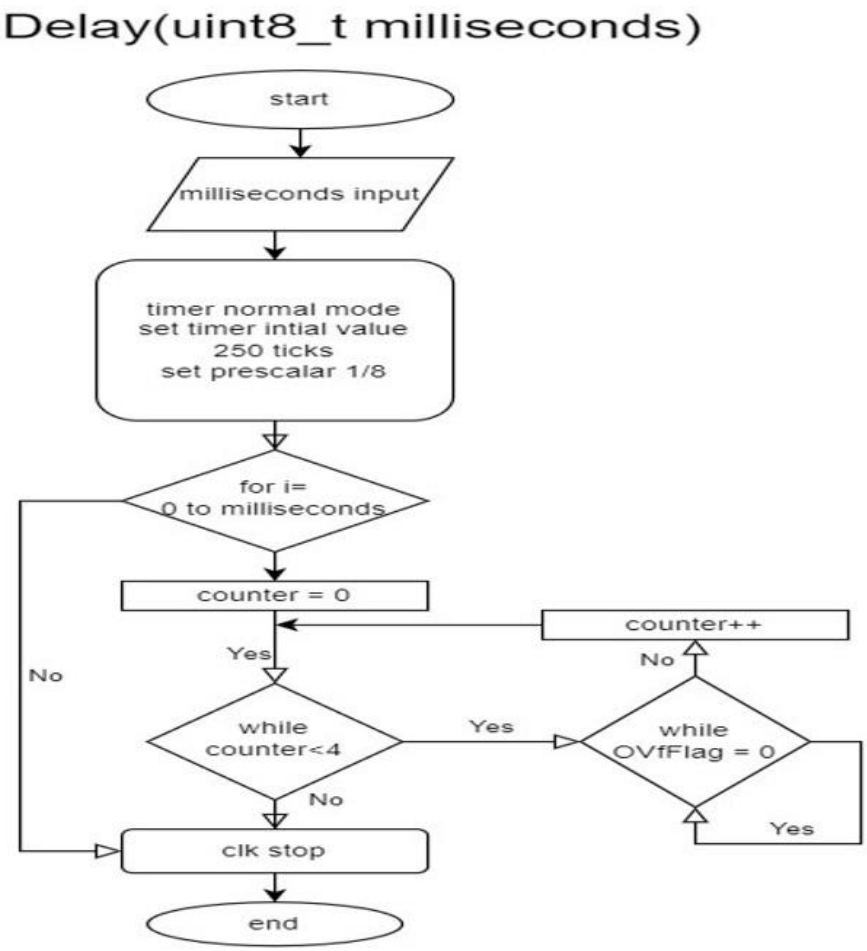


LCD_floattostring (f32_t float_value)

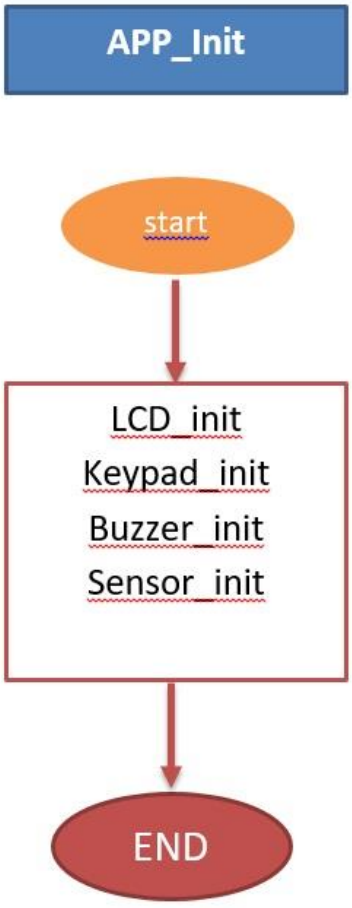


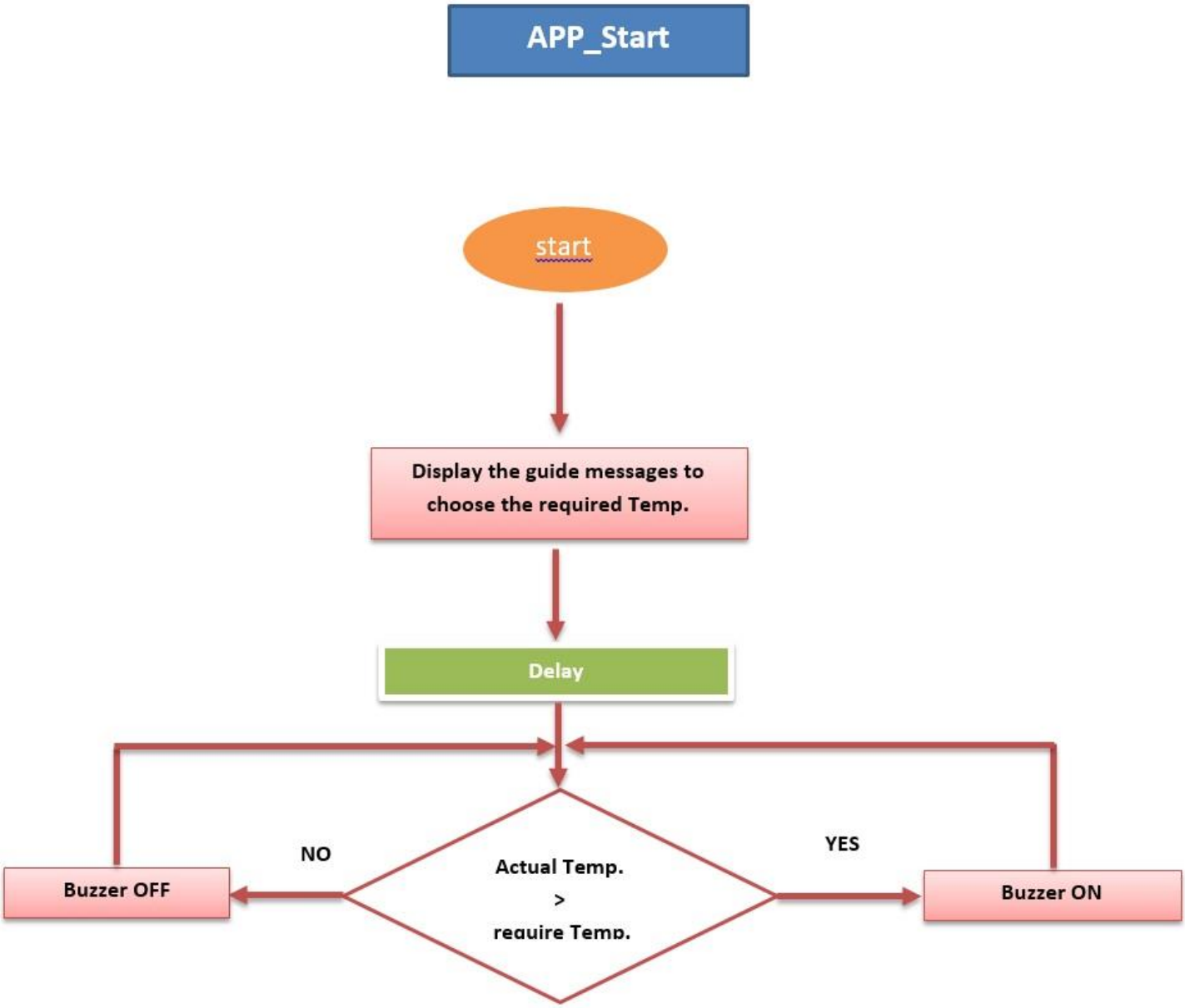
SERVICES FLOWCHARTS:

Delay APIS	void Delay(uint32_t milliseconds);



APPLICATION FLOWCHARTS:




APP_Start
start
Display the guide messages to choose the required Temp.
Delay
Actual Temp.
>
require Temp.
NO
YES
Buzzer OFF
Buzzer ON