Simple ATM Project

TEAM 8 (RETURN BRAIN)

ASSINED BY:

MOHAMED SAYAD

MOHAMED ABDELSALAM

EBRAHEM MOSTAFA

Table of contents

Contents

Sprints

Firstly: Project Description:

Hardware Requirements

ATM ECU

CARD ECU

KEYPAD

BUZZER

BUTTON

EEPROM

Software Requirements

1. *ATM MCU*
    1. *This MCU will handle transaction main flows*
    2. *After Reset*
        1. *Welcome message is displayed for 1s "Welcome to ATM"*
        2. *"Insert a Card" message is displayed in the first line*
        3. *No action can be taken further and all other input devices are blocked until a trigger signal came from the CARD ECU*
    3. *After a trigger signal is received from the CARD ECU*
        1. *"Enter Your PIN" message is displayed in the first line*
        2. *Waiting for the input from the keypad and type it in '****' format in the second line*
        3. *PIN is only four numeric characters*
        4. *Pressing the Enter/Zero button for 2s will initiate a communication between the ATM ECU and the CARD ECU to validate if the PIN is correct or not*
        5. *If the PIN is not correct, repeat for further 2 trials, and then if it is still wrong, sound the alarm and lock every input in the ATM, this blocking can be revealed by hard reset.*
        6. *If the PIN is correct, then "Enter Amount" message is displayed in the first line and wait for the amount to be entered from the keypad and appeared in the second line.*
        7. *Amount is a float string with max 4 integer digits and 2 decimal digits "0000.00"*
        8. *You can enter '0' when pressing Enter/Zero button for less than 2s*

9. After entering the amount to withdraw, several checks on the database are done to finalize the transaction
    1. Check if there is an account attached to this card
    2. Check if the card is blocked or not
    3. Check if the amount required exceeds the maximum daily limit or not
    4. Check for available amount
10. If one of the checks failed, a declined message will appear accordingly
    1. "This is a fraud card" – if the card PAN is not found – Alarm will be initiated
    2. "This card is stolen" – if the card is blocked– Alarm will be initiated
    3. "Maximum limit is exceeded" – if the required amount exceeds the maximum allowed limit
    4. "Insufficient fund" – if the balance is lower than the required amount
11. If all checks are passed then "Approved Transaction" message is displayed for 1s and the remaining balance is displayed for 1s "Remaining Balance: 0000.00"
12. After the checks are done and messages are displayed, display "Ejecting Card" message for 1s
13. Repeat from the after reset again

4. Data base

1. The data base will be hard coded array of structures for accounts that contains (PAN, Account State (blocked/running), and balance)
2. The maximum allowed limit will be hardcoded "5000.00"

2. CARD MCU

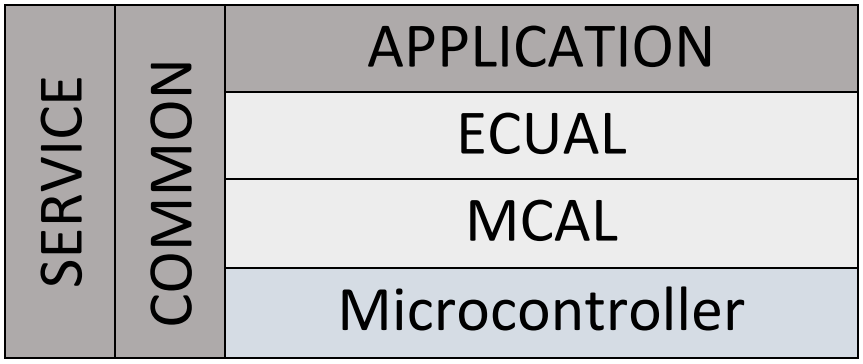
1. The CARD MCU has two modes of operations
    1. Programming Mode
        1. The CARD MCU will enter this mode after reset
        2. For the first time only the MCU will send the following messages to the terminal
            1. "Please Enter Card PAN:" and wait for the PAN
            2. "Please Enter New PIN:" and wait for the PIN
            3. "Please Confirm New PIN:" and wait for the PIN
        4. If PIN is matched, then change to user mode
        5. If PIN is not matched, not numeric, and exceeds 4 characters, then "Wrong PIN message is displayed and repeat from step

3. For any further after resets
    1. “Please press 1 for entering user mode and 2 for programming mode: ” message is sent to the terminal and wait for a valid response, only accepts 1 or 2
4. PAN is 16 to 19 length numeric string
5. PIN is 4 numeric digits
6. All data taken will be stored in the EEPROM

2. User Mode
1. The CARD MCU will enter this mode
    1. After completing the programming mode
    2. Or after choosing 2 in any further after resets
2. In this mode, the CARD ECU will send a trigger signal to the ATM ECU that will make the ATM initiate its flow

Secondly: Layered architecture:

1- Microcontroller
2- MCAL
3- ECUAL
4- COMMON
5- SERVICE
6- Application



Thirdly: System modules:

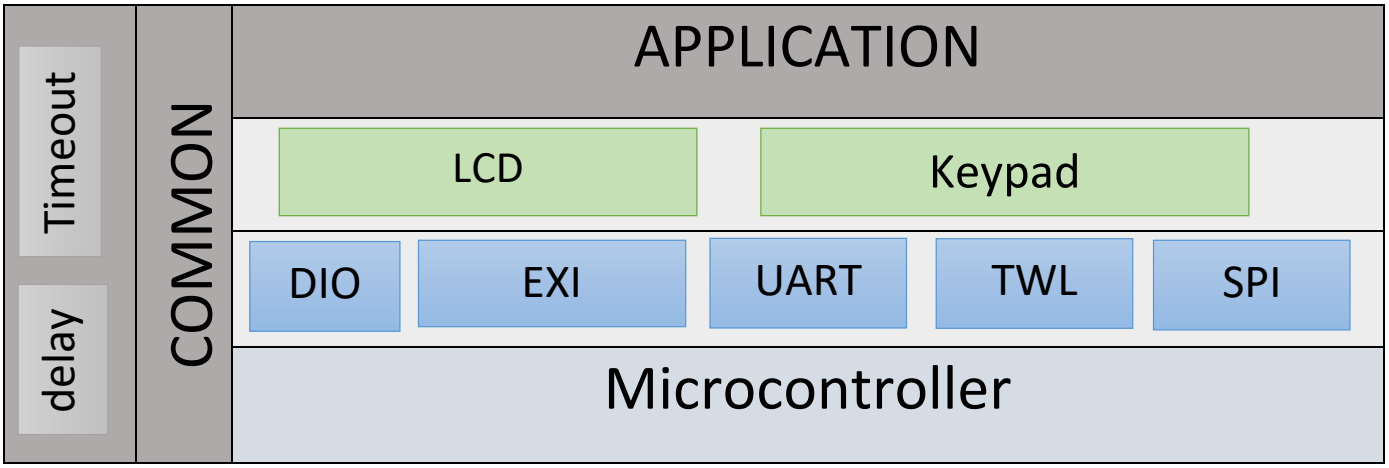
- **CARD MCU:**

1- Specify system modules/drivers:

- DIO, EXTERNAL INTERRUPT,UART,TWL,SPI
- LCD, KEYPAD
- DELAY, TIME OUT
- APPLICATION

2- Assign each module to its related layer:

- By drawing



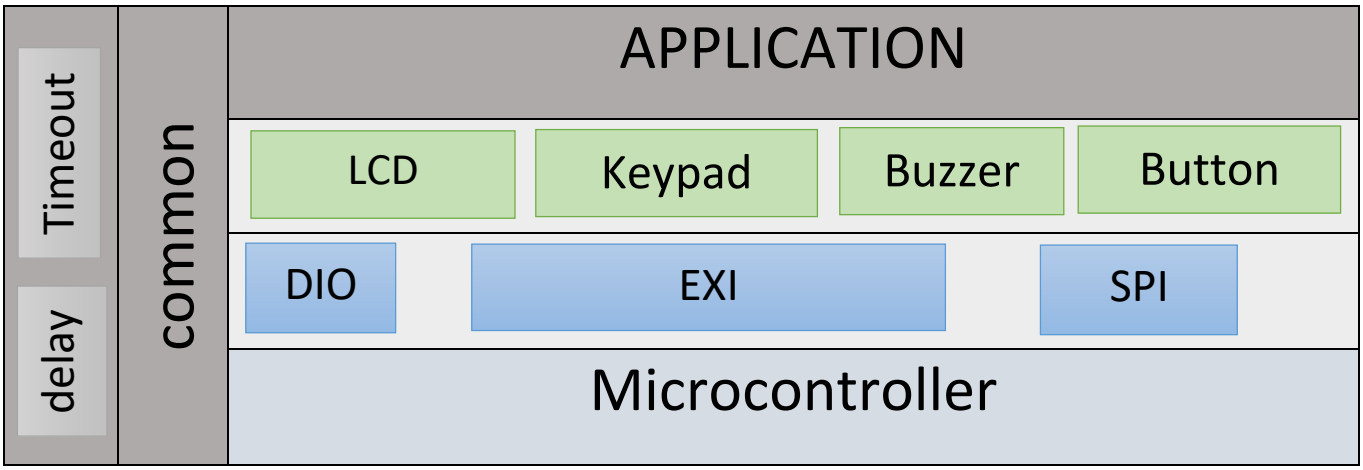
- **ATM MCU**

1- Specify system modules/drivers:

- DIO, EXTERNAL INTERRUPT,SPI
- LCD, KEYPAD
- DELAY, TIME OUT
- 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);

void DIO_TogglePin(PIn_name pin);
```

Sprints

TIMER **APIs:**

```
void TIMER_init (u8 Mode,u8 intial_value);

void TIMER_start (u8 prescaler_value);

void TIMER_set(u8 intial_value);

void TIMER_getStatus(u8 *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);
```

EXTERNAL INTERRUPT **APIs:**

```
void EXI_Enable (ExInterruptSource_type Interrupt);

void EXI_Disable (ExInterruptSource_type Interrupt);

void EXI_Trigger(ExInterruptSource_type Interrupt,TriggerEdge_type trigger);

void EXI_SetCallBack(ExInterruptSource_type Interrupt,void(*pf)(void));
```

UART **APIs:**

```
void uart_init(Uart_BaudRate_en BaudRate,Uart_Speed_en Speed,Uart_Mode_en
mode,Uart_Parity_en parity,Uart_StopBit_en stopbit,Uart_DataBit_en
databit,Uart_Enable_en enable);

void uart_transmit(u8 data);

void uart_transmitNoBlock(u8 data);
```

```
void uart_transmitComPlete_InterruptEnable(void);

void uart_transmitComPlete_InterruptDisable(void);

void uart_transmitComPlete_InterruptSetCallback(void(*fptr)(void));

u8 uart_reciever(void);

u8 uart_recieverNoBlock(void);

void uart_recieveComPlete_InterruptEnable(void);

void uart_recieveComPlete_InterruptDisable(void);

void uart_recieveComPlete_InterruptSetCallback(void(*fptr)(void));
```

UART Service **APIs:**

```
void uart_transmitString(u8*str);

void uart_transmitString_Interrupt(u8*str);

void uart_transmitNumber_MCU(u32 num);

void uart_transmitNumber_MCU_Interrupt(u32* num);

void uart_recieverString(u8*str);

void uart_recieverString_Interrupt(u8*str);

void uart_recieveNumber_MCU(u32* num);

void uart_recieveNumber_MCU_Interrupt(u32* num);
```

SPI **APIs:**

TWL **APIs:**

LCD **APIs:**

```
void LCD_init(void);

void LCD_sendcommand (u8 cmnd);

void LCD_sendChar (u8 char_data);

void LCD_sendString (u8 *str);

void LCD_createCustomCharacter (u8 *pattern , u8 location );

void LCD_clear(void);

void LCD_floattostring (f32 float_value);

void LCD_setCursor (u8 row , u8 column);

void LCD_WriteNumber(s32 num);
```

KEYPAD **APIs:**

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

Keypad_Status_en KEYPAD_GetNum_time(u8 timeout, u8* key);
```

BUTTON **APIs:**

```
void button_init(PIn_name pin);

Button_Status Button_Check(u8 Button);

Button_Status_Time Button_Check_Time(u8 Button);
```

Sprints

BUZZER APIs:

```
void bazz_init(PIn_name pin_num);

void bazz_ON(PIn_name pin_num);

void bazz_OFF(PIn_name pin_num);
```

DELAY APIs:

```
void Delay_ms(u32 milliseconds);

void delay_us(u32 microseconds);
```

TIMEOUT APIs:

```
void TIME_out (u32 milliseconds);

static void TIMER_out_Stop (void);

static void TIME_out_init (void);

static void TIMER_ISR(void);

static void Timer0_Ovf_CALLBACK (void (*copyFuncptr) (void));
```

APPLICATION APIs:

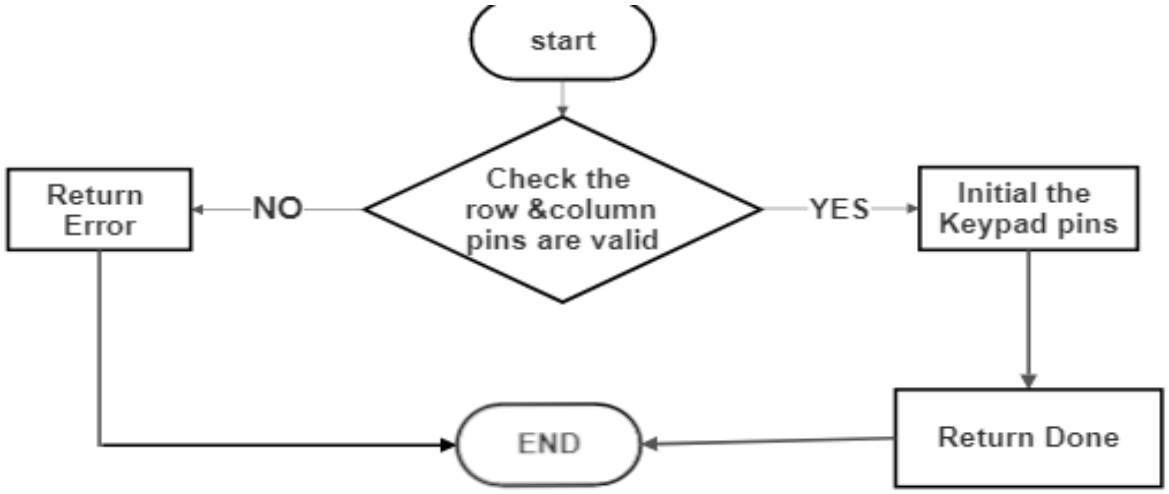
```
void App_init(void);

void App_start(void);
```

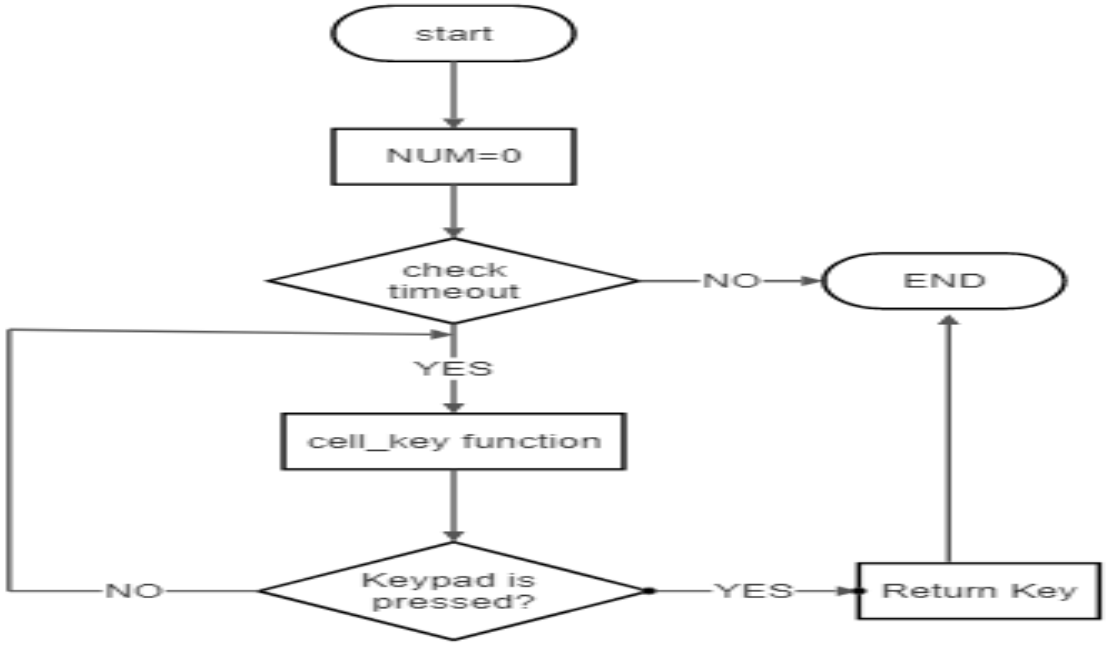
Fifthly: Flowcharts APIs:

ECUAL FLOWCHARTS:

- Keypad



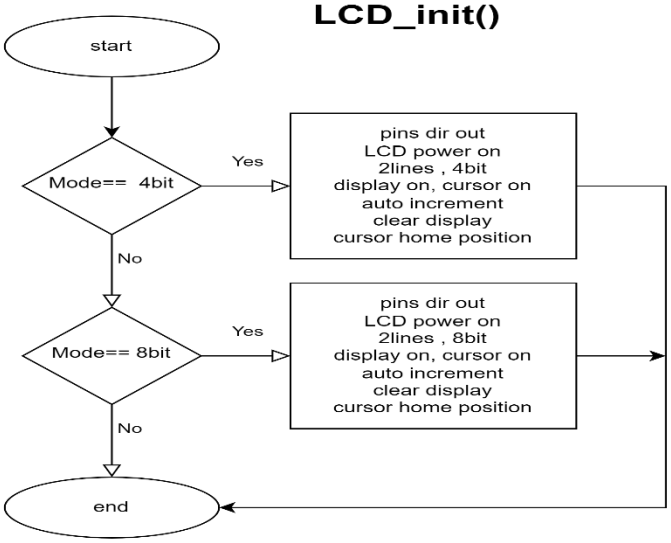
Keypad_Status_en KEYPAD_Init(Pin_name First_Output,Pin_name Firs_Input);



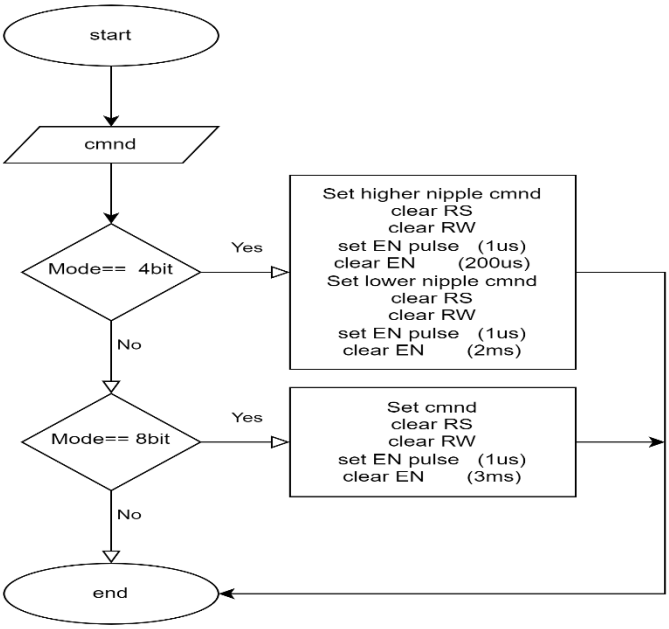
Keypad_Status_en KEYPAD_GetNum_time(u8 timeout, u8* key);

- LCD

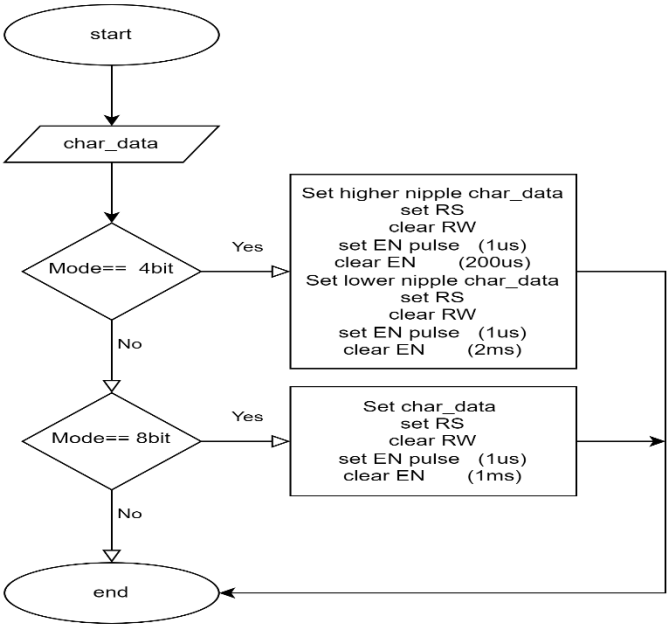
LCD_init()



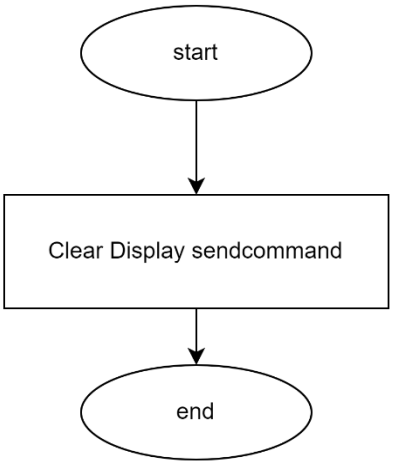
LCD_sendcommand(uint8_t cmnd)



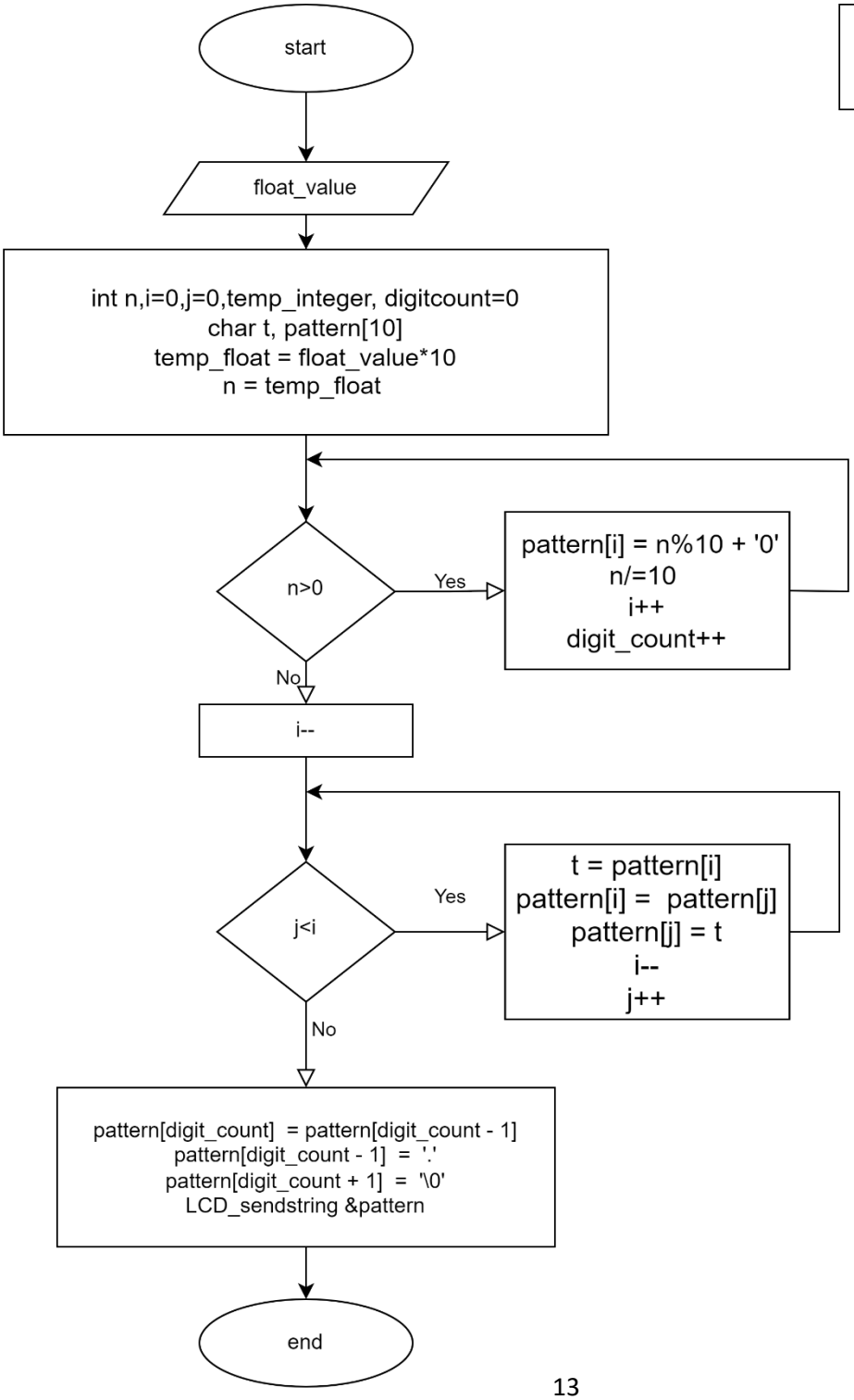
LCD_sendchar (uint8_t char_data)

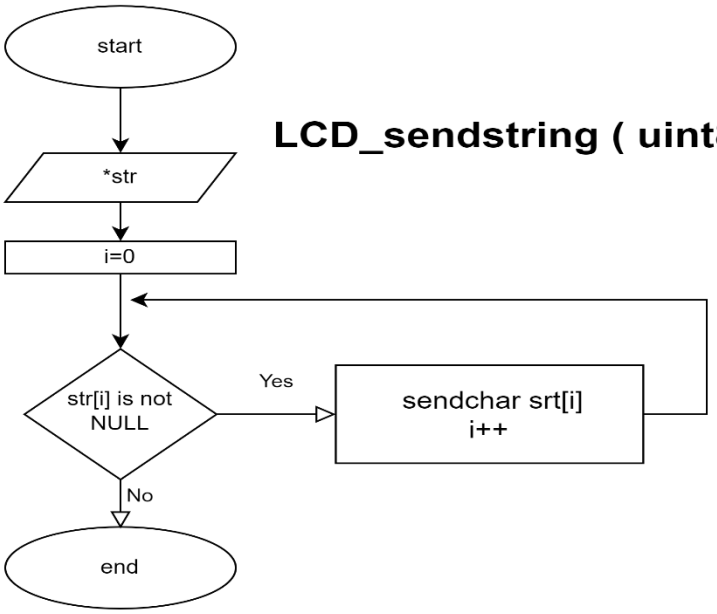


LCD_clear ()



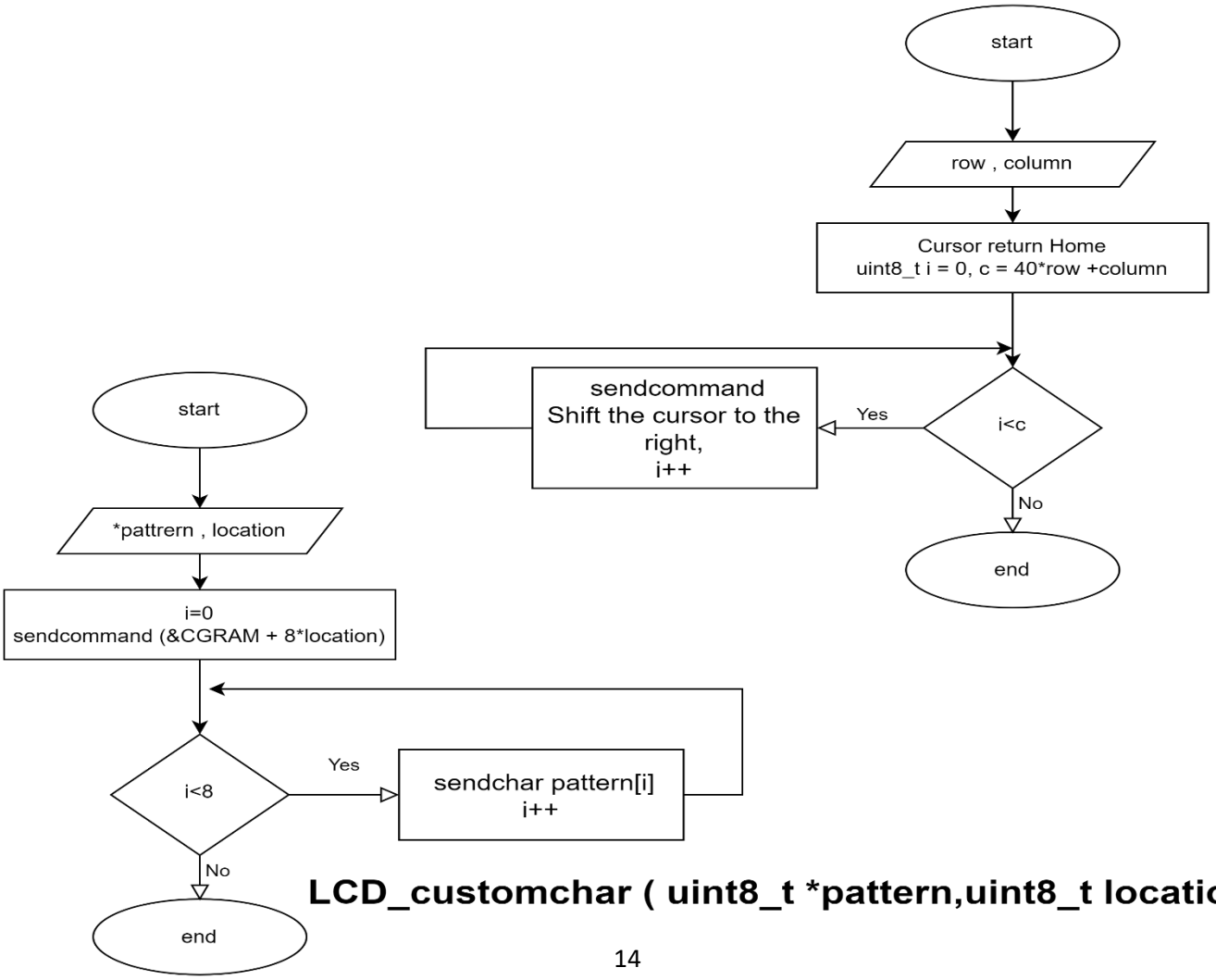
LCD_floattostring (f32_t float_value)





LCD_sendstring (uint8_t *str)

LCD_setcursor (uint8_t row,uint8_t column)



LCD_customchar (uint8_t *pattern,uint8_t location)