In the name of Allah

University of Khartoum

Electrical and Electronics Engineering

Microcontroller Project

SMS Editing System

Group:

1- Altahir Mansour 144017

2- Abduljabbar Ahmed 144041

3- Mohammed Omer 144068

4- Monzer Hashim 144097

5- Megat Ali 144101

Supervised By:

Ustaza Sara Omer

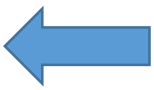
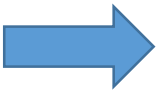
Functional Requirements :

- Our project allows us to write messages, modify the message by:
  - Inserting within the messages letters (capital and small), numbers, and spaces.
  - Deleting characters anywhere within the message (backspace and delete).
  - Scrolling around the message (left, right, up, and down)
  - After finishing the message, send the message and clear the screen.
  - Maximum length of message is 80 characters (16*5).

Technical requirements:

- We use keypad (4*4) to write the characters. 16 keys divided as follows:
  - 9 of them for characters and space
  - 4 keys for scrolling(up, down, left, and right)
  - 1 key for backspace and delete.

- 1 key for choosing between capital and small letters
- 1 key for sending the message and clearing screen after finishing the message.
  - Note: when using small letters backspace is activated. And when using capital letters delete is activated.

a b c 1	d e f 2	g h i 3	Del / BKSP
j k l 4	m n o 5	p q r 6	CapsLock
s t u 7	v w x 8	y z sp 9	
	send		

- We use LCD (Liquid Crystal Display) to print the messages.
- Atmega32 (microcontroller) is used.

Algorithm:

initializeLCD();

initilizeUSART();

A:

waitForKeypadReleasing();

getPressedKey();

B:

char or command?

if (pressed key is char){

C:

print();

startTimer();

getNextKey();

if(same key){

checkTimer();

```
if(time is done){
jump to B;
}
Else{
deleteLastChar();
GetCharFromNextArray();
Jump to C;
}
else{
Jump to B;
}
}
   else (when pressed key is command){
left: decrease x;
right: increase x;
up: decrease y;
down: increase y;
//limits are considered
CapsLock: toggle CAP_small value;
Del/BKSP:
If(CapsLock = small){
Back_space();
```

}

Else{

Delete();

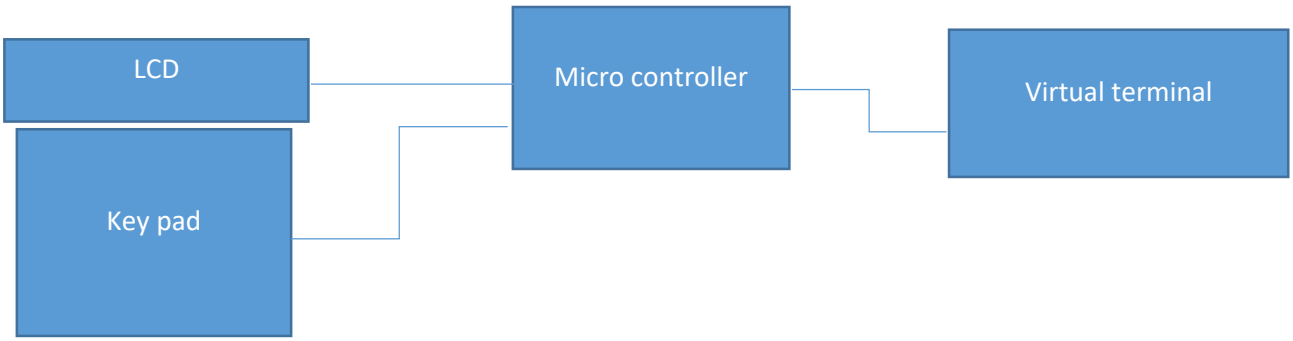
}

Send: send message by serial port(USART) to the virtual terminal;

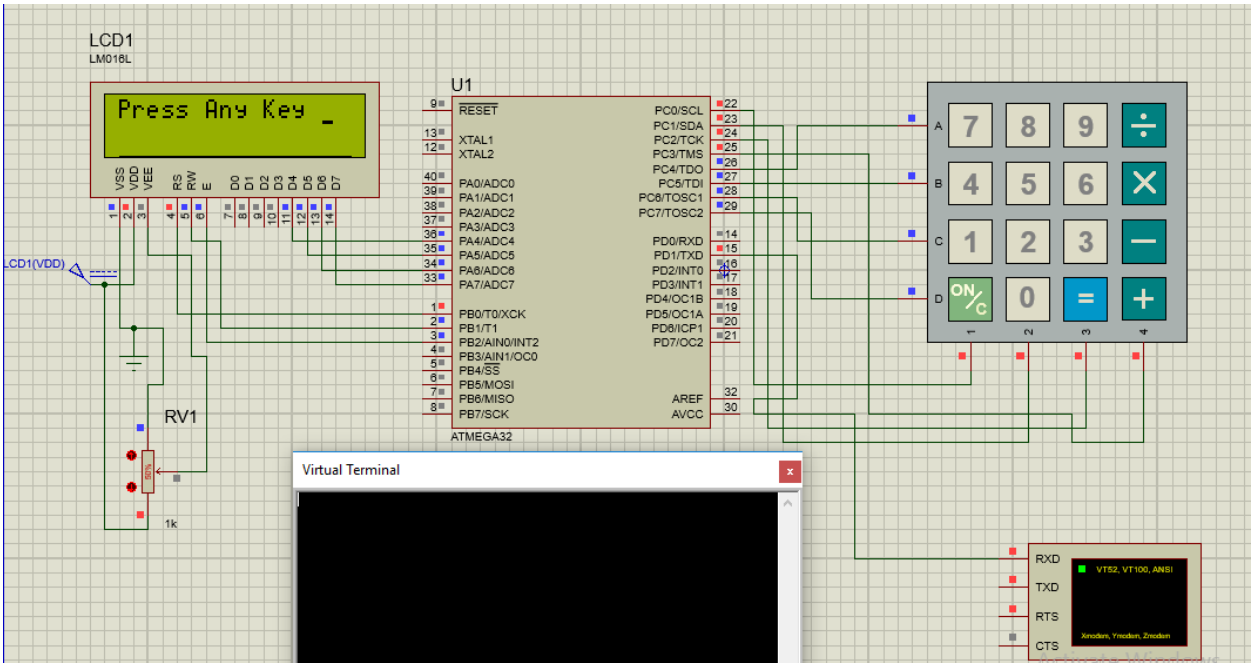
}

Design:

***block diagram:-**



***connection(detailed) diagram:-**



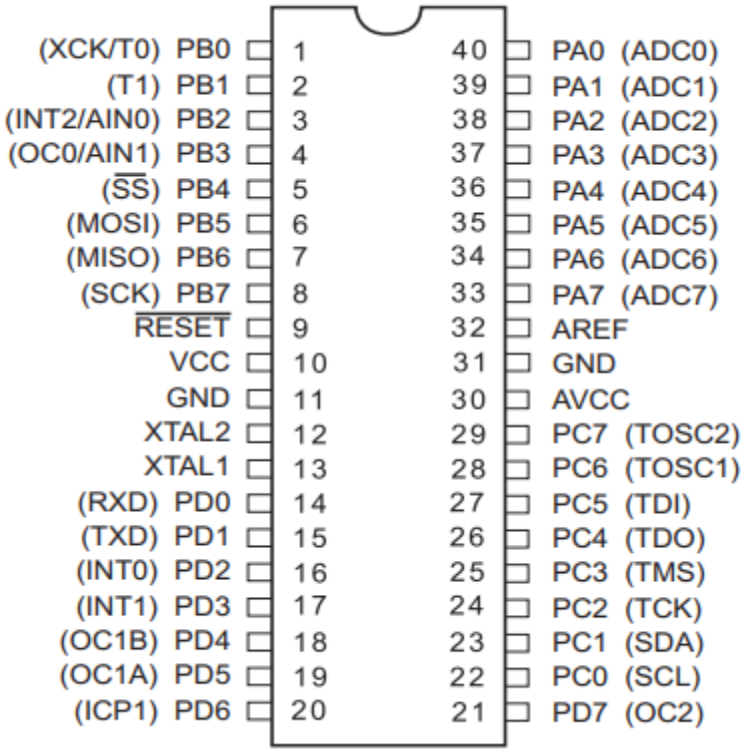
- Keypad is connected to port C (the four high bits for rows and other low four bit for columns).

-LCD is connected to micro controller through 7 bits. Four of them for the data bus and 3 of them for (register select), (read or write), and (enable ”falling edge”).

-Virtual terminal is connected to the microcontroller (TX of microcontroller to RX of VT).

Implementation outlines:

- Oscillation frequency (f) : 1MHz.
- Prescalar is clk/8 → 125KHz
- 1 clock cycle = 1/(f after prescaling) = 8 us.
- Max delay between 2 presses of keys to be considered as next character is:2^16 clock cycles = 65,536 clock cycles = 0.524288 second.
- Baud rate = 2000 bps (assumed and virtual terminal configured as that).
- UBRR value = f/(16*baudrate) -1 = 30.25 ~ 30 with error = 0.25/30.25 = 0.83%
- Atmega32:


(XCK/T0) PB0 1
(T1) PB1 2
(INT2/AIN0) PB2 3
(OC0/AIN1) PB3 4
(SS) PB4 5
(MOSI) PB5 6
(MISO) PB6 7
(SCK) PB7 8
RESET 9
VCC 10
GND 11
XTAL2 12
XTAL1 13
(RXD) PD0 14
(TXD) PD1 15
(INT0) PD2 16
(INT1) PD3 17
(OC1B) PD4 18
(OC1A) PD5 19
(ICP1) PD6 20
40 PA0 (ADC0)
39 PA1 (ADC1)
38 PA2 (ADC2)
37 PA3 (ADC3)
36 PA4 (ADC4)
35 PA5 (ADC5)
34 PA6 (ADC6)
33 PA7 (ADC7)
32 AREF
31 GND
30 AVCC
29 PC7 (TOSC2)
28 PC6 (TOSC1)
27 PC5 (TDI)
26 PC4 (TDO)
25 PC3 (TMS)
24 PC2 (TCK)
23 PC1 (SDA)
22 PC0 (SCL)
21 PD7 (OC2)

LCD:

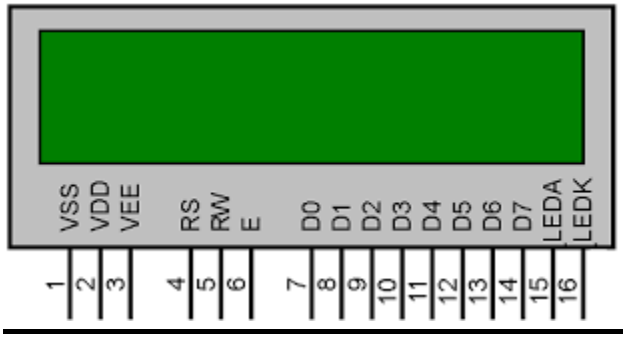
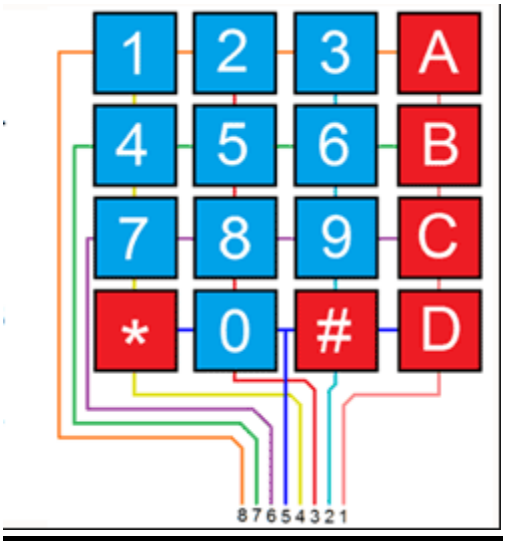

VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
LEDA
LEDK
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

Table 12-2: LCD Command Codes

Code (Hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning of 1st line
C0	Force cursor to beginning of 2nd line
28	2 lines and 5 × 7 matrix (D4–D7, 4-bit)
38	2 lines and 5 × 7 matrix (D0–D7, 8-bit)

KeyPad:



Code:

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>


#define F_CPU 1000000 //1MHZ
#define KEY_PRT  PORTC
#define KEY_DDR       DDRC
#define KEY_PIN       PINC
#define timerStart 0b00000010 //value for TCCR0 (CLK/8)
#define LCD_MAX 15 //16-1
#define MAX_lines 4// 0 1 2 3 4 = 5 lines
#define baudrate 2000
#define X (unsigned char)(F_CPU/(16*baudrate)-1)

unsigned char TimeDone = 0;
unsigned char CAP_small=0; // 0 = small , 1 = CAP
unsigned char x=0,y=0,y_msg=0;
unsigned char start=0;
ISR(TIMER1_OVF_vect){
      TCCR1B = 0; //STOP
```

```
	TCNT1 = 0;
	TimeDone = 1;
}

unsigned char keypad[4][4] = {	{'A','D','G',0},
{'J','M','P',0},
{'S','V','Y',0},
{0,0,0,0}};

unsigned char keypad2[4][4] =  { {'B','E','H',0},
{'K','N','Q',0},
{'T','W','Z',0},
{0,0,0,0}};

unsigned char keypad3[4][4] ={	{'C','F','I',0},
{'L','O','R',0},
{'U','X',' ',0},
{0,0,0,0}};

unsigned char keypad4[4][4] = {  {'1','2','3',0},
{'4','5','6',0},
{'7','8','9',0},
{0,0,0,0}};

unsigned char Message[5][16]= {{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0} ,
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}        }; //80 chars


unsigned char colloc, rowloc;

unsigned char keyfind()
{
	while(1)
	{
		KEY_DDR = 0xF0;		/* set port direction as input-
output */
		KEY_PRT = 0xFF;

		do
		{
			KEY_PRT &= 0x0F;	/* mask PORT for column read
only */
			asm("NOP");
```

```
                colloc = (KEY_PIN & 0x0F); /* read status of column */
          }while(colloc != 0x0F);

          do
          {
                do
                {
                      _delay_ms(20);             /* 20ms key debounce
time */
                      colloc = (KEY_PIN & 0x0F); /* read status of
column */
                      }while(colloc == 0x0F);        /* check for any
key press */

                      _delay_ms (40);              /* 40 ms key
debounce time */
                      colloc = (KEY_PIN & 0x0F);
                }while(colloc == 0x0F);

                /* now check for rows */
                KEY_PRT = 0xEF; //1110            /* check for pressed
key in 1st row */
                asm("NOP");
                colloc = (KEY_PIN & 0x0F);
                if(colloc != 0x0F)
                {
                      rowloc = 0;
                      break;
                }

                KEY_PRT = 0xDF;       /* check for pressed key in 2nd
row */
                asm("NOP");
                colloc = (KEY_PIN & 0x0F);
                if(colloc != 0x0F)
                {
                      rowloc = 1;
                      break;
                }

                KEY_PRT = 0xBF;       /* check for pressed key in 3rd
row */
                asm("NOP");
                colloc = (KEY_PIN & 0x0F);
                if(colloc != 0x0F)
                {
```

```
                        rowloc = 2;
                        break;
                }

                KEY_PRT = 0x7F;       /* check for pressed key in 4th
row */
                asm("NOP");
                colloc = (KEY_PIN & 0x0F);
                if(colloc != 0x0F)
                {
                        rowloc = 3;
                        break;
                }
        }

        if(colloc == 0x0E){
        colloc =0;
        return(keypad[rowloc][colloc]);}
        else if(colloc == 0x0D){
        colloc =1;
        return(keypad[rowloc][colloc]);}
        else if(colloc == 0x0B){
        colloc =2;
        return(keypad[rowloc][colloc]);}
        else
        {
        colloc =3;
        return(keypad[rowloc][colloc]);}
    }



void writeCommand(unsigned char cd){
    PORTB &= 0xFC ; //clear RS and R/W 11111100
    PORTA = cd & 0xF0;
    PORTB |= (1<<2); //E = high 00000100
    _delay_us(1);
    PORTB &= ~(1<<2); //E = low 11111011
    //HIGH TO LOW PULSE
    _delay_us(100);

    //swap nibbles
    cd = (cd<<4)|(cd>>4);

    PORTA = cd & 0xF0;
    PORTB |= (1<<2); //E = high
```

```
	_delay_us(1);
	PORTB &= ~(1<<2); //E = low
	//HIGH TO LOW PULSE
	_delay_us(100);
}


void writeData(unsigned char cd){
	setXY(x,y);

	PORTB &= 0xFD ; //clear R/W
	PORTB |= 0x01 ; //set RS
	if(CAP_small==0){
		if(cd>='A' && cd<='Z'){ //all CAP letters
			cd = cd+0x20; // CAPITAL to small
		}
	}
	Message[y_msg][x]=cd;
	PORTA = cd & 0xF0;
	PORTB |= (1<<2); //E = high
	_delay_us(1);
	PORTB &= ~(1<<2); //E = low
	//HIGH TO LOW PULSE
	_delay_us(100);
	//swap nibbles
	cd = (cd<<4)|(cd>>4);

	PORTA = cd & 0xF0;
	PORTB |= (1<<2); //E = high
	_delay_us(1);
	PORTB &= ~(1<<2); //E = low
	//HIGH TO LOW PULSE
	_delay_us(100);

	x++;
	if(x>15){
		x=0;
		if(y==0){
			y++;
			y_msg++;
	}
	else{
		y_msg++;
		start++;
		printMsg();
	}
```

```
}
}

void setXY(unsigned char xx,unsigned char yy){
      writeCommand(0x80+0x40*y+x);
}
void left(){
      if(x>0){
      x--;
      }
      else{
      if(y>0){
      x=15;
      y--;
      y_msg--;
      }
      else{
            if(y_msg>0){
            start--;
            y_msg--;
            x=15;
            printMsg();
            }
      }
      }
    setXY(x,y);
}

void right(){
      if(x<LCD_MAX){
            x++;
      }
      else{ // x=15
            if(y==0){
                  x=0;
                  y++;
            }
            else{
                  if(y_msg<4){
                  x=0;
                  start++;
                  y_msg++;
                  printMsg();
                  }
            }
      }
```

```
	setXY(x,y);
}
void up(){
	if(y_msg>start){//y==1
		y--;
		y_msg--;
		setXY(x,y);
	}
	else { //y_msg == start
		if(start>0){
		y_msg--;
		start--;
		printMsg();
		}
	}
}

void down(){
	if(y_msg==start){
	y++;
	y_msg++;
	setXY(x,y);
	}
	else{
		if(y_msg<4){
			y_msg++;
			start++;
			printMsg();
		}
	}
}

void BKSP(){
	unsigned char i,j;
	if(x!=0){
	for(j=x-1;j<15;j++){
		Message[y_msg][j]=Message[y_msg][j+1];
	}
	}
	else{

		Message[y_msg-1][15]=Message[y_msg][0] ;
	}
	for(i=y_msg+1;i<5;i++){
		Message[i-1][15] = Message[i][0];
		for(j=0;j<15;j++){
```

```
                Message[i][j]=Message[i][j+1];
            }
        }

}

void shiftLCD(){
      //shifting
      unsigned char i,j;


            for(i=4;i>y_msg;i--){
            for(j=15;j>0;j--){
                  Message[i][j]=Message[i][j-1];
            }
            Message[i][0] = Message[i-1][15];
      }
      for(j=15;j>x;j--){
            Message[y_msg][j]=Message[y_msg][j-1];
      }

}
void printMsg(){
writeCommand(0x80);
unsigned char i,j,a;
PORTB &= 0xFD ; //clear R/W
PORTB |= 0x01 ; //set RS

for(i=start;i< (start+2);i++){
      for(j=0;j<16;j++){
a=Message[i][j];
PORTA =  a & 0xF0;
PORTB |= (1<<2); //E = high
_delay_us(1);
PORTB &= ~(1<<2); //E = low
//HIGH TO LOW PULSE
_delay_us(100);
//swap nibbles
a = (a<<4)|(a>>4);

PORTA = a & 0xF0;
PORTB |= (1<<2); //E = high
_delay_us(1);
PORTB &= ~(1<<2); //E = low
//HIGH TO LOW PULSE
_delay_us(100);
```

```
      }
      writeCommand(0xC0);
    PORTB |= 0x01 ; //set RS
}
setXY(x,y);
}

void writeString(unsigned char* str,unsigned char h){
unsigned char cd,i;
for(i=0;i<h;i++){
      cd = str[i];
      PORTB &= 0xFD ; //clear R/W
      PORTB |= 0x01 ; //set RS
      PORTA = cd & 0xF0;
      PORTB |= (1<<2); //E = high
      _delay_us(1);
      PORTB &= ~(1<<2); //E = low
      //HIGH TO LOW PULSE
      _delay_us(100);
      //swap nibbles
      cd = (cd<<4)|(cd>>4);

      PORTA = cd & 0xF0;
      PORTB |= (1<<2); //E = high
      _delay_us(1);
      PORTB &= ~(1<<2); //E = low
      //HIGH TO LOW PULSE
      _delay_us(100);
}
}




void usart_send(unsigned char a){
      while((UCSRA & (1<<UDRE)) ==0);
      UDR = a;
}


int main(void)
{
      //ports as outputs
      DDRA |= 0xF0;
      DDRB |= 0x07;
```

```
UCSRB|=(1<<TXEN);
UCSRC=(1<<UCSZ1)|(1<<UCSZ0)|(1<<URSEL);
UBRRL=X & 0xFF;

TIMSK |= (1<<TOIE1);

unsigned char x1,x2;
//LCD initialization
writeCommand(0x33);
writeCommand(0x32);
writeCommand(0x28);
writeCommand(0x0E);
writeCommand(0x01);
_delay_ms(2);



//welcome:
unsigned char welcome[]="Press Any Key";
writeString(welcome,14);


x1=keyfind();
writeCommand(0x01); //clear screen
_delay_ms(2);
goto print;
while(1)
{
      read:
     x1=keyfind();
      print:
      if(rowloc < 3 & colloc < 3 )
      {
      TimeDone = 0;
      TCNT1=0;
      TCCR1B=timerStart;
      sei();
      shiftLCD();
      printMsg();
      writeData(x1);
      x2=keyfind();
      if(x2 != x1){
            x1=x2;
            goto print;
      }
      else if(TimeDone ==1){
```

```
                goto print;//print x1 again
        }
else {
                left(); //shift cursor left
                writeData(keypad2[rowloc][colloc]);
                TCNT1=0;
                TCCR1B=timerStart;
                TimeDone=0;
                x2=keyfind();
                if(x2 != x1){
                        x1=x2;
                        goto print;
                }
                else if(TimeDone ==1){
                        goto print;//print x1 again
                }
                else {
                        left(); //shift cursor left
                        writeData(keypad3[rowloc][colloc]);
                TCNT1=0;
                TCCR1B=timerStart;
                TimeDone=0;
                x2=keyfind();
                if(x2 != x1){
                        x1=x2;
                        goto print;
                }
                else if(TimeDone ==1){
                        goto print;//print x1 again
                }
                else {
                        left(); //shift cursor left
                        writeData(keypad4[rowloc][colloc]);
                        TCNT1=0;
                        TCCR1B=timerStart;
                        TimeDone=0;
                        x2=keyfind();
                        if(x2 != x1){
                                x1=x2;
                                goto print;
                        }
                        else if(TimeDone ==1){
                                goto print;//print x1 again
                        }
                        else {
                                BKSP();
```

```
                              left();
                              goto print; //again
                }

                        }
                        }
        }
}
else{

         if(rowloc==3 && colloc==0){
                left();
                goto read;
        }
        else if(rowloc==3 && colloc==2){
                right();
                goto read;
        }
        else if(rowloc==2 && colloc==3){ //up
              up();
                goto read;
        }
        else if(rowloc==3 && colloc==3){ //down
                down();
                goto read;
        }
        else if(rowloc==1 && colloc==3){
                CAP_small^=1; //toggle , if small --> CAP and if CAP -->
small
                goto read;
        }
        else if(rowloc==0 && colloc==3){
                if(CAP_small==1){
                        right();
                }
                BKSP();
                printMsg();
                left();
        }
        else if(rowloc==3 && colloc == 1){
        start=0;
        unsigned char p,q;
        for(p=0;p<5;p++){
                for(q=0;q<16;q++){
                        usart_send(Message[p][q]);
                        Message[p][q]=0;
```

```
            }
        }
        usart_send('\r');


        writeCommand(0x01); //clear lcd
        _delay_ms(2);
        writeCommand(0x80);
        writeString("DONE !",6);
        _delay_ms(1000);
        writeCommand(0x01);
        _delay_ms(2);
        x=0;y=0;
        writeCommand(0x80);

        }

}
        }
        return 0;
}
```