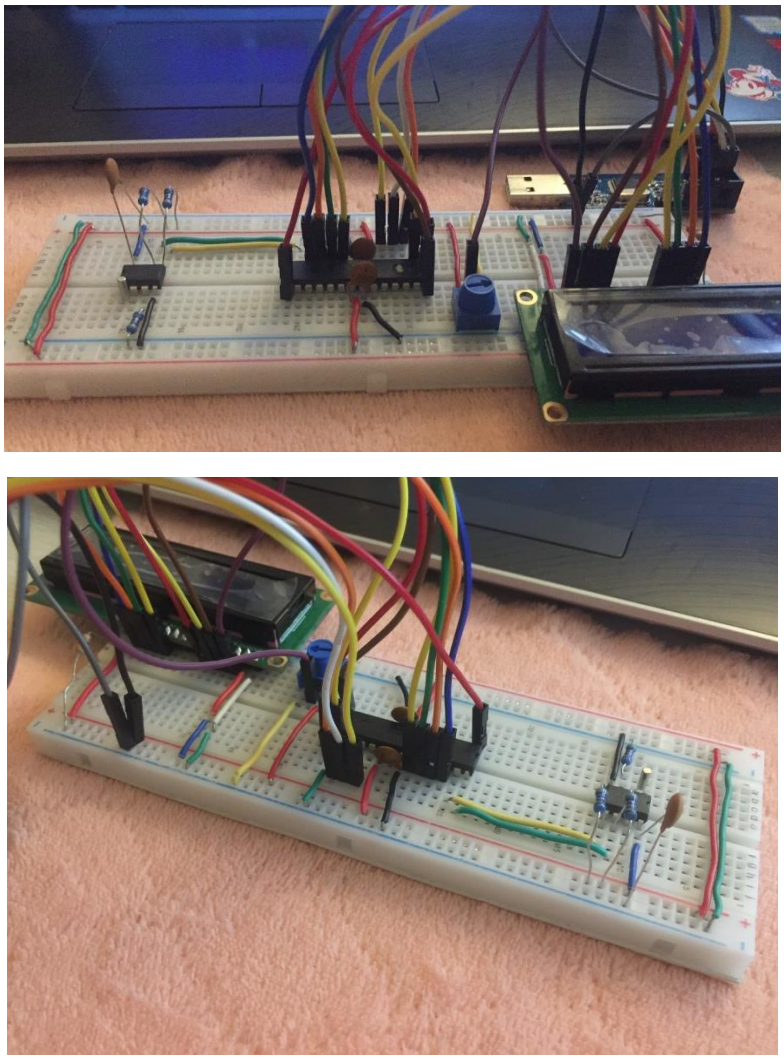
Lab 6 : AVR I^{2}C Programming

Circuit in Lab 6



รูปที่ 1 : Circuit in Lab 6

Code in Lab 6

```
#define F_CPU 8000000UL

#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>
#include <string.h>

//define data and cmd port, RS, RW, E ports
#define LCD_DATA_DDR DDRC
#define LCD_DATA_PORT PORTC
#define LCD_CMD_DDR DDRB
```

```
#define LCD_CMD_PORT PORTB
#define RS PORTB0
#define E PORTB1
#define RW PORTB2

void commit_data()
{
    //pulse enable
    LCD_CMD_PORT |= (1 << E);
    _delay_us(1000);
    LCD_CMD_PORT &= ~(1 << E);
    _delay_us(1000);
}

void send_data(uint8_t data)
{
    LCD_DATA_PORT = (data >> 4);
    commit_data();
    LCD_DATA_PORT = (data & 0x0F);
    commit_data();
}

void send_lcd_command(uint8_t command)
{
  //set to cmd mode, write mode
  LCD_CMD_PORT &= ~(1 << RS);

  //send command
  send_data(command);
}

void send_lcd_data(uint8_t data)
{
  //set to data mode, write mode
  LCD_CMD_PORT |= (1 << RS);

  //send data
  send_data(data);
}

//LCD init, call once
void lcd_init()
{
  LCD_CMD_DDR |= (1 << RS) | (1 << E) | (1 << RW);
  LCD_DATA_DDR = 0x0F;
  LCD_CMD_PORT &= ~((1 << RS) | (1 << E) | (1 << RW));
  LCD_DATA_PORT = 0x00;
```

```
  send_lcd_command(0x03);   //4-bit mode
  send_lcd_command(0x02);

  send_lcd_command(0x28);   //4-bit comm, 2 lines, 5x8 font
  send_lcd_command(0x0C);   //display ON, cursor OFF, blink OFF
  send_lcd_command(0x01);   //clear screen
  send_lcd_command(0x80);   //cursor go to top left corner
  _delay_ms(1);
}

/* Terms
 * S = Start
 * SR = Repeated Start
 * P = Stop
 * SLA+W = Slave Address Write mode
 * SLA+R = Slave Address Read mode
 * ACK = Acknowledge
 * NACK = Not ACK
 */

uint8_t readDS1307(uint8_t address, uint8_t data[])
{
  //send S
  TWCR = (1 << TWEN) | (1 << TWINT) | (1 << TWSTA);
  //wait complete then check status
  while(!(TWCR & (1 << TWINT)));
  if((TWSR & 0xF8) != 0x08)
    return 1;

  //send SLA + W
  TWDR = 0b11010000;
  TWCR = (1 << TWEN) | (1 << TWINT);
  //wait complete then check status
  while(!(TWCR & (1 << TWINT)));
  if((TWSR & 0xF8) != 0x18)
    return 2;

  //send register address
  TWDR = address;
  TWCR = (1 << TWEN) | (1 << TWINT);
  //wait complete then check status
  while(!(TWCR & (1 << TWINT)));
  if((TWSR & 0xF8) != 0x28)
    return 3;
```

```
  //send SR
  TWCR = (1 << TWEN) | (1 << TWINT) | (1 << TWSTA);
  //wait complete then check status
  while(!(TWCR & (1 << TWINT)));
  if((TWSR & 0xF8) != 0x10)
    return 4;

  //send SLA + R
  TWDR = 0b11010001;
  TWCR = (1 << TWEN) | (1 << TWINT);
  //wait complete then check status
  while(!(TWCR & (1 << TWINT)));
  if((TWSR & 0xF8) != 0x40)
    return 5;

  //wait for data
  int i;
  for(i = 0; i < 6; ++i){
      TWCR = (1 << TWEN) | (1 << TWINT) | (1 << TWEA);
      while(!(TWCR & (1 << TWINT)));
      data[i] = TWDR;
      if((TWSR & 0xF8) != 0x50)
        return 6;
  }
  TWCR = (1 << TWEN) | (1 << TWINT);
  TWCR &= ~(1 << TWEA);
  while(!(TWCR & (1 << TWINT)));
  data[i] = TWDR;
  if((TWSR & 0xF8) != 0x58)
    return 6;

  //send P
  TWCR = (1 << TWEN) | (1 << TWINT) | (1 << TWSTO);
  return 0;
}

uint8_t writeDS1307(uint8_t address, uint8_t data)
{
  //send S
  TWCR = (1 << TWEN) | (1 << TWINT) | (1 << TWSTA);
  //wait complete then check status
  while(!(TWCR & (1 << TWINT)));
  if((TWSR & 0xF8) != 0x08)
    return 1;

  //send SLA + W
  TWDR = 0b11010000;
  TWCR = (1 << TWEN) | (1 << TWINT);
```

```
  //wait complete then check status
  while(!(TWCR & (1 << TWINT)));
  if((TWSR & 0xF8) != 0x18)
    return 2;

  //send register address
  TWDR = address;
  TWCR = (1 << TWEN) | (1 << TWINT);
  //wait complete then check status
  while(!(TWCR & (1 << TWINT)));
  if((TWSR & 0xF8) != 0x28)
    return 3;

  //send data
  TWDR = data;
  TWCR = (1 << TWEN) | (1 << TWINT);
  //wait complete then check status
  while(!(TWCR & (1 << TWINT)));
  if((TWSR & 0xF8) != 0x28)
    return 5;

  //send P
  TWCR = (1 << TWEN) | (1 << TWINT) | (1 << TWSTO);
  _delay_ms(10);
  return 0;
}

void twi_init()
{
  //SCL, SDA as output
  DDRC |= (1 << DDC4) | (1 << DDC5);
  //init I2C
  //100kHz @ prescaler /4
  TWBR = 8;
  TWSR |= (1 << TWPS0);
  //enable I2C
  TWCR |= (1 << TWEN);
}

int main(void) {
    lcd_init();
    twi_init();
    char msg[10] = {};
    char days[][4] = {"Sun","Mon", "Tue", "Wed", "Thu", "Fri", "Sat"};
    char month[][4] = {"Jan", "Feb", "Mar", "Apr", "May", "Jun",
                       "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};
```

```
    // set initial time in 28/2/2021 time 22.19.45
    writeDS1307(0x00, 0x45); // second
    writeDS1307(0x01, 0x19); // minute
    writeDS1307(0x02, 0x22); // hour
    writeDS1307(0x03, 0x01); // day
    writeDS1307(0x04, 0x28); // date
    writeDS1307(0x05, 0x02); // month
    writeDS1307(0x06, 0x21); // year

    while (1) {
        uint8_t time[7];
        uint8_t error = readDS1307(0x00, time);
        char temp1[5] = {}, temp2[5] = {};
        char times[10] = {}, dates[16] = {};
        // date
        strcat(dates, days[((time[3] >> 4) * 10 + (time[3] & 0x0F)) - 1]);
        sprintf(temp1, " %.2u ", ((time[4] >> 4) * 10 + (time[4] & 0x0F)));
        strcat(dates, temp1);
        strcat(dates, months[((time[5] >> 4) * 10 + (time[5] & 0x0F)) - 1]);
        sprintf(temp2, " %.2u", 2000 + ((time[6] >> 4) * 10 + (time[6] & 0x0F)));
        strcat(dates, temp2);
        // time
        sprintf(times, "Times:%.2u:%.2u:%.2u",
                ((time[2] >> 4) * 10 + (time[2] & 0x0F)),
                ((time[1] >> 4) * 10 + (time[1] & 0x0F)),
                ((time[0] >> 4) * 10 + (time[0] & 0x0F)));

        int i = 0;
        send_lcd_command(0x80);
        while(dates[i] != 0)
        {
            send_lcd_data(dates[i++]);
        }
        _delay_ms(100);
        send_lcd_command(0xC0);
        i = 0;
        while(times[i] != 0)
        {
            send_lcd_data(times[i++]);
        }
        _delay_ms(100);
    }
}
```

Result in Lab 6

รูปที่ 2 : Time on desktop

รูปที่ 3 : Time on LCD