# B31DD – Embedded Systems

# Lab 3

# Serial Communication and LCD

**SUBMISSION DEADLIN 17 nov 2021**

**JANNIN Sylvain**

**H00387879**

Msc Robotics

Date of submission: 12 nov 2021

# *Task 1*

## Electric circuit

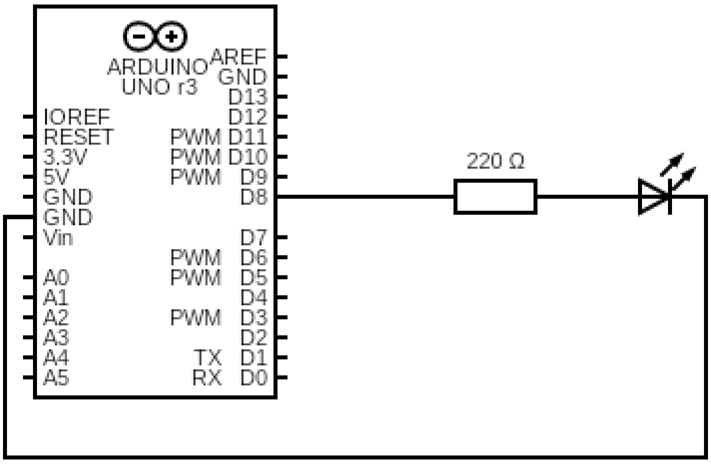

*Figure 1: task 1 electric schematic.*

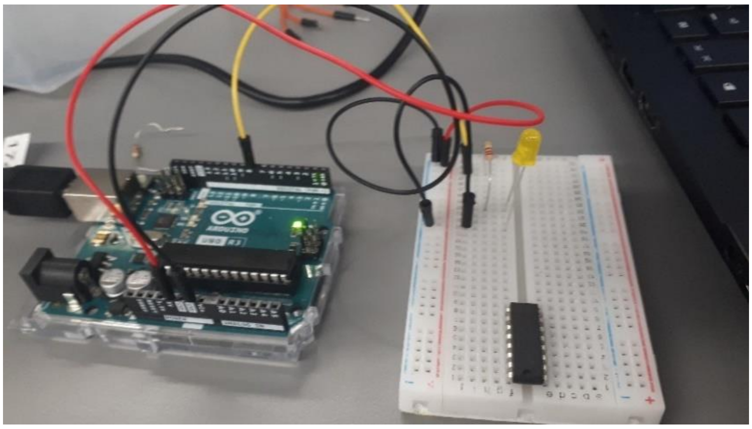

*Figure 2: task 1 montage: LED off.*

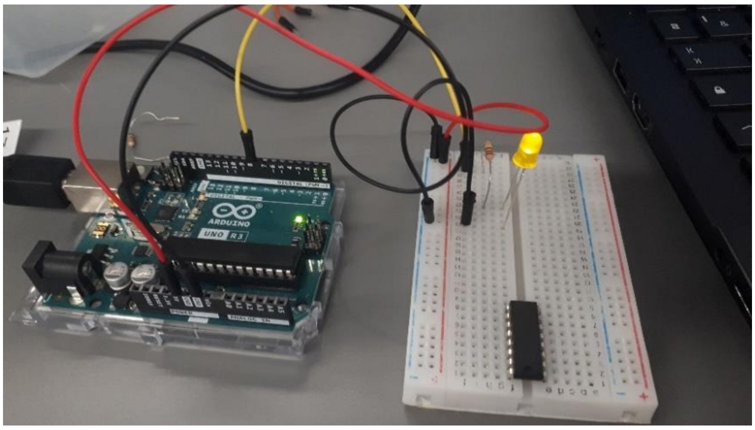

*Figure 3: task 1 montage: LED on.*

The circuit is very simple, it is an LED with a resistor of 220Ω in series.

## Programming

### Program idea

The goal is to turn on or off an LED when the microcontroller received the good data, “*Y*” for on “*N*” for off.

To make this, in the main we first call the UART initialisation, and put an LED as output. For this, in the while loop we wait for the controller to receive a data thanks to “*USART_receive*” and then there is a switch case which turn on or off the LED weather it receives an “*Y*” or an “*N*”.

The controls the LED by typing letters on a PC and it transmits it to the Arduino.

### Register

“*USART_init”* is the function which initialise UART. In *UBR0* we set the prescaler of the baud rate we chose. In *UCSR0B* we set to 1 RXEN and TXEN0 to enable the UART transmit and in *UCSR0C*, *UCZ0* is set to 0b011 to set to 8 bit the data.

In “*USART_receive*” it waits for the *RXC0* flag in the *UCSR0A* to be set. Once the flag is set, it means data has been received and then, the function returns the data in *UDR0*.

## Code

```
/*
 * lab3__JANNIN_Sylvain.c
 *
 * Created: 03/11/2021 10:23:52
 * Author : sylva
 */

#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>
#define BAUDRATE 9600
#define BAUD_PRESCALLER (((F_CPU / (BAUDRATE * 16UL))) - 1)


void USART_init(void)
{
       UBRR0H = (uint8_t)(BAUD_PRESCALLER>>8); //set baud rate
       UBRR0L = (uint8_t)(BAUD_PRESCALLER);
```

```
    UCSR0B = (1<<RXEN0)|(1<<TXEN0); //enable transmit
    UCSR0C = (3<<UCSZ00); //set 8-bit (default)
}


unsigned char USART_receive(void)
{
    while(!(UCSR0A & (1<<RXC0))); // Wait to receive data
    return(UDR0); // Read data from UDR
}

int main()
{
    USART_init();       //init UART

    DDRB |= 1 << PINB0; //set PB0 as output

    unsigned char data; //init the type of data UART will received

    while(1)
    {
        data = USART_receive();    //wait for the User to send a data
        switch(data)
        {
            case('Y'):    //if it is 'Y'
                PORTB |= (1 << PINB0);     //Turn on led PB0
                break;
            case('N'):    //if it is 'N'
                PORTB &= ~(1 << PINB0); //Turn off led PB0
                break;

            default:      //Otherwise, do nothing
                break;
        }
    }
}
```

# *Task 2*

## Electric circuit

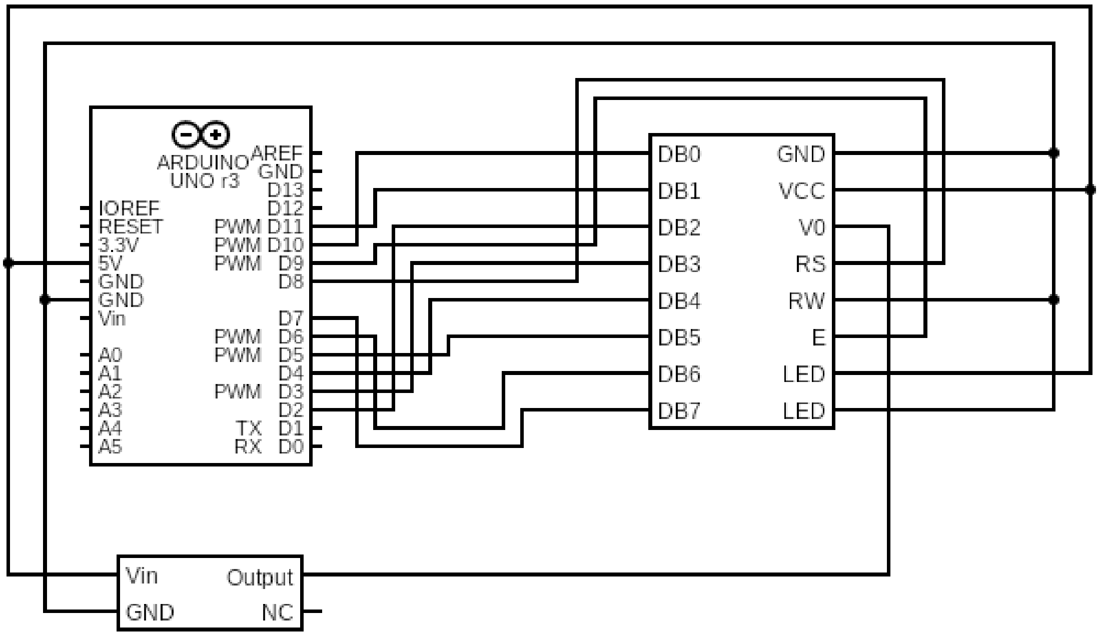

*Figure 4: task 2 electric schematic.*

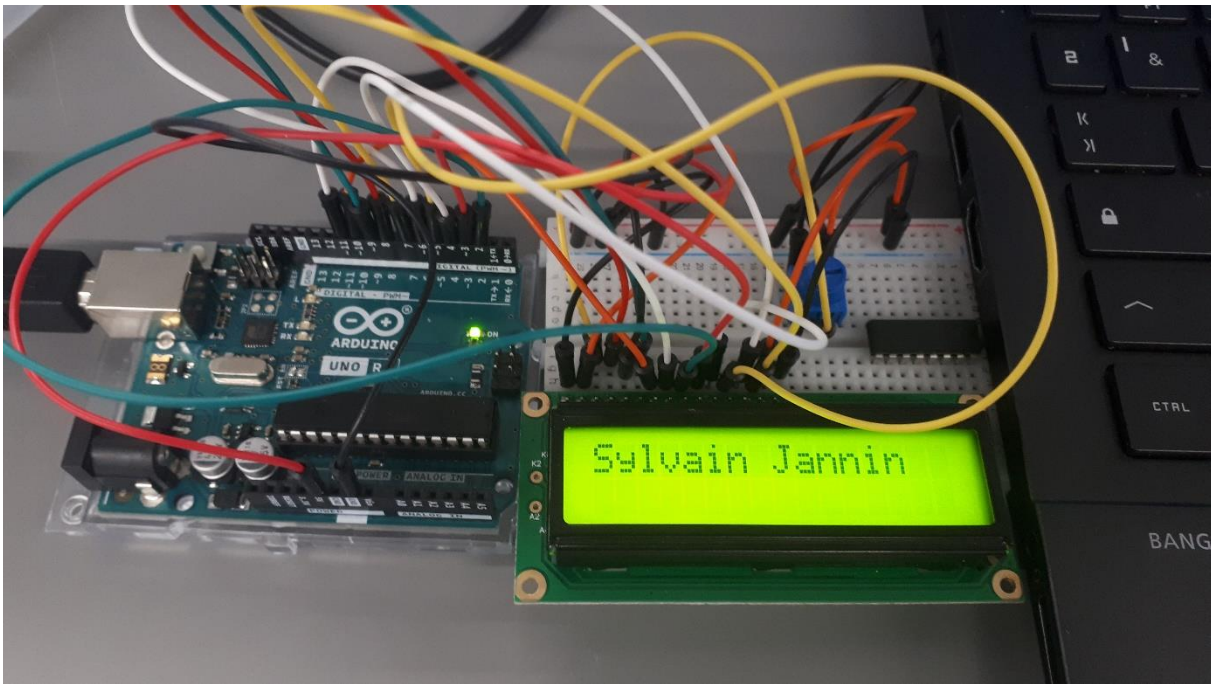

*Figure 2: task 2 montage with the name displayed.*

The electric circuit is a bit hard to understand because there are a lot of wires but in the end, it is quite easy.

The circuit at the right of the schematic is the LCD and at the one at the bottom is the potentiometer.

PB0 is connected to RS and PB1 to E (enable). All the other pins from the Arduino are connected from DB0 to DB7 of the LCD. The output of the potentiometer is connected to V0, it controls the luminosity of the LCD.

## Programming

### Program Idea

In defines at the beginning, we change D0 and D1 from PD0 and PD1 toPB2 and PB3 because they were first connected on the Tx and Rx pins, since we use UART we cannot use these pins for both UART and for the LCD.

We set as output all the port B and D. we initialise the LCD thanks to *lcd.h* and the UART with *USART_init*. There is a static list of char called "*name*" with a length of 14 because "*Sylvian Jannin*" has 14 characters. We also initialise *counter* to 0, this variable represents in which cell of name we are going to write data in.

For the while loops, do a "do while":

- Wait for the data to be received with *USART_receive* and we store it in *data*.
- Send this data to check if we received the good data with *USART_send*.
- In *name*, we set the data we received.
- We increment counter of 1.

We do it one time then there is a while loop, and it does it as long as *counter* is lower than the length of *name*. Then we add *'\0'* in *name* to indicates it is the end of the list of chars.

Finally, we set the cursor at the beginning of the LCD. And we use the "*function Lcd8_Write_String"* from *lcd.h* to display *name*. then the programs wait for 10 seconds before cleaning the LCD and redo the program.

To display the name, the user has to type one letter on a computer and send it to the Arduino. The programs stores the information in *name* and once 14 characters has been written, the Arduino send the information to the LCD.

### Register

"*USART_init"* is the function which initialise UART. In *UBR0* we set the prescaler of the baud rate we chose. In *UCSR0B* we set to 1 RXEN and TXEN0 to enable the UART transmit and in *UCSR0C*, *UCZ0* is set to 0b011 to set to 8 bit the data.

In “*USART_send*”, it does not send data as long as the *UDRE0* flag in the *UCSR0A* is not set, so program wait for it, it is uses to check if the data is sent. Once the flag is set, we store the data in *UDR0*.

“*USART_receive*” is very close to the previous function. Instead of waiting the *UDRE0* flag, it waits for the *RXC0* flag. Once it is set it means data has been received and then, the function returns the data in *UDR0*.

**Code:**

```
/*
 * Lab3_task2__JANNIN_Sylvain.c
 *
 * Created: 03/11/2021 15:22:15
 * Author : sylva
 */

#define D0 eS_PORTB2
#define D1 eS_PORTB3
#define D2 eS_PORTD2
#define D3 eS_PORTD3
#define D4 eS_PORTD4
#define D5 eS_PORTD5
#define D6 eS_PORTD6
#define D7 eS_PORTD7
#define RS eS_PORTB0
#define EN eS_PORTB1
#define F_CPU 16000000UL
#include <avr/io.h>
#include "lcd.h"
#include <util/delay.h>
#include <string.h>

#define BAUDRATE 9600
#define BAUD_PRESCALLER (((F_CPU / (BAUDRATE * 16UL))) - 1)



void USART_init(void)
{
	UBRR0H = (uint8_t)(BAUD_PRESCALLER>>8); //set baud rate
	UBRR0L = (uint8_t)(BAUD_PRESCALLER);
	UCSR0B = (1<<RXEN0)|(1<<TXEN0); //enable transmit
	UCSR0C = (3<<UCSZ00); //set 8-bit (default)
}
void USART_send(unsigned char data)
{
	while(!(UCSR0A & (1<<UDRE0))); // check if data is sent
	UDR0 = data; // load new data to transmit
}

unsigned char USART_receive(void)
{
	while(!(UCSR0A & (1<<RXC0))); // Wait to receive data
	return(UDR0); // Read data from UDR
}
```

```
int main(void)
{
       DDRD = 0xFF;  //set as output all PORTD
       DDRB = 0xFF;  //set as output all PORTB
       Lcd8_Init();  //init LCD
       USART_init(); //init UART

       unsigned char data;        //declare the variable used to get data from the PC
       int len_name = 14;         //"Sylvain Jannin" --> 14
       char name[len_name];//create a list of char with a length of 14
       int counter = 0;           //variable uses to add information in "name"

       while(1)
       {
              data = USART_receive();           //do a first wait until user send a
data trought the PC
              USART_send(data);                 //resend the data to the PC to check if
it is correct or not
              name[counter] = data;             //add this information in "name"
              counter ++;                             //increment the counter
              while(counter < len_name)  //redo these steps until  we have a
              {
                     data = USART_receive();    //get the information send by user
trought UART
                     USART_send(data);          //resend the data to the PC to check if
it is correct or not

                     name[counter] = data;      //add this information in "name"
                     counter ++;                      //increment the counter
              }

              name[counter] = '\0';      //add '\0' to indicates the end of the string
              counter = 0;               //reset counter

              Lcd8_Set_Cursor(0,0);      //set the position of the cursor at the
begining
              Lcd8_Write_String(name);//write on the LCD the name
              _delay_ms(10000);          //wait 10s
              Lcd8_Clear();              //clear the LCD
       }
       return(0);
}
```