Jacobs University Bremen

Natural Science Laboratory

Embedded Systems Laboratory
CO26-300312

Fall Semester 2017

Blink LED in C Program

Mailbox :
104

Instructor :
Ph.D. Fangning Hu

Author :
Biruk Amare

September 19, 2017

JACOBS
UNIVERSITY

1 Introducion

The main objective of this lab is to use the predefined register names, delay functions, and interrupt vector names to control LED's by implementing our program in C.

1.1 Introduction to AVR Digital Input/output

Atmel AVR 16-bit microcontrollers provide pins to take in/output information from/to the outside world in the form of logic values. The pins are organized in groups and referred to as a port. The AVR uses the alphabet to name these ports, which are `PortA, PortB, PortC, and PortD`. The pins of PortA are : `PA0 - PA7`. It is also wise to notice the alternate name of the ports pins. The AVR microcontrollers are designed to allow dual use of most of its pins. This has the advantage of allowing a developer to use these pins as I/O pins if the function they are provided for is not being utilized. In this lab, we are only concerned with the pins in their digital I/O functions.

1.2 Associated I/O Registers

Each of the AVR Digital I/O ports is associated with three(3) I/O registers. A Data Direction Register(DDRx), A pin Register (PINx) and a Port Register (PORTx), where x is the port of `A, B, C...etc`.

1.2.1 DDRx - Port X Data Direction Register

DDRx is an 8-bit register which stores configuration information for the pins of Portx. Writing a 1 in the pin location in the DDRx makes the physical pin of that port an output and writing a 0 makes that pin an input pin.

1.2.2 PINx - Port X Input Pins Register

PINx is an 8-bit register that stores the logic values, the current state, of the physical pins on Portx. So to read the values on the pins of Portx, you read the values that are in its PIN register.

1.2.3 PORTx - Port X Data Register

PORTx is an 8-bit register which stores the logic values that currently being outputted on the physical pins of Portx if the pins are configured as output pins. So to write values to a port, you write the values to the PORT register of that part.

Setup for blinking three LEDs half second one after another.

2 Prelab

2.1 Read the AVR C Programming Basic in my Introduction Chapter and get familiar with the logic operation.

We needed to include the following header files :

```
1  #include <avr/io.h>
2  #define F_CPU 8000000UL
3  #include <util/delay.h>
4  #include <avr/interrupt.h>
5
6  int main(void)
7  {
8
9  DDRD = 0xFF;
10
11 //add yourcodes here
12 }
```

Operator	Function
&	Binary AND operator
|	Binary OR operator
~	One complement operator
^	Binary XOR operator
<<	Binary left shift operator
>>	Binary right shift operator

2.2 Read the content related to <avr/io.h> <util/delay.h> <avr/interrupt.h>

`<avr/io.h>` is a compressed header file which includes 4 other header files. This header file includes the appropriate IO definitions for the device being specified by the command-line switch. The 4 other header files are : `#include <avr/sfr_defs.h>`, `#include <avr/portpins.h>`, `#include <avr/common.h>`, and `#include <avr/version.h>`.

`<util/delay.h>` is a header file that includes two delay functions :

— `void _delay_ms()` is used for milliseconds and

— `void _delay_us()` is used for microseconds.

`<avr/interrupt.h>` is a header file for handling interrupts. this is the header files that handles the sei. This means that interrupt vector table are contained here.

2.3 Find out the meaning of line 2 of the program in section 2.1

```
1  #define F_CPU 8000000UL
```

This is to define the clock rate or frequency to $8MHz$. `UL` stands for `Unsigned Long`, which is used for unsigned large numbers.

2.4 Find the interrupt vector names in <avr/interrupt.h>

For this session, we used `INT0_vec` and `INT1_vec`.

3 Lab Assignments

3.1 Connect 3 LEDs to PORTD and make them blink half second one after another continuously (you need to use the delay function to produce the half second delay).

The program used for this task is :

```
/*
 * lab3.c
 *
 * Created: 12.09.2017 14:25:23
 * Author : biruk
 */

#include <avr/io.h>
#define F_CPU 8000000UL
#include <util/delay.h>
#include <avr/interrupt.h>

int main(void)
{
  DDRD = 0xFF;
  PORTD = 0x00;

  while(1){
     PORTD = 0b10000000;
     _delay_ms(500); //half a second
     PORTD = 0x00;
     _delay_ms(500);
     PORTD = 0b01000000;
     _delay_ms(500); //half a second
     PORTD = 0x00;
     _delay_ms(500);
     PORTD = 0b00100000;
     _delay_ms(500); //half a second
     PORTD = 0x00;
     _delay_ms(500);
  }

  //add your codes here
}
```

And its corresponding circuit diagram is :

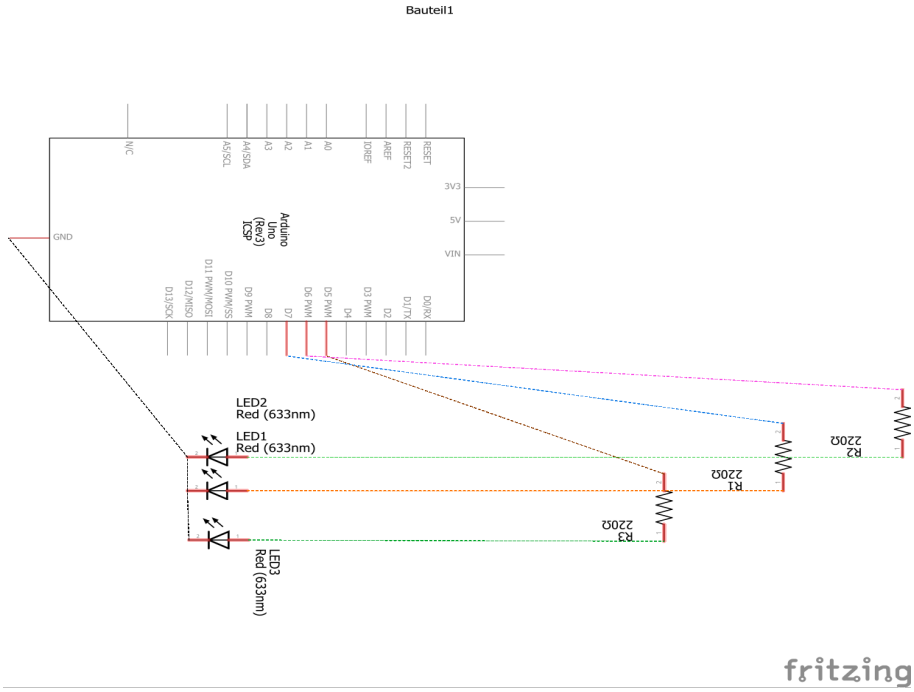

FIGURE 1 – Circuit Diagram

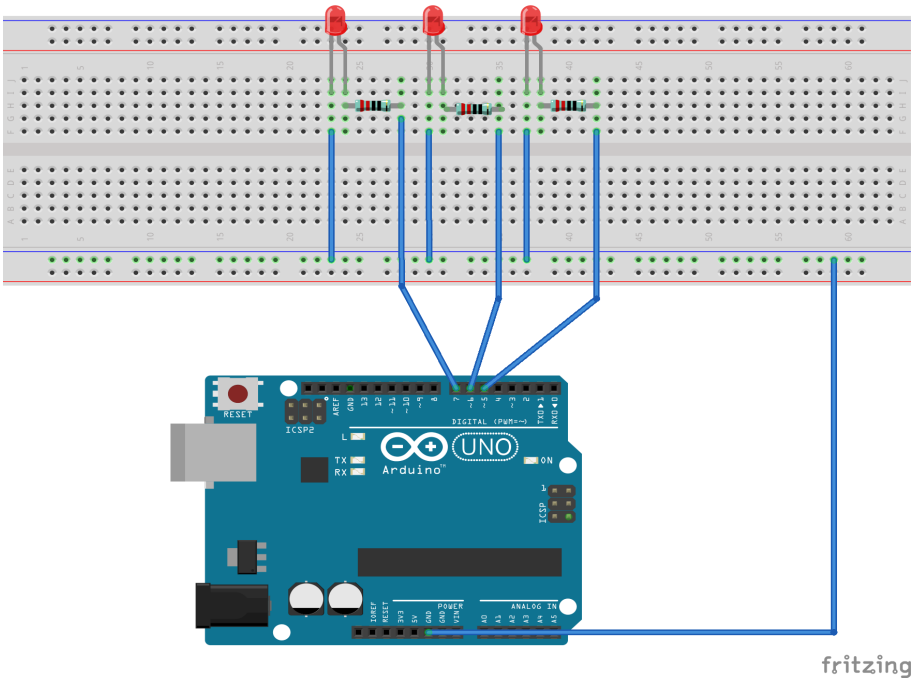

FIGURE 2 – Breadboard Setup

3.2 Triggering an external interrupt by a pressing button, after pressing the button, all the 3 LEDs blink three times simultaneously.

```
/*
 * lab3.c
 *
 * Created: 12.09.2017 14:25:23
 * Author : biruk
 */

#include <avr/io.h>
#define F_CPU 8000000UL
#include <util/delay.h>
#include <avr/interrupt.h>

int main(void)
{
  EICRA = 0b00001010;
  EIMSK = 0b00000011;

  DDRD = 0xF0;
  PORTD = 0x00;
  sei();

  while(1){
     PORTD = 0b10000000;
     _delay_ms(500); //half a second
     PORTD = 0x00;
     _delay_ms(500);
     PORTD = 0b01000000;
     _delay_ms(500); //half a second
     PORTD = 0x00;
     _delay_ms(500);
     PORTD = 0b00100000;
     _delay_ms(500); //half a second
     PORTD = 0x00;
     _delay_ms(500);
  }

  //add your codes here
}

ISR(INT1_vect)
{
    unsigned char i;
    for(i = 0; i<3 ;i++){
      PORTD = 0xFF;
      _delay_ms(500);
      PORTD = 0x00;
      _delay_ms(500);
      PORTD = 0xFF;
      _delay_ms(500);
```

```
50        }
51
52    }
```

The corresponding circuit diagram is :

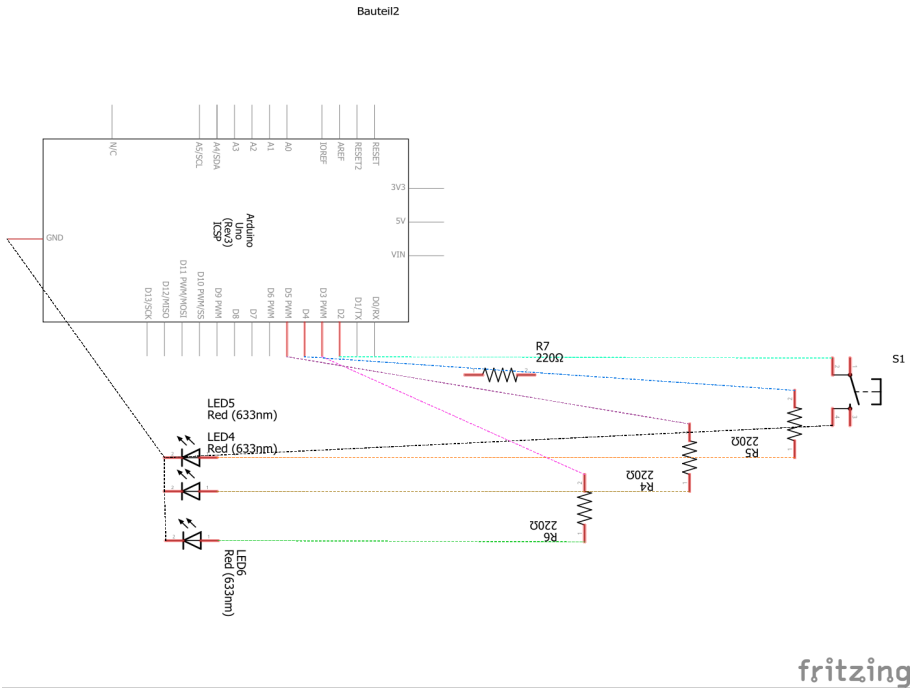

FIGURE 3 – Circuit Diagram

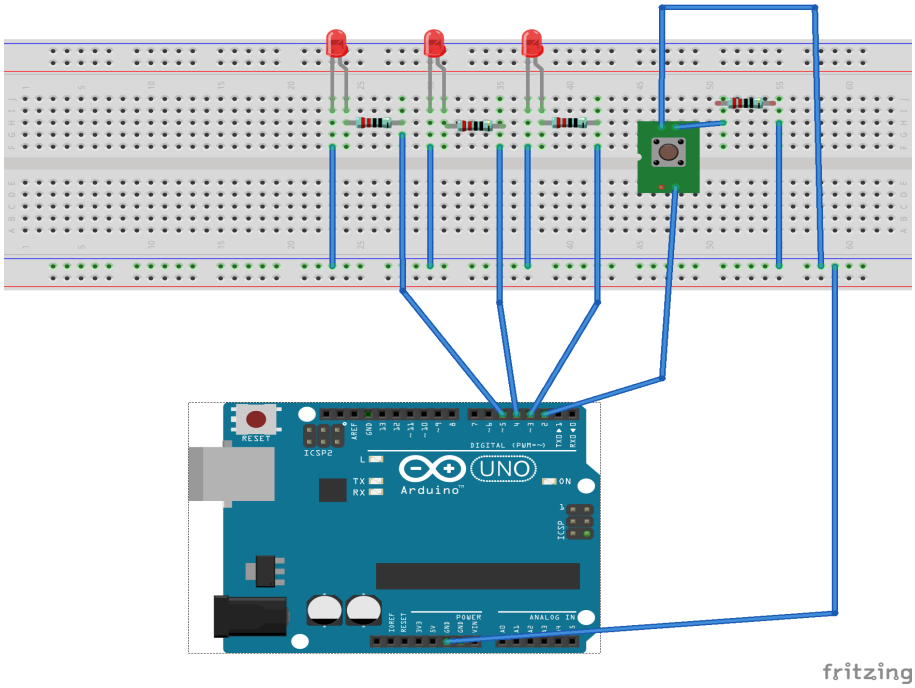

FIGURE 4 – Breadboard Setup

3.3 Debug your code in Atmel Studio and you will see a Deassembly window.

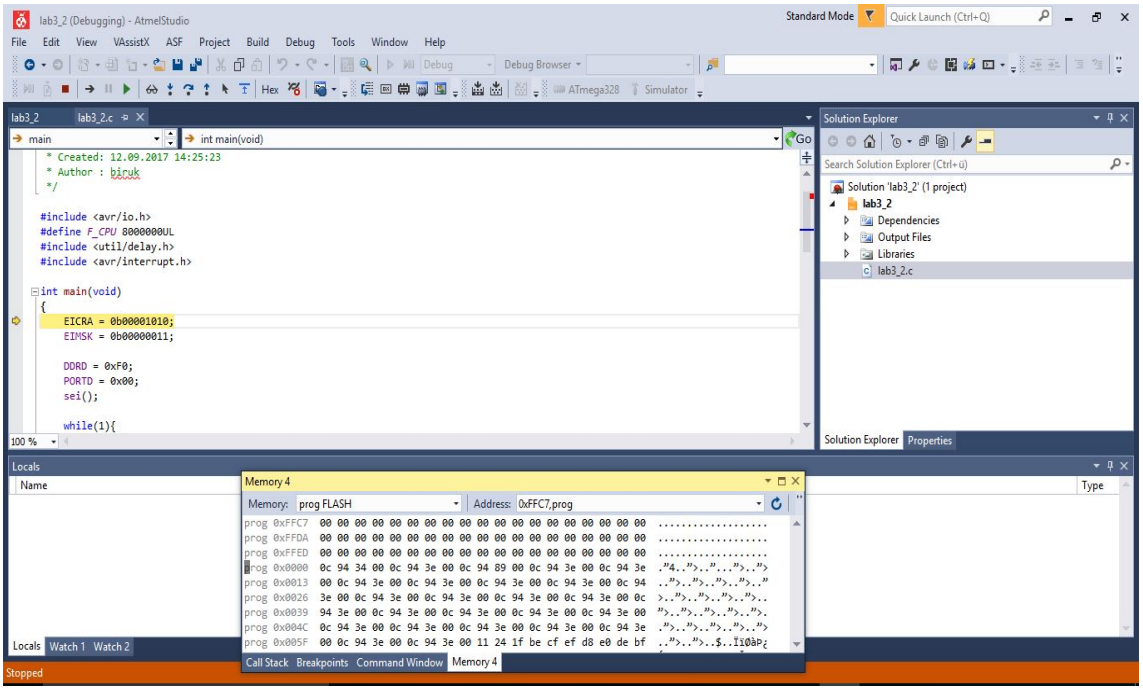

FIGURE 5 – Deassembly Window

4 References

[1] http://www.atmel.com/images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-
datasheet_Complete.pdf1

[2] http://embsys-fhu.user.jacobs-university.de/?page_id=49