Jacobs University Bremen

Natural Science Laboratory

Embedded Systems Laboratory
CO26-300312

Fall Semester 2017

Blinking LED : Assembler

Mailbox :
104

Instructor :
Ph.D. Fangning Hu

Author :
Biruk Amare

September 4, 2017

JACOBS UNIVERSITY

1 Introducion

A microcontroller is a single chip containing at least a CPU, non-volatile memory, volatile memory, a timer and an I/O control unit.

Basically, a computer can be described as a computer on a chip. The difference between a microcontroller and a regular PC is that the PC is a general purpose computer while a microcontroller is a computer dedicated to one or just a few tasks.

Apart from the above mentioned, microcontroller components usually also include serial communication capabilities, interrupt controls and analog I/O capabilities.

2 Prelab

2.1 Study the datasheet Chapter I/O Ports, find out how to use the above three registers

Three I/O registers are used to control each PORTx. They are :

— Data Register : PORTx - This is the Data Register and contains the current value of the port. The port pin register bit PORTxn are accessed at the PORTx I/O address.
$\Rightarrow$ If the port is written a logic one and the particular pin is configured as an output, then the port pin is driven high, as is similar when the port is written a logic zero where the port pin is driven low.
$\Rightarrow$ If the port is written a logic one and the particular pin is configured as an input, then the pull-up resistor is activated, whereas a logic zero at an input configured pin would switch the pull-up resistor off.

— Data Direction Register : DDRx - This is the Data Direction Register and controls the direction (output or input) of the pin. The port pin register bit DDxn are accessed at the DDRx I/O address.
$\Rightarrow$ If the specific bit is applied a logic one, then the pin number of the applied bit number is set as output.
$\Rightarrow$ The same goes if a logic zero is applied, then the that pin is set as an input.

— Port Input Pin Register : Pinx - This is the Port Input Pin Register and contains the input value of the read-only port. The port pin register bit PINxn are accessed at the PINxn I/O address.

2.2 Study the assembly instructions LDI, OUT, SBI, CBI, JMP/RJMP, CALL/RCALL, RET, DEC, BRNE, CLI and try to understand my assembly examples

Instruction	Operands	Description	Operation	#Clock Note
LDI	Rd,K	Load Immediate	Rd$\leftarrow$ K	1
OUT	P,Rr	Out Port	P$\leftarrow$ Rr	1
SBI	P,b	Set Bit in I/O Register	I/O(P,b)$\leftarrow$ 1	2
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) $\leftarrow$ 0	2
JMP	k	Jump	$PC \leftarrow$ k	3
RJMP	k	Relative Jump	$PC \leftarrow$ PC+K+1	2
CALL	k	Call Subroutine	$PC \leftarrow$ k	4
RCALL	k	Relative Call Subroutine	$PC \leftarrow$ PC+k+1	3
RET		Subroutine Return	$PC \leftarrow$ STACK	4
DEC	Rd	Decrement	Rd$\leftarrow$ Rd-1	1
BRNE	k	Branch if Not Equal	if (Z=0) then PC $\leftarrow$ PC+k+1	1/2
CLI		Global Interrupt Disable	I$\leftarrow$ 0	1

2.3 The CPU clock is 20Mhz (you can check the datasheet system clock chapter), calculate how many CPU clock cycles you need to have 1 second delay. Assume implementing each assembly instruction need one CPU clock cycle, change the code in the last examples such that the Delay subroutine produce 1 second delay

Since the CPU clock is $20MHz$, it means that each cycle has a period of $\frac{1}{20{,}000{,}000}s = 50ns$ and hence, we need 20,000,000 CPU clock cycles to obtain the 1 second delay we require.

```
.include "m328def.inc"
.org 0x0000
RJMP begin ; Jump to begin
.org 0x0034
begin:
        CLI                 ; Clear Interrupt

        LDI R16, low(RAMEND)
        OUT SPL, R16
        LDI R16, high(RAMEND)
        OUT SPH, R16

        LDI R16, 0xFF
        OUT DDRD, R16
```

```
mainloop:
        LDI R16, 0xFF
        OUT PORTD, R16
        RCALL Delay
        LDI R16, 0x00
        OUT PORTD, R16
        RCALL Delay
        RJMP mainloop

Delay: LDI R17, 0xFF
loop0: LDI R20, 0x80
loop1: LDI R19, 0x7B
loop2: DEC R19
           BRNE loop2
           DEC R20
           BRNE loop1
           DEC R17
           BRNE loop0
           RET
```

3 Lab Assignment

3.1 Debug the assembly examples in the simulator and observe how the registers, the program counter (PC) change after implementing each instruction

We used the Debug Option (F11) to go step-by-step through the code and observed the changes to the memory - register states.

3.2 Write the assembly codes to toggle the PORTD and delay 1 second after toggling

```
.include "m328def.inc"
.org 0x0000
RJMP begin ; Jump to begin
.org 0x0034
begin:
        CLI               ; Clear Interrupt

        LDI R16, low(RAMEND)
        OUT SPL, R16
        LDI R16, high(RAMEND)
        OUT SPH, R16
```

```
        LDI R16, 0xFF
        OUT DDRD, R16

mainloop:
        LDI R16, 0xFF
        OUT PORTD, R16
        RCALL Delay
        LDI R16, 0x00
        OUT PORTD, R16
        RCALL Delay
        RJMP mainloop

Delay: LDI R17, 0xFF
loop0: LDI R20, 0x80
loop1: LDI R19, 0x7B
loop2: DEC R19
           BRNE loop2
           DEC R20
           BRNE loop1
           DEC R17
           BRNE loop0
           RET
```

3.3 Connect one pin from PORTD, an LED, a resister between 200 Ohm to 500 Ohm to either a 5V power suppler or Ground. An example diagram of the circuit is shown below.

This was performed and approved by the professor.

3.4 Upload your code to the chip, make the LED continuously blink with period of 1 second

This was performed and approved by the professor.

4 Evaluation

4.1 Give your circuit diagram. Explain you circuit design

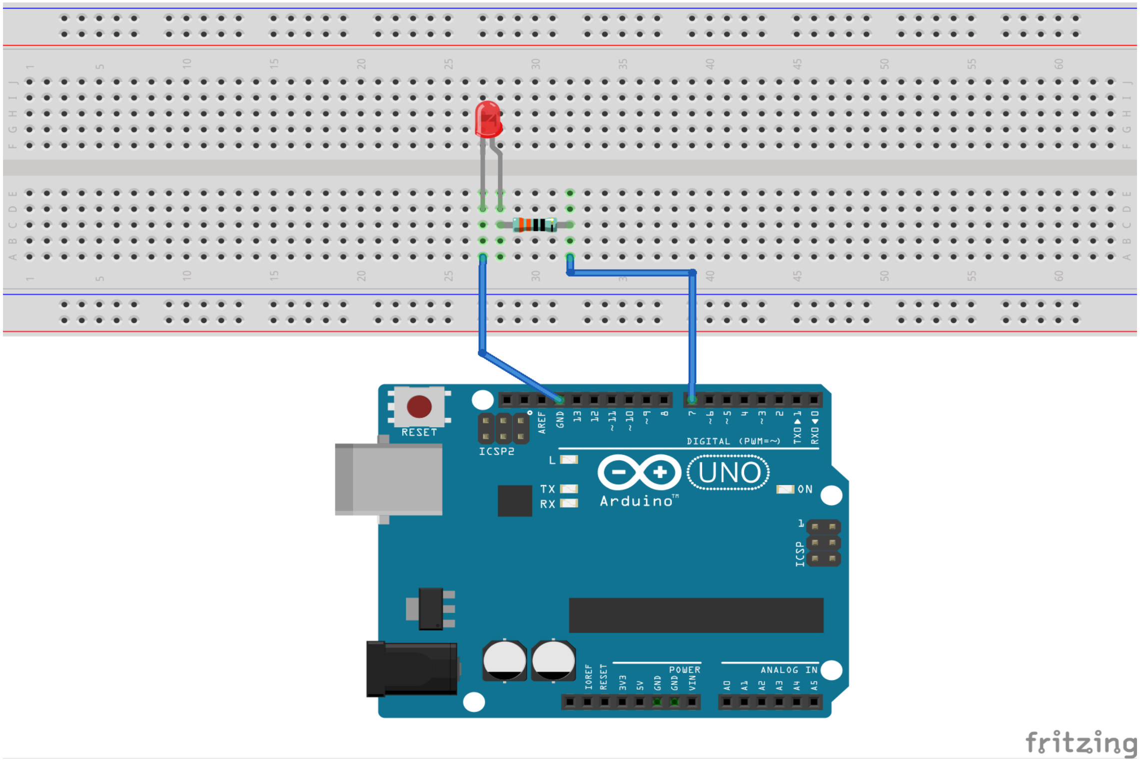

Figure 1 – Breadboard Diagram

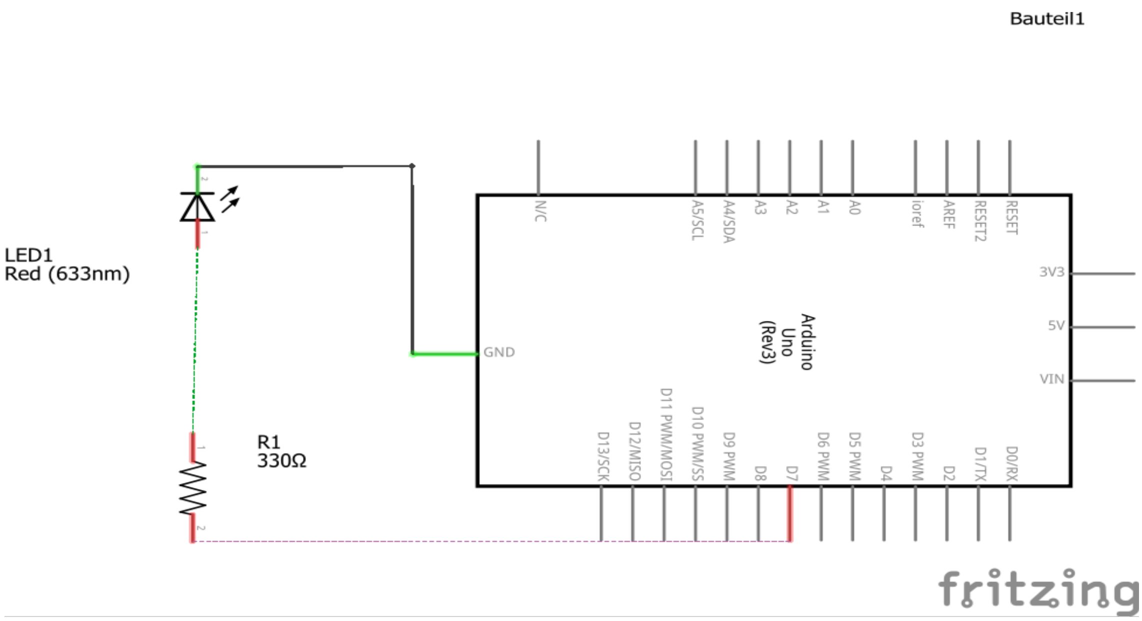

Figure 2 – Schematic Diagram

- LED is connected from D10(Anode) to D11(Cathode)
- 330Ω Resistor goes from B11 to anywhere on breadboard ground (GND)
- Black jumper wire goes from Arduino GND to breadboard Ground
- Red jumper wire goes from Arduino 5V to Breadboard Power
- Yellow jumper wire goes from Arduino PIN13 to breadboard E10

4.2 Give the program codes with comments. Explain you codes, especially how you set each registers bits and what they mean

```
.include "m328def.inc"
.org 0x0000
RJMP begin ; Jump to begin
.org 0x0034
begin:
        CLI; Clear Interrupt

        LDI R16, low(RAMEND)
        OUT SPL, R16
        LDI R16, high(RAMEND)
        OUT SPH, R16

        LDI R16, 0xFF
        OUT DDRD, R16; set all as output pins

mainloop:
```

```
        LDI R16, 0xFF ; turn on LED
        OUT PORTD, R16
        RCALL Delay ; call delay
        LDI R16, 0x00
        OUT PORTD, R16; turn off LED
        RCALL Delay ; call delay again
        RJMP mainloop ; and loop again

Delay: LDI R17, 0xFF ; 255
loop0: LDI R20, 0x80 ; 128 : manually chosen to get a second delay blink
loop1: LDI R19, 0x7B ; 123 : manually chosen to get a second delay blink
loop2: DEC R19 ; decrement until it reaches loop 2
           BRNE loop2 ; check if it reached loop2
           DEC R20 ; decrement until it reaches loop 1
           BRNE loop1 ; check if it reached loop2
           DEC R17 ; decrement until it reaches loop 0
           BRNE loop0 ; check if it reached loop2
           RET ; return
```

4.3 Answer the questions in the Pre Lab Tasks and the Lab Assignments, for example how the PC changes after each instruction.

This is done in sections 2 and 3 of the lab report.

5 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