

Microprocessor Systems Design

EEE 42101

Experiment 1: Environment Setup and LED Blink

Objectives:

- Study Basic peripherals and modules of microcontroller.
- Be familiar with the development environment.
- Put hand on the basics of compilation, linking and loading(flashing).

Tools:

- 1- PC
- 2- Arduino Nano board
- 3- Testing board
- 4- MiniB-USB cable

Note: all material and sources of this course will be available on:

https://github.com/ashrafmalraheem/Mircoprocessor_Course

Feel free to download, study and modify for your own projects.

Part 1: Compilation

Create or open the template **main.c** files. Study its content. Ensure that it will compile successfully.

You can add any code make some random arithmetic operations.

You can check the output files that you get: **main.asm, mian.o, mian.hex, ..etc**

Part 2: Setting Registers

In the platform that you are using: Atmega328p in Arudiuno Nano boards. The LED L5 is connected to Port B pin no. 5. You should set this LED to blink at different rate.

First:

you should configure the direction register as output. **DDRB**

Second:

You can then set a value (0 or 1) in the port to switch on the LED.

Use loop to act as a delay.

13.2.1 Configuring the Pin

Each port pin consists of three register bits: DDxn, PORTxn, and PINxn. As shown in [Section 13.4 "Register Description" on page 72](#), the DDxn bits are accessed at the DDRx I/O address, the PORTxn bits at the PORTx I/O address, and the PINxn bits at the PINx I/O address.

The DDxn bit in the DDRx register selects the direction of this pin. If DDxn is written logic one, Pxn is configured as an output pin. If DDxn is written logic zero, Pxn is configured as an input pin.

If PORTxn is written logic one when the pin is configured as an input pin, the pull-up resistor is activated. To switch the pull-up resistor off, PORTxn has to be written logic zero or the pin has to be configured as an output pin. The port pins are tri-stated when reset condition becomes active, even if no clocks are running.

If PORTxn is written logic one when the pin is configured as an output pin, the port pin is driven high (one). If PORTxn is written logic zero when the pin is configured as an output pin, the port pin is driven low (zero).

Figure 1 Atmega328p I/O pin configuring

Table 13-1. Port Pin Configurations

DDxn	PORTxn	PUD (in MCUCR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output low (sink)
1	1	X	Output	No	Output high (source)

Figure 2 Atmega328p port pin configuration

Part 3: Include libraries

Instead of using the loops as delays now you should include a library to use the delay function.

```
#include <delay.h>

_delay_ms();
```

These libraries are precompiled so you can't find their source code (delay.c). You can only find object codes.

You should search for (delay.h) explore it and find other functions. It is on avr-gcc folder in the MinGW folder in C:/

Part 4: Make your code smarter

Instead of accessing the registers directly, now you should use functions and macros.

Define a function to set the LED on and OFF depend on the parameter that you pass to them.

- 1- use macros
- 2- use functions