# Introduction to Bare Metal Programming in

# Arduino Uno with Assembler

## Components and supplies



### Introduction

In this tutorial, we are going to see how to program the Arduino Uno without using the Arduino IDE. We will see how the Arduino IDE works under the hood.

### How Arduino IDE works

The Arduino IDE uses the avr-gcc compiler and avrdude to upload our program in the microcontroller. So, we are going to compile using avr-gcc the source code (written in C) to obtain the corresponding object file.

Then through avr-gcc, we link the system libraries to the object file to produce the executable or the ELF file.

Using avr-objcopy, we can translate the executable into a binary file that can be uploaded in the Arduino board using avrdude.

**Install the tools**

The commands to install the tools are for Ubuntu/Debian machine.

>$ sudo apt-get update  
>$ sudo apt-get upgrade -y

And then we install the package required by avr and avrdude.

>$ sudo apt-get install avra avrdude

### Turning LED on PB5 on

Here's the code to turn the built-in led on. (**led\_on.asm**).

source:

[https://www.instructables.com/Command-Line-Assembly-Language-Programming-for-Ard/](https://www.instructables.com/Command-Line-Assembly-Language)

; name: led\_on.asm

; assemble: avra led\_on.asm

; flash: avrdude -F -V -c arduino -p ATMEGA328P -P /dev/ttyACM0 -b 115200  
; -U flash:w:led\_on.hex

; description: turns led on port 13 on.

; slightly modified for my own needs and coding style.

; It turns on the LED which is connected to PB5 (digital out 13).

.nolist

.include "./m328Pdef.inc"

.list

start:

ldi r16,0xFF ; r16 = 0xFF (255)

out DDRB,r16 ; Out writes to SRAM, which is one way of accessing pins. DDRB controls PORTB's in/out state.

ldi r16,0x00 ; r16 is where we'll store current LED state, 0x20 means pin 13 high.

out PORTB,r16 ; set all B pins to current state. PORTB is where our favorite flashing pin is (pin 13)!

loop:

rjmp loop

### Assemble and upload

Attach the Arduino Rev3 to your USB port. At my system here it’s default mounted onb /dev/ttyACM0 so I use this in the next commandlines.

Unlike C we don’t need to compile and link the program. We just assemble it with  
  
**avra led\_on.asm -l led\_on.lst**

The command creates a led\_on.hex file which can be uploaded to the ATMEGA328P microcontroller on the Arduino Uno Rev3 .

**avrdude -F -V -c arduino -p ATMEGA328P -P /dev/ttyACM0 -b 115200 -U flash:w:led\_on.hex**

A look at the microcontroller board shows that the led on digital out port 13 is on.

### List file

I’ve additionally creeated a list file with the commandline option -l led\_on.lst. A simple **cat led\_on.lst** reveals the created listing:

AVRA   Ver. 1.3.0 led\_on.asm Wed Nov  3 16:55:09 2021   
  
  
        ; name:        led\_on.asm   
        ; assemble:    avra led\_on.asm   
        ; flash:       avrdude -F -V -c arduino -p ATMEGA328P -P /dev/ttyACM0 -b 115200 -U flash:w:led\_on.hex   
        ; description: turns led on port 13 on.   
        ;              this is an example from https://www.instructables.com/Command-Line-Assembly-Language-Programming-for-Ard/   
        ;              slightly modified for my own needs and coding style.   
        ;              It turns on the LED which is connected to PB5 (digital out 13).   
           
         .list   
           
         start:   
C:000000 e200           ldi r16,0b00100000         ;r16 = 0b00100000   
C:000001 b904           out 0x04,r16               ;out DDRB,r16   
C:000002 b905           out 0x05,r16               ;out PortB,r16   
         loop:   
C:000003 cfff           rjmp loop   
  
  
Segment usage:   
  Code      :         4 words (8 bytes)   
  Data      :         0 bytes   
  EEPROM    :         0 bytes   
  
Assembly completed with no errors.

### Turning LED on PB5 off

The build-in LED on PB5 can be turned off with: