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
```

The code

Here's the code to blink the built-in led (**led_on.asm**).
source:
https://www.instructables.com/Command-Line-Assembly-Language-Programming-for-Ard/

```
;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
;next line is commented.  If you like to use DDRB and PortB from the include file then
;comment out the next line.
;
;.include "./m328Pdef.inc"

.list

start:
       ldi r16,0b00100000         ;r16 = 0b00100000
       out 0x04,r16               ;out DDRB,r16
       out 0x05,r16               ;out PortB,r16
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.
```