ECE 375 Lab 1

Introduction to AVR Tools

Lab Time: Thursday 4-6

Bradley Heenk
Aaron Vaughan

TA Signature

1 Introduction

Text goes here

2 Internal Register Definitions and Constants

Text goes here

3 Interrupt Vectors

Text goes here

4 Program Initialization

Text goes here

5 Main Program

Text goes here

6 A Subroutine

Text goes here

7 Stored Program Data

Text goes here

8 Additional Questions

1. Go to the lab webpage and download the template write-up. Read it thoroughly and get familiar with the expected format. What specific font is used for source code, and at what size? From here on, when you include your source code in your lab write-up, you must adhere to the specified font type and size.

   (a) The code should be in mono-spaced font and can go down to 8-Pt font to make it fit. When copying the code over it can sometimes get messed up so make sure to clean it up so it looks readable.

2. Go to the lab webpage and read Syllabus carefully. Expected format and naming convention are very important for submission. If you do not follow naming conventions and formats, you will lose some points. What is the naming convention for source code (asm)? What is the naming convention for source code files if you are working with your partner?

(a) When naming your .asm file you need to include you and your partners name if you worked together. For example the naming convention needs to be"**First name_Last name_Lab4_sourcecode (e.g. Bradley_Heenk_Lab1_sourcecode.asm)**"

3. Take a look at the code you downloaded for today's lab. Notice the lines that begin with .def and .equ followed by some type of expression. These are known as pre-compiler directives. Define pre-compiler directive. What is the difference between the .def and .equ directives? (HINT: see Section 5.1 of the AVR Starter Guide).

   (a) Pre-compuler directives are defined as a list of special instructions that are executed before the code is even compiled and it directs the compiler. The .DEF is a way to set a symbolic name on a register and the .EQU is to se a symbol equal to an expression. For example we can define .DEF Symbol = Register which can have symbolic names attached to the register to make the coding process easier. On the other hand for the .EQU instructions which assign a value to a label. For example we can run the command .EQU label = expression. This is where a label assigned to a value by the EQU directive and is indeed a constant and cannot be changed.

4. Take another look at the code you downloaded for today's lab. Read the comment that describes the macro definitions. From that explanation, determine the 8 bit binary value that each of the following expressions evaluates to. Note: the numbers below are decimal values.

   (a) $(1 \ll 3)$

   (b) $(2 \ll 2)$

   (c) $(8 \gg 1)$

   (d) $(1 \ll 0)$

   (e) $(6 \gg 1|1 \ll 6)$

   (a)

5. Go to the lab webpage and read the AVR Instruction Set Manual. Based on this manual, describe the instructions listed below. ADIW, BCLR, BRCC, BRGE, COM, EOR, LSL, LSR, NEG, OR, ORI, ROL, ROR, SBC, SBIW, and SUB.

   - ADIW ADIW falls under the category of Arithmetic and Logic Instructions where these instructions take use of the microcontrollers ALU. AWIW falls under the Addition category based on the table we can see the AD followed by I which is immediate and W which is for the word operation which is a 16-bit operation
   - BCLR BCLR falls under the category of Bit Manipulation where these instructions allow the programmer to manipulate individual bits within a register. BCLR will set and clear respectively any bit in any I/O register.
   - BRCC BRCC falls under the category of Conditional Branches this specific case is when the test case $R_d \geq R_r$ and the boolean is C = 0.

- BRGE BRGE also falls under the category of conditional branches and this specific case is when the boolean equation will be $Z + (N \oplus V) = 1$.
- COM COM falls under the category of arithmetic and logic instructions meaning the compliments.
- EOR EOR also falls under the arithmetic and logic instructions under the category of logic
- LSL LSL falls under the category of shift and rotate which stands for Logical Shift Left to shirt bits around in a register.
- LSR LSR is also under the same category as LSL where bits are shifted around in the register expect LSR means Left Shift right instead of shifting it left.
- NEG Also seems to fall under the category of Arithmetic and Logic instructions where it checks to see if the number is negative
- OR OR Is under the arithmetic and logical operations meaning to use the OR operator in the ALU.
- ORI ORI is also under the arithmetic and logical operations but the I stands for Immediate value that is passed as the second argument
- ROL ROL is under the shift and rotate category where we are going to rotate right through carry respectively
- ROR ROR is the opposite of ROL where we are going to rotate left through our carry respectively.
- SBC SBC falls under the arithmetic and logic instructions and under the category of Subtraction and the C at the end stands with carry.
- SBIW SBIW falls under the category of arithmetic and logical instructions under subtraction and the last two letters like the I and W, I stands for an operation that involves an immediate value that is passed as the second argument. While the W stands for the Word value meaning the operation is a 16-bit operation
- SUB Falls under the category as the same as SBIW and is the main SUB is for subtraction and is the main basic operation using the ALU in the micro controller.

9 Difficulties

Text goes here

10 Conclusion

Text goes here

11 Source Code

```
Source code goes here. It looks best if each line is no
more than 60 characters.
```