# SECTION ONE INTRODUCTION TO AVR DEVELOPMENT TOOLS

# SECTION OVERVIEW

**Complete the following objectives:**

- Connect your AVR microcontroller board to a TekBot (optional).
- Create a new Atmel Studio project.
- Download and compile the sample AVR Assembly source code.
- Understand how to run and operate the Universal Programmer / Teensy 2.0.
- Upload and run the sample program on the TekBots AVR microcontroller board.

# PRELAB

In labs to come you will be required to complete a prelab for each lab. The prelab will cover concepts and knowledge that is required for the lab. The prelabs are due at the beginning of your lab section each week. If you do not have your prelab done at the beginning of the lab period you will receive no credit for the prelab. Prelabs are to be submitted on paper, not by email. **For this lab, no prelab is required.**

# PROCEDURE

## Wiring Your TekBot

1. **Use of the Tekbot is optional for the ECE 375 lab**. If you choose to use your Tekbot, follow the following instructions to ensure it is properly wired to the mega128 board. Otherwise, skip ahead to the section entitled “Looking at some AVR Source Code”.
2. Take a look at the wiring diagrams available on the lab webpage. To make the wires for connecting your TekBot boards together you will need to use the ribbon cable that came with your kit with male headers soldered to the ends and a bit of heat shrink tubing on the connections. There is a short tutorial on the TekBots webpage that explains this process in detail.
3. Because we are working with a modular programmable AVR microcontroller board, you can connect the whisker inputs and motor controller outputs to any pins on any port on the board. But for the lab to work properly, these cables need to be connected in a certain way. This configuration should be used throughout the course. It uses the pin’s alternative functions rather than its primary functions. As the course progresses, you will learn what both of these functions are. Table 1 (next page) shows the proper pin connections.

| Connection | Port | Pin | Alternative Function |
|---|---|---|---|
| Right Whisker | D | 0 | External Interrupt 0 |
| Left Whisker | D | 1 | External Interrupt 1 |
| Right Motor Enable | B | 4 | PWM Output for Timer/Counter0 |
| Right Motor Direction | B | 5 | PWM Output A for Timer/Counter1 |
| Left Motor Direction | B | 6 | PWM Output B for Timer/Counter1 |
| Left Motor Enable | B | 7 | PWM Output for Timer/Counter2 |

Table 1: TekBots Connections to AVR Microcontroller Board

4. When you have completed the wiring, be sure to ask your TA to look it over before you turn the power on. If you have wired things incorrectly (especially if you have switched Vcc and ground), you WILL destroy parts of your TekBot. Please have the TA initial in the space below that you have shown your TekBot to them.

TA Initials: _____

## Looking at some AVR Source Code

1. Download the sample code available on the web page. This is a simple AVR assembly program that is well commented and ready to compile. All code that you produce should be as well commented as this code. Save this code where you can find it.

2. Atmel Studio is the Integrated Development Environment (IDE) that you will be using to develop your AVR assembly code throughout the remainder of the course (with the exception of Lab 2, which uses C). Atmel Studio is a powerful IDE created by Atmel for their line of AVR microcontrollers. You will be using it to write assembly programs for your AVR microcontroller board that uses an ATmega128 microcontroller. Section 2 of the AVR Starter Guide, which can be found on the lab website, contains a good overview on how to use the program as well as some step-by-step tutorials. Briefly read through this section to gain a basic understanding of the IDE.

   (Optional) If the IDE is not already installed on the computer, or if you are using your personal computer, you can download and install the latest version of Atmel Studio from atmel.com.

3. Follow the steps in Section 2.1.2 of the AVR Starter Guide to create a new project. In most IDE tools, a project is the base starting area to your program. It consists of all files you use and any settings for the program. When following this tutorial, you'll want to include the AVR assembly source code you downloaded from the lab website in step 1 into the project you have just created.
4. With the current project created in step 3, follow the Project Simulation tutorial in Section 2.1.3 of the AVR Starter Guide to learn how to compile and simulate your program. By the end of this step, you should know how to successfully create an AVR project from scratch and be able to compile it into usable program hex code.
5. When assembly source code is compiled, it creates a binary program file (called a HEX file with a .hex extension). This HEX file contains the actual binary instructions that are used by the ATmega128 and is what needs to be uploaded onto the AVR Microcontroller Board.
6. The next step is to program your board with the HEX file you just created from the sample code. The instructions for this step depend on which programmer device you have:
    - If you have the **old-style Universal Programmer** device, you can use the Universal GUI software, which should be already available on the lab machines. This is an open source program that can upload the AVR HEX programs to the various AVR microcontrollers. You simply connect the AVR board to your PC using the TekBots Universal Programmer, USB cable and 10-pin IDC cable. For more detailed instructions, please refer to Section 3 of the AVR Starter Guide.
    - If you have the **new-style programmer**, which is a Teensy 2.0 device programmed to act as an AVRISP mkII clone, you can program your board directly from Atmel Studio by following these steps:
        1. First, make sure that the programmer is connected and that your ATmega128 board is powered on. Click on **Tools -> Device Programming**. In the window that opens, click on the **Tool** dropdown list and select your AVRISP mkII programmer.
        2. Make sure that ATmega128 is the selected device, and that ISP is the selected interface, and then click the **Apply** button.
        3. In the sidebar that appears, click on **Memories**, then click on the "…" button next to the Flash dropdown list. Navigate to the .hex file that you created earlier by compiling the example assembly file, select it, and click **Open**.
        4. Make sure that the "Erase device before programming" and "Verify Flash after programming" boxes are both checked, and then click the **Program** button and wait for the process to complete.

7. With the program uploaded into the microcontroller, unplug the TekBot from the computer and turn it on. Observe the behavior; the TekBot should be performing a basic BumpBot routine. Demonstrate your TekBot to your TA.

TA Signature: _____

Theory of Operation for Lab 1 AVR Assembly Code

- Initializes key components of the ATmega128
- Starts the TekBot moving forward
- Polls the whiskers for input
- If right whisker is hit
    - Backs up for a second
    - Turns left for a second
    - Continues Forward
- If left whisker is hit
    - Backs up for a second
    - Turns right for a second
    - Continues Forward

# STUDY QUESTIONS/ REPORT

For all labs you will be required to submit a short write-up that details what you did and why. You must use the template write-up available on the lab webpage. See the webpage and template for specific details on what should be included in the lab write-up. You must submit a hard copy of your write-up and code to your TA by the start of the following lab. NO LATE WORK IS ACCEPTED.

Regarding the formatting of your write-up, it should be typed. You are responsible for turning in a clean, organized and professional document free of misspelled words. The code you turn in must include sufficient comments for ANOTHER STUDENT to be able to understand your code. *Code that is not well documented will be penalized severely*. If you are interested in the style and detail expected of your comments, look at the code you just downloaded. Generally you should have a comment for every line of code.

There is no write-up required for this lab, but you will turn in the answers to the study questions given below.

## Study Questions:

Most of the labs you do will have study questions that are to be answered within your lab write-up. This lab's study questions are given below and will be due at the start of lab next week. Although you will be exposed to some information that has not been covered in class, keep in mind that as a student accepted into pro school you will need to get into the habit of being pro-active. This involves reading ahead and preparing yourself before going to lab or class.

1. Go to the lab webpage and download the template write-up. Read it thoroughly and get familiar with the expected format. Specifically look at the included source code. What type of font is used? What size is the font? *From here on when you include your source code in your lab write-up you must adhere to that font type and size.*

2. 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 given on the lab webpage).

3. 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 of the following expressions. Note: the numbers below are decimal values.
    a. (1<<2)
    b. (2<<1)
    c. (4>>1)
    d. (1<<4)
    e. (6>>1|1<<6)

# CHALLENGE

Challenge problems are additional tasks that can be performed for the labs. By successfully completing a challenge problem you can get extra credit for your lab. To get credit for the challenge problem you must successfully demonstrate it to your TA as well as document it in your lab write up. Today's challenge question is given below:

1. You have learned to use a simple tool to download a program into your TekBot. Modify the program so the TekBot reverses twice as long before turning away and resuming forward motion. Demonstrate this to your TA and turn in a printed copy of your modified assembly program with your study questions.