RoboJackets Firmware Training Week 3 Lab Guide

Marine Maisonneuve, Logan Schick

October 8, 2020
v1.0

Contents

1 Background

1.1 Topics

The important topics being discussed this week in lab include datasheets, registers, and Pulse-Width Modulation (PWM).

1.2 Premise

The lab premise is to use a push button change the brightness of a LED. This system will use interrupts to change PWM registers to control the power output.

1.3 Pulse Width Modulation

PWM is a method of generating variable voltage output with a static source. By alternating between a low and high voltage, the average effect of a middle voltage can be achieved. The average output voltage is determined by the value of the "high time" (when the voltage is high) over the total time (the time the voltage is high and low).

In this case, the source of the high and low voltage is a specific pin and the time it's high and low is controlled by a timer. As seen in lecture, timers use a counter that counts up to a max value to For Fast PWM (the PWM mode we're using in this lab) the pin is set to a value (either high or low) when the counter is at 0 then the pin is set to the opposite when the timer reaches its max value (255) it then resets to 0 starting the process again.

1.4 ATMega328P Microcontroller

The Arduino Uno is based on ATMega328P microcontroller which has 3 timer setup controlled using timer registers. We can therefore look at the datasheet for the microcontroller to figure out the values for those control registers. To save a bit of time skip to Section 12-15 (specifically Section 15) which explains the timer you'll be controlling in the lab. To complete the lab the only required readings are the register descriptions for the timers, but other sections such as modes of operation will give a better understanding of what these timers can do.

1.5 Simulation

If you are using a simulation instead of the hardware, do not worry. The steps are exactly the same. Go to the TinkerCAD link and you will see the circuit that is a subset of the the hardware. The Arduino you see will be what you use, with the LEDs and buttons replicated as they would be on the actual board.

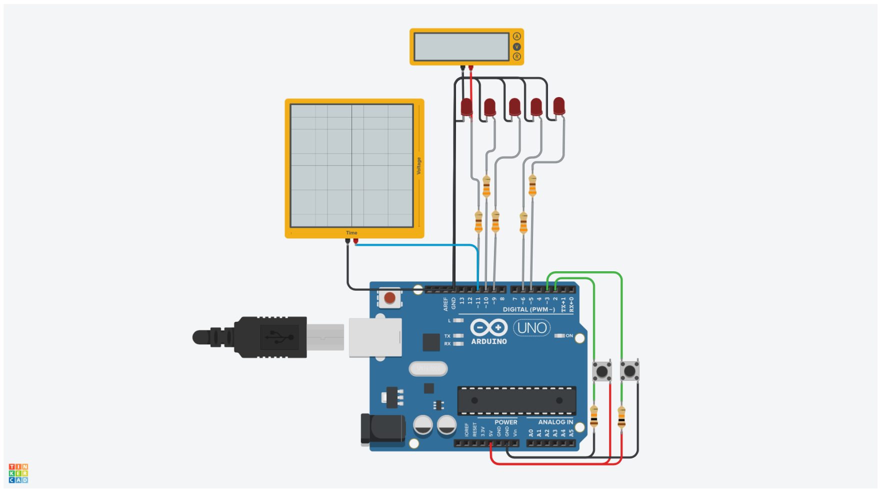
Figure 1: The circuit window of TinkerCAD for this project

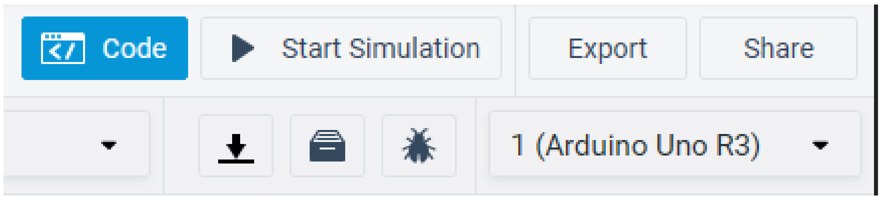

Figure 2: The area which you can use to select your target and compile

2 Materials

- AutoDesk Education Account
- TinkerCAD

3 Objectives

3.1 Task 1 - Setup

1. Use the datasheet section 15 to fill in the empty values in the `setup()` function.
    - You can directly set the values of these specific variables in binary, saving you the need to convert it.
    - To do that, use the `0b` notation for the binary values in C++. For example `int var = 0b101;` sets the value of the variable `var` to 5.
2. Fill in TCCR2A value.
    - The register description of TCCR2A is given in Section 15.11.1 of the datasheet.
    - Pin 11 (the pin the led, oscilloscope, and voltmeter are on) is controlled by OC2A which should be put in timer mode Fast PWM mode with the TOP being 0xFF. OC2B can be disconnected. The desired output compare mode of OC2A is either Clear on Compare Match or Set on Compare Match.
    - For information on setting the output compare mode, refer to Section 15.11.1 Table 15-3.
    - For information on setting the timer mode refer to Section 15.11.1 Table 15-8. Make sure to read the description of the table above it.

- A timing diagram for Fast PWM mode can be found in Section 15.7.3

3. Fill in TCCR2B value.
    - The register description of TCCR2B is given in Section 15.11.2 of the datasheet.
    - The prescaler value can be set to any as long as the clock isn't stopped. A higher prescaler leads to a lower frequency duty cycle, which will make the wave show up more clearly on the oscilloscope.
    - For information on setting the prescaler, refer to Section 15.11.2 Table 15-9.
4. OCR2A is the output compare register which controls when the output compare match triggers.
5. Composition of OCR2A is in Section 15.11.4.

3.2 Task 2 - Create Interrupt

1. Have the interrupt method increment the value that goes into the output compare register.
    - Note that when using fast PWM the 8 bit output compare register takes in a value from 0 to 255.
    - Values higher will overflow losing bits above 8, so you may want to account for that.
    - Set up the interrupt to trigger when the button is pushed.

4 Troubleshooting

4.1 Solutions

We have included the solutions below if you do not complete the lab during the session or if you want to verify your answer. If you need help during the lab ask an instructor!

- TinkerCAD Solution