Physics 364, Fall 2014, reading due 2014-11-09.

Email your answers to `ashmansk@hep.upenn.edu` by 11pm on Sunday

Course materials and schedule are at `positron.hep.upenn.edu/p364`

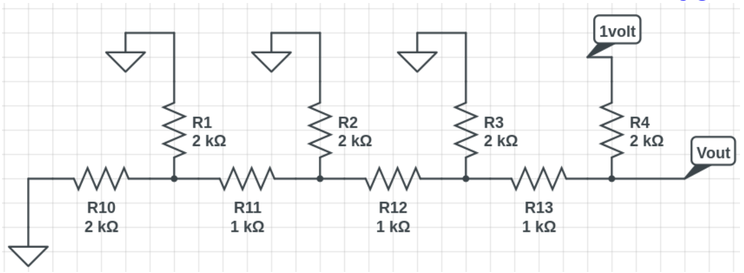
Assignment: (a) First read (or at least skim) whatever portion of Eggleston's chapter 8 (Digital circuits and devices, pages 200–233) you did not read last weekend. You might have stopped at page 220. (b) Then read through my notes (starting on next page), which this week are mostly a set of links to online materials that you should read. (c) Then email me your answers to the questions below.

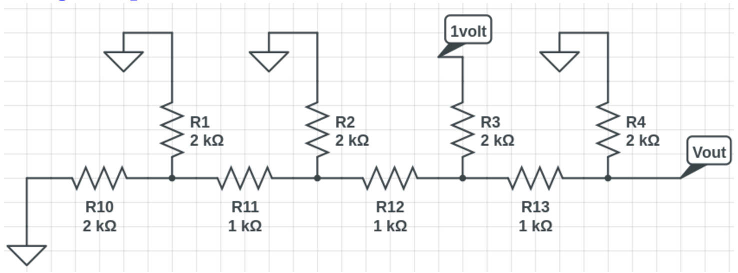
1. What are the names of the two functions one needs to write in order to make a working Arduino program? What is the purpose of each one of these two functions?

2. What does this little fragment of Arduino code do? In which of the two functions from Question 1 would you be more likely to find these few lines of code?

```
buttonState = digitalRead(buttonPin);
if (buttonState == HIGH) {
  digitalWrite(ledPin, HIGH);
} else {
  digitalWrite(ledPin, LOW);
}
```

3. The digital-to-analog converter you will build in Lab 21 includes a six-input verson of a clever circuit called an R–$2R$ ladder. I've drawn below a four-input R–$2R$ ladder. What is the voltage V_{out} in the two cases shown (left and right)? In each case, one input is fixed at +1 V and the other three inputs are fixed at 0 V. The left case is not hard to figure out; for the right case, I used a Thévenin equivalent for the half of the circuit to the left of R_{13}. If you're stuck, look at my CircuitLab models:
`www.circuitlab.com/circuit/bcw7vg/reading11-q3a/`
`www.circuitlab.com/circuit/858jgd/reading11-q3b/`





4. Is there anything from this reading assignment that you found confusing and would like me to try to clarify? If you didn't find anything confusing, what topic did you find most interesting?

5. How much time did it take you to complete this assignment?

Please read the Monday/Tuesday lab write-up (Lab 20), a draft of which is online at
positron.hep.upenn.edu/wja/p364/2014/files/lab20.pdf

Reading through this lab (draft) will help you to assess how carefully you need to read the Arduino background material. If you are relatively new to programming, then you may want to read the background material more closely than if you are already very comfortable with programming.

Then read the very brief *What is Arduino* and *Why Arduino* sections at
arduino.cc/en/Guide/Introduction

Look over a few examples from the "Basics," "Digital," and "Control Structures" sections of this page of Arduino code: arduino.cc/en/Tutorial/HomePage

I especially recommend looking at these examples:
arduino.cc/en/Tutorial/BareMinimum
arduino.cc/en/Tutorial/Blink
arduino.cc/en/Tutorial/Button
arduino.cc/en/Tutorial/Fade
arduino.cc/en/Tutorial/ReadAnalogVoltage
arduino.cc/en/Tutorial/IfStatement
arduino.cc/en/Tutorial/ForLoop

If you have never done any programming before, I think you will find that programming an Arduino is a fun and relatively unintimidating way to get your feet wet. In my opinion, if you're a science student nowadays, learning at least a little bit about computer programming is as important as learning calculus. So if programming is new to you, the next few labs are a painless way to to give it a try.

Next, read the Wednesday/Thursday lab write-up (Lab 21), which is currently online in draft form at positron.hep.upenn.edu/wja/p364/2014/files/lab21.pdf

After reading through Lab 21, you might want to look again at Eggleston's very brief coverage of DACs and ADCs, on pages 227–229. Also, this web page has a very brief but pretty good description of the three main techniques used for Analog-to-Digital Conversion: hyperphysics.phy-astr.gsu.edu/hbase/electronic/adc.html

To record 16 bits per sample (as is done for computer audio, `.wav` files, MP3, etc.), a digital-ramp ADC (also known as a Wilkinson ADC) would require only 1 comparator but needs $\mathcal{O}(2^{16})$ clock cycles to record each sample. A 16-bit successive-approximation (a.k.a. binary-search) ADC also requires only 1 comparator but needs only 16 clock cycles (one clock cycle per bit) to record each sample. A 16-bit flash ADC requires 2^{16} comparators but can record each sample in a single clock cycle. Flash ADCs are a big part of the cost of digital oscilloscopes, which typically acquire several samples per nanosecond (typicaly at just 8 bits per sample). Most medium-speed ADCs nowadays use a hybrid between the successive-approximation and flash approaches.