Electronics (Engn/Phys 208) Assignment 3
A Trip Down Memory Lane: FETs and Flash

Due date: 13 Feb 2018, end of business.

Vocab List:

1. Compose a poem, song lyrics, or other similar artistic expression using the following vocabulary list to explain basic operational principles of MOSFETs and Flash memory. You may make a video of your performance or do it live in class!

   FETs

   - n-type doping
   - p-type doping
   - depletion region
   - drain
   - source
   - gate
   - metal oxide layer/insulated gate
   - enhancement mode nMOS
   - pMOS

   Flash Memory

   - Hot carrier injection/quantum tunneling
   - Floating gate
   - threshold voltage
   - screening effect
   - read operation
   - write operation
   - flash erase
   - non-volatile memory
   - wear-leveling

Conceptual Questions:

1. What is the difference between a JFET and MOSFET? Specifically, what is different about their construction, and how does their operation differ?
2. What are the advantages of MOSFETS over JFETs and BJTs?
3. What's the difference between nMOS and pMOS?
4. Why are flash memory devices limited in lifetime? Explain why erase/write operations are damaging, but reads are not.
5. What are the (dis)advantages of NAND and NOR flash architectures?

6. What is the *fundamental* difference between flash memory and RAM?
7. In your own words, explain the function of flash memory used in a microcontroller?
8. In your own words, explain how RAM is used in a microcontroller; what's the difference between the ***stack*** and the ***heap***?

Practical Stuff: Measuring Flash and RAM Usage in MCUs

1. Write code that to enable your Arduino to solve the quadratic equation in the main `loop{}`. Start with the code skeleton provided on the course website. Your code should also compute the amount of time required for computations. The built-in `micros()` function is your friend here.
    a. How much Flash memory is used? How much RAM is required for static data variables?
    b. How much free RAM space is there? Does it change over time? This Adafruit tutorial is stellar, and details how to measure free RAM space.
    c. How much computation time is required to solve the quadratic equations + print all of the information to Serial on average?
    d. Printing strings is a very common operation, but can take up a lot of RAM. How much RAM can you free up using the famous F() macro (scroll to bottom). Modify your code and report the updated values for Flash and RAM usage.
    e. How much MCU time is spent on computing vs. printing to Serial? To find out, modify your code to print only the timestamp data (not the results of the quadratic solver or any other extraneous info). Compare and contrast the amount of MCU time spent on math operations vs. Serial.print(). Which is the fast operation? Which is the time/MCU hog?
2. Revise your quadratic solver code. *Start by copying and pasting your current code into a new sketch.* Instead of doing all the computations in the main `loop{}`, write a function called `solveQuad` that does all of the heavy lifting. It should take 3 float inputs (the coefficients a, b, and c. It should output an integer value of 0, 1, or 2 depending on the nature of the roots (double root, both real, complex conjugate pair). Essentially, you'll move the bulk of your code from `loop{}` into `solveQuad`.
    a. Report your findings in terms of Flash and RAM memory usage.
    b. Compare and contrast memory usage for both test cases you just performed, quadratic solver in `loop{}` vs. your custom function `solveQuad`.

What to submit: You should submit a clear and concise summary report of your results obtained from the above explorations in Arduino- (or Teensy-) land. A summary figure or table would be highly effective/highly advised. Please append your code in an appendix as well as a screenshots of it properly solving three test cases: (a,b,c) = (1, 0, 1); (1, 2, 1); (-1, 2, 3). Please also submit a copy of your code to the box folder. The filename name MUST have your Last Name appended, e.g. *QuadraticSolver_Erickson.ino*

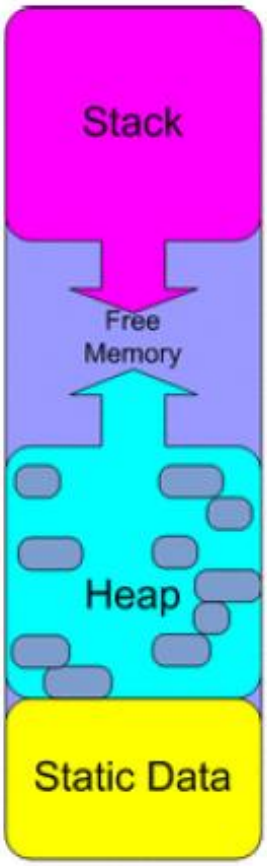

Figure 1. SRAM memory usage summary. Image credit: Adafruit.com. Great tutorial here, highly recommend you spend 30 min reading all of it.