Assignment 4: ADCs, Flash Memory, and I2C
ENGN/PHYS 208—Winter 2021
Due: 25 Mar 2021, 4pm

Background

The physical world 'speaks' (mostly!) ***analog***. For instance, acceleration and temperature are continuous aka analog *quantities*, taking on *any* value within a (sensible) range. By contrast, computers speak only in ***digital***. So we need some 'translators'. This what the role of analog-to-digital converters ***ADCs***[1].

Another core function of computers or microcontrollers (MCUs) is to "remember forever, even when the power goes out". This is the role of flash memory. Flash memory can be used to store photos, mp3s, documents or files of any kind. In the context of MCUs, the flash memory remembers the "instructions" the MCU must carry out upon power up or reset. If you program and Arduino, Teensy, or Feather once, then cycle the power, it still remembers the code you most recently uploaded!

Flash Memory

We discussed the basics of flash memory in class. It's a super clever technology that modifies a MOSFET to include a "floating gate".

1. **Back in a Flash: Cheaper than a hamburger**

   (a) In your own words, what is the purpose of the floating gate and why is it crucial for the design of flash memory?

   (b) The Samsung Evo 128 GB microSD flash memory card—for example, this model from Best Buy (linked here) is a typical modern day storage device. Given the tech specs—look for the "Specifications" tab on the product page—what is the cost per 1 GB? Specifically, is the cost of a gigabyte actually "cheaper than a hamburger"? Suggest another everyday item for price point comparison demonstrating just how cheap a gigabyte worth of flash memory costs in modern day terms.

   (c) Estimate the footprint of a single flash cell in units of nm^2. Estimate the number of silicon atoms that fit could be packed into that footprint? To compare to a human size scale, how much area (units nm^2) would the same number of M&Ms occupy?

   (d) Recall the flash memory worksheet from class: (linked here). Complete section 2, reporting only flash memory usage. Make a good graphic illustrating relative flash usage for each of the basic functions implemented (blink and LED; make an analog read; control a servo; etc).

[1]The inverse operation is converting a digital code back into a quasi-continuous quantity. This operation is and digital-to-analog converters (***DAC***. We won't study them formally in 208, but it is an easy conceptual extension

ADCs, I2C, and Sampling Theorem

1. **Getting Oriented**

   The brand new (as of this writing) Bosch BNO055 9-axis absolute orientation sensor is ridiculously awesome. And all for the low, low price of 20 bucks on adafruit.The BNO055 not only measures 9-axis IMU data (3 linear acceleration + 3 rotational motion + 3 magnetic field axes), it does a lot of fairly complicated math to optimally fuse this sensor information into absolute orientation. A dedicated Cortex M0 microcontroller does the math.[2].

Figure 1: Bosch BNO055 graphic highlighting measurements and output. Image credit: Bosch BNO055 datasheet. This is truly amazing technology at your finger tips, all for the low price of 20 bucks. Unthinkable just a decade ago!

   One key part from the datasheet about the device specs is quoted below.

   > *The BNO055 is a System in Package (SiP), integrating a triaxial 14-bit accelerometer, a triaxial 16-bit gyroscope with a range of ±2000 degrees per second, a triaxial geomagnetic sensor and a 32-bit cortex M0+ microcontroller running Bosch Sensortec sensor fusion software, in a single package...For optimum system integration the BNO055 is equipped with digital bidirectional I2C and UART interfaces.*

   Being smart and savvy Electronics consumers and designers, this raises several questions. Let's answer them!

   (a) How many different digital codes are there for the accelerometer measurements? For the gyroscope measurements?

   (b) What is the smallest difference in acceleration value that can be resolved using the accelerometer readings? Quote your answer in units of g/bit, where ($1g = 9.8\text{m/s}^2$). Assume the acceleration range is programmed for ±8g.

   (c) What is the ***resolution*** of the gyroscope angular velocity measurements, in units of deg/s/bit?

   (d) Imagine you are logging data for these 6 axes (a_x, a_y, a_z; ω_x, ω_y, ω_z). Each of these readings must be stored as an integer number of bytes. Let's say you are recording data at a sampling rate of 40 Hz. How big will your data file be after logging data for 1 hr? 24 hr? Are these data files "big" or "small". Compare them to a familiar file type (e.g. mp3 or jpg) to put this in context.

[2]For the math curious, check out what's happening under the hood, e.g., implementing the Madgwick filter. Warning: this is a fascinating piece of work, but is not the easiest bedtime read with quaternions and optimal estimators, oh my!

(e) The Bosch sensor uses the ***I2C digital communication protocol***. In terms of the I2C data transmission, sketch a timing diagram of SDA and SCL given that you are trying to read a_x, the acceleration along the x-axis. Hint: See the Bosch BN0055 data sheet sections 4.6 ("I2C Protocol") as well as the memory register map descriptions, specifically sections 4.3.9 and 10. The main point here is to impress the dear reader how amazing "just downloading libraries" is[3]. Someone had to do a LOT of work to make a single line of code like `accel.x()` properly transmit the bytes to actually read the acceleration value.

2. **Look into your heart:**

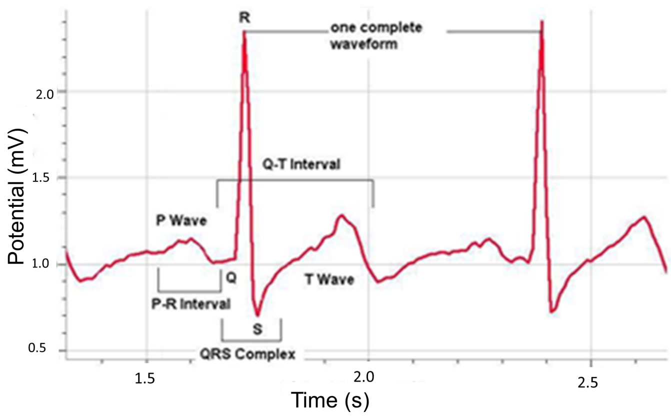

Figure 2: Electrocardiogram (ECG) signal. The P wave is generated contraction of the atria. The prominent QRS complex corresponds to contraction of the large, muscular ventricles. The T wave is formed when the ventricles repolarize (getting ready for the next heart beat).

Figure 2 shows an idealized electrocardiogram (ECG) signal, an electrical measure of heart activity.

(a) Measure the beat-to-beat intervals T_{beat} and corresponding frequency f_{beat}. Measure/approximate the timescale of the QRS complex T_{QRS}, and compute the equivalent (Fourier frequency) $f_{QRS} = 1/T_{QRS}$.

(b) Assume you have a 16-bit DAC with a sampling rate of $f_s = 2$ Hz. Carefully mark by hand the data points that will be taken, starting with the first sample at 1.5 s. Then connect them together to see what the recorded waveform will look like. Does the recording accurately represent the underlying ECG waveform? What features does it appropriately capture? Which ones does it miss?

(c) Repeat part a, but this time using a sampling rate of $f_s = 4$ Hz.

(d) What would select as a suitable sampling rate in order to accurately capture the QRS complex. Use your answer from part a to fully justify.

[3]E.g., this library provided by Adafruit `https://github.com/adafruit/Adafruit_BNO055`

3. **Music to our ears: A Good Resolution**

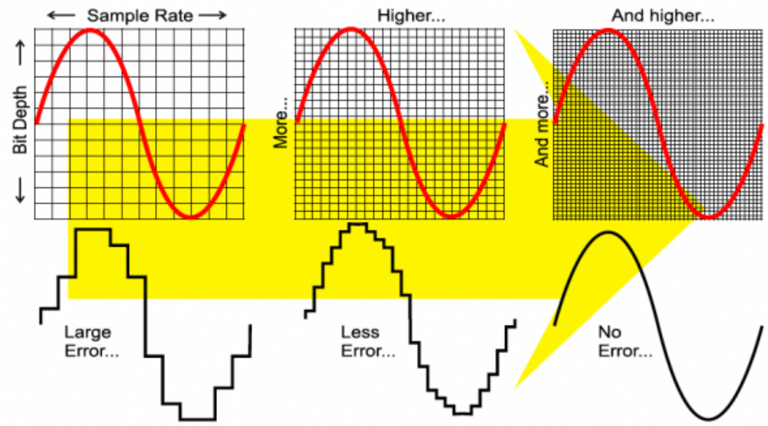

Figure 3: Sampling rate and resolution in audio recordings. Image credit: `https://www.izotope.com/en/learn/digital-audio-basics-sample-rate-and-bit-depth.html`

Last but not least, a real world consideration about music—who doesn't love a good tune. Digital audio recordings are almost always recorded at an industry standard sampling rate of $f_s = 44,100$ Hz.

(a) Explain why the 44,100 Hz standard sampling rate is a "good" choice. To help get you started, you should consider the human audible range of frequencies. The Nyquist frequency $f_N = 2f_s$ is important here. As well, you should consider file size and data storage.

(b) This sampling rate standard has been in existence since at least the early 1990s. A gigabyte of storage space wasn't exactly "as cheap as a hamburger" back then[4] Given modern day data storage, would you still argue for 44,100 Hz as a sensible trade-off or possibly argue for a new industry standard sampling rate? What practical considerations make higher sampling rates potentially less appealing? In any event, justify your answer based on quantitative rationale. Also, this article may be extremely useful; be sure to cast your answer in your own words.

4. ***Your move:*** Write a problem and solution about any topic we covered in Electronics this year that was particularly challenging for you. It would be anything from MOSFET operation to logic gates, to pull-up/down resistors to I2C communication or anything in between. This is basically an opportunity for you to reflect on a chunk of 208. As well, your crafted problem and solution will serve to help the instructor and future generations of Electronics students better navigate the crux of some of the difficult concepts.

[4] At the risk of dating myself, but for the sake of providing historical context: yours truly saved up a lot of weekly allowance to buy a 6 pack of optical disks, each of which stored 650 MB. In 1997, a 6-pack costs something like 70 bucks.