Tim Kanarsky

ECE 102 M6 – Power Consumption Measurement

Abstract: For Assignment M4, we created a wireless controller for a Pong game. My implementation of the controller publishes the IMU data as a BLE characteristic at a regular sampling rate, and a host computer running my Pong client can subscribe to changes in the characteristic to stream live motion data into the game. This remote is intended to be wireless, so the power consumption of the unit directly correlates with battery life. We will explore how changing the sampling rate affects power consumption of the wireless controller, and determine the most power-efficient sampling rate that nevertheless delivers a good user experience.

Question: What effect does varying the sampling rate of the wireless controller have on its power consumption?

Hypothesis: To a certain extent, increasing sampling rate will increase power consumption. However, it's possible that the power consumption increase will not be very significant compared to the baseline power draw of the Arduino.

Experimental Setup: The Arduino is loaded with the same firmware that was used in M4. Its VIN and GND pins are connected to the positive and negative terminals, respectively, of a desktop programmable power supply (in our case, the Rigol DP832). This power supply can talk with a computer over various protocols, and the computer can set its output voltage and current limits and measure the voltage, current, and power consumption of its output. The measurements update every ½ second, which is sufficient for our purposes as we are looking for an average power measurement.

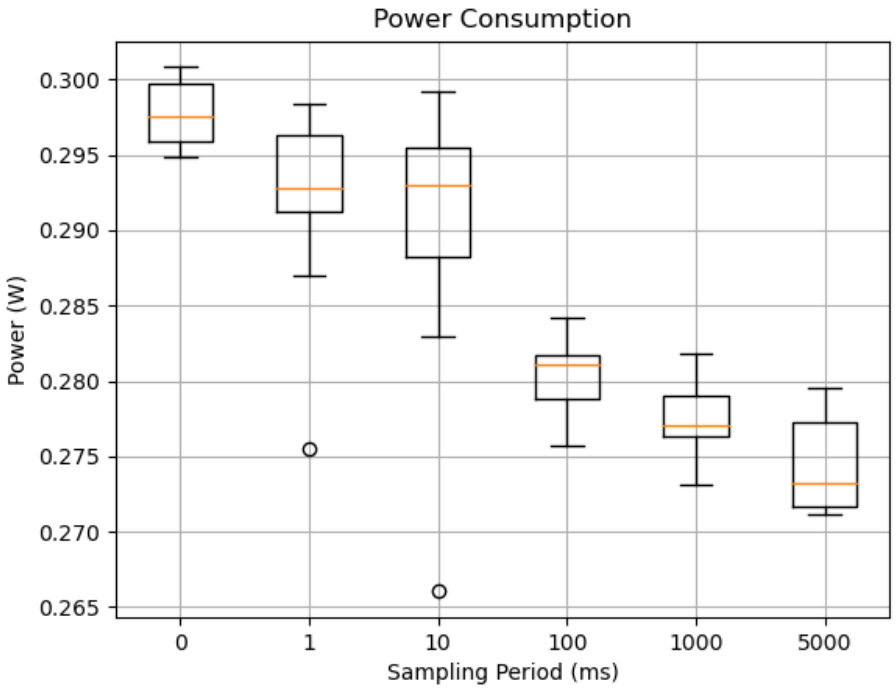
The power supply is connected to the control laptop with USB, which runs a Python script which performs the following:

- Sets the output voltage to 5V, and current limit to 500mA.
  - This is what the Arduino would normally receive from the USB connector. It is stepped down to 3.3V by a linear regulator, which introduces a bit of inefficiency. We tried powering it with 3.3V directly to the 3.3V pin, but the CPU would not run stably at that voltage and we did not want to risk breaking the Arduino by trying higher voltages.23
- Turns on the power supply, and waits a few seconds for the Arduino to boot.
- Connects to the Arduino via BLE, and subscribes to the IMU data characteristic to simulate the functionality of a real Pong client reading data as fast as it is published.
- Takes 15 seconds worth of samples. In our case, we measured 20 times a second so that we wouldn't run into aliasing issues with the ½ second measurement rate of the power supply.
- Find the average value of the samples, and write it to a file.
- Turn off the power supply, and wait 5 seconds for the computer to lose connection with the Arduino.
- Repeat this 10 times, resulting in 10 average power consumption measurements written to the file.

The M4 firmware used 100ms as its default sample period (a sample rate of 10Hz). We repeated the measurement procedure above with sample periods of 0ms (> 1000Hz, as fast as the Arduino can run), 1ms (1000Hz), 10ms (100Hz), 100ms (10hz), 1000ms (1Hz), and 5000ms (0.2Hz).

Results

We ended up collecting ~10 average power measurements over 15 seconds for 6 different sampling rates. The raw data is available in the GitHub repository. The data is summarized in a boxplot:



We see that increased sampling rate (decreased sampling period) has a positive correlation with increased power consumption. However, the increase plateaus when we sample faster than 1000Hz; in fact, letting the Arduino sample as fast as it is physically able results in a very slightly higher power consumption than limiting sampling to 1000Hz. Moreover, the increase appears to be linear with respect to the log of the sampling rate (because of the sampling rates we analyzed); therefore, the increase in power consumption is sub-linear with respect to sampling rate. Finally, the overall increase in power consumption is a relatively small fraction of the baseline power consumption. As sampling rate increases from 0.2 to 1000Hz, the median power consumption increases from 0.272 W to 0.292 W, a power consumption increase of 7.3% on a sampling rate increase of 5000%. We can e

Conclusions: Yes, we found that power consumption does increase with increased sampling rate. However, we noticed that this increase is 1) sub-linear w.r.t increased sampling rate 2) a small fraction of the overall power consumption, as there is still significant power usage when sampling incredibly slowly. Therefore, we should not worry about sampling rate's effect on power consumption, and set it to reasonable value that delivers a good user experience. Power consumption should be reduced in other ways, including finding more power-efficient electronics.