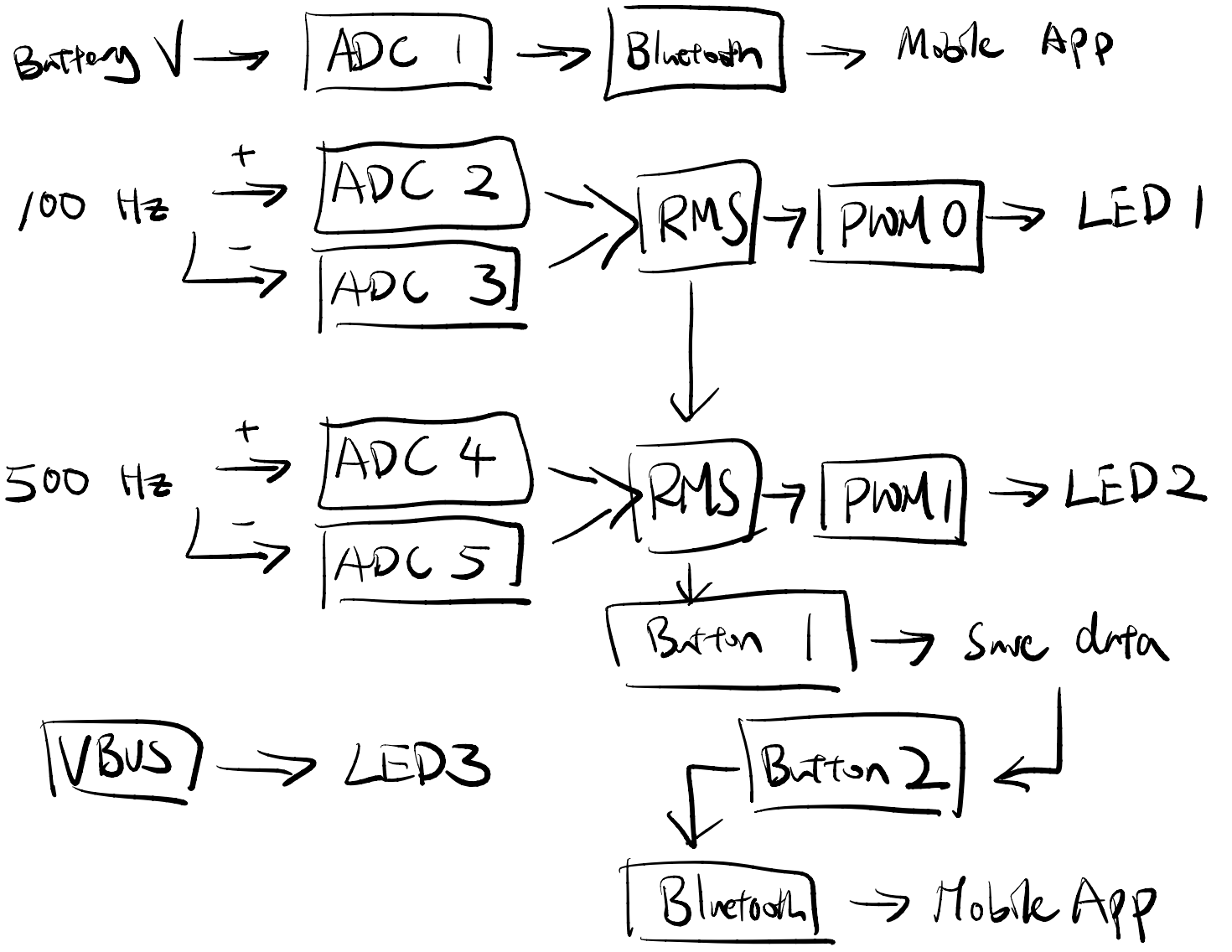

Battery V
ADC 1
Bluetooth
Mobile App
100 Hz
+
-
ADC 2
ADC 3
RMS
PWM 0
LED 1
500 Hz
+
-
ADC 4
ADC 5
RMS
PWM 1
LED 2
Button 1
Save data
VBUS
LED 3
Button 2
Bluetooth
Mobile App

Tim Ho
BME590L Final Project

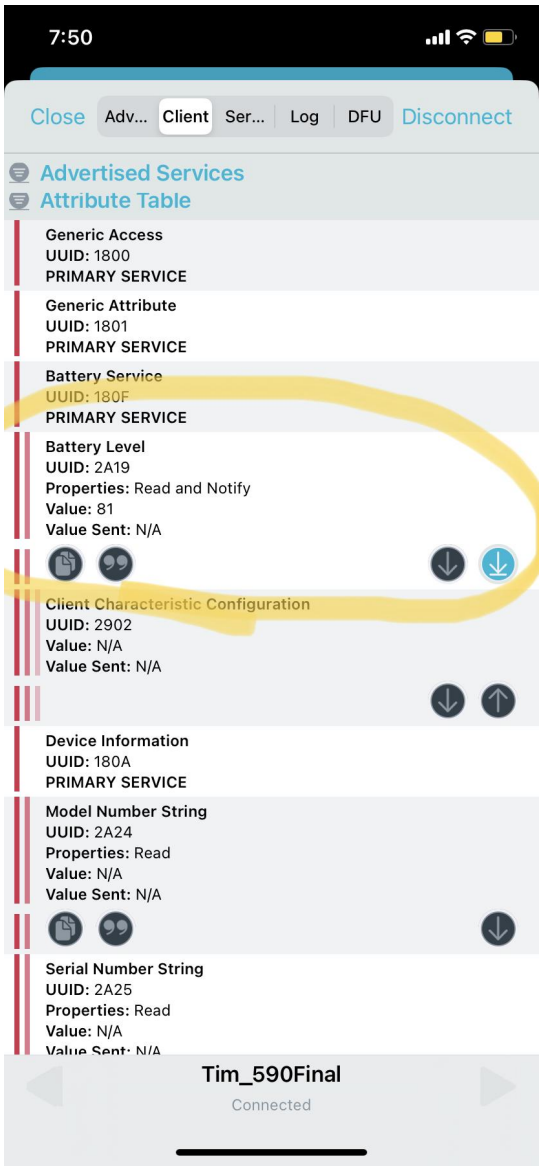
Verify the accuracy of your battery level measurement for 0-3.7 V.

The battery level measurement was relatively accurate (off by < +/- 0.007) for all data points except for 3.7 V. 3.7 V was an anomaly as the gain and reference voltage selection of 0.6 V and ⅛ gain, only allowed for a max measurement of 3.6 V. As a result, the 3.7 V data cannot be accounted for.

Voltage Input (V)	Voltage measured (V)
0	0.001
0.2	0.196
0.9	0.904
1.6	1.596
2.3	2.307
3	3.006
3.7	1.693

Verify that you can send those battery levels to the nRF Connect app using the Bluetooth Battery Level GATT.

The highlighted portion in the screenshotted figure below shows that battery levels were indeed able to be sent to the nRF Connect app. The value, “81”, that was sent was also cross-referenced in the nrf terminal, which corresponded to a normalized battery level of 81.67 (or 3022 mV raw).

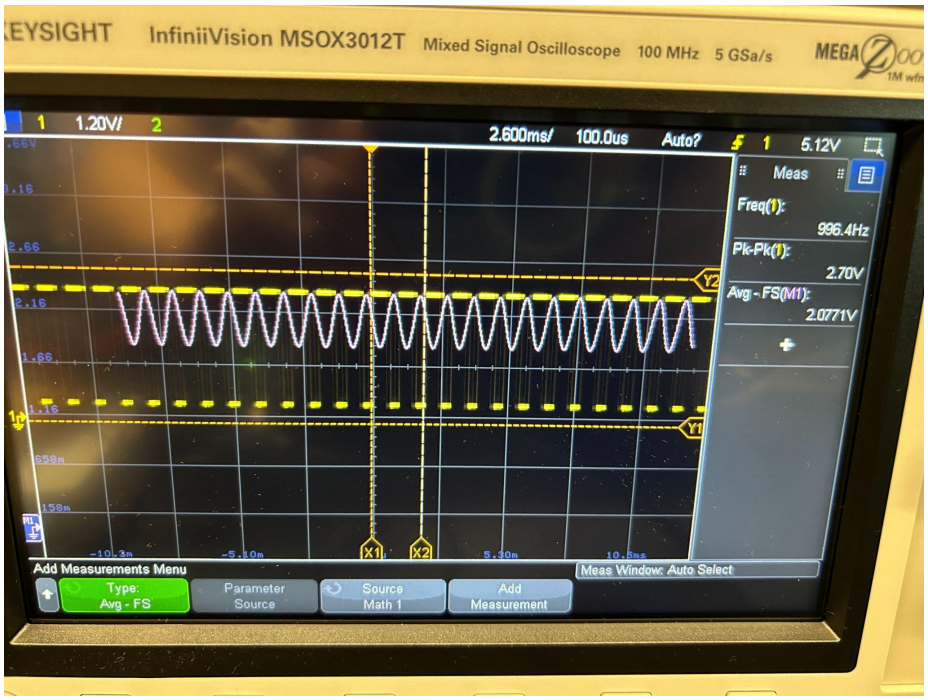



```
[00:00:03.886,108] <inf> finalproject: BT connected
[00:00:04.444,946] <wrn> bt_l2cap: Ignoring data for unknown channel ID 0x003a
[00:00:10.008,697] <dbg> bt: bluetooth_set_battery_level: Raw Battery Level: 3008
[00:00:10.008,728] <inf> bt: Normalized Battery Level: 81.297295
vbat is 3038 vbat is 3015 vbat is 3012 [00:00:14.389,801] <inf> bas: BAS Notifications enabled
vbat is 3055 vbat is 3026 [00:00:20.008,697] <dbg> bt: bluetooth_set_battery_level: Raw Battery Level: 3010
[00:00:20.008,728] <inf> bt: Normalized Battery Level: 81.351349
[00:00:30.008,697] <dbg> bt: bluetooth_set_battery_level: Raw Battery Level: 3007
[00:00:30.008,728] <inf> bt: Normalized Battery Level: 81.270271
[00:00:40.008,697] <dbg> bt: bluetooth_set_battery_level: Raw Battery Level: 3006
[00:00:40.008,728] <inf> bt: Normalized Battery Level: 81.243240
```

Using the function generator and the oscilloscope, make measurements varying the 100 and 500 Hz input signals linearly over their Vp−p ranges, and measure the corresponding LED brightness output for each input.

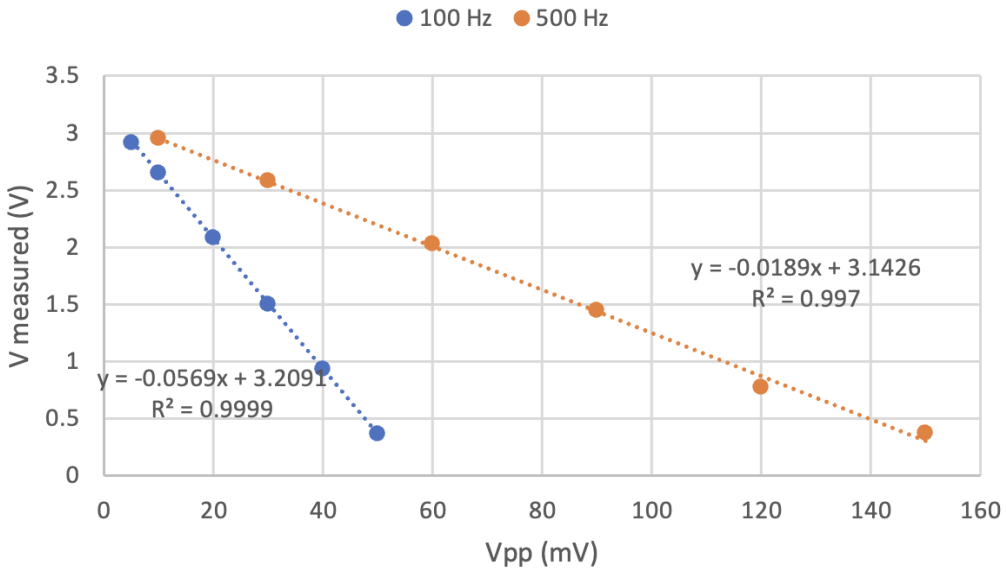
The LED brightness is measured by outputting the respective GPIO pin on the oscilloscope. The higher the voltage measured the lower the brightness and vice versa.

100 Hz		500 Hz	
Vpp (mV)	Vmeas (V)	Vpp (mV)	Vmeas (V)
5	2.9138	10	2.9507
10	2.6457	30	2.5828
20	2.084	60	2.0326
30	1.5011	90	1.4467
40	0.93337	120	0.77135
50	0.36443	150	0.37326



How linear is the relationship between Vp−p and PWM output? Quantify this relationship with linear regressions and associated R2 values.

Vp-p and PWM output has a strong linear relationship as shown through the high R^2 values for both signals. The high R^2 value of 0.9999 and 0.997 for 100 Hz and 500 Hz signals respectively shows that there is a high proportion of the variation in the dependent variable (PWM output) that is predictable from the independent variable (Vp-p).



Demonstrate that your 5 second data saving algorithm works by amplitude modulating your input sinusoids over those 5 second windows in a known manner to compare with your saved arrays.

The figure below shows an experiment done where the amplitude is modulated over a short 5-second window while the algorithm to save RMS values was running. Here the Vpp was changed from 10 mV to 50 mV. The numerical sequence shows the change of the 5-digit array over time. This experiment was run ten times (varying Vpp change each iteration) to verify that the algorithm indeed works.

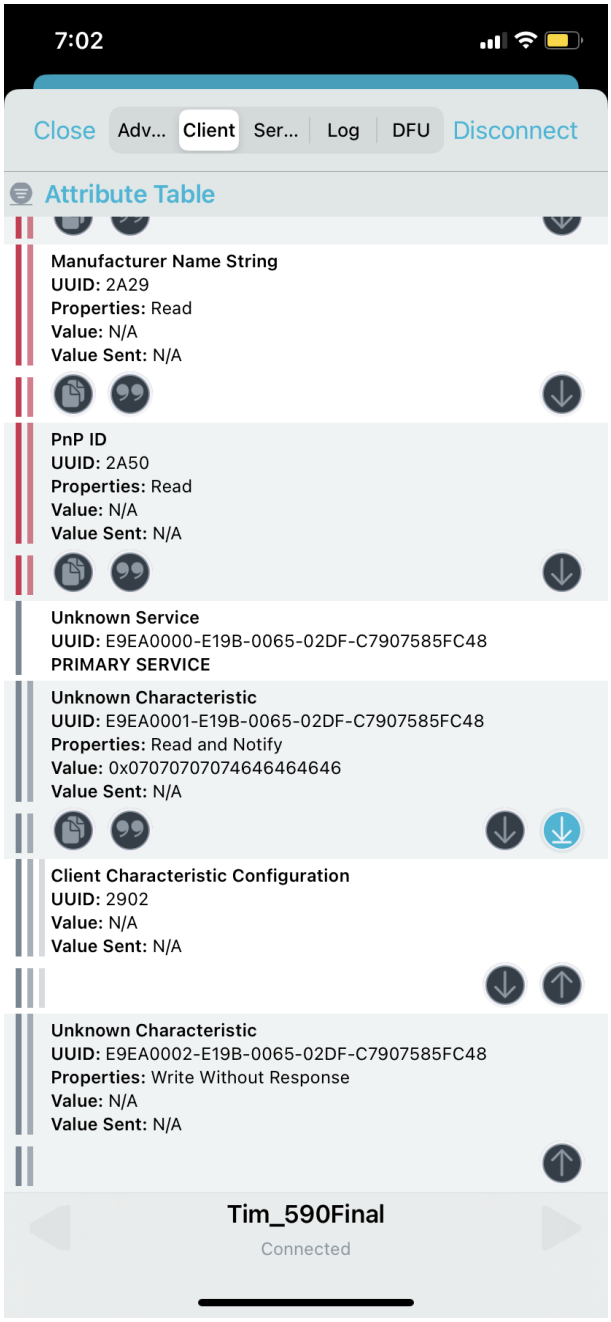
1s -> 7, 0, 0, 0, 0
2s -> 7, 7, 0, 0, 0
3s -> 7, 7, 11, 0, 0
4s -> 7, 7, 11, 31, 0
4s -> 7, 7, 11, 31, 35

```
[00:00:00.004,211] <inf> bt: Initializing Bluetooth
[00:00:00.004,302] <inf> sdc_hci_driver: SoftDevice Controller build revision:
                                         6d 90 41 2a 38 e8 ad 17  29 a5 03 38 39 27 d7 85 |m.A*8... )..89'.
.
                                         1f 85 d8 e1                                     |....

[00:00:00.007,263] <inf> bt_hci_core: No ID address. App must call settings_load()
[00:00:00.013,641] <err> bt_gatt: Failed to save Database Hash (err -2)
7 0 0 0 0 7 7 0 0 0 7 7 11 0 0 7 7 11 31 0 7 7 11 31 35 *** Booting Zephyr OS build v3.2.99-ncs1 ***
```

Demonstrate the ability to receive your two data arrays on the nRF Connect app.

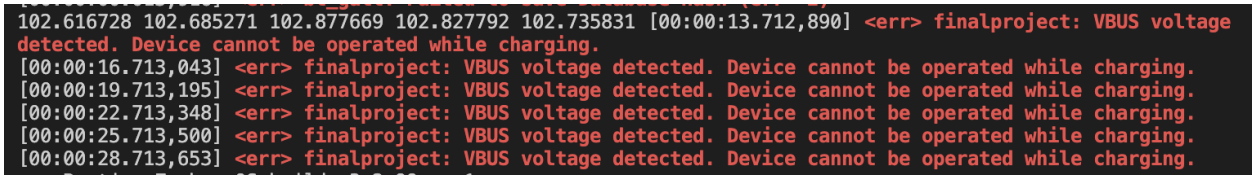
The phone screenshot below shows that the concatenated data array (length 10) is indeed transmitted to the phone). This is verified on the nrf terminal (the next figure), where the notification was indicated. The RMS can also be verified on the figure, where each index of the two arrays is side by side. The hex 0x07 is 7 in decimal and 0x46 is 70.




```
[00:00:00.226,135] <inf> finalproject: BT connected
[00:00:00.795,593] <wrn> bt_l2cap: Ignoring data for unknown channel ID 0x003a
[00:00:05.475,585] <inf> bas: BAS Notifications enabled
[00:00:10.500,457] <inf> bt: Notifications: enabled
[00:00:10.500,488] <inf> finalproject: BT notifications enabled
7.337983 70.852317 7.285534 70.868150 7.451577 70.906136 7.440564 70.946990 7.344522 70.851786 [00:00:22.81
5,032] <dbg> bt: set_compliance_data: Compliance data set via memcpy (size = 10).
[00:00:22.815,124] <inf> finalproject: BT data transmitted.
[00:00:22.875,152] <inf> bt: Notification sent on connection 0x200020b8
```

Demonstrate the safety feature of your device to cease function and blink LED3 when VBUS is HIGH.

The figure below shows that VBUS was indeed detected. On the board, LED3 starts blinking at 1 Hz and LED1 and LED2 both cease operations.



If your device wasn't limited to 100 and 500 Hz inputs, what is the maximum input frequency that your device can support without aliasing? Answer this question from a theoretical perspective and experimentally.

My device is sampling the analog signal every 1 ms, which corresponds to 1000 Hz. According to Nyquist frequency theorem, the maximum frequency signal my device can sample is half of the sampling frequency, or 500 Hz. Therefore, the maximum input frequency my device support without aliasing is 500 Hz.

Experimentally, the calculated Vp-p from the measurements and the actual Vp-p was compared to test aliasing. The experimentation shows that the actual maximum frequency the device can support is higher than 500 Hz or even 1000 Hz. This was unexpected as it was assumed that Nyquist frequency theorem holds. This could be due to the way the algorithm is implemented. Because the sampling was not implemented with a timer at fixed intervals, the implementation interval may vary over time, resulting in unstable sampling, which actually could bypass the aliasing problem by random chance. Another possibility is that comparing Vp-p is not an ideal experiment, as the sliding window range is too large, which allows for random chance to dictate whether sampling quality.

100 Hz			500 Hz		
Freq Signal (Hz)	Vp-p (mV)	RMS calculated Vp-p (mV)	Freq Signal (Hz)	Vp-p (mV)	RMS calculated Vp-p
600	50	49.886	600	150	172.696
700	50	51.754	700	150	204.614
800	50	49.983	800	150	173.897
900	50	49.869	900	150	172.552

1000	50	49.732	1000	150	172.490