I/O System for the PSYONIC Advanced Bionic Hand

Team 28
Byron Hopps and Steven Sun
ECE 445 Senior Design
Fall 2017

Background

- PSYONIC is a startup at the University of Illinois working on bringing low-cost prosthetics to third-world countries
- Their product is the PSYONIC Advanced Bionic Hand
- Current prototypes contain all necessary functionality
- User interface for core functionality such as battery charging and hand configuration are lacking
- To improve these interfaces, we created the I/O Board

Battery Charging Procedure

Without the I/O Board

1. Disassemble the hand
2. Remove the battery
3. Charge the battery with a lithium-ion battery charger
4. Wait for the battery to charge
5. Put the battery back in the hand
6. Reassemble the hand

Battery Charging Procedure

With the I/O Board

1. Attach hand to a USB charger

2. Wait for hand to charge

Hand Configuration Procedure

Without the I/O Board

1. Disassemble the hand
2. Change configuration constants in the source code
3. Recompile the hand’s code
4. Attach JTAG programmer to hand
5. Reprogram hand with updated code
6. Reassemble the hand

Hand Configuration Procedure

With the I/O Board

1. Connect to hand with Bluetooth or USB
2. Use smartphone app or desktop software to reconfigure hand settings

High-Level Requirements

- The I/O System shall be capable of powering the prosthetic hand from both an external power source and an internal battery of at least 2.2Ah

- The I/O System shall be capable of communicating with external devices using Bluetooth or USB

- The I/O System shall be capable of sending commands and data to the EMG board at a baud rate of at least 115200 symbols per second

Hardware

Byron Hopps

I/O Board Interfaces

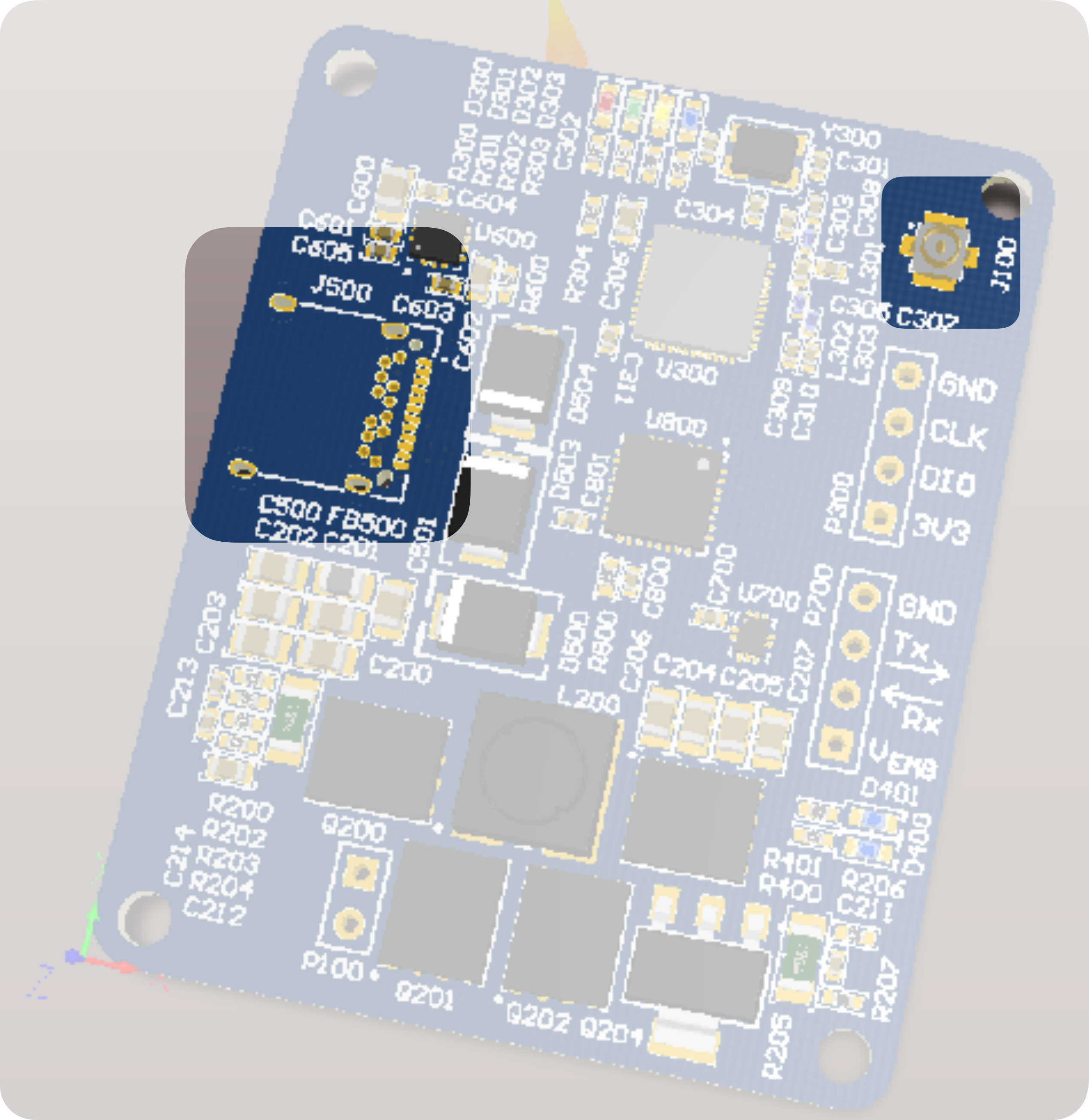
This section is done, all it needs is actual board pictures instead of the Altium board renderings

- Wireless communications using Bluetooth Low Energy

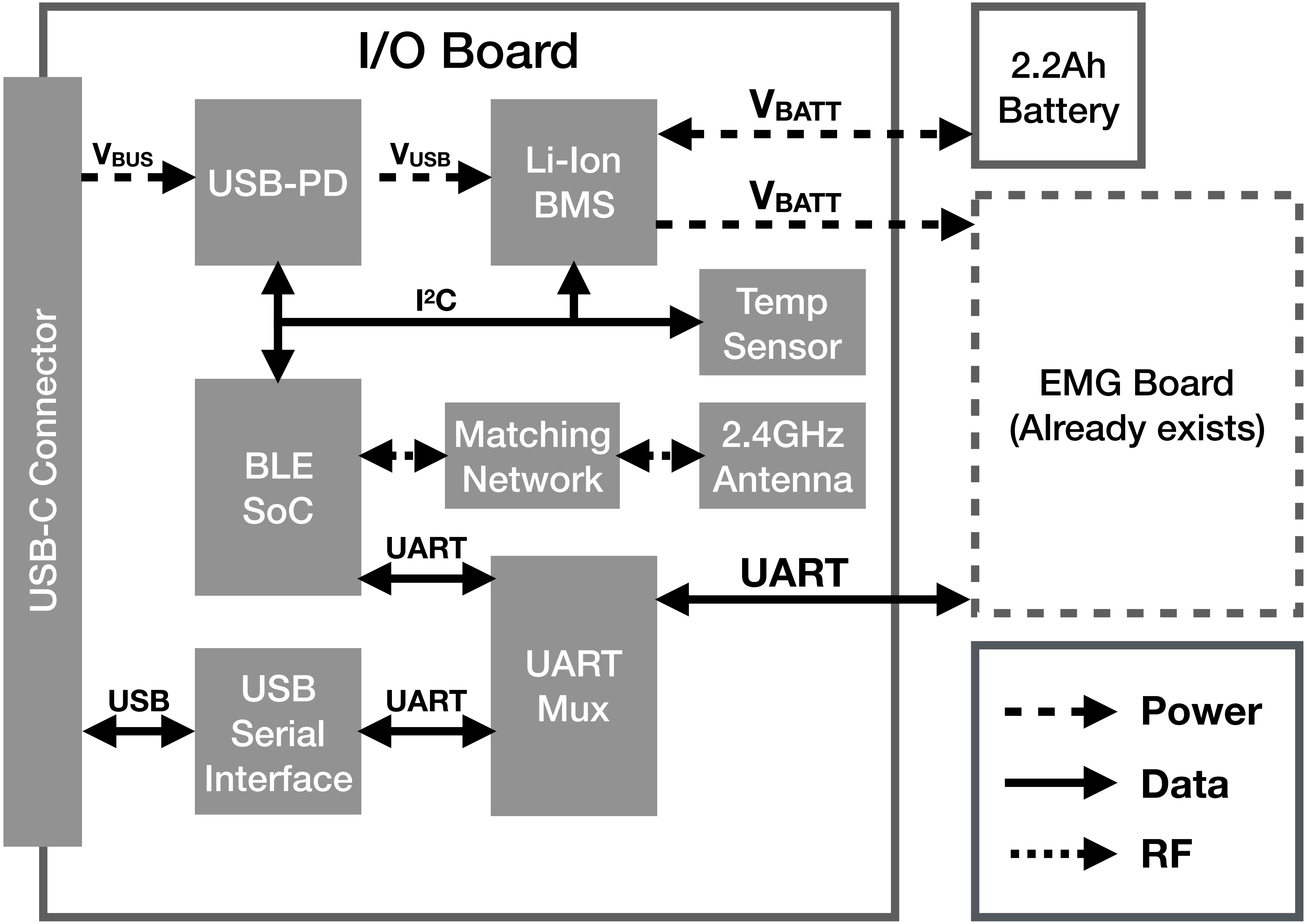
- USB Type-C Port

    - Wired communications using USB 2.0

    - Power Delivery via USB Power Delivery 2.0



System Architecture



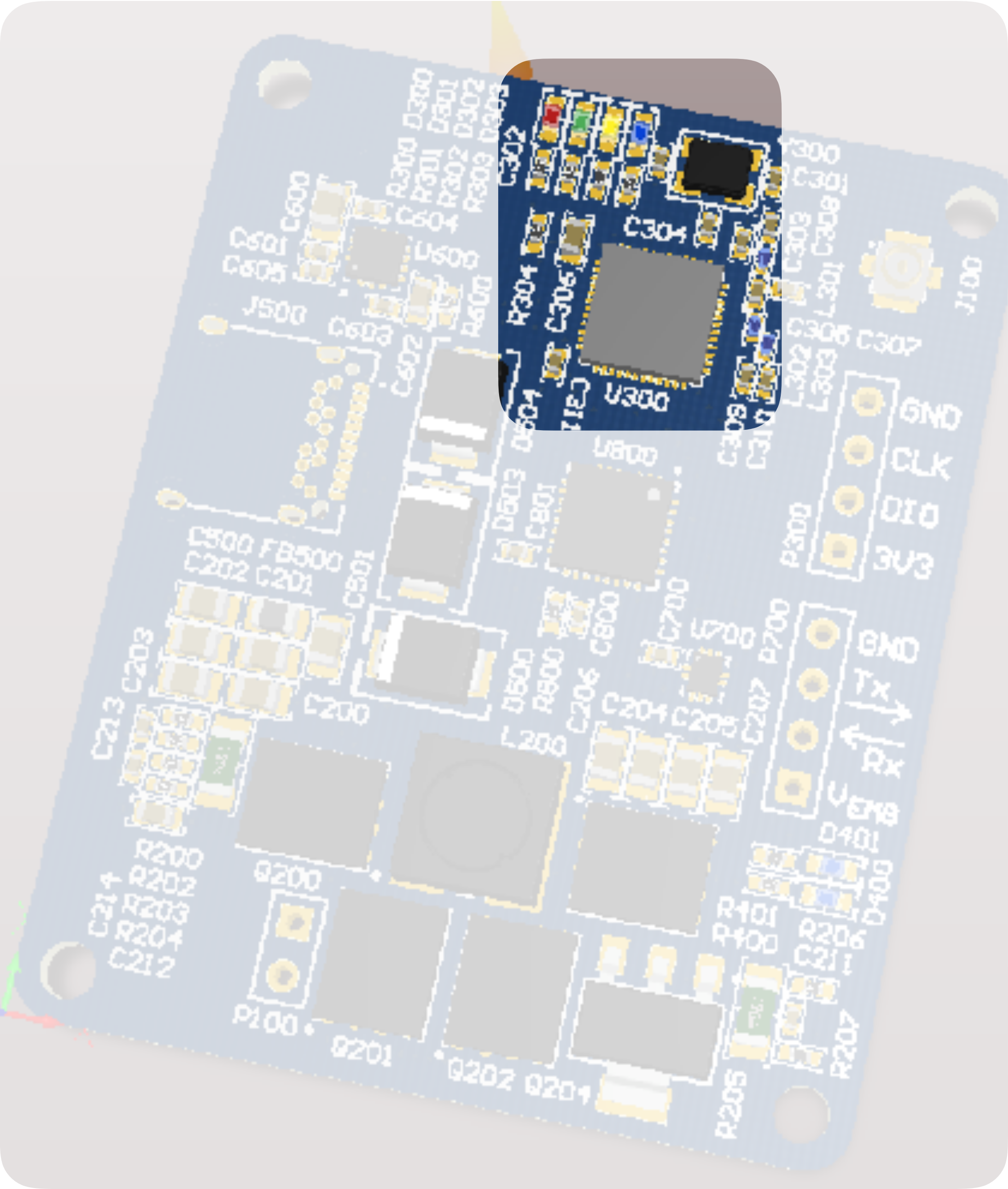
Bluetooth Module

- The Bluetooth Module is the brains of the I/O Board

- An embedded Microcontroller allows it to execute code

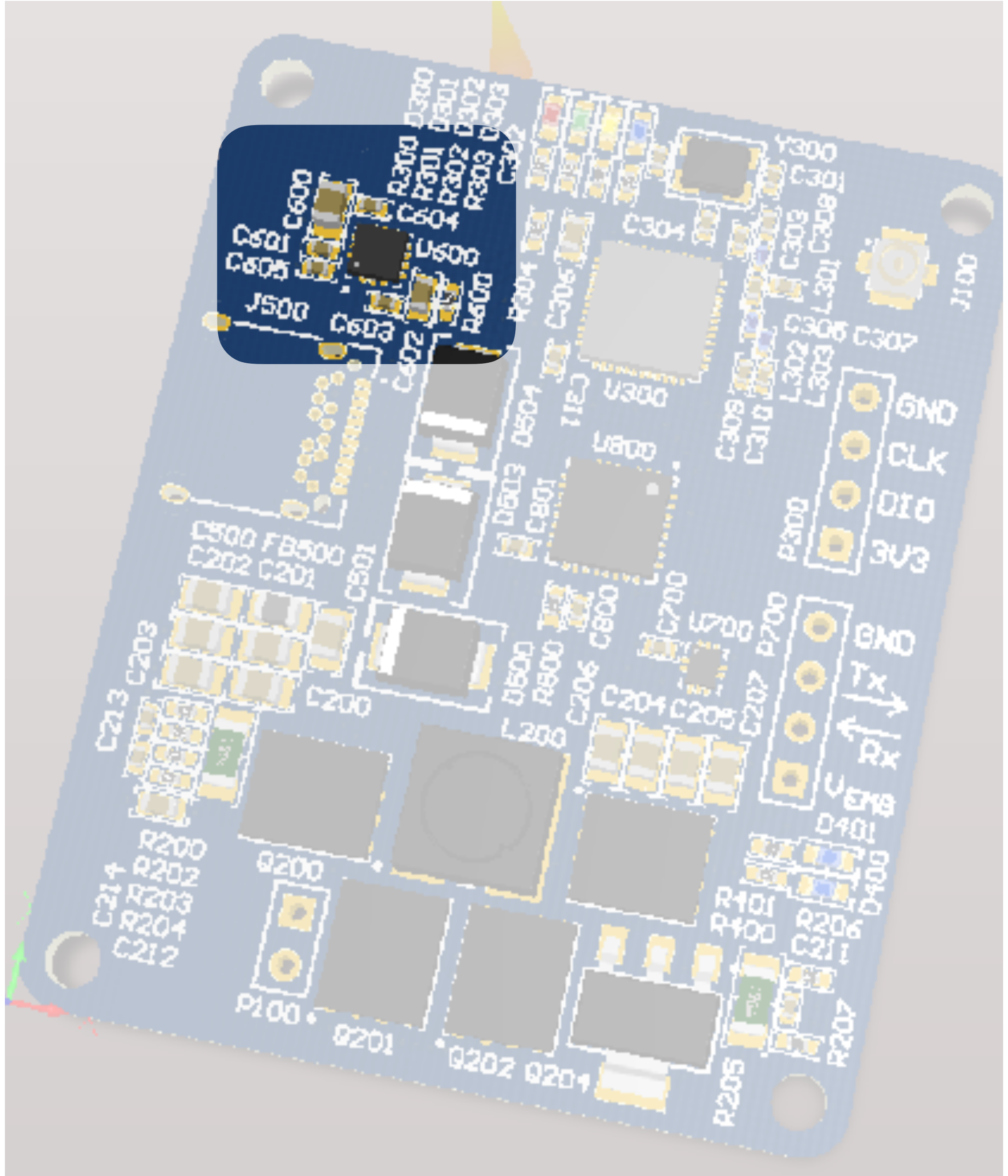
- A Bluetooth Low Energy radio gives it very efficient wireless communications

- It controls the other peripherals on the I/O Board



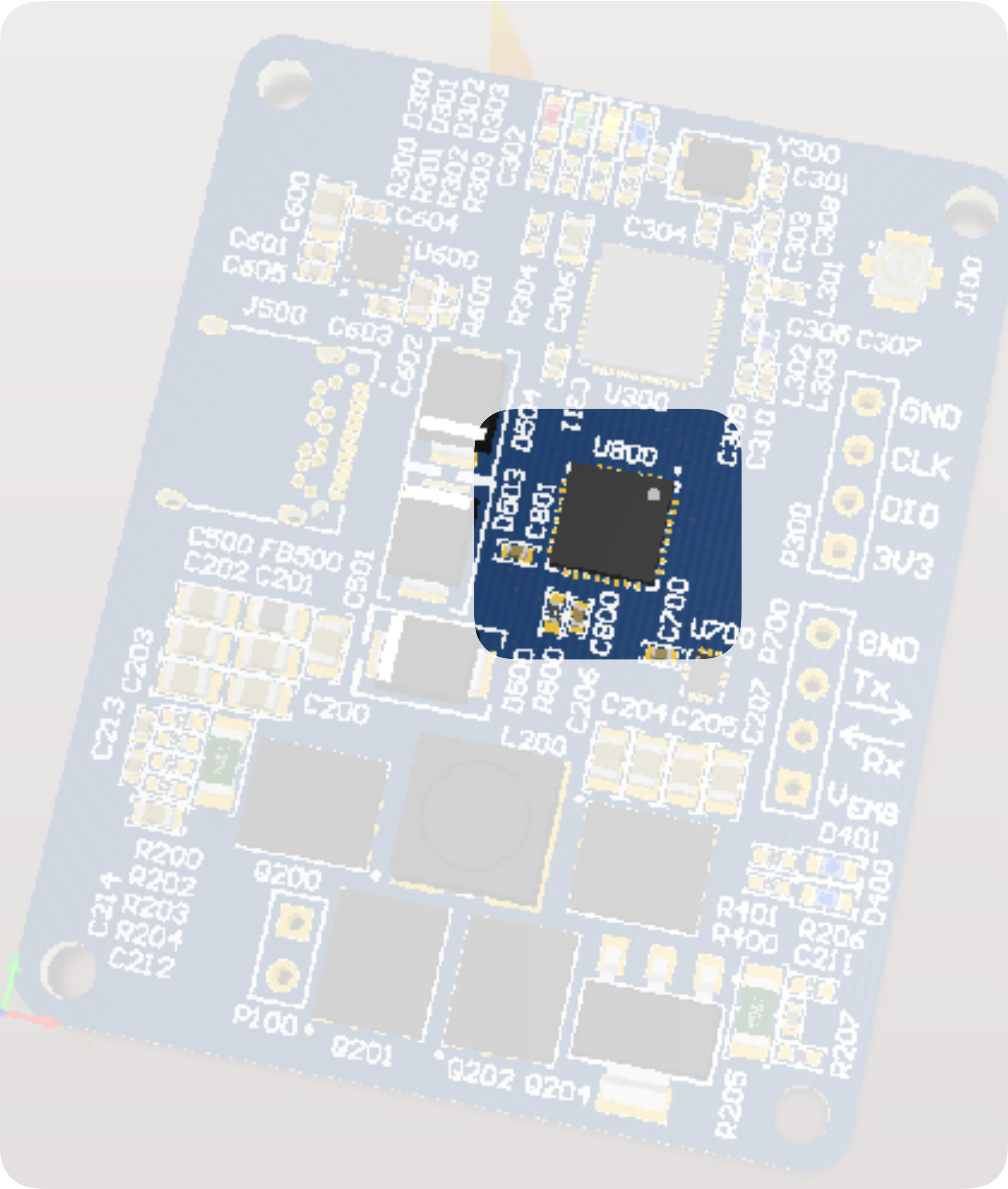
USB Power Delivery

- The USB Power Delivery Negotiator allows the I/O Board to draw more power from USB chargers
- More input power reduces battery charging times
- Is controlled by the Bluetooth Module



USB Serial Interface

- The USB Serial Interface allows the converts USB data to a serial format
- Allows the I/O Board to communicate over USB

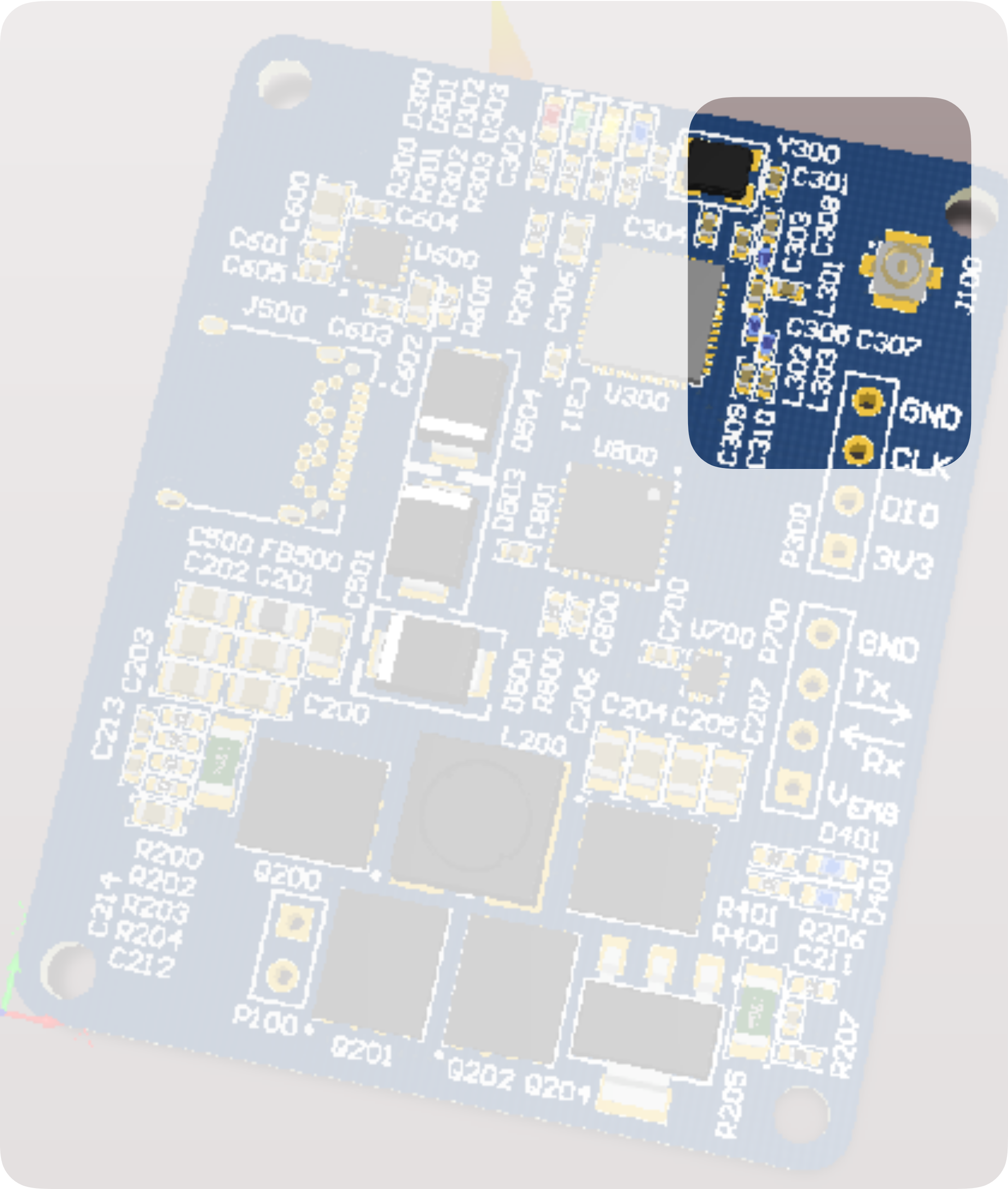


Radiofrequency Front End

- The Radiofrequency Front End allows the Bluetooth module to drive an antenna

- Consists of a lumped-element matching network

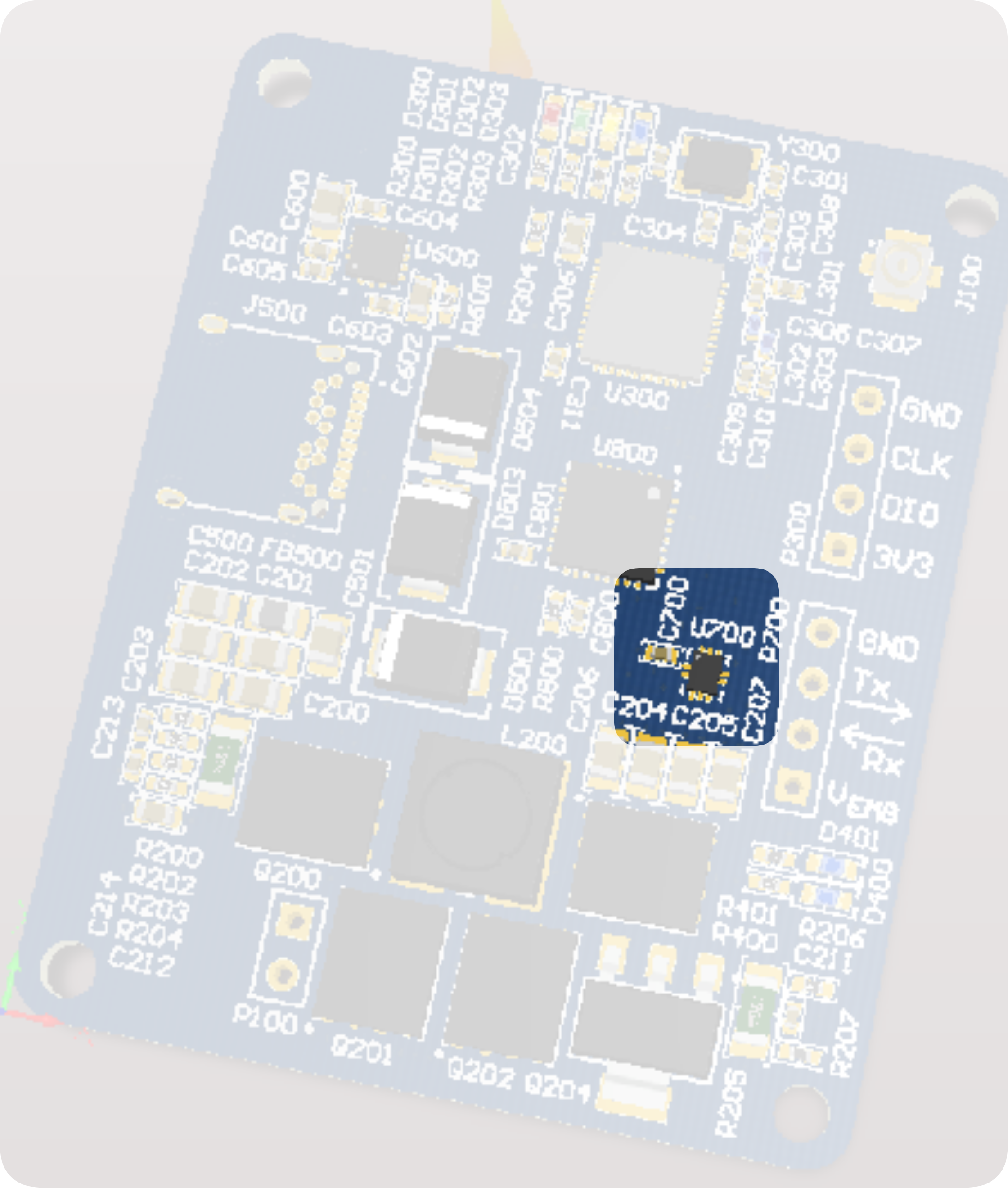
- An external antenna is connected to the USC jack



Serial Multiplexer

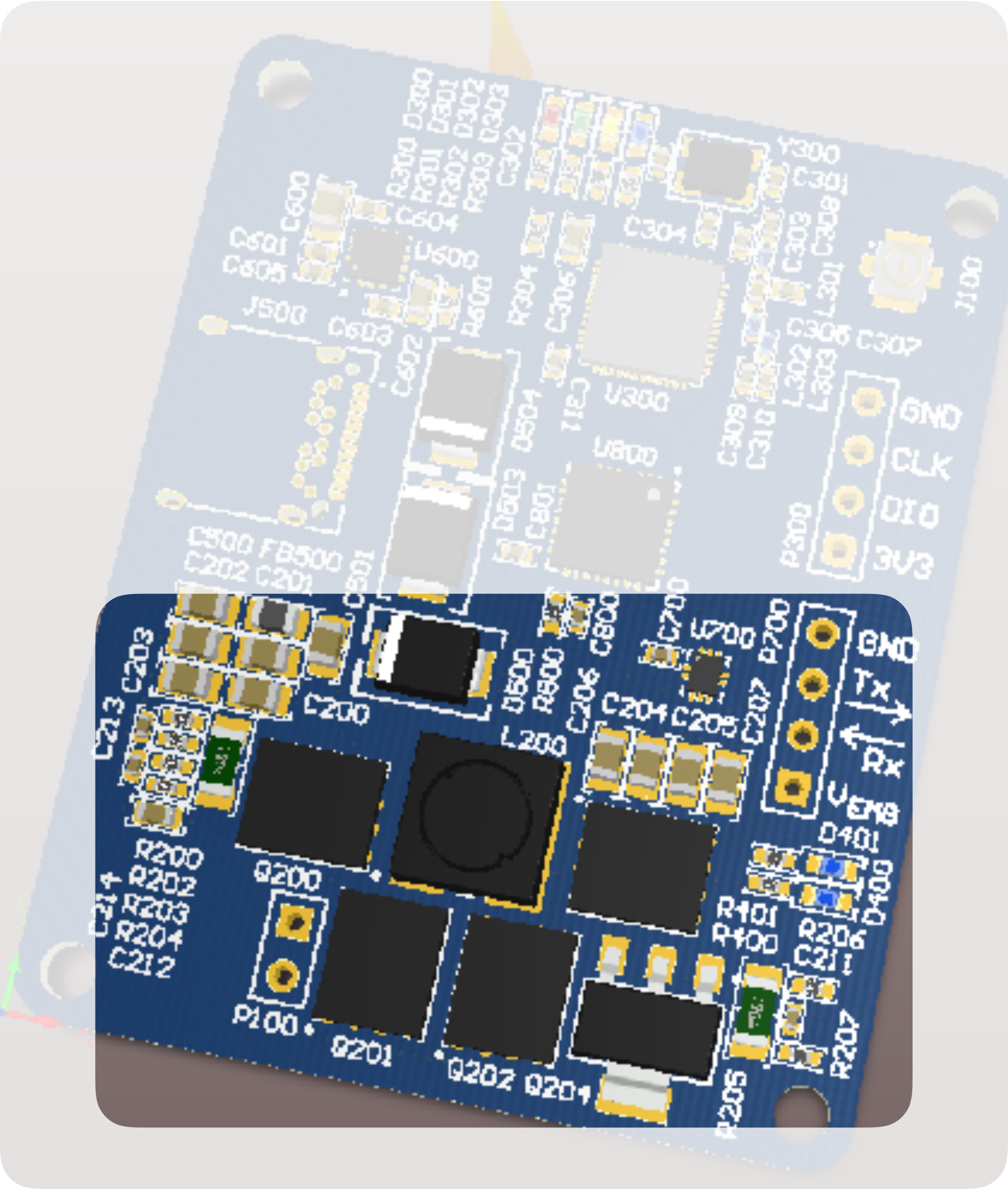
- The Serial Multiplexer allows the I/O Board to select whether the USB or Bluetooth interface controls the hand

- Is controlled by the Bluetooth Module



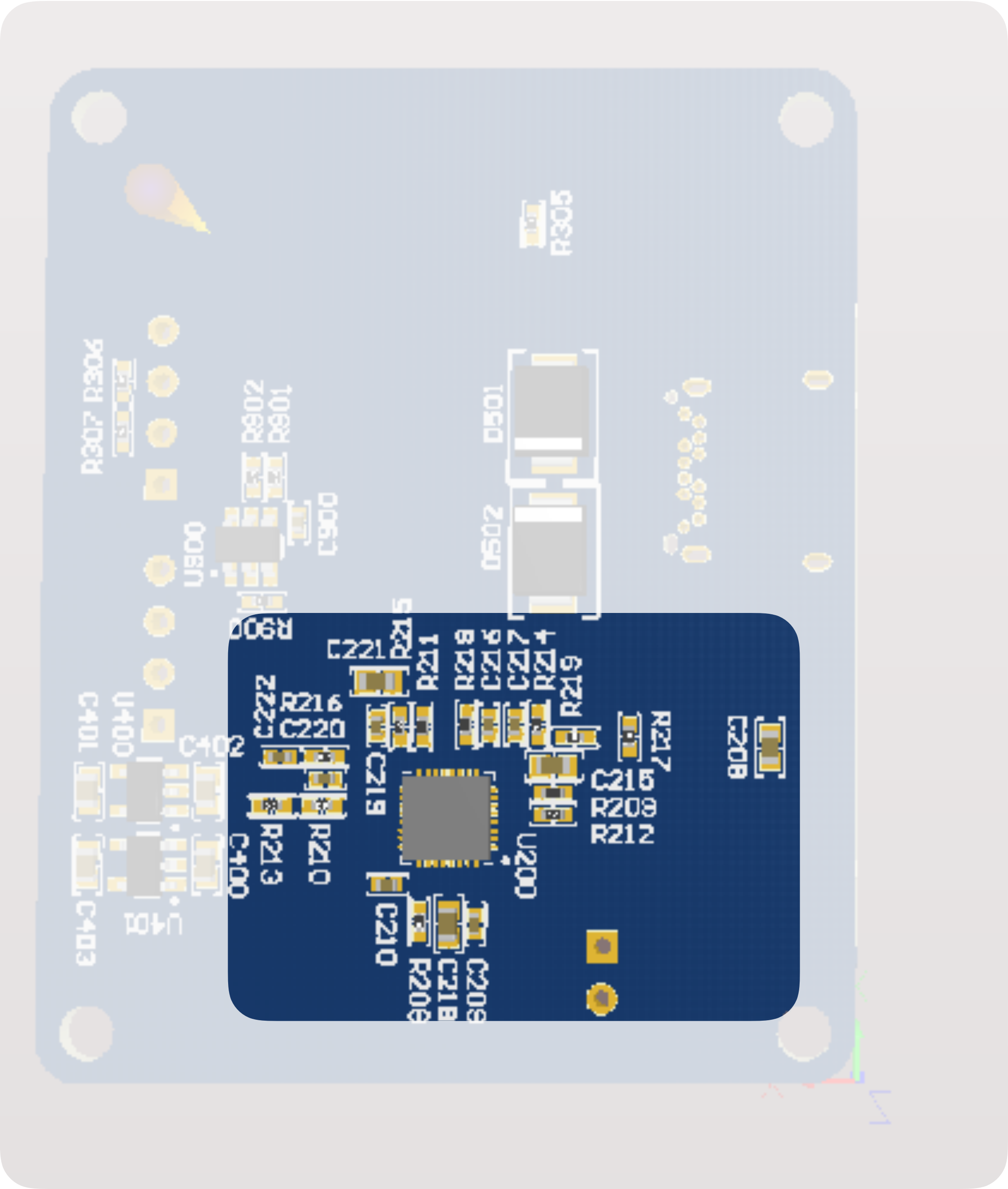
Battery Management System

- The Battery Management system charges and discharges the battery
- Keeps the batteries from catching fire
- Monitors the battery's operating condition, ensuring that it is operating safely
- Is controlled by the Bluetooth Module



Battery Management System

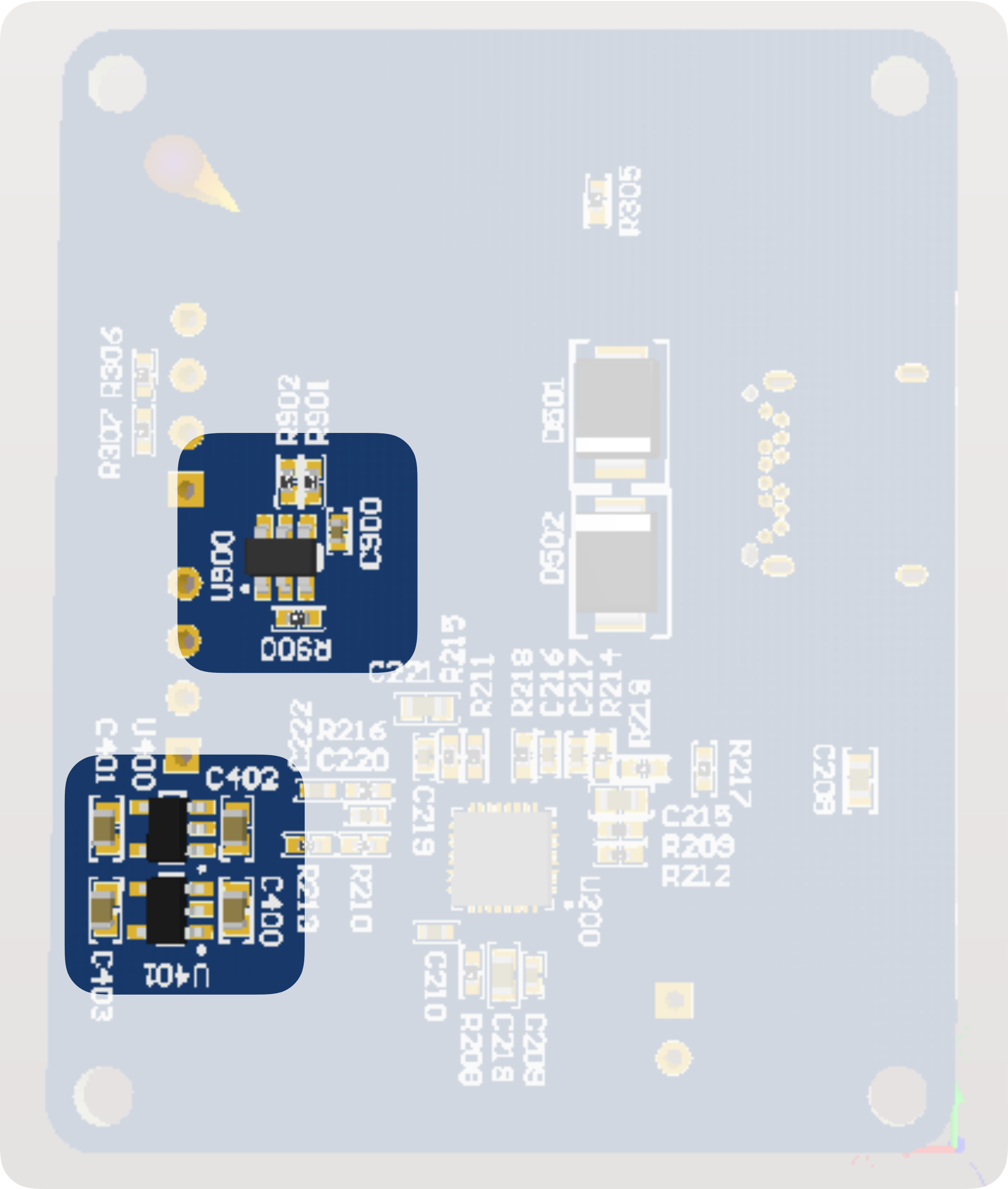
- The Battery Management system charges and discharges the battery
- Keeps the batteries from catching fire
- Monitors the battery's operating condition, ensuring that it is operating safely
- Is controlled by the Bluetooth Module



Supporting Circuitry

- Linear voltage regulators power onboard electronics

- A temperature sensor monitors the board temperature



Firmware

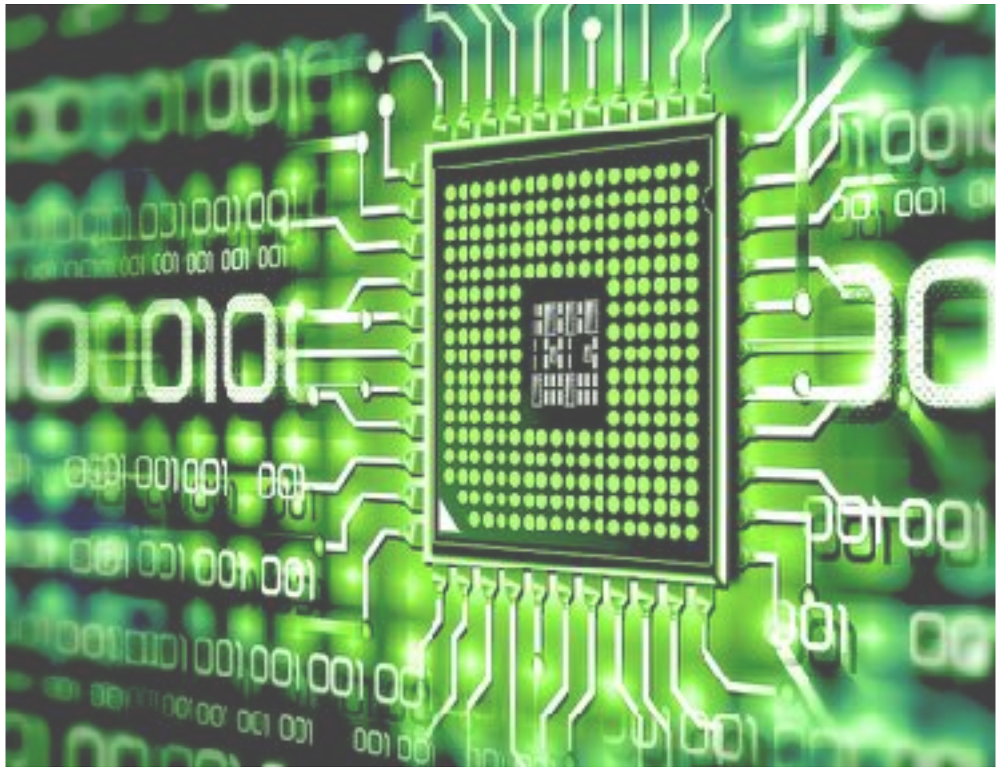
Steven Sun

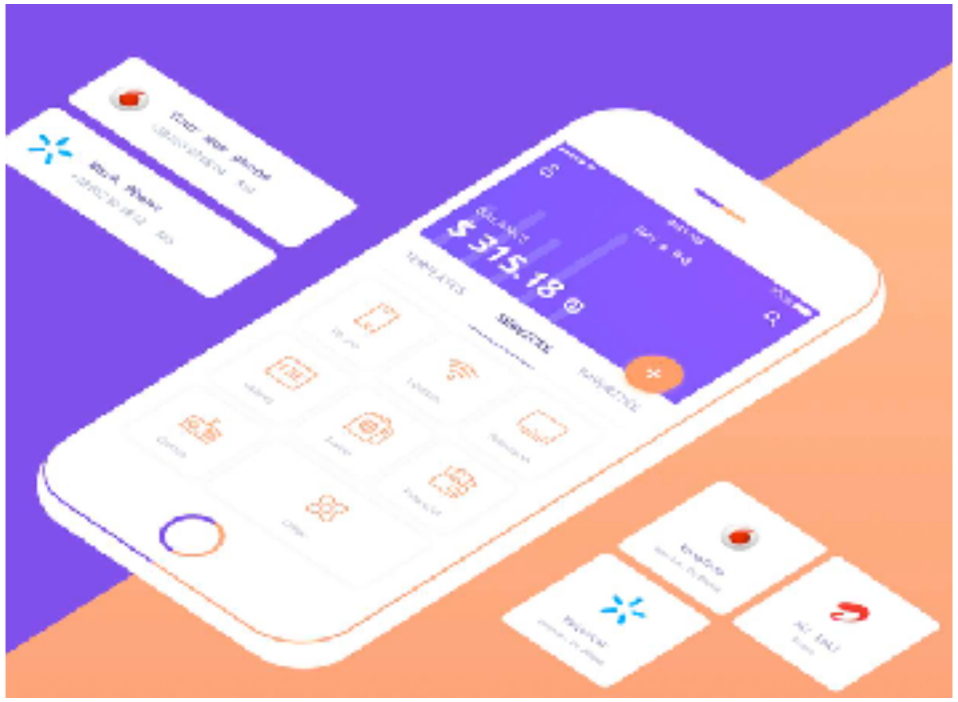
-Firmware: How hardware is configured to do specific tasks

-Implement I2C bus following I2C bus protocol

-Write specific driver for each device to configure the behavior by reading through data-sheet and understand the function of each register

-Implement BLE stack to make bluetooth connection between Phone and Hand





-Software: Interface between User and Hardware

-iOS device software to connect to the hand BLE hardware

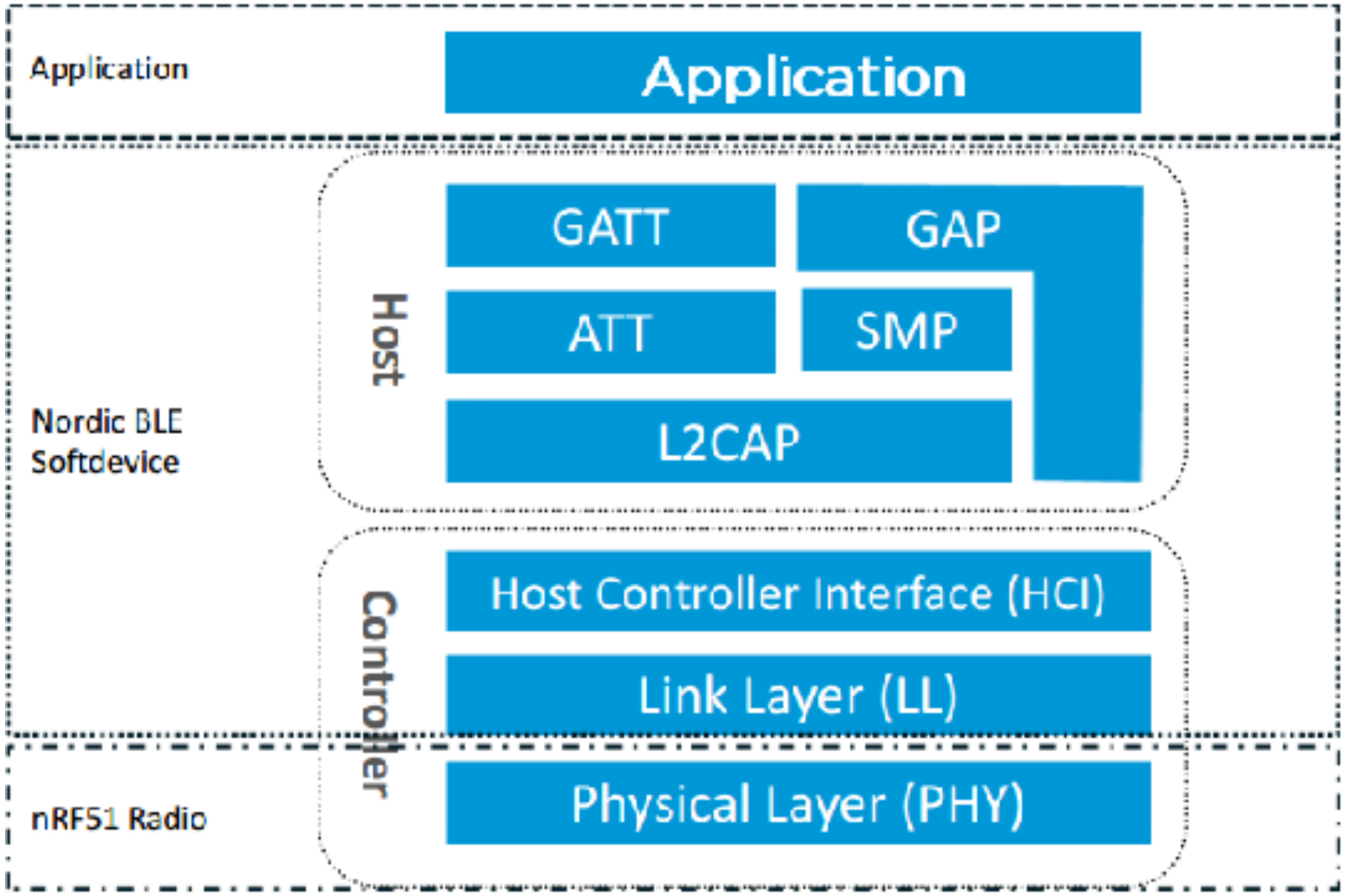
-EMG signal display

-Stimulator configuration

-Hand motor parameter configuration

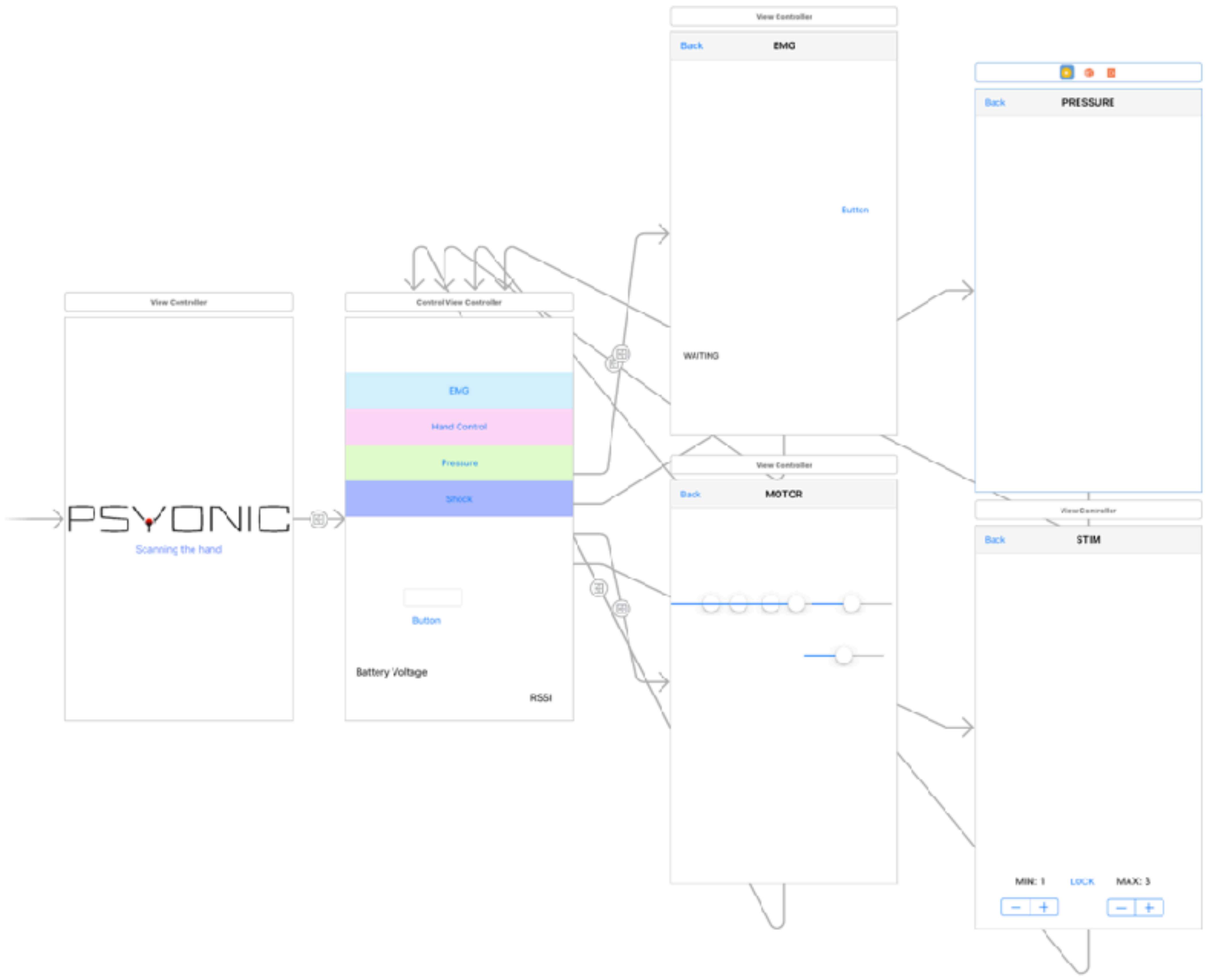
iOS PSYONIC APP

1. **Communicates to Bluetooth hardware following the Bluetooth Core Specification with different protocols**



2. **Provide easy to use GUI (user interface) to user. Using sliders and plots**

iOS PSYONIC APP - UI layout

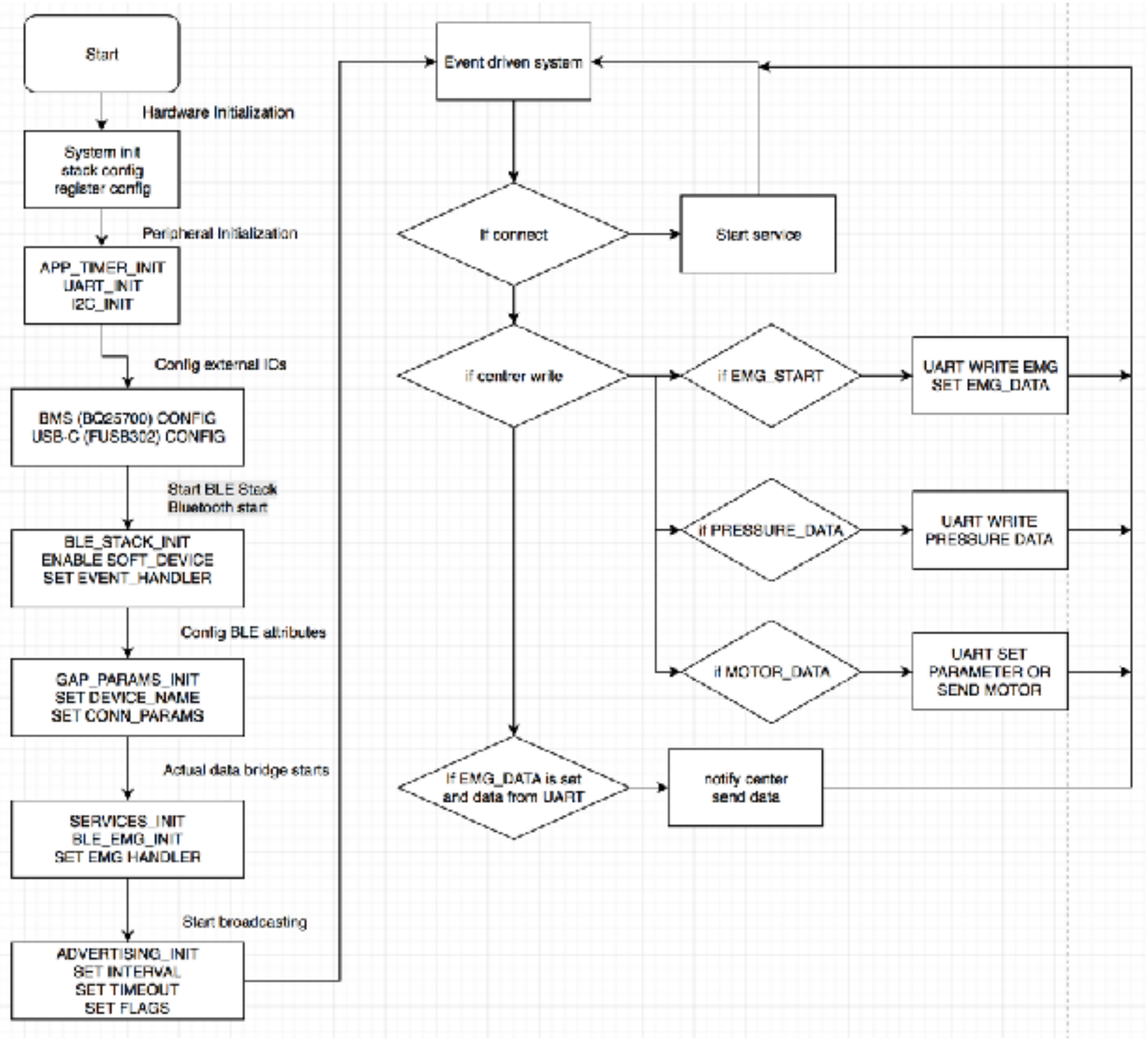


iOS PSYONIC APP - EMG data plot

Bluetooth Low Energy

-Firmware that hardware needed to emit bluetooth signal and communicate with other bluetooth device such as iPhone

-Defines the stability of connection, connection speed, encryption of connection.



I2C Data Bus Protocol

-How Micro-controller talk to the world.

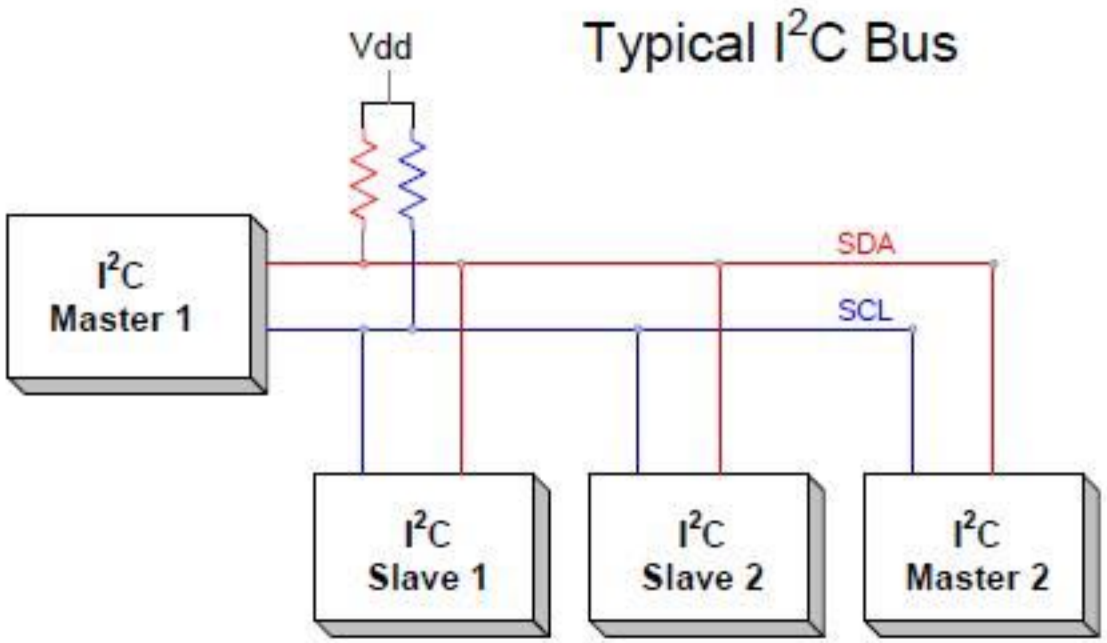
-We have Three devices (Battery Management, USB-C Power delivery, Temperature Sensor).

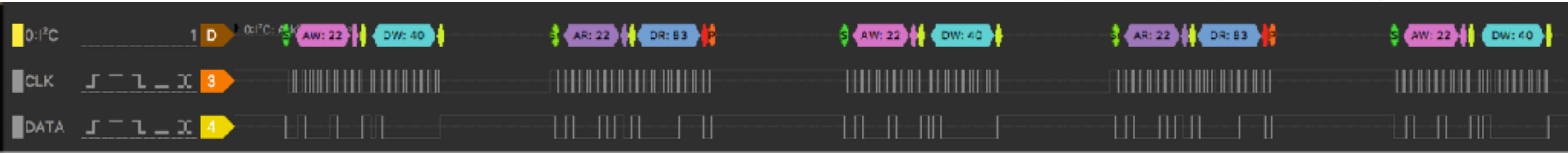
-Send data through Clock and Data to read/write to specific devices.

-Write to Battery Manage to set the charging current and voltage, Read from Battery Manage to know the status of the battery such as percentage and discharge current.

-Write to USB-C Power Delivery chip to select right voltage from charger

-Read from temperature sensor to know the Temperature





Programming (continued)

8.5.1.5 *Slave Address and Data Direction Bit*

After the START, a slave address is sent. This address is 7 bits long followed by the eighth bit as a data direction bit (bit R/W). A zero indicates a transmission (WRITE) and a one indicates a request for data (READ).

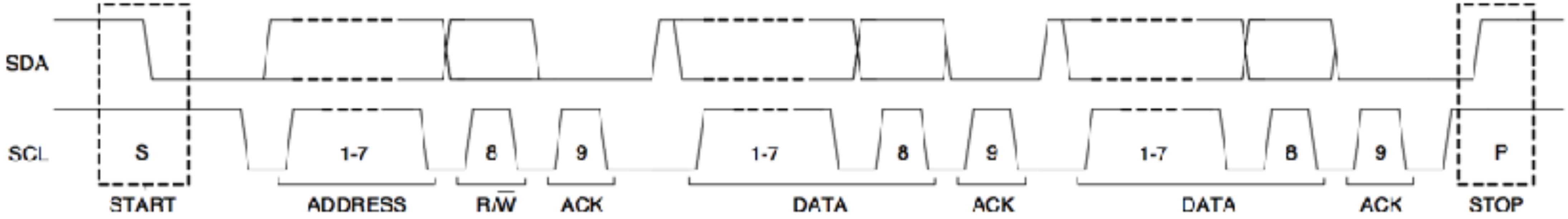
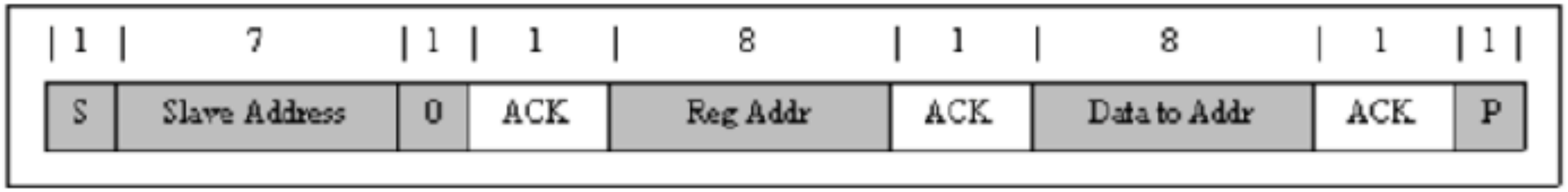


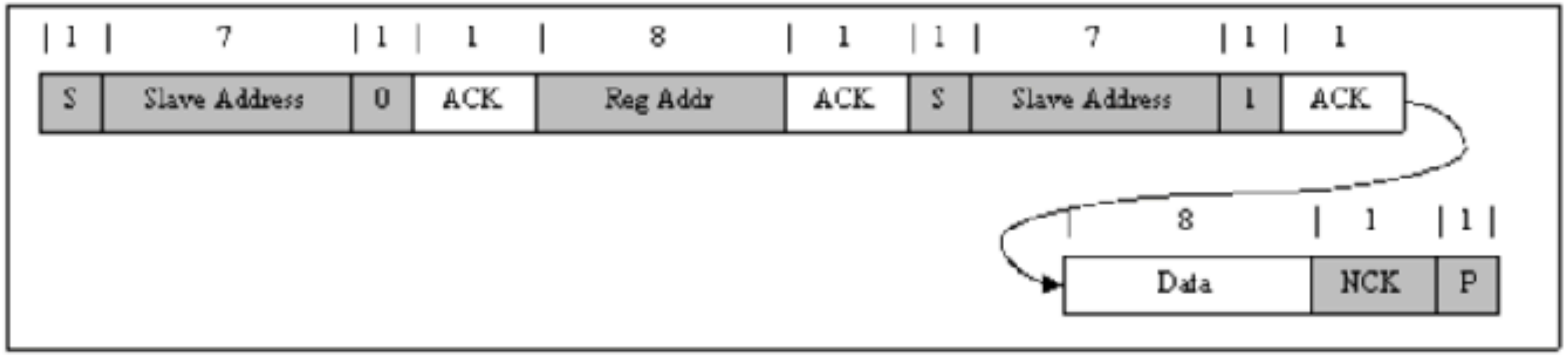
Figure 18. Complete Data Transfer

8.5.1.6 *Single Read and Write*



1	7	1	1	8	1	8	1	1
S	Slave Address	0	ACK	Reg Addr	ACK	Data to Addr	ACK	P

Figure 19. Single Write



1	7	1	1	8	1	1	7	1	1
S	Slave Address	0	ACK	Reg Addr	ACK	S	Slave Address	1	ACK

8	1	1
Data	NCK	P

Figure 20. Single Read

Configure the Battery Management System

-Chip we used is BQ25703A, which supports I2C bus protocol.

-I2C address is 0xD6 and support up to fast mode (400 kbits)

-Need to be configured to start charging (safety feature)

-Charge option (0x01 register address): set charging buck-boost converter switch frequency. And Enable charging.

-

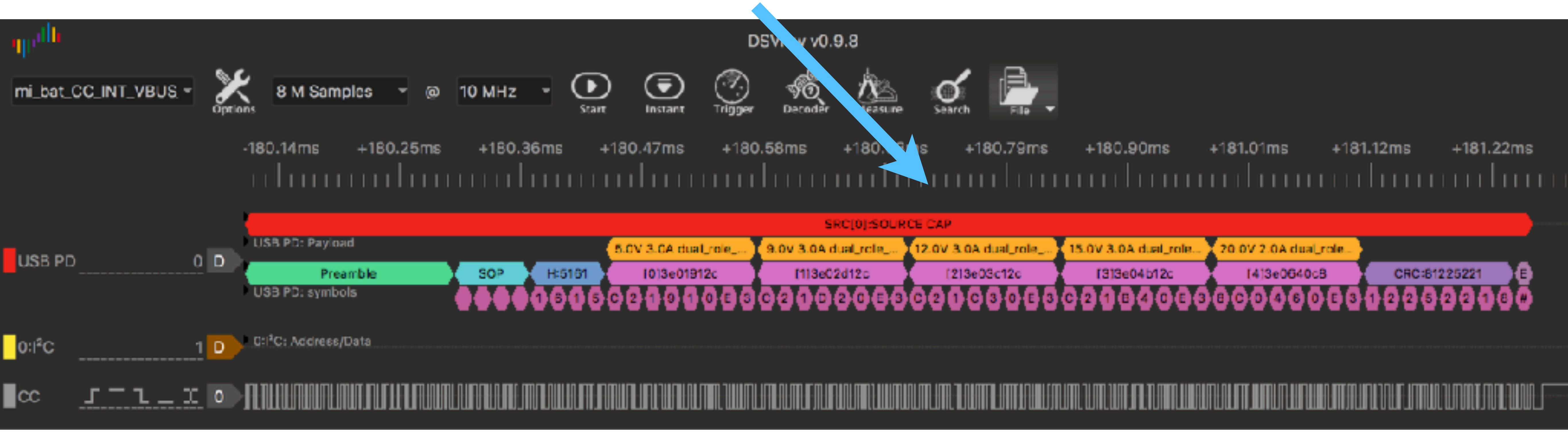
USB Type-C with Power Delivery

Function:

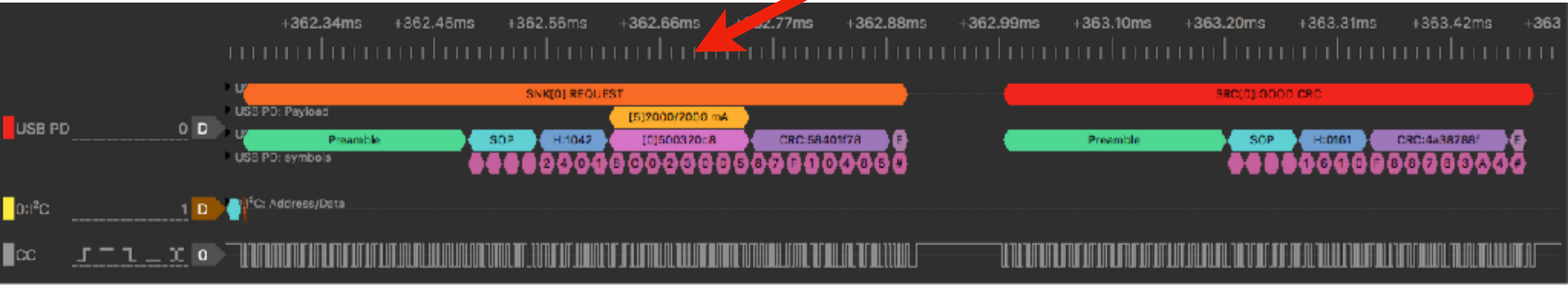
-Communicates to all USB Type-C chargers from the market (Google Pixel Charger, Apple Macbook pro Charger, Microsoft surface book charger) to get desired voltage setting for Battery Management System. So users can use any USB Type-C charger to rapidly charge their hand.

-Compatible with legacy USB, get 5V 1A for slow charging

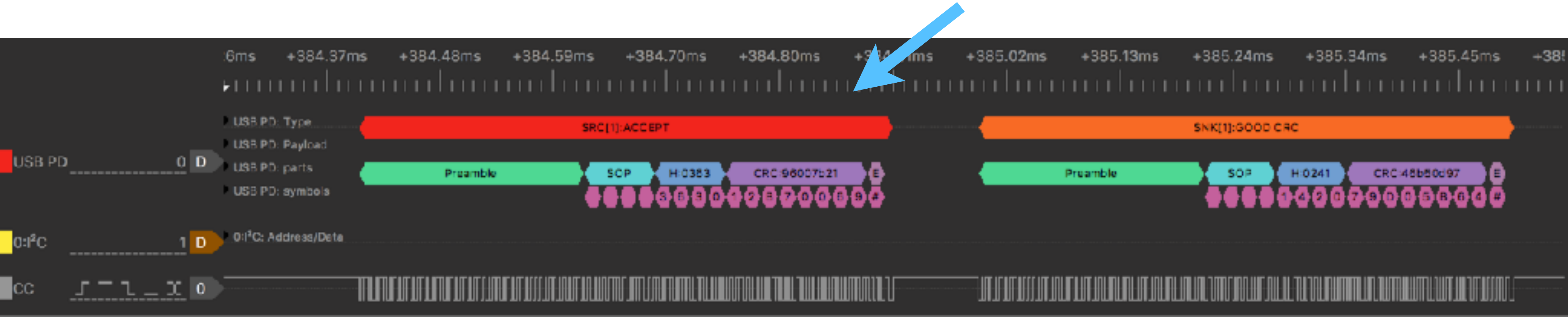
1.Hello, I am Charger, I can provide these power settings



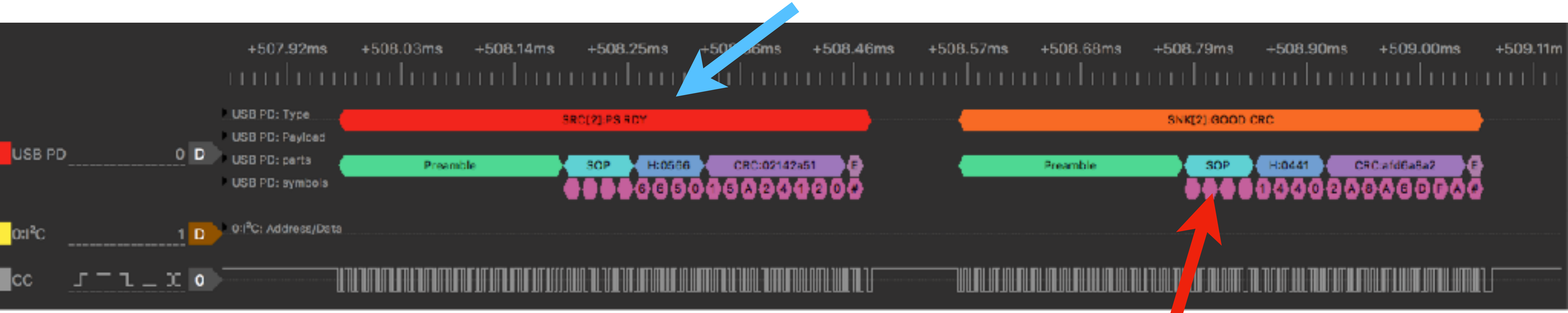
2.Hello, I am the hand, can I get this power setting please



3.Hello, I am Charger, that sounds good, let me get ready.

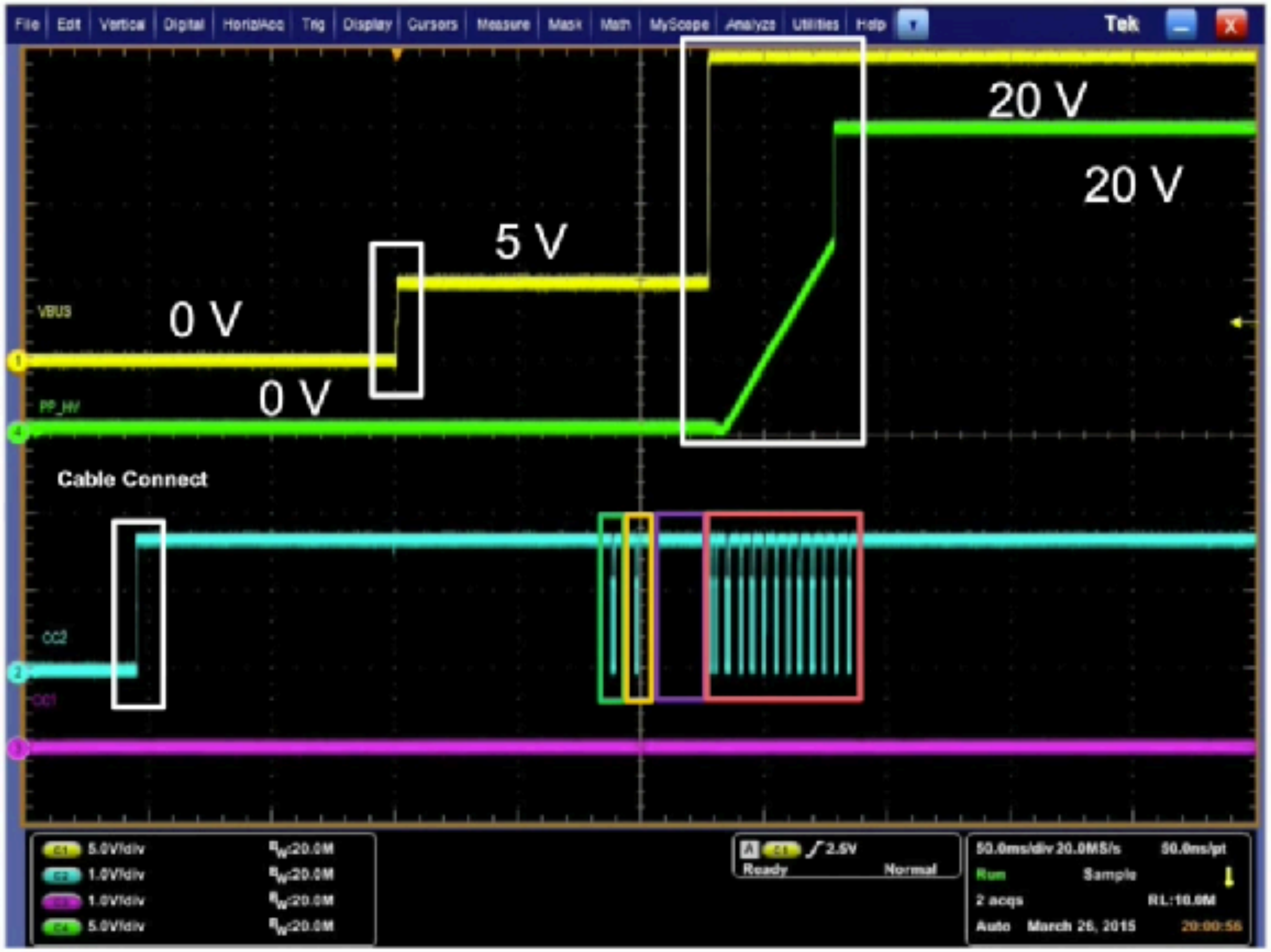


4.Hello, I am Charger, I AM READY, here you go

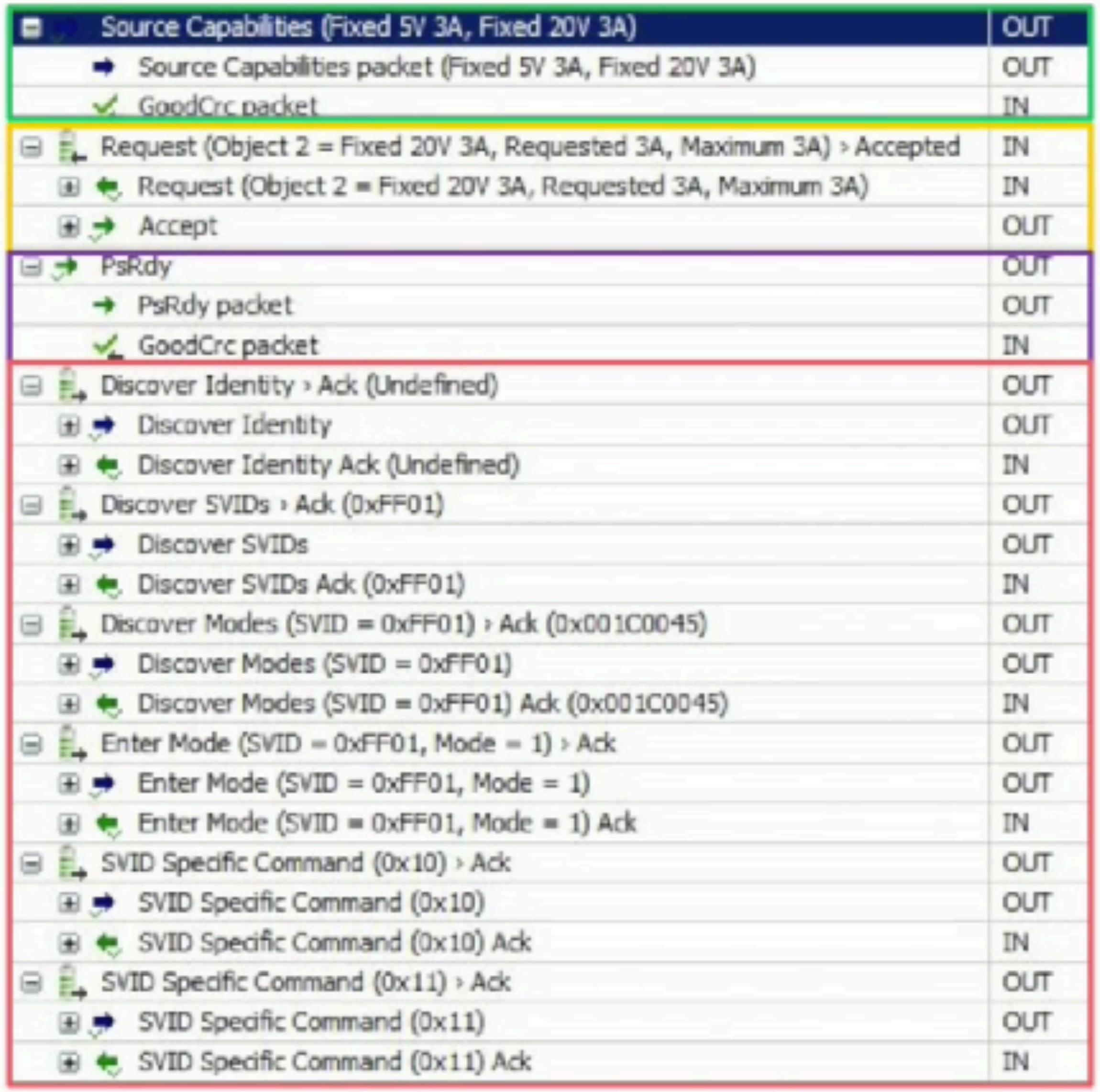


5.Hello, I am hand, awesome, thank you

Scope Trace of Notebook (as a consumer)



Ellisys Analyzer – PD Negotiation



Source Capabilities (Fixed 5V 3A, Fixed 20V 3A)	OUT
Source Capabilities packet (Fixed 5V 3A, Fixed 20V 3A)	OUT
GoodCrc packet	IN
Request (Object 2 = Fixed 20V 3A, Requested 3A, Maximum 3A) › Accepted	IN
Request (Object 2 = Fixed 20V 3A, Requested 3A, Maximum 3A)	IN
Accept	OUT
PsRdy	OUT
PsRdy packet	OUT
GoodCrc packet	IN
Discover Identity › Ack (Undefined)	OUT
Discover Identity	OUT
Discover Identity Ack (Undefined)	IN
Discover SVIDs › Ack (0xFF01)	OUT
Discover SVIDs	OUT
Discover SVIDs Ack (0xFF01)	IN
Discover Modes (SVID = 0xFF01) › Ack (0x001C0045)	OUT
Discover Modes (SVID = 0xFF01)	OUT
Discover Modes (SVID = 0xFF01) Ack (0x001C0045)	IN
Enter Mode (SVID = 0xFF01, Mode = 1) › Ack	OUT
Enter Mode (SVID = 0xFF01, Mode = 1)	OUT
Enter Mode (SVID = 0xFF01, Mode = 1) Ack	IN
SVID Specific Command (0x10) › Ack	OUT
SVID Specific Command (0x10)	OUT
SVID Specific Command (0x10) Ack	IN
SVID Specific Command (0x11) › Ack	OUT
SVID Specific Command (0x11)	OUT
SVID Specific Command (0x11) Ack	IN

Results

Battery Management System

- Battery can be charged properly across the entire battery voltage range
- Capable of preventing the battery from operating outside of its safe operating area
- Battery voltage and current can be monitored by the Bluetooth Module
- Meets all requirements set in the Design Document

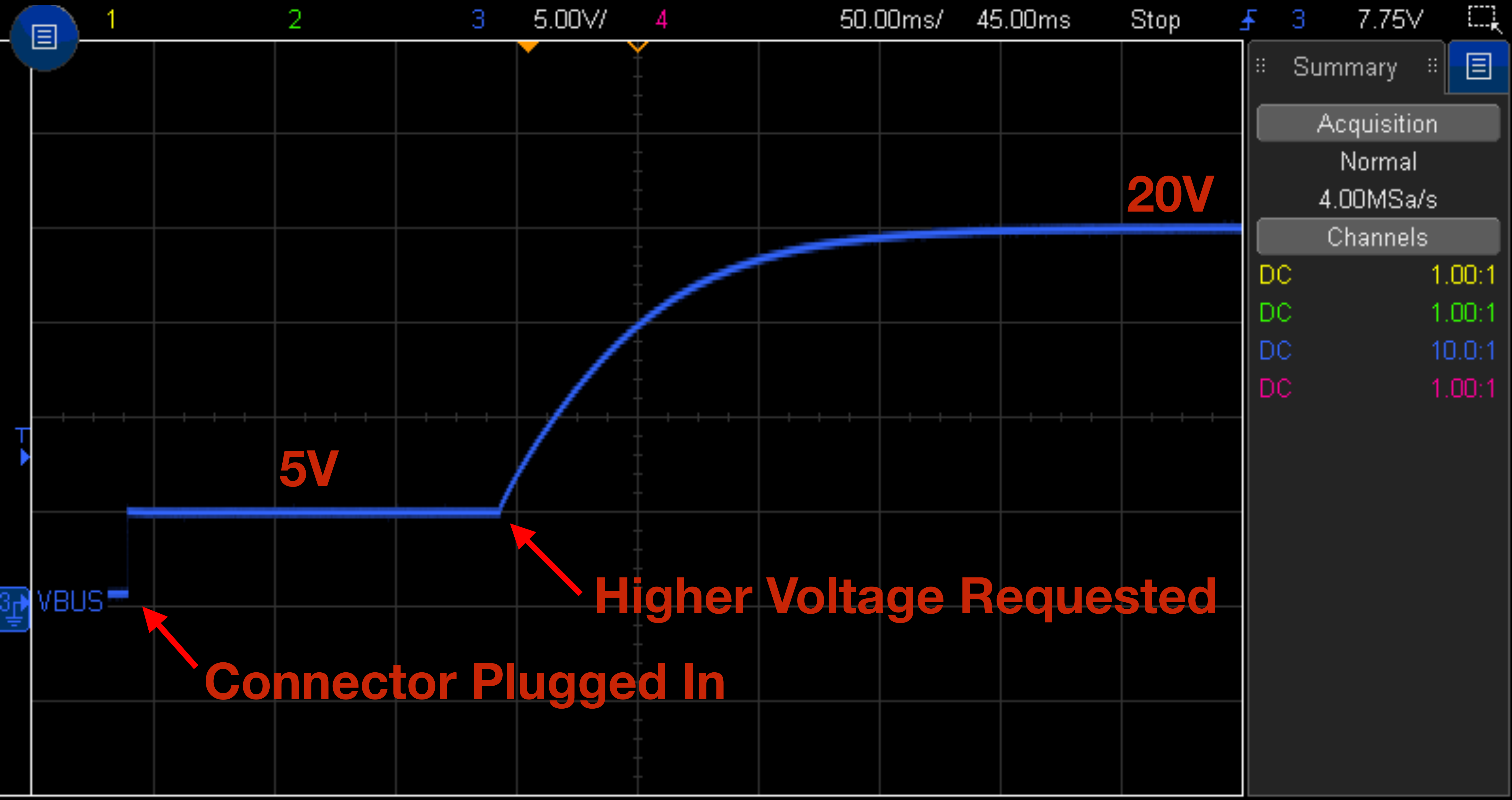
Slide with charts of BMS performance

Bluetooth Low Energy

Stim Video goes here
Byron will add it

USB Power Delivery

- Able to communicate with chargers from Samsung, Google, and a variety of other brands
- The USB Power Delivery code could not be run concurrently with the Bluetooth code
- Successfully increased the supply voltage upon request
- Met requirements set in the Design Document



3 5.00V/

50.00ms/ 45.00ms Stop 3 7.75V

Summary

Acquisition

Normal

4.00MSa/s

Channels

DC	1.00:1
DC	1.00:1
DC	10.0:1
DC	1.00:1

20V

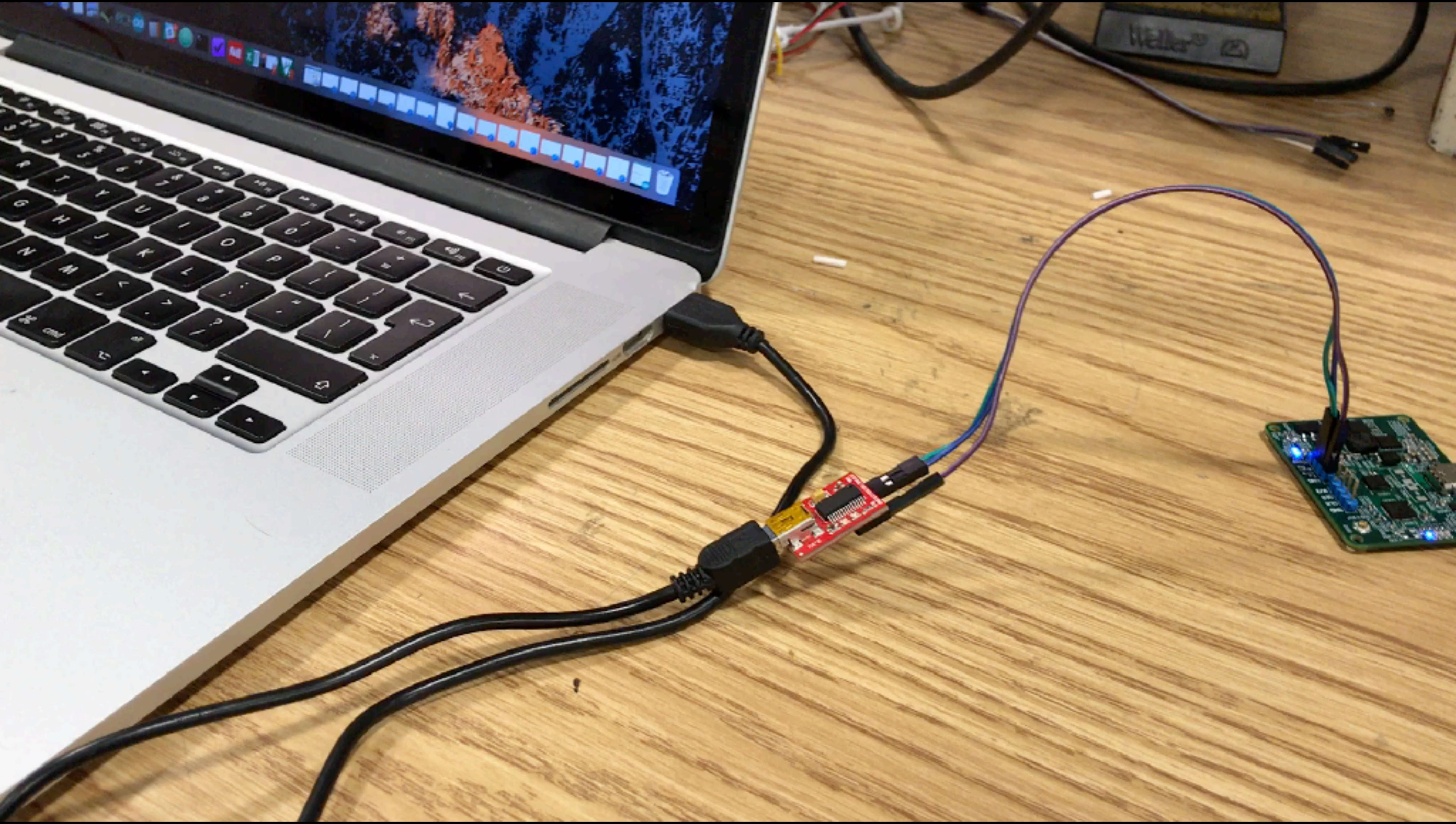
5V

Higher Voltage Requested

Connector Plugged In

USB Serial Converter

- Enumerated as a USB serial device to both Windows and MacOS computers
- Supported a baud rate of 115200
- Met requirements set in the Design Document


Weller

Lithium-Ion Battery

- Battery supported the maximum expected discharge rate without a serious voltage drop
- Met requirements set in the Design Document

Chart with battery discharge characteristics goes here

Lessons Learned

- The NRF51822 does not have enough compute resources to handle BLE and USB-PD concurrently

- Lack of UART pull-up resistors generates garbage data when no device is attached

- USB protection diodes had too much capacitance, preventing USB communications

- No protection diodes on the Serial Multiplexer created the potential for static electricity damage

- Potential back-powering issue due to separate 5V and 3.3V domains

Next Steps

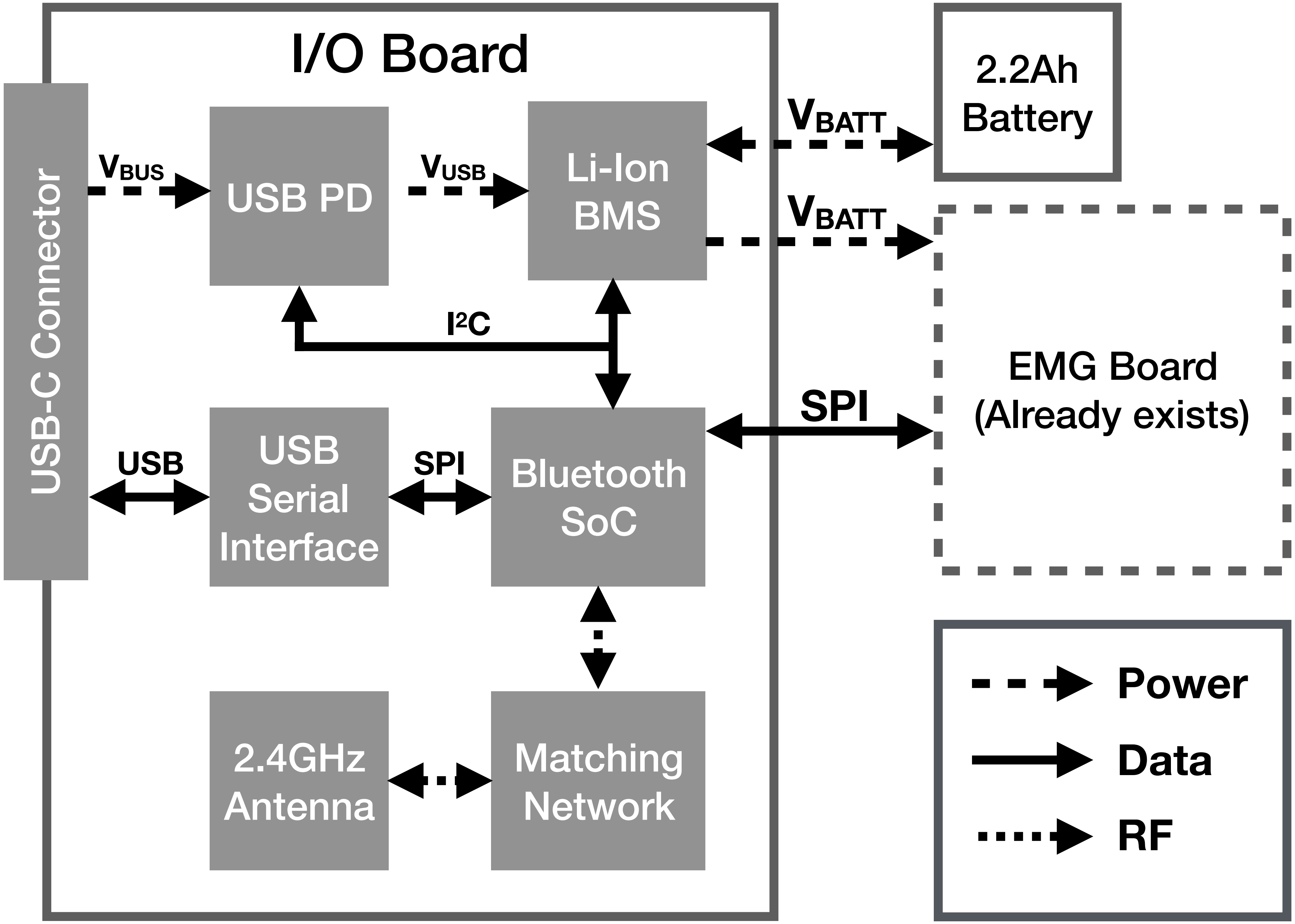
- Evaluate architecture changes
- Spin I/O Board incorporating lessons learned
- Integrate I/O Board onto backside of the Hand Board

Questions?

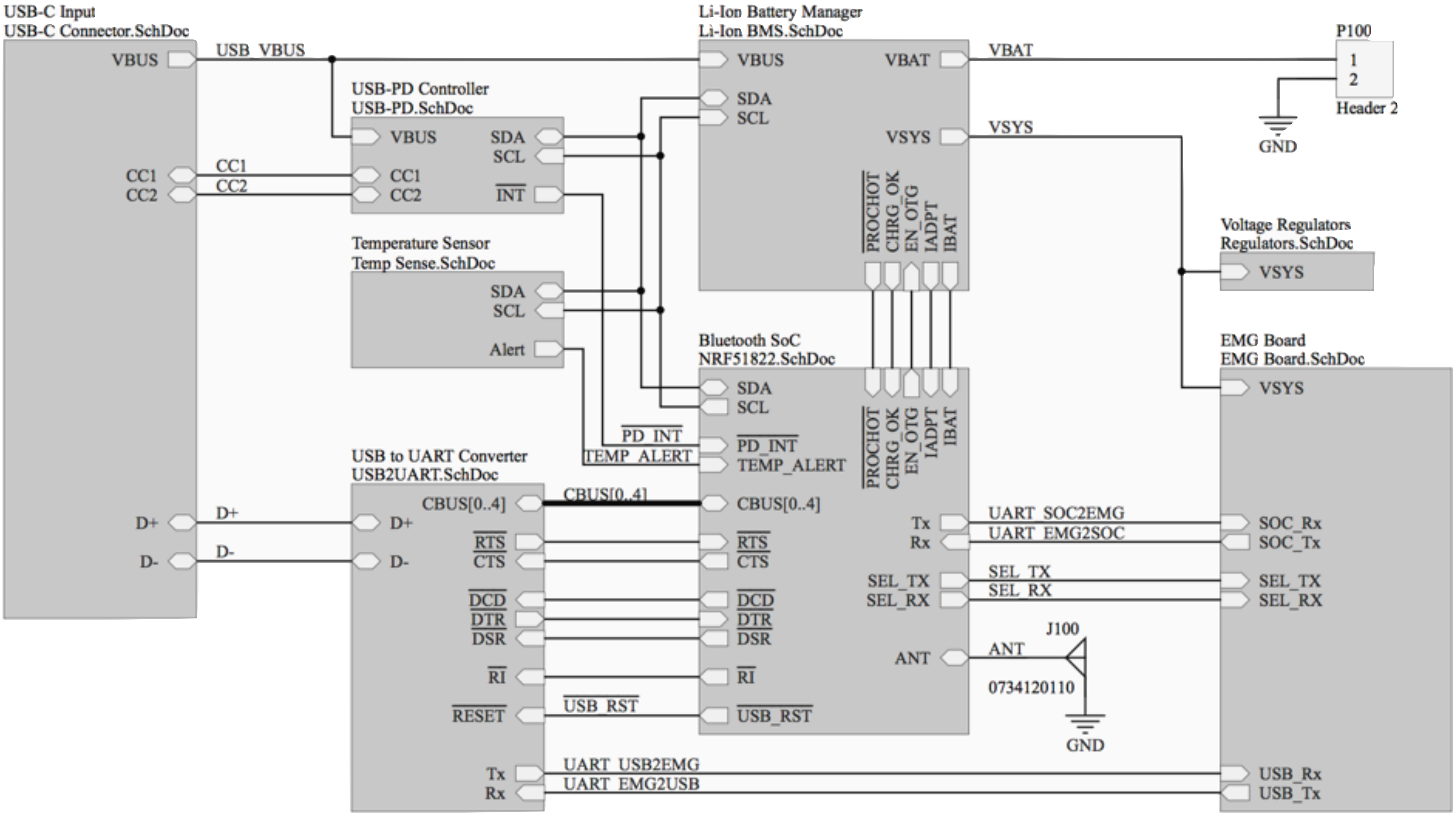
Thank You

Backup Slides

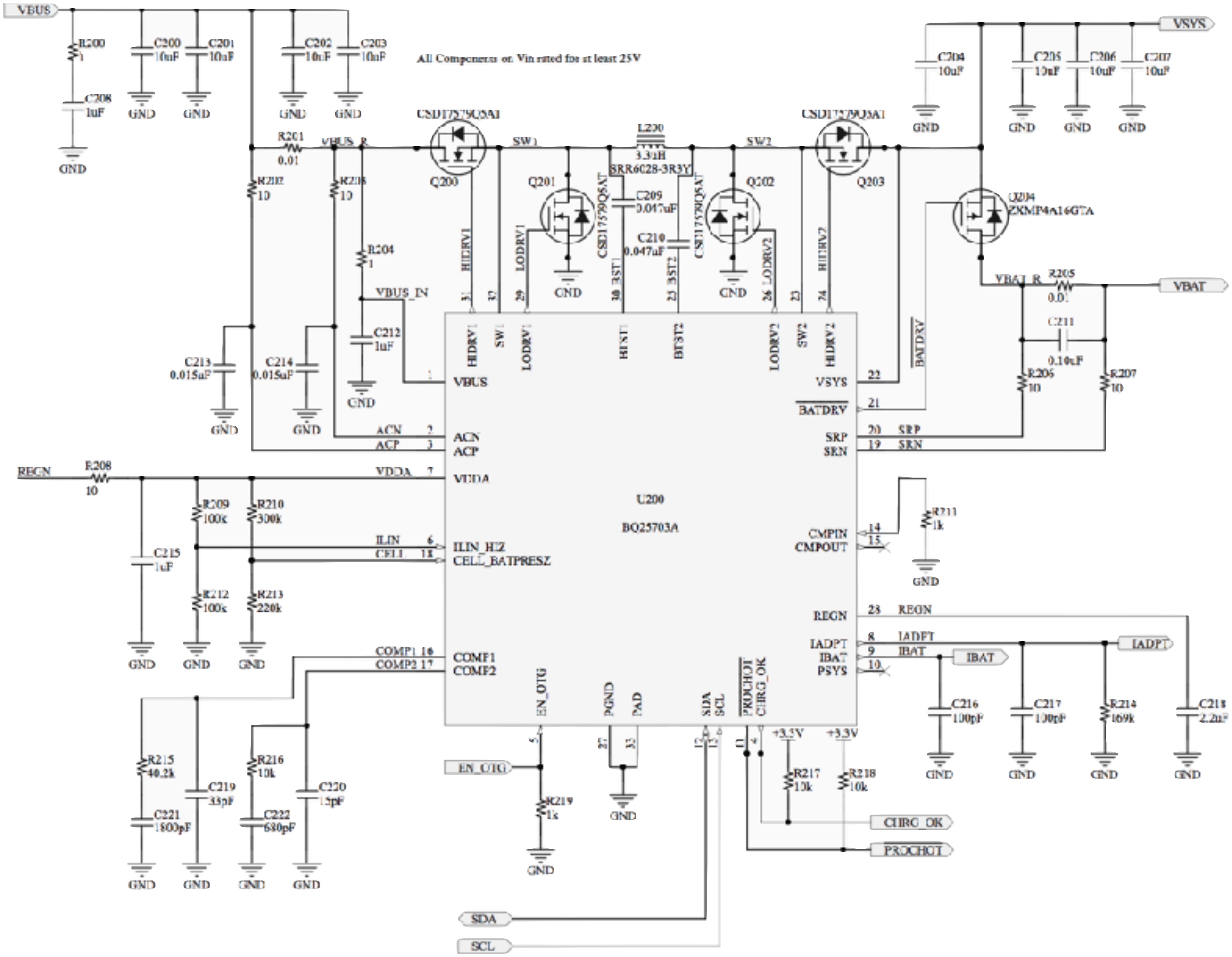
Initial System Architecture



Top-Level Schematic

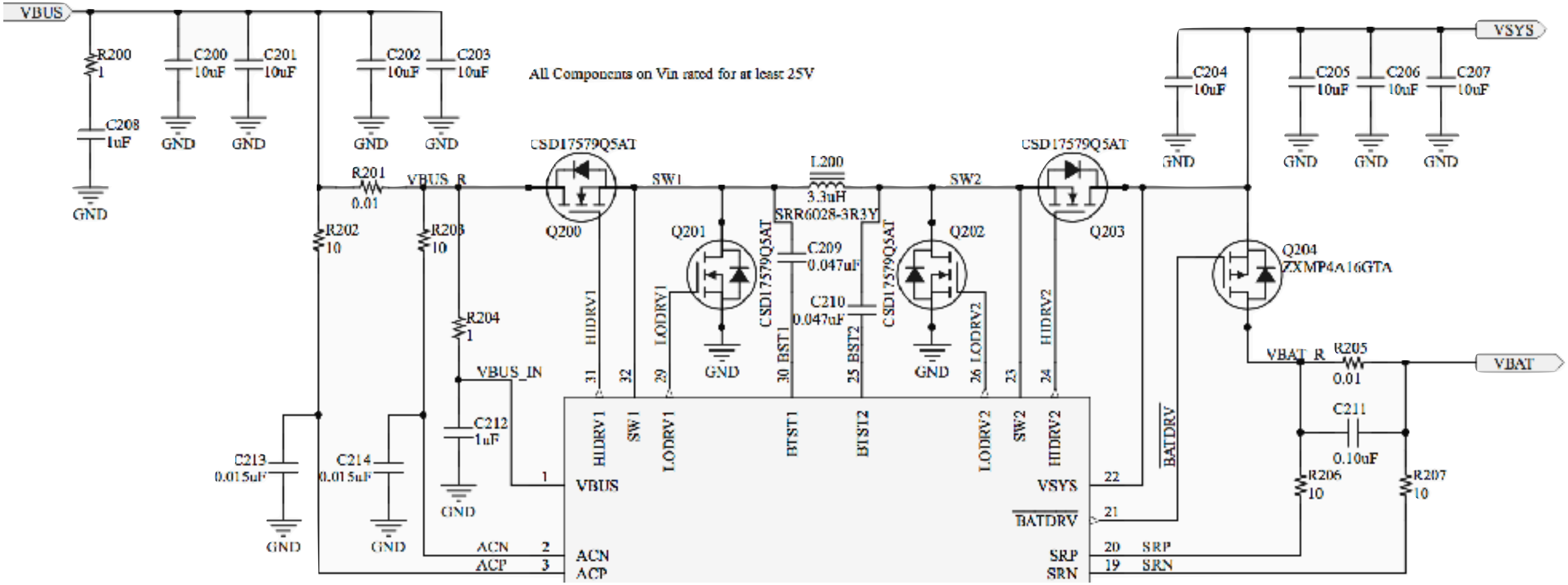


Battery Management System

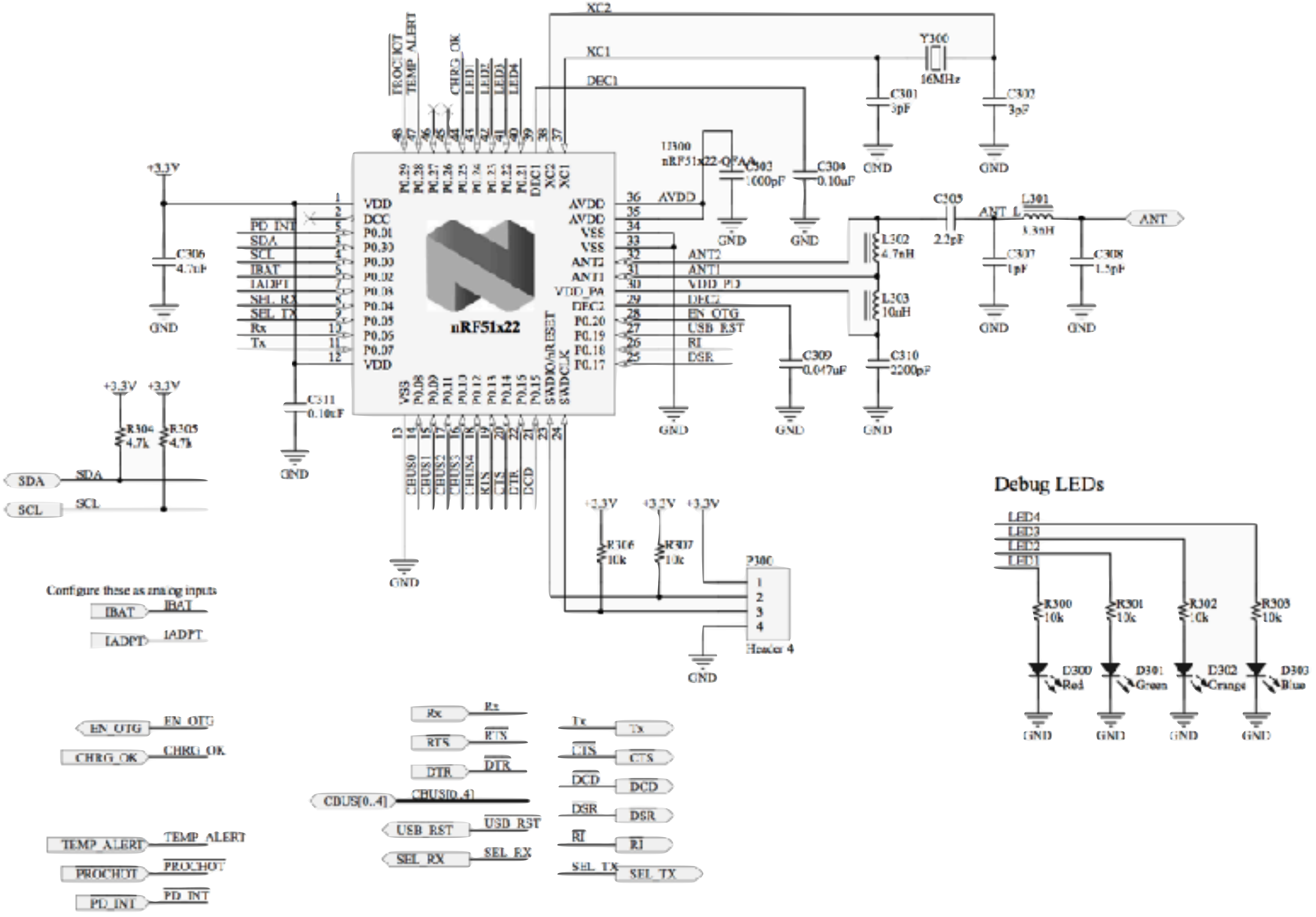


All Components on Vin rated for at least 25V

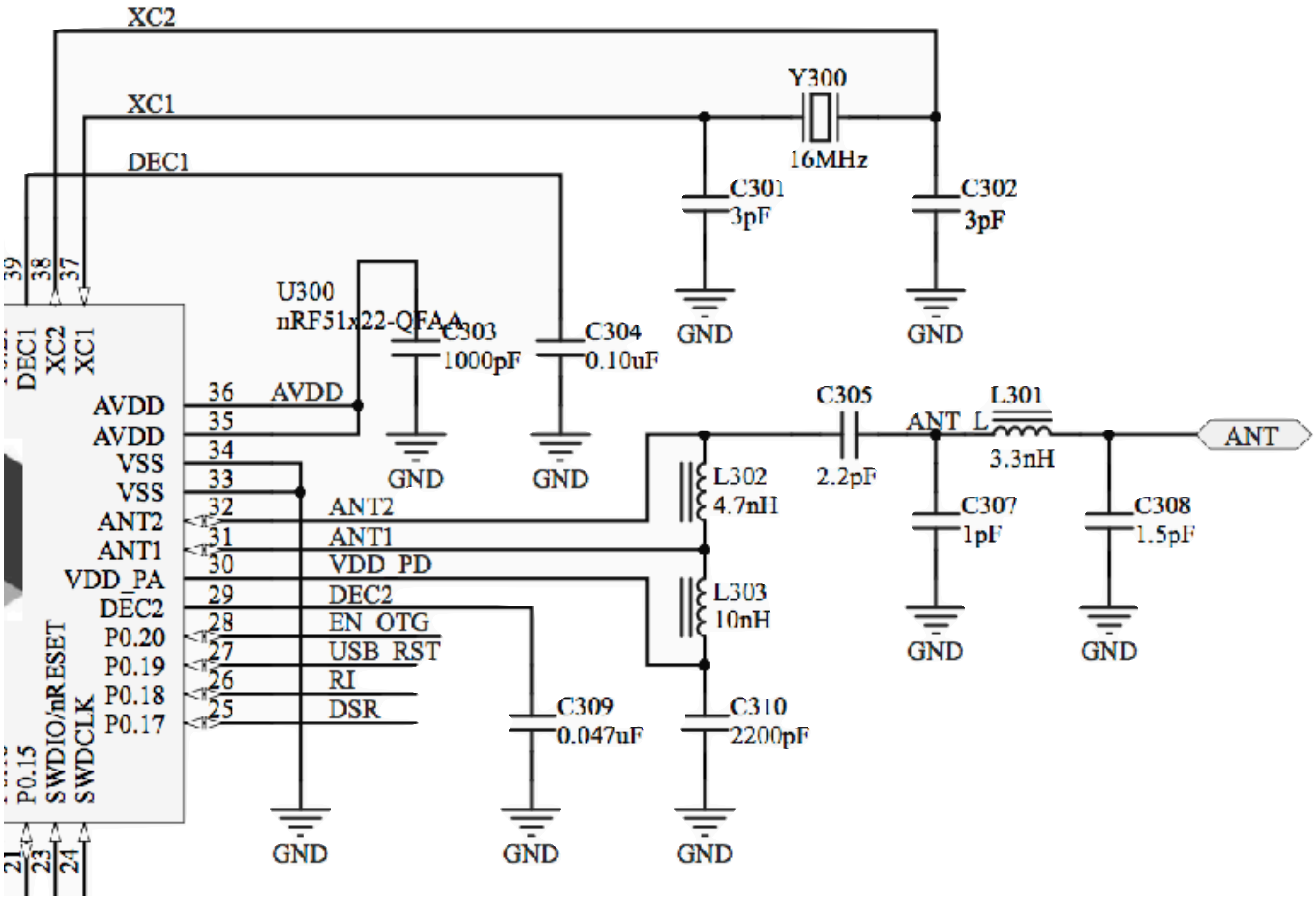
BMS Power Path



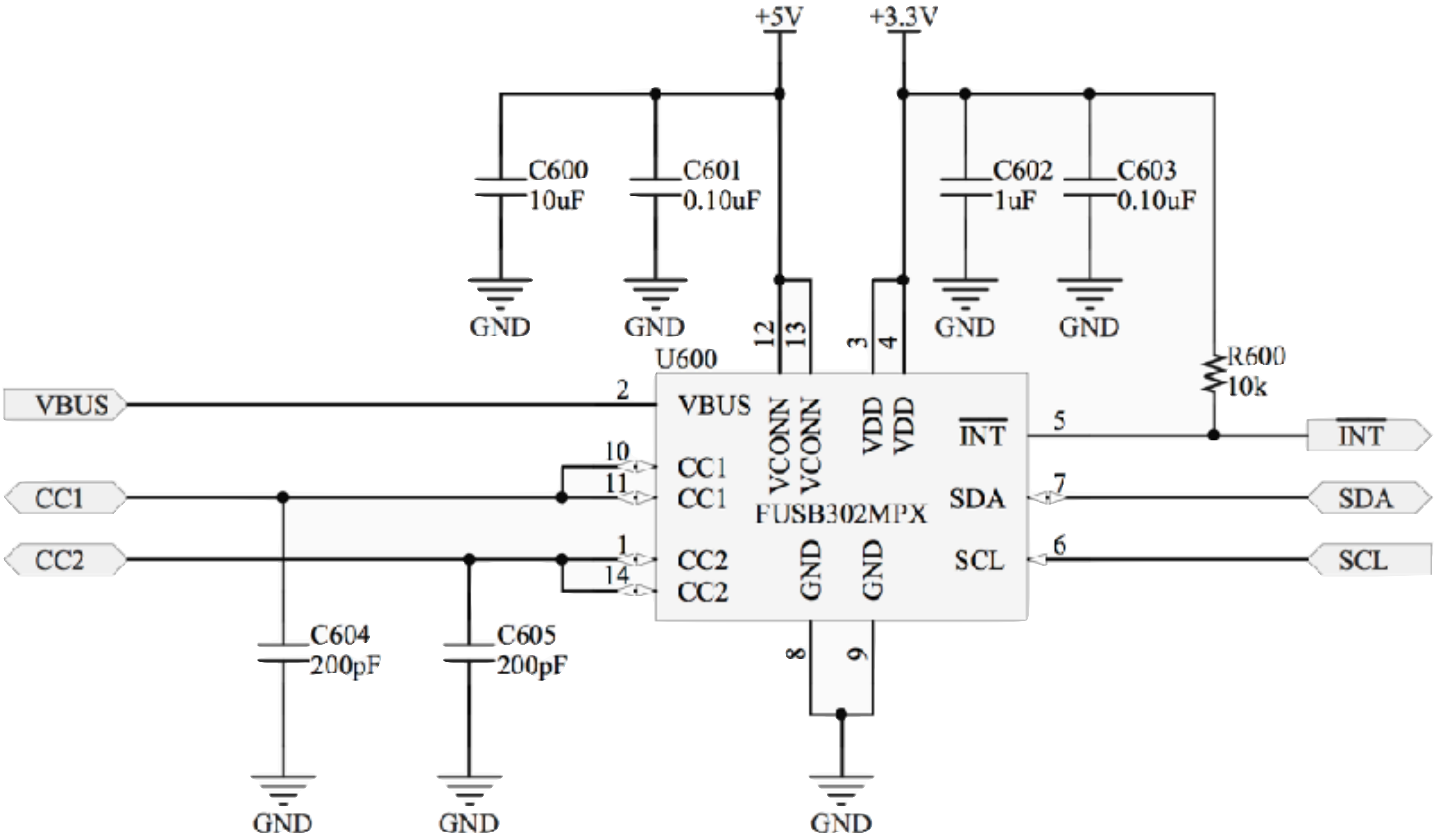
Bluetooth SoC



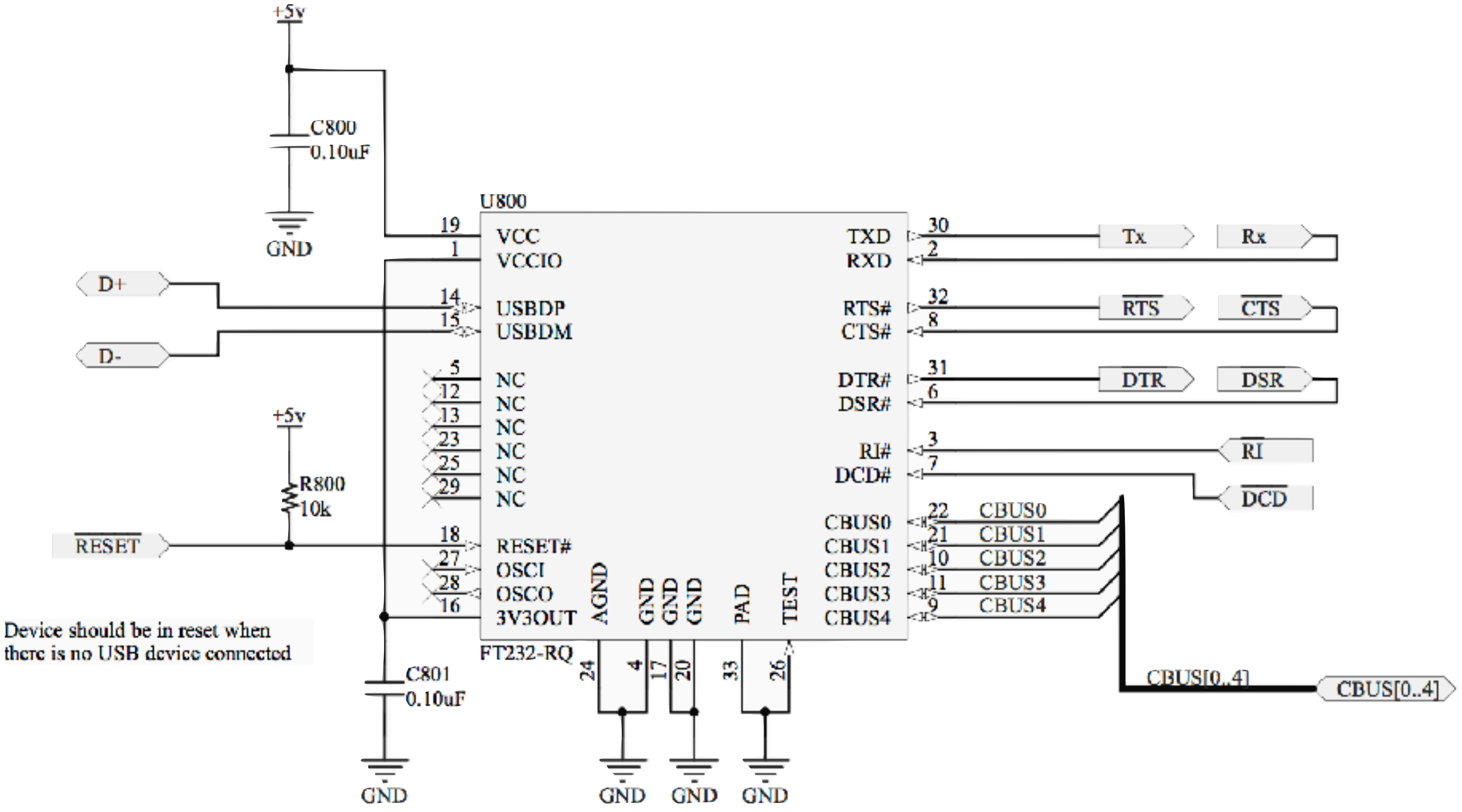
RF Matching Network



USB-PD Negotiator

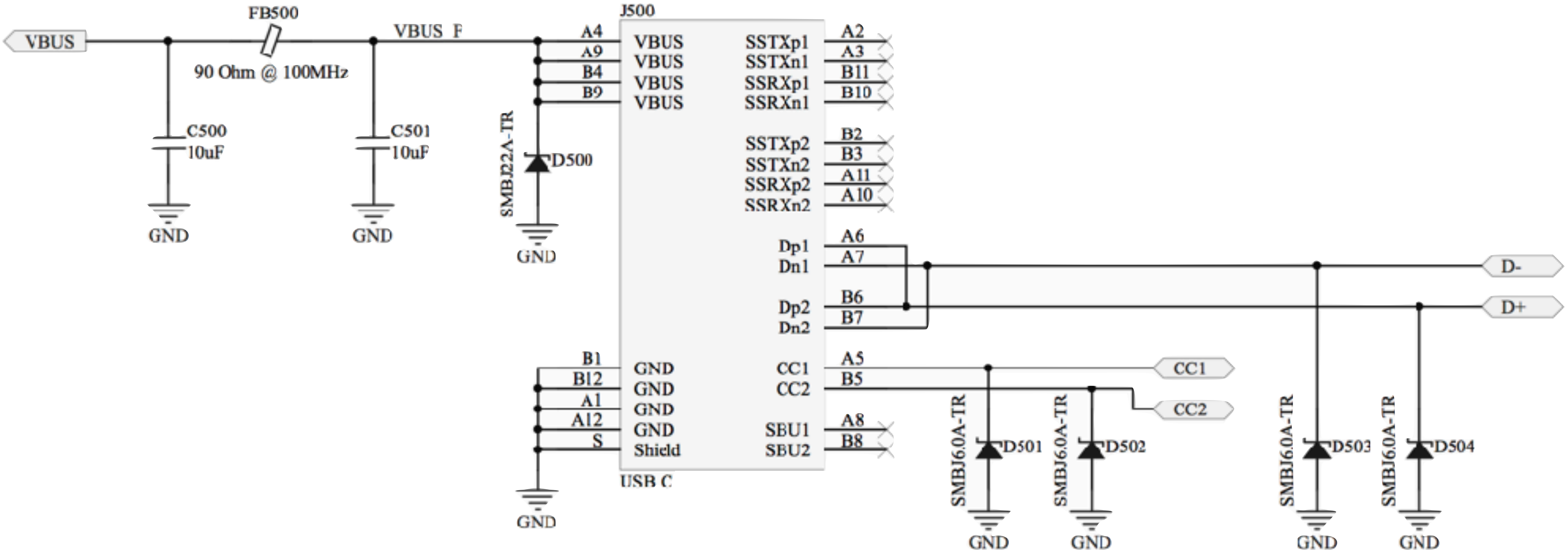


USB Serial Interface

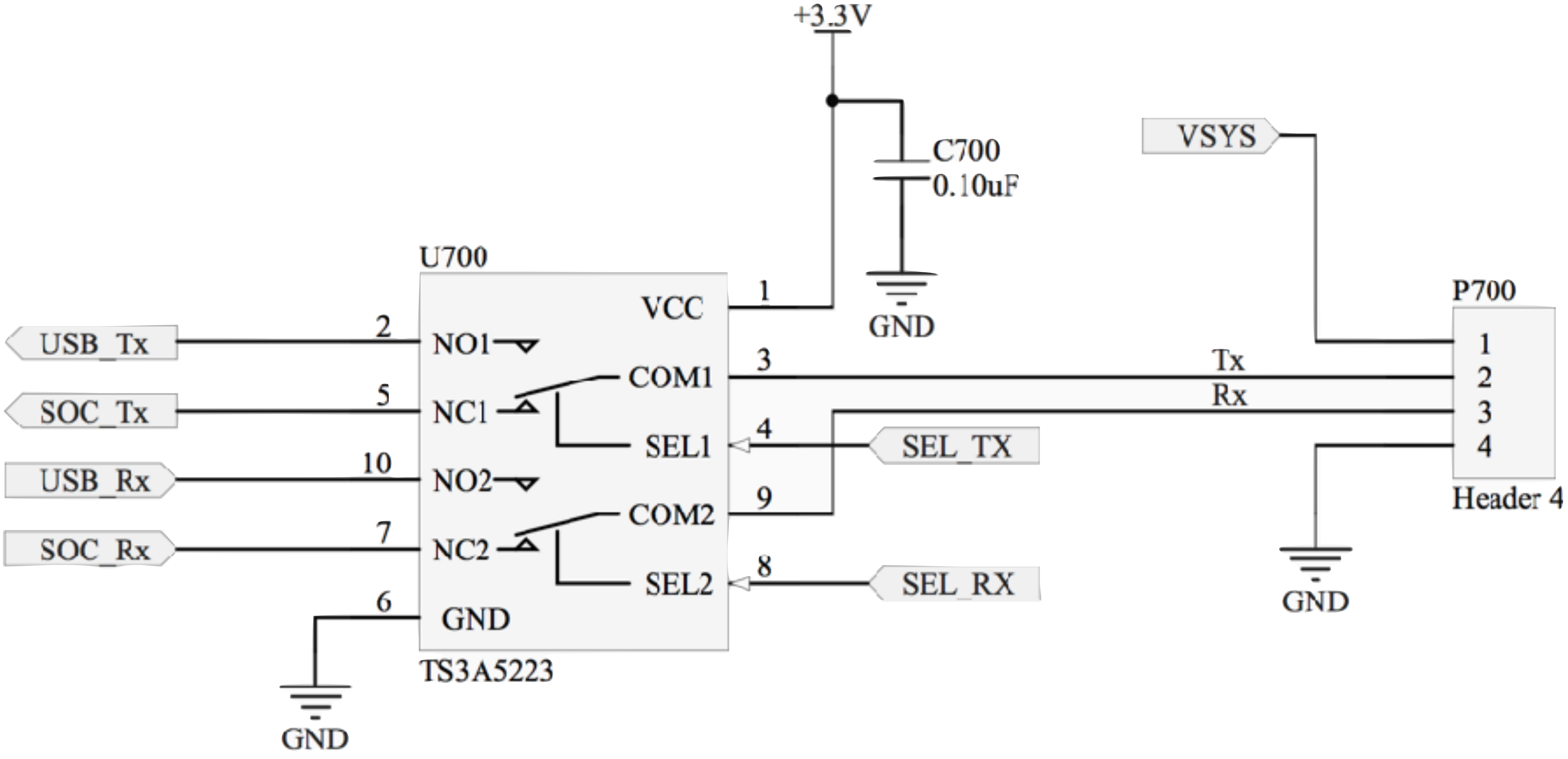


Device should be in reset when
there is no USB device connected

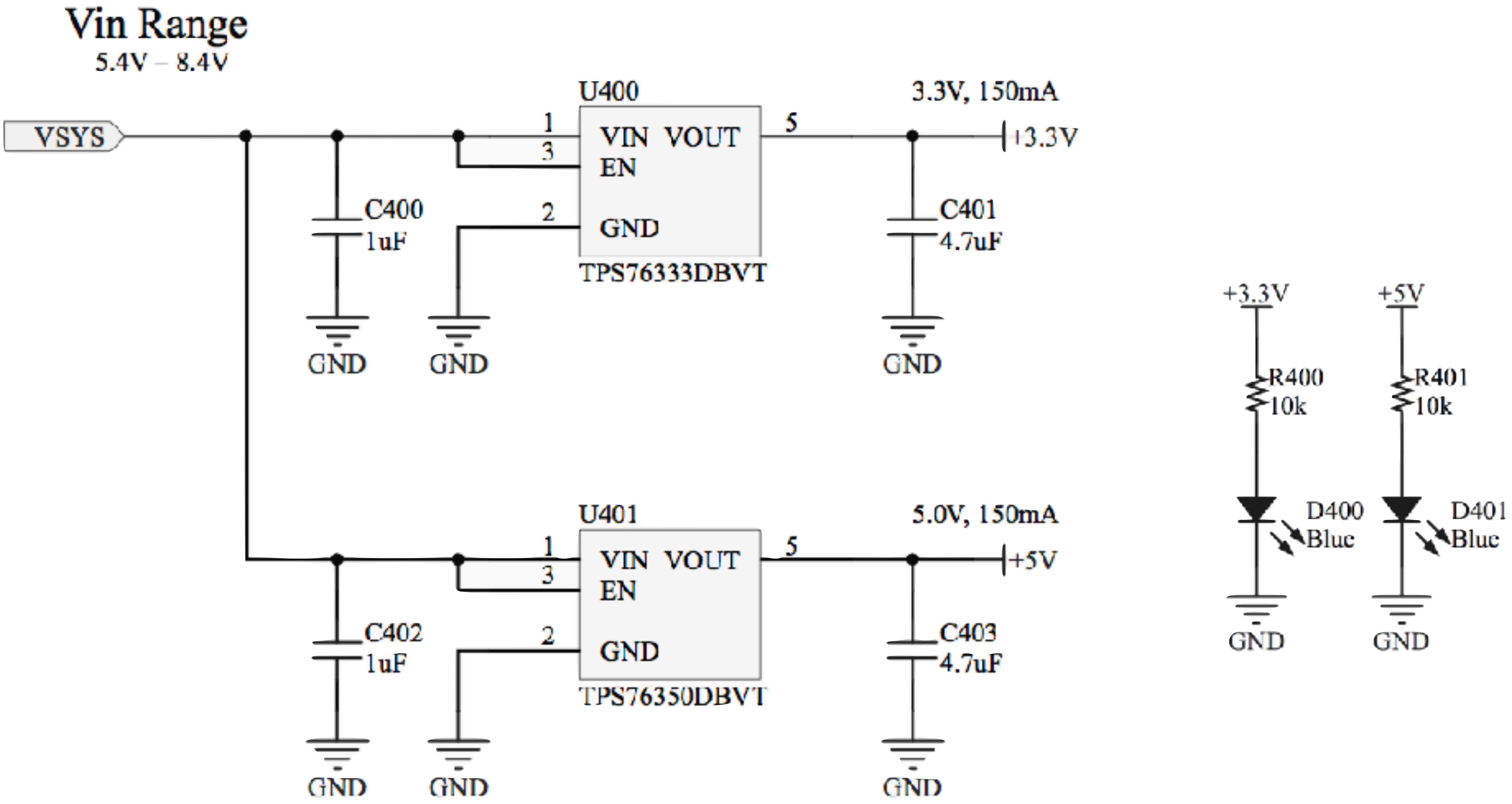
USB Type-C Connector



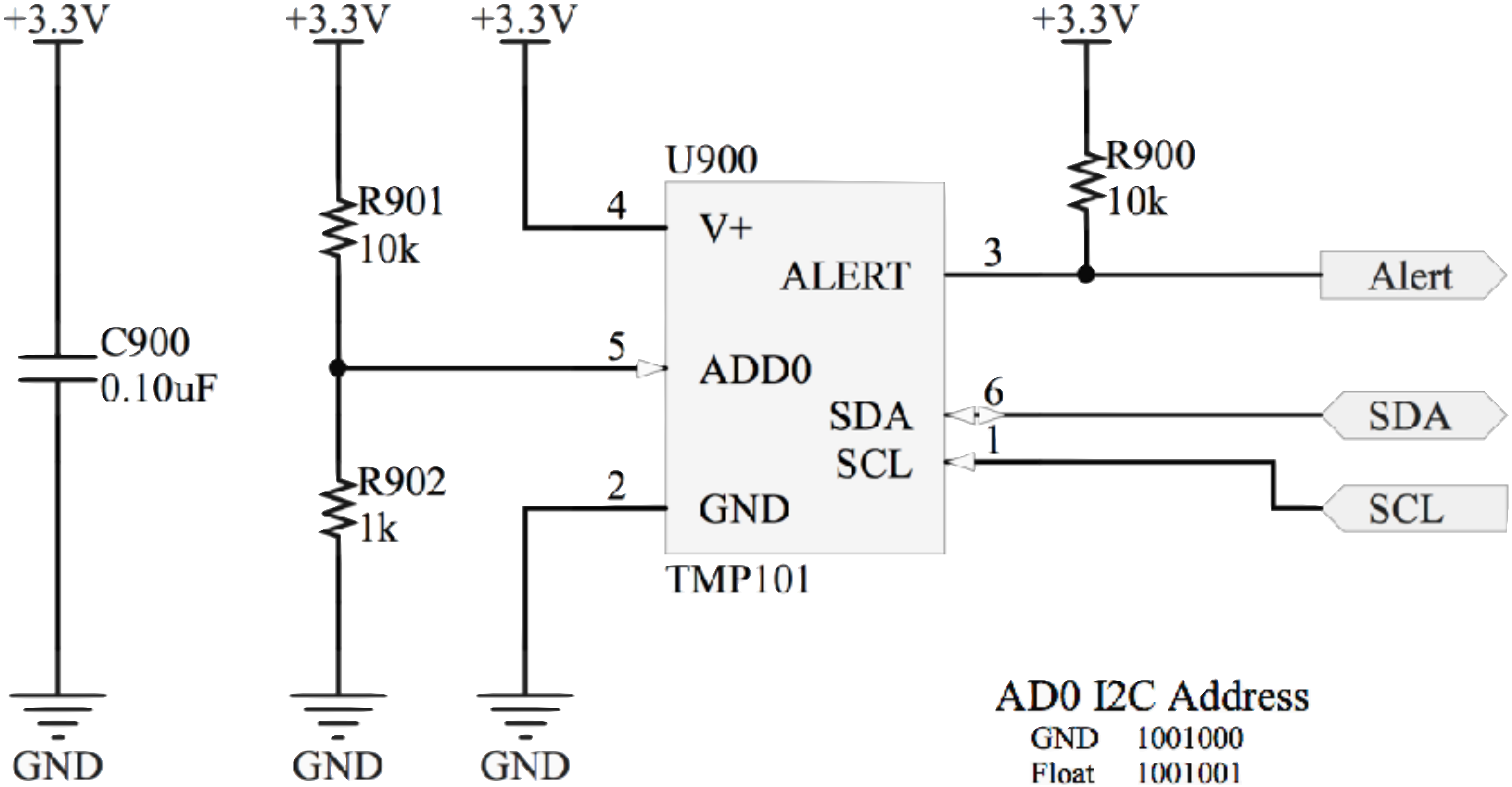
UART Mux


+3.3V
C700
0.10uF
GND
VSYS
U700
P700
VCC
1
USB_Tx
2
NO1
COM1
3
Tx
SOC_Tx
5
NC1
Rx
SEL1
4
SEL_TX
USB_Rx
10
NO2
COM2
9
SOC_Rx
7
NC2
SEL2
8
SEL_RX
6
GND
TS3A5223
GND
1
2
3
4
Header 4
GND

Voltage Regulators



Temperature Sensor

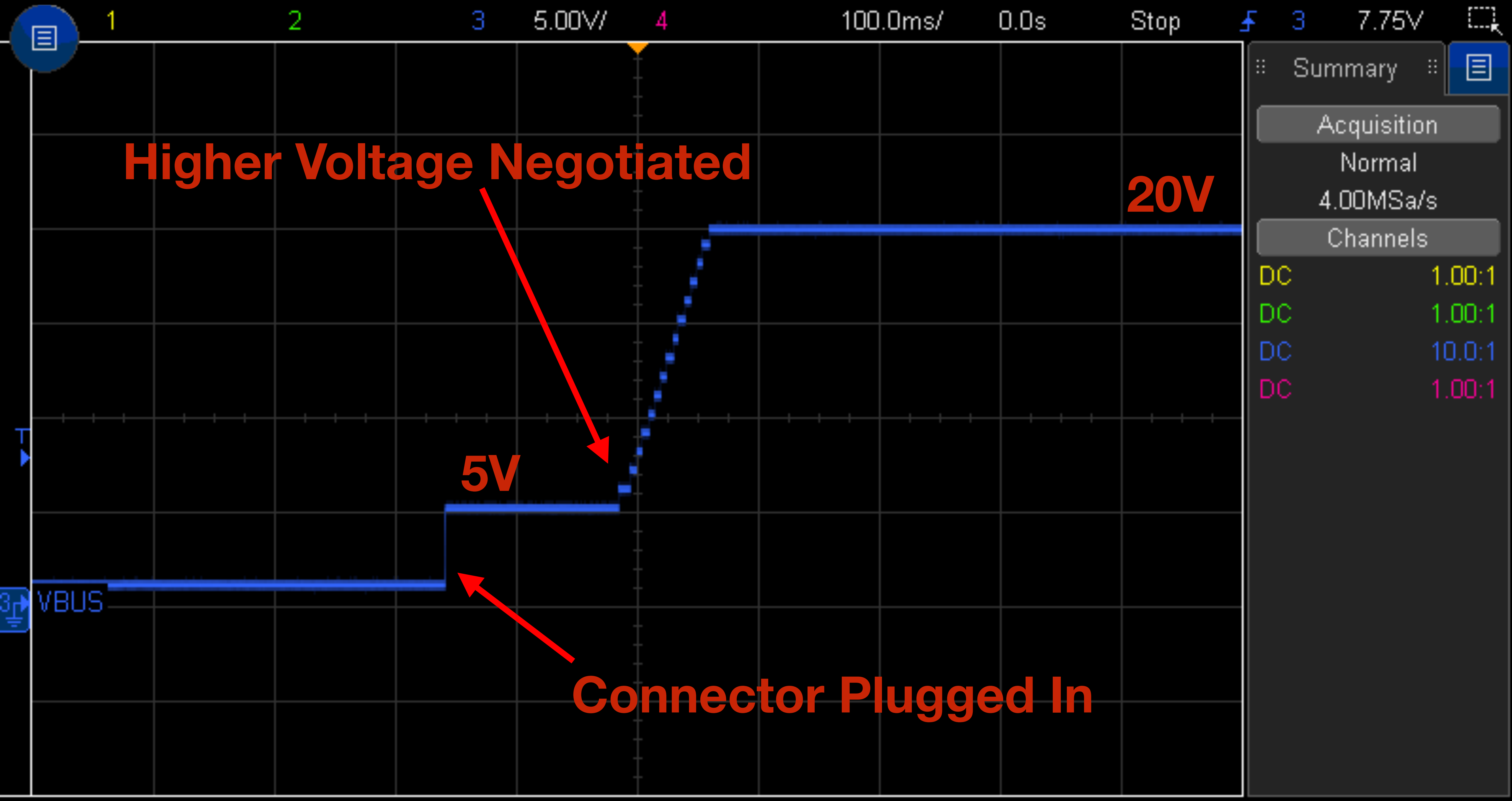


AD0	I2C Address
GND	1001000
Float	1001001
V+	1001010

Requirements

Block	Points
Battery Management System	15
Bluetooth Low Energy	10
USB Power Delivery	10
USB Serial Converter	5
Radiofrequency Front End	5
Lithium-Ion Battery	5

Additional Results



Higher Voltage Negotiated

20V

5V

Connector Plugged In

1 2 3 5.00V/ 4 100.0ms/ 0.0s Stop 3 7.75V

Summary

Acquisition
Normal
4.00MSa/s

Channels

DC	1.00:1
DC	1.00:1
DC	10.0:1
DC	1.00:1