|  |  |
| --- | --- |
| Test Case | 1.0 |
| Test Type | Functional/Hardware |
| Test Name | Power-On |
| Test Purpose | Bring-up the power regulator circuitry safely and check verify that the board is receiving 3.3v correctly. |
| Tools Required | 1. DC power supply (with current limiting functionality). 2. Multimeter. |
| Steps Required | 1. Set DC power supply to 4.2v and limited to 100mA. 2. Connect DC power supply to VBAT and GND planes on the board. 3. Connect multimeter to measure voltage on the 3.3v plane. 4. Turn on the output of the DC power supply and observe the current output. If the current is maxing out at 100mA there may be a short or a badly placed IC. 5. Observe the voltage reading from the DC power supply. It should close to 3.3v. |
| Test Result  (P/F) | Pass! Yay! |
| Observations? | First attempt was a fail. The power regulator output was ~0v. We resoldered the regulator IC and tried again.  Second attempt was a fail. The power regulator was outputting 4.56v. We found the issue. The voltage divider circuit was telling the IC to output 4.56v. The voltage divider was using a 470kohm and 100kohm resistor. We changed the voltage divider to 470kohms and 150kohms.  Third attempt was a success! We observed ~40mA being drawn from the DC power supply. This is well within a safe region and indicates that we do not have any shorting. |

|  |  |
| --- | --- |
| Test Case | 2.0 |
| Test Type | Functional/Unit |
| Test Name | CAN |
| Test Purpose | Purpose of the test is to confirm that we can receive correct data over CAN |
| Tools Required | 1. CAN client software required to send CAN messages from laptop to transceiver. 2. Minicom needs to be installed for serial output. |
| SW document name | app\_test\_CAN |
| Steps Required | 1. Build and load app\_test\_CAN code to the microcontroller. 2. Send CAN message from client SW to transceiver. 3. Print sent message on serial monitor using minicom. 4. Message on the serial monitor should be the same message sent from CAN client software. |
| Test Result  (P/F) |  |
| Observations? |  |

|  |  |
| --- | --- |
| Test Case | 2.1 |
| Test Type | Functional/Unit |
| Test Name | DC motor test. |
| Test Purpose | Spin DC motor without current feedback. |
| Tools | 1. ARM toolchain. 2. OpenOCD |
| SW document name | app\_test\_brushed |
| Steps Required | 1. Build and load app\_test\_brushed code to the microcontroller. 2. Motor should be spinning.    1. Confirm that the motor can speed up and down.    2. Confirm the motor can switch directions. |
| Test Result  (P/F) | PASS |
| Observations? | 1. The motor spins both directions and changes speed. 2. There is an error in the Rev1 flat-sat board layout. There is no connection between the 0ohm and NP resistors on the N$20 net. We had to create a solder bridge here so that the Vs pin on the STSPIN250 can receive VBATT. |

|  |  |
| --- | --- |
| Test Case | 2.1.0 |
| Test Type | Functional/Integration |
| Test Name | DC Current Control |
| Test Purpose | Spin DC motor **with** current feedback. |
| Tools | 1. ARM toolchain. 2. OpenOCD. 3. Minicom |
| SW document name | app\_test\_brushed\_adc |
| Steps Required | 1. Make sure the serial connections are in place. 2. Build and load app\_test\_brushed\_adc code to the microcontroller. 3. Motor should be spinning moments after loading code.    1. Confirm that the ADC is reading expected current value.    2. **(add info for exact value expected)** |
| Test Result  (P/F) |  |
| Observations? |  |

|  |  |
| --- | --- |
| Test Case | 2.2 |
| Test Type | Functional/Unit |
| Test Name | Read encoder output. |
| Test Purpose | Observe the encoder is providing correct output from corrected angle register. |
| Tools | 1. ARM toolchain. 2. OpenOCD. 3. Minicom |
| SW document name | app\_test\_encoder |
| Steps Required | 1. Make sure the serial connections are in place. 2. Build and load app\_test\_encoder code to the microcontroller. 3. Spin the motor manually and observe that the output ranges from 0 to 16383. |
| Test Result  (P/F) | PASS |
| Observations? | 1. Reacts to a magnet consistently. |

|  |  |
| --- | --- |
| Test Case | 2.3 |
| Test Type | Functional/Unit |
| Test Name | BLDC Motor Test |
| Test Purpose | Observe the bldc motor spins via brute force method. |
| Tools | 1. ARM toolchain. 2. OpenOCD. |
| SW document name | app\_test\_bldc |
| Steps Required | 1. Build and load app\_test\_bldc code to the microcontroller. 2. The motor must be spinning moments after the mcu is flashed. |
| Test Result  (P/F) | PASS |
| Observations? | 1. Functions as expected. |

|  |  |
| --- | --- |
| Test Case | 2.3.0 |
| Test Type | Functional/Integration |
| Test Name | BLDC Current control test |
| Test Purpose | Observe ADC output while bldc motor is spinning. |
| Tools | 1. ARM toolchain 2. OpenOCD 3. Minicom |
| SW document name | app\_test\_bldc\_adc |
| Steps Required | 1. Make sure the serial connections are in place. 2. Build and load app\_test\_bldc\_adc code to the microcontroller. 3. The motor must be spinning moments after the mcu is flashed. 4. Observe the ADC output through minicom. |
| Test Result  (P/F) |  |
| Observations? |  |

|  |  |
| --- | --- |
| Test Case | 2.4 |
| Test Type | Integration |
| Test Name | BLDC Integration |
| Test Purpose | Spin the motor using encoder feedback while ADC provides voltage. |
| Tools | 1. ARM toolchain 2. OpenOCD 3. Minicom |
| SW document name | app\_test\_bldc\_integrated |
| Steps Required | 1. Make sure the serial connections are in place. 2. Build and load app\_test\_bldc\_integrated code to the microcontroller. 3. The motor must be spinning moments after the mcu is flashed. 4. Observe that the output ranges from 0 to 16383. 5. Observe output from the ADC |
| Test Result  (P/F) | PASS |
| Observations? | 1. Encoder feedback works with motor. |

|  |  |
| --- | --- |
| Test Case | 2.5 |
| Test Type | Integration |
| Test Name | System test |
| Test Purpose | Ensure the entire system functions. |
| Tools | 1. ARM toolchain 2. OpenOCD 3. Minicom |
| SW document name | app\_test\_acs |
| Steps Required | 1. Make sure the serial connections are in place. 2. Build and load app\_test\_acs code to the microcontroller. 3. Send CAN message to activate BLDC motor.    1. The motor must be spinning    2. Observe that the output ranges from 0 to 16383.    3. Observe output from the ADC 4. Send CAN message to shutdown BLDC motor.    1. The motor must stop spinning. 5. Send CAN message to start DC motor.    1. DC motor must be spinning.    2. Observe ADC is reading voltage. |
| Test Result  (P/F) |  |
| Observations? |  |