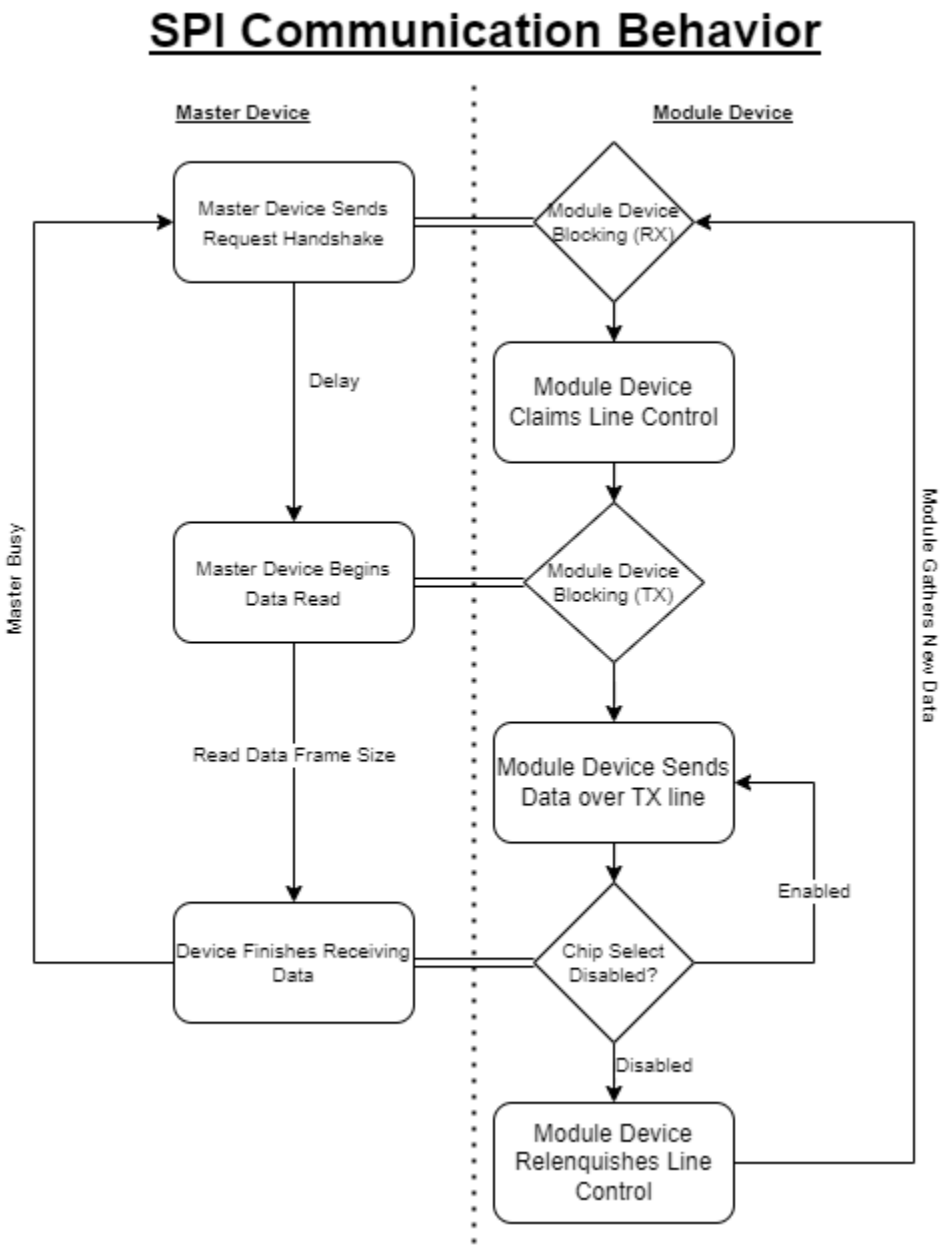
Alpha Test Plan – Prototroller

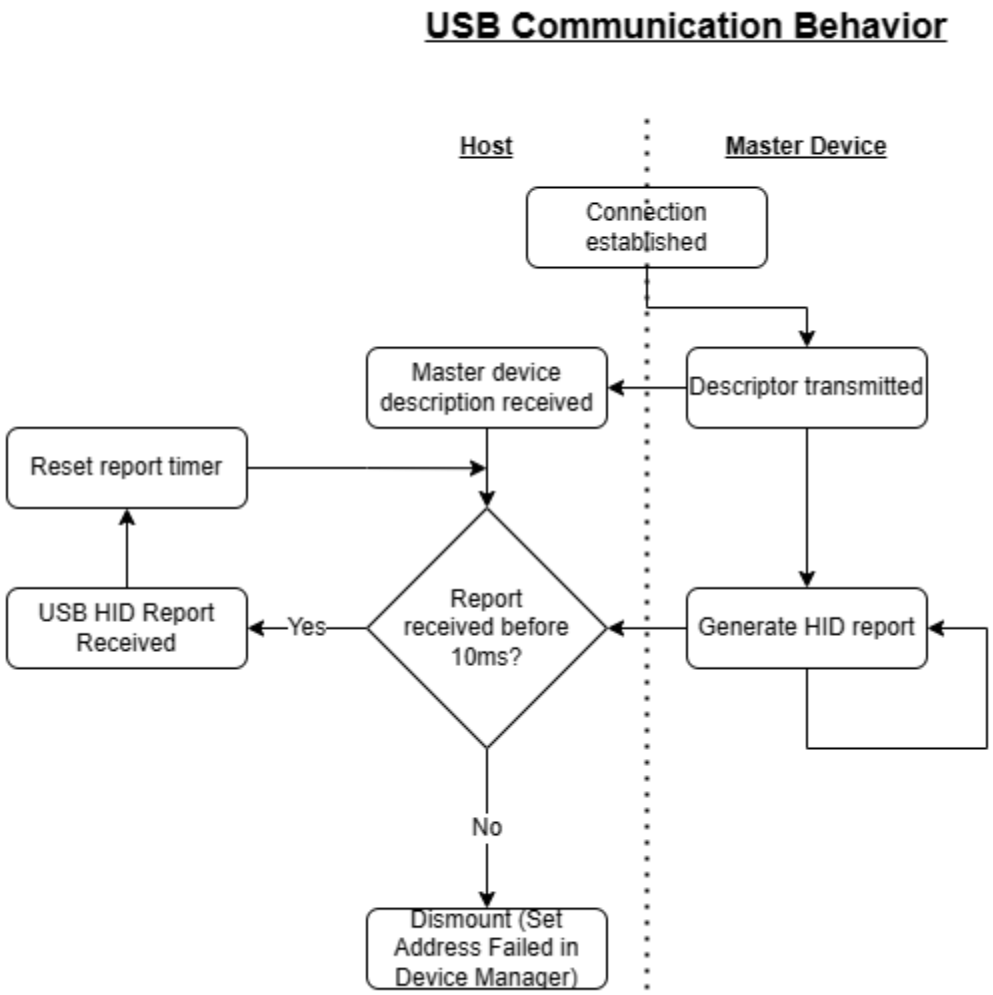
Merrick R., Britton M., Caleb O., Evan Z., Yu-yang H.

Expected Behavior

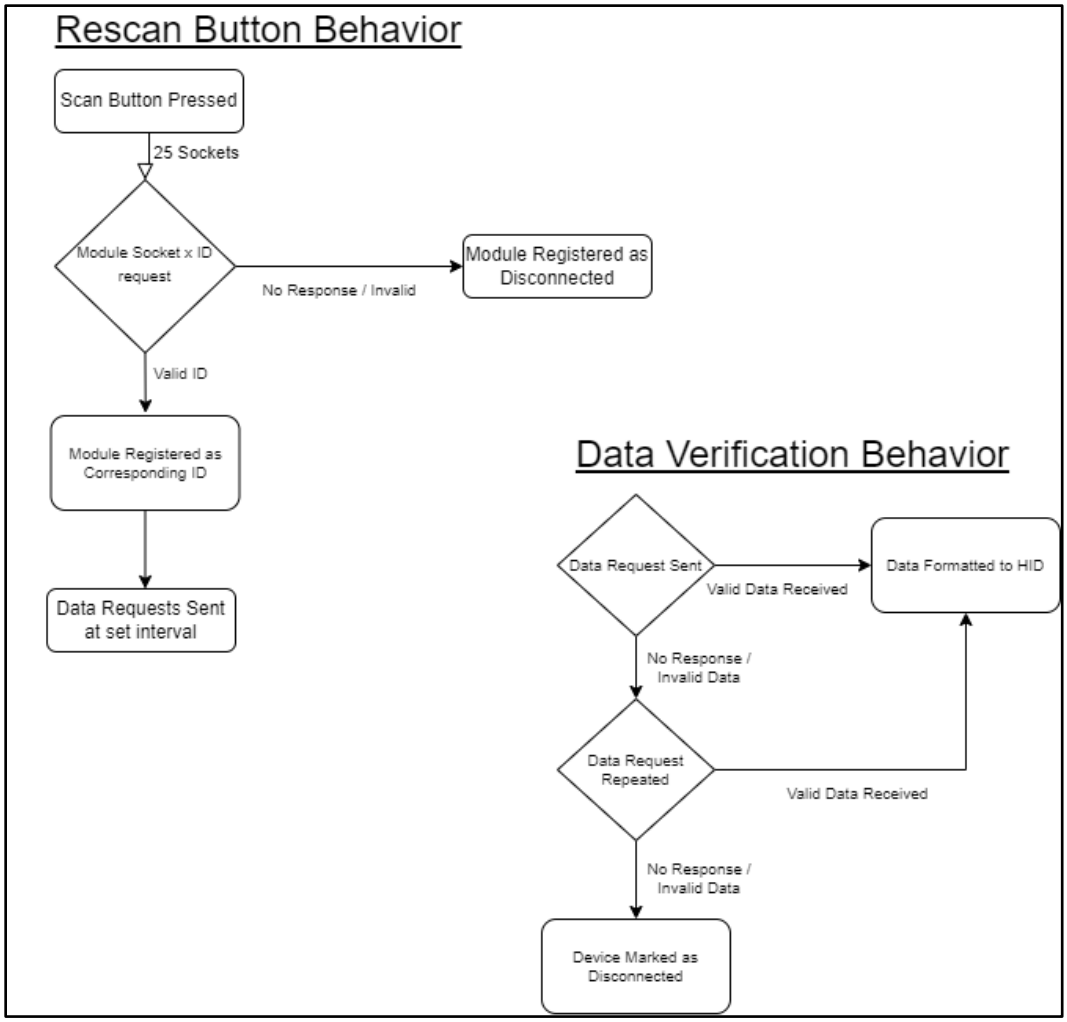
Master ⇔ Module SPI Communication Expected Behaviors



Host ⇔ Master USB Communication / HID Report Expected Behaviors



Module Rescanning / Data Verification Expected Behaviors



Additionally, most logical blocks of source-code are documented with functional purpose, parameters, edge-cases, etc.

Test Procedures

Firmware Behavioral Testing (Debug Firmware)

Master Device	
Test	**Expected Result**
Device Boots and outputs serial (console) data	Visible COM Port when using debug firmware version
Device completes initial scan	Console output of connected modules
Device Rescan Button begins scan	Console output of rescanned modules
Device receives correct data packets	Console output displays proper data/state of module device
Device maintains module connection	Console does not display disconnect messages
Module Device	
Test	**Expected Result**
Device Boots and outputs serial (console) data	Visible COM Port when using debug firmware version
Device reads ADC/GPIO data	Console output of raw data
Device calculates computation of data	Console output of computed data
Device sends data when requested	Console output of successful transmission
Device blocks when data not requested	Console does not display additional messages
Data read is within accurate bounds	Console outputs acceptable data value (tolerance differs based on module type)

Verification of Average Power Draw by Software Measurement

Test Procedures	**Expected Results (0 Modules)**	**Expected Results (*n* Modules)**
1. Choose USB port to plug Prototroller into on Windows host	N/A	N/A
2. Measure power consumption for this port using USBDeview	Consuming 0mA	Consuming 0mA
3. Plug Prototroller into host	N/A	Windows does not report "Power Limit Exceeded"
4. Measure power consumption for port using USBDeview	Consuming ≤ 50mA	Consuming ≤ 500mA (following $n * 20$mA line)

Verification of Module Rescanning

Test Procedures	Expected Results
1. Ensure Prototroller is plugged into host with no connected modules	N/A
2. Establish a serial connection to the master board with a console such as PuTTY	Output is displayed on serial console
3. Press and release the rescan button	Module rescan initiated and all modules identified as disconnected on serial console
4. Plug-in a module with an invalid ID to a random slot *k*	No change on serial console
5. Press and release the rescan button	Module rescan initiated, no change on serial console (module *k* shown as disconnected).
6. Plug-in a module with a valid ID to another random slot *m*	No change on serial console
7. Press and release the rescan button	Module rescan initiated, only module *m* shown as connected with corresponding ID
8. Observe serial console	Data from module in slot *m* is output
9. Disconnect the module in slot *m*	No output on serial console
10. Press and release the rescan button	Module rescan initiated; all modules shown as disconnected
11. Disconnect the module in slot *k*	No change on serial console
12. Repeat 6-9 until every slot has been evaluated	See 6-9 expected results

Verification of SPI Communication

Test	Expected Result
Probe Rx and Tx bus lines for master board with oscilloscope triggers	Handshake is sent, and data is correctly received
Probe CS bus lines for master board with oscilloscope triggers	All modules currently connected are visibly shown on the oscilloscope

Verification of HID Drivers

Test	Expected Result
Use Device Monitoring Studio to view HID reports	All data packets via reports are correct, and are consistent with module board inputs

Verification of Latency Constraints

Measuring the latency of controllers in general is difficult, but not impossible with the right hardware. In our case, the methodology is to aim a high-speed camera at our host display and a LED connected to a button module mapped to a left-click schema. Then we can compare – in editing software - the delta time between the LED lighting and a UI element on the display changing in response to the left-click. Preferably, use a monitor with 144+Hz refresh rate for a more accurate measurement.

Test Procedures	**Expected Results**
1. Plug in Prototroller to host with one module connected: button, with left-click schema. Connect Switch => LED => Resistor => +3V3	N/A
2. Press and release the rescan button	N/A
3. Set up mouse on element to be left-clicked (action must modify UI in some way)	N/A
4. Start recording with 240fps, giving smallest possible increment of 4.16ms	N/A
5. Press and release the button	LED flashes for as long as the button is pressed.
6. Stop recording	N/A
7. In post-processing software, find the frame the LED lights up and the frame the UI responds.	N/A
8. Convert the difference in frames to a response time: #frames * (1/240)	Response time should be ≤ 10ms