Overview

The USBHub3+ is an 8-port software-programmable USB 3.1 (Gen1; 5 Gbps) hub designed for demanding industrial environments where advanced control and monitoring of USB ports is required. Ideal for testing or development environments where standard "always-on" behavior of a consumer-grade USB hub is not desirable.

Software control of the USBHub3+ is established and maintained over one of two available upstream-facing host ports or via a dedicated Control Port connection.

The USBHub3+ can be used to enable/disable individual USB ports, measure current or voltage on downstream USB ports, set programmable current limits, set USB charging protocol behavior and otherwise automate USB port behaviors in development and testing.

Typical applications include:

- USB device manufacturing
- USB device validation and development
- Plug/unplug cycle testing
- Functional testing
- Battery charging
- USB device resets
- USB power monitoring
- Controlling USB device enumeration sequence

Features

- Supports USB hosts and devices up to USB 3.1 (5Gbps)
- Individually enable/disable any of 8 downstream ports
- Measure voltage and current on each downstream port
- Hi-Speed data, SuperSpeed data and power lines can be separately enabled for each downstream port
- Set programmable current limits for each downstream port (up to 4A)
- Dedicated Control Port for software control; independent of the selected upstream port
- Automatic or programmed selection for either of 2 upstream-facing host port connections
- All ports support USB link speeds up to 5Gbps
- Detect established link speed on each port: SuperSpeed (5Gbps) or Hi-Speed (480Mbps)
- Selectively enable USB charging mode behaviors: SDP (Standard Downstream Port) or CDP (Charging Downstream Port) modes[1]
- Deliver up to 4.0A per port (in CDP mode)
- Set enumeration delay for discovery of attached downstream devices
- Backward compatible with USB 2.0 and USB 1.x devices
- Boost USB 2.0 upstream and downstream signal levels
- DIN-rail mountable
- Alternate Euro-style terminal block power input connector
- Certified to +/-15kV ESD strikes (IEC61000-4-2 level 4)
- Overvoltage and reverse current protected V_{bus} outputs

Description

The USBHub3+ gives engineers advanced flexibility and configurability over USB ports in testing and development applications.

The USBHub3+ hub architecture consists of two layers of internal hubs to achieve true 8-port hub functionality.

Each downstream USB channel implements separately and independently switched data lines and current-limited power lines. USB power, data and SS data can be independently disconnected for advanced USB testing applications. Pin interfaces are protected against reverse polarity and over-voltage. Connections are designed to operate from 0°C to 50°C ambient with no external cooling or fans.

Each USBHub3+ is uniquely addressable and controllable from a host PC via the selected USB host input or through a dedicated Control Port. Acroname's BrainStem™ link is then established over the USB input and allows a connection to the on-board controller in the USBHub3+. USBHub3+ can be controlled via a host running BrainStem APIs or alternately, it can operate independently by running locally embedded, user-defined programs based on Acroname's BrainStem Reflex language.

[1] See http://www.usb.org/developers/docs/devclass_docs/ under the category Battery Charging for full details.

Absolute Maximum Ratings

Stresses beyond those listed under ABSOLUTE MAXIMUM RATINGS can cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions beyond those indicated under RECOMMENDED OPERATING CONDITIONS is not implied. Exposure to absolute-maximum rated conditions for extended periods affects device reliability and may permanently damage the device.

Parameter	Minimum	Maximum	Units
Input Voltage, V_{supply}	0.0	36.0	V
Input Power		85	W
V_{bus} Ouput Power		65	W
Voltage on any V_{bus} line, upstream and downstream	0.0	5.1	V
Voltage on any USB D+/D-, upstream and downstream	-0.3	5.1	V

Table 1: Absolute Maximum Ratings

Handling Ratings

Parameter	Conditions/Notes	Minimum	Typical	Maximum	Units
Ambient Operating Temperature, T_A	Non-Condensing	0.0	25.0	50.0	°C
Relative Humidity Range	Non-Condensing	5	-	95	%RH
Storage Temperature, T_{STG}		-10.0	-	85.0	°C
Electrostatic Discharge, V_{ESD}	Meets IEC 61000-4-2, level 4, air-discharge	-15	-	+15	kV
	Meets IEC 61000-4-2, level 4, contact-discharge	-8	-	+8	kV

Table 2: Handling Ratings

Recommended Operating Ratings

Specifications are valid at 25°C unless otherwise noted. Intended for indoor use only.

Parameter	Conditions/Notes	Minimum	Typical	Maximum	Units
Input Voltage, V_{supply}		9.0	12.0	24.0	V
USB V_{bus} on downstream ports	Hub powered; Port power enabled	4.5	5.0	13.2	V
	Hub powered; Port power disabled	0.0	0.0	13.2	V
Relative Humidity Range	Non-Condensing	5	-	80	%RH

Table 3: Recommended Operating Ratings

Block Diagram

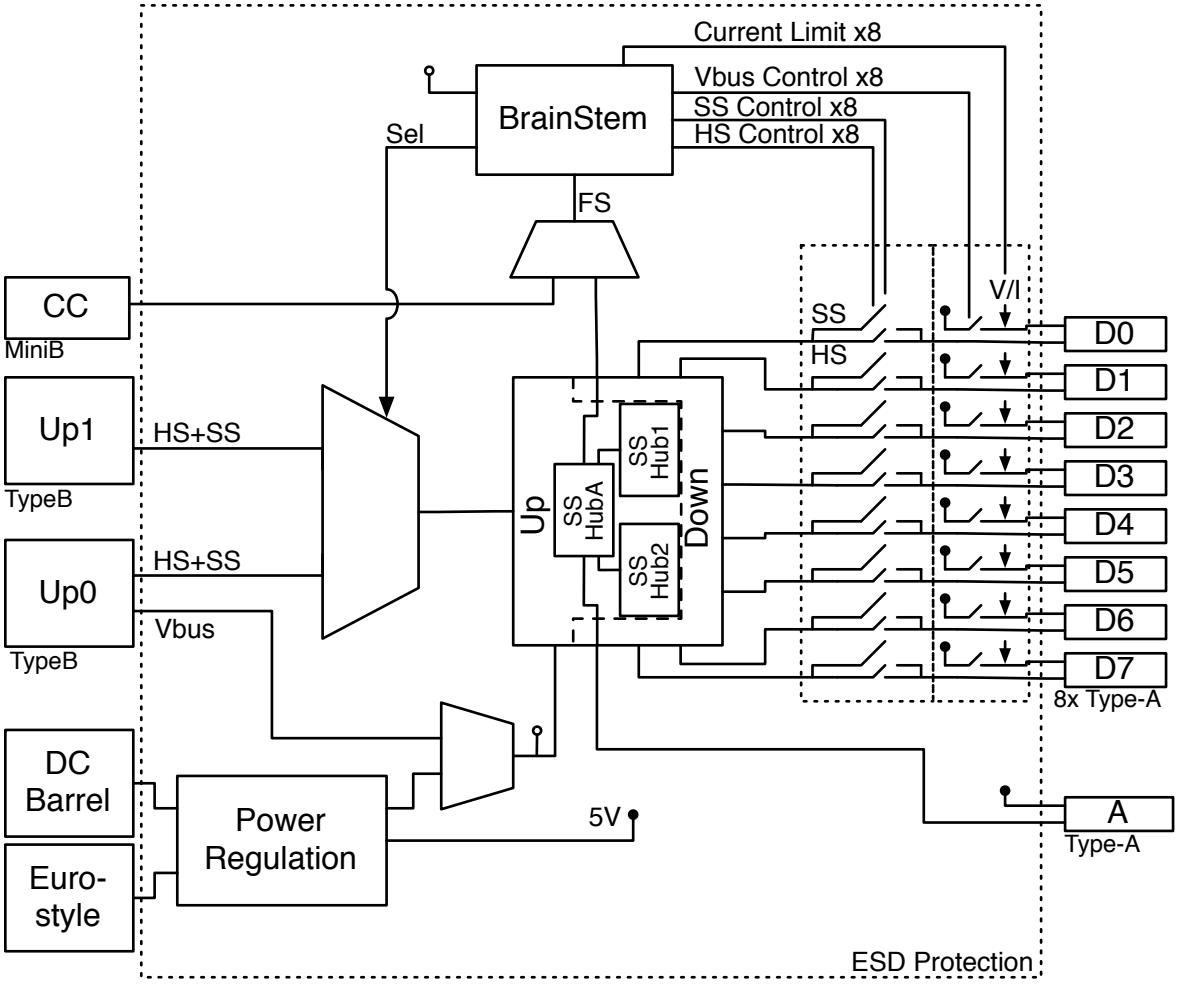


Figure 1: USBHub3+ Block Diagram

Typical Performance Characteristics

Specifications are valid at 25°C unless otherwise noted. Indoor application use only. Sample rates are typically limited by the USB throughput of the host operating system except where bulk capture is supported.

Parameter	Conditions/Notes	Min	Typ.	Max	Units
Input Power, W_{supply}, no downstream devices attached		-	2.2	-	W
V_{supply} Under Voltage Lockout (UVLO)		7.5	8.0	8.2	V
V_{supply} Over Voltage Lockout (OVLO)		26.0	26.8	27.8	V
Wide Input Range System Efficiency	At 12.0V input, nominal 8A load[2]	84	-	92	%
USB Downstream Output Voltage, V_{bus}	No load on downstream USB ports	4.947	5.10	5.25	V
V_{bus} Measurement Resolution		-	8.0	-	mV
V_{bus} Measurement Accuracy		-2.0	-	2.0	%
V_{bus} Short-circuit Trip Current, I_{limit}		4.8	5.0	5.4	A
V_{bus} Short-circuit Trip Time, t_{limit}		-	0.7	-	µS
V_{bus} Short-circuit Average Current, I_{short}	After trip	0.3	0.5	2.0	A
V_{bus} Current Measurement Resolution		-	1.0	-	mA
V_{bus} Current Measurement Accuracy	V_{bus} current < 4.00A	-1.0	-	1.0	%
V_{bus} Current Measurement Range		0	-	4095	mA
V_{bus} Current Limit Trip Point Range	Software programmable	0	-	4095	mA
V_{bus} Current Limit Trip Point Resolution		-	1.0	-	mA
V_{bus} Overcurrent Trip Time t_{trip}	Time from overcurrent load to port power switch disconnect.	.20	1.0	3.4	ms
USB SuperSpeed Data Rate	May depend on host or devices	-	-	5	Gbps
USB Hi-Speed Data Rate	May depend on host or devices	-	-	480	Mbps
V_{bus} Current Supply (SDP mode)	USB 2.0 data lines disabled or no USB host present, device limited	-	100	-	mA
V_{bus} Current Supply (SDP mode)	USB 2.0 data lines enabled and USB host present, device limited	-	500	-	mA
V_{bus} Current Supply (CDP mode)	USB 2.0 data lines enabled, USB host present, device limited	-	1500	-	mA
V_{bus} Current Supply (DCP mode)	USB 2.0 data lines enabled, no USB host present, device limited	-	5000	-	mA
Input current CH0 upstream port, I_{Vbus}	No V_{supply} present, USB 2.0 type-B cable	-	180	-	mA
Input current CH0 upstream port, I_{Vbus}	No V_{supply} present, USB 3.0 type-B cable	-	425	-	mA
Input current measurement resolution, I_{supply}	Through barrel jack or Euro-style connector only	-	4.0	-	mA
I_{supply} accuracy		-2.0	-	2.0	%
Input voltage measurement resolution, V_{supply}	Through barrel jack or Euro-style connector only	-	8.0	-	mV
V_{supply} measurement accuracy		-2.0	-	2.0	%

Table 4: Typical Performance Characteristics

[2] Representative 8A load based on 8 USB downstream devices running in CDP mode consuming approximately 1.0A each.

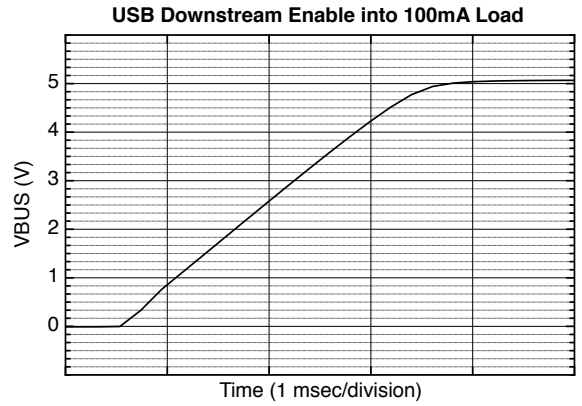


Figure 2: V_{bus} rise time after with 100mA constant load.

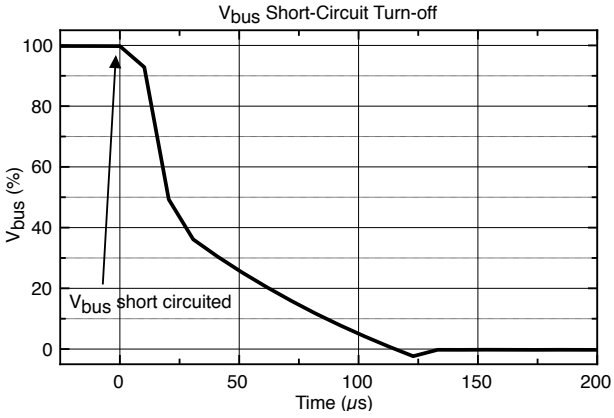


Figure 3: V_{bus} turn-off time after short-circuit.

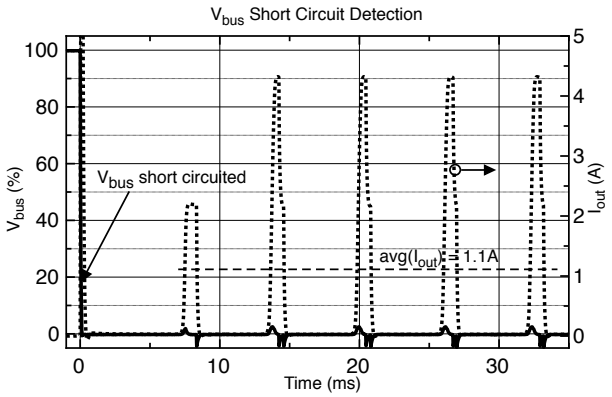


Figure 4: V_{bus} short-circuit mode behavior.

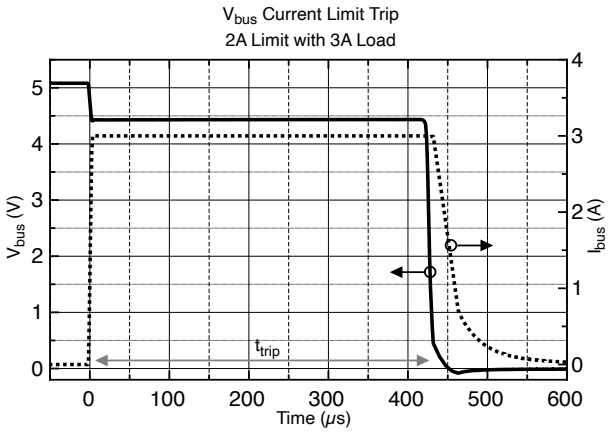


Figure 5: V_{bus} programmable current limit response, t_{trip}.

Module Hardware and Software Default Values

The USBHub3p leverages a hardware-specific subset of BrainStem Entity implementations. The `aUSBHub3p.h` C++ header file includes macro definitions for many parameters specific to the USBHub3p. Table 5: USBHub3p Hardware and Software Default Values provides an overview of these values.

Parameter	Index	Macro Name or Implemented Options	Notes
Module Definitions:			
Module Base Address	6	aUSBHUB3P_MODULE_ADDRESS	
Router Base Address	6		
Entity Class Definitions:			
`timer` Entity Quantity	8	aUSBHUB3P_NUM_TIMERS	
`usb` Entity Quantity	1	aUSBHUB3P_NUM_USB	
`store` Entity Quantity	2	aUSBHUB3P_NUM_STORES	
`system` Entity Quantity	1		
`app` Entity Quantity	4	aUSBHUB3P_NUM_APPS	
`pointer` Entity Quantity	4	aUSBHUB3P_NUM_POINTERS	

Table 5: USBHub3p Hardware and Software Default Values[3]

Overview

With advanced programmable control features, the USBHub3+ is targeted at industrial and production systems requiring software and embedded control of USB devices. It is ideal for manufacturing test of consumer electronics and control of industrial systems.

Power Subsystem

An over-voltage, under-voltage, over-temperature, current limited switch mode power supply regulated input supply power to generate V_{bus} for all downstream ports as well as the USBHub3+'s system power. As noted in the system block diagram, the system power can also come from Up0's V_{bus}. However, if powering the system from Up0, no downstream port power is available. This feature is useful for programming and testing the hub's features without the need of an external power supply.

Grounding

USB is sensitive to grounding and ground loops. Is important to understand your system's grounding strategy to ensure the USB shield or ground is not the primary ground current return path. The USBHub3+ shorts USB shield and ground and provides an earth ground connection point. Instructions for separating shield and earth ground from USB and system ground can be obtained by contacting Acroname support.

Ground isolation methods such as optical isolators and isolated power supplies have been demonstrated to work well with the USBHub3+.

Bus Measurement Subsystem

The measurement subsystem of the USBHUb3+ samples V_{bus} voltage and current for all 8 programmable downstream ports and the input voltage. Voltage and current measurements for a single port are done sequentially and interleaved with 2 other port measurements. Each sample of voltage or current takes 140µs. For most USB applications these measurements are considered "instantaneous" and "simultaneous" so they make a reasonable measurement of port power consumption. Measured samples of voltage or current are then averaged in a first-order IIR exponential moving average filter with a decay factor of 4. The sampling time diagram and averaging filter are illustrated in Figure 6 and Figure 7.

Average filter operation begins by converting a sample of voltage or current for one port. This value is then subtracted from the previous value that was present in the corresponding measurement output. This difference is then divided by 4 and stored in an internal accumulation register. The computed measurement result is then added to the previous measurement

[3] Refer to `aUSBHub3p.h` within the BrainStem Development Kit download for actual file.

output value, and the resulting value is updated as the new measurement output. After the update, the next signal to be measured follows the same process.

The filter response to a step in the input takes 8 samples (6.7ms) to be within 10% of the input value; 16 samples (13.4ms) to be within 1% of the input value.

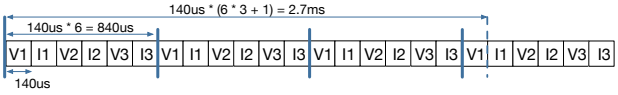


Figure 6: Measurement sample timing diagram

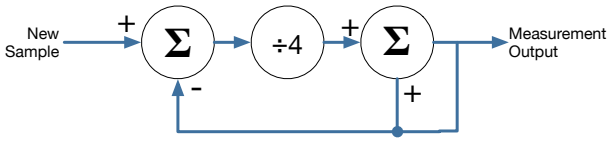


Figure 7: Averaging filter flow

Current Limiting Behavior

There are two current limits in the USBHub3+: the programmable limit and short-circuit limit. The programmable current limit sets a circuit-breaker style trip point which disables the port if the current goes above the set limit. V_{bus} current for this trip point is based on individual samples, not the averaged samples. As described earlier, each sample takes 140µs for conversion. If any single current sample on a port is higher than the programmed limit for that port, the port trips off. When a port trips, the USBHub3+ sets the overcurrent flag (see USB Port Error Status Mapping) and disables the port after t_{trip}. A device is allowed to draw up to the short-circuit current limit until the port is disabled. This behavior is detailed in Figure 5.

A short-circuit on V_{bus} is current limited to I_{limit}. Any current over I_{limit} will cause V_{bus} to enter short-circuit mode after t_{limit}. Higher current may flow within t_{limit}. After entering short-circuit mode, assuming the programmable current limit is not tripped, the port will supply an average of I_{short}. The short-circuit behavior is detailed in Figure 3 and Figure 4. Short-circuit mode will continue until the short-circuit is removed or the load current drops below I_{limit}.

Software Control

Software control of the features of the USBHub3+ is done with the BrainStem API via a BrainStem link. BrainStem links are done over USB and can be established via upstream port 0 (Up0), upstream port 1 (Up1), or the Control Port. After one or more of these ports is connected to a host machine, a user can connect to it via software API:

```
stem.link.discoverAndConnect(USB)
```

When multiple hubs are connected to a host, connecting to a specific hub can be done by providing the hub serial number. Further, all connected hubs can be found using

```
brainstem.discover.findAllModules(USB)
```

BrainStem Control Port

The USBHub3+ also has a dedicated control channel on the USB mini-B connector. This is a full-speed USB 2.0 connection for BrainStem interface only. No USB hub traffic can flow on this connection. When a cable is connected to the mini-B connector, the BrainStem link can only be established through the Control Port, independent of the selected upstream port. The USB 3.0 type-B connectors are then used only for USB hub traffic to connect downstream USB devices. When the Control Port is not used, the BrainStem link will share the active upstream USB connection. Using the Control Port provides the ability to completely disconnect both USB upstream host connections while maintaining sofware control of the hub. Figure 8 details the decision tree for selecting the port used to establish a BrainStem link.

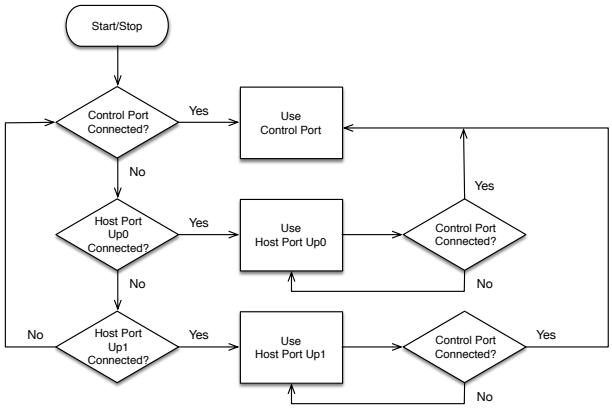


Figure 8: Determining connection used for BrainStem communications

Using Multiple Hosts with USBHub3+

The two upstream-facing host ports can be connected to two different host computers. The control port can be attached to no computer, one of the same computers attached to the upstream ports, or a third host computer. Due to limitations of USB specification, only one host computer can access downstream USB ports at any time. Through the BrainStem API, the upstream port used can be controlled, or the system can automatically select the upstream port (see USB Hub Upstream Mode). When automatically selecting the upstream port, the USBHub3+ will favor using Up0 if it is connected.

Device Drivers

The USBHub3+ leverages operating system user space interfaces that do not require custom drivers for operation on modern operating systems.

Some older operating systems may require the installation of a BrainStem USB driver to enable software control. Installation details on installing USB drivers can be found within the BrainStem Development Kit under the "drivers" folder. For example, Windows 7 requires the supplied INF to communicate with BrainStem USB devices.

Capabilities and Interfaces

The USBHub3+ is built on Acroname's BrainStem system, which provides simple high-level APIs, a real-time embedded runtime engine and modular expandability. Functionality details unique to the USBHub3+ are described in the following sections. Please refer to Table 15: Supported USBHub3+ BrainStem Entity API Methods for a complete list of all available API functionality. All shortened code snippets are loosely based on the C++ method calls and meant to be psuedocode – Python and Reflex are virtually the same. Please consult the BrainStem Reference for implementation details.[4]

System Entities

Every BrainStem module includes a single System Entity. The System Entity allows access to configuration settings such as the module address, input voltage, control over the user LED and many more.

Serial Number

Every USBHub3+ is assigned a unique serial number at the factory. This facilitates an arbitrary number of USBHub3+ devices attached to a host computer. The following method call can retrieve the unique serial number for each device.

```
stem.system.getSerialNumber(serialNumber)
```

Module Default Base Address

BrainStems are designed to be able to form a reactive, extensible network. All BrainStem modules come with a default network base address for identification on the BrainStem network bus. The default module base address for USBHub3+ is factory-set as 6, and can be accessed with:

```
stem.system.getModule(module)
```

Saving USB Entity Settings

Some entities can be configured and saved to non-volatile memory. This allows a user to modify the startup and operational behavior for the USBHub3+ away from the factory default settings. Saving system settings preserves the settings as the new default. Most changes to system settings require a save and reboot before taking effect. For example, upstream and downstream USB Boost settings will not take effect unless a system save operation is completed, followed by a reset or power cycle. Use the following command to save changes to system settings before reboot:

```
stem.system.save()
```

Pressing the reset button two times within 5 seconds will return all settings to factory defaults: all ports' data (HS and SS) and power enabled, CDP mode, enumeration delay of 0, 4095mA current limit.

Saved Configurations	
Software Offset	I2C Rate
Router Address	Port Enumeration Delay
Boot Slot	Downstream Boost
Port Mode (SDP, CDP) – each port	Current Limit – per port
Upstream Boost	Port state (data and power)

Table 6: Saved Parameters

USB Entity

The `usb` entity provide a mechanism to control all functionality for the upstream and downstream USB ports.

USB Downstream Channels

Downstream USB channels can be manipulated through the `usb` entity command to enable and disable USB data and V_{bus} lines, measure current, measure V_{bus} voltage, boost data line signals, and measure temperature.

Manipulating Hi-Speed data, SuperSpeed data, and V_{bus} lines simultaneously for a single port can be done by calling the following methods with channel in [0-7] being the port index:

```
stem.usb.setPortEnable(channel)
stem.usb.setPortDisable(channel)
```

[4] See BrainStem software API reference at https://acroname.com/reference/ for further details about all BrainStem API methods and information.

Manipulating Hi-Speed data and SuperSpeed data lines while not affecting the V_{bus} lines simultaneously for a single port can be done by calling the following method with channel [0-7]:

```
stem.usb.setDataEnable(channel)
stem.usb.setDataDisable(channel)
```

Manipulating just the USB 2.0 Hi-Speed data lines for a single port can be done by calling the following method with channel [0-7]:

```
stem.usb.setHiSpeedDataEnable(channel)
stem.usb.setHiSpeedDataDisable(channel)
```

Manipulating just the USB 3.1 SuperSpeed data lines for a single port can be done by calling the following method with channel [0-7]:

```
stem.usb.setSuperSpeedDataEnable(channel)
stem.usb.setSuperSpeedDataDisable(channel)
```

Manipulating just the USB V_{bus} line for a single port can be done by calling the following method with channel [0-7]:

```
stem.usb.setPowerEnable(channel)
stem.usb.setPowerDisable(channel)
```

To affect multiple ports and lines simultaneously, see `usb.setHubMode()` later in this section.

The USB V_{bus} voltage, as well as the current consumed on V_{bus}, can be read for each channel by calling the following methods with channel [0-7], where the second variable passed into the method is the location for the measurement result:

```
stem.usb.getPortVoltage(channel,μV)
stem.usb.getPortCurrent(channel,μA)
```

Current-limit trip point settings can be accessed for each port by calling the following methods with channel [0-7], where the second variable passed into the method is either the set value or the write location of the result:

```
stem.usb.getPortCurrentLimit(channel,μA)
stem.usb.setPortCurrentLimit(channel,μA)
```

The enumeration state and speed of each downstream port can be read with

```
stem.usb.getDownstreamDataSpeed(ch,speed)
```

with ch in [0-7] and speed values retuned as:

Value	Hub Downstream Speed Descriptions
0	No device enumerated
1	Hi-Speed device enumerated
2	SuperSpeed device enumerated

Table 7: Hub downstream speed value descriptions

USB Downstream Operational Mode

The USB port operational mode controls the behavior of each downstream port's charging behavior. Each port can be setup to support different modes in the USB Battery Charge Specification 1.2 (BC1.2). Standard Downstream Port (SDP) mode will cause BC1.2 compliant or older USB devices to consume 500mA or less. Configuring a port as a Charging Downstream Port (CDP) will cause the hub signal to downstream devices that devices may consume up to 5A, the maximum allowed by BC1.2. If there is no upstream USB host connected to the hub, downstream ports set to CDP will behave as Dedicated Charging Ports (DCP).

The actual current consumed by the device is controlled by the downstream device and not the USBHub3+. Devices which are not compliant with BC1.2 or the previous USB power specifications may draw more current than specified above.

The operational mode is set or read by calling the methods:

```
stem.usb.setPortMode(channel,mode)
stem.usb.getPortMode(channel,mode)
```

Available options for Downstream Operational Mode are:

Value	Hub Port Mode Descriptions
0	Standard downstream port (SDP)
1	Charging downstream port (CDP)

Table 8: Hub port mode value descriptions

USB Downstream Enumeration Delay

Once a USB device is detected by the USBHub3+ it is possible to delay its connection to an upstream host computer and subsequent enumeration on the USB bus. The enumeration delay can mitigate or eliminate host kernel instabilities by forcing devices to enumerate in slow succession, allowing a focus on validation of drivers and software. The enumeration delay is configured in milliseconds, representing the time delay between enabling each successive downstream port from 0 to 7. Enumeration delay is applied when the hub powers on or when a new upstream connection is made.

```
stem.usb.setEnumerationDelay(delay)
stem.usb.getEnumerationDelay(delay)
```

USB Boost Mode

Boost mode increases the drive strength of the USB 2.0 Hi-Speed data signals (SuperSpeed data and power signals are not changed). Boosting the data signal drive strength may help to overcome connectivity issues when using long cables or connecting through relays, "pogo" pins or other adverse conditions. This setting is applied after a `system.save()` call and reset or power cycle of the hub. The system setting is persistent until changed or the hub is hard reset. After a hard reset, the default value of 0% boost is restored. A hard reset is done by pressing the "Reset" button on the back of the hub while the hub is powered.

Boost mode can be applied to both the upstream and downstream USB ports with the follow methods:

```
stem.usb.getDownstreamBoostMode(setting)
stem.usb.setDownstreamBoostMode(setting)
stem.usb.getUpstreamBoostMode(setting)
stem.usb.setUpstreamBoostMode(setting)
```

The setting parameter is an integer that correlates to the following:

Value	Hub Boost Mode Descriptions
0	Normal drive strength
1	4% increase in drive strength
2	8% increase in drive strength
3	12% increase in drive strength

Table 9: Hub boost mode value descriptions

USB Hub Upstream Channels

The USBHub3+ is perfect for environments where multiple devices need to be shared or switched between two host computers using two host (upstream) connections via USB standard-B connectors. The upstream connection can be automatically detected or specifically selected using the following methods:

```
stem.usb.getUpstreamMode(mode)
stem.usb.setUpstreamMode(mode)
```

The mode parameter can be defined as the following:

Value	Hub Upstream Mode Descriptions
0	Force upstream port 0 to be selected
1	Force upstream port 0 to be selected
2	Automatically detect upstream port
255	Disconnect both upstream ports

Table 10: Hub upstream mode value descriptions

Predefined C++ macros for these can be found in aProtocoldef.h, and Python's built-in help interface.

The default operational mode is to auto detect which upstream USB port is selected. Automatic detection uses the presence of V_{bus} on the USB type-B upstream connector to determine presence of a host. If only one upstream port is connected to a host, it will be used for upstream USB. If both upstream ports are connected, the hub will use upstream port 0.

If the Hub Upstream Mode is set to disconnect both upstream ports (or the only active upstream port), the only path available to establish a BrainStem link to the USBHub3+ will be via a host connected to the BrainStem Control Port. See Figure 8 for more details.

USB Hub Upstream State

The USBHub3+ can provide status information on which upstream port is actively selected as data path to the downstream ports:

```
stem.usb.getUpstreamState()
```

This command returns a 32-bit value which indicates:

Value	Hub Upstream State Descriptions
0	Upstream port 0 is actively selected
1	Upstream port 1 is actively selected
2	No upstream port is selected

Table 11: Hub upstream state value descriptions

USB Hub Operational Mode

In addition to targeting individual downstream USB ports, a bit-mapped hub state interface is also available. This interface allows the reading or setting of all USB downstream ports in one functional call.

```
stem.usb.getHubMode(mode)
stem.usb.setHubMode(mode)
```

The value mode is 32-bit word, defined as the following:

Bit	Hub Operational Mode Word Definition
0	USB Ch 0 USB Hi-Speed Data Enabled
1	USB Ch 0 USB V_{bus} Enabled
2	USB Ch 1 USB Hi-Speed Data Enabled
3	USB Ch 1 USB V_{bus} Enabled
4	USB Ch 2 USB Hi-Speed Data Enabled
5	USB Ch 2 USB V_{bus} Enabled
6	USB Ch 3 USB Hi-Speed Data Enabled
7	USB Ch 3 USB V_{bus} Enabled
8	USB Ch 4 USB Hi-Speed Data Enabled
9	USB Ch 4 USB V_{bus} Enabled
10	USB Ch 5 USB Hi-Speed Data Enabled
11	USB Ch 5 USB V_{bus} Enabled
12	USB Ch 6 USB Hi-Speed Data Enabled
13	USB Ch 6 USB V_{bus} Enabled
14	USB Ch 7 USB Hi-Speed Data Enabled
15	USB Ch 7 USB V_{bus} Enabled
16	USB Ch 0 USB SuperSpeed Data Enabled
17	Reserved
18	USB Ch 1 USB SuperSpeed Data Enabled
19	Reserved
20	USB Ch 2 USB SuperSpeed Data Enabled
21	Reserved
22	USB Ch 3 USB SuperSpeed Data Enabled
23	Reserved
24	USB Ch 4 USB SuperSpeed Data Enabled
25	Reserved
26	USB Ch 5 USB Super Speed Data Enabled
27	Reserved
28	USB Ch 6 USB SuperSpeed Data Enabled
29	Reserved
30	USB Ch 7 USB SuperSpeed Data Enabled
31	Reserved

Table 12: Hub Operational Mode Result Bitwise Description

USB Port State

Each downstream port reports information regarding its operating state represented in bit-packed results from:

```
stem.usb.getPortState(channel,state)
```

where channel can be [0-7], and the value status is 32-bit word, defined as the following:

Bit	Port State: Result Bitwise Description
0	USB V_{bus} Enabled
1	USB2 Data Enabled
2	Reserved
3	USB3 Data Enabled
4:10	Reserved
11	USB2 Device Attached
12	USB3 Device Attached
13:18	Reserved
19	USB Error Flag
20	USB2 Boost Enabled
23	Device Attached
24:31	Reserved

Table 13: Port State: Result Bitwise Description

USB Port Error Status Mapping

Error states for all downstream ports are bit-packed in 32-bit words available from:

```
stem.usb.getPortError(channel)
```

where channel is [0-7].

Errors can be cleared on each individual channel by calling the following method:

```
stem.usb.clearPortErrorStatus(channel)
```

Calling this command clears the port-related error bit flags (see Table 7) in the port error state. Global bits for hub errors cannot be cleared by this command.

Details about the port error status 32-bit word are as follows:

Bit	Port Error Status (channel) Result Bitwise Description
0	USB port current limit exceeded
1	USB port back-drive condition detected
2	Hub external power not present
3	Hub overtemperature condition
4:31	Reserved

Table 14: Port Error Status Result Bitwise Description

USB System Temperature

The temperature of the USB subsystem in the USBHub3+ can be measured with:

```
stem.temperature(0).getTemperature(µC)
```

where temperature is in micro-degress Celcius.

USBHub3+ Supported Entity Methods Summary

Detailed entity class descriptions can be found in the BrainStem Reference (https://acroname.com/reference/entities/index.html). A summary of USBHub3+ class options are shown below. Note that when using Entity classes with a single index (aka, 0), the index parameter can be dropped. For example:

```
stem.system[0].setLED(1) → stem.system.setLED(1)
```

Entity Class	Entity Option	Variable(s) Notes
store[0-1]	getSlotState	
	loadSlot	
	unloadSlot	
	slotEnable	
	slotDisable	
	slotCapacity	
	slotSize	
system[0]	save	
	reset	
	setLED	
	getLED	
	setSleep	
	setBootSlot	
	getBootSlot	
	getInputVoltage	
	getInputCurrent	
	getVersion	
	setHBInterval	
	getHBInterval	
	getModule	
	getSerialNumber	
	getModel	
temperature[0]	getTemperature	
timer[0-8]	getExpiration	
	setExpiration	
	getMode	
	setMode	
usb[0]	setPortEnable	Channels 0-7
	setPortDisable	Channels0-7
	setDataEnable	Channels 0-7
	setDataDisable	Channels 0-7
	setHiSpeedDataEnable	Channels 0-7
	setHiSpeedDataDisable	Channels 0-7
	setSuperSpeedDataEnable	Channels 0-7
	setSuperSpeedDataDisable	Channels 0-7
	setPowerEnable	Channels 0-7
	setPowerDisable	Channels 0-7
	getPortVoltage	Channels 0-7
	getPortCurrent	Channels 0-7
	getPortCurrentLimit	Channels 0-7
	setPortCurrentLimit	Channels 0-7
	setPortMode	Channels 0-7
	getPortMode	Channels 0-7

Entity Class	Entity Option	Variable(s) Notes
	getDownstreamDataSpeed	Channels 0-7
	getHubMode	
	setHubMode	
	getPortState	Channels 0-7
	getPortError	
	getEnumerationDelay	
	setEnumerationDelay	
	clearPortErrorStatus	
	getUpstreamMode	
	setUpstreamMode	
	getUpstreamState	
	getUpstreamBoostMode	
	setUpstreamBoostMode	
	getDownstreamBoostMode	
	setDownstreamBoostMode	
Pointer[0-3]	getOffset	
	setOffset	
	getMode	
	setMode	
	getTransferStore	
	setTransferStore	
	initiateTransferToStore	
	initiateTransferFromStore	
	getChar	
	setChar	
	getShort	
	setShort	
	getInt	
	setInt	
App[0-3]	execute	

Table 15: Supported USBHub3+ BrainStem Entity API Methods[5]

[5] See BrainStem software API reference at https://acroname.com/reference/ for further details about all BrainStem API methods and information.

Connections and LED Indicators

The front and back of the USBHub3+ have a set of connectors and LEDs to indicate control information and connectivity status. The meaning and location of each connector and LED are described in the following tables and diagrams.

LED Name	Color	Description
Link Status LED	Yellow	On once a host device has enumerated the BrainStem controller
User LED	Blue	Can be manipulated through any of the available APIs
Heartbeat LED	Green	Indicates active BrainStem connection; pulses at a rate determined by the system heartbeat rate
Power/Watchdog LED	Red and flashing blue	Solid red indicates the system is powered. Flashing blue is indication the internal watchdog is running and the USBHub3+ firmware is healthy
Upstream Operational Speed LED	Yellow or green	Upstream enumeration speed to host: green for SuperSpeed; yellow for Hi-Speed or lower USB 2.0 speeds.
Upstream 0 LED	Green	Indicates an active connection on upstream port
Upstream 1 LED	Green	
Control Port LED	Yellow	
Downstream Operational Speed LED	Yellow or green	Downstream device enumeration speed: green for SuperSpeed; yellow for Hi-Speed or lower USB 2.0 speeds; off when no device is enumerated
Downstream Power LED	Red	LED is on when downstream V_{bus} is enabled

Table 16: LED indicators

Connection Name	Type	Description
Front Panel, 0-7	USB 3.0 Type A	Downstream-facing connections to USB devices Each port is controlled and monitored by BrainStem API
Down A	USB 3.0 Type A	Downstream-facing connection to USB devices This port is always on and is not controlled or monitored by BrainStem API
Control Port	USB 2.0 mini-B	Dedicated BrainStem control command access
Up0	USB 3.0 Type B	Upstream-facing connections for a USB host to access downstream devices Selected UpX provides BrainStem control command access if Control Port is not connected
Up1	USB 3.0 Type B	
Barrel Power Input	Black Barrel-type	External power input
Euro-Style Power Input	Green Phoenix-type	External power input

Table 17: USBHub3+ External Connections

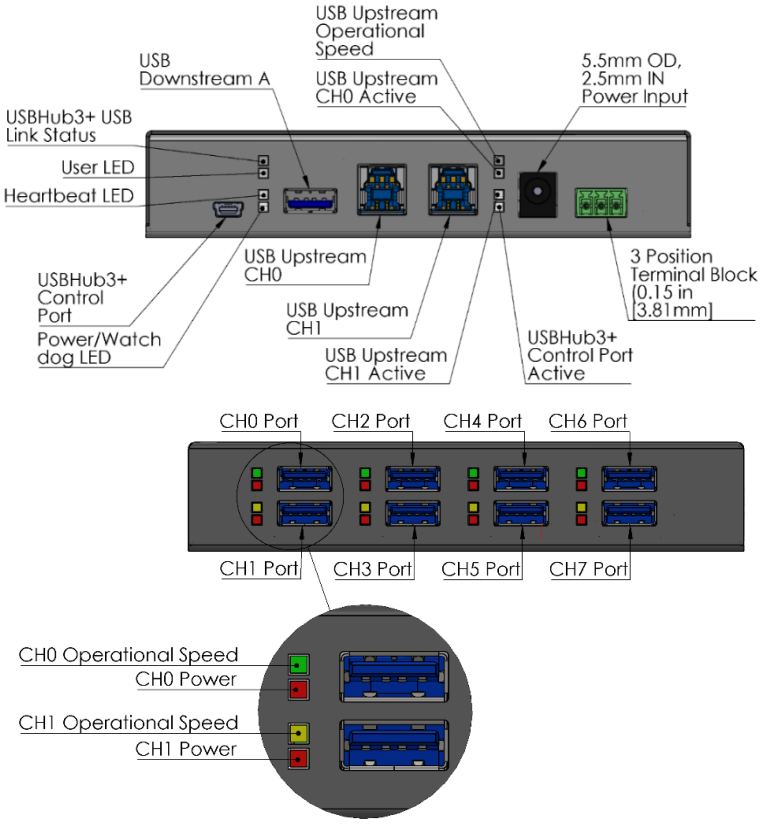


Figure 9: Connections and LED Indicators

Mechanical

Dimensions are shown in inches [mm]. 3D CAD models available from https://acroname.com.

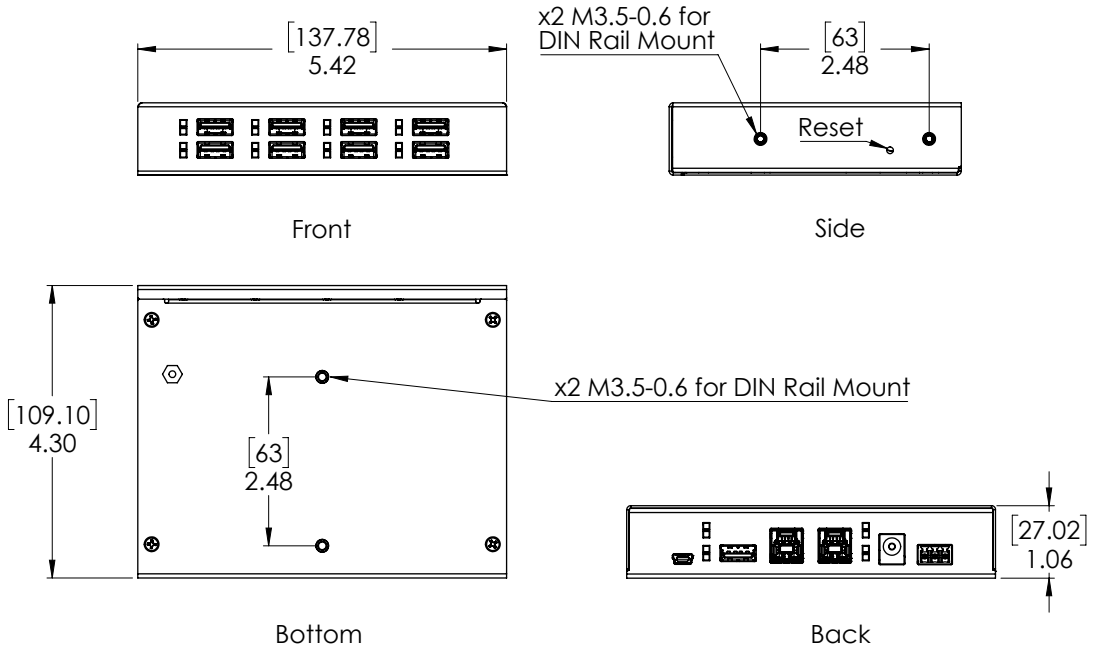


Figure 10: USBHub3+ Mechanical

DIN Rail Mounting

DIN rail mounting provisions have been designed into the USBHub3+ case. Holes for a DIN rail clip/adapter are provided to allow mounting of the USB3+ hub to standard DIN rails. Mounting clip hardware is available separately in a kit from Acroname: part number C31-DINM-1. The diagrams below illustrate USBHub3+ mounted in two orientations:

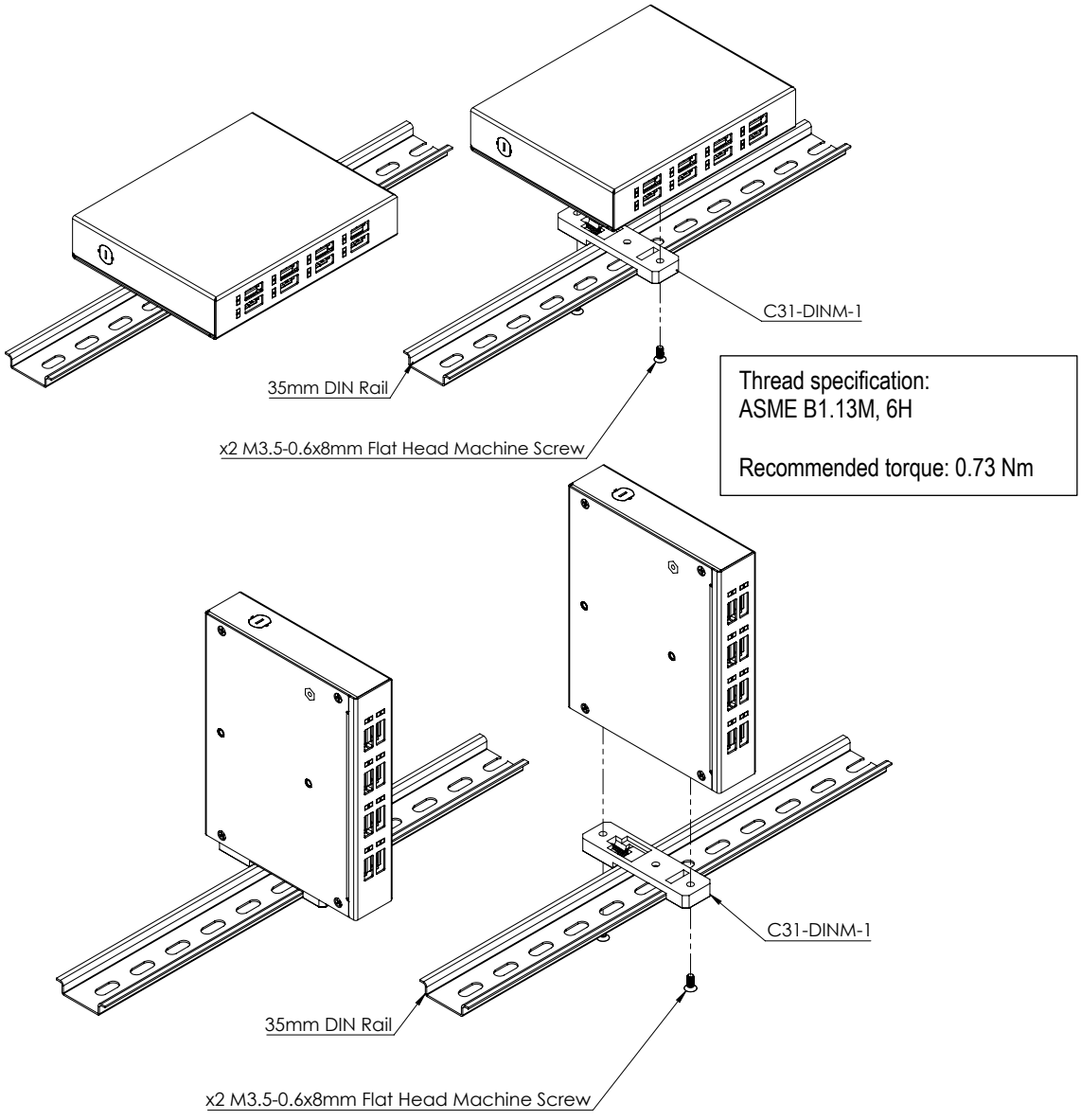


Figure 11: USBHub3+ DIN Rail Mount

Important Note

See instructions included in the C31-DINM-1 Kit. Acroname recommends using only the hardware provided in the C31-DINM-1 kit. Installing the DIN rail mount bracket without the included washers or with screws that are too long may damage the product and void any warranty.

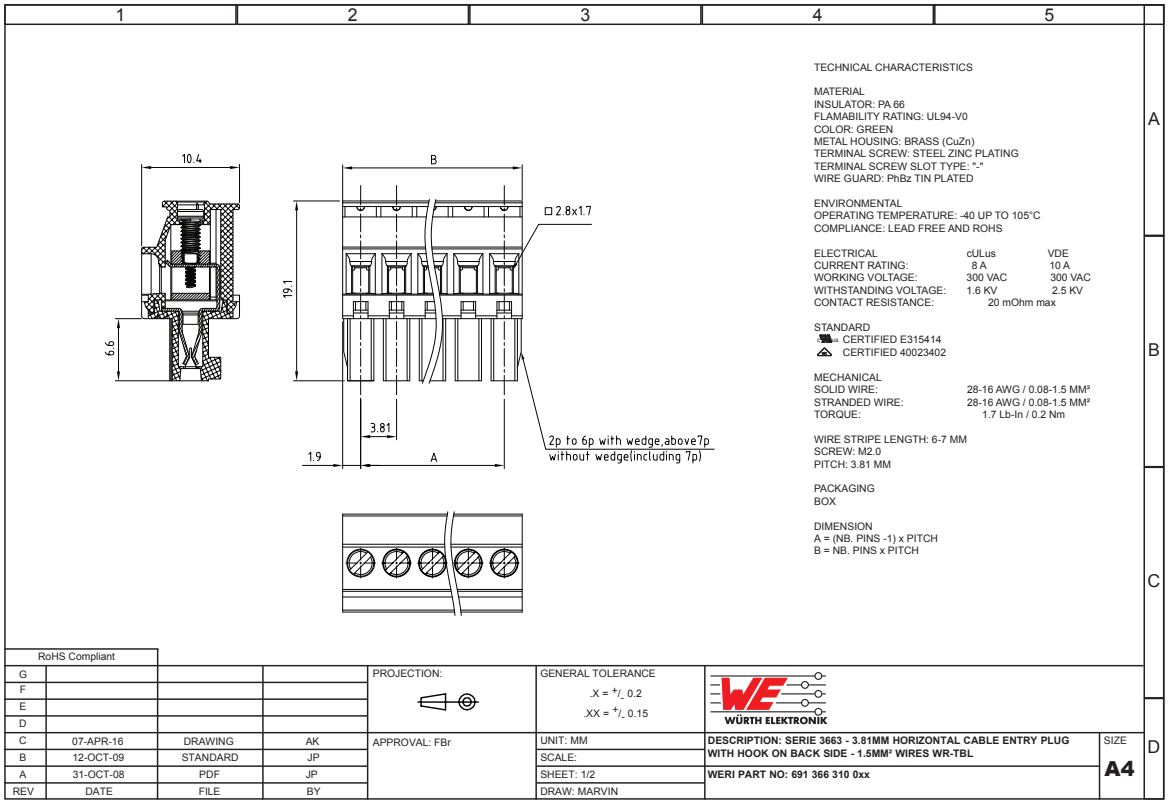
Input Power Connections

The USBHub3+ can be powered in three ways: bus powered from upstream port 0; the DC “barrel-jack”; a “Euro-style” terminal block.

When bus-powered from upstream port Up0, the system’s BrainStem controller is powered and available, however all downstream ports are automatically disabled and cannot be enabled. This power mode is useful for testing while developing code without the need for an external power adapter. To enable downstream ports, external power must be supplied.

The DC barrel-jack is a standard 6.5mm outside diameter, 2.5mm inside diameter, 9.5mm mating length connector. The mating plug is a 5.5mm outside diameter, 2.5mm inside diameter plug. Many manufacturers make compatible mating plug connectors; one example is the DC barrel plug from CUI: part number PP3-002B (https://www.cui.com/product/resource/pp3-002b.pdf).

In addition to the standard barrel jack power input an alternative power input, a Euro-style 3-pin terminal block (“Euroblock”), is also provided. This connector provides the additional benefits of a dedicated earth connection and a higher current rating (8A) than the barrel connector (5A). Many manufacturers make compatible mating connectors; one example is the Wurth Elektronik Serie 3663- 3.81mm Horizontal with Hook on Back Side WR-TBL, part number 691366310003 (see diagram below).



Source: http://katalog.we-online.com/en/em/TBL_3_81_3663_HORIZONTAL_W_HOOK_ON_BACK_SIDE_69136631000X

FCC Compliance Statement

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

```
This device complies with part 15 of
FCC Rules.  Operation is subject to the
following two conditions; (1) This
device may not cause harmful
interference, and (2) this device must
accept any interference received,
including interference that may cause
undesired operation.
```

Document Revision History

All major documentation changes will be marked with a dated revision code

Revision	Date	Engineer	Description
1.0	September 2016	JTD	Initial Release
1.1	September 2016	JLG	Update block diagram
1.2	October 2016	LCD	Update Overview, Features, Description Text
1.3	October 2016	LCD	Update ESD compliance info
1.4	January 2017	JLG	Add V_{bus} overvoltage information; add voltage and current measurement accuracy
1.5	February 2017	JLG	Add port state to saved parameters
1.6	March, 2017	JTD	Updated DIN mount screw spec
1.7	October, 2017	JLG	Update overvoltage and reverse current spec
1.8	April 2018	RMN	Removed Hub State for Port State
1.9	September 2018	LCD	Updated control path diagram
1.9.1	October 2018	LCD	Updates to support online BrainStem API documentation
1.9.2	October 2018	LCD	Added documentation on usb.getUpstreamState(); other minor corrections
1.9.3	November 2018	LCD	Improved Input Power Connections section; minor corrections
1.10	March 2019	JLG	Added spec for DIN mount hole PEM nut and torque Correct max current specification Add details on current limit behavior
1.11	February 2020	JLG	Added measurement subsystem section Corrected t_{trip} specification
1.12	July 2020	ACRO	Formatting update; clarify connection information; add humidity spec