ACRONAME

Programmable USB Hub Datasheet

S77-USBHUB-2X4

Overview

The USBHub2x4 is a 4-port software-programmable USB 2.0 (480Mbps) hub that is designed for demanding industrial environments where advanced control and monitoring of USB ports is required. This is very useful in testing or development environments where standard "always-on" behavior of a consumer-grade USB hub is not desirable.

Software control of the USBHub2x4 is established and maintained over the selected one of the two available host-facing ports.

The USBHub2x4 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
– Functional testing
– Camera control
– Battery charging
– USB device resets
– USB monitoring
– Sequential firmware load/updates

Features

– Individually enable/disable any of 4 downstream ports
– Data and power lines can be separately enabled for each downstream port
– Measure voltage and current on each downstream port
– Set programmable current limits for each downstream port (500mA to 2.5A)
– Automatic or programmed selection for either of 2 host port connections
– All ports support USB link speeds up to 480Mb/s
– Selectively enable USB charging mode behaviors:
  SDP (Standard Downstream Port) or
  CDP (Charging Downstream Port) modes[1]
– Deliver up to 2.5A per port (in CDP mode)
– Set enumeration delay for discovery of attached downstream devices
– Boost USB upstream and downstream data signal levels
– DIN-rail mountable
– Certified to withstand +/- 30kV ESD strikes (IEC6100-4-2 level 4)

Description

The USBHub2x4 gives engineers advanced flexibility and configurability over USB ports in testing and development applications.

Each downstream USB channel implements separately and independently switched data lines and 2500mA current-limited power lines. USB power and data can be independently disconnected for advanced USB testing applications. Pin interfaces are protected against reverse polarity and over-voltage, and connections are designed to operate from 0°C to 50°C ambient with no external cooling or fans.

Each USBHub2x4 is uniquely addressable and controllable from a host PC via the selected USB host input. Acroname's BrainStem™ link is then established over the USB input and allows a connection to the on-board controller in the USBHub2x4. USBHub2x4 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}	6.0	26.0	V
V_{supply} current	0.0	14.0	A
Voltage on V_{bus} inputs	0.0	24.0	V
Voltage on V_{bus} outputs	0.0	6.0	V
Voltage on any USB D+/D- inputs and outputs	-0.3	5.3	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
Storage Temperature, T_{STG}		-10.0	-	85.0	°C
Electrostatic Discharge, V_{ESD}	Exceeds IEC 61000-4-2, level 4, air and contact discharge	0.0	-	±30	kV

Table 2: Handling Ratings

Recommended Operating Ratings

Values presented apply to the full operating temperature range.

Parameter	Conditions/Notes	Minimum	Typical	Maximum	Unit
Input Voltage, V_{supply}		9.0	12	24.0	V
Input Current, I_{supply}		0.15	-	11.0	A
Voltage on V_{bus} inputs and outputs		4.5	5.0	5.5	V

Table 3: Recommended Operating Ratings

Block Diagram

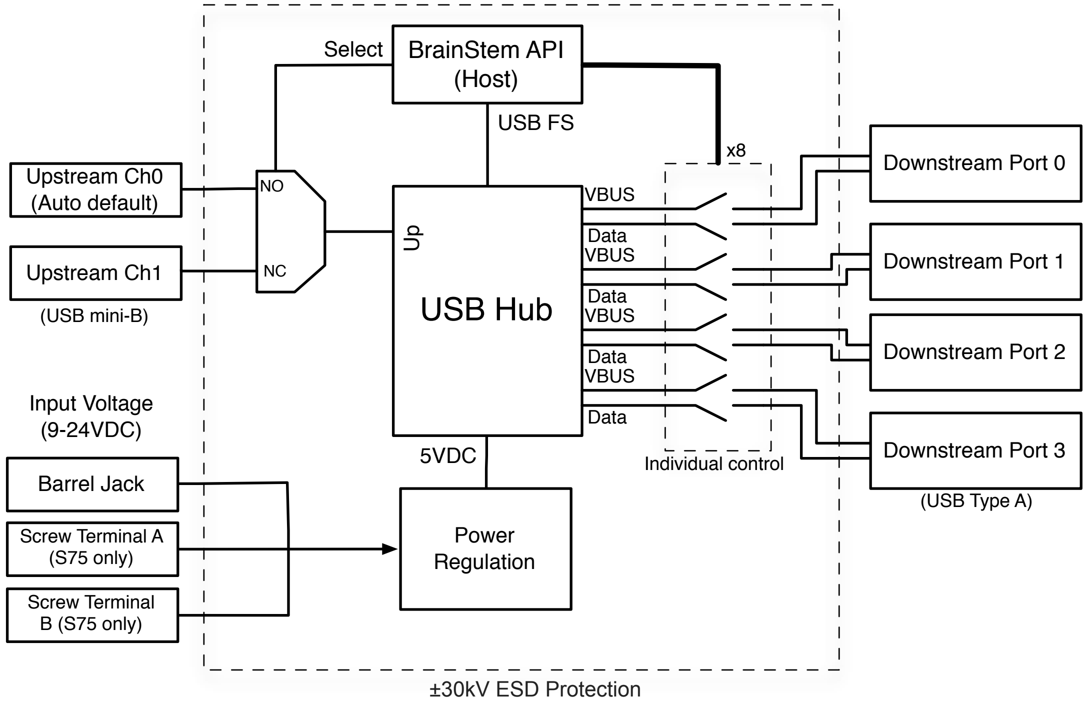


Figure 1: USBHub2x4 Block Diagram

Typical Performance Characteristics

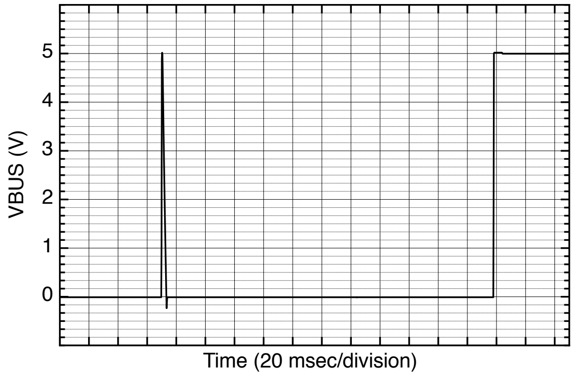
Values presented apply to the full operating temperature range.

Parameter	Conditions/Notes	Minimum	Typical	Maximum	Unit
USB Downstream (V_{bus})		4.5	5.0	5.5	V
USB Downstream Current	I_{LIM}=2.5A	0.0	-	2.5	A
System Efficiency	@12.0V input, nominal 6.5A load[2]	84	-	86	%
Current Measurement Range		6.4	-	2500	mA
Current Measurement Resolution		-	9.76	-	mA
Current Measurement Accuracy	I_{LIM} not exceeded		±2		%
V_{bus} Voltage Measurement		0.0	-	5.5	V
V_{bus} Voltage Resolution		-	1.2	-	mV
V_{bus} Output Rise Time	I_{LIM} = 1.0A	-	-	1.0	ms
V_{supply} Measurement Resolution		-	8	-	mV
Selectable Current Limits I_{LIM}	I_{LIM} = 500mA I_{LIM} = 900mA I_{LIM} = 1000mA I_{LIM} = 1200mA I_{LIM} = 1500mA I_{LIM} = 1800mA I_{LIM} = 2000mA I_{LIM} = 2500mA	- - - - - - - -	480 850 950 1130 1400 1720 1910 2370	500 900 1000 1200 1500 1800 2000 2500	mA
Short Circuit Response Time	Time from detection of short to current limit applied.	-	-	1.5	µs
Short Circuit Detection Time	Time from detection of short to port power switch disconnect.	-	-	6.0	ms
USB Downstream VBUS Current Supply (SDP mode)	USB 2.0 data lines disabled or no USB host present	-	-	100	mA
USB Downstream VBUS Current Supply (SDP mode)	USB 2.0 data lines enabled and USB host must be present	-	-	500	mA
USB Downstream VBUS Current Supply (CDP mode)	USB 2.0 data lines must be enabled	-	-	1500	mA

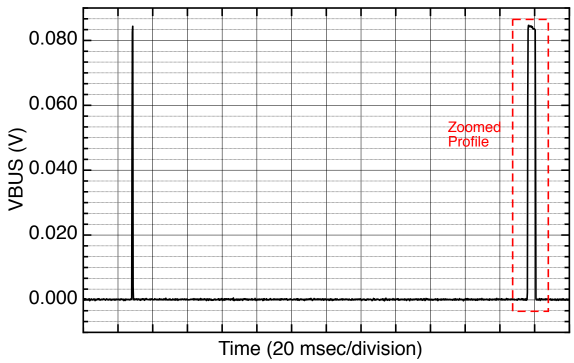
Table 4: Electrical Characteristics

[2] 6.5A selected as representative load based on 4 USB downstream devices running in CDP mode consuming approximately 1.5A each.

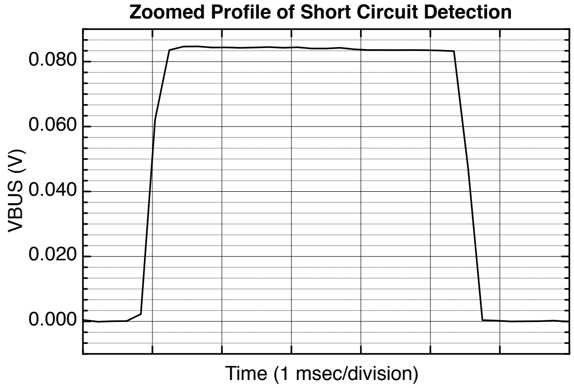
USB Downstream Enable (100mA load)



Short Circuit Detection



Zoomed Profile of Short Circuit Detection



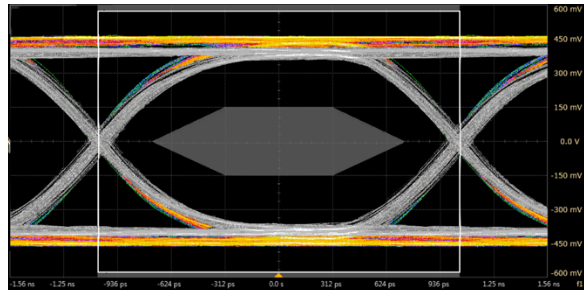


Figure 2: Upstream USB Eye diagram through USB Mini-B to host with 0.3m cable. Boost 0% in greyscale; Boost 12% in color.

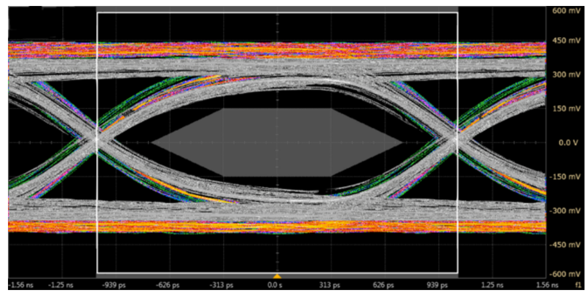


Figure 3: Upstream USB eye diagram USB Mini-B to host with 3.2m cable. Boost 0% in greyscale; Boost 12% in color.

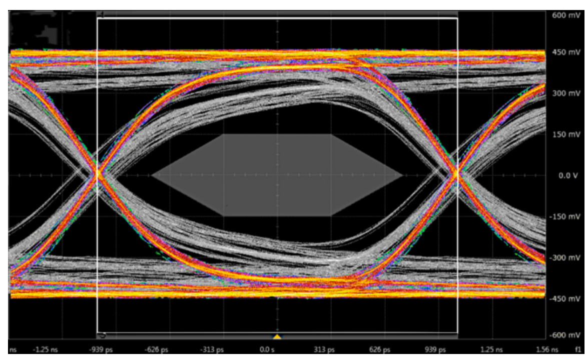


Figure 4: Downstream USB eye diagram USB A to device. 3.2m cable in greyscale; 0.3m cable in color.

Module Hardware and Software Default Values

The USBHub2X4 leverages a hardware-specific subset of BrainStem Entity implementations. The `aUSBHub2x4.h` C++ header file includes macro definitions for many parameters specific to the USBHub2x4. Table 5: USBHub2x4 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	aUSBHUB2X4_MODULE_ADDRESS	
Router Base Address	6		
Entity Class Definitions:			
`timer` Entity Quantity	8	aUSBHUB2X4_NUM_TIMERS	
`usb` Entity Quantity	1	aUSBHUB2X4_NUM_USB	
`store` Entity Quantity	2	aUSBHUB2X4_NUM_STORES	
`system` Entity Quantity	1		

Table 5: USBHub2x4 Hardware and Software Default Values[3]

[3] Refer to `aUSBHub2x4.h` within the BrainStem Development Kit download for actual file.

Device Drivers

USBHub2x4 leverages operating system user space interfaces that do not require custom drivers for operation on modern operating systems.

Outdated and unsupported operating systems may require the installation of a BrainStem USB driver. Installation details on installing USB drivers can be found within the BrainStem Development Kit under the “drivers” folder. Please note that Windows 7 will require looking at the supplied INF file found in the “drivers\brainstem_WinUSB” folder of the download.

Capabilities and Interfaces

The USBHub2X4 has a technological foundation built around BrainStem available to the user. Functionality details unique to the USBHub2X4 is described in the following sections; refer to Table 9: Supported USBHub2x4 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 like – 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.

Saving USB Entity Settings

Some USB entities can be configured and saved to non-volatile memory. This allows a user to modify the startup and operational behavior for the USBHub2x4 away from the factory default settings. Saving system settings preserves the settings to become the new default. Most changes to system settings require a save and reboot before taking effect; however, some, such as USB Port Enumeration Delay and USB Downstream Current Limit, can be changed with immediate effect. USB Boost mode settings (both upstream and downstream), for example, will not take effect unless a system save operation is completed, followed by a full power cycle. Use the following command to save changes to system settings before reboot:

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

Saved Configurations	
Software Offset	Reflex Boot Slot
Router Address	Current Limit (per port)
Heartbeat Rate	Port Mode (SDP, CDP) – each port
Upstream Boost	Port Enumeration Delay
Downstream Boost	

USB Entities

`usb` entities 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/disable USB data and VBUS lines, measure current, measure VBUS voltage, boost data line signals, and measure temperature.

Manipulating both data lines and VBUS lines for a single channel simultaneously can be done by calling the following method (channel 0, 1, 2, or 3):

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

Manipulating just the USB data line for a single channel can be done by calling the following method (channel 0, 1, 2, or 3):

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

Manipulating just the USB VBUS line for a single channel can be done by calling the following method (channel 0, 1, 2, or 3):

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

The USB VBUS voltage, as well as the current consumed on VBUS, can be read for each channel by calling the following methods (channel 0, 1, 2, or 3), where the second

---

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

variable passed into the method is the write location of the result:

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

Current-limit settings can be accessed for each channel by calling the following methods (channel 0, 1, 2, or 3), 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 USB power regulation stage (5.0V-regulated for VBUS) can be monitored for temperature stability, using the following method, where the variable passed into the method is the write location of the result:

```
stem.usb.getSystemTemperature(μC)
```

USB Downstream Operational Mode

The mode setting defaults to Standard Downstream Port (SDP) mode and a current-limit of 500 milliamps – the device can alternately be set to Charging Downstream Port (CDP) mode for devices that require high port charge current above 500 milliamps.

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

Available options for Downstream Operational Mode are:

- 0 – Standard Downstream Port (SDP)
- 1 – Charging Downstream Port (CDP)

USB Downstream Enumeration Delay

One a USB device is detected by the USBHub2x4 it is possible to create a host computer connection (enumeration) delay. The enumeration delay can mitigate or eliminate host kernel instabilities by forcing devices to enumerate in succession, allowing a focus on validation of drivers and software. The enumeration delay is in milliseconds, and applied to each downstream port.

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

USB Boost Mode

Boost mode increases the drive strength of the USB data signals (power signals are not changed). Boosting the data signal strength may help to overcome connectivity issues when using long cables or connecting through "pogo" pins. This setting is not applied until a system save call and power cycle of the hub. The system setting is then persistent until changed (followed by a save and reboot) or the hub is hard reset. After a hard reset, the default value of 0% boost is restored unless a new boost value has been applied and saved.

Boost mode can be applied to both the upstream and downstream USB ports.

```
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:

- 0 – no boost
- 1 – 4% boost
- 2 – 8% boost
- 3 – 12% boost

USB Hub Upstream Channels

The USBHub2x4 is perfect for environments where multiple devices need to be shared or switched between two host computers using two host (upstream) connections via USB Mini-B connectors. The upstream connection can be detected or specified using the following methods:

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

The *mode* parameter can be defined as the following (C++ macro name in parentheses):

- 2 (usbUpstreamModeAuto)
- 0 (usbUpstreamModePort0)
- 1 (usbUpstreamModePort1)

The default operational mode is to have the USB Upstream (to a host computer) be enabled through Channel 1 and auto detect which USB port to use. Automatic detection uses the absence/presence of a VBUS connection coming through the USB Mini-B connector on Channel 0.

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(state)
stem.usb.setHubMode(state)
```

The value *state* must be a 32-bit word, defined as the following:

Bit	Hub Operational Mode Result Bitwise Description
0	USB Channel 0 USB Hi Speed Data Enabled
1	USB Channel 0 USB VBUS Enabled
2	USB Channel 1 USB Hi Speed Data Enabled
3	USB Channel 1 USB VBUS Enabled
4	USB Channel 2 USB Hi Speed Data Enabled
5	USB Channel 2 USB VBUS Enabled
6	USB Channel 3 USB Hi Speed Data Enabled
7	USB Channel 3 USB VBUS Enabled
8:31	Reserved

Table 6: Hub Operational Mode Result Bitwise Description

USB Hub State

In addition to targeting individual downstream USB ports, overall hub state information can be represented in a bit packed result showing every port's state:

```
stem.usb.getHubState(bank=0,status)
```

The value *status* must be a 32-bit word, defined as the following:

Bit	Hub State Result Bitwise Description
0	USB Channel 0 device is attached
1	USB Channel 0 constant current
2	USB Channel 0 over temperature
3	USB Channel 0 USB error. See USB Hub Error Status Mapping
4:7	Reserved
8	USB Channel 1 device is attached
9	USB Channel 1 constant current
10	USB Channel 1 over temperature
11	USB Channel 1 USB error
12:15	Reserved
16	USB Channel 2 device is attached
17	USB Channel 2 constant current
18	USB Channel 2 over temperature
19	USB Channel 2 USB error
20:23	Reserved
24	USB Channel 3 device is attached
25	USB Channel 3 constant current
26	USB Channel 3 over temperature
27	USB Channel 3 USB error
28:31	Reserved

Table 7: Hub State Result Bitwise Description

USB Hub Error Status Mapping

It is possible to retrieve current error states for all downstream ports in a single 32-bit word. As only 4 downstream USB ports are available, the *bank* parameter should be set to 0 for the USBHub2x4.

```
stem.usb.getHubErrorStatus(bank=0,status)
```

Errors can be cleared on each individual channel (0, 1, 2 or 3) by calling the following method:

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

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

Bit	Hub Error Status Result Bitwise Description
0	USB CH0 overcurrent limit exceeded[5]
1	USB CH0 VBUS back drive[6]
2	USB CH0 hub power system failure
3	USB CH0 VBUS discharge[7]
4:7	Reserved
8	USB CH1 overcurrent limit exceeded[5]
9	USB CH1 VBUS back drive[6]
10	USB CH1 hub power system failure
11	USB CH1 VBUS discharge[7]
12:15	Reserved
16	USB CH2 overcurrent limit exceeded[5]
17	USB CH2 VBUS back drive[6]
18	USB CH2 hub power system failure
19	USB CH2 VBUS discharge[7]
20:23	Reserved
24	USB CH3 overcurrent limit exceeded
25	USB CH3 VBUS back drive[6]
26	USB CH3 hub power system failure
27	USB CH3 VBUS discharge[7]
28:31	Reserved

Table 8: Hub Error Status Result Bitwise Description

[5] Current limit value is defined by API settings section on USB Downstream Channels

[6] VBUS exceeds 5.150V for longer than 5ms

[7] VBUS discharge circuitry is activated. At the end of the 200ms the hub will confirm that VBUS was discharged if the VBUS voltage is not below 0.750V

USBHub2x4 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 USBHub2x4 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	
	getVersion	
	getModuleBaseAddress	
	getModuleSoftwareOffset	
	setModuleSoftwareOffset	
	setHBInterval	
	getHBInterval	
	getRouterAddressSetting	
	getModule	
	getSerialNumber	
	setRouter	
	getRouter	
	getModel	
timer[0-8]	getExpiration	
	setExpiration	
	getMode	
	setMode	

`usb[0]`	`setPortEnable`	channel can be 0, 1, 2, or 3
	`setPortDisable`	channel can be 0, 1, 2, or 3
	`setDataEnable`	channel can be 0, 1, 2, or 3
	`setDataDisable`	channel can be 0, 1, 2, or 3
	`setHiSpeedDataEnable`	channel can be 0, 1, 2, or 3
	`setHiSpeedDataDisable`	channel can be 0, 1, 2, or 3
	`setPowerEnable`	channel can be 0, 1, 2, or 3
	`setPowerDisable`	channel can be 0, 1, 2, or 3
	`getPortVoltage`	channel can be 0, 1, 2, or 3
	`getPortCurrent`	channel can be 0, 1, 2, or 3
	`getPortCurrentLimit`	channel can be 0, 1, 2, or 3
	`setPortCurrentLimit`	channel can be 0, 1, 2, or 3
	`setPortMode`	channel can be 0, 1, 2, or 3. mode can be 0 (SDP) or 1 (CDP)
	`getPortMode`	channel can be 0, 1, 2, or 3
	`getHubMode`	
	`getHubState`	bank is always 0
	`setHubState`	bank is always 0
	`getHubErrorStatus`	bank is always 0
	`getSystemTemperature`	
	`setEnumerationDelay`	
	`getEnumerationDelay`	
	`clearPortErrorStatus`	channel can be 0,1,2, or 3
	`getUpstreamMode`	
	`getUpstreamState`	
	`setUpstreamBoostMode`	
	`setDownstreamBoostMode`	
	`getUpstreamBoostMode`	
	`getDownstreamBoostMode`	

Table 9: Supported USBHub2x4 BrainStem Entity API Methods[8]

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

LED Indicators

Built into the board are a number of LED indicators to assist in system troubleshooting using the USBHub2X4.

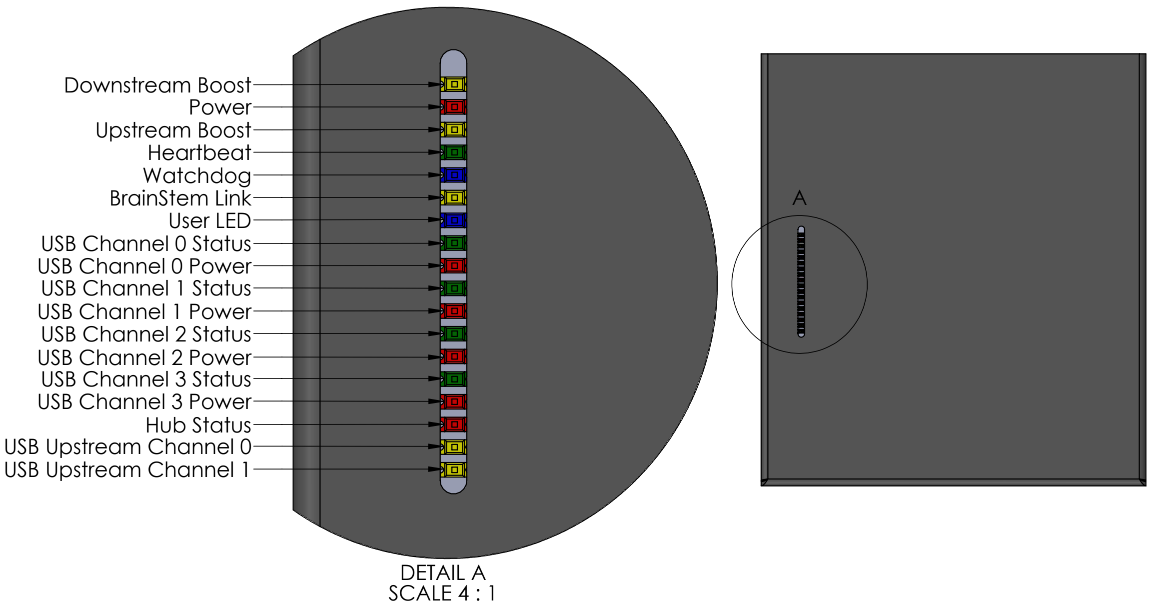


Figure 5: USBHub2x4 LED Indicators

The BrainStem Link LED will illuminate when the BrainStem USB interface is created on a host computer.

The Heartbeat indicator informs the user when communication is occurring with the BrainStem module, including a periodic heartbeat signal and response. Additional details on Heartbeats can be found in the BrainStem Terminology section of the Reference Manual.

The Logic Power indicator shows that a 3.3V voltage regulation system is up and running properly.

The User LED is a software controllable indicator accessed via the System BrainStem Entity. Detailed information can be found in the System Entity section of the reference manual.

Each downstream USB connection has corresponding LED indicators to show when power (VBUS) is enabled to the downstream device with the red LED labeled USB Channel X Power. Additionally, a green LED, labeled USB Channel X status, indicates whether the downstream device has enumerated on the host computer.

The Hub Status indicator illuminates when the USB hub communicates with a host computer.

Host Port Control Application Notes

The two upstream ports can be connected to two different host computers. Because of how USB architecture works, only one of the host computers can be selected to have access to the 4 downstream ports. Through the BrainStem APIs, you can change which of the two host computers has access to the downstream ports. Note that when a host upstream connection is changed, the software connection to the BrainStem module should follow appropriately.

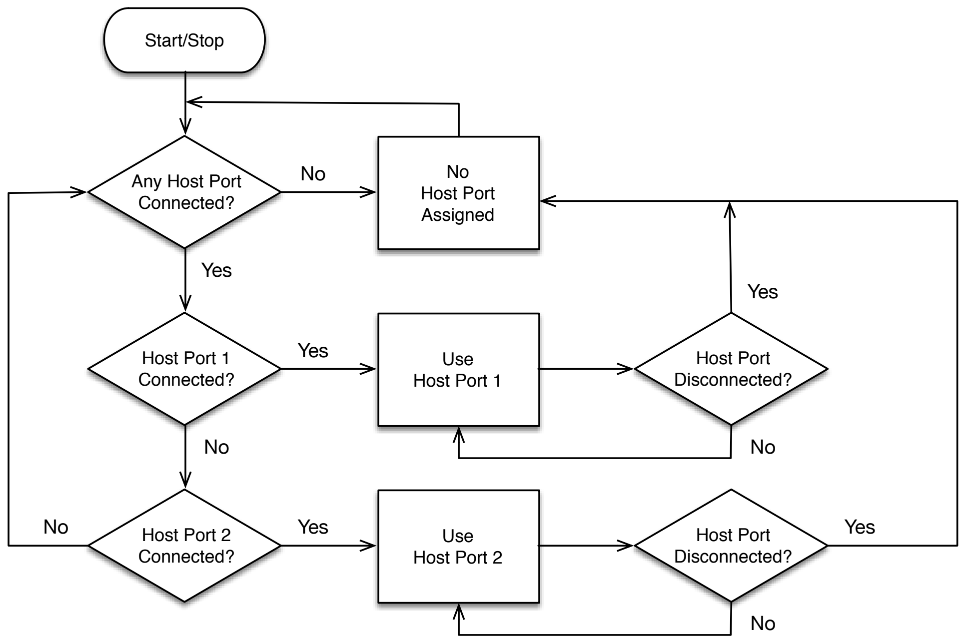


Figure 6: USBHub2X4 Host Port Control Flow

Mechanical

Dimensions are shown in inches [mm]. 3D CAD models are available through the USBHub2x4 product page's Downloads section

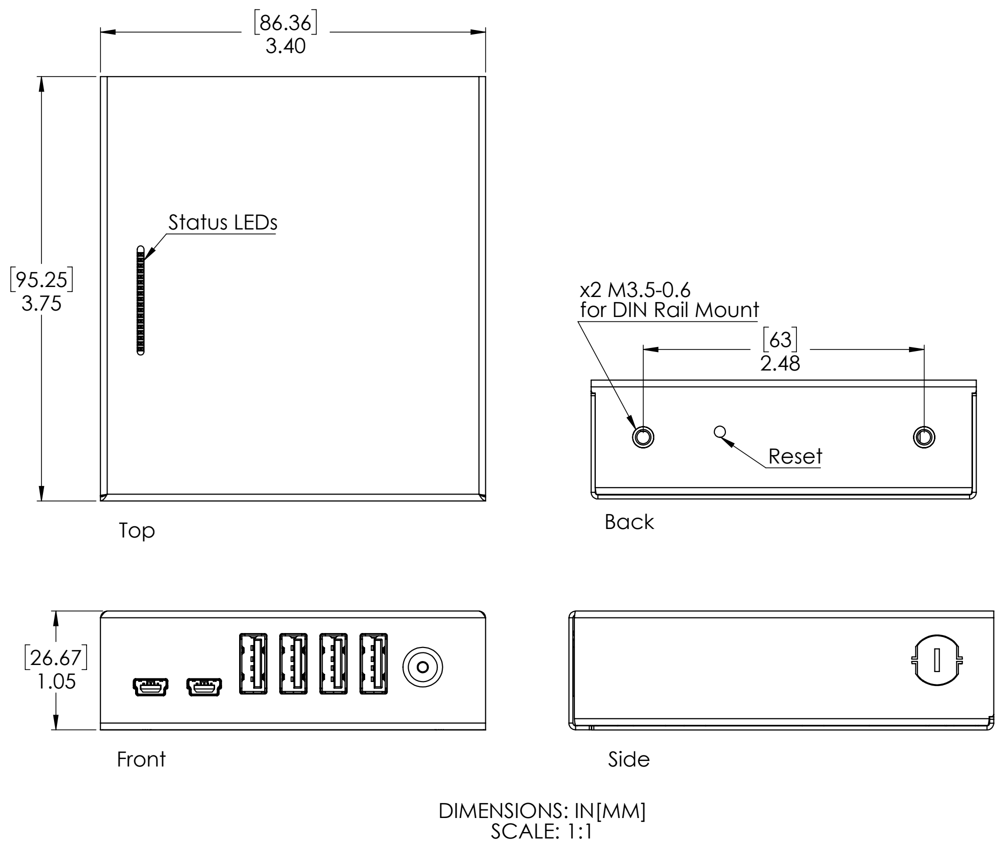


Figure 7: USBHub2x4 Mechanical

DIN Rail Mounting

DIN rail mounts have been designed into the USBHub2x4 case with an appropriate clip as often used for industrial control equipment. Mounting clip hardware is not included with the USBHub2x4. The mounting holes are compatible with many widely available "small" DIN rail mounting clips, and Acroname part number C31-DINM-1. The USBHub2x4 can be mounted in two positions:

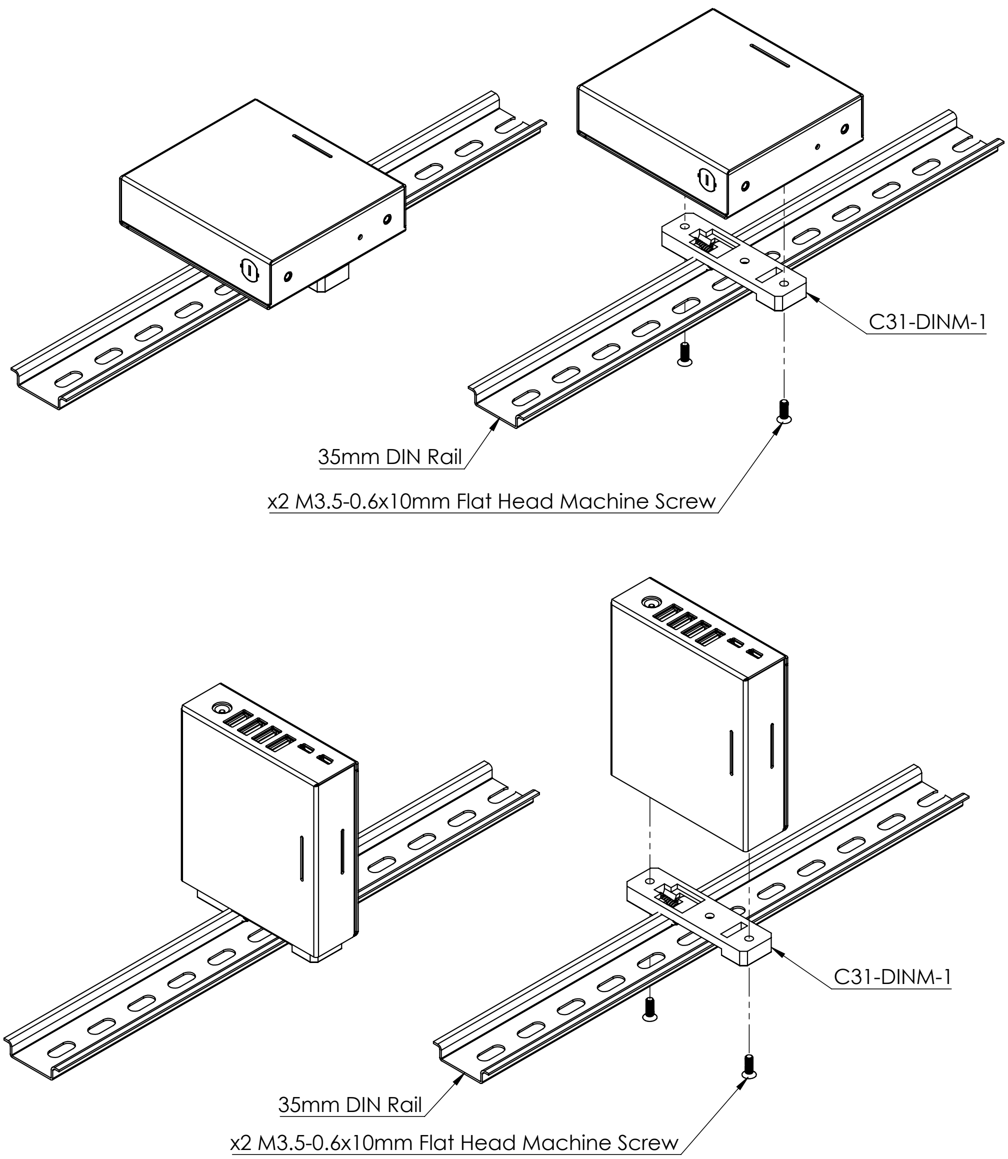


Figure 8: USBHub2X4 DIN Rail Mount

FCC Compliance Statement

Note: 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	April 15, 2015	MJK	Initial Revision
1.1	September 30, 2015	JTD	Reformatted. Added Entity Section Specifics, DIN rail mounting
1.2	November 30, 2015	JTD	Updated DIN rail mounting
1.3	December 29, 2015	JTD	Updated ESD rating
1.4	January 23, 2016	JLG	Typographical and formatting fixes
1.6	February 17, 2016	JLG	Update part number for DIN rail mount; FCC Compliance; add block diagram
1.7	February 25, 2016	JLG	Update Electrical Characteristics table
1.8	March 31, 2016	JTD	Updated CAD to v2
1.9	September 26, 2016	JTD	Updated formatting
1.91	October 11, 2016	LCD	Updated Overview, Features, Description sections