Overview

The USB-C-Switch is a 4:1 software-programmable USB Type-C multiplexer switch designed for demanding industrial environments where advanced control and monitoring of USB Type-C ports is required.

The USB-C-Switch can be used to selectively switch a USB connection between one of 4 available ports, conduct Type-C cable flip operations, measure current and voltages on V_{BUS} and V_{CONN} lines of all ports, and independently control USB data and power connections on each port.

Typical applications include:

- Manufacturing testing of USB Type-C ports
- USB device validation and development
- USB functional testing
- USB peripheral management
- USB Alt-mode testing
- USB PD profile testing
- Regression test environments
- Camera control
- Automating USB Type C port "flip"
- Automating USB plug/unplug operations
- Automation of Apple CarPlay® or AndroidAuto® testing

Features

- Selectively connect one USB Type-C connection to any one of 4 ports
- Bi-directional (supports 1:4 or 4:1 configurations)
- All ports support USB 3.1 Gen 2 link speeds up to 10Gbps
- All ports support USB PD profiles up to 100W (20V, 5A)
- Execute Type-C cable flip via software control[1]
- Supports pass through of USB Alt-Modes (DisplayPort, HDMI, ThunderBolt® and Digital Audio)
- HS Data, SS Data, CC/V_{CONN}, SBU, and power lines can be independently enabled/disabled for each port
- Measure V_{BUS} voltage on each port
- Measure V_{CONN} voltage and current
- Measure V_{BUS} current on selected port
- Configurable to be completely passive
- Configurable to boost USB HS data signals
- Dedicated control port for software control independent of the switched ports
- DIN-rail mountable
- Certified to withstand ±15kV ESD strikes (IEC61000-4-2 level 4)

Description

The USB-C-Switch gives engineers advanced control of USB connections in testing and development applications.

The USB-C-Switch hub architecture consists of several layers of internal switches to achieve the 4:1 switch and USB line control functionality. All type-C signals, including CC and SBU traffic, is passed through the USB-C-Switch. Data link and power negotiations between USB devices are provided by the attached devices themselves, allowing the USB-C-Switch to be used bi-directionally in 1:4 or 4:1 configurations.

Power and software control connections to the USB-C-Switch are established and maintained over the dedicated control port.

Each USB channel implements independently and separately switched HS/SS, V_{BUS}, CC and SBU lines. Pin interfaces are protected against reverse polarity and over-voltage. The device and all connections are designed to operate from 0°C to 50°C ambient with no external cooling or fans.

Each USB-C-Switch is uniquely addressable and controllable from a host PC via the selected USB host input. Built with Acroname's BrainStem® platform, the USB-C-Switch is easily controlled over USB with simple high-level APIs in C, C++, Python and LabView.

[1] Require use of one Acroname USB-C-TestCable

Block Diagram

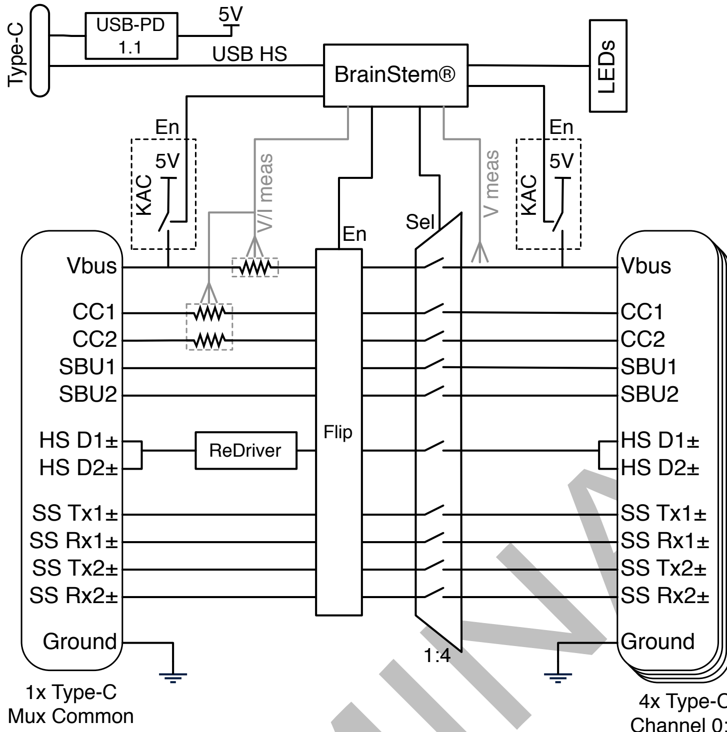

Figure 1: USB-C-Switch Functional Block Diagram

Application Diagrams

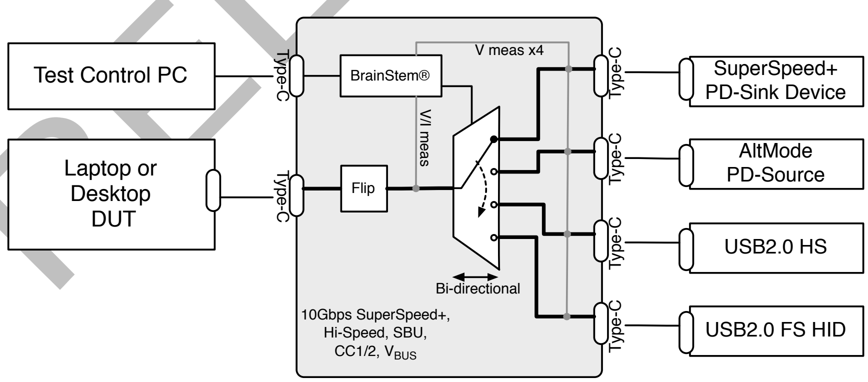

Figure 2: Typical testing application for validation against multiple types of devices.

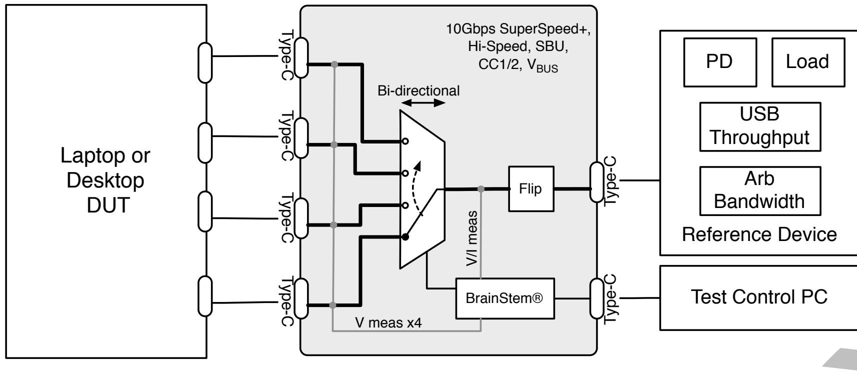

Figure 3: Typical testing application for validation against multiple types of devices.

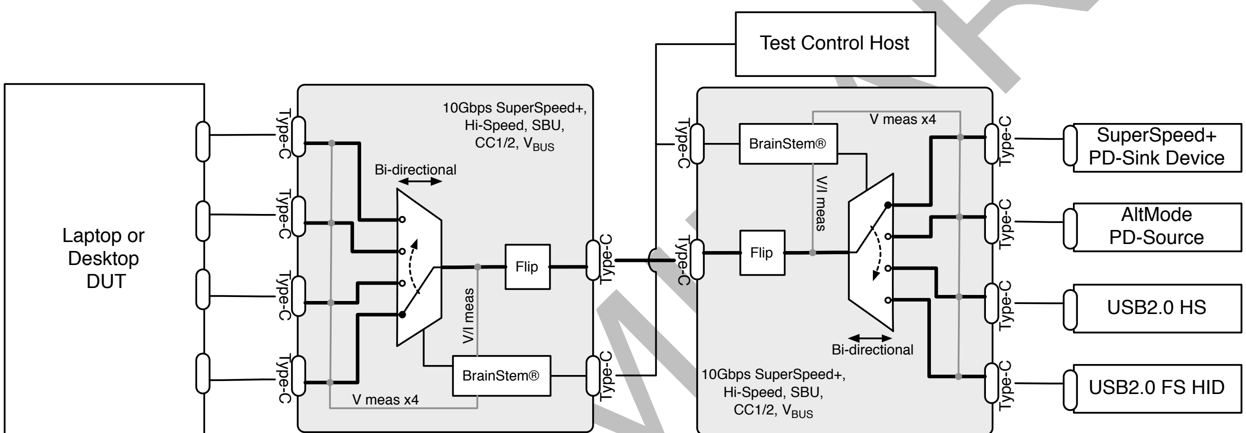

Figure 4: Typical testing application for validating multiple ports against multiple types of devices.

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 on V_{BUS} control port pin	-0.3	6.0	V
Voltage on any V_{BUS}, CC pin	-0.3	30.0	V
V_{BUS} current (bi-directional)	0.0	5.0	A
Voltage on any SuperSpeed+ (SS) data pin	-0.3	2.5	V
Voltage on any USB HiSpeed (HS) data and SBU pins	-0.3	4.5	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}	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

Values presented apply to the full operating temperature range.

Parameter	Conditions/Notes	Minimum	Typical	Maximum	Units
Input voltage on V_{BUS} control port pin		4.0	5.0	6.5	V
Voltage on any V_{BUS} pin		0.0	-	20.0	V
V_{BUS} current	Bi-directional	0.0	-	5.0	A
Voltage on SS data pin	Common mode	0.0	-	2	V
	Differential	0.0	-	1.8	Vpp
Voltage on any HS data pin		0.0	-	4.3	V
Voltage on any SBU pin		0.0	-	4.3	V
Voltage on any CC pin		0.0	-	5.0	V
Keep-alive charge (KAC) current			500	600	mA

Table 3: Recommended Operating Ratings

Typical Performance Characteristics

Values presented apply to the full operating temperature range.

Parameter	Conditions/Notes	Minimum	Typical	Maximum	Units
V_{BUS} common to mux port ON resistance		200	250	350	mΩ
V_{BUS} current measurement resolution		-	1.95	-	mA
V_{BUS} current measurement accuracy		-	±0.5	-	%
V_{BUS} voltage measurement resolution		-	8	-	mV
V_{BUS} voltage measurement accuracy		-	±0.2	-	%
CCx current measurement resolution		-	976	-	µA
CCx current measurement accuracy		-	±0.5	-	%
CCx voltage measurement resolution		-	4	-	mV
CCx voltage measurement accuracy		-	±0.5	-	%
Keep-alive charge (KAC) voltage	Sourced from control port V_{BUS}	4.5	5.0	5.5	V
Keep-alive charge (KAC) current limit	Constant current mode short circuit to ground	600	800	1000	mA
SS data differential insertion loss	f = 0.3 GHz preliminary	-	-1.6	-	dB
	f = 1.6 GHz preliminary	-	-3.2	-	dB
	f = 2.5 GHz preliminary	-	-4.2	-	dB
	f = 4 GHz preliminary	-	-5.9	-	dB
	f = 5 GHz preliminary	-	-7.1	-	dB
SS data differential retrun loss	f = 0.3 MHz preliminary	-	-25	-	dB
	f = 2.5 GHz preliminary	-	-13	-	dB
	f = 4.0 GHz preliminary	-	-13	-	dB
	f = 5.0 GHz preliminary	-	-12	-	dB
SS data differential OFF isolation	f = 0.3 MHz preliminary	-	-100	-	dB
	f = 2.5 GHz preliminary	-	-80	-	dB
	f = 4.0 GHz preliminary	-	-80	-	dB
	f = 5.0 GHz preliminary	-	-80	-	dB
SS data differental crosstalk	f = 0.3 MHz preliminary	-	-90	-	dB
	f = 2.5 GHz preliminary	-	-35	-	dB
	f = 4.0 GHz preliminary	-	-32	-	dB
	f = 5.0 GHz preliminary	-	-32	-	dB
SS data propagation delay	preliminary	-	3.0	-	ns
SS data intra-pair skew	preliminary	-	10	-	ps
SS data inter-pair skew	preliminary	-	30	-	ps

HS data ON resistance	preliminary	-	9.0	-	Ω
HS data ON resistance imbalance	preliminary	-	0.5	-	Ω
HS data ON resistance flatness	V=0.0-1.0, VI=30mA	-	1.5	-	Ω
HS data propagation delay	preliminary	-	0.6	-	Ns
HS data OFF isolation	preliminary	-	-48	-	dB
HS data crosstalk	preliminary	-	-30	-	dB
HS data 3dB bandwith	preliminary	-	1200	-	MHz
SuperSpeed+ data rate	Depends on host and devices and cable loss link budget	-	-	10	Gbps
HiSpeed data rate	Depends on host and devices; SS data disabled	-	-	480	Mbps

Table 4: Typical Performance Characteristics

Figure 5: Typical SS data differential insertion loss

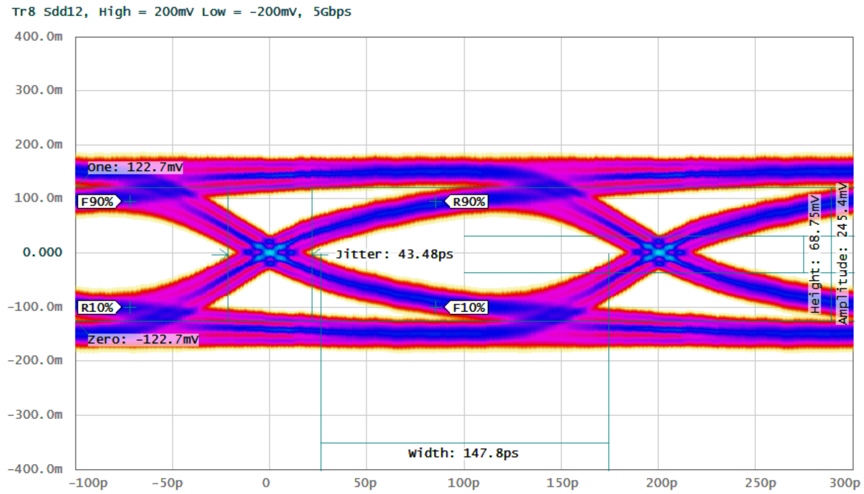

Figure 6: Simulated SS data 5Gbps eye diagram

Overview

The USB-C-Switch is designed as a platform to simplify switching of multiple USB type-C ports. The switch is a bi-directional four-to-one or one-to-four multiplexer (mux) for type-C connections. It is compatible with USB type-C applications including SuperSpeed+ (10Gbps) and all alternate modes (alt-mode). Supported alt-modes include Thunderbolt (40Gbps), HDMI, and DisplayPort.

At its core the switch is a passive analog mux for USB HiSpeed (HS), SuperSpeed+ (SS) and side band use (SBU) signals. V_{BUS} and CC signals go through current and voltage measurement functional blocks for use in testing and debugging of USB type-C systems. Further, CC lines have USB compliant cable orientation detection circuitry. This enables the USB-C-Switch to properly route signals when using two standards compliant USB cables. Further, when used with an Acroname Universal Orientation Cable (C38-USBC-UOC), the USB-C-Switch includes circuitry to create a cable flip which reverses the apparent cable orientation to connected devices.

There are two sides to the USB-C-Switch: common and mux. Via the BrainStem software API, the user can select which mux port is routed to the common port. Only one mux port can be active at a time. All signals are bi-directional, so either side can be connected to a host or device (see application examples above).

Cable Flip

A key feature of the USB type-C connector is it's orientation independence. This makes the type-C connector user friendly, and complicates the development of devices using the type-C standard. USB's type-C implementation makes the orientation determination a responsibility of the active devices in the system. The orientation is defined by the cable or downstream device in the system; more specifically, the orientation is defined by the type-C male or plug side of the connection.

A key feature of the USB-C-Switch is it ability to make the type-C orientation independence work even when there are multiple cables (specifically, more than two type-C male connectors). When used with standards compliant USB cables (e.g. full-featured type-C male-to-male with electronic markers ("e-marks") or with an "R_A" pull-down, or type-C male to type-A male, etc.), the USB-C-Switch detects the orientation of both the common and mux ports. The switch then internally routes the signals so that all signals are appropriately connected. That is, the USB-C-Switch acts like a normal cable for the active path between the common port and the active mux port. Connections using standards compliant cables will always "just work". Figure 7 shows example block diagrams of this feature when connecting a host through a full-featured, non-marked cable to a downstream device. All USB SS, HS and SBU lines are also routed appropriate, though omitted from the diagram for clarity.

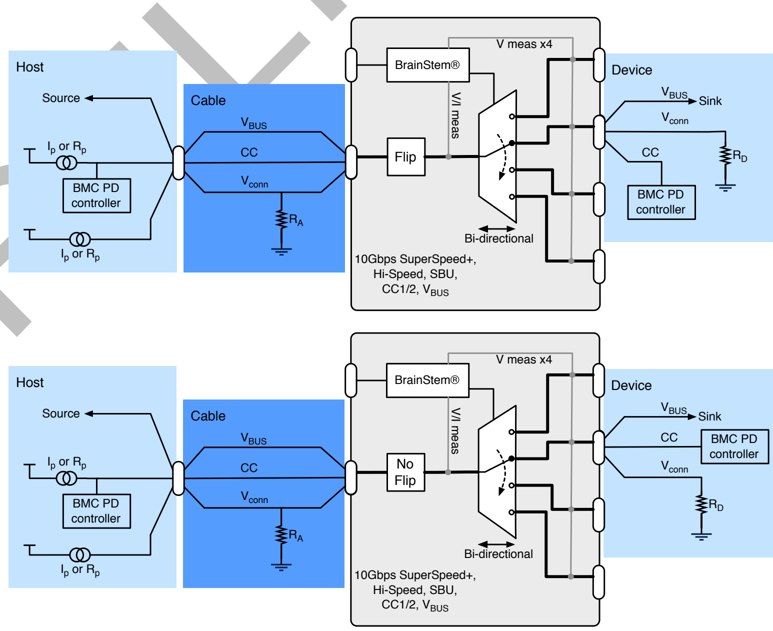

Figure 7: Flip and no-flip using for full-featured cable and device

When used with an Acroname C38-USBC-UOC cable, the USB-C-Switch enables the unique ability to affect a cable orientation "flip". When this orientation flip occurs, it will appear to connected devices that the orientation of their connection has reversed. It is incredibly import when testing any system with a USB type-C female connector to ensure that any internal muxes are functioning and connected to the female connector. Normally this is done by manually flipping a cable connection, which is time consuming, subjective and error prone. The USB-C-Switch allows this to be programmatically automated.

Keep-Alive Charging (KAC)

It is common to use battery powered devices on either side of the USB-C-Switch. When these devices are not in the active path, either on the common or mux side, the device battery may discharge. The USB-C-Switch has the unique feature of keep-alive charging (KAC) for the mux and common ports. When these ports are not active, enabling KAC on the port supplies 5V from the control port to the port's V_{BUS}. The switch does not provide any USB power-delivery (USB-PD), USB battery charge specification (BC1.2) or QuickCharge® to these inactive ports. Most mobile devices will support some level of charging from KAC. KAC is current limited. If a connected device pulls more than the allowed current, the USB-C-Switch will go into a constant current mode, dropping the voltage to maintain the current. KAC outputs are thermally protected and will disable the KAC output if needed.

Device Drivers

The USB-C-Switch leverages operating system user space interfaces that do not require custom drivers for operation on all modern operating systems including Windows, Linux and MacOS X.

Legacy operating systems like Windows 7 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.

Capabilities and Interfaces

The USB-C-Switch 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 USB-C-Switch are described in the following sections; refer to Table 7: Supported USB-C-Switch 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.[2]

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. Please see the Brainstem Reference materials on the website for a full description.

Serial Number

Every USB-C-Switch is assigned a unique serial number at the factory. This facilitates an arbitrary number of USB-C-Switch devices attached to a host computer. The following method call can retrieve the unique serial number for the currently connected device. The BrainStem C++ and python libraries both provide API calls for discovering attached BrainStem devices to facilitate connecting when multiple BrainStem devices are available.

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

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 USB-C-Switch 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. USB Boost settings, for example, will not take effect unless a system save operation is completed, followed by a reset or power cycle. Pressing the reset button will return all settings to factory defaults. Use the following command to save changes to system settings before reboot:

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

Saved Configurations	
USB Port Mode	

USB Entity

The `usb` entity provides a mechanism to control all USB functionality.

USB Channels

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.

The USBCSwitch has a single USB channel at channel 0. It uses the mux entity to switch between one of 4 active mux channels.

Manipulating Hi-Speed data, SuperSpeed data, and V_{BUS} lines simultaneously for the USB channel can be done by calling the following methods with channel 0 as the index:

```
stem.usb.setPortEnable(0)
stem.usb.setPortDisable(0)
```

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:

```
stem.usb.setDataEnable(0)
stem.usb.setDataDisable(0)
```

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:

```
stem.usb.setHiSpeedDataEnable(0)
stem.usb.setHiSpeedDataDisable(0)
```

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

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

```
stem.usb.setSuperSpeedDataEnable(0)
stem.usb.setSuperSpeedDataDisable(0)
```

Manipulating just the USB V_{BUS} line for a single port can be done by calling the following method with channel 0:

```
stem.usb.setPowerEnable(0)
stem.usb.setPowerDisable(0)
```

To enable or disable or get the current status of the automatic orientation detection and connection functionality, the following methods can be called on the USB channel.

```
stem.usb.setConnectMode(0, 0|1)
stem.usb.getConnectMode(0, mode)
```

Enabling or disabling or getting the current status of just the Type-C CC lines for the USB channel can be done by calling the following methods.

```
stem.usb.setCC1Enable(0, 0|1)
stem.usb.setCC2Enable(0, 0|1)
stem.usb.getCC1Enable(0, value)
stem.usb.getCC1Enable(0, value)
```

Enabling or Disabling or gettting the current status of the Type-C SBU lines for the USB channel can be done by calling the following methods.

```
stem.usb.setSBUEnable(0, 0|1)
Stem.usb.getSBUEnable(0, value)
```

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

```
stem.usb.getPortVoltage(0,μV)
stem.usb.getPortCurrent(0,μA)
```

The Type-C CC line current and voltage can be read for the USB channel by calling the following methods with channel 0, where the second variable passed into the method is the location for the measurement result.

```
stem.usb.getCC1Voltage(0, μV)
stem.usb.getCC2Voltage(0, μV)
stem.usb.getCC1Current(0, μA)
stem.usb.getCC2Current(0, μA)
```

USB Type-C Cable Flip

The USB-C-Switch can simulate a Type-C cable/connector flip by electrically switching the Side A and Side B CC/V_{CONN} and SBU lines, and swapping the USB data lines accordingly. This flip can be done, and checked, by calling the following methods:

```
stem.usb.getCableFlip(setting)
stem.usb.setCableFlip(setting)
```

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

- 0 – normal
- 1 – flipped

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 no 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 the USB ports with the follow methods:

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

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

- 0 – normal drive strength
- 1 – Boosted drive strength

BrainStem Control Port

The USB-C-Switch has a dedicated control channel. This is a full-speed USB 2.0 connection for BrainStem interface only. No USB switch traffic can flow on this connection.

USB Port Operational Mode

In addition to the individual port calls affecting specific functionality. The API includes a bitmapped method for setting specific port modes which allows for more granular control of the individual connections. The method takes a channel argument of 0 and a mode.

```
stem.usb.getPortMode(0, mode)
stem.usb.setPortMode(0, mode)
```

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

Bit	Hub Operational Mode Result Bitwise Description (1=Enabled, 0=Disabled)
0	Reserved
1	Reserved
2	Keep Alive Charging enable
3	Reserved
4	USB Hi-Speed Data Enable
5	USB V_{BUS} Enable
6	USB Super-Speed Lane 1 Data Enable
7	USB Super-Speed Lane 2 Data Enable
8:9	Reserved
10	Auto Connect Enable
11	CC1 Enable
12	CC2 Enable
13	SBU Enable
14	Reserved
15	CC Flip Enable
16	Super-Speed Flip Enable
17	SBU Flip Enable
18:31	Reserved

Table 5: Hub Operational Mode Result Bitwise Description

USB Port Operational State

Setting port modes and affecting the port mod via individual API calls can affect the USB channel state. The API includes a bitmapped method for getting the current port state. The method takes a channel argument of 0 and a mode.

```
stem.usb.getPortState(0, state)
```

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

Bit	Hub Operational State Result Bitwise Description (1=Enabled, 0=Disabled)
0	V_{BUS} Enable
1	Hi-Speed Enable
2	SBU Enable
3	Super-Speed Lane 1 Enable
4	Super-Speed Lane 2 Enable
5	CC1 Enable
6	CC2 Enable
7:8	Common side orientation status
9:10	Mux side orientation status
11:12	Speed status
13	CC Flip Enable
14	Super-Speed Flip Enable
15	SBU Flip Enable
16:18	Daughter-Card status
19	Error Flag
20	USB 2.0 Boost Enable
21	USB 3.0 Boost Enable
22	Connection Established

Table 6: Hub Operational State Result Bitwise Description

Mux Entity

The USB-C-Switch provides the ability to switch the connection of the common type-C connector to one of four type-C switched channels. This is done with the `mux` entity with the desired channel as the parameter:

```
stem.mux.setChannel(channel)
```

where channel is an index 0-3.

USB-C-Switch 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 USB-C-Switch 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	
	getInputVoltage	
	getVersion	
	getModuleBaseAddress	
	setHBInterval	
	getHBInterval	
	getModule	
	getSerialNumber	
	getRouter	
	getModel	
timer[0-8]	getExpiration	
	setExpiration	
	getMode	
	setMode	
usb[0]	setPortEnable	
	setPortDisable	
	setDataEnable	
	setDataDisable	
	setHiSpeedDataEnable	
	setHiSpeedDataDisable	
	setSuperSpeedDataEnable	
	setSuperSpeedDataDisable	

	setPowerEnable	
	setPowerDisable	
	getPortVoltage	
	getPortCurrent	
	getPortMode	
	setPortMode	
	getPortState	
	getDownstreamBoostMode	
	setDownstreamBoostMode	
	setCableFlip	
	getCableFlip	
	setConnectMode	
	getConnectMode	
	setCC1Enable	
	getCC1Enable	
	setCC2Enable	
	getCC2Enable	
	getCC1Voltage	
	getCC2Voltage	
	getCC1Current	
	getCC2Current	
	setSBUEnable	
	getSBUEnable	
mux[0]	setEnable	
	getEnable	
	setChannel	
	getChannel	
	getVoltage	Channels 0-3
		Channels 0-3
		Channels 0-3

Table 7: Supported USB-C-Switch BrainStem Entity API Methods[3]

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

Pinouts

The USB-C-Switch uses standard USB pin outs for the type-C female receptacles shown in Figure 8. The side-A and side-B USB HS D+ and D- are shorted in the USB-C-Switch, as recommended by the USB Implementers Forum.

A1 GND	A2 TX1+	A3 TX1-	A4 VBUS	A5 CC1	A6 D+	A7 D-	A8 SBU1	A9 VBUS	A10 RX2-	A11 RX2+	A12 GND
GND B12	RX1+ B11	RX1- B10	VBUS B9	SBU2 B8	D- B7	D+ B6	CC2 B5	VBUS B4	TX2- B3	TX2+ B2	GND B1

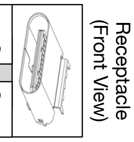

Figure 8: USB type-C receptacle pin out

LED Indicators

The control side of the USB-C-Switch has a set of indicators that show control information and connectivity status. The meaning and location of each LED are described in the following tables and diagrams.

LED Name	Color	Description
User	Blue	Can be manipulated through the available APIs
Power/ Heartbeat	Red/Yellow	Indicates system is powered, and indicates heartbeat status. Pulses at a rate determined by the system heartbeat rate to indicate an active BrainStem link.
Side A USB Status	Green	See Figure 10 for status indications.
Side B USB Status	Green	
Channel 0 Status	Blue	Indicates Mux channel selection.
Channel 1 Status	Blue	
Channel 2 Status	Blue	
Channel 3 Status	Blue	

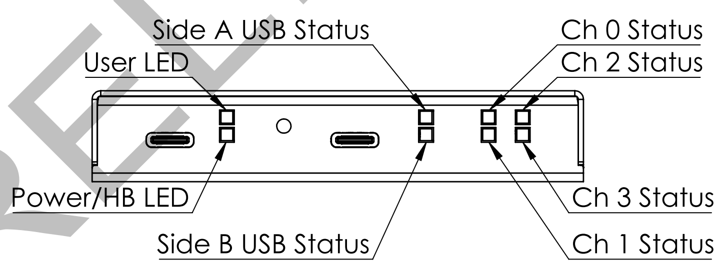

Figure 9: LED positions

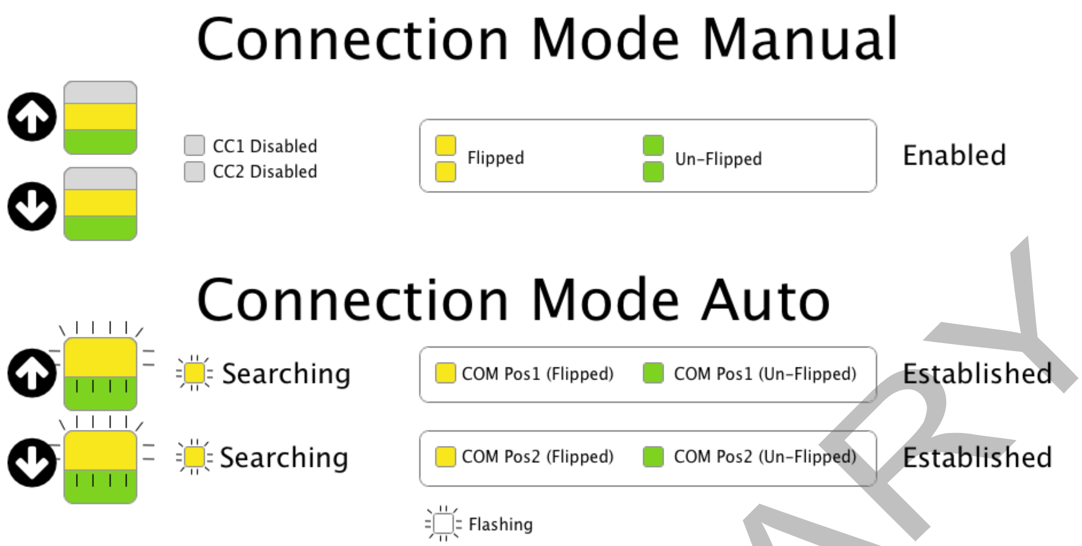

Figure 10: LED status

USB Connections

The control side of the USB-C-Switch has two USB Type-C connections – BrainStem control/power, and the single port side of the switch, referred to as Common.

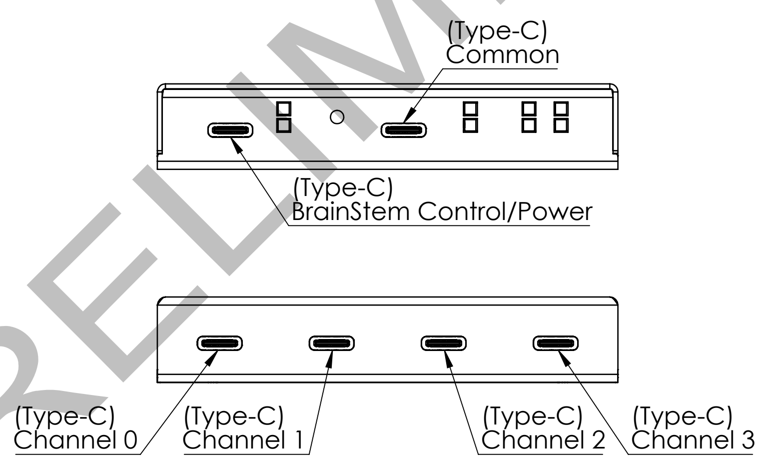

Figure 11: USB type-C connector names

Power Input

Power for the USB-C-Switch is provided by the V_{BUS} line on the BrainStem control port. This port supports USB Power Delivery 1.1 (USB-PD) high current mode of 5V at 3A. See Table 3: Recommended Operating Ratings for input voltage and power requirements.

Unit Reset

The USB-C-Switch can be reset to factory default settings using the reset button on the control side.

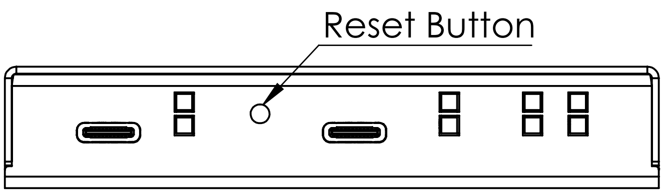

Figure 12: Reset button location

Mechanical

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

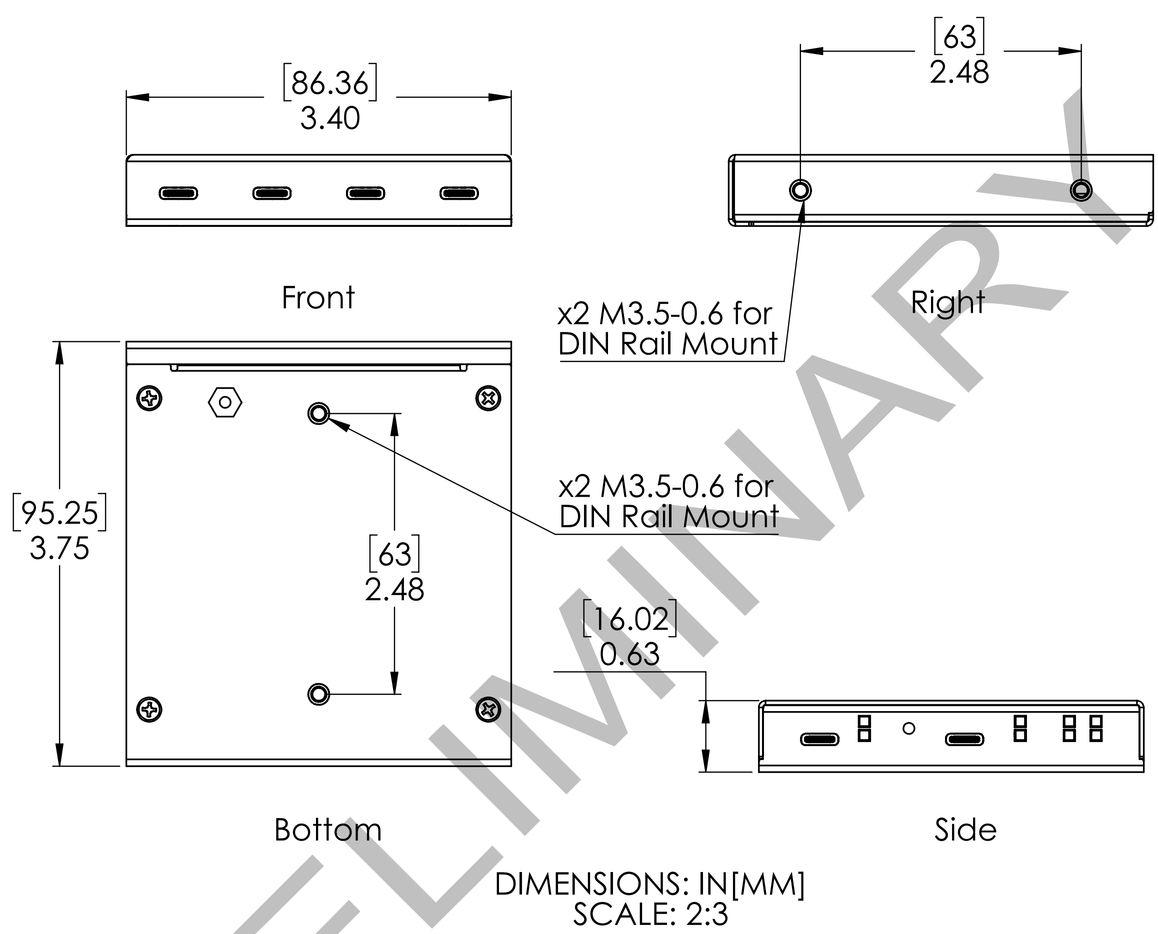

Figure 13: USB-C-Switch Mechanical Dimensions

DIN Rail Mounting

DIN rail mounts have been designed into the USB-C-Switch case with an appropriate clip as often used for industrial control equipment. Mounting clip hardware is not included with the USB-C-Switch. The mounting holes are compatible with many widely available “small” DIN rail mounting clips, and Acroname part number C31-DINM-1. The USB-C-Switch can be mounted in two positions as shown in figure Figure 14.

Warning: Care should be taken to only use M3.5 8mm long screws. Longer screws will cause irreparable damage the the USB-C-Switch.

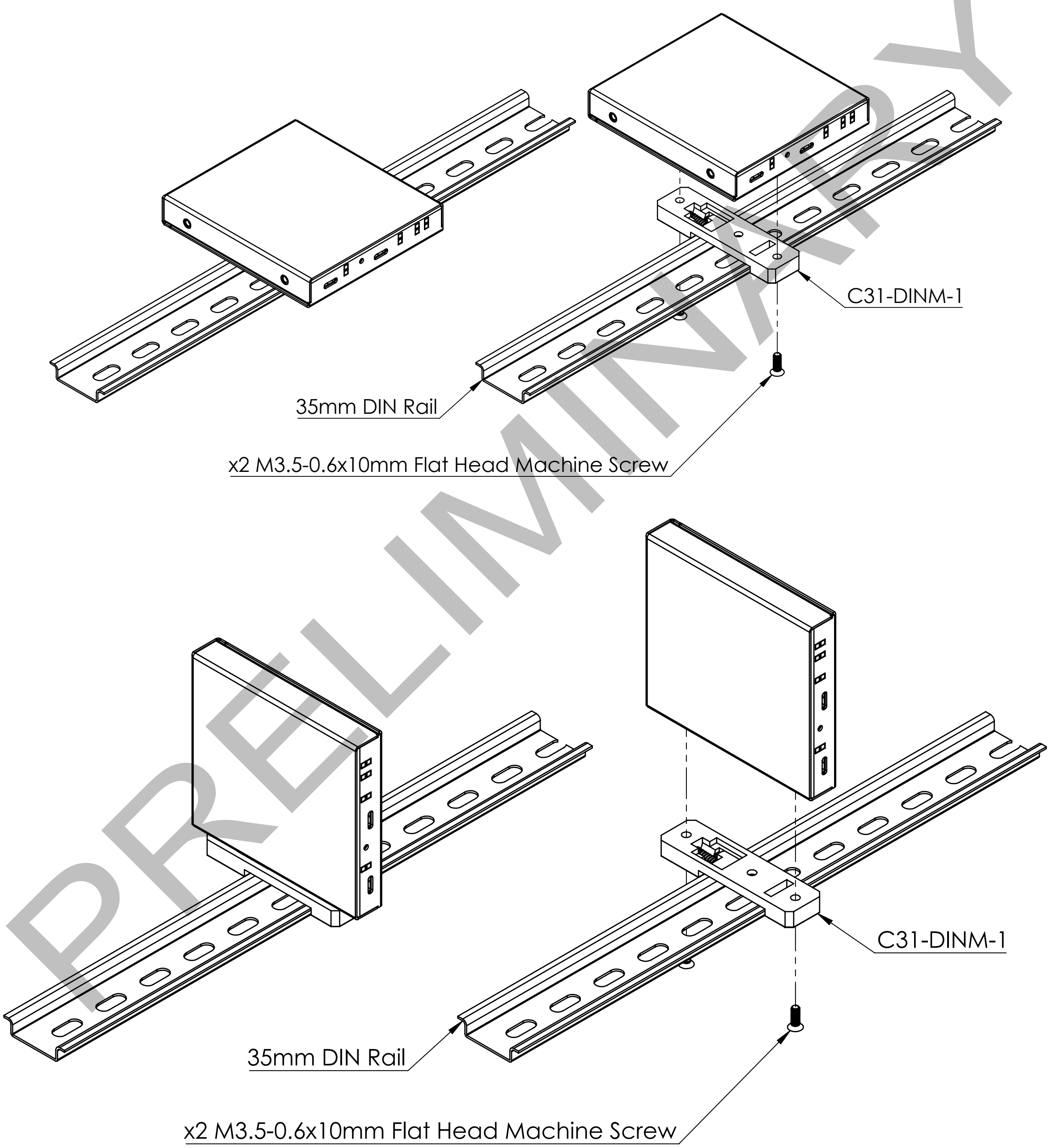

Figure 14: USB-C-Switch DIN Rail Mounting

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
0.1	January 2017	JTD	Pre-Release
0.2	July 2017	JLG	Preliminary release
0.3	September 2017	JRS	API updates to preliminary release