

# **IO Ninja Hardware Manual**

# Table of Contents

<b>Introduction</b>	<b>1</b>
<b>Serial Tap</b>	<b>1</b>
Block Diagram .....	4
Using the Serial Tap .....	4
Wedge RS232 monitoring .....	5
Specifications .....	7
<b>I2C/SPI Tap</b>	<b>8</b>
Using I2C/SPI Tap .....	9
I2C Mode Screenshot .....	11
SPI Mode Screenshots .....	12
Using Build-in Test Generator .....	14
Specifications .....	16
<b>Ethernet Tap</b>	<b>17</b>
Using Ethernet Tap .....	19
Ethernet Tap Plugin Screenshot .....	20
Specifications .....	20
<b>Update history</b>	<b>21</b>

# Introduction

Last update: 09DEC2019

[Manual Update History](#)

This Manual covers IO Ninja-compatible hardware:

- [Serial Tap](#)
- [I2C/SPI Tap](#)
- [Ethernet Tap](#)

Other types of hardware sniffers will be available in the near future.

## Serial Tap



Serial Tap is an affordable serial sniffer that can monitor RS232, RS485, and TTL-level UART communications. The Tap connects to your PC via a USB cable and is compatible with IO Ninja terminal/sniffer software.

### Features

- The Serial Tap is designed for use with IO Ninja software (<http://ioninja.com/>).
- Monitors TX, RX, RTS, CTS, DTR, and DSR lines <sup>[Note 1]</sup> of serial ports.
- All inputs are connected to an 18-position, quick-release terminal block.
- Has three operating modes: RS232, RS485 <sup>[Note 2]</sup>, and UART (TTL <sup>[Note 3]</sup>), selectable via a slide switch:
  - RS232 mode:
    - Two onboard DB9 connectors — male and female — offer true "cable wedge" sniffing <sup>[Note 4]</sup>;
    - Six jumpers for swapping and loopbacking signals within RX/TX, RTS/CTS, and DTR/DSR pairs <sup>[Note 5]</sup>;
    - Six bi-color LEDs indicate the status of monitored lines and allow distinguishing between the positive voltage (red), negative voltage (green), and zero voltage (off) states <sup>[Note 5]</sup>;
    - DB9 connectors additionally pass through DCD and RI signals (without monitoring them).
  - RS485 mode allows to monitor TX and RX signal pairs.
  - UART (TTL) mode allows to monitor UARTs of IC chips with logic levels from 3.3V to 5V.
- Full-speed (12Mbps) USB2.0 interface on a USB-C connector.
- Six yellow LEDs indicate the state of six monitored lines as they enter the USB controller. LEDs turn on when the lines go LOW <sup>[Note 6]</sup>.
- USB-powered, no additional external power necessary.
- Supplied with a USB-C cable and two DB9 gender changers.
- Compact, outside dimensions only 82 x 74 x 30 mm.

#### Note 1

A more accurate version of this statement would be: "Monitors two TX, two RTS, and two DTR lines," but that would be somewhat confusing to the reader. For clarity, we chose to list the lines as they are commonly known: TX, RX, RTS, CTS, DTR, and DSR. Keep in mind, however, that when it comes to sniffers, these names are relative and depend on the view. For two interconnected serial devices, a TX line on one end is an RX line on the other end. If so, which line is a "TX" and which line is an "RX" from the sniffer's point of view? Only you can answer this question. The Serial Tap itself has two inputs capable of monitoring serial data lines. Although they are both identical inputs, one is marked "TX" and another one is marked "RX." The same goes for other signals — RTS, CTS, DTR, and DSR. All these lines are inputs of the Tap. We gave them "regular" names to make the use of the Serial Tap intuitively easier. The same goes for RS485 lines. RX+/- and TX+/- pairs are both inputs of the Serial Tap.

This said, the Serial Tap incorporates two DB9 connectors, for which the distinction between TX and RX lines, RTS and CTS lines, and DTR and DSR lines is real. To give our terminology an anchored point of view, we have decided that the DB9-M connector on the left of the Tap is our "primary side" RS232 connector. For the entire board, all signal names are consistent with the standard pin assignment of this primary DB9-M connector. Of course, all "anchoring" disappears as soon as you abandon DB9s and plug into the terminal block lines.

#### Note 2

---

These inputs can also be used for monitoring RS422 communications.

Note 3

"TTL" here means "digital signals of microcontrollers, microprocessors, and other ICs." TTL inputs of the Serial Tap are also compatible with CMOS circuitry. Most importantly, "TTL" serial signals are usually inverted with respect to RS232 signals: a LOW (negative voltage) on an RS232 line corresponds to a HIGH level on a serial TTL (CMOS) line, and HIGH (positive voltage) on an RS232 line corresponds to a LOW level on a serial TTL (CMOS) line.

Note 4

TX, RX, RTS, CTS, DTR, and DSR lines of the primary DB9 connector located on the left of the Serial Tap are connected in parallel with RS232 inputs on the terminal block. This can be seen on the [Block Diagram](#) of the Serial Tap.

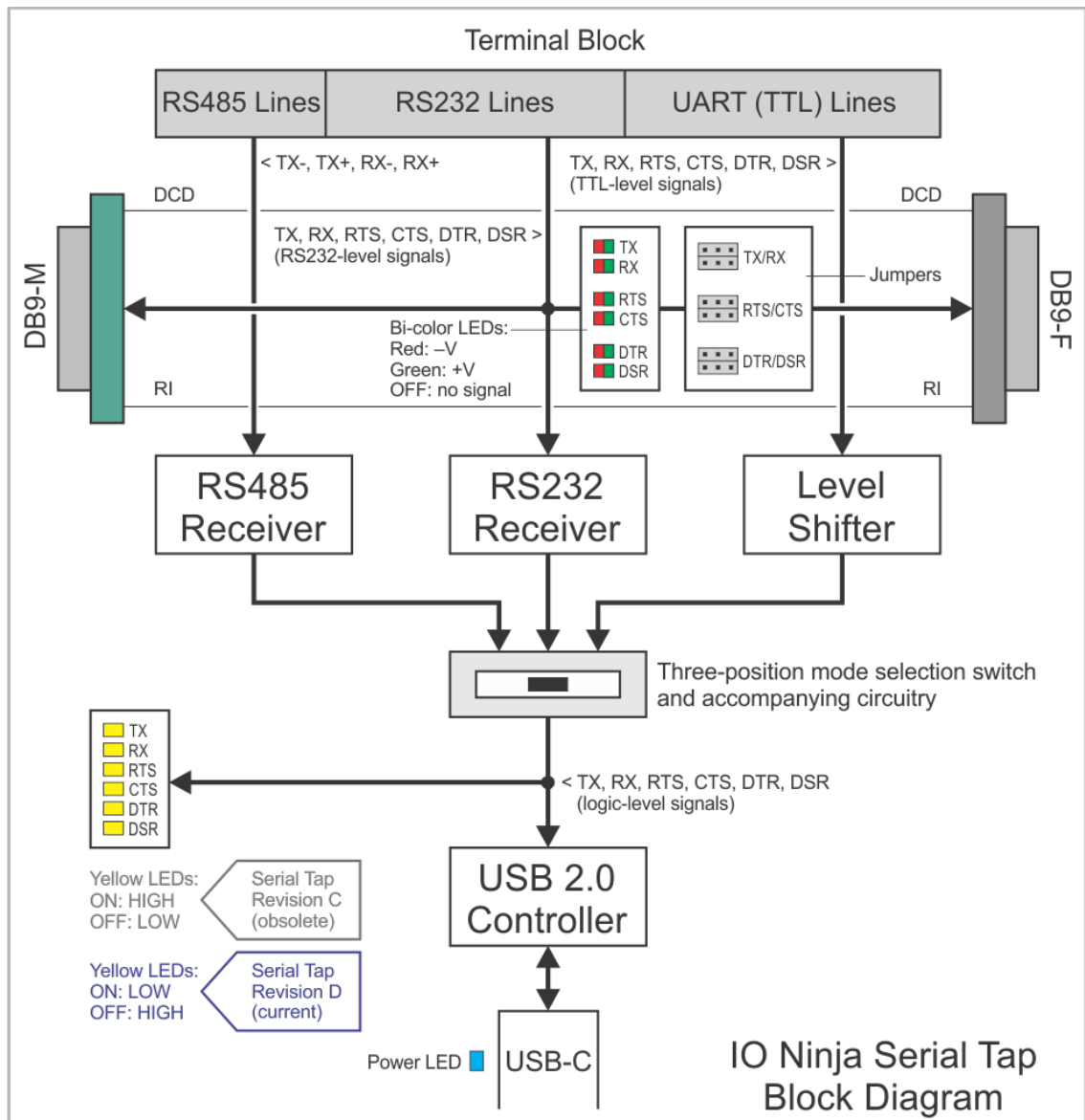
Note 5

Jumpers and bi-color LEDs on RS232 lines give the Serial Tap the second use as a simple mating and signal checking tool for serial devices.

Note 6

This statement pertains to the latest revision D of the product. On the earlier (and now obsolete) revision C, yellow LEDs used to turn on when the lines went HIGH.

## Block Diagram

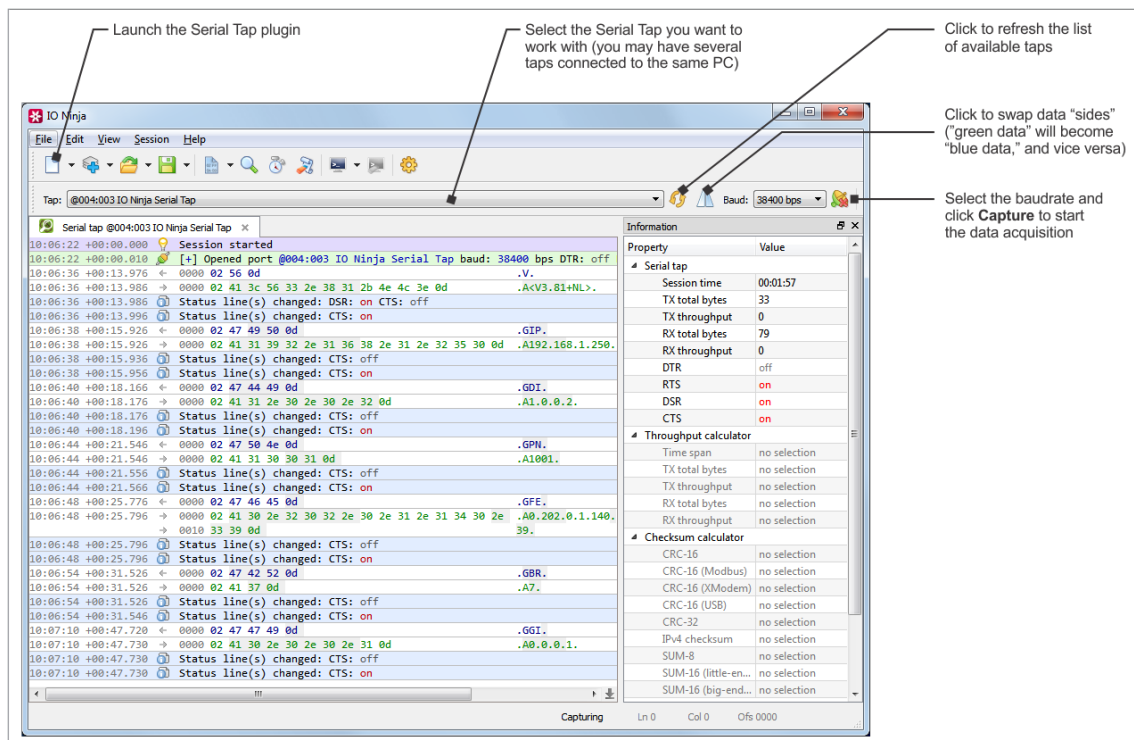


## Using the Serial Tap

To start using the Serial Tap:

- Install and run IO Ninja software (<http://ioninja.com/>).
- On the Serial Tap, slide the [mode selection switch](#) into the desired position depending on what interface you will be using — RS485, RS232, or UART (TTL).
- Connect into the corresponding group of terminals on the terminal block. When in the RS232 mode, you can also employ [wedge RS232 monitoring](#) method.
- Plug the Serial Tap into the USB port of your PC.
- Launch the **Serial Tap** plugin.

- Click on the **Tap** drop-down and select the Serial Tap. Note: you can connect several Serial Taps to one PC. If necessary, refresh the list of available Taps.
- Set the baudrate and click the **Listen** button to start the data acquisition.



### A note on the Serial Tap limitations

Please note that the Tap monitors serial traffic through two independent serial-over-USB channels. The first channel monitors TX, RTS, and DTR lines, while the second channel deals with RX, CTS, and DSR lines. Because these serial-over-USB channels are independent of each other, respective timing errors are always introduced on Windows and IO Ninja levels when receiving and recording the serial data and signal state changes.

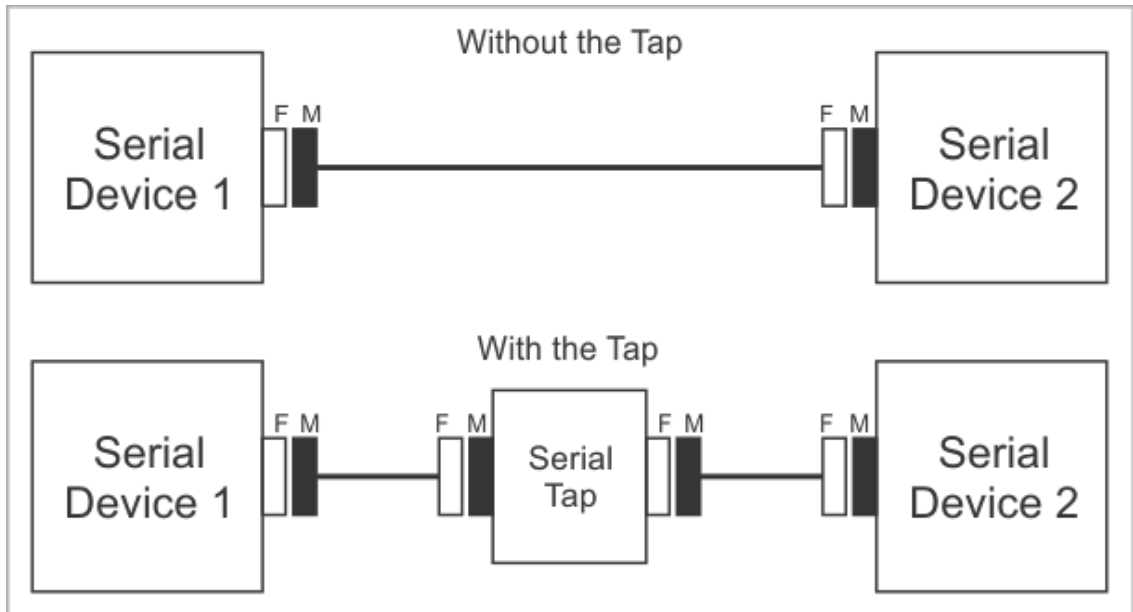
For example, let's suppose that you are monitoring serial communications between two interconnected serial devices and both devices have sent out some data at the same time. One of the USB channels will be luckier and get service first, while the other USB channel will experience a slight delay in service. IO Ninja, therefore, will show the data from one of the devices as having arrived first, and the data from the second device as having arrived second... while in reality, both devices have transmitted at the same time.

## Wedge RS232 monitoring

The Serial Tap allows you to insert (wedge) it in between two RS232 devices. Here is how this is done.

Let's suppose that two serial devices are interconnected by a serial cable. Let's also suppose that the first device has a DB9-F connector, while the second device has a DB9-M connector. The serial cable is, therefore, of the M-to-F type.

To wedge the Serial Tap between these two devices, you will need the second M-to-F cable:



Of course, DB9 genders on the serial devices may differ from the above example, and so your serial arrangement may be different. To aid you in the "wedging process," each Serial Tap comes with two DB9 gender changers.

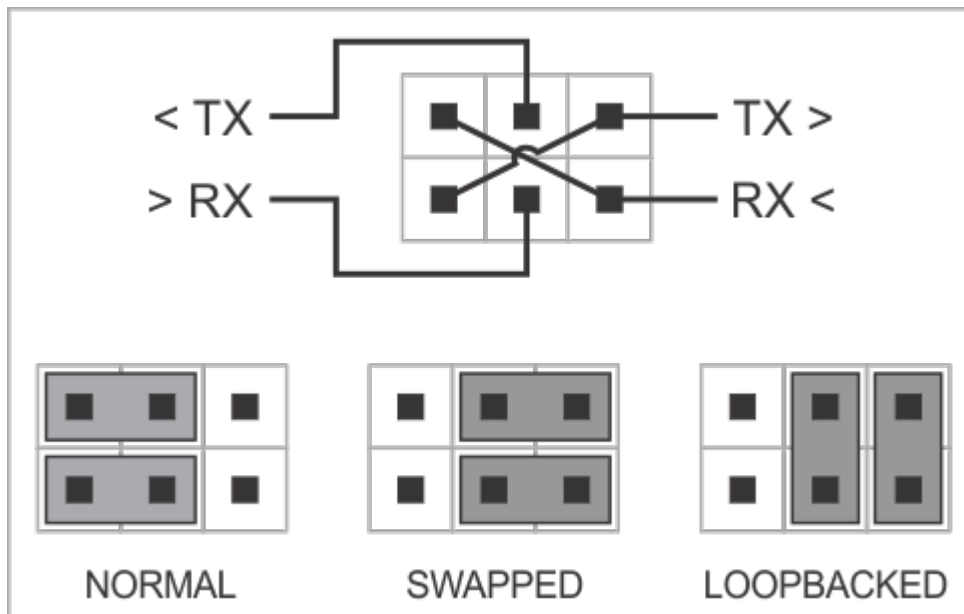
### Jumper pairs

The Serial Tap has an additional useful feature allowing you to swap and loopback the signals in TX/RX, RTS/CTS, and DTR/DSR signal pairs. To achieve this, two jumpers are provided for each of the three pairs. There are three standard jumper configurations:

- **Normal** — In this position, the lines are arranged in such a way that wedging the Tap between the serial devices does not change anything. Meaning, TX on one end goes to RX on another end, and vice versa.
- **Swapped** — This swaps signals in a pair. Meaning, TX goes to TX, and RX goes to RX.
- **Loopbacked** — Both serial devices "receive back" their own signals. Meaning, the TX line on each side "comes back" through the RX line.

The following diagram illustrates the jumper arrangements. The diagram shows the jumpers for the TX and RX signal pair. RTS/CTS and DTR/DSR jumpers work in the same way.





## Specifications

The Serial Tap is supplied with a USB cable and two DB9 gender changers.

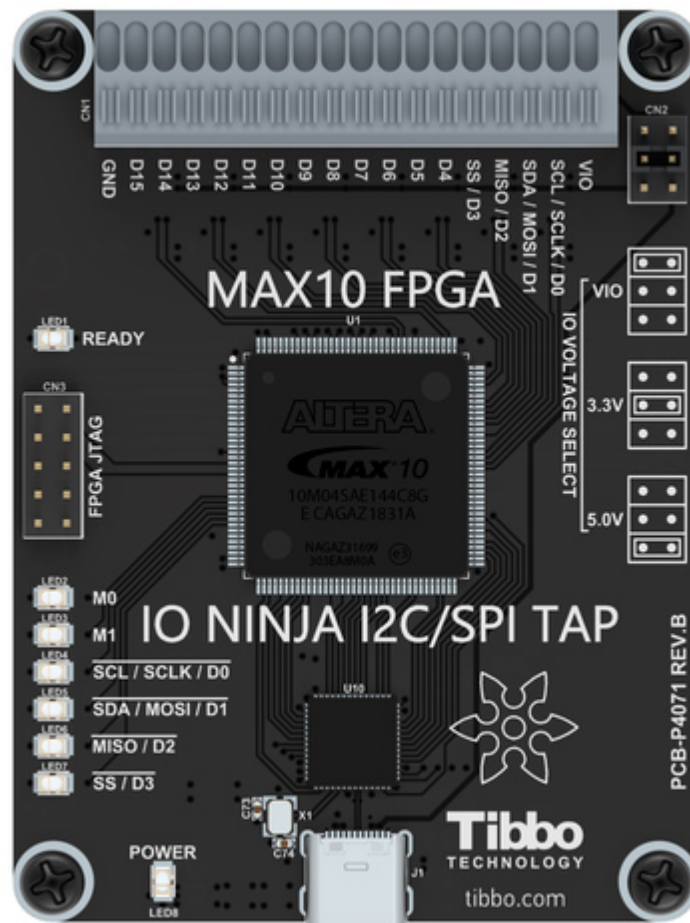
### Hardware specifications

USB port	USB 2.0, full-speed (12MHz), USB-C connector
Maximum baudrate	2,764,800 bps
Operating temperature	0 to +60 degrees C
Operating relative humidity	10-90%
Mechanical dimensions	82 x 74 x 30 mm

*All specifications are subject to change without notice and are for reference only. Tibbo assumes no responsibility for any errors in this Manual and does not make any commitment to update the information contained herein.*

*Is the data you are looking for missing from the above table? Do not just assume that you know the answer — talk to Tibbo! Remember, that the ultimate responsibility for all decisions you make regarding the use and the mode of use of Tibbo products lies with you, our Customer.*

## I2C/SPI Tap



I2C/SPI Tap is an affordable sniffer that can monitor I2C and SPI communications. The Tap connects to your PC via a USB cable and is compatible with IO Ninja terminal/sniffer software.

### Features

- Designed for use with IO Ninja software (<http://ioninja.com/>).
- Reliably captures I2C traffic at clock speeds of up to 700KHz and SPI traffic at clock speeds of up to 24MHz <sup>[Note 1]</sup>; all four SPI modes are supported.
- Incorporates an I2C and SPI test sequence generator for self-testing the Tap.
- Based on Intel MAX10 FPGA.
- Has 16 buffered IO lines D0~15 connected to a quick-release terminal block.
- All buffers act as level shifters allowing the Tap to work with logical signals in the 1.8V-5V range.
- Eight LEDs onboard:
  - Green Ready LED;

- Two general-purpose green LEDs M0 and M1 for mode indication;
- Four yellow LEDs for indicating the state of IO lines D0~3 (sufficient for monitoring I2C and SPI communications);
- Blue Power LED.
- Built-in [test generator](#) outputs fixed I2C and SPI data sequences.
- High-speed (480Mbps) USB2.0 interface on a USB-C connector.
- Supplied with a USB-C cable.
- USB-powered, no additional external power necessary.
- Compact, outside dimensions only 82 x 74 x 30 mm.
- Product functionality may be expanded in the future:
  - Onboard FPGA is reconfigurable via the USB interface;
  - Sixteen IO lines provide ample room to grow;
  - It is envisioned that the product's abilities may be expanded to handle such protocols as...
    - Quad-SPI;
    - SDIO;
    - Single-wire bus;
    - and more!

#### Note 1

This number depends on the host PC's performance and its "workload." Captured data may be lost if a PC's USB channel is unable to receive it in a timely manner. Our tests have shown that an average i7-based desktop PC (that is not performing other time-critical tasks) easily captures sustained SPI traffic with clock speeds up to 24MHz.

## Using I2C/SPI Tap

### Connecting I2C and SPI signals to the Tap

To monitor I2C traffic, connect your I2C bus to terminals D0 (SCL) and D1 (SDA) terminals.

To monitor SPI traffic, connect your SPI bus to terminals D0 (SCLK), D1 (MOSI), D2 (MISO), and D3 (SS).

Note that the Tap has a built-in [generator of test signals](#).

### Selecting the logic voltage levels

I2C/SPI Tap can work with logic voltage levels from 1.8V to 5V. For correct operation of the Tap, you must select correct reference voltage for the Tap's level shifters.

This is done using the IO VOLTAGE SELECT jumper. Two reference voltages — 3.3V and 5V — are provided by the Tap itself. Other logic levels require an external reference voltage applied to the VIO terminal. To select the VIO input, put the jumper into the VIO position and connect the VIO terminal to the external voltage source (typically, the VCC line of the device you are monitoring). For example, if you

are tapping into the I2C bus of a 1.8V device, then connect the VIO terminal to the 1.8V power used by this device.

### No SDA pull-up resistor on the Tap

The SDA line of I2C interface requires a pull-up resistor. This Tap does not contain such a resistor and expects that the monitored device will have it.

### Setting up IO Ninja

- Install and run IO Ninja (<http://ioninja.com/>).
- Plug the Serial Tap into the USB port of your PC.
- Launch the **I2C/SPI Tap** plugin.
- Click on the **Tap** drop-down and select the target I2C/SPI Tap. Note: you can connect several I2C/SPI Taps to one PC. If necessary, refresh the list of available Taps.
- Select the I2C or SPI protocol in the **Protocol** dropdown.
- In the case of SPI protocol, set the SPI mode, word length (SPI data bits), and endianness in the **Settings** dialog.
- Click the **Capture** button to start the data acquisition. At this moment, IO Ninja will check the Tap's configuration. If the Tap is configured for a different protocol (for example, for SPI while you need I2C), IO Ninja will reconfigure the Tap. This will only take a couple of seconds. Note that the Tap has two green LEDs M0 and M1. These LEDs indicate the Tap's configuration:
  - In the I2C mode, M0 is ON, and M1 is OFF;
  - In the SPI mode, M0 is OFF, and M1 is ON.
- When monitoring SPI traffic, you can also flip (swap) MISO and MOSI lines' data. From the Tap's point of view, MOSI and MISO are abstract labels. This is because the Tap is neither SPI slave nor SPI master. The tap is a passive observer, and so it is up to you to decide which of the two SPI data lines is MISO, and which is MOSI. Rather than physically swapping the wires connected to the Tap's terminal block, you can achieve the same result by clicking **Flip SPI MISO/MOSI**.

The above is illustrated by screenshots in [I2C Mode Screenshot](#) and [SPI Mode Screenshots](#). The screenshot depicting IO Ninja in I2C mode also contains important comments on Ninja's representation of I2C traffic, specifically, how unacknowledged (unACKed) bytes are shown.

## I2C Mode Screenshot

The screenshot displays the IO Ninja interface with the following annotations:

- Launch the I2C/SPI Sniffer plugin**: Points to the I2C/SPI Tap icon in the top toolbar.
- Select the I2C/SPI Tap you want to work with (you may have several Taps connected to the same PC)**: Points to the dropdown menu showing the selected tap: "@003:003 IO Ninja I2C/SPI Tap".
- If necessary, click to refresh the list of available Taps**: Points to the refresh icon next to the tap selection dropdown.
- Select the I2C protocol**: Points to the "Protocol: I2C" dropdown menu.
- Click Capture to start the data acquisition**: Points to the "Capture" button in the top toolbar.

The main window shows a list of I2C transactions. Annotations explain the color coding and status of data bytes:

- Address byte(s) are shown in red**: Points to the address field in a transaction (e.g., "I2C start (read, I2C 7-bit address: 0x32)").
- Data bytes in write transactions are shown in blue, data bytes in read transactions are shown in green**: Points to the data bytes in a write transaction (blue) and a read transaction (green).
- When a byte of data is crossed out, this means that this byte was not ACKed. Data bytes that are not crossed out received an ACK.**: Points to a crossed-out data byte in a transaction.
- IO Ninja shows transaction type (read or write) and the address contained in the address byte(s)**: Points to the transaction description and address field.

The right-hand pane shows the "Information" section with a table of properties:

Property	Value
<b>USB connection</b>	
Session time	00:00:20
TX total bytes	9
TX throughput	0
RX total bytes	9
RX throughput	0
<b>Throughput calculator</b>	
Time span	no selection
TX total bytes	no selection
TX throughput	no selection
RX total bytes	no selection
RX throughput	no selection
<b>Checksum calculator</b>	
CRC-16	no selection
CRC-16 (Mod...	no selection
CRC-16 (XMo...	no selection
CRC-16 (USB)	no selection
CRC-32	no selection
IPv4 checksum	no selection
SUM-8	no selection
SUM-16 (little...	no selection
SUM-16 (big-e...	no selection
<b>Log statistics</b>	
Line count	23
Record count	35
Record file size	975
Index file size	184

At the bottom of the window, it shows "Capturing Ln 23 Col 9 Of 0000".

## SPI Mode Screenshots

There are two screenshots in this section.

Annotations in the screenshot:

- Launch the I2C/SPI Sniffer plugin
- Select the I2C/SPI Tap you want to work with (you may have several Taps connected to the same PC)
- Click **Settings** to select SPI mode, number of bits per word, endianness, and other parameters
- If necessary, click to refresh the list of available Taps
- Select the SPI protocol
- Click **Capture** to start the data acquisition
- This button allows you to flip (swap) MOSI and MISO data

The main window displays the following log entries:

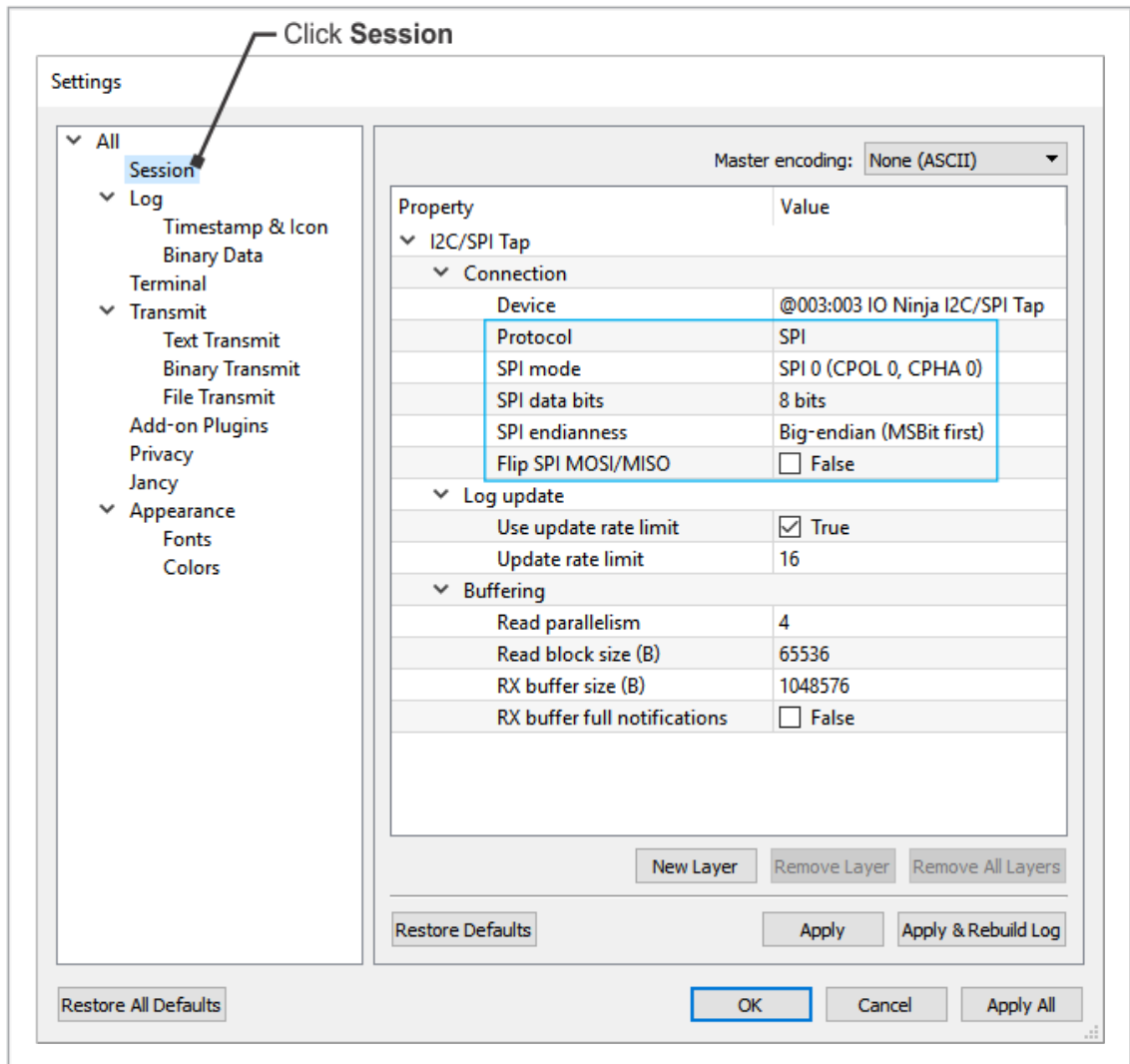
```

18:06:39 +00:00.000 Session started
18:06:39 +00:00.000 Opened tap device @003:003 IO Ninja I2C/SPI Tap
18:06:39 +00:00.001 FPGA firmware is up-to-date
18:06:39 +00:00.202 Hardware buffer cleared
18:06:39 +00:00.202 Capture started
18:06:42 +00:02.765 SPI SS low
18:06:42 +00:02.765 <-> 0000 01 23 45 67 89 ab cd ef .#Eg.¶i 0000 fe dc ba 98 76 54 32 10 p0°.vT2.
18:06:42 +00:02.765 <-> 0008 aa bb cc dd ee ff *¶IYiy 0008 55 44 33 22 11 00 UD3"..
18:06:42 +00:02.765 SPI SS high
18:06:42 +00:02.765 SPI SS low
18:06:42 +00:02.765 <-> 0000 a5 81 54 69 62 62 6f 20 ¥.Tibbo 0000 5a 7e ab 96 9d 9d 90 df Z~e....8
18:06:42 +00:02.765 SPI SS high
18:06:42 +00:02.765 SPI SS low
18:06:42 +00:02.765 <-> 0000 01 23 45 67 89 ab cd ef .#Eg.¶i 0000 fe dc ba 98 76 54 32 10 p0°.vT2.
18:06:42 +00:02.765 <-> 0008 aa bb cc dd ee ff *¶IYiy 0008 55 44 33 22 11 00 UD3"..
18:06:42 +00:02.765 SPI SS high
18:06:42 +00:02.765 SPI SS low
18:06:42 +00:02.765 <-> 0000 a5 81 54 69 62 62 6f 20 ¥.Tibbo 0000 5a 7e ab 96 9d 9d 90 df Z~e....8
18:06:42 +00:02.765 SPI SS high
  
```

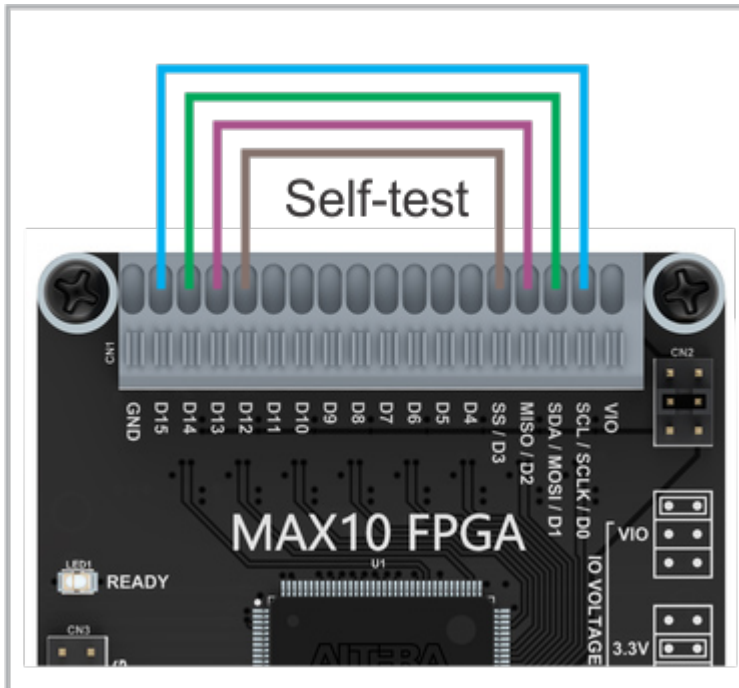
The information panel on the right shows the following statistics:

Property	Value
Session time	00:00:38
TX total bytes	44
TX throughput	0
RX total bytes	44
RX throughput	0
<b>Throughput calculator</b>	
Time span	no selection
TX total bytes	no selection
TX throughput	no selection
RX total bytes	no selection
RX throughput	no selection
<b>Checksum calculator</b>	
CRC-16	no selection
CRC-16 (Mod...)	no selection
CRC-16 (XMo...)	no selection
CRC-16 (USB)	no selection
CRC-32	no selection
IPv4 checksum	no selection
SUM-8	no selection
SUM-16 (little-...)	no selection
SUM-16 (big-e-...)	no selection
<b>Log statistics</b>	
Line count	19
Record count	17
Record file size	589
Index file size	184

Bottom status bar: Capturing Ln 19 Col 12 Ofc 0000



## Using Build-in Test Generator



I2C and SPI firmware (FPGA configurations) for this Tap include test generators producing sustained test traffic at 400KHz and 24MHz respectively.

Test I2C signals are generated on D14 (SDA) and D15 (SCL).

Test SPI signals are generated on D12 (SS), D13 (MISO), D14 (MOSI), and S15 (SCLK).

To receive the test traffic, connect two or four wires as shown on the diagram above.

Here is the data you will receive from the test generators.

### I2C Test Sequence



```

I2C start (read, I2C 7-bit address: 0x08)
→ 0000 11 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e .....
→ 0010 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e .....
→ 0020 1f 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e . !"#%&'()*+,-.
→ 0030 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e /0123456789;<=>
→ 0040 3f 40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e ?@ABCDEFGHIJKLMN
→ 0050 4f 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e OPQRSTUVWXYZ[\]^
→ 0060 5f 60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e _`abcdefghijklmn
→ 0070 6f 70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e opqrstuvwxyz{|}~
→ 0080 7f 80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e .....
→ 0090 8f 90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d 9e .....
→ 00a0 9f a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad ae . |<f*¥!$"e^«-.@
→ 00b0 af b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 ba bb bc bd be ~°±³´µ¶·¸¹º»¼½
→ 00c0 bf c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 ca cb cc cd ce ¿ÀÁÂÃÄÅÇÈÉÊËÌÍÎ
→ 00d0 cf d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd de ÌÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞ
→ 00e0 df e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 ea eb ec ed ee Æàáâãäåæçèéêëìíî
→ 00f0 ef f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd fe ïðñóôõö÷øùúûýþ
→ 0100 ff 79 7a- ÿÿ=

I2C stop
I2C start (write, I2C 7-bit address: 0x08)
← 0000 10 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e .....
← 0010 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e .....
← 0020 1f 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e . !"#%&'()*+,-.
← 0030 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e /0123456789;<=>
← 0040 3f 40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e ?@ABCDEFGHIJKLMN
← 0050 4f 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e OPQRSTUVWXYZ[\]^
← 0060 5f 60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e _`abcdefghijklmn
← 0070 6f 70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e opqrstuvwxyz{|}~
← 0080 7f 80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e .....
← 0090 8f 90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d 9e .....
← 00a0 9f a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad ae . |<f*¥!$"e^«-.@
← 00b0 af b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 ba bb bc bd be ~°±³´µ¶·¸¹º»¼½
← 00c0 bf c0 c1 c2 c3 c4 c5 c6 c7 c8 c9 ca cb cc cd ce ¿ÀÁÂÃÄÅÇÈÉÊËÌÍÎ
← 00d0 cf d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 da db dc dd de ÌÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞ
← 00e0 df e0 e1 e2 e3 e4 e5 e6 e7 e8 e9 ea eb ec ed ee Æàáâãäåæçèéêëìíî
← 00f0 ef f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd fe ïðñóôõö÷øùúûýþ
← 0100 ff 79 7a- ÿÿ=

I2C stop

```

## SPI Test Sequence

```

SPI SS low
0000 00 01 02 03 04 05 06 07 ..... 0000 00 80 40 c0 20 a0 60 e0 ..@À `à
0008 08 09 0a 0b 0c 0d 0e 0f ..... 0008 10 90 50 d0 30 b0 70 f0 ..PÐ0°pð
0010 10 11 12 13 14 15 16 17 ..... 0010 08 88 48 c8 28 a8 68 e8 ..HÈ("hè
0018 18 19 1a 1b 1c 1d 1e 1f ..... 0018 18 98 58 d8 38 b8 78 f8 ..XØ8,xø
0020 20 21 22 23 24 25 26 27 !"#$%&' 0020 04 84 44 c4 24 a4 64 e4 ..DÄ$xdä
0028 28 29 2a 2b 2c 2d 2e 2f ()*+,-./ 0028 14 94 54 d4 34 b4 74 f4 ..TÔ4'tô
0030 30 31 32 33 34 35 36 37 01234567 0030 0c 8c 4c cc 2c ac 6c ec ..LÌ,-li
0038 38 39 3a 3b 3c 3d 3e 3f 89:;<=>? 0038 1c 9c 5c dc 3c bc 7c fc ..\Û<¼|ü
0040 40 41 42 43 44 45 46 47 @ABCDEFG 0040 02 82 42 c2 22 a2 62 e2 ..BÂ"ebâ
0048 48 49 4a 4b 4c 4d 4e 4f HIJKLMNO 0048 12 92 52 d2 32 b2 72 f2 ..RÒ2'rò
0050 50 51 52 53 54 55 56 57 PQRSTUW 0050 0a 8a 4a ca 2a aa 6a ea ..JÊ*+jë
0058 58 59 5a 5b 5c 5d 5e 5f XYZ[\]^_ 0058 1a 9a 5a da 3a ba 7a fa ..ZÛ:°zú
0060 60 61 62 63 64 65 66 67 `abcdefg 0060 06 86 46 c6 26 a6 66 e6 ..FÊ&|fæ
0068 68 69 6a 6b 6c 6d 6e 6f hijklmno 0068 16 96 56 d6 36 b6 76 f6 ..VÖ6vö
0070 70 71 72 73 74 75 76 77 pqrstuvw 0070 0e 8e 4e ce 2e ae 6e ee ..NÎ.óní
0078 78 79 7a 7b 7c 7d 7e 7f xyz{|}~. 0078 1e 9e 5e de 3e be 7e fe ..^P>~p
0080 80 81 82 83 84 85 86 87 ..... 0080 01 81 41 c1 21 a1 61 e1 ..AÁ!;aá
0088 88 89 8a 8b 8c 8d 8e 8f ..... 0088 11 91 51 d1 31 b1 71 f1 ..QŃ1±qñ
0090 90 91 92 93 94 95 96 97 ..... 0090 09 89 49 c9 29 a9 69 e9 ..IÉ)@ié
0098 98 99 9a 9b 9c 9d 9e 9f ..... 0098 19 99 59 d9 39 b9 79 f9 ..YÛ9'yù
00a0 a0 a1 a2 a3 a4 a5 a6 a7 ;e&¥$ 00a0 05 85 45 c5 25 a5 65 e5 ..EÂ&¥eâ
00a8 a8 a9 aa ab ac ad ae af "°«-.-° 00a8 15 95 55 d5 35 b5 75 f5 ..UÕ5puõ
00b0 b0 b1 b2 b3 b4 b5 b6 b7 °±²³´µ¶· 00b0 0d 8d 4d cd 2d ad 6d ed ..MÍ-.mí
00b8 b8 b9 ba bb bc bd be bf ,¹º»¼½ 00b8 1d 9d 5d dd 3d bd 7d fd ..]Ý=½}ý
00c0 c0 c1 c2 c3 c4 c5 c6 c7 ÀÁÂÃÄÅÇ 00c0 03 83 43 c3 23 a3 63 e3 ..CÃ#&cã
00c8 c8 c9 ca cb cc cd ce cf ÈÉÊËÌÍÎ 00c8 13 93 53 d3 33 b3 73 f3 ..SÓ3'só
00d0 d0 d1 d2 d3 d4 d5 d6 d7 ĐŃŎŌŎŎŎ* 00d0 0b 8b 4b cb 2b ab 6b eb ..KĚ+«kĚ
00d8 d8 d9 da db dc dd de df ØÙÚÛÜÝÞ 00d8 1b 9b 5b db 3b bb 7b fb ..[Û;»{û
00e0 e0 e1 e2 e3 e4 e5 e6 e7 áâãäåæç 00e0 07 87 47 c7 27 a7 67 e7 ..GÇ'Šgç
00e8 e8 e9 ea eb ec ed ee ef èéêëìíî 00e8 17 97 57 d7 37 b7 77 f7 ..W×7-w×
00f0 f0 f1 f2 f3 f4 f5 f6 f7 ðñòóôõö÷ 00f0 0f 8f 4f cf 2f af 6f ef ..OÏ/°oï
00f8 f8 f9 fa fb fc fd fe ff øùúûüýþ 00f8 1f 9f 5f df 3f bf 7f ff .._B?¿.ÿ
SPI SS high

```

## Specifications

I2C/SPI Tap is supplied with a USB cable.

### Hardware specifications

USB port	USB 2.0, high-speed (480MHz), USB-C connector
Supported I2C clock speeds	Up to 700KHz
Supported SPI clock speeds	Up to 24MHz <sup>[Note 1]</sup>
Operating temperature	0 to +60 degrees C
Operating relative humidity	10-90%
Mechanical dimensions	82 x 74 x 30 mm

### Note 1

This number depends on the host PC's performance and its "workload." Captured data may be lost if PC's USB channel is unable to receive it in a timely manner. Our tests have shown that an average i7-based desktop PC (that is not performing other time-critical tasks) easily captures sustained SPI traffic with clock speeds up to 24MHz.

*All specifications are subject to change without notice and are for reference only. Tibbo assumes no responsibility for any errors in this Manual and does not make any commitment to update the information contained herein.*

*Is the data you are looking for missing from the above table? Do not just assume that you know the answer — talk to Tibbo! Remember, that the ultimate responsibility for all decisions you make regarding the use and the mode of use of Tibbo products lies with you, our Customer.*

## Ethernet Tap



Ethernet Tap is an affordable sniffer for monitoring Ethernet traffic flowing through a single Ethernet connection. To monitor Ethernet traffic, an Ethernet cable that was initially connecting a switch (hub) to an Ethernet device is replaced by two Ethernet cables and the Ethernet Tap is inserted between them, in what is known as a

"wedge" fashion. The Tap passes the Ethernet traffic through with a delay equal to about 5.2uS.

The Tap connects to your PC via a USB cable and is compatible with IO Ninja terminal/sniffer software.

### Features

- Designed for use with IO Ninja software (<http://ioninja.com/>).
- Reliably captures 10Base-T and 100Base-T Ethernet traffic<sup>[Note 1][Note 2]</sup>.
- Based on Intel MAX10 FPGA.
- Six LEDs onboard:
  - Green Ready LED;
  - Green 100Mbps and yellow Link/Activity LEDs for the left RJ45 jack;
  - Green 100Mbps and yellow Link/Activity LEDs for the right RJ45 jack;
  - Blue Power LED.
- High-speed (480Mbps) USB2.0 interface on a USB-C connector.
- Supplied with a USB-C cable.
- USB-powered, no additional external power necessary.
- Compact, outside dimensions only 82 x 74 x 30 mm.
- The product is field-upgradeable via the USB interface.

#### Note 1

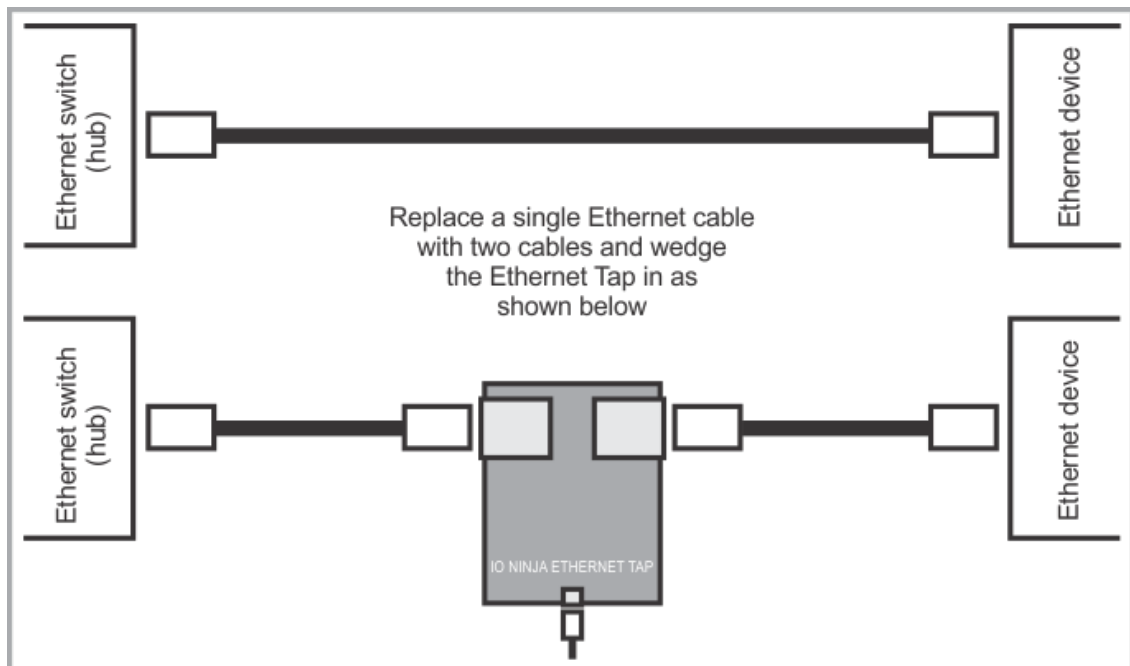
As long as your PC is fast enough to receive all the captured data. Some data may be lost if a PC's USB channel is unable to receive it on time. Our tests have shown that an average i7-based desktop PC (that is not performing other time-critical tasks) easily captures 100BaseT traffic even under high Ethernet load conditions.

#### Note 2

The Tap does not allow mixing 10BaseT and 100BaseT connections. Both sides of the Tap must have the same connection speed. That means that you may not connect one side of the Tap to a 100Base-T device, and another side—to a 10Base-T device.

## Using Ethernet Tap

### Connecting Ethernet Tap



### Setting up IO Ninja

- Install and run IO Ninja (<http://ioninja.com/>).
- Plug the Ethernet Tap into the USB port of your PC—the blue **Power** LED shall turn on. If the Tap is loaded with a valid configuration file, the green **Ready** LED will also turn on.
- Launch the **Ethernet Tap** plugin (we have a screenshot [here](#)).
- Click on the **Tap** drop-down and select the target Ethernet Tap (you can connect several Ethernet Taps to one PC, so selecting the right one may be necessary). You can refresh the list of available Ethernet Taps by clicking the **Refresh** button to the right of the Tap drop-down.
- Click the **Capture** button to start the data acquisition. At this moment, IO Ninja will check the Tap's internal firmware. If the Tap firmware is outdated, IO Ninja will upload the new firmware into the Tap. This will only take a couple of seconds.

### Filtering captured data

Most networks experience constant and varied traffic, while your attention is usually focused on something particular. Filtering allows you to only display the packets that satisfy the defined criteria. The filter is set by entering a filter string into a **Filter** textbox and pressing the **Apply** button to the right of the field. When setting a filter, follow the "pcap filter" format, and if you don't know what that is, Google "pcap-filter."

Clicking the **Apply** button verifies the filter string correctness and enables that filter if no problems are found. If the filter string is invalid, the **Filter** textbox will turn red.

Note that the Ethernet Tap captures all data flowing through it, and the filtering is performed on the already recorded data. Changing the filter affects what packets are displayed on the screen and is retroactive, meaning that setting a new filter causes IO Ninja to re-output the entire log with the new filter setting applied.

## Ethernet Tap Plugin Screenshot

The screenshot displays the IO Ninja application window. At the top, there is a menu bar (File, Edit, View, Session, Help) and a toolbar. Below the toolbar, a filter is set to "ip host 192.168.1.210 and tcp". A dropdown menu shows the selected tap: "@002:017 IO Ninja Ethernet Tap".

Annotations on the left side of the screenshot indicate different levels of data visibility:

- Master log:** Points to the top section of the packet list.
- Packets appear with a payload preview showing up to two lines of data:** Points to the first few lines of the packet list.
- Packet details:** Points to the detailed view of a selected packet.
- Packet contents, field-by-field:** Points to the expanded view of the packet's internal structure.
- HEX dump of the entire packet:** Points to the raw hexadecimal data at the bottom of the packet details.

Annotations on the right side of the screenshot describe the controls:

- Launch the Ethernet Tap plugin:** Points to the gear icon in the toolbar.
- Optionally, set the packet filter (for syntax, Google "pcap-filter"):** Points to the filter text field.
- Click the Apply button to verify and activate the filter:** Points to the green checkmark button.
- Select the Ethernet Tap you want to work with (you may have several Taps connected to the same PC):** Points to the tap selection dropdown.
- Click Capture to start the data acquisition:** Points to the red stop icon in the toolbar.
- This button allows you to export the log into the .pcap (Wireshark) format:** Points to the document icon in the toolbar.
- If necessary, click Refresh to refresh the list of available Taps:** Points to the refresh icon in the toolbar.

The main packet list shows several TCP packets. One packet is selected, and its details are shown on the right. The details include:

- ETHERNET II:** Destination: 40:8D:5C:E3:AA:1D, Source: 00:24:77:52:E4:00, Type: Ip.
- IP:** Header length: 5, Version: 4, Type of service: 0, Total length: 93, Identification: 0x01DF, Flags: DF, Fragment offset: 0, Time to live: 255, Protocol: Tcp, Header checksum: 0xF56B, Source: 192.168.1.210, Destination: 192.168.1.45.
- TCP:** 1000 - 63777 [...PA..]
- TCP Payload:** A hex dump of the payload data, with a string preview: "This string was sent from IO Nin...".

At the bottom right, there is an "Information" panel showing properties for the selected Ethernet tap, such as Session time, TX total bytes, and RX total bytes. Below that is a "Log statistics" panel showing line count, record count, record file size, and index file size.

## Specifications

Ethernet Tap is supplied with a USB cable.

### Hardware specifications

USB port	USB 2.0, high-speed (480MHz), USB-C connector
Throughput	Up to 100BaseT Ethernet's real-life throughput [Note 1] [Note 2]
Operating temperature	0 to +60 degrees C
Operating relative humidity	10-90%
Mechanical dimensions	82 x 74 x 30 mm

### Note 1

As long as your PC is fast enough to receive all the captured data. Some data may be lost if a PC's USB channel is unable to receive it on time. Our tests have shown that an average i7-based desktop PC (that is not performing other time-critical tasks) easily captures 100BaseT traffic even under high Ethernet load conditions.

### Note 2

The Tap does not allow mixing 10BaseT and 100BaseT cables. Both sides of the Tap must have the same connection speed. That means that you may not connect one side of the Tap to a 100Base-T device, and another side—to a 10Base-T device.

*All specifications are subject to change without notice and are for reference only. Tibbo assumes no responsibility for any errors in this Manual and does not make any commitment to update the information contained herein.*

*Is the data you are looking for missing from the above table? Do not just assume that you know the answer — talk to Tibbo! Remember, that the ultimate responsibility for all decisions you make regarding the use and the mode of use of Tibbo products lies with you, our Customer.*

## Update history

### **09DEC2019**

- Documented the [Ethernet Tap](#).

### **09SEP2019**

- Corrected an error in the [Serial Tap](#) documentation (UART (TTL) mode allows to monitor UARTs of IC chips with logic levels from 3.3V, not 1.8V as was previously stated).

### **26MAR2019**

- Documented the [I2C/SPI Tap](#).

### **18SEP2018**

- Updated Serial Tap documentation to reflect the changes made in revision D.

### **03JUL2018**

- Documented the [Serial Tap](#).