# bbtkv2

*Release 0.0.9*

**Christophe Pallier**

**Apr 26, 2024**

# CONTENTS:

CHAPTER

ONE

# ACQUIRING TIMING DATA USING THE BBTKV2 PYTHON MODULE


**The BlackBox ToolKit v2 is a device that allows psychologists to**

measure the timing of audio-visual stimuli with sub-millisecond accuracy. It replaces a digital oscilloscope (capturing activity on sound and visual sensors, or TTL signals) and a signal generator (generating sound or TTL signal. (See https://www.blackboxtoolkit.com/bbtkv2.html for more information)[1].

The principle of operation is simple. Three pieces of equipement are needed:

1. A stimulation device (typically a computer)

[1] Nowadays, one can build a "poor man's" Blackboxtoolkit around an Arduino or a Raspberry Pi. But it takes quite a bit of time to build the right sensors and validate them. If you have a BBTKv2 around you, or enough money to acquire one, it will save you time. Another alternative, od course, is to use a digital oscilloscope, but these beasts can be complicated to use.

2. The bbtkv2 with input sensors (photodiodes, sound detectors, TTL detectors) attached to the stimulation device.
3. A host computer driving the bbtkv2 (hooked to it via a USB cable).

---

**Note:** The stimulation PC and the host PC *can* be the same computer (but do not have to). As data are recorded asynchronously by the BBTKv2, it is possible for a single PC to switch the BBTKv2 into "capture mode", launch the stimulation program and, when done, download the timing data from the BBTKv2 memory.

---

**Note:** Under Windows, you need to install a driver to communicate with the BBTKv2. You can install the mbed-cli available from https://os.mbed.com/docs/mbed-os/v6.16/quick-start/build-with-mbed-cli.html a,d check install driver during the setup.

---

The BBTKv2 and the host PC communicate via a serial protocol over USB. The *bbtkv2* Python module provided here encapsulates (some of) the commands documented in *The BBTKv2 API Guide* sold by the parent company.

For example, one of the most useful commands, `capture()`, monitors the input sensors for a certain time period and records all changes on any input line (raising or falling closing edges). Once the capture period has elapsed, the BBTKv2 sends the list of all the events, timestamped, to the host computer.

## 1.1 Installing the Python module

Just run:

```
pip install bbtkv2
```

(Note: Should the pip command not work, just copy bbtkv2.py, and make sure you have the pyserial module installed)

## 1.2 Using the module

### 1.2.1 Configuration

To use the module you need to know two thing:

- The name of the serial port associated to the BBTKv2.

  On most Linux system, it will be `/dev/ttyACM0` (which can be identified by running the `dmesg` command just after plugging the BBTKv2).

- the baudrate, that is, the speed of transmission of information over the the serial connection.

  When you plug the BBTKv2 in, a USB storage device named `BBTKV2` is mounted, which contains a file `BBTK.ini` specifying this parameter. On my computer:

  ```
  $ cat BBTKV2/BBTK.ini
  [BaudRate]
  57600
  ```

Then, you can create a `bbtkv2.toml` configuration file such as the one below (change the `serialport` and `baudrate` variables according to your setup. For example, under Windows, `serialport` might be *COM3*):

```
serialport="/dev/ttyACM0"
baudrate=57600
debug=1

[thresholds]
Mic1=100
Mic2=0
Sounder1=63
Sounder2=63
Opto1=110
Opto2=0
Opto3=0
Opto4=0

[smoothing]
smoothing='11000011'
```

You will modify the thresholds and smoothing parameters later if needed (read the official documentation of the bbtkv2).

### 1.2.2 Using the bbtkv2 module to capture events

Launch `ipython` and type:

```
import bbtkv2

BBTK_CONF_FILE = "./bbtkv2.toml"  # change to the conf file you created

bb = bbtkv2.BlackBoxToolKit(BBTK_CONF_FILE)

bb.adjust_thresholds()  # adjust the thresholds manually
bb.clear_timing_data()  # clear the internal memory of the BBTKv2
text = bb.capture(30)   # start capturing events for 30sec

# convert the results into human readable formats:
df1 = bbtkv2.capture_output_to_dataframe(text)
processed_events = bbtkv2.capture_dataframe_to_events(df1)
print(processed_events)
```

If things do not seem to work, you may first to test that the link with the bbtk2 works correclty by running an interactive serial communication software. This procedure is described for Windows in *The BBTKv2 API Guide* using *TeraTerm VT*. In the next chapter, we explain how to perform the same test under Linux, using *minicom*.

For example, if you notice that the transmission is garbled, you should decrease this speed in the `BBTK.ini` file and reboot the BBTKv2 box.

CHAPTER

TWO

# CHECK THE BBTK V2 USING A SERIAL COMMUNICATION SOFTWARE

Before using the python module, we recommend to check the communication between the BBTKv2 and the PC using a serial communication program, following instructions in the next section.

1. Power on the Bbtk_v2, then link it through USB cable to your computer

   A usb-storage device `BBTKV2` should be detected and mounted, and a USB ACM device `/dev/ttyACM0` should have been created.

   Determine the Baud Rate:

```
cat /media/*/BBTKV2/BBTK.ini
[BaudRate]
230400
```

2. Launch a serial communication program (e.g. minicom if you are runnning Linux — you may have to install it with `sudo apt install minicom`), and open `/dev/ttyACM0`, set the configuration to 8bits, NoParity, and a Buda RAte equal to the one you read from the `BBTK.ini` file.

   minicom -8 -D /dev/ttyACM0 -b 230400

   The terminal should display something like:

```
Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyACM0, 17:31:53

Press CTRL-A Z for help on special keys
```

   Press `CTRL-A Z` and select `Local Echo 'on'`, and `Add Carriage Ret`, and `Send a Break`

3. Run a short acquisition session:

- Send the command `CONN`. In return, you should get the messgge `BBTK;`
- Enter the command GEPV read the sensors' thresholds.
- Enter `SEPV` and 8 times `63` to set the sensors' thresholds
- Enter `SPIE` to erase internal memory. Check the display of the Bbtkv2.
- Enter `ICHK` and create events (e.g. putting a light on/off on a photodetector), each event should generate the display of a line with 12 binary digits. Send a Break `CTRL-A Z F` to interupt the process;
- To capture events for 10s:

```
DSCM
TIML 10000000
RUDS
```

```
File  Edit  View  Search  Terminal  Help

Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyACM0, 17:45:39

Press CTRL-A Z for help on special keys

CONN
BBTK;
GEPV
000,000,000,000,000,000,000,000;
SEPV
63
63
63
63
63
63
63
63
GEPV
063,063,063,063,063,063,063,063;
SPIE
FRMT;
DONE;
DSCM
TIML
10000000
RUDS

SDAT;
27;
9999999;
40000;
000000000000000000000000000000000;
000010010000000000000000000000250;
000000010000000000000000000000500;
000000000000000000000000000765750;
000000010000000000000000001091750;
000000000000000000000000001948500;
000000010000000000000000002168000;
000000000000000000000000003054500;
000000010000000000000000003201500;
000000000000000000000000003989250;
000000010000000000000000004150000;
000000000000000000000000005030250;
000000010000000000000000005165500;
000000000000000000000000006064000;
000000010000000000000000006223750;
000000000000000000000000006947750;
000000010000000000000000007094000;
000000000000000000000000008100250;
000000010000000000000000008272500;
000000000000000000000000008963000;
000000010000000000000000009068500;
000000000000000000000000009460000;
000000010000000000000000009524250;
000000000000000000000000009711000;
000000010000000000000000009745250;
000000000000000000000000009917000;
000000010000000000000000009933000;
EDAT;
```

The information about the events is provided in the `SDAT;` and `EDAT;` lines. The lines containing 32 digits encode the events in the following manner: the first 12 digits reprenet the status of input ports, the next 8 digits describe the ouput ports (and should all be zero with the DSC command), and the last 12 digits indicate the time in microseconds since

the start of the acquisition run.

CHAPTER

THREE

# API

CHAPTER

FOUR

# INDICES AND TABLES

- genindex
- modindex
- search

# PYTHON MODULE INDEX

## b

# INDEX