RTBox v1: USTC Response Time Box

http://lobes.osu.edu/rt-box.php

What is RTBox for?

Computer keyboard and mouse can be used to record response time (RT) to an event, such as visual or auditory stimulus. However, the accuracy depends on many factors, such as computer hardware specification, operating system, programming software, user code, and so on. Even if the user code is well written, these devices are inaccurate, and even worse, often introduce bias. The major reason is that, the user program can get the time when the program reads the key or mouse response, not the time when the key or mouse is pressed.

The USTC Response Time Box (RTBox) is designed to measure response time with high accuracy. The microcontroller in the device will record the event time and button identity. The user code can read them anytime when convenient. Unlike keyboard or mouse response, the timing is independent of time your program reads the response.

Features

- Compatible to major computer systems, including Windows, MAC and Linux
- USB 1.1 and 2.0 compatible
- Measure both the button press and release time
- Built-in light port and pulse port for trigger and calibration
- Built-in firmware upgrade so the device will never expire
- Analog-to-Digital Converter function

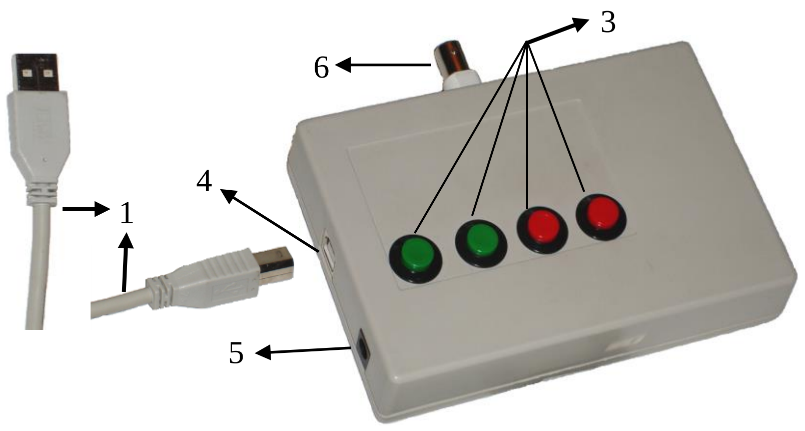
Specification

- Four buttons allowing user to label with descriptive names
- Detection time resolution: about 6 μs (90 μs prior to v1.9)
- Dimensions: 5.5 x 4.5 x 1 (h) inches, 14 x 11 x 2.5 (h) cm
- Weight: ~5 oz

Parts of the Device

1. USB cable

2. Photodiode with suction and cable
3. Four buttons
4. USB port: connect to computer USB port with the cable (1)
5. Light port: receive light signal from the provided photodiode (2)
6. Pulse port: receive pulse signal from stimulator (BNC connector)



How RTBox works?

When the device is connected to a USB port of a computer, it works as a serial port. The device is powered by the USB port.

For principle about how the device works, you can check the paper on *Behavior Research Methods*.

Basically, the device detects button and port events with interval about 6 μs. When it detects an event, it records the event code and time based on its own clock, and sends them to the computer serial port. Each event contains 7 bytes of data. The first byte is the event code, and the rest 6 bytes contain the timestamp.

At the computer side, the device driver reads the data from serial buffer, identifies the event type, and calculates the response time based on device clock.

Install driver for the first time

Modern operating systems all include the USB-serial driver for RTBox, or will automatically install the driver when the device is connected to a computer for the first time.

Starting from October 2017, RTBox code can use FTDI D2XX driver, which can access the RTBox directly even if the operating system won't recognize it as a serial port. It is recommended to use this driver if possible, but some tricks are needed for OSX and Linux. Here are some details for how to choose different drivers for different operating systems.

1. Windows system

Simply follow system instruction to install FTDI driver for the first time. RTBox code can use either D2XX driver or VCP driver. The latency timer issue is solved by D2XX driver easily.

2. OSX

Later OSX systems all include Apple's VCP driver. RTBox code will work, but the latency timer change is almost impossible. If you don't use FTDI devices other than RTBox, the best solution is to disable the built-in Apple driver by following command:

```
cd /System/Library/Extensions/
sudo mv AppleUSBFTDI.kext AppleUSBFTDI.disabled
sudo touch /System/Library/Extensions
```

We don't realize any advantage of FTDI VCP driver for OSX, so please don't install it. You can enable the VCP driver by changing back the name:

```
cd /System/Library/Extensions/
sudo mv AppleUSBFTDI.disabled AppleUSBFTDI.kext
sudo touch /System/Library/Extensions
```

To use FTDI D2XX driver, you will need to download it for your OS from http://www.ftdichip.com/Drivers/D2XX.htm, and follow its instruction to install it.

3. Linux

The USB-serial driver is part of the Linux kernel. If Psychtoolbox is configured correctly, no any issue is expected. If there is latency timer complain, following the instruction, and set up Psychtoolbox with sudo.

In case the Matlab or Python code fails to open the serial port, it is likely due to access issue, and you will need to do the following once:

```
sudoedit /etc/udev/rules.d/50_RTBox.rules
```

Paste the following line into the file:

```
SUBSYSTEM=="usb", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001", MODE="0666"
```

Then save the file, and re-plug the USB cable. If you like to try D2XX driver, you can remove VCP driver module temporarily by:

```
sudo rmmod ftdi_sio
```

After you restart system or re-plug the USB, the module will be enabled.

You can also prevent the built-in FTDI driver to be loaded by blacklisting it:

```
sudo gedit /etc/modprobe.d/blacklist.conf
```

Insert following line into the file, save and reboot:

```
blacklist ftdi_sio
```

To use FTDI D2XX driver, you will need to download it for your OS from http://www.ftdichip.com/Drivers/D2XX.htm, and follow its instruction to install it.

How to use RTBox?

There are two ways to use the device to measure response time. If the user code has accurate stimulus onset timestamp, we need only to get the button time based on the same clock as the onset timestamp, and then do a subtraction to get the response time. This is the recommended method. This method relies on the method we developed to synchronize the device clock with computer clock.

The second way is to provide a trigger to the device to indicate the onset of stimulus. The device will detect both trigger and button events. We get the response time by calculating the time difference between the two events. This method is only needed when the user code doesn't have accurate stimulus onset time.

If there is a TTL pulse synchronized with stimulus, you can connect it to the sound port with a cable (not provided). Some stimulus equipment, such as an electrical stimulator and audio stimulator, has built-in trigger output for this purpose.

For computer-controlled visual stimulus, it may not be easy to generate an accurate trigger. In this case, the possible timing error could be from many factors, such as the time difference between nominal onset in user code and real display onset, time difference due to the stimulus location on computer screen etc. We provide a photodiode as light trigger for visual experiments. You can mount the suction onto your screen, and program a light square which is within the same frame as the onset of your stimulus. If you use only grayscale visual stimulus, you can use our video switcher to provide a pulse trigger a pulse trigger (http://lobes.usc.edu/videoswitcher).

The second method requires additional hardware connection. Although it is the most accurate solution, it is less convenient. Also the photodiode and the trigger light may be a distractor for the subjects.

Driver code in MatLab/Octave and Python

We provided code for MatLab/Octave (RTBox.m, RTBoxClass.m) based on PsychToolbox and Python (RTBox.py). The code can detect the device automatically, and use all its functionality. If you don't use these packages, you may "translate" the code into your own language, suppose your software environment has similar functions for serial communication and timestamp.

There are also some demo codes, showing how to use RTBox in an experiment.

How to do calibration?

The timing of device is very accurate, and you normally don't need to calibrate it. However, there may be a time difference between the nominal and the real onset of the stimulus, and you can measure this difference and apply the difference to the measured response time. Ideally, this difference should be fixed for a certain setup.

The light and pulse/sound ports at the RTBox hardware can be used for calibration purpose, especially for the computer-based visual stimulus. The included light sensor is designed to detect the light onset, so we can compute the difference from nominal onset from video buffer flip time.

Both Matlab and Python code package have demos for the calibration.

How to test synchronization of computer and device clocks?

We synchronize the computer and device clocks by a serial trigger. When we send a trigger to the device, we record the computer time when the trigger is sent, and get the device time when the device receives the trigger. Then we get the difference between the two clocks. Later when we have device time for an event, we can calculate its computer time based on the clock difference. When possibly inaccurate clock synchronization is detected, you will see a warning message. This typically indicates your system is not accurate on timing, not only for the response box, but also for other timing related measurement.

The speed of the computer and device clocks may be slightly different. The driver code will recommend correcting clock ratio for each host computer. The code will measure and apply the correction automatically.

Frequently asked questions and answers

1. I am warned to do run ClockRatio. What is it for, and do I have to do it?

Answer: the short answer is no. However, clock ratio correction will further improve timing accuracy. And you'd better to run it once a while, or if you have any change to your host computer, like hardware change or major system update. Another simple way is to run the 30-sec RTBoxSyncTest, and it will inform to correct ratio if needed.

2. I am asked to change latency timer to 2ms, or warned "Failed to change latency timer due to insufficient privilege" or asked sudo password to change it. What does that mean, and how do I proceed?

Answer: This indicates that you are using VCP driver under OSX or Linux. If you don't have to use VCP driver, you can follow the instruction in "Install driver for the first time" to disable the VCP driver.

If you do need VCP driver (only necessary if you have to use VCP driver for another FTDI USB serial device), you can follow the instruction below to change the latency timer to 2ms. The latency timer setting won't affect the accuracy of RTBox timing, but can make serial port reading faster. For example, on Windows and MACI systems, if the latency time is default 16 ms, 'clear' method in each trial will take about 380 ms, while with 2 ms, it takes about only 60 ms.

You can either follow the instruction in the warning to change the timer, or do it by yourself. The manual method varies with operating system. Under Linux, the driver will likely adjust the timer by itself.

Under Windows, you need to go to Device Manager (right click Computer -> Manage) -> Ports (COM & LPT) -> Right click the RTBox USB serial port -> Properties -> Port Settings -> Advanced -> Change Latency Timer (msec) to 2.

Under latest OSX, there may be no easy way to change the latency timer, except to unload VCP driver.

3. Does it make difference if I plug the RTBox to different USB port?

Answer: from our test, we didn't see time accuracy difference for different USB ports. However, we do see, on some computers, that reading speed is a little slower if the RTBox is connected to a port from a USB hub. As a rule of thumb, the USB ports at the back of a desktop computer may be better.

Contact us

If you have question or suggestion about the device, please contact us at:

Xiangrui Li
B71 Psychology Building
The Ohio State University
1835 Neil Ave
Columbus, OH 43210

Phone: (614) 292-1847
Email: xiangrui.li@gmail.com

Appendix 1: Serial Commands

This lists all the serial commands, and the returned data, if any, by the device. All commands are a single byte data. The following are for firmware v1.91+. If you are using early version, please update to the latest firmware.

Uint8	Char	Returned	Function	Comments
63	?	[63 state]	Ask button states	Return [state 63] for 1.4-
66	B		Enter boot mode from simple	R to return from boot
69	E	[69 state]	Ask Enable state	1.4+
101	e	E	Wait for enable byte	1.91+
16			Send data to EEPROM	1.91+
17			Read data from EEPROM	1.91+
88	X	USTCRTBOX,92160	Ask device identity	Also switch to advance
120	x		Switch to simple mode	1.4+
89	Y	[89 6-byte time]	Ask device time	
71	G		Go to ADC firmware	1.8+

For command "?", bits 0~3 of the returned state byte are for buttons 1 to 4.

Command "B" is used for firmware update, so you should not use it.

For command "E", the 1~6th bits of the returned state byte are for button press,
button release, pulse/sound, light, TR and aux respectively.

Command "T" is used by firmware developers to test the round trip speed, and is not available for normal device.

Command "X" switches device into advanced mode, and returns the device ID "USTCRTBOX", device clock frequency 921600, and version of firmware