Boks reference guide (draft)

Sebastiaan Mathôt
Last updated: 11/18/12

Table of Contents

Boks reference guide (draft)....................1
About Boks....................1
Availability....................2
OpenSesame plug-in....................2
Python module....................2
USB Serial port communication....................2
Timestamps....................2
Serial port settings....................2
CMD_RESET [001]....................2
CMD_IDENTIFY [002]....................2
CMD_WAIT_PRESS [003]....................2
CMD_WAIT_RELEASE [004]....................3
CMD_WAIT_SLEEP [005]....................3
CMD_BUTTON_STATE [006]....................3
CMD_SET_T1 [007]....................3
CMD_SET_T2 [008]....................3
CMD_SET_TIMEOUT [009]....................3
CMD_SET_BUTTONS [010]....................3
CMD_GET_T1 [011]....................3
CMD_GET_T2 [012]....................3
CMD_GET_TD [013]....................3
CMD_GET_TIME [014]....................3
CMD_GET_TIMEOUT [015]....................4
CMD_GET_BUTTONS [0016]....................4
Arduino schematics....................4

About Boks

Boks is an open-source response box for use in psychological/ neuroscientific experiments. The aim is to provide a low-cost, professional grade response device, which allows researchers to collect participant responses with sub-millisecond temporal precision. The Boks is primarily intended for use with the OpenSesame experiment builder, but can in principle be used with any software package for conducting experiments, as long as the software supports either Python scripting or serial port communication.

Availability

Boks is currently under development, and is not yet available for purchase.

OpenSesame plug-in

The OpenSesame Boks plug-in is located in:

```
opensesame/boks
```

To install the plug-in, copy the plug-in folder to the OpenSesame plug-in folder, as described here: http://osdoc.cogsci.nl/plug-ins/12-plug-in-installation. After you have installed the plug-in and restarted OpenSesame, you should see the Boks plug-in icon appear in the item toolbar:

You can test the plug-in with the test experiment, which is located in:
`misc/boks_example.opensesame`

Python module

The Python module is libboks.py. For documentation, see doc/libboks.html.

USB Serial port communication

Communication occurs by sending a single command byte to the Arduino. Depending on the command, the command byte should be followed by one or more bytes that serve as parameters. Depending on the command, the Arduino responds by sending zero or more bytes in response. The command bytes are indicated in decimal notation in the square brackets.

Timestamps

All timestamps are in microseconds. Only the CMD_WAIT_SLEEP command will fall back to millisecond precision when the timeout has been set to more than 16 milliseconds. This is a limitation of the Arduino.

Serial port settings

The Arduino is connected via USB as a virtual serial port device. The baudrate is 115200.

CMD_RESET [001]

Resets the Boks to the initial state.

CMD_IDENTIFY [002]

Returns a sequence of 21 bytes, which should be interpreted as two ASCII strings. The first string is 5 bytes and contains the firmware version of the Boks. The second string is 16 bytes and contains the model description, optionally right-padded with whitespace.

Return example

```
'0.1.0dev.boks        '
```

CMD_WAIT_PRESS [003]

A single byte that corresponds to the button that is pressed, i.e. 1, 2, 3, or 4. If a timeout occurs (see CMD_SET_TIMEOUT), 255 is returned. Only buttons that are pressed after the command has been send are returned. Function also sets T2 to the moment that the button press has been detected.

CMD_WAIT_RELEASE [004]

Returns a single byte that corresponds to the button that is released. For more information, see CMD_WAIT_PRESS.

CMD_WAIT_SLEEP [005]

Waits for the interval specified by CMD_SET_TIMEOUT.

CMD_BUTTON_STATE [006]

Returns a single byte that contains the state for each of the buttons. The first bit (rightmost) indicates the state of the first button, the second bit indicates the state of the second button, etc.

Return example (binary)

```
00001111 # All buttons pressed
00000010 # Button 2 pressed
```

CMD_SET_T1 [007]

Sets the T1 (Timestamp 1) of the Arduino to the current time in microseconds, as measured by the Arduino's internal clock. Does not return a value.

CMD_SET_T2 [008]

Sets the T2. For more information, see CMD_SET_T1.

CMD_SET_TIMEOUT [009]

Sets the timeout value, which is used by CMD_WAIT_PRESS, CMD_WAIT_RELEASE, and CMD_SLEEP. The command byte should be followed by an unsigned long (4 bytes) that indicates the timeout in microseconds. The value 0 disables the timeout.

CMD_SET_BUTTONS [010]

Sets the buttons that should be polled by CMD_WAIT_PRESS and CMD_WAIT_RELEASE. The command byte should be followed by a single byte that indicates which buttons should be pressed. The first (rightmost) bit corresponds to the first button, etc. To prevent deadlocks, you cannot turn off all buttons. If you try do this, by sending a 0-byte, all buttons will be switched on instead.

Parameter example (binary)

```
00001111 # Poll all buttons
00000011 # Only poll buttons one and two
00000000 # Turn on all buttons (special case!)
```

CMD_GET_T1 [011]

Returns T1 as an unsigned long (4 bytes).

CMD_GET_T2 [012]

Returns T2 as an unsigned long (4 bytes)

CMD_GET_TD [013]

Returns the difference between T2 and T1 (T2 – T1) as an unsigned long (4 bytes).

CMD_GET_TIME [014]

Returns the current time according to the Arduino's internal clock as an unsigned long (4 bytes).

CMD_GET_TIMEOUT [015]

Returns the timeout value as set by CMD_SET_TIMEOUT as an unsigned long (4 bytes).

CMD_GET_BUTTONS [0016]

Returns the active buttons as set by CMD_SET_BUTTONS as a single byte. For more information, see CMD_SET_BUTTONS.

Arduino schematics

TODO