USE SyncBoxUser Guide

Written by Christopher Thomas – December 10, 2018.
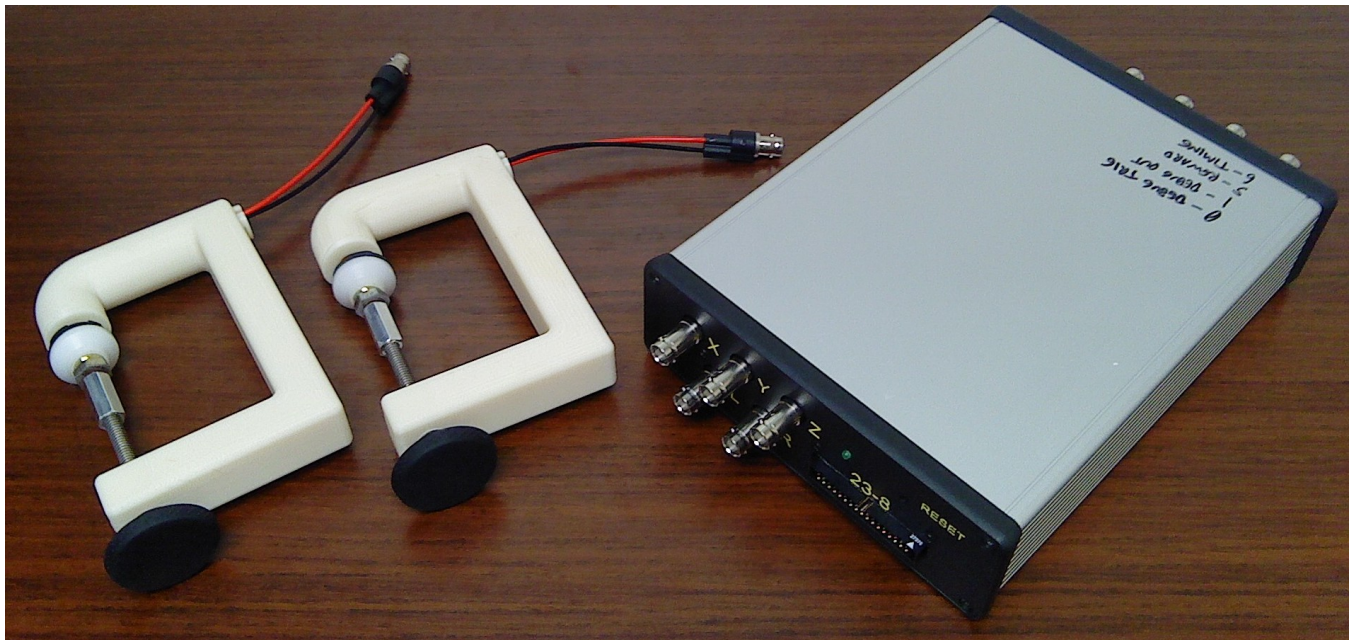
Contents

Chapter 1

Overview

The USE SyncBox is a device that lets a host computer and several instruments talk to each other. It was commissioned by the Attention Circuits Control Laboratory to facilitate their experiments[1]. A system diagram of the York University installation is shown in Figure 1.1. A diagram of the I/O signals used by the USE SyncBox is shown in Figure 1.2.
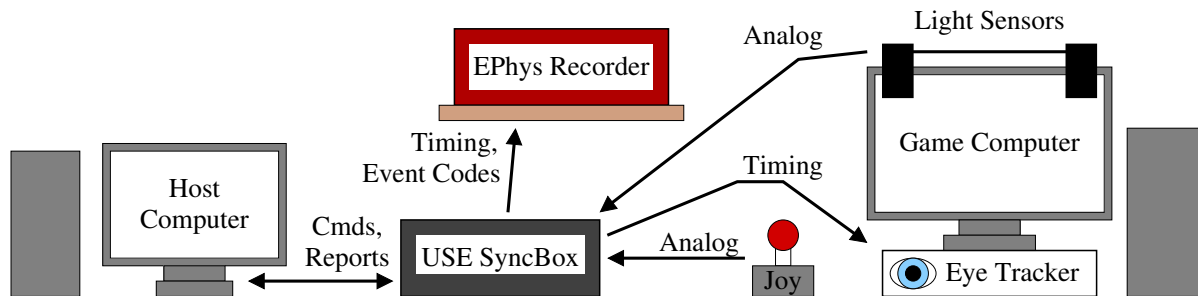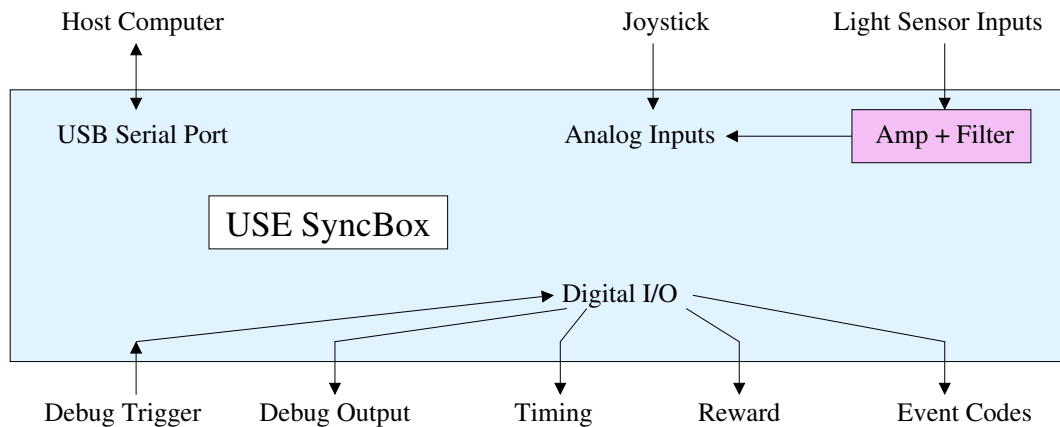

Figure 1.1: System block diagram.

Figure 1.2: USE SyncBox inputs and outputs.

---

[1]USE SyncBox integration is described in: "USE: An integrative suite for temporally-precise psychophysical experiments in virtual environments", Watson, M. R., Voloh, B., Thomas, C. J., Hasan, A. M., and Womelsdorf, T. (2018), *bioRxiv*, 434944 doi:10.1101/434944

The USE SyncBox handles several types of communication:

- It talks to the host computer over a USB serial port connection.

- It generates timing pulses. These are controlled using the **Txx** series of commands. These are typically used to provide events with known timestamps to equipment such as eye-trackers and electrophysiology instruments.

- It generates single pulses of variable duration. These are controlled using the **RWD** command. These are typically used for dispensing a reward to test animals.

- It emits binary number values over a parallel digital interface. These are controlled using the **Nxx** series of commands. These are typically used as event codes for electrophysiology equipment.

- It reads analog information from three general-purpose analog inputs, and shows this data during logging. These are typically used for joystick inputs.

- It reads light sensors to synchronize with the game computer's display. The light sensor inputs have hardware amplification and filtering (they are special-purpose inputs, not general analog inputs). Light levels are shown during logging.

- Logged data is controlled using the **Lxx** series of commands.

The commands are described in detail in Section 2.

To started, connect to the USE SyncBox (USB serial link at 115200 baud, 8N1), and type "**?**" (and enter) for help. Type "**QRY**" to see the USE SyncBox's settings.

The front and back panels of the USE SyncBox are shown in Figure 1.3. The pinout of the parallel output is shown in Figure 1.4.
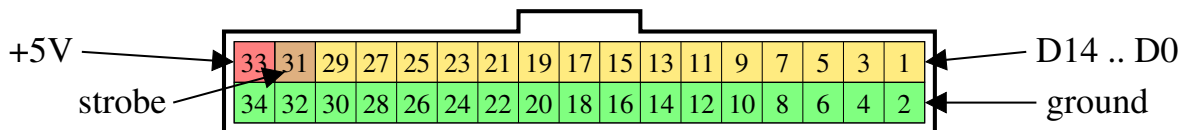

Figure 1.3: USE SyncBox front and back panels.

Figure 1.4: Event code connector pinout.

Chapter 2

Command Interface

The USE SyncBox is connected to the host computer via a USB serial link at 115200 baud. The host issues commands, and the USE SyncBox reports events that occur.

The USE SyncBox issues timing pulses at regular intervals; issues reward pulses as one-off events; writes parallel event code data to the ephys I/O port; and reports analog input from the joystick and light sensors at regular intervals.

In addition to analog light sensor values, the USE SyncBox also reports whether each sensor is above- or below-threshold. Light sensor thresholds are calibrated by measuring sensor values during a time window, taking the average of these measured values, and then saving the average as the new threshold (overwriting the old threshold). "Free-running" calibration does this repeatedly (starting a new window when the old window ends).

A list of available commands is shown in Figure 2.1. A timing diagram of user-commanded output events is shown in Figure 2.2. A diagram illustrating light sensor filtering is shown in Figure 2.3.

Sample output (including echoed commands) is shown in Figure 2.4. Format details for verbose, terse, and packed log entries are given in Figure 2.5. Typical configuration/status information is shown in Figure 2.6.

There are several important things to note about interactions with the USE SyncBox:

- The command parser does **not** recognize backspaces.

- **All times**, whether they're timestamps or duration arguments, are given in **"clock ticks", not milliseconds**. By default, clock ticks are 0.1 ms in length. One second is 10000 ticks.

  Milliseconds would not have been sufficiently precise, and microseconds would have overflowed a 32-bit counter in slightly more than one hour.

  Bear in mind that the maximum argument value is 65535 (a duration or interval of about 6.5 seconds).

```
Commands:
 ?, HLP  :  Help screen.
    QRY  :  Query system state.
  ECH 1/0:  Start/stop echoing typed characters back to the host.
    IDQ  :  Device identification string query.
    INI  :  Reinitialize (clock reset and events idled).
  LOG 1/0:  Start/stop reporting log data.
    LIN n:  Set the data reporting interval to n ticks.
LVB 2/1/0:  Set data report verbosity (2 = full, 1 = terse, 0 = packed hex).
    TPW n:  Set the timing pulse duration to n ticks.
    TPP n:  Set the timing pulse period to n ticks.
  TIM 1/0:  Enable/disable timing pulses.
    RWD n:  Send a reward pulse lasting n ticks.
    NSU n:  Set the event code pre-strobe setup time to n ticks.
    NHD n:  Set the event code post-strobe hold time to n ticks.
    NPD n:  Set the event code strobe pulse duration to n ticks.
    NEU n:  Emit value n over the event code parallel interface.
    NDW n:  Set event code data width to n bits (8 or 16).
CSL/R/I  :  Light sensor calibration slaved to left, right, or independent.
  CAO/F n:  Calibrate light sensors over n-tick window (one-shot/free-running).
  CTR/L n:  Force left/right light sensor threshold value to n.
    FIL n:  Set analog noise filter window to 2^n samples (0 disables).
    FST n:  Set analog spike rejection threshold to 2^n times the variance.
    FSW n:  Set spike rejection variance window to 2^n samples (0 disables).

Debugging commands:
  XAL 1/0:  Start/stop streaming full-rate analog light sample data.
  XJL 1/0:  Swaps or un-swaps joystick XY and light sensor LR inputs.
  XTR 1/0:  Enable/disable reward triggered by rising edge of GP00.
  XTT 1/0:  Enable/disable timing active on high level on GP00.
  XTN 1/0:  Enable/disable event code triggered by rising edge of GP00.
    XMA n:  Emit a rising edge on GP01 when analog 0..4 (X/Y/Z/L/R) rises.
    XMD  :  Disable GP01 analog diagnostic output.
```
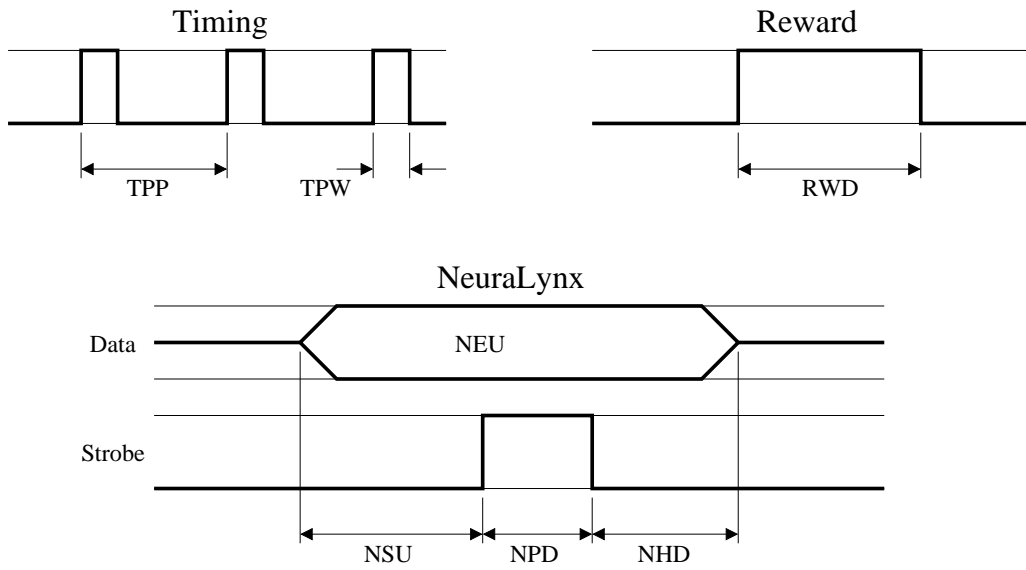
Figure 2.1: Command summary.
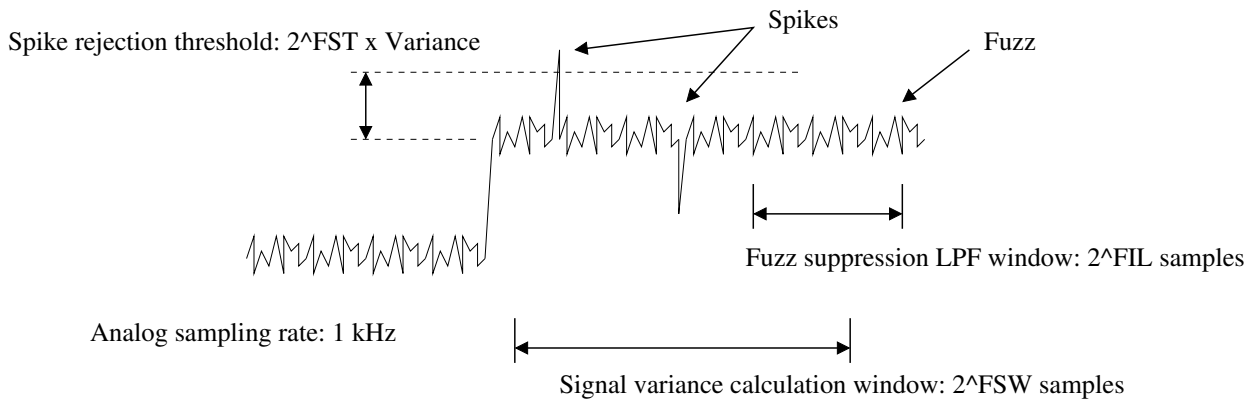
Figure 2.2: Output signal timing diagram.

Figure 2.3: Light sensor filtering.

- Number values entered by the user are given in **decimal**, but number values reported from the USE SyncBox are in **hexadecimal** (with the exception of status information).

  Processing user information in decimal made the command parser simpler (and makes life easier for manually entering data). Reporting information in hexadecimal makes output more compact (important for high data rate output), and avoids time-consuming data conversion (converting to hexadecimal does not require divisions, only bit-shifts; converting to decimal does require divisions). The data rate is high enough that this matters.

The data reporting rate is limited by the serial connection. If events occur too fast to report (or if full-rate analog debug reporting exceeds the serial connection's ability to report), reports will be dropped. Events themselves will still occur.

At 115200 baud, the maximum reporting rate without packet loss is 150 Hz with verbose logging, 200 Hz with terse logging, and 300 Hz with packed hex logging.

```
log 1
Time: 0005764f  Joy (x/y/c): 0c 00 00  Opt (l/r): 76 75 BLK BLK
Time: 00057a37  Joy (x/y/c): 0f 00 00  Opt (l/r): 76 76 BLK BLK
Time: 00057e1f  Joy (x/y/c): 0e 00 00  Opt (l/r): 76 76 BLK BLK
lvb 1
T: 000631d0  XYC: 0d 00 00  LR: 76 76 B B
T: 000635b8  XYC: 0f 00 00  LR: 76 76 B B
T: 0006399f  XYC: 0e 00 00  LR: 76 76 B B
lvb 0
T0006ddb0J0f0000L7676BB
T0006e198J0e0000L7676BB
T0006e57fJ0e0000L7676BB
lvb 2
Time: 000785a7  Joy (x/y/c): 0c 00 00  Opt (l/r): 76 76 BLK BLK
Time: 0007898f  Joy (x/y/c): 0d 00 00  Opt (l/r): 76 76 BLK BLK
Time: 00078d77  Joy (x/y/c): 0f 00 00  Opt (l/r): 76 76 BLK BLK
log 0

tim 1
Synch: 00087e8e
Synch: 0008a59e
Synch: 0008ccae
tim 0

rwd 1000
Reward: 000a3433 03e8
rwd 1000
Reward: 000aa549 03e8
rwd 500
Reward: 000ba7ee 01f4

xal 1
99387676
99457675
99527675
995e7675
xal 0

neu 12345
Code: 000f2e19 3039
neu 32767
Code: 00101097 7fff
```

Figure 2.4: Sample USE SyncBox output.

Verbose:

`Time:` `0003f6ef` `Joy (x/y/c):` `062a` `063f` `0647` `Opt (l/r):` `0610` `061b` `BLK` `BLK`

Terse:

`T:` `0004d853` `XYC:` `055f` `058f` `05b3` `LR:` `05b0` `056c` `B` `B`

Packed:

`T` `00054937` `J` `05b8` `05c2` `05c5` `L` `056f` `05cb` `BB`

Figure 2.5: Log entry format details.

```
devicetype: USE SyncBox  subtype: v1  revision: 20181210

System state (all values in base 10):
  Version:  20181210
  Timestamp:  88602 ticks
  Clock ticks per second:  10000
  Reward I/O:  D11/B5/GP05
  Timing I/O:  D12/B6/GP06
  Debug trigger input:  D6/H3/GP00
  Debug monitor output:  D7/H4/GP01
  Logging:  OFF
  Log interval:  100 ticks
  Verbosity:  Full
  Timing:  OFF
  Timing pulse width/period:  10 / 10000 ticks
  Event code data front-porch/strobe/back-porch:  2 / 2 / 2 ticks
  Event code data width:  16 bits
  Light sensor thresholds (left/right):  65535 / 65535
  Light threshold slaving:  INDEPENDENT
  Analog sample size:   8 bits
  Noise suppression filter:  2 bits  (decay length 4 samples)
  Spike suppression filter:
  Variance LPF:  6 bits  (decay length 64 samples)
  Spike rejection threshold:  2^2 (4x)
  External event triggers:  none
  Analog input being monitored:  none
  Joystick analog threshold:  127
End of system state.
```
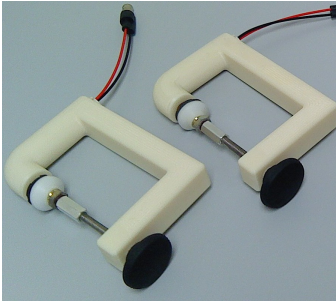
Figure 2.6: USE SyncBox identification string and status report.

Chapter 3

Accessories

Accessories produced for the USE SyncBox include the following:

Light sensor clamps may be attached to a wide variety of monitors (including notebook computers). Patches of the display are strobed, and this flickering is recorded in order to measure when frame redraws occur.
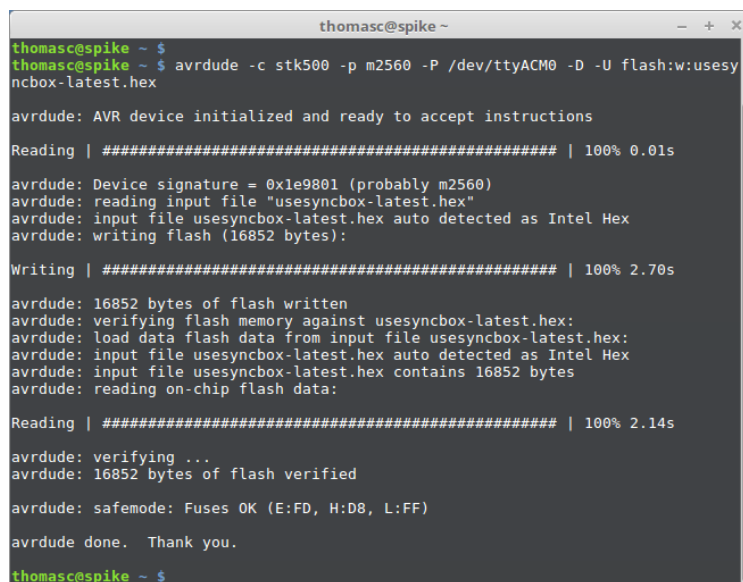
Chapter 4

Maintenance

From time to time, new versions of the USE SyncBox firmware are released. These can be flashed using the "`avrdude`" utility, which is bundled with the Arduino development environment or may be downloaded on its own.

To use "`avrdude`", plug in the USE SyncBox, determine what the name of the Arduino's USB serial device is, bring up a "command terminal" window, navigate to the directory with the firmware file, and type:

```
avrdude -c stk500 -p m2560 -P (device) -D -U flash:w:(firmware file)
```

For a firmware file named "`usesyncbox-latest.hex`", and a USB serial device called "`/dev/ttyACM0`", the result will be similar to the following:

```
thomasc@spike ~
thomasc@spike ~ $
thomasc@spike ~ $ avrdude -c stk500 -p m2560 -P /dev/ttyACM0 -D -U flash:w:usesy
ncbox-latest.hex

avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.01s

avrdude: Device signature = 0x1e9801 (probably m2560)
avrdude: reading input file "usesyncbox-latest.hex"
avrdude: input file usesyncbox-latest.hex auto detected as Intel Hex
avrdude: writing flash (16852 bytes):

Writing | ################################################## | 100% 2.70s

avrdude: 16852 bytes of flash written
avrdude: verifying flash memory against usesyncbox-latest.hex:
avrdude: load data flash data from input file usesyncbox-latest.hex:
avrdude: input file usesyncbox-latest.hex auto detected as Intel Hex
avrdude: input file usesyncbox-latest.hex contains 16852 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 2.14s

avrdude: verifying ...
avrdude: 16852 bytes of flash verified

avrdude: safemode: Fuses OK (E:FD, H:D8, L:FF)

avrdude done.  Thank you.

thomasc@spike ~ $
```

The names of USB serial devices vary widely from operating system to operating system and version to version. Consult appropriate documentation to determine how to find the USE SyncBox's USB serial device on your operating system.

10