A Teensy microcontroller-based interface for wide-field optical imaging and behavioral experiments

Michael Romano, Mark Bucklin, Dev Mehrotra, Robb Kessel, Howard Gritton, Xue Han

**Abstract**

Systems neuroscience experiments often require the integration of precisely timed data acquisition and behavioral monitoring. While specialized commercial systems have been designed to meet various needs of data acquisition and device control, they often fail to offer flexibility to interface with new instruments and variable behavioral experimental designs. For example, it has been difficult to integrate recently developed sCMOS cameras with various input and output devices for high speed, large scale calcium imaging experiments during behavior. We here developed a Teensy (version 3.2) microcontroller-based interface that offers high-speed, precisely timed behavioral data acquisition, and digital and analog outputs for controlling sCMOS cameras and other devices. We demonstrate the efficacy and the temporal precision of the Teensy interface in two experimental settings. In one example, we used the Teensy interface for reliable recordings of an animal’s directional movement on a spherical treadmill, while delivering repeated digital pulses that can be used to control image acquisition from a sCMOS camera. In another example, we used the Teensy interface to control temporally precise delivery of auditory and visual stimuli in a trace conditioning behavioral paradigm, while delivering repeated digital pulses that can be used to control image acquisition from a sCMOS camera. These examples demonstrate that the Teensy interface, developed here consisting of a Teensy 3.2 and custom software functions, provides a low-cost and flexible platform to integrate a sCMOS camera into behavioral experiments, allowing high-speed and temporally precise imaging analysis during behavior.

**Introduction**

Recent advances in sCMOS camera technology and genetically encoded calcium sensors enable large scale fluorescence imaging of thousands of individual cells’ activity during behavior (Klaus, et al., 2017; Barbera, et al., 2016; Mohammed, et al., 2016; Markowitz, et al., 2018). One key technical aspect of neural network analysis during behavior is the temporal precision, where neural activities need to be precisely aligned with behavioral features (Solari, Sviatkó, Laszlovsky, Hegedüs, & Hangya, 2018). However, it has been difficult to integrate sCMOS cameras, deployed in large scale calcium imaging studies, with devices needed to monitor and control behavioral experiments. Traditional Analog/Digital interfaces are often operated by programs, such as MATLAB, that offer a wide range of applications. However, using MATLAB or other PC-based programs can lead to undesired temporal delays if they rely on the PC for timing, as the operating system needs to balance the demands of many system operations at once.

Over the last decade, microcontrollers marketed to hobbyists have gained popularity across a variety of scientific fields (Sanders & Kepecs, 2014; D'Ausilio, 2012; Chen & Li, 2017; Husain, Hadad, & Zainal Alam, 2016). Microcontrollers are small, low-cost, and capable of delivering digital outputs with microsecond time precision, while using user-friendly, open-source software functions. The Arduino was the first major microcontroller to gain substantial popularity. Recently, Teensy microcontrollers were developed, which have all the key features of Arduino microcontrollers, as well as the additional feature of delivering analog output. Teensy’s utilize the same open-source Arduino software environments, and thus they are easy to program (D'Ausilio, 2012). Microcontrollers therefore represent an attractive solution for systems neuroscience that can be easily adapted to various behavioral experimental needs, including the integration of newly developed instruments.

For example, Arduino devices have recently been integrated into two-photon imaging experiments (Wilms & Häusser, 2015; Takahashi, Oertner, Hegemann, & Larkum, 2016). In one study, an Arduino was used to generate a digital command to encode the duration of an image sequence (Micallef, Takahashi, Larkum, & Palmer, 2017). One other way to precisely time sCMOS image acquisition with behavior is by using a microcontroller interface to trigger each frame independently and simultaneously to acquire behavioral data. To do this, one can use the external trigger setting of the camera, which has been mentioned previously (Micallef, Takahashi, Larkum, & Palmer, 2017). Monitoring both aspects of the experiment with a single device improves one’s ability to accurately and precisely map imaging data onto behavioral data.

Here, we demonstrate and characterize a flexible Teensy interface for temporally precise data acquisition and delivery of analog and digital signals, in two experimental paradigms, during voluntary movement and during a trace conditioning experiment. The Teensy interface can deliver digital pulses with microsecond precision to initiate individual image frame capture at a desired frequency, while simultaneously collecting animal behavioral data. We also demonstrate the ability of the Teensy interface to generate analog high frequency sound waveforms simultaneously with other types of digital output. Together, these results demonstrate that the Teensy interface, consisting of a Teensy microcontroller equipped with a set of custom software functions, offers a flexible, accurate, and user friendly environment for imaging experiments during behavior, along with the option of analog output.

**Methods**

*General overview of construction of Teensy boards*

The two experimental designs used are shown in Figure 1, and the specialty components required to build these designs are shown in Tables 1 and 2. In both experiments, a Teensy 3.2 (PJRC.COM, LLC, part #: TEENSY32) (Figure 1A), or a Teensy 3.2 soldered to a prop shield (PJRC.COM, part #: PROP\_SHIELD) (Figure 1B), is mounted on top of a printed circuit board via standard female headers (such as SparkFun Electronics, PRT-00115). Female headers were then soldered to the PCB for stability. Output from the Teensy was directed from pins on the female headers to standard SMA connectors (such as: Digi-Key, part # CON-SMA-EDGE-S-ND) via 22 gauge wires (for example: Digi-Key, part #1528-1743-ND). Coaxial cables were then attached the SMA connectors to connect the Teensy to external devices. The Teensy was connected to a computer via a standard USB-microUSB cable (for example: Digi-Key, part # AE11229-ND). To easily upload our code to the Teensy, we used PlatformIO (<https://platformio.org/>), an add-on to the widely-used Atom text editor (<https://atom.io/>), instead of the default Arduino programming environment. In our motor acquisition experiment, to easily set the sampling frequency and length of an experiment for the Teensy, we developed a simple MATLAB graphical user interface. We used a similar MATLAB graphical user interface to set the number of trials and length of trials in our trace eye blink conditioning experiment. Throughout both experiments, to turn digital pins on and off, and also to change their modes to either “input” or “output”, we used the DigitalIO library (<https://github.com/greiman/DigitalIO>), which decreases the amount of time spent performing each of these actions.

*Motor acquisition experiment*

In this experiment, we performed motion tracking using two ADNS-9800 gaming sensors (<https://www.tindie.com/products/jkicklighter/adns-9800-laser-motion-sensor/>, Tindie, part: “[NS-9800 Laser Motion Sensor](https://www.tindie.com/products/jkicklighter/adns-9800-laser-motion-sensor/)”, see Table 1), while delivering digital pulses that can be used to trigger a sCMOS camera for image capture every 50 ms. The overall design of this experiment is shown in Figure 1A. A mouse was positioned on top of a buoyant Styrofoam ball floated by house air as described previously (Dombeck, Khabbaz, Collman, Adelman, & Tank, 2007). Two ADNS-9800 gaming sensors were positioned at the equator of the sphere, at an angle of approximately 75 degrees from one another. For the counts per inch setting of the sensor, which determines the sensitivity of the sensors to external movement, we used a value of 3400 counts per inch. Thus, the total distance travelled at any time point was computed using the following equation:

Where yR and yL are the y readings from the left and right sensors, and is the angle between the two sensors (75 degrees). We also acquired readings in the “x” direction from both sensors. Velocity was computed as the distance divided by the time between two subsequent readings. These two sensors were connected to a Teensy via simple serial peripheral interface (SPI) connections with insulated 22 gauge wires as shown in Figure 2A.

To control experimental timing with high precision, we utilized the “IntervalTimer” function unique to the standard Teensy library, which repeatedly calls a function at specified intervals, to coordinate both motor acquisition and digital pulse timing. Using IntervalTimer, we repeatedly called a function that reported to the attached PC the accumulated displacement of the motion sensor readings since the previous function call. Readings from motion sensors were extracted with freely available functions on Github (<https://github.com/markbucklin/NavigationSensor>). These functions read motion data from the “motion burst” register of each sensor, from which we acquired the x and y displacement readings from each sensor. At the end of this function, a digital “on” pulse that lasted for 1 ms was sent out of a digital pin directed at a sCMOS camera. These pulses can be used as an external trigger to initiate frame capture. We set the interval between calls to this function to 50,000 microseconds (50 ms) or 20 Hz in our experiment.

To characterize the temporal precision, of the respective digital pulses, we recorded the digital outputs with a commercial system (Tucker Davis Technologies RZ5D (TDT RZ5D)) at 3051.76 Hz.

*Trace eye blink conditioning experiment*

In this experiment, a Teensy was programmed to deliver outputs capable of eliciting a sound, turning on an LED light and initiating an eye puff, while delivering digital pulses that can be used to trigger a sCMOS camera to capture an image every 50 ms. An example schematic of the complete experimental design is shown in Figure 1B. To deliver an audible sound through the Teensy, we used a prop shield module available for Teensy (PJRC.COM, LLC., part #: PROP\_SHIELD). This add-on component amplifies analog output to drive speakers with resistances up to 8 ohms (shown in Figure 2B as pin A14). The prop shield was soldered to the bottom of a Teensy with 14x1 double insulator pins (PJRC.COM, LLC., part #: HEADER\_14x1\_D), and the output was connected to a speaker. The Teensy was then mounted onto the female headers separated by the prop shield, as shown in Figure 1B. In the current experiment, the speaker, camera, and air valve for the eye puff were then attached to the microcontroller through coaxial cables as described above (also shown in Figure 1B).

In order to periodically elicit a tone, we initialized a 9500 Hz sine wave at the beginning of each experiment using the Teensy Audio library function “AudioSynthWaveformSine,” and originally set the amplitude to “0”, so that the tone was off. This function continuously outputs a sine wave with a sampling rate of 44.1 kHz from the analog pin. In order to toggle the tone “on” or “off” during the experiment, we switched the amplitude to 0.05 or 0 (out of a maximum of 1), respectively. A value of 0.05 in combination with the amplifier and our speaker generated a tone of approximately 75 dB.

After the tone was initialized in the code, a single function was called every 50 ms. We used the “elapsedMicros” function to time the frequency of this function. This function updated the status of the digital pins controlling the air valve for the “puff” stimulus and the LED light stimulus, and updated the amplitude of the sine wave. Immediately following these updates and within the same function, a brief, 1 ms digital pulse capable of eliciting frame capture was delivered to a sCMOS camera. At the completion of each trial, this single function also initiated the following trial or signaled to terminate the experiment.

While image capture was not performed in this experiment, output from the respective digital pin was monitored via an external device to characterize temporal precision. Puff and speaker outputs were monitored audibly to ensure proper functioning and via an external device as well. We recorded the digital outputs with a commercial system (TDT RZ5D) at 3051.76 Hz, and the analog output at 24414.0625 Hz without additional amplification.

*Statistics*

Linear models were constructed using the “fitlm” function in MATLAB 2017b. Theoretical timings, to which measured timings were compared, were each taken to be timings beginning at 0 seconds in equal increments of 50 milliseconds for both experiments.

**Results**

Microcontrollers such as Arduinos have gained popularity in neuroscience research due to their user-friendly interface, open-source nature, and their flexibility of device integration (D'Ausilio, 2012; Chen & Li, 2017; Micallef, Takahashi, Larkum, & Palmer, 2017). Recently, the Teensy 3.2 has been developed, which has analog output, a major improvement over the popular Arduinos. Teensy also has a comprehensive Audio library, as well as the IntervalTimer function, which can facilitate precisely timed, repeated events. Here, we present a Teensy-based interface to integrate on a frame-by-frame basis sCMOS camera image capture with behavioral experimental control and data acquisition.

*Mice Motion tracking experiment*

To measure locomotion from awake head fixed mice, we implemented a novel design wherein we used the Teensy interface to record from two ADNS-9800 motion sensors (Figures 1A and 2A). These sensors were affixed to a “spherical treadmill” setup, as described by Dombeck, Khabbaz, Collman, Adelman, & Tank (2007).

ADNS-9800 sensor boards are low cost, and are more sensitive than regular computer mice. They can measure up to 8200 counts per inch, allowing for more precise measurement of mouse movement than a normal computer mouse. For example, a standard computer mouse, the Logitech M100 (Logitech, PN: 910-001601), can only read up to 1000 counts per inch, making the ADNS-9800 sensor over 8 times more precise at its highest setting. Further, wiring ADNS-9800 sensors to the Teensy is simple (Figure 2A). The connections are demonstrated using dotted lines in Figure 2A, and can be replaced with jumper wires or sturdier, longer lasting wire, as detailed in the *Methods*.

In this example experiment (Figure 3A), we recorded a 10 minute long session of a mouse running on the spherical treadmill. Motion data was acquired at 20 Hz concomitantly with digital outputs that can be used to trigger individual image frame capture from a sCMOS camera. We calculated the velocity of the mouse, which averaged 7.1 + 6.9 cm/s (mean + std) with a maximum velocity of 47.0 cm/s , which is in general agreement with previous studies (Dombeck, Khabbaz, Collman, Adelman, & Tank, 2007)). We measured the timing of the Teensy digital output in order to assess its precision and accuracy, and constructed a linear model comparing the measured timings to theoretical timings at exactly 20 Hz. We found that digital outputs have a near-perfect linear relationship with the theoretical timings, with a small, 28.9 µs per second positive drift (Figure 3B). Thus, the actual frequency was 19.999 Hz instead of 20 Hz.

To examine whether the small timing drift varied with the frequency of data acquisition or the timing of the digital outputs, we performed 5 minute long recording sessions without a live mouse at 20, 50, and 100 Hz. These recordings used an identical script, except due to the shorter frame duration we used a 500 microsecond long digital pulse for camera triggering instead of a 1 millisecond pulse. We found that the actual frequencies were 19.999, 49.999, and 99.997 Hz, respectively. These all equate to an approximately 30 µs delay per second, suggesting that the timing drift is independent of the data acquisition rate.

Having assessed the accuracy of the camera trigger timing, we next wanted to assess its precision. To do this, we looked at the root mean squared error (RMSE) of the model fit, which represents the square root of the average squared difference between measured values and values predicted by a linear model. Thus, it is similar to the standard deviation of the residual values of this model. The linear model fit demonstrated a root mean squared error of 38.9 microseconds, indicating that the camera trigger has at least microsecond-level precision. Together, these results demonstrate that a Teensy can be used to trigger precise and accurate image frame capture during long behavioral experiments while maintaining alignment of imaging data with behavioral parameters.

*Trace eye blink conditioning behavioral experiment*

In the second setting (Figure 1B and 2B), we utilized the Teensy interface in a trace conditioning learning experimental design. In this experiment, a mouse can be trained to associate conditioned stimuli (700ms long tone and/or light stimuli) with a subsequent unconditioned stimulus (a 100ms long eye puff) separated by a brief time window (250ms). In each session, 50 trials were performed with each trial lasted 20 seconds long.

In order to assess the accuracy and precision of the camera-directed digital pulses in this setting, we recorded the timings of each camera pulse and compared them to the theoretical rate of 20 Hz using a linear model (Figure 4A). Similar our observation in the locomotion experimental design, the measured timings exhibit a perfect linear relationship with the theoretical timings, with a 33.4 microsecond delay per sample. Therefore, although this Teensy must control three other types of output in this particular experimental setting, it has similar accuracy to the Teensy used in the other experimental setting. The root mean squared error for the model fit was 13.3 us, consistent with the motor experiment in its microsecond-level precision.

We next wanted to characterize the ability of the Teensy to deliver an analog output in the form of a tone simultaneously with repeated camera-directed digital pulses. To do this, we first determined the length of time between when the tone was signaled to turn on and when an external device (TDT RZ5D) measured output from the analog pin. Changes in tone state were synchronized with the beginnings of camera-directed pulses, so we used the timings of the camera-directed pulses to benchmark the delay in tone onset. Latency was defined the time delay between the onset of the analog output and the respective camera-directed digital pulse, both as measured by an external device. (Figure 4Bi). In order to measure the onset of the audio signal, we took the raw analog recording and high-pass filtered the signal using a 6th-order Butterworth filter, a bandpass frequency of 1000 Hz, and a “zero-phase digital filter” (MATLAB command “filtfilt”). Then, we took the absolute value of the Hilbert transform of the filtered signal to acquire an amplitude envelope. After finding the amplitude envelope, we found those values that exceeded a value of 0.005. The first time point that the amplitude crossed this threshold was considered the tone onset, and the next time point that dropped below this threshold was considered the tone termination.

The measured tone latency was precise; it averaged 7.6 + 0.9 milliseconds (Figure 4Bi). Because of this precision in latency, it would be easy to align this output with the onset of a camera frame if so desired. We further characterized the precision and accuracy of the tone component of our platform by measuring the length of the tone (Figure 4Bii). It lasted for 700 + 1 ms, with a range of 2.9 ms, which is not different from the programmed tone length (700 ms). Other implementations of the Audio library could potentially offer even more precision and accuracy. For example, if one needed to utilize a precise sound sequence in an experiment, they could upload the sound sequence as a .wav file and utilize the Teensy to play the pre-recorded sound (Solari, Sviatkó, Laszlovsky, Hegedüs, & Hangya, 2018). However, our design can be implemented very simply, utilizing only a few lines of code within a single script.

We next wanted to characterize the precision and accuracy of our platform’s longer duration digital output, the puff, by quantifying puff latency and puff duration. The latency of the puff was measured analogously to the latency of the sound. Each puff start and termination were aligned with the onset of a camera-directed digital pulse. Therefore, we defined the latency to be the delay between the onset of the digital pulse from the puff pin and the onset of the simultaneously initiated camera-directed digital pulse. As shown in Figure 4Biii, the digital output for the eye puff showed no delay from the corresponding camera trigger (mean latency = -0.004 + 0.012 (+ std) ms, range=0.04 ms seconds). The duration of the puff digital pulse was also highly accurate and precise, as shown in Figure 4Biv, and lasted 100.03+0.02 (+ std) ms over the course of the 50 trials, hardly differing from the expected duration of precisely 100 ms. Therefore, the Teensy interface is precise and accurate in each of these three experimental capacities: tone generation, puff output, and simultaneous high-frequency digital pulse generation for sCMOS camera image capture.

**Conclusion and Discussion**

We demonstrate the accuracy and precision of a Teensy 3.2 interface in integrating sCMOS camera image capture with behavioral parameter reading and behavioral output via two different experimental settings. In one setting, the Teensy interface utilizes recently developed ADNS-9800 gaming sensors. These sensors provide precise and high speed locomotion tracking while the Teensy interface simultaneously issues digital pulses capable of eliciting image frame capture. The ease with which these sensors were utilized also illustrates the flexibility of the Teensy in designing experiments that require novel instrumentation. In a second experiment, we demonstrate that the Teensy interface, in conjunction with a prop shield, is capable of controlling three devices with precise timing during a trace conditioning experiment, with an option of including a fourth (an LED light, not utilized here). This Teensy interface can be immediately be adopted for the designed locomotion and trace conditioning behavioral experiments, or customized for other types of behavioral experiments, where sCMOS camera-based imaging is desired. These implementations are additionally low-cost, open-source, and can be easily customized for various types of behavioral experiments.

In both experiments, the timings of camera-directed digital pulses sent by the Teensy interface were precise, as measured by the root mean square error of the model fits, to the level of microseconds, and were accurate to within approximately 30 µs per second. This small 0.003% drift of the Teensy processing clock is linear and can thus be calibrated if desired. This finding additionally underscores the necessity of having a highly precise central device for experimental control. Synchronizing different devices only with a single signal at the start of an experiment can lead to problems if these devices have different temporal drifts, particularly if the experiment is long in duration. Alternatively, initiating or measuring experimental events on a frame-by-frame basis using a PC for timing can introduce timing jitter due to the multitude of tasks that a PC must attend to at any given point in time. This jitter can have a significant impact depending on the study. For example, a recent calcium imaging study in the striatum finds additional neurological structure related to motor activity on very short timescales, but finds only velocity correlated with neural activity on longer timescales (Markowitz, et al., 2018). This suggests that with sufficient timing jitter, correlations and information on short time scales could be missed, yielding erroneous conclusions.

Temporal accuracy is not only important for behavioral data acquisition. In our trace conditioning experiment, for example, precisely timed stimuli are desired as well. Therefore, in this experimental setting, we characterized the accuracy and precision of digital and analog outputs, which the Teensy interface had to balance with repeated digital pulses directed at a sCMOS camera. First, we show that our Teensy interface accurately and precisely delivers a 9500 Hz tone using the built-in Audio library. This Audio library can also be used for sound synthesis, reading, and mixing, all at 44.1 kHz, which is stereo quality. This provides a tool for many experimental conditions, especially those needing sound as a stimulus. Secondly, we show that our Teensy interface accurately and precisely delivers a longer digital pulse that can drive “puffs” while simultaneously producing camera-directed digital pulses. Ultimately, the precisions of both our puff and sound output are comparable to expensive, available systems such as the Habitest Modular system in conjunction with Coulbourn Graphic State 4 software, which itself offers 1 ms precision (<http://www.coulbourn.com/v/vspfiles/assets/manuals/Graphic%20State%204%20Users%20Manual.pdf>) , making the Teensy a viable, inexpensive alternative that is also able to simultaneously capture imaging data using our simple software design.

A major advantage of the Teensy over other microcontrollers is its ability to generate a true, 12 bit analog signal, which has been, to this point, illustrated only insofar as it can be used to produce a sound. While Arduino microcontrollers can generate an analog-like signal, they need extra devices such as resistors and capacitors in order to do so. This makes any task that requires some type of analog output easier to implement with a Teensy, which makes it highly flexible for diverse experimental designs than benefit from any type of high resolution analog output. In sum, the precision and flexibility of our Teensy 3.2 microcontroller interface makes this a user-friendly, easily adaptable, accurate, and precise tool for use with simultaneous image capture, behavioral control and data acquisition.

**Figures**

**Figure 1.** Diagrams of the two complete experimental device setups using a Teensy interface. **A** A floating, spherical treadmill setup for locomotion recording. This experimental design consists of a Teensy 3.2 connected to two ADNS-9800 sensors via serial-peripheral interfaces, and a CMOS camera through a coaxial cable. Every 50 milliseconds, a digital pulse was sent to the CMOS camera to elicit an image frame capture. Simultaneously, the Teensy acquired motion data from both ADNS sensors and sent them to a PC via a USB. The PC initiates each experiment by sending serial data consisting of the length of the experiment and the digital output frequency to the Teensy. **B** A trace eye-blink conditioning setup. In order to generate a sound through the speaker, the Teensy is soldered to a prop-shield, which contains an amplifier. The length and the number of experimental trials were specified in MATLAB. The Teensy 3.2 sends time stamps, trial, and stimulus information (audio, LED, and puff pin states) via the USB back to the PC. In each trial, the Teensy generates a 9500 Hz tone at a sampling rate of 44.1 kHz. The tone stimuli, delivered optionally along with turning on an LED light (not implemented in our example experiment), is followed by an eye puff.

**Figure 2.** Detailed electrical wiring schematics for both the locomotion recording and trace eye-blink conditioning experiments **A.** A schematic demonstrating the wiring of a Teensy 3.2 to two ADNS-9800 sensors via serial peripheral interface connections (SPIs). Solid dots at intersections between lines indicate connections. Some unused pins on the Teensy 3.2 were not included in this schematic. **B** A schematic demonstrating the electrical connections between a Teensy 3.2, a prop shield, and an external speaker. Dotted lines indicate connections. All connections between the Teensy 3.2 and prop shield were made using 14x1 double insulated pins, and the output to the speaker from the prop shield was constructed using 22 gauge wire and a coaxial cable. Some extraneous and unused pins on the Teensy and the prop shield were not included in this diagram.

**Figure 3.** Example recording using the locomotion recording experimental design **A** Part of a sample 10 minute recording session during which a head-fixed mouse was allowed to run on the three-dimensional treadmill shown in Figure 1A. **B** Times of digital pulses sent by the Teensy as measured by an external device, vs theoretical times of the digital pulses at exactly 20 Hz. Red indicates linear model prediction, and in black are experimental data, down-sampled by a factor of 200. The linear model estimates a slope of 1.000028927 + 0.000000005 (t(11997)= 2.0381e+08, p < 0.001, R2=1; intercept = 0.000107 + 0.000002, t(11997) = 63.243, p < 0.001), indicating an excellent fit and very nearly a 1:1 correspondence of time stamps.

**Figure 4.** Example recording using the trace conditioning tone-puff setup **A** Timing of the sCMOS camera-directed digital pulses as measured by an external device vs theoretical timing assuming a frequency of precisely 20 Hz. Linear model fit is shown in red, and in black are experimental data down-sampled by a factor of 200 for visualization. These measurements have a correspondence near 1:1 (R2=1, slope: 1.0000334 + 0 (to machine precision), t(14998)=infinite, p<0.001). **B.** Timing of the analog output corresponding to tone delivery (i-ii) and digital output corresponding to puff delivery (iii-iv), both measured by an external device over the course of fifty trials; (i) shows the latency between the programmed onset of the tone and the timing of the tone onset as measured by the TDT device (mean=7.6 + 0.9 ms, range=2.9 ms); (ii) shows the accuracy and precision of the length of tone intervals across all trials (mean=700 + 1 ms, range=2.9 ms); (iii) shows the latency of the puff digital pulse, as measured by the TDT system (mean= -0.004 + 0.012 ms, range=0.04 ms); (iv) shows the accuracy and precision of the length of the puff digital pulse across all trials (mean = 100.03+0.02 ms). (all + std).

**Tables**

Table 1. Specialty components necessary to build a motor output system

|  |  |  |  |
| --- | --- | --- | --- |
| **Part name** | **Website** | **Part number** | **Cost per unit** |
| Teensy 3.2 | https://www.pjrc.com/store/teensy32.html | TEENSY32 | $19.80 |
| ADNS-9800 sensors | https://www.tindie.com/products/jkicklighter/adns-9800-laser-motion-sensor/ | ADNS-9800 Laser Motion Sensor | $27.50 |

Table 2. Specialty components necessary to build a tone-puff system.

|  |  |  |  |
| --- | --- | --- | --- |
| **Part name** | **Website** | **Part number** | **Cost per unit** |
| Teensy 3.2 | https://www.pjrc.com/store/teensy32.html | TEENSY32 | $19.80 |
| 14x1 Double insulator pins | https://www.pjrc.com/store/header\_14x1\_d.html | HEADER\_14x1\_D | $0.85 |
| Prop shield | https://www.pjrc.com/store/prop\_shield.html | PROP\_SHIELD | $19.50 |

# References

Aranov, D., & Tank, D. W. (2014). Engagement of Neural Circuits Underlying 2D Spatial Navigation in a Rodent Virtual Reality System. *Neuron, 84*(2), 442-456.

Barbera, G., Liang, B., Zhang, L., Gerfen, C. R., Culurciello, E., Chen, R., . . . Lin, D.-T. (2016, October 5). Spatially Compact Neural Clusters in the Dorsal Striatum Encode Locomotion Relevant Information. *Neuron, 92*(1), 202-213.

Chen, X., & Li, H. (2017, December). ArControl: An Arduino-Based Comprehensive Behavioral Platform with Real-Time Performance. *Frontiers in Behavioral Neuroscience, 11*, 1-9.

D'Ausilio, A. (2012). Arduino: A Low-Cost Multipurpose Lab Equipment. *Behavior Research Methods, 44*(2), 305-313.

Dombeck, D., Khabbaz, A. N., Collman, F., Adelman, T. L., & Tank, D. W. (2007). Imaging Large-Scale Neural Activity with Cellular Resolution in Awake, Mobile Mice. *Neuron, 56*(1), 43-57.

Klaus, A., Martins, G. J., Paixao, V. B., Zhou, P., Paninski, L., & Costa, R. M. (2017). The Spatiotemporal Organization of the Striatum Encodes Action Space. *Neuron, 95*(5), 1171-1180.e7.

Markowitz, J. E., Gillis, W. F., Beron, C. C., Neufeld, S. Q., Robertson, K., Bhagat, N. D., . . . Datta, S. R. (2018). The Striatum Organizes 3D Behavior via Moment-to-Moment Action Selection. *Cell, 174*(1), 44-58.e17.

Micallef, A. H., Takahashi, N., Larkum, M. E., & Palmer, L. M. (2017, May). A Reward-Based Behavioral Platform to Measure Neural Activity during Head-Fixed Behavior. *Frontiers in Cellular Neuroscience*, 1-8.

Mohammed, A. I., Gritton, H. J., Tseng, H.-a., Bucklin, M. E., Yao, Z., & Han, X. (2016). An Integrative Approach for Analyzing Hundreds of Neurons in Task Performing Mice Using Wide-Field Calcium Imaging. *Scientific Reports, 6*, 20986.

Solari, N., Sviatkó, K., Laszlovsky, T., Hegedüs, P., & Hangya, B. (2018). Open Source Tools for Temporally Controlled Rodent Behavior Suitable for Electrophysiology and Optogenetic Manipulations. *Frontiers in Systems Neuroscience, 12*(May).

Yoav, A., Kim, J. J., Brinks, D., Lou, S., Wu, H., Mostajo-Radji, M. A., . . . Cohen, A. E. (2018). All-Optical Electrophysiology Reveals Brain-State Dependent Changes in Hippocampal Subthreshold Dynamics and Excitability. *bioRxiv*. doi:https://doi.org/10.1101/281618