Flyback Opto Stim: 10/01/2018

Chris Deister
cdeister@brown.edu

Problem:
When using photomultiplier tubes (PMTs) to detect fluorescence, as is done with standard two-photon laser scanning microscopy (2P), it is unwise and/or scientifically problematic to turn on light sources that emit in the range the detectors are tuned to. The specific reason for this technical note is to address and document a way to minimize light contamination of 2P imaging data from blue, red, green and orange LEDs used for many optogenetic experiments. Jakob Voigts and I used this in the course of experiments we did for his graduate work on the functions of layer 6 neurons. We got the idea from Winfred Denk's retina papers, where his group has needed to activate the retina with UV light while also conducting 2P imaging.

Approach:
What the Denk group does is to use the short period of time in between scanned lines where the y-galvonometer moves the 2P excitation beam to its next line. This is often referred to as the "flyback." The flyback is short at ~10-50 microseconds, but it occurs at relatively high rates. For resonant scanning, the repetition rate between lines (including the flyback itself) is 64 microseconds. For galvo-galvo scanning, with typical dwell times, the time between lines is slower at about 600-800 microseconds, again including the flyback itself. In either case, the repetition rate of flyback is fast enough for us to use the time between flybacks as a clock with which we can time the optogenetic stimuli. However, the "trick" relies on more than just the flyback occurrences as a clock. Nevertheless, that concept is the baseline, and the "sample rate" our flyback clock affords us is 1-15KS/s. Thus, factoring in Nyquist, we should be able to generate pulse trains with frequencies in the range of 500 Hz - 7 KHz — plenty for most optogenetic experiments that typically use 1-50 Hz.

Sampling with end of line triggers
As mentioned just using the onset of a flyback as our clock isn't the whole story. Because the flyback time is 10-50 microseconds, we are afforded enough time to gate a light source, assuming we can turn it on and off on the timescale of 10 microseconds or so. Some diode lasers can be modulated on the KHz timescales, which may be sufficient in some cases, but in general these lasers are expensive. An acousto-optic modulator can conceptually gate even a poor laser at sufficient speeds. But, solid-state LEDs are ideal. The LEDs used in most labs for optogenetic stimuli are driven by current source driver circuits that are poor overall, and impossible to use for our purposes. But, the LED itself can be gated at arbitrarily high rates. This is not the place to complain about "buck-type" LED drivers, and others can do that better. But, most fiber-coupled LEDs operate when driven by 300-1000 mA of constant current. Buck drivers can provide a constant current source, but they have a variety of capacitors and coils in them which ultimately lead to a slight exponential rise and decay to their steady-emission state when turned on. The time constant of this rise is hundreds of microseconds, thus finite predictable light pulses of 10 microsecond duration are hard (impossible) to achieve with buck drivers. You can just gate the LEDs with beefy mosfets, but then it becomes tricky to change current levels on the fly (dimming). I recommend Jon Newman's amazing "Cyclops" LED driver, which is an open source driver that provides all the upsides of mosfets, but is purpose built for optogenetic experiments and fluorescence imaging.

https://github.com/jonnew/cyclops
https://www.open-ephys.org/store/cyclops-led-driver

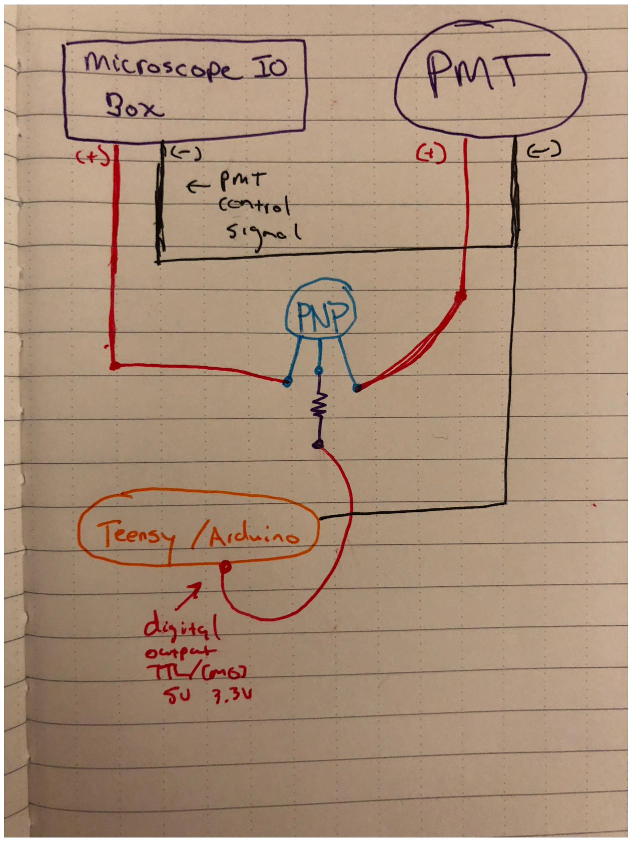
Use of mosfets or the Cyclops allows us to assume that even if we trigger the driver for 10 microseconds it will turn on, emit, and turn off all within that period of time. Thus, if you have a 20 microsecond flyback, as long as we can take a signal signifying the end of a line and turn it into a trigger, we can turn an LED on only during the flyback.

PMT blanking and PMT decay

We have established that we can use the onset of flybacks as a sample clock for drawing optogenetic stimulus waveforms, and we have established that LEDs can be gated at the speeds needed to "inject" photons in a narrow period of time where we aren't saving imaging data. We aren't done though. In most systems the PMT stays on during the flyback, its just that the data isn't saved. In some cases the Pockels Cell is blanked during the flyback (as in resonant scanning), but the PMT remains on. There is good reason for this as PMTs do not turn on instantly, instead, they take microseconds to reach their steady state in both sensitivity and decay. Meaning that if we were to apply some voltage to a PMT and deposit a few photons onto it for 1 nS, and then watch what happens over the next 100 microseconds, we would see the PMT takes 10 microseconds or so to fully decay back to 0. Likewise, if we leave a PMT off for 100 years, then turn it on for that 1 nS where we place 10 photons onto it the value the PMT will tell us is a fraction of what it would tell us if we waited a hundred microseconds or so. This means that we need to rapidly turn the PMT off during the flyback, inject some photons, turn it back on and wait a bit before we scan the next line. All of this is possible, but requires some electronics and programming.

I want to say at the outset that no matter what we do, turning the PMT off and back on leads to settling time that we can not fully work around. We found that some small proportion of each line, where a light was gated in the flyback, shows signs of signal stabilization (usually dimmer that it would be).

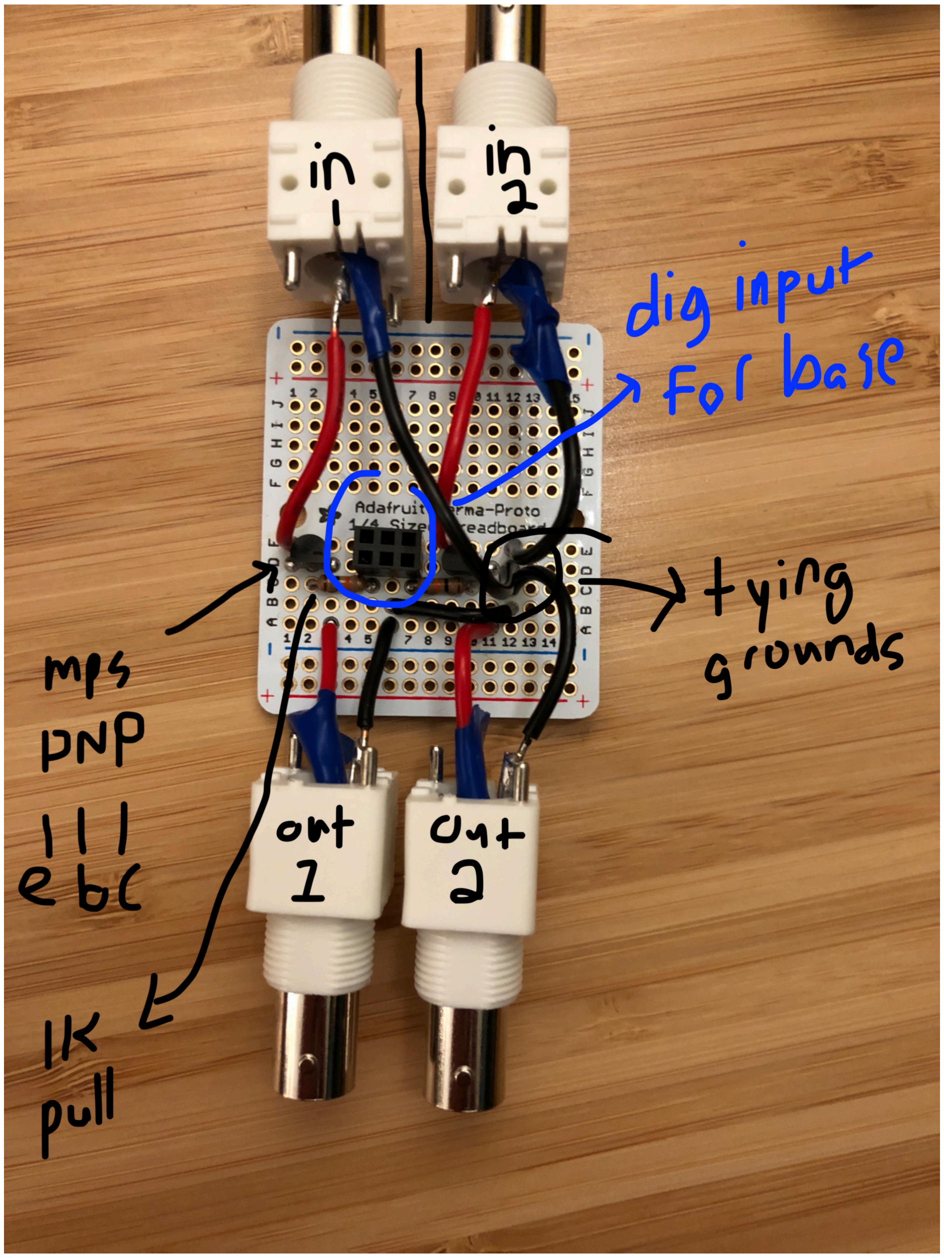
To rapidly "blank" a PMT, the control signal to it needs to be gated by a PNP transistor (normally on transistor). When you vary the PMT sensitivity in your microscope software, you are regulating the voltage on the PMTs. This control signal is an analog signal that is likely 0-5V. A PNP transistor allows the control signal to be on like normal, but you can send a TTL/CMOS pulse to turn the control signal off. In this case, your system will work like normal, unless you send a signal to intervene. For most people, controlling up to two PMTs is typical (green and red). To the right is an example circuit using basic and cheap MPS2907A PNP transistors:



https://cdn-shop.adafruit.com/product-files/2975/MPS2907A-D.PDF

These transistors are "high load," which means they can gate 800 mA of current. That is PLENTY the control signal to the PMT, which is a buffered control voltage and has 10-20 mA, at most.

Here is an actual prototype for 2 channel control:



For this, the orientation of the PNP is emitter-base-collector (ebc), from left to right, respectively. The input's signal provided by its BNC is connected to the emitter and the routed our of the transistors collector unless the base is pulled high. The base is connected to any digital pin on a teensy/arduino via a 1-10K pull up resistor. Note, I soldered in a 2x3 female pin receptical such that the pull-up is on the same column (#1 and #3) and the middle row is connected to ground. This allows you to easily plug in a wire from an arduino/teensy that is gate, ground, gate.

A teensy/arduino can be programmed to respond to a trigger indicating the onset of a flyback and turn the PMT off using the above circuit. At the same time, the teensy can gate the Cyclops driver, and then the PMT back on.

I've added these routines into "csStateBehavior" and they are triggered by a digital input on a Teensy 3.5/6's pin 35 (by default), set as an interrupt. Thus, in any state, at any time, an input on 35 will blank the PMTs, and determine what voltage to send to a Cyclops led driver, based on a desired pulse waveform (also handled by the teensy.)

Here is the routine (just for conversation purposes; details aren't crazy important):

```
void flybackStim() {
  elapsedMicros pfTime;
  pfTime = 0;
  digitalWrite(PMTPin,HIGH);
  while (pfTime <= knownValues[15]) {
    stimGen(pulseTrainVars);
    analogWrite(DAC1, pulseTrainVars[0][7]);
    analogWrite(DAC2, pulseTrainVars[1][7]);
  }
  analogWrite(DAC1, 0);
  analogWrite(DAC2, 0);
  digitalWrite(PMTPin,LOW);
}
```

What it does is it a) starts a microsecond timer called pfTime, b) resets the timer, c) sets the PMT base high (to blank the PMT), d) uses the stimGen routine to determine what DAC values should be written based on your desired variables, then writes the values on the DACs, e) when the timer is greater than the desired flyback duration (stored in knownValues[15]) it returns, f) then sets the DACs to 0, and turns the base off (allowing the PMT to come back on).

The details about analog outputs and how to vary them is in the main csStateBehavior documentation.

This function is attached to a trigger pin as an interrupt. How we generate a TTL to signal a flyback is our last remaining detail.

Signaling/Predicting/Detecting Flybacks:
(Will finish 10/3/2018)