



**Boston University  
Electrical & Computer Engineering  
EC464 Senior Design Project**

**Final Testing Report**

**Integrated Laser and Electronic Model for  
Photoacoustic Neural Stimulation**

By Team 19  
Team Members

Tian Huang [tianh1@bu.edu](mailto:tianh1@bu.edu)  
Raphael Mok [rmok@bu.edu](mailto:rmok@bu.edu)  
Ian Lee [ianxlee@bu.edu](mailto:ianxlee@bu.edu)  
Mateus Toppin [mtoppin@bu.edu](mailto:mtoppin@bu.edu)  
Viet Nguyen [vietng99@bu.edu](mailto:vietng99@bu.edu)

### **Equipment and Setup:**

The equipment for our final prototype test was aligned with the requirements detailed in our test plan. Utilizing a Teensy 4.1 board as the core component, an oscilloscope, BNC to alligator clip wires, a red and black M/M jumper wires for secure connections, USB-A to USB-C port to , and a computer to run the Python-based software application designed for parameter adjustments. This setup aimed to facilitate a seamless interface between the hardware and software components, ensuring the generation of precise pulse (square waves) for neural stimulation purposes. Lastly, the equipment also involves a multimeter and an extra set of BNC to alligator clip wires for measuring the voltage and current level.

The Setup is very similar to the previous testing and it is aligned with the setup mentioned in the final prototype test plan. The first step is to turn on the Oscilloscope and gather all the equipment mentioned above. Verify and Upload the .ino code from the Arduino IDE onto Teensy 4.1 board to generate the pulse. Then, connects the computer with the Teensy 4.1 using the USB-A to USB-C cable and runs the software application for preparation of testing. Lastly, connect pin 3 of the Teensy 4.1 board and the red alligator clip with the red M/M jumper wire. Do the same for the ground pin of Teensy 4.1 board and the black alligator clip with the black M/M jumper wire. In short, run both the Arduino IDE and the software GUI application files on the software sides. Then, connect all the wires on the hardware side in preparation for final testing.

Overall, the equipment and setup procedure is very straightforward and all for the purpose of smooth testing and protocol. The primary objective was to verify the system's capability to generate pulses with specified parameters input through the software application, focusing on frequency, duty cycle, and pulse width modulation (a sequence of on/off time). The setup aimed for seamless interaction between the software application and the hardware, ensuring accurate pulse generation on the oscilloscope. As described above, the hardware side was set up exactly as described in the test plan and the same could be said for the software side where GitHub stores all our code and the team initializes all components before the test starts..

### **Measurement Taken:**

Throughout the testing phase, the team conducted a series of measurements focusing on the accurate generation of pulse waves based on specified parameters. These parameters included variations in frequency, duty cycle, and pulse width modulation, which were inputted through the software application. Many different test cases were tested, but below will primarily discuss measurement that is most commonly used by our client and benchmark for testing.

The first step is to check if initialization is good. This is done quickly by checking the Port and Baud Rate part of the software application. Successful connection means the port and baud rate is auto detected which it is in our case with Com5 port and 9600 Baud rate. The first baseline test measured is by setting the frequency at 1,000 Hz, duty cycle at 50%, and PWM sequence at [10ms, 10ms]. Note, an even index including 0 means on and an odd index means off. By observing the pulse generated on the oscillator scope, this based test is performed

successfully. The output measured from the oscillator scope shows 10 square pulses over the course of 10 ms and followed by 10 ms of offtime. The number is 10 because  $1000 \text{ Hz} * 0.01 \text{ s} = 10$  (Note Hz = cycle/time so unit cancels out). Each pulse is also exactly outputting at 50% duty cycle as shown by the grid on the scope. One extra feature that is tested is matching the generated scope with the new preview functionality that is newly implemented. Preview window shows the correct generated pulsed under ideal circumstances like no noise, etc. The preview generated using parameter above is identical to the oscillator scope. In short, the pulse generated is correct with 10 pulse on over 10 ms and then 10 ms off.

Subsequent tests involved altering the frequency, duty cycle, and PWM sequences. The output was closely monitored for each set of parameters to ensure it matched the expected results. For instance, if the parameter duty cycle was altered from a lower value to a higher value while everything else remained constant, the width of the individual square wave should increase as with the value. One test measurement successfully demonstrated this idea is holding frequency at 2000 Hz with PWM sequence of [1ms, 1ms] constant while changing the duty cycle. Since duty cycle only affects the width of individual pulse, the outputting wave for above should be 2 square waves over 1 ms and 1 ms of offtime. Now, as shown during the testing, the duty cycle of 25% only shows a quarter of the whole pulse, 50% shows half of each individual pulse, 75% shows three quarters and so on. The team scales the grid according to the horizontal axis such that each grid is aligned with 100% duty cycle. Thus, a 50% duty cycle shows the pulse covering half of the grid. Overall, the measurement for duty cycle is measured successfully since the width increases according as the value is being adjusted.

Generally speaking, the testing of parameters is very straightforward by just making sure the number of pulses is valid and the on/off time is accurate. The duty cycle is almost always held at 30%. One last test measurement that is worthy of mentioning is input of high frequency at 10,000 Hz and PWM sequence of [0.1 ms, 0.1 ms, 0.5 ms, 0.5 ms]. This needs to be tested because the lab laser is driven at the threshold and most common frequency of 10,000 Hz. Naturally, the resulting square wave generated on the oscillator scope is correct. The sequence of pulse shown is 1 cycle of pulse over first 0.1 ms, 0 pulses over next 0.1 ms, 5 cycles of pulse over the next 0.5 ms, and finally no pulse for the last 0.5 ms. The whole sequence takes 1.2 ms. All these values are tested and measured for accuracy. One last function tested is the timer function. How the timer is measured is just setting the timer to any value, and once the timer is up, no pulse will be generated. For above, the timer is setted to 10 second, once 10 second hit, no pulse is shown on the oscillator scope.

Overall, the measurement is mainly measured using the oscilloscope. Each pulse test case was first calculating the number of pulse and timing which was then checked. The measurement here is mainly the accuracy of each case which is all accurately performed. Lastly, the teams measured the output voltage at high of 3.3 V and low of 0 V along with the output current at roughly 29 mA.

## **Conclusion:**

In conclusion, the final prototype test provides extraordinary results. In fact, the team would argue that the current stage of the project has fulfilled essentially 95% of all the total functionality. This is because the client requires two major functions for this project. A hardware component that can generate the needed square wave to pulse their laser and a software component that can act as an interface for the laser by being able to control it with certain parameters. All that is left is printing out the enclosure and fixing the current issue.

The current software applications fulfills all the requirements. There is an auto detection of port and baud rate for each system, a timer function that will automatically stop generating the pulse once the time is set, and the self-explanatory start, stop, and reset buttons. Moreover, the software application also has a preview functionality that generates the correct pulse wave. Overall, the software application prioritizes the concept of user-friendliness that allows the user to send the required parameters to the hardware board. The hardware on the other hand consisted mainly of the Teensy 4.1 board. When connecting the two components, the device is able to perform what is needed for the client as shown in the final prototype testing.

To sum everything up, the team is able to demonstrate a working final prototype that is able to deliver the correct pulse with almost no jittering from the data inputted on the software application. All work as intended on both the software and hardware sided during this test. However, one last minute issue the team encountered outside of the testing is the current project is likely generating too little of current based on the installation of the device onto the client's lab. Thus, the remaining task is to implement a current amplifier. In short, the team is able to demonstrate a functioning prototype that can generate the required pulse and a working software application during this testing. The team estimates 95% of total functionality is completed, with the remaining current amplifier working to be implemented.