DALHOUSIE UNIVERSITY
FACULTY OF ENGINEERING

Group #19

DAXSONICS
ultrasound

Department of Electrical and Computer Engineering

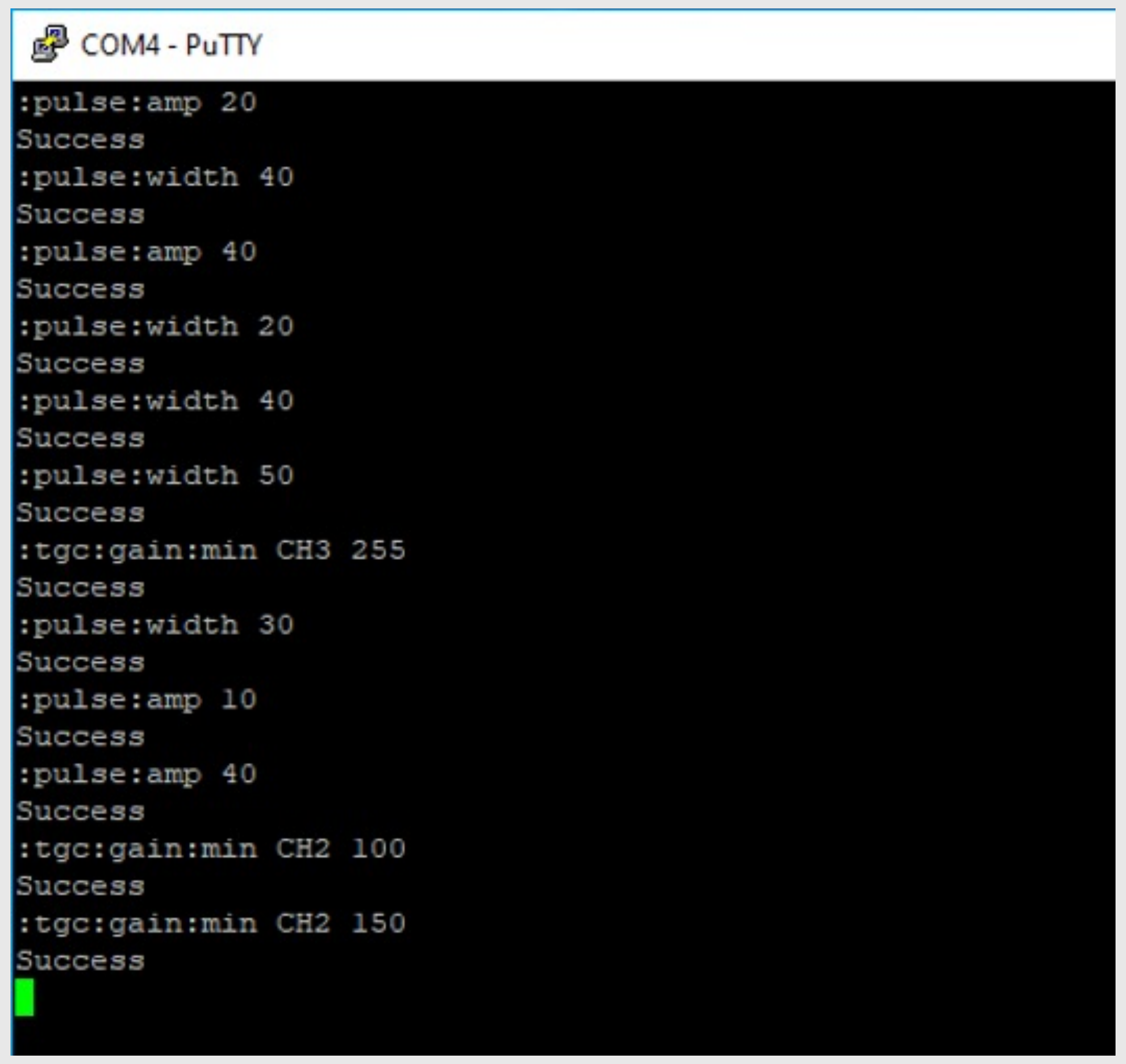
Graphical User Interface for Pulser/Receiver

5 Channel Pulser/Receiver

Pulser Receiver is a 5-channel high-frequency ultrasonic analog front-end with a programmable transmit beamformer.

- Variable gain, 14.5 – 75.5 dB
- Adjustable analog time-gain compensation
- Up to ±100V pulse amplitude
- 1 ns delay resolution
- 2 ns step size for transmit pulse widths
- 5 – 75 MHz operating frequency

Given the pulser is programmable, there is a need for some interface for user control over the device parameters. Currently, the pulser is interfaced using serial commands.


COM4 - PuTTY

```
:pulse:amp 20
Success
:pulse:width 40
Success
:pulse:amp 40
Success
:pulse:width 20
Success
:pulse:width 40
Success
:pulse:width 50
Success
:tgc:gain:min CH3 255
Success
:pulse:width 30
Success
:pulse:amp 10
Success
:pulse:amp 40
Success
:tgc:gain:min CH2 100
Success
:tgc:gain:min CH2 150
Success
```

Our project is to create a GUI application that lets users easily control the device. Our task is to enable all the functionality that is on the user manual without the requiring any sort of serial command entry.

Design Process

Before we started designing a UI, we needed to think about requirements and the feature set. We have written 5 questions, based on the requirements, that we considered when choosing frameworks.

Design Considerations

- Which language?
- Cross-platform?
- Pulser communication?
- GUI library?
- Application deployment?

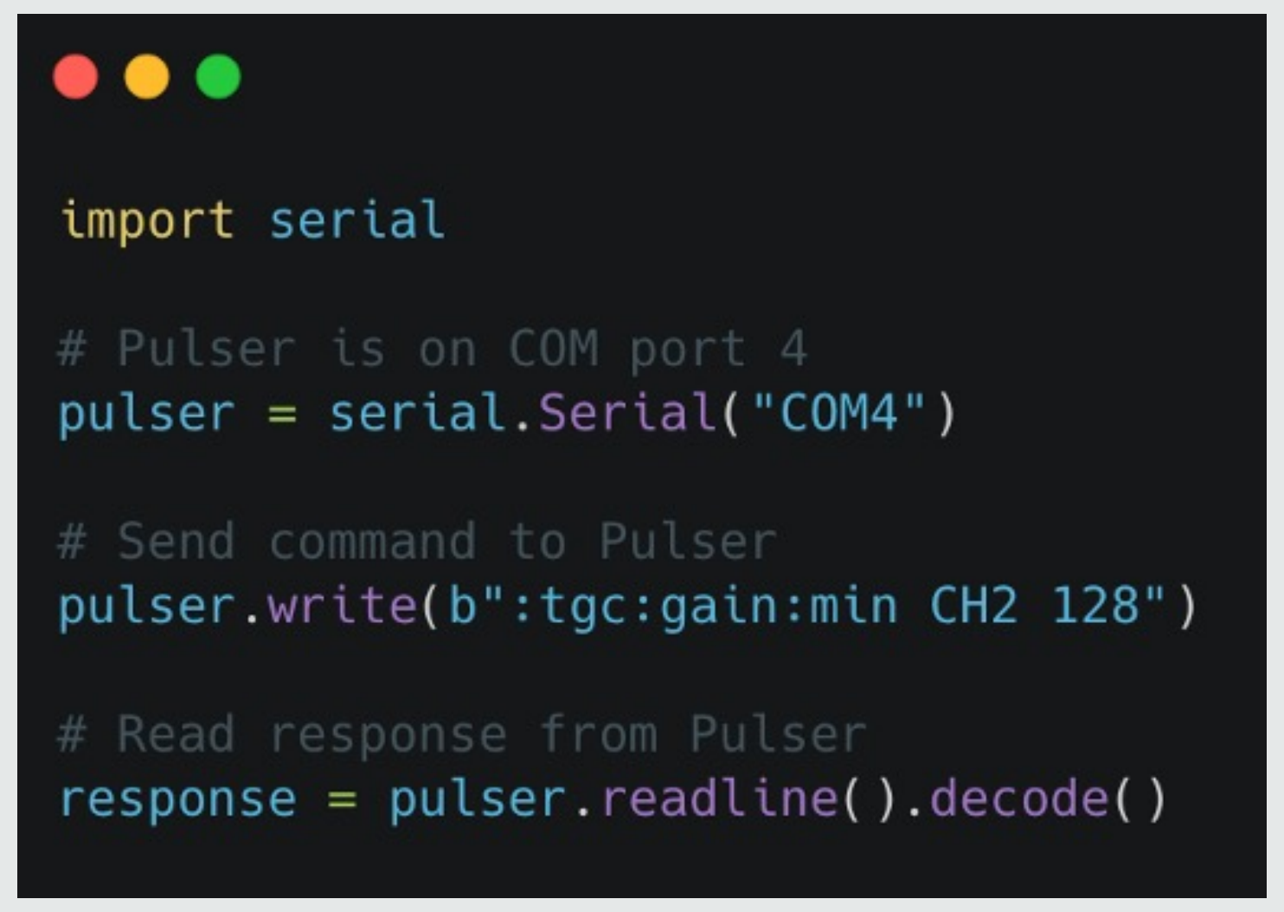
Application Details

Cross-platform

- Python and all our frameworks are cross-platform
- Focus is Windows

Python 3

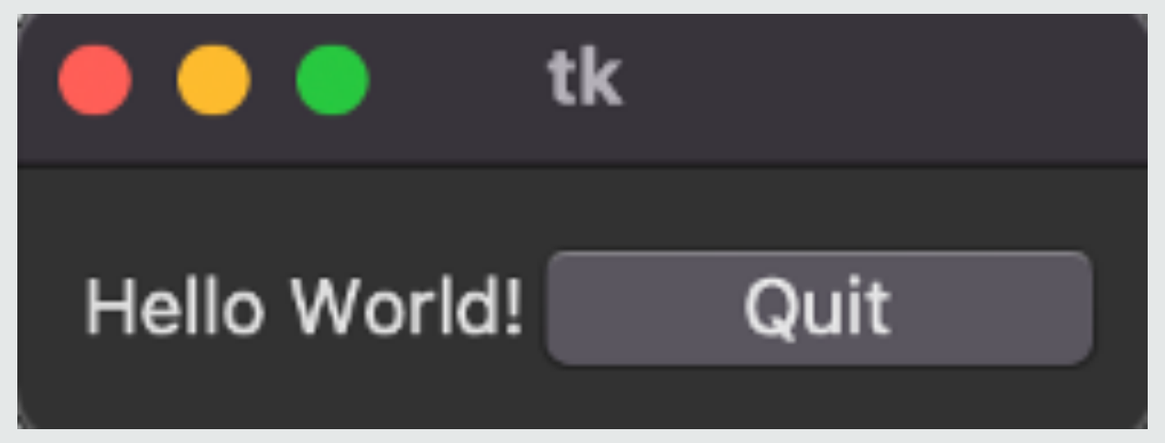
- Most popular programming language
- Easy to learn
- Public libraries make it easy to implement high-level programming functionality



```
import serial

# Pulser is on COM port 4
pulser = serial.Serial("COM4")

# Send command to Pulser
pulser.write(b":tgc:gain:min CH2 128")

# Read response from Pulser
response = pulser.readline().decode()
```

PySerial

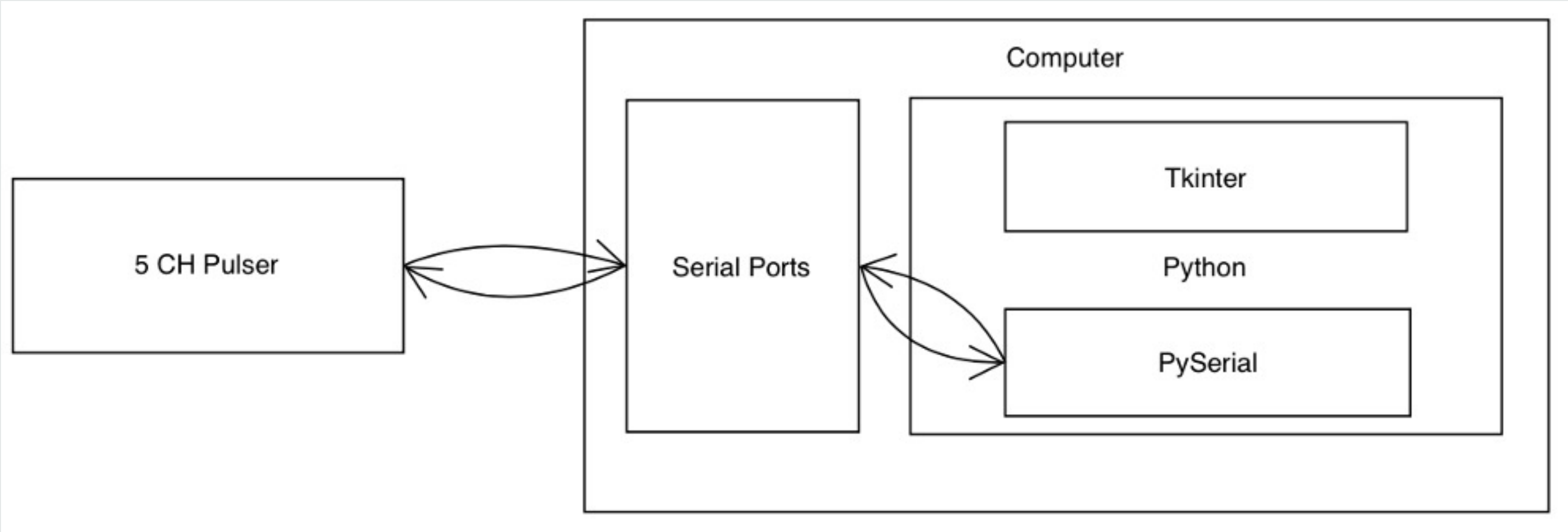
- Used to communicate with the pulser
- Encapsulates the access for the serial port



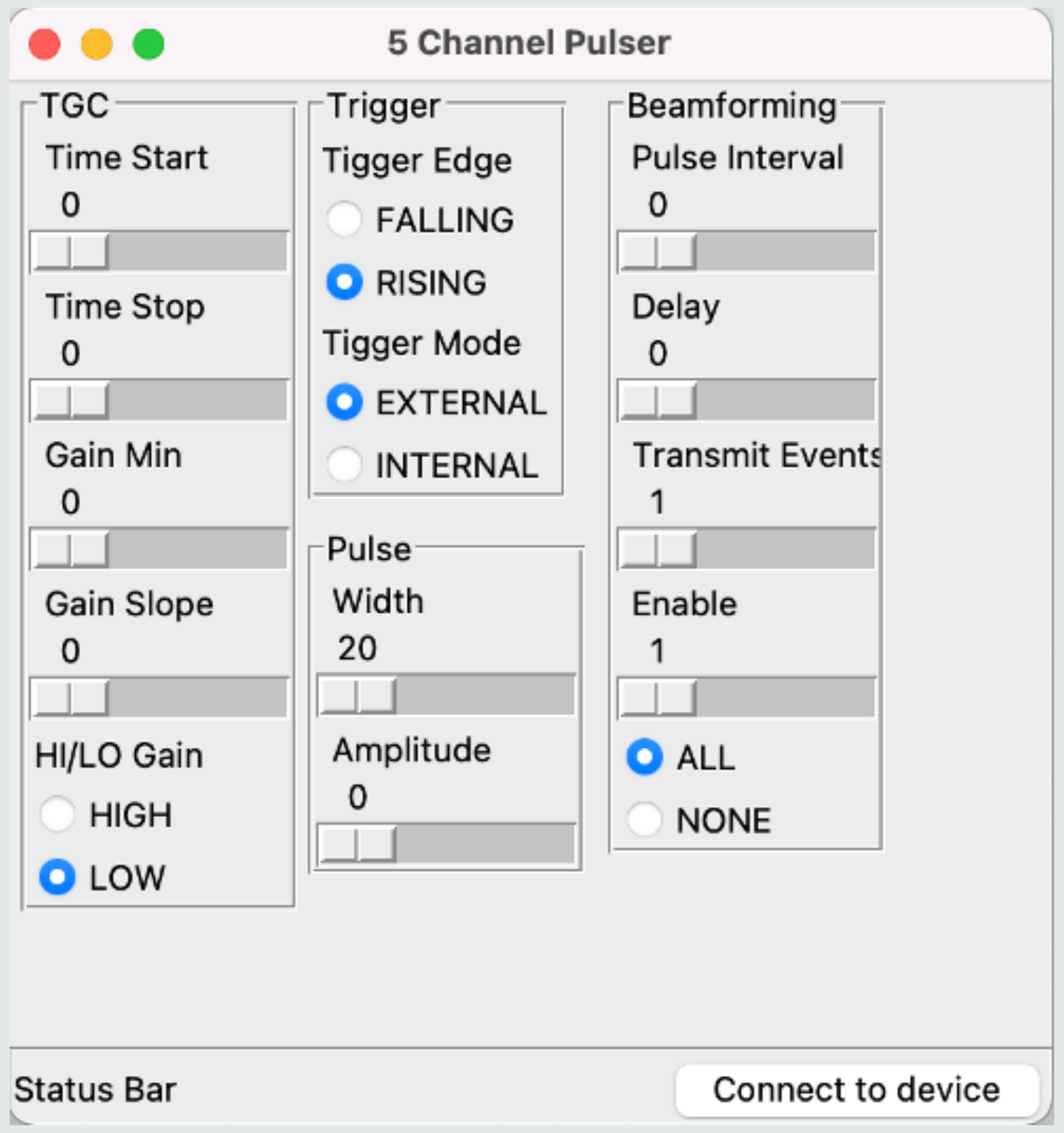
Tkinter

- UI Framework
- Includes many widgets to use as UI elements enabling user input to trigger events using input data

High-Level Architecture



Final Product and Conclusion


5 Channel Pulser

TGC
Time Start 0
Time Stop 0
Gain Min 0
Gain Slope 0
HI/LO Gain ○ HIGH ● LOW

Trigger
Tigger Edge ○ FALLING ● RISING
Tigger Mode ● EXTERNAL ○ INTERNAL

Pulse
Width 20
Amplitude 0

Beamforming
Pulse Interval 0
Delay 0
Transmit Events 1
Enable 1
● ALL ○ NONE

Status Bar

Instead of configuring a serial client and entering terminal commands, user can use the graphical user interface control the device. The UI is designed around the 13 commands detailed inside the user manual. Project requirements are met for this term.

Future Features

Future features that will be added in the following term include the ability to simulate the output generated pulser waveform right within the GUI. This would eliminate the need for an oscilloscope to see what the output waveform looks like.

References

- https://www.daxsonics.com
- https://pyserial.readthedocs.io/en/latest/
- https://docs.python.org/3/library/tkinter.html