PYUSBUS: OPENING USB ULTRASOUND PROBES

A POLYGLOT DOCUMENTATION FILE

Luc Jonveaux[*]
Curious, Charleville-Mézières, France
pyusbus@un0rick.cc

December 22, 2021

ABSTRACT

A number of USB ultrasound probes can be found on the general, open online markets. Bibliography shows that the first probes to be released where mechanical probes, such as the "SeeMore" series by Interson, promising to "turn every computer in your facility into an ultrasound imaging system", providing a "affordable, portable, and easy to use" platform.

Having open drivers and libraries to communicate with ultrasound probes could help of course both probes' users, by supplying an easy to use API, but also researchers, by providing additional insights in the workings of the probes.

Still, the communication with probes is locked by proprietary hardware drivers. Vendors may open their SDKs, but those could be limited, on several levels. The purpose of this exercise is to show that it is feasible to put together the bases of a common, portable platform to use these c.a. 1000 USD probes.

This PDF is also a ZIP that contains the sources files of this PDF and of the code, don't hesitate to have a look. Just rename the file from .PDF to .ZIP and you're ready to go .

Keywords open-source · ultrasound · hardware · retro-engineering · usb · probes

1 Overview

Getting signals and images from ultrasound mechanical probes is an interesting step to know what radiofrequency signals mean in ultrasound imaging, be it for non-destructive testing or medical imaging. We have developed two pulse-echo boards [4, 5], which are achieving this. However, operating them requires extra hardware (electronics, probes, a fair bit of soldering, ..).

Getting ultrasound images could be more straightforward, especially considering 1kUSD usb probes available on the market. Their image quality is quite good at first glance.

1.1 Modus operandi

We examined the exchanges on the USB ports between the vendor software and the probes, using the excellent Wireshark software. For the sake of operation, the four probes described below were used from within a virtual machine.

The packets were captured using Wireshark, and parsed using scapy as a pcap-parsing library and pandas for sense-making. A particular attention was paid to the initial packets, preceding the acquisition of first imaging data, identified as the transfer of large packets.

We distinguish between simple commands sent to the probe, which we tried to assemble in patterns, from larger packets which are pages of data sent to the probe for different purposes (beamforming, gain setting, depth, ...). Fig. 1 shows the content of a parameters packet supposed to setup the beamforming parameters on the UP20L.

We logged the different commands sent by software and named them where possible in the subsequent library, *pyusbus*.

[*]More on the website http://un0rick.cc/goodies/usbprobes. This paper has its on Zenodo DOI 10.5281/zenodo.5792256

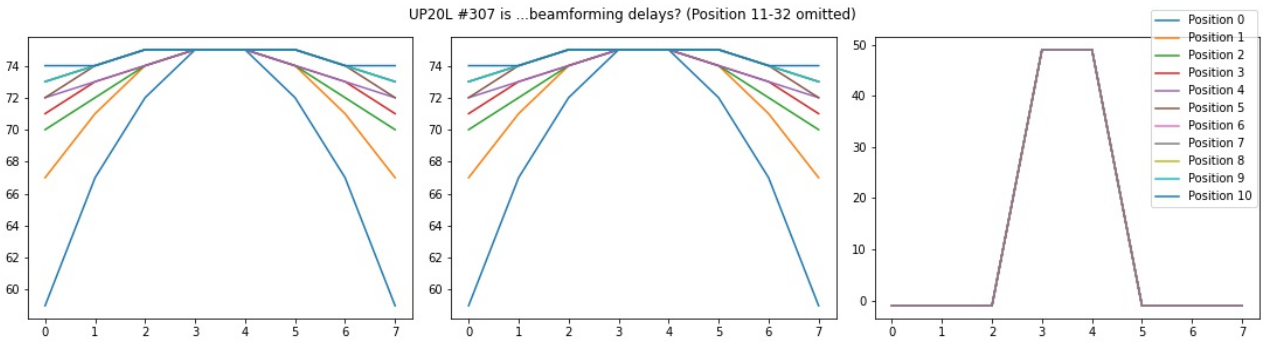

Figure 1: Example of the content of a packet

1.2 Probes

1.2.1 Linear, B-mode

We chose a simple, linar Healson UP20L probe to start the exercise. It appeared to be using from Cypress FX2/FX2LP (USB2) bridge chip, so we used the fx2 python library. The startup pattern was relatively complex, and we tried to simplify it as much as possible.

After a startup, the freeze/unfreeze commands were determined. Images were then obtained.

1.2.2 Convex, B-mode

The convex probe was a BMV probe, relatively straightforward to setup.

By default, it exposes images of 80 lines (possibly aligned with a 80-elements probe), each of which has 3900 points per line. This means the "images" are not the enveloppe signals, rather direct radio-frequency, or IQ sampled signals, as seen in Fig. 2.

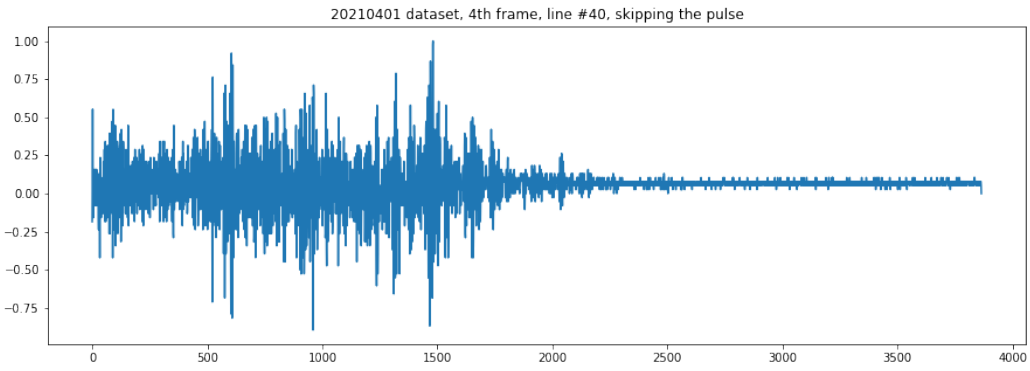

Figure 2: Example of the content of a RF packet

1.2.3 Interson, B-mode

The Interson is an interesting probe. Older than the rest of the probes, it is based on a different USB2 chip. The device is identified as 0x1921, 0x0001 before being programmed, and 0x1921, 0xf001 after. Interestingly, the family of device was already identified in public repositories [3].

The setup is prepared in two steps, the first ending with a rebook of the probe in a controlable mode. The second is easily verifiable: as the probe is a mechanical one, motor control can be felt and heard.

The major challenge is that as of today, the number of lines per image can vary, as the motor can have slight speed changes. Downstream software postprocessing is needed.

1.2.4 Linear, Doppler

This USB3 probe, by default, exposes images of 128 lines for 3584 points per line. Doppler images are arrays of 336 lines of 764 points each. We are not yet clear of the content of the doppler images. We have the option to only get B-mode images or interleaved images, alternating Doppler mode and B-Mode frames. Implementation is still uncertain on the geometry and location of the doppler window.

2 Challeges

The major challenge we face is the lack of documentation of the probes, and the time needed to understand how to "cleanly" set them up. One can record all commands sent to set a probe at a given depth, with a given TGC curve, and log everything - but this does not yield a fine view over the controls of the probe. We therefore keep the code open for possible contributors, for newer probes, but also to open finer control over the probes, for example over the beamforming parameters.

3 Results

A python class has been developped for each of the above-mentioned probes. It allows a standard acquisition, as well as a freeze/unfreeze control, and leads to allow better TGC controls. Significant work is required to transform this as a full-fledge solution, usable on the market. Figure 3 shows examples of images acquired from the library, or from an OEM software, on a phantom.

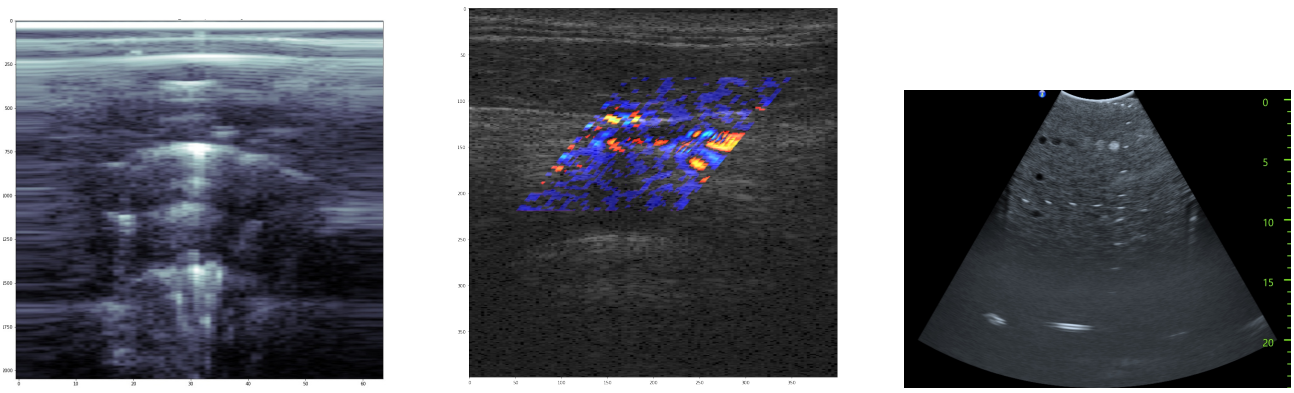

Figure 3: Example of images using the interson, doppler and convex probes.

4 Next steps

We hope that delivering this library would open a window of opportunity for current ultrasound researcher.

The note showed the state of the library, at the end of 2021. Four devices were examined and made accessible through its API. It has shown a variety of devices, which surprisingly may have more options accessible for researchers, and not exposed by the manufacturers - for example exposition of raw signals, and not image enveloppes, or access to configuration registers not thought accessible.

This can be promising to allow working on affordable development platforms.

Plenty to do on the next steps! The current shopping list (non-exhaustive) may include:

- Improving the documentation in general.
- Work with other probes.
- Explore remaining controls on current devices.
- Keep pcaps for each probe on given settings - pcaps without any large data transfers.
- Onboard your contribution to open the world of ultrasound imaging!

Feel free to use and cite this work!

References

[1] Luc Jonveaux 2017. Arduino-like development kit for single-element ultrasound imaging. In *Journal of Open Hardware, 1(1), p.3*. DOI: 10.5334/joh.2

[2] Luc Jonveaux 2018. The pyusbus repository. Website at *https://github.com/kelu124/pyusbus*

[3] Matthieu Heitz 2014. Interson Manager. Website at *https://github.com/KitwareMedical/IntersonManager*

[4] Luc Jonveaux 2019. un0rick : open-source fpga board for single element ultrasound imaging On *Zenodo*. DOI: 10.5281/zenodo.3364559

[5] Luc Jonveaux 2021. lit3rick: an up5k ultrasound pulse-echo device On *Zenodo*. DOI: 10.5281/zenodo.5792245

[6] Luc Jonveaux 2021. An open-source max14866 development board On *Zenodo*. DOI: 10.5281/zenodo.5792252

[7] Luc Jonveaux 2021. pyubus: opening usb ultrasound probes On *Zenodo*. DOI: 10.5281/zenodo.5792256