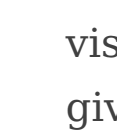
Onam break dreamOf Piano dabling transformed2 Linux Lockdown, Fedora BLS, Midi monitoring and a minimal DSOQuad Plot with Guided DigitalDecod prg



Hanish KVC Menon
Systems Architect / Tech Entrepreneur / Consultant
Published Sep 16, 2022

Short and Sweet

https://github.com/hanishkvc/prgs-hw-dsoquad-plotter

A program for plotting buf and dat format captures from DSOQuad as well as guided digital decode of analog signal to allow either synchronous or asynchronous bus data, including what ever be the bit pattern differences between the different standards, to be interpreted easily, without having to code too much (me being lazy and with short mind span per experiment), I have used a mechanism of

- allowing a custom virtual clock signal to be assumed / overlaid (for visual feedback) on top of the captured signal, starting from any given position decided by the user,

- and inturn use this along with a simple string of markers (chars 0-7 [bit positions in a 8bit value, in any required order], S[tart], s[top], p[rint bit], P[rint 8bit hex value], H[alf period]) to guide how to interpret the analog signal amplitude at the corresponding time step|div|period.

to easily guess / interpret the captured analog signal as digital data. It also shows the voltage and time difference between any two clicked positions, as well as counts the frequency. May add fft based freq plot in future.

the Long of it

Had originally planned to dust off my hardly opened decades old keyboard and learn some piano playing by connecting it to a laptop

over midi and pianobooster/tutka to explore, during the onam break.

Started out by patching pianobooster to map c# to sa, d# to re and so while displaying, so that the youtube harmonium leasons I had found could be mapped easily without having to also map to cdef... (me being lazy and a music newbie).

Inturn brought a down-to-earth-priced (CH345 based) usb-midi adaptor and connected the keyboard to laptop and there goes nothing. The leds on usb-midi kept blinking, but there didnt seem to be any midi data coming in to the laptop.

So thought of using usbmon to check if the data was coming through and or if there was some issue with aconnect-ing or the midi apps I was trying. But then dont we want linux to be more secure than windows, so jumps in kernel lockdown. Be it the old and no longer valid sysrq key combo or try cheat using kernel cmdline lockdown/security or lsm, lockdown wouldnt release :(

Well lockdown is a useful feature to have, and the trapping of possible paths to bypass it from the get go is also the right thing to do. But then instead of forcing disabling of secure-boot to be the only way to relax lockdown and or its parts, ideally one should allow secure validated kernel cmdline mechanism to achieve the same, especially atleast for end-user machines. The current solution is ok for Servers but Not end User machines. Also given that TPM is a necessity for secure boot, rather sealed/secured hash and pcr extending and or some such mechanism, which validates appropriate time physical access to machine++ should be the path that should be used to secure it. If in a year or two no one else has adds a patch for something similar, I may dabble at it.

For this and few other reasons, decided to patch the usbmidi audio path kernel code and inturn create a custom kernel for my fedora installation to see if any of the existing quirks there was creating some issue with a newer ch345 (just in case) and also to dump and or force certain things, if needed. And after compiling and signing the kernel and modules, I opened grub.cfg to add my entries and oops there was no fedora kernel entries in there. Cross checked the fedora admin guide and it still talks about the entries in there, but ... also if I remember correctly now using grubby (the recommeded tool in fedora), still didnt seem to setup things properly (have forgotten). Well it turns out, fedora has decided to use BootLoaderSpec based support in grub for its entries, which is a good thing at some levels. Tried creating a user account on fedora site, to respond to a similar query about missing fedora entries and direct the discussion towards

BLS mechanism and /boot/loader/entries and the need to update the sys admin guide and wiki etal, but for some reason there was some issue with creating the account, so if anyone with access to fedora documentation team reads this, do update the docs as well as forum posts to provide info about the new bls based fedora entries.

However even after adding the entries for my custom kernel and booting, the midi messages still eluded me, while the leds kept blinking like crazy while I stomped on the keyboard. So now the other possibility was that there was some timing configuration issue with the chip on the usb-midi module, given that midi uses a non standard 31250 transfer rate. Also as usb-midi class standard doesnt seem to have any way of querying or setting the same, decided to use the midi-out port of the usb-midi module to see if the generated messages from it had the right timing or not.

SideNote/Nostalgia1: Now such silly timing issue/oversight is not a unknown from my previous experiences. A decade back I had the fortune of scratching head with a few $k++ server for aircraft with 429 port support, not supporting the non standard for uart but standard for 429 - 100k clocking. What with motherboard superio chips being stupidly over-optimised to generate only the standard uart rates. Had to bring in our arm based control panel inbetween the server and the backend PSS system to get them to talk finally. So when few $K++ certified server can miss the mark, cant expect a <5$ usb-midi product to not have such oversight.

Nostalgia2: Again not wanting to hunt for the bulky tek/agilent scope at the last moment, decided to try out a old DSOQuad / DS203 pocket oscilloscope which I hadnt used for ages now. Had brought it initially to help with board bring up of a Ti DM3530 (most probably, have forgotten now) based board I had designed ages back, burning mid night oil sitting at hotel in Texas, next to the pcb assembly house, and Seattle aix expo hanging over the head. When you are looking at the 1st iteration of your design, you dont want to experiment with a pcb assembly house, who havent worked with the chip before, especially if there is a major expo just few days out ;) So India was out and USA was in for that board assembly. And I was debuging few signals well beyond the bandwidth of the pocket scope, but by looking at the relative variations in the distorted and aliased captures ;)

Addendum: Sometimes you will also have the privilage of working with assembly houses, which dont allow the team which designed the board to proactively work with them when they are assemblying your board for the 1st time. As crazy as it may sound, it was the reality, for some time locally here, a decade back, which has changed for better

now atleast with new assembly houses now. And this stupidity was inspite of the boards having assembling issues, due to this brain dead process. And in that respect the assembly house at Texas (people who have worked with Ti chips would know them) was friendly and gave free access to their assembly team, so that we could cross verify things as well as provide info where needed, unlike the assembly house here.

With **#DSOQuad** being a open source design, even thou the original device vendor firmware was very finiky with capturing signals then, the newer open source **#Wildcat** firmware now has overcome the issues greatly (there is some missing midi messages when monitoring, which I need to look at some time in the future). Thumbs up to **#OpenSource** movement and people bringing out such products.

Using same, captured the midi out signal. Then inturn to go through the same on PC, decided to create a simple yet flexible plotter and guided digital signal decoder of the captured analog signal, given that the data capture from this DSO is bit odd (its more of a kind of screen capture with some meta data), but is good enough for the job at hand.

To allow either synchronous or asynchronous bus data, including what ever be the bit pattern differences between the different standards, to be interpreted easily, without having to code too much, I have used a mechanism of

- allowing a custom virtual clock signal to be assumed / overlaid (for visual feedback) on top of the captured signal, starting from any given position decided by the user,
- and inturn use this along with a simple string of markers (chars 0-7 [bit positions in a 8bit value], S[tart], s[top], p[rint bit], P[rint 8bit hex value], H[alf period]) to guide how to interpret the analog signal amplitude at the corresponding time step|div|period.

to easily guess / interpret the captured analog signal as digital data. It also shows the voltage and time difference between any two clicked positions, as well as counts the frequency. May add fft based freq plot in future.

Using the same it appears like atleast as far as Midi Out is concerned, the usb-midi module is generating midi signals with proper 31250 Hz based timing. The voltage also seems to be ok.

Now this is bit odd, the module is blinking when ever I click keys. Its internal timing seems to be proper wrt midi standards, atleast based on what its midi-out generates/passes through. Still on the usb side wrt Midi-In, there is nothing but emptiness :([I scratch my head vigorously now]

At some point in future, have spent too much time on this for now, I need to open up the module/product and look at the values of the current limiter resistors as well as the part number of the optocoupler used in its midi-input path and see if its turn-on current and voltage requirements are being met or not. Unless I can think of something else, I may be missing here. Or does it require some special message to be sent before it starts sending out midi messages or ...

Or else I will have to rig up a usb-midi module on my own, provided I have a uC dev kit with usb device support, some where in my attic, to setup + program for this task.

#Midi #CH345 #USBMidi #Linux #Fedora #Oscilloscope #Hardware #DS203 #TPM #SecureBoot #Lockdown #Grub #Analog #Digital #SignalDecode #Python #Timing