Note 18_1_23

(13:27)
Today, I'm trying to start writing code for my mouse driver. I think the following will be my main tasks for today.

- Disable the standard linux mouse driver
- Re-build the kernel with Rust USB patches
- Begin writing USB Code

I'm going to do this on a clone in case I break things, this way I still have a fallback.

(13:54)
RustForLinux kernel successfully built on virtual setup so issues with workstation setup are most likely to be a problem with the config.

(14:09)
Currently re-building after adding in what USB support is available.

(14:11)
Successful kernel build with the current Rust USB support.

(14:20)
Calling the USB library doesn't cause any noticeable issues.

(14:23)
Rather than writing out my own C version of the driver, I spent some time yesterday looking for potential equivalents that I could either re-implement or use as inspiration for my own Rust driver. I might not get very far with this but any result can be used I suppose.

I'm not going to disable the standard linux mouse driver at the moment, the hope is that I can write something very basic that simply registers the mouse and prints information to `dmesg` to indicate that the registration was successful. This way, I can build it step by step and not rush too far forward or anything.

As a reminder to myself, I'll also need to remember to generate a new ssh key so I can save my progress to GitHub. But right now, I'd like to focus on writing some code.

(15:08)
After tinkering for a little bit, i've ran into issues when trying to build the driver with calls to the necessary usb libraries. I'd also like to note that the documentation contain within the library code isn't great and has little demonstrations [this also seems to be noted in code reviews for the relevant pull request.]

I'm still trying to make sense of this library. I copied the example from within the lib code to my own code to get things started, upon building I run into these errors.

```
  RUSTC [M] /home/kyle/Hons-Rust-DeviceDriver/rust_mouse.o
error[E0432]: unresolved imports `kernel::usb`, `kernel::define_usb_id_table`
  --> /home/kyle/Hons-Rust-DeviceDriver/rust_mouse.rs:20:14
   |
20 | use kernel::{usb, define_usb_id_table};
   |              ^^^  ^^^^^^^^^^^^^^^^^^^ no `define_usb_id_table` in the root
   |              |
   |              no `usb` in the root

error: cannot determine resolution for the macro `define_usb_id_table`
  --> /home/kyle/Hons-Rust-DeviceDriver/rust_mouse.rs:43:5
   |
43 |     define_usb_id_table! {u64, [
   |     ^^^^^^^^^^^^^^^^^^^
   |
   = note: import resolution is stuck, try simplifying macro imports

error: aborting due to 2 previous errors
```

Clearly there's a calling issue that I need to solve but I'm not 100% sure on that macro error, perhaps it's a byproduct of the lib-call issue?

(20:05)
I'm still not 100% sure on what do with this - will definitely need research. The usb library is set up in much the same way as all other rust kernel libs so either there is some issue with its setup that I will discover or the example in the documentation is plain wrong (which is a little frustrating and unfortunate).

If I manage to fix this but continue to run into other issues then I will 100% need to rethink the project deliverable. I know for a fact that I can do this, it's just figuring out what to implement feasibly within the next couple of months...

To spice things up, I'm going to work on some disseration writing/research. Might be good to get my Rust section started.

(22:00)
I've finished around half of Rust section within the literature review. Most of the content is working on improvements from the interim report, I still have some bullet points to work on. Another dissertation writing sessions and I expect the section on Rust to be fully or mostly finished.

If I find the time during my commute to UWS Hamilton tomorrow then I'll continue working on it. This concludes today.