Project Management: Developing Device Drivers in Rust

Kyle Christie
B00415210

School of Computing, Engineering and Physical Sciences

BSc (Honours) Computing Science
University of the West of Scotland

Supervisor: Paul Keir
Moderator: Stephen Devine

Table of Contents

Table of Figures

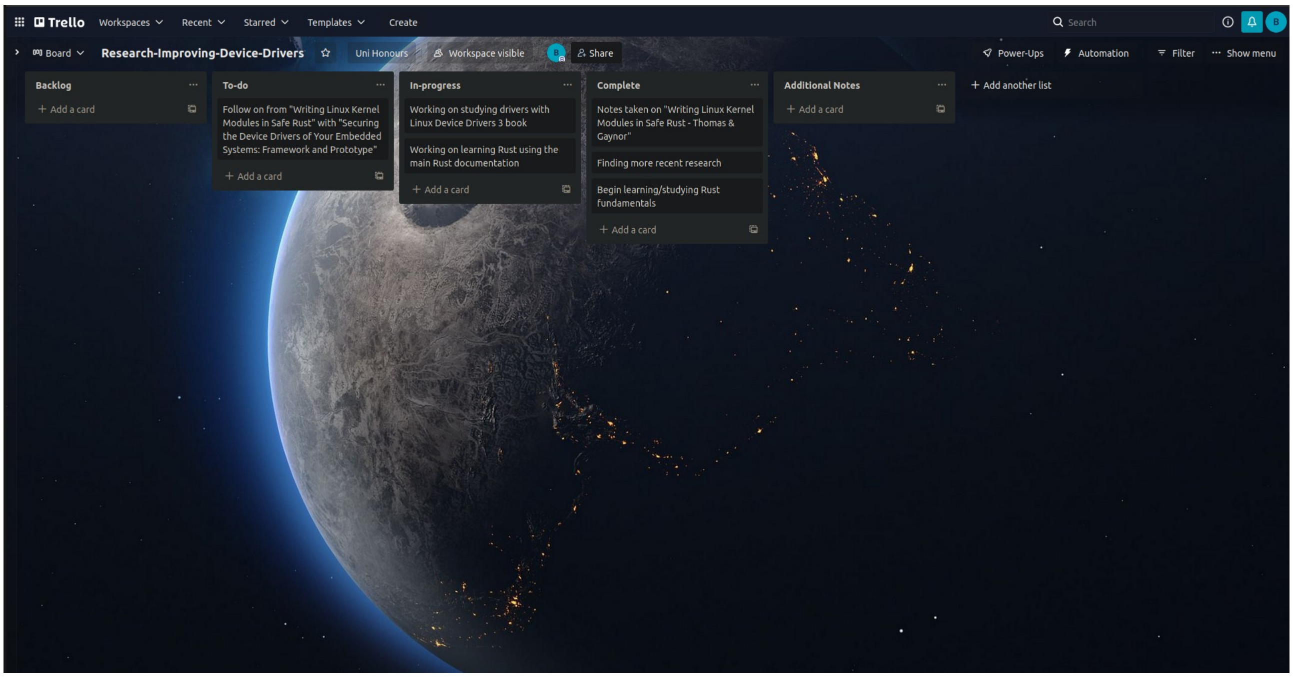
Abstract

Project management is a crucial aspect of any project and there are several tools available to assist users. In this report, the tools used to manage “Developing Device Drivers in Rust” are discussed. From GitHub repositories to note-taking, diaries, formal meetings to task visualisation, many tools were used to track progress and effectively manage the project.

1. Trello

Trello is a visual project and task management tool (Trello, 2023) which uses a format of lists, boards and cards to facilitate project management for users. Trello typically makes use of a 'kanban' style layout where a given board categorises tasks into distinct lists (or columns). Such lists commonly encompass three main types of task:

1. Tasks to be started or are not yet in progress.
2. Tasks that are in progress or have been undertaken.
3. Tasks that have been completed.



Backlog
+ Add a card

To-do
Follow on from "Writing Linux Kernel Modules in Safe Rust" with "Securing the Device Drivers of Your Embedded Systems: Framework and Prototype"
+ Add a card

In-progress
Working on studying drivers with Linux Device Drivers 3 book
Working on learning Rust using the main Rust documentation
+ Add a card

Complete
Notes taken on "Writing Linux Kernel Modules in Safe Rust - Thomas & Gaynor"
Finding more recent research
Begin learning/studying Rust fundamentals
+ Add a card

Additional Notes
+ Add a card

Figure 1: Research-Improving-Device-Drivers Trello Board (5/8/22)

Kanban is a lean workflow management method which helps to "visualize work, maximize efficiency, and improve continuously" (Kanbanize, 2023). It originated within the manufacturing field, specifically from the Toyota Production System, eventually being adopted within Agile software development.

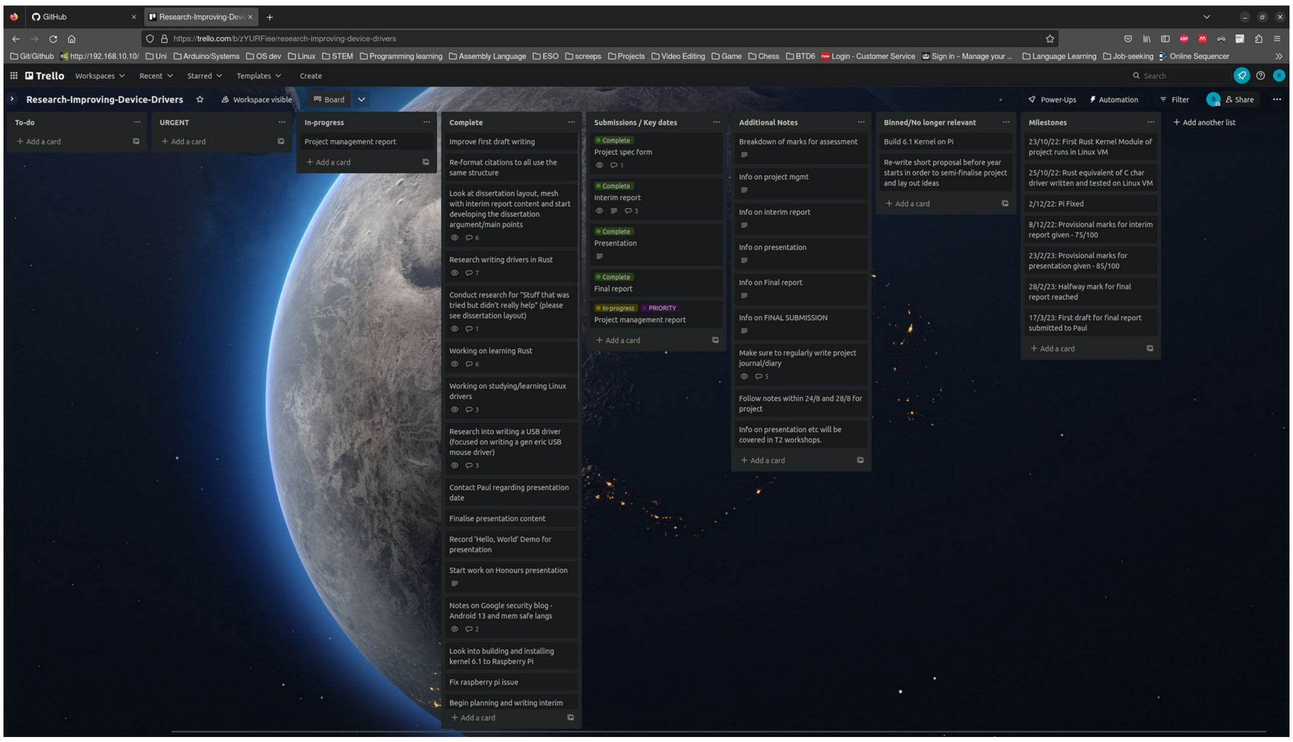
Throughout the project, Trello was used to track all tasks and key submission dates as well as serving to hold notes on submissions as necessary. The board 'Research-Improving-Device-Drivers' (found in figures 1 and 2) was split into nine lists as follows:

- 'Backlog'
- 'To-do' [sic]
- 'URGENT' [sic]

- ‘In-progress’
- ‘Complete’
- ‘Submissions / Key dates’ [sic]
- ‘Additional Notes’
- ‘Binned/No longer relevant’ [sic]
- ‘Milestones’

Within this board, present tasks are represented within ‘To-do’, ‘URGENT’, ‘In-progress’ and Complete with ‘URGENT’ being an additional list (when compared to common kanban boards) to highlight critical or time-sensitive tasks. ‘Backlog’ is used to hold tasks that are no longer relevant but may eventually see completion. ‘Submissions / Key Dates’ holds a list of core submissions to be made with some cards holding information regarding the specific date in question. ‘Additional Notes’ contains various necessary information placed within cards alongside short, simple reminders. ‘Binned/No longer relevant’ represents tasks will not be completed. The ‘Milestones’ list contains various notable milestones achieved throughout the duration of the project.

The previously mentioned Trello board was used to track various aspects of the project from achievements to task statuses and more. Within some cards, commits from related GitHub repositories were written as comments in order to directly link and categorise work.



To-do

URGENT

In-progress
- Project management report

Complete
- Improve first draft writing
- Re-format citations to all use the same structure
- Look at dissertation layout, mesh with interim report content and start developing the dissertation argument/main points (6)
- Research writing drivers in Rust (7)
- Conduct research for "Stuff that was tried but didn't really help" (please see dissertation layout) (1)
- Working on learning Rust (6)
- Working on studying/learning Linux drivers (3)
- Research into writing a USB driver (focused on writing a gen eric USB mouse driver) (3)
- Contact Paul regarding presentation date
- Finalise presentation content
- Record 'Hello, World' Demo for presentation
- Start work on Honours presentation
- Notes on Google security blog - Android 13 and mem safe langs (2)
- Look into building and installing kernel 6.1 to Raspberry Pi
- Fix raspberry pi issue
- Begin planning and writing interim

Submissions / Key dates
- Complete — Project spec form (1)
- Complete — Interim report (3)
- Complete — Presentation
- Complete — Final report
- In-progress, PRIORITY — Project management report

Additional Notes
- Breakdown of marks for assessment
- Info on project mgmt
- Info on interim report
- Info on presentation
- Info on Final report
- Info on FINAL SUBMISSION
- Make sure to regularly write project journal/diary (5)
- Follow notes within 24/8 and 28/8 for project
- Info on presentation etc will be covered in T2 workshops.

Binned/No longer relevant
- Build 6.1 Kernel on Pi
- Re-write short proposal before year starts in order to semi-finalise project and lay out ideas

Milestones
- 23/10/22: First Rust Kernel Module of project runs in Linux VM
- 25/10/22: Rust equivalent of C char driver written and tested on Linux VM
- 2/12/22: Pi Fixed
- 8/12/22: Provisional marks for interim report given - 75/100
- 23/2/23: Provisional marks for presentation given - 85/100
- 28/2/23: Halfway mark for final report reached
- 17/3/23: First draft for final report submitted to Paul

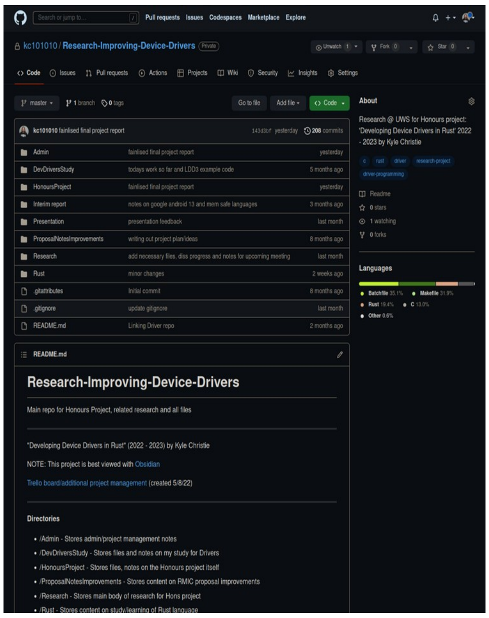
Figure 2: Research-Improving-Device-Drivers Trello Board (28/3/22)

2. GitHub

GitHub is a common tool used in Software development and engineering which is used to store project source code and track the history of all changes to such code (Lutkevich and Courtemanche, 2023). Alongside this, GitHub also acts as a means for development teams to communicate and collaborate with each other via Pull Requests and Commits (as showcased in Figure 4) which contain facilities for users to discuss and review a given contribution. GitHub is often used to host open source software development projects where all interested users are encouraged to contribute where possible. Several projects and tools used within this project are hosted on GitHub including Rust-for-Linux, Rust, the Linux kernel and more.

2.1 Git

GitHub relies on 'Git', a source code management tool created by Linus Torvalds (Lutkevich and Courtemanche, 2023). Git effectively provides the core features of GitHub (in source code storage and tracking) and is commonly used to manage projects where multiple developers may introduce conflicts with their changes.



kc101010 fainlised final project report 143d3bf yesterday 208 commits

Admin	fainlised final project report	yesterday
DevDriversStudy	todays work so far and LDD3 example code	5 months ago
HonoursProject	fainlised final project report	yesterday
Interim report	notes on google android 13 and mem safe languages	3 months ago
Presentation	presentation feedback	last month
ProposalNotesImprovements	writing out project plan/ideas	8 months ago
Research	add necessary files, diss progress and notes for upcoming meeting	last month
Rust	minor changes	2 weeks ago
.gitattributes	Initial commit	8 months ago
.gitignore	update gitignore	last month
README.md	Linking Driver repo	2 months ago

Research @ UWS for Honours project: 'Developing Device Drivers in Rust' 2022 - 2023 by Kyle Christie

README.md

Research-Improving-Device-Drivers

Main repo for Honours Project, related research and all files

"Developing Device Drivers in Rust" (2022 - 2023) by Kyle Christie

NOTE: This project is best viewed with Obsidian

Trello board/additional project management (created 5/8/22)

Directories

- /Admin - Stores admin/project management notes
- /DevDriversStudy - Stores files and notes on my study for Drivers
- /HonoursProject - Stores files, notes on the Honours project itself
- /ProposalNotesImprovements - Stores content on RMIC proposal improvements
- /Research - Stores main body of research for Hons project
- /Rust - Stores content on study/learning of Rust language

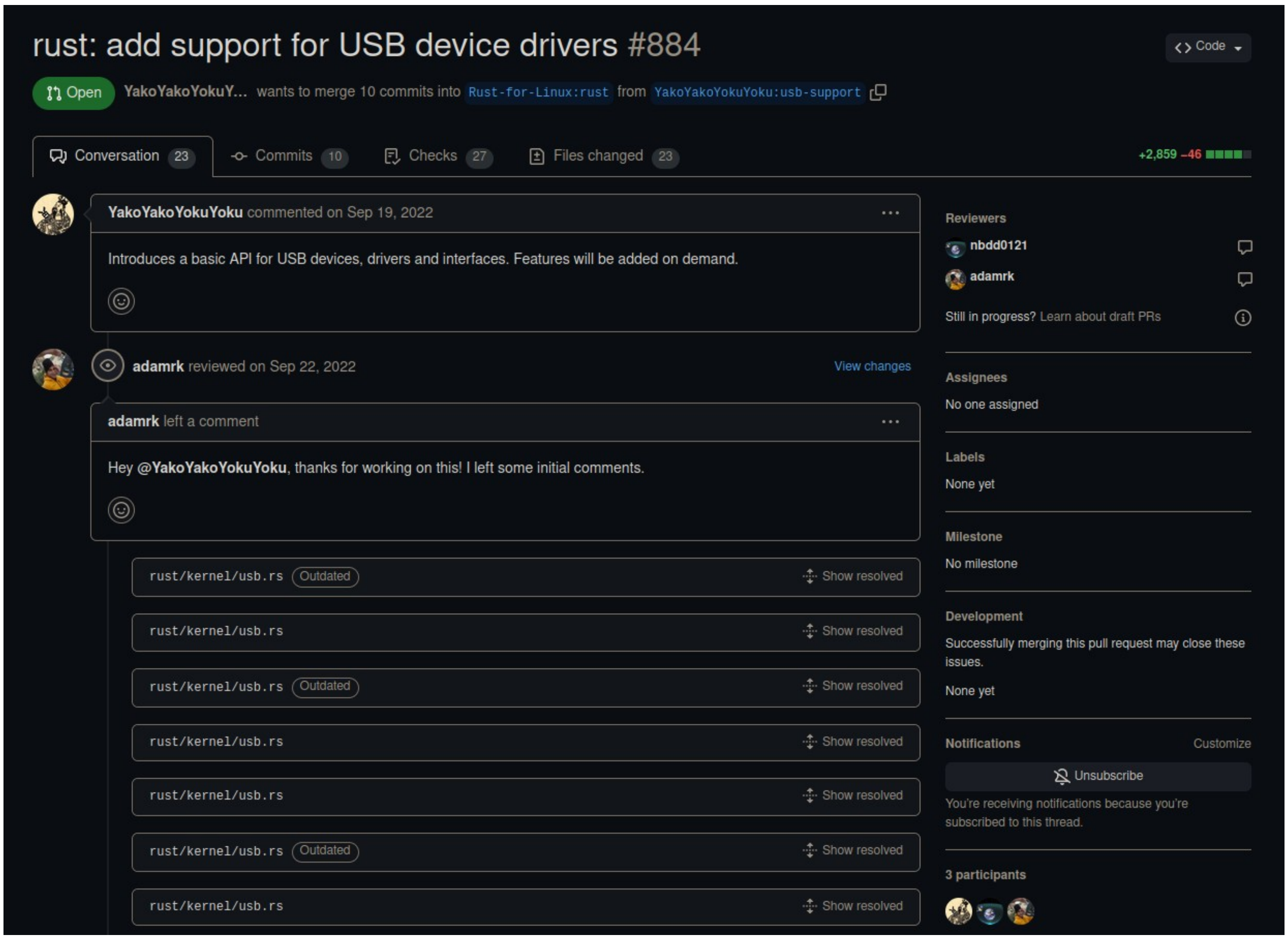
Figure 3: kc101010/Research-Improving-Device-Drivers repository main page

2.2 'Research-Improving-Device-Drivers'

With over 200 commits, created at the very start of the project (around July 2022). This repository (found above in Figure 3) served to store all main research and development undertaken during the project as well as several project components such as the Interim report and Presentation. The repository was generally split into several folders including 'Admin', 'Research', 'Rust', 'Presentation' and so on which categorised screenshots, research notes, research papers, source code and more. As well as acting as remote storage that facilitated easy access across several machines, the repository served as a full backup which could be easily re-stored in the case of machine failure, data loss or similar issues.

2.3 'Hons-Rust-DeviceDriver'

This repository was created in January with the intent of storing source code for the 'USB Mouse' Rust driver. As the planned driver was not implemented (as discussed in the final report), the repository instead stores a 'Hello, World' driver. Alongside the source code, the projects makefile can be found. This repository was also used as a backup and to allow for access outwith the virtual machine in which the driver was written.


rust: add support for USB device drivers #884

Open YakoYakoYokuY... wants to merge 10 commits into Rust-for-Linux:rust from YakoYakoYokuYoku:usb-support

YakoYakoYokuYoku commented on Sep 19, 2022

Introduces a basic API for USB devices, drivers and interfaces. Features will be added on demand.

adamrk reviewed on Sep 22, 2022

adamrk left a comment

Hey @YakoYakoYokuYoku, thanks for working on this! I left some initial comments.

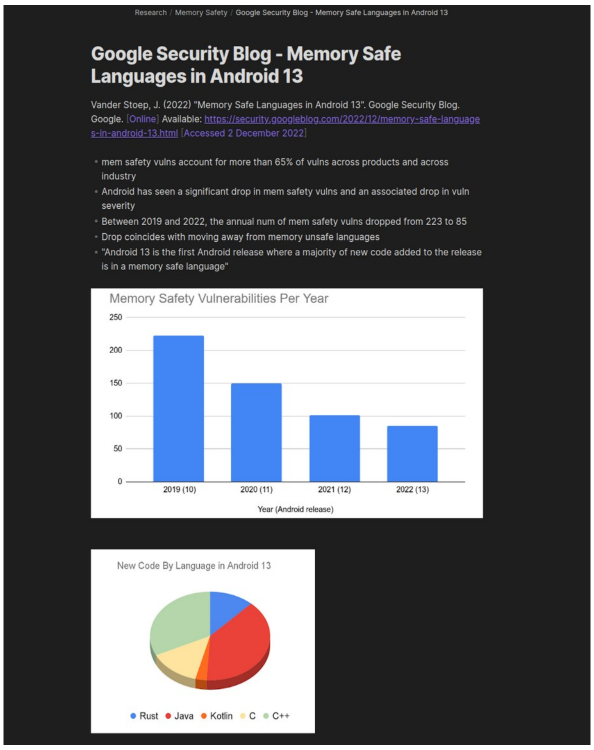
rust/kernel/usb.rs Outdated
rust/kernel/usb.rs
rust/kernel/usb.rs Outdated
rust/kernel/usb.rs
rust/kernel/usb.rs
rust/kernel/usb.rs Outdated
rust/kernel/usb.rs

Reviewers: nbdd0121, adamrk
Still in progress? Learn about draft PRs
Assignees: No one assigned
Labels: None yet
Milestone: No milestone
Development: Successfully merging this pull request may close these issues. None yet
Notifications: You're receiving notifications because you're subscribed to this thread.
3 participants

Figure 4: Pull Request #884 "add support for USB device drivers" (Rust for Linux, 2023)

3. Obsidian

Obsidian is a knowledge base and note-taking application (Obsidian, 2023) which was heavily used throughout the duration of the project. Its main use was holding research notes and holding early drafts of writing. Obsidian uses ‘markdown’ files allowing users to utilise markdown formatting within notes which is formatted and rendered in real time. Alongside the inclusion of markdown is the potential to link notes which can then be represented as a connected graph. Its design is such that users can “structure their thoughts and knowledge in a flexible, non-linear way” (Wikipedia, 2023). It is also possible to customise Obsidian via its ‘plugin’ functionality where plugins are created by both the Obsidian team and the community.


Research / Memory Safety / Google Security Blog - Memory Safe Languages in Android 13

Google Security Blog - Memory Safe Languages in Android 13

Vander Stoep, J. (2022) "Memory Safe Languages in Android 13". Google Security Blog. Google. [Online] Available: https://security.googleblog.com/2022/12/memory-safe-languages-in-android-13.html [Accessed 2 December 2022]

- mem safety vulns account for more than 65% of vulns across products and across industry
- Android has seen a significant drop in mem safety vulns and an associated drop in vuln severity
- Between 2019 and 2022, the annual num of mem safety vulns dropped from 223 to 85
- Drop coincides with moving away from memory unsafe languages
- "Android 13 is the first Android release where a majority of new code added to the release is in a memory safe language"

Memory Safety Vulnerabilities Per Year
250
200
150
100
50
0
2019 (10)
2020 (11)
2021 (12)
2022 (13)
Year (Android release)
New Code By Language in Android 13
● Rust ● Java ● Kotlin ● C ● C++

Figure 5: Research notes on Memory Safe Languages in Android 13

3.1 Diaries

Alongside research and so on, Obsidian was also used to keep regular diaries which were intended to track work and progress, contribute to project management and note thoughts while undertaking the work. Diaries were kept between August 2022 and March 2023, providing deeper insight into specific issues, thoughts and processes throughout the projects.

4. Formal Meetings

Several formal ‘feedback and management’ meetings took place with the project supervisor where insight was given into progress of the project as well as discussions on arising problems, current work, future tasks and so on. Alongside the provided formal forms intended to document these meetings, informal meeting minutes were taken in Obsidian with the intention of holding more in-depth, specific notes and important details that were to be kept.

References

Lutkevich, B. Courtemanche, M. (2023) “Definition: GitHub”. TechTarget. [Online] Available: https://www.techtarget.com/searchitoperations/definition/GitHub [Accessed 28 March 2023]

Kanbanize (2023) “What is Kanban? Explained for Beginners.” [Online] Available: https://kanbanize.com/kanban-resources/getting-started/what-is-kanban [Accessed 28 March 2023]

Obsidian (2023) “About Obsidian” [Online] Available: https://obsidian.md/about [Accessed 28 March 2023]

Rust for Linux (2023) “Rust-for-Linux/linux: Pull Request 884”. [Online] Available: https://github.com/Rust-for-Linux/linux/pull/884 [Accessed 28 March 2023]

Trello (2023) “What is Trello: Learn Features, Uses & More.” [Online] Available: https://trello.com/tour [Accessed 28 March 2023]

Wikipedia (2023) “Obsidian (software)”. [Online] Available: https://en.wikipedia.org/wiki/Obsidian_(software) [Accessed 28 March 2023]

Formal Meeting Forms

FORM A

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA
(To be completed ***before*** *the scheduled meeting)*

Student: Kyle Christie Supervisor: Paul Keir

Meeting Number: 1 (Formal for Week 2/3) Date/Time: 3/10/22 @ 1400

PROGRESS

Over the last month, the following tasks have been completed:

- I've worked on solidifying exact project, title, research questions.
- I've worked on research relating to OS driver differences, frameworks, how these relate to the project.
- I've worked on learning Rust specifically for Systems programming, have spent time studying how Rust manages memory. I have a diary with a short comparison between the same program in C and Rust.
- I've obtained a Raspberry Pi 400 and have configured/tested it for Linux driver development, I plan on using it for the final development task.
- I've found and taken notes on various articles that relate to drivers, rust and recent significant work/developments.

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

- None

AGENDA
1. **Provide summary of work/tasks undertaken since last meeting (12/9/22) [X]**
2. **Discussion regarding project specification form and final project plan [X]**

3. **Discussion on next major milestones/submission (interim report?)**
4. **Discussion on tasks to be carried out in meantime [X]**
5. **Any other business**
6. **Decide next meeting date**

FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

(To be completed ***after*** *the scheduled meeting)*

Student: Kyle Christie **Supervisor: Paul Keir**

Meeting Number: 1 (Formal Week 2/3) **Date/Time: 3/10/22 @ 1400**

MINUTES

The following tasks and issues were discussed and specific actions agreed:

1. Provided a summary of work/tasks undertaken since last meeting/recently.
2. Discussion regarding project specification form and final project plan
3. Discussed project deliverable and related points
4. Discussed use of short articles within project
5. Short discussion on next major milestones
6. Decided next meeting date
7. Discussed other business

PLAN

The following tasks and timelines have been agreed both for the next month and beyond:

For the next month:

- Finish project specification.
- Begin research into how drivers are written in Rust.
- Continue work, focusing on literature review.
- ..

Beyond the next month

- ..
- ..
- ..
- ..

FORM A

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA

(To be completed ***before*** *the scheduled meeting)*

Student: Kyle Christie **Supervisor: Paul Keir**

Meeting Number: 2 (Week 6/7) **Date/Time: 26/10/22 @ 1300**

PROGRESS

Over the last month, the following tasks have been completed:

- The Honours project specification form has been submitted, signed by both Supervisor and Moderator.
- The interim report has been started, Overview section complete with Literature Review in progress. On track for successful submission.
- I have successfully set up a QEMU VM with the latest Linux kernel with Rust features. I have tested this machine and its configuration.
- I have written an equivalent of the C char driver but in Rust on Linux-Rust VM, I have tested it and it runs without issue.
- I am starting to conduct research into writing a Generic USB mouse driver as a C driver, intending to use those resources to write a mouse driver in Rust.

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

- None

AGENDA

1. Summary/Showcase of work undertaken since last meeting (3/10/22) [X]
2. Discussion regarding end project deliverable – a Mouse driver. Is this suitable? Recommended backup? [X]

3. Discussion on next major milestones – Completing interim report and setting up Linux Kernel 6.1 and necessary tools on Raspberry Pi.
4. Discussion on tasks to be carried out in the meantime.
5. Any other business.
6. Decide next meeting date.

FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

(To be completed ***after*** *the scheduled meeting)*

Student: Kyle Christie **Supervisor: Paul Keir**

Meeting Number: 2 (Week 6/7) **Date/Time: 26/10/22 @ 1300**

MINUTES

The following tasks and issues were discussed and specific actions agreed:

1. Discussed current work/progress, showcased current work.
2. Discussed end project deliverable – decided on a Generic USB Mouse Driver with the LDD3 Scull Driver chosen as backup.
3. Discussed interim report layout and word count target.
4. Decided next meeting as Wednesday 9th November at 1300.

PLAN

The following tasks and timelines have been agreed both for the next month and beyond:

For the next month:

- Acquire additional storage for Pi, Configure Raspberry Pi with Rust for Linux Kernel 6 in order to start developing the mouse driver. Also install related tools for Rust kernel modules.
- Continue with and complete Interim report.

- Continue Research into writing a USB Linux Kernel Module.
-

Beyond the next month

-
-
-
-

FORM A

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA

*(To be completed **before** the scheduled meeting)*

Student: Kyle Christie **Supervisor: Paul Keir**

Meeting Number: 3 **Date/Time: 11/11/22 @ 1000**

PROGRESS

Over the last month, the following tasks have been completed:

- Interim report first draft has been submitted.
- Linux Kernel 6.1-rc3 has been successfully built in a test environment.
- Additional storage has been acquired and installed to Pi workstation.
- Additional reading material is being obtained in Linux Device Driver Dev Cookbook as well as a physical copy of Linux Device Drivers 3.

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

- In the test environment, an issue has arised with rust-analyzer which prevents Rust drivers from building. This has been noted in diaries and a successful solution has yet to be applied.
- The above issue has also froze research on Linux USB devices.

AGENDA FOR FORMAL MEETING (Example)

1. Summary/Showcase of work undertaken since last meeting (26/10/22) [X]
2. Discussion regarding main present issue i.e. Linux 6.1-rc3 and rust-analyzer [X]
3. Discussion on next major milestones – Building Kernel 6.1, related tools on the Pi and testing Rust driver on hardware, Start development on Rust module project deliverable. [X]

4. Discussion on tasks to be carried out in meantime, looking into the internals of Rust Linux. [X]
5. Any other business.
6. Decide/Discuss next meeting date.

FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

*(To be completed **after** the scheduled meeting)*

Student: **Kyle Christie** **Supervisor: Paul Keir**

Meeting Number: **3** **Date/Time: 11/11/22 @ 1000**

MINUTES

The following tasks and issues were discussed and specific actions agreed:

1. Short summary of work undertaken since last meeting
2. Feedback on interim report
3. Discussion and insight into Rust-analyzer problem
4. Quick discussion on next milestones, tasks to work on in meantime
5. Will email to arrange next meeting (post interim report)
6. ..

PLAN

The following tasks and timelines have been agreed both for the next month and beyond:

For the next month:

- I have decided to restart the VirtualBox test and start fresh as there are a lot of issues with the current VMs. I aim to be more careful with the new tests.
- Submit a second draft of the interim report with recommended fixes/changes to Paul
- Continue Research as necessary (USB capabilities of Linux kernel etc)

- Formally submit final version of interim report.

Beyond the next month

- Workstation pending, start development on Rust ‘production’ driver
- ..
- ..
- ..

FORM A

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA

(To be completed ***before*** *the scheduled meeting)*

Student: Kyle Christie	**Supervisor: Paul Keir**
Meeting Number: 4	**Date/Time: 2/2/23 @ 1100**

PROGRESS

Over the last month, the following tasks have been completed:

- Improving writing between interim report and dissertation.
- More industry figures have been contacted.
- More content written into dissertation.
- An attempt made to start development.

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

- Starting driver development – Rust USB library is not officially integrated and the integration in my own kernel does not seem to work.
- ..

AGENDA

1. Showcase of work undertaken since last meeting (11/11/22). [X]
2. Discussion of issue – Issues with Rust USB library and direction of final project deliverable. (there is more work from RustForLinux project on USB) [X]
3. Discussion on next major milestones – Creating and presenting project presentation, Working through dissertation writing. These will essentially make up the core tasks until end of project. [X]
4. Any other business.

5. Discuss next meeting date.

FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

(To be completed ***after*** *the scheduled meeting)*

Student: Kyle Christie **Supervisor: Paul Keir**

Meeting Number: 4 **Date/Time: 2/2/23 @ 1100**

MINUTES

The following tasks and issues were discussed and specific actions agreed:

1. Regarding the USB library issue: Use the USB contributors full repository. Continue to try working on this component, try communicating with the contributor to find help in solving any problems.
2. Discussion on dissertation writing, highlighting new sections and their placement within the report.
3. Discussion of new content that could be added to report to enrich existing sections.
4. Brief discussion on expectations of presentation.

PLAN

The following tasks and timelines have been agreed both for the next month and beyond:

For the next month:

- Presentation writing.
- Continuing work and investigation into the Rust for Linux USB contributions.

Beyond the next month

- Continue work on writing for dissertation/final report.

FORM A

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA

(To be completed ***before*** *the scheduled meeting)*

Student: Kyle Christie **Supervisor: Paul Keir**

Meeting Number: 5 **Date/Time: 3/3/23 @ 1400**

PROGRESS

Over the last month, the following tasks have been completed:

- Project presentation
- Continuing of dissertation writing

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

- N/A

AGENDA FOR FORMAL MEETING (Example)

1. Discussion of presentation, marking & feedback (22/2/23) [X]
2. Showcase of work undertaken since last meeting (2/2/23) – Effectively rounding out Lit. review, added in summary to bring points together [X]
3. Discussion on next major milestones – Completing final project report, carrying out any further research as required. (Was noted through presentation that development is effectively complete and that final report now takes precedence) [X]
4. Any further business. [X]

5. Discuss next meeting date. [X]

FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

(To be completed ***after*** *the scheduled meeting)*

Student: Kyle Christie **Supervisor: Paul Keir**

Meeting Number: 5 **Date/Time: 3/3/23 @ 1400**

MINUTES

The following tasks and issues were discussed and specific actions agreed:

1. Discussion of work undertaken since last meeting (2/2/23)
    - Next meeting will be after first draft submission, with focus on discussing said draft
2. Discussion of approaching development section in report
3. Discussion on approaching acknowledgements section
4. Discussion on job applications/aspirations

PLAN

The following tasks and timelines have been agreed both for the next month and beyond:

For the next month:

- Completing final report and related work as necessary

Beyond the next month

- N/A

Note 28_8_22

28/8/22 -- Notes on recent happenings

In my previous diary, I noted that I'd reached out to Jon Blow and Dave Plummer. Jon got back to me quite quickly and this was noted in the same diary. Dave got back to me a few hours after I'd finished work.

In his email, He told me that he had mostly worked with cache software & filesystems and hadn't actually worked much at the driver level. He recommended a book called 'Windows 7 Device Driver' by Ronald D Reeves. It's unfortunate that he turned out to not have done much driver work and didn't offer his opinion but I am at least thankful he showed me that book. In the same way that I have done and can do with Linux, I now have an avenue to look back at how Windows drivers were developed and compare this to documentation and such on Windows 10 and 11.

A note that I wanted to take but didn't manage to; At some point I'd really like to conduct some research to find out if device drivers are developed under a specific/custom methodology or if they even utilise a methodology at all? I suppose this would change between vendors and open source etc. I just want to make sure I'm encompassing all/most areas within drivers and that all bases are covered in terms of software. My thoughts at the moment are that; realistically if drivers used a methodology they'll likely just use similar methodologies to those within Software Development such as Agile.

I do think that some changes will be made to the project and maybe the scope will grow a bit. This is mostly because of the messages I recieved from Jon Blow. His points were really interesting and, in terms of his feedback on some of my project, I don't know if I can quite agree but I can see his overall point cc shooting too low.

I haven't worked on the project today or since my last diary. At first I took a days break but I wound up busy with work and other things. Tomorrow I'm going to try and get back into my working habit.

Another thing I'd to cover! Specific/Worst security issues caused by or related to drivers. I think to continue being in-depth, it would be necessary to discuss specific security issues caused by drivers as well as the potential security issues that might arise in the future.

So, another summary of points of research/points to be made & discussed in the paper.

- Explanation of drivers and their functionality
- Methodologies used within driver development
- How they differ between vendors/Operating Systems
- Discuss various problems regarding drivers and related
- Specific security issues
- Developments etc that have helped to solve these problems

Note 9_9_22

9/9/22 -- Feedback

(12:12)
Here are some feedback notes from the project supervisor;

(Honours Project Update: Early September)

"Not an area that has much focus" - it's not an a field that is often 'in the limelight'

This phrasing feels much more suitable and formal. I might need to adapt a teeny bit further but I'll use something like this going forward.

Potential task: Quantifying number of device driver engineers

This is an interesting task, it would make sense but how exactly could I apply this? Potentially by approaching companies and asking about their driver/systems departments? I suppose that for open-source vendors, I could look back at Git/GitHub commits and specifcally look for commits related to drivers? I'll have to brainstorm exactly how this would be incorporated but I think it would link quite well to the above point - not a field that's in the limelight often

"... also work in a harsh environment" - the word harsh is a little too informal

While writing my reports, project management etc. I'll need to really take care and make sure that I'm writing in a formal matter. I'll also need to make sure that I don't over-explain concepts etc and that I'm as clear and concise as possible.

Research tools don't expect to get much use. It's more about the ideas and conversations.

Yes, I had semi thought that too (especially when I saw that WHOOP was essentially dead in the water) but I think my wider point was that these are all potential routes to fundamentally improving drivers. Maybe I was/am being a bit naive. I *can at least use these tools that I've researched and read about to discuss how we might improve drivers. Such as if Dingo saw widespread use. We have a lot of options.*

Mentioned Valgrind, Helgrind, AddressSanitizer - "I always measure a tool's popularity as to whether Ubuntu has an APT package for it."

These are new tools that I've never heard of before. I'll need to really look into them and how widely used they are etc. I might be able to incorporate them into the project also.

What is Apple's solution? Will Apple's solution be applicable to other Linux flavours? Can you apply it to Linux?

Originally, Apple utilised just Kernel extensions (I believe in a similar way to Linux kernel modules) but in the last few years decided to change this and introduce 2 more categories of 'extensions'. System extensions, used for features that need kernel level co-operation and DriverKit extensions, used for handling custom hardware. These new extensions run in user space and Apple claims there's a lot of benefits to using their new extensions over kernel extensions. Here's the talk I watched as part of my Research https://developer.apple.com/videos/play/wwdc2019/702/ and obviously you'll be able to look at my notes too. The issue with Linux is that it offers a user-space option for writing drivers but it seems to be useless. There are benefits but these seemed to be outweighed by several drawbacks so it's generally better for live drivers to run in kernel space.

The ideal C-to-Rust driver conversion to demonstrate might be one where the original C code shows the *potential* for a memory error which Rust would not permit.

I'd been trying to figure out myself how to navigate writing a C and Rust driver and what the comparison would be. This makes sense over the original analysis etc that I'd planned and - I think - noted about somewhere so I think I'll focus more on showcasing Rusts strengths over C and demonstrating this through code and what each language allows etc.

(12:38)
Today I want to work on the following tasks

1. Read into FreeBSD and its drivers

2. Write another Rust Systems lab *or* find a new systems concept that I implement in Rust.
3. Work on the project structure/figure out exactly what it is that I'm going to do.

Right now, I'm going to work on task 1. Later on tonight, I hope to come back and complete tasks 2 and 3. I think task 3 is something I might need to work on with Paul or at least get feedback on.

(22:46)
Earlier today, I took notes on freeBSD. I'm going to continue working on this task and then do the prep for task 2. This way I can then work on task 2 tommorrow morning before work.

(23:02)
I've completed notes from that specific page on FreeBSD drivers, I'm satisfied that I've completed notes and might return to gather in-depth information as necessary. Now, I'm going to look into systems tasks that I could write in Rust. I'd like to find something new or fun to work on.

Note 2_10_22

2/10/22 -- Return from Holiday

(19:24)
Today I wrote out sheet A for the formal feedback and management meeting tomorrow with Paul. I have minutes ready for the meeting tomorrow. Tonight, I need to;

- Read up on emails between myself and Paul and remind myself of anything I need to do. [X]
- Read up on finalised project plan, research questions so this can be used for the project spec form [X]
- Read and take notes on any relevant articles that I haven't already [X]

For myself, I should place any important notes in this diary.

Emails

In my last update before the holiday, I did the following

- Reached out to Stephen Devine to act as moderator.
- Obtained a Rasp Pi 400 and tested the char driver on it.
- Spent time finalising research questions/the project - my title at this stage is "How can Rust be used to improve relability in Linux Device Drivers"

Tasks for coming back should be

- Finalising project details
- Completing project spec sheet
- Continue the project w/ a focus on the new layout

Paul's thoughts

- Research questions are good but consider breaking them down further - "Can Rust improve reliability of Linux Device Drivers?", to make research questions more challenging.

He also provided 2 articles and presentation slides.

Paul also sent out another email while I was away containing article, I'll need to gather all these articles up and work through them.

Article List

- https://www.zdnet.com/article/linus-torvalds-rust-will-go-into-linux-6-1/ [X]
- https://www.phoronix.com/news/LPC-2022-Rust-Linux [X]
- https://www.phoronix.com/news/Rust-Apple-DRM-Cube-Milestone [X]
- https://www.theregister.com/2022/09/28/is_it_time_to_retire_c/ [X]
- https://www.zdnet.com/article/programming-languages-why-these-developers-like-rust-in-their-cars/ [X]

Finalised project plan

I'll need to ask Paul tomorrow if he is happy with the layout as is, he didn't specify in any emails to me.

I'm happy with the first section, Driver issues. It makes for a good to the point opener.

The second section, Stuff tried, I think needs a bit of work/research poured in, in hindsight this was literally copy and pasted from its original note so I might pull it unless those improvements make a difference.

(20:54)
The third section, tools that might help, needs a bit of cleaning up (which i'm in the middle of doing) but should be fine when I'm done. I've re-ordered this section so that the strongest points come first which will make up most of the word count for that section (hopefully).

Section 4, Rust Linux Proposition, is being cleaned up too but should be fine also. After cleaning up, I'm pretty much happy. It is effectively decided that the end product of this project should be a Rust driver for a generic computer mouse on my RPi. Failing that, the backup could be a driver the simulates a device in memory? I should brainstorm this.

I don't really think Section 5, driver results, needs any changes.

I'm going to slightly redo the final section as its just another copy-paste. I've wound up totally re-doing it. This is much better and it means I can keep tabs on whats going on in driver and Rust world and make things as up-to-date as is feasible.

Closing out (21:09)

Tomorrow, I'll have a feedback/management meeting with Paul at 2pm. This note will be useful for that and, I guess, for continuing work on the project. Post project specification, I think the main task will be updating research where needed and getting the interim report out the way.

To completely finish the day, I'm going to write a draft of the project outline for the project spec sheet.

(21:38)
Finished - can work on improvements tomorrow.

Note 12_11_22

(17:21)
I am installing Ubuntu to a virtual machine to act as a new test. Ubuntu runs on my workstation which is the only device that I have managed to get Rust Linux working so far.

(17:39)
I'm going to use the GitHub repo of the Linux kernel so that I can simply git pull and update rather than constantly download new tar files etc

(17:43)
For this versions rust install, I decided to customise my install with the following options;

default host triple: x86_64-unknown-linux-gnu
default toolchain: 1.62.0
profile: complete
modify PATH variable: Yes

(18:15)
It seems that Rust isn't being built within Make?

(19:38)
Trying to build the kernel leads to errors. I have the same problems as in the old machine. I'll have to continue this tomorrow.

(20:45)
So I just decided to come back. I'm trying to figure out exactly whats going wrong and what's causing issues.

(20:53)
So I found in basically all these test instances that the rust option is hidden in make menuconfig. This documentation

states "The option is only shown if a suitable Rust toolchain is found (see above), as long as the other requirements are met. In turn, this will make visible the rest of options that depend on Rust."

I can at least confirm that rustc works on this machine.

(20:59)
The option issue is obviously due to the fact that I'm not yet running 6.1

(21:22)
I've been trying to compile a 32 bit kernel on a 64 bit machine, I've downloaded and am currently building 6.1-rc4 from kernel.org to see if it resolves issues. I'm also making sure to build as the root user.

(21:34)
It seems no matter what distro I try to use, I constantly run into issues this is really frustrating.

(21:36)
I moved back to working from the GitHub and realised I hadn't checked 64-bit kernel. I am currently re-building and hopefully this will provide better results.

(21:39)
Still threw an error. I hate this.

(22:04)
So now for some reason the Rust support option is showing, i'll need to figure out the boot error and I think - touch wood - everything might be smooth sailing.

Rust-analyzer at least works.

(22:11)
I think now I have the opposite damn issue with this test! Rust works but I don't think this kernel will actually boot goddamnit.

Note for compiling on Pi; Just get the kernel working then rebuild it with the stuff installed.

I'll try to get this working before I move to the pi.

(22:34)
So I'm going to start again, configure the 6.1 kernel and *then* configure rust. I'm also going to make sure not to totally screw my config.

(23:42)
https://askubuntu.com/questions/1329538/compiling-the-kernel-5-11-11

(00:49) (13/11/22)
I continue to let the kernel build. This is compiling a lot more and will hopefully give better results.

(01:13)
Kernel successfully compiled! Now moving onto install Rust and recompiling.

(01:53)
I should remember to run `make defconfig` which seems to allow the option of rust support! Now to quickly test and then run off to bed.

Rust-analyzer works
Rust doc works
Rusttest, I assume, works

Well first, I'll rebuild the kernel with Rust and can test code tomorrow.

(02:37)
Kernel has been recompiled, samples loaded but I can't seem to get a test driver running. A task for tommorow. I'm at least happy to have finally solved this problem.

(02:52)
I'd say everything works and I just need to figure out how make is used to build the rust code into .ko files. Hopefully documentation will pop up soon.

https://lwn.net/Articles/910762/
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/Documentation/rust/quick-start.rst

Note 28_12_22

(12:55)
Today, I'm going to start working on writing my Rust Mouse driver on the VirtualBox instance. By the end of today, I'd like a very very basic skeleton to be set up.

(13:58)
For some reason, the rustc compiler can't be found.... I have no clue how this happened but it seems I can't have a day of development work where there is some kind of roadblock...

It's really frustrating because rustc is clearly installed its just that the kernel build can't find it.

Currently rebuilding the kernel and we'll see if this fixes it.

(15:00)
I've decided to look for Rust driver repos on GitHub in order to test them to check whether I can actually compile a Rust driver on this VM instance.

https://github.com/alexandruradovici/rust_linux_drivers

(15:05)
Fairly certain that this VM simply can't build any sort of Rust driver. Trying to build with Cargo doesn't work at all and likely isn't intended to work. Trying to build as if it were a traditional kernel module (with Make and `obj-m`) does not work either and returns errors.

(16:07)
After taking a break and trying the repo driver again, I'm now seeing Rust errors.

(16:15)
I can confirm that Rust itself works so it seems I have some kind of Rust kernel build issue.... and once again the rustc compiler has somehow disappeared from existence.

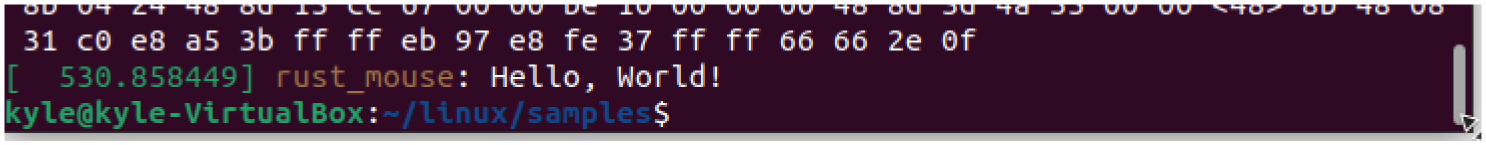
Rebooting and trying to build that repo driver once again provides weird results. I keep running make and it writes some kind of output but doesn't produce anything.

I'm unable to get any sort of result from the Virtual Machine. I find this very strange considering 2 months ago I was at least able to produce a .ko file (even if insmod would reject it).

(16:49)
On the QEMU virtualbox which I set up earlier in the project, I managed to build the Char driver in Rust by adding it to the samples list (Kconfig and Makefile). I'm hoping I can pull off the same thing in the VirtualBox instance and essentially force my driver to be built.

This will then mean for testing, I may need to rebuild the entire kernel or at most the modules though (if this works) I think it is my best option at the moment. Trying to build Rust modules in the same way as C drivers doesn't seem to work. At least, I wasn't able to get it properly working again. It still bothers me though that earlier in the project, I was able to get things building this way.

(17:02)
After enabling the samples and trying to rebuild, I ran into the same errors which were experienced earlier. After comparing a Rust sample that sucessfully compiled, I found that several issues with my code that are likely to be the cause of the previous errors. Fixing these issues has now allowed my kernel to build and I now expect the module to be able to run.

(17:09)


```
 31 c0 e8 a5 3b ff ff eb 97 e8 fe 37 ff ff 66 66 2e 0f
[  530.858449] rust_mouse: Hello, World!
kyle@kyle-VirtualBox:~/linux/samples$
```

After successful compilation and execution of the mouse driver (which at the moment, simply prints hello world). I decided to try the fixed code with a makefile and can confirm the build system has no problems. I should now be able to build my driver with no issues.

(17:18)
The driver is able to independently build with zero issues. I can now start providing the skeleton for my mouse driver.

I think before I do that it might be a good idea to figure out what I can do with Rust and what I'll need to do in unsafe C. The current Rust code available in the Kernel is limited and I previously established that the USB library is not yet available.

I'd imagine that a lot of the data structures and so on can pretty much all be kept in Rust so it might be that most of the functionality is programmed in unsafe C. I'll also need to look into using C within Rust as it has just ocurred to me that I have never worked with this feature before.

Will be good to also write up an explanation of how my driver works etc.

(18:22)
I worked from a very short tutorial and created a very basic project that links to C from Rust
I'm going to continue later by watching this talk which discusses embedding Rust in C/C++.
(19:57)
Working on RustConf 2018 Talk notes.

(20:57)
I finished my notes a short while ago and will be ending the day here.

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.

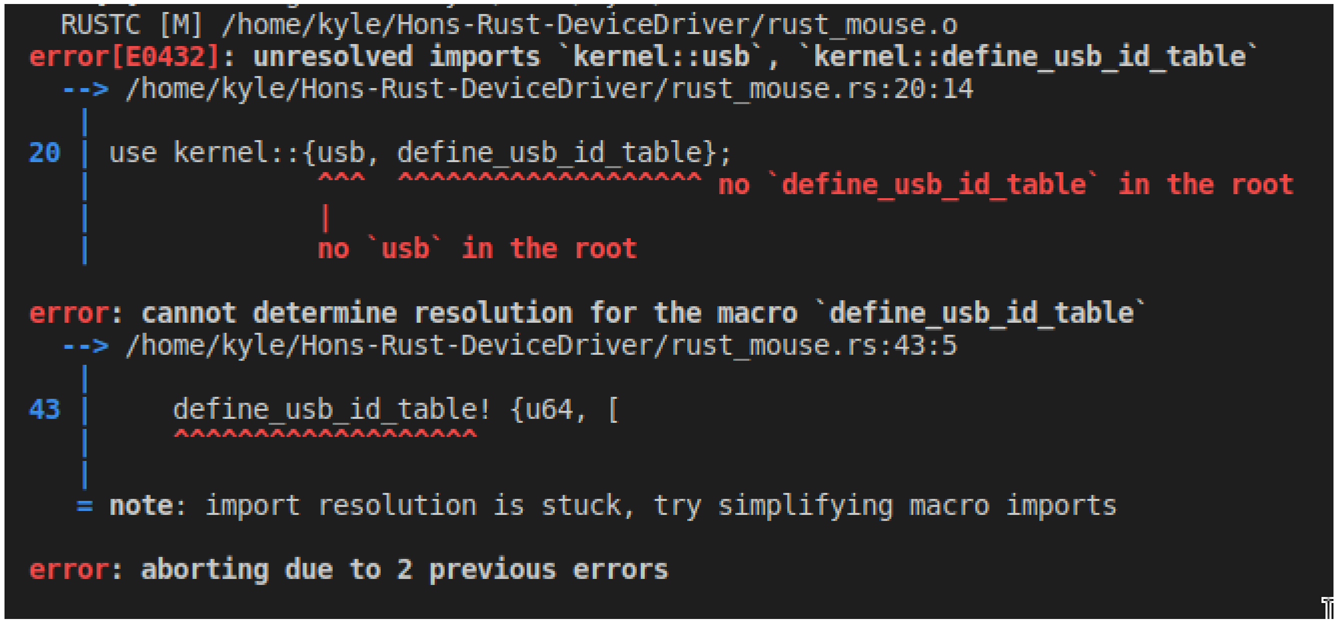
(14:23)
Rather than writing out my own C version of the driver, I spent some time yesterday looking for potential equivalents that I could either re-implement or use as inspiration for my own Rust driver. I might not get very far with this but any result can be used I suppose.

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.

Note 24_2_23

(17:52)
Dissertation word count currently sits at 4,230 words. I've completed the 'general concepts' subsection in the background. I think some more work should be carried out on the Garbage collection section, then the section on exo-kernel. After these, I feel I should move onto other sections within the dissertation.

Obviously, there has been a short gap in diary entries. Recently, most time has spent on creating my presentation for the project. On Wednesday (22nd), I presented to my supervisor. The next day, provisional marks were given to me at 85! I'm very proud of this and I'm seriously aiming to gain a first class for this project.

At the moment, I see development as 'finished' in terms of the project. If there are any serious developments then I may consider going back to developing but as of now, my main focus is completing the dissertation and making sure everything goes smoothly for the next month.

Note 2_3_23

(15:13)
Final deadline is under a month away for this project. It feels like yesterday I was only writing the proposal...

Today, I made progress on dissertation writing! Word count sits at 5,722. I'm trying to wrap up the literature review to then move onto other sections of the report. Within Lit review: Memory safety, I expanded a little bit on Garbage collection and created a new but related subsection 'Reference Counting'. To round out my literature review, I also wrote out a summary of the main points of each section and tried bringing them together to make some wider points/claims. I made sure to write the summary in a way that each paragraph links to the next but also links back to the report content and connects most if not everything.

I've already started planning out my development section and what I want to cover. Funnily enough, its the only section of my report that I *didn't* pre-plan. I've also been thinking about how to write my acknowledgements and I might discuss this with Paul tomorrow. I already made clear in my presentation Q+A that development is essentially finished but I may continue to pursue the mouse driver as a future project.

When I make my final submissions, I'll need to look into the lectures that Mark has put out to make sure my final report is as suitable as possible.

(15:20)
At the moment, I'm going to take a break from writing and work on paperwork for tomorrows formal management meeting.

Meeting 28-7-22

Kyle Christie
Paul Keir

14:00 to 14:58

Notes (as written on paper)

- Quite a rare subject
- Parallel programming
- Chandler Carruthers
  - LLVM
  - C++ Commitee
- Carbon
  - Doesn't have mem safety
  - Building on LLVM
  - Lean C++
- C++
  - Optimisation, undefined behaviours
- Rust
  - Dev idea works
  - Answers scientific question

Take an existing driver & improve

- Research rust and drivers
- Research by MS on Rust, taking it farther
- Tools are much more younger
  - Build on core ideas of rust
- C & Rust & Analysis
  - Good idea!!!

(#) of memory-related issues
- Proper mem mgmt

GPU used for normal calculations, GPGU

opencl
cuda |
| |
| |
V V
Write similar to C/C++ but not the same, a subset

<u>Software Fuzzing</u>
Fuzzing

- Reliably expose weakness
- Randomly mutate code
- Make syntax errors

- Simple brute-force method

Money & appetite
Very employable project

CodePlay bought over by intel

Which type of driver?

User-level driver?
- Client-level bluetooth reader
- Cliet-level interface
- C API's

<u>Ideas</u>
Arduino (read from Serial port?)
USB stick (pass through to VM)
(old computer hardware
that is useless)
Raspberry Pi (SSH)
Trade-off between VM & Pi

Mouse, keyboard, USB
Whatevers most fun

Quite likely to go ahead

Starting early isn't an advantage

Potential mods

| - Stephen

| - Joanna

|

V

not much input from mods

Meeting 12-9-22

Kyle Christie
Paul Keir

14:00 - 14:55

Monday's work for meetings

Discussing project, scope, ideas etc

Next step up from char driver - something like scull that 'simulates' a

`make VERBOSE=1` - allows insight into commands being run by makefile, take commands and break the process down by running the commands → this might allow me to create bindings into rust

making and/or looking into mistakes/vulnerabilities in rust

good reserach should always explain to peers

idea of research question
devise a question that doesn't have an obvious answer

user-study is possible

about whether ideas can improve performance and then benchmark

best to spend next few months to learn how to improve reliability etc
keep reading literature and see how it can link
see if you can measure safety before and after a problem

create as good a research question as possible

narrowing down research question will inform scope
what do you imagine you would create

final output can be driver that controls physical device

Project spec sheet

Don't rush proj spec doc

resolve essential research question

Moderator

Possible to ask

To-do

- Work on improving research question - as good a research question as possible
- I need to work on figuring the project out and knowing exactly what it is that I want to do
- Start project specification sheet, no need to rush - it can be finalised/completed when I come back from holiday

Meeting 26-10-22

Kyle Christie
Paul Keir

13:00 - 13:30

Happy with Generic USB mouse.
Scull driver as a backup.

Create some sort of indication of how time will be spent - gantt chart, some sort of GitHub feature.

Can use commit info from repo as an Appendix, optional for interim report - may be more suitable for a final submission.

Meeting 11_11_22

Interim report feedback
Font too large on front sheet
Different pages for ToC, ToF, Abstract
New page for a new section
Code excerpts should use a text-based render - different text fonts from machines might cause issues down the line, more convenient to copy code excerpts
Keep less than half the page full with code excerpts. Around a third or a half. Code is all saved in repo, might want to tag the repo to insert into dissertation

Abstract
Start with Memory safety
Can try the abstract the again
Line up font on abstract

Spelling mistake
Page 5 immediately

Diary has a good home in the interim report but won't translate into dissertation
Management report could stand separately from dissertation and could hold diaries as appendices - Github commit history?

Be sure to insert project spec as appendice

Meeting 2_2_23

- Improved writing from interim report
- Contacted more industry figures - this is a dedicated subsection within my background section which details who I've contacted and their feedback/advice etc
  - Miguel Ojeda (resp)
  - Alex Gaynor (resp)
  - Asahi Lina (no resp)
- More content written into dissertation, sitting at 3,900 words.
  - General Concepts section - short notes on key concepts; Kernel, User/Kernel space, etc
  - OS Drivers and differents - written about Linux, Windows, Apple and FreeBSD
  - Currently expanding memory safety section with garbage collection, more to be added as per feedback
  - Expanded on Rust section - Discussed Rust for Linux & its background, Criticisms of Rust

Rust USB

- Previously had issues building, example documentation seemed unreliable
- More work has popped up with a full sample able to build though none of the work has been merged (as of writing)
- As discussed, this is an issue which affects the project deliverable

As briefly discussed, would it be suitable to swap the marking for Literature review and Development sections.

Main focus now

- Starting presentation (remember to ask expectations)
- Contuing dissertation writing, post presentation this will be the main task

---

A-class submission for interim report.

Possibly use Yakos full repo.
Should just open an issue to ask about use.

Send Paul Yakos repo link after meeting.

Maybe a modular driver with USB and for Mouse.

Communication might help solve the problems.

Don't let the problem stop me from trying to work on the project.
Don't give in but time running out isn't bad

General concepts could go very nicely in background section.

Haskell - Glasgow Uni, Haskell Currie. No functions allow I/O. MONADS. No global variables, everythings pure. Similarly has backdoors. Should make a comparison between Rust and Haskell. Very comparable.

Happy to hear every and re-state project for presentation. Explain to Moderator.
Structure can be similar to dissertation. Tell what you want to tell.

Meeting 3_3_23

- Continuing diss. writing
    - Finished General Concepts
    - Finished my section on garbage collection
    - Added a section discussing reference counting
    - Wrote and finished exokernel section
    - Added a summary section to the end of the literature review - connect the wider points and re-summarise.
    - Moving onto development section
- Next major milestone - continuing work as is and completing final report

---

Don't overthink it, don't hide reader from bad news.

Final meeting after first draft submission to discuss said submission.

Development:

- Write exactly what I did - produce more text than expected
- Code excerpts - be picky, take up less than half of the page - each one smaller than that
- Try to use a text figure rather than a jpeg
- Ideally, use text and keep everything consecutive. Good idea: size would be about same size of document text

Ask about acknowledgement.

- Just thank people.
- Anyone
- 2/3 sentences

- Dedicate the document to them

James reirden - drone project
Should consider codeplay, employees have went onto big companies - a lot of work with compilers, LLVM. C++ Clang is a big part. Maybe look into building and slight modifications.
A lot go straight to AMD

High-performance Computing, code running as fast as possible. Starts with little C++ programs. MPI lang, hard to learn on own. High performance computing @ Edinburgh. OpenMP, pragmas. MPI makes code run very quickly.