

Proposal for Rust integration on NuttX

Erick Silva

ericksilvarod@gmail.com

This document was written based on the topics suggested by the “Applying for GSoC” section on [this](#) Apache Software Foundation’s page.

About me

I’m a last-year Control and Automation Engineering student at CEFET/RJ, Rio de Janeiro, Brazil. During the past four and a half years, I was deeply involved in the IEEE Student Branch’s RAS chapter in my university. As a member, I was part of the Trekking Pro project, working on a PCB that integrated a PIC from the 18F family and a Raspberry Pi. Now I’m leading the mini-sumo project and also act as a programming team member, responsible for writing code in C to the 18F PIC family. I’ve also been a member of the Electronics team, projecting PCBs, and Mechanics, working with the SOLIDWORKS software.

In my professional experiences, I was a Patent Intern for one of the largest IP offices in Rio and a year ago migrated to a startup where I work with embedded development and circuit prototyping. I think my first experience was really important for me to understand better the technical way of writing and also enlarging my interest in documenting steps, an ability I believe will be very important in this work. My current work experience on the other hand is what is giving me the tools to be in this project. I work mostly with the Arduino and C++ language but also got in touch with nRF52’s SDK, which is written in C. Most of this info, in a bit more detailed view about my career, can be seen in my [LinkedIn profile](#); also, a part of my non-official work is available in my [GitHub account](#).

Background

During the preliminary research, I’ve seen many times users saying how much of an incredible idea it was to join an RTOS and Rust, like in [this Reddit post from 4 years ago](#), [the issue used as a base to this work](#) and [another Reddit post that expands more the idea of Rust + Nuttx](#). I’ve also had the chance to look into some of the existing work of integrating FreeRTOS, like [this crate](#) for example.

When we understand that Rust has become a really relevant language, said to be C++ in new clothes, it makes a lot of sense to think about joining NuttX, an RTOS able of amazing applications like audio, robotics, and smart gadgets, with the said language. Rust claims to have as its core performance and security, both very interesting characteristics to build applications. Therefore, the aim of this project is to enable the development of NuttX applications using Rust.

Regarding the previous work done, the only thing that appeared on a first look was a work mentioned both in [the issue used as a base to this work](#) and the [Reddit post that expands more the idea of Rust + Nuttx](#), related to Yoshinori Sugino, which was able to run a multithreaded web server, porting the Rust libstd to

NuttX/Cortex-M4F and prototyping a simple web server with it, all available [here](#). I intend to contact Yoshinori in order to get in touch with all the work and possible documentation done by him. As presented in the last Reddit post referred to above, I understand that this project aims to bring support for Rust's standard library in order to fulfill the integration.

I can't measure properly either the amount of non-compatibility issues that I might find developing this project or by consequence, the time that it would take to make the full integration, considering only one person in the team. I've tried to look after [freertos.rs](#) commits but had no luck. With that said, the next sections will speak on my perspective of what I need to get tested and coded in order to finish GSoC 2021 with as much development as possible.

The issue is said to have 672h needed hours, which would take me 22 weeks of 30 hours of work to fulfill, I'm not sure if this first dimension of time was made thinking about someone who already understands both Rust and NuttX deeply. In spite of everything, I believe that a good source of knowledge about the issue and its difficulties might be looking into how this process was made for languages like C and C++.

Design/description of work

A good point to make is that I'm not an advanced user of either NuttX or Rust, but I think this project is a wonderful way of not just learning these relevant technologies but also contributing to two very big open source communities. With that said, I'm truly available to push myself within these two months before the coding starts.

Considering it, my first task will be the **pre-coding and ambiance**, which already started in April, and consists of getting involved with the language, the OS, and their communities too. I'll put the end of this task two weeks after the coding starts since when the real work begins, more study will for sure be needed. After that, I plan on taking a whole week to look into what was done for the previous languages, and also Yoshinori's work. I want to document this last one in the said week too if it wasn't already done. This phase will be called a **previous work review**.

Assuming that we'll have all the pieces needed and a roadmap to the integration by then, the next seven weeks will be made of a cycle containing **two weeks developing the integration and one week for documenting and bugfix**, this will be done twice. The last part is optional, which will consist of creating an app with Rust for NuttX.

These three weeks of documenting are a key factor in my opinion, for us to make this integration reliable and scalable to as many users as possible.

Results for the Apache community

As said above, many users think that integrating an RTOS with Rust is a very nice combination. If we manage to do this task with all the above-mentioned milestones, I believe this will serve as a good platform for many applications to come.

Deliverables and Scheduling

It seemed more practical speaking of these on the same topic. There will be a value delivery by the end of each period mentioned above, as follows:

- first period (2 weeks): General integration design and Roadmap;
- second period (1 week): documentation of previous work done in Nuttx;
- third period (2 weeks): first integration code;
- fourth period (1 week): first integration documentation and next steps;
- fifth period (2 weeks): second integration code;
- sixth period (1 week): second integration documentation and closure; and
- seventh period(1 week): Nuttx app made in Rust (optional test).

The work schedule can be advanced or delayed, depending on how the **pre-code and ambience** period goes. This can also be seen in the project graph below, in which the weeks with deliveries are highlighted in strong orange:

	April	May	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10
Pre-code and ambience												
Previous work review												
First integration period												
Integration review*												
Second integration period												
Application coding												
* documenting + bugfix												

Other commitments

As said before, I'm an intern of embedded development in a startup here in Rio. I'm currently working from home, with no perspective of going to the office in person. My schedule consists of 30 flexible hours a week, which most of the time gives me the freedom to choose when to spend my time working. In addition to that, it's perfectly possible to ask for relief of some responsibilities of development since this project can be very useful for the study of new technologies to my professional side and to the business too.

Given that, my approach would be to work for 8 hours a day, equally divided between GSoC and my internship. I believe it's a very reasonable amount of time and I'm also able to ask for a vacation period if we see in advance that the deadline is getting too tight. Regarding the university, I'm currently in my last but one period, finishing the most complicated subjects there are left. In the next one I'll

only have two more subjects to study, that aren't that complex, and also the presentation of my thesis, which is already written; additionally, my classes might start in July, after a month doing this project. To summarize, my biggest concern will be to coordinate internship and GSoC, which I think is very doable.

In addition to that, there's a possibility the research group I'm part of at the university will attend to ICUAS, the International Conference on Unmanned Aircraft Systems, in Greece. This event is scheduled to happen from the 15th to the 18th of June. Our travel is still uncertain since we're still waiting for the paper's approval and governmental funding. Giving all that, I don't think our presence is that probable, but if it does happen, I will gladly compensate for my days off in the following weekends.

Community engagement

I plan on submitting all deliveries to community review, which will be crucial to fulfill the aforementioned reliability and scalability. I've already interacted with some members of both the Nuttx and the Rust community. I'm very excited about the first information I was able to gather and plan on keeping this flow through the whole project.