Code Formatting and Style Check for RTEMS SuperCore (score)

Google Summer of Code Program 2021 Project Proposal

Meh Mbeh Ida Delphine
idadelm@gmail.com
University of Bamenda
Bamenda, Cameroon
#rtems Freenode: idadel
Github: @idadelveloper
Twitter: @meh_ida

Project Abstract

The supercore also known as score provides services for all APIs and for the core parts of RTEMS, there is no automatic checking of the coding style. Therefore for people new to RTEMS, it can get tricky to get it right when sending patches. This project will improve the situation and work towards automatic style and formatting checking for RTEMS score. By doing so, it will be able to tell if patches need changes before being sent for review.
This project is aimed at creating a tool for automatic style checking of RTEMS. It will be achieved by firstly finding a code checker or "formatter" that matches RTEMS' coding style, writing a script built into git commit hook that can run over patches before committing or submitting via email, as well as creating a standalone tool in rtems-tools to run over the entire codebase (excluding some parts) checking for any mismatches in the coding style. A script similar to checkpatch.pl of Linux will also be added to check whether patches will need changes before being submitted.
The code checker tool to be used will be clang-format since it is the closest whose output matches RTEMS coding style and the goal is to find a combination of clang-format settings combined with changes to the RTEMS style and changes to clang-format such that the output with the right settings matches the RTEMS code style.

Project Description.

What's missing?

There is no automatic checking of the coding style for the core parts of RTEMS. As a result, people new to RTEMS hardly get it right when sending patches. The RTEMS score format is specific to this project and a "formatter" needs to be taught how to handle the format.

It's importance

This project will improve the situation and work towards automatic style and formatting checking for RTEMS score. By doing so, it will be able to tell if patches need changes before being sent for review or applied.

Why this project?

I look forward to sticking around the RTEMS community for some time and that means I will have to make changes to the codebase and submit patches from time to time. Going for this project will make me understand and master the RTEMS coding style as well as understand the build system such that I will be able to make quality contributions.

Project Deliverables

GSoC Timeline: https://developers.google.com/open-source/gsoc/timeline

June 7 (coding begins)

- Set up my RTEMS Development environment and must have done the HelloWorld example.
- Created a GitHub repository for RTEMS so that my code can be reviewed by mentors.

- Understand the RTEMS coding style.

July 12-16 (Midterm Evaluation)

- Convert some files based on clang-format and submit patches.
- Add a standalone script in rtems-tools.git that can be run over rtems.git which creates a report about style problems. Configured the script to ignore some files or to add exceptions to the style rules for certain cases.
- Produce documentation on how the tool works.

August 16-August 23 (Final Evaluation)

- Test to make sure the tool works on all major development platforms for RTEMS (Linux, FreeBSD, Windows, MacOS)
- Write a script built into the git commit hook that can run over patches before committing or submitting via email.
- Add a script similar to Linux "checkpatch.pl" that could check whether patches would need changes before they are submitted and submit related patches. (this might take a little more time)

August 31 (Final Results Announced)

- Make improvements or add more changes to the script similar to checkpatch.pl

Post GSOC

- I will go to the RTEMS issue tracker and find issues I could work on.

Proposed Schedule

March 29 - April 13 (Application Period)

- Read the RTEMS style guide documentation - https://docs.rtems.org/branches/master/eng/coding-conventions.html
- Ask questions on the devel mailing list to understand my project.
- Submit my first patch.

April 14 - May 17 (Acceptance Waiting Period)

- Find a code checker or “formatter” that can produce results that match the RTEMS

coding conventions.
- Test clang-format on some files on RTEMS.
- Prepare a list of style/formatting differences between the tool output and "new RTEMS Style" and start a discussion to find a way forward.

<u>May 17 - June 6 (Community Bonding Period)</u>

- Convert some files using clang-format that matches RTEMS style and submit patches.
- Test style changes by running the full RTEMS testsuite to make sure they do not cause any implementation bugs.

<u>June 7 - July 16 (First Half)</u>

- Getting clang-format to match RTEMS coding style and testing.
- Add a standalone script in rtems-tools.git that can be run over the rtems.git which creates a report about style problems.
- Configure the script to ignore some files or to add exceptions to the style rules for certain cases.
- Test to make sure the tool works on all major development platforms for RTEMS (Linux, FreeBSD, Windows, MacOS)

<u>July 17 - August 23 (Second Half)</u>

- Produce documentation on how the tool works.
- Write a script built into the git commit hook that can run over patches before committing or submitting via email.
- Add a script similar to Linux "checkpatch.pl" that could check whether patches would need changes before they are submitted.

<u>Future Improvements</u>

- Make sure the tool works on other parts like posix, rtems, sapi and libcsupport.

<u>Continued Involvement</u>

After GSoC, I will love to stick around learning more about RTEMS while submitting patches to issues I find. Hopefully, I become a junior software engineer with RTEMS.

<u>Conflict of Interests or Commitment</u>

No conflicts of interest or commitment.

<u>Eligibility</u>

Yes. I am eligible.

<u>Major Challenges foreseen</u>

- Fixing any differences between clang-format output and the new RTEMS coding style.

- Integrating clang-format for automatic style checking.

References

- https://devel.rtems.org/ticket/3860
- https://docs.rtems.org/branches/master/eng/coding-conventions.html

Relevant Background Experience

- For over a year now, I have been consistently coding in Python and considering the fact that this project requires mostly Python skills, I have confidence I am up to the task. Also, I have been an Outreachy intern for the Yocto Project which is an open-source collaboration project that helps developers create custom Linux-based systems for embedded products, regardless of the hardware architecture. Being an intern for the Yocto Project sparked my interest for embedded systems making me want to get involved and learn more about them. My git skills were highly improved, I learned about patch submission as well as open source best practices. I believe with this experience, I'll be able to work on this RTEMS project.

Personal

I study at the University of Bamenda in Cameroon precisely at the National Higher Polytechnic Institute. I am in my second year and majoring in Computer Engineering.

It is amazing how embedded systems are being applied in communication, transportation, space, and many other areas and it's interesting to know how the operating system is being programmed.

I learned about RTEMS as I was lurking the Organizations page of GSoC looking for embedded systems-related organizations I can contribute to. I happened to come across RTEMS which I had never heard of, read about it and it excited me after reading what it is all about.

Experience

Free Software Experience/Contributions (optional):

- Added SPDX   headers to Yocto Project source files – https://lists.yoctoproject.org/g/yocto/message/52226

- Added HOMEPAGE and DESCRIPTION for Yocto Project recipes - https://lists.openembedded.org/g/openembedded-core/message/148841

- Updated Yocto Project license configuration file to match SPDX names - https://lists.openembedded.org/g/openembedded-core/message/143372

- Enhance Yocto Project license tracing - https://lists.openembedded.org/g/openembedded-core/message/149556

- I am currently a Google Developer Student Clubs lead of my university and I have hosted a session where I shared tips on getting the Outreachy internship and also getting started with contributing to open source. Here’s the link to my presentation: https://docs.google.com/presentation/d/18eU9f85StxWfEjC7_WK2CgIqlc__T59pGU8wFmNT_yI/edit?usp=sharing

Language Skill Set

● Python –	Intermediate
● JavaScript –	Intermediate
● Java –	Intermediate
● C/C++ -	Beginner

Related Research and Work Experience (if any):

- I was an Outreachy Intern for the December 2020 to March 2021 round with the Yocto Project

Reference Links and Web URLs (optional):

- Personal Blog: https://idadelveloper.github.io/blog/
- GitHub: https://github.com/Idadelveloper