Posix Asynchronous I/O (AIO) Implementation

Google Summer of Code Program 2024 Project Proposal

Alessandro Nardin
ale.daluch@gmail.com
Università degli Studi di Trento
Trento, Italy
tel. +39 3280854655

Project Abstract.

Portable Operating System Interface (POSIX) is a set of standards that are defined by the IEEE society. It defines various APIs, command-line interfaces and utility interfaces to maintain software compatibility between various operating systems.

POSIX Compliance provides maximum flexibility to write applications that can be executed on various platforms and increases the portability of code. This project will improve the compliance to POSIX standard of the RTEMS real-time operating system, implementing missing functions in the asynchronous I/O interface.

Project Scope.

Medium (approx. 175 hours).

Project Description.

The aim of this project is to refine and complete the implementation of the Asynchronous I/O interface (AIO). At the current state, the implementation is lacking some methods and functionalities.

The functions that are fully supported are:

- aio_read()
- aio_write()
- aio_error()
- aio_cancel()

The functions that lack some functionalities are:

- aio_fsync() : does not support O_DSYNC as value for op parameter.
- aio_return() : if the AIO operation incurred an error, the method should return -1 and set errno to indicate the error, the current implementation only returns the return_value present in the AIO control block.

the functions that are missing are:

- io_listio() : implemented as a stub that returns -1 and sets errno to ENOSYS.
- aio_suspend(): implemented as a stub that returns -1 and sets errno to ENOSYS.
- aio_init() (this method is not part of the standard functions but is an extension

provided by GNU C library)

The project has been thought to be done in increments, so various patches will be submitted during the development. For testing and development I will use the erc32 architecture.

Project Deliverables

- **May 27 (coding begins)**

At the beginning of the coding period, I aim to have a working setup that will allow me to code and test efficiently. I will work to acquire a comprehensive understanding of the area I will need to work and I will dedicate effort towards crafting a detailed plan delineating the approach for project execution.

- **July 8 - 12 (Midterm Evaluation)**

In the first half of the coding period I will focus on the following deliverables:

- Refactoring of the AIO processing. The aim is to create various helper methods that would handle the processing of the various type of requests, making the server thread method more readable.
- Correct the deficiencies found in the existing methods.

- **August 19 - 26 (Final Evaluation)**

In the second half of the coding period I will focus on the following deliverables:

- Modify the message structure between the aio_ functions and the server to include different messages. Specifically one for all aio_ operations and a user provided count for lio_listio.
- Modify the server thread to make it capable of managing a list of AIO operations rather than being limited to a single one.
- Implementing lio_listio() method, including the generation of a signal upon list completion if necessary.
- Implementing aio_suspend() method.

Tests will be written to ensure the correctness of both of the implementations.

- **August 31 (Final Results Announced)**

During this timeframe, I will dedicate my time to enhancing the documentation to ensure that all the features I introduced are adequately documented.

- **Post GSOC**

After the conclusion of GSOC, my intention is to continue my contributions to RTEMS. At the very least, I aim to maintain the code as needed, but I would like to actively address additional issues within the project.

Proposed Schedule

March 14 - March 24 (Application Period)

During this period, I will focus on my proposal by thoroughly studying AIO documentation and analyzing the current implementation to identify any missing functionalities, eventually a ticket will be filed for each deficiency found. A list with all the issues will be put in the project description. I will then draft my proposal and submit it for review to potential mentors, incorporating their feedback and suggestions.I will also set up an effective working and testing environment by installing the RTEMS tool suite and the erc32 BSP to ensure my ability to effectively work on the project.

March 26 - April 22 (Acceptance Waiting Period)

During this period, I aim to familiarize myself with the RTEMS contribution process by actively engaging in resolving smaller issues within the project.
If not done before I will also proceed to file tickets about the various deficiencies found during the application period.

April 23 - May 23 (Community Bonding Period)

During this period, my focus will be on project documentation and research. I will dedicate time to study all available documentation regarding the current implementation state within RTEMS. Additionally, I will delve into the POSIX standard specifications relevant to the functions I need to implement, ensuring a solid understanding of the requirements. I will engage with the RTEMS community through the Discord server and the mailing list communicating with mentors and previous students who have undertaken similar projects, gaining insights into potential challenges and effective strategies for overcoming them.

May 24 - June 26 (First Half)

During this phase, I will focus on correcting deficiencies in the existing methods and on preparing the ground for subsequent modifications. As the project is structured to progress in incremental steps, I plan to submit two patches during this period.

One patch will focus on the refactoring of the AIO processing. Currently the aio_handle() method incorporates both the thread logic and the request processing, this affects the code readability. My plan is to create a helper method for each type of request. That will allow for a shorter and more readable server_tread().

The other patch will focus on solving the issues in the existing methods. I will work on the aio_fsync() method by adding support for the O_DSYNC value for the op parameter. I will also work on the aio_return() method by making it set errno in case of an error.

The code produced will be discussed with the mentor to ensure it is up to RTEMS standards.

June 27 - August 23 (Second Half)

The main focus of this part of the coding period will be the implementation of missing methods.

Before I start implementing the new methods I will submit another patch, this patch will incorporate the modifications to the message structure and likely some modification to the server thread.

The modifications to the message structure between the aio_ functions and the server are made to include different messages. I plan to add one for all aio_ operations and a user provided count that will be later used for lio_listio. The modifications to the server thread

will be made to ensure that the server thread is able to process a list of requests. By a quick look at the code it seems that some work in this direction has already been done but a double check is necessary.

After that I will start the work on implementing the missing methods, specifically aio_suspend() and io_listio(). These two methods will be developed sequentially, i will start with io_listio() and then aio_suspend(). I will submit two patches, one for each method.

As also stated above, all the code produced in this phase will be discussed with the mentor to ensure it is up to RTEMS standards.

Future Improvements

After the coding period concludes, my plan is to document any new functionality that has been added to the project. In the event that the project is completed earlier than anticipated, I intend to keep working on the project by working on related tasks.

An issue that could be addressed is the implementation of the aio_init() method, that I have not included in the main project since it's not a method defined by the POSIX standard but part of an extension provided by GNU C library.

After this i would continue working on other issues related to POSIX compliance, like those listed in ticket #2966.

Continued Involvement

In the future, I am committed to continuing my contributions to RTEMS, and I am open to exploring different areas of interest within the project. As I aspire to pursue a career in embedded systems, with a particular interest in the space field, I recognize the immense value of contributing to RTEMS.

Conflict of Interests or Commitment

The only conflict is with my final examinations, which will take place in three sessions during June, July, and September. I don't plan to take any exams during the July session in order to focus solely on GSoC. I do not have specific dates for the exams yet, but considering past sessions I will likely have written exams in the first half of the month with oral exams in the second half. Considering that I will likely have reduced availability in the month of June. During the weeks occupied by the exams, I plan to dedicate around 15-20 hours to coding for the project. During the other weeks, I will be completely free, so I will be able to put in more work, around 25 - 30 hours. Considering that the coding period lasts 12 weeks a conservative estimate of the time i can allocate to the project would be the following:

- 6 weeks of reduced availability (6 * 15 = 90 hours)
- 5 weeks of of full commitment (5 * 25 = 125 hours)
- 1 week of unavailability (0 hours)

The sum amounts to 215 hours, considering also an eventual week of unavailability, that should allow me to comfortably complete my project and possibly do some additional work. As I'll have reduced availability for GSOC during the initial part of the summer, I've chosen to allocate more tasks to the latter half. Should I complete the workload for the first half ahead of schedule, I'll proceed with tasks scheduled for the second half in advance.

Eligibility

I confirm eligibility

Major Challenges foreseen

As this is my first involvement in a project of this scale, I anticipate encountering some challenges initially in understanding the appropriate approach for developing and submitting contributions. To avoid being slowed down by this during development, I plan to tackle simpler and smaller issues during the acceptance waiting period, aiming to familiarize myself with the RTEMS contribution process.

References

- aio(7) Posix specification Linux man page
- aio_read(3) Posix specification Linux man page
- aio_write(3) Posix specification Linux man page
- aio_fsync(3) Posix specification Linux man page
- aio_error(3) Posix specification Linux man page
- aio_return(3) Posix specification Linux man page
- aio_suspend(3) Posix specification Linux man page
- aio_cancel(3) Posix specification Linux man page
- lio_listio(3) Posix specification Linux man page
- aio_init(3) Linux man page

Relevant Background Experience

While I lack practical experience in real-time operating systems, my academic background is robust. I'm currently enrolled in a course of Operating Systems that will give me a comprehensive view on the subject. I am confident in my ability to quickly assimilate new information and bridge any knowledge gaps.

Personal

My name is Alessandro Nardin, and I'm currently a second-year Computer Engineering student at the University of Trento, located in the North of Italy.

I went to school in a technical institute, where I pursued the program of computer engineering. There, I received a comprehensive education covering fundamental concepts in computer science and engineering. I gained insight into computer architecture, operating systems, and was introduced to programming languages such as C and Java. Additionally, I obtained a basic understanding of electronics. In university, I made the decision to continue my studies in computer engineering. This program provided me with the opportunity to delve deeper into the subjects I was introduced to.

Throughout my academic journey, I developed a passion for tinkering with microcontrollers, particularly Arduino, which led me to participate in some hackathons organized by my university. Additionally, I served as a tutor during high school, teaching younger students the basics of Arduino programming.

My decision to apply for Google Summer of Code (GSoC) as an RTEMS contributor stemmed from the alignment of this project with my passions and future career aspirations. I have always been captivated by the space industry and aspire to work in this field someday. Given that RTEMS is a software widely used in the aerospace industry, I believe that gaining experience with it now will provide me with a valuable advantage in my future endeavors.

Language Skill Set

C : Intermediate
Rust : Intermediate
C++ : Intermediate
Java : Intermediate

I also have basic experience with different types of assembly (x86, Risc-V and Arm) and a basic knowledge of Python.

Related Research and Work Experience (if any):

In high school, I undertook an internship at an engineering firm, where I served as a software developer, primarily utilizing C++ for desktop applications. While the focus of the internship differed from embedded systems and real-time operating systems, during that time I gained great practical experience in using version control software such as Git, which I'm confident will also be beneficial during GSoC.