

POSIX Compliance

Google Summer of Code Program 2019 Project Final Report

Vaibhav Gupta
vaibhavgupta40@gmail.com
#rtems irc: varodek

Student:

1. Vaibhav Gupta

Mentors:

1. Joel Sherrill
2. Aditya Upadhyay
3. Hesham Almatary

Project Abstract:

Developers and community have made the Real Time Executive for Multiprocessor Systems (RTEMS) well compliant with various standards like POSIX IEEE 1003.1, C99, C11, and various profiles of FACE and SCA. This makes it possible for RTEMS to be executed on a large variety of architectures, devices and environments.

But this is an indefinite task as with advancement in technology and necessity, these standards keep updating and hence creates the need for updating the compliance status too.

The goal of the project is to update the compliance status of RTEMS with POSIX IEEE and FACE General purpose profile. Various functions and headers were required to be implemented. As the project advanced, many headers and test-suites were implemented. Code structure for architecture specific header, fenv, was also discussed and decided.

Project Tasks:

- Modify test-suite for inttypes.h
- Port ndbm.h to newlib
- Design test-suite for newlib
- Port fenv.h for ARM, PPC and SPARC and design testsuite for it. (It is still going on)

My Work:

- <https://devel.rtems.org/changeset/d9fcb22/rtems>

- https://github.com/VARoDeK/MiniProject_1-Add_newlib_object_files_to_rtems_libraries
- <https://sourceware.org/git/gitweb.cgi?p=newlib-cygwin.git;a=commit;h=e50ad9fbdc5adfb5c13954a2798d1ea0ad53db34>
- <https://sourceware.org/git/gitweb.cgi?p=newlib-cygwin.git;a=commit;h=b39cd00f07e8df11c5446fb35da490ef85f12821>
- <https://git.rtems.org/rtems/commit/?id=57e0173584a914b1a92aa73deade6afb463186db>
- <https://git.rtems.org/rtems-docs/commit/?id=fac2873533c5bb1278285e6e7e5deff749cfab8b>
- <https://git.rtems.org/rtems-docs/commit/?id=360560024726cd6166b1eaeda55ee8512594f1ca>

New methods ported:

- `int dbm_clearerr(DBM *);`
- `void dbm_close(DBM *);`
- `int dbm_delete(DBM *, datum);`
- `int dbm_error(DBM *);`
- `datum dbm_fetch(DBM *, datum);`
- `datum dbm_firstkey(DBM *);`
- `datum dbm_nextkey(DBM *);`
- `DBM *dbm_open(const char *, int, mode_t);`
- `int dbm_store(DBM *, datum, datum, int);`
- `int dbm_dirfno(DBM *);`

RTEMS Wiki of my project:

https://devel.rtems.org/wiki/GSoC/2019/POSIX_Compliance

Notes I made on GitHub:

- <https://github.com/VARoDeK/MyNotes/tree/master/RTEMS>
- <https://github.com/VARoDeK/MyNotes/tree/master/Newlib>

Project Repositories on GitHub:

- <https://github.com/VARoDeK/rtems>
- <https://github.com/VARoDeK/newlib>
- <https://github.com/VARoDeK/rtems-docs>
- <https://github.com/VARoDeK/rsb>

Weekly progress:

<https://devel.rtems.org/wiki/GSoC/2019#VaibhavGupta>

Timeline (*points mentioned in bullets are also in chronological*)

order):

1. Phase 1

- Explored RTEMS Documentations.
- Build RTEMS for ARM and SPARC with tests enabled.
- Build Newlib for ARM and SPARC.
- Cloned RTEMS Docs and build it using Sphinx for contribution.
- Submitted first version of changes in inttypes test-suite and iterated the process of taking reviews from mentors for it and modifying it accordingly.
- Made arrangements for porting ftw.h. Explored NetBSD and FreeBSD repositories as they are under favourable license.
- Figured correct location to place/modify files to successfully compile ndbm. This got a bit long as it involved discussions within RTEMS and Newlib communities. This was because according to the definition it should be placed in libc/posix directory but according to environment and support, it was shifted to libc/search directory.
- inttypes test-suite got merged.
<https://devel.rtems.org/changeset/d9fcb22/rtems>
- While porting ftw, it requires fts.h, the functions of fts.h are defined in fts.c which requires mount.h, again this header requires multiple other headers, and dependency chain is formed. This raised some questions about code structure and I reported it to Newlib community. After not getting any response for two weeks I shifted to another task.
- While updating newlib/libc/search directory, updating hash.h, creates need for several other headers to be updated. Those updated headers now require new headers, so a chain of dependencies was formed. The issue was discussed in RTEMS and Newlib communities and it got settled.
- Ported ndbm on local newlib repository and started working on procedures to apply patch of changes to RSB. Newlib build for ARM and SPARC, on my system, were generating function symbols of ndbm. But RSB patch was not able to recreate the same. This was discussed in the community.

2. Phase 2

- To keep my project moving without losing pace, I wrote a shell script https://github.com/VARoDeK/MiniProject_1-Add_newlib_object_files_to_rtems_libraries to manually add object file of ndbm from local build of Newlib SPARC to RTEMS. This was important as ndbm test-suite could not be developed until RTEMS has ndbm in its environment.
- Submitted first version of ndbm test-suite. After that, I was also able to apply ndbm patch to RSB successfully.
- Once the first version of test-suite was developed and gave positive results, I sent ndbm patch to newlib community to be merged in their source tree.
- ndbm patch was merged by newlib community after a long discussion with my mentors with some required changes.
<https://sourceware.org/git/gitweb.cgi?p=newlib-cygwin.git;a=commit;h=e50ad9fbd5c13954a2798d1ea0ad53db34>
- The ndbm port from FreeBSD had a function declaration with no definition provided. The changes were pushed for the same.

<https://sourceware.org/git/gitweb.cgi?p=newlib-cygwin.git;a=commit;h=b39cd00f07e8df11c5446fb35da490ef85f12821>

- After reviews and various cycles of changes, the ndbm test-suite was ready. I sent the patch to Mailing list.
- There were some issues with the latest release of libm in newlib hence RTEMS was not using the version in which ndbm was pushed. Hence ndbm test-suite could not be added directly. The test-suite got added in conditional way. This can be explained with the commit message.

<https://git.rtems.org/rtems/commit/?id=57e0173584a914b1a92aa73deade6afb463186db>

3. Phase 3

- This phase started with work on fenv. I explored possible sources (FreeBSD and NetBSD) to port from. The architectures I initially targeted after discussing with my mentors were ARM, PPC, RISC-V, x86 and SPARC. Although RISC-V had existing implementation, it required changes in its code structure.
- The sources for the architectures were mixed, some could be ported from FreeBSD and other from NetBSD. As both the sources have different code structures, a template was required for newlib.
- My mentors took this discussion to newlib community and made required changes. Whole RISC-V code was refactored. It and its supporting headers serve as template for implementation for other Architectures.
- Meanwhile I started working on RISC-V Testsuite. The fenv environment for RISC-V is not working properly for now and hence created a blocker. I tried to make it work for `rv32imac` with and without HTIF, `rv64imac` and `rv32imafdc` BSPs. Building it and my efforts to troubleshoot in the case of each of the BSP, took much of the time.
- So I created the test-suite for Linux environment. Once the fenv issues with RISC-V are solved, the test-suite can be ported to RTEMS.
- Meanwhile I also updated rtems-docs. Added steps to test newlib patches with RSB.
- After the code structure was finalised and changes of my mentor were pushed by newlib, I started working on porting fenv for ARM. The machine/arm/include/fenv.h and machine/arm/sys/fenv.h are implemented and now I am working on stubs.

Post GSoC Work:

I will continue porting fenv for other architectures.

Since, I have got a good hand at compliance tasks, I will continue to complete other tasks too which are mentioned under this ticket, like socketmark(), pselect(), file descriptor functions, fmtmsg.h, memcpy(), sys/ipc.h, getdate() etc.

Contributions in the organization in the coming future:

I am planning to pick up some tasks related to hardware like BeagleBoard or Raspberry Pi.

In this project, my major focus was on SPARC and ARM. And now I want to work on other Architectures too.

I will also apply for GCI Mentorship this year.

There are many open projects which amazed me when I was preparing for GSoC, would love to work on few.