

Implementation of the ISO9660 filesystem

Google Summer of Code Program 2011 Project Proposal

Christophe HURIAUX
c.huriaux@gmail.com
École Nationale Supérieure des Sciences Appliquées et de Technologie
Lannion, France
IRC Nickname : Shisui

Project Abstract.

The goal of this project is to give RTEMS the ability to read CD, DVD and/or other media using ISO9660 read-only file system.

Project Description.

RTEMS is currently able to interact with various file systems such as IMFS, FAT or NFS, but lacks ISO9660 management. With the recent work on disk support made by previous GSoC student Claudio Silva which brings partial SATA and ATAPI compliancy to RTEMS, a working ISO9660 file system implementation would allow to easily handle read-only media through a block devices. Furthermore, while adding the various features of this specific file system, I could refactor the existing helper from the file system layer instead on duplicating existing code, thus cleaning a bit some file system routines.

ISO9660 has been designed to allow fast searching of file through directories, but was also constrained with various limitations in terms of file name length ("8.3" uppercase formatted characters in ISO9660 Level 1), directory depth (8 maximum in the specification, including the root directory).

Several extensions to this filesystem tends to enhance the potential of ISO9660 like *Joliet* which allows longer filenames and unicode encoding, *Rock Ridge* which brings POSIX compliancy (file modes, symbolic links devices files) or *El Torito* for bootable CD. Those extensions (especially Joliet) should be as well supported to ensure a wide support of read-only media.

Such a filesystem would be perfectly suitable for a real-time system because it is by nature optimized for reading and lowering seek time. In the long term, the implementation could also be able to support write operations on ISO9660 devices through multisession, providing a high-end mass storage device, but others more specific Write-Once Read-Many filesystems could be used for this purpose (such as UDF).

Project Deliverables

- May 23 (coding begins)
 - Check if the current SATA/ATAPI implementation present in the CVS is ready to use
 - Have a working test environment (with ATAPI layer or QEMU CDROM IDE emulation or even with small ISO image in ram)
 - Reevaluate the goals and deliverables of the project if needed
- June 15th (First Milestone)
 - Design the ISO9660 stack and requirements
 - The test program should be able to iterate trough the directory structure of the filesystem with a basic user interface
- July 12th (Midterm Evaluation)
 - Working read-only implementation of level 1 & 2 ISO9660

- Testsuites to quickly identify regressions
- August 9th (Final Evaluation)
 - Support of at least Joliet extension
 - Write down the documentation
- August 23rd (Final Results Announced)
 - Refine the documentation and clean up the code

Proposed Schedule

April 26 - May 23 2010 (Setup)

This period will lead me to a fully working development environment. Since I'll have to work on ATAPI, I will use the PC386 BSP (as suggested by *DrJoel* on IRC) emulated by QEMU, I already managed to have a working environment, but I will need to play with some things like remote debugging before being really ready.

Additionally, I can dig into the NetBSD SATA driver port made during the previous GSoC. Even if I can develop the filesystem layer with another block device, the support of the SATA driver is somehow required in order to make the ISO9660 implementation really useful in the long term.

If it came that the SATA driver isn't really functional, I can test my work on others block devices such as the actual ATA/IDE layer (QEMU would emulate the CDRom) or directly through a ramdisk where an ISO image could be copied.

May 24 - June 15 2010 (Design)

Note : During this period I will still have some courses, the time commitment for the project will be between 15 to 20 hours per week. I will be completely off the Internet between June 1st and June 4th since i'm participating in the *French Robotic Cup*, aka *Eurobot* (see [this link](#) [in french] for more details)

When the coding period will begin (or even before if the SATA drivers works), I'll start to write the various helpers required for basic functionalities of the filesystem, such as record tree traversing, session/track identification, ...

This first period will also helps me to identify helpers from others filesystems which could be moved to a generic layer.

June 16 - July 12 2010 (Prototyping)

Note : Starting from this period I won't have any courses and will allocate a full time commitment for the project, just like a summer job (40-50 hours per week)

Over this period I will elaborate a working prototype of the ISO9660 filesystem layer, able to read the several descriptors and successfully retrieve a file through the various records. I will need to determinate whether I implement a tree traversing via each directory or through the static table of paths, or maybe both of them.

July 13 - August 9 2010 (Extending)

Upon successful prototyping I will need to integrate Joliet and Rock Ridge extensions to the ISO9660 implementation and elaborate testcases.

These two extensions will allow to have read access to virtually any ISO9660 formatted media. If I have more time I could elaborate routines to load applications from an ISO9660 enabled block device for example.

The end of this period will be dedicated to cleaning up code and beginning a official documentation gathering all informations written down on the wiki all along the summer.

August 10 - August 23 2010 (Polishing)

After the final deadline of GSoC my main goal will be to finish the polishing of the code and writing down the documentation in order to make it really usable by users

Continued Involvement

_____I'd like to continue my work on this part of the system after the GSoC, there is many aspects of modern hardware to bring up to RTEMS.

Future Improvements

Further development on this implementation could include write support in write-enabled block devices (creation of ISO files) and implementation of filesystems derived from ISO9660 such as UDF for example.

As bonus work if I fulfill my goals early, I could work on the SATA/ATAPI NetBSD port if it's not mergeable, since it would provide a better exposure to this new filesystem.

Major Challenges foreseen

- Adapt myself to RTEMS architecture and its filesystem layer.
- Diving into Claudio Silva SATA driver code.
- Find the filesystem layer helpers which would need to be refactored.

References

- ECMA-119 release of the unreviewed ISO9660 specification : <http://www.ecma-international.org/publications/standards/Ecma-119.htm>
- Wikipedia pages :
 - ISO9660: http://en.wikipedia.org/wiki/ISO_9660
 - Rock Ridge : http://en.wikipedia.org/wiki/Rock_Ridge
 - Joliet : [http://en.wikipedia.org/wiki/Joliet_\(file_system\)](http://en.wikipedia.org/wiki/Joliet_(file_system))
- Various specifications for the several existing extensions of ISO9660

Relevant Background Experience

- University courses about system programming, real-time systems.
- Few years of experience in general programming and system programming

Personal

I'm a 21 years old french student in second year at École Nationale Supérieure des Sciences Appliquées et de Technologie, with a major in electronics, embedded systems and computer sciences (equivalent of a US Master's Degree in engineering). I'm also an enthusiast for a long time in system development and general programming.

I applied last year for GSoC and find my whole summer rewarding from many points of view. This year when I was looking for an interesting project I tried to be closer to my studies and my future jobs, that's why I choose to apply to RTEMS.

I have some experiences in RTOS like VxWorks thanks to my studies, but this is the first time I use RTEMS "for real", one of my professors mentioned it as an valuable alternative RTOS.

I am interested in OS theory, low-level software programming and hardware design (through HDL languages and IC design). I learnt x86 assembler by creating a basic x86 OS from scratch (capable of the most basic things such as jumping into 64bit protected mode, handle interrupts, memory management and do task switching). I am quite self taught and love to learn how things works, especially in computers and embedded systems domains.

Experience

Free Software Experience/Contributions:

- Haiku contributor as part of GSoC 2010 with the "*Integration of Services Kit*" project, the goal was to write a network layer providing high-level protocols access (HTTP, FTP, ...), and to provide a native layer to the WebKit port of Haiku
 - Wrap-up report : http://www.haiku-os.org/blog/shisui/2010-08-19_services_kit_features_overview
 - Most of of the code can be seen in (unfortunately the webkit port haven't any web interface): <http://dev.haiku-os.org/browser/haiku/trunk/src/kits/network/libnetapi>

Language Skill Set

- Proficient in C, C++
- Significant experiences with Java, web technologies (PHP, XHTML, Javascript, ...) and SQL
- Moderate level in assembler for several platforms (mostly x86, Microchip PIC and ARM)
- Beginner in hardware description languages (Verilog, VHDL)
- Some contacts with Python

Related Research and Work Experience (if any):

- Design and implementation of a line following program on a NIOS 2 powered Altera Cyclone 2 FPGA during a 3 months internship in 2009. The program was first designed to work on a dsPIC (a 16-bits DSP from Microchip) and then ported to a fully integrated FPGA. The goal was to design a line following robot implementing a PID regulation to let students learn control engineering.

