USB Audio Functionality for PortalPlayer Targets

A Google Summer of Code 2008 Proposal for Rockbox

Paul Hinze

7 April 2008

Abstract

Rockbox has established itself as a viable Free Software alternative to the firmware on many Digital Audio Players (DAPs). One of the keys to its appeal is the fact that new features are constantly being developed and improved. In August 2007 development began on a USB stack, which allows a DAP to connect to a computer to transfer files and information without having to reboot into the manufacturer firmware. This is a proposal to add USB Audio support to Rockbox which will make it possible to use a DAP as a USB sound card. This can allow PCs without a sound card (or with a poor sound card) to use a DAP running Rockbox for audio output. Even on PCs with sound cards, it may still be desirable to send the PC's audio output to a DAP so that music from Rockbox and sounds from the PC can be mixed to one set of speakers or headphones. Another useful application would be the ability for a PC to record audio directly from a DAP's FM tuner.

Task Division & Timeline

This project divides into several major tasks, each of which constitutes a phase of the timeline:

- **Phase 0: Learn about Rockbox internals** *April 21 – May 26* [5 weeks]

  From the date of acceptance to the first day of coding, I will spend time familiarizing myself with the Rockbox codebase by editing code and reading debug messages. I will focus on gaining a general understanding of Rockbox as a whole and more detailed knowledge the existing USB stack.

  Deliverables for this Phase:

  - A working understanding of Rockbox from a high level
  - Detailed knowledge of the USB stack

- **Phase 1: Expose Volume Control over USB** *May 26 – June 2* [1 week]

  This is a relatively small introductory project that will allow me to deal with the structures necessary for USB programming in Rockbox. This will involve writing code that allows the main Rockbox volume to be controlled over USB.

  Deliverables for this Phase:

  - Code that exposes Rockbox volume control over USB

- **Phase 2: Implement isochronous transfer in the USB driver** *June 2 – June 16* [2 weeks]

  Isochronous transfer mode is the core feature of USB utilized by audio applications. It allows for the reservation of a certain guaranteed stream of bandwidth for data transfer. In this phase I will add isochronous transfer support into the Rockbox USB Driver. This will include both input and output data streams.

  Deliverables for this Phase:

- Successful read of input isochronous data stream from USB driver
- Successful write of output isochronous data stream to USB driver

- **Phase 3: Implement Basic USB Audio Out** *June 16 – June 23* [1 week]

  In this phase, I will use the isochronous transfer mode from Phase 2 to direct a decoded Rockbox audio stream out over USB such that it is recognized by a connected PC as a recording source and can be played through the PC's sound card.

  Deliverables for this Phase:

  - Successfully stream Rockbox audio output to PC over USB

- **Phase 4: Implement Basic USB Audio In** *June 23 – June 30* [1 week]

  Similar to Phase 3, this task involves playing an audio stream as received from a PC. Initially this stream will be played instead of the normal Rockbox audio, with stream mixing to be implemented in Phase 5.

  Deliverables for this Phase:

  - Successfully stream PC audio output to Rockbox over USB

- **Phase 5: Extra features** *June 30 – July 21* [3 weeks]

  Once USB Audio is in place, various features should be implemented to allow the user to control it. This includes the ability to switch between recording sources and to switch between line out and headphones output. Another crucial feature is the integration of USB Audio input from a PC and the Rockbox music stream. These features will be implemented in this phase.

  Deliverables for this Phase:

  - Successfully mix USB Audio input and Rockbox music stream
  - Ability to switch between recording sources
  - Ability to switch between line out and headphones

- **Phase 6: Testing and Debugging** *July 21 – August 11* [4 weeks]

  As in any project, the code must be thoroughly tested and debugged.

Benefit to the Rockbox project

Besides the fact that it is open source software, the draw of the Rockbox project is the fact that it provides software for your DAP that is more useful than the firmware that is shipped from the manufacturer. Rockbox attempts to accomplish this in two ways: by being user friendly and by providing more features. Adding USB Audio support will allow Rockbox audio players to be used in several unique ways. The ability to mix PC audio with music playback is a feature that is unavailable on any other portable DAP, and would be a valuable addition to the things users can do with Rockbox that they cannot do anywhere else.

About the Author

I've been tinkering around with computers ever since I was 7 years old and realized that my dad's 386 had games. I have taken that interest and enthusiasm for technology into my studies, and I will be graduating in May 2008 with a degree in Computer Science from Marquette University. My summer is free for full-time GSoC work, barring minimal vacation time, which is as-yet unplanned. I have taken classes in Hardware Systems, Operating Systems, and Embedded Systems, and for several years I've been a member of the Embedded XINU research lab. This lab has been porting and XINU (XINU Is Not Unix), a small but fully featured educational operating system, to the MIPS platform targeting Linksys WRT54GL routers. This

ongoing project is in a way very similar to the Rockbox project, as it involves implementing new open source software on a completely undocumented platform through reverse engineering. My time with the Embedded XINU Lab, which has included two full summers of work, has included several highlights. When we were just starting the MIPS port, I made the first breakthrough on the serial driver (one character and then "Hello, world!" across the line) after the whole lab had spent months banging its collective head against a wall. This past summer I managed to get the network driver to send its first packet (after which it crashed and burned, but that's another story), and I also participated in an intensive code review of every portion of the operating system. These experiences with a large project written in C for an embedded system I believe have prepared me well to work on the Rockbox project.

I have long believed in the philosophy of open source, and I use a great deal of Free Software. I have never before had the chance to contribute in a significant way, and I am excited by the opportunity to do so. I have owned an iRiver H10 20GB for several years and I put Rockbox on it soon after learning about it. I have since been an avid Rockbox user, though I have never before contributed code. I have successfully downloaded and compiled the latest source from Subversion, and I look forward to getting my hands dirty.