Building an audio platform for embedded devices in Rust

Josef Utbult

Dept. of Computer Science, Electrical and Space Engineering
Luleå University of Technology
Luleå, Sweden

Supervisor:

Per Lindgren

To my father, who believed in me all the way...

Abstract

Begin with abstract

Contents

Acknowledgments

I want to acknowledge the following people, who have helped me tremendously. Without their contributions, this report would never have been made.

- Per Lindgren, who was my supervisor for this master thesis, and helped me to get on the right path.
- Emil Fresk, who functioned as my Rust-guru and helped me with problems I would have never have solved myself.
- The dude who made the usbd_audio module for Rust.

Expand on usbd_audio

I truly stand on the shoulders of giants.

Josef Utbult
June 2023
Luleå

Todo list

Chapter 1

Introduction: Why create an audio platform in Rust?

Add an Introduction

Chapter 2

Background

Add Background

2.1 Theory

Add section on USB history to Theory

Add section on USB 2.0 to Theory

Add description on speed modes for USB 2.0 in Theory

Add section on codecs to theory

Add section on I2S to theory

Add section on STM32 SAI to theory

2.1.1 Development board

A *development board*, or in short hand form a *devboard*, is a PCB containing a microprocessor and pin headers for interfacing with the legs of the microprocessor IC. It also contains everything needed to run the microprocessor, such as voltage regulators and oscillator circuitry. Many development boards also include on-board programming functionality, which lets you as a programmer program the board using only a USB connection to the development board.

Source I guess?

2.1.2 Rust

Add A LOT about Rust

2.1.3 The USB protocol

USB 1.0

USB 2.0

USB audio class device

2.2 Method

Add requirements to method

Add task list and sprint plan in method

2.2.1 Goals

The goals for this master thesis is to have made the following.

Design a PCB

The project aims to create a platform for audio devices in Rust. For this we will need some hardware that can be tested and measured on.

The PCB for this will therefore have to contain the following in order to meet the goals.

- A codec for converting analog signals from a microphone or sensor to digital samples for processing. It will also need to be able to convert the digital samples back to analog signals for playback on a speaker or headphones. This codec needs to be able to process four analog input signals, and four analog output signals.
- A microprocessor which has support for the protocol used by the codec, and support for high speed USB.
- A USB 2.0 port, or a USB type-C port using the USB 2.0 protocol.
- A display for displaying metrics such as signal amplitude, current volume and gain.
- A potentiometer for each analog input. These should be connected to the microprocessor for controlling the gain and/or volume of the specific audio input.
- A pin header which exposes GPIO pins to the microprocessor, used for easy implementation of future parts for proof of concept designs

Reformulate

.

- A pin header which exposes the different wires between the microprocessor and the codec, for debugging purposes.

The goal with the PCB design is to start by creating a board without a microprocessor, which functions as a HAT

Define HAT

for an external development board. This design is then extended to incorporate the microprocessor, which will let it function in a stand-alone mode.

Software layer

The software layer that this project aims to create should function as a platform on which future software can be implemented upon. This software layer is the main object of this thesis, and it needs to incorporate the following.

- Communication with the codec for initialization and streaming of data. This software needs to be extendable for allowing exchanging of the codec.
- Exposing of the sample input and output data in some sort of queue, where an implementation that uses the software layer should be able to read the samples, process it and write samples back.
- The option to include a USB audio class layer, where an implementation should be able to read and write samples to a USB host. This should be made compatible with the queue system of the codec, so that an implementation can read samples from the codec, process them and send them to the USB host (and vice versa).
- Unit tests for all the functionality of the software layer to prove software stability and robustness.

Documentation

This software layer should be documented on how each function in it works, how the overall structure of the software layer works, and examples on how to implement it.

Signal testing

Tests should be run on the PCB and the running software. The following needs to be measured.

- Signal to noise ratio

  Add description of signal to noise ratio

.

Add appropriate signal tests

2.2.2 Plan

Should this be in the report?

To be able to complete this project, the following parts needs to be done.

- Studying of relevant topics such as the *USB protocol*, the *USB Audio Class*, implementation the *I2S protocol* and the *STM32 Serial Audio Interface (SAI)* functionality.
- Selecting a high quality codec that can handle four input and four output channels, all running at a sample rate of 48000 Hz.
- Selecting a microprocessor that can interface with the chosen codec, and also has support for high speed USB. A development board for this specific microprocessor also needs to be available.
- Designing and building of a PCB that can interface with a development board, containing a codec, a display, potentiometers and a pin header for debugging the wires between the development board and the codec.
- Study the existing software modules for I2S communication and USB Audio classes. This is to make sure that they contain all the necessary components needed for this project

  Such as?

  . If these parts aren't present, they will need to be implemented.
- Creating an architecture of the software layer, containing how each part of it should work and the flow of information.
- Build a software layer that follows the described architecture.
- Create unit tests for parts of the software layer that can be feasible tested.
- Write documentation for the software layer.
- Create and build a final PCB that can be used without the development board.
- Test the audio signals for the board.

A time plan for the completion of the master thesis can be found in Table 2.1. This project started at 6 July 2023, and is set to be finished at 18 October 2023.

The time plan is sectioned of in three sprints. The work that should be done in these sprints will be selected at the start of each sprint.

Table 2.1: Time plan for the completion of the master thesis

I assume this shouldn't be in the final report either

Week	Date	Part
V23	05 jun	Planning
V24	12 jun	Rapport structuring + Pre study
V25	19 jun	Pre study
V26	26 jun	Pre study + selection of codec and devboard
V27	03 jul	Sprint 1
V28	10 jun	Sprint 1
V29	17 jun	Sprint 1
V30	24 jun	Sprint 1 + Report writing and evaluation
V31	31 jun	Sprint 2
V32	07 aug	Sprint 2
V33	14 aug	Sprint 2 + Report writing and evaluation
V34	21 aug	Sprint 3
V35	28 aug	Sprint 3
V36	04 sep	Sprint 3 + Report writing and evaluation
V37	11 sep	Final touches on development
V38	18 sep	Report writing
V39	25 sep	Report writing
V40	02 okt	Report writing (First draft)
V41	09 okt	Report writing

2.3 Result

Add results

2.4 Discussion

Add Discussion

2.5 Conclusion

Add Conclusion

References