Suitability of the Rust programming language for embedded systems

Student

Noah Kälin

Introduction: Software that runs on embedded devices is predominantly written in C or C++.
Rust (Fig. 1) is a rather new programming language, focuses on performance and memory-efficiency and at the same time guarantees memory-safety and thread-safety.
Performance and efficiency are very important features for embedded devices. Therefore, the suitability of Rust for embedded software development should be assessed by comparing it to C and C++ by the means of specific examples.

Approach: First, general knowledge about the programming language itself is acquired.
Various examples from concurrency to low-level applications are created and compared to similar solutions written in C and C++. The comparisons are based primarily on memory-safety, memory-efficiency and runtime speed.
Existing tools and libraries targeting embedded devices are inspected and used in combination with the example applications.

Conclusion: These results show that Rust delivers on its promises, and it allows creating good abstractions which help to accomplish the demanded properties.
Several tools and frameworks that already exist can considerably facilitate the development process of embedded software. Projects to unify Hardware Abstraction Layers (HALs) for embedded devices show great potential (Fig. 2, Fig. 3).
Development of the language continues, therefore it is constantly improving and evolving for the better which should allow it to be used in an increasing number of applications.

Figure 1: Rust logo
https://www.rust-lang.org/

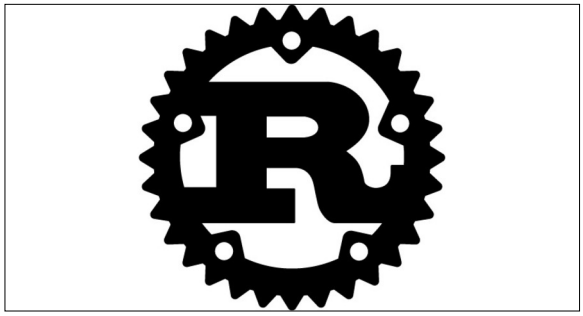


Figure 2: "embedded-hal" layers of an example application
https://docs.rust-embedded.org/book/portability

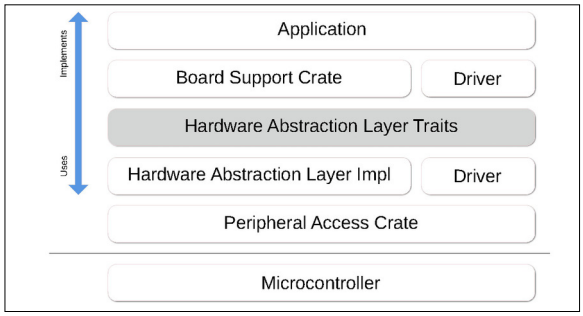
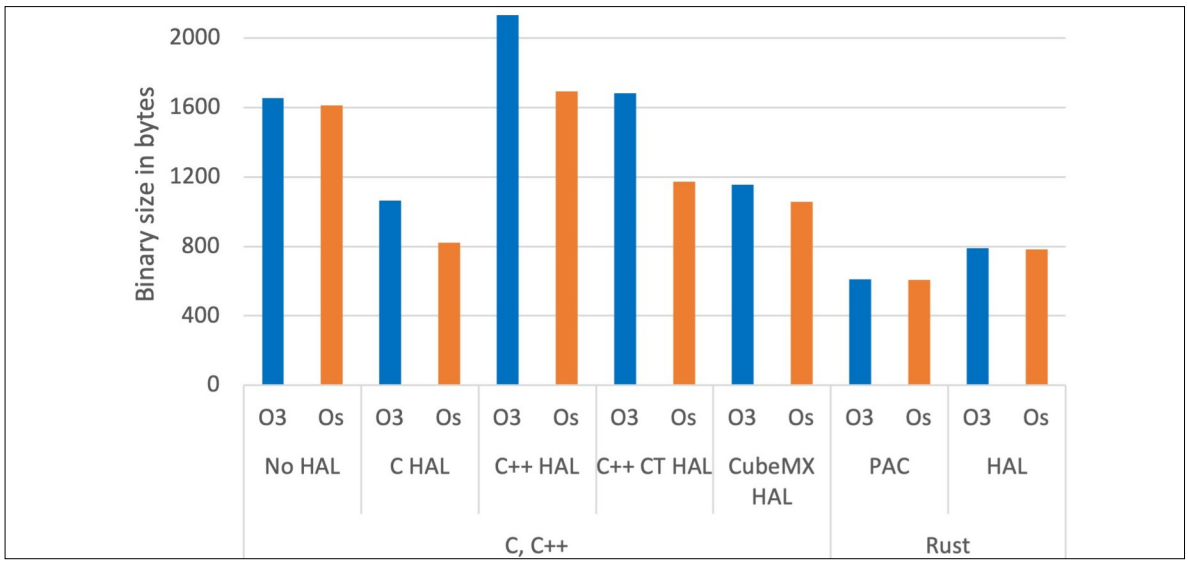


Figure 3: Binary size comparison of abstractions with optimization levels, size of an empty main binary is subtracted
Own presentment



Advisor
Prof. Reto Bonderer

Subject Area
Electrical Engineering