technische universität dortmund

computer science 12

Fachprojekt

Real-Time Scheduler Implementation on Unikraft

Christian Hakert
Junjie Shi
Prof. Dr. Jian-Jia Chen
Otto-Hahn Str. 16
Technische Universität Dortmund
27.04.2020

Real-time applications are widely present in many fields of modern computer systems. No matter if controlling safety-critical systems in a car, in a plane or in a spaceship, providing autonomous driving or controlling a nuclear power plant, guaranteeing a worst-case time until a task completes is crucially important to prevent catastrophes. The area or real-time scheduling provides algorithms and methodologies, which can guarantee the worst-case time for a task completion under given conditions. Thus, using real-time scheduling allows to control the aforementioned systems with computer based controllers.

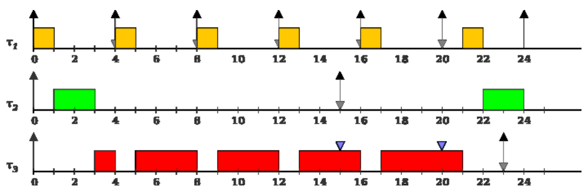

Figure 1: Earliest-Deadline-First (EDF) schedule

When it comes to embedded systems, resources are typically constrained. For instance, a system may only provide several kB of RAM or only a single CPU core with a few MHz clock frequency. Therefore, using full-blown operating systems (e.g. linux) is not practicable on many embedded systems. Special real-time operating systems have been developed in the past, which only require minimal resources and still provide the necessary functionality to schedule tasks under real-time constraints. However, these systems usually lack of providing standard application libraries or legacy interfaces.

Unikraft [1] is a configurable unikernel, which only provides minimal functionality in the basic configuration. The resource consumption in this configuration is minimal and it suits even small embedded systems. Unikraft further allows to configure advanced features into the system, for instance a python interpreter library can be selected, if it is required. Due to the fact, that such a functionality can be selected separately and does not face many dependencies, high level functionality (e.g. python) can be even provided on heavy resource constrained systems.

In this work, students should implement a simple real-time scheduling library for the unikraft kernel. The library should not be invasive, i.e. when not selected, it should have no effect on the kernel. Students are requested to implement the requires task abstraction for real-time tasks and to implement a simple, priority-based scheduling strategy. Hardware independence is not required, but would be useful.

Required Skills:

- Knowledgeable of C and C++ programming
- Basic knowledge of tasks and schedulers

Acquired Skills after the work:

- Knowledge about modern configurable unikernels
- Knowledge about implementation of real-time systems.

[1] Kuenzer, Simon. "Unikraft: Unikernels Made Easy." (2018).