Master Thesis:

Safe Software for Safe Flights
Implementation of a Mixed-Criticality Multi-Core Scheduling Framework

The Problem: Complex embedded systems are typically mixed-criticality systems, where functionalities of different importances (criticalities) co-exist. As an example, the airborne software systems are usually categorized as flight critical or mission critical. For flight critical functionalities, like the autopilot, a failure could result in a airplane crash, whereas for the mission critical functionalities, like the radio communication, the failure consequences are not as severe. At the same time, various unexpected situations may arise during the operation of the airplane, since neither the hardware nor the software that we build are perfect. How should a system react to such unexpected situations depending on its criticality? And what must / can we guarantee in such dynamic environments? Efficient scheduling and online adaptation are keys to answering those questions.

The Thesis: Recently several scheduling policies have been proposed for mixed-criticality applications on single-core and multi-core systems. Although the theory behind these policies is well-founded, there is a lack of relevant implementations. Namely, there are not (enough) CPU schedulers that implement these policies in existing operating systems.

The goal of this thesis is to develop one of the first frameworks for mixed-criticality scheduling. The framework will feature at least two mixed-criticality CPU schedulers, implemented in the kernel of a real-time operating system. Your task is (i) to implement the schedulers and (ii) evaluate their runtime behavior and scalability on a multi-core platform. Depending on your interests, you can also get involved in the design of new scheduling policies or the co-scheduling of other platform resources, such as shared cache or memory.

During the thesis, you will be exposed to the cutting-edge research of mixed-criticality scheduling, while you gain practical programming skills on real-time operating systems (e.g. Litmus\textsuperscript{RT}) and massively parallel hardware platforms (e.g. Intel Xeon Phi).

Requirements:
Familiarity with C/C++ and the Linux OS.

Interested? Please contact us for more details!

Contacts
- Georgia Giannopoulou: ggeorgia@tik.ee.ethz.ch, ETZ G77
- Pengcheng Huang: phuang@tik.ee.ethz.ch, ETZ G77
- Lukas Sigrist: sigristl@tik.ee.ethz.ch, ETZ G78.1