Towards RTOS support for mixed time-triggered and event-triggered task sets


Abstract
Many embedded systems have complex timing constraints and, at the same time, have flexibility requirements which prohibit offline planning of the entire system. To support a mixture of time-triggered and event-triggered tasks, some industrial systems deploy a real-time operating system (RTOS) with a table-driven dispatcher complemented with a preemptive scheduler to allocate free time slots to event-driven tasks. Rather than allocating dedicated time-slots to time-triggered tasks, we propose to dynamically re-allocate time-slots of time-triggered tasks within a pre-computed time range to maximize the available processing capacity for event-triggered tasks.

Although the concept - called slotshifting - is not new, we are unaware of a commercial RTOS with such support. After identifying the mechanisms for an RTOS implementation of slotshifting, we discuss the run-time overheads for admitting a-periodic requests into the system.

1. Slotshifting recapitulated [1]
- Table-driven (interval-based) scheduling of periodic tasks.
- Online admission of a-periodic requests.
- Holistic EDF scheduling of all admitted task.

2. RTOS support
We have extended a COTS RTOS - Micrium µC/OS-II - with:
- Time-keeping for time-triggered tasks
- EDF scheduler
- WCET monitoring:
  - At an interval start: estimate the spare capacity based on the monitored WCET values of the allocated tasks.
  - Estimation of spare capacity from just the periodic tasks allocated to the specific interval.

Recording and visualization of the execution of slotshifting in µC/OS-II is done by an extended version of Grasp [2].
- Admission control:
  - Trade-off: use estimated spare capacities or re-compute the exact spare capacity.
  - Demand-bound test [3] for computing the impact of sporadic tasks can be computationally expensive.
  - Reserve execution time for the admission control itself by means of a sporadic task.

3. Future work
- Computationally efficient online admission for aperiodic tasks?

4. References