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

Martijn van den Heuvel,
R.J. Bril, J.J. Lukkien, D. Isovici
G. Sankar Ramachandran

Thanks to:
Syed M.J. Abdullah and Xiaodi Zhang
Complexity of industrial applications

Efficient resource usage vs. complex constraints
Task model

• **Tasks**
  - Time-triggered: Periodic task ($P_i$)
  - Event-triggered: Sporadic task ($S_i$), A-periodic task ($A_i$)

• **Job**
  - Relative deadline = D
  - Absolute deadline = arrival time + D

Preemptive EDF scheduling: based on absolute deadline
How slotshifting works?

**Offline**

Periodic tasks with simple and complex constraints

Offline schedule with simple constraints and information about unused resources

Sporadic tasks guaranteed for the worst-case arrival pattern on top of the offline schedule.

**Online**

At run-time, periodic tasks can run flexibly within their assigned intervals.

Keep track on sporadic arrivals to reduce the pessimism of the offline guarantee.

Firm aperiodic tasks are guaranteed on top of offline schedule and sporadic tasks.

[Isovic and Fohler, RTSJ 2009]
Creating the off-line schedule (1/2)

• Offline reservation of periodic tasks into intervals

- Divide an interval into reserved capacity and spare capacity

\[ I_0 \]
\[ I_1 \]

\[ \text{Periodic tasks} \]

\[ \text{length of the interval} \]

\[ \text{reserved capacity} \]

\[ \text{spare capacity} \]
Creating the off-line schedule (2/2)

• reserved capacity of an interval:

• Periodic tasks requiring more capacity must execute in an earlier interval
How slotshifting works?

Offline

Periodic tasks with simple and complex constraints

Offline schedule with simple constraints and information about unused resources

Sporadic tasks guaranteed for the worst-case arrival pattern on top of the offline schedule.

Online

At run-time, periodic tasks can run flexibly within their assigned intervals

Keep track on sporadic arrivals to reduce the pessimism of the offline guarantee

Firm aperiodic tasks are guaranteed on top of offline schedule and sporadic tasks

[Isovic and Fohler, RTSJ 2009]
Off-line test for sporadic tasks

Schedulability test of sporadic tasks

- Classical demand-based analysis
- Supply by slotshifting (spare capacity)

\[
\text{Demand during an interval of length } L \leq \text{ supply based on slotshifting}
\]

[Baruah RTSS06]
How slot-shifting works?

Offline

Periodic tasks with simple and complex constraints

Offline schedule with simple constraints and information about unused resources

Sporadic tasks guaranteed for the worst-case arrival pattern on top of the offline schedule.

Online

At run-time, periodic tasks can run flexibly within their assigned intervals

Slot shifting

Keep track on sporadic arrivals to reduce the pessimism of the offline guarantee

Firm aperiodic tasks are guaranteed on top of offline schedule and sporadic tasks

[Sovic and Fohler, RTSJ 2009]
How slot-shifting works?

**Offline**

**Periodic tasks** with simple and complex constraints

**Offline schedule** with simple constraints and information about unused resources

**Sporadic tasks** guaranteed for the worst-case arrival pattern on top of the offline schedule.

**Online**

At run-time, **periodic tasks** can run flexibly within their assigned intervals

Keep track on **sporadic arrivals** to reduce the pessimism of the offline guarantee

**Firm aperiodic tasks** are guaranteed on top of offline schedule and sporadic tasks

[Isovic and Fohler, RTSJ 2009]
Extending a commercial RTOS

• **MicroC/OS-II is**
  - Event-driven fixed-priority scheduling
  - Available at [http://micrium.com/](http://micrium.com/)

• **We extended MicroC/OS-II with**
  - Time-keeping for time-triggered tasks
  - WCET monitoring
  - EDF scheduler
  - Admission control (accounting spare time)
Extending a commercial RTOS

• **MicroC/OS-II is**
  - Event-driven fixed-priority scheduling
  - Available at [http://micrium.com/](http://micrium.com/)

• **We extended MicroC/OS-II with**
  - Time-keeping for time-triggered tasks
  - See paper for pseudo-code
  - WCET monitoring
  - EDF scheduler
  - Admission control (accounting spare time)
Extending a commercial RTOS

• **MicroC/OS-II is**
  - Event-driven fixed-priority scheduling
  - Available at [http://micrium.com/](http://micrium.com/)

• **We extended MicroC/OS-II with**
  - Time-keeping for time-triggered tasks
  - EDF scheduler
  - WCET monitoring
  - Admission control (accounting spare time)
WCET monitoring

- System queue with absolute deadlines
- A **virtual** interval timer to track WCETs
  - (De)Activate upon context switch
- Timer expirations require overload handling

Extending a commercial RTOS

• **MicroC/OS-II is**
  - Event-driven fixed-priority scheduling
  - Available at [http://micrium.com/](http://micrium.com/)

• **We extended MicroC/OS-II with**
  - Time-keeping for time-triggered tasks
  - WCET monitoring
  - EDF scheduler
  - **Admission control (accounting spare time)**
Implementation complexity

- Efficiency of monitoring spare capacities:
  “executing a periodic task in an other interval than designated swaps spare capacities.”

- Updating spare capacities may propagate through the entire hyper period…
Handling a-periodic requests

- **Trade-off:**
  
  estimated spare capacities vs. exact spare capacity.

- **Hard real-time a-periodic tasks:**
  - Pseudo-polynomial test.
  - Reserve time for the admission control itself (sporadic server)

- **Soft real-time tasks run in the background**
Extending a commercial RTOS

• **MicroC/OS-II is**
  - Event-driven fixed-priority scheduling
  - Available at [http://micrium.com/](http://micrium.com/)

• **We extended MicroC/OS-II with**
  - Time-keeping for time-triggered tasks
  - WCET monitoring
  - EDF scheduler
  - Admission control (accounting spare time)
Evaluation

- Overheads of event-handling and accounting

Worst-case  Off-line scheduled
Recorded example

- Instrumentation of MicroC/OS-II
- Synthetic workload

Borrowing and lending

Resource reclaiming

Summary

- Implementation of slotshifting in µC/OS-II:

  - Handles task sets with mixed real-time behavior
  - Modular run-time mechanisms

- Performance evaluation of RTOS overheads
Future work

• Future work
  • Resource sharing between sporadic and periodic tasks

New offline analysis + New online scheduling mechanism