Hierarchical Scheduling of Complex Embedded Real-Time Systems

Thomas Nolte

2009-09-03 @ ETR’09 Paris
We are dependent on various devices to manage our everyday life.

Most of these devices contain built-in computers, and we, “the researchers”, call these computer-based devices **embedded systems**.
Looking at automotive electronic/computer internals
Looking at automotive electronic/computer internals

Embedded systems implement *functions* of a device, and a device can consist of one or more functions.
... and the number of functions are increasing dramatically!

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ignition</td>
<td></td>
<td></td>
<td>Fuel Injection</td>
<td>Enginecontrol Otto</td>
<td>FSI</td>
<td>electromagn. Valves?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Valve control</td>
<td>Diesel pump</td>
<td>Pumpe-Düse-ECU</td>
<td>32bit Controller</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Slip control</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chassis</td>
<td></td>
<td></td>
<td>ABS</td>
<td>ESP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Bremsassistent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Autom. Cruise Control</td>
<td>autom. emerg. stop</td>
<td>ACC Stop-go</td>
<td>UVF?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Lenkhilfe</td>
<td>Überlagerungshilfen</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>controlled Damping</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Safety</td>
<td></td>
<td></td>
<td>Airbag</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comfort</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power+Wiring</td>
<td></td>
<td></td>
<td>CAN</td>
<td>D2B</td>
<td>MOST, LIN</td>
<td>TTP/Flexray</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>watercooled Generator</td>
<td>Startergenerator</td>
<td>APU?</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12V</td>
<td></td>
<td>elektron. ZE</td>
<td>power module</td>
</tr>
<tr>
<td>Information</td>
<td>Radio</td>
<td></td>
<td></td>
<td>Sound systems</td>
<td>TV</td>
<td>Infotainment</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Satellite radio</td>
<td>DAB</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Trip computer</td>
<td>GPS Navigation</td>
<td>Internet</td>
<td>Veh.-Veh.-Comm.</td>
</tr>
</tbody>
</table>
The Importance of Software

...and a word on complexity
Automotive E/E Systems

- 90% of new innovations is realized by software
- 23% of total cost, estimated to increase to 35% 2010
Industrial Automation Software Evolution

KLOC = 1000 lines of code

~20'000 KLOC
multiple development sites
3rd Part OS, Graphics, DB, Office Suite, Components

~1'500 – 3'000 KLOC
3-5 development sites
3rd Party OS, Graphics, DB

~200 – 300 KLOC
single development site
No 3rd party content

1980 1990

Telecommunication Evolution

Ericsson:
The World’s 5th largest Software Company

80% R&D related to Software

SonyEricsson:
Software doubled every 18 months
Challenges

- Complexity
- Integration
- Predictable quality

Hierarchical scheduling is a technology that has potential towards meeting these challenges.

Trends
- Increased global competition
- Shorter time-to-market and shorter product life-cycles
  - Increase reuse of existing legacy code
  - Increased use of external components/outsourcing
Example

Functions and tasks
Electronic Control Unit (ECU)

- Robust encapsulated computers
- Interconnected with fieldbusses
A total of 5 tasks are implementing the function of the Engine Control System.

These 5 tasks are executing on the Engine Control System ECU.
Engine Control System

...the scheduler always selects the available and ready task with the highest priority for execution.
ABS System

A total of 5 tasks are implementing the function of the ABS System

These 5 tasks are executing on the ABS System ECU
ABS System

Task Priority

- low
- medium low
- medium
- medium high
- high

time

- task1
- task2
- task3
- task4
- task5
Integration of functions
Number of ECUs
Automotive architectures

- Example network architectures

- Reduce the number of ECUs
  - Trend: fewer and more powerful
Putting 2 functions on 1 ECU

Interference in the temporal domain!
So, we have an integration problem

Possible solution 1: Servers
The integration problem

• Integration problem known for a long time

• What we see here is (temporal) interference inherent in the integration

• 80ies-90ies, techniques have been developed to limit the interference of the unknown

• Server-based scheduling cope with systems of (partly) unknown parameters

• However, server-based scheduling techniques goes only half way...
What is a server?

• Conceptual entity that mediate access to CPU
• Guarantees predictable interference on the rest of the system
• Commonly used to handle aperiodic tasks
• Objective is predictable interference and minimize response-time of aperiodic tasks
Why servers?
A World with servers!
tasks would like to execute

the server dispatch tasks
So, servers provide predictable interference

Can we do better?

Possible solution 2: Hierarchical Scheduling
Hierarchical scheduling
Concept of hierarchical scheduling

"Scheduling inside the server"

"the original system"

a periodic server

"the original system" inside the server
Deriving of server parameters

Working with server parameters: 2 ways

- Some math and algorithms
- Task WCETs
- Task periods
- Period
- Capacity

PROGRESS
Example

Hierarchical scheduling
Example system
The Engine Control

the 5 individual “Engine Control” tasks

the 5 individual tasks executing directly on their ECU

the server executing the 5 tasks
The ABS

te the 5 individual “ABS” tasks


te the 5 individual tasks executing directly on their ECU


te the server executing the 5 tasks
Engine Control + ABS
The Hierarchical Scheduling Framework

...getting under the hood
Basic Scheduling Framework

CPU

Scheduler

Task 1

Task 2
Hierarchical Scheduling

- CPU
- EDF
- RM
  Application 1 (server)
- Task
  Task
- EDF
  Application 2 (server)
- Task
  Task
Hierarchical Scheduling

• Open environments
  – Applications may be developed and validated independently in different environments

• Partitioning
  – Supporting temporal partitioning among applications for fault containment
Hierarchical Scheduling - Issues

• System-level scheduler’s viewpoint

What are the real-time requirements of each application?
Hierarchical Scheduling - Issues

- Application-level scheduler’s viewpoint

Real-time guarantees from CPU supply?

How can we achieve schedulability analysis with this CPU share?
Proposed Framework - Overview

• Interface-based hierarchical scheduling framework
Approaches

• Interface-based hierarchical scheduling framework
• Approach
  – Propose/use a (new) real-time resource model (periodic)
  – Extend real-time scheduling theories with the new resource model
  – Develop interfaces with these results
  – Use interfaces for compositional schedulability analysis
Assumptions

• Tasks
  – periodic
  – independent
  – fully preemptable
  – synchronously released

• Uni-processor scheduling

• Scheduling algorithms: EDF / RM
Real-Time Resource Modeling

- Real-time virtual resource model
  - Characterize the timing property of resource allocations provided to a single task (application/server)

- Previous approaches
  - Rate-based resource model

- Our approach
  - Temporally partitioned resource model
Resource Modeling

• Dedicated resource
  – available at all times at full capacity

• Rate-based shared resource
  – available at fractional capacity at all times

• Time-shared resource
  – available at full capacity at some times
Periodic Resource Model

- Periodic resource model \( \Gamma(P,Q) \)
  - a time-shared resource,
  - characterizes periodic resource allocations
  - period \( P \) and allocation time \( Q \)
  - Resource utilization \( U_{\Gamma} = Q/P \)
  - Example, \( P = 3, Q = 2 \)
Traditional Schedulability Analysis

• Demand-based analysis with dedicated resource

resource demand during an interval of length $t \leq t$

Scheduler

Task  Task
Schedulability Analysis

• Demand- and supply-based analysis with periodic (time-shared) resource

resource demand during an interval of length \( t \)

\( \leq \)

resource supply, during an interval of length \( t \)
Resource Demand Bound

• Resource demand bound function
  – $\text{dbf}(W,A,t)$: the maximum possible resource demand of a task set $W$ under algorithm $A$ during an interval of length $t$
Demand Bound Functions

• For a periodic task set $W = \{T_i(p_i,e_i)\}$,
  
  – $\text{dbf} (W, A, t)$ for EDF [Barua et al., '90]

  $$
  \text{dbf} (W, \text{EDF}, t) = \sum_{Ti \in W} \left[ \frac{t}{p_i} \right] \cdot e_i
  $$

  – $\text{dbf} (W, A, t, i)$ for RM [Lehoczky et al., ‘89]

  $$
  \text{dbf} (W, \text{RM}, t, i) = e_i + \sum_{Tk \in \text{HP}(Ti)} \left[ \frac{t}{p_k} \right] \cdot e_k
  $$
Resource Supply

- Resource supply bound function
  \[ \text{sbf}_\Gamma(t) : \text{the minium resource supply by resource } \Gamma \text{ over all intervals of length } t \]

- Periodic resource \( \Gamma(3,2) \)
Periodic Resource Model

- Supply bound function \( \text{sbf}_\Gamma(t) \)

\[
\text{sbf}_\Gamma(t) = \begin{cases} 
 t - (k+1)(P - Q) & \text{if } t \in [(k+1)P - 2Q, (k+1)P - Q] \\
 (k-1)P & \text{otherwise}
\end{cases}
\]
Schedulability Condition - EDF

- A periodic task set $W$ is schedulable under EDF if and only if over the worst-case resource supply of periodic resource model $\Gamma(P,Q)$

\[
\forall t > 0 \quad dbf(W, EDF, t) \leq t
\]

[Baruah et al. ‘90]

\[
\forall t > 0 \quad dbf(W, EDF, t) \leq sbf(t)
\]

[Shin & Lee, ’03]
Schedulability Condition - RM

- A periodic task set $W$ is schedulable under RM over the worst-case resource supply of periodic resource model $\Gamma(P, Q)$ if and only if [Shin & Lee, ’03]

\[ \exists 0 < t \leq p_i \quad \forall T_i \in W \quad \text{dbf}(W, RM, t, i) \leq \text{sbf}(t) \]
Schedulability Analysis

• Demand- and supply-based analysis

naturally extensible with other schedulers, task models, and/or resource models, as long as they can provide resource demand and supply bounds.
Synchronization

...what if the servers aren’t independent?
Synchronization

• Over the years, there has been a growing attention to hierarchical scheduling of real-time systems.

• Research on hierarchical scheduling frameworks started with the assumption that subsystems are independent.

• Recently, two SRP-based synchronization protocols for resource sharing in FPPS systems have been presented; HSRP and SIRAP.

Global Resource Sharing Problem

- **Task preemption** (use SRP locally)
- **Subsystem preemption** (use SRP globally)
- Budget *expiry inside* critical section
Overrun Mechanism without Payback

Local schedulability analysis

\[ \forall \tau_i, 0 < t \leq D_i \quad rbf_{FP}(i, t) + b_i \leq sbf(t), \]

\[ rbf_{FP}(i, t) = C_i + \sum_{\tau_k \in \text{HP}(i)} \left[ \frac{t}{T_k} \right] \cdot C_k, \]

Global schedulability analysis

\[ \forall S_i, 0 < t \leq P, \quad LBF_s(t) = \text{RBF}_s(t) + D_s \]

\[ \text{RBF}_s(t) = (Q_s + O_s(t)) + \sum_{S_k \in \text{WS}(s)} \left[ \frac{t}{I_k} \right] (Q_k + O_k(t)) \]
Overrun Mechanism with Payback

Local schedulability analysis

\[ \forall \tau_i, 0 < t \leq D_i \quad \text{rbf}_{FP}(i, t) + b_i \leq \text{sbf}^*(t), \]

\[ \text{rbf}_{FP}(i, t) = C_i + \sum_{\tau_k \in \text{HP}(i)} \left[ \frac{t}{P_k} \right] \cdot C_k, \]

Global schedulability analysis

\[ \forall S_s, 0 < t \leq P_s \quad LBF_s(t) \leq t \]

\[ LBF_s(t) = \text{RBF}_s(t) + B_s, \text{ where} \]

\[ \text{RBF}_s(t) = (Q_s + X_s) + \sum_{S_k \in \text{HP}(s)} \left( \left[ \frac{t}{P_k} \right] \cdot Q_k + X_k \right) \]
Skipping

\[ X_j \leq Q^* \]

\[ X_j > Q^* \]

System ceiling = \( \pi(j) \)
Local schedulability analysis

\[ \forall \tau_i, 0 < t \leq D_i \quad rbf_{FP}(i, t) + b_i \leq sbf(t), \]

\[ rbf_i(t) = C_i + I_i^S + I_i^H(t) + b_i, \]

\[ I_i^S = \sum_{R_j \in \{R^i\}} X_{i,j}, \]

\[ I_i^H(t) = \sum_{\tau_k \in \{hp(i) \land r_{c_j} \geq i\}} \left[ \frac{t}{T_k} \right] (C_k + \sum_{R_j \in \{R^k\}} X_{k,j}), \]

\[ b_i = \max_{\tau_k \in \{hp(i) \land r_{c_j} \geq i\}} (2 \cdot (X_{k,j})). \]

Global schedulability analysis

\[ \forall S_s, 0 < t \leq P_s \quad LBF_s(t) \leq t \]

\[ LBF_s(t) = RBF_s(t) + B_s, \text{ where} \]

\[ RBF_s(t) = Q_s + \sum_{S_k \in HP(s)} \left[ \frac{t}{P_k} \right] \cdot Q_k. \]

63
What approach to use when then?

• Results show that skipping can perform better than the overrun mechanism if the task periods are much larger than their subsystem period and the number of shared resources is low.

• Otherwise, for a high difference between subsystems periods, the overrun with payback can give better results.

• For equal or close to equal subsystem periods, the overrun without payback performs better.
From theory to practice
Towards Hierarchical Scheduling on top of VxWorks, Moris Behnam, Thomas Nolte, Insik Shin, Mikael Åsberg (MRTC), Reinder J. Bril (Eindhoven University of Technology, The Netherlands), Proceedings of the Fourth International Workshop on Operating Systems Platforms for Embedded Real-Time Applications (OSPERT'08), p 63-72, Prague, Czech Republic, July, 2008
Summary
Why hierarchical scheduling?

- **Isolation**: whatever happens inside a server stays there
- **Predictable execution**: given an interface, internally, tasks are scheduled as on an exclusive CPU
- **IP protection**: no need for system wide task level analysis, i.e., no need to reveal server internals
Some Open Issues

• Multiprocessor
  – Resource models
    • Bini et al., RTCSA 2009
  – Synchronization
    • Nemati et al., XRTS 2009
    • Behnam et al., ETFA 2009

• Events
  – XRTS 2009, Vienna, Austria, September 22.
  – CRTS 2009, in conjunction with RTSS 2009,
    Washington, DC, USA, December 1.

www.rtss.org
Hierarchical Scheduling of Complex Embedded Real-Time Systems

Thank you for your attention

Thomas Nolte,
Associate Professor (MRTC/Mälardalen University),
www.mrtc.mdh.se  thomas.nolte@mdh.se