Optimizing Inter-Core Data-Propagation Delays in Industrial Embedded Systems under Partitioned Scheduling

Lamija Hasanagić¹
Tin Vidović¹
Saad Mubeen¹
Mohammad Ashjaei¹
Matthias Becker²

¹Mälardalen University, Sweden
²KTH Royal Institute of Technology, Sweden

(2021-01-20, Tokyo, Japan)
Agenda

- Motivation and Background
- System Model
- Optimization Technique
- Evaluation
- Conclusions and Future Work
Motivation and Background

- The requirements for high computational power lead to multi-core architectures being employed
- Legacy systems need to retain functionality
- Single-core timing analysis no longer applicable
- Temporal isolation when accessing shared resources
- Inter-Core Data-Propagation Delays not optimized
Motivation and Background

- **Read-Execute-Write Model**
- Tasks split into:
  - Two memory access phases
  - One computational phase
- Contention for shared resources can be avoided by design
Motivation and Background

- Schedulability Analysis
- Worst-Case Execution Time Analysis
- Response-Time Analysis
- End-to-End Data-Propagation Delays
- Task Chains - sequences of tasks executing one after another

![Diagram showing task chains and data propagation delays]
Motivation and Background

- **Inter-Core Data-Propagation Delays**
  - Latency between the time a variable at the consumer is read and the time the variable was last written by the producer

- **Rubus-ICE tool suite**
  - Used in the automotive industry for over 25 years
Contributions

- Constraint Programming Scheduling Technique
  - Read-Execute-Write structured tasks
  - Partitioned Scheduling
  - Inter-Core Data-Propagation Delays Optimized
- Extensive Evaluation
  - Scalability
  - Schedulability Ratio
- Automotive Industrial Case Study
Agenda

- Motivation and Background
- System Model
- Optimization Technique
- Evaluation
- Conclusions and Future Work
System Model: Platform Model

- Cores operate synchronously with the system clock
- Shared memory accessed in a 32 bit granularity
- Based on several COTS microcontroller architectures
  - MPC5675K
  - MPC5777C
System Model: Application Model

- N tasks statically allocated to M cores
- Register-communication paradigm
- Inter-Core communication between two tasks $\tau_i$ and $\tau_k$ denoted by $\psi^i_k$
Agenda

- Motivation and Background
- System Model
- Optimization Technique
- Evaluation
- Conclusions and Future Work
Optimization Technique: Interval Decision Variables

- One interval decision variable for each phase of a task instance stored in a set $J_m$
- One interval decision variable for the entire instance
Optimization Technique: Integer Decision Variables

- For each potentially communicating producer-consumer pair $(\tau_{ij}, \tau_{kl})$

- $Z_{kl}^{ij} = \text{startOf}(\tau_{kl}) - \text{endOf}(\tau_{ij})$
Optimization Technique: Integer Decision Variables

- An instance of a consumer can potentially communicate with many producers
- In the final schedule each consumer instance actually communicates with exactly one producer instance

\[ \tau_{2,1}: \quad Z_{2,1}^{1,1} = s(\tau_{2,1}^{R}) - e(\tau_{1,1}^{W}) \]

\[ H_{2,1}^{1,1} = Z_{2,1}^{1,1} \]

\[ \tau_{2,2}: \quad Z_{2,2}^{1,1} = s(\tau_{2,2}^{R}) - e(\tau_{1,1}^{W}), \quad Z_{2,2}^{1,2} = s(\tau_{2,2}^{R}) - e(\tau_{1,2}^{W}), \quad Z_{2,2}^{1,3} = s(\tau_{2,2}^{R}) - e(\tau_{1,3}^{W}) \]

\[ H_{2,2}^{1,1} = 0, \quad H_{2,2}^{1,2} = 0, \quad H_{2,2}^{1,3} = Z_{2,2}^{1,3} \]
Optimization Technique: Basic Scheduling Constraints

- Read phases should be scheduled at least 0 time units before the execute phases.
- Execute phases should be scheduled at least 0 time units before the write phases.

![Diagram showing scheduling constraints](image-url)
Optimization Technique: Basic Scheduling Constraints

- Task instances on the same core are not allowed to overlap

\[ \text{noOverlap}(\tau_{1,1}^{\text{All}}, \tau_{1,2}^{\text{All}}, \tau_{1,3}^{\text{All}}, \tau_{2,1}^{\text{All}}, \tau_{2,2}^{\text{All}}) \]
Optimization Technique: Memory Access Constraints

- Temporal isolation when accessing the shared memory

\[
\text{noOverlap}(\tau_{1,1}^R, \tau_{1,1}^W, \tau_{1,2}^R, \tau_{1,2}^W, \tau_{1,3}^R, \tau_{1,3}^W, \tau_{2,1}^R, \tau_{2,1}^W, \tau_{2,2}^R, \tau_{2,2}^W)
\]
The main objective is to find the optimal schedule from the perspective of inter-core data-propagation delays.

\[ f(x) = \arg\min \sum_{\forall H_{kl} \in H} H_{ij} \]
Agenda

- Motivation and Background
- System Model
- Optimization Technique
- Evaluation
- Conclusions and Future Work
Evaluation: Synthetic Experiments

- Synthetic Test Case Generator
  - Based on recommendations in [2]

Evaluation: Synthetic Experiments

- CP problem size
  - Number of tasks: 60
  - Number of cores: [2, 4, 8]
  - Core utilization: 50%
  - Inter-core communications: [0, 2, 4, 8, 16, 32, 64, 128]
Evaluation: Synthetic Experiments

![Graph showing the average number of constraints with respect to the number of communication pairs for different core counts.

- Blue line: 2 Cores
- Orange line: 4 Cores
- Green line: 8 Cores

The graph indicates that as the number of communication pairs increases, the average number of constraints also increases, with 8 Cores showing the highest growth rate compared to 2 Cores and 4 Cores.]
Evaluation: Synthetic Experiments

- CP problem size
  - Number of tasks: [15, 20, 25, 30, 35, 40, 45, 50, 55, 60]
  - Number of cores: 2
  - Core utilization: 50%
  - Inter-core communications: 16
Evaluation: Synthetic Experiments

![Graph showing the average number of constraints vs. the number of tasks.

- Y-axis: Average number of constraints (ranging from 0 to 2500).
- X-axis: Number of tasks (ranging from 15 to 60).

The graph indicates a linear relationship, with the number of constraints increasing as the number of tasks increases.]
Evaluation: Synthetic Experiments

- Schedulability ratio
  - Number of tasks: 30
  - Number of cores: 2
  - Core utilization: [10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%]
  - Inter-core communications: 16
Evaluation: Synthetic Experiments

![Graph showing schedulability ratio vs U Core 1 for different U Core 0 values.]

- U Core 0 = 0.1
- U Core 0 = 0.2
- U Core 0 = 0.3
- U Core 0 = 0.4
- U Core 0 = 0.5
- U Core 0 = 0.6
- U Core 0 = 0.7
- U Core 0 = 0.8
- U Core 0 = 0.9
Evaluation: Automotive Application Case Study

- Based on a model industrial engine-control application

<table>
<thead>
<tr>
<th>Task</th>
<th>$C_{Ri}[\text{ns}]$</th>
<th>$C_{Ei}[\text{ns}]$</th>
<th>$C_{Wi}[\text{ns}]$</th>
<th>$C_i[\text{ns}]$</th>
<th>$T_i[\text{ms}]$</th>
<th>$p_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>CylNumObserver</td>
<td>908</td>
<td>573000</td>
<td>25</td>
<td>573933</td>
<td>1000</td>
<td>0</td>
</tr>
<tr>
<td>IgnitionSync</td>
<td>958</td>
<td>2461000</td>
<td>195</td>
<td>2462153</td>
<td>1000</td>
<td>1</td>
</tr>
<tr>
<td>MassAirFlow</td>
<td>908</td>
<td>86000</td>
<td>28</td>
<td>86936</td>
<td>50</td>
<td>0</td>
</tr>
<tr>
<td>ThrottleSensor</td>
<td>908</td>
<td>169000</td>
<td>55</td>
<td>169963</td>
<td>50</td>
<td>1</td>
</tr>
<tr>
<td>APedSensor</td>
<td>908</td>
<td>482000</td>
<td>55</td>
<td>482963</td>
<td>50</td>
<td>0</td>
</tr>
<tr>
<td>APedVoter</td>
<td>55</td>
<td>144000</td>
<td>28</td>
<td>144083</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>ThrottleCtrl</td>
<td>990</td>
<td>2892000</td>
<td>55</td>
<td>2893045</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>ThrottleActuator</td>
<td>1013</td>
<td>2957000</td>
<td>83</td>
<td>2958096</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>BaseFuelMass</td>
<td>990</td>
<td>2892000</td>
<td>55</td>
<td>2893045</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>ThrottleChange</td>
<td>1013</td>
<td>2957000</td>
<td>83</td>
<td>2958096</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>TransFuelMass</td>
<td>1148</td>
<td>3188000</td>
<td>28</td>
<td>3189176</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>Ignition</td>
<td>1060</td>
<td>2269000</td>
<td>25</td>
<td>2270085</td>
<td>1000</td>
<td>0</td>
</tr>
<tr>
<td>TotalFuelMass</td>
<td>988</td>
<td>677000</td>
<td>28</td>
<td>678016</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>OperatingMode</td>
<td>965</td>
<td>19641000</td>
<td>390</td>
<td>19642355</td>
<td>200</td>
<td>1</td>
</tr>
<tr>
<td>IdleSpeedCtrl</td>
<td>833</td>
<td>843000</td>
<td>240</td>
<td>844173</td>
<td>200</td>
<td>0</td>
</tr>
<tr>
<td>APedSensorDiag</td>
<td>908</td>
<td>118000</td>
<td>0</td>
<td>118908</td>
<td>1000</td>
<td>0</td>
</tr>
<tr>
<td>InjBattVoltCorr</td>
<td>28</td>
<td>274000</td>
<td>28</td>
<td>274056</td>
<td>1000</td>
<td>0</td>
</tr>
</tbody>
</table>
Based on a model industrial engine-control application
Evaluation: Automotive Application Case Study

- **Start Time Jitter**

<table>
<thead>
<tr>
<th>Task</th>
<th>Jitter CP [ms]</th>
<th>Jitter RCM [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CylNumObserver</td>
<td>1.95</td>
<td>0</td>
</tr>
<tr>
<td>IgnitionSync</td>
<td>8.04</td>
<td>0</td>
</tr>
<tr>
<td>MassAirFlow</td>
<td>7.48</td>
<td>0</td>
</tr>
<tr>
<td>ThrottleSensor</td>
<td>1.65</td>
<td>0</td>
</tr>
<tr>
<td>APedSensor</td>
<td>0.39</td>
<td>0</td>
</tr>
<tr>
<td>APedVoter</td>
<td>1.81</td>
<td>0</td>
</tr>
<tr>
<td>ThrottleCtrl</td>
<td>3.59</td>
<td>0</td>
</tr>
<tr>
<td>ThrottleActuator</td>
<td>3.04</td>
<td>0</td>
</tr>
<tr>
<td>BaseFuelMass</td>
<td>42.61</td>
<td>0</td>
</tr>
<tr>
<td>ThrottleChange</td>
<td>8.06</td>
<td>0</td>
</tr>
<tr>
<td>TransFuelMass</td>
<td><strong>43.67</strong></td>
<td>0</td>
</tr>
<tr>
<td>Ignition</td>
<td>2.53</td>
<td>0</td>
</tr>
<tr>
<td>TotalFuelMass</td>
<td>3.04</td>
<td>0</td>
</tr>
<tr>
<td>OperatingMode</td>
<td>1.29</td>
<td>0</td>
</tr>
<tr>
<td>IdleSpeedCtrl</td>
<td>0.31</td>
<td>0</td>
</tr>
<tr>
<td>APedSensorDiag</td>
<td>0.09</td>
<td>0</td>
</tr>
<tr>
<td>InjBattVoltCorr</td>
<td>0.21</td>
<td>0</td>
</tr>
<tr>
<td>Injection</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Evaluation: Automotive Application Case Study

- Inter-Core Data-Propagation Delays

<table>
<thead>
<tr>
<th>Chain</th>
<th>(Producer, Consumer)</th>
<th>CP[ms]</th>
<th>Rubus-ICE[ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chain A</td>
<td>(APedVoter, ThrottleCtrl) (ThrottleCtrl, ThrottleActuator)</td>
<td>0</td>
<td>54.86</td>
</tr>
<tr>
<td>Chain B</td>
<td>(ThrottleCtrl, ThrottleActuator)</td>
<td>0</td>
<td>91.43</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>47.21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>1.38</td>
</tr>
</tbody>
</table>

Diagram:

- Red: Chain A
- Green: Chain B
- Blue: Chain C

Nodes:
- APedeSensor
- MassAirFlow
- APedVoter
- Throttle Sensor
- ThrottleCtrl
- Throttle Actuator
- BaseFuel Mass
- TransFuel Mass
- TotalFuel Mass
- Injection
Evaluation: Automotive Application Case Study

- Data Age Delays

<table>
<thead>
<tr>
<th>Chain</th>
<th>CP[ms]</th>
<th>Rubus-ICE[ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chain A</td>
<td>56.49</td>
<td>153.53</td>
</tr>
<tr>
<td>Chain B</td>
<td>9.61</td>
<td>100.4</td>
</tr>
<tr>
<td>Chain C</td>
<td>101.65</td>
<td>152.06</td>
</tr>
</tbody>
</table>

![Diagram showing data flow and node interactions](image)
Agenda

- Motivation and Background
- System Model
- Optimization Technique
- Evaluation
- Conclusions and Future Work
Conclusions and Future Work

- CP model is developed for the scheduling of real-time embedded system applications
- The generated schedule optimized with respect to inter-core data-propagation delays
- CP problem size scales linearly
- 18% of task sets schedulable with 90% core utilizations
- Industrial case study, demonstrates the applicability of the proposed technique
- Incorporate data-propagation delays constraints of task chains
- Optimization of jitter
Thanks!
Related Work

- Maiza et al.\textsuperscript{[1]} categorize research into timing verification of multi-core systems:
  - Full integration
  - Integrating interference effects into schedulability analysis
  - Mapping and Scheduling
  - \textbf{Temporal isolation}

- Several works focus on generating schedules using resource optimization techniques
  - The CP approach is shown to scale better than the Integer Linear Programming approach