An Improved Scheduling Technique for Time-Triggered Embedded Systems

Paul Pop, Petru Eles, Zebo Peng

Dept. of Computer and Information Science
Linköping University
Sweden
Outline

- Motivation
- System Architecture
- Problem Formulation
- Scheduling Strategy
- Experimental Results
- Conclusions
Motivation

- Embedded System Design.
- Scheduling, Communication, Bus Access.

Characteristics:

- Static nonpreemptive scheduling.
- System model captures both the flow of data and that of control.
- Heterogeneous system architecture.
- Communications using the time-triggered protocol (TPP).

Message:

- Improved schedule quality by considering the characteristics of the communication protocol.
Hardware Architecture

- Safety-critical distributed embedded systems.
- Nodes interconnected by a broadcast communication channel.
- Nodes consisting of: TTP controller, CPU, RAM, ROM, I/O interface, (maybe) ASIC.
- Communication between nodes is based on the time-triggered protocol.

- Bus access scheme: time-division multiple-access (TDMA).
- Schedule table located in each TTP controller: message descriptor list (MEDL).
• Real-Time Kernel running on the CPU in each node.

• There is a local schedule table in each kernel that contains all the information needed to take decisions on activation of processes and transmission of messages.

• Time-Triggered System: no interrupts except the timer interrupt.

• The worst case administrative overheads (WCAO) of the system calls are known:

\[
\begin{align*}
U_t & \quad \text{WCAO of the timer interrupt routine} \\
\delta_{PA} & \quad \text{process activation overhead} \\
\delta_S & \quad \text{overhead for sending a message on the same node} \\
\delta_{KS} & \quad \text{overhead for sending a message between nodes} \\
\delta_{KR} & \quad \text{overhead for receiving a message from another node}
\end{align*}
\]
Problem Formulation

Input

• Safety-critical application with several operating modes.
• Each operating mode is modelled by a conditional process graph.
• The system architecture and mapping of processes to nodes are given.
• The worst case delay of a process is known:

$$T_{P_i} = \left( \delta_{PA} + t_{P_i} + \theta_{C_1} + \theta_{C_2} \right)$$

$$\theta_{C_1} = \sum_{i=1}^{N_{out}^{local} (P_i)} \delta_{S_i} \quad \theta_{C_2} = \sum_{i=1}^{N_{out}^{remote} (P_i)} \delta_{KS_i} + \sum_{i=1}^{N_{in}^{remote} (P_i)} \delta_{KR_i}$$

Output

• Local schedule tables for each node and the MEDL for the TTP controllers.
• Delay on the system execution time for each operating mode, so that this delay is as small as possible.
Scheduling Example

24 ms

Round 1 | Round 2 | Round 3 | Round 4 | Round 5
---|---|---|---|---
$P_1$ | $m_1$ | $P_2$ | $m_2$ | $P_3$ | $m_3$ | $P_4$ | $m_4$

22 ms

Round 1 | Round 2 | Round 3 | Round 4
---|---|---|---
$S_0$ | $S_1$ | $m_1$ | $P_2$ | $m_2$ | $P_3$ | $m_3$ | $P_4$ | $m_4$

20 ms

Round 1 | Round 2 | Round 3
---|---|---
$S_0$ | $S_1$ | $m_1$ | $m_2$ | $P_2$ | $P_3$ | $m_3$ | $m_4$ | $P_4$
1. The scheduling algorithm has to take into consideration the TTP.
   - priority function for the list scheduling

2. The optimisation of the TTP parameters is driven by the scheduling.
   - sequence and lengths of the slots in a TDMA round are determined to reduce the delay
   - two approaches: Greedy heuristic, Simulated Annealing (SA).
   - two variants: Greedy 1 tries all possible slot lengths, Greedy 2 uses feedback from the scheduling algorithm.
   - SA parameters are set to guarantee near-optimal solutions in a reasonable time.
Partial Critical Path Scheduling

\[ L_{PA} = \max(T_{curr} + t_A + \lambda_A, T_{curr} + t_A + t_B + \lambda_B) \]
\[ L_{PB} = \max(T_{curr} + t_B + \lambda_B, T_{curr} + t_B + t_A + \lambda_A) \]

Select the alternative with the smaller delay:

\[ L = \max(L_{PA}, L_{PB}) \]

\[ \lambda_A > \lambda_B \Rightarrow L_{PA} < L_{PB} \]
\[ \lambda_B > \lambda_A \Rightarrow L_{PB} < L_{PA} \]

Use \( \lambda \) as a priority criterion.
Priority Function Example

Round 1

<table>
<thead>
<tr>
<th>P_1</th>
<th>P_2</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>S_0=10</td>
<td>S_1=8</td>
<td></td>
</tr>
</tbody>
</table>

Round 2

<table>
<thead>
<tr>
<th>P_3</th>
<th>P_4</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td></td>
</tr>
</tbody>
</table>

Round 1

<table>
<thead>
<tr>
<th>P_2</th>
<th>P_1</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td></td>
</tr>
</tbody>
</table>

Round 2

<table>
<thead>
<tr>
<th>P_3</th>
<th>P_4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

40 ms

36 ms
Experimental Results

Average percentage deviations from the lengths of the best schedule between PCP and PCP2

- PCP
- PCP2 has knowledge about TTP
The Greedy approach is producing accurate results in a very short time (few seconds for graphs with 400 processes).

Greedy 1 produces better results than Greedy 2 (but it is slightly slower).

SA finds near-optimal results in a reasonable time.

A real-life example implementing a vehicle cruise controller validated our approach.
Conclusions

• An approach to process scheduling for the synthesis of safety-critical distributed embedded systems.

• Communication of data and conditions based on TTP.

• Scheduling algorithm tailored to the communication protocol.

• Communication has been optimised through packaging of messages into slots with a properly selected order and lengths.

• Improved schedule quality by considering the overheads of the real-time kernel and of the communication protocol.

• Evaluation based on experiments using a large number of graphs generated for experimental purpose as well as real-life examples.