Scheduling with Optimized Communication for Time-Triggered Embedded Systems

Paul Pop, Petru Eles and Zebo Peng

Dept. Of Computer and Information Science
Linkoping University
Sweden
Outline

- Motivation
- Conditional Process Graph
- System Architecture
- Problem Formulation
- Scheduling Example
- Scheduling Strategy
- Experimental Results
- Conclusions
System-Level Specification

Architecture Selection

Partitioning

High-level Synthesis

Compilation

Integration

Motivation

- System model captures both the flow of data and that of control.
- Heterogeneous system architecture: nodes connected by a broadcast communication channel.
- Communication of conditions and messages considered for a time-triggered protocol (TTP) implementation.
- Improved schedule quality by considering the characteristics of TTP and the overheads of the real-time kernel.
- Scheduling algorithms proposed can be used both for performance estimation and for system synthesis.
Conditional Process Graph

Subgraph corresponding to $D \land C \land K$

- First processor
- Second processor
- ASIC
Hardware Architecture

- Safety-critical distributed embedded systems.
- Nodes connected 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:

\[
T_{P_i} = (\delta_{PA} + t_{P_i} + \theta_{C_1} + \theta_{C_2})
\]

\[
\theta_{C_1} = \sum_{i=1}^{N_{\text{local}}(P_i)} \delta_{S_i}
\]

\[
\theta_{C_2} = \sum_{i=1}^{N_{\text{out}}(P_i)} \delta_{KS_i} + \sum_{i=1}^{N_{\text{in}}(P_i)} \delta_{KR_i}
\]

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

Round 1 | Round 2 | Round 3 | Round 4 | Round 5
---|---|---|---|---
P1 | P2 | P3 | P4
24 ms

Round 1 | Round 2 | Round 3 | Round 4
---|---|---|---
P1 | P2 | P3 | P4
22 ms

Round 1 | Round 2 | Round 3
---|---|---
P1 | P2 | P3 | P4
20 ms

Scheduling with Optimized Communication for Time-Triggered Embedded Systems
Scheduling Strategy


- Previous work extended to handle scheduling of messages within TTP for a given TDMA configuration: schedule_message.

- 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 schedule_message function.

- SA parameters are set to guarantee finding near-optimal solutions in a reasonable time.
Experimental Results

- The Greedy Approach is producing accurate results in a very short time (few seconds for graphs with 400 processes).
- Greedy 1 performs slightly better than Greedy 2, but it is a bit slower.
- SA finds near-optimal results in a reasonable time (few minutes for graphs with 80 processes and 275 minutes for graphs with 400 processes).
- 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.

- Process level representation which captures both data flow and the flow of control.

- Communication of data and conditions based on TTP.

- Communication has been optimized 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.