Model based Generation of Static Schedules for Safety-Critical Multi-Core Systems

Robert Hilbrich
Outline

1. Project Goals
2. Approach
3. Tool-Demo
4. Future Work & Conclusions
Model based Generation of Static Schedules for Multi-Core Systems

PROJECT GOALS
Background: Partition Scheduling in Avionics

- Avionics = Aviation Electronics
- RTCA ARINC 653 Architecture
- Static Schedule
Research Project

Starting Point:
- 25 applications, 4 processors (Single-Core)
- 1 application = 10 – 30 Developers
- Schedule creation with Excel (1PM!)
- Achievable system load approx. 50 %
  - Many degrees of freedom
  - Functional dependencies
  - Varying timing requirements, …

Multi-Core Future:
- Software gets more complex
  (A380: >400MB Software, >400 000 Signal Interfaces)
- Multi-Core increases functional density and the use of shared ressources
- **Manual** generation becomes infeasable
Model based Generation of Static Schedules for Multi-Core Systems

APPROACH
Basic Idea – “Correctness by Construction”

Application Requirements
- Duration, period
- Dependencies
- Resource usage

Allocation
Which application runs on which processor?

Hardware Characteristics
- Processors, cores
- Shared resources (e.g. AFDX, energy, ...)

Scheduling Tool

Static Schedule
- Satisfies all requirements
- Operating system independent
Modeling Notation - Example

def_basic
    scheduling_period(200),
    ...

def_processor
    id('P.1'),
    cores(2).
    ...

def_partition
    id('A.1'),
    processor('P.1'),
    parallel(2),
    duration(1, 5),
    duration(2, 10),
    period(1,20),
    period(2,40),

Homogeneous Multi-Core
Benefits

- Project specific:
  - Higher system load (>90%)
  - Shortened engineering cycles
  - Lower cost

- Beyond:
  - Early resource estimation
  - Early evaluation of architectures
Model based Generation of Static Schedules for Multi-Core Systems

DEMO
Simple Example: Brake Assistance

Goal: keep a safe distance to vehicle in front

Application consists of three tasks:
- Task 1: "Distance measurement"
- Task 2: "Speedometer"
- Task 3: "Reaction"

Input specification for PRECISION PRO:
- Hardware: Dual-Core architecture (Core 1, Core 2)
- Software: Dependency - Task 3 ('Reaction') requires distance and speed values
- Software: Duration - Task 1 needs 1.3ms, Task 2 need 2.7ms, Task 3 need 2.0ms
- Allocation: Task 1 and Task 3 → Core 1, Task 2 → Core 2
Demo
Model based Generation of Static Schedules for Multi-Core Systems

FUTURE WORK
Certification support by Model Checking

Application Requirements → Allocation → Hardware Characteristics

Schedule Validator

Static Schedule
Automated Application Mapping

- Application Requirements
- Allocation
- Hardware Characteristics

Mapper

Scheduling Tool

Schedule Validator

Static Schedule
Summary

- Goal: Correctness by Construction
- Formalized but simple model
- Quick generation of static schedules
  - Every solution is valid(!)
- Homogeneous Multi-Core Support
- Extensions: Validator & Mapping
Thank you for your attention

Robert Hilbrich
robert.hilbrich@first.fraunhofer.de

Fraunhofer FIRST
Department Embedded Systems