Virtual Platforms for Mixed Time-Criticality Applications: The CoMPSoC Architecture and SDF$^3$ Design Flow

Benny Åkesson, Sander Stuijk, Anca Molnos, Martijn Koedam, Radu Stefan, Andrew Nelson, Ashkan Beyranvand Nejad, and Kees Goossens

http://compsoc.eu

Introduction

• SoC design gets increasingly complex
• Exponential increase in number of use-cases
  - Number of applications is growing
  - Increasing application-level parallelism
• Applications have mixed time-criticality
  - Mix of firm, soft, and no real-time requirements
  - All requiring different design and verification methods
• Resources are shared to reduce cost

Problem Statement

• Resource sharing causes temporal interference
  - Applications become inter-dependent
• Three problems with design and verification
  1. Many verification approaches are not applicable
  2. Use-case verification becomes a circular process
  3. Difficult to support independent design flows
• Problems make design and verification costly!

CoMPSoC Platform

• CoMPSoC addresses the problem by
  - executing applications in virtual platforms
  - using the SDF$^3$ design flow to map applications to virtual platforms
• Virtualization based on concepts of composability and predictability
• Composability
  - Virtualization of actual execution time
  - Removes all inter-application interference
  - Enables independent design, verification & execution
• Predictability
  - Virtualization of worst-case execution time (WCET)
  - Budgets enable independent formal verification

Platform Architecture

• Platform has a tiled architecture
  - MicroBlaze tiles with CompOSe RTOS
  - Memory tiles with SRAM and SDRAM (Predator)
  - Interconnected by Æthereal NoC
• Techniques for composability
  1. Preemption prevents starvation
  2. Delay to WCET eliminates request dependencies
  3. Composable schedulers, e.g. TDM
• Techniques for predictability
  1. Worst-case analysis per resource
  2. Resources with bounded response time
  3. Predictable schedulers, e.g. TDM, RR, CCSP
• Platform instantiated on Xilinx ML605 FPGA

SDF$^3$ Design Flow

• Flow is based on dataflow model of computation
  - Applications modeled as CSDF graphs
  - Actors model tasks and edges model dependencies
• Steps in design flow:
  1. Bind actors to platform resources
  2. Analyzes trade-off between storage and timing constraints
  3. Find minimum schedules that satisfy requirements
  4. Generate C-code for actors and configuration