Predictable SoC architecture based on COTS multi-core

Nitin Shivaraman, Sriram Vasudevan, Arvind Easwaran

Nanyang Technological University

April 12, 2016
Motivation

Why COTS multi-cores?

1. Size, Weight and Power (SWaP) benefits,
2. Service history and
3. Increased complexity at the application level.
State of the art

**WCET Multi-core**

- **Unmodified middleware and custom hardware (HW)**
  - Caches (Random Replacement & PEG-C)
  - Scratchpads (MERASA, PRET & Dynamic SPM)
  - Shared Bus (MERASA, PRET & Dynamic SPM)
  - Shared Bus (Probabilistic bus design & MERASA)

- **COTS HW & Unmodified Middleware**
  - Caches (Timing Analysis of concurrent programs, WCET analysis of instruction cache hierarchies and using bypass to tighten WCET)
  - Shared Bus (Multi-core interference-sensitive WCET Analysis Leveraging run-time resource capacity enforcement)

- **COTS HW & Modified Middleware**
  - PREM (Time predictable execution, Single core equivalence)
  - Cache locking/partitioning (Exploring locking & partitioning, Co-ordinated cache management, semi-partitioned Hard real-time scheduling, Making shared caches more predictable on multi-core platforms)
Proposed architecture

A hybrid Solution

1. Retain as much COTS components as possible.
2. Disable all the shared resources.
3. Implement shared resources with predictable sharing policies on the FPGA.
4. Xilinx Zynq 706 platform is used to demonstrate the feasibility of the architecture.
From the graph we can observe a consistent latency to access the external memory when compared against a COTS processor.
Future work

What next?

1. Replace MIG with custom controller.
2. Implement a cache mechanism.
3. Develop interconnect designs for such platforms.
Thank You

Sriram Vasudevan
Nanyang Technological University,
e-mail: sriram006@e.ntu.edu.sg