SW Debugging for Multi-tile Systems: The EURETILE Methodology and Tools

Luis Gabriel Murillo, Rainer Leupers

MAD Workshop, HiPEAC Fall CSW
9.10.14, Athens, Greece
EURETILE Overview

- **EURETILE**: EUropean REference TIled architecture Experiment ([www.euretile.eu](http://www.euretile.eu))
  - FET Concurrent Tera-Device Computing (FP7)
  - 6M EUR
- **Duration**: 2010 – 2014
- **Partners**:

  ![Partners Logos]

- **Goal**:
  - Brain-inspired and fault-tolerant foundational innovations on massively parallel tiled architectures
  - Corresponding programming paradigm
EURETILE, How to Debug?

DAL Programming Model
- Process Code (C)
- App. Model
- Execution Scenarios

Architecture
- Cluster
- Cluster

DAL Programming Model
- P
- W
- C
- APP1
- APP2
- APP3
- APP4

DSE
- App. Refinement
- Process Duplication
- Mapping Optimization

DSE
- App. Refinement
- Process Duplication
- Mapping Optimization

SW Synthesis
- DNA-OS
- Component Selection
- Driver
- COM
- HAL

Process Synthesis
- Runtime Manager
- Bootstrap Code

Targets
- Embedded Style (VEP)
- HPC Style (QUonG)

Binary
- Embedded Style (VEP)
- HPC Style (QUonG)
Agenda

Introduction

The Virtual EURETILE Platform

Tools for Whole-system Debugging

Concurrency Event Monitors and Concurrency Analysis

Conclusions
**Advantages**
- Early availability
- Run unmodified target SW binary
- Optimal for debugging concurrency issues
- Non-intrusive inspection and reproducibility

**VEP Supporting Technologies**
- Multiple levels of abstraction
- 2 parallel SystemC kernels
  - parSC (2.2x in quad-core)
  - SCope (4x in quad-core)
- Distributed SystemC (diSC)
  - Runs on multiple hosts
- Fault Injection

---

**Graph**

- CA: cycle accurate
- IAF: instruction accurate JIT-CC, full
- IAP: instruction accurate JIT-CC, plain (no debug)
- IAD: instruction accurate, DBT
- AED: abstract execution device (host-compiled)
VP Debugging Features

- **Traditional debug augmentations**
  - System loggers: single- or multi-file, tile and component filtering, packets over the network, SW, buses, memories, peripherals...
  - GDB coupling

… but many GDB windows, huge traces…
Agenda

- Introduction
- The Virtual EURETILE Platform
- Tools for Whole-system Debugging
  - Concurrency Event Monitors and Concurrency Analysis
- Conclusions
WSDB: Whole-system Debugger

- Source Debugger Back-end
  - Single interface for:
    - Inspection/control of multiple cores
  - Different targets, core ABIs, unwinders and OS-trackers
- Component-based architecture for portability and extensibility
- C++ and SWIG tcl APIs
- Command-line interface
- Network protocol (with Eclipse CDT/DSF plug-in)
- Can be linked into the VP
- Multi-tile program analysis, concurrency and HW/SW bugs
- Easy way to capture user knowledge, covering:
  - SW contexts (thread, process), variables
  - HW devices, signals, registers
  - Concurrency-related events (e.g., OS events)
  - Linear Temporal Logic (LTL)

... in a single non-intrusive assertion!

```c
assert thread $a=Task1, $b=Task2
scope $s=$a@init.cpp:198
never rread($a::mInit)
before incl req $s
```

**SWAT: Language for System-Wide Assertions**

- Debug inspection / control
- Tasks, threads, cores
- HW signals and registers
- C software elements

```c
... if(tile1)
var1=5;
else if(tile2)
var1=8;
23:...
```
The SWAT Language

Examples

```
assert thread $a=Task1, $b=Task2
scope $s=$a@init.cpp:198
never rread($a::mInit)
  before incl req $s

assert iterator tile it
scope r = it@irq_mask_and_backup.c:6
  s = it@irq_mask_restore:entry
always if lwrite( r::irq_stat ) then next s
```
Introduction

The Virtual EURETILE Platform

Tools for Whole-system Debugging

Concurrency Event Monitors and Concurrency Analysis

Conclusions
Concurrency Event Monitors

- High-level events for analysis but fully trackable to origins
  - Approach
    - Grouping of low level events into programmer-relevant events
    - Propagation of semantic information to higher-level trace
Concurrency Analysis: Ordering Constraints Detection

- High-level trace reveals the order of dependent events
- Analyzes: happens-before, shared resource (visit/modify), dependency, domination, false-dependency
- Bug Exploration: ordering constraint swap

- Drawback:
  - VP slowdown
  - ~2x-30x for the VEP

Example:
* 2016 High-level events
* 2,3x slowdown

No source code instrumentation, no changes to target SW…
Agenda

Introduction

The Virtual EURETILE Platform

Tools for Whole-system Debugging

Concurrency Event Monitors and Concurrency Analysis

Conclusions
Conclusions

- **Debuggers for multi-tile systems:**
  - Facilitate intuitive ways to deal with problems at system-level
  - Present information to developer at the right abstraction
  - Consider different concurrent concurrent interleavings

- **EURETILE’s debugging infrastructure:**
  - Virtual Platform in the loop
  - Debugger able to control/inspect all the tiles and correlate inter-tile data
  - Framework for non-intrusive system-wide assertions
  - Programmer-level (DAL) monitoring framework with concurrency analysis
Thanks!
&
Questions?


Roberto Ammendola, et.al. “Design and implementation of a modular, low latency, fault-aware, FPGA-based Network Interface” in ReConFig 2013


L. G. Murillo, J. Harnath, R. Leupers and G. Ascheid. Scalable and Retargetable Debugger Architecture for Heterogeneous MPSoCs. System, Software, SoC and Silicon Debug Conference (S4D ’12), Sep 2012, Vienna Austria


X. Guerin and F. Pétrot, A System Framework for the Design of Embedded Software Targeting Heterogeneous Multi-core SoCs, in ASAP 2009