Exploiting Dynamically Changing Parallelism with a Reconfigurable Array of Homogeneous Sub-cores (a.k.a. Field Programmable Core Array or FPCA)

Sponsored by SRC and NSF as a Part of Multicore Chip Design and Architecture (MCDA) Program

Department of Computer Science
University of Virginia

P. Wu, B. Calhoun and J. Lach
Department of Electrical & Computer Engineering
University of Virginia
Motivation

Limits of Fixed/Homogeneous Architectures

• **Problem**: execution characteristics change over time within single application or vary across different applications

• **Consequence**: suboptimal performance
Motivation

Limited Homogeneous Parallelism

- **Problem**: limited TLP/DLP and increasing communication patterns become mismatched with scaled hardware
- **Consequence**: diminishing incremental performance
**Approach**

**Splitting Core Into FE and PE**

- **Concept:**
  - Create sub-cores: split IO core into:
    - Front-end (FE)
    - Back-end (PE)
  - Treat sub-cores as building blocks
  - Recombine sub-cores to make arbitrary cores
  - Support multiple core configurations simultaneously

- **Benefits:**
  - Application-specific performance
  - Granularity of configuration:
    - Finer than multicore – more flexibility
    - Coarser than FPGA – less overhead
  - One fabric to exploit multiple levels of parallelism at near-optimal efficiency
  - Flexibility to utilize all resources
Approach

Configurable Interconnect

• **Concept:**
  • Organize groups of FE and PEs into subnets connected via global links
  • Support generic packet-switched and configurable circuit-switched network

• **Benefits:**
  • Exploit fast communication in FE-PE configurations that take advantage of local subnets
  • Flexibility to choose network type more suitable for implemented FE-PE configuration
Approach

SISD and Related Execution Modes

Available

- Single SISD core (single IO core)
- Multiple SISD cores (MIMD, multicore)

Available

In Development

- Improved single-thread performance:
  - Federated SISD cores (single OO core)

In Development

- Improved single-core reliability:
  - Lock-step SISD core (redundant cores)
Approach

SIMD and Related Execution Modes

In Development

Single SIMD core

Multiple SIMD cores

Improved SIMD reliability:

Lock-step SIMD core (redundant cores)
Outline

• Motivation

• Approach

• Design Framework

• Implementation

• Future Goals
Design Framework

Leon3 Platform and Partition
FPGA Verification and Prototyping:

- Target device: Xilinx Virtex 2 Pro board
- Simulated Leon3 with various test programs to verify functionality
- Preparing to simulate sub-core/SIMD/FED designs and create testbenches

Test Chip Design and Simulation:

- Target technologies: MITLL 0.13 um (ultra-low power cell library) and ST 0.13 um
- Cadence Ncsim behavioral simulation of Leon3, sub-core and SIMD/FED designs
- Cadence Ncsim post-synthesis simulation of Leon3 and sub-core designs
Lessons Learned

• **Splitting of Leon3 Pipeline, Lack of Modularity**
  
  - Leon3 pipeline stages are tightly coupled, splitting of the core resulted in many individual signals forwarded between FE and PE partitions
  
  - Some of the shared components in Leon3 such as bus interface and MMU were difficult to split or duplicate and then integrate back into the system

• **Difficulties with MITLL 0.13um Synthesis, Placing and Routing**
  
  - MITLL technology is limited to 3 metal layers which resulted in large areas reserved for routing and low core density
  
  - Original vias in MITLL technology were too large for some of the cells, they had to be resized to avoid spacing violations
  
  - Encountered timing violations due to spacing, had to change wire model, tune the fanout and clock synthesis parameters
Outline

• Motivation

• Approach

• Design Framework

• Implementation

• Future Goals
Implementation

Test Chip

On-chip Components:
- 2 FE + 2 PE with I/D caches
- System bus
- Self monitoring capability

Off-chip Components:
- Resource Manager
- Debug capability
- DRAM
SIMD

- SIMD execution is enabled by connecting multiple PEs to a single FE with an instruction sequencer.

- Single FE connects to multiple PEs and issues identical instructions to PEs, some PEs are masked off if their flow diverges upon branch.

- Main structures include: feedback connections and SIMD unit that informs FE about PEs’ status, stalls and exceptions.

- Task manager dynamically adjusts network and # of PEs to provide SIMD width requested by application at any time.
Out-of-order execution is enabled by federating two in-order cores with minimal hardware overhead.

Each FE fetches instructions and issues them to its PE while verifying dependencies with the FED unit.

Main structures include: subscription-based issue queue in place of broadcasting logic, memory alias table in place of alias checking logic.

Task manager dynamically enables federation mode when requested by the application at any time.
Implementation

Lessons Learned

• Problems with SIMD/FED Development
  
  • Signals and variables between pipeline stages in Leon3 RTL code are difficult to interpret, naming is unintuitive and purpose is obscured
  
  • Multi-port memory structures are required for both register file and tables in Federation, RTL interfaces or library cells are not available and need to be developed
  
  • Register window feature in Leon3 makes it difficult to implement OO execution, instructions that change register window pointer cannot be executed OO

• Limitations of Leon3, Change of Platforms Considered
  
  • Leon3 is robust and mature, however, a research core with more modular pipeline stages, better documentation and more open-source components would be preferred
  
  • Considering to change the development platform, but other open-source cores do not appear to be mature or sophisticated enough
Outline

• Motivation

• Approach

• Design Framework

• Implementation

• Future Goals
Future Goals

- Develop Resource Manager and reconfigurable network to allow automatic switching between modes
- Implement remaining core configurations
- Support many architectural configurations simultaneously
- Perform comprehensive design-space study, including cache/memory/NoC organization
- Decide about API, start with OpenCL/CUDA-like and move toward more general OpenMP-style
- Demonstrate functionality with test chips
Technology Transfer

• **Industry Interactions:**
  
  • Liaisons: Srilatha Manne (AMD), Prabhakar Kudva (IBM), Jamison Collings and Perry Wang (Intel)
  
  • Regular conferences liaisons and their input on the project
  
  • Would like to identify new liaisons

• **Internships:**
  
  • Lukasz Szafaryn, Lawrence Livermore National Laboratory, Jun-Sep 2010
  
  • Marisabel Guevara, AMD Sunnyvale, May-Aug 2010

• **Presentations / Publications:**
  
  • Project status updates published on SRC website
  
  • Publications on the project in preparation
Questions/Discussion