Configuration Prefetching and Reuse for Preemptive Hardware Multitasking on Partially Reconfigurable FPGAs

Aurelio Morales-Villanueva,
Rohit Kumar and Ann Gordon-Ross

UF UNIVERSITY of FLORIDA

CHREC NSF Center for High-Performance Reconfigurable Computing

Introduction

Goals

- Develop services that leverage partially reconfigurable (PR) FPGAs for high performance embedded reconfigurable computing (RC) systems

Motivations

- PR FPGAs enable preemption/resumption of hardware (HW) tasks in PR regions (PRRs) without losing tasks' execution state
- PR FPGAs enable HW multitasking over shared resources
- Time to reconfigure a PRR delays HW task execution
- Reconfiguration time can be reduced/hidden using:
  - Configuration prefetching
  - Configuration reuse
- Prior works only provide partial solutions, and no physical implementation

Approach

- Leverage capabilities of PR FPGAs
- Implement services with portability across different FPGA architectures

Accomplishments

- Novel implementation of configuration prefetching and reuse for preemptive HW multitasking on a Virtex-5 FPGA

Configuration Prefetching and Reuse on PR FPGAs

Overview

- Configuration prefetching and configuration reuse reduce the time to reconfigure a PRR on any PR system

Approach

- Leverage ICAP and bitstream manipulations
- Use internal GSR signal and protection/unprotection mechanism for static region and PRRs
  - Protection: avoids GSR reinitialization of flip-flops and BRAMs
  - Unprotection: allows GSR reinitialization of flip-flops and BRAMs

Benefits

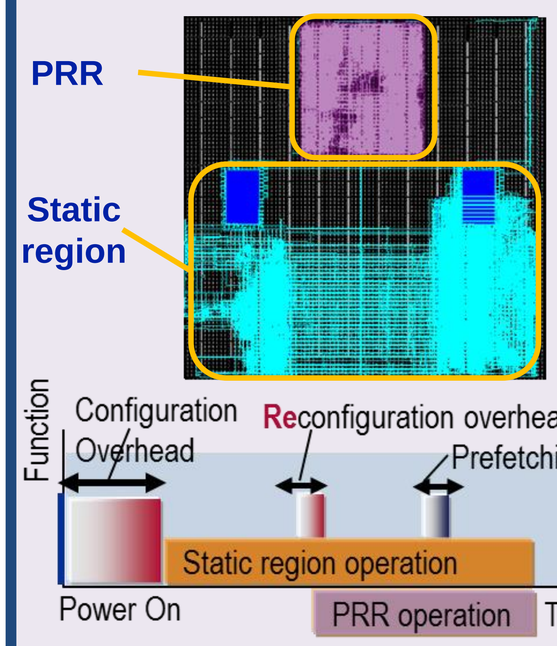
- Prefetching: PRR reconfiguration overlaps HW task execution over the same PRR w/o affecting execution of current HW task
- Reuse: No PRR reconfiguration needed, if preempted HW task needs to resume last execution
- No tool flow changes needed
- Fundamentals can be extended to newer device families (Series-7, Zynq 7000, UltraScale)

Experiments

- Testbed: Virtex5 LX110T, 100 MHz, one PRR, OpenSPARC board, embedded Linux OS
- PRRs: implement two HW tasks, with CLBs and BRAMs
- Static region: MicroBlaze, Ethernet interface, FSLs, ICAP, GPIOs, DDR2 SDRAM, Compact Flash interface

FPGA – Field Programmable Gate Array
PRR – Partially Reconfigurable Region
CLB – Configurable Logic Block
ICAP – Internal Configuration Access Port
GSR – Global Set and Reset
FSL – Fast Simplex Link
GPIO – General Purpose Input/Output

Configuration Prefetching and Configuration Reuse Details

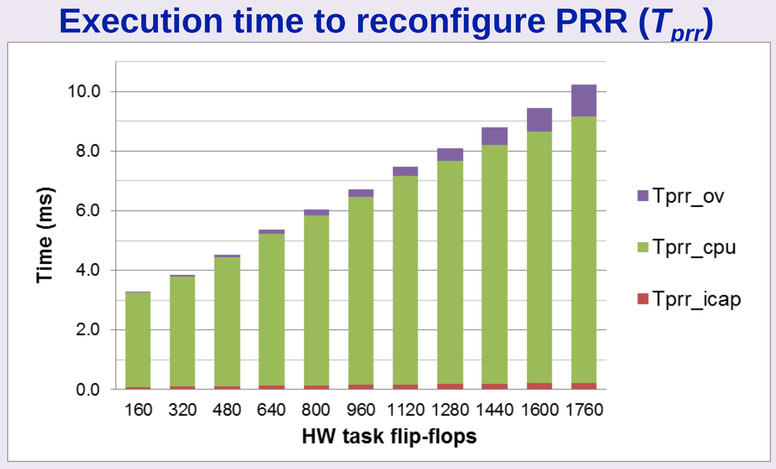


- Single and multiple PRR systems
  - On task preemption, save the task's state (i.e., context save (CS))
  - On task resumption, restore the task's state (i.e., context restore (CR))
- Protection (single PRR system)
  - Entire FPGA after power on
- Unprotection (single PRR system)
  - Only on PRR after power on
- Protection (multiple PRR system)
  - Entire FPGA after power on
  - Only on PRR after CS and CR
- Unprotection (multiple PRR system)
  - Only on PRR before CS and CR

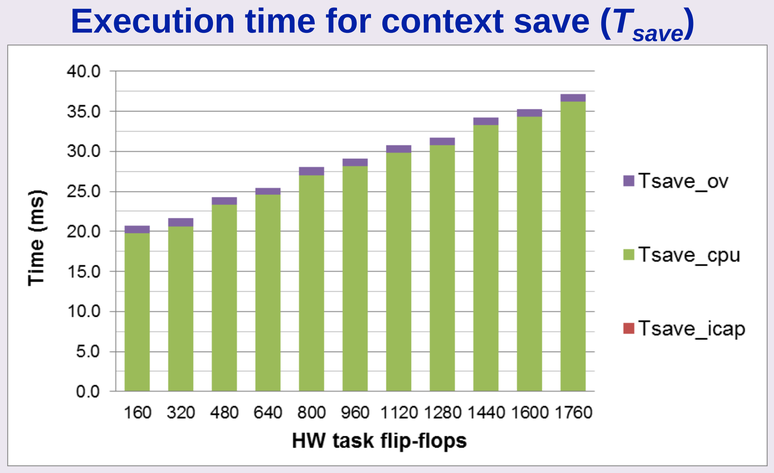
- Use of GSR signal for configuration prefetching (single and multiple PRR system)
  - After CS of the preempted task. New task was already prefetched before CS of preempted task
- Use of GSR signal for configuration reuse (single and multiple PRR system)
  - Allows re-execution of task from last task resumption state, without reconfiguring the PRR

Experimental Results

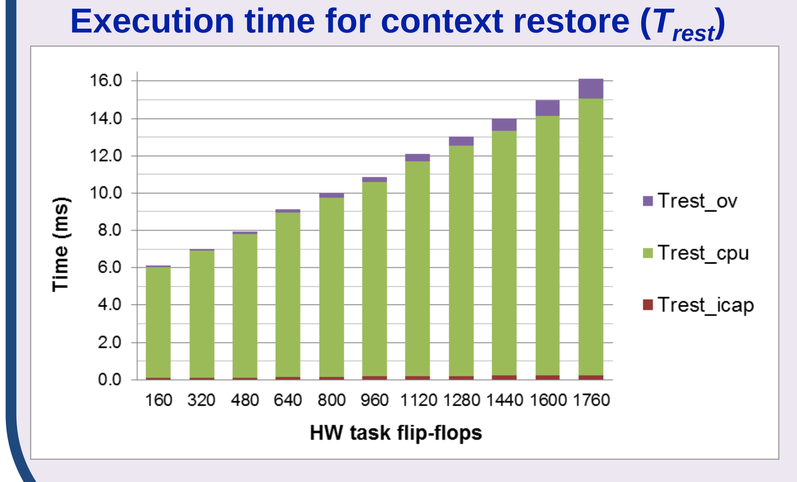
Execution time to reconfigure PRR (T_{prr})



Execution time for context save (T_{save})



Execution time for context restore (T_{rest})



- PR system with a MicroBlaze softcore processor
  - Executes Linux OS
  - Executes a software application that orchestrates CS, CR, configuration prefetching and reuse
- T_{prr} : linear growth rate
  - Depends on PRR size
  - T_{prr_icap}, T_{prr_cpu}, and T_{prr_ov} are the ICAP, CPU, and overhead execution times
- T_{save} : linear growth rate
  - Depends on the number of flip-flops and BRAMs used in the HW task
  - Includes unprotection and protection of PRR
  - T_{save_icap}, T_{save_cpu}, and T_{save_ov}, are the ICAP, CPU, and overhead execution times
- T_{rest} : linear growth rate
  - Depends on PRR size
  - Includes unprotection and protection of the PRR
  - T_{rest_icap}, T_{rest_cpu}, and T_{rest_ov}, are the ICAP, CPU, and overhead execution times