PR-centric Embedded Systems for RC Design Productivity

F4

CHREC NSF Center for High-Performance Reconfigurable Computing

Aurelio Morales and Elizabeth Graham
Dr. Ann Gordon-Ross

UF UNIVERSITY of FLORIDA

F4

Introduction

Goals

- Develop services that leverage partial reconfiguration (PR) for high performance embedded reconfigurable computing (RC) systems
- Tool transparency to give user control over application development

Motivations

- PR enables area and power savings
- Distributed computing provides increased system computation capability
- Cost-effective, scalable, and flexible base architecture (VAPRES) facilitates distributed computing system development and management

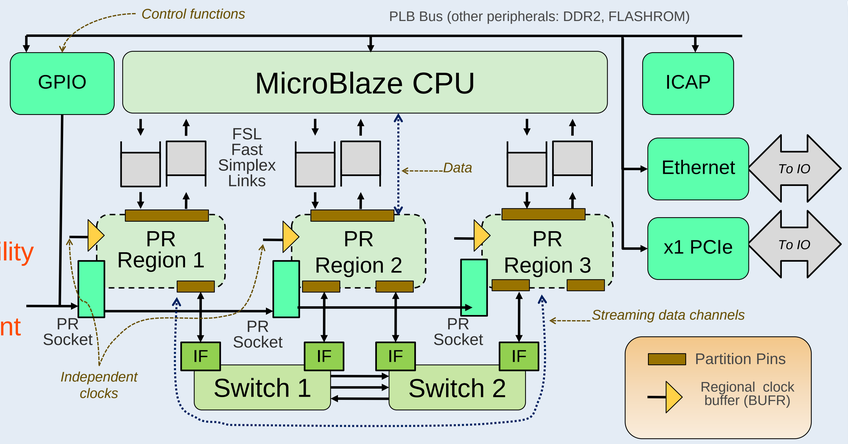
Approach

- Leverage network of PR-capable FPGAs (e.g., VAPRES FPGAs) for increased resource pool
- Distributed system services efficiently manage networked resources
  - E.g., context save and restore (CSR), bitstream relocation (BR), and distributed dynamic resource manager (DDRM)

VAPRES

Accomplished Tasks

- Implemented CSR to save and restore PRM's execution context
- Implemented BR to allow PRRs to execute any PRM
- Implemented DDRM simulation model to balance processing load on network of VAPRES nodes



VAPRES Overview

- Multi-purpose, flexible, and scalable PR base architecture
- Simplifies use of and communication between PRRs and PRMs
- DDRM efficiently manages VAPRES resources and task execution
  - DDRM leverages CSR to prevent loss of processed information
  - DDRM leverages BR to maximize PRR utilization and save memory

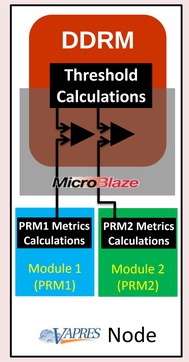
PRM – Partially Reconfigurable Module
PRR – Partially Reconfigurable Region

CLB – Configurable Logic Block
ICAP – Internal Configuration Access Port
GPIO – General Purpose Input/Output

Distributed Dynamic Resource Manager (DDRM)

DDRM enables automatic module relocation between nodes (node switching) to balance processing load

- Node switching is determined by PRM's metrics
  - Metrics describe PRM's current state
  - Metrics are user-selected and application-specific
- Thresholds define specific metric values that trigger node switching
  - Metrics outside lower- or upper-bound thresholds (i.e., outside threshold range) trigger node switch
    - Trigger node switch when metric enters threshold buffer to account for node switching latency
    - Metric's low and high critical values define required switching times
  - User provides threshold calculation formula
    - Hardware for critical thresholds – fast, limited space
    - Software (MicroBlaze) for additional thresholds – slow



- After trigger, determine best destination node for PRM
  - PRM's current node polls open nodes for metrics and thresholds
  - After calculation, nodes return metrics and thresholds to current node
  - Current node migrates PRM to best-fit neighboring node
    - Best-fit: Metrics are within threshold ranges and farthest from bounds
- Node switching leverages CSR, BR, and Ethernet

On-chip Context Save and Restore (CSR)

Overview

- CSR provides a mechanism to stop and capture state of running PRMs and resume execution in the future

Approach

- Leverage PR capabilities of FPGAs to implement CSR
- Context save (CS) and context restore (CR) via ICAP
  - CS: Checkpoint PRM's current execution state (i.e., context)
  - CR: Restore checkpointed PRM execution state
- Implement as a C program on MicroBlaze

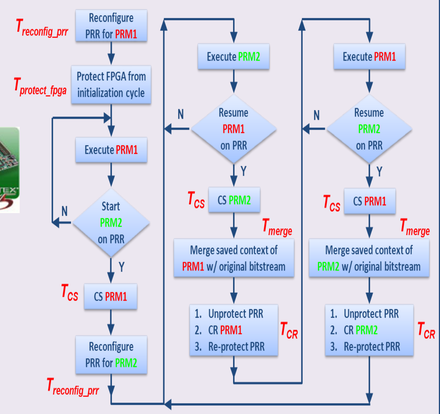
Benefits

- On-chip, no external device required
- Running on MicroBlaze, no custom hardware needed

Experiments

- VAPRES static region: MicroBlaze, UART, Ethernet interface, FSLs, ICAP, GPIOs, DDR2 SDRAM, Compact Flash interface
- PRRs (CLBs only): Each PRR implements two or more PRMs
- Testbed: Virtex5 LX110T, 100 MHz, 1 PRR, ML509, Linux OS

Times	Function definition	Exec. Time	rows	cols	frames/col	type	PRR nets
$T_{protect\_fpga}$	f_1(rows, cols)	67.0 ms	8	65	vary	all	---
$T_{reconfig\_prr}$	f_2(rows, cols, frames/col)	1.8 ms	1	2	36	CLB	---
T_{CS}	f_3(rows, cols, frames/col)	8.2 ms	1	2	36	CLB	---
T_{merge}	f_4(PRR nets)	19.5 ms	---	---	---	CLB	106
T_{CR}	f_5(rows, cols, frames/col)	27.4 ms	1	2	36	CLB	---



Context Save and Restore (1 PRR, 2 PRMs)

Detailed CSR Flow

- Example: 1 PRR, 2 PRMs
  - PRR is CLBs only
  - PRR is 1 row, many columns
- Protection
  - Whole FPGA before first CR
  - Only PRR, after CR
- Unprotection
  - Only PRR, before CR

On-chip Bitstream Relocation (BR)

Overview

- BR provides a mechanism to relocate previously saved context of a PRM to any PRR

Approach

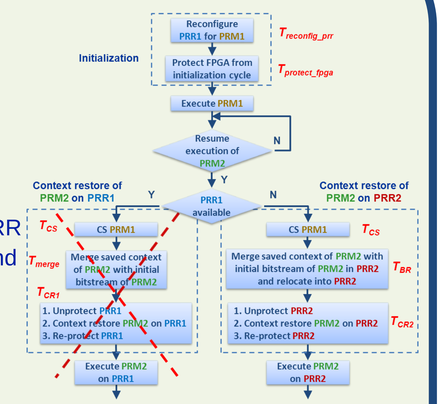
- Leverage PR capabilities of FPGAs to implement BR
- Leverage CSR implementation
  - Relocates saved context using CS on one PRR and CR to a different PRR
  - Generate relocated partial bitstream from initial-state partial bitstream and saved context partial bitstream
- On-the-fly relocated partial bitstream generation
  - PRRs can have different sizes and locations
  - PRRs must have similar resources (currently CLBs only)
  - PRMs must have same height (1 row) but can have different widths

Benefits

- Reduced storage requirements
- Reduced bitstream generation time (milliseconds vs. hours)
- Runs as a C program on MicroBlaze, no custom hardware needed

Testbed: Virtex5 LX110T, 100 MHz, 2 PRRs, ML509

Times	Function definition	Exec. Time	src PRR	dst PRR	frames/col	type	PRR nets
T_{CS}	f_3(rows, cols, frames/col)	8.2 ms	r=1,c=2	---	36	CLB	---
		15.7 ms	r=1,c=4	---	36	CLB	---
T_{BR}	f_6(PRR nets, bit positions)	31.9 ms	r=1,c=2	r=1,c=4	---	CLB	106
		30.3 ms	r=1,c=4	r=1,c=2	---	CLB	106
T_{CR}	f_5(rows, cols, frames/col)	26.9 ms	---	r=1,c=2	36	CLB	---
		36.1 ms	---	r=1,c=4	36	CLB	---



Bitstream Relocation (2 PRRs, PRM2 in both PRRs)

Detailed BR Flow

- Example: 2 PRR, 2 PRMs/PRR
  - PRM2 for both PRRs
  - PRRs are CLBs
- Relocation
  - Only during context restore
  - Primary PRR is not available for PRM2
- Protection/Unprotection
  - Same as CSR