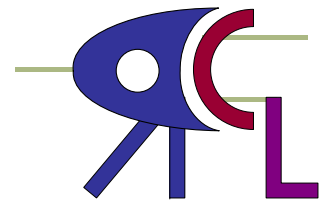


Simplifying Application-Specific System Design for SoCs and Multi-FPGA Platforms

Lesley Shannon

School of Engineering Science
Faculty of Applied Sciences
Simon Fraser University, Burnaby, B.C.

CMOS ET, September 2009





Motivation

Designers spend ~30% of design time on integration and 30-50% on testing and verification



Motivation

Designers spend ~30% of design time on integration and 30-50% on testing and verification

Objective

Create a system Model to reduce design time and facilitate design reuse



Outline

- System Model Requirements
- Design Reuse Challenges
- Introducing the SIMPPL Model
- Benefits and Overhead
- Conclusions and Future Work



System Model Requirements

1. Enable designers to create application-specific architectures on single/multi-FPGA platforms.



System Model Requirements

1. Enable designers to create application-specific architectures on single/multi-FPGA platforms.
2. Allow the combination of processor(s) and dedicated Processing Elements (PEs).



System Model Requirements

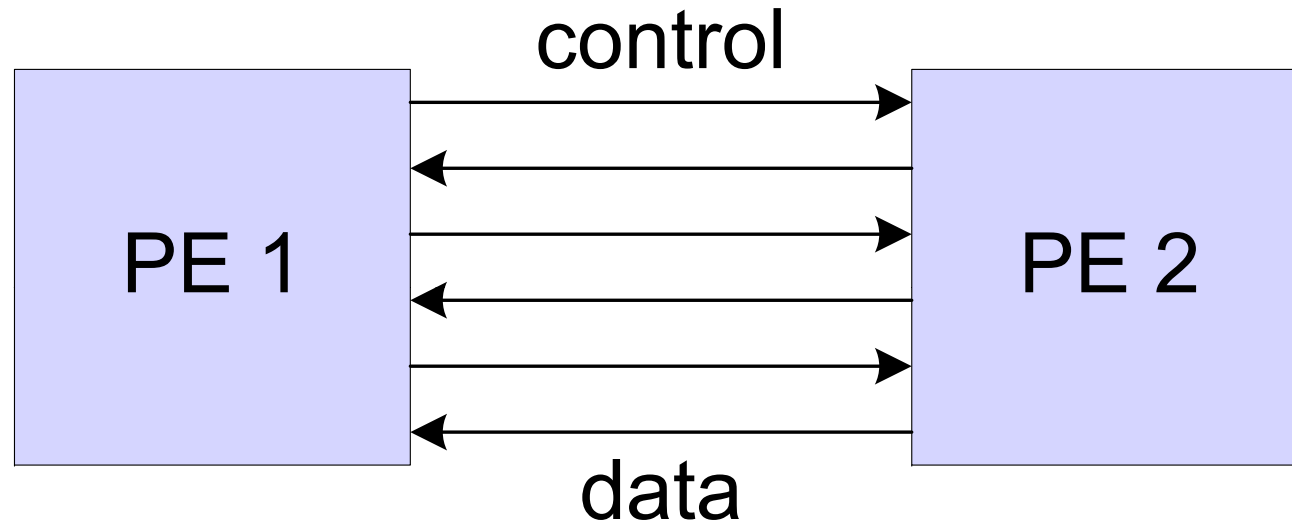
1. Enable designers to create application-specific architectures on single/multi-FPGA platforms.
2. Allow the combination of processor(s) and dedicated Processing Elements (PEs).
3. Facilitate the communication between processors and hardware PEs.



System Model Requirements

1. Enable designers to create application-specific architectures on single/multi-FPGA platforms.
2. Allow the combination of processor(s) and dedicated Processing Elements (PEs).
3. Facilitate the communication between processors and hardware PEs.
4. Simplify the reuse of hardware PEs among systems.

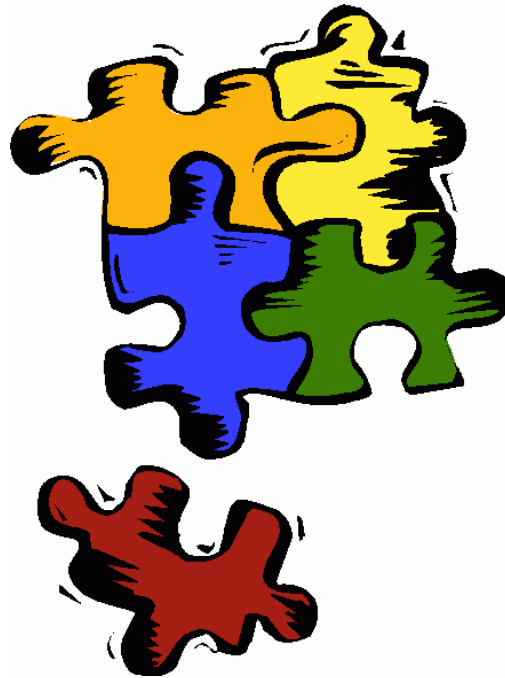
Dedicated Logic Design Features



- Dedicated logic runs ***FAST***
- Dedicated logic is difficult to design
- Reuse previously designed PEs to reduce design time

Challenges for PE Reuse

1. Differences in the physical interface



Solution: Standardize the physical interconnections between modules

Challenges for PE Reuse

2. Differences in the communication protocols



Solution: Standardize the communication protocols for passing data between modules



Challenges for PE Reuse

Previous work includes:

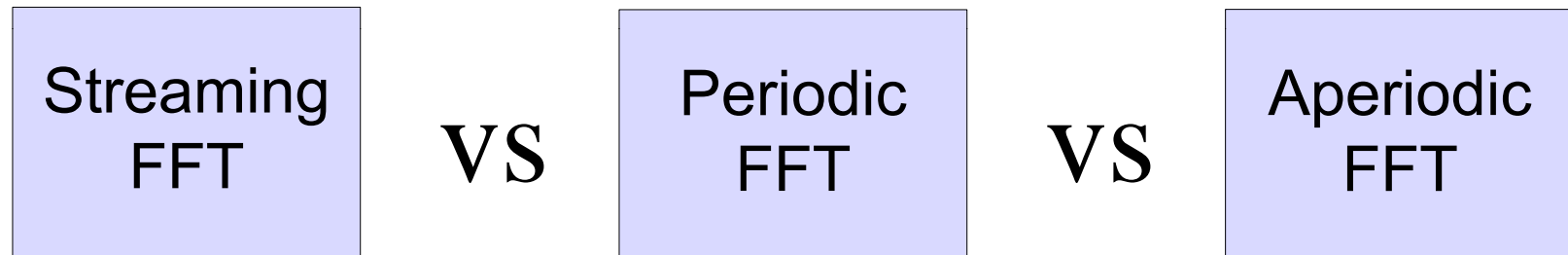
- 1) Open Core Protocol (OCP)
- 2) Wishbone
- 3) Amba

Interested Groups include:

- 1) VSIA
- 2) Spirit Consortium

Challenges for PE Reuse

3. Differences in the functional requirements



Solution: Separate the functionality from the communication protocols



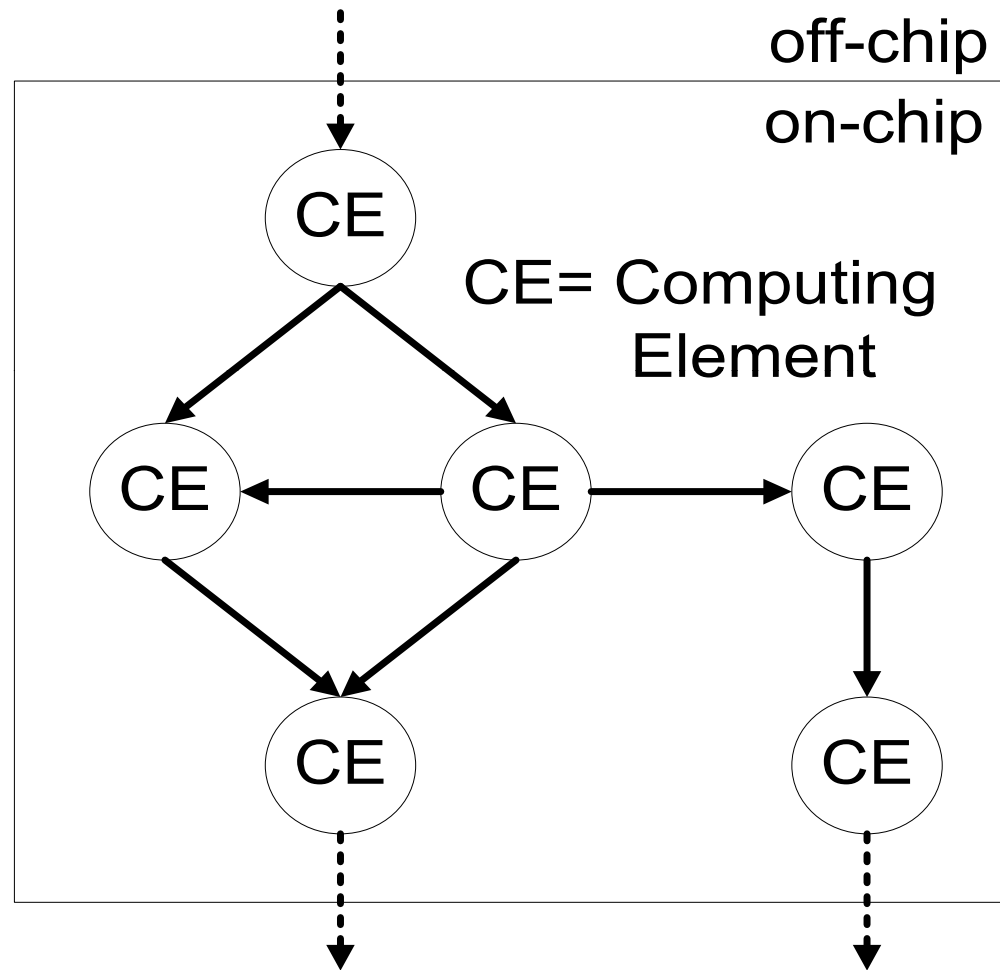
The SIMPPL Model



SIMPPL Defined

- SIMPPL stands for “System Integrating Modules with Predefined Physical Links”
- It is a system architectural framework
- Used to reduce integration time and facilitate Intellectual Property (IP) reuse

The SIMPPL Model



Systems Integrating Modules with Predefined Physical Links

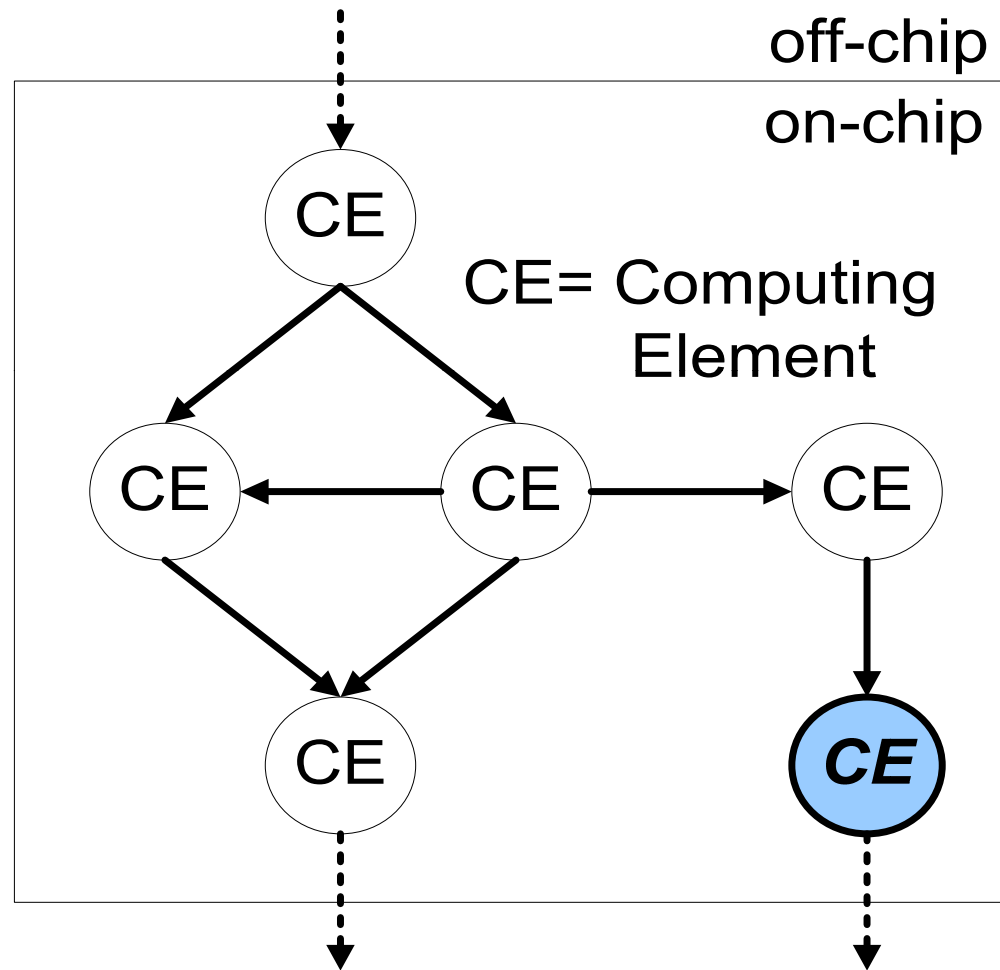


The SIMPPL Model

Previous work includes:

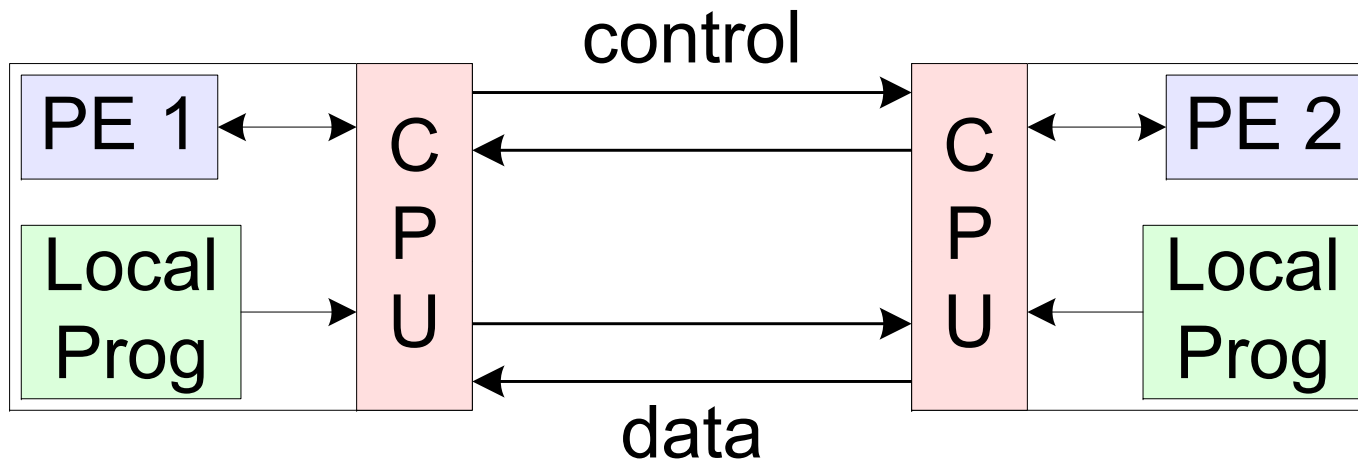
- 1) Kahn Process Networks [Kahn '74]
- 2) Data Process Networks [Lee et al '95]
- 3) YAPI [Kock et al '00]

The SIMPPL Model



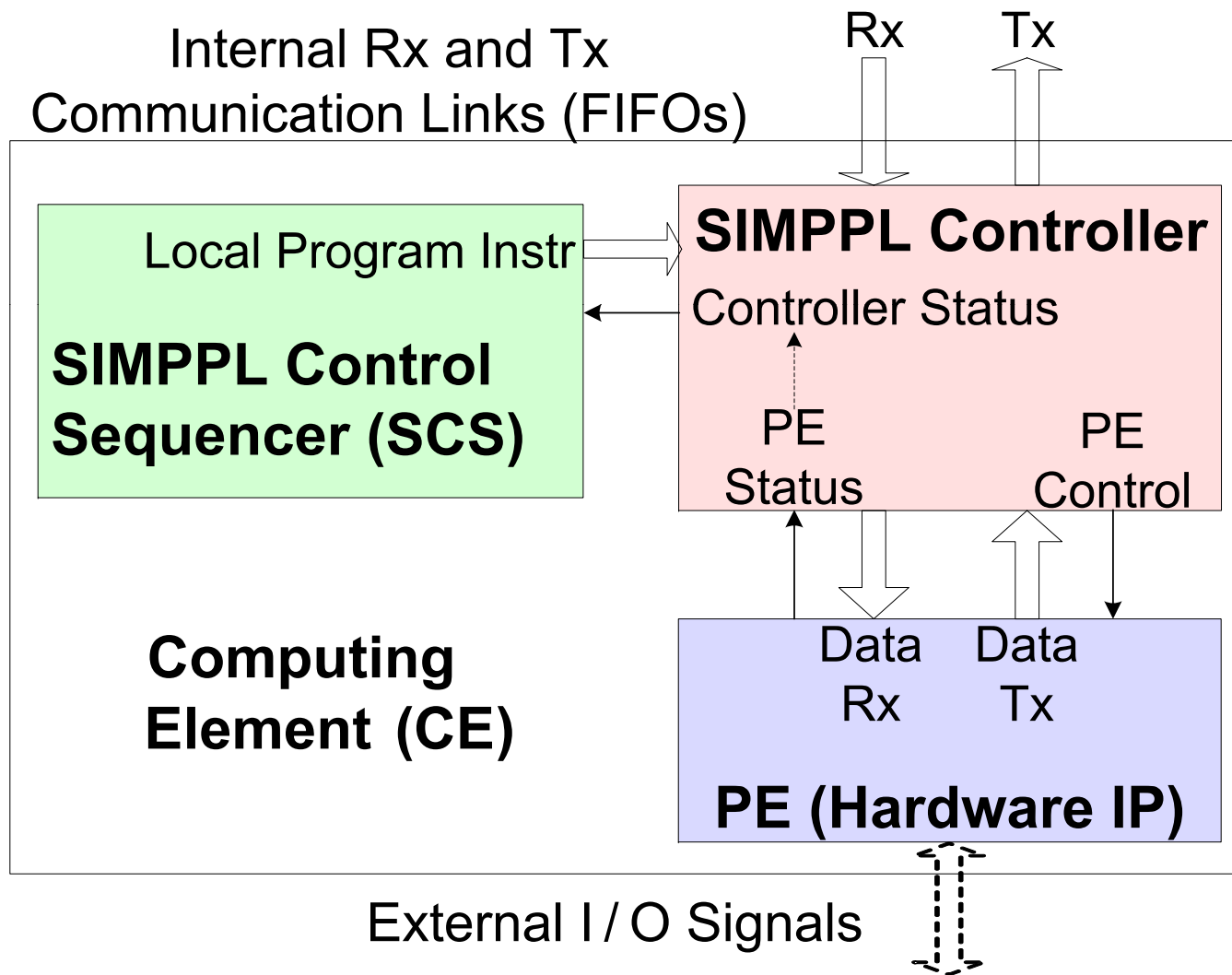
Systems Integrating Modules with Predefined Physical Links

How a Hardware CE Works

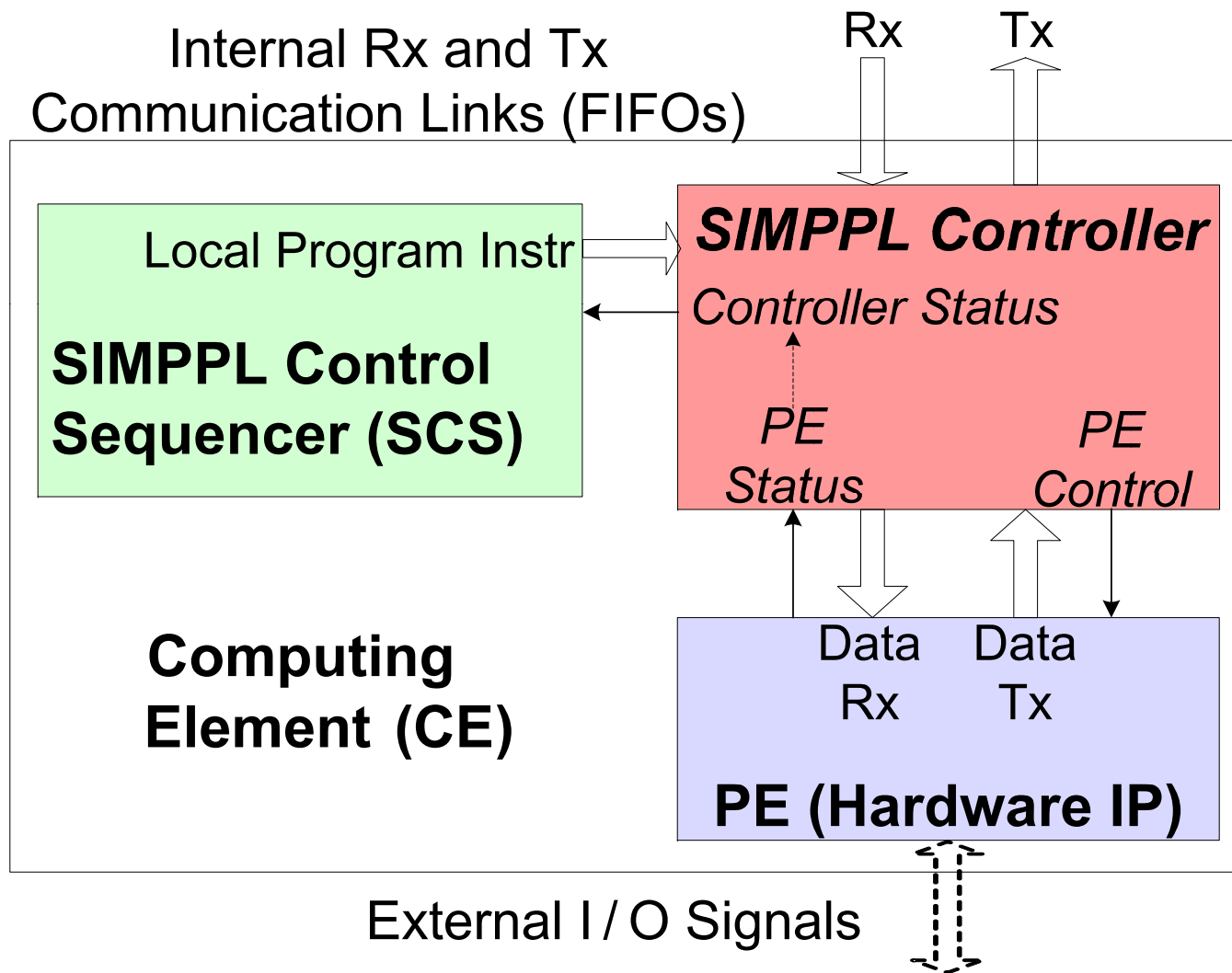


- Easy to program
- But interface is ***SLOW!***

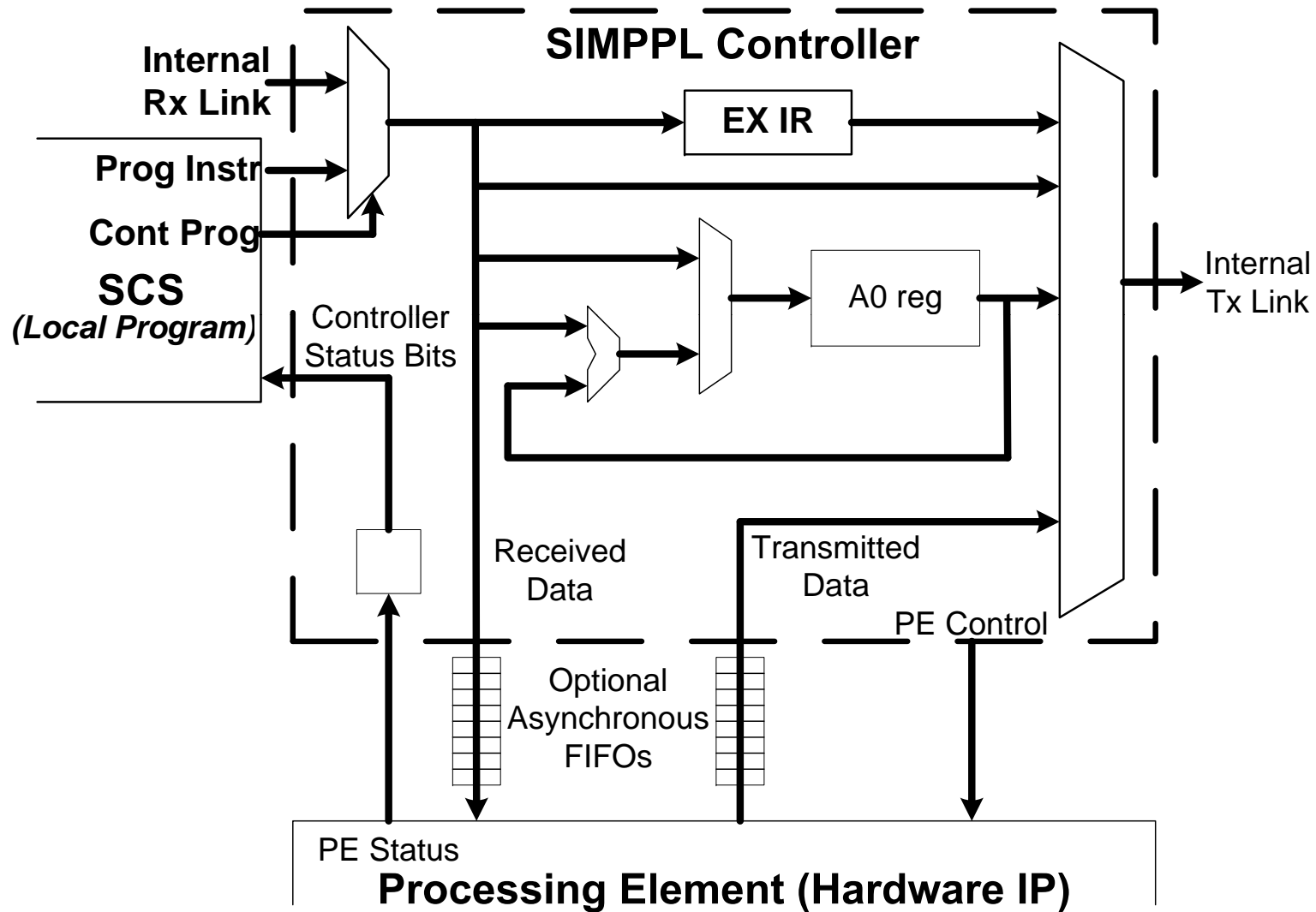
Hardware CE Abstraction



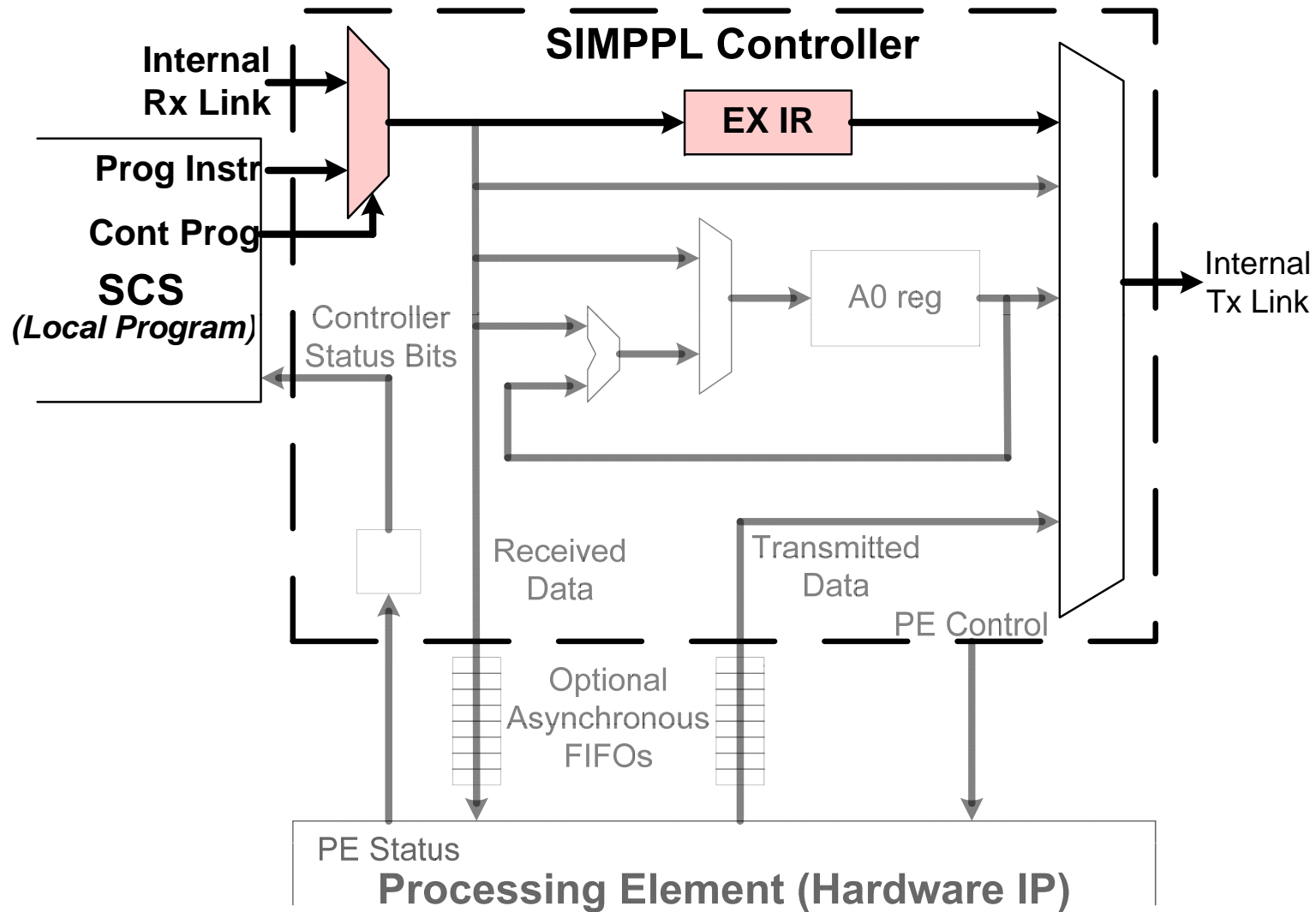
Hardware CE Abstraction



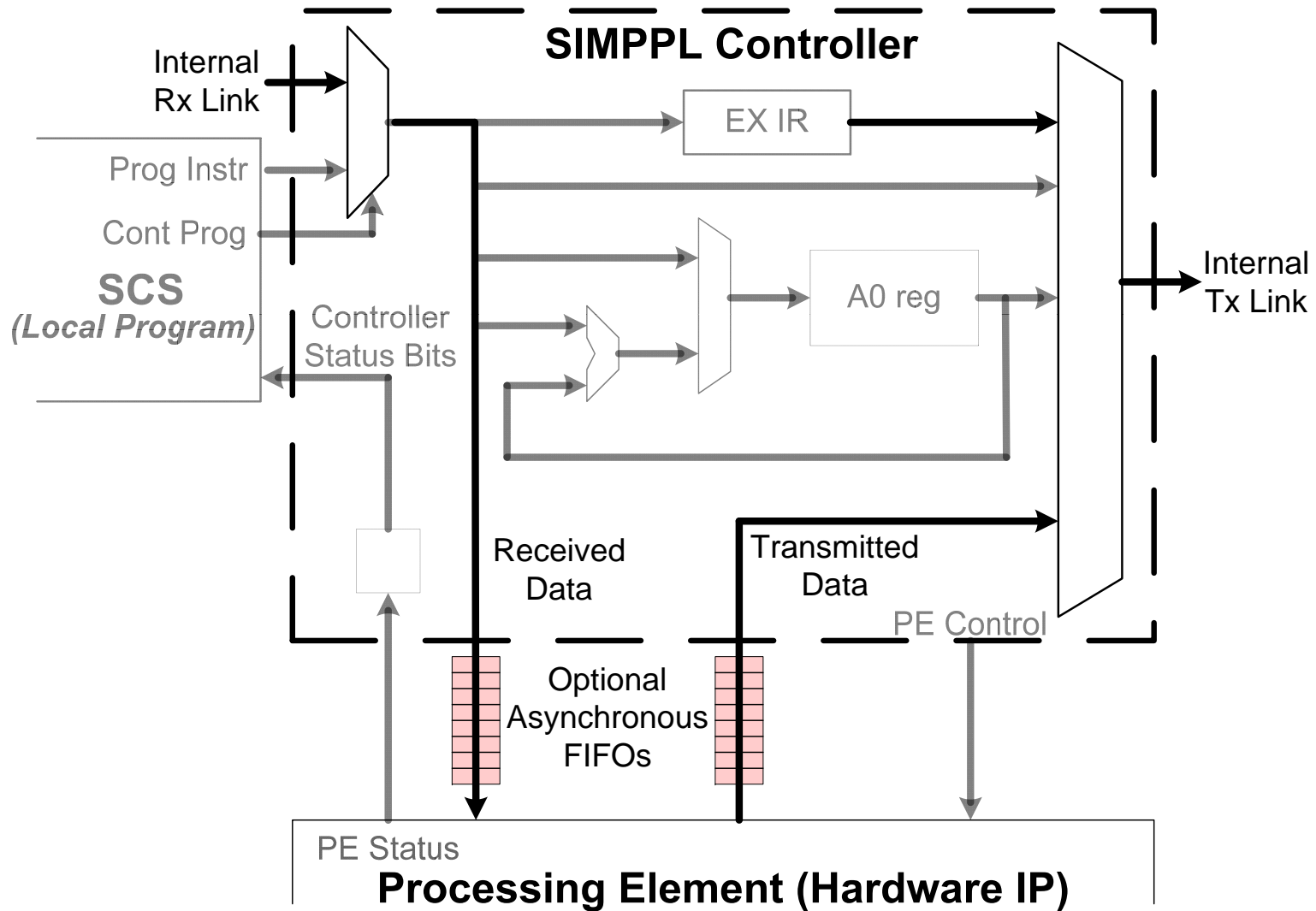
SIMPPL Controller



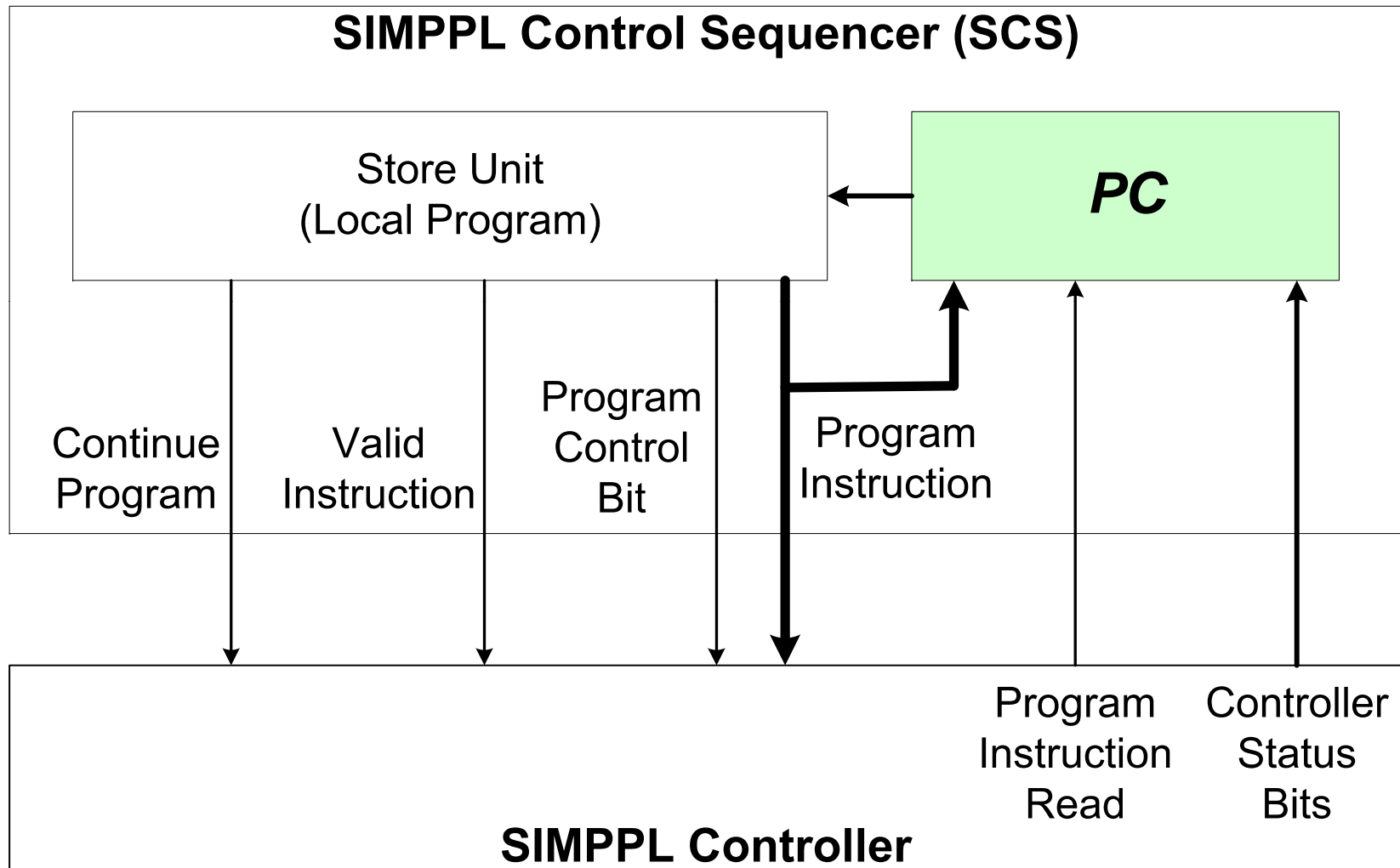
SIMPPL Controller



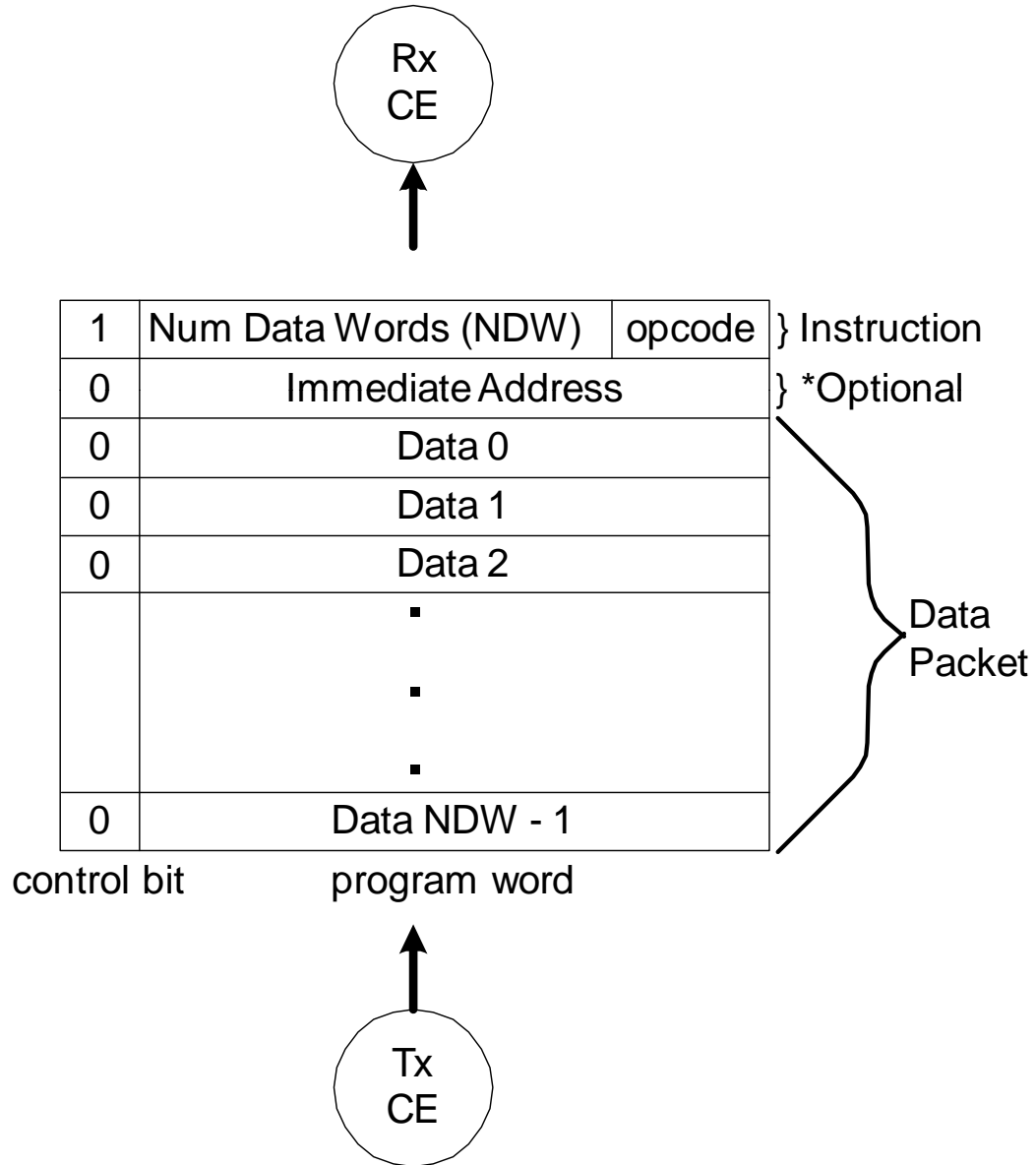
SIMPPL Controller



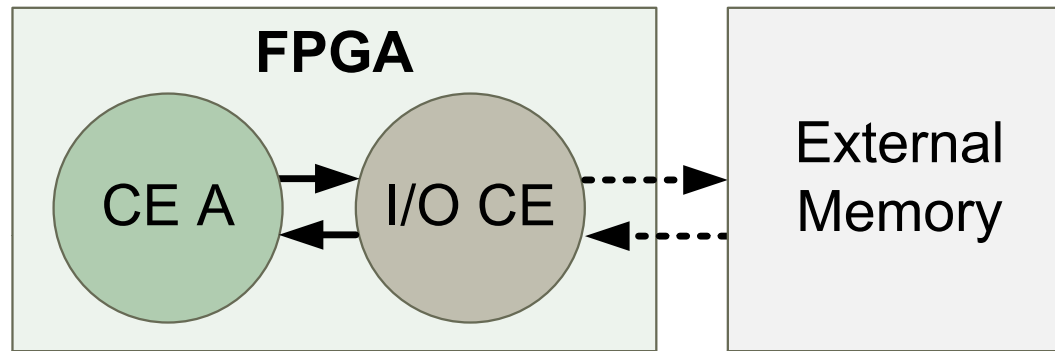
SIMPPL Control Sequencer (SCS)



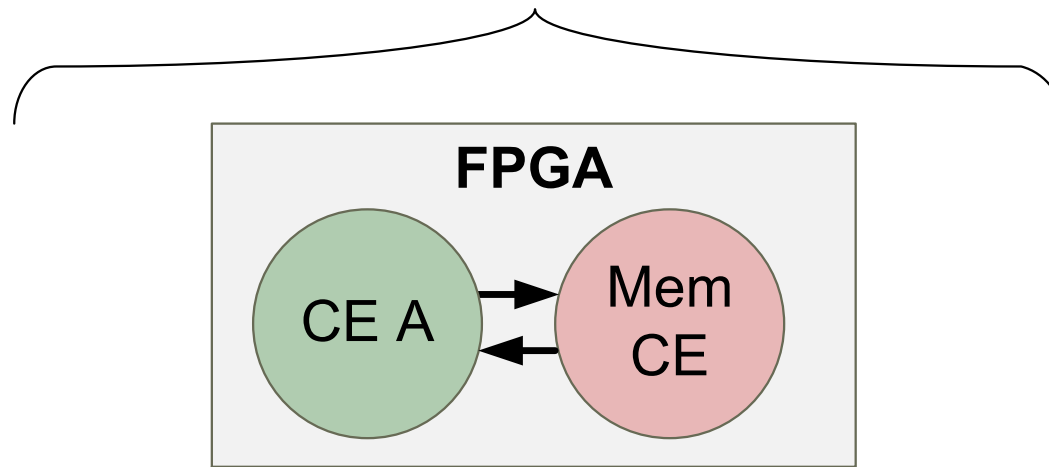
Instruction Packet Format



SIMPPL I/O Model



Is modelled as ...

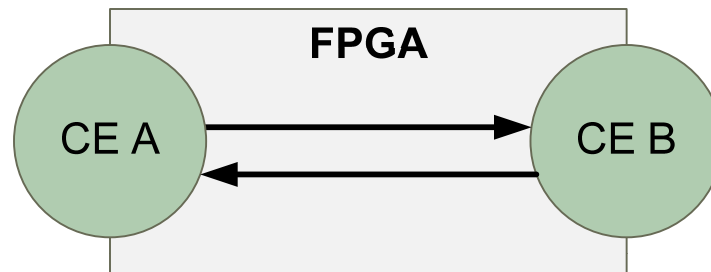




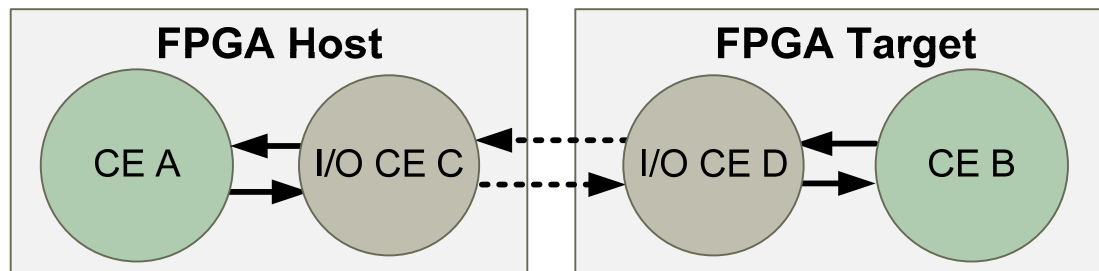
Multi-FPGA Systems

- Large systems may not fit on a single FPGA
- May require multiple FPGAs for implementation
- SIMPPL CEs on different FPGAs may need to communicate with each other
- This communication differs from SIMPPL I/O in an SoC model

Using Standard SIMPPL I/O



Desired system doesn't fit on a single FPGA



Try using I/O CEs for Inter-FGPA Communication

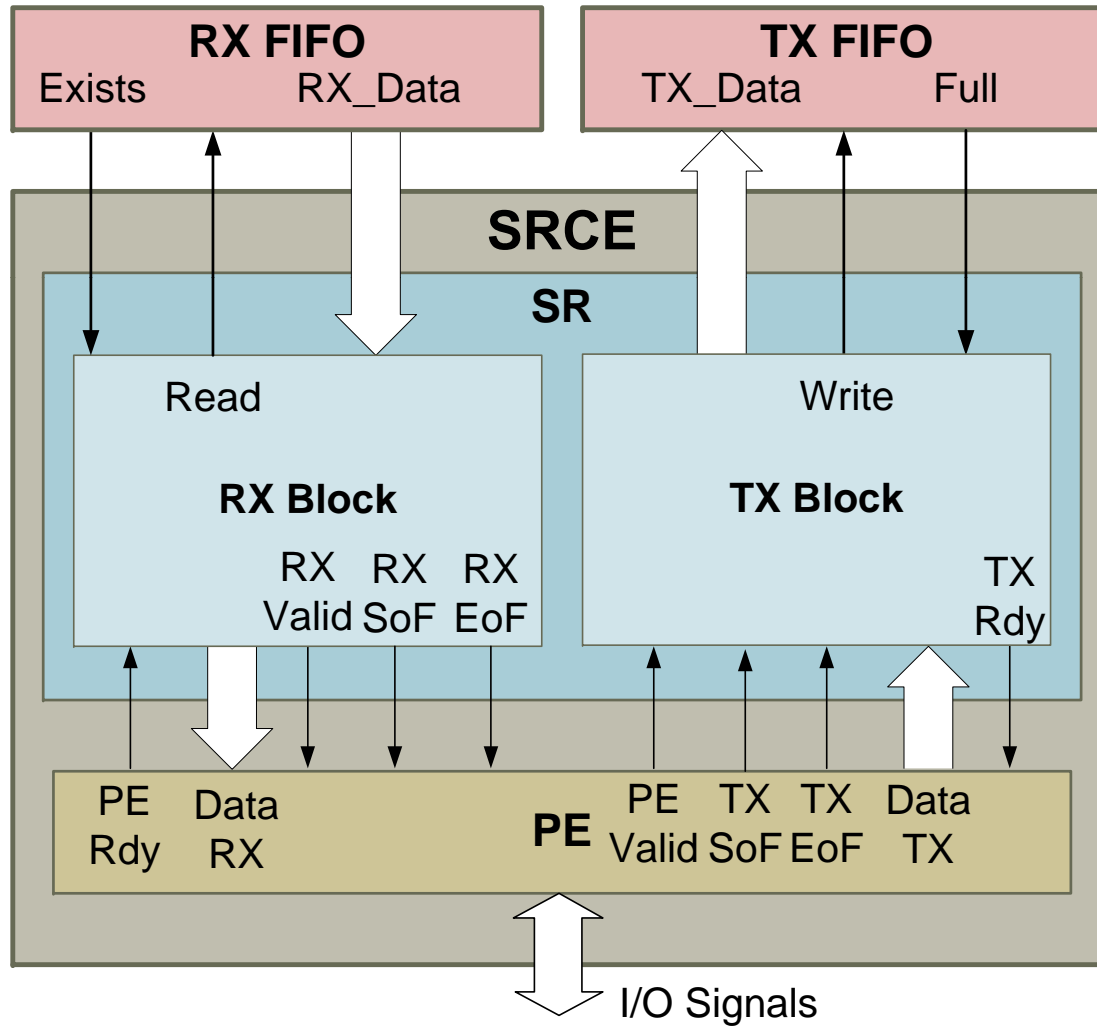
Problem: I/O SIMPPL Controllers execute instructions



The SIMPPL Repeater (SR)

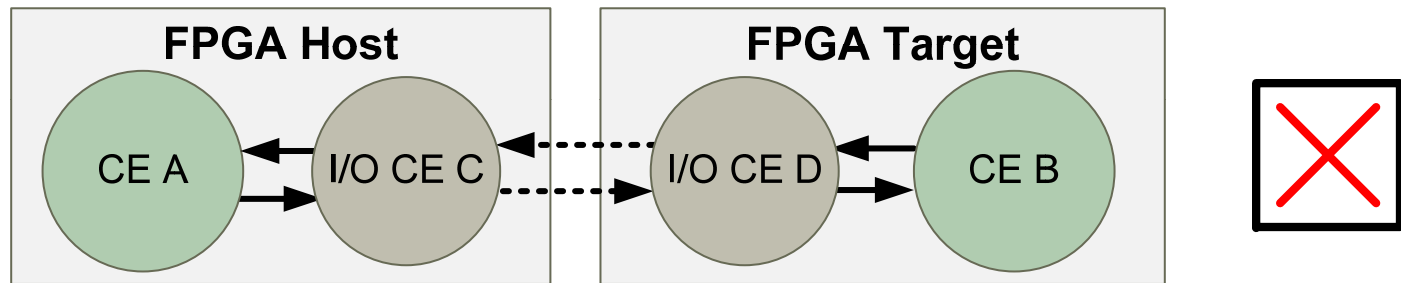
- Replaces the SIMPPL Controller in an I/O CE
- Forwards SIMPPL instructions (Does not execute)
- Not programmable, so no SCS is needed
- Enables communication between two CEs on different FPGAs is abstracted

SIMPPL Repeater Computing Element (SRCE)

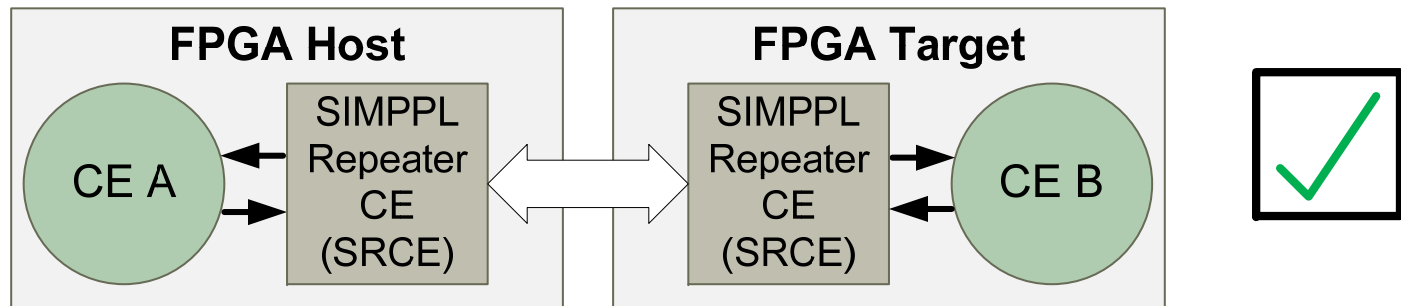


SIMPPL Inter-FPGA Communication

- SRCEs instead of I/O CEs for inter-FPGA I/O



Try using I/O CEs for Inter-FPGA Communication



SRCEs for Inter-FPGA Communication



Benefits and Overhead of SIMPPL

■ Benefits

- Abstracts PE operation from system communication and control
- Maintains throughput
- Reduces system integration time to <5%
- Programmable PE control to increase reusability
- Facilitates on-chip profiling, testing and verification when using an FPGA
 - Programmable fabric enables reconfiguration



Benefits and Overhead of SIMPPL

- Overhead

- “Fetch, Decode, Execute” increases latency
- Programmability increases area



Conclusions and Future Work

- Technology sizes are continually decreasing, enabling designers to create increasingly complex SoCs
- Abstractions that reduce design time are necessary to facilitate application-specific architectures
- SIMPPL is an example of an abstraction that allows designers to focus on processing instead of system-level control and communication



Conclusions and Future Work

- Controller's instruction set can be optimized to PE usage
- Working to develop an HLL programming environment for individual CE (SCS) programs.
- Eventually develop an HLL programming environment for describing system-level control that can be used to generate individual SCSs for system CEs



Acknowledgements

- NSERC
- CMC Microsystems
- Xilinx, Inc.
- Altera



Thank You for Listening

- Questions?