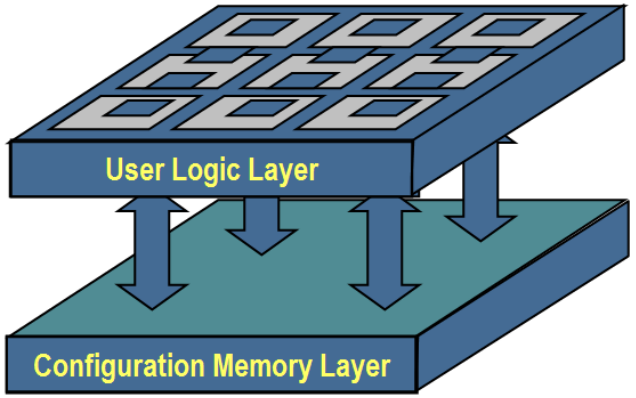
A System Architecture for Aerospace Applications Employing Partial Reconfiguration and High-Level Simulation

Daniel McMurtrey, Ross Hymel
Sandia National Laboratories
dmcmurt, rwhymel @sandia.gov

September 18, 2008

Design Process Considerations

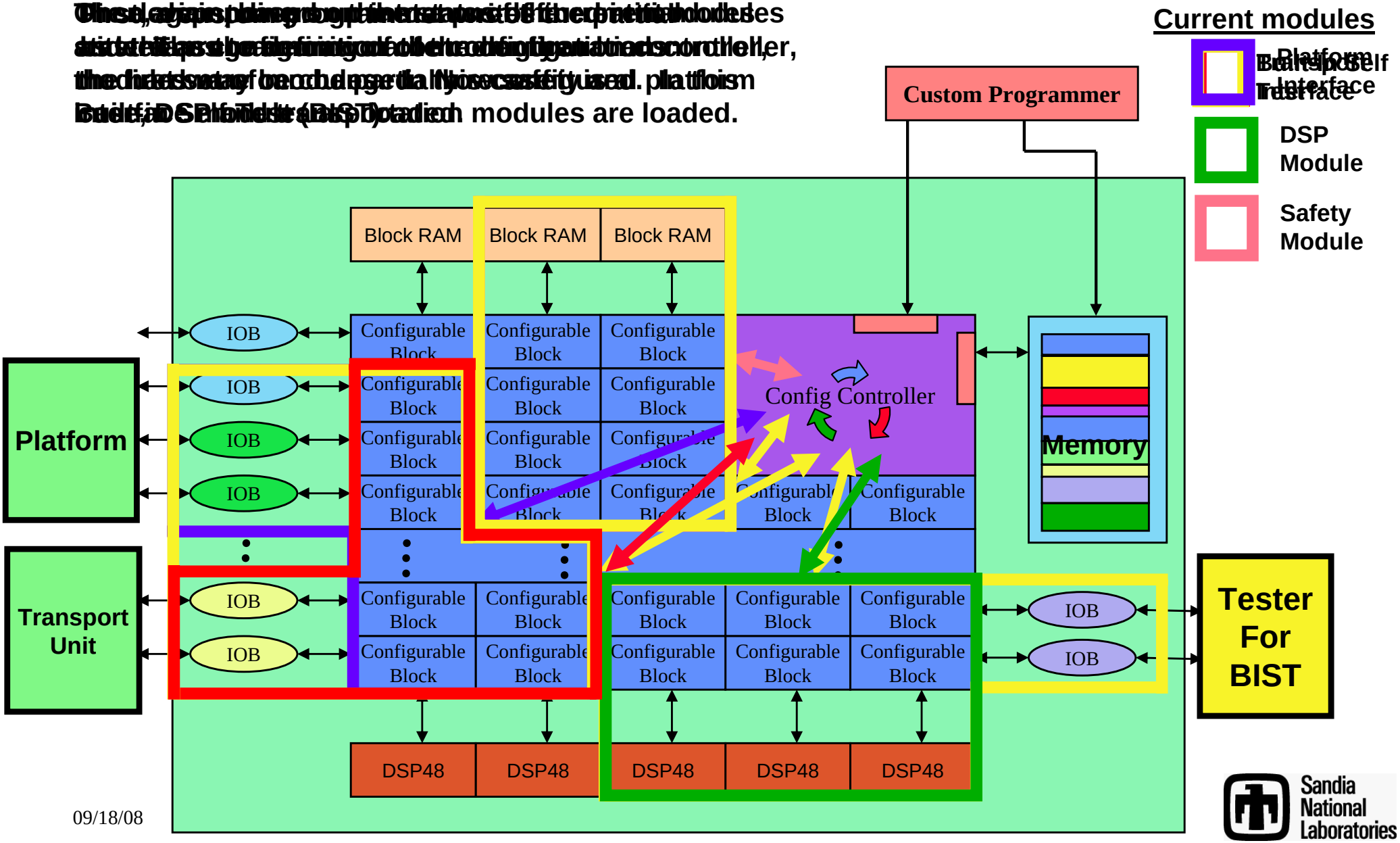
- Typical design scenarios employing FPGAs:
  - Prototyping platform
    - Validate designs targeting ASICs
      - Generally for large volume applications
  - Targeting a delivered product
    - Avoids the time and cost of an ASIC
    - Permits field upgrades of the design
      - Through full reconfiguration (easy, common)
      - Through partial reconfiguration (hard, rare)
- Can we exploit PR in delivered products?
  - PR has existed throughout the Virtex series
  - Historically avoided due to poor tool support and the complexity of reconfiguration
  - However, new tools and applications are facilitating its use
    - Xilinx PR overlay (special request access only, currently not released)



New System Design Paradigm

- **We propose a fundamentally new system architecture**
  - **Based on reconfigurable resources that are time-multiplexed by different hardware modules using partial reconfiguration**
  - **Hardware modules are designed using object-oriented software tools with vast simulation capabilities**
  - **Autonomous configuration controller to manage PR**
  - **Generic hardware can then be qualified to very strict environmental standards and used across many platforms and upgrades**
  - **This type of architecture presents many challenges and potential problems that need to be examined and addressed**

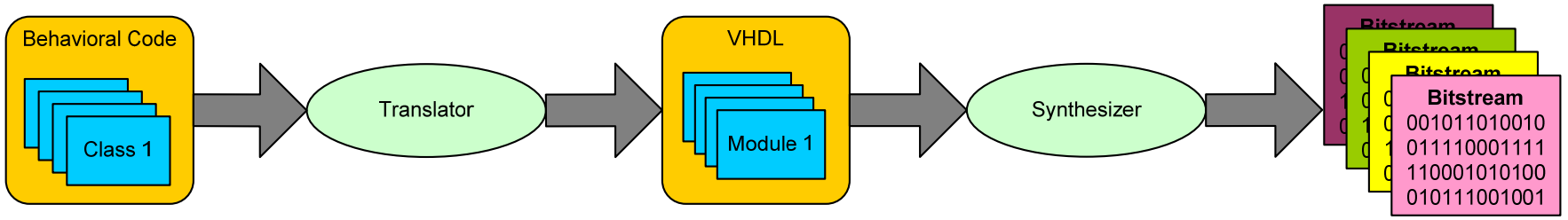
Example Implementation Targeting the Architecture



Sandia National Laboratories

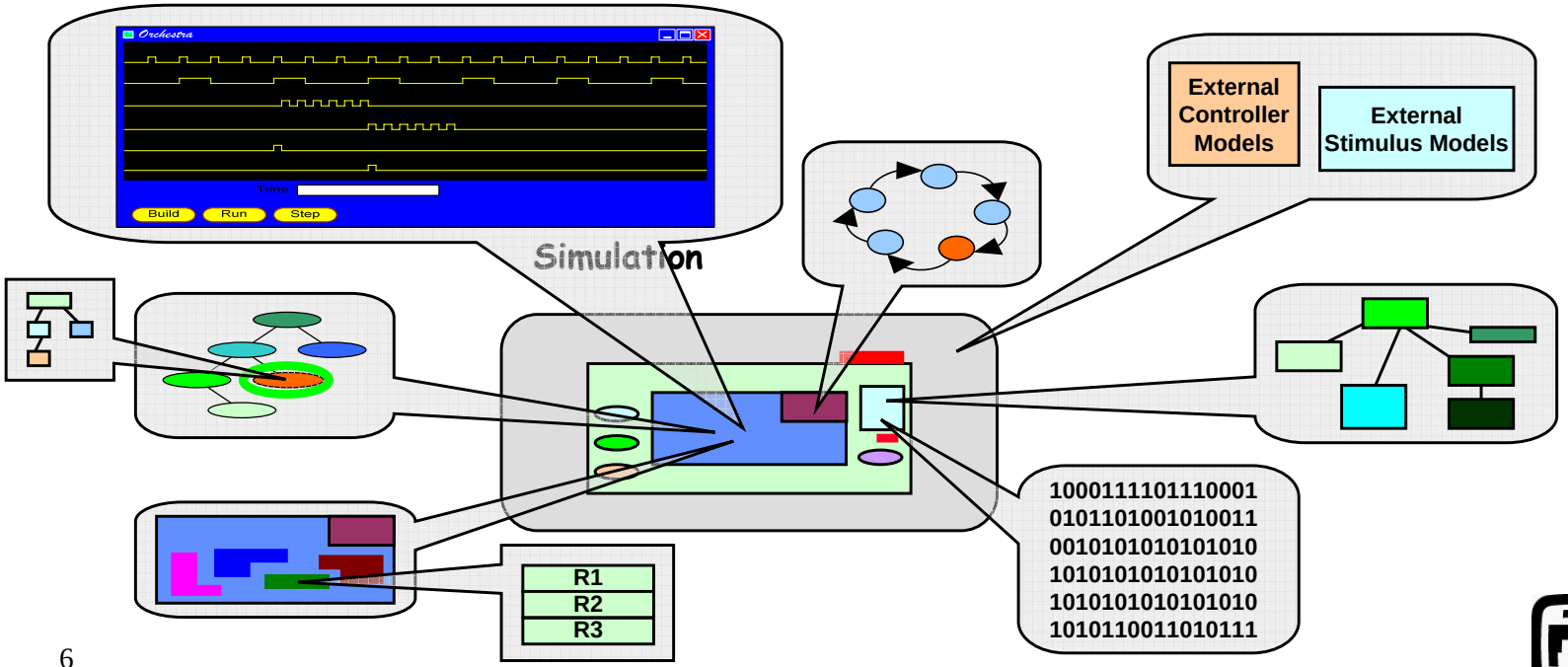
Challenges of this Configurable Architecture

- **Need a design framework (new design paradigm)**
  - Allow development of object-oriented functional models
  - Provide cycle-accurate system-level simulation capabilities covering partial reconfiguration
- **Need implementation tools**
  - Must transform high-level simulation models to partial bitstreams
  - Requires configuration controller to manage the reconfiguration process
- **Need to reliably analyze and verify the system**
  - We must ensure the safety and reliability of the hardware architecture
  - System simulations must be accurate
  - We must have confidence in the design flow



New Design Paradigm

- **Altered design flow**
  - **Look at the lifetime of the system and break the behavior into basic modules**
  - **Decide when modules need to exist**
  - **Identify states (which modules exist at which times) and state transitions**
- **Commercial HDL simulation tools (ModelSim, Active-HDL, etc.) do not support design and testing of this architecture**
  - **Unable to simulate real-time partial reconfiguration**
  - **Large VHDL simulations are very slow**

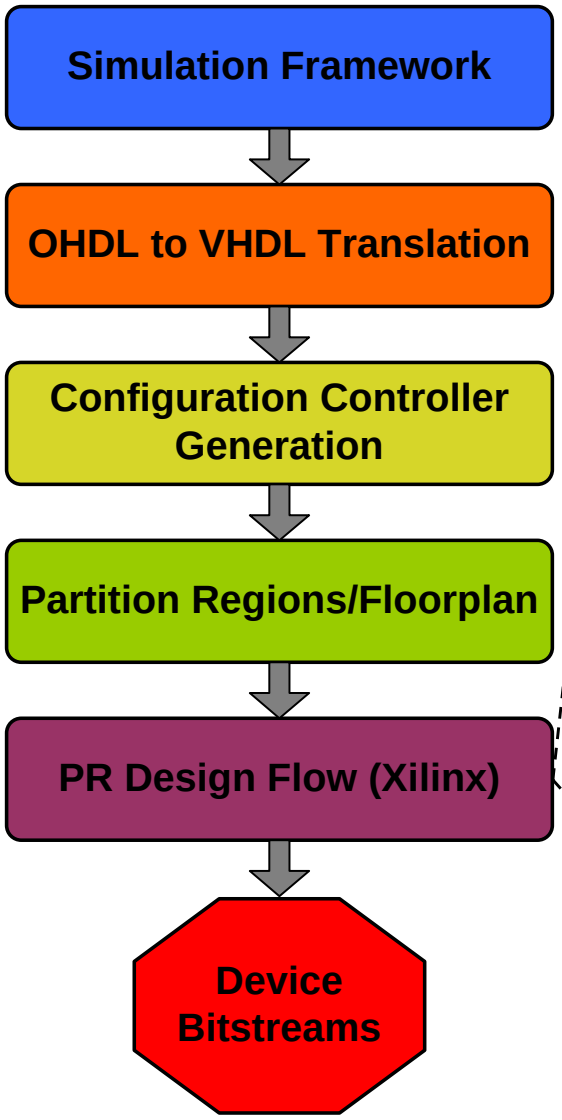


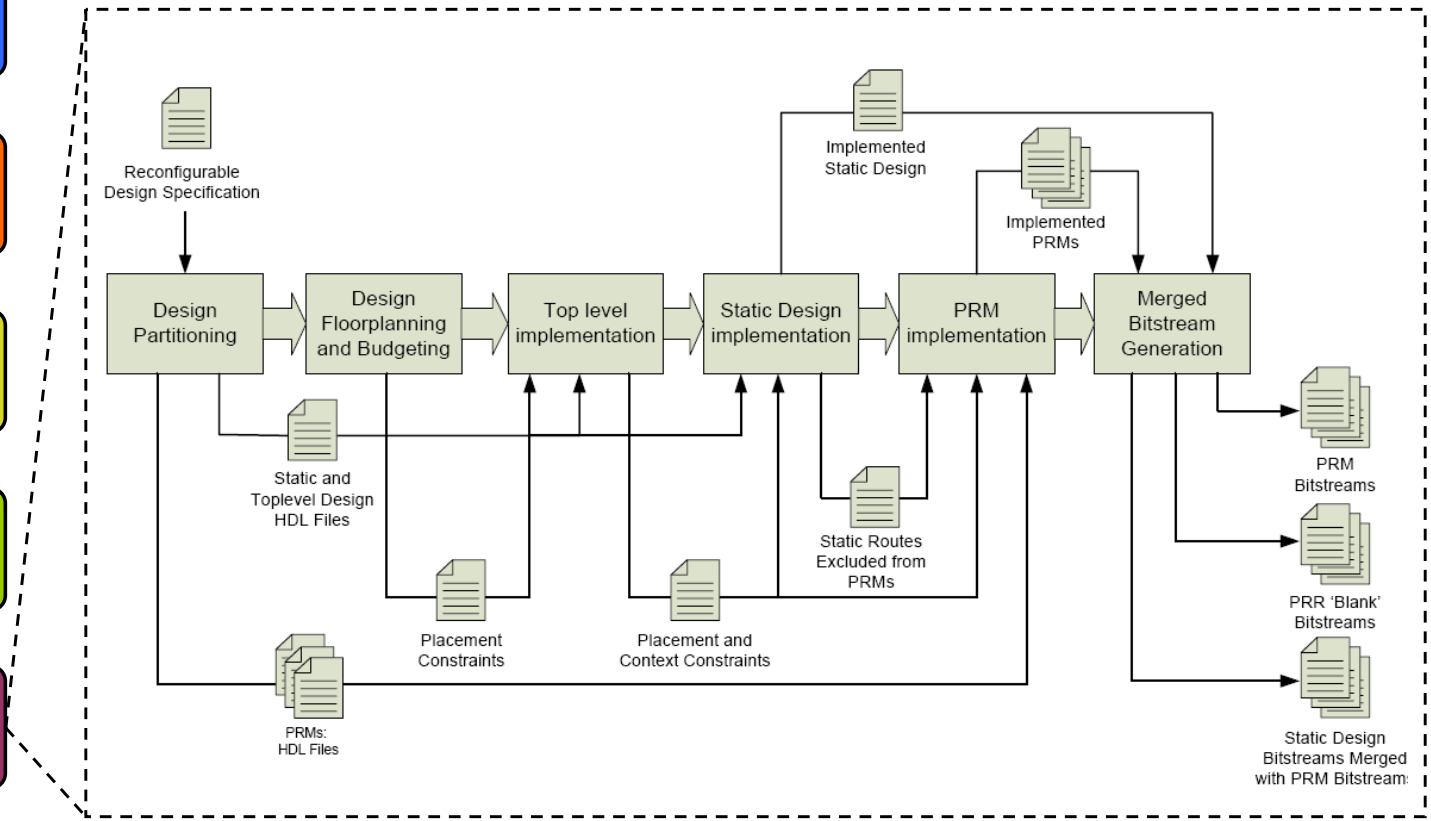
Sandia National Laboratories

New Design Paradigm (2)

- We chose to utilize Orchestra (custom Sandia design/simulation environment)
  - Java-based (Object-oriented, GUIs, mature)
  - Includes Orchestra-HDL (OHDL)
- Design process using the Orchestra tools:
  - Implement design and testbed using abstract, high-level Java models
  - Refine the models to cycle-accurate OHDL models
  - Automated tools:
    - Translate OHDL to VHDL
    - Generate a hardware-equivalent of the configuration controller in VHDL
    - Synthesize, map and place configurable regions
    - Create and load reconfigurable bitstreams onto the target platform

Design and Implementation Flow





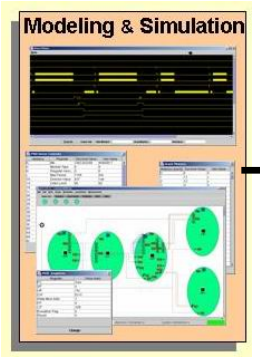
Sandia National Laboratories

Advantages of this Architecture

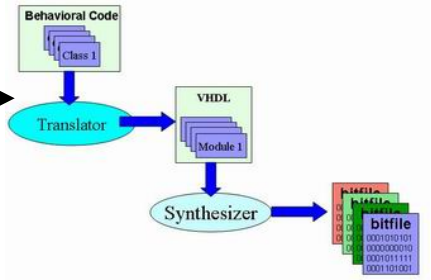
- Generic reusable hardware
  - Qualify hardware to strict environmental standards and then reuse across platforms
- Intrinsic built-in test capabilities
  - At any time, the reconfigurable regions can be configured to perform built-in testing
  - During normal operation, unused configurable regions can be used to monitor and collect information
- Responsive architecture
  - Can introduce new features or designs without changing hardware
- Design done in high-level simulation environment using object-oriented software tools
  - Provides enhanced visibility into the design and powerful simulation capabilities
  - Faster design time
- Inherent benefits:
  - This architecture easily lends itself to redundancy
    - Can improve safety and reliability
  - Encryption can be built into the bitstream loader
    - Design security

The Future of System Design?
Faster, Better, Cheaper?

Develop Functional Modules

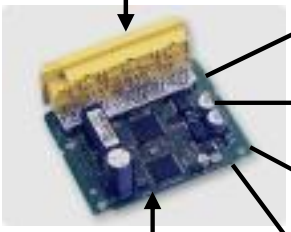


Synthesize Modules



Load Modules into Hardware

Custom Programmer

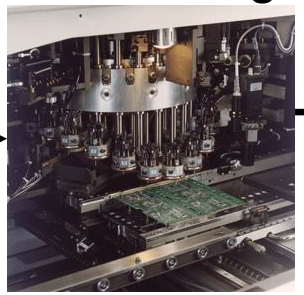


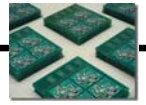
Install in Different Platforms

One Time Process

Design

Manufacturing and Testing





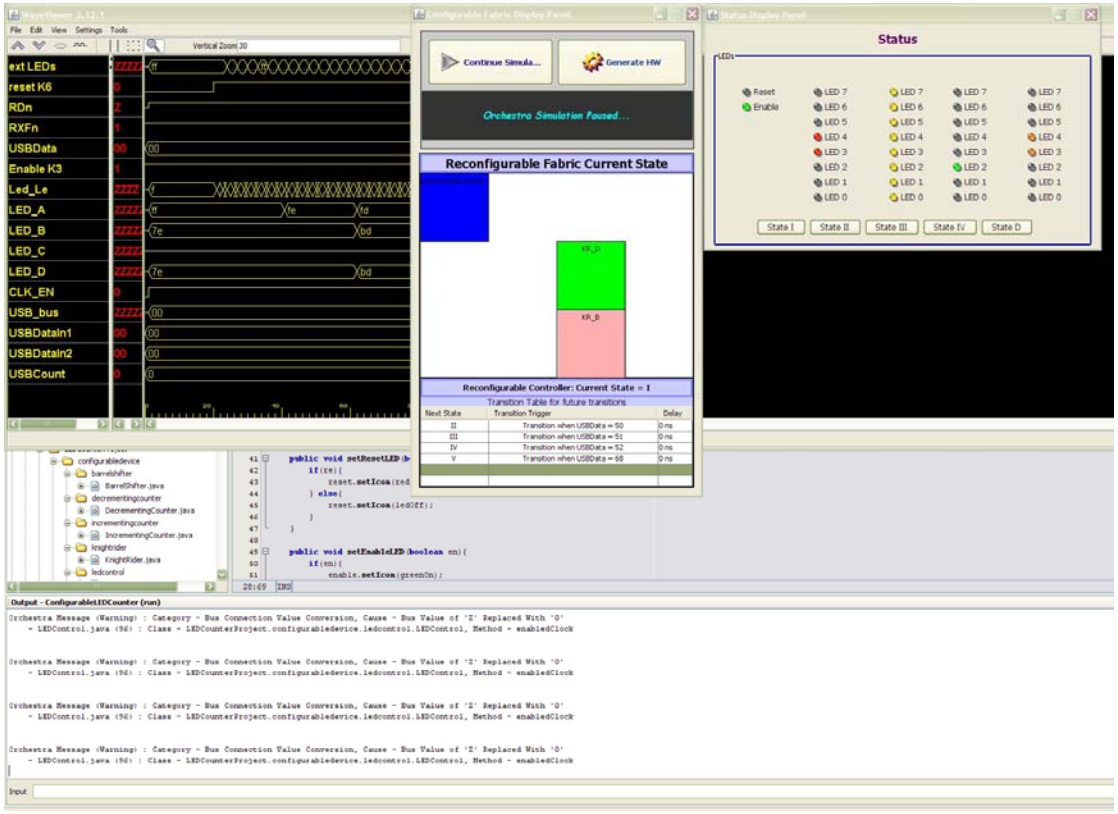
Storage

Sandia National Laboratories

Additional Slides

High-level Modeling/Simulation in Orchestra

- Design is done at higher levels of abstraction (Java and OHDL)
- Orchestra environment provides for cycle-accurate simulations
  - Enhanced visibility and debugging tools
  - Allows you to watch the condition of the configurable logic and see state transitions
- Fast, event-driven simulation engine
  - Quick debug cycles
- Significantly reduces the development time and cost

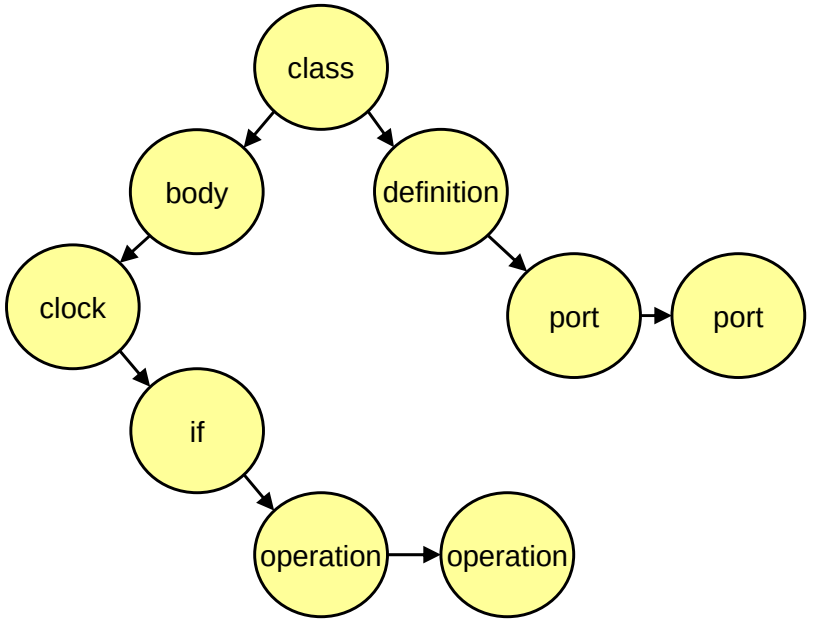


Sandia National Laboratories

OHDL to VHDL Translation

- **Modules within the reconfigurable fabric are translated to VHDL from OHDL**
  - **File is parsed into a tree structure containing all the necessary information**
  - **Tree structure is run through a VHDL generation process to generate the file**
- **OHDL and VHDL modules are behaviorally identical**
- **Hooks included to allow for translation to any other HDL or ML**

File.java



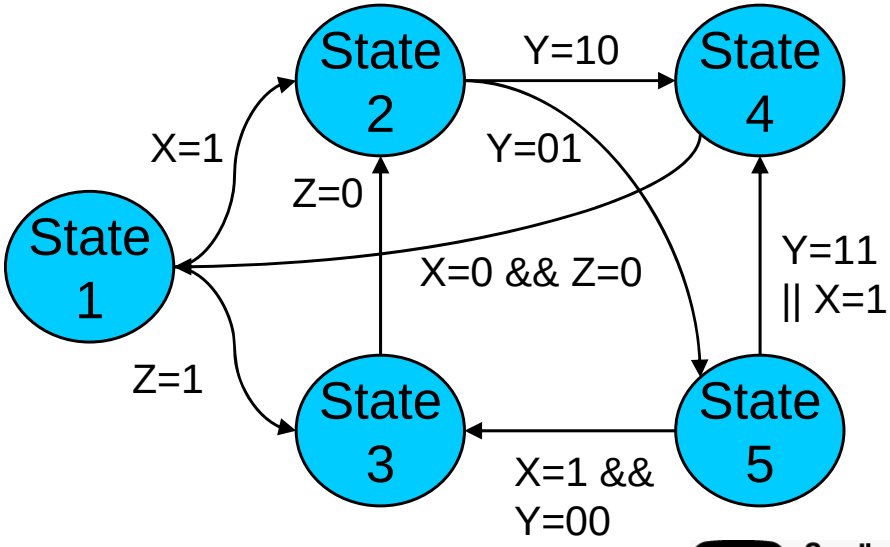
File.vhd

Sandia National Laboratories

Configuration Controller Generation

- In software, the state table must be defined
  - Determine which modules exist in which states
  - Create a unique identifier for each state as well as a priority
- Also in software, the transition table must be defined
  - What signals cause transitions from each state
  - What state to enter due to each transition
- These tables define the configuration controller and allow simulation of partial reconfiguration
- Hardware behavior during reconfiguration is user-controlled
- The tools will generate a hardware-equivalent of the configuration controller in VHDL

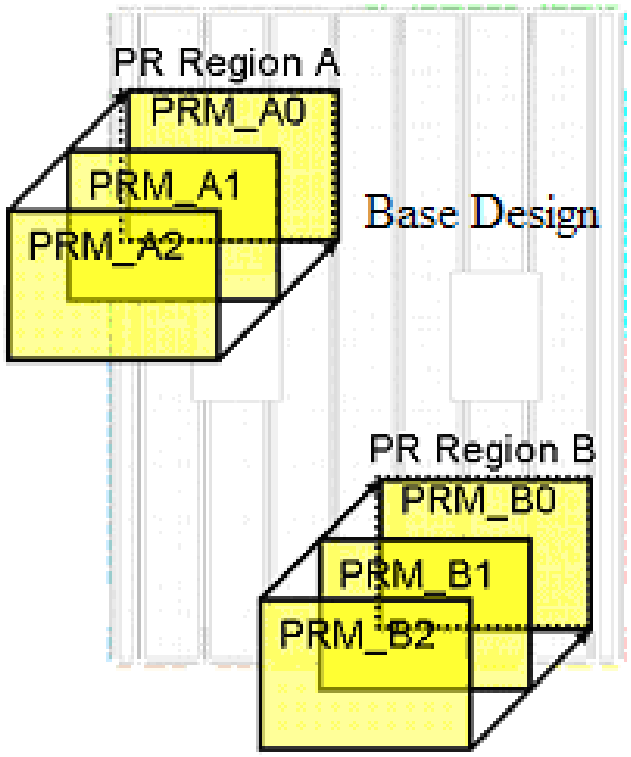
	Hardware Module			
State	A	B	C	D
1	0	0	1	1
2	0	1	1	0
3	1	0	0	1
4	1	1	0	0
5	0	1	0	0



Sandia National Laboratories

Region Partitioning/Floorplanning

- **Knowing the state table, the tools optimally allocate sets of hardware modules to share distinct regions**
  - **Requirements:**
    - **Modules that share a region must not exist at the same time in the state table**
    - **Modules that share an output port must exist in the same region**
- **Once the regions have been partitioned, they must be physically constrained onto the chip**
  - **Define the size, shape, and placement of the region**
    - **Must be large enough to accommodate the biggest module**
    - **Must be shaped and placed to allow the design to meet timing**
  - **Must obey internal FPGA placement restrictions**
    - **Frame Boundaries**
    - **Embedded I/O**
    - **RAMB16/DSP48 Column Boundaries**
  - **Must also locate and constrain bus macros**



Sandia National Laboratories

Partial Bitstream Generation

- Vendor/FPGA specific step
- Special PR Service Pack
  - Early access flow not available to the general public at this time
- Vendor design flow accessed through generated script files
- The tools generate:
  - Partial bitstreams for each reconfigurable module
  - Total bitstreams for each individual state
  - Blanking bitstreams for each reconfigurable region
- Timing is checked for each individual state
- Bitstreams are parsed and downloaded to a bank of non-volatile flash memories