Introduction to Partial Reconfiguration Methodology

Objectives

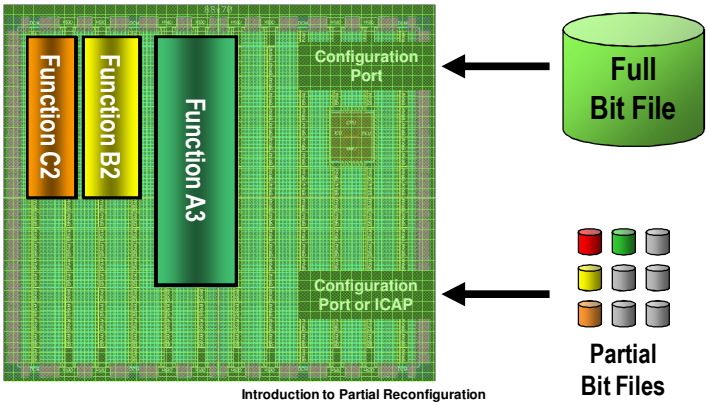
- **After completing this module, you will be able to:**
  - Define Partial Reconfiguration technology
  - List common applications for using Partial Reconfiguration
  - Define Partial Reconfiguration terminology
  - State the Partial Reconfiguration flow

Outline

- **What is Partial Reconfiguration(PR)?**
- PR Technology
- PR Terminology
- PR Design Flow
- Summary

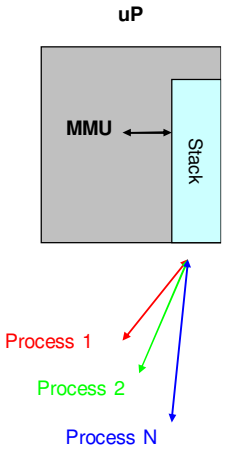
What is Partial Reconfiguration?

- **Partial Reconfiguration is the ability to dynamically modify blocks of logic by downloading partial bit files while the remaining logic continues to operate without interruption.**



PR Applications Analogy

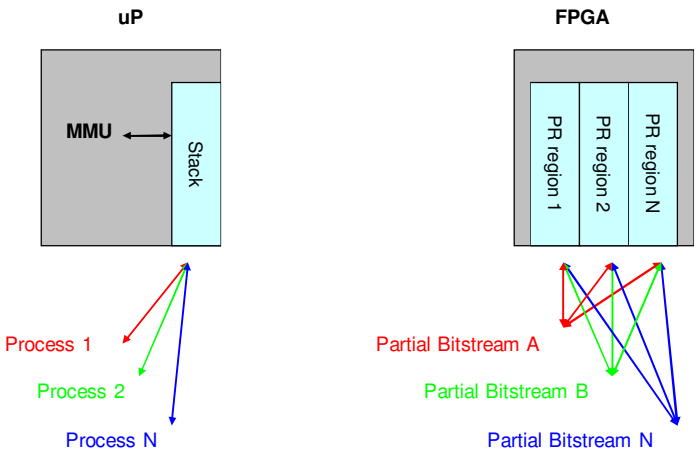
Processor Context Switch



PR Applications Analogy

Processor Context Switch

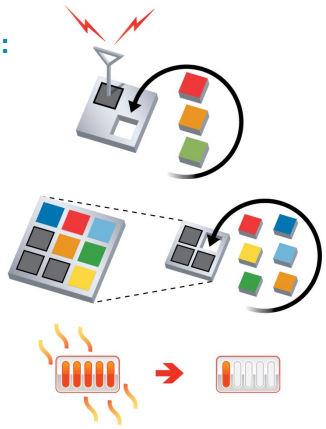
FPGA Configuration Switch



Partial Reconfiguration

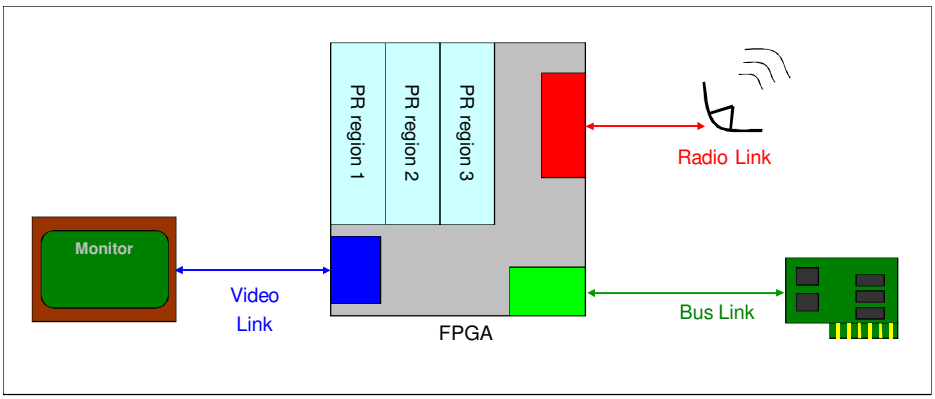
Technology and Benefits

- **Partial Reconfiguration enables:**
  - System Flexibility
    - Perform more functions while maintaining communication links
  - Size and Cost Reduction
    - Time-multiplex the hardware to require a smaller FPGA
  - Power Reduction
    - Shut down power-hungry tasks when not needed



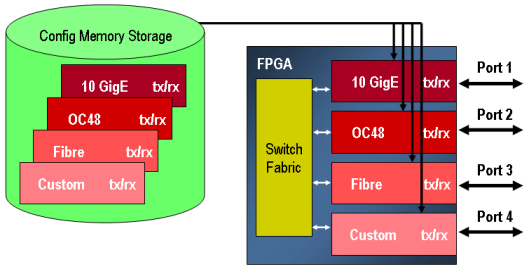
System Flexibility: Communication Hub

- **The FPGA can be a communications hub and must remain active**
  - Cannot perform full reconfiguration due to established links



Size and Cost Reduction: Time Multiplexing

- **Applications need to be able handle a variety of functions**
  - Supporting many at once can use a great deal of space
- The library of functions use case covers a wide number of applications
  - Time-based multiplexing of functions reduces device size requirement

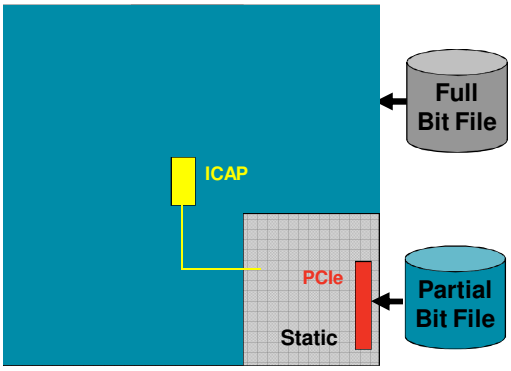


Power Reduction Techniques with PR

- **Board space and resources are limited**
  - Multi-chip solutions consume extra area, cost, and power
- **Many techniques can be employed to reduce power**
  - Swap out high-power functions for low-power functions when maximum performance is not required
  - Swap out black boxes for inactive regions
  - Swap high-power I/O standards for lower-power I/O when specific characteristics are not needed
  - Time-multiplexing functions will reduce power by reducing amount of configured logic

Common Applications: Configuration via PCIe Interface

- **PCIe enumeration time is difficult to meet with larger devices**
  - 100ms requirement is becoming more challenging to achieve
- **Configuration via PCIe is a solution that will meet two specific needs:**
  - Mechanism for reducing initial configuration time, using compressed bitfile
    - Two-stage configuration allows users to meet requirement first, then load remainder of design
  - Standard interface as access to ICAP for Partial Reconfiguration

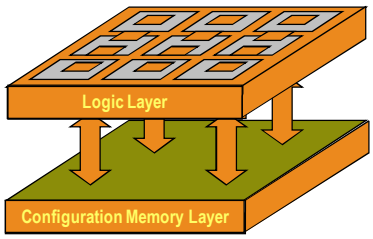


Outline

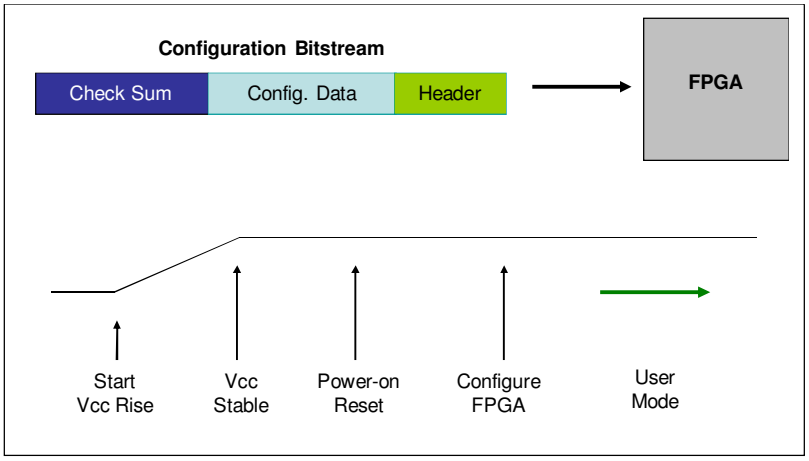
- What is Partial Reconfiguration(PR)?
- **PR Technology**
- PR Terminology
- PR Design Flow
- Summary

Programmability 101

- **Think of an FPGA as two layered device:**
  - Configuration memory layer
  - Logic layer
- **Configuration memory controls function computed on logic layer**

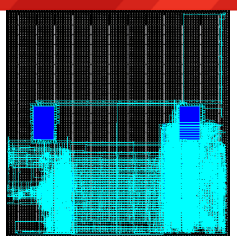


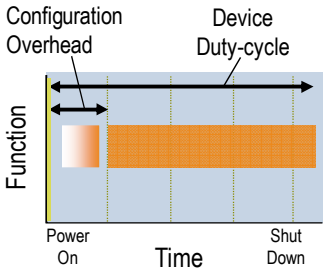
"Normal" Configuration



‘Typical’ Configuration Mode

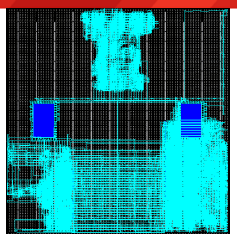
- **Fixed configuration**
  - Data loads from PROM or other source at power on
  - Configuration fixed until the end of the FPGA duty cycle
- **Used extensively during traditional design flow**
  - Evaluate functionality of design as it is developed

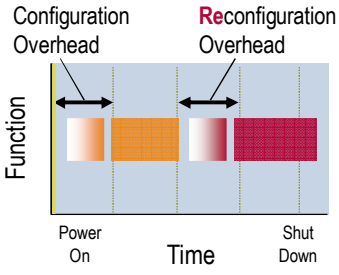




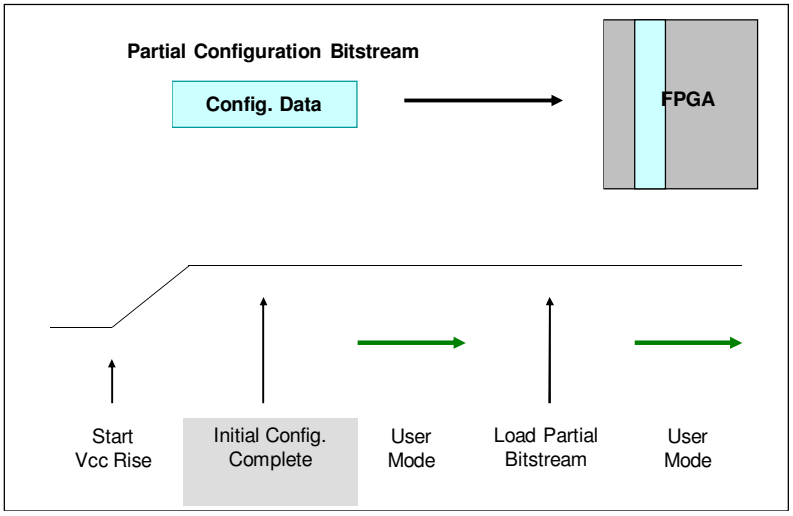
Reconfiguration

- **Configuration memory is no longer fixed during the system duty cycle**
- **Initial bitstream loaded at power-on**
- **Different, full device bitstreams loaded over time**



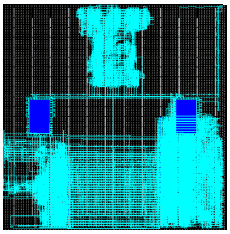


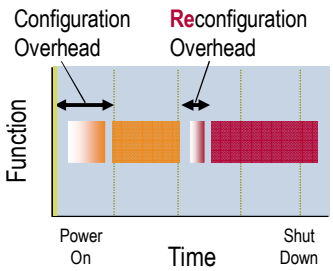
Partial Configuration



Partial Reconfiguration

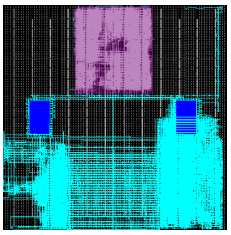
- Only a subset of configuration data is altered
- But all computation halts while modification is in progress…
- Main benefit: reduced configuration overhead

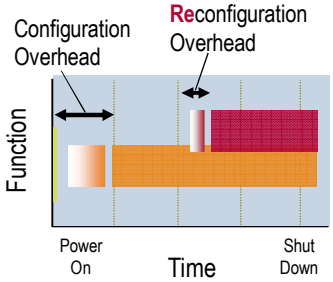




Dynamic Reconfiguration

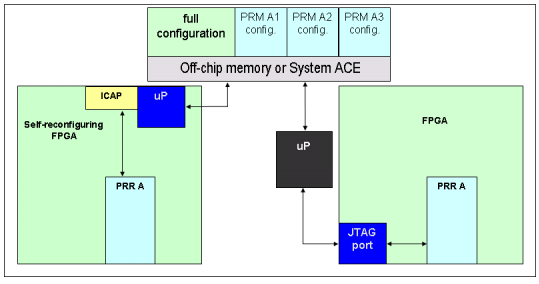
- A subset of the configuration data changes…
- But logic layer continues operating while configuration layer is modified…
- Configuration overhead limited to circuit that is changing…





How Can We Reconfigure?

- Initiation of reconfiguration is determined by the designer
  - On-chip state machine, processor or other logic
  - Off-chip microprocessor or other controller
- Delivery of the partial bit file uses standard interfaces
  - FPGA can be partially reconfigured through the SelectMap, Serial or JTAG configuration ports, or the Internal Configuration Access Port
- Logic decoupling should be synchronized with the initiation and completion of partial reconfiguration
  - Enable registers
  - Issue local reset



Outline

- What is Partial Reconfiguration(PR)?
- PR Technology
- **PR Terminology**
- PR Design Flow
- Summary

Hierarchical Implementation Definitions

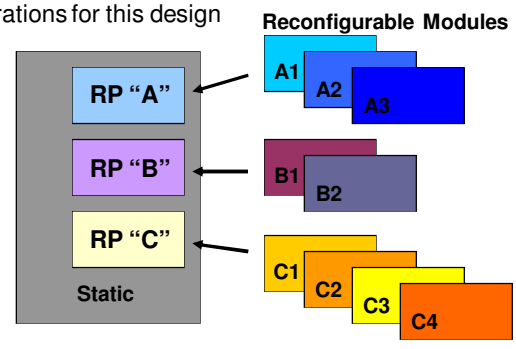
- **Partition**
  - A logical block (entity or instance) to be used for design reuse
  - User determines implementation versus preservation for each block
- **Bottom-up synthesis**
  - Separate synthesis projects resulting in multiple netlists
  - No optimization across projects
- **Top-down synthesis; NOT used for Partial Reconfiguration (normal flow)**
  - One synthesis project where synthesis flattens design for optimization
  - Often called flat synthesis
  - No support for hierarchical implementation

Terminology

- **Reconfigurable Partition (RP)**
  - Design hierarchy instance marked by the user for reconfiguration
- **Reconfigurable Module (RM)**
  - Portion of the logical design that occupies the Reconfigurable Partition
  - Each RP may have multiple Reconfigurable Modules
- **Static Logic**
  - All logic in the design that is not reconfigurable
- **Configuration**
  - A full design consisting of Static Logic and one Reconfigurable Module for each Reconfigurable Partition
- **Partition Pins**
  - Ports on a Partition; Interface between Static and Reconfigurable Logic
- **Proxy Logic**
  - LUT1 inserted on each Partition Pin to act as anchor points for RP

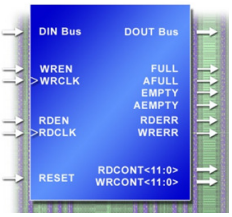
Configurations

- **A Configuration is a complete FPGA design**
  - Consists of Static Logic and one variant for each reconfigurable instance
- **Maximum number of RMs for any RP determines minimum number of Configurations required**
  - Example: Possible Configurations for this design
    1. Static + A1 + B1 + C1
    2. Static + A2 + B2 + C2
    3. Static + A3 + B2 + C3
    4. Static + A3 + B2 + C4
  - Static Logic and repeated RMs are imported
  - Any combination of RMs can be selected to create unique full bit files



Reconfigurable Elements

- **What is reconfigurable?**
  - Nearly everything in the FPGA
    - Slice logic (LUTs, flip-flops, and carry logic, for example)
    - Memories (block RAM, distributed RAM, shift register LUTs)
    - DSP blocks
    - I/O components (IOLOGIC, IODELAY, IDELAYCTRL)
- **Logic that must remain in static logic includes**
  - Clock-modifying blocks (MMCM, DCM, PLL, PMCD)
  - Global clock buffers (BUFG)
  - Device feature blocks (BSCAN, ICAP, STARTUP, or PCIE, for example)



Reconfigurable Elements

- **Granularity of reconfigurable regions vary by device family**
  - Boundaries recommended, but not required, to align to Clock Regions
  - **Virtex-6 examples**
    - **Slice region: 40 CLB high by 1 CLB wide**
    - **BRAM region: 8 RAMB36**
    - **DSP region: 16 DSP48**
    - **IOB region: 80 IOB (one bank)**
  - Virtex-5 examples
    - Slice region: 20 CLB high by 1 CLB wide
    - BRAM region: 4 RAMB36
    - DSP region: 8 DSP48
    - IOB region: 40 IOB (one bank)
  - Virtex-4 examples
    - Slice region: 16 CLB high by 1 CLB wide
    - BRAM region: 4 RAMB16 and 4 FIFO16
    - DSP region: 8 DSP48
    - IOB region: 32 IOB (one bank)
  - Bit file sizes for each of these resource types will vary

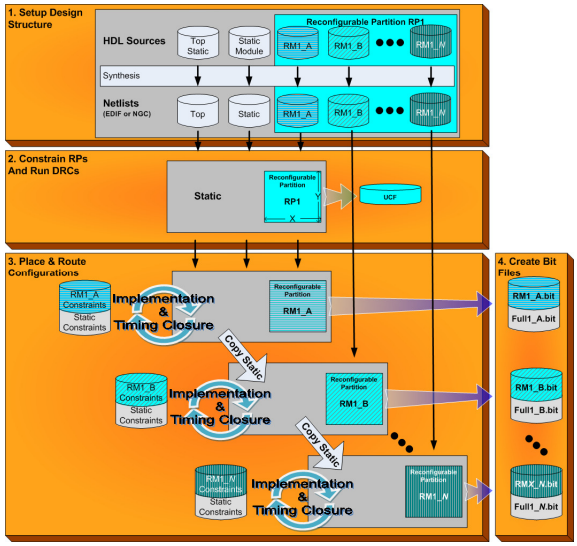
Outline

- What is Partial Reconfiguration(PR)?
- PR Technology
- PR Terminology
- **PR Design Flow**
- Summary

Flow Differences

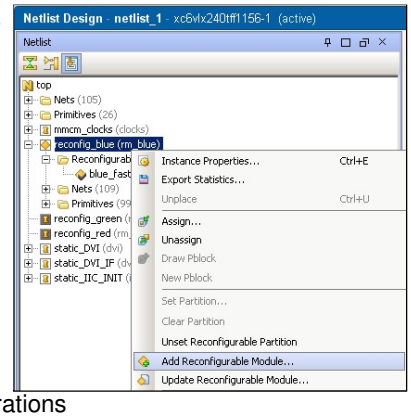
Flow	Placement	Routing
Standard	No limitations beyond device restrictions.	No limitations beyond device restrictions.
Partitions	Imported logic is placed first. Implemented logic is placed second. No Area Group requirements.	Imported logic is routed first. Implemented logic is routed second.
Partial Reconfiguration	Only reconfigurable logic can be placed in RP Area Groups unless explicitly forced with a LOC constraint. Routing restrictions considered during placement phase.	Routing resources that extend outside the RP Area Groups are not available for reconfigurable logic. Imported logic is routed first. Implemented logic is routed second.

PR Design Flow



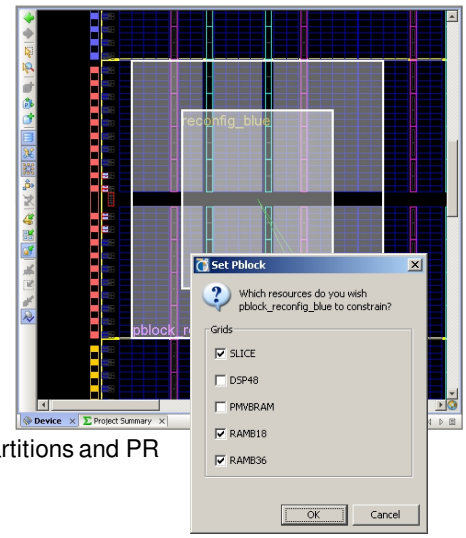
Design Flow:
1. Set Up Design Structure

- **Bottom-up synthesis creates netlists for static and reconfigurable logic**
  - Any synthesis tool can be used
- **Create PlanAhead tool PR project**
  - Import static logic and constraints
- **Define partitions and set as reconfigurable**
- **Import netlists as Reconfigurable Modules for each partition**
  - Set RMs active to build different configurations



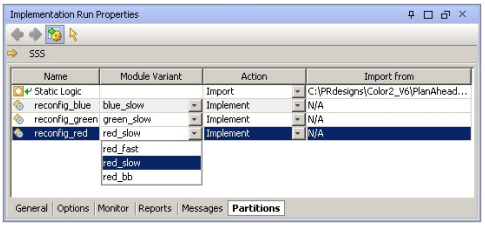
Design Flow:
2. Constrain RPs and run DRCs

- **Floorplan partition regions by creating Pblock rectangles**
  - Uses AREA_GROUP constraints to assign range
  - These declare what will be reconfigured
- **Create timing constraints**
  - Requirements should consider the entire design
  - Budget timing on both sides of partition boundary (if needed)
- **Run DRCs in the PlanAhead tool**
  - Specific sets of rules checked for partitions and PR



Design Flow:
3. Place & Route Configurations

- **Uses existing Run command**
  - Give unique name to each Configuration
    - Select the RMs desired for each RP
  - Allows multiple runs to be created for same configuration for exploration



- **Strategy: Implement most difficult configuration first**
  - Once the largest or most timing-critical RMs are resolved, the other scenarios should be easier to manage
- **Promote "golden" versions of each RM and static logic**
  - Import these for subsequent configurations

Design Flow: 4. Create Bit Files

- **First, verify that all PR design rules have been followed**
  - PR_Verify will check validity of selected Configurations
- **Use Bitstream Generation command in PlanAhead**
  - Will run **bitgen** on all selected Configurations
    - Generates full and partial bit files for each run in implementation directory
  - Can be launched for any design run created for any configuration
- **Normal simulation and timing analysis can be performed on any Configuration**
  - A Configuration is a complete FPGA design
  - Build any Configuration through Place & Route to simulate that combination of active Reconfigurable Modules

Configuration Details

- **Partial bit files are processed just like full bit files**
  - Bit file sizes will vary depending on region size and resource type
  - Contain just address & data, sync & desync words, final CRC value
    - No startup sequences, DONE flag
- **Partial Reconfiguration time depends on two factors:**
  1. Configuration bandwidth

Configuration Mode	Max Clock Rate	Data Width	Max Bandwidth
SelectMap / ICAP	100 MHz	32-bit	3.2 Gbps
Serial Mode	100 MHz	1-bit	100 Mbps
JTAG	66 MHz	1-bit	66 Mbps

  2. Partial bit file size
     - Estimate in PlanAhead, confirm in Rawbit file

Outline

- What is Partial Reconfiguration(PR)?
- PR Technology
- PR Terminology
- PR Design Flow
- **Summary**

Summary

- **Partial Reconfiguration is an Expert Flow**
- **Understanding PR terminology provides a commonality for PR design communication**
- **PR enables**
  - System flexibility
  - Size and cost reduction
  - Power reduction
- **The PR flow has four primary steps**
  1. Set up the design structure
  2. Constrain RPs and run DRCs
  3. Place & Route configurations
  4. Create bit files