Introduction to Partial Reconfiguration Methodology

Version 2016.3

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

XILINX ❯ ALL PROGRAMMABLE.

Outline

> *What is Partial Reconfiguration(PR)?*

> PR Technology

> PR Terminology

> PR Design Flow

> Summary
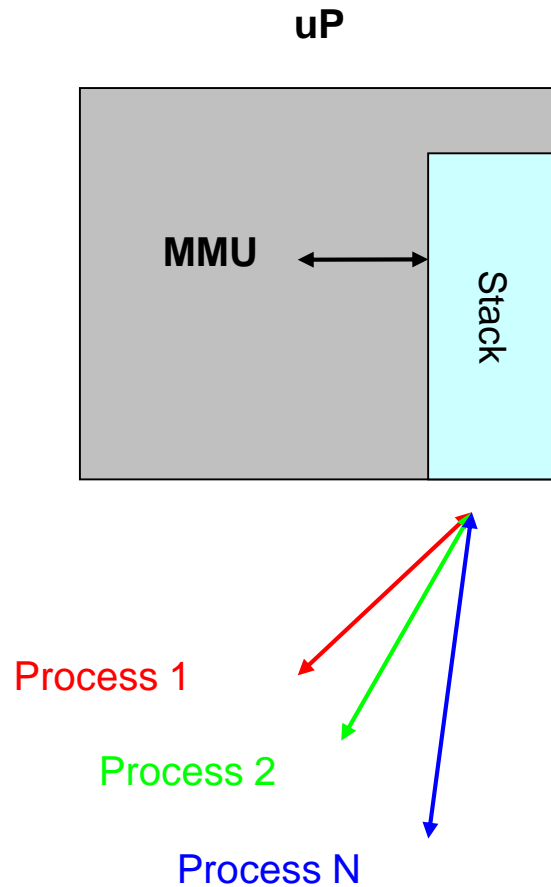
⊱ XILINX ➤ ALL PROGRAMMABLE.™

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.
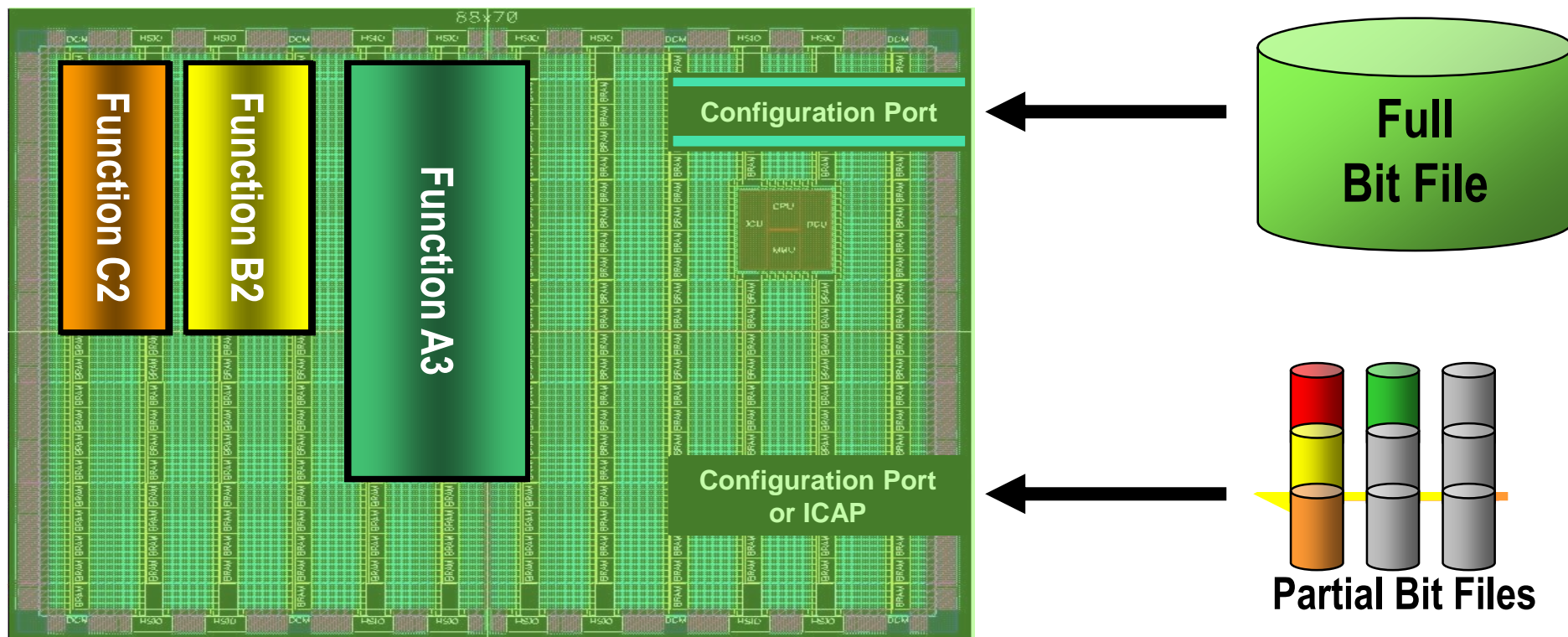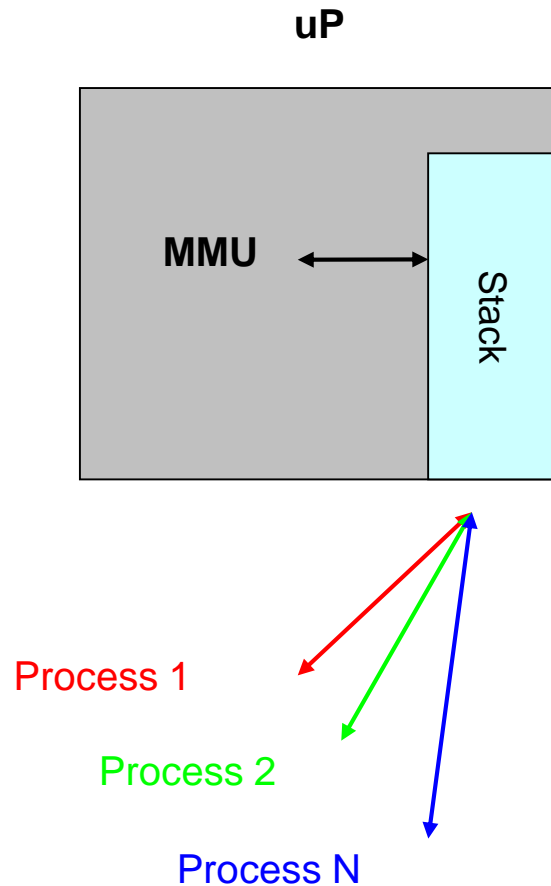
PR Applications Analogy

Processor Context Switch

PR Applications Analogy

Processor Context Switch

FPGA Configuration Switch

uP

MMU ←→ Stack

Process 1

Process 2

Process N

FPGA

PR region 1 | PR region 2 | PR region N

Partial Bitstream A

Partial Bitstream B

Partial Bitstream N

© Copyright 2016 Xilinx

XILINX ➤ ALL PROGRAMMABLE.
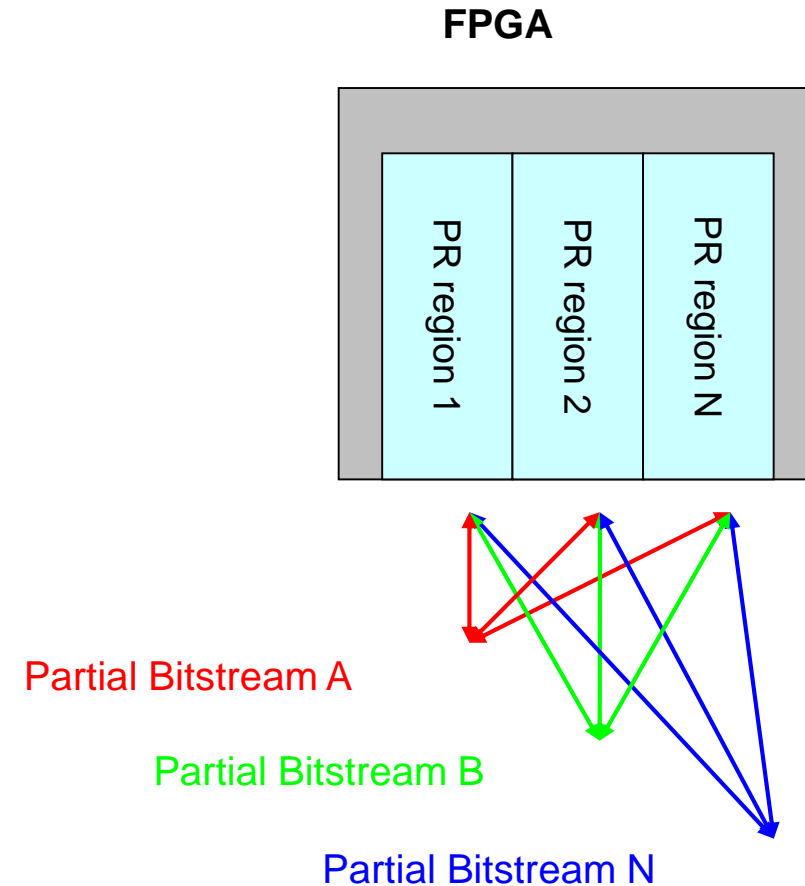
Partial Reconfiguration

Technology and Benefits

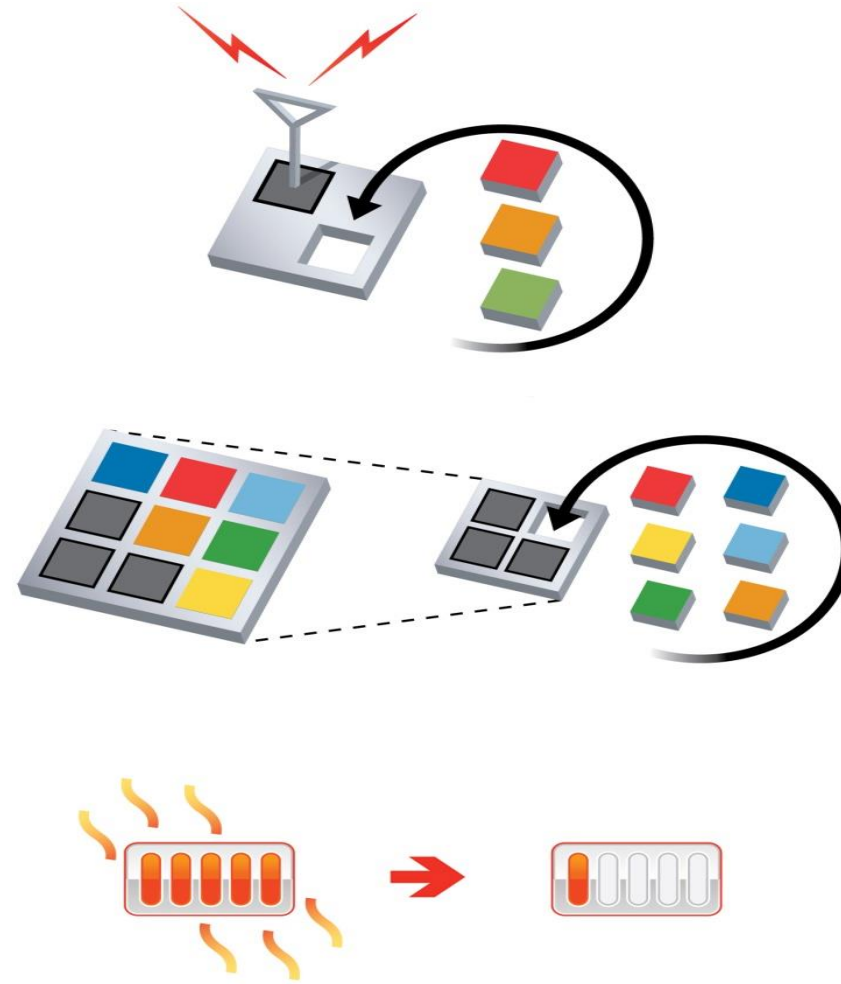➤ **Partial Reconfiguration enables:**

– System Flexibility

  • Swap functions and perform remote updates while system is operational

– Size and Cost Reduction

  • Time-multiplexing hardware requires a smaller FPGA

  • Reduces board space

  • Minimizes bitstream storage

– Power Reduction

  • Via smaller or/and fewer devices

  • Swap out power-hungry tasks

⚡ XILINX ➤ ALL PROGRAMMABLE™

System Flexibility: Communication Hub

➤ The FPGA can be a communications hub and must remain active
  – Cannot perform full reconfiguration due to established links
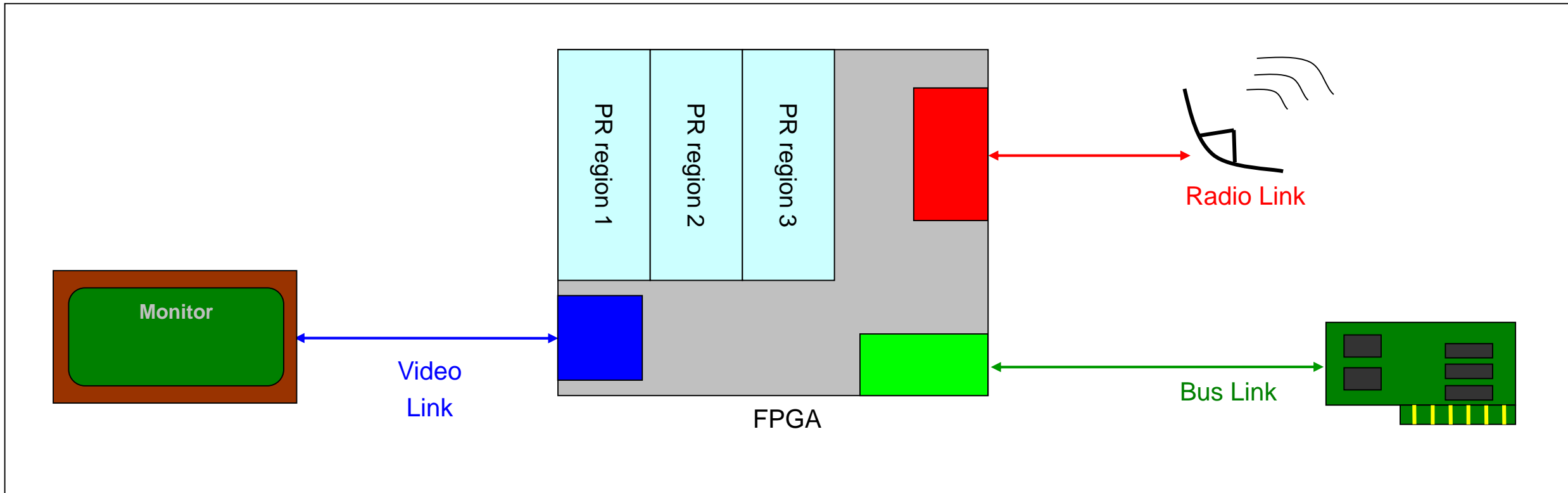
Size and Cost Reduction: Time Multiplexing

❯ **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

€ XILINX ❯ ALL PROGRAMMABLE.™
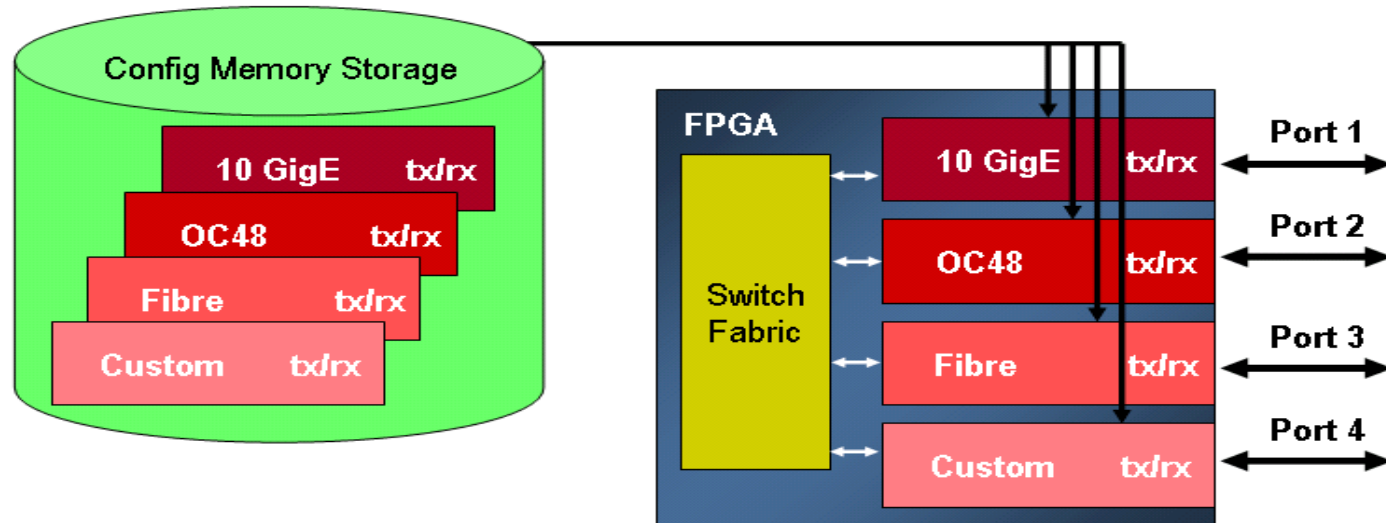
Power Reduction Techniques with PR

> **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

≨ XILINX ➤ ALL PROGRAMMABLE.

Customer Example

Accelerated Parallel Processing

- ❯ Edico Genome has created a bio-IT processor designed to analyze next-generation sequencing data
  - – Performs genome and exome sequencing for a variety of applications
  - – Load many pipelines such as genome, exome, transcriptome, microbiome and cancer
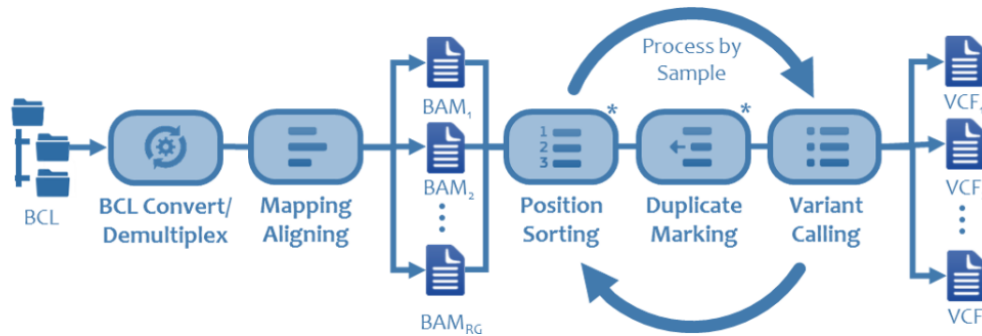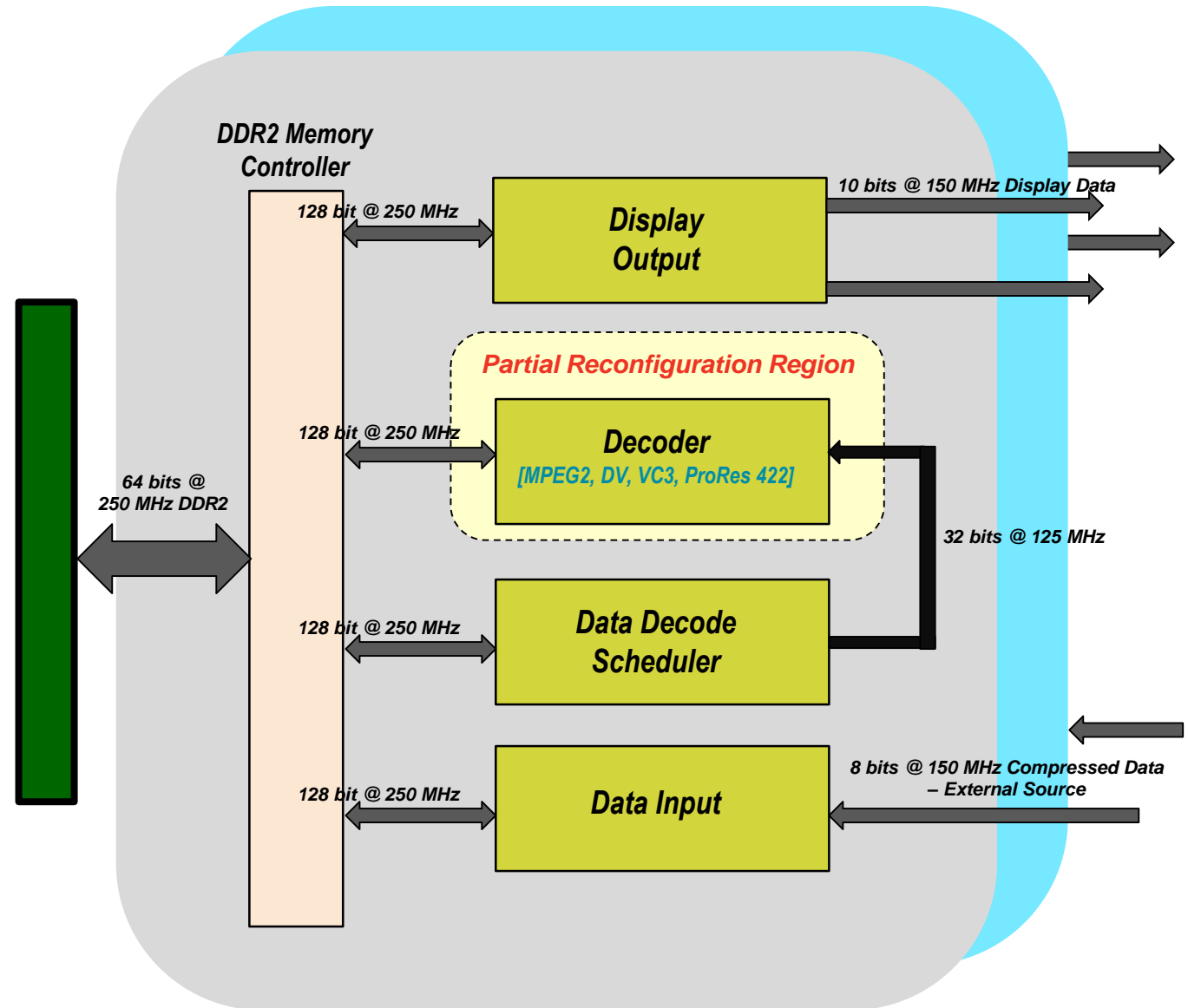

- ❯ DRAGEN card serves as a hardware accelerator
  - – 7VX980T has four Reconfigurable Partitions to load accelerator engines on the fly
  - – Partial Reconfiguration helps improve performance on hardware by **70-80X!**
    - • Compared to software-only solution on 24-core dual-CPU Intel-based server

© Copyright 2016 Xilinx

XILINX ❯ ALL PROGRAMMABLE.

Customer Example

Flexible Video Processing

> ❯ Swap decoders on the fly
- One channel remains up while the other changes

> ❯ Customer released "flat" version first
- Two decoders per channel

> ❯ Expanded functionality without changing hardware
- Deployed new bitstreams for more decoders without changing hardware

DDR2 Memory Controller

128 bit @ 250 MHz

Display Output

10 bits @ 150 MHz Display Data

Partial Reconfiguration Region

128 bit @ 250 MHz

Decoder
[MPEG2, DV, VC3, ProRes 422]

64 bits @ 250 MHz DDR2

32 bits @ 125 MHz

128 bit @ 250 MHz

Data Decode Scheduler

128 bit @ 250 MHz

Data Input

8 bits @ 150 MHz Compressed Data – External Source

© Copyright 2016 Xilinx

❯ XILINX ❯ ALL PROGRAMMABLE.

Customer Example
Hardware Acceleration

> **Dyplo = DYnamic Process LOader**
> – Solution distributes software functions between hardware and software spaces

> **Dyplo's unique selling points**
> – Optimized use of Zynq SoC device
> – Software driven Hardware development
> – Abstraction of implementation choices to system level
> – Simple use of partial reconfiguration blocks in hardware
> – Configuration Wizard tool to guarantee ease of use

> **Product launched in March 2015**
> – Read more at http://topic.nl/en/dyplo or Xcell issue 85
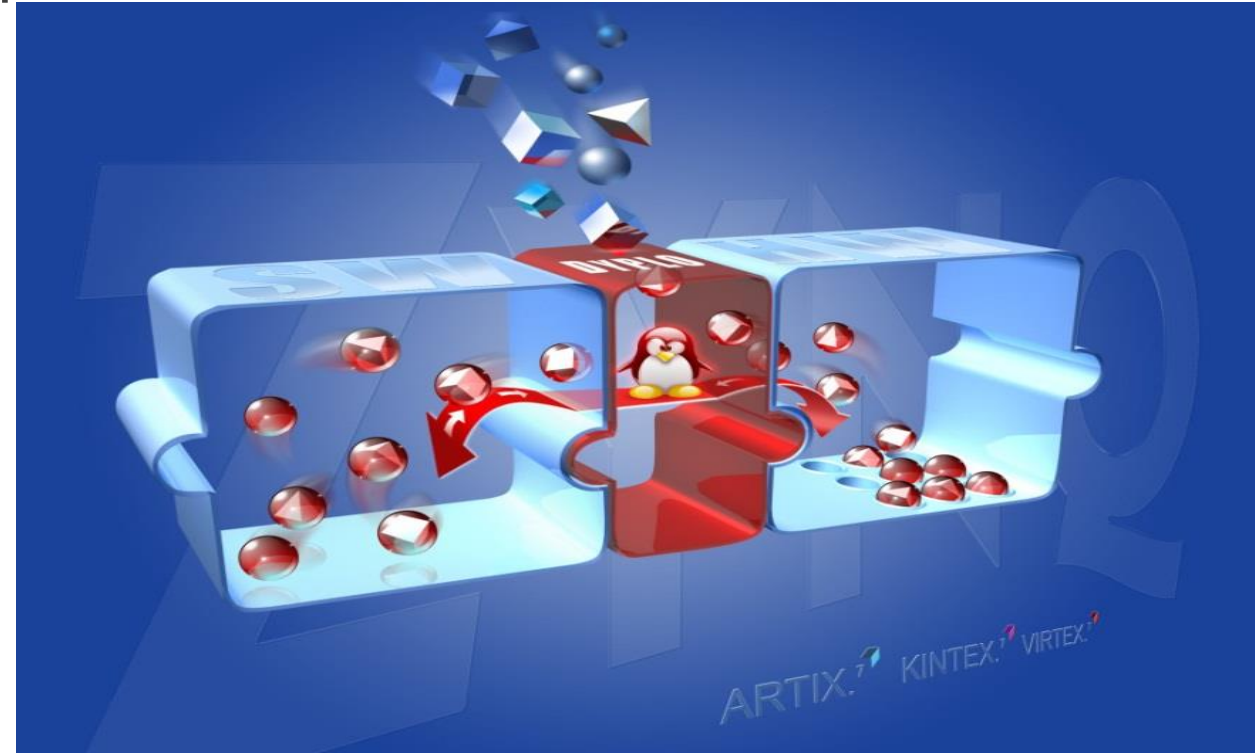
Outline

> **What is Partial Reconfiguration(PR)?**

> *PR Technology*

> PR Terminology

> PR Design Flow

> Summary

XILINX > ALL PROGRAMMABLE.

Programmability 101

❯ Think of an FPGA as two layered device:
– Configuration memory layer
– Logic layer

❯ Configuration memory controls function computed on logic layer

Logic Layer

Configuration Memory Layer

XILINX ❯ ALL PROGRAMMABLE™
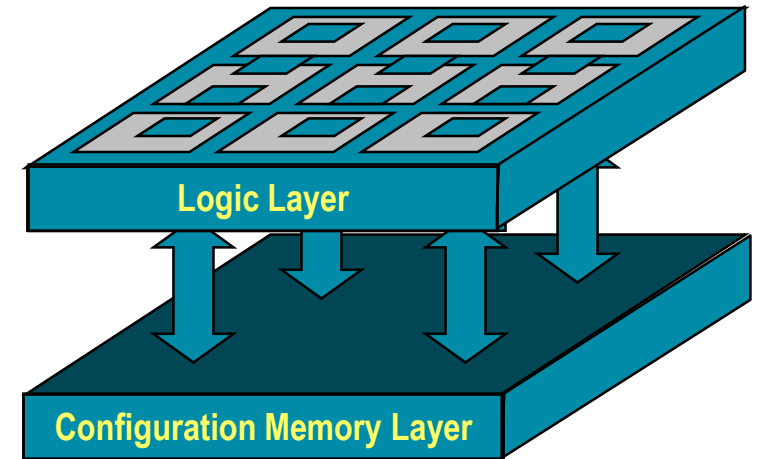
"Standard" Configuration

© Copyright 2016 Xilinx
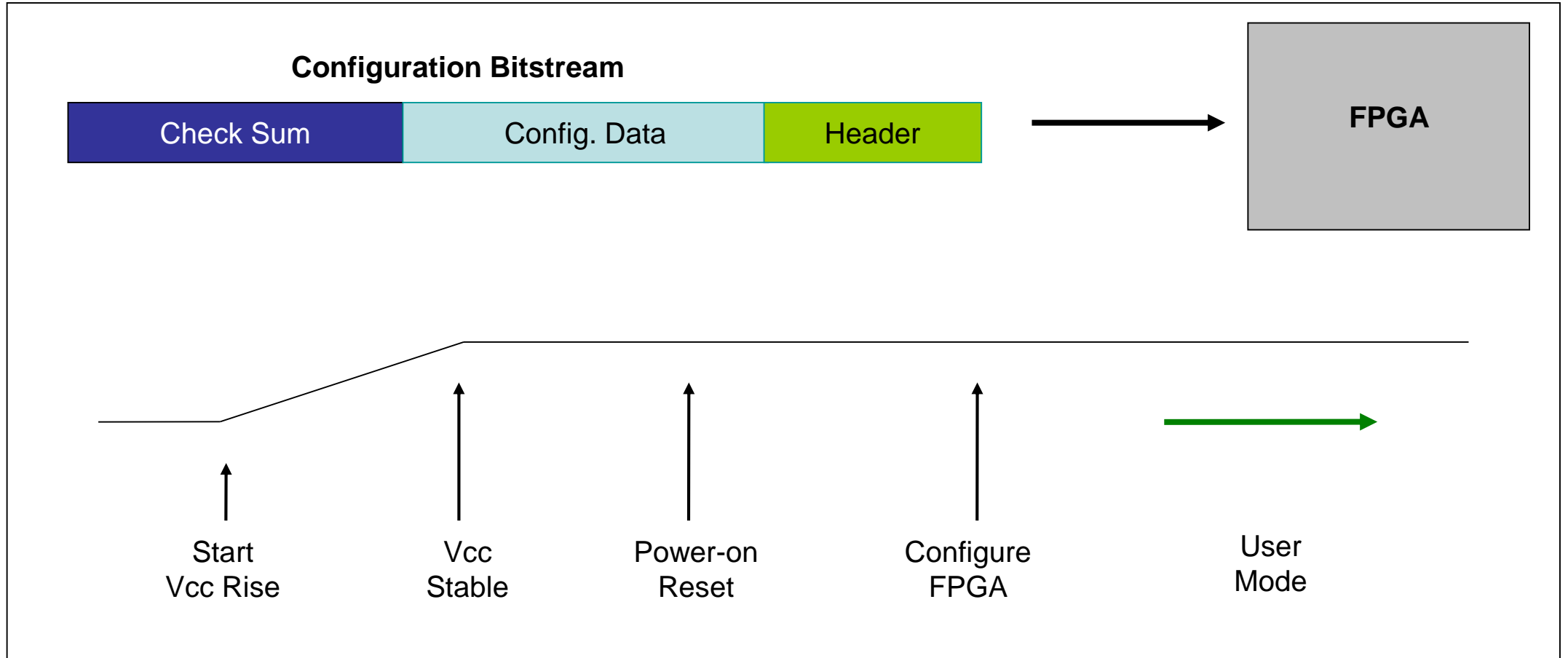
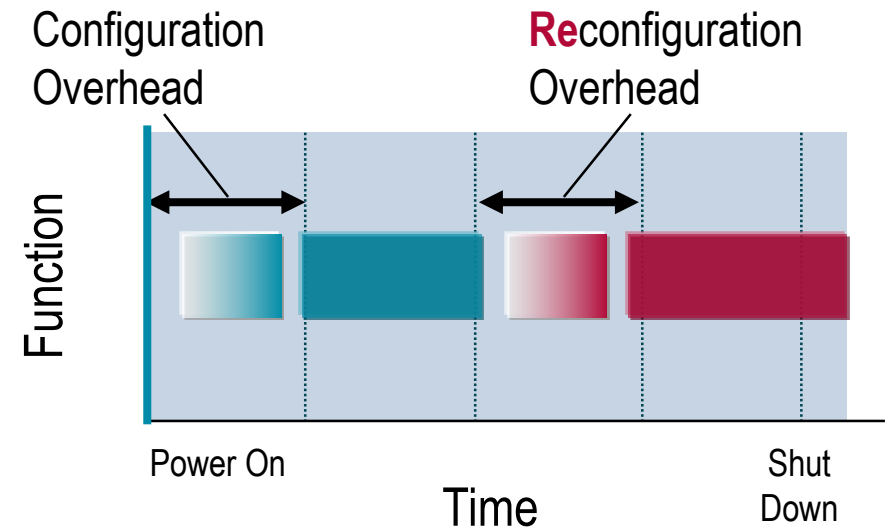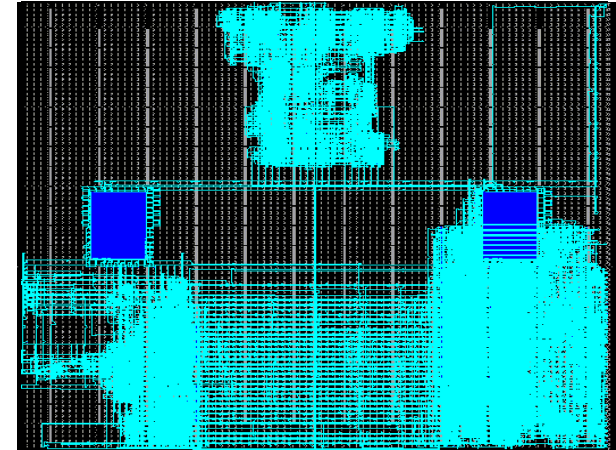# "Typical" Configuration Mode

▶ 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

Configuration Overhead   Device Duty-cycle

Function

Power On   Shut Down

Time

ΣXILINX ➤ ALL PROGRAMMABLE.
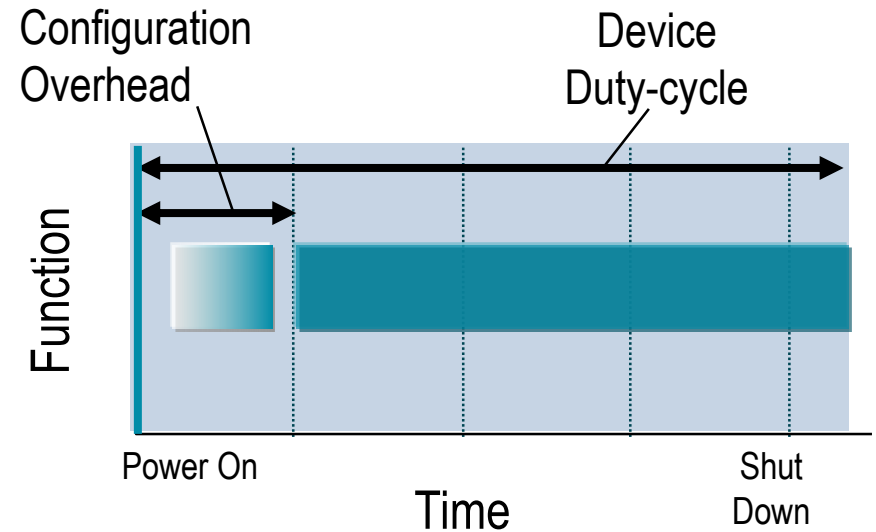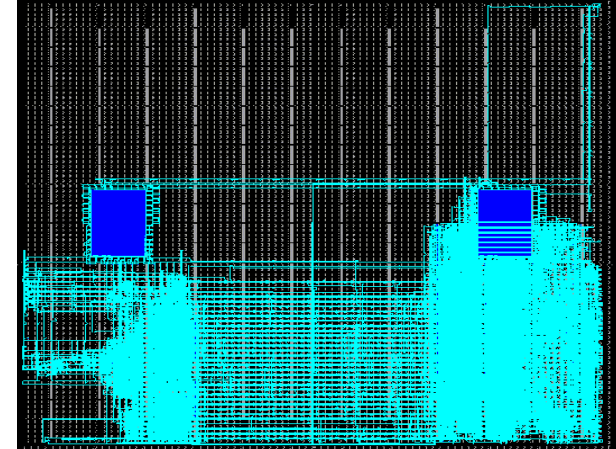
Reconfiguration

▸ Configuration memory is no longer fixed during the system duty cycle

▸ Initial bitstream loaded at power-on

▸ Different, full device bitstreams loaded over time

Configuration Overhead   **Re**configuration Overhead

Function

Power On   Time   Shut Down

XILINX ➤ ALL PROGRAMMABLE.

Partial Configuration

Partial Configuration Bitstream

Config. Data

FPGA

Start
Vcc Rise

Initial Config.
Complete

User
Mode

Load Partial
Bitstream

User
Mode

XILINX ➤ ALL PROGRAMMABLE.
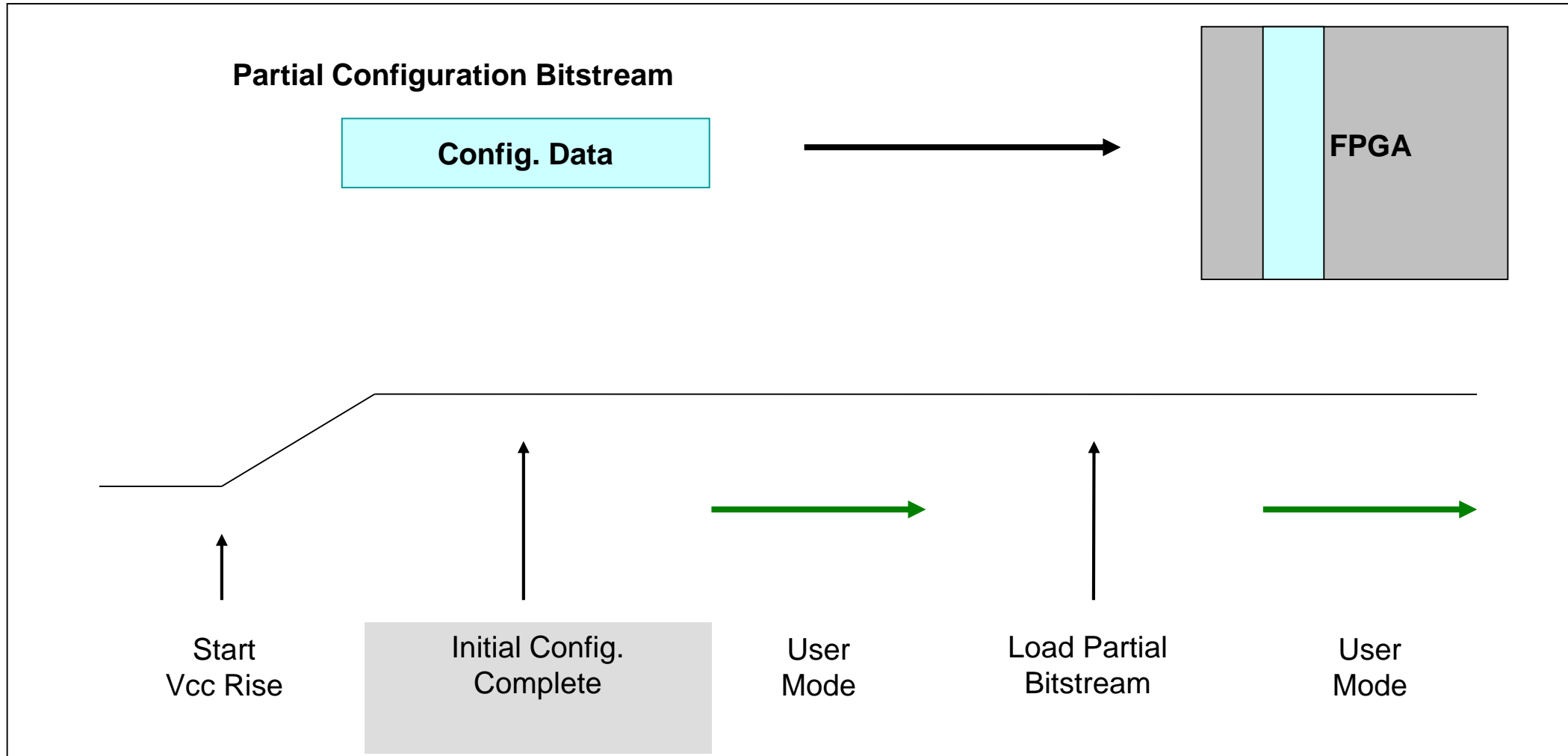
Partial Reconfiguration

❯ Only a subset of configuration data is altered

❯ But all computation halts while modification is in progress…

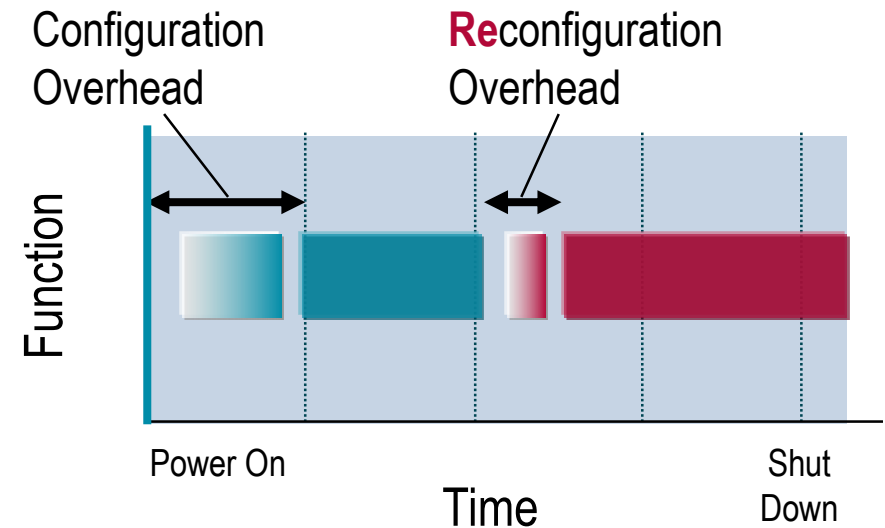❯ Main benefit: reduced configuration overhead

Configuration Overhead

Reconfiguration Overhead

Function

Power On

Time

Shut Down

XILINX ❯ ALL PROGRAMMABLE.
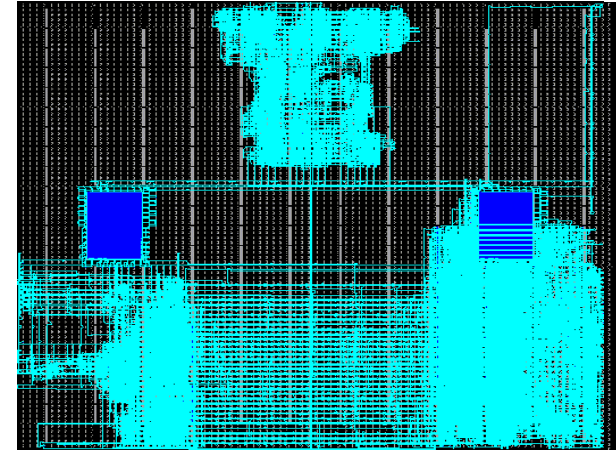
Dynamic Reconfiguration

➤ 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…

© Copyright 2016 Xilinx

XILINX ➤ ALL PROGRAMMABLE.
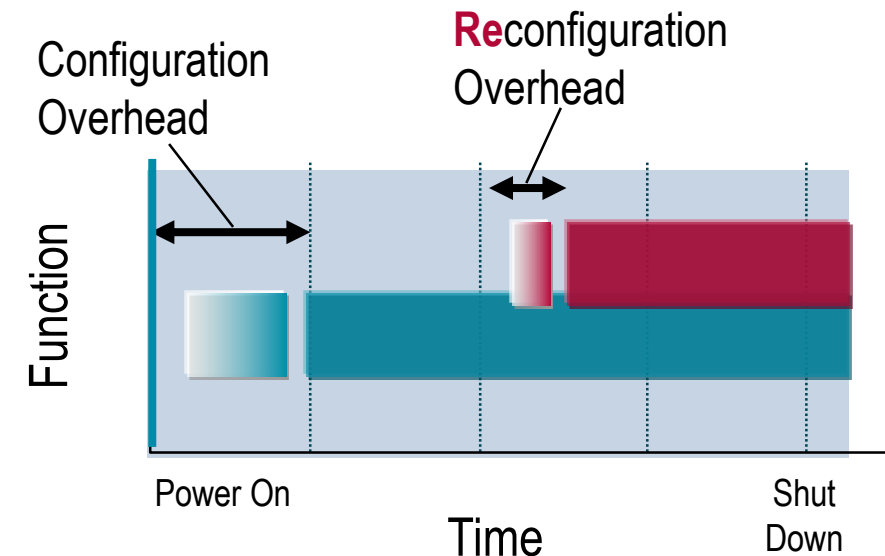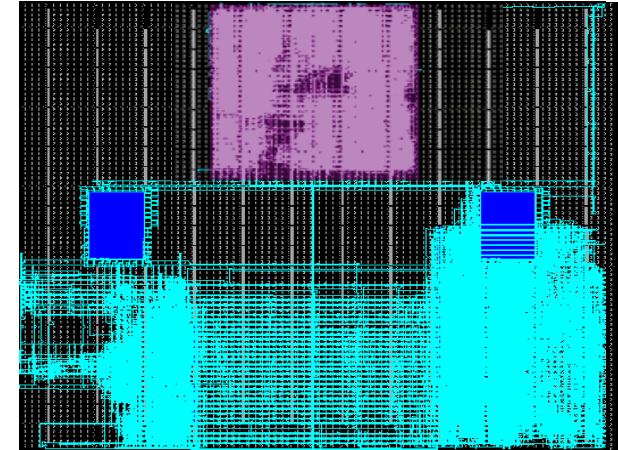
How Can We Reconfigure?

➤ 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, Processor Configuration Access Port (PCAP) in Zynq devices or the Internal Configuration Access Port (ICAP)

➤ Logic decoupling should be synchronized with the initiation and completion of partial reconfiguration
- Enable registers
- Issue local reset
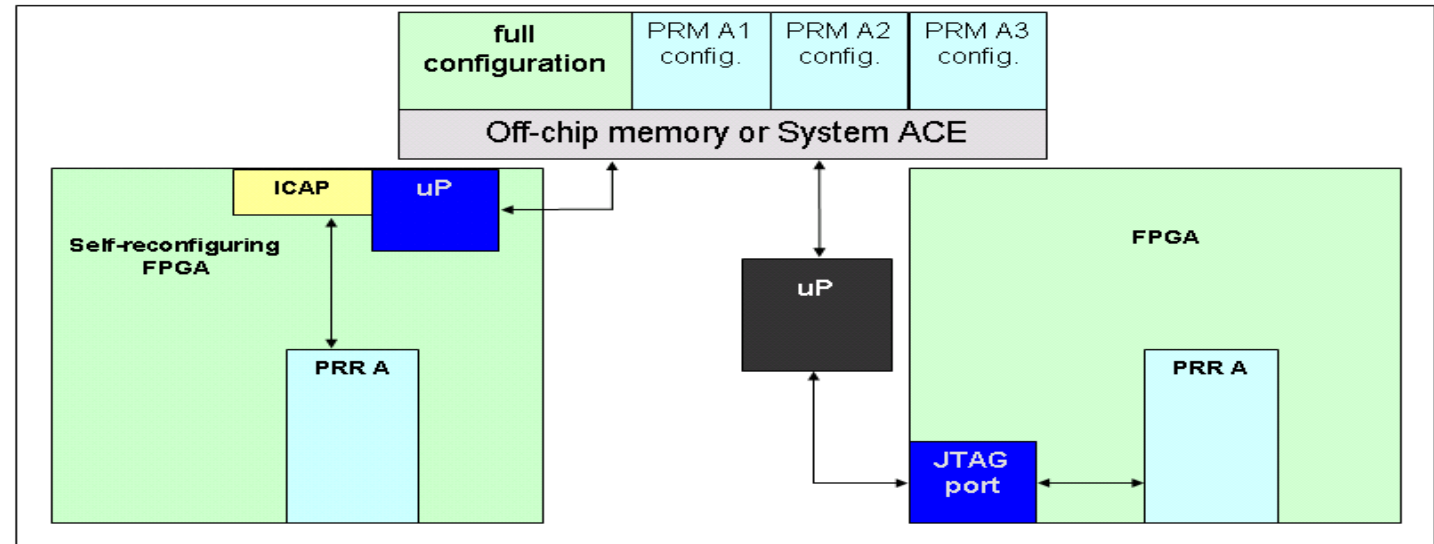
ΣΧ XILINX ➤ ALL PROGRAMMABLE.

Outline

> **What is Partial Reconfiguration(PR)?**

> **PR Technology**

> *PR Terminology*

> **PR Design Flow**

> **Summary**

≨ XILINX ➤ ALL PROGRAMMABLE.

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 or design checkpoints
  – 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

🔆 XILINX ➤ ALL PROGRAMMABLE.

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 image consisting of Static Logic and one Reconfigurable Module for each Reconfigurable Partition

➤ Partition Pins

– Ports on a Partition; Interface between Static and Reconfigurable Logic

🗲 XILINX ➤ ALL PROGRAMMABLE.

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 Modules

RP "A"

RP "B"

RP "C"

Static

A1 A2 A3

B1 B2

C1 C2 C3 C4

XILINX ➤ ALL PROGRAMMABLE.
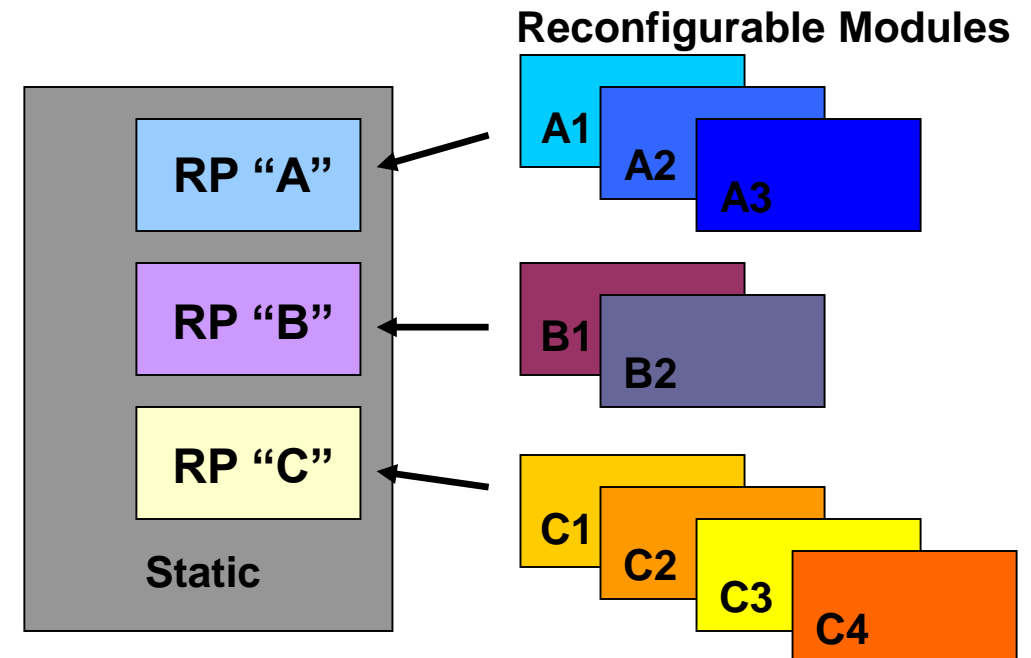
Reconfigurable Elements in 7 Series

➤ 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

➤ 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)

– I/O components (IOLOGIC, IODELAY, IDELAYCTRL)

🗲 XILINX ➤ ALL PROGRAMMABLE.

Reconfigurable Elements

❯ Granularity of reconfigurable regions vary by device family

– Boundaries recommended, but not required, to align to Clock Regions

– 7 Series and Zynq-7000

- Slice region: 50 CLB high by 1 CLB wide
- BRAM region: 10 RAMB36
- DSP region: 20 DSP48

– UltraScale / UltraScale+

- Slice region: 1 CLB high by 2 CLB wide
- BRAM region: 1 RAMB36 paired with 5 CLBs
- DSP region: 1 DSP48 paired with 5 CLBs
- GT region: 1 quad paired with one column of CLBs
- IO region: 1 bank, including MMCM and PLL resources, paired with one column of CLBs

❮ XILINX ❯ ALL PROGRAMMABLE.

Outline

❯ What is Partial Reconfiguration(PR)?

❯ PR Technology

❯ PR Terminology

❯ *PR Design Flow*

❯ Summary

ΣΣ XILINX ❯ ALL PROGRAMMABLE™

Intuitive Design Flow

Project Creation and Floorplanning

❯ Structure your design

 – Static Logic (unchanging design)
 – Reconfigurable Partitions (RP)
   • Instances to be reconfigured
 – Reconfigurable Modules (RM)
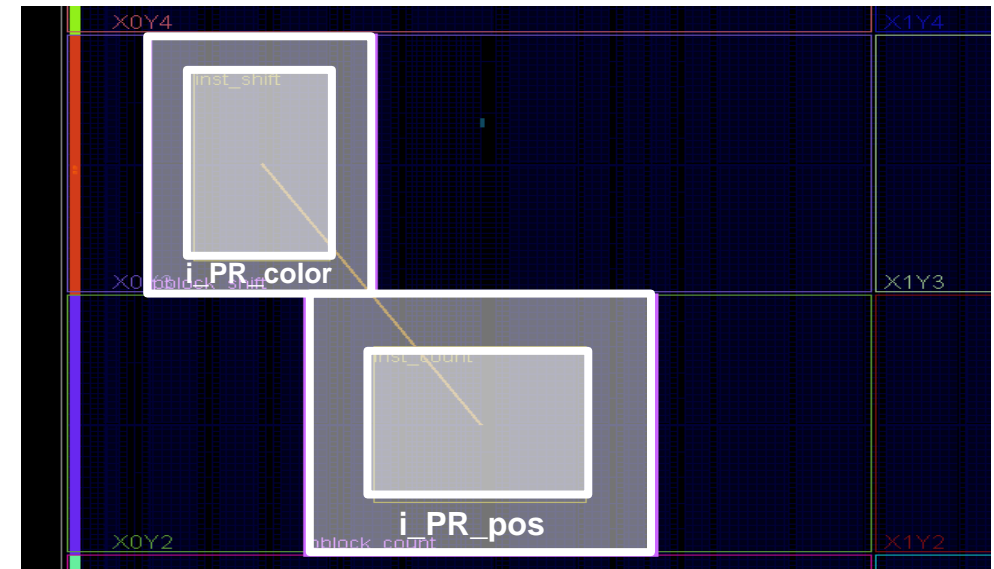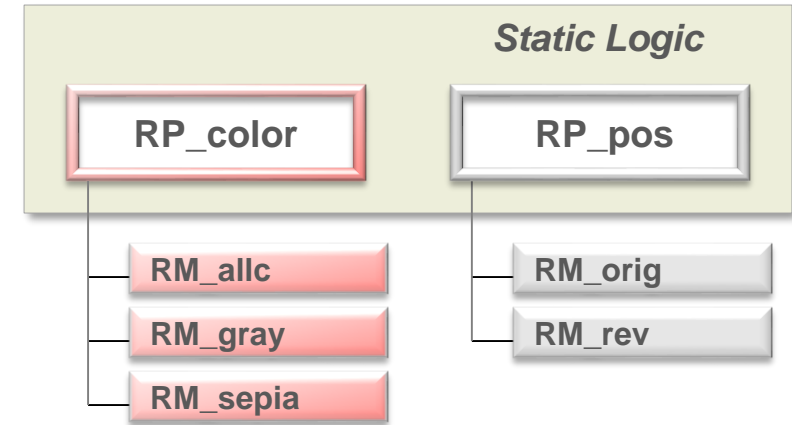   • Functional variations for each RP

❯ Synthesize bottom-up

 – `synth_design –mode out_of_context`

❯ Define resources to be reconfigured

 – Pblocks map design modules to physical regions
   • Define XY ranges and resource types

❯ Mark pblocks as reconfigurable

 – `HD.RECONFIGURABLE` initiates flow

© Copyright 2016 Xilinx

 XILINX ❯ ALL PROGRAMMABLE.

Leveraging Module Checkpoints for Partial Reconfiguration

- ❯ Partition methodology enables Partial Reconfiguration
  - Allows clear separation of static logic and Reconfigurable Modules
  - Floorplan to identify silicon resources to be reconfigured

- ❯ Design preservation accelerates design closure
  - Lock static design database while implementing new modules

© Copyright 2016 Xilinx

XILINX ❯ ALL PROGRAMMABLE.
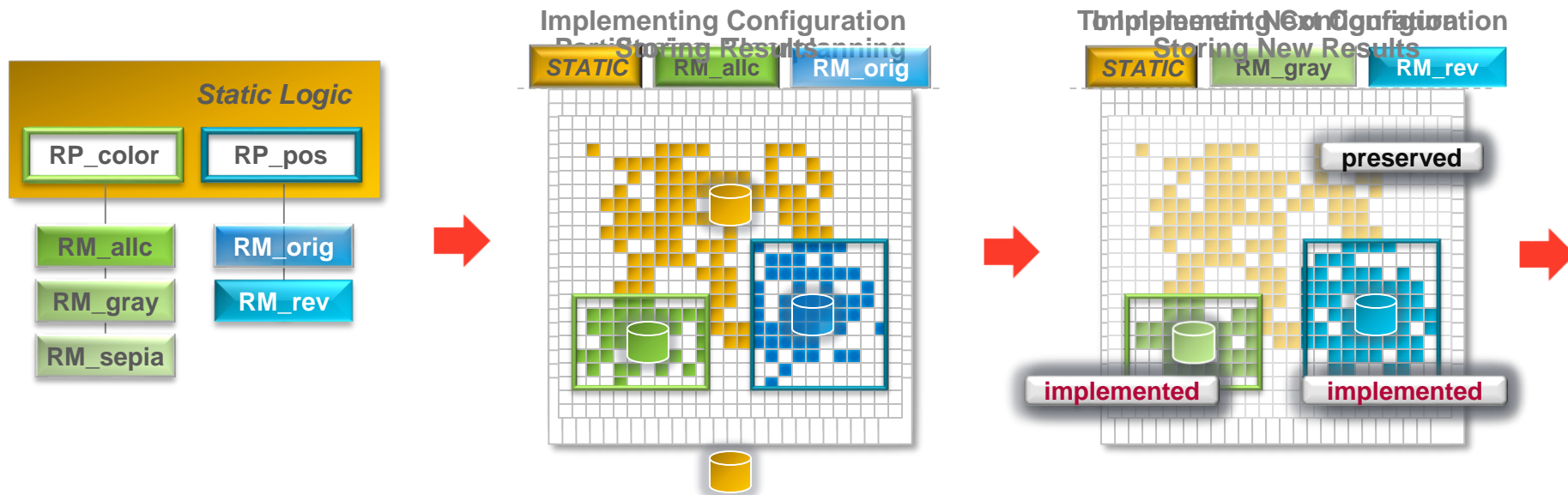
Vivado Software Features

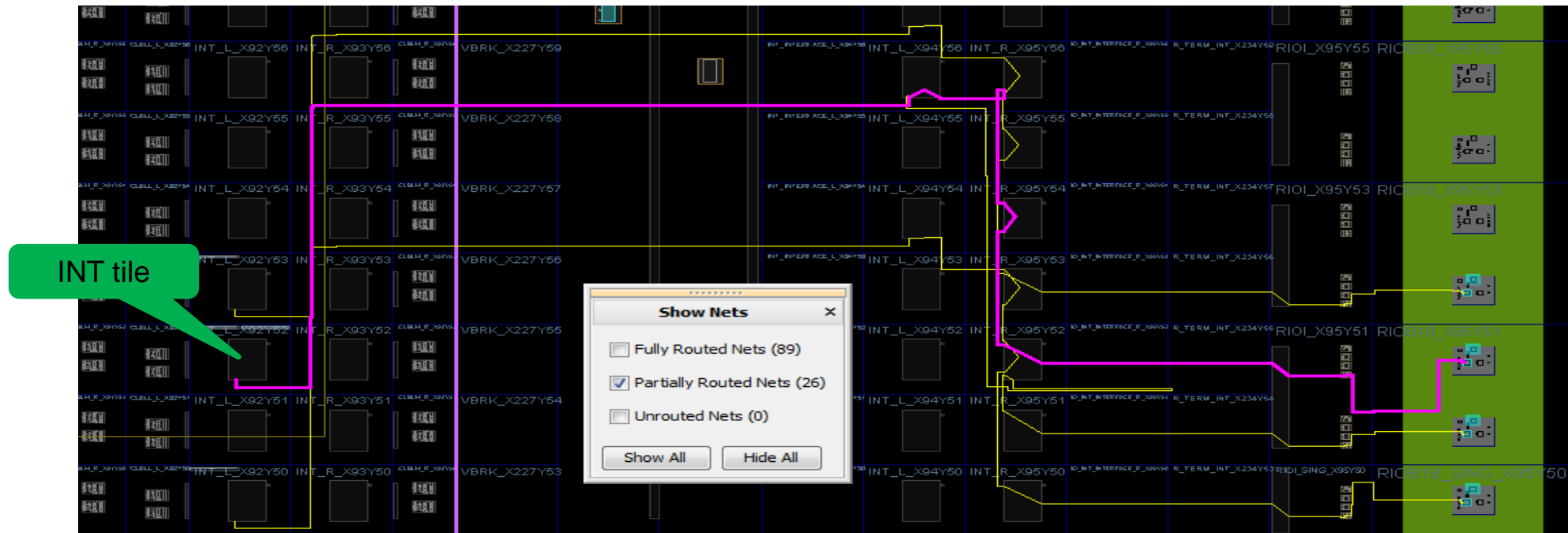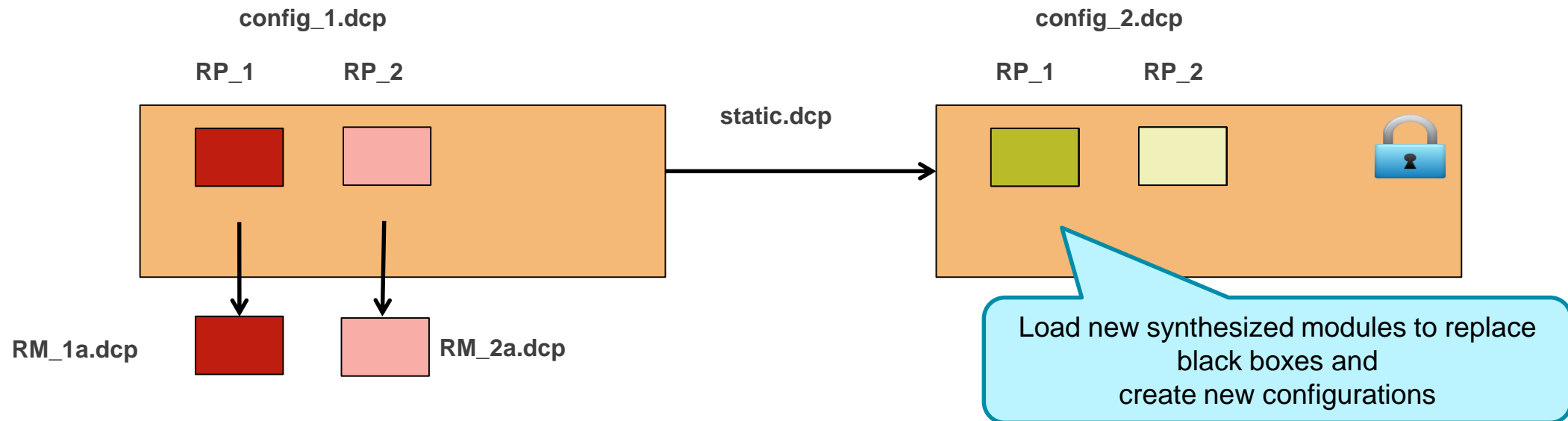*No Proxy Logic Required*

> Partition Pins are junction between static and reconfigured logic

- Interface wires can be broken at interconnect tile site
- "Anchor" between static and reconfigurable established mid-route
- No overhead at reconfigurable partition interface
- Decoupling logic still highly recommended

Vivado Software Design Management

Checkpoints for Each Partition

> ## Vivado stores design data in checkpoints

- Save full design as a configuration checkpoint for bitstream creation
- Save static-only checkpoint to be reused across multiple configurations
  - Routed static checkpoint can remain open in memory
  - Results are locked at the routing level
- Reconfigurable modules can also be stored as their own checkpoints

config_1.dcp

RP_1　　**RP_2**

static.dcp

config_2.dcp

RP_1　　**RP_2**

RM_1a.dcp　　　**RM_2a.dcp**

Load new synthesized modules to replace black boxes and create new configurations

XILINX ➤ ALL PROGRAMMABLE.

Intuitive Design Flow

Implementation

❯ Place and Route all design configurations

- Apply full design constraints in-context
- Use normal timing closure, simulation and verification techniques
- Use scripted non-project flow or new RTL-based project flow

❯ Final Verification

- Validates consistency of place and routed results across the entire system

❯ Generate Bitstreams

- write_bitstream automatically creates all full and partial bit files by default
- Selectively generate full bitstreams or specific partial bitstreams

❳ XILINX ❯ ALL PROGRAMMABLE.

Partial Reconfiguration Project Support

Initial release in Vivado 2016.3

❯ Support for RTL-based projects included in 2016.3

– IP within RMs, IP Integrator flow planned for 2017.1

❯ Flow included in documentation

– Solution details in UG909

– Tutorial lab in UG947

❯ Basic flow:

1. Define Reconfigurable Partitions
2. Populate Reconfigurable Modules
3. Create Configuration
4. Create Design Runs

– Tools manage sources, dependencies, run, PR Verify

XILINX ❯ ALL PROGRAMMABLE.
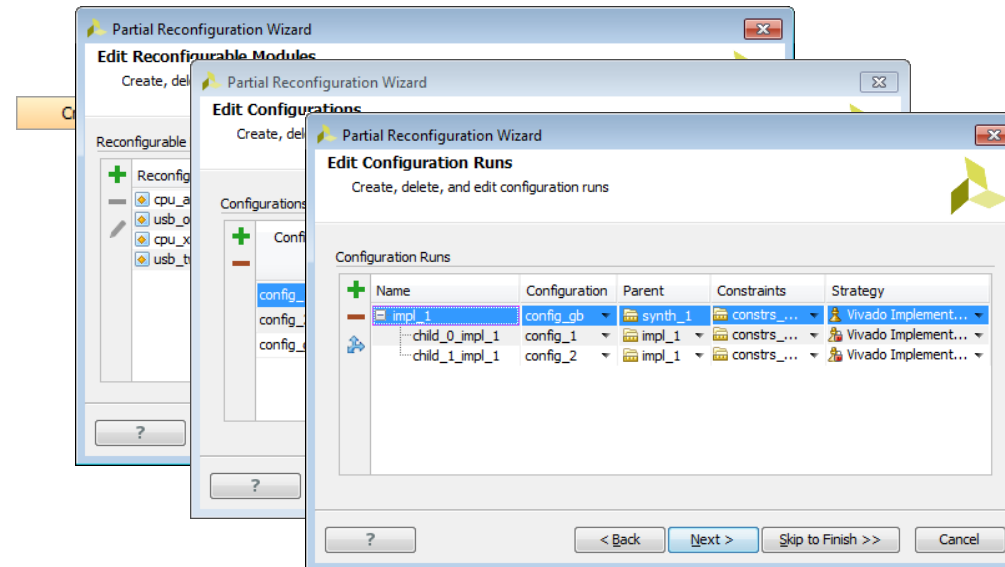
Configuration Details

➤ 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, optionally final and frame-based CRC value
  - No startup sequences or other overhead

➤ 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
  - Reported during bitstream generation

 XILINX ➤ ALL PROGRAMMABLE.

Partial Reconfiguration Collateral

▸ **Learn about Partial Reconfiguration**

– User Guide [UG909](#), Tutorial [UG947](#)

– [XAPP1231](#) shows Zynq solution

– [XAPP1261](#) shows PR + SEM

– PR Controller [IP page](#)

– PR Decoupler [IP page](#)

– PR Design Hub in DocNav

▸ **Training and Support**

– Training Course available via ATPs

– [QuickTake Video](#) reviews Vivado flow

• One for [UltraScale](#) features as well

– [XUP training](#) for Zynq flows

– Two Lunch & Learn modules available

Partial Reconfiguration in Vivado
V2016.3 - Published 2016-10-27

⊟ Getting Started

Introduction
- Partial Reconfiguration Home Page
- Vivado Design Suite User Guide: Partial Reconfiguration
- Vivado Design Suite Tutorial: Partial Reconfiguration
- Partial Reconfiguration for UltraScale
- Partial Reconfiguration in Vivado (7 Series)

Key Concepts
- What Does Partial Reconfiguration Software Flow Look Like?
- How Do I Program the Full and Partial BIT files?
- What Are the Key Design Considerations for Partial Reconfiguration with 7 Series Devices?
- What Are the Key Design Considerations for Partial Reconfiguration with UltraScale Devices?
- How Do I Floorplan My Reconfigurable Modules?
- When Do I Need to Use a Clearing BIT file for UltraScale Devices?

Frequently Asked Questions
- How Do I Obtain a License for Partial Reconfiguration?
- How Do I Use the SNAPPING_MODE Property for Partial Reconfiguration?
- How Do I Load a Bitstream Across the PCI Express Link in UltraScale Devices for Tandem PCIe and Partial Reconfiguration
- How Do I Manually Control the Placement of the PartPins in Partial Reconfiguration Flow?
- How Do I Debug Partial Reconfiguration Designs?
- How Do I Update BRAM with ELF file for Partial Reconfiguration when MicroBlaze is Inside of the Reconfigurable Module?

⊟ Partial Reconfiguration Resources

Partial Reconfiguration IP
- Partial Reconfiguration Controller Product Page
- Partial Reconfiguration Decoupler Product Page

Application Notes
- Loading Partial Bitstreams using TFTP
- Partial Reconfiguration of a Hardware Accelerator with Vivado

XILINX ➤ ALL PROGRAMMABLE™

Outline

> **What is Partial Reconfiguration(PR)?**

> **PR Technology**

> **PR Terminology**

> **PR Design Flow**

> ***Summary***

XILINX ➤ ALL PROGRAMMABLE™

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

ΣXILINX ❯ ALL PROGRAMMABLE.