

Ph.D. Defense
December 2nd, 2015

Christophe HURIAUX

Enhanced FPGA Architecture and CAD Flow for Efficient Runtime Hardware Reconfiguration

*Architecture FPGA améliorée et flot de conception
pour une reconfiguration matérielle en ligne efficace*



Introduction

- High demand for computing architectures
- Design choices
 - Computing performance
 - Low development costs
 - Flexibility
 - Energy efficiency



Towards heterogeneous manycores

- Performance constraints lead to multicore
- The end of Dennard's scaling: utilization wall

Conventional scaling | Leakage-limited scaling^[Ven+10]

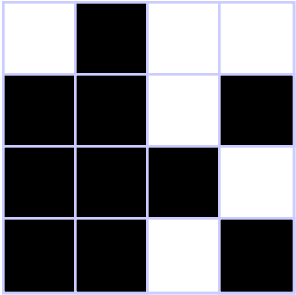
Device count	S^2	S^2
Device frequency	S	S
Capacitance	$1/S$	$1/S$
Device power	$1/S^2$	1
Utilization	1	$1/S^2$

Towards heterogeneous manycores

- Performance constraints lead to multicore
- The end of Dennard's scaling: utilization wall

	Conventional scaling	Leakage-limited scaling ^[Ven+10]
Device count	S^2	S^2
Device frequency	S	S
Capacitance	$1/S$	$1/S$
Device power	$1/S^2$	1
Utilization	1	$1/S^2$

→

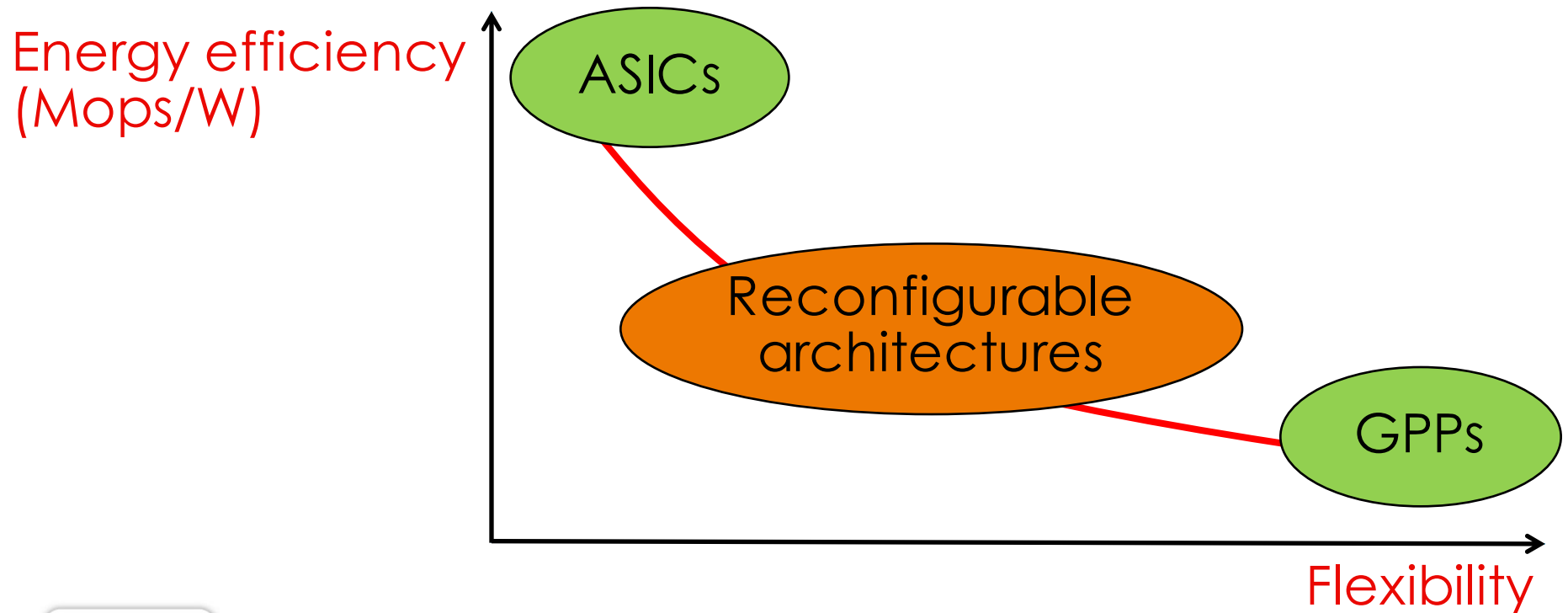


All cores of the chip cannot work at the same time

- Adding hardware accelerators in manycore architectures improves power efficiency and circuit utilization

Hardware accelerators

- At fixed power budget, reconfigurable architectures offer a trade-off between performance and flexibility



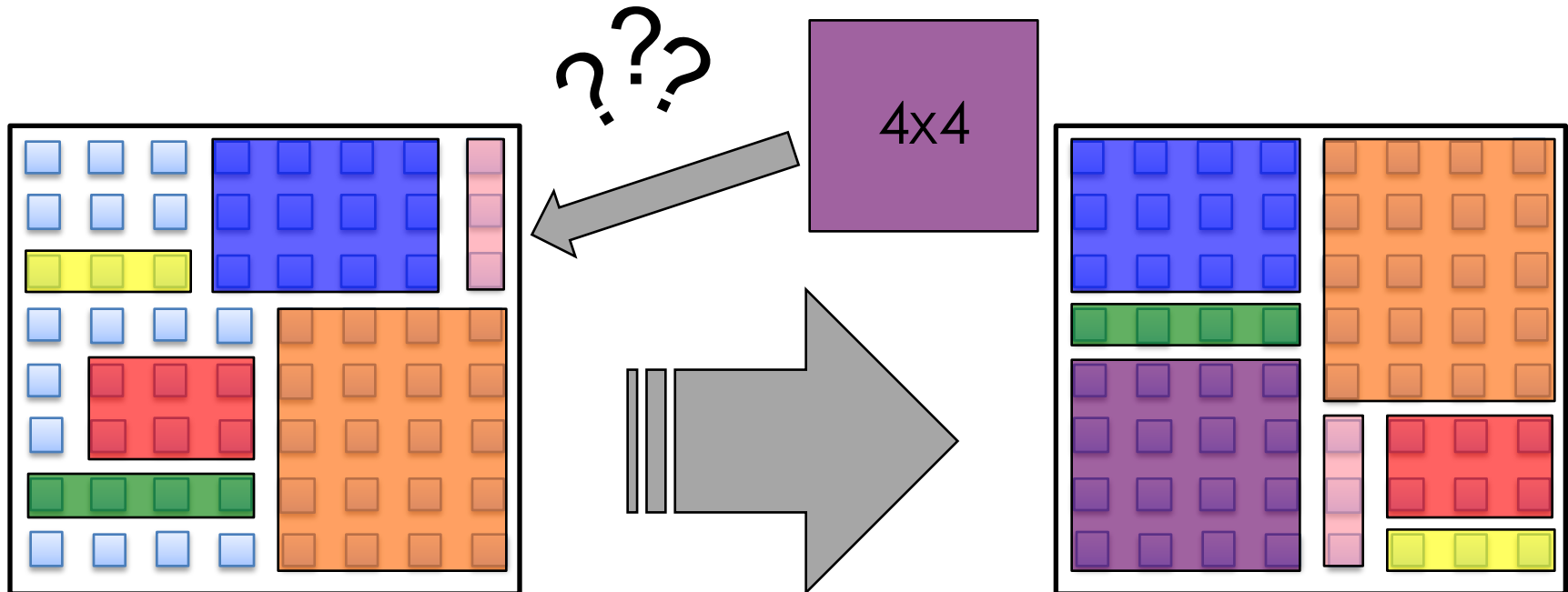
FPGAs to speed up software

- Field-Programmable Gate Arrays (FPGAs) dominate the reconfigurable hardware market
- Recent interest of major actors
 - Intel to buy Altera and announced FPGA-accelerated Xeon processors by 2016
 - Microsoft speed up Bing's search requests with FPGAs [Put+14]
- *« They are actually going to push FPGAs into their data centers, [...] to keep them there, we're going to have to really improve the future of programmability »*

- Andrew Putnam, Microsoft (2014)

Limitations of FPGA acceleration

- FPGAs can parallelize multiple tasks...
- ... but do not offer flexibility for task placement or migration [Com+02]



Challenges and goals

- Position-independent hardware accelerators in FPGAs
 - **Simple** online decoding algorithms
 - **No predefined** configuration domains
- Cope with the **heterogeneity**
 - Resource sharing/distribution easiness
 - How to move a task across the logic fabric?
- Dedicated Computer-Aided Design (CAD) flow
- Enclose the task target region
 - Allows the system to pursue its operations during a reconfiguration

Outline

- Context
 - FlexTiles project
 - State of the art on hardware relocation
- Contributions
 - Position-independent tasks: Virtual Bit-Streams
 - Routing architecture enhancements
- Conclusions & Perspectives

The FlexTiles FP7 project



- FlexTiles: *Self adaptive heterogeneous manycore based on Flexible Tiles*
- Provide a **heterogeneous manycore** architecture offering
 - Large flexibility
 - High-performance, energy efficiency
 - Raised programming efficiency
 - Self-adaptation through virtualization

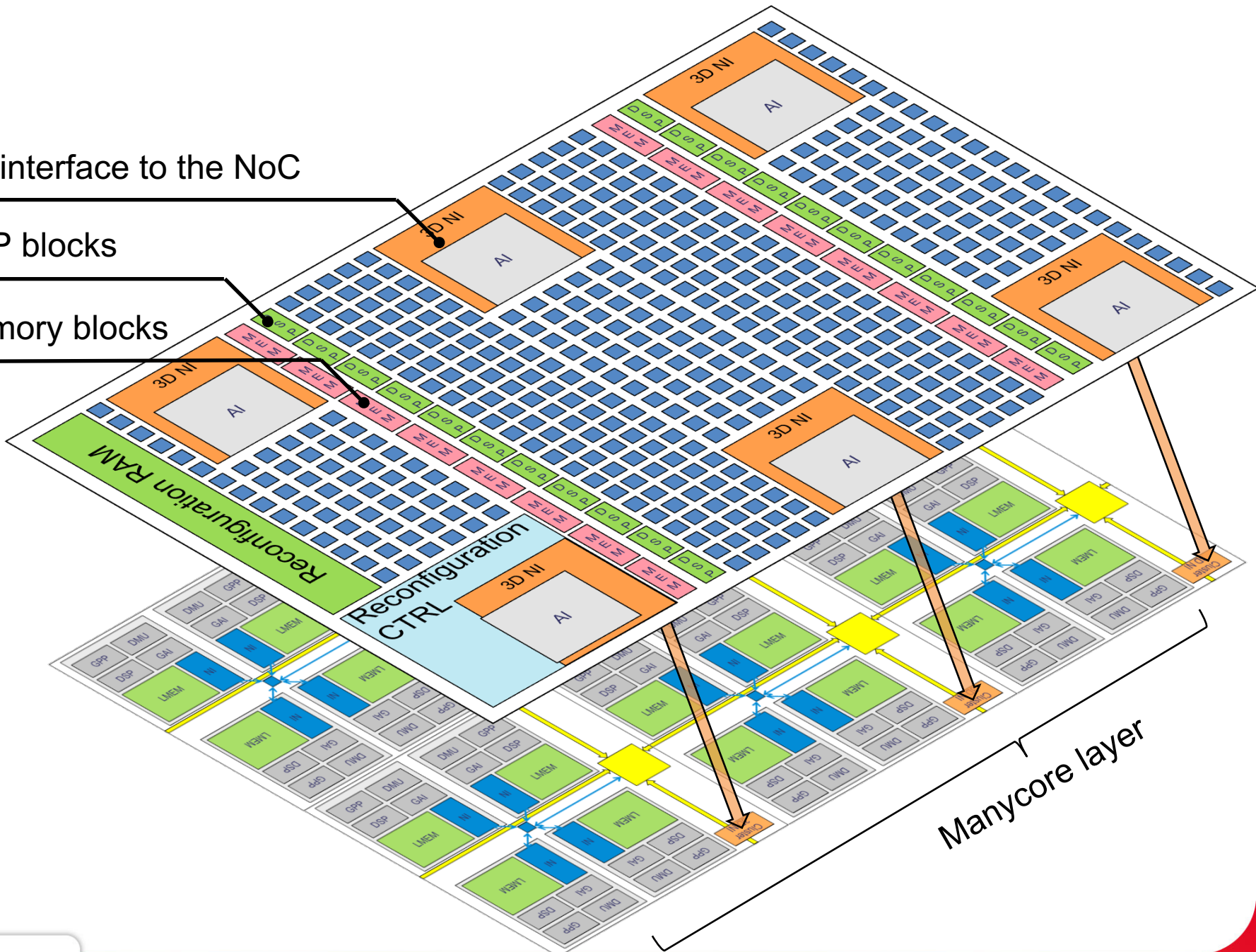
FlexTiles goals

- **3D-Stacked** Heterogeneous manycore
 - General Purpose Processors (GPP)
 - for flexibility and programming homogeneity
 - **Dedicated hardware accelerators mapped at run-time on a reconfigurable layer**
 - Network On Chip
- Reconfigurable FPGA layer with seamless **task migration** capabilities
- Virtualization layer to provide an abstraction of the manycore and self adaptive services
- Tool-chain for parallelization and compilation

3D interface to the NoC

DSP blocks

Memory blocks



Manycore layer

Outline

- Context
 - FlexTiles project
 - State of the art on hardware relocation
- Contributions
 - Routing architecture enhancements
 - Position-independent tasks: Virtual Bit-Streams
- Conclusions & Perspectives

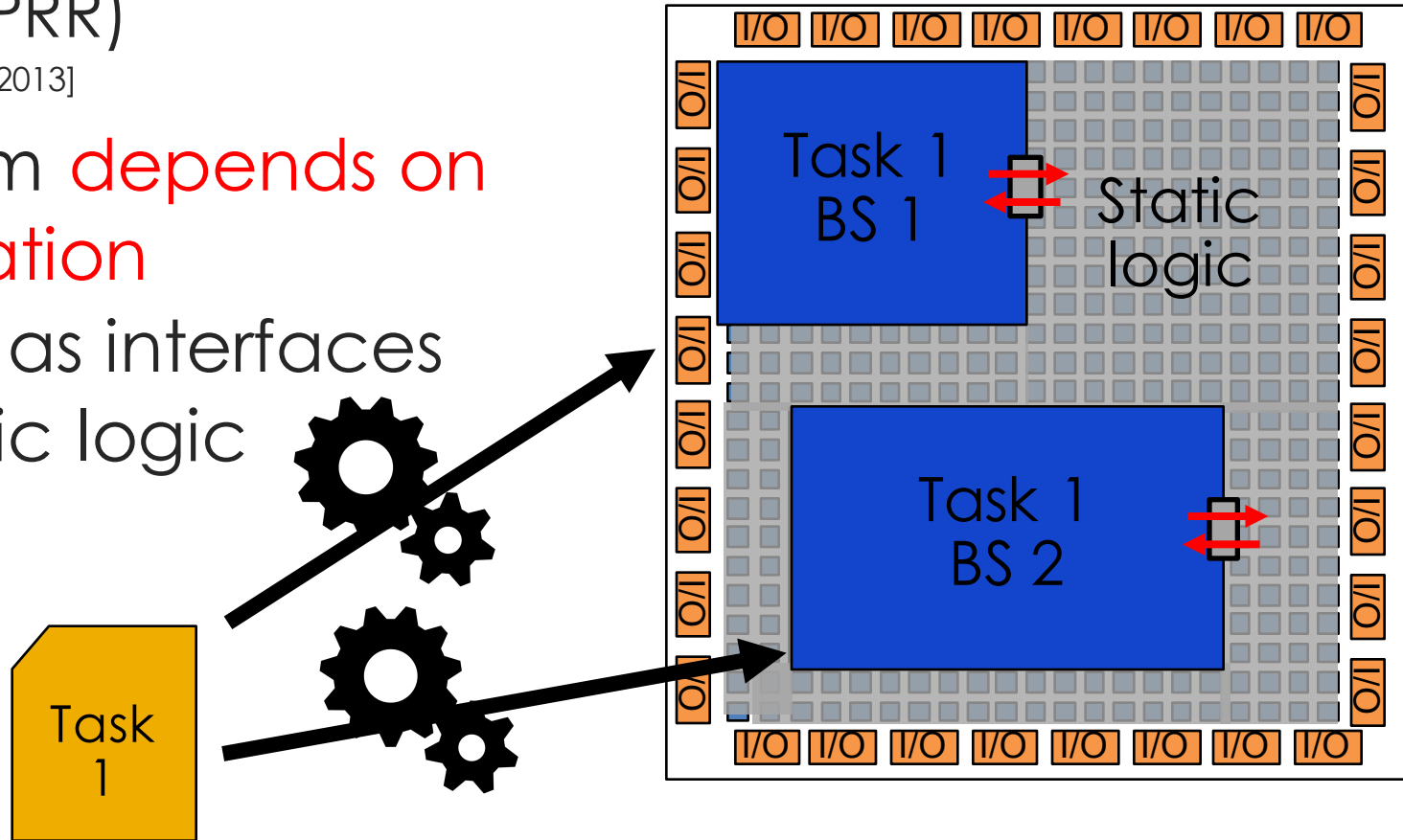
FPGA vendors CAD flows

- **Predefined** reconfigurable regions (PRR)

[Altera2010][Xilinx2013]

- Bit-stream **depends on task location**

- Use LUTs as interfaces with static logic



Research on relocation

- Main relocation techniques



- One bit-stream per PRR
- Compression of similar bit-streams
- Online rewriting (*filters*)
- Bit-stream merging
- Just-in-time routing

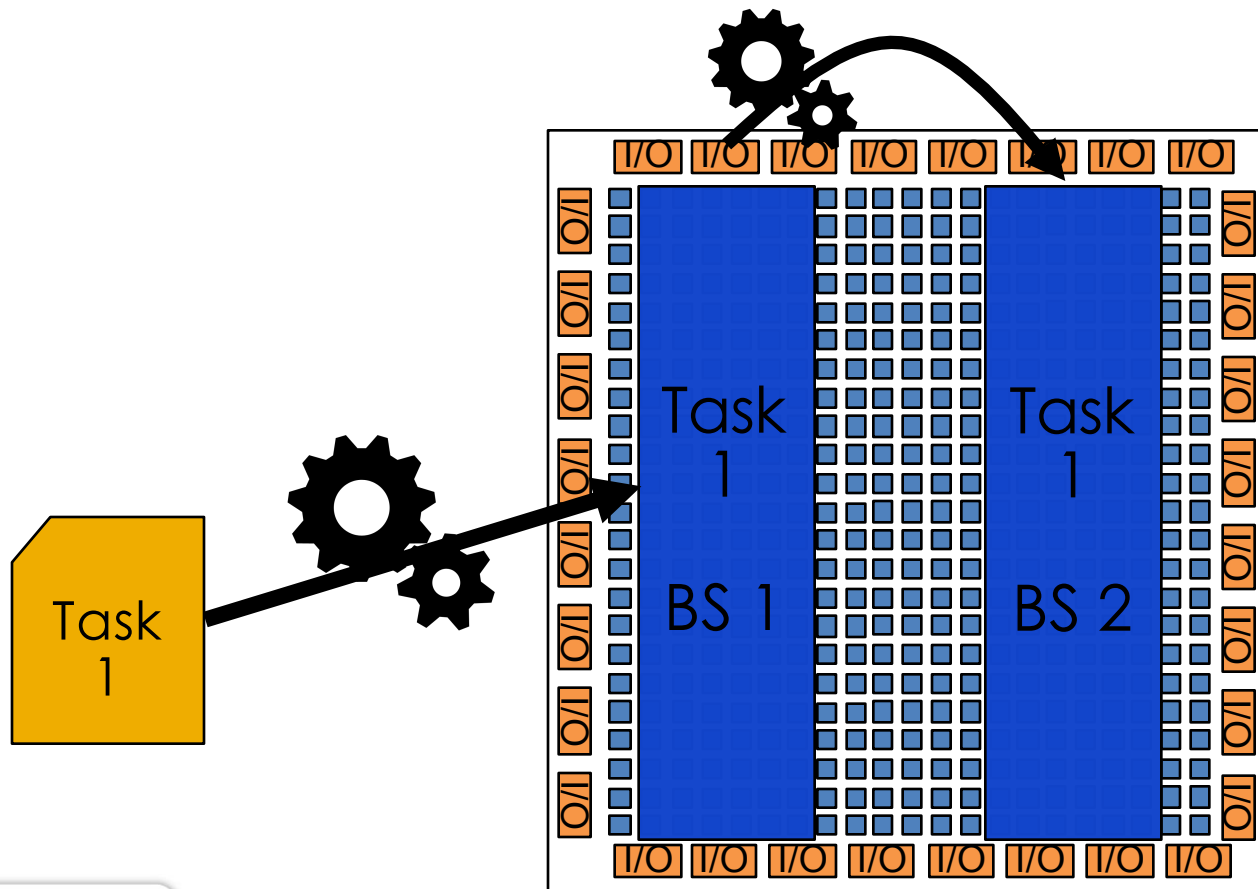


Online complexity

Flexibility

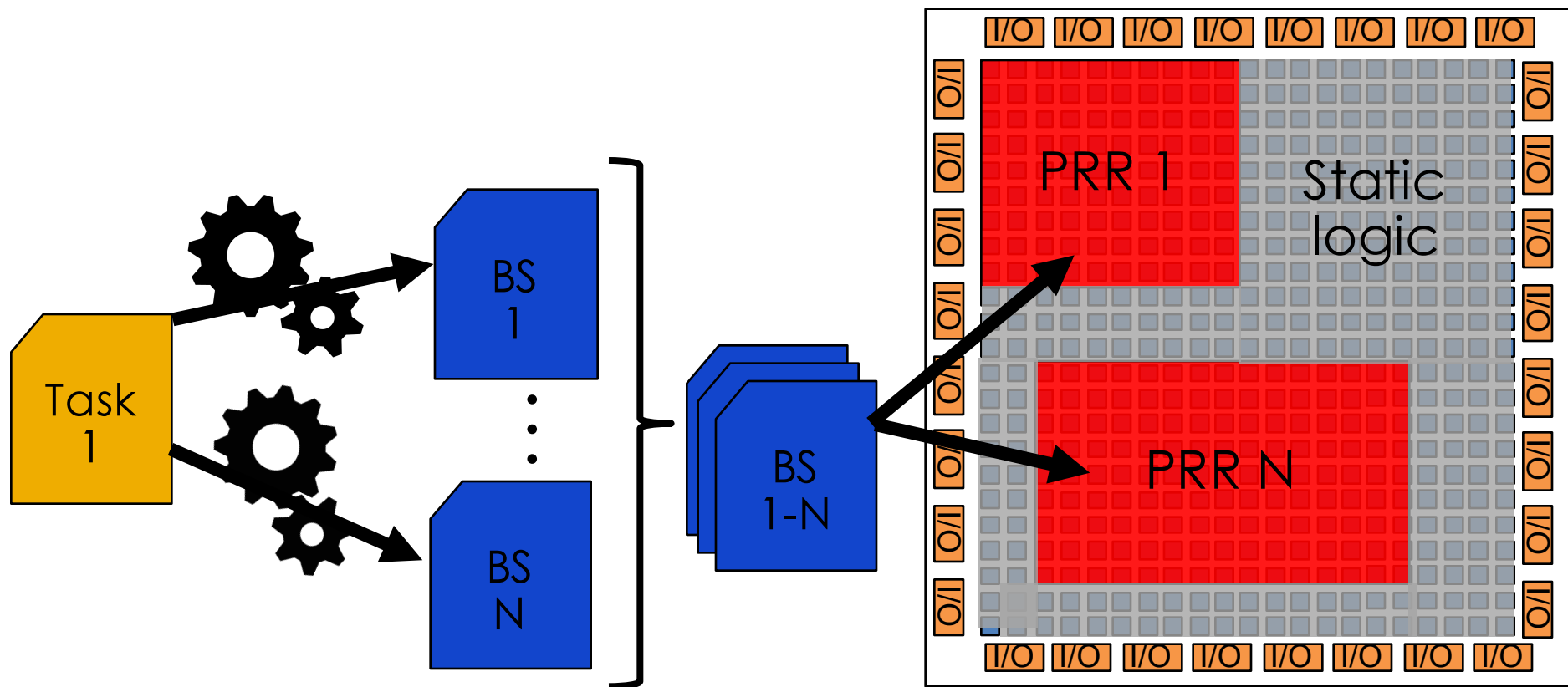
Online rewriting

- PARBIT [HL+01], Replica/Replica2 [Kal+05], BiRF [Cor+07]
 - Limited flexibility to identical columns



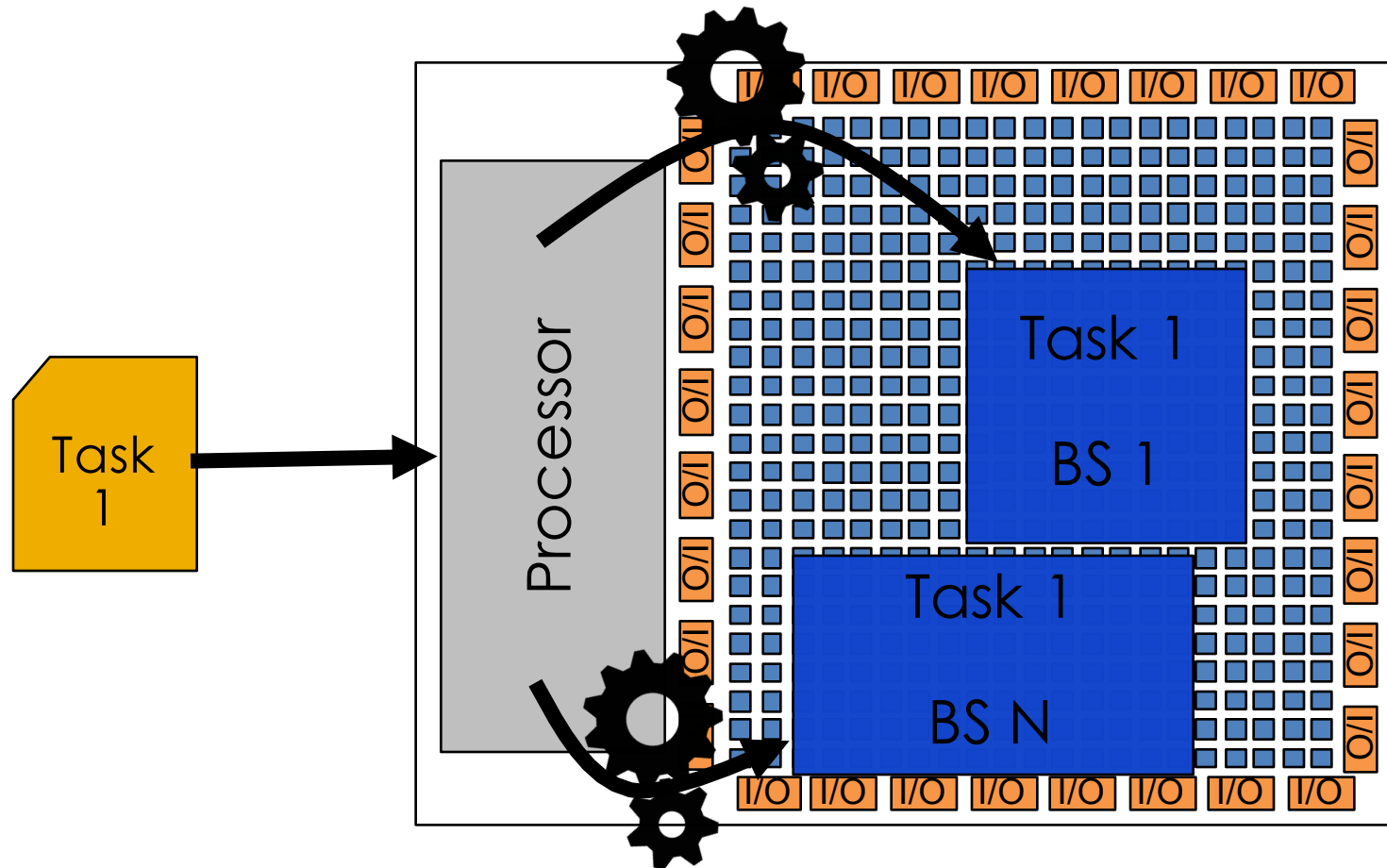
Bit-stream compression

- OORBIT [Tou+12], GoAhead [BKT12] [BKT14]
 - Memory consuming



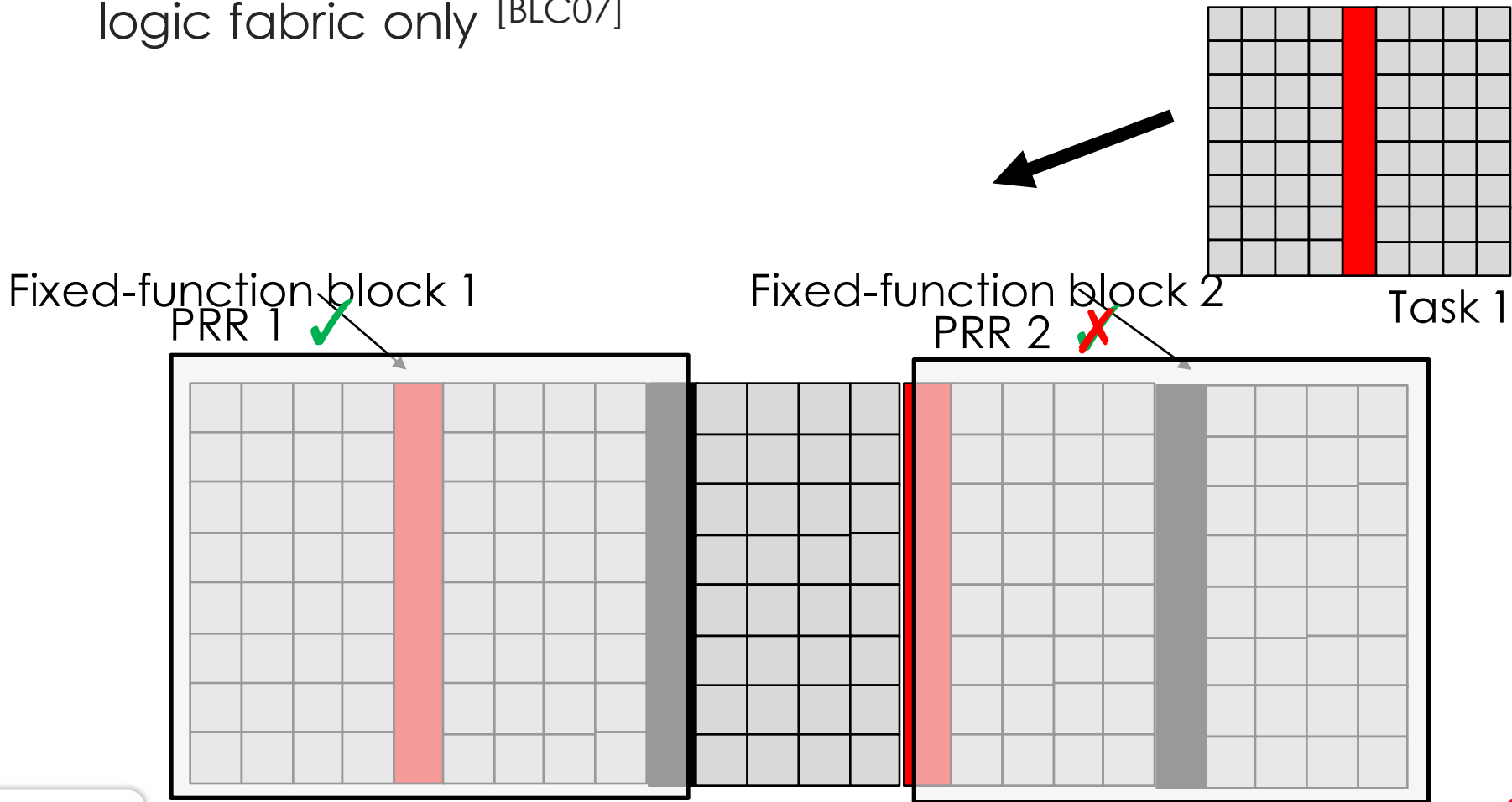
Just-in-Time place-and-route

- Riverside On-Chip Router (ROCR) [LVT04], [SF10]
 - Time (up to 13s) and memory consuming



Logic fabric heterogeneity

- No work on relocation of **heterogeneous** tasks
 - Relocation of **homogeneous** resources on a **heterogeneous** logic fabric only [BLC07]



Contributions

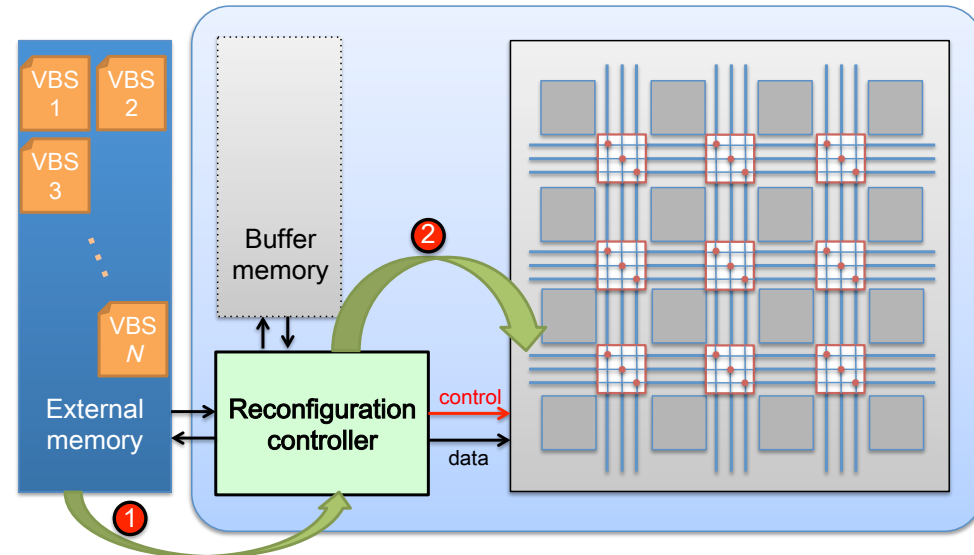
- **Virtual Bit-Streams (VBS)**: globally **pre-routed**, position **independent** task bit-streams
- CAD flow for the **generation** of VBS
- Online controller for real-time **decoding** of VBS
- Enhanced **routing architecture** to relocate tasks on a **heterogeneous** logic fabric
- **Flexible** configuration memory

Outline

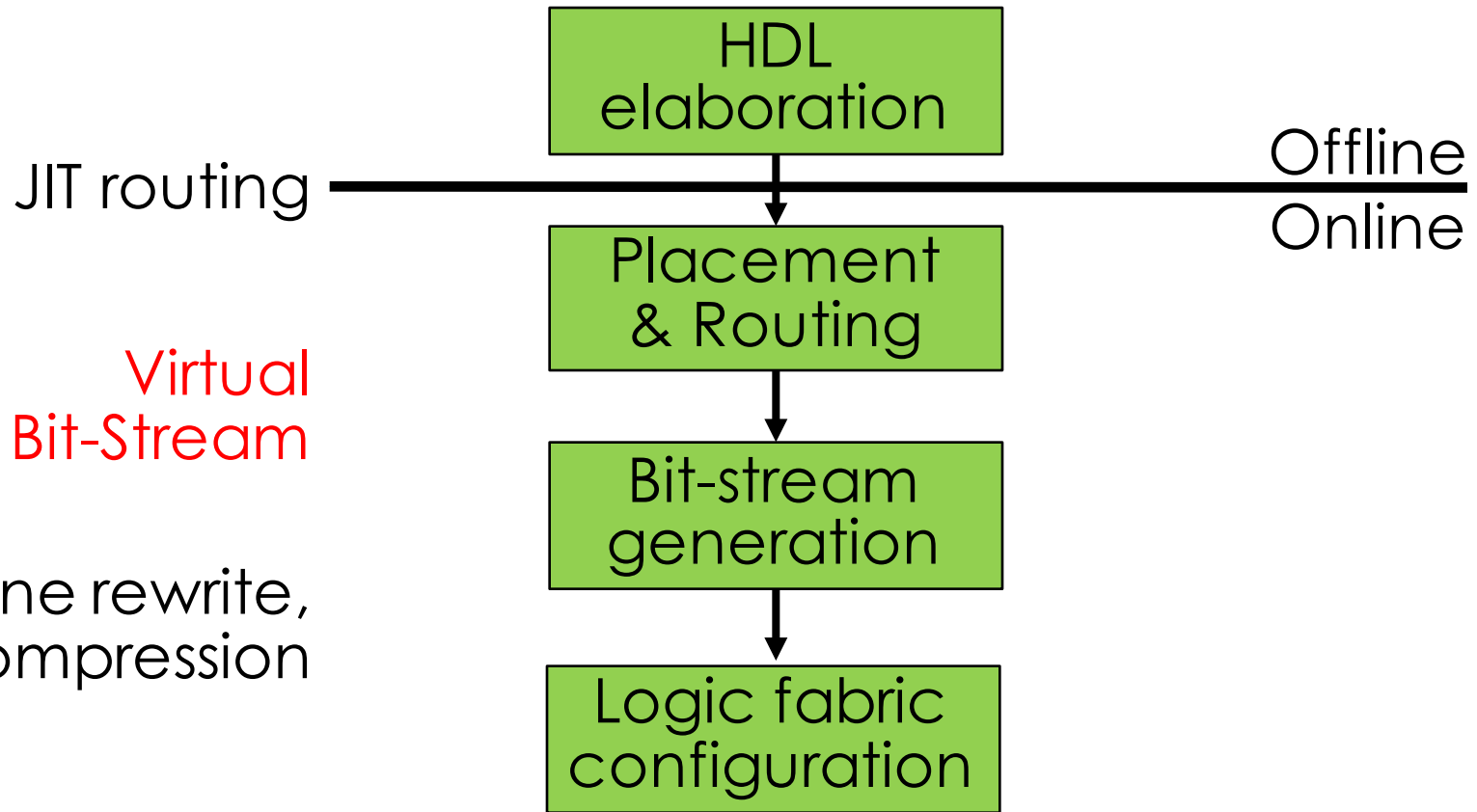
- Context
 - FlexTiles project
 - State of the Art on hardware relocation
 - Challenges
- Contributions
 - Position-independent tasks: Virtual Bit-Streams
 - Principles
 - CAD Flow
 - Online decoding
 - Routing architecture enhancements
- Conclusions & Perspectives

Virtual Bit-Stream

- A task is synthesized, placed & routed into a *Virtual Bit-Stream* (VBS)
 - Independent from task physical location in the fabric
 - No predefined configuration domains
- A reconfiguration controller generates final BS at *run-time*

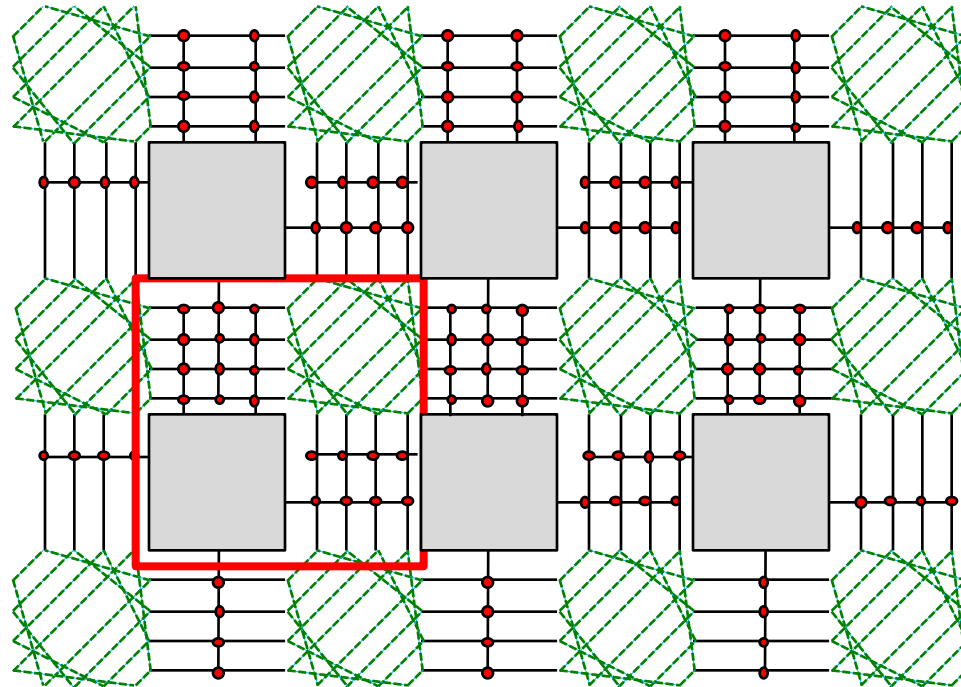


Offline/online distribution



Virtual Bit-Stream representation

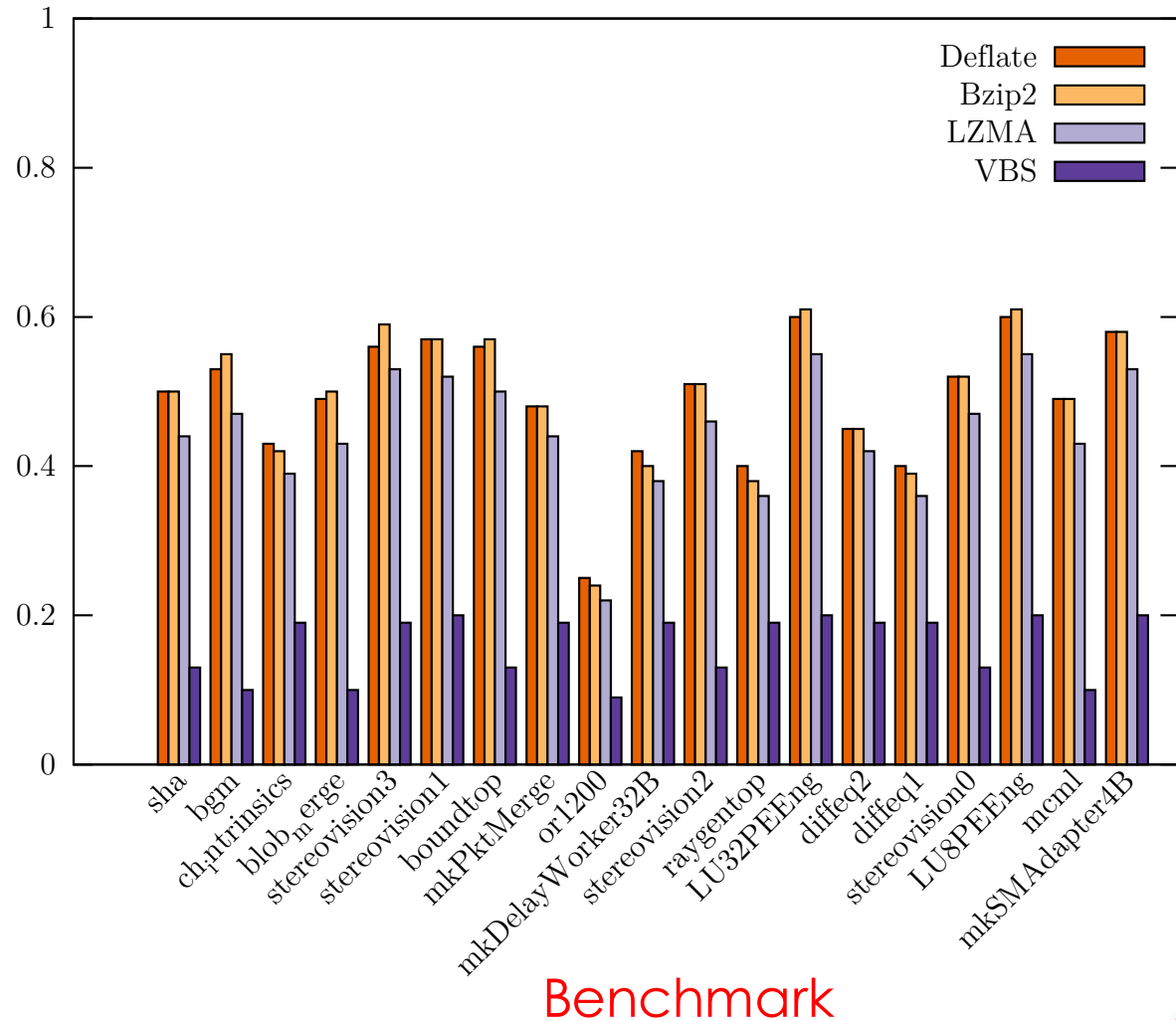
- Island-style FPGA
 - Logic grid
 - Mesh routing lines
 - Switch boxes
 - Interconnect
- The VBS encodes each island (macro-cell) separately



Virtual Bit-Stream compression

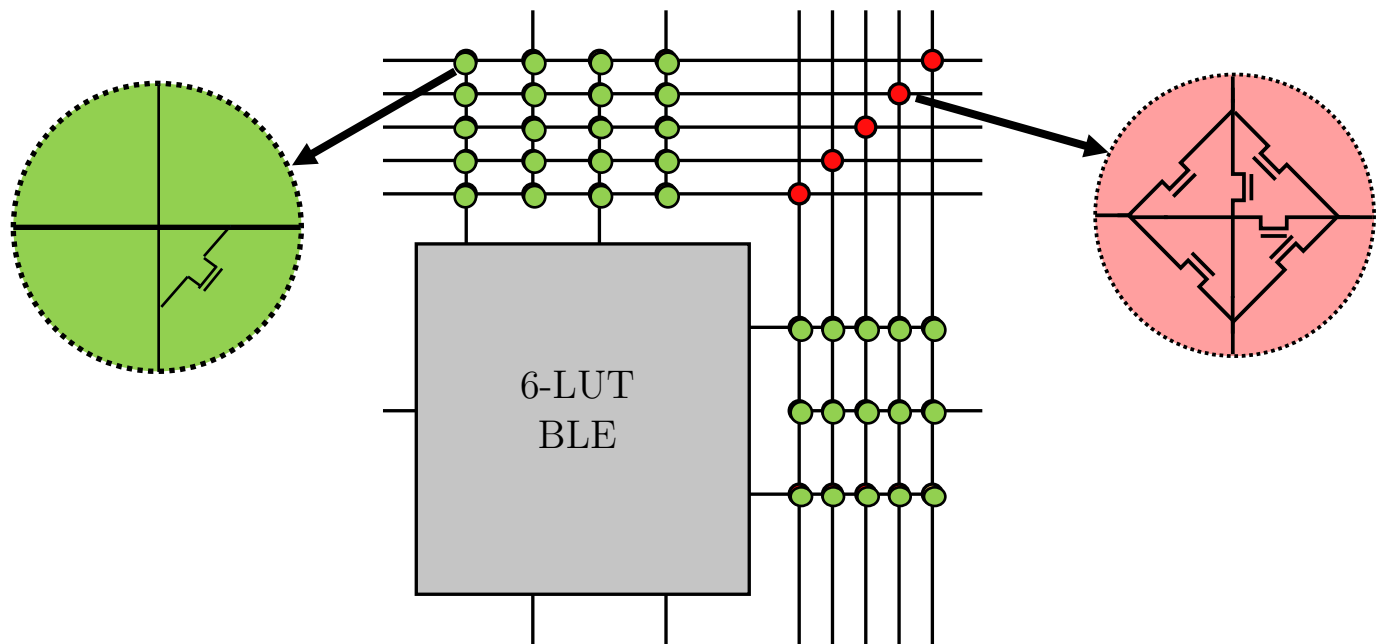
Compression ratio

- The VBS is an **abstract** representation of the interconnect
- **Compression** can be a side-effect of the representation on some architectures



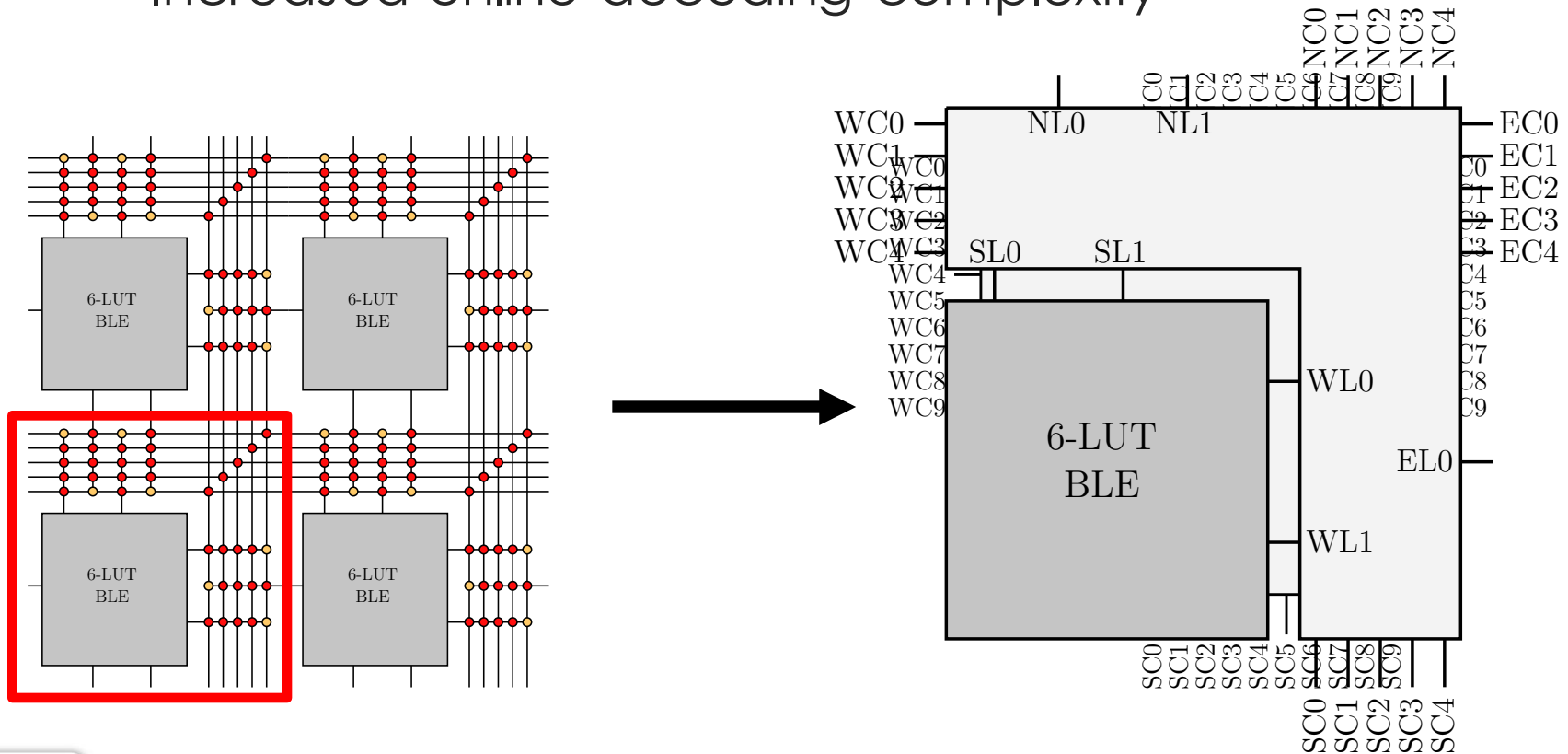
Virtual Bit-Stream representation

- Modern architectures need much **less configuration states**
 - Significantly lower compression ratio
 - Easier online routing
- May lead to **other representations** of the Virtual Bit-Streams



Virtual Bit-Stream representation

- **Clustering** multiple islands together can improve the abstract representation
 - Increased online decoding complexity



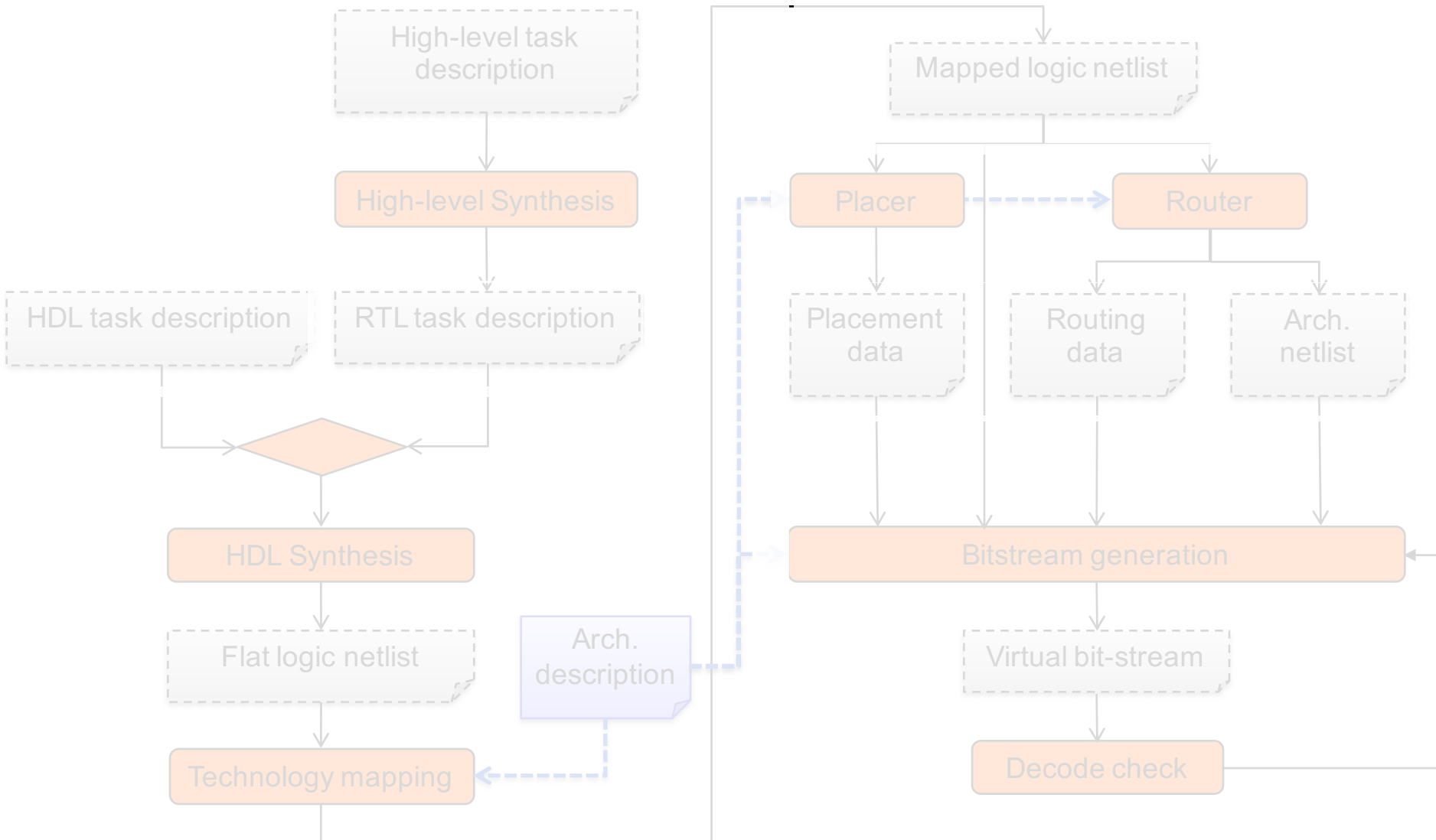
Outline

- Context
 - FlexTiles project
 - State of the Art on hardware relocation
 - Challenges
- Contributions
 - Position-independent tasks: Virtual Bit-Streams
 - Principles
 - CAD Flow
 - Online decoding
 - Routing architecture enhancements
- Conclusions & Perspectives

VBS: CAD Flow

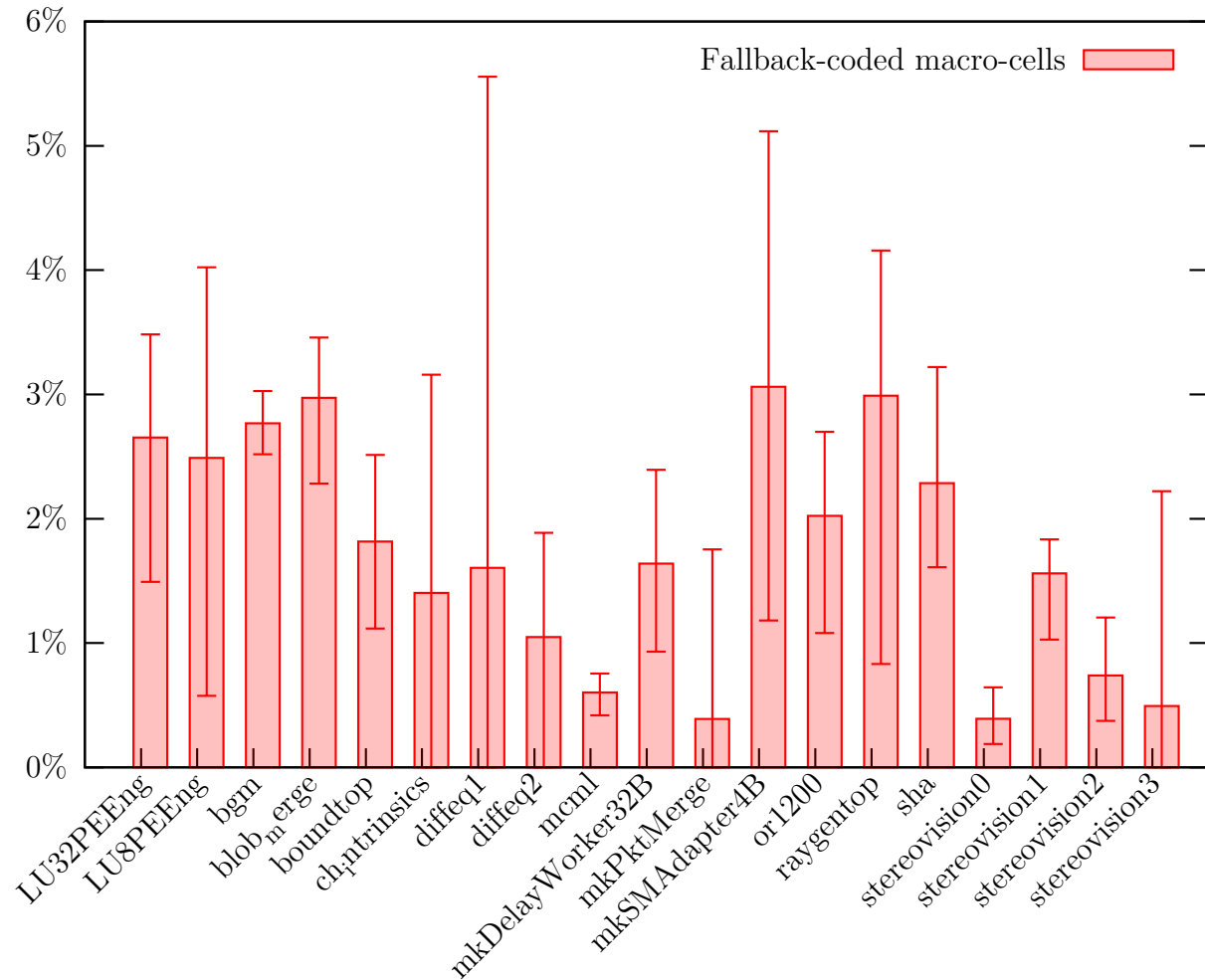
- Based on the Verilog-To-Routing (VTR) framework
 - Allows to describe **any FPGA architecture** and perform place and route operations
- Uses Versatile Place and Route
 - Widely used for academic FPGA architecture research
- A **custom backend** reads the placement and routing data to generate Virtual Bit-Streams

VBS: CAD Flow



VBS: CAD Flow

Raw-coded rate



Benchmark

- Generated VBS are checked offline, parts failing to decode are raw-coded
- The **encoding** effort is spent **offline**

Outline

- Context
 - FlexTiles project
 - State of the Art on hardware relocation
 - Challenges
- Contributions
 - Position-independent tasks: Virtual Bit-Streams
 - Principles
 - CAD Flow
 - Online decoding
 - Routing architecture enhancements
- Conclusions & Perspectives

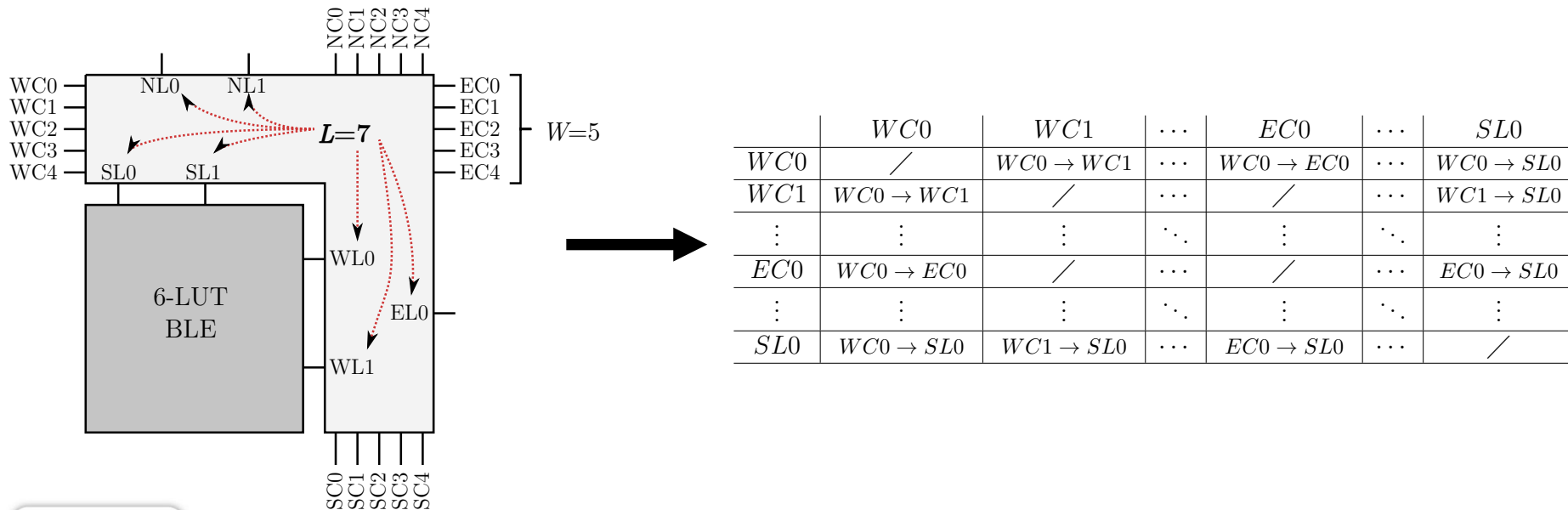
VBS: Online Decoding

- A dedicated controller is responsible for the **decoding** of **Virtual Bit-Streams** into **configuration bit-streams**
- The offline generation flow of the VBS ensures the feasibility of the decoding



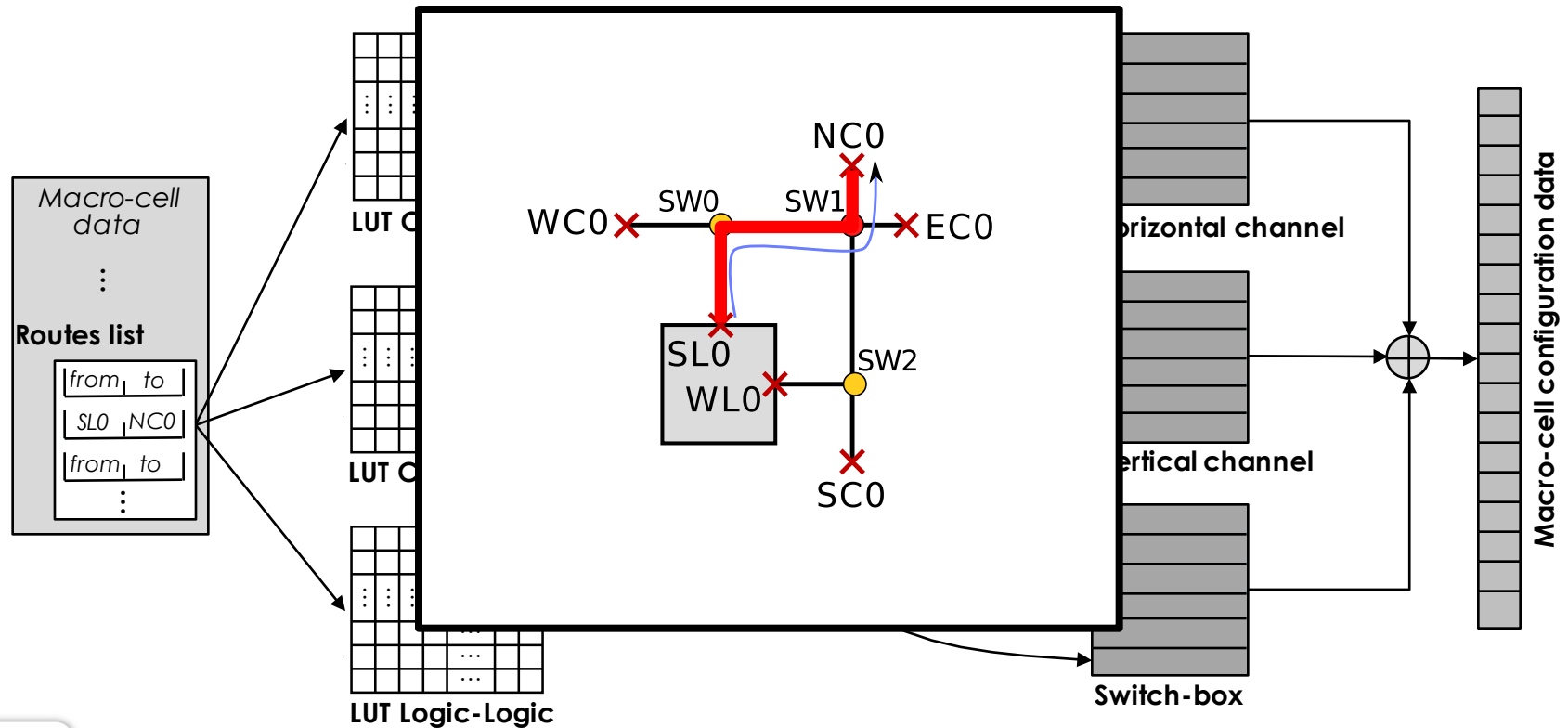
VBS: Online Decoding

- Two algorithms proposed
- Both rely on a **fixed** set of point-to-point routes between I/Os of macro-cells
- The **interconnection matrix** depends on the target interconnect architecture



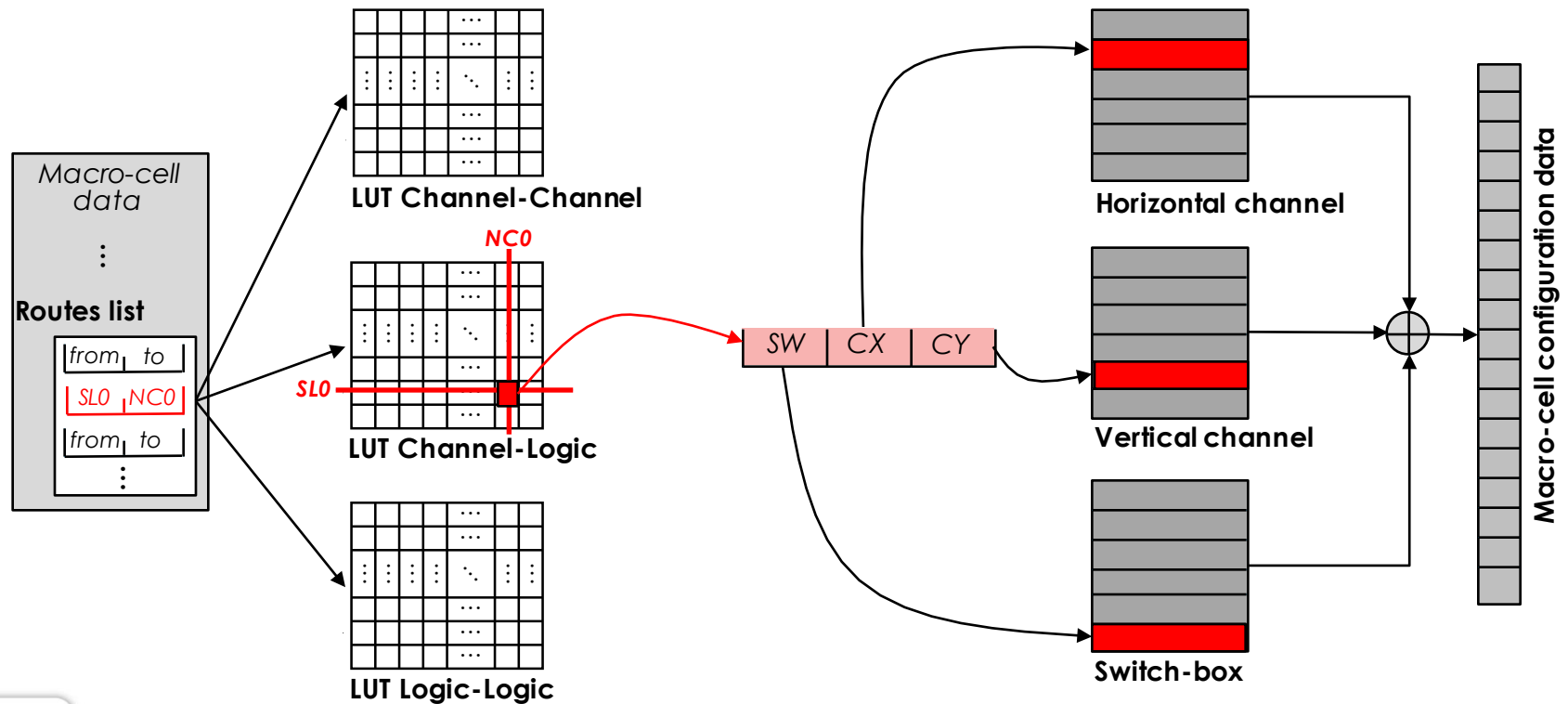
VBS: Online Decoding

- Configuration data may be built from a **Look-Up Table**
 - The memory complexity depends on N_{IO}^2 !



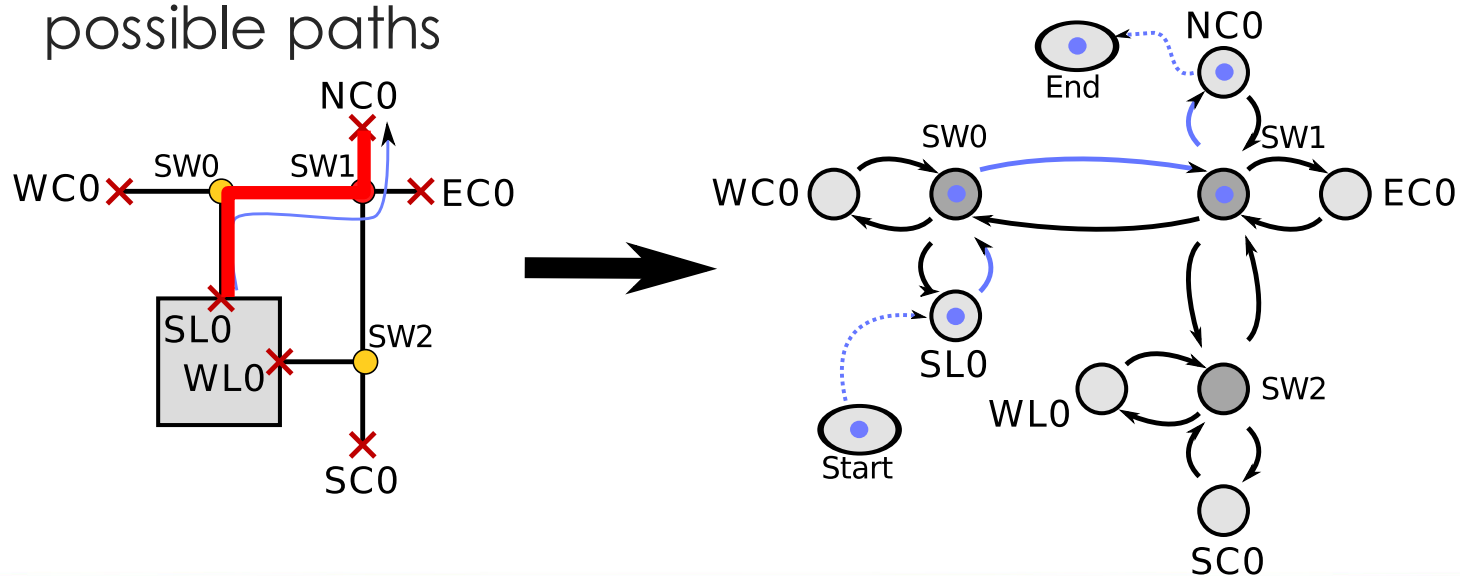
VBS: Online Decoding

- Configuration data may be built from a **Look-Up Table**
 - The memory complexity depends on N_{IO}^2 !



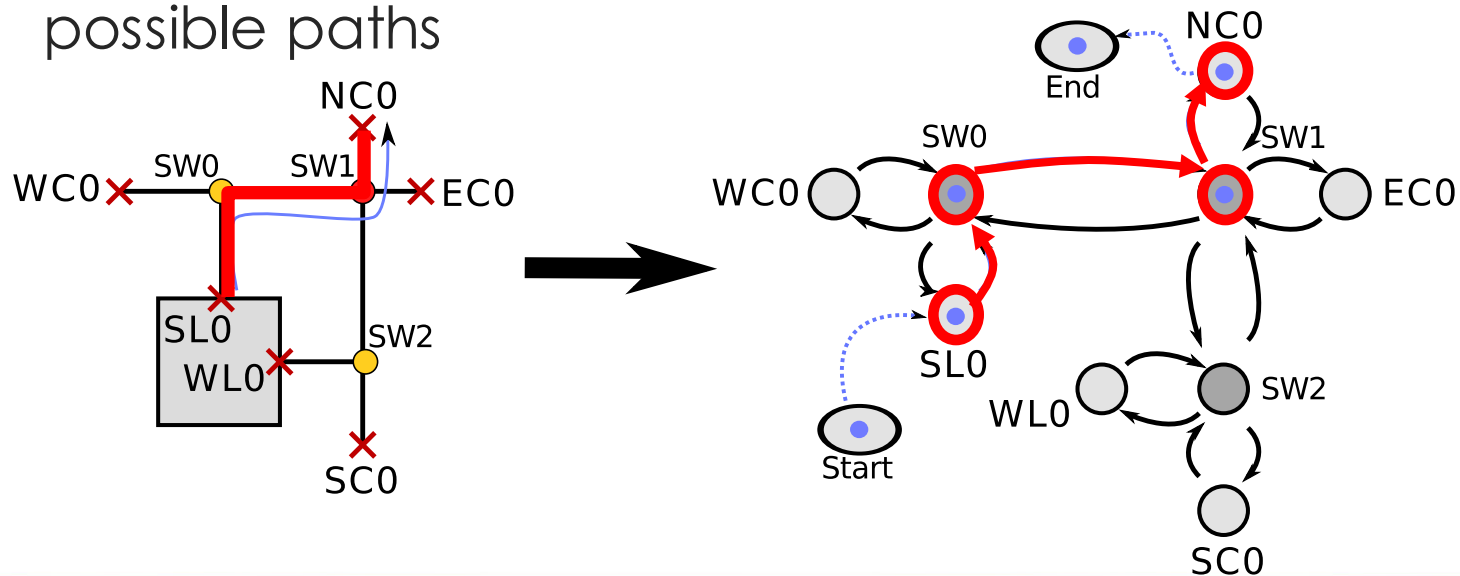
VBS: Online Decoding

- A **Finite State Machine** can recover the configuration data
 - Decoding time depends on the number of switches crossed
- Similar to the classical routing problem
 - The VBS decoding is **constrained** to a **limited** set of possible paths



VBS: Online Decoding

- A **Finite State Machine** can recover the configuration data
 - Decoding time depends on the number of switches crossed
- Similar to the classical routing problem
 - The VBS decoding is **constrained** to a **limited** set of possible paths



Virtual Bit-Stream overview

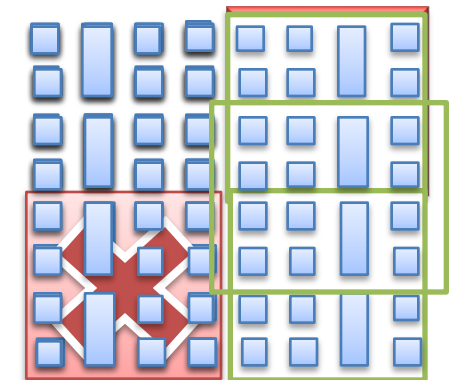
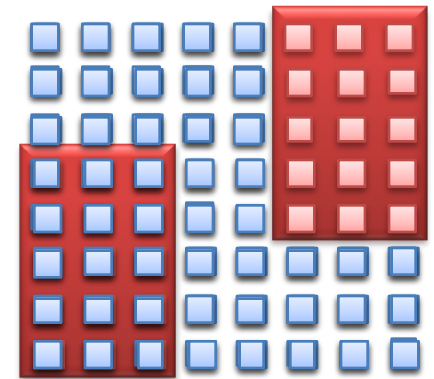
- The VBS encoding is **position independent**
 - The final bit-stream can be calculated from the VBS for different positions in the fabric
- The online decoding algorithm is **simple** since the **global** routing and **detailed** routing have been determined offline
- The resulting VBS can be **smaller** than the equivalent raw bit-stream
 - Clustering islands prior to coding can improve the compression ratio

Outline

- Context
 - FlexTiles project
 - State of the Art on hardware relocation
 - Challenges
- Contributions
 - Position-independent tasks: Virtual Bit-Streams
 - Routing architecture enhancements
- Conclusion & Perspectives

Heterogeneous relocation

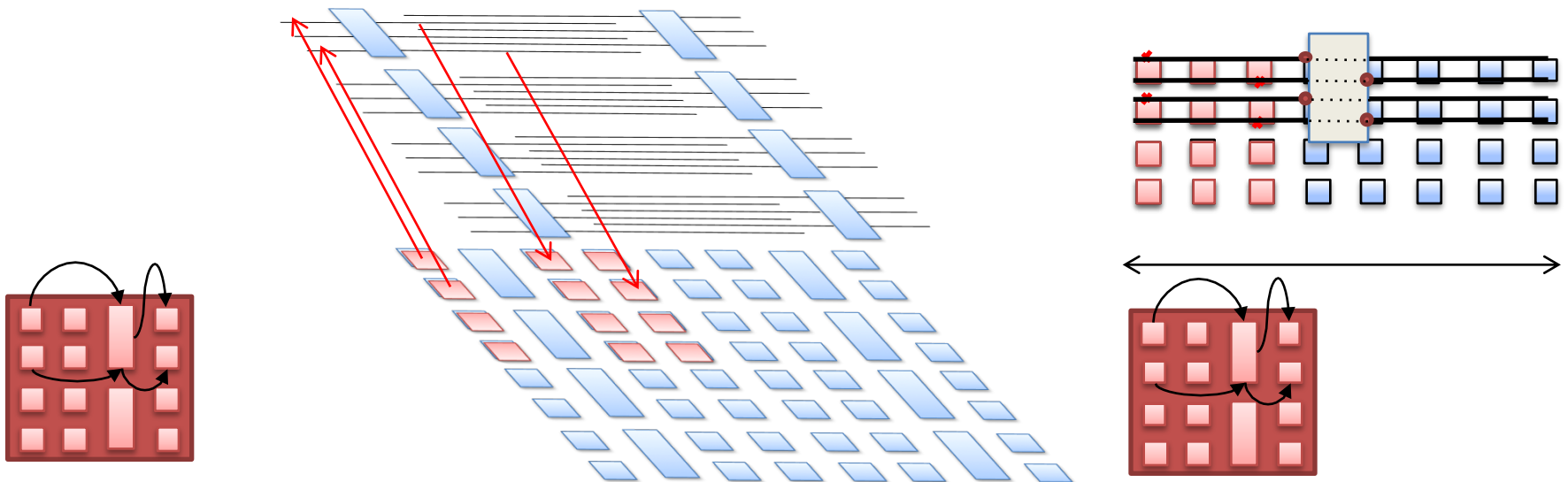
- Homogeneous case
 - No constraint on task placement
 - Regular routing architecture
- Cope with heterogeneity
 - RAM, DSP, 3D I/Os
 - Migration is limited
 - vertically to the same column
 - to the next column containing same complex blocks



- Logic Element (LE)
- Configured LE
- Task

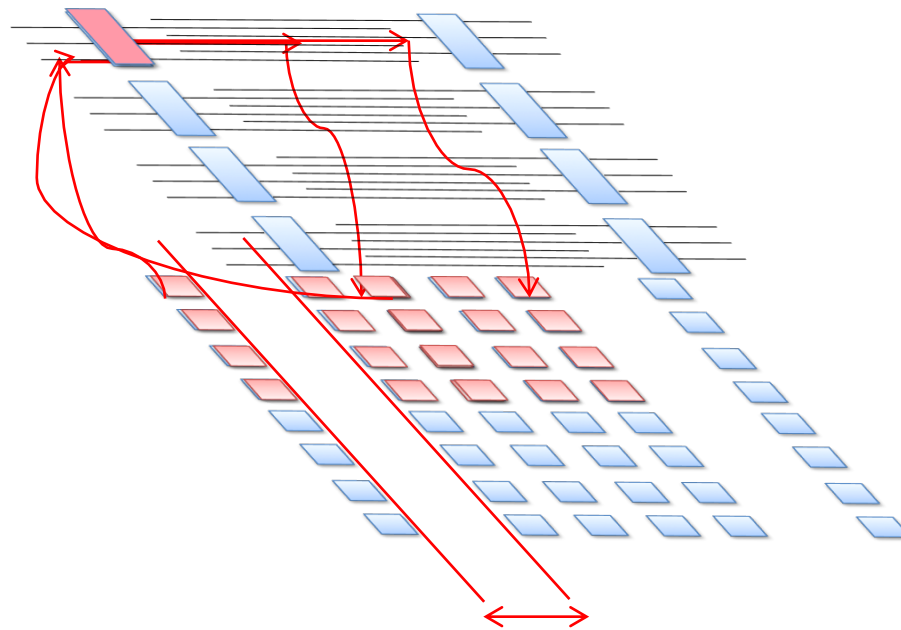
Routing abstraction

- Heterogeneous blocks routing is abstracted from logic routing
 - Long lines allow a trade-off between placement flexibility and routing complexity
 - A two-level routing is performed at runtime:
 - Logic routing (as in the homogeneous case)
 - Heterogeneous block routing through *long lines*

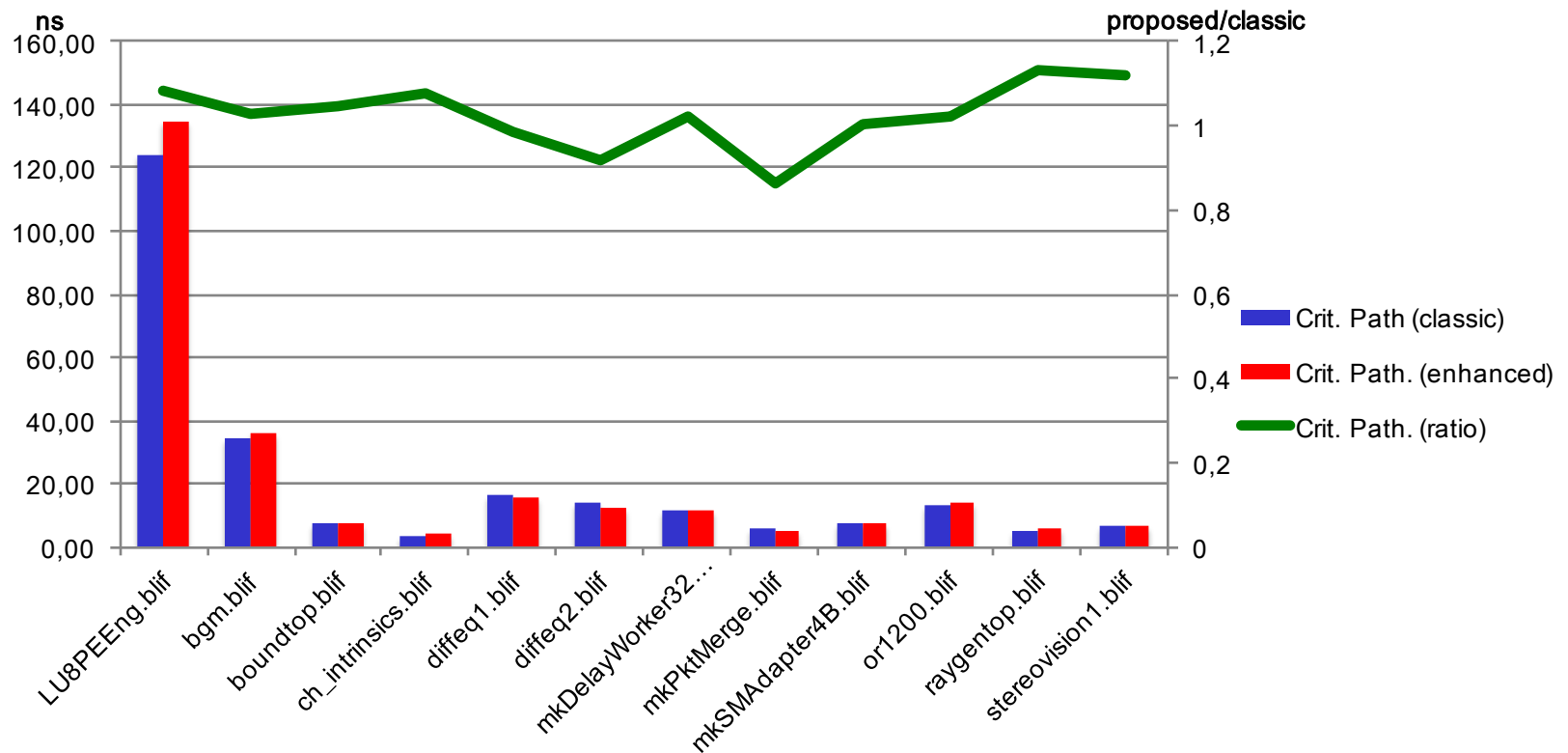


Architecture evaluation

- Estimation of the worst case delay
 - Impossible to predict where connections to long lines will be done
 - Some channels crossing fixed-function blocks are longer

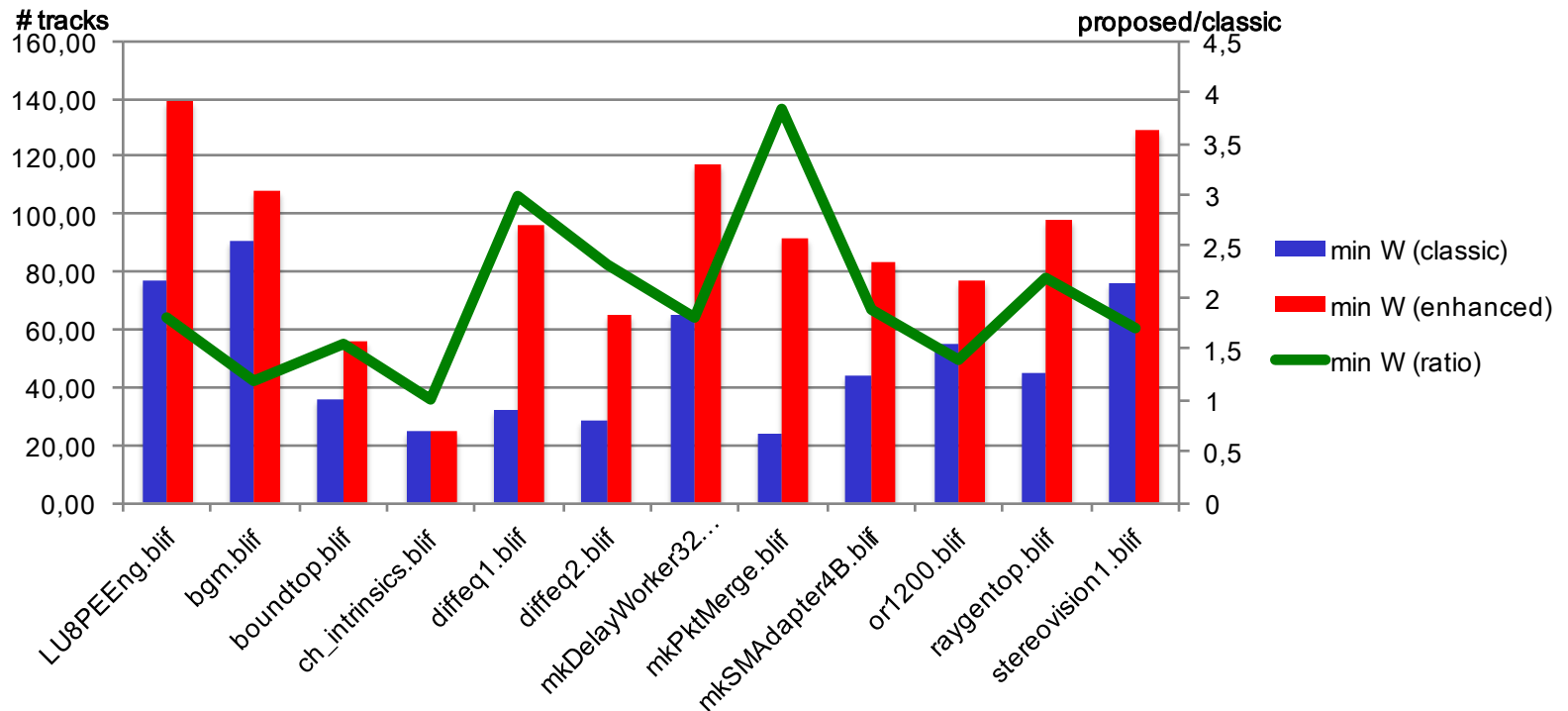


Results: critical path delay



- Simplified Stratix IV architecture with memories and multipliers
- Only 2% delay increase (in average)

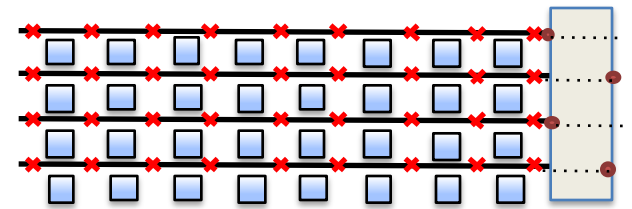
Results: routing resources



- 1.8X channel width increase on average
- Need for specific routing algorithms to deal with the heterogeneous interconnection network

Architecture enhancement overview

- Increases the flexibility of a task placement
- Implemented in a modified version of Versatile Place & Route (VPR)
- Evaluation on critical path delay and area overhead:
 - Only 2% delay increase in average
 - Area overhead
 - Long lines
 - Interconnection switches
 - Area/Flexibility trade-off



Outline

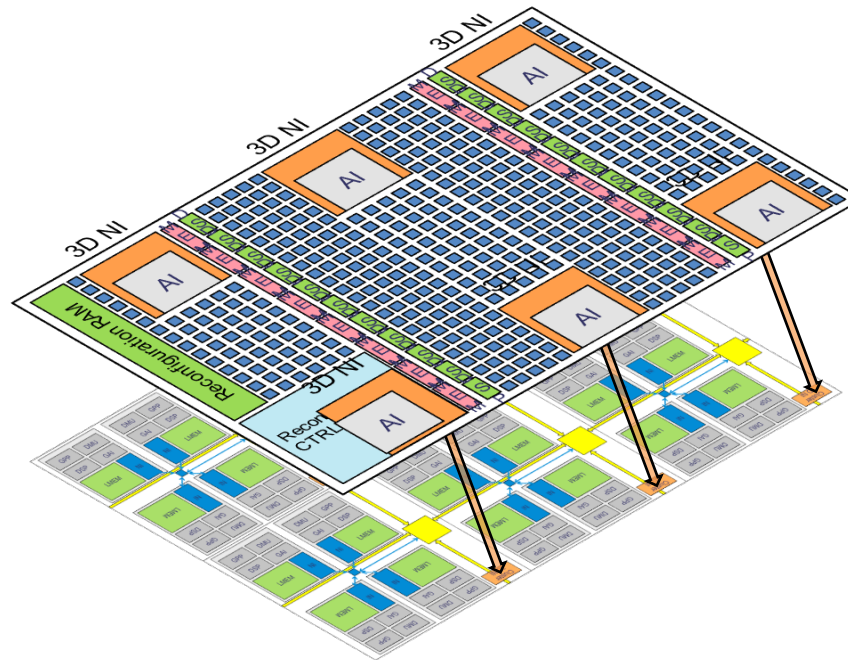
- Context
 - FlexTiles project
 - State of the Art on hardware relocation
 - Challenges
- Contributions
 - Position-independent tasks: Virtual Bit-Streams
 - Routing architecture enhancements
- Conclusions & Perspectives

Conclusions

- Proposed solutions to enhance the placement flexibility in FPGAs
 - Virtual Bit-Streams allow seamless placement of tasks on a logic fabric
 - Enhanced architecture to allow for heterogeneity flexibility
 - A configurable shift-register for enclosing the configurable area
- Genericity and reusability of the VBS
 - Can be extended to other architectures

Conclusions

- Several contributions to the FlexTiles project
 - Specification of the embedded FPGA
 - RTL model implementing VBS relocation
 - Integration of the CAD tools into the FlexTiles tool-flow
 - Main contributor to the eFPGA deliverables



Perspectives

- Evaluation on modern uni-directional architectures and CGRAs
 - May lead to alternate representations of the VBS
- VPR adaptation to the enhanced heterogeneous routing network
- Towards software/hardware virtualization on MPSoC

Perspectives

- Integration of the VBS generation into new features offered by VPR and VTR
 - Latest work on VPR introduced bit-stream generation
- Alternate representation of the logic data in the Virtual Bit-Streams
 - Hash-map of logic equations
- Higher-level of abstraction of routing resources
 - Exploit regularities in the routing

Q&A

Thank you for your attention

References

[Altera2010] *Increasing Design Functionality with Partial and Dynamic Reconfiguration in 28-nm FPGAs*, Altera Corporation, 2010.

[BKT14] C. Beckhoff, D. Koch, and J. Torresen, *Portable Module Relocation and Bitstream Compression for Xilinx FPGAs*, in the Proceedings of the 24th conference of Field Programmable Logic, pp. 30–30.

[Com+02] K. Compton, Z. Li, J. Cooley, S. Knol, and S. Hauck, “Configuration relocation and defragmentation for run-time reconfigurable computing,” *IEEE Transactions on VLSI Systems*, vol. 10, no. 3, pp. 209–220, 2002.

[HL+01] E. Horta, J. W. Lockwood. *PARBIT: a tool to transform bitfiles to implement partial reconfiguration of field programmable gate arrays (FPGAs)*, Tech. Rep. WUCS-01-13, Washington University, 2001.

[HST14] C. Huriaux, O. Sentieys, and R. Tessier, *FPGA Architecture Support for Heterogeneous, Relocatable Partial Bitstreams*, in the Proceedings of the 24th conference of Field Programmable Logic, pp. 30–30.

[HCS15] C. Huriaux, A. Courtay, O. Sentieys, *Design Flow and Run-Time Management for Compressed FPGA Configurations*, in the Proceedings of the 18th DATE conference, Mar. 2015, pp. 1551–1554.

[Jan+15] B. Janßen, F. Schwiegelshohn, M. Koedam, F. Duhem, L. Masing, S. Werner, C. Huriaux, A. Courtay, E. Wheatley, K. Goossens, F. Lemonnier, P. Millet, J. Becker, O. Sentieys, M. Hübner. *Designing Applications for Heterogeneous ManyCore Architectures with the FlexTiles Platform*. In: SAMOS. Samos Island, Greece, July 2015.

[Kal+06] H. Kalte and M. Porrmann, *REPLICA2Pro: Task Relocation by Bit-stream Manipulation in Virtex-II/Pro FPGAs*, in the Proceedings of the 3rd conference on computing frontiers (CF). ACM, 2006, pp. 403–412.

[LVT04] R. Lysecky, F. Vahid, and S. X.-D. Tan, *Dynamic FPGA routing for just-in-time FPGA compilation*, in the Proceedings of the 41th Design Automation Conference, 2004, pp. 954–959.

References

- [Put+14] A. Putnam et al., *A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services*, 41st Int. Symp. on Computer Architecture (ISCA), 2014.
- [Sen+14] O. Sentieys, A. Courtay, C. Hurlaux and S. Pillement, *Method and Device for Programming an FPGA*, European Patent EP2894572. Jan. 2014.
- [Swi+15] P. Swierczynski, M. Fybriak, and C. Paar, C. Hurlaux, and R. Tessier, *Protecting against Cryptographic Trojans in FPGAs*, in the Proceedings of the 23rd IEEE International Symposium on Field-Programmable Custom Computing Machines, FCCM. IEEE. Vancouver, Canada, May 2015.
- [Tou+12] M. Touiza, G. Ochoa-Ruiz, E.-B. Bourenane, A. Guessoum, and K. Messaoudi, *A novel methodology for accelerating bitstream relocation in partially reconfigurable systems*, *Microprocessors and Microsystems*, vol. 37, no. 3, pp. 358–372, 2012.
- [Ven+10] G. Venkatesh, J. Sampson, N. Goulding, et al. *Conservation cores: reducing the energy of mature computations*. ACM SIGARCH Computer Architecture News. ACM, 2010. p. 205-218.
- [Xilinx2013] *Partial Reconfiguration User Guide, UG702*, Xilinx, Inc., 2013.