Automated Physical Hierarchy Generation: Tools and Methodology

Ali El-Zein, Alvan Ng, Benedikt Geukes,
Maya H. Safieddine, Wolfgang Roesner

Outline

- Motivation
- Morph-Hier Overview
- Morph-Hier Building Blocks
- Large Synthesizable Block Creation
- Chiplet Integration Entity Creation
- Distributed Pervasive Logic Application
- Conclusion

Why Hierarchy Morphing? – Different Constraints

Logical Organization Preference

Verification Focus

Logic Ownership

Functional Adjacency

Physical Organization Preference

Implementation Focus

Physical Optimization

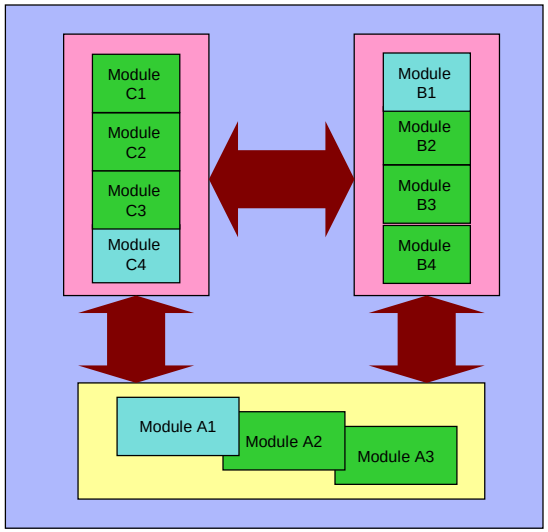
Geographic Adjacency

Combined Single Hierarchy

Iterative PD Annotation

High Coordination Effort

Less Efficient

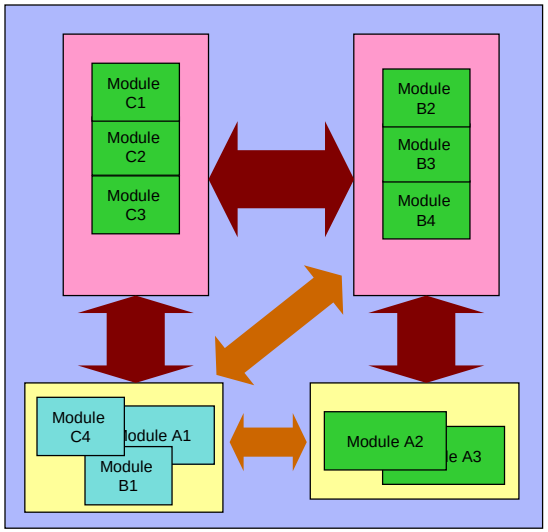


Design Quality

Performance

Power

Area



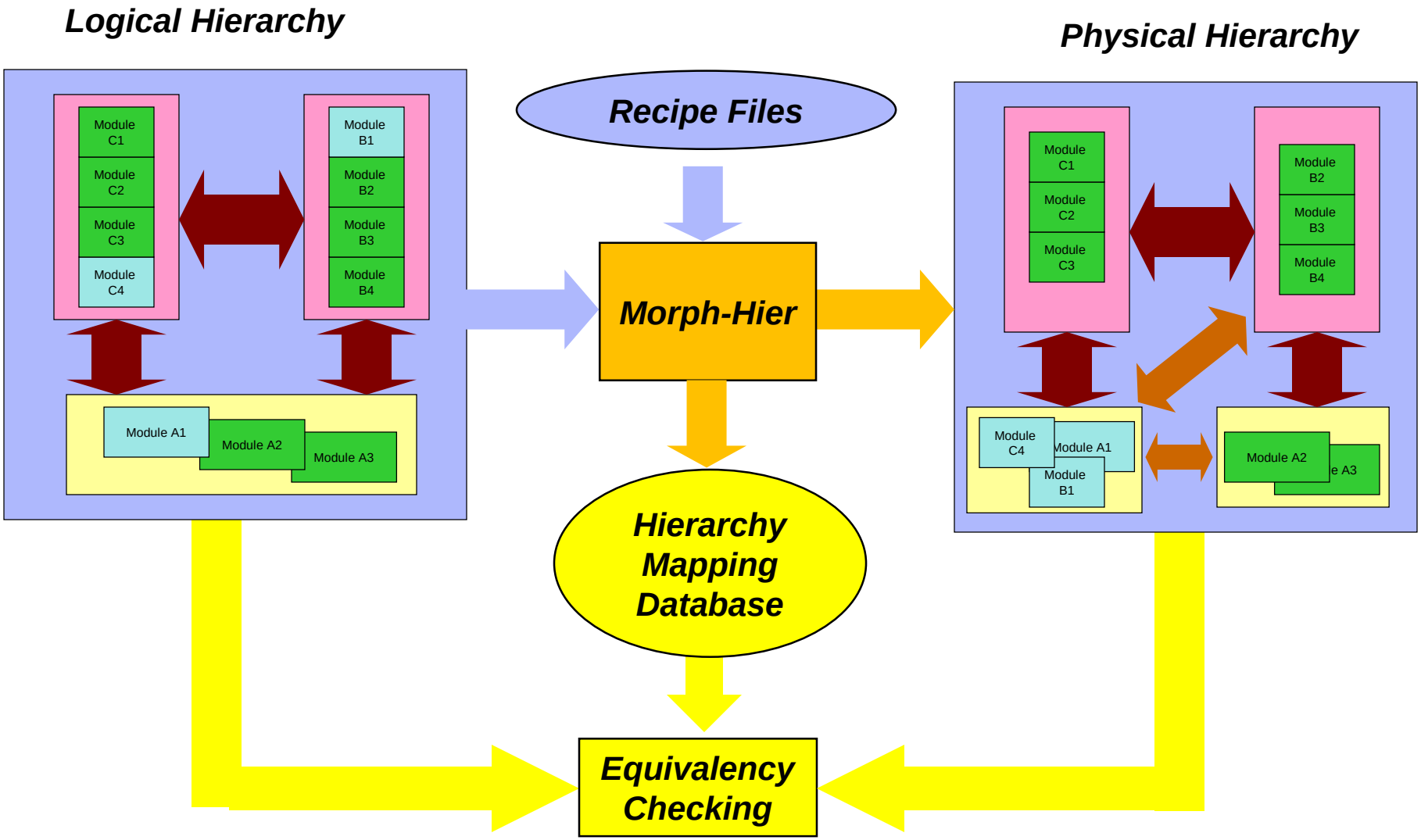
Outline

- Motivation
- Hierarchy Morphing (Morph-Hier) Overview
- Morph-Hier Building Blocks
- Large Synthesizable Block Creation
- Chiplet Integration Entity Creation
- Distributed Pervasive Logic Application
- Conclusion

What is Morph-Hier?

- An IBM VHDL manipulation tool
- Takes any legal VHDL as an input
- Mainly manipulates hierarchy
  - Does not change functionality
- Starts with a logical hierarchy as the golden source
- Converts to a physical hierarchy based on a set of recipes
- Makes it feasible to have both the logical and physical hierarchies

Overview Of Morph-Hier Flow



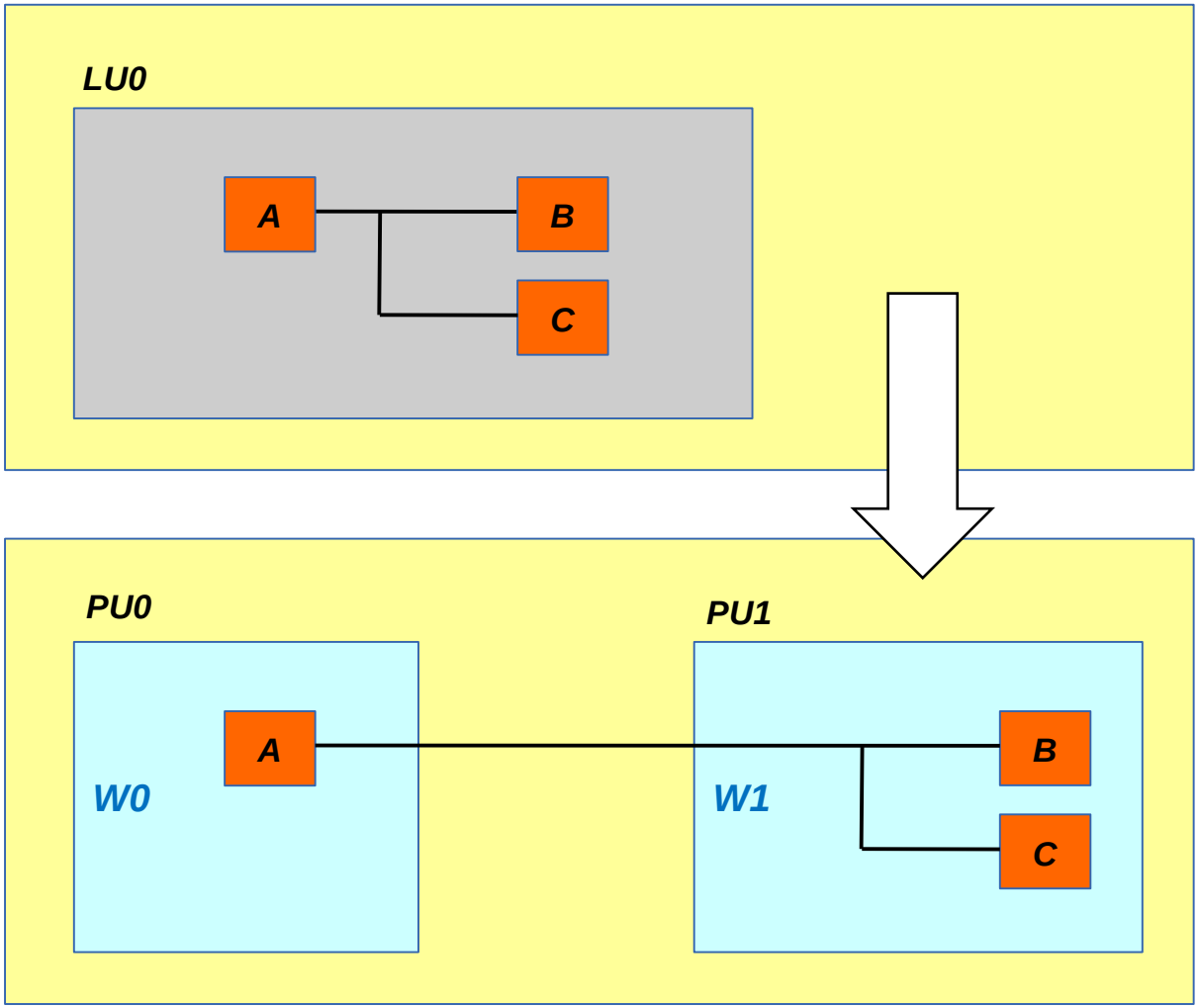
Outline

- Motivation
- Morph-Hier Overview
- Morph-Hier Building Blocks
- Large Synthesizable Block Creation
- Chiplet Integration Entity Creation
- Distributed Pervasive Logic Application
- Conclusion

Morph-Hier Major Building Blocks

- Recipe files - simple text files contain Morph-Hier statements
- Wrappers Creation, Instantiation, and Deletion
- Instance Move Statement
- Port Optimization
- Pin Cloning
- Subway Creation – tunnels a net through an entity
- Scheduler - Reorder the statements to yield consistent result
- Equivalency Checker

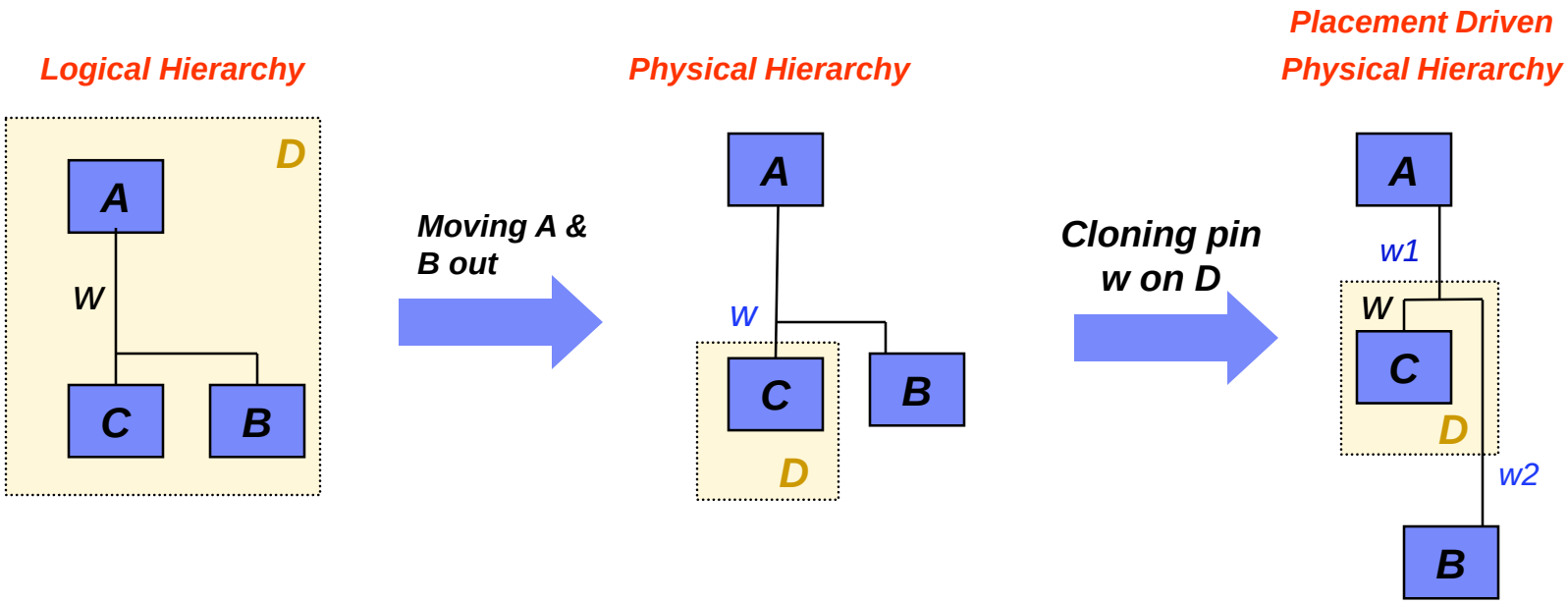
Morph-Hier Basic Operations



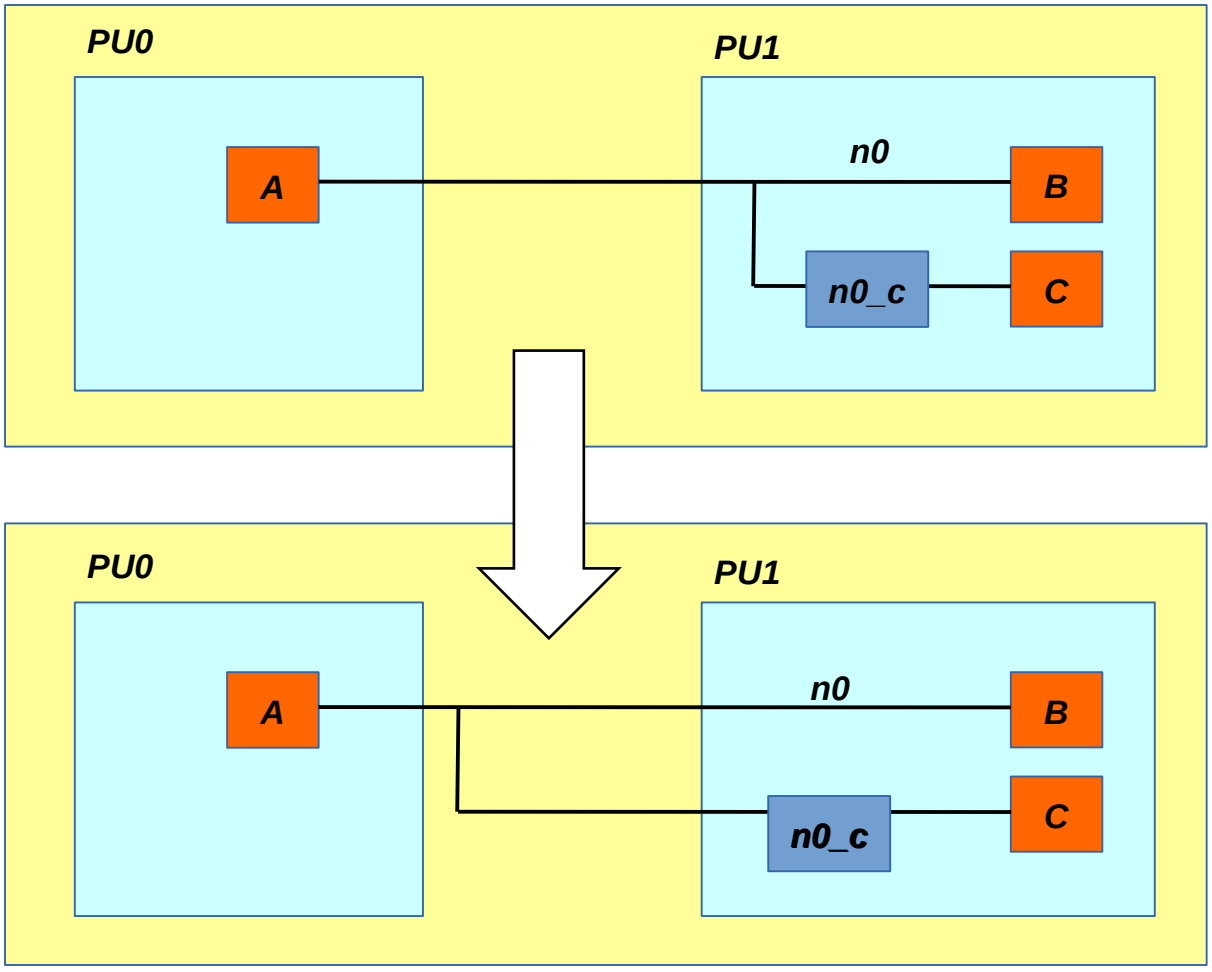

```
#wrapper creation and
instantiation

proto w0;
inst w0 PU0;

proto w1
inst w1 PU1;

#instances move

move LU0/A PU0/A;
move LU0/B PU1/B;
move LU0/C PU1/C;
```

Pin Cloning Requirement

- ***Instead of optimizing out pins, sometimes we need to duplicate pins***
  - ***This is usually done for routing and timing reasons***



Pin Cloning Implementation

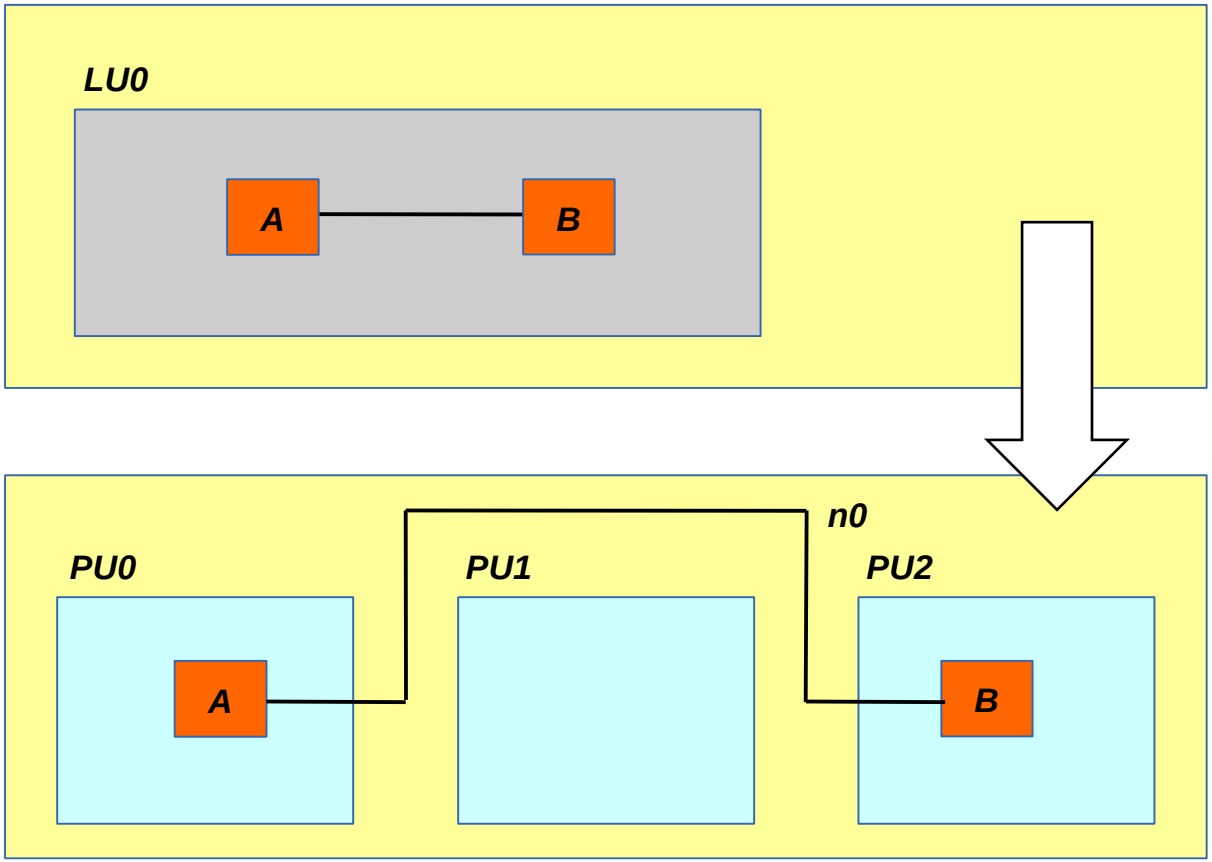


#pseudo command
Insert virtual buffer n0_c between net n0 and instance C in PU1;

move PU1/n0_c n0_c top;

The virtual buffer n0_c could be dissolved or implemented as real buffer or inverter pair

Subway Requirement



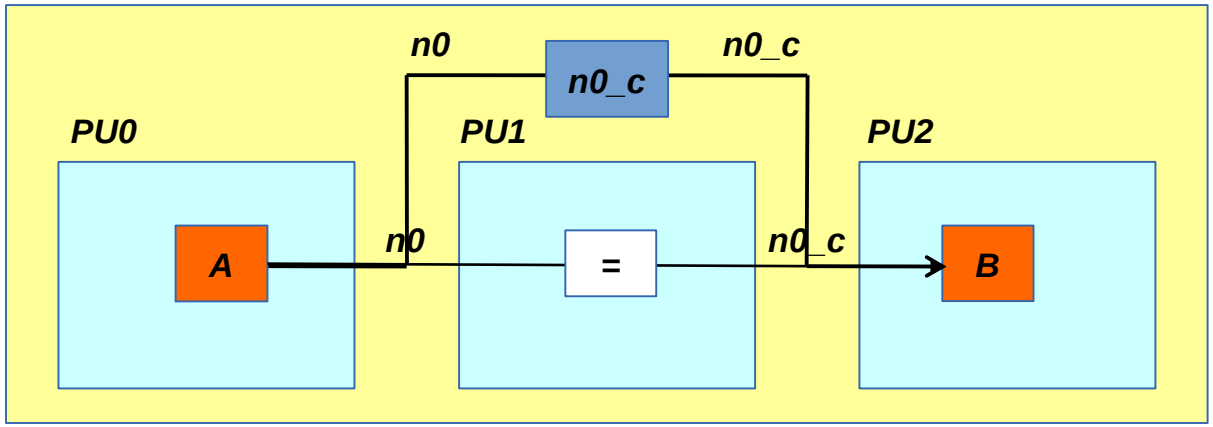

```
#wrapper creation and
instantiation

proto w0;
inst w0 PU0;

proto w1
inst w1 PU1;

proto w2
inst w2 PU2;

#instances move

move LU0/A PU0/A;
move LU0/B PU2/B;
```

- ***The wire n0 runs around PU1 is not timing/power optimal***

Subway Creation

Clone ASSIGN buffer converted to assignment

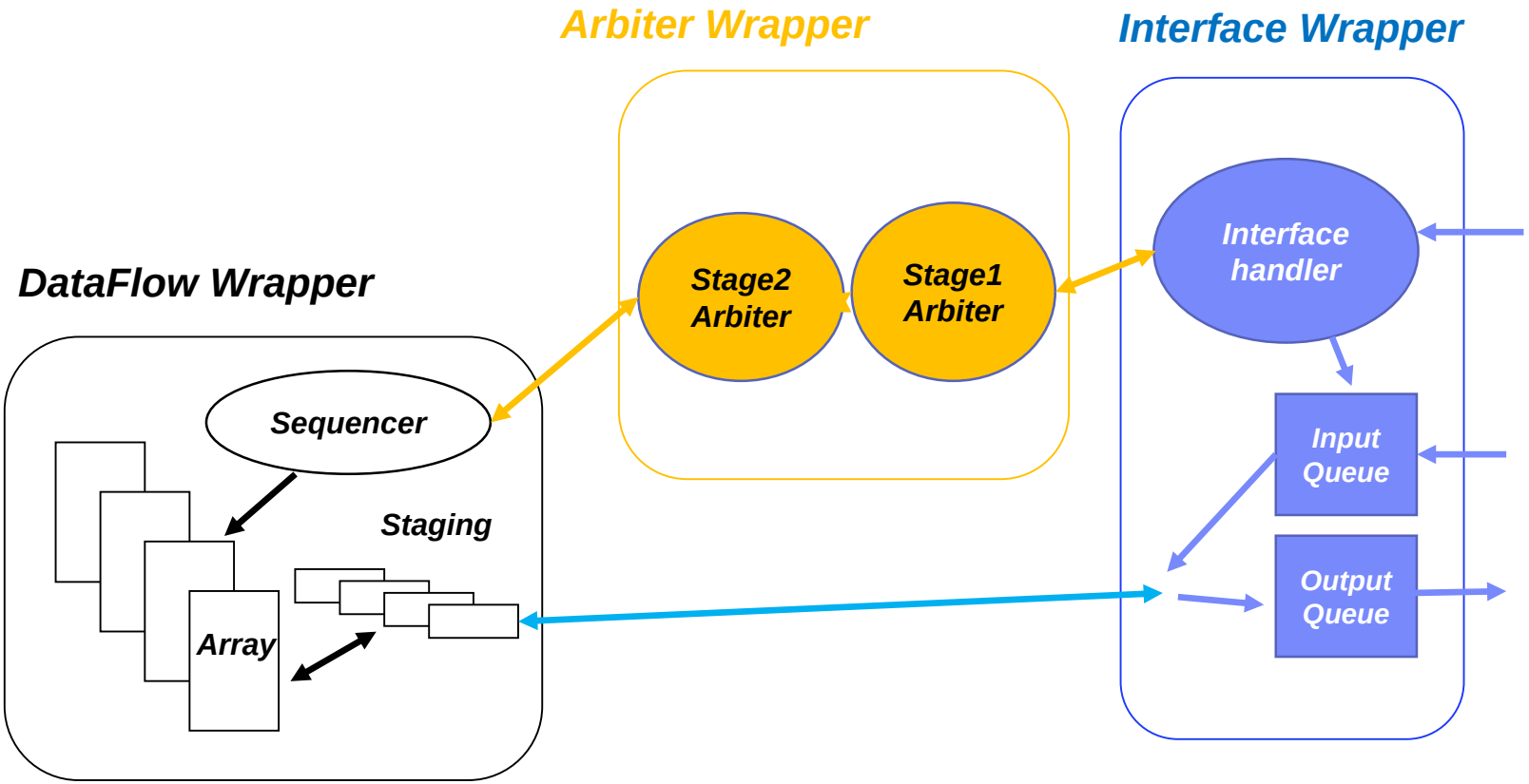
pclone associated PD_VHD10 : PU2;



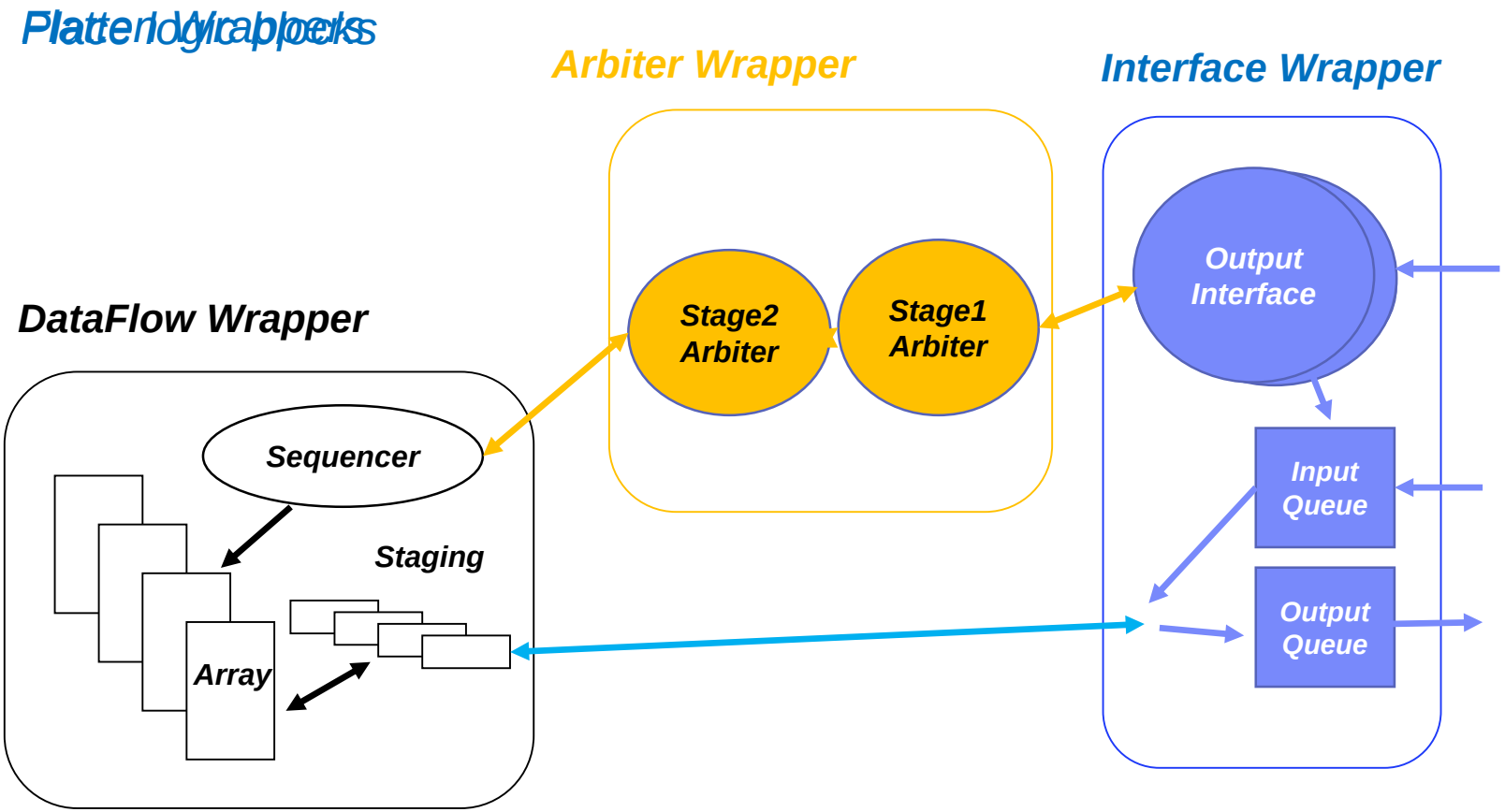
Outline

- Motivation
- Morph-Hier Overview
- Morph-Hier Building Blocks
- Large Synthesizable Block Creation
- Chiplet Integration Entity Creation
- Distributed Pervasive Logic Application
- Conclusion

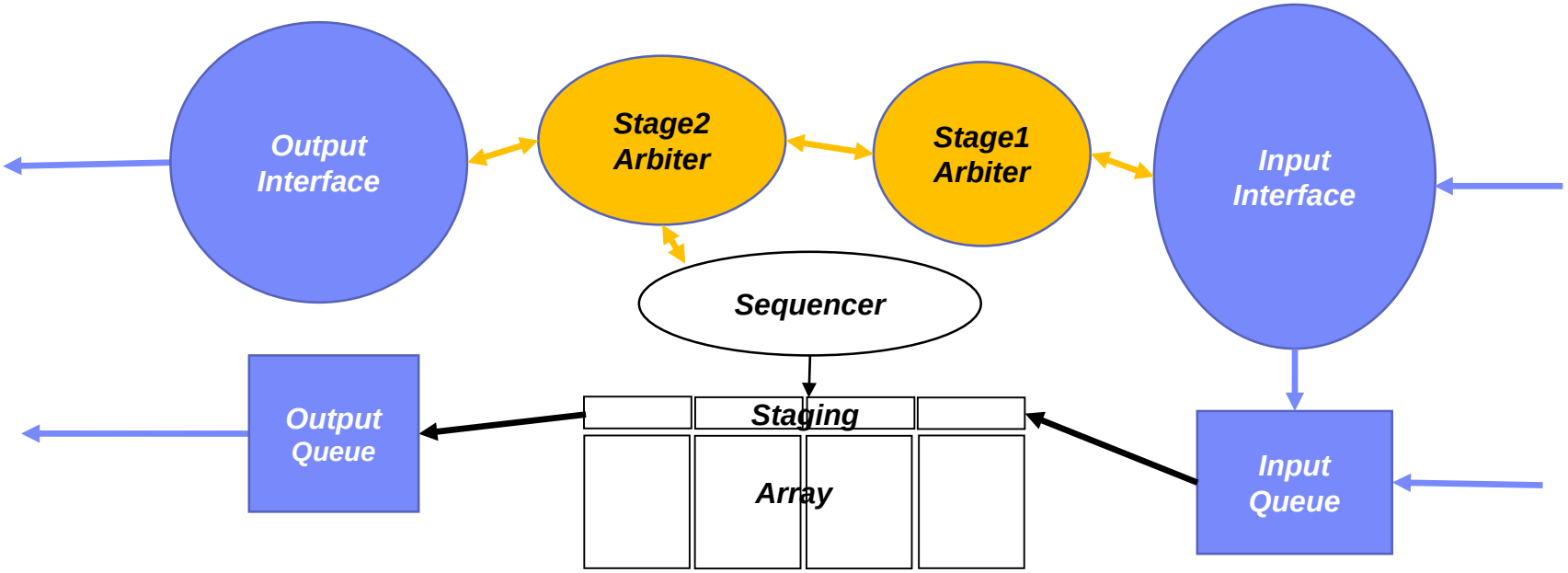
Unit Logical VHDL Organization



Integration Floorplanning

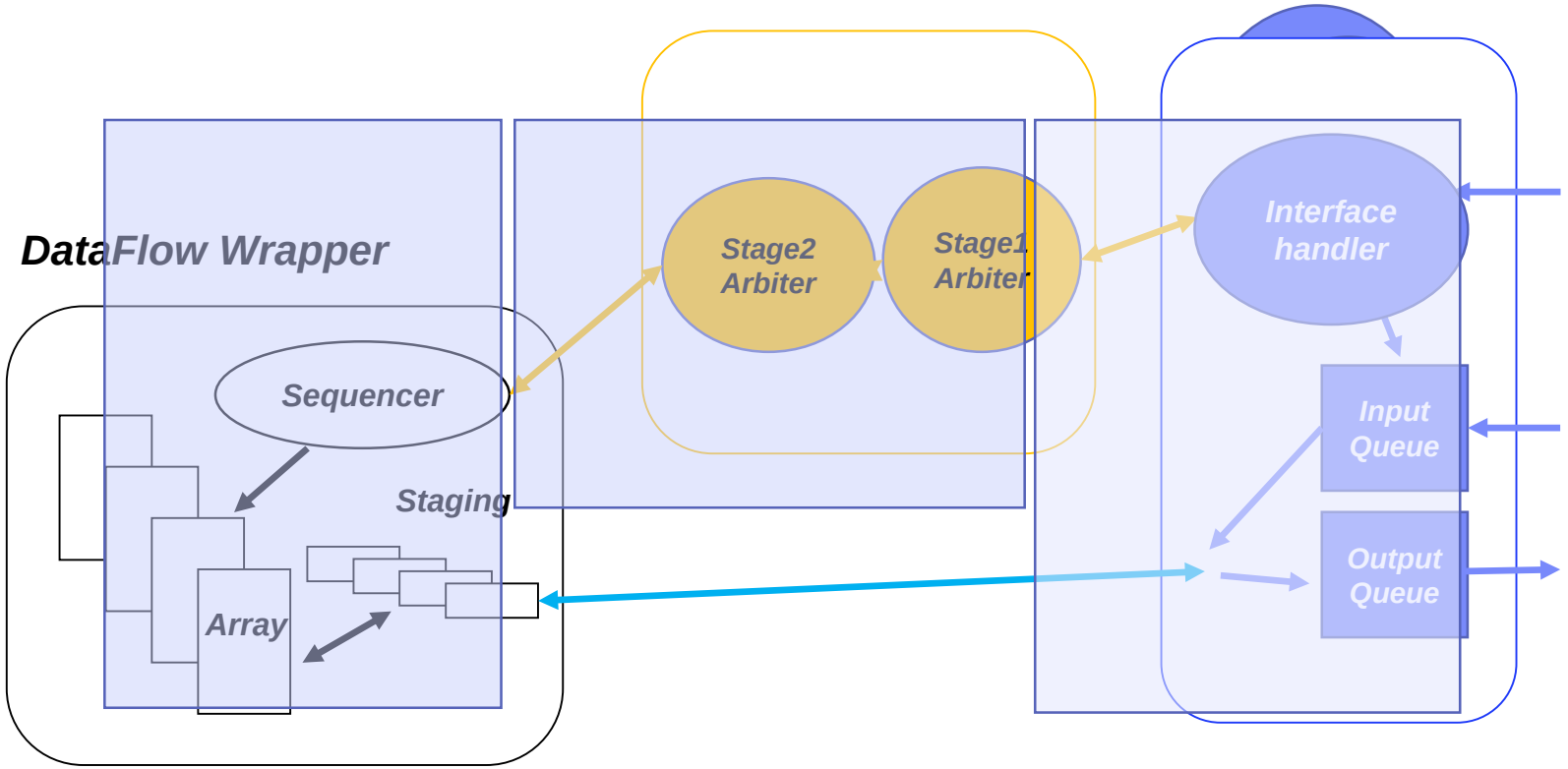


Unit Physical Floorplan With Small Logical Blocks



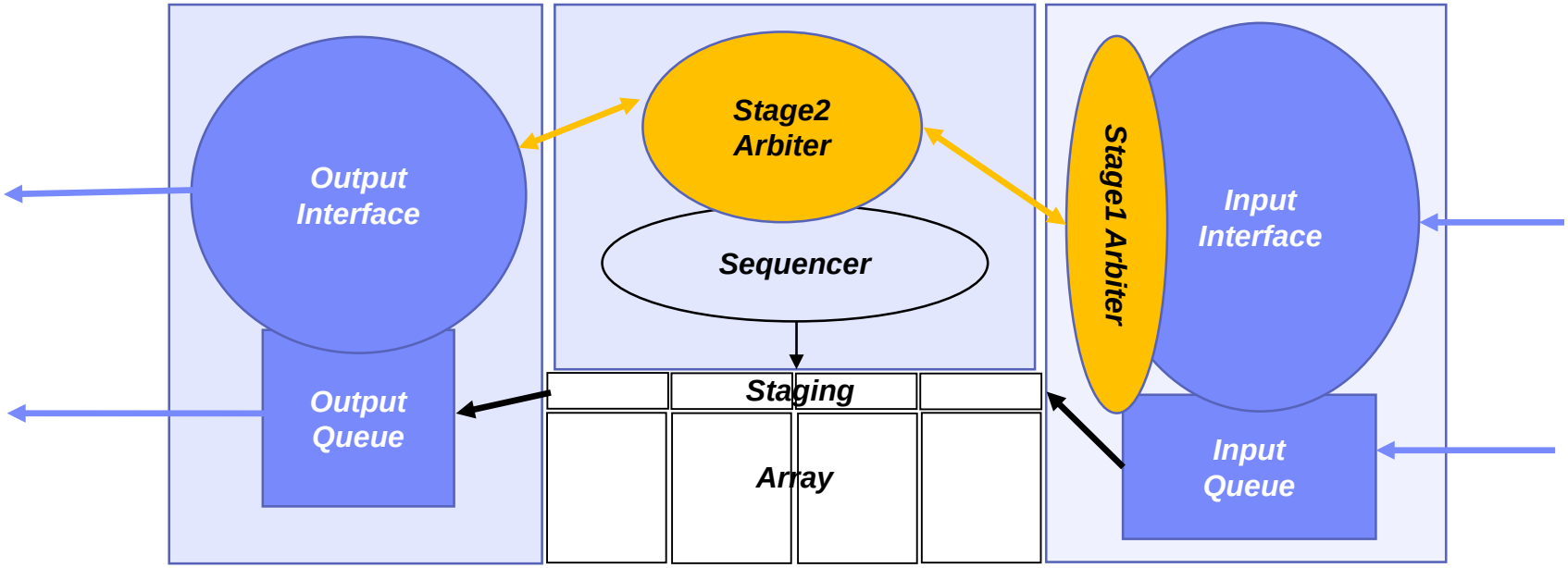
Large Synthesizable Blocks Creation Using Morph-Hier

Creating large blocks optimizes block timing and block floorplan



Unit Physical Floorplan With Large Synthesizable Blocks

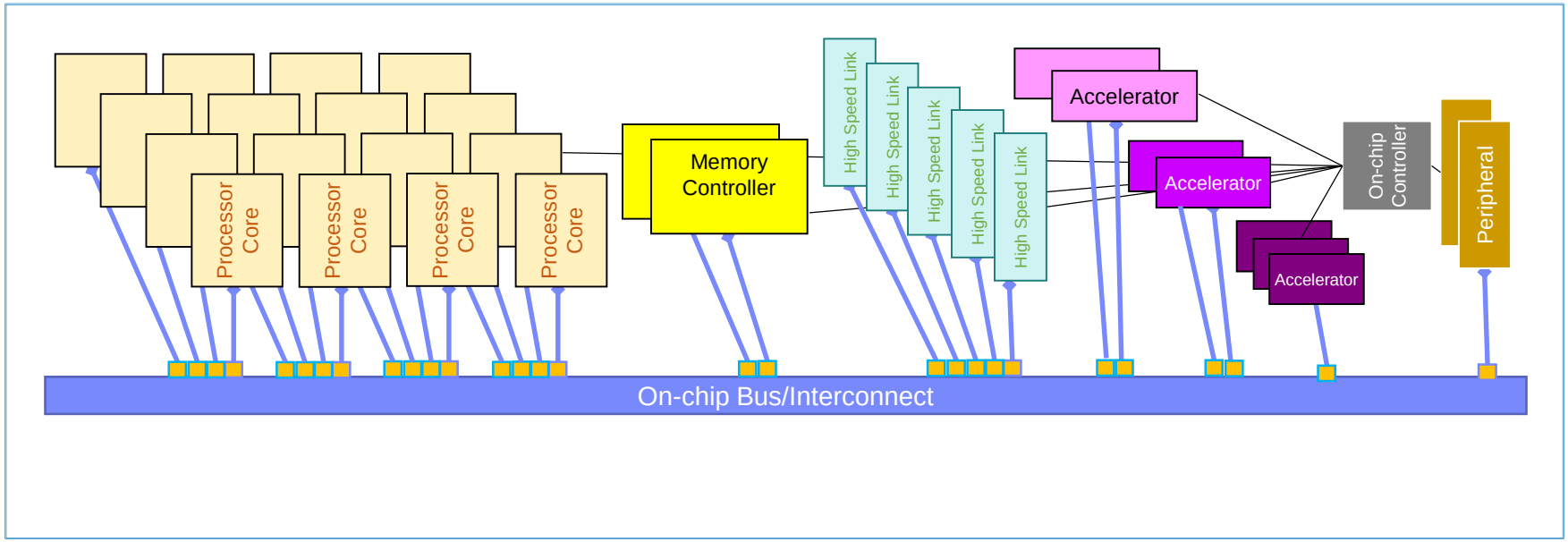
Benefits: Better area, timing, and power optimizations



Outline

- Motivation
- Morph-Hier Overview
- Morph-Hier Building Blocks
- Large Synthesizable Block Creation
- Chiplet Integration Entity Creation
- Distributed Pervasive Logic Application
- Conclusion

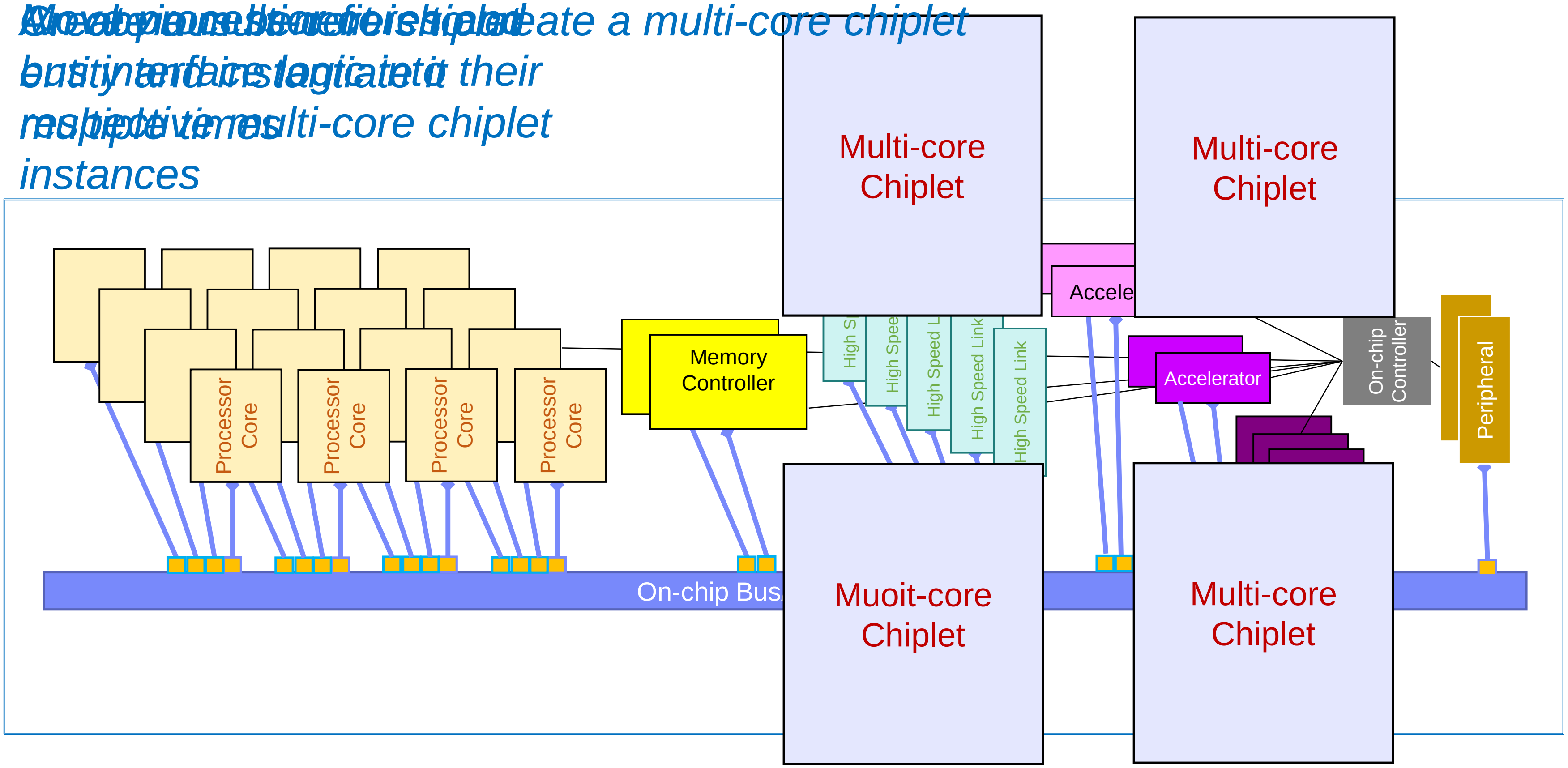
Multi-core Processor Chip Logical VHDL Organization



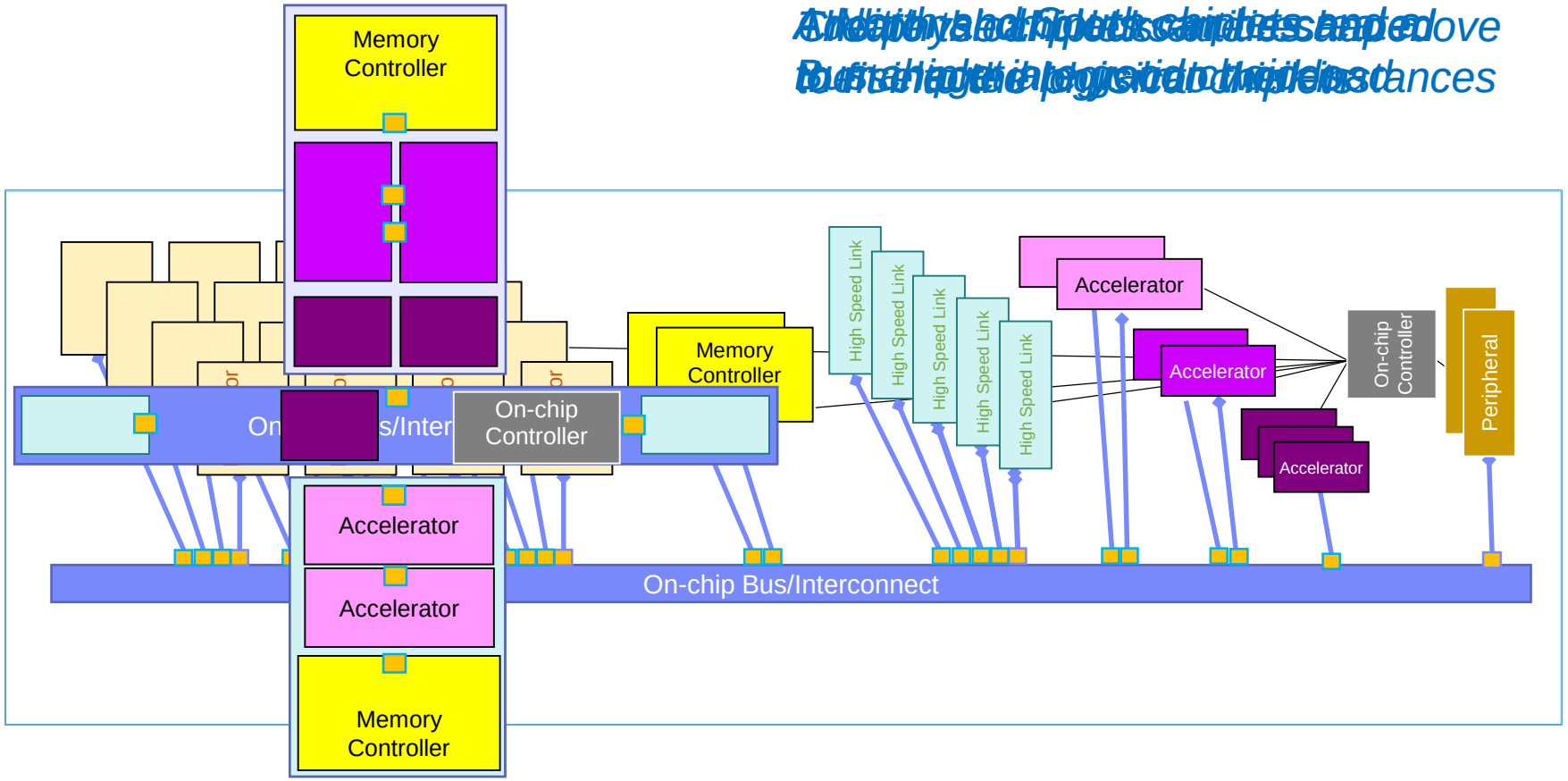
Creation Of Multi-core Chiplet For Integration Reusability

Create a multi-core chiplet entity and instantiate it multiple times

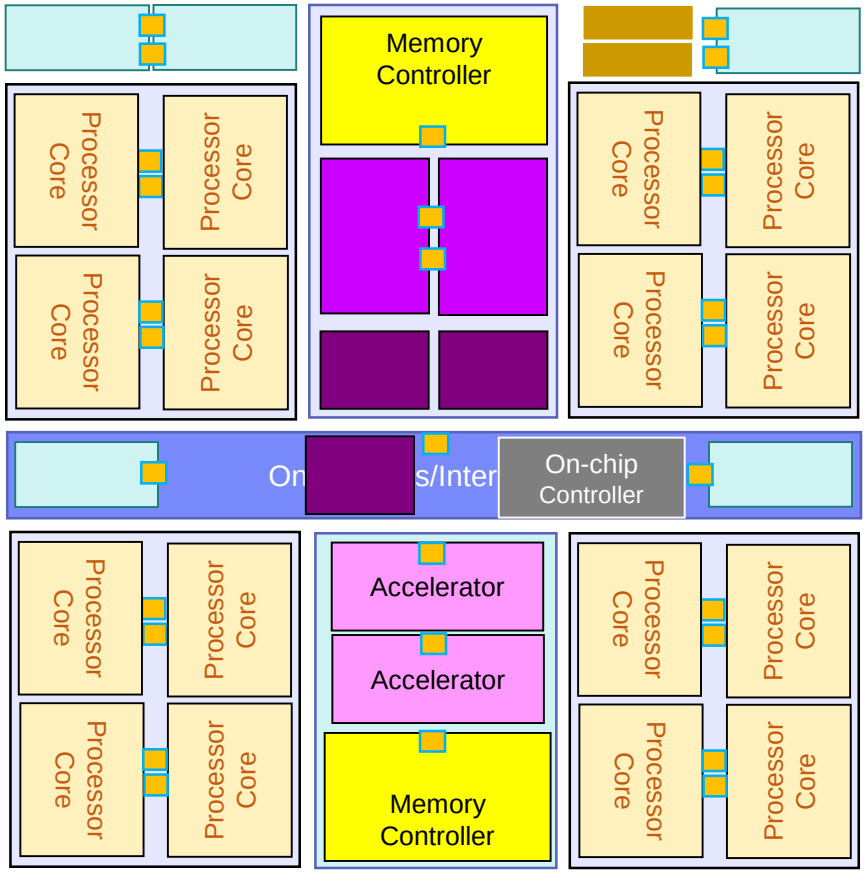
Move processor cores and bus interface logic into their respective multi-core chiplet instances



Create Integration Chiplets For Manageability

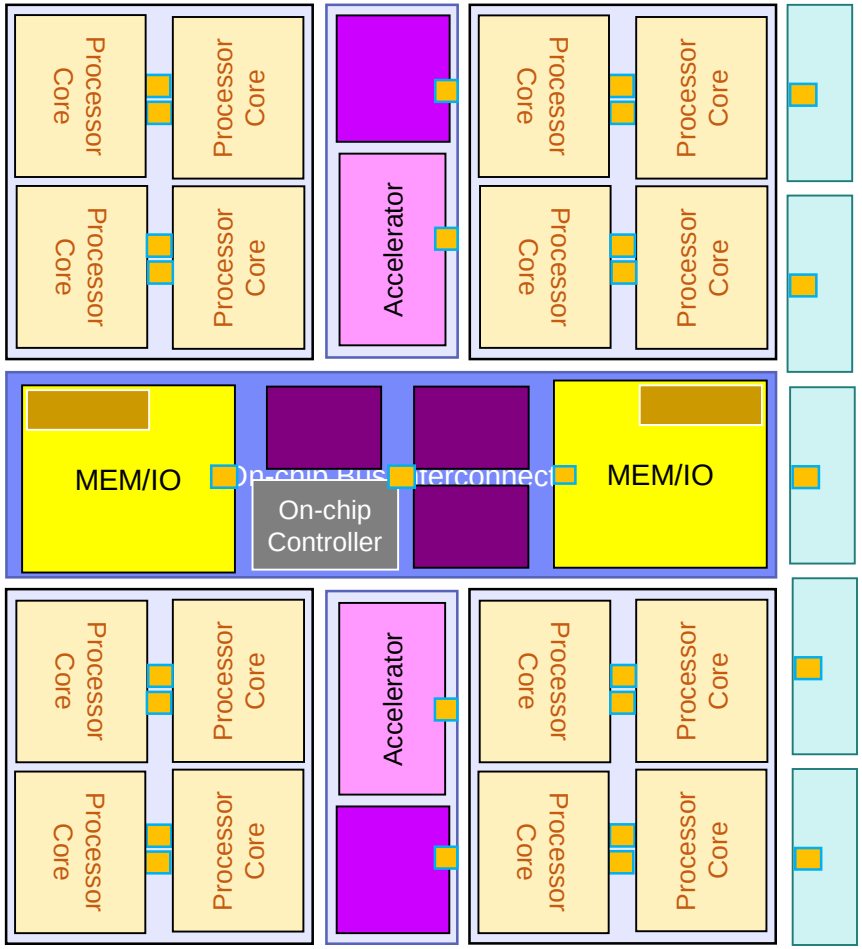


Multi-core Processor Chip Physical Floorplan



- ***Quad-core chiplet instantiated 4 times***
- ***Center stripe bus chiplet with 2x high speed link, 1 small accelerator, and the on-chip controller***
- ***Top chiplet contains 1 memory controller, 2 small accelerators, and 2 medium accelerators***
- ***Bottom chiplet contains 1 memory controller and 2 large accelerators***
- ***Stack the rest of circuitries in the open spaces at the top***

Alternative Chip Physical Floorplan



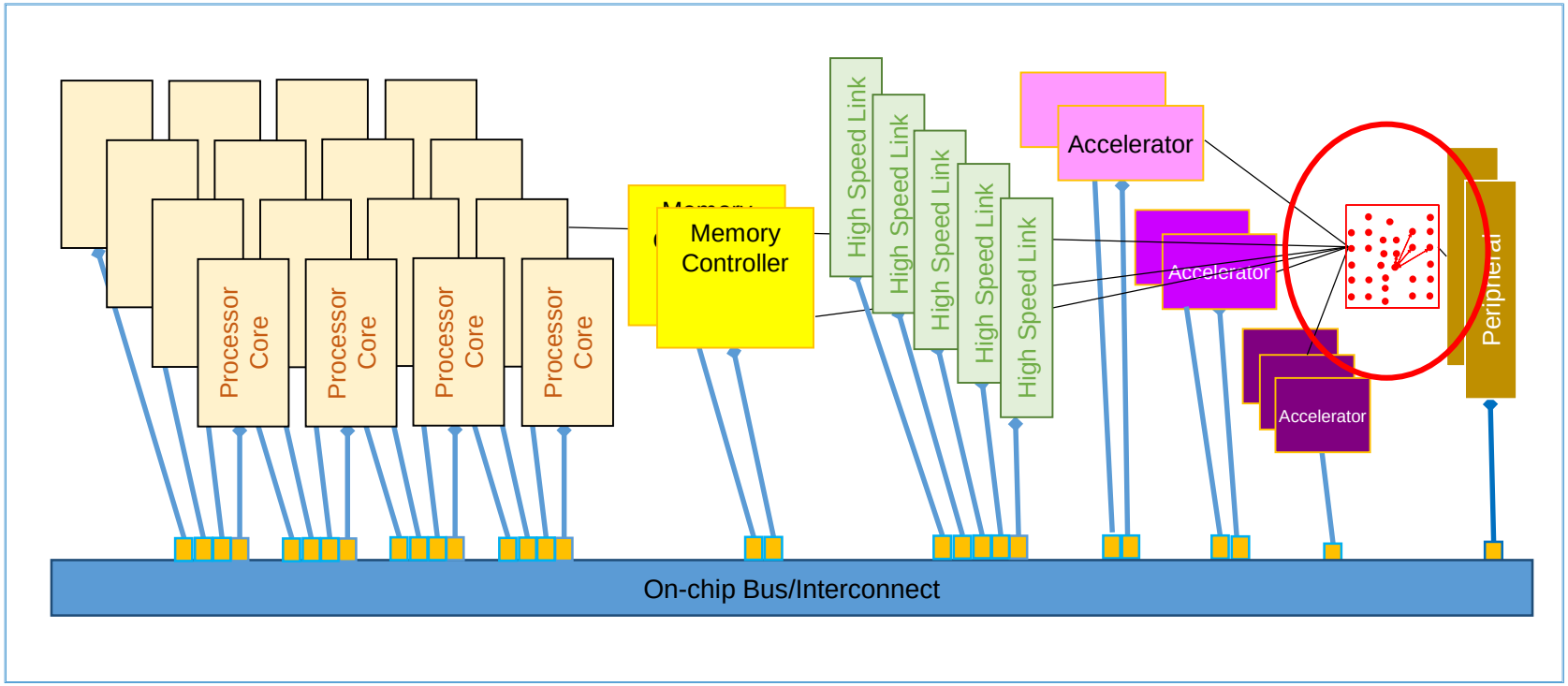
- **Quad-core chiplet instantiated 4 times**
- **Center stripe bus chiplet contains 2x Memory-Peripheral combined unit, 3x small accelerator, and the on-chip controller**
- **One accelerator chiplet instantiated twice which contains a large and a medium accelerator**
- **Stack the High-Speed Links on the right**

Outline

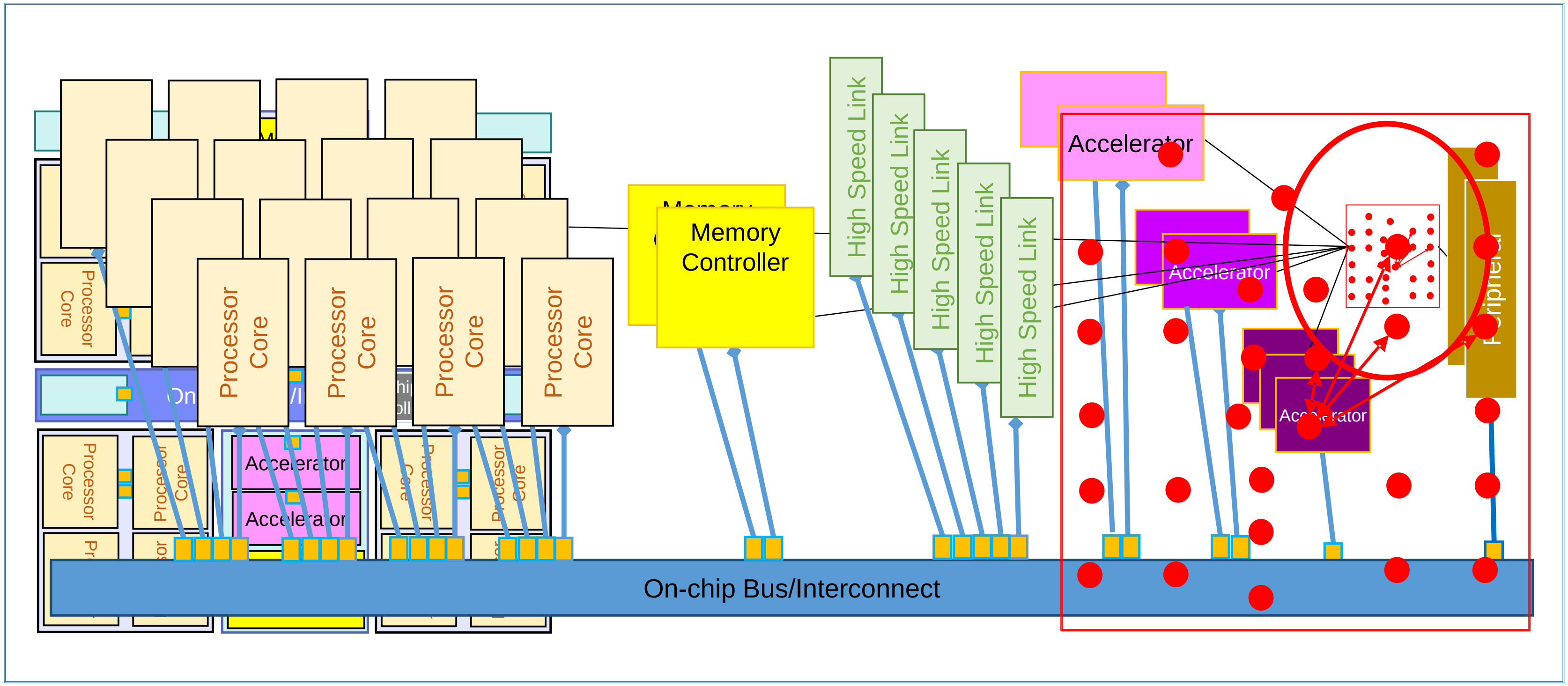
- Motivation
- Morph-Hier Overview
- Morph-Hier Building Blocks
- Large Synthesizable Block Creation
- Chiplet Integration Entity Creation
- Distributed Pervasive Logic Application
- Conclusion

Pervasive Logic Centralized VHDL Organization

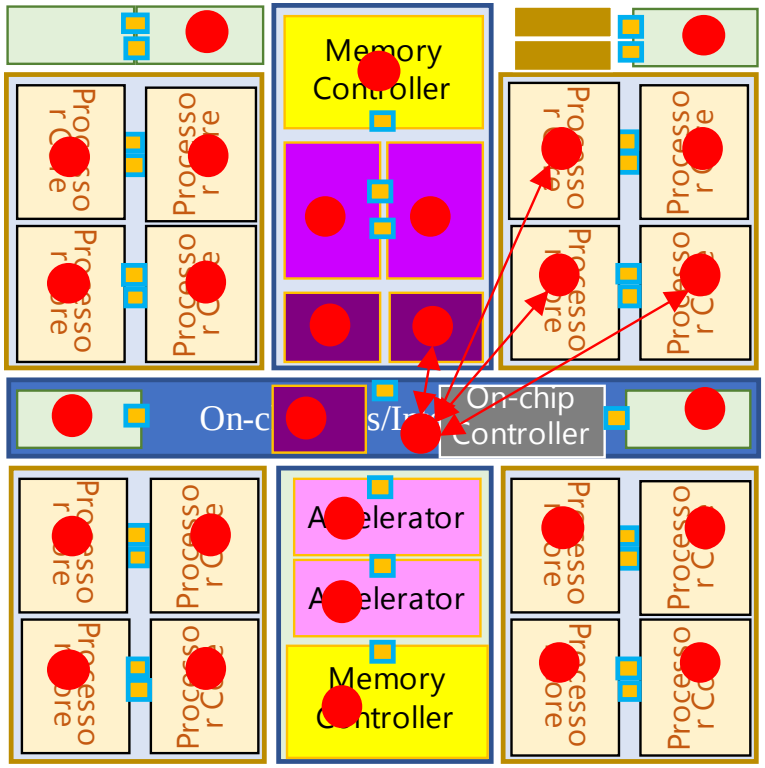
- On-chip Controller Logic
- Test Logic
- Miscellaneous Circuitries



Distribute Pervasive Logic Using Morph-Hier



Centralized Pervasive Logic Distributed To Physical Units



Benefits:

- Parallel logic design
  - Concurrent with functional units
- Verification Speedup
  - Self contained unit
- Design quality
  - Lower bug rate

Outline

- Motivation
- Morph-Hier Overview
- Morph-Hier Building Blocks
- Large Synthesizable Block Creation
- Chiplet Integration Entity Creation
- Distributed Pervasive Logic Application
- Conclusion

Conclusion

- IBM Power9 chip has validated the value of using a hierarchy mapping tool like Morph-Hier to transform logical hierarchy for physical implementation
- To a smaller extent, it benefits area, timing, and power optimizations by expediting the creation of large synthesizable blocks
- To a larger extent, it expands floorplanning flexibility for chip integration by effortlessly creating alternative chiplet partitions
- The biggest value is recognized in the streamlining of verification model and the resulting reduction of logic bugs