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

[Diagram showing hierarchical organization and morphing process]
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

**Logical Hierarchy**
- Module A1
- Module A2
- Module A3
- Module A4
- Module B1
- Module B2
- Module B3
- Module B4
- Module C1
- Module C2
- Module C3
- Module C4

**Physical Hierarchy**
- Module A1
- Module A2
- Module A3
- Module A4
- Module B1
- Module B2
- Module B3
- Module B4
- Module C1
- Module C2
- Module C3
- Module C4

**Recipe Files**

**Morph-Hier**

**Hierarchy Mapping Database**

**Equivalency Checking**
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

**LU0**

- A
- B
- C

**PU0**

- A

**W0**

**PU1**

- B
- C

**W1**

# 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

![Diagram showing logical and physical hierarchies with pin cloning]

**Logical Hierarchy**

- A
- B
- C
- D

**Physical Hierarchy**

- Moving A & B out
- Cloning pin w on D
Pin Cloning Implementation

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

#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

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

Clone ASSIGN is converted to assignment in generated PD VHDL.

pu0 = pu0c

pu0c = pu0

pu1 = pu1c

pu1c = pu1

pu2 = pu2

pu2c = 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

DataFlow Wrapper

Arbiter Wrapper

Stage2 Arbiter

Stage1 Arbiter

Interface Wrapper

Interface handler

Input Queue

Output Queue

Sequencer

Staging

Array

Arbiter Wrapper
Integration Floorplanning

**DataFlow Wrapper**

- **Array**
  - **Sequencer**
  - **Staging**

**Arbiter Wrapper**

- **Stage1 Arbiter**
- **Stage2 Arbiter**

**Interface Wrapper**

- **Output Interface**
  - **Input Queue**
  - **Output Queue**
Unit Physical Floorplan With Small Logical Blocks

- Output Interface
- Stage 2 Arbiter
- Stage 1 Arbiter
- Sequencer
- Output Queue
- Staging Array
- Input Queue
- Input Interface
Large Synthesizable Blocks Creation Using Morph-Hier

Breaking grouping entities block into small block floorplan

Arbiter Wrapper

Interface Wrapper

DataFlow Wrapper

Sequencer

Staging

Array

Stage2 Arbiter

Stage1 Arbiter

Interface handler

Input Queue

Output Queue

Create large block entities

Examine grouping opportunities using the small block floorplan

Large Synthesizable Blocks Creation Using Morph-Hier
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

An obvious benefit is to 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

Additional chiplets can be created to manage integration workload. A North and South chiplets and a Bus chiplet are good choices. Create the chiplets entities and move the selected logic into their instances. The physical blocks are reshaped to fit into the physical chiplets.
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

The pervasive logic is pushed into the physical entities 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