Andreas Koch
akoch@cs.berkeley.edu

FLAME
Flexible API for Module-based Environments
Overview

System architecture
- Modular
- Open

Flexible interfaces
- FLAME
- Basics
- Capabilities
- Examples

Summary
System Architecture

- System Synthesis
  - Datapath Composing
    - Datapath DFG Controller
  - Functional Specifications
    - Delay, Area, Power
  - Abstracted Control
- Module Selection
  - Microplacement
  - Compaction
- Floorplanning
  - Regular Datapath Layout
- Back-End P&R
  - Layout
- Module Library
  - Structure
    - Interconnects
    - Topology
  - Datapath Layout
  - API
- C, C++, Java, ...
- Controller
- System Architecture
- API
Andreas Koch • DARPA Review 9803

**FLAME – Basic Concepts**

- Currently aimed at unified library access
- "Dynamic databook"
- Query/Reply scheme
- Active Interface
- Queries constrained by
- Operand widths
- Data types
- Functional
- Synthesises
- Topology

**FLAME**

Replies use restricted views on data
Sample FLAME Scenario

Module Library

Datapath Composer

Available functions on XC4000?

addsub, add, increment

Size, speed, and usage of addsub for 16-bit operands?

17 logic cells, 10.1 ns, control input = 0 for add, = 1 for sub

Netlist of the add-2 slice in EON format?

1x carry-init, 16x add-2

Bit-sliced structure of addsub, op width = 16?

s = a \lor b \lor \text{cin}

1x carry-init = 1 for sub?

Synthesis

Netlist of the add-2 slice in EON format?

S1^2 + S1^0 \cdot \text{cin} + S0^1 \cdot \text{cin}

for 16-bit operands?

Size, speed, and usage of addsub?

addsub, add, increment

Available functions on XC4000?
<table>
<thead>
<tr>
<th>Concise representations of FLAME Capabilities</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Module functions</td>
<td></td>
</tr>
<tr>
<td>Control interfaces</td>
<td></td>
</tr>
<tr>
<td>Module functions</td>
<td></td>
</tr>
<tr>
<td>Timing</td>
<td></td>
</tr>
<tr>
<td>Area</td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td></td>
</tr>
<tr>
<td>Internal composition</td>
<td></td>
</tr>
<tr>
<td>Topology</td>
<td></td>
</tr>
<tr>
<td>Regular bit-slices</td>
<td></td>
</tr>
<tr>
<td>Hierarchical and symmetric array</td>
<td></td>
</tr>
<tr>
<td>Logic and interconnect density</td>
<td></td>
</tr>
<tr>
<td>Embedded netlists and simulation models</td>
<td></td>
</tr>
<tr>
<td><strong>Embedded netlists and simulation models</strong></td>
<td></td>
</tr>
</tbody>
</table>
FLAME Properties

- Flexible
  - Extensible by introducing new parameters
- Efficient
  - Quick generation, parsing and manipulation
- Portable
  - Single procedural entry point
  - Single data structure: associative list
- Easy-to-use
  - Compatible with previous specifications
  - Gradual refinement of specifications
- Minimal initial requirements
Determine library functions for synthesis

Example 1
Determine resource and control specifications
FLAME wrapper for current generators

- Generation
- Manipulation
- Parsing

Implementation support library

Complete API specification
Summary

Directions for future work

Unified Library access

Examples

Capabilities

Properties

Flexible API for Module-based Environments

Need for powerful Interfaces

System architecture