High-level Synthesis of Memory Bound and Irregular Parallel Applications with Bambu

Vito Giovanni Castellana, Antonino Tumeo, and Fabrizio Ferrandi

Pacific Northwest NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

High Performance Reconfigurable Computing

Application Domain

- Several emerging application are **irregular**
  - Process **large** pointer-based data structures (graphs, trees, grids)
  - Generate unpredictable memory accesses
  - Memory bandwidth bound, high synchronization intensity
  - Feature high degrees of (task-level) parallelism
- Examples: bioinformatics, knowledge discovery, data analytics

Hybrid Architectures

- CPUs + FPGA co-processors (e.g., Convey HC-x systems)
- Idea: execute performance critical kernel on dedicated hardware components
- Promising for memory bound and irregular applications **however...**
- **...** requires experienced designers for the implementation of the custom hardware accelerators
  - High development costs
  - Long time for prototype for complex designs

High Level Synthesis (HLS)

- Allows the automatic implementation of a digital circuit, starting from its behavioral description (C/C++ code)
- May fill the gap between development costs and benefits of hardware acceleration (performance/power)

Main Limitations of Typical HLS Frameworks

Coarse-grained parallelism exploitation

- Most HLS approaches focus on Instruction-level Parallelism
- Limitations due to
  - Input language semantics
  - Architectural model (e.g., Finite State Machine with Datapath, FSMD)
- **Both inherently serial**

Support for complex memory subsystems

- Large amounts of data to process
- Memory operations may represent a bottleneck for performance
- Fundamental to support shared/distributed memories
  - Higher memory availability/bandwidth

Proposed Solutions

Novel architectural model for HLS

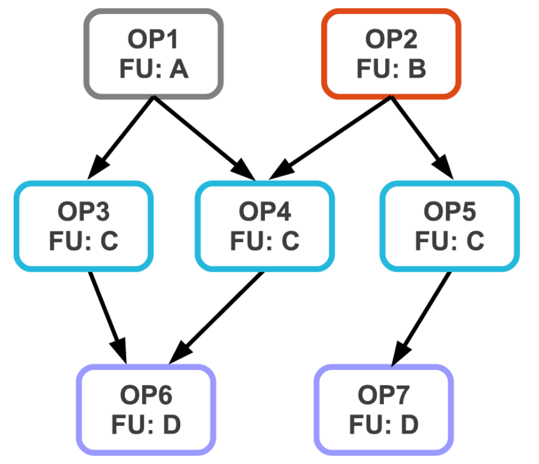
- Exploit s a Parallel Controller for managing concurrent execution flows

Hardware modules for interfacing shared parallel memories

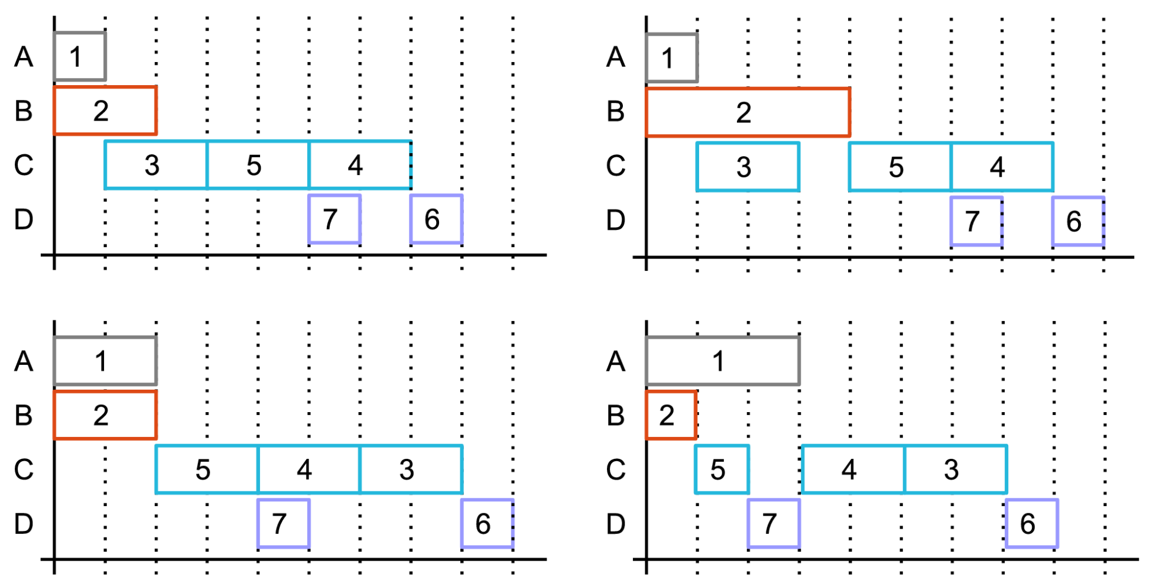
- Concurrency and synchronization management
  - Modules automatically integrated in the final design

Parallel Controller Architecture

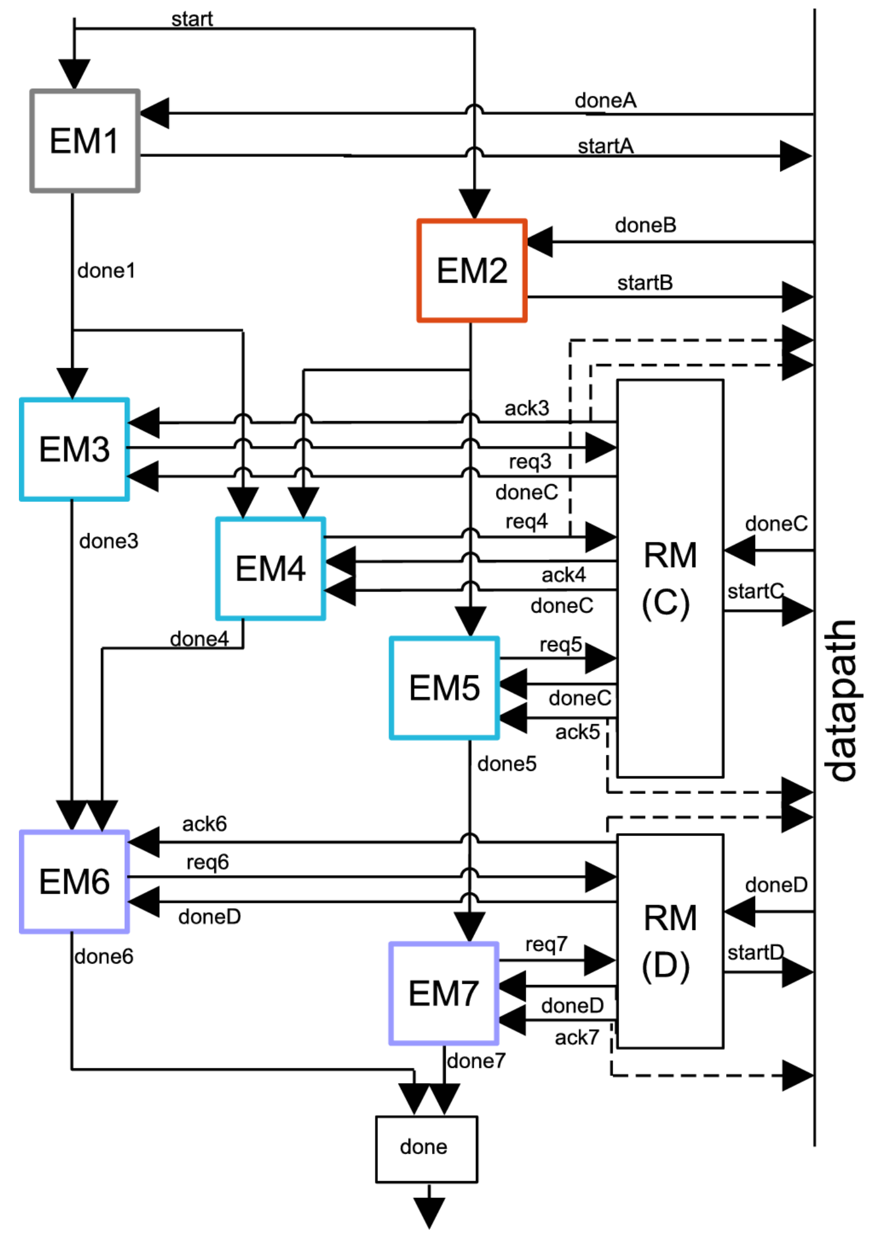
- Set of communicating control elements (Execution Managers, EM)
- Each EM establishes when operations/tasks can start at runtime
  - **Dynamic execution** paradigm
- Dedicated hardware for checking
  - Satisfaction of dependency constraints
  - Resource availability → interaction with arbiters (Resource Managers, RM)
- **Linear complexity** with respect to the number of operations/tasks
  - Regardless of their latency



Example Dependence Graph



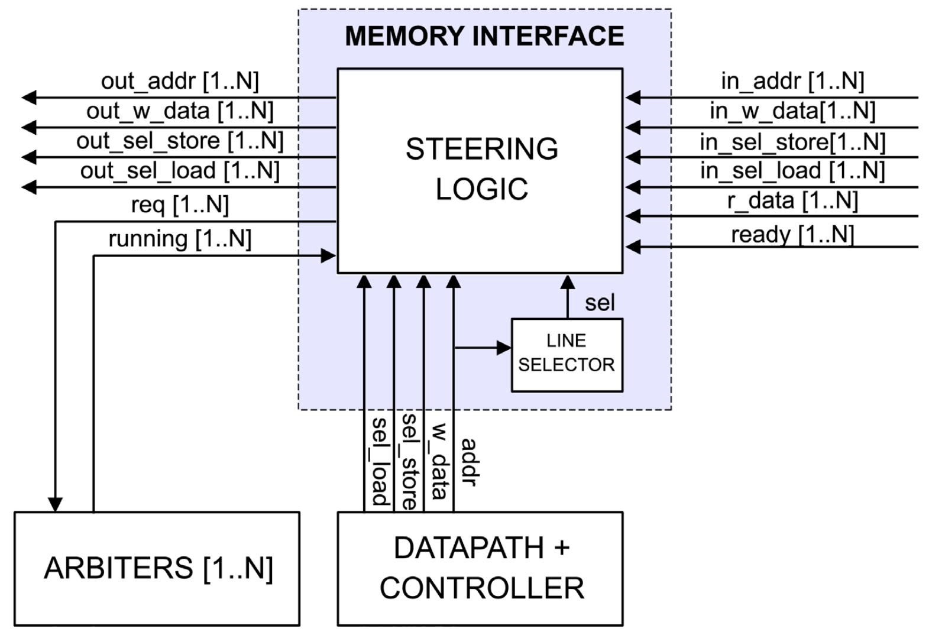
Dynamic Behavior: Example of Execution Traces when Varying Runtime Latencies



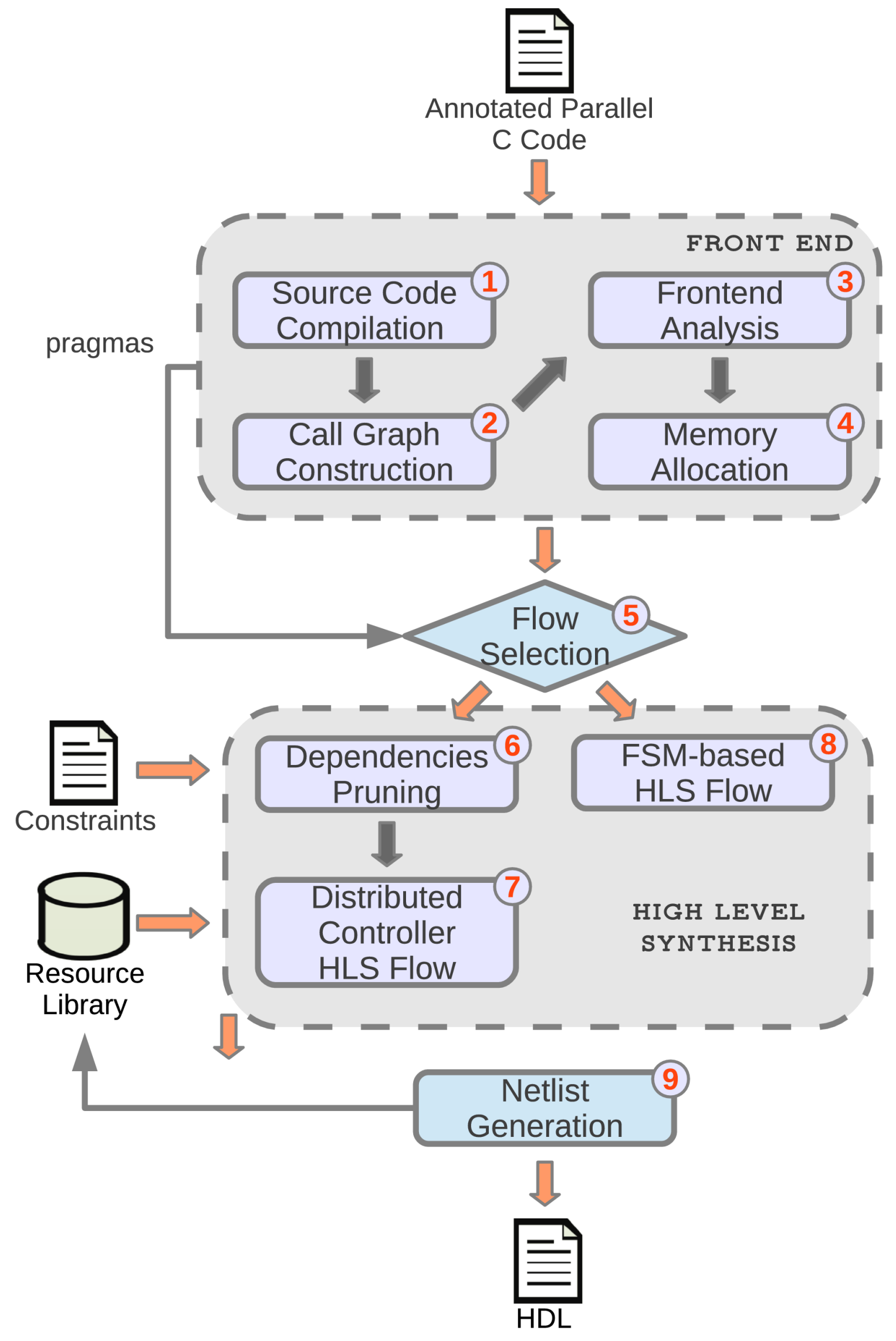
Associated Parallel Controller

Memory Interface Controllers

- Map unpredictable memory accesses to several memory ports
  - Dynamic resolution of memory addresses → support for pointer arithmetics
- Manage **concurrency**
  - lightweight arbiters associated with no delay penalties
- Manage **synchronization**
  - hardware implementation of atomic memory operations



Proposed High-level Synthesis Flow



Extends the ***Bambu*** framework

- Input: C-code specification
  - Optionally annotated
- Output: Verilog implementation
- Hierarchical/modular synthesis process
  - Follows the call structure of the input
  - Enhances modules reusability
  - Reduces final design complexity
- Exploits both Parallel and FSM controllers
  - FSM is preferred for serial kernels
- Exploits both local and external memories benchmarks
  - Local BRAMs on FPGAs
  - Shared memories
  - Distributed/multi-ported memories

Bambu is freely available at
http://panda.dei.polimi.it/

Experimental Evaluation

- Evaluation on a set of OpenMP benchmarks
  - Up to six concurrent kernels

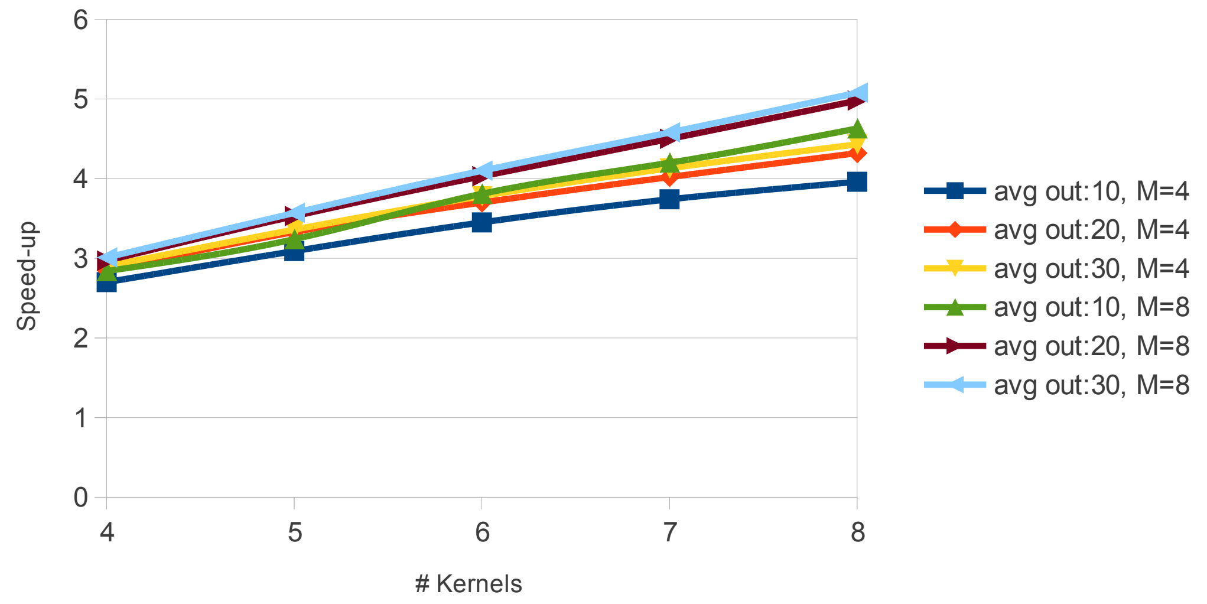
- Comparison (Performance/Area) against Bambu 0.9 and LegUP 3.0 (accelerators only)
  - Target device: Altera Cyclone II
  - Target frequency: 100 MHz
  - Target memory architectures: single and 4-banked shared memories

	Speedup, 1ch		*Speedup, 4ch*	
Benchmark	**vs LegUp**	**vs Bambu**	**vs LegUp**	**vs Bambu**
ADD	1.73	1.36	9.46	4.98
DOT PRODUCT	1.15	1.42	5.71	5.96
HISTOGRAM	3.14	2.74	9.68	5.63
LOS	2.77	2.29	5.43	5.62
MATRIXMUL	0.96	1.2	2.93	3.27
MATRIXTRANS	0.82	1.23	3.55	3.56
PERFECTHASH	1.23	1.57	4.56	3.87
Average	***1.62***	***1.77***	***4.29***	***4.7***

Performance Evaluation

	Area ratio, 1ch		*Area ratio, 4ch*	
Benchmark	**vs LegUp**	**vs Bambu**	**vs LegUp**	**vs Bambu**
ADD	1.05	2.62	1.8	4.51
DOT PRODUCT	1.34	3.5	2.02	5.28
HISTOGRAM	3.08	3.26	3.74	3.96
LOS	3.18	4.98	3.93	6.14
MATRIXMUL	1.14	2.75	1.85	4.45
MATRIXTRANS	1.07	4.3	1.8	7.25
PERFECTHASH	3.4	4.1	5.61	6.77
Average	***2.04***	***3.64***	***2.96***	***5.48***

Area Evaluation



BFS case study: speed-ups varying number of kernels/memory banks

Vito Giovanni Castellana, Antonino Tumeo
Pacific Northwest National Laboratory, HPC – Richland, WA, USA
vitogiovanni.castellana@pnnl.gov, antonino.tumeo@pnnl.gov

Fabrizio Ferrandi
Politecnico di Milano, DEIB – Milano, Italy
fabrizio.ferrandi@polimi.it

U.S. DEPARTMENT OF ENERGY

www.pnnl.gov