IEEE 13th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC-2019)
Nanyang Executive Centre, Nanyang Technological University, Singapore
October 1-4, 2019

Designing Application-Specific Heterogeneous Architectures from Performance Models

Minh Thanh CONG - Irisa / Université de Rennes 1

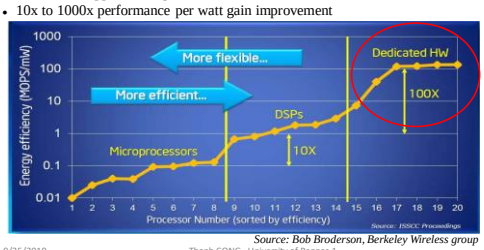
François CHAROT - Inria

October 03, 2019

UNIVERSITÉ DE RENNES 1 — UMR IRISA — Inria

Processor Design Trends

- Potential of application-specific accelerators
- 10x to 1000x performance per watt gain improvement



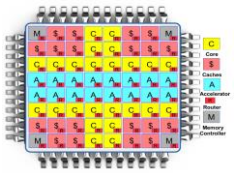
Source: Bob Broderson, Berkeley Wireless group

Designing Heterogeneous Architectures

How can we design such architectures?
- Simulation: a way to explore the design space
- Reusing knowledge and design tools of the multi/many-core domain

Existing works
- Standalone accelerator: Aladdin [Shao, 2014]
- System-level simulation: gem5+Aladdin [Shao, 2016], PARADE [Cong, 2015]
- Cycle-accurate simulators are still quite slow

What can we do?
- Speeding up the cycle-accurate simulator
- Building performance models of accelerators



Accelerator-Rich Architectures

Harnessing FPGAs for Simulation not Prototyping

Implement (parts of) the simulator on an FPGA
- Parallelism
- FPGA attachment technology
- FPGA is more and more powerful

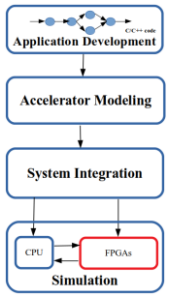
Speed of the simulator ↑

Performance model
- Functionally equivalent and logically isomorphic
- Allowing abstractions which simplify model development, enabling modularity

Complexity ↓

Our contribution

- A methodology for generating performance models of accelerators on FPGAs
- A cycle-accurate simulator targeting CPU-FPGA platforms with the goal of speeding up the simulation

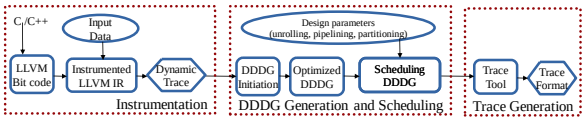


Outline

1. Accelerator modeling flow
   - ➢ Flow explanation by an example
2. System Integration
   - ➢Simulation framework
   - ➢Experimental study
3. Conclusion & Future work

Accelerator modeling flow

- Trace-based simulation
- Constructing a dynamic data dependence graph (DDDG)
- Optimizing and scheduling the graph
- Transforming the scheduled graph into compact instructions



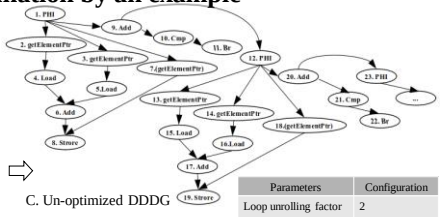
Flow explanation by an example

```
for(i=0; i<N; ++i)
  c[i] = a[i] + b[i];
```

A. C Code

```
0. r0=0  //i = 0
1. r4=load (r0 + r1) //load a[i]
2. r5=load (r0 + r2) //load b[i]
3. r6=r4 + r5
4. store(r0 + r3, r6) //store c[i]
5. r0=r0 + 1  //++i
6. r4=load(r0 + r1) //load a[i]
7. r5=load(r0 + r2) //load b[i]
8. r6=r4 + r5
9. store(r0 + r3, r6) //store c[i]
10.r0 = r0 + 1  //++i
…
```

B. IR Trace:

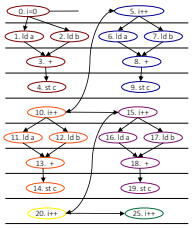


C. Un-optimized DDDG

Parameters	Configuration
Loop unrolling factor	2
Loop pipeline	Off
Memory partitioning	Cyclic, factor 2

Accelerator Model Targeting FPGAs

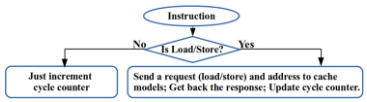
- *A factor of 2 loop iteration parallelism, partitioning factor 2, and without loop pipelining*



D. Scheduled DDDG

	Instruction	Description
Nodes	7 bits: Op_code 1 bit : Is new cycle	
Address	32 bits	op_code = load/store

E. Accelerator instructions

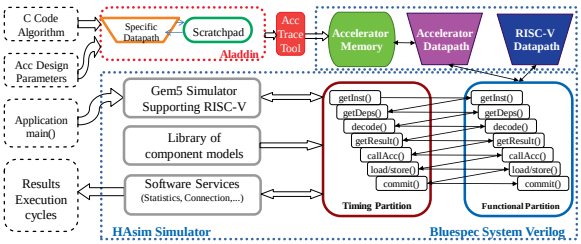


F. Timing model of accelerator

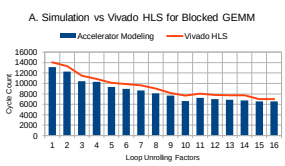
Outline

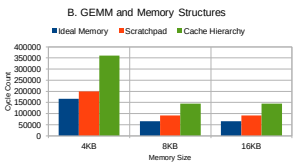
1. Accelerator modeling
   - ➢ Flow explanation by an example
2. System Integration
   - ➢Simulation framework
   - ➢Experimental study
3. Conclusion & Future work

Simulation Framework



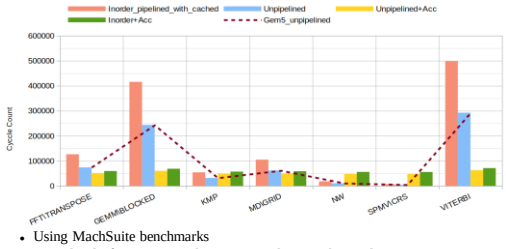
Framework Validation Using Blocked GEMM





- The latency decreases when the loop unrolling factor increases.
- The performance is very close to Vivado HLS.
- The number of cycles increases as the complexity of memory grows.

Architecture Exploration



- Using MachSuite benchmarks
- Same level of precision as the gem5 simulator is observed.

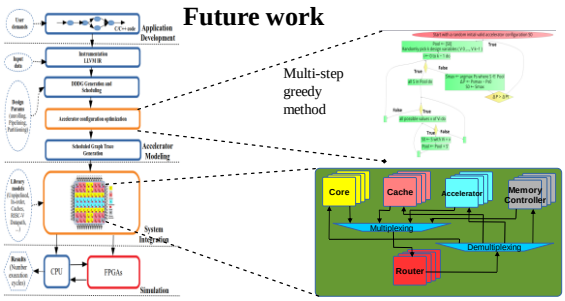
Outline

1. Accelerator modeling
   - ➢ Flow explanation by an example
2. System Integration
   - ➢ Simulation framework
   - ➢ Experimental study
3. Conclusion & Future work

Conclusion

- Generation of accelerator models targeting FPGAs
  - ➢ Exploited in HAsim-based simulation framework
- Simulation framework allowing architecture exploration
  - ➢ Tested with single-core and accelerator models
  - ➢ Compared with other tools

Future work



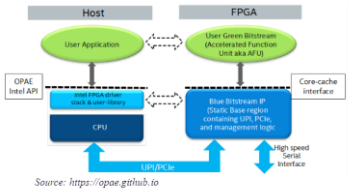
References

- **[Shao, 2014]** Y. S. Shao, B. Reagen, G. Wei, and D. Brooks, "Aladdin: A pre-rtl, power-performance accelerator simulator enabling large design space exploration of customized architectures," in 2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA), pp. 97–108, June 2014.
- **[Shao, 2016]** Y. S. Shao, S. L. Xi, V. Srinivasan, G. Y. Wei, and D. Brooks, "Codesigning accelerators and soc interfaces using gem5-aladdin," MICRO, 2016.
- **[Cong, 2015]** J. Cong, Z. Fang, M. Gill, and G. Reinman. PARADE: A cycle-accurate full-system simulation platform for accelerator-rich architectural design and exploration. In Proceedings ofthe IEEE/ACM International Conference on Computer-Aided Design, ICCAD '15, 2015.
- **[Chen, 2016]** Yu-Ting Chen, Jason Cong, Zhenman Fang, and Peipei Zhou. 2016. ARAPrototyper: Enabling Rapid Prototyping and Evaluation for Accelerator-Rich Architecture (Abstact Only). In Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA '16). ACM, New York, NY, USA, 281-281.
- **[Pellauer, 2011]** M. M. I. Pellauer. HAsim : cycle-accurate multicore performance models on FPGAs. Thesis, Massachusetts Institute of Technology, 2011.
- **[Intel]**https://www.altera.com/solutions/acceleration-hub/overview.html
- **[Xilinx]**https://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html

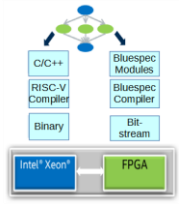
Questions

Intel Xeon-FPGA platform

- The Intel® Xeon®-FPGA tightly-coupled FPGA platform
- Minimizing communications bottlenecks



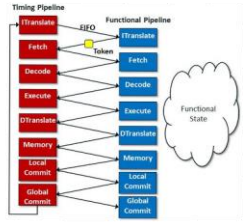
Source: https://opae.github.io



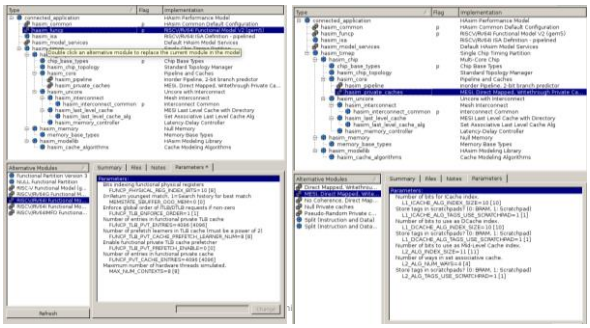
Programming tool flow

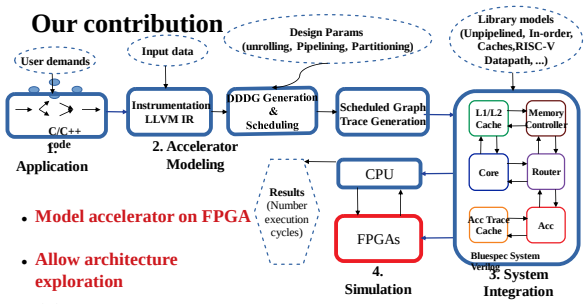
RISC-V processor models

- Support RISC-V 32, 64 bits Integer Instruction Set Architecture



- Build two timing models: unpipelined and in-order pipelined




Our contribution
User demands
Input data
Design Params
(unrolling, Pipelining, Partitioning)
Library models
(Unpipelined, In-order,
Caches,RISC-V
Datapath, ...)
C/C++
code
Instrumentation
LLVM IR
DDDG Generation
&
Scheduling
Scheduled Graph
Trace Generation
L1/L2
Cache
Memory
Controller
Core
Router
Acc Trace
Cache
Acc
Bluespec System
Verilog
CPU
FPGAs
Results
(Number
execution
cycles)
1.
Application
2. Accelerator
Modeling
3. System
Integration
4.
Simulation
• Model accelerator on FPGA
• Allow architecture
exploration
9/25/2019
Thanh CONG - University of Rennes 1
22