Co-synthesis of FPGA-Based Application-Specific Floating Point SIMD Accelerators

Andrei Hagiescu and Weng-Fai Wong

National University of Singapore
Iterative fitting

FPGA with embedded processor

Application

(1) ▶ constraint

(2) ◀ constraint

execution time

2
What IF some compilation decisions are postponed?
What IF we carry semantic information?
Virtualized FP SIMD

- SIMD floating point instructions
- *Disconnect* semantics from implementation → folding

\[ \forall i = [1, n] \quad a_i = b_i + c_i d_i \]
Flexible SIMD instructions

Integer SIMD

CPU

Slow CPU

Tightly connected instructions
Flexible SIMD instructions

Integer SIMD

CPU

Tightly connected instructions

Slow CPU

Tensilica Xtensa

VESPA
Flexible SIMD instructions

Integer SIMD

CPU

Slow CPU
Tightly connected instructions

versus

Floating point SIMD

CPU

Fast CPU
Interface overhead

Tensilica Xtensa
Extending AltiVec

- More ILP encapsulated in each vector instruction
- Extension relies on the generalizing the patterns

![Diagram of vector register file]
Extending AltiVec

- More ILP encapsulated in each vector instruction
- Extension relies on the generalizing the patterns
Compilation flow

Program compilation & vectorization

- Loop 1
- Loop N
- Source code

Vectorization

- x4/x8/x16 SIMD loop versions

SIMD DSE algorithm

- All variants of all vectorizable loops in all compilation units of app

Performance analyzer

- x4/x8/x16 SIMD loop versions

Synthesis

- HDL glue logic
- FPGA synthesis
- Library of HDL blocks

Generate hardware:
- ✓ Global optimization
- ✓ Pareto-optimal performance
- ✓ Non-iterative algorithm

Include correct loop versions (LTO)

Vector length undecided.
Types and number of execution units undecided.

Vector length of each loop determined.
Types and number of execution units determined.
Implementing SIMD Instructions

- Instruction blocks fold the execution units as needed
Hardware

- Cohabitation of Xilinx scalar FP and our SIMD extension
- Shared register file between all SIMD vector lengths
  - Multiport: multiple shadow copies
- Folding the bulky SIMD instructions
  - Arithmetic
  - Permutation
  - Loads / stores
- Folding parameters:
  - Instruction blocks
  - Execution units
Hardware DSE algorithm

- **Intuition:**
  - Reduce execution units → free area → evaluate new designs
  - Max achievable performance changes monotonically
- **Leverage nature of exploration to evaluate a small set of design points**
Results

- Execution units (E) vs. Instruction blocks (IB)
- Speedup trend

Resources for E and I (LUTs):
- 'qmr' design points
- 'linear' design points
- 'vecmat' design points

- Resources for IB (LUTs):
- 'qmr' design points

Minimum resources:
- x4 without folding
- x4/x8/x16 mix

Graph showing speedup for different design points and resource utilization.
Conclusions

- Fully automated non-iterative toolchain
- Folding to match resource constraints
- Improved energy consumption
  - 22% – 57% of scalar Xilinx FP - measured

- Future directions:
  - Partial reconfiguration
  - Operator fusing
Questions?