Exposing ILP in Custom Hardware with a Dataflow Compiler IR
Ali Mustafa Zaidi (Author), David Greaves (Supervisor)
ali-mustafa.zaidi@cl.cam.ac.uk, david.greaves@cl.cam.ac.uk

Core Question: Can we Achieve Superscalar Performance without Superscalar Overheads?

Poor Wire Scaling [1]
• Conventional Processors rely on centralized buses & memories, which leads to:
  - High wiring complexity,
  - Poor Scalability,
  - Low Computational Density,
  - High Energy Overheads,
  - Diminishing Returns on resources.

NEED FOR MODULAR, EXPLICITLY COMMUNICATION-CENTRIC ARCHITECTURES

The 'Dark Silicon' Problem [2]
• End of performance scaling with Moore's Law due to:
  - Amdahl's Law: Insufficient parallelism in applications.
  - Utilization Wall: Only a diminishing fraction of chip resources can be active in order to meet the power budget.

NEED FOR HIGHSEQUENTIAL PERFORMANCE, AS WELL AS HIGH ENERGY EFFICIENCY

Spatially Programmed Architectures [3]

PROS
• Scalable, decoupled, Communication-centric, high Computational Density.
• 10-1000x improvements in Efficiency & Performance [4].

CONS
• Programmability: difficult to design and implement – often requires low level HW design knowledge.
• Amenability: poor performance on irregular, control-flow intensive, 'general-purpose' code [5].

Possible Answer

Conventional Processors employ:
• Aggressive Control-Flow Speculation, AND
• Dynamic, Out-of-order execution, In order to achieve high ILP on such code [6].

Despite their advantages, Spatial Architectures are not currently suitable for general-purpose computing, since the cost/benefit trade-off would not be justifiable.

Our Solution: When compiling for spatial hardware, instead of [CDFG IR + Statically Scheduled Execution], use our [VSFG IR + Dynamically Scheduled Execution].

Our Solution: VSFG IR + Dataflow

Motivation

for (i = 0; i < 100; i++)
bar();

if (A[i] > 0) foo();

Speculation
branches execute speculatively by default

Guarded Actions
• E.g. Custom & Reconfigurable Hardware

NEED FOR MODULAR, EXPLICITLY COMMUNICATION-CENTRIC ARCHITECTURES

VSFG + dynamic execution scheduling enables Aggressive Speculation, Control Dependence Analysis, even Multiple Flows of Control [8]!

HLS Toolchain

Results

VSEF + Dataflow Execution achieves 1.12x speed-up over LegUp without unrolling, and 1.55x speed-up with loop unrolling, (with max speedup of 4.05x). Performance approaches or exceeds simulated Nehalem Core 17.

Peak Performance at 3-4x energy cost of LegUp, due to Speculation Overhead + increased resources (and their Clock + Static power). This is still 0.25x the Energy cost for the in-order Altera Nios II f processor.

Future Work: Further Energy / Performance improvements possible through Alias Analysis for Memory Level Parallelism & Partitioning, & other optimizations.

Future Work: Build Coarse-Grained Spatial Processor to replace the FPGA – achieving higher efficiency, density and clock frequencies (e.g. [3]).

References

Conclusions

Methodology

Frontend: VSFG IR + Dataflow

- CHStone benchmark Suite [9], + 2 more: neural-net simulator, and matrix transpose. Compiled to LLVM IR with -O2 flags; no Link-time opts.
- 3 configurations tested: VSFG_0 (no unrolling), VSFG_1 (all loops unrolled once), & VSFG_3 (all loops unrolled three times).
- Compared with LegUp 2.0 (no op-chaining), and Altera Nios II f for Altera Stratix IV GX FPGA, Nehalem Core 17 cycle counts from Sniper Simulator [10].