## Polymorphic Architecture II

- PipeRench: A Reconfigurable Architecture and Compiler
- Space-Time Scheduling of Instruction-Level Parallelism on a Raw Machine

Joel Coburn

John Kim

## PipeRench

- co-processor for streaming multimedia
- fine-grain polymorphic architecture (FPGA-like)
  - Interconnect network of configurable logic and storage element
  - PE (processing element) – 8bit datapath, register file
  - Stripe – row of PE's representing a pipestage with local interconnects

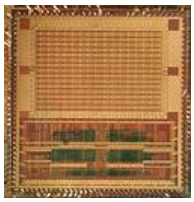
## PipeRench

- Pipelined reconfigurable architecture - Virtualize hardware
  - Compiler separated from hardware (physical stripe vs virtual strip)
  - restricts model of computation to pipelined datapath
- Configuration information read from the on-chip configuration store
- DIL(dataflow intermediate language) – single-assignment language with C operators

## Piperench Results

- Limited data comparison
- raw speedup of 10x-200x possible on various kernels (but I/O limited)
- 10x speedup on IDEA compared to general purpose and custom hardware

## PipeRench - hardware



0.18um

49 $mm^2$

120MHz

16 physical stripes/
128 virtual strips

## PipeRench

- Advantage
  - disadvantages of FPGA
    - forward compatibility
    - rapid reconfiguration
    - compilability
  - easy of design
- Disadvantage
  - Limited bandwidth between processor & main memory
  - Limited application
  - Attached processor

## Space-Time Scheduling

- Premise: Polymorphic architectures have resources with non-uniform access latencies
- Instruction scheduling is a spatial and temporal problem
- RAWCC – compiler for general-purpose sequential programs on the RAW machine
- Exploit ILP within basic blocks through space-time scheduling

## Basic-block Orchestration

- Transform basic block into a set of parallel operations across RAW tiles
  - Spatial scheduling: assignment of instructions to processing units
  - Temporal scheduling: instruction scheduling on individual tiles
  - Assignment of data to tiles
  - Communication across a mesh interconnect

## Control Orchestration

- Control flow must be communicated between all tiles
- Asynchronous global branching
  - Broadcast branch value
  - Each tile and switch performs branch without synchronization at end of its basic block
- Control localization
  - Treat a branch-containing code sequence as a single instruction
  - Execute different branches concurrently on different tiles

## Results

- Claims that RAW compiler exploits ILP (really DLP) across tiles for all benchmarks
  - Loops are unrolled enough times to distribute across all tiles
  - Small data sets to take advantage of low communication overhead
- RAW machine tolerates dynamic events
  - Dynamic event only affects the processor on which the event occurred

## Critique

- Strengths
  - Compiles general-purpose programs written in C and FORTRAN
  - Fully distributed processor provides scalable ILP and handles control flow

## Critique (2)

- Weaknesses
  - Only shown to be advantageous for programs with ILP
  - Compiler focuses on static references and does not examine dynamic references
  - Benchmarks are well suited to the RAW architecture

## Discussion Questions

- What is architecture decoupling?
- Why keep adding registers if you add ALU's?
- What is hardware virtualization?
- How much should the compiler know about the hardware?
- How much knowledge of the architecture should the programmer have?

## Discussion Questions

- Compare with the Smart Memories/TRIPS architecture
  - Which allow more flexibility/configurability?
  - Which architecture is more power efficient?
- What should be the configuration granularity architecture?
  - Fine-grained ( 8-bits)
  - Conventional 32 bit datapath
- Which architectures are better suited for the different parallelism (or different applications)?
  - DLP
  - ILP
  - TLP