Automated, Retargetable Back-Annotation for Host Compiled Performance and Power Modeling

Zhuoran Zhao, Suhas Chakravarty and Andreas Gerstlauer
Electrical and Computer Engineering, The University of Texas at Austin

Introduction

Motivation

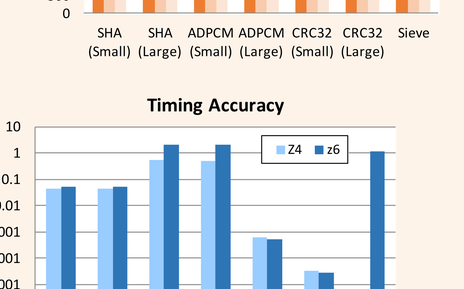
- Increasing design complexities
  - Rapid design space exploration desired
  - ➢ Performance and power validation
- Traditional simulation models
  - Instruction Set Simulator (ISS)
  - RTL/Gate level
  - ➢ Too slow or too inaccurate
- Modeling at higher abstraction levels
  - Fast and accurate
  - ➢ Host-compiled simulation

Challenges & Solutions
- Annotation granularity?
  - Speed vs. accuracy tradeoff
  - ➢ Block (BB) granularity
- Compiler optimizations?
  - Mapping between source and binary
  - ➢ Work with intermediate representation (IR)
- Dynamic architecture effects?
  - Pipelining state
  - Pairwise characterization

Host-Compiled Modeling

- Modeling above ISS level
  - Compile and execute application natively
  - Annotate application with target timing and power
  - Wrap with SystemC code for platform integration

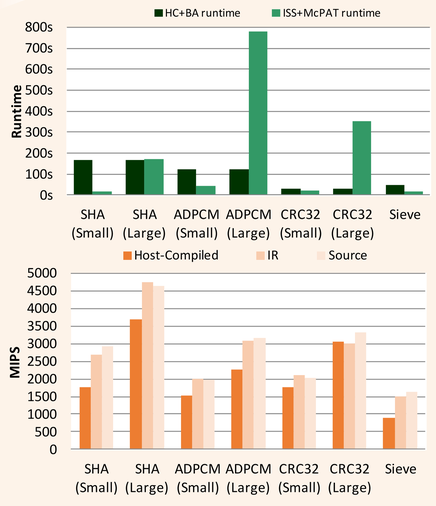
➢ Fast and accurate simulation to complement ISS



Experimental Results

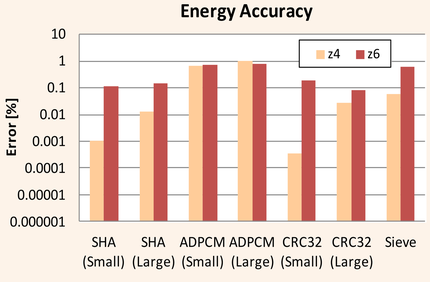
Speed Results

- Telecom & security applications [MiBench]
  - SHA, ADPCM, CRC32 & custom Eratosthenes' Sieve
  - Small and large data sets, 10 to 700 million instr.
- One-time back-annotation
  - 3min. to 3s BA runtime
- Host-compiled simulation vs. traditional ISS
  - ➢ 2000 MIPS vs. 0.8-1 MIPS
  - ➢ Close to source-level speeds



Accuracy Results

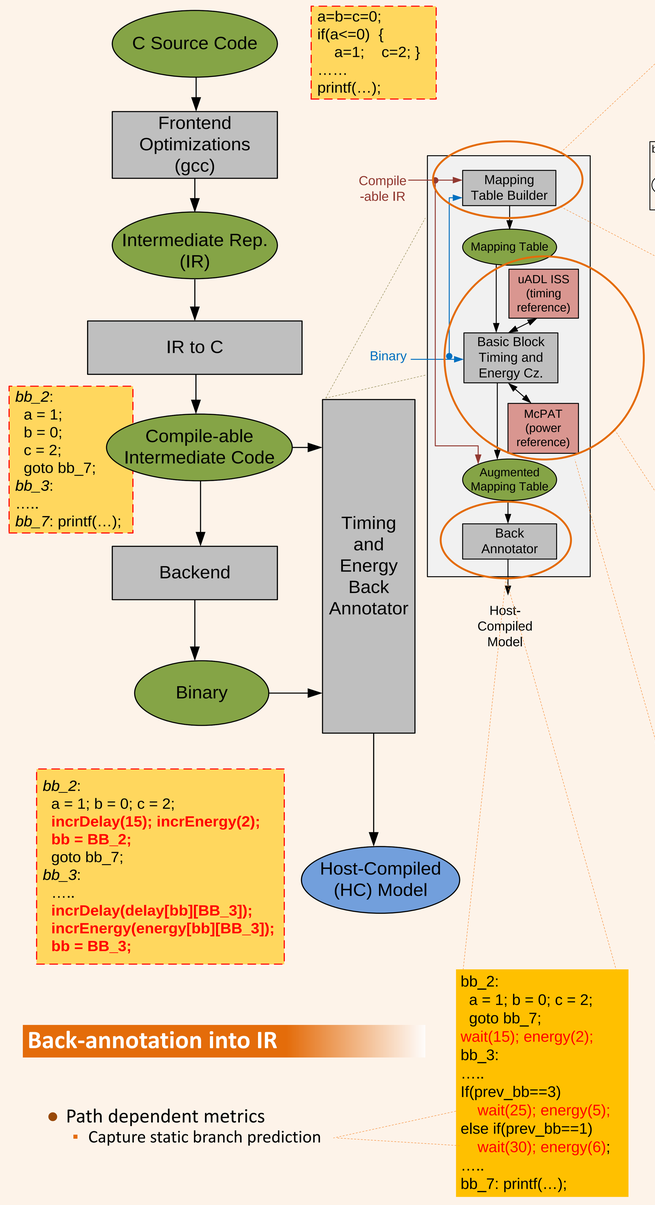
- Single- (z4-like) and dual-issue (z6-like) e200 PowerPC
  - No cache, static branch prediction
- Compare against cycle-accurate reference ISS+McPAT
  - ➢ >99% average timing and energy accuracy @ 2000 MIPS



Summary & Conclusions
- Retargetable power/performance back-annotation
- Running at source level speed with >98% accuracy
- Future work
  - Integrated other metrics into host-compiled simulation (thermal, reliability)

Retargetable Back Annotator

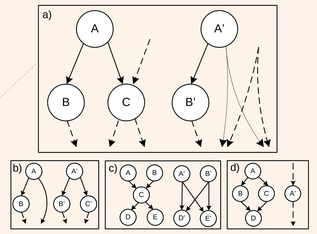
Retargetable Back-Annotation Flow



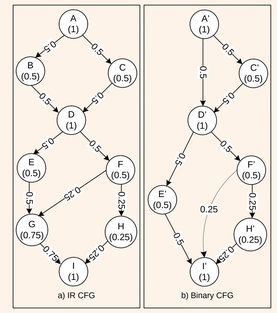
Back-annotation into IR

- Path dependent metrics
  - Capture static branch prediction

Binary-to-IR Mapping



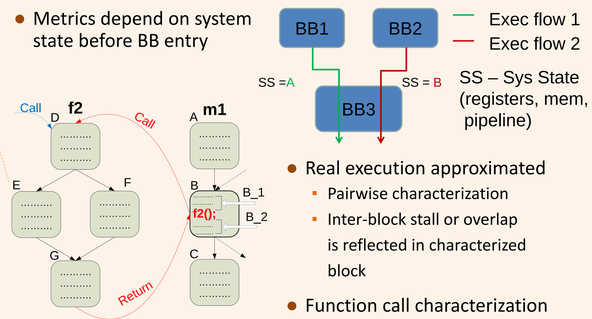
- Backend optimizations
  - Instruction scheduling
    - Blocks added/removed
    - Predicated execution
  - ➢ Control flow mismatches



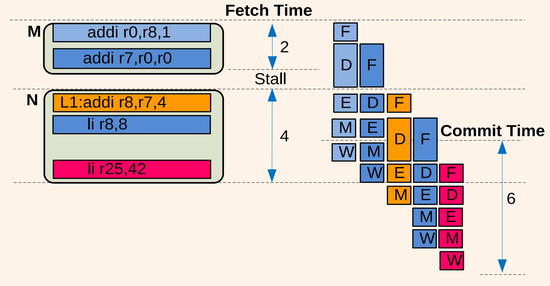
➢ Graph matching heuristic
- Synchronized, recursive depth-first traversal
  - Compatibility: loop and branch nesting levels
  - Cost: sum of unmatched nodes in subgraphs rooted at node
  - Return least-cost mapping between all successors (incl. skips)
- Resolve ambiguities using debug information

Basic Block Characterization

- Metrics depend on system state before BB entry



- Real execution approximated
  - Pairwise characterization
  - Inter-block stall or overlap is reflected in characterized block
- Function call characterization



Technology Transfer

- Reports and publications
  - S. Chakravarty, A. Gerstlauer. "Performance and Power Modeling Case Study", Report, Jan. 2011.
  - S. Chakravarty, A. Gerstlauer. "ADL Driven Back-Annotation and Simulation Methodology for Host Compiled Performance and Power Modeling", Report, July 2011.
  - S. Chakravarty, A. Gerstlauer. "Host Compiled Performance and Power Modeling of Embedded Systems," SRC TECHCON, August 2011.
  - S. Chakravarty, A. Gerstlauer, "Automated, ADL-Driven Generation of Host-Compiled Platform Models," Report, December 2012.
  - S. Chakravarty, Z. Zhao, A. Gerstlauer, "Automated, Retargetable Back-Annotation for Host-Compiled Power and Performance Modeling," *CODES+ISSS*, Montreal, Canada, September 2013.
- Industrial interactions
  - Leveraging Freescale's open-source ADL technology
    - http://opensource.freescale.com/fsl-oss-projects/
  - Regular (bi-weekly) project discussions (Freescale, TI)
  - Internships at Freescale (6/2013-8/2013)
    - Automated discovery of ADL based cycle-accurate reference models