PrEsto: An FPGA-Accelerated Power Estimation Methodology for Complex Systems

Dam Sunwoo, Gene Wu, Nikhil Patil, Derek Chiou
Electrical and Computer Engineering
The University of Texas at Austin
Power Modeling

- Detailed power simulations are extremely slow
  - Cannot run realistic benchmarks
  - Results may not be representative due to short run
  - “Spreadsheet power modeling” often used instead

- **Hardware-accelerated** Simulation
  - Dramatically speeds up generation of toggle activity
  - Would be useful to run power models at the same speed while maintaining high accuracy
Accelerating Power Modeling

- Finds mapping between signal values and power
  - Identifies key contributor signals to power
- **Systematic and automated** method to construct power models
- High accuracy
  - Within 2~3% of gate-level for **average** power
  - Within 6% of gate-level for **cycle-by-cycle** power
- FPGA implementation for acceleration
  - Several orders of magnitude faster than detailed simulation

**PrEsto (PoweR ESTimatOr)**
Agenda

- Motivation
- **Automated Power Model Generation**
- Integration with FAST Simulators
- Experimental Results
- Related Work and Conclusions
Linear Model of Dynamic Power

\[ P_m = c_0 + c_1 s_1 + c_2 s_2 + \ldots + c_n s_n + c_{n+1} (s_1 s_2) + c_{n+2} (s_1 s_3) + \ldots + c_{2n} (s_1 s_2 \ldots s_n) \]

- \( s_i \): signals in module, \( n \): num of signals,
- \( c_i \): coefficients obtained by regression
- Includes “crosses” of factors
- Total number of terms: \( 2^n \)
- More signals → Better accuracy, Higher complexity
Reducing Complexity

- Use only high-level signals
  - E.g., signals in architectural simulators
- Use Hamming Distance from cycle to cycle for aggregate signals (e.g. data/addr buses)
- Limit maximum number of factors per cross-term in linear model
  - At most ‘two control signals + one bus’ per term
- Limit total number of terms in the linear model
Automated Power Model Generation (APMG)

Start

Construct Linear Model

For each term, calculate MaxVal = coeff \times \max(\text{HD}(t))

Sort terms by MaxVal

Select Top N terms

Coverage High?

Refine benchmark / Increase N

VCD/Power from Training Benchmark

Done

Typical term:
\[ c \cdot \text{HD}(b_i) \cdot s_j \cdot s_k \]

Maximum # of terms per linear model

Coverage Threshold
Using PrEsto with Data Generated by Other Methods

- PrEsto automatically generates fast power models using data obtained elsewhere
- Data can be generated by
  - RTL, Gate-level, Layout, SPICE, etc.
  - Analytical models such as Wattch or CACTI
- Accelerates and/or automatically extrapolates from their results
Agenda

- Motivation
- Automated Power Model Generation
- Integration with FAST Simulators
- Experimental Results
- Related Work and Conclusions
FAST: FPGA-Accelerated Simulation Technologies


- Functionality
- ISA / Peripherals
- ~No Timing

- Timing
- Microarchitectural Details
  - Caches, scheduler, ROB, rename, BP, …
- ~No Functionality
● FM/TM are placed on platforms where each model runs well
  ● FM on microprocessors, TM on FPGAs
● Communication latency tolerated by loosely-coupled and parallel execution

FAST: FPGA-Accelerated Simulation Technologies
Incorporate Power Models into FAST Simulators

- Most other known FPGA-based power estimation works map entire RTL to FPGA
  - FAST TM is much smaller than full design
  - FAST TM allows multiple FPGA cycles to model single target cycle
Nallatech ACP Platform Block Diagram

- Xilinx Virtex5 LX110 (+ LX330s)
- Communicate over FSB (1066 MHz)
  - Snoop FSB to keep Sync Region coherent
Linear Power Models for FPGAs

- 7 FPGA cycles at 133 MHz including enqueue/dequeue
- Multipliers can be implemented with FPGA DSP slices or adders/shifters
Agenda

- Motivation
- Automated Power Model Generation
- Integration with FAST Simulators
- **Experimental Results**
- Related Work and Conclusions
Experiment Setup

- LEON3 (SPARC V8, 7-stage in-order pipeline)
- ARM Cortex-A8 (Dual Issue, in-order)

Diagram:
- Libraries
- Target RTL code
- Synopsys Design Compiler
- Gate-level Netlist
- Training Benchmark
- ModelSim
- VCD
- Synopsys PrimeTime PX
- Gate-level Power
- PrEsto APMG
- Python + R
- Power Models
Experiment Setup (II)

- LEON3
  - Chartered 130nm Library
  - MiBench Benchmark Suite
  - Ran 1M cycles (due to gate-level power simulation speed)
- ARM Cortex-A8
  - TSMC 65nm Library, Post-Place&Route with parasitic caps
  - EEMBC Benchmark Suite
  - Ran 100K cycles (samples of 20K throughout benchmark)
    - Palladium platform to fast-forward
- Used same trained power models across benchmarks
Power Modeling Accuracy

- LEON3
Power Modeling Accuracy

- ARM Cortex-A8
Power Waveforms of LEON3

Snippet of dijkstra from MiBench suite (RMS error: 1.8%)
Power Waveform of ARM Cortex-A8

Snippet of aes from EEMBC suite (RMS error: 3.8%)
Simulation Speed

- PrEsto on FPGA running SPEC2000

- **Speedup**
  - Software implementation: ~40x over PrimeTime PX
  - FPGA: ~70,000x over PrimeTime PX for LEON3
  - Potentially up to 700,000x for ARM Cortex-A8
What about other *irregular* logic?

- Macromodeling [Gupta et al. TVLSI 2000]
  - Avg Input Signal Probability
    - How many ones?
  - Avg Input Transition Density
    - How many bits flip?
  - Avg Input Spatial Correlation Coeff
    - How close are the ones?

- Look-up Table populated using input vectors
- Efficient FPGA implementation (more details in paper)
Agenda

- Motivation
- Automated Power Model Generation
- Integration with FAST Simulators
- Experimental Results
- Related Work and Conclusions
Related Work

- Architectural Simulators
  - Wattch: Analytical models. Doesn’t use data
  - SimplePower: Empirical models. No control path

- FPGA-based Power Modeling
  - Power Emulation [Coburn+, DAC 2005]
    - Maps RTL to FPGA with power models
  - [Bhattacharjee+, ISLPED 2008]
    - Maps LEON RTL to FPGA
    - Off-line power models based on performance counters
Conclusions

- **PrEsto**
  - Automated method to generate efficient and accurate power models for early-stage design
  - Runs significantly faster than other tools while maintaining high accuracy even for cycle-by-cycle power
  - Useful to:
    - Early-stage architects: explore power-sensitive design space
    - RTL designers: evaluate alternative designs
    - Software developers: power-tune their programs
Backup Slides
Future Directions in Power Modeling

- Apply to more complex cores
  - Working on Freescale e500 design  
  [Gene Wu]
- Early-stage Power Modeling
  - Continuously increase power model accuracies by refining TM
- Thermal / Leakage
  - Repopulate coefficients and LUT entries of power models as temperature changes
Other FPGA-based Power Estimation

- All are *emulation-based*
  - Maps entire target RTL
  - Disadvantages:
    - Difficult to port most full-custom/ASIC designs to FPGAs due to very different structures
    - Difficult to integrate more accurate power models that take several FPGA cycles
      - Emulation assumes single cycle between pipeline registers
## Power Model Coverage

<table>
<thead>
<tr>
<th>Module</th>
<th>Coverage</th>
<th>Module</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>RegFile</td>
<td>95.35%</td>
<td>Regfile</td>
<td>92.37%</td>
</tr>
<tr>
<td>I/D Cache Data</td>
<td>93.67%</td>
<td>L2 Ctrl</td>
<td>85.87%</td>
</tr>
<tr>
<td>I/D Cache Tag</td>
<td>94.32%</td>
<td>LSQ Ctrl</td>
<td>78.83%</td>
</tr>
<tr>
<td>I-Cache Ctrl</td>
<td>92.08%</td>
<td>Decode</td>
<td>87.20%</td>
</tr>
<tr>
<td>D-Cache Ctrl</td>
<td>85.82%</td>
<td>Scoreboard</td>
<td>67.12%</td>
</tr>
<tr>
<td>Pipeline</td>
<td>84.06%</td>
<td>BrPred Ctrl</td>
<td>74.12%</td>
</tr>
</tbody>
</table>
FPGA Resource Usage

- Resource usage of a Linear Model per module

<table>
<thead>
<tr>
<th># of terms</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slice Registers</td>
<td>99 (0.05%)</td>
<td>163 (0.07%)</td>
<td>227 (0.10%)</td>
<td>317 (0.15%)</td>
<td>569 (0.27%)</td>
</tr>
<tr>
<td>Slice LUTs</td>
<td>284 (0.13%)</td>
<td>580 (0.27%)</td>
<td>899 (0.43%)</td>
<td>1464 (0.70%)</td>
<td>2915 (1.40%)</td>
</tr>
</tbody>
</table>

(Percentages of Xilinx Virtex-5 LX330)
Modeling Flow

Start

Existing Component?

Yes

Run RTL / Gate-level Power Estimation

PrEsto APMG

Validate Power Model

Accurate enough?

No

Yes

No

All models done?

Library

New Component

Divide component into building blocks in library or use analytical models

Create power model by using/extrapolating library models

Yes

Done
Macromodeling Inputs

- **Avg Input Signal Probability**
  - How many ones?
  
  \[ P_{in} = \frac{1}{n} \sum_{i=1}^{n} P_i \]

- **Avg Input Transition Density**
  - How many bits flip?
  
  \[ D_{in} = \frac{1}{n} \sum_{i=1}^{n} D_i \]

- **Avg Input Spatial Correlation Coeff**
  - How close are the ones?
  
  \[ SC_{in} = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \mathbb{P}\{x_i = 1, x_j = 1\} \]
Macromodels for FPGAs

- 3-bits per statistic: 9-bit index
- 512x32 Power Look-up Table (1 BRAM)
Macromodeling Accuracy

- Separate experiments on ISCAS-85 circuits
- Synopsys PrimeTime PX with TSMC 0.13um
- 2000 random input vectors
  - Compared against gate-level power estimates for same vectors
Modeling Irregular Logic

- Macromodeling! [Gupta et al. TVLSI 2000]
- ISCAS-85 Benchmark circuits (ALUs, etc.)
- Compared against Synopsys PrimeTime estimates (TSMC 130nm library)
- Cycle-by-Cycle power modeled within 20%!
### FPGA Resource Usage

- **Macromodels**

<table>
<thead>
<tr>
<th>Input Width</th>
<th>32-bit</th>
<th>64-bit</th>
<th>96-bit</th>
<th>128-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slice Registers</td>
<td>117 (0.05%)</td>
<td>204 (0.09%)</td>
<td>283 (0.13%)</td>
<td>366 (0.17%)</td>
</tr>
<tr>
<td>Slice LUTs</td>
<td>275 (0.13%)</td>
<td>431 (0.20%)</td>
<td>585 (0.28%)</td>
<td>746 (0.35%)</td>
</tr>
<tr>
<td>Block RAMs</td>
<td>1 (0.34%)</td>
<td>1 (0.34%)</td>
<td>2 (0.69%)</td>
<td>2 (0.69%)</td>
</tr>
</tbody>
</table>

(Percentages of Xilinx Virtex-5 LX330)