FPGA-Based Power Models for Early System-Level Power Estimation

Dam Sunwoo¹, Hassan Al-Sukhni², Jim Holt², Derek Chiou¹

¹Department of Electrical and Computer Engineering
The University of Texas at Austin

²Freescale Semiconductor Inc.

(sunwoo@ece.utexas.edu)

Theme/Task: 1633.001
Power

• Power consumption is a key factor in modern processor design
• Architectural design decisions make great impact on power
  • Need better models to make better decisions!

• The Dilemma:
  • Architectural-level: Wattch, SimplePower, etc.
    • Still inaccurate
  • RTL/Gate-level: PowerTheater, Cadence RC, etc.
    • Requires full design
    • Can be very slow
FAST

- FPGA-Accelerated Simulation Technology (FAST)
- Partition simulator into:
  - Functional Model (FM)
    - simulates ISA, peripheral functionality
  - Timing Model (TM)
    - simulates timing, details of microarchitecture
- Place TM on hardware!
- Runs significantly faster without losing accuracy
FAST Bird’s-eye view

Host Processor

Functional Model (FM)

√

FPGA

Timing Model (TM)

Traces

Interface

Feedback

Dam Sunwoo
Power Models for FAST

• Identify **key contributor signals** to power consumption and create models based on them

• “Architectural Signals”
  - Available at high-level
  - Cache hit/miss signals, RegFile write data
  - Have more impact on power as they drive more logic and have larger fan-out
Experiments

• Applied proposed approach to Freescale z650 embedded processor
  • 32-bit Power ISA
  • 7-stage in-order pipeline
  • 32KB Unified L1 cache (8-way)
  • MMU
• Run PowerTheater on entire RTL
Example (Caches)

- Only used “Sum of Cache Bank Enable signals”
- Cache Reads: Access all ways in a set-associative cache
- Cache Writes: Access specific way only

Modeled Power

Cache Read

Cache Write

Dam Sunwoo
• Only used “Hamming Distance of Reg File Write Data”

• The spikes in power indicate more bits have flipped, resulting in more power consumption
Total Power (z650 core)

- Only used Cache / Register File signals
- All other modules are lumped as constant
- Linear model \((aX + bY + c)\)
- Modeled power tracks PowerTheater fairly accurately

Dam Sunwoo
Results
(Average Power)

- Less than 2% off of RTL PowerTheater estimation

Dam Sunwoo
Results
(Cycle-by-Cycle Power)

- Measured as *RMS average* of difference between PowerTheater and Modeled power
- *Less than 20% off!*

Dam Sunwoo
What about other *irregular* logic?

- Macromodeling [GuptaTVLSI2000]
- 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 carefully crafted input vectors
- Markov Chain Sequence Generator [LiuICCAD2002]
- Generate 2000 vectors per LUT entry

Dam Sunwoo
Macromodeling Results

- ISCAS-85 Benchmark circuits (ALUs, etc.)
- Compared against Synopsys PrimeTime estimates
- TSMC 130nm library
- Cycle-by-Cycle power modeled within \textit{20\%}!
Integration with Architectural Simulators

- Can use any architectural simulator
- Power models only use signals that are available at high-level
- Runs significantly faster than RTL simulation
- FPGA-Accelerated Simulation Technologies (FAST) Simulators
  - Power Models very suitable for FPGAs
    - Virtually no overhead on simulation performance
  - Even faster! (10 MIPS?)
Conclusion

- High-level Power Models
  - Can model average power very accurately (<2%)
  - Can model cycle-by-cycle power at reasonable accuracy (<20%)
  - Can run significantly faster than RTL tools
- Useful to not only hardware designers but also to software developers
Tech Transfer

- Industrial Liaisons
  - James C. Holt (Freescale)
  - Carl E. Lemonds (AMD)
  - Peng Yang (Freescale)

- Internships
  - IBM Austin Research Lab (2006)
  - Freescale (2008)

- Publications
Thank you!

• Questions?
A programmable processor's high overhead derives from overhead instructions, and exposing the pipeline. Algorithms are also evolving more rapidly, making it problematic to freeze them into hardwired implementations. Increasingly, embedded applications are demanding flexibility as well as efficiency.

Figure 1 shows that the processor spends most of its energy supplying data and supplying instructions. The 8-Kbyte instruction cache consumes the bulk of the energy, while fetching an instruction down the five-stage RISC pipeline. Thus, the energy required to supply data and instructions is 42%, while instruction supply accounts for 53%. The remaining 11% is attributed to other components such as cache and pipeline registers.

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instruction supply</td>
<td>42%</td>
</tr>
<tr>
<td>Cache</td>
<td>36%</td>
</tr>
<tr>
<td>Pipeline registers</td>
<td>26%</td>
</tr>
<tr>
<td>Cache array</td>
<td>21%</td>
</tr>
<tr>
<td>Cache tags</td>
<td>18%</td>
</tr>
<tr>
<td>Cache controller</td>
<td>9%</td>
</tr>
<tr>
<td>Pipeline registers</td>
<td>8%</td>
</tr>
<tr>
<td>Cache array</td>
<td>6%</td>
</tr>
<tr>
<td>Cache tags</td>
<td>4%</td>
</tr>
<tr>
<td>Cache controller</td>
<td>3%</td>
</tr>
<tr>
<td>Pipeline registers</td>
<td>2%</td>
</tr>
<tr>
<td>Cache array</td>
<td>1%</td>
</tr>
<tr>
<td>Cache tags</td>
<td>1%</td>
</tr>
<tr>
<td>Cache controller</td>
<td>1%</td>
</tr>
</tbody>
</table>

Table 1 lists each component's energy costs. Pipeline registers consume an additional 12 pJ, passing each word of data to control a 10-pJ arithmetic operation. Moreover, two words must be supplied and one consumed per instruction register file (IRF) as soon as the processor identifies the block to transfer—not as a cache, where hardware performs transfers. Here the 8-Kbyte data cache (array, tags, and control) accounts for 50% of the data supply energy. The 40-word multiported general-purpose register file accounts for 50 percent of the data supply energy. The ELM processor eliminates this word from the register file requires 17 pJ of energy. Two words must be supplied and one consumed per instruction cache, to reduce this number the processor uses a deeper hierarchy with explicit control, eliminating overhead instructions, and exposing the pipeline.

Stanford ELM processor [DallyIEEEComputers2008]
Why is “Reg File Write Data” important?

- Write Data is broadcast

- Although only one register is “write-enabled”, switching still occurs at all register inputs

- Write Data is globally routed
Benchmarks

- Profiled power model using one benchmark
- Ran 12 different benchmarks from standard benchmark suites
  - Dhrystone, Hiperstone, etc.
- Sampled 10,000ns from VCD dump
  - Simulating 10,000ns in PowerTheater takes around 4 hours
Modeling Flow

Start

Existing Component?

Yes

Run RTL / Gate-level Power Estimation

Identify key contributor signals

Calculate coefficients

Accurate enough?

Yes

Done

No

No

New Component

Divide component into building blocks in library

Library

Create power model by using/extrapolating library models

All models done?

Yes

No