Power Model

ESESC Tutorial

Speaker: Meeta Sinha

ESESC

Department of Computer Engineering, University of California, Santa Cruz
http://masc.soe.ucsc.edu
You will learn:
- High level view of ESESC power model
- Run a power simulation
- Use report.pl to view power numbers
- LibPeq
References

**CACTI**

**McPAT**

**LibPeq**
Outline

• Overview of the ESESC power model
  • High level description of the power model
  • Structure & components
  • Configuration
• Power Model Demo & Reported values
• LibPeq
• Code structure
• Dynamic Energy =

Event Counters \( \times \) Energy associated with each event.

- Read accesses
- Write accesses
- Miss buffer accesses
- Branch instructions
- ALU accesses

Etc.

- # memory ports
- Linesize
- Core tech node
- Cache Size
- Issue width

Etc.

Power Model

Dynamic Power
Power Model - Structure

ESESC Timing Model

Activity counters

ESESC Power Model

Modified McPAT [3]

Cache models
CACTI [1,2] based

Core models
Array based Structures
CACTI based

Execution Units
McPAT core model

Power Numbers

- Uses ESESC configuration instead of XML files to build models
- Uses GStats as activity counters
- Runs McPAT on the fly instead of traditional execution in the end.

Modified McPAT [3]

Cache models
CACTI [1,2] based

Core models
Array based Structures
CACTI based

Execution Units
McPAT core model
Renaming Unit (RNU)
- iFRAT
- iRRAT
- ifreeL
- fFRAT
- fRRAT
- ffreeL

Fetch Unit
- global BPT
- L1_local BPT
- chooser
- RAS
- BTB
- IB

Icache
- mmu-lllb
- ifu-icache

dcache
- mmu-dttl
- lsu-dcache
- lsu-dcachecc

Load Store Unit (LSU)
- lsu-LSQ
- lsu-LoadQ
- lsu-ssit
- lsu-lfst

Execution Unit (EXE)
- EXEU
- fp_u
- int_inst_window
- fp_inst_window

Register File
- IRF
- FRF

Reorder Buffer (ROB)

Array-based structures modeled by CACTI for McPAT
McPAT Core Model
• Enable or disable the power model
• Technology Parameters

ESESC Timing Model

ESESC Power Model

Activity counters

esesc.conf

Description of the architecture

simu.conf

pwth.conf

CACTI based components

McpAT core

Power Numbers

Translate ESESC GSTAT counters to McPAT understandable counters
Outline

• Overview of the ESESC power model
  • High level description of the power model.
  • Structure & components
  • Configuration

• Power Model Demo & Reported values
• LibPeq
• Code structure
Power Model - Demo

- Enable the power model and run a benchmark
- Use report.pl to view the power numbers
Reported numbers: report.pl

```plaintext

Sampled 0 (Procs 0)
  Rabbit Warmup Detail Timing Total KIPS
  KIPS    N/A    11653    525   527   6054   0.0%   49.7%   12.7%   37.6% : Sim Time (s) 16.529 Exe 1.921 ms Sim (1700MHz)
  Inst    0.0%   95.6%   1.1%   3.3% : Approx Total Time 64.854 ms Sim (1700MHz)

  0 : 31.074 : hybrid : 89.69% : (100.00% of 8.18%) : 94.10% : ( 89.23% of 35.13%) : 0.00%

  0 : 3270620 : 6410740 : 6410881 : 62.54% : 5.86% : 0.11% : 16.79% : 14.70% : 0.00% : N/A : SUNIT_AALU 1.32

Proc IPC uIPC Active Cycles Busy LDQ STQ IWin ROB Regs IO maxBr MisBr Br4Clk brDelay
  0 01.00 1.96 0.90 3264934 49.1 2.1 2.1 15.1 1.5 0.0 1.8 0.0 8.9 0.0 1.4

Cache Occ AvgMemLat MemAccesses MissRate (RD, WR, BUS) Dyn_Pow (mW) Lkg_Pow (mW)
  IL1(0) 0.0 3.2 2254695 0.93% (99.0%, 0.0%, 0.0%) 250 1
  ITLB(0) 0.0 3.3 2254376 0.82% (100.0%, 0.0%, 0.0%) 0 0
  DL1(0) 0.0 9.8 2510395 0.58% (98.9%, 98.3%, 0.0%) 1322 51
  L2(0) 0.0 51.6 36128 16.58% (85.7%, 26.9%, 0.0%) 0 77
  L3(0) 0.0 166.0 5998 97.22% (3.2%, 0.6%, 0.0%) 0 355
  MemBus(0) 0.0 141.9 5833 0.80% (100.0%, 0.0%, 0.0%) 0 36
  PTLB(0) 0.0 9.9 2509957 2.39% (97.6%, 0.0%, 0.0%) 0 0

CPU Power Metrics: (Dynamic Power,Leakage Power)
  Proc | RF (mW) | ROB (mW) | fetch (mW) | EXE (mW) | RNU (mW) | LSU (mW) | Total (mW)
  0    | 115 , 0    | 115 , 0    | 78 , 1    | 185 , 0    | 173 , 78    | 210 , 0    | 2.45 , 0.60

Power number for the caches
Power number per architectural block
```
Outline

• Overview of the ESESC power model
  • High level description of the power model.
  • Structure & components
  • Configuration

• Power Model Demo & Reported values

• Power Model options
  • LibPeq
  • Code structure
Power Model – LibPeq (alpha)

• What is LibPeq[4,5]?
  • Simplified analytical power model derived using statistical techniques on CACTI results.

• Why LibPeq?
  • CACTI initialization time very high.
  • CACTI covers a subset of design search space. Full exploration extremely time consuming.
  • Increasing complexity of microprocessors lead to exponentially increasing design search space.
LibPeq: Modeling Technique

• **LibPeq**: Pareto efficient power model.

• **Keywords:**
  • **Pareto Efficiency**: Optimal approach for analyzing tradeoff between parameters.
  • **Pareto Frontier**: Set of choices that are pareto efficient.
LibPeq: Modeling Technique

- **Plot**: Energy and Delay value from CACTI runs for a particular array based structure.
LibPeq - Structure

- ESESC Timing Model
- Activity counters
- ESESC Power Model

- esesc.conf
- simu.conf
- CACTI
- LibPeq
- peq.conf
- pwth.conf
- McPAT

Power Model

Power Numbers
• Running crafty with the default parameters without LibPeq

• Running crafty with the default parameters with LibPeq
Outline

- Overview of the ESESC power model
  - High level description of the power model.
  - Structure & components
  - Configuration
- Power Model Demo & Reported value
- Power Model options
- LibPeq
- Code structure
### Code Structure: Main files

<table>
<thead>
<tr>
<th>Files / Directories</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>simu/libsampler/Powermodel.*</td>
<td>Main hook to the ESESC power model</td>
</tr>
<tr>
<td>ppth/libmcpat</td>
<td>McPAT source code (modified to support ESESC)</td>
</tr>
<tr>
<td>ppth/libmcpat/XML_parse.cpp</td>
<td>Reading ppth.conf, translating ESESC GSTATS to McPAT counters</td>
</tr>
<tr>
<td>ppth/libmcpat/processor.cpp</td>
<td>McPAT models for various blocks in a processor</td>
</tr>
<tr>
<td>ppth/libmcpat/core.cpp</td>
<td>McPAT models for core components.</td>
</tr>
<tr>
<td>ppth/libpwrmodel</td>
<td>Wrapper for the power model</td>
</tr>
<tr>
<td>ppth/libpeq</td>
<td>Library to parse LibPeq equations.</td>
</tr>
</tbody>
</table>
Power Model

Code Structure: Flow Chart

- **esesc.conf**
- **simu/libsampler/PowerModel.***
  - Initialize McPAT structures
  - Stat counters created
  - Read pwth.conf
- **pwth/libpwrmodel Wrapper.cpp**
  - Create xml_parse structure
  - Compute power
- **simu/libsampler/PowerGlue.cpp**
- **Pwth/libmcpat XML_parse.cpp**
- **simu.conf**
- **pwth/libmcpat**
- **Processor.cpp**
- **Core.cpp**
Code Structure: Pwth.conf

# not showing all templates

[OoOPwrCounterTemplate]
template[0] = "stats[%d] = %s_core_total_instructions +%s:nCommitted +testCounter"

[inorderPwrCounterTemplate]
template[0] = "stats[%d] = %s_core_total_instructions +%s:nCommitted +testCounter"

[MemPwrCounterTemplate]
template[0] = "stats[%d] = %s_read_accesses +%s:readHit +%s:readMiss +%s:readHalfMiss +testCounter"

[MCPwrCounterTemplate]
template[0] = "stats[%d] = mc_read_accesses +L3:readMiss +testCounter"

- Component Category e.g IL1,DL1,L2,L3
  - McPAT counter
  - Gstat counters
  - Delimiter

- McPAT to Gstat translation for membus write access [MCPwrCounterTemplate]:
  - mc_write_accesses = L3:writeBack +L3:writeMiss
  - Component Category mapping in simu/libsampler/PowerGlue.cpp
Important Input Files

- **esesc.conf**: Enable/Disable Power Model
- **simu.conf**: Description of architecture
- **pwth.conf**: Translate GStat counters to McPAT understandable counters. (Don’t modify unless you know what you are doing)
- **peq.conf**: Contains equation for SRAM and caches. CAM equation and leakage equation can be added in future.
Questions?
(Backup slide) Power Model – LibPeq

- LibPeq
  - **Alpha** version
  - Analytical model
  - Developed from statistical analysis of thousands of CACTI simulations covering exhaustive design search space
  - Significantly faster

- Only models SRAM and cache structures

- Does not model leakage
Equations defined for array based structures

[SRAM_Small1]
dynamic = "exp(-4.982+2.196* ln(tech)+0.4961* ln(ports)-0.00986*
sqrt(size)+0.5464* ln(size)-0.016961* width+0.4027* sqrt(width))* (10^-9)"

[SRAM_Large1]
dynamic = "exp(-5.446+2.094* ln(tech)+0.886* ln(ports)+0.000458*
sqrt(size)+0.5296* ln(size)-0.011965* width+0.31001* sqrt(width))* (10^-9)"

Included by esesc.conf
Should not be modified (unless you know what you are doing!)
(backup) Code Structure: Core.cpp

- Important Functions:
  # not all components shown

```cpp
Core::Core(ParseXML* XML_interface, int ithCore_, InputParameter* interface_ip_)
:XML(XML_interface),
  ithCore(ithCore_),
  interface_ip(*interface_ip_),
  ifu (0),
  lsu (0),
  mmu (0),
  exu_ (0).
```

Sub-components

```cpp
void Core::computeEnergy(bool is_tdp)
```

```cpp
void Core::displayEnergy(uint32_t indent, int plevel, bool is_tdp)
```
void Processor::dumpStatics(ChipEnergyBundle *eBundle) {
    sprintf(name, "P(%d)_RNU", ii);
    tdp = (cores[i]->rnu->iFRAT->power_readOp.dynamic + 
          cores[i]->rnu->iRRAT->power_readOp.dynamic + 
          cores[i]->rnu->ifreeL->power_readOp.dynamic + 
          cores[i]->rnu->fFRAT->power_readOp.dynamic + 
          cores[i]->rnu->fRRAT->power_readOp.dynamic + 
          cores[i]->rnu->ffreeL->power_readOp.dynamic);
}

Power for RNU sub-components