Power Estimation of a C algorithm on a VLIW Processor

Nathalie Julien, Eric Senn, Johann Laurent, Eric Martin

LESTER, University of South Brittany, Lorient, France

Eric.Senn@univ-ubs.fr
C-level estimation WITHOUT compilation

```c
int test1(int IU, int JU, int KU)
{
    int i, j, k;
    for(i=0; i<IU; i++)
        for(j=0; j< JU; j++)
            for(k=0; k<KU; k++)
                A[k] = A[k-8];
        B[i][j] = B[i+1][j] + A[i];
    for(i=0; i<IU; i++)
        for(j=0; j<JU; j++)
            B[i][j] = B[i][j] + B[i+1][j];
}
```

Context

P ?

Complete power model
Power Estimation

**Gate level estimation:**
- very accurate but long simulation time
- RTL description needed

RTL description not available

**Instruction Level P. A.**
- accurate
- limited for VLIW processor
- compiler dependent
- memories and pipeline stalls not taken into account

**Functional Level P. A.**
- accurate & fast
- based on architecture analysis
- compiler independent
- memories and pipeline stalls taken into account
**Methodology: Model Definition**

**Algorithmic parameters**
- Parallelism, Processing units
- Cache miss, Pipeline Stalls...

**Configuration Parameters**
- Frequency, Memory Mode...

- Processor
  - FLPA
  - Measures
  - Model Definition
  - Parameters

\[ P = 4 \alpha + 1 \]
Methodology: Estimation Process

Configuration parameters with the application

Algorithmic parameters with prediction models

Assumptions on the compiler efficiency

Parameter values

C algorithm

Power Model

C-level Power Estimation

Estimation Process

Processor

FLPA

Measures

Model Definition

α = 0.5

P = 4α + 1

P = 3 W
TI C6x: Model Definition

TI TMS320C6201: VLIW processor
up to 8 instructions in parallel, deep pipeline (up to 11 stages),
4 memory modes: mapped, bypass, cache and freeze

FLPA: Functional-Level Power Analysis

α: parallelism rate
β: number of processing units
γ: cache miss rate
PSR: pipeline stall rate
F: clock frequency
MM: memory mode

α: parallelism rate
β: number of processing units
γ: cache miss rate
PSR: pipeline stall rate
F: clock frequency
MM: memory mode
Power consumption rule in **mapped mode**

\[
P_{\text{core}} = V_{DD} \times (a \beta (1-PSR) + b_m) F + \alpha (1-PSR) [a_m F + c_m] + d_m
\]

**measurements**: \( a = 0.64, a_m = 5.21, b_m = 4.19, c_m = 42.401, \) and \( d_m = 7.6 \)
Parameters extraction

\[ X = a + b; \]
\[ Y = c + d; \]
\[ for \ (i = 0; i < 10; i++) \]
\[ y[i] = c[i] * d[i+1]; \]
\[ Z = a + d; \]
\[ for \ (j = 0; j < 50; j++) \]
\[ \{ \]
\[ for \ (k = 0; k < 32; k++) \]
\[ tab[k] = h[k-1] + l[k+1] \]
\[ \} \]

Loop nests analysis

Local parameters prediction (\(\alpha, \beta\))

Global parameters (\(\alpha, \beta\)) : average of local values

Local parameters prediction (\(\alpha, \beta\))
Parameters extraction

For (i=0; i<512; i++) Y= x[i]*(h[i] + h[i+1] + h[i-1]) + y;

**Loop body**: 8 instructions = 4 LD, 4 OP

NFP = 1; NPU = 8

\[
\alpha = \frac{NFP}{NEP} \leq 1; \quad \beta = \frac{1}{8} \frac{NPU}{NEP} \leq 1
\]

<table>
<thead>
<tr>
<th>PREDICTION MODEL</th>
<th>EP1</th>
<th>EP2</th>
<th>EP3</th>
<th>EP4</th>
<th>(\alpha), (\beta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEQ</td>
<td></td>
<td>8 EP</td>
<td></td>
<td></td>
<td>0.125</td>
</tr>
<tr>
<td>MAX</td>
<td>2 LD</td>
<td>2 LD</td>
<td>-</td>
<td>-</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>4 OP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MIN</td>
<td>1 LD</td>
<td>1 LD</td>
<td>1 LD</td>
<td>1 LD</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td>4 OP</td>
<td></td>
<td>4 OP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATA</td>
<td>2 LD</td>
<td>1 LD</td>
<td>1 LD</td>
<td></td>
<td>0.33</td>
</tr>
<tr>
<td></td>
<td>4 OP</td>
<td></td>
<td>4 OP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NFP: Number of Fetch Packets
NPU: Number of Processing Units
NEP: Number of Execution Packets
Prediction models

- Program RAM/cache
- Data RAM
- Program/data buses
- EMIF
- CPU
- DMA
- PU1
- PU2
- PU3
- PU4

**Max model**
- Fully exploitation of the architecture

**Min model**
- Load/store never executed in parallel

**Data model**
- Load/store executed in parallel only on different data

**SEQ model**
- Instructions executed sequentially
## Results

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Measures</th>
<th>Estimation vs Measures (%)</th>
<th>Application</th>
<th>MM</th>
<th>INT/EXT</th>
<th>P (W)</th>
<th>SEQ</th>
<th>MAX</th>
<th>MIN</th>
<th>DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIR</td>
<td>MM&lt;sub&gt;M&lt;/sub&gt;</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td>4.5</td>
<td>-39%</td>
<td>+5%</td>
<td>-33%</td>
<td>+5%</td>
</tr>
<tr>
<td>FFT</td>
<td>MM&lt;sub&gt;M&lt;/sub&gt;</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td>2.65</td>
<td>-11%</td>
<td>+12%</td>
<td>-3%</td>
<td>-2.6%</td>
</tr>
<tr>
<td>LMS</td>
<td>MM&lt;sub&gt;B&lt;/sub&gt;</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td>4.97</td>
<td>+1%</td>
<td>+3%</td>
<td>+2%</td>
<td>+3%</td>
</tr>
<tr>
<td>LMS</td>
<td>MM&lt;sub&gt;C&lt;/sub&gt;</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td>5.67</td>
<td>-55%</td>
<td>+5.8%</td>
<td>-16%</td>
<td>+5.8%</td>
</tr>
<tr>
<td>DWT 64*64</td>
<td>MM&lt;sub&gt;M&lt;/sub&gt;</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td>3.75</td>
<td>-25%</td>
<td>+13%</td>
<td>-13%</td>
<td>-5.9%</td>
</tr>
<tr>
<td>DWT 64*64</td>
<td>MM&lt;sub&gt;M&lt;/sub&gt;</td>
<td>EXT</td>
<td></td>
<td></td>
<td></td>
<td>2.55</td>
<td>-10%</td>
<td>+3%</td>
<td>-5.9%</td>
<td>-3.5%</td>
</tr>
<tr>
<td>DWT 512*512</td>
<td>MM&lt;sub&gt;M&lt;/sub&gt;</td>
<td>EXT</td>
<td></td>
<td></td>
<td></td>
<td>2.55</td>
<td>-11%</td>
<td>+2.4%</td>
<td>-7%</td>
<td>-3.9%</td>
</tr>
<tr>
<td>EFR vocoder</td>
<td>MM&lt;sub&gt;M&lt;/sub&gt;</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td>5.08</td>
<td>-50%</td>
<td>+11%</td>
<td>-24%</td>
<td>+1%</td>
</tr>
<tr>
<td>MPEG decoder</td>
<td>MM&lt;sub&gt;M&lt;/sub&gt;</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td>5.82</td>
<td>-54%</td>
<td>+9.6%</td>
<td>-32%</td>
<td>-8%</td>
</tr>
<tr>
<td>Average error</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>32%</td>
<td>7.8%</td>
<td>17%</td>
<td>4.8%</td>
<td></td>
</tr>
</tbody>
</table>

- Estimation vs Measures < 8%
- Minimum and maximum bounds provided
Consumption "maps"

- Consumption maps for the EFR Vocoder

In mapped mode

In cache mode
PSR estimation

- PSR = NPS / NTC
  - NPS: number of cycles where the pipeline is stalled
  - NTC: total number of cycles

- NPS = NPS_τ + NPS_{bc} + NPS_γ
  - NPS_τ: external data access - NEXT - Data Mapping (C-level)
  - NPS_{bc}: internal data bank conflict - NCONFLICT - Data Mapping (C-level)
  - NPS_γ: program cache misses - NFRAME - Compilation (A-level)
## Complexity reduction

- Only a portion of the code is to be studied
- Optimization effort can be focussed

<table>
<thead>
<tr>
<th>Application</th>
<th># of code lines</th>
<th># of lines studied</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C</td>
<td>ASM</td>
</tr>
<tr>
<td>FFT</td>
<td>77</td>
<td>408</td>
</tr>
<tr>
<td>LMS</td>
<td>30</td>
<td>408</td>
</tr>
<tr>
<td>DWT 64*64</td>
<td>46</td>
<td>714</td>
</tr>
<tr>
<td>EFR</td>
<td>118</td>
<td>1323</td>
</tr>
<tr>
<td>MPEG</td>
<td>2267</td>
<td>8488</td>
</tr>
</tbody>
</table>
Conclusion

- Original and general approach validated on a VLIW DSP architecture
- Estimation of minimum and maximum bounds of an algorithm power consumption
- Fast and accurate power estimation at the C-level (error max = 8%)
- Refining at the assembly level (error max = 3%)
  - but compilation is needed then
Conclusion

• Co-design HW/SW, SOC
• High level abstraction decision
  – no compilation
  – no physical measurements
  – no development tools and evaluation boards
• Fast feedback on software performances
  – hot spots
  – pieces of code not suitable for compilation yet
• Complexity reduction
Current and Future works

- Development of an automatic tool in progress (available on-line before 2003)
- Extension of the power model library in progress (TI C55, ARM7)
- Execution time estimation for energy consumption
- Generic model for external memories