Guided Design Space Exploration of Heterogeneous MPSoCs

Bastian Ristau

1st Workshop on Mapping Applications to MPSoCs, Schloss Rheinfels
June 17, 2008
The Design Space Exploration Problem: Finding the Right System and Mapping

Application
(e.g. 802.11a)

MPSoC
(e.g. Tomahawk)

Objectives: Power, Performance, Flexibility, Area, ...
Platform Concept
Design Space Exploration Approach

Application Modeling → Parallelism Analysis

Parallelism Analysis → Guidance

Guidance → Automatic Mapping

Automatic Mapping → Evaluation

Evaluation → Architecture Modeling

Architecture Modeling → Getting 1st Architectures

Getting 1st Architectures → Mapping

Mapping → Modeling

Modeling → Application Modeling
Kind of Big Picture

Read PIM  
(Matlab, C, UML, …)

Play Around

Dump PSM  
(Simulink, SystemC, …)
Modeling

Concept: Nodes communicating via ports over edges

- Nodes = Tasks | Processors
- Ports = Input and output data | Ports
- Edges = Ctrl and data flow | Interconnect
Application Modeling

- Task = Fetch → execute → write back (no side effects)
- Loops & option blocks wrapped into single node

→ Multiple abstraction levels (nodes can contain nodes)
→ Graphs almost acyclic and in SSA form

- Limitations: Streaming
→ Moving to … Data Flow Graphs
Providing Mapping Options

Excel tables generated from application model

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Mapping Table for ieee80211aRxS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Please insert cycle counts for architectures in columns. (At least one entry in each white row required for DSE.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Level</td>
<td>Function</td>
<td>EVP</td>
<td>XCE</td>
<td>ARM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>ieee80211aRxS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>DecodingEpilog</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>uInitDecisionNode</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>Acquisition</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>AcquisitionInnerProlog</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>AcquisitionMain</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>AcquisitionMainInnerEpilog</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>AcquisitionMainInnerProlog</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>ieee80211aSRxAcquisition</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>Decoding</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>DecodingEpilog</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>uInitDecisionNode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>LTFProcessing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>3</td>
<td>LTFProcessingInnerEpilog</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>3</td>
<td>LTFProcessingInnerProlog</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>3</td>
<td>ieee80211aSRxLCE</td>
<td>108</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>3</td>
<td>LTFFFT</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Model outline**
- **Cycle Counts** (Power figures, etc.) provided by system designer
- Excluding suboptimal mappings by leaving cells blank

➔ Utilizing Knowledge of System Designer
Parallelism Analysis Basics

Parallelism profiles reveal distribution of parallelism.

parallelism profile

available parallelism

time

Useful for determining 1st architecture, if
- Profile is independent of concrete architecture
- Profile is independent of underlying schedule
Getting 1st Architectures

Using parallelism profiles for performance estimation

- **Profile for Unlimited Processor Case**
  - Available parallelism increases over time.
  - Number of processors fixed.

- **Profile for a Given Architecture**
  - Available parallelism distributed over time.
  - Number of processors fixed.
Parallelism Analysis Example
Parallelism Analysis Results

(Exemplary results for a task graph w/ 1000 tasks and two processor types)

relative speed-up curve  estimation error compared to TLM simulation
Mapping

Mapping = Assignment of graph elements to graph elements

- Node $\rightarrow$ (Node, start) (task & data transfer mapping)
- Edge $\rightarrow$ (Node, address) (data mapping)
- Port $\rightarrow$ Port (task & data transfer mapping)
Task Mapping

= Strip Packing

Currently implemented methods:

- Strip packing
- ASAP & ALAP w/o machine restrictions
- List ASAP, EDF & LL
TGFF Task Mapping Results

- Application: 32 nodes, 8 node types, max in/out degree = 2
- Architecture: 1 RISC, 2 DSPs
Comparison of ListEDF and ListASAP

ListEDF

Problem in ListASAP: Data liveness
Data Mapping

= Shelf Packing

- Fixed total height and width  
  → Minimize heights in each shelf
- Memory selection  
  → weighted shelf selection
- Interference  
  → “order” conditions
Data Transfer Mapping

1. Two graph transformations
   - Insert transfer nodes, if data transfer is needed
     - Task A → Task B
   - Insert task dependencies, if two tasks mapped to same resource

2. Task mapping (w/ given spatial task mapping)
Evaluation of Mapping Result

MS Project XML generated from mapping result

Visualization and analysis allows identification of bottlenecks and unveils possibilities for system improvement
Instruction Level Test Bench

**Applications:** Instruction Level

- Large Control & Data Flow Graphs

**MP-SoC:** SAMIRA DSP

- 21 Processors, 5 Memories
Example of Utilizing Mapping Result for System Improvement

**Load Analysis** (for various Signal Processing Algorithms)

<table>
<thead>
<tr>
<th>Functional Unit</th>
<th>Average Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>decoder</td>
<td>67%</td>
</tr>
<tr>
<td>salu1</td>
<td>42%</td>
</tr>
<tr>
<td>salu4</td>
<td>16%</td>
</tr>
<tr>
<td>sshift</td>
<td>15%</td>
</tr>
<tr>
<td>salu2</td>
<td>14%</td>
</tr>
<tr>
<td>smem</td>
<td>12%</td>
</tr>
<tr>
<td>smul</td>
<td>12%</td>
</tr>
<tr>
<td>vmem</td>
<td>9%</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Example of Utilizing Mapping Result for System Improvement

**Load Analysis:** High Load in Decoder

**Idea:** Add Second Decoder

**Result:**

![Graph showing scaled latency for various functions with given architecture, additional decoder, and theoretical min.]

- convserial
- dct2d88
- dct2d168
- fft648
- fft1288
- fft2568
- firserial
- firparallel
- iirserial
- iirparallel
- imserial

- **Function**
- **given architecture**
- **additional decoder**
- **theoretic min**

TU Dresden, 6/17/2008  Bastian Ristau  Slide 22
Passing Results to TLM

- Application Modeling
- Architecture Modeling
- Guidance
- Automatic Mapping
- Evaluation

- TaskC
- XML
- TLM Simulator
Next Steps

- Head towards multiple standards running in parallel

- Automate generation of task graphs