Platform-based Design and the First Generation Dilemma

By Jiang Xu and Wayne Wolf
Dept. of ELE Princeton University
April. 2002

Outline

♦ Background
♦ Typical system-level design methodology
♦ Platform-based design methodology
♦ First generation dilemma
♦ Conclusions
Background

♦ Growing gap between design complexity and productivity.
♦ System-level design tool is a solution.
♦ Some system-level design tools.
  - Cadence Virtual Component Co-design (VCC).
  - CoWare N2C.
  - Synopsys CoCentric System Studio.
  - Innoveda Visual Elite.
  - Elanix SystemView.

Typical System-level Design Methodology

♦ Besides specification, previous design experience is an important input for system-level design.
♦ HW/SW partitioning is converted in function mapping.
♦ Performance analysis plays a more important role in PBD.
  - The accuracy of early performance analysis significantly affects following design steps.
Platform-based Design Methodology

- Three key steps replace the typical HW/SW partitioning.
- Behavior modeling
- Architecture modeling eases PBD and IP reuse.
- Mapping uses trial-and-error method.
- Close cooperation with other tools.

Performance Analysis

- It takes place as early as in mapping step to get a quick estimation of a design.
- It helps timing requirements propagate through each design steps.
- Performance modeling is under surface.
  - Performance analysis is based upon performance models.
  - Accuracy of performance models is guaranteed by designers, so does accuracy of performance analysis.
Behavior Modeling

- Behavior model is used to capture specification details and develop HW and SW.
- Different languages are used.
  - Choosing programming style is an old and unsolved problem.
- White Box C is preferable to others.

<table>
<thead>
<tr>
<th>Language</th>
<th>Black Box</th>
<th>White Box</th>
<th>Clear Box</th>
</tr>
</thead>
<tbody>
<tr>
<td>C++, SPW, SDL, OMI</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Simulated</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Analyzed</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Synthesizing</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Outline

- Background
- Typical system-level design methodology
- Platform-based design methodology
- First generation dilemma
- Conclusions
Another View of the PBD Methodology

- Two steps for architecture: choosing and mapping.
- Decisions of choosing platform and IP cores and designing customized modules are based upon performance analysis results.
- Performance analysis can not be made without IP cores or customized modules.

First Generation Dilemma

- PBD assumes designers have enough IP cores and customized modules to build a platform.
- In many first generation designs, only a few IP cores and customized modules are in libraries.
- In the original PBD methodology, we can not:
  - Get performance results without enough IP cores and customized modules.
  - Choose IP cores and design customized modules without performance results.
### A Way Out of the Dilemma

- First decide an architecture, and assign estimated requirements to unavailable modules.
- Adjust the requirements using performance analysis in a trial-and-error fashion.
- Based upon the requirements purchase IP cores and design customized modules.
- May need several iterations to reach a final design.
- It is very helpful, if designers can get performance models of IP cores before buy them.

---

### Example -- A Multimedia Embedded Chip

<table>
<thead>
<tr>
<th>Component</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor1</td>
<td>Available</td>
</tr>
<tr>
<td>Processor2</td>
<td>Unavailable IP Core</td>
</tr>
<tr>
<td>SRAM</td>
<td>Available</td>
</tr>
<tr>
<td>SRAM Controller</td>
<td>Available</td>
</tr>
<tr>
<td>Central Bus</td>
<td>Available</td>
</tr>
<tr>
<td>Bridge</td>
<td>Available</td>
</tr>
<tr>
<td>DMA Controller</td>
<td>Available</td>
</tr>
<tr>
<td>Smart Monitor</td>
<td>Unavailable Customized Modules</td>
</tr>
<tr>
<td>Media Accelerator</td>
<td>Available</td>
</tr>
<tr>
<td>Peripheral Bus</td>
<td>Available</td>
</tr>
<tr>
<td>PCI Interface</td>
<td>Available</td>
</tr>
<tr>
<td>External Memory Interface</td>
<td>Available</td>
</tr>
<tr>
<td>GPIO</td>
<td>Available</td>
</tr>
</tbody>
</table>
Conclusions

♦ First generation dilemma

– PBD is not purely choosing-and-mapping, it still includes architecture design and module design.
– Easily accessed performance models will be very helpful.

♦ Behavior modeling is a critical step in PBD, and modeling language is still an important issue.

♦ PBD needs a systematic method to guarantee performance analysis accuracy.