4541.775

Topics on Compilers

Spring 2011
Syllabus

• **Instructor**  Bernhard Egger  
  bernhard@snu.ac.kr  

  **Office Hrs**  301 동 413 호 on Tuesdays, Thursdays 09:30 – 11:30 a.m.

• **Lecture**  302 동 106 호 on Mondays, Wednesdays 11:00 a.m. – 12:15 p.m.

  **Website**  http://aces.snu.ac.kr/~bernhard/teaching/4541.775/  
  **Language**  English

• **The Class**  This class gives an overview over optimizing compiler technologies and discusses compilation techniques for modern embedded architectures.

  **Format**  Mondays: lecture  
  Wednesdays: lecture/presentations  
  + bi-weekly assignments
Syllabus

• **Goals**
  - get an idea of what's going on in current compiler research
  - improve paper reading, discussion and presentation skills

• **Materials**
  the lecture does not follow one particular textbook, however, the first half of the lecture is based on “Optimizing Compilers for Modern Architectures” by Randy Allen and Ken Kennedy. Presentation slides and papers for assignments will be made available on the course website.

• **Coursework**
  the assigned coursework consists of small problem sets to encourage active understanding of the lecture. It will be handed out (bi-)weekly and discussed in class. Coursework needs to be turned in on the indicated deadline.
  You get two “free” 24-hour deadline extensions, after that a late submission results in a reduced grade for that assignment.

• **Exams**
  you may bring whatever you want as long as it's in paper form.
  No electronic devices whatsoever.
• **Presentations** each student will give two to three presentations of research papers. A presentation should last 20 minutes, then there will be Q&A plus a short discussion. A paper list will be mailed to you/posted on the course website until the next class. Unless you can convince me otherwise, the oral presentation and the materials must be in English.

• **Grading**
  
  coursework 15%
  presentations 40%
  midterm 20%
  final 25%
  participation bonus +5%

• **Honor Code** you must complete written assignments on your own. For the presentations you are permitted to use materials from books/the web as long as you indicate the source.
<table>
<thead>
<tr>
<th>Week</th>
<th>Monday</th>
<th>Wednesday</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (2/28)</td>
<td>Introduction</td>
<td></td>
</tr>
<tr>
<td>2 (3/7)</td>
<td>Dependence</td>
<td><em>no class (compensation: 4/6)</em></td>
</tr>
<tr>
<td>3 (3/14)</td>
<td>Dependence</td>
<td>Dependence Testing</td>
</tr>
<tr>
<td>4 (3/21)</td>
<td>Dependence Testing</td>
<td>Dependence Testing</td>
</tr>
<tr>
<td>5 (3/28)</td>
<td>Dependence-based Transformations</td>
<td>Dependence-based Transformations</td>
</tr>
<tr>
<td>6 (4/4)</td>
<td>Enhancing Parallelism</td>
<td>Presentation Block #1</td>
</tr>
<tr>
<td>7 (4/11)</td>
<td>Fine-Grained Parallelism</td>
<td>Presentation Block #1</td>
</tr>
<tr>
<td>8 (4/18)</td>
<td>Fine-Grained Parallelism</td>
<td>Coarse-Grained Parallelism</td>
</tr>
<tr>
<td>9 (4/25)</td>
<td>Coarse-Grained Parallelism</td>
<td>Mid-term Exam</td>
</tr>
<tr>
<td>10 (5/2)</td>
<td>Compiling for VLIW Architectures</td>
<td>Compiling for VLIW Architectures</td>
</tr>
<tr>
<td>11 (5/9)</td>
<td>Compiling for VLIW Architectures</td>
<td>Compiling for VLIW Architectures</td>
</tr>
<tr>
<td>12 (5/16)</td>
<td>Presentation Block #2</td>
<td>Presentation Block #2</td>
</tr>
<tr>
<td>13 (5/23)</td>
<td>Compiling for CGRA</td>
<td>Compiling for CGRA</td>
</tr>
<tr>
<td>14 (5/30)</td>
<td>Compiling for CGRA</td>
<td>Compiling for CGRA</td>
</tr>
<tr>
<td>15 (6/6)</td>
<td><em>no class (Memorial Day)</em></td>
<td>Presentation Block #3</td>
</tr>
<tr>
<td>16 (6/13)</td>
<td>Presentation Block #3</td>
<td>Final Exam</td>
</tr>
<tr>
<td>17 (6/20)</td>
<td>Wrap-up</td>
<td>Wrap-up</td>
</tr>
</tbody>
</table>
### Presentation Block #1:
**Data Dependencies, Coarse and Fine Grained Parallelism**

<table>
<thead>
<tr>
<th>Title</th>
<th>Published In</th>
<th>DOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>The program dependence graph in a software development environment</td>
<td>Softw. Eng. Notes 9, ‘84</td>
<td><a href="http://doi.acm.org/10.1145/390010.808263">http://doi.acm.org/10.1145/390010.808263</a></td>
</tr>
<tr>
<td>The program dependence graph and its use in optimization</td>
<td>TOPLAS’87</td>
<td><a href="http://doi.acm.org/10.1145/24039.24041">http://doi.acm.org/10.1145/24039.24041</a></td>
</tr>
<tr>
<td>Optimal loop parallelization</td>
<td>PLDI ‘88</td>
<td><a href="http://doi.acm.org/10.1145/53990.54021">http://doi.acm.org/10.1145/53990.54021</a></td>
</tr>
<tr>
<td>Dependence analysis for pointer variables</td>
<td>PLDI ‘89</td>
<td><a href="http://doi.acm.org/10.1145/73141.74821">http://doi.acm.org/10.1145/73141.74821</a></td>
</tr>
<tr>
<td>The program dependence web: a representation supporting …</td>
<td>PLDI’90</td>
<td><a href="http://doi.acm.org/10.1145/93542.93578">http://doi.acm.org/10.1145/93542.93578</a></td>
</tr>
<tr>
<td>Constant propagation with conditional branches</td>
<td>TOPLAS’91</td>
<td><a href="http://doi.acm.org/10.1145/103135.103136">http://doi.acm.org/10.1145/103135.103136</a></td>
</tr>
<tr>
<td>Efficient and exact data dependence analysis</td>
<td>PLDI’91</td>
<td><a href="http://doi.acm.org/10.1145/113445.113447">http://doi.acm.org/10.1145/113445.113447</a></td>
</tr>
<tr>
<td>Practical dependence testing</td>
<td>PLDI’91</td>
<td><a href="http://doi.acm.org/10.1145/113445.113448">http://doi.acm.org/10.1145/113445.113448</a></td>
</tr>
</tbody>
</table>
### Presentation Block #1:
**Data Dependencies, Coarse and Fine Grained Parallelism**

<table>
<thead>
<tr>
<th>Title</th>
<th>Published In</th>
<th>DOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data-centric multi-level blocking</td>
<td>PLDI’97</td>
<td><a href="http://doi.acm.org/10.1145/258915.258946">http://doi.acm.org/10.1145/258915.258946</a></td>
</tr>
<tr>
<td>Dynamic speculation and synchronization of data dependences</td>
<td>ISCA’97</td>
<td><a href="http://doi.acm.org/10.1145/384286.264189">http://doi.acm.org/10.1145/384286.264189</a></td>
</tr>
<tr>
<td>Cache-conscious structure layout</td>
<td>PLDI’99</td>
<td><a href="http://doi.acm.org/10.1145/301618.301633">http://doi.acm.org/10.1145/301618.301633</a></td>
</tr>
<tr>
<td>Data and memory optimization techniques for embedded systems</td>
<td>TODAES’01</td>
<td><a href="http://doi.acm.org/10.1145/375977.375978">http://doi.acm.org/10.1145/375977.375978</a></td>
</tr>
<tr>
<td>An empirical evaluation of chains of recurrences for array dep. testing</td>
<td>PACT’06</td>
<td><a href="http://doi.acm.org/10.1145/1152154.1152198">http://doi.acm.org/10.1145/1152154.1152198</a></td>
</tr>
<tr>
<td>Auto-vectorization of interleaved data for SIMD</td>
<td>PLDI’06</td>
<td><a href="http://doi.acm.org/10.1145/1133981.1133997">http://doi.acm.org/10.1145/1133981.1133997</a></td>
</tr>
<tr>
<td>Dryad: distributed data-parallel programs from sequential building ...</td>
<td>EuroSys’07</td>
<td><a href="http://doi.acm.org/10.1145/1272998.1273005">http://doi.acm.org/10.1145/1272998.1273005</a></td>
</tr>
<tr>
<td>Speculative parallelization using state separation and multiple value pred.</td>
<td>ISMM’10</td>
<td><a href="http://doi.acm.org/10.1145/1806651.1806663">http://doi.acm.org/10.1145/1806651.1806663</a></td>
</tr>
</tbody>
</table>
### Presentation Block #1:
**Data Dependencies, Coarse and Fine Grained Parallelism**

<table>
<thead>
<tr>
<th>Date</th>
<th>Presenter</th>
<th>Paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>April 6</td>
<td>Honggyu Kim</td>
<td>The program dependence graph and its use in optimization</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://doi.acm.org/10.1145/24039.24041">http://doi.acm.org/10.1145/24039.24041</a></td>
</tr>
<tr>
<td>April 6</td>
<td>In-goo Heo</td>
<td>Optimal loop parallelization</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://doi.acm.org/10.1145/53990.54021">http://doi.acm.org/10.1145/53990.54021</a></td>
</tr>
<tr>
<td>April 6</td>
<td>Gangwon Jo</td>
<td>Dependence analysis for pointer variables</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://doi.acm.org/10.1145/73141.74821">http://doi.acm.org/10.1145/73141.74821</a></td>
</tr>
<tr>
<td>April 13</td>
<td>Bertram Schmitt</td>
<td>Auto-vectorization of interleaved data for SIMD</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://doi.acm.org/10.1145/1133981.1133997">http://doi.acm.org/10.1145/1133981.1133997</a></td>
</tr>
<tr>
<td>April 13</td>
<td>Christine Wagner</td>
<td>Speculative parallelization using state separation and multiple value</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://doi.acm.org/10.1145/1806651.1806663">http://doi.acm.org/10.1145/1806651.1806663</a></td>
</tr>
</tbody>
</table>
### Presentation Block #2: Compilation for VLIW architectures

<table>
<thead>
<tr>
<th>Title</th>
<th>Published In</th>
<th>DOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Long Instruction Word architectures and the ELI-512</td>
<td>ISCA '83</td>
<td><a href="http://dx.doi.org/10.1145/800046.801649">http://dx.doi.org/10.1145/800046.801649</a></td>
</tr>
<tr>
<td>Software pipelining: an effective scheduling technique for VLIW machines</td>
<td>PLDI’88</td>
<td><a href="http://dx.doi.org/10.1145/53990.54022">http://dx.doi.org/10.1145/53990.54022</a></td>
</tr>
<tr>
<td>A Unified Modulo Scheduling and Register Allocation Technique for Clustered Processors</td>
<td>PACT’01</td>
<td><a href="http://portal.acm.org/citation.cfm?id=645988.674300">http://portal.acm.org/citation.cfm?id=645988.674300</a></td>
</tr>
<tr>
<td>Loop fusion for clustered VLIW architectures</td>
<td>LCTES/SCOPES’02</td>
<td><a href="http://dx.doi.org/10.1145/513829.513850">http://dx.doi.org/10.1145/513829.513850</a></td>
</tr>
<tr>
<td>Stream execution on wide-issue clustered VLIW architectures</td>
<td>LCTES’07</td>
<td><a href="http://dx.doi.org/10.1145/1254766.1254797">http://dx.doi.org/10.1145/1254766.1254797</a></td>
</tr>
<tr>
<td>Inter-cluster communication in VLIW architectures</td>
<td>ACM TACO, Voumel 4, Issue 2, 2007</td>
<td><a href="http://dx.doi.org/10.1145/1250727.1250731">http://dx.doi.org/10.1145/1250727.1250731</a></td>
</tr>
<tr>
<td>Heterogeneous Clustered VLIW Microarchitectures</td>
<td>CGO’07</td>
<td><a href="http://dx.doi.org/10.1109/CGO.2007.15">http://dx.doi.org/10.1109/CGO.2007.15</a></td>
</tr>
<tr>
<td>Impact of intercluster comm. mech. on ILP in clustered VLIW architectures</td>
<td>ACM TODAES, Vol 12, Issue 1, 2007</td>
<td><a href="http://dx.doi.org/10.1145/1188275.1188276">http://dx.doi.org/10.1145/1188275.1188276</a></td>
</tr>
<tr>
<td>Enabling compiler flow for embedded VLIW DSP procs with distributed RFs</td>
<td>LCTES’07</td>
<td><a href="http://dx.doi.org/10.1145/1254766.1254793">http://dx.doi.org/10.1145/1254766.1254793</a></td>
</tr>
<tr>
<td>Achieving Out-of-Order Performance with Almost In-Order Complexity</td>
<td>ISCA’08</td>
<td><a href="http://dx.doi.org/10.1109/ISCA.2008.23">http://dx.doi.org/10.1109/ISCA.2008.23</a></td>
</tr>
</tbody>
</table>
### Presentation Block #2:
Compilation for VLIW architectures

<table>
<thead>
<tr>
<th>Title</th>
<th>Published In</th>
<th>DOI/URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal vs. heuristic integrated code generation for clust. VLIW arch.</td>
<td>SCOPES’08</td>
<td><a href="http://portal.acm.org/citation.cfm?id=1361096.1361099">http://portal.acm.org/citation.cfm?id=1361096.1361099</a></td>
</tr>
<tr>
<td>Register allocation by puzzle solving</td>
<td>PLDI’08</td>
<td><a href="http://dx.doi.org/10.1145/1375581.1375609">http://dx.doi.org/10.1145/1375581.1375609</a></td>
</tr>
<tr>
<td>Register coalescing techniques for heterogeneous register architecture with copy sifting</td>
<td>ACM TECS, Volume 8, Issue 2, 2009</td>
<td><a href="http://dx.doi.org/10.1145/1457255.1457263">http://dx.doi.org/10.1145/1457255.1457263</a></td>
</tr>
<tr>
<td>Integrated Modulo Scheduling for Clustered VLIW Architectures</td>
<td>HiPEAC’09</td>
<td><a href="http://dx.doi.org/10.1007/978-3-540-92990-1_7">http://dx.doi.org/10.1007/978-3-540-92990-1_7</a></td>
</tr>
<tr>
<td>Hybrid multithreading for VLIW processors</td>
<td>CASES’09</td>
<td><a href="http://dx.doi.org/10.1145/1629395.1629403">http://dx.doi.org/10.1145/1629395.1629403</a></td>
</tr>
<tr>
<td>Optimal trace scheduling using enumeration</td>
<td>ACM TACO, Volume 5, Issue 4, 2009</td>
<td><a href="http://dx.doi.org/10.1145/1498690.1498694">http://dx.doi.org/10.1145/1498690.1498694</a></td>
</tr>
<tr>
<td>Simultaneous Multithreading VLIW DSP Architecture with Dynamic Dispatch Mechanism</td>
<td>DSD’09</td>
<td><a href="http://dx.doi.org/10.1109/DSD.2009.128">http://dx.doi.org/10.1109/DSD.2009.128</a></td>
</tr>
<tr>
<td>Dynamically reconfigurable register file for a softcore VLIW processor</td>
<td>DATE’10</td>
<td><a href="http://portal.acm.org/citation.cfm?id=1870926.1871162">http://portal.acm.org/citation.cfm?id=1870926.1871162</a></td>
</tr>
<tr>
<td>Design Space Exploration for Memory Subsystems of VLIW Architectures</td>
<td>NAS’10</td>
<td><a href="http://dx.doi.org/10.1109/NAS.2010.14">http://dx.doi.org/10.1109/NAS.2010.14</a></td>
</tr>
<tr>
<td>Design and chip implementation of a heterogeneous multi-core DSP</td>
<td>ASPDAC’11</td>
<td><a href="http://portal.acm.org/citation.cfm?id=1950815.1950837">http://portal.acm.org/citation.cfm?id=1950815.1950837</a></td>
</tr>
</tbody>
</table>
### Presentation Block #2: Compilation for VLIW architectures

<table>
<thead>
<tr>
<th>Date</th>
<th>Presenter</th>
<th>Paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 16</td>
<td>Honggyu Kim</td>
<td>A Unified Modulo Scheduling and Register Allocation Technique for Clustered Processors <a href="http://portal.acm.org/citation.cfm?id=645988.674300">http://portal.acm.org/citation.cfm?id=645988.674300</a></td>
</tr>
<tr>
<td>May 16</td>
<td>Bertram Schmitt</td>
<td>Loop fusion for clustered VLIW architectures <a href="http://dx.doi.org/10.1145/513829.513850">http://dx.doi.org/10.1145/513829.513850</a></td>
</tr>
<tr>
<td>May 18</td>
<td>Gangwon Jo</td>
<td>Achieving Out-of-Order Performance with Almost In-Order Complexity <a href="http://dx.doi.org/10.1109/ISCA.2008.23">http://dx.doi.org/10.1109/ISCA.2008.23</a></td>
</tr>
<tr>
<td>May 18</td>
<td>Christine Wagner</td>
<td>Hybrid multithreading for VLIW processors <a href="http://dx.doi.org/10.1145/1629395.1629403">http://dx.doi.org/10.1145/1629395.1629403</a></td>
</tr>
<tr>
<td>May 23</td>
<td>Ingoo Heo</td>
<td>Design Space Exploration for Memory Subsystems of VLIW Architectures <a href="http://dx.doi.org/10.1109/NAS.2010.14">http://dx.doi.org/10.1109/NAS.2010.14</a></td>
</tr>
</tbody>
</table>
### Paper List

- **Presentation Block #3: Compilation for CGRA**

<table>
<thead>
<tr>
<th>Title</th>
<th>Published In</th>
<th>DOI/URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture Exploration for a Reconfigurable Architecture Template</td>
<td>IEEE Design &amp; Test Volume 22, Issue 2, 2005</td>
<td><a href="http://dx.doi.org/10.1109/MDT.2005.27">http://dx.doi.org/10.1109/MDT.2005.27</a></td>
</tr>
<tr>
<td>Placement-and-routing-based register allocation for coarse-grained reconfigurable arrays</td>
<td>LCTES’08</td>
<td><a href="http://dx.doi.org/10.1145/1375657.1375678">http://dx.doi.org/10.1145/1375657.1375678</a></td>
</tr>
<tr>
<td>CGRA express: accelerating execution using dynamic operation fusion</td>
<td>CASES’09</td>
<td><a href="http://dx.doi.org/10.1145/1629395.1629433">http://dx.doi.org/10.1145/1629395.1629433</a></td>
</tr>
<tr>
<td>Edge-centric modulo scheduling for coarse-grained reconfigurable architectures</td>
<td>PACT’08</td>
<td><a href="http://dx.doi.org/10.1145/1454115.1454140">http://dx.doi.org/10.1145/1454115.1454140</a></td>
</tr>
<tr>
<td>Heterogeneous coarse-grained processing elements: a template architecture for embedded processing acceleration</td>
<td>DATE’09</td>
<td><a href="http://portal.acm.org/citation.cfm?id=1874620.1874752">http://portal.acm.org/citation.cfm?id=1874620.1874752</a></td>
</tr>
<tr>
<td>Coarse-grained reconfigurable architecture for multiple application domains: a case study</td>
<td>ICHIT’09</td>
<td><a href="http://dx.doi.org/10.1145/1644993.1645095">http://dx.doi.org/10.1145/1644993.1645095</a></td>
</tr>
<tr>
<td>Recurrence cycle aware modulo scheduling for coarse-grained reconfigurable architectures</td>
<td>LCTES’09</td>
<td><a href="http://dx.doi.org/10.1145/1542452.1542456">http://dx.doi.org/10.1145/1542452.1542456</a></td>
</tr>
</tbody>
</table>
### Presentation Block #3: Compilation for CGRA

<table>
<thead>
<tr>
<th>Title</th>
<th>Published In</th>
<th>DOI/URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polymorphic pipeline array: a flexible multicore accelerator with virtualized execution for mobile multimedia applications</td>
<td>MICRO 42</td>
<td><a href="http://dx.doi.org/10.1145/1669112.1669160">http://dx.doi.org/10.1145/1669112.1669160</a></td>
</tr>
<tr>
<td>Operation and data mapping for CGRAs with multi-bank memory</td>
<td>LCTES’10</td>
<td><a href="http://dx.doi.org/10.1145/1755888.1755892">http://dx.doi.org/10.1145/1755888.1755892</a></td>
</tr>
<tr>
<td>Resource recycling: putting idle resources to work on a composable accelerator</td>
<td>CASES’10</td>
<td><a href="http://dx.doi.org/10.1145/1878921.1878925">http://dx.doi.org/10.1145/1878921.1878925</a></td>
</tr>
<tr>
<td>Diet SODA: a power-efficient processor for digital cameras</td>
<td>ISLPED’10</td>
<td><a href="http://dx.doi.org/10.1145/1840845.1840862">http://dx.doi.org/10.1145/1840845.1840862</a></td>
</tr>
<tr>
<td>An instruction-scheduling-aware data partitioning technique for coarse-grained reconfigurable architectures</td>
<td>LCTES’11</td>
<td><a href="http://dx.doi.org/10.1145/1967677.1967699">http://dx.doi.org/10.1145/1967677.1967699</a></td>
</tr>
</tbody>
</table>
## Paper List

### Presentation Block #3: Compilation for CGRA

<table>
<thead>
<tr>
<th>Date</th>
<th>Presenter</th>
<th>Paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>June 1</td>
<td>Ingoo Heo</td>
<td>Architecture Exploration for a Reconfigurable Architecture Template</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://dx.doi.org/10.1109/MDT.2005.27">http://dx.doi.org/10.1109/MDT.2005.27</a></td>
</tr>
<tr>
<td>June 8</td>
<td>Christine Wagner</td>
<td>Edge-centric modulo scheduling for coarse-grained reconfigurable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>architectures</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://dx.doi.org/10.1145/1454115.1454140">http://dx.doi.org/10.1145/1454115.1454140</a></td>
</tr>
<tr>
<td>June 8</td>
<td>Bertram Schmitt</td>
<td>Polymorphic pipeline array: a flexible multicore accelerator with</td>
</tr>
<tr>
<td></td>
<td></td>
<td>virtualized execution for mobile multimedia applications</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://dx.doi.org/10.1145/1669112.1669160">http://dx.doi.org/10.1145/1669112.1669160</a></td>
</tr>
<tr>
<td>June 13</td>
<td>Hongyu Kim</td>
<td>Resource recycling: putting idle resources to work on a composable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>accelerator</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://dx.doi.org/10.1145/1878921.1878925">http://dx.doi.org/10.1145/1878921.1878925</a></td>
</tr>
<tr>
<td>June 13</td>
<td>Gangwon Jo</td>
<td>Placement-and-routing-based register allocation for coarse-grained</td>
</tr>
<tr>
<td></td>
<td></td>
<td>reconfigurable arrays</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://dx.doi.org/10.1145/1375657.1375678">http://dx.doi.org/10.1145/1375657.1375678</a></td>
</tr>
</tbody>
</table>