Appendix A

Description of Profiling Format

For reference purposes, Table A.1 introduces the different packets and packet fields that exist in the profiling format introduced in Chap. 3. The entries AllocBegin and AllocEnd, and DeallocBegin and DeallocEnd, are two profiling tokens that represent a single allocation or de-allocation event of the application. However, it is useful to split them because this allows to analyze the memory accesses that are performed during the (de)allocation process itself.
Table A.1  Structure of the profiling packet for each specific event in the behavior of the application

<table>
<thead>
<tr>
<th>Log packet</th>
<th>Field 1</th>
<th>Field 2</th>
<th>Field 3</th>
<th>Field 4</th>
<th>Field 5</th>
<th>Field 6</th>
<th>Field 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector Construct</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td></td>
</tr>
<tr>
<td>Vector Duplicate</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>old SequenceId</td>
</tr>
<tr>
<td>Vector Destruct</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td></td>
</tr>
<tr>
<td>Vector Swap</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>old SequenceId</td>
</tr>
<tr>
<td>Vector Resize</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td></td>
</tr>
<tr>
<td>Iterator Next</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>address</td>
</tr>
<tr>
<td>Iterator Previous</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>address</td>
</tr>
<tr>
<td>Iterator Add</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>address</td>
</tr>
<tr>
<td>Iterator Sub</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>address</td>
</tr>
<tr>
<td>Iterator Get</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>address</td>
</tr>
<tr>
<td>Vector Get</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>index</td>
</tr>
<tr>
<td>Vector Add</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>index</td>
</tr>
<tr>
<td>Vector Remove</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td>index</td>
</tr>
<tr>
<td>Vector Clear</td>
<td>threadId</td>
<td>element</td>
<td>sequenceId</td>
<td>instanceId</td>
<td>element</td>
<td>Size</td>
<td></td>
</tr>
<tr>
<td>Vector End</td>
<td>threadId</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Var Read</td>
<td>threadId</td>
<td>VarId</td>
<td>address</td>
<td>size</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Var Write</td>
<td>threadId</td>
<td>VarId</td>
<td>address</td>
<td>size</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(continued)
<table>
<thead>
<tr>
<th>Log packet</th>
<th>Field 1</th>
<th>Field 2</th>
<th>Field 3</th>
<th>Field 4</th>
<th>Field 5</th>
<th>Field 6</th>
<th>Field 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alloc Begin</td>
<td>threadId</td>
<td>allocated Id</td>
<td>size</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alloc End</td>
<td>threadId</td>
<td>allocated Id</td>
<td>size</td>
<td>address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dealloc Begin</td>
<td>threadId</td>
<td>allocated Id</td>
<td>address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dealloc End</td>
<td>threadId</td>
<td>allocated Id</td>
<td>address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scope Begin</td>
<td>threadId</td>
<td>scopeId</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scope End</td>
<td>threadId</td>
<td>scopeId</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thread Begin</td>
<td>threadId</td>
<td>oldThread Id</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thread End</td>
<td>threadId</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
References


© Springer International Publishing Switzerland 2015
D. Atienza Alonso et al., Dynamic Memory Management for Embedded Systems, DOI 10.1007/978-3-319-10572-7


References

32. D. Bulka, D. Mayhew, Efficient C++ (Addison-Wesley, Wokingham, 2001)
55. G. de Micheli, L. Benin (eds.), Networks on Chips (Technology and Tools (Morgan Kaufmann Publishers, San Francisco, 2006)
64. E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software (Addison-Wesley Longman Publishing Co., Inc, Boston, 1995)
75. J. Holland, Adaptation in Natural and Artificial Systems (University of Michigan Press, Ann Arbor, 1975)
76. CACTI 5.3 (2008), http://quid.hpl.hp.com:9081/cacti/
80. K. Ishibashi, K. Osada (eds.), Low Power and Reliable SRAM Memory Cell and Array Design (Springer, Germany, 2011)
84. N. Jouppi, Western research laboratory, cacti (2002), http://research.compaq.com/wrl/people/jouppi/CACTI.html
88. L. Kharevych, R. Khan, 3D Physics Engine for Elastic and Deformable Bodies (University of Maryland, College Park, 2002)
89. O. Kiselyov, General ways to traverse collections (2000).
References

References

118. Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs (Springer-Verlag, Berlin, 1996)
120. MICRON, Mobile LPDDR2 SDRAM–MT42L64M32D1 Rev. N 3/12 EN (Micron Technology Inc, 2012).
122. Moby games, a game documentation and review project, http://www.mobygames.com/


160. Target jr (2002), http://www.robots.ox.ac.uk/tgtjr/


