Is Chip-Multiprocessing the End of Real-Time Scheduling?

Martin Schoeberl
Peter Puschner
Time-Predictable CMPs?

• Future CMP will contain hundreds of cores
• Embedded systems are a good fit for CMPs
• One thread per core in reach

⇒ Can we build a time-predictable CMP system?
⇒ What are the cost?
Schedule Memory Access, not Tasks

What if #cores > #tasks?

• Traditional task scheduling disappears
• Memory is shared ⇒ schedule memory accesses
  – TDMA based memory arbitration
  – Adapt (static) schedule to application needs
  – Context switch overhead of arbiter is small
  ⇒ Fine-grained scheduling of memory accesses
Memory-Access Schedules

- Tasks and memory-access schedules are not synchronized
- Tasks may have arbitrary periods
- One task per core ⇒ task can use 100% of its period
  ⇒ Revision of memory-access schedule:
    ⇒ Assign lower bandwidth to tasks with slack time
    ⇒ Other tasks get additional bandwidth
- Use knowledge about memory-access schedule in WCET analysis
- Use results from WCET analysis to improve schedules
CMP System

- Multiple JOP cores (JOP … Java-optimized processor)
  - time-predictable JVM in hardware
  - local method cache (simplifies cache analysis)
  - local stack cache and scratch pad for local data
- Shared memory (4-cycle read, 6-cycle write)
  - TDMA memory arbitration
  - Lock-based synchronization (Java)
JOP CMP System

JOP core

4-stage pipeline

M$
S$
SPM

JOP core

4-stage pipeline

M$
S$
SPM

JOP core

4-stage pipeline

M$
S$
SPM

TDMA arbiter

JOP chip-multiprocessor (FPGA)

Memory controller

Main memory (SRAM)
WCET Analysis

- At Java bytecode level
  - Standard IPET approach
- Simple bytecodes, constant execution time
- Memory access times
  - Depend on memory arbiter
  - TDMA schedule input for low-level analysis
TDMA Schedule Generation

1. Generate initial arbiter schedule
2. Task 1 → WCET Analysis
3. Task 2 → WCET Analysis
4. Task 3 → WCET Analysis
5. WCET 1 → Deadlines met?
6. WCET 2 → Slack left?
7. WCET 3 → Generate arbiter schedule

- Yes: System schedulable
- No: System not schedulable
Evaluation: Gain CMP vs. Single Core?

- 5 cores, one thread per core, same period
- Bus arbitration: RR, 6 cycles per thread

<table>
<thead>
<tr>
<th>Function</th>
<th>Task</th>
<th>Single</th>
<th>5 cores</th>
<th>WCET increase</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipLink.run()</td>
<td>$\tau_1$</td>
<td>1061</td>
<td>2525</td>
<td>2.4</td>
</tr>
<tr>
<td>resultServer()</td>
<td>$\tau_2$</td>
<td>1526</td>
<td>3443</td>
<td>2.3</td>
</tr>
<tr>
<td>request()</td>
<td>$\tau_3$</td>
<td>8825</td>
<td>23445</td>
<td>2.7</td>
</tr>
<tr>
<td>macServer()</td>
<td>$\tau_4$</td>
<td>7004</td>
<td>17104</td>
<td>2.4</td>
</tr>
<tr>
<td>net.run()</td>
<td>$\tau_5$</td>
<td>8188</td>
<td>18796</td>
<td>2.3</td>
</tr>
<tr>
<td>Sum</td>
<td></td>
<td>26604</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Variations of Arbiter Schedule

- Double-sized slots for long threads (1)
- Several (short) slots per thread per round (2, 3)
- Slot size variation: 6-/8-cycle slots (4, 5, 6)

<table>
<thead>
<tr>
<th>Task</th>
<th>Equal</th>
<th>Schedule 1</th>
<th>Schedule 2</th>
<th>Schedule 3</th>
<th>Schedule 4</th>
<th>Schedule 5</th>
<th>Schedule 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \tau_1 )</td>
<td>2525</td>
<td>3605</td>
<td>3605</td>
<td>5045</td>
<td>4325</td>
<td>5285</td>
<td>5525</td>
</tr>
<tr>
<td>( \tau_2 )</td>
<td>3443</td>
<td>4865</td>
<td>4865</td>
<td>6761</td>
<td>5813</td>
<td>7077</td>
<td>7393</td>
</tr>
<tr>
<td>( \tau_3 )</td>
<td>23445</td>
<td>25313</td>
<td>20211</td>
<td>20201</td>
<td>20473</td>
<td>17955</td>
<td>18681</td>
</tr>
<tr>
<td>( \tau_4 )</td>
<td>17104</td>
<td>15284</td>
<td>14860</td>
<td>19468</td>
<td>14276</td>
<td>16756</td>
<td>17376</td>
</tr>
<tr>
<td>( \tau_5 )</td>
<td>18796</td>
<td>17856</td>
<td>16432</td>
<td>14832</td>
<td>16204</td>
<td>18924</td>
<td>18364</td>
</tr>
</tbody>
</table>

Speedup factor: 1.4  (26604 cycles for single core)
Conclusion

• Speedup of 1.4 on 5-core CMP is not impressive, but:
  • Predictable task timing by TDMA mem. arbitration
  • No scheduling and context switch overhead as in single-processor system (cost not included above)
  • No cache thrashing (cost not included above)
  • Task periods don’t have to be harmonic
  • Low complexity, low risk WCET & timing analysis!
    (JOP CMP and WCET tool available under GNU GPL)
... thank you!

http://ti.tuwien.ac.at/rts