Course Projects
Distributed Embedded Systems

Andrea Acquaviva
University of Verona, ITALY

Overall Objectives

- The main objective of the course project will be to evaluate how a set of networking and non-networking applications can be implemented on Freescale multicore platforms. In particular, the following aspects will be addressed:
  - Methodology to go to multicore
  - Scalability
  - Optimization
  - Debugging
Project Topics

• Methodology to go to multicore
  – programming models, compilers, runtime engines

• Scalability
  – Increasing number of cores, effects of interconnects, cost of synchronization, coherency

• Optimization
  – Load balancing, task allocation, energy and thermal management

• Debugging
  – Runtime support for multicore debugging

Requirements

• Test scalability between 1 to N cores
  – how control-plane is implemented and how it scales

• Networking applications
  – Header processing: routing IPv6, NAT, packet filtering (DiffServ)
  – Content processing: security, encapsulation, antivirus, spam filtering

• Multimedia applications
  – Video transcoding, face/image recognition, VoIP

• Application scenarios
  – Networking, multimedia, networking + multimedia

• Programming methodologies considered for evaluation
  – Thread library application agnostic (Pthreads)
  – Parallelizing compilers (OpenMP)

• Other requirements
  – Automated and repeatable tests
  – Standard API, GPL license, Open source
## Project List

<table>
<thead>
<tr>
<th>Project name</th>
<th>Short description</th>
<th>Notes</th>
<th>Target</th>
</tr>
</thead>
</table>
| 1. Security offload using BMP | Study and implementation of optimal allocation of security application on multicore architectures using BMP strategy and comparison with default SMP | 1. Performance profiling of security functions (encryption, key management, etc)  
2. Optimal task mapping and communication definition  
3. Implementation of demonstrator of the strategy on the target platform | Freescale MPC8572 |
| 2. VoIP offload using BMP    | Study and implementation of optimal allocation of VoIP applications in multicore architectures using BMP strategy and comparison with default SMP | 1. Performance profiling of VoIP application and task graph definition  
2. Optimal task mapping and communication definition  
3. Implementation of demonstrator of the strategy on the target platform | Freescale MPC8572 |
| 3. Runtime support for asymmetric multiprocessing in Linux | Implementation of runtime support for acceleration of security applications on Linux SMP using master-slave configuration | 1. Use CPU affinity support on Linux to bound processor control  
2. Implementation of memory areas for code, data and buffering of accelerators | Freescale MPC8572 |
| 4. Security offload using AMP | Optimization of security applications (OpenSSL, IPsec) on multicore architectures using AMP strategy | 1. Implementation of hardware and software accelerators of security functions and protocols  
2. Integration in the AMP runtime engine  
3. Comparison with default Linux SMP performance | Freescale MPC8572 |
| 5. VoIP offload using AMP    | Optimization of a VoIP applications (asterisk) on multicore architectures using AMP strategy | 1. Implementation of hardware and software accelerators of VoIP engines  
2. Integration in the AMP runtime engine  
3. Comparison with default Linux SMP performance | Freescale MPC8572 |
## Project List

<table>
<thead>
<tr>
<th>Project name</th>
<th>Short description</th>
<th>Notes</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>6. Virtualization</td>
<td>Implementation of an Hypervisor on SMP architecture</td>
<td>XEN</td>
<td>Freescale MPC8572</td>
</tr>
<tr>
<td>7. OpenMP</td>
<td>Evaluation of OpenMP efficiency and overhead on embedded multicore SMP architecture and comparison with direct Pthread implementation</td>
<td>GCC + OpenMP</td>
<td>Freescale MPC8572</td>
</tr>
</tbody>
</table>

## Additional Projects

<table>
<thead>
<tr>
<th>Project name</th>
<th>Short description</th>
<th>Notes</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDF</td>
<td>Modelling an application using SDF. Comparison of analytical and real throughput.</td>
<td>1. Modelling of communication 2. Comparison using MPARM simulator</td>
<td>MPARM</td>
</tr>
<tr>
<td>Ray tracing</td>
<td>Implement a graphical application on Cell processor using ILP+CP</td>
<td>Ray-tracing application, Cell simulator and SDK</td>
<td>IBM Cell</td>
</tr>
<tr>
<td>Task migration</td>
<td>Implementation of runtime engine for migration on Cell</td>
<td></td>
<td>IBM Cell</td>
</tr>
<tr>
<td>Compiler assisted task migration</td>
<td>Optimization of migration points placement using GCC</td>
<td></td>
<td>Freescale, MPARM, Cell</td>
</tr>
</tbody>
</table>
Proposed Projects

<table>
<thead>
<tr>
<th>Project name</th>
<th>Short description</th>
<th>Notes</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic Linux Driver Implementation</td>
<td>Methodology to design linux device drivers</td>
<td>-</td>
<td>Freescale MPC8572</td>
</tr>
</tbody>
</table>

Project Organization

- **Common phase**
  - Experimental environment set-up (set-up of the simulator and porting of the application as it is)
  - Modeling and profiling the considered applications using task graph (TG)
    - Classification of the application as control or data-flow oriented
    - Thread organization, communication and synchronization model (either shared memory or message passing)
    - Execution times of building blocks
    - Memory utilization (global and private data structures, stack, heap)
    - I/O utilization
  - Metric definition (performance figure depending on the selected application)
Common Phase: Objectives

- The student should come-up with a suitable modeling methodology
  - Task graph (TG) model should be defined
  - The model can be refined during successive phases of the project
  - The usage of memory and I/O have to be profiled, distinguishing between global and private data structures
  - Type of system calls performed (either blocking, non blocking or asynchronous).
- A performance metric must be defined for successive evaluation of scalability properties
  - The metric is application dependent, but it is preferable to define sets of metrics depending on the application category (i.e., networking and non-networking).
  - For instance, frame rate, jitter, delay for image processing applications or packet rate, waiting time and packet losses for networking applications
- Milestone 1. Application modeling and metric definition

Projects Timeline

- New deadline for project registration: April 4
  - For projects beginning on April 2008
- Maximum duration
  - 3 months
- Synchronization
  - Each week: update meeting
  - After 1 month: objectives re-negotation
  - 2 weeks before submission: preliminary presentation of results
Reporting

• Written report or presentation slides
• Report structure
  – State of the art and background
  – Work description
    • Provide the documentation needed to repeat the experiments
  – Comment and justify obtained results!
• Use of English language

Project Grading

• 70% objectives
• 30% timeliness
Project Duration

- Course projects: 2.5 months + 2 weeks write up
  - Initial common phase during the course (1PM)
  - Second phase continues after the course (2PM)
- Thesis projects: 5 months + 1 month write up
  - Initial common phase (1PM)
  - Second phase (2PM x number of small projects)

Projects Periods

- Course projects start on April 2008
- Thesis projects start on demand
  - A student can continue a course project for two more months