Managing Hardware Power Saving Modes for High Performance Computing
Second International Green Computing Conference 2011, Orlando

Timo Minartz, Michael Knobloch, Thomas Ludwig, Bernd Mohr
timo.minartz@informatik.uni-hamburg.de
Scientific Computing
Department of Informatics
University of Hamburg
26-07-2011
Motivation

High Performance Computing

- Increasing performance and efficiency of calculation units
  - But: Increasing need for calculation power increases size of installations
- High operational costs for large-scale high performance installations
- High carbon footprint of installations should be reduced for environmental and political reasons
1 Introduction

2 Hardware Power Saving Modes in HPC

3 Power Mode Control System

4 Evaluation

5 Conclusion and Future Work
High Performance Computing (HPC)

- Important tool in natural sciences to analyze scientific questions in silico
- Modeling and simulation instead of performing time consuming and error prone experiments
- Models from weather systems to protein folding to nanotechnology
- Leads to new observations and understanding of phenomena
Introduction (2)

Top500 list – http://www.top500.org

- Since 1993 the Top500 list gathers information about achieved performance of supercomputers
- Exponential growth in computing performance can be observed
- Current (June 2011) rank #1: K computer by Fujitsu
  - Peak performance: 8773.63 TFlop/s
  - Power consumption: 9898.56 KW

Green500 list – http://www.green500.org

- Ranking based on energy-efficiency
- Energy-efficiency is defined as Flop/s per Watt
- K computer rank (June 2011): #6
Exascale computing

**Roadmap**

- "Practical power limit" of 20 MW (U.S. Department of Energy)
- Energy-efficiency must be increased from different viewpoints
  - The data-center itself including cooling facilities etc.
  - The hardware running the scientific applications
  - The scientific applications themselves
Hardware power saving modes

Adaption of mechanism from mobile devices

- Idle / power saving modes of hardware
- Dynamic Voltage and Frequency Scaling of processors (DVFS)
- Adapt frequency and disable ports of network devices
- Spin down and flush caches of hard disks

HPC related problems

- Synchronization problems
- OS Jitter increase
- Possible performance loss
Idle power saving potential

- **Opteron:** up to 11% power savings
- **Xeon:** up to 18% power savings
CPU power saving potential for Xeon node

- 30% power savings
- Interesting in phases of busy-waiting or memory-boundness
When to switch which component...

Performance/usage prediction

- Utilization based approach
- Instructions Per Second (IPS)
- Other performance counters (e.g. memory bandwidth)

Knowledge about future hardware use

- Application developer
- Compiler
- Libraries
Daemon architecture

Diagram showing the interaction between the application, server daemon, operating system, and various hardware components like processor, hard disk, and network card. The diagram illustrates the flow of requirements and mode control, as well as utilization between these components and the server daemon.
Daemon design

Server daemon

- Make decision about hardware power states
- Processor
  - Reduce frequency (P-States) using `cpufreq`
- Network card
  - Reduce speed / switch duplex mode using `ethtool`
- Harddisk
  - Reduce OS access and enforce standby modes using `hdparm`

Client library

- Linked to (MPI) application
- Forwards desired device power state via sockets to server
Test MPI applications

**partdiff-par**
- PDE solver
- Computation intensive phases, communication intensive phases and IO phases

**PEPC**
- Pretty Efficient Parallel Coulomb-solver
- Computation intensive phases and communication intensive phases
Hardware

Details

- 3 × LMG 450 power meter
  - 4 channels each
  - up to 20 samples per second

- 5 × AMD Opteron 6168
  - Dual socket
  - 24 cores per node

- 5 × Intel Xeon X5560
  - Dual socket
  - 8 cores per node
  - SMT disabled
Experimental setup

Application test setup

- **Instrumented**: State switching dependent on phase
- **CPU Max Freq**: Processor frequency set to maximum
- **CPU Min Freq**: Processor frequency set to minimum
- **CPU Ondemand**: Processor governor set to ondemand

Results

- **Time-to-Solution** (TTS): Total time for test setup
- **Energy-to-Solution** (ETS): Total energy for test setup
TTS and ETS for \texttt{partdiff-par} on Xeon nodes

- 5\% savings in Energy-to-Solution
- Time-to-Solution increase of about 4\%
TTS and ETS for **partdiff-par** on Opteron nodes

- **8%** energy savings
- **Runtime increase of 9%**
TTS and ETS for **PEPC** on Opteron nodes

- 7% energy savings, 4% runtime increase compared to
- 4% energy savings, 2% runtime increase
Conclusions

- Power consumption of idle nodes can be reduced by 11% and 18% respectively.
- Power consumption can be decreased by more than 30% in phases with unnecessary high utilization (e.g. busy-waiting).
- Control device power states from userspace by introducing a hardware management daemon.
- Reduce the Energy-to-Solution by up to 8% with an Time-to-Solution increase of about 9% for presented applications.
Future work

- Identify energy saving possibilities (Scalasca enhancement)
- Benchmark to measure power consumption in various states
- Enhanced measurements with larger count of applications
CPU power saving potential (C-States enabled)