OpenCL based HMC Development Snapshot

Lars Zeidlewicz

Group Meeting – May 9, 2011

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Institut für Theoretische Physik
Goethe-Universität Frankfurt

Outline

- LOEWE-CSC
- Our code: desired features
- Code & project management
- Code overview
- Some words on OpenCL

LOEWE-CSC

Some numbers…

- **Top500 rank 22 (Nov 2010)**
  (285 Tflops, #2 in Germany)
- **Green500 rank 8**
  (741 Mflops/W, #4 in Germany)
- **About 800 nodes with**
  - 2 AMD Opteron 12 core
  - 1 AMD ATI Radeon 5870 (Cypress)

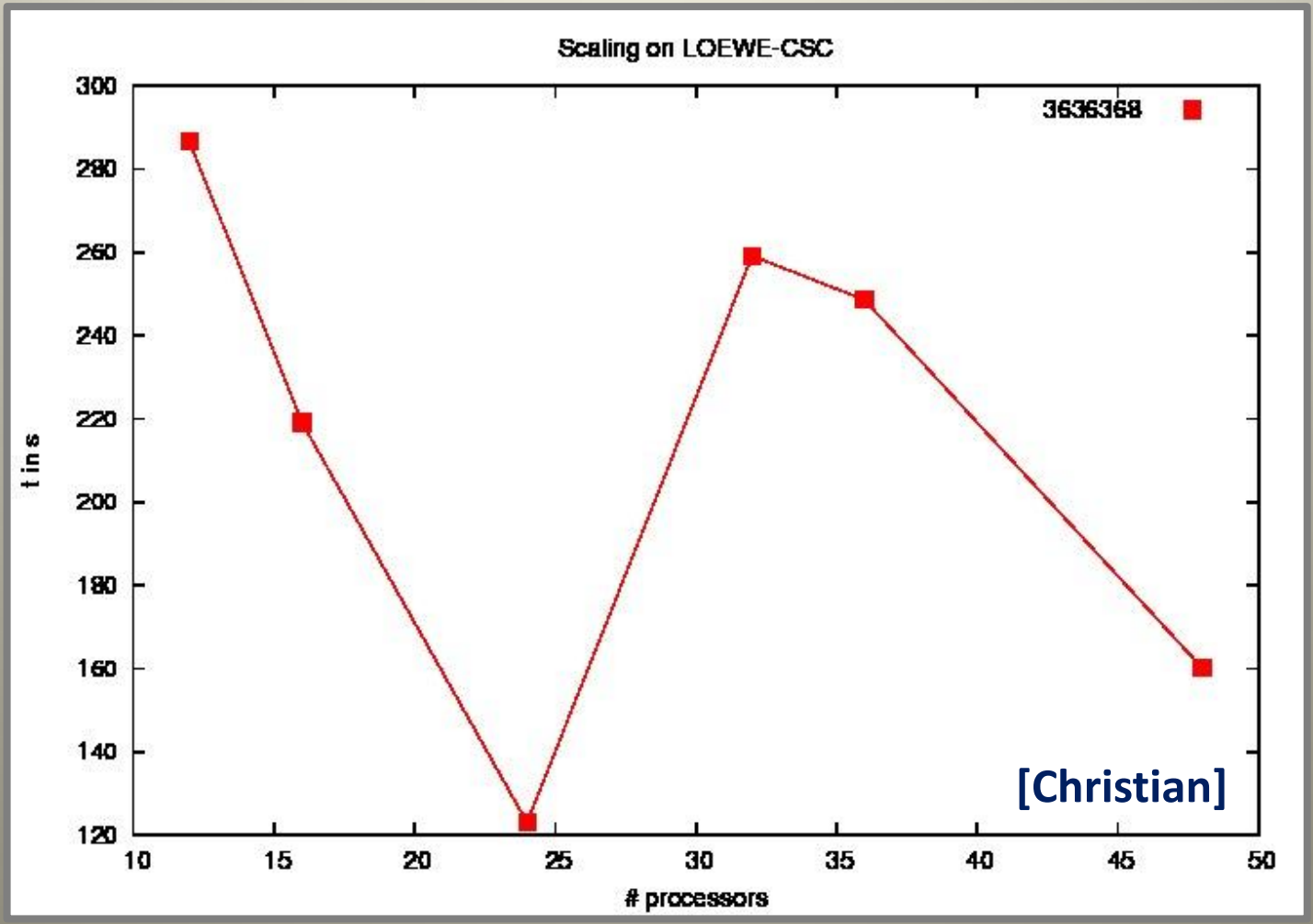
LOEWE-CSC

Status

- **Testing phase**
- **Monthly CSC-admin meetings**
- **Current issues:**
  - Queue: Large jobs vs. single-core (Jobs wait for days and weeks)
  - Infiniband problems
  - 2 week downtime starts on Friday
- **GPUs so-far unused**

LOEWE-CSC

Status



SLURM

- **Admins hope to overcome queuing problems by switching to the SLURM scheduler**

  (https://computing.llnl.gov/linux/slurm/slurm.html)
- **Today: 29 nodes**

  After downtime: 100 nodes
- **OpenMPI tested *(only!)***

LOEWE-CSC

SLURM

```
#! /bin/sh
#
#SBATCH --nodes=4
#SBATCH --time=1
#SBATCH --cpus-per-task=1
#SBATCH --job-name=testjob
#
OMP_NUM_THREADS=1
Srun -ln4 --mpi=none hostname
```

- How does that work for a "real" MPI program?
- SLURM can be loaded as a module:
  - module load slurm
  - module show slurm
  - (sinfo, sqeue,…)
- Working examples have been promised

- OpenMPI tested *(only!)*

Code Features

Physics

- SU(3) heatbath
- Fermion inverter (different actions)
- Hybrid Monte-Carlo
- *Chemical potential*
- *Smearing*
- *…*

Code Features

Programming

- **Hybrid structure**
  - ***Within nodes***
  - ***Beyond one node***
- **OOP for host code**
- **Large flexibility**
  - **Switches (single/double etc)**
  - **User defined types, access functions to faciliate changes**

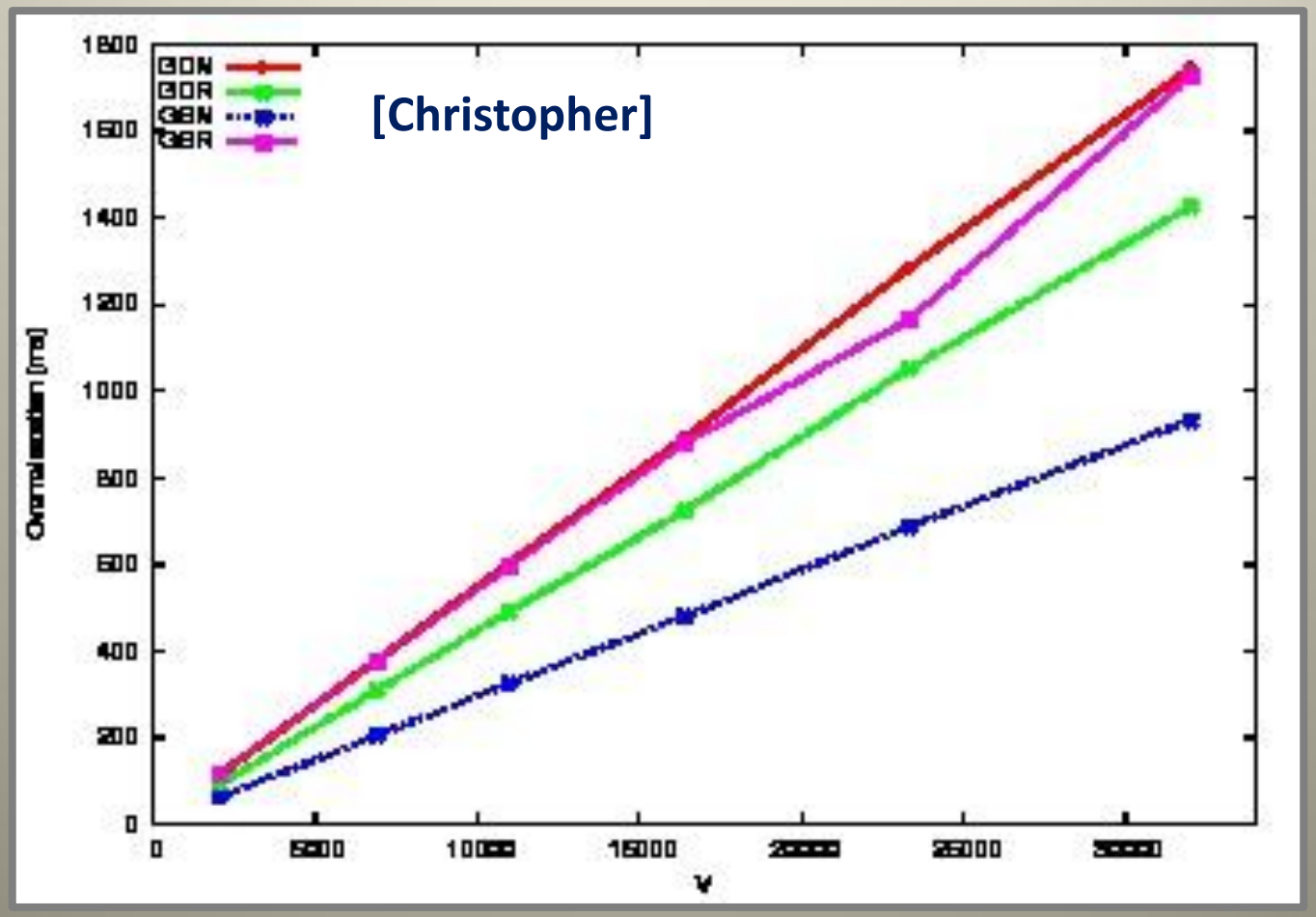
Code Features

GPUs

- **Today:**
  - Memory Management **[Matthias]**
    - Benchmarking
  - Workgroup sizes **[Matthias]**
- **Future:**
  - Several devices

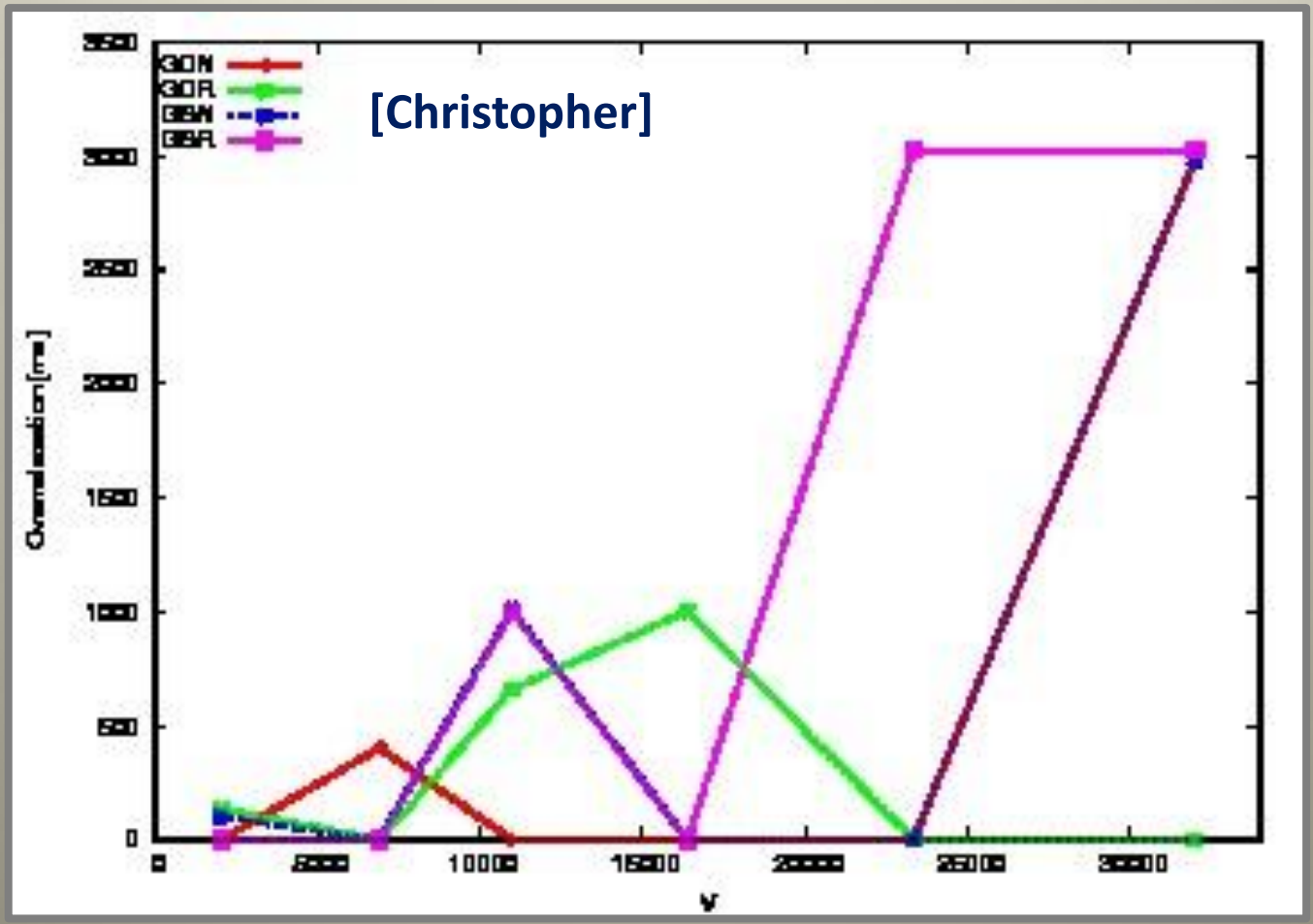
Code Features

GPUs benchmark – gpu-dev



Code Features

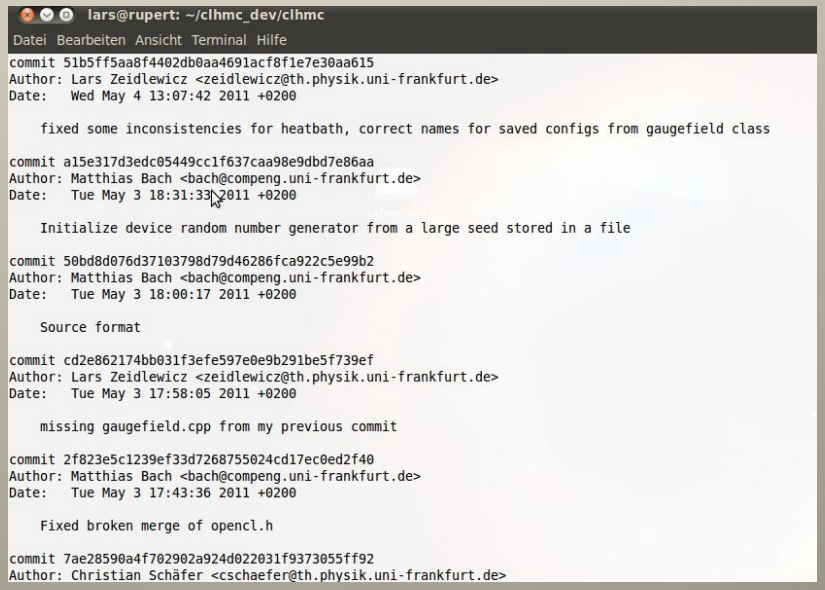
GPUs benchmark – loewe



Code & Project Management

git

- Code is subject to version management by git




```
commit 51b5ff5aa8f4402db0aa4691acf8f1e7e30aa615
Author: Lars Zeidlewicz <zeidlewicz@th.physik.uni-frankfurt.de>
Date:   Wed May 4 13:07:42 2011 +0200

    fixed some inconsistencies for heatbath, correct names for saved configs from gaugefield class

commit a15e317d3edc05449cc1f637caa98e9dbd7e86aa
Author: Matthias Bach <bach@compeng.uni-frankfurt.de>
Date:   Tue May 3 18:31:33 2011 +0200

    Initialize device random number generator from a large seed stored in a file

commit 50bd8d076d37103798d79d46286fca922c5e99b2
Author: Matthias Bach <bach@compeng.uni-frankfurt.de>
Date:   Tue May 3 18:00:17 2011 +0200

    Source format

commit cd2e862174bb031f3efe597e0e9b291be5f739ef
Author: Lars Zeidlewicz <zeidlewicz@th.physik.uni-frankfurt.de>
Date:   Tue May 3 17:58:05 2011 +0200

    missing gaugefield.cpp from my previous commit

commit 2f823e5c1239ef33d7268755024cd17ec0ed2f40
Author: Matthias Bach <bach@compeng.uni-frankfurt.de>
Date:   Tue May 3 17:43:36 2011 +0200

    Fixed broken merge of opencl.h

commit 7ae28590a4f702902a924d022031f9373055ff92
Author: Christian Schäfer <cschaefer@th.physik.uni-frankfurt.de>
```

Code & Project Management

Online repository

[Matthias]

- *http://code.compeng.uni-frankfurt.de/projects/clhmc*

OpenCL based HMC...

code.compeng.uni-frankfurt.de/projects/clhmc

Hauptseite Meine Seite Projekte Hilfe

Angemeldet als zeidlewicz Mein Konto Abmelden

OpenCL based HMC

Suche: OpenCL based HMC

Übersicht Aktivität Tickets Neues Ticket Projektarchiv Konfiguration

Übersicht

We want to write a complete HMC simulation programme with Wilson-type quarks (twisted mass action) using OpenCL for a hybrid structure consisting of CPUs and GPUs.

Tickets

- Bug: 8 offen / 8
- Feature: 7 offen / 9
- Unit Test: 0 offen / 0
- Benchmark: 0 offen / 0
- Test Setup: 0 offen / 0

Alle Tickets anzeigen | Kalender | Gantt-Diagramm

Git Repository

Read Only URL:

- git://code.compeng.uni-frankfurt.de/clhmc.git

Developer URL:

- git@code.compeng.uni-frankfurt.de:clhmc.git

Mitglieder

Manager: Matthias Bach
Developer: Christian Schäfer, Christopher Pinke, Lars Zeidlewicz, Matthias Bach, Michael Fromm, Stefano Lottini
Reporter: Christian Schäfer, Christopher Pinke, Lars Zeidlewicz, Matthias Bach, Michael Fromm, Stefano Lottini

Aufgewendete Zeit

18.25 Stunden

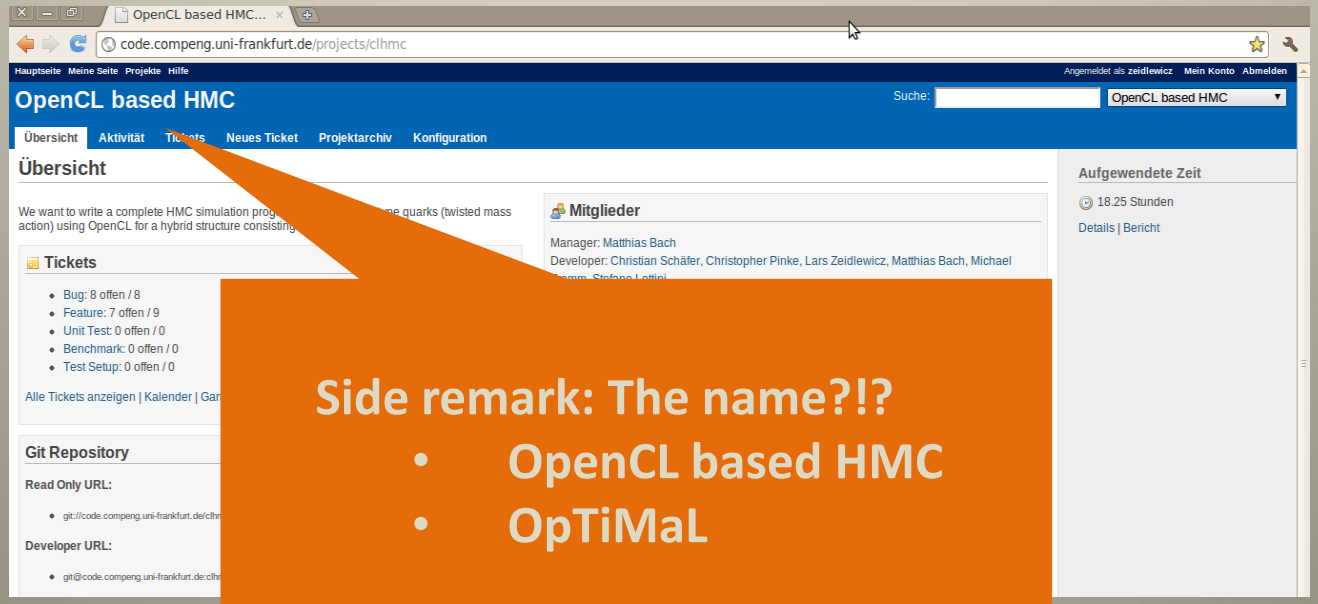
Details | Bericht

Code & Project Management

Online repository

[Matthias]

- *http://code.compeng.uni-frankfurt.de/projects/clhmc*



Side remark: The name?!?

- OpenCL based HMC
- OpTiMaL

Code & Project Management

Common style

[Matthias]

- Matthias has suggested to enforce a common coding style via

```
astyle --options=astylerc <FILENAME>
```

- To be discussed…

Code & Project Management

Doxygen

```
File Edit Options Buffers Tools C Help
class gaugefield {
 public:
/**
 * Initialize gaugefield and devices.
 *
 * @param[in] numdevs Number of wanted devices (so far, only 1 makes sense).
 * @param[in] devicetypes Array of wanted cl_device_types for the devices.
 * @param[in] input_parameters instance of inputparameters that contains information from input file
 * @param[in,out] timer Return initialization time.
 * @return Error code as defined in hmcerrs.h
 */
  hmc_error init(int numdevs, cl_device_type* devicetypes, inputparameters* input_parameters, usetimer* timer);
/**
 * Free gaugefield and device allocations.
 */
  hmc_error finalize();
/**
 * Save gaugefield to file.
 */
  hmc_error save(int number);
/**
 * Copy gaugefield to devices (currently: to device).
 * @param[in,out] timer copy-time
 */
  hmc_error copy_gaugefield_to_devices(usetimer* timer);
/**
 * Copy random array to devices (currently: to device).
 * @param[in,out] timer copy-time
 */
  hmc_error copy_rndarray_to_devices(hmc_rndarray host_rndarray,  usetimer* timer);
/**
 * Copy random array from devices (currently: from device).
 * @param[in,out] timer copy-time
 */
  hmc_error copy_rndarray_from_devices(hmc_rndarray rndarray, usetimer* timer);
/**
-UU-:----F1  gaugefield.h    14% L62    Git-master  (C/l Abbrev)----------------------------------------------
```

Code & Project Management

Doxygen

clhmc: gaugefield C...

file:///home/lars/clhmc_dev/clhmc/prog/doc/html/classgaugefield.html

```
hmc_error gaugefield::heatbath ( const size_t  local_work_size,
                                 const size_t  global_work_size,
                                 int           nheat,
                                 int           nover,
                                 usetimer *    timer_heat,
                                 usetimer *    timer_over
                               )
```

Perform a number of heatbath and (afterwards) overrelaxation steps.

Parameters:

[in]	*local_work_size*	OpenCL local_work_size
[in]	*global_work_size*	OpenCL global_work_size
[in]	*nheat*	number of heatbath steps
[in]	*nover*	number of overrelaxation steps
[in,out]	*timer_heat*	time for heatbath steps
[in,out]	*timer_over*	time for overrelaxation steps

```
hmc_error gaugefield::init ( int                numdevs,
                             cl_device_type *   devicetypes,
                             inputparameters *  input_parameters,
                             usetimer *         timer
                           )
```

Initialize gaugefield and devices.

Parameters:

[in]	*numdevs*	Number of wanted devices (so far, only 1 makes sense).
[in]	*devicetypes*	Array of wanted cl_device_types for the devices.
[in]	*input_parameters*	instance of inputparameters that contains information from input file
[in,out]	*timer*	Return initialization time.

Returns:

Error code as defined in hmcerrs.h

```
hmc_error gaugefield::overrelax ( const size_t  local_work_size,
                                  const size_t  global_work_size,
                                  usetimer *    timer
```

Use doxygen format for comments

Code & Project Management

cmake

- **Nice build-tool**
- **Have a look at clhmc/prog/INSTALL**
- **Creates Makefile:**
  - make hmc
  - make heatbath

Code status

Overview

- **Three different types of code:**
  - **Host management code**

    (→ C++)
  - **Host functional code**

    (deprecated?)
  - **Device functional code**

    (→ OpenCL)

Code status

Heatbath

- **Based on Christopher's code**
- **Check: Boyd et al.:**

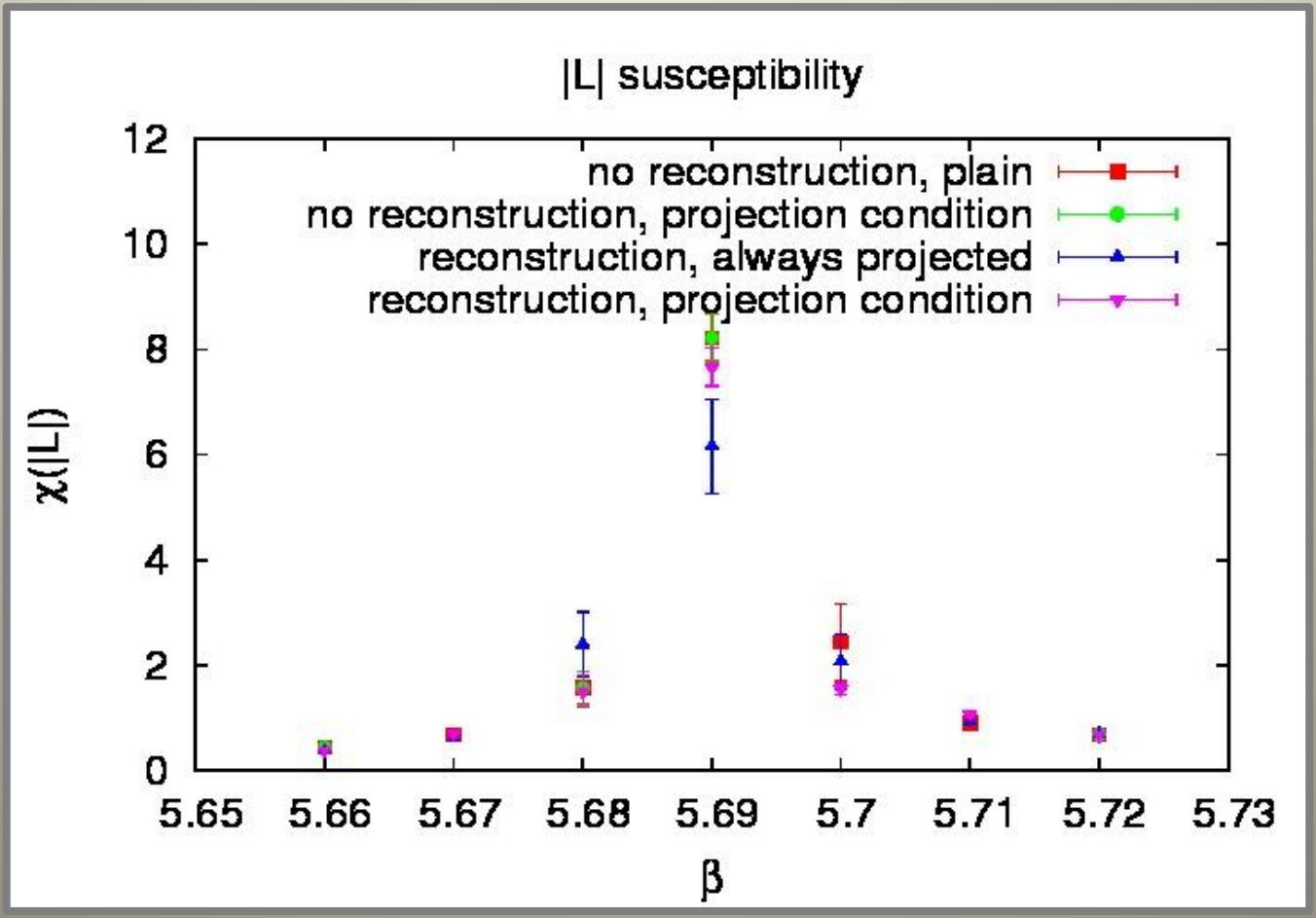
  Nucl. Phys. B469 (1996) 419-444
- **Host and device versions**

  still to check device version with final status
- **Current improvement:**

  Implement an optimal projection to SU(3)
  [Christian]

Code status

Heatbath



Code status

HMC & inverter

- Work in progress [Christopher, Steo]
- Host code inverter needs to be checked [Lars]
- So-far host code only
- Molecular dynamics force? [Christopher]

Code status

Management

- `int main(){}` in
  - hmc.cpp

    (mixes all progs by compiler flags)
  - heatbath.cpp
  - …
- Management in classes:
  - gaugefield
  - opencl
  - …

Code status

globaldefs.h

- `NC, NSPIN, NDIM,…`
- Right now: `NT, NS`
- Right now: work_sizes
- Derived constants (`VOLSPACE…`)
- Available in all of the code

  (sometimes useful:

  kernel flag `_INKERNEL_`)

Code status

types.h

- `hmc_float`
- `hmc_complex`
- `hmc_su3matrix`
- …
- Switches: `INKERNEL`, `USEDOUBLEPREC`, `RECONSTRUCT_TWELVE`

Code status

inputparameters

- Class `inputparameters`
- Reads input file
- Look at method
  `set_defaults()`

```
kappa=.125
mu=0.06
cgmax=1000
beta=6
#cgmax=3
prec=64
startcondition=cold
sourcefile=conf.save
thermalizationsteps=5
heatbathsteps=1000
```

Code status

gaugefield.h

- ILDG storage format
- Gauge field is really the central object:
  - hmc trajectories
  - heatbath
  - inversion
- Gaugefield class: the place to manage inter-device communication

Code status

opencl.h

- Wraps all device operations
- `class opencl`

opencl Class Reference

`#include <opencl.h>`

List of all members.

Public Member Functions

	opencl (cl_device_type wanted, const size_t ls, const size_t gs, **usetimer** *timer, **inputparameters** *parameters)
	opencl ()
hmc_error	**init** (cl_device_type wanted_device_type, const size_t **local_work_size**, const size_t **global_work_size**, **usetimer** *timer, **inputparameters** *parameters)
hmc_error	**copy_gaugefield_to_device** (**hmc_gaugefield** *host_gaugefield, **usetimer** *timer)
hmc_error	**copy_rndarray_to_device** (**hmc_rndarray** host_rndarray, **usetimer** *timer)
hmc_error	**copy_rndarray_from_device** (**hmc_rndarray** rndarray, **usetimer** *timer)
hmc_error	**get_gaugefield_from_device** (**hmc_gaugefield** *host_gaugefield, **usetimer** *timer)
hmc_error	**run_heatbath** (**hmc_float** beta, const size_t **local_work_size**, const size_t **global_work_size**, **usetimer** *timer)
hmc_error	**run_overrelax** (**hmc_float** beta, const size_t **local_work_size**, const size_t **global_work_size**, **usetimer** *timer)
hmc_error	**gaugeobservables** (const size_t **local_work_size**, const size_t **global_work_size**, **hmc_float** *plaq, **hmc_float** *tplaq, **hmc_float** *splaq, **hmc_complex** *pol, **usetimer** *timer1, **usetimer** *timer2)
hmc_error	**finalize** ()

Public Attributes

std::vector< std::string >	**cl_kernels_file**

Code status

Random numbers

- Random numbers on GPU are tricky; avoid cross correlations between threads
- Vectorized, parallel RNG?!?
- Right now: NR generator
- Implement RanLux [Matthias]

Code status

Operations

- Implement operations for different types:

  host_operations_complex
  host_operations_fermionmatrix
  host_operations_gaugefield
  host_operations_spinor
  host_operations_spinorfield

- Maybe we can get that a bit more sorted?

[currently: Steo, Christian?]

31

Code status

Testing function

- Deprecated?

```
void testing_correlator(hmc_gaugefield* gf, inputparamete
void testing_fermionmatrix();
void testing_eoprec_spinor();
void print_fullspinorfield(hmc_spinor* in);

void testing_spinor();
void print_su3mat(hmc_su3matrix* A);
void print_staplemat(hmc_staplematrix* A);
void print_linkplusstaplematrix(hmc_gaugefield * in, int
void testing_su3mat();
void testing_gaugefield();
void testing_geometry();
void testing_su3matrix(hmc_gaugefield * in, int spacepos,
void testing_adjoin(hmc_gaugefield * in, int spacepos, in
void testing_det_global(hmc_gaugefield * in);
void testing_matrix_ops(hmc_gaugefield * in);

void testing_heatbath_norandommat_no123(hmc_su3matrix * s

void testing_heatbath_no123(hmc_su3matrix * su3_in, hmc_s
y, int * cter_out);

void testing_heatbath(hmc_su3matrix * su3_in, hmc_staplem
* cter_out);

void testing_colorvector_ops();
void testing_matrix_spinor_ops();
void testing_matrix_spinor_functions();
void testing_fermionmatrix_functions();

void unit_spinor(hmc_spinor * in);
void i_spinor(hmc_spinor * in);
void print_spinor(hmc_spinor * in);
void set_comp_to_one_spinor(hmc_spinor * in, int comp);
```

Code status

Compile time flags

- **USEDOUBLEPREC, USE_EOPREC**
- **RECONSTRUCT_TWELVE**
- **FERMIONS, HMC, BENCHMARKS**

➔ Untangle into different programs

Code status

SU(3) reconstruct

- **M. Clark et al.**

  Comput. Phys. Commun., 181:1517-1528, 2010

$$\begin{pmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \end{pmatrix} = \begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix}$$

$$\mathbf{c} = (\mathbf{a} \times \mathbf{b})^*$$

- `hmc_complex reconstruct_su3(hmc_su3matrix *in, int j);`

OpenCL

Device code

- `.cl` files read in by `opencl.cpp`
- Compilation at runtime
- `cl_kernelsfiles.cl` for debug info

OpenCL

Device code

```
/** @file
 * Inclusion and definition of types and definitions required in the Device code.
 */

//opencl_header.cl

#ifdef _USEDOUBLEPREC_
#pragma OPENCL EXTENSION cl_amd_fp64 : enable
//#pragma OPENCL EXTENSION cl_khr_fp64 : enable
#endif

#include "globaldefs.h" //NDIM, NSPIN, NC
#include "types.h"

//!!CP: why is this here?
// #define VOLSPACE NSPACE*NSPACE*NSPACE
#define VOL4D VOLSPACE*NTIME

//for hmc_ocl_su3matrix
#ifdef _RECONSTRUCT_TWELVE_
#define SU3SIZE NC*(NC-1)
#define STAPLEMATRIXSIZE NC*NC
#else
#define SU3SIZE NC*NC
```

OpenCL

Some remarks…

- Build-log
- Kernels, kernel args
- Pragmas (single, double)
- Optimise worksizes [Matthias…]

OpenCL

Some remarks…

```
init device #0
OpenCL being initialized...
        CL_PLATFORM_NAME:     ATI Stream
        CL_PLATFORM_VENDOR:   Advanced Micro Devices, Inc.
        CL_PLATFORM_VERSION:  OpenCL 1.1 ATI-Stream-v2.3 (451)

        1 device of wanted type has been found.
        Device information:
                CL_DEVICE_NAME:    Intel(R) Core(TM) i5 CPU       M 430  @ 2.27GHz
                CL_DEVICE_VENDOR:  GenuineIntel
                CL_DEVICE_TYPE:    CPU
                CL_DEVICE_VERSION: OpenCL 1.1 ATI-Stream-v2.3 (451)
                CL_DEVICE_EXTENSIONS: cl_amd_fp64 cl_khr_global_int32_base_atomics cl_khr_global_int32_extended_atomics
cl_khr_local_int32_base_atomics cl_khr_local_int32_extended_atomics cl_khr_int64_base_atomics cl_khr_int64_extended_atom
ics cl_khr_byte_addressable_store cl_khr_gl_sharing cl_ext_device_fission cl_amd_device_attribute_query cl_amd_media_ops
 cl_amd_popcnt cl_amd_printf
Create context...
Create command queue...
Read kernel source from file: /home/lars/clhmc_dev/clhmc/prog/opencl_header.cl
Read kernel source from file: /home/lars/clhmc_dev/clhmc/prog/opencl_geometry.cl
Read kernel source from file: /home/lars/clhmc_dev/clhmc/prog/opencl_random.cl
Read kernel source from file: /home/lars/clhmc_dev/clhmc/prog/opencl_operations_complex.cl
Read kernel source from file: /home/lars/clhmc_dev/clhmc/prog/opencl_operations_gaugefield.cl
Read kernel source from file: /home/lars/clhmc_dev/clhmc/prog/opencl_update_heatbath.cl
Read kernel source from file: /home/lars/clhmc_dev/clhmc/prog/opencl_gaugeobservables.cl
Create program...
Build program...
        build options:  -D_INKERNEL_ -DNSPACE=8 -DNTIME=4 -DVOLSPACE=512 -DSPINORSIZE=12 -DHALFSPINORSIZE=6 -DSPINORFIEL
DSIZE=24576 -DEOPREC_SPINORFIELDSIZE=12288 -D_USEDOUBLEPREC_ -I/home/lars/clhmc_dev/clhmc/prog
finished building program
Build Log:
Create buffer for gaugefield...
Create buffer for random numbers...
Create buffer for gaugeobservables...
Create heatbath kernels...
Create gaugeobservables kernels...
writing gaugefield to lime file
```

That's it

To do

- Gaugefield class [Lars]
- Inter-device strategy
- Device code (esp. hmc) [Christopher]
- Memory management [Matthias]

- OpenSource, GPL?