

# Efficient Implementation of High Performance Algorithms on Intel Xeon Phi

Generated by Doxygen 1.8.9.1

Tue May 19 2015 02:59:25



# Contents

<b>1</b>	<b>Main Page</b>	<b>1</b>
<b>2</b>	<b>CROSS-COMPILATION OF HDF5, SZIP, ZLIB AND GNU MAKE</b>	<b>3</b>
<b>3</b>	<b>FFT TEST (MKL) ON THE MIC</b>	<b>5</b>
<b>4</b>	<b>MATRIX MULTIPLICATION</b>	<b>7</b>
<b>5</b>	<b>MATRIX VECTOR MULTIPLICATION</b>	<b>9</b>
<b>6</b>	<b>N-BODY SIMULATION</b>	<b>11</b>
<b>7</b>	<b>NUMPY &amp; SCIPY DGEMM TEST</b>	<b>15</b>
<b>8</b>	<b>Namespace Index</b>	<b>17</b>
8.1	Namespace List . . . . .	17
<b>9</b>	<b>Class Index</b>	<b>19</b>
9.1	Class List . . . . .	19
<b>10</b>	<b>File Index</b>	<b>21</b>
10.1	File List . . . . .	21
<b>11</b>	<b>Namespace Documentation</b>	<b>23</b>
11.1	numpy_dgemm Namespace Reference . . . . .	23
11.1.1	Variable Documentation . . . . .	23
11.1.1.1	a1 . . . . .	23
11.1.1.2	a2 . . . . .	23
11.1.1.3	ITER . . . . .	23
11.1.1.4	K . . . . .	23
11.1.1.5	M . . . . .	24
11.1.1.6	m1 . . . . .	24
11.1.1.7	m2 . . . . .	24
11.1.1.8	mf2 . . . . .	24
11.1.1.9	N . . . . .	24

11.1.1.10 <code>t_end</code>	24
11.1.1.11 <code>t_init</code>	24
11.1.1.12 <code>t_start</code>	24
11.2 <code>scipy_dgemm</code> Namespace Reference	24
11.2.1 Variable Documentation	25
11.2.1.1 <code>a1</code>	25
11.2.1.2 <code>a2</code>	25
11.2.1.3 <code>ITER</code>	25
11.2.1.4 <code>K</code>	25
11.2.1.5 <code>M</code>	25
11.2.1.6 <code>m1</code>	25
11.2.1.7 <code>m2</code>	25
11.2.1.8 <code>mf2</code>	25
11.2.1.9 <code>N</code>	25
11.2.1.10 <code>t_end</code>	25
11.2.1.11 <code>t_start</code>	25
<b>12 Class Documentation</b>	<b>27</b>
12.1 <code>t_particle</code> Struct Reference	27
12.1.1 Detailed Description	27
12.1.2 Member Data Documentation	27
12.1.2.1 <code>pos_x</code>	27
12.1.2.2 <code>pos_y</code>	27
12.1.2.3 <code>pos_z</code>	27
12.1.2.4 <code>vel_x</code>	27
12.1.2.5 <code>vel_y</code>	28
12.1.2.6 <code>vel_z</code>	28
12.1.2.7 <code>weight</code>	28
12.2 <code>t_particles_DA</code> Struct Reference	28
12.2.1 Detailed Description	28
12.2.2 Member Data Documentation	28
12.2.2.1 <code>ax</code>	28
12.2.2.2 <code>ay</code>	28
12.2.2.3 <code>az</code>	29
12.2.2.4 <code>fx</code>	29
12.2.2.5 <code>fy</code>	29
12.2.2.6 <code>fz</code>	29
12.2.2.7 <code>pos_x</code>	29
12.2.2.8 <code>pos_y</code>	29
12.2.2.9 <code>pos_z</code>	29

12.2.2.10 vel_x . . . . .	29
12.2.2.11 vel_y . . . . .	29
12.2.2.12 vel_z . . . . .	29
12.2.2.13 weight . . . . .	29
<b>13 File Documentation</b>	<b>31</b>
13.1 cross-compilation/README.md File Reference . . . . .	31
13.2 fft-mkl/README.md File Reference . . . . .	31
13.3 matmul/README.md File Reference . . . . .	31
13.4 matvec/README.md File Reference . . . . .	31
13.5 nbody/README.md File Reference . . . . .	31
13.6 python-mkl/README.md File Reference . . . . .	31
13.7 README.md File Reference . . . . .	31
13.8 fft-mkl/cpu/fftw-3d-omp.c File Reference . . . . .	31
13.8.1 Detailed Description . . . . .	31
13.8.2 Function Documentation . . . . .	32
13.8.2.1 main . . . . .	32
13.9 fft-mkl/mic/fftw-3d-omp.c File Reference . . . . .	32
13.9.1 Detailed Description . . . . .	32
13.9.2 Function Documentation . . . . .	32
13.9.2.1 main . . . . .	32
13.10fft-mkl/mic/fftw-3d-omp-bpage.c File Reference . . . . .	32
13.10.1 Detailed Description . . . . .	33
13.10.2 Function Documentation . . . . .	33
13.10.2.1 main . . . . .	33
13.11matmul/cpu/c_dgemm.c File Reference . . . . .	33
13.11.1 Function Documentation . . . . .	33
13.11.1.1 main . . . . .	33
13.11.2 Variable Documentation . . . . .	33
13.11.2.1 ALPHA . . . . .	33
13.11.2.2 BETA . . . . .	33
13.12matmul/mic/c_dgemm.c File Reference . . . . .	33
13.12.1 Function Documentation . . . . .	34
13.12.1.1 main . . . . .	34
13.12.2 Variable Documentation . . . . .	34
13.12.2.1 ALPHA . . . . .	34
13.12.2.2 BETA . . . . .	34
13.13matvec/dynamic-aligned/main.cpp File Reference . . . . .	34
13.13.1 Function Documentation . . . . .	34
13.13.1.1 main . . . . .	34

13.13.1.2 mat_init . . . . .	35
13.13.1.3 vec_init . . . . .	35
13.13.1.4 vec_sum . . . . .	35
13.14matvec/naive/main.cpp File Reference . . . . .	35
13.14.1 Function Documentation . . . . .	36
13.14.1.1 main . . . . .	36
13.14.1.2 mat_init . . . . .	36
13.14.1.3 vec_init . . . . .	36
13.14.1.4 vec_sum . . . . .	36
13.15matvec/omp-parallel/main.cpp File Reference . . . . .	36
13.15.1 Function Documentation . . . . .	37
13.15.1.1 main . . . . .	37
13.15.1.2 mat_init . . . . .	37
13.15.1.3 vec_init . . . . .	37
13.15.1.4 vec_sum . . . . .	37
13.16matvec/omp-parallel-mic/main.cpp File Reference . . . . .	38
13.16.1 Function Documentation . . . . .	38
13.16.1.1 main . . . . .	38
13.16.1.2 mat_init . . . . .	38
13.16.1.3 vec_init . . . . .	38
13.16.1.4 vec_sum . . . . .	38
13.17matvec/vec-padding/main.cpp File Reference . . . . .	39
13.17.1 Function Documentation . . . . .	39
13.17.1.1 main . . . . .	39
13.17.1.2 mat_init . . . . .	39
13.17.1.3 vec_init . . . . .	39
13.17.1.4 vec_sum . . . . .	40
13.17.2 Variable Documentation . . . . .	40
13.17.2.1 COL_PAD . . . . .	40
13.18nbody/cache-block-mic/main.cpp File Reference . . . . .	40
13.18.1 Function Documentation . . . . .	40
13.18.1.1 main . . . . .	40
13.19nbody/naive/main.cpp File Reference . . . . .	40
13.19.1 Function Documentation . . . . .	41
13.19.1.1 main . . . . .	41
13.20nbody/no-jump-auto-opt/main.cpp File Reference . . . . .	41
13.20.1 Function Documentation . . . . .	41
13.20.1.1 main . . . . .	41
13.21nbody/offload/main.cpp File Reference . . . . .	41
13.21.1 Function Documentation . . . . .	42

13.21.1.1 main . . . . .	42
13.22nbody/omp-parallel/main.cpp File Reference . . . . .	42
13.22.1 Function Documentation . . . . .	42
13.22.1.1 main . . . . .	42
13.23nbody/omp-parallel-mic/main.cpp File Reference . . . . .	42
13.23.1 Function Documentation . . . . .	43
13.23.1.1 main . . . . .	43
13.24matvec/dynamic-aligned/matvec.cpp File Reference . . . . .	43
13.24.1 Function Documentation . . . . .	43
13.24.1.1 mat_vec_mul . . . . .	43
13.25matvec/naive/matvec.cpp File Reference . . . . .	43
13.25.1 Function Documentation . . . . .	44
13.25.1.1 mat_vec_mul . . . . .	44
13.26matvec/omp-parallel/matvec.cpp File Reference . . . . .	44
13.26.1 Function Documentation . . . . .	44
13.26.1.1 mat_vec_mul . . . . .	44
13.27matvec/omp-parallel-mic/matvec.cpp File Reference . . . . .	44
13.27.1 Function Documentation . . . . .	45
13.27.1.1 mat_vec_mul . . . . .	45
13.28matvec/vec-padding/matvec.cpp File Reference . . . . .	45
13.28.1 Function Documentation . . . . .	45
13.28.1.1 mat_vec_mul . . . . .	45
13.29matvec/dynamic-aligned/matvec.h File Reference . . . . .	45
13.29.1 Function Documentation . . . . .	46
13.29.1.1 mat_vec_mul . . . . .	46
13.30matvec/naive/matvec.h File Reference . . . . .	47
13.30.1 Function Documentation . . . . .	47
13.30.1.1 mat_vec_mul . . . . .	47
13.31matvec/omp-parallel/matvec.h File Reference . . . . .	47
13.31.1 Function Documentation . . . . .	47
13.31.1.1 mat_vec_mul . . . . .	47
13.32matvec/omp-parallel-mic/matvec.h File Reference . . . . .	48
13.32.1 Function Documentation . . . . .	48
13.32.1.1 mat_vec_mul . . . . .	48
13.33matvec/vec-padding/matvec.h File Reference . . . . .	48
13.33.1 Function Documentation . . . . .	48
13.33.1.1 mat_vec_mul . . . . .	48
13.34nbody/cache-block-mic/gen.cpp File Reference . . . . .	48
13.34.1 Function Documentation . . . . .	49
13.34.1.1 main . . . . .	49

13.34.1.2 randf . . . . .	49
13.35nbody/naive/gen.cpp File Reference . . . . .	49
13.35.1 Function Documentation . . . . .	49
13.35.1.1 main . . . . .	49
13.35.1.2 randf . . . . .	49
13.36nbody/no-jump-auto-opt/gen.cpp File Reference . . . . .	50
13.36.1 Function Documentation . . . . .	50
13.36.1.1 main . . . . .	50
13.36.1.2 randf . . . . .	50
13.37nbody/offload/gen.cpp File Reference . . . . .	50
13.37.1 Function Documentation . . . . .	50
13.37.1.1 main . . . . .	50
13.37.1.2 randf . . . . .	50
13.38nbody/omp-parallel/gen.cpp File Reference . . . . .	51
13.38.1 Function Documentation . . . . .	51
13.38.1.1 main . . . . .	51
13.38.1.2 randf . . . . .	51
13.39nbody/omp-parallel-mic/gen.cpp File Reference . . . . .	51
13.39.1 Function Documentation . . . . .	51
13.39.1.1 main . . . . .	51
13.39.1.2 randf . . . . .	51
13.40nbody/cache-block-mic/nbody.cpp File Reference . . . . .	52
13.40.1 Function Documentation . . . . .	52
13.40.1.1 particles_alloc . . . . .	52
13.40.1.2 particles_free . . . . .	52
13.40.1.3 particles_init . . . . .	52
13.40.1.4 particles_read . . . . .	53
13.40.1.5 particles_simulate . . . . .	53
13.40.1.6 particles_write . . . . .	53
13.40.2 Variable Documentation . . . . .	53
13.40.2.1 BLOCK . . . . .	53
13.40.2.2 SML_FLT . . . . .	53
13.41nbody/naive/nbody.cpp File Reference . . . . .	53
13.41.1 Function Documentation . . . . .	54
13.41.1.1 particles_read . . . . .	54
13.41.1.2 particles_simulate . . . . .	54
13.41.1.3 particles_write . . . . .	54
13.42nbody/no-jump-auto-opt/nbody.cpp File Reference . . . . .	54
13.42.1 Macro Definition Documentation . . . . .	55
13.42.1.1 SML_FLT . . . . .	55



13.42.2 Function Documentation	55
13.42.2.1 particles_alloc	55
13.42.2.2 particles_free	55
13.42.2.3 particles_init	55
13.42.2.4 particles_read	55
13.42.2.5 particles_simulate	56
13.42.2.6 particles_write	56
13.43nbody/offload/nbody.cpp File Reference	56
13.43.1 Function Documentation	56
13.43.1.1 __attribute__	56
13.43.1.2 particles_alloc	57
13.43.1.3 particles_free	57
13.43.1.4 particles_read	57
13.43.1.5 particles_write	57
13.44nbody/omp-parallel/nbody.cpp File Reference	57
13.44.1 Function Documentation	58
13.44.1.1 particles_alloc	58
13.44.1.2 particles_free	58
13.44.1.3 particles_init	58
13.44.1.4 particles_read	58
13.44.1.5 particles_simulate	59
13.44.1.6 particles_write	59
13.44.2 Variable Documentation	59
13.44.2.1 SML_FLT	59
13.45nbody/omp-parallel-mic/nbody.cpp File Reference	59
13.45.1 Function Documentation	60
13.45.1.1 particles_alloc	60
13.45.1.2 particles_free	60
13.45.1.3 particles_init	60
13.45.1.4 particles_read	60
13.45.1.5 particles_simulate	60
13.45.1.6 particles_write	60
13.45.2 Variable Documentation	61
13.45.2.1 SML_FLT	61
13.46nbody/cache-block-mic/nbody.h File Reference	61
13.46.1 Macro Definition Documentation	61
13.46.1.1 G	61
13.46.2 Function Documentation	61
13.46.2.1 particles_alloc	61
13.46.2.2 particles_free	62

13.46.2.3 particles_init . . . . .	62
13.46.2.4 particles_read . . . . .	62
13.46.2.5 particles_simulate . . . . .	62
13.46.2.6 particles_write . . . . .	62
13.47nbody/naive/nbody.h File Reference . . . . .	63
13.47.1 Macro Definition Documentation . . . . .	63
13.47.1.1 G . . . . .	63
13.47.2 Typedef Documentation . . . . .	63
13.47.2.1 t_particles . . . . .	63
13.47.3 Function Documentation . . . . .	63
13.47.3.1 particles_read . . . . .	63
13.47.3.2 particles_simulate . . . . .	64
13.47.3.3 particles_write . . . . .	64
13.48nbody/no-jump-auto-opt/nbody.h File Reference . . . . .	64
13.48.1 Macro Definition Documentation . . . . .	65
13.48.1.1 G . . . . .	65
13.48.2 Function Documentation . . . . .	65
13.48.2.1 particles_alloc . . . . .	65
13.48.2.2 particles_free . . . . .	65
13.48.2.3 particles_init . . . . .	65
13.48.2.4 particles_read . . . . .	65
13.48.2.5 particles_simulate . . . . .	66
13.48.2.6 particles_write . . . . .	67
13.49nbody/offload/nbody.h File Reference . . . . .	67
13.49.1 Macro Definition Documentation . . . . .	67
13.49.1.1 G . . . . .	67
13.49.2 Function Documentation . . . . .	67
13.49.2.1 __attribute__ . . . . .	67
13.49.2.2 particles_alloc . . . . .	68
13.49.2.3 particles_free . . . . .	68
13.49.2.4 particles_read . . . . .	68
13.49.2.5 particles_write . . . . .	68
13.50nbody/omp-parallel/nbody.h File Reference . . . . .	69
13.50.1 Macro Definition Documentation . . . . .	69
13.50.1.1 G . . . . .	69
13.50.2 Function Documentation . . . . .	69
13.50.2.1 particles_alloc . . . . .	69
13.50.2.2 particles_free . . . . .	70
13.50.2.3 particles_init . . . . .	71
13.50.2.4 particles_read . . . . .	71

---

13.50.2.5 particles_simulate . . . . .	71
13.50.2.6 particles_write . . . . .	71
13.51 nbody/omp-parallel-mic/nbody.h File Reference . . . . .	71
13.51.1 Macro Definition Documentation . . . . .	72
13.51.1.1 G . . . . .	72
13.51.2 Function Documentation . . . . .	72
13.51.2.1 particles_alloc . . . . .	72
13.51.2.2 particles_free . . . . .	72
13.51.2.3 particles_init . . . . .	72
13.51.2.4 particles_read . . . . .	73
13.51.2.5 particles_simulate . . . . .	73
13.51.2.6 particles_write . . . . .	73
13.52 python-mkl/numpy_dgemm.py File Reference . . . . .	73
13.53 python-mkl/scipy_dgemm.py File Reference . . . . .	74
<b>Index</b>	<b>75</b>



# Chapter 1

## Main Page

Bachelor thesis

Dominik Simek [xsimek23@stud.fit.vutbr.cz](mailto:xsimek23@stud.fit.vutbr.cz)

FIT VUT 2015

### Abstract

This thesis is dedicated to the implementation of high performance algorithms on the Intel Xeon Phi coprocessor. The Xeon phi was introduced by Intel as a new MIC (Many Integrated Core) architecture in 2012. The theoretical part of the thesis is focused on the architecture of the coprocessor (with peak performance of 2 tFLOPS for a single precision data) and on the procedure of algorithms implementation and optimization. The theoretical knowledge is then applied to a practical examples with demonstration of the implementation and the optimization of algorithms and work with the coprocessor. In the practical part of the thesis, simple benchmarks such as a vector matrix multiplication and a matrix multiplication are explained and implemented. In the first benchmark 6.5% of theoretical coprocessor performance was achieved, in the second it was much more. In following chapter a more complex benchmark, simulation of a particles system (N-Body), that reached more than 35% of coprocessor performance (725 gFLOPS), is discussed. The following section is dedicated to some interesting problems such as optimization of a MATLAB module k-Wave (propagation of the ultrasound waves), extraction of I-vector (speech processing), cross-compilation of existing libraries, modules and programs. In the conclusion of the thesis the usage the potential of the Intel Xeon Phi is evaluated.

### Repository structure

```
1 -.
2 |-cross-compilation/
3 |-Doxyfile
4 |-fft-mkl/
5 |-matmul/
6 |-matvec/
7 |-nbody/
8 |-python-mkl/
9 |-README.md
10 |-text/
```

### Description of each directory

- cross-compilation:
  - the directory contains a instructions to the cross-compilation of the libraries HDF5, SZIP, ZLIB and GNU MAKE
  - see [cross-compilation/README.md](#) for more informations
- fft-mkl:

- the directory contains the source code of a test of fft routines from the Intel MKL on the MIC
  - see [fft-mkl/README.md](#) for more informations
- matmul:
  - the directory contains the source code of the matrix multiplication (MKL)
  - see [matmul/README.md](#) for more informations
- matvec:
  - the directory contains the source code of the matrix vector multiplication
  - see [matvec/README.md](#) for more informations
- nbody:
  - the directory contains the source code of the N-Body simulation
  - see [nbody/README.md](#) for more informations
- python-mkl:
  - the directory contains the source code of the DGEMM (Python modules Numpy and Scipy lined with MKL)
  - see [python-mkl/README.md](#) for more informations
- text:
  - Latex source code of the thesis

## Chapter 2

# CROSS-COMPILATION OF HDF5, SZIP, ZLIB AND GNU MAKE

Dominik Simek [xsimek23@fit.vutbr.cz](mailto:xsimek23@fit.vutbr.cz)

Bachelor thesis

FIT VUT 2015

Basic informations

- each .txt file contains instructions of a cross-compilation of some library





## Chapter 3

# FFT TEST (MKL) ON THE MIC

Dominik Simek [xsimek23@fit.vutbr.cz](mailto:xsimek23@fit.vutbr.cz)

Bachelor thesis

FIT VUT 2015

### Basic informations

- all directories have a same structure:

```
1 -.
2 |-fftw-3d-omp.c (source file)
3 |-Makefile (make, make run, make clean)
4 |-papi_cntr.h (simpler work with PAPI)
5 |-run_test.sh (run program for various matrix sizes)
```

- if you are working on the Anselm:

- create interactive job on a MIC node  
qsub -l -q qmic -A PROJECT-ID
- load necessary modules  
module load intel  
module load papi

- usage:

- cd /ffw-mkl/cpu
- make # create executable file
- make run # run computation
- make clean # clean up
- or  
./fftw-3d-omp MATRIX\_DIMENSION (e.g. \$ ./fftw-3d-omp 128)
- or  
./run\_test.sh # run bash script

### A description of each directories

- cpu:
  - implementation for the cpu
  - set a number of threads  
export OMP\_SET\_NUM\_THREADS=8

- run for matrixes of size 128  
make run
  - directory contains also bash script run\_test.sh
  - run it for various matrix sizes and various number of threads  
./run\_test.sh
- mic:
    - compile program on a host  
[host]\$ make
    - and run it on the MIC  
[host]\$ ssh mic  
[mic0]\$ # MIC shared libraries (it can be different for specific system)  
[mic0]\$ export LD\_LIBRARY\_PATH=/apps/intel/composer\_xe\_2015.2.164/compiler/lib/mic:\$LD\_LIBRARY\_PATH  
[mic0]\$ export LD\_LIBRARY\_PATH=/apps/intel/composer\_xe\_2015.2.164/mkl/lib/mic/:\$LD\_LIBRARY\_PATH  
[mic0]\$ cd fftw-mkl/mic  
[mic0]\$ export OMP\_NUM\_THREADS=120 # 60, 120, 180, 240  
[mic0]\$ ./fftw-3d-omp 2048
    - directory contains also bash script run\_test.sh
    - run it for various matrix sizes and various number of threads  
bash run\_test.sh

## Chapter 4

# MATRIX MULTIPLICATION

Dominik Simek [xsimek23@fit.vutbr.cz](mailto:xsimek23@fit.vutbr.cz)

Bachelor thesis

FIT VUT 2015

### Basic informations

- all directories have a same structure:

```
1 - .
2 | -c_dgemm.c (source file)
3 | -Makefile (make, make run, make clean)
4 | -papi_cntr.h (simpler work with PAPI)
5 | -run_c.sh (run program for various matrix sizes)
```

- if you are working on the Anselm:

- create interactive job on a MIC node  
qsub -I -q qmic -A PROJECT-ID
- load necessary modules  
module load intel

- usage:

- cd cpu
- make # create executable file
- make run # run computation
- make clean # clean up
- or  
./c\_dgemm MATRIX\_DIMENSION (e.g. \$ ./c\_dgemm 2048)
- or  
./run\_c.sh # run bash script

### A description of each directories

- cpu:
  - implementation for the cpu
  - set a number of threads  
export OMP\_SET\_NUM\_THREADS=8

- run for matrixes of size 1024  
make run
- directory contains also bash script run\_c.sh
- run it for various matrix sizes  
./run\_c.sh
- mic:
  - compile program on a host  
[host]\$ make
  - and run it on the MIC  
[host]\$ ssh mic  
[mic0]\$ # MIC shared libraries (it can be different for specific system)  
[mic0]\$ export LD\_LIBRARY\_PATH=/apps/intel/composer\_xe\_2015.2.164/compiler/lib/mic:\$LD\_LIBRARY\_PATH  
[mic0]\$ export LD\_LIBRARY\_PATH=/apps/intel/composer\_xe\_2015.2.164/mkl/lib/mic/:\$LD\_LIBRARY\_PATH  
[mic0]\$ cd matmul/mic  
[mic0]\$ export OMP\_NUM\_THREADS=120 # 60, 120, 180, 240  
[mic0]\$ ./c\_dgemm 2048
  - directory contains also bash script run\_c.sh
  - run it for various matrix sizes  
bash run\_c.sh

## Chapter 5

# MATRIX VECTOR MULTIPLICATION

Dominik Simek [xsimek23@fit.vutbr.cz](mailto:xsimek23@fit.vutbr.cz)

Bachelor thesis

FIT VUT 2015

### Basic informations

- all directories have a same structure:

```
1 - .
2 |-main.cpp (main function)
3 |-Makefile (make, make run, make clean)
4 |-matvec.cpp (matvec kernel)
5 |-matvec.h (matvec header file)
6 |-papi_cntr.h (simpler work with PAPI)
```

- if you are working on the Anselm:

- create interactive job on a MIC node  
qsub -l -q qmic -A PROJECT-ID
- load necessary modules  
module load intel  
module load papi

- in the Makefile you can set:

- ROWS, COLS (matrix size)
- RUNS (number of repetitions)
- PADDING (implemented only in the vec-padding version !)
- PAPI\_EVENTS (set PAPI counters to monitoring of HW events)

- usage:

- cd naive # e.g.
- make # create executable file
- make run # run computation
- make clean # clean up

## A description of each directories

- naive:
  - naive implementation of algorithm
  - no vectorization, no optimizations
- vec-padding:
  - vectorization, compiler optimizations
- dyn-aligned:
  - dynamic allocation
  - data aligned to 64B
- omp-parallel:
  - parallel version (OpenMP)
  - set number of threads  
export OMP\_NUM\_THREADS=N
  - you can also set binding of threads  
export KMP\_AFFINITY=compact|scatter
  - make run
  - directory contains also bash script run\_exp.sh
    - \* compile the program with your favourite matrix size at first
    - \* run the script  
./run\_exp.sh
    - \* this will run the program gradually on 1, 2, 4, 8 and 16 threads
- omp-parallel-mic:
  - parallel version (OpenMP) for the MIC
  - compile program on a host  
[host]\$ make
  - and run it on the MIC  
[host]\$ ssh mic  
[mic0]\$ # MIC shared libraries (it can be different for specific system)  
[mic0]\$ export LD\_LIBRARY\_PATH=/apps/intel/composer\_xe\_2015.2.164/compiler/lib/mic:\$LD\_LIBRARY\_PATH  
[mic0]\$ export LD\_LIBRARY\_PATH=/apps/intel/composer\_xe\_2015.2.164/mkl/lib/mic:\$LD\_LIBRARY\_PATH  
[mic0]\$ cd naive  
[mic0]\$ export OMP\_NUM\_THREADS=120 # 60, 120, 180, 240  
[mic0]\$ ./matvec
  - directory contains also bash script run\_exp.sh
    - \* compile the program with your favourite matrix size at first
    - \* run the script  
bash run\_exp.sh
    - \* this will run the program gradually on 1, 2, 4, 8, 16, 30, 60, 120, 180 and 240 threads

## Chapter 6

# N-BODY SIMULATION

Dominik Simek [xsimek23@fit.vutbr.cz](mailto:xsimek23@fit.vutbr.cz)

Bachelor thesis

FIT VUT 2015

### Basic informations

- all directories have a same structure:

```
1 - .
2 |-gen.cpp (generate input file)
3 |-main.cpp (main function)
4 |-Makefile (make, make run, make clean)
5 |-nbody.cpp (nbody kernel)
6 |-nbody.h (nbody header file)
7 |-papi_cntr.h (simpler work with PAPI)
```

- if you are working on the Anselm:
  - create interactive job on a MIC node  
qsub -l -q qmic -A PROJECT-ID
  - load necessary modules  
module load intel  
module load papi
- in the Makefile you can set:
  - N (number of particles)
  - STEPS (number of simulation steps)
  - DT (time between steps)
  - PAPI\_EVENTS (set PAPI counters to monitoring of HW events)
- usage:
  - cd naive # e.g.
  - make # create executable file
  - make run # run computation
  - make clean # clean up
- Makefile generate also executable file 'gen':
  - you can generate own input file (with "random" particle attributes)  
./gen N input.dat # N is number of particles

## A description of each directories

- naive:
  - naive implementation of algorithm
  - bad structure of loop
- no-jump-auto-opt:
  - enhanced loop structure
  - no branches
  - dynamic allocation
  - data aligned to 64B
  - compiler optimizations
- omp-parallel:
  - parallel version (OpenMP)
  - set number of threads  
export OMP\_NUM\_THREADS=N
  - you can also set binding of threads  
export KMP\_AFFINITY=compact|scatter
  - make run
  - directory contains also bash script run\_exp.sh
    - \* compile the program with your favourite matrix size at first
    - \* run the script  
./run\_exp.sh
    - \* this will run the program gradually on 1, 2, 4, 8 and 16 threads
- omp-parallel-mic:
  - parallel version (OpenMP) for the MIC
  - compile program on a host  
[host]\$ make
  - and run it on the MIC  
[host]\$ ssh mic  
[mic0]\$ # MIC shared libraries (it can be different for specific system)  
[mic0]\$ export LD\_LIBRARY\_PATH=/apps/intel/composer\_xe\_2015.2.164/compiler/lib/mic:\$LD\_LIBRARY\_PATH  
[mic0]\$ export LD\_LIBRARY\_PATH=/apps/intel/composer\_xe\_2015.2.164/mkl/lib/mic:\$LD\_LIBRARY\_PATH  
[mic0]\$ cd naive  
[mic0]\$ export OMP\_NUM\_THREADS=120 # 60, 120, 180, 240  
[mic0]\$ ./nbody ../input/1m.dat ../output/omp-mic-out.dat
  - directory contains also bash script run\_exp.sh
    - \* compile the program with your favourite matrix size at first
    - \* run the script  
bash run\_exp.sh
    - \* this will run the program gradually on 1, 2, 4, 8, 16, 30, 60, 120, 180 and 240 threads
- cache-block-mic:
  - effort to better cache utilization
  - you gave to set a correct number of particles and a correct block size
    - \* the block size have to be set in the file nbody.cpp, constant BLOCK



- \* e.g.
  - `const int BLOCK = 2048;` (nbody.cpp)
  - `N=1208320` (in Makefile)
  - now there are lot of bodies - data are bigger than caches
  - $1208320/2048=590$  (590 blocks of particles, it must be integer !)
- to compile and run use process from omp-parallel-mic paragraph
- `make; ssh mic0; ...`
- offload:
  - offload mode
  - you can set number of a MIC threads directly on the source code (main.cpp)  
`const int mic_threads = 120;`
  - or by enviroment variable  
`export MIC_ENV_PREFIX=MIC`  
`export MIC_OMP_NUM_THREADS=120`
  - compile and run on the host  
`[host]$ make [host]$ make run`
- input:
  - input files with particles attributes
- output:
  - output files are created here (after make run)



## Chapter 7

# NUMPY & SCIPY DGEMM TEST

Dominik Simek [xsimek23@fit.vutbr.cz](mailto:xsimek23@fit.vutbr.cz)

Bachelor thesis

FIT VUT 2015

### Basic informations

- test DGEMM from numpy/scipy modules (on the CPU)
- for best performance, numpy/scipy module have to be linked with Intel MKL
- usage:
  - set number of threads  
export OMP\_NUM\_THREADS=8
  - run python script  
python [numpy\\_dgemm.py](#) MATRIX\_DIMENSION  
python [numpy\\_dgemm.py](#) MATRIX\_DIMENSION
  - e.g.  
python [numpy\\_dgemm.py](#) 2048



## Chapter 8

# Namespace Index

### 8.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">numpy_dgemm</a>	23
<a href="#">scipy_dgemm</a>	24



## Chapter 9

# Class Index

### 9.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">t_particle</a>	.....	<a href="#">27</a>
<a href="#">t_particles_DA</a>	.....	<a href="#">28</a>





## Chapter 10

# File Index

### 10.1 File List

Here is a list of all files with brief descriptions:

<a href="#">fft-mkl/cpu/fftw-3d-omp.c</a>	31
<a href="#">fft-mkl/mic/fftw-3d-omp-bpage.c</a>	32
<a href="#">fft-mkl/mic/fftw-3d-omp.c</a>	32
<a href="#">matmul/cpu/c_dgemm.c</a>	33
<a href="#">matmul/mic/c_dgemm.c</a>	33
<a href="#">matvec/dynamic-aligned/main.cpp</a>	34
<a href="#">matvec/dynamic-aligned/matvec.cpp</a>	43
<a href="#">matvec/dynamic-aligned/matvec.h</a>	45
<a href="#">matvec/naive/main.cpp</a>	35
<a href="#">matvec/naive/matvec.cpp</a>	43
<a href="#">matvec/naive/matvec.h</a>	47
<a href="#">matvec/omp-parallel-mic/main.cpp</a>	38
<a href="#">matvec/omp-parallel-mic/matvec.cpp</a>	44
<a href="#">matvec/omp-parallel-mic/matvec.h</a>	48
<a href="#">matvec/omp-parallel/main.cpp</a>	36
<a href="#">matvec/omp-parallel/matvec.cpp</a>	44
<a href="#">matvec/omp-parallel/matvec.h</a>	47
<a href="#">matvec/vec-padding/main.cpp</a>	39
<a href="#">matvec/vec-padding/matvec.cpp</a>	45
<a href="#">matvec/vec-padding/matvec.h</a>	48
<a href="#">nbody/cache-block-mic/gen.cpp</a>	48
<a href="#">nbody/cache-block-mic/main.cpp</a>	40
<a href="#">nbody/cache-block-mic/nbody.cpp</a>	52
<a href="#">nbody/cache-block-mic/nbody.h</a>	61
<a href="#">nbody/naive/gen.cpp</a>	49
<a href="#">nbody/naive/main.cpp</a>	40
<a href="#">nbody/naive/nbody.cpp</a>	53
<a href="#">nbody/naive/nbody.h</a>	63
<a href="#">nbody/no-jump-auto-opt/gen.cpp</a>	50
<a href="#">nbody/no-jump-auto-opt/main.cpp</a>	41
<a href="#">nbody/no-jump-auto-opt/nbody.cpp</a>	54
<a href="#">nbody/no-jump-auto-opt/nbody.h</a>	64
<a href="#">nbody/offload/gen.cpp</a>	50
<a href="#">nbody/offload/main.cpp</a>	41
<a href="#">nbody/offload/nbody.cpp</a>	56
<a href="#">nbody/offload/nbody.h</a>	67
<a href="#">nbody/omp-parallel-mic/gen.cpp</a>	51
<a href="#">nbody/omp-parallel-mic/main.cpp</a>	42

nbody/omp-parallel-mic/ <a href="#">nbody.cpp</a>	59
nbody/omp-parallel-mic/ <a href="#">nbody.h</a>	71
nbody/omp-parallel/ <a href="#">gen.cpp</a>	51
nbody/omp-parallel/ <a href="#">main.cpp</a>	42
nbody/omp-parallel/ <a href="#">nbody.cpp</a>	57
nbody/omp-parallel/ <a href="#">nbody.h</a>	69
python-mkl/ <a href="#">numpy_dgemm.py</a>	73
python-mkl/ <a href="#">scipy_dgemm.py</a>	74

# Chapter 11

## Namespace Documentation

### 11.1 numpy\_dgemm Namespace Reference

#### Variables

- tuple `K` = `int(sys.argv[1])`
- `M` = `K`
- `N` = `K`
- int `ITER` = 1
- tuple `t_init` = `time.time()`
- tuple `a1` = `np.array(np.random.random((M, K)), dtype=np.double, order='C', copy=False)`
- tuple `a2` = `np.array(np.random.random((K, N)), dtype=np.double, order='C', copy=False)`
- tuple `m1` = `np.matrix(a1, dtype=np.double, copy=False)`
- tuple `m2` = `np.matrix(a2, dtype=np.double, copy=False)`
- tuple `t_start` = `time.time()`
- tuple `mf2` = `np.dot(m1, m2)`
- tuple `t_end` = `time.time()`

#### 11.1.1 Variable Documentation

11.1.1.1 tuple `numpy_dgemm.a1` = `np.array(np.random.random((M, K)), dtype=np.double, order='C', copy=False)`

Definition at line 25 of file `numpy_dgemm.py`.

11.1.1.2 tuple `numpy_dgemm.a2` = `np.array(np.random.random((K, N)), dtype=np.double, order='C', copy=False)`

Definition at line 26 of file `numpy_dgemm.py`.

11.1.1.3 int `numpy_dgemm.ITER` = 1

Definition at line 18 of file `numpy_dgemm.py`.

11.1.1.4 tuple `numpy_dgemm.K` = `int(sys.argv[1])`

Definition at line 15 of file `numpy_dgemm.py`.

#### 11.1.1.5 `numpy_dgemm.M = K`

Definition at line 16 of file `numpy_dgemm.py`.

#### 11.1.1.6 `tuple numpy_dgemm.m1 = np.matrix(a1, dtype=np.double, copy=False)`

Definition at line 28 of file `numpy_dgemm.py`.

#### 11.1.1.7 `tuple numpy_dgemm.m2 = np.matrix(a2, dtype=np.double, copy=False)`

Definition at line 29 of file `numpy_dgemm.py`.

#### 11.1.1.8 `tuple numpy_dgemm.mf2 = np.dot(m1, m2)`

Definition at line 34 of file `numpy_dgemm.py`.

#### 11.1.1.9 `numpy_dgemm.N = K`

Definition at line 17 of file `numpy_dgemm.py`.

#### 11.1.1.10 `tuple numpy_dgemm.t_end = time.time()`

Definition at line 37 of file `numpy_dgemm.py`.

#### 11.1.1.11 `tuple numpy_dgemm.t_init = time.time()`

Definition at line 23 of file `numpy_dgemm.py`.

#### 11.1.1.12 `tuple numpy_dgemm.t_start = time.time()`

Definition at line 31 of file `numpy_dgemm.py`.

## 11.2 `scipy_dgemm` Namespace Reference

### Variables

- `tuple K = int(sys.argv[1])`
- `M = K`
- `N = K`
- `int ITER = 1`
- `tuple a1 = np.array(np.random.random((M, K)), dtype=np.double, order='C', copy=False)`
- `tuple a2 = np.array(np.random.random((K, N)), dtype=np.double, order='C', copy=False)`
- `tuple m1 = np.matrix(a1, dtype=np.double, copy=False)`
- `tuple m2 = np.matrix(a2, dtype=np.double, copy=False)`
- `tuple t_start = time.time()`
- `tuple mf2 = sp.dgemm(alpha=1.0, a=m1, b=m2)`
- `tuple t_end = time.time()`

### 11.2.1 Variable Documentation

11.2.1.1 `tuple scipy_dgemm.a1 = np.array(np.random.random((M, K)), dtype=np.double, order='C', copy=False)`

Definition at line 24 of file `scipy_dgemm.py`.

11.2.1.2 `tuple scipy_dgemm.a2 = np.array(np.random.random((K, N)), dtype=np.double, order='C', copy=False)`

Definition at line 25 of file `scipy_dgemm.py`.

11.2.1.3 `int scipy_dgemm.ITER = 1`

Definition at line 19 of file `scipy_dgemm.py`.

11.2.1.4 `tuple scipy_dgemm.K = int(sys.argv[1])`

Definition at line 16 of file `scipy_dgemm.py`.

11.2.1.5 `scipy_dgemm.M = K`

Definition at line 17 of file `scipy_dgemm.py`.

11.2.1.6 `tuple scipy_dgemm.m1 = np.matrix(a1, dtype=np.double, copy=False)`

Definition at line 27 of file `scipy_dgemm.py`.

11.2.1.7 `tuple scipy_dgemm.m2 = np.matrix(a2, dtype=np.double, copy=False)`

Definition at line 28 of file `scipy_dgemm.py`.

11.2.1.8 `tuple scipy_dgemm.mf2 = sp.dgemm(alpha=1.0, a=m1, b=m2)`

Definition at line 35 of file `scipy_dgemm.py`.

11.2.1.9 `scipy_dgemm.N = K`

Definition at line 18 of file `scipy_dgemm.py`.

11.2.1.10 `tuple scipy_dgemm.t_end = time.time()`

Definition at line 37 of file `scipy_dgemm.py`.

11.2.1.11 `tuple scipy_dgemm.t_start = time.time()`

Definition at line 30 of file `scipy_dgemm.py`.



# Chapter 12

## Class Documentation

### 12.1 t\_particle Struct Reference

```
#include <nbody.h>
```

#### Public Attributes

- float [pos\\_x](#)
- float [pos\\_y](#)
- float [pos\\_z](#)
- float [vel\\_x](#)
- float [vel\\_y](#)
- float [vel\\_z](#)
- float [weight](#)

#### 12.1.1 Detailed Description

Definition at line 22 of file nbody.h.

#### 12.1.2 Member Data Documentation

##### 12.1.2.1 float t\_particle::pos\_x

Definition at line 24 of file nbody.h.

##### 12.1.2.2 float t\_particle::pos\_y

Definition at line 25 of file nbody.h.

##### 12.1.2.3 float t\_particle::pos\_z

Definition at line 26 of file nbody.h.

##### 12.1.2.4 float t\_particle::vel\_x

Definition at line 27 of file nbody.h.

#### 12.1.2.5 float t\_particle::vel\_y

Definition at line 28 of file nbody.h.

#### 12.1.2.6 float t\_particle::vel\_z

Definition at line 29 of file nbody.h.

#### 12.1.2.7 float t\_particle::weight

Definition at line 30 of file nbody.h.

The documentation for this struct was generated from the following file:

- nbody/naive/[nbody.h](#)

## 12.2 t\_particles\_DA Struct Reference

```
#include <nbody.h>
```

### Public Attributes

- float \* [pos\\_x](#)
- float \* [pos\\_y](#)
- float \* [pos\\_z](#)
- float \* [vel\\_x](#)
- float \* [vel\\_y](#)
- float \* [vel\\_z](#)
- float \* [weight](#)
- float \* [fx](#)
- float \* [fy](#)
- float \* [fz](#)
- float \* [ax](#)
- float \* [ay](#)
- float \* [az](#)

### 12.2.1 Detailed Description

Definition at line 23 of file nbody.h.

### 12.2.2 Member Data Documentation

#### 12.2.2.1 float \* t\_particles\_DA::ax

Definition at line 35 of file nbody.h.

#### 12.2.2.2 float \* t\_particles\_DA::ay

Definition at line 36 of file nbody.h.



### 12.2.2.3 float \* t\_particles\_DA::az

Definition at line 37 of file nbody.h.

### 12.2.2.4 float \* t\_particles\_DA::fx

Definition at line 32 of file nbody.h.

### 12.2.2.5 float \* t\_particles\_DA::fy

Definition at line 33 of file nbody.h.

### 12.2.2.6 float \* t\_particles\_DA::fz

Definition at line 34 of file nbody.h.

### 12.2.2.7 float \* t\_particles\_DA::pos\_x

Definition at line 25 of file nbody.h.

### 12.2.2.8 float \* t\_particles\_DA::pos\_y

Definition at line 26 of file nbody.h.

### 12.2.2.9 float \* t\_particles\_DA::pos\_z

Definition at line 27 of file nbody.h.

### 12.2.2.10 float \* t\_particles\_DA::vel\_x

Definition at line 28 of file nbody.h.

### 12.2.2.11 float \* t\_particles\_DA::vel\_y

Definition at line 29 of file nbody.h.

### 12.2.2.12 float \* t\_particles\_DA::vel\_z

Definition at line 30 of file nbody.h.

### 12.2.2.13 float \* t\_particles\_DA::weight

Definition at line 31 of file nbody.h.

The documentation for this struct was generated from the following file:

- nbody/cache-block-mic/[nbody.h](#)



## Chapter 13

# File Documentation

### 13.1 cross-compilation/README.md File Reference

### 13.2 fft-mkl/README.md File Reference

### 13.3 matmul/README.md File Reference

### 13.4 matvec/README.md File Reference

### 13.5 nbody/README.md File Reference

### 13.6 python-mkl/README.md File Reference

### 13.7 README.md File Reference

### 13.8 fft-mkl/cpu/fftw-3d-omp.c File Reference

```
#include <stdlib.h>
#include <fftw3.h>
#include <omp.h>
#include <mkl.h>
#include "papi_cntr.h"
```

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 13.8.1 Detailed Description

Simple FFTW benchmark Test FFTW performance on the Intel Xeon processor MKL version

**Author**

Dominik Simek [xsimek23@stud.fit.vutbr.cz](mailto:xsimek23@stud.fit.vutbr.cz) Bachelor thesis FIT VUT 2015

**13.8.2 Function Documentation****13.8.2.1 int main ( int argc, char \*\* argv )**

Definition at line 19 of file fftw-3d-omp.c.

**13.9 fft-mkl/mic/fftw-3d-omp.c File Reference**

```
#include <stdlib.h>
#include <fftw3.h>
#include <omp.h>
#include <mkl.h>
```

**Functions**

- int [main](#) (int argc, char \*\*argv)

**13.9.1 Detailed Description**

Simple FFTW benchmark Test FFTW performance on the Intel Xeon Phi MKL version Compare with Intel Xeon performance

**Author**

Dominik Simek [xsimek23@stud.fit.vutbr.cz](mailto:xsimek23@stud.fit.vutbr.cz) Bachelor thesis FIT VUT 2015

**13.9.2 Function Documentation****13.9.2.1 int main ( int argc, char \*\* argv )**

Definition at line 19 of file fftw-3d-omp.c.

**13.10 fft-mkl/mic/fftw-3d-omp-bpage.c File Reference**

```
#include <stdlib.h>
#include <fftw3.h>
#include <omp.h>
#include <mkl.h>
#include <sys/mman.h>
```

**Functions**

- int [main](#) (int argc, char \*\*argv)

### 13.10.1 Detailed Description

Simple fftwf benchmark Test fftwf performance at Intel Xeon Phi This use MKL Compare with Intel Xeon performance

### 13.10.2 Function Documentation

#### 13.10.2.1 `int main ( int argc, char ** argv )`

Definition at line 16 of file `fftw-3d-omp-bpage.c`.

## 13.11 matmul/cpu/c\_dgemm.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "mkl.h"
#include <omp.h>
#include "papi_cntr.h"
```

### Functions

- `int main (int argc, char *argv[ ])`

### Variables

- `const double ALPHA = 1.0`
- `const double BETA = 0.0`

### 13.11.1 Function Documentation

#### 13.11.1.1 `int main ( int argc, char * argv[ ] )`

Definition at line 22 of file `c_dgemm.c`.

### 13.11.2 Variable Documentation

#### 13.11.2.1 `const double ALPHA = 1.0`

Definition at line 19 of file `c_dgemm.c`.

#### 13.11.2.2 `const double BETA = 0.0`

Definition at line 20 of file `c_dgemm.c`.

## 13.12 matmul/mic/c\_dgemm.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "mkl.h"
#include <omp.h>
```

## Functions

- int `main` (int argc, char \*argv[])

## Variables

- const double `ALPHA` = 1.0
- const double `BETA` = 0.0

### 13.12.1 Function Documentation

#### 13.12.1.1 int main ( int argc, char \* argv[] )

Definition at line 21 of file `c_dgemm.c`.

### 13.12.2 Variable Documentation

#### 13.12.2.1 const double ALPHA = 1.0

Definition at line 18 of file `c_dgemm.c`.

#### 13.12.2.2 const double BETA = 0.0

Definition at line 19 of file `c_dgemm.c`.

## 13.13 matvec/dynamic-aligned/main.cpp File Reference

```
#include <stdio>
#include <cmath>
#include "matvec.h"
#include "papi_cntr.h"
```

## Functions

- void `mat_init` (int row, int col, float off, float \*a)
- void `vec_init` (int length, float off, float \*a)
- double `vec_sum` (int length, float \*vec)
- int `main` (int argc, char \*\*argv)

### 13.13.1 Function Documentation

#### 13.13.1.1 int main ( int argc, char \*\* argv )

Main function

## Parameters

<i>argc</i>	
<i>argv</i>	

Definition at line 71 of file main.cpp.

13.13.1.2 void mat\_init ( int *row*, int *col*, float *off*, float \* *a* )

Initialize matrix with "random" values

## Parameters

<i>row</i>	- matrix rows
<i>col</i>	- matrix col
<i>off</i>	- auxiliary offset
<i>a</i>	- matrix to initialize

Definition at line 26 of file main.cpp.

13.13.1.3 void vec\_init ( int *length*, float *off*, float \* *a* )

Initialize vector with "random" values

## Parameters

<i>length</i>	- vector length
<i>off</i>	- auxiliary offset
<i>a</i>	- vector to initialize

Definition at line 42 of file main.cpp.

13.13.1.4 double vec\_sum ( int *length*, float \* *vec* )

Compute sum of final vector

## Returns

sum - final vector sum

Definition at line 55 of file main.cpp.

## 13.14 matvec/naive/main.cpp File Reference

```
#include <stdio>
#include <cmath>
#include "matvec.h"
#include "papi_cntr.h"
```

## Functions

- void [mat\\_init](#) (int row, int col, float off, float a[ ][COLS])
- void [vec\\_init](#) (int length, float off, float a[])
- double [vec\\_sum](#) (int length, float vec[])
- int [main](#) (int argc, char \*\*argv)

### 13.14.1 Function Documentation

#### 13.14.1.1 `int main ( int argc, char ** argv )`

Main function

Parameters

<i>argc</i>	
<i>argv</i>	

Definition at line 71 of file main.cpp.

#### 13.14.1.2 `void mat_init ( int row, int col, float off, float a ][COLS] )`

Initialize matrix with "random" values

Parameters

<i>row</i>	- matrix rows
<i>col</i>	- matrix col
<i>off</i>	- auxiliary offset
<i>a</i>	- matrix to initialize

Definition at line 26 of file main.cpp.

#### 13.14.1.3 `void vec_init ( int length, float off, float a [ ] )`

Initialize vector with "random" values

Parameters

<i>length</i>	- vector length
<i>off</i>	- auxiliary offset
<i>a</i>	- vector to initialize

Definition at line 42 of file main.cpp.

#### 13.14.1.4 `double vec_sum ( int length, float vec [ ] )`

Compute sum of final vector

Returns

sum - final vector sum

Definition at line 55 of file main.cpp.

## 13.15 matvec/omp-parallel/main.cpp File Reference

```
#include <stdio>
#include <cmath>
#include <omp.h>
#include "matvec.h"
#include "papi_cntr.h"
```



## Functions

- void [mat\\_init](#) (int row, int col, float off, float \*a)
- void [vec\\_init](#) (int length, float off, float \*a)
- double [vec\\_sum](#) (int length, float \*vec)
- int [main](#) (int argc, char \*\*argv)

### 13.15.1 Function Documentation

#### 13.15.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Parameters

<i>argc</i>	
<i>argv</i>	

Definition at line 76 of file main.cpp.

#### 13.15.1.2 void mat\_init ( int *row*, int *col*, float *off*, float \* *a* )

Initialize matrix with "random" values

Parameters

<i>row</i>	- matrix rows
<i>col</i>	- matrix col
<i>off</i>	- auxiliary offset
<i>a</i>	- matrix to initialize

Definition at line 27 of file main.cpp.

#### 13.15.1.3 void vec\_init ( int *length*, float *off*, float \* *a* )

Initialize vector with "random" values

Parameters

<i>length</i>	- vector length
<i>off</i>	- auxiliary offset
<i>a</i>	- vector to initialize

Definition at line 45 of file main.cpp.

#### 13.15.1.4 double vec\_sum ( int *length*, float \* *vec* )

Compute sum of final vector

Returns

sum - final vector sum

Definition at line 60 of file main.cpp.

## 13.16 matvec/omp-parallel-mic/main.cpp File Reference

```
#include <stdio>
#include <cmath>
#include <omp.h>
#include "matvec.h"
```

### Functions

- void [mat\\_init](#) (int row, int col, float off, float \*a)
- void [vec\\_init](#) (int length, float off, float \*a)
- double [vec\\_sum](#) (int length, float \*vec)
- int [main](#) (int argc, char \*\*argv)

### 13.16.1 Function Documentation

#### 13.16.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Parameters

<i>argc</i>	
<i>argv</i>	

Definition at line 81 of file main.cpp.

#### 13.16.1.2 void mat\_init ( int *row*, int *col*, float *off*, float \* *a* )

Initialize matrix with "random" values

Parameters

<i>row</i>	- matrix rows
<i>col</i>	- matrix col
<i>off</i>	- auxiliary offset
<i>a</i>	- matrix to initialize

Definition at line 26 of file main.cpp.

#### 13.16.1.3 void vec\_init ( int *length*, float *off*, float \* *a* )

Initialize vector with "random" values

Parameters

<i>length</i>	- vector length
<i>off</i>	- auxiliary offset
<i>a</i>	- vector to initialize

Definition at line 48 of file main.cpp.

#### 13.16.1.4 double vec\_sum ( int *length*, float \* *vec* )

Compute sum of final vector

**Returns**

sum - final vector sum

Definition at line 65 of file main.cpp.

**13.17 matvec/vec-padding/main.cpp File Reference**

```
#include <stdio>
#include <cmath>
#include "matvec.h"
#include "papi_cntr.h"
```

**Functions**

- void [mat\\_init](#) (int row, int col, float off, float a[ ][[COL\\_PAD](#)])
- void [vec\\_init](#) (int length, float off, float a[ ])
- double [vec\\_sum](#) (int length, float vec[ ])
- int [main](#) (int argc, char \*\*argv)

**Variables**

- const unsigned [COL\\_PAD](#) = COLS + PADDING

**13.17.1 Function Documentation****13.17.1.1 int main ( int *argc*, char \*\* *argv* )**

Main function

**Parameters**

<i>argc</i>	
<i>argv</i>	

Definition at line 105 of file main.cpp.

**13.17.1.2 void mat\_init ( int *row*, int *col*, float *off*, float *a*[ ][[COL\\_PAD](#)] )**

Initialize matrix with "random" values

**Parameters**

<i>row</i>	- matrix rows
<i>col</i>	- matrix col
<i>off</i>	- auxiliary offset
<i>a</i>	- matrix to initialize

Definition at line 28 of file main.cpp.

**13.17.1.3 void vec\_init ( int *length*, float *off*, float *a*[ ] )**

Initialize vector with "random" values

**Parameters**

<i>length</i>	- vector length
<i>off</i>	- auxiliary offset
<i>a</i>	- vector to initialize

Definition at line 62 of file main.cpp.

13.17.1.4 `double vec_sum ( int length, float vec[ ] )`

Compute sum of final vector

**Returns**

sum - final vector sum

Definition at line 87 of file main.cpp.

**13.17.2 Variable Documentation**

13.17.2.1 `const unsigned COL_PAD = COLS + PADDING`

Definition at line 18 of file main.cpp.

**13.18 nbody/cache-block-mic/main.cpp File Reference**

```
#include <stdio>
#include <cmath>
#include <omp.h>
#include "nbody.h"
```

**Functions**

- int [main](#) (int argc, char \*\*argv)

**13.18.1 Function Documentation**

13.18.1.1 `int main ( int argc, char ** argv )`

Main function

**Parameters**

<i>argc</i>	
<i>argv</i>	

Definition at line 24 of file main.cpp.

**13.19 nbody/naive/main.cpp File Reference**

```
#include <stdio>
```

```
#include <cmath>
#include "nbody.h"
#include "papi_cntr.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.19.1 Function Documentation

13.19.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Parameters

<i>argc</i>	
<i>argv</i>	

Definition at line 24 of file main.cpp.

## 13.20 nbody/no-jump-auto-opt/main.cpp File Reference

```
#include <stdio>
#include <cmath>
#include <omp.h>
#include "nbody.h"
#include "papi_cntr.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.20.1 Function Documentation

13.20.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Parameters

<i>argc</i>	
<i>argv</i>	

Definition at line 25 of file main.cpp.

## 13.21 nbody/offload/main.cpp File Reference

```
#include <stdio>
```

```
#include <cmath>
#include <omp.h>
#include <offload.h>
#include "nbody.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.21.1 Function Documentation

13.21.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Parameters

<i>argc</i>	
<i>argv</i>	

Definition at line 27 of file main.cpp.

## 13.22 nbody/omp-parallel/main.cpp File Reference

```
#include <stdio>
#include <cmath>
#include <omp.h>
#include "nbody.h"
#include "papi_cntr.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.22.1 Function Documentation

13.22.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Parameters

<i>argc</i>	
<i>argv</i>	

Definition at line 25 of file main.cpp.

## 13.23 nbody/omp-parallel-mic/main.cpp File Reference

```
#include <stdio>
```

```
#include <cmath>
#include <omp.h>
#include "nbody.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.23.1 Function Documentation

13.23.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Parameters

<i>argc</i>	
<i>argv</i>	

Definition at line 24 of file main.cpp.

## 13.24 matvec/dynamic-aligned/matvec.cpp File Reference

```
#include "matvec.h"
```

## Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float \*a, float \*b, float \*c)

### 13.24.1 Function Documentation

13.24.1.1 void mat\_vec\_mul ( int *rows*, int *cols*, float \* *a*, float \* *b*, float \* *c* )

Compute matrix and vector multiplication

Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector
<i>c</i>	- final vector

Definition at line 23 of file matvec.cpp.

## 13.25 matvec/naive/matvec.cpp File Reference

```
#include "matvec.h"
```

## Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float a[][cols], float b[cols], float c[rows])

### 13.25.1 Function Documentation

13.25.1.1 void [mat\\_vec\\_mul](#) ( int *rows*, int *cols*, float *a*[][cols], float *b*[cols], float *c*[rows] )

Compute matrix and vector multiplication

#### Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector
<i>c</i>	- final vector

Definition at line 23 of file matvec.cpp.

## 13.26 matvec/omp-parallel/matvec.cpp File Reference

```
#include "matvec.h"
#include <omp.h>
```

## Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float \*a, float \*b, float \*c)

### 13.26.1 Function Documentation

13.26.1.1 void [mat\\_vec\\_mul](#) ( int *rows*, int *cols*, float \* *a*, float \* *b*, float \* *c* )

Compute matrix and vector multiplication

#### Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector
<i>c</i>	- final vector

Definition at line 24 of file matvec.cpp.

## 13.27 matvec/omp-parallel-mic/matvec.cpp File Reference

```
#include "matvec.h"
#include <omp.h>
```



## Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float \*a, float \*b, float \*c)

### 13.27.1 Function Documentation

13.27.1.1 void [mat\\_vec\\_mul](#) ( int *rows*, int *cols*, float \* *a*, float \* *b*, float \* *c* )

Compute matrix and vector multiplication

#### Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector
<i>c</i>	- final vector

Definition at line 24 of file matvec.cpp.

## 13.28 matvec/vec-padding/matvec.cpp File Reference

```
#include "matvec.h"
```

## Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float a[][cols], float b[cols], float c[rows])

### 13.28.1 Function Documentation

13.28.1.1 void [mat\\_vec\\_mul](#) ( int *rows*, int *cols*, float *a*[][cols], float *b*[cols], float *c*[rows] )

Compute matrix and vector multiplication

#### Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector
<i>c</i>	- final vector

Definition at line 23 of file matvec.cpp.

## 13.29 matvec/dynamic-aligned/matvec.h File Reference

## Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float \*a, float \*b, float \*c)

### 13.29.1 Function Documentation

13.29.1.1 `void mat_vec_mul ( int rows, int cols, float * a, float * b, float * c )`

Compute matrix and vector multiplication

## Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector
<i>c</i>	- final vector

Definition at line 23 of file matvec.cpp.

## 13.30 matvec/naive/matvec.h File Reference

## Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float a[][cols], float b[cols], float c[rows])

### 13.30.1 Function Documentation

13.30.1.1 void [mat\\_vec\\_mul](#) ( int *rows*, int *cols*, float *a*[][cols], float *b*[cols], float *c*[rows] )

Compute matrix and vector multiplication

## Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector
<i>c</i>	- final vector

Definition at line 23 of file matvec.cpp.

## 13.31 matvec/omp-parallel/matvec.h File Reference

## Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float \*a, float \*b, float \*c)

### 13.31.1 Function Documentation

13.31.1.1 void [mat\\_vec\\_mul](#) ( int *rows*, int *cols*, float \* *a*, float \* *b*, float \* *c* )

Compute matrix and vector multiplication

## Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector

<i>c</i>	- final vector
----------	----------------

Definition at line 23 of file matvec.cpp.

## 13.32 matvec/omp-parallel-mic/matvec.h File Reference

### Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float \*a, float \*b, float \*c)

#### 13.32.1 Function Documentation

13.32.1.1 void [mat\\_vec\\_mul](#) ( int *rows*, int *cols*, float \* *a*, float \* *b*, float \* *c* )

Compute matrix and vector multiplication

##### Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector
<i>c</i>	- final vector

Definition at line 23 of file matvec.cpp.

## 13.33 matvec/vec-padding/matvec.h File Reference

### Functions

- void [mat\\_vec\\_mul](#) (int rows, int cols, float a[][cols], float b[cols], float c[rows])

#### 13.33.1 Function Documentation

13.33.1.1 void [mat\\_vec\\_mul](#) ( int *rows*, int *cols*, float *a*[][cols], float *b*[cols], float *c*[rows] )

Compute matrix and vector multiplication

##### Parameters

<i>rows</i>	- matrix rows
<i>cols</i>	- matrix cols
<i>a</i>	- matrix
<i>b</i>	- vector
<i>c</i>	- final vector

Definition at line 23 of file matvec.cpp.

## 13.34 nbbody/cache-block-mic/gen.cpp File Reference

```
#include <cstdlib>
```

```
#include <stdio>
#include <float>
#include <time>
```

## Functions

- float [randf](#) ()
- int [main](#) (int argc, char \*\*argv)

### 13.34.1 Function Documentation

#### 13.34.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Definition at line 36 of file gen.cpp.

#### 13.34.1.2 float randf ( )

Generate "random" float

Definition at line 23 of file gen.cpp.

## 13.35 nbody/naive/gen.cpp File Reference

```
#include <cstdlib>
#include <stdio>
#include <float>
#include <time>
```

## Functions

- float [randf](#) ()
- int [main](#) (int argc, char \*\*argv)

### 13.35.1 Function Documentation

#### 13.35.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Definition at line 36 of file gen.cpp.

#### 13.35.1.2 float randf ( )

Generate "random" float

Definition at line 23 of file gen.cpp.

## 13.36 nbody/no-jump-auto-opt/gen.cpp File Reference

```
#include <cstdlib>
#include <stdio>
#include <float>
#include <ctime>
```

### Functions

- float [randf](#) ()
- int [main](#) (int argc, char \*\*argv)

#### 13.36.1 Function Documentation

##### 13.36.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Definition at line 36 of file gen.cpp.

##### 13.36.1.2 float randf ( )

Generate "random" float

Definition at line 23 of file gen.cpp.

## 13.37 nbody/offload/gen.cpp File Reference

```
#include <cstdlib>
#include <stdio>
#include <float>
#include <ctime>
```

### Functions

- float [randf](#) ()
- int [main](#) (int argc, char \*\*argv)

#### 13.37.1 Function Documentation

##### 13.37.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Definition at line 36 of file gen.cpp.

##### 13.37.1.2 float randf ( )

Generate "random" float

Definition at line 23 of file gen.cpp.

## 13.38 nbody/omp-parallel/gen.cpp File Reference

```
#include <cstdlib>
#include <stdio>
#include <float>
#include <ctime>
```

### Functions

- float [randf](#) ()
- int [main](#) (int argc, char \*\*argv)

#### 13.38.1 Function Documentation

##### 13.38.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Definition at line 36 of file gen.cpp.

##### 13.38.1.2 float randf ( )

Generate "random" float

Definition at line 23 of file gen.cpp.

## 13.39 nbody/omp-parallel-mic/gen.cpp File Reference

```
#include <cstdlib>
#include <stdio>
#include <float>
#include <ctime>
```

### Functions

- float [randf](#) ()
- int [main](#) (int argc, char \*\*argv)

#### 13.39.1 Function Documentation

##### 13.39.1.1 int main ( int *argc*, char \*\* *argv* )

Main function

Definition at line 36 of file gen.cpp.

##### 13.39.1.2 float randf ( )

Generate "random" float

Definition at line 23 of file gen.cpp.

## 13.40 nbody/cache-block-mic/nbody.cpp File Reference

```
#include <cfloat>
#include <cmath>
#include <omp.h>
#include <algorithm>
#include "nbody.h"
```

### Functions

- [t\\_particles\\_DA](#) \* [particles\\_alloc](#) (size\_t size)
- void [particles\\_free](#) ([t\\_particles\\_DA](#) \*p)
- void [particles\\_init](#) ([t\\_particles\\_DA](#) p)
- void [particles\\_simulate](#) ([t\\_particles\\_DA](#) p)
- void [particles\\_read](#) (FILE \*fp, [t\\_particles\\_DA](#) p)
- void [particles\\_write](#) (FILE \*fp, [t\\_particles\\_DA](#) p)

### Variables

- const float [SML\\_FLT](#) = 1e-9f
- const int [BLOCK](#) = 512

### 13.40.1 Function Documentation

#### 13.40.1.1 [t\\_particles\\_DA](#)\* [particles\\_alloc](#) ( size\_t size )

Allocate memory for particles

##### Parameters

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

##### Returns

pointer to structure

Definition at line 29 of file nbody.cpp.

#### 13.40.1.2 void [particles\\_free](#) ( [t\\_particles\\_DA](#) \* *p* )

Free memory

##### Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 166 of file nbody.cpp.

#### 13.40.1.3 void [particles\\_init](#) ( [t\\_particles\\_DA](#) *p* )

Initialize structure of particles NUMA First Touch Policy



## Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 198 of file nbody.cpp.

13.40.1.4 void particles\_read ( FILE \* *fp*, t\_particles\_DA *p* )

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 334 of file nbody.cpp.

13.40.1.5 void particles\_simulate ( t\_particles\_DA *p* )

Simulate particle system

## Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 226 of file nbody.cpp.

13.40.1.6 void particles\_write ( FILE \* *fp*, t\_particles\_DA *p* )

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 355 of file nbody.cpp.

## 13.40.2 Variable Documentation

## 13.40.2.1 const int BLOCK = 512

Definition at line 20 of file nbody.cpp.

## 13.40.2.2 const float SML\_FLT = 1e-9f

Definition at line 19 of file nbody.cpp.

## 13.41 nbody/naive/nbody.cpp File Reference

```
#include <cfloat>
#include <cmath>
#include "nbody.h"
```

## Functions

- void [particles\\_simulate](#) ([t\\_particles](#) p)
- void [particles\\_read](#) (FILE \*fp, [t\\_particles](#) p)
- void [particles\\_write](#) (FILE \*fp, [t\\_particles](#) p)

### 13.41.1 Function Documentation

#### 13.41.1.1 void [particles\\_read](#) ( FILE \* *fp*, [t\\_particles](#) *p* )

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

##### Parameters

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 92 of file nbody.cpp.

#### 13.41.1.2 void [particles\\_simulate](#) ( [t\\_particles](#) *p* )

Simulate particle system

##### Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 21 of file nbody.cpp.

#### 13.41.1.3 void [particles\\_write](#) ( FILE \* *fp*, [t\\_particles](#) *p* )

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

##### Parameters

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 113 of file nbody.cpp.

## 13.42 nbody/no-jump-auto-opt/nbody.cpp File Reference

```
#include <cfloat>
#include <cmath>
#include <omp.h>
#include <algorithm>
#include "nbody.h"
```

## Macros

- #define [SML\\_FLT](#) 1e-9f

## Functions

- [t\\_particles\\_DA](#) \* [particles\\_alloc](#) (size\_t size)

- void [particles\\_free](#) (t\_particles\_DA \*p)
- void [particles\\_init](#) (t\_particles\_DA p)
- void [particles\\_simulate](#) (t\_particles\_DA p)
- void [particles\\_read](#) (FILE \*fp, t\_particles\_DA p)
- void [particles\\_write](#) (FILE \*fp, t\_particles\_DA p)

### 13.42.1 Macro Definition Documentation

#### 13.42.1.1 #define SML\_FLT 1e-9f

Definition at line 19 of file nbody.cpp.

### 13.42.2 Function Documentation

#### 13.42.2.1 t\_particles\_DA\* particles\_alloc ( size\_t size )

Allocate memory for particles

Parameters

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

Returns

pointer to structure

Definition at line 28 of file nbody.cpp.

#### 13.42.2.2 void particles\_free ( t\_particles\_DA \* p )

Free memory

Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 101 of file nbody.cpp.

#### 13.42.2.3 void particles\_init ( t\_particles\_DA p )

Initialize structure of particles

Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 126 of file nbody.cpp.

#### 13.42.2.4 void particles\_read ( FILE \* fp, t\_particles\_DA p )

Read particles from file Format: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

Parameters

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 230 of file nbody.cpp.

#### 13.42.2.5 void particles\_simulate ( t\_particles\_DA p )

Simulate particle system

Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 147 of file nbody.cpp.

#### 13.42.2.6 void particles\_write ( FILE \* fp, t\_particles\_DA p )

Write particles to file Format: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

Parameters

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 251 of file nbody.cpp.

### 13.43 nbody/offload/nbody.cpp File Reference

```
#include <cfloat>
#include <cmath>
#include <omp.h>
#include <algorithm>
#include "nbody.h"
```

#### Functions

- [\\_\\_attribute\\_\\_](#) ((target(mic))) const float [SML\\_FLT](#)
- void [particles\\_alloc](#) (t\_particles\_DA \*p, size\_t size)
- void [particles\\_free](#) (t\_particles\_DA \*p)
- void [particles\\_read](#) (FILE \*fp, t\_particles\_DA p)
- void [particles\\_write](#) (FILE \*fp, t\_particles\_DA p)

#### 13.43.1 Function Documentation

##### 13.43.1.1 \_\_attribute\_\_ ( (target(mic)) ) const

Initialize structure of particles NUMA First Touch Policy

Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Simulate particle system

## Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 179 of file nbody.cpp.

13.43.1.2 void particles\_alloc ( t\_particles\_DA \* *p*, size\_t *size* )

Allocate memory for particles

## Parameters

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

## Returns

pointer to structure

Definition at line 28 of file nbody.cpp.

13.43.1.3 void particles\_free ( t\_particles\_DA \* *p* )

Free memory

## Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 154 of file nbody.cpp.

13.43.1.4 void particles\_read ( FILE \* *fp*, t\_particles\_DA *p* )

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 308 of file nbody.cpp.

13.43.1.5 void particles\_write ( FILE \* *fp*, t\_particles\_DA *p* )

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 329 of file nbody.cpp.

## 13.44 nbody/omp-parallel/nbody.cpp File Reference

```
#include <cfloat>
#include <cmath>
#include <omp.h>
#include <algorithm>
#include "nbody.h"
```

## Functions

- `t_particles_DA * particles_alloc (size_t size)`
- `void particles_free (t_particles_DA *p)`
- `void particles_init (t_particles_DA p)`
- `void particles_simulate (t_particles_DA p)`
- `void particles_read (FILE *fp, t_particles_DA p)`
- `void particles_write (FILE *fp, t_particles_DA p)`

## Variables

- `const float SML_FLT = 1e-9f`

### 13.44.1 Function Documentation

#### 13.44.1.1 `t_particles_DA* particles_alloc ( size_t size )`

Allocate memory for particles

##### Parameters

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

##### Returns

pointer to structure

Definition at line 28 of file nbody.cpp.

#### 13.44.1.2 `void particles_free ( t_particles_DA * p )`

Free memory

##### Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 165 of file nbody.cpp.

#### 13.44.1.3 `void particles_init ( t_particles_DA p )`

Initialize structure of particles NUMA First Touch Policy

##### Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 197 of file nbody.cpp.

#### 13.44.1.4 `void particles_read ( FILE * fp, t_particles_DA p )`

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

**Parameters**

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 329 of file nbody.cpp.

#### 13.44.1.5 void particles\_simulate ( t\_particles\_DA p )

Simulate particle system

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 225 of file nbody.cpp.

#### 13.44.1.6 void particles\_write ( FILE \* fp, t\_particles\_DA p )

Write particles to file Format: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

**Parameters**

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 350 of file nbody.cpp.

### 13.44.2 Variable Documentation

#### 13.44.2.1 const float SML\_FLT = 1e-9f

Definition at line 19 of file nbody.cpp.

## 13.45 nbody/omp-parallel-mic/nbody.cpp File Reference

```
#include <cfloat>
#include <cmath>
#include <omp.h>
#include <algorithm>
#include "nbody.h"
```

**Functions**

- [t\\_particles\\_DA \\* particles\\_alloc](#) (size\_t size)
- void [particles\\_free](#) (t\_particles\_DA \*p)
- void [particles\\_init](#) (t\_particles\_DA p)
- void [particles\\_simulate](#) (t\_particles\_DA p)
- void [particles\\_read](#) (FILE \*fp, t\_particles\_DA p)
- void [particles\\_write](#) (FILE \*fp, t\_particles\_DA p)

**Variables**

- const float [SML\\_FLT](#) = 1e-9f

### 13.45.1 Function Documentation

#### 13.45.1.1 `t_particles_DA* particles_alloc ( size_t size )`

Allocate memory for particles

##### Parameters

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

##### Returns

pointer to structure

Definition at line 28 of file nbody.cpp.

#### 13.45.1.2 `void particles_free ( t_particles_DA * p )`

Free memory

##### Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 165 of file nbody.cpp.

#### 13.45.1.3 `void particles_init ( t_particles_DA p )`

Initialize structure of particles NUMA First Touch Policy

##### Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 197 of file nbody.cpp.

#### 13.45.1.4 `void particles_read ( FILE * fp, t_particles_DA p )`

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

##### Parameters

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 328 of file nbody.cpp.

#### 13.45.1.5 `void particles_simulate ( t_particles_DA p )`

Simulate particle system

##### Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 225 of file nbody.cpp.

#### 13.45.1.6 `void particles_write ( FILE * fp, t_particles_DA p )`

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight



## Parameters

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 349 of file nbody.cpp.

### 13.45.2 Variable Documentation

#### 13.45.2.1 `const float SML_FLT = 1e-9f`

Definition at line 19 of file nbody.cpp.

## 13.46 nbody/cache-block-mic/nbody.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
```

### Classes

- struct [t\\_particles\\_DA](#)

### Macros

- `#define G 6.67384e-11f`

### Functions

- [t\\_particles\\_DA \\* particles\\_alloc](#) (size\_t size)
- void [particles\\_free](#) (t\_particles\_DA \*p)
- void [particles\\_init](#) (t\_particles\_DA p)
- void [particles\\_simulate](#) (t\_particles\_DA p)
- void [particles\\_read](#) (FILE \*fp, t\_particles\_DA p)
- void [particles\\_write](#) (FILE \*fp, t\_particles\_DA p)

### 13.46.1 Macro Definition Documentation

#### 13.46.1.1 `#define G 6.67384e-11f`

Definition at line 20 of file nbody.h.

### 13.46.2 Function Documentation

#### 13.46.2.1 `t_particles_DA* particles_alloc ( size_t size )`

Allocate memory for particles

**Parameters**

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

**Returns**

pointer to structure

Definition at line 29 of file nbody.cpp.

**13.46.2.2 void particles\_free ( t\_particles\_DA \* p )**

Free memory

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 166 of file nbody.cpp.

**13.46.2.3 void particles\_init ( t\_particles\_DA p )**

Initialize structure of particles

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Initialize structure of particles NUMA First Touch Policy

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 198 of file nbody.cpp.

**13.46.2.4 void particles\_read ( FILE \* fp, t\_particles\_DA p )**

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

**Parameters**

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 334 of file nbody.cpp.

**13.46.2.5 void particles\_simulate ( t\_particles\_DA p )**

Simulate particle system

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 226 of file nbody.cpp.

**13.46.2.6 void particles\_write ( FILE \* fp, t\_particles\_DA p )**

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 355 of file nbody.cpp.

## 13.47 nbody/naive/nbody.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
```

## Classes

- struct [t\\_particle](#)

## Macros

- #define [G](#) 6.67384e-11f

## Typedefs

- typedef [t\\_particle](#) [t\\_particles](#)[N]

## Functions

- void [particles\\_simulate](#) ([t\\_particles](#) p)
- void [particles\\_read](#) (FILE \*fp, [t\\_particles](#) p)
- void [particles\\_write](#) (FILE \*fp, [t\\_particles](#) p)

### 13.47.1 Macro Definition Documentation

#### 13.47.1.1 #define G 6.67384e-11f

Definition at line 19 of file nbody.h.

### 13.47.2 Typedef Documentation

#### 13.47.2.1 typedef t\_particle t\_particles[N]

Definition at line 34 of file nbody.h.

### 13.47.3 Function Documentation

#### 13.47.3.1 void particles\_read ( FILE \* fp, t\_particles p )

Read particles from file Format: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

**Parameters**

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 92 of file nbody.cpp.

**13.47.3.2 void particles\_simulate ( t\_particles p )**

Simulate particle system

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 21 of file nbody.cpp.

**13.47.3.3 void particles\_write ( FILE \*fp, t\_particles p )**

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

**Parameters**

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 113 of file nbody.cpp.

**13.48 nbody/no-jump-auto-opt/nbody.h File Reference**

```
#include <stdlib>
#include <stdio>
```

**Classes**

- struct [t\\_particles\\_DA](#)

**Macros**

- #define [G](#) 6.67384e-11f

**Functions**

- [t\\_particles\\_DA \\* particles\\_alloc](#) (size\_t size)
- void [particles\\_free](#) (t\_particles\_DA \*p)
- void [particles\\_init](#) (t\_particles\_DA p)
- void [particles\\_simulate](#) (t\_particles\_DA p)
- void [particles\\_read](#) (FILE \*fp, t\_particles\_DA p)
- void [particles\\_write](#) (FILE \*fp, t\_particles\_DA p)

### 13.48.1 Macro Definition Documentation

#### 13.48.1.1 `#define G 6.67384e-11f`

Definition at line 20 of file nbody.h.

### 13.48.2 Function Documentation

#### 13.48.2.1 `t_particles_DA* particles_alloc ( size_t size )`

Allocate memory for particles

Parameters

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

Returns

pointer to structure

Definition at line 29 of file nbody.cpp.

#### 13.48.2.2 `void particles_free ( t_particles_DA * p )`

Free memory

Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 166 of file nbody.cpp.

#### 13.48.2.3 `void particles_init ( t_particles_DA p )`

Initialize structure of particles

Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Initialize structure of particles NUMA First Touch Policy

Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 198 of file nbody.cpp.

#### 13.48.2.4 `void particles_read ( FILE * fp, t_particles_DA p )`

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

Parameters

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 334 of file nbody.cpp.

13.48.2.5 void particles\_simulate ( t\_particles\_DA p )

Simulate particle system

## Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 226 of file nbody.cpp.

13.48.2.6 void particles\_write ( FILE \* *fp*, t\_particles\_DA *p* )

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 355 of file nbody.cpp.

## 13.49 nbody/offload/nbody.h File Reference

```
#include <stdlib>
#include <stdio>
```

## Classes

- struct [t\\_particles\\_DA](#)

## Macros

- #define [G](#) 6.67384e-11f

## Functions

- void [particles\\_alloc](#) (t\_particles\_DA \*p, size\_t size)
- void [particles\\_free](#) (t\_particles\_DA \*p)
- [\\_\\_attribute\\_\\_](#) ((target(mic))) void [particles\\_init](#)(t\_particles\_DA p)
- void [particles\\_read](#) (FILE \*fp, t\_particles\_DA p)
- void [particles\\_write](#) (FILE \*fp, t\_particles\_DA p)

## 13.49.1 Macro Definition Documentation

## 13.49.1.1 #define G 6.67384e-11f

Definition at line 20 of file nbody.h.

## 13.49.2 Function Documentation

13.49.2.1 [\\_\\_attribute\\_\\_](#) ( (target(mic)) )

Initialize structure of particles

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Simulate particle system

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Initialize structure of particles NUMA First Touch Policy

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Simulate particle system

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 179 of file nbody.cpp.

**13.49.2.2 void particles\_alloc ( t\_particles\_DA \* p, size\_t size )**

Allocate memory for particles

**Parameters**

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

**Returns**

pointer to structure

Definition at line 28 of file nbody.cpp.

**13.49.2.3 void particles\_free ( t\_particles\_DA \* p )**

Free memory

**Parameters**

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 166 of file nbody.cpp.

**13.49.2.4 void particles\_read ( FILE \* fp, t\_particles\_DA p )**

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

**Parameters**

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 334 of file nbody.cpp.

**13.49.2.5 void particles\_write ( FILE \* fp, t\_particles\_DA p )**

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight



## Parameters

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 355 of file nbody.cpp.

## 13.50 nbody/omp-parallel/nbody.h File Reference

```
#include <cstdlib>
#include <cstdio>
```

## Classes

- struct [t\\_particles\\_DA](#)

## Macros

- #define [G](#) 6.67384e-11f

## Functions

- [t\\_particles\\_DA \\* particles\\_alloc](#) (size\_t size)
- void [particles\\_free](#) (t\_particles\_DA \*p)
- void [particles\\_init](#) (t\_particles\_DA p)
- void [particles\\_simulate](#) (t\_particles\_DA p)
- void [particles\\_read](#) (FILE \*fp, t\_particles\_DA p)
- void [particles\\_write](#) (FILE \*fp, t\_particles\_DA p)

### 13.50.1 Macro Definition Documentation

#### 13.50.1.1 #define G 6.67384e-11f

Definition at line 20 of file nbody.h.

### 13.50.2 Function Documentation

#### 13.50.2.1 t\_particles\_DA\* particles\_alloc ( size\_t size )

Allocate memory for particles

## Parameters

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

## Returns

pointer to structure

Definition at line 29 of file nbody.cpp.

13.50.2.2 void particles\_free ( t\_particles\_DA \* p )

Free memory

## Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 166 of file nbody.cpp.

13.50.2.3 void particles\_init ( t\_particles\_DA *p* )

Initialize structure of particles

## Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Initialize structure of particles NUMA First Touch Policy

## Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 198 of file nbody.cpp.

13.50.2.4 void particles\_read ( FILE \* *fp*, t\_particles\_DA *p* )

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

<i>fp</i>	- input file descriptor
<i>p</i>	- structure of particles

Definition at line 334 of file nbody.cpp.

13.50.2.5 void particles\_simulate ( t\_particles\_DA *p* )

Simulate particle system

## Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 226 of file nbody.cpp.

13.50.2.6 void particles\_write ( FILE \* *fp*, t\_particles\_DA *p* )

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

<i>fp</i>	- output file descriptor
<i>p</i>	- structure of particles

Definition at line 355 of file nbody.cpp.

## 13.51 nbody/omp-parallel-mic/nbody.h File Reference

```
#include <cstdlib>
#include <stdio>
```

## Classes

- struct [t\\_particles\\_DA](#)

## Macros

- #define [G](#) 6.67384e-11f

## Functions

- [t\\_particles\\_DA](#) \* [particles\\_alloc](#) (size\_t size)
- void [particles\\_free](#) ([t\\_particles\\_DA](#) \*p)
- void [particles\\_init](#) ([t\\_particles\\_DA](#) p)
- void [particles\\_simulate](#) ([t\\_particles\\_DA](#) p)
- void [particles\\_read](#) (FILE \*fp, [t\\_particles\\_DA](#) p)
- void [particles\\_write](#) (FILE \*fp, [t\\_particles\\_DA](#) p)

### 13.51.1 Macro Definition Documentation

#### 13.51.1.1 #define [G](#) 6.67384e-11f

Definition at line 20 of file nbody.h.

### 13.51.2 Function Documentation

#### 13.51.2.1 [t\\_particles\\_DA](#)\* [particles\\_alloc](#) ( size\_t size )

Allocate memory for particles

##### Parameters

<i>size</i>	- number of particles to alloc
-------------	--------------------------------

##### Returns

pointer to structure

Definition at line 29 of file nbody.cpp.

#### 13.51.2.2 void [particles\\_free](#) ( [t\\_particles\\_DA](#) \* p )

Free memory

##### Parameters

<i>p</i>	- structure of particles
----------	--------------------------

Definition at line 166 of file nbody.cpp.

#### 13.51.2.3 void [particles\\_init](#) ( [t\\_particles\\_DA](#) p )

Initialize structure of particles

## Parameters

$p$	- structure of particles
-----	--------------------------

Initialize structure of particles NUMA First Touch Policy

## Parameters

$p$	- structure of particles
-----	--------------------------

Definition at line 198 of file nbody.cpp.

13.51.2.4 void particles\_read ( FILE \*  $fp$ , t\_particles\_DA  $p$  )

Read particles from file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

$fp$	- input file descriptor
$p$	- structure of particles

Definition at line 334 of file nbody.cpp.

13.51.2.5 void particles\_simulate ( t\_particles\_DA  $p$  )

Simulate particle system

## Parameters

$p$	- structure of particles
-----	--------------------------

Definition at line 226 of file nbody.cpp.

13.51.2.6 void particles\_write ( FILE \*  $fp$ , t\_particles\_DA  $p$  )

Write particles to file Fomat: pos\_x pos\_y pos\_z vel\_x vel\_y vel\_z weight

## Parameters

$fp$	- output file descriptor
$p$	- structure of particles

Definition at line 355 of file nbody.cpp.

## 13.52 python-mkl/numpy\_dgemm.py File Reference

## Namespaces

- [numpy\\_dgemm](#)

## Variables

- tuple [numpy\\_dgemm.K](#) = int(sys.argv[1])
- [numpy\\_dgemm.M](#) = K
- [numpy\\_dgemm.N](#) = K
- int [numpy\\_dgemm.ITER](#) = 1
- tuple [numpy\\_dgemm.t\\_init](#) = time.time()
- tuple [numpy\\_dgemm.a1](#) = np.array(np.random.random((M, K)), dtype=np.double, order='C', copy=False)
- tuple [numpy\\_dgemm.a2](#) = np.array(np.random.random((K, N)), dtype=np.double, order='C', copy=False)

- tuple `numpy_dgemm.m1` = `np.matrix(a1, dtype=np.double, copy=False)`
- tuple `numpy_dgemm.m2` = `np.matrix(a2, dtype=np.double, copy=False)`
- tuple `numpy_dgemm.t_start` = `time.time()`
- tuple `numpy_dgemm.mf2` = `np.dot(m1, m2)`
- tuple `numpy_dgemm.t_end` = `time.time()`

## 13.53 python-mkl/scipy\_dgemm.py File Reference

### Namespaces

- `scipy_dgemm`

### Variables

- tuple `scipy_dgemm.K` = `int(sys.argv[1])`
- `scipy_dgemm.M` = `K`
- `scipy_dgemm.N` = `K`
- int `scipy_dgemm.ITER` = `1`
- tuple `scipy_dgemm.a1` = `np.array(np.random.random((M, K)), dtype=np.double, order='C', copy=False)`
- tuple `scipy_dgemm.a2` = `np.array(np.random.random((K, N)), dtype=np.double, order='C', copy=False)`
- tuple `scipy_dgemm.m1` = `np.matrix(a1, dtype=np.double, copy=False)`
- tuple `scipy_dgemm.m2` = `np.matrix(a2, dtype=np.double, copy=False)`
- tuple `scipy_dgemm.t_start` = `time.time()`
- tuple `scipy_dgemm.mf2` = `sp.dgemm(alpha=1.0, a=m1, b=m2)`
- tuple `scipy_dgemm.t_end` = `time.time()`

# Index

- `__attribute__`
    - `offload/nbody.cpp`, [56](#)
    - `offload/nbody.h`, [67](#)
- a1
  - `numpy_dgemm`, [23](#)
  - `scipy_dgemm`, [25](#)
- a2
  - `numpy_dgemm`, [23](#)
  - `scipy_dgemm`, [25](#)
- ALPHA
  - `cpu/c_dgemm.c`, [33](#)
  - `mic/c_dgemm.c`, [34](#)
- ax
  - `t_particles_DA`, [28](#)
- ay
  - `t_particles_DA`, [28](#)
- az
  - `t_particles_DA`, [28](#)
- BETA
  - `cpu/c_dgemm.c`, [33](#)
  - `mic/c_dgemm.c`, [34](#)
- BLOCK
  - `cache-block-mic/nbody.cpp`, [53](#)
- COL\_PAD
  - `matvec/vec-padding/main.cpp`, [40](#)
- `cache-block-mic/gen.cpp`
  - `main`, [49](#)
  - `randf`, [49](#)
- `cache-block-mic/nbody.cpp`
  - BLOCK, [53](#)
  - `particles_alloc`, [52](#)
  - `particles_free`, [52](#)
  - `particles_init`, [52](#)
  - `particles_read`, [53](#)
  - `particles_simulate`, [53](#)
  - `particles_write`, [53](#)
  - SML\_FLT, [53](#)
- `cache-block-mic/nbody.h`
  - G, [61](#)
  - `particles_alloc`, [61](#)
  - `particles_free`, [62](#)
  - `particles_init`, [62](#)
  - `particles_read`, [62](#)
  - `particles_simulate`, [62](#)
  - `particles_write`, [62](#)
- `cpu/c_dgemm.c`
  - ALPHA, [33](#)
  - BETA, [33](#)
  - `main`, [33](#)
- `cpu/fftw-3d-omp.c`
  - `main`, [32](#)
- `cross-compilation/README.md`, [31](#)
- `dynamic-aligned/matvec.cpp`
  - `mat_vec_mul`, [43](#)
- `dynamic-aligned/matvec.h`
  - `mat_vec_mul`, [46](#)
- `fft-mkl/README.md`, [31](#)
- `fft-mkl/cpu/fftw-3d-omp.c`, [31](#)
- `fft-mkl/mic/fftw-3d-omp-bpage.c`, [32](#)
- `fft-mkl/mic/fftw-3d-omp.c`, [32](#)
- `fftw-3d-omp-bpage.c`
  - `main`, [33](#)
- fx
  - `t_particles_DA`, [29](#)
- fy
  - `t_particles_DA`, [29](#)
- fz
  - `t_particles_DA`, [29](#)
- G
  - `cache-block-mic/nbody.h`, [61](#)
  - `naive/nbody.h`, [63](#)
  - `no-jump-auto-opt/nbody.h`, [65](#)
  - `offload/nbody.h`, [67](#)
  - `omp-parallel-mic/nbody.h`, [72](#)
  - `omp-parallel/nbody.h`, [69](#)
- ITER
  - `numpy_dgemm`, [23](#)
  - `scipy_dgemm`, [25](#)
- K
  - `numpy_dgemm`, [23](#)
  - `scipy_dgemm`, [25](#)
- M
  - `numpy_dgemm`, [23](#)
  - `scipy_dgemm`, [25](#)
- m1
  - `numpy_dgemm`, [24](#)
  - `scipy_dgemm`, [25](#)
- m2
  - `numpy_dgemm`, [24](#)
  - `scipy_dgemm`, [25](#)
- main
  - `cache-block-mic/gen.cpp`, [49](#)

- cpu/c\_dgemm.c, 33
- cpu/fftw-3d-omp.c, 32
- fftw-3d-omp-bpage.c, 33
- matvec/dynamic-aligned/main.cpp, 34
- matvec/naive/main.cpp, 36
- matvec/omp-parallel-mic/main.cpp, 38
- matvec/omp-parallel/main.cpp, 37
- matvec/vec-padding/main.cpp, 39
- mic/c\_dgemm.c, 34
- mic/fftw-3d-omp.c, 32
- naive/gen.cpp, 49
- nbody/cache-block-mic/main.cpp, 40
- nbody/naive/main.cpp, 41
- nbody/no-jump-auto-opt/main.cpp, 41
- nbody/offload/main.cpp, 42
- nbody/omp-parallel-mic/main.cpp, 43
- nbody/omp-parallel/main.cpp, 42
- no-jump-auto-opt/gen.cpp, 50
- offload/gen.cpp, 50
- omp-parallel-mic/gen.cpp, 51
- omp-parallel/gen.cpp, 51
- mat\_init
  - matvec/dynamic-aligned/main.cpp, 35
  - matvec/naive/main.cpp, 36
  - matvec/omp-parallel-mic/main.cpp, 38
  - matvec/omp-parallel/main.cpp, 37
  - matvec/vec-padding/main.cpp, 39
- mat\_vec\_mul
  - dynamic-aligned/matvec.cpp, 43
  - dynamic-aligned/matvec.h, 46
  - naive/matvec.cpp, 44
  - naive/matvec.h, 47
  - omp-parallel-mic/matvec.cpp, 45
  - omp-parallel-mic/matvec.h, 48
  - omp-parallel/matvec.cpp, 44
  - omp-parallel/matvec.h, 47
  - vec-padding/matvec.cpp, 45
  - vec-padding/matvec.h, 48
- matmul/README.md, 31
- matmul/cpu/c\_dgemm.c, 33
- matmul/mic/c\_dgemm.c, 33
- matvec/README.md, 31
- matvec/dynamic-aligned/main.cpp, 34
  - main, 34
  - mat\_init, 35
  - vec\_init, 35
  - vec\_sum, 35
- matvec/dynamic-aligned/matvec.cpp, 43
- matvec/dynamic-aligned/matvec.h, 45
- matvec/naive/main.cpp, 35
  - main, 36
  - mat\_init, 36
  - vec\_init, 36
  - vec\_sum, 36
- matvec/naive/matvec.cpp, 43
- matvec/naive/matvec.h, 47
- matvec/omp-parallel-mic/main.cpp, 38
  - main, 38
- mat\_init, 38
- vec\_init, 38
- vec\_sum, 38
- matvec/omp-parallel-mic/matvec.cpp, 44
- matvec/omp-parallel-mic/matvec.h, 48
- matvec/omp-parallel/main.cpp, 36
  - main, 37
  - mat\_init, 37
  - vec\_init, 37
  - vec\_sum, 37
- matvec/omp-parallel/matvec.cpp, 44
- matvec/omp-parallel/matvec.h, 47
- matvec/vec-padding/main.cpp, 39
  - COL\_PAD, 40
  - main, 39
  - mat\_init, 39
  - vec\_init, 39
  - vec\_sum, 40
- matvec/vec-padding/matvec.cpp, 45
- matvec/vec-padding/matvec.h, 48
- mf2
  - numpy\_dgemm, 24
  - scipy\_dgemm, 25
- mic/c\_dgemm.c
  - ALPHA, 34
  - BETA, 34
  - main, 34
- mic/fftw-3d-omp.c
  - main, 32
- N
  - numpy\_dgemm, 24
  - scipy\_dgemm, 25
- naive/gen.cpp
  - main, 49
  - randf, 49
- naive/matvec.cpp
  - mat\_vec\_mul, 44
- naive/matvec.h
  - mat\_vec\_mul, 47
- naive/nbody.cpp
  - particles\_read, 54
  - particles\_simulate, 54
  - particles\_write, 54
- naive/nbody.h
  - G, 63
  - particles\_read, 63
  - particles\_simulate, 64
  - particles\_write, 64
  - t\_particles, 63
- nbody/README.md, 31
- nbody/cache-block-mic/gen.cpp, 48
- nbody/cache-block-mic/main.cpp, 40
  - main, 40
- nbody/cache-block-mic/nbody.cpp, 52
- nbody/cache-block-mic/nbody.h, 61
- nbody/naive/gen.cpp, 49
- nbody/naive/main.cpp, 40
  - main, 41



- nbody/naive/nbody.cpp, 53
- nbody/naive/nbody.h, 63
- nbody/no-jump-auto-opt/gen.cpp, 50
- nbody/no-jump-auto-opt/main.cpp, 41
  - main, 41
- nbody/no-jump-auto-opt/nbody.cpp, 54
- nbody/no-jump-auto-opt/nbody.h, 64
- nbody/offload/gen.cpp, 50
- nbody/offload/main.cpp, 41
  - main, 42
- nbody/offload/nbody.cpp, 56
- nbody/offload/nbody.h, 67
- nbody/omp-parallel-mic/gen.cpp, 51
- nbody/omp-parallel-mic/main.cpp, 42
  - main, 43
- nbody/omp-parallel-mic/nbody.cpp, 59
- nbody/omp-parallel-mic/nbody.h, 71
- nbody/omp-parallel/gen.cpp, 51
- nbody/omp-parallel/main.cpp, 42
  - main, 42
- nbody/omp-parallel/nbody.cpp, 57
- nbody/omp-parallel/nbody.h, 69
- no-jump-auto-opt/gen.cpp
  - main, 50
  - randf, 50
- no-jump-auto-opt/nbody.cpp
  - particles\_alloc, 55
  - particles\_free, 55
  - particles\_init, 55
  - particles\_read, 55
  - particles\_simulate, 56
  - particles\_write, 56
  - SML\_FLT, 55
- no-jump-auto-opt/nbody.h
  - G, 65
  - particles\_alloc, 65
  - particles\_free, 65
  - particles\_init, 65
  - particles\_read, 65
  - particles\_simulate, 65
  - particles\_write, 67
- numpy\_dgemm, 23
  - a1, 23
  - a2, 23
  - ITER, 23
  - K, 23
  - M, 23
  - m1, 24
  - m2, 24
  - mf2, 24
  - N, 24
  - t\_end, 24
  - t\_init, 24
  - t\_start, 24
- offload/gen.cpp
  - main, 50
  - randf, 50
- offload/nbody.cpp
  - \_\_attribute\_\_, 56
  - particles\_alloc, 57
  - particles\_free, 57
  - particles\_read, 57
  - particles\_write, 57
- offload/nbody.h
  - \_\_attribute\_\_, 67
  - G, 67
  - particles\_alloc, 68
  - particles\_free, 68
  - particles\_read, 68
  - particles\_write, 68
- omp-parallel-mic/gen.cpp
  - main, 51
  - randf, 51
- omp-parallel-mic/matvec.cpp
  - mat\_vec\_mul, 45
- omp-parallel-mic/matvec.h
  - mat\_vec\_mul, 48
- omp-parallel-mic/nbody.cpp
  - particles\_alloc, 60
  - particles\_free, 60
  - particles\_init, 60
  - particles\_read, 60
  - particles\_simulate, 60
  - particles\_write, 60
  - SML\_FLT, 61
- omp-parallel-mic/nbody.h
  - G, 72
  - particles\_alloc, 72
  - particles\_free, 72
  - particles\_init, 72
  - particles\_read, 73
  - particles\_simulate, 73
  - particles\_write, 73
- omp-parallel/gen.cpp
  - main, 51
  - randf, 51
- omp-parallel/matvec.cpp
  - mat\_vec\_mul, 44
- omp-parallel/matvec.h
  - mat\_vec\_mul, 47
- omp-parallel/nbody.cpp
  - particles\_alloc, 58
  - particles\_free, 58
  - particles\_init, 58
  - particles\_read, 58
  - particles\_simulate, 59
  - particles\_write, 59
  - SML\_FLT, 59
- omp-parallel/nbody.h
  - G, 69
  - particles\_alloc, 69
  - particles\_free, 69
  - particles\_init, 71
  - particles\_read, 71
  - particles\_simulate, 71
  - particles\_write, 71

- particles\_alloc
  - cache-block-mic/nbody.cpp, 52
  - cache-block-mic/nbody.h, 61
  - no-jump-auto-opt/nbody.cpp, 55
  - no-jump-auto-opt/nbody.h, 65
  - offload/nbody.cpp, 57
  - offload/nbody.h, 68
  - omp-parallel-mic/nbody.cpp, 60
  - omp-parallel-mic/nbody.h, 72
  - omp-parallel/nbody.cpp, 58
  - omp-parallel/nbody.h, 69
- particles\_free
  - cache-block-mic/nbody.cpp, 52
  - cache-block-mic/nbody.h, 62
  - no-jump-auto-opt/nbody.cpp, 55
  - no-jump-auto-opt/nbody.h, 65
  - offload/nbody.cpp, 57
  - offload/nbody.h, 68
  - omp-parallel-mic/nbody.cpp, 60
  - omp-parallel-mic/nbody.h, 72
  - omp-parallel/nbody.cpp, 58
  - omp-parallel/nbody.h, 69
- particles\_init
  - cache-block-mic/nbody.cpp, 52
  - cache-block-mic/nbody.h, 62
  - no-jump-auto-opt/nbody.cpp, 55
  - no-jump-auto-opt/nbody.h, 65
  - omp-parallel-mic/nbody.cpp, 60
  - omp-parallel-mic/nbody.h, 72
  - omp-parallel/nbody.cpp, 58
  - omp-parallel/nbody.h, 71
- particles\_read
  - cache-block-mic/nbody.cpp, 53
  - cache-block-mic/nbody.h, 62
  - naive/nbody.cpp, 54
  - naive/nbody.h, 63
  - no-jump-auto-opt/nbody.cpp, 55
  - no-jump-auto-opt/nbody.h, 65
  - offload/nbody.cpp, 57
  - offload/nbody.h, 68
  - omp-parallel-mic/nbody.cpp, 60
  - omp-parallel-mic/nbody.h, 73
  - omp-parallel/nbody.cpp, 58
  - omp-parallel/nbody.h, 71
- particles\_simulate
  - cache-block-mic/nbody.cpp, 53
  - cache-block-mic/nbody.h, 62
  - naive/nbody.cpp, 54
  - naive/nbody.h, 64
  - no-jump-auto-opt/nbody.cpp, 56
  - no-jump-auto-opt/nbody.h, 65
  - omp-parallel-mic/nbody.cpp, 60
  - omp-parallel-mic/nbody.h, 73
  - omp-parallel/nbody.cpp, 59
  - omp-parallel/nbody.h, 71
- particles\_write
  - cache-block-mic/nbody.cpp, 53
  - cache-block-mic/nbody.h, 62
- naive/nbody.cpp, 54
- naive/nbody.h, 64
- no-jump-auto-opt/nbody.cpp, 56
- no-jump-auto-opt/nbody.h, 67
- offload/nbody.cpp, 57
- offload/nbody.h, 68
- omp-parallel-mic/nbody.cpp, 60
- omp-parallel-mic/nbody.h, 73
- omp-parallel/nbody.cpp, 59
- omp-parallel/nbody.h, 71
- pos\_x
  - t\_particle, 27
  - t\_particles\_DA, 29
- pos\_y
  - t\_particle, 27
  - t\_particles\_DA, 29
- pos\_z
  - t\_particle, 27
  - t\_particles\_DA, 29
- python-mkl/README.md, 31
- python-mkl/numpy\_dgemm.py, 73
- python-mkl/scipy\_dgemm.py, 74
- README.md, 31
- randf
  - cache-block-mic/gen.cpp, 49
  - naive/gen.cpp, 49
  - no-jump-auto-opt/gen.cpp, 50
  - offload/gen.cpp, 50
  - omp-parallel-mic/gen.cpp, 51
  - omp-parallel/gen.cpp, 51
- SML\_FLT
  - cache-block-mic/nbody.cpp, 53
  - no-jump-auto-opt/nbody.cpp, 55
  - omp-parallel-mic/nbody.cpp, 61
  - omp-parallel/nbody.cpp, 59
- scipy\_dgemm, 24
  - a1, 25
  - a2, 25
  - ITER, 25
  - K, 25
  - M, 25
  - m1, 25
  - m2, 25
  - mf2, 25
  - N, 25
  - t\_end, 25
  - t\_start, 25
- t\_end
  - numpy\_dgemm, 24
  - scipy\_dgemm, 25
- t\_init
  - numpy\_dgemm, 24
- t\_particle, 27
  - pos\_x, 27
  - pos\_y, 27
  - pos\_z, 27

- vel\_x, [27](#)
- vel\_y, [27](#)
- vel\_z, [28](#)
- weight, [28](#)
- t\_particles
  - naive/nbody.h, [63](#)
- t\_particles\_DA, [28](#)
  - ax, [28](#)
  - ay, [28](#)
  - az, [28](#)
  - fx, [29](#)
  - fy, [29](#)
  - fz, [29](#)
  - pos\_x, [29](#)
  - pos\_y, [29](#)
  - pos\_z, [29](#)
  - vel\_x, [29](#)
  - vel\_y, [29](#)
  - vel\_z, [29](#)
  - weight, [29](#)
- t\_start
  - numpy\_dgemm, [24](#)
  - scipy\_dgemm, [25](#)
- vec-padding/matvec.cpp
  - mat\_vec\_mul, [45](#)
- vec-padding/matvec.h
  - mat\_vec\_mul, [48](#)
- vec\_init
  - matvec/dynamic-aligned/main.cpp, [35](#)
  - matvec/naive/main.cpp, [36](#)
  - matvec/omp-parallel-mic/main.cpp, [38](#)
  - matvec/omp-parallel/main.cpp, [37](#)
  - matvec/vec-padding/main.cpp, [39](#)
- vec\_sum
  - matvec/dynamic-aligned/main.cpp, [35](#)
  - matvec/naive/main.cpp, [36](#)
  - matvec/omp-parallel-mic/main.cpp, [38](#)
  - matvec/omp-parallel/main.cpp, [37](#)
  - matvec/vec-padding/main.cpp, [40](#)
- vel\_x
  - t\_particle, [27](#)
  - t\_particles\_DA, [29](#)
- vel\_y
  - t\_particle, [27](#)
  - t\_particles\_DA, [29](#)
- vel\_z
  - t\_particle, [28](#)
  - t\_particles\_DA, [29](#)
- weight
  - t\_particle, [28](#)
  - t\_particles\_DA, [29](#)