Brownian motion of interacting particles

Generated by Doxygen 1.8.13

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 code_zippull_server Namespace Reference

Variables

- string zip_file_name = "code_brownian_int.zip"
- string string_copy = "scp -r " + "motthoma@161.116.80.211://home/motthoma/" + zip_file_name + ' ./'
- string string_unzip = "unzip " + zip_file_name + " -d ./code_unzip"

4.1.1 Variable Documentation

4.1.1.1 string_copy

```
string code_zippull_server.string_copy = "scp -r " + "motthoma@161.116.80.211://home/motthoma/"
+ zip_file_name + ' ./'
```

4.1.1.2 string_unzip

```
string code_zippull_server.string_unzip = "unzip " + zip_file_name + " -d ./code_unzip"
```

4.1.1.3 zip_file_name

```
string code_zippull_server.zip_file_name = "code_brownian_int.zip"
```

4.2 code_zippush_server Namespace Reference

Variables

- string zip_file_name = "zip code_brownian_int.zip"
- string string_zip = zip_file_name + " ∗.c ∗.h ∗.py makefile"
- string string_copy = "scp -r " + zip_file_name + " motthoma@161.116.80.211://home/motthoma"

4.2.1 Variable Documentation

4.2.1.1 string_copy

```
string code_zippush_server.string_copy = "scp -r " + zip_file_name + " motthoma@161.116.80.↩
211://home/motthoma"
```

4.2.1.2 string_zip

```
string code_zippush_server.string_zip = zip_file_name + " ∗.c ∗.h ∗.py makefile"
```

4.2.1.3 zip_file_name

```
string code_zippush_server.zip_file_name = "zip code_brownian_int.zip"
```

4.3 compile_and_gen_header Namespace Reference

Functions

- def write_conf_int_header (conf_string, int_string)
- def call_make_file (conf_string, int_string)

Variables

- args = list(sys.argv)
- conf_string = args[1]
- int_string = args[2]

4.3.1 Function Documentation

4.3.1.1 call_make_file()

```
def compile_and_gen_header.call_make_file (
              conf_string,
              int_string )
```

4.3.1.2 write_conf_int_header()

```
def compile_and_gen_header.write_conf_int_header (
              conf_string,
              int_string )
```

4.3.2 Variable Documentation

4.3.2.1 args

```
compile_and_gen_header.args = list(sys.argv)
```

4.3.2.2 conf_string

```
compile_and_gen_header.conf_string = args[1]
```

4.3.2.3 int_string

```
compile_and_gen_header.int_string = args[2]
```

4.4 masterinteract Namespace Reference

Variables

- int f = 20
- int sn = 20
- int numtasks = 1
- str_run_script = \
- mpi_valid = os.popen("which mpicc").read()
- int mpi_flag = 0
- int find = f
- string scriptname = "LJ_sn_{0}_F_{1:.2f}".format(sn,f)
- file_name = open(scriptname, "w")
- string runscript = 'qsub ' + scriptname

4.4.1 Variable Documentation

4.4.1.1 f

```
int masterinteract.f = 20
```

4.4.1.2 file_name

```
masterinteract.file_name = open(scriptname, "w")
```

4.4.1.3 find

```
int masterinteract.find = f
```

4.4.1.4 mpi_flag

```
int masterinteract.mpi_flag = 0
```

4.4.1.5 mpi_valid

```
masterinteract.mpi_valid = os.popen("which mpicc").read()
```

4.4.1.6 numtasks

```
int masterinteract.numtasks = 1
```

4.4.1.7 runscript

```
string masterinteract.runscript = 'qsub ' + scriptname
```

4.4.1.8 scriptname

```
string masterinteract.scriptname = "LJ_sn_{0}_F_{1:.2f}".format(sn,f)
```

4.4.1.9 sn

```
int masterinteract.sn = 20
```

4.4.1.10 str_run_script

```
masterinteract.str_run_script = \
```

Chapter 5

Class Documentation

5.1 TAG_SimParams Struct Reference

```
#include <par_sim.h>
```

Public Attributes

- int N
- int setnumb
- int numbtest
- int stepnumb
- int simlong
- double accur
- double deffaccur
- double initwidth
- double F
- int plotpoints
- int testab
- int reset_stepnumb
- double time_step
- int numtasks

5.1.1 Member Data Documentation

5.1.1.1 accur

```
double TAG_SimParams::accur
```

5.1.1.2 deffaccur

```
double TAG_SimParams::deffaccur
```

5.1.1.3 F

```
double TAG_SimParams::F
```

5.1.1.4 initwidth

```
double TAG_SimParams::initwidth
```

5.1.1.5 N

```
int TAG_SimParams::N
```

5.1.1.6 numbtest

```
int TAG_SimParams::numbtest
```

5.1.1.7 numtasks

```
int TAG_SimParams::numtasks
```

5.1.1.8 plotpoints

```
int TAG_SimParams::plotpoints
```

5.1.1.9 reset_stepnumb

```
int TAG_SimParams::reset_stepnumb
```

5.1.1.10 setnumb

```
int TAG_SimParams::setnumb
```

5.1.1.11 simlong

```
int TAG_SimParams::simlong
```

5.1.1.12 stepnumb

```
int TAG_SimParams::stepnumb
```

5.1.1.13 testab

```
int TAG_SimParams::testab
```

5.1.1.14 time_step

```
double TAG_SimParams::time_step
```

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

- par_sim.h

5.2 TransportCoeffs Struct Reference

Public Attributes

- long double meanx
- double meanspd
- long double meanxsqu
- long double meanxqub
- long double msd
- long double mthree
- double deff
- double mu

5.2.1 Detailed Description

structure which contains transport coefficients that are calculated within individual threads.

5.2.2 Member Data Documentation

5.2.2.1 deff

```
double TransportCoeffs::deff
```

5.2.2.2 meanspd

```
double TransportCoeffs::meanspd
```

5.2.2.3 meanx

```
long double TransportCoeffs::meanx
```

5.2.2.4 meanxqub

```
long double TransportCoeffs::meanxqub
```

5.2.2.5 meanxsqu

```
long double TransportCoeffs::meanxsqu
```

5.2.2.6 msd

```
long double TransportCoeffs::msd
```

5.2.2.7 mthree

```
long double TransportCoeffs::mthree
```

5.2.2.8 mu

```
double TransportCoeffs::mu
```

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

- main_brownconf.c

Chapter 6

File Documentation

6.1 code_zippull_server.py File Reference

Namespaces

- code_zippull_server

Variables

- string code_zippull_server.zip_file_name = "code_brownian_int.zip"
- string code_zippull_server.string_copy = "scp -r " + "motthoma@161.116.80.211://home/motthoma/" + zip↩ _file_name + ' ./'
- string code_zippull_server.string_unzip = "unzip " + zip_file_name + " -d ./code_unzip"

6.2 code_zippush_server.py File Reference

Namespaces

- code_zippush_server

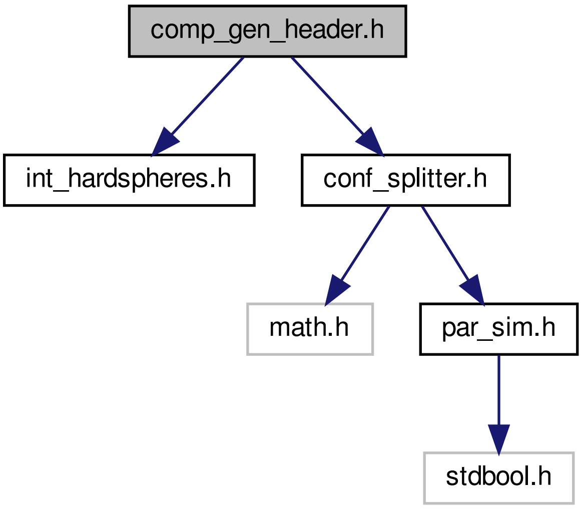
Variables

- string code_zippush_server.zip_file_name = "zip code_brownian_int.zip"
- string code_zippush_server.string_zip = zip_file_name + " ∗.c ∗.h ∗.py makefile"
- string code_zippush_server.string_copy = "scp -r " + zip_file_name + " motthoma@161.116.80.211↩ ://home/motthoma"

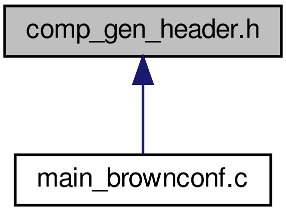
6.3 comp_gen_header.h File Reference

```
#include "int_hardspheres.h"
#include "conf_splitter.h"
```

Include dependency graph for comp_gen_header.h:



This graph shows which files directly or indirectly include this file:



6.4 compile_and_gen_header.py File Reference

Namespaces

- compile_and_gen_header

Functions

- def compile_and_gen_header.write_conf_int_header (conf_string, int_string)
- def compile_and_gen_header.call_make_file (conf_string, int_string)

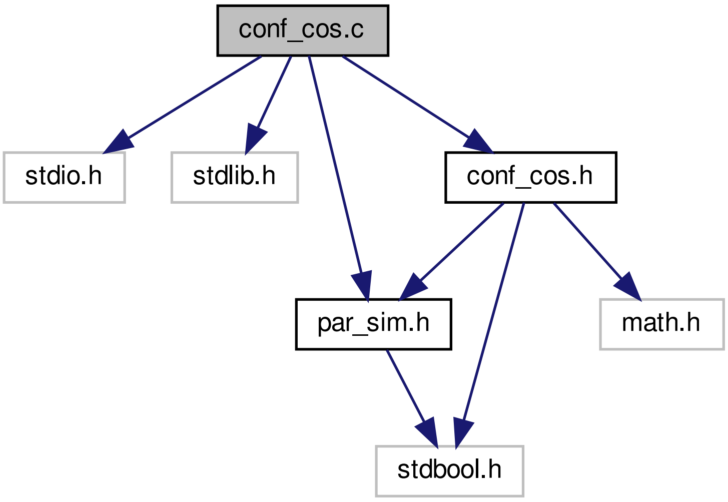
Variables

- compile_and_gen_header.args = list(sys.argv)
- compile_and_gen_header.conf_string = args[1]
- compile_and_gen_header.int_string = args[2]

6.5 conf_cos.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "par_sim.h"
#include "conf_cos.h"
```

Include dependency graph for conf_cos.c:



Functions

- void specs_conf (double binx, double biny, double bin2d)
- void copycode_conf ()
- char ∗ prfx_conf ()

6.5.1 Function Documentation

6.5.1.1 copycode_conf()

```
void copycode_conf ( )
```

6.5.1.2 prfx_conf()

```
char* prfx_conf ( )
```

function to provide prefix in name of working directory

6.5.1.3 specs_conf()

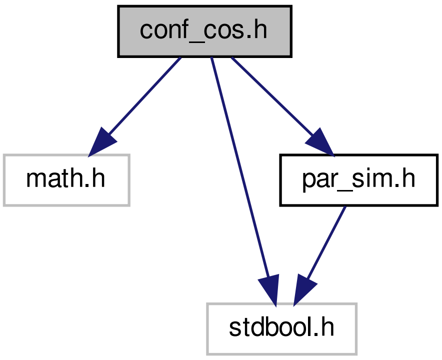
```
void specs_conf (
            double binx,
            double biny,
            double bin2d )
```

function to write confinement specific information in specs file

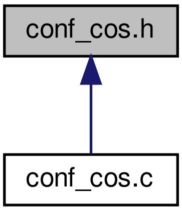
6.6 conf_cos.h File Reference

```
#include <math.h>
#include <stdbool.h>
#include "par_sim.h"
```

Include dependency graph for conf_cos.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define AMP (1.0/(2.0∗M_PI))
- #define MAX_HALF_WIDTH (2∗AMP+B)
- #define CHECKN 5
- #define K_COS (2.0∗M_PI/L)
- #define XMAX (L/4.0-R_CONF)
- #define SQRT_SHIFT R_CONF∗sqrt(1+AMP∗AMP∗K_COS∗K_COS)

Functions

- void specs_conf (double binx, double biny, double bin2d)
- void copycode_conf ()
- char ∗ prfx_conf ()

6.6.1 Macro Definition Documentation

6.6.1.1 AMP

```
#define AMP (1.0/(2.0*M_PI))
```

ensure that only one header for spatial confinement is included in code

6.6.1.2 CHECKN

```
#define CHECKN 5
```

6.6.1.3 K_COS

```
#define K_COS (2.0*M_PI/L)
```

6.6.1.4 MAX_HALF_WIDTH

```
#define MAX_HALF_WIDTH (2*AMP+B)
```

6.6.1.5 SQRT_SHIFT

```
#define SQRT_SHIFT R_CONF*sqrt(1+AMP*AMP*K_COS*K_COS)
```

6.6.1.6 XMAX

```
#define XMAX (L/4.0-R_CONF)
```

6.6.2 Function Documentation

6.6.2.1 copycode_conf()

```
void copycode_conf ( )
```

6.6.2.2 prfx_conf()

```
char* prfx_conf ( )
```

function to provide prefix in name of working directory

function that provides name of confinement

6.6.2.3 specs_conf()

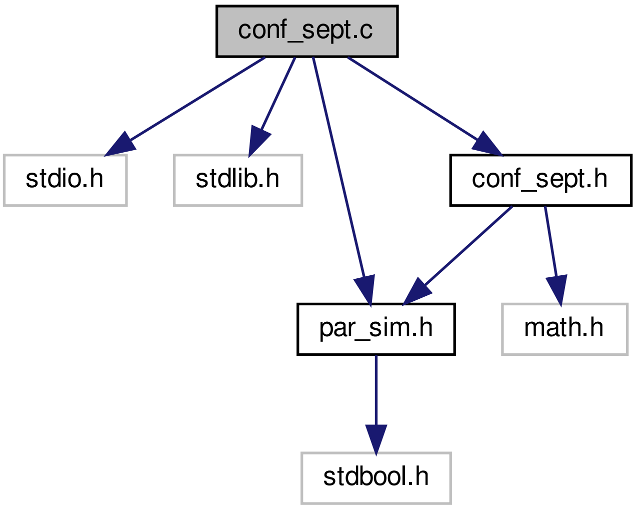
```
void specs_conf (
             double binx,
             double biny,
             double bin2d )
```

function to write confinement specific information in specs file

6.7 conf_sept.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "par_sim.h"
#include "conf_sept.h"
```

Include dependency graph for conf_sept.c:



Functions

- void specs_conf (double binx, double biny, double bin2d)
- void copycode_conf ()
- char ∗ prfx_conf ()

6.7.1 Function Documentation

6.7.1.1 copycode_conf()

```
void copycode_conf ( )
```

6.7.1.2 prfx_conf()

```
char* prfx_conf ( )
```

function to provide prefix in name of working directory

6.7.1.3 specs_conf()

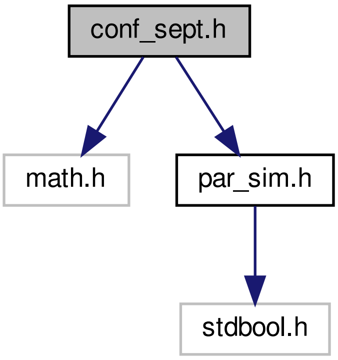
```
void specs_conf (
            double binx,
            double biny,
            double bin2d )
```

function to write confinement specific information in specs file

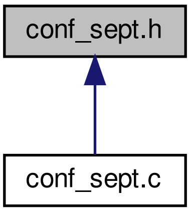
6.8 conf_sept.h File Reference

```
#include <math.h>
#include "par_sim.h"
```

Include dependency graph for conf_sept.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define AMP 1.0
- #define MAX_HALF_WIDTH (AMP+B)
- #define R_CONF_SQ R_CONF∗R_CONF

Functions

- void specs_conf (double binx, double biny, double bin2d)
- void copycode_conf ()
- char ∗ prfx_conf ()

6.8.1 Macro Definition Documentation

6.8.1.1 AMP

```
#define AMP 1.0
```

ensure that only one header for spatial confinement is included in code

6.8.1.2 MAX_HALF_WIDTH

```
#define MAX_HALF_WIDTH (AMP+B)
```

6.8.1.3 R_CONF_SQ

```
#define R_CONF_SQ R_CONF*R_CONF
```

6.8.2 Function Documentation

6.8.2.1 copycode_conf()

```
void copycode_conf ( )
```

6.8.2.2 prfx_conf()

```
char* prfx_conf ( )
```

function to provide prefix in name of working directory

function that provides name of confinement

6.8.2.3 specs_conf()

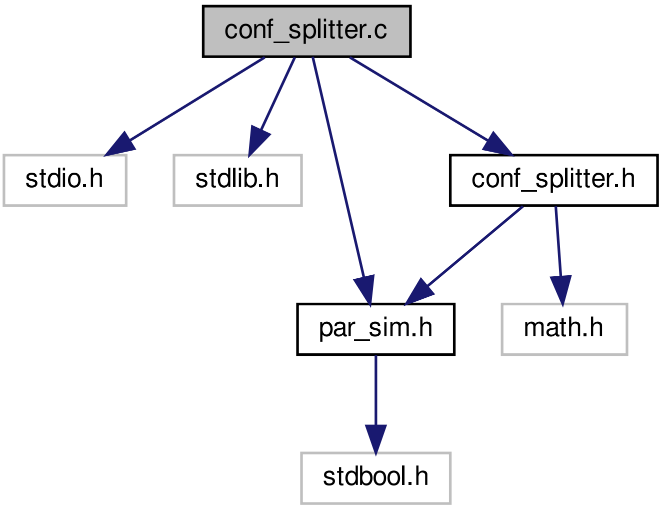
```
void specs_conf (
             double binx,
             double biny,
             double bin2d )
```

function to write confinement specific information in specs file

6.9 conf_splitter.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "par_sim.h"
#include "conf_splitter.h"
```

Include dependency graph for conf_splitter.c:



Functions

- void specs_conf (double binx, double biny, double bin2d)
- void copycode_conf ()
- char ∗ prfx_conf ()

6.9.1 Function Documentation

6.9.1.1 copycode_conf()

```
void copycode_conf ( )
```

6.9.1.2 prfx_conf()

```
char* prfx_conf ( )
```

function that provides name of confinement

6.9.1.3 specs_conf()

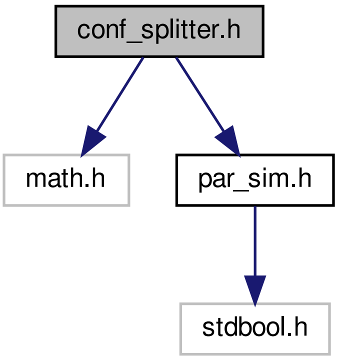
```
void specs_conf (
             double binx,
             double biny,
             double bin2d )
```

function to write confinement specific information in specs file

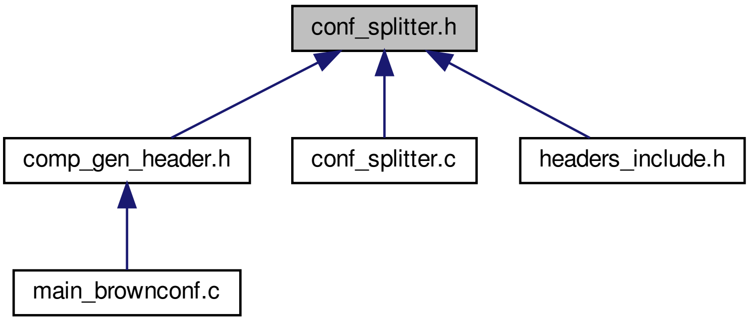
6.10 conf_splitter.h File Reference

```
#include <math.h>
#include "par_sim.h"
```

Include dependency graph for conf_splitter.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define M 0.9
- #define MAX_HALF_WIDTH (M∗L + B)
- #define SQRT_SHIFT R_CONF∗sqrt(1 + M∗M)
- #define R_CONF_SQ R_CONF∗R_CONF
- #define Lp (L - (R_CONF∗M)/(sqrt(1 + M∗M)))

Functions

- void specs_conf (double binx, double biny, double bin2d)
- void copycode_conf ()
- char ∗ prfx_conf ()

6.10.1 Macro Definition Documentation

6.10.1.1 Lp

```
#define Lp (L - (R_CONF*M)/(sqrt(1 + M*M)))
```

6.10.1.2 M

```
#define M 0.9
```

ensure that only one header for spatial confinement is included in code

6.10.1.3 MAX_HALF_WIDTH

```
#define MAX_HALF_WIDTH (M*L + B)
```

6.10.1.4 R_CONF_SQ

```
#define R_CONF_SQ R_CONF*R_CONF
```

6.10.1.5 SQRT_SHIFT

```
#define SQRT_SHIFT R_CONF*sqrt(1 + M*M)
```

6.10.2 Function Documentation

6.10.2.1 copycode_conf()

```
void copycode_conf ( )
```

6.10.2.2 prfx_conf()

```
char* prfx_conf ( )
```

function to provide prefix in name of working directory

function that provides name of confinement

6.10.2.3 specs_conf()

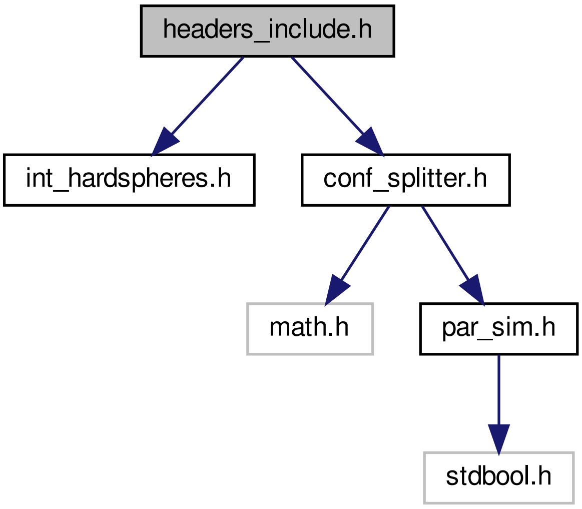
```
void specs_conf (
             double binx,
             double biny,
             double bin2d )
```

function to write confinement specific information in specs file

6.11 headers_include.h File Reference

```
#include "int_hardspheres.h"
#include "conf_splitter.h"
```

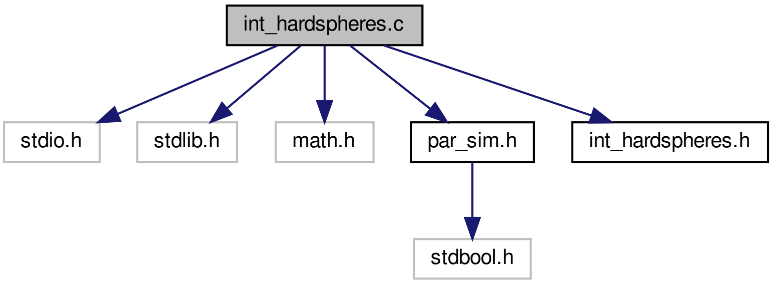
Include dependency graph for headers_include.h:



6.12 int_hardspheres.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "par_sim.h"
#include "int_hardspheres.h"
```

Include dependency graph for int_hardspheres.c:



Functions

- void specs_int (double f_cut)
- void copycode_int ()
- char ∗ prfx_int ()

Variables

- double fintx = 0
- double finty = 0
- double fintxpair = 0
- double fintypair = 0

6.12.1 Function Documentation

6.12.1.1 copycode_int()

```
void copycode_int ( )
```

6.12.1.2 prfx_int()

```
char* prfx_int ( )
```

6.12.1.3 specs_int()

```
void specs_int (
             double f_cut )
```

6.12.2 Variable Documentation

6.12.2.1 fintx

```
double fintx = 0
```

6.12.2.2 fintxpair

```
double fintxpair = 0
```

6.12.2.3 finty

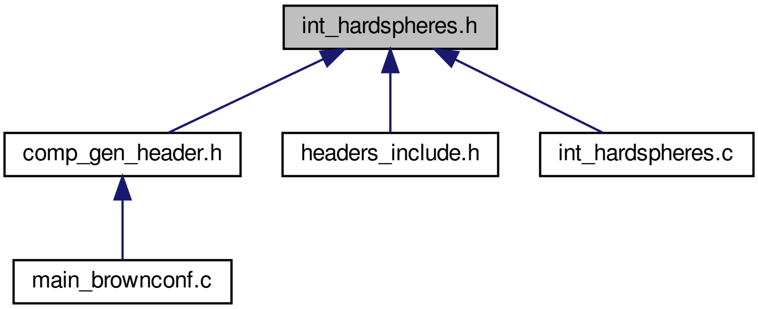
```
double finty = 0
```

6.12.2.4 fintypair

```
double fintypair = 0
```

6.13 int_hardspheres.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define R_INT (1.0∗R_CONF)
- #define INT_CUTOFF 0

Functions

- void specs_int (double f_cut)
- void copycode_int ()
- char ∗ prfx_int ()

Variables

- double fintx
- double finty
- double fintxpair
- double fintypair

6.13.1 Macro Definition Documentation

6.13.1.1 INT_CUTOFF

```
#define INT_CUTOFF 0
```

6.13.1.2 R_INT

```
#define R_INT (1.0*R_CONF)
```

ensure that only one header for intra- particle interaction is included in code

6.13.2 Function Documentation

6.13.2.1 copycode_int()

```
void copycode_int ( )
```

6.13.2.2 prfx_int()

```
char* prfx_int ( )
```

6.13.2.3 specs_int()

```
void specs_int (
             double f_cut )
```

function for the calculation of the intra-particle force arising from a Lennard-Jones Potential. The potential is described by two parameters, position of minimum LJ_MIN and depth of minimum LJ_EPS. The potential is truncated at INT_CUTOFF.

6.13.3 Variable Documentation

6.13.3.1 fintx

```
double fintx
```

6.13.3.2 fintxpair

```
double fintxpair
```

6.13.3.3 finty

```
double finty
```

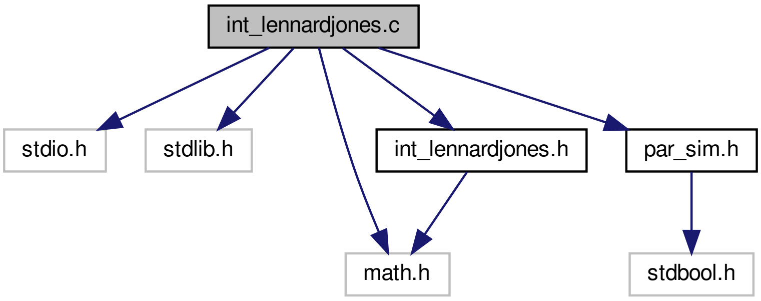
6.13.3.4 fintypair

```
double fintypair
```

6.14 int_lennardjones.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "par_sim.h"
#include "int_lennardjones.h"
```

Include dependency graph for int_lennardjones.c:



Functions

- void specs_int (double f_cut)
- void copycode_int ()
- char ∗ prfx_int ()

Variables

- double fintx = 0
- double finty = 0
- double fintxpair = 0
- double fintypair = 0

6.14.1 Function Documentation

6.14.1.1 copycode_int()

```
void copycode_int ( )
```

6.14.1.2 prfx_int()

```
char* prfx_int ( )
```

6.14.1.3 specs_int()

```
void specs_int (
            double f_cut )
```

function for the calculation of the intra-particle force arising from a Lennard-Jones Potential. The potential is described by two parameters, position of minimum LJ_MIN and depth of minimum LJ_EPS. The potential is truncated at INT_CUTOFF.

6.14.2 Variable Documentation

6.14.2.1 fintx

```
double fintx = 0
```

6.14.2.2 fintxpair

```
double fintxpair = 0
```

6.14.2.3 finty

```
double finty = 0
```

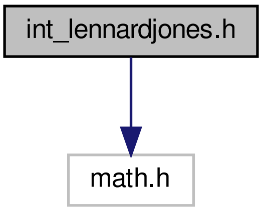
6.14.2.4 fintypair

```
double fintypair = 0
```

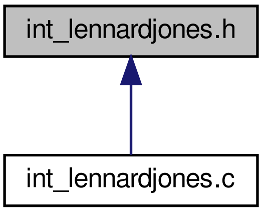
6.15 int_lennardjones.h File Reference

```
#include <math.h>
```

Include dependency graph for int_lennardjones.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define R_INT (R_CONF)
- #define INT_CUTOFF (0.2∗L)
- #define EPS_L (1.0)
- #define LJMIN (0.5∗B)
- #define LJMINPOW (pow(LJMIN,6))
- #define LJPREFAC 12.0∗EPS_L∗LJMINPOW

Functions

- void specs_int (double f_cut)
- void copycode_int ()
- char ∗ prfx_int ()

Variables

- double fintx
- double finty
- double fintxpair
- double fintypair

6.15.1 Macro Definition Documentation

6.15.1.1 EPS_L

```
#define EPS_L (1.0)
```

6.15.1.2 INT_CUTOFF

```
#define INT_CUTOFF (0.2*L)
```

6.15.1.3 LJMIN

```
#define LJMIN (0.5*B)
```

6.15.1.4 LJMINPOW

```
#define LJMINPOW (pow(LJMIN,6))
```

6.15.1.5 LJPREFAC

```
#define LJPREFAC 12.0*EPS_L*LJMINPOW
```

6.15.1.6 R_INT

```
#define R_INT (R_CONF)
```

ensure that only one header for intra- particle interaction is included in code

6.15.2 Function Documentation

6.15.2.1 copycode_int()

```
void copycode_int ( )
```

6.15.2.2 prfx_int()

```
char* prfx_int ( )
```

6.15.2.3 specs_int()

```
void specs_int (
            double f_cut )
```

function for the calculation of the intra-particle force arising from a Lennard-Jones Potential. The potential is described by two parameters, position of minimum LJ_MIN and depth of minimum LJ_EPS. The potential is truncated at INT_CUTOFF.

6.15.3 Variable Documentation

6.15.3.1 fintx

```
double fintx
```

6.15.3.2 fintxpair

```
double fintxpair
```

6.15.3.3 finty

```
double finty
```

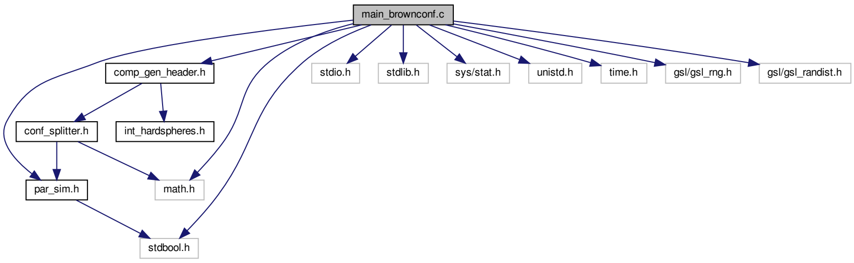
6.15.3.4 fintypair

```
double fintypair
```

6.16 main_brownconf.c File Reference

```
#include "par_sim.h"
#include "comp_gen_header.h"
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <unistd.h>
#include <math.h>
#include <time.h>
#include <stdbool.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
```

Include dependency graph for main_brownconf.c:



Classes

- struct TransportCoeffs

Macros

- #define MASTER 0

Functions

- void calc_transpcoeffs (int setn_per_task, double t, long int **posshift, long int **negshift, double **posx, double **x_init)
- char * makedirectory (char *confprfx, char *intprfx)
- void copy_main ()
- void delerrorfiles ()
- void print_runtime (clock_t start)
- void print_positions (int m, double **posx, double **posy)
- void print_runtime_threads (clock_t start, int numtasks, int taskid)
- void print_resallthreads (long double msdall, double meanspdall, double muall, double deffall, long double meanxall, long double meanxsquall, long double mthreeall, char fname[ ], char fnamemom[ ])
- void print_muoverf (double muall, double deffall, char *namefile)
- int histogramm_mpi_reduce (int m, double backshift, double length, double bin, double **positions, char *fname, int taskid)
- int histogramm2d_mpi_reduce (int m, double bin2d, double **positionsx, double **positionsy, char *fname, int taskid)
- void init_particle_pos (int setn_per_task, double **positionx, double **positiony, double **xstart, gsl_rng *r)
- void init_particle_int (int setn_per_task, double **positionx, double **positiony, double **fintxarray, double **fintyarray)
- void print_hist_countercheck (int xcheck, int ycheck, int twodcheck, char *fname_specs)
- double reset_pos_time (int setn_per_task, long int **posshift, long int **negshift)
- void print_results_over_time (char *fname, char *fnamemom, double t, int abb, int abbdeff)
- void adapt_posshifts (int shiftind, int i, int j, long int **posshift, long int **negshift)
- int update_equcounter (double tran_quant, double tran_quanto, double accurarcy, int equcounter)
- double ** calloc_2Ddouble_array (int m, int n)
- long int ** calloc_2Dlint_array (int m, int n)
- int main (int argc, char **argv)

Variables

- struct TransportCoeffs tcoeff

6.16.1 Macro Definition Documentation

6.16.1.1 MASTER

```
#define MASTER 0
```

6.16.2 Function Documentation

6.16.2.1 adapt_posshifts()

```
void adapt_posshifts (
             int shiftind,
             int i,
             int j,
             long int ** posshift,
             long int ** negshift )
```

function to update shifts which are monitored to calculate absolute position in x-direction from sitcoeff.mulation with cyclic boundary conditions

6.16.2.2 calc_transpcoeffs()

```
void calc_transpcoeffs (
             int setn_per_task,
             double t,
             long int ** posshift,
             long int ** negshift,
             double ** posx,
             double ** x_init )
```

Function that caluclates transport coefficients such as mobility, means-squared displacement or thir moment of position. Function takes relative x-positions and values of shift to first call function for calculation of individual absolute particle positions and subsequently ensemble averages of the first three moments of the position.

6.16.2.3 calloc_2Ddouble_array()

```
double** calloc_2Ddouble_array (
             int m,
             int n )
```

Function that allocates memory for an 2 dimensional array

6.16.2.4 calloc_2Dlint_array()

```
long int** calloc_2Dlint_array (
             int m,
             int n )
```

6.16.2.5 copy_main()

```
void copy_main ( )
```

copies main_brownianconf.c to directory created by 'makedirectory'

6.16.2.6 delerrorfiles()

```
void delerrorfiles ( )
```

deletes error and log files used on albeniz

6.16.2.7 histogramm2d_mpi_reduce()

```
int histogramm2d_mpi_reduce (
             int m,
             double bin2d,
             double ** positionsx,
             double ** positionsy,
             char * fname,
             int taskid )
```

function that stores 2 dimensional histogram in file named fname. during a spatial scan over all nx∗ny rectangular fields, all particle positions are checked and the number of particles in the respective field is increased if a particle is detected. The number counter of all particles in the field is stored in the file together with the upper and lower boundaries in both directions of the field.

6.16.2.8 histogramm_mpi_reduce()

```
int histogramm_mpi_reduce (
             int m,
             double backshift,
             double length,
             double bin,
             double ** positions,
             char * fname,
             int taskid )
```

function that stores 1 dimensional histogram in file named fname. during a spatial sweep through all length/bin slices from position 'backshift' to position 'length', all particle positions are checked and the number of particles in the respective slice is increased if a particle is detected. The number counter of all particles in the slice is stored in the file together with the upper and lower boundary of the slice.

6.16.2.9 init_particle_int()

```
void init_particle_int (
             int setn_per_task,
             double ** positionx,
             double ** positiony,
             double ** fintxarray,
             double ** fintyarray )
```

6.16.2.10 init_particle_pos()

```
void init_particle_pos (
             int setn_per_task,
             double ** positionx,
             double ** positiony,
             double ** xstart,
             gsl_rng * r )
```

6.16.2.11 main()

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

main function of Brownian motion sitcoeff.mulation

calculate number of samples of interacting particles

calculate number of samples of interacting particles per task

6.16.2.12 makedirectory()

```
char* makedirectory (
             char * confprfx,
             char * intprfx )
```

creates directory, where code and data is transferred to moves to this directory as working directory

0700 is modus for determining access rights to dir

6.16.2.13 print_hist_countercheck()

```
void print_hist_countercheck (
             int xcheck,
             int ycheck,
             int twodcheck,
             char * fname_specs )
```

Function that prints the result of the counter checks available from the histogram functions. There, the total number of particles is counted in order to check if all particles are represented in the histogram and situated in the confinement.

6.16.2.14 print_muoverf()

```
void print_muoverf (
             double muall,
             double deffall,
             char * namefile )
```

prints results to file outside of working directory

6.16.2.15 print_positions()

```
void print_positions (
             int m,
             double ** posx,
             double ** posy )
```

prints particle positions to file

6.16.2.16 print_resallthreads()

```
void print_resallthreads (
             long double msdall,
             double meanspdall,
             double muall,
             double deffall,
             long double meanxall,
             long double meanxsquall,
             long double mthreeall,
             char fname[ ],
             char fnamemom[ ] )
```

prints results of sitcoeff.mulation to file

6.16.2.17 print_results_over_time()

```
void print_results_over_time (
             char * fname,
             char * fnamemom,
             double t,
             int abb,
             int abbdeff )
```

function to plot online results of sitcoeff.mulation over time

6.16.2.18 print_runtime()

```
void print_runtime (
             clock_t start )
```

prints run time of code into file

6.16.2.19 print_runtime_threads()

```
void print_runtime_threads (
             clock_t start,
             int numtasks,
             int taskid )
```

prints individual runtime of each thread

measure program run time of task

6.16.2.20 reset_pos_time()

```
double reset_pos_time (
            int setn_per_task,
            long int ** posshift,
            long int ** negshift )
```

Function to reset the sitcoeff.mulated system time t and the positions to originate values in order to skip transient effects from start of sitcoeff.mulation.

6.16.2.21 update_equcounter()

```
int update_equcounter (
            double tran_quant,
            double tran_quanto,
            double accurarcy,
            int equcounter )
```

function for the update of the counter that is used to monitore the equilibration of the system. A transport quantity tran_quant is compared with a previous value. If the difference is below the demanded accurarcy, the value of the counter is increased. If the difference is larger than twice the accurarcy the counter is decreased

6.16.3 Variable Documentation

6.16.3.1 tcoeff

```
struct TransportCoeffs tcoeff
```

6.17 masterinteract.py File Reference

Namespaces

- masterinteract

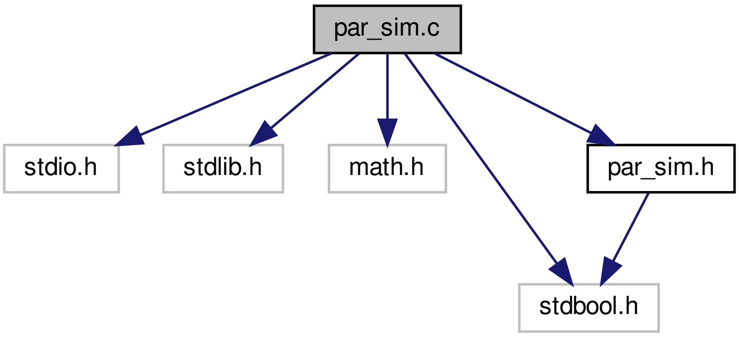
Variables

- int masterinteract.f = 20
- int masterinteract.sn = 20
- int masterinteract.numtasks = 1
- masterinteract.str_run_script = \
- masterinteract.mpi_valid = os.popen("which mpicc").read()
- int masterinteract.mpi_flag = 0
- int masterinteract.find = f
- string masterinteract.scriptname = "LJ_sn_{0}_F_{1:.2f}".format(sn,f)
- masterinteract.file_name = open(scriptname, "w")
- string masterinteract.runscript = 'qsub ' + scriptname

6.18 par_sim.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <stdbool.h>
#include "par_sim.h"
```

Include dependency graph for par_sim.c:



Functions

- void init_simparams ()
- bool check_parameter_consistency ()
- double time_step (double lscale_conf, double lscale_part)
- void specs_basic (char ∗fnamespec)
- void copycode_par ()

Variables

- T_SimParams SimParams

6.18.1 Function Documentation

6.18.1.1 check_parameter_consistency()

```
bool check_parameter_consistency ( )
```

6.18.1.2 copycode_par()

```
void copycode_par ( )
```

function that copies module in working directory

6.18.1.3 init_simparams()

```
void init_simparams ( )
```

Function that initializes hard coded parameters of simulation

6.18.1.4 specs_basic()

```
void specs_basic (
             char * fnamespec )
```

function used to read simulation parameters in specs file

6.18.1.5 time_step()

```
double time_step (
             double lscale_conf,
             double lscale_part )
```

Function that initializes value of simulation time steps. Time steps are calculated by means of confinement and particle quantitites. Therefore, the function has to be called in main, where header files for confinement and particle-particle interactions are included.

6.18.2 Variable Documentation

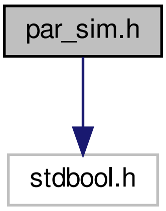
6.18.2.1 SimParams

```
T_SimParams SimParams
```

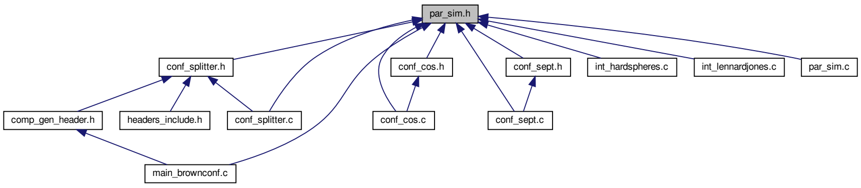
6.19 par_sim.h File Reference

```
#include <stdbool.h>
```

Include dependency graph for par_sim.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct TAG_SimParams

Macros

- #define B 0.1
- #define RADF 0.5
- #define R_CONF (RADF∗B)
- #define BOTTRAD ((RADF == 0 ? 1.0 : (1.0/RADF)))

Typedefs

- typedef struct TAG_SimParams T_SimParams

Functions

- double time_step (double lscale_conf, double lscale_part)
- void init_simparams ()
- bool check_parameter_consistency ()
- void specs_basic (char ∗fnamespec)
- void copycode_par ()

Variables

- unsigned int nbin
- T_SimParams SimParams

6.19.1 Macro Definition Documentation

6.19.1.1 B

```
#define B 0.1
```

6.19.1.2 BOTTRAD

```
#define BOTTRAD ((RADF == 0 ?  1.0 :  (1.0/RADF)))
```

6.19.1.3 R_CONF

```
#define R_CONF (RADF*B)
```

6.19.1.4 RADF

```
#define RADF 0.5
```

6.19.2 Typedef Documentation

6.19.2.1 T_SimParams

```
typedef struct TAG_SimParams T_SimParams
```

6.19.3 Function Documentation

6.19.3.1 check_parameter_consistency()

```
bool check_parameter_consistency ( )
```

6.19.3.2 copycode_par()

```
void copycode_par ( )
```

function that copies module in working directory

6.19.3.3 init_simparams()

```
void init_simparams ( )
```

Function that initializes hard coded parameters of simulation

6.19.3.4 specs_basic()

```
void specs_basic (
             char * fnamespec )
```

function used to read simulation parameters in specs file

6.19.3.5 time_step()

```
double time_step (
             double lscale_conf,
             double lscale_part )
```

Function that initializes value of simulation time steps. Time steps are calculated by means of confinement and particle quantitites. Therefore, the function has to be called in main, where header files for confinement and particle-particle interactions are included.

6.19.4 Variable Documentation

6.19.4.1 nbin

```
unsigned int nbin
```

6.19.4.2 SimParams

```
T_SimParams SimParams
```