phasefield-accelerator-benchmarks

pre-alpha

Generated by Doxygen 1.8.8

Thu Aug 24 2017 11:18:03

Contents

1 Class Index

1.1 Class List

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

ResidualSumOfSquares2D **2**

2 File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

cpu/openmp/boundaries.c **Implementation of boundary condition functions with OpenMP threading**	**3**
cpu/serial/boundaries.c **Implementation of boundary condition functions without threading**	**3**
gpu/cuda/boundaries.c **Implementation of boundary condition functions with OpenMP threading**	**4**
gpu/openacc/boundaries.c **Implementation of boundary condition functions with OpenMP threading**	**5**
boundaries.cpp **Implementation of boundary condition functions with TBB threading**	**5**
cpu/diffusion.h **Declaration of diffusion equation function prototypes for CPU benchmarks**	**6**
gpu/diffusion.h **Declaration of diffusion equation function prototypes for CPU benchmarks**	**7**
cpu/openmp/discretization.c **Implementation of boundary condition functions with OpenMP threading**	**8**

3 Class Documentation

3.1 ResidualSumOfSquares2D Class Reference

Public Member Functions

- **ResidualSumOfSquares2D** (fp_t ∗∗conc_new, int nx, int ny, fp_t dx, fp_t dy, int nm, fp_t elapsed, fp_t D, fp_t c)
- **ResidualSumOfSquares2D** (ResidualSumOfSquares2D &a, tbb::split)
- void **operator()** (const tbb::blocked_range2d< int > &r)
- void **join** (const ResidualSumOfSquares2D &a)

Public Attributes

- fp_t **my_rss**

3.1.1 Detailed Description

Definition at line 131 of file discretization.cpp.

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

- discretization.cpp

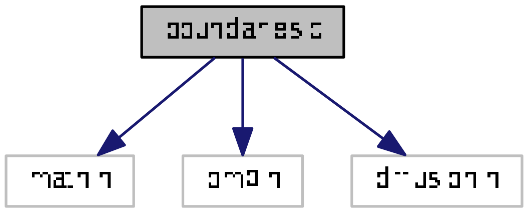
4 File Documentation

4.1 boundaries.c File Reference

Implementation of boundary condition functions with OpenMP threading.

```
#include <math.h>
#include <omp.h>
#include "diffusion.h"
```

Include dependency graph for cpu/openmp/boundaries.c:



Functions

- void **set_boundaries** (fp_t bc[2][2])
- void **apply_initial_conditions** (fp_t ∗∗conc, int nx, int ny, int nm, fp_t bc[2][2])
- void **apply_boundary_conditions** (fp_t ∗∗conc, int nx, int ny, int nm, fp_t bc[2][2])

4.1.1 Detailed Description

Implementation of boundary condition functions with OpenMP threading.

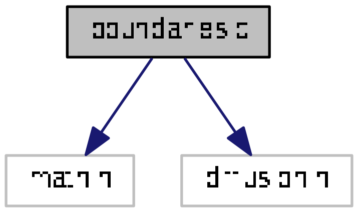
Definition in file cpu/openmp/boundaries.c.

4.2 boundaries.c File Reference

Implementation of boundary condition functions without threading.

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

Include dependency graph for cpu/serial/boundaries.c:



Functions

- void **set_boundaries** (fp_t bc[2][2])
- void **apply_initial_conditions** (fp_t ∗∗conc, int nx, int ny, int nm, fp_t bc[2][2])
- void **apply_boundary_conditions** (fp_t ∗∗conc, int nx, int ny, int nm, fp_t bc[2][2])

4.2.1 Detailed Description

Implementation of boundary condition functions without threading.

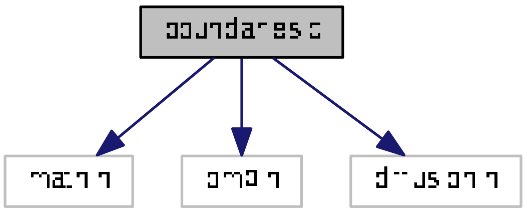
Definition in file cpu/serial/boundaries.c.

4.3 boundaries.c File Reference

Implementation of boundary condition functions with OpenMP threading.

```
#include <math.h>
#include <omp.h>
#include "diffusion.h"
```

Include dependency graph for gpu/cuda/boundaries.c:



Functions

- void **set_boundaries** (fp_t bc[2][2])

- void **apply_initial_conditions** (fp_t $**$conc, int nx, int ny, int nm, fp_t bc[2][2])
- void **apply_boundary_conditions** (fp_t $**$conc, int nx, int ny, int nm, fp_t bc[2][2])

4.3.1 Detailed Description

Implementation of boundary condition functions with OpenMP threading.

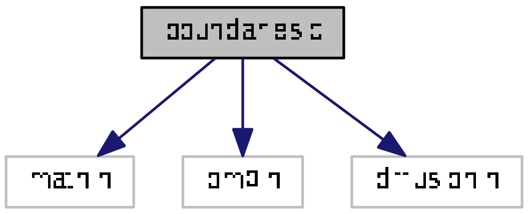
Definition in file gpu/cuda/boundaries.c.

4.4 boundaries.c File Reference

Implementation of boundary condition functions with OpenMP threading.

```
#include <math.h>
#include <omp.h>
#include "diffusion.h"
```

Include dependency graph for gpu/openacc/boundaries.c:



Functions

- void **set_boundaries** (fp_t bc[2][2])
- void **apply_initial_conditions** (fp_t $**$conc, int nx, int ny, int nm, fp_t bc[2][2])
- void **apply_boundary_conditions** (fp_t $**$conc, int nx, int ny, int nm, fp_t bc[2][2])

4.4.1 Detailed Description

Implementation of boundary condition functions with OpenMP threading.

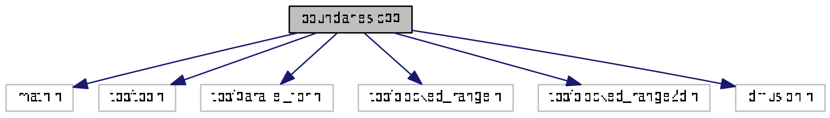
Definition in file gpu/openacc/boundaries.c.

4.5 boundaries.cpp File Reference

Implementation of boundary condition functions with TBB threading.

```
#include <math.h>
#include <tbb/tbb.h>
#include <tbb/parallel_for.h>
#include <tbb/blocked_range.h>
#include <tbb/blocked_range2d.h>
#include "diffusion.h"
```

Include dependency graph for boundaries.cpp:



Functions

- void **set_boundaries** (fp_t bc[2][2])
- void **apply_initial_conditions** (fp_t ∗∗conc, int nx, int ny, int nm, fp_t bc[2][2])
- void **apply_boundary_conditions** (fp_t ∗∗conc, int nx, int ny, int nm, fp_t bc[2][2])

4.5.1 Detailed Description

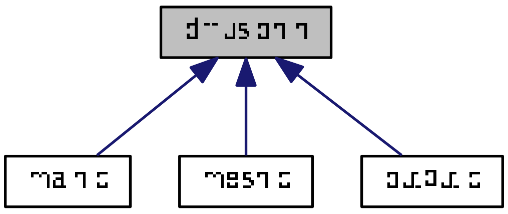
Implementation of boundary condition functions with TBB threading.

Definition in file boundaries.cpp.

4.6 diffusion.h File Reference

Declaration of diffusion equation function prototypes for CPU benchmarks.

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



Typedefs

- typedef double **fp_t**

Functions

- void **make_arrays** (fp_t ∗∗∗conc_old, fp_t ∗∗∗conc_new, fp_t ∗∗∗conc_lap, fp_t ∗∗∗mask_lap, int nx, int ny, int nm)
- void **free_arrays** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap)
- void **swap_pointers** (fp_t ∗∗∗conc_old, fp_t ∗∗∗conc_new)
- void **set_boundaries** (fp_t bc[2][2])

- void **apply_initial_conditions** (fp_t ∗∗conc_old, int nx, int ny, int nm, fp_t bc[2][2])
- void **apply_boundary_conditions** (fp_t ∗∗conc_old, int nx, int ny, int nm, fp_t bc[2][2])
- void **set_threads** (int n)
- void **set_mask** (fp_t dx, fp_t dy, int nm, fp_t ∗∗mask_lap)
- void **compute_convolution** (fp_t ∗∗conc_old, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap, int nx, int ny, int nm)
- void **solve_diffusion_equation** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, int nx, int ny, int nm, fp_t D, fp_t dt, fp_t ∗elapsed)
- void **analytical_value** (fp_t x, fp_t t, fp_t D, fp_t bc[2][2], fp_t ∗c)
- void **check_solution** (fp_t ∗∗conc_new, int nx, int ny, fp_t dx, fp_t dy, int nm, fp_t elapsed, fp_t D, fp_t bc[2][2], fp_t ∗rss)
- void **print_progress** (const int step, const int steps)
- void **write_csv** (fp_t ∗∗conc, int nx, int ny, fp_t dx, fp_t dy, int step)
- void **write_png** (fp_t ∗∗conc, int nx, int ny, int step)
- void **StartTimer** ()
- double **GetTimer** ()

4.6.1 Detailed Description

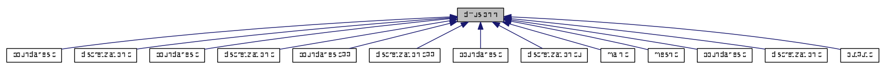
Declaration of diffusion equation function prototypes for CPU benchmarks.

Definition in file cpu/diffusion.h.

4.7 diffusion.h File Reference

Declaration of diffusion equation function prototypes for CPU benchmarks.

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



Typedefs

- typedef double **fp_t**

Functions

- void **make_arrays** (fp_t ∗∗∗conc_old, fp_t ∗∗∗conc_new, fp_t ∗∗∗conc_lap, fp_t ∗∗∗mask_lap, int nx, int ny, int nm)
- void **free_arrays** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap)
- void **swap_pointers** (fp_t ∗∗∗conc_old, fp_t ∗∗∗conc_new)
- void **set_boundaries** (fp_t bc[2][2])
- void **apply_initial_conditions** (fp_t ∗∗conc_old, int nx, int ny, int nm, fp_t bc[2][2])
- void **apply_boundary_conditions** (fp_t ∗∗conc_old, int nx, int ny, int nm, fp_t bc[2][2])
- void **set_threads** (int n)
- void **set_mask** (fp_t dx, fp_t dy, int nm, fp_t ∗∗mask_lap)
- void **compute_convolution** (fp_t ∗∗conc_old, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap, int nx, int ny, int nm, int bs)
- void **solve_diffusion_equation** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, int nx, int ny, int nm, int bs, fp_t D, fp_t dt, fp_t ∗elapsed)
- void **analytical_value** (fp_t x, fp_t t, fp_t D, fp_t bc[2][2], fp_t ∗c)
- void **check_solution** (fp_t ∗∗conc_new, int nx, int ny, fp_t dx, fp_t dy, int nm, int bs, fp_t elapsed, fp_t D, fp_t bc[2][2], fp_t ∗rss)

- void **print_progress** (const int step, const int steps)
- void **write_csv** (fp_t ∗∗conc, int nx, int ny, fp_t dx, fp_t dy, int step)
- void **write_png** (fp_t ∗∗conc, int nx, int ny, int step)
- void **StartTimer** ()
- double **GetTimer** ()

4.7.1 Detailed Description

Declaration of diffusion equation function prototypes for CPU benchmarks.

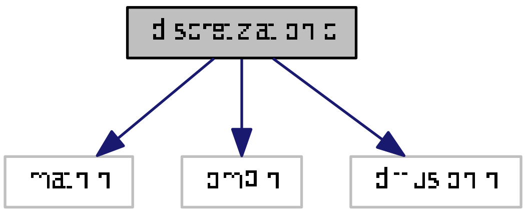
Definition in file gpu/diffusion.h.

4.8 discretization.c File Reference

Implementation of boundary condition functions with OpenMP threading.

```
#include <math.h>
#include <omp.h>
#include "diffusion.h"
```

Include dependency graph for cpu/openmp/discretization.c:



Functions

- void **set_threads** (int n)
- void **five_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **slow_nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **set_mask** (fp_t dx, fp_t dy, int nm, fp_t ∗∗mask_lap)
- void **compute_convolution** (fp_t ∗∗conc_old, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap, int nx, int ny, int nm)
- void **solve_diffusion_equation** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, int nx, int ny, int nm, fp_t D, fp_t dt, fp_t ∗elapsed)
- void **analytical_value** (fp_t x, fp_t t, fp_t D, fp_t bc[2][2], fp_t ∗c)
- void **check_solution** (fp_t ∗∗conc_new, int nx, int ny, fp_t dx, fp_t dy, int nm, fp_t elapsed, fp_t D, fp_t bc[2][2], fp_t ∗rss)

4.8.1 Detailed Description

Implementation of boundary condition functions with OpenMP threading.

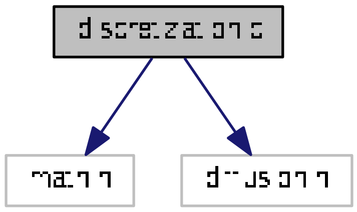
Definition in file cpu/openmp/discretization.c.

4.9 discretization.c File Reference

Implementation of boundary condition functions without threading.

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

Include dependency graph for cpu/serial/discretization.c:



Functions

- void **set_threads** (int n)
- void **five_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **slow_nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **set_mask** (fp_t dx, fp_t dy, int nm, fp_t ∗∗mask_lap)
- void **compute_convolution** (fp_t ∗∗conc_old, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap, int nx, int ny, int nm)
- void **solve_diffusion_equation** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, int nx, int ny, int nm, fp_t D, fp_t dt, fp_t ∗elapsed)
- void **analytical_value** (fp_t x, fp_t t, fp_t D, fp_t bc[2][2], fp_t ∗c)
- void **check_solution** (fp_t ∗∗conc_new, int nx, int ny, fp_t dx, fp_t dy, int nm, fp_t elapsed, fp_t D, fp_t bc[2][2], fp_t ∗rss)

4.9.1 Detailed Description

Implementation of boundary condition functions without threading.

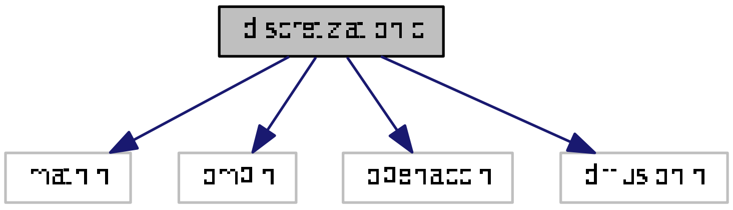
Definition in file cpu/serial/discretization.c.

4.10 discretization.c File Reference

Implementation of boundary condition functions with OpenACC threading.

```
#include <math.h>
#include <omp.h>
#include <openacc.h>
#include "diffusion.h"
```

Include dependency graph for gpu/openacc/discretization.c:



Functions

- void **set_threads** (int n)
- void **five_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **slow_nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **set_mask** (fp_t dx, fp_t dy, int nm, fp_t ∗∗mask_lap)
- void **compute_convolution** (fp_t ∗∗conc_old, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap, int nx, int ny, int nm, int bs)
- void **solve_diffusion_equation** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, int nx, int ny, int nm, int bs, fp_t D, fp_t dt, fp_t ∗elapsed)
- void **check_solution** (fp_t ∗∗conc_new, int nx, int ny, fp_t dx, fp_t dy, int nm, int bs, fp_t elapsed, fp_t D, fp_t bc[2][2], fp_t ∗rss)

4.10.1 Detailed Description

Implementation of boundary condition functions with OpenACC threading.

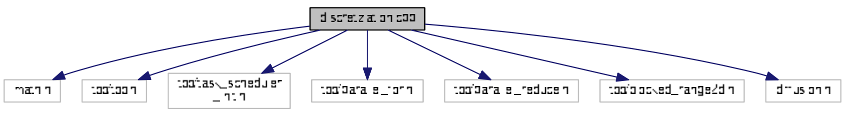
Definition in file gpu/openacc/discretization.c.

4.11 discretization.cpp File Reference

Implementation of boundary condition functions with TBB threading.

```
#include <math.h>
#include <tbb/tbb.h>
#include <tbb/task_scheduler_init.h>
#include <tbb/parallel_for.h>
#include <tbb/parallel_reduce.h>
#include <tbb/blocked_range2d.h>
#include "diffusion.h"
```

Include dependency graph for discretization.cpp:



Classes

- class ResidualSumOfSquares2D

Functions

- void **set_threads** (int n)
- void **five_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **slow_nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **set_mask** (fp_t dx, fp_t dy, int nm, fp_t ∗∗mask_lap)
- void **compute_convolution** (fp_t ∗∗conc_old, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap, int nx, int ny, int nm)
- void **solve_diffusion_equation** (fp_t ∗∗conc_old, fp_t ∗∗B, fp_t ∗∗conc_lap, int nx, int ny, int nm, fp_t D, fp_t dt, fp_t ∗elapsed)
- void **analytical_value** (fp_t x, fp_t t, fp_t D, fp_t chi, fp_t ∗c)
- void **check_solution** (fp_t ∗∗conc_new, int nx, int ny, fp_t dx, fp_t dy, int nm, fp_t elapsed, fp_t D, fp_t bc[2][2], fp_t ∗rss)

4.11.1 Detailed Description

Implementation of boundary condition functions with TBB threading.

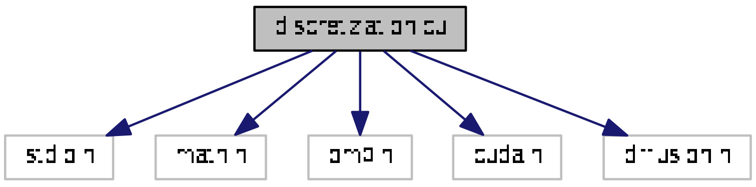
Definition in file discretization.cpp.

4.12 discretization.cu File Reference

Implementation of boundary condition functions with CUDA acceleration.

```
#include <stdio.h>
#include <math.h>
#include <omp.h>
#include <cuda.h>
#include "diffusion.h"
```

Include dependency graph for discretization.cu:



Macros

- #define **MAX_TILE_W** 32
- #define **MAX_TILE_H** 32
- #define **MAX_MASK_W** 3
- #define **MAX_MASK_SIZE** (MAX_MASK_W ∗ MAX_MASK_W)

Functions

- void **set_threads** (int n)
- void **five_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **slow_nine_point_Laplacian_stencil** (fp_t dx, fp_t dy, fp_t ∗∗mask_lap)
- void **set_mask** (fp_t dx, fp_t dy, int nm, fp_t ∗∗mask_lap)
- __global__ void **convolution_kernel** (fp_t ∗conc_old, fp_t ∗conc_lap, int nx, int ny, int nm)
- void **compute_convolution** (fp_t ∗∗conc_old, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap, int nx, int ny, int nm, int bs)
- __global__ void **diffusion_kernel** (fp_t ∗conc_old, fp_t ∗conc_new, fp_t ∗conc_lap, int nx, int ny, int nm, fp_t D, fp_t dt)
- void **solve_diffusion_equation** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, int nx, int ny, int nm, int bs, fp_t D, fp_t dt, fp_t ∗elapsed)
- void **analytical_value** (fp_t x, fp_t t, fp_t D, fp_t bc[2][2], fp_t ∗c)
- void **check_solution** (fp_t ∗∗conc_new, int nx, int ny, fp_t dx, fp_t dy, int nm, int bs, fp_t elapsed, fp_t D, fp_t bc[2][2], fp_t ∗rss)

Variables

- __constant__ fp_t **Mc** [MAX_MASK_SIZE]

4.12.1 Detailed Description

Implementation of boundary condition functions with CUDA acceleration.

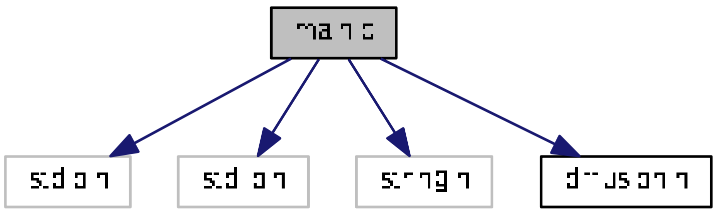
Definition in file discretization.cu.

4.13 main.c File Reference

Implementation of semi-infinite diffusion equation.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "diffusion.h"
```

Include dependency graph for cpu/main.c:



Functions

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

4.13.1 Detailed Description

Implementation of semi-infinite diffusion equation.

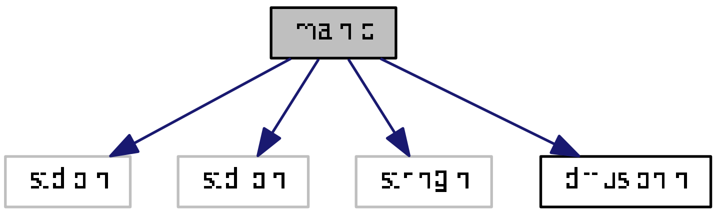
Definition in file cpu/main.c.

4.14 main.c File Reference

Implementation of semi-infinite diffusion equation.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "diffusion.h"
```

Include dependency graph for gpu/main.c:



Functions

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

4.14.1 Detailed Description

Implementation of semi-infinite diffusion equation.

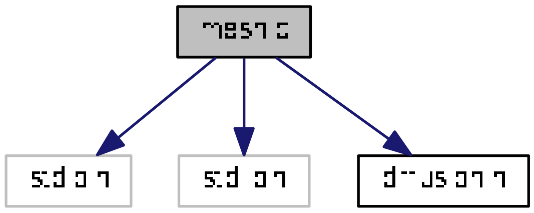
Definition in file gpu/main.c.

4.15 mesh.c File Reference

Implemenatation of mesh handling functions.

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

Include dependency graph for cpu/mesh.c:



Functions

- void **make_arrays** (fp_t ∗∗∗conc_old, fp_t ∗∗∗conc_new, fp_t ∗∗∗conc_lap, fp_t ∗∗∗mask_lap, int nx, int ny, int nm)
- void **free_arrays** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap)
- void **swap_pointers** (fp_t ∗∗∗conc_old, fp_t ∗∗∗conc_new)

4.15.1 Detailed Description

Implemenatation of mesh handling functions.

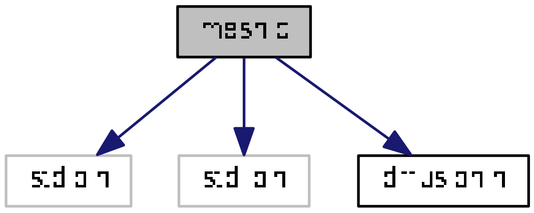
Definition in file cpu/mesh.c.

4.16 mesh.c File Reference

Implemenatation of mesh handling functions.

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

Include dependency graph for gpu/mesh.c:



Functions

- void **make_arrays** (fp_t ∗∗∗conc_old, fp_t ∗∗∗conc_new, fp_t ∗∗∗conc_lap, fp_t ∗∗∗mask_lap, int nx, int ny, int nm)
- void **free_arrays** (fp_t ∗∗conc_old, fp_t ∗∗conc_new, fp_t ∗∗conc_lap, fp_t ∗∗mask_lap)
- void **swap_pointers** (fp_t ∗∗∗conc_old, fp_t ∗∗∗conc_new)

4.16.1 Detailed Description

Implemenatation of mesh handling functions.

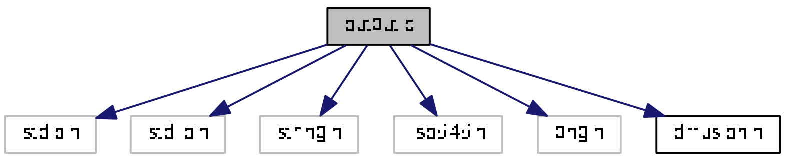
Definition in file gpu/mesh.c.

4.17 output.c File Reference

Implementation of file output functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <iso646.h>
#include <png.h>
#include "diffusion.h"
```

Include dependency graph for cpu/output.c:



Functions

- void **print_progress** (const int step, const int steps)
- void **write_csv** (fp_t ∗∗conc, int nx, int ny, fp_t dx, fp_t dy, int step)
- void **write_png** (fp_t ∗∗conc, int nx, int ny, int step)

4.17.1 Detailed Description

Implementation of file output functions.

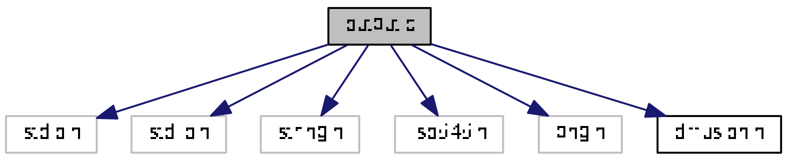
Definition in file cpu/output.c.

4.18 output.c File Reference

Implementation of file output functions.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <iso646.h>
#include <png.h>
#include "diffusion.h"
```

Include dependency graph for gpu/output.c:



Functions

- void **print_progress** (const int step, const int steps)
- void **write_csv** (fp_t ∗∗conc, int nx, int ny, fp_t dx, fp_t dy, int step)
- void **write_png** (fp_t ∗∗conc, int nx, int ny, int step)

4.18.1 Detailed Description

Implementation of file output functions.

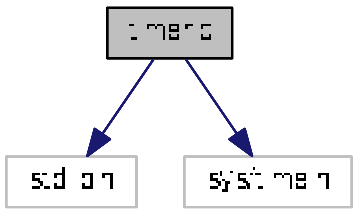
Definition in file gpu/output.c.

4.19 timer.c File Reference

High-resolution cross-platform machine time reader.

```
#include <stdlib.h>
#include <sys/time.h>
```

Include dependency graph for cpu/timer.c:



Functions

- void **StartTimer** ()
- double **GetTimer** ()

Variables

- struct timeval **timerStart**

4.19.1 Detailed Description

High-resolution cross-platform machine time reader.

Author

NVIDIA

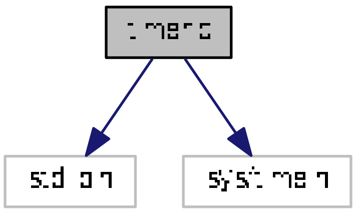
Definition in file cpu/timer.c.

4.20 timer.c File Reference

High-resolution cross-platform machine time reader.

```
#include <stdlib.h>
#include <sys/time.h>
```

Include dependency graph for gpu/timer.c:



Functions

- void **StartTimer** ()
- double **GetTimer** ()

Variables

- struct timeval **timerStart**

4.20.1 Detailed Description

High-resolution cross-platform machine time reader.

Author

NVIDIA

Definition in file gpu/timer.c.