

phasefield-accelerator-benchmarks  
pre-alpha

Generated by Doxygen 1.8.8

Thu Aug 24 2017 00:43:33

## Contents

|          |  |          |
|----------|--|----------|
| <b>1</b> | <b>Class Index</b>   | <b>1</b> |
| 1.1      | Class List . . . . .   | 1        |
| <b>2</b> | <b>File Index</b>  | <b>1</b> |
| 2.1      | File List . . . . .  | 1        |
| <b>3</b> | <b>Class Documentation</b>   | <b>2</b> |
| 3.1      | ResidualSumOfSquares2D Class Reference . . . . .   | 2        |
| 3.1.1    | Detailed Description . . . . .   | 3        |
| <b>4</b> | <b>File Documentation</b>  | <b>3</b> |
| 4.1      | /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/diffusion.h File Reference . .                   | 3        |
| 4.1.1    | Detailed Description . . . . .   | 4        |
| 4.2      | /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/diffusion.h File Reference . .                   | 4        |
| 4.2.1    | Detailed Description . . . . .   | 5        |
| 4.3      | /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/main.c File Reference . . .                      | 5        |
| 4.3.1    | Detailed Description . . . . .   | 5        |
| 4.4      | /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/main.c File Reference . . .                      | 5        |
| 4.4.1    | Detailed Description . . . . .   | 6        |
| 4.5      | /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/mesh.c File Reference . . .                      | 6        |
| 4.5.1    | Detailed Description . . . . .   | 7        |
| 4.6      | /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/mesh.c File Reference . . .                      | 7        |
| 4.6.1    | Detailed Description . . . . .   | 7        |
| 4.7      | /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/openmp/boundaries.c File Reference . . . . .     | 8        |
| 4.7.1    | Detailed Description . . . . .   | 8        |
| 4.8      | /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/serial/boundaries.c File Reference . . . . .     | 8        |
| 4.8.1    | Detailed Description . . . . .   | 9        |
| 4.9      | /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/cuda/boundaries.c File Reference . . . . .       | 9        |
| 4.9.1    | Detailed Description . . . . .   | 10       |
| 4.10     | /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/openacc/boundaries.c File Reference . . . . .    | 10       |
| 4.10.1   | Detailed Description . . . . .   | 11       |
| 4.11     | /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/openmp/discretization.c File Reference . . . . . | 11       |
| 4.11.1   | Detailed Description . . . . .   | 12       |
| 4.12     | /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/serial/discretization.c File Reference . . . . . | 12       |
| 4.12.1   | Detailed Description . . . . .   | 13       |

|        |   |    |
|--------|---|----|
| 4.13   | <a href="#">/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/openacc/discretization.c File Reference</a> | 13 |
| 4.13.1 | <a href="#">Detailed Description</a>  | 14 |
| 4.14   | <a href="#">/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/output.c File Reference</a>                 | 14 |
| 4.14.1 | <a href="#">Detailed Description</a>  | 15 |
| 4.15   | <a href="#">/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/output.c File Reference</a>                 | 15 |
| 4.15.1 | <a href="#">Detailed Description</a>  | 16 |
| 4.16   | <a href="#">/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/tbb/boundaries.cpp File Reference</a>       | 16 |
| 4.16.1 | <a href="#">Detailed Description</a>  | 16 |
| 4.17   | <a href="#">/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/tbb/discretization.cpp File Reference</a>   | 16 |
| 4.17.1 | <a href="#">Detailed Description</a>  | 17 |
| 4.18   | <a href="#">/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/timer.c File Reference</a>                  | 17 |
| 4.18.1 | <a href="#">Detailed Description</a>  | 18 |
| 4.19   | <a href="#">/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/timer.c File Reference</a>                  | 18 |
| 4.19.1 | <a href="#">Detailed Description</a>  | 19 |
| 4.20   | <a href="#">/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/cuda/discretization.cu File Reference</a>   | 19 |
| 4.20.1 | <a href="#">Detailed Description</a>  | 20 |

## 1 Class Index

### 1.1 Class List

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

|   |          |
|---|----------|
| <a href="#"><b>ResidualSumOfSquares2D</b></a> | <b>2</b> |
|---|----------|

## 2 File Index

### 2.1 File List

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

|  |           |
|--|-----------|
| <a href="#"><b>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/diffusion.h</b></a><br>Declaration of diffusion equation function prototypes for CPU benchmarks | <b>3</b>  |
| <a href="#"><b>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/main.c</b></a><br>Implementation of semi-infinite diffusion equation                            | <b>5</b>  |
| <a href="#"><b>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/mesh.c</b></a><br>Implementatation of mesh handling functions                                   | <b>6</b>  |
| <a href="#"><b>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/output.c</b></a><br>Implementation of file output functions                                     | <b>14</b> |

|  |    |
|--|----|
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/timer.c</code><br>High-resolution cross-platform machine time reader                                     | 17 |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/openmp/boundaries.c</code><br>Implementation of boundary condition functions with OpenMP threading       | 8  |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/openmp/discretization.c</code><br>Implementation of boundary condition functions with OpenMP threading   | 11 |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/serial/boundaries.c</code><br>Implementation of boundary condition functions without threading           | 8  |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/serial/discretization.c</code><br>Implementation of boundary condition functions without threading       | 12 |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/tbb/boundaries.cpp</code><br>Implementation of boundary condition functions with TBB threading           | 16 |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/tbb/discretization.cpp</code><br>Implementation of boundary condition functions with TBB threading       | 16 |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/diffusion.h</code><br>Declaration of diffusion equation function prototypes for CPU benchmarks           | 4  |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/main.c</code><br>Implementation of semi-infinite diffusion equation                                      | 5  |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/mesh.c</code><br>Implementation of mesh handling functions   | 7  |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/output.c</code><br>Implementation of file output functions   | 15 |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/timer.c</code><br>High-resolution cross-platform machine time reader                                     | 18 |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/cuda/boundaries.c</code><br>Implementation of boundary condition functions with OpenMP threading         | 9  |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/cuda/discretization.cu</code><br>Implementation of boundary condition functions with CUDA acceleration   | 19 |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/openacc/boundaries.c</code><br>Implementation of boundary condition functions with OpenMP threading      | 10 |
| <code>/home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/openacc/discretization.c</code><br>Implementation of boundary condition functions with OpenACC threading | 13 |

## 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:

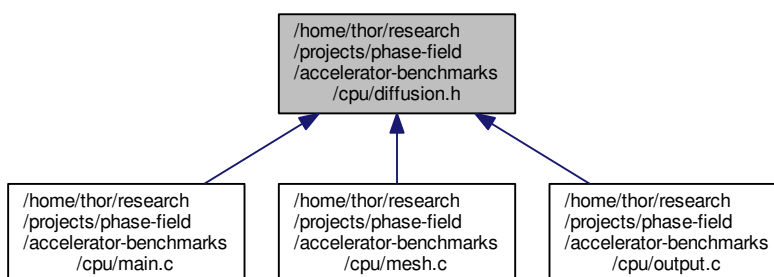
- `/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/tbb/discretization.cpp`

## 4 File Documentation

4.1 `/home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/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** ()

#### 4.1.1 Detailed Description

Declaration of diffusion equation function prototypes for CPU benchmarks.

Definition in file [diffusion.h](#).

## 4.2 /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/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** ()

#### 4.2.1 Detailed Description

Declaration of diffusion equation function prototypes for CPU benchmarks.

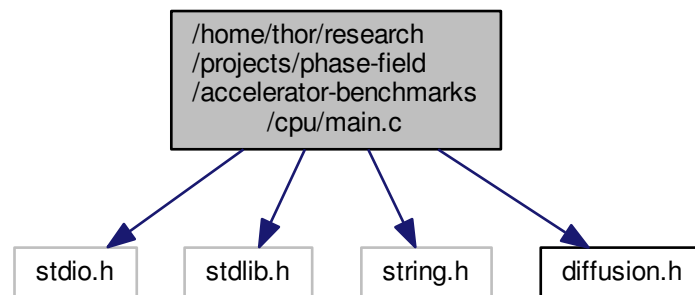
Definition in file [diffusion.h](#).

### 4.3 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/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 main.c:



#### Functions

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

#### 4.3.1 Detailed Description

Implementation of semi-infinite diffusion equation.

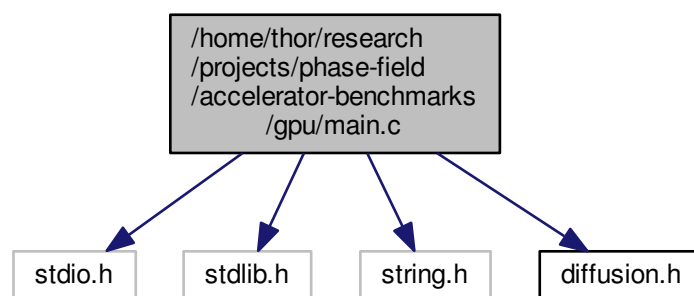
Definition in file [main.c](#).

### 4.4 /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/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 main.c:



## Functions

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

### 4.4.1 Detailed Description

Implementation of semi-infinite diffusion equation.

Definition in file [main.c](#).

## 4.5 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/mesh.c File Reference

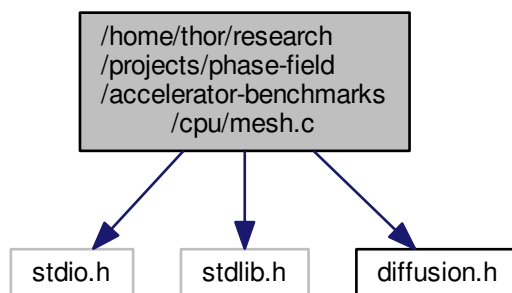
Implemenatation of mesh handling functions.

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

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

```
#include "diffusion.h"
```

Include dependency graph for 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)

### 4.5.1 Detailed Description

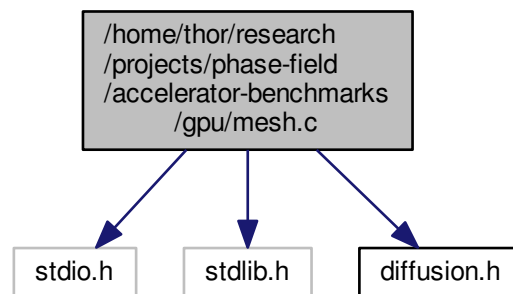
Implementation of mesh handling functions.

Definition in file [mesh.c](#).

## 4.6 /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/mesh.c File Reference

Implementation of mesh handling functions.

```
#include <stdio.h>
#include <stdlib.h>
#include "diffusion.h"
Include dependency graph for 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)

### 4.6.1 Detailed Description

Implementation of mesh handling functions.

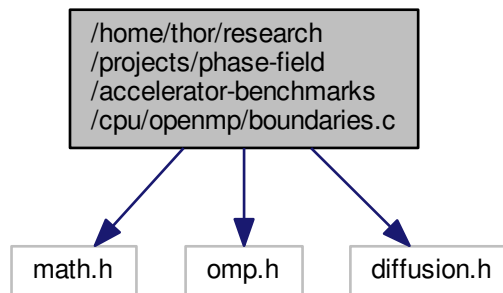
Definition in file [mesh.c](#).

#### 4.7 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/openmp/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 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])

##### 4.7.1 Detailed Description

Implementation of boundary condition functions with OpenMP threading.

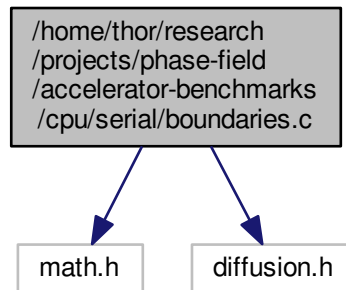
Definition in file [boundaries.c](#).

#### 4.8 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/serial/boundaries.c File Reference

Implementation of boundary condition functions without threading.

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

Include dependency graph for 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])

### 4.8.1 Detailed Description

Implementation of boundary condition functions without threading.

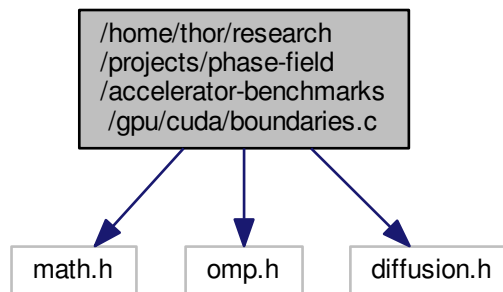
Definition in file [boundaries.c](#).

## 4.9 /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/cuda/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 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])

### 4.9.1 Detailed Description

Implementation of boundary condition functions with OpenMP threading.

Definition in file [boundaries.c](#).

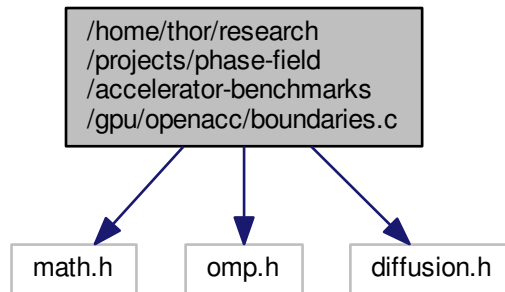
## 4.10 /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/openacc/boundaries.c File Reference

Implementation of boundary condition functions with OpenMP threading.

```

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

Include dependency graph for 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])

### 4.10.1 Detailed Description

Implementation of boundary condition functions with OpenMP threading.

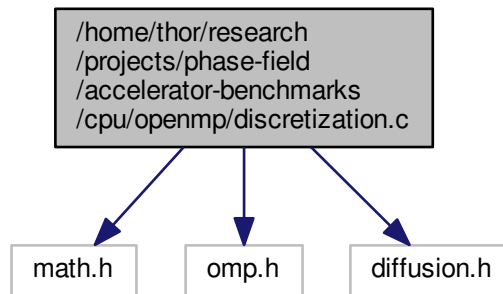
Definition in file [boundaries.c](#).

## 4.11 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/openmp/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 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)

### 4.11.1 Detailed Description

Implementation of boundary condition functions with OpenMP threading.

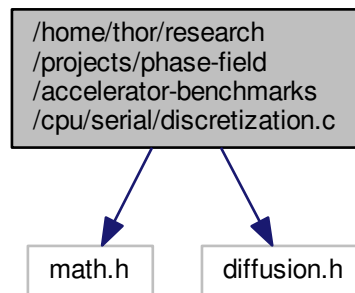
Definition in file [discretization.c](#).

## 4.12 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/serial/discretization.c File Reference

Implementation of boundary condition functions without threading.

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

Include dependency graph for 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)

### 4.12.1 Detailed Description

Implementation of boundary condition functions without threading.

Definition in file [discretization.c](#).

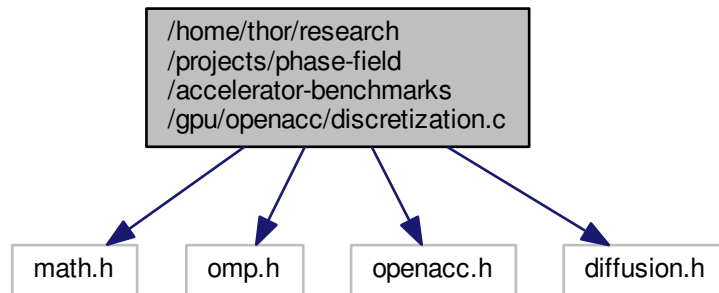
## 4.13 /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/openacc/discretization.c File Reference

Implementation of boundary condition functions with OpenACC threading.

```

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

Include dependency graph for 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)

### 4.13.1 Detailed Description

Implementation of boundary condition functions with OpenACC threading.

Definition in file [discretization.c](#).

## 4.14 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/output.c File Reference

Implementation of file output functions.

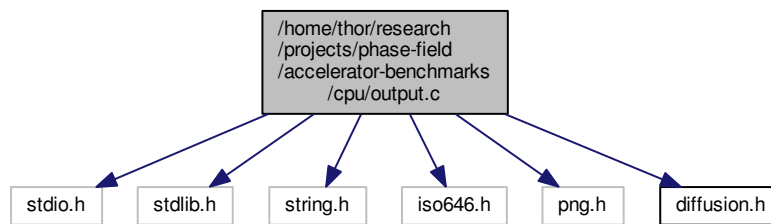
```

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

```



Include dependency graph for 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)

### 4.14.1 Detailed Description

Implementation of file output functions.

Definition in file [output.c](#).

## 4.15 /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/output.c File Reference

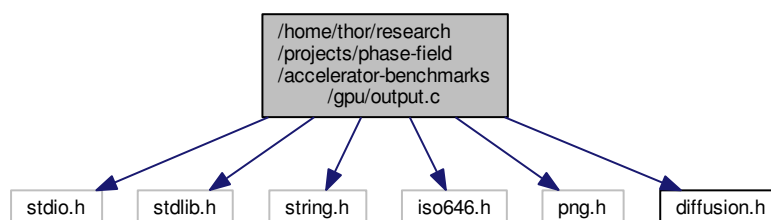
Implementation of file output functions.

```

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

```

Include dependency graph for 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)

#### 4.15.1 Detailed Description

Implementation of file output functions.

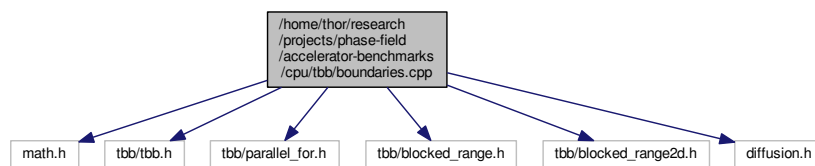
Definition in file [output.c](#).

### 4.16 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/tbb/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])

#### 4.16.1 Detailed Description

Implementation of boundary condition functions with TBB threading.

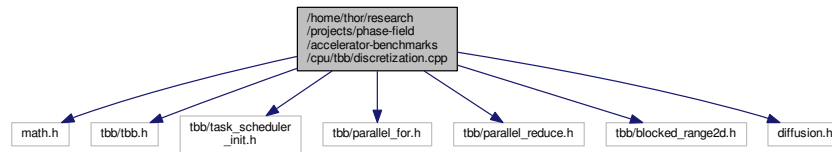
Definition in file [boundaries.cpp](#).

### 4.17 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/tbb/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)

### 4.17.1 Detailed Description

Implementation of boundary condition functions with TBB threading.

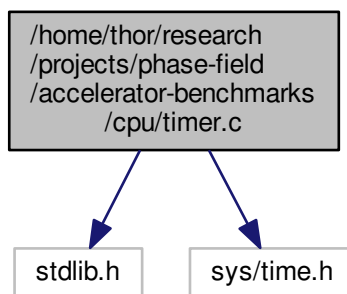
Definition in file [discretization.cpp](#).

## 4.18 /home/thor/research/projects/phase-field/accelerator-benchmarks/cpu/timer.c File Reference

High-resolution cross-platform machine time reader.

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

Include dependency graph for timer.c:



#### Functions

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

#### Variables

- struct timeval **timerStart**

#### 4.18.1 Detailed Description

High-resolution cross-platform machine time reader.

#### Author

NVIDIA

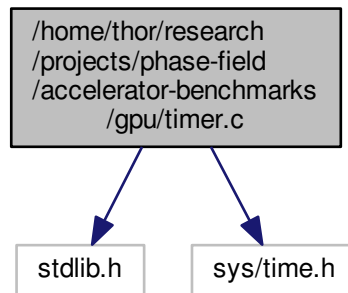
Definition in file [timer.c](#).

### 4.19 /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/timer.c File Reference

High-resolution cross-platform machine time reader.

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

Include dependency graph for 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

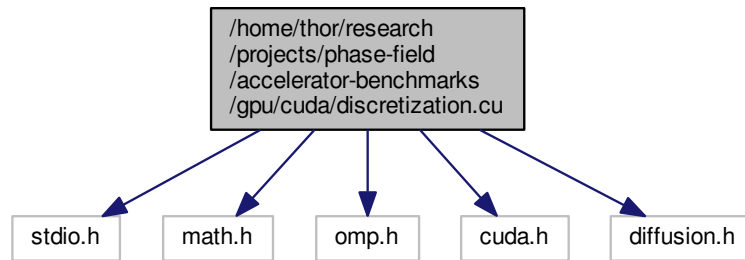
Definition in file [timer.c](#).

## 4.20 /home/thor/research/projects/phase-field/accelerator-benchmarks/gpu/cuda/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]`

#### 4.20.1 Detailed Description

Implementation of boundary condition functions with CUDA acceleration.

Definition in file [discretization.cu](#).