WARP v0.1 Source Reference

Generated by Doxygen 1.8.14

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 ace Namespace Reference

Classes

- class AceTable
- class DosimetryTable
- class ElectronTable
- class EnergyDistribution
- class Library
- class NeutronDiscreteTable
- class NeutronMGTable
- class NeutronTable
- class PhotoatomicMGTable
- class PhotoatomicTable
- class PhotonuclearTable
- class Reaction
- class SabTable

Functions

- def fromstring_split (s, sep=None, dtype=float)
- def fromstring_token (s, sep=" ", bint, inplace=False, int, maxsize=-1)
- def ascii_to_binary (ascii_file, binary_file)

Variables

- NP_LE_V15
- dictionary table_types

5.1.1 Detailed Description

```
This module is for reading ACE-format cross sections. ACE stands for "A
Compact ENDF" format and originated from work on MCNP_. It is used in a number
of other Monte Carlo particle transport codes.

ACE-format cross sections are typically generated from ENDF_ files through a
cross section processing program like NJOY_. The ENDF data consists of tabulated
thermal data, ENDF/B resonance parameters, distribution parameters in the
unresolved resonance region, and tabulated data in the fast region. After the
ENDF data has been reconstructed and Doppler-broadened, the ACER module
generates ACE-format cross sections.

.. _MCNP: https://laws.lanl.gov/vhosts/mcnp.lanl.gov/
.. _NJOY: http://t2.lanl.gov/codes.shtml
.. _ENDF: http://www.nndc.bnl.gov/endf

.. moduleauthor:: Paul Romano <paul.k.romano@gmail.com>, Anthony Scopatz <scopatz@gmail.com>
```

5.1.2 Function Documentation

5.1.2.1 ascii_to_binary()

```
def ace.ascii_to_binary (
              ascii_file,
              binary_file )
```

```
Convert an ACE file in ASCII format (type 1) to binary format (type 2).

Parameters
----------
ascii_file : str
    Filename of ASCII ACE file
binary_file : str
    Filename of binary ACE file to be written
```

5.1.2.2 fromstring_split()

```
def ace.fromstring_split (
              s,
              sep = None,
              dtype = float )
```

```
A replacement for numpy.fromstring() using the Python str.split()
and np.array().

Parameters
----------
s : str
    String of data.
sep : str or None
    String of separator characters, has the same meaning as in
    str.split().
```

```
dtype : np.dtype
    Numpy dtype to cast elements enough.

Returns
-------
data : ndarray, 1d
    Will always return a 1d array of dtype.  You must reshape to the
    appropriate shape.

See Also
--------
fromstring_token : May faster depending on the data.
```

5.1.2.3 fromstring_token()

```
def ace.fromstring_token (
              s,
              sep = " ",
              bint,
              inplace = False,
              int,
              maxsize = -1 )
```

```
A replacement for numpy.fromstring() using the C standard
library atof() and strtok() functions.

Parameters
----------
s : str
    String of data.
sep : str
    String of separator characters.  Unlike numpy.fromstring(),
    all characters are separated on independently.
inplace : bool
    Whether s should tokenized in-place or whether a copy should
    be made.  If done in-place, the first instance of sep between
    any tokens will replaced with the NULL character.
maxsize : int
    Specifies the size of the array to pre-allocate.  If negative,
    this will be set to the maximum possible number of elements,
    ie len(s)/2 + 1.

Returns
-------
data : ndarray, 1d, float64
    Will always return a 1d float64 array.  You must cast and reshape
    to the appropriate type and shape.

See Also
--------
fromstring_split : May faster depending on the data.
```

5.1.3 Variable Documentation

5.1.3.1 NP_LE_V15

```
ace.NP_LE_V15
```

5.1.3.2 table_types

```
dictionary ace.table_types
```

Initial value:

```
=  {
     "c": NeutronTable,
     "t": SabTable,
     "y": DosimetryTable,
     "d": NeutronDiscreteTable,
     "p": PhotoatomicTable,
     "m": NeutronMGTable,
     "g": PhotoatomicMGTable,
     "e": ElectronTable,
      "u": PhotonuclearTable}
```

5.2 unionize Namespace Reference

Classes

- class cross_section_data

  handles cross section data

Variables

- bool ace_available = True

5.2.1 Variable Documentation

5.2.1.1 ace_available

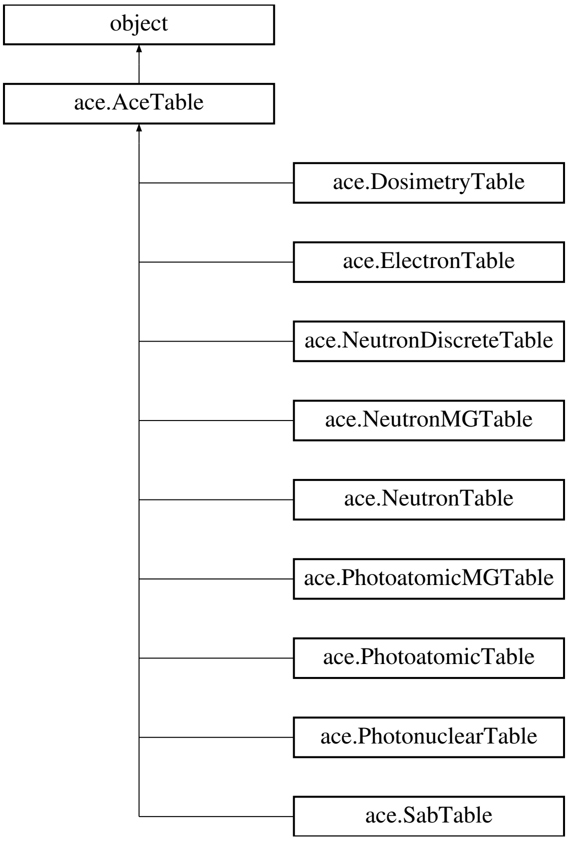
```
bool unionize.ace_available = True
```

Chapter 6

Class Documentation

6.1 ace.AceTable Class Reference

Inheritance diagram for ace.AceTable:



Public Member Functions

- def __init__ (self, name, awr, temp)

Public Attributes

- name
- awr
- temp

Private Member Functions

- def _read_all (self)

6.1.1 Detailed Description

```
Abstract superclass of all other classes for cross section tables.
```

6.1.2 Constructor & Destructor Documentation

6.1.2.1 __init__()

```
def ace.AceTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.1.3 Member Function Documentation

6.1.3.1 _read_all()

```
def ace.AceTable._read_all (
              self )  [private]
```

6.1.4 Member Data Documentation

6.1.4.1 awr

```
ace.AceTable.awr
```

6.1.4.2 name

```
ace.AceTable.name
```

6.1.4.3 temp

```
ace.AceTable.temp
```

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

- ace.pyx

6.2 cross_section_data Struct Reference

structure that holds the topmost level of cross section data

```
#include <datadef.h>
```

Public Attributes

- unsigned n_isotopes
- unsigned energy_grid_len
- unsigned total_reaction_channels
- unsigned ∗ rxn_numbers
- unsigned ∗ rxn_numbers_total
- float ∗ energy_grid
- float ∗ Q
- float ∗ xs
- float ∗ awr
- float ∗ temp
- dist_container ∗ dist_scatter
- dist_container ∗ dist_energy

6.2.1 Detailed Description

structure that holds the topmost level of cross section data

contains pointers and parameters to compute any reaction in this requested isotope set - number of isotopes, length of main energy grid, total number of reactions channels, reaction number vector, total reaction channels for each isotope, unionized energy grid vector, reaction Q values, cross section data, isotope atomic weight ratios, isotope temperatures, energy and scattering data distributions

6.2.2 Member Data Documentation

6.2.2.1 awr

```
float* cross_section_data::awr
```

isotope atomic weight ratio (AWR) list

6.2.2.2 dist_energy

```
dist_container* cross_section_data::dist_energy
```

energy distribution data redirection matrix

6.2.2.3 dist_scatter

```
dist_container* cross_section_data::dist_scatter
```

scattering distribution data redirection matrix

6.2.2.4 energy_grid

```
float* cross_section_data::energy_grid
```

unionized energy grid vector

6.2.2.5 energy_grid_len

```
unsigned cross_section_data::energy_grid_len
```

length of main energy grid

6.2.2.6 n_isotopes

```
unsigned cross_section_data::n_isotopes
```

number of isotopes

6.2.2.7 Q

```
float* cross_section_data::Q
```

reaction Q values

6.2.2.8 rxn_numbers

```
unsigned* cross_section_data::rxn_numbers
```

reaction number vector

6.2.2.9 rxn_numbers_total

```
unsigned* cross_section_data::rxn_numbers_total
```

total reaction channels for each isotope

6.2.2.10 temp

```
float* cross_section_data::temp
```

isotope temperature list (MeV)

6.2.2.11 total_reaction_channels

```
unsigned cross_section_data::total_reaction_channels
```

total number of reactions channels

6.2.2.12 xs

```
float* cross_section_data::xs
```

cross section data matrix

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

- datadef.h

6.3 unionize.cross_section_data Class Reference

handles cross section data

Public Member Functions

- def __init__ (self)

    cross section data class to hold it all together with the functions necessary for WARP

Public Attributes

- num_isotopes

  number of isotopes
- isotope_list

  isotope list
- datapath

  data path
- tables

  cross section tables
- libraries

  cross section libraries
- awr

  AWR array.
- temp

  temp array
- Q

  Q-value array.
- num_main_E

  main energy
- reaction_numbers

  reaction numbers array
- reaction_numbers_total

  total reaction numbers array
- num_reactions

  number of reactions
- MT_E_grid

  MT energy grid.
- MT_array

  MT number array.
- last_loaded

  Last valid table loaded.
- isotropic_tol

  isotropic tolerance
- xsdirstring

Private Member Functions

- def _init_from_string (self, this_string)

  initializes material from isotope list string
- def _add_isotope (self, isotope)

  appends the input isotope the the input material's list of isotopes
- def _read_tables (self, datapath_in)

  reads in cross section tables
- def _resolve_library (self, tope)

  finds path the isotope files
- def _unionize (self)

  unionization function
- def _insert_reactions (self)

  insert reactions function

- def _allocate_arrays (self)

  array allocation function
- def _interpolate (self)

  interpolation function
- def _get_MT_numbers_pointer (self)

  gets pointer to MT numbers
- def _get_awr_pointer (self)

  gets pointer to AWR values
- def _get_temp_pointer (self)

  gets pointer to temperature values
- def _get_Q_pointer (self)

  gets pointer to Q-values
- def _get_MT_array_pointer (self)

  gets pointer to MT numbers
- def _get_main_Egrid_pointer (self)

  gets pointer to main energy grid
- def _get_length_numbers_pointer (self)

  creates array of size number of isotopes + main energy grid + number of reactions
- def _get_MT_numbers_total_pointer (self)

  gets pointer to total MT numbers
- def _print_isotopes (self)

  prints list of isotopes in a material
- def _get_scatter_data (self, row, col)

  gets table of scattering data
- def _get_energy_data (self, row, col)

  gets table of energy data

6.3.1 Detailed Description

handles cross section data

6.3.2 Constructor & Destructor Documentation

6.3.2.1 __init__()

```
def unionize.cross_section_data.__init__ (
              self )
```

cross section data class to hold it all together with the functions necessary for WARP

initializes number of isotopes to zero; isotope list as an empty array; temperature extension as '.03c'; tables, libraries, AWR list, and Q as empty arrays; main energy as zero; reaction numbers and total reaction numbers as empty arrays; number of reactions to zero. sets the MT energy grid # and array as empty.

Parameters

in	*self*	- this cross_section_data object

6.3.3 Member Function Documentation

6.3.3.1 _add_isotope()

```
def unionize.cross_section_data._add_isotope (
              self,
              isotope )  [private]
```

appends the input isotope the the input material's list of isotopes

Parameters

in	*self*	- this cross_section_data object
in	*isotope*	- isotope to be appended

6.3.3.2 _allocate_arrays()

```
def unionize.cross_section_data._allocate_arrays (
              self )  [private]
```

array allocation function

allocates a 2D array of size number of all reactions x number of energy points

Parameters

in	*self*	- this cross_section_data object

6.3.3.3 _get_awr_pointer()

```
def unionize.cross_section_data._get_awr_pointer (
              self )  [private]
```

gets pointer to AWR values

Parameters

in	*self*	- this cross_section_data object

Returns

AWR_array - array of AWR values

6.3.3.4 _get_energy_data()

```
def unionize.cross_section_data._get_energy_data (
              self,
              row,
              col )  [private]
```

gets table of energy data

table returned in form of [nextDex, length, mu, cdf]

Parameters

in	*self*	- this cross_section_data object
in	*row*	- point in energy grid
in	*col*	- MT number

6.3.3.5 _get_length_numbers_pointer()

```
def unionize.cross_section_data._get_length_numbers_pointer (
              self )  [private]
```

creates array of size number of isotopes + main energy grid + number of reactions

Parameters

in	*self*	- this cross_section_data object

Returns

lengths - lengths array

6.3.3.6 _get_main_Egrid_pointer()

```
def unionize.cross_section_data._get_main_Egrid_pointer (
              self )  [private]
```

gets pointer to main energy grid

Parameters

in	*self*	- this cross_section_data object

Returns

E_grid - array of energy grid points

6.3.3.7 _get_MT_array_pointer()

```
def unionize.cross_section_data._get_MT_array_pointer (
              self )  [private]
```

gets pointer to MT numbers

Parameters

in	*self*	- this cross_section_data object

Returns

MT_array - array of MT numbers

6.3.3.8 _get_MT_numbers_pointer()

```
def unionize.cross_section_data._get_MT_numbers_pointer (
              self )  [private]
```

gets pointer to MT numbers

Parameters

in	*self*	- this cross_section_data object

Returns

MT_num_array - array of MT numbers

6.3.3.9 _get_MT_numbers_total_pointer()

```
def unionize.cross_section_data._get_MT_numbers_total_pointer (
              self )  [private]
```

gets pointer to total MT numbers

Parameters

in	*self*	- this cross_section_data object

Returns

numbers - array of total MT numbers

6.3.3.10 _get_Q_pointer()

```
def unionize.cross_section_data._get_Q_pointer (
              self )  [private]
```

gets pointer to Q-values

Parameters

in	*self*	- this cross_section_data object

Returns

Q_array - array of Q-values

6.3.3.11 _get_scatter_data()

```
def unionize.cross_section_data._get_scatter_data (
              self,
              row,
              col )  [private]
```

gets table of scattering data

if scattering data exists, table returned in form of [nextDex, length, mu, cdf]

Parameters

in	*self*	- this cross_section_data object
in	*row*	- point in energy grid
in	*col*	- MT number

6.3.3.12 _get_temp_pointer()

```
def unionize.cross_section_data._get_temp_pointer (
              self )  [private]
```

gets pointer to temperature values

Parameters

in	*self*	- this cross_section_data object

Returns

temp_array - array of temperature values

6.3.3.13 _init_from_string()

```
def unionize.cross_section_data._init_from_string (
              self,
              this_string )  [private]
```

initializes material from isotope list string

Parameters

in	*self*	- this cross_section_data object
in	*this_string*	- comma-separated isotope list

6.3.3.14 _insert_reactions()

```
def unionize.cross_section_data._insert_reactions (
              self )  [private]
```

insert reactions function

appends ones to the front, appends the isotope's AWR to the table, appends the isotope's total reaction numbers to the table. appends all reaction numbers to the reaction list.

Parameters

in	*self*	- this cross_section_data object

6.3.3.15 _interpolate()

```
def unionize.cross_section_data._interpolate (
              self ) [private]
```

interpolation function

linearly interpolates the cross sections for each isotope in a material

Parameters

in	*self*	- this cross_section_data object

6.3.3.16 _print_isotopes()

```
def unionize.cross_section_data._print_isotopes (
              self ) [private]
```

prints list of isotopes in a material

Parameters

in	*self*	- material for which to print isotope list

6.3.3.17 _read_tables()

```
def unionize.cross_section_data._read_tables (
              self,
              datapath_in ) [private]
```

reads in cross section tables

for each isotope in the material, the acefile is appended to the library list, then all of the libraries are read in. the material's number of isotopes is set to how many libraries were retrieved.

Parameters

in	*self*	- this cross_section_data object
in	*datapath↩ in*	- the input datapath for the xs data

6.3.3.18 _resolve_library()

```
def unionize.cross_section_data._resolve_library (
              self,
              tope )  [private]
```

finds path the isotope files

finds the relative path from DATAPATH to a file that contains the specified isotope cross sections. It reads the xsdir file to do this.

Parameters

in	*self*	- this cross_section_data object
in	*tope*	- isotope to seach for

6.3.3.19 _unionize()

```
def unionize.cross_section_data._unionize (
              self )  [private]
```

unionization function

unionizes MT energy grid.

Parameters

in	*self*	- this cross_section_data object

6.3.4 Member Data Documentation

6.3.4.1 awr

```
unionize.cross_section_data.awr
```

AWR array.

6.3.4.2 datapath

```
unionize.cross_section_data.datapath
```

data path

6.3.4.3 isotope_list

```
unionize.cross_section_data.isotope_list
```

isotope list

6.3.4.4 isotropic_tol

```
unionize.cross_section_data.isotropic_tol
```

isotropic tolerance

6.3.4.5 last_loaded

```
unionize.cross_section_data.last_loaded
```

Last valid table loaded.

6.3.4.6 libraries

```
unionize.cross_section_data.libraries
```

cross section libraries

6.3.4.7 MT_array

```
unionize.cross_section_data.MT_array
```

MT number array.

6.3.4.8 MT_E_grid

```
unionize.cross_section_data.MT_E_grid
```

MT energy grid.

6.3.4.9 num_isotopes

```
unionize.cross_section_data.num_isotopes
```

number of isotopes

6.3.4.10 num_main_E

```
unionize.cross_section_data.num_main_E
```

main energy

6.3.4.11 num_reactions

```
unionize.cross_section_data.num_reactions
```

number of reactions

6.3.4.12 Q

```
unionize.cross_section_data.Q
```

Q-value array.

6.3.4.13 reaction_numbers

```
unionize.cross_section_data.reaction_numbers
```

reaction numbers array

6.3.4.14 reaction_numbers_total

```
unionize.cross_section_data.reaction_numbers_total
```

total reaction numbers array

6.3.4.15 tables

```
unionize.cross_section_data.tables
```

cross section tables

6.3.4.16 temp

```
unionize.cross_section_data.temp
```

temp array

6.3.4.17 xsdirstring

```
unionize.cross_section_data.xsdirstring
```

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

- unionize.py

6.4 dist_container Struct Reference

container for pointers that map the nearest distributions to the energy grid point where it resides

```
#include <datadef.h>
```

Public Attributes

- dist_data ∗ lower
- dist_data ∗ upper

6.4.1 Detailed Description

container for pointers that map the nearest distributions to the energy grid point where it resides

pointers to the nearest distribution lower and greater in energy

6.4.2 Member Data Documentation

6.4.2.1 lower

```
dist_data* dist_container::lower
```

pointer to distribution data of grid point below current energy

6.4.2.2 upper

```
dist_data* dist_container::upper
```

pointer to distribution data of grid point above current energy

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

- datadef.h

6.5 dist_data Struct Reference

contains information that defines an ENDF cross section distribution

```
#include <datadef.h>
```

Public Attributes

- float erg
- unsigned len
- unsigned law
- unsigned intt
- float * var
- float * pdf
- float * cdf

6.5.1 Detailed Description

contains information that defines an ENDF cross section distribution

ernergy, length, law, interpolation type, variable/pdf/cdf arrays

6.5.2 Member Data Documentation

6.5.2.1 cdf

```
float* dist_data::cdf
```

cumulative density function array

6.5.2.2 erg

```
float dist_data::erg
```

energy point of this distribution

6.5.2.3 intt

```
unsigned dist_data::intt
```

interpolation flag of this distribution

6.5.2.4 law

```
unsigned dist_data::law
```

sampling law of this distribution

6.5.2.5 len

```
unsigned dist_data::len
```

length of the arrays in this distribution

6.5.2.6 pdf

```
float* dist_data::pdf
```

probability density function array

6.5.2.7 var

```
float* dist_data::var
```

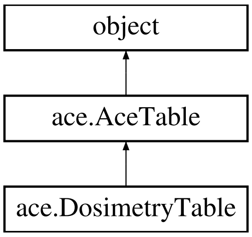
independent variable (mu, E, etc.)

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

- datadef.h

6.6 ace.DosimetryTable Class Reference

Inheritance diagram for ace.DosimetryTable:



Public Member Functions

- def __init__ (self, name, awr, temp)
- def __repr__ (self)

Additional Inherited Members

6.6.1 Constructor & Destructor Documentation

6.6.1.1 __init__()

```
def ace.DosimetryTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.6.2 Member Function Documentation

6.6.2.1 __repr__()

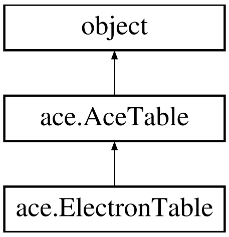
```
def ace.DosimetryTable.__repr__ (
              self )
```

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

- ace.pyx

6.7 ace.ElectronTable Class Reference

Inheritance diagram for ace.ElectronTable:



Public Member Functions

- def __init__ (self, name, awr, temp)
- def __repr__ (self)

Additional Inherited Members

6.7.1 Constructor & Destructor Documentation

6.7.1.1 __init__()

```
def ace.ElectronTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.7.2 Member Function Documentation

6.7.2.1 __repr__()

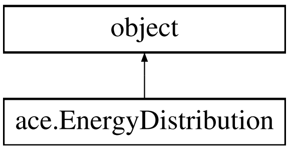
```
def ace.ElectronTable.__repr__ (
              self )
```

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

- ace.pyx

6.8 ace.EnergyDistribution Class Reference

Inheritance diagram for ace.EnergyDistribution:



Public Member Functions

- def __init__ (self)

6.8.1 Constructor & Destructor Documentation

6.8.1.1 __init__()

```
def ace.EnergyDistribution.__init__ (
              self )
```

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

- ace.pyx

6.9 geom_data Struct Reference

contains parameters of a wgeometry

```
#include <datadef.h>
```

Public Attributes

- float min [3]
- float max [3]
- float loc [3]
- int cellnum
- int talnum
- int matnum
- int is_fissile

6.9.1 Detailed Description

contains parameters of a wgeometry

extrema arrays, location array, cell and material numbers, tally number, fissile flag

6.9.2 Member Data Documentation

6.9.2.1 cellnum

```
int geom_data::cellnum
```

cell number

6.9.2.2 is_fissile

```
int geom_data::is_fissile
```

fissile flag

6.9.2.3 loc

```
float geom_data::loc[3]
```

array of coordinate (x,y,z) locations

6.9.2.4 matnum

```
int geom_data::matnum
```

material number

6.9.2.5 max

```
float geom_data::max[3]
```

array of coordinate (x,y,z) maxima

6.9.2.6 min

```
float geom_data::min[3]
```

array of coordinate (x,y,z) minima

6.9.2.7 talnum

```
int geom_data::talnum
```

tally index

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

- datadef.h

6.10 intersection_point Struct Reference

contains information pertinent to an intersection point, used in OptiX

```
#include <datadef.h>
```

Public Attributes

- float x
- float y
- float z
- float surf_dist
- int cell
- int mat
- int fiss
- float norm [3]
- int sense
- int tally_index

6.10.1 Detailed Description

contains information pertinent to an intersection point, used in OptiX

3D cartesian coordinates of intersection point, distance to nearest surface, first cell potentially hit, material of hit cell, fissile flag of hit cell, normal vector of hit surface, hit cell sense value, index of tally for hit cell

6.10.2 Member Data Documentation

6.10.2.1 cell

```
int intersection_point::cell
```

most recently hit cell number

6.10.2.2 fiss

```
int intersection_point::fiss
```

most recently hit fissile flag

6.10.2.3 mat

```
int intersection_point::mat
```

most recently hit material number

6.10.2.4 norm

```
float intersection_point::norm[3]
```

most recently hit normal

6.10.2.5 sense

```
int intersection_point::sense
```

most recently hit cell sense

6.10.2.6 surf_dist

```
float intersection_point::surf_dist
```

distance to nearest surface

6.10.2.7 tally_index

```
int intersection_point::tally_index
```

tally index of most recently hit cell

6.10.2.8 x

```
float intersection_point::x
```

x-coordinate

6.10.2.9 y

```
float intersection_point::y
```

y-coordinate

6.10.2.10 z

```
float intersection_point::z
```

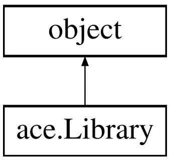
z-coordinate

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

- datadef.h

6.11 ace.Library Class Reference

Inheritance diagram for ace.Library:



Public Member Functions

- def __init__ (self, filename)
- def read (self, table_names=None)
- def find_table (self, name)
- def __del__ (self)

Public Attributes

- f
- binary
- verbose
- tables

Private Member Functions

- def _read_binary (self, table_names, recl_length=4096, entries=512)
- def _read_ascii (self, table_names)

6.11.1 Detailed Description

```
A Library objects represents an ACE-formatted file which may contain
multiple tables with data.

Parameters
----------
filename : str
    Path of the ACE library file to load.

Attributes
----------
binary : bool
    Identifies Whether the library is in binary format or not
tables : dict
    Dictionary whose keys are the names of the ACE tables and whose
    values are the instances of subclasses of AceTable (e.g. NeutronTable)
verbose : bool
    Determines whether output is printed to the stdout when reading a
    Library
```

6.11.2 Constructor & Destructor Documentation

6.11.2.1 __init__()

```
def ace.Library.__init__ (
              self,
              filename )
```

6.11.2.2 __del__()

```
def ace.Library.__del__ (
              self )
```

6.11.3 Member Function Documentation

6.11.3.1 _read_ascii()

```
def ace.Library._read_ascii (
              self,
              table_names )  [private]
```

6.11.3.2 _read_binary()

```
def ace.Library._read_binary (
              self,
              table_names,
              recl_length = 4096,
              entries = 512 )  [private]
```

6.11.3.3 find_table()

```
def ace.Library.find_table (
              self,
              name )
```

```
find_table(name)

Returns a cross-section table with a given name.

Parameters
----------
name : str
    Name of the cross-section table, e.g. 92235.70c
```

6.11.3.4 read()

```
def ace.Library.read (
              self,
              table_names = None )
```

```
read(table_names=None)

Read through and parse the ACE-format library.

Parameters
----------
table_names : None, str, or iterable, optional
    Tables from the file to read in.  If None, reads in all of the
    tables. If str, reads in only the single table of a matching name.
```

6.11.4 Member Data Documentation

6.11.4.1 binary

```
ace.Library.binary
```

6.11.4.2 f

```
ace.Library.f
```

6.11.4.3 tables

```
ace.Library.tables
```

6.11.4.4 verbose

```
ace.Library.verbose
```

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

- ace.pyx

6.12 material_def Struct Reference

contains information that defines a material

```
#include <datadef.h>
```

Public Attributes

- unsigned id
- unsigned matnum
- unsigned is_fissile
- unsigned num_isotopes
- float density
- std::vector< std::string > isotopes
- float ∗ fractions

6.12.1 Detailed Description

contains information that defines a material

material index, label number, fissile flag, number of isotopes, density, isotope list, isotope atom fraction list

6.12.2 Member Data Documentation

6.12.2.1 density

```
float material_def::density
```

density [g/cc]

6.12.2.2 fractions

```
float∗ material_def::fractions
```

isotope atom fractions

6.12.2.3 id

```
unsigned material_def::id
```

material index

6.12.2.4 is_fissile

```
unsigned material_def::is_fissile
```

fissile flag

6.12.2.5 isotopes

```
std::vector<std::string> material_def::isotopes
```

isotope list

6.12.2.6 matnum

```
unsigned material_def::matnum
```

material label number

6.12.2.7 num_isotopes

```
unsigned material_def::num_isotopes
```

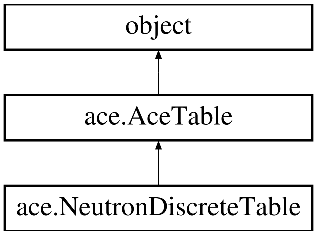
number of isotopes

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

- datadef.h

6.13 ace.NeutronDiscreteTable Class Reference

Inheritance diagram for ace.NeutronDiscreteTable:



Public Member Functions

- def __init__ (self, name, awr, temp)
- def __repr__ (self)

Additional Inherited Members

6.13.1 Constructor & Destructor Documentation

6.13.1.1 __init__()

```
def ace.NeutronDiscreteTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.13.2 Member Function Documentation

6.13.2.1 __repr__()

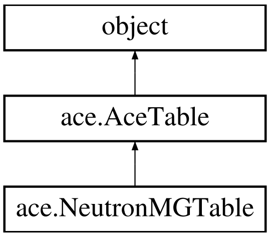
```
def ace.NeutronDiscreteTable.__repr__ (
              self )
```

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

- ace.pyx

6.14 ace.NeutronMGTable Class Reference

Inheritance diagram for ace.NeutronMGTable:



Public Member Functions

- def __init__ (self, name, awr, temp)
- def __repr__ (self)

Additional Inherited Members

6.14.1 Constructor & Destructor Documentation

6.14.1.1 __init__()

```
def ace.NeutronMGTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.14.2 Member Function Documentation

6.14.2.1 __repr__()

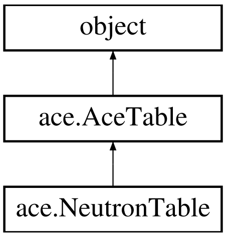
```
def ace.NeutronMGTable.__repr__ (
              self )
```

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

- ace.pyx

6.15 ace.NeutronTable Class Reference

Inheritance diagram for ace.NeutronTable:



Public Member Functions

- def __init__ (self, name, awr, temp)
- def __repr__ (self)
- def find_reaction (self, mt)
- def __iter__ (self)

Public Attributes

- reactions
- photon_reactions
- heating
- nu_t_type
- nu_t_interp_NBT
- nu_t_interp_INT
- nu_t_energy
- nu_t_value
- nu_p_type
- nu_p_interp_NBT
- nu_p_interp_INT
- nu_p_energy
- nu_p_value
- nu_d_interp_NBT
- nu_d_interp_INT
- nu_d_energy
- nu_d_value
- nu_d_precursor_const
- nu_d_precursor_energy
- nu_d_precursor_prob
- nu_d_precursor_interp_NBT
- nu_d_precursor_interp_INT
- nu_d_energy_dist
- e_dist_energy_outNE
- e_dist_LP
- e_dist_EG
- sigma_photon
- sigma
- a_dist_energy_in
- a_dist_mu_out
- MT_for_photon_yield
- IE_fission
- sigma_f
- urr_energy
- urr_table

Private Member Functions

- def _read_all (self)
- def _read_cross_sections (self)
- def _read_nu (self)
- def _read_angular_distributions (self)
- def _read_energy_distributions (self)
- def _get_energy_distribution (self, location_start, delayed_n=False)
- def _read_gpd (self)
- def _read_mtrp (self)
- def _read_lsigp (self)
- def _read_sigp (self)
- def _read_landp (self)
- def _read_andp (self)
- def _read_yp (self)
- def _read_fis (self)
- def _read_unr (self)

6.15.1 Detailed Description

```
A NeutronTable object contains continuous-energy neutron interaction data
read from an ACE-formatted Type I table. These objects are not normally
instantiated by the user but rather created when reading data using a
Library object and stored within the ``tables`` attribute of a Library
object.

Parameters
----------
name : str
    ZAID identifier of the table, e.g. ’92235.70c’.
awr : float
    Atomic mass ratio of the target nuclide.
temp : float
    Temperature of the target nuclide in eV.

Attributes
----------
awr : float
    Atomic mass ratio of the target nuclide.

energy : list of floats
    The energy values (MeV) at which reaction cross-sections are tabulated.

name : str
    ZAID identifier of the table, e.g. 92235.70c.

nu_p_energy : list of floats
    Energies in MeV at which the number of prompt neutrons emitted per
    fission is tabulated.

nu_p_type : str
    Indicates how number of prompt neutrons emitted per fission is
    stored. Can be either "polynomial" or "tabular".

nu_p_value : list of floats
    The number of prompt neutrons emitted per fission, if data is stored in
    "tabular" form, or the polynomial coefficients for the "polynomial"
    form.

nu_t_energy : list of floats
    Energies in MeV at which the total number of neutrons emitted per
    fission is tabulated.

nu_t_type : str
    Indicates how total number of neutrons emitted per fission is
    stored. Can be either "polynomial" or "tabular".

nu_t_value : list of floats
    The total number of neutrons emitted per fission, if data is stored in
    "tabular" form, or the polynomial coefficients for the "polynomial"
    form.

reactions : list of Reactions
    A list of Reaction instances containing the cross sections, secondary
    angle and energy distributions, and other associated data for each
    reaction for this nuclide.

sigma_a : list of floats
    The microscopic absorption cross section for each value on the energy
    grid.

sigma_t : list of floats
    The microscopic total cross section for each value on the energy grid.

temp : float
    Temperature of the target nuclide in eV.
```

6.15.2 Constructor & Destructor Documentation

6.15.2.1 __init__()

```
def ace.NeutronTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.15.3 Member Function Documentation

6.15.3.1 __iter__()

```
def ace.NeutronTable.__iter__ (
              self )
```

6.15.3.2 __repr__()

```
def ace.NeutronTable.__repr__ (
              self )
```

6.15.3.3 _get_energy_distribution()

```
def ace.NeutronTable._get_energy_distribution (
              self,
              location_start,
              delayed_n = False )  [private]
```

```
Returns an EnergyDistribution object from data read in starting at
location_start.
```

6.15.3.4 _read_all()

```
def ace.NeutronTable._read_all (
              self )  [private]
```

6.15.3.5 _read_andp()

```
def ace.NeutronTable._read_andp (
              self )  [private]
```

```
Find the angular distribution for each photon-producing reaction
MT.
```

6.15.3.6 _read_angular_distributions()

```
def ace.NeutronTable._read_angular_distributions (
              self )  [private]
```

```
Find the angular distribution for each reaction MT
```

6.15.3.7 _read_cross_sections()

```
def ace.NeutronTable._read_cross_sections (
              self )  [private]
```

```
Reads and parses the ESZ, MTR, LQR, TRY, LSIG, and SIG blocks. These
blocks contain the energy grid, all reaction cross sections, the total
cross section, average heating numbers, and a list of reactions with
their Q-values and multiplicites.
```

6.15.3.8 _read_energy_distributions()

```
def ace.NeutronTable._read_energy_distributions (
              self )  [private]
```

```
Determine the energy distribution for secondary neutrons for
each reaction MT
```

6.15.3.9 _read_fis()

```
def ace.NeutronTable._read_fis (
              self )  [private]
```

```
Read total fission cross-section data if present. Generally,
this table is not provided since it is redundant.
```

6.15.3.10 _read_gpd()

```
def ace.NeutronTable._read_gpd (
              self )  [private]
```

Read total photon production cross section.

6.15.3.11 _read_landp()

```
def ace.NeutronTable._read_landp (
              self )  [private]
```

Determine location of angular distribution for each photon-producing reaction MT.

6.15.3.12 _read_lsigp()

```
def ace.NeutronTable._read_lsigp (
              self )  [private]
```

Determine location of cross sections for each photon-producing reaction MT.

6.15.3.13 _read_mtrp()

```
def ace.NeutronTable._read_mtrp (
              self )  [private]
```

Get the list of reaction MTs for photon-producing reactions for this cross-section table. The MT values are somewhat arbitrary.

6.15.3.14 _read_nu()

```
def ace.NeutronTable._read_nu (
              self )  [private]
```

Read the NU block -- this contains information on the prompt
and delayed neutron precursor yields, decay constants, etc

6.15.3.15 _read_sigp()

```
def ace.NeutronTable._read_sigp (
              self )  [private]
```

Read cross-sections for each photon-producing reaction MT.

6.15.3.16 _read_unr()

```
def ace.NeutronTable._read_unr (
              self )  [private]
```

Read the unresolved resonance range probability tables if present.

6.15.3.17 _read_yp()

```
def ace.NeutronTable._read_yp (
              self )  [private]
```

Read list of reactions required as photon production yield multipliers.

6.15.3.18 find_reaction()

```
def ace.NeutronTable.find_reaction (
              self,
              mt )
```

6.15.4 Member Data Documentation

6.15.4.1 a_dist_energy_in

ace.NeutronTable.a_dist_energy_in

6.15.4.2 a_dist_mu_out

```
ace.NeutronTable.a_dist_mu_out
```

6.15.4.3 e_dist_EG

```
ace.NeutronTable.e_dist_EG
```

6.15.4.4 e_dist_energy_outNE

```
ace.NeutronTable.e_dist_energy_outNE
```

6.15.4.5 e_dist_LP

```
ace.NeutronTable.e_dist_LP
```

6.15.4.6 heating

```
ace.NeutronTable.heating
```

6.15.4.7 IE_fission

```
ace.NeutronTable.IE_fission
```

6.15.4.8 MT_for_photon_yield

```
ace.NeutronTable.MT_for_photon_yield
```

6.15.4.9 nu_d_energy

```
ace.NeutronTable.nu_d_energy
```

6.15.4.10 nu_d_energy_dist

```
ace.NeutronTable.nu_d_energy_dist
```

6.15.4.11 nu_d_interp_INT

```
ace.NeutronTable.nu_d_interp_INT
```

6.15.4.12 nu_d_interp_NBT

```
ace.NeutronTable.nu_d_interp_NBT
```

6.15.4.13 nu_d_precursor_const

```
ace.NeutronTable.nu_d_precursor_const
```

6.15.4.14 nu_d_precursor_energy

```
ace.NeutronTable.nu_d_precursor_energy
```

6.15.4.15 nu_d_precursor_interp_INT

```
ace.NeutronTable.nu_d_precursor_interp_INT
```

6.15.4.16 nu_d_precursor_interp_NBT

```
ace.NeutronTable.nu_d_precursor_interp_NBT
```

6.15.4.17 nu_d_precursor_prob

```
ace.NeutronTable.nu_d_precursor_prob
```

6.15.4.18 nu_d_value

```
ace.NeutronTable.nu_d_value
```

6.15.4.19 nu_p_energy

```
ace.NeutronTable.nu_p_energy
```

6.15.4.20 nu_p_interp_INT

```
ace.NeutronTable.nu_p_interp_INT
```

6.15.4.21 nu_p_interp_NBT

```
ace.NeutronTable.nu_p_interp_NBT
```

6.15.4.22 nu_p_type

```
ace.NeutronTable.nu_p_type
```

6.15.4.23 nu_p_value

```
ace.NeutronTable.nu_p_value
```

6.15.4.24 nu_t_energy

```
ace.NeutronTable.nu_t_energy
```

6.15.4.25 nu_t_interp_INT

```
ace.NeutronTable.nu_t_interp_INT
```

6.15.4.26 nu_t_interp_NBT

```
ace.NeutronTable.nu_t_interp_NBT
```

6.15.4.27 nu_t_type

```
ace.NeutronTable.nu_t_type
```

6.15.4.28 nu_t_value

```
ace.NeutronTable.nu_t_value
```

6.15.4.29 photon_reactions

```
ace.NeutronTable.photon_reactions
```

6.15.4.30 reactions

```
ace.NeutronTable.reactions
```

6.15.4.31 sigma

```
ace.NeutronTable.sigma
```

6.15.4.32 sigma_f

```
ace.NeutronTable.sigma_f
```

6.15.4.33 sigma_photon

```
ace.NeutronTable.sigma_photon
```

6.15.4.34 urr_energy

```
ace.NeutronTable.urr_energy
```

6.15.4.35 urr_table

```
ace.NeutronTable.urr_table
```

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

- ace.pyx

6.16 optix_stuff Class Reference

OptiX stuff class.

```
#include <optix_stuff.h>
```

Public Member Functions

- optix_stuff (unsigned, unsigned)

  constructor
- optix_stuff ()

  "default" constructor
- ∼optix_stuff ()

  destructor
- void init (wgeometry, unsigned, std::string)

  initialization function
- void trace ()

  creates a trace
- void trace (unsigned)

  creates a trace
- void trace (unsigned, unsigned)

  creates a trace
- void set_trace_type (unsigned)

  sets trace type in the OptiX context,
- void print ()

  prints a summary of OptiX information
- void make_color (float ∗, unsigned, unsigned, unsigned)

  function to test tracing
- float get_rand ()

  returns a random float
- unsigned get_outer_cell ()

  returns the outermost cell
- unsigned get_outer_cell_type ()

  returns the outermost cell type

Public Attributes

- CUdeviceptr positions_ptr
- CUdeviceptr rxn_ptr
- CUdeviceptr done_ptr
- CUdeviceptr cellnum_ptr
- CUdeviceptr talnum_ptr
- CUdeviceptr matnum_ptr
- CUdeviceptr remap_ptr
- unsigned stack_size_multiplier
- unsigned N
- float outer_cell_dims [6]

Private Member Functions

- void make_geom_xform (wgeometry)

  makes a geometry with a transform
- void make_geom_xform_common (wgeometry)

  makes a geometry with a transform
- void make_geom_prim (wgeometry)

  makes a primitive geometry (no transform)
- void init_internal (wgeometry, unsigned, std::string)

  initializes internal variables needed for OptiX stuff

Private Attributes

- optix::Context context
- std::string accel_type
- std::string traverse_type
- unsigned mincell
- unsigned maxcell
- unsigned outer_cell
- unsigned boundary_condition
- unsigned outer_cell_type
- unsigned n_materials
- unsigned compute_device
- unsigned optix_device
- unsigned GEOM_FLAG

6.16.1 Detailed Description

OptiX stuff class.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 optix_stuff() [1/2]

```
optix_stuff::optix_stuff (
            unsigned Nin,
            unsigned mult )
```

constructor

sets stack size multiplier and number of histories.

Parameters

`in`	*Nin*	- number of histories
`in`	*mult*	- stack size multiplier

6.16.2.2 optix_stuff() `[2/2]`

```
optix_stuff::optix_stuff ( )
```

"default" constructor

empty

6.16.2.3 ∼optix_stuff()

```
optix_stuff::∼optix_stuff ( )
```

destructor

6.16.3 Member Function Documentation

6.16.3.1 get_outer_cell()

```
unsigned optix_stuff::get_outer_cell ( )
```

returns the outermost cell

Returns

outer_cell - number of outermost cell

6.16.3.2 get_outer_cell_type()

```
unsigned optix_stuff::get_outer_cell_type ( )
```

returns the outermost cell type

Returns

outer_cell_type - geometrical primitive type of the outermost cell

6.16.3.3 get_rand()

```
float optix_stuff::get_rand ( )
```

returns a random float

6.16.3.4 init()

```
void optix_stuff::init (
            wgeometry problem_geom,
            unsigned compute_device_in,
            std::string accel_type_in )
```

initialization function

sets minimum and maximum cell numbers, gets material numbers, tries to initialize OptiX and throws an error if not.

Parameters

in	*problem_geom*	- problem geometry
in	*compute_device← _in*	- compute device to use (always 0)
in	*accel_type_in*	- acceleration type

6.16.3.5 init_internal()

```
void optix_stuff::init_internal (
            wgeometry problem_geom,
            unsigned compute_device_in,
            std::string accel_type_in )  [private]
```

initializes internal variables needed for OptiX stuff

sets compute device and acceleration type; sets geometry and image types; creates OptiX context; gets compute device information; sets up scene information; sets stack size; renders the buffers for particles, reactions, done flags, cell numbers, material numbers, and remaps; attaches all buffers to variables; gets CUDA pointers for buffer variables; creates programs for ray generation, exceptions, and misses; sets boundary condition for outer cell; sets trace type; sets the outer cell and gets its dimensions; creates all geometry instances; and validates and compiles the context.

Parameters

in	*problem_geom*	- problem geometry
in	*compute_device← _in*	- compute device to use (always zero)
in	*accel_type_in*	- acceleration type

6.16.3.6 make_color()

```
void optix_stuff::make_color (
             float * color,
             unsigned x,
             unsigned min,
             unsigned max )
```

function to test tracing

gets cell minimza and maxima, randomizes starting positions, copies starting positions to a pointer, traces a place to generate an image, copies the data to a local buffer, creates the images, makes the distribution random, copies the data to a pointer, executes and times the trace, and returns the time it took do to the trace.

Returns

time_out - time taken to do the trace creates a color map

Parameters

in	*color*	- color map
in	*x*	- used to check for a miss or normalize the color
in	*min,max*	- values used to normalize the color

6.16.3.7 make_geom_prim()

```
void optix_stuff::make_geom_prim (
             wgeometry problem_geom )  [private]
```

makes a primitive geometry (no transform)

makes the top level group/acceleration as children of the top level object. for each primitive in the geometry, creates the geometry types, sets the intersection and bounding box programs, sets the hit programs to the geometry material, sets the program variables for the instance, creates the instances, sets cell-specific variables, makes the geometry group fro the primitive, and puts the geometry instance into its group.

Parameters

in	*problem_geom*	- problem geometry

6.16.3.8 make_geom_xform()

```
void optix_stuff::make_geom_xform (
             wgeometry problem_geom )  [private]
```

makes a geometry with a transform

makes the top level group/acceleration as children of the top level object. for each primitive in the geometry, creates the geometry type, sets the intersection and bounding box programs, sets the hit programs to the geometry material, sets the program variables for the instance, creates the instances, sets cell-specific variables, makes the geometry group for the primitive, puts the geometry instance into its corresponding group, makes any necessary transforms, and attaches to the root node.

Parameters

in	*problem_geom*	- problem geometry

6.16.3.9 make_geom_xform_common()

```
void optix_stuff::make_geom_xform_common (
             wgeometry problem_geom )  [private]
```

makes a geometry with a transform

makes the top level group/acceleration as children of the top level object. for each primitive in the geometry, creates the geometry type, sets the intersection and bounding box programs, sets the hit programs to the geometry material, sets the program variables for the instance, creates the instances, sets cell-specific variables, makes the geometry group for the primitive, puts the geometry instance into its corresponding group, makes any necessary transforms, and attaches to the root node.

Parameters

in	*problem_geom*	- problem geometry

6.16.3.10 print()

```
void optix_stuff::print ( )
```

prints a summary of OptiX information

prints out instancing, image type, compute device, acceleration type, traverse type, stack size, and print buffer size.

6.16.3.11 set_trace_type()

```
void optix_stuff::set_trace_type (
             unsigned trace_type )
```

sets trace type in the OptiX context,

sets trace type, 2=transport (finds nearest surface, normal, writes cell number and material number), 3=fissile query(writes fissile flag into material number, writes cell number), 4=geometry plot(same as 2, but misses are squelched, no normals/intersection distances reported)

6.16.3.12 trace() [1/3]

```
void optix_stuff::trace ( )
```

creates a trace

launches the trace on the compute device with N histories.

6.16.3.13 trace() [2/3]

```
void optix_stuff::trace (
             unsigned trace_type )
```

creates a trace

sets the trace type, then launches the trace on the compute device with N histories.

Parameters

in	*trace_type*	- trace type for OptiX context

6.16.3.14 trace() [3/3]

```
void optix_stuff::trace (
             unsigned trace_type,
             unsigned n_active )
```

creates a trace

sets the trace type, then launches the trace on the compute device with n_active histories.

Parameters

in	*trace_type*	- trace type for OptiX context, 2=transport (finds nearest surface, normal, writes cell number and material number), 3=fissile query(writes fissile flag into material number, writes cell number), 4=geometry plot(same as 2, but misses are squelched, no normals/intersection distances reported)
in	*n_active*	- number of active histories

6.16.4 Member Data Documentation

6.16.4.1 accel_type

```
std::string optix_stuff::accel_type  [private]
```

acceleration type

6.16.4.2 boundary_condition

```
unsigned optix_stuff::boundary_condition  [private]
```

boundary condition of outermost cell

6.16.4.3 cellnum_ptr

```
CUdeviceptr optix_stuff::cellnum_ptr
```

CUDA cell numbers pointer

6.16.4.4 compute_device

```
unsigned optix_stuff::compute_device  [private]
```

compute device number

6.16.4.5 context

```
optix::Context optix_stuff::context  [private]
```

OptiX context

6.16.4.6 done_ptr

```
CUdeviceptr optix_stuff::done_ptr
```

CUDA done flags pointer

6.16.4.7 GEOM_FLAG

```
unsigned optix_stuff::GEOM_FLAG  [private]
```

geometry flag: 0 = primitive instancing, 1 = transform instancing, 2 = transform instancing with common primitives

6.16.4.8 matnum_ptr

```
CUdeviceptr optix_stuff::matnum_ptr
```

CUDA material numbers pointer

6.16.4.9 maxcell

```
unsigned optix_stuff::maxcell  [private]
```

maximum (usually outermost) cell

6.16.4.10 mincell

```
unsigned optix_stuff::mincell  [private]
```

minimum (usually innermost) cell

6.16.4.11 N

```
unsigned optix_stuff::N
```

number of histories

6.16.4.12 n_materials

```
unsigned optix_stuff::n_materials  [private]
```

number of materials

6.16.4.13 optix_device

```
unsigned optix_stuff::optix_device  [private]
```

optix device number, always zero since the optix device list should only have the specified cuda device in it

6.16.4.14 outer_cell

```
unsigned optix_stuff::outer_cell  [private]
```

outermost cell

6.16.4.15 outer_cell_dims

```
float optix_stuff::outer_cell_dims[6]
```

outermost cell dimensions

6.16.4.16 outer_cell_type

```
unsigned optix_stuff::outer_cell_type  [private]
```

outermost cell type

6.16.4.17 positions_ptr

```
CUdeviceptr optix_stuff::positions_ptr
```

CUDA positions pointer

6.16.4.18 remap_ptr

```
CUdeviceptr optix_stuff::remap_ptr
```

CUDA remaps pointer

6.16.4.19 rxn_ptr

```
CUdeviceptr optix_stuff::rxn_ptr
```

CUDA reactions pointer

6.16.4.20 stack_size_multiplier

```
unsigned optix_stuff::stack_size_multiplier
```

stack size multiplier

6.16.4.21 talnum_ptr

```
CUdeviceptr optix_stuff::talnum_ptr
```

CUDA tally numbers pointer

6.16.4.22 traverse_type

```
std::string optix_stuff::traverse_type  [private]
```

traverse type

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

- optix_stuff.h
- optix_stuff.cpp

6.17 particle_data Struct Reference

structure that holds all the arrays that define a particle's state

```
#include <datadef.h>
```

Public Attributes

- spatial_data ∗ space
- unsigned ∗ rxn
- float ∗ E
- float ∗ Q
- unsigned ∗ rn_bank
- unsigned ∗ cellnum
- unsigned ∗ matnum
- unsigned ∗ isonum
- int ∗ talnum
- unsigned ∗ yield
- float ∗ weight
- unsigned ∗ index

6.17.1 Detailed Description

structure that holds all the arrays that define a particle's state

Hold arrays that define a neutron's state and/or need to be passed between kernels. Data locality efficiency dictates that this must be a structure of arrays (SoA) and not be a structure built into arrays (array of structures - AoS). This container structure is passed to almost all kernels so they can access neutron state data.

6.17.2 Member Data Documentation

6.17.2.1 cellnum

```
unsigned* particle_data::cellnum
```

current cell number array

6.17.2.2 E

```
float* particle_data::E
```

energy array

6.17.2.3 index

```
unsigned* particle_data::index
```

current energy grid index array

6.17.2.4 isonum

```
unsigned* particle_data::isonum
```

current isotope number array

6.17.2.5 matnum

```
unsigned* particle_data::matnum
```

current material number array

6.17.2.6 Q

```
float* particle_data::Q
```

current reaction Q value array

6.17.2.7 rn_bank

```
unsigned* particle_data::rn_bank
```

random number seed array

6.17.2.8 rxn

```
unsigned* particle_data::rxn
```

current reaction array

6.17.2.9 space

```
spatial_data* particle_data::space
```

spatial data array

6.17.2.10 talnum

```
int* particle_data::talnum
```

current tally number array

6.17.2.11 weight

```
float* particle_data::weight
```

statistical weight array

6.17.2.12 yield

```
unsigned* particle_data::yield
```

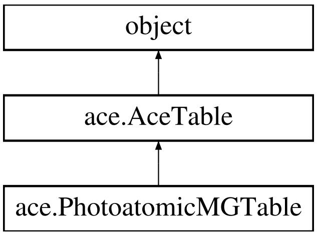
total yield of history array

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

- datadef.h

6.18 ace.PhotoatomicMGTable Class Reference

Inheritance diagram for ace.PhotoatomicMGTable:



Public Member Functions

- def __init__ (self, name, awr, temp)
- def __repr__ (self)

Additional Inherited Members

6.18.1 Constructor & Destructor Documentation

6.18.1.1 __init__()

```
def ace.PhotoatomicMGTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.18.2 Member Function Documentation

6.18.2.1 __repr__()

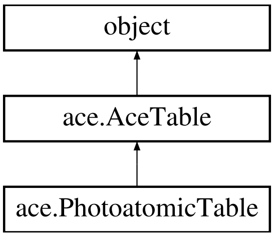
```
def ace.PhotoatomicMGTable.__repr__ (
              self )
```

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

- ace.pyx

6.19 ace.PhotoatomicTable Class Reference

Inheritance diagram for ace.PhotoatomicTable:



Public Member Functions

- def __init__ (self, name, awr, temp)
- def __repr__ (self)

Additional Inherited Members

6.19.1 Constructor & Destructor Documentation

6.19.1.1 __init__()

```
def ace.PhotoatomicTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.19.2 Member Function Documentation

6.19.2.1 __repr__()

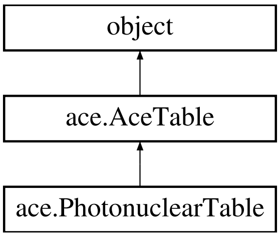
```
def ace.PhotoatomicTable.__repr__ (
              self )
```

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

- ace.pyx

6.20 ace.PhotonuclearTable Class Reference

Inheritance diagram for ace.PhotonuclearTable:



Public Member Functions

- def __init__ (self, name, awr, temp)
- def __repr__ (self)

Additional Inherited Members

6.20.1 Constructor & Destructor Documentation

6.20.1.1 __init__()

```
def ace.PhotonuclearTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.20.2 Member Function Documentation

6.20.2.1 __repr__()

```
def ace.PhotonuclearTable.__repr__ (
              self )
```

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

- ace.pyx

6.21 primitive Class Reference

primitive class

```
#include <wprimitive.h>
```

Public Member Functions

- primitive ()
- primitive (int, unsigned, std::vector< float >, std::vector< float >, std::vector< float >)
- ∼primitive ()
- unsigned add_transform ()
- unsigned add_transform (unsigned, float, float, float, float, float)
- unsigned add_transform (unsigned, unsigned, float, float, float, float, float)
- void print_transform ()
- void print_transform (int)
- void make_hex_array (int, float, float, float, unsigned)

Public Attributes

- float min [3]
- float max [3]
- float location [3]
- int type
- int primitive_id
- int n_transforms
- int material
- std::vector< wtransform > transforms

Static Public Attributes

- static int num_primitives =0

6.21.1 Detailed Description

primitive class

6.21.2 Constructor & Destructor Documentation

6.21.2.1 primitive() [1/2]

```
primitive::primitive ( )
```

box default constuctor. sets all coordinate extrema to zero, sets location to origin, sets type and material to zero.

6.21.2.2 primitive() [2/2]

```
primitive::primitive (
            int ptype,
            unsigned cellmat,
            std::vector< float > mins,
            std::vector< float > maxs,
            std::vector< float > locs )
```

box valued constructor. sets coordinate extrema to input values, sets location to input values, sets material to input material, sets type to input type, creates a wtransform, adds cell number and material to the wtransform, adds the transform to the transform list.

Parameters

in	*ptype*	- primitive type
in	*cellmat*	- cell material
in	*mins*	- object minima
in	*maxs*	- object maxima
in	*locs*	- object center points

6.21.2.3 ∼primitive()

```
primitive::∼primitive ( )
```

primitive destructor

6.21.3 Member Function Documentation

6.21.3.1 add_transform() [1/3]

```
unsigned primitive::add_transform ( )
```

adds a "default" transform - all coordinate values and angles are zero.

Returns

index of the added transform

6.21.3.2 add_transform() [2/3]

```
unsigned primitive::add_transform (
            unsigned cellnum,
            float dx,
            float dy,
            float dz,
            float theta,
            float phi )
```

adds a transform, defaults to primitive material. cell number is set to input value, coordinates are set to input values, angles are set to input values.

Parameters

in	*cellnum*	- cell number
in	*dx,dy,dz*	- transform coordinates
in	*theta,phi*	- transform azimuthal and polar angles, respectively

Returns

index of the added transform

6.21.3.3 add_transform() [3/3]

```
unsigned primitive::add_transform (
            unsigned cellnum,
            unsigned cellmat,
            float dx,
            float dy,
            float dz,
            float theta,
            float phi )
```

adds a transform. cell number is set to input value, material is set to input material, coordinates are set to input values, angles are set to input values.

Parameters

in	*cellnum*	- cell number
in	*cellmat*	- cell material
in	*dx*	- displacement in x
in	*dy*	- displacement in y
in	*dz*	- displacement in z
in	*theta*	- transform azimuthal and polar angles
in	*phi*	- transform azimuthal and polar anglesrespectively

Returns

index of the added transform

6.21.3.4 make_hex_array()

```
void primitive::make_hex_array (
            int n,
            float x,
            float y,
            float PD_ratio,
            unsigned starting_index )
```

creates a hexagonal array of elements.

Parameters

in	*n*	- edge length
in	*x*	- x dispacement coordinates
in	*y*	- y dispacement coordinates
in	*PD_ratio*	- pitch-to-diameter ratio
in	*starting_index*	- starting index

6.21.3.5 print_transform() [1/2]

```
void primitive::print_transform ( )
```

prints primitive ID, coordinate extrema, location, type and material. for each transform, prints the number, cell number, cell material, transform coordinates, and transform angles.

6.21.3.6 print_transform() [2/2]

```
void primitive::print_transform (
            int tnum )
```

prints out the properties of the input transform.

Parameters

in	*tnum*	- transform number

6.21.4 Member Data Documentation

6.21.4.1 location

```
float primitive::location[3]
```

coordinate location array

6.21.4.2 material

```
int primitive::material
```

material number

6.21.4.3 max

```
float primitive::max[3]
```

coordinate maxima array

6.21.4.4 min

```
float primitive::min[3]
```

coordinate minima array

6.21.4.5 n_transforms

```
int primitive::n_transforms
```

number of transforms

6.21.4.6 num_primitives

```
int primitive::num_primitives =0  [static]
```

number of primitives

6.21.4.7 primitive_id

```
int primitive::primitive_id
```

primitive ID number

6.21.4.8 transforms

```
std::vector<wtransform> primitive::transforms
```

transform vector

6.21.4.9 type

```
int primitive::type
```

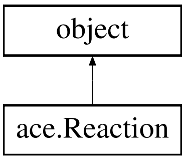
primitive type: 0 = box, 1 = cylinder, 2 = hexagon, 3 = sphere

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

- wprimitive.h
- wprimitive.cpp

6.22 ace.Reaction Class Reference

Inheritance diagram for ace.Reaction:



Public Member Functions

- def __init__ (self, MT, table=None)
- def broaden (self, T_high)
- def threshold (self)
- def __repr__ (self)

Public Attributes

- table
- MT
- Q
- TY
- IE
- sigma

6.22.1 Detailed Description

```
Reaction(MT, table=None)

A Reaction object represents a single reaction channel for a nuclide with
an associated cross section and, if present, a secondary angle and energy
distribution. These objects are stored within the ``reactions`` attribute on
subclasses of AceTable, e.g. NeutronTable.

Parameters
----------
MT : int
    The ENDF MT number for this reaction. On occasion, MCNP uses MT numbers
    that don’t correspond exactly to the ENDF specification.
table : AceTable
    The ACE table which contains this reaction. This is useful if data on
    the parent nuclide is needed (for instance, the energy grid at which
    cross sections are tabulated)

Attributes
----------
ang_energy_in : list of floats
    Incoming energies in MeV at which angular distributions are tabulated.

ang_energy_cos : list of floats
    Scattering cosines corresponding to each point of the angular distribution
    functions.

ang_energy_pdf : list of floats
    Probability distribution function for angular distribution.

ang_energy_cdf : list of floats
    Cumulative distribution function for angular distribution.

e_dist_energy : list of floats
    Incoming energies in MeV at which energy distributions are tabulated.

e_dist_law : int
    ACE law used for secondary energy distribution.

IE : int
    The index on the energy grid corresponding to the threshold of this
    reaction.

MT : int
    The ENDF MT number for this reaction. On occasion, MCNP uses MT numbers
    that don’t correspond exactly to the ENDF specification.

Q : float
    The Q-value of this reaction in MeV.

sigma : list of floats
    Microscopic cross section for this reaction at each point on the energy
    grid above the threshold value.

TY : int
    An integer whose absolute value is the number of neutrons emitted in
    this reaction. If negative, it indicates that scattering should be
    performed in the center-of-mass system. If positive, scattering should
    be preformed in the laboratory system.
```

6.22.2 Constructor & Destructor Documentation

6.22.2.1 __init__()

```
def ace.Reaction.__init__ (
              self,
              MT,
              table = None )
```

6.22.3 Member Function Documentation

6.22.3.1 __repr__()

```
def ace.Reaction.__repr__ (
              self )
```

6.22.3.2 broaden()

```
def ace.Reaction.broaden (
              self,
              T_high )
```

6.22.3.3 threshold()

```
def ace.Reaction.threshold (
              self )
```

```
threshold()

Return energy threshold for this reaction.
```

6.22.4 Member Data Documentation

6.22.4.1 IE

```
ace.Reaction.IE
```

6.22.4.2 MT

```
ace.Reaction.MT
```

6.22.4.3 Q

```
ace.Reaction.Q
```

6.22.4.4 sigma

```
ace.Reaction.sigma
```

6.22.4.5 table

```
ace.Reaction.table
```

6.22.4.6 TY

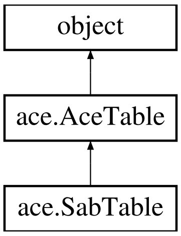
```
ace.Reaction.TY
```

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

- ace.pyx

6.23 ace.SabTable Class Reference

Inheritance diagram for ace.SabTable:



Public Member Functions

- def __init__ (self, name, awr, temp)
- def __repr__ (self)

Public Attributes

- inelastic_e_in
- inelastic_sigma
- elastic_e_in
- elastic_P
- elastic_type
- inelastic_e_out
- inelastic_mu_out
- elastic_mu_out

Private Member Functions

- def _read_all (self)
- def _read_itie (self)
- def _read_itce (self)
- def _read_itxe (self)
- def _read_itca (self)

6.23.1 Detailed Description

```
A SabTable object contains thermal scattering data as represented by
an S(alpha, beta) table.

Parameters
----------
name : str
    ZAID identifier of the table, e.g. lwtr.10t.
awr : float
    Atomic mass ratio of the target nuclide.
temp : float
    Temperature of the target nuclide in eV.

Attributes
----------
awr : float
    Atomic mass ratio of the target nuclide.

elastic_e_in : list of floats
    Incoming energies in MeV for which the elastic cross section is
    tabulated.

elastic_P : list of floats
    Elastic scattering cross section for data derived in the incoherent
    approximation, or Bragg edge parameters for data derived in the coherent
    approximation.

elastic_type : str
    Describes the behavior of the elastic cross section, i.e. whether it was
    derived in the incoherent or coherent approximation.

inelastic_e_in : list of floats
    Incoming energies in MeV for which the inelastic cross section is
    tabulated.
```

```
inelastic_sigma : list of floats
    Inelastic scattering cross section in barns at each energy.

name : str
    ZAID identifier of the table, e.g. 92235.70c.

temp : float
    Temperature of the target nuclide in eV.
```

6.23.2 Constructor & Destructor Documentation

6.23.2.1 __init__()

```
def ace.SabTable.__init__ (
              self,
              name,
              awr,
              temp )
```

6.23.3 Member Function Documentation

6.23.3.1 __repr__()

```
def ace.SabTable.__repr__ (
              self )
```

6.23.3.2 _read_all()

```
def ace.SabTable._read_all (
              self ) [private]
```

6.23.3.3 _read_itca()

```
def ace.SabTable._read_itca (
              self ) [private]
```

```
Read angular distributions for elastic scattering.
```

6.23.3.4 _read_itce()

```
def ace.SabTable._read_itce (
              self )  [private]
```

Read energy-dependent elastic scattering cross sections.

6.23.3.5 _read_itie()

```
def ace.SabTable._read_itie (
              self )  [private]
```

Read energy-dependent inelastic scattering cross sections.

6.23.3.6 _read_itxe()

```
def ace.SabTable._read_itxe (
              self )  [private]
```

Read coupled energy/angle distributions for inelastic scattering.

6.23.4 Member Data Documentation

6.23.4.1 elastic_e_in

```
ace.SabTable.elastic_e_in
```

6.23.4.2 elastic_mu_out

```
ace.SabTable.elastic_mu_out
```

6.23.4.3 elastic_P

```
ace.SabTable.elastic_P
```

6.23.4.4 elastic_type

```
ace.SabTable.elastic_type
```

6.23.4.5 inelastic_e_in

```
ace.SabTable.inelastic_e_in
```

6.23.4.6 inelastic_e_out

```
ace.SabTable.inelastic_e_out
```

6.23.4.7 inelastic_mu_out

```
ace.SabTable.inelastic_mu_out
```

6.23.4.8 inelastic_sigma

```
ace.SabTable.inelastic_sigma
```

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

- ace.pyx

6.24 spatial_data Struct Reference

contains the spatial parameters of the neutron

```
#include <datadef.h>
```

Public Attributes

- float x
- float y
- float z
- float xhat
- float yhat
- float zhat
- float surf_dist
- float norm [3]
- unsigned enforce_BC
- unsigned weight

6.24.1 Detailed Description

contains the spatial parameters of the neutron

3D Cartesian coordinates, direction vector, distance to nearest surface, total macroscopic cross section, surface normal vector, boundary condition enforcement flag, neutron statsitical weight

6.24.2 Member Data Documentation

6.24.2.1 enforce_BC

```
unsigned spatial_data::enforce_BC
```

boundary condition enforcement flag

6.24.2.2 norm

```
float spatial_data::norm[3]
```

normal vector of surface intersection

6.24.2.3 surf_dist

```
float spatial_data::surf_dist
```

distance to nearest surface

6.24.2.4 weight

```
unsigned spatial_data::weight
```

particle statistical weight

6.24.2.5 x

```
float spatial_data::x
```

x-coordinate

6.24.2.6 xhat

```
float spatial_data::xhat
```

direction vector x-component

6.24.2.7 y

```
float spatial_data::y
```

y-coordinate

6.24.2.8 yhat

```
float spatial_data::yhat
```

direction vector y-component

6.24.2.9 z

```
float spatial_data::z
```

z-coordinate

6.24.2.10 zhat

```
float spatial_data::zhat
```

direction vector z-component

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

- datadef.h

6.25 tally_data Struct Reference

Tally data that lives on the device side.

```
#include <datadef.h>
```

Public Attributes

- float $*$ score
- float $*$ square
- unsigned $*$ count
- unsigned cell
- unsigned length
- float E_min
- float E_max

6.25.1 Detailed Description

Tally data that lives on the device side.

Tally data that lives on the device side. Everything needed to properly index a tally and the vectors to store the scores and keep track of statistics

6.25.2 Member Data Documentation

6.25.2.1 cell

```
unsigned tally_data::cell
```

tally cell (input)

6.25.2.2 count

```
unsigned* tally_data::count
```

tally count

6.25.2.3 E_max

```
float tally_data::E_max
```

maximum energy (input)

6.25.2.4 E_min

```
float tally_data::E_min
```

minimum energy (input)

6.25.2.5 length

```
unsigned tally_data::length
```

tally length, edges are equi-log (input)

6.25.2.6 score

```
float* tally_data::score
```

tally score

6.25.2.7 square

```
float* tally_data::square
```

tally square

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

- datadef.h

6.26 tally_data_host Struct Reference

Tally data that lives on the host side.

```
#include <datadef.h>
```

Public Attributes

- float ∗ score
- float ∗ square
- unsigned ∗ count
- double ∗ score_total
- double ∗ square_total
- long unsigned ∗ count_total
- unsigned cell
- unsigned length
- float E_min
- float E_max

6.26.1 Detailed Description

Tally data that lives on the host side.

Tally data that lives on the host side. Basically holds the same thing as the device array but also contains 64-bit arrays that the tallies are accumulated into to avoid too much roundoff error.

6.26.2 Member Data Documentation

6.26.2.1 cell

```
unsigned tally_data_host::cell
```

tally cell (input)

6.26.2.2 count

```
unsigned* tally_data_host::count
```

tally count

6.26.2.3 count_total

```
long unsigned* tally_data_host::count_total
```

tally count accumulated total

6.26.2.4 E_max

```
float tally_data_host::E_max
```

maximum energy (input)

6.26.2.5 E_min

```
float tally_data_host::E_min
```

minimum energy (input)

6.26.2.6 length

```
unsigned tally_data_host::length
```

tally length, edges are equi-log (input)

6.26.2.7 score

```
float* tally_data_host::score
```

tally score

6.26.2.8 score_total

```
double* tally_data_host::score_total
```

tally score accumulated total

6.26.2.9 square

```
float* tally_data_host::square
```

tally square

6.26.2.10 square_total

```
double* tally_data_host::square_total
```

tally square accumulaed total

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

- datadef.h

6.27 wfloat3 Class Reference

class definitions for device vector operations

```
#include <wfloat3.h>
```

Public Member Functions

- __device__ wfloat3 ()

  sets x,y,z to 0,0,0
- __device__ wfloat3 (float)

  sets x,y,z to a,a,a
- __device__ wfloat3 (float, float, float)

  sets x,y,z to a,b,c
- __device__ wfloat3 operator+ (wfloat3)

  vector addition operator
- __device__ wfloat3 operator- (wfloat3)

  vector subtraction operator
- __device__ wfloat3 operator* (wfloat3)

  vector multiplication operator
- __device__ wfloat3 operator+ (float)

  scalar addition operator
- __device__ wfloat3 operator- (float)

  scalar subtraction operator
- __device__ wfloat3 operator* (float)

  scalar multiplication operator
- __device__ wfloat3 operator/ (float)

  scalar divison operator
- __device__ wfloat3 cross (wfloat3)

  cross product operator
- __device__ float dot (wfloat3)

  dot product operator
- __device__ void rodrigues_rotation (wfloat3, float)

  Rodrigues' rotation operator.
- __device__ wfloat3 rotate (float, float)

  rotation about random cosine
- __device__ float norm2 ()

  returns square root of sum of squares of coordinates

Public Attributes

- float x
- float y
- float z

6.27.1 Detailed Description

class definitions for device vector operations

6.27.2 Constructor & Destructor Documentation

6.27.2.1 wfloat3() [1/3]

```
__device__ wfloat3::wfloat3 ( )  [inline]
```

sets x,y,z to 0,0,0

6.27.2.2 wfloat3() [2/3]

```
__device__ wfloat3::wfloat3 (
              float a )  [inline]
```

sets x,y,z to a,a,a

Parameters

in	*a*	- point to set

6.27.2.3 wfloat3() [3/3]

```
__device__ wfloat3::wfloat3 (
              float a,
              float b,
              float c )  [inline]
```

sets x,y,z to a,b,c

Parameters

in	*a,b,c*	- points to set

6.27.3 Member Function Documentation

6.27.3.1 cross()

```
__device__ wfloat3 wfloat3::cross (
             wfloat3 arg )  [inline]
```

cross product operator

returns the cross product of the vector and arg

Parameters

in	*arg*	- vector to cross

Returns

result - resultant wfloat3

6.27.3.2 dot()

```
__device__ float wfloat3::dot (
             wfloat3 arg )  [inline]
```

dot product operator

returns the dot product of the vector and arg

Parameters

in	*arg*	- vector to dot

6.27.3.3 norm2()

```
__device__ float wfloat3::norm2 ( )  [inline]
```

returns square root of sum of squares of coordinates

6.27.3.4 operator*() [1/2]

```
__device__ wfloat3 wfloat3::operator* (
             wfloat3 arg )  [inline]
```

vector multiplication operator

multiplies x and x-component of input wfloat3, etc.

Parameters

in	*arg*	- wfloat3 coordinates to be multiplied

Returns

result - resultant wfloat3

6.27.3.5 operator*() [2/2]

```
__device__ wfloat3 wfloat3::operator* (
             float arg )  [inline]
```

scalar multiplication operator

multiplies x and arg, etc.

Parameters

in	*arg*	- number by which to multiply

Returns

result - resultant wfloat3

6.27.3.6 operator+() [1/2]

```
__device__ wfloat3 wfloat3::operator+ (
             wfloat3 arg )  [inline]
```

vector addition operator

adds x and x-component of input wfloat3, etc.

Parameters

in	*arg*	- wfloat3 coordinates to be added

Returns

result - resultant wfloat3

6.27.3.7 operator+() [2/2]

```
__device__ wfloat3 wfloat3::operator+ (
             float arg )  [inline]
```

scalar addition operator

adds x and arg, etc.

Parameters

in	*arg*	- number to be added

Returns

result - resultant wfloat3

6.27.3.8 operator-() [1/2]

```
__device__ wfloat3 wfloat3::operator- (
             wfloat3 arg )  [inline]
```

vector subtraction operator

subtracts x-component of input wfloat3 from x, etc.

Parameters

in	*arg*	- wfloat3 coordinates to be subtracted

Returns

result - resultant wfloat3

6.27.3.9 operator-() [2/2]

```
__device__ wfloat3 wfloat3::operator- (
              float arg )  [inline]
```

scalar subtraction operator

subtracts arg from x, etc.

Parameters

in	*arg*	- number to be subtracted

Returns

result - resultant wfloat3

6.27.3.10 operator/()

```
__device__ wfloat3 wfloat3::operator/ (
              float arg )  [inline]
```

scalar divison operator

divides x by arg, etc.

Parameters

in	*arg*	- number by which to divide

Returns

result - resultant wfloat3

6.27.3.11 rodrigues_rotation()

```
__device__ void wfloat3::rodrigues_rotation (
              wfloat3 k,
              float theta )  [inline]
```

Rodrigues' rotation operator.

rotates a vector in space, given axis and angle of rotation

Parameters

in	*k*	- unit vector describing axis of rotation about which to rotate
in	*theta*	- angle by which to rotate

6.27.3.12 rotate()

```
__device__ wfloat3 wfloat3::rotate (
             float mu,
             float rn )  [inline]
```

rotation about random cosine

borrowed from OpenMC

Parameters

in	*mu*	- random cos(theta)
in	*rn*	- random number

6.27.4 Member Data Documentation

6.27.4.1 x

```
float wfloat3::x
```

x-coordinate

6.27.4.2 y

```
float wfloat3::y
```

y-coordinate

6.27.4.3 z

```
float wfloat3::z
```

z-coordinate

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

- wfloat3.h

6.28 wgeometry Class Reference

wgeometry class

```
#include <wgeometry.h>
```

Public Member Functions

- wgeometry ()
- ∼wgeometry ()
- unsigned get_minimum_cell ()
- unsigned get_maximum_cell ()
- unsigned get_minimum_material ()
- unsigned get_maximum_material ()
- unsigned get_primitive_count ()
- unsigned get_transform_count ()
- unsigned add_primitive ()
- unsigned add_primitive (int, unsigned, std::vector< float >, std::vector< float >, std::vector< float >)
- void add_tally (unsigned)
- void update ()
- void print_summary ()
- void print_all ()
- void set_outer_cell (unsigned, unsigned)
- unsigned get_outer_cell ()
- unsigned get_boundary_condition ()
- unsigned get_outer_cell_type ()
- void add_material (unsigned, unsigned, unsigned, float, std::vector< std::string >, std::vector< float >)
- int check ()
- unsigned get_outer_cell_dims (float ∗)
- unsigned get_material_count ()
- void make_material_table ()
- void get_material_table (unsigned ∗, unsigned ∗, float ∗∗)
- void print_materials_table ()
- void set_datapath (std::string)
- unsigned check_fissile ()
- unsigned add_transform (unsigned)
- unsigned add_transform (unsigned, unsigned, float, float, float, float, float)
- unsigned add_transform (unsigned, unsigned, unsigned, float, float, float, float, float)
- void make_hex_array (unsigned, int, float, float, float, unsigned)
- void delete_transform (unsigned, unsigned)
- void delete_primitive (unsigned)

Public Attributes

- unsigned n_box
- unsigned n_cyl
- unsigned n_hex
- unsigned n_sph
- unsigned n_primitives
- unsigned n_transforms
- unsigned outer_cell
- unsigned n_materials

- unsigned n_isotopes
- unsigned n_tallies
- unsigned fissile_flag
- unsigned boundary_condition
- unsigned ∗ material_num_list
- unsigned ∗ cell_num_list
- std::string datapath
- std::vector< primitive > primitives
- std::vector< unsigned > tally_cells
- std::vector< material_def > materials
- std::vector< std::string > isotopes
- std::string isotope_list
- unsigned ∗ isotope_list_array
- unsigned ∗ material_list_array
- float ∗ concentrations_matrix
- float ∗ awr_list

6.28.1 Detailed Description

wgeometry class

6.28.2 Constructor & Destructor Documentation

6.28.2.1 wgeometry()

```
wgeometry::wgeometry ( )
```

wgeometry constructor

6.28.2.2 ∼wgeometry()

```
wgeometry::∼wgeometry ( )
```

wgeometry destructor

6.28.3 Member Function Documentation

6.28.3.1 add_material()

```
void wgeometry::add_material (
            unsigned matnum,
            unsigned is_fissile,
            unsigned num_topes,
            float density,
            std::vector< std::string > isotopes,
            std::vector< float > fractions )
```

adds a material and its properties to the geometry, allocates space for all of the material information.

Parameters

in	*matnum*	- material number
in	*is_fissile*	- fissile flag
in	*num_topes*	- number of isotopes in material
in	*density*	- density of material
in	*isotopes*	- list of isotopes
in	*fractions*	- fractions of the constituent isotopes

6.28.3.2 add_primitive() [1/2]

```
unsigned wgeometry::add_primitive ( )
```

adds a primitive to the geometry object

6.28.3.3 add_primitive() [2/2]

```
unsigned wgeometry::add_primitive (
             int ptype,
             unsigned cellmat,
             std::vector< float > mins,
             std::vector< float > maxs,
             std::vector< float > origin )
```

6.28.3.4 add_tally()

```
void wgeometry::add_tally (
             unsigned cellnum_in )
```

adds a tally for the specified cell to the geometry object

6.28.3.5 add_transform() [1/3]

```
unsigned wgeometry::add_transform (
             unsigned index )
```

adds a default transform to a primitive

Parameters

in	*index*	- the index of the primitive to add the transform to

Returns

the index of the transform for the specified primitive

6.28.3.6 add_transform() [2/3]

```
unsigned wgeometry::add_transform (
            unsigned index,
            unsigned cellnum,
            float dx,
            float dy,
            float dz,
            float theta,
            float phi )
```

adds a transform to a primitive

Parameters

in	*index*	- the index of the primitive to add the transform to
in	*cellnum*	- cell number associated with the transform
in	*dx*	- displacement in x
in	*dy*	- displacement in y
in	*dz*	- displacement in z
in	*theta*	- polar rotation in the xz plane
in	*phi*	- azimuthal rotation in the xy plane

Returns

the index of the transform for the specified primitive

6.28.3.7 add_transform() [3/3]

```
unsigned wgeometry::add_transform (
            unsigned index,
            unsigned cellnum,
            unsigned cellmat,
            float dx,
            float dy,
            float dz,
            float theta,
            float phi )
```

adds a transform to a primitive

Parameters

in	*index*	- the index of the primitive to add the transform to
in	*cellnum*	- cell number associated with the transform
in	*cellmat*	- material number associated with the transform
in	*dx*	- displacement in x
in	*dy*	- displacement in y
in	*dz*	- displacement in z
in	*theta*	- polar rotation in the yz plane
in	*phi*	- azimuthal rotation in the xy plane

Returns

the index of the transform for the specified primitive

6.28.3.8 check()

```
int wgeometry::check ( )
```

checks that all cells have unique IDs, checks that there are materials for each number specified in the geometry, checks to make sure that the outer cell exists, checks to see if there are any fissile isotopes.

6.28.3.9 check_fissile()

```
unsigned wgeometry::check_fissile ( )
```

checks whether or not the geometry contains a fissile material.

Returns

fissile_flag

6.28.3.10 delete_primitive()

```
void wgeometry::delete_primitive (
            unsigned index )
```

delete a primitive

Parameters

in	*index*	- primitive index

6.28.3.11 delete_transform()

```
void wgeometry::delete_transform (
            unsigned index,
            unsigned element )
```

delete a transform

Parameters

in	*index*	- primitive index
in	*element*	- transform index

6.28.3.12 get_boundary_condition()

```
unsigned wgeometry::get_boundary_condition ( )
```

returns the outermost cell type

Returns

outer_cell_type

6.28.3.13 get_material_count()

```
unsigned wgeometry::get_material_count ( )
```

returns the number of materials.

Returns

n_materials

6.28.3.14 get_material_table()

```
void wgeometry::get_material_table (
            unsigned * n_mat_in,
            unsigned * n_tope_in,
            float ** conc_mat_in )
```

creates material and isotope arrays, creates concentration matrix. copies memory for all of those arrays.

Parameters

in	*n_mat_in*	- number of input materials
in	*n_tope_in*	- number of input isotopes
in	*conc_mat↩ _in*	- pointer to pointer to set as location of the now-generated concentration matrix

6.28.3.15 get_maximum_cell()

```
unsigned wgeometry::get_maximum_cell ( )
```

returns the largest cell number, typically the outermost cell

6.28.3.16 get_maximum_material()

```
unsigned wgeometry::get_maximum_material ( )
```

returns the largest material number

6.28.3.17 get_minimum_cell()

```
unsigned wgeometry::get_minimum_cell ( )
```

returns the smallest cell number, typically the innermost cell

6.28.3.18 get_minimum_material()

```
unsigned wgeometry::get_minimum_material ( )
```

returns the smallest material number

6.28.3.19 get_outer_cell()

```
unsigned wgeometry::get_outer_cell ( )
```

returns the outermost cell and its boundary conition

Returns

outer_cell

6.28.3.20 get_outer_cell_dims()

```
unsigned wgeometry::get_outer_cell_dims (
             float * input_array )
```

returns the dimensions of the outermost cell.

Parameters

in	*input_array*	

Returns

primitives[k].type

6.28.3.21 get_outer_cell_type()

```
unsigned wgeometry::get_outer_cell_type ( )
```

returns the boundary condition flag of outermost cell type

Returns

boundary_condition

6.28.3.22 get_primitive_count()

```
unsigned wgeometry::get_primitive_count ( )
```

returns the number of primitves in the geometry object

6.28.3.23 get_transform_count()

```
unsigned wgeometry::get_transform_count ( )
```

returns the number of transforms in the geometry object

6.28.3.24 make_hex_array()

```
void wgeometry::make_hex_array (
             unsigned index,
             int n,
             float x,
             float y,
             float phi,
             unsigned starting_index )
```

make a bunch of transforms in a hex arrangement

Parameters

in	*index*	- the index of the primitive to add the transform to
in	*n*	- the number of elements on an edge
in	*x*	- x position (not used)
in	*y*	- y position (not used)
in	*phi*	- azimuthal xy plane rotation position (not used)
in	*starting_index*	- the first index of the new transforms added, spans range starting_index to starting_index + n

6.28.3.25 make_material_table()

```
void wgeometry::make_material_table ( )
```

makes a table of all of the materials.

allocates and copies the isotope and material number lists to their respsective arrays, allocates and copies the isotope fractions to the concentration matrix, converts the fractions into number densities, normalizes the fractions, gets the average number density, prints each isotope's material, isotope, and density. memcpy(isotope_list_←↩ array,isotopes.data(),n_isotopes∗sizeof(unsigned));

6.28.3.26 print_all()

```
void wgeometry::print_all ( )
```

prints all of the transforms of all the primitives, then prints a geometry summary.

6.28.3.27 print_materials_table()

```
void wgeometry::print_materials_table ( )
```

prints out all materials, including each material's constituent isotopes and their number densities.

6.28.3.28 print_summary()

```
void wgeometry::print_summary ( )
```

prints a summary of the geometry object: numbers of the different kinds of shapes in the geometry, total numbers of primitives and transforms, outer cell, numbers of materials and isotopes, isotope list, properties (density, fissile flag, isotopes) of each material.

6.28.3.29 set_datapath()

```
void wgeometry::set_datapath (
             std::string path_in )
```

sets the data path

Parameters

in	*path↩ _in*	- input DATAPATH to the xs data

6.28.3.30 set_outer_cell()

```
void wgeometry::set_outer_cell (
             unsigned ocell,
             unsigned BC )
```

goes through all the cells of all of the primitives and checks that the outer cell is set

Parameters

in	*ocell*	- the outermost cell
in	*BC*	- the boundary condiction for the outermost cell (1=vacuum 2=mirror)

6.28.3.31 update()

```
void wgeometry::update ( )
```

updates the numbers of all shapes, compiles the list of all isotopes, creates an isotope table, sets tally indicies to cells.

6.28.4 Member Data Documentation

6.28.4.1 awr_list

```
float* wgeometry::awr_list
```

atomic weight ratio (AWR) list

6.28.4.2 boundary_condition

```
unsigned wgeometry::boundary_condition
```

flag for the cell's boundary condition

6.28.4.3 cell_num_list

```
unsigned* wgeometry::cell_num_list
```

list of cell numbers

6.28.4.4 concentrations_matrix

```
float* wgeometry::concentrations_matrix
```

concentrations matrix

6.28.4.5 datapath

```
std::string wgeometry::datapath
```

path to xsdir and data

6.28.4.6 fissile_flag

```
unsigned wgeometry::fissile_flag
```

indicates whether or not a material is fissile

6.28.4.7 isotope_list

```
std::string wgeometry::isotope_list
```

isotope list

6.28.4.8 isotope_list_array

```
unsigned* wgeometry::isotope_list_array
```

isotope list array

6.28.4.9 isotopes

```
std::vector<std::string> wgeometry::isotopes
```

isotopes vector

6.28.4.10 material_list_array

```
unsigned* wgeometry::material_list_array
```

material list array

6.28.4.11 material_num_list

```
unsigned* wgeometry::material_num_list
```

list of material numbers

6.28.4.12 materials

```
std::vector<material_def> wgeometry::materials
```

materials vector

6.28.4.13 n_box

```
unsigned wgeometry::n_box
```

number of boxes

6.28.4.14 n_cyl

```
unsigned wgeometry::n_cyl
```

number of cylinders

6.28.4.15 n_hex

```
unsigned wgeometry::n_hex
```

number of hexagons

6.28.4.16 n_isotopes

```
unsigned wgeometry::n_isotopes
```

number of isotopes

6.28.4.17 n_materials

```
unsigned wgeometry::n_materials
```

number of materials

6.28.4.18 n_primitives

```
unsigned wgeometry::n_primitives
```

number of primitives

6.28.4.19 n_sph

```
unsigned wgeometry::n_sph
```

number of spheres

6.28.4.20 n_tallies

```
unsigned wgeometry::n_tallies
```

number of tallies

6.28.4.21 n_transforms

```
unsigned wgeometry::n_transforms
```

number of transforms

6.28.4.22 outer_cell

```
unsigned wgeometry::outer_cell
```

outermost cell (usually used for tallying)

6.28.4.23 primitives

```
std::vector<primitive> wgeometry::primitives
```

primitives vector

6.28.4.24 tally_cells

```
std::vector<unsigned> wgeometry::tally_cells
```

primitives vector

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

- wgeometry.h
- wgeometry.cpp

6.29 whistory Class Reference

whistory class

```
#include <whistory.h>
```

Public Member Functions

- whistory (unsigned, wgeometry)

  constructor
- ∼whistory ()

  destructor
- void print_xs_data ()

  prints cross section data information
- void print_pointers ()

  prints pointer information
- void print_materials_table ()

  prints table of properties of geometry materials
- void run ()

  runs history
- void write_xs_data (std::string)

  writes cross section data to file
- void write_tally ()

  writes all tally values to file
- void set_run_type (unsigned)

  sets run type
- void set_run_type (std::string)

  sets run type
- void set_run_param (unsigned, unsigned)

  sets number of cycles to skip and number of active cycles
- void init ()

  initialization function
- void device_report ()

  prints out details (model, memory, compute capability, etc.) of all available compute devices
- void set_device (unsigned)

  sets device number to input value
- void set_acceration (std::string)

  does nothing
- void set_filename (std::string)

  sets filename to input string
- void set_print_level (unsigned level)

  sets amount of information printed to stdout
- void set_dump_level (unsigned level)

  sets what types of information are dumped to files
- void plot_geom (std::string type)

  produces png images of the geometry, named filename-[xy,xz,yz].png
- void make_color (float ∗, unsigned, unsigned, unsigned)

  creates a color map
- void hot2 (float ∗, long unsigned, long unsigned, long unsigned)

  creates a hot2 color map
- void nonzero (float ∗, unsigned, unsigned, unsigned)

  creates a binary colormap, black iff 0
- void bin_fission_points (spatial_data ∗, unsigned)

  bins and accumulates fission points to grid
- void write_fission_points ()

  writes binned fission point image to a .png
- void print_banner ()

prints an amazing and beautiful WARP banner to stdout

- void get_Py_buffer_dims (unsigned ∗, unsigned ∗, unsigned ∗, Py_buffer ∗)

  does size logic on python buffer and writes values into pointers passed. Made since python interface routines don't set first/second shape value if the array size is 0/1

Private Member Functions

- void init_RNG ()

  initializes the random number generator
- void update_RNG ()

  updates the random number
- void init_CUDPP ()

  initializes CUDPP
- void init_host ()

  initializes data on the host
- void init_device ()

  initializes data on the device
- void copy_data_to_device ()

  copies data from the host device to the compute device
- void init_cross_sections ()

  loads cross sections
- void trace (unsigned)

  does an OptiX trace
- void trace (unsigned, unsigned)

  does an OptiX trace for a given number of active histories
- unsigned reduce_done ()

  reduces done values
- void reset_cycle (float)

  resets the cycle in criticality mode
- void reset_fixed ()

  resets the cycle in fixed-source mode
- void converge (unsigned)

  not called in whistory.cpp
- void sample_fissile_points ()

  samples fissile points
- unsigned reduce_yield ()

  reduces yield values
- float reduce_weight ()

  reduces weight values
- void accumulate_keff (unsigned, unsigned, double ∗, float ∗)

  accumulates yields into host side values
- void accumulate_tally ()

  raccumulates the flux tally
- float get_time ()

  returns how long it takes to do something
- void prep_secondaries ()

  prepares for secondary neutrons
- unsigned map_active ()

  maps done histories

- void remap_active (unsigned ∗, unsigned ∗, unsigned ∗, unsigned ∗, unsigned ∗, unsigned ∗, unsigned ∗, unsigned ∗, unsigned ∗)

    remaps active histories
- void write_to_file (spatial_data ∗, unsigned, std::string, std::string)

    prints the locations of the source points to file
- void write_to_file (spatial_data ∗, float ∗, unsigned, std::string, std::string)

    prints the locations of the source points to file
- void write_to_file (unsigned ∗, unsigned, std::string, std::string)

    prints the source points to a file
- void write_to_file (unsigned ∗, unsigned ∗, unsigned, std::string, std::string)

    prints the locations of the source points to file
- void write_results (float, float, std::string)

    writes results to file
- void copy_python_buffer (float ∗∗, float ∗∗, std::string)

    calls python function, copys returned buffer to C and CUDA pointers, float overload
- void copy_python_buffer (unsigned ∗∗, unsigned ∗∗, std::string)

    calls python function, copys returned buffer to C and CUDA pointers, unsigned overload
- void copy_python_buffer (float ∗∗, std::string)

    calls python function, copys returned buffer to C pointer (no cuda), float overload
- void copy_python_buffer (unsigned ∗∗, std::string)

    calls python function, copys returned buffer to C pointer (no cuda), unsigned overload
- int init_python ()

    initialize cross section data object in python
- void copy_scatter_data ()
- void copy_energy_data ()

Private Attributes

- wgeometry problem_geom
- std::string accel_type
- CUDPPHandle theCudpp
- CUDPPConfiguration compact_config
- CUDPPConfiguration scan_int_config
- CUDPPConfiguration redu_int_config
- CUDPPConfiguration redu_float_config
- CUDPPConfiguration radix_config
- CUDPPHandle scanplan_int
- CUDPPHandle reduplan_int
- CUDPPHandle reduplan_float
- CUDPPHandle compactplan
- CUDPPHandle radixplan
- CUDPPResult res
- curandGenerator_t rand_gen
- unsigned N
- unsigned Ndataset
- unsigned NUM_THREADS
- unsigned blks
- unsigned compute_device
- cudaStream_t stream [5]
- cross_section_data ∗ d_xsdata
- cross_section_data dh_xsdata
- cross_section_data h_xsdata

- particle_data ∗ d_particles
- particle_data dh_particles
- particle_data h_particles
- tally_data ∗ d_tally
- tally_data ∗ dh_tally
- tally_data_host ∗ h_tally
- dist_container ∗ dh_dist_energy
- dist_container ∗ dh_dist_scatter
- unsigned n_edges
- unsigned ∗ edges
- unsigned ∗ d_edges
- unsigned ∗ reduced_yields
- float ∗ reduced_weight
- unsigned ∗ d_reduced_yields
- float ∗ d_reduced_weight
- long unsigned reduced_yields_total
- double reduced_weight_total
- double initial_weight_total
- unsigned ∗ material_list
- unsigned ∗ isotope_list
- float ∗ number_density_matrix
- unsigned ∗ d_material_list
- unsigned ∗ d_isotope_list
- float ∗ d_number_density_matrix
- unsigned ∗ remap
- unsigned ∗ d_remap
- unsigned ∗ zeros
- unsigned ∗ d_zeros
- unsigned ∗ ones
- int ∗ mones
- float ∗ fones
- PyObject ∗ xsdat_instance
- unsigned RUN_FLAG
- unsigned outer_cell
- unsigned outer_cell_type
- unsigned n_materials
- unsigned n_isotopes
- unsigned n_tallies
- unsigned n_skip
- unsigned n_cycles
- float keff_sum
- float keff2_sum
- float keff_err
- std::string filename
- unsigned is_initialized
- unsigned print_flag
- unsigned dump_flag
- size_t mem_free
- size_t mem_total
- unsigned ∗ d_valid_result
- unsigned ∗ d_valid_N
- float ∗ d_fissile_energy
- spatial_data ∗ d_fissile_points
- unsigned ∗ d_scanned
- unsigned ∗ d_num_completed

- unsigned ∗ d_num_active
- spatial_data ∗ d_bank_space
- float ∗ d_bank_E
- std::vector< std::string > isotopes
- std::vector< unsigned > xs_num_rxns
- std::vector< unsigned > xs_isotope_ints
- unsigned total_bytes_scatter
- unsigned total_bytes_energy
- unsigned MT_rows
- unsigned MT_columns
- float outer_cell_dims [6]
- long unsigned ∗ fiss_img

6.29.1 Detailed Description

whistory class

6.29.2 Constructor & Destructor Documentation

6.29.2.1 whistory()

```
whistory::whistory (
             unsigned Nin,
             wgeometry problem_geom_in )
```

constructor

makes geometry, sets tally vector length, creates dataset size, sets compute device and acceleration type, creates CUDA streams.

6.29.2.2 ∼whistory()

```
whistory::∼whistory ( )
```

destructor

6.29.3 Member Function Documentation

6.29.3.1 accumulate_keff()

```
void whistory::accumulate_keff (
             unsigned converged,
             unsigned iteration,
             double ∗ keff,
             float ∗ keff_cycle )  [private]
```

accumulates yields into host side values

Parameters

in	*converged*	- tally accumulation flag
in	*iteration*	- the active iteration number (starts at 0)
in	*keff*	- the running cumulative keff
in	*keff_cycle*	- the keff of the last cycle, used to renormalize the source vector

6.29.3.2 accumulate_tally()

```
void whistory::accumulate_tally ( )  [private]
```

raccumulates the flux tally

6.29.3.3 bin_fission_points()

```
void whistory::bin_fission_points (
             spatial_data * d_space,
             unsigned N )
```

bins and accumulates fission points to grid

Parameters

in	*d_space*	- device space points
in	*N*	- dataset size

6.29.3.4 converge()

```
void whistory::converge (
             unsigned  )  [private]
```

not called in whistory.cpp

6.29.3.5 copy_data_to_device()

```
void whistory::copy_data_to_device ( )  [private]
```

copies data from the host device to the compute device

copies history data, cross section data, and the device pointer array. zeros out the tally arrays.

6.29.3.6 copy_energy_data()

```
void whistory::copy_energy_data ( )  [private]
```

high level function that does the energy distribution data copying

6.29.3.7 copy_python_buffer() [1/4]

```
void whistory::copy_python_buffer (
            float ** device_pointer,
            float ** host_pointer,
            std::string function_name )  [private]
```

calls python function, copys returned buffer to C and CUDA pointers, float overload

Parameters

in	*device_pointer*	- device pointer
in	*host_pointer*	- host pointer
in	*function_name*	- python function name

6.29.3.8 copy_python_buffer() [2/4]

```
void whistory::copy_python_buffer (
            unsigned ** device_pointer,
            unsigned ** host_pointer,
            std::string function_name )  [private]
```

calls python function, copys returned buffer to C and CUDA pointers, unsigned overload

Parameters

in	*device_pointer*	- device pointer
in	*host_pointer*	- host pointer
in	*function_name*	- python function name

6.29.3.9 copy_python_buffer() [3/4]

```
void whistory::copy_python_buffer (
            float ** host_pointer,
            std::string function_name )  [private]
```

calls python function, copys returned buffer to C pointer (no cuda), float overload

Parameters

in	*host_pointer*	- host pointer
in	*function_name*	- python function name

6.29.3.10 copy_python_buffer() [4/4]

```
void whistory::copy_python_buffer (
            unsigned ** host_pointer,
            std::string function_name )  [private]
```

calls python function, copys returned buffer to C pointer (no cuda), unsigned overload

Parameters

in	*host_pointer*	- host pointer
in	*function_name*	- python function name

6.29.3.11 copy_scatter_data()

```
void whistory::copy_scatter_data ( )  [private]
```

high level function that does the scatter distribution data copying

6.29.3.12 device_report()

```
void whistory::device_report ( )
```

prints out details (model, memory, compute capability, etc.) of all available compute devices

6.29.3.13 get_Py_buffer_dims()

```
void whistory::get_Py_buffer_dims (
            unsigned * rows,
            unsigned * columns,
            unsigned * bytes,
            Py_buffer * pBuff )
```

does size logic on python buffer and writes values into pointers passed. Made since python interface routines don't set first/second shape value if the array size is 0/1

6.29.3.14 get_time()

```
float whistory::get_time ( )  [private]
```

returns how long it takes to do something

6.29.3.15 hot2()

```
void whistory::hot2 (
             float * color,
             long unsigned val,
             long unsigned min,
             long unsigned max )
```

creates a hot2 color map

Parameters

in	*color*	- rgb colors, float[3]
in	*val*	- value
in	*min,max*	- values used to normalize the color

6.29.3.16 init()

```
void whistory::init ( )
```

initialization function

initializes OptiX stuff and CUDA stuff, allocates device data, creates host data arrays, initializes counters to zero, copies outermost cell dimensions and isotope list, maps edge array, initializes host values, initializes the random number generator and CUDPP, loads cross sections, and copies data to the compute device.

6.29.3.17 init_cross_sections()

```
void whistory::init_cross_sections ( )  [private]
```

loads cross sections

makes isotope list, initializes the cross section libraries, reads the cross section tables, unionizes the main energy grid across all isotopes, makes the total MT reaction list from all isotopes, allocates the unionized array, inserts and interpolates the cross sections, gets the MT array buffer, gets and copies the unionized MT array, gets the unionized main energy grid buffer, gets the MT number vector, gets the MT number total vector, gets the lengths vector, gets the AWR vector, gets the Q vector. does scattering stuff and energy stuff. passes information to the geometry.

6.29.3.18 init_CUDPP()

```
void whistory::init_CUDPP ( )  [private]
```

initializes CUDPP

initializes global objects, sorting stuff, int reduction stuff, float reduction stuff, int scan stuff, radix sort stuff.

6.29.3.19 init_device()

```
void whistory::init_device ( )  [private]
```

initializes data on the device

prepares data arrays by filling them with zeros

6.29.3.20 init_host()

```
void whistory::init_host ( )  [private]
```

initializes data on the host

prepares data arrays by filling them with zeros

6.29.3.21 init_python()

```
int whistory::init_python ( )  [private]
```

initialize cross section data object in python

6.29.3.22 init_RNG()

```
void whistory::init_RNG ( )  [private]
```

initializes the random number generator

6.29.3.23 make_color()

```
void whistory::make_color (
             float * color,
             unsigned x,
             unsigned min,
             unsigned max )
```

creates a color map

Parameters

in	*color*	- rgb colors, float[3]
in	*x*	- value
in	*min,max*	- values used to normalize the color

6.29.3.24 map_active()

```
unsigned whistory::map_active ( )  [private]
```

maps done histories

flips done flag, remaps data to active histories, flips done flag back.

Returns

num_active - number of active histories

6.29.3.25 nonzero()

```
void whistory::nonzero (
            float * color,
            unsigned val,
            unsigned min,
            unsigned max )
```

creates a binary colormap, black iff 0

Parameters

in	*color*	- rgb colors, float[3]
in	*val*	- value
in	*min,max*	- values used to normalize the color; unused, only present to keep arguments the same as other colormaps

6.29.3.26 plot_geom()

```
void whistory::plot_geom (
            std::string type )
```

produces png images of the geometry, named filename-[xy,xz,yz].png

Parameters

in	*type*	- type of plot, ie color is based 'cell' or 'material'

6.29.3.27 prep_secondaries()

```
void whistory::prep_secondaries ( )  [private]
```

prepares for secondary neutrons

scans yields to determine where threads write into the done data, compacts done data to know where to write.

6.29.3.28 print_banner()

```
void whistory::print_banner ( )
```

prints an amazing and beautiful WARP banner to stdout

6.29.3.29 print_materials_table()

```
void whistory::print_materials_table ( )
```

prints table of properties of geometry materials

6.29.3.30 print_pointers()

```
void whistory::print_pointers ( )
```

prints pointer information

6.29.3.31 print_xs_data()

```
void whistory::print_xs_data ( )
```

prints cross section data information

6.29.3.32 reduce_done()

```
unsigned whistory::reduce_done ( )  [private]
```

reduces done values

Returns

reduced_done - number of done values

6.29.3.33 reduce_weight()

```
float whistory::reduce_weight ( )  [private]
```

reduces weight values

Returns

total

6.29.3.34 reduce_yield()

```
unsigned whistory::reduce_yield ( )  [private]
```

reduces yield values

Returns

total

6.29.3.35 remap_active()

```
void whistory::remap_active (
            unsigned * num_active,
            unsigned * lscatter_N,
            unsigned * lscatter_start,
            unsigned * mscatter_N,
            unsigned * mscatter_start,
            unsigned * cscatter_N,
            unsigned * cscatter_start,
            unsigned * fission_N,
            unsigned * fission_start )  [private]
```

remaps active histories

sorts reaction values, launches edge detection kernels, calculates values for reaction blocks, calculates the total number of active histories, and rezeros the edge vector.

Parameters

in	*num_active*	- number of active histories
in	*lscatter_N*	- number of elastic scatters
in	*lscatter_start*	- elastic scattering start
in	*mscatter_N*	- number of inelastic scatters
in	*mscatter_start*	- inelastic scattering start
in	*cscatter_N*	- number of compound scatters
in	*cscatter_start*	- compound scattering start
in	*fission_N*	- number of fissions
in	*fission_start*	- fission start

6.29.3.36 reset_cycle()

```
void whistory::reset_cycle (
            float keff_cycle )  [private]
```

resets the cycle in criticality mode

rebases the yield so that keff is 1, scans the yield to see where to write, sorts the reaction vector, populates the source, resets the run arrays, and updates the random numbers.

Parameters

in	*keff_cycle*	- keff value of previous cycle

6.29.3.37 reset_fixed()

```
void whistory::reset_fixed ( )  [private]
```

resets the cycle in fixed-source mode

resets the read-in run arrays, samples the fixed source, and updates the random numbers.

6.29.3.38 run()

```
void whistory::run ( )
```

runs history

initializes run variables, clears fissile points file, opens run stats file, records stats. finds the material and nearest surfact distance, finds the main energy grid index, finds interaction length, computes spectra, finds reaction type, remaps threads, does scattering reactions, reduces the yield, resets the cycle, recalculates the running average, and prints the total transport runtime.

6.29.3.39 sample_fissile_points()

```
void whistory::sample_fissile_points ( )  [private]
```

samples fissile points

updates the random numbers, sets uniformly random positions, runs OptiX to get the cell number, compacts data, copies the data back, copies new values in, writes starting positions to file, and updates the random numbers.

6.29.3.40 set_acceration()

```
void whistory::set_acceration (
             std::string accel_in )
```

does nothing

Parameters

in	*accel↩ _in*	- acceleration type

6.29.3.41 set_device()

```
void whistory::set_device (
             unsigned dev_in )
```

sets device number to input value

Parameters

in	*dev↩ _in*	- device number

6.29.3.42 set_dump_level()

```
void whistory::set_dump_level (
             unsigned level )
```

sets what types of information are dumped to files

Parameters

in	*level*	- dump type flag

6.29.3.43 set_filename()

```
void whistory::set_filename (
             std::string filename_in )
```

sets filename to input string

Parameters

in	*filename_in*	- filename

6.29.3.44 set_print_level()

```
void whistory::set_print_level (
             unsigned level )
```

sets amount of information printed to stdout

Parameters

in	*level*	- level of verbosity

6.29.3.45 set_run_param()

```
void whistory::set_run_param (
             unsigned n_cycles_in,
             unsigned n_skip_in )
```

sets number of cycles to skip and number of active cycles

Parameters

in	*n_cycles_in*	- number of active cycles
in	*n_skip_in*	- number of cycles to skip

6.29.3.46 set_run_type() [1/2]

```
void whistory::set_run_type (
            unsigned type_in )
```

sets run type

Parameters

in	*type← _in*	- run type

6.29.3.47 set_run_type() [2/2]

```
void whistory::set_run_type (
            std::string type_in )
```

sets run type

Parameters

in	*type← _in*	- run type

6.29.3.48 trace() [1/2]

```
void whistory::trace (
            unsigned type )  [private]
```

does an OptiX trace

Parameters

in	*type*	- trace type

6.29.3.49 trace() [2/2]

```
void whistory::trace (
            unsigned type,
            unsigned n_active )  [private]
```

does an OptiX trace for a given number of active histories

Parameters

in	*type*	- trace type
in	*n_active*	- number of active histories

6.29.3.50 update_RNG()

```
void whistory::update_RNG ( )  [private]
```

updates the random number

6.29.3.51 write_fission_points()

```
void whistory::write_fission_points ( )
```

writes binned fission point image to a .png

6.29.3.52 write_results()

```
void whistory::write_results (
             float runtime,
             float keff,
             std::string opentype )  [private]
```

writes results to file

Parameters

in	*runtime*	- runtime
in	*keff*	- keff
in	*opentype*	- file extension

6.29.3.53 write_tally()

```
void whistory::write_tally ( )
```

writes all tally values to file

6.29.3.54 write_to_file() [1/4]

```
void whistory::write_to_file (
             spatial_data * array_in,
             unsigned N,
             std::string filename,
             std::string opentype )  [private]
```

prints the locations of the source points to file

Parameters

in	*array_in*	- source point array
in	*N*	- number of histories
in	*filename*	- filename
in	*opentype*	- file extension

6.29.3.55 write_to_file() [2/4]

```
void whistory::write_to_file (
             spatial_data * array_in,
             float * array_in2,
             unsigned N,
             std::string filename,
             std::string opentype )  [private]
```

prints the locations of the source points to file

Parameters

in	*array_in*	- source point array
in	*array_in2*	- second array
in	*N*	- number of histories
in	*filename*	- filename
in	*opentype*	- file extension

6.29.3.56 write_to_file() [3/4]

```
void whistory::write_to_file (
             unsigned * array_in,
             unsigned N,
             std::string filename,
             std::string opentype )  [private]
```

prints the source points to a file

Parameters

in	*array_in*	- source point array
in	*N*	- number of histories
in	*filename*	- filename
in	*opentype*	- file extension

6.29.3.57 write_to_file() [4/4]

```
void whistory::write_to_file (
            unsigned * array_in,
            unsigned * array_in2,
            unsigned N,
            std::string filename,
            std::string opentype )  [private]
```

prints the locations of the source points to file

Parameters

in	*array_in*	- source point array
in	*array_in2*	- second array
in	*N*	- number of histories
in	*filename*	- filename
in	*opentype*	- file extension

6.29.3.58 write_xs_data()

```
void whistory::write_xs_data (
            std::string filename )
```

writes cross section data to file

Parameters

in	*filename*	- filename

6.29.4 Member Data Documentation

6.29.4.1 accel_type

```
std::string whistory::accel_type  [private]
```

acceleration type

6.29.4.2 blks

```
unsigned whistory::blks  [private]
```

number of blocks

6.29.4.3 compact_config

```
CUDPPConfiguration whistory::compact_config  [private]
```

CUDPP compact configuration

6.29.4.4 compactplan

```
CUDPPHandle whistory::compactplan  [private]
```

CUDPP compact plan handle

6.29.4.5 compute_device

```
unsigned whistory::compute_device  [private]
```

compute device

6.29.4.6 d_bank_E

```
float* whistory::d_bank_E  [private]
```

device bank energy

6.29.4.7 d_bank_space

```
spatial_data* whistory::d_bank_space  [private]
```

device bank space

6.29.4.8 d_edges

```
unsigned* whistory::d_edges  [private]
```

device mapped array of edges

6.29.4.9 d_fissile_energy

```
float* whistory::d_fissile_energy  [private]
```

device fissile energy

6.29.4.10 d_fissile_points

```
spatial_data* whistory::d_fissile_points  [private]
```

device fissile points

6.29.4.11 d_isotope_list

```
unsigned* whistory::d_isotope_list  [private]
```

device isotope list

6.29.4.12 d_material_list

```
unsigned* whistory::d_material_list  [private]
```

device material list

6.29.4.13 d_num_active

```
unsigned* whistory::d_num_active  [private]
```

device number of active histories

6.29.4.14 d_num_completed

```
unsigned* whistory::d_num_completed  [private]
```

device number of completed histories

6.29.4.15 d_number_density_matrix

```
float* whistory::d_number_density_matrix  [private]
```

device isotope number density matrix

6.29.4.16 d_particles

```
particle_data* whistory::d_particles  [private]
```

device particle data structure containing device pointers

6.29.4.17 d_reduced_weight

```
float* whistory::d_reduced_weight  [private]
```

device reduced weight

6.29.4.18 d_reduced_yields

```
unsigned* whistory::d_reduced_yields  [private]
```

device reduced yields

6.29.4.19 d_remap

```
unsigned* whistory::d_remap [private]
```

remap pointer

6.29.4.20 d_scanned

```
unsigned* whistory::d_scanned [private]
```

device scanned pointer

6.29.4.21 d_tally

```
tally_data* whistory::d_tally [private]
```

device tally data structure containing device pointers

6.29.4.22 d_valid_N

```
unsigned* whistory::d_valid_N [private]
```

valied number of histories pointer

6.29.4.23 d_valid_result

```
unsigned* whistory::d_valid_result [private]
```

valid result pointer

6.29.4.24 d_xsdata

```
cross_section_data* whistory::d_xsdata [private]
```

device cross section data structure containing device pointers

6.29.4.25 d_zeros

```
unsigned* whistory::d_zeros [private]
```

device zeros array

6.29.4.26 dh_dist_energy

```
dist_container* whistory::dh_dist_energy [private]
```

6.29.4.27 dh_dist_scatter

`dist_container* whistory::dh_dist_scatter [private]`

6.29.4.28 dh_particles

`particle_data whistory::dh_particles [private]`

host particle data structure containing device pointers

6.29.4.29 dh_tally

`tally_data* whistory::dh_tally [private]`

host tally data structure containing device pointers

6.29.4.30 dh_xsdata

`cross_section_data whistory::dh_xsdata [private]`

host cross section data structure containing device pointers

6.29.4.31 dump_flag

`unsigned whistory::dump_flag [private]`

dump level

6.29.4.32 edges

`unsigned* whistory::edges [private]`

mapped array of edges

6.29.4.33 filename

`std::string whistory::filename [private]`

file name

6.29.4.34 fiss_img

`long unsigned* whistory::fiss_img [private]`

fissile image accumulation

6.29.4.35 fones

```
float* whistory::fones  [private]
```

float ones array

6.29.4.36 h_particles

```
particle_data whistory::h_particles  [private]
```

host particle data structure containing host pointers

6.29.4.37 h_tally

```
tally_data_host* whistory::h_tally  [private]
```

host tally data structure containing host pointers (has long values for accumulation)

6.29.4.38 h_xsdata

```
cross_section_data whistory::h_xsdata  [private]
```

host cross section data structure containing host pointers

6.29.4.39 initial_weight_total

```
double whistory::initial_weight_total  [private]
```

double for accumulating the starting weight numbers accurately on the host

6.29.4.40 is_initialized

```
unsigned whistory::is_initialized  [private]
```

init flag

6.29.4.41 isotope_list

```
unsigned* whistory::isotope_list  [private]
```

isotope list

6.29.4.42 isotopes

```
std::vector<std::string> whistory::isotopes  [private]
```

cross section isotope string

6.29.4.43 keff2_sum

```
float whistory::keff2_sum  [private]
```

keff squared sum

6.29.4.44 keff_err

```
float whistory::keff_err  [private]
```

keff error

6.29.4.45 keff_sum

```
float whistory::keff_sum  [private]
```

keff sum

6.29.4.46 material_list

```
unsigned* whistory::material_list  [private]
```

material list

6.29.4.47 mem_free

```
size_t whistory::mem_free  [private]
```

device memory free

6.29.4.48 mem_total

```
size_t whistory::mem_total  [private]
```

device memory total

6.29.4.49 mones

```
int* whistory::mones  [private]
```

int negative ones array

6.29.4.50 MT_columns

```
unsigned whistory::MT_columns  [private]
```

MT number columns

6.29.4.51 MT_rows

```
unsigned whistory::MT_rows  [private]
```

MT number rows

6.29.4.52 N

```
unsigned whistory::N  [private]
```

number of histories

6.29.4.53 n_cycles

```
unsigned whistory::n_cycles  [private]
```

number of active cycles

6.29.4.54 n_edges

```
unsigned whistory::n_edges  [private]
```

mapped array of number of edges

6.29.4.55 n_isotopes

```
unsigned whistory::n_isotopes  [private]
```

number of isotopes

6.29.4.56 n_materials

```
unsigned whistory::n_materials  [private]
```

number of materials

6.29.4.57 n_skip

```
unsigned whistory::n_skip  [private]
```

number of cycles to skip

6.29.4.58 n_tallies

```
unsigned whistory::n_tallies  [private]
```

number of tallies

6.29.4.59 Ndataset

```
unsigned whistory::Ndataset [private]
```

dataset size for number of histories

6.29.4.60 NUM_THREADS

```
unsigned whistory::NUM_THREADS [private]
```

number of threads per block

6.29.4.61 number_density_matrix

```
float* whistory::number_density_matrix [private]
```

isotope number density matrix

6.29.4.62 ones

```
unsigned* whistory::ones [private]
```

int ones array

6.29.4.63 outer_cell

```
unsigned whistory::outer_cell [private]
```

outermost cell

6.29.4.64 outer_cell_dims

```
float whistory::outer_cell_dims[6] [private]
```

outer cell minima and maxima

6.29.4.65 outer_cell_type

```
unsigned whistory::outer_cell_type [private]
```

outermost cell type

6.29.4.66 print_flag

```
unsigned whistory::print_flag [private]
```

print verbosity level

6.29.4.67 problem_geom

```
wgeometry whistory::problem_geom  [private]
```

problem geometry

6.29.4.68 radix_config

```
CUDPPConfiguration whistory::radix_config  [private]
```

CUDPP radix configuration

6.29.4.69 radixplan

```
CUDPPHandle whistory::radixplan  [private]
```

CUDPP radix plan handle

6.29.4.70 rand_gen

```
curandGenerator_t whistory::rand_gen  [private]
```

random number generator handle

6.29.4.71 redu_float_config

```
CUDPPConfiguration whistory::redu_float_config  [private]
```

CUDPP reduced float configuration

6.29.4.72 redu_int_config

```
CUDPPConfiguration whistory::redu_int_config  [private]
```

CUDPP reduced int configuration

6.29.4.73 reduced_weight

```
float* whistory::reduced_weight  [private]
```

reduced weight

6.29.4.74 reduced_weight_total

```
double whistory::reduced_weight_total  [private]
```

double for accumulating weight numbers accurately on the host

6.29.4.75 reduced_yields

```
unsigned* whistory::reduced_yields  [private]
```

reduced yields

6.29.4.76 reduced_yields_total

```
long unsigned whistory::reduced_yields_total  [private]
```

long unsigned for accumulating yield numbers accurately on the host

6.29.4.77 reduplan_float

```
CUDPPHandle whistory::reduplan_float  [private]
```

CUDPP reduce plan float handle

6.29.4.78 reduplan_int

```
CUDPPHandle whistory::reduplan_int  [private]
```

CUDPP reduce plan int handle

6.29.4.79 remap

```
unsigned* whistory::remap  [private]
```

remap

6.29.4.80 res

```
CUDPPResult whistory::res  [private]
```

CUDPP result

6.29.4.81 RUN_FLAG

```
unsigned whistory::RUN_FLAG  [private]
```

run flag

6.29.4.82 scan_int_config

```
CUDPPConfiguration whistory::scan_int_config  [private]
```

CUDPP scan int configuration

6.29.4.83 scanplan_int

```
CUDPPHandle whistory::scanplan_int  [private]
```

CUDPP scan plan int handle

6.29.4.84 stream

```
cudaStream_t whistory::stream[5]  [private]
```

CUDA streams cor concurrent kernels

6.29.4.85 theCudpp

```
CUDPPHandle whistory::theCudpp  [private]
```

CUDPP handle

6.29.4.86 total_bytes_energy

```
unsigned whistory::total_bytes_energy  [private]
```

total size of energy data

6.29.4.87 total_bytes_scatter

```
unsigned whistory::total_bytes_scatter  [private]
```

total size of scattering data

6.29.4.88 xs_isotope_ints

```
std::vector<unsigned> whistory::xs_isotope_ints  [private]
```

cross section isotope numbers

6.29.4.89 xs_num_rxns

```
std::vector<unsigned> whistory::xs_num_rxns  [private]
```

cross section number of reactions

6.29.4.90 xsdat_instance

```
PyObject* whistory::xsdat_instance  [private]
```

Python object that loads and manipulates the cross section data

6.29.4.91 zeros

```
unsigned* whistory::zeros  [private]
```

zeros array

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

- whistory.h
- whistory.cpp

6.30 wtransform Struct Reference

contains parameters of a wtransform, a transform used to create a new instance of a wgemetry object

```
#include <datadef.h>
```

Public Attributes

- unsigned cellnum
- unsigned cellmat
- int tally_index
- float dx
- float dy
- float dz
- float theta
- float phi

6.30.1 Detailed Description

contains parameters of a wtransform, a transform used to create a new instance of a wgemetry object

cell number and material, transform coordinates and angles

6.30.2 Member Data Documentation

6.30.2.1 cellmat

```
unsigned wtransform::cellmat
```

cell material

6.30.2.2 cellnum

```
unsigned wtransform::cellnum
```

cell number

6.30.2.3 dx

```
float wtransform::dx
```

displacement in x

6.30.2.4 dy

```
float wtransform::dy
```

displacement in y

6.30.2.5 dz

```
float wtransform::dz
```

displacement in z

6.30.2.6 phi

```
float wtransform::phi
```

roation in azimuthal angle

6.30.2.7 tally_index

```
int wtransform::tally_index
```

index of tally associated with this instance

6.30.2.8 theta

```
float wtransform::theta
```

roation in polar angle

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

- datadef.h

Chapter 7

File Documentation

7.1 ace.pyx File Reference

Classes

- class ace.Library
- class ace.AceTable
- class ace.NeutronTable
- class ace.EnergyDistribution
- class ace.SabTable
- class ace.Reaction
- class ace.DosimetryTable
- class ace.NeutronDiscreteTable
- class ace.NeutronMGTable
- class ace.PhotoatomicTable
- class ace.PhotoatomicMGTable
- class ace.ElectronTable
- class ace.PhotonuclearTable

Namespaces

- ace

Functions

- def ace.fromstring_split (s, sep=None, dtype=float)
- def ace.fromstring_token (s, sep=" ", bint, inplace=False, int, maxsize=-1)
- def ace.ascii_to_binary (ascii_file, binary_file)

Variables

- ace.NP_LE_V15
- dictionary ace.table_types

7.2 benchmarks.cpp File Reference

```
#include "warp.h"
```

Functions

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

7.2.1 Function Documentation

7.2.1.1 main()

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

7.3 box_mesh.cu File Reference

```
#include <optix.h>
#include <optixu/optixu_math_namespace.h>
#include <optixu/optixu_matrix_namespace.h>
#include <optixu/optixu_aabb_namespace.h>
#include "datadef.h"
```

Functions

- rtDeclareVariable (optix::Ray, ray, rtCurrentRay,)
- rtDeclareVariable (unsigned, cellnum, attribute cell_num,)
- rtDeclareVariable (int, celltal, attribute cell_tal,)
- rtDeclareVariable (unsigned, cellmat, attribute cell_mat,)
- rtDeclareVariable (unsigned, cellfissile, attribute cell_fis,)
- rtDeclareVariable (unsigned, sense, attribute cell_sense,)
- rtDeclareVariable (float3, normal, attribute normal,)
- static __device__ float3 boxnormal (float t, float3 t0, float3 t1)
- RT_PROGRAM void intersect (int object_dex)
- RT_PROGRAM void bounds (int object_dex, float result[6])

Variables

- rtBuffer< geom_data, 1 > dims

7.3.1 Function Documentation

7.3.1.1 bounds()

```
RT_PROGRAM void bounds (
             int object_dex,
             float result[6] )
```

7.3.1.2 boxnormal()

```
static __device__ float3 boxnormal (
             float t,
             float3 t0,
             float3 t1 )  [static]
```

7.3.1.3 intersect()

```
RT_PROGRAM void intersect (
             int object_dex )
```

7.3.1.4 rtDeclareVariable() [1/7]

```
rtDeclareVariable (
             optix::Ray ,
             ray ,
             rtCurrentRay  )
```

7.3.1.5 rtDeclareVariable() [2/7]

```
rtDeclareVariable (
             unsigned ,
             cellnum ,
             attribute cell_num )
```

7.3.1.6 rtDeclareVariable() [3/7]

```
rtDeclareVariable (
             int ,
             celltal ,
             attribute cell_tal )
```

7.3.1.7 rtDeclareVariable() [4/7]

```
rtDeclareVariable (
             unsigned ,
             cellmat ,
             attribute cell_mat )
```

7.3.1.8 rtDeclareVariable() [5/7]

```
rtDeclareVariable (
             unsigned ,
             cellfissile ,
             attribute cell_fis )
```

7.3.1.9 rtDeclareVariable() [6/7]

```
rtDeclareVariable (
             unsigned ,
             sense ,
             attribute cell_sense )
```

7.3.1.10 rtDeclareVariable() [7/7]

```
rtDeclareVariable (
             float3 ,
             normal ,
             attribute normal )
```

7.3.2 Variable Documentation

7.3.2.1 dims

```
rtBuffer<geom_data,1> dims
```

7.4 camera.cu File Reference

```
#include "optix.h"
#include <optix_world.h>
#include "datadef.h"
```

Functions

- rtDeclareVariable (rtObject, top_object,,)
- rtDeclareVariable (uint, launch_index_in, rtLaunchIndex,)
- rtDeclareVariable (uint, launch_dim, rtLaunchDim,)
- rtDeclareVariable (unsigned, outer_cell,,)
- rtDeclareVariable (unsigned, trace_type,,)
- rtDeclareVariable (unsigned, boundary_condition,,)
- RT_PROGRAM void camera ()
- RT_PROGRAM void exception ()

Variables

- rtBuffer< spatial_data, 1 > positions_buffer
- rtBuffer< unsigned, 1 > rxn_buffer
- rtBuffer< unsigned, 1 > remap_buffer
- rtBuffer< unsigned, 1 > cellnum_buffer
- rtBuffer< unsigned, 1 > matnum_buffer
- rtBuffer< unsigned, 1 > talnum_buffer

7.4.1 Function Documentation

7.4.1.1 camera()

```
RT_PROGRAM void camera ( )
```

7.4.1.2 exception()

```
RT_PROGRAM void exception ( )
```

7.4.1.3 rtDeclareVariable() [1/6]

```
rtDeclareVariable (
             rtObject ,
             top_object  )
```

7.4.1.4 rtDeclareVariable() [2/6]

```
rtDeclareVariable (
             uint ,
             launch_index_in ,
             rtLaunchIndex  )
```

7.4.1.5 rtDeclareVariable() [3/6]

```
rtDeclareVariable (
             uint ,
             launch_dim ,
             rtLaunchDim  )
```

7.4.1.6 rtDeclareVariable() [4/6]

```
rtDeclareVariable (
             unsigned ,
             outer_cell  )
```

7.4.1.7 rtDeclareVariable() [5/6]

```
rtDeclareVariable (
             unsigned ,
             trace_type  )
```

7.4.1.8 rtDeclareVariable() [6/6]

```
rtDeclareVariable (
             unsigned ,
             boundary_condition  )
```

7.4.2 Variable Documentation

7.4.2.1 cellnum_buffer

```
rtBuffer<unsigned,1> cellnum_buffer
```

7.4.2.2 matnum_buffer

```
rtBuffer<unsigned,1> matnum_buffer
```

7.4.2.3 positions_buffer

```
rtBuffer<spatial_data,1> positions_buffer
```

7.4.2.4 remap_buffer

```
rtBuffer<unsigned,1> remap_buffer
```

7.4.2.5 rxn_buffer

```
rtBuffer<unsigned,1> rxn_buffer
```

7.4.2.6 talnum_buffer

```
rtBuffer<unsigned,1> talnum_buffer
```

7.5 check_cuda.h File Reference

Macros

- #define check_cuda(ans) { check_cuda((ans), __FILE__, __LINE__); }

Functions

- __host__ void check_cuda (cudaError_t code, const char ∗file, int line, bool abort=true)

  CUDA error check wrapper, host only.

7.5.1 Macro Definition Documentation

7.5.1.1 check_cuda

```
#define check_cuda(
              ans ) { check_cuda((ans), __FILE__, __LINE__); }
```

7.5.2 Function Documentation

7.5.2.1 check_cuda()

```
__host__ void check_cuda (
             cudaError_t code,
             const char * file,
             int line,
             bool abort = true )  [inline]
```

CUDA error check wrapper, host only.

this inline function prints detailed information about the return code of host-side CUDA functions and where they occur in the code

Parameters

in	*code*	- CUDA error code output from some host-side CUDA function call
in	*file*	- file where erroring fuction is, written inline by preprocessor
in	*line*	- line of file where erroring fuction is, written inline by preprocessor
in	*abort*	- flag to exit on error, default is true

7.6 check_pointers.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "warp_device.cuh"
#include "check_cuda.h"
```

```
#include "wfloat3.h"
```

Functions

- __global__ void check_pointers_kernel (unsigned N, unsigned dex0, unsigned dex1, cross_section_data ∗d_xsdata)
- void check_pointers (unsigned NUM_THREADS, unsigned dex0, unsigned dex1, cross_section_data ∗d_↩ xsdata)

  a

7.6.1 Function Documentation

7.6.1.1 check_pointers()

```
void check_pointers (
             unsigned NUM_THREADS,
             unsigned dex0,
             unsigned dex1,
             cross_section_data * d_xsdata )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*dex0*	- starting index
in	*dex1*	- ending index
in	*d_xsdata*	- device pointer to cross section data pointer array

7.6.1.2 check_pointers_kernel()

```
__global__ void check_pointers_kernel (
             unsigned N,
             unsigned dex0,
             unsigned dex1,
             cross_section_data * d_xsdata )
```

7.7 copy_points.cu File Reference

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

```
#include "datadef.h"
```

Functions

- __global__ void copy_points_kernel (unsigned Nout, unsigned ∗Nvalid, unsigned current_index, unsigned ∗to_valid, spatial_data ∗positions_out, spatial_data ∗positions_in, float ∗E_out, float ∗E_in)
- void copy_points (unsigned NUM_THREADS, unsigned Nout, unsigned ∗Nvalid, unsigned current_index, unsigned ∗to_valid, spatial_data ∗positions_out, spatial_data ∗positions_in, float ∗E_out, float ∗E_in)

  copy points between two sets of space and energy data buffers, redirected with a mapping array

7.7.1 Function Documentation

7.7.1.1 copy_points()

```
void copy_points (
            unsigned NUM_THREADS,
            unsigned Nout,
            unsigned * Nvalid,
            unsigned current_index,
            unsigned * to_valid,
            spatial_data * positions_out,
            spatial_data * positions_in,
            float * E_out,
            float * E_in )
```

copy points between two sets of space and energy data buffers, redirected with a mapping array

copy points between two sets of space and energy data buffers, redirected with a mapping array

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*Nout*	- the total number of threads to launch on the grid
in	*Nvalid*	- the total number of device elements to copy from
in	*current_index*	- starting index
in	*to_valid*	- device pointer to data remapping vector
in	*positions_out*	- device pointer to spatial data array destination
in	*positions_in*	- device pointer to spatial data array source
in	*E_out*	- device pointer to energy data array destination
in	*E_in*	- device pointer to energy data array source

7.7.1.2 copy_points_kernel()

```
__global__ void copy_points_kernel (
            unsigned Nout,
            unsigned * Nvalid,
            unsigned current_index,
            unsigned * to_valid,
            spatial_data * positions_out,
            spatial_data * positions_in,
            float * E_out,
            float * E_in )
```

7.8 cylinder_mesh.cu File Reference

```
#include <optix.h>
#include <optixu/optixu_math_namespace.h>
#include <optixu/optixu_matrix_namespace.h>
#include <optixu/optixu_aabb_namespace.h>
#include "datadef.h"
```

Functions

- rtDeclareVariable (optix::Ray, ray, rtCurrentRay,)
- rtDeclareVariable (unsigned, cellnum, attribute cell_num,)
- rtDeclareVariable (int, celltal, attribute cell_tal,)
- rtDeclareVariable (unsigned, cellmat, attribute cell_mat,)
- rtDeclareVariable (unsigned, cellfissile, attribute cell_fis,)
- rtDeclareVariable (unsigned, sense, attribute cell_sense,)
- rtDeclareVariable (float3, normal, attribute normal,)
- RT_PROGRAM void intersect (int object_dex)
- RT_PROGRAM void bounds (int object_dex, float result[6])

Variables

- rtBuffer< geom_data, 1 > dims

7.8.1 Function Documentation

7.8.1.1 bounds()

```
RT_PROGRAM void bounds (
            int object_dex,
            float result[6] )
```

7.8.1.2 intersect()

```
RT_PROGRAM void intersect (
             int object_dex )
```

7.8.1.3 rtDeclareVariable() [1/7]

```
rtDeclareVariable (
             optix::Ray ,
             ray ,
             rtCurrentRay  )
```

7.8.1.4 rtDeclareVariable() [2/7]

```
rtDeclareVariable (
             unsigned ,
             cellnum ,
             attribute cell_num )
```

7.8.1.5 rtDeclareVariable() [3/7]

```
rtDeclareVariable (
             int ,
             celltal ,
             attribute cell_tal )
```

7.8.1.6 rtDeclareVariable() [4/7]

```
rtDeclareVariable (
             unsigned ,
             cellmat ,
             attribute cell_mat )
```

7.8.1.7 rtDeclareVariable() [5/7]

```
rtDeclareVariable (
             unsigned ,
             cellfissile ,
             attribute cell_fis )
```

7.8.1.8 rtDeclareVariable() [6/7]

```
rtDeclareVariable (
            unsigned ,
            sense ,
            attribute cell_sense )
```

7.8.1.9 rtDeclareVariable() [7/7]

```
rtDeclareVariable (
            float3 ,
            normal ,
            attribute normal )
```

7.8.2 Variable Documentation

7.8.2.1 dims

```
rtBuffer<geom_data,1> dims
```

7.9 datadef.h File Reference

```
#include <vector>
#include <string>
```

Classes

- struct wtransform

  contains parameters of a wtransform, a transform used to create a new instance of a wgemetry object
- struct geom_data

  contains parameters of a wgeometry
- struct spatial_data

  contains the spatial parameters of the neutron
- struct intersection_point

  contains information pertinent to an intersection point, used in OptiX
- struct material_def

  contains information that defines a material
- struct dist_data

  contains information that defines an ENDF cross section distribution
- struct dist_container

  container for pointers that map the nearest distributions to the energy grid point where it resides
- struct cross_section_data

  structure that holds the topmost level of cross section data
- struct particle_data

  structure that holds all the arrays that define a particle's state
- struct tally_data_host

  Tally data that lives on the host side.
- struct tally_data

  Tally data that lives on the device side.

7.10 device_copies.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
```

Functions

- void copy_to_device (void ∗dest, void ∗source, unsigned bytes)

    function to do a host-to-device copy
- void copy_from_device (void ∗dest, void ∗source, unsigned bytes)

    function to do a device-to-host copy
- void allocate_on_device (void ∗∗dest, unsigned bytes)

    function to do a device memory allocation
- void deallocate_on_device (void ∗dest)

    function to do a device memory allocation

7.10.1 Function Documentation

7.10.1.1 allocate_on_device()

```
void allocate_on_device (
            void ** dest,
            unsigned bytes )
```

function to do a device memory allocation

Parameters

in	*dest*	- pointer on host
in	*bytes*	- number of bytes to copy

7.10.1.2 copy_from_device()

```
void copy_from_device (
            void * dest,
            void * source,
            unsigned bytes )
```

function to do a device-to-host copy

Parameters

in	*dest*	- pointer on host
in	*source*	- cuda pointer on device
in	*bytes*	- number of bytes to copy

7.10.1.3 copy_to_device()

```
void copy_to_device (
            void * dest,
            void * source,
            unsigned bytes )
```

function to do a host-to-device copy

something

Parameters

in	*dest*	- cuda pointer on device
in	*source*	- pointer on host
in	*bytes*	- number of bytes to copy

7.10.1.4 deallocate_on_device()

```
void deallocate_on_device (
            void * dest )
```

function to do a device memory allocation

Parameters

in	*dest*	- pointer on device to deallocate

7.11 device_copies.h File Reference

Functions

- void copy_to_device (void ∗, void ∗, unsigned)

  function to do a host-to-device copy
- void copy_from_device (void ∗, void ∗, unsigned)

  function to do a device-to-host copy

- void allocate_on_device (void ∗∗, unsigned)

  function to do a device memory allocation
- void deallocate_on_device (void ∗)

  function to do a device memory allocation

7.11.1 Function Documentation

7.11.1.1 allocate_on_device()

```
void allocate_on_device (
            void ** dest,
            unsigned bytes )
```

function to do a device memory allocation

Parameters

in	*dest*	- pointer on host
in	*bytes*	- number of bytes to copy

7.11.1.2 copy_from_device()

```
void copy_from_device (
            void * dest,
            void * source,
            unsigned bytes )
```

function to do a device-to-host copy

Parameters

in	*dest*	- pointer on host
in	*source*	- cuda pointer on device
in	*bytes*	- number of bytes to copy

7.11.1.3 copy_to_device()

```
void copy_to_device (
            void * dest,
            void * source,
            unsigned bytes )
```

function to do a host-to-device copy

something

Parameters

in	*dest*	- cuda pointer on device
in	*source*	- pointer on host
in	*bytes*	- number of bytes to copy

7.11.1.4 deallocate_on_device()

```
void deallocate_on_device (
            void * dest )
```

function to do a device memory allocation

Parameters

in	*dest*	- pointer on device to deallocate

7.12 find_E_grid_index.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
```

Functions

- __global__ void find_E_grid_index_kernel (unsigned N, cross_section_data *d_xsdata, unsigned *remap, float *E, unsigned *index, unsigned *rxn)
- void find_E_grid_index (unsigned NUM_THREADS, unsigned N, cross_section_data *d_xsdata, unsigned *d_remap, float *d_E, unsigned *d_index, unsigned *d_rxn)

  a

7.12.1 Function Documentation

7.12.1.1 find_E_grid_index()

```
void find_E_grid_index (
            unsigned NUM_THREADS,
            unsigned N,
            cross_section_data * d_xsdata,
            unsigned * d_remap,
            float * d_E,
            unsigned * d_index,
            unsigned * d_rxn )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_remap*	- device pointer to data remapping vector
in	*d_E*	- device pointer to energy data array
in	*d_index*	- device pointer to index array (stores the unionized grid index of the current energy)
in	*d_rxn*	- device pointer of the reaction number array

7.12.1.2 find_E_grid_index_kernel()

```
__global__ void find_E_grid_index_kernel (
            unsigned N,
            cross_section_data * d_xsdata,
            unsigned * remap,
            float * E,
            unsigned * index,
            unsigned * rxn )
```

7.13 fission.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "wfloat3.h"
#include "warp_device.cuh"
#include "check_cuda.h"
```

Functions

- __global__ void fission_kernel (unsigned N, unsigned starting_index, cross_section_data ∗d_xsdata, particle_data ∗d_particles, unsigned ∗d_remap)
- void fission (cudaStream_t stream, unsigned NUM_THREADS, unsigned N, unsigned starting_index, cross↩_section_data ∗d_xsdata, particle_data ∗d_particles, unsigned ∗d_remap)

  a

7.13.1 Function Documentation

7.13.1.1 fission()

```
void fission (
             cudaStream_t stream,
             unsigned NUM_THREADS,
             unsigned N,
             unsigned starting_index,
             cross_section_data ∗ d_xsdata,
             particle_data ∗ d_particles,
             unsigned ∗ d_remap )
```

a

b

Parameters

in	*stream*	- CUDA stream to launch the kernel on
in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid for fission
in	*starting_index*	- starting index of the fission block in the remap vector
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.13.1.2 fission_kernel()

```
__global__ void fission_kernel (
             unsigned N,
             unsigned starting_index,
             cross_section_data ∗ d_xsdata,
             particle_data ∗ d_particles,
             unsigned ∗ d_remap )
```

7.14 hex_mesh.cu File Reference

```
#include <optix.h>
#include <optixu/optixu_math_namespace.h>
#include <optixu/optixu_matrix_namespace.h>
#include <optixu/optixu_aabb_namespace.h>
#include "datadef.h"
```

Functions

- rtDeclareVariable (optix::Ray, ray, rtCurrentRay,)
- rtDeclareVariable (unsigned, cellnum, attribute cell_num,)
- rtDeclareVariable (int, celltal, attribute cell_tal,)
- rtDeclareVariable (unsigned, cellmat, attribute cell_mat,)
- rtDeclareVariable (unsigned, cellfissile, attribute cell_fis,)
- rtDeclareVariable (unsigned, sense, attribute cell_sense,)
- rtDeclareVariable (float3, normal, attribute normal,)
- rtDeclareVariable (uint, launch_index_in, rtLaunchIndex,)
- static __device__ bool accept_point (float3 pnt, float a, float x1, float x2, float zmin, float zmax)
- static __device__ float get_t (float3 hat, float3 dir, float3 diff_points)
- RT_PROGRAM void intersect (int object_dex)
- RT_PROGRAM void bounds (int object_dex, float result[6])

Variables

- rtBuffer< geom_data, 1 > dims

7.14.1 Function Documentation

7.14.1.1 accept_point()

```
static __device__ bool accept_point (
             float3 pnt,
             float a,
             float x1,
             float x2,
             float zmin,
             float zmax )  [static]
```

7.14.1.2 bounds()

```
RT_PROGRAM void bounds (
             int object_dex,
             float result[6] )
```

7.14.1.3 get_t()

```
static __device__ float get_t (
            float3 hat,
            float3 dir,
            float3 diff_points )  [static]
```

7.14.1.4 intersect()

```
RT_PROGRAM void intersect (
            int object_dex )
```

7.14.1.5 rtDeclareVariable() [1/8]

```
rtDeclareVariable (
            optix::Ray ,
            ray ,
            rtCurrentRay  )
```

7.14.1.6 rtDeclareVariable() [2/8]

```
rtDeclareVariable (
            unsigned ,
            cellnum ,
            attribute cell_num )
```

7.14.1.7 rtDeclareVariable() [3/8]

```
rtDeclareVariable (
            int ,
            celltal ,
            attribute cell_tal )
```

7.14.1.8 rtDeclareVariable() [4/8]

```
rtDeclareVariable (
            unsigned ,
            cellmat ,
            attribute cell_mat )
```

7.14.1.9 rtDeclareVariable() [5/8]

```
rtDeclareVariable (
            unsigned ,
            cellfissile ,
            attribute cell_fis )
```

7.14.1.10 rtDeclareVariable() [6/8]

```
rtDeclareVariable (
            unsigned ,
            sense ,
            attribute cell_sense )
```

7.14.1.11 rtDeclareVariable() [7/8]

```
rtDeclareVariable (
            float3 ,
            normal ,
            attribute normal )
```

7.14.1.12 rtDeclareVariable() [8/8]

```
rtDeclareVariable (
            uint ,
            launch_index_in ,
            rtLaunchIndex  )
```

7.14.2 Variable Documentation

7.14.2.1 dims

```
rtBuffer<geom_data,1> dims
```

7.15 hits_mesh.cu File Reference

```
#include <optix.h>
#include <optixu/optixu_math_namespace.h>
#include "datadef.h"
```

Functions

- rtDeclareVariable (intersection_point, payload, rtPayload,)
- rtDeclareVariable (float, int_dist, rtIntersectionDistance,)
- rtDeclareVariable (optix::Ray, ray, rtCurrentRay,)
- rtDeclareVariable (unsigned, cellnum, attribute cell_num,)
- rtDeclareVariable (int, celltal, attribute cell_tal,)
- rtDeclareVariable (unsigned, cellmat, attribute cell_mat,)
- rtDeclareVariable (unsigned, cellfissile, attribute cell_fis,)
- rtDeclareVariable (unsigned, sense, attribute cell_sense,)
- rtDeclareVariable (float3, normal, attribute normal,)
- RT_PROGRAM void closest_hit ()

7.15.1 Function Documentation

7.15.1.1 closest_hit()

```
RT_PROGRAM void closest_hit ( )
```

7.15.1.2 rtDeclareVariable() [1/9]

```
rtDeclareVariable (
             intersection_point ,
             payload ,
             rtPayload  )
```

7.15.1.3 rtDeclareVariable() [2/9]

```
rtDeclareVariable (
             float ,
             int_dist ,
             rtIntersectionDistance  )
```

7.15.1.4 rtDeclareVariable() [3/9]

```
rtDeclareVariable (
             optix::Ray ,
             ray ,
             rtCurrentRay  )
```

7.15.1.5 rtDeclareVariable() [4/9]

```
rtDeclareVariable (
             unsigned ,
             cellnum ,
             attribute cell_num )
```

7.15.1.6 rtDeclareVariable() [5/9]

```
rtDeclareVariable (
             int ,
             celltal ,
             attribute cell_tal )
```

7.15.1.7 rtDeclareVariable() [6/9]

```
rtDeclareVariable (
             unsigned ,
             cellmat ,
             attribute cell_mat )
```

7.15.1.8 rtDeclareVariable() [7/9]

```
rtDeclareVariable (
             unsigned ,
             cellfissile ,
             attribute cell_fis )
```

7.15.1.9 rtDeclareVariable() [8/9]

```
rtDeclareVariable (
             unsigned ,
             sense ,
             attribute cell_sense )
```

7.15.1.10 rtDeclareVariable() [9/9]

```
rtDeclareVariable (
             float3 ,
             normal ,
             attribute normal )
```

7.16 macro_micro.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "warp_device.cuh"
#include "check_cuda.h"
```

Functions

- __global__ void macro_micro_kernel (unsigned N, unsigned converged, unsigned n_materials, unsigned n_tallies, cross_section_data ∗d_xsdata, particle_data ∗d_particles, tally_data ∗d_tally, unsigned ∗d_remap, float ∗d_number_density_matrix)
- void macro_micro (unsigned NUM_THREADS, unsigned N, unsigned converged, unsigned n_materials, unsigned n_isotopes, unsigned n_tallies, cross_section_data ∗d_xsdata, particle_data ∗d_particles, tally_data ∗d_tally, unsigned ∗d_remap, float ∗d_number_density_matrix)

  a

7.16.1 Function Documentation

7.16.1.1 macro_micro()

```
void macro_micro (
            unsigned NUM_THREADS,
            unsigned N,
            unsigned converged,
            unsigned n_materials,
            unsigned n_isotopes,
            unsigned n_tallies,
            cross_section_data ∗ d_xsdata,
            particle_data ∗ d_particles,
            tally_data ∗ d_tally,
            unsigned ∗ d_remap,
            float ∗ d_number_density_matrix )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*converged*	- flag for tally scoring
in	*n_materials*	- number of materials
in	*n_isotopes*	- number of isotopes
in	*n_tallies*	- number of tallies
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_tally*	- device pointer to tally array
in	*d_remap*	- device pointer to data remapping vector
in	*d_number_density_matrix*	- device pointer to material number density array

7.16.1.2 macro_micro_kernel()

```
__global__ void macro_micro_kernel (
             unsigned N,
             unsigned converged,
             unsigned n_materials,
             unsigned n_tallies,
             cross_section_data * d_xsdata,
             particle_data * d_particles,
             tally_data * d_tally,
             unsigned * d_remap,
             float * d_number_density_matrix )
```

7.17 main.cpp File Reference

```
#include "warp.h"
```

Functions

- int main (int argc, char ∗argv[ ])

  a

7.17.1 Function Documentation

7.17.1.1 main()

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

a

b

Parameters

in	*argc*	- argument count
in	*argv*	- arguments

7.18 miss.cu File Reference

```
#include <optix_world.h>
#include "datadef.h"
```

Functions

- rtDeclareVariable (optix::Ray, ray, rtCurrentRay,)
- rtDeclareVariable (intersection_point, payload, rtPayload,)
- rtDeclareVariable (uint, launch_index_in, rtLaunchIndex,)
- rtDeclareVariable (unsigned, trace_type,,)
- rtDeclareVariable (unsigned, outer_cell,,)
- RT_PROGRAM void miss ()

Variables

- rtBuffer< unsigned, 1 > rxn_buffer
- rtBuffer< spatial_data, 1 > positions_buffer
- rtBuffer< unsigned, 1 > matnum_buffer
- rtBuffer< unsigned, 1 > cellnum_buffer
- rtBuffer< unsigned, 1 > remap_buffer

7.18.1 Function Documentation

7.18.1.1 miss()

```
RT_PROGRAM void miss ( )
```

7.18.1.2 rtDeclareVariable() [1/5]

```
rtDeclareVariable (
             optix::Ray ,
             ray ,
             rtCurrentRay   )
```

7.18.1.3 rtDeclareVariable() [2/5]

```
rtDeclareVariable (
             intersection_point ,
             payload ,
             rtPayload   )
```

7.18.1.4 rtDeclareVariable() [3/5]

```
rtDeclareVariable (
             uint ,
             launch_index_in ,
             rtLaunchIndex  )
```

7.18.1.5 rtDeclareVariable() [4/5]

```
rtDeclareVariable (
             unsigned ,
             trace_type  )
```

7.18.1.6 rtDeclareVariable() [5/5]

```
rtDeclareVariable (
             unsigned ,
             outer_cell  )
```

7.18.2 Variable Documentation

7.18.2.1 cellnum_buffer

```
rtBuffer<unsigned,1> cellnum_buffer
```

7.18.2.2 matnum_buffer

```
rtBuffer<unsigned,1> matnum_buffer
```

7.18.2.3 positions_buffer

```
rtBuffer<spatial_data,1> positions_buffer
```

7.18.2.4 remap_buffer

```
rtBuffer<unsigned,1> remap_buffer
```

7.18.2.5 rxn_buffer

```
rtBuffer<unsigned,1> rxn_buffer
```

7.19 mt19937ar.cpp File Reference

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

Macros

- #define N 624
- #define M 397
- #define MATRIX_A 0x9908b0dfUL /∗ constant vector a ∗/
- #define UPPER_MASK 0x80000000UL /∗ most significant w-r bits ∗/
- #define LOWER_MASK 0x7fffffffUL /∗ least significant r bits ∗/

Functions

- void init_genrand (unsigned long s)
- void init_by_array (unsigned long init_key[ ], int key_length)
- unsigned long genrand_int32 (void)
- long genrand_int31 (void)
- double genrand_real1 (void)
- double genrand_real2 (void)
- double genrand_real3 (void)
- double genrand_res53 (void)

Variables

- static unsigned long mt [N]
- static int mti =N+1

7.19.1 Macro Definition Documentation

7.19.1.1 LOWER_MASK

```
#define LOWER_MASK 0x7fffffffUL /* least significant r bits */
```

7.19.1.2 M

```
#define M 397
```

7.19.1.3 MATRIX_A

```
#define MATRIX_A 0x9908b0dfUL /* constant vector a */
```

7.19.1.4 N

```
#define N 624
```

7.19.1.5 UPPER_MASK

```
#define UPPER_MASK 0x80000000UL /* most significant w-r bits */
```

7.19.2 Function Documentation

7.19.2.1 genrand_int31()

```
long genrand_int31 (
            void  )
```

7.19.2.2 genrand_int32()

```
unsigned long genrand_int32 (
            void  )
```

7.19.2.3 genrand_real1()

```
double genrand_real1 (
             void  )
```

7.19.2.4 genrand_real2()

```
double genrand_real2 (
             void  )
```

7.19.2.5 genrand_real3()

```
double genrand_real3 (
             void  )
```

7.19.2.6 genrand_res53()

```
double genrand_res53 (
             void  )
```

7.19.2.7 init_by_array()

```
void init_by_array (
             unsigned long init_key[],
             int key_length )
```

7.19.2.8 init_genrand()

```
void init_genrand (
             unsigned long s )
```

7.19.3 Variable Documentation

7.19.3.1 mt

```
unsigned long mt[N]  [static]
```

7.19.3.2 mti

```
int mti =N+1  [static]
```

7.20 optix_stuff.cpp File Reference

```
#include <vector>
#include <iostream>
#include <sstream>
#include <cmath>
#include <assert.h>
#include <time.h>
#include <string.h>
#include <png++/png.hpp>
#include "datadef.h"
#include "wprimitive.h"
#include "wgeometry.h"
#include "optix_stuff.h"
#include "device_copies.h"
```

7.21 optix_stuff.h File Reference

```
#include <optix_world.h>
```

Classes

- class optix_stuff

    OptiX stuff class.

7.22 optixtest.cpp File Reference

```
#include "warp.h"
#include "optix_stuff.h"
```

Functions

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

7.22.1 Function Documentation

7.22.1.1 main()

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

7.23 pop_fission.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "warp_device.cuh"
#include "check_cuda.h"
```

Functions

- __global__ void pop_fission_kernel (unsigned N, cross_section_data *d_xsdata, particle_data *d_particles, unsigned *d_scanned, spatial_data *fission_particles, float *fission_energy)
- void pop_fission (unsigned NUM_THREADS, unsigned N, cross_section_data *d_xsdata, particle_data *d↩ _particles, unsigned *d_scanned, spatial_data *fission_particles, float *fission_energy)

  a

7.23.1 Function Documentation

7.23.1.1 pop_fission()

```
void pop_fission (
            unsigned NUM_THREADS,
            unsigned N,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_scanned,
            spatial_data * fission_particles,
            float * fission_energy )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_scanned*	- device pointer to array of the cumulative sum (scan) of the yield array, used to find final index where new particles will be written
in	*fission_particles*	- device pointer to intermadiate spatial data array where popped values will be written
in	*fission_energy*	- device pointer to intermadiate energy data array where popped values will be written

7.23.1.2 pop_fission_kernel()

```
__global__ void pop_fission_kernel (
            unsigned N,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_scanned,
            spatial_data * fission_particles,
            float * fission_energy )
```

7.24 reaction_edges3.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "check_cuda.h"
```

Functions

- __global__ void reaction_edges_kernel (unsigned N, unsigned *edges, unsigned *rxn)
- void reaction_edges (unsigned NUM_THREADS, unsigned N, unsigned *d_edges, unsigned *d_rxn)

  a

7.24.1 Function Documentation

7.24.1.1 reaction_edges()

```
void reaction_edges (
            unsigned NUM_THREADS,
            unsigned N,
            unsigned * d_edges,
            unsigned * d_rxn )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*d_edges*	- device pointer to the edges array (stores to indices of edges the reaction blocks)
in	*d_rxn*	- device pointer of the reaction number array

7.24.1.2 reaction_edges_kernel()

```
__global__ void reaction_edges_kernel (
            unsigned N,
            unsigned * edges,
            unsigned * rxn )
```

7.25 rebase_yield.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "warp_device.cuh"
```

Functions

- __global__ void rebase_yield_kernel (unsigned N, float keff, particle_data ∗d_particles)
- void rebase_yield (unsigned NUM_THREADS, unsigned N, float keff, particle_data ∗d_particles)

7.25.1 Function Documentation

7.25.1.1 rebase_yield()

```
void rebase_yield (
            unsigned NUM_THREADS,
            unsigned N,
            float keff,
            particle_data * d_particles )
```

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*keff*	- k-effective of the current cycle
in	*d_particles*	- device pointer to particle data pointer array

7.25.1.2 rebase_yield_kernel()

```
__global__ void rebase_yield_kernel (
            unsigned N,
            float keff,
            particle_data * d_particles )
```

7.26 safety_check.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "warp_device.cuh"
#include "check_cuda.h"
#include "wfloat3.h"
```

Functions

- __global__ void safety_check_kernel (unsigned N, cross_section_data *d_xsdata, particle_data *d_particles, unsigned *d_remap)
- void safety_check (unsigned NUM_THREADS, unsigned N, cross_section_data *d_xsdata, particle_data *d_particles, unsigned *d_remap)

  a

7.26.1 Function Documentation

7.26.1.1 safety_check()

```
void safety_check (
            unsigned NUM_THREADS,
            unsigned N,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_remap )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.26.1.2 safety_check_kernel()

```
__global__ void safety_check_kernel (
            unsigned N,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_remap )
```

7.27 sample_fissile_energy.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "warp_device.cuh"
#include "check_cuda.h"
```

Functions

- __global__ void sample_fissile_energy_kernel (unsigned N, float a, float b, unsigned ∗rn_bank, float ∗E)
- void sample_fissile_energy (unsigned NUM_THREADS, unsigned N, float a, float b, unsigned ∗rn_bank, float ∗E)

  a

7.27.1 Function Documentation

7.27.1.1 sample_fissile_energy()

```
void sample_fissile_energy (
            unsigned NUM_THREADS,
            unsigned N,
            float a,
            float b,
            unsigned * rn_bank,
            float * E )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*a*	- Watt spectrum, parameter a
in	*b*	- Watt spectrum, parameter b
in	*rn_bank*	- device pointer to random number array
in	*E*	- device pointer to energy data array

7.27.1.2 sample_fissile_energy_kernel()

```
__global__ void sample_fissile_energy_kernel (
             unsigned N,
             float a,
             float b,
             unsigned * rn_bank,
             float * E )
```

7.28 sample_fixed_source.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "LCRNG.cuh"
#include "check_cuda.h"
```

Functions

- __global__ void sample_fixed_source_kernel (unsigned N, unsigned ∗active, unsigned ∗rn_bank, float ∗E, source_point ∗space)
- void sample_fixed_source (unsigned NUM_THREADS, unsigned N, unsigned ∗active, unsigned ∗rn_bank, float ∗E, source_point ∗space)

  a

7.28.1 Function Documentation

7.28.1.1 sample_fixed_source()

```
void sample_fixed_source (
             unsigned NUM_THREADS,
             unsigned N,
             unsigned * active,
             unsigned * rn_bank,
             float * E,
             source_point * space )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*active*	- device pointer to remapping vector
in	*rn_bank*	- device pointer to random number array
in	*E*	- device pointer to energy data array
in	*space*	- device pointer to spatial data array

7.28.1.2 sample_fixed_source_kernel()

```
__global__ void sample_fixed_source_kernel (
             unsigned N,
             unsigned * active,
             unsigned * rn_bank,
             float * E,
             source_point * space )
```

7.29 scatter_conti.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "wfloat3.h"
#include "warp_device.cuh"
#include "check_cuda.h"
```

Functions

- __global__ void scatter_conti_kernel (unsigned N, unsigned starting_index, cross_section_data *d_xsdata, particle_data *d_particles, unsigned *d_remap)
- void scatter_conti (cudaStream_t stream, unsigned NUM_THREADS, unsigned N, unsigned starting_index, cross_section_data *d_xsdata, particle_data *d_particles, unsigned *d_remap)

    a

7.29.1 Function Documentation

7.29.1.1 scatter_conti()

```
void scatter_conti (
            cudaStream_t stream,
            unsigned NUM_THREADS,
            unsigned N,
            unsigned starting_index,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_remap )
```

a

b

Parameters

in	*stream*	- CUDA stream to launch the kernel on
in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid for continuum scattering
in	*starting_index*	- starting index of the continuum scatter block in the remap vector
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.29.1.2 scatter_conti_kernel()

```
__global__ void scatter_conti_kernel (
            unsigned N,
            unsigned starting_index,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_remap )
```

7.30 scatter_level.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "wfloat3.h"
#include "warp_device.cuh"
#include "check_cuda.h"
```

Functions

- __global__ void scatter_level_kernel (unsigned N, unsigned starting_index, cross_section_data ∗d_xsdata, particle_data ∗d_particles, unsigned ∗d_remap)
- void scatter_level (cudaStream_t stream, unsigned NUM_THREADS, unsigned N, unsigned starting_index, cross_section_data ∗d_xsdata, particle_data ∗d_particles, unsigned ∗d_remap)

  a

7.30.1 Function Documentation

7.30.1.1 scatter_level()

```
void scatter_level (
             cudaStream_t stream,
             unsigned NUM_THREADS,
             unsigned N,
             unsigned starting_index,
             cross_section_data * d_xsdata,
             particle_data * d_particles,
             unsigned * d_remap )
```

a

b

Parameters

in	*stream*	- CUDA stream to launch the kernel on
in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid for level scattering
in	*starting_index*	- starting index of the level scatter block in the remap vector
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.30.1.2 scatter_level_kernel()

```
__global__ void scatter_level_kernel (
             unsigned N,
             unsigned starting_index,
             cross_section_data * d_xsdata,
             particle_data * d_particles,
             unsigned * d_remap )
```

7.31 scatter_multi.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "wfloat3.h"
#include "warp_device.cuh"
#include "check_cuda.h"
```

Functions

- __global__ void scatter_multi_kernel (unsigned N, unsigned starting_index, cross_section_data ∗d_xsdata, particle_data ∗d_particles, unsigned ∗d_remap)
- void scatter_multi (cudaStream_t stream, unsigned NUM_THREADS, unsigned N, unsigned starting_index, cross_section_data ∗d_xsdata, particle_data ∗d_particles, unsigned ∗d_remap)

  a

7.31.1 Function Documentation

7.31.1.1 scatter_multi()

```
void scatter_multi (
             cudaStream_t stream,
             unsigned NUM_THREADS,
             unsigned N,
             unsigned starting_index,
             cross_section_data ∗ d_xsdata,
             particle_data ∗ d_particles,
             unsigned ∗ d_remap )
```

a

b

Parameters

in	*stream*	- CUDA stream to launch the kernel on
in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid for multiplicity scattering
in	*starting_index*	- starting index of the multiplicity scatter block in the remap vector
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.31.1.2 scatter_multi_kernel()

```
__global__ void scatter_multi_kernel (
             unsigned N,
             unsigned starting_index,
             cross_section_data ∗ d_xsdata,
             particle_data ∗ d_particles,
             unsigned ∗ d_remap )
```

7.32 set_positions_rand.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
#include "warp_device.cuh"
```

Functions

- __global__ void set_positions_rand_kernel (unsigned N, unsigned outer_cell_type, spatial_data ∗positions↩ _ptr, unsigned ∗rn_bank, float x_min, float y_min, float z_min, float x_max, float y_max, float z_max)
- void set_positions_rand (unsigned NUM_THREADS, unsigned N, unsigned outer_cell_type, spatial_data ∗d_space, unsigned ∗d_rn_bank, float ∗outer_cell_dims)

  sets starting cycle points uniformly random

7.32.1 Function Documentation

7.32.1.1 set_positions_rand()

```
void set_positions_rand (
            unsigned NUM_THREADS,
            unsigned N,
            unsigned outer_cell_type,
            spatial_data * d_space,
            unsigned * d_rn_bank,
            float * outer_cell_dims )
```

sets starting cycle points uniformly random

sets starting cycle points uniformly random in the specified volume

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*outer_cell_type*	- the outer cell type, sets the shape of the sampling
in	*d_space*	- device pointer to spatial data array
in	*d_rn_bank*	- device pointer to random number array
in	*outer_cell_dims*	- host pointer to array of outer cell extrema

7.32.1.2 set_positions_rand_kernel()

```
__global__ void set_positions_rand_kernel (
            unsigned N,
            unsigned outer_cell_type,
            spatial_data * positions_ptr,
            unsigned * rn_bank,
            float x_min,
            float y_min,
            float z_min,
            float x_max,
            float y_max,
            float z_max )
```

7.33 sphere_mesh.cu File Reference

```
#include <optix.h>
#include <optixu/optixu_math_namespace.h>
#include <optixu/optixu_matrix_namespace.h>
#include <optixu/optixu_aabb_namespace.h>
#include "datadef.h"
```

Functions

- rtDeclareVariable (optix::Ray, ray, rtCurrentRay,)
- rtDeclareVariable (unsigned, cellnum, attribute cell_num,)
- rtDeclareVariable (int, celltal, attribute cell_tal,)
- rtDeclareVariable (unsigned, cellmat, attribute cell_mat,)
- rtDeclareVariable (unsigned, cellfissile, attribute cell_fis,)
- rtDeclareVariable (unsigned, sense, attribute cell_sense,)
- rtDeclareVariable (float3, normal, attribute normal,)
- RT_PROGRAM void intersect (int object_dex)
- RT_PROGRAM void bounds (int object_dex, float result[6])

Variables

- rtBuffer< geom_data, 1 > dims

7.33.1 Function Documentation

7.33.1.1 bounds()

```
RT_PROGRAM void bounds (
            int object_dex,
            float result[6] )
```

7.33.1.2 intersect()

```
RT_PROGRAM void intersect (
             int object_dex )
```

7.33.1.3 rtDeclareVariable() [1/7]

```
rtDeclareVariable (
             optix::Ray ,
             ray ,
             rtCurrentRay  )
```

7.33.1.4 rtDeclareVariable() [2/7]

```
rtDeclareVariable (
             unsigned ,
             cellnum ,
             attribute cell_num )
```

7.33.1.5 rtDeclareVariable() [3/7]

```
rtDeclareVariable (
             int ,
             celltal ,
             attribute cell_tal )
```

7.33.1.6 rtDeclareVariable() [4/7]

```
rtDeclareVariable (
             unsigned ,
             cellmat ,
             attribute cell_mat )
```

7.33.1.7 rtDeclareVariable() [5/7]

```
rtDeclareVariable (
             unsigned ,
             cellfissile ,
             attribute cell_fis )
```

7.33.1.8 rtDeclareVariable() [6/7]

```
rtDeclareVariable (
            unsigned ,
            sense ,
            attribute cell_sense )
```

7.33.1.9 rtDeclareVariable() [7/7]

```
rtDeclareVariable (
            float3 ,
            normal ,
            attribute normal )
```

7.33.2 Variable Documentation

7.33.2.1 dims

```
rtBuffer<geom_data,1> dims
```

7.34 test_function.cu File Reference

```
#include <cuda.h>
#include <stdio.h>
#include "datadef.h"
```

Functions

- __global__ void test_kernel (unsigned N, cross_section_data ∗d_xsdata, particle_data ∗d_particles, tally_data ∗d_tally, unsigned ∗d_remap)
- void test_function (unsigned NUM_THREADS, unsigned N, cross_section_data ∗d_xsdata, particle_data ∗d_particles, tally_data ∗d_tally, unsigned ∗d_remap)

7.34.1 Function Documentation

7.34.1.1 test_function()

```
void test_function (
             unsigned NUM_THREADS,
             unsigned N,
             cross_section_data * d_xsdata,
             particle_data * d_particles,
             tally_data * d_tally,
             unsigned * d_remap )
```

7.34.1.2 test_kernel()

```
__global__ void test_kernel (
             unsigned N,
             cross_section_data * d_xsdata,
             particle_data * d_particles,
             tally_data * d_tally,
             unsigned * d_remap )
```

7.35 unionize.py File Reference

Classes

- class unionize.cross_section_data

  handles cross section data

Namespaces

- unionize

Variables

- bool unionize.ace_available = True

7.36 warp.h File Reference

```
#include <vector>
#include <iostream>
#include <sstream>
#include <cmath>
#include <assert.h>
#include <time.h>
#include <string.h>
#include <cuda.h>
#include <curand.h>
#include <cudpp_hash.h>
#include <Python.h>
#include <png++/png.hpp>
#include "datadef.h"
#include "wprimitive.h"
#include "wgeometry.h"
#include "warp_cuda.h"
#include "whistory.h"
```

7.37 warp_cuda.h File Reference

```
#include "device_copies.h"
```

Functions

- void write_to_file (unsigned ∗, unsigned, std::string)

  writes CUDA array to a text file
- void set_positions_rand (unsigned, unsigned, unsigned, spatial_data ∗, unsigned ∗, float ∗)

  sets starting cycle points uniformly random
- void copy_points (unsigned, unsigned, unsigned ∗, unsigned, unsigned ∗, spatial_data ∗, spatial_data ∗, float ∗, float ∗)

  copy points between two sets of space and energy data buffers, redirected with a mapping array
- void sample_fixed_source (unsigned, unsigned, unsigned ∗, unsigned ∗, float ∗, spatial_data ∗)
- void macro_micro (unsigned, unsigned, unsigned, unsigned, unsigned, unsigned, cross_section_data ∗, particle_data ∗, tally_data ∗, unsigned ∗, float ∗)

  a
- void scatter_level (cudaStream_t, unsigned, unsigned, unsigned, cross_section_data ∗, particle_data ∗, unsigned ∗)

  a
- void scatter_conti (cudaStream_t, unsigned, unsigned, unsigned, cross_section_data ∗, particle_data ∗, unsigned ∗)

  a
- void scatter_multi (cudaStream_t, unsigned, unsigned, unsigned, cross_section_data ∗, particle_data ∗, unsigned ∗)

  a
- void fission (cudaStream_t, unsigned, unsigned, unsigned, cross_section_data ∗, particle_data ∗, unsigned ∗)

  a
- void find_E_grid_index (unsigned, unsigned, cross_section_data ∗, unsigned ∗, float ∗, unsigned ∗, unsigned ∗)

  a
- void pop_fission (unsigned, unsigned, cross_section_data ∗, particle_data ∗, unsigned ∗, spatial_data ∗, float ∗)

  a
- void rebase_yield (unsigned, unsigned, float, particle_data ∗)
- void reaction_edges (unsigned, unsigned, unsigned ∗, unsigned ∗)

  a
- void sample_fissile_energy (unsigned, unsigned, float, float, unsigned ∗, float ∗)

  a
- void safety_check (unsigned, unsigned, cross_section_data ∗, particle_data ∗, unsigned ∗)

  a
- void check_pointers (unsigned, unsigned, unsigned, cross_section_data ∗)

  a

7.37.1 Function Documentation

7.37.1.1 check_pointers()

```
void check_pointers (
             unsigned NUM_THREADS,
             unsigned dex0,
             unsigned dex1,
             cross_section_data * d_xsdata )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*dex0*	- starting index
in	*dex1*	- ending index
in	*d_xsdata*	- device pointer to cross section data pointer array

7.37.1.2 copy_points()

```
void copy_points (
             unsigned NUM_THREADS,
             unsigned Nout,
             unsigned * Nvalid,
             unsigned current_index,
             unsigned * to_valid,
             spatial_data * positions_out,
             spatial_data * positions_in,
             float * E_out,
             float * E_in )
```

copy points between two sets of space and energy data buffers, redirected with a mapping array

copy points between two sets of space and energy data buffers, redirected with a mapping array

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*Nout*	- the total number of threads to launch on the grid
in	*Nvalid*	- the total number of device elements to copy from
in	*current_index*	- starting index
in	*to_valid*	- device pointer to data remapping vector
in	*positions_out*	- device pointer to spatial data array destination
in	*positions_in*	- device pointer to spatial data array source
in	*E_out*	- device pointer to energy data array destination
in	*E_in*	- device pointer to energy data array source

7.37.1.3 find_E_grid_index()

```
void find_E_grid_index (
            unsigned NUM_THREADS,
            unsigned N,
            cross_section_data * d_xsdata,
            unsigned * d_remap,
            float * d_E,
            unsigned * d_index,
            unsigned * d_rxn )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_remap*	- device pointer to data remapping vector
in	*d_E*	- device pointer to energy data array
in	*d_index*	- device pointer to index array (stores the unionized grid index of the current energy)
in	*d_rxn*	- device pointer of the reaction number array

7.37.1.4 fission()

```
void fission (
            cudaStream_t stream,
            unsigned NUM_THREADS,
            unsigned N,
            unsigned starting_index,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_remap )
```

a

b

Parameters

in	*stream*	- CUDA stream to launch the kernel on
in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid for fission
in	*starting_index*	- starting index of the fission block in the remap vector
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.37.1.5 macro_micro()

```
void macro_micro (
             unsigned NUM_THREADS,
             unsigned N,
             unsigned converged,
             unsigned n_materials,
             unsigned n_isotopes,
             unsigned n_tallies,
             cross_section_data * d_xsdata,
             particle_data * d_particles,
             tally_data * d_tally,
             unsigned * d_remap,
             float * d_number_density_matrix )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*converged*	- flag for tally scoring
in	*n_materials*	- number of materials
in	*n_isotopes*	- number of isotopes
in	*n_tallies*	- number of tallies
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_tally*	- device pointer to tally array
in	*d_remap*	- device pointer to data remapping vector
in	*d_number_density_matrix*	- device pointer to material number density array

7.37.1.6 pop_fission()

```
void pop_fission (
             unsigned NUM_THREADS,
             unsigned N,
             cross_section_data * d_xsdata,
             particle_data * d_particles,
             unsigned * d_scanned,
             spatial_data * fission_particles,
             float * fission_energy )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_scanned*	- device pointer to array of the cumulative sum (scan) of the yield array, used to find final index where new particles will be written
in	*fission_particles*	- device pointer to intermadiate spatial data array where popped values will be written
in	*fission_energy*	- device pointer to intermadiate energy data array where popped values will be written

7.37.1.7 reaction_edges()

```
void reaction_edges (
            unsigned NUM_THREADS,
            unsigned N,
            unsigned * d_edges,
            unsigned * d_rxn )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*d_edges*	- device pointer to the edges array (stores to indices of edges the reaction blocks)
in	*d_rxn*	- device pointer of the reaction number array

7.37.1.8 rebase_yield()

```
void rebase_yield (
            unsigned NUM_THREADS,
            unsigned N,
            float keff,
            particle_data * d_particles )
```

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*keff*	- k-effective of the current cycle
in	*d_particles*	- device pointer to particle data pointer array

7.37.1.9 safety_check()

```
void safety_check (
            unsigned NUM_THREADS,
            unsigned N,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_remap )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.37.1.10 sample_fissile_energy()

```
void sample_fissile_energy (
            unsigned NUM_THREADS,
            unsigned N,
            float a,
            float b,
            unsigned * rn_bank,
            float * E )
```

a

b

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*a*	- Watt spectrum, parameter a
in	*b*	- Watt spectrum, parameter b
in	*rn_bank*	- device pointer to random number array
in	*E*	- device pointer to energy data array

7.37.1.11 sample_fixed_source()

```
void sample_fixed_source (
            unsigned ,
            unsigned ,
            unsigned * ,
            unsigned * ,
            float * ,
            spatial_data *  )
```

7.37.1.12 scatter_conti()

```
void scatter_conti (
            cudaStream_t stream,
            unsigned NUM_THREADS,
            unsigned N,
            unsigned starting_index,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_remap )
```

a

b

Parameters

in	*stream*	- CUDA stream to launch the kernel on
in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid for continuum scattering
in	*starting_index*	- starting index of the continuum scatter block in the remap vector
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.37.1.13 scatter_level()

```
void scatter_level (
            cudaStream_t stream,
            unsigned NUM_THREADS,
            unsigned N,
            unsigned starting_index,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_remap )
```

a

b

Parameters

in	*stream*	- CUDA stream to launch the kernel on
in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid for level scattering
in	*starting_index*	- starting index of the level scatter block in the remap vector
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.37.1.14 scatter_multi()

```
void scatter_multi (
            cudaStream_t stream,
            unsigned NUM_THREADS,
            unsigned N,
            unsigned starting_index,
            cross_section_data * d_xsdata,
            particle_data * d_particles,
            unsigned * d_remap )
```

a

b

Parameters

in	*stream*	- CUDA stream to launch the kernel on
in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid for multiplicity scattering
in	*starting_index*	- starting index of the multiplicity scatter block in the remap vector
in	*d_xsdata*	- device pointer to cross section data pointer array
in	*d_particles*	- device pointer to particle data pointer array
in	*d_remap*	- device pointer to data remapping vector

7.37.1.15 set_positions_rand()

```
void set_positions_rand (
            unsigned NUM_THREADS,
            unsigned N,
            unsigned outer_cell_type,
            spatial_data * d_space,
            unsigned * d_rn_bank,
            float * outer_cell_dims )
```

sets starting cycle points uniformly random

sets starting cycle points uniformly random in the specified volume

Parameters

in	*NUM_THREADS*	- the number of threads to run per thread block
in	*N*	- the total number of threads to launch on the grid
in	*outer_cell_type*	- the outer cell type, sets the shape of the sampling
in	*d_space*	- device pointer to spatial data array
in	*d_rn_bank*	- device pointer to random number array
in	*outer_cell_dims*	- host pointer to array of outer cell extrema

7.37.1.16 write_to_file()

```
void write_to_file (
             unsigned * array_in,
             unsigned N,
             std::string filename )
```

writes CUDA array to a text file

copies the cuda array to a local buffer, writes to buffer to a new file, then frees the local memory

Parameters

in	*array←* *_in*	- device pointer to array to write
in	*N*	- number of elements to write
in	*filename*	- name for the file

7.38 warp_device.cuh File Reference

7.39 wfloat3.h File Reference

Classes

- class wfloat3

    class definitions for device vector operations

7.40 wgeometry.cpp File Reference

```
#include <vector>
#include <iostream>
#include <sstream>
#include <stdio.h>
#include <cmath>
```

```
#include <assert.h>
#include <time.h>
#include <string.h>
#include "datadef.h"
#include "wprimitive.h"
#include "wgeometry.h"
```

7.41 wgeometry.h File Reference

```
#include "wprimitive.h"
```

Classes

- class wgeometry

  wgeometry class

7.42 whistory.cpp File Reference

```
#include <vector>
#include <iostream>
#include <sstream>
#include <stdio.h>
#include <cmath>
#include <assert.h>
#include <time.h>
#include <string.h>
#include <cuda.h>
#include <curand.h>
#include <cudpp.h>
#include <Python.h>
#include <png++/png.hpp>
#include "datadef.h"
#include "wprimitive.h"
#include "wgeometry.h"
#include "optix_stuff.h"
#include "warp_cuda.h"
#include "whistory.h"
#include "check_cuda.h"
```

Variables

- optix_stuff optix_obj

7.42.1 Variable Documentation

7.42.1.1 optix_obj

```
optix_stuff optix_obj
```

7.43 whistory.h File Reference

Classes

- class whistory

    whistory class

7.44 wprimitive.cpp File Reference

```
#include <vector>
#include <iostream>
#include <sstream>
#include <cmath>
#include <assert.h>
#include <time.h>
#include "datadef.h"
#include "wprimitive.h"
```

7.45 wprimitive.h File Reference

```
#include "datadef.h"
```

Classes

- class primitive

    primitive class

7.46 write_to_file.cu File Reference

```
#include <iostream>
#include <sstream>
#include <stdio.h>
#include <cuda.h>
```

Functions

- void write_to_file (unsigned ∗array_in, unsigned N, std::string filename)

    writes CUDA array to a text file

7.46.1 Function Documentation

7.46.1.1 write_to_file()

```
void write_to_file (
             unsigned * array_in,
             unsigned N,
             std::string filename )
```

writes CUDA array to a text file

copies the cuda array to a local buffer, writes to buffer to a new file, then frees the local memory

Parameters

in	*array←* *_in*	- device pointer to array to write
in	*N*	- number of elements to write
in	*filename*	- name for the file

Index