Sapphire

v1.0

Generated by Doxygen 1.8.5

Thu Apr 9 2020 14:00:09

Contents

Chapter 1

README

This is the original version of Sapphire by Mary Beard. Only the README-file has been changed.

General Information

Compilation & Execution

Requirements

Compiling Sapphire requires CMake, ROOT, and GNU Scientific Library (GSL). Check the files in cmake/Modules whether cmake is looking for these packages in the right place.

For an automized generation of documentation files, one needs doxygen installed on the system.

For the Message Passing Interface (MPI) build, boost 1.4 libraries are required. The MPI build is optional and can be enabled in the CMakeLists.txt.

Build

1. Create a build directory under the main Sapphire directory and change to that directory. “‘bash mkdir build cd build “‘
2. Run CMake against the main directory, optionally specifying the desired C++ compiler. “‘bash cmake -DCMAKE_CXX_COMPILER=icpc -DCMAKE_C_COMPILER=icc “‘
3. Type make install to build Sapphire. The executable is put in the build directory. “‘bash make install “‘

Be aware that Sapphire links the paths to the needed tables at compile time. While the executable can be moved, the main Sapphire directory should stay in place. If moved, the build process should be repeated.

Run

To execute the code just enter

“‘ ./sapphire X+a “‘

where X is the heavy nucleus and a is the projectile. Examples are 25Mg+a or 60Fe+n.

Features

Calculate Cross Sections

References

About Sapphire

If you are using *Sapphire* you should cite one or more of the references below.

- M. Beard, E. Uberseder, R. Crowter, and M. Wiescher, *Comparison of statistical model calculations for stable isotope neutron capture*, Physical Review C **90**, 034619 (2014), DOI: 10.1103/PhysRevC.90.034619
- Mary Beard, Ethan Uberseder, and M. Wiescher, *Statistical Model Calculations for (n,g) Reactions*, EPJ Web of Conferences **93**, 04001 (2015), DOI: 10.1103/PhysRevC.90.034619

Using Sapphire

If you are using *Sapphire* for your research, please send us a reference of your work and we will list it here.

- X. Fang, W. P. Tan, M. Beard, R. J. deBoer, G. Gilardy, H. Jung, Q. Liu, S. Lyons, D. Robertson, K. Setoodehnia, C. Seymour, E. Stech, B. Vande Kolk, M. Wiescher, R. T. deSouza, S. Hudan, V. Singh, X. D. Tang, and E. Uberseder, *Experimental measurement of 12C+16O fusion at stellar energies*, Physical Review C **96**, 045804 (2017), DOI: 10.1103/PhysRevC.96.045804
- M. Beard, E. Uberseder, R. Crowter, and M. Wiescher, *Comparison of statistical model calculations for stable isotope neutron capture*, Physical Review C **90**, 034619 (2014), DOI: 10.1103/PhysRevC.90.034619
- Mary Beard, Ethan Uberseder, and M. Wiescher, *Statistical Model Calculations for (n,g) Reactions*, EPJ Web of Conferences **93**, 04001 (2015), DOI: 10.1103/PhysRevC.90.034619

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 boost Namespace Reference

Namespaces

- serialization

6.2 boost::serialization Namespace Reference

Functions

- void serialize (Archive &ar, DecayData &g, const unsigned int version)
- void serialize (Archive &ar, DecayProduct &g, const unsigned int version)

6.2.1 Function Documentation

6.2.1.1 void boost::serialization::serialize (Archive & *ar,* DecayData & *g,* const unsigned int *version*)

6.2.1.2 void boost::serialization::serialize (Archive & *ar,* DecayProduct & *g,* const unsigned int *version*)

References DecayProduct::A_, DecayProduct::excitationEnergy_, DecayProduct::fragmentEnergy_, DecayProduct::fragmentEnergyCM_, DecayProduct::fragmentMomentumX_, DecayProduct::fragmentMomentumY_, DecayProduct::fragmentMomentumZ_, DecayProduct::J_, DecayProduct::particleEnergy_, DecayProduct::particleEnergyCM_, DecayProduct::particleMomentumX_, DecayProduct::particleMomentumY_, DecayProduct::particleMomentumZ_, DecayProduct::particlePhiCM_, DecayProduct::particleThetaCM_, DecayProduct::particleType_, DecayProduct::Pi_, and DecayProduct::Z_.

6.3 std Namespace Reference

Namespaces

- tr1

Classes

- struct equal_to< MassKey >

6.4 std::tr1 Namespace Reference

Classes

- struct hash< MassKey >

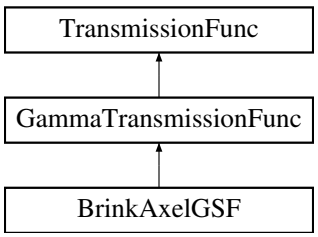
Chapter 7

Class Documentation

7.1 BrinkAxelGSF Class Reference

`#include <BrinkAxelGSF.h>`

Inheritance diagram for BrinkAxelGSF:



Public Member Functions

- BrinkAxelGSF (int, int, double, int, double, int, double, double, double, double, TransmissionFunc ∗)
- double CalcStrengthFunction (double)

Additional Inherited Members

7.1.1 Detailed Description

Class for the Brink-Axel-GSF

7.1.2 Constructor & Destructor Documentation

7.1.2.1 BrinkAxelGSF::BrinkAxelGSF (int *z2,* int *m2,* double *jInitial,* int *piInitial,* double *jFinal,* int *piFinal,* double *maxL,* double *totalWidthForCorrection,* double *uncorrTotalWidthForCorrection,* double *uncorrTotalWidthSqrdForCorrection,* TransmissionFunc ∗ *previous*)

BrinkAxelGSF.h

7.1.3 Member Function Documentation

7.1.3.1 double BrinkAxelGSF::CalcStrengthFunction (double *energy*) [virtual]

Implements GammaTransmissionFunc.

References GDRParameters::E_, GammaTransmissionFunc::gdrParameters_, GDRParameters::kSigmaGamma_, and TransmissionFunc::maxL_.

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/BrinkAxelGSF.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/BrinkAxelGSF.cpp

7.2 CDFEntry Class Reference

`#include <Decayer.h>`

Public Member Functions

- CDFEntry (int pairIndex, double energy, double value)

Public Attributes

- int pairIndex_
- double energy_
- double value_

7.2.1 Constructor & Destructor Documentation

7.2.1.1 CDFEntry::CDFEntry (int *pairIndex,* double *energy,* double *value*) [inline]

7.2.2 Member Data Documentation

7.2.2.1 double CDFEntry::energy_

7.2.2.2 int CDFEntry::pairIndex_

7.2.2.3 double CDFEntry::value_

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/Decayer.h

7.3 CoulFunc Class Reference

`#include <CoulFunc.h>`

Public Member Functions

- CoulFunc (int z1, int z2, double redmass, bool useGSLFunctions)
- int z1 () const
- int z2 () const

- double redmass () const
- int lLast () const
- double radiusLast () const
- double energyLast () const
- struct CoulWaves coulLast () const
- void setLast (int, double, double, CoulWaves)
- CoulWaves operator() (int, double, double)
- double Penetrability (int, double, double)
- double PEShift (int, double, double)
- double PEShift_dE (int, double, double)

Static Public Member Functions

- static void GSLErrorHandler (const char *, const char *, int, int)

7.3.1 Constructor & Destructor Documentation

7.3.1.1 CoulFunc::CoulFunc (int *z1,* int *z2,* double *redmass,* bool *useGSLFunctions*)

References CoulWaves::dF, CoulWaves::dG, CoulWaves::F, and CoulWaves::G.

7.3.2 Member Function Documentation

7.3.2.1 struct CoulWaves CoulFunc::coulLast ( ) const

Returns the last Coulomb functions which were calculated.

Referenced by operator()().

7.3.2.2 double CoulFunc::energyLast ( ) const

Returns the last energy value at which the Coulomb functions were calculated.

Referenced by operator()().

7.3.2.3 void CoulFunc::GSLErrorHandler (const char * *reason,* const char * *file,* int *line,* int *errorCode*) [static]

Referenced by Initialize().

7.3.2.4 int CoulFunc::lLast ( ) const

Returns the last orbital angular momentum value at which the Coulomb functions were calculated.

Referenced by operator()(), and setLast().

7.3.2.5 CoulWaves CoulFunc::operator() (int *l,* double *radius,* double *energy*)

The parenthesis operator is defined to make the class instance callable as a function. The orbital angular momentum, radius, and energy in the center of mass system are the dependent variables. The function returns the Coulomb waves.

References coulLast(), CoulWaves::dF, CoulWaves::dG, energyLast(), CoulWaves::F, Coulomb_wave_functions::F_dF(), fstruc, CoulWaves::G, Coulomb_wave_functions::G_dG(), hbarc, lLast(), radiusLast(), redmass(), uconv, z1(), and z2().

Referenced by Penetrability(), and PEShift().

7.3.2.6 double CoulFunc::Penetrability (int *l,* double *radius,* double *energy*)

Returns the penetrability as a function of orbital angular momentum, radius, and energy in the center of mass system.

References CoulWaves::F, CoulWaves::G, hbarc, operator()(), redmass(), and uconv.

Referenced by EquivSquareWell::CalcTransmission().

7.3.2.7 double CoulFunc::PEShift (int *l,* double *radius,* double *energy*)

Returns the positive energy shift function a function of orbital angular momentum, radius, and energy in the center of mass system.

References CoulWaves::dF, CoulWaves::dG, CoulWaves::F, CoulWaves::G, hbarc, operator()(), redmass(), and uconv.

7.3.2.8 double CoulFunc::PEShift_dE (int *l,* double *radius,* double *energy*)

Returns the energy derivative of the shift function a function of orbital angular momentum, radius, and energy in the center of mass system.

References CoulWaves::F.

7.3.2.9 double CoulFunc::radiusLast ( ) const

Returns the last radius value at which the Coulomb functions were calculated.

Referenced by operator()().

7.3.2.10 double CoulFunc::redmass ( ) const

Returns the reduced mass of the particle pair.

Referenced by operator()(), Penetrability(), and PEShift().

7.3.2.11 void CoulFunc::setLast (int *lLast,* double *rLast,* double *eLast,* CoulWaves *coulLast*)

Sets the last calculated Coulomb waves and the values for which they were calculated.

References CoulWaves::dF, CoulWaves::dG, CoulWaves::F, CoulWaves::G, and lLast().

7.3.2.12 int CoulFunc::z1 ( ) const

Returns the atomic number of the first particle in the pair.

Referenced by operator()().

7.3.2.13 int CoulFunc::z2 ( ) const

Returns the atomic number of the second particle in the pair.

Referenced by operator()().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/CoulFunc.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/CoulFunc.cpp

7.4 Coulomb_wave_functions Class Reference

`#include <cwfcomp.H>`

Public Member Functions

- Coulomb_wave_functions (const bool is_it_normalized_c, const std::complex< double > &l_c, const std::complex< double > &eta_c)
- ~Coulomb_wave_functions (void)
- void F_dF_init (const std::complex< double > &z, const std::complex< double > &F, const std::complex< double > &dF)
- void F_dF (const std::complex< double > &z, std::complex< double > &F, std::complex< double > &dF)
- void G_dG (const std::complex< double > &z, std::complex< double > &G, std::complex< double > &dG)
- void H_dH (const int omega, const std::complex< double > &z, std::complex< double > &H, std::complex< double > &dH)
- void H_dH_scaled (const int omega, const std::complex< double > &z, std::complex< double > &H, std::complex< double > &dH)

Public Attributes

- const std::complex< double > l
- const std::complex< double > eta
- const bool is_it_normalized

7.4.1 Constructor & Destructor Documentation

7.4.1.1 Coulomb_wave_functions::Coulomb_wave_functions (const bool *is_it_normalized_c*, const std::complex< double > & *l_c*, const std::complex< double > & *eta_c*) [inline]

References eta, and is_it_normalized.

7.4.1.2 Coulomb_wave_functions::~Coulomb_wave_functions (void) [inline]

7.4.2 Member Function Documentation

7.4.2.1 void Coulomb_wave_functions::F_dF (const std::complex< double > & *z*, std::complex< double > & *F*, std::complex< double > & *dF*)

References is_it_normalized, isfinite(), SIGN, and sqrt_precision.

Referenced by G_dG(), H_dH(), H_dH_scaled(), and CoulFunc::operator()().

7.4.2.2 void Coulomb_wave_functions::F_dF_init (const std::complex< double > & *z*, const std::complex< double > & *F*, const std::complex< double > & *dF*)

7.4.2.3 void Coulomb_wave_functions::G_dG (const std::complex< double > & *z*, std::complex< double > & *G*, std::complex< double > & *dG*)

References F_dF(), H_dH(), is_it_normalized, isfinite(), and sqrt_precision.

Referenced by CoulFunc::operator()().

7.4.2.4 void Coulomb_wave_functions::H_dH (const int *omega,* const std::complex< double > & *z,* std::complex< double > & *H,* std::complex< double > & *dH*)

References F_dF(), is_it_normalized, isfinite(), and sqrt_precision.

Referenced by G_dG().

7.4.2.5 void Coulomb_wave_functions::H_dH_scaled (const int *omega,* const std::complex< double > & *z,* std::complex< double > & *H,* std::complex< double > & *dH*)

References F_dF(), is_it_normalized, isfinite(), and sqrt_precision.

7.4.3 Member Data Documentation

7.4.3.1 const std::complex<double> Coulomb_wave_functions::eta

Referenced by Coulomb_wave_functions().

7.4.3.2 const bool Coulomb_wave_functions::is_it_normalized

Referenced by Coulomb_wave_functions(), F_dF(), G_dG(), H_dH(), and H_dH_scaled().

7.4.3.3 const std::complex<double> Coulomb_wave_functions::l

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/include/cwfcomp.H
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/src/cwfcomp.cpp

7.5 CoulWaves Struct Reference

`#include <CoulFunc.h>`

Public Attributes

- double F
- double dF
- double G
- double dG

7.5.1 Member Data Documentation

7.5.1.1 double CoulWaves::dF

Referenced by Potential::CalcTransmission(), CoulFunc::CoulFunc(), CoulFunc::operator()(), CoulFunc::PEShift(), and CoulFunc::setLast().

7.5.1.2 double CoulWaves::dG

Referenced by Potential::CalcTransmission(), CoulFunc::CoulFunc(), CoulFunc::operator()(), CoulFunc::PEShift(), and CoulFunc::setLast().

7.5.1.3 double CoulWaves::F

Referenced by Potential::CalcTransmission(), CoulFunc::CoulFunc(), CoulFunc::operator()(), CoulFunc::Penetrability(), CoulFunc::PEShift(), CoulFunc::PEShift_dE(), and CoulFunc::setLast().

7.5.1.4 double CoulWaves::G

Referenced by Potential::CalcTransmission(), CoulFunc::CoulFunc(), CoulFunc::operator()(), CoulFunc::Penetrability(), CoulFunc::PEShift(), and CoulFunc::setLast().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/CoulFunc.h

7.6 CrossSection Class Reference

`#include <CrossSection.h>`

Public Member Functions

- CrossSection (int, int, int, std::string, bool, int entranceState=0, std::vector< int > exitStates=std::vector< int >(4,-1))
- bool IsValid () const
- void Calculate ()
- void PrintCrossSections ()
- void PrintTransmissionTerms ()
- std::pair< double, double > CalcAverageSWaveResWidth ()
- std::pair< double, double > CalcAveragePWaveResWidth ()
- std::pair< double, double > CalcAverageDWaveResWidth ()
- void CalculateReactionRates (bool)
- void PrintReactionRates (bool)

Static Public Member Functions

- static void SetResidualGamma (bool residual)
- static void SetResidualNeutron (bool residual)
- static void SetResidualProton (bool residual)
- static void SetResidualAlpha (bool residual)
- static void SetCalculateGammaCutoff (bool calc)
- static void CreateTempVector ()
- static void CreateMACSEnergiesVector ()

7.6.1 Constructor & Destructor Documentation

7.6.1.1 CrossSection::CrossSection (int *Z*, int *A*, int *pType*, std::string *energyFile*, bool *forRates*, int *entranceState* = `0`, std::vector< int > *exitStates* = `std::vector<int>(4,-1)`)

References Level::energy_, NuclearLevels::FindLevels(), hbarc, Level::J_, pi, Level::Pi_, NuclearMass::QValue(), and uconv.

7.6.2 Member Function Documentation

7.6.2.1 std::pair< double, double > CrossSection::CalcAverageDWaveResWidth ( )

References pi.

Referenced by main().

7.6.2.2 std::pair< double, double > CrossSection::CalcAveragePWaveResWidth ( )

References pi.

Referenced by main().

7.6.2.3 std::pair< double, double > CrossSection::CalcAverageSWaveResWidth ( )

References pi.

Referenced by main().

7.6.2.4 void CrossSection::Calculate ( )

References TransitionRateFunc::SetGammaCutoffEnergy().

Referenced by main().

7.6.2.5 void CrossSection::CalculateReactionRates (bool *macs*)

References avagadroNum, boltzConst, gsl_reactionrate_integrand(), lightSpeedInCmPerS, pi, and uconv.

Referenced by main().

7.6.2.6 void CrossSection::CreateMACSEnergiesVector ( ) [static]

Referenced by Initialize().

7.6.2.7 void CrossSection::CreateTempVector ( ) [static]

Referenced by Initialize().

7.6.2.8 bool CrossSection::IsValid ( ) const [inline]

Referenced by main().

7.6.2.9 void CrossSection::PrintCrossSections ( )

References NuclearMass::FindElement().

Referenced by main().

7.6.2.10 void CrossSection::PrintReactionRates (bool *macs*)

References NuclearMass::FindElement().

Referenced by main().

7.6.2.11 void CrossSection::PrintTransmissionTerms ( )

References NuclearMass::FindElement().

Referenced by main().

7.6.2.12 static void CrossSection::SetCalculateGammaCutoff (bool *calc*) [inline], [static]

Referenced by Initialize(), and parseCommandLineForOptions().

7.6.2.13 static void CrossSection::SetResidualAlpha (bool *residual*) [inline], [static]

Referenced by Initialize(), and parseCommandLineForOptions().

7.6.2.14 static void CrossSection::SetResidualGamma (bool *residual*) [inline], [static]

Referenced by Initialize(), and parseCommandLineForOptions().

7.6.2.15 static void CrossSection::SetResidualNeutron (bool *residual*) [inline], [static]

Referenced by Initialize(), and parseCommandLineForOptions().

7.6.2.16 static void CrossSection::SetResidualProton (bool *residual*) [inline], [static]

Referenced by Initialize(), and parseCommandLineForOptions().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/CrossSection.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/CrossSection.cpp
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Setup.cpp

7.7 CrossSectionValues Class Reference

`#include <CrossSection.h>`

Public Member Functions

- CrossSectionValues (double gamma, double neutron, double proton, double alpha, double gammaStellar, double neutronStellar, double protonStellar, double alphaStellar)

Public Attributes

- double gamma_
- double neutron_
- double proton_
- double alpha_
- double gammaStellar_
- double neutronStellar_
- double protonStellar_
- double alphaStellar_

7.7.1 Constructor & Destructor Documentation

7.7.1.1 CrossSectionValues::CrossSectionValues (double *gamma,* double *neutron,* double *proton,* double *alpha,* double *gammaStellar,* double *neutronStellar,* double *protonStellar,* double *alphaStellar*) [inline]

7.7.2 Member Data Documentation

7.7.2.1 double CrossSectionValues::alpha_

7.7.2.2 double CrossSectionValues::alphaStellar_

7.7.2.3 double CrossSectionValues::gamma_

7.7.2.4 double CrossSectionValues::gammaStellar_

7.7.2.5 double CrossSectionValues::neutron_

7.7.2.6 double CrossSectionValues::neutronStellar_

7.7.2.7 double CrossSectionValues::proton_

7.7.2.8 double CrossSectionValues::protonStellar_

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/CrossSection.h

7.8 DecayController Class Reference

`#include <DecayController.h>`

Public Member Functions

- DecayController (int Z, int A, double jInitial, int piInitial, double energy, int initialNeutronNumber=-1, int initialNeutronHoleNumber=-1, int initialProtonNumber=-1, int initialProtonHoleNumber=-1)
- bool Decay (double &, double &, double &, double &, double &, double &, double &)
- std::vector< DecayProduct > DecayProducts () const
- void PrintDecays ()

7.8.1 Constructor & Destructor Documentation

7.8.1.1 DecayController::DecayController (int *Z,* int *A,* double *jInitial,* int *piInitial,* double *energy,* int *initialNeutronNumber* = −1, int *initialNeutronHoleNumber* = −1, int *initialProtonNumber* = −1, int *initialProtonHoleNumber* = −1) [inline]

7.8.2 Member Function Documentation

7.8.2.1 bool DecayController::Decay (double & *neutronEntrance,* double & *protonEntrance,* double & *alphaEntrance,* double & *gammaEntrance,* double & *neutronTotalWidth,* double & *protonTotalWidth,* double & *alphaTotalWidth,* double & *gammaTotalWidth*)

References Decayer::AlphaEntranceWidth(), Decayer::AlphaTotalWidth(), Decayer::Decay(), PreEqDecayer::Decay(), Decayer::GammaEntranceWidth(), Decayer::GammaTotalWidth(), Decayer::NeutronEntranceWidth(), Decayer::NeutronTotalWidth(), Decayer::ProtonEntranceWidth(), and Decayer::ProtonTotalWidth().

Referenced by main().

7.8.2.2 std::vector<DecayProduct> DecayController::DecayProducts ( ) const [inline]

Referenced by main().

7.8.2.3 void DecayController::PrintDecays ( )

Referenced by main().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/DecayController.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/DecayController.cpp

7.9 DecayData Class Reference

`#include <DecayProduct.h>`

Public Member Functions

- DecayData ()
- DecayData (double energy, double neutronEntranceWidth, double protonEntranceWidth, double alphaEntranceWidth, double gammaEntranceWidth, double neutronTotalWidth, double protonTotalWidth, double alphaTotalWidth, double gammaTotalWidth)
- double energy () const
- double neutronEntranceWidth () const
- double protonEntranceWidth () const
- double alphaEntranceWidth () const
- double gammaEntranceWidth () const
- double neutronTotalWidth () const
- double protonTotalWidth () const
- double alphaTotalWidth () const
- double gammaTotalWidth () const

7.9.1 Constructor & Destructor Documentation

7.9.1.1 DecayData::DecayData ( ) [inline]

7.9.1.2 DecayData::DecayData (double *energy,* double *neutronEntranceWidth,* double *protonEntranceWidth,* double *alphaEntranceWidth,* double *gammaEntranceWidth,* double *neutronTotalWidth,* double *protonTotalWidth,* double *alphaTotalWidth,* double *gammaTotalWidth*) [inline]

7.9.2 Member Function Documentation

7.9.2.1 double DecayData::alphaEntranceWidth ( ) const [inline]

7.9.2.2 double DecayData::alphaTotalWidth ( ) const [inline]

7.9.2.3 double DecayData::energy ( ) const [inline]

7.9.2.4 double DecayData::gammaEntranceWidth ( ) const [inline]

7.9.2.5 double DecayData::gammaTotalWidth ( ) const [inline]

7.9.2.6 double DecayData::neutronEntranceWidth ( ) const [inline]

7.9.2.7 double DecayData::neutronTotalWidth ( ) const [inline]

7.9.2.8 double DecayData::protonEntranceWidth ( ) const [inline]

7.9.2.9 double DecayData::protonTotalWidth ( ) const [inline]

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/DecayProduct.h

7.10 Decayer Class Reference

`#include <Decayer.h>`

Public Member Functions

- Decayer (int Z, int A, double jInitial, int piInitial, double energy, double totalWidthForCorrection=0., double uncorrTotalWidthForCorrection=0., double uncorrTotalWidthSqrdForCorrection=0., Decayer *widthCorrectedDecayer=NULL)
- ~Decayer ()
- bool Decay (int &, int &, double &, int &, double &, double &)
- void PrintFunctions ()
- void PrintCDF ()
- void CorrectWidthFluctuations ()
- double NeutronEntranceWidth () const
- double ProtonEntranceWidth () const
- double AlphaEntranceWidth () const
- double GammaEntranceWidth () const
- double GammaTotalWidth () const
- double NeutronTotalWidth () const
- double AlphaTotalWidth () const
- double ProtonTotalWidth () const

Static Public Member Functions

- static void SetCrossSection (bool isCrossSection)
- static void SetMaxL (double maxL)
- static double GetMaxL ()

Friends

- class CrossSection

7.10.1 Constructor & Destructor Documentation

7.10.1.1 Decayer::Decayer (int *Z,* int *A,* double *jInitial,* int *piInitial,* double *energy,* double *totalWidthForCorrection* = 0., double *uncorrTotalWidthForCorrection* = 0., double *uncorrTotalWidthSqrdForCorrection* = 0., Decayer * *widthCorrectedDecayer* = NULL)

References NuclearLevels::FindLevels(), TransitionRateFunc::GroundStateTransmission(), TransitionRateFunc::Integral(), and NuclearMass::QValue().

Referenced by CorrectWidthFluctuations().

7.10.1.2 Decayer::~Decayer ( )

7.10.2 Member Function Documentation

7.10.2.1 double Decayer::AlphaEntranceWidth ( ) const [inline]

Referenced by DecayController::Decay().

7.10.2.2 double Decayer::AlphaTotalWidth ( ) const [inline]

Referenced by DecayController::Decay().

7.10.2.3 void Decayer::CorrectWidthFluctuations ( )

References Decayer().

7.10.2.4 bool Decayer::Decay (int & *Z,* int & *A,* double & *jFinal,* int & *piFinal,* double & *excitationEnergy,* double & *decayEnergy*)

References randomSeed.

Referenced by DecayController::Decay().

7.10.2.5 double Decayer::GammaEntranceWidth ( ) const [inline]

Referenced by DecayController::Decay().

7.10.2.6 double Decayer::GammaTotalWidth ( ) const [inline]

Referenced by DecayController::Decay().

7.10.2.7 static double Decayer::GetMaxL ( ) [inline], [static]

Referenced by parseCommandLineForOptions().

7.10.2.8 double Decayer::NeutronEntranceWidth ( ) const [inline]

Referenced by DecayController::Decay().

7.10.2.9 double Decayer::NeutronTotalWidth ( ) const [inline]

Referenced by DecayController::Decay().

7.10.2.10 void Decayer::PrintCDF ( )

7.10.2.11 void Decayer::PrintFunctions ( )

7.10.2.12 double Decayer::ProtonEntranceWidth ( ) const [inline]

Referenced by DecayController::Decay().

7.10.2.13 double Decayer::ProtonTotalWidth ( ) const [inline]

Referenced by DecayController::Decay().

7.10.2.14 static void Decayer::SetCrossSection (bool *isCrossSection*) [inline], [static]

Referenced by Initialize(), and main().

7.10.2.15 static void Decayer::SetMaxL (double *maxL*) [inline], [static]

Referenced by Initialize(), and parseCommandLineForOptions().

7.10.3 Friends And Related Function Documentation

7.10.3.1 friend class CrossSection [friend]

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/Decayer.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Decayer.cpp
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Setup.cpp

7.11 DecayProduct Class Reference

`#include <DecayProduct.h>`

Public Member Functions

- DecayProduct ()
- DecayProduct (int Z, int A, double J, int Pi, double excitationEnergy, double fragmentEnergyCM, double fragmentEnergy, double fragmentMomentumX, double fragmentMomentumY, double fragmentMomentumZ, int decayType, double particleThetaCM, double particlePhiCM, double particleEnergyCM, double particleEnergy, double particleMomentumX, double particleMomentumY, double particleMomentumZ)

Public Attributes

- int Z_
- int A_

- int Pi_
- int particleType_
- double J_
- double excitationEnergy_
- double fragmentEnergyCM_
- double fragmentEnergy_
- double fragmentMomentumX_
- double fragmentMomentumY_
- double fragmentMomentumZ_
- double particleThetaCM_
- double particlePhiCM_
- double particleEnergyCM_
- double particleEnergy_
- double particleMomentumX_
- double particleMomentumY_
- double particleMomentumZ_

7.11.1 Constructor & Destructor Documentation

7.11.1.1 DecayProduct::DecayProduct ( ) [inline]

7.11.1.2 DecayProduct::DecayProduct (int *Z,* int *A,* double *J,* int *Pi,* double *excitationEnergy,* double *fragmentEnergyCM,* double *fragmentEnergy,* double *fragmentMomentumX,* double *fragmentMomentumY,* double *fragmentMomentumZ,* int *decayType,* double *particleThetaCM,* double *particlePhiCM,* double *particleEnergyCM,* double *particleEnergy,* double *particleMomentumX,* double *particleMomentumY,* double *particleMomentumZ*) [inline]

7.11.2 Member Data Documentation

7.11.2.1 int DecayProduct::A_

Referenced by boost::serialization::serialize().

7.11.2.2 double DecayProduct::excitationEnergy_

Referenced by boost::serialization::serialize().

7.11.2.3 double DecayProduct::fragmentEnergy_

Referenced by boost::serialization::serialize().

7.11.2.4 double DecayProduct::fragmentEnergyCM_

Referenced by boost::serialization::serialize().

7.11.2.5 double DecayProduct::fragmentMomentumX_

Referenced by boost::serialization::serialize().

7.11.2.6 double DecayProduct::fragmentMomentumY_

Referenced by boost::serialization::serialize().

7.11.2.7 double DecayProduct::fragmentMomentumZ_

Referenced by boost::serialization::serialize().

7.11.2.8 double DecayProduct::J_

Referenced by boost::serialization::serialize().

7.11.2.9 double DecayProduct::particleEnergy_

Referenced by boost::serialization::serialize().

7.11.2.10 double DecayProduct::particleEnergyCM_

Referenced by boost::serialization::serialize().

7.11.2.11 double DecayProduct::particleMomentumX_

Referenced by boost::serialization::serialize().

7.11.2.12 double DecayProduct::particleMomentumY_

Referenced by boost::serialization::serialize().

7.11.2.13 double DecayProduct::particleMomentumZ_

Referenced by boost::serialization::serialize().

7.11.2.14 double DecayProduct::particlePhiCM_

Referenced by boost::serialization::serialize().

7.11.2.15 double DecayProduct::particleThetaCM_

Referenced by boost::serialization::serialize().

7.11.2.16 int DecayProduct::particleType_

Referenced by boost::serialization::serialize().

7.11.2.17 int DecayProduct::Pi_

Referenced by boost::serialization::serialize().

7.11.2.18 int DecayProduct::Z_

Referenced by boost::serialization::serialize().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/DecayProduct.h

7.12 DecayResults Class Reference

`#include <DecayResults.h>`

Public Member Functions

- DecayResults (int, int, double, int, double, double, int)
- ~DecayResults ()
- void AddResults (std::vector< std::pair< DecayData, std::vector< DecayProduct > > > &)

7.12.1 Constructor & Destructor Documentation

7.12.1.1 DecayResults::DecayResults (int *Z,* int *A,* double *J,* int *Pi,* double *initialEnergyLow,* double *initialEnergyHigh,* int *suffixNo*)

References NuclearMass::FindElement().

7.12.1.2 DecayResults::~DecayResults ( )

7.12.2 Member Function Documentation

7.12.2.1 void DecayResults::AddResults (std::vector< std::pair< DecayData, std::vector< DecayProduct > > > & *results*)

Referenced by main().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/DecayResults.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/DecayResults.cpp

7.13 EntrancePairs Struct Reference

Public Member Functions

- EntrancePairs (int Z, int A, int pType)

Public Attributes

- int Z_
- int A_
- int pType_

7.13.1 Constructor & Destructor Documentation

7.13.1.1 EntrancePairs::EntrancePairs (int *Z,* int *A,* int *pType*) [inline]

References A_, pType_, and Z_.

7.13.2 Member Data Documentation

7.13.2.1 int EntrancePairs::A_

Referenced by EntrancePairs().

7.13.2.2 int EntrancePairs::pType_

Referenced by EntrancePairs().

7.13.2.3 int EntrancePairs::Z_

Referenced by EntrancePairs().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Sapphire.cpp

7.14 std::equal_to< MassKey > Struct Template Reference

`#include <NuclearMass.h>`

Public Member Functions

- bool operator() (MassKey const &left, MassKey const &right) const

7.14.1 Member Function Documentation

7.14.1.1 bool std::equal_to< MassKey >::operator() (MassKey const & *left*, MassKey const & *right*) const [inline]

References MassKey::A_, and MassKey::Z_.

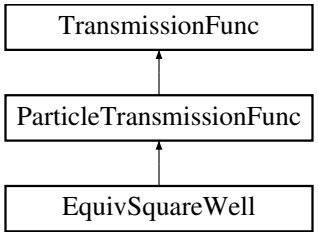
The documentation for this struct was generated from the following file:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearMass.h

7.15 EquivSquareWell Class Reference

`#include <EquivSquareWell.h>`

Inheritance diagram for EquivSquareWell:



Public Member Functions

- EquivSquareWell (int z1, int m1, int z2, int m2, double jInitial, int piInitial, double jFinal, int piFinal, double spin, int parity, double maxL, double totalWidthForCorrection, double uncorrTotalWidthForCorrection, double uncorrTotalWidthSqrdForCorrection, TransmissionFunc *previous)
- ~EquivSquareWell ()
- double CalcTransmission (double, int, double)

Additional Inherited Members

7.15.1 Constructor & Destructor Documentation

7.15.1.1 EquivSquareWell::EquivSquareWell (int *z1,* int *m1,* int *z2,* int *m2,* double *jInitial,* int *piInitial,* double *jFinal,* int *piFinal,* double *spin,* int *parity,* double *maxL,* double *totalWidthForCorrection,* double *uncorrTotalWidthForCorrection,* double *uncorrTotalWidthSqrdForCorrection,* TransmissionFunc * *previous*) [inline]

References ParticleTransmissionFunc::redmass_, ParticleTransmissionFunc::z1_, and TransmissionFunc::z2_.

7.15.1.2 EquivSquareWell::~EquivSquareWell ( ) [inline]

7.15.2 Member Function Documentation

7.15.2.1 double EquivSquareWell::CalcTransmission (double *s,* int *l,* double *energy*) [virtual]

Implements ParticleTransmissionFunc.

References hbarc, CoulFunc::Penetrability(), pi, ParticleTransmissionFunc::redmass_, and uconv.

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/EquivSquareWell.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/EquivSquareWell.cpp

7.16 GammaTransition Class Reference

`#include <NuclearLevels.h>`

Public Member Functions

- GammaTransition (int levelIndex, double energy, double probability)

Public Attributes

- int levelIndex_
- double energy_
- double probability_

7.16.1 Constructor & Destructor Documentation

7.16.1.1 GammaTransition::GammaTransition (int *levelIndex,* double *energy,* double *probability*) [inline]

7.16.2 Member Data Documentation

7.16.2.1 double GammaTransition::energy_

7.16.2.2 int GammaTransition::levelIndex_

7.16.2.3 double GammaTransition::probability_

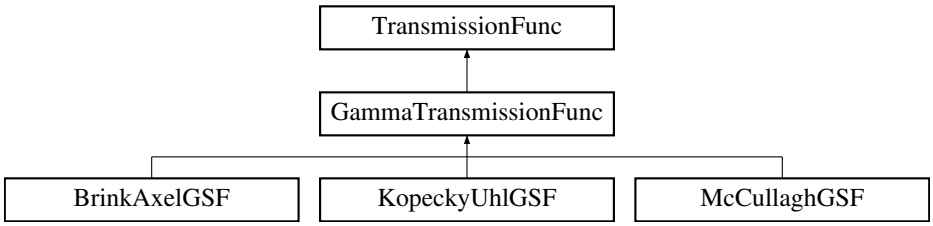
The documentation for this class was generated from the following file:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearLevels.h

7.17 GammaTransmissionFunc Class Reference

`#include <GammaTransmissionFunc.h>`

Inheritance diagram for GammaTransmissionFunc:



Public Member Functions

- GammaTransmissionFunc (int, int, double, int, double, int, double, double, double, double, TransmissionFunc ∗)
- virtual ∼GammaTransmissionFunc ()
- bool IsValid ()
- double operator() (double)
- virtual double CalcStrengthFunction (double)=0

Static Public Member Functions

- static GammaTransmissionFunc ∗ CreateGammaTransmissionFunc (int, int, double, int, double, int, double, LevelDensity ∗, double, double, double, TransmissionFunc ∗, double)
- static void InitializeGDRParameters (std::string)
- static void SetEGDRType (int)
- static void SetPorterThomas (bool)

Protected Attributes

- GDRParameters gdrParameters_

Static Protected Attributes

- static GDRTable gdrTable_
- static int egdrType_
- static const int mgdrType_ =0
- static const int egqrType_ =0
- static bool porterThomas_

7.17.1 Constructor & Destructor Documentation

7.17.1.1 GammaTransmissionFunc::GammaTransmissionFunc (int *z2,* int *m2,* double *jInitial,* int *piInitial,* double *jFinal,* int *piFinal,* double *maxL,* double *totalWidthForCorrection,* double *uncorrTotalWidthForCorrection,* double *uncorrTotalWidthSqrdForCorrection,* **TransmissionFunc** * *previous*)

References NuclearMass::FindMass(), fstruc, gdrParameters_, gdrTable_, hbarc, GDRParameters::kSigmaGamma_, TransmissionFunc::m2_, pi, and TransmissionFunc::z2_.

7.17.1.2 virtual GammaTransmissionFunc::~GammaTransmissionFunc ( ) [inline], [virtual]

7.17.2 Member Function Documentation

7.17.2.1 virtual double GammaTransmissionFunc::CalcStrengthFunction (double) [pure virtual]

Implemented in BrinkAxelGSF, KopeckyUhlGSF, and McCullaghGSF.

Referenced by operator()().

7.17.2.2 GammaTransmissionFunc * GammaTransmissionFunc::CreateGammaTransmissionFunc (int *z2,* int *m2,* double *jInitial,* int *piInitial,* double *jFinal,* int *piFinal,* double *maxL,* **LevelDensity** * *levelDensity,* double *totalWidthForCorrection,* double *uncorrTotalWidthForCorrection,* double *uncorrTotalWidthSqrdForCorrection,* **TransmissionFunc** * *previous,* double *compoundE*) [static]

References egdrType_, egqrType_, and mgdrType_.

Referenced by TransitionRateFunc::TransitionRateFunc().

7.17.2.3 void GammaTransmissionFunc::InitializeGDRParameters (std::string *filename*) [static]

References gdrTable_.

Referenced by Initialize().

7.17.2.4 bool GammaTransmissionFunc::IsValid ( ) [inline], [virtual]

Implements TransmissionFunc.

7.17.2.5 double GammaTransmissionFunc::operator() (double *energy*) [virtual]

Implements TransmissionFunc.

References CalcStrengthFunction(), TransmissionFunc::maxL_, pi, porterThomas_, TransmissionFunc::previous_, randomSeed, TransmissionFunc::totalWidthForCorrection_, TransmissionFunc::uncorrTotalWidthForCorrection_, and TransmissionFunc::uncorrTotalWidthSqrdForCorrection_.

7.17.2.6 void GammaTransmissionFunc::SetEGDRType (int *type*) [static]

References egdrType_.

Referenced by Initialize(), and parseCommandLineForOptions().

7.17.2.7 void GammaTransmissionFunc::SetPorterThomas (bool *yn*) [static]

References porterThomas_.

Referenced by Initialize(), and parseCommandLineForOptions().

7.17.3 Member Data Documentation

7.17.3.1 int GammaTransmissionFunc::egdrType_ [static], [protected]

Referenced by CreateGammaTransmissionFunc(), and SetEGDRType().

7.17.3.2 const int GammaTransmissionFunc::egqrType_ =0 [static], [protected]

Referenced by CreateGammaTransmissionFunc().

7.17.3.3 GDRParameters GammaTransmissionFunc::gdrParameters_ [protected]

Referenced by McCullaghGSF::CalcStrengthFunction(), KopeckyUhlGSF::CalcStrengthFunction(), BrinkAxelGSF-::CalcStrengthFunction(), and GammaTransmissionFunc().

7.17.3.4 GDRTable GammaTransmissionFunc::gdrTable_ [static], [protected]

Referenced by GammaTransmissionFunc(), and InitializeGDRParameters().

7.17.3.5 const int GammaTransmissionFunc::mgdrType_ =0 [static], [protected]

Referenced by CreateGammaTransmissionFunc().

7.17.3.6 bool GammaTransmissionFunc::porterThomas_ [static], [protected]

Referenced by operator()(), and SetPorterThomas().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/GammaTransmissionFunc.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/GammaTransmissionFunc.cpp
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Setup.cpp

7.18 GDRParameters Class Reference

`#include <GammaTransmissionFunc.h>`

Public Member Functions

- GDRParameters ()
- GDRParameters (double eta, double E1, double W1, double kSigmaGamma1, double E2, double W2, double kSigmaGamma2)

Public Attributes

- double eta_
- double E_ [2]
- double W_ [2]
- double kSigmaGamma_ [2]

7.18.1 Constructor & Destructor Documentation

7.18.1.1 GDRParameters::GDRParameters ( ) [inline]

References E_, and W_.

7.18.1.2 GDRParameters::GDRParameters (double *eta,* double *E1,* double *W1,* double *kSigmaGamma1,* double *E2,* double *W2,* double *kSigmaGamma2*) [inline]

References E_, kSigmaGamma_, and W_.

7.18.2 Member Data Documentation

7.18.2.1 double GDRParameters::E_[2]

Referenced by McCullaghGSF::CalcStrengthFunction(), KopeckyUhlGSF::CalcStrengthFunction(), BrinkAxelGSF-::CalcStrengthFunction(), and GDRParameters().

7.18.2.2 double GDRParameters::eta_

7.18.2.3 double GDRParameters::kSigmaGamma_[2]

Referenced by McCullaghGSF::CalcStrengthFunction(), KopeckyUhlGSF::CalcStrengthFunction(), BrinkAxelGSF-::CalcStrengthFunction(), GammaTransmissionFunc::GammaTransmissionFunc(), and GDRParameters().

7.18.2.4 double GDRParameters::W_[2]

Referenced by KopeckyUhlGSF::CalcStrengthFunction(), and GDRParameters().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/GammaTransmissionFunc.h

7.19 gsl_partfunc_params Struct Reference

Public Attributes

- double temperature
- LevelDensity * density

7.19.1 Member Data Documentation

7.19.1.1 LevelDensity* gsl_partfunc_params::density

Referenced by gsl_partfunc_integrand(), LevelDensity::operator()(), and LevelDensity::TotalLevelDensity().

7.19.1.2 double gsl_partfunc_params::temperature

Referenced by gsl_partfunc_integrand().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/CrossSection.cpp

7.20 gsl_reactionrate_params Struct Reference

Public Attributes

- double temperature
- TGraph * graph
- bool useSpline

7.20.1 Member Data Documentation

7.20.1.1 TGraph* gsl_reactionrate_params::graph

Referenced by gsl_reactionrate_integrand().

7.20.1.2 double gsl_reactionrate_params::temperature

Referenced by gsl_reactionrate_integrand().

7.20.1.3 bool gsl_reactionrate_params::useSpline

Referenced by gsl_reactionrate_integrand().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/CrossSection.cpp

7.21 std::tr1::hash< MassKey > Struct Template Reference

`#include <NuclearMass.h>`

Public Member Functions

- std::size_t operator() (MassKey const &key) const

7.21.1 Member Function Documentation

7.21.1.1 std::size_t std::tr1::hash< MassKey >::operator() (MassKey const & *key*) const [inline]

References MassKey::A_, and MassKey::Z_.

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearMass.h

7.22 InitialNucleusData Class Reference

`#include <SapphireMPITypes.h>`

Public Member Functions

- InitialNucleusData ()
- InitialNucleusData (int Z, int A, double J, int Pi, double lowEnergy, double highEnergy, bool preEq)
- bool preEq () const
- int Z () const
- int A () const
- int Pi () const
- double J () const
- double lowEnergy () const
- double highEnergy () const

Friends

- class boost::serialization::access

7.22.1 Constructor & Destructor Documentation

7.22.1.1 InitialNucleusData::InitialNucleusData ( ) [inline]

7.22.1.2 InitialNucleusData::InitialNucleusData (int *Z,* int *A,* double *J,* int *Pi,* double *lowEnergy,* double *highEnergy,* bool *preEq*) [inline]

7.22.2 Member Function Documentation

7.22.2.1 int InitialNucleusData::A ( ) const [inline]

7.22.2.2 double InitialNucleusData::highEnergy ( ) const [inline]

7.22.2.3 double InitialNucleusData::J ( ) const [inline]

7.22.2.4 double InitialNucleusData::lowEnergy ( ) const [inline]

7.22.2.5 int InitialNucleusData::Pi ( ) const [inline]

7.22.2.6 bool InitialNucleusData::preEq ( ) const [inline]

7.22.2.7 int InitialNucleusData::Z ( ) const [inline]

7.22.3 Friends And Related Function Documentation

7.22.3.1 friend class boost::serialization::access [friend]

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/SapphireMPITypes.h

7.23 int_double_pair_compare Struct Reference

```
#include <CrossSection.h>
```

Public Member Functions

- bool operator() (const int_double_pair &lhs, int_double_pair const &rhs)

7.23.1 Member Function Documentation

7.23.1.1 bool int_double_pair_compare::operator() (const int_double_pair & *lhs*, int_double_pair const & *rhs*) [inline]

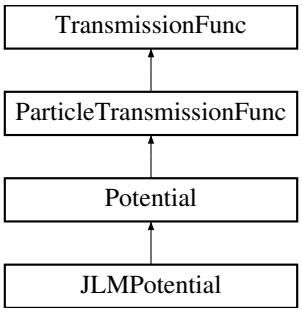
The documentation for this struct was generated from the following file:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/CrossSection.h

7.24 JLMPotential Class Reference

`#include <JLMPotential.h>`

Inheritance diagram for JLMPotential:



Public Member Functions

- JLMPotential (int, int, int, int, double, int, double, int, double, int, double, double, double, double, TransmissionFunc ∗)
- double CalculateDensity (double, int) const
- std::complex< double > Calculate (double, int, double, double, double) const

Static Public Member Functions

- static double GetA (int i, int j)

Additional Inherited Members

7.24.1 Constructor & Destructor Documentation

7.24.1.1 JLMPotential::JLMPotential (int *z1*, int *m1*, int *z2*, int *m2*, double *jInitial*, int *piInitial*, double *jFinal*, int *piFinal*, double *spin*, int *parity*, double *maxL*, double *totalWidthForCorrection*, double *uncorrTotalWidthForCorrection*, double *uncorrTotalWidthSqrdForCorrection*, TransmissionFunc ∗ *previous*)

References Potential::boundaryRadius_, Potential::coulombRadius_, TransmissionFunc::m2_, pi, and TransmissionFunc::z2_.

7.24.2 Member Function Documentation

7.24.2.1 std::complex< double > JLMPotential::Calculate (double *r*, int *l*, double *s*, double *j*, double *E*) const [virtual]

Implements Potential.

References CalculateDensity(), Potential::coulombRadius_, fstruc, Potential::GetZ1Z2(), and hbarc.

7.24.2.2 double JLMPotential::CalculateDensity (double *r*, int *which*) const

Referenced by Calculate().

7.24.2.3 static double JLMPotential::GetA (int *i*, int *j*) [inline], [static]

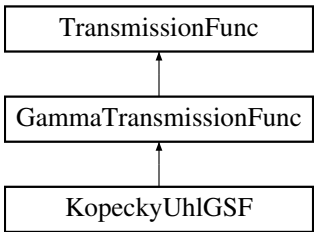
The documentation for this class was generated from the following files:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/JLMPotential.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/JLMPotential.cpp

7.25 KopeckyUhlGSF Class Reference

`#include <KopeckyUhlGSF.h>`

Inheritance diagram for KopeckyUhlGSF:



Public Member Functions

- KopeckyUhlGSF (int, int, double, int, double, int, double, double, double, double, TransmissionFunc ∗, LevelDensity ∗, double)
- double CalcStrengthFunction (double)

Additional Inherited Members

7.25.1 Constructor & Destructor Documentation

7.25.1.1 KopeckyUhlGSF::KopeckyUhlGSF (int *z2*, int *m2*, double *jInitial*, int *piInitial*, double *jFinal*, int *piFinal*, double *maxL*, double *totalWidthForCorrection*, double *uncorrTotalWidthForCorrection*, double *uncorrTotalWidthSqrdForCorrection*, TransmissionFunc ∗ *previous*, LevelDensity ∗ *levelDensity*, double *compoundE*)

References LevelDensity::backshift_, LevelDensity::CalcDensityParam(), and NuclearMass::QValue().

7.25.2 Member Function Documentation

7.25.2.1 double KopeckyUhlGSF::CalcStrengthFunction (double *energy*) [virtual]

Implements GammaTransmissionFunc.

References GDRParameters::E_, GammaTransmissionFunc::gdrParameters_, GDRParameters::kSigmaGamma_, pi, and GDRParameters::W_.

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/KopeckyUhlGSF.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/KopeckyUhlGSF.cpp

7.26 Level Class Reference

`#include <NuclearLevels.h>`

Public Member Functions

- Level (double J, int Pi, double energy)

Public Attributes

- int Pi_
- double J_
- double energy_
- std::vector< GammaTransition > gammas_

7.26.1 Constructor & Destructor Documentation

7.26.1.1 Level::Level (double *J,* int *Pi,* double *energy*) [inline]

7.26.2 Member Data Documentation

7.26.2.1 double Level::energy_

Referenced by CrossSection::CrossSection().

7.26.2.2 std::vector<GammaTransition> Level::gammas_

Referenced by LevelsContainer::LevelsContainer().

7.26.2.3 double Level::J_

Referenced by CrossSection::CrossSection().

7.26.2.4 int Level::Pi_

Referenced by CrossSection::CrossSection().

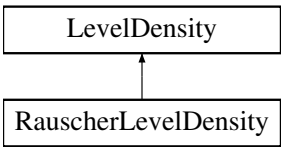
The documentation for this class was generated from the following file:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearLevels.h

7.27 LevelDensity Class Reference

`#include <LevelDensity.h>`

Inheritance diagram for LevelDensity:



Public Member Functions

- LevelDensity (int Z, int A, double J)
- virtual ~LevelDensity ()
- double operator() (double)
- double TotalLevelDensity (double)

Protected Member Functions

- virtual void CalcBackShift ()=0
- virtual double CalcDensityParam (double)=0
- virtual double CalcNuclearTemp (double)=0
- void CalcConstantTempTerms ()

Protected Attributes

- int Z_
- int A_
- double J_
- double backshift_
- double criticalU_
- double constAngTerm_
- double nuclearTemp_
- double e0_

Static Protected Attributes

- static constexpr double zeta_ = 1.0
- static constexpr double r0_ = 1.25

Friends

- class KopeckyUhlGSF

7.27.1 Constructor & Destructor Documentation

7.27.1.1 LevelDensity::LevelDensity (int *Z*, int *A*, double *J*) [inline]

References criticalU_.

7.27.1.2 virtual LevelDensity::~LevelDensity ( ) [inline], [virtual]

7.27.2 Member Function Documentation

7.27.2.1 virtual void LevelDensity::CalcBackShift ( ) [protected], [pure virtual]

Implemented in RauscherLevelDensity.

7.27.2.2 void LevelDensity::CalcConstantTempTerms ( ) [protected]

References A_, backshift_, CalcDensityParam(), CalcNuclearTemp(), constAngTerm_, criticalU_, e0_, hbarc, J_, nuclearTemp_, r0_, uconv, and zeta_.

Referenced by RauscherLevelDensity::RauscherLevelDensity().

7.27.2.3 virtual double LevelDensity::CalcDensityParam (double) [protected], [pure virtual]

Implemented in RauscherLevelDensity.

Referenced by CalcConstantTempTerms(), KopeckyUhlGSF::KopeckyUhlGSF(), operator()(), and TotalLevelDensity().

7.27.2.4 virtual double LevelDensity::CalcNuclearTemp (double) [protected], [pure virtual]

Implemented in RauscherLevelDensity.

Referenced by CalcConstantTempTerms().

7.27.2.5 double LevelDensity::operator() (double *E*)

References A_, backshift_, CalcDensityParam(), constAngTerm_, criticalU_, gsl_partfunc_params::density, e0_, hbarc, J_, nuclearTemp_, r0_, uconv, and zeta_.

7.27.2.6 double LevelDensity::TotalLevelDensity (double *E*)

References A_, backshift_, CalcDensityParam(), criticalU_, gsl_partfunc_params::density, e0_, hbarc, nuclearTemp_, r0_, uconv, and zeta_.

Referenced by TransitionRateFunc::CalcTotalLevelDensity().

7.27.3 Friends And Related Function Documentation

7.27.3.1 friend class KopeckyUhlGSF [friend]

7.27.4 Member Data Documentation

7.27.4.1 int LevelDensity::A_ [protected]

Referenced by RauscherLevelDensity::CalcBackShift(), CalcConstantTempTerms(), operator()(), and TotalLevelDensity().

7.27.4.2 double LevelDensity::backshift_ [protected]

Referenced by RauscherLevelDensity::CalcBackShift(), CalcConstantTempTerms(), KopeckyUhlGSF::KopeckyUhlGSF(), operator()(), and TotalLevelDensity().

7.27.4.3 double LevelDensity::constAngTerm_ [protected]

Referenced by CalcConstantTempTerms(), and operator()().

7.27.4.4 double LevelDensity::criticalU_ [protected]

Referenced by CalcConstantTempTerms(), LevelDensity(), operator()(), and TotalLevelDensity().

7.27.4.5 double LevelDensity::e0_ [protected]

Referenced by CalcConstantTempTerms(), operator()(), and TotalLevelDensity().

7.27.4.6 double LevelDensity::J_ [protected]

Referenced by CalcConstantTempTerms(), and operator()().

7.27.4.7 double LevelDensity::nuclearTemp_ [protected]

Referenced by CalcConstantTempTerms(), operator()(), and TotalLevelDensity().

7.27.4.8 constexpr double LevelDensity::r0_ = 1.25 [static], [protected]

Referenced by CalcConstantTempTerms(), operator()(), and TotalLevelDensity().

7.27.4.9 int LevelDensity::Z_ [protected]

Referenced by RauscherLevelDensity::CalcBackShift().

7.27.4.10 constexpr double LevelDensity::zeta_ = 1.0 [static], [protected]

Referenced by CalcConstantTempTerms(), operator()(), and TotalLevelDensity().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/LevelDensity.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/LevelDensity.cpp

7.28 LevelsContainer Class Reference

`#include <NuclearLevels.h>`

Public Member Functions

- LevelsContainer ()
- LevelsContainer (std::istream &, int, int)

Public Attributes

- std::vector< Level > levels_

7.28.1 Constructor & Destructor Documentation

7.28.1.1 LevelsContainer::LevelsContainer ( ) [inline]

7.28.1.2 LevelsContainer::LevelsContainer (std::istream & *in,* int *numLevels,* int *numComplete*)

References Level::gammas_, and levels_.

7.28.2 Member Data Documentation

7.28.2.1 std::vector<Level> LevelsContainer::levels_

Referenced by NuclearLevels::InitializeLevels(), and LevelsContainer().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearLevels.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/NuclearLevels.cpp

7.29 MassEntry Class Reference

`#include <NuclearMass.h>`

Public Member Functions

- MassEntry ()
- MassEntry (double expMass, double thMass, double microEnergyCorr, unsigned int mask)

Public Attributes

- double expMass_
- double thMass_
- double microEnergyCorr_
- unsigned int mask_

7.29.1 Constructor & Destructor Documentation

7.29.1.1 MassEntry::MassEntry ( ) [inline]

References expMass_, mask_, microEnergyCorr_, and thMass_.

7.29.1.2 MassEntry::MassEntry (double *expMass,* double *thMass,* double *microEnergyCorr,* unsigned int *mask*) [inline]

7.29.2 Member Data Documentation

7.29.2.1 double MassEntry::expMass_

Referenced by MassEntry().

7.29.2.2 unsigned int MassEntry::mask_

Referenced by MassEntry().

7.29.2.3 double MassEntry::microEnergyCorr_

Referenced by MassEntry().

7.29.2.4 double MassEntry::thMass_

Referenced by MassEntry().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearMass.h

7.30 MassKey Class Reference

`#include <NuclearMass.h>`

Public Member Functions

- MassKey (int Z, int A)
- bool operator< (const MassKey &right) const

Public Attributes

- int Z_
- int A_

7.30.1 Constructor & Destructor Documentation

7.30.1.1 MassKey::MassKey (int *Z*, int *A*) [inline]

7.30.2 Member Function Documentation

7.30.2.1 bool MassKey::operator< (const MassKey & *right*) const [inline]

References A_, and Z_.

7.30.3 Member Data Documentation

7.30.3.1 int MassKey::A_

Referenced by std::tr1::hash< MassKey >::operator()(), std::equal_to< MassKey >::operator()(), and operator<().

7.30.3.2 int MassKey::Z_

Referenced by std::tr1::hash< MassKey >::operator()(), std::equal_to< MassKey >::operator()(), and operator<().

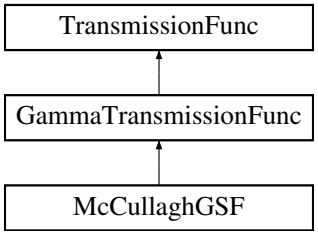
The documentation for this class was generated from the following file:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearMass.h

7.31 McCullaghGSF Class Reference

`#include <McCullaghGSF.h>`

Inheritance diagram for McCullaghGSF:



Public Member Functions

- McCullaghGSF (int, int, double, int, double, int, double, double, double, double, TransmissionFunc *)
- double CalcStrengthFunction (double)

Additional Inherited Members

7.31.1 Constructor & Destructor Documentation

7.31.1.1 McCullaghGSF::McCullaghGSF (int *z2*, int *m2*, double *jInitial*, int *piInitial*, double *jFinal*, int *piFinal*, double *maxL*, double *totalWidthForCorrection*, double *uncorrTotalWidthForCorrection*, double *uncorrTotalWidthSqrdForCorrection*, TransmissionFunc * *previous*)

7.31.2 Member Function Documentation

7.31.2.1 double McCullaghGSF::CalcStrengthFunction (double *energy*) [virtual]

Implements GammaTransmissionFunc.

References GDRParameters::E_, GammaTransmissionFunc::gdrParameters_, GDRParameters::kSigmaGamma_, and TransmissionFunc::maxL_.

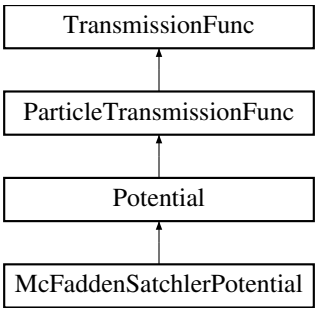
The documentation for this class was generated from the following files:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/McCullaghGSF.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/McCullaghGSF.cpp

7.32 McFaddenSatchlerPotential Class Reference

`#include <McFaddenSatchlerPotential.h>`

Inheritance diagram for McFaddenSatchlerPotential:



Public Member Functions

- McFaddenSatchlerPotential (int, int, int, int, double, int, double, int, double, int, double, double, double, double, TransmissionFunc ∗)
- std::complex< double > Calculate (double, int, double, double, double) const

Additional Inherited Members

7.32.1 Constructor & Destructor Documentation

7.32.1.1 **McFaddenSatchlerPotential::McFaddenSatchlerPotential (int *z1,* int *m1,* int *z2,* int *m2,* double *jInitial,* int *piInitial,* double *jFinal,* int *piFinal,* double *spin,* int *parity,* double *maxL,* double *totalWidthForCorrection,* double *uncorrTotalWidthForCorrection,* double *uncorrTotalWidthSqrdForCorrection,* TransmissionFunc ∗ *previous*)**

References Potential::boundaryRadius_, Potential::coulombRadius_, and TransmissionFunc::m2_.

7.32.2 Member Function Documentation

7.32.2.1 **std::complex< double > McFaddenSatchlerPotential::Calculate (double *r,* int *l,* double *s,* double *j,* double *energy*) const** `[virtual]`

Implements Potential.

References Potential::coulombRadius_, fstruc, Potential::GetZ1Z2(), and hbarc.

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/McFaddenSatchlerPotential.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/McFaddenSatchlerPotential.cpp

7.33 NuclearLevels Class Reference

`#include <NuclearLevels.h>`

Static Public Member Functions

- static void InitializeLevels (std::string levelsDirectory, std::string spinFile)
- static void PrintLevels (int, int)
- static std::vector< Level > FindLevels (int, int)

7.33.1 Member Function Documentation

7.33.1.1 std::vector< Level > NuclearLevels::FindLevels (int *Z*, int *A*) [static]

Referenced by CrossSection::CrossSection(), Decayer::Decayer(), and TransitionRateFunc::TransitionRateFunc().

7.33.1.2 void NuclearLevels::InitializeLevels (std::string *levelsDirectory*, std::string *spinFile*) [static]

References LevelsContainer::levels_.

Referenced by Initialize().

7.33.1.3 void NuclearLevels::PrintLevels (int *Z*, int *A*) [static]

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearLevels.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/NuclearLevels.cpp
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Setup.cpp

7.34 NuclearMass Class Reference

`#include <NuclearMass.h>`

Static Public Member Functions

- static void InitializeElements ()
- static void InitializeMasses (std::string)
- static int FindZ (std::string)
- static std::string FindElement (int)
- static bool FindMass (int, int, double &)
- static bool MassDifference (int, int, int, int, double &)
- static bool QValue (int, int, int, int, double &)
- static bool NeutronPairingGap (int, int, double &)
- static bool ProtonPairingGap (int, int, double &)
- static bool MicroEnergyCorr (int, int, double &)
- static bool HighestBoundEnergy (int, int, double &)
- static double CalculateLDMMass (int, int)

7.34.1 Member Function Documentation

7.34.1.1 double NuclearMass::CalculateLDMMass (int *Z*, int *A*) [static]

Calculates liquid drop model mass based on TALYS parametrization.

References uconv.

7.34.1.2 std::string NuclearMass::FindElement (int *Z*) [static]

Referenced by DecayResults::DecayResults(), CrossSection::PrintCrossSections(), CrossSection::PrintReactionRates(), and CrossSection::PrintTransmissionTerms().

7.34.1.3 bool NuclearMass::FindMass (int *Z,* int *A,* double & *M*) [static]

References HAS_EXP_MASS, and HAS_TH_MASS.

Referenced by GammaTransmissionFunc::GammaTransmissionFunc(), MassDifference(), ParticleTransmissionFunc::ParticleTransmissionFunc(), and QValue().

7.34.1.4 int NuclearMass::FindZ (std::string *element*) [static]

Referenced by parseCommandLineForDecay(), and parseCommandLineForXS().

7.34.1.5 bool NuclearMass::HighestBoundEnergy (int *Z,* int *A,* double & *energy*) [static]

References QValue().

Referenced by TransitionRateFunc::TransitionRateFunc().

7.34.1.6 void NuclearMass::InitializeElements ( ) [static]

Referenced by Initialize().

7.34.1.7 void NuclearMass::InitializeMasses (std::string *filename*) [static]

References HAS_EXP_MASS, HAS_TH_MASS, and uconv.

Referenced by Initialize().

7.34.1.8 bool NuclearMass::MassDifference (int *Z1,* int *A1,* int *Z2,* int *A2,* double & *difference*) [static]

References FindMass().

Referenced by NeutronPairingGap(), ProtonPairingGap(), and QValue().

7.34.1.9 bool NuclearMass::MicroEnergyCorr (int *Z,* int *A,* double & *correction*) [static]

Referenced by RauscherLevelDensity::RauscherLevelDensity().

7.34.1.10 bool NuclearMass::NeutronPairingGap (int *Z,* int *A,* double & *pairingGap*) [static]

References MassDifference().

Referenced by RauscherLevelDensity::CalcBackShift().

7.34.1.11 bool NuclearMass::ProtonPairingGap (int *Z,* int *A,* double & *pairingGap*) [static]

References MassDifference().

Referenced by RauscherLevelDensity::CalcBackShift().

7.34.1.12 bool NuclearMass::QValue (int *Z1,* int *A1,* int *Z2,* int *A2,* double & *qValue*) [static]

References FindMass(), and MassDifference().

Referenced by CrossSection::CrossSection(), Decayer::Decayer(), HighestBoundEnergy(), KopeckyUhlGSF::KopeckyUhlGSF(), ParticleHoleLevelDensity::ParticleHoleLevelDensity(), and PreEqDecayer::PreEqDecayer().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearMass.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/NuclearMass.cpp
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Setup.cpp

7.35 ODE_integration Class Reference

`#include <ode_int.H>`

Public Member Functions

- ODE_integration (const std::complex< double > &l_1, const std::complex< double > &two_eta_1)
- void operator() (const std::complex< double > &r0, const std::complex< double > &u0, const std::complex< double > &du0, const std::complex< double > &r, std::complex< double > &u, std::complex< double > &du) const

7.35.1 Constructor & Destructor Documentation

7.35.1.1 ODE_integration::ODE_integration (const std::complex< double > & *l_1*, const std::complex< double > & *two_eta_1*) [inline]

7.35.2 Member Function Documentation

7.35.2.1 void ODE_integration::operator() (const std::complex< double > & *r0*, const std::complex< double > & *u0*, const std::complex< double > & *du0*, const std::complex< double > & *r*, std::complex< double > & *u*, std::complex< double > & *du*) const

References inf_norm(), and precision.

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/include/ode_int.H
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/src/ode_int.cpp

7.36 ParticleHoleLevelDensity Class Reference

`#include <ParticleHoleLevelDensity.h>`

Public Member Functions

- ParticleHoleLevelDensity (int, int, double, int, int, int, int)
- double operator() (double energy, bool correct=true, bool spin=false)
- double PauliCorrection (int, int, int, int)
- double PairingCorrection (double energy)
- double gNu () const
- double gPi () const
- double FiniteDepth (double)

7.36.1 Constructor & Destructor Documentation

7.36.1.1 ParticleHoleLevelDensity::ParticleHoleLevelDensity (int *Z,* int *A,* double *J,* int *neutronNumber,* int *neutronHoleNumber,* int *protonNumber,* int *protonHoleNumber*)

References PauliCorrection(), and NuclearMass::QValue().

7.36.2 Member Function Documentation

7.36.2.1 double ParticleHoleLevelDensity::FiniteDepth (double *energy*)

Referenced by operator()().

7.36.2.2 double ParticleHoleLevelDensity::gNu ( ) const [inline]

7.36.2.3 double ParticleHoleLevelDensity::gPi ( ) const [inline]

7.36.2.4 double ParticleHoleLevelDensity::operator() (double *energy,* bool *correct* = `true`, bool *spin* = `false`)

References FiniteDepth(), and PairingCorrection().

7.36.2.5 double ParticleHoleLevelDensity::PairingCorrection (double *energy*)

Referenced by operator()().

7.36.2.6 double ParticleHoleLevelDensity::PauliCorrection (int *neutronNumber,* int *neutronHoleNumber,* int *protonNumber,* int *protonHoleNumber*)

Referenced by ParticleHoleLevelDensity().

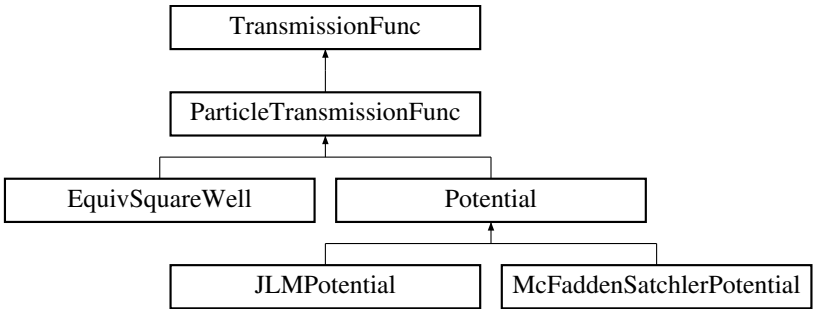
The documentation for this class was generated from the following files:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/ParticleHoleLevelDensity.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/ParticleHoleLevelDensity.cpp

7.37 ParticleTransmissionFunc Class Reference

`#include <ParticleTransmissionFunc.h>`

Inheritance diagram for ParticleTransmissionFunc:



Public Member Functions

- ParticleTransmissionFunc (int z1, int m1, int z2, int m2, double jInitial, int piInitial, double jFinal, int piFinal, double spin, int parity, double maxL, double totalWidthForCorrection, double uncorrTotalWidthForCorrection, double uncorrTotalWidthSqrdForCorrection, TransmissionFunc *previous)
- virtual ~ParticleTransmissionFunc ()
- bool IsValid ()
- double operator() (double)
- double operator() (double, int)
- void CalcSLDependentFunctions (double, std::map< SLPair, double > &)

Static Public Member Functions

- static ParticleTransmissionFunc * CreateParticleTransmissionFunc (int, int, int, int, double, int, double, int, double, int, double, double, double, double, TransmissionFunc *)
- static void SetAlphaFormalism (int formalism)
- static void SetNeutronFormalism (int formalism)
- static void SetProtonFormalism (int formalism)
- static void SetPorterThomas (bool)

Protected Member Functions

- virtual double CalcTransmission (double, int, double)=0

Protected Attributes

- int z1_
- int m1_
- int pType_
- int parity_
- double redmass_
- double spin_

7.37.1 Constructor & Destructor Documentation

7.37.1.1 ParticleTransmissionFunc::ParticleTransmissionFunc (int *z1*, int *m1*, int *z2*, int *m2*, double *jInitial*, int *piInitial*, double *jFinal*, int *piFinal*, double *spin*, int *parity*, double *maxL*, double *totalWidthForCorrection*, double *uncorrTotalWidthForCorrection*, double *uncorrTotalWidthSqrdForCorrection*, TransmissionFunc * *previous*) [inline]

References NuclearMass::FindMass(), m1_, TransmissionFunc::m2_, pType_, redmass_, uconv, z1_, and TransmissionFunc::z2_.

7.37.1.2 virtual ParticleTransmissionFunc::~ParticleTransmissionFunc ( ) [inline], [virtual]

7.37.2 Member Function Documentation

7.37.2.1 void ParticleTransmissionFunc::CalcSLDependentFunctions (double *energy*, std::map< SLPair, double > & *functions*)

References CalcTransmission(), TransmissionFunc::jFinal_, TransmissionFunc::jInitial_, TransmissionFunc::maxL_, parity_, TransmissionFunc::piFinal_, TransmissionFunc::piInitial_, and spin_.

Referenced by operator()().

7.37.2.2 virtual double ParticleTransmissionFunc::CalcTransmission (double , int , double) [protected], [pure virtual]

Implemented in EquivSquareWell, and Potential.

Referenced by CalcSLDependentFunctions().

7.37.2.3 ParticleTransmissionFunc * ParticleTransmissionFunc::CreateParticleTransmissionFunc (int *z1,* int *m1,* int *z2,* int *m2,* double *jInitial,* int *piInitial,* double *jFinal,* int *piFinal,* double *spin,* int *parity,* double *maxL,* double *totalWidthForCorrection,* double *uncorrTotalWidthForCorrection,* double *uncorrTotalWidthSqrdForCorrection,* TransmissionFunc * *previous*) [static]

Referenced by TransitionRateFunc::TransitionRateFunc().

7.37.2.4 bool ParticleTransmissionFunc::IsValid ( ) [inline], [virtual]

Implements TransmissionFunc.

References pType_, and redmass_.

7.37.2.5 double ParticleTransmissionFunc::operator() (double *energy*) [virtual]

Implements TransmissionFunc.

References CalcSLDependentFunctions(), TransmissionFunc::previous_, randomSeed, TransmissionFunc::totalWidthForCorrection_, TransmissionFunc::uncorrTotalWidthForCorrection_, and TransmissionFunc::uncorrTotalWidthSqrdForCorrection_.

7.37.2.6 double ParticleTransmissionFunc::operator() (double *energy,* int *which*)

References CalcSLDependentFunctions(), TransmissionFunc::previous_, TransmissionFunc::totalWidthForCorrection_, TransmissionFunc::uncorrTotalWidthForCorrection_, and TransmissionFunc::uncorrTotalWidthSqrdForCorrection_.

7.37.2.7 static void ParticleTransmissionFunc::SetAlphaFormalism (int *formalism*) [inline], [static]

Referenced by Initialize(), and parseCommandLineForOptions().

7.37.2.8 static void ParticleTransmissionFunc::SetNeutronFormalism (int *formalism*) [inline], [static]

Referenced by Initialize(), and parseCommandLineForOptions().

7.37.2.9 void ParticleTransmissionFunc::SetPorterThomas (bool *yn*) [static]

Referenced by Initialize(), and parseCommandLineForOptions().

7.37.2.10 static void ParticleTransmissionFunc::SetProtonFormalism (int *formalism*) [inline], [static]

Referenced by Initialize(), and parseCommandLineForOptions().

7.37.3 Member Data Documentation

7.37.3.1 int ParticleTransmissionFunc::m1_ [protected]

Referenced by Potential::CalcTransmission(), and ParticleTransmissionFunc().

7.37.3.2 int ParticleTransmissionFunc::parity_ [protected]

Referenced by CalcSLDependentFunctions().

7.37.3.3 int ParticleTransmissionFunc::pType_ [protected]

Referenced by IsValid(), and ParticleTransmissionFunc().

7.37.3.4 double ParticleTransmissionFunc::redmass_ [protected]

Referenced by EquivSquareWell::CalcTransmission(), EquivSquareWell::EquivSquareWell(), Potential::GetRedMass(), IsValid(), and ParticleTransmissionFunc().

7.37.3.5 double ParticleTransmissionFunc::spin_ [protected]

Referenced by CalcSLDependentFunctions().

7.37.3.6 int ParticleTransmissionFunc::z1_ [protected]

Referenced by EquivSquareWell::EquivSquareWell(), Potential::GetZ1Z2(), and ParticleTransmissionFunc().

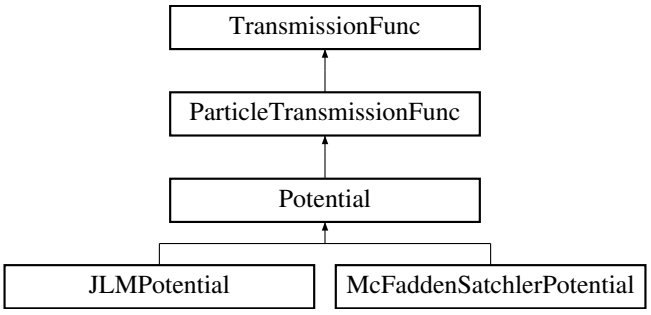
The documentation for this class was generated from the following files:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/ParticleTransmissionFunc.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/ParticleTransmissionFunc.cpp
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Setup.cpp

7.38 Potential Class Reference

`#include <Potential.h>`

Inheritance diagram for Potential:



Public Member Functions

- Potential (int, int, int, int, double, int, double, int, double, int, double, double, double, double, TransmissionFunc ∗)
- ∼Potential ()
- int GetZ1Z2 () const
- double GetBoundaryRadius () const
- double GetRedMass () const
- double GetRMax () const
- virtual std::complex< double > Calculate (double, int, double, double, double) const =0
- std::complex< double > CalcBeta (double, int, double, double, double) const
- double CalcTransmission (double, int, double)
- void NormalizeInternally (std::vector< std::complex< double > > &, double) const
- void NormalizeOverAllSpace (std::vector< std::complex< double > > &, double) const
- std::vector< std::complex < double > > Solve (double, int, double, double, double) const

Protected Attributes

- CoulFunc ∗ coulFunc_
- double boundaryRadius_
- double coulombRadius_

Additional Inherited Members

7.38.1 Constructor & Destructor Documentation

7.38.1.1 Potential::Potential (int *z1,* int *m1,* int *z2,* int *m2,* double *jInitial,* int *piInitial,* double *jFinal,* int *piFinal,* double *spin,* int *parity,* double *maxL,* double *totalWidthForCorrection,* double *uncorrTotalWidthForCorrection,* double *uncorrTotalWidthSqrdForCorrection,* TransmissionFunc ∗ *previous*)

References coulFunc_, and GetRedMass().

7.38.1.2 Potential::∼Potential ( )

References coulFunc_.

7.38.2 Member Function Documentation

7.38.2.1 std::complex< double > Potential::CalcBeta (double *energy,* int *l,* double *s,* double *j,* double *rStep*) const

References GetBoundaryRadius(), NormalizeInternally(), and Solve().

Referenced by CalcTransmission().

7.38.2.2 double Potential::CalcTransmission (double *s,* int *l,* double *energy*) [virtual]

Implements ParticleTransmissionFunc.

References CalcBeta(), coulFunc_, CoulWaves::dF, CoulWaves::dG, CoulWaves::F, CoulWaves::G, GetBoundaryRadius(), GetRedMass(), hbarc, TransmissionFunc::jInitial_, ParticleTransmissionFunc::m1_, and uconv.

7.38.2.3 virtual std::complex<double> Potential::Calculate (double , int , double , double , double) const [pure virtual]

Implemented in JLMPotential, and McFaddenSatchlerPotential.

Referenced by Solve().

7.38.2.4 double Potential::GetBoundaryRadius ( ) const [inline]

References boundaryRadius_.

Referenced by CalcBeta(), CalcTransmission(), NormalizeInternally(), and Solve().

7.38.2.5 double Potential::GetRedMass ( ) const [inline]

References ParticleTransmissionFunc::redmass_.

Referenced by CalcTransmission(), Potential(), and Solve().

7.38.2.6 double Potential::GetRMax ( ) const [inline]

References boundaryRadius_.

7.38.2.7 int Potential::GetZ1Z2 ( ) const [inline]

References ParticleTransmissionFunc::z1_, and TransmissionFunc::z2_.

Referenced by JLMPotential::Calculate(), and McFaddenSatchlerPotential::Calculate().

7.38.2.8 void Potential::NormalizeInternally (std::vector< std::complex< double > > & *waveFunction,* double *rStep*) const

References GetBoundaryRadius().

Referenced by CalcBeta().

7.38.2.9 void Potential::NormalizeOverAllSpace (std::vector< std::complex< double > > & *waveFunction,* double *rStep*) const

7.38.2.10 std::vector< std::complex< double > > Potential::Solve (double *energy,* int *L,* double *S,* double *J,* double *rStep*) const

References Calculate(), GetBoundaryRadius(), GetRedMass(), hbarc, and uconv.

Referenced by CalcBeta().

7.38.3 Member Data Documentation

7.38.3.1 double Potential::boundaryRadius_ [protected]

Referenced by GetBoundaryRadius(), GetRMax(), JLMPotential::JLMPotential(), and McFaddenSatchlerPotential::-McFaddenSatchlerPotential().

7.38.3.2 CoulFunc* Potential::coulFunc_ [protected]

Referenced by CalcTransmission(), Potential(), and ~Potential().

7.38.3.3 double Potential::coulombRadius_ [protected]

Referenced by JLMPotential::Calculate(), McFaddenSatchlerPotential::Calculate(), JLMPotential::JLMPotential(), and McFaddenSatchlerPotential::McFaddenSatchlerPotential().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/Potential.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Potential.cpp

7.39 PreEqCDFEntry Class Reference

`#include <PreEqDecayer.h>`

Public Member Functions

- PreEqCDFEntry (int pairIndex, double energy, double value)

Public Attributes

- int pairIndex_
- double energy_
- double value_

7.39.1 Constructor & Destructor Documentation

7.39.1.1 PreEqCDFEntry::PreEqCDFEntry (int *pairIndex,* double *energy,* double *value*) [inline]

7.39.2 Member Data Documentation

7.39.2.1 double PreEqCDFEntry::energy_

7.39.2.2 int PreEqCDFEntry::pairIndex_

7.39.2.3 double PreEqCDFEntry::value_

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/PreEqDecayer.h

7.40 PreEqDecayer Class Reference

`#include <PreEqDecayer.h>`

Public Member Functions

- PreEqDecayer (int initialNeutronNumber, int initialNeutronHoleNumber, int initialProtonNumber, int initialProtonHoleNumber, int Z, int A, double jInitial, int piInitial, double energy)
- ~PreEqDecayer ()
- bool Decay (int &, int &, double &, int &, int &, int &, int &, int &, double &, double &)
- void PrintCDF ()

Static Public Member Functions

- static void SetCrossSection (bool isCrossSection)
- static void SetMaxL (double maxL)
- static double GetMaxL ()

7.40.1 Constructor & Destructor Documentation

7.40.1.1 PreEqDecayer::PreEqDecayer (int *initialNeutronNumber,* int *initialNeutronHoleNumber,* int *initialProtonNumber,* int *initialProtonHoleNumber,* int *Z,* int *A,* double *jInitial,* int *piInitial,* double *energy*)

References PreEqTransitionRateFunc::Integral(), and NuclearMass::QValue().

7.40.1.2 PreEqDecayer::~PreEqDecayer ( )

7.40.2 Member Function Documentation

7.40.2.1 bool PreEqDecayer::Decay (int & *Z,* int & *A,* double & *jFinal,* int & *piFinal,* int & *neutronNumber,* int & *neutronHoleNumber,* int & *protonNumber,* int & *protonHoleNumber,* double & *excitationEnergy,* double & *decayEnergy*)

References randomSeed.

Referenced by DecayController::Decay().

7.40.2.2 static double PreEqDecayer::GetMaxL ( ) [inline],[static]

7.40.2.3 void PreEqDecayer::PrintCDF ( )

7.40.2.4 static void PreEqDecayer::SetCrossSection (bool *isCrossSection*) [inline],[static]

Referenced by Initialize().

7.40.2.5 static void PreEqDecayer::SetMaxL (double *maxL*) [inline],[static]

Referenced by Initialize().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/PreEqDecayer.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/PreEqDecayer.cpp
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Setup.cpp

7.41 PreEqSpinRatePair Class Reference

`#include <PreEqDecayer.h>`

Public Member Functions

- PreEqSpinRatePair (int neutronNumber, int neutronHoleNumber, int protonNumber, int protonHoleNumber, int Z, int A, double spin, int parity, double qValue, PreEqTransitionRateFunc *rateFunc, double integral)

Public Attributes

- PreEqTransitionRateFunc * rateFunc_
- int neutronNumber_
- int neutronHoleNumber_
- int protonNumber_
- int protonHoleNumber_
- int Z_
- int A_
- int parity_
- double spin_
- double qValue_
- double integral_

7.41.1 Constructor & Destructor Documentation

7.41.1.1 PreEqSpinRatePair::PreEqSpinRatePair (int *neutronNumber,* int *neutronHoleNumber,* int *protonNumber,* int *protonHoleNumber,* int *Z,* int *A,* double *spin,* int *parity,* double *qValue,* PreEqTransitionRateFunc * *rateFunc,* double *integral*) [inline]

7.41.2 Member Data Documentation

7.41.2.1 int PreEqSpinRatePair::A_

7.41.2.2 double PreEqSpinRatePair::integral_

7.41.2.3 int PreEqSpinRatePair::neutronHoleNumber_

7.41.2.4 int PreEqSpinRatePair::neutronNumber_

7.41.2.5 int PreEqSpinRatePair::parity_

7.41.2.6 int PreEqSpinRatePair::protonHoleNumber_

7.41.2.7 int PreEqSpinRatePair::protonNumber_

7.41.2.8 double PreEqSpinRatePair::qValue_

7.41.2.9 PreEqTransitionRateFunc* PreEqSpinRatePair::rateFunc_

7.41.2.10 double PreEqSpinRatePair::spin_

7.41.2.11 int PreEqSpinRatePair::Z_

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/PreEqDecayer.h

7.42 PreEqTransitionRateFunc Class Reference

`#include <PreEqTransitionRateFunc.h>`

Public Member Functions

- PreEqTransitionRateFunc (int, int, int, int, int, int, int, int, int, int, int, int, double, int, double, int, double, int, double, double, double)
- ~PreEqTransitionRateFunc ()
- double Integral () const
- std::vector< XYPair > const CumulativeSum ()

7.42.1 Constructor & Destructor Documentation

7.42.1.1 PreEqTransitionRateFunc::PreEqTransitionRateFunc (int *z1*, int *m1*, int *z2*, int *m2*, int *initialNeutronNumber*, int *initialNeutronHoleNumber*, int *initialProtonNumber*, int *initialProtonHoleNumber*, int *finalNeutronNumber*, int *finalNeutronHoleNumber*, int *finalProtonNumber*, int *finalProtonHoleNumber*, double *jInitial*, int *piInitial*, double *jFinal*, int *piFinal*, double *spin*, int *parity*, double *maxL*, double *compoundE*, double *qValue*)

7.42.1.2 PreEqTransitionRateFunc::~PreEqTransitionRateFunc ( ) [inline]

7.42.2 Member Function Documentation

7.42.2.1 std::vector<XYPair> const PreEqTransitionRateFunc::CumulativeSum ( ) [inline]

7.42.2.2 double PreEqTransitionRateFunc::Integral ( ) const [inline]

Referenced by PreEqDecayer::PreEqDecayer().

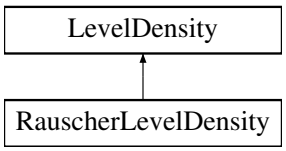
The documentation for this class was generated from the following files:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/PreEqTransitionRateFunc.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/PreEqTransitionRateFunc.cpp

7.43 RauscherLevelDensity Class Reference

`#include <RauscherLevelDensity.h>`

Inheritance diagram for RauscherLevelDensity:



Public Member Functions

- RauscherLevelDensity (int Z, int A, double J)
- ~RauscherLevelDensity ()
- void CalcBackShift ()
- double CalcDensityParam (double)
- double CalcNuclearTemp (double)

Additional Inherited Members

7.43.1 Constructor & Destructor Documentation

7.43.1.1 RauscherLevelDensity::RauscherLevelDensity (int *Z*, int *A*, double *J*) [inline]

References CalcBackShift(), LevelDensity::CalcConstantTempTerms(), and NuclearMass::MicroEnergyCorr().

7.43.1.2 RauscherLevelDensity::~RauscherLevelDensity ( ) [inline]

7.43.2 Member Function Documentation

7.43.2.1 void RauscherLevelDensity::CalcBackShift ( ) [virtual]

Implements LevelDensity.

References LevelDensity::A_, LevelDensity::backshift_, NuclearMass::NeutronPairingGap(), NuclearMass::ProtonPairingGap(), and LevelDensity::Z_.

Referenced by RauscherLevelDensity().

7.43.2.2 double RauscherLevelDensity::CalcDensityParam (double *u*) [virtual]

Implements LevelDensity.

Referenced by CalcNuclearTemp().

7.43.2.3 double RauscherLevelDensity::CalcNuclearTemp (double *u*) [virtual]

Implements LevelDensity.

References CalcDensityParam().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/RauscherLevelDensity.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/RauscherLevelDensity.cpp

7.44 SLPair Class Reference

`#include <ParticleTransmissionFunc.h>`

Public Member Functions

- SLPair (double s, int l)
- bool operator< (const SLPair &right) const

Public Attributes

- double s_
- int l_

7.44.1 Constructor & Destructor Documentation

7.44.1.1 SLPair::SLPair (double *s,* int *l*) [inline]

7.44.2 Member Function Documentation

7.44.2.1 bool SLPair::operator< (const SLPair & *right*) const [inline]

References l_, and s_.

7.44.3 Member Data Documentation

7.44.3.1 int SLPair::l_

Referenced by operator<().

7.44.3.2 double SLPair::s_

Referenced by operator<().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/ParticleTransmissionFunc.h

7.45 SpinRatePair Class Reference

`#include <Decayer.h>`

Public Member Functions

- SpinRatePair (int Z, int A, double spin, int parity, double qValue, TransitionRateFunc *rateFunc, double integral)

Public Attributes

- TransitionRateFunc * rateFunc_
- int Z_
- int A_
- int parity_
- double spin_
- double qValue_
- double integral_

7.45.1 Constructor & Destructor Documentation

7.45.1.1 SpinRatePair::SpinRatePair (int *Z,* int *A,* double *spin,* int *parity,* double *qValue,* TransitionRateFunc * *rateFunc,* double *integral*) [inline]

7.45.2 Member Data Documentation

7.45.2.1 int SpinRatePair::A_

7.45.2.2 double SpinRatePair::integral_

7.45.2.3 int SpinRatePair::parity_

7.45.2.4 double SpinRatePair::qValue_

7.45.2.5 TransitionRateFunc* SpinRatePair::rateFunc_

7.45.2.6 double SpinRatePair::spin_

7.45.2.7 int SpinRatePair::Z_

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/Decayer.h

7.46 TransitionRateFunc Class Reference

`#include <TransitionRateFunc.h>`

Public Member Functions

- TransitionRateFunc (int, int, int, int, double, int, double, int, double, int, double, double, double, double, double, double, TransitionRateFunc *, bool)
- ~TransitionRateFunc ()
- std::vector< XYPair > const Function ()
- std::vector< XYPair > const CumulativeSum ()
- double Integral () const
- double CalcLevelDensity (double energy)
- double CalcTransmissionFunc (double energy)
- double CalcTotalLevelDensity (double energy)
- double ExclusiveBranching () const
- double GroundStateTransmission () const

Static Public Member Functions

- static void SetGammaCutoffEnergy (double energy)
- static double GetGammaCutoffEnergy ()

7.46.1 Constructor & Destructor Documentation

7.46.1.1 TransitionRateFunc::TransitionRateFunc (int *z1*, int *m1*, int *z2*, int *m2*, double *jInitial*, int *piInitial*, double *jFinal*, int *piFinal*, double *spin*, int *parity*, double *maxL*, double *compoundE*, double *qValue*, double *totalWidthForCorrection*, double *uncorrTotalWidthForCorrection*, double *uncorrTotalWidthSqrdForCorrection*, TransitionRateFunc * *previous*, bool *isCrossSection*)

References CalcLevelDensity(), CalcTransmissionFunc(), GammaTransmissionFunc::CreateGammaTransmissionFunc(), ParticleTransmissionFunc::CreateParticleTransmissionFunc(), NuclearLevels::FindLevels(), NuclearMass::HighestBoundEnergy(), and TransmissionFunc::IsValid().

7.46.1.2 TransitionRateFunc::~TransitionRateFunc ( ) [inline]

7.46.2 Member Function Documentation

7.46.2.1 double TransitionRateFunc::CalcLevelDensity (double *energy*) [inline]

Referenced by TransitionRateFunc().

7.46.2.2 double TransitionRateFunc::CalcTotalLevelDensity (double *energy*) [inline]

References LevelDensity::TotalLevelDensity().

7.46.2.3 double TransitionRateFunc::CalcTransmissionFunc (double *energy*) [inline]

Referenced by TransitionRateFunc().

7.46.2.4 std::vector<XYPair> const TransitionRateFunc::CumulativeSum ( ) [inline]

7.46.2.5 double TransitionRateFunc::ExclusiveBranching ( ) const [inline]

7.46.2.6 std::vector<XYPair> const TransitionRateFunc::Function ( ) [inline]

7.46.2.7 static double TransitionRateFunc::GetGammaCutoffEnergy ( ) [inline], [static]

Referenced by parseCommandLineForOptions().

7.46.2.8 double TransitionRateFunc::GroundStateTransmission ( ) const [inline]

Referenced by Decayer::Decayer().

7.46.2.9 double TransitionRateFunc::Integral ( ) const [inline]

Referenced by Decayer::Decayer().

7.46.2.10 static void TransitionRateFunc::SetGammaCutoffEnergy (double *energy*) [inline], [static]

Referenced by CrossSection::Calculate(), Initialize(), and parseCommandLineForOptions().

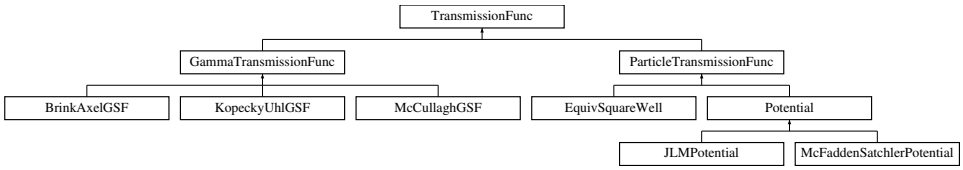
The documentation for this class was generated from the following files:

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/TransitionRateFunc.h
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Setup.cpp
- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/TransitionRateFunc.cpp

7.47 TransmissionFunc Class Reference

`#include <TransmissionFunc.h>`

Inheritance diagram for TransmissionFunc:



Public Member Functions

- TransmissionFunc (int z2, int m2, double jInitial, int piInitial, double jFinal, int piFinal, double maxL, double totalWidthForCorrection, double uncorrTotalWidthForCorrection, double uncorrTotalWidthSqrdForCorrection, TransmissionFunc ∗previous)
- virtual ∼TransmissionFunc ()
- virtual double operator() (double)=0
- virtual bool IsValid ()=0

Protected Attributes

- int z2_
- int m2_
- int piInitial_
- int piFinal_
- double jInitial_
- double jFinal_
- double maxL_
- double totalWidthForCorrection_
- double uncorrTotalWidthForCorrection_
- double uncorrTotalWidthSqrdForCorrection_
- TransmissionFunc ∗ previous_

7.47.1 Constructor & Destructor Documentation

7.47.1.1 TransmissionFunc::TransmissionFunc (int *z2*, int *m2*, double *jInitial*, int *piInitial*, double *jFinal*, int *piFinal*, double *maxL*, double *totalWidthForCorrection*, double *uncorrTotalWidthForCorrection*, double *uncorrTotalWidthSqrdForCorrection*, TransmissionFunc ∗ *previous*) [inline]

7.47.1.2 virtual TransmissionFunc::∼TransmissionFunc ( ) [inline], [virtual]

7.47.2 Member Function Documentation

7.47.2.1 virtual bool TransmissionFunc::IsValid ( ) [pure virtual]

Implemented in GammaTransmissionFunc, and ParticleTransmissionFunc.

Referenced by TransitionRateFunc::TransitionRateFunc().

7.47.2.2 virtual double TransmissionFunc::operator() (double) [pure virtual]

Implemented in GammaTransmissionFunc, and ParticleTransmissionFunc.

7.47.3 Member Data Documentation

7.47.3.1 double TransmissionFunc::jFinal_ [protected]

Referenced by ParticleTransmissionFunc::CalcSLDependentFunctions().

7.47.3.2 double TransmissionFunc::jInitial_ [protected]

Referenced by ParticleTransmissionFunc::CalcSLDependentFunctions(), and Potential::CalcTransmission().

7.47.3.3 int TransmissionFunc::m2_ [protected]

Referenced by GammaTransmissionFunc::GammaTransmissionFunc(), JLMPotential::JLMPotential(), McFaddenSatchlerPotential::McFaddenSatchlerPotential(), and ParticleTransmissionFunc::ParticleTransmissionFunc().

7.47.3.4 double TransmissionFunc::maxL_ [protected]

Referenced by ParticleTransmissionFunc::CalcSLDependentFunctions(), McCullaghGSF::CalcStrengthFunction(), BrinkAxelGSF::CalcStrengthFunction(), and GammaTransmissionFunc::operator()().

7.47.3.5 int TransmissionFunc::piFinal_ [protected]

Referenced by ParticleTransmissionFunc::CalcSLDependentFunctions().

7.47.3.6 int TransmissionFunc::piInitial_ [protected]

Referenced by ParticleTransmissionFunc::CalcSLDependentFunctions().

7.47.3.7 TransmissionFunc* TransmissionFunc::previous_ [protected]

Referenced by ParticleTransmissionFunc::operator()(), and GammaTransmissionFunc::operator()().

7.47.3.8 double TransmissionFunc::totalWidthForCorrection_ [protected]

Referenced by ParticleTransmissionFunc::operator()(), and GammaTransmissionFunc::operator()().

7.47.3.9 double TransmissionFunc::uncorrTotalWidthForCorrection_ [protected]

Referenced by ParticleTransmissionFunc::operator()(), and GammaTransmissionFunc::operator()().

7.47.3.10 double TransmissionFunc::uncorrTotalWidthSqrdForCorrection_ [protected]

Referenced by ParticleTransmissionFunc::operator()(), and GammaTransmissionFunc::operator()().

7.47.3.11 int TransmissionFunc::z2_ [protected]

Referenced by EquivSquareWell::EquivSquareWell(), GammaTransmissionFunc::GammaTransmissionFunc(), Potential::GetZ1Z2(), JLMPotential::JLMPotential(), and ParticleTransmissionFunc::ParticleTransmissionFunc().

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/TransmissionFunc.h

7.48 XYPair Class Reference

`#include <TransitionRateFunc.h>`

Public Member Functions

- XYPair (double X, double Y)

Public Attributes

- double X_
- double Y_

7.48.1 Constructor & Destructor Documentation

7.48.1.1 XYPair::XYPair (double *X*, double *Y*) [inline]

7.48.2 Member Data Documentation

7.48.2.1 double XYPair::X_

7.48.2.2 double XYPair::Y_

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

- /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/TransitionRateFunc.h

Chapter 8

File Documentation

8.1 CMakeFiles/3.13.2/CompilerIdC/CMakeCCompilerId.c File Reference

Macros

- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define C_DIALECT

Functions

- int main (int argc, char ∗argv[])

Variables

- char const ∗ info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const ∗ info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const ∗ info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char ∗ info_language_dialect_default

8.1.1 Macro Definition Documentation

8.1.1.1 #define ARCHITECTURE_ID

8.1.1.2 #define C_DIALECT

8.1.1.3 #define COMPILER_ID ""

8.1.1.4 #define DEC(*n*)

Value:

```
('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

8.1.1.5 #define HEX(*n*)

Value:

```
('0' + ((n)>>28 & 0xF)), \
  ('0' + ((n)>>24 & 0xF)), \
  ('0' + ((n)>>20 & 0xF)), \
  ('0' + ((n)>>16 & 0xF)), \
  ('0' + ((n)>>12 & 0xF)), \
  ('0' + ((n)>>8  & 0xF)), \
  ('0' + ((n)>>4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

8.1.1.6 #define PLATFORM_ID

8.1.1.7 #define STRINGIFY(*X*) STRINGIFY_HELPER(X)

8.1.1.8 #define STRINGIFY_HELPER(*X*) #X

8.1.2 Function Documentation

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

References info_arch, info_compiler, info_language_dialect_default, and info_platform.

8.1.3 Variable Documentation

8.1.3.1 char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"

Referenced by main().

8.1.3.2 char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"

Referenced by main().

8.1.3.3 const char* info_language_dialect_default

Initial value:

```
=
  "INFO" ":" "dialect_default[" C_DIALECT "]"
```

Referenced by main().

8.1.3.4 char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"

Referenced by main().

8.2 CMakeFiles/3.13.2/CompilerIdCXX/CMakeCXXCompilerId.cpp File Reference

Macros

- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define CXX_STD __cplusplus

Functions

- int main (int argc, char ∗argv[])

Variables

- char const ∗ info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const ∗ info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const ∗ info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char ∗ info_language_dialect_default

8.2.1 Macro Definition Documentation

8.2.1.1 #define ARCHITECTURE_ID

8.2.1.2 #define COMPILER_ID ""

8.2.1.3 #define CXX_STD __cplusplus

8.2.1.4 #define DEC(*n*)

Value:

```
('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

8.2.1.5 #define HEX(*n*)

Value:

```
('0' + ((n)>>28 & 0xF)), \
  ('0' + ((n)>>24 & 0xF)), \
  ('0' + ((n)>>20 & 0xF)), \
  ('0' + ((n)>>16 & 0xF)), \
  ('0' + ((n)>>12 & 0xF)), \
  ('0' + ((n)>>8  & 0xF)), \
  ('0' + ((n)>>4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

8.2.1.6 #define PLATFORM_ID

8.2.1.7 #define STRINGIFY(*X*) STRINGIFY_HELPER(X)

8.2.1.8 #define STRINGIFY_HELPER(*X*) #X

8.2.2 Function Documentation

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

References info_compiler, info_language_dialect_default, and info_platform.

8.2.3 Variable Documentation

8.2.3.1 char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"

8.2.3.2 char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"

8.2.3.3 const char* info_language_dialect_default

Initial value:

```
= "INFO" ":" "dialect_default["




  "98"
"]"
```

8.2.3.4 char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"

8.3 CMakeFiles/feature_tests.c File Reference

Functions

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

Variables

- const char features []

8.3.1 Function Documentation

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

References features.

8.3.2 Variable Documentation

8.3.2.1 const char features[ ]

Referenced by main().

8.4 CMakeFiles/feature_tests.cxx File Reference

Functions

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

Variables

- const char features [ ]

8.4.1 Function Documentation

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

References features.

8.4.2 Variable Documentation

8.4.2.1 const char features[ ]

8.5 CMakeFiles/FindOpenMP/OpenMPCheckVersion.c File Reference

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

Functions

- int main (void)

Variables

- const char ompver_str [ ]

8.5.1 Function Documentation

8.5.1.1 int main (void)

References ompver_str.

8.5.2 Variable Documentation

8.5.2.1 const char ompver_str[]

Initial value:

```
= { 'I', 'N', 'F', 'O', ':', 'O', 'p', 'e', 'n', 'M',
                            'P', '-', 'd', 'a', 't', 'e', '[',
                            ('0' + ((_OPENMP/100000)%10)),
                            ('0' + ((_OPENMP/10000)%10)),
                            ('0' + ((_OPENMP/1000)%10)),
                            ('0' + ((_OPENMP/100)%10)),
                            ('0' + ((_OPENMP/10)%10)),
                            ('0' + ((_OPENMP/1)%10)),
                            ']', '\0' }
```

Referenced by main().

8.6 CMakeFiles/FindOpenMP/OpenMPCheckVersion.cpp File Reference

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

Functions

- int main (void)

Variables

- const char ompver_str []

8.6.1 Function Documentation

8.6.1.1 int main (void)

References ompver_str.

8.6.2 Variable Documentation

8.6.2.1 const char ompver_str[]

Initial value:

```
= { 'I', 'N', 'F', 'O', ':', 'O', 'p', 'e', 'n', 'M',
                            'P', '-', 'd', 'a', 't', 'e', '[',
                            ('0' + ((_OPENMP/100000)%10)),
                            ('0' + ((_OPENMP/10000)%10)),
                            ('0' + ((_OPENMP/1000)%10)),
                            ('0' + ((_OPENMP/100)%10)),
                            ('0' + ((_OPENMP/10)%10)),
                            ('0' + ((_OPENMP/1)%10)),
                            ']', '\0' }
```

8.7 CMakeFiles/FindOpenMP/OpenMPTryFlag.c File Reference

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

Functions

- int main (void)

8.7.1 Function Documentation

8.7.1.1 int main (void)

8.8 CMakeFiles/FindOpenMP/OpenMPTryFlag.cpp File Reference

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

Functions

- int main (void)

8.8.1 Function Documentation

8.8.1.1 int main (void)

8.9 generated/Sapphire_config.h File Reference

```
#include <string>
```

Functions

- std::string sourceDirectory ()

8.9.1 Function Documentation

8.9.1.1 std::string sourceDirectory ( )

Referenced by Initialize().

8.10 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/generated/Sapphire_config.h File Reference

```
#include <string>
```

Functions

- std::string sourceDirectory ()

8.10.1 Function Documentation

8.10.1.1 std::string sourceDirectory ( )

8.11 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/include/complex_functions.H File Reference

```
#include <complex>
#include <iostream>
#include <cstdlib>
```

Macros

- #define SIGN(a) (((a) < 0) ? (-1) : (1))

Functions

- double inf_norm (const std::complex< double > &z)
- bool isfinite (const std::complex< double > &z)
- std::complex< double > operator+ (const std::complex< double > &z, const int n)
- std::complex< double > operator- (const std::complex< double > &z, const int n)
- std::complex< double > operator* (const std::complex< double > &z, const int n)
- std::complex< double > operator/ (const std::complex< double > &z, const int n)
- std::complex< double > operator+ (const int n, const std::complex< double > &z)
- std::complex< double > operator- (const int n, const std::complex< double > &z)
- std::complex< double > operator* (const int n, const std::complex< double > &z)
- std::complex< double > operator/ (const int n, const std::complex< double > &z)
- std::complex< double > operator+ (const std::complex< double > &z, const unsigned int n)
- std::complex< double > operator- (const std::complex< double > &z, const unsigned int n)
- std::complex< double > operator* (const std::complex< double > &z, const unsigned int n)
- std::complex< double > operator/ (const std::complex< double > &z, const unsigned int n)
- std::complex< double > operator+ (const unsigned int n, const std::complex< double > &z)
- std::complex< double > operator- (const unsigned int n, const std::complex< double > &z)
- std::complex< double > operator* (const unsigned int n, const std::complex< double > &z)
- std::complex< double > operator/ (const unsigned int n, const std::complex< double > &z)
- bool operator== (const std::complex< double > &z, const int n)
- bool operator!= (const std::complex< double > &z, const int n)
- bool operator== (const int n, const std::complex< double > &z)
- bool operator!= (const int n, const std::complex< double > &z)
- bool operator== (const std::complex< double > &z, const unsigned int n)
- bool operator!= (const std::complex< double > &z, const unsigned int n)
- bool operator== (const unsigned int n, const std::complex< double > &z)
- bool operator!= (const unsigned int n, const std::complex< double > &z)
- std::complex< double > expm1 (const std::complex< double > &z)
- std::complex< double > log1p (const std::complex< double > &z)
- std::complex< double > log_Gamma (const std::complex< double > &z)
- std::complex< double > sigma_l_calc (const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > log_Cl_eta_calc (const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > log_cut_constant_AS_calc (const int omega, const std::complex< double > &l, const std::complex< double > &eta)

- std::complex< double > log_cut_constant_CFa_calc (const bool is_it_normalized, const int omega, const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > log_cut_constant_CFb_calc (const bool is_it_normalized, const int omega, const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > sin_chi_calc (const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > exp_I_omega_chi_calc (const int omega, const std::complex< double > &l, const std::complex< double > &eta)

Variables

- const double precision = 1E-10
- const double sqrt_precision = 1E-5

8.11.1 Macro Definition Documentation

8.11.1.1 #define SIGN(*a*) (((a) < 0) ? (-1) : (1))

Referenced by exp_I_omega_chi_calc(), and Coulomb_wave_functions::F_dF().

8.11.2 Function Documentation

8.11.2.1 std::complex<double> exp_I_omega_chi_calc (const int *omega,* const std::complex< double > & *l,* const std::complex< double > & *eta*)

References sigma_l_calc(), SIGN, and sin_chi_calc().

8.11.2.2 std::complex<double> expm1 (const std::complex< double > & *z*)

References expm1().

Referenced by expm1(), log_cut_constant_AS_calc(), and log_cut_constant_CFa_calc().

8.11.2.3 double inf_norm (const std::complex< double > & *z*) [inline]

Referenced by ODE_integration::operator()().

8.11.2.4 bool isfinite (const std::complex< double > & *z*) [inline]

Referenced by Coulomb_wave_functions::F_dF(), Coulomb_wave_functions::G_dG(), Coulomb_wave_functions::H_dH(), Coulomb_wave_functions::H_dH_scaled(), and log_Gamma().

8.11.2.5 std::complex<double> log1p (const std::complex< double > & *z*)

References log1p().

Referenced by log1p(), log_cut_constant_AS_calc(), and log_cut_constant_CFa_calc().

8.11.2.6 std::complex<double> log_Cl_eta_calc (const std::complex< double > & *l,* const std::complex< double > & *eta*)

References log_Gamma().

Referenced by log_cut_constant_CFa_calc(), log_cut_constant_CFb_calc(), and sin_chi_calc().

8.11.2.7 std::complex<double> log_cut_constant_AS_calc (const int *omega*, const std::complex< double > & *l*, const std::complex< double > & *eta*)

References expm1(), and log1p().

8.11.2.8 std::complex<double> log_cut_constant_CFa_calc (const bool *is_it_normalized*, const int *omega*, const std::complex< double > & *l*, const std::complex< double > & *eta*)

References expm1(), log1p(), and log_CI_eta_calc().

8.11.2.9 std::complex<double> log_cut_constant_CFb_calc (const bool *is_it_normalized*, const int *omega*, const std::complex< double > & *l*, const std::complex< double > & *eta*)

References log_CI_eta_calc().

8.11.2.10 std::complex<double> log_Gamma (const std::complex< double > & *z*)

References isfinite(), and log_Gamma().

Referenced by log_CI_eta_calc(), log_Gamma(), and sigma_l_calc().

8.11.2.11 bool operator!= (const std::complex< double > & *z*, const int *n*) [inline]

8.11.2.12 bool operator!= (const int *n*, const std::complex< double > & *z*) [inline]

8.11.2.13 bool operator!= (const std::complex< double > & *z*, const unsigned int *n*) [inline]

8.11.2.14 bool operator!= (const unsigned int *n*, const std::complex< double > & *z*) [inline]

8.11.2.15 std::complex<double> operator* (const std::complex< double > & *z*, const int *n*) [inline]

8.11.2.16 std::complex<double> operator* (const int *n*, const std::complex< double > & *z*) [inline]

8.11.2.17 std::complex<double> operator* (const std::complex< double > & *z*, const unsigned int *n*) [inline]

8.11.2.18 std::complex<double> operator* (const unsigned int *n*, const std::complex< double > & *z*) [inline]

8.11.2.19 std::complex<double> operator+ (const std::complex< double > & *z*, const int *n*) [inline]

8.11.2.20 std::complex<double> operator+ (const int *n*, const std::complex< double > & *z*) [inline]

8.11.2.21 std::complex<double> operator+ (const std::complex< double > & *z*, const unsigned int *n*) [inline]

8.11.2.22 std::complex<double> operator+ (const unsigned int *n*, const std::complex< double > & *z*) [inline]

8.11.2.23 std::complex<double> operator- (const std::complex< double > & *z*, const int *n*) [inline]

8.11.2.24 std::complex<double> operator- (const int *n*, const std::complex< double > & *z*) [inline]

8.11.2.25 std::complex<double> operator- (const std::complex< double > & *z*, const unsigned int *n*) [inline]

8.11.2.26 std::complex<double> operator- (const unsigned int *n*, const std::complex< double > & *z*) [inline]

8.11.2.27 std::complex<double> operator/ (const std::complex< double > & *z*, const int *n*) [inline]

8.11.2.28 std::complex<double> operator/ (const int *n*, const std::complex< double > & *z*) [inline]

8.11.2.29 std::complex<double> operator/ (const std::complex< double > & *z*, const unsigned int *n*) [inline]

8.11.2.30 std::complex<double> operator/ (const unsigned int *n*, const std::complex< double > & *z*) [inline]

8.11.2.31 bool operator== (const std::complex< double > & *z*, const int *n*) [inline]

8.11.2.32 bool operator== (const int *n*, const std::complex< double > & *z*) [inline]

8.11.2.33 bool operator== (const std::complex< double > & *z*, const unsigned int *n*) [inline]

8.11.2.34 bool operator== (const unsigned int *n*, const std::complex< double > & *z*) [inline]

8.11.2.35 std::complex<double> sigma_l_calc (const std::complex< double > & *l*, const std::complex< double > & *eta*)

References log_Gamma().

Referenced by exp_I_omega_chi_calc().

8.11.2.36 std::complex<double> sin_chi_calc (const std::complex< double > & *l*, const std::complex< double > & *eta*)

References log_CI_eta_calc().

Referenced by exp_I_omega_chi_calc().

8.11.3 Variable Documentation

8.11.3.1 const double precision = 1E-10

Referenced by ODE_integration::operator()().

8.11.3.2 const double sqrt_precision = 1E-5

Referenced by Coulomb_wave_functions::F_dF(), Coulomb_wave_functions::G_dG(), Coulomb_wave_functions::H_dH(), and Coulomb_wave_functions::H_dH_scaled().

8.12 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/include/cwfcomp.H File Reference

```
#include "ode_int.H"
```

Classes

- class Coulomb_wave_functions

8.13 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/include/ode_int.H File Reference

```
#include "complex_functions.H"
```

Classes

- class ODE_integration

8.14 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/src/complex_functions.cpp File Reference

```
#include "complex_functions.H"
```

Functions

- std::complex< double > expm1 (const std::complex< double > &z)
- std::complex< double > log1p (const std::complex< double > &z)
- std::complex< double > log_Gamma (const std::complex< double > &z)
- std::complex< double > sigma_l_calc (const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > log_Cl_eta_calc (const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > log_cut_constant_AS_calc (const int omega, const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > log_cut_constant_CFa_calc (const bool is_it_normalized, const int omega, const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > log_cut_constant_CFb_calc (const bool is_it_normalized, const int omega, const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > sin_chi_calc (const std::complex< double > &l, const std::complex< double > &eta)
- std::complex< double > exp_I_omega_chi_calc (const int omega, const std::complex< double > &l, const std::complex< double > &eta)

8.14.1 Function Documentation

8.14.1.1 std::complex<double> exp_I_omega_chi_calc (const int *omega*, const std::complex< double > & *l*, const std::complex< double > & *eta*)

References sigma_l_calc(), SIGN, and sin_chi_calc().

8.14.1.2 std::complex<double> expm1 (const std::complex< double > & *z*)

References expm1().

Referenced by expm1(), log_cut_constant_AS_calc(), and log_cut_constant_CFa_calc().

8.14.1.3 std::complex<double> log1p (const std::complex< double > & *z*)

References log1p().

Referenced by log1p(), log_cut_constant_AS_calc(), and log_cut_constant_CFa_calc().

8.14.1.4 std::complex<double> log_Cl_eta_calc (const std::complex< double > & *l*, const std::complex< double > & *eta*)

References log_Gamma().

Referenced by log_cut_constant_CFa_calc(), log_cut_constant_CFb_calc(), and sin_chi_calc().

8.14.1.5 std::complex<double> log_cut_constant_AS_calc (const int *omega*, const std::complex< double > & *l*, const std::complex< double > & *eta*)

References expm1(), and log1p().

8.14.1.6 std::complex<double> log_cut_constant_CFa_calc (const bool *is_it_normalized*, const int *omega*, const std::complex< double > & *l*, const std::complex< double > & *eta*)

References expm1(), log1p(), and log_Cl_eta_calc().

8.14.1.7 std::complex<double> log_cut_constant_CFb_calc (const bool *is_it_normalized*, const int *omega*, const std::complex< double > & *l*, const std::complex< double > & *eta*)

References log_Cl_eta_calc().

8.14.1.8 std::complex<double> log_Gamma (const std::complex< double > & *z*)

References isfinite(), and log_Gamma().

Referenced by log_Cl_eta_calc(), log_Gamma(), and sigma_l_calc().

8.14.1.9 std::complex<double> sigma_l_calc (const std::complex< double > & *l*, const std::complex< double > & *eta*)

References log_Gamma().

Referenced by exp_I_omega_chi_calc().

8.14.1.10 std::complex<double> sin_chi_calc (const std::complex< double > & *l*, const std::complex< double > & *eta*)

References log_Cl_eta_calc().

Referenced by exp_I_omega_chi_calc().

8.15 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/src/cwfcomp.cpp File Reference

```
#include "cwfcomp.H"
```

8.16 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/coul/src/ode_int.cpp File Reference

```
#include "ode_int.H"
```

8.17 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/BrinkAxelGSF.h File Reference

Header file for the Brink-Axel-GSF class.

```
#include "GammaTransmissionFunc.h"
```

Classes

- class BrinkAxelGSF

8.17.1 Detailed Description

Header file for the Brink-Axel-GSF class.

8.18 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/Constants.h File Reference

```
#include <complex>
#include <vector>
#include <cstdlib>
```

Typedefs

- typedef std::complex< double > complex
- typedef std::vector< double > vector_r
- typedef std::vector < std::complex< double > > vector_c
- typedef std::vector < std::vector< double > > matrix_r
- typedef std::vector < std::vector< std::complex < double > > > matrix_c
- typedef std::vector < std::vector< std::vector < double > > > vector_matrix_r
- typedef std::vector < std::vector< std::vector < std::complex< double > > > > vector_matrix_c

Variables

- const double pi =3.141592650
- const double hbarc =197.32696310
- const double uconv =931.4940880
- const double fstruc =1.00/137.0359996790
- const double boltzConst =8.6171e-2
- const double lightSpeedInCmPerS =29979245800.
- const double avagadroNum =6.02214179e23
- const double eMass = 548.579894

8.18.1 Typedef Documentation

8.18.1.1 typedef std::complex<double> complex

8.18.1.2 typedef std::vector<std::vector<std::complex<double> > > matrix_c

8.18.1.3 typedef std::vector<std::vector<double> > matrix_r

8.18.1.4 typedef std::vector<std::complex<double> > vector_c

8.18.1.5 typedef std::vector<std::vector<std::vector<std::complex<double> > > > vector_matrix_c

8.18.1.6 typedef std::vector<std::vector<std::vector<double> > > vector_matrix_r

8.18.1.7 typedef std::vector<double> vector_r

8.18.2 Variable Documentation

8.18.2.1 const double avagadroNum =6.02214179e23

Referenced by CrossSection::CalculateReactionRates().

8.18.2.2 const double boltzConst =8.6171e-2

Referenced by CrossSection::CalculateReactionRates(), gsl_partfunc_integrand(), and gsl_reactionrate_integrand().

8.18.2.3 const double eMass = 548.579894

8.18.2.4 const double fstruc =1.00/137.0359996790

Referenced by JLMPotential::Calculate(), McFaddenSatchlerPotential::Calculate(), GammaTransmissionFunc::GammaTransmissionFunc(), and CoulFunc::operator()().

8.18.2.5 const double hbarc =197.32696310

Referenced by LevelDensity::CalcConstantTempTerms(), Potential::CalcTransmission(), EquivSquareWell::CalcTransmission(), McFaddenSatchlerPotential::Calculate(), JLMPotential::Calculate(), CrossSection::CrossSection(), GammaTransmissionFunc::GammaTransmissionFunc(), LevelDensity::operator()(), CoulFunc::operator()(), CoulFunc::Penetrability(), CoulFunc::PEShift(), Potential::Solve(), and LevelDensity::TotalLevelDensity().

8.18.2.6 const double lightSpeedInCmPerS =29979245800.

Referenced by CrossSection::CalculateReactionRates().

8.18.2.7 const double pi =3.141592650

Referenced by CrossSection::CalcAverageDWaveResWidth(), CrossSection::CalcAveragePWaveResWidth(), CrossSection::CalcAverageSWaveResWidth(), KopeckyUhlGSF::CalcStrengthFunction(), EquivSquareWell::CalcTransmission(), CrossSection::CalculateReactionRates(), CrossSection::CrossSection(), GammaTransmissionFunc::GammaTransmissionFunc(), JLMPotential::JLMPotential(), and GammaTransmissionFunc::operator()().

8.18.2.8 const double uconv =931.4940880

Referenced by LevelDensity::CalcConstantTempTerms(), Potential::CalcTransmission(), EquivSquareWell::CalcTransmission(), NuclearMass::CalculateLDMMass(), CrossSection::CalculateReactionRates(), CrossSection::CrossSection(), NuclearMass::InitializeMasses(), LevelDensity::operator()(), CoulFunc::operator()(), ParticleTransmissionFunc::ParticleTransmissionFunc(), CoulFunc::Penetrability(), CoulFunc::PEShift(), Potential::Solve(), and LevelDensity::TotalLevelDensity().

8.19 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/CoulFunc.h File Reference

Classes

- struct CoulWaves
- class CoulFunc

8.20 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/CrossSection.h File Reference

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

Classes

- struct int_double_pair_compare
- class CrossSectionValues
- class CrossSection

Typedefs

- typedef std::vector< std::pair < Decayer *, std::vector < SpinRatePair * > > > DecayerVector
- typedef std::pair< int, double > int_double_pair

8.20.1 Typedef Documentation

8.20.1.1 typedef std::vector<std::pair<Decayer*,std::vector<SpinRatePair*> > > DecayerVector

8.20.1.2 typedef std::pair<int,double> int_double_pair

8.21 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/DecayController.h File Reference

```
#include <vector>
#include "Decayer.h"
#include "DecayProduct.h"
```

Classes

- class DecayController

8.22 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/Decayer.h File Reference

```
#include <vector>
#include <cstdlib>
```

Classes

- class SpinRatePair
- class CDFEntry
- class Decayer

8.23 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/DecayProduct.h File Reference

Classes

- class DecayData
- class DecayProduct

8.24 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/DecayResults.h File Reference

```
#include <TTree.h>
#include <TFile.h>
#include <vector>
```

Classes

- class DecayResults

8.25 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/elements.h File Reference

8.26 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/EquivSquareWell.h File Reference

```
#include "CoulFunc.h"
#include "ParticleTransmissionFunc.h"
```

Classes

- class EquivSquareWell

8.27 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/GammaTransmissionFunc.h File Reference

```
#include "TransmissionFunc.h"
#include "Constants.h"
#include "NuclearMass.h"
#include <math.h>
```

Classes

- class GDRParameters
- class GammaTransmissionFunc

Typedefs

- typedef
  std::tr1::unordered_map
  < MassKey, GDRParameters > GDRTable

8.27.1 Typedef Documentation

8.27.1.1 typedef std::tr1::unordered_map<MassKey, GDRParameters > GDRTable

8.28 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/JLMPotential.h File Reference

```
#include "Potential.h"
```

Classes

- class JLMPotential

8.29 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/KopeckyUhlGSF.h File Reference

```
#include "GammaTransmissionFunc.h"
```

Classes

- class KopeckyUhlGSF

8.30 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/LevelDensity.h File Reference

Classes

- class LevelDensity

8.31 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/McCullaghGSF.h File Reference

```
#include "GammaTransmissionFunc.h"
```

Classes

- class McCullaghGSF

8.32 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/McFaddenSatchlerPotential.h File Reference

```
#include "Potential.h"
```

Classes

- class McFaddenSatchlerPotential

8.33 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearLevels.h File Reference

```
#include "NuclearMass.h"
```

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

Classes

- class GammaTransition
- class Level
- class LevelsContainer
- class NuclearLevels

Typedefs

- typedef
  std::tr1::unordered_map
  < MassKey, LevelsContainer > LevelsTable

8.33.1 Typedef Documentation

8.33.1.1 typedef std::tr1::unordered_map<MassKey, LevelsContainer> LevelsTable

8.34 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/NuclearMass.h File Reference

```
#include <tr1/unordered_map>
#include <string>
```

Classes

- class MassKey
- struct std::tr1::hash< MassKey >
- struct std::equal_to< MassKey >
- class MassEntry
- class NuclearMass

Namespaces

- std
- std::tr1

Typedefs

- typedef
  std::tr1::unordered_map
  < MassKey, MassEntry > MassTable
- typedef
  std::tr1::unordered_map
  < std::string, int > ElementTable

8.34.1 Typedef Documentation

8.34.1.1 typedef std::tr1::unordered_map<std::string, int > ElementTable

8.34.1.2 typedef std::tr1::unordered_map<MassKey, MassEntry> MassTable

8.35 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/ParticleHoleLevelDensity.h File Reference

Classes

- class ParticleHoleLevelDensity

8.36 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/ParticleTransmissionFunc.h File Reference

```
#include "TransmissionFunc.h"
#include "NuclearMass.h"
#include "Constants.h"
#include <map>
```

Classes

- class SLPair
- class ParticleTransmissionFunc

8.37 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/Potential.h File Reference

```
#include <vector>
#include <complex>
#include "ParticleTransmissionFunc.h"
```

Classes

- class Potential

8.38 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/PreEqDecayer.h File Reference

```
#include <vector>
```

Classes

- class PreEqSpinRatePair

- class PreEqCDFEntry
- class PreEqDecayer

8.39 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/PreEqTransitionRateFunc.h File Reference

```
#include "TransitionRateFunc.h"
#include "ParticleHoleLevelDensity.h"
```

Classes

- class PreEqTransitionRateFunc

8.40 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/RauscherLevelDensity.h File Reference

```
#include "LevelDensity.h"
#include "NuclearMass.h"
#include <iostream>
#include <stdlib.h>
#include <math.h>
```

Classes

- class RauscherLevelDensity

8.41 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/SapphireMPITypes.h File Reference

```
#include <boost/serialization/access.hpp>
#include <boost/serialization/vector.hpp>
#include <boost/serialization/utility.hpp>
#include <boost/mpi/datatype.hpp>
#include <vector>
#include "DecayProduct.h"
```

Classes

- class InitialNucleusData

Namespaces

- boost
- boost::serialization

Enumerations

- enum SapphireTags_t { SapphireTagProcess, SapphireTagDone, SapphireTagResults }

Functions

- BOOST_IS_MPI_DATATYPE (InitialNucleusData)
- void boost::serialization::serialize (Archive &ar, DecayData &g, const unsigned int version)
- void boost::serialization::serialize (Archive &ar, DecayProduct &g, const unsigned int version)
- BOOST_IS_MPI_DATATYPE (DecayData)
- BOOST_IS_MPI_DATATYPE (DecayProduct)

8.41.1 Enumeration Type Documentation

8.41.1.1 enum SapphireTags_t

Enumerator

SapphireTagProcess

SapphireTagDone

SapphireTagResults

8.41.2 Function Documentation

8.41.2.1 BOOST_IS_MPI_DATATYPE (InitialNucleusData)

8.41.2.2 BOOST_IS_MPI_DATATYPE (DecayData)

8.41.2.3 BOOST_IS_MPI_DATATYPE (DecayProduct)

8.42 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/TransitionRateFunc.h File Reference

```
#include <vector>
#include "LevelDensity.h"
#include "TransmissionFunc.h"
```

Classes

- class XYPair
- class TransitionRateFunc

8.43 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/include/TransmissionFunc.h File Reference

Classes

- class TransmissionFunc

8.44 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/README.md File Reference

8.45 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/BrinkAxelGSF.cpp File Reference

```
#include "BrinkAxelGSF.h"
```

8.46 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/CoulFunc.cpp File Reference

```
#include "Constants.h"
#include "CoulFunc.h"
#include <iostream>
#include "cwfcomp.H"
#include <gsl/gsl_sf_coulomb.h>
#include <gsl/gsl_deriv.h>
#include <gsl/gsl_errno.h>
```

8.47 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/CrossSection.cpp File Reference

```
#include "NuclearLevels.h"
#include "CrossSection.h"
#include "Decayer.h"
#include "TransitionRateFunc.h"
#include "RauscherLevelDensity.h"
#include "Constants.h"
#include <math.h>
#include <fstream>
#include <iomanip>
#include <time.h>
#include <iostream>
#include <sstream>
#include <gsl/gsl_integration.h>
#include <TGraph.h>
#include <algorithm>
```

Classes

- struct gsl_reactionrate_params
- struct gsl_partfunc_params

Functions

- double gsl_reactionrate_integrand (double x, void *p)
- double gsl_partfunc_integrand (double x, void *p)

8.47.1 Function Documentation

8.47.1.1 double gsl_partfunc_integrand (double *x*, void * *p*)

References boltzConst, gsl_partfunc_params::density, and gsl_partfunc_params::temperature.

8.47.1.2 double gsl_reactionrate_integrand (double *x*, void * *p*)

References boltzConst, gsl_reactionrate_params::graph, gsl_reactionrate_params::temperature, and gsl_reactionrate_params::useSpline.

Referenced by CrossSection::CalculateReactionRates().

8.48 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/DecayController.cpp File Reference

```
#include "DecayController.h"
#include "Constants.h"
#include "NuclearMass.h"
#include "PreEqDecayer.h"
#include <iostream>
#include <iomanip>
#include <TVector3.h>
#include <stdlib.h>
#include <omp.h>
```

Variables

- unsigned int randomSeed [12]

8.48.1 Variable Documentation

8.48.1.1 unsigned int randomSeed[12]

Referenced by Decayer::Decay(), PreEqDecayer::Decay(), main(), ParticleTransmissionFunc::operator()(), and GammaTransmissionFunc::operator()().

8.49 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Decayer.cpp File Reference

```
#include "Decayer.h"
#include "TransitionRateFunc.h"
#include "NuclearMass.h"
#include "NuclearLevels.h"
#include <iostream>
#include <fstream>
#include <stdlib.h>
#include <math.h>
#include <omp.h>
```

Variables

- unsigned int randomSeed [12]

8.49.1 Variable Documentation

8.49.1.1 unsigned int randomSeed[12]

8.50 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/DecayResults.cpp File Reference

```
#include "DecayResults.h"
#include "DecayProduct.h"
#include "NuclearMass.h"
#include <fstream>
#include <iostream>
```

8.51 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/EquivSquareWell.cpp File Reference

```
#include "EquivSquareWell.h"
```

8.52 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/GammaTransmissionFunc.cpp File Reference

```
#include "GammaTransmissionFunc.h"
#include "BrinkAxelGSF.h"
#include "KopeckyUhlGSF.h"
#include "McCullaghGSF.h"
#include "Constants.h"
#include <iostream>
#include <iomanip>
#include <fstream>
#include <sstream>
#include <stdlib.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include <omp.h>
```

Variables

- unsigned int randomSeed [12]

8.52.1 Variable Documentation

8.52.1.1 unsigned int randomSeed[12]

8.53 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/JLMPotential.cpp File Reference

```
#include "JLMPotential.h"
#include <assert.h>
```

8.54 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/KopeckyUhlGSF.cpp File Reference

```
#include "KopeckyUhlGSF.h"
#include "LevelDensity.h"
#include <iostream>
```

8.55 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/LevelDensity.cpp File Reference

```
#include "LevelDensity.h"
#include "Constants.h"
#include <math.h>
```

8.56 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/McCullaghGSF.cpp File Reference

```
#include "McCullaghGSF.h"
```

8.57 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/McFaddenSatchlerPotential.cpp File Reference

```
#include "McFaddenSatchlerPotential.h"
```

8.58 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/NuclearLevels.cpp File Reference

```
#include "NuclearLevels.h"
#include <fstream>
#include <sstream>
#include <iostream>
#include <iomanip>
#include <math.h>
#include <string.h>
```

8.59 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/NuclearMass.cpp File Reference

```
#include "NuclearMass.h"
#include "Constants.h"
#include <sstream>
#include <fstream>
#include <iostream>
#include <stdlib.h>
#include "elements.h"
```

Macros

- #define HAS_EXP_MASS 1
- #define HAS_TH_MASS 2
- #define ELEMENT(Z, EL) {elementTable_[std::string(EL)]=Z;}

8.59.1 Macro Definition Documentation

8.59.1.1 #define ELEMENT(*Z, EL*) {elementTable_[std::string(EL)]=Z;}

8.59.1.2 #define HAS_EXP_MASS 1

Referenced by NuclearMass::FindMass(), and NuclearMass::InitializeMasses().

8.59.1.3 #define HAS_TH_MASS 2

Referenced by NuclearMass::FindMass(), and NuclearMass::InitializeMasses().

8.60 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/ParticleHoleLevelDensity.cpp File Reference

```
#include "ParticleHoleLevelDensity.h"
#include "gsl/gsl_sf_gamma.h"
#include "math.h"
#include <algorithm>
#include <iostream>
#include "NuclearMass.h"
```

8.61 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/ParticleTransmissionFunc.cpp File Reference

```
#include "ParticleTransmissionFunc.h"
#include "EquivSquareWell.h"
#include "McFaddenSatchlerPotential.h"
#include "JLMPotential.h"
#include "Constants.h"
#include <iostream>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include <omp.h>
```

Variables

- unsigned int randomSeed [12]

8.61.1 Variable Documentation

8.61.1.1 unsigned int randomSeed[12]

8.62 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Potential.cpp File Reference

```
#include "Potential.h"
#include "Constants.h"
#include "CoulFunc.h"
#include <iostream>
#include <float.h>
```

8.63 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/PreEqDecayer.cpp File Reference

```
#include "PreEqDecayer.h"
#include "NuclearMass.h"
#include "PreEqTransitionRateFunc.h"
#include <iostream>
#include <math.h>
#include <fstream>
#include <iomanip>
#include <cstdlib>
#include <omp.h>
```

Variables

- unsigned int randomSeed [12]

8.63.1 Variable Documentation

8.63.1.1 unsigned int randomSeed[12]

8.64 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/PreEqTransitionRateFunc.cpp File Reference

```
#include "PreEqTransitionRateFunc.h"
#include "ParticleTransmissionFunc.h"
#include <iostream>
```

8.65 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/RauscherLevelDensity.cpp File Reference

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

8.66 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Sapphire.cpp File Reference

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <sstream>
#include <time.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <algorithm>
#include "DecayController.h"
#include "NuclearMass.h"
#include "DecayResults.h"
#include "CrossSection.h"
#include "omp.h"
#include "TransitionRateFunc.h"
#include "ParticleTransmissionFunc.h"
#include "GammaTransmissionFunc.h"
```

Classes

- struct EntrancePairs

Typedefs

- typedef struct EntrancePairs EntrancePairs

Functions

- void Initialize ()
- void printHelp ()
- void parseCommandLineForOptions (std::vector< std::string > &args, int &suffixNo, bool &preEq, int &numPiParticles, int &numPiHoles, int &numNuParticles, int &numNuHoles, bool &calcAverageWidth, bool &calcRates, bool &asciiIn, std::string &inFile, int &entranceState, std::vector< int > &exitStates, bool &printTrans)
- bool parseCommandLineForDecay (std::vector< std::string > &args, int &Z, int &A, double &J, int &Pi, double &lowEnergy, double &highEnergy, int &events)
- bool parseCommandLineForXS (std::vector< std::string > &args, int &Z, int &A, int &pType, std::string &energyFile, bool asciiIn)
- int main (int argc, char *argv[])

Variables

- unsigned int randomSeed [12]

8.66.1 Typedef Documentation

8.66.1.1 typedef struct EntrancePairs EntrancePairs

8.66.2 Function Documentation

8.66.2.1 void Initialize ( )

... text ...

References CrossSection::CreateMACSEnergiesVector(), CrossSection::CreateTempVector(), CoulFunc::GSLErrorHandler(), NuclearMass::InitializeElements(), GammaTransmissionFunc::InitializeGDRParameters(), NuclearLevels::InitializeLevels(), NuclearMass::InitializeMasses(), ParticleTransmissionFunc::SetAlphaFormalism(), CrossSection::SetCalculateGammaCutoff(), Decayer::SetCrossSection(), PreEqDecayer::SetCrossSection(), GammaTransmissionFunc::SetEGDRType(), TransitionRateFunc::SetGammaCutoffEnergy(), Decayer::SetMaxL(), PreEqDecayer::SetMaxL(), ParticleTransmissionFunc::SetNeutronFormalism(), GammaTransmissionFunc::SetPorterThomas(), ParticleTransmissionFunc::SetPorterThomas(), ParticleTransmissionFunc::SetProtonFormalism(), CrossSection::SetResidualAlpha(), CrossSection::SetResidualGamma(), CrossSection::SetResidualNeutron(), CrossSection::SetResidualProton(), and sourceDirectory().

Referenced by main().

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

References DecayResults::AddResults(), CrossSection::CalcAverageDWaveResWidth(), CrossSection::CalcAveragePWaveResWidth(), CrossSection::CalcAverageSWaveResWidth(), CrossSection::Calculate(), CrossSection::CalculateReactionRates(), DecayController::Decay(), DecayController::DecayProducts(), Initialize(), CrossSection::IsValid(), parseCommandLineForDecay(), parseCommandLineForOptions(), parseCommandLineForXS(), CrossSection::PrintCrossSections(), DecayController::PrintDecays(), printHelp(), CrossSection::PrintReactionRates(), CrossSection::PrintTransmissionTerms(), randomSeed, and Decayer::SetCrossSection().

8.66.2.3 bool parseCommandLineForDecay (std::vector< std::string > & *args,* int & *Z,* int & *A,* double & *J,* int & *Pi,* double & *lowEnergy,* double & *highEnergy,* int & *events*)

References NuclearMass::FindZ().

Referenced by main().

8.66.2.4 void parseCommandLineForOptions (std::vector< std::string > & *args,* int & *suffixNo,* bool & *preEq,* int & *numPiParticles,* int & *numPiHoles,* int & *numNuParticles,* int & *numNuHoles,* bool & *calcAverageWidth,* bool & *calcRates,* bool & *asciiIn,* std::string & *inFile,* int & *entranceState,* std::vector< int > & *exitStates,* bool & *printTrans*)

References TransitionRateFunc::GetGammaCutoffEnergy(), Decayer::GetMaxL(), ParticleTransmissionFunc::SetAlphaFormalism(), CrossSection::SetCalculateGammaCutoff(), GammaTransmissionFunc::SetEGDRType(), TransitionRateFunc::SetGammaCutoffEnergy(), Decayer::SetMaxL(), ParticleTransmissionFunc::SetNeutronFormalism(), GammaTransmissionFunc::SetPorterThomas(), ParticleTransmissionFunc::SetPorterThomas(), ParticleTransmissionFunc::SetProtonFormalism(), CrossSection::SetResidualAlpha(), CrossSection::SetResidualGamma(), CrossSection::SetResidualNeutron(), and CrossSection::SetResidualProton().

Referenced by main().

8.66.2.5 bool parseCommandLineForXS (std::vector< std::string > & *args,* int & *Z,* int & *A,* int & *pType,* std::string & *energyFile,* bool *asciiIn*)

References NuclearMass::FindZ().

Referenced by main().

8.66.2.6 void printHelp ( )

Referenced by main().

8.66.3 Variable Documentation

8.66.3.1 unsigned int randomSeed[12]

Referenced by Decayer::Decay(), PreEqDecayer::Decay(), main(), ParticleTransmissionFunc::operator()(), and GammaTransmissionFunc::operator()().

8.67 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/Setup.cpp File Reference

```
#include "NuclearMass.h"
#include "GammaTransmissionFunc.h"
#include "NuclearLevels.h"
#include "Decayer.h"
#include "Sapphire_config.h"
#include "TransitionRateFunc.h"
#include "CrossSection.h"
#include "PreEqDecayer.h"
#include "ParticleTransmissionFunc.h"
#include "CoulFunc.h"
#include <iostream>
#include <gsl/gsl_errno.h>
```

Functions

- void Initialize ()

8.67.1 Function Documentation

8.67.1.1 void Initialize ( )

... text ...

References CrossSection::CreateMACSEnergiesVector(), CrossSection::CreateTempVector(), CoulFunc::GSLErrorHandler(), NuclearMass::InitializeElements(), GammaTransmissionFunc::InitializeGDRParameters(), NuclearLevels::InitializeLevels(), NuclearMass::InitializeMasses(), ParticleTransmissionFunc::SetAlphaFormalism(), CrossSection::SetCalculateGammaCutoff(), Decayer::SetCrossSection(), PreEqDecayer::SetCrossSection(), GammaTransmissionFunc::SetEGDRType(), TransitionRateFunc::SetGammaCutoffEnergy(), Decayer::SetMaxL(), PreEqDecayer::SetMaxL(), ParticleTransmissionFunc::SetNeutronFormalism(), GammaTransmissionFunc::SetPorterThomas(), ParticleTransmissionFunc::SetPorterThomas(), ParticleTransmissionFunc::SetProtonFormalism(), CrossSection::SetResidualAlpha(), CrossSection::SetResidualGamma(), CrossSection::SetResidualNeutron(), CrossSection::SetResidualProton(), and sourceDirectory().

Referenced by main().

8.68 /afs/crc.nd.edu/user/p/pscholz/Private/sapphire-devel/src/TransitionRateFunc.cpp File Reference

```
#include "TransitionRateFunc.h"
#include "ParticleTransmissionFunc.h"
#include "GammaTransmissionFunc.h"
#include "RauscherLevelDensity.h"
#include "NuclearLevels.h"
#include <iostream>
#include <stdlib.h>
```

Index