MANUAL

INL/EXT-19-55002
Revision 0
Printed August 1, 2019

PHISICS User Manual

Andrea Alfonsi, Aaron S. Epiney, Yaqi Wang, Paolo Balestra, Cristian Rabiti

Prepared by
Idaho National Laboratory
Idaho Falls, Idaho 83415

Idaho National Laboratory

INL/EXT-19-55002
Revision 0
Printed August 1, 2019

PHISICS User Manual

Andrea Alfonsi

Aaron S. Epiney

Yaqi Wang

Paolo Balestra

Cristian Rabiti

Contents

1 Introduction

PHISICS (Parallel and Highly Innovative Simulation for INL Code System) is a reactor physics package developed at the Idaho National Laboratory. It is composed of several modules: a nodal and semi-structured transport core solver (INSTANT), a depletion module (MRTAU), a time-dependent solver (TimeIntegrator), a cross-section interpolation and manipulation framework (MIXER), a SuperHomogenization scheme (SPH) a criticality search module (CRITICALITY), a fuel management and shuffling component (SHUFFLE) and a coupling API with RELAP5-3D.

PHISICS has been built in a highly modular fashion so that different physical models can be activated or deactivated depending on the physics relevant to the particular type of analysis. This approach simplifies development as well as mid- and long-term maintenance of the code. PHISICS is entirely written in FORTRAN 95/2003. Each of its components is fully parallelized and designed to take full advantage of shared-memory computers (multiprocessor desktops and workstations) and middle to large high-performance computing systems (10 to 1000 processors).

The aim of this document is to detail the input requirements for PHISICS (and all its modules) focusing on the input structure.

2 Manual Formats

In order to highlight some parts of the Manual having a particular meaning (e.g. input structure, examples, terminal commands, etc.), specific formats have been used. In this sections all the formats with a specific meaning are reported:

- ***Python Coding:***

```
class AClass():
  def aMethodImplementation(self):
    pass
```

- ***XML input example:***

```
<MainXMLBlock>
  ...
  <anXMLnode name='anObjectName' anAttribute='aValue'>
     <aSubNode>body</aSubNode>
  </anXMLnode>
  ...
</MainXMLBlock>
```

- ***Bash Commands:***

```
cd PHISICS/doc
./make_docs.sh
```

3 Installation

3.1 Overview

The PHISICS software is always released in source form. A compilation of the software is always necessary.

In the following sections, the recommended installation/compilation procedure is outlined. For alternatives, we encourage checking the RAVEN wiki. The machines on which PHISICS is tested and developed, however, use the standard installation procedures outlined below.

The installation process will involve three steps:

- Installing prerequisites, which depends on your operating system (mostly compilers);
- Setting up CMake;
- Compiling/installing PHISICS.

3.2 Pre-requisites

PHISICS has few pre-requisites and some optional ones in case advanced features need to be activated. The baseline in terms of dependencies is represented by:

- Git manager downloadable at ***git-scm website***
- Fortran compiler. PHISICS can compile with any Fortran90+ compilers:
  - GCC (see ***gcc***)
  - Intel Fortran Compiler (see ***intel***)
  - etc
- CMake, downloadable at ***here*** .
- Lapack and Blas. Both packages can be downloaded at ***netlib***, ***openblas*** or are distributed with the compiler of choice (e.g. INTEL fortran compiler - MKL)

CMake, *Fortran Compiler* and *Lapack/Blas* allow to compile PHISICS in serial mode.
If the ***MPI*** version (parallel) wants to be generated, an MPI library needs to be downloaded and installed(e.g ***MPICH*** , ***OPEN-MPI*** , ***INTEL-MPI*** , etc.)

3.2.1 Triangle

PHISICS supports unstructured mesh. In order to use this feature, an additional dependency needs to be downloaded: ***Triangle*** . *Triangle* generates exact Delaunay triangulations, constrained Delaunay triangulations, conforming Delaunay triangulations, Voronoi diagrams, and high-quality triangular meshes.
Once the library is obtained, its content needs to be placed in the folder:

```
./PHISICS/contrib/Triangle
```

in order to be found by the CMake compilation procedure.

3.3 Obtaining PHISICS

The code can be directly downloaded from INL GITLAB server. In order to be able to clone PHISICS, SSH Keys need to be set. To add a new SSH Key (and get help on how to generate it) see the following link ***here*** . Once a SSH key has been generated, just type the following command in your terminal:

```
   git clone git@hpcgitlab.hpc.inl.gov:Phisics/PHISICS.git
```

3.4 CMake configuration

As mentioned in the previous sections, PHISICS uses **CMake** to maintain and generate its Make-File, which is used for the compilation process.
Once **CMake** is downloaded and installed, the *CMakeLists.txt* contained in the folder:

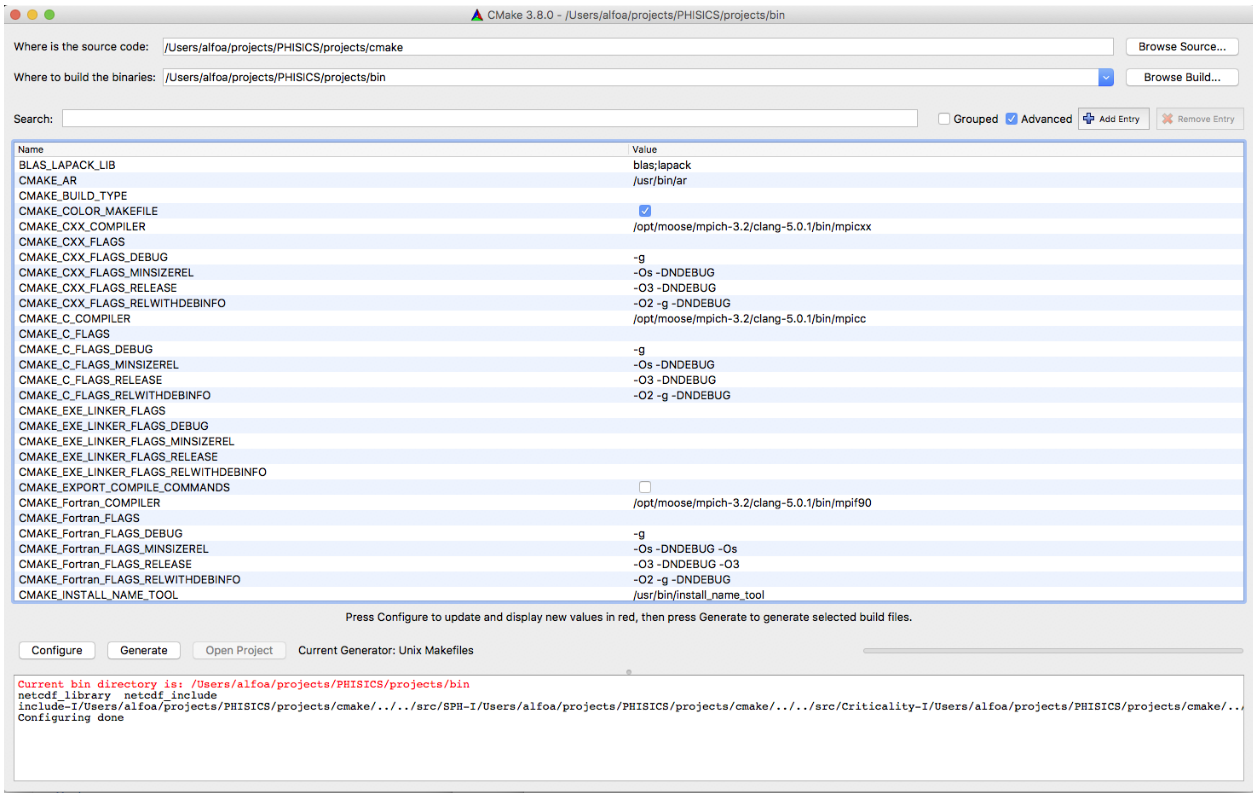
```
   ./PHISICS/projects/cmake/
```

can be open.

In the following subsections, a brief explaination of CMake is reported, depending on the Operative System.

3.4.1 Running CMake for Windows

Run CMakeSetup.exe, which should be in in the Start menu under Program Files, there may also be a shortcut on your desktop, or if built from source, it will be in the build directory. A GUI will appear similar to what is shown below (but possibly different, as CMake is still being developed).



Where is the source code: /Users/alfoa/projects/PHISICS/projects/cmake

Where to build the binaries: /Users/alfoa/projects/PHISICS/projects/bin

Search:

Name	Value
BLAS_LAPACK_LIB	blas;lapack
CMAKE_AR	/usr/bin/ar
CMAKE_BUILD_TYPE	
CMAKE_COLOR_MAKEFILE	✓
CMAKE_CXX_COMPILER	/opt/moose/mpich-3.2/clang-5.0.1/bin/mpicxx
CMAKE_CXX_FLAGS	
CMAKE_CXX_FLAGS_DEBUG	-g
CMAKE_CXX_FLAGS_MINSIZEREL	-Os -DNDEBUG
CMAKE_CXX_FLAGS_RELEASE	-O3 -DNDEBUG
CMAKE_CXX_FLAGS_RELWITHDEBINFO	-O2 -g -DNDEBUG
CMAKE_C_COMPILER	/opt/moose/mpich-3.2/clang-5.0.1/bin/mpicc
CMAKE_C_FLAGS	
CMAKE_C_FLAGS_DEBUG	-g
CMAKE_C_FLAGS_MINSIZEREL	-Os -DNDEBUG
CMAKE_C_FLAGS_RELEASE	-O3 -DNDEBUG
CMAKE_C_FLAGS_RELWITHDEBINFO	-O2 -g -DNDEBUG
CMAKE_EXE_LINKER_FLAGS	
CMAKE_EXE_LINKER_FLAGS_DEBUG	
CMAKE_EXE_LINKER_FLAGS_MINSIZEREL	
CMAKE_EXE_LINKER_FLAGS_RELEASE	
CMAKE_EXE_LINKER_FLAGS_RELWITHDEBINFO	
CMAKE_EXPORT_COMPILE_COMMANDS	☐
CMAKE_Fortran_COMPILER	/opt/moose/mpich-3.2/clang-5.0.1/bin/mpif90
CMAKE_Fortran_FLAGS	
CMAKE_Fortran_FLAGS_DEBUG	-g
CMAKE_Fortran_FLAGS_MINSIZEREL	-Os -DNDEBUG -Os
CMAKE_Fortran_FLAGS_RELEASE	-O3 -DNDEBUG -O3
CMAKE_Fortran_FLAGS_RELWITHDEBINFO	-O2 -g -DNDEBUG
CMAKE_INSTALL_NAME_TOOL	/usr/bin/install_name_tool

Press Configure to update and display new values in red, then press Generate to generate selected build files.

Current Generator: Unix Makefiles

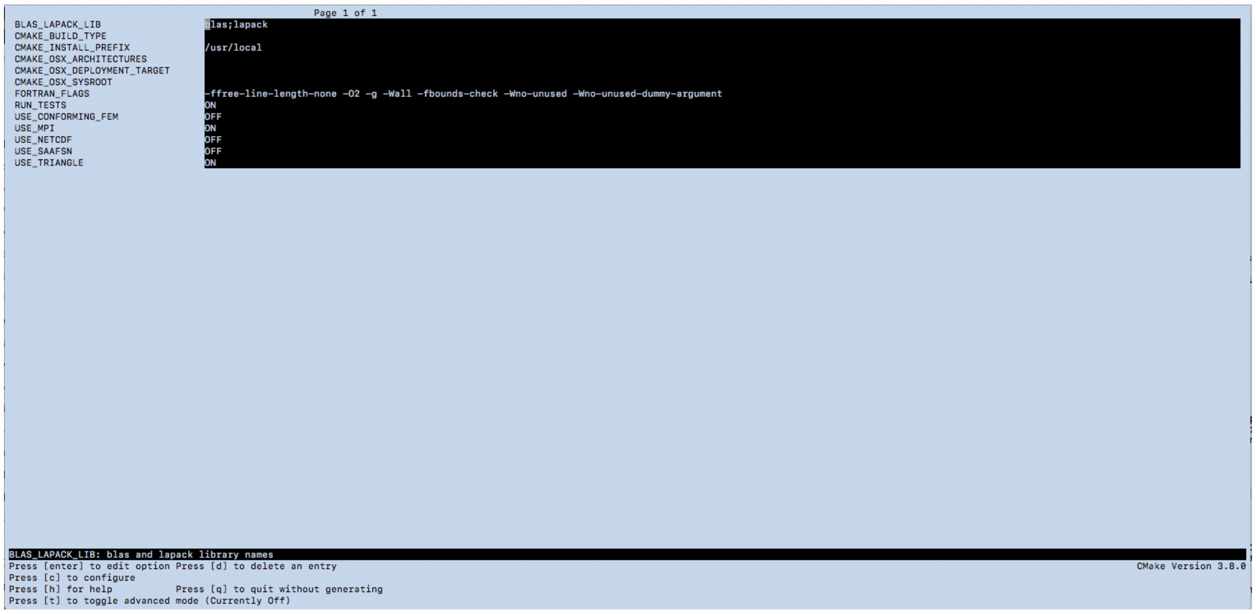
```
Current bin directory is: /Users/alfoa/projects/PHISICS/projects/bin
netcdf_library  netcdf_include
include-I/Users/alfoa/projects/PHISICS/projects/cmake/../../src/SPH-I/Users/alfoa/projects/PHISICS/projects/cmake/../../src/Criticality-I/Users/alfoa/projects/PHISICS/projects/cmake/../
Configuring done
```

Figure 1: CMake GUI Windows

The top two entries are the source code and binary directories. They allow to specify where the PHISICS *CMakeLists.txt* (located in *./PHISICS/projects/cmake/*) is and where the resulting binaries should be placed. If the binary directory specified does not exist, it will be created at the configuration stage.

3.4.2 Running CMake on Unix

On most unix platforms, if the curses library is supported, cmake will build an executable called ccmake. This interface is a terminal based text application that is very similar to the windows GUI. To run ccmake, change directories into the directory where you want the binaries to be placed (e.g. *./PHISICS/projects/bin*).Then run ***ccmake*** with the path to the directory containing PHISICS *CMakeLists.txt* (located in *./PHISICS/projects/cmake/*) on the command line. This will start the text interface (see Fig. 2). Hitting the "c" key, it will configure the project. That should be used as values in the cache get changed. To change values, use the arrow keys to select cache entries, and the enter key to edit them. Boolean values will toggle with the enter key. Once the values are set, hitting the "G" key will generate the makefiles and exit. You can also hit "h" for help, "q" to quit, and "t" to toggle the viewing of advanced cache entries.



```
                                          Page 1 of 1
 BLAS_LAPACK_LIB                  blas;lapack
 CMAKE_BUILD_TYPE
 CMAKE_INSTALL_PREFIX             /usr/local
 CMAKE_OSX_ARCHITECTURES
 CMAKE_OSX_DEPLOYMENT_TARGET
 CMAKE_OSX_SYSROOT
 FORTRAN_FLAGS                    -ffree-line-length-none -O2 -g -Wall -fbounds-check -Wno-unused -Wno-unused-dummy-argument
 RUN_TESTS                        ON
 USE_CONFORMING_FEM               OFF
 USE_MPI                          ON
 USE_NETCDF                       OFF
 USE_SAAFSN                       OFF
 USE_TRIANGLE                     ON

BLAS_LAPACK_LIB: blas and lapack library names
Press [enter] to edit option Press [d] to delete an entry                                    CMake Version 3.8.0
Press [c] to configure
Press [h] for help           Press [q] to quit without generating
Press [t] to toggle advanced mode (Currently Off)
```

Figure 2: CMake Interface Unix

3.5 Compilation

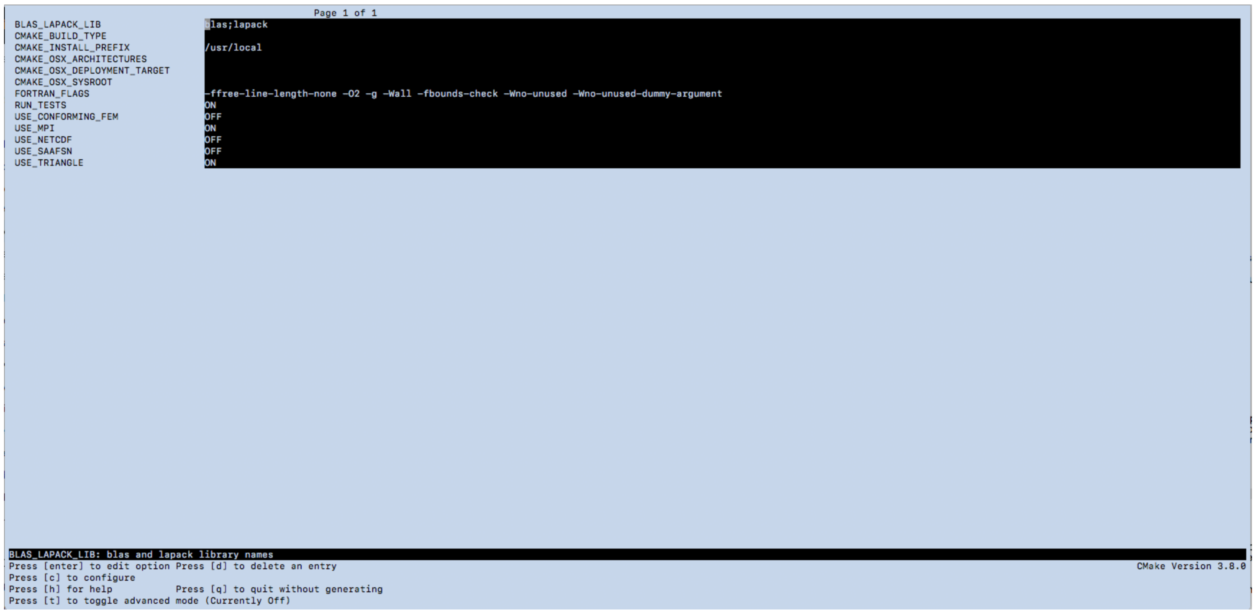
Once the PHISICS dependencies and pre-requisites have been installed (see section 3.2), the PHISICS can be compiled.

The compilation of PHISICS involves the following steps:

- Creation of the PHISICS MakeFile via CMake,
- Compilation,
- Testing

3.5.1 Creation of the PHISICS MakeFile via CMake

Once CMake is installed and the source code present,navigate to `PHISICS/projects` and make a new directory (e.g. `bin`), which will contain the compiled binary files at the end of the compilation process.
Depending on the Operative System (either Unix or Windows), follow the instructions in section 3.4 for opening the PHISICS CMake file (i.e. *CMakeLists.txt*).
Independently on the OS, the procedure to follow is the same. Hence,without loss of generality, in the following it is assumed that the OS is unix-based (e.g. Linux, Apple IOS, etc.).



```
                                    Page 1 of 1
BLAS_LAPACK_LIB                 blas;lapack
CMAKE_BUILD_TYPE
CMAKE_INSTALL_PREFIX            /usr/local
CMAKE_OSX_ARCHITECTURES
CMAKE_OSX_DEPLOYMENT_TARGET
CMAKE_OSX_SYSROOT
FORTRAN_FLAGS                   -ffree-line-length-none -O2 -g -Wall -fbounds-check -Wno-unused -Wno-unused-dummy-argument
RUN_TESTS                       ON
USE_CONFORMING_FEM              OFF
USE_MPI                         ON
USE_NETCDF                      OFF
USE_SAAFSN                      OFF
USE_TRIANGLE                    ON

BLAS_LAPACK_LIB: blas and lapack library names
Press [enter] to edit option Press [d] to delete an entry                                          CMake Version 3.8.0
Press [c] to configure
Press [h] for help           Press [q] to quit without generating
Press [t] to toggle advanced mode (Currently Off)
```

Figure 3: PHISICS CMake Interface

For example, if the compilation directory was `PHISICS/projects/bin` the following command (run from the `PHISICS/projects/bin` directory) will open the CMake interface for PHISICS (see Fig. 3, using the definition in the PHISICS *CMakeLists.txt*:

```
ccmake ../cmake/
```

At this point, the compilation options can be inputted. The following steps need to be followed:

- Hit the “c” key (“configure” botton in Windows), followed by the “e” key.
- Hit the “t” key (check the “advanced” box in Windows). This toggles the advanced interface where all the special options can be inputted
- Find the content of CMAKE_Fortran_COMPILER, and hit the “enter” key to edit and replace its content with:
  - *for MPI compilation :* a) “mpiifort” for Intel Fortran Compiler, or b) “mpif90”, for Gfortran Compiler
  - *for serial compilation:* a) “ifort” for Intel Fortran Compiler, or b) “gfortran”, for Gfortran Compiler
- Hit the “enter” key to confirm
- Find the content of CMAKE_C_COMPILER, and hit the “enter” key to edit and replace its content with:

  - *for MPI compilation :* a) “mpiicc” for Intel Fortran Compiler, or b) “mpicc”, for Gfortran Compiler
  - *for serial compilation:* a) “icc” for Intel Fortran Compiler, or b) “gcc”, for Gfortran Compiler

- Hit the “enter” key to confirm
- Hit the “c” key (“configure” botton in Windows) to configure. After the configuration, CMake should have been able to find the C and Fortran compilers in the enviroment (replacing the content of CMAKE_Fortran_COMPILER and CMAKE_C_COMPILER with the full path to the compiler executables). If an Error is raised, the full path to the compilers need to be manually inputted in the relative CMAKE variables.
- Hit the “e” key
- Find the content of FORTRAN_FLAGS, and hit the “enter” key to edit and replace its content with the fortran flags of choice. For example:
  - *for Intel Compilers:*
    * *DEBUG:* $-O0 - g - traceback - checkall - debugall.$
    * *RELEASE:* $-O2$ or $-O3$

    In addition, the default precision for integers and reals can be inputted: $-i8 - r8$, double precision; $-i4$ single precision.
  - *for Gfortran Compilers:*
    * *DEBUG:* $-O0 - Wall - fbounds - check - Wno - unused - Wno - unused - dummy - argument - fcheck = all - fbacktrace - g3 - nostdinc - cpp.$
    * *RELEASE:* $-O2 - nostdinc - cpp$ or $-O3 - nostdinc - cpp$
- Hit the “c” key (“configure” botton in Windows) to configure.
- Find the boolean CMAKE entries, such as:
  - *USE_CONFORMING_FEM:* Hit “enter” key to change its value from OFF to ON. “ON” is experimental.
  - *USE_MPI:* Hit “enter” key to change its value from OFF to ON. “ON” is for MPI (parallel) version of the software, “OFF” is for serial.
  - *USE_TRIANGLE:* Hit “enter” key to change its value from OFF to ON. “ON” is for including the Triangle library (contrib for unstructured mesh)
  - *RUN_TESTS:* Hit “enter” key to change its value from ON to OFF. This flag allows to enable or disable the execution of regression tests at the end of the compilation.

  A typical configuration is having USE_MPI and RUN_TESTS to ON only.

- Hit the “c” key (“configure” botton in Windows) to configure.
- Hit the “g” key (“Generate” botton in Windows) to generate the make file and exit the CMAKE interface.

At this point, the PHISICS MakeFile is generated and the compilation can be performed: Navigate to the folder where the MakeFile has been generated (e.g. `PHISICS/projects/bin`) and type:

```
make -j <number-processors>
```

where $< number - processors >$ must be replaced with the number of CPUs need to be used for the compilation (e.g. $make - j4$, using 4 CPUs).
At the end of the compilation, the following executables are going to be generated:

3.5.1.1 INSTANT

The transport core solver **INSTANT** (executable *instant*) is the key kernel of the **PHISICS** framework. **INSTANT** is parallelized and is designed to take full advantage of medium to large clusters (10 to 1000 processors). It is based on the second-order formulation of the transport equation discretized in angle by spherical harmonics while in space it uses orthonormal polynomials of an arbitrary order. In addition to steady-state solutions, **INSTANT** is able to solve time-dependent problems. For that, a scheme based on a second-order backward Euler scheme with explicit delayed neutron treatment has been implemented as a new module for the **PHISICS** suite.

3.5.1.2 MRTAU

MRTAU (executable *mrtau_standalone*) is a generic depletion code developed at INL. In addition to core depletion, the code can be utilized for stand-alone decay heat calculations. It tracks the time evolution of the isotopic concentration of a given material accounting for nuclear reactions occurring in the presence of neutron flux and also due to natural decay (Bateman equation). The code uses a Taylor series expansionbased algorithm at arbitrary order and the Chebyshev Rational Approximation Method (CRAM) for computation of the exponential matrix.

3.5.1.3 MIXER

The **MIXER** (executable *mixer*) module does all the cross-section handling for the different kernels. **MIXER** can handle macroscopic, microscopic, and “mixed” cross sections. A macroscopic cross-section library contains macroscopic cross sections for each type of material used in the calculation (fuel, reflector, etc.) tabulated for the state parameters (temperature, burnup, control rod position, etc.). **MIXER** interpolates these cross sections at the requested state parameters, and no limits in tabulation dimensions or neutron energy groups exist. A microscopic or mixed cross section library contains the tabulated cross sections for each isotope considered in the calculation. **MIXER** reads a description containing a list of isotopes and corresponding densities for each material and interpolates the microscopic cross sections at the requested state parameters. Macroscopic cross sections for each material are generated with the corresponding number densities. With this capability, mixed macroscopic and microscopic cross sections are also possible. For example, it is possible to provide macroscopic absorption cross sections without xenon for a material and the microscopic xenon absorption cross section together with a xenon density. **MIXER** will calculate the xenon absorption contribution and add it to the macroscopic cross section. **MIXER** can read different cross-section library formats. Among them is an original simple XML-based format, but also AMPX, ISOTXS, and ECCO library formats that allow cross-section libraries to be prepared with SCALE, ERANOS, or MC2. More library types are planned to be supported in the future.

3.5.1.4 SPH

The **SPH** (executable *sph_run*) module is aimed to employ an SuperHomogenization scheme for the generation of corrected neutron cross sections. It allows for the generation of SPH factors that can be later applied for any PHISICS calculation coupled and not with RELAP5-3D.

3.5.1.5 Quasi-Static

The **Quasi-Static** (executable *qs_stat*) module is aimed to deploy oll the capabilities for employing time-dependent transport calculation with a quasi static advancement scheme. This executable is mostly a test program, since the main features are deployed for the coupling with RELAP5-3D.

3.5.1.6 TimeIntegrator

The **TimeIntegrator** (executable *time_dep*) module is aimed to deploy oll the capabilities for employing time-dependent transport calculation. This executable is used to perform time-dependent calculations in standalone mode (no coupled with RELAP5-3D). For example, it can be used for control-rod ejection transients, etc.

3.5.1.7 PHISICS

PHISICS (executable *dpl_instant_run*) is the driver that combines all the other nuclear physics kernels to perform time-dependent, depletion-based core simulation, including multicycle fuel shuffling maneuvers. This is the crowning binary of PHISICS.

3.5.2 Testing PHISICS

To test the installation of PHISICS, navigate to `PHISICS/tests`, then run the command

```
./run_tests -j2
```

where `-j2` signifies running with 2 processors. If more processors are available, this can be increased, but using all or more than all of the available processes can slow down the testing dramatically. This command runs PHISICS's regression tests, analytic tests, and unit tests. The number of tests changes frequently as the code's needs change, and the time taken to run the tests depends strongly on the number of processors and processor speed.

At the end of the tests, a number passed, skipped, and failing will be reported. Having some skipped tests is expected; PHISICS has many tests that apply only to particular configurations or codes that are not present on all machines. However, no tests should fail; if there are problems, consult the troubleshooting section on the PHISICS wiki.

The $./run\_tests$ script accepts several command line arguments that can be inputted to perform specific tasks:

- $-pbd$, this option can be used to specify the specific directory where the executables have been generated. If not provided, the Test suite assumes that the executables are located in `PHISICS/projects/bin`.
  Usage example: $./run\_tests\ -\ pbd\ /projects/PHISICS/bin\_debug$.
- $--re$, this option can be used to specify the spefic test or test group to be executated.
  Usage example: $./run\_tests --re = DECAY\_HEAT\_SM/decay\_heat\_dmd\_operational$.

3.5.3 Updating PHISICS

PHISICS updates frequently, and new features are added while bugs are fixed on a regular basis. To update PHISICS, navigate to `PHISICS`, then run the commands

```
git pull
```

Then navigate to the binary folder used for the compilation and run "make" again.

3.5.4 In-use Testing

At any time, tests can be checked by re-running the installation tests as described in Section 3.5.2.

4 PHISICS Input Structure

As mentioned in the previous sections, PHISICS is a modular software composed by different modules/calculation schemes.
This means that, depending on the module/executable to be run, one or more input files need to be provided.
The main input XML blocks/files can be categorized as follows:

- **`<INSTANT_INPUTS>`**: In this XML block all the settings/options for controlling/ modeling the transport/diffusion solver INSTANT are inputted. Geometry, Mesh, PN order, etc.
- **`<INSTANT_TIME_DEP>`**: In this XML block all the settings/options for controlling/ modeling the transport/diffusion solver INSTANT (in time-dependent mode) are inputted
- **`<DENSITIES>`**: In this XML block all the initial compositions of the different zones in the mesh (geometry) are defined
- **`<CRITICALITY_SEARCH>`**: In this XML block all the settings/options for activating the criticality search module are inputted
- **`<XS-library>`**: In this XML block all the cross-sections (Micro or Macro or Mixed) are inputted (either in PHISICS native XML format or other supported formats)
- **`<DEPLETION_INPUT>`**: In this XML block all depletion information is inputted.
- **`<SHUFFLE_INPUT>`**: In this XML block all the information for simulating fuel management and multi-cycle operations is inputted.

Other XML inputs/blocks are available depending on the calculation scheme/module to be used. Each of these blocks are explained in dedicated sections in the following chapters.

5 INSTANT_INPUTS

As previously mentioned, In the **<INSTANT_INPUTS>** XML input/main block all the settings/options for controlling/modeling the transport/diffusion solver INSTANT are inputted. Geometry, Mesh, PN order, etc.
An example of the this block is reported below:

```
<INSTANT_INPUTS>
  <Control_parameters>
     <title>Azmy 1-group benchmark problem</title>
     <calcul_typ>2</calcul_typ>
  </Control_parameters>
  <Output_controls>
     <vtkgen>1</vtkgen>
  </Output_controls>
  <Mesh>
     <multimesh>0</multimesh>
     <pn>3</pn>
     <pint>2</pint>
     <psurf>0</psurf>
     <psour>2</psour>
  </Mesh>
  <Materials>
     <Macros NG="1">
        <material ID="1" NA="0" fissile="false">
          <name>material for benchmark</name>
          <TotalXS>1.0</TotalXS>
          <Profile>1 1</Profile>
          <ScatteringXS>0.5</ScatteringXS>
        </material>
        <material ID="2" NA="0" fissile="false">
          <name>material for benchmark</name>
          <TotalXS>2.0</TotalXS>
          <ScatteringXS>0.1</ScatteringXS>
        </material>
     </Macros>
  </Materials>
  <Source>
     <vsrc_filename>inp.xml</vsrc_filename>
     <Volumetric_Sources n_energy_groups="1">
           <VSource ID="1" primal="true">
             <Item moment_n="0" moment_k="0" group="1">
```

```
               1.0
             </Item>
        </VSource>
      </Volumetric_Sources>
   </Source>
   <IterativeSolver>
      <echo>3</echo>
      <epsinner>1.0E-07</epsinner>
      <maxinner>600</maxinner>
      <PNinner>0</PNinner>
   </IterativeSolver>
   <Geometry>
     <NX>2</NX><NY>2</NY>
     <DX>5.0 5.0</DX>
     <DY>5.0 5.0</DY>
     <BC>1 0 1 0</BC>
     <prerefine>4</prerefine>^{    }
     <MaterialID>
      1 2
      2 2
     </MaterialID>
     <SourceID>
      1 0
      0 0
     </SourceID>
     <SubDomainID>
      1 2
      2 2
     </SubDomainID>
   </Geometry>
</INSTANT_INPUTS>
```

Each of these blocks are explained in dedicated sections in the following sections.

5.1 *Control_parameters*

Parameters in this block or element describe the problem and determine what type of calculations will be conducted.

5.1.1 *title*

Description: a short comment on the input deck

Data type: a string with the maximum length of 132. (Extra characters will be dropped.)

Default value: 'INSTANT test problem'

5.1.2 *ndim*

Description: the dimension of the problem to be solved

Data type: integer

Default value: 2

Selection: 2/3 – 2D/3D

Note: This value must be consistent with the geometry description in Geometry block. Specifically, if ndim is equal to 2, no elements with tags NZ, DZ, IZ in Cartesian geometry and NZ, DZ, IZ ZBC in hexagonal geometry are allowed.

5.1.3 *run*

Description: a flag to show if the problem will be solved

Data type: logical

Default value: true

Selection: true/false – to analysis and run the input deck/to only analysis the input deck

5.1.4 *calcul_typ*

Description: to indicate the calculation type

Data type: integer

Default value: 1

Selection: 1/2 – eigenvalue/source problem

5.1.5 *adjoint*

Description: to determine if the primal and/or adjoint calculations will be performed

Data type: integer

Default value: 0

Selection: 0/1/2 – primal/adjoint/primal+adjoint (Note: only 0 is supported so far.)

5.1.6 *snorpn*

Description: to determine which transport solver is selected

Data type: integer

Default value: 1

Selection: 0/1 – SN+DFEM/PN+HFEM (Note: only PN+HFEM is supported so far.)

5.1.7 *power*

Description: the thermal power in Watts used to normalize the solution of an eigenvalue problem

Data type: real

Default value: -1.0

Note: If the thermal power is non-positive, INSTANT will normalize the solution so that the total fission neutron source is equal to one.

5.2 *Output_controls*

Parameters in this block control how the output files are generated.

5.2.1 *dbggeo*

Description: to determine if debug information about the geometry will be generated in the standard output file

Data type: logical

Default value: false

Selection: true/false – generate/not generate

Note: if dbggeo is turned on, four files, *Initial-Mesh-px.ps*, *Initial-Mesh-px.vtk*, *Initial-Material-px.vtk* and *Initial-Color-px.vtk* will be generated. x is the processor ID, equal to 0 if there is no parallelization. *Initial-Material-px.vtk* contains the material distribution over the solution domain. *Initial-Color-px.vtk* tells how the solution domain is painted for the multi-color iteration (red-black algorithm). The other two files show the mesh.

5.2.2 ***dbgmsh***

Description: to determine if debug information about the mesh will be generated in the standard output file

Data type: logical

Default value: false

Selection: true/false – generate/not generate

5.2.3 ***dbgsrc***

Description: to determine if debug information about the external source will be generated in the standard output file

Data type: logical

Default value: false

Selection: true/false – generate/not generate

5.2.4 ***dbgmat***

Description: to determine if debug information about the materials will be generated in the standard output file

Data type: logical

Default value: true

Selection: true/false – generate/not generate

5.2.5 ***dbgmlg***

Description: to determine if debug information about the multigroup driver will be generated in the standard output file

Data type: logical

Default value: true

Selection: true/false – generate/not generate

5.2.6 ***dbging***

Description: to determine if debug information about the in-group driver will be generated in the standard output file

Data type: logical

Default value: true

Selection: true/false – generate/not generate

5.2.7 ***momprod***

Description: to determine to what order the VTK files of flux moments will be generated

Data type: integer

Default value: 1

Selection: 0–PN order – (0 means only VTK file for the scalar flux will be generated; 1 means the scalar flux plus the currents will be generated)

5.2.8 ***vtkgen***

Description: to control the generation of VTK files for flux moments

Data type: integer

Default value: 1

Selection: 0/1 – not generate/generate VTK files

5.2.9 *vtkfile*

Description: the prefix of the VTK file name

Data type: word

Default value: 'flux'

5.2.10 *vtkcellcut*

Description: to control how many cuts in each of cells to generate VTK files for the flux moments and the fission source

Data type: integer

Default value: 2

Note: zero means no cuts. In this case, only solution values on the cell vertices will be outputted into the VTK files.

5.2.11 *vtkflatfis*

Description: true to output the flat mode of the fission source only

Data type: logical

Default value: false

Note: when vtkflatfis is turned on, vtkcellcut will not apply. No cell cut will be necessary and will not be conducted.

5.2.12 *csvgeomod*

Description: true to print the csv files by the geometry layout, false to print the csv files in a table format

Data type: logical

Default value: false

Note: csvgeomod = true not available yet.

5.2.13 *flxmatcsv*

Description: true to print the flux by materials in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.2.14 *flxsubcsv*

Description: true to print the flux by subdomains in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.2.15 *pwmatcsv*

Description: true to print the power by materials in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.2.16 *pwsubcsv*

Description: true to print the power by subdomains in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.2.17 *rpwmatcsv*

Description: true to print the relative power by materials in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.2.18 *rpwsubcsv*

Description: true to print the relative power by subdomains in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.2.19 *pwdenmatcsv*

Description: true to print the power density by materials in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.2.20 *pwdensubcsv*

Description: true to print the power density by subdomains in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.2.21 *rpwdenmatcsv*

Description: true to print the relative power density by materials in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.2.22 *rpwdensubcsv*

Description: true to print the relative power density by subdomains in a csv file

Data type: logical

Default value: false

Note: the output format depends on the value of the flag csvgeomod

5.3 *Geometry*

Five types of geometries are supported, the Cartesian, hexagonal geometries, the unstructured geometry described by a Planar Straight Line Graph (PSLG), LWR (Light Water Reactor) geometry and a geometry described in EXODUS-II format. The attribute *type* of Geometry tells which type of geometries is contained in the Geometry. By default the attribute is ”regular”, which is the Cartesian geometry. User can use ”hex” to input hexagonal geometry and use ”PSLG” to input a PSLG and use ”LWR” to input a typical LWR geometry and use ”‘EXODUS-II”’ to read an EXODUS-II file. Depending the type of geometry, there are different sets of elements.

The location of the origin in a geometry is not significant to the calculations. By default, the origin is at the left-bottom vertex in 2D or left-front-bottom vertex in 3D of the first grid in x, y (and z) directions for a Cartesian geometry. The origin is at the center of the central hexagon in 2D or at the center of the bottom hexagonal side of the first central hexagonal cell in z-direction in 3D for a hexagonal geometry. The location of all vertices in a PSLG is explicitly specified. The origin of a LWR geometry is at the left-bottom vertex in 2D or left-front-bottom vertex in 3D of the first assembly in x, y (and z) directions. The location of all vertices in a EXODUS-II is explicitly specified.

5.3.1 Cartesian geometry

5.3.1.1 *NX*

Description: number of intervals in x-direction

Data type: integer

Default value: 1

5.3.1.2 *NY*

Description: number of intervals in y-direction

Data type: integer

Default value: 1

5.3.1.3 *NZ*

Description: number of intervals in z-direction

Data type: integer

Default value: 1

Note: This element should only present in 3-D calculations

5.3.1.4 *DX*

Description: size of intervals in x-direction (cm)

Data type: real array

Default value: 1.0 cm

Note: Number of elements in this real array must be equal to NX.

5.3.1.5 *DY*

Description: size of intervals in y-direction (cm)

Data type: real array

Default value: 1.0 cm

Note: Number of elements in this real array must be equal to NY.

5.3.1.6 *DZ*

Description: size of intervals in z-direction (cm)

Data type: real array

Default value: 1.0 cm

Note: Number of elements in this real array must be equal to NZ. This array is ordered from bottom to top. This element should only present in 3-D calculations.

5.3.1.7 *IX*

Description: subdivision of intervals in x-direction

Data type: integer array

Default value: 1

Note: Number of elements in this integer array must be equal to NX.

5.3.1.8 *IY*

Description: subdivision of intervals in y-direction

Data type: integer array

Default value: 1

Note: Number of elements in this integer array must be equal to NY.

5.3.1.9 *IZ*

Description: subdivision of intervals in z-direction

Data type: integer array

Default value: 1

Note: Number of elements in this integer array must be equal to NZ. This array is ordered from bottom to top. This element should only present in 3-D calculations.

5.3.1.10 *triangulated*

Description: to determine if the geometry is triangulated

Data type: logical

Default value: false

Selection: true/false – true when the geometry is triangulated, i.e., a triangular mesh will be used for calculation.

Note: Every node will be triangulated like in Fig. 4.

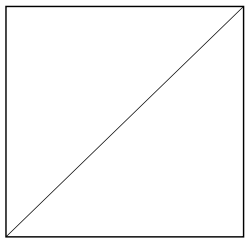


Figure 4: Triangulation of a quad node.

5.3.1.11 *prerefine*

Description: number of times to uniformly subdivide the geometry

Data type: integer

Default value: 0

Note: 0 means no refinement before the calculation

5.3.1.12 *BC*

Description: boundary type in $\mp x$, $\mp y$ and $\mp z$ directions

Data type: integer array

Default value: all zeros

Note: Number of elements in this integer array must be equal to 4 in 2D or 6 in 3D. Supported boundary types are,

- 0 – Vacuum
- 1 – Reflecting
- 2 – Nodal center symmetry (refer to IAEA benchmark problems as examples)
- 3 – Periodic (If $-x$ has 3, then $+x$ must have 3 too. The same applies for y and z directions)

5.3.1.13 *MaterialID*

Description: material IDs in the geometry

Data type: integer array with size being equal to NX*NY in 2D or NX*NY*NZ in 3D

Default value: all ones

Note: Material IDs are ordered in x-direction first then y and z-directions. A cell with 0 material ID means a void and will be cut from the calculation. Rules apply with the placement of void material IDs. In general, interior void materials are not allowed.

5.3.1.14 *SourceID*

Description: external source IDs in the geometry

Data type: integer array with size being equal to NX*NY in 2D or NX*NY*NZ in 3D

Default value: all ones

Note: Source IDs are ordered in x-direction first then y and z-directions. Source IDs are ignored for eigenvalue problems.

5.3.1.15 *ProcessorID*

Description: processor IDs in the geometry

Data type: integer array with size being equal to NX*NY in 2D or NX*NY*NZ in 3D

Default value: all zeros

Note: Processor IDs are ordered in x-direction first then y and z-directions. They are used to do the domain decomposition for parallel calculations will MPI. They are ignored for the serial version of INSTANT.

5.3.1.16 *SubDomainID*

Description: Sub-domain IDs in the geometry

Data type: integer array with size being equal to NX*NY in 2D or NX*NY*NZ in 3D

Default value: all ones

Note: Sub-domain IDs are ordered in x-direction first then y and z-directions. The sub-domain IDs control the generation of balance table in the standard output file. They do not affect the calculation.

5.3.2 Hexagonal geometry

5.3.2.1 *N*

Description: Number of rings surrounding the central hexagon node

Data type: integer

Default value: 0

5.3.2.2 *NZ*

Description: number of intervals in z-direction

Data type: integer

Default value: 1

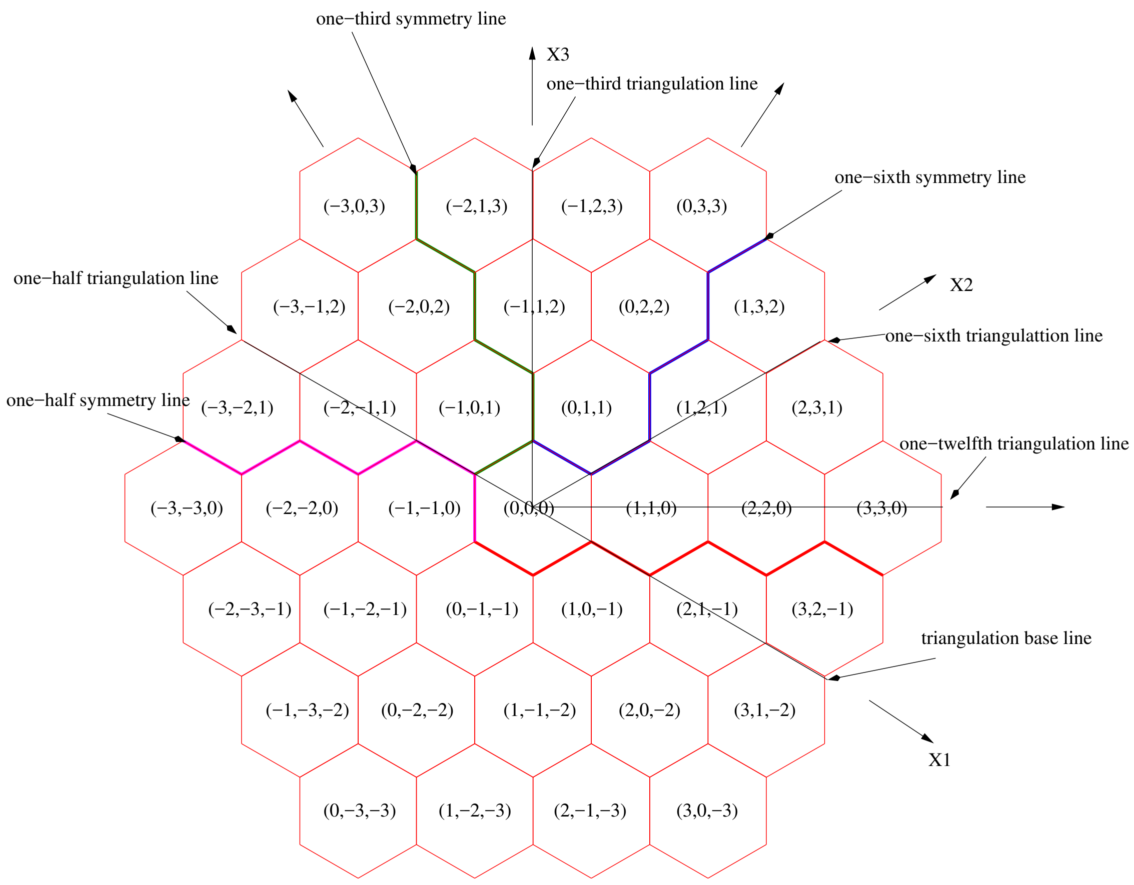


Figure 5: The hexagonal geometry with $N = 3$.

Note: This element should only present in 3-D calculations.

5.3.2.3 *A*

Description: side size of a hexagon node (cm)

Data type: real

Default value: 1.0 cm

5.3.2.4 *DZ*

Description: size of intervals in z-direction (cm)

Data type: real array

Default value: 1.0 cm

Note: number of elements in this real array must be equal to NZ. This array is ordered from bottom to top. This element should only present in 3-D calculations.

5.3.2.5 *IZ*

Description: subdivision of intervals in z-direction

Data type: integer array

Default value: 1

Note: number of elements in this integer array must be equal to NZ. This array is ordered from bottom to top. This element should only present in 3-D calculations.

5.3.2.6 *BC*

Description: condition type of all surrounding boundary sides

Data type: integer

Default value: 0

Selection: 0/1 – vacuum boundary/reflecting boundary.

5.3.2.7 *ZBC*

Description: boundary type in $\mp z$ directions

Data type: integer array

Default value: all zeros

Note: number of elements in this integer array must be equal to 2. Supported boundary types can be found in Section 5.3.1.12.

5.3.2.8 *Symmetry*

Description: symmetry type of the hexagonal geometry in x-y plane

Data type: word

Default value: 'none'

Selection: 'none'/'one-sixth'/'one-third'/'one-half'/ – Symmetry type

Note: this parameter will affect how the arrays of the material, source, processor and subdomain IDs will be inputted. Refer to Fig. 5 for inputting these arrays. Number of hexagon nodes M,

$$M = \begin{cases} (3N(N+1)+1) \times NZ, & 'none' \\ (\frac{1}{2}N(N+1)+1) \times NZ, & 'one-sixth' \\ (N(N+1)+1) \times NZ, & 'one-third' \\ (\frac{3}{2}N(N+1)+1) \times NZ, & 'one-half' \end{cases} \tag{1}$$

5.3.2.9 *triangulated*

Description: to determine if the geometry is triangulated

Data type: logical

Default value: false

Selection: true/false – true when the geometry is triangulated, i.e., a triangular mesh will be used for calculation. The domain boundaries after triangulation with different symmetry configurations are illustrated in Fig. 5. If triangulated is turned on, INSTANT will analyze the symmetry of the one-sixth configuration and will do one-twelfth calculation if the symmetry is recognized.

5.3.2.10 *prerefine*

Description: number of times to uniformly subdivide the geometry

Data type: integer

Default value: 0

Note: 0 means no refinement before the calculation. prerefine will be neglected if triangulated in Geometry block is not turned on.

5.3.2.11 *subnodes*

Description: material IDs of six triangles of a hexagon node

Data type: integer array

Default value: N/A

Note: Every 6 integers indicate the material IDs of six triangles of a hexagon node. Size of this integer array is the number of sub-nodes times 6. Sub-node IDs are numbered from 1. The sub-nodes ID can be referred in the array of the material IDs with the negative value of the sub-node ID. Numbering of six triangles of a hexagon is illustrated in Fig. 6. subnodes will be neglected if triangulated in Geometry block is not turned on.

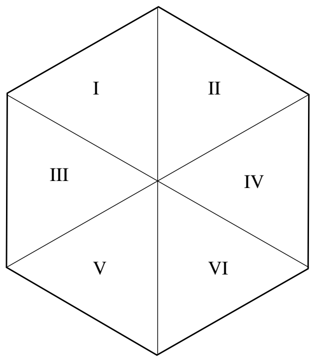


Figure 6: Sub-node of a hexagon.

5.3.2.12 *MaterialID*

Description: material IDs in the geometry

Data type: integer array with size being equal to M

Default value: all ones

Note: Material IDs are inputted as showed in Fig. 5 in the 2D plane. They are then ordered in z-directions in 3D. A cell with 0 material ID means a void and will be cut from the calculation. Negative values means the corresponding hexagon is filled with a sub-node, whose ID is the absolute value.

5.3.2.13 *SourceID*

Description: external source IDs in the geometry

Data type: integer array with size being equal to M

Default value: all ones

Note: Source IDs are ignored for eigenvalue problems. Source IDs are inputted as showed in Fig. 5 in the 2D plane. They are then ordered in z-directions in 3D.

5.3.2.14 *ProcessorID*

Description: processor IDs in the geometry

Data type: integer array with size being equal to M

Default value: all zeros

Note: They are used to do the domain decomposition for parallel calculations will MPI. Processor IDs are inputted as showed in Fig. 5 in the 2D plane. They are then ordered in z-directions in 3D. They are ignored for the serial version of INSTANT.

5.3.2.15 *SubDomainID*

Description: sub-domain IDs in the geometry

Data type: integer array with size being equal to M

Default value: all ones

Note: The sub-domain IDs controls the generation of balance table in the standard output file.

Sub-domain IDs are inputted as showed in Fig. 5 in the 2D plane. They are then ordered in z-directions in 3D.

5.3.2.16 *Remarks about geometry input*

The input of material IDs of the hexagonal geometry is done similarly to the other geometries; once the number of rings surrounding the central hexagonal node **N** and the symmetry conditions **symmetry** are set, the packing of the core is carried on the material ID node **MaterialID**. The packing is performed from top left to right bottom. For example. let's consider a 2-ring hexagonal block (the N node is set to 2) modeled with 4 different materials in full core (the symmetry node is set to none). The material ID is set as follows:

```
        1  4  1
      1  2  2  1
    1  2  3  2  1
      4  2  2  4
        1  1  1
```

The core built is pictured below (Fig. 7), along with the position number of the blocks in the material ID node. For example, the 10th entry in the material ID is defined as material "3". The

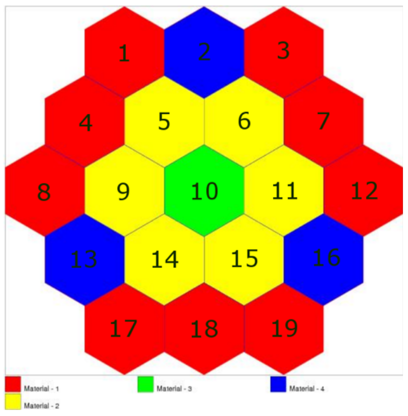


Figure 7: Hexagonal geometry no symmetry example.

equivalent *one-third* core model (the symmetry node is set to one-third) is represented below. The material ID in *one-third* symmetry would be set as follows:

```
        4  1
          2  1
            3  2  1
```

Only the blocks numbered in Fig. 8 is modeled in the material ID node. Note that the blocks split in half or third by the y-symmetry line (block number 1 and block number 5 in the example) shall not be repeated on the x- symmetry line. Hence the right-most blue block, located below block 6 and 7 shall not be modeled in the one-third model.

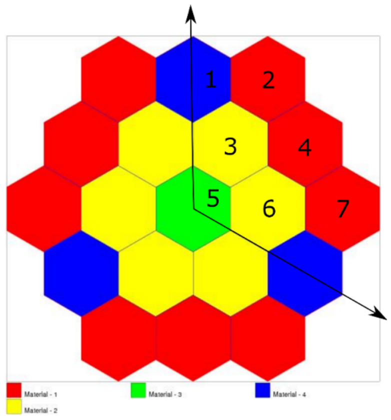


Figure 8: Hexagonal geometry one-third example.

5.3.3 Light Water Reactor (LWR) core geometry

This block is used to describe a typical LWR core geometry. The geometry is further triangulated by INSTANT through calling Triangle, a open-source two-dimensional quality mesh generator and delaunay triangulator developed by Jonathan R. Shewchuk. A typical *LWR core* is composed of *fuel assemblies*. The reflectors surrounding the core can be thought as special assemblies. An assembly is composed of *pin cells*. These pin cells can be a fuel pin, control-rod pin, guide tube, part of assembly box, etc. One can use this hierachical structure to describe the LWR core efficiently.

Different levels of homogenization may be used for the calculation: fully heterogeneous calculation resolving every details of pin cells; pin-cell level homogenization; assembly-wise homogenization. For 2D geometry, INSTANT will generate a unstructured triangular mesh. For 3D geometry, INSTANT will generate a extruded mesh with wedges, which is unstructured in the x-y plane and regular in axial direction. A sample mesh for the 3-by-3 pin problem is showed in Fig. 9.

5.3.3.1 *Controls*

Elements contained in this element control how the LWR geometry be triangulated. If this element is missing, the default values of all child elements will be adopted.

1. *MaxArea*

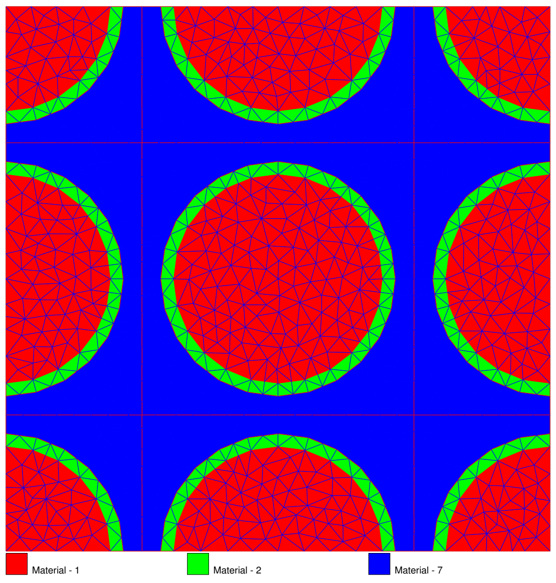


Figure 9: An example unstructured 2D mesh generated with LWR geometry.

Description: the maximum area allowed for all triangles in a fully heterogeneous pin cell (cm^2)

Data type: real

Default value: N/A

Note: By default, there is no area constraint on the triangulation.

2. *PinMaxArea*

   Description: the maximum area allowed for all triangles in a homogenized pin cell (cm^2)

   Data type: real

   Default value: N/A

   Note: By default, there is no area constraint.

3. *AssemblyMaxArea*

   Description: the maximum area allowed for all triangles in a homogenized assembly (cm^2)

   Data type: real

   Default value: N/A

   Note: By default, there is no area constraint.

4. *MinAngle*

   Description: the minimum angle in degree allowed for all triangles in the x-y plane

   Data type: real

Default value: N/A

Note: By default, there is no angle constraint on the triangulation.

5. *DebugOutput*

   Description: if the triangulation debugging files are generated

   Data type: logical

   Default value: false

   Note: if this flag is turned on, several triangulation debugging files will be generated. The prefix of these file names are the core name. Their suffix indicate their contents.

   - '.in.poly' file – the input file to Triangle;
   - '.ele' file – the list of cells;
   - '.node' file – the list of nodes;
   - '.edge' file – the list of edges in the x-y plane;
   - '.neigh' file – the list of triangles neighborings;
   - '.poly' file – the PSLG file generated by Triangle;
   - '.v.node' and '.v.edge' file – the Voronoi diagram;
   - '.off' – an Object File Format (OFF) file, suitable for viewing with the Geometry Center's Geomview package.

5.3.3.2 *Pins*

This block collects all the descriptions of pin cells with the tag name *Pin* having the following attributes:

- *ID*

  Description: the unique pin cell ID (> 0)

  Data type: integer

  Default value: the ordering among all pin cells

  Note: it is recommanded to explicitly give the pin ID.

- *shape*

  Description: the shape of the pin cell

  Data type: character string

  Selection: 'cylindrical'/'rectangular' – cylindrical/rectanglar shape

  Default value: 'cylindrical'

- *type*

  Description: the type of the cylindrical pin cell

  Data type: character string

  Selection: ’full’/’half1’/’half2’/’half3’/’half4’/’quad1’/’quad2’/’quad3’/’quad4’ – meaning of these selections is showed in Fig. 10.

  Default value: ’full’

  Note: this attribute only works for the cylindrical shape. It will be ignored for the rectangular shape.

- *name*

  Description: the name of the pin cell

  Data type: character string

  Note: this name is only used to help managing the input.

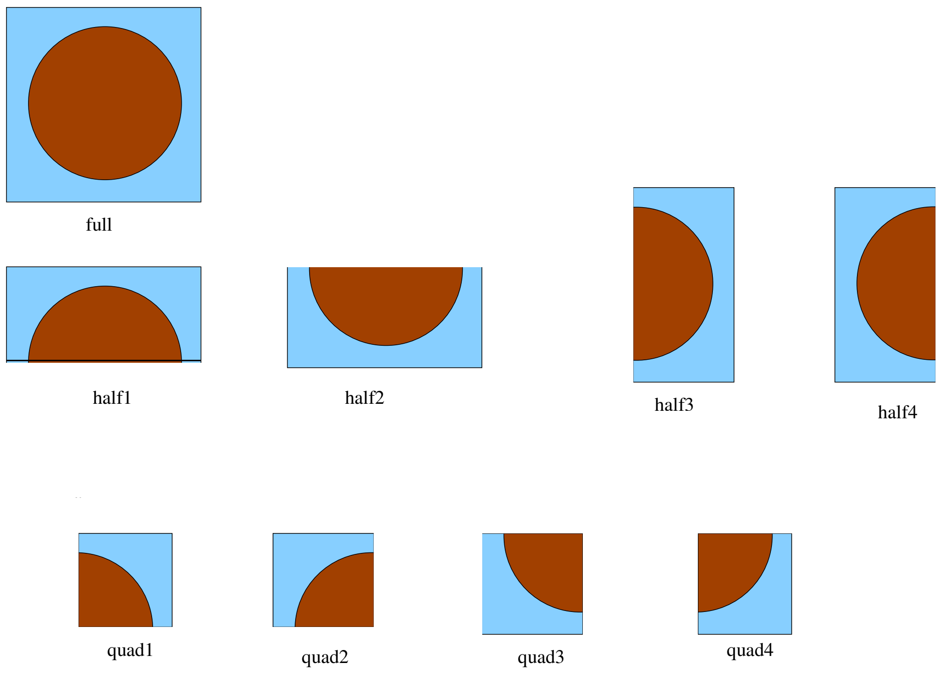


Figure 10: Types of cylindrical pin cell.

Different shapes of pin cells have different sets of sub-elements.

1. Cylindrical pin

- *NRad*

  Description: number of annular regions

  Data type: integer

  Default value: the size of Radius

- *Radius*

  Description: outer radius of the annular regions (cm)

  Data type: real array

  Default value: N/A

  Note: radii should be entered in the increasing order, moving outward. The final entry is half the pin pitch. This element must be specified.

- *MaterialID*

  Description: material IDs of annular regions

  Data type: integer array

  Default value: all 1

  Note: the final entry is for the coolant region surrouinded by a square. Size of material IDs is the same as the size of Radius.

- *NSides*

  Description: number of even-distributed sides used to approximate circles

  Data type: integer array

  Default value: all 12

  Note: Size of this array is equal to NRad minus one. For a full cell, the default angule of one side is $2\pi/12$. For a half cell, the default angule of one side is $2\pi/24$. For a quarter cell, the default angule of one side is $2\pi/48$. INSTANT uses the polygon to approximate circles while keeping the area unchanged for each region.

- *Rotation*

  Description: the angle of a point in degree of all circles with respect to x-coordinate

  Data type: real array

  Default value: all zero

  Note: Size of this array is equal to NRad minus one. for half or quarter cells, this element will be ignored. The default values for 'half1', 'half2' and all quarter cells are zero; while the values for 'half3' and 'half4' are 90 degree.

- *HomogenizedXS*

  Description: pin homogenized material ID

  Data type: integer

  Default value: 0 (not homogenized)

  Note: the homogenized material ID must be specified when pin-homogenized calculations are conducted.

2. Rectangular pin

    - *NX*
      Description: number of intervals in x-direction
      Data type: integer
      Default value: 1
    - *NY*
      Description: number of intervals in y-direction
      Data type: integer
      Default value: 1
    - *IX*
      Description: number of cuts of all intervals in x-direction
      Data type: integer array
      Default value: all 1
      Note: the size of this array must be equal to NX.
    - *IY*
      Description: number of cuts of all intervals in y-direction
      Data type: integer array
      Default value: all 1
      Note: the size of this array must be equal to NY.
    - *DX*
      Description: length of all intervals in x-direction (cm)
      Data type: real array
      Default value: all 1.0 cm
      Note: the size of this array must be equal to NX.
    - *DY*
      Description: length of all intervals in y-direction (cm)
      Data type: real array
      Default value: all 1.0 cm
      Note: the size of this array must be equal to NY.
    - *MaterialID*
      Description: material IDs of all grids of the rectangular pin
      Data type: integer array
      Default value: all 1
      Note: the size of this array must be equal to NX times NY.

- *HomogenizedXS*
  Description: pin homogenized material ID
  Data type: integer
  Default value: 0 (not homogenized)
  Note: the homogenized material ID must be specified when pin-homogenized calculations are conducted.

5.3.3.3 *Assemblies*

This block collects all the descriptions of assemblies with the tag name *Assembly*. This block must present to specify a LWR geometry.

An assembly has the following two attributes:

- *ID*
  Description: the unique assembly ID (> 0)
  Data type: integer
  Default value: the ordering among all assemblies
  Note: it is recommanded to explicitly give the assembly ID.
- *name*
  Description: the name of the assembly
  Data type: character string
  Note: this name is only used to help managing the input.

For each assembly, there are following sub-elements:

- *NXPin*
  Description: the number of pins in x-direction
  Data type: integer
  Default value: 1
- *NYPin*
  Description: the number of pins in y-direction
  Data type: integer
  Default value: 1

- *PinArrangement*

  Description: arrangement of pins in the assembly

  Data type: integer array

  Default value: all 1

  Note: the size of this array must be equal to NXPin times NYPin. Pin IDs should be entered in x-direction first then in y-direction in the increasing order. All pins in a row must have the same size in y-direction and all pins in a column must have the same size in x-direction. If a pin used by an assembly and the assembly is used by a core configuration and full heterogeneous calculation is pursued, the corresponding pin must present.

- *XT*

  Description: the width of the assembly (cm)

  Data type: real

  Default value: the summation of sizes of all pins in a row

  Note: if this element is specified, it must be equal to the summation of sizes of all pins in a row if the pin arrangement is specified. This redudant information helps to detect possible input errors.

- *YT*

  Description: the height of the assembly (cm)

  Data type: real

  Default value: the summation of sizes of all pins in a column

  Note: if this element is specified, it must be equal to the summation of sizes of all pins in a column if the pin arrangement is specified. This redudant information helps to detect possible input errors.

- *HomogenizedXS*

  Description: assembly homogenized material ID

  Data type: integer

  Default value: 0 (not homogenized)

  Note: the homogenized material ID must be specified when assembly-homogenized calculations are conducted. When an assembly is used only as a homogenized assembly, the detailed pin arrangement, NXpin, NYpin and PinArrangement, do not have to be inputted. If this is the case, The width XT and YT must be specified.

5.3.3.4 *Core*

This block describe the composed assemblies of the core and the structured arrangement in z-direction. This block must present to specify a LWR geometry.

A core may have a name as its attribute:

- *name*

  Description: the name of the core

  Data type: character string with maximum length 80

  Default value: 'testcore'

  Note: this name will be used as the prefix of debug file names.

A core has following sub-elements:

- *NX*

  Description: the number of assemblies in x-direction

  Data type: integer

  Default value: 1

- *NY*

  Description: the number of assemblies in y-direction

  Data type: integer

  Default value: 1

- *NZ*

  Description: number of layers in z-direction

  Data type: integer

  Default value: 1

- *IZ*

  Description: number of cuts of all layers in z-direction

  Data type: integer array

  Default value: all 1

  Note: the size of this array must be equal to NZ.

- *DZ*

  Description: height of all layers in z-direction (cm)

  Data type: real array

  Default value: all 1.0 cm

  Note: the size of this array must be equal to NZ.

- *BC*

  Description: boundary condition flags

  Data type: integer array

  Default value: all 1 (reflecting)

  Note: the size of this array must be equal to 4 or 6. Its entries indicate the boundary conditions in $\mp x, \mp y, \mp z$ boundaries. So far ONLY 0 and 1 can be used for vacuum and reflecting boundary conditions respectively. For 2D calculations, only the first four entries are used. For 3D calculations, if the $\mp z$ indicators are missing, reflecting boundary will be applied.

- *Layout*

  Description: assemblies arrangement

  Data type: integer array

  Default value: all 1

  Note: the size of this array must be equal to NX times NY times NZ. Assembly IDs should be entered in x-direction first then in y-direction and z-direction in the increasing order.

- *ZMark2D*

  Description: the layer used to generate the unstructured 2D mesh or used for 2D calculations

  Data type: integer

  Default value: 1

  Note: This element make sure that user can describe the full 3D LWR geometry even he or she just want to do 2D calculations of one layer of the full 3D geometry. This element also acts as an indicatior to which layer the x-y plane unstructured mesh will be generated. To be able to act as a layer indicator for 3D calculations, all assemblies in this layer must have the minimum homogenization, i.e., homogenization indicator Homogenization of an assembly in this layer must be smallest in all assemblies in the same location of all layers.

- *Homogenization*

  Description: homogenization specification of all assemblies

  Data type: integer array

  Default value: all 0 (full heterogeneous calculation for all assemblies)

Note: the size of this array must be equal to NX times NY times NZ. Homogenization specification should be entered in x-direction first then in y-direction and z-direction in the increasing order. Each value could be 0/1/2 - full heterogeneous/pin homogenization/assembly homogenization.

- *ProcessorID*

  Description: processor allocations of all assemblies

  Data type: integer array

  Default value: all 0 (all assemblies are calculated on one single processor, i.e. no parallelization)

  Note: the size of this array must be equal to 2 times NX times NY times NZ. Each pair of numbers tells the starting and ending processors of the corresponding assembly. All computational cells in the assembly are evenly distributed into these processors. Processors are numbered from 0. These processor allocations are ignored for the serial version of INSTANT.

5.3.4 Unstructured geometry with PSLG

This block describes a geometry which is unstructured in x-y plan with PSLG and is regular in z-direction. An example mesh is showed in Fig. 11.

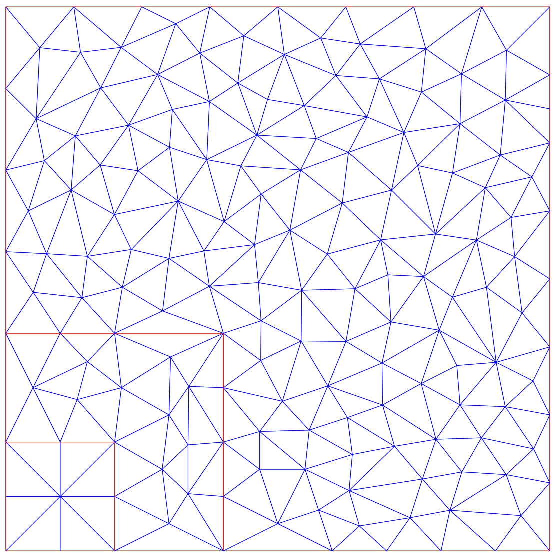


Figure 11: An example unstructured 2D mesh generated with PSLG.

A Planar Straight Line Graph (PSLG) is a collection of vertices and segments. In this manual, A PSLG is always a 2D graph. Segments are edges whose endpoints are vertices in the PSLG, and whose presence in any mesh generated from the PSLG is enforced. A geometry described with PSLG will be further triangulated by calling Triangle, a open-source two-dimensional quality mesh generator and delaunay triangulator developed by Jonathan R. Shewchuk.

5.3.4.1 *TriangulationControls*

Elements contained in this element control how the PSLG be triangulated. If this element is missing, the default values of all child elements will be accepted.

1. *MaxArea*

   Description: the maximum area allowed for all triangles in the x-y plane in cm^2

   Data type: real

   Default value: N/A

   Note: By default, there is no area constraint on the triangulation.

2. *MinAngle*

   Description: the minimum angle in degree allowed for all triangles in the x-y plane

   Data type: real

   Default value: N/A

   Note: By default, there is no angle constraint on the triangulation.

3. *Prerefine*

   Description: number of globally uniform refinement (every cell is cut into 4 in 2D or 8 in 3D equal child cells) before calculation

   Data type: integer

   Default value: 0 (no prerefinement)

4. *name*

   Description: name of the PSLG

   Data type: character array with length 80

   Default value: 'PSLG'

5. *DebugOutput*

   Description: if the triangulation debugging files are generated

   Data type: logical

   Default value: false

   Note: if this flag is turned on, several triangulation debugging files will be generated. Meanings of these files can be found in LWR section.

5.3.4.2 *Points*

This element describes all points in the PSLG. This element must present.

1. *N*

   Description: the number of points in the PSLG

   Data type: integer

   Default value: number of entries in point Coordinates divided by 2

   Note: Users are encouraged to specify it to make sure the point coordinates are inputted as desired.

2. *Coordinates*

   Description: the x and y coordinates of all points in the PSLG in cm

   Data type: real array

   Default value: N/A

   Note: Coordinates are inputted point by point. Each point has two values corresponding to its x and y coordinates. If N_{point} is specified, the number of entries must be equal to $2N_{point}$.

5.3.4.3 *Segments*

This element describes all segments in the PSLG. This element must present. Segments are edges whose presence in the triangulation is enforced (although each segment may be subdivided into smaller edges).

1. *N*

   Description: the number of segments in the PSLG

   Data type: integer

   Default value: number of entries in segment EndPoints divided by 2

   Note: Users are encouraged to specify it to make sure the end points of all segments are inputted as desired.

2. *EndPoints*

   Description: the indices of two end points of all segments in the PSLG

   Data type: integer array

   Default value: N/A

Note: Points are numbered starting from 1. The index of a point is determined by their sequence in point Coordinates. The order of two end points for any segment is irrelevant. End points are inputted segment by segment. Each segment has two values corresponding to its two end points. If $N_{segment}$ is specified, the number of entries must be equal to $2N_{segment}$.

3. *BoundaryMarkers*

   Description: the boundary markers of all segments

   Data type: integer array

   Default value: zero for all boundary segments

   Note: 0/1 means vacuum/reflecting boundaries. Although values of all interior segments are irrelevant, users are encouraged to specify them with -1 to indicate these segments are located inside of the solution domain. The number of entries must be equal to the number of segments.

5.3.4.4 *Holes*

The element lists holes in the triangulation. If this element is missing, there are no holes in the PSLG.

(Cited from Triangle:) Holes are specified by identifying a point inside each hole. After the triangulation is formed, Triangle creates holes by eating triangles, spreading out from each hole point until its progress is blocked by PSLG segments. You must be careful to enclose each hole in segments, or your whole triangulation might be eaten away. If the two triangles abutting a segment are eaten, the segment itself is also eaten. Do not place a hole directly on a segment; if you do, Triangle will choose one side of the segment arbitrarily.

1. *N*

   Description: the number of holes in the PSLG

   Data type: integer

   Default value: number of entries in InteriorPoints of holes divided by 2

   Note: Users are encouraged to specify it to make sure the interior points of all holes are inputted as desired.

2. *InteriorPoints*

   Description: the coordinates of interior points of all holes in the PSLG in cm

   Data type: real array

   Default value: N/A

Note: The interior point can be choose arbitrarily unless it is located inside of the hole. Interior points are inputted hole by hole. Each hole has two values corresponding to x and y coordinates of its interior points. If N_{hole} is specified, the number of entries must be equal to $2N_{hole}$.

5.3.4.5 *Regions*

The element lists regions in the triangulation. If this element is missing, there is only one region covering the entire solution domain. The material ID will be one; there will be no external source; only one processor will be used; there will be only one sub-domain.

1. *N*

   Description: the number of regions in the PSLG

   Data type: integer

   Default value: number of entries in InteriorPoints of regions divided by 2

   Note: Users are encouraged to specify it to make sure the interior points of all regions are inputted as desired.

2. *InteriorPoints*

   Description: the coordinates of interior points of all regions in the PSLG in cm

   Data type: real array

   Default value: N/A

   Note: The interior point can be choose arbitrarily unless it is located inside of the region. Interior points are inputted region by region. Each region has two values corresponding to x and y coordinates of its interior points. If N_{region} is specified, the number of entries must be equal to $2N_{region}$.

3. *MaterialID*

   Description: the material ID of all regions in the PSLG

   Data type: integer array

   Default value: all 1

   Note: Material IDs are inputted region by region (then layer by layer in 3D). The number of entries must be equal to NZ times N_{region}.

4. *SourceID*

   Description: the external source ID of all regions in the PSLG

   Data type: integer array

Default value: all 0

Note: Source IDs are inputted region by region (then layer by layer in 3D). The number of entries must be equal to NZ times N_{region}.

5. *ProcessorID*

   Description: the processor ID of all regions in the PSLG

   Data type: integer array

   Default value: all 0

   Note: Processor IDs are inputted region by region (then layer by layer in 3D). The number of entries must be equal to NZ times N_{region}. They are ignored for the serial version of INSTANT.

6. *SubDomainID*

   Description: the sub-domain ID of all regions in the PSLG

   Data type: integer array

   Default value: all 1

   Note: Sub-domain IDs are inputted region by region (then layer by layer in 3D). The number of entries must be equal to NZ times N_{region}.

5.3.4.6 *ZDirection*

The element describe the structure of layers in 3D. For 2D calculations, this element will just be ignored if it presents.

1. *NZ*

   Description: the number of layers in z-direction

   Data type: integer

   Default value: 1

2. *IZ*

   Description: the subdivisions of all layers

   Data type: integer array

   Default value: all 1 (no subdivision)

   Note: Length of this array must be equal to NZ.

3. *DZ*

   Description: the height of all layers (cm)

   Data type: real array

   Default value: all 1 cm

   Note: Length of this array must be equal to NZ.

4. *ZBC*

   Description: the boundary condition of $\mp z$-direction

   Data type: integer array

   Default value: all 1 (reflecting)

   Note: Length of this array must be equal to 2. Supported boundary types can be found in Section 5.3.1.12.

5.3.5 EXODUS geometry

This block describes a geometry which is detailed by an EXODUS-II file. EXODUS-II format allows a larger set of types of cells. When the cell type is not supported by INSTANT, INSTANT will error out with messages. For complete description of EXODUS-II format, ”EXODUS II: A Finite Element Data Model” by Larry A. Schoof, et. al. is referred.

5.3.5.1 *ExodusFile*

Description: the name of the EXODUS-II file

Data type: word

Default value: N/A

Note: user must provide the full name of the file. The file is located in the directory relative to the input directory unless the file name contains the absolute path.

5.3.5.2 *Blocks*

All elements in a EXODUS-II file are arranged in blocks. INSTANT needs material, processor, source and subdomain IDs be assigned to these blocks for the calculation. So this element must present in the input file. No default values are available.

1. *IDs*

   Description: the IDs of blocks

   Data type: integer array

   Default value: N/A

   Note: users are supposed to know what block IDs are contained in the EXODUS-II file. Users do not have to give the full list of the block IDs. If this is the case, the unspecified blocks will assume default material ID 1, default processor ID 0, default subdomain ID 1 and default source ID 0. However, it is highly recommended that users specify all block IDs here.

2. *MaterialID*

   Description: the material IDs of all specified blocks

   Data type: integer array

   Default value: all 1

   Note: if this element is given, number of material IDs in it must be equal to the number of block IDs in IDs.

3. *ProcessorID*

   Description: the processor IDs of all specified blocks

   Data type: integer array

   Default value: all 0

   Note: if this element is given, number of processor IDs in it must be equal to the number of block IDs in IDs.

4. *SubDomainID*

   Description: the subdomain IDs of all specified blocks

   Data type: integer array

   Default value: all 1

   Note: if this element is given, number of subdomain IDs in it must be equal to the number of block IDs in IDs.

5. *SourceID*

   Description: the source IDs of all specified blocks

   Data type: integer array

   Default value: all 1

   Note: if this element is given, number of source IDs in it must be equal to the number of block IDs in IDs.

5.3.5.3 *SideSets*

Selected sides are arranged in a list of side sets in a EXODUS-II file. INSTANT needs to use this side sets to assign the proper boundary conditions and possibly surface sources IDs for the calculation. So this element must present in the input file. No default values is available.

1. *IDs*

   Description: the IDs of side sets

   Data type: integer array

   Default value: N/A

   Note: users are supposed to know what side sets are contained in the EXODUS-II file. Users do not have to give the full list of the side set IDs. If this is the case, the unspecified side sets will assume default boundary condition (vacuum) and default source ID (0 no source). However, it is highly recommended that users specify all side set IDs here.

2. *BCs*

   Description: the boundary condition of side sets

   Data type: integer array

   Default value: all 0

   Note: if this element is given, number of material IDs in it must be equal to the number of side set IDs in IDs. Only vacuum (0) and reflecting (1) boundary condition can be assigned.

5.4 *Mesh*

Parameters in this block control how the computational mesh is formed.

5.4.1 *multimesh*

Description: to determine how meshes to be assigned to all energy groups (different energy groups may have different meshes)

Data type: integer

Default value: 0

Selection: 0/1/2/ – all groups use the same mesh/all groups use their own mesh/user-defined assignment

Note: when multimesh is set 2, M2G data block must be presented.

5.4.2 *M2G*

Description: a data block to assign mesh IDs to all energy groups

- *NG*

  Number of energy groups. This integer number must be equal to the number of groups in the Materials section.

- *NM*

  Number of meshes.

- *MeshID*

  An integer array with size of number of energy groups. Elements in this array are the mesh IDs of corresponding energy groups. The maximum value of this array must be less than or equal to NM. The minimum value of this array must be greater than 0.

5.4.3 *pn*

Description: PN order

Data type: integer

Default value: 3

Note: only odd number is allowed. The maximum PN order the code supports now is 33. Refer to Section 5.8 for how to choose the PN order.

5.4.4 *pint*

Description: spatial interior expansion order of a cell

Data type: integer

Default value: 4

Note: Refer to Section 5.8 for how to choose the interior expansion order.

5.4.5 *psurf*

Description: spatial interface expansion order

Data type: integer

Default value: 1

Note: Refer to Section 5.8 for how to choose the interface expansion order.

5.4.6 *psour*

Description: spatial source expansion order

Data type: integer

Default value: equal to pint

Note: Refer to Section 5.8 for how to choose the source expansion order.

5.4.7 *geom_filename*

Description: the name of the file which contains the geometry data

Data type: word

Default value: ’inp.xml’

Note: user can have a different file containing the geometry description. Only the XML format of geometry data is allowed. The file is located in the directory relative to the input directory unless the file name contains the absolute path.

5.5 *Materials*

The Materials XML block is inputted in case of **INSTANT standalone calculations only**, where a set of mixture Macroscopic cross sections are expected. For any other executable and analysis flow of PHISICS, the Materials and, consequentially, neutron cross sections are inputted in microscopic/mixed form and, therefore, the **`<Materials>`** XML node is omitted.

5.5.1 *mat_filename*

Description: the name of the file which contains the material data

Data type: word

Default value: 'inp.xml'

Note: user can have a different file containing the material description. Now only the XML format of material data is supported. The file is located in the directory relative to the input directory unless the file name contains the absolute path.

5.5.2 *Macros*

All first type of materials are contained in this block.

Attribute: *NG* – the number of energy groups. Default value is 1.

5.5.2.1 *material*

Attributes:

- *ID* – the unique material ID (> 0)
- *NA* – the order of scattering anisotropy (≥ 0) (0 means isotropic scattering)
- *fissile* – true/false - a fissile/not a fissile material

Elements:

- *TotalXS* – total cross sections of all groups $\sigma_{t,g}$. The unit is $1/cm$. Every material must have the total cross sections.
- *NuFissionXS* – fission cross sections times the averaged neutrons emitted per fission $\nu\sigma_{f,g}$. The unit is $1/cm$. Only fissile materials have this cross section.
- *FissionXS* – fission cross sections $\sigma_{f,g}$. The unit is $1/cm$. Only fissile materials have this cross section. Note that the fission cross section $\sigma_{f,g}$ is not required by INSTANT for calculation but only used for post-processing to generate the fission rate.
- *KappaXS* – energy deposited per fission times the fission cross sections $\kappa\sigma_{f,g}$. The unit is J/cm. Only fissile materials have this cross section. Note that the Kappa cross section $\kappa\sigma_{f,g}$ is not required by INSTANT for calculation but only used for post-processing.
- *ChiXS* – fission spectrum χ_g. Only fissile materials have this cross section. The summation of fission neutron yield of all energy groups should be equal to one, otherwise, INSTANT will normalize the fission spectrum before it is being used.

- *Profile* – profile of the scattering matrix $\sigma_{s,n}^{g'\to g}, g' = 1, \cdots, NG; g = 1, \cdots, NG; n = 0, \cdots, NA$. *Profile* is designed to save the input effort and the size of the file containing the scattering cross sections. There are $NG \times (NA+1)$ pairs of integers. All these pairs are ordered by energy index g first then anisotropy group n, i.e. $((g = 1, G), n = 0, NA)$. Each pair has the first departure scattering group and the last departure scattering group for the anisotropy n and the group g. *Profile* is used to reduce the amount of inputs for the scattering matrix. Users do not have to give *Profile*. If *Profile* is absent, all entries of the scattering matrix must be inputted including all zero entries. This also means the default value of all pairs of *Profile* is $(1, NG)$. For example, the following 7-by-7 scattering matrix where non-zeros are marked with $\times$,

$$\begin{array}{cc} & \rightarrow g' \\ \downarrow_g & \begin{bmatrix} \times & & & & & & \\ \times & \times & & & & & \\ \times & \times & \times & & & & \\ & & \times & \times & \times & & \\ & & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times & \times \end{bmatrix}_{7\times 7} \end{array}$$

  has *Profile* $[1 \quad 1 \qquad 1 \quad 2 \qquad 1 \quad 3 \qquad 3 \quad 5 \qquad 3 \quad 7 \qquad 3 \quad 7 \qquad 3 \quad 7]$.

- *ScatteringXS* – the $NA+1$ scattering matrix $\sigma_{s,n}^{g'\to g}$. The unit is $1/cm$. All scattering matrices are inputted sequentially. For each scattering matrix, entries specified by *Profile* are inputted row by row. The column index of the scattering matrix is g'; the row index of the scattering matrix is g. If there is no up-scatterings, the scattering matrix is strictly low-triangular. Note that INSTANT treats all groups without up-scatterings as the "fast" group and all groups with up-scatterings as the "thermal" group. "fast" here does not necessarily mean the energy of neutrons in the group is high. Thermal iteration may be required for solving problems with up-scattering materials. Every material must have the scattering cross sections.

Note on reproducing the diffusion results with the P1 calculation:

In the diffusion equation, the material data required are the diffusion coefficient D_g, the removal cross section $\sigma_{r,g}$ and the scattering matrix $\sigma_{s,0}^{g'\to g}, (g' \neq g)$. The scattering must be isotropic because of the limitation of the diffusion theory. These material data set is different from PN transport equation, where the total cross section $\sigma_{t,g}$ and the scattering matrix $\sigma_{s,n}^{g'\to g}$, including the in-group scattering cross sections and all higher scattering moments, are required. So to reproduce diffusion results with P_1 transport, Users need to do,

$$\sigma_{t,g} = \frac{1}{3D_g} \tag{2}$$

$$\sigma_{s,0}^{g\to g} = \frac{1}{3D_g} - \sigma_{r,g} \tag{3}$$

$$\sigma_{s,0}^{g'\to g} = \sigma_{s,0}^{g'\to g}, (g' \neq g) \tag{4}$$

$$\sigma_{s,1}^{g'\to g} = 0 \tag{5}$$

The total cross section calculated from the diffusion coefficient may not equal to the real total cross section, neither the in-group scattering cross section. Note that

$$\sigma_{r,g} = \sigma_{a,g} + \sum_{\substack{g'=1 \\ g'\neq g}}^{G} \sigma_{s,0}^{g\to g'}, \tag{6}$$

so instead of having the removal cross section in the diffusion equation, we may have the absorption cross section $\sigma_{a,g}$. In this case the in-group scattering cross section is calculated as,

$$\sigma_{s,0}^{g\to g} = \frac{1}{3D_g} - \sigma_{a,g} - \sum_{\substack{g'=1 \\ g'\neq g}}^{G} \sigma_{s,0}^{g\to g'} \tag{7}$$

5.6 *External_Sources*

This block is used to describe the external sources. Now only the volumetric source is supported.

5.6.1 *vsrc_filename*

Description: the name of the file which contains the volumetric source element

Data type: word

Default value: 'inp.xml'

Note: user can have a different file containing the volumetric source description. The file is located in the directory relative to the input directory unless the file name contains the absolute path.

5.6.2 *Volumetric_Sources*

Attribute: *n_energy_groups* – the number of energy groups. Default value is 1. This number must equal to the number of group *NG* in the Materials block.

5.6.2.1 *VSource*

Attributes:

- *ID* – the unique volumetric source ID
- *primal* – true/false - primal/adjoint source

The source for the multigroup transport equation is

$$Q_g(\boldsymbol{r},\boldsymbol{\Omega}) = \left\{ \begin{array}{ll} \sum_{n=0}^{N}\sum_{k=0}^{n} s_{n,k}^{g}(\boldsymbol{r})Y(\boldsymbol{\Omega}), & 2D \\ \sum_{n=0}^{N}\sum_{k=-n}^{n} s_{n,k}^{g}(\boldsymbol{r})Y(\boldsymbol{\Omega}), & 3D \end{array} \right. \tag{8}$$

Multiple *Item* elements are contained in VSource. Each *Item* is for one spherical harmonics moment. The value of the *Item* is its strength in unit of $1/cm^3$. Its attributes describe which spherical harmonics moments $s_{n,k}^{g}$ this *Item* is for. If a particular moment is not specified with *Item*, its strength is zero. Index k can not be less than zero in 2D calculations.

- *moment_n* – n
- *moment_k* – k
- *group* – group ID

Note: only source constant in space is supported.

5.7 *Iterative_Solver*

This block contains all the control parameters for the transport solvers.

5.7.1 *oacc_typ*

Description: the type of acceleration for the power iteration

Data type: integer

Default value: 0

Selection: 0/1 – no acceleration/Chebyshev acceleration

Note: Only valid for eigenvalue problem.

5.7.2 *tacc_typ*

Description: the type of acceleration for the thermal iteration

Data type: integer

Default value: 0

Selection: 0 – no acceleration

Note: no thermal re-balance is performed. Only valid with up-scatterings.

5.7.3 *epskeff*

Description: ϵ_{keff} – convergence tolerance on k-effective

Data type: real

Default value: 1d-8

Note: Only valid for eigenvalue problem. Power iteration may stop when

$$\frac{\Delta_{keff}^{(n)}}{k_{eff}^{(n)}} = \frac{\left|k_{eff}^{(n)} - k_{eff}^{(n-1)}\right|}{k_{eff}^{(n)}} < \epsilon_{keff}, \tag{9}$$

where n is the iteration index.

5.7.4 *epsflux*

Description: ϵ_{flux} – convergence tolerance on the scalar flux

Data type: real

Default value: 1d-6

Note: Only valid for eigenvalue problem. Power iteration may stop when

$$\Delta_{flux}^{(n)} = max_{1\leq g\leq G, K\in\mathcal{T}} \frac{\left\|\Phi_{g,K}^{(n)}(\boldsymbol{r}) - \Phi_{g,K}^{(n-1)}(\boldsymbol{r})\right\|_2}{\left\|\Phi_{g,K}^{(n)}(\boldsymbol{r})\right\|_2} < \epsilon_{flux}. \tag{10}$$

where G is the number of energy groups; $\mathcal{T}$ is the mesh and K is one cell on the mesh.

5.7.5 *maxouter*

Description: N_{power} – the maximum number of power iterations

Data type: integer

Default value: 2000

Note: Only valid for eigenvalue problem.

5.7.6 *epsthml*

Description: ϵ_{thml} – convergence tolerance on the scalar flux for the thermal iteration

Data type: real

Default value: 1d-6

Note: Only valid with up-scatterings. Thermal iteration may stop when

$$\Delta_{thml}^{(m)} = max_{nfg+1\leq g\leq G} \frac{\left\|\Phi_{g}^{(m)}(\boldsymbol{r}) - \Phi_{g}^{(m-1)}(\boldsymbol{r})\right\|_2}{\left\|\Phi_{g}^{(m)}(\boldsymbol{r})\right\|_2} < \epsilon_{thml}, \tag{11}$$

where nfg is the number of energy groups without up-scatterings; m is the thermal iteration index. Note that the power iteration index is dropped in the above equation.

5.7.7 *facthml*

Description: c_{thml} – a factor to the power iteration error on controlling the convergence on the scalar flux for the thermal iteration

Data type: real

Default value: 0.002

Note: Only valid for eigenvalue problems. Thermal iteration may stop when

$$\Delta_{thml}^{(m)} < c_{thml}\Delta_{flux}. \tag{12}$$

Note that the power iteration index is dropped in the above equation.

5.7.8 *maxthml*

Description: N_{thml} – the maximum number of thermal iterations per power iteration

Data type: integer

Default value: 10

Note: Only valid with up-scatterings.

5.7.9 *keff0*

Description: initial guess of the k-effective for the power iteration

Data type: real

Default value: 1.0

Note: Only valid for eigenvalue problem.

5.7.10 *echo*

Description: to control the screen output of the multigroup solver

Data type: integer

Default value: 1

Note: the larger this number is, the more outputs you will see on the screen during the solving. 0 means that the multigroup solver is silenced. There is no screen output.

5.7.11 *PNinner*

Description: to choose the iterative solvers for the one-group source problem (the inner solve)

Data type: integer

Default value: 0

Selection: 0/1/2 – red-black iteration algorithm/CG/GMRes

Note: the red-black iteration algorithm is recommended because it is highly optimized currently.

5.7.12 *iacc_typ*

Description: to control the inner acceleration

Data type: integer

Default value: 0

Selection: 0/1 – no acceleration/preconditioned with the lower-order red-black iteration

Note: When red-black iteration algorithm is selected as the inner solver, setting iacc_typ to 1 means turn the partitioned algorithm on. For the Krylov solvers, it determines whether the lower-order red-black preconditioner will be used or not.

5.7.13 *epsinner*

Description: τ_{inner} convergence tolerance for the inner iteration

Data type: real

Default value: 1d-7

Note: this tolerance applies to different inner iterative solvers, such as the red-black iteration algorithm or the matrix-free Krylov solvers.

5.7.14 *facinner*

Description: c_{inner} – a factor to the power iteration error on controlling the convergence of the inner iteration

Data type: real

Default value: 0.001

Note: Only valid for eigenvalue problems.

$$\Delta_{inner} < c_{inner}\Delta_{flux}. \tag{13}$$

Note that the power iteration index is dropped in the above equation.

5.7.15 *maxinner*

Description: the maximum number of red-black iterations for a one-group source problem (an inner solve)

Data type: integer

Default value: 1000

5.7.16 *omegaon*

Description: true to turn the ω-acceleration on for the red-black iteration

Data type: logical

Default value: false

5.7.17 *maxcg*

Description: the maximum number of CG iterations for a one-group source problem (an inner solve)

Data type: integer

Default value: 1000

5.7.18 *maxgmres*

Description: the maximum number of GMRes iterations for a one-group source problem (an inner solve)

Data type: integer

Default value: 1000

5.7.19 *lrestart*

Description: the restarting number for GMRes iterations

Data type: integer

Default value: 20

5.8 Spatial and angular discretization

The code uses the hybrid FEM for the spatial discretization and the PN method for the angular discretization. With the hybrid FEM, the even flux moments including the scalar flux on each cell in the solution domain are expanded with the interior shape functions. The number of the shape functions of a cell is also called the number of degrees of freedom of the cell. We use polynomials as the shape function. The expansion order uniquely determines the number of degrees of freedom of a cell. On the other hand, with the hybrid FEM, the odd flux moments on cell interfaces are expanded with the face shape functions. We also use polynomials as the interface shape functions. The interface expansion order uniquely determines the number of degrees of freedom of an face. If a uniform mesh is used, the number of degrees of freedom of all cells are the same and the number of degrees of freedom of all faces are also the same. For angular discretization, a PN order is set for each cell and each face. If the mesh is uniform, PN orders of all cells and all faces are the same.

So we only need three parameters to describe the discretization: *the interior expansion order* p_{int}*, the interface expansion order* p_{surf} *and the PN order* p_n.

On each cell the number of degrees of freedom of the cell must be at least greater than or equal to the number of degrees of freedom of all its surrounding faces. Moreover, in order to make sure the convergence of the iterative solver, the number of degrees of freedom of each cell has to be higher than the allowed minimum value. The spatial discretization interferes with the angular discretization in the transport calculation. For the Cartesian and triangular geometry, the good combinations of the interior and interface expansion order are $p_{int} \geq p_{surf} + 2$ in 2D and $p_{int} \geq p_{surf} + 3$ in 3D. For the hexagonal geometry, the good combinations of the interior and interface expansion order are $p_{int} \geq p_{surf} + 3$ in 2D and $p_{int} \geq p_{surf} + 5$ in 3D. For a given PN order, the number of degrees of freedom of a face must be greater than a value to make sure the numerical solution is physical. With $p_n = 1$, the minimum interface expansion order is 0, i.e., the minimum number of degrees of freedom of an face is 1. With $p_n > 1$, the minimum interface expansion order is 1 for Cartesian and hexagonal geometries, i.e., the minimum number of degrees of freedom of an face is 2. With $p_n > 1$, the minimum interface expansion order is 2 for triangular geometries, i.e., the minimum number of degrees of freedom of an face is 3.

There is another parameter controlling the spatial expansion of the in-group source: the source expansion order p_{sour}. This parameter can be from 0 to the interior expansion order. Lower source expansion order requires smaller computing effort by sacrificing the accuracy of discretization. Usually the decrease of accuracy with a lower source expansion order is small with a great reduce of CPU time.

6 INSTANT_TIME_DEP

In the **<INSTANT_TIME_DEP>** XML input/main block is the main node to input the settings to activate the driver for neutron time-dependent transport calculation (i.e. delayed neutron source terms). An example of the this block is reported below:

```
<INSTANT_TIME_DEP>
    <NDelFam> 6 </NDelFam>
    <k_real> 1.0 </k_real>
    <lambda> 0.0124 0.0305 0.111 0.301 1.14 3.01</lambda>
    <!-- NDT_array: number of time steps at a certain DT -->
    <NDT_array>2 5 3</NDT_array>
    <!-- DT_array: the time step size for each of the grid
       points identified by NDT_array -->
    <DT_array> 0.01 0.1 0.001</DT_array>
    <!-- output_ts_array: output control. The output is going to
       be printed
        at the listed time steps (in this case we output the
           solutions at TS=1 and TS=10) -->
    <output_ts_array>1 10</output_ts_array>
    <NVelo> 1.73783e+7 3.85905e+5 </NVelo>
</INSTANT_TIME_DEP>
```

As shown in the examples above, multiple blocks can be inputted. Each of these blocks are explained in dedicated sections in the following.

6.1 *k_real*

Block name: **<k_real>**

Description: The real $k - eff$ that will be used for the normalization of the fission source.

Data type : float

Default value : 1.0

6.2 *NVelo*

Block name: **<NVelo>**

Description: The neutron velocities for each energy group.

Data type : float array (number of energy groups)

Required Entry

6.3 *NDelFam*

Block name: **`<NDelFam>`**

Description: The number of delayed neutron families.

Data type : integer

Required Entry

6.4 *lambda*

Block name: **`<lambda>`**

Description: The decay constants for each delayed neutron family.

Data type : float array

Required Entry

6.5 *NDT*

Block name: **`<NDT>`**

Description: Number of time steps.

Data type : integer

Required Entry

Note: This node is mutually exclusive with respect to the **`<NDT_array>`**. Only one of the two can be inputted.

6.6 *DT*

Block name: `<DT>`

Description: The time step size.

Data type : float

Required Entry

Note: This node is mutually exclusive with respect to the `<DT_array>`. Only one of the two can be inputted.

6.7 *NDT_array*

Block name: `<NDT_array>`

Description: Number of time steps at a certain DT

Data type : integer array

Required Entry

Note: This node is mutually exclusive with respect to the `<NDT>`. Only one of the two can be inputted.

6.8 *DT_array*

Block name: `<DT_array>`

Description: The time step size for each of the grid points identified by NDT_array

Data type : float array

Required Entry

6.9 *output_ts_array*

Block name: `<output_ts_array>`

Description: output control. The output is going to be printed at the listed time steps. For example, if `<output_ts_array>`1 10 `<output_ts_array>`, we output the solutions at TS=1 and TS=10.

Data type : integer array

Required Entry

7 DENSITIES

In the **<DENSITIES>** XML input/main block, all the initial compositions of the different zones in the mesh (geometry) can be defined.
An example of the this block is reported below:

```
<DENSITIES base="case_name">
  <density_type>2</density_type>
  <print_out>t</print_out>
  <print_xml>t</print_xml>

  <mat_map_to_instant N_instant="10" N_mat="3">
    <mat id="fuel1">
      1   2   3   7   8   9
     </mat>
    <mat id="fuel2BP">
      4   5   6
    </mat>
    <mat id="reflector">
      10
    </mat>
  </mat_map_to_instant>

  <time_grid type="seconds">0.0  0.1  0.2 </time_grid>

  <mat id="fuel1" used="true" lib_name="fuel1">
    <isotope id="U-235" density="1.40E-04">
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
    <isotope id="U-238" density="6.31E-03" >
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
    <isotope id="BP" density="0.0" >
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
    <isotope id="STRM" density="1.00E+00" >
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
    <isotope id="SB" density="1200"
          shift_values="1200␣1300␣1000">
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
  </mat>

  <mat id="fuel2BP" used="true" lib_name="fuel2noBP">
    <time_grid type="seconds">0.2 10.0</time_grid>
    <isotope id="U-235" density="1.60E-04" >
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
    <isotope id="U-238" density="6.29E-03" >
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
    <isotope id="BP" density="8.00E-06" >
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
    <isotope id="STRM" density="1.00E+00" >
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
    <isotope id="SB" density="1200"
           shift_values="1300␣1000">
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
  </mat>

  <mat id="reflector" used="false" lib_name="reflector">
    <isotope id="STRM" density="1.0" force_density="T" >
        <tab name="BURN-UP" value="0.0" />
        <tab name="T_fuel" value="310" />
    </isotope>
  </mat>
</DENSITIES>
```

Each of these blocks are explained in dedicated sections in the following.

7.1 *density_type*

Block name: `<density_type>`

Description: the name of this material

Data type : string

Default value : 2

Selection : 1/2 – $\frac{atoms}{cm^3}$/$\frac{atoms}{barn*cm}$

7.2 *print_out*

Block name: `<print_out>`

Description: true to print out the densities for each material after each iteration (if not iteration skipping pattern is inputted)

Data type : logical

Default value : False

Selection : T/F – printing/no printing

7.3 *print_xml*

Block name: `<print_xml>`

Description: true to print out the densities for each material after each iteration (if not iteration skipping pattern is inputted) into an XML output file.

Data type : logical

Default value : False

Selection : T/F – xml printing/no xml printing

Note : the XML output file can be used to restart the calculation from an advanced "depletion" time

7.4 *mat_map_to_instant*

In this block (***Block name***: **`<mat_map_to_instant>`**) the mapping between the initial concentrations of the different zones of the mesh (geometry) can be linked with the geometry IDs (See MaterialID).
In case this block is not inputted, a one-to-one mapping between the zones here defined and the MaterialID is assumed (e.g. 1^{st} Zone here corresponds to MaterialID 1, 2^{nd} Zone here corresponds to MaterialID 2, etc.).
This XML block can/must contain the following XML attributes:

N_instant:

Description: the number of INSTANT MaterialIDs (see MaterialID)

Data type : integer

Required Entry

N_mat:

Description: the of materials (i.e. initial composition sets in this input file)

Data type : integer

Required Entry

7.4.1 *mat*

Block name: **`<mat>`**

Description: it contains the list of the the MaterialID that corresponds to this material (mixture). In this XML node the list of integer Geometry Material IDs needs to be inputted.

Data type : integer array

Required Entry

This XML block can/must contain the following XML attributes:

id:

Description: the name of this material

Data type : string

Required Entry

Note: the name inputted here must correspond to one of the material (initial compositions) defined in this input file.

7.5 *time_grid*

In this block a global time grid can be defined. This grid will be used in case of “time-dependent” problems for specifying pre-defined isotope density evolution. This corresponds, from a modeling perspective, to material (in case of isotopes associated with Macroscopic neutron cross sections) or isotope (in case of Microscopic neutron cross sections) movements.

> ***Block name***: **`<time_grid>`**
>
> ***Description***: list of time values
>
> ***Data type*** : real array
>
> **Required Entry**

This XML block must contain the following XML attributes:

> *type*:
>
> > *Description*: the units of the **`<time_grid>`**
> >
> > *Data type* : string
> >
> > *Default value* : days
> >
> > *Selection* : seconds/days – time in days/time in seconds

7.6 *mat*

In this XML block (***Block name***: **`<mat>`**) the information for describing a material initial compositions) of a zone in the mesh (geometry) is defined. This XML block can/must contain the following XML attributes:

> *id*:
>
> > *Description*: unique material name
> >
> > *Data type* : string
> >
> > **Required Entry**
>
> *used*:
>
> > *Description*: If true (T, true), the material has to be depleted. If it is false (F, false) the material shall not be burned (e.g., reflector materials).

Data type : logical

Default value : False

Selection : T/F – deplete/not deplete

lib_name:

Description: name of the Neutron Cross Section Library associated with this material (see **??**)

Data type : string

Required Entry

sph_set:

Description: name of the sph set associated to this library (see **??**)

Data type : string

Default value : n/a

The XML sub-nodes contained by the **`<mat>`** are listed in the following sub-sections.

7.6.1 *isotope*

In this XML block (***Block name***: **`<isotope>`**) the information for describing the isotope contained in the mixture associated with this material is reported. This XML block can/must contain the following XML attributes:

id:

Description: isotope identifier (name)

Data type : string

Required Entry

density:

Description: initial atomic density

Data type : real

Required Entry

shift_values:

Description: the density values for each time inputted on the local or global **`<time_grid>`**

Data type : float array

Default value : n/a

force_density:

Description: If true (T, true), the isotope density (even if the material is burned) will be kept equal to the initial one. If false (F, false) and the material is used in depletion, the density will be computed by the depletion module

Data type : logical

Default value : False

Selection : T/F – force constant/not force constant

The XML sub-nodes contained in the **`<isotope>`** are listed in the following sub-sections.

7.6.1.1 *tab*

In this XML block (***Block name***: **`<tab>`**) the information for describing the initial tabulation coordinate (s) associated with this isotope is reported. This XML block does not need to be inputted in case the associated Neutron Cross Sections are not tabulated.

This XML block must contain the following XML attributes:

name:

Description: name of the tabulation dimension

Data type : string **Required Entry**

value:

Description: coordinate (value) of this tabulation

Data type : float **Required Entry**

8 CRITICALITY_SEARCH

In the **<criticality_search>** XML input/main block is the main node to input the settings for performing criticality search schemes. The Criticality Search module is aimed to to adjust selected materials' densities in order to keep the reactor critical, ($k - eff = 1.0$ or any other target $k - eff$) for a certain configuration or during a burning cycle. The criticality search can be performed:

1. either in *isotope concentration search* ***MODE***: search acting on multiple cells (i.e. Materials) simultaneously (e.g. to find the critical soluble boron concentration). See section 8.5.
2. or *material movement search* ***MODE***: search acting on pre-defined ordered sequence of computational cells (e.g. to find the critical configuration for control rods positioning). See section 8.6.

An example (for case 1 above) of the this block is reported below:

```
<criticality_search max_iter="30">
   <desired_keff>1.0</desired_keff>
   <keff_tollerance>1.0e-6</keff_tollerance>
   <initial_perturb>-0.0015625</initial_perturb>
   <density_tune N="5" boundary="0.0␣1500.0">
     <iso_target mat_ids="fuel1">SB</iso_target>
     <iso_target mat_ids="fuel2noBP">SB</iso_target>
     <iso_target mat_ids="fuel2BP">SB</iso_target>
     <iso_target mat_ids="fuel3noBP">SB</iso_target>
     <iso_target mat_ids="fuel3BP">SB</iso_target>
   </density_tune>
</criticality_search>
```

An example (for case 2 above) of the this block is reported below:

```
<criticality_search max_iter="70">
    <desired_keff>-1.0</desired_keff>
    <keff_tollerance>1.0e-4</keff_tollerance>
    <material_movement N="5">
        <movement id="movement1" N_moves="1">
            <move_ids type="axial" isotope="PPM"
               Exclude="10-11-12-13-14-15-16-17-18"
               recompute_isos="H20" N_cells="9">
                <cell boundary=  "0-1150"> 129 </cell>
                <cell boundary=  "0-1150"> 77 </cell>
                <cell boundary=  "0-1150"> 79 </cell>
                <cell boundary=  "0-1150"> 109 </cell>
                <cell boundary=  "0-1150"> 143 </cell>
                <cell boundary=  "0-1150"> 179 </cell>
                <cell boundary=  "0-1150"> 181 </cell>
                <cell boundary=  "0-1150"> 115 </cell>
                <cell boundary=  "0-1150"> 149 </cell>
            </move_ids>
        </movement>
        <movement id="movement2" N_moves="2">
            <move_ids type="axial" isotope="PPM"
               Exclude="1-2-3-4-5-6-7-8-9" recompute_isos="H20"
               N_cells="9">
                <cell boundary=  "0-1150"> 129 </cell>
                <cell boundary=  "0-1150"> 77 </cell>
                <cell boundary=  "0-1150"> 79 </cell>
                <cell boundary=  "0-1150"> 109 </cell>
                <cell boundary=  "0-1150"> 143 </cell>
                <cell boundary=  "0-1150"> 179 </cell>
                <cell boundary=  "0-1150"> 181 </cell>
                <cell boundary=  "0-1150"> 115 </cell>
                <cell boundary=  "0-1150"> 149 </cell>
            </move_ids>
            <move_ids type="axial" isotope="PPM"
               Exclude="10-11-12-13-14-15-16-17-18"
               recompute_isos="H20" N_cells="8">
                <cell boundary=  "0-1150"> 59 </cell>
                <cell boundary=  "0-1150"> 91 </cell>
                <cell boundary=  "0-1150"> 159 </cell>
                <cell boundary=  "0-1150"> 195 </cell>
                <cell boundary=  "0-1150"> 199 </cell>
                <cell boundary=  "0-1150"> 167 </cell>
                <cell boundary=  "0-1150"> 99 </cell>
                <cell boundary=  "0-1150"> 63 </cell>
            </move_ids>
        </movement>
        <sequence> movement1 movement2 </sequence>
    </material_movement>
</criticality_search>
```

In the `<criticality_search>` XML node, the following attribute could be inputted:

max_iter:

Description: Maximum number of iterations.

Data type : integer

Default value : 15

As shown in the examples above, multiple blocks can be inputted. Each of these blocks(for both option 1 and 2) are explained in dedicated sections in the following.

8.1 *desired_keff*

Block name: `<desired_keff>`

Description: The target $k-eff$ to be used in the search

Data type : float

Default value : 1.0

8.2 *keff_tollerance*

Block name: `<keff_tollerance>`

Description: Convergence criterion. The search will stop when $\left|\frac{keff_i - desired\_keff}{desired\_keff}\right| \leq keff\_tolerance$

Data type : float

Default value : 5.e-3

8.3 *type_perturb*

Block name: `<type_perturb>`

Description: Perturbation type:

1. Relative Perturbation: The Newton-Raphson method is initiated with a relative perturbation ($density_{i+1} = density_i * (1 + initial\_perturb)$)
2. Absolute Perturbation: The Newton-Raphson method is initiated with an absolute perturbation ($density_{i+1} = initial\_perturb$)

Data type : integer

Default value : 1

Selection : 1/2 – Relative Perturbation/Absolute Perturbation

8.4 *initial_perturb*

Block name: **`<initial_perturb>`**

Description: Initial perturbation to initiate the search for the Newton-Raphson method.

Data type : float

Default value : 5.e-3

Note: **`<initial_perturb>`** usage depends on the node **`<type_perturb>`**

8.5 *density_tune*

The **`<density_tune>`** XML node is aimed to allow the user to specify the inputs required to activate the criticality search in *isotope concentration search* ***MODE***.This XML node is mutually exclusive with the **`<material_movement>`** (see section 8.6).
In the **`<density_tune>`** XML node, the following attributes could/must be inputted:

N:

Description: Number of **`<iso_target>`** nodes that will be inputted

Data type : integer

Required Entry

boundary:

Description: Lower and upper bound for the search. If the search ends up outside this boundaries, the criticality search stops and the "burning cycle" (in case of burning/depletion analysis) is ended.

Data type : float array (2 entries)

Required Entry

The **`<density_tune>`** allows only one type of XML sub-node (**`<iso_target>`**). The user must input *N* **`<iso_target>`** sub-nodes. The input requirements for this sub-node are reported in the following section.

8.5.1 *iso_target*

Block name: `<iso_target>`

Description: The name of the isotope by which the search is performed (e.g. B-10)

Data type : string

Required Entry

In the `<iso_target>` XML node, the following attributes could/must be inputted:

mat_ids:

> *Description*: Material ID (see section 7.6) containing the isotope specified in `<iso_target>`
>
> *Data type* : string
>
> **Required Entry**

recompute_isos:

> *Description*: List of isotopes for which the neutron cross sections (i.e. Macroscopic) need to be recomputed. This attribute can be used to accelerate the criticality search iterations since it allows to specify only the most important isotopes impacted by the density tuning.
>
> *Data type* : string array
>
> *Default value* : All isotopes (all Macroscopic) cross sections are recomputed.

multi_cycle_iso_to_keep:

> *Description*: List of isotopes for which the densities must be kept equal to their final state (i.e. cycle end) for multi cycle analysis.
>
> *Data type* : string array
>
> *Default value* : None

multi_cycle_init_dens:

> *Description*: initialization density for the `<iso_target>` isotope in case of multi-cycle analysis (first criticality search iteration)
>
> *Data type* : float
>
> *Default value* : None

8.6 *material_movement*

The `<material_movement>` XML node is aimed to allow the user to specify the inputs required to activate the criticality search in *material movement search* ***MODE***.This XML node is mutually exclusive with the `<density_tune>` (see section 8.5).
In the `<material_movement>` XML node, the following attributes could/must be inputted:

N:

Description: Number of `<movement>` nodes that will be inputted

Data type : integer

Required Entry

The `<material_movement>` allows only 2 types of XML sub-nodes (`<movement>` and `<sequence>`). The user must input *N* `<movement>` sub-nodes and emph1 `<sequence>` node. The input requirements for these sub-nodes are reported in the following sections.

8.6.1 *movement*

The `<movement>` is used to define a set of movements that needs to be performed simultaneously. In the `<movement>` XML node, the following attributes could/must be inputted:

id:

Description: A unique name for this movement

Data type : string

Required Entry

N_moves:

Description: Number of moves (unique groups) that will be inputted (`<move_ids>`)

Data type : integer

Required Entry

The `<movement>` allows only one type of XML sub-node (`<move_ids>`). The user must input *N_moves* `<move_ids>` sub-nodes. The input requirements for this sub-node are reported in the following section.

8.6.1.1 *move_ids*

The `<move_ids>` XML node is aimed to group computation cells (i.e. MaterialID) on which the criticality search module needs to act for achieving a certain target $k-eff$.
In the `<move_ids>` XML node, the following attributes could/must be inputted:

isotope:

> *Description*: Isotope IDs that will be used for the criticality search process.
>
> *Data type* : string array
>
> **Required Entry**

N_cells:

> *Description*: Number of `<cell>` XML sub-nodes that will be inputted/
>
> *Data type* : integer
>
> **Required Entry**

type:

> *Description*: type of connections. If "axial", PHISICS will automatically connect the the inputted cell IDs with the ones above and below in the axial (z) direction. For example, if:
>
> > the cell "10" is inputted in one of the `<cell>` XML nodes
> >
> > and the cell "34" and cell "67" are below the cell "10" in the axial direction, the movement will start from the cell "10" to continue in cell "34" and end in cell "67".
>
> *Data type* : string
>
> ***Selection*** : custom/axial – No connection/Axial connection
>
> *Default value* : custom

Exclude:

> *Description*: In case the attribute *type="axial"* , the axial layers to be excluded from the connectivity can be inputted in this optional attribute.
>
> *Data type* : dash separated list (e.g. "1-10-14").
>
> *Default value* : All the axial layers are going to be connected in case *type="axial"*

recompute_isos:

Description: List of isotopes for which the neutron cross sections (i.e. Macroscopic) need to be recomputed. This attribute can be used to accelerate the criticality search iterations since it allows to specify only the most important isotopes impacted by the density tuning.

Data type : string array

Default value : All isotopes (all Macroscopic) cross sections are recomputed.

The `<move_ids>` allows only one type of XML sub-node: `<cell>`, whose input requirements are reported in the following section.

8.6.1.1.1 *cell*

Block name: `<cell>`

Description: The cell ID (i.e. MaterialID) belonging to this move.

Data type : integer

Required Entry

In the `<cell>` XML node, the following attribute must be inputted:

boundary:

Description: Lower and upper bound (density) for this particular cell. This boundaries represent the "filling" densities for this particular cell; lower bound corresponds to 0 % insertion and the upper bound corresponds to 100 % insertion.

Data type : dash separated float array (2 entries)

Required Entry

8.6.2 *sequence*

Block name: `<sequence>`

Description: Ordered list of movement IDs (`<movement>`). In order to match a certain target $k - eff$, the code will execute the `<movement>`s in order.

Data type : string array

Required Entry

9 XS-LIBRARY

The `<Xs-Library>` XML input/main block is aimed to allow the user to input different type of neutron cross section libraries. Those libraries can than be linked to different material zones (see section 7.6, attribute named *lib_name*).
Currently, PHISICS supports 4 neutron cross section formats:

1. AMPX format (`type`="1");
2. Native PHISICS XML format (`type`="2");
3. ISOTXS format (`type`="3");
4. ECCO format (`type`="4").

An example (for case `type`="1" above) of the this block is reported below:

```
<?xml version="1.0" ?>
<XS-library n_library_sets="1" print_libraries="1" type="1">
  <na_cut>0</na_cut>
  <library_set_info ID="1">
    <library_info N_lib="1">
      <lib lib_name="ft30f001">1 4</lib>
    </library_info>
    <tab_values N_tab="3">
      <tab name="BURN-UP">600.0 900.0</tab>
      <tab name="fuel_temperature">600.0 900.0</tab>
      <tab name="mod_temperature">600.00</tab>
    </tab_values>
    <RELAP_tab_corr>
         <tab_corr name="fuel_temperature">1 1</tab_corr>
         <tab_corr name="mod_temperature">1 2</tab_corr>
     </RELAP_tab_corr>
  </library_set_info>

  <library_set ID="1">
    <BURNUP-STEP N="1" type="GWd">
      <tabulation ID="0.0" P="2">
        <POINT ID="1">
          <filename>ampx_xs/ft30f001_1</filename>
          <tab name="fuel_temperature">600.00</tab>
          <tab name="mod_temperature">600.00</tab>
          <disadvantage_factors mix="1">1 1
             1</disadvantage_factors>
          <disadvantage_factors mix="4">1 1
             0.8</disadvantage_factors>
        </POINT>
        <POINT ID="2">
          <filename>ampx_xs/ft30f001_2</filename>
          <tab name="fuel_temperature">900.00</tab>
          <tab name="mod_temperature">600.00</tab>
          <disadvantage_factors mix="1">1 1
             1</disadvantage_factors>
          <disadvantage_factors mix="4">1 1
             0.8</disadvantage_factors>
        </POINT>
      </tabulation>
    </BURNUP-STEP>
  </library_set>
</XS-library>
```

An example (for case type="2" above) of the this block is reported below:

```
<XS-library type="2" n_library_sets="1">
   <library_set_info ID="1">
      <tab_values N_tab="5">
         <BURN-UP> 0 </BURN-UP>
         <tab name="T_fuel">329.0 493.0</tab>
         <tab name="T_mod">381.0</tab>
      </tab_values>
      <RELAP_tab_corr>
         <tab_corr name="T_fuel">1 1</tab_corr>
         <tab_corr name="T_mod">1 2</tab_corr>
      </RELAP_tab_corr>
   </library_set_info>

  <library_set ID="1">
    <Micro NG="2" n_libraries="1">
      <NeutronEnergyBound>200.0 1 1.0E-6</NeutronEnergyBound>
      <BURNUP-STEP N="1" type="GWd">
        <tabulation ID="0.0" P="2">
          <POINT ID="1">
            <tab name="T_fuel">329.0</tab>
            <tab name="T_mod">381.0</tab>
            <scalar_flux> 3.739e+14 1.501e+16</scalar_flux>
```

```
<library lib_name="fuel1" n_iso="5" na="0">
  <isotope id="U235" fissile="true">
     <TotalXS>0.0 1.0</TotalXS>
     <ngXS>0.0 0.0</ngXS>
     <nalphaXS>0 0</nalphaXS>
     <npXS>0 0</npXS>
     <n2nXS>0 0</n2nXS>
     <NuFissionXS>0 2</NuFissionXS>
     <FissionXS>0 1</FissionXS>
     <ChiXS>1 0</ChiXS>
     <ScatteringXS>
        <XS>
           0.0  0.0
           0.0  0.0
        </XS>
     </ScatteringXS>
  </isotope>
  <isotope id="U238" fissile="true">
     <TotalXS>0.0 1.0</TotalXS>
     <ngXS>0.0 1.0</ngXS>
     <nalphaXS>0 0</nalphaXS>
     <npXS>0 0</npXS>
     <n2nXS>0 0</n2nXS>
     <NuFissionXS>0.0 0.00</NuFissionXS>
     <FissionXS>0.0 0.00</FissionXS>
     <ChiXS>1 0</ChiXS>
     <ScatteringXS>
        <XS>
           0.0  0.0
           0.0  0.0
        </XS>
     </ScatteringXS>
  </isotope>
  <isotope id="STRM" fissile="false">
     <TotalXS>1 1</TotalXS>
     <ngXS>0.0 0.0</ngXS>
     <nalphaXS>0 0</nalphaXS>
     <npXS>0 0</npXS>
     <n2nXS>0 0</n2nXS>
     <ScatteringXS>
        <XS>
           0.5 0.0
```

```
              0.5 1.0
           </XS>
        </ScatteringXS>
     </isotope>
   </library>
 </POINT>
 <POINT ID="2">
     <tab name="T_fuel">493.0</tab>
     <tab name="T_mod">381.0</tab>
     <scalar_flux> 3.839e+14 1.301e+16</scalar_flux>
     <library lib_name="fuel1" n_iso="5" na="0">
         <isotope id="U235" fissile="true">
             <TotalXS>0.0 1.0</TotalXS>
             <ngXS>0.0 0.0</ngXS>
             <nalphaXS>0 0</nalphaXS>
             <npXS>0 0</npXS>
             <n2nXS>0 0</n2nXS>
             <NuFissionXS>0 2</NuFissionXS>
             <FissionXS>0 1</FissionXS>
             <ChiXS>1 0</ChiXS>
             <ScatteringXS>
                 <XS>
                     0.0  0.0
                     0.0  0.0
                 </XS>
             </ScatteringXS>
         </isotope>
         <isotope id="U238" fissile="true">
             <TotalXS>0.0 1.0</TotalXS>
             <ngXS>0.0 1.0</ngXS>
             <nalphaXS>0 0</nalphaXS>
             <npXS>0 0</npXS>
             <n2nXS>0 0</n2nXS>
             <NuFissionXS>0.0 0.00</NuFissionXS>
             <FissionXS>0.0 0.00</FissionXS>
             <ChiXS>1 0</ChiXS>
             <ScatteringXS>
                 <Profile>
                     1 2
                     1 2
                 </Profile>
                 <XS>
```

```
                              0.0  0.0
                              0.0  0.0
                          </XS>
                      </ScatteringXS>
                  </isotope>
                  <isotope id="STRM" fissile="false">
                    <TotalXS>1 1</TotalXS>
                    <ngXS>0.0 0.0</ngXS>
                    <nalphaXS>0 0</nalphaXS>
                    <npXS>0 0</npXS>
                    <n2nXS>0 0</n2nXS>
                    <ScatteringXS>
                       <XS>
                          0.5 0.0
                          0.5 1.0
                       </XS>
                    </ScatteringXS>
                 </isotope>
              </library>
          </POINT>
       </tabulation>
     </BURNUP-STEP>
     </Micro>
  </library_set>
</XS-library>
```

An example (for case type="3" above) of the this block is reported below:

```
<?xml version="1.0" ?>
<XS-library type="3" n_library_sets="1">
  <na_cut>0</na_cut>
  <library_set_info ID="1">
      <tab_values N_tab="3">
        <tab name="BURN-UP">0.0</tab>
        <tab name="fuel_temperature">800.00</tab>
        <tab name="mod_density">833.04</tab>

      </tab_values>
  </library_set_info>

  <library_set ID="1" filename="ISOTXS000007" lib_name="fuel_1">
     <BURNUP-STEP N="1" type="GWd" >
```

```
        <tabulation ID="0.0" P="1">
          <POINT ID="1">
            <tab name="fuel_temperature">800.00</tab>
            <tab name="mod_density">833.04</tab>
            <library lib_name="fuel_1">
                <scalar_flux>3.9615E+13 3.7311E+13 3.7680E+13
                 3.2800E+13 3.0379E+13 1.5155E+13 4.3715E+13
                 3.1340E+13 3.4288E+13 1.2476E+13</scalar_flux>
            </library>
          </POINT>
         </tabulation>
      </BURNUP-STEP>
   </library_set>
</XS-library>
```

An example (for case `type`="4" above) of the this block is reported below:

```
<?xml version="1.0" ?>
<XS-library type="4" n_library_sets="4">
  <na_cut>0</na_cut>
  <reference_table>ECCOLIB_JEFF_31.REF</reference_table>
  <sigma_tot>2</sigma_tot>
  <library_set_info ID="1">
      <tab_values N_tab="2">
        <tab name="BURN-UP">0.0</tab>
        <tab name="fuel_temperature">1073.16</tab>
      </tab_values>
  </library_set_info>

  <library_set_info ID="2">
      <tab_values N_tab="2">
        <tab name="BURN-UP">0.0</tab>
        <tab name="fuel_temperature">1073.16</tab>
      </tab_values>
  </library_set_info>

  <library_set_info ID="3">
      <tab_values N_tab="3">
        <tab name="BURN-UP">0.0</tab>
        <tab name="fuel_temperature">823.15 1623.15</tab>
        <tab name="pb_den">10580.0 10452.0</tab>
      </tab_values>
```

```
</library_set_info>

<library_set_info ID="4">
    <tab_values N_tab="3">
      <tab name="BURN-UP">0.0</tab>
      <tab name="fuel_temperature">823.15 1623.15</tab>
      <tab name="pb_den">10580.0  10452.0</tab>
    </tab_values>
</library_set_info>

<library_set ID="1" lib_name="zext_lib">
   <BURNUP-STEP N="1" type="GWd" >
      <tabulation ID="0.0" P="1">
        <POINT ID="1">
          <filename>./LIB/ZEXT</filename>
          <tab name="fuel_temperature">1073.16</tab>
        </POINT>
      </tabulation>
   </BURNUP-STEP>
</library_set>
<library_set ID="2" lib_name="dummy_lib">
   <BURNUP-STEP N="1" type="GWd" >
      <tabulation ID="0.0" P="1">
        <POINT ID="1">
          <filename>./LIB/DUMM</filename>
          <tab name="fuel_temperature">1073.16</tab>
        </POINT>
      </tabulation>
   </BURNUP-STEP>
</library_set>
<library_set ID="3" lib_name="finn_lib">
   <BURNUP-STEP N="1" type="GWd" >
      <tabulation ID="0.0" P="4">
        <POINT ID="1">
          <filename>./LIB/FINN_c400_f550</filename>
          <tab name="fuel_temperature">823.15</tab>
          <tab name="pb_den">10580.0</tab>
        </POINT>
        <POINT ID="2">
          <filename>./LIB/FINN_c500_f550</filename>
          <tab name="fuel_temperature">823.15</tab>
          <tab name="pb_den">10452.0</tab>
          </POINT>
          <POINT ID="3">
            <filename>./LIB/FINN_c400_f1350</filename>
            <tab name="fuel_temperature">1623.15</tab>
            <tab name="pb_den">10580.0</tab>
          </POINT>
          <POINT ID="4">
            <filename>./LIB/FINN_c500_f1350</filename>
            <tab name="fuel_temperature">1623.15</tab>
            <tab name="pb_den">10452.0</tab>
          </POINT>
        </tabulation>
      </BURNUP-STEP>
  </library_set>
  <library_set ID="4" lib_name="fout_lib">
      <BURNUP-STEP N="1" type="GWd" >
         <tabulation ID="0.0" P="4">
          <POINT ID="1">
            <filename>./LIB/FOUT_c400_f550</filename>
            <tab name="fuel_temperature">823.15</tab>
            <tab name="pb_den">10580.0</tab>
          </POINT>
          <POINT ID="2">
            <filename>./LIB/FOUT_c500_f550</filename>
            <tab name="fuel_temperature">823.15</tab>
            <tab name="pb_den">10452.0</tab>
          </POINT>
          <POINT ID="3">
            <filename>./LIB/FOUT_c400_f1350</filename>
            <tab name="fuel_temperature">1623.15</tab>
            <tab name="pb_den">10580.0</tab>
          </POINT>
          <POINT ID="4">
            <filename>./LIB/FOUT_c500_f1350</filename>
            <tab name="fuel_temperature">1623.15</tab>
            <tab name="pb_den">10452.0</tab>
          </POINT>
        </tabulation>
      </BURNUP-STEP>
  </library_set>
</XS-library>
```

The XML block `<Xs-Library>` can/must contain the following XML attributes:

type:

Description: The type of the neutron cross section library format

Data type : integer

Selection : 1/2/3/4 – AMPX/XML/ISOTXS/ECCO

Required Entry

n_library_sets:

Description: Number of library sets (containing cross section libraries that share the same tabulation grid)

Data type : integer

Selection : any integer > 0

Required Entry

print_libraries:

Description: Export the read cross section libraries in an output file for checking/debugging

Data type : integer

Selection : 0/1/2 – No Print/ASCII/Tabular Format for dimensionality reduction

Default value : 0 – No Print

In the XML block `<Xs-Library>` several sub-nodes need to be inputted in order to define the cross section library to be constructed. Each of these blocks are explained in dedicated sections in the following.

9.1 na_cut

Block name: `<na_cut>`

Description: maximum order for the scattering anisotropy.

Data type : integer

Default value : 0 – isotropic scattering

Selection : any integer > 0

9.2 library_set_info

In this block the information about the library set(s) that will be inputted in the `<library_set>`(s) are listed. This XML block is in charge to define the information about the tabulation grid and mixtures (e.g. for AMPX format) common to all the libraries contained in the associated library set. This XML block can/must contain the following XML attributes:

ID:

Description: unique identifier of this library set info block. This ID will need to match the ID specified in the corresponding `<library_set>` xml block;

Data type : integer

Required Entry

Within each `<library_set_info>` node, multiple sub-nodes can/must be inputted. Each of these sub-blocks are explained in dedicated sub-sections in the following.

9.2.1 library_info

The `<library_info>` XML block can/must contain the following XML attributes:

N_lib:

Description: number of libraries will be generated by combining the mixture ids. This integer values sets the number of `<lib>` XML nodes that must be inputted within the `<library_info>`.

Data type : integer

Required Entry

Within the `<library_info>` block the user must input *N_lib* `<lib>` XML nodes:

9.2.1.1 lib

Block name: `<lib>`

Description: contains the mixture information contained in the binary library. In the body of this xml node the mixture ids that need to be retrieved are listed.

Data type : integer array

Required Entry

Note: **This XML block is required only for library `type`="1" (AMPX). It has no effect for the other library types.**

Within the **`<lib>`** block the user must input the following attribute:

lib_name:

Description: the library ID that will be generated combining the mixture ids contained in the body of the **`<lib>`** node. This ID will need to be used in the **`<DENSITIES>`** input for linking the library to the material (e.g. **`lib_name`**).

Data type : string

Required Entry

9.2.2 tab_values

The **`<tab_values>`** XML block can/must contain the following XML attributes:

N_tab:

Description: number of tabulation dimensions belonging to this **`<library_set>`** (**`ID`**)

Data type : integer

Required Entry

Within the **`<tab_values>`** block the user must input *N_tab* **`<tab>`** XML nodes:

9.2.2.1 tab

Block name: **`<tab>`**

Description: contains the tabulation points (values) for a specific tabulation dimension

Data type : float array

Required Entry

Note: The values here reported must correspond to the ones specified for each **<POINT>** in the **<library_set>** associated to this **<library_set_info>** ID.

Within the **<tab>** block the user must input the following attribute:

name:

Description: the tabulation name.

Data type : string

Required Entry

9.2.3 RELAP_tab_corr

The **<RELAP_tab_corr>** XML block is a special node that is in charge of mapping the tabulation dimensions in the cross- section library with the RELAP5-3D feedback features. This node is valid in coupling mode only (when PHISICS is coupled with RELAP5-3D). It does not produce any effect in PHISICS standalone mode.
Within the **<RELAP_tab_corr>** block the user can input a number of **<tab_corr>** equal to the number of RELAP5-3d feedback features that need to be mapped:

9.2.3.1 tab_corr

Description: an integer array of 2 elements. The first element represents the feedback type in RELAP5-3D:

- *0*, no feedback
- *1*, HR feedback. Heat Structure temperature (i.e. fuel temperature). In this case the second element of the array corresponds to the $Heat\ Structure\ number$
- *2*, VR rho moderator. Moderator density. In this case the second element of the array corresponds to the $Volume\ region\ number$
- *3*, VR T moderator. Moderator temperature. In this case the second element of the array corresponds to the $Volume\ region\ number$
- *4*, VR rho poison. Poison density (concentration). In this case the second element of the array corresponds to the $Volume\ region\ number$
- *5*, CR position. Control Rod position (insertion between 0 and 1). In this case the second element of the array has no meaning and should be set to 1
- *6*, DR position. Driver Rod position (insertion between 0 and 1). In this case the second element of the array has no meaning and should be set to 1

Data type : integer array (2 elements)

Required Entry

Note: **Only meaning when PHISICS is coupled with RELAP5-3D**

Within the **`<tab_corr>`** block the user must input the following attribute:

name:

Description: the tabulation name.

Data type : string

Required Entry

9.3 library_set

In this block the cross sections are inputted. Depending on the **`<Xs-Library>`** `type` the format of the information that need to be inputted slightly differs.
This XML block can/must contain the following XML attributes:

ID:

Description: unique identifier of this library set. This ID will need to match the ID specified in the corresponding **`<library_set_info>`** xml block;

Data type : integer

Required Entry

Within each **`<library_set>`** node, multiple sub-nodes can/must be inputted, depending on the library format. Each of these sub-blocks are explained in dedicated sub-sections in the following.

9.3.1 AMPX XML input format

For the AMPX format, the user must input a single XML node named **`<BURNUP-STEP>`** within the **`<library_set>`** block. Its content is explained/reported in the following section.
This XML block can/must contain the following XML attributes:

type:

Description: Burn-up Units

Data type : string

Selection : GWd/MWd – $\frac{GWd}{MtHM}$/$\frac{MWd}{MtHM}$

Required Entry

N:

Description: Number of burn-up steps (coordinates) in this library

Data type : integer

Required Entry

Within the **<BURNUP-STEP>** node, the user must input *N* **<tabulation>** XML sub-nodes containing the sub-grid for each specific Burn-up step.
Each **<tabulation>** node must contain the following XML attributes:

ID:

Description: The Burn-Up coordinate of this sub-grid

Data type : float

Required Entry

P:

Description: Number of points (coordinates) this burn-up step will contain

Data type : integer

Required Entry

Within the **<tabulation>** node, the user must input *P* **<POINT>** XML sub-nodes containing the information on how to retrieve the neutron cross sections from the AMPX library.
Each **<POINT>** node must contain the following XML attributes:

ID:

Description: the unique identifier of this tabulation coordinate

Data type : integer

Required Entry

Within the **<POINT>** XML node, several sub-nodes must be inputted. These sub-nodes are explained in the following sections.

9.3.1.1 filename

Block name: **`<filename>`**

Description: contains the path to the AMPX binary file

Data type : string

Required Entry

9.3.1.2 scalar_flux

In this XML block (***Block name***: **`<scalar_flux>`**) the reference group-wise scalar flux for this specific **`<POINT>`** is reported. This flux is required if the library needs to be used for SPH calculations.

Block name: **`<scalar_flux>`**

Description: group-wise scalar flux

Data type : float array

9.3.1.3 tab

In this XML block (***Block name***: **`<tab>`**) the information for describing the tabulation coordinate (s) associated with this **`<POINT>`** is reported.

Block name: **`<tab>`**

Description: tabulation coordinate

Data type : float

Note: The user must input *N_tab* -1 (see **`<tab_values>`**) **`<tab>`** nodes

This XML block must contain the following XML attribute:

name:

Description: name of the tabulation dimension

Data type : string **Required Entry**

9.3.1.4 disadvantage_factors

Block name: `<disadvantage_factors>`

Description: contains the group-wise disadvantage factors (see SCALE manual)

Data type : float array

Default : None (no disadvantage factors are applied)

Note: The disadvantage factors must be inputted for each mixture specified in the `<library_set_info>` `<library_info>` `<lib>`

Within the `<tab>` block the user must input the following attribute:

mix:

Description: the mixture this disadvantage factor set belongs to

Data type : string

Required Entry

9.3.2 Native PHISICS XML input format

For the Native XML PHISICS format, the user must input a single XML node named `<Micro>` within the `<library_set>` block. Its content is explained/reported in the following section. This XML block can/must contain the following XML attributes:

NG:

Description: Number of energy groups

Data type : integer

Required Entry

n_libraries:

Description: Number of sub-libraries will be inputted

Data type : integer

Required Entry

Within the `<Micro>` node, the user must input a sub-node indicating the energy bounds:

Block name: `<NeutronEnergyBound>`

Description: contains the neutron energy bounds

Data type : float array (NG+1 entries)

Required Entry

In addition, the user must input a XML node named `<BURNUP-STEP>` within the `<library_set>` block. Its content is explained/reported in the following section.
This XML block can/must contain the following XML attributes:

type:

Description: Burn-up Units

Data type : string

Selection : GWd/MWd – $\frac{GWd}{MtHM}$/$\frac{MWd}{MtHM}$

Required Entry

N:

Description: Number of burn-up steps (coordinates) in this library

Data type : integer

Required Entry

Within the `<BURNUP-STEP>` node, the user must input N `<tabulation>` XML sub-nodes containing the sub-grid for each specific Burn-up step.
Each `<tabulation>` node must contain the following XML attributes:

ID:

Description: The Burn-Up coordinate of this sub-grid

Data type : float

Required Entry

P:

Description: Number of points (coordinates) this burn-up step will contain

Data type : integer

Required Entry

Within the **`<tabulation>`** node, the user must input *P* **`<POINT>`** XML sub-nodes containing the actual neutron cross sections' libraries.
Each **`<POINT>`** node must contain the following XML attributes:

ID:

Description: the unique identifier of this tabulation coordinate

Data type : integer

Required Entry

Within the **`<POINT>`** XML node, several sub-nodes must be inputted. These sub-nodes are explained in the following sections.

9.3.2.1 tab

In this XML block (***Block name***: **`<tab>`**) the information for describing the tabulation coordinate (s) associated with this **`<POINT>`** is reported.

Block name: **`<tab>`**

Description: tabulation coordinate

Data type : float

Note: The user must input *N_tab* -1 (see **`<tab_values>`**) **`<tab>`** nodes

This XML block must contain the following XML attribute:

name:

Description: name of the tabulation dimension

Data type : string **Required Entry**

9.3.2.2 scalar_flux

In this XML block (***Block name***: **`<scalar_flux>`**) the reference group-wise scalar flux for this specific **`<POINT>`** is reported. This flux is required if the library needs to be used for SPH calculations.

Block name: **`<scalar_flux>`**

Description: group-wise scalar flux

Data type : float array

9.3.2.3 library

In this XML block (***Block name***: **`<library>`**) the cross sections (per isotope) can be inputted.
The user must input *n_libraries* (see **`<Micro>`** *n_libraries*) **`<library>`** nodes.
Each **`<library>`** node must contain the following XML attributes:

lib_name:

Description: the unique identifier of this library

Data type : string

Required Entry

n_iso:

Description: number of isotopes

Data type : integer

Required Entry

na:

Description: anisotropy order

Data type : integer

Required Entry

Within the **`<library>`** node, the user must input *n_iso* **`<isotope>`** XML sub-nodes containing the actual neutron cross sections' for the specific isotope.
Each **`<isotope>`** node must contain the following XML attributes:

id:

Description: unique identifier of this specific isotope (e.g. U-235)

Data type : string **Required Entry**

fissile:

Description: unique identifier of this specific isotope (e.g. U-235)

Data type : string

Default : false (no disadvantage factors are applied)

Note: The keyword is valid only when coupled with RELAP5-3D

Finally, within each **`<isotope>`** XML node, the neutron cross sections can be inputted:

- **`<TotalXS>`**

  Block name: **`<TotalXS>`**

  Description: group-wise Total Neutron Cross Section

  Data type : float array

  Required Entry

- **`<ngXS>`**

  Block name: **`<ngXS>`**

  Description: group-wise N,GAMMA Cross Section

  Data type : float array

  Required Entry

- **`<nalphaXS>`**

  Block name: **`<nalphaXS>`**

  Description: group-wise N,ALPHA Cross Section

  Data type : float array

  Default : None

- **`<npXS>`**

  Block name: **`<npXS>`**

  Description: group-wise N,Proton Cross Section

  Data type : float array

  Default : None

- **`<n2nXS>`**

  Block name: **`<n2nXS>`**

Description: group-wise N,2n Cross Section

Data type : float array

Default : None

- **<NuFissionXS>**

  Block name: **<NuFissionXS>**

  Description: group-wise Nu Fission Cross Section

  Data type : float array

  – **Required Entry**

- **<FissionXS>**

  Block name: **<scalar_flux>**

  Description: group-wise Fission Cross Section

  Data type : float array

  Default : None

- **<ChiXS>**

  Block name: **<scalar_flux>**

  Description: group-wise Total Spectrum

  Data type : float array

  – **Required Entry**

- **<ScatteringXS>**

  In this XML block (*Block name*: **<ScatteringXS>**) the scattering cross section can be inputted.
  Two sub-nodes must be inputted:

  – **<Profile>**:

    Block name: **<Profile>**

    Description: profile of the scattering matrix $\sigma_{s,n}^{g'\to g}, g' = 1, \cdots, NG; g = 1, \cdots, NG; n = 0, \cdots, NA$. **<Profile>** is designed to save the input effort and the size of the file containing the scattering cross sections. There are $NG \times (NA + 1)$ pairs of integers. All these pairs are ordered by energy index g first then anisotropy group n, i.e. $((g = 1, G), n = 0, NA)$. Each pair has the first departure scattering group and the last departure scattering group for the anisotropy n and the group g. **<Profile>** is used to reduce the amount of inputs for the scattering matrix. Users do not have to give **<Profile>**. If **<Profile>** is absent, all entries of the scattering matrix must be inputted including all zero entries. This also means the default value of all pairs of **<Profile>** is $(1, NG)$. For example, the following 7-by-7 scattering matrix where non-zeros are marked with $\times$,

$$\begin{array}{cc} & \rightarrow g' \\ \downarrow_g & \begin{bmatrix} \times & & & & & & \\ \times & \times & & & & & \\ \times & \times & \times & & & & \\ & & \times & \times & \times & & \\ & & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times & \times \end{bmatrix}_{7\times 7} \end{array}$$

has **<Profile>** [1 1 1 2 1 3 3 5 3 7 3 7 3 7].

Data type : float array

- **<XS>**:

  Block name: **<XS>**

  Description: the $NA+1$ scattering matrix $\sigma_{s,n}^{g'\rightarrow g}$. The unit is $1/cm$. All scattering matrices are inputted sequentially. For each scattering matrix, entries specified by *Profile* are inputted row by row. The column index of the scattering matrix is g'; the row index of the scattering matrix is g. If there is no up-scatterings, the scattering matrix is strictly low-triangular. Note that PHISICS treats all groups without up-scatterings as the "fast" group and all groups with up-scatterings as the "thermal" group. "fast" here does not necessarily mean the energy of neutrons in the group is high. Thermal iteration may be required for solving problems with up-scattering materials. Every material must have the scattering cross sections.

  Data type : float array

9.3.3 ISOTXS XML input format

For the ISOTXS format, the user can decided to have a single ISOTXS file (appending multiple libraries in the same file) or multiple files like in the AMPX formats.
In case a single file is inputted, the user must input the following attribute in the **<library_set>**:

filename:

Description: ISOTOXS combined file

Data type : string

Note: In this case all the libraries and POINTS must be ordered in the ISOTOXS file

In addition,for the ISOTXS format, the user must input a single XML node named **`<BURNUP-STEP>`** within the **`<library_set>`** block. Its content is explained/reported in the following section.
This XML block can/must contain the following XML attributes:

type:

Description: Burn-up Units

Data type : string

Selection : GWd/MWd – $\frac{GWd}{MtHM}$/$\frac{MWd}{MtHM}$

Required Entry

N:

Description: Number of burn-up steps (coordinates) in this library

Data type : integer

Required Entry

Within the **`<BURNUP-STEP>`** node, the user must input *N* **`<tabulation>`** XML sub-nodes containing the sub-grid for each specific Burn-up step.
Each **`<tabulation>`** node must contain the following XML attributes:

ID:

Description: The Burn-Up coordinate of this sub-grid

Data type : float

Required Entry

P:

Description: Number of points (coordinates) this burn-up step will contain

Data type : integer

Required Entry

Within the **`<tabulation>`** node, the user must input *P* **`<POINT>`** XML sub-nodes containing the information on how to retrieve the neutron cross sections from the ISOTXS library.
Each **`<POINT>`** node must contain the following XML attributes:

ID:

Description: the unique identifier of this tabulation coordinate

Data type : integer

Required Entry

Within the **`<POINT>`** XML node, several sub-nodes must be inputted. These sub-nodes are explained in the following sections.

9.3.3.1 filename

Block name: **`<filename>`**

Description: contains the path to the AMPX binary file

Data type : string

Note: This node MUST be inputted if the *filename* attribute in the **`<library_set>`** node has not been inputted

9.3.3.2 tab

In this XML block (***Block name***: **`<tab>`**) the information for describing the tabulation coordinate (s) associated with this **`<POINT>`** is reported.

Block name: **`<tab>`**

Description: tabulation coordinate

Data type : float

Note: The user must input *N_tab* -1 (see **`<tab_values>`**) **`<tab>`** nodes

This XML block must contain the following XML attribute:

name:

Description: name of the tabulation dimension

Data type : string **Required Entry**

9.3.3.3 scalar_flux

In this XML block (***Block name***: **`<scalar_flux>`**) the reference group-wise scalar flux for this specific **`<POINT>`** is reported. This flux is required if the library needs to be used for SPH calculations.

> ***Block name***: **`<scalar_flux>`**
>
> ***Description***: group-wise scalar flux
>
> ***Data type*** : float array

9.3.4 ECCO XML input format

For using the ECCO format, the user MUST input 2 additional XML nodes within the **`<XS-library>`** root node:

- **`<reference_table>`**

  Block name: **`<reference_table>`**

  Description: the ECCO library reference table

  Data type : string

  Required Entry

- **`<sigma_tot>`**

  Block name: **`<sigma_tot>`**

  Description: Sigma Tot Flag

  Data type : integer

  Selection : 1/2/3 – The total cross section is the P0 one (consistent PN)/The transport XS is used as total (For P0 approximation)/The total cross section is the P1 one (correction on the anisotropic scattering XSs)

  Default : 1

For the ECCO format, the user must input a single XML node named **`<BURNUP-STEP>`** within the **`<library_set>`** block. Its content is explained/reported in the following section. This XML block can/must contain the following XML attributes:

> *type*:

Description: Burn-up Units

Data type : string

Selection : GWd/MWd – $\frac{GWd}{MtHM}$/$\frac{MWd}{MtHM}$

Required Entry

N:

Description: Number of burn-up steps (coordinates) in this library

Data type : integer

Required Entry

Within the **`<BURNUP-STEP>`** node, the user must input *N* **`<tabulation>`** XML sub-nodes containing the sub-grid for each specific Burn-up step.
Each **`<tabulation>`** node must contain the following XML attributes:

ID:

Description: The Burn-Up coordinate of this sub-grid

Data type : float

Required Entry

P:

Description: Number of points (coordinates) this burn-up step will contain

Data type : integer

Required Entry

Within the **`<tabulation>`** node, the user must input *P* **`<POINT>`** XML sub-nodes containing the information on how to retrieve the neutron cross sections from the ECCO library.
Each **`<POINT>`** node must contain the following XML attributes:

ID:

Description: the unique identifier of this tabulation coordinate

Data type : integer

Required Entry

Within the **`<POINT>`** XML node, several sub-nodes must be inputted. These sub-nodes are explained in the following sections.

9.3.4.1 filename

Block name: `<filename>`

Description: contains the path to the AMPX binary file

Data type : string

Required Entry

9.3.4.2 scalar_flux

In this XML block (***Block name***: `<scalar_flux>`) the reference group-wise scalar flux for this specific `<POINT>` is reported. This flux is required if the library needs to be used for SPH calculations.

Block name: `<scalar_flux>`

Description: group-wise scalar flux

Data type : float array

9.3.4.3 tab

In this XML block (***Block name***: `<tab>`) the information for describing the tabulation coordinate (s) associated with this `<POINT>` is reported.

Block name: `<tab>`

Description: tabulation coordinate

Data type : float

Note: The user must input *N_tab* -1 (see `<tab_values>`) `<tab>` nodes

This XML block must contain the following XML attribute:

name:

Description: name of the tabulation dimension

Data type : string **Required Entry**

10 DEPLETION_INPUT

In the **<DEPLETION_INPUT>** XML input/main block, all the information regarding the depletion (MRTAU) settings are inputted.
An example of the this block is reported below:

```
<DEPLETION_INPUT>
  <title>a title</title>
  <input_files>lib_inp_path_2d_bench.xml</input_files>
    <calculation_control>
      <calc_type>2</calc_type>
      <power_calc_type>1</power_calc_type>
      <exp_method>CRAM</exp_method>
      <method_order>14</method_order>
      <spectrum_type>thermal</spectrum_type>
      <decay_heat_type>2</decay_heat_type>
      <burn_up>3</burn_up>
      <total_power>kW</total_power>
      <multi_cycle_control NTP="2">
         <shuffle filename="shuffle_input.xml" num_shuffle="1">
            <cycle eoc="1" scheme="fullCore"></cycle>
         </shuffle>
      </multi_cycle_control>
      <n_tab_interval>2</n_tab_interval>
      <power_history>1.0 1.0</power_history>
      <sep_eff_control>1</sep_eff_control>
    </calculation_control>
    <output_control>
      <plot_type>2</plot_type>
      <extra_plot>3</extra_plot>
      <densities_csv>t</densities_csv>
      <time_step_skip_out>1 0</time_step_skip_out>
      <output_representation>user</output_representation>
    </output_control>
    <time_control>
      <tab_time_step>20 80</tab_time_step>
      <tabulation_boundaries>
        <low>0.0 200.0</low>
        <up>200.0  1000.0</up>
      </tabulation_boundaries>
      <cycle_parameters>
        <cycle_lenght>1000.0</cycle_lenght>
        <begin_life>0.0</begin_life>
        <end_life>1000.0</end_life>
      </cycle_parameters>
    </time_control>
</DEPLETION_INPUT>
```

Each of these blocks are explained in dedicated sections in the following.

10.1 *title*

Block name: `<title>`

Description: the name associated with this depletion run

Data type : string

Default value : “”

10.2 *input_files*

Block name: `<input_files>`

Description: path (relative to the running directory) of the xml file containing the names and paths of the MRTAU library files

Data type : string

Default value : lib_inp_path.xml

The input listed in the `<input_files>` is an XML file.
An example of the this block is reported below:

```
<mrtau_lib_inp_list>
  <input type="1">
    <iso_list_inp>input_dir/IsotopeList.dat</iso_list_inp>
    <mass_a_weight_inp>input_dir/mass.inp</mass_a_weight_inp>
  </input>
  <libraries type="1">
    <decay_lib>lib_dir/decay.dat</decay_lib>
    <xs_sep_lib> lib_dir/budep.inp</xs_sep_lib>
    <fiss_yields_lib>lib_dir/FissionYield.dat</fiss_yields_lib>
    <fiss_q_values_lib>lib_dir/FissQValues.dat</fiss_q_values_lib>
    <cram_lib> lib_dir/CRAM_coeff_PF.dat</cram_lib>
  </libraries>
  <paths type="1">
    <n_reactions>
      <n_gamma>path/N,G.path</n_gamma>
      <n_gamma_ex>path/N,Gx.path</n_gamma_ex>
      <n_2n>path/N,2N.path</n_2n>
      <n_p>path/N,P.path</n_p>
      <n_alpha>path/N,ALPHA.path</n_alpha>
    </n_reactions>
    <decay>
      <alpha>path/AlphaDecay.path</alpha>
      <beta>path/BetaDecay.path</beta>
      <beta_ex>path/BetaxDecay.path</beta_ex>
      <beta_plus>path/Beta+Decay.path</beta_plus>
      <beta_plus_ex>path/Beta+xDecay.path</beta_plus_ex>
      <int_tra>path/IntTraDecay.path</int_tra>
    </decay>
  </paths>
  <output>
    <reactions>reactions.out</reactions>
    <atoms_plot>numbers.out</atoms_plot>
    <atoms_csv>numbers.csv</atoms_csv>
    <decay_heat>DecayHeat.out</decay_heat>
    <bu_power>BuPower.out</bu_power>
    <flux>flux_out.out</flux>
  </output>
</mrtau_lib_inp_list>
```

The main tag is `<mrtau_lib_inp_list>`. Within this XML several XML blocks should be defined. Each of these blocks are explained in dedicated sub-sections in the following.

10.2.1 *input*

In this block (***Block name***: `<input>`) the input files (i.e. list of Isotope to be tracked and initial masses (standalone only)/atomic weights for each isotopes) are listed in dedicated sub-XML nodes (see the following).
This XML block can/must contain the following XML attributes:

type:

Description: The type (format) of the files. “1” for Text and “2” for XML-based files.

Data type : integer

Default value : 1

Selection : 1/2 – Text/XML

Note: Only Text format is currently available.

Within this XML sub-block 2 XML nodes must be defined. Each of these nodes are explained in dedicated sub-sections in the following.

10.2.1.1 *iso_list_inp*

Block name: `<iso_list_inp>`

Description: it contains the path (relative to the running directory) of the Isotope List input file.

Data type : string

Required Entry

10.2.1.2 *mass_a_weight_inp*

Block name: `<mass_a_weight_inp>`

Description: it contains the path (relative to the running directory) of the input file containing the Initial Masses (standalone only) and the Atomic Weights.

Data type : string

Required Entry

10.2.2 *libraries*

In this block (***Block name***: `<libraries>`) the nuclear data libraries’ files for the depletion/burn-up calculations are listed in dedicated sub-XML nodes (see the following).
This XML block can/must contain the following XML attributes:

type:

Description: The type (format) of the files. “1” for Text and “2” for XML-based files.

Data type : integer

Default value : 1

Selection : 1/2 – Text/XML

Note: Only Text format is currently available.

Within this XML sub-block 5 XML nodes must be defined. Each of these nodes are explained in dedicated sub-sections in the following.

10.2.2.1 *decay_lib*

Block name: `<decay_lib>`

Description: it contains the path (relative to the running directory) of the library files that contains all the decay constants and modes for each isotope

Data type : string

Required Entry

10.2.2.2 *xs_sep_lib*

Block name: `<xs_sep_lib>`

Description: it contains the path (relative to the running directory) of the library that contains the separation efficiencies and 1 group cross sections (standalone only).

Data type : string

Required Entry

10.2.2.3 *fiss_yields_lib*

Block name: `<fiss_yields_lib>`

Description: it contains the path (relative to the running directory) of the library that contains the fission yields for each fission product or light element.

Data type : string

Required Entry

10.2.2.4 *fiss_q_values_lib*

Block name: `<fiss_q_values_lib>`

Description: it contains the path (relative to the running directory) of the library that contains the fission Q values (energy released per fission) for each heavy element (e.g. actinides, fertile and fissile materials).

Data type : string

Required Entry

10.2.2.5 *cram_lib*

Block name: `<cram_lib>`

Description: it contains the path (relative to the running directory) of the library that contains the Chebyshev Rational Approximation Method coefficients for each allowable order (up to 30^{th} order).

Data type : string

Required Entry

10.2.3 *paths*

In this block (***Block name***: `<paths>`) the isotope chains for both decay and neutron interactions are listed in dedicated sub-XML nodes (see the following).
This XML block can/must contain the following XML attributes:

type:

Description: The type (format) of the files. "1" for Text and "2" for XML-based files.

Data type : integer

Default value : 1

Selection : 1/2 – Text/XML

Note: Only Text format is currently available.

Within this XML sub-block, 2 XML nodes must be defined. Each of these nodes are explained in dedicated sub-sections in the following.

- `<n_reactions>`, for listing the reaction chains for neutron interactions. The number of nodes within this block is variable depending on how may neutron reactions need to be considered.
- `<decay>`, for listing the reaction chains for neutron interactions. The number of nodes within this block is variable depending on how may neutron reactions need to be considered.

10.2.3.1 *n_reactions*

Block name: `<n_reactions>`

Description: Node for listing the raction chains for neutron interactions. Within this node, a variable number of XML nodes can be listed depending on the number of neutron reactions that need to be considered. The tags (e.g. `<n_gamma>`, etc.) are needed for file identification only and can be completely user-defined.

Data type : XML node

Required Entry

10.2.3.2 *decay*

Block name: `<decay>`

Description: Node for listing the decay chains for each isotope. Within this node, a variable number of XML nodes can be listed depending on the number of decay modes that need to be considered. The tags (e.g. `<alpha>`, etc.) are needed for file identification only and can be completely user-defined.

Data type : XML node

Required Entry

10.2.4 *output*

In this block (***Block name***: `<output>`) the default output file names for the depletion/burn-up calculations are listed in dedicated sub-XML nodes (see the following). Within this XML-subblock 6 XML nodes must be defined. Each of these nodes are explained in dedicated sub-sections in the following.

10.2.4.1 *reactions*

Block name: `<reactions>`

Description: it contains the file name for the output of the reaction rates (if requested).

Data type : string

Required Entry

Note: Used in standalone calculations only.

10.2.4.2 *atoms_plot*

Block name: `<atoms_plot>`

Description: it contains the file name for the output (Text file) of the density (or mass) evolutions for each isotope during the depletion calculation (if requested)

Data type : string

Required Entry

10.2.4.3 *atoms_csv*

Block name: `<atoms_csv>`

Description: it contains the output file name for the output (Comma Separated Value - CSV file) of the density (or mass) evolutions for each isotope during the depletion calculation.

Data type : string

Required Entry

10.2.4.4 *decay_heat*

Block name: `<decay_heat>`

Description: it contains the file name for the output of the decay heat evolution(if requested).

Data type : string

Required Entry

Note: Used in standalone calculations only.

10.2.4.5 *bu_power*

Block name: `<bu_power>`

Description: it contains the file name for the output of the burn up and power evolution(if requested).

Data type : string

Required Entry

Note: Used in standalone calculations only.

10.2.4.6 *flux*

Block name: `<flux>`

Description: it contains the file name for the output of the flux evolution(if requested).

Data type : string

Required Entry

Note: Used in standalone calculations only.

10.3 *calculation_control*

This section (***Block name***: `<calculation_control>`) is used to set up the control settings of the depletion/burn-up calculation. Within this XML block several options (XML nodes) can/must be defined. Each of these nodes are explained in dedicated sub-sections in the following.

10.3.1 *calc_type*

Block name: `<calc_type>`

Description: Calculation type. If 1, then just Actinides are included in the depletion calculation. If 2, both Actinides and Fission Products (light elements) will be included.

Data type : integer

Required Entry

Selection : 1/2 – Actinides/Actinides+Fission Products

Note: Used in standalone calculations only.

10.3.2 *power_calc_type*

Block name: `<power_calc_type>`

Description: Power Calculation type. If 1, calculation is performed at constant power. If 2, calculation is performed at constant flux. If 3, calculation is performed at constant thermal power (power in W).

Data type : integer

Default value : 1

Selection : 1/2/3 – Constant Power/Constant Flux/Constant(specified) thermal power

Note: Used in standalone calculations only.

In case of option 3, the following attribute must be inputted:

power:

Description: Power in Watts.

Data type : float

Required Entry

10.3.3 *exp_method*

Block name: `<exp_method>`

Description: Burnup Matrix Exponential solution method. The method to be used for the depletion calculation. Two methods are available: 1) CRAM, the Chebychev Rational Approximation Method, and 2) TAYLOR, truncated Taylor series based approximation

Data type : string

Default value : CRAM

Selection : CRAM/Taylor – CRAM/Taylor

10.3.4 *method_order*

Block name: `<method_order>`

Description: Order of the Burnup Matrix Exponential solution method (either CRAM or Taylor) Values greater than 2 can be inputted.

Data type : integer

Default value : 14

10.3.5 *spectrum_type*

Block name: `<spectrum_type>`

Description: fast / thermal. Depletion Spectrum Flag. This is the neutron flux spectrum characteristic of this depletion case (Used for Fission Yields and, in standalone, for cross section selection).

Data type : integer

Default value : 1

Selection : 1/2 – Thermal/Fast

10.3.6 *decay_heat_type*

Block name: `<decay_heat_type>`

Description: Decay Heat Calculation Flag. If 1, no decay heat is calculated. If 2, decay heat is calculated and it is expressed in kW. If 3, decay heat is calculated and it is expressed in MeV/s.

Data type : integer

Default value : 1

Selection : 1/2/3 – No Decay/Yes in kW/ Yes in MeV/s

10.3.7 *decay_heat_sm*

Block name: **`<decay_heat_sm>`**

Description: Decay Heat Surrogate Model. It contains the path (relative to the running directory) of the file containing the surrogate model (Generated by the RAVEN software) to surrogate the Decay Heat power/

Data type : string

Default value : “”. No surrogate model

The following attribute must be inputted:

name:

Description: Name of the surrogate model to be used in the file specified in node **`<decay_heat_sm>`**

Data type : string

Required Entry

10.3.8 *burn_up*

Block name: **`<burn_up>`**

Description: Burnup Calculation Flag. If 1, no burnup calculation is performed. If 2, burnup calculation is performed and it is expressed in $MWd/MtHM$. If 3, burnup calculation is performed and it is expressed in $GWd/MtHM$.

Data type : integer

Default value : 1

Selection : 1/2/3 – No Burnup/Yes in $MWd/MtHM$/ Yes in $GWd/MtHM$

10.3.9 *total_power*

Block name: **`<total_power>`**

Description: Total Power (Fission+Decay) Calculation Flag. If NO, no total power is computed. If kW, total power is computed and it is expressed in kW. If MW, total power is computed and it is expressed in MW.

Data type : string

Default value : No

Selection : No/kW/MW – No Computation/Yes in kW/ Yes in MW

10.3.10 *interpolate*

Block name: `<interpolate>`

Description: Cross Section interpolation Flag. If True, perform linear interpolation, if False, do not perform interpolation. This flag is available only for the standalone mode. In coupled mode the cross section manipulation belongs to an external module.

Data type : bool

Default value : F

Selection : T/F – Linear Interpolation/No

Note: Used in standalone calculations only.

10.3.11 *multi_cycle_control*

In this block (***Block name***: `<multi_cycle_control>`) the multi-cycle control is inputted. In this XML block the multi-cycle analysis can be controlled, both in terms of number of "depletion" cycles and shuffling schemes. The following attribute must be inputted:

NTP:

Description: Number of cycles.

Data type : integer

Required Entry

If the *NTP* > 1, shuffling information (Fuel Management schemes need to be inputted). In order to input the shuffling info, the sub-node `<shuffle>` needs to be inputted.
This XML block must contain the following XML attributes:

filename:

Description: it contains the path (relative to the running directory) of the file containing the shuffling information (fuel managment schemes)

Data type : string

Required Entry

num_shuffle:

Description: the number of shuffling schemes that will be applied and listed

Data type : integer

Required Entry

Within this XML sub-block a number of XML nodes (named **`<cycle>`**) equal to *num_shuffle* must be inputted, containing the following attributes:

eoc:

Description: ID of the cycle (end of cycle) on which this shuffling scheme needs to be applied to.

Data type : integer

Required Entry

scheme:

Description: scheme name

Data type : string

Required Entry

Information regarding how to input the shuffling schemes can be found in section ??.

10.3.12 *n_tab_interval*

Block name: **`<n_tab_interval>`**

Description: Number of time main intervals. This number represents the number of subdivisions of cycle, in which different number of time-steps or power histories can be inputted.

Data type : integer

Default value : 1

10.3.13 *flux*

Block name: `<flux>`

Description: Tabulated Fluxes. Fluxes for each `<n_tab_interval>` interval. The number of values here must match `<n_tab_interval>`.

Data type : float array

Note: Only available in standalone mode.

The following attribute must be inputted:

NG:

Description: number of energy groups

Data type : integer

Required Entry

10.3.14 *power_history*

Block name: `<power_history>`

Description: Tabulated Power History. Power level (in per unit) for each `<n_tab_interval>` interval. The number of values here must match `<n_tab_interval>`.

Data type : float array

Required Entry

10.3.15 *sep_eff_control*

Block name: `<sep_eff_control>`

Description: Separation Efficiency Flag. If 1, separation efficiencies are applied at Lower Bound of Tabulation Interval. If 2, separation efficiencies are applied at the Entire Tabulation Interval.

Data type : integer

Default value : 1

Note: Only available in standalone mode.

10.4 *output_control*

This section (***Block name***: `<output_control>`) is used to set up the control for the output of the depletion/burn-up calculation. Within this node several XML sub-nodes can be inputted. Each of these nodes are explained in dedicated sub-sections in the following.

10.4.1 *output_representation*

Block name: `<output_representation>`

Description: Output Representation Settings. In order to maximize the speed and the accuracy of the exponential methods, the code automatically sorts the isotopes with respect to their atomic number. In case the user wants to keep the order of the input, this flag can be used.

Data type : string

Default value : ATOMIC

Selection : ATOMIC/USER – Sorted order/User order

10.4.2 *plot_type*

Block name: `<plot_type>`

Description: Plot type flag. This flag controls the output type of the depletion calculation (units).

Data type : integer

Selection : 1/2/3 – No output densities/Output in kg/Output in Atom Numbers

10.4.3 *plot*

Block name: `<plot>`

Description: Output controller. With this flag all the outputs of the depletion/burn-up calculations can be disabled.

Data type : boolean

Default value : T

Selection : T/F – On/Off

10.4.4 *plot_input*

Block name: `<plot_input>`

Description: Input data print option flag. It controls the output of the input settings of the depletion/burn-up calculation.

Data type : boolean

Default value : T

Selection : T/F – Print input settings/Do not print input settings.

10.4.5 *average_bu*

Block name: `<average_bu>`

Description: Flag to activate the computation of the average burn-up over the different depletion zones (i.e. materials).

Data type : boolean

Default value : F

Selection : T/F – On/Off

10.4.6 *print_bu_csv*

Block name: `<print_bu_csv>`

Description: Flag to export the burn-up evolution into a CSV file.

Data type : integer

Default value : 0

Selection : 1/0 – On/Off

10.4.7 *print_dh_csv*

Block name: `<print_dh_csv>`

Description: Flag to export the decay heat evolution into a CSV file.

Data type : integer

Default value : 0

Selection : 1/0 – On/Off

10.4.8 *print_power_csv*

Block name: `<print_power_csv>`

Description: Flag to export the thermal power evolution into a CSV file.

Data type : integer

Default value : 0

Selection : 1/0 – On/Off

10.4.9 *print_power_dens_csv*

Block name: `<print_power_dens_csv>`

Description: Flag to export the thermal power density evolution into a CSV file.

Data type : integer

Default value : 0

Selection : 1/0 – On/Off

10.4.10 *densities_csv*

Block name: `<densities_csv>`

Description: Flag to export the Atomic densities into a CSV file.

Data type : boolean

Default value : F

Selection : T/F – On/Off

10.4.11 *time_step_skip_out*

Block name: **`<time_step_skip_out>`**

Description: Frequency of printing in terms of time-steps. (e.g. 1 means print every other time-steps, 2 means print every 2 other time-steps, etc.).

Data type : integer

Default value : 0 (all time-steps will be printed)

10.4.12 *tab_time_step_skip*

Block name: **`<tab_time_step_skip>`**

Description: Frequency of printing, for each time interval (see **`<n_tab_interval>`**), in terms of time-steps. (e.g. 1 means print every other time-steps, 2 means print every 2 other time-steps, etc.).

Data type : integer array (size **`<n_tab_interval>`**)

Default value : all time-steps will be printed

Note: : This node has priority over the **`<time_step_skip_out>`**

10.4.13 *out_time_points*

Block name: **`<out_time_points>`**

Description: List of times of printing.

Data type : real array

Default value : all time stamps will be printed

Note: : This node has priority over the **`<time_step_skip_out>`** and **`<tab_time_step_skip>`**

10.5 *time_control*

This section (***Block name***: **`<time_control>`**) is used to set up the calculation in terms of temporal settings. The following attribute could optionally be inputted:

type:

Description: time units

Data type : string

Default value : DAYS

Selection : SECONDS/HOURS/DAYS – time in seconds/time in hours/time in days

Within this node several XML sub-nodes can be inputted. Each of these nodes are explained in dedicated sub-sections in the following.

10.5.1 *tab_time_step*

Block name: `<tab_time_step>`

Description: Number of time-steps to be used in each time interval (see `<n_tab_interval>`)

Data type : integer array

Required Entry

10.5.2 *tabulation_boundaries*

This section (***Block name***: `<tabulation_boundaries>`) is used to specify the lower and upper boundaries (in time) of the depletion problem. Within this node 2 XML sub-nodes can be inputted. Each of these nodes are explained in dedicated sub-sections in the following.

10.5.2.1 *low*

Block name: `<low>`

Description: List of time lower boundaries. A number equal to `<n_tab_interval>` must be inputted.

Data type : float array

- **Required Entry**

10.5.2.2 *up*

Block name: `<up>`

Description: List of time upper boundaries. A number equal to `<n_tab_interval>` must be inputted.

Data type : float array

- **Required Entry**

10.5.3 *cycle_parameters*

This section (***Block name***: `<cycle_parameters>`) is used to specify the temporal parameters characterizing the cycle (or cycles if `<multi_cycle_control>` attribute *NTP* is ≥ 1). Within this node 3 XML sub-nodes must be inputted. Each of these nodes are explained in dedicated sub-sections in the following.

10.5.3.1 *cycle_lenght*

Block name: `<cycle_lenght>`

Description: Cycle length either in days, hours or seconds depending on the attribute *type* of the `<time_control>` XML node.

Data type : float

Required Entry

10.5.3.2 *begin_life*

Block name: `<begin_life>`

Description: Begin of the burning either in days, hours or seconds depending on the attribute *type* of the `<time_control>` XML node.

Data type : float

Required Entry

10.5.3.3 *end_life*

Block name: `<end_life>`

Description: End of the burning either in days, hours or seconds depending on the attribute *type* of the `<time_control>` XML node.

Data type : float

Required Entry

11 SHUFFLE_INPUT

In the <Shuffle_Input> XML input/main block, all the information regarding the fuel management strategy (i.e. Shuffling schemes) settings are inputted.
An example of the this block is reported below:

```
<Shuffle_Input>
  <materials N_mat="1">
    <new>
     <mat id="fuel3noBP" used="true" lib_name="fuel3noBP">
       <isotope id="U-235" density="2.20E-04" />
       <isotope id="U-238" density="6.23E-03" />
        <isotope id="BP" density="0.0"/>
        <isotope id="STRM" density="1.00E+00" />
        <isotope id="SB" density="1200" />
      </mat>
    </new>
  </materials>
  <assemblies N_assemblies="193">
    <assembly id="1" type="Cartesian">
      1 2 3 22 23 24 43 44 45
    </assembly>
    <assembly id="2" type="Cartesian">
      4 5 6 25 26 27 46 47 48
    </assembly>
    . . .
  </assemblies>
  <shuffle_scheme N_schemes="2">
    <scheme id="fullCore" N_move="3">
      <move from="core" to="out">
        <from_assembly>
          100 52 94 142 101 38 93 156 99 67 95 127 85 53 51 79
             109 141 143 115
          73 15 11 61 121 179 183 133 87 26 24 77 107 168 170
             117 57 28 22 47
          137 166 172 147 71 40 36 63 123 154 158 131 43 29 21
             33 151 165 173
          161 69 65 125 129 83 81 111 113
        </from_assembly>
      </move>
      <move from="new" to="core">
        <from>
          fuel3noBP
        </from>
        <to_assembly>
          104 4 90 190 89 5 3 75 105 189 191 119 74 6 2 60 120
             188 192 134 59 7 1
          45 135 187 193 149 87 26 24 77 107 168 170 117 58 16
             10 46 136 178 184 148
          44 17 9 32 150 177 185 162 31 18 8 19 163 176 186 175
             42 34 152 160 30
          20 164 174
        </to_assembly>
      </move>
      <move from="core" to="core">
        <from_assembly>
          98 82 96 112 74+90 6+90 2+90 60+90 120+90
          188+90 192+90 134+90 104+180 4+180 90+180
          190+180  119 5 75 189 56 41 35 48 138 153 159
          146 70 54 50 64 124 140 144 130 84 68 66 80 110
          126 128 114 88+90 14+90 12+90 76+90 106+90
          180+90 182+90 118+90 58+90 16+90 10+90 46+90
          136+90 178+90 184+90 148+90 103+180 13+180 91+180
          181+180 102+180 25+180 92+180 169+180 89+180 3+180
          105+180 191+180 174+180 30+180 20+180 164+180
          160+180 42+180 34+180 152+180 31+180 18+180 8+180
          19+180 163+180 176+180 186+180 175+180
          86+180 39+180 37+180 78+180 108+180 155+180 157+180
          116+180 59+180 7+180 1+180 45+180 135+180 187+180
          193+180 149+180 44+180 17+180 9+180 32+180 150+180
          177+180 185+180 162+180 72+180 27+180 23+180 62+180
          122+180 167+180 171+180 132+180
        </from_assembly>
        <to_assembly>
          102 25 92 169 85 53 51 79 109 141 143 115 98 82 96 112
             103
          13 91 181 88 14 12 76 106 180 182 118 56
          41 35 48 138 153 159 146 43 29 21 33 151 165 173 161
             73 15
          11 61 121 179 183 133 57 28 22 47 137 166
          172 147 99 67 95 127 100 52 94 142 101 38 93 156 113
             83 81
          111 129 69 65 125 84 68 66 80 110 126 128 114
          86 39 37 78 108 155 157 116 70 54 50 64 124 140 144
```

```
          130 71
        40 36 63 123 154 158 131 72 27 23 62 122 167
        171 132
      </to_assembly>
    </move>
  </scheme>
  <scheme id="node_by_node" N_move="4">
    <move from="core" to="pool">
      <from>
        832 833 834 877 878 879 922 923 924  418 419 420 463
           464 465 508 509 510
      </from>
    </move>
    <move from="core" to="out">
      <from>
        814 815 816 859 860 861 904 905 906 1228 1229 1230
           1273 1274 1275 1318 1319 1320
      </from>
    </move>
    <move from="new" to="core">
      <from>
        fuel3noBP
      </from>
      <to>
        832 833 834 877 878 879 922 923 924  10 11 12 31 32 33
           52 53 54
      </to>
    </move>
    <move from="core" to="core">
      <from>
         832 833 834 877 878 879 922 923 924  10 11 12 31 32
            33 52 53 54
      </from>
      <to>
        418 419 420 463 464 465 508 509 510  832 833 834 877
           878 879 922 923 924
      </to>
    </move>
  </scheme>
 </shuffle_scheme>
</Shuffle_Input>
```

Each of these blocks are explained in dedicated sections in the following.

11.1 *materials*

In this block (***Block name***: **`<materials>`**) a list of materials to be used in any of the shuffling schemes can be inputted.
This XML block can/must contain the following XML attributes:

N_mat:

Description: the number of materials that are inputted

Data type : integer

Required Entry

The XML sub-nodes contained by the **`<materials>`** are listed in the following sub-sections.

11.1.1 *new*

In this XML block (***Block name***: **`<new>`**) all the new materials are inputted within **`<mat>`** XML nodes. The syntax and information to be specified in the **`<mat>`** are equivalent to the ones reported in section 7.6 and, consequentially, are not repeated here.

11.1.2 *predefined*

In this XML block (***Block name***: **`<predefined>`**) all the materials pre-defined in the **`<DENSITIES>`** XML input (see 7) can be listed. **Note:** The materials are listed as integer IDs.

Description: it contains the list of the the MaterialID that will be used

Data type : integer array

Default value : None

11.2 *assemblies*

This optional XML block (***Block name***: **`<assemblies>`**) is aimed to group cells/material zones. For example, it can be used to input the IDs (see section 5.3) that should be grouped together to

represent an assembly.
This XML block can/must contain the following XML attributes:

N_assemblies:

Description: the number of assemblies (groups) that are going to be inputted

Data type : integer

Required Entry

The XML sub-nodes contained by the **`<assemblies>`** XML node are listed in the following sub-sections.

11.2.1 *assembly*

Block name: **`<assembly>`**

Description: it contains the list of the the MaterialID that will be grouped in one "assembly"

Data type : integer array

Default value : None

This XML block must contain the following XML attributes:

id:

Description: a unique identifier associated with this assembly

Data type : integer

Required Entry

type:

Description: the type of "assembly". This attribute will determine the way rotations are performed in case of schemes that require them.

Data type : string

Required Entry

Selection : Cartesian/Hexagonal – Cartesian geometry/Hexagonal geometry

Note: if the model is neither in Cartesian nor in Hexagonal geometry, no rotation is possible but the assembly node can still be used to group ids. In such case, it is suggested the type to be "Cartesian".

11.3 *shuffle_scheme*

In this XML block (***Block name***: **`<shuffle_scheme>`**) multiple shuffling schemes can be inputted. These schemes can be used at the end or begin of any depletion cycle, as reported in section 10.3.11.
This XML block can/must contain the following XML attribute:

N_schemes:

Description: the number of schemes are going to be inputted

Data type : integer

Required Entry

Only a type of XML sub-node can be contained in the **`<shuffle_schemes>`** XML node: **`<scheme>`**. A number of **`<scheme>`** equal to the number indicated in the attribute *N_schemes* must be inputted.

11.3.1 *scheme*

In this XML block (***Block name***: **`<scheme>`**) a shuffling scheme (strategy) is inputted.
This XML block can/must contain the following XML attributes:

id:

Description: the unique identifier associated with this scheme (name of the scheme)

Data type : string

Required Entry

N_move:

Description: the number of movements are going to be inputted

Data type : integer

Required Entry

Only a type of XML sub-node can be contained in the **`<shuffle_schemes>`** XML node: **`<move>`**. A number of **`<move>`** equal to the number indicated in the attribute *N_move* must be inputted.

11.3.1.1 *move*

In this XML sub-block (***Block name***: `<move>`) a single move must be inputted. A "move" represents an action to be performed at the end or begin of an operational cycle. For example, it can represent the movement of assemblies from the periphery to the center of the reactor or from the reactor to the discharge (decay) pool.
This XML block can/must contain the following XML attributes:

from:

Description: location from where the movement is initiated

Data type : string

Required Entry

Selection : core/new/pool – in-core/new fresh materials/decay pool

to:

Description: "landing" location

Data type : string

Required Entry

Selection : core/out/pool – in-core/discharge/decay pool

Within the `<move>` XML node, 2 different couple of XML sub-nodes can be inputted:

If no `<assemblies>` have been inputted:

`<from>`:

Description: cell IDs (see MaterialID) from where the movement is initiated
Data type : integer array
Required Entry

`<to>`:

Description: "landing" cell IDs (see MaterialID)
Data type : integer array
Note: : **Only available when the attribute *to* in the `<move>` node is *core***

If `<assemblies>` have been inputted:

`<from_assembly>`:

Description: assembly IDs (see 11.2) from where the movement is initiated

Data type : integer array

Required Entry

`<to_assembly>`:

Description: “landing” assembly IDs (see 11.2)

Data type : integer array

Note: : **Only available when the attribute *to* in the `<move>` node is *core***

In this case, the user can specify rotations (in the `<from_assembly>` node) for the assemblies when moved. In order to do so, the rotation degrees can be appended, with the $+$ sign, next to the assembly ID; for example, if a rotation of $90 and 180 degrees$ needs to be applied on the assemblies $81 and 75$ respectively, the `<from_assembly>` node would look like as follows:

```
<Shuffle_Input>
  <shuffle_scheme N_schemes="1">
    <scheme id="fullCore" N_move="3">
      ...
      <move from="core" to="core">
        <from_assembly> 81+90 75+180 </from_assembly>
        <to_assembly> 102 25 </to_assembly>
      </move>
      ...
    </scheme>
  </shuffle_scheme>
</Shuffle_Input>
```

12 SCALING_LIBRARY

In the **<scaling_library>** XML input/main block, the user can specify scaling factors for the neutron cross sections. This XML input is used mainly for Uncertainty Quantification analyses but can be freely used for any scaling purposes. The **<scaling_library>** XML block can/must contain the following XML attributes:

print_xml:

Description: logical flag to activate the printing of the scaled cross sections in an output xml file (named "scaled_xs.xml").

Data type : logical

Default value : F

Selection : T/F – Print/Not Print

Two examples of the this block is reported below:

- No tabulation example:

```
<scaling_library print_xml="t">
    <set>
        <library lib_name="fuel1">
            <isotope id="U235" type="multiplier">
                <FissionXS g="1">1.00</FissionXS>
                <KappaXS g="1">1.00</KappaXS>
                <NuFissionXS g="1">1.00</NuFissionXS>
                <ScatteringXS g="1">1.00</ScatteringXS>
                <n2nXS g="1">1.339</n2nXS>
                <npXS g="1">1.001</npXS>
                <nalphaXS g="1">1.001</nalphaXS>
            </isotope>
            <isotope id="BP" type="absolute">
                <n2nXS g="1␣2">2.222 4.444</n2nXS>
                <ngXS g="2">1.21E+6</npXS>
                <nalphaXS g="1">1.001</nalphaXS>
            </isotope>
        </library>
        <library lib_name="fuel3noBP">
            <isotope id="U235" type="multiplier">
                <FissionXS g="1">1.000</FissionXS>
                <KappaXS g="1">1.000</KappaXS>
                <NuFissionXS g="1">1.000</NuFissionXS>
            </isotope>
            <isotope id="U238" type="multiplier">
                <ngXS g="1">1.001</ngXS>
            </isotope>
            <isotope id="BP" type="additive">
                <ngXS g="1">2000</ngXS>
            </isotope>
        </library>
    </set>
</scaling_library>
```

- Tabulation example:

```
<scaling_library print_xml="t">
    <set>
        <tab name="fuel_temperature">900.</tab>
        <tab name="mod_temperature">500.</tab>
        <library lib_name="fuel1">
            <isotope id="U235" type="multiplier">
                <FissionXS g="1">1.00</FissionXS>
                <KappaXS g="1">1.00</KappaXS>
                <NuFissionXS g="1">1.00</NuFissionXS>
                <ScatteringXS g="1">1.00</ScatteringXS>
                <n2nXS g="1">1.339</n2nXS>
                <npXS g="1">1.001</npXS>
                <nalphaXS g="1">1.001</nalphaXS>
            </isotope>
            <isotope id="BP" type="absolute">
                <n2nXS g="1␣2">2.222 4.444</n2nXS>
                <ngXS g="2">1.21E+6</npXS>
                <nalphaXS g="1">1.001</nalphaXS>
            </isotope>
        </library>
        <library lib_name="fuel3noBP">
            <isotope id="U235" type="multiplier">
                <FissionXS g="1">1.000</FissionXS>
                <KappaXS g="1">1.000</KappaXS>
                <NuFissionXS g="1">1.000</NuFissionXS>
            </isotope>
            <isotope id="U238" type="multiplier">
                <ngXS g="1">1.001</ngXS>
            </isotope>
            <isotope id="BP" type="additive">
                <ngXS g="1">2000</ngXS>
            </isotope>
        </library>
    </set>
     <set>
        <tab name="fuel_temperature">600.</tab>
        <tab name="mod_temperature">400.</tab>
        <library lib_name="fuel1">
            <isotope id="U235" type="multiplier">
                <FissionXS g="1">1.00</FissionXS>
                <KappaXS g="1">1.00</KappaXS>
                <NuFissionXS g="1">1.00</NuFissionXS>
                <ScatteringXS g="1">1.00</ScatteringXS>
                <n2nXS g="1">1.339</n2nXS>
                <npXS g="1">1.001</npXS>
                <nalphaXS g="1">1.001</nalphaXS>
            </isotope>
            <isotope id="BP" type="absolute">
                <n2nXS g="1␣2">2.222 4.444</n2nXS>
                <ngXS g="2">1.21E+6</npXS>
                <nalphaXS g="1">1.001</nalphaXS>
            </isotope>
        </library>
        <library lib_name="fuel3noBP">
            <isotope id="U235" type="multiplier">
                <FissionXS g="1">1.000</FissionXS>
                <KappaXS g="1">1.000</KappaXS>
                <NuFissionXS g="1">1.000</NuFissionXS>
            </isotope>
            <isotope id="U238" type="multiplier">
                <ngXS g="1">1.001</ngXS>
            </isotope>
            <isotope id="BP" type="additive">
                <ngXS g="1">2000</ngXS>
            </isotope>
        </library>
    </set>
</scaling_library>
```

Each of these blocks are explained in dedicated sections in the following.

12.1 *set*

In this block (***Block name***: **`<set>`**), the user will specify the scaling factors to apply to different libraries. The user can input as many **`<set>`** nodes as needed. If no **`<tab>`** XML nodes are inputted with the **`<set>`**, the scaling factors will be applied to all the tabulation coordinates of the neutron cross section grid. The **`<set>`** can/must contain 2 XML nodes' types: **`<library>`** and **`<tab>`**. A detailed description is reported in the following subsections.

12.1.1 *tab*

In this XML block (***Block name***: **`<tab>`**) the information for describing the initial tabulation coordinate (s) associated with this **`<set>`** is reported. This XML block does not need to be inputted in case the scaling factors need to be applied to entire tabulation grid of the associated Neutron Cross Sections.

Block name: **`<tab>`**

Description: tabulation coordinate

Data type : float

This XML block must contain the following XML attribute:

name:

Description: name of the tabulation dimension

Data type : string **Required Entry**

12.1.2 *library*

This XML block (***Block name***: **`<library>`**) is the container of the scaling factors for a specific cross section library. The user can specify as many **`<library>`** as needed.
This XML node must contain the following XML attributes:

lib_name:

Description: the name of the library these scaling factors belong to

Data type : string

Required Entry

Within this XML node as many **`<isotope>`** nodes as needed can be inputted.

12.1.2.1 isotope

This XML block (***Block name***: `<isotope>`) is the container of the scaling factors for a specific isotope within a specific `<library>`. The user can specify as many `<isotope>` nodes as needed.
This XML node must contain the following XML attributes:

id:

Description: the name of the isotope the scaling factors belong to

Data type : string

Required Entry

type:

Description: the type of scaling to be performed. Three methods are available (“multiplier”, “absolute”,“additive”)

Data type : string

Required Entry

Selection : multiplier/absolute/additive – Multiplicative factors/replace the xs value/add the value to the xs value

Within the `<isotope>` the following sub-nodes can be inputted:

`<FissionXS>`, Fission Cross Section scaling;

`<KappaXS>`, Kappa Cross Section scaling;

`<NuFissionXS>`, Nu Fission Cross Section scaling

`<ScatteringXS>`, Total Cross Section scattering scaling

`<n2nXS>`, n, 2n Cross Section scaling

`<npXS>`, n,p Cross Section scaling

`<nalphaXS>`, n, alpha Cross Section scaling

`<ngXS>`, n, gamma Cross Section scaling

Each of the above sob-nodes must contain the following attribute:

g:

Description: the group number this scaling factor belongs to

Data type : integer

Required Entry

13 SPH_CONTROLLER

In the `<sph_controller>` XML input/main block, the user can specify the settings to perform the calculations of the SPH factors. This input is used for this purpose only. At the end of the SPH calculations, an output file that can be used as input for the PHISICS simulation is produced (see section 14) The `<sph_controller>` XML block can/must contain the following XML attributes:

max_iter:

Description: maximum number of iterations to be performed

Data type : integer

Default value : 30

Selection : any integer > 1

An example of the this block is reported below:

```
<sph_controller max_iter="10">
    <!-- iteration schemes available are newton or successive
       substitution -->
    <iteration_scheme>substitution</iteration_scheme>
    <!-- diffusion equivalence (diffusion) or transport
       equivalence (transport) -->
    <sph_type>diffusion</sph_type>
    <!-- strategy:
         - op  -> operational (compute SPH on operational point
            (coming from Material file) and apply the
                 correction factors to all tabulation points
         - all -> compute SPH for each tabulation point
            separately
     -->
    <strategy>op</strategy>
    <!-- expand_library:
         - f (or false) if the library is associated to multiple
            nodes, an average SPH is computed
         - t (or true)  if the library is associated to multiple
            nodes, a local SPH is performed and the
           number of libraries get expanded
     -->
    <!-- convergence criterion -->
    <tolerance>1.0e-1</tolerance>
</sph_controller>
```

Each of these blocks are explained in dedicated sections in the following.

13.1 *iteration_scheme*

Block name: `<iteration_scheme>`

Description: the type of iteration scheme to be used

Data type : string

Default value : substitution

Selection : substitution/newton – successive substitution/newton scheme

13.2 *sph_type*

Block name: `<sph_type>`

Description: the type of sph equivalence to use

Data type : string

Default value : diffusion

Selection : diffusion/transport – diffusion equivalence/transport equivalence

13.3 *strategy*

Block name: `<strategy>`

Description: the type of strategy to use in case of tabulated cross sections (and fluxes).

Data type : string

Default value : op

Selection : op/all – operational (compute SPH on operational point (coming from Material file) and apply the correction factors to all tabulation points/compute SPH for each tabulation point separately

13.4 *tolerance*

Block name: **`<tolerance>`**

Description: the converge tolerance to stop the iterations

Data type : float

Required Entry

14 SPH_FACTORS

In the **<sph_factors>** XML input/main block, the user can specify the SPH factors to use as correction for a specific neutron cross section library. This XML input/block is generated by the SPH calculation module but it can be provided by the user directly (if the SPH factors have been externally generated with an equivalent Transport solver/formulation). The **<sph_factors>** XML block can/must contain the following XML attributes:

n_sph_sets:

Description: number of sph sets (correction sets)

Data type : integer

Required Entry

n_library_sets:

Description: number of sph library sets

Data type : integer

Required Entry

An example of the this block is reported below:

```
<sph_factors n_sph_sets="6" n_library_sets="1">
  <library_set_info ID="1">
    <tab_values N_tab="3">
      <tab name="T_fuel">450.0 1234.0</tab>
      <tab name="BURN-UP">0.0</tab>
      <tab name="T_mod">400.0 972.0</tab>
    </tab_values>
  </library_set_info>
  <library_set ID="1">
    <!--
      here all the tapulated sph factors are stored.
      The number of points must correspond to the total
      combinations of tabulation points.
     -->
  <POINT ID="1">
      <tab name="T_fuel">450.0</tab>
      <tab name="BURN-UP">0.0</tab>
      <tab name="T_mod">400.0</tab>
      <sph_set ID="fuel1">
        <mu_g>0.99 1.01 0.99 1.01 0.99 1.01 0.99 1.01 0.99
           1.01</mu_g>
      </sph_set>
      <sph_set ID="fuel2noBP">
          <mu_g>0.99 1.01 0.99 1.01 0.99 1.01 0.99 1.01 0.99
             1.01</mu_g>
      </sph_set>
      <sph_set ID="refl">
          <mu_g>0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8</mu_g>
      </sph_set>
  </POINT>
  <POINT ID="2">
      <tab name="T_fuel">1234.0</tab>
      <tab name="BURN-UP">0.0</tab>
      <tab name="T_mod">400.0</tab>
      <sph_set ID="fuel1">
          <mu_g>0.99 1.01 0.99 1.01 0.99 1.01 0.99 1.01 0.99
             1.01</mu_g>
      </sph_set>
      <sph_set ID="fuel2noBP">
          <mu_g>0.99 1.01 0.99 1.01 0.99 1.01 0.99 1.01 0.99
             1.01</mu_g>
      </sph_set>
      <sph_set ID="refl">
          <mu_g>0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8</mu_g>
      </sph_set>
  </POINT>
  <POINT ID="3">
      <tab name="T_fuel">450.0</tab>
      <tab name="BURN-UP">0.0</tab>
      <tab name="T_mod">972.0</tab>
      <sph_set ID="fuel1">
          <mu_g>0.99 1.01 0.99 1.01 0.99 1.01 0.99 1.01 0.99
             1.01</mu_g>
      </sph_set>
      <sph_set ID="fuel2noBP">
          <mu_g>0.99 1.01 0.99 1.01 0.99 1.01 0.99 1.01 0.99
             1.01</mu_g>
      </sph_set>
      <sph_set ID="refl">
          <mu_g>0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8</mu_g>
      </sph_set>
  </POINT>
  <POINT ID="4">
      <tab name="T_fuel">1234.0</tab>
      <tab name="BURN-UP">0.0</tab>
      <tab name="T_mod">972.0</tab>
      <!-- the number of SPH sets does not need to match the
         number of materials -->
      <sph_set ID="fuel1">
          <mu_g>0.99 1.01 0.99 1.01 0.99 1.01 0.99 1.01 0.99
             1.01</mu_g>
      </sph_set>
      <sph_set ID="fuel2noBP">
          <mu_g>0.99 1.01 0.99 1.01 0.99 1.01 0.99 1.01 0.99
             1.01</mu_g>
      </sph_set>
      <sph_set ID="refl">
          <mu_g>0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8</mu_g>
      </sph_set>
  </POINT>
 </library_set>
</sph_factors>
```

Each of these blocks are explained in dedicated sections in the following.

14.1 library_set_info

In this block the information about the library set(s) (sph factor sets) that will be inputted in the **<library_set>**(s) are listed. This XML block is in charge to define the information about the tabulation grid common to all the sph libraries contained in the associated library set.
This XML block can/must contain the following XML attributes:

ID:

Description: unique identifier of this library set info block. This ID will need to match the ID specified in the corresponding **<library_set>** xml block;

Data type : integer

Required Entry

Within each **<library_set_info>** node, a single sub-node must be inputted, explained in the following sub-section.

14.1.1 tab_values

The **`<tab_values>`** XML block can/must contain the following XML attributes:

N_tab:

Description: number of tabulation dimensions belonging to this **`<library_set>`** (**`ID`**)

Data type : integer

Required Entry

Within the **`<tab_values>`** block the user must input *N_tab* **`<tab>`** XML nodes:

14.1.1.1 tab

Description: contains the tabulation points (values) for a specific tabulation dimension

Data type : float array

Required Entry

Note: The values here reported must correspond to the ones specified for each **`<POINT>`** in the **`<library_set>`** associated to this **`<library_set_info>`** **`ID`**.

Within the **`<tab>`** block the user must input the following attribute:

name:

Description: the tabulation name.

Data type : string

Required Entry

14.2 library_set

In this block the library set(s) (sph factor sets) are inputted. This XML block is in charge to contain tabulated sph factors' libraries (sph sets).
This XML block can/must contain the following XML attributes:

ID:

Description: unique identifier of this library set info block. This ID will need to match the ID specified in the corresponding **`<library_set_info>`** xml block;

Data type : integer

Required Entry

Within each **`<library_set>`** node, a number of **`<POINT>`** equal to the total number of combinations of the tabulation dimensions need to be inputted. For example, in the previous example 4 **`<POINT>`** nodes must be inputted (total number of coordinates in the established tabulation grid).

14.2.1 POINT

The **`<POINT>`** contains the sph sets belonging to a specific coordinate in the tabulation grid. The **`<POINT>`** XML block can/must contain the following XML attributes:

ID:

Description: unique identifier for this POINT

Data type : integer

Required Entry

Within the **`<POINT>`** block the user must input *N_tab* **`<tab>`** XML nodes and a user-dependent **`<sph_set>`** containing the sph factors

14.2.1.1 *tab*

In this XML block (***Block name***: **`<tab>`**) the information for describing the tabulation coordinate (s) associated with this **`<POINT>`** is reported. This XML block does not need to be inputted in case the associated Neutron Cross Sections are not tabulated.

Description: coordinate (value) of this tabulation

Data type : float

Required Entry

This XML block must contain the following XML attributes:

name:

Description: name of the tabulation dimension

Data type : string

Required Entry

14.2.1.2 *sph_set*

In this XML block (***Block name***: **`<sph_set>`**) the sph set belonging to this specific coordinate (**`<POINT>`**) is inputted. This XML block must contain the following XML attributes:

ID:

Description: unique ID of this sph set

Data type : string

Required Entry

Note: This unique identifier is the one that can be inputted in the **`<DENSITIES>`** **`<mat>`** nodes to associate a set to a specific material/zone (see section 7.6)

Within this XML node a single node must be inputted:

`<mu_g>`:

Description: the sph factors (group-wise)

Data type : float array

Required Entry

Note: a float array of size number of energy groups must be inputted

15 PHISICS_AD

In the **<PHISICS_AD>** XML input/main block, the user can specify the settings to control some additional features of the PHISICS code when coupled with RELAP5-3D or export some specific quantities in case of standalone cases. Each of the entries here reported will be taged with 2 different keywords:

- R5, when the option is available in coupled mode with RELAP5-3D only;
- PHISICS, when the option is available in PHISICS standalone only;
- PR, when the option is available both in coupled mode with RELAP5-3D and in PHISICS standalone.

An example of the this block is reported below:

```
<PHISICS_AD>
   <xs_tol>0.1</xs_tol>
   <BURN_TH>T</BURN_TH>
   <TH_between_BURN> 10  </TH_between_BURN>
   <VTK_isotopes>XE135 PU239 PU240 PU241</VTK_isotopes>
   <VTK_tabulations>fuel_temperature
      mod_temperature</VTK_tabulations>
   <VTK_pow>T</VTK_pow>
   <VTK_rpow>F</VTK_rpow>
   <VTK_pow_dens>F</VTK_pow_dens>
   <VTK_rpow_dens>F</VTK_rpow_dens>
   <TH_VTK_time></TH_VTK_time>
   <TH_VTK_time_mult>3 5</TH_VTK_time_mult>
   <IX>1 2 2</IX>
   <IY>2 2 2</IY>
   <IZ>2 2 2 2</IZ>
   <MRTAU_print>T</MRTAU_print>
   <print_macro_xml>T</print_macro_xml>
   <Density_out>10 23 47</Density_out>
   <BURN_TH>T</BURN_TH>
   <qs_on>F</qs_on>
   <qs_dt_skip>10</qs_dt_skip>
   <qs_print>T</qs_print>
   <mixer_input>mat.xml</mixer_input>
   <instant_input>inp.xml</instant_input>
   <criticality_input>mat.xml</criticality_input>
   <library_input>lib.xml</library_input>
   <depletion_input>dep.xml</depletion_input>
   <macro_library>macro.xml</macro_library>
   <radial_operation operator="average" exclude="1-17-18">
     BURN-UP
    </radial_operation>
    <radial_operation operator="average" exclude="1-17-18">
     fuel_temperature
    </radial_operation>
   <PHISICS_off>100.0</PHISICS_off>
</PHISICS_AD>
```

Each of these blocks are explained in dedicated sections in the following.

15.1 *xs_tol*

Block name: **<xs_tol>**

Description: For steady state calculations, INSTANT is skipped if the relative cross section change between the last INSTANT execution and the current thermal-hydraulics iteration is less that xs_tol. Default is 0.0, i.e. INSTANT is called at every thermal-hydraulics iteration.

Data type : float

Default value : 0.0

AVAILABILITY : **R5**

15.2 *BURN_TH*

Block name: **<BURN_TH>**

Description: True if multiple iterations between thermal- hydraulics and burning are required. This enables the “Depletion time evolution” mode described in sec. 10. RELAP5-3D has to be run in steady-state mode. **<TH_between_BURN>** has to be given. Cannot be used with **<skip_dpl>** equal true.

Data type : logical

Default value : False

AVAILABILITY : **R5**

15.3 *TH_between_BURN*

Block name: `<TH_between_BURN>`

Description: Gives the thermal hydraulic times at which burning is required. `<BURN_TH>` has to be true. For a new problem, the calculation sequence is:

1. Pre-burning at time zero (the pre-burning cannot be skipped)
2. TH
3. burning
4. TH
5. etc.

The calculation always ends with thermal- hydraulics, not with burning. Therefore, the number of thermal hydraulic times must be one less than burning steps in the DEPLETION input (keyword `<n_tab_interval>`). For a restart calculation, the calculation sequence is

1. TH
2. burning
3. TH
4. etc.

Therefore the number of thermal hydraulic times must be equal than burning steps in the MRTAU input (keyword `<n_tab_interval>`).

Data type : float array

Default value : empty array

AVAILABILITY : **R5**

15.4 *VTK_isotopes*

Block name: `<VTK_isotopes>`

Description: array of isotopes for which a VTK output file needs to be generated on the nodalization geometry.

Data type : string array

Default value : None

AVAILABILITY : **PR**

15.5 *VTK_tabulations*

Block name: `<VTK_tabulations>`

Description: array of tabulation dimensions for which a VTK output file needs to be generated on the nodalization geometry.

Data type : string array

Default value : None

AVAILABILITY : **PR**

15.6 *VTK_pow*

Block name: `<VTK_pow>`

Description: True if the power needs to be exported in a VTK output file generated on the nodalization geometry.

Data type : logical

Default value : False

AVAILABILITY : **PR**

15.7 *VTK_rpow*

Block name: `<VTK_rpow>`

Description: True if the relative power needs to be exported in a VTK output file generated on the nodalization geometry.

Data type : logical

Default value : False

AVAILABILITY : **PR**

15.8 *VTK_pow_dens*

Block name: `<VTK_pow_dens>`

Description: True if the power density needs to be exported in a VTK output file generated on the nodalization geometry.

Data type : logical

Default value : False

AVAILABILITY : **PR**

15.9 *TH_VTK_time*

Block name: `<TH_VTK_time>`

Description: time intervals for VTK output. `<TH_VTK_time_mult>` has to be given if `<TH_VTK_time>` is used. The time intervals can be used for steady state and transient calculations. Multiple time intervals can be used in conjunction with `<TH_VTK_time_mult>`, e.g. print the VTK every 10 seconds for the first 40 seconds, then print VTK all 5 seconds.

Data type : float array

Default value : print at every time step.

AVAILABILITY : **R5**

15.10 *TH_VTK_time_mult*

Block name: `<TH_VTK_time_mult>`

Description: multiplier for time intervals for VTK output. `<TH_VTK_time>` has to be given if `<TH_VTK_time_mult>` is used. The number of time multipliers given has to be the same as the number of `<TH_VTK_time>`. The first `<TH_VTK_time>` interval will be used the number of times indicated by the first `<TH_VTK_time_mult>`, then the second `<TH_VTK_time>` will be used the number of times given in the second `<TH_VTK_time_mult>` and so on. Zero as a multiplier indicates infinite times, i.e. zero has to be used for the last multiplier.

Data type : float array

Default value : empty array

AVAILABILITY : **R5**

15.11 *radial_operation*

Block name: **`<radial_operation>`**

Description: Tabulation dimension on which a radial operation (e.g. integral, average, etc.) needs to be performed. This option will perform the operation and output the results in a CSV output file.

Data type : string

Default value : None

AVAILABILITY : **PR**

This XML block must/can contain the following XML attributes:

operator:

Description: the operator to apply

Data type : string

Selection : average/integral – radial volumetric average/ radial volumetric integral **Required Entry**

exclude:

Description: axial layers that should be excluded by this operation

Data type : dash separated array

Default value : None

15.12 *IX*

Block name: **`<IX>`**

Description: This is IX from the **`<Geometry>`** block in INSTANT input (see sec. 5.3). Since the **`<Geometry>`** block from INSTANT is not accessible in the RELAP5-3D coupled with PHISICS, IX can be given in the **`<PHISICS_AD>`** block. It represents the subdivision of intervals in x-direction

Data type : integer array

Default value : 1

AVAILABILITY : **R5**

15.13 *IY*

Block name: **`<IY>`**

Description: This is IY from the **`<Geometry>`** block in INSTANT input (see sec. 5.3). Since the **`<Geometry>`** block from INSTANT is not accessible in the RELAP5-3D coupled with PHISICS, IY can be given in the **`<PHISICS_AD>`** block. It represents the subdivision of intervals in y-direction

Data type : integer array

Default value : 1

AVAILABILITY : **R5**

15.14 *IZ*

Block name: **`<IZ>`**

Description: This is IZ from the **`<Geometry>`** block in INSTANT input (see sec. 5.3). Since the **`<Geometry>`** block from INSTANT is not accessible in the RELAP5-3D coupled with PHISICS, IZ can be given in the **`<PHISICS_AD>`** block. It represents the subdivision of intervals in z-direction

Data type : integer array

Default value : 1

AVAILABILITY : **R5**

15.15 *MRTAU_print*

Block name: **`<MRTAU_print>`**

Description: Controls the MRTAU printing in transients. If true, the full MRTAU output (densities, burnup, etc. for each material) is printed for each time-step in the RELAP5-3D output file.

Data type : logical

Default value : False

AVAILABILITY : **R5**

15.16 *print_macro_xml*

Block name: `<print_macro_xml>`

Description: If true, the MACROSCOPIC cross sections for each Transport solve are printed in a XML output file

Data type : logical

Default value : False

AVAILABILITY : **R5**

15.17 *Density_out*

Block name: `<Density_out>`

Description: Array of node numbers. The materials for the given node numbers will be printed in file "Density_out-(processor number)" for every material update. During steady state calculations, the mixer materials are printed in sequence for each iteration without a delimiting indication. For transients, the mixer materials are printed for each time step. In the transient case, the problem time is printed for each time step before the mixer materials output. The mixer material information contains burnup, tabulation point and densities for all isotopes in the node.

Data type : integer array

Default value : None

AVAILABILITY : **R5**

15.18 *mixer_input*

Block name: `<mixer_input>`

Description: filename for the material description input. Only used if phis_mi cross section option is used.

Data type : string

Default value : mat.xml

AVAILABILITY : **R5**

15.19 *instant_input*

Block name: `<instant_input>`

Description: filename for the instant description input.

Data type : string

Default value : inst_cont.xml

AVAILABILITY : **R5**

15.20 *criticality_input*

Block name: `<criticality_input>`

Description: filename for criticality search settings. Only used if phis_mi cross section option is used.

Data type : string

Default value : mat.xml

AVAILABILITY : **R5**

15.21 *library_input*

Block name: `<library_input>`

Description: filename for MICROSCOPIC cross section library input. Only used if phis_mi cross section option is used.

Data type : string

Default value : lib.xml

AVAILABILITY : **R5**

15.22 *depletion_input*

Block name: **`<depletion_input>`**

Description: filename for DEPLETION input. Only used if phis_mi cross section option is used.

Data type : string

Default value : dep.xml

AVAILABILITY : **R5**

15.23 *macro_library*

Block name: **`<macro_library>`**

Description: filename for the PHISICS-macroXS file containing the macroscopic cross sections for the phis_ma option.

Data type : string

Default value : phisics_xs.xml

AVAILABILITY : **R5**

15.24 *PHISICS_off*

Block name: **`<PHISICS_off>`**

Description: thermal hydraulic time at which the kinetic calculation is switched off. After this time, PHISICS is not called anymore and the fission power is set to zero.

Data type : float

Default value : infinity, i.e. kinetics is always on

AVAILABILITY : **R5**

15.25 *qs_on*

Block name: `<qs_on>`

Description: Flag to turn on the quasi static approach.

Data type : logical

Default value : F

AVAILABILITY : **PR**

15.26 *qs_dt_skip*

Block name: `<qs_dt_skip>`

Description: Frequency of new shape calculation: i.e. Number of time steps to be skipped before a new shape calculation is performed in the quasi static approach.

Data type : integer

Default value : 0

AVAILABILITY : **PR**

15.27 *qs_print*

Block name: `<qs_print>`

Description: Print the quasi-static quantities computed (E.g. reactivity, etc.)?

Data type : logical

Default value : False

AVAILABILITY : **PR**

16 Execution Schemes and Input requirements

16.1 Overview

In the previous sections several main XML/input blocks have been reported and explained. Depending on the execution that needs to be performed, different inputs/XML main blocks need (mandatory or optional) to be provided.
The compilation of the PHISICS software will generate multiple executables that require or can accept different inputs:

- INSTANT standalone executable (Name: ***instant***), used for eigenvalue and source standalone problems;
- MRTAU standalone executable (Name: ***mrtau_standalone***), used for depletion and burn-up standalone calculations;
- Time Integrator standalone executable (Name: ***time_dep***), used for time-dependent standalone simulations;
- SPH calculation executable (Name: ***spu_run***), used to compute SPH factors usable by any other PHISICS module;
- Depletion Evolution executable (Name: ***dpl_instant_run***), main PHISICS driver (Depletion, Transport, Fuel Management, Criticality Search, etc.)

In the following section, the input requirements of each of this executables will be reported

16.2 INSTANT standalone

The instant standalone executable is aimed to allow for Diffusion/Transport Eigenvalue and Source problems' execution. In order to execute this module, a single XML input file must be provided with the information required in the following XML main nodes:

- **REQUIRED:** **`<INSTANT_INPUTS>`**, detailed in section 5

The module can be executed as follows:

```
instant -i instant_in.xml out.outp
```

where:

- ***instant_in.xml*** is any user-defined file name containing the **<INSTANT_INPUTS>** block. Default: ***inp.xml***
- ***out.outp*** is any user-defined file name containing the output file
  Default: ***INSTANT.outp***

16.3 MRTAU standalone

The MRTAU standalone executable is aimed to allow standalone 0-D Burn-Up, Decay Heat and Transmutation problems' execution. In order to execute this module, a single XML input file must be provided (with the additional Nuclear Data Files) with the information required in the following XML main nodes:

- **REQUIRED: <DEPLETION_INPUT>**, detailed in section 10

The module can be executed as follows:

```
mrtau_standalone depletion_input_file.xml
```

where:

- ***depletion_input_file.xml*** is any user-defined file name containing the **<DEPLETION_INPUT>** block.
  Default: ***dep.xml***

16.4 Time Integrator standalone

The Time Integrator standalone executable is aimed to allow standalone neutron transport/diffusion time dependent problems' execution. In order to execute this module, the following XML input files must/can be provided with the information required in the following XML main nodes:

- **REQUIRED: <INSTANT_INPUTS>**, detailed in section 5
- **REQUIRED: <INSTANT_TIME_DEP>**, detailed in section 6
- **OPTIONAL: <XS-LIBRARY>**, detailed in section 9
- **OPTIONAL: <DENSITIES>**, detailed in section 7

In case only the required input files/ XML nodes are provided, the calculation is performed using the MACROscopic cross sections provided in the **<INSTANT_INPUTS>** XML blocks. If MICROscopic/Tabulated cross sections can be used, the **<XS-LIBRARY>** and **<DENSITIES>** need to be provided.

The module can be executed as follows:

```
time_dep -i instant_in.xml -mat mats.xml -xs xss.xml -o out.outp
```

where:

- ***instant_in.xml*** is any user-defined file name containing the **<INSTANT_INPUTS>** block.
  Default: ***inp.xml***
- ***out.outp*** is any user-defined file name containing the output file
  Default: ***INSTANT.outp***
- ***mats.xml*** is any user-defined file name containing the **<DENSITIES>** block.
  Default: ***mat.xml***
- ***xss.xml*** is any user-defined file name containing the **<XS-LIBRARY>**block.
  Default: ***lib.xml***

16.5 SPH calculation

The SPH executable is aimed to allow the computations of SPH factors for later usage in the other PHISICS modules. In order to execute this module, the following XML input files must/can be provided with the information required in the following XML main nodes:

- **REQUIRED:** **<sph_controller>**, detailed in section 13
- **REQUIRED:** **<INSTANT_INPUTS>**, detailed in section 5
- **REQUIRED:** **<XS-LIBRARY>**, detailed in section 9
- **REQUIRED:** **<DENSITIES>**, detailed in section 7

In case only the required input files/ XML nodes are provided, the calculation is performed using the MACROscopic cross sections provided in the **<INSTANT_INPUTS>** XML blocks. If MICROscopic/Tabulated cross sections can be used, the **<XS-LIBRARY>** and **<DENSITIES>** need to be provided.

The module can be executed as follows:

```
sph_run -sph sph_control.xml -i instant_in.xml
                              -mat mats.xml -xs xss.xml -o out.outp
```

where:

- ***sph_control.xml*** is any user-defined file name containing the **`<sph_controller>`** block.
  Default: ***sph.xml***
- ***instant_in.xml*** is any user-defined file name containing the **`<INSTANT_INPUTS>`** block.
  Default: ***inp.xml***
- ***out.outp*** is any user-defined file name containing the output file
  Default: ***INSTANT.outp***
- ***mats.xml*** is any user-defined file name containing the **`<DENSITIES>`** block.
  Default: ***mat.xml***
- ***xss.xml*** is any user-defined file name containing the **`<XS-LIBRARY>`**block.
  Default: ***lib.xml***

16.6 Depletion Evolution (PHISICS) calculation

The Depletion Evolution executable is the main DRIVER of the PHISICS software. It deploys different calculation schemes and options:

1. Diffusion/Transport (MICROscopic/MACROscopic/Mixed Cross Sections)
2. Diffusion/Transport (MICROscopic/MACROscopic/Mixed Cross Sections) + Depletion/Burn-Up and Decay Heat
3. Diffusion/Transport (MICROscopic/MACROscopic/Mixed Cross Sections) + Depletion/Burn-Up and Decay Heat + Criticality Search
4. Diffusion/Transport (MICROscopic/MACROscopic/Mixed Cross Sections) + Depletion/Burn-Up and Decay Heat + Criticality Search + Multi-Cycle operational analysis (Fuel Management)

In order to execute this DRIVER, the following XML input files must/can be provided with the information required in the following XML main nodes:

- **REQUIRED:** **`<INSTANT_INPUTS>`**, detailed in section 5

- **REQUIRED:** **`<XS-LIBRARY>`**, detailed in section 9
- **REQUIRED:** **`<DENSITIES>`**, detailed in section 7
- **REQUIRED:** **`<DEPLETION_INPUT>`**, detailed in section 10
- **OPTIONAL:** **`<criticality_search>`**, detailed in section 8
- **OPTIONAL:** **`<sph_factors>`**, detailed in section 14
- **OPTIONAL:** **`<scaling_library>`**, detailed in section 12
- **OPTIONAL:** **`<Shuffle_Input>`**, detailed in section 11
- **OPTIONAL:** **`<PHISICS_AD>`**, detailed in section 15

The DRIVER can be executed as follows:

```
dpl_instant_run -dep depl.xml -i instant_in.xml
                          -mat mats.xml -xs xss.xml -o out.outp
```

where:

- ***depl.xml*** is any user-defined file name containing the **`<DEPLETION_INPUT>`** block.
  Default: ***dep.xml***
- ***instant_in.xml*** is any user-defined file name containing the **`<INSTANT_INPUTS>`** block.
  Default: ***inp.xml***
- ***out.outp*** is any user-defined file name containing the output file
  Default: ***Dpl_INSTANT.outp***
- ***mats.xml*** is any user-defined file name containing the **`<DENSITIES>`** block.
  Default: ***mat.xml***
- ***xss.xml*** is any user-defined file name containing the **`<XS-LIBRARY>`**block.
  Default: ***lib.xml***

Document Version Information

0729e616f2be9f5b1cea07f218a9882b5c1a1e48 Andrea Alfonsi Thu, 1 Aug 2019 09:18:35 -0600

Idaho National Laboratory