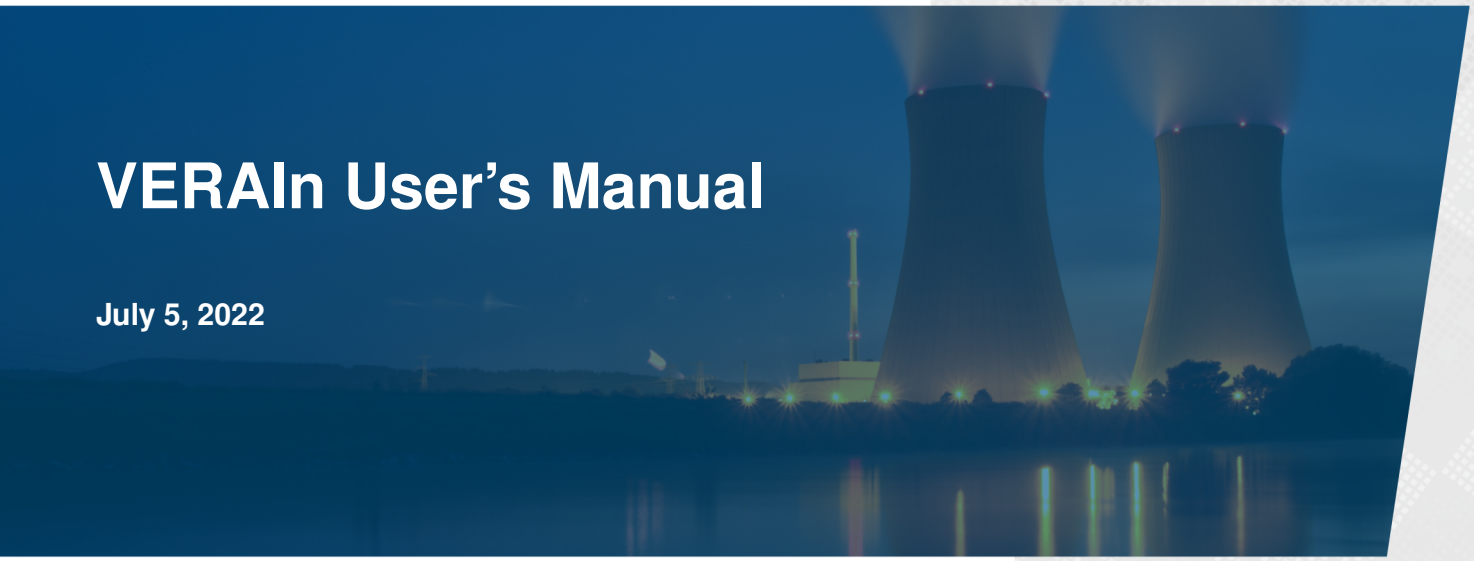
VERA Virtual Environment for Reactor Applications

ORNL/SPR-2022/2509



VERAIn User's Manual

July 5, 2022

Scott Palmtag[1], Andrew Godfrey[2], Mark Baird[2], and Erik Walker[2]

[1]North Carolina State University
[2]Oak Ridge National Laboratory

OAK RIDGE National Laboratory

U.S. DEPARTMENT OF ENERGY

ORNL is managed by UT-Battelle LLC for the US Department of Energy.

ORNL/SPR-2022/2509

VERAIN USER'S MANUAL

Scott Palmtag[1], Andrew Godfrey[2], Mark Baird[2], and Erik Walker[2]

[1]North Carolina State University
[2]Oak Ridge National Laboratory

Date Published: July 5, 2022

Prepared by
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, TN 37831-6283
managed by
UT-Battelle, LLC
for the
US DEPARTMENT OF ENERGY
under contract DE-AC05-00OR22725

VERAIn User's Manual

Revision Log

Revision	Date	Affected Pages	Revision Description
0	6/17/2022	All	New template and card updates for VERA 4.3 release. This version supersedes previous document version CASL-U-2014-0014-005.

Document pages that are:

Export Controlled:	None
IP/Proprietary/NDA Controlled:	None
Sensitive Controlled:	None
Unlimited:	All

VERAIn User's Manual

Approvals:

Erik Walker	08/24/2022
Erik Walker, VERAIO Product Software Manager	**Date**
Aaron Graham	08/24/2022
Aaron Graham, Independent Reviewer	**Date**

CONTENTS

LIST OF FIGURES

ABBREVIATIONS

AOA	axial offset anomaly
CASL	Consortium for Advanced Simulation of Light Water Reactors
BEAVRS	Benchmark for Evaluation And Validation of Reactor Simulations
BOC	beginning of cycle
BWR	boiling water reactor
CFD	computational fluid dynamics
CHF	critical heat flux
CILC	crud-induced localized corrosion
CIPS	crud-induced power shift (also called *axial offset anomaly* [AOA])
CMFD	coarse-mesh finite difference
CTF	COBRA-TF (subchannel code)
DNB	departure from nucleate boiling
EFPD	effective full power day
EOC	end of cycle
GT	guide tube
GWd/MT	gigawatt-days per metric ton heavy metal
HFP	hot full power
HZP	hot zero power
IT	instrument tube
LWR	light-water reactor
MOC	middle of cycle
MOF	mixed oxide fuel
ModSim	modeling and simulation
MWd/MT	megawatt-days per metric ton heavy metal
PCI	pellet–cladding interaction
PCM	per cent mille (10^{-5})
PPM	parts per million (usually boron)
PWR	pressurized water reactor
QA	quality assurance
RIA	reactivity insertion accident
STH	system thermal hydraulic
UQ	uncertainty quantification
VERA	Virtual Environment for Reactor Applications

1. INTRODUCTION

1.1 INTRODUCTION TO CASL

The Consortium for Advanced Simulation of Light Water Reactors (CASL), the first US Department of Energy (DOE) Energy Innovation Hub, was established in July 2010 to provide advanced modeling and simulation (ModSim) solutions for commercial nuclear reactors.

CASL's objective was to predict, with confidence, the performance of nuclear reactors through comprehensive, science-based modeling and simulation technology that would be deployed and applied broadly throughout the nuclear energy industry to enhance safety, reliability, and economics.

CASL's mission was to provide the coupled, high-fidelity, usable modeling and simulation capabilities needed to address light-water reactor (LWR) operational and safety performance–defining phenomena.

CASL's foundational technology products include CASL solutions and CASL ModSim Technologies. CASL's ModSim technology, the Virtual Environment for Reactor Applications (VERA), provides higher fidelity results than those offered by the current industry approach by incorporating coupled physics and science-based models, state-of-the-art numerical methods, modern computational science, integrated uncertainty quantification (UQ) and validation against data from operating pressurized water reactors (PWRs), single-effect experiments, and integral tests.

1.2 VERA

VERA is a specific collection of multiphysics computer codes used to model and simulate depletion of an LWR core over multiple cycles. Examples of the separate physics modeled in the core simulator include cross section generation, neutron transport, isotopic depletion, thermal hydraulics, and fuel performance.

The purpose of the core simulator is to simulate depletion of the reactor core and provide data and boundary conditions to model CASL Challenge Problems such as crud-induced power shift (CIPS), crud-induced localized corrosion (CILC), departure from nucleate boiling (DNB), pellet-cladding interaction (PCI), and reactivity insertion accident (RIA) analyses.

One important feature of the core simulator is that a single common input file is used to drive all of the different physics codes.[1] One benefit of using a single common input is that users only need to understand and be proficient with one input instead of having to understand multiple inputs for multiple physics codes. Another benefit of using a single common input is that all codes work from a single geometry description, and this reduces errors due to inconsistent geometries in different codes.

The most up-to-date version of this document resides in the VERA Git repository file "VERAIO/verain/docs/verain_UM.pdf"; please refer to this location for the latest version of the input manual.

Additional information can be found on the VERA website: https://vera.ornl.gov/.

[1] The only exception to this is for computational fluid dynamics codes, which generally require a detailed CAD file to support mesh generation and perform meaningful analysis.

1.3 MANUAL ORGANIZATION

This manual is organized into three main parts.

The first part, which includes chapters 2 through 4, is the "User's Manual," which describes how a user would set up a typical input. This part of the manual gives the most common input cards a user would need and describes how to use them. It does not include a complete list of cards or show every available option.

The second part of the manual, Chapter 5, is a "Reference Manual" and includes a complete list of every available input card.

The third part of the manual, Chapter 6, gives several example input decks. Additional example input files can be found in the code installation directory.

In addition, a description of the VERARun script that is used to run VERA jobs is given in Chapter 7.

Note that the VERA input processor, VERAIn, is an open-source software project and can be found on the CASL Github website github.com/casl/verain. The open-source input processor does not include any physics packages.

1.4 TRAINING REQUIREMENTS

There is no required training for running VERA, but users should have a basic understanding of LWR technology. Users who perform any engineering or safety-related work with VERA should follow the procedures of their own organizations.

Optional user training is periodically available from the VERA Users Group. Please contact support (contact information given below) to inquire about training opportunities.

1.5 PURPOSE AND FUNCTIONAL REQUIREMENTS

The purpose and functional requirements of the VERA common input processor (VERAIn) are:

1. Read an ASCII input provided by the user as described in this manual.
2. Perform basic error checking on the ASCII input. Additional error checking is performed by other VERA components.
3. Perform basic geometry processing such as expanding input maps from octant to full geometry where applicable.
4. Create an extensible markup language (XML) output file that can easily be read by other VERA components.

The purpose of the VERA run script (VERARun) is to provide a single interface to run the VERA codes, usually in parallel computing environments. The specific functional requirements for VERARun are:

1. Run VERAIn to create an XML file that can be read by other VERA components.
2. Run any input preprocessors as necessary (such as XML2CTF or XML2Bison).
3. Submit jobs to a parallel computing cluster.

1.6 CODE CAPABILITIES AND LIMITATIONS

The current code capabilities of VERAIn and VERARun are specified by the functional requirements listed above. Requirements not explicitly stated in this list are assumed to be limitations.

One general limitation in the input processor is that the input is limited to standard LWR designs. For example, the input processor does not support reactors with hexagonal or plate fuel or with coolant that is not water.

Other VERA components may have limitations; the user should refer to the documentation of the other VERA components for these limitations.

1.7 TESTING

Information regarding system testing can be found in the respective VERAIn and VERARun Software Test Plan Requirements and Test Reports. These documents contain summaries of all system testing and associated requirements. Any feature not covered in these reports is considered to be untested.

1.8 COMPUTER SYSTEM VULNERABILITIES

Running VERAIn or VERARun on any machine is not known to expose the system to any security vulnerabilities at this time. VERAIn and VERARun should not be run with administrative level access permissions.

1.9 SOFTWARE SUPPORT

For specific questions about the use of VERAIn or VERARun, the licensing of the code, or to report bugs, users should send an email to `vera-support@ornl.gov`.

Additional user information may also be found on the VERA website https://vera.ornl.gov/.

2. USER MANUAL

The VERA common input is an ASCII file and is designed to be modular. The input is divided into separate modules (or blocks) to describe the different geometric objects in the core and to define specific modeling options for each of the physics codes.

Geometric objects are defined as the physical "parts" of the reactor core, which includes fuel assemblies, control rod assemblies, removable burnable poison assemblies, and detectors. By defining each geometric object as a separate block, the objects can be described independently of each other, relying on very little global information. The independent descriptions make quality assurance (QA) easier, they allow objects to be defined in one cycle, and they allow objects to be reused in subsequent cycles without concerns about input conflicts. Another advantage of the module approach is that it is easier to shuffle fuel assemblies and to insert and withdraw "inserts" such as control rods, detectors, and removable burnable poison assemblies into the fuel assemblies as the core configuration changes.

Additional modules/blocks are used to define modeling options and parameters for each of the physics codes. Separating the geometry description from the modeling options allows all of the physics codes to share the same geometry description, and it also allows the same input to be used with multiple physics codes.

The VERA input blocks are as follows:

CASEID This block contains an input title card.

CORE This block describes the core layout, including the core map, assembly locations, control rod locations, and assembly insert locations. The CORE block contains data that does not change during a cycle depletion.

STATE These blocks describe reactor core operating parameters (statepoint values) at a particular point in time. Parameters include inlet temperature, pressure, power, control rod positions, and others. STATE values can, and usually do, change at each statepoint.

ASSEMBLY These blocks contain the geometry and physical description of the nuclear fuel assemblies. The assembly descriptions do not include control rods, detectors, or inserts.

INSERT These blocks contain the geometry and physical description of the assembly inserts. An *insert* is a generic term used to describe a removable burnable poison assembly or a thimble plug assembly.

CONTROL These blocks contain the geometry and physical description of a control rod assembly. A control rod assembly is similar to an assembly insert, except that it can move during operations.

DETECTOR This block contains the geometry and physical description of a detector string.

EDITS This block contains information about what edits the code should produce.

COUPLING This block contains parameters for coupling different physics codes together.

In addition to the blocks listed above, additional code-specific blocks contain options specific to each physics code. Examples of code-specific blocks are **COBRATF**, **MPACT**, and **SHIFT**. Additional code-specific input blocks can be added as new physics codes are added to the core simulator.

The following sections in this chapter describe the most common concepts and features of each input block. This section does not provide a comprehensive list of each input card or option on each card. Refer to Chapter 5 for a detailed list of all inputs and options.

2.1 INPUT SYNTAX

VERA input files are text files that contain standard printable ASCII characters. The data are organized in blocks with names and purposes, as described in the introduction. The start of a block is denoted by the block name enclosed in square brackets (e.g., [STATE]). The file block structure is flat, so there is no hierarchy in the block segments. The start of a new block also implies the end of the previous block. There can be multiple instances of [ASSEMBLY], [INSERT], and [STATE] blocks. Other blocks, like [CORE] and the code-specific blocks, are unique, so a new block with the same name of an existing block will overwrite the existing block data. There is no required order of the blocks in the input file except for the [STATE] blocks, in which each statepoint must be entered in chronological order.

The blocks contain input cards that are generally organized as keyword-value pairs or keyword-tag-value triplets, where *tag* denotes the keyword name tag that can be referenced in the other related commands. Keywords should not have blank spaces since the spaces typically imply delimiters in the card data. A value can be a single entry or a list entry. Input card value entries can contain different data types, depending on the card format. The data types are real numbers, integers, characters, and character strings. String entries that include spaces should be enclosed in single or double quote pairs.

The block, keyword, and tag names are case sensitive. Therefore, users should not depend on capitalization for differentiation among entries in the file.

In this manual, all input examples are shown in `typewriter font`. When input cards are used in the text (not in the examples), they are listed in *italic font*. All block names are enclosed in square brackets.

The exclamation mark, "!", is a special character used for adding comments in an input file; everything from an exclamation point to the end of the line is a comment and is ignored by the processor.

The keyword *include* can be used to insert the contents of another file into the input file.

Short commands are expected to complete within a single line. Longer commands, like input maps, can be split across multiple lines.

An example input fragment with blocks, comments, and cards is shown below.

```
! comments start with an exclamation point

[STATE]             ! block names are enclosed in square brackets
  power   85.0      ! cards with parameters(s)
  flow    80.0      ! cards and parameters are separated by one or more spaces

  rodbank A 228     ! cards can span more than one line
          B 228
          C 228
          D 228

[CORE]              ! start of second block
  title  "Title must be enclosed in quotes if spaces are used"
```

Lists of values can be generated by using the following bracket nomenclature:

$$< n..m \text{ x } i >$$

where n is the starting list number, m is the ending list number, x is a delimiter, and i is the step. If "x i" is ommitted, then the step size is one. Examples of generated lists include:

```
<0..5>          0, 1, 2, 3, 4, 5
<10..16x2>      10, 12, 14, 16
<0..4x0.5>      0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0
```

Additional list options can be found on the List::Maker webpage.

2.2 CORE DESCRIPTION

The [CORE] block describes the nuclear reactor core configuration. This block describes the core layout, including the placement of nuclear fuel assemblies, control rods, detectors, inserts, and other core parameters that do not change during a cycle depletion.

The geometric objects inside the core are defined in separate input blocks; the [CORE] block simply describes how all of these objects are placed together.

2.2.1 Core Geometry

The reactor core geometry must be defined first. The overall *size* of the core is given by the number of assemblies across one major axis of the core. The assembly pitch (*apitch*) defines the width of each assembly, including the assembly gap. The distance from the top of the lower core plate to the bottom of the upper core plate is given by the parameter *height*. The assembly layout is given by the *core_shape* map. Note that the core shape map is the only "square" core map in the input, and it must be of *size* assemblies by *size*. Once the core shape is defined, subsequent core maps only include entries for actual fuel assembly locations.

```
size 15             ! number of assemblies across one axis
apitch 21.5         ! assembly pitch (cm)
height 406.337      ! distance from lower core plate to upper core plate (cm)

core_shape
  0 0 0 0 1 1 1 1 1 1 1 0 0 0 0
  0 0 1 1 1 1 1 1 1 1 1 1 1 0 0
  0 1 1 1 1 1 1 1 1 1 1 1 1 1 0
  0 1 1 1 1 1 1 1 1 1 1 1 1 1 0
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  0 1 1 1 1 1 1 1 1 1 1 1 1 1 0
  0 1 1 1 1 1 1 1 1 1 1 1 1 1 0
  0 0 1 1 1 1 1 1 1 1 1 1 1 0 0
  0 0 0 0 1 1 1 1 1 1 1 0 0 0 0
```

The *core_shape* map is unique because it is square in shape and composed of the integers 1 and 0. The 1 represents a location with a fuel assembly, and a 0 is an unoccupied location. The purpose of this map is to define the shape for subsequent core maps.

Most physics codes support both calculations run in either full-core or quarter-core symmetry. If a calculation is run in quarter-core symmetry, the code must know whether the symmetry is mirror symmetric or rotationally symmetric. The type of quarter-core symmetry is defined with the *bc_sym* input card. The symmetry option is ignored if the calculation is run in full-core.

```
bc_sym mir    ! define quarter-core symmetry as mirror
```

2.2.2 Core Maps

Core maps are used to define the location of geometry objects in the core. There are different core maps to define types and locations of assemblies, inserts, detectors, and control rods. The entries in the maps are composed of arbitrary-length character strings. Even though the character strings can be any size, it is recommended to use compact names so the maps remain legible.

All of the maps require one entry for each assembly location defined in the *core_shape* map. However, the input parser can be used to take advantage of core symmetry. If the core is symmetric, then the user only needs to input the maps in quarter or octant symmetry, and the input parser will automatically unfold the map to full symmetry using mirror symmetry, regardless of the value for *bc_sym*. The symmetry used in the core maps is independent of the symmetry used to run the actual calculations. For example, the user can enter all of the core maps in octant symmetry and still run the calculations in quarter or full symmetry. The quadrant and octant that the parser expects is shown in Figure 1.

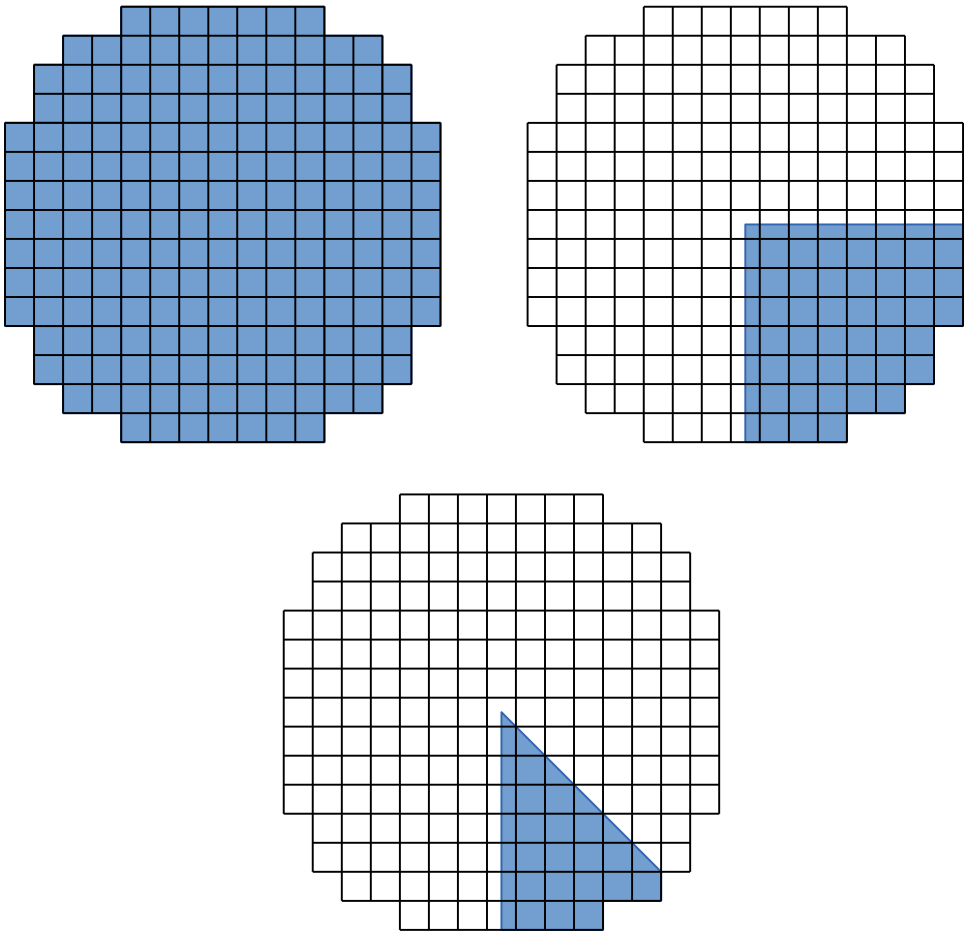


Figure 1. Full, quarter, and octant symmetry regions for a core map.

If there is an empty location in the map (e.g., if there is no detector or no control rod in an assembly), then enter a dash (also known as a *hyphen*) "-" for that location. The dash is significant and signifies an empty location in the core map. (The dash indicates that something is missing, but it is still a valid assembly location. The "0" in the *core_shape* represents an invalid assembly location.)

The *assm_map* shows where the assembly types are located within the core. In the example below, there are three assembly types that will be defined in [ASSEMBLY] block(s).

```
assm_map
                A3 A3 A3 A3 A3 A3 A3
            A3 A3 A3 A1 A3 A1 A3 A1 A3 A3 A3
         A3 A3 A2 A1 A2 A1 A2 A1 A2 A1 A2 A3 A3
         A3 A2 A2 A2 A1 A2 A1 A2 A1 A2 A2 A2 A3
      A3 A3 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A3 A3
      A3 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A3
      A3 A3 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A3 A3
      A3 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A3
      A3 A3 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A3 A3
      A3 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A3
      A3 A3 A1 A2 A1 A2 A1 A2 A1 A2 A1 A2 A1 A3 A3
         A3 A2 A2 A2 A1 A2 A1 A2 A1 A2 A2 A2 A3
         A3 A3 A2 A1 A2 A1 A2 A1 A2 A1 A2 A3 A3
            A3 A3 A3 A1 A3 A1 A3 A1 A3 A3 A3
                  A3 A3 A3 A3 A3 A3 A3
```

The following map is equivalent to the previous map but demonstrates the use of input with octant symmetry. Only values in the octant shown in Figure 1 are entered in the map, and the parser automatically unfolds the map to full symmetry.

```
assm_map
   A1
   A2 A1
   A1 A2 A1
   A2 A1 A2 A1
   A1 A2 A1 A2 A2
   A2 A1 A2 A1 A2 A3
   A1 A3 A1 A3 A3 A3
   A3 A3 A3 A3         ! assembly map with octant symmetry
```

The *insert_map* is used to show where assembly inserts are located within the core. In the following quarter-symmetry example, the inserts are burnable poison assemblies with different numbers of Pyrex rods. The *insert_map* can also be used to place geometry objects such as thimble plugs. The geometry description of the inserts will be given in the [INSERT] block.

```
insert_map
    -    BP20   -    BP20   -    BP20   -    BP12
   BP20   -    BP24   -    BP20   -    BP24   -
    -    BP24   -    BP20   -    BP16   -    BP8
   BP20   -    BP20   -    BP20   -    BP16   -
    -    BP20   -    BP20   -    BP24   -
   BP20   -    BP16   -    BP24  BP12   -
    -    BP24   -    BP16   -     -
   BP12   -    BP8    -
```

The *insert_map* is optional if no inserts are present in the core. A dash "-" is used to specify assembly locations without an insert.

The *det_map* is used to show where detectors are located in the core. The geometry description of the corresponding detector strings is given in the [DETECTOR] block. In this example, there is only one detector type, denoted with a "1". Since the "1" occurs in a core map, it is treated as a character string. This example uses a full-symmetry map.

```
det_map
                - - 1 - - 1 -
          1 - - 1 - 1 - - - - -
      - - - - - - 1 - 1 - 1 - 1
      1 1 - - - - 1 - - - - - -
  - - - - 1 - - - 1 - 1 - 1 - -
  1 - 1 - - 1 - 1 - - - - - 1 -
  - - - 1 - - 1 - - 1 - - 1 - -
  1 - 1 - 1 - 1 - - 1 - 1 1 1 -
  - 1 - - - - - - 1 - 1 - - - 1
  - - - - 1 - 1 - - - - 1 - - -
  1 - - - 1 - - 1 - - 1 - - - 1
    - - - - 1 - - 1 - - 1 - -
    - 1 - 1 - - 1 - - - - - 1
      1 - - - 1 - - 1 - 1 -
            1 - - 1 - - -
```

The *det_map* is optional if no detectors are present in the core. A dash "-" is used to specify assembly locations without a detector.

The control rod assemblies are described with two maps. The *crd_map* defines the control rod types and locations in the core. The *crd_bank* map assigns control rod locations to control rod banks. The control rod maps are optional if no control rods are present in the core. In the following example, there is only one control rod type, labeled "1".

```
crd_map
                - - - - - - -
          - 1 - 1 - 1 - 1 - 1 -
      - - - 1 - 1 - 1 - 1 - - -
      1 - 1 - - - 1 - - - 1 - 1
  - - 1 - 1 - - - - - 1 - 1 - -
  - 1 - - - 1 - 1 - 1 - - - 1 -
  - - 1 - - - - - - - - - 1 - -
  - 1 - 1 - 1 - 1 - 1 - 1 - 1 -
  - - 1 - - - - - - - - - 1 - -
  - 1 - - - 1 - 1 - 1 - - - 1 -
  - - 1 - 1 - - - - - 1 - 1 - -
    1 - 1 - - - 1 - - - 1 - 1
    - - - 1 - 1 - 1 - 1 - - -
      - 1 - 1 - 1 - 1 - 1 -
            - - - - - - -
```

```
crd_bank
                -  -  -  -  -  -  -
          - SA  -  B  -  C  -  B  - SA  -
       -  -  - SD  - SB  - SB  - SC  -  -  -
     SA  -  D  -  -  -  D  -  -  -  D  - SA
   -  - SC  -  A  -  -  -  -  -  A  - SD  -  -
   -  B  -  -  -  C  -  A  -  C  -  -  -  B  -
   -  - SB  -  -  -  -  -  -  -  -  - SB  -  -
   -  C  -  D  -  A  -  D  -  A  -  D  -  C  -
   -  - SB  -  -  -  -  -  -  -  -  - SB  -  -
   -  B  -  -  -  C  -  A  -  C  -  -  -  B  -
   -  - SD  -  A  -  -  -  -  -  A  - SC  -  -
     SA  -  D  -  -  -  D  -  -  -  D  - SA
      -  -  - SC  - SB  - SB  - SD  -  -  -
         - SA  -  B  -  C  -  B  - SA  -
                -  -  -  -  -  -  -
```

2.2.3 Core Baffle and Vessel

The *core baffle* or *shroud* is a steel reflector that closely surrounds the fuel assemblies in the core. The *barrel* is a round steel structure that surrounds the baffle, and the *vessel* is the round outer pressure vessel. These structures are shown in Figure 2.

The *baffle* is defined with a single material, the size of the gap between the outer assembly and baffle, and the baffle thickness.

```
baffle   SS304 0.19 1.26   ! material, gap (cm), and thickness (cm)
```

The barrel and vessel are defined with a *vessel* card. This card allows the user to enter any arbitrary number of rings surrounding a core by specifying the ring radii and the materials between the rings.

```
vessel   mod 166.7 SS304 169.2 mod 175.0 SS304 176.0   ! materials and radii (cm)
```

There is currently no input defined to specify the neutron pad.

2.2.4 Core Plates

The core plates are large steel plates at the top and bottom of the core that have various flow holes passing through them. All of the axial core heights are defined relative to the top of the bottom core plate, and the total core *height* is defined as the distance between the top of the bottom core plate and the bottom of the top core plate.

The core plates are modeled in the neutronics codes as smeared materials. The upper and lower core plates are defined with a material composition, a thickness, and a volume fraction of the structural material. The remainder of the volume fraction is filled with coolant.

```
lower_plate SS304  5.0  0.5   ! material, thickness (cm), volume fraction
upper_plate SS304  7.6  0.5   ! material, thickness (cm), volume fraction
```

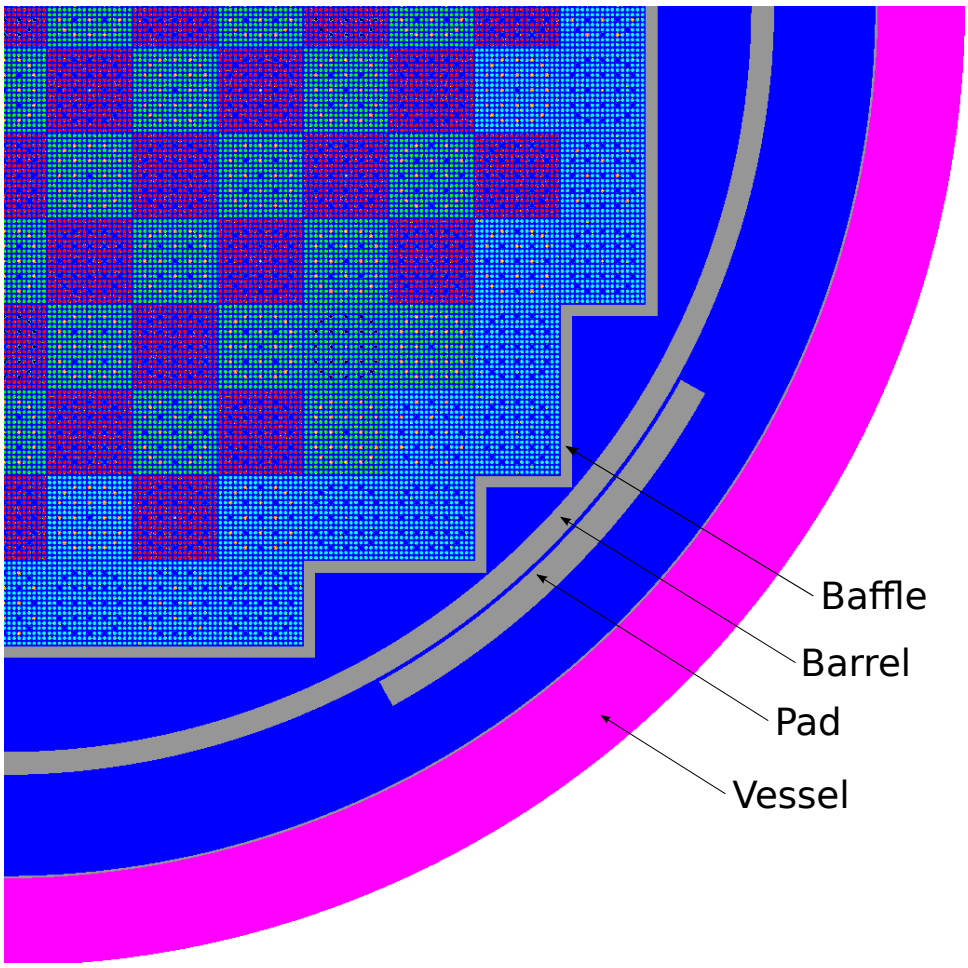


Figure 2. Core baffle and vessel (Image courtesy of Andrew Godfrey).

2.2.5 Small Core Geometries

Even though the VERA input is designed for "real" core geometries, it can accommodate smaller problems as well. For example, to run a single-assembly calculation, the user would define the core size as one assembly by one assembly, and all of the core maps would contain a single assembly.

```
size 1                ! core composed of a single-assembly
core_shape
   1
```

To model a single fuel rod, the user would define a core with one assembly and an assembly with one rod in it.

2.3 ASSEMBLY DESCRIPTION

The [ASSEMBLY] block contains the geometric description of a unique fuel assembly design (type). Multiple [ASSEMBLY] blocks are permitted to describe different assembly designs in the core.

If there are multiple assembly designs that are geometrically identical (i.e., everything is the same except the enrichments), then they can all be defined in a single [ASSEMBLY] block. Each assembly type will have a unique *axial* card with possibly unique axial levels and lattice types. Assemblies within a single reload typically have a design similar enough that they can share a single [ASSEMBLY] block.

If assembly designs are not geometrically identical (e.g., if they are from different vendors or different generations), then they need to be defined in separate [ASSEMBLY] blocks. One advantage to having separate blocks for each assembly design is that each design can be modeled (and archived) independently without the need to rely on global definitions.

A typical PWR assembly is shown in Figure 3. Refer to this figure in the following discussions.

A complete list of all the input cards in the [ASSEMBLY] block is located in Chapter 5.

2.3.1 Initial Data

Each assembly block must contain a geometry description with the number of pins across the assembly and the pin pitch. An assembly block can also include an optional title card.

```
title "Westinghouse 17x17"      ! assembly title
npin 17                         ! number of pins across one side
ppitch 1.260                    ! pin pitch (cm)
```

The number of pins *npin* must be the same for every assembly in a core.

The inter-assembly gap on each side of the assembly is calculated as $[apitch - npin * ppitch]/2$

For a boiling water reactor (BWR), a channel box may be specified with the following cards. See Section 2.7 for a complete description of the channel box inputs.

```
channel_box zirc 0.254 0.10         ! mat, corner thickness (cm), corner radius (cm)
channel_box_segments 0.18 2.5 0.2   ! segment thickness (cm), segment length (cm),
                                    ! segment ramp (cm)
```

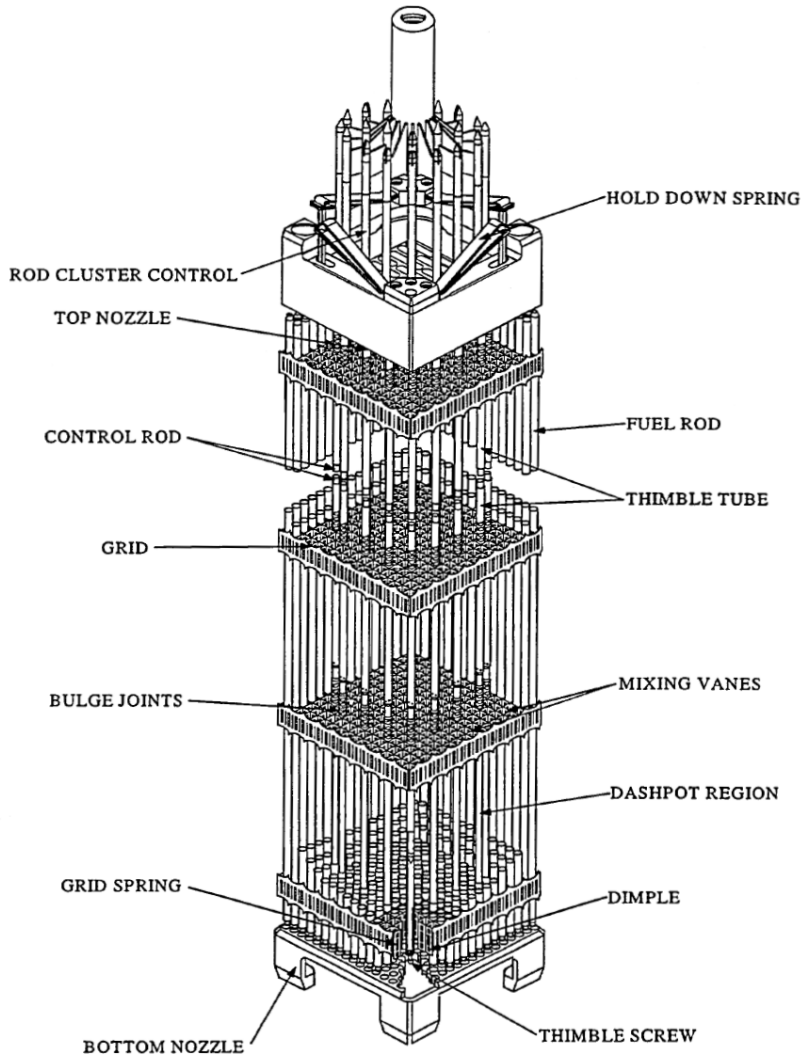


Figure 3. PWR fuel assembly (Image courtesy of the US Nuclear Regulatory Commission).

The fuel and structural materials are defined with the following cards. See Chapter 3 for a complete description of the material inputs.

```
fuel U31 10.257 95.0 / 3.1   ! mat, density (g/cc), Theoretical density (%)
                             !    / U-235 enrichment (%)
mat inc   8.19               ! mat, density (g/cc)
mat ss    8.0
mat zirc4 6.56
```

2.3.2 Cell Descriptions

Cell cards are used to describe "pincells." A pincell is defined as a configuration of concentric cylinders (or rings) centered in a square region of coolant. Cell configurations can be used to model fuel rods or guide tubes (GTs), as shown in Figure 4.

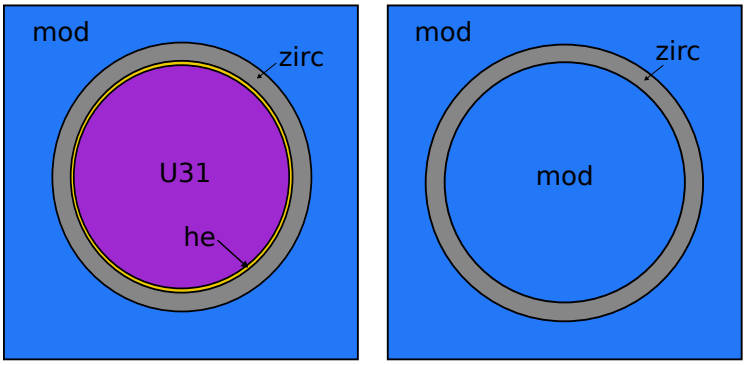


Figure 4. Pincell diagrams of a fuel rod and a guide tube.

The first parameter on the *cell* card is the cell ID. This is followed by a list of radii for each ring in the cell, followed by a slash. After the slash is a list of materials that comprise each ring. The cell IDs are used in the rod maps described in the next section.

```
cell 1     0.4096 0.418 0.475 / U31 he zirc4
cell GT           0.561 0.602 / mod    zirc4      ! guide tube
cell IT           0.561 0.602 / mod    zirc4      ! instrument tube
cell 7            0.418 0.475 / mod    mod        ! empty location
cell 8            0.418 0.475 /     he zirc4      ! plenum
cell 9                  0.475 /        zirc4      ! pincap
```

In this example, in cell "1," the material "U31" extends from radius 0 to 0.4096. The material "he" extends from a radius 0.4096 to 0.418. The materials "U31" and "he" are defined on *fuel* and *mat* cards, respectively. (Refer to Chapter 3 for a complete description of material definitions.)

The outside of each cell is automatically filled with the special material "mod," which refers to the moderator (or coolant). The composition of "mod" is calculated by the codes using the local thermal hydraulic (T/H) conditions and the soluble boron concentration and cannot be specified by a user on a *mat* card.

In the example above, the GT and instrument tube (IT) descriptions use the special moderator material "mod" to define the moderator material on both the inside and outside of the tubes.

Large water rods that span more than one lattice cell can be specified by adding an optional keyword "large4" to the end of the *cell* card.

```
! large CE 16x16 water rod
  ppitch  1.28524
  cell WR 1.26 1.28 / mod zirc4 / large4
```

2.3.3 Lattice Descriptions

Once the cells are defined, they are placed into 2D "lattices" as shown below. Like the core maps, lattice maps can be entered with either full-symmetry, quarter-symmetry, or octant-symmetry. The maps below are octant-symmetric maps for 17×17 assembly designs.

```
rodmap FUEL1
    IT
     1 1
     1 1 1
    GT 1 1 GT
     1 1 1  1 1
     1 1 1  1 1 GT
    GT 1 1 GT 1  1 1
     1 1 1  1 1  1 1 1
     1 1 1  1 1  1 1 1 1

rodmap LGAP1
    IT
     7 7
     7 7 7
    GT 7 7 GT
     7 7 7  7 7
     7 7 7  7 7 GT
    GT 7 7 GT 7  7 7
     7 7 7  7 7  7 7 7
     7 7 7  7 7  7 7 7 7

rodmap PLEN1
    IT
     8 8
     8 8 8
    GT 8 8 GT
     8 8 8  8 8
     8 8 8  8 8 GT
    GT 8 8 GT 8  8 8
     8 8 8  8 8  8 8 8
     8 8 8  8 8  8 8 8 8

rodmap PCAP1
    IT
     9 9
     9 9 9
    GT 9 9 GT
     9 9 9  9 9
     9 9 9  9 9 GT
    GT 9 9 GT 9  9 9
     9 9 9  9 9  9 9 9
     9 9 9  9 9  9 9 9 9
```

Rod maps define each unique axial level in the assembly. The first parameter is the lattice name (e.g., FUEL1 and PCAP1), followed by a map of the *cell* IDs.

Each entry in a rod map must be a valid cell ID.

2.3.4 Axial Descriptions

After rod maps are defined for each axial level, the lattices are "stacked" into an assembly using an *axial* card as shown below.

```
axial A1    6.050
    LGAP1  10.281
    PCAP1  11.951
    FUEL1 377.711
    PLEN1 393.711
    PCAP1 395.381
    LGAP1 397.501
```

The *axial* card tells the code how to place the lattices axially. The first parameter is the name of the assembly (A1), followed by a list of elevations and lattice types. For example, lattice "FUEL1" extends from 11.951 to 377.711 cm axially.

Multiple assembly types can be defined in a single [ASSEMBLY] block by using multiple *axial* cards, each with a unique assembly ID.

All axial elevations are defined relative to the top of the lower core plate.

2.3.5 Grid Spacer Descriptions

Grid cards are used to define unique grid spacer types. The following example defines two grid types, "END" and "MID."

```
grid END inc    3.866 1017 / loss=0.9070  ! material, height(cm), mass (g), loss coef
grid MID zirc4  3.810 875 / loss=0.9065
```

The grid types are placed axially with the *grid_axial* card:

```
grid_axial
    END  13.884
    MID  75.2
    MID 127.4
    MID 179.6
    MID 231.8
    MID 284.0
    MID 336.2
    END 388.2
```

The elevations are the midpoints of the spacer grid and are relative to the top of the lower core plate.

2.3.6 Nozzle Descriptions

The assembly nozzles are modeled in the neutronics codes as smeared materials. This approximation is very good since the nozzles are not in the active fuel region and are mostly composed of water, steel, and zirconium. The user only specifies a nozzle mass and a nozzle height. The total volume of the nozzle region is calculated from the assembly pitch and nozzle height. The volume of the nozzle is calculated from the nozzle mass and density. The volume of the coolant is then calculated as the total volume minus the volume of the nozzle. The coolant density is updated with the local T/H conditions.

```
lower_nozzle  ss 6.05  6250.0  ! mat, height (cm), mass (g)
upper_nozzle  ss 8.827 6250.0  ! mat, height (cm), mass (g)
```

Only a single material can be specified on a nozzle card. To use more than one material to define a nozzle, the user can define a custom material that is a mixture of the materials and then use the custom material in the nozzle card.

Note that the *lower_nozzle* height should match the bottom elevation on the *axial* card. The *upper_nozzle* height + the top elevation on the *axial* card must match the core *height* in the [CORE] block. The input parser does not currently perform a check to make sure the elevations are consistent. Therefore, this check should be performed in each of the individual physics codes.

2.4 CONTROL ROD ASSEMBLY DESCRIPTION

The [CONTROL] block contains the geometric description of a control assembly.

A control rod assembly is defined in a manner similar to that used to define a fuel assembly. The user specifies cells, lattices, and axial descriptions of the control rod assembly. The main difference between the control rod assembly and the fuel assembly is that the control rod assembly describes what is inside the GTs, whereas the fuel assembly defines the GTs themselves.

Control rod positions change during operation, so the geometric description of a control rod should always be for a rod in the **fully inserted** position. In the example below, the bottom of the control rod in the fully inserted position is at an axial location of 15.46 cm.

```
title "B4C control rods with AIC tips"
npin 17

cell 1  0.382 0.386 0.484 / aic he ss      ! AIC cell
cell 2  0.373 0.386 0.484 / b4c he ss      ! B4C cell

rodmap  AIC
   -
   - -
   - - -
   1 - - 1
   - - - - -
   - - - - - 1
   1 - - 1 - - -
   - - - - - - - -
   - - - - - - - - -

rodmap  B4C
   -
   - -
   - - -
   2 - - 2
   - - - - -
   - - - - - 2
   2 - - 2 - - -
   - - - - - - - -
   - - - - - - - - -

axial CR1 15.46 AIC  376.44 B4C  394.3
```

The name of the control rod "CR1" refers to the control rod type in the *crd_map* in the [CORE] block.

Control rod positions are assigned to a control rod bank with the *crd_bank* map in the [CORE] block, and then the banks are positioned with the *rodbank* card in the [STATE] block.

Note that the locations of the control rod fingers must match the GT locations in the corresponding [ASSEMBLY] block descriptions. Furthermore, the outer radii of the control rod fingers must be smaller than the inner radii of the GTs. The input parser does not currently perform a check to make sure the control rod finger descriptions are consistent with the GT descriptions. This check should be performed in each of the individual physics codes.

The user can define materials in the [CONTROL] block. These materials only have scope in this block and are not accessible by other blocks. See Chapter 3 for details.

A complete list of all the input cards in the [CONTROL] block is provided in Chapter 5.

2.4.1 Control Rod Stroke

The difference between control rod descriptions and assembly descriptions is that the control rods move during operation. This movement is defined with a *stroke* card.

The first value on the *stroke* card is the total length of the control rod travel (stroke) from fully inserted to fully withdrawn.

The second value on the *stroke* card is the number of steps in the fully withdrawn position. Step 0 is the fully inserted position. The number of steps in the fully withdrawn position is specified by the user, but 228 steps is often the number used for typical Westinghouse PWRs.

```
stroke  360.0 228      ! stroke (cm), number of steps fully withdrawn
```

To position the control rods in percent withdrawn (%), the number of steps should be set to 100, and each step will signify 1% withdrawn.

The geometry description in the input is for a control rod in the fully inserted position (step 0).

2.4.2 Control Rod Position Example

From the *axial* card shown above, the bottom of the AIC at the fully inserted position is 15.46 cm. From the *stroke* card, the total stroke is 360.0 cm, and the number of steps in the fully withdrawn position is 228 steps. Therefore, the bottom elevation of the AIC lattice at step N will be

$$E(N) = 15.46 + \frac{360.0 \cdot N}{228} \tag{1}$$

Using this formula, the bottom elevation of the AIC lattice at the following step positions is as follows:

- step 228 (fully withdrawn) = 15.46 + 360.0 * 228 / 228 = 375.46 cm
- step 100 = 15.46 + 360.0 * 100 / 228 = 173.35 cm
- step 0 (fully inserted) = 15.46 + 360.0 * 0 / 228 = 15.46 cm

The steps withdrawn can be specified as real numbers and fractions of a step.

2.5 INSERT DESCRIPTION

An assembly insert is defined in the same way as a fuel assembly or control rod assembly. The user defines the insert using cells, lattices, and axial descriptions.

The fuel assembly description should contain the GT descriptions, and the insert description defines what is inserted into the GTs. Assembly inserts can be inserted and withdrawn during a core shuffle (by specifying an *insert_map* card in the [CORE] block), but they cannot be moved during a cycle depletion.

The insert and control rod descriptions are very similar, with the only difference being that the insert cannot change position axially during a cycle depletion, and a control rod moves axially during operations.

The following example shows a definition of a Pyrex insert.

```
[INSERT]
    title "Pyrex"
    npin 17
    mat pyrx1 2.25 pyrex-vera
    cell 1  0.214 0.231 0.241 0.427 0.437 0.484 / he ss he pyrx1 he ss
    rodmap  PY24
       -
       - -
       - - -
       1 - - 1
       - - - - -
       - - - - - 1
       1 - - 1 - - -
       - - - - - - - -
       - - - - - - - - -

    axial INS24  15.76 PY24 376.441
```

The name of the insert "INS24" refers to an insert type defined in the *insert_map* in the [CORE] block.

The locations of the insert fingers must match the GT locations in the corresponding [ASSEMBLY] block descriptions. In addition, the outer radii of the insert fingers must be smaller than the inner radii of the GTs. The input parser does not currently perform a check to make sure the insert finger descriptions are consistent with the GT descriptions. This check should be performed in each physics code.

As with [ASSEMBLY] blocks, multiple insert types can be defined in a single [INSERT] block by using multiple *axial* cards, each with a unique insert ID.

A complete list of all the input cards in the [INSERT] block is provided in Chapter 5.

2.6 DETECTOR DESCRIPTION

A detector string is defined in the same way that a fuel assembly or insert assembly is defined. The user defines cells, lattices, and axial descriptions for the detector string.

The insert and detector descriptions are very similar, with the difference being that detectors have special properties used to calculate instrumentation signals.

```
[DETECTOR]
  title "Incore instrument thimble"
  npin 17

  mat he 0.0001786
  mat ss 8.0

  cell 1  0.258 0.382 / he ss

  rodmap  LAT
     1
     - -
     - - -
     - - - -
     - - - - -
     - - - - - -
     - - - - - - -
     - - - - - - - -
     - - - - - - - - -

  axial D1  0.0 LAT 406.337
```

The name of the detector "D1" refers to a detector type defined in the *det_map* in the [CORE] block.

A complete list of all the input cards in the [DETECTOR] block is located in Chapter 5.

2.7 CHANNEL BOX

The channel box that surrounds a BWR assembly may be modeled with the *channel_box* and *channel_box_segments* cards. The *channel_box* card allows the representation of a normal box with a nominal thickness and rounded corners. When paired with the *channel_box_segments* card, a general explicit geometry of thick-thin designs may be modeled.

The card inputs are shown in Figure 5.

```
channel_box  <material>  <corner_thickness>  <corner_radius>
channel_box_segments  <thickness_i>  <length_i>  <ramp_length_i>
```

Figure 5. Description of the channel box cards.

All of the dimensional quantities are in units of centimeters. The segment length and segment ramp are deltas in length. The segment ramp should be entered as the horizontal distance and not the slope. The definitions of the inputs are illustrated in Figure 6.

The first channel box segment is entered at the halfspan of the box. Each additional segment is entered in the direction of the corner. It is assumed that the channels are symmetric about the corner and that all corners are alike. The thick corner length is internally calculated from the given channel segments.

For both the normal and thick-thin channel box designs, the wide and narrow gaps should be entered relative to the corner thickness. For the normal box, this is equivalent to the nominal channel thickness.

Sample inputs are provided in Figure 7.

A complete list of the channel box cards is located in Chapter 5.

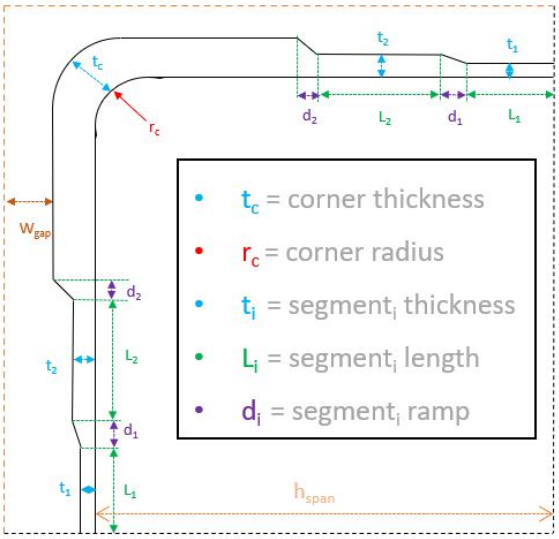


Figure 6. Demonstration of the channel box cards.

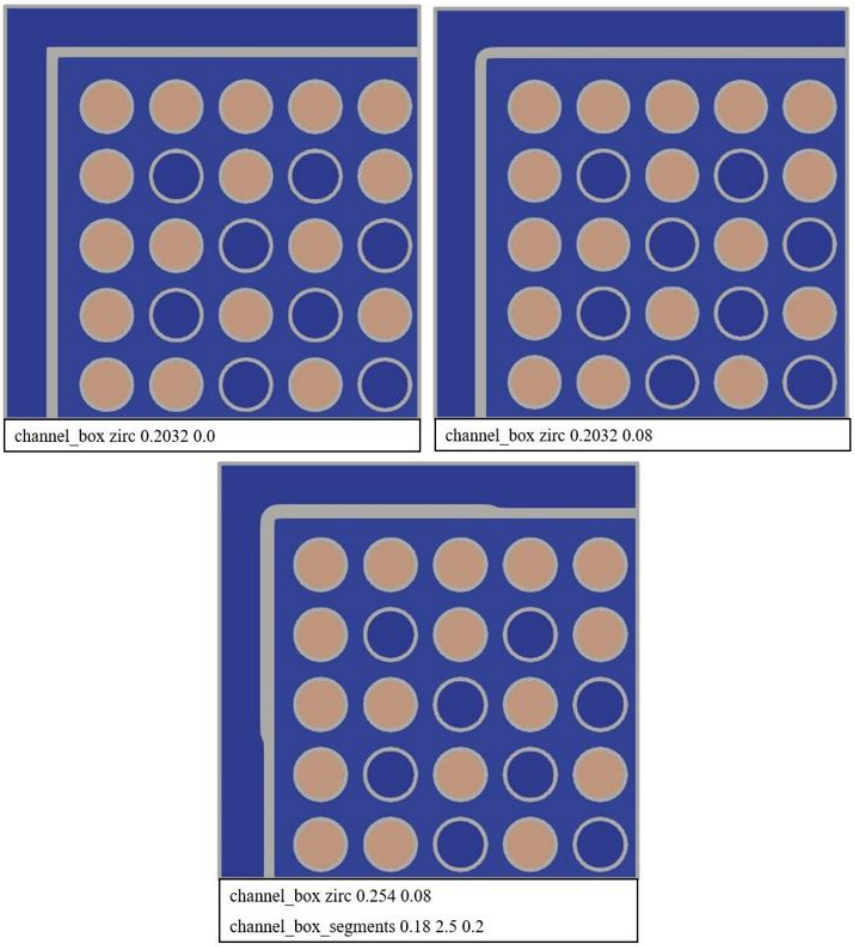


Figure 7. Sample inputs for the channel box cards.

2.8 STATE DESCRIPTION

The [STATE] block defines the state of the core (power, flow, pressure, inlet temperature, rod positions, boron concentration, etc.) at a particular point in time. These values will typically change during a cycle depletion.

An example showing the most common input cards in the [STATE] block is shown below. A complete listing of all the input cards in the [STATE] block is located in Chapter 5.

```
[STATE]
  power   98.0       ! % of rated power - rated values defined in [CORE] block
  flow   100.0       ! % of rated flow
  pressure 2250.0    ! psia
  tinlet 557.33 F    !
  feedback on        ! turn on T/H feedback

  boron   1285       ! initial boron ppmB
  search  boron      ! turn on boron search

  sym qtr            ! run problem in qtr-symmetry

  rodbank SA 228
          SB 228
          SC 228
          SD 228
           A 228
           B 228
           C 228
           D 167
```

The *sym* card tells the code to run the calculation in full-core or quarter-core symmetry. If the calculation is run in quarter-core symmetry, then the symmetry is either set to quarter-core rotational or quarter-core mirror by the *bc_sym* card in the [CORE] block.

The *rodbank* card is used to position the control rods. The *rodbank* input includes pairs of bank names and bank positions. The bank names correspond to the *crd_map* in the [CORE] block. The positions indicate the position of the control rod bank in steps. Step 0 is fully inserted. The number of steps for a rod to be completely withdrawn is set by the *stroke* card in the [CONTROL] block (see Section 2.4.1). For Westinghouse PWRs, a typical value of fully withdrawn is 228 steps.

2.9 EDITS DESCRIPTION

The [EDITS] block is used to control the output edits.

One of the edits produced by the core simulator is the rod power. The user has the ability to specify the axial levels that the power is averaged over with the *axial_edit_bounds* card. The user may choose to average power over uniform axial intervals (like most nodal codes), or to specify the edit intervals manually.

(Note: the edit options are under development and more options will be added in the future.)

A complete listing of all the input cards in the [EDITS] block is located in Chapter 5.

2.9.1 CTF Nodalization

The *axial_edit_bounds* card is also used to set the axial nodalization when coupling the neutronics physics code to the COBRA-TF (CTF) subchannel code.

When running CTF, the grid boundaries must be explicitly included in the *axial_edit_bounds*. This process can get a little complicated for the user. In the VERA input, spacer grids are defined in the [ASSEMBLY] block by specifying the grid heights on the *grid* card and the elevations of the grid midpoints on the *grid_axial* card. From the grid heights and midpoints, the elevations at the top and bottom of the spacer grid can be calculated, and then the top and bottom elevations must be included in the *axial_edit_bounds*.

For example, if a grid is defined with a centerline at 75.0 and a height of 2.5, then the *axial_edit_bounds* must include the points 75.0 ± 1.25 = 73.75 and 76.25.

```
[ASSEMBLY]
 grid GRID1 inc 1000 2.5    ! grid name, material, mass (g), and height (cm)
 grid_axial                 ! locations of grid midpoints (cm)
     GRID1 75.0

[EDITS]
 axial_edit_bounds
    ...
    73.75           ! this array must include top and bottom grid boundaries
    76.25
    ...
```

The reason for this restriction is that the power is calculated on the *axial_edit_bounds*, so it is natural to use the same power distribution to couple to the CTF model as well. The grids must be explicitly included in the CTF boundaries so that the loss coefficients are calculated correctly.

In the future, this restriction may be lifted and an additional edit bounds array may be added explicitly for CTF calculations.

2.10 COUPLING DESCRIPTION

The [COUPLING] block defines the relaxation parameters and convergence criteria to be used when coupling different physics codes. These values are used to determine convergence *between* physics codes. Convergence criteria *within* a physics code is controlled by the code-specific block.

Refer to Chapter 5 for a complete listing of all the cards in the [COUPLING] block.

No code-specific information is included in the [COUPLING] block; all code-specific information is contained in the code-specific blocks. The [COUPLING] block is only used to define generic coupling parameters.

As an example, consider the following multiphysics code coupling:

1. Run T/H calculation
2. Run neutronics calculation
3. Check eigenvalue convergence
4. Check power convergence

5. Relax/dampen the power shape
6. If not converged, go to step 1.

The eigenvalue convergence in step 3 uses the card *epsk* to check the change in eigenvalue between coupled iterations. There are additional eigenvalue convergence criteria *within* the neutronics code, but the internal parameters are specified in the individual code blocks.

The power convergence in step 4 uses the card *epsp* to check the change in power between coupled iterations.

Additional convergence checks are made on the peak fuel temperature, maximum change in density, and change in boron concentration (if applicable).

The example shown above uses a Picard iteration to converge. Picard iterations usually need to apply a relaxation factor (also called a damping factor or under-relaxation factor) to one or more of the calculated quantities to converge. The relaxation factors are applied in the following manner:

$$x = \omega\, x^{\text{new}} + (1 - \omega) x^{\text{old}} \tag{2}$$

where x is the calculated parameter and ω is the relaxation factor. A relaxation factor of 1.0 signifies that no relaxation is performed. A relaxation factor < 1.0 signifies under-relaxation.

Relaxation factors can be specified for the point-wise power, point-wise temperature, and/or point-wise density. The relaxation is applied to the transferred quantities sent between physics codes. The state variables within each physics code are not changed.

An example [COUPLING] input block is shown below.

```
[COUPLING]
  epsk        5.0  ! eigenvalue convergence (pcm)
  eps_temp    1.0  ! temperature convergence (deg C)
  eps_boron   0.1  ! boron convergence (ppm)
  rlx_power   0.5  ! power relaxation factor
  rlx_tfuel   1.0  ! fuel temperature relaxation factor
  rlx_den     1.0  ! density relaxation factor
  maxiter     20   ! maximum number of coupled iterations
```

A complete listing of all the input cards in the [COUPLING] block is located in Chapter 5.

3. MATERIALS

This chapter contains a description of the material input. There are two types of materials in the input file: structural materials (input with a *mat* card) and fuel materials (input with a *fuel* card).

Structural materials can be defined in either the [CORE] block or in the geometry object blocks [ASSEMBLY], [INSERT], [CONTROL], and [DETECTOR]. If the materials are defined in the [CORE] block, they have global scope. If the materials are defined in the geometry object blocks, then they only have scope in the block in which they are defined. This maintains the modularity of the geometry objects.

Fuel materials can only be defined in [ASSEMBLY] blocks.

Materials are used in many different input cards. They are used to define cells, nozzles, core plates, baffles, grids, reflectors, etc. Every material used in the input must be defined with either a *mat* card or a *fuel* card (see notes on the material "mod" below).

3.1 STRUCTURAL MATERIALS

Structural materials are not fuel and do not deplete. Structural materials are defined with the following input card:

mat *user-mat density* (*library-name*$_i$, *frac*$_i$, i=1, I)

where:

- *user-mat* is a user-defined material name. The name is case sensitive. *user-mat* is used to define material names in other input cards such as *cell*, *grid*, *nozzle*, etc. (No default).
- *density* is the material density in grams per cubic centimeter (g/cc). (No default).
- *library-name* is a corresponding library name(s) for the user material. The library name must be defined in the cross section library. (Default = *user-mat*). Multiple library materials can be mixed to form a single user material.
- *frac* is the fraction of the library material in the user material. (Default=1.0 if there is only one library material in the user material).

There are two special user materials, "mod" and "vacuum." The user can use these materials in cell definitions, but the code will automatically determine the composition of these materials based on T/H feedback and soluble boron concentrations. The user is not allowed to define a user material named "mod" or "vacuum" on a *mat* card.

Some example material cards are shown below.

```
mat zirc4 6.56                   ! library-name defaults to user-name
mat zirx  6.56 zirc4 1.0         ! user-name does not equal to library-name
mat B10   12.0 boron 1.0
mat XYZ   6.0  zirc4 0.8 ss 0.2  ! define new mixture of 80% zirc4 and 20% ss
mat ABCD  8.0  zirc4 0.8 ss 0.15 b4c 0.05
mat waba  3.65 b-10  1.36210E-02 ! creates material from isotopes in the XS-library
               b-11  6.02818E-02
               c-00  2.05259E-02
               o-16  4.26297E-01
               al-27 4.79274E-01
```

All of the material fractions must sum to either +1.0 or -1.0. If positive fractions are used, then the fractions refer to weight fractions. If negative fractions are used, then the fractions refer to atomic fractions.

3.1.1 Search Order

Structural materials can be defined in either the [CORE] block or one of the geometry object blocks. When a material is referred to in a block, it will look for the material definition in the following order:

1. The code will first look for the material name in the local block ([ASSEMBLY], [INSERT], [CONTROL], or [DETECTOR]).
2. If the material is not found in the local block, then it will look in the [CORE] block.

If materials are defined in the [CORE] block, then they have global scope over the entire input; if materials are defined in other blocks, they only have scope over the local block. This means that two geometry object blocks can use different material definitions with the same name. One example of this is that two assemblies can be defined with the material "zirc," but "zirc" can have different compositions in each of the assemblies.

3.2 DEFAULT MATERIALS

There are many default files available to the users. The default materials and their compositions are defined on the initialization file CORE.ini. A list of default materials is given in Table 1.

Table 1. Default Material List

Material	Density (g/cc)	Notes
air	1.189E-03	
aic	10.2	silver-indium-cadmium
al2o3	3.96	
b2o3	2.55	
b4c	1.7597	boron carbide
boron	2.37	
cs	7.85	carbon steel
gad	7.407	
gap	0.17860E-03	
he	0.17860E-03	
inc	8.19	inconel
pyrex	2.34249	
pyrex-vera	2.24419	
sio2	2.18	
ss	8.0	stainless steel
tungsten	19.3	
water	0.743	
waba	3.65	
zirc2	6.56	Zircaloy-2
zirc4	6.56	Zircaloy-4
clad	6.56	
zirc4-xhf	6.55934	Zircaloy-4 with no Hf
zr	6.506	natural zirconium

3.3 FUEL MATERIALS

Fuel materials are defined with *fuel* cards. Fuel materials are heavy metal oxides, usually UO_2 with different ^{235}U enrichments. Fuel materials may also include mixed oxide (MOX) fuel, which consists of mixtures of

uranium, plutonium and other actinides. Fuel materials may also contain integral burnable absorbers, such as gadolinia. Fuel materials are different from structural materials in that they deplete and have additional properties, as described below.

Fuel can only be defined in [ASSEMBLY] blocks, and fuel materials can only be referenced by *cell* cards in the [ASSEMBLY] block in which they are defined.

Fuel materials are defined with the following input card:

fuel *user-mat density thden* / *U-235_enrichment* {$HM\_material_i$=$HM\_enrichment_i$, i=1, N}
{ / *gad_material=gad_fraction* }

Where:

- *user-mat* is a user-defined fuel name. It is case sensitive (no default).
- *density* is the fuel material density in g/cc (no default). The density is used to calculate number densities.
- *thden* is the percent of theoretical density in the pellet (%) (no default). The theoretical density is only used to look up material properties in the fuel performance; it is not used to calculate number densities. There is no "double counting" between *density* and *thden*.
- *U-235_enrichment* is the ^{235}U enrichment in the fuel in weight % (no default).
  - If ^{234}U and ^{236}U are not specified, then they will automatically be added to the fuel by a pre-determined function (see below).
  - If the sum of the heavy metal (HM) enrichments does not equal 100%, then the remainder of the HM composition will be assigned to ^{238}U.
- $HM\_material_i$ is the material name for HM isotope *i* (^{239}Pu, ^{241}Pu, etc.) (optional). The names of the HM materials must be valid library names.
- $HM\_enrichment_i$ is the enrichment of HM isotope *i* in weight % (optional).
- *gad_material* is the material name for gadolinia or other integral burnable absorber material (optional). The gad material is usually a mixture defined on a separate *mat* card.
- *gad_fraction* is the weight percent of the gad material relative to the total fuel mass (optional).

Oxygen should not be included on the *fuel* card. The correct amount of oxygen will automatically be added to the HM to create an oxide (either UO_2 or $(HM)O_2$).

The *density* is the "stack density" or "smeared density" and should include the volume of the pellet dishing and chamfers. It is calculated as the total mass of the fuel pellets divided by the total volume of the fuel

$$\text{stack density} = \frac{\text{(fuel mass)}}{\pi(\text{pellet radius})^2\,\text{(fuel height)}} \tag{3}$$

The *thden* refers to the actual theoretical density of the pellet. This quantity may be used in fuel performance codes to evaluate material properties.

If ^{234}U or ^{236}U enrichments are not included in the fuel definition, then they are automatically added to the fuel with the following formulas:

$$W_{234} = 0.0089 \cdot W_{235} \tag{4}$$

$$W_{236} = 0.0046 \cdot W_{235}, \tag{5}$$

where W_{23x} is the enrichment of each of the uranium isotopes in percent.[2]

If the user specifically does NOT want ^{234}U or ^{236}U, then a ^{234}U and/or ^{236}U enrichment of zero should be specified.

Examples of typical *fuel* cards are shown below. The user only has to specify the ^{235}U enrichment, and the code will automatically add ^{234}U, ^{236}U, ^{238}U, and oxygen to the fuel.

```
fuel U21 10.4 95.2 / 2.1        ! 2.1% enriched UO2 fuel, no gad
fuel UO2-35 10.297 95.0 / 3.5   ! 3.5% enriched UO2 fuel, no gad
fuel U23    10.111 / 2.3        ! fuel with default thden
```

An example of a *fuel* card with gadolinia burnable poison is shown next. In this example, the gadolinia oxide is first defined with a *mat* card and is mixed with the fuel as 5% gad oxide and 95% UO_2 (weight percents).

```
mat gad5 7.407 gd2o3 1.0                ! define gad material separately
fuel U49 10.111 94.5 / 1.8  / gad5=5.0  ! 1.8% enriched fuel with 5% gad
```

Some examples of MOX fuel cards are shown next. In these cards, the user specifies the ^{235}U enrichment (the ^{235}U enrichment is usually small in MOX fuel) and the plutonium isotope enrichments. The code will automatically add ^{234}U, ^{236}U, ^{238}U, and oxygen.

```
fuel MOX1 10.11 94.5 / 0.16174 u-234 0 u-236 0 pu-238 0.40232 pu-239 10.42187
                pu-240 4.78046 pu-241 1.77834 pu-242 1.22383 am-241 0.51632
```

Only oxide fuel can be defined on the *fuel* card. Metallic fuel is not supported.

[2]Earlier versions of the code used a different formula for the default ^{234}U concentration.

4. DEPLETION

This chapter describes depletion and working with restart files.

Depletion and restart files are only available with MPACT.

4.1 DEPLETION

Depletion refers to taking a step in time and calculating the change in number densities (isotopics) in the core.

A problem is depleted by including a *deplete* card in the [STATE] block, as in the following example:

```
[STATE]
  deplete EFPD 0.0 1.0 10.0 30.0
```

The first parameter on the card is the units used in the depletion and can be "EFPD" for effective full power days, "GWDMT" for giga-watt days per metric ton of initial heavy metal, or "hours". Following the unit is a list of depletion steps to take. Each depletion step is referred to as a "statepoint" calculation. The first depletion step must always be zero.

Listing multiple depletion steps on a single *deplete* card will deplete with all of the other values in the [STATE] block held constant. To change a state parameter between depletion steps (power, flow, etc.), the user can split the depletion over multiple [STATE] blocks. In the following example, the code depletes three statepoints at 50% power, changes the power to 100%, and depletes for four more statepoints. The depletion step at 10 EFPDs is run at both 50 and 100% power.

```
[STATE]
  power 50.0
  deplete EFPD 0.0 1.0 10.0
[STATE]
  power 100.0
  deplete EFPD 10.0 30.0 60.0 90.0
```

The automatic list generation feature described in Section 2.1 is especially useful when defining depletion cases. An example of a *deplete* card with automatic list generation is as follows:

```
deplete EFPD 0 1 5 <10..200x10>
```

4.2 WRITING RESTART FILES

A user will often want to run a depletion and save the isotopic data to a file that can be used to restart a calculation at a later time. This feature is useful if a calculation is long-running and needs to be divided into multiple cases. Other times, a user may want to save certain statepoints to go back and run perturbation or flux map calculations at the saved points.

The restart file includes **only** isotopic data needed to restart a calculation and data from the [STATE] block that the file was saved. The restart file does not include the geometry description, so a regular input deck must also be used. A user should set up an input deck for a fresh core and then use the restart file to overwrite the fresh isotopic concentrations with the isotopic concentrations on the restart file.

A restart file can be written at any statepoint by using a *restart_write* card,

```
restart_write filename restart_label
```

where “filename” is the name of the restart file, and “restart_label” is an arbitrary user label used to differentiate multiple statepoints written to the same file. Examples of restart labels include “100EFPD”, “HZP”, “22.56”, “100EFPD_ARO”, etc. A restart file can include multiple statepoints, as long as each one uses a different restart label.

If a *restart_label* card is used with a *deplete* card, then the restart file is written at the last exposure step of the depletion.

In the following example, a depletion is performed, and restart files are written at multiple statepoints.

```
[STATE]
  deplete EFPD 0.0
  restart_write  restart_cyc12.h5 "BOC"
[STATE]
  deplete EFPD 20 40 80 100
 ! restart file is written at last exposure step on deplete card
  restart_write  restart_cyc12.h5 "100EFPD"
[STATE]
  deplete EFPD 150 200
  restart_write  restart_cyc12.h5 "200EFPD"
[STATE]
  deplete EFPD 250 300
  restart_write  restart_cyc12.h5 "300EFPD"
[STATE]
  deplete EFPD 350 400
  restart_write  restart_cyc12.h5 "400EFPD"
[STATE]
  op_date 1994/05/23       ! include shutdown date for EOC
  power 80.0
  deplete EFPD 423.4
  restart_write  restart_cyc12.h5 "EFPD423_EOC"
```

Another application of restart files is to write the final isotopic information at the end of cycle (EOC) so the data can be shuffled to a new cycle. (Core shuffles are discussed in a later section.) If writing a restart file at the EOC, the shutdown date should be included using the *op_date* card. The reason for including the shutdown date is so that the code will be able to calculate the isotopic decay during the outage. An example of the *op_date* card is shown in the last [STATE] block in the example above.

4.3 READING RESTART FILES

A restart file can be read by including a *restart_read* card in the [STATE] block,

```
restart_read filename restart_label
```

where “restart_label” is the label that was used to write the restart file. The *restart_read* card is used to restart an existing calculation; it is not used to do core shuffles.

In the following example, one of the restart files from the previous example is read, and a new calculation is performed with a different power and boron concentration.

```
[STATE]
  power 50.0
  boron 800
  restart_read restart_cyc12.h5 "200EFPD"
```

It is possible to write a statepoint in quarter-symmetry and then read the restart back in full-symmetry, or *vice versa*.

There is currently a restriction that a user should not include a *deplete* card in any [STATE] block where a restart is read. Instead, the user should divide the restart read and depletion into separate blocks, as shown here:

```
[STATE]
  restart_read restart_cycx.h5 "EFPD30"  ! read restart at 30 EFPD
[STATE]
  deplete EFPD 60 90
```

4.4 CORE SHUFFLING

A core shuffle occurs when fuel assemblies are rearranged in a core and/or new fuel is added to the core. Fuel assemblies discharged in previous cycles can be brought in from the fuel pool. Even fuel discharged from other units can be added (cross-unit shuffle).

When performing a core shuffle, the user must specify the location from which existing fuel assemblies were moved and what the new fuel assemblies look like.

When fuel isotopics are written to a restart file, the assembly locations are saved based on the *xlabel* and *ylabel* labels. The *xlabel*s start on the left side of the map and run horizontally. The *ylabel*s start at the top of the map and run down. For example, with the following labels defined:

```
[CORE]
  xlabel   R  P  N  M  L  K  J  H  G  F  E  D  C  B  A
  ylabel  01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
```

the assembly locations are defined as "xlabel dash ylabel":

```
                  L-01 K-01 J-01 H-01 G-01 F-01 E-01
          N-02 M-02 L-02 K-02 J-02 H-02 G-02 F-02 E-02 D-02 C-02
     P-03 N-03 M-03 L-03 K-03 J-03 H-03 G-03 F-03 E-03 D-03 C-03 B-03
     P-04 N-04 M-04 L-04 K-04 J-04 H-04 G-04 F-04 E-04 D-04 C-04 B-04
R-05 P-05 N-05 M-05 L-05 K-05 J-05 H-05 G-05 F-05 E-05 D-05 C-05 B-05 A-05
R-06 P-06 N-06 M-06 L-06 K-06 J-06 H-06 G-06 F-06 E-06 D-06 C-06 B-06 A-06
R-07 P-07 N-07 M-07 L-07 K-07 J-07 H-07 G-07 F-07 E-07 D-07 C-07 B-07 A-07
R-08 P-08 N-08 M-08 L-08 K-08 J-08 H-08 G-08 F-08 E-08 D-08 C-08 B-08 A-08
R-09 P-09 N-09 M-09 L-09 K-09 J-09 H-09 G-09 F-09 E-09 D-09 C-09 B-09 A-09
R-10 P-10 N-10 M-10 L-10 K-10 J-10 H-10 G-10 F-10 E-10 D-10 C-10 B-10 A-10
R-11 P-11 N-11 M-11 L-11 K-11 J-11 H-11 G-11 F-11 E-11 D-11 C-11 B-11 A-11
     P-12 N-12 M-12 L-12 K-12 J-12 H-12 G-12 F-12 E-12 D-12 C-12 B-12
     P-13 N-13 M-13 L-13 K-13 J-13 H-13 G-13 F-13 E-13 D-13 C-13 B-13
          N-14 M-14 L-14 K-14 J-14 H-14 G-14 F-14 E-14 D-14 C-14
                    L-15 K-15 J-15 H-15 G-15 F-15 E-15
```

The restart file also includes the cycle number (which is stored as a label), so the combination of the cycle number and location can be used to uniquely define any assembly location in any cycle. For example, "3K-12" refers to location "K-12" of cycle "3." If no cycle number is specified, then the cycle label defaults to the previous cycle number (i.e., cycle N-1). Therefore, cycle labels are limited to integers only.

New fresh assemblies are defined by using a plus sign followed by an optional string. (The string is not currently used for anything, but it may be used to refer to the fresh fuel assembly type in the future.) For example, "+ASMA" signifies a fresh fuel assembly.

Using these naming conventions, a new core loading pattern can be defined using a *shuffle_label* map. The *shuffle_label* map is a core map showing the the previous assembly locations and new assembly fuel types. It is also possible to shuffle assembly inserts independently in a similar fashion using a separate *insert_shuffle_label* map.

The following example is the full-core loading pattern for cycle 2 of the Benchmark for Evaluation And Validation of Reactor Simulations (BEAVRS) benchmark. The cycle numbers are not used in the location labels because all of the assemblies were moved from the previous cycle (cycle 1), and the default behavior is to use the previous cycle number if no cycle label is specified.

```
[CORE]
  cycle 2
  op_date 1996/03/02     ! cycle startup date
[STATE]
  shuffle_label
                     L-10 +X34 +X32 +X34 +X32 +X34 E-10
           G-10 +X32 +X32 L-02 P-12 N-03 B-12 E-02 +X32 +X32 J-10
      F-09 +X34 N-02 N-10 +X32 D-11 R-10 M-11 +X32 C-10 C-02 +X34 K-09
      +X32 P-03 L-08 +X32 M-09 E-15 G-08 L-15 D-09 +X32 H-05 B-03 +X32
 F-05 +X32 F-03 +X32 M-04 +X32 M-03 A-10 D-03 +X32 D-04 +X32 K-03 +X32 K-05
 +X34 P-05 +X32 G-04 +X32 N-08 R-09 G-14 A-09 H-03 +X32 J-04 +X32 B-05 +X34
 +X32 D-02 E-12 A-11 N-04 G-01 B-09 H-15 J-14 J-01 C-04 R-11 L-12 M-02 +X32
 +X34 N-13 F-15 H-07 F-01 B-07 A-08 F-14 R-08 P-09 K-15 H-09 K-01 C-03 +X34
 +X32 D-14 E-04 A-05 N-12 G-15 G-02 H-01 P-07 J-15 C-12 R-05 L-04 M-14 +X32
 +X34 P-11 +X32 G-12 +X32 H-13 R-07 J-02 A-07 C-08 +X32 J-12 +X32 B-11 +X34
 F-11 +X32 F-13 +X32 M-12 +X32 M-13 R-06 D-13 +X32 D-12 +X32 K-13 +X32 K-11
      +X32 P-13 H-11 +X32 M-07 E-01 J-08 L-01 D-07 +X32 E-08 B-13 +X32
      F-07 +X34 N-14 N-06 +X32 D-05 A-06 M-05 +X32 C-06 C-14 +X34 K-07
           G-06 +X32 +X32 L-14 P-04 C-13 B-04 E-14 +X32 +X32 J-06
                     L-06 +X34 +X32 +X34 +X32 +X34 E-06
```

The next example shows a quarter-core shuffle map. This map is not realistic, but it shows how fresh assemblies are inserted, along with assemblies from cycles 8, 19, 20, and 21. The fresh assemblies all have fuel type "A12."

```
[CORE]
  cycle 22               ! new cycle number
  op_date 2012/10/02     ! cycle startup date
[STATE]
  shuffle_label
     8H-10  +A12  21E-03  +A12  21E-13 21G-02 21G-08 21N-04
     +A12  21O-08 20C-04 19L-07  +A12  21E-06  +A12  21K-03
    21C-11 20D-03 21E-08  +A12  21M-04  +A12  21K-04 21A-07
     +A12  19G-10  +A12  21P-08  +A12  21O-06 21B-06
    21O-11  +A12  21D-11  +A12  21B-07  +A12  21G-11
    21B-09 21F-05  +A12  21F-13  +A12  21L-06
    21H-09  +A12  21D-09 21F-02 21M-07
    21D-04 21C-09 21G-01
```

At this time, there is a restriction that also requires the user to include an *assm_map* card in the input to specify the fresh fuel assemblies. This restriction will be removed in the future so that the fresh assembly types specified after the plus sign on the *shuffle_label* card will be used.

In addition to the loading patterns, a list of restart files must be included to define the restart search path. The order of the restart files is important: they must be in reverse chronological order.

```
restart_shuffle
  restart_file_12.h5 EOC12
  restart_file_11.h5 EOC11
  restart_file_10.h5 EOC10
  restart_file_5.h5  EOC5
```

The first restart file is used to define the "previous" cycle number. The cycle number from this file will be used as the default cycle number in the shuffle map. The code will search for the assembly on the first file. If the assembly is not found, then the code will go to the second restart file, and so on.

The next section gives an example of a core shuffle.

4.4.1 Core Shuffle Example

Consider an example of a core shuffle occuring at the beginning of cycle 3. There are two EOC restart files that have been written from cycles 1 and 2.

These examples are not complete; they show only the pertinent cards needed to perform the core shuffle.

The EOC 1 restart file was generated with the following input:

```
[CORE]
  cycle 1    ! could be any arbitrary string like CYC1, etc.
  xlabel   R  P  N  M  L  K  J  H  G  F  E  D  C  B  A
  ylabel  01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
[STATE]
  deplete EFPD ... 327.3   ! only last depletion date shown
  op_date "1993/03/01"     ! shutdown date
  restart_write restart_cyc1.h5 "EOC1"
[ASSEMBLY]
  ! this input includes a definition of assembly type ASMA
```

The EOC 2 restart file was generated with the following input:

```
[CORE]
  cycle 2
  xlabel   R  P  N  M  L  K  J  H  G  F  E  D  C  B  A
  ylabel  01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
[STATE]
  deplete EFPD ... 426.3
  op_date "1994/03/05"    ! shutdown date cycle 2
  restart_write restart_cyc2.h5 "EOC_with_coastdown"
[ASSEMBLY]
 ! this input includes a definition of assembly type ASMB
 ! and ASMA from cycle 1
```

The following input is used to shuffle to cycle 3:

```
[CORE]
  cycle 3
  op_date "1994/04/07"    ! start-up date of cycle 3
  xlabel   R  P  N  M  L  K  J  H  G  F  E  D  C  B  A
  ylabel  01 02 03 04 05 06 07 08 09 10 11 12 13 14 15

[STATE]
  shuffle_label
   1H-10 +ASMC  E-03 +ASMC  E-13  G-02  G-08  N-04
   +ASMC  O-08  C-04  L-07 +ASMC  E-06 +ASMC  K-03
    C-11  D-03  E-08 +ASMC  M-04 +ASMC  K-04  A-07
   +ASMC  G-10 +ASMC  P-08 +ASMC  O-06  B-06
    O-11 +ASMC  D-11 +ASMC  B-07 +ASMC  G-11
    B-09  F-05 +ASMC  F-13 +ASMC  L-06
    H-09 +ASMC  D-09  F-02  M-07
    D-04  C-09  G-01

 ! One assembly was loaded from cycle 1 (in the center)
 ! This assembly had to have the cycle number prepended to it

 ! All of the other assemblies came from cycle 2.  This is the default cycle,
 ! and the cycle number did not have to be prepended.

 ! restart using the EOC restart files from cycles 1 and 2
  restart_shuffle
     restart_cyc2.h5 EOC_with_coastdown
     restart_cyc1.h5 EOC1

[ASSEMBLY]
 ! include descriptions for ASMA, ASMB, ASMC if they are
 ! all used in cycle 3
```

4.4.2 Shutdown Decay

When performing a core shuffle, a shutdown decay is performed on each assembly to account for the shutdown decay time. The shutdown decay calculation is important for calculating the decay and buildup of

fission products such as xenon and samarium.

The shutdown decay time is calculated using the shutdown date from when the assembly was discharged and the new cycle startup date. The discharge date is the *op_date* on the restart file the assembly data was written. The cycle startup date is the *op_date* in the core shuffle deck.

4.4.3 Cross Unit Shuffle

The shuffling methodology can support cross-unit shuffles.

To use cross-unit shuffling, the unit number must be specified in the [CORE] block.

```
unit 1    ! unit 1 of a 2 unit site
```

To read an assembly from a different unit, the unit label is prepended to the front of the location label in the *shuffle_label* card using a colon. For example, "U2:C3G-04" is used to read the assembly from Unit "U2," cycle "C3," and location "G-04."

Once the location labels have been defined, the user can mix and match restart files from different units in the *restart_shuffle* card:

```
restart_shuffle
  restart_file_U1_12.h5 EOC12
  restart_file_U2_5.h5  EOC
  restart_file_U1_11.h5 EOC11
  restart_file_U2_4.h5  EOC
  restart_file_U1_10.h5 EOC10
  restart_file_U2_3.h5  EOC
  restart_file_U1_5.h5  EOC5
```

The only "trick" is to list the restart points in the correct reverse chronological order since an assembly could theoretically go from U2:CYC3 to U1:CYC10 and then back to U2:CYC6. Therefore, the restarts must be in the correct reverse chronological order. Remember that the cycle numbers are arbitrary strings, so there is no natural "order" to them. The order is defined by the order specified in the *restart_shuffle* input.

The shutdown dates are written to each restart file so that the shutdown decay will be correctly calculated for each assembly. It does not matter what unit the assembly came from—the correct shutdown dates will be used.

5. INPUT CARD DESCRIPTIONS

This chapter contains a complete list of the available input cards.

The input for each block is given in separate subsections.

In this chapter, input cards are given in **bold** text followed by the parameters on the card. Following each input card is a description of the parameters on that card.

5.1 BLOCK CASEID

title case_name

case_name	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Problem name		
Notes: None		

5.2 BLOCK STATE

title state_name

state_name	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: State name		
Notes: None		

op_date operating_date

operating_date	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): Limited to “MM/DD/YYYY” or “YYYY/MM/DD”		
Description: This card contains the operating date of this statepoint. It is used when writing restart files. The operating date must be entered for any restart file that is used in a core shuffle so that the isotopic decay can be calculated during an outage		
Notes: None		

power percent_power

percent_power	Float	Optional
Units: Percent (default)		
Applicable Value(s): 1.0×10^{-8} (default), ≥ 0		
Limitation(s): None		
Description: Percent of rated operating power		
Notes: Cannot be zero when depleting		

flow percent_flow

percent_flow	Float	Optional
Units: Percent (default)		

continued on next page...

`percent_flow`, continued...

Applicable Value(s): 1.0×10^{-8} (default), ≥ 0
Limitation(s): None
Description: Percent of rated operating flow
Notes: None

flow_dist nominal_flow_multiplier

`nominal_flow_multiplier`	2D Float Map	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: This is a 2D array that must match the shape of assm_map in [CORE]. It gives a multiplier that will be applied to nominal inlet mass flow rate in each assembly		
Notes: This map is not normally used		

blade_pos blade_pos

`blade_pos`	2D Float Map	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Steps withdrawn for each blade location		
Notes: None		

pout_dist outlet_pressure_adder

`outlet_pressure_adder`	2D Float Map	Optional
Units: psi (default)		
Applicable Value(s): 0.0 (default), Any float		
Limitation(s): None		
Description: This is a 2D array that must match the shape of assm_map in [CORE]. It gives an adder that will be added to nominal outlet pressure in each assembly		
Notes: This map is not normally used		

bypass bypass_option

`bypass_option`	Float or String	Optional
Units: N/A, Percent		
Applicable Value(s): 0 (default), ≥ 0, table		
Limitation(s): None		
Description: This is the bypass flow fraction applied to the actual flow, or the word "table," indicating that a bypass flow rate table specified in [CORE] should be used		
Notes: None		

tinlet inlet_temperature units

`inlet_temperature`	Float or String	Optional
Units: °C (default), °F, K		
Applicable Value(s): 326.85 °C (default), > 0, table		
Limitation(s): None		
Description: This is the core inlet temperature in given units. Examples of this card are `tinlet 560 F` or `tinlet 600 K`. Alternatively, `tinlet table` can be used to invoke the tinlet_table		
Notes: This is required when coupling to CTF		

tinlet_dist inlet_temperature_adder

`inlet_temperature_adder`	2D Float Map	Optional
Units: C (default)		
Applicable Value(s): 0 (default)		
Limitation(s): None		
Description: This is a 2D array that must match the shape of assm_map in [CORE]. It gives an adder that will be applied to the nominal inlet temperature in each assembly		
Notes: This map is not normally used		

subcool inlet_subcooled_enthalpy

`inlet_subcooled_enthalpy`	Float	Optional
Units: BTU/lbm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): Cannot be used along with `tinlet` card. Requires `pressure` card		
Description: The enthalpy below the saturated liquid enthalpy at the inlet		
Notes: `tinlet` or `subcool` cards are required and are exclusive		

void void_distribution

`void_distribution`	Float	Optional
Units: Percent (default)		
Applicable Value(s): > 0, < 100		
Limitation(s): None		
Description: Assembly-wise radial void distribution in percent		
Notes: BWR only		

axial_void axial_void_map axial_void_bounds

`axial_void_map`	Float	Optional
Units: Percent (default)		
Applicable Value(s): ≥ 0, ≤ 100		
Limitation(s): None		
Description: List of axial void fractions		
Notes: BWR only		

axial_void_bounds	Float	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: List of axial elevations for axial void fractions		
Notes: BWR only		

tfuel fuel_temperature units

fuel_temperature	Float	Optional
Units: K (default), °F, °C		
Applicable Value(s): 600 K (default), $> 0K$, $< 1600K$		
Limitation(s): None		
Description: Fixed fuel temperatures		
Notes: This is only used if feedback is turned OFF. Examples of this card are `900 K` or `1200 F`		

modden mod_density

mod_density	Float	Optional
Units: g/cc (default)		
Applicable Value(s): 0.743 (default), > 0.01, < 1.2		
Limitation(s): None		
Description: Fixed moderator density		
Notes: Only used if feedback is turned OFF		

xenon xenon_treatment

xenon_treatment	Character String	Optional
Units: N/A		
Applicable Value(s): dep (default), zero, equil		
Limitation(s): None		
Description: Xenon treatment option: • `zero`: sets ^{135}I and ^{135}Xe number densities to zero • `equil`: sets ^{135}I and ^{135}Xe number densities to calculated equilibrium values • `dep`: treats ^{135}I and ^{135}Xe explicitly as other isotopes in transport calculation		
Notes: None		

samar samarium_treatment

samarium_treatment	Character String	Optional
Units: N/A		
Applicable Value(s): dep (default), zero, equil, peak		
Limitation(s): None		

continued on next page...

`samarium_treatment`, continued...

Description: Samarium treatment option: • `zero`: sets ^{149}Pm and ^{149}Sm number densities to zero • `equil`: sets ^{149}Pm and ^{149}Sm number densities to calculated equilibrium values • `dep`: treats ^{149}Pm and ^{149}Sm explicitly as other isotopes in transport calculation • `peak`: adds ^{149}Pm number density to ^{149}Sm number density and then sets ^{149}Pm number density to zero
Notes: None

rlx_xesm Xe-Sm_relaxation

`Xe-Sm_relaxation`	Float	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), > 0		
Limitation(s): None		
Description: Xenon-samarium equilibrium relaxation factor		
Notes: Recommend value: 1.0		

pred_order predictor_order

`predictor_order`	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), ≥ 0		
Limitation(s): None		
Description: This card is used to specify the order of polynomial approximation to use for extrapolation of microscopic cross sections and fluxes over predictor depletion substeps		
Notes: The methodology employed for high-order depletion is described in the article by G. G. Davidson et al., "Nuclide Depletion Capabilities in the Shift Monte Carlo Code," *Annals of Nuclear Energy*, 114, pp. 259–276 (2018). For any given timestep, the code will attempt the highest polynomial order approximation without exceeding the user specification, as is allowed by the generated data thus far. For example, if the user designates order 2, then on the first timestep, order 0 will be used since no previous time data are available. On the second timestep, order 1 will be used since only one previous set of time data is available, and on the third and subsequent timesteps order 2 will be used since sufficient data from previous timesteps are available to perform an order 2 fit		

corr_order corrector_order

`corrector_order`	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), ≥ 0		
Limitation(s): None		
Description: This card is used to specify the order of polynomial approximation to use for interpolation of microscopic cross sections and fluxes over corrector depletion substeps		
Notes: This follows the same methodology as described for `pred_order`		

boron boron_concentration

boron_concentration	Float	Optional
Units: ppm (default)		
Applicable Value(s): 0.0 (default), ≥ 0		
Limitation(s): None		
Description: Soluble boron concentration in the moderator		
Notes: None		

b10 b10_fraction b10_depletion

b10_fraction	Float	Optional
Units: N/A, Atom fraction of B-10 in boron		
Applicable Value(s): 0.199 (default), ≥ 0		
Limitation(s): None		
Description: Boron-10 fraction in coolant		
Notes: None		

b10_depletion	Boolean	Optional
Units: N/A		
Applicable Value(s): False (default), True		
Limitation(s): None		
Description: Flag to enable B-10 depletion in coolant		
Notes: Required when using input parameter b10		

kcrit target_eigenvalue

target_eigenvalue	Float	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), ≥ 0		
Limitation(s): None		
Description: Target eigenvalue used in boron search or rod search		
Notes: None		

search search_option

search_option	Character String	Optional
Units: N/A		
Applicable Value(s): keff (default), boron, rod		
Limitation(s): None		
Description: Search option		
Notes: None		

search_bank rod_search_bank

rod_search_bank	Character String	Optional

continued on next page...

`rod_search_bank`, continued...

Units: N/A
Applicable Value(s):
Limitation(s): None
Description: Control rod bank to be moved in rod search problems
Notes: Required when input parameter `search` is set to `rod`

pressure outlet_pressure

`outlet_pressure`	Float	Optional
Units: psia (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Core exit pressure		
Notes: This is required for BWR calculations. For PWR calculations, the default is 2,250 psia. When using simplified TH calculations for PWRs, this value must be 2,250 psia if it is specified		

deplete deplete_units depletion_steps

`deplete`	Float	Optional
Units: N/A, GWDMT, MWDMT, EFPD, hours		
Applicable Value(s): `depletion_steps` must be listed in ascending order		
Limitation(s): None		
Description: Specification of depletion units and a single or multiple depletion steps		
Notes: Recommended that depletion step sizes are less than 1 GWDMT, 1,000 MWDMT, or 30 EFPD		

jump_in_file file_name

`jump_in_file`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Name of h5 file containing data needed to perform detailed isotopic burn for cycle jump in. The inclusion of this card/file name will initiate an ORIGEN point depletion based “burn in” according to the data in the h5 file		
Notes: ORIGEN must be present and enabled in order to use this feature		

edit state_edits

`state_edits`	Character String	Optional
Units: N/A		

continued on next page...

state_edits, continued...

Applicable Value(s): pin_powers, pin_exposures, core_flux, pin_fuel_temp, pin_isotopes_Xe-135, pin_mod_dens (if PWR), pin_mod_temp (if PWR), pin_cool_dens (if BWR), pin_cool_temp (if BWR), pin_cool_void (if BWR) (default), none, pin_isotopes_<ZZ-AAA> (isotope "all"), intrapin_<mat_type>_isotopes_<ZZ-AAA> (isotope "all"), fsr_isotopes_<ZZ-AAA> (isotope "all"), point_isotopes_<ZZ-AAA> (isotope "all"), pin_flux_<GGG> ("fast" "thermal" "2g" 3-digit group omit for all groups), fsr_flux_<GGG> ("fast" "thermal" "2g" 3-digit group omit for all groups), point_flux_<GGG> ("fast" "thermal" "2g" 3-digit group omit for all groups), core_flux, intrapin_<mat_type>_<edit_type> (mat_type: fuel gap clad mod cool edit_type: dens temp volume isotopes_<ZZ-AAA>) intrapin_fluence, intrapin_exposure, fsr_delayed-chi, fsr_dnpy
Limitation(s): None

continued on next page...

state_edits, continued...

Description: This is a list of state variables to be edited. By default, MPACT edits pin_powers, pin_exposures, core_flux, pin_fuel_temp, and pin_isotopes_Xe-135. For PWR cases, pin_mod_dens and pin_mod_temp are added to the default edits; for BWR cases, pin_cool_dens, pin_cool_temp, and pin_cool_void are added to the default edits. Once an edit is enabled, it remains enabled until all edits are disabled using the none option. If the none option appears in the same state as other edit names, then all previously enabled edits are disabled first, and then the edits named at the same state as none are enabled. The default edits are never disabled, even if none is specified.

Individual isotopes can be edited using pin_isotopes_ followed by the isotope in ZZ-AAA format. For example, pin_isotopes_U-235. The same is true for intrapin_<mat_type>_isotopes_, fsr_isotopes_, and point_isotopes_. Similarly, group fluxes can be edited using pin_flux_ followed by a 3-digit group identifier. For example, pin_flux_001 gives the fast group, and pin_flux_051 gives the most thermal group when using a 51-group cross section library. The same is true for point_flux_ and fsr_flux_. It should be noted that the core_flux edit automatically implies the core-averaged flux for each group, so core_flux_001 is not a valid option.

Intrapin edits may be requested using the following entry format intrapin_<mat_type>_<edit_type>. Currently supported mat_type values include fuel, gap, clad, mod, and cool. intrapin_<edit_type> may also be used for editing all material intrapin values. Currently supported edit_type values include dens, temp, volume, and isotopes_. Two additional intrapin edits—intrapin_fluence for the fluence of all intrapin regions, and intrapin_exposure for the intrapin fuel burnup distribution—are also supported. When intrapin edits are requested, they are compressed into a 1D array in the output HDF5 file. This 1D array can be translated into a more traditional core map representation using the accompanying intrapin_<mat_type>_PinFirstRegionIndex and intrapin_<mat_type>_PinNumRegions datasets generated for each mat_type requested. intrapin_<mat_type>_PinFirstRegionIndex indicates the starting index of a given pin location in the 1D array, and intrapin_<mat_type>_PinNumRegions indicates the number of values associated with said pin that occur sequentially in the 1D array starting at the corresponding intrapin_<mat_type>_PinFirstRegionIndex starting index.

MPACT provides a number of collections of edits for ease of use:

- pin_flux_2g implies both pin_flux_fast and pin_flux_thermal. The same is true for point_flux_2g and fsr_flux_2g
- pin_flux implies pin_flux_001, pin_flux_002, etc., up to the number of energy groups in the problem. The same is true for point_flux and fsr_flux
- pin_isotopes_all implies every isotope that exists in the transport and depletion libraries. The same is true for fsr_isotopes_all, point_isotopes_all, and intrapin_isotopes_all
- fsr_delayed-chi implies the delayed chi edits for each delayed neutron group
- fsr_dnpy implies the delayed neutron precursor yields edits for each delayed neutron group
- fsr_dnpy implies the delayed neutron precursor decay constants edits for each delayed neutron group

Users can define "edit groups" using the edit_group option. These can be specified in this card as well to enable whole groups of edits. This is a convenient shorthand for when a particular set of edits will be enabled and disabled multiple times during a simulation

continued on next page...

state_edits, continued...

Notes: One limitation in this card occurs when a user enables an edit using its name, a user-defined group, and an internal group of edits in MPACT all in the same calculation. For example, enabling pin_flux_001 by referring to pin_flux_001, pin_flux, and a user-defined edit_group option that includes pin_flux_001 will result in unpredictable behavior and potentially crash the code. It is recommended that if any of the above built-in shorthand options are used, then the user should not name any of their components inside an edit_group option. However, in this example, if the edit_group option included pin_flux instead of pin_flux_001, then the references would be correctly resolved

reset_sol solution_reset_bool

solution_reset_bool	Boolean	Optional
Units: N/A		
Applicable Value(s): False (default), True		
Limitation(s): None		
Description: Resets the initial guess of the flux in MPACT		
Notes: None		

rodbank bank_labels bank_pos

bank_labels	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a list of control rod banks to position. The labels correspond to crd_map in CORE block		
Notes: Every bank_label must have a corresponding bank_pos		

bank_pos	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Steps withdrawn for each bank in list		
Notes: Every bank_pos must have a corresponding bank_label. Example: rodbank SA 228 SB 50 SD 0 A 228		

feedback feedback_option

feedback_option	Character String	Optional
Units: N/A		
Applicable Value(s): off (default), on		
Limitation(s): None		
Description: Flag to turn on and off T/H feedback		
Notes: None		

fuel_performance fuel_performance_option

fuel_performance_option	Character String	Optional
Units: N/A		
Applicable Value(s): off (default), on		
Limitation(s): None		
Description: Flag to turn on and off BISON coupling		
Notes: None		

crud crud_option

crud_option	Character String	Optional
Units: N/A		
Applicable Value(s): off (default), on		
Limitation(s): None		
Description: Flag to turn on and off MAMBA CRUD deposition coupling		
Notes: None		

excore_transport excore_transport_option

excore_transport_option	Character String	Optional
Units: N/A		
Applicable Value(s): off (default), on		
Limitation(s): None		
Description: Flag to turn on and off Shift excore transport coupling		
Notes: Additional SHIFT options are included in SHIFT block		

thexp thermal_expansion_option

thermal_expansion_option	Character String	Optional
Units: N/A		
Applicable Value(s): on (default), off		
Limitation(s): None		
Description: Flag to turn on and off thermal expansion		
Notes: Additional thermal expansion options are given on other input cards		

thexp_tfuel fuel_thermal_expansion_temperature units

fuel_thermal_expansion_temperature	Float	Optional
Units: K (default), F, C		
Applicable Value(s): 293 K (default)		
Limitation(s): None		
Description: This is the temperature to use for thermal expansion of fuel. If not present, `tfuel` is used instead. If both `thexp_tfuel` and `tfuel` are not specified, `tinlet` will be used		
Notes: Example: `900 K`		

thexp_tclad clad_thermal_expansion_temperature units

clad_thermal_expansion_temperature	Float	Optional
Units: K (default), F, C		
Applicable Value(s): 293 K (default)		
Limitation(s): None		
Description: This is the temperature to use for thermal expansion of clad. If not present, thexp_tmod is used instead. If both thexp_tfuel and thexp_tmod are not specified, tinlet will be used		
Notes: Example: 560 F		

thexp_tmod moderator_thermal_expansion_temperature units

moderator_thermal_expansion_temperature	Float	Optional
Units: N/A, F, C		
Applicable Value(s): 293 K (default)		
Limitation(s): None		
Description: This is the temperature to use for thermal expansion of moderator and structural materials. If not present, tinlet is used instead		
Notes: Example: 560 F		

expand3D 3D_thermal_expansion_option

3D_thermal_expansion_option	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This is an option to perform 3D thermal expansion. If set to false, thermal expansion will only be performed in the radial direction. When set to true, both radial and axial thermal expansion will be performed		
Notes: None		

thexp_outfile thermal_expansion_outfile

thermal_expansion_outfile	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the name of the thermally expanded XML output. If the name of the outfile is the same as the XML input used to execute MPACT, the input file will be renamed to input_filename.bak, and the thermally expanded XML output will be in the output file. If not specified, no thermally expanded XML output file will be generated		
Notes: None		

thexp_info thermal_expansion_info

thermal_expansion_info	Boolean	Optional
Units: N/A		

continued on next page...

`thermal_expansion_info`, continued...

Applicable Value(s): false (default), true
Limitation(s): None
Description: Logical flag to edit additional thermal expansion information to the output file
Notes: None

apitch_tec assembly_pitch_expansion_coefficient

`assembly_pitch_expansion_coefficient`	Float	Optional
Units: K^{-1} (default)		
Applicable Value(s): $\geq 0, < 50.0 \times 10^{-6}$		
Limitation(s): None		
Description: This is the thermal expansion coefficient to be used when expanding the assemblies in the problem. If not specified, the expansion coefficient will be calculated internally assuming a core plate nominal density for SS 304		
Notes: None		

ppitch_tec pin_pitch_expansion_coefficient

`pin_pitch_expansion_coefficient`	Float	Optional
Units: K^{-1} (default)		
Applicable Value(s): $\geq 0, < 50.0 \times 10^{-6}$		
Limitation(s): None		
Description: This is the thermal expansion coefficient to be used when expanding the pins in the problem. If not specified, the expansion coefficient will be calculated internally assuming Zircaloy-4 for grid materials		
Notes: None		

axial_tec axial_tec

`axial_tec`	Float	Optional
Units: K^{-1} (default)		
Applicable Value(s): $\geq 0, < 50.0 \times 10^{-6}$		
Limitation(s): None		
Description: This is the thermal expansion coefficient to be used when expanding the axial dimension of the problem. If not specified, the expansion coefficient will be calculated internally using the UO_2 thermal expansion coefficient and the fuel temperature. This is only done if 3D expansion is enabled		
Notes: None		

sym symmetry_option

`symmetry_option`	Character String	Optional
Units: N/A		
Applicable Value(s): full (default), qtr		
Limitation(s): None		

continued on next page...

symmetry_option, continued...

Description: This is an option for specifying the symmetry of the problem. The full option specifies that the problem will be modeled in full and that ray tracing will be performed across the whole geometry. The qtr option will only model the south-east quarter of the geometry. In quarter-symmetry, the boundary conditions along the symmetry boundary are determined by the bc_sym card
Notes: For multistate simulations, if sym is not specified in the first state, any sym options specified in future states will be ignored

kmul_beta kmul_beta

kmul_beta	Float	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), Real numbers on the interval (0.0,1.0]		
Limitation(s): Cannot specify exactly 0		
Description: This option is used to specify the direct multiplier on beta, the delayed neutron fraction. This option is used to apply conservatism to transient calculations specifically for RIA		
Notes: None		

kmul_doppler kmul_doppler

kmul_doppler	double	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), Real numbers on the interval (0.0,1.0]		
Limitation(s): Cannot specify exactly 0		
Description: This option is used to specify the direct multiplier on the temperature difference that the fuel experiences when evaluating cross sections. It is used to apply conservatism to transient calculations. It can be used in steady-state calculations to iterate to the desired value		
Notes: None		

kmul_modtemp kmul_modtemp

kmul_modtemp	double	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), Real numbers on the interval (0.0,1.0]		
Limitation(s): Cannot specify exactly 0		
Description: This option is used to specify the direct multiplier on the temperature difference that the moderator experiences when evaluating cross sections. It is used to apply conservatism to transient calculations. It can be used in steady-state calculations to iterate to the desired value		
Notes: None		

kmul_crw kmul_crw

kmul_crw	double	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), Real numbers on the interval (0.0,1.0]		
Limitation(s): Cannot specify exactly 0		

continued on next page...

kmul_crw, continued...

Description: This option is used to specify the direct multiplier on the critical rod worth. This option is used to apply conservatism to transient calculations specifically for RIA
Notes: None

scram_type scram_type scram_rate scram_time

scram_type	Free Form Character String, Pairs of Doubles	Optional
Units: false (default), true		
Applicable Value(s):		
Limitation(s): Can only be used for transient cases, and at least one "trip_" card must be present to specify trip conditions		
Description: This option is used to specify the scram type (of which the only current option is "trip") and the scram bank movement speed intervals. These are specified using rate/time pairs, where each rate is associated with the following time interval. The units are RUs/second and seconds, respectively. At least one time/rate pair must be present		
Notes: This card is only used in transient calculations		

bank_wd bank_wd

bank_wd	List of Free Form Character Strings Paired with an Integer and Float	Optional
Units: N/A		
Applicable Value(s): Any control rod bank label and real values > 0		
Limitation(s): Currently only works for one control rod bank and for transient cases		
Description: This card is used to specify the position of a control rod bank at a specified time. There is no limit to the number of time and position pairs, and at least one pair must exist. • bank_label: the bank to be withdrawn • time_N: the selected transient time that corresponds to the bank position • pos_N: the number of steps withdrawn at a given time step		
Notes: This feature is still under development and is not yet operational		

scram_lock bank_label

bank_label	List of Free Form Character Strings	Optional
Units: N/A		
Applicable Value(s): At least one valid bank label		
Limitation(s): Requires scram_type card		
Description: This option is used to specify the bank labels of the banks that will not participate in the scram. These banks will continue their normal movement. At least one bank label must be specified		
Notes: This card is only used in transient calculations		

trip_time trip_time

trip_time	double	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): Requires scram_type card		
Description: This option is used to specify the simulation time when a trip will occur (in seconds) for scram functionality		
Notes: This card is only used in transient calculations		

trip_power high_power low_power delay number_detectors

trip_power	double, double, double, integer	Optional
Units: N/A		
Applicable Value(s): > 0, ≥ 0, ≥ 0, ≥ 0		
Limitation(s): Requires scram_type card		
Description: This option is used to specify the trip power conditions for scram functionality. The high and low power entries are in units of % full power. The delay entry is the specified time after delay before scram bank movement occurs (seconds). The last entry is the number of detectors required to meet these conditions before a trip occurs (currently, only the full-core power can be assessed, with an option of 0)		
Notes: This card is only used in transient calculations		

trip_rate upper_power_threshold lower_power_threshold delay number_detectors

trip_rate	double, double, double, integer	Optional
Units: N/A		
Applicable Value(s): > 0, ≥ 0, ≥ 0, ≥ 0		
Limitation(s): Requires scram_type card		
Description: This option is used to specify the trip power rate change conditions for scram functionality. The first two entries are the upper and lower power rate change thresholds in units of % full power/second. The delay entry is the specified time after delay before scram bank movement occurs (seconds). The last entry is the number of detectors required to meet these conditions before a trip occurs (currently, only the full-core power can be assessed, with an option of 0)		
Notes: This card is only used in transient calculations		

restart_shuffle_errorchecking restart_shuffle_errorchecking

restart_shuffle_errorchecking	character string	Optional
Units: N/A		
Applicable Value(s): on (default), off, true, false		
Limitation(s): None		
Description: This card is used to toggle more thorough geometry input checking between the shuffle_label map and the assm_map. If the unexpanded xml input parameter list is stored on the restart file, then the assembly parameter list at a shuffle label position is compared to the corresponding assembly parameter list at the same core assm_map position		
Notes: None		

restart_jumpin target_location restart_file restart_label source_location

restart_jumpin	Array of Strings	Optional
Units: N/A		
Applicable Value(s): None (default)		
Limitation(s): None		
Description: This card is used to specify sets of assembly isotopic data for assembly batches that do not have full simulation histories. These assemblies have approximated histories, so a user can "jump in" to any later cycle without explicitly simulating all previous cycles. The user is required to specify all of the following parameters. • target_location: location to load isotopics in current model • restart_file: the end time of perturbation • restart_label: restart label in restart file with assembly data • source_location: core label coordinate for assembly position when restart data were written		
Notes: This is a multiline input, so multiple entries may be given		

restart_shuffle restart_shuffle_file restart_shuffle_label

restart_shuffle	arrays of character strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: To perform a restart shuffle, the user is required to specify the restart files to use, as well as the labels from within those files to use during the shuffle. They must be listed in matching file-label pairs		
Notes: See Section 4.4 for more information and examples		

restart_read restart_read_file restart_read_label

restart_read	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: To perform a restart, the user is required to specify the restart file to use, as well as the label from that file to use to begin the restart. The file and label must be listed as a matching file-label pair		
Notes: See Section 4.3 for more information and examples		

restart_write restart_write_file restart_write_label

restart_write	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: To write a restart file, the user is required to specify a restart file name to write to, as well as a label to call that state in the restart. The file and label must be listed as a matching file-label pair		
Notes: See Section 4.2 for more information and examples		

restart_isotope_set restart_isotope_set_option

restart_isotope_set_option	Character String	Optional
Units: N/A		
Applicable Value(s): transport (default), depletion		
Limitation(s): None		
Description: This option selects the isotope set to be edited to the restart file		
Notes: None		

shuffle_label shuffle_label

shuffle_label	2D map of character strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The label of the assembly or assemblies to be used in the restart shuffle. The shape of the shuffle_label must match core_shape, and any assembly that is not to be shuffled uses a - in place of the assembly label to maintain the core_shape		
Notes: See Section 4.4 for more information and examples		

insert_shuffle_label insert_shuffle_label

insert_shuffle_label	2D map of character strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The label of the assembly inserts to be used in the restart shuffle. The shape of insert_shuffle_label must match core_shape, and any inserts that are not to be shuffled use a - in place of the insert label to maintain the core_shape		
Notes: None		

shuffle_homog shuffle_homog

shuffle_homog	Character String	Optional
Units: N/A		
Applicable Value(s): none (default), center, all		
Limitation(s): None		
Description: The homogenization option for quarter-symmetric restart shuffle cases. By default, no homogenization occurs. If the center option is used, the center assembly alone will be homogenized, and then a quarter of it is used in the calculation with reflective boundary conditions. The all option does not currently have a function		
Notes: None		

crud_cleaning crud_cleaning_map

crud_cleaning_map	2D Float Map	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a 2D array that must match the shape of assm_map in [CORE]. Map specifies the assembly-wise crud cleaning fractions. For any assemblies that are not to be cleaned, use a dash "-" in place of the cleaning fraction		
Notes: For shuffle cases only		

crud_removal crud_removal

crud_removal	Float	Optional
Units: N/A		
Applicable Value(s): $\geq 0, \leq 1$		
Limitation(s): None		
Description: Core-wide crud removal fraction		
Notes: Does not carry over from state to state		

crud_replenish_b10 crud_replenish_b10

crud_replenish_b10	Float	Optional
Units: N/A		
Applicable Value(s): $\geq 0, \leq 1$		
Limitation(s): None		
Description: Crud b10 replenishment fraction. A value of 1 resets the crud b10 to the coolant b10 ratio		
Notes: Does not carry over from state to state		

cool_chem h_conc li_conc ni_sol ni_par fe_sol

cool_chem	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Coolant chemistry concentrations to be used for crud formation: • h_conc: dissolved hydrogen in coolant [ppm] • li_conc: coolant lithium concentration [ppm] • ni_sol: coolant soluble nickel concentration [ppb] • ni_par: coolant particulate nickel concentration [ppb] • fe_sol: coolant soluble iron concentration [ppb]		
Notes: None		

vh2 h2_specific_volume

h2_specific_volume	Float	Optional
Units: N/A		

continued on next page...

h2_specific_volume, continued...

Applicable Value(s): ≥ 0
Limitation(s): None
Description: Specific volume of hydrogen in the coolant to be used for crud formation
Notes: None

ni_s soluble_ni_concentration

soluble_ni_concentration	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Soluble nickel concentration in the coolant to be used for crud formation		
Notes: None		

ni_p particulate_ni_concentration

particulate_ni_concentration	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Particulate nickel concentration in the coolant to be used for crud formation		
Notes: None		

cleanup_flow cleanup_flow

cleanup_flow	Float	Optional
Units: Percent (default)		
Applicable Value(s): ≥ 0, ≤ 100		
Limitation(s): None		
Description: Percent of rated chemistry cleanup flow rate		
Notes: Only used when coupled to CTF		

temp_pert temperature_multiplier temperature_adder

temp_pert	Float	Optional
Units: C(adder) (default)		
Applicable Value(s):		
Limitation(s): None		
Description: A multiplier and adder to be used to perform fuel temperature perturbations. The variables are used in the following equation: `perturbTemp=fuelTemp*multiplier+adder`		
Notes: This option is only used when using fuel temperature tables		

5.3 BLOCK CORE

name core_name

core_name	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Name of the reactor core		
Notes: None		

cycle cycle_num

cycle_num	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): Currently, only integers are allowed for the cycle label		
Description: Cycle number		
Notes: None		

unit unit

unit	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the reactor plant unit name. It is only used for multi-unit sites with cross-unit shuffle		
Notes: None		

op_date operation_date

operation_date	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): Limited to “MM/DD/YYYY” or “YYYY/MM/DD”		
Description: Startup date of core reload		
Notes: Only used when performing core shuffle		

size core_size

core_size	Integer	Required
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Number of assemblies across one axis in full-core geometry		
Notes: None		

rated rated_power rated_flow

rated_power	Float	Required
Units: MW (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Rated thermal power at 100% power		
Notes: None		

rated_flow	Float	Required
Units: Mlbs/hr (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Rated vessel flow at 100% flow		
Notes: None		

bypass_flow_table bypass_percent

bypass_flow_table	2D Float Table	Optional
Units: Percent (default)		
Applicable Value(s): 0 (default), ≥ 0		
Limitation(s): None		
Description: This is a 2D array with % rated power as the left-most column, % rated flow as the top row, with the remainder of the table being bypass flow rate as a percent of full-core flow at the corresponding data points. For example: bypass_flow_table 30 70 100 105 50 13.3 14.2 14.8 15.0 100 13.0 13.9 14.5 14.7 105 12.9 13.8 14.4 14.6		
Notes: BWR only		

tinlet_table percent_power inlet_temperature units

tinlet_table	2D Float/String Table	Optional
Units: N/A		
Applicable Value(s): ≥ 0, ≥ 0, C|K|F		
Limitation(s): Number of rows must be > 1		

continued on next page...

`tinlet_table`, continued...

Description: This is a 2D array with % rated power as the left-most column, temperature as the center column, and the right-most column being units. All units must match. For example: tinlet_table 20 598 K 50 595 K 70 593 K 90 590 K 100 587 K
Notes: None

rcs_volume rcs_volume

`rcs_volume`	Float	Optional
Units: m^3 (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Volume of the reactor coolant system		
Notes: Only used with B-10 depletion		

apitch apitch

`apitch`	Float	Required
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Assembly pitch		
Notes: None		

baffle baffle_mat baffle_gap baffle_thick

`baffle_mat`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Baffle material		
Notes: The `baffle` input is not valid for BWR calculations		

`baffle_gap`	Float	Optional
Units: cm (default)		
Applicable Value(s): = 0, or ≥ assembly gap and ≤ apitch − assembly gap		
Limitation(s): None		
Description: Gap between outside assembly (including assembly gap) and baffle		

continued on next page...

baffle_gap, continued...

Notes: The end of the baffle_gap must not fall in the assembly gap portion of the reflector assembly, as specified above. Additionally, baffle_gap + baffle_thick must also not fall in the assembly gap portion of the assembly. Their sum must be exactly 0.0, exactly the assembly pitch, or between the assembly gap and the assembly pitch minus the assembly gap. In this context, the assembly gap is defined as $\frac{apitch-npin*ppitch}{2}$

baffle_thick	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Thickness of baffle		
Notes: Restrictions on baffle_thick values are dependent on the baffle_gap value. See notes on baffle_gap for further detail		

pad pad_mat pad_inner_radius pad_outer_radius pad_arc pad_azi_locs

pad_mat	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This card defines the material to be used for all neutron pads		
Notes: None		

pad_inner_radius	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: This card defines the inner radius to be used to construct all neutron pads		
Notes: None		

pad_outer_radius	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: This card defines the outer radius to be used to construct all neutron pads		
Notes: None		

pad_arc	Float	Optional
Units: degrees (default)		
Applicable Value(s):		
Limitation(s): None		

continued on next page...

pad_arc, continued...

Description: This card defines the arc length to be used to construct all neutron pads
Notes: None

pad_azi_locs	Array of Floats	Optional
Units: degrees (default)		
Applicable Value(s):		
Limitation(s): None		
Description: This card defines the azimuthal angle location of each pad. These values should correspond to the centerpoint of each arc		
Notes: None		

pad_nonuniform_arc pad_nonuniform_arc

pad_nonuniform_arc	Array of Floats	Optional
Units: degrees (default)		
Applicable Value(s):		
Limitation(s): None		
Description: This card is used to define the arc length for each corresponding neutron pad location defined in the pad card. Therefore, pads can be of different arc lengths. If all pads are the same arc length, this card is not needed, and the single pad_arc value from the pad card will suffice		
Notes: This card requires the pad card to be defined		

vessel vessel_mats vessel_radii

vessel_mats	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Vessel materials		
Notes: Every vessel_mats must have a corresponding vessel_radii		

vessel_radii	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Vessel radii		
Notes: Every vessel_radii must have a corresponding vessel_mats. Example: vessel mod 187.9 ss 193.7 mod 219.1 ss 219.7 cs 241.3		

hole hole_x hole_y hole_radius

hole_x	Float	Optional
Units: cm (default)		

continued on next page...

hole_x, continued...

Applicable Value(s):
Limitation(s): None
Description: This card is used to specify the *x* location of the centerpoint of the hole being defined
Notes: None

hole_y	Float	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: This card is used to specify the *y* location of the centerpoint of the hole being defined		
Notes: None		

hole_radius	Float	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: This card is used to specify the radius of the hole being defined		
Notes: None		

core_shape core_shape

core_shape	2D Integer Map	Required
Units: N/A		
Applicable Value(s): 0 or 1		
Limitation(s): None		
Description: This is a square map showing the fuel assembly locations. Enter 1 for fuel assembly locations and 0 for empty locations		
Notes: None		

assm_map assm_map

assm_map	2D Character String Map	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a core map of the fuel assembly types. The assembly types correspond to assembly labels in the [ASSEMBLY] block. All fuel assemblies must have a type defined		
Notes: None		

inlet_orifice_map inlet_orifice_map

inlet_orifice_map	2D Character String Map	Optional
Units: N/A		

continued on next page...

`inlet_orifice_map`, continued...

Applicable Value(s):
Limitation(s): None
Description: This is a 2D array that must match the shape of `assm_map` in [CORE]. It specifies an ID for the inlet loss in that core location. The actual loss for the ID is later defined on the `inlet_orifice_loss` card. This input is only valid for BWRs
Notes: None

inlet_orifice_loss label form_loss_coefficient / area

`label`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a label for the form loss being defined. It must match a label given in `inlet_orifice_map`		
Notes: All labels in the `inlet_orifice_map` array must be defined in the `inlet_orifice_loss` input		

`form_loss_coefficient`	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0.0		
Limitation(s): None		
Description: Form loss coefficient for the loss		
Notes: None		

`area`	Float	Optional
Units: cm^2 (default)		
Applicable Value(s): 65.416 (default), > 0.0		
Limitation(s): None		
Description: Flow area of the orifice being modeled by this loss. This input shall be identified by keyword as `area=value`		
Notes: This input is optional on this card and will default if not provided		

rotate_map rotate_map

`rotate_map`	2D Integer Map	Optional
Units: N/A		
Applicable Value(s): 0, 1, 2, or 3		
Limitation(s): None		
Description: Core map of assembly rotations		
Notes: None		

insert_rotate_map insert_rotate_map

insert_rotate_map	2D Integer Map	Optional
Units: N/A		
Applicable Value(s): 0, 1, 2, or 3		
Limitation(s): None		
Description: Core map of assembly insert rotations		
Notes: None		

insert_map insert_map

insert_map	2D Character String Map	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a core map of the fuel insert types and locations. The insert types correspond to insert labels in the [INSERT] block. Use a dash to specify assemblies with no inserts		
Notes: None		

det_map det_map

det_map	2D Character String Map	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a core map of the detector types and locations. The detector types correspond to detector labels in the [DETECTOR] block. Use a dash to specify assemblies with no detectors		
Notes: None		

crd_map crd_map

crd_map	2D Character String Map	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a core map of the control rod types and locations. The control rod types correspond to control rod labels in the [CONTROL] block. Use a dash to specify assemblies with no control rods		
Notes: None		

crd_bank crd_bank

crd_bank	2D Character String Map	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a core map of the control rod bank labels. These labels are used to position groups of control rods by bank label. Use a dash to specify assemblies with no control rods		
Notes: None		

nblade blade_size

blade_size	Integer	Required
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Maximum number of blades across one axis in full-core geometry		
Notes: None		

blade_map blade_map

blade_map	2D Character String Map	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a core map of the control blade types and locations. The control blade types correspond to control rod labels in the [CONTROL] block. Use a dash to specify assemblies with no control blades		
Notes: None		

nbwrdet bwrdet_size

bwrdet_size	Integer	Required
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Maximum number of detectors across one axis in full-core geometry		
Notes: None		

bwrdet_map bwrdet_map

bwrdet_map	2D Character String Map	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a core map of the detector types and locations. The detector types correspond to detector labels in the [DETECTOR] block. Use a dash to specify assemblies with no detectors		
Notes: None		

lower_plate lower_mat lower_thick lower_vfrac

lower_mat	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Lower core plate material		

continued on next page...

lower_mat, continued...

Notes: None

lower_thick	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Lower core plate thickness		
Notes: None		

lower_vfrac	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0,≤ 1		
Limitation(s): None		
Description: This is the lower core plate material volume fraction. The remainder of the volume fraction will be filled with moderator for PWRs and coolant for BWRs		
Notes: None		

upper_plate upper_mat upper_thick upper_vfrac

upper_mat	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Upper core plate material		
Notes: None		

upper_thick	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Upper core plate thickness		
Notes: None		

upper_vfrac	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0,≤ 1		
Limitation(s): None		
Description: This is the upper core plate material volume fraction. The remainder of the volume fraction will be filled with moderator for PWRs and coolant for BWRs		
Notes: None		

bc_sym bc_sym

bc_sym	Character String	Optional
Units: N/A		
Applicable Value(s): rot (default), mir		
Limitation(s): None		
Description: This is the symmetry flag for the core when using quarter-symmetry. This flag is not used in full-symmetry		
Notes: None		

bc_bot bc_bot

bc_bot	Character String	Optional
Units: N/A		
Applicable Value(s): vacuum (default), reflecting		
Limitation(s): None		
Description: Bottom neutron transport boundary condition		
Notes: None		

bc_top bc_top

bc_top	Character String	Optional
Units: N/A		
Applicable Value(s): vacuum (default), reflecting		
Limitation(s): None		
Description: Top neutron transport boundary condition		
Notes: None		

bc_rad bc_rad

bc_rad	Character String	Optional
Units: N/A		
Applicable Value(s): vacuum (default), reflecting, periodic		
Limitation(s): None		
Description: Radial neutron transport boundary condition		
Notes: None		

xlabel xlabel

xlabel	Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a list of 2-character assembly position labels in the x-direction. These values are used in the edit maps		
Notes: See Section 4.4 for more information and Section 6.1 for examples		

ylabel ylabel

ylabel	Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a list of 2-character assembly position labels in the y-direction. These values are used in the edit maps		
Notes: See Section 4.4 for more information and Section 6.1 for examples		

label_format label_format

label_format	Character String	Optional
Units: N/A		
Applicable Value(s): x-y (default), y-x, .x-y, .y-x		
Limitation(s): None		
Description: This is the format of label entries in shuffle_label card		
Notes: None		

height height

height	Float	Required
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: This is the total axial distance from bottom core plate to upper core plate. Distance does not include core plate thicknesses		
Notes: None		

mat mat

mat	Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Refer to the detailed materials description given in Chapter 3		
Notes: None		

lower_ref lower_refl_mats lower_refl_thicks lower_refl_vfracs

lower_refl_mats	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Lower reflector materials		
Notes: None		

lower_refl_thicks	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Lower reflector thicknesses		
Notes: None		

lower_refl_vfracs	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0,≤ 1		
Limitation(s): None		
Description: Lower reflector volume fractions. If less than one, the remainder of the volume fraction will be filled with moderator for PWRs and coolant for BWRs		
Notes: None		

upper_ref upper_refl_mats upper_refl_thicks upper_refl_vfracs

upper_refl_mats	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Upper reflector materials		
Notes: None		

upper_refl_thicks	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Upper reflector thicknesses		
Notes: None		

upper_refl_vfracs	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0,≤ 1		
Limitation(s): None		
Description: Upper reflector volume fractions. If less than one, the remainder of the volume fraction will be filled with moderator for PWRs and coolant for BWRs		
Notes: None		

reactor_type reactor_type

reactor_type	Character String	Optional
Units: N/A		

continued on next page...

`reactor_type`, continued...

Applicable Value(s): PWR (default), BWR
Limitation(s): None
Description: Model reactor type
Notes: None

source mat_id iso_id iso_scal / spectrum(:) / stt_str str_mult

`mat_id`	Integer	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is an integer ID corresponding to the material of external source		
Notes: None		

`iso_id`	Integer	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is an integer value representing an isotope on whose absolute atom quantity the source strength will be scaled. The input should follow the format ZZAAA. Omitting this value indicates that no isotope will be used, and the user will provide an absolute strength flux spectrum		
Notes: None		

`iso_scale`	Float	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a positive real corresponding to the scaling factor to use when scaling the fractional flux spectrum to its absolute strength. This scaling is in terms of the number of atoms of the scaling isotope which appears in a given FSR. Units are in neutrons per second per unit volume (cc) per number of isotope atoms. This value is only required if the user provides `iso_id(i)`		
Notes: None		

`spectrum`	Float	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This contains positive real values corresponding to the either the fractional or absolute source spectrum. If scaling isotope information is provided, this represents a fractional spectrum; otherwise, it represents an absolute spectrum in units of neutrons per second per unit of volume (cc). The number of values must match the number of energy groups of the problem. Values cannot be negative and must sum to nearly 1.0 if the input corresponds to a fractional spectrum		

continued on next page...

`spectrum`, continued...

Notes: None

`stt_str`	Float	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a real number greater than 0.0 and less than or equal to 1.0 corresponding to the fractional starting strength of the source. This is how much of the source will be applied during the first external source iteration. If this value is not provided, it will default to full strength		
Notes: None		

`str_mult`	Float	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a real number greater than 1.0 corresponding to the multiplicative increase of the source strength. If stt_str is provided, then this value must be provided		
Notes: None		

steam_generator sg_type sg_alloy sg_area sg_plug_frac

`sg_type`	Character String	Optional
Units: none (default)		
Applicable Value(s): oncethrough, utube		
Limitation(s): None		
Description: Steam generator type		
Notes: Used in the chemistry source term calculation to calculate coolant temperatures		

`sg_alloy`	Integer	Optional
Units: none (default)		
Applicable Value(s): 600, 690, 800, 304		
Limitation(s): None		
Description: Steam generator tubing stainless steel alloy number		
Notes: This is used in the chemistry source term calculation to determine surface material properties		

`sg_area`	Float	Optional
Units: m^2 (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Total surface area of steam generator tubing		
Notes: This is used in the chemistry source term calculation to determine amount of source term created		

sg_plug_frac	Float	Optional
Units: none (default)		
Applicable Value(s): ≥ 0,≤ 1		
Limitation(s): None		
Description: Steam generator plugged area fraction		
Notes: The effective area of the steam generator used in the chemistry source term calculation is $sg\_area*(1-plug\_frac)$		

hot_leg_piping hot_leg_alloy hot_leg_area

hot_leg_alloy	Integer	Optional
Units: none (default)		
Applicable Value(s): 600, 690, 800, 304		
Limitation(s): None		
Description: Hot leg piping stainless steel alloy number		
Notes: This is used in the chemistry source term calculation to determine surface material properties		

hot_leg_area	Float	Optional
Units: m^2 (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Total surface area of hot leg piping		
Notes: This is used in the chemistry source term calculation to determine amount of source term created		

cold_leg_piping cold_leg_alloy cold_leg_area

cold_leg_alloy	Integer	Optional
Units: none (default)		
Applicable Value(s): 600, 690, 800, 304		
Limitation(s): None		
Description: Cold leg piping stainless steel alloy number		
Notes: This is used in the chemistry source term calculation to determine surface material properties		

cold_leg_area	Float	Optional
Units: m^2 (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Total surface area of cold leg piping		
Notes: This is used in the chemistry source term calculation to determine amount of source term created		

cleanup_rated_flow cleanup_rated_flow

cleanup_rated_flow	Float	Optional

continued on next page...

`cleanup_rated_flow`, continued...

Units: kg/s (default)
Applicable Value(s): ≥ 0
Limitation(s): None
Description: Rated flow rate of coolant chemistry cleanup system
Notes: None

material_perturbation_file material_perturbation_file

`material_perturbation_file`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Name of the file to read in that contains material perturbation information in h5 format		
Notes: None		

bioshield bioshield

`bioshield`	Strings and doubles	Optional
Units: N/A		
Applicable Value(s): None (default)		
Limitation(s): None		
Description: Bioshield materials and radii beyond the vessel used to automatically generate an Omnibus excore input		
Notes: Materials must be defined in the Omnibus template input		

det det

`det`	Strings and doubles	Optional
Units: N/A		
Applicable Value(s): None (default)		
Limitation(s): None		
Description: These are the defined detector types for automatic generation of an Omnibus excore input—requires `bioshield` card		
Notes: Materials must be defined in the Omnibus template input		

det_locations det_locations

`det_locations`	Strings and doubles	Optional
Units: N/A		
Applicable Value(s): None (default)		
Limitation(s): None		
Description: These are the defined detector locations for automatic generation of an Omnibus excore input—requires `bioshield` and `det` cards		
Notes: Materials must be defined in the Omnibus template input		

5.4 BLOCK ASSEMBLY

title title

`title`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Long descriptive title for assembly		
Notes: None		

npin npin

`npin`	Integer	Required
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of rods along the edge of an assembly		
Notes: None		

ppitch ppitch

`ppitch`	Float	Required
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Pincell pitch		
Notes: None		

cell cell

`cell`	Character Strings and Floats	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Refer to the cell description given in Section 2.3.2		
Notes: None		

lattice lattice

`lattice`	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is an obsolete alias for `rodmap`. Use `rodmap` instead		
Notes: None		

rodmap axial_label cell_map

`axial_label`	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Label for this axial elevation description		
Notes: See Section 2.3.3 for examples		

`cell_map`	2D Character String Map	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the lattice map for this axial elevation. Use a dash for an empty location		
Notes: See Section 2.3.3 for examples		

axial Label axial_labels axial_elevations

`Label`	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the label for this assembly. The label corresponds to `assm_map` in [CORE] block		
Notes: See Section 2.3.4 for examples		

`axial_labels`	Character Strings	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a list of axial labels for this assembly description, corresponding to labels in lattice maps		
Notes: See Section 2.3.4 for examples		

`axial_elevations`	Float	Required
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: List of axial elevations for this assembly description		
Notes: See Section 2.3.4 for examples		

rpdlm label rpdlm_exposure rpdlm_power

`label`	Character String	Optional

continued on next page...

`label`, continued...

Units: N/A
Applicable Value(s):
Limitation(s): None
Description: Label for this assembly. Label corresponds to `assm_map` in [CORE] block
Notes: None

`rpdlm_exposure`	Float	Optional
Units: GWDMT (default)		
Applicable Value(s): ≥ 0		
Limitation(s): Values must be listed in increasing order		
Description: Assembly averaged exposure values that form the intervals of the piecewise rpdlm function. The first exposure value is implied to be zero. Consecutive exposure points form an exposure interval over which the provided linear heat rate limits are interpolated. The last exposure value is taken to be infinity		
Notes: None		

`rpdlm_power`	Float	Optional
Units: kW/ft (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Linear heat rate limit values that will be used to form the piecewise rpdlm function. The first rpdlm segment is assumed to be constant at the first provided heat rate over the exposure interval from 0.0 GWd/MT to the first exposure value. Subsequent limits are defined by the line segment between consecutive exposure/power pairs. The final segment is constant at the last specified heat rate from the last specified exposure through infinite exposure		
Notes: None		

grid label material height mass / loss lossmap blockage gridmap yhl1 yhl2 area

`label`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Grid label for a single grid type		
Notes: See Section 2.3.5 for examples		

`material`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Grid material for this grid type		
Notes: See Section 2.3.5 for examples		

height	Float	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Grid height for this grid type		
Notes: See Section 2.3.5 for examples		

mass	Float	Optional
Units: g (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Grid mass for this grid type		
Notes: See Section 2.3.5 for examples		

loss	Float	Optional
Units: N/A		
Applicable Value(s): 0.0 (default), ≥ 0.0		
Limitation(s): None		
Description: Loss coefficient for the spacer grid. Do not provide `lossmap` input if providing this input. This input shall be identified by keyword as `loss=value`		
Notes: This input is not required on this card		

lossmap	String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Provides the name of a loss coefficient map, to be entered elsewhere in the [ASSEMBLY] block. This name must match a `lossmap_name` entered in a `lossmap` entry of the input. This optional input shall be identified by keyword as `lossmap=lossmap_name`		
Notes: This input is not required on this card		

blockage	Float	Optional
Units: N/A		
Applicable Value(s): 0.0 (default), ≥ 0.0 and <1.0		
Limitation(s): None		
Description: Provides the blockge ratio of this spacer grid. A value of zero means the spacer grid does not block the flow area and a value of 0.1 means it blocks 10% of the flow area (for example). Do not provide this input if providing the `gridmap` optional input. This input shall be identified by keyword as `blockage=value`		
Notes: This input is not required on this card		

gridmap	String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Provides the name of the ROTHCON dataset to use for modeling grid heat transfer and turbulence enhancement effects. Do not provide a `blockage` input for this grid if specifying this input. The specified name must exist in a ROTHCON data file that is in the directory where the simulation will be run. This input shall be identified by keyword as `gridmap=value`		
Notes: This input is not required on this card		

yhl1	Float	Optional
Units: N/A		
Applicable Value(s): 5.55 (default), ≥0.0		
Limitation(s): None		
Description: This is a coefficient in the Yao-Hochreiter-Leech grid spacer heat transfer enhancement model. The model has the form, $M = \left(1 + yhl_1 \epsilon^2 \exp\left[yhl_2 \frac{z}{D}\right]\right)$, where M is the heat transfer coefficient multiplier, z is the downstream distance from the grid location, and D is the hydraulic diameter. The ϵ (blockage ratio), yhl_1, and yhl_2 terms are all user inputs that may be set for this spacer grid. This input provides the yhl_1 value. All parameters will default if not set by the user. This input shall be identified by keyword as `yhl1=value`		
Notes: This input is not required on this card. Do not enter this input if using a *gridmap*		

yhl2	Float	Optional
Units: N/A		
Applicable Value(s): -0.13 (default)		
Limitation(s): None		
Description: This is a coefficient in the Yao-Hochreiter-Leech grid spacer heat transfer enhancement model. The model has the form, $M = \left(1 + yhl_1 \epsilon^2 \exp\left[yhl_2 \frac{z}{D}\right]\right)$, where M is the heat transfer coefficient multiplier, z is the downstream distance from the grid location, and D is the hydraulic diameter. The ϵ (blockage ratio), yhl_1, and yhl_2 terms are all user inputs that may be set for this spacer grid. This input provides the yhl_2 value. All parameters will default if not set by the user. This input shall be identified by keyword as `yhl2=value`		
Notes: This input is not required on this card. Do not enter this input if using a *gridmap*		

area	Float	Optional
Units: cm^2 (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Provides the reference area for the spacer grid loss coefficient. If this is a BWR model and this input is not provided, then this will have a default of 64.516 cm^2 (10 in^2). If this is a PWR model, this input has no effect. This reference area is used to adjust the form loss coefficient for a reference area. If the form loss coefficient provided is the actual form loss coefficient, then the true flow area of the bundle should be provided		
Notes: None		

lossmap lossmap_name loss_coeff_map

`lossmap_name`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The name for this map of loss coefficients		
Notes: None		

`loss_coeff_map`	2D Float Map	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: A 2D map of loss coefficients. The shape of the map shall be identical to the rod lattice shape for the assembly which the loss map is being defined. Each loss coefficient will be applied to the flow adjacent to the rod for which it is defined		
Notes: None		

grid_axial grid_map grid_elev

`grid_map`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a list of spacer grid labels for all grids in an assembly. All labels must correspond to `grid` card		
Notes: See Section 2.3.5 for examples		

`grid_elev`	Float	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: This is a list of spacer grid elevations for all grids in an assembly. Elevations refer to the grid midpoint		
Notes: See Section 2.3.5 for examples		

lower_nozzle lower_nozzle_comp lower_nozzle_height lower_nozzle_mass / loss area

`lower_nozzle_comp`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Lower nozzle material		
Notes: None		

lower_nozzle_height	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Lower nozzle height		
Notes: None		

lower_nozzle_mass	Float	Optional
Units: g (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: This is the lower nozzle mass. The code will calculate the volume of the nozzle given the nozzle mass and will fill the remaining volume with either moderator for PWRs or coolant for BWRs		
Notes: None		

loss	Float	Optional
Units: N/A		
Applicable Value(s): 0.0 (default), ≥ 0.0		
Limitation(s): None		
Description: Loss coefficient associated with the lower nozzle. The input shall be identified by keyword as loss=value		
Notes: This input is optional on this card and will default if not provided		

area	Float	Optional
Units: cm^2 (default)		
Applicable Value(s): 64.516 (default), > 0.0		
Limitation(s): None		
Description: Area associated with the lower nozzle. This input shall be identified by keyword as area=value		
Notes: This input is optional on this card and will default if not provided		

upper_nozzle upper_nozzle_comp upper_nozzle_height upper_nozzle_mass / loss area

upper_nozzle_comp	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Upper nozzle material		
Notes: None		

upper_nozzle_height	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		

continued on next page...

`upper_nozzle_height`, continued...

Limitation(s): None
Description: Upper nozzle height
Notes: None

`upper_nozzle_mass`	Float	Optional
Units: g (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: This is the upper nozzle mass. The code will calculate the volume of the nozzle given the nozzle mass and will fill the remaining volume with either moderator for PWRs or coolant for BWRs		
Notes: None		

`loss`	Float	Optional
Units: N/A		
Applicable Value(s): 0.0 (default), ≥ 0.0		
Limitation(s): None		
Description: Loss coefficient associated with the upper nozzle. The input shall be identified by keyword as `loss=value`		
Notes: This input is optional on this card and will default if not provided		

`area`	Float	Optional
Units: cm^2 (default)		
Applicable Value(s): 64.516 (default), > 0.0		
Limitation(s): None		
Description: Area associated with the upper nozzle. The input shall be identified by keyword as `area=value`		
Notes: This input is optional on this card and will default if not provided		

fuel fuel

`fuel`	Character String and Floats	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Refer to the detailed fuel materials description given in Section 3.3		
Notes: None		

mat mat

`mat`	Character Strings and Floats	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		

continued on next page...

mat, continued...

Description: Refer to the detailed materials description given in Section 3.1
Notes: None

gap gapw gapn

gapw	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Wide-gap width		
Notes: BWR only		

gapn	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Narrow-gap width		
Notes: BWR only		

channel_box chanmat cornerth chanrad

chanmat	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Channel box material		
Notes: None		

cornerth	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Channel box corner thickness		
Notes: None		

chanrad	Float	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Channel box inside corner radius		
Notes: None		

channel_box_segments chanth chanlen chanramp

chanth	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Thickness of a channel box segment		
Notes: None		

chanlen	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Length of channel box segment		
Notes: None		

chanramp	Float	Optional
Units: N/A, cm		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Length of channel box segment ramp		
Notes: None		

temptable temptable

temptable	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): Requires a temperature table file to be included below		
Description: This flag defines a temperature table in the assembly block that can be used in the cell definitions; each cell can have a separate table if desired		
Notes: Tables as generated through the BISON temperature table process, which define `temptable_boundary`, `temptable_qprime`, and `temptable_polynomial`, can be included after the tag is specified. See the example below for usage with specification in the cell flag: temptable U26 include u26.tab temptable GAD include ug3.tab cell 2 0.4096 0.418 0.475 / U26 he zirc4 / U26 cell 3 0.4096 0.418 0.475 / UG3 he zirc4 / GAD		

5.5 BLOCK CONTROL

title title

`title`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Long descriptive title for control rod description		
Notes: None		

npin num_pins

`num_pins`	Integer	Required
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of rods along the edge of an assembly		
Notes: None		

stroke stroke maxstep

`stroke`	Float	Required
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Control rod stroke—distance between full insertion and full withdrawal		
Notes: See Section 2.4.1 for examples		

`maxstep`	Float	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Total number of steps between full insertion and full withdrawal		
Notes: See Section 2.4.1 for examples		

cell cell

`cell`	Character Strings and Floats	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Refer to the cell description given in Section 2.4		
Notes: None		

lattice lattice

`lattice`	Character String	Required
Units: N/A		

continued on next page...

`lattice`, continued...

Applicable Value(s):
Limitation(s): None
Description: This is an obsolete alias for `rodmap`. Use `rodmap` instead
Notes: None

rodmap label cell_map

`label`	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Label for this axial elevation description		
Notes: See Section 2.4 for examples		

`cell_map`	2D Character String Map	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a lattice map for this axial elevation. Use a dash for no control rod		
Notes: See Section 2.4 for examples		

axial control_label axial_labels axial_elevations

`control_label`	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the label for this control rod description—corresponds to `crd_map` in [CORE] block		
Notes: See Section 2.4 for examples		

`axial_labels`	Character Strings	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: List of axial labels for this control rod description—corresponds to labels in rod maps		
Notes: See Section 2.4 for examples		

`axial_elevations`	Float	Required
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: List of axial elevations for this control rod description		

continued on next page...

`axial_elevations`, continued...

Notes: See Section 2.4 for examples

mat mat

`mat`	Character Strings and Floats	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Refer to the detailed materials description given in Section 3.1		
Notes: None		

blade ntube tubecell bladespan bladeth bladerad bladesheath bladewing blademat

`ntube`	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Number of rodlets in control blade wing		
Notes: None		

`tubecell`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Cell ID for rodlet		
Notes: None		

`bladespan`	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Control blade span from center to wing tip		
Notes: None		

`bladeth`	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Control blade wing thickness		
Notes: None		

bladerad	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Radius of control blade tip		
Notes: None		

bladesheath	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Control blade sheath thickness		
Notes: None		

bladewing	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Blade central structure wing length		
Notes: None		

blademat	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Sheath and wing material		
Notes: None		

bladegeom bladegeomlabel span thickness tipradius sheaththickness winglength

bladegeomlabel	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Unique label name for each blade geometry specification		
Notes: None		

span	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Control blade span from center to wing tip		

continued on next page...

span, continued...

Notes: None

thickness	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Control blade wing thickness		
Notes: None		

tipradius	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Radius of control blade tip		
Notes: None		

sheaththickness	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Control blade sheath thickness		
Notes: None		

winglength	Float	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Blade central structure wing length		
Notes: None		

bladetype bladetypelabel bladegeomlabel blademat tubeorient ntube tubecelllist

bladetypelabel	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Unique label name for each blade geometry, material, and absorber loading		
Notes: None		

bladegeomlabel	Character String	Optional
Units: N/A		

continued on next page...

bladegeomlabel, continued...

Applicable Value(s):
Limitation(s): None
Description: Unique label name for each blade geometry specification
Notes: None

blademat	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Sheath and wing material		
Notes: None		

tubeorient	Character String	Optional
Units: N/A		
Applicable Value(s): vert (default), hor		
Limitation(s): None		
Description: This specifies whether the tubes are inserted vertically or horizontally in the control blade. If the option is horizontal, MPACT will internally rotate and adjust them to model them vertically		
Notes: None		

ntube	Integer	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Number of rodlets in control blade wing		
Notes: None		

tubecelllist	Character String Array	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: A spatially dependent list of cell (rodlet) labels. The list of cells are added from left to right from the center of the control blade to the tip. The number of labels in the list must match the value of ntube		
Notes: None		

5.6 BLOCK INSERT

title title

title	Character String	Optional
Units: N/A		

continued on next page...

`title`, continued...

Applicable Value(s):
Limitation(s): None
Description: Long descriptive title for assembly insert description
Notes: None

npin num_pins

`num_pins`	Integer	Required
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of rods along the edge of an assembly		
Notes: None		

cell cell

`cell`	Character Strings and Floats	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Refer to the cell description given in Section 2.5		
Notes: None		

lattice lattice

`lattice`	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is an obsolete alias for `rodmap`. Use `rodmap` instead		
Notes: None		

rodmap label cell_map

`label`	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Label for this axial elevation description		
Notes: See Section 2.5 for examples		

`cell_map`	2D Character String Map	Required
Units: N/A		
Applicable Value(s):		

continued on next page...

cell_map, continued...

Limitation(s): None
Description: This is the lattice map for this axial elevation. Use a dash for no insert rod
Notes: See Section 2.5 for examples

axial control_label axial_labels axial_elevations

control_label	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the label for this assembly insert description. It corresponds to insert_map in [CORE] block		
Notes: See Section 2.5 for examples		

axial_labels	Character Strings	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a list of axial labels for this assembly insert description. It corresponds to labels in rod maps		
Notes: See Section 2.5 for examples		

axial_elevations	Float	Required
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: List of axial elevations for this assembly insert description		
Notes: See Section 2.5 for examples		

mat mat

mat	Character Strings and Floats	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Refer to the detailed materials description given in Section 3.1		
Notes: None		

5.7 BLOCK DETECTOR

title title

title	Character String	Optional
Units: N/A		

continued on next page...

`title`, continued...

Applicable Value(s):
Limitation(s): None
Description: Long descriptive title for detector
Notes: None

type detector_type

`detector_type`	Character String	Optional
Units: N/A		
Applicable Value(s): u235, v, rh, gamma, gamma_approx		
Limitation(s): None		
Description: This is a flag used to specify the type of detector to be modeled. `u235` is the ^{235}U fission reaction rate, `v` is the absorption reaction rate in vanadium, `rh` is the absorption reaction rate in rhodium, `gamma` is the total gamma reaction rate in titanium, and `gamma_approx` is an approximate gamma response using the fastest neutron flux		
Notes: None		

npin num_pins

`num_pins`	Integer	Required
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of rods along the edge of an assembly		
Notes: None		

cell cell

`cell`	Character Strings and Floats	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Refer to the cell description given in Section 2.6		
Notes: None		

lattice lattice

`lattice`	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is an obsolete alias for `rodmap`. Use `rodmap` instead		
Notes: None		

rodmap label cell_map

label	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Label for this axial elevation description		
Notes: See Section 2.6 for examples		

cell_map	2D Character String Map	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the lattice map for this axial elevation. Use a dash for no detector rod		
Notes: See Section 2.6 for examples		

axial control_label axial_labels axial_elevations

control_label	Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the label for this detector description. It corresponds to det_map in [CORE] block		
Notes: See Section 2.6 for examples		

axial_labels	Character Strings	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the list of axial labels for this detector description. It corresponds to labels in rod maps		
Notes: See Section 2.6 for examples		

axial_elevations	Float	Required
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: List of axial elevations for this detector description		
Notes: See Section 2.6 for examples		

mat mat

mat	Character Strings and Floats	Optional
Units: N/A		
Applicable Value(s):		

continued on next page...

mat, continued...

Limitation(s): None
Description: Refer to the detailed materials description given in Section 3.1
Notes: None

5.8 BLOCK EDITS

axial_edit_bounds axial_edit_bounds

axial_edit_bounds	Float	Required
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: The boundaries of the axial regions over which axial information should be printed		
Notes: See Section 2.8 for examples		

axial_edit_mesh_delta axial_edit_mesh_delta

axial_edit_mesh_delta	Float	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Produces a uniform axial output grid (integrates pin powers over a uniform axial mesh)		
Notes: None		

points points_type points_dim1 points_dim2 points_dim3

points_type	Character String	Optional
Units: N/A		
Applicable Value(s): CART,RTHETA		
Limitation(s): None		
Description: Type of coordinate system to be used to define point edits		
Notes: None		

points_dim1	Float	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: This is the first dimension in point edit. If points_type is CART, then dim1 represents X. If points_type is RTHETA, then dim1 represents R		
Notes: None		

points_dim2	Float	Optional
Units: cm(CART), degrees(RTHETA) (default)		

continued on next page...

points_dim2, continued...

Applicable Value(s):
Limitation(s): None
Description: This is the second dimension in point edit. If points_type is CART, then dim2 represents Y. If points_type is RTHETA, then dim2 represents Theta
Notes: None

points_dim3	Float	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: This is the third dimension in point edit. If points_type is CART, then dim3 represents Z. If points_type is RTHETA, then dim3 represents Z		
Notes: None		

edit_group edit_group

edit_group	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is a list of edits that can be turned on or off as a group using the edit card in the [STATE] block		
Notes: There are known issues with naming the same edit in multiple groups. It is best to name each edit in at most one edit_group input to prevent unpredictable behavior		

edit_scrape edit_scrape

edit_scrape	Table of string, doubles and ints. Each row in the table has length 8	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This card specifies an area on the specified rod surface over which a crud scrape is generated. The scrape location is specified as follows: • <scrape_id>: String. Unique scrape identifier • <asm_col_row>: String. Assembly label. Dashed delimited ex: 'H-2' • <pin_row>: Int. CTF pin row in assembly • <pin_col>: Int. CTF pin column in assembly • <min_th>: Float. Minimum azimuthal scrape angle in degrees, 0 degrees points due east • <max_th>: Float. Maximum azimuthal scrape angle in degrees • <min_z>: Float. Minimum axial scrape location in cm • <max_z>: Float. Maximum axial scrape location in cm		
Notes: This is only needed for specifying crud scrape locations		

detector_mesh detector_mesh_type detector_mesh

`detector_mesh_type`	Character String	Optional
Units: N/A		
Applicable Value(s): pointwise, integral		
Limitation(s): None		
Description: Defines which detector responses should be edited. `pointwise` will edit the detector response at the meshes provided, whereas `integral` will provide the integral detector response between meshes		
Notes: None		

`detector_mesh`	Array of Floats	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): Must be listed in ascending order		
Description: Axial heights used to define detector response		
Notes: None		

5.9 BLOCK SHIFT

num_threads num_threads

`num_threads`	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: Number of threads per processor		
Notes: Applicable to threaded machines		

seed seed

`seed`	Integer	Optional
Units: N/A		
Applicable Value(s): 121434 (default), > 0		
Limitation(s): None		
Description: Initial seed for random number generator (global)		
Notes: None		

ce_lib_path ce_lib_path

`ce_lib_path`	String	Optional
Units: N/A		
Applicable Value(s): ce_v7.1_endf.h5 (default)		
Limitation(s): None		
Description: Path to SCALE CE data library file		
Notes: None		

transfer transfer

transfer	String	Optional
Units: N/A		
Applicable Value(s): Depends on coupling (default), all, fiss_src, isotopics, temps		
Limitation(s): None		
Description: What to transfer with VERA		
Notes: None		

temp_transfer temp_transfer

temp_transfer	String	Optional
Units: N/A		
Applicable Value(s): all (default), all, none, pin		
Limitation(s): None		
Description: Which temperatures to couple with CTF		
Notes: None		

verbosity verbosity

verbosity	string	Optional
Units: N/A		
Applicable Value(s): none (default), none, low, medium, high		
Limitation(s): None		
Description: How often to print about particles being transported		
Notes: None		

balance_tol balance_tol

balance_tol	double	Optional
Units: N/A		
Applicable Value(s): 0.5 (default), (0,1)		
Limitation(s): None		
Description: Tolerance for checking balance of CE cross sections		
Notes: None		

n_energy_min n_energy_min

n_energy_min	double	Optional
Units: eV (default)		
Applicable Value(s): 0.00001 (default), > 0		
Limitation(s): None		
Description: Minimum neutron energy for transport		
Notes: None		

n_energy_max n_energy_max

n_energy_max	double	Optional
Units: eV (default)		
Applicable Value(s): 20.0e6 (default), > 0		
Limitation(s): None		
Description: Maximum neutron energy for transport		
Notes: None		

broaden_xs broaden_xs

broaden_xs	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Cross section Doppler broadening for temperature		
Notes: None		

temperature_tol temperature_tol

temperature_tol	double	Optional
Units: K (default)		
Applicable Value(s): 4.0 (default), > 0		
Limitation(s): None		
Description: Tolerance for reusing existing broadened cross sections		
Notes: None		

union_energy union_energy

union_energy	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Unionize lower and upper library temperature energy grids		
Notes: None		

delta_t delta_t

delta_t	double	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), > 0		
Limitation(s): None		
Description: Finite difference grid spacing for Leal-Hwang temperature interpolation of cross sections		
Notes: None		

energy_tol energy_tol

energy_tol	double	Optional
Units: N/A		
Applicable Value(s): 1.0E-10 (default), (0,1)		
Limitation(s): None		
Description: Relative difference for considering two energy points equal		
Notes: None		

kinematics kinematics

kinematics	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Apply broadening to collision data		
Notes: None		

dbrc dbrc

dbrc	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Apply doppler broadening resonance correction		
Notes: None		

global_log global_log

global_log	string	Optional
Units: N/A		
Applicable Value(s): info (default), debug, diagnostic, status, info, warning, error, critical		
Limitation(s): None		
Description: Level of global log information		
Notes: None		

local_log local_log

local_log	string	Optional
Units: N/A		
Applicable Value(s): error (default), debug, diagnostic, status, info, warning, error, critical		
Limitation(s): None		
Description: Level of local node log information		
Notes: None		

do_debug_history_tally do_debug_history_tally

do_debug_history_tally	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Output a particle history diagnostic tally if error occurs		
Notes: None		

log_memory log_memory

log_memory	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Output debug memory usage to stderr		
Notes: None		

do_micro_tally do_micro_tally

do_micro_tally	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Tally micro reactions in eigenvalue mode		
Notes: Eigenvalue mode only		

do_transport do_transport

do_transport	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Perform Monte Carlo transport		
Notes: None		

do_output do_output

do_output	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Do Shift output		
Notes: None		

micro_zaids micro_zaids

micro_zaids	Array of integers	Optional
Units: N/A		
Applicable Value(s): 92235, 92238 (default)		
Limitation(s): None		
Description: Nuclides to tally micro reactions in eigenvalue mode		
Notes: Eigenvalue mode only		

micro_rxns micro_rxns

micro_rxns	Array of integers	Optional
Units: N/A		
Applicable Value(s): 18, 102 (default)		
Limitation(s): None		
Description: MT of micro reactions to tally in eigenvalue mode		
Notes: Eigenvalue mode only		

gamma_flux gamma_flux

gamma_flux	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Tally the photon flux in each pincell		
Notes: None		

lost_particle_error_tol lost_particle_error_tol

lost_particle_error_tol	double	Optional
Units: N/A		
Applicable Value(s): 1E-06 (default), > 0		
Limitation(s): None		
Description: Fraction of lost particles to tolerate before aborting		
Notes: None		

num_cycles num_cycles

num_cycles	Integer	Optional
Units: N/A		
Applicable Value(s): 50 (default), > 0		
Limitation(s): None		
Description: Number of eigenvalue cycles		
Notes: None		

num_inactive_cycles num_inactive_cycles

num_inactive_cycles	Integer	Optional
Units: N/A		
Applicable Value(s): 10 (default), > 0		
Limitation(s): None		
Description: Number of inactive eigenvalue cycles		
Notes: None		

Np Np

Np	double	Optional
Units: N/A		
Applicable Value(s): 1000 (default), > 0		
Limitation(s): None		
Description: Number of particles to transport		
Notes: None		

transport transport

transport	string	Optional
Units: N/A		
Applicable Value(s): ce (default), ce, mg		
Limitation(s): None		
Description: Type of physics		
Notes: None		

problem_mode problem_mode

problem_mode	string	Optional
Units: N/A		
Applicable Value(s): eigenvalue (default), cadis, eigenvalue, forward, fwcadis		
Limitation(s): None		
Description: Run mode		
Notes: None		

problem_name problem_name

problem_name	string	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Base prefix name for all Shift-produced output files		
Notes: None		

mode mode

mode	string	Optional
Units: N/A		
Applicable Value(s): n (eigenvalue), np (forward) (default), n, np		
Limitation(s): None		
Description: Type of particles to transport		
Notes: None		

output_geometry output_geometry

output_geometry	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Output HDF5 files of ray-traced geometry (initial) and compositions (each state)		
Notes: None		

output_fission_source output_fission_source

output_fission_source	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Output the initial fission source for each state		
Notes: None		

output_external_source output_external_source

output_external_source	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Output the external source for each state		
Notes: None		

output_micro_tally output_micro_tally

output_micro_tally	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Output micro reaction tallies		
Notes: None		

output_ww output_ww

output_ww	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Output the weight windows		
Notes: None		

thermal_energy_cutoff thermal_energy_cutoff

thermal_energy_cutoff	double	Optional
Units: eV (default)		
Applicable Value(s): 10.0 (default), > 0		
Limitation(s): None		
Description: Cutoff for treatment of thermal neutrons		
Notes: None		

excore_filename excore_filename

excore_filename	string	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Name of Omnibus XML file with excore features and tallies		
Notes: None		

raytrace_levels raytrace_levels

raytrace_levels	array of doubles	Optional
Units: N/A		
Applicable Value(s): midpoint of active fuel (default)		
Limitation(s): None		
Description: Z levels to raytrace geometry and output		
Notes: None		

raytrace_resolution raytrace_resolution

raytrace_resolution	Integer	Optional
Units: N/A		
Applicable Value(s): 1024 (default), > 0		
Limitation(s): None		
Description: Resolution for geometry raytrace		
Notes: None		

vera_pressure_vessel vera_pressure_vessel

vera_pressure_vessel	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Pull in the pressure vessel from the VERA geometry		
Notes: Applicable to excore only		

fiss_src_spectrum fiss_src_spectrum

fiss_src_spectrum	String	Optional
Units: N/A		
Applicable Value(s): nuclide_watt (default), u235_watt,mpact,nuclide_watt		
Limitation(s): None		
Description: The type of fission source spectrum to use		
Notes: None		

rtk_output_format rtk_output_format

rtk_output_format	String	Optional
Units: N/A		
Applicable Value(s): hdf5 (default), hdf5,xml		
Limitation(s): None		
Description: The type of file format for the core dumped geometry description		
Notes: Applicable to eigenvalue and CADIS modes with output_geometry on		

use_pole_data use_pole_data

use_pole_data	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Use the pole data for on-the-fly Doppler broadening		
Notes: None		

use_reduced_xs use_reduced_xs

use_reduced_xs	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): Only applies to CADIS and FWCADIS		
Description: Use reduced number of multigroup cross sections for adjoint		
Notes: None		

use_fission_source use_fission_source

use_fission_source	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Use the fission source provided by MPACT		
Notes: None		

use_external_source use_external_source

use_external_source	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Use the external source provided by MPACT		
Notes: None		

hybrid_tally_names hybrid_tally_names

hybrid_tally_names	array of strings	Required if problem mode is CADIS
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Excore tally name to optimize for CADIS		
Notes: Applicable to hybrid simulations		

hybrid_multiplier_names hybrid_multiplier_names

hybrid_multiplier_names	array of strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Excore tally multipliers to optimize for CADIS		
Notes: Applicable to hybrid simulations		

src_disc_l2_error src_disc_l2_error

src_disc_l2_error	double	Optional
Units: N/A		
Applicable Value(s): 0.01 (default), (0,1)		
Limitation(s): None		
Description: Maximum L2 error for point-sampling discretization		
Notes: Applicable to hybrid simulations		

src_disc_samples_per_batch src_disc_samples_per_batch

src_disc_samples_per_batch	Integer	Optional
Units: N/A		
Applicable Value(s): 1E05 (default), > 0		
Limitation(s): None		
Description: Number of samples per point-sampling batch		
Notes: Applicable to hybrid simulations		

src_disc_max_samples src_disc_max_samples

src_disc_max_samples	Integer	Optional
Units: N/A		
Applicable Value(s): 1E10 (default), > 0		
Limitation(s): None		
Description: Maximum number of discretization samples		
Notes: Applicable to hybrid simulations		

ww_decomp ww_decomp

ww_decomp	string	Optional
Units: N/A		
Applicable Value(s): separable (default), full		
Limitation(s): None		
Description: Whether the weight window adjoint flux should be decomposed		
Notes: Applicable to hybrid simulations		

radial_mesh radial_mesh

radial_mesh	array of doubles	Optional
Units: N/A		
Applicable Value(s): vessel radii (default)		
Limitation(s): None		
Description: Radii for flux tally		
Notes: None		

num_theta num_theta

num_theta	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: Number of theta divisions for flux tallies in $[0, 2\pi]$		
Notes: None		

num_axial num_axial

num_axial	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: Number of axial levels for flux tallies		
Notes: None		

n_bounds n_bounds

n_bounds	array of decreasing doubles	Optional
Units: eV (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Neutron energy bounds for tallies		
Notes: None		

p_bounds p_bounds

p_bounds	array of decreasing doubles	Optional
Units: eV (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Photon energy bounds for tallies		
Notes: None		

homog_type homog_type

homog_type	string	Optional
Units: N/A		
Applicable Value(s): assem, rings		
Limitation(s): None		
Description: If using homogenization, homogenize each assembly or in rings		
Notes: Experimental capability		

homog_ring_radii homog_ring_radii

homog_ring_radii	List of Floats	Optional
Units: cm (default)		
Applicable Value(s): Depends on create_unique_pins (default)		
Limitation(s): None		
Description: Radii of rings for homogenization		
Notes: Applicable if homog_type is rings; experimental capability		

homog_pin_rings homog_pin_rings

homog_pin_rings	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Homogenize according to pin locations or assembly locations		
Notes: Applicable when homog_type is rings; experimental capability		

homog_explicit_ring homog_explicit_ring

homog_explicit_ring	Integer	Optional
Units: cm (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Radius within which to homogenize within and to have explicit pins outside		
Notes: Experimental capability		

bc_bnd_mesh bc_bnd_mesh

bc_bnd_mesh	array of 6 strings	Optional
Units: N/A		
Applicable Value(s): vacuum, vacuum, vacuum, vacuum, vacuum, vacuum (default), vacuum, reflect		
Limitation(s): None		
Description: Boundary mesh boundary conditions on -x, +x, -y, +y, -z, +z		
Notes: None		

x_bnd_mesh x_bnd_mesh

x_bnd_mesh	array of increasing doubles	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Boundary mesh x-axis edges		
Notes: None		

y_bnd_mesh y_bnd_mesh

y_bnd_mesh	array of increasing doubles	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Boundary mesh y-axis edges		
Notes: None		

z_bnd_mesh z_bnd_mesh

z_bnd_mesh	array of increasing doubles	Optional
Units: cm (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Boundary mesh z-axis edges		
Notes: None		

subblock_procs subblock_procs

subblock_procs	array of integers	Optional
Units: N/A		
Applicable Value(s): Unit array of size number of Shift blocks (default), ≥ 1		
Limitation(s): Only applies to domain-decomposed problems		
Description: Number of sub-block processors per domain block		
Notes: This is an advanced parameter to further specify how to domain decompose the Shift problem to handle load imbalance		

overlap overlap

overlap	double	Optional
Units: N/A		
Applicable Value(s): 0.0 (default), [0,1]		
Limitation(s): Only applies to domain-decomposed problems		
Description: Overlap percentage of domain blocks		
Notes: This is an advanced parameter to help with message passing at domain boundaries in domain-decomposed problems		

core_translate core_translate

core_translate	array of 3 doubles	Optional
Units: cm (default)		
Applicable Value(s): 0.0, 0.0, 0.0 (default)		
Limitation(s): None		
Description: Position to translate center of core		
Notes: None		

create_unique_pins create_unique_pins

create_unique_pins	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Make all pincells unique compositions		
Notes: None		

track_isotopes track_isotopes

track_isotopes	string	Optional
Units: N/A		
Applicable Value(s): short (default), full		
Limitation(s): None		
Description: Which set of isotopes to transfer		
Notes: None		

xs_library xs_library

xs_library	string	Required if problem mode is CADIS
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Name of SCALE multigroup data library file		
Notes: Applicable to hybrid simulations		

mesh mesh

mesh	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Number of mesh cells per pincell		
Notes: Applicable to hybrid simulations		

refl_mesh_size refl_mesh_size

refl_mesh_size	double	Optional
Units: N/A		
Applicable Value(s): > 0.0		
Limitation(s): None		
Description: Radial reflector region mesh size		
Notes: Applicable to hybrid simulations		

extend_axial_mesh_size extend_axial_mesh_size

extend_axial_mesh_size	double	Optional
Units: N/A		
Applicable Value(s): > 0.0		
Limitation(s): None		
Description: Axial excore region mesh size		
Notes: Applicable to hybrid simulations		

output_adjoint output_adjoint

output_adjoint	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Output adjoint flux to Shift HDF5 file and adjoint source to a separate HDF5 file		
Notes: Applicable to hybrid simulations		

adjoint adjoint

adjoint	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Perform adjoint solve		
Notes: Applicable to hybrid simulations		

num_blocks_i num_blocks_i

num_blocks_i	Integer	Optional
Units: N/A		
Applicable Value(s): depends on number of processors (default), > 0		
Limitation(s): None		
Description: Number of partitions (processors) in x		
Notes: Applicable to hybrid simulations		

num_blocks_j num_blocks_j

num_blocks_j	Integer	Optional
Units: N/A		
Applicable Value(s): depends on number of processors (default), > 0		
Limitation(s): None		
Description: Number of partitions (processors) in y		
Notes: Applicable to hybrid simulations		

num_z_blocks num_z_blocks

num_z_blocks	Integer	Optional
Units: N/A		
Applicable Value(s): depends on mesh (default), > 0		
Limitation(s): None		
Description: Number of pipelining blocks in z		
Notes: Applicable to hybrid simulations		

num_sets num_sets

num_sets	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: Number of energy sets		
Notes: Applicable to hybrid simulations		

num_groups num_groups

num_groups	Integer	Required for hybrid
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Number of energy groups		
Notes: Applicable to hybrid simulations		

max_delta_z max_delta_z

max_delta_z	double	Optional
Units: N/A		
Applicable Value(s): > 0.0		
Limitation(s): None		
Description: Maximum mesh size in z		
Notes: Applicable to hybrid simulations		

partition_upscatter partition_upscatter

partition_upscatter	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Partition energy over upscatter groups only		
Notes: Applicable to hybrid simulations		

store_fulcrum_string store_fulcrum_string

store_fulcrum_string	Boolean	Optional
Units: N/A		
Applicable Value(s): true if using 35 nodes or fewer (default), false		
Limitation(s): None		
Description: Save Fulcrum string as file		
Notes: Applicable to hybrid simulations		

upscatter_solver upscatter_solver

upscatter_solver	string	Optional
Units: N/A		
Applicable Value(s): gauss_seidel (default), gauss_seidel, gmres		
Limitation(s): None		
Description: Which upscatter solver to use		
Notes: Applicable to hybrid simulations		

within_group_solver within_group_solver

within_group_solver	string	Optional
Units: N/A		
Applicable Value(s): gmres (default)		
Limitation(s): None		
Description: Which within group solver to use		
Notes: Applicable to hybrid simulations		

iterate_downscatter iterate_downscatter

iterate_downscatter	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Iterate over downscatter groups		
Notes: Applicable to hybrid simulations		

downscatter downscatter

downscatter	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Downscatter only		
Notes: Applicable to hybrid simulations		

Pn_order Pn_order

Pn_order	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), > 0		
Limitation(s): None		
Description: Order of moments		
Notes: Applicable to hybrid simulations		

upscatter_subspace_size upscatter_subspace_size

upscatter_subspace_size	Integer	Optional
Units: N/A		
Applicable Value(s): 30 (default), > 0		
Limitation(s): None		
Description: Maximum subspace size for upscatter solver		
Notes: Applicable when upscatter_solver is gmres		

within_group_subspace_size within_group_subspace_size

within_group_subspace_size	Integer	Optional
Units: N/A		
Applicable Value(s): 20 (default), > 0		
Limitation(s): None		
Description: Maximum subspace size for within group solver		
Notes: Applicable when within_group_solver is gmres		

upscatter_max_itr upscatter_max_itr

upscatter_max_itr	Integer	Optional
Units: N/A		
Applicable Value(s): 1000 (default), > 0		
Limitation(s): None		
Description: Maximum number of iterations for upscatter solve		
Notes: Applicable to hybrid simulations		

within_group_max_itr within_group_max_itr

within_group_max_itr	Integer	Optional
Units: N/A		
Applicable Value(s): 1000 (default), > 0		
Limitation(s): None		
Description: Maximum number of iterations for within group solve		
Notes: Applicable to hybrid simulations		

eq_set eq_set

eq_set	string	Optional
Units: N/A		
Applicable Value(s): sc (default), bld, bld_2d, ld, sc		
Limitation(s): None		
Description: Solution method or spatial discretization		
Notes: Applicable to hybrid simulations		

upscatter_verbosity upscatter_verbosity

upscatter_verbosity	string	Optional
Units: N/A		
Applicable Value(s): low (default), none, low, medium, high		
Limitation(s): None		
Description: Solver verbosity		
Notes: Applicable to hybrid simulations		

within_group_verbosity within_group_verbosity

within_group_verbosity	string	Optional
Units: N/A		
Applicable Value(s): low (default), none, low, medium, high		
Limitation(s): None		
Description: Solver verbosity		
Notes: Applicable to hybrid simulations		

new_grp_bounds new_grp_bounds

new_grp_bounds	array of decreasing doubles	Optional
Units: eV (default)		
Applicable Value(s): > 0.0		
Limitation(s): None		
Description: Collapsed group boundaries		
Notes: Applicable to hybrid simulations		

grp_collapse_src grp_collapse_src

grp_collapse_src	array of doubles	Optional
Units: N/A		
Applicable Value(s): depends on xs_library (default)		
Limitation(s): None		
Description: Source to do group collapse		
Notes: Applicable to hybrid simulations		

quad_type quad_type

quad_type	string	Optional
Units: N/A		
Applicable Value(s): qr (default), qr, levelsym, galerkin, glproduct, ldfe		
Limitation(s): None		
Description: Type of S_N quadrature		
Notes: Applicable to hybrid simulations		

polars_octant polars_octant

polars_octant	Integer	Optional
Units: N/A		
Applicable Value(s): 6 (2 if adjoint) (default), > 0		
Limitation(s): None		
Description: Number of polar angles per octant for S_N quadrature		
Notes: Applicable to hybrid simulations		

azimuthals_octant azimuthals_octant

azimuthals_octant	Integer	Optional
Units: N/A		
Applicable Value(s): 8 (4 if adjoint) (default), > 0		
Limitation(s): None		
Description: Number of azimuthal angles per octant for S_N quadrature		
Notes: Applicable to hybrid simulations		

Sn_order Sn_order

Sn_order	Integer	Optional
Units: N/A		
Applicable Value(s): 4 (default), > 0		
Limitation(s): None		
Description: Level-symmetric quadrature set order		
Notes: Applicable to hybrid simulations		

upscatter_tolerance upscatter_tolerance

upscatter_tolerance	double	Optional
Units: N/A		
Applicable Value(s): 1E-04 (default), (0,1)		
Limitation(s): None		
Description: Upscatter solver convergence tolerance		
Notes: None		

within_group_tolerance within_group_tolerance

within_group_tolerance	double	Optional
Units: N/A		
Applicable Value(s): 1E-04 (default), (0,1)		
Limitation(s): None		
Description: Within-group solver convergence tolerance		
Notes: None		

cell_homogenize cell_homogenize

cell_homogenize	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Homogenize material in cells		
Notes: None		

Pn_correction Pn_correction

Pn_correction	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Use outscatter-corrected diffusion coefficient to reduce memory in solve		
Notes: None		

pin_partitioning pin_partitioning

pin_partitioning	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Partition mesh over pincells		
Notes: None		

5.10 BLOCK COBRATF

nfuel nfuel

nfuel	Integer	Optional
Units: N/A		
Applicable Value(s): 10 (default), > 0		
Limitation(s): None		
Description: The number of rings in the fuel rod pellet (only effective when nc> 0)		
Notes: None		

min_steps min_steps

min_steps	Integer	Optional
Units: N/A		
Applicable Value(s): 4 (default), ≥ 0		
Limitation(s): None		
Description: The minimum number of iterations CTF should take during a solve		
Notes: None		

imox imox

imox	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0, 1, 2, 3, 4, 5		
Limitation(s): None		
Description: The fuel thermal conductivity model to use in CTF (only effective when nc> 0). Options are: 0 - MATPRO-11 1 - Modified NFI (UO2) 2 - Halden (UO2) 3 - Duriez/Modified NFI (MOX) 4 - Halden (MOX) 5 - Amaya (MOX)		
Notes: None		

nc nc

nc	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 0, 1, 2, 3		
Limitation(s): None		
Description: This is the fuel rod conduction model. Options are: (0) No conduction, power is supplied as a surface heat flux (can lead to numerical stability issues), (1) Conduction in the radial direction only, (2) Conduction in the radial and azimuthal directions, and (3) Conduction in the radial, azimuthal and axial directions		
Notes: None		

solve_heat_end solve_heat_end

solve_heat_end	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 1		
Limitation(s): None		
Description: Set to 1 to perform the heat transfer and conduction solve performed by CTF after the steady-state fluid solve instead of during the fluid solve		
Notes: This option should not be used when modeling a transient. Also note that this must not be used for two-phase problems (cases in which significant void is expected) because it will cause an inaccurate vapor generation rate to be calculated		

chf chf

chf	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 0, 1, 2		
Limitation(s): None		
Description: CHF model option. Options are: (0) - No CHF check during the transient (a check will be made using W-3 at the completion of the CTF steady-state solve) (1) - Check CHF during transient using W-3 (2) - No CHF check during or after simulation (set CHF to infinity) (3) - No CHF check during the transient (a check will be made using the Groeneveld lookup tables at the completion of the steady-state CTF solve)		
Notes: None		

tp_fric_model tp_fric_model

tp_fric_model	String	Optional
Units: N/A		
Applicable Value(s): wallis (default), chisholm, lockhart		
Limitation(s): None		
Description: Sets the two-phase multiplier model to use in CTF. Options are as follows: • wallis: two-phase multiplier calculated based on void • chisholm: two-phase multiplier calculated using the Chisholm model • lockhart: two-phase multiplier calculated using the Lockhart-Martinelli model Descriptions of the models can be found in the CTF Theory Manual		
Notes: None		

debug debug

debug	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0, 1, 2		
Limitation(s): None		
Description: Setting to 1 will cause CTF to print every power distribution it receives before doing the solve to a separate HDF5 file. Setting to 2 will cause CTF to print every power distribution it receives similar to Option 1, but it will also print the solution after the solve. This can be used to run CTF standalone on a power distribution that causes it to crash, or it may be used to observe coupled convergence behavior		
Notes: None		

disable_xml2ctf disable_xml2ctf

disable_xml2ctf	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0, 1		
Limitation(s): None		
Description: Setting to 0 will allow xml2ctf to run during code initialization and generate the CTF input file. This is the normal VERA behavior. If set to 1, xml2ctf will not run. In this case, it is up to the user to ensure that a CTF input file called “deck.inp” is present in the simulation directory and that that model is consistent with the MPACT model. This option is provided so that a user may customize the CTF input file with options not provided through xml2ctf		
Notes: None		

irfc irfc

irfc	Integer	Optional
Units: N/A		
Applicable Value(s): 2 (default), 1, 2, 3, 4		
Limitation(s): None		
Description: Friction model: 1 - original CTF model 2 - new CTF model 3 - Colebrook 4 - Sylvester		
Notes: None		

dhfrac dhfrac

dhfrac	Float	Optional
Units: N/A		
Applicable Value(s): 0.026 (default), ≥ 0.0		
Limitation(s): None		
Description: Percentage of rod heat directly deposited into fluid (gamma heating)		
Notes: None		

bwr_dp_tol bwr_dp_tol

bwr_dp_tol	Float	Optional
Units: psi (default)		
Applicable Value(s): 0.1 (default), > 0.0		
Limitation(s): None		
Description: Solver tolerance for the pressure balance iteration loop performed in CTF for BWR models. The pressure drop in all bundles must be the same to within this tolerance for the pressure loop to exit. The pressure balance loop is used for adjusting inlet flow rates to balance the pressure drops in all assemblies in the core		
Notes: None		

crud_evap_coeff crud_evap_coeff

crud_evap_coeff	Float	Optional
Units: N/A		
Applicable Value(s): 0.0 (default), ≥ 0.0 and ≤ 1.0		
Limitation(s): None		
Description: This is the amount of crud chimney boiling that results in vapor generation in CTF. A value of zero means that none of the chimney boiling results in vapor generation in CTF, and a value of 1.0 means that 100% of the chimney boiling results in vapor generation in CTF		
Notes: None		

crud_boil_coeff_model crud_boil_coeff_model

crud_boil_coeff_model	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 1		
Limitation(s): None		
Description: Set to 1 to switch to using the coefficient-based crud boiling model when solving crud problems. Set to 0 to use the traditional explicit crud boiling model		
Notes: None		

guide_tube_coefficient guide_tube_coefficient

guide_tube_coefficient	Float	Optional
Units: N/A		
Applicable Value(s): 0.5 (default), 0.0 ≤ guide_tube_coefficient ≤ 1.0		
Limitation(s): None		

continued on next page...

guide_tube_coefficient, continued...

Description: This is used to determine the temperature rise in guide tubes using the following: T_guide_tube(z) = (T_fluid(z)-tinlet)*guide_tube_coefficient+tinlet, where T_guide_tube is the temperature in the guide tube, T_fluid is the temperature in the channels adjacent to the guide tube, and tinlet is the inlet temperature. 0.0 means the guide tube outlet temperature will be the same as the inlet temperature, and 1.0 means it will be equal to the fluid side outlet temperature
Notes: None

beta_htc beta_htc

beta_htc	Float	Optional
Units: N/A		
Applicable Value(s): 0.2 (default), > 0.0		
Limitation(s): None		
Description: This is the boiling heat transfer coefficient under-relaxation coefficient. Because of the semi-implicit coupling of the fluid and energy equations in the CTF numerical solution, it is necessary to under-relax the heat transfer coefficient in time for numerical stability. For some boiling cases, it may be necessary to increase the under-relaxation		
Notes: None		

beta_clad_creep beta_clad_creep

beta_clad_creep	Float	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), > 0.0 and <= 1.0		
Limitation(s): None		
Description: This is an under-relaxation coefficient on the clad creep effect predicted by CTFFuel. Setting this less than 1.0 will slow the impact of the effect in the CTF solution, but it will not affect results for steady-state and depletion simulations. The coefficient will only have an effect if the dynamic gap model is enabled in CTF and if a depletion is being modeled that would result in clad creep		
Notes: None		

fuel_gap_htc_beta fuel_gap_htc_beta

fuel_gap_htc_beta	Float	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), > 0.0 and ≤ 1.0		
Limitation(s): None		
Description: This is the under-relaxation coefficient used on the fuel rod gap heat transfer coefficient. This is only used when modeling a steady-state simulation. During transient portions of the simulation, this value will be ignored. This parameter only has an effect when using gap_model="dynamic"		
Notes: None		

rothcon_temp_beta rothcon_temp_beta

rothcon_temp_beta	Float	Optional
Units: N/A		
Applicable Value(s): 0.3 (default), > 0.0 and < 1.0		
Limitation(s): None		
Description: This is the under-relaxation coefficient used when calculating rod surface temperatures on the rod surface coupling mesh set up by CTF for coupling to MAMBA. It may be necessary to reduce this value if many iterations are failing during the temperature reconstruction process when using the rod thermal-hydraulic reconstruction (ROTHCON) grid files		
Notes: None		

hgap hgap

hgap	Float	Optional
Units: W/m^2/K (default)		
Applicable Value(s): 5678.3 (default), > 0.0		
Limitation(s): None		
Description: This sets the gap conductance in the fuel rod gap (only applicable when using a constant gap conductance fuel rod model)		
Notes: None		

epso epso

epso	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		
Limitation(s): None		
Description: This is the relative tolerance for the linear solver (pressure matrix solve). It is only applicable when using an iterative solver. Setting this too high can lead to numerical instability		
Notes: None		

iitmax iitmax

iitmax	Integer	Optional
Units: N/A		
Applicable Value(s): 160 (default), > 0		
Limitation(s): None		
Description: This is the maximum number of iterations to take in the linear solve (pressure matrix solve). It is only applicable when using an iterative solver. Setting this too low can lead to numerical instabilities		
Notes: None		

dtmin dtmin

dtmin	Float	Optional
Units: s (default)		
Applicable Value(s): 1.0×10^{-6} (default), > 0		

continued on next page...

dtmin, continued...

Limitation(s): None
Description: This sets the minimum allowable timestep size. It is used for both transients and steady-state, because CTF solves a transient to get to steady-state. If the timestep size needs to be reduced smaller than this value, the code will crash with a "cannot reduce timestep size" error
Notes: None

dtmax dtmax

dtmax	Float	Optional
Units: s (default)		
Applicable Value(s): 0.1 (default), > 0		
Limitation(s): None		
Description: This sets the maximum allowable timestep size. It is used for both transients and steady-state, because CTF solves a transient to get to steady-state. CTF uses dynamic timestep selection, which is mainly a function of the Courant number. This puts a ceiling on the dynamic timestep size to prevent numerical instability		
Notes: None		

rtwfp rtwfp

rtwfp	Float	Optional
Units: N/A		
Applicable Value(s): 100.0 (default), ≥ 1.0		
Limitation(s): None		
Description: This sets the ratio between the conduction and fluid timestep sizes. For steady-state problems, the timestep sizes of the conduction equation can be set larger than the fluid timestep sizes to reduce computational time. For transients, CTF will override this to be 1.0. Setting this too high can lead to numerical instability		
Notes: None		

maxits maxits

maxits	Integer	Optional
Units: N/A		
Applicable Value(s): 10000 (default), ≥ 1		
Limitation(s): None		
Description: This sets the maximum number of iterations CTF will take during any individual steady-state solve. If the iterations go over this maximum value, CTF will crash on an unable-to-converge exception		
Notes: None		

courant courant

courant	Float	Optional
Units: N/A		
Applicable Value(s): 0.8 (default), > 0.0		

continued on next page...

`courant`, continued...

Limitation(s): None
Description: This sets the Courant number to use when setting timesteps size. Setting this value lower will lead to overall smaller timestep sizes being used in CTF, and setting it higher will lead to overall larger timestep sizes being used. It is not recommended that the user adjust this value, as it typically is not an effective means of improving CTF convergence
Notes: None

solver solver

`solver`	Integer	Optional
Units: N/A		
Applicable Value(s): 3/5 (default), 0, 3, 5, 6, 7, 8		
Limitation(s): None		
Description: Selects the linear solver to use for the pressure matrix solve. Options are: 0 - Direct 3 - Internal Krylov solver (BiCGStab) (serial runs only, default for serial run) 5 - PETSc BiCGStab (default for parallel run) 6 - PETSc with pressure matrix reduced to root and solved in serial (used only for parallel verification cases, do not use for production parallel runs) 7 - PETSc BiCGStab using block Jacobi preconditioner 8 - Trillinos BiCGStab solver		
Notes: None		

parallel parallel

`parallel`	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 0, 1		
Limitation(s): None		
Description: Instructs CTF to run in serial (0) or in parallel (1)		
Notes: None		

domain_decomp domain_decomp

`domain_decomp`	2D Integer Map	Optional
Units: N/A		
Applicable Value(s): ≥ 1		
Limitation(s): None		
Description: This is a core map of the assembly domains. A domain is a group of entities that will be solved by one processor in a parallel simulation. This input is only required for nodal parallel models and will have no impact on pin-resolved models. The shape shall be the same as the core map, and each entry shall define an ID for the domain. All assemblies with the same ID will be solved by the same processor. Note that domain IDs must begin at 1 and increase incrementally. Generally, domains should be organized so that they are as compact as possible, meaning that "surface area" or interaction with adjacnet domains is minimized to limit the number of communications required between solution domains		
Notes: None		

global_energy_balance global_energy_balance

global_energy_balance	Float	Optional
Units: percent (default)		
Applicable Value(s): 0.01 (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance for energy balance (energy in minus energy out normalized to energy in) for steady-state runs		
Notes: None		

global_mass_balance global_mass_balance

global_mass_balance	Float	Optional
Units: percent (default)		
Applicable Value(s): 0.01 (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance for mass balance (mass in minus mass out normalized to mass in) for steady-state runs		
Notes: None		

fluid_energy_storage fluid_energy_storage

fluid_energy_storage	Float	Optional
Units: percent (default)		
Applicable Value(s): 0.5 (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance for energy storage in the fluid (change in energy over a timestep) for steady-state runs. It is only applicable when using the storage-based convergence criteria (when use_sol_stop_crit is 0). See the CTF user manual for more details		
Notes: None		

solid_energy_storage solid_energy_storage

solid_energy_storage	Float	Optional
Units: percent (default)		
Applicable Value(s): 0.5 (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance for energy storage in the solid (change in energy over a timestep) for steady-state runs. It is only applicable when using the storage-based convergence criteria (when use_sol_stop_crit is 0). See the CTF user manual for more details		
Notes: None		

mass_storage mass_storage

mass_storage	Float	Optional
Units: percent (default)		
Applicable Value(s): 0.5 (default), > 0.0		

continued on next page...

mass_storage, continued...

Limitation(s): None
Description: This sets the tolerance for mass storage in the fluid (change in mass in system over a timestep) for steady-state runs. It is only applicable when using the storage-based convergence criteria (when use_sol_stop_crit is 0). See the CTF user manual for more details
Notes: None

pressure_criteria pressure_criteria

pressure_criteria	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance on l-infinity of pressure change for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

Tcool_criteria Tcool_criteria

Tcool_criteria	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-3} (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance on l-infinity of coolant temperature for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

Tsolid_criteria Tsolid_criteria

Tsolid_criteria	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-3} (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance on l-infinity of solid temperature for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

void_criteria void_criteria

void_criteria	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		

continued on next page...

`void_criteria`, continued...

Limitation(s): None
Description: This sets the tolerance on l-infinity of void for steady-state runs. It is only applicable when using the change-based convergence criteria (when `use_sol_stop_crit`=1). Note that when using the change-based criteria, all criteria are optional. It is not checked for single-phase runs. See the CTF user manual for more details
Notes: None

vliq_criteria vliq_criteria

`vliq_criteria`	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-3} (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance on l-infinity of liquid velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when `use_sol_stop_crit`=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

vvap_criteria vvap_criteria

`vvap_criteria`	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-2} (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance on l-infinity of vapor velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when `use_sol_stop_crit`=1). Note that when using the change-based criteria, all criteria are optional. It is not checked for single-phase runs. See the CTF user manual for more details		
Notes: None		

vdrop_criteria vdrop_criteria

`vdrop_criteria`	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-2} (default), > 0.0		
Limitation(s): None		
Description: This sets the tolerance on l-infinity of droplet velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when `use_sol_stop_crit`=1). Note that when using the change-based criteria, all criteria are optional. It is not checked for single-phase runs. See the CTF user manual for more details		
Notes: None		

pressurea_criteria pressurea_criteria

pressurea_criteria	Float	Optional
Units: bar (default)		
Applicable Value(s): 1.0×10^{-3} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-infinity of pressure for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

Tcoola_criteria Tcoola_criteria

Tcoola_criteria	Float	Optional
Units: K (default)		
Applicable Value(s): 1.0×10^{-3} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-infinity of coolant temperature for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

Tsolida_criteria Tsolida_criteria

Tsolida_criteria	Float	Optional
Units: K (default)		
Applicable Value(s): 1.0×10^{-3} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-infinity of solid temperature for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

vliqa_criteria vliqa_criteria

vliqa_criteria	Float	Optional
Units: m/s (default)		
Applicable Value(s): 1.0×10^{-3} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-infinity of liquid velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

vvapa_criteria vvapa_criteria

vvapa_criteria	Float	Optional
Units: m/s (default)		
Applicable Value(s): 1.0×10^{-2} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-infinity of vapor velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. It is not used for single-phase runs. See the CTF user manual for more details		
Notes: None		

vdropa_criteria vdropa_criteria

vdropa_criteria	Float	Optional
Units: m/s (default)		
Applicable Value(s): 1.0×10^{-2} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-infinity of droplet velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. It is not used for single-phase runs. See the CTF user manual for more details		
Notes: None		

pressure_criteria_l2 pressure_criteria_l2

pressure_criteria_l2	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-5} (default), > 0.0		
Limitation(s): None		
Description: This sets the relative tolerance on l-2 of pressure for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

Tcool_criteria_l2 Tcool_criteria_l2

Tcool_criteria_l2	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		
Limitation(s): None		
Description: This sets the relative tolerance on l-2 of coolant temperature for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

Tsolid_criteria_l2 Tsolid_criteria_l2

Tsolid_criteria_l2	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		
Limitation(s): None		
Description: This sets the relative tolerance on l-2 of solid temperature for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

void_criteria_l2 void_criteria_l2

void_criteria_l2	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-5} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-2 of void for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. It is not used in single-phase runs. See the CTF user manual for more details		
Notes: None		

vliq_criteria_l2 vliq_criteria_l2

vliq_criteria_l2	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		
Limitation(s): None		
Description: This sets the relative tolerance on l-2 of liquid velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

vvap_criteria_l2 vvap_criteria_l2

vvap_criteria_l2	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		
Limitation(s): None		
Description: This sets the relative tolerance on l-2 of vapor velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. It is not used for single-phase runs. See the CTF user manual for more details		
Notes: None		

vdrop_criteria_l2 vdrop_criteria_l2

vdrop_criteria_l2	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		
Limitation(s): None		
Description: This sets the relative tolerance on l-2 of droplet velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. It is not used for single-phase runs. See the CTF user manual for more details		
Notes: None		

pressurea_criteria_l2 pressurea_criteria_l2

pressurea_criteria_l2	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-5} (default), > 0.0		
Limitation(s): None		
Description: This sets the relative tolerance on l-2 of pressure for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

Tcoola_criteria_l2 Tcoola_criteria_l2

Tcoola_criteria_l2	Float	Optional
Units: K (default)		
Applicable Value(s): 1.0×10^{-5} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-2 of coolant temperature for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

Tsolida_criteria_l2 Tsolida_criteria_l2

Tsolida_criteria_l2	Float	Optional
Units: K (default)		
Applicable Value(s): 1.0×10^{-5} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-2 of solid temperature for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

vliqa_criteria_l2 vliqa_criteria_l2

vliqa_criteria_l2	Float	Optional
Units: m/s (default)		
Applicable Value(s): 1.0×10^{-5} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-2 of liquid velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. See the CTF user manual for more details		
Notes: None		

vvapa_criteria_l2 vvapa_criteria_l2

vvapa_criteria_l2	Float	Optional
Units: m/s (default)		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-2 of vapor velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. It is not used for single-phase runs. See the CTF user manual for more details		
Notes: None		

vdropa_criteria_l2 vdropa_criteria_l2

vdropa_criteria_l2	Float	Optional
Units: m/s (default)		
Applicable Value(s): 1.0×10^{-4} (default), > 0.0		
Limitation(s): None		
Description: This sets the absolute tolerance on l-2 of droplet velocity for steady-state runs. It is only applicable when using the change-based convergence criteria (when use_sol_stop_crit=1). Note that when using the change-based criteria, all criteria are optional. It is not used for single-phase runs. See the CTF user manual for more details		
Notes: None		

use_sol_stop_crit use_sol_stop_crit

use_sol_stop_crit	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		

continued on next page...

use_sol_stop_crit, continued...

Description: Selects the stopping criteria to use for steady-state runs. Options are: 0 - storage-based criteria (global_energy_balance, global_mass_balance, fluid_energy_storage, solid_energy_storage, mass_storage) 1 - change-based criteria (global_energy_balance, global_mass_balance, pressure_criteria, pressurea_criteria, Tcool_criteria, Tcoola_criteria, Tsolid_criteria, Tsolida_criteria, void_criteria, vliq_criteria, vliqa_criteria, vvap_criteria, vvapa_criteria, vdrop_criteria, vdropa_criteria, void_criteria_l2, Tcool_criteria_l2, Tcoola_criteria_l2, Tsolid_criteria_l2, Tsolida_criteria_l2, pressure_criteria_l2, pressurea_criteria_l2, vliq_criteria_l2, vliqa_criteria_l2, vvap_criteria_l2, vvapa_criteria_l2, vdrop_criteria_l2, vdropa_criteria_l2). All criteria are optional with defaults
Notes: None

proc_per_assem proc_per_assem

proc_per_assem	Integer	Optional
Units: N/A		
Applicable Value(s): 9 (default), 1 4 9 16		
Limitation(s): None		
Description: This sets the number of domains to divide each full assembly into for parallel runs. It is only applicable for parallel runs. The higher the number, the more cores CTF will use and the faster it will run in a parallel model. However, the number of cores required by CTF must be less than or equal to the number required by VERA and the number of cores available on the system		
Notes: For BWR models, proc_per_assem can only be set to 1		

edit_gaps edit_gaps

edit_gaps	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write an output file from CTF specifying gap (lateral flow path) solution data. This file will be large for full-core models		
Notes: This optional only works for serial models		

edit_main_text_output edit_main_text_output

edit_main_text_output	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write the main text output file from CTF summarizing solution data. This file will be large for full-core models		
Notes: This optional only works for serial models		

edit_channels edit_channels

edit_channels	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write the channel text output file from CTF summarizing solution data. This file will be large for full-core models		
Notes: This optional only works for serial models		

edit_th_details edit_th_details

edit_th_details	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 1		
Limitation(s): None		
Description: Set this to 1 to write more detailed fluid solution data from CTF to the VERA HDF5 output file		
Notes: None		

edit_rods edit_rods

edit_rods	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write rod data to the main text output file from CTF. This file will be large for full-core models		
Notes: This optional only works for serial models		

edit_dnb edit_dnb

edit_dnb	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write DNB data to the VERA HDF5 file		
Notes: None		

edit_dnb_text_file edit_dnb_text_file

edit_dnb_text_file	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write the DNB text file. This file will be large for full-core models		
Notes: This optional only works for serial models		

edit_convergence edit_convergence

edit_convergence	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write the convergence information output file from CTF		
Notes: None		

edit_hdf5 edit_hdf5

edit_hdf5	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write CTF data to the VERA HDF5 file		
Notes: None		

edit_native_hdf5 edit_native_hdf5

edit_native_hdf5	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write the CTF native HDF5 file. This file writes information for all pins in the model in a more arbitrary way than the VERA HDF5 file, which is organized by assembly and core location. This file contains more detailed information than the VERA HDF5 file		
Notes: None		

edit_fluid_vtk edit_fluid_vtk

edit_fluid_vtk	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to write the CTF fluid VTK file. This allows the user to visualize solution results using a VTK reader, but this file will be large for full-core models		
Notes: None		

edit_rod_vtk edit_rod_vtk

edit_rod_vtk	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		

continued on next page...

edit_rod_vtk, continued...

Description: Set this to 1 to write the CTF rod VTK file. This allows the user to visualize solution results using a VTK reader, but this file will be large for full-core models
Notes: None

hi2lo_sub_axial hi2lo_sub_axial

hi2lo_sub_axial	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: This is used to set the number of sub-levels to divide each CTF axial level into when forming the coupling mesh with MAMBA. It is only applicable when using ROTHCON to reconstruct rod surface temperatures and TKE		
Notes: None		

hi2lo_sub_theta hi2lo_sub_theta

hi2lo_sub_theta	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: This is used to set the number of sub-sectors to divide each CTF rod sector into when forming the coupling mesh with MAMBA. It is only applicable when using ROTHCON to reconstruct rod surface temperatures and TKE		
Notes: None		

hi2lo_grid hi2lo_grid

hi2lo_grid	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: This specifies which grids should have their coupling mesh refined using hi2lo_sub_theta and hi2lo_sub_axial. Leaving this out means that all grid spans will be refined. The grid span numbering ranges from 1 to the number of grid spans (number of grid_axial entries in the ASSEMBLY block). The region below the first grid does not count as a span		
Notes: This only has an effect when using the ROTHCON capability for reconstructing grid heat transfer and turbulence enhancement behavior		

model_corrosion model_corrosion

model_corrosion	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		

continued on next page...

`model_corrosion`, continued...

Description: Set this to 1 to turn on the clad corrosion model in CTF. It is only applicable for crud simulations
Notes: None

int_drag_model int_drag_model

`int_drag_model`	String	Optional
Units: N/A		
Applicable Value(s): legacy (default), drift_flux		
Limitation(s): None		
Description: Use to set the interfacial drag model that CTF shall use		
Notes: None		

flow_regime_map flow_regime_map

`flow_regime_map`	String	Optional
Units: N/A		
Applicable Value(s): legacy (default), ge_nonprop		
Limitation(s): None		
Description: Selects the flow regime map to be used in CTF		
Notes: None		

th_solver th_solver

`th_solver`	String	Optional
Units: N/A		
Applicable Value(s): ctf (default), fireant, ants		
Limitation(s): None		
Description: Selects the fluid solver to use for the TH solution. Options are `ctf`, `fireant`, or `ants`. The `fireant` and `ants` options are steady-state solvers that are much faster than `ctf`, but they only do an axial sweep, and they lack many of the more advanced models of `ctf`. Regardless of the TH solver, CTFFuel will be used for the pin temperature solution		
Notes: None		

gap_model gap_model

`gap_model`	string	Optional
Units: N/A		
Applicable Value(s): constant (default), constant dynamic		
Limitation(s): None		
Description: This sets the fuel rod pellet/clad gap thermal conductivity model. It can either be constant (user-specified value) or dynamic (CTF will calculate based on thermal expansion and burnup effects)		
Notes: None		

boil_ht_cor boil_ht_cor

boil_ht_cor	string	Optional
Units: N/A		
Applicable Value(s): thom (default), chen, thom, gorenflo		
Limitation(s): None		
Description: This sets the boiling heat transfer model. Note that when gorenflo is selected, the ONB model is also used in CTF for determining when heat transfer transitioning to boiling heat transfer		
Notes: None		

property_evaluations property_evaluations

property_evaluations	string	Optional
Units: N/A		
Applicable Value(s): iapws1997_lookup (default), asme1968 iapws1997_direct iapws1997_lookup flibe		
Limitation(s): None		
Description: This sets the equation of state source to use for fluid properties. Options are: asme1968 - ASME 1968 tables iapws1997_direct - IAPWS 1997 standard using direct correlation evaluations (will be computationally slower) iapws1997_lookup - IAPWS 1997 standard lookup tables built from the direct correlation evaluations during initialization (computationally faster to evaluate) flibe - Generic properties for FLiBe salt coolant		
Notes: None		

beta_sp beta_sp

beta_sp	Float	Optional
Units: N/A		
Applicable Value(s): 0.037 (default), ≥ 0.0		
Limitation(s): None		
Description: This sets the strength of turbulent mixing causing lateral cross-flow in CTF. The default is currently 0.037		
Notes: None		

k_void_drift k_void_drift

k_void_drift	Float	Optional
Units: N/A		
Applicable Value(s): 1.4 (default), ≥ 0.0		
Limitation(s): None		
Description: This sets the equilibrium distribution weighting factor in the void drift model. Decreasing this value leads to less void drift and increasing it leads to more		
Notes: None		

crud_tool crud_tool

crud_tool	string	Optional
Units: N/A		
Applicable Value(s): MAMBA (default), MAMBA cicada		

continued on next page...

`crud_tool`, continued...

Limitation(s): None
Description: This sets the crud modeling tool. Only applicable during a crud simulation. Note that Cicada is an experimental feature
Notes: None

max_crud_step_size max_crud_step_size

`max_crud_step_size`	Float	Optional
Units: day (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: This is the maximum number of days in a crud grow. Setting this smaller than the depletion step size will result in multiple crud grows being made during the depletion step, with source term data being updated during each substep		
Notes: None		

crud_dT_feedback crud_dT_feedback

`crud_dT_feedback`	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 0		
Limitation(s): None		
Description: Set this to 0 to shut off the crud thermal resistance effect on the rod internal temperature calculation. Note that the crud thermal resistance will still affect the corrosion growth calculation		
Notes: None		

cicada_outer_radial_zone_num_cells_r cicada_outer_radial_zone_num_cells_r

`cicada_outer_radial_zone_num_cells_r`	Integer	Optional
Units: N/A		
Applicable Value(s): 100 (default), ≥ 1		
Limitation(s): None		
Description: This sets the number of rings in the oxide region of the clad for Cicada runs. It is only applicable when Cicada is used as the crud tool. It is only applicable when `cicada_dimension`=3. Note that Cicada is an experimental feature		
Notes: None		

cicada_inner_radial_zone_num_cells_r cicada_inner_radial_zone_num_cells_r

`cicada_inner_radial_zone_num_cells_r`	Integer	Optional
Units: N/A		
Applicable Value(s): 20 (default), ≥ 1		
Limitation(s): None		

continued on next page...

`cicada_inner_radial_zone_num_cells_r`, continued...

Description: This sets the number of rings in the clad region of the clad for Cicada runs. Only applicable when Cicada used as the crud tool. It is only applicable when `cicada_dimension`=3. Note that Cicada is an experimental feature
Notes: None

cicada_outer_radial_zone_thickness cicada_outer_radial_zone_thickness

`cicada_outer_radial_zone_thickness`	Float	Optional
Units: m (default)		
Applicable Value(s): 100.0×10^{-6} (default), > 0.0		
Limitation(s): None		
Description: This sets the thickness of the oxide modeling region of the clad. It is only applicable for crud simulations in which Cicada is being used as the modeling tool. It is only applicable when `cicada_dimension`=3. Note that Cicada is an experimental feature		
Notes: None		

cicada_dimensions cicada_dimension

`cicada_dimensions`	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 1 3		
Limitation(s): None		
Description: This chooses the dimensions of the clad/oxide conduction solution in Cicada. It is only applicable when doing a crud simulation using Cicada as the crud tool. Can either be 1 for radial conduction only or 3 for radial/axial/azimuthal conduction. Note that Cicada is an experimental feature		
Notes: None		

enable_corrosion_lithium enable_corrosion_lithium

`enable_corrosion_lithium`	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to turn on the lithium effect on clad corrosion. It is only has an effect when modeling a crud simulation using MAMBA as the crud code		
Notes: None		

crud_details crud_details

`crud_details`	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to turn on additional edits to the VERA HDF5 file related to the crud simulation		
Notes: None		

rod_details rod_details

rod_details	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to turn on additional edits to the VERA HDF5 file related to the rod solution		
Notes: None		

oxide_thermal_conductivity oxide_thermal_conductivity

oxide_thermal_conductivity	Float	Optional
Units: W/cm/K (default)		
Applicable Value(s): 1.5 (default), Greater than or equal to 0.0		
Limitation(s): None		
Description: The thermal conductivity of the clad oxide layer		
Notes: None		

clad_corrosion_model clad_corrosion_model

clad_corrosion_model	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 1 2 3		
Limitation(s): None		
Description: This selects the corrosion model to use. The corrosion model is based on the clad material. Options include the following: 1 - Zirc 4 2 - M5 3 - ZIRLO		
Notes: None		

trans_dnb trans_dnb

trans_dnb	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0 1		
Limitation(s): None		
Description: Set this to 1 to enable the transient CHF model. It is only applicable for transients		
Notes: None		

cross_flow cross_flow

cross_flow	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 0 1		
Limitation(s): None		
Description: Set this to 0 to shut off lateral cross flow in CTF		
Notes: None		

plr_gap_effect plr_gap_effect

plr_gap_effect	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 1		
Limitation(s): None		
Description: When modeling geometry that includes part-length fuel rods, setting this to 1 will include the effect of the gap (between adjacent fuel rods) width increasing downstream of where the part-length rod disappears. Leaving it to the default of 0 will keep the gap width constant for the entire model. It has been found that the discontinuity in the gap width axially in the model can lead to more difficulty in converging CTF. Because the effect of the gap width is less significant than the change channel area and wetted perimeter (which is always captured), this was made to be optional		
Notes: None		

allow_fuzzy_grid_placement allow_fuzzy_grid_placement

allow_fuzzy_grid_placement	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 1		
Limitation(s): None		
Description: If this is set to 1 (true), then it relaxes the requirement that mesh cell boundaries line up with the spacer grid bottom and top. In this case, the grid effects will be moved to the nearest mesh boundary in the CTF model, and a warning will be printed. If this is set to 0, then an error will be raised if the grid top and bottom do not line up with mesh cell boundaries		
Notes: None		

nodal_inter_assem_gap_width_uniform nodal_inter_assem_gap_width_uniform

nodal_inter_assem_gap_width_uniform	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 0		
Limitation(s): None		
Description: If this is set to 1 (true) and it is creating a nodal model, then the widths of the gaps between assemblies will use the gap width of the gaps inside the assemblies, which results in more uniform gap widths throughout the model. This only affects nodal models		
Notes: None		

nodal_gap_len_node_centers nodal_gap_len_node_centers

nodal_gap_len_node_centers	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 0		
Limitation(s): None		

continued on next page...

nodal_gap_len_node_centers, continued...

Description: If this is set to 1 (true) and it is creating a nodal model, then the lengths of the gaps will be calculated using the distance between the node centers. If it is set to 0 (false), then the distance between the actual subchannels on either side of the node boundary (pin pitch plus assembly spacing) will be used. This only affects nodal models
Notes: None

nodal_inter_assem_loss nodal_inter_assem_loss

nodal_inter_assem_loss	Float	Optional
Units: N/A		
Applicable Value(s): 0.5 (default), ≥ 0.0		
Limitation(s): None		
Description: Sets the form loss coefficient in the gaps between assemblies in nodal models. This only affects nodal models		
Notes: None		

5.11 BLOCK COUPLING

epsk epsk

epsk	Float	Optional
Units: pcm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Eigenvalue convergence criteria		
Notes: None		

epsp epsp

epsp	Float	Optional
Units: L2 norm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Power convergence criteria		
Notes: None		

eps_temp eps_temp

eps_temp	Float	Optional
Units: degrees F (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Temperature convergence criteria		
Notes: None		

ctf_iters_max ctf_iters_max

ctf_iters_max	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Maximum number of CTF time-steps per coupled iteration		
Notes: None		

ctf_iters_growth ctf_iters_growth

ctf_iters_growth	Float	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Fractional change in ctf_iters_max by coupled iteration		
Notes: Value of 1 is no change		

eps_boron eps_boron

eps_boron	Float	Optional
Units: ppm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Boron convergence criteria		
Notes: None		

rlx_power rlx_power

rlx_power	Float	Optional
Units: N/A		
Applicable Value(s): > 0,≤ 1		
Limitation(s): None		
Description: Power relaxation factor		
Notes: Recommend 0.5		

rlx_tfuel rlx_tfuel

rlx_tfuel	Float	Optional
Units: N/A		
Applicable Value(s): > 0,≤ 1		
Limitation(s): None		
Description: Fuel temperature relaxation factor		
Notes: Recommend 1.0		

rlx_den rlx_den

rlx_den	Float	Optional
Units: N/A		
Applicable Value(s): > 0,≤ 1		
Limitation(s): None		
Description: Density relaxation factor		
Notes: Recommend 1.0		

extend_coupling_mesh extend_coupling_mesh

extend_coupling_mesh	String	Optional
Units: N/A		
Applicable Value(s): none (default), above, below, both		
Limitation(s): None		
Description: This card is used to specify whether to enable coupling above and below the active fuel when using CTF. The extended coupling can be enabled only above the fuel with above, only below it with below, or both above and below with both. Deprecated options true and false correspond to both and none, respectively		
Notes: None		

rlx_crud rlx_crud

rlx_crud	Float	Optional
Units: N/A		
Applicable Value(s): > 0,≤ 1		
Limitation(s): None		
Description: Crud relaxation factor		
Notes: Recommend 0.5		

nonlinear_coupling_method unknown_quantity method

unknown_quantity	String	Optional
Units: N/A		
Applicable Value(s): pin_powers, mod_dens		
Limitation(s): None		
Description: Specifies unknown quantities for which the user would like to choose the solution procedure		
Notes: This is only effective for coupled (multiphysics) simulations. If the nonlinear_coupling_method card is not used, then all unknowns default to being solved with Picard iterations		

method	String	Optional
Units: N/A		
Applicable Value(s): picard (default), anderson, partconv		
Limitation(s): None		
Description: The solution method to be used for the specified coupling unknown		

continued on next page...

method, continued...

Notes: Only effective for coupled (multiphysics) simulations. When partconv is is used, anderson should not be used for the other parameters and relaxation factor should be 1

anderson_options unknown_quantity depth mixing_parameter starting_iteration

unknown_quantity	String	Optional
Units: N/A		
Applicable Value(s): pin_powers, mod_dens		
Limitation(s): None		
Description: Specifies the unknown quantities for which the user would like to set the Anderson solver control parameters		
Notes: Only effective for coupled (multiphysics) simulations and for cases in which Anderson was chosen for the nonlinear_coupling_method corresponding to the specified unknown_quantity		

depth	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), ≥ 1, < 1000		
Limitation(s): None		
Description: In Anderson Acceleration, the depth is the number of previous iterates the solver should use in generating better future iterates. Larger choices for depth result in more aggressive acceleration; however, this can lead to instability and slower speed overall for highly nonlinear problems. A depth of 0 is equivalent to classic Picard iteration		
Notes: Only effective for coupled (multiphysics) simulations		

mixing_parameter	Float	Optional
Units: N/A		
Applicable Value(s): 0.5 (default), > 0.0, ≤ 1.0		
Limitation(s): None		
Description: The mixing_parameter can be viewed as a damping or under-relaxation factor in the Anderson solution scheme. Obviously, this means larger choices for this parameter can result in more aggressive acceleration, although that does not always translate into better performance. The optimum choice will be problem dependent, with the default of 0.5 being quite conservative		
Notes: Only effective for coupled (multiphysics) simulations		

starting_iteration	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0, < 1000		
Limitation(s): None		
Description: The starting_iteration is the iteration at which the user would like Anderson to actually start accelerating the chosen unknown. While the iteration count is below this number, the Anderson solver will proceed like classic Picard, using the mixing_parameter as an under-relaxation factor. Normally the default choice of 1 is best		
Notes: Only effective for coupled (multiphysics) simulations		

partconv_opt gammamode est_fdintense dperturb fdmul par_nchk par_multchk

gammamode	String	Optional
Units: N/A		
Applicable Value(s): hyb (default), debug, maxave		
Limitation(s): None		
Description: This card is used to define the solver method for the nearly optimal partial converged CMFD nonlinear solver; described as • debug: print out the feedback intensity for each iteration • maxave: use either the maximum or flux-weighted feedback intensity • hyb: use maximum feedback intensity when shielding calculation has been performed; use flux-weighted feedback intensity when not		
Notes: The card is only used when partconv is specified in nonlinear_coupling_method		

est_fdintense	Float	Optional
Units: N/A		
Applicable Value(s): 1.5e-3 (default), > 0, < 1		
Limitation(s): None		
Description: This card is used to specify the estimated feedback intensity before calculating the feedback intensity		
Notes: The card is only used when partconv is specified in nonlinear_coupling_method		

dperturb	Float	Optional
Units: N/A		
Applicable Value(s): 5e-3 (default), > 0		
Limitation(s): None		
Description: This card is used to specify the perturbation factor to calculate the feedback intensity		
Notes: The card is only used when partconv is specified in nonlinear_coupling_method		

fdmul	Float	Optional
Units: N/A		
Applicable Value(s): 1.0 (default)		
Limitation(s): None		
Description: This card is used to specify the multiplication factor applied to the feedback intensity calculated during the perturbation. When gammamode is maxave or debug, a positive input makes the feedback intensity the maximum of all the feedback intensities; while a negative input makes value the flux-weighted of all the feedback intensities		
Notes: The card is only used when partconv is specified in nonlinear_coupling_method. The multiplication factor is applied to the feedback intensities that is calculated from perturbation		

par_nchk	Integer	Optional
Units: N/A		

continued on next page...

par_nchk, continued...

Applicable Value(s): 4 (default), > 0
Limitation(s): None
Description: The initial guess of the flux is flat and far from its converged value for the single-state simulation and the first state of the multi-state simulation. The par_nchk is the index of the iteration at which the user believes the flux is close to its converged value for the first state and the estimation of the feedback intensity can be performed
Notes: The card is only used when partconv is specified in nonlinear_coupling_method, and the calculation is a multi-state simulation

par_multchk	Integer	Optional
Units: N/A		
Applicable Value(s): 2 (default), > 0		
Limitation(s): None		
Description: The par_multchk is the index of the iteration at which the user believes the flux is close to its converged value for the states other than the first state and the estimation of the feedback intensity can be performed		
Notes: The card is only used when partconv is specified in nonlinear_coupling_method		

maxiter maxiter

maxiter	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Maximum number of coupled iterations		
Notes: None		

read_restart read_restart

read_restart	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This is the name of coupling restart file. Leave this blank for no coupling restart		
Notes: None		

5.12 BLOCK TIAMAT

solver solver

solver	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): jacobi,gauss-seidel		
Limitation(s): None		
Description: Tiamat solver type		

continued on next page...

`solver`, continued...

Notes: None

run_transient run_transient

`run_transient`	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Boolean indicating if transient or not		
Notes: None		

conserve_power_in_tiamat_transfer conserve_power

`conserve_power`	Boolean	Optional
Units: N/A		
Applicable Value(s): true, false		
Limitation(s): None		
Description: Boolean indicating power is conserved in transfers		
Notes: None		

num_subcycle_iterations_before_tiamat_ramping num_subcycle

`num_subcycle`	Integer	Optional
Units: N/A		
Applicable Value(s): > -1		
Limitation(s): None		
Description: Number of subcycle iterations before ramping in Tiamat		
Notes: None		

coupling_approach coupling_approach

`coupling_approach`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): inline (default), coupled		
Limitation(s): None		
Description: Describes whether inline (one-way) or coupled (two-way) coupling between MPACT/CTF and BISON is used		
Notes: None		

overlap_procs overlap_procs

`overlap_procs`	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		

continued on next page...

overlap_procs, continued...

Limitation(s): None
Description: Describes whether BISON processors are overlapped with those for MPACT or if they are completely separate
Notes: None

5.13 BLOCK MPACT

transport_method transport_method

transport_method	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): MOC (default), moc, sn, nodal-nem, nodal-senm-2, nodal-senm-4		
Limitation(s): None		
Description: This card is used to specify whether method of characteristics, Sn, or nodal diffusion transport methods are used for the global problem solution method		
Notes: None		

gamma_transport gamma_transport

gamma_transport	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card is used to enable/disable the gamma transport calculation		
Notes: None		

sn_numcart sn_numcart

sn_numcart	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default)		
Limitation(s): None		
Description: This card is used to specify the number of X and Y pincell sub-divisions in which to divide each pincell into for the Sn Transport sweeper		
Notes: None		

ray_spacing ray_spacing

ray_spacing	Floating-Point Real Number	Optional
Units: cm (default)		
Applicable Value(s): 0.05 (default), Positive floating-point real numbers		
Limitation(s): None		

continued on next page...

`ray_spacing`, continued...

Description: This card is used to specify the characteristic ray spacing for the rays used in the MOC calculation. A finer spacing will permit a more detailed calculation (with finer spatial features) at the cost of computing time. However, the decomposition of rays across multiple threads parallelizes very efficiently. Finally, one should be cognizant of minimum feature size (i.e., minimum flat-source region size) to ensure that there are an adequate number of rays traversing each region to have an accurate solution in that region. More information regarding the MOC methodology and implications of `ray_spacing` on the overall calculation is available in the MPACT Theory Manual
Notes: None

gamma_ray_spacing gamma_ray_spacing

`gamma_ray_spacing`	Floating-Point Real Number	Optional
Units: cm (default)		
Applicable Value(s): 0.05 (default), Positive floating-point real numbers		
Limitation(s): None		
Description: This card is used to specify the characteristic ray spacing for the rays used in the MOC calculation for gammas		
Notes: None		

shield_ray_spacing shield_ray_spacing

`shield_ray_spacing`	Floating-Point Real Number	Optional
Units: cm (default)		
Applicable Value(s): 0.05 (default), Positive floating-point real numbers		
Limitation(s): None		
Description: This card is used to specify the characteristic ray spacing for the rays used in the MOC shielding calculation. A finer spacing will permit a more detailed calculation (with finer spatial features) at the cost of computing time. However, the decomposition of rays across multiple threads parallelizes very efficiently. Finally, one should be cognizant of minimum feature size (i.e., minimum flat-source region size) to ensure that there are an adequate number of rays traversing each region to have an accurate solution in that region. More information regarding the MOC methodology and implications of `ray_spacing` on the overall calculation is available in the MPACT Theory Manual		
Notes: None		

log_message log_message

`log_message`	Character String	Optional
Units: N/A		
Applicable Value(s): warn (default), debug, basic		
Limitation(s): None		
Description: This card is used to specify which type of messages should be written to the log file		
Notes: None		

refl_assembly_layers refl_assembly_layers

refl_assembly_layers	String	Optional
Units: N/A		
Applicable Value(s): 1 (if PWR), 2 (if BWR) (default), ≥ 0, all, none, default		
Limitation(s): None		
Description: This card is used to specify the number of assembly layers to be added for the radial reflector region: that is, anything outside of the fuel radially. A given number is used as a hard limit on the number of assembly layers added, `all` puts no limit on the number of assemblies added; however, only those needed for core baffle, barrel, and vessel modeling will be added		
Notes: Only features or parts of features that fit within the given reflector thickness or default thickness will be modeled		

refl_highres refl_highres

refl_highres	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card is used to enable the reflector high resolution flag. If enabled, vessel components are read as holes instead		
Notes: None		

moc_kernel moc_kernel

moc_kernel	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): MG (default), 1G,kokkos		
Limitation(s): None		
Description: This card is used to specify whether one-group or multi-group MOC kernels are used for neutron transport		
Notes: None		

gamma_moc_kernel gamma_moc_kernel

gamma_moc_kernel	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): MG (default), 1G		
Limitation(s): None		
Description: This card is used to specify whether one-group or multi-group MOC kernels are used for gamma transport		
Notes: None		

shield_moc_kernel shield_moc_kernel

shield_moc_kernel	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): same value as moc_kernel (default), 1G,MG		

continued on next page...

shield_moc_kernel, continued...

Limitation(s): None
Description: This card is used to specify whether one-group or multi-group MOC kernels are used for the shielding sweeper
Notes: None

moc_mg_data_passing moc_mg_data_passing

moc_mg_data_passing	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: This card is used to specify whether one-group or multi-group MOC data passing is used		
Notes: This is primarily to bypass the message passing interface (MPI) issues observed with the multi-group angular flux and is only applicable when using moc_kernel=MG		

moc_blocking_data_passing moc_blocking_data_passing

moc_blocking_data_passing	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card is used to specify whether blocking or non-blocking MOC data passing is used		
Notes: None		

moc_rational_frac_tol moc_rational_frac_tol

moc_rational_frac_tol	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): $\max\left(0.001, \min\left(0.02, 10^{(-1.2653-0.0271\cdot n_{azi})}\right)\right)$ (default), Positive floating-point real numbers		
Limitation(s): None		
Description: This card is used to set the tolerance of the rational fractions calculation that is part of the modular angle-spacing pair setup. A default is defined to set reasonable values for the likely azimuthals_octant values ranging between 2 and 64. However, azimuthals_octant>= 64 may require a tighter tolerance to achieve accurate results. Therefore, the user may use this card to override the default behavior as needed		
Notes: None		

moc_min_flux moc_min_flux

moc_min_flux	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		

continued on next page...

moc_min_flux, continued...

Description: This card is used to prevent a non-positive MOC scalar flux. At the end of each MOC sweep, any MOC flux less than moc_min_flux is set to moc_min_flux
Notes: A very small positive real number such as 1.0×10^{-20} is recommended

volume_corr volume_corr

volume_corr	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): none (default), integral, angledep		
Limitation(s): None		
Description: This card is used to specify the volume correction being applied to the MOC segments		
Notes: The integral option can be used to significantly improve the convergence properties of certain transient calculations		

modular_rays modular_rays

modular_rays	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): two (default), decart, two, three, cactus, ratfrac		
Limitation(s): None		
Description: This card is used to specify the volume correction being applied to the MOC segments		
Notes: None		

radial_src_order radial_src_order

radial_src_order	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0,1		
Limitation(s): Zero or positive integers		
Description: This card is used to read the source order in the radial direction		
Notes: Currently only flat(0) and linear(1) are implemented		

axial_src_order axial_src_order

axial_src_order	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 0,1		
Limitation(s): Zero or positive integers		
Description: This card is used to read the source order in the axial direction		
Notes: Currently only flat(0) and linear(1) are implemented		

power_edit power_edit

power_edit	Fixed Character String	Optional

continued on next page...

power_edit, continued...

Units: N/A
Applicable Value(s): KAPPA-FISSION (default), FISSION,GAMMA-SMEARED
Limitation(s): None
Description: This card is used to specify a cross section used for the "power" calculations. KAPPA-FISSION is the standard power calculation, whereas FISSION actually produces the normalized fission reaction rate distribution, and GAMMA-SMEARED calculates the normalized gamma smeared power distribution
Notes: None

jagged jagged

jagged	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): See notes regarding potential inefficiencies when running a parallel-processing simulation		
Description: This card is used to specify whether the reflector region will be modeled using a jagged (stair-step) representation or by filling the full square extent of the modeling domain with moderator material		
Notes: When a jagged core is used, care should be taken if the user elects to perform manual parallel domain decomposition to ensure proper load balancing. Additional information is provided with the par_file		

rod_treatment rod_treatment

rod_treatment	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): none (default), polynomial,1dcpm		
Limitation(s): The pre-generated polynomials were generated using AIC, B4C, and tungsten control rods for Watts Bar Unit 1. Materials with any other name will be ignored, and the results may not be improved as much for reactors other than Watts Bar Unit 1		
Description: This card toggles the use of volume weighting for control rods to minimize the effect of control rod cusping on the calculated results. Rod cusping is a calculational effect that occurs when a control rod is partially inserted into a calculational plane. This causes an artificial reduction in the local flux, which in turn causes an error in the calculated eigenvalue and global power distribution. Enabling this rod treatment card will correct for these effects. The polynomial option uses pre-generated polynomials to reduce the volume fraction of the control rod material during the homogenization step, providing better solutions near the tip of the control rod. The 1D collision probabilities method (1dcpm) is used to generate radial shape functions for rodded and unrodded regions, and then these shape functions are used to flux-volume homogenize the cross sections for the MOC calculations		
Notes: This card only has an effect when used in a 3D calculation (i.e., a calculation with axial planes). Options other than none and polynomial require that one of subplane_max, subplane_target, or num_subplanes be used as well. All options requiring subplane to be enabled are considered experimental		

ppm_method ppm_method

ppm_method	Character String	Optional
Units: N/A		
Applicable Value(s): 2 (default), 1		
Limitation(s): None		
Description: This card is used to specify which method should be used for computing soluble boron in the critical boron search. The options are as follows: • 1: this is the method suggested by nuclear vendors that just adds boron to water and does not conserve moderator density • 2: this is the original MPACT method that conserves moderator density		
Notes: None		

checkpoint_mode checkpoint_mode

checkpoint_mode	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): I (default), T, F, R, W, RW		
Limitation(s): File system permissions must be configured such that MPACT can interact with files as needed		
Description: This card is used to control whether the calculation is restarted from a checkpoint file. The input option descriptions are: • I: specifies that a checkpoint file may be written through a user interrupt • T: specifies that the case will be started from a checkpoint file • F: disables initialization of the checkpoint file • R: same as T • W: specifies that a checkpoint file is to be written • RW: same as T and R, but after the checkpoint file is read, it can be overwritten during the calculation The user can send the interrupt signal to MPACT after execution has begun by creating a file named "MPACT_CHECKPOINT_FILE" in the simulation's working directory. The existence of this file causes a checkpoint file to be written after every outer iteration. Likewise, the removal of "MPACT_CHECKPOINT_FILE" disables the writing of a checkpoint file. See the checkpoint_file card regarding checkpoint file naming		
Notes: None		

checkpoint_file checkpoint_file

checkpoint_file	Free-form Character String	Optional
Units: N/A		
Applicable Value(s): <CASEID>.mcp (default)		
Limitation(s): The filename must be specified with characters valid for use on the computer system being used for execution. As a general practice, one should avoid the use of special characters. Similarly, one must not specify a name that conflicts with other files that are (or will be) created within the MPACT directory		

continued on next page...

`checkpoint_file`, continued...

Description: If a checkpoint file will be used, the name of this file can be chosen and specified by the user
Notes: There is no strict limit on how many characters can be used to to specify the filename; however, good judgment should be used to keep the filename a reasonable length

rst_compress rst_compress

`rst_compress`	Free-form Character String	Optional
Units: N/A		
Applicable Value(s): 5 (default), none, 0 through 9		
Limitation(s): This only affects the WRITING of the restart file. `restart_read` cases and `restart_shuffle` cases are not affected		
Description: "None" means the HDF5 Filter for gzip compression is NOT used when writing the restart file. The numeric value indicates the level of compression to use in gzip. The higher the number, the more aggressive the compression, and the more resources used. See documentation of gzip for information		
Notes: The primary reason for this option is to disable compression, because on some platforms, decompression by HDF5 while reading may lead to an allocation error in HDF5 due to heap fragmentation. See documentation on the h5repack utility installed with the HDF5 library for removing compression after the file is written (e.g., `h5repack -f NONE <old_file> <new_file>`)		

vis_edits vis_edits

`vis_edits`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): core (default), none, fsr		
Limitation(s): None		
Description: This card is used to specify the type of visualization outputs (edits). The visualization outputs are created in the form of the VTK legacy file format which is suitable for use with VisIt (`https://wci.llnl.gov/simulation/computer-codes/visit/`) or other suitable programs capable of reading the format. These options are described as follows: 1. `core`: will print pin level edits of power for the full-core 2. `none`: will not print any visualization files 3. `fsr`: will print all available edits in the code on a flat source region-basis which includes material boundaries, mesh identification indices, and group-wise scalar flux		
Notes: The FSR edits will be very large and may require considerable time to generate the visualization files		

rr_edits rr_edits

`rr_edits`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): none (default), hdf5, out, both		
Limitation(s): None		

continued on next page...

`rr_edits`, continued...

Description: This card is used to specify the type of reaction rate outputs (edits). The reaction rate of an isotope is currently smeared over the problem domain when being printed to the output file, but the HDF5 file contains full information of reaction rates in geometry mesh. These options are described as follows: 1. `none`: will not generate reaction rate edits 2. `hdf5`: will generate reaction rate edits in HDF5 3. `out`: will print reaction rate in the output file 4. `both`: will do both `hdf5` and `out`
Notes: The reaction rate edits could be slow and memory-consuming for a large problem

rr_edits_opt rr_edits_opt

`rr_edits_opt`	Array of Pre-defined Format Strings	Optional
Units: N/A		
Applicable Value(s): none (default), `isotope_reaction` (absorption, fission, nu*fission, inscatter, outscatter, selfscatter)		
Limitation(s): This card can only be used if `rr_edits` is turned on		
Description: This card is used to specify the reaction rate edits for user-specified isotopes and reactions. The `isotope` is in a format of xx-AAA, (e.g., U-235 and Pu-239). The available reaction types are `absorption`, `fission`, `nu*fission`, `inscatter`, `outscatter` and `selfscatter`		
Notes: Select the important isotopes and reactions for edits can reduce the computing time and memory requirements for a large problem		

xe135m_opt xe135m_opt

`xe135m_opt`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): ignore (default), combine, explicit		
Limitation(s): None		
Description: This card is used to specify the treatment of ^{135m}Xe. By default, MPACT ignores ^{135m}Xe when performing transport calculation, although depletion solver may consider it. When combining ^{135m}Xe into ^{135}Xe, cross sections of the two isotopes are assumed to be the same. Explicit treatment can be enabled only for the latest MG library that has ^{135m}Xe data based on TENDL data. These options are described as follows: 1. `ignore`: will ignore ^{135m}Xe in transport calculation 2. `combine`: will combine ^{135m}Xe into ^{135}Xe in transport calculation 3. `explicit`: will treat ^{135m}Xe explicitly as other isotopes in transport calculation		
Notes: None		

azimuthal_xs azimuthal_xs

`azimuthal_xs`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): none (default), full, fuel, gad		

continued on next page...

azimuthal_xs, continued...

Limitation(s): None
Description: This card is used to specify the azimuthally dependent cross section region option. By default, MPACT will generate cross sections for radial rings only to minimize computational resource requirements. For BWR applications, or other special cases, increased accuracy may be needed by treating cross sections radially and azimuthally in the fuel pellet. These options are described as follows: 1. `none`: original cross section generation scheme 2. `full`: use azimuthal XS for all pin cell types 3. `fuel`: use azimuthal XS for fuel pin cell types only 4. `gad`: use azimuthal XS for gadolinium pin cell types only
Notes: Several options are provided to fine tune the accuracy vs. computational resources. The `full` option causes the largest increase in run time and memory usage, followed by `fuel` and `gad`. The depletion calculation will be performed for specified azimuthal regions as well

explicit_erg_deposit explicit_erg_deposit

`explicit_erg_deposit`	Boolean	Optional
Units: N/A		
Applicable Value(s): `false` (default), `true`		
Limitation(s): None		
Description: This card is used to specify whether the explicit energy deposition is used. Explicit energy deposition will compute the energy deposited in all regions from neutron fission, capture, and slowing-down		
Notes: Presently, the capture kappa data are only available in simplified AMPX library. MPACT library uses the hard-coded values. Other libraries do not support this card		

nodal_edits nodal_edits

`nodal_edits`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): false (default), nem, sanm-2, sanm-4, senm-2, senm-4		
Limitation(s): None		
Description: This card is used to enable or disable MPACT's nodal cross section capability. If enabled, node-averaged cross sections, flux moments, kinetics data, TH data, discontinuity factors, and other information will be written to each block of the HDF5 file. The different options will generate ADFs using different kernels. The `true` option is a deprecated option that is still supported for backwards compatibility and is the same as `senm-4`. These options are described as follows: 1. `nem`: enables MPACT nodal cross section edits using a quartic NEM kernel to generate ADFs 2. `sanm-2`: enables MPACT nodal cross section edits using a quadratic SANM kernel to generate ADFs 3. `sanm-4`: enables MPACT nodal cross section edits using a quartic SANM kernel to generate ADFs 4. `senm-2`: enables MPACT nodal cross section edits using a quadratic SENM kernel to generate ADFs 5. `senm-4`: enables MPACT nodal cross section edits using a quartic SENM kernel to generate ADFs 6. `false`: disables MPACT nodal cross section edits		

continued on next page...

`nodal_edits`, continued...

Notes: The SANM input options do the same thing as the SENM options

nodal_edits_energy_cutoff nodal_edits_energy_cutoff

`nodal_edits_energy_cutoff`	Float	Optional
Units: eV (default)		
Applicable Value(s): Any energy greater than 0.0 that is also an energy boundary in the transport library used for the calculation		
Limitation(s): None		
Description: This card is used to set the energy cut-off between the two groups when generating nodal data. The default cut-off is the energy between the last group with no up-scatter and the first group with up-scatter. This value is library-dependent and automatically determined during the calculation. The user may specify any of the energy group boundaries defined by the transport library as an input to this card		
Notes: None		

nodal_data_filename nodal_data_filename

`nodal_data_filename`	Free-Form Character String, Max. Length = 200	Optional
Units: N/A		
Applicable Value(s): N/A (default), Filename of any valid HDF5 file with user defined nodal data		
Limitation(s): This card must be present if using the Nodal transport method, and nodal data must be provided for every state		
Description: This card is used to indicate the name of the file containing the nodal data for each state		
Notes: The format of the HDF5 file must follow the same format as the HDF5 output nodal edits. The head dataset of the file must contain STATE datasets following the *STATE_####* nomenclature, which are populated with `NODAL_XS` datasets. `NODAL_XS` must include *ADF*, *CHI*, *KXSF*, *NXSF*, *XSF*, *XSRM*, *XSS*, and *XSTR*. These nodal datasets must have the same shapes as their corresponding output counterparts		

nodal_edits_adapt_adf nodal_edits_adapt_adf

`nodal_edits_adapt_adf`	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card is used to enable or disable MPACT's adaptive ADF calculations. When enabled, MPACT will adjust the outgoing current on vacuum boundaries until the ADF is equal to 1.0. The removal cross section will then be modified to preserve neutrons, and the diffusion cross section will be modified to be consistent with the removal cross section. These options are described as follows: 1. `true`: enables MPACT adaptive ADF calculations 2. `false`: disables MPACT adaptive ADF calculations		
Notes: Has no effect if `nodal_edits` is set to `false`		

nodal_edits_collapse_axial_reflectors nodal_edits_collapse_axial_reflectors

nodal_edits_collapse_axial_reflectors	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card is used to enable or disable collapsing the axial reflector nodal data. When enabled, each reflector will be collapsed into a single plane, regardless of how many planes are in the reflector. These options are described as follows: 1. `true`: collapses each axial reflector into a single plane 2. `false`: treats each axial reflector plane separately		
Notes: Has no effect if `nodal_edits` is set to `false`		

native_excore_detector native_excore_detector

native_excore_detector	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): `false` (default), `true`		
Limitation(s): None		
Description: This card is used to specify whether to perform the excore detector edits using the native simplified MPACT solver		
Notes: None		

grid_treatment grid_treatment

grid_treatment	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): `homogenize` (default), `equal_mass`, `equal_thickness`		
Limitation(s): For grids with large masses that fall within axially narrow lattices, there is a possibility that the grid will intersect one or more pins for the `equal_thickness` and `equal_mass` options. If this occurs, then MPACT will raise an error, and the user will be required to change the axial meshing options, change the geometry of the lattice, or simply use the `homogenize` option for the `grid_treatment` card. These options are described as follows: • `homogenize`: will take the mass specified in the grid card, calculate the moderator volume of the lattice where the grid is located, and use the two values to compute the density of the material. This option applies the grid material uniformly throughout the lattice. • `equal_thickness`: uses the grid mass and the corresponding grid material density to compute the total grid volume for that lattice. The volume is then used to determine what the grid thickness would be within each pin cell and is modeled as an additional rectangular mesh around the perimeter of each pin cell in the lattice. • `equal_mass`: similar to the `equal_thickness` option, except that the thickness of the grid in each pin cell is changed throughout the lattice so that every pin cell contains the same grid material mass.		
Description: This card is used to indicate the method of applying the grid structure in a lattice on the mesh		
Notes: None		

axial_buckling axial_buckling

axial_buckling	Float	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: Value used for critical buckling calculations		
Notes: None		

uniform_crud uniform_crud

uniform_crud	Floating-Point Real Numbers	Optional
Units: microns, mg/cm^2, mg/cm^2 (default)		
Applicable Value(s): 0.0, 0.0, 0.0 (default)		
Limitation(s): None		
Description: This card is used to define a uniform layer of CRUD on all fuel pins. The `thickness` is the CRUD thickness in microns, the `crud_mass` is the surface mass density of Ni Fe_2 O_4 in mg/cm^2, and the `boron_mass` is the surface mass density of Li B_4 O_7 in mg/cm^2		
Notes: None		

crud_depletion flag crud_depfrac

flag	Boolean	Optional
Units: N/A		
Applicable Value(s): true, false		
Limitation(s): None		
Description: This card is used to enable or disable crud depletion		
Notes: None		

crud_depfrac	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: This card is used to specify the fraction of crud to be depleted		
Notes: None		

meshing_method meshing_method

meshing_method	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): `useraxialmesh` (axial_mesh card present), *or* `matbound` (axial_mesh card not present) (default), `nonfuel`, `all`		
Limitation(s): Must be set in conjunction with the `axial_edit_bounds` card in the `EDIT` block of the VERA input when the option is not `useraxialmesh`. These data are required to set up the axial mesh for every input option except the `useraxialmesh`, where it is separately specified		

continued on next page...

meshing_method, continued...

Description: This card specifies the type of axial meshing to be used. If this card is not present, then the method will default to useraxialmesh if the axial_mesh card is present, or it will default to matbound if the axial_mesh card is not present.

- useraxialmesh: requires the use of the axial mesh card, and no auto meshing is performed in this instance. This option will not use the values specified by the automesh_bounds because it does not perform any automeshing.
- matbound: calculates the axial mesh just at the axial material boundaries of the problem and uses the axial_edit_bounds as the mesh within the fuel regions. No further meshing is performed. This option will not use the values specified by the automesh_bounds because it does not perform any automeshing.
- nonfuel: will take the material boundaries and automesh the regions below and above the fuel. The minimum and maximum bounds (or default values) specified by the automesh_bounds will be used to determine the sizing.
- all: will take the material boundaries and automesh all regions. The minimum and maximum bounds (or default values) specified by the automesh_bounds will be used to determine the sizing. When using the all option, fuel regions will not be homogenized with non-fuel regions. Homogenization will only occur within those regions.

Notes: When using the useraxialmesh option, it is possible to specify a mesh that does not conform or align with the problem's geometry. Warnings will be printed to the log file stating that the mesh does not match the geometry boundaries, and those regions will be homogenized

automesh_bounds automesh_bounds

automesh_bounds	Array of Floating-Point Real Numbers, Length = 2	Optional
Units: cm (default)		
Applicable Value(s): 2.0 10.0 (when automeshing is enabled) (default), Positive real numbers greater than zero. The maximum value must be at least 1.0 greater than the minimum value		
Limitation(s): None		
Description: This card specifies the minimum and maximum desired axial mesh for the auto axial meshing. Any geometry or mesh region larger than the specified value will be divided into smaller mesh regions that have heights between the maximum and minimum values. Any geometry or mesh region smaller than the specified value will be homogenized and added to a neighboring mesh region until the value is above the minimum and below the maximum		

continued on next page...

automesh_bounds, continued...

Notes: The region where these values are applied is specified by the meshing_method card. This card is ignored when the useraxialmesh and matbound method is specified. It should also be noted that specifying min and max values that are close together will most likely result in more axial homogenization than may be desired by the user. This would mean that most of the material interfaces will be homogenized to some degree. Also, this routine in no way optimizes the axial meshing for a given problem. It is primarily designed to reduce user burden from specifying a typically troublesome input parameter. It is best suited for problems with a large number of planes that vary in thickness. It is also useful for setting a problem up if the user is unsure about the axial discretization. Using this card will save time spent on recalculating values whenever the axial mesh needs to be adjusted

axial_mesh axial_mesh

axial_mesh	Array of Floating-Point Real Numbers, Length = User Specified	Optional
Units: cm (default)		
Applicable Value(s): N/A (default), Array of positive real numbers		
Limitation(s): The sum of the values specified within this card must be equal to the total geometric height of the problem		
Description: This card is used to specify the axial mesh used in the 2D/1D simulation. The input is the thickness of each axial section the user wishes to model. This card is optional if the meshing_method card specifies an option other than useraxialmesh. If the meshing_method is useraxialmesh, then it is required		
Notes: If the array of axial meshes sums to less than the problem height, the geometry at the top will be truncated. If it sums to more than the problem height, the top geometry will be extended all the way to the upper mesh height. Therefore, it is very important to make sure the axial mesh is specified in accordance with the geometry		

inter_assembly_gapmeshnum wide_gap_normal wide_gap_parallel narrow_gap_normal narrow_gap_parallel inner_gap

wide_gap_normal	Integer	Optional
Units: N/A		
Applicable Value(s): 3 (default), values must be on the interval [1,10]		
Limitation(s): Only applicable to BWR cores		
Description: This card defines the number of MOC fine source mesh regions in a pin cell in the wide gap along the direction normal to the channel box		
Notes: The inter_assembly_gapmeshnum card is optional, but this parameter is required on this card		

wide_gap_parallel	Integer	Optional
Units: N/A		

continued on next page...

wide_gap_parallel, continued...

Applicable Value(s): 3 (default), values must be on the interval [1,10]
Limitation(s): Only applicable to BWR cores
Description: This card defines the number of MOC fine source mesh regions in a pin cell in the wide gap along the direction parallel to the channel box
Notes: The inter_assembly_gapmeshnum card is optional, but this parameter is required on this card

narrow_gap_normal	Integer	Optional
Units: N/A		
Applicable Value(s): 3 (default), values must be on the interval [1,10]		
Limitation(s): Only applicable to BWR cores		
Description: This card defines the number of MOC fine source mesh regions in a pin cell in the narrow gap along the direction normal to the channel box		
Notes: The inter_assembly_gapmeshnum card is optional, but this parameter is required on this card		

narrow_gap_parallel	Integer	Optional
Units: N/A		
Applicable Value(s): 3 (default), values must be on the interval [1,10]		
Limitation(s): Only applicable to BWR cores		
Description: This card defines the number of MOC fine source mesh regions in a pin cell in the narrow gap along the direction parallel to the channel box		
Notes: The inter_assembly_gapmeshnum card is optional, but this parameter is required on this card		

inner_gap	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), values must be on the interval [1,10]		
Limitation(s): Only applicable to BWR cores		
Description: This card defines the number of MOC fine source mesh regions in a pin cell in the inner gap of the channel box and fuel pins parallel to channel box		
Notes: The inter_assembly_gapmeshnum card is optional, but this parameter is required on this card. The number of inner gap and channel box mesh divisions normal to the channel box uses the corresponding values of the wide_gap_normal and narrow_gap_normal parameters		

control_blade_meshnum sheath_num CR_rodlet_num

sheath_num	Integer	Optional
Units: N/A		
Applicable Value(s): 3 (default), values must be on the interval [1,10]		
Limitation(s): Only applicable to BWR cores		
Description: This card defines the number of MOC fine source regions along the length of the sheath of the control blade in a pin cell		
Notes: The control_blade_meshnum card is optional, but this parameter is required		

CR_rodlet_num	Integer	Optional
Units: N/A		
Applicable Value(s): 2 (default), values must be on the interval [1,10]		
Limitation(s): Only applicable to BWR cores		
Description: This card defines the number of MOC fine source radial mesh in the rodlets of the control blade		
Notes: The control_blade_meshnum card is optional, but this parameter is required		

detector_meshnum DT_gap_num DT_rodlet_num

DT_gap_num	Integer	Optional
Units: N/A		
Applicable Value(s): 2 (default), values must be on the interval [1,10]		
Limitation(s): Only applicable to detectors in BWR cores		
Description: This card defines the gap mesh number of the detector region		
Notes: The detector_meshnum card is optional, but this parameter is required		

DT_rodlet_num	Integer	Optional
Units: N/A		
Applicable Value(s): 2 (default), values must be on the interval [1,10]		
Limitation(s): Only applicable to detectors in BWR cores		
Description: This card defines the rodlet mesh number of the detector		
Notes: The detector_meshnum card is optional, but this parameter is required		

pin_cell_mod_mesh pin_cell_mod_mesh

pin_cell_mod_mesh	Array of mixed types int and string, Length = 2	Optional
Units: N/A		
Applicable Value(s): num_rings = 1 and pin_cell_type = fuel (default), > 0 nonfuel, > 0 both		
Limitation(s): This option does not work with explicit grid spacers. To use with grid spacers, set the grid_treatment option to homogenize		
Description: This card is used to specify the MOC flat source region mesh in the moderator outside the defined cylindrical geometry in specified pin cells. The radius of the outermost moderator ring is fixed at 0.95*sqrt(2)/2*pitch. This gives more refined meshing in the pin cell corners, which improves the accuracy of calculations at room temperature. • num_rings: positive integers. Practically less than 10. • pin_cell_type: "fuel," "nonfuel," "both."		
Notes: When this card is not specified, the following value is used for the 1 default moderator radius: max_radii = 0.75*(pitch*0.5 - r_last)+r_last. When this card is specified, that value changes to max_radii = 0.95*(0.5*pitch*sqrt(2)), which is equal to 95% of the distance from the pin cell's center to the corner. The default moderator radius is also applied to pin cells which do not match the pin_cell_type when the card is used. So, for example, if pin_cell_type is set to "nonfuel," then the fuel pins would still use the default moderator radius		

rad_fuel_mesh rad_fuel_mesh

rad_fuel_mesh	Float	Optional
Units: N/A		
Applicable Value(s): > 0.0 or < 1.0		
Limitation(s): Size must match radial divisions specified for fuel on the `mesh fuel` card—not applicable to annular fuel		
Description: Fractional radii of MOC source regions in fuel		
Notes: None		

crud_mesh crud_mesh

crud_mesh	One Floating-Point Real and One Integer	Optional
Units: microns (default)		
Applicable Value(s): N/A (default)		
Limitation(s): None		
Description: This card is used to specify the radial mesh that is added for each cell to account for CRUD build-up on the surfaces of the fuel pins. The options are positive real numbers for `max_rad` and integers greater than 0 for `num_rad`. The `max_rad` is the maximum thickness of the outermost CRUD region in microns and `num_rad` is the number of radial subdivisions in the CRUD region		
Notes: None		

quad_type quad_type

quad_type	Fixed Character String	Required
Units: N/A		
Applicable Value(s): None (default), CHEBYSHEV-CHEBYSHEV (Product), CHEBYSHEV-GAUSS (Product), CHEBYSHEV-BICKLEY (Product), CHEBYSHEV-YAMAMOTO (Product), QUADRUPLE-RANGE (Product), LEVEL-SYMMETRIC (Base)		
Limitation(s): None		
Description: This card is used to specify the name of the angular quadrature to use when determining the angles at which the rays are traced throughout the problem.		
Notes: None		

Quadrature Name	Type	Order	Order Θ
`CHEBYSHEV-CHEBYSHEV`	Product	integers > 0	integers > 0
`CHEBYSHEV-GAUSS`	Product	integers > 0	integers > 0
`CHEBYSHEV-BICKLEY`	Product	integers > 0	1, 2, 3, or 4
`CHEBYSHEV-YAMAMOTO`	Product	integers > 0	1, 2, or 3
`LEVEL-SYMMETRIC`	General	even integers in [2,16]	N/A
`QUADRUPLE-RANGE`	Product	integers in [1,37]	integers in [1,18]

gamma_quad_type gamma_quad_type

gamma_quad_type	Fixed Character String	Required

continued on next page...

gamma_quad_type, continued...

Units: N/A

Applicable Value(s): None (default), CHEBYSHEV-CHEBYSHEV (Product), CHEBYSHEV-GAUSS (Product), CHEBYSHEV-BICKLEY (Product), CHEBYSHEV-YAMAMOTO (Product), QUADRUPLE-RANGE (Product), LEVEL-SYMMETRIC (Base)

Limitation(s): None

Description: This card is used to specify the name of the angular quadrature to use when determining the angles at which the rays are traced throughout the problem for gamma transport.

Quadrature Name	Type	Order	Order Θ
CHEBYSHEV-CHEBYSHEV	Product	integers > 0	integers > 0
CHEBYSHEV-GAUSS	Product	integers > 0	integers > 0
CHEBYSHEV-BICKLEY	Product	integers > 0	1, 2, 3, or 4
CHEBYSHEV-YAMAMOTO	Product	integers > 0	1, 2, or 3
LEVEL-SYMMETRIC	General	even integers in [2,16]	N/A
QUADRUPLE-RANGE	Product	integers in [1,37]	integers in [1,18]

Notes: None

shield_quad_type shield_quad_type

shield_quad_type	Fixed Character String	Required

Units: N/A

Applicable Value(s): None (default), CHEBYSHEV-CHEBYSHEV (Product), CHEBYSHEV-GAUSS (Product), CHEBYSHEV-BICKLEY (Product), CHEBYSHEV-YAMAMOTO (Product), QUADRUPLE-RANGE (Product), LEVEL-SYMMETRIC (Base)

Limitation(s): None

Description: This card is used to specify the name of the angular quadrature to use when determining the angles at which the rays are traced throughout the problem for the shielding calculation.

Quadrature Name	Type	Order	Order Θ
CHEBYSHEV-CHEBYSHEV	Product	integers > 0	integers > 0
CHEBYSHEV-GAUSS	Product	integers > 0	integers > 0
CHEBYSHEV-BICKLEY	Product	integers > 0	1, 2, 3, or 4
CHEBYSHEV-YAMAMOTO	Product	integers > 0	1, 2, or 3
LEVEL-SYMMETRIC	General	even integers in [2,16]	N/A
QUADRUPLE-RANGE	Product	integers in [1,37]	integers in [1,18]

Notes: None

azimuthals_octant azimuthals_octant

azimuthals_octant	Integer	Required

Units: N/A

Applicable Value(s): None (default), Column Order in the above table

Limitation(s): None

Description: This card is used to specify the number of azimuthal angles per octant and corresponds to the "Order" column in the table in quad_type card

continued on next page...

azimuthals_octant, continued...

Notes: None

gamma_azimuthals_octant gamma_azimuthals_octant

gamma_azimuthals_octant	Integer	Required
Units: N/A		
Applicable Value(s): None (default), Column Order in the above table		
Limitation(s): None		
Description: This card is used to specify the number of azimuthal angles per octant for gamma transport and corresponds to the "Order" column in the table in quad_type card		
Notes: None		

polars_octant polars_octant

polars_octant	Integer	Required
Units: N/A		
Applicable Value(s): None (default), Column Order Θ in the above table		
Limitation(s): None		
Description: This card is used to specify the number of polar angles per octant and corresponds to the "Order Θ" column in the quadrature table specified in quad_type card. Note the number of polar angles may be limited by the quadrature type used. Also, any non-product quadrature types will not use this input card (i.e., in the only applicable case LEVEL-SYMMETRIC)		
Notes: None		

gamma_polars_octant gamma_polars_octant

gamma_polars_octant	Integer	Required
Units: N/A		
Applicable Value(s): None (default), Column Order Θ in the above table		
Limitation(s): None		
Description: This card is used to specify the number of polar angles per octant for gamma transport and corresponds to the "Order Θ" column in the quadrature table specified in quad_type card. Note the number of polar angles may be limited by the quadrature type used. Also, any non-product quadrature types will not use this input card (i.e., in the only applicable case LEVEL-SYMMETRIC)		
Notes: None		

shield_azimuthals_octant shield_azimuthals_octant

shield_azimuthals_octant	Integer	Required
Units: N/A		
Applicable Value(s): None (default), Column Order in the above table		
Limitation(s): None		
Description: This card is used to specify the number of azimuthal angles per octant for the shielding sweeper and corresponds to the "Order" column in the table in quad_type card		
Notes: None		

shield_polars_octant shield_polars_octant

shield_polars_octant	Integer	Required
Units: N/A		
Applicable Value(s): None (default), Column Order Θ in the above table		
Limitation(s): None		
Description: This card is used to specify the number of polar angles per octant for the shielding calculation and corresponds to the "Order Θ" column in the quadrature table specified in quad_type card. Note the number of polar angles may be limited by the quadrature type used. Also, any non-product quadrature types will not use this input card (i.e., in the only applicable case LEVEL-SYMMETRIC)		
Notes: None		

xs_type xs_type

xs_type	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): NONE (default), ORNL, SIMPLIFIED_AMPX, or HELIOS		
Limitation(s): None		
Description: This card is used to specify the type of cross section file to use		
Notes: None		

xs_filename xs_filename

xs_filename	Free-Form Character String, Max. Length = 200	Required
Units: N/A		
Applicable Value(s): No default value (default), filename of a supported cross section library		
Limitation(s): None		
Description: This card is used to specify the name of the cross section file to use		
Notes: None		

ce_filename ce_filename

ce_filename	Free-Form Character String, Max. Length = 200	Optional
Units: N/A		
Applicable Value(s): No default value (default), filename of an indexing file for CE library		
Limitation(s): None		
Description: This card is used to specify the name of the indexing file of continuous-energy cross section library to be used when quasi_1D is toggled on		
Notes: None		

shield_method shield_method

shield_method	Fixed Character String	Optional
Units: N/A		

continued on next page...

shield_method, continued...

Applicable Value(s): subgroup (default), essm, subgroup-cell, essm-cell, sdessm-cell
Limitation(s): The xs_shielder card must be enabled (default) in order to enable this card; otherwise, unshielded cross section (infinite-dilute) will be used
Description: This card is used to specify the method used to shield the cross sections. The values are described below: • subgroup: uses the whole-core subgroup self-shielding method to calculate equivalence cross sections • essm: uses the whole-core embedded self-shielding method (ESSM) to calculate equivalence cross sections • subgroup-cell: uses a cell-based subgroup self shielding method to calculate equivalence cross sections • essm-cell: uses a cell-based ESSM to calculate equivalence cross sections • sdessm-cell: uses a cell-based spatially dependent ESSM to calculate equivalence cross sections
Notes: The subgroup method has a few advantages over ESSM, such as a better representation of distributed self-shielding within the fuel and the resonance category treatment (resonance isotopes are grouped into categories). Therefore, the subgroup method is an option with better accuracy than in the current version. The cell-based shelf-shielding methods still use a one-group whole-core subgroup calculation to treat spacer grids, cladding, and other similar materials. The cell-based method is then applied to the fuel rods, control rods, and other important resonance materials that have multiple subgroup categories and levels

shield_nbatch shield_nbatch

shield_nbatch	Integer	Optional
Units: N/A		
Applicable Value(s): 5 (default)		
Limitation(s): None		
Description: This card is used to specify the number of batches used to divide the pseudogroups of the MG shielding sweeper		
Notes: None		

xs_shielder xs_shielder

xs_shielder	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): true (default), false, t, f		
Limitation(s): None		
Description: This card is used to specify whether to shield the cross sections or not: true–enabled, false–disabled		
Notes: If shielder is disabled, the infinite-dilute cross sections for the resonance energy groups are used		

spatial_essm spatial_essm

spatial_essm	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): `false` (default), `true`, `t`, `f`		
Limitation(s): None		
Description: This card is used to specify whether to perform the spatial essm correction for self-shielding calculation. Currently, this option can only be toggled on with `essm`		
Notes: None		

quasi_1D quasi_1D

quasi_1D	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): `false` (default), `true`, `t`, `f`		
Limitation(s): None		
Description: This card is used to specify whether to perform the quasi-1D slowing-down correction for self-shielding calculation. Currently, this option can only be toggled on with `essm`		
Notes: None		

res_up_scatter res_up_scatter

res_up_scatter	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): `false` (default), `true`		
Limitation(s): None		
Description: This card is used to specify whether to use the resonance data that incorporates the epithermal upscattering model. Currently, this option is only supported for ORNL library from version 4 onward		
Notes: None		

subgr_temp_average subgr_temp_average

subgr_temp_average	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): `plane` (default), `pin`		
Limitation(s): None		
Description: This card is used to specify the fuel temperature averaging scheme for the subgroup temperature correction		
Notes: The averaged temperature is not directly used for cross section calculation. It is used to correct the non-uniform temperature effect in calculating the equivalence cross sections for subgroup method		

dep_filename dep_filename

dep_filename	Free-Form Character String, Max. Length = 200	Required
Units: N/A		
Applicable Value(s): No default value (default), filename of a supported cross section library		

continued on next page...

dep_filename, continued...

Limitation(s): The format of this file should be consistent with the standard MPACT depletion library file MPACT.dpl
Description: This card is used to specify the depletion file to use, which provides all the data required, in addition to the data in the transport library for depletion calculation
Notes: None

mats_file mats_file

mats_file	Free-Form Character String, Max. Length = 200	Optional
Units: N/A		
Applicable Value(s): No default value (default), filename of a HDF5 material database file		
Limitation(s): None		
Description: This card is used to specify the name of the HDF5 material database file. This file is used to overwrite the isotopic and weight fraction values for default VERA material		
Notes: Marked for deprecation—do not use!		

mod_mat mod_mat

mod_mat	Free-Form Character String, Max. Length = 200	Optional
Units: N/A		
Applicable Value(s): mod (default), any user-defined name of the moderator material		
Limitation(s): None		
Description: This card is used to rename the moderator material		
Notes: None		

subgroup_set subgroup_set

subgroup_set	Integer	Optional
Units: N/A		
Applicable Value(s): 4 (default), integers 1 through 9		
Limitation(s): The shield_method must be set to subgroup or subgroup-cell. Other shielding options ignore this input		
Description: This card is used to specify the subgroup set		
Notes: In most cases, 4 (the default) should be used. This option provides a good balance of accuracy and computing time. In general, the numbering is from 1 to 9, with 1 being the simplest set (fast), and 9 being the most explicit set (slow)		

cat_onegroup cat_onegroup

cat_onegroup	Array of Integers, Length = User Specified	Optional
Units: N/A		
Applicable Value(s): 3(if subgroup_set = 4) (default), any integer number		

continued on next page...

cat_onegroup, continued...

Limitation(s): The shield_method must be set to subgroup. ESSM ignores the cat_onegroup option
Description: This card is used to specify the categories that use one-group subgroup
Notes: The user can specify the categories that will use one-group subgroup treatment, which results in a fast, approximate subgroup calculation in that category. If subgroup_set = 4 (default), the default value of this option is 3 (clad category); otherwise, no default category will be assigned to one-group subgroup unless specified by the user. The user can also specify zero or a negative integer number to use MG-subgroup for all categories

shld_range shld_range

shld_range	Array of Integers, Length = 2	Optional
Units: N/A		
Applicable Value(s): 1,ng (default), between 1 and ng		
Limitation(s): Currently only simplified AMPX library supports this option		
Description: This card is used to specify the beginning and ending groups that resonance self-shielding calculation will be performed		
Notes: None		

k_tolerance k_tol

k_tol	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.0E-5 (default), >0.0		
Limitation(s): None		
Description: This card is used to specify the global tolerance on convergence of the eigenvalue		
Notes: None		

flux_tolerance flux_tolerance

flux_tolerance	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.0E-4 (default), >0.0		
Limitation(s): None		
Description: This card is used to specify the tolerance on the convergence of the 2-norm of the flux		
Notes: None		

gamma_flux_tolerance gamma_flux_tolerance

gamma_flux_tolerance	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.0E-4 (default), >0.0		
Limitation(s): None		
Description: This card is used to specify the tolerance on the convergence of the 2-norm of the gamma flux		
Notes: None		

num_outers num_outers

num_outers	Integer	Optional
Units: N/A		
Applicable Value(s): 500 (default), ≥1		
Limitation(s): None		
Description: This card is used to specify the maximum number of outer eigenvalue iterations. If the case is not converged to within the specified tolerances, this input value is compared to the current outer iteration value. If the current outer iteration value is equal to the input value, the program execution will exit saying that the maximum number of iterations has been reached		
Notes: None		

num_inners num_inners

num_inners	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), ≥1		
Limitation(s): None		
Description: This card is used to specify the number of inner 1-group transport sweeps done during group sweeping every outer iteration		
Notes: For 2D/1D problems, it is usually optimal for num_inners to be set to 1. However, numerical instability is frequently an issue. The instability presents as an inability to converge to the desired tolerance. The solution will stagnate to within some tolerance and oscillate around that value until the maximum number of outers are reached. In this case, it is advised to use additional inner sweeps for stabilization. If so,num_inners=2 or 3 (with up_scatter=1) is a typical value		

gamma_num_inners gamma_num_inners

gamma_num_inners	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), ≥1		
Limitation(s): None		
Description: This card is used to specify the number of inner 1-group transport sweeps for gamma transport performed during group sweeping for every outer iteration		
Notes: None		

up_scatter up_scatter

up_scatter	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), ≥0		
Limitation(s): None		
Description: This card is used to specify the number of upscattering iterations that occur during group sweeping, i.e., between fission source iterations		
Notes: Increasing up_scatter is one way to potentially remedy issues with numerical instability		

gamma_up_scatter gamma_up_scatter

gamma_up_scatter	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), ≥0		
Limitation(s): None		
Description: This card is used to specify the number of upscattering iterations that occur during group sweeping for gamma transport: that is, between fission source iterations		
Notes: Increasing `up_scatter` is one way to potentially remedy issues with numerical instability		

num_extsrc_itrs num_extsrc_itrs

num_extsrc_itrs	Integer	Optional
Units: N/A		
Applicable Value(s): num_outers (default), ≥1		
Limitation(s): None		
Description: This card is used to specify the number of outer iterations an external source strength iteration will perform before increasing the source strength. If the current outer iteration value is equal to it, the source strengths will be increased by the strength multiplication factor, and outer iterations will be started again from count zero. This will repeat until the source is at full strength, wherein the full `num_outers` value will be used for the full strength iterations		
Notes: None		

scattering scattering

scattering	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): TCP0 (default), P0, P1, P2, P3, P4, P5, Pn0, LTCP0, FLTCP0		
Limitation(s): None		
Description: This card is used to specify the scattering treatment to be used by the radial neutron transport calculations in MPACT. There are two primary categories: those which use the P0 sweeper, and those which use the Pn sweeper. The P0 sweeper options are described as follows: • `P0`: performs transport calculations using isotropic scattering with no transport correction • `TCP0`: performs transport calculations using isotropic scattering with transport-corrected cross sections—this is the default • `LTCP0`: performs the same transport calculations as TCP0, except that for energies above 1 MeV, the transport correction is limited to prevent negative self-scatter cross sections. This can hinder accuracy, but it can help stability in certain cases • `FLTCP0`: does the same thing as LTCP0, but for all energies The Pn sweeper options are described as follows: • `Pn0`: uses the same physics as TCP0, but with the Pn sweeper • `P1`: uses linearly anisotropic scattering for the transport calculations • `P2`: uses second-order anisotropic scattering for transport calculations • `P3`: uses third-order anisotropic scattering for transport calculations • `P4`: uses fourth-order anisotropic scattering for transport calculations • `P5`: uses fifth-order anisotropic scattering for transport calculations		

continued on next page...

`scattering`, continued...

Notes: None

gamma_scattering gamma_scattering

`gamma_scattering`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): TCP0 (default), P0, LTCP0, FLTCP0		
Limitation(s): None		
Description: This card is used to specify the scattering treatment to be used by the radial gamma transport calculations in MPACT. The options for gamma transport are described as follows: • `P0`: performs transport calculations using isotropic scattering with no transport correction • `TCP0`: performs transport calculations using isotropic scattering with transport-corrected cross sections—this is the default • `LTCP0`: performs the same transport calculations as TCP0, except that for energies above 1 MeV, the transport correction is limited to prevent negative self-scatter cross sections. This can hinder accuracy, but it can help stability in certain cases • `FLTCP0`: does the same thing as LTCP0, but for all energies		
Notes: None		

trim_Pn_moments trim_Pn_moments

`trim_Pn_moments`	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), true, false		
Limitation(s): None		
Description: This card is used to toggle the logic to trim unused scattering moments when using Pn scattering techniques		
Notes: None		

boundary_update boundary_update

`boundary_update`	string	Optional
Units: N/A		
Applicable Value(s): P0 (default), none, DP0, P1		
Limitation(s): None		
Description: This card is used to specify the CMFD boundary update method to accelerate convergence of problems using CMFD. The following options are available: • `NONE`: use no boundary update • `P0`: use CMFD scalar fluxes to scale transport angular fluxes (default) • `DP0`: use CMFD partial currents to scale transport angular fluxes • `P1`: use CMFD currents to scale transport angular fluxes		
Notes: The `DP0` and `P1` options are more complex and generally do no provide significant convergence improvement. The default option of `P0` is recommended		

depl_time_method depl_time_method

depl_time_method	Fixed Character String	Required
Units: N/A		
Applicable Value(s): p-c(predictor-corrector) (default), semip-c(semi-predictor-corrector), or postcorrector(semi-predictor-corrector-post-corrector)		
Limitation(s): None		
Description: This card is used to specify the time stepping method in depletion. The p-c method computes a predicted nuclide concentration based on the steady-state flux condition at the beginning of time step, which is then averaged with the corrected nuclide concentration based on the steady-state flux condition at the end of time step. Two steady-state eigenvalue calculations are performed for each depletion time step. The p-c method is a well demonstrated method, and it can be used for large time steps. The semip-c method simplifies the p-c method by skipping the second steady-state eigenvalue calculation, and thus it becomes more efficient in small time- step depletion calculation. The postcorrector method is identical to the semip-c method, except that the number densities used for the beginning of timestep steady-state eigenvalue calculation are "post-corrected" so that they more closely represent the averaged number densities of the full p-c method. This allows for accuracy comparable to the full p-c method while still skipping the second steady-state eigenvalue calculation		
Notes: The semip-c method can result in an inconsistency when restarting. However, the differences that arise from a semip-c restart are smaller in magnitude than the differences between semip-c and p-c. The inconsistency in the semip-c restart arises from an extra flux calculation that occurs on restart, so presumably the difference results in a more accurate solution		

gad_dep_method gad_dep_method

gad_dep_method	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): none (default), qgd2		
Limitation(s): None		
Description: This card is used to request a higher order treatment of gadolinium isotopes during burnup calculations. Currently, the only option is qgd2. When called with the qgd2 option, gadolinium isotopes are dealt with using the high order methodology described in D. Lee, J. Rhodes, and K. Smith. "Quadratic Depletion Method for Gadolinium Isotopes in CASMO-5," *Nuclear Science and Engineering* 174 (2013), pp. 79-86. If the none option is used, then no special treatment of gadolinium isotopes will be considered		
Notes: None		

depl_origen_solver depl_origen_solver

depl_origen_solver	Fixed Character String	Required
Units: N/A		
Applicable Value(s): cram(CRAM solver) (default), matrex(MATREX solver)		
Limitation(s): None		
Description: This card is used to specify the solver method used by ORIGEN when performing depletion calculations. The cram method is the Chebyshev Rational Approximation Method (CRAM). The matrex method is a hybrid matrix exponential / linear chain method and is the legacy ORIGEN solution method		

continued on next page...

depl_origen_solver, continued...

Notes: Compared to the matrex solver, cram has similar runtimes but is more accurate and robust on a larger range of problems. Unlike matrex, the length of a step does not significantly affect the accuracy of cram in the absence of substep power renormalization. Thus, it is recommended that cram be used for ORIGEN depletion solves

num_space num_space

num_space	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), Integer greater than 0 and less than the number of CPU cores		
Limitation(s): None		
Description: This card is used to specify the number of spatial decomposition regions used in a parallel execution step. this value can be: 1. a subset of the number of planes in the model, 2. the total number of planes, or 3. a product of all of the planes and any number of radial regions comprised of groups of quarter assemblies for PWRs or groups of assemblies for BWRs. The ability to decompose a problem by planes can be used with the DEFAULT partition method. Any partition that decomposes the problem radially requires the EXPLICITFILE partition method		
Notes: See description of card num_angle for explanation of using spatial and angular decomposition in conjunction		

num_angle num_angle

num_angle	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), Integer greater than 0 and less than the number of CPU cores		
Limitation(s): Specifying a value greater than 2*azimuthals_octant will cause an exception error		
Description: This input options specifies the number of parallel partitions used to decompose the problem based on the azimuthal angle (i.e., ray directions in the x-y plane). To get the 2D MOC solution for a single x-y plane, rays are traced through the domain in multiple azimuthal directions as specified by the user in the option azimuthals_octant. The user should note that the terms *octant* and *quadrant* are interchangeable in the context of azimuthal angles). The azimuthal angles are divided into num_angle groups, and each group is assigned to a parallel partition (i.e., process). If spatial decomposition is used in the same problem, then each spatial decomposition region is copied to num_angle partitions. Therefore, the total number of parallel partitions is num_angle*num_space		
Notes: The user is cautioned against using too many processes to decompose the problem. Due to the increase in inter-process communication with increased parallel decomposition, excessive parallelization will not yield speedup of the solution. The proper amount of parallelization must be determined on a case-by-case basis		

num_energy num_energy

num_energy	Integer	Optional
Units: N/A		
Applicable Value(s): 1		
Limitation(s): None		
Description: Energy decomposition is not yet supported. MPACT will only run with num_energy=1		
Notes: None		

num_threads num_threads

num_threads	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), Integer greater than 0 and less than the number of CPU cores		
Limitation(s): None		
Description: This card is used to specify the number of threads used in parallel execution. The number of threads specified are used only during the MOC transport sweep. For a given ray direction (i.e., angle), threads are used to sweep multiple rays in parallel		
Notes: It is recommended that num_angle*num_space*num_threads does not exceed the total number of physical CPU cores. MPACT will still run if the user exceeds this limit, but the parallel performance will be degraded		

par_method par_method

par_method	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): ASSEMBLY (default), GRAPH, ASSEMBLY, EXPLICITFILE, EXPLICITRADIAL, PS		
Limitation(s): The EXPLICITFILE option may be used only if the user has created a partition file. For a description of the partition file, see the input option par_file		
Description: This card is used to specify the method of parallel decomposition. • GRAPH: spatially decomposes the core using graph partitioning methods. This method is automated, more flexible, and generally provides better load-balance than the other options. • ASSEMBLY: the parallelization scheme for decomposing a problem spatially. The problem will be decomposed radially first, and if there are more processors, it will then attempt to parallelize the problem axially. This process is done automatically, and the user is only required to specify the number of spatial processors available in the num_space card described below. It is the recommended method for large problems. • EXPLICITFILE: for more advanced users who are running large problems, using the EXPLICITFILE option may enable the user to parallelize the problem more effectively. For a description of the EXPLICITFILE method, see the input option par_file. • EXPLICITRADIAL: same as EXPLICITFILE, except that the decomposition is provided for a single plane and is applied to all planes. • PS: same as ASSEMBLY.		
Notes: None		

par_file par_file

par_file	Free-Form Character String	Optional
Units: N/A		
Applicable Value(s): partition.txt (default)		
Limitation(s): No comments are allowed in the file		
Description: This card is used to specify the parallel decomposition file if EXPLICITFILE is used. This is an advanced feature that is not recommended for most users. The MPACT domain is divided into a regular grid of ray trace modules; the partition file allows the user to specify the spatial decomposition of the domain by listing the ray trace modules in each spatial partition via their (x,y,z) indices (this is explained more in the following paragraphs). The partition file also allows the user to decompose the MPACT domain radially, which is not possible with the DEFAULT partition method. The file structure itself has two header lines followed by the specification of the radial partition regions. The first line has 3 values: the first is the number of MPACT ray trace modules in the x direction, the second is the number of ray trace modules in the y direction, and the third is the number of axial planes in the model. The second line also has 3 values. The first two pertain specifically to how MPACT partitions ray trace modules in space, and these values should always be 0 and 1, respectively. The third value should be the number of radial partitions being subsequently specified. The following lines should describe all radial partition regions for the problem, including any regions that will be used with a jagged core. The input for each line is 6 integers. The first pair of integers is the starting and stopping module indices in the x direction, the second pair is the starting and stopping module indices in the y direction, and the last pair is for the z direction, but these integers are ignored currently, and all radial partitions are assumed to be the same for each axial plane. The coordinate system point of origin when specifying the starting and stopping indices is the lower left (southwest) corner of the module. When specifying the starting and stopping indices, it is important to note that these are not necessarily the assembly positions. Typically, in the case of modeling a full reactor, the ray trace modules represent a quarter of an assembly. In this case, the number of ray trace modules in a given direction will be about twice the number of assemblies in that direction		
Notes: If the core is jagged, additional attention is required to keep track of the actual number of processors being used by MPACT. Even though the nonexistent assemblies are "partitioned" in the explicit file, nothing there will be run. Therefore, the user cannot simply take the third value from the second line and multiply it by the third value from the first line to get the total number of spatial partitions for this case. In the example below, the third value in the second line must have the number of "jagged" partitions subtracted from it. In this case, the actual number of processors per plane becomes 49 - 8 = 41. That number can then be multiplied by the number of planes to get 2,378 processors, which should be input into the num_space card. Also, it may be unclear to the user how many planes will be created in MPACT before the case is run. The output file has a summary of the axial mesh information, including the total number of planes. If the case crashes when using the partition file, the user should check that the number of planes specified matches the value in the output file		

par_xdim par_xdim

par_xdim	Integer	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This card specifies the x dimension of the model when using the par_map option		

continued on next page...

par_xdim, continued...

Notes: None

par_ydim par_ydim

par_ydim	Integer	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This card specifies the y dimension of the model when using the par_map option		
Notes: None		

par_map par_map

par_map	2D Integer Map	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: If the EXPLICITRADIAL partition method is used and a file is not specified, then a par_map must be provided. This multi-line map should contain the indexes containing each module. These domains must be contiguous (all modules in a domain must neighbor at least one other module in the domain) and must have no concave boundaries		
Notes: None		

graph_part_method graph_part_method

graph_part_method	Array of Fixed Character Strings	Optional
Units: N/A		
Applicable Value(s): 'REB' (default), 'RSB', 'RIB'		
Limitation(s): Only applicable if par_method is GRAPH		
Description: This card is used to read the decomposition/partition algorithms to be used for spatial decomposition		
Notes: None		

graph_refn_method graph_refn_method

graph_refn_method	Array of Fixed Character Strings	Optional
Units: N/A		
Applicable Value(s): 'KL', 'SKL', 'None'		
Limitation(s): Only applicable if par_method is GRAPH		
Description: This card is used to read the communication refinement algorithms to be used during spatial decomposition		
Notes: Should be of size 1 or same size as GRAPH_PART_METHOD		

graph_cond graph_cond

graph_cond	Array of Integers	Optional
Units: number of modules (default)		
Applicable Value(s): > 0		
Limitation(s): Only applicable if par_method is GRAPH		
Description: This card reads inputs for smallest graph size (modules) for each decomposition method. This card should not be used by typical users		
Notes: Should be of size 1 less than GRAPH_PART_METHOD		

coupling_method coupling_method

coupling_method	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): simplified (if not configured with COBRA-TF) *or* ctf (if configured with COBRA-TF) (default), ctf_external, tiamat, user_defined, hybrid, none		
Limitation(s): The feedback card in the STATE block must be set to on for any of the TH coupling methods		
Description: This card is used to indicate which TH coupling method should be used. The simplified option uses MPACT's internal TH solver. The ctf option internally couples COBRA-TF to MPACT, and ctf_external couples MPACT and COBRA-TF through the lime interface. The user_defined option uses TH conditions defined in the HDF5 file specified by the user_defined_th_filename card in the MPACT block. The none option will use parameters from the STATE block: fuel temperatures will be constant and equal to tfuel, moderator temperatures will be constant and equal to tinlet, and moderator densities will be constant and equal to modden. The hybrid option will use the simplified option for the first several iterations to obtain an approximate solution before switching to ctf		
Notes: For either the ctf or ctf_external options, MPACT must be configured with COBRA-TF. The internal option may be used regardless of whether MPACT was configured with COBRA-TF or not		

shielder_th shielder_th

shielder_th	Integer, Floating-Point Real Number, Floating-Point Real Number	Optional
Units: {unitless, K, g/cm^3} (default)		
Applicable Value(s): 100, 5.0, 0.01 (default), > 0 > 0.0 and > 0.0		
Limitation(s): If the xs_shielder card is set to f or false, then this card does nothing since cross section shielding calculations will never be performed		

continued on next page...

shielder_th, continued...

Description: This card is used to control the number of cross section shielding calculations performed when using TH feedback. It sets a maximum number of iterations with shielding calculations, and it also sets parameters to stop the shielding calculations earlier if the TH feedback effects on temperature and moderator density are small enough. The first input is the maximum number of outer iterations for which MPACT will perform cross section shielding calculations following a TH update. The second input is the minimum change in temperature for which MPACT will perform cross section shielding calculations following a TH update. The third and final input is the minimum change in moderator density for which MPACT will perform cross section shielding calculations following a TH update
Notes: If multiple state points are performed in the calculation, then the counter for the `shield_max_outers` input is reset for each state point. If the `xs_shielder` card is not set to `f` or `false`, then shielding calculations will always be performed on the first iteration

outers_per_TH outers_per_TH

`outers_per_TH`	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), ≥ 0		
Limitation(s): None		
Description: This card is used to indicate how many outer iterations MPACT should perform before performing an additional TH update		
Notes: None		

init_from_STH_outers init_from_STH_outers

`init_from_STH_outers`	Integer	Optional
Units: N/A		
Applicable Value(s): 5 (BWR), or 3 (Non-BWR) (default), ≥ 0		
Limitation(s): None		
Description: If `coupling_method` is set to `hybrid`, then this card is used to determine how many simplified TH solves will be performed before switching to CTF		
Notes: If `coupling_method` is not set to `hybrid`, this card does nothing. Additionally, if the solution converges before reaching the number of iterations specified in this option, then the code will switch to CTF immediately for the subsequent iteration; this ensures that CTF is run at least once before the solution is considered converged		

average_ftemp average_ftemp

`average_ftemp`	Boolean	Optional
Units: N/A		
Applicable Value(s): `true` (default), `false`		
Limitation(s): None		

continued on next page...

average_ftemp, continued...

Description: If true, this card applies a volume-averaged fuel temperature to each fuel pin. If false, it applies a radially dependent fuel temperature to each fuel pin
Notes: None

radial_power_ctf_coupling radial_power_ctf_coupling

radial_power_ctf_coupling	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), true		
Limitation(s): None		
Description: If true, this card calculates the radial power Zernike coefficients to pass to CTF. If false, no coefficients are calculated		
Notes: None		

radial_burnup_ctf_coupling radial_burnup_ctf_coupling

radial_burnup_ctf_coupling	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), true		
Limitation(s): None		
Description: If true, this card calculates the radial burnup Zernike coefficients to pass to CTF. If false, no coefficients are calculated		
Notes: None		

radial_temp_ctf_coupling radial_temp_ctf_coupling

radial_temp_ctf_coupling	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), true		
Limitation(s): None		
Description: If true, this card will use the radial fuel temperature Zernike coefficients from CTF to set the fuel temps in MPACT. If false, the coefficients are not used and the volume-averaged fuel temp is used in all fuel rings		
Notes: None		

ctf_basename ctf_basename

ctf_basename	Free-Form Character String, Max. Length = 200	Optional
Units: N/A		
Applicable Value(s): deck (when COBRA-TF is run in serial) *or* pdeck (when COBRA-TF is run in parallel) (default), Any filename base for valid COBRA-TF input decks		
Limitation(s): Filename must have ".inp" extension		
Description: This card is used to indicate the "basename" of the CTF input files for CTF coupling. The "basename" is the section of the CTF input filename(s) without any extensions		

continued on next page...

`ctf_basename`, continued...

Notes: Absolute or relative paths to the file are both acceptable

sth_dhfrac sth_dhfrac

`sth_dhfrac`	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 0.02 (default), 0.0–1.0		
Limitation(s): It is ignored if feedback is off or if coupling with COBRA-TF is being used		
Description: This card is used to set the fraction of the power which is directly deposited in the moderator in internal TH calculations		
Notes: None		

sth_hgap sth_hgap

`sth_hgap`	Floating-Point Real Number	Optional
Units: W/m^2· K (default)		
Applicable Value(s): 4500.0 (default), > 0.0		
Limitation(s): It is ignored if feedback is off or if coupling with COBRA-TF is being used		
Description: This card is used to set the gap conductance value for internal TH calculations		
Notes: Typical values range from 1,000 (very low) to 10,000 (very high)		

sth_channeltype sth_channeltype

`sth_channeltype`	Character String	Optional
Units: N/A		
Applicable Value(s): assem, node, chan		
Limitation(s): None		
Description: This card is used to set the size of the region over which average moderator conditions will be applied. Acceptable values are assembly, node (quarter assembly), or pin (flow channel between four fuel pins)		
Notes: None		

sth_avgpin sth_avgpin

`sth_avgpin`	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card is used to determine whether an average pin is used for each region or if fuel conduction calculations are done for each pin uniquely. If true, a representative pin will be used. If `sth_channeltype` is set to pin, this card is ignored		
Notes: None		

temptable_shape temptable_shape

temptable_shape	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Logical to interpolate shape onto fuel temperature table value		
Notes: None		

temptable_boundary temptable_boundary

temptable_boundary	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): N/A (default), Boundary for applying the temperature tables		
Limitation(s): If the card is present, then temperature tables in the named file will be used to calculate fuel temperatures instead of the internal conduction solvers or COBRA-TF. If this card is not present, then internal or COBRA-TF solvers are used		
Description: This card is used to define the boundary from which the table was generated and which should be used to apply the table		
Notes: Temperature tables contain fuel temperature values as functions of power and burnup. When depleting, the thermal properties of the fuel change significantly. Internal TH and COBRA-TF do not know how these properties change when depleting, so temperature tables can be used to more accurately perform TH calculations during depletion simulations using tabulated data rather than fuel conduction solvers		

temptable_qprime temptable_qprime

temptable_qprime	Floating-Point Real Numbers	Optional
Units: N/A		
Applicable Value(s): N/A (default), Heat flux used to generate the fuel temperatures		
Limitation(s): If the card is present, then temperature tables in the named file will be used to calculate fuel temperatures instead of the internal conduction solvers or COBRA-TF. If this card is not present, then internal or COBRA-TF solvers are used		
Description: This card is used to define the heat flux used to generate fuel temperature tables		
Notes: Temperature tables contain fuel temperature values as functions of power and burnup. When depleting, the thermal properties of the fuel change significantly. Internal TH and COBRA-TF do not know how these properties change when depleting, so temperature tables can be used to more accurately perform TH calculations during depletion simulations using tabulated data rather than fuel conduction solvers		

temptable_polynomial temptable_polynomial

temptable_polynomial	Floating-Point Real Numbers	Optional
Units: GWD/MT, K, K (default)		
Applicable Value(s): N/A (default), Fuel temperature table values		
Limitation(s): If the card is present, then temperature tables will be used to calculate fuel temperatures instead of the internal conduction solvers or COBRA-TF. If this card is not present, then internal or COBRA-TF solvers are used		

continued on next page...

`temptable_polynomial`, continued...

Description: This card is used to indicate the data for temperature tables
Notes: Temperature tables contain fuel temperature values as functions of power and burnup. When depleting, the thermal properties of the fuel change significantly. Simplified TH and COBRA-TF do not know how these properties change when depleting, so temperature tables can be used to more accurately perform TH calculations during depletion simulations using tabulated data rather than fuel conduction solvers

user_defined_th_filename user_defined_th_filename

`user_defined_th_filename`	Free-Form Character String, Max. Length = 200	Optional
Units: N/A		
Applicable Value(s): N/A (default), Filename of any valid HDF5 file with user-defined TH conditions		
Limitation(s): If the card is present, TH conditions defined for each state and pincell will be used to set the TH variables of each pincell in the model instead of calculating the TH condition using the internal TH solver or CTF		
Description: This card is used to indicate the name of the file containing the pin-wise TH conditions for each state		
Notes: The format of the HDF5 file must follow the same format as the HDF5 output edits. The head dataset of the file must contain STATE datasets following the *STATE_**** nomenclature, which are populated with pin-wise data with the same names as their output edit counterparts. Currently supported dataset names are *pin_fuel_temp*, *pin_clad_temp*, *pin_mod_temp*, *pin_mod_dens*, *pin_gtube_temp*, and *pin_gtube_dens*. The TH datasets must have the same shapes as their corresponding output counterparts. Users are not required to provide the TH conditions for all states and TH variables, and those which are absent will be populated based on the global state variables such as `tinlet`		

user_defined_crud_filename user_defined_crud_filename

`user_defined_crud_filename`	Free-Form Character String, Max. Length = 200	Optional
Units: N/A		
Applicable Value(s): N/A (default), Filename of any valid HDF5 file with user defined CRUD conditions		
Limitation(s): If the card is present, CRUD conditions defined for each state and pincell will be used to set the CRUD variables of each pincell in the model instead of calculating the CRUD condition using MAMBA		
Description: This card is used to indicate the name of the file containing the pin-wise CRUD conditions for each state		
Notes: The format of the HDF5 file must follow the same format as the HDF5 output edits. The head dataset of the file must contain STATE datasets following the *STATE_**** nomenclature, which are populated with pin-wise data with the same names as their output edit counterparts. Currently supported dataset names are *pin_avg_crud_thickness*, *pin_avg_crud_massdensity*, and *pin_avg_crud_borondensity*. The CRUD datasets must have the same shapes as their corresponding output counterparts. Users are required to provide the CRUD conditions for all states and CRUD variables; otherwise, an error will be thrown		

dep_edit dep_edit

dep_edit	Fixed Character String	Optional
Units: atomsvolume of pincell (default)		
Applicable Value(s): `true` (default), `false`		
Limitation(s): None		
Description: This card is used to specify if the depletion `Isum` and `Pnum` files are written, which print the pin-wise averaged isotope number densities. `Isum` prints the isotope summary file with isotopes tracked in XSMesh, and `Pnum` file prints the particle number density file with all isotopes in the depletion library		
Notes: The option has excessive memory requirements and is not advised for general usage. Only use when absolutely necessary		

dep_shielder_dt dep_shielder_dt

dep_shielder_dt	Floating-Point Real Number	Optional
Units: GWD/MTU (default)		
Applicable Value(s): 10.0 (default), > 0.0		
Limitation(s): If the `xs_shielder` card is set to `f` or `false`, this card does nothing, since cross section shielding calculations will never be performed		
Description: This card is used to control how often cross section shielding calculations are performed when depleting. It sets the maximum time in GWD/MTU that can be simulated without running a new shielding calculations		
Notes: None		

dep_substep dep_substep

dep_substep	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: This card is used to read the number of substeps for the depletion predictor and corrector step. The substep method is applied to perform multiple depletion calculations between transport calculations. Substeps should be set to 1 if using CRAM and no high-order depletion or substep renormalization. Because the depletion calculation typically takes less time than the transport calculation, using this card with high-order depletion or renormalization will often save computational time		
Notes: When not using the high-order depletion methodology or substep renormalization, 1 substep is recommended for CRAM, and 3 substeps are recommended for MATREX or internal BATEMAN. This card is also valid in OPTION block		

dep_substep_pred dep_substep_pred

dep_substep_pred	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		

continued on next page...

dep_substep_pred, continued...

Description: This card is used to read the number of substeps for the depletion predictor step. The substep method is applied to perform multiple depletion calculations between transport calculations. Substeps should be set to 1 if using CRAM and no high-order depletion or substep renormalization. Because the depletion calculation typically takes less time than the transport calculation, using this card with high-order depletion or renormalization will often save computational time
Notes: When not using the high-order depletion methodology or substep renormalization, 1 substep is recommended for CRAM, and 3 substeps are recommended for MATREX or internal BATEMAN. This card is also valid in OPTION block

dep_substep_corr dep_substep_corr

dep_substep_corr	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: This card is used to read the number of substep for the depletion corrector step. The substep method is applied to perform multiple depletion calculations between transport calculations. Substeps should be set to 1 if using CRAM and no high-order depletion or substep renormalization. Because the depletion calculation typically takes less time than the transport calculation, using this card with high-order depletion or renormalization will often save computational time		
Notes: When not using the high-order depletion methodology or substep renormalization, 1 substep is recommended for CRAM, and 3 substeps are recommended for MATREX or internal BATEMAN. This card is also valid in OPTION block		

dep_kernel dep_kernel

dep_kernel	Fixed Character String	Required
Units: N/A		
Applicable Value(s): internal(MPACT's internal depletion kernel) (default), origen(coupled origen kernel)		
Limitation(s): None		
Description: This card is used to specify the depletion kernel to use. The MPACT internal depletion kernel is based on the same methodology as origen but uses simplified depletion chains and runs faster than origen		
Notes: None		

include_depl_mats include_depl_mats

include_depl_mats	Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This card is used to list the names of materials the user wishes to deplete. The inputs for this card are a 1D array of strings. The default value is an empty array		
Notes: None		

exclude_depl_mats exclude_depl_mats

exclude_depl_mats	Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This card is used to list the names of materials the user does not wish to deplete. The inputs for this card are a 1D array of strings. The default value is an empty array		
Notes: None		

cmfd cmfd

cmfd	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): cmfd (default), odcmfd, scmfd, mlcmfd, msed, none (2D only)		
Limitation(s): None		
Description: This card is used to specify which CMFD method will be used. The options are described as follows: • cmfd: default CMFD method (currently odcmfd.) • adcmfd: artificially diffusive CMFD method. (deprecated) • odcmfd: optimally diffusive CMFD method. (replaces adcmfd) • scmfd: standard CMFD method. • mlcmfd: a multi-level (currently 2) cmfd method. • msed: same as odcmfd, but the CMFD system is now solved via the MSED method • none: disables CMFD and can only be used in 2D problems.		
Notes: CMFD must be present for every 3D problem because it is the basis for the solution transfer between 2D and 1D		

multilevel multilevel

multilevel	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): energy (default), space		
Limitation(s): None		
Description: This card is used to specify whether space, energy, or both space and energy multilevel CMFD is used		
Notes: Only active when mlcmfd is specified for the cmfd card		

max_v_cycles max_v_cycles

max_v_cycles	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), Any positive integer		
Limitation(s): None		
Description: The maximum number of multilevel CMFD V-cycles to be performed on each outer iteration		
Notes: Only active when mlcmfd is specified for the cmfd card		

prolongation prolongation

prolongation	Character String	Optional
Units: N/A		
Applicable Value(s): flat (default), linear		
Limitation(s): None		
Description: Flag to indicate if flat or linear prolongation will be used for CMFD		
Notes: None		

cmfd_solver cmfd_solver

cmfd_solver	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): mgnode (default), mggroup, 1gsweep, 1grbsor, mgrbsor, reducedmg		
Limitation(s): None		
Description: This card is used to specify how the CMFD linear system is setup and solved. The options are described as follows: • 1gsweep: sweeps through all of the energy groups one by one using Gauss-Seidel iteration in energy. • mgnode: sets up a full multigroup CMFD matrix in node-major ordering (e.g., each node is a group-by-group block). • mggroup: sets up a full multigroup CMFD matrix in group-major ordering. • 1grbsor: sweeps through all of the energy groups one by one using Red-Black Successive Over-Relaxation iteration. • mgrbsor: sets up a full multigroup CMFD matrix in node-major ordering (e.g., each node is a group-by-group block) and uses Red-Black Successive Over-Relaxation iteration. • reducedmg: same as mgnode, except it solves the groups without an upscattering source one group at a time before forming a multigroup matrix with only the upscattering groups. DOES NOT WORK WITH WIELANDT SHIFT. k_shift (or lambda_shift) must be 0.		
Notes: 1gsweep requires less memory than the others, but it is generally slower to converge than mgnode		

cmfd_linear_solver cmfd_linear_solver

cmfd_linear_solver	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): PETSC (default), TRILINOS or NATIVE		
Limitation(s): None		
Description: This card is used to specify which linear solver package will be used. The options are described as follows: • PETSC: uses PETSC (ANL) for linear solver and SLEPC for eigenvalue problems. • TRILINOS: uses Trilinos (SNL) solvers Belos for linear solves and Anasazi for eigenvalue problems. • NATIVE: uses native Futility code for linear solves and eigenvalue problems.		
Notes: CMFD must be present for every 3D problem because it is the basis for the solution transfer between 2D and 1D		

petsc_linear_solver_method petsc_linear_solver_method

petsc_linear_solver_method	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): gmres (default), bicgstab, or multigrid		
Limitation(s): None		
Description: This card is used to specify which linear solver from PETSc will be used. It does nothing if Trilinos is chosen as the linear solver		
Notes: None		

petsc_linear_solver_method_1G petsc_linear_solver_method_1G

petsc_linear_solver_method_1G	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): gmres (default), bicgstab, or multigrid		
Limitation(s): None		
Description: This card is used to specify which linear solver from PETSc will be used in 1 group calculations. It does nothing if Trilinos is chosen as the linear solver		
Notes: None		

multigrid_cg_solver multigrid_cg_solver

multigrid_cg_solver	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): gmres (default), sor, bjacobi, bicgstab, or lu		
Limitation(s): None		
Description: This card is used to control the solver used on the coarsest grid of multigrid. The options are: • gmres – Standard GMRES solver in PETSc, with a preconditioner that is ILU-like locally and Jacobi-like between processors. • bicgstab – Standard BiCGSTAB solver in PETSc, same preconditioner as GMRES. • lu – Exact LU solver. In parallel, superLU package must be enabled to use this. • Any of the options for the multigrid_smoother card		
Notes: Only active when msed is specified for the cmfd card		

multigrid_cg_solver_its multigrid_cg_solver_its

multigrid_cg_solver_its	Integer	Optional
Units: N/A		
Applicable Value(s): 15 (default), > 0		
Limitation(s): None		
Description: Number of cg_solver iterations to perform on coarsest grid of the multigrid solver		
Notes: Only active when msed is specified for the cmfd card		

multigrid_cg_tol multigrid_cg_tol

multigrid_cg_tol	Float	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Set the tolerance for the coarsest grid on the multigrid system		
Notes: Only active when msed is specified for the cmfd card		

multigrid_cg_solver_1G multigrid_cg_solver_1G

multigrid_cg_solver_1G	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): gmres (default), sor, bjacobi, bicgstab, or lu		
Limitation(s): None		
Description: This card is used to control the solver used on the coarsest grid of multigrid. The options are: • gmres – Standard GMRES solver in PETSc, with a preconditioner that is ILU-like locally and Jacobi-like between processors. • bicgstab – Standard BiCGSTAB solver in PETSc, same preconditioner as GMRES. • lu – Exact LU solver. In parallel, superLU package must be enabled to use this. • Any of the options for the multigrid_smoother card		
Notes: Only active when msed is specified for the cmfd card		

multigrid_cg_solver_its_1G multigrid_cg_solver_its_1G

multigrid_cg_solver_its_1G	Integer	Optional
Units: N/A		
Applicable Value(s): 15 (default), > 0		
Limitation(s): None		
Description: Number of cg_solver iterations to perform on coarsest grid of the multigrid solver		
Notes: Only active when msed is specified for the cmfd card		

multigrid_cg_tol_1G multigrid_cg_tol_1G

multigrid_cg_tol_1G	Float	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Set the tolerance for the coarsest grid on the 1G multigrid system		
Notes: Only active when msed is specified for the cmfd card		

multigrid_smoother multigrid_smoother

multigrid_smoother	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): sor (default), sor, bjacobi		

continued on next page...

multigrid_smoother, continued...

Limitation(s): None
Description: This card is only used when petsc_linear_solver_method is set to multigrid or if petsc_linear_solver_method_1G is set to multigrid and the corresponding 1G quantity is not available. The same is true of any card beginning with "multigrid_". This card is used to control the smoother that is used on all but the coarsest grid in multigrid. The options are: • sor – PCSOR from PETSc. It is not really SOR since it does not give it a relaxation parameter. It is Gauss-Seidel locally and Jacobi between processors. • bjacobi – Block Jacobi preconditioner where each proc is a block in the global matrix. Each block is partially inverted by an ILU iteration. (ILU locally, Jacobi globally)
Notes: Only active when msed is specified for the cmfd card

multigrid_num_smooth multigrid_num_smooth

multigrid_num_smooth	Integer	Optional
Units: N/A		
Applicable Value(s): 1/0 (default), > 0		
Limitation(s): None		
Description: This card is used to control the number of smoother iterations used on each level of the multigrid scheme except the the coarsest. If no value is given, it will do one smoother iteration on the way down and no smoother iterations on the way up. If a value is given, it will do that many iterations on the way up and on the way down. The only way to achieve the default behavior is to leave this entry blank		
Notes: Only active when msed is specified for the cmfd card		

multigrid_smoother_1G multigrid_smoother_1G

multigrid_smoother_1G	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): sor (default), sor, bjacobi		
Limitation(s): None		
Description: This card is only used when petsc_linear_solver_method_1G is set to multigrid. This card is used to control the smoother that is used on all but the coarsest grid in multigrid. The options are: • sor – PCSOR from PETSc. It is not really SOR since it has no relaxation parameter. It is Gauss-Seidel locally and Jacobi between processors. • bjacobi – Block Jacobi preconditioner where each proc is a block in the global matrix. Each block is partially inverted by an ILU iteration. (ILU locally, Jacobi globally)		
Notes: Only active when msed is specified for the cmfd card		

multigrid_num_smooth_1G multigrid_num_smooth_1G

multigrid_num_smooth_1G	Integer	Optional
Units: N/A		
Applicable Value(s): 1/0 (default), > 0		
Limitation(s): None		

continued on next page...

multigrid_num_smooth_1G, continued...

Description: This card is used to control the number of smoother iterations used on each level of the multigrid scheme except the the coarsest. If no value is given, it will do one smoother iteration on the way down and no smoother iterations on the way up. If a value is given, it will do that many iterations on the way up and on the way down. The only way to achieve the default behavior is to leave this entry blank
Notes: Only active when msed is specified for the cmfd card

multigrid_log_flag multigrid_log_flag

multigrid_log_flag	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card must be set to true for PETSc to printout performance and logging information for the multigrid solver. However, setting this to true is not sufficient. The user must also provide MPACT with the command line option -pc_mg_log at runtime		
Notes: Only active when msed is specified for the cmfd card		

multigrid_log_flag_1G multigrid_log_flag_1G

multigrid_log_flag_1G	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card must be set to true for PETSc to printout performance and logging information for the multigrid solver. However, setting this to true is not sufficient. The user must also provide MPACT with the command line option -pc_mg_log at runtime		
Notes: Only active when msed is specified for the cmfd card		

multigrid_precond_flag multigrid_precond_flag

multigrid_precond_flag	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Setting this card to true makes the code use multigrid as a preconditioner to GMRES rather than as a standalone solver		
Notes: Only active when msed is specified for the cmfd card		

multigrid_precond_flag_1G multigrid_precond_flag_1G

multigrid_precond_flag_1G	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		

continued on next page...

multigrid_precond_flag_1G, continued...

Description: Setting this card to true makes the code use multigrid as a preconditioner to GMRES rather than as a standalone solver
Notes: Only active when msed is specified for the cmfd card

preconditioner preconditioner

preconditioner	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): default (default), ilu, bilu, bjacobi_ilu, mg, eisenstat, none		
Limitation(s): None		
Description: This card is used to specify which preconditioner should be used in the CMFD solver. The "default" preconditioner depends on the method used to solve the CMFD eigenvalue problem. The "mg" preconditioner in PETSC is a bit misleading. It is not actually a multigrid preconditioner since PETSc is never provided with any information regarding the grid or interpolation/restriction; it simply performs a smoothing step on the fine grid level using its default smoother		
Notes: None		

cmfd_eigen_solver cmfd_eigen_solver

cmfd_eigen_solver	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): power (default), jd, gd, arnoldi, slepc_power		
Limitation(s): None		
Description: This card is used to specify which eigenvalue solver will be used. The options are described as follows: • power: standard power iteration. • JD: SLEPc Jacobi-Davidson Solver. • GD: SLEPc or Anasazi Generalized Davidson Solver depends on cmfd_linear_solver. • Arnoldi: SLEPc Arnoldi Solver. • SLEPc_power: SLEPc power iteration for comparison.		
Notes: CMFD must be present for every 3D problem because it is the basis for the solution transfer between 2D and 1D		

k_shift k_shift

k_shift	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.5 (default)		
Limitation(s): Can only be used with the mgnode CMFD solver. This card is irrelevant unless the constant option is used for the cmfd_shift_method card		
Description: This card is used to specify a shifted eigenvalue problem for the CMFD power iterations		
Notes: k_shift should be larger than the eigenvalue of the system. Even a value of 2 would provide some enhanced convergence properties over not using k_shift		

k_shift_1G k_shift_1G

k_shift_1G	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.5 (default)		
Limitation(s): Can only be used with the mgnode CMFD solver. This card is irrelevant unless the constant option is used for the cmfd_shift_method_1G card and the msed option is used for the cmfd card		
Description: This card is used to specify a shifted eigenvalue problem for the 1G CMFD power iterations		
Notes: k_shift_1G should be larger than the eigenvalue of the system. Even a value of 2 would provide some enhanced convergence properties over not using k_shift_1G		

cmfd_relaxation cmfd_relaxation

cmfd_relaxation	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.0 (default)		
Limitation(s): None		
Description: This card is for specifying the relaxation parameter for the CMFD flux update. The default value (1.0) corresponds to no relaxation of the update. Values below 1.0 under-relax the CMFD flux update to provide stability for cases with T/H or other feedback. For standalone neutronics problems, no under-relaxation should be needed to achieve stability when using the odcmfd option, and any under-relaxation will probably degrade the convergence rate		
Notes: None		

cmfd_relax_negative cmfd_relax_negative

cmfd_relax_negative	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This option is to under-relax the CMFD flux when negative CMFD flux is calculated. If a negative CMFD flux is calculated at the end of CMFD iteration, then the CMFD flux is under-relaxed by the group-dependent relaxation factor. The under-relaxation factor, f_g, is determined to satisfy the following inequality: $f_g\phi_g^{\text{new}} + (1 - f_g)\phi_g^{\text{old}} > 0$		
Notes: This card can be used with cmfd_relaxation. This card changes the CMFD flux itself only if a negative CMFD flux is calculated. However, the cmfd_relaxation card changes the projection factor		

cmfd_dhat_relaxation cmfd_dhat_relaxation

cmfd_dhat_relaxation	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.0 (default)		
Limitation(s): None		

continued on next page...

cmfd_dhat_relaxation, continued...

Description: This card is for specifying the relaxation parameter for the CMFD dHat update. The default value (1.0) corresponds to no relaxation of the update. Values below 1.0 under-relax the CMFD dHat update to provide stability for cases with very large flux gradients. For typical neutronics problems, no under-relaxation should be needed to achieve stability when using the CMFD option, and any under-relaxation will probably degrade the convergence rate. When running an external source driven problem, under-relaxation may be necessary to obtain convergence, In the most extreme cases, under-relaxation may be set to 0.0, which effectively removes the dHat correction coefficient in the CMFD calculation, and results in CMFD calculating a more traditional diffusion solution. When running with no dHat correction coefficient, the equivalence between CMFD and fine mesh transport solutions is no longer guaranteed!
Notes: None

cmfd_shift_c0 cmfd_shift_c0

cmfd_shift_c0	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 0.02 (default), > 0.0		
Limitation(s): Can only be used with the mgnode CMFD solver. This card is irrelevant unless the adaptive, ileps, or ilaps shift is being used		
Description: This card is used to specify the c0 parameter used in the adaptive/ileps/ilaps shift. c0 is used to reduce the shift to ensure both a positive fission source and a subcritical diffusion operator (i.e., to prevent overshifting)		
Notes: None		

cmfd_shift_method cmfd_shift_method

cmfd_shift_method	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): constant (default), none, adaptive, sdws-ileps, sdws-ilaps, sdws-laps, adap-ratio		
Limitation(s): These methods can only be used with the mgnode CMFD solver or the mgrbsor solver		

continued on next page...

cmfd_shift_method, continued...

Description: This card is used to specify which Wielandt shift method will be used to accelerate the power iterations on the CMFD problem. The options are described as follows:

- `none`: does not apply a shift to the CMFD system
- `constant`: applies a constant, iteration-independent shift to the CMFD system. The constant is given by the reciprocal of the input to the `cmfd` card
- `adaptive`: uses a traditional Wielandt shift method. The shift parameter is an iteration-dependent, spatially constant quantity defined by:

$$\lambda_{adaptive}^{(n)} = \max\left\{\lambda^{(n)} - c_1\left|\lambda^{(n)} - \lambda^{(n-1)}\right| - c_0\,, \lambda_{min}\right\}\,.$$

  c_1, c_0, and λ_{min} have been hard-coded to 10, 0.02, and 0.3, respectively. Future implementations of the method may allow the user to specify these parameters
- `sdws-ileps`: uses a space- and iteration-dependent Wielandt shift based on the local infinite-medium eigenvalues, $\lambda_{adaptive}$, and the current guess of the eigenvalue:

$$\lambda_{IPS}^{(n)}(\mathbf{x}) = \max\left\{\lambda_{adaptive}^{(n)}\,, \min\left\{\lambda_{\infty}(\mathbf{x}), \lambda^{(n)} - 0.01\right\}\right\}$$

- `sdws-ilaps`: combines sdws-laps with the adaptive shift
- `sdws-laps`: uses a space- and iteration-dependent Wielandt shift based on the local absorption values. The shift is limited in order to ensure a non-negative fission source
- `adap-ratio`: uses a traditional Wielandt shift method. The shift parameter is an iteration-dependent, spatially constant quantity defined by:

$$\lambda_{adaptive}^{(n)} = \max\left\{r\lambda^{(n)} - c_1\left|\lambda^{(n)} - \lambda^{(n-1)}\right|\,, \lambda_{min}\right\}\,.$$

  r is defined by adap-ratio

Notes: None

cmfd_shift_method_1G cmfd_shift_method_1G

`cmfd_shift_method_1G`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): constant (default), none, adaptive, sdws-ileps, sdws-ilaps, sdws-laps, adap-ratio		
Limitation(s): This card is only used if the CMFD card is set to `msed`		
Description: This card is used to specify which Wielandt shift method will be used to accelerate the power iterations on the 1G CMFD problem. See `cmfd_shift_method` card for description. This card is only applicable if a 1G CMFD system is being used to accelerate the MG CMFD system		
Notes: None		

cmfd_ktol cmfd_ktol

`cmfd_ktol`	Floating Point Number	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-6} (default), > 0.0		
Limitation(s): None		

continued on next page...

`cmfd_ktol`, continued...

Description: This card is used to specify the tolerance for the convergence of k in the overall CMFD eigenvalue problem
Notes: None

cmfd_rtol cmfd_rtol

`cmfd_rtol`	Floating Point Number	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-6} (default), > 0.0		
Limitation(s): None		
Description: This card is used to specify the tolerance for the relative residual reduction in a CMFD linear system solved each power iteration		
Notes: None		

cmfd_ktol_1G cmfd_ktol_1G

`cmfd_ktol_1G`	Floating Point Number	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-6} (default), > 0.0		
Limitation(s): None		
Description: This card is used to specify the tolerance for the convergence of k in the 1G CMFD eigenvalue problem in MSED		
Notes: None		

cmfd_flxtol_1G cmfd_flxtol_1G

`cmfd_flxtol_1G`	Floating Point Number	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-6} (default), > 0.0		
Limitation(s): None		
Description: This card is used to specify the tolerance for the convergence of the flux in the 1G CMFD eigenvalue problem in MSED		
Notes: None		

max_1G_eig_its max_1G_eig_its

`max_1G_eig_its`	Integer	Optional
Units: N/A		
Applicable Value(s): 20 (default), > 0		
Limitation(s): None		
Description: This card is used to specify the maximum number of power power iterations allowed on the 1G CMFD system in MSED		
Notes: None		

cmfd_num_inners cmfd_num_inners

cmfd_num_inners	Integer	Optional
Units: N/A		
Applicable Value(s): 100 (default), > 0		
Limitation(s): None		
Description: This card is used to specify the maximum number of linear solver iterations per power iteration during a CMFD acceleration calculation		
Notes: None		

cmfd_num_inners_1G cmfd_num_inners_1G

cmfd_num_inners_1G	Integer	Optional
Units: N/A		
Applicable Value(s): 100 (default), ≥ 0		
Limitation(s): None		
Description: This card is used to specify the maximum number of linear solver iterations allowed per power iterations in the 1G CMFD system in MSED		
Notes: None		

linear_solver_tol linear_solver_tol

linear_solver_tol	Floating Point Number	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-10} (default), > 0.0		
Limitation(s): None		
Description: This card is used to specify the tolerance of linear solver used at each power iteration during a CMFD acceleration calculation		
Notes: None		

linear_solver_tol_1G linear_solver_tol_1G

linear_solver_tol_1G	Floating Point Number	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-10} (default), > 0.0		
Limitation(s): None		
Description: This card is used to specify the tolerance of linear solver used at each power iteration on the 1G system in MSED		
Notes: None		

cmfd_num_outers cmfd_num_outers

cmfd_num_outers	Integer	Optional
Units: N/A		
Applicable Value(s): 20 (default), > 0		
Limitation(s): None		
Description: This card is used to specify the number of outer eigenvalue power iterations to perform during a CMFD acceleration calculation		

continued on next page...

`cmfd_num_outers`, continued...

Notes: None

cmfd_up_scatter cmfd_up_scatter

`cmfd_up_scatter`	Integer	Optional
Units: N/A		
Applicable Value(s): 2 (default), ≥0		
Limitation(s): Only applies to `1gsweep` CMFD solver		
Description: This card is used to specify the number of upscatter iterations when doing `1gsweep` CMFD. This can help to converge the scattering source in thermal energy groups before updating the fission source. In general, this can be used to help optimize run time for a given problem		
Notes: None		

cmfd_num_outers_th cmfd_num_outers_th

`cmfd_num_outers_th`	Integer	Optional
Units: N/A		
Applicable Value(s): 5 (default), > 0		
Limitation(s): None		
Description: This card is used to specify the number of outer eigenvalue power iterations to perform during a CMFD acceleration calculation when near-optimal partial convergence CMFD is used		
Notes: None		

cmfd_shift_r cmfd_shift_r

`cmfd_shift_r`	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 0.667 (default), > 0.0		
Limitation(s): Can only be used with the mgnode CMFD solver. This card is irrelevant unless the adap-ratio is used		
Description: This card is used to specify the r parameter used in the adaptive-ratio shift. r is used to reduce the shift to ensure both a positive fission source and a subcritical diffusion operator (i.e., to prevent overshifting)		
Notes: None		

cmfd_shift_r_1G cmfd_shift_r_1G

`cmfd_shift_r_1G`	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): 0.667 (default), > 0.0		
Limitation(s): Can only be used with the MSED solver. This card is irrelevant unless the adap-ratio is used		
Description: This card is used to specify the r parameter used in the adaptive-ratio shift. r is used to reduce the shift to ensure both a positive fission source and a subcritical diffusion operator (i.e., to prevent overshifting)		

continued on next page...

`cmfd_shift_r_1G`, continued...

Notes: None

subplane_target subplane_target

`subplane_target`	Floating-Point Real Number	Optional
Units: cm (default)		
Applicable Value(s): N/A (default), > 0.0		
Limitation(s): None		
Description: This card is used to designate the target thickness of axial meshes in the CMFD system		
Notes: None		

subplane_max subplane_max

`subplane_max`	Floating-Point Real Number	Optional
Units: cm (default)		
Applicable Value(s): N/A (default), > 0.0		
Limitation(s): None		
Description: This card is used to designate the maximum thickness of axial meshes in the CMFD system. All MOC planes with thicknesses greater than this will be subdivided in the CMFD system using the `subplane_target` value		
Notes: None		

subgrid_spacers subgrid_spacers

`subgrid_spacers`	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card is used to designate whether or not spacer grids are used in subgrid solver setup		
Notes: None		

subgrid_reflector subgrid_reflector

`subgrid_reflector`	Logical	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: This card is used to designate whether or not axial reflectors are used in subgrid solver setup		
Notes: None		

subgrid_feedback subgrid_feedback

`subgrid_feedback`	Boolean	Optional
Units: N/A		

continued on next page...

`subgrid_feedback`, continued...

Applicable Value(s): false (default), true
Limitation(s): None
Description: This card is used to designate whether or not feedback logic is used in subgrid solver setup
Notes: None

num_subplanes num_subplanes

`num_subplanes`	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), > 0		
Limitation(s): None		
Description: This card is used to designate the number of subplanes used for each MOC plane in the CMFD system. Every MOC plane will be divided into `num_subplanes` subplanes. This card overrides both the `subplane_target` and `subplane_max` cards. Any of these cards may be used to control the subplane meshing, but this card is recommended since the other two result in parallel imbalance		
Notes: None		

cmfd_angle_decomp cmfd_angle_decomp

`cmfd_angle_decomp`	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): If angle decomposition or CMFD is not used, this card has no effect		
Description: This card is used to specify whether or not the angular decomposition processors for MOC are to be used during the CMFD setup/solve. The default for this treatment is `true` and is recommended for better parallel efficiency		
Notes: None		

split_TL split_TL

`split_TL`	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): Only applies to 3D models run with 2D/1D		
Description: This card is used to specify whether transverse leakage splitting will be enabled for a calculation using a 2D/1D method. In the 2D/1D method, the axial transverse leakage is subtracted from the total fission and scattering sources, so in regions with relatively large axial streaming sources, the total source may become negative. To avoid negative total sources, the transverse leakage is split between the right-hand side and the left-hand side of the 2D transport equation, thus ensuring positivity of the total source and neutron balance		
Notes: None		

split_TL_tol split_TL_tol

split_TL_tol	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): ≥0.0		
Limitation(s): None		
Description: This card is used to specify the flux tolerance used in the transverse leakage splitting or MOC source splitting. If scalar flux is less than the flux flux tolerance, the transverse leakage splitting or MOC source splitting will not be performed		
Notes: This card should be used with `split_TL true` or `moc_source_splitting full`		

split_RTL split_RTL

split_RTL	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): Only applies to 3D models run with 2D/1D		
Description: This card is used to specify whether radial leakage splitting will be enabled for a calculation using a 2D/1D method. In the 2D/1D method, the radial transverse leakage is subtracted from the total fission and scattering sources, so in regions with relatively large radial streaming sources, the total source may become negative. To avoid negative total sources, the radial leakage is divided between the right-hand side and left-hand side of the 1D transport equation, thus ensuring positivity of the total source and neutron balance		
Notes: None		

TL_treatment TL_treatment

TL_treatment	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): lflat (default), flat		
Limitation(s): None		
Description: This card is used to specify the type of spatial shape of the axial transverse leakage applied to the 2D problem. `flat` means it is constant over a pin cell. This is primarily used to ensure stability of the iteration. These options are described as follows: • `lflat`: checks the total / transport cross section. If the value is below the threshold, then leakage will not be put into that region. This process is usually to avoid leakage in the fuel- clad gap. The leakage will then be redistributed to the other regions in that pin. • `flat`: does not perform leakage threshold checks.		
Notes: None		

moc_source_splitting moc_source_splitting

moc_source_splitting	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): none (default), full, linear		

continued on next page...

`moc_source_splitting`, continued...

Limitation(s): None
Description: This card is used to describe the type of source splitting that is done before the MOC calculation to improve the stability of the calculation. These options are described as follows: • `none`: no source splitting is performed in MOC. • `full`: if the source is negative before the MOC sweep, then all of the source is moved from the right to left side of the equation by modifying the transport cross section. • `linear`: if the linear source is negative before the MOC sweep, then the gradient of the linear source and the quantity of source splitting are adjusted to ensure the non-negativity of the linear source.
Notes: Minor changes in the converged solution will occur when source splitting is enabled. Source splitting is never implemented for gamma transport because of the detrimental effects on accuracy for highly anisotropic calculations. This card has no impact on the splitting of the axial transverse leakage source; the splitting controlled by this card is performed after the axial leakage splitting and after adding in the self-scatter source for the radial transport sweep

nodal_method nodal_method

`nodal_method`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): sp3 (default), nem, senm, sn-0, sn-1, sn-2, sn-3, p1, p3, p5, hyp3, fhp1, fhp3, senmp1, senmp3, senmp5, none		
Limitation(s): Only applies to 3D models run with 2D/1D		
Description: This card is used to specify the type of nodal axial solver that will be used to solve the 1D portion of the 2D/1D solution.		

Described as follows:

Input Option	Full Name
SENM	Two-Node Source Expansion Nodal Method
NEM	Nodal Expansion Method
NEM-MG	Multi-Group Nodal Expansion Method
SN-0	Discrete Ordinates with 0th Spatial Moment
SN-1	Discrete Ordinates with 1st Spatial Moment
SN-2	Discrete Ordinates with 2nd Spatial Moment
SN-3	Discrete Ordinates with 3rd Spatial Moment
P1	Pn 1st Order with One-Node NEM
P3	Pn 3rd Order with One-Node NEM
P5	Pn 5th Order with One-Node NEM
HYP3	Hybrid Pn 3rd Order with NEM
FHP1	1st Order with Full Height NEM
FHP3	3rd Order with Full Height NEM
SENMP1	Pn 1st Order with One-Node SENM
SENMP3	Pn 3rd Order with One-Node SENM
SENMP5	Pn 5th Order with One-Node SENM
NONE	Finite-Difference Method

continued on next page...

nodal_method, continued...

Notes: The Sn methods are the most computationally intensive. SP3 is recommended as the best balance of accuracy and speed. If convergence/stability issues are encountered with SP3, then try running with NEM

gamma_nodal_method gamma_nodal_method

gamma_nodal_method	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): sp3 (default), nem, senm, sn-0, sn-1, sn-2, sn-3, p1, p3, p5, hyp3, fhp1, fhp3, senmp1, senmp3, senmp5, none		
Limitation(s): Only applies to 3D models run with 2D/1D		
Description: This card is used to specify the type of nodal axial solver that will be used to solve the 1D portion of the 2D/1D solution for gamma transport. Described as follows: Input Option — Full Name SENM — Two-Node Source Expansion Nodal Method NEM — Nodal Expansion Method NEM-MG — Multi-Group Nodal Expansion Method SN-0 — Discrete Ordinates with 0th Spatial Moment SN-1 — Discrete Ordinates with 1st Spatial Moment SN-2 — Discrete Ordinates with 2nd Spatial Moment SN-3 — Discrete Ordinates with 3rd Spatial Moment P1 — Pn 1st Order with One-Node NEM P3 — Pn 3rd Order with One-Node NEM P5 — Pn 5th Order with One-Node NEM HYP3 — Hybrid Pn 3rd Order with NEM FHP1 — 1st Order with Full Height NEM FHP3 — 3rd Order with Full Height NEM SENMP1 — Pn 1st Order with One-Node SENM SENMP3 — Pn 3rd Order with One-Node SENM SENMP5 — Pn 5th Order with One-Node SENM NONE — Finite-Difference Method		
Notes: The Sn methods are the most computationally intensive. SP3 is recommended as the best balance of accuracy and speed. If convergence/stability issues are encountered with SP3, then try running with NEM		

nodal_inners nodal_inners

nodal_inners	Integer	Optional
Units: N/A		
Applicable Value(s): varies depending on nodal method (default), ≥1		
Limitation(s): None		
Description: This card is used to specify the number of inner 1-group nodal sweeps performed during group sweeping for every outer iteration		
Notes: None		

nodal_group_loop nodal_group_loop

nodal_group_loop	Integer	Optional
Units: N/A		
Applicable Value(s): varies depending on nodal method (default), ≥1		
Limitation(s): None		
Description: This card is used to specify the number of iterations over energy groups performed during the nodal solve for every outer iteration		
Notes: None		

nodal_leakage_order nodal_leakage_order

nodal_leakage_order	Integer	Optional
Units: N/A		
Applicable Value(s): 2 (default), 0, 1, or 2		
Limitation(s): None		
Description: This card is used to specify the interpolation order for radial transverse leakage for nodal solves		
Notes: None		

nodal_group_start nodal_group_start

nodal_group_start	Integer	Optional
Units: N/A		
Applicable Value(s): >0		
Limitation(s): None		
Description: This card is used to specify the starting group index of nodal group iterations when nodal_group_loop is larger than 1		
Notes: If this card is not specified, then the starting group index is determined according to the range of up-scattering		

nodal_inner_tol nodal_inner_tol

nodal_inner_tol	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): >0.0		
Limitation(s): None		
Description: This card is used to specify the tolerance on the convergence of the 2-norm of the nodal flux residual during within group nodal inner iterations		
Notes: None		

nodal_group_tol nodal_group_tol

nodal_group_tol	Floating-Point Real Number	Optional
Units: N/A		
Applicable Value(s): >0.0		

continued on next page...

`nodal_group_tol`, continued...

Limitation(s): None
Description: This card is used to specify the tolerance on the convergence of the 2-norm of the nodal flux residual during nodal group iterations
Notes: None

nodal_relax_negative nodal_relax_negative

`nodal_relax_negative`	Boolean	Optional
Units: N/A		
Applicable Value(s): `false` (default), `true`		
Limitation(s): None		
Description: This option is used to under-relax the nodal flux when the negative nodal flux is calculated. The group-dependent under-relaxation factor, f_g, is determined to satisfy the following inequality: $f_g\phi_g^{\text{new}} + (1 - f_g)\phi_g^{\text{old}} > 0$		
Notes: None		

gamma_nodal_inners gamma_nodal_inners

`gamma_nodal_inners`	Integer	Optional
Units: N/A		
Applicable Value(s): varies depending on nodal method (default), ≥1		
Limitation(s): None		
Description: This card is used to specify the number of inner 1-group nodal sweeps performed during gamma group sweeping for every outer iteration/		
Notes: None		

gamma_nodal_group_loop gamma_nodal_group_loop

`gamma_nodal_group_loop`	Integer	Optional
Units: N/A		
Applicable Value(s): varies depending on nodal method (default), ≥1		
Limitation(s): None		
Description: This card is used to specify the number of iterations over energy groups performed during the gamma nodal solve for every outer iteration		
Notes: None		

gamma_nodal_leakage_order gamma_nodal_leakage_order

`gamma_nodal_leakage_order`	Integer	Optional
Units: N/A		
Applicable Value(s): 2 (default), 0, 1, or 2		
Limitation(s): None		
Description: This card is used to specify the interpolation order for radial transverse leakage for gamma nodal solves		
Notes: None		

sntype sntype

sntype	Character String	Optional
Units: N/A		
Applicable Value(s): isotropic, aziint, explicit, moment, p3-moment, none		
Limitation(s): Only applies to 3D models run with 2D/1D		
Description: This card is used to specify the type of axial sn sweeper that will be used to solve the 1D portion of the 2D/1D solution		
Notes: None		

rtltype rtltype

rtltype	Character String	Optional
Units: N/A		
Applicable Value(s): isotropic, aziint, explicit, moment, p3-moment, p3-quad, p3-quadratic, p3-even, sym, none		
Limitation(s): None		
Description: The type of radial transverse leakage to use		
Notes: None		

atltype atltype

atltype	Character String	Optional
Units: N/A		
Applicable Value(s): isotropic, aziint, explicit, moment, azi, exp, mom, sym, none		
Limitation(s): None		
Description: The type of angular transverse leakage treatment to be used		
Notes: None		

rtlmom rtlmom

rtlmom	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of azimuthal Fourier moments to be used in the radial transverse leakage construction		
Notes: None		

homtype homtype

homtype	Character String	Optional
Units: N/A		
Applicable Value(s): isotropic (default), polar, moment, explicit, symmetric, none		
Limitation(s): None		

continued on next page...

homtype, continued...

Description: The homtype option specifies the type of homogenization to use for the 1D solver. ANGLE_POL is polar polar-dependent homogenization. This can be used with the P3-MOMENT, MOMENT-MOMNET, or P3-EVENODD radial TL options. ANGLE_MOM is not recommended because it is much slower. ANGLE_EXP can be used with EXPLICIT-EXPLICIT radial TL. With all explicit options, the 2D/1D method uses exact angular TL and exact homogenized anisotropic XS, which are the most accurate (but expensive)
Notes: None

under_relax under_relax

under_relax	Float	Optional
Units: N/A		
Applicable Value(s): > 0, < 2		
Limitation(s): None		
Description: The underrelaxation factor to use when doing 2D/1D		
Notes: None		

mesh mesh

mesh	Fixed Character String Followed by Two Arrays of Integers Separated by a '/'	Required
Units: N/A		
Applicable Value(s): num_rad = 3, 1 and num_azi = 1, 8, 8, 8, 12 (default), For num_rad, positive integers greater than zero. For num_azi, 1, 4, 8, 12, or 16. The length for num_rad is the number of geometric radii, and the length for num_azi is the sum of the sub-divided radii		
Limitation(s): None		
Description: This card is used to specify the radial and azimuthal mesh for each cell. Currently, two cell types are used: fuel and gtube. Cells containing fuel materials are flagged to use the fuel mesh, and all other cells use the gtube meshing. For the inputs, num_rad is the number of radial subdivisions in each ring specified in the cell, and num_azi is the number of azimuthal regions in each subdivided radial ring. The last azimuthal value applies to the region outside the pin		

continued on next page...

`mesh`, continued...

Notes: Currently insert, control, and detector rods have predefined mesh that cannot be overwritten. In both cases, the last entry will be used for any remaining unspecified regions. For example, if a given fuel pin has 3 radial and material regions, and the fuel mesh had a `num_rad` of 3,1 and `num_azi` of 1,4,8, then the third ring in the fuel pin would have 1 radial subdivision, and the fourth subdivided radius to the end of the pin cell would have 8 azimuthal subdivisions, including the region outside the pincell. If the mesh is specified too finely—or rather, finer than the value for ray spacing—instabilities may occur in which a ray is NOT traced through a flat source region, and no flux is calculated for that region. The code will automatically adjust the azimuthal discretization if the given ray spacing value is too coarse (or because the azimuthal mesh is too fine). Another way to cause the above instability would be to specify a very large number of radial subdivisions for the first `num_rad value`. That large number being the area of the first radius divided by the first `num_rad` value would have to yield a radius that is smaller than the ray spacing. For a typical PWR fuel pin radius, the first `num_rad value` must be well over 100 for this problem to arise, and this number is impractical given the memory it will consume

1gacceltr 1gacceltr

`1gacceltr`	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Flag for 1GCMFD acceleration for transport part of transient simulations		
Notes: None		

1gaccel 1gaccel

`1gaccel`	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Flag for 1GCMFD acceleration of transient simulations		
Notes: None		

delayenergy delayenergy

`delayenergy`	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), `true`		
Limitation(s): None		
Description: This option is used to specify whether to use explicit delayed energy kernel during the transient calculation. The default is false. The equilibrium delayed energy (about 7 percent of total fission energy including delayed beta and gamma) is assumed as default		
Notes: None		

kinetics_data kinetics_data

kinetics_data	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): library (default), scale, keepin, tuttle, jeff3, santamarina, library, spert70f, spert250f, spert500f		
Limitation(s): None		
Description: This card is used to specify the set of kinetics data used in the transient calculation. This card is only applied to the MPACT cross section library for now. By default, MPACT uses the 6-group transient data provided in the MPACT MG cross section library. These are described as follows: 1. `scale`: the 6-group transient data from SCALE 2. `keepin`: the 6-group transient data from G. R. Keepin's paper 3. `tuttle`: the 6-group transient data from R. J. Tuttle's paper 4. `jeff3`: the 8-group transient data from JEFF3 with uniform lambda 5. `santamarina`: the 8-group transient data suggested by A. Santamarina (a slight modification of JEFF3) 6. `library`: the 6-group transient data in the MPACT cross section library from ENDF 7. `spert70f,spert250f,spert500f`: the 6-group transient data measured in spert experiments		
Notes: None		

kinetics_lambda kinetics_lambda

kinetics_lambda	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): fissweight (default), isotopic, fissweight, precursorconsv		
Limitation(s): None		
Description: This card controls the calculation of decay constants for each fissile region. The isotopic lambda is the exact approach but can use a lot more memory. In general, it is recommended to use the precursor conservation option rather than fission source weighting. These options are described as follows: 1. `isotopic`: use exact isotope-dependent lambdas 2. `fissweight`: collapse isotopic lambdas by fission rate 3. `precursorconsv`: collapse isotopic lambdas by preserving the initial precursors		
Notes: None		

kinetics_otfbeta kinetics_otfbeta

kinetics_otfbeta	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), `true`		
Limitation(s): None		
Description: This option specifies whether to compute the problem-dependent nu-bar for on-the-fly calculation of beta. By default, the problem-independent beta computed from a typical PWR spectrum is used		
Notes: None		

rx_components rx_components

rx_components	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): Can only be used when acceleration is enabled		
Description: This option is used to specify whether to calculate component reactivity values. The default is false. This option is ignored for steady-state calculations		
Notes: None		

sep_flux_comp sep_flux_comp

sep_flux_comp	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: When rx_components is set to true, this card is used for separating flux shape reactivity		
Notes: None		

summary_edits summary_edits

summary_edits	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): Only used for transient cases		
Description: This option is used to specify whether to print a summary file (<caseid>.sum) containing data for each transient timestep. The default is false. This option is ignored for steady-state calculations		
Notes: None		

tml1gmg tml1gmg

tml1gmg	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): none (default), sweep, hybrid		
Limitation(s): None		
Description: The tml1gmg option specifies how to use 1GCMFD level for new TML. These are described as follows: 1. none: is to use the 1GCMFD to update the flux 2. sweep: is to use 1GCMFD for fission source and MGCMFD for flux 3. hybrid: is to not use 1GCMFD level when there is external reactivity, and 1GCMFD for flux when there is no external reactivity		
Notes: None		

tml tml

tml	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Flag to enable the transient multi-level (TML) method		
Notes: None		

tmllevel nCMFD nEPKE/n1GCMFD n1GCMFD/nEPKE

nCMFD	Integer	Optional
Units: N/A		
Applicable Value(s): 5 (default), > 0		
Limitation(s): None		
Description: The number of CMFD acceleration steps taken for every transport time step, this is always the first entry in the tmllevel card		
Notes: Is only used when tml is set to true		

nEPKE	Integer	Optional
Units: N/A		
Applicable Value(s): 10 (default), > 0		
Limitation(s): None		
Description: The number of EPKE calculation steps taken for every transport time step. If there are only 2 numbers listed in the tmllevel card, the second number should be nEPKE. If there are 3 numbers listed in the tmllevel card, the third number should be nEPKE		
Notes: Is only used when tml is set to true		

n1GCMFD	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), > 0		
Limitation(s): None		
Description: The number of 1G CMFD acceleration steps taken for every transport time step. This variable is only set if there are 3 numbers listed in the tmllevel card, in which case, the second number is nEPKE		
Notes: Is only used when tml is set to true		

transmethod transmethod

transmethod	Fixed Character Array, Positive Real Number or Integer. Length 1	Optional
Units: N/A		
Applicable Value(s): {theta 0.5},{BDF 2} (default), {theta 0.0-1.0},{BDF 1-6}		
Limitation(s): None		

continued on next page...

transmethod, continued...

Description: The first option is used to specify the time discretization method, theta refers to the theta method, and BDF refers to the BDF method. The <value> defines the option for the theta method [0.0,1.0] or the BDF method. For the BDF method, the value is an integer that ranges from 1 to 6. If only BDF is specified, then the default order is 2
Notes: None

perturb perturb

perturb	Array of String and doubles	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: This card is used to specify the parameters to drive the transient. The options for perturbing the system are as follows: • t1: the start time of perturbation • t2: the end time of perturbation • dt: the time step of perturbation (optional) For the option mat, its interpretation varies with the perturbation type. • STEP: for this perturbation type, the initial condition for the *i*th material is mat0(i), at time t2, it is turned into mat0 instantaneously • RAMP: for this perturbation type, the initial condition for the ith material is mat0(i). Then it changes gradually from mat1(i) to mat2(i). This change is a fractional time- weighted mixture of the two materials. The mixture at t1 is only mat1; halfway through the perturbation, it is 0.5 (mat1) and 0.5 mat2. At the end of the perturbation, it is only mat2 • CONST: for this perturbation, there are no changes to the system • MVCR: this perturbation is for moving a bank of control rods. The rod positions are specified in [STATE] blocks corresponding to each time step occurring during the perturbation		
Notes: None		

checkpoint_read checkpoint_read_file checkpoint_read_label

checkpoint_read_file	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): When used, must have an associated checkpoint_read_label		
Description: The name of the file that should be read when restarting a a transient from a previously created checkpoint		
Notes: None		

checkpoint_read_label	Float	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): Can only be used in conjunction with checkpoint_read_file		
Description: The perturbation completed before the checkpoint file was written		

continued on next page...

`checkpoint_read_label`, continued...

Notes: None

checkpoint_write checkpoint_write_file checkpoint_write_label

`checkpoint_write_file`	Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): When used, must have an associated checkpoint_write_label		
Description: The name of the file to be created when a checkpoint of a transient case is desired		
Notes: None		

`checkpoint_write_label`	Float	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): Can only be used in conjunction with checkpoint_write_file		
Description: The perturbation to be completed prior to writing the checkpoint file		
Notes: None		

mat_emit_src mat_emit_src

`mat_emit_src`	Boolean	Optional
Units: N/A		
Applicable Value(s): `false` (default), `true`		
Limitation(s): Should only be used for subcritical systems; otherwise, no steady-state solution will ever be achieved		
Description: When just `mat_emit_src` is input without `true|false`, the option is set to `false`. This option is used to specify whether or not neutron emission sources from the decay of model materials will be treated. The default is set to false		
Notes: None		

5.14 BLOCK MAMBA

A_NiFe2O4_out surface_prefactor

`surface_prefactor`	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Prefactor for $NiFe_2O_4$ surface growth		
Notes: None		

E_NiFe2O4_out surface_activation_energy

`surface_activation_energy`	Floating-point Real Number	Optional

continued on next page...

surface_activation_energy, continued...

Units: N/A
Applicable Value(s): > 0
Limitation(s): None
Description: Activation energy for $NiFe_2O_4$ surface growth
Notes: None

A_NiFe2O4_in nucleation_prefactor

nucleation_prefactor	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Prefactor for $NiFe_2O_4$ nucleation		
Notes: None		

E_NiFe2O4_in nucleation_activation_energy

nucleation_activation_energy	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Activation energy for $NiFe_2O_4$ nucleation		
Notes: None		

ksnb_Fe2O4 boiling_growth_rate

boiling_growth_rate	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Boiling enhanced surface growth rate		
Notes: None		

D_mult diffusion_coefficient_prefactor

diffusion_coefficient_prefactor	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Diffusion coefficient prefactor for all species		
Notes: None		

D_Ni diffusion_coefficient

diffusion_coefficient	Floating-point Real Number	Optional
Units: cm^2/s (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Diffusion coefficient for Ni		
Notes: None		

D_Fe diffusion_coefficient

diffusion_coefficient	Floating-point Real Number	Optional
Units: cm^2/s (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Diffusion coefficient for Fe		
Notes: None		

D_BOH3 diffusion_coefficient

diffusion_coefficient	Floating-point Real Number	Optional
Units: cm^2/s (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Diffusion coefficient for Boric Acid		
Notes: None		

D_Li diffusion_coefficient

diffusion_coefficient	Floating-point Real Number	Optional
Units: cm^2/s (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Diffusion coefficient for Li		
Notes: None		

D_H2 diffusion_coefficient

diffusion_coefficient	Floating-point Real Number	Optional
Units: cm^2/s (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Diffusion coefficient for H_2		
Notes: None		

CRUD_porosity porosity

porosity	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): 0.7 (default), > 0		
Limitation(s): None		
Description: Initial porosity of CRUD layer		
Notes: None		

CRUD_solid_dens density

density	Floating-point Real Number	Optional
Units: g/cm^3 (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Density of solid $NiFe_2O_4$		
Notes: None		

CRUD_dep_frac fraction

fraction	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): 0.25 (default), ≥ 0		
Limitation(s): None		
Description: Fraction of the ^{10}B reaction rate applied to depletion		
Notes: None		

LTB_dissolve_scale LTB_dissolve_scale

LTB_dissolve_scale	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): 0.0 (default), ≥ 0, ≤ 1		
Limitation(s): None		
Description: Lithium tetraborate (LTB) dissolution parameter		
Notes: 0 to disable dissolution. 0.5 recommended for LTB dissolution		

chimney_gamma_l_c chimney_gamma_l_c

chimney_gamma_l_c	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Species chimney liquid carryover fraction		
Notes: None		

chimney_gamma_v_mult chimney_gamma_v_mult

chimney_gamma_v_mult	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Crud chimney vapor fraction multiplier		
Notes: None		

chimney_htc_model chimney_htc_model

chimney_htc_model	Integer	Optional
Units: N/A		
Applicable Value(s): 1 (default), 2		
Limitation(s): None		
Description: Boiling heat transfer model inside a chimney. A value of 1 uses a chimney boiling htc model of the form $h_{boil,chimney} = h + T_{sup}h_{lin}$, where h is the chimney htc given by the input **chimney_htc** with units of W/cm^2K, h_{lin} is the chimney htc given by the input **chimney_htc_lin** with units of W/cm^2K^2, and T_{sup} is the local superheat inside the crud layer in Kelvin. A value of 2 uses a conduction limited boiling heat transfer coefficient model of the form $h_{boil,chimney} = hf(k, c_n)$, where $f(k, c_n)$ is a function of the crud thermal conductivity, k, and c_n is the chimney surface density		
Notes: None		

chimney_htc htc

htc	Floating-point Real Number	Optional
Units: W/cm^2-K (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Heat transfer coefficient inside a chimney		
Notes: None		

chimney_htc_lin htc_lin

htc_lin	Floating-point Real Number	Optional
Units: W/cm^2-K^2 (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Linear-in-T heat transfer coefficient inside a chimney		
Notes: None		

chimney_dens dens

dens	Floating-point Real Number	Optional
Units: num/cm^2 (default)		
Applicable Value(s): > 0		
Limitation(s): None		

continued on next page...

`dens`, continued...

Description: Surface density of chimney
Notes: None

chimney_rad radius

`radius`	Floating-point Real Number	Optional
Units: cm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Radius of average chimney		
Notes: None		

chimney_vf void_fraction

`void_fraction`	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): > 0, < 1		
Limitation(s): None		
Description: Void fraction of steam exiting chimney		
Notes: None		

CRUD_therm_cond k_crud

`k_crud`	Floating-point Real Number	Optional
Units: W/cm-K (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Thermal conductivity of precipitate in CRUD		
Notes: None		

CRUD_heat_capacity Cp

`Cp`	Floating-point Real Number	Optional
Units: J/g-K (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Heat capacity for the CRUD skeleton		
Notes: Currently Cp is unused		

tke_scale factor

`factor`	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.0E-12 (default), > 0		

continued on next page...

`factor`, continued...

Limitation(s): None
Description: Scaling factor to convert from TKE to erosion
Notes: None

src_mult_A multiplier

`multiplier`	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), > 0		
Limitation(s): None		
Description: Multiplier for prefactor for source term model		
Notes: None		

src_mult_E multiplier

`multiplier`	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), > 0		
Limitation(s): None		
Description: Multiplier for activation energy for source term model		
Notes: None		

steam_generator_age age

`age`	Floating-point Real Number	Optional
Units: years (default)		
Applicable Value(s): 0.0 (default), ≥ 0		
Limitation(s): None		
Description: Initial age of the steam generator		
Notes: This is only needed for the first cycle simulated or for a steam generator replacement; default behavior is to retrieve these data from the restart file		

sg_mass sg_mass

`sg_mass`	Floating-point Real Number	Optional
Units: kg (default)		
Applicable Value(s): 0.0 (default), ≥ 0		
Limitation(s): None		
Description: Initial surface particulate mass on steam generator		
Notes: This is ignored in case of restart		

sg_mult multiplier

`multiplier`	Floating-point Real Number	Optional

continued on next page...

`multiplier`, continued...

Units: N/A
Applicable Value(s): 1.0 (default), ≥ 0
Limitation(s): None
Description: Multiplier on steam generator source term
Notes: This is required to scale the source term model to smaller reactor geometries (i.e., single assem)

mass_mult multiplier

`multiplier`	Floating-point Real Number	Optional
Units: N/A		
Applicable Value(s): 1.0 (default), ≥ 0		
Limitation(s): None		
Description: Multiplier on crud deposition mass term in the mass balance		
Notes: This is required to scale the source term model to smaller reactor geometries (i.e., single assem)		

piping_age age

`age`	Floating-point Real Number	Optional
Units: years (default)		
Applicable Value(s): 0.0 (default), ≥ 0		
Limitation(s): None		
Description: Initial age of the hot and cold leg		
Notes: This is only needed for the first cycle simulated , default behavior is to retrieve this data from the restart file		

chem_mass_bal option

`option`	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), 1		
Limitation(s): None		
Description: Option to select the mass balance model. The options are: 1. 0: no mass balance enabled (user must specify particulate $NiFe_2O_4$ concentration) 2. 1: Mass balance will be calculated by MAMBA		
Notes: None		

model_erosion option

`option`	Integer	Optional
Units: N/A		
Applicable Value(s): 2 (default), 0,1		
Limitation(s): None		

continued on next page...

option, continued...

Description: Option to select the erosion model. The options are: 1. 0: no crud erosion model 2. 1: calculate from shear so that average TKE is 0.1 J/kg 3. 2: use the Bradshaw model to calculate TKE from shear
Notes: None

nrmax nrmax

nrmax	Integer	Optional
Units: N/A		
Applicable Value(s): 200 (default), ≥ 1		
Limitation(s): None		
Description: Option to set maximum number of radial crud nodes		
Notes: None		

min_substeps min_substeps

min_substeps	Integer	Optional
Units: N/A		
Applicable Value(s): 4 (default), ≥ 1		
Limitation(s): None		
Description: Option to set minimum number of crud substeps per outer MAMBA step call		
Notes: None		

coupled_t_ltb_solve coupled_t_ltb_solve

coupled_t_ltb_solve	Integer	Optional
Units: N/A		
Applicable Value(s): 0 (default), ≥ 0		
Limitation(s): None		
Description: Option to set coupled temperature and LTB solver in MAMBA. Set to 0 for fast uncoupled solve. The uncoupled solve is the default. The uncoupled solve is appropriate for cases in which down-powers are not explicitly modeled, but instead are modeled using the crud_replenish_b10 card. Set to 1 for fully coupled solve. The fully coupled solve is more accurate for resolving fast crud transients which may occur in simulation of down-power events with small time step size between VERA state points		
Notes: None		

deltar deltar

deltar	Floating-point Real Number	Optional
Units: cm (default)		
Applicable Value(s): 0.0001 (default), ≥ 0		
Limitation(s): None		
Description: Radial mesh spacing in MAMBA		

continued on next page...

`deltar`, continued...

Notes: None

maxthick maxthick

`maxthick`	Floating-point Real Number	Optional
Units: cm (default)		
Applicable Value(s): 0.02 (default), ≥ 0		
Limitation(s): None		
Description: Max allowed crud thickness in MAMBA		
Notes: None		

li_table boron lithium

`lithium`	List of Two Floating-point Real Numbers	Optional
Units: ppm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Table of boron then lithium concentrations to define the lithium concentration based on boron concentration		
Notes: None		

5.15 BLOCK BISON

fuel_grain_radius_initial fuel_grain_radius_initial

`fuel_grain_radius_initial`	Float	Optional
Units: meters (default)		
Applicable Value(s): 2.5×10^{-6} (default), > 0		
Limitation(s): None		
Description: The initial grain radius of the fuel		
Notes: Defines the initial_condition parameter for the grain_radius object in the BISON AuxVariables block		

mechanical_contact_penalty mechanical_contact_penalty

`mechanical_contact_penalty`	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{7} (default), ≥ 0		
Limitation(s): None		
Description: The penalty applied within the fuel-cladding mechanical contact solver		
Notes: Defines the penalty parameter for the pellet_clad_mechanical object in the BISON Contact block		

thermal_contact_tol thermal_contact_tol

thermal_contact_tol	Float	Optional
Units: meters (default)		
Applicable Value(s): 1.0×10^{-6} (default), ≥ 0		
Limitation(s): None		
Description: The tangential distance to extend the edges of contact surfaces within the fuel-cladding thermal contact solver		
Notes: Defines the tangential_tolerance parameter for the thermal_contact object in the BISON Thermal-Contact block		

fuel_densification fuel_densification

fuel_densification	Float	Optional
Units: N/A		
Applicable Value(s): 0.005 (default), ≥ 0, ≤ 1		
Limitation(s): None		
Description: The fuel densification that will occur given as a fraction of its theoretical density		
Notes: Defines the total_densification parameter for the fuel_swelling object in the BISON Materials block		

temp_max_increment temp_max_increment

temp_max_increment	Float	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The maximum Newton increment when solving for the temperature of the fuel and cladding		
Notes: Defines the max_increment parameter for the limitT object in the BISON Dampers block		

linear_tol linear_tol

linear_tol	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-5} (default), > 0		
Limitation(s): None		
Description: The convergence tolerance applied to linear iterations		
Notes: Defines l_tol in the BISON Executioner block		

nonlinear_max_its nonlinear_max_its

nonlinear_max_its	Integer	Optional
Units: N/A		
Applicable Value(s): 25 (default), > 0		
Limitation(s): None		
Description: The maximum number of nonlinear iterations		
Notes: Defines nl_max_its in the BISON Executioner block		

nonlinear_rel_tol nonlinear_rel_tol

nonlinear_rel_tol	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-3} (default), > 0		
Limitation(s): None		
Description: The relative convergence tolerance applied to nonlinear iterations		
Notes: Defines nl_rel_tol in the BISON Executioner block		

nonlinear_abs_tol nonlinear_abs_tol

nonlinear_abs_tol	Float	Optional
Units: N/A		
Applicable Value(s): 1.0×10^{-10} (default), > 0		
Limitation(s): None		
Description: The absolute convergence tolerance applied to nonlinear iterations		
Notes: Defines nl_abs_tol in the BISON Executioner block		

lhr_axial_peaking_data_file lhr_axial_peaking_data_file

lhr_axial_peaking_data_file	Fixed Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The external axial peaking factors data file to be used for the BISON simulation		
Notes: Defines the data_file parameter for the axial_peaking_factors object in the BISON Functions block. Typically, the VERA power file is used, but this allows users to input a file if desired		

bc_temp_data_file bc_temp_data_file

bc_temp_data_file	Fixed Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The external boundary condition temperature data file to be used for the BISON simulation		
Notes: Defines the data_file parameter for the bc_temperature object in the BISON Functions block. Typically, VERA temperatures are used, but this allows users to input a file if desired		

bcs_plenumpressure_plenumpressure_initial_pressure rod_initial_pressure

rod_initial_pressure	Float	Optional
Units: Pascals (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Initial plenum pressure of (non-IFBA) fuel rods		
Notes: None		

bcs_plenumpressure_plenumpressure_initial_pressure_ifba rod_initial_pressure

rod_initial_pressure	Float	Optional
Units: Pascals (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Initial plenum pressure of IFBA-bearing fuel rods		
Notes: None		

bcs_plenumpressure_plenumpressure_startup_time startup_time

startup_time	Float	Optional
Units: seconds (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Time over which the simulation builds in the plenum pressure		
Notes: For numerical stability (if necessary)		

burnup_burnup_num_radial num_radial

num_radial	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 1		
Limitation(s): None		
Description: Number of radial mesh points in the burnup mesh		
Notes: None		

burnup_burnup_num_axial num_axial

num_axial	Float	Optional
Units: N/A		
Applicable Value(s): ≥ 1		
Limitation(s): None		
Description: Number of axial mesh points in the burnup mesh		
Notes: None		

executioner_start_time start_time

start_time	Float	Optional
Units: seconds (default)		
Applicable Value(s): Any float (+/-)		
Limitation(s): None		
Description: The starting time for the simulation		
Notes: Most standalone BISON cases use 0, but Tiamat typically uses -100 seconds		

executioner_dt dt

dt	Float	Optional
Units: seconds (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The initial timestep of the simulation		
Notes: Adaptive timestepping takes over from this immediately		

executioner_dtmin dtmin

dtmin	Float	Optional
Units: seconds (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The minimum timestep at which the adaptive timestepping will terminate the simulation		
Notes: None		

executioner_dtmax dtmax

dtmax	Float	Optional
Units: seconds (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The maximum timestep at which the adaptive timestepping will terminate the simulation		
Notes: None		

executioner_end_time end_time

end_time	Float	Optional
Units: seconds (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The end time of the simulation		
Notes: None		

globalparams_a_lower a_lower

a_lower	Float	Optional
Units: meters (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The lower bound of the active fuel		
Notes: None		

globalparams_a_upper a_upper

a_upper	Float	Optional
Units: meters (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The upper bound of the active fuel		
Notes: None		

globalparams_energy_per_fission energy_per_fission

energy_per_fission	Float	Optional
Units: Joules/fission (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The energy per fission		
Notes: None		

mesh_file meshfilename

meshfilename	Fixed Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The external mesh file for the simulation		
Notes: Typically, the internal mesh generator is used, but this allows the user to input a file if desired		

avg_lhr_data_file avg_lhr_data_file

avg_lhr_data_file	Fixed Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The external linear heat rate profile data file to be used for the BISON simulation		
Notes: Defines the data_file parameter for the linear_heat_rate_profile object in the BISON Functions block. Typically, the VERA power file is used, but this allows users to input a file if desired		

mesh_nx_p nradial_pellet

nradial_pellet	Float	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of radial elements in the fuel pellet		
Notes: None		

mesh_ny_p naxial_pellet

naxial_pellet	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of axial elements in the fuel pellet		
Notes: None		

mesh_nx_c nradial_clad

nradial_clad	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of radial elements in the cladding		
Notes: None		

mesh_ny_c naxial_clad

naxial_clad	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of axial elements in the cladding		
Notes: None		

mesh_bx_p radial_bias

radial_bias	Integer	Optional
Units: N/A		
Applicable Value(s): > 0,< 2		
Limitation(s): None		
Description: The biasing parameter for the fuel radial mesh		
Notes: This is used to enforce a nonuniform radial mesh to enhance accuracy		

mesh_clad_bot_gap_height bot_gap_height

bot_gap_height	Float	Optional
Units: meters (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The distance between the bottom of the fuel stack and the top of the lower clad region		
Notes: None		

outputs_file_base output_filename

output_filename	Fixed Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The filename of the output file for the simulation		
Notes: None		

fuel_pin_input_file_template fuel_inp_filename

fuel_inp_filenamee	Fixed Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The filename of the template for fuel rods		
Notes: None		

non_fuel_pin_input_file_template nonfuel_inp_filename

nonfuel_inp_filename	Fixed Character String	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The filename of the template for non-fuel rods		
Notes: None		

power_file power_filenames

power_filenames	List of Free Form Character Strings	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The filenames of the VERA-generated HDF5 files with power and temperature data		
Notes: None		

cycle_xml cycle_xml_filenames

cycle_xml_filenames	List of Free Form Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The filenames of the corresponding VERA cycle depletion XML files		
Notes: None		

shuffle_xml shuffle_xml_filenames

shuffle_xml_filenames	List of Free Form Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The filenames of the corresponding VERA shuffle XML files		
Notes: None		

only_cycle only_cycle

only_cycle	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The cycle index for which to generate BISON input files		
Notes: Otherwise, all cycles of files will be generated		

only_assemblies assembly_locations

assembly_locations	List of Free Form Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The list of assembly locations for which to generate BISON input files		
Notes: Otherwise, all assemblies will be generated		

mesh_type mesh_type

mesh_type	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): smeared_pellet (default), unit_test		
Limitation(s): None		
Description: The mesh type		
Notes: Used for Tiamat testing		

output_average_axial_values output_average_axial_values

output_average_axial_values	Boolean	Optional
Units: N/A		
Applicable Value(s): true, false		
Limitation(s): None		
Description: Logical governing whether or not to output average axial values		
Notes: None		

solve_type solve_type

solve_type	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): standalone, tiamat, tiamat_inline, temp_table		
Limitation(s): None		
Description: The solver scheme for the simulation		
Notes: Some options in the template are different, depending on the solve type		

bc_type bc_type

bc_type	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): bulk_cool (default), clad_outer		
Limitation(s): None		
Description: The boundary condition type used for the simulation		
Notes: Standalone BISON can use clad outer surface and bulk coolant temperature as the boundary condition. Tiamat exclusively uses clad outer surface temperature		

axial_shape axial_shape

axial_shape	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): midpoint (default), constant		
Limitation(s): None		
Description: Governs the axial shaping of the power and temperature variables input from VERA		
Notes: Some options in the template are different depending on the solve type		

fast_flux fast_flux

fast_flux	Boolean	Optional
Units: N/A		
Applicable Value(s): false (default), true		
Limitation(s): None		
Description: Logical governing whether or not to use pin clad fast flux from VERA		
Notes: None		

materials_fuel_relocation_relocation_activation1 activation_threshold

activation_threshold	Float	Optional
Units: W/m (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Threshold for the first level of relocation activation		
Notes: None		

auxkernels_fast_neutron_flux_factor flux_factor

flux_factor	Float	Optional
Units: n/m2-s per W/m (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Fast flux factor for approximating fast neutron flux from local power		
Notes: None		

materials_fuel_mech_model_creep model_creep

model_creep	Boolean	Optional
Units: N/A		
Applicable Value(s): true (default), false		
Limitation(s): None		
Description: Enables the fuel mechanics creep model		
Notes: None		

materials_fuel_relocation_burnup_relocation_stop relocation_stop

relocation_stop	Boolean	Optional
Units: fissions per initial metal atom (fima) (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Burnup at which the relocation model turns off		
Notes: None		

thermalcontact_roughness_fuel roughness_fuel

roughness_fuel	Boolean	Optional
Units: μm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Fuel roughness value		
Notes: None		

thermalcontact_roughness_clad roughness_clad

roughness_clad	Boolean	Optional
Units: μm (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Clad roughness value		
Notes: None		

power_ramp_times power_ramp_times

power_ramp_times	Float	Optional
Units: seconds (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Power ramp times before and after each cycle input file. These should be input as lists of numbers in pairs, with the first number corresponding to the beginning of a cycle, and the second corresponding to the end of a cycle		
Notes: None		

temp_ramp_times temp_ramp_times

temp_ramp_times	Float	Optional
Units: seconds (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Temperature ramp times before and after each cycle input file. These should be input as a list of numbers in pairs, with the first number corresponding to the beginning of a cycle, and the second corresponding to the end of a cycle		
Notes: None		

functions_coolant_pressure_ramp_x pressure_ramp_time_values **functions_coolant_pressure_ramp_x** pressure_ramp_time_values

pressure_ramp_time_values	Fixed Character String	Optional
Units: seconds (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Time values for coolant/system pressure ramp		
Notes: None		

functions_coolant_pressure_ramp_y pressure_ramp_pressure_values

pressure_ramp_pressure_values	Fixed Character String	Optional
Units: Pascals (default)		
Applicable Value(s):		
Limitation(s): None		
Description: Pressure values for coolant/system pressure ramp		
Notes: None		

5.16 BLOCK FAST

initial_plenum_pressure rod_initial_pressure

rod_initial_pressure	Float	Optional
Units: Pascals (default)		
Applicable Value(s): ≥ 0		

continued on next page...

rod_initial_pressure, continued...

Limitation(s): None
Description: Initial plenum pressure of (non-IFBA) fuel rods
Notes: None

initial_plenum_pressure_ifba rod_initial_pressure_ifba

rod_initial_pressure_ifba	Float	Optional
Units: Pascals (default)		
Applicable Value(s): ≥ 0		
Limitation(s): None		
Description: Initial plenum pressure of IFBA-bearing fuel rods		
Notes: None		

flux_to_power_ratio flux_to_power_ratio

flux_to_power_ratio	Float	Optional
Units: neutrons/m2/s per W/g of fuel (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The flux to power ratio		
Notes: None		

mesh_nr nradial_pellet

nradial_pellet	Float	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of fuel radial nodes		
Notes: None		

mesh_na naxial_cells

naxial_cells	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: This is the number of fuel axial nodes. FAST will automatically create a uniformly sized mesh		
Notes: None		

mesh_max_deltaz mesh_max_deltaz

mesh_max_deltaz	Float	Optional

continued on next page...

mesh_max_deltaz, continued...

Units: meters (default)
Applicable Value(s): > 0
Limitation(s): None
Description: This is the maximum axial mesh size. This will refine the VERA axial grid so that all the axial cells are less than or equal to mesh_max_deltaz
Notes: None

mesh_nc nradial_clad

nradial_clad	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The number of clad radial nodes		
Notes: None		

max_deltat max_deltat

max_deltat	Float	Optional
Units: seconds (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: This is the maximum size for timesteps. Otherwise, the default timestep will be the time between states		
Notes: None		

fuel_pin_input_file_template fuel_inp_filename

fuel_inp_filenamee	Fixed Character String	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The filename of the template for fuel rods		
Notes: None		

power_file power_filenames

power_filenames	List of Free Form Character Strings	Required
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The filenames of the VERA-generated HDF5 files with power and temperature data listed in sequential order		
Notes: None		

cycle_xml cycle_xml_filenames

cycle_xml_filenames	List of Free Form Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The filenames of the corresponding VERA cycle depletion XML files in sequential order		
Notes: None		

shuffle_xml shuffle_xml_filenames

shuffle_xml_filenames	List of Free Form Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: These are the filenames of the corresponding VERA shuffle XML files. There should be a shuffle file for the beginning of each cycle after the first		
Notes: None		

only_cycle only_cycle

only_cycle	Integer	Optional
Units: N/A		
Applicable Value(s): > 0		
Limitation(s): None		
Description: The cycle index for which to generate FAST input files		
Notes: Otherwise, all cycles of files will be generated		

only_assemblies assembly_locations

assembly_locations	List of Free Form Character Strings	Optional
Units: N/A		
Applicable Value(s):		
Limitation(s): None		
Description: The list of assembly locations for which to generate FAST input files		
Notes: Otherwise, all assemblies will be generated		

solve_type solve_type

solve_type	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): standalone		
Limitation(s): None		

continued on next page...

`solve_type`, continued...

Description: The solver scheme for the simulation
Notes: None

bc_type bc_type

`bc_type`	Fixed Character String	Optional
Units: N/A		
Applicable Value(s): bulk_cool (default), clad_outer		
Limitation(s): None		
Description: The boundary condition type used for the simulation		
Notes: Standalone FAST can use clad outer surface and bulk coolant temperature as the boundary condition		

thermalcontact_roughness_fuel roughness_fuel

`roughness_fuel`	Float	Optional
Units: m (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Fuel roughness value		
Notes: None		

thermalcontact_roughness_clad roughness_clad

`roughness_clad`	Float	Optional
Units: m (default)		
Applicable Value(s): > 0		
Limitation(s): None		
Description: Clad roughness value		
Notes: None		

5.17 BLOCK RUN

email list_of_emails

`email`	Character String	Optional
Units: N/A		
Applicable Value(s): Default email is the users system email (default)		
Limitation(s): None		
Description: This is the email address that is used to inform the user of job status. A list of emails can be input by comma separating the email addresses		
Notes: None		

exe_mode exe_mode

exe_mode	Character String	Optional
Units: N/A		
Applicable Value(s): th		
Limitation(s): None		
Description: If given an input of th, then VERARun will execute multistate_cobra instead of MPACT.exe		
Notes: None		

pmem memory per processor

pmem	Floating-point Number	Optional
Units: GB (default)		
Applicable Value(s): System memory per processor (default), > 0.0		
Limitation(s): None		
Description: Memory per processor		
Notes: None		

ppn processors per node

ppn	Integer	Optional
Units: N/A		
Applicable Value(s): System processors per node (default), > 0		
Limitation(s): None		
Description: Number of processors that will be used per node		
Notes: None		

nprocs number of processors

nprocs	Integer	Optional
Units: N/A		
Applicable Value(s): Total number of system processors (default), > 0		
Limitation(s): None		
Description: Total number of processors that will be used		
Notes: None		

walltime maximum expected runtime

walltime	Floating-point Number	Optional
Units: hours (default)		
Applicable Value(s): 24 hours (default), > 0.0		
Limitation(s): None		
Description: The walltime that is used for pbs submission		
Notes: None		

6. EXAMPLES

This chapter includes several input examples. Additional examples can be found in the VERAIn Git repository.

6.1 EXAMPLE 1 – FULL CORE

The first example is a complete input for a full-core problem. This problem is Problem 7 of the VERA Core Physics Benchmark Progression Problem Specifications and is based upon the publicly available description of the Watts Bar reactors.

More information on the CASL Progression Benchmark Problems can be found in the following CASL report:

- A. Godfrey, "VERA Core Physics Benchmark Progression Problem Specifications," CASL Technical Report: CASL-U-2012-0131-004, August 2014.

More details on Problem 7 can be found in the following CASL report:

- "Demonstration and Neutronics Coupled to Thermal-Hydraulics for a Full-Core Problem using VERA", CASL Technical Report: CASL-U-2013-0196-000, December 2013.

```
!
!  Sample Test case for Problem 7 (Full-Core HFP)
!
[CASEID]
  title 'CASL Progression Problem 7 - Watts Bar Unit 1 Cycle 1 - Public'

[STATE]
  power  100.0          ! % of rated power
  flow   100.0          ! % of rated flow
  pressure 2250.0       ! pressure (psia)
  feedback on

  tinlet 565.0 K        ! inlet temperature
  tfuel  900.0 K        ! typical HFP value
  boron  1285           ! ppmB
  modden 0.743          ! g/cc
  sym qtr

  rodbank SA 230
          SB 230
          SC 230
          SD 230
           A 230
           B 230
           C 230
           D 167

[CORE]
  size 15               ! assemblies across core
  rated 3411 131.68     ! rated power and flow - MW, Mlbs/hr
  apitch 21.5           ! assembly pitch (cm)
  height 406.337        ! assembly height (cm)

  core_shape
    0 0 0 0 1 1 1 1 1 1 1 0 0 0 0
    0 0 1 1 1 1 1 1 1 1 1 1 1 0 0
    0 1 1 1 1 1 1 1 1 1 1 1 1 1 0
    0 1 1 1 1 1 1 1 1 1 1 1 1 1 0
    1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
    1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
    1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
    1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
    1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
    1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
    1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
    0 1 1 1 1 1 1 1 1 1 1 1 1 1 0
    0 1 1 1 1 1 1 1 1 1 1 1 1 1 0
    0 0 1 1 1 1 1 1 1 1 1 1 1 0 0
    0 0 0 0 1 1 1 1 1 1 1 0 0 0 0

  assm_map
    1
    2 1
    1 2 1
```

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

insert_map
   -
  20  -
   - 24  -
  20  - 20  -
   - 20  - 20  -
  20  - 16  - 24 12
   - 24  - 16  -  -
  12  -  8  -

crd_map
  1
  - -
  1 - 1
  - - - 1
  1 - - - 1
  - 1 - 1 - -
  1 - 1 - 1 -
  - - - -

crd_bank
  D  -  A  -  D  -  C  -
  -  -  -  -  - SB  -  -
  A  -  C  -  -  -  B  -
  -  -  -  A  - SC  -  -
  D  -  -  -  D  - SA
  - SB  - SD  -  -  -
  C  -  B  - SA  -
  -  -  -  -

det_map
          1 - - 1 - - -
      1 - - - 1 - - 1 - 1 -
    - 1 - 1 - - 1 - - - - - 1
    - - - - 1 - - 1 - - 1 - -
  1 - - - 1 - - 1 - - 1 - - - 1
  - - - - 1 - 1 - - - - 1 - - -
  - 1 - - - - - - 1 - 1 - - - 1
  1 - 1 - 1 - 1 - - 1 - 1 1 1 -
  - - - 1 - - 1 - - 1 - - 1 - -
  1 - 1 - - 1 - 1 - - - - - 1 -
  - - - - 1 - - - 1 - 1 - 1 - -
    1 1 - - - - 1 - - - - - -
    - - - - - - 1 - 1 - 1 - 1
      1 - - 1 - 1 - - - - -
          - - 1 - - 1 -

baffle ss 0.19 2.85       ! baffle material, gap, and thickness (cm)
```

```
  vessel mod 219.71 cs 241.70

  lower_plate ss  5.0 0.5   ! mat, thickness, vol frac
  upper_plate ss  7.6 0.5   ! mat, thickness, vol frac

  lower_ref  mod 20.0 1.0   ! mat, thickness, vol frac
  upper_ref  mod 20.0 1.0   ! mat, thickness, vol frac

  xlabel  R P N M L K J H G  F  E  D  C  B  A
  ylabel  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

  mat he     0.0001786
  mat inc    8.19
  mat ss     8.0
  mat zirc   6.56 zirc4

[ASSEMBLY]
  title "Westinghouse 17x17 Assembly"
  npin 17                ! number of pins across assembly
  ppitch 1.260           ! pin pitch (cm)

  fuel U21 10.257 94.5 / 2.110
  fuel U26 10.257 94.5 / 2.619
  fuel U31 10.257 94.5 / 3.100

  cell 1     0.4096 0.418 0.475 / U21 he zirc
  cell 2     0.4096 0.418 0.475 / U26 he zirc
  cell 3     0.4096 0.418 0.475 / U31 he zirc
  cell 4            0.561 0.602 / mod    zirc       ! guide/instrument tube
  cell 5            0.418 0.475 /     he zirc       ! plenum

  rodmap LAT21
       4
       1 1
       1 1 1
       4 1 1 4
       1 1 1 1 1
       1 1 1 1 1 4
       4 1 1 4 1 1 1
       1 1 1 1 1 1 1 1
       1 1 1 1 1 1 1 1 1

  rodmap LAT26
       4
       2 2
       2 2 2
       4 2 2 4
       2 2 2 2 2
       2 2 2 2 2 4
       4 2 2 4 2 2 2
       2 2 2 2 2 2 2 2
       2 2 2 2 2 2 2 2 2

  rodmap LAT31
```

```
          4
          3 3
          3 3 3
          4 3 3 4
          3 3 3 3 3
          3 3 3 3 3 4
          4 3 3 4 3 3 3
          3 3 3 3 3 3 3 3
          3 3 3 3 3 3 3 3 3

    rodmap PLEN
          4
          5 5
          5 5 5
          4 5 5 4
          5 5 5 5 5
          5 5 5 5 5 4
          4 5 5 4 5 5 5
          5 5 5 5 5 5 5 5
          5 5 5 5 5 5 5 5 5

! define three assemblies with labels 1, 2, 3

    axial  1  11.951 LAT21 377.711 PLEN 393.711
    axial  2  11.951 LAT26 377.711 PLEN 393.711
    axial  3  11.951 LAT31 377.711 PLEN 393.711

    grid END inc  1017 3.866   ! grid mass (g) and thickness (cm)
    grid MID zirc  875 3.810

    grid_axial                 ! axial grid positions - midpoints (cm)
        END  13.884
        MID  75.2
        MID 127.4
        MID 179.6
        MID 231.8
        MID 284.0
        MID 336.2
        END 388.2

    lower_nozzle  ss 6.053 6250.0  ! mat, height, mass (g)
    upper_nozzle  ss 8.827 6250.0  ! mat, height, mass (g)

  [INSERT]
    title "Pyrex"
    npin 17

    mat pyrx1 2.25 pyrex-vera

    cell 1  0.214 0.231 0.241 0.427 0.437 0.484 / he ss he pyrx1 he ss

    rodmap  PY8
       -
       - -
```

```
      - - -
      1 - - -
      - - - - -
      - - - - - 1
      - - - - - - - -
      - - - - - - - - -
      - - - - - - - - - -

   rodmap  PY12
      -
      - -
      - - -
      1 - - -
      - - - - -
      - - - - - -
      - - - 1 - - -
      - - - - - - - - -
      - - - - - - - - - -

   rodmap  PY16
      -
      - -
      - - -
      1 - - -
      - - - - -
      - - - - - 1
      - - - 1 - - -
      - - - - - - - - -
      - - - - - - - - - -

   rodmap  PY20
      -
      - -
      - - -
      1 - - -
      - - - - -
      - - - - - 1
      1 - - 1 - - -
      - - - - - - - - -
      - - - - - - - - - -

   rodmap  PY24
      -
      - -
      - - -
      1 - - 1
      - - - - -
      - - - - - 1
      1 - - 1 - - -
      - - - - - - - - -
      - - - - - - - - - -

! define 5 insert types with labels 8, 12, 16, 20, and 24
```

```
  axial   8  15.761 PY8  376.441
  axial  12  15.761 PY12 376.441
  axial  16  15.761 PY16 376.441
  axial  20  15.761 PY20 376.441
  axial  24  15.761 PY24 376.441

[CONTROL]
  title "B4C with AIC tips"
  npin 17
  stroke  365.125 230     ! approx for 1.5875 step sizes and 230 max stroke

  mat aic 10.2
  mat b4c 1.76

  cell 1  0.382 0.386 0.484 / aic he ss
  cell 2  0.373 0.386 0.484 / b4c he ss

  rodmap AIC
     -
     - -
     - - -
     1 - - 1
     - - - - -
     - - - - - 1
     1 - - 1 - - -
     - - - - - - - -
     - - - - - - - - -

  rodmap B4C
     -
     - -
     - - -
     2 - - 2
     - - - - -
     - - - - - 2
     2 - - 2 - - -
     - - - - - - - -
     - - - - - - - - -

  axial  1    17.031
         AIC 118.631
         B4C 377.711

[DETECTOR]
  title "Incore instrument thimble"
  npin 17

  mat he 0.0001786
  mat ss 8.0

  cell 1  0.258 0.382 / he ss

  rodmap  LAT
     1
```

```
      - -
      - - -
      - - - -
      - - - - -
      - - - - - -
      - - - - - - -
      - - - - - - - -
      - - - - - - - - -

  axial 1  0.0 LAT 406.337

[EDITS]
  axial_edit_bounds
       11.951   15.817   24.028   32.239   40.45
       48.662   56.873   65.084   73.295   77.105
       85.17    93.235  101.3    109.365  117.43
      125.495  129.305  137.37   145.435  153.5
      161.565  169.63   177.695  181.505  189.57
      197.635  205.7    213.765  221.83   229.895
      233.705  241.77   249.835  257.9    265.965
      274.03   282.095  285.905  293.97   302.035
      310.1    318.165  326.23   334.295  338.105
      346.0262 353.9474 361.8686 369.7898 377.711

[COBRATF]

[COUPLING]
```

6.2 EXAMPLE 2 – SINGLE ASSEMBLY

The second example is a partial input for a single-assembly with T/H feedback. This problem is Problem 6 of the VERA Core Physics Benchmark Progression Problem Specifications. See:

- A. Godfrey, "VERA Core Physics Benchmark Progression Problem Specifications," CASL Technical Report: CASL-U-2012-0131-004, August 2014.

A single assembly is defined by creating a core with one assembly in it, as described in the small-core geometry discussion in Section 2.2.5.

This input is also used to demonstrate the modular structure of the input. The [ASSEMBLY], [EDITS], [COBRATF], and [COUPLING] blocks are identical to Example 1, and they show how blocks can be re-used in different input decks. These blocks are not included here, but they can be copied directly from the first example problem if the user wishes to run this problem.

```
[CASEID]
  title 'CASL Benchmark Progression Problem 6'
!=====================================================================
!  Sample input for Problem 6 (Single-assembly with T/H feedback)
!=====================================================================

[STATE]
  power  100.0         ! %
  tinlet 559.0 F       !
  boron  1300          ! ppmB
  pressure 2250        ! psia

  feedback on
  sym full

[CORE]
  size 1               ! 1x1 single-assembly

 ! The rated power and flow are scaled down for a single-assembly
  rated 17.67   0.6824  ! rated power and flow (MW, Mlbs/hr)

  apitch 21.5          ! assembly pitch (cm)
  height 406.328       ! core height (cm)

  core_shape
    1                  ! core map with a single assembly

  assm_map
    A1                 ! name of assembly

  lower_plate ss 5.0 0.5    ! material, thickness (cm), vol frac
  upper_plate ss 7.6 0.5    ! material, thickness (cm), vol frac
  lower_ref   mod 26.0 1.0  ! material, thickness (cm), vol frac
  upper_ref   mod 25.0 1.0  ! material, thickness (cm), vol frac

  bc_rad reflecting         ! radial boundary condition

! Materials defined in the [CORE] block are global and can be accessed
! from any assembly, insert, etc.

  mat he     0.0001786
  mat inc    8.19
  mat ss     8.0
  mat zirc   6.56 zirc4

include  assembly.inc   ! Include [ASSEMBLY] block from Example 1
include  edits.inc      ! Include [EDITS]    block from Example 1
include  cobratf.inc    ! Include [COBRATF]  block from Example 1
```

6.3 EXAMPLE 3 – 2D LATTICE GEOMETRY

The third example is a complete input for a 2D lattice. This problem is Problem 2A of the VERA Core Physics Benchmark Progression Problem Specifications. See:

- A. Godfrey, "VERA Core Physics Benchmark Progression Problem Specifications," CASL Technical Report: CASL-U-2012-0131-004, August 2014.

A single assembly is defined by creating a core with one assembly in it, as described in the small-core geometry description in Section 2.2.5.

The 2D lattice is defined by specifying an *axial* card with one level and defining reflective boundary conditions on the top and bottom of the core with the *bc_top* and *bc_bot* input cards.

This example problem also shows how multiple assembly, insert, and control types can be defined by using multiple *axial* cards in a single input block.

```
[CASEID]
  title 'CASL AMA Benchmark Problem 2A - Fuel Lattice - Public'

[STATE]
  power 0.0                ! %
  tinlet 557.33 F          !
  tfuel  565 K             !
  modden 0.743             ! g/cc
  boron 1300               ! ppm
  rodbank A 1              ! rod fully withdrawn
  sym qtr

[CORE]
  size 1
  apitch 21.50
  height 1.0
  rated 0.01 0.01

  core_shape
    1

  assm_map
    ASSY

  insert_map
    -

  crd_map
    AIC

  crd_bank
    A

  bc_rad reflecting
  bc_top reflecting   ! specify top reflective boundary conditions
  bc_bot reflecting   ! specify bottom reflective boundary conditions

[ASSEMBLY]
  npin 17
  ppitch 1.26

! material definitions in an ASSEMBLY block only have scope in this block

  fuel U31 10.257 94.5 / 3.1
  mat he   0.000176
  mat zirc 6.56 zirc4

  cell 1 0.4096 0.418 0.475 / U31 he zirc
  cell 2        0.561 0.602 / mod zirc

  lattice LAT
    2
    1 1
```

```
      1 1 1
      2 1 1 2
      1 1 1 1 1
      1 1 1 1 1 2
      2 1 1 2 1 1 1
      1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1 1

    axial ASSY 0.0 LAT 1.0

  [INSERT]
    title "Pyrex"
    npin 17

  ! material definitions in an INSERT block only have scope in this block

    mat he     0.0001786
    mat pyrx1  2.25 pyrex-vera
    mat ss     8.0

    cell 1  0.214 0.231 0.241 0.427 0.437 0.484 / he ss he pyrx1 he ss

! define multiple inserts corresponding to 8, 12, 16, 20, and 24 fingers

    lattice  LAT8
       -
       - -
       - - -
       1 - - -
       - - - - -
       - - - - - 1
       - - - - - - -
       - - - - - - - -
       - - - - - - - - -

    lattice  LAT12
       -
       - -
       - - -
       1 - - -
       - - - - -
       - - - - - -
       - - - 1 - - -
       - - - - - - - -
       - - - - - - - - -

    lattice  LAT16
       -
       - -
       - - -
       1 - - -
       - - - - -
       - - - - - 1
       - - - 1 - - -
```

```
    - - - - - - - -
    - - - - - - - - -

  lattice  LAT20
    -
    - -
    - - -
    1 - - -
    - - - - -
    - - - - - 1
    1 - - 1 - - -
    - - - - - - - -
    - - - - - - - - -

  lattice  LAT24
    -
    - -
    - - -
    1 - - 1
    - - - - -
    - - - - - 1
    1 - - 1 - - -
    - - - - - - - -
    - - - - - - - - -

! multiple INSERT types can be defined by defining separate axial cards

  axial PY8   0.0 LAT8  1.0
  axial PY12  0.0 LAT12 1.0
  axial PY16  0.0 LAT16 1.0
  axial PY20  0.0 LAT20 1.0
  axial PY24  0.0 LAT24 1.0

[CONTROL]
  title "B4C and AIC RCCAs"
  npin 17
  stroke  1.0 1           ! 1 step for in/out

! material definitions in a CONTROL block only have scope in this block

  mat he     0.0001786
  mat ss     8.0
  mat aic   10.2
  mat b4c    1.76

  cell 1  0.382 0.386 0.484 / aic he ss
  cell 2  0.373 0.386 0.484 / b4c he ss

  lattice LAT_AIC
    -
    - -
    - - -
    1 - - 1
    - - - - -
```

```
    - - - - - 1
    1 - - 1 - - -
    - - - - - - - -
    - - - - - - - - -

  lattice LAT_B4C
    -
    - -
    - - -
    2 - - 2
    - - - - -
    - - - - - 2
    2 - - 2 - - -
    - - - - - - - -
    - - - - - - - - -

  axial AIC  0.0 LAT_AIC 1.0
  axial B4C  0.0 LAT_B4C 1.0

[MPACT]

  ! include SHIFT and/or MPACT block here
```

7. VERARUN

This chapter describes running cases with the VERARun script. VERARun is the driver script that runs the input processor and corresponding VERA component codes. VERARun also submits the job to the parallel job queue.

7.1 RUNNING A CASE

VERARun is run by specifying `verarun` on the command line, followed by the name of the input file. There are additional command line options that are shown below.

```
--> verarun <input file>
```

For example, if you input deck is called "2a.inp", you would enter:

```
--> verarun 2a
```

To see what versions of VERA are available, use the -l option:

```
--> verarun -l
Available VERA versions (newest to oldest, * for default):
  VERA_4.1
  VERA_4.2 *
```

To use a different VERA version, use the -v option:

```
--> verarun -v VERA_4.2 file.inp
```

To use a development version of VERA (not usually recommended), use the –devs option:

```
--> verarun --devs -v VERA_4.3RC2 file.inp
```

To see additional verarun command line options, just execute verarun with no other command line input. To see detailed help and customizable environment variables use the -h option. Doing so will return the following:

```
usage: verarun [--devs] [-x] [-e email_addr] [-h] [-c config_file]
               [-N job_name] [-l] [-n nprocs] [-O] [--ppn cpus_per_node]
               [-m mem_per_process] [-p project] [-q queue] [-s subdir]
               [-d vera_install_dir] [-v vera_version] [--verbose] [-W]
               [-w job_id] [-t walltime] [--chain] [--debug] [--hostname host]
               [-r {overwrite,readwrite}]
               [--vera-installs-dir vera_installs_dir]
               [input_path [input_path ...]]
```

```
Creates and optionally submits machine-specific VERA jobs.

positional arguments:
  input_path            path to VERA input (.inp) or XML (.xml) files

optional arguments:
  --devs, --allow-devs  override VERA_PROD_VERSIONS and allow development VERA
                        versions, implies -l
  -x, --dry-run         dry run only, create but don't execute the PBS script
  -e email_addr, --email-addrs email_addr
                        comma-delimited list of email addresses to notify of
                        job completion, defaults to ${USER}@$(hostname)
  -h, --help            print detailed help
  -c config_file, --host-config-file config_file
                        override host configuration file, supercedes
                        --hostname and --vera-installs-dir
  -N job_name, --job-name job_name
                        name for the PBS job
  -l, --list-vera-versions
                        list available VERA versions
  -n nprocs, --np nprocs, --nprocs nprocs, --num-procs nprocs
                        total number of processors need for the MPACT run
                        (mpiexec -np param), defaults to value computed from
                        input
  -O, --output-job-name
                        print the job id to stdout
  --ppn cpus_per_node, --pnode cpus_per_node
                        specify processors per node, by default this is
                        calculated
  -m mem_per_process, --pmem mem_per_process, --proc-memory mem_per_process
                        specify memory required per processor in GB, defaults
                        to undefined
  -p project, --project project
                        optional project or account to specify for the job,
                        overriding any default, where a value of "none" omits
                        a project
  -q queue, --partition queue, --queue queue
                        Torque queue or Slurm partition
  -s subdir, --subdir subdir
                        create subdir, a value of "." specifies automatically
                        generated subdir name
  -d vera_install_dir, --vera-dir vera_install_dir
                        path to VERA installation, superceding --vera-
                        installs-dir, --vera-version, and the host
                        configuration
  -v vera_version, --vera-version vera_version
                        name of VERA version to use
  --verbose             turn on verbose messaging
  -W                    wait on job last submitted via verarun, overrides -w
  -w job_id, --wait-job-id job_id
                        ID of job which must complete before starting this job
  -t walltime, --wall-time walltime
                        wallclock execution time in floating point hours,
                        defaults to 24.0
```

```
advanced arguments:
  --chain, --chain-jobs
                        each job depends on its predecessor
  --debug               debug mode
  --hostname host       force the hostname
  -r {overwrite,readwrite}, --restart {overwrite,readwrite}
                        optional restart mode
  --vera-installs-dir vera_installs_dir
                        path to vera_installs directory containing VERA
                        versions

Version 1.11

Verarun uses machine-specific characteristics to determine batch system
directives applied to a job script template.  Additional template parameters,
such as which models and utilities to execute, are determined from the VERA
input file (.inp).  Configurations are provided for machines commonly in the
VERA program.  In order to port 'verarun' to another environment it will be
necessary to create a machine spec file.

Execution
---------
There are two verarun usage modes:

* If the '-l' command-line argument is specified, available VERA installations
  are listed.  If '--verbose' is also specified, required files that are
  missing in each vera_installs_dir subdir are listed.

* If one or more VERA input files (.inp) are specified, a job script is created
  based on each input file.  Note pre-processed VERA input XML (.xml) may be
  also be specified.  If no file extension is given '.inp' is assumed.

By default the current directory will be used for all processing.
Alternatively, you may specify that a subdir in the current directory be
created and used as the working directory via the '-s' command-line argument.
Specifying a directory value of '.' will tell verarun to create a subdir with
name {input_name}_{datetime}, where {input_name] is the name of the input file
without the '.inp' or '.xml' extension, and {datetime} is the ISO 8601 extended
date format: yyyy-mm-ddThh:mm:ss.

A job script named with a '.pbs' extension is created in the working directory.
If the '-x' argument is not specified, the job is also submitted.  Output from
the job will be in a file with extension '.stdout' in the working directory.

Machine Specifications
----------------------
Machine specifications are stored in JSON ('.json') files under the verarun
Python package installation directory:

  ../lib/python2.7/site-packages/verarun-<version>-py2.7.egg/verarun/config/

It is also possible to specify a configuration file with the '-c' command-line
argument.
```

```
Job Script Template
-------------------
The Job template is a Bash script named 'vera.pbs' stored under the verarun
installation directory:

  ../lib/python2.7/site-packages/verarun-<version>-py2.7.egg/verarun/job/

Parameter values are substituted by name for curly-brace expressions in the
template (.e.g., {job_name}).

Environment Variables
---------------------
There are environment variables which can be set to override behavior.

* PBS_EMAIL: sets the email address to be used for PBS notifications,
  overridden by the '-e' (or '--email-addrs') parameter.

* PBS_PROJECT: sets the project for the job, overriding any default set in
  the host configuration but overridden by the '-p' (or '--project') parameter.

* VERA_BUILDS: path to the directory containing VERA builds or installs,
  overriding the "$(hostname).json" file but overridden by the
  '--vera-installs-dir' parameter.  May also be specified as
  VERA_BUILDS_DIR or VERA_INSTALLS_DIR.

* VERA_BUILDS_DIR: alternative name for VERA_BUILDS, where VERA_BUILDS has
  priority.

* VERA_DEFAULT_VERSION: names the default version of VERA to use, overriding
  what is specified in the host configuration file but overridden by the '-v'
  (or '--vera-version') parameter.

* VERA_INSTALLS_DIR: alternative name for VERA_BUILDS or VERA_BUILDS_DIR,
  priority order being VERA_BUILDS, VERA_BUILDS_DIR, then VERA_INSTALLS_DIR

* VERA_PROD_VERSIONS: comma-delimited list of allowed production VERA versions.

* VERA_QSUB: custom 'qsub' command to execute for hosts running Torque or
  PBS Pro.

* VERA_SBATCH: custom 'sbatch' command to execute for hosts running SLURM.

* VERA_SUMMARY_EMAIL:  if set and not blank, the 'case.sum' file is emailed
  to the user when MPACT completes.

* VERARUN_HOST_CONFIG_FILE: path to a host configuration file to use,
  overriding the distributed "$(hostname).json" file but overridden by
  the '-c' parameter.
```

Note that verarun will create a .pbs file that can also be modified and submitted manually using qsub. After job submittal, the job is managed by the typical queueing system commands (qdel, qhold, qrls, etc.)

With VERARun 1.11 and beyond, you can receive emails summarizing your job when it completes success-

fully. Depending on the font of your email client, this summary can be more or less readable. In Outlook, you can force plain text emails to use a certain font (such as courier new) by going to Options -> Mail -> Stationary and Fonts -> Composing and reading plain test messages....

To get information on past jobs submitted with verarun, use the verastat command. This will provide a listing of job ID, date/time, and file location. To get info on a specific job, use:

```
--> verastat job_id
```

For additional questions or support, please contact vera-support@ornl.gov.

7.2 VERARUN OUTPUT

Upon completion of a VERA job, several output files may be created depending on the code options used. Some typical outputs include:

- VERAIn XML file. This file is written upon the successful completion of VERAIn.
- VERA HDF output file. This is a binary file with results that can be visualized in VERAView, or post-processed with user utility codes.
- MPACT output file. This file is written upon the successful completion of MPACT (if applicable).
- MPACT log File. This file is written upon the successful completion of MPACT (if applicable).
- MPACT summary File. This file is written upon the successful completion of MPACT (if applicable).
- Standard output file. This file is a log of all output written to the standard output.
- Standard error file. This file is a log of all output written to the standard error file.

7.3 INPUT ERRORS

If the `verarun` command does not work, the user should make sure that it is in the path. The user may need to consult with the system administrator for the correct path.

The next step when looking for an error is to look at the standard error file. If the job ran successfully, the size of this file will be zero.

If there were any errors in the input processor (VERAIn), the errors will be written to the standard error file. Common errors from the input processor include:

1. Invalid keywords
2. Invalid map sizes
3. Invalid input options

If the input processor works correctly, an error may still occur in one of the VERA component codes. The user should look at the error message and consult the user manual for the component code.

8. ACKNOWLEDGMENTS

This research was supported by the US Department of Energy and the Nuclear Energy Advanced Modeling and Simulation Program.