LA-UR-10-06235

Approved for public release;
distribution is unlimited.

Title:	MCNP5-1.60 Release Notes
Author(s):	Forrest Brown, Brian Kiedrowski, Jeffrey Bull
Intended for:	MCNP documentation and release

Form 836 (7/06)

MCNP5 1.60 Release Notes

Forrest Brown, Brian Kiedrowski, Jeffrey Bull

Monte Carlo Codes, XCP-3
X-Computational Physics Division
Los Alamos National Laboratory

I. Introduction

The latest release of the MCNP5 [1] Monte Carlo code is designated MCNP5-1.60. This report serves as a summary of updates in going from the RSICC release of MCNP-1.51 [2,3,4] to the current version MCNP5-1.60. Users should obtain MCNP5-1.60 directly from RSICC or the OECD/NEA Data Bank (*rsicc.ornl.gov* or *www.nea.fr/databank*).

This release includes many minor code modifications to fix reported bugs, output formats, error checking, and other difficulties present with previous versions of MCNP. In some cases, the problems that were fixed date back to the 1990s, but were only recently reported and fixed. In addition, there are enhancements to several MCNP capabilities: maximum number of cells, surfaces, materials, and tallies; isotopic reaction rates for mesh tallies; and adjoint-weighting for computing effective lifetimes and delayed neutron parameters. It should be noted that no errors were found that would affect the code results for basic criticality calculations. In nearly all cases, the bug fixes addressed problems with infrequently-used combinations of code options. All previously existing code capabilities have been preserved, including physics options, geometry, tallying, plotting, cross-section handling, etc. Tally results from MCNP5-1.60 are expected to match the tally results of problems that can be run with the previous MCNP5-1.51, except where bugs were discovered and fixed.

The verification of MCNP5-1.60 is reported in Reference [5], and a supplement to the MCNP manual is provided in Reference [6].

The following sections list the new features, bugs that were fixed, work in progress, system and compiler information, and installation instructions.

II. New Features

1. Adjoint-weighted Tallies for Point Kinetics Parameters

Many quantities in reactor physics such as reactor kinetics parameters require adjoint or importance weighting. The adjoint response function used by MCNP5 for criticality calculations is the iterated fission probability. Lumped average parameters describing the kinetics for a reactor system can be derived from the linear Boltzmann equation, with the resulting parameters involving ratios of adjoint-weighted reaction rates. In particular, the neutron generation time Λ and the effective delayed neutron fraction β_{eff} are expressed as ratios of adjoint-weighted integrals. For the first time, MCNP5-1.60 includes the capability to generate adjoint-weighted reactor kinetics parameters from continuous-energy Monte Carlo. The theory and MCNP5 input instructions for this capability are described in Reference [6], and the verification suite associated with this feature is described in Reference [5].

2. Mesh Tallies for Isotopic Reaction Rates

MCNP5 has always had the capability to perform mesh tallies for fluxes and material reaction rates on an arbitrary, user-specified mesh that is independent of the actual problem geometry [1]. An important previous enhancement permitted users to specify a material number of *0* on the associated FM card, which caused the mesh tally routines to use the actual material number densities in the problem, which were dependent on a particle's actual location in the problem geometry. This important capability to "wild-card" the number densities was previously limited to materials, and could not be used to obtain individual isotopic reaction rates. With MCNP5-1.60, the use of material number *0* as a wild-card on the FM cards for mesh tallies has been extended to handle isotopic mesh tallies as well, so that users do not need to enter number densities directly for isotopic mesh tallies; the mesh tally routines can now look up the actual problem materials and use the appropriate number densities in performing the isotopic reaction rate tallies. This enhancement to the mesh tally capabilities is described in detail in Reference [6], along with input instructions and examples.

3. Greatly Increased Limits for Geometry, Tally, and Source Specifications

MCNP5-1.60 includes modifications that extend the limits on the number of cells, surfaces, materials, etc., from a maximum of 99,999 (or 10^5-1) to 99,999,999 (or 10^8-1). The maximum number of tally card identifiers was also raised from 999 to 9,999. Detailed discussion of these changes is found in Reference [6]

(1) Many previously hardwired constants, such as 1000003, were used in geometry, source, and tally routines as flags for operators (e.g., ":", "(", "[", etc.). These have been parameterized, with parameters defined in *mcnp_params.F90*.

(2) The maximum card number for MCNP5-1.60 input (e.g., m12345 is material 12345) has been raised from 99,999 to parameter MAX_CARD_NUMBER = 99,999,999 (or 10^8-1).

(3) Many print statements were changed to allow for larger cell & surface numbers. Typically *I5* or *I6* Fortran print formats were changed to *I9*. This results in more blanks in many lines.

(4) The line length for the *big_strings* print routines was changed from 90 to 120. As a result, some lines sent to the screen window may now be up to 120 characters long.

(5) The limit on the size of logical arrays for complicated cells was raised from 1,000 to 9,999. That is, the cell card specification includes a list of surfaces (with ± to denote sense), union operators (:), and parentheses. The maximum length for the list for any single cell is given in MCNP5 by the *mlgc* parameter. The *mlgc* parameter was increased from 1,000 to 9,999. Users should be cautioned that a very long list of surfaces and operators will severely impact MCNP5 tracking performance and result in much longer computer run times. Tracking through the problem geometry is more efficient when the surface lists are shorter, and especially so when the use of union operators is minimized.

4. Web-based Documentation

The documentation for MCNP5-1.60 is organized in an easy to access, web-based format that can be viewed in any web browser. The documentation includes installation notes for Windows, Linux, and Macs; information on the verification testing of MCNP5-1.60; supplemental information for the MCNP5 manual; and descriptions of the nuclear data libraries. New sections have been added to include a number of MCNP5 utility programs and an expanded collection of reference documents. This material can be accessed by opening the file `ABOUT_MCNP5.htm` on the DVD distribution in a web browser.

5. Additional Test Suites for MCNP5

The test suites available with MCNP5 have been expanded to include the `KOBAYASHI` and `POINT_KINETICS` suites, located in the MCNP5/Testing directory. These include a variety of input files intended to be run for minutes and hours rather than seconds. Documentation on these test sets is included in this distribution in Reference [5].

6. Modifications to the `Regression` Test Suite

The standard MCNP5 `Regression` test suite was expanded from 52 to 66 problems, with new tests added to cover new code features or to explicitly test that particular bugs were fixed. Previous analysis of MCNP5 has indicated that the tests cover approximately 80-90% of the total lines of coding. The MCNP5 build system specifically includes capabilities for running any or all of the regression tests and for comparing results with the reference templates.

7. Modifications to Criticality Test Suites

The `VALIDATION_CRITICALITY` Suite was modified to permit running with either ENDF/B-VI data libraries or the newer ENDF/B-VII.0 data libraries. Typing "`make ENDF=7`" will invoke

the test suite using ENDF/B-VII.0 data, while "`make`" or "`make ENDF=6`" will invoke the test suite using the older ENDF/B-VI data.

The `VERIFICATION_KEFF` Suite of analytic test problems was slightly modified.

- A new make target for testing was added – `ten`. Invoking the test suite by "`make ten`" will result in running the ten problems 11, 14, 18, 23, 32,41, 44, 54, 63, and 75. These are the ten problems used in Reference [5] as part of the MCNP5 verification testing.
- Problem 30 in the suite was modified. This problem is a 1-D slab problem with a central fuel region, reflector regions on both sides, and a moderator on one of the sides. Previously, the moderator was thicker than the specified value. The input file had surface 5 specified incorrectly with PX 7.757166007. The correct value is PX 7.439828546 (see LA-UR-01-3082). By making this change, Problem 30 now gives a correct result of 1.00003 +/- 0.00011.

8. Modifications to Shielding Validation Suite

The MCNP5 Shielding Validation Suite was overhauled. Previously, MCNP5 only ran regression tests by comparing current answers with those previously generated. Now, MCNP5 results are compared against experimental data. See the MCNP5-1.60 verification report [5] for details.

9. Enhancements to the *merge_mctal* and *merge_meshtal* Utilities

The *merge_mctal* and *merge_meshtal* utilities are used to merge separate *mctal* or *meshtal* files, respectively, from different independent MCNP5 jobs.

- *merge_mctal* was modified so that it will now handle the merging of perturbed tally results. A few minor corrections were also made to the parsing of the *mctal* file.
- *merge_meshtal* had previously been limited to merging *meshtal* files containing only a single mesh tally. This restriction has been eliminated. *merge_meshtal* is now a wrapper script around the utility program *merge_meshtal_one*, and the *merge_meshtal* will now handle the merging of *meshtal* files with any number of mesh tallies contained in each file.

10. MCNP5-1.60 Build System and Directory Structure

Two new directories were added to the MCNP5 directory structure, MCNP5/bin and MCNP5/Utilities.

When MCNP5 and MAKXSF are compiled from source code, the resulting executables are now copied into the MCNP5/bin directory. All installation and verification/validation test suites use the executable from MCNP5/bin when running the tests.

The MCNP5/Utilities directory has subdirectories for various utility programs and scripts, including: *mcnp_pstudy, merge_mctal, merge_meshtal,* the event log analyzer, onegxs, and stand-alone versions of the MCNP5 random number generators. When MCNP5 is built from source code, the utility programs and scripts are also compiled if necessary and copied into the MCNP5/bin directory.

11. Continue Runs May Use Different Number of Threads

Previously, if a user ran a problem with some number of threads, a continue run would need to use that same number. This was because of how data was written to the *runtpe* file. Arrays have been restructured and written in such a way to allow the number of threads between runs to change.

12. RAND Card Allowed in Continue Run

Previously, the parameters used by the random number generator could not be changed on restart/continue runs. This was changed to allow the RAND card on continue runs. Users can use this feature to change the problem random number seed in a continue run, which can sometimes be a work-around for avoiding a troublesome particle history. Users should note that this is not a generally recommended procedure, since MCNP5 has no provision for keeping a log of changes to the random number seed, and that repeatability of calculations is very difficult if this feature is used.

13. Miscellaneous Enhancements/Changes

- *Improved Support for G95, GFortran, and Absoft Compilers in Windows*

  The optimization has been reduced to -O0 for G95 in Windows, resulting in correct tracking. Plotting is now supported in Windows for all three compilers.

- *More Efficient Sorting of Fission Source Points in KCODE*

  For KCODE problems involving threading, the fission source bank (FSO) must be reordered. Previously, a sorting scheme of $O(N^2)$ was used. A new scheme has been implemented that provides a unique reordering without sorting and is $O(N)$. For details see [7].

- *FSO Array Improvements*

  The fission source bank array (FSO) for KCODE problems has been made two dimensional with the references into column for data type (e.g. x position) parameterized to make source code easier to follow. Also, additional space for URAN was allocated whether used or not. Only if this memory is needed is it allocated.

- *Increased Efficiency of Fission Source Guess Checks*

  The routine *skcode* will locate what cell contains a user-specified guess for a fission source point. Previously, this logic would check all of the other cells to ensure that it is not in multiple cells. As problems become larger and use more fission source points, this additional check for multiple cells becomes prohibitively time consuming. The check to verify that user-defined source points are contained in one and only one cell was removed. This improves

efficiency of problem set up for large problems and simply defers the detection of invalid source points to run-time during histories.

- *Option to Skip Geometry Checking for Very Large Problems*

  For problems with millions or 100s of millions of cells, some of the geometry checking in the *chekcs* routine can take excessive amounts of computing time. An option was added so that if *dbcn*(49) is set to be any positive number, then the geometry checking in *chekcs* is skipped for the problem set up. This option is only recommended for expert users who are sure that their geometry has been constructed correctly; most users should not use this option.

- *Printer Plots Removed for KCODE Results*

  The printer plots that were displayed in the output file after a KCODE calculation are obsolete and have been removed. They may be restored by using the PRINT card to print table 179.

- *Allowed Universe Nesting Increased*

  The maximum number of nested levels for universes has been increased from 10 in MCNP5-1.51 to 20 in MCNP5-1.60.

- *Random Universe Translations Now Unlimited*

  As of MCNP5-1.60, a user may now specify an arbitrary number of random universe translations on the URAN card. Previously, this was limited to two.

- *IDUM and RDUM Sizes Increased*

  The available memory in the IDUM and RDUM arrays has been increased from 50 to 2000. There are now 2,000 entries allowed on each respective card.

- *Size of DBCN Array Increased*

  The DBCN card has been expanded from 30 to 100 allowed entries. The newer entries are currently unused, but reserved for future debug and compatibility options.

- *Shannon Entropy Prints to Output File*

  Previously, certain writes related to the Shannon entropy were only done for the screen. They are now also written to the output file. Answers are not changed.

- *PVM Support Ended*

  PVM has not been supported for many years, and LANL no longer has the capability to test its implementation in MCNP5. All relevant coding for PVM has been removed.

- **WARNING: Co-existence of Weight Windows with Geometry Splitting**

  It is possible for users to be confused over what MCNP is doing when weight window cards (i.e., WWP, WWN, etc.) and cell IMP cards appear in the same input file.

  First and foremost, the cell-based geometry splitting and Russian roulette IMP games are **not** played when either cell-based or mesh-based weight window games are played. (IMP games, when played, are only played at surfaces.) The IMP games are not played even if the weight window game is played at collision points and not at surfaces. Note: Weight window lower bounds can be determined from the IMP values when SWITCHN on the WWP card is positive, but the restriction still holds.

  However, since the weight cutoff game can rely on cell importances, either the default IMP value (default = 1) or the user specified value is used when the weight window in the cell is zero.

  When IMP cards are present with the weight window cards, MCNP checks to see if the IMP values differ from 1 and then prints the warning:

  ```
  this problem has both weight windows and cell importances.
  ```

  An additional warning now appears below this warning and says:

  ```
  geometry splitting and Russian roulette are not played.
  ```

  This is intended to tell the users that geometry splitting and Russian roulette are not done at surfaces when there is a weight window present.

II. Bugs That Were Fixed

This release includes many minor code modifications to fix reported bugs, output formats, error checking, and other difficulties present with previous versions of MCNP. In some cases, the problems that were fixed date back to the 1990s, but were only recently reported and fixed.

In addition, many of the Monte Carlo Codes Group developers are working on MCNP6 for either new feature development or the inclusion of features from MCNPX. In the course of that development, a many minor bugs were discovered and fixed. Any bug fixes applied to MCNP6 were examined to see if they were applicable to MCNP5 as well. When it was confirmed that the same fix should be applied to MCNP5, that was done, with careful attention to differences in data layout, restructuring, or different features between MCNP5 and MCNP6.

- *Incorrect F1 Tally Segment Result with Repeated Structures*

  An error was found that resulted in F1 tallies sometimes returning incorrect results because coordinates were not handled appropriately. This is now fixed.

- *Source Particles Starting on Surfaces Incorrectly Tracked*

  When a source point lies on a surface (in the example, a plane) used elsewhere to define a cell of the problem, that cell seems to be invisible to the source particle, whether the cell is void or filled with a material. Regression Test 116 demonstrates this bug fix.

- *Lethargy Plots and YTITLE*

  A lethargy plot will not be produced if only a YTITLE card is used to label the y axis. This is now fixed.

- *Total User Bin not Calculated with F8 Tallies*

  MCNP would print all zeroes if a total bin was requested for an F8 tally. MCNP5 now prints out the sum of all the bins.

- *Fully Specified Lattice Fill may Cause MCNP5 to Crash*

  Not enough memory was reserved for lattice fills. In the case where it was fully specified, this could cause memory access errors. This is now fixed. See x-3:09-045.

- *Integer Overflow Causes Superfluous Prints to Output Files*

  If the error counter that is passed to the errprn routine is greater that the upper limit of a 4-byte integer, (2147483647), then the error message is printed to the output file.

  Since the error doing this occurs very often, the output file quickly gets very large, potentially filling the disc. This has been fixed.

- *FM Card Specification with Double Parentheses*

  The following cards:

```
fm4     (-1   4   -6)
fm14    ((-1   4   -6))
```

  should yield the same results. MCNP5-1.51, however, would assume the user specified an attenuator in the second case and use junk data in the calculation. MCNP5-1.60 correctly does not apply an attenuator effectively ignoring the second set of parentheses in the second case. The new version gives consistent results.

- *FM Card Invalid Attenuator Specification*

  The following card is incorrect:

```
fm104    ((-1   6   -2)   (-1   6   -2)   (1.0   -1   3   0.1   1))
```

The last bin in the FM card is an incorrectly specified attenuator. In this case, MCNP5 v1.51 would assume a zero attenuation factor for an attenuator of type material 1. While this could be considered correct, the result may not be what the user intends and violates input specifications. MCNP v1.60 gives a fatal error in this case.

- *Fatal Errors for Using Colons with FM Photon Reactions*

  MCNP5 would incorrectly interpret colons in an FM card associated with a photon tally. This led to fatal errors and has been fixed.

- *No Lost Particle Message with NOTRN*

  MCNP5 will appear to run point detector contributions with the NOTRN card even if these contributions get lost. All histories will be logged with "escape" in the Problem Summary Table. Without the NOTRN card, the lost detector contributions result in a message of a lost particle on its way to the detector. MCNP5 now gives reports a lost particle regardless of whether the NOTRN card is present or not.

- *Exponential Transform Incorrect for Multi-Universe Problems*

  The exponential transform stretching direction (specified by the VEC cards) is defined only in the "real world" level of the problem geometry, level-0. Thus if a particle is in a higher geometry level, then the transformation could be applied in the wrong direction. This has been made more general and the problem is fixed.

- *Incorrect Weight Balance When Using PHTVR*

  When the pulse height tally variance reduction (PHTVR) is used, the particle creation/loss tables, and the weight balance in each cell table (print table 130), do not balance out to zero. In other words, the number of particles created and their total weight, does not equal the number of particles lost, and their total weight. MCNP5 now accounts for the weight balance correctly.

- *Direct Source Contribution to FIR tally may be Incorrect with Parallel*

  The TAL array was not set up to have enough memory for parallel processing. This component array has been restructured such that it handles the data correctly.

- *Integer Overflow Causes Incorrect KCODE Tallies for More than $2x10^9$ Histories*

  There are a number of instances where the total number of neutrons (ie, neutrons per cycle times the number of cycles) may exceed 2,147,483,647 (about 2 billion) and overflow 32-bit integers. The variable NSRCK is cast to a 64-bit integer and solves the problem.

- *Bug in Unique File Naming Scheme*

  When a file name is exactly eight characters long and ends with an uppercase character, MCNP5 fails to create the file. This has been fixed.

- *FM Cards with Mesh Tallies may Produce Incorrect Results*

  If FM cards with reaction rate multipliers (i.e., cross-sections) are used with mesh tallies, usually only the last mesh tally will give the correct answers, while the other mesh tallies would report all zeros. This is because the cross sections needed by the other mesh tallies would incorrectly be deleted to save memory. MCNP5 now retains the needed cross sections.

- *Workarounds for Portland Group (PGI) Compiler*

  PGI compilers (versions 8.0.1 and 9.0.2), object to functions with no arguments, as in *angl.F90* and *dbmin.F90*. PGF90 successfully compiles MCNP when a dummy argument is added. In addition, PGI FORTRAN compiler has no *-m64* flag and this has been removed from the processing scripts. The 'reshape' function has been with a data statement for the variable KTL in *mcnp_input.F90*. The Windows PGI compiler crashes on that statement.

- *DOPPLERP Sampling Bug Caused by Form Factor Range*

  With the current photon transport data (mcplib04 and earlier) a provision must be made for the limited range of the tabulation of form factors. Among other issues, this has led to a bug in the sampling of Compton Doppler broadening. Subroutine *dopplerp* is appropriately called when the Klein-Nishina-sampled scattering angle falls within the range of tabulated incoherent form factors (and when the sample is accepted), but not when the sampled angle is above the tabulated range (where no rejection game is played). This can cause an abrupt change in the scattered photon energy spectrum below and above an energy-dependent scattering angle. The correction for this bug is straightforward: simply include the appropriate call to *dopplerp* on the logical branch where the Klein-Nishina sample is always accepted (above the form-factor tabulation).

- *DXTRAN Produces Wrong Answers if used with KCODE*

  If DXTRAN spheres were specified in a KCODE calculation, MCNP would not handle them consistently during the inactive cycles. MCNP5-1.51 would not generate any DXTRAN particles, however, it would also kill all non-DXTRAN particles once they reached a DXTRAN sphere. MCNP5-1.60 completely ignores all DXTRAN spheres during the inactive cycles of a KCODE calculation.

- *Errors in PHTVR if Number of Saved runtpe Dumps Exceeds One*

  If the value of NDMP (the number of dumps to save on the *runtpe* file and set on the PRDMP card) is greater than one, then wrong answers can be produced in pulse height tallies if variance reduction is turned on. MCNP5 now handles multiple dumps to the *runtpe* file correctly.

- *Linear Interpolation in the Input File Fails for Large Integers*

MCNP5 does not perform the correct linear interpolation when processing the input file if the upper limit is an integer, and needs to be a 64-bit integer because of its size. MCNP5 now handles this interpolation correctly by casting to a real number when appropriate.

- *URAN Initialization Bug for Threading in Fixed-Source Problems*

  In the *uran_mod.F90* file, the private array URAN_UIX is not initialized correctly when threading. This affects fixed-source problems, but not KCODE problems. MCNP5 aborts when trying to allocate URAN_UIX, when it is already allocated. MCNP5 now handles memory allocation for this array correctly.

- *MCPLOT Crashes When Reading a runtpe File After Reading a mctal File*

  When invoking MCPLOT with the 'mcnp5 z' command, MCNP5 can crash after the MCPLOT command is given. This is because of incorrect processing of dynamic memory. MCNP5 now handles this memory correctly.

- *Large Number of Neutrons Per Cycle Can Cause Crashes if Source Overrun Occurs*

  More memory has been added to prevent MCNP5 crashes when resizing for source overruns. For threading, it was possible the variable IXAK (handling the index into the fission bank) to be changed by another thread when resizing. An OMP critical section has been added to prevent this.

- *FM Card Reaction Information Not Printed for Mesh Tallies*

  In some neutron mesh tallies, the multiplicative constant, material numbers, and reaction list from the associated FM card are not printed to the output file. These are now printed.

- *Array Bounds Corrections*

  Several arrays in MCNP5 were being accessed in ways that either (a) accessed memory inappropriately, or (b) accessed valid memory for the array, but did so in a way that strict array bounds were not obeyed. (Recall Fortran multidimensional arrays are structured in memory as a one-dimensional array with column major ordering.) The following arrays were affected:

  1) SC, a local array in *mbody*
  2) LJA in *dotrcl*
  3) RKPL in *mcplot_module* when KCODE plots are requested
  4) Dynamic resizing of PHTVR arrays when threading. The array LENG_DE_BR_ARRAYS_THREADS, an array with elements local to each thread, may have been overwritten by other threads.
  5) NMT in *nxtit1*, when 0 is used for material on FM card and first reaction number is negative.

- *Variable Not Initialized in TALLYD*

The local variable *T* was not initialized to a default value. This variable is used for direct contributions to radiography tallies. Since this variable was not used under the conditions it would not be initialized, this could not change answers, but is now initialized to preclude potential problems in future development.

- *Overflow when Printing History Tracking Rate in KCODE*

  As computers get faster, higher rates of neutron histories per hour can be observed that exceed the specified field width. A change has been made to *sumary* that prevents this.

- *MAKXSF – Memory Size Increase for ENDF/B-VII.0*

  Some ENDF/B-VII datasets are too large for the working arrays used in MAKXSF. While these arrays should properly be allocated dynamically, a fast, practical fix is just to increase the size of several arrays. This has been fixed and now works with all current ENDF/B-VII datasets.

- *MAKXSF – Missing S(a,b) Temperatures in ENDF/B-VII.0 xsdir*

  There is a problem with temperature-interpolating the ENDF/B-VII S(a,b) data using MAKXSF. These are now included in the *xsdir* files. Also added an error check for interpolation of S(a,b) datasets where MAKXSF will exit if requested temperature is not between existing low/high temperature datasets.

III. Work in progress

The list below contains items that are known issues with MCNP5-1.60. Work is in progress to address all of these items.

- *MPI hangs for large calculations*

  For very large calculations, e.g., >15K material or >1M cells, some MPI calculations with MCNP5 hang while transmitting the initial problems specification data from the master to the worker processes. It is suspected that the overall message size when sending many arrays together in one message is too large. This should be straightforward to fix.

- *PERT Card May Give Inconsistent Results*

  Users are advised to exercise caution with the PERT card in MCNP5 v1.60 and earlier when using reaction MT numbers with cross section dependent tallies such as k-effective. There are three major known issues:

1) Adding FM cards on one tally may change results of perturbations of another tally. Logic is handled incorrectly in MCNP5 as to how memory is accessed into an array containing tally information.

2) There are two ways to specify a fission reaction that are equivalent using either special ENDF reaction numbers of MTs: -6 in the special ENDF reaction number case or the sum of 18 19 20 21 and 38 for MTs. MCNP5 may yield inconsistent results between these two depending upon how the FM card is specified. Users are strongly encouraged to use -6 for fission always when using the perturbation feature. For instance, perturbing MT = 38, the fourth chance of fission, should have little impact upon a reactor type problem since the fourth chance of fission only occurs at energies typically greater than 15 MeV. MCNP5 gives unreasonable answers for this case.

3) F6 and F7 tallies produce inconsistent perturbation results with equivalent F4 tallies with FM cards when the reaction specified is not the total cross section. Users are encouraged to perturb the cases with F4 type tallies with equivalent FM cards.

These issues will be fixed in MCNP6.

- *Point Kinetics Continue Runs May Crash with Parallel*

  On certain systems, crashes have been observed when reading in a *runtpe* file involving Point Kinetics information. This has only been observed for parallel compilations with PGI and is unpredictable. Should this problem occur, users are encouraged to use a different compiler.

IV. Systems & Compilers

A. Fortran-90 Compilers, OpenMPI, OpenMP

The Fortran-90 compilers supported are listed below along with the recommended compilation options. The gcc C compiler is used for all versions.

The *mpi* notation after a compiler indicates that the compiler can be used to build a parallel, MPI-based executable for MCNP5 using the OpenMPI implementation of MPI. (In general the same Fortran-90 and C compilers should be used for building MCNP5 as were used for building the MPI libraries.)

The *omp* notation after a compiler indicates that OpenMP threading can be used when building an MCNP5 executable. Today, nearly all computers have multi-core processors (e.g., Intel Core2 duo, Intel quad-core Xeon, etc.), and MCNP5 should generally be compiled with the *omp* option. Note that some compilers are not yet suitable for compiling a threaded version of MCNP5.

Linux - 64-bit

- Intel Fortran 10.0.23 + gcc 4.3.3 *[mpi,omp]*

  -O1, -r8, -openmp, -no-vec, -heap-arrays 1024, -m64,

-traceback, -mcmodel=medium, -lm, -fpp0, -pc64, -Vaxlib, -W0

- Portland Group (PGI) – PGI 7.0-5, PGI 9.0-3 *[mpi, omp]*

  -O1, -r8, -m64, -pc 64, -mp, -Mnosgimp

- gfortran

Linux – 32-bit

- **(TBD)**

Windows – 32-bit

- Absoft Pro Fortran 11.0.0

  -O1, -N113 (equiv to –r8)

- Compaq Visual Fortran Optimizing Compiler Version 6.6 (Update B), Compaq Visual Fortran 6.6-2518-47C86

  /optimize:5, /nopdbfile, /real_size:64

- g95 + gcc version 4.0.3 (g95 0.92!) Jun 17 2009

  -O0, -r8, -i4, -fdollar-ok, -mieee-fp, -fast-math, -fcase-upper, -fno-underscoring, -fsloppy-char

- Intel(R) Visual Fortran Compiler Professional for applications running on IA-32, Version 11.1 Build 20100203 Package ID: w_cprof_p_11.1.060

  /O1, /threads, /real_size:64

Windows – 64-bit

- Intel(R) Visual Fortran Compiler Professional for applications running on IA-32, Version 11.1 Build 20100203 Package ID: w_cprof_p_11.1.060

  /O1, /threads, /real_size:64

  The 64-bit Windows executables do not support MCNP plotting, due to lack of 64-bit X11 (X Windows) libraries. Most users should use the 32-bit Windows version of MCNP5, since it supports plotting and will run on both 32- and 64-bit Windows systems.

Mac – Intel-based, OS X 10.6.4 & OS X 10.5.8

- Intel 10.1.008 Fortran *[mpi, omp]*

  -O1, -r8, -openmp, -traceback, -heap-arrays 1024

- Intel 11.1.088 Fortran *[mpi, omp]*

  -O1, -r8, -openmp, -traceback, -heap-arrays 1024, -m32 or -m64

- Absoft 11.0

  -O1, -N113, -YEXT_NAMES=LCS, -YEXT_SFX=_,
  -YCFRL=1, -lU77

- G95 0.92 (gcc 4.0.3)

- Gfortran 4.4.3

Mac – PowerPC-based

- Not supported.

PVM-based parallelism is obsolete and is no longer supported for MCNP5.

The following executables for MCNP5-1.60 were compiled using the CONFIG option strings shown, tested, and included with the new MCNP5-1.60 release package. The typical commands to build and test MCNP5 are:

```
cd MCNP5/Source
make realclean
make CONFIG='config-options'  install  GNUJ=8
```

- **Mac (Intel, 32-bit) -** OS X 10.6.4, Intel F90 compiler 10.1.008, gcc 4.2.1

  mcnp5_i386 `CONFIG='intel gcc plot omp'`

- **Mac (PowerPC) -** not supported

- **Linux (32-bit) –** Fedora Core release 3 (Heidelberg), Intel Fortran compiler 9.1.037, gcc 3.4.4

  mcnp5_i386 `CONFIG='intel plot'  MARCH=M32`

  mcnp5_i386_omp `CONFIG='intel plot omp' MARCH=M32`

- **Linux (64-bit) -** Red Hat Enterprise Linux ES release 4 (Nahant update 3),
  Intel Fortran compiler 10.0.023, gcc 3.4.5

mcnp5_x86_64 `CONFIG='intel plot'  MARCH=M64`

mcnp5_x86_64_omp `CONFIG='intel plot omp' MARCH=M64`

- **Windows (32-bit) -** Windows XP (Service Pack 2), Cygwin 1.5.24-2, Intel Visual Fortran Compiler 11.1.060, gcc 3.4.4, MPICH2 1.0.6p1

  mcnp5.exe `CONFIG='intel plot omp'`

  mcnp5_seq.exe `CONFIG='intel plot'`

  mcnp5_mpi.exe `CONFIG='intel plot mpi'`

- **Windows (64-bit) -** Windows XP (Service Pack 2), Cygwin 1.5.24-2, Intel Visual Fortran Compiler 11.1.060, gcc 3.4.4, MPICH2 1.0.6p1

  mcnp5_x86_64.exe `CONFIG='intel omp'`

  note: does not support plotting, due to lack of 64-bit X11 libraries

V. Release Package

The release package for MCNP5-1.60 was burned to DVD. Using the DVD, the release was tested on Windows, Linux, and Mac OS X systems, including: installation, compilation, and the Regression Test Suite. The DVD for the MCNP5-1.60 release package will be conveyed to RSICC for distribution.

The reference templates for output and tally files used in the MCNP5-160 Regression test suite (i.e., installation tests) were generated for Linux and Windows as follows:

Windows: 32 Bit Pentium IV, Windows 2000 Professional SP4, Intel 9.1 f90, gcc

Linux: 64 bit AMD Opteron 2.0 GHz, Red Hat Linux 9, Intel 10.0.023 f90, gcc

For other systems, the Linux reference templates are used. While the MCNP5 developers strive to reduce or eliminate arithmetic roundoff differences that lead to differences in results, users should be aware that compiling and testing MCNP5 with different hardware, operating system, Fortran90 compilers, C compilers, or different build options may lead to differences in results due to computer roundoff. (Results should agree with the reference templates within the combined statistical precision.) For the release of MCNP5-1.60, all of the configuration files for building MCNP5 made explicit use of the "*-r8*" Fortran option and typically used the *"-O1*" optimization level when compiling MCNP5. The use of these option produces better (but not perfect) consistency of results among MCNP5 executables run on different systems (Windows/Linux/Mac/Unix) or with different Fortran compilers. For more details on this, see the report "Verification of MCNP5-1.60" from the `ABOUT_MCNP5.htm` file.

VI. Installation instructions

Any previous versions of MCNP5 or the nuclear data libraries should be renamed, relocated, or deleted prior to installing the latest versions.

Before installing MCNP5 and the new MCNP Data Libraries on Unix, Linux, and Mac OS X systems, users should make sure that at least 12 GB of free disk space is available.

On Windows systems, additional disk space is required when using the Windows Installer, and users should make sure that at least 25 GB of free disk space is available.

Installation of the latest release of MCNP-1.60 and the data libraries on Windows systems is the same as described in the PDF file 'MCNP5 Installation Guide' that can be accessed from the file `ABOUT_MCNP5.htm` on the RSICC DVD.

For Linux and Mac OS X systems, the installation script is available from the file `ABOUT_MCNP5.htm` on the RSICC DVD.

VII. References

1. X-5 Monte Carlo Team, "MCNP – A General N-Particle Transport Code, Version 5 – Volume I: Overview and Theory", LA-UR-03-1987, Los Alamos National Laboratory (April, 2003).
2. T.E. Booth, et al., "MCNP-1.50 Release Notes", LA-UR-08-2300 (2008).
3. Forrest B. Brown, Jeffrey S. Bull, John T. Goorley, Avneet Sood, Jeremy E. Sweezy, "MCNP5-1.51 Release Notes", LA-UR-09-00384 (2009).
4. Forrest B. Brown, Jeremy E. Sweezy, Jeffrey S. Bull, Avneet Sood, "Verification of MCNP5-1.50", LA-UR-08-3443 (2008).
5. Forrest Brown, Brian Kiedrowski, Jeffrey Bull, Matthew Gonzales, Nathan Gibson, "Verification of MCNP5-1.60", LA-UR-10-05611 (2010).
6. Brian Kiedrowski, Forrest Brown, Jeffrey Bull, "MCNP5-1.60 Feature Enhancements", LA-UR-10-xxxxx (2010).
7. F.B. Brown, T.M. Sutton, "Reproducibility and Monte Carlo Eigenvalue Calculations", *Trans. Am. Nuc. Soc.* **65**, 235 (1992).