

VERSION UPDATE

4N REACTION MODELS IN ST_MoNA

AND OTHER HELPFUL INFORMATION

ST_MoNA 0.01

Author:

Michael D. JONES

December 16, 2013

Contents

1.0 Introduction	2
1.1 An Ideal Path	2
2.0 Dealing with Wheezy	3
2.1 Making ST_MoNA Work on Wheezy	3
2.2 Jenna's notes from 2 Days of Debugging	6
3.0 Input Files	10
3.1 Selecting an Input File	10
3.2 A Sample Input File.	12
3.3 Things you should NOT do. (AKA bugs)	14
4.0 A New Interpolator: CycSrim	18
4.1 How is dE Calculated in ST_MoNA?	18
4.2 How to Create a CycSrim Target	19
4.3 Changing the Energy Range of CycSrim	21
4.4 Adding the Target to CycSrim	22
4.5 Changes to ST_MoNA	24
5.0 Installing CycSrim	25
6.0 Multi-Neutron Reactions (4n and 3n)	31
6.1 Available Decay Modes	32
7.0 Misc. Errata	34
8.0 Sample Files	35
8.1 4n Phase Space Example	35
8.2 A working bashrc	37
8.3 A working rootrc	39
8.4 A working bash_env	53

1.0 Introduction

This guide serves to document the major changes to ST_MoNA post-wheezy. It is largely a compilation of guides put together so that you may avoid some pitfalls and quickly get back to happily doing physics.

The changes that come with this version of ST_MoNA are as follows:

1. Making ROOT and ST_MoNA compatible with Wheezy.
2. The implementation of Input Files.
3. A new Energy Loss Interpolator (CycSrim)
4. 4n and 3n Reaction Models
5. Misc. Errata (ex. new distributions for random numbers)

Each section will outline the changes, their limitations and examples on how to use them. This is not guaranteed to solve all your problems, but hopefully it will give you a good starting point.

1.1 An Ideal Path

Ideally all the files you need should be in their respective *relative* directories, so you should only have to worry about absolute directories and otherwise have everything you need. The version referred to in this document has its home in `"/projects/mona-sim/jonesm."` Many usefull scripts also reside there. A recommended course of action is:

1. Get ROOT working with wheezy. (Section 2)
2. Build CycSrim (a directory change will be needed here) (Also in Section 2, details in 4 and 5)
3. Build ST_MoNA, Rebuild root2pythia. (Section 2)
4. Understand the Input files and new Reaction Models so you are aware of they can and cannot do (Sections 3 and 6).
5. Get back to some awesome physics

An example script to run this new version, as well as a working bashrc, bash_env, etc. can be found at the very end.

2.0 Dealing with Wheezy

Before we can even run ST_MoNA we need to make ROOT happy with wheezy and make a few minor changes in the GEANT code. This section is largely written by Jenna Smith with a small addition by myself to account for CycSrim.

2.1 Making ST_MoNA Work on Wheezy

WARNING FOR THE READER: These are notes and guides worked out by Jenna Smith for getting ROOT to work and *her version of st_mona* to compile. Her version is NOT the 4n version, and it did not include CycSrim. However, this information is still useful and applicable. If you are installing my version (input files, 4n reactions) you will need to first make CycSrim with the following command before you try to compile ST_MoNA:

```
1 make CycSrimDict.cxx
```

It may be useful to check out the section on installing CycSrim as there are a couple of additional files you will need for it to successfully run. You may also have to change your .rootrc file. A copy of the one I use as of 12/12/2013 (and works post-wheezy apocalypse) can be found in the appendix.

Step I

Make ROOT work. Wheezy uses modules. For more information, see the computer department webpage here. Add the following to your bashrc:

```
1 if test -n "$(typeset -F|grep \ module\$)" ; then
  module load null root texlive geant4 gcc
3 fi
```

We use a library called libm.so. Usually this is in the form of a symbolic link in n2analysis/lib/root or in your LoadGSLClasses.C file. On fishtank, the library is located at /usr/lib/libm.so. On wheezy, the library is located at

```
1 /usr/lib/x86_64-linux-gnu/libm.so
```

This location needs to be changed for wheezy, as well as changed back if you move back to fishtank during the testing period. Recompile your n2analysis libraries probably using compile_libs.sh from the n2analysis/src directory. If you switch back to fishtank, you'll need to recompile these again.

The location of all the root documents have changed from some directory that looks like

```
1 /opt/lucid/root/5.26.00
```

to another one that is set to the environment variable ROOTSYS when the root module is loaded (see above).

To fix this, change any mention of the previous path:

```
1 (/opt/lucid/root/5.26.00)
```

to

```
1 $ROOTSYS or $(ROOTSYS) or ${ROOTSYS}
```

depending on where the mention is. Files to check include .bashrc, .bash.env, .rootrc, your ROOT logon and logoff files, and any files that are called from these. Log off and re-logon to wheezy to clean out all of the mentions of old file paths. If you have more problems at this point, check the more detailed notes below.

Step II

Make st_mona work. Most of the fixes you need youve already done. Recompile the program and test it out. Use make clean, then make. If you have more problems, check the detailed notes below. If you switch back to fishtank, youll need to do this every time you switch.

Step III

Make GEANT4 work. root2pythia needs to be recompiled. You do not do this by typing make in the control directory. You do this by executing the file Makefile with the following command:

```
1 ./ Makefile
```

You will have to do this again if you switch back to fishtank. GEANT4 update: the G4Exception function now requires at least three different arguments. This has been a GEANT4 code update. See [here](#) and [here](#) for more information. This update requires you to make changes to menate_R.cc (located in n2_vault/src). Change all G4Exception(text here) to:

```

1 enum G4ExceptionSeverity severity;
   severity = JustWarning;
3 G4Exception("text here", "", severity, "");

```

Recompile n2_vault, without sourcing env.sh. This should create an executable, n2_vault, in the directory:

```

1 ${G4WORKDIR}/bin/${G4SYSTEM}.

```

If you switch back to fishtank, you'll need to do this again, but this time source env.sh beforehand. The directory with the executable needs to be added to your PATH. You can do this in whatever way you like, but I added the following line just below my module declarations in .bashrc:

```

1 export PATH=${PATH}":${G4WORKDIR}/bin/${G4SYSTEM}" source

```

Step IV

Make mona_analysis work. The following applies if you have use Ragny as your energy-loss interpolator. If you have the CycSrim interpolation, you will encounter a totally different (but simple) problem.

For Ragny users:

The GSL interpolators behavior has been changed. Previously, if the interpolator was given a value outside of its range, it extrapolated. Now it just spits back a NaN. This happens in the function gsl_interp_eval which is only called in st_interp_invers.cc. Specifically its when the following line in st_par_mat_ia.cc is called:

```

1 fData.ri = (*interp)(fData.ti);

```

For a quick fix, take the loop that sets up the interpolator in st_par_mat_ia.cc:

```

1 for (fData.ti=1; fData.ti < enOrig*2; fData.ti*=1.01){

```

and extend the maximum energy from enOrig*2 to enOrig*20:

```
1 for (fData.ti=1; fData.ti < enOrig*20; fData.ti*=1.01){
```

For a more permanent fix, see Mike about installing CycSrim, which doesnt use the GSL interpolator at all. Dont forget to recompile!

For CycSrim users:

Before you remake ST_MoNA, you have to remake the dictionary file for CycSrim:

```
1 make CycSrimDict.cxx
```

then make clean. There seems to be an issue with the GetSrimDataDir() function. It returns a TString but this isnt being recognized by the Form function in the initialization of cycsrim. i.e. it cant find the data file. A quick fix for this is to open CycSrim.cxx and go to CycSrim::Init(). On the line that says:

```
1 srimFile = Form("%s/srim/%s", GetSrimDataDir(), srimfile[fMaterial])
```

Remove the first %s and replace it with the path to your srim directory. Then remove GetSrimDataDir() from the function argument. This is a crude workaround, but it should be ok since the function GetSrimDataDir() is only called here.

2.2 Jenna's notes from 2 Days of Debugging

Changes to make for wheezy:

Add the following to your bashrc:

```
1 if test -n "$(typeset -F|grep \ module\$)" ; then
  module load null root texlive geant4 gcc
3 fi
```

Error when trying to run st_mona:

```
1 ./st_mona: error while loading shared libraries: libf2c.so.2:
  cannot open shared object file: No such file or directory
```

Error when trying to make clean:

```
rageny.c:23:17: fatal error: f2c.h: No such file or directory
```

This means you need to put f2c.h in your include directory. Or that the computer department needs to put f2c.h back in /usr/include. NOTE: Computer department has done this. Next error when trying to make after make clean (occurs after st_mona.cc is made):

```
1 /usr/bin/ld: cannot find -lf2c
```

Computer department loaded f2c library. st_mona suite compiles. It does not, however, run properly. Error message:

```
1 Fatal in <TVirtualStreamerInfo::Factory>: Cannot find the plugin handler
  for TVirtualStreamerInfo!
However $ROOTSYS/etc/plugins/TVirtualStreamerInfo is accessible, Check the
  content of this directory!
3 Error occurs between      st_mona.cc:1459 (main) :: 11Li proposal one-
  neutron simulation      and Number of events:      500000 events.
```

Any links to libm.so need to be re-linked from

```
1 /usr/lib/libm.so to /usr/lib/x86_64-linux-gnu/libm.so.
```

Including in customize_root/LoadGSLClasses.C and in n2analysis/lib/root. If you move back to fishtank, these will need to be changed back. Recompile the libraries. Any instances of /opt/lucid/root/5.26.00 (or the like) in .rootlogon or .rootrc need to be changed to

```
1 $(ROOTSYS)
```

This should make it robust regardless of fishtank or wheezy. Recompile st_mona after ROOT works without any errors. In the n2_vault directory, source the env.sh file:

```
1 source env.sh
```

and then recompile n2_vault as well:

```
1 make
```


env.sh resets a bunch of Geant4 variables. It sets G4ABLADATA, which is not set in the geant module. It also doesn't set G4DEBUG. I made without sourcing env.sh with more success, until I got the following error while compiling menate_R.cc:

```

1 src/menate_R.cc:906:79: error: no matching function for call to '
    G4Exception(const char [58])'
src/menate_R.cc:906:79: note: candidates are:
3 In file included from include/menate_R.hh:111:0,
    from src/menate_R.cc:54:
5 /mnt/misc/sw/x86_64/Debian/7/geant4/gnu/4.9.6.02/include/Geant4/globals.hh
    :78:6:
    note: void G4Exception(const char*, const char*, G4ExceptionSeverity, const
        char*)
7 /mnt/misc/sw/x86_64/Debian/7/geant4/gnu/4.9.6.02/include/Geant4/globals.hh
    :78:6:
    note: candidate expects 4 arguments, 1 provided
9 /mnt/misc/sw/x86_64/Debian/7/geant4/gnu/4.9.6.02/include/Geant4/globals.hh
    :82:6:
    note: void G4Exception(const char*, const char*, G4ExceptionSeverity,
        G4ExceptionDescription&)
11 /mnt/misc/sw/x86_64/Debian/7/geant4/gnu/4.9.6.02/include/Geant4/globals.hh
    :82:6:
    note: candidate expects 4 arguments, 1 provided
13 /mnt/misc/sw/x86_64/Debian/7/geant4/gnu/4.9.6.02/include/Geant4/globals.hh
    :86:6:
    note: void G4Exception(const char*, const char*, G4ExceptionSeverity,
        G4ExceptionDescription&, const char*)
15 /mnt/misc/sw/x86_64/Debian/7/geant4/gnu/4.9.6.02/include/Geant4/globals.hh
    :86:6:
    note: candidate expects 5 arguments, 1 provided

```

From this error message, I tracked down the files globals.hh (on fishtank and on wheezy) and discovered that the G4Exception function now requires at least three different arguments. This has been a GEANT4 code update. See [here](#) and [here](#) for more information.

Solution: Change all G4Exception("text here") to:

```

enum G4ExceptionSeverity severity;
2 severity = JustWarning;
G4Exception("text here", "", severity, "");

```

root2pythia also needs to be recompiled. You do not do this by typing make in the control directory. You do this by executing the file Makefile with the following command:

```
1 ./ Makefile
```

The next problem to solve is that my PATH doesn't include the proper directory. You need to add the following to your path:

```
1 ${G4WORKDIR}/bin/${G4SYSTEM}
```

Where is the best place to do this? At the very least, it can be done in the command line. That makes GEANT work. Now mona_analysis.

Error message:

```
1 I /projects/mona-sim/smithj/st_mona/src/../../include/st_par_mat_ia.hh:97 (
  StRageny) :: Setting
up energy loss with interpolator
3 I mona_analysis.cc:2303 (main) :: Using settings for fragment 9Li from
  10007
I st_par_mat_ia.cc:80 (do_de) :: Setting up interpolator for energy loss
  for beam A Z: 9.0 3.0
5 I rageny.c:3913 (rageny_calc) :: Allocating integration work space
I st_par_mat_ia.cc:103 (do_de) :: Done
7 gsl: interp.c:150: ERROR: interpolation error
Default GSL error handler invoked.
9 ./9li_1ndecay_path1.sh: line 22: 18500 Aborted
```

So there is some interpolation error after the interpolator is created. Oh look, this page and this page suggest that the GSL library behavior has been changed so that the interpolation function now returns NaN when x is out of range instead of extrapolating. Specifically, this would be using the function `gsl_interp_eval` which is only called in `st_interp_invers.cc`. Yep, that's exactly what's happening. Specifically it's when the following line in `st_par_mat_ia.cc` is called:

```
fData.ri = (*interp)(fData.ti);
```

Why is this a problem in `mona_analysis` and not in `st_mona`? Because when you do the initial input, you have a well-behaved and orderly input array of energies (hypothetically). You track these through the magnet with a 5-parameter forward map. The 4-parameter inverse map, however, has the possibility of giving you values slightly outside the range of the interpolator. I could always extend the range of the interpolator. That would be

a quick fix. In addition, the interpolator is only created once, so its not as if Im seriously extending the time it takes to run simulations.

Solution(s): For a quick fix, take the loop that sets up the interpolator in `st_par_mat_ia.cc` and extend the maximum energy from `enOrig*2` to `enOrig*20`. For a more permanent fix, see Mike about installing CycSrim.

3.0 Input Files

This section serves to explain how to write input files for ST_MoNA, as well as its limitations. Using an input file allows the user to change variables (or change experiments) in the code without recompiling. For example the target thickness, `dTarget`, or the variation in the beam energy `dEbeam`. This is convenient, and allows one to run many simulations while varying some parameter, and in general simplifies the code. The input files are in .txt format. ST_MoNA simulations are still run in the same manner with shell scripts. **It is important to note that the arguments in executing `st_mona` take precedence over anything written in the input file.** This is by design. If a variable is set in the input file, and then changed in the shell script, the program will run with the new value and overwrite the one set by the input file.

3.1 Selecting an Input File

The full filepath to the input file must be specified in `st_mona.cc`. Once the input file location is given, you can freely edit the input file (such that it still reads in properly). In `st_mona.cc`, after setting the default values at around line 280, the location of the input file is specified:

```
1 ////////////// st_mona.cc line 280 //////////////
string INPUT_DATA_DIR = "/projects/mona-sim/jonesm/st_mona/input_files/";
3 string INPUT_FILE = "input.txt"
inputfile = INPUT_DATA_DIR + INPUT_FILE;
```

Where `INPUT_DATA_DIR` is the location to the input file, and `INPUT_FILE` is the name. Change BOTH of these to correspond to the input file you want to read and compile `st_mona`. Note that `INPUT_DATA_DIR` must be the full filepath. You must also change the SAME lines in `mona_analysis.cc`.

But doing it in this manner defeats the purpose of the input file.

To have modular Input File

One solution is to use a bash script which writes your desired inputs to a temporary

input file "input.txt" that will be read by BOTH st_mona.c and mona_analysis.c, in addition to creating an additional copy with a custome name (so that you know what conditions your simulation has run under). This can be accomplished with the script "write_input.sh" which accepts up to 60 input variables. The way to call this script within another script is to first define the parameters you wish to set:(in bash)

```

2      declare -a parameters=( "eBeam           = 59.34;"
4                                "beamA           = 14;"
6                                "beamZ           = 4;"
8                                "dTarget         = 435.0;"
10                               "dEbeam          = 0.0215;"
12                               "nNeutr          = 2;"
14                               "nProt           = 2;"
16                               "bSpotCx         = 0.0;"
18                               "bSpotCtx        = 0.0;"
20                               "bSpotCy         = 0.0;"
22                               "bSpotCty        = 0.0;"
24                               "bSpotDx         = 0.003;"
26                               "bSpotDy         = 0.0045;"
                                "bSpotDtx        = 0.006;"
                                "bSpotDty        = 0.00325;"
                                "crdc1MaskLeft    = 0.15;"
                                "crdc1MaskRight   = -0.15;"
                                "crdc2dist        = 1.88;"
                                "monaDist         = 8.2;"
                                "cosymap          = \"m8he_jesse\";"
                                "maxbulge         = 0.000000000005;"
                                "tflatfoil        = 0.000000125;"
                                "fragA           = 8.0;"
                                "fragZ           = 2.0;"
                                "targA           = 9.0;"
                                "targZ           = 4.0;" )

```

and then call the script "write_input.sh". The first argument is the name of the copy input file that will be made. You can include the date and time in the filename if you so desire.

```

cd ${inputpath}
2 sh write_input.sh "test_input.txt" "${parameters[0]}" "${parameters
   [1]}" "${parameters[2]}" "${parameters[3]}" "${parameters[4]}" "${
   parameters[5]}" "${parameters[6]}" "${parameters[7]}" "${parameters
   [8]}" "${parameters[9]}" "${parameters[10]}" "${parameters[11]}" "${
   parameters[12]}" "${parameters[13]}" "${parameters[14]}" "${

```

```
parameters[15]}" "${parameters[16]}" "${parameters[17]}" "${
parameters[18]}" "${parameters[19]}" "${parameters[20]}" "${
parameters[21]}" "${parameters[22]}" "${parameters[23]}" "${
parameters[24]}" "${parameters[25]}"
```

This script will create a file called "test_input.txt" in the directory where "write_input.sh" resides, and write to a file called "input.txt" the array "parameters" that you set earlier in the script. In this way, you can use multiple input files (and change them dynamically if you want) without recompiling. Making copies is important, otherwise you will forget what settings your simulations was run with! So be sure to be smart!

3.2 A Sample Input File.

Here is a sample input file for ST_MoNA:

```
////////// INPUT.FILE.txt //////////
2 #The pound-sign denotes a comment which will not be read.
   reacType = "d,p"; #Strings are notated with quotes on either
   side
4 eBeam    = 82.4; #Doubles and integers are notated normally.
   beamA    = 24;
6 beamZ    = 8;
   dTarget  = 400.0;
8 dEbeam    = 0.0215;
   cosymap  = "m24O";
```

If a variable is not set in the input file, the default value will be taken (specified in st_mona.cc). EXCEPT for the cosymap. There exists no default cosymap. ALWAYS define the cosymap, your simulation cannot run without it. Many pre-generated input files can be found in the directory(s):

```
\path\to\location\of\st_mona\input_files\e12345
```

Where e12345 is your experiment number. WARNING: These pre-generated input files DO NOT set the cosymaps. You must set them appropriately. Variables which are defined as integers or doubles are set in the input file by typing:

```
1 variable_name = value;
```

The equals sign serves as a delimiter between the variable and the value to set. The semicolon at the end marks the end of the line. Whitespace or tab before and after the variable_name, or value is, in principle ok. Comments are notated with a pound-sign (#). Lines begining with a #-sign will not be read. If a value is the result of a mathematical calculation, type the result of that calculation and do not type the operation itself. For example:

```
1 eBeam = 50 + 10; #THIS CODE WILL NOT WORK!
```

Will not work because the value after the equals sign is read in as a string “50 + 10”, which is then converted to a double, and this conversion does not perform the addition. Instead only the first value will be read, so eBeam will be set to 50, NOT 60. If the variable you want to set is a string, you must put quotations on both ends of the string:

```
1 var_string = "a string of characters";
```

The order in which the variables are defined in the input file does not matter. If a variable name does not exist within a predefined set of variables, the input file will ignore that variable. For example adding:

```
1 bogus = 3.1415;
```

Will do nothing. The code will read in both “bogus” and “3.1415” but will find that “bogus” does not exist within a list of variables and thus not set any values for bogus. To add a variable to the input file, simply extend the list of accepted variables in st_mona.c:

```
1 /////////////// st_mona.cc line 370 ///////////////
  else if( strcmp(var_name,"velocityshift") == 0) velocityshift =
    dvalue;
3 else if( strcmp(var_name,"asymMomentum") == 0) asymMomentum =
    dvalue;
  else if( strcmp(var_name,"my_new_double") == 0) my_new_double =
    dvalue;
5 ....
```

You must differentiate between datatypes when adding new variables to the list of accepted variables. There are seperate lists for ints, doubles and strings. Integers are set with “ivalue”, doubles with “dvalue”, and strings with “var_value.” A list of accepted variable names and cosymaps as of December 16, 2013, can be found in tables 1 and 2.

3.3 Things you should NOT do. (AKA bugs)

This section outlines some things that will cause the input file to crash and explode, and will hopefully provide some debugging insight. When commenting out a line, one does not have the freedom of commenting as if they were in a real coding environment. For example, if you comment out a variable, the subsequent variable will not be read in. The code will still run, just the following value will not have the value the user specified and rather will be the default:

```
1 #THIS CODE WILL NOT WORK!
   reacType = "d,p"; # Comments after the variable are OK.
3 # eBeam = 82.4;
  # Commenting in this manner will cause beamA to not be read.
5 beamA = 24;
   beamZ = 8;
7 dTarget = 400.0;
   dEbeam = 0.0215;
```

In the example above, the beam energy eBeam is commented out (#). Commenting in this manner causes the code to use default values for BOTH eBeam (which was not read), AND the following line (beamA in this case). Why this is the case still is a mystery, but comments following the value will not affect reading in any of the values.:

```
  #Comments before any variables are ok.
2 #Multiple comments are also ok.
   reacType = "d,p"; # Comments after the variable are OK.
4 eBeam = 82.4;
   beamA = 24 ; #You can even put whitespace after the
   value
6 beamZ = 8;
   dTarget = 400.0;
8 dEbeam = 0.0215;
```

Do not delete the semicolons. The input file will not read correctly without semicolons:

```
   reacType = "d,p" # THIS CODE WILL NOT WORK! The semicolon is
   missing.
2 eBeam = 82.4
```

As stated in the previous section, do not add mathematical operations in the input file. In addition, including empty lines can cause the program to not read the following line. For example:

```
2   reacType  = "d,p"; # THIS CODE WILL NOT WORK!  
4   eBeam     = 82.4; #An empty line between values.  
    ...  
#Trailing whitespace at the end is ok.
```

The whitespace inbetween the two variables will cause eBeam to not be properly read. Another possible error is giving the input file a character when it expects a number or vice-versa. Be sure to check that the data type of the value you are setting matches the variable you are setting it to.

Table 1: List of Accepted Variables

Doubles (dvalue)	More Doubles (dvalue)	Integers (ivalue)	String (var_value)
reacQval	crdc1MaskLeft	resMonaBar	reacType
reacQvalSpread	crdc1MaskRight	beamA	aDistType
reacASpread	crdc2MaskLeft	beamZ	debugSwitch
reacALowLim	crdc2MaskTop	nNeutr	exp
reacAUpLim	crdc2MaskBot	nProt	cosymap
eBeam	distTypeMx	fragA	
dEbeam	velocityshift	fragZ	
dTarget	asymMomentum	targA	
resTime		targZ	
resTargetX			
resCRDC1X			
resCRDC1Y			
resCRDC1ThetaX			
resCRDC1ThetaY			
resMonaX1			
resMonaX2			
resMonaP			
scaleGlaub			
scaleStrag			
bSpotDx			
bSpotDy			
bSpotDtx			
bSpotDty			
bSpotCx			
bSpotCty			
crdc2dist			
monaDist			

Table 2: List of Cosy Maps ordered by increasing Z (left to right) and A (top to bottom)

Helium	Lithium	Beryllium	Boron	Carbon	Oxygen	Fluorine	Neon
m6He	m8Li	m10Bea	m13B	m16C	m18Of	m26f	m25Ne
mJenna	m9Li	m10Beb	m15B_06025		m19O		m26ne_calem
m8he_jesse	m11Li	m11Bec			m20O		
	m11Li_BeBeam	m12be_jesse			m21Of		
		m14Be_B17beam			m22O		
		m14Be			m220a		
					m22Of		
					m22o_09028		
					m23O		
					m23Oa		
					m23o_09028		
					m24O		
					m24o_09028		
					m24o_hope		

4.0 A New Interpolator: CycSrim

CycSrim is an energy loss interpolator that relies on data files produced by SRIM (The Stopping and Range of Ions in Matter). SRIM is a collection of programs that calculate the stopping power and range of a given ion Z up to 2 GeV/u using a quantum mechanical treatment of the ion-atom collision. More information on SRIM can be found at <http://www.srim.org/#SRIM>.

The current implementation of CycSrim is limited by SRIM to energies less than 2 GeV/u and $Z = [1-92]$. CycSrim does NOT incorporate energy straggling. Given an initial kinetic energy, CycSrim will always return the same value.

4.1 How is dE Calculated in ST_MoNA?

The energy loss is calculated on an event by event basis in `st_material.cc`, and then again in `mona_analysis.c` where the decay energy reconstruction is performed. Specifically the energy loss is calculated in the function `do_de` in `st_par_mat_ia.cc`, which is called in `StMaterial::act(StParticle *p)`. Here the beam is propagated a random distance d into the target, whereby the energy loss is calculated by the function call `do_de()`:

```
1 // (StMaterial.c)
  de = fPMInt->do_de(this, d, p);
3 de *= p->getA();
```

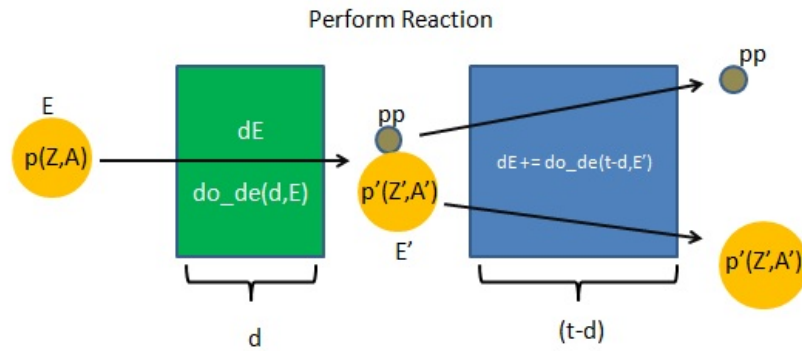
The `do_de()` function itself returns the energy loss in the target, and sets the final kinetic energy of the particle passed to the function. Note that `do_de()` returns the energy loss in MeV/u. The angular straggling is calculated by `stragAng()`. At this point the reaction is performed (instantaneously) at the new energy, and the beam particle changes in A and/or Z by the appropriate amount dictated by the reaction. In addition, a new particle(s) is also created. These new particles then inherit an energy and momentum dependent on the input decay energy and reaction mechanism. The energy loss is then calculated for the fragment through the remaining thickness of the target ($t-d$). NOTE: The energy loss of the created particle is NOT calculated, presumably because they are neutrons and have no charge.

```
1 // energy loss and straggling after
  DSV(fThickness-d);
3 de += fPMInt->do_de(this, fThickness-d, p) * p->getA();
  strag += fPMInt->stragAng(this, fThickness-d, p, fStragAngScale);
5 DSV(strag);
  DSV(de);
7 fEnergyLoss.push_back(de);
  fStragAng.push_back(strag);
```

The total energy loss and straggling are then applied to the fragment with the `push_back` functions and passed on to the remainder of the code. Due to the sparseness of the SRIM tables at energies above and around 50 MeV/u, it is necessary to make sure that CycSrim's interpolator has a sufficient amount of data points to work with. (Otherwise your dE spectrum will contain spikes or funky values.) This is done by manipulating the 4th argument in the `Interp` function in `CycSrim.cxx`:

```
Double_t range = Interp( evals , ranges , numE, 4 , e_a ) ; // 2
```

Where `evals` and `ranges` are arrays of length `numE`, and the input energy to interpolate is `e_a`. The integer in the 4th argument controls the number of terms in the fitting polynomial. The maximum is approximately 10, at which point the code segfaults. 2 is insufficient above 50 MeV/u.



4.2 How to Create a CycSrim Target

To create the energy loss tables it is necessary to have both SRIM and ROOT installed on a Windows machine. This is due to the fact that CycSrim pulls its data from SRIM which can currently only run on Windows. SRIM and ROOT can easily be downloaded from the internet, although you may need help to complete the installations. This guide will assume these programs are installed. Depending on your ROOT installation you may have to include the bin path in your Environment Variables on Windows (Accessible under Control Panel → System → Advanced System Settings → Environment Variables).

Once SRIM and ROOT are installed and running on windows you must acquire a ROOT macro called `GenerateRangeTables.C` either from the author or Zach Kohley, or someone in possession of said macro. For simplicity, place it in the root folder along

with SCOEF03.dat and SNUC03.dat (data files for the SRModule). The macro GenerateRangeTables.C creates three files: SR.IN, SR.OUT, and test.out. SR.IN serves as an input file to the SRIM Module which is executed within the macro. SR.OUT is then output from SRIM, and is used to create test.out which contains the data tables that ST_MoNA will use.

NOTE: In order to execute the SRModule it may be necessary to have the aforementioned files in the directory where you run the ROOT macro GenerateRangeTables.C:

- SCOEF03.dat
- SNUC03.dat

To create a new target start ROOT and execute GenerateRangeTables.C.

```
*****
*                                     *
*           W E L C O M E  t o  R O O T           *
*                                     *
*   Version   5.12/00           10 July 2006   *
*                                     *
*   You are welcome to visit our Web site   *
*           http://root.cern.ch           *
*                                     *
*****

Compiled on 11 July 2006 for win32.

CINT/ROOT C/C++ Interpreter version 5.16.13, June 8, 2006
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0] .x GenerateRangeTables.C _
```

You will be prompted to enter:

- Target Name
- Number of Elements
- Density, State of Matter (solid = 0, gas = 1)
- Z, Name, %Abundance, A

In this example we will use ^9Be :

```
root [0] .x GenerateRangeTables.C
Info in <TGeoManager::TGeoManager>: Geometry geom, radionuclides created
Enter the name of the absorber (don't use spaces!) 9Be
Enter num elements in absorber 1
Enter the target density (g/cm^3), and solid(0)/gas(1) 1.85 0
Enter z name %abundance a of target 1 4 9Be 100 9_
```

For targets with multiple elements you will be prompted multiple times for each element. Once all the prompts are entered, the macro will run and produce dE/dx and range tables for a range of Z (1-92) in a data file called test.out. It is recommended to rename the file to something more meaningful like Your_Element_Stopping.dat. Now that the stopping file is generated it is necessary to move it to the bin/srim directory in your ST_MoNA installation.

4.3 Changing the Energy Range of CycSrim

The maximum energy that CycSrim can interpolate to is limited by the maximum energy SRIM can accept: 2 GeV/u. To alter the energy range of your data file, Your_Element_Stopping.dat, you must edit two lines in the macro GenerateRangeTables.C. In GenerateRangeTables.C there is a function called WriteSRIM(), which creates an input file for SRIM. Within this function is a series of nested for loops that create the SR.IN file to be read by SRIM.

Set the two values in the definition of minEnergy and maxEnergy to your desired minimum and maximum energy respectively.(Note the units!) They are located within WriteSRIM() at approximately line 157 of GenerateRangeTables.C:

```

1 else if(iline == 14) {
   Double_t minEnergy = 0.1 * a * 1000.; //factor 1000 for keV
3   Double_t maxEnergy = 200. * a * 1000.; //factor 1000 for keV
   out<<Form(" %9.3f %9.3f ",minEnergy,maxEnergy);
5 }

```

where the first coefficient is the desired energy in units of MeV, a is the number of nucleons, and 1000 is the conversion factor from keV to MeV. In this case the maximum energy is set to 200 MeV, so maxEnergy should be: $\text{maxEnergy} = 200.0 * a * 1000$ [MeV], where the 200 is the desired maximum energy in MeV. Again, the maximum energy the interpolater can handle is limited by SRIM, which is 2 GeV/u. WARNING: Although CycSrim may be able to handle energy around 2 GeV/u, there is no guarantee the rest of ST_MoNA can. You may run into trouble with cross sections not being defined in MENATE_R for high energies, in addition the GEANT4 Physics lists might have to be changed.

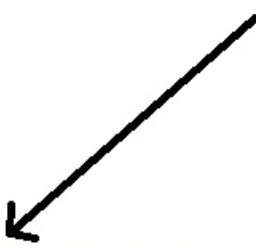
4.4 Adding the Target to CycSrim

To add the newly generated stopping file to CycSrim open both the source and header files: CycSrim.cxx and CycSrim.h. At line 114 in CycSrim.h you must add the name of your material to the list of current targets in the format SrimMaterial_Your_Element, and expand the size of the list by one (SrimNumberOfMaterials):

```

enum EMaterials {
  //materials
  SrimMaterialSi           = 0,
  SrimMaterialMylar        = 1,
  SrimMaterialIsobutane    = 2,
  SrimMaterialPentane      = 3,
  SrimMaterialAluminum     = 4,
  SrimMaterialNickel       = 5,
  SrimMaterialBC420        = 6,
  SrimMaterialCsI          = 7,
  SrimMaterialCarbon       = 8,
  SrimMaterialAu           = 9,
  SrimMaterialPolypropylene = 10,
  SrimMaterialLowDensityPolyethylene = 11,
  SrimMaterialTh           = 12,
  SrimMaterialBeryllium    = 13,
  SrimNumberOfMaterials    = 14, //numb material (largest indx+1)
  SrimMaxNumberOfEnergies  = 200 //upper_limit
};

```



The value of SrimMaxNumberOfEnergies defines the maximum number of data points to be read in for a given Z. Even at the maximum energy of 2 GeV/u, the number of energies is less than 150. SrimNumberOfEnergies should not have to change.

In CycSrim.cxx at line 30 you must now add the name of the stopping file you created with GenerateRangeTables.C:

```
const Char_t * srimfile[] = {  
    "SiliconStopping.dat",  
    "MylarStopping.dat",  
    "IsobutaneStopping.dat",  
    "PentaneStopping.dat",  
    "AluminumStopping.dat",  
    "NickelStopping.dat",  
    "BC420Stopping.dat",  
    "CsIStopping.dat",  
    "CarbonStopping.dat",  
    "AuStopping.dat",  
    "PolypropyleneStopping.dat",  
    "LowDensityPolyethyleneStopping.dat",  
    "ThStopping.dat",  
    "BerylliumStopping.dat"  
};
```



WARNING: BE SURE TO SET THE DATA PATH CORRECTLY! In CycSrim.cxx at line 471 is a function called GetSrimDataDir(). You must change the string CYC_SRIM_DATA_DIR to match the location of ALL your stopping files. There is probably a smart way to set this as an environmental variable, but that is not implemented yet. Be careful to not forget the commas after adding in the new material. Once the list is expanded in both the header and source files we can now move on to initializing the target in ST_MoNA.

4.5 Changes to ST_MoNA

In `st_mona.cc` at approximately line 1572 is the initialization of the `CycSrim` object. An example initialization of a `CycSrim` target is shown below:

```
1 CycSrim* srimTarget = new CycSrim(CycSrim::Name_of_Material, Thickness,
    CycSrim::My_Favourite_Units);
```

where `CycSrim::Name_of_Material` is the name of the material as defined in `CycSrim.h`, and `CycSrim::My_Favourite_Units` defines which unit system you want to use for a given Thickness. The following options are available:

- `kUnitsMgCm2`
- `kUnitsCm`
- `KUnitsMicron`
- `KUnitsTorr`
- `kUnitsMbar`

It is important to note that `CycSrim` does not do any energy straggling, and that angular straggling is handled by a different function which calls on the object `matTarget` – a member of `StMaterial`. Thus it is still necessary to define the target as BOTH a `CycSrim` object and as a `StMaterial`. It is also important to name your target “`srimTarget`” since the subsequent code that passes this object to `st_par_mat_ia.cc` where the energy loss calculation is done, uses this variable name. In addition, initializing the target in this manner causes the code to read the entire stopping file, hence if you create your target many times you will quickly run out of memory.

Due to the structure of `ST_MoNA` as a whole, which chains many executables together, the target needs to be defined again in `mona_analysis.cc` where the decay energy reconstruction is performed.

The `srimTarget` is most easily included in the definition of the fragment for your experiment in the function `MonaAnalyzedEvent()`, an example declaration is shown below:

```
1 else if (frag == "24O_Jones") {
    ...
3     e->targA = 9;
    e->targZ = 4;
5     CycSrim *srimBe = new CycSrim(CycSrim::SrimMaterialBeryllium, e->
    dTarg, CycSrim::kUnitsMgCm2);
    e->setCycSrim(srimBe);
```

7 | }

Be sure to change targA and targZ to the appropriate values for your target. Once this is done, compile and enjoy your new target. It is strongly recommended to cross-check the energy loss values with LISE++.

WARNING: In st_reaction.cc around line 1627, for the StReacStripEvapTwoBody, the mass of the target is *hard coded*. Thus if your reaction relies on the 2-body code, you will have to change the following:

```
1 double Etot_lab = p->getEtotal() + 9.0*931.494028; //adds beam energy to  
   tgt for total energy.
```

where the term 9.0*931.494028 is adding in the rest energy of the target:

$$E_{target} = A_{target} * 931.5 [MeV/u]$$

Hence, the 9.0 is specific to ^9Be (or really anything with $A = 9$). If your target has a different mass, the 9.0 will need to be changed to your target's A .

5.0 Installing CycSrim

This section details a way to install CycSrim by itself, should you want to upgrade the energy loss interpolator but not want the changes that come with the 4-neutron code. Lucky for you, I made a backup version of ST_MoNA once I got CycSrim working way back in October 2012. In principle these should be the minimal changes you need to get CycSrim running.

New Files you will need:

- CycSrim.cxx (place in src directory)
- CycSrim.h*
- A new Makefile
- Stopping files for each material. You need to make a directory called srim in your bin and place these files there:
 - BerylliumStopping.dat
 - DeuteriumStopping.dat
 - KaptonStoppping.dat

– NickelStopping.dat

Existing Files that will need to be changed:

- st_mona.cc
- mona_analysis.cc
- mona_analysis.hh
- st_material.cc
- st_par_mat_ia.cc
- st_par_mat_ia.hh

IMPORTANT: In CycSrim.cxx the directory to the stopping.dat files is hard coded.

From my notes following the Wheezy Apocalypse:

There seems to be an issue with the GetSrimDataDir() function. It returns a TString but this isnt being recognized by the Form function in the initialization of cycsrim. I.e. it cant find the data file. A quick fix for this is to open CycSrim.cxx and go to CycSrim::Init(). On the line that says:

```
1 srimFile = Form(    %s/srim/%s    ,GetSrimDataDir() ,srimfile[fMaterial])
```

Remove the first %s and replace it with the path to your srim directory. Then remove GetSrimDataDir() from the function argument. This is a crude workaround, but it should be ok since the function GetSrimDataDir() is only called here.

Ok, so if your version of these files is identical to this old version of ST_MoNA then you should be able to just copy everything, compile and be done. However thats probably too good to be true. The following details incorporating CycSrim into ST_MoNA:

Compile CycSrim with ST_MoNA

Add:

```
1 #include "CycSrim.h"
```

To the list of included files in st_mona, mona_analysis, st_par_mat_ia, and st_material (headers included). **NOTE:** This is probably redundant and has something to do with the fact that CycSrim compiles twice. Partly because I didnt know a priori where CycSrim belonged in ST_MoNA. But mostly because mona_analysis is a separate program from st_mona. **WARNING:** Before you make st_mona, you must make a dictionary file for CycSrim.cxx. This is done by typing:

```
1 make CycSrimDict.cxx
```

into the terminal. Unless you change CycSrim.cxx this only has to be done once. Once this file is made, you should be able to compile ST_MoNA normally. If we can compile, then we should be able to access the classes within CycSrim.

Instantiate the Target in ST_Mona.cc

When you make a CycSrim object you also read in the entire .dat file (tables of dE/dx), which if repeated a million times will just be unacceptably slow and stupid. So we initialize the target once in st_mona.cc and pass it to the rest of the code. To do this, go to the portion of st_mona.c where matTarget is made (around line 1570 in this version). Type:

```
1 CycSrim *srimTarget = new CycSrim(CycSrim::SrimMaterialBeryllium ,  
    dTarget , CycSrim::kUnitsMgCm2);
```

NOTE: Its important to call the pointer srimTarget as the rest of the code refers to it this way. Shortly after the creation of the rageny object, add the last line (6):

```
1 // use lookup table for energy if there are more than 100 events  
  StRageny* rageny;  
3 rageny = new StRageny(1); // always use interpolator!!!  
  rageny->setRngUniform(new StRNGUniformGSL(r));  
5 rageny->setRngGaus(new stRNGGaussGSL(r));  
  rageny->setCycSrim(srimTarget);
```

Inform st_par_mat_ia.cc of its new target

In the function:

```
double StRageny::do_de(StMaterial* m, double thickness , StParticle* p,  
    int range)
```

You will find the grave of Rageny. Do not be alarmed.

```
1 /*  
  //-----\\  
3  ||          HERE LIES THE RAGENY ENERGY          ||
```

		LOSS INTERPOLATOR.	
5		IN COMMEMORATION OF IT'S BRAVE	
		SERVICE TO THE MONA COLLABORATION	
7		R. I. P.	
		10/10/2012	
9	\\	_____	//
	*/		

Brutally delete all the code in the function and replace it with:

```

2 // CycSrim Energy Loss
4 if (thickness <0) ERR("thickness <0");
6 if( thickness == 0) return 0;
8 if (p->getZ() < 0) ERR("Don't know how to deal with negatively
   charged particles!");
10 if (p->getA() <= 0) ERR("Don't know how to deal with particles with A
   <= 0!");
12 double dE; // energy loss
14 double etotal = p->getGammaMinOneU()*(p->getA()); // total kinetic
   energy
16 fSrimTarget->SetThickness(thickness);
18 dE = fSrimTarget->CycSrim::GetEnergyLoss(p->getZ(), p->getA(), etotal);
   dE = dE/(p->getA()); // MeV/u could also do dE /= p->getA();
20 double final_ke = etotal/(p->getA()) - dE; // calculate final kinetic
   energy [MeV/u]
22 p->setGammaMinOneU(final_ke);
24 return dE;

```

As you might expect, this kill Ragny. Immediately following this function are two others, StRagny::range and StRagny::energy. They must go. Replace them with:

```

1 // CYCSRIM range and dE
2
3 double StRageny::range(CycSrim* fSrimTarget, StParticle* p){
4     double etotal = p->getGammaMinOneU()*(p->getA());
5     double range = fSrimTarget->Getrange(p->getZ(), p->getA(), etotal,
6     kTRUE);
7     return range;
8 }

```

```

9
double StRageny::energy(cycSrim* fSrimTarget, double thickness,
    StParticle* p){
11     double etotal = p->getGammaMinOneU()*(p->getA());

13     fSrimTarget->SetThickness(thickness);
    double dE = fSrimTarget->cycSrim::GetEnergyLoss(p->getZ(), p->getA
        (),etotal);
15     dE = dE/(p->getA());

17     return dE;

19 }

```

These functions are used with mona_analysis.cc.

Add the function "setCycSrim" to st_par_mat_ia.hh

As a public method:

```

1 public:
    void setRngUniform(stRNGUniform* p) {fRngUniform = p;}
3    void setRngGauss (stRngGauss* p) {fRngGauss = p;}
    void setCycSrim (CycSrim* p) {fSrimTarget = p;}

```

In the class stParMatInteraction replace the rageny range and energy functions with:

```

// CycSrim Definitions
2 virtual double range(CycSrim* fSrimTarget, StParticle* p) = 0;
virtual double energy(CycSrim* fSrimTarget, double thickness,
    StParticle* p) = 0;

```

In the class StRageny add the same lines again, but like this:

```

1 // CycSrim Definitions
double range(CycSrim* fSrimTarget, StParticle* p);
3 double energy(CycSrim* fSrimTarget, double thickness, StParticle* p);

```

Be sure to define your pointers in the private variables:

```

1  /// RNGs needed for straggling
   StRNGUniform*   fRngUniform;
3  StRNGGauss*     fRngGauss;
   CycSrim*        fSrimTarget;

```

Add the function "setCycSrim" to mona_analysis.hh:

In the class MonanAnalyzedEvent, add the function:

```

void setCycSrim(CycSrim* fSrimTarget){srimTarget = fSrimTarget;}

```

And define the srimTarget:

```

1  public:
   StMaterial* matTarget; //! material
3  //StMaterial* matTest;
   CycSrim* srimTarget;

```

Instantiate the target in mona_analysis.cc:

Around line 2000 the MonanAnalyzedEvent is created, and we should add the target to it:

```

cycSrim *srimBe = new CycSrim(CycSrim::SrimMaterialBeryllium, e->dTarg
    , CycSrim::kUnitsMgCs2);
2 e->setCycSrim(srimBe);

```

Eviscerate the Rageny calculation in mona_analysis.cc:

We don't want Rageny anymore so in mona_analysis.cc, around line 1350, we add back the energy loss from a half-target. The code needs to be:

```

// adding the manual control of deltaE for Schiller map
2 if (manDESitch == 1){
   deltaE[1] = mandDe;
4 }
   else {
6   p.setEkin(e0);
   deltaE[1] = p.getA()*(fRageny.energy(srimTarget, dTarg/2,&p));
8 }
// CycSrim Implementation
10 p.setEkin(e);

```

```

12 // Calculate energy loss at dTarg/2 from CycSrim:
   deltaE [2] = p.getA()*(fRageny.energy(srimTarget,dTarg/2,&p));

```

Even though it says fRageny, it's not! Remember you gutted it. If you have continued problems, feel free to contact the Author.

6.0 Multi-Neutron Reactions (4n and 3n)

Here we cover the implementation of the 4n and 3n reactions in ST_MoNA, some of the assumptions that are made, and most importantly **their limitations**. Not every possible combination for the breakup of 4n (or 3n) is currently coded into this version of ST_MoNA (although in principle they can). For example, dineutron to dineutron is NOT implemented, or 1n, to dineutron to 1n is also NOT implemented – *but they can be*. The power is yours. Some very important things to know about the 3n and 4n reactions:

- They are coded as an extension of the 2n implementation. This means that simulations time scales linearly with the number of neutrons (roughly). Each neutron is handled individually by GEANT, *one-at-a-time*. This means that typically, a 4n simulation will take twice as long as a 2n simulation simply because GEANT is called 4 times compared to 2. The execution of ST_MoNA is not slowed down in this fashion (but may be depending on how you choose your random numbers from your excitation distribution etc.) As far as I know, GEANT does not handle multiple-threads so it can't do them simultaneously (but you might be able to spawn 4 instances). This might sound minor, but if you plan to do hundreds of simulations it can add up.
- None of the GEANT code has changed. Everything else has however (st_mona, root2pythia, mona_analysis), as additional branches to the ROOT files have been added.
- The code for these lives in st_reaction.cc and are labeled (1n,2n,3n,4n)
- Proper anti-symmetrisation of the neutrons **IS NOT DONE FOR YOU**. This is imbedded in the model you choose. You can select a distribution for each individual neutron, but whether or not that distribution is physical and accounts for the other neutrons is not checked at all. It is possible to tell which neutron is which pre-geant.
- **E and P are only partially conserved!** In the limit of no energy loss from the target, no glauher kick, and no decay energy. Energy and Momentum will be

perfectly conserved *IF* the mass of a neutron is equal to 1 amu. (It's not, its about 8 MeV fatter). This is because we approximate $m_n \sim 1amu$ and $M_{frag} \sim A * 1amu$ in our code. This does not matter for the decay energy because these terms will cancel out, however it does slightly perturb the momentum. For a 4n decay, simply comparing initial and final states gives a $\delta E_{tot} = 32MeV$, which is an effect on the order of 0.3% – so it's not a big deal *but you should be aware*.

6.1 Available Decay Modes

As stated earlier, not every possible decay mode is implemented so here are the lists of available reactions and their decays. Quotations indicate the string used for the reaction or decay flag when running ST_MoNA.

"3neutron"

Currently performs a 1n decay followed by either a dineutron or a sequential decay. The first neutron is set by "-e1 my_dist." and the second by "-e2 other_dist". The third neutron is set implicitly.

"4body_decay"

Performs the 4body phase space decay. No weighting, and no correlations. The distribution for frag+3n is set by "-e my_dist." Currently no other decay modes accompany this reaction flag.

"4neutron" Performs a 2n decay where the first two neutrons can have independent distributions. The 2n decay is then followed by a sequential decay. It is also possible to instead specify the distribution of each neutron. In this model each neutron will come off *one at a time*, with e1 being the first to decay. This is useful if, for example, we want to do a 4-neutron thermal.

"5body_decay" Performs a 5-body phase-space decay from a single input distribution. No weighting and no correlations. The distribution for frag+4n is set by "-e my_dist". Currently no other decay modes accompany this reaction flag.

"5body_seq" Performs a 2n decay where the first two neutrons can have independent distributions. The 2n decay is then followed by a dineutron emission. The default is a 5body phase-space decay (borrows from the same reaction class as 5body_decay, just this one can accept the dineutron argument).

Individual neutrons can be set with "-e1, -e2, -e3" or "-e4", following the appropriate reaction flag. e1 is the FIRST neutron to decay (up to e4, the last). Besides the typical Breit-Wigner, thermal, swaves, distributions that were previously available for e1 and e2, e2 has the additional:

"4body_dineutron" For use with the "3neutron" reaction flag. It models the decay of 1n followed by a dineutron emission. The distribution for the first neutron is chosen by "-e1 my_dist," and the second by "-e2 4body_dineutron \$Energy \$Width \$Scattering_Length." The third neutron is implicitly set by the dineutron decay. An example way to setup this reaction is:

```
arguments="-exp 09067_He10 -geant -reac 3neutron -e1 therm ${P1} -e2 4
body_dineutron ${P2} ${P3} ${P4} -nn 1 -np 2 -n ${P5}"
```

Here the first neutron is a thermal neutron followed by the dineutron decay. Be sure to set -nn and -np properly or your cosymap will not work.

"Volya_2nSeq" For use with the "3neutron" reaction flag. This models the decay of 1n followed by a sequential decay. The distribution for the first neutron is chosen by "-e1 my_dist," and the second by "-e2 Volya_2nSeq P1 P2 P3 P4 P5" This decay has the option of adding an angular correlation between the two-neutrons decaying in the sequential decay given by the square of a legendre polynomial with an L-value set by "-legendre_lvalue \$L." To remove this correlation simply specify an L-value less than zero. This will still perform the sequential decay but will not correlate the two-neutrons. An example way to setup this reaction is:

```
1 arguments="-exp 09067_He10 -geant -reac 3neutron -e1 therm 4.0 -e2
Volya_2nSeq ${P2} ${P3} 1.0 0.5 ${P4} ${P5} -nn 1 -np 2 -n ${P6} -
strag 1 -glaub 1.0 -velocityshift 0.97 -asymMomentum 1.13 -
legendre_lvalue ${P4}" #-n number of events.
```

The third neutron (-e3) has the options:

"5body_dineutron" For use with the "5body_seq" reaction flag. This models the decay of 2n followed by a dineutron emission. The distribution of the first neutron is chosen by "-e1 my_dist," and the second by "-e2 my_dist." The third neutron is set by "-e3 5body_dineutron \$Energy \$Width \$Scattering_Length" which implicitly sets the fourth neutron. **WARNING:** If the "5body_seq" flag is chosen and "-e3" is set to *any other available distribution* the third and fourth neutrons will default to a **3-body phase space decay!** If the inputs are not in the order st_mona expects, it will not run. In addition, you may fool yourself here. Sequential decays are handled with a different reaction flag. An example way to setup this reaction is:

```

1 arguments="-exp 09067_He10 -geant -reac 5body_seq -e1 therm ${P1}
3   -e2 therm ${P2} -e3 5body_dineutron ${P3} ${P4} ${P5} -n ${P6}"

```

"Volya_2nSeq" For use with the "4neutron" reaction flag. This models the decay of 2n followed by a sequential decay. The first neutron is chosen by "-e1 my_dist" and the second by "-e2 my_dist." The third neutron is set by "-e3 Volya_2nSeq P1 P2 P3 P4 P5" which implicitly sets the fourth. This decay has the option of adding an angular correlation between the two-neutrons decaying in the sequential decay given by the square of a legendre polynomial with an L-value set by "-legendre.lvalue \$L." To remove this correlation simply specify an L-value less than zero.

The fourth neutron (e4) currently has no additional distributions from the standard.

7.0 Misc. Errata

Here are some notes for some minor additions:

Bulging Target

The geometry for a bulged target (useful for the LD2 target) has been coded into the energy loss calculation (and add-back in mona.analysis.cc). The "bulge," defined from the center of a flat target to the center of a spherical cap above it is set by the input file with the "maxbulge" variable. The kapton foils of the target are currently also hard-coded into the simulation, their thickness is set by "tflatfoil" and they follow the geometry of the bulge. **SET THESE VARIABLES TO ZERO (OR SOMETHING REALLY SMALL LIKE 10^{-12} IF YOU DESIRE A FLAT TARGET WITH NO FOILS AROUND IT.)** As a part of this code, ST_MoNA will spit out a warning if the beamspot is larger than the size of the foil. This really doesn't affect anything in the flat-foil limit. It just means your forward tracking placed the particle outside a radius of 1.9 cm.

New RNG

The StRNGLegnedre function has been added to the list of available random number generators, this allows one to sample a random number from a Legendre Polynomial with a user-specified L-value.

Momentum Dists.

The parallel and perpendicular momentum distributions of the fragment have been

added to mona_analysis.cc, along with the total momentum. These variable names are: Ppar_Frag, Pperp_Frag, and Ptot_Frag.

Phase Space Weighting

The option to weight events in a phase-space decay has been added. This is set with the argument "-phase_space_weight," a value of 0 is no weighting, a value greater than zero will perform the weighting. This has only been implemented for the 3body phase space decay, (2N reaction), but can easily be extended to the 3 and 4N reactions. The only files you will have to edit will be st_mona.cc (as it is added as an input to the reaction), st_reaction.hh (the function definition needs to change slightly), and st_reaction.cc (where you do the weighting).

New s-wave lineshape

The s-wave lineshape is now calculated slightly differently following the formalism of Y. Aksyutina, Phys. Lett. B 666, 430 (2008).

8.0 Sample Files

Here is a sample script I use to run the 5-body phase space decay of "12He":

8.1 4n Phase Space Example

```
#!/bin/sh
2
#First implementation to save time in running simulations.
4 #Adjust the FileName and the arguments, then save the shell script as a new
  file.
#Run the shell script to run through all three portions of the simulation.
6 #This will allow for ease of archival and reproduction of simulations.

8 P1=$1 #5-body Resonant Energy
  P2=$2 #5-body Resonant Width

10
  FileName="Be14_He8+4n" # Tag for output files
12
# These arguments are fed to ST\MoNA and set the reaction and decay mode.
  They are:(in order)
14 # -experiment, -use geant, -use 5body_decay reaction
  # -e energy P1 width P2 L-value 1, -n Do 10^5 events
16 # -yes straggling, -yes glauverkick, -use velocity shift, -set asmMomentum
  arguments="-exp 09067_He10 -geant -reac 5body_decay -e asymbw ${P1} ${P2} 1
    0 -n 100000 -strag 1 -glaub 1.0 -velocityshift 0.97 -asymMomentum 1.13"
  # -n number of events.
```

```

18
20 #for geant simulation through python script control.py (-n number of
    neutrons)
    # You must set this to four, this tells geant that it needs to take care of
        4 neutrons
22 controlargs="-n 4"

24 #fragment for mona_analysis.cc
    FRAGname="He8_09067"
26
    #MoNA Detector Config File
28 MoNADetFile="detector_config_jesse09067"

30 #Change the random seed if long runs are needed, but you want to do them as
    several short runs.
    export GSL_RNG_SEED=527391
32
    #Paths to access the executables, and to save the data files.
34 #You will have to change these
    analysishpath="/user/jonesm/tetra_neutrons/He10"
36 stmonapath="/projects/mona-sim/jonesm/st_mona"
    n2geantpath="/projects/mona-sim/jonesm/n2_geant"
38 DataPath="/mnt/simulations/MoNA_EfficiencyTest/Be14_He10/fine_mesh/4
        N_Simulations/5body/"

40 #used to allow multiple geant runs simultaneously
    UniqueID="_5body_E${P1}_G${P2}_L1"
42
    #Make file names based on the file name given.
44 PreGeant=${DataPath}/${FileName}"${UniqueID}.root"
    PostGeant=${DataPath}/${FileName}"-geant${UniqueID}.root"
46 FinalName=${DataPath}/${FileName}"-geant-anal${UniqueID}.root"
    HistDumpName=${DataPath}/${FileName}"-hist${UniqueID}.root"
48
    #Run the simulations.
50
    #st_mona simulation
52 cd ${stmonapath}
    ./bin/st_mona $arguments -f ${PreGeant}
54
    #geant afterburner
56 #Using light threshold of 0.15 MeVee.
    cd ${n2geantpath}/control
58 control.py -i ${PreGeant} -o ${PostGeant} -t 0.38 ${controlargs} -u ${
        UniqueID} -d ${DataPath} -c ${MoNADetFile}

```

```

60 #MoNA Analysis
    #Needed for reconstructed target parameters
62 cd ${stmonapath}
    ./bin/mona_analysis -frag ${FRAGname} -if ${PostGeant} -of ${FinalName}
64
    #Dump Edecay Histograms (Cleanup code to save space, you will likely have
        something different)
66 #needed for running large jobs and minimzing space used
    cd ${analysispath}
68 root -q -b 'DumpHistsMoNASim.12He.C(" ${HistDumpName}" , " ${PostGeant}" , " "
        ${FinalName}" )'

70 #Delete created event (tree) root files
    # Here I go through and delete files I don't need anymore
72 cd ${DataPath}
    rm neutron*
74 rm ${PreGeant} ${PostGeant} ${FinalName}

```

8.2 A working bashrc

Here is a working bashrc file post Wheezy:

```

2 #echo sourcing .bashrc

4 if [ -f ~/.bash_functions ]; then
    . ~/.bash_functions
6 fi
    if [ -f ~/.bash_aliases ]; then
        . ~/.bash_aliases
    fi
10 #bash_env
    if [ -f ~/.bash_env ]; then
        . ~/.bash_env
12 fi

14 if test -n "$(typeset -F|grep \ module\$)"; then
16     module load null root texlive geant4 gcc
    fi

18 export PATH=${PATH}":${G4WORKDIR}/bin/${G4SYSTEM}"
20
    #if [ ${TERM} != "dumb" ]; then

```

```

22 # source /opt/lucid/GEANT4/current/env.sh
    #fi
24
    export PS1='\[[1m\]<\h:\W >\[[m\] '
26 export RSYNC_RSH='ssh'

28 # add the current directory to the PATH

30 PATH=${PATH}:.
    umask 002
32

34 PATH=${PATH}:.

36 #Paths created by zwk
    #export ROOTSYS=/opt/lucid/root/current
38 export ROOTSYS=${ROOTSYS}
    #export ROOTSYS=/opt/lucid/root/5.26.00/
40 export CVSROOT=:ext:monasoft@nscglw2.nsc1.msu.edu:/user/monasoft/cvsroot/
    #export CVSROOT=:ext:zkohley@guardia.tamu.edu:/home/jbngroup/jbngroup/
        CVSROOT/
42 #export CVS_RSH='ssh'
    export PATH=${PATH}:/${ROOTSYS}/include/root:/user/jonesm/Simulations/
        st_mona/bin/
44 export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}
    #export MENATEG4XS=/projects/mona-sim/hope-geant/MENATE_R_CrossSections/
46 export MENATEG4XS=/projects/mona-sim/kohley/MENATE_R_CrossSections/

48
    #source the root directory
50 source $ROOTSYS/bin/thisroot.sh
    #source /opt/lucid/GEANT4/geant4.8.2.p01/env.sh
52 #source /opt/lucid/GEANT4/current/env.sh

54 #if [ -f /opt/lucid/GEANT4/current/env.sh ];
    #then
56 # source /opt/lucid/GEANT4/current/env.sh
    #fi
58

60

62 #aliases
    alias ls='ls --color=auto'
64
    case $OSTYPE in

```

```

66 Linux*)
    export PATH
68 ;;
70 SunOS*)
    export PATH
72 ;;
74 OSF*)
    export PATH
76 ;;
78 esac
80
#source ~/.bash_env

```

8.3 A working rootrc

Here is a working rootrc post wheezy

```

1 # @(#)root/config:$Id: rootrc.in 39275 2011-05-19 18:17:37Z pcanal $
# Author: Fons Rademakers 22/09/95
3
# ROOT Environment settings are handled via the class TEnv. To see
5 # which values are active do: gEnv->Print().
7
# Path used by dynamic loader to find shared libraries.
# This path will be appended to the (DY)LD_LIBRARY_PATH on Unix
9 # and to PATH on Windows.
# Paths are different for Unix and Windows. The example shows the defaults
11 # for all ROOT applications for either Unix or Windows.
Unix.*.Root.DynamicPath:    .: $(ROOTSYS)/lib:
13 WinNT.*.Root.DynamicPath:  .; $(ROOTSYS)/bin;
15
# Path used to find macros.
# Paths are different for Unix and Windows. The example shows the defaults
17 # for all ROOT applications for either Unix or Windows.
Unix.*.Root.MacroPath:      .: $(ROOTSYS)/macros:
19 WinNT.*.Root.MacroPath:    .; $(ROOTSYS)/macros;
21
# Path used to find plugin macros.
# Paths are different for Unix and Windows. The example shows the defaults
23 # for all ROOT applications for either Unix or Windows.

```



```

Unix.*.Root.PluginPath:      :$(ROOTSYS)/etc/plugins:
25 WinNT.*.Root.PluginPath:    ;$(ROOTSYS)/etc/plugins;

27 # Path where to look for TrueType fonts.
Unix.*.Root.UseTTFonts:      true
29 *.*.Root.TTFontPath:       $(ROOTSYS)/fonts

31 # Use Net* API functions.
WinNT.UseNetAPI:             true
33
35 # Use thread library (if exists).
Unix.*.Root.UseThreads:      false

37 # Select the compression algorithm (0=old zlib , 1=new zlib)
# Note, setting this to '0' may be a security vulnerability.
39 Root.ZipMode:              1

41 # Show where item is found in the specified path.
Root.ShowPath:               false
43
45 # Activate malloc/new, free/delete calls via the TMemStat class
# the parameter buffersize is the number of calls to malloc or free that
# can be stored in one memory buffer.
# when the buffer is full , the calls to malloc/free pointing to the same
# location
47 # are eliminated and not written to the final Tree. The default value
# 100000
# is such that between 50 and 90% of the calls are eliminated depending on
# the application.
49 # the parameter TMemStat.maxcalls is the maximum number of new/delete calls
# to be monitored
# 5 million calls is a reasonable number.
51 # if your code has been compiled with -fno-omit-frame-pointer you can
# specify
# gubuiltin for Root.TMemStat.system. In this case the backtrace is much
# faster.

53 Root.TMemStat:              0
Root.TMemStat.buffersize:    100000
55 Root.TMemStat.maxcalls:     5000000
#Root.TMemStat.system:       gnubuiltin
57 Root.TMemStat.system:

59 # Activate memory statistics (size and cnt is used to trap allocation of
# blocks of a certain size after cnt times).
61 Root.MemStat:               0
Root.MemStat.size:           -1

```

```

63 Root.MemStat.cnt:      -1
   Root.ObjectStat:      0
65
   # Activate memory leak checker (use in conjunction with $ROOTSYS/bin/
     memprobe).
67 # Currently only works on Linux with gcc.
   Root.MemCheck:        0
69 Root.MemCheckFile:     memcheck.out

71 # Global debug mode. When >0 turns on progressively more details debugging.
   Root.Debug:           0
73 Root.Stacktrace:       yes
   # Allow for a customized backtrace script.
75 #Root.StacktraceScript: $(ROOTSYS)/etc/gdb-backtrace.sh
   # Allow for a customized backtrace messages (e.g. referencing your own
     project
77 # bug tracking system). Change this message rather than the entire script.
   # The % characters will be replaced by newlines.
79 #Root.StacktraceMessage: The lines below might hint at the cause of the
     crash.%If they do not help you then please submit a bug report at%http
       ://myproject/bugs. Please post the ENTIRE stack trace%from above as an
       attachment in addition to anything else%that might help us fixing this
       issue.

81 # Ignore errors lower than the ignore level. Possible values:
   # Print, Info, Warning, Error, Break, SysError and Fatal.
83 Root.ErrorIgnoreLevel: Print

85 # Settings for X11 behaviour.
   X11.Sync:              no
87 X11.FindBestVisual:     yes
   X11.UseXft:            no
89 X11.XInitThread:        yes

91 # Settings for Win32 behavior.
   Win32.UseSysPointers:  no
93
   # Default editor.
95 Unix.*.Editor:         vi
   WinNT.*.Editor:        notepad
97
   # Default 3d Viewer.
99 # By default 3-D views are shown in the pad,
   # if the next line is activated, the default viewer will be OpenGL.
101 #Viewer3D.DefaultDrawOption: ogl

```

```

103 # Default Fitter (current choices are Minuit and Fumili).
    Root.Fitter:                Minuit
105
    # Specify list of file endings which TTabCom (TAB completion) should ignore
    .
107 #TabCom.FileIgnore:          .cpp:.h:.cmz

109 # TCanvas specific settings. Opaque move and resize show full pad during
    # the operation instead of only the outline. Especially for resize you'll
111 # need serious CPU power. UseScreenFactor=true means to size canvas
    according
    # to size of screen, so a canvas still looks good on a low resolution
113 # laptop screen without having to change canvas size in macros.
    # HighLightColor 2 = red. ShowEventStatus allows the event status bar to
115 # be turned on by default. AutoExec allows TExec objects to be executed
    # on mouse and key events.
117 Canvas.MoveOpaque:           false
    Canvas.ResizeOpaque:         false
119 Canvas.UseScreenFactor:       false
    Canvas.HighLightColor:        2
121 Canvas.ShowEventStatus:       false
    Canvas.ShowToolTips:          false
123 Canvas.ShowToolBar:           false
    Canvas.ShowEditor:            false
125 Canvas.AutoExec:              true
    Canvas.PrintDirectory:         .
127 #set the default precision when writing floating point numbers in TCanvas::
    SaveSource
    Canvas.SavePrecision:         7
129 # Set the default TStyle
    # Predefined styles are: Plain, Bold, Video, Pub, Classic, Modern.
131 Canvas.Style:                 Modern

133 # Printer settings.
    #WinNT.*.Print.Command:       AcroRd32.exe
135 #Unix.*.Print.Command:        a2ps -P%p —landscape —columns=2 —margin=30
    -rf8.0 %f
    #Print.Printer:                32—rb20—hp
137 #Print.Directory:              .
    #Print.FileType:               pdf
139
    # Default histogram binnings for TTree::Draw().
141 Hist.Binning.1D.x:             100

143 Hist.Binning.2D.x:             40
    Hist.Binning.2D.y:             40

```

```

145 Hist.Binning.2D.Prof:          100

147 Hist.Binning.3D.x:            20
    Hist.Binning.3D.y:          20
149 Hist.Binning.3D.z:            20
    Hist.Binning.3D.Profx:       100
151 Hist.Binning.3D.Profy:       100

153 # Default statistics parameters names.
    Hist.Stats.Entries:    Entries
155 Hist.Stats.Mean:         Mean
    Hist.Stats.MeanX:      Mean x
157 Hist.Stats.MeanY:       Mean y
    Hist.Stats.MeanZ:      Mean z
159 Hist.Stats.RMS:         RMS
    Hist.Stats.RMSX:       RMS x
161 Hist.Stats.RMSY:       RMS y
    Hist.Stats.RMSZ:       RMS z
163 Hist.Stats.Underflow:   Underflow
    Hist.Stats.Overflow:   Overflow
165 Hist.Stats.Integral:    Integral
    Hist.Stats.Skewness:   Skewness
167 Hist.Stats.SkewnessX:   Skewness x
    Hist.Stats.SkewnessY:   Skewness y
169 Hist.Stats.SkewnessZ:   Skewness z
    Hist.Stats.Kurtosis:   Kurtosis
171 Hist.Stats.KurtosisX:   Kurtosis x
    Hist.Stats.KurtosisY:   Kurtosis y
173 Hist.Stats.KurtosisZ:   Kurtosis z

175 # THtml specific settings (for more see doc of THtml class).
    Root.Html.SourceDir:    .
177 Root.Html.Root:          http://root.cern.ch/root/html
    Root.Html.ViewCVS:      http://root.cern.ch/viewcvs/trunk/%f?view=log
179 Root.Html.Search:        http://www.google.com/search?q=%s+site%3A%u+site%3
    A%u%2Fsrc%2F+site%3A%u%2Fexamples%2F
    #Root.Html.OutputDir:    htmdoc/
181 #Root.Html.Homepage:
    #Root.Html.Header:
183 #Root.Html.Footer:
    #Root.Html.Description: // -----
185 #Root.Html.Author:       // Author:
    #Root.Html.LastUpdate: // @(#)
187 #Root.Html.Copyright:    * Copyright

189 # GUI specific settings.

```

```

Gui.Backend:                native
191 Gui.Factory:              native
# GUI style: Modern (flat popup menus) or Classic (win 95 style)
193 Gui.Style:               Modern
Gui.DefaultFont:            --helvetica-medium-r-*-12-*-*-*-*-*iso8859
-1
195 Gui.MenuFont:            --helvetica-medium-r-*-12-*-*-*-*-*iso8859
-1
Gui.MenuHiFont:             --helvetica-bold-r-*-12-*-*-*-*-*iso8859-1
197 Gui.DocFixedFont:        --courier-medium-r-*-12-*-*-*-*-*iso8859-1
Gui.DocPropFont:           --helvetica-medium-r-*-12-*-*-*-*-*iso8859
-1
199 Gui.IconFont:            --helvetica-medium-r-*-10-*-*-*-*-*iso8859
-1
Gui.StatusFont:            --helvetica-medium-r-*-10-*-*-*-*-*iso8859
-1
201 Gui.BackgroundColor:     #e0e0e0
Gui.ForegroundColor:        black
203 Gui.HighLightColor:      #d0d0d0
Gui.SelectBackgroundColor:  #86abd9
205 Gui.SelectForegroundColor: white
Gui.DocumentBackgroundColor: white
207 Gui.DocumentForegroundColor: black
Gui.TooltipBackgroundColor: LightYellow
209 Gui.TooltipForegroundColor: black
Gui.IconPath:               $(HOME)/icons:$(ROOTSYS)/icons:.
211 Gui.MimeTypeFile:        $(HOME)/.root.mimes
# If above does not exists defaults to this:
213 #Gui.MimeTypeFile:        $(ROOTSYS)/etc/root.mimes
# Type of Browser: TRootBrowser or TRootBrowserLite
215 Browser.Name:            TRootBrowser
# Browser Options (plugins)
217 # F: File browser E: Text Editor H: HTML browser
# C: Canvas I: I/O redirection P: Proof G: GL viewer
219 Browser.Options:         FCEI
# Can be either small, big, list, details
221 Browser.IconStyle:       small
# Can be either name, type, size, date
223 Browser.SortBy:          name
Browser.GroupView:          10000
225 Browser.ShowHidden:      no
Browser.AutoThumbnail:      yes
227 # Start URL for the TRootBrowser embedded HTML renderer
Browser.StartUrl:           http://root.cern.ch/root/html/ClassIndex.html
229
# Rint (interactive ROOT executable) specific alias, logon and logoff

```

```

    macros.
231 Rint.Load:                rootalias.C
    Rint.Logon:              rootlogon.C
233 Rint.Logoff:             rootlogoff.C
### Prompt colors
235 # Whether to use default colors for light-on-dark (i.e. reverse) color
    scheme:
    #Rint.ReverseColor:      no
237 # Prompt colors: #rgb or #rrggbb or color names:
    # "black", "red", "green", "yellow", "blue", "magenta", "cyan", "white"
239 # "default" will keep the current terminal color.
    # can be combined with string containing "under" and "bold".
241 #Rint.TypeColor:          bold blue
    #Rint.BracketColor:      bold green
243 #Rint.BadBracketColor:    bold red underlined
    #Rint.PromptColor:       default
245 #Rint.TabComColor:        magenta
    #
247 ### History
    # Record session commands, set to "-" to turn off command recording.
249 Rint.History:              $(HOME)/.root_hist
    # History file size, once HistSize is reached remove all but HistSave
    entries,
251 # set to 0 to turn off command recording.
    # Can be overridden by environment variable ROOT.HIST=size[:save],
253 # the ":save" part is optional.
    # Rint.HistSize:          500
255 # Set to -1 for sensible default (80% of HistSize), set to 0 to disable
    history.
    # Rint.HistSave:          400
257 # Print a single line welcome message instead of the default verbose
    version
    # Rint.WelcomeLite:        no
259 # When the interactive ROOT starts, it can automatically load some
    frequently
    # used includes. However, this introduces several overheads
261 # - A long list of CINT and system files will be kept open during the
    session
    # - The initialisation takes more time (noticeable when using gdb or
    valgrind)
263 # - Memory overhead of about 5 Mbytes (1/3 of the base ROOT executable)
    when
    # including <vector>
265 # You can set the variable below to 0 to disable the loading of these
    # includes at startup. You can set the variable to 1 (default) to load
267 # only <iostream> , <string> and <RTypesCint.h>. You can set it to 2 to

```

```

# load in addition <vector> and <pair>. We strongly recommend setting
269 # the variable to 2 if your scripts include <vector> and you execute
# your scripts multiple times.
271 Rint.Includes:      1

273 # ACLiC customization.
# ACLiC.Linkdef specifies the suffix that will be added to the script name
# to
275 # try to locate a custom linkdef file when generating the dictionary.
ACLiC.Linkdef:      _linkdef
277 # Set a top directory for storing the libraries produced by ACLiC.
#ACLiC.BuildDir:      /where/I/would/like/my/compiled/scripts
279 # Add additional include directives for ACLiC compilations.
#ACLiC.IncludePaths:  -I/where/the/includes/are
281 # Select whether and how ACLiC tracks the dependency of the libraries.
# 0 - No tracking
283 # 1 - [Default] New libraries are explicitly linked to all currently
# loaded libraries
# 2 - Generate and use rootmap file for each library
285 # 3 - Generate rootmap file for each library _and_ explicitly link to the
# needed libraries.
287 # On Windows, the default is 3
#ACLiC.LinkLibs:      1
289
# PROOF related variables
291 #
# PROOF debug options.
293 # Proof.DebugLevel:      0
# Proof.DebugMask:      -1
295 #
# PROOF GDB hooks
297 # allows a debugger to be attached early in the startup phase of proofserv
# 0 - don't wait
299 # 1 - master proofserv enters wait loop
# 2 - slave proofserv enters wait loop
301 # 3 - any proofserv enters wait loop
#
303 # Proof.GdbHook: 0
#
305 # To control the number of processes in PROOF-Lite (0 disables PROOF-Lite).
# This setting cannot be overwritten in the user rootrc files.
307 # ProofLite.MaxWorkers: -1
#
309 # On the master enable parallel startup of workers using threads
# Proof.ParallelStartup: no
311 #

```

```

# Proof.StatsHist:      no
313 # Proof.StatsTrace:    no
# Proof.SlaveStatsTrace: no
315 #
# Proof.CondorHome:      /opt/condor
317 # Proof.CondorConfig:  /opt/condor/etc/condor_config
#
319 # PEAC.GmUrl:           http://somewhere:8080/clarens/
# PEAC.LmUrl:           http://elsewhere:8080/clarens/
321
# Certificate and key
323 # Clarens.CertFile:     $(HOME)/.globus/usercert.pem
# Clarens.KeyFile:      $(HOME)/.globus/userkey.pem
325
# Variables related to authentication to rootd and proofd.
327 #
# Default authentication method for rootd and proofd.
329 # These are supported for backward compatibility but have a very
# low priority. System defaults are generated by configure as a list
331 # in system.rootauthrc in $ROOTSYS/etc/ or /etc/root; the file
# $HOME/.rootauthrc can be used to override the system defaults.
333 # (0=UsrPwd, 1=SRP, 2=Krb5, 3=Globus,4=SSH, 5=UidGid)
Rootd.Authentication:  0
335 Proofd.Authentication:  0

337 # Connection is shutdown at timeout expiration. Timeout is in seconds.
# Negotiation cannot be attempted at low level (i.e. inside
339 # TAuthenticate::Authenticate()) because of synchronization
# problems with the server.
341 # At higher level, TAuthenticate::HasTimedOut() gives information
# about timeout: 0 = no timeout; 1 = timeout, no methods left;
343 # 2 = timeout, still methods to be tried .
# Caller should decide about an additional attempt.
345 # Timeout disabled (< 0) by default. Can be changed on-the-fly
# with the static method TAuthenticate::SetTimeOut(to_value)
347 #
# Auth.Timeout:         -1
349
# Password dialog box.
351 # Set to 0 if you do not want a dialog box to be popped-up
# when a password is requested.
353 # Default is 1.
#
355 # Auth.UsePasswdDialogBox: 0

357 # Set this to 1 if you want full SRP authentication in PROOF

```



```

# (Client-to-Master and Master-to-Slave).
359 Proofd.SendSRPPwd:      0

361 # Set this to 1 to use SSH authentication in PROOF servers
# (Master-to-Slave or Slaves-to-DataServers). This is switched
363 # off by default because credentials forwarding for SSH is not
# controlled by the system; however the user may have other
365 # ways to guarantee it, so it may want to switch it on.
ProofServ.UseSSH:      0
367

# Default login name (if not defined is taken from $(HOME)).
369 #UsrPwd.Login:           qwerty
#SRP.Login:              qwerty
371 #Krb5.Login:            qwerty@LOCAL.DOM.AIN
#Globus.Login:          cd:~/ .globus cf:usercert.pem  kf:userkey.pem  ad:/
                        etc/grid-security/certificates
373 #SSH.Login:             qwerty
#UidGid.Login:          qwerty
375

# To be prompted for login information.
377 #UsrPwd.LoginPrompt:    yes
#SRP.LoginPrompt:       yes
379 #Krb5.LoginPrompt:     yes
#Globus.LoginPrompt:    yes
381 #SSH.LoginPrompt:      yes
#UidGid.LoginPrompt:    yes
383

# To reuse established security context.
385 UsrPwd.ReUse:          yes
SRP.ReUse:              no
387 Krb5.ReUse:           no
Globus.ReUse:           yes
389 SSH.ReUse:            yes

391 # Duration validity of the sec context for UsrPwd, SRP and SSH.
# Format: <hours>:<minutes> (default 24:00)
393 #UsrPwd.Valid:         24:00
#SRP.Valid:             24:00
395 #SSH.Valid:           24:00

397 # To control password encryption for UsrPwd authentication.
UsrPwd.Crypt:           yes
399

# Globus miscellanea.
401 # Globus Proxy duration: HH:MM (ex 12:15 for 12 hours and 15 min)
# 'default' for system default.

```

```

403 Globus.ProxyDuration:    default
#Globus.ProxyDuration:    14
405 # Number of bits for the initial key.
Globus.ProxyKeyBits:      1024
407
# Path to alternative 'ssh' (to override $PATH if ever needed).
409 #SSH.ExecDir:            /usr/bin

411 # In case of error, SSH returns 1 (or 256 = 0x100).
# To trap those errors for which one should retry, error printouts
413 # must be parsed; any substring found under the Env SSH.ErrorRetry
# triggers a retry condition; strings can be added here
415 # in the form (including double quotes):
#
#           +SSH.ErrorRetry:    "<error_string>"
417 # This is what one usually gets if the server has reached the maximum
# number of sshd daemons (defined by MaxStartups in sshd_config);
419 # this is a typical case in which one should retry.
SSH.ErrorRetry:    "Connection closed by remote host"
421 # Max number of retries for SSH in case of retry error (see above).
SSH.MaxRetry:      100
423
# Type of key to be used for RSA encryption:
425 # 0 = local; 1 = SSL (default if openssl available).
RSA.KeyType:    1
427
# In case of 'RSA.KeyType: 1' this specifies the number of bits to
429 # be used for the Blowfish key used to encrypt the exchanged information
# Default 256, minimum 128, maximum 15912.
431 #SSL.BFBits:    256

433 # Server authentication in TServerSocket.
#
435 # General: file with server access rules
#SrvAuth.DaemonRc: /etc/root/system.daemonrc
437 #
# UsrPwd: check of host equivalence via /etc/hosts.equiv or $HOME/.rhosts.
439 #SrvAuth.CheckHostsEquivalence: 1
#
441 # SRP: pass file (default $HOME/.srootdpass).
#SrvAuth.SRPpassfile: $HOME/.srootdpass
443 #
# Globus/GSI: hostcert configuration file.
445 #SrvAuth.HostCert: /etc/root/hostcert.conf
# Globus/GSI: gridmap file.
447 #SrvAuth.GridMap: /etc/grid-security/grid-mapfile
#

```

```

449 # SSH: port for the sshd daemon.
      #SrvAuth.SshdPort: 22
451
      # Force file opening via TNetFile (TXNetFile) if a hostname is specified
453 # in the Url.
      # By default, for local files TFile::Open() invokes directly TFile
455 #TFile.ForceRemote: yes

457 # Control the action to be taken when opening an existing ROOT file which
      # looks corrupted; by default an attempt to recover the file is made; if
459 # this variable is set to no the file is just flagged as zombie.
      #TFile.Recover: no

461
      # Control the usage of asynchronous reading capabilities eventually
463 # supported by the underlying TFile implementation. Default is yes.
      #TFile.AsyncReading: no

465
      # Control the usage of asynchronous prefetching capabilities irrespective
467 # of the TFile implementation. By default it is disabled.
      #TFile.AsyncPrefetching: no

469
      # Special cases for the TUrl parser, where the special cases are parsed
471 # in a protocol + file part, like rfio:host:/path/file.root,
      # castor:/path/file.root or /alien/path/file.root.
473 # In case the file namespace descriptor ends with - the namespace
      # is not a part of the filename.
475 # Extend in private .rootrc with a +Url.Special line.
      Url.Special: file: rfio: hpss: castor: gfal: dcache:
477 +Url.Special: /alien/- /castor/

479 # PROOF XRD client variables
      # Debug level (<0 : errors, 0 : minimal, 1 : low, 2 : medium, 3 : high)
      [-1]
481 # XProof.Debug: 0
      # Socket read timeout [in secs: default 10 secs]
483 # XProof.ReadTimeout: 10

485 # The following env vars are handled by TXNetFile and related classes
      # (module netx, libNetx.so).
487 #
      # XNet.ConnectTimeout - maximum time to wait before server's
489 # response on a connect [120 s]
      # XNet.RequestTimeout - maximum time to wait before considering
491 # a read/write failure [300 s]
      # XNet.ConnectDomainAllowRE
493 # - sequence of TRegexp regular expressions

```

```

#                               separated by a |.
495 #                               A domain is granted access to for the
#                               first connection if it matches one of these
497 #                               regexps. Example:
#                               slac.stanford.edu|pd.infn.it|fe.infn.it
499 # XNet.ConnectDomainDenyRE
#                               - sequence of TRegexp regular expressions
501 #                               separated by a |.
#                               A domain is denied access to for the
503 #                               first connection if it matches one of these
#                               regexps. Example:
505 #                               slac.stanford.edu|pd.infn.it|fe.infn.it
# XNet.RedirDomainAllowRE
507 #                               - sequence of TRegexp regular expressions
#                               separated by a |.
509 #                               A domain is granted access to for a
#                               redirection if it matches one of these
511 #                               regexps. Example:
#                               slac.stanford.edu|pd.infn.it|fe.infn.it
513 # XNet.RedirDomainDenyRE
#                               - sequence of TRegexp regular expressions
515 #                               separated by a |.
#                               A domain is denied access to for a
517 #                               redirection if it matches one of these
#                               regexps. Example:
519 #                               slac.stanford.edu|pd.infn.it|fe.infn.it
# XNet.MaxRedirectCount - maximum number of redirections from
521 #                               server [16]
# XNet.Debug - log verbosity level
523 #                               (0=nothing ,
#                               1=messages of interest to the user ,
525 #                               2=messages of interest to the developers
#                               (includes also user messages),
527 #                               3=dump of all sent/received data buffers
#                               (includes also user and developers
529 #                               messages). [0]
#                               4=dump also the data received/sent at the lower
#                               level
531 # XNet.ReconnectWait - sleep-time before going back to the
#                               load balancer (or rebouncing to the same
533 #                               failing host) after a read/write error
#                               [5 s]
535 # XNet.ReadAheadSize - read ahead value , in bytes , to use with the
#                               XrdClient
#                               internal caching scheme
537 #                               Generally not much useful together with the TFile

```

```

        cache ,
#           which will disable the readahead as soon it
starts operating ,
539 #           i.e. requesting async data through XrdClient.
#           Note that we cannot suppose that the TFile cache
541 #           will start working at the beginning of the data
processing .
#           Setting a read ahead size > 0 will involve this
phase ,
543 #           sometimes it could be an advantage , sometimes not
.
#           If > 0, TXNetFile will switch it on again if it
sees that
545 #           synchronous ReadBuffer requests are issued again

XNet.ReadAheadSize: 0

549 # XNet.ReadCacheSize      - Default read cache size inside XrdClient, in
    bytes .
#           TXNetFile/TFileReadCache automatically sets this
    to a
551 #           'correct' value when needed.
#           Set a value here to override a *minimum*
553 #           size. It is not preallocated.
#           TFile/TXNetFile can anyway make it grow
555 #           automatically when needed.

XNet.ReadCacheSize: 10000000

559 # XNet.FirstConnectMaxCnt
#           - Number of connect retries to the first host
561
# XNet.PrintTAG             - Print a particular string the developers
563 #           can choose to quickly recognize the
#           version at run time [0]
565
# XNet.ParStreamsPerPhyConn
567 #           - Number of additional TCP streams to use per
#           each physical connection
569 #           0=monostream mode
#           1-15=multistream mode
571
XNet.ParStreamsPerPhyConn: 0

573 # XNet.DfltTcpWindowSize
575 #           - Default size of the TCP window; set to 0 to use

```

```

    the
#                               system default , which means automatic scaling
    under Linux
577 #                               and MacOSX (at least). Value in bytes [0]
# XNet.DfltTcpWindowSize 262144
579
# XNet.TransactionTimeout
581 #                               – The maximum amount of time either a request or a
    server
#                               wait request can be before the client declares an
    error .
583 # XNet.TransactionTimeout: 28800

585 # Example of custom setting for the Rint application (root.exe).
# This overrides the default specified above for a generic application.
587 # Color 5 is yellow .
Rint.Canvas.HighlightColor:      5

```

8.4 A working bash_env

Here is a working bash_env post-wheezy

```

# For e09067:
2 # .bash_env will the location of all mona
# environment variables that are sourced.
4 #
# Should be sourced be .bashrc
6 #
#
8 # MoNA DAQ configuration environment variables.

10
#ROOT environment settings
12 PATH=$PATH:${ROOTSYS}/bin:${ROOTSYS}/include:${ROOTSYS}/lib:${ROOTSYS}/lib/
    root"
LD_LIBRARY_PATH=":${ROOTSYS}/lib:${ROOTSYS}/lib/root"
14
#
16 export DAQHOST=spdaq40.nsl.msu.edu # Where readout runs.
export DAQVERSION=8.2
18 export SPECTCLVERSION=3.2
export SCALERINTERVAL=2
20
#variables concerning location of evtfiles

```

```

22 export STAGEAREA=${HOME}/stagearea
export EVENTHOST=${STAGEAREA} # Where the eventlogging is done.
24 export EVTFILES=${STAGEAREA}/complete
export EXPNUMBER=${HOME}/experiment
26
#Cosmic environment variables (for Hope)
28 #export COSMIC_SPECTCL=${HOME}/shared/spectcl-cosmics
#export COSMIC_READOUT=${HOME}/shared/readout-cosmics
30
32 #export SHARED_SCALER=${HOME}/shared/scaler ###??used by what?###
34
#Tandem enviroment variables
export TCONFIG=${HOME}/config
36 export TSPECTCL=${HOME}/spectcl_tandem
export TANDEM_READOUT=${HOME}/readout_tandem
38
#MoNA only environment variables
40 #export MONA_DAQ=${HOME}/mona/daq
export MONA_READOUT=${HOME}/readout_mona
42 #export MONA_SCALER=${HOME}/shared/scaler
export MONA_SPECTCL=${HOME}/spectcl_mona
44 export MONA_CONFIGDIR=${HOME}/config # Where the configuration files are.
#export CONFIGARCHV=${HOME}/spectcl_tcl/archive_config ##need???###
46 export MONA_SETUP=${HOME}/config/MoNA_setup_run.tcl ## MoNA_setup_file
48
#FPGA environment variables
export FPGA_DIR=${HOME}/fpga
50 export FPGA_BITDIR=${HOME}/fpga/bitfiles #where xlm_bitfiles are
export XLM_LV1_BITFILE=${FPGA_BITDIR}/LV1.G4.BIT
52 export XLM_LV2_BITFILE=${FPGA_BITDIR}/vtx_b15.bit
export XLM_VALUES_FILE=${HOME}/fpga/xlm_values.dat #where xlm_values.dat is
54 export XLM_VAL_FILE_COSMIC=${FPGA_DIR}/xlm_values_cos.dat
56
#tools
export TOOLS=${HOME}/noncvstools
58 export CFDtools=${TOOLS}/CFD
export XLMtools=${HOME}/tools/xlm
60
export SPECTCL_HOME=/usr/opt/spectcl/2.2
62
export TCLLIBPATH=/usr/opt/daq/$DAQVERSION/TclLibs
64
#Additions to the path:
66 #export ROOTSYS=/opt/etch/root/root5.14

```

```

68 PATH=$PATH":./:${HOME}/bin/:${READOUT}:${SPECTCL}:${SCALERS}:/usr/opt/
    spectrodaq/bin/:$ROOTSYS/bin:$ROOTSYS/include:/user/scheit/isoft/root_v5
    .12_00/bin:/user/scheit/bin:/autooptd/optd1/bin:/usr/opt/daq/Bin/"
    export PYTHONPATH=$ROOTSYS/lib/root:$PYTHONPATH
70
72 export PATH PS1 TCLLIBPATH
74 export DAQROOT=/usr/opt/daq          # Where the NSCL DAQ stuff lives.
    #export LD_LIBRARY_PATH=$ROOTSYS/lib/root:/opt/etch/CLHEP/2.0.3.1/lib:
    $LD_LIBRARY_PATH
76
78 export BUFFERSIZE=8192
80 export CVS_RSH=ssh
    export CVSROOT=:ext:monasoft@fishtank:/user/monasoft/cvsroot
    export THEHOME=/user/e09067

```