

# **DELAYED GAMMA SIMULATOR v2.0**

Written By Robert Crabbs

---

# 1. INTRODUCTION

---

Fission-delayed gamma spectra provide unique radiation signatures for fissionable nuclides. This makes the method of interest in Active Non-Destructive Interrogation (ANDA). In a delayed gamma experiment, a fissile target is irradiated with a tailored neutron spectrum, which subsequently undergoes fission. The fission fragments themselves tend to be highly radioactive, releasing 7-8 delayed gamma rays on average. The uniqueness of each nuclide's fission fragment distribution makes for measurable differences in the emitted delayed gamma spectra.

For this research project, we focused on the delayed gamma signal during the first few seconds after irradiation (and fission) takes place. Beyond these short times, the gamma emissions only grow weaker, as the fission fragments that produce them decay away. Therefore, our setup uses a pulsed neutron irradiation at the target to continually refresh the fission fragment populations. Each pulse cycle includes a short (~100ms) irradiation step, during which neutrons impinge on the target. A shorter (~20ms) cool-off period follows, allowing the interrogating radiation and prompt signal to die away. Then we begin counting the gamma signal for a relatively long (~10s) time, before starting the next cycle.

Simulations of such an experiment proved to be fairly difficult. No one tool exists that is able to transport the irradiating neutrons, compute fission rates, follow the time evolution of the fission fragments, and then start another irradiation step afterwards. ORIGEN-S does have the capability to track the decays of the fission fragments, but its irradiation computations only apply to common reactor designs. MCNPX can compute reactor probabilities, but has no awareness of time or reaction products. An RSICC tool called MOCUP attempts to couple both ORIGEN-S and MCNPX, but we found it to be so jumbled in its operation as to be unusable.

Hence the need for writing our own code for the application. This tool is composed of a series of Python scripts, aimed to compute the approximate delayed gamma emissions of a pulsed neutron interrogating experiment. It depends on ORIGEN-S to compute the populations of fission products over time. The scripts process the ORIGEN-S output to create delayed gamma spectra during each counting period. Leftover decay products, combined with new fragments produced during fission, are fed back into ORIGEN for each new irradiation cycle.

---

## 2. REQUIREMENTS

---

NOTE: You will need Python, SCALE6, and a Windows PC in order to run these scripts.

There are 14 required files to start the tool chain. The rest are created by the scripts. The input should all be in the same directory.

- Delayed Gamma Simulator.py
- 1-createORIGENInput.py
- 2-readORIGENcomp.py
- 3-readORIGENspec.py
- 4-formatComposition.py
- Delayed Gammas - Skeleton.inp
- ORIGEN - Light Elements.txt
- settings.csv
- Pu239 Fast Fragments.csv
- Pu239 Thermal Fragments.csv
- U235 Fast Fragments.csv
- U235 Thermal Fragments.csv
- U238 Fast Fragments.csv
- ORIGEN - Fission Products.txt

“Delayed Gamma Simulator.py” is a master control script which launches the scripts sequentially. Complete file descriptions for the input, output, and scripts is provided in Chapter 4: File Descriptions.

Note that you will also need to change the SCALE script located at C:\SCALE6\cmds\runscale.bat. By default, the last two lines prompt the user for to press any key once SCALE finishes. This will interrupt execution of the Delayed Gamma Simulator.

---

## 3. EXECUTION

---

### Delayed Gamma Simulator.py

(Loops the following for each time step)

#### 1-createORIGENInput.py

- 2-readORIGENcomp.py

- 3-readORIGENspec.py

- 4-formatComposition.py

1.) First, modify the settings.csv file according to the details of your problem. There are 13 settings specific for each run that need to be set. They are detailed in the “File Descriptions” chapter.

2.) Next, you need to manually modify the ORIGEN-S input template, called “Delayed Gammas - Skeleton.inp”. ORIGEN-ARP will greatly help in the construction of this file, though there are several parameters specific to the Delayed Gamma Simulator to account for. These are detailed later.

3.) Run the “1-createORIGENInput.py” script to generate a complete ORIGEN-S input deck for the current pulse cycle. This script requires a composition data file called “initialComposition.csv”, which specifies the amounts of fission fragments present (in grams). This file is created automatically by “Delayed Gamma Simulator.py”. For the first pulse, the initial composition is simply the sum of all fission fragments produced during the irradiation step. For all later pulses, the composition contains leftover (non-decayed) fission products as well as new fission fragments.

4.) The output of 1-createORIGENInput.py is called “ORIGEN\_input.inp”, and can be fed into SCALE6 to calculate delayed gamma spectra and time-dependent compositions. It will create a file called “ORIGEN\_input.inp”. The plotOPUS outputs are all deleted automatically by “Delayed Gamma Simulator.py”.

5.) The “2-readORIGENcomp.py” script then reads time-dependent composition data from the ORIGEN-S output, and outputs it to a large comma-separated text value called masterArray.csv. Every time step specified in the ORIGEN-S output file corresponds to two columns in masterArray.csv. The first column lists all nuclides present at the given time, while the second column lists their respective amounts in grams. Note that masterArray.csv is compatible with the decayCalc tool to produce transported gamma responses.

“2-readORIGENcomp.py” also produces a “finalComposition.csv” file that corresponds to the last two columns in masterArray.csv.

5.) Next, run "3-readORIGENspec.py" to extract the delayed gamma information from the rather long ORIGEN-S output file. This script reads plotOPUS data to produce a more succinct comma-separated table of the emissions. The output is called "Delayed Gamma Spectrum.csv".

6.) The final script is "4-formatComposition.py", which reads the "finalComposition.csv" file produced by "2-readORIGENcomp.py". ORIGEN outputs composition data in a different format than it takes as input. (The input uses ZAIDs, such as 922350, while the output gives nuclide IDs such as U 235.) This script converts the nuclide IDs into ZAIDs for preparation, creating a "leftoverFragments.csv" file that is combined with new fragments at the beginning of the next pulse cycle.

7.) Finally, "Delayed Gamma Simulator.py" creates a separate directory to store the test results, and moves all of the run's input settings and output files into it.

I highly recommend using decayCalc.py to automate the scripts.

---

## 4. FILE DESCRIPTIONS

---

### SCRIPTS

- **Delayed Gamma Simulator.py**: Master control program, responsible for running each script in turn and organizing output.
- **1-createORIGENInput.py**: Substitutes composition and energy binning input data into the “Delayed Gammas - Skeleton.inp” ORIGEN-S input deck.
- **2-readORIGENcomp.py**: Reads the time-dependent composition data from the larger ORIGEN output, and summarizes it to “masterArray.csv”.
- **3-readORIGENspec.py**: Reads gamma emission data due to the fission fragment decays, as computed by ORIGEN.
- **4-formatComposition.py**: Creates a composition file for leftover fragments present at the beginning of the next pulse cycle. Nuclide IDs are converted to ZAID form.

### INPUT

- **Delayed Gammas - Skeleton.inp**: A generic ORIGEN-S input file without specific composition or energy bin definitions.
- **settings.csv**: This file is meant to make this tool chain easy to port and reuse. Ideally, the scripts themselves need not be modified between runs--all the necessary settings that might change are located instead in this \*.csv file. Note that the file is case-insensitive, and also that the tool has no error catching ability if the settings are not input correctly. The settings remain constant for every pulse cycle. The settings are:
  - "**scriptDir**": The Windows directory in which the toolchain is located. Must conform to the Windows standard and include Python escape characters as necessary. For example, use `C:\\scale6\\"Delayed Gamma Simulator\\"` if this tool is located in `C:\scale6\Delayed Gamma Simulator\`.
  - "**scaleDir**": Similar to the above, but specifies the location of the root installation directory for SCALE6. Usually this setting is `C:\\scale6\\`.
  - "**projectName**": The main script creates a folder of this name in the main script directory to store input and output. Be sure to change the runName setting before each run, as files may be overwritten otherwise.
  - "**pulseCycles**": Number of irradiation/counting cycles to simulate. Must be a positive integer.

- **"timeSteps"**: Number of time bins specified in the ORIGEN-S input deck during the counting period for a cycle. Must be a positive integer.

- **"countingTime"**: Length of each counting period in seconds; naturally it should be positive.

- **"binStruct"**: Defines the structure of energy bins used for output gamma spectra. One option has the form **"linear:maxErg"**. This creates an evenly-spaced energy bin structure in decreasing order of energy. E\_max is the highest energy bin boundary in keV and the bins; the lowest bin is from 0 keV to some fraction of the binWidth. Naturally, bins cannot be negative in energy.

If **"binStruct"** is set to **"manual"**, one must create a file called **"binStruct.txt"** in the working script directory. The lines in this file correspond to the upper energies (in keV) for the tally bins, and must be given in decreasing order.

- **"binWidth"**: Width of the energy bins in keV. Is used to convert plotOPUS gamma spectra (in gammas/sec/MeV) to more useful units. All energy bins must have the same width.

- **"U235Thermal"**: Number of U235 thermal fissions during each irradiation step.

- **"U235Fast"**: Number of U235 fast fissions during each irradiation step.

- **"Pu239Thermal"**: Number of Pu239 thermal fissions during each irradiation step.

- **"Pu239Fast"**: Number of Pu239 fast fissions during each irradiation step.

- **"U238Fast"**: Number of U238 fast fissions during each irradiation step.

- **Pu239 Fast Fragments.csv**: The fission fragment distribution of Pu239 when induced by fast neutrons. Measured by England and Rider, and given in grams per fission.

- **Pu239 Thermal Fragments.csv**: The fission fragment distribution of Pu239 when induced by thermal neutrons. Measured by England and Rider, and given in grams per fission.

- **U235 Fast Fragments.csv**: The fission fragment distribution of U235 when induced by fast neutrons. Measured by England and Rider, and given in grams per fission.

- **U235 Thermal Fragments.csv**: The fission fragment distribution of U235 when induced by thermal neutrons. Measured by England and Rider, and given in grams per fission.

- **U238 Fast Fragments.csv**: The fission fragment distribution of U238 when induced by fast neutrons. Measured by England and Rider, and given in grams per fission.

- **ORIGEN - Light Elements.txt**: ORIGEN sorts nuclides into three categories: actinides, fission fragments, and light elements. Each has a different data set. This text file lists all nuclides which fall into the "Light Element" category.

- **ORIGEN - Fission Products.txt**: Similar to "ORIGEN - Light Elements", it lists the nuclides that are considered fission products by ORIGEN.

- **binStruct.txt**: If the “binStruct” option in “settings.csv” is set to “manual”, then this file must be present to define energy bins. Each line in this file represents an upper energy limit (in keV) for a bin. Only one energy is allowed per line, and the energies must be listed in increasing order. Be sure to include a catch-all bin below the first bin you really care about—the lowest bin always has zero counts, because its upper bound represents the lowest energy of interest.

## OUTPUT

- **initialComposition.csv**: A file containing the target’s composition before irradiation begins during a given pulse cycle.

- **ORIGEN\_input.inp**: A complete ORIGEN-S input deck, created by 1-createORIGENInput.py.

- **ORIGEN\_input.out**: The output file generated by ORIGEN-S for the above input deck.

- **masterArray.csv**: Contains the time-dependent composition data computed by ORIGEN-S, in a format that is compatible with decayCalc.py.

- **Delayed Gamma Spectrum.csv**: Lists the delayed gamma emissions due to decay within the sample. Each entry in this output table is given in counts registered for a given energy/time window.

- **leftoverFragments.csv**: Lists the set of nuclides that are present at the end of the counting period in a given cycle. These nuclides are then used as part of the initial composition for the next pulse cycle.

---

## 6. SAMPLE INPUT

---

Below is a working example of a decayCalc problem. The main goal is to calculate the background radiation due to a depleted uranium witness foil in an experimental setup. I have constructed the two basic input files required: settings.csv and “Delayed Gammas - Skeleton.inp”.

### settings.csv

For a fast example, we’ll use a very minimally-demanding set of options. On the workstation this simulation is run, SCALE is located in C:\scale6\, while the scripts are being run from C:\scale6\Delayed Gamma Simulator\.

The simulation calls for 15 pulse cycles. Each counting period lasts 10s, and is broken into 20 different time bins (500ms apiece). Hence, we will call the project “10s Counting”.

For the purposes of calculating the gamma emissions, each energy bin is 3keV wide, and all emissions from 0keV to 4MeV are counted.

Finally, each pulse produces 400 U235 thermal fissions, 500 U235 fast fissions, 100 Pu239 thermal fissions, 300 Pu239 fast fissions, and 200 U238 fast fissions. This isn’t a physical test so much as a simple example, of course.

```
scriptDir,C:\scale6\Delayed Gamma Simulator\,\,Location of scripts
scaleDir,C:\scale6\,\,Location of the SCALE6 installation root
projectName,"10s Counting",Folder name in which to save results
pulseCycles,15,Number of pulses to simulate
timeSteps,200,Number of time bins per counting period
countingTime,10,Length of counting period in seconds
binWidth,3,Width of energy bins in keV
binStruct,linear:4000,What energy bin structure to use
U235Thermal,400,Number of U-235 thermal fissions per pulse
U235Fast,500,Number of U-235 fast fissions per pulse
Pu239Thermal,100,Number of Pu-239 thermal fissions per pulse
Pu239Fast,300,Number of Pu-239 fast fissions per pulse
U238Fast,200,Number of U-238 fast fissions per pulse
```

The text above translates to the following table:

<b>scripDir</b>	C:\\scale6\\Delayed Gamma Simulator\\"\\	Location of scripts
<b>scaleDir</b>	C:\\scale6\\	Location of the SCALE6 installation root
<b>projectName</b>	"10s Counting"	Folder name in which to save results
<b>pulseCycles</b>	15	Number of pulses to simulate
<b>timeSteps</b>	20	Number of time bins per counting period
<b>countingTime</b>	10	Length of counting period in seconds
<b>binWidth</b>	3	Width of energy bins in keV
<b>binStruct</b>	linear:4000	What energy bin structure to use
<b>U235Thermal</b>	400	Number of U-235 thermal fissions per pulse
<b>U235Fast</b>	500	Number of U-235 fast fissions per pulse
<b>Pu239Thermal</b>	100	Number of Pu-239 thermal fissions per pulse
<b>Pu239Fast</b>	300	Number of Pu-239 fast fissions per pulse
<b>U238Fast</b>	200	Number of U-238 fast fissions per pulse

## Delayed Gammas - Skeleton.inp

The input deck below illustrates the peculiarities of a Delayed Gamma Simulator input deck. A skeleton file has several parameters left undefined, which the 1-createORIGENInput.py script takes care of. Such parameters are marked in bold underline in this sample input deck. Note that the keywords used must be present for 1-createORIGENInput.py to properly substitute their values. They are case sensitive, as well.

E\_BINS-1: The number of energy bins used, minus 1. ORIGEN-S requires this number for some reason.

NUCLIDES: The number of nuclides defining the input composition.

ENERGY\_BINS: A string defining the upper energy bin boundaries. Bins are listed in decreasing order in eV.

NUCLIDE\_ZAIDS: A string of nuclide ZAIDs in the input composition.

NUCLIDE\_MASSES: Masses corresponding to the entries in NUCLIDE\_ZAIDS.

NUCLIDE\_LIBRARIES: Library IDs corresponding to the entries in NUCLIDE\_ZAIDS. Either a "1" for Light Elements or "3" for Fission Products.

Note that 5 energy bins are given per line and 7 nuclides per line, so as to avoid the 80-character line limit imposed in ORIGEN-S input decks.

**'This SCALE input file was generated by**

'OrigenArp Version 5.1.01 March 22, 2007

#origens

-1\$\$ 20000000

0\$\$ a11 71 e t

Decay Case

3\$\$ 21 1 1 0 a16 2 a33 **E\_BINS-1** e t

35\$\$ 0 t

54\$\$ a8 1 a11 0 e

56\$\$ a2 10 a6 1 a10 0 a13 **NUCLIDES** a14 1 a15 3 a17 2 e

57\*\* 0 a3 1e-05 e

95\$\$ 0 t

Decay (0.0s - 5.0s)

0 MTU

60\*\* 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0

61\*\* f1E-24

65\$\$

'Gram-Atoms Grams Curies Watts-All Watts-Gamma

3z 1 0 0 3z 3z 3z 6z

3z 1 0 0 3z 3z 3z 6z

3z 1 0 0 3z 3z 3z 6z

81\$\$ 2 0 26 1 e

82\$\$ 2 2 2 2 2 2 2 2 2 e

83\*\* **ENERGY\_BINS** e

73\$\$ **NUCLIDE\_ZAIDS**

74\*\* **NUCLIDE\_MASSES**

75\$\$ **NUCLIDE\_LIBRARIES**

t

56\$\$ 0 0 a10 1 e t

56\$\$ 0 0 a10 2 e t

56\$\$ 0 0 a10 3 e t

56\$\$ 0 0 a10 4 e t

56\$\$ 0 0 a10 5 e t

56\$\$ 0 0 a10 6 e t

56\$\$ 0 0 a10 7 e t

56\$\$ 0 0 a10 8 e t

56\$\$ 0 0 a10 9 e t

56\$\$ 0 0 a10 10 e t

54\$\$ a8 1 a11 0 e

56\$\$ a2 10 a6 1 a10 10 a14 1 a15 3 a17 2 e

57\*\* 0.5 a3 1e-05 e

95\$\$ 0 t

Decay (5.0s - 10.0s)

0 MTU

60\*\* 5.5 6.0 6.5 7.0 7.5 8.0 8.5 9.0 9.5 10.0

61\*\* f1E-24

65\$\$

'Gram-Atoms Grams Curies Watts-All Watts-Gamma

3z 1 0 0 3z 3z 3z 6z

3z 1 0 0 3z 3z 3z 6z

3z 1 0 0 3z 3z 3z 6z

81\$\$ 2 0 26 1 e

82\$\$ 2 2 2 2 2 2 2 2 2 e

83\*\* **ENERGY\_BINS** e

t

56\$\$ 0 0 a10 1 e t

56\$\$ 0 0 a10 2 e t

56\$\$ 0 0 a10 3 e t

56\$\$ 0 0 a10 4 e t

56\$\$ 0 0 a10 5 e t

56\$\$ 0 0 a10 6 e t

56\$\$ 0 0 a10 7 e t

56\$\$ 0 0 a10 8 e t

```
56$$ 0 0 a10 9 e t
56$$ 0 0 a10 10 e t
54$$ a8 1 a11 0 e
56$$ f0 t
end
=opus
LIBUNIT=21
TYPARAMS=NUCLIDES
UNITS=GRAMS
LIBTYPE=ALL
TIME=SEC
NPOSITION=1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 end
end
=opus
LIBUNIT=21
TYPARAMS=GSPECTRUM
UNITS=GRAMS
TIME=SEC
NPOSITION=1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 end
end
#shell
end
```