

2-1-2016

PREDICTING ACTIVATION OF EXPERIMENTS INSIDE THE ANNULAR CORE RESEARCH REACTOR

Joseph Greenberg

Follow this and additional works at: https://digitalrepository.unm.edu/ne_etds

Recommended Citation

Greenberg, Joseph. "PREDICTING ACTIVATION OF EXPERIMENTS INSIDE THE ANNULAR CORE RESEARCH REACTOR." (2016). https://digitalrepository.unm.edu/ne_etds/43

This Thesis is brought to you for free and open access by the Engineering ETDs at UNM Digital Repository. It has been accepted for inclusion in Nuclear Engineering ETDs by an authorized administrator of UNM Digital Repository. For more information, please contact disc@unm.edu.

Joseph Isaac Greenberg

Candidate

Nuclear Engineering

Department

This thesis is approved, and it is acceptable in quality and form for publication:

Approved by the Thesis Committee:

Dr. Gary W. Cooper, Chairperson

Dr. Cassiano Ricardo Endres De Oliveira, Member

Dr. Ronald A. Knief, Member

**PREDICTING ACTIVATION OF EXPERIMENTS INSIDE
THE ANNULAR CORE RESEARCH REACTOR**

by

JOSEPH ISAAC GREENBERG

**B.S. CHEMICAL ENGINEERING, UNIVERSITY OF NEW MEXICO, 2008
C.O.C. ADVANCED NUCLEAR POWER COURSE, NAVAL NUCLEAR POWER
TRAINING COMMAND, 2009**

THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

**Master of Science
Nuclear Engineering**

The University of New Mexico
Albuquerque, New Mexico

December 2015

Dedication

I dedicate this paper to my wife, Lesley Greenberg, for constantly encouraging me, feeding me, cleaning up after me, and doing anything she could to help me keep a positive attitude and to never give up. She motivates me to do great work no matter what and I would not have been able complete this without her.

Acknowledgments

I would like to thank Dr. Gary Cooper, my advisor, for his guidance and support which made it possible for me to work on a topic that is of great interest to me.

I am grateful for Isaac McGarvey for his C# programming help and patience teaching me the language and taking time to make sure I understood exactly what was happening in my code.

I acknowledge with gratitude Dr. Billy Martin who introduced me to CINDER and gave me a thorough tutorial and quickly answered question that arose as the project progressed.

I thank Dr. Edward Parma for providing the base ACRR MCNP models and helping me determine how to adjust them to fit my needs.

I am extremely grateful for ACRR Staff (Lonnie Martin, Lance Lippert, Dave Clovis, Krista Kaiser, Jim Klein, Josh Emmer, John Hall, and Rafe Campbell) and RCTs (Roger Martinez, Shelly Siddoway, Skip Danielson, Chris Williams, and Steve Bonadore) for their support, encouragement, patience, and willingness to help whenever they could.

PREDICTING ACTIVATION OF EXPERIMENTS INSIDE THE ANNULAR CORE RESEARCH REACTOR

**by
JOSEPH ISAAC GREENBERG**

B.S. CHEMICAL ENGINEERING, UNIVERSITY OF NEW MEXICO, 2008

C.O.C. ADVANCED NUCLEAR POWER COURSE, NAVAL NUCLEAR POWER
TRAINING COMMAND, 2009

MASTER'S OF SCIENCE DEGREE IN NUCLEAR ENGINEERING, UNIVERSITY OF NEW
MEXICO DECEMBER 2015

Abstract

The objective of this thesis is to create a program to quickly estimate the radioactivity and decay of experiments conducted inside of the Annular Core Research Reactor (ACRR) at Sandia National Laboratories and eliminate the need for users to write code. This estimation is achieved by using MCNP to model the neutron fluxes in the reactor's central cavity where experiments are conducted using one of the four possible neutron spectra available in the ACRR. The desired neutron spectrum, experiment material composition, and reactor power level are then input into CINDER2008 burnup code to obtain activation and decay information for every isotope generated. DREAD creates all of the files required for CINDER2008 through user selected inputs in a graphical user interface and executes the program for the user and displays the resulting estimation for dose rate at various distances. The DREAD program was validated by weighing and measuring various experiments in the different spectra and then collecting dose rate information after they were irradiated and comparing it with the dose rates that DREAD predicted. The program provides results with an average of 17% higher estimates than the actual values and takes seconds to execute.

Table of Contents

Table of Contents

List of Figures	vii
List of Tables	viii
List of Equations.....	ix
Preface	x
1. Introduction	1
2. Background	4
3. Computational Approach.....	11
Overview	11
MCNP	11
CINDER	18
DREAD	22
4. Empirical Approach	29
5. Results	36
6. Conclusion	44
Future Work	45
Appendix A – MCNP Code.....	46
APPENDIX B – Example CINDER Files	73
Input:.....	73
Fluxes:	73
Locate:.....	73
Material:.....	73
APPENDIX C – DREAD C# Code.....	78
Bibliography	114

List of Figures

<i>Figure 1 – Three Experimental Cavities at the Annular Core Research Reactor Facility (ACRRF) During a Pulse</i>	1
<i>Figure 2- Experiment locations at ACRRF</i>	7
<i>Figure 3 - View of the rod control drive motors (and me) surrounding the opening for the central cavity</i>	8
<i>Figure 4 - Free Field Neutron Energy Flux per MW</i>	12
<i>Figure 5 - Vised side cross section view of Free Field MCNP input model</i>	13
<i>Figure 6 - 44 inch Lead Boron Bucket</i>	14
<i>Figure 7 - 44 inch lead boron bucket with 100,000 particles energy flux per MW</i>	15
<i>Figure 8 - 44 inch lead boron bucket with 10,000,000 particles energy flux per MW</i>	15
<i>Figure 9 - Poly-Lead-Graphite bucket cross section</i>	16
<i>Figure 10 - Bucket energy flux per MW comparison</i>	16
<i>Figure 11 - Bateman equation w/ variable explanations from CINDER User Manual prepared by Billy Martin 2014</i>	20
<i>Figure 12 - Example of flux differences in central cavity based on location- box vs. sphere</i>	21
<i>Figure 13- DREAD Thesis Example Run of a dosimetry packet for a 100 MJ reactor pulse with a decay time of 1 hour 2 minutes and 3 seconds</i>	24
<i>Figure 14 - Steady state example</i>	25
<i>Figure 15 - Teletector</i>	30
<i>Figure 16 - Thermo Eberline RO20</i>	30
<i>Figure 17 - Example of On-Contact dose rate measurement</i>	33
<i>Figure 18 - Example of 1-foot dose rate measurement</i>	33
<i>Figure 19 - Dosimetry pack (left) Microchip (right)</i>	36
<i>Figure 20 - Example of more complex electronics assembly</i>	36
<i>Figure 21 - 10 MJ pulse on 1 gram of commonly used metals</i>	42
<i>Figure 22 - 10 MJ pulse on metals with higher activation and dose rates</i>	42
<i>Figure 23 - 10 MJ pulse on 1 gram of common non-metals</i>	43

List of Tables

<i>Table I - Gamma flux-to-dose rate conversion factors</i>	<i>27</i>
<i>Table II - Free Field Results of DREAD vs. Actual Dose Rates</i>	<i>38</i>
<i>Table III - Lead Boron Bucket Results of DREAD vs. Actual Dose Rates</i>	<i>39</i>
<i>Table IV - Poly Lead Graphite Bucket Results of DREAD vs. Actual Dose Rates</i>	<i>40</i>
<i>Table V - Lead Poly Bucket Results of DREAD vs. Actual Dose Rates</i>	<i>40</i>
<i>Table VI - Comparison of DREAD vs. Actual results and factors between 1-foot and on-contact</i>	<i>40</i>

List of Equations

<i>Equation 1- Total Neutron Flux Scaling Factor for MCNP F4 Tally</i>	<i>17</i>
--	-----------

Preface

This thesis is written to complete a master's of science in nuclear engineering. I currently hold a bachelor's of science in chemical engineering from the University of New Mexico as well as a certificate of completion for the Advanced Nuclear Power Course from the Naval Nuclear Power Training Command. I was formerly a shielding engineer for Knolls Atomic Power Laboratory (KAPL). I then became a qualified Engineering Officer of the Watch (EOOW), an instructor, and drill coordinator at the Kesselring site for KAPL. For the past 5 years I have been a nuclear reactor engineer and operator at Sandia National Laboratories (SNL) at the Annular Core Research Reactor (ACRR).

My work experiences inspired me to attain higher education in nuclear engineering and presented me with a problem that had no easy solution, which has become the topic for this thesis. The goal is to provide myself and coworkers with a tool that is easy to learn and will run quickly on any computer for accurate and conservative results to aid in ALARA practices.

1. Introduction

Understanding the effects of radiation interactions with matter is an ongoing effort that has wide variety of applications. These applications include: radiation hardening, radiation damage, gamma and x-ray environment testing, neutron environment testing, space testing, reactor component testing, radiation therapy, and numerous others (Attix, 1986). The use of test reactors is an excellent method for conducting experiments for any of the aforementioned applications. However, the benefits of the radiation studies come with consequences that can be viewed as undesirable or adverse; particularly with respect to health of personnel. This paper focuses on a method to better control and reduce these detrimental effects of using nuclear reactors for radiation studies by providing an improved method for predicting both the radioactivity that will be induced in the experiments placed in the reactor and the dose rates that personnel will be exposed to in handling the activated experiment.

Reactor operators and experimenters at the Annular Core Research Reactor (ACRR) are exposed to various sources of radiation on a daily basis. The facility is home to neutron and gamma sources, an open pool nuclear reactor with three dry



Figure 1 – Three Experimental Cavities at the Annular Core Research Reactor Facility (ACRRF) During a Pulse

experiment cavities, transuranic isotopes, reactor fuel, and various forms of radioactive materials (Figure 1). The sources of radiation that the operators are potentially exposed to are all necessary for carrying out the mission of Sandia National Laboratories (SNL). While exposure of personnel to radiation is unavoidable, it is essential that all work be planned and conducted in such a manner that all exposures are As Low As Reasonably Achievable (ALARA). By practicing ALARA, deleterious effects of radiation to personnel are minimized and the public and governing agencies will view the work as being conducted in a safe and responsible manner. Understanding the existing sources of radiation and the levels at which personnel are exposed is a relatively simple task and the radiation can be controlled accordingly. The radioactivity of experiments that are irradiated inside of one of the cavities at the ACRR, however, currently cannot be measured prior to removal from the reactor. Thus, the exposure rate generated by an experiment upon removal from the reactor is not accurately known and presents a potential health risk to personnel.

One method for predicting the activation of experiments is to rely on the experience of the operators. This approach is discussed below and is necessarily subject to several limitations. Computer simulations, on the other hand, probably represent the best method for predicting the activation of materials irradiated inside of the ACRR. Historically, experiments have been modeled using the neutron photon Monte Carlo transport code, MCNP (LANL). While MCNP will provide accurate predictions if the experiment and neutron source are accurately

modeled, Using MCNP routinely presents several practical problems. MCNP is a complex code that requires experienced users for both modeling new experiments and to implement changes to pre-existing models as experiments are modified. The calculations take a considerable amount of time to run (upwards of a month on a PC) or, alternatively, must be placed in queues on the supercomputer. As a result of the long calculation times, modeling an ever-increasing number of experiments becomes less and less practical and essentially eliminates the modeling of any last minute changes. To solve these problems, this thesis created a graphical user interface (GUI) in C# that uses the flux profile from various MCNP reactor models and applies it to the experiments with the CINDER (Holloway) activation code to predict the activity induced in an experiment and to estimate the associated dose rate that operators and experimenters will be exposed to when removing irradiated experiments from the reactor core. This resulting program is titled "Dose Rate Estimator for Activation and Decay": DREAD.

2. Background

The biological effects of radiation have been well studied in multiple groups of people: medical radiation recipients, radium-dial painters, uranium miners, nuclear accidents, and atomic bomb survivors (Turner, 2007). The effects can be grouped into two categories; acute and delayed. The acute effects are commonly referred to as radiation sickness, which is defined as, [“The complex of symptoms characterizing the disease known as radiation injury, resulting from excessive exposure (greater than 200 rads or 2 gray) of the whole body (or large part) to ionizing radiation. The earliest of these symptoms are nausea, fatigue, vomiting, and diarrhea, which may be followed by loss of hair (epilation), hemorrhage, inflammation of the mouth and throat, and general loss of energy. In severe cases, where the radiation exposure has been approximately 1000 rad (10 gray) or more, death may occur within two to four weeks. Those who survive six weeks after the receipt of a single large dose of radiation to the whole body may generally be expected to recover”] (NRC Glossary).

The delayed effects, also known as somatic or latent effects, may take a long time to manifest themselves. Some of these effects are cancer, cataracts, life-shortening, sterility, effects on the fetus, and multiple others. The likelihood of developing one of these effects is dependent on the amount of dose received and over what period of time it was received. The U.S. government regulates the amount of radiation a worker can receive in a year to 5 rem, and companies will typically make separate, more conservative policies on how they handle dose. For example, the maximum amount of dose a worker can receive in a year

without needing an approval to increase their allowed level at Sandia National Laboratories is 250 millirem. Organizations within the labs further limit and regulate the dose rates that their personnel are allowed to be exposed to and monitor the staff daily to ensure that radiation level limits are not exceeded. The long term and short term effects on humans from excessive radiation exposure can be very serious and minimizing exposure is a very high priority. An accurate prediction of the doses workers can expect from irradiated packages at the ACRR can aid in reducing the exposure of the workers.

For the acute cases of radiation effects, the rad (radiation absorbed dose) unit is typically used. However, for delayed effects of radiation the units of dose received by the individual are measured in terms of rem (roentgen equivalent man) for a dose equivalent quantity (Shleien, 1984). The dose rates displayed in DREAD are in mrem/hr, which is the unit used in the documentation controlling work at the ACRR. Millirem is based on the radiation absorbed and is scaled by a quality factor which is a measure of how severely the body will react to a given type of radiation. Gamma and beta have the lowest quality factors and neutrons and alphas have the highest. Since the radiation of interest is primarily gamma radiation from irradiated materials, the quality factor used in this work is just “one”. While, it is possible to use the program to solve transuranic experiments, it would require changes to the factors and programming. Since experiments of this type are infrequent and are planned far in advance, it would be better to fully

model these experiments with MCNP rather than use the approach developed in this thesis.

The Annular Core Research Reactor is very unique. It utilizes a BeO-UO₂ ceramic fuel to conduct pulsing operations. A pulse is generated when the reactivity insertion in the reactor is greater than the effective delayed neutron fraction for that particular reactor. This value is commonly referred to as β for pulse type reactors. This means that the fission chain reaction has enough neutrons from the fission events to continue a constant or increasing power level, which is referred to as prompt critical. There is no safety system fast enough to control a reactor of this type, and without a mechanism to shut down, the reactor will turn into a single use reactor (bomb). The ACRR is not highly enriched, and does have a mechanism to shut down. This mechanism is called Doppler broadening and is the result of the cross section resonances of U-238 expanding and absorbing neutrons as the fuel heats up. These neutrons do not cause fissions and are essentially eliminated from the neutron lifecycle. Once the absorption rate becomes fast enough to make the reactor subcritical, the reactor power will rapidly decrease. By the time the fuel cools off enough to reduce the negative reactivity induced by the Doppler broadening, the control rods have been dropped back into the core and the excursion is terminated. The rapid rise and fall of power occurs in about 7ms at Full Width at Half Maximum (FWHM) and is called a pulse.

Most experiments at the ACRRF are conducted inside the ACRR central cavity.

The central cavity is a 9" diameter stainless steel cylinder that extends from above the upper bridge plate to the bottom of the reactor tank. It is equipped with a cavity purge system that creates a negative pressure inside of the cavity by drawing air through a series of filter banks and then releasing it through a stack above the facility. Typically, a 32-inch pedestal is placed at the bottom of the central cavity so that experiments can easily be placed at the flux centerline of the reactor, which is

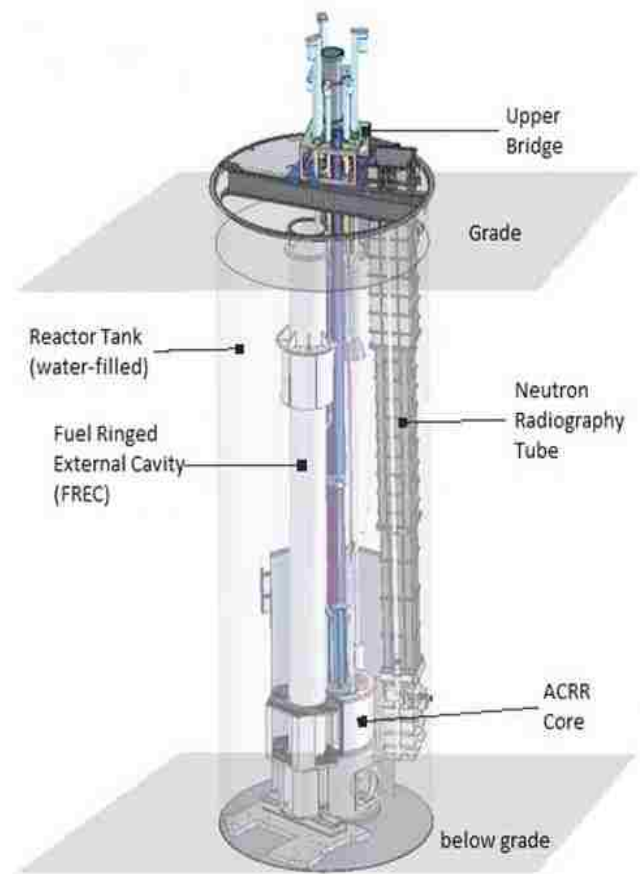


Figure 2- Experiment locations at ACRRF

about 14 inches above the top of the pedestal. This is the location where DREAD is currently calibrated. There are other locations for experiments to take place at the facility including the neutron radiography tube, the FREC-II external core, and in-core test locations (Figure 2). DREAD currently only models the ACRR central cavity neutron activation, but in the future could be used for any of the locations or expanded for larger experiments where accuracy is more important than conservatism.

The experiments at ACRR range from small dosimetry foils to large complex electronics assemblies. The neutrons in the core are mostly epithermal, about 10 keV on average, and, for many experiments, this is not the desired energy for the neutrons. For example, an experimenter may want to determine how their electronics will react to a nuclear weapon detonation in the atmosphere where the neutron spectrum is significantly harder than the spectrum in a water moderated reactor. In general, thermal neutrons are much more readily absorbed by materials versus fast neutrons. Many exhibit $1/v$ absorption characteristics, meaning that the lower the energy of the neutron, the more likely it is to be absorbed. To modify the energy of the neutrons that the experiment is subjected to, buckets composed of different materials are used. There are four different modifier regimes that the reactor currently utilizes: free field, a lead boron bucket for emphasis on fast neutron interaction, a lead poly bucket for thermal interaction, and a poly lead graphite bucket that serves as another thermal modifier. These spectrum modifiers are discussed and compared in more detail in the computer codes section.

After the target has been irradiated, the reactor operators must remove it from the central cavity using a 3-ton crane.

Upon removal from the cavity,



Figure 3 - View of the rod control drive motors (and me) surrounding the opening for the central cavity

experimenters will climb a few steps, and standing on the upper bridge plate (Figure 3) will guide it to its next destination, which could be into a floor storage hole, onto a workbench, or into a shielded holding cell. This process involves the experimenter putting a plastic sleeve around the experiment and guiding it by hand to the proper area to make sure it does not come into contact with other structures or components. The typical work control allows experiments to be removed from the central cavity if the contact exposure rate is less than 30 R/hour. DREAD will give these experimenters and operators a better idea of how long to wait prior to removing an experiment, what exposure rates to expect, and how to prepare when pulling up an experiment. Minimizing exposure once the experiment package has been removed is still reliant on time, distance, and shielding; all of which are controlled by the operators and experimenters to some degree.

Currently, the expectations and wait times are all generally estimated from prior experiences with similar experiments. The reliance on experience falls short in a few areas. Aging staff members who retire and staff members who choose to change jobs have a lot of prior knowledge that leaves when they leave. New employees trying to determine wait times and expected radiation dose rates will lack this essential historical knowledge and may result in workers receiving more dose than “As Low As Reasonably Achievable”. Further, new experiments and materials irradiated may not be fully understood by the senior staff and the best guess techniques would typically be used. Given the limited number of staff

members, the low, allowable exposure limits for personnel, and the significant increase in the number of experiments that must be performed requires better anticipation of irradiate experiment dose rates. DREAD is a solution to these issues.

3. Computational Approach

Overview

DREAD utilizes three separate computer codes to function. The three codes are: MCNP 6.1, CINDER 2008, and Microsoft Visual Studio Professional 2013 © in C# language. MCNP is used to generate a model of the neutron flux in ACRR as a function of location and energy and this output is used as input to the energy group fluxes required for running CINDER. CINDER uses an algorithm to calculate the activation/transmutation of isotopes and outputs radiation, energies, curies, among other quantities for each isotope entered and created. The C# programming generates a GUI that anyone familiar with the reactor will be able to use. The user will be able to select from a list of a number of typical, reactor set-up configurations and from a number of commonly used materials that are to be irradiated. The GUI generates the four input decks required by CINDER based on the user's input and automatically will run CINDER. It then deciphers the output files from CINDER for the user and will display the mrem/hr they can expect to see after their selected irradiation and wait times.

MCNP

MCNP was chosen because the reactor models that have been validated for the ACRR have been created in MCNP and the code is available for use. The website description of MCNP states, "MCNP is a general-purpose Monte Carlo N-Particle code that can be used for neutron, photon, electron, or coupled neutron/photon/electron transport. Specific areas of application include, but are not limited to, radiation protection and dosimetry,

radiation shielding, radiography, medical physics, nuclear criticality safety, Detector Design and analysis, nuclear oil well logging, Accelerator target design, Fission and fusion reactor design, decontamination and decommissioning. The code treats an arbitrary three-dimensional configuration of materials in geometric cells bounded by first- and second-degree surfaces and fourth-degree elliptical tori” (LANL).

Modifications were done to the MCNP model (Appendix A) to better represent the present structure of the reactor, narrow the scope of the model to look solely at neutron interactions vice photon and neutron interactions, and form energy groups consistent with the group numbers required by CINDER. Different models needed to be run for each of the different spectrum modifying buckets.

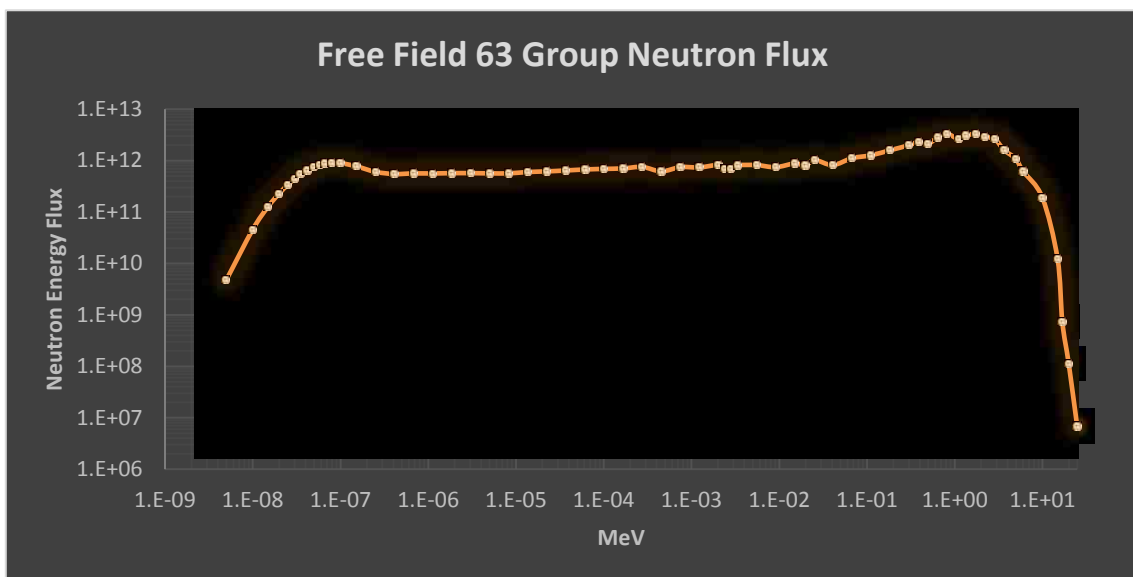


Figure 4 - Free Field Neutron Energy Flux per MW

Alterations of the code can be seen commented out in Appendix A. The most basic reactor model is the free field model (Figures 4 & 5). As the name states,

there is not a spectrum modifier in the neutron field. Experimenters who are satisfied with the neutron spectrum of the reactor without modification will load their experiments into the central cavity on aluminum stands or in thin walled aluminum cans which have little effect on reactivity. Lines 1079 to 1115 in Appendix A are the f4 tally setup, with 1001 signifying a 6-cm sphere of interest at the centerline of the flux, number of particles to start with, and the mode for running the MCNP model. The number of starting particles was the number required to result in statistically smooth function graphs.

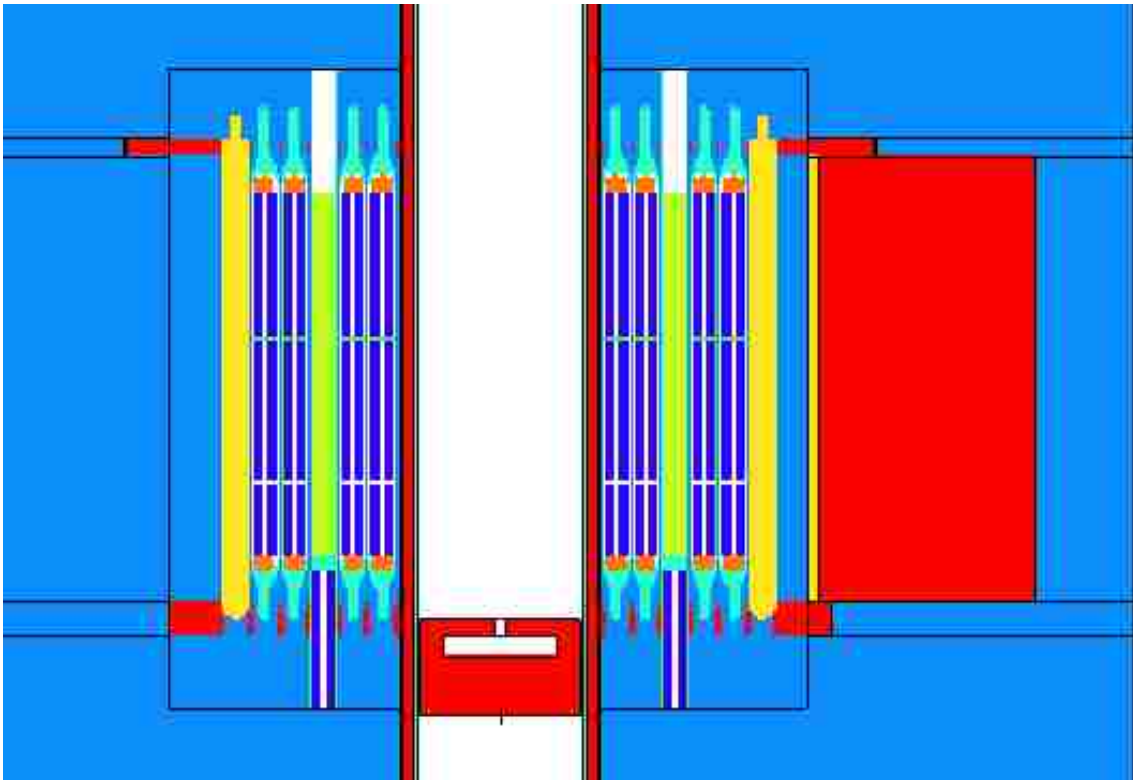
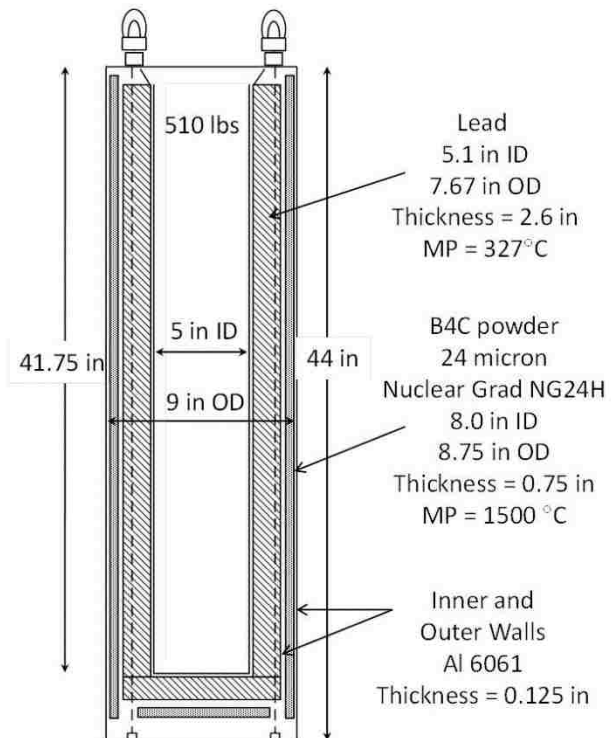


Figure 5 - Vised side cross section view of Free Field MCNP input model

Fewer particles resulted in energy band fluxes that appeared to jump orders of magnitude between adjacent energy bands, which is not indicative of actual neutron thermalization. These numbers stayed the same for all kcode runs. The

variations in the code begin at line 265, where the commented sections were uncommented to add various spectrum modifiers to the code accordingly.

The 44 inch lead boron bucket (Figure 6) configuration exhibited the most sensitivity to the number of particle histories ran as shown in Figures 7 and 8. It is clear that from the 100,000 particle run that the thermal neutron energy flux was subject to unrealistic statistical fluctuations. The 10,000,000 particle k-code run, however, produced much better results.



However, it required using Sandia *Figure 6 - 44 inch Lead Boron Bucket* National Laboratories' super computer to finish within a reasonable amount of time versus a desktop PC which would have taken over a month of runtime.

The portion of the spectrum that is non-physical in the 100,000 particle run is similar in shape but not magnitude to that of the 10,000,000 particle run. Since the thermal region for neutrons is where the majority of absorptions take place, however, it needs to be as accurate as possible to get accurate, useful results from CINDER. While the calculated spectra from the other geometries did not display the same degree of sensitivity, they were rerun on the supercomputer

with 10,000,000 particles for consistency and the all the resulting calculated spectra were smoothed to give a better representation of the actual neutron energy flux in the 6-cm sphere.

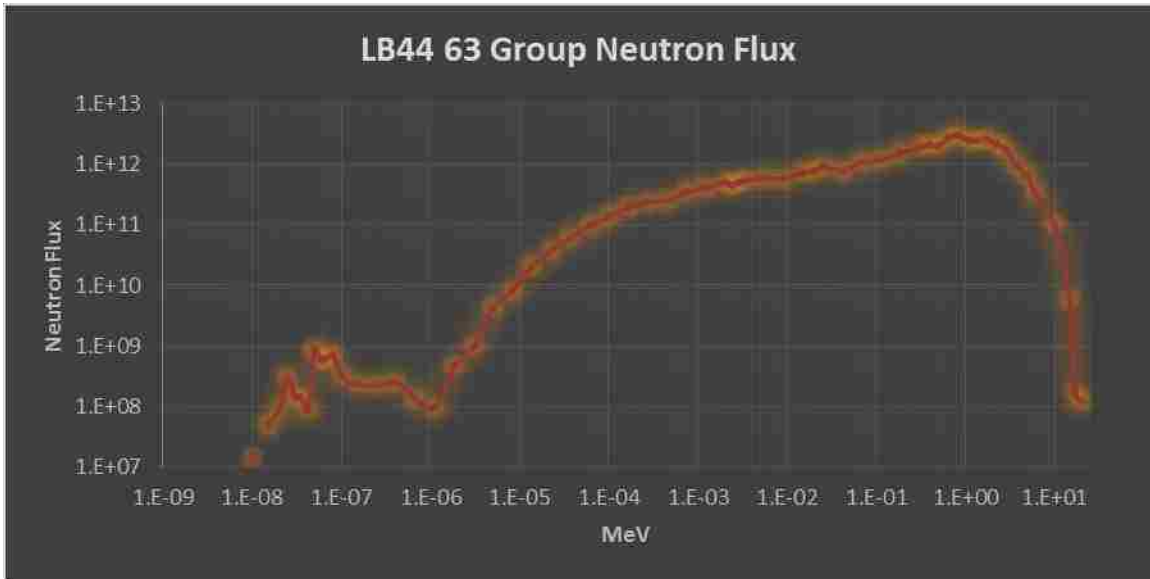


Figure 7 - 44 inch lead boron bucket with 100,000 particles energy flux per MW

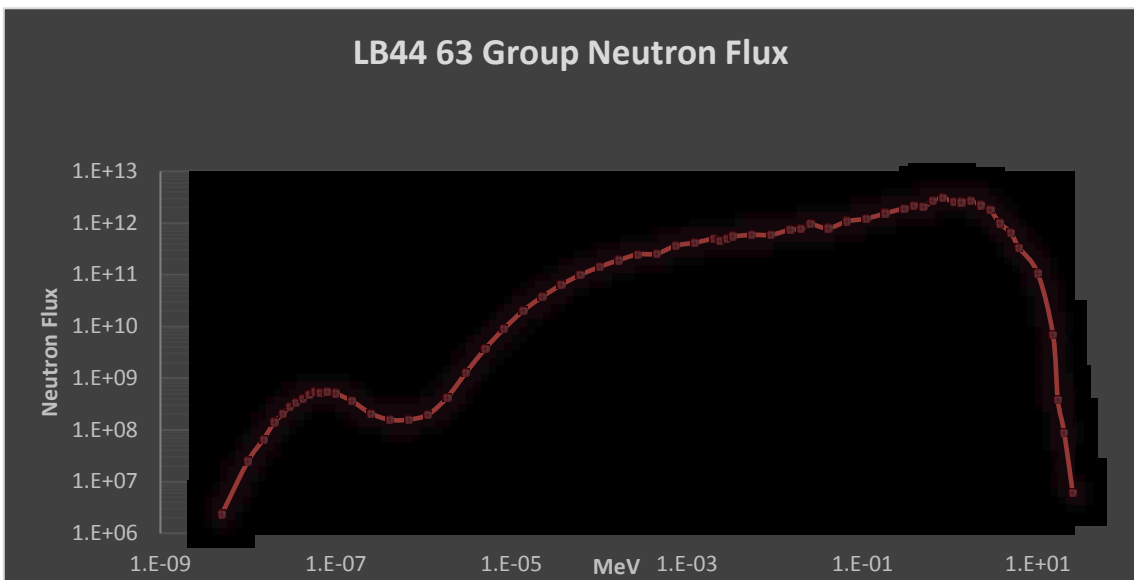


Figure 8 - 44 inch lead boron bucket with 10,000,000 particles energy flux per MW

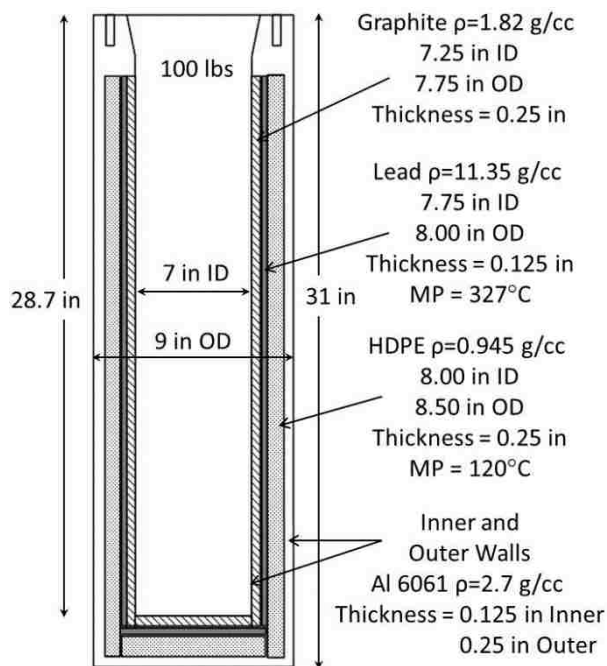


Figure 9 - Poly-Lead-Graphite bucket cross section

Plots for the poly-lead-graphite

(Figure 9) and lead-poly buckets

were also generated. All of the plots

were compared on a single graph to

better visualize their effects on the

neutron energy inside of each

bucket (Figure 10). This graph is

on a log-log scale to fit all of the

information in one view. To the eye

this log-log view tends to

underemphasize the significant

differences between the spectra, but several large differences can be noted. The

most obvious difference is between

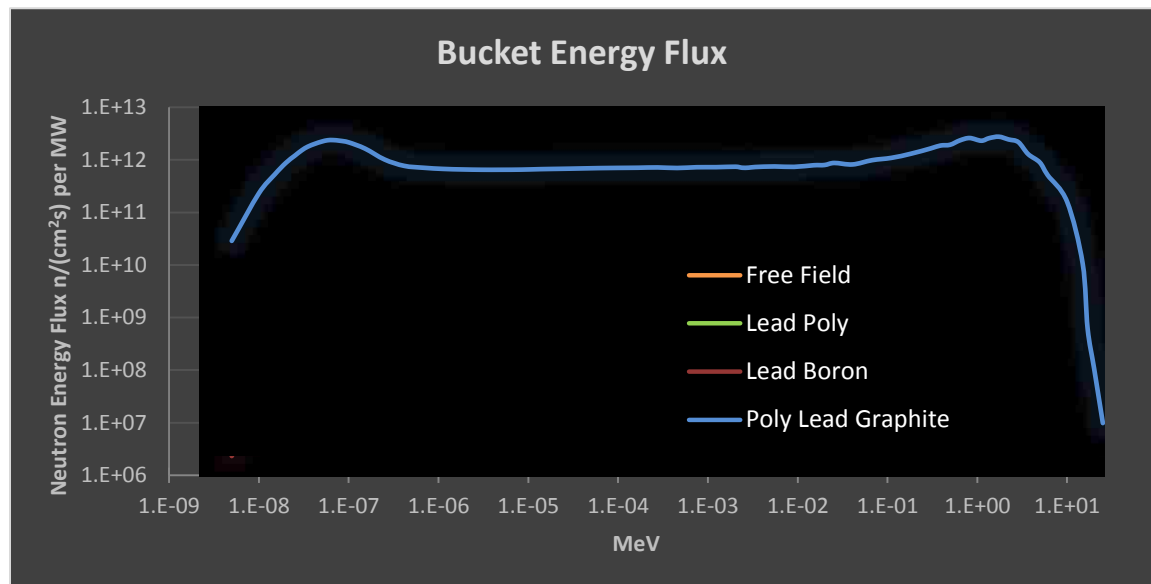


Figure 10 - Bucket energy flux per MW comparison

the lead boron bucket and the rest of the buckets. The thermal neutron flux reaching the 6 cm sphere of interest is many orders-of-magnitude lower than the next lowest thermal neutron flux: free field. This information is useful to experimenters who desire to remove or generate thermal neutrons for their experiments. An experimenter who is not familiar with nuclear reactors may make requests that do not suit his/her needs and it is the duty of the reactor operations staff to help interpret their goals. Neutron damage to electronic parts is often related to 1-MeV, silicone-equivalent neutrons (Williams, 2007). An experimenter who wants thermal neutron absorption would be advised to use the poly lead graphite bucket. These buckets require about 1,000 to 10,000 times less energy from the reactor to achieve the same number of thermal neutrons interacting with the target as in a lead boron bucket operation and 10 times less energy than a free field operation. Experimenters only interested in fast neutron interaction might consider using the lead boron bucket, which filters out a factor of 1,000 thermal neutrons from interacting with the experiment.

The plots (Figures 4, 8, and 10) were created from the 63 group neutron tally calculated from MCNP. To convert the output of MCNP into useful information,

$$\Phi \left[\frac{\text{neutron}}{\text{cm}^2 \text{s}} \right] = \frac{P [\text{W}] \bar{\nu} \left[\frac{\text{neutron}}{\text{fission}} \right]}{\left(1.6022 \cdot 10^{-13} \frac{\text{J}}{\text{MeV}} \right) w_f \left[\frac{\text{MeV}}{\text{fission}} \right]} \frac{1}{k_{\text{eff}}} \Phi_{F4} \left[\frac{1}{\text{cm}^2} \right]$$

Equation 1- Total Neutron Flux Scaling Factor for MCNP F4 Tally

P – Power of the reactor in watts
 $\bar{\nu}$ – Average neutrons per fission 2.44
 w_f – energy released per fission 192.4 MeV
 ϕ – neutron flux in energy band
 ϕ_{f4} – f4 tally result from MCNP
 k_{eff} – MCNP effective multiplication factor

several transformations of the data were required. The output from MCNP for

these runs is per source neutron. To convert the fission source neutrons into neutron flux at different energy levels, a scaling factor was used to convert source fission neutron flux into flux per MW. This scaling factor was then multiplied by the f4 tally flux and divided by the keff of the kcode run in order to determine the flux inside each energy band. Equation 1 (Snoj, 2006) represents the calculation to determine each energy group flux. Once the flux in each energy group was determined, it was averaged over the energy group width to display on a point on the graph per average MeV in the band rather than per energy band used, which would form a step function graph. All of the MCNP computations were completed once all four plots were generated and fluxes were determined for each energy group for each of the four spectrum modifiers.

CINDER

CINDER2008 is a code used to calculate the inventory of nuclides in an irradiated material (England, 1964). This is also referred to as activation/depletion, burn-up, transmutation, etc. CINDER was chosen because of its extremely fast run time and it uses nuclide data libraries that have already been established, such as ENDF-7. It requires a very specific set of inputs consisting of at least four different files and many variables that have stringent numbers of characters. Creating the input files for CINDER is a very tedious and time consuming process. The four different file types required for CINDER to run are: input, fluxes, locate, and material.

The input file includes a calculation name, a free form comma separated string

which includes the volume of material and the flux multiplier, a description, the flux file's name, the material file name, the flux on time and multiplier, and the wait time. This file tells CINDER what files to use for data generation, how much to multiply the basic flux of 1 MW to achieve the user desired flux, how long to run the reactor, and how long to wait before removing the sample. The way input is created results in two sets of output, the isotope information at the time the reactor was shut down and the information when the experiment is going to be removed (See Input in Appendix B).

The Fluxes file is what CINDER uses as the base flux for the input file. It consists of a title, the number of neutron energy groups, the flux name, the total neutron flux, and the neutron flux in each energy group. The fluxes entered in the total and individual energy group fluxes are reliant on the type of spectrum that the user wants to input. These values were calculated in Microsoft Excel 2013© from the MCNP generated spectra (See Fluxes in Appendix B).

The locate file tells CINDER where the library and executable for CINDER are located. It can be modified from the default setting, but it is in the C: folder by default. The string for the location is limited to 80 characters, so burying the file inside multiple folders may prove problematic. This file does not change from run to run unless the path for the CINDER files changes.

The material file is the largest of the files, as it includes the total number of nuclides to run, the total number of atoms/barn-centimeter, the AZS identifier for each nuclide, and the fraction of total atoms that is present from each nuclide. CINDER will recognize 0 as the fraction for each nuclide, so including every nuclide in a base file and writing values only for the nuclides present is acceptable. For DREAD, every stable isotope for each element up to lead is included in the material file. An example of a material file can be seen in Appendix B under Material. DREAD creates all of these files for the user and automatically runs the CINDER code with the desired spectrum and materials. The method for this is described in the next section.

The method CINDER uses to determine the activation and decay is the Bateman equation shown in Figure 11.

$$\frac{dN_m(t)}{dt} = -N_m(t)\beta_m + \bar{Y}_m + \sum_{k \neq m} N_k(t)\gamma_{k \rightarrow m}$$

$$\beta_m = \lambda^m + \phi_n \sigma_{n,abs}^m + \phi_g \sigma_{g,abs}^m$$

Number Density of Isotope m points to $N_m(t)$.
 Loss Fraction for Isotope m points to β_m .
 "Other" Loss or Gain for Isotope m points to $\bar{Y}_m$.
 Fraction of Isotope k that Transmute to Isotope m points to $\gamma_{k \rightarrow m}$.
 Rate of Natural Decay for Isotope m points to λ^m .
 Neutron Flux points to ϕ_n .
 (n,abs) Cross-Section for Isotope m points to $\sigma_{n,abs}^m$.
 Photoflux points to ϕ_g .
 (gamma,x) Cross-Section for Isotope m , resulting in isotope $\neq m$ points to $\sigma_{g,abs}^m$.

Figure 11 - Bateman equation w/ variable explanations from CINDER User Manual prepared by Billy Martin 2014

The equation is a sum of the losses, gains, and transmutations of a particular isotope. CINDER solves all of the decay chains simultaneously for all of the isotopes present and generated in the input problem. The output files from cinder are very detailed and provide information about every nuclide at each time step. The two output files of most interest for this paper are the tables_by_grp and tables_by_major. The tables_by_grp file contains the gamma radiation information that is used to determine dose rates. It displays the gamma contribution including the number of gammas and the average energy for each isotope and a t all isotopes with an overall average energy. The tables_by_major file contains information about each isotope such as mass, curies, decay power, etc.

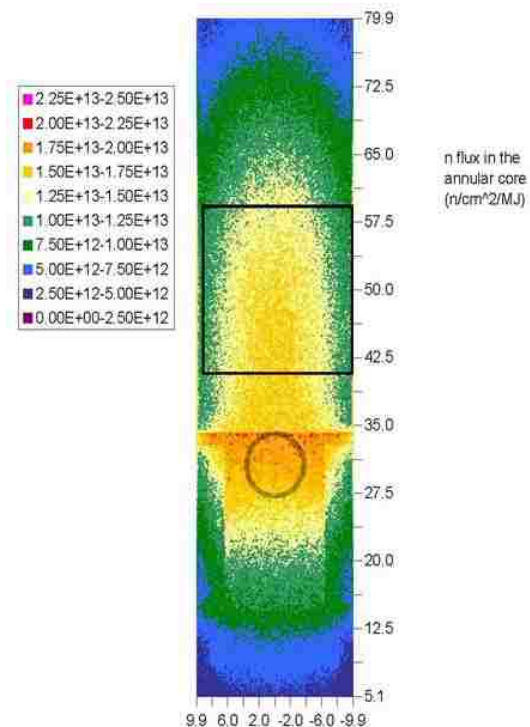


Figure 12 - example of flux differences in central cavity based on location- box vs. sphere

With the information from these two tables it is possible to determine dose rates, keeping in mind the limitations of using CINDER. The program does not account for self-shielding and assumes all of the atoms see the full flux that the user inputs. For smaller parts and materials this is very close to reality, but for larger objects or isotopes with very large neutron cross sections, the inside of the part or experiment will see a significantly lower flux

than the portion nearest the reactor. The self-shielding also applies to determining a dose rate. If the part is assumed to be a point source then there is no accounting for self-shielding the gamma radiation it is emitting. Both of these cases will result in a conservative dose calculation, which is desired if determining a wait time to safely remove an experiment from the central cavity. Larger experiments that are outside the small 6-cm sphere that was determined to be the largest flux in the reactor will also have overestimated calculated dose. The flux in the reactor changes outside of the 6-cm sphere of interest as seen in Figure 12. For example, if the experiment were 9 inches in diameter and higher up in the cavity, such as the location of the cylinder in Figure 12, then the flux would be about half of that in the sphere region in Figure 12 which is what was analyzed in the computer models. A new flux tally would be required to get model results closer to empirical data.

DREAD

DREAD – Dose Rate Estimator for Activation and Decay – is a graphical user interface that the operator and experimenters who do not know how to code CINDER/MCNP can use to get a quick estimation of dose rates (and other activation information) when removing experiments from the central cavity at ACRR. The program was created using Microsoft Visual Studio in the C# programming language. This style of programming allows the programmer to easily create the layout of the windows they are working on and then program the objects created in the window to perform however he/she desires. This method was chosen because it is in a style similar to many common programs used in

windows, it is easy to manipulate the user interface, the programming language is common, and it is simple to debug. The code is located in Appendix C.

DREAD has two slightly different user input choices based on the type of experiment being conducted. ACRR operates in steady state modes and in pulse modes. The most common mode is the pulse mode in a free field spectrum, which is the default setting in DREAD (see Figure 13). A typical DREAD run takes about 5 seconds to run and display results.

The screenshot shows the DREAD software interface with the following settings:

- Thesis Example Run:** Free Field (selected), Lead Boron Bucket, Poly Lead Graphite Bucket, Lead Poly Bucket.
- Select Mode:** Pulse (selected), Steady State.
- Megajoules:** 100.0
- Wait Time:** 1 hours, 2 minutes, 3 seconds.
- Component or Element:**
 - ☐ 5 Mil Nickel
 - ☒ 10 Mil Nickel (1.0)
 - ☒ Standard Sulfur (4)
 - ☐ Large Sulfur
 - ☒ TLD (4)
 - ☐ 6061 Aluminum Grams
 - ☐ 316 Stainless Steel Grams
 - ☐ Circuit Board Grams
 - ☐ PCB Electronics Grams
 - ☐ Cardboard
 - ☐ Copper 63
 - ☒ Polyethylene/propylene (0.050)
 - ☐ PVC
 - ☐ Teflon
- Element Grams Table:**

Element	Grams
Hydrogen	0.00000000
Helium	0.00000000
Lithium	0.00000000
Beryllium	0.00000000
Boron	0.00000000
Carbon	0.00000000
Nitrogen	0.00000000
Oxygen	0.00000000
Fluorine	0.00000000
Neon	0.00000000
Sodium	0.00000000
- Isotope Atoms/barn-cm Table:**

Isotope	Atoms/barn-cm
Hydrogen-1	0.004326997
Hydrogen-2	0.000000498
Helium-3	0.00000000
Helium-4	0.00000000
Lithium-6	0.00000000
Lithium-7	0.00000000
Beryllium-9	0.00000000
Boron-10	0.00000000
Boron-11	0.00000000
Carbon-12	0.002123423
Carbon-13	0.000022972
- Total Atoms/barn-cm:** 0.091347904
- Summary:**
 - Total Gammas: 2.485E+006
 - Average Gamma MeV: 1.071E+000
 - mRem/hr on Contact: 13.702
 - mRem/hr @ 1 foot: 0.457
 - mRem/hr @ 1 meter: 0.041

Figure 13- DREAD Thesis Example Run of a dosimetry packet for a 100 MJ reactor pulse with a decay time of 1 hour 2 minutes and 3 seconds

In the pulse mode, the user inputs the megajoules for the pulse, the time to wait before removing the experiment from the central cavity, and the material composition of the experiment. There are several common components that have been generated under the “Component or Element” menu (Figures 13 & 14). The user also can select the grams of any naturally occurring element. For isotopes that are enriched, the user enters the atoms/barn-cm or can request to generate a component in the component list, such as copper-63. In the example in Figure 13, the reactor is setup to perform a 100 MJ pulse on a dosimetry pack consisting of one nickel piece, four TLDs, four Sulfur tablets, and some plastic holding them together. The inputs to cinder can be seen in Appendix B for this

example. The dose-rate results boxes in DREAD are hidden until the user runs the program and the output files from CINDER are read by DREAD and dose-rate calculations are performed. The total number of gammas and the average gamma MeV are read from the output files of CINDER. DREAD calculates the dose rates at the various distances and displays the on-contact, the 1-foot, and 1-meter expected dose rates. The assumption in DREAD for a pulse is a 1-second time interval over which all of the energy is deposited.

The steady-state mode is slightly different from the pulse mode. The user must input the power level, either in % power or in MW, and the amount of time to run at that power. A separate calculation screen appears to the right of these inputs and tells the user how many MJ their run will produce in the reactor (Figure 14).

Thesis Example Run

Select Desired Spectrum

- ☒ Free Field
- ☐ Lead Boron Bucket
- ☐ Poly Lead Graphite Bucket
- ☐ Lead Poly Bucket

Select Mode

- ☐ Pulse
- ☒ Steady State

Enter %Power or Reactor Power (MW) **Total Energy (MJ)**

1.00000 0.0239 100.0

Wait Time **Irradiation Time**

1 hours 1 hours

2 minutes 9 minutes

3 seconds 13 seconds

Total Atoms/barn-cm: 0.091347904

Component or Element

- ☐ 5 Mil Nickel
- ☒ 10 Mil Nickel 1.0
- ☒ Standard Sulfur 4
- ☐ Large Sulfur
- ☒ TLD 4
- ☐ 6061 Aluminum Grams
- ☐ 316 Stainless Steel Grams
- ☐ Circuit Board Grams
- ☐ PCB Electronics Grams
- ☐ Cardboard
- ☐ Copper 63
- ☒ Polyethylene/propylene 0.050
- ☐ PVC
- ☐ Teflon

Element Grams

Hydrogen	0.00000000
Helium	0.00000000
Lithium	0.00000000
Beryllium	0.00000000
Boron	0.00000000
Carbon	0.00000000
Nitrogen	0.00000000
Oxygen	0.00000000
Fluorine	0.00000000
Neon	0.00000000
Sodium	0.00000000

Isotope Atoms/barn-cm

Hydrogen-1	0.004326997
Hydrogen-2	0.000000498
Helium-3	0.000000000
Helium-4	0.000000000
Lithium-6	0.000000000
Lithium-7	0.000000000
Beryllium-9	0.000000000
Boron-10	0.000000000
Boron-11	0.000000000
Carbon-12	0.002123423
Carbon-13	0.000022972

Total Gammas **Average Gamma MeV** **mRem/hr on Contact** **mRem/hr @ 1 foot** **mRem/hr @ 1 meter**

2.485E+006 1.071E+000 13.702 0.457 0.041

Figure 14 - Steady state example

This input will set up a different input deck configuration for CINDER by dividing the reactor power in MW by the initial default level of 1 MW to give a scaling factor to use when applying the spectrum flux to the experiment. There is currently only one step for this mode and it applies the flux evenly over the irradiation time. Further this mode treats the build-up and decay of the various radionuclides during the irradiation, which will result in a realistic activation profile at the end of the run and the end of the wait. The rest of the program is not affected by switching modes.

The user selects what spectrum they are interested in doing a prediction and spectrum's flux from MCNP is imported into the fluxes input file for CINDER. The material file is generated when the user runs the program and is generated based on the user inputs selected from the element grams or in the component section. Every atoms/barn-cm are updated whenever a value is changed in the elements or component section, so users desiring to use this feature are advised not to change the component or elements or their entries may be erased.

DREAD was written to give operators an idea of what dose rate to expect when removing experiments from the central cavity, but there is a large amount of data that is also produced from CINDER's calculations. The "View Input and Output Files" button opens the folder (has only been tested in Windows) where the input files and output files are located. Here, all of the input and output files from CINDER are located and can be opened in almost any text editing software.

From these files the user can access the information discussed in the CINDER section. If the experimenter wants to create a certain amount of curies of a particular isotope, DREAD can be used to provide an estimation of the number of curies created and what type of dose rate to expect from the irradiated material.

The dose calculation is based on a point source emitting gamma radiation uniformly in all directions. Lines 2205 to 2234 convert the gamma rays at the source to a flux at the desired distance (30 cm) from the source using this equation: $\phi = S/4\pi r^2$ where S is the source strength, r is the distance from the source in cm, and ϕ represents the flux per cm^2 - second. Simultaneously, the code multiplies the gamma flux by the corresponding gamma ray flux-to-dose conversion factor in Table I (Shleien, 1984).

Table I - Gamma flux-to-dose rate conversion factors

Gamma Ray Flux-to-Dose Rate Conversion Factors Polynomial Coefficients in Analytic Form $\ln D(E) = A + Bx + Cx^2 + Fx^3$ $D(E) = (\text{rem/h})(\text{cm}^2\text{-s})$, E = Photon Energy in MeV and $x = \ln E$ (After Unger and Trabey ORNL/RSIC-43 1981)				
Photon Energy (MeV)	A	B	C	F
0.01 to 0.03	-20.477	-1.7454		
0.03 to 0.5	-13.626	-0.57117	-1.0954	-0.24897
0.5 to 5.0	-13.133	0.72008	-0.033603	
5.0 to 15.0	-12.791	0.28309	0.10873	

The dose at 30 cm (~1-foot) is then used to determine the approximate dose at 1 meter and on-contact. The 1 meter calculation used the $1/r^2$ point source approximation (Lamarsh, 1983), which is $30^2/100^2$ or a factor of 0.09. The dose on-contact is a little more difficult to estimate. This is due to detector geometry,

geometry of the irradiated material, and orientation of the material. The original factor is estimated to be 30 times larger than the dose rate at 1-foot for a standard size part. This value may be changed once validation experiments are performed to provide more realistic results.

4. Empirical Approach

Before DREAD can be implemented as a reliable method for predicting dose rates from an experiment given its geometry and material make up and the irradiation spectrum to which it will be exposed, it must be validated by comparing dose rate predictions for a large variety of experiments and spectrum combinations with the experimentally measured dose rates. Since it is desired to obtain dose rates on-contact and at 30 cm (one foot), and since the dose rates by definition are not known, or at least not known well, strict experimental procedures must be established and followed to ensure that ALARA is practiced and that the experimenter does not exceed the Sandia National Laboratory dose limit of 250 mrem/yr.

Performing potentially hazardous work at ACRR requires multiple levels of paperwork and approvals which are required per corporate policy and DOE orders. Two documents were required to be written prior to performing the dose rate measurements: a facility work plan (FWP) and an engineered job safety analysis (EJSA). The facility work plan describes the work taking place and ensures that it is not outside of the operating envelope described in the facility's documented safety analysis. It is reviewed and approved by the manager of the facility as well as the facility supervisor. The engineered job safety analysis describes the steps that take place to perform the work and a hazard analysis of each step is documented as well as how to control each hazard. It is reviewed and approved by the facility supervisor. The FWP and EJSA are performed under a radiological technical work document (RTWD) which describes the

Personal Protective Equipment hazards, hold and void points, acceptable dose rates, and various other details when handling experiments removed from the ACRR central cavity. The RTWD is reviewed and approved by a radiation protection department engineer, the facility supervisor, and the manager for the facility.

The description and steps in the EJSA and FWP are fairly simple: follow the



Figure 15 - Teletector

guidance of the RTWD and use an appropriate gamma/beta meter to measure the dose rates of experiments that have been irradiated.

The detectors used for measuring dose rates of the experiments were the Thermo Eberline RO20 (Figure 15), the Teletector (Figure 16). The RO20 is calibrated to a Cs-137 source which emits gamma radiation at 662 keV energy. It is an air-filled ion chamber used for portable radiation measurement. The display can read from 0 to 5 mrem/hr and the adjustment dial can support readings up to 50 rem/hr. The



Figure 16 - Thermo Eberline RO20

response is accurate within 15% (+/-0.5 mrem) for photons from 33 keV to 1.3 MeV whether the user is taking reading through the window or through the sides of the instrument within about 5 seconds. The accuracy of the RO20 makes it the preferred instrument to use for accurate dose rate measurements. However, it requires the user to have his/her hand within about 6 inches of the part being measured to retrieve an on-contact dose rate. The Teletector has the advantage of allowing the user to distance himself/herself from the irradiated part by up to 15 additional feet. The teletector uses a ZP1300 gamma tube, which is a small sized Geiger counter tube that is approximately +/- 40% accurate. The small size prevents the detector from becoming saturated at higher dose rates. This detector is typically used when irradiated experiments are removed from the central cavity and when determining dose rates with the RO20 cannot safely be performed. The dose rate range on the teletector is from 10 μ R/h to 1000 R/h and is digitally displayed on the instrument.

Worker safety is always prioritized ahead of dose-rate measurements, so on-contact reading measurements and 1-foot measurements are performed rapidly and the distances are eyed using the skill of the worker rather than a ruler or yardstick. The teletector is used for experiments exiting the cavity to obtain dose rates and ensure that minimum dose is received by the individual taking the measurement. Once the dose rate from an irradiated part is determined to be at an acceptable level (either initially upon removal or after sufficient decay), the

RO20 is used to obtain more accurate readings at measured distances for longer periods of time.

Prior to the completion of DREAD, dose rates were collected on numerous samples but only on-contact measurements were taken. The samples were measured using multiple types of meters. Once DREAD was completed it was realized that the data from these samples was not consistent and could not be extrapolated to other distances with any accuracy or consistency. To address this issue, measurements were taken at both one foot and on-contact to provide a measurement that should be more successfully predicted (the one foot measurement) and to determine a relationship between the on-contact and one foot measurements that can be used as a calibration factor in DREAD (Figures 17&18).



Figure 17 - Example of On-Contact dose rate measurement



Figure 18 - Example of 1-foot dose rate measurement

The process for collecting data requires several factors to be recorded for

completeness. These factors are: the spectrum used, the time of irradiation or pulse mode, the time after irradiation before a measurement was conducted, the measured radiation at 1-foot and on-contact, and any notes about the objects size/ composition that would affect dose rate readings. The more data obtained from each type of spectrum-modified experiment will result in better correction factors in DREAD. However, there is a tradeoff, high-dose rates and more samples results in more dose to the worker collecting the dose-rate information. Optimizing the number and length of the measurements is important for ALARA reasons. The maximum total effective dose for obtaining measurements for this project was 100 mrem which was determined by the operations staff. To stay below this limit, a maximum of 40 samples was proposed, using only the teletector to measure parts over 500 mrem/hr on-contact dose rates. The length of the measurements was determined by how quickly the detector readings stabilized. The Teletector is very quick to respond and gives nearly instantaneous readings. It also has a peak hold function which makes determining the highest dose rate measured very easy to determine. The RO-20 readings take up to 5 seconds to reach a final stable reading and requires the user to select the appropriate range on the detector for the dose rate measurement.

Another means that can be used to validate the accuracy of DREAD is to make a “curie measurement.” Here small samples of pure materials are irradiated and the absolute activity is measured for comparison to the activity (curie) prediction

by DREAD. The absolute activities were determined by measuring small samples with a portable gamma spectrometer. Because there was no gamma ray spectrometer available locally to make these measurements, they were made by a separate organization at SNL, which limited the number of measurements that could be made. Additional such measurements can be done in the future and added to the data base in DREAD to improve its predictive capabilities.

5. Results

The experimental results were tabulated along with the results from DREAD and are summarized in Tables II through VI. Many different types of experiments were weighed and the materials were



Figure 19 - Dosimetry pack (left) Microchip (right)

characterized to accurately model activation in DREAD. These experiments ranged from dosimetry of a single electronics

chip (Figure 19) to



Figure 20 - Example of more complex electronics assembly

complicated setups containing a wide variety of electronics, cables, housings, and dosimetry (Figure 20). The dose rates from some experiments were measured, but since the material composition was unknown or is still

undergoing determination the results could not yet be included in DREAD. Any future experiment with known composition and dose rates can later be added to the summary tables and factors in DREAD can be adjusted as necessary.

The tables for all of the experiments show what spectrum was used, whether it was a pulse or, if steady-state, the run time, the elapsed time before the experiment was measured, the on-contact and 1-foot measurements, and the DREAD predictions. The data for every spectrum provide similar results. The longer wait times result in the DREAD 1-foot over estimation converging on the actual measured dose rate or, in some cases, falling below the measured dose rate. Table VI represents the average percent difference between the 1-foot measurements from DREAD and actual values and the average ratio between actual on-contact and 1-foot measurements.

The free field spectrum results are presented in Table II. All of the data for this spectrum were collected in the pulse mode. Only one sample's wait time was over night, but the difference between the actual and DREAD dose rates is very small compared with the shorter wait times. The average 1-foot to on-contact ratio and the difference between the on-contact and 1-foot, free-field measurements were in the middle for the four spectra.

Table II - Free Field Results of DREAD vs. Actual Dose Rates

#	Bucket	Pulse/Run Time	Wait Time	On-Contact mrem/hr	1-foot mrem/hr	DREAD on-contact	DREAD 1 ft
1	Free Field	Pulse	31 min	1300	60	2000	67
2	Free Field	Pulse	48 min	1500	65	2100	70
3	Free Field	Pulse	104 min	1300	48	1600	53
4	Free Field	Pulse	19 hr	130	5	180	6
5	Free Field	Pulse	35 min	2100	90	3056	101
6	Free Field	Pulse	90 min	1500	70	2376	79
7	Free Field	Pulse	18 hr	800	50	1536	52
8	Free Field	Pulse	40 min	600	25	951	31
9	Free Field	Pulse	45 min	500	22	837	27
10	Free Field	Pulse	20 min	2000	100	4048	135

The lead boron bucket was used for a variety of electronic parts for short, steady-state runs with rapid experiment recovery from the central cavity. Table III shows the results of these operations. Several of the sample points were also collected for pulse mode operations. These results are very similar to those of the free field pulse results. Sample #8 has a large difference between the on-contact reading and the DREAD prediction. This was a larger piece of electronics and the measurements were all measured from the edge of the experiment parts, so the distance from the most radioactive parts may have been farther than the DREAD model. The lead boron bucket average 1-foot measurement had the largest discrepancy from the DREAD model at 22% higher predictions from DREAD. This over prediction may be due to larger experiments and the method for collecting the data for measurements at 1-foot. The difference is not large enough to warrant changes to the DREAD code.

Table III - Lead Boron Bucket Results of DREAD vs. Actual Dose Rates

#	Bucket	Pulse/Run Time	Wait Time	On-Contact mrem/hr	1-foot mrem/hr	DREAD on-contact	DREAD 1 ft
1	LB 44"	~7 min	30 s	14	-	3.2	-
2	LB 44"	~14 min	60 s	2.75	-	3.4	-
3	LB 44"	~6 min	60 s	9	0.5	16.2	0.523
4	LB 44"	~6 min	30 s	14	1.1	34	1.197
5	LB 44"	~6 min	5 min	700	20	805	27.9
6	LB 44"	~6 min	5 min	1100	50	1784	59.4
7	LB 44"	~6 min	20 min	250	9	440	14
8	LB 44"	~6 min	2 hr	60	5.1	203	6.7
9	LB 44"	Pulse	45 min	160	8	273	9.1
10	LB 44"	Pulse	1 hr	800	28	1011	33
11	LB 44"	Pulse	2 hr	170	7	225	7.5

The poly lead graphite bucket (Table IV) and lead poly bucket (Table V) displayed similar results to each other and a similar trend to the free field and lead boron bucket. These spectra are not used as frequently as the lead boron bucket or free field spectrums. The buckets average between 5 to 10 times as many thermal neutrons per megawatt of power on the experiment. Since most activation is due to thermal neutrons, the experiments were activated significantly and resulted in higher dose rates. All of the measurements were taken quickly so the experiment could be stored in a shielded cell immediately and reduce exposure to personnel involved in removing the experiment packages from the central cavity. A sheet of paper was used as a reference for taking a 1-foot measurement. The center of the teletector was located at 12 inches from the edge of the experiment and the edge of the detector was on the 11-inch side of the paper. The total dose received taking measurements was less than 10 mrem, well below the administrative limit of 100 mrem.

Table IV - Poly Lead Graphite Bucket Results of DREAD vs. Actual Dose Rates

#	Bucket	Pulse/Run Time	Wait Time	On-Contact mrem/hr	1-foot mrem/hr	DREAD on-contact	DREAD 1 ft
1	PLG	Pulse	3 hr	6500	250	8424	280
2	PLG	Pulse	3.5 hr	6800	250	9154	305
3	PLG	Pulse	3 hr	4000	140	5148	171
4	PLG	12 min	2 hr	5800	225	7930	264
5	PLG	18 min	2 hr	10000	415	14243	474

Table V - Lead Poly Bucket Results of DREAD vs. Actual Dose Rates

#	Bucket	Pulse/Run Time	Wait Time	On-Contact mrem/hr	1-foot mrem/hr	DREAD on-contact	DREAD 1 ft
1	LP	Pulse	2 hr	5500	278	7100	237
2	LP	Pulse	3 hr	9000	500	12698	423
3	LP	Pulse	1.5 hr	5000	260	7260	242
4	LP	Pulse	5	14000	720	19426	647
5	LP	30 min	20 hr	5500	220	7321	244

Table VI - Comparison of DREAD vs. Actual results and factors between 1-foot and on-contact

Spectrum	Average % DREAD vs. Actual	Average Factor between 1-foot and on-contact
Free Field	16% higher	23
Lead Boron	22% higher	22
Poly Lead Graphite	18% higher	26
Lead Poly	12% higher	20
Average	17%	22.75

The average over estimation at 1-foot was calculated as well as the average factor for actual dose rates measured by the instruments at 1-foot and on-contact as shown in Table VI. The 17% average over-estimation of DREAD is within the error range for the teletector, which was the most frequently used instrument for the higher dose rates. The lower dose rates obtained with the RO20 were taken

after longer decay times and were closer to the DREAD predictions. The over estimations are desired to provide conservative results to allow for safe experiment handling. The factor of 22.75 difference between the on-contact and 1-foot measurements could be entered into DREAD to replace the factor of 30, but it is only about a 32% over estimation if left alone, which would continue to ensure conservative estimations continue and prevent operators from exceeding administrative dose rate limits.

DREAD was also used to generate decay trends for different materials and the dose rates of each material at 1-foot at different times. These graphs (Figures 21 to 23) are useful to experimenters and operators to help determine materials to use for experiment support and, more importantly, what not to use. This was done by entering 1 gram of each material separately at different wait times for a 10 MJ pulse.

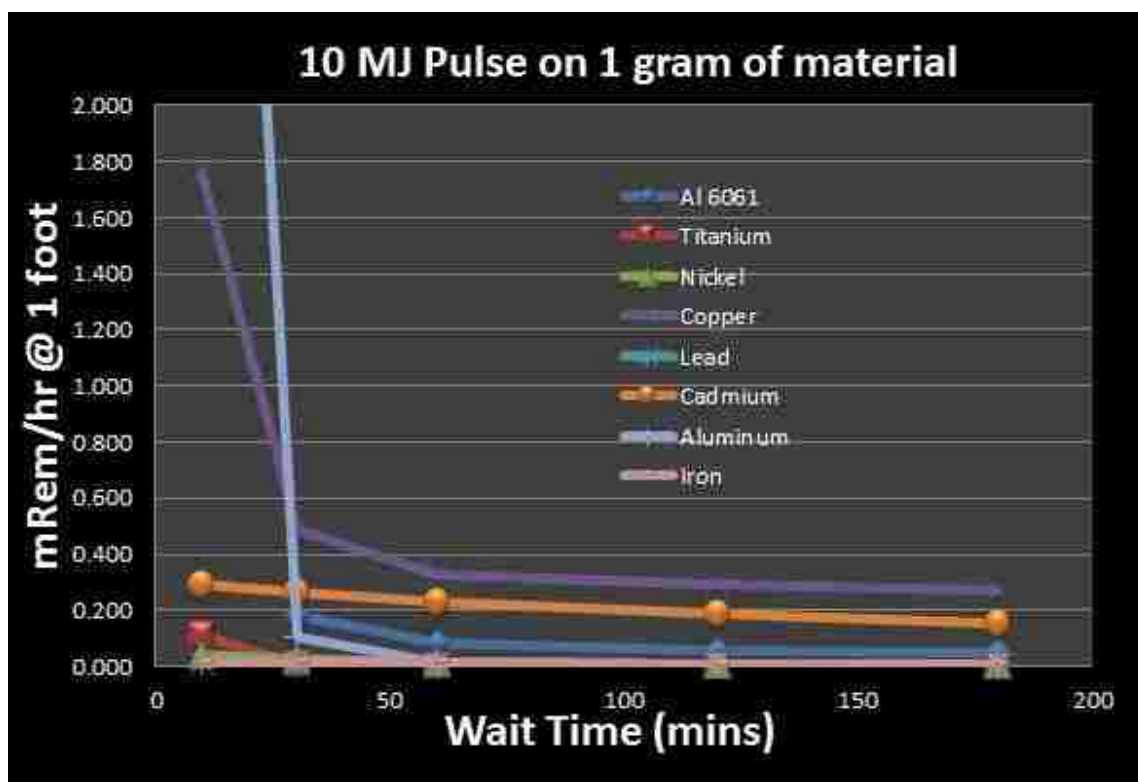


Figure 21 - 10 MJ pulse on 1 gram of commonly used metals

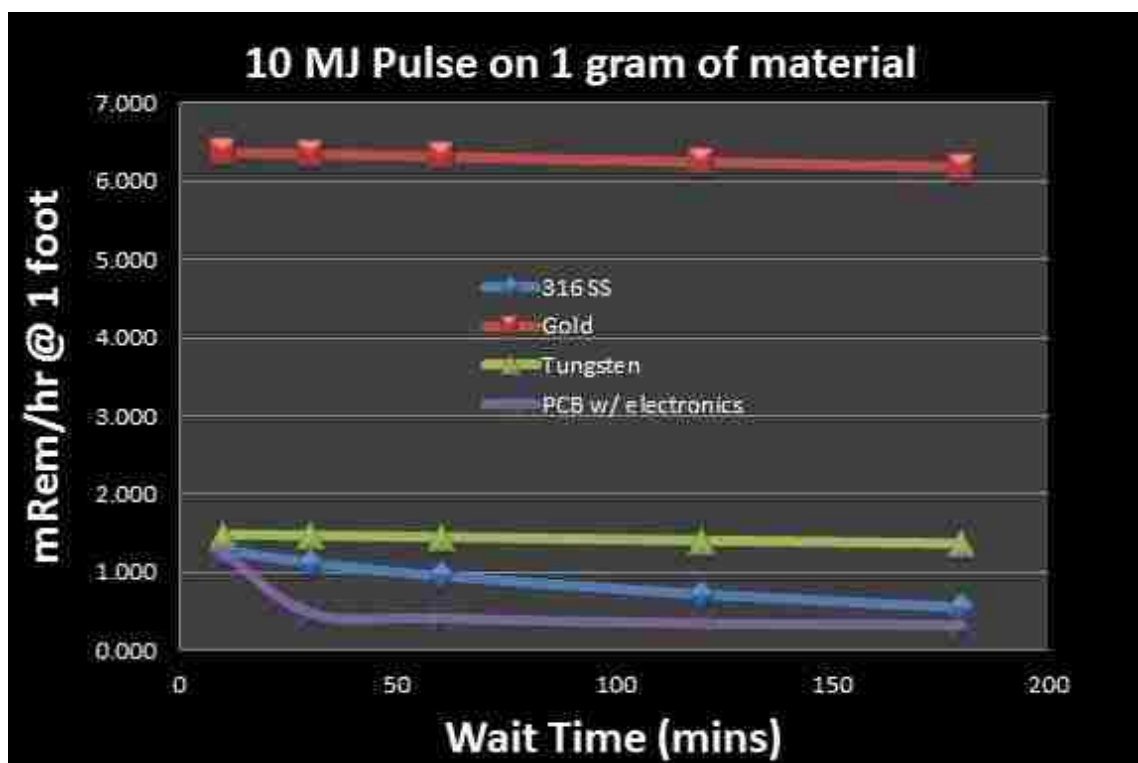


Figure 22 - 10 MJ pulse on metals with higher activation and dose rates

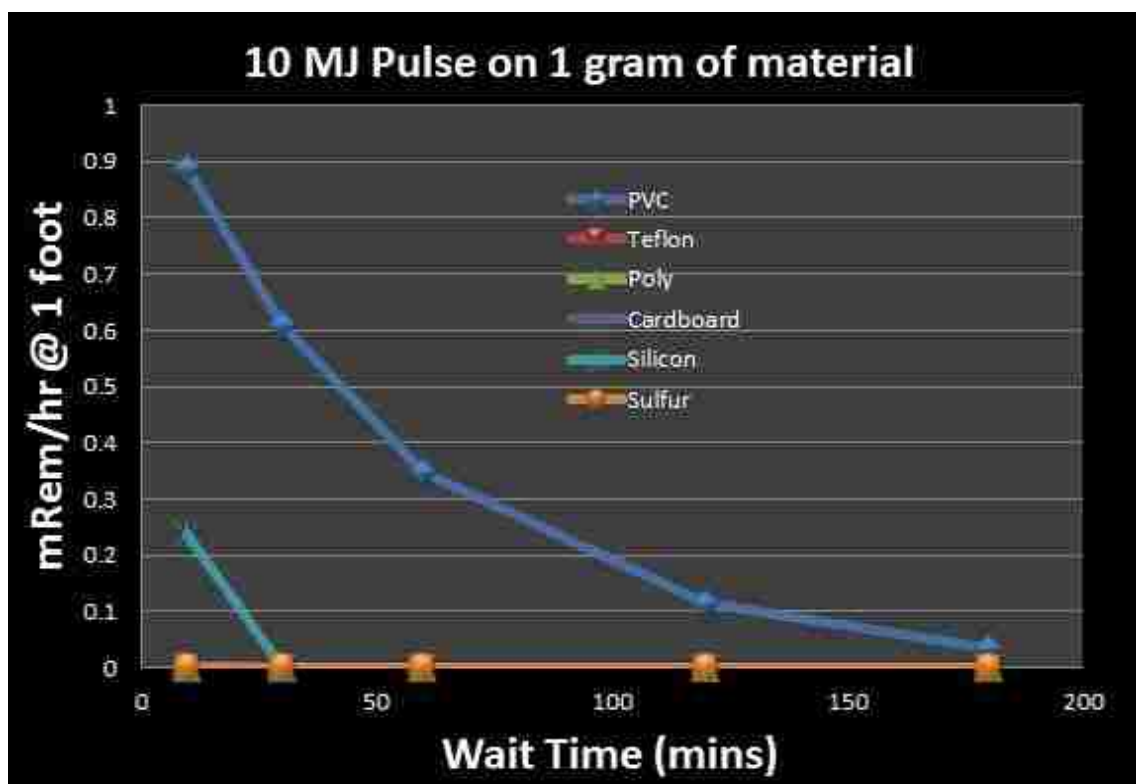


Figure 23 - 10 MJ pulse on 1 gram of common non-metals

DREAD was also used to predict curie amounts for medical isotope generation. The reactor was run to activate a small sample of copper-63 beads and create copper-64. The tables_by_major curie estimate was within 10% of the experimentally measured curie content obtained from a gamma-ray spectroscopy measurement. A request has been made to run more variations of the experiment and to estimate a larger variety of isotopes by the University of New Mexico medical school.

6. Conclusion

Longer waits resulted in the actual dose rate approaching the estimations of DREAD. The convergence could be due to long lived activated impurities that do not contribute much for the short wait times, but start to contribute for long wait times once the shorter lived known materials entered into DREAD begin to decay away. The higher dose rates were also measured with the teletector which does not scale with increasing energy and has a larger error band. The error band for this detector spans the results from DREAD. The error band of the RO20 combined with instrument calibration allowable errors and detector variances also is within the tolerance for the DREAD results.

The estimates that DREAD calculates are conservative, consistent, and accurate. An average estimate of 17% above the actual dose rates measured provides operators and experiments with a very good approximation of what they can expect to handle when removing experiments from the central cavity. The tool is installed on the computers for the reactor operators and is currently being utilized to effectively estimate the activation of smaller electronic parts and assemblies. As the program evolves and expands, its usefulness for operators and experimenters will do the same.

The DREAD code could be changed to calculate on-contact readings based on what spectrum is chosen and adjust to the averages for each bucket. However, the small amount of additional conservatism would likely be valued over a lower than actual estimation.

Future Work

There are several constraints inherent in DREAD that should be examined for the next revision. The size of the experiments is limited and DREAD does not perform well for larger experiments that are outside of the center flux originally tallied in MCNP. DREAD currently cannot give estimates of the dose rates expected from the spectrum modifier buckets, which are the first to exit the cavity. MCNP runs with tallies of the entire bucket with different ratios for different heights inside of the central cavity/bucket could help correct this issue. New buckets (water bucket, cd poly bucket. etc.) should be added as they are made available. A spectrum could be added to the code would that allow for dose rate estimations in the external cavities at ACRR.

In addition to the calculation limits, other features could be added to make DREAD a more user-friendly program. Adding more commonly used materials such as coaxial cables, fiber optic cables, and connectors would aid in speed and consistency for different runs. DREAD could also be used to automatically generate graphs in Microsoft excel for the decay rates of the materials rather than numerous runs manually entered by the user.

Appendix A – MCNP Code

```

1  STANDARD ACRR Model (Extended Cavity, 32" Pedestal, Pb-B4C Bucket)
2  C
3  C  Original Model Developed by W. Fan
4  C  Modified by P. Cooper and E. Parma with new cavity
5  C  Macrobody Model Developed by R. DePriest
6  C  New 44" Pb-B4C Bucket LB44 Model Developed by T. Trinh
7  C  New information and energy groups updated for Cinder by J. Greenberg
8  C
9  C   standard 236-element core configuration with new cavity
10 C   no FREC
11 C   room temp 70c cross sections with S(a,b)
12 C   LB-44-cl-32 - 44inch lead/boron bucket on 32inch pedestal
13 C   tally is a 6cm diameter sphere at fuel centerline
14 C       89 and 640 neutron energy groups
15 C       48 gamma energy groups
16 C
17 C   control rods   : variable
18 C   safety rods   : variable
19 C   transient rods : variable
20 C
21 C
22 C   1       2       3       4       5       6       7       8
23 C 3456789012345678901234567890123456789012345678901234567890
24 C *****
25 C *           CELL CARDS           *
26 C *****
27 C
28 C
29 C Universe definitions for the standard 236-element core.
30 C
31 C   U=1:fuel rods      U=2:water rods
32 C   U=3:control rods   U=4:safety rods
33 C   U=5:transient rods U=6:nickel rods
34 C   U=7:90% fuel rods  U=9:al rods (empty)
35 C
36 C   ***** U=8 is the reactor core fill. *****
37 C
38 C
39 C Regular Fuel Elements

```

40 C
 41 10 0 -10 U=1 IMP:N,P=1 \$Void
 42 11 1 -3.3447 10 -11 U=1 IMP:N,P=1 \$UO2-BeO fuel
 43 14 0 11 -14 U=1 IMP:N,P=1 \$Void
 44 15 2 -8.4000 14 -15 U=1 IMP:N,P=1 \$Niobium
 45 16 0 15 -16 U=1 IMP:N,P=1 \$Void Gap
 46 17 4 -2.8000 -17 U=1 IMP:N,P=1 \$Lower BeO Plug
 47 18 4 -2.8000 -18 U=1 IMP:N,P=1 \$Upper BeO Plug
 48 19 3 -8.0300 17 -19 U=1 IMP:N,P=1 \$Lower SS Plug
 49 20 3 -8.0300 18 -20 U=1 IMP:N,P=1 \$Upper SS Plug
 50 21 3 -8.0300 19 20 16 -21 U=1 IMP:N,P=1 \$SS304
 51 22 5 -1.0000 21 -22 U=1 IMP:N,P=1 \$Water
 52 C
 53 C
 54 C Water Rods
 55 C
 56 23 5 -1.0000 -22 U=2 IMP:N,P=1 \$Water
 57 C
 58 C
 59 C Control Rods: Poison section
 60 C
 61 25 8 -2.4800 -25 U=3 IMP:N,P=1 \$B4C poison
 62 26 0 25 -26 U=3 IMP:N,P=1 \$Void Cap
 63 27 3 -8.0300 26 -27 U=3 IMP:N,P=1 \$Poison sleeve
 64 28 3 -8.0300 -28 U=3 IMP:N,P=1 \$Magnaform plug
 65 29 5 -1.0000 27 28 -29 U=3 IMP:N,P=1 \$Water
 66 C
 67 C Control Rods: Fuel follower
 68 C
 69 30 0 -30 U=3 IMP:N,P=1 \$Void
 70 31 1 -3.3447 30 -31 U=3 IMP:N,P=1 \$UO2-BeO fuel
 71 32 0 31 -32 U=3 IMP:N,P=1 \$Void
 72 33 2 -8.4000 32 -33 U=3 IMP:N,P=1 \$Niobium
 73 34 0 33 -34 U=3 IMP:N,P=1 \$Void gap
 74 35 4 -2.8000 -35 U=3 IMP:N,P=1 \$BeO plug
 75 36 0 -36 U=3 IMP:N,P=1 \$Void
 76 37 3 -8.0300 34 35 36 -37 U=3 IMP:N,P=1 \$SS304
 77 38 5 -1.0000 37 -38 U=3 IMP:N,P=1 \$Water
 78 C
 79 C
 80 C Safety Rods: Poison section
 81 C
 82 39 8 -2.4800 -39 U=4 IMP:N,P=1 \$B4C poison

83 40 0 39 -40 U=4 IMP:N,P=1 \$Void cap
84 41 3 -8.0300 40 -41 U=4 IMP:N,P=1 \$Poison sleeve
85 42 3 -8.0300 -42 U=4 IMP:N,P=1 \$Magnaform plug
86 43 5 -1.0000 41 42 -43 U=4 IMP:N,P=1 \$Water
87 C
88 C Safety Rods: Fuel follower
89 C
90 44 0 -44 U=4 IMP:N,P=1 \$Void
91 45 1 -3.3447 44 -45 U=4 IMP:N,P=1 \$UO2-BeO fuel
92 46 0 45 -46 U=4 IMP:N,P=1 \$Void
93 47 2 -8.4000 46 -47 U=4 IMP:N,P=1 \$Niobium
94 48 0 47 -48 U=4 IMP:N,P=1 \$Void gap
95 49 4 -2.8000 -49 U=4 IMP:N,P=1 \$BeO plug
96 50 0 -50 U=4 IMP:N,P=1 \$Void
97 51 3 -8.0300 48 49 50 -51 U=4 IMP:N,P=1 \$SS304
98 52 5 -1.0000 51 -52 U=4 IMP:N,P=1 \$Water
99 C
100 C
101 C Transient Rods: Void section
102 C
103 53 0 -53 U=5 IMP:N,P=1 \$Void
104 54 7 -2.7000 53 -54 58 60 61 U=5 IMP:N,P=1 \$Al tubing
105 55 5 -1.0000 54 -55 U=5 IMP:N,P=1 \$Water
106 56 7 -2.7000 55 -56 U=5 IMP:N,P=1 \$Al guidex
107 57 5 -1.0000 56 -57 U=5 IMP:N,P=1 \$Water
108 58 7 -2.7000 -58 U=5 IMP:N,P=1
109 C
110 C Transient Rods: Poison section
111 C
112 59 8 -2.4800 -59 U=5 IMP:N,P=1 \$Poison
113 60 7 -2.7000 59 -60 U=5 IMP:N,P=1 \$Inner sleeve
114 61 0 -61 U=5 IMP:N,P=1 \$Void
115 62 7 -2.7000 -62 54 U=5 IMP:N,P=1 \$End plug
116 C
117 C
118 C Nickel Rods
119 C
120 65 6 -8.9000 -21 U=6 IMP:N,P=1 \$Nickel
121 66 5 -1.0000 21 -22 U=6 IMP:N,P=1 \$Water
122 C
123 C
124 C 90% Fuel Element
125 C

```

126 70 0 -10 U=7 IMP:N,P=1 $Void
127 71 11 -3.0102 10 -11 U=7 IMP:N,P=1 $UO2-BeO fuel
128 74 0 11 -14 U=7 IMP:N,P=1 $Void
129 75 2 -8.4000 14 -15 U=7 IMP:N,P=1 $Niobium
130 76 0 15 -16 U=7 IMP:N,P=1 $Void Gap
131 77 4 -2.8000 -17 U=7 IMP:N,P=1 $Lower BeO Plug
132 78 4 -2.8000 -18 U=7 IMP:N,P=1 $Upper BeO Plug
133 79 3 -8.0300 17 -19 U=7 IMP:N,P=1 $Lower SS Plug
134 80 3 -8.0300 18 -20 U=7 IMP:N,P=1 $Upper SS Plug
135 81 3 -8.0300 19 20 16 -21 U=7 IMP:N,P=1 $SS304
136 82 5 -1.0000 21 -22 U=7 IMP:N,P=1 $Water
137 C
138 C
139 C Empty Aluminum Rod
140 C
141 600 0 -90 U=25 IMP:N,P=1 $Void
142 601 7 -2.7000 90 -21 U=25 IMP:N,P=1 $Al Rod
143 602 5 -1.0000 21 -22 U=25 IMP:N,P=1 $Water
144 C
145 C
146 C Empty Aluminum Rod
147 C
148 90 0 -90 U=9 IMP:N,P=1 $Void
149 91 7 -2.7000 90 -21 U=9 IMP:N,P=1 $Al Rod
150 92 5 -1.0000 21 -22 U=9 IMP:N,P=1 $Water
151 C
152 C
153 C Core (UNIVERSE = 8)
154 C
155 1 0 -300 311 210 211 213 220 fill=8 IMP:N,P=1
156 C
157 2 5 -1.0000 -320 lat=2 U=8 IMP:N,P=1
158 fill -12:12 -12:12 0:0
159 C
160 C
161 C This fuel loading reflects the board as of May 2003.
162 C
163 C 1 2 3 4 5 6 7 8
164 C 34567890123456789012345678901234567890123456789012345678901234567890
165 C
166 2 24r
167 2 24r
168 2 24r

```

```

169      2 9r 2 6 1   1 1 1 1 1 1 1 1   1 6 2 2 $ interface with frec
170      2 8r 6 6 1 1   1 1 1 1 1 1 1   1 1 6 6 2
171      2 7r 6 1 1 1 1   1 1 1 1 1 1 1   1 1 1 1 6 2
172      2 6r 6 7 1 1 1 3 1 1 5 1 1 3 1 1 1 7 6 2
173      2 5r 6 7 1 1 1 1 1 1 1 1 1 1 1 1 1 7 6 2
174      2 4r 6 7 1 1 1 1 1 1 1 1 1 1 1 1 1 7 6 2
175      2 3r 9 6 1 1 1 4 1 1 2 2 2 2 1 1 1 1 1 1 6 2 2
176      2 2r 2 9 6 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1 6 6 2 2
177      2 2 2 6 6 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 6 2 2 2
178      2 2 2 6 1 1 3 1 1 2 2 2 2 2 2 2 1 1 3 1 1 6 2 2 2 $ center line
179      2 2 6 6 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 9 6 2 2 2
180      2 2 6 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1 1 9 2 2 2r
181      2 6 1 1 1 1 5 1 1 2 2 2 2 1 1 5 1 1 1 1 2 2 3r
182      2 6 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 4r
183      2 6 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 5r
184      2 6 1 1 1 1 3 1 1 4 1 1 3 1 1 1 1 1 2 2 6r
185      2 6 7 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 7r
186      2 6 6 6 1 1 1 1 1 1 1 1 1 9 9 2 2 8r
187      2 2 9 6 6 6 1 1 1 1 6 6 6 6 2 2 9r
188      2 2 2 9 2 6 7 7 7 6 2 6 2 2 2 10r
189      2 2 2 2 25 6 6 6 6 9 2 2 2 2 11r
190      2 24r
191  C
192  C
193  C ***** END OF UNIVERSE DEFINITIONS AND CORE FILL *****
194  C
195  C
196  C NEW CENTRAL CAVITY
197  C
198  C To add 32-in pedestal, remove C from line 2.
199  C To add 8-in pedestal, remove C from line 2 and 3.
200  C You must also remove the C's from the cells in the pedestal
201  C descriptions (Cells 110-116).
202  C
203  C Use Line 4 of Cell 100 to exclude surface of buckets and experiments.
204  C Exclude surface 706 for Pb-B4C; Exclude surface 711 for Al dosimetry bucket;
205  C Exclude surface 725 for LP-1
206  C Exclude surfaces 730, 731, and 734 for Boom Box
207  C
208  100 702 -1.0245e-3 -100
209  110
210  C      113 114 116
211      1001      IMP:N,P=1 $Void

```

212	C	for LB44	899	IMP:N,P=1 \$Void
213	101	3 -8.0300	100 -101	IMP:N,P=1 \$Stainless liner
214	102	7 -2.7000	-311 101 -102	IMP:N,P=1 \$Aluminum
215	103	5 -1.0000	-311 102	IMP:N,P=1 \$Water
216	C			
217	C			
218	C	Central Cavity Additions (32" and 8" Pedestals)		
219	C			
220	C	32-in pedestal		
221	C			
222	110	7 -2.7000	-110 111 112	IMP:N,P=1 \$32-in pedestal
223	111	702 -1.0245e-3	-111	IMP:N,P=1 \$32-in pedestal inset
224	112	702 -1.0245e-3	-112	IMP:N,P=1 \$Inset Notch
225	C			
226	C			
227	C	8-in pedestal		
228	C			
229	C	113 7 -2.7000	-113	IMP:N,P=1 \$Bottom plate
230	C	114 7 -2.7000	-114	IMP:N,P=1 \$Top plate
231	C	115 702 -1.0245e-3	-115	IMP:N,P=1 \$Center Void
232	C	116 7 -2.7000	-116 115	IMP:N,P=1 \$Support Tube
233	C			
234	C			
235	C	End of Central Cavity Additions		
236	C			
237	C			
238	C	Top and Bottom Grid Plates		
239	C			
240	200	7 -2.7000	-200 311 201	IMP:N,P=1 \$Top plate
241	201	5 -1.0000	-200 220 -201	IMP:N,P=1 \$Water
242	202	7 -2.7000	-202 311	IMP:N,P=1 \$Bottom plate
243	C			
244	C			
245	C	Nickel Plate and Window to the Radiography Lab		
246	C			
247	210	6 -8.9000	-210	IMP:N,P=1 \$Nickel Plate
248	211	5 -1.0000	-211 210 -900	IMP:N,P=1 \$Water
249	212	0	-212 -900	IMP:N,P=1 \$Void
250	213	7 -2.7000	-213 212 -900	IMP:N,P=1 \$Aluminum
251	C			
252	C			
253	C	FREC-II Side Ni Plate		
254	C			

255 220 6 -8.9000 -220 IMP:N,P=1
 256 C
 257 C
 258 C Surrounding Water
 259 C
 260 230 5 -1.0000 -900 220 213 212 211 202 200
 261 300 311 IMP:N,P=1
 262 C
 263 C
 264 C
 265 C EXPERIMENTAL or SPECTRUM MODIFYING BUCKETS (700's)
 266 C
 267 C Pb-B4C Bucket (700-706)
 268 C Weight of Bucket per L. Martin (8/21/2003) - 446 lbs
 269 C Weight of Model Bucket - 450.81
 270 C Density of B4C layer changed to 2.12 g/cc to make weight 446.19 lbs
 271 C
 272 C
 273 C 700 702 -1.0245e-3 -700 IMP:N,P=1 \$Inside Bucket
 274 C 701 7 -2.7100 -701 700 IMP:N,P=1 \$1/16" Al liner
 275 C 702 700 -11.350 -702 701 7091 7092 IMP:N,P=1 \$1" Pb on bottom
 276 C 703 701 -2.5300 -703 7091 7092 IMP:N,P=1 \$Boral on bottom
 277 C 707 0 -707 702 IMP:N,P=1 \$Slop between Cannister and Pb
 278 C 708 8 -2.1200 -708 IMP:N,P=1 \$B4C layer on the bottom
 279 C
 280 C 704 7 -2.7100 -704 703 707 708
 281 C 7091 7092 IMP:N,P=1 \$Al layer
 282 C 705 8 -2.1200 -705 704 IMP:N,P=1 \$B4C layer
 283 C 706 7 -2.7100 -706 705 7091 7092 IMP:N,P=1 \$Al exterior
 284 C 7091 3 -8.0300 -7091 IMP:N,P=1 \$Dowel 1
 285 C 7092 3 -8.0300 -7092 IMP:N,P=1 \$Dowel 2
 286 C
 287 C
 288 C
 289 C
 290 C Standard Aluminum Experiment Bucket (710-711)
 291 C Add -900 to 710 and 711 if using 24" Bucket
 292 C
 293 C 710 702 -1.0245e-3 -710 IMP:N,P=1 \$Inside Bucket
 294 C 711 7 -2.7000 -711 710 IMP:N,P=1 \$Aluminum Bucket
 295 C
 296 C
 297 C

298 C
 299 C Pb-Poly Bucket (720-725) -- Designated as LP-1
 300 C
 301 C 720 702 -1.0245e-3 -720 IMP:N,P=1 \$Bottom of Inside
 302 C 721 7 -2.7000 -721 720 726 IMP:N,P=1 \$1/16" Al Liner
 303 C 722 700 -11.350 -722 721 724 726 IMP:N,P=1 \$0.4" Pb Layer
 304 C 723 704 -0.9450 -723 722 726 IMP:N,P=1 \$0.8" HDPE
 305 C 724 704 -0.9450 -724 IMP:N,P=1 \$HDPE fill-in
 306 C 725 7 -2.7000 -725 721 723 726 IMP:N,P=1 \$Al Container
 307 C 726 702 -1.0245e-3 -726 IMP:N,P=1 \$Top of Inside
 308 C
 309 C
 310 C
 311 C
 312 C
 313 C Boombox for NG testing (730-738)
 314 C
 315 C 730 765 -7.28 -730 736 737 738 IMP:N,P=1 \$Lower Boom Box
 316 C 731 765 -7.28 -731 733 IMP:N,P=1 \$Upper part of clamping ring
 317 C 732 765 -7.28 -732 733 IMP:N,P=1 \$Lower part of clamping ring
 318 C 733 702 -1.0245e-3 -733 IMP:N,P=1 \$"Void" in clamping ring
 319 C 734 702 -1.0245e-3 -734 732 IMP:N,P=1 \$"Void" at ring lip
 320 C 735 765 -7.28 -735 IMP:N,P=1 \$Plug
 321 C 736 702 -1.0245e-3 -736 735 IMP:N,P=1 \$"Void" around the plug
 322 C 737 702 -1.0245e-3 -737 IMP:N,P=1 \$Lower "void"
 323 C 738 702 -1.0245e-3 -738 IMP:N,P=1 \$Lip "void"
 324 C
 325 C
 326 C
 327 C
 328 C
 329 C
 330 C New 44" Pb-B4C Bucket
 331 C Base Plate w/ B4C Volume
 332 C From Ktech drawing labeled "PbB BASEII"
 333 C 800 8 -1.274704138 (-802):(817 -815):(818 -816) IMP:N,P=1 \$ B4C Cavity
 334 C 801 766 -7.83 -803:-809 IMP:N,P=1 \$ All-threads
 335 C From McMaster-Carr catalog, Item # 98914A033, Threaded Rods and Studs, General Purpose Steel
 336 C 802 767 -7.82 (805 -806):(811 -812) IMP:N,P=1 \$ Washers
 337 C From McMaster-Carr catalog, Item #94744A285, Zinc-Plated Steel Washer for Soft Materials (Type
 338 C 803 767 -7.82 (803 -807):(809 -813) IMP:N,P=1 \$ All-thread nuts
 339 C From McMaster-Carr catalog, Item # 93939A823, Hex Nut, Grade 8 Steel
 340 C 804 702 -1.0245e-3 (803 -804):(809 -810):(803 -805):(809 -811):

341 C (806 807 803 -808):(812 813 809 -814) IMP:N,P=1 \$ Void inside this thread, washe
 342 C 805 7 -2.704 (-817 819):(-818 820) IMP:N,P=1 \$ Al6061 plugs
 343 C From McMaster-Carr catalog, Item # 44705K334, Low-Pressure Aluminum Threaded Square-Socket Plug
 344 C 806 702 -1.0245e-3 (-819):(-820) IMP:N,P=1 \$ Void in this area
 345 C 807 7 -2.704 (-800:-801) 802 804 808 810 814 815 816
 346 C IMP:N,P=1 \$ Al6061 Base Plate
 347 C Containment Base
 348 C From Ktech drawing labeled "CONTAINMENT BASE II"
 349 C 830 702 -1.0245e-3 801 -830 IMP:N,P=1 \$ Void
 350 C 831 7 -2.704 (830 833 834 -831):(831 -832) IMP:N,P=1 \$ Al6061 Containment Base
 351 C 832 766 -7.83 -833:-834 IMP:N,P=1 \$ All-threads, General Purpose St
 352 C Lead Material
 353 C Unchamfered lead rings
 354 C 840 702 -1.0245e-3 -853 IMP:N,P=1 \$ Void inside lead rings
 355 C 841 766 -7.83 -847:-848 IMP:N,P=1 \$ All-threads, General Purpose St
 356 C 842 702 -1.0245e-3 (847 -843):(848 -844) IMP:N,P=1 \$ Void between All-threads and Pb
 357 C 843 702 -1.0245e-3 -849:-851 IMP:N,P=1 \$ Void inside Al6061 tubing
 358 C 844 7 -2.704 (849 -850):(851 -852) IMP:N,P=1 \$ Al6061 tubing
 359 C 845 702 -1.0245e-3 (850 -845):(852 -846) IMP:N,P=1 \$ Void between tubing and Pb hole
 360 C 846 700 -11.35 (862 863 860 843 844 845 846 -842):(-840 843 844 845 846):
 361 C (853 -854 843 844 845 846) IMP:N,P=1 \$ Unchamfered lead rings
 362 C Inner Aluminum 6061 Sleeves (Items #14 and 15 in DWG titled "LEAD BORON BUCKET ASSEMBLY II")
 363 C 860 7 -2.704 -860 IMP:N,P=1 \$ Al6061 bottom plate
 364 C 861 702 -1.0245e-3 (-861 1001):-863 IMP:N,P=1 \$ Aluminum sleeve void
 365 C 862 7 -2.704 861 -862 IMP:N,P=1 \$ Al6061 sleeve
 366 C Al6061 Double Wall Weldment (Item #9 in DWG titled "LEAD BORON BUCKET ASSEMBLY II")
 367 C 870 702 -1.0245e-3 (840 842 831 854 -868):
 368 C (840 842 831 854 -870):
 369 C (840 842 831 854 -872) IMP:N,P=1 \$ Void Between Pb Ring and Double
 370 C 871 7 -2.704 (868 -869):(870 -871):(872 -873):
 371 C (874 -875):(876 -877):(878 -879) IMP:N,P=1 \$ Al6061 Inner and Outer Skins
 372 C 872 8 -1.449249072 (869 -874):(871 -885):(886 -876):
 373 C (873 -883):(-878 884) IMP:N,P=1 \$ B4C Powder
 374 C PbB Top: Top Plate (Item #1 in DWG titled "LEAD BORON BUCKET ASSEMBLY II")
 375 C 880 702 -1.0245e-3 -880:-881:(-887 848):(-888 847):
 376 C -891:-892:-893:-894 IMP:N,P=1 \$ Voids
 377 C 881 7 -2.704 (880 881 -882 887 888 889 890 891 892 893 894):
 378 C (883 -884):(885 -886) IMP:N,P=1 \$ Top Plate
 379 C Modified Hex Head Plugs, 1/4 NPT, AL6061-T6 (Item #16 in DWG titled "LEAD BORON BUCKET ASSEMBLY")
 380 C From McMaster-Carr catalog, Item # 3867T65, High-Pressure Aluminum Pipe Fitting
 381 C 882 7 -2.704 (-821 852):(-822 852):(-823 850):(-824 850)
 382 C IMP:N,P=1 \$ Modified Hex Head Plugs
 383 C Outside World

```

384 C 889 702 -1.0245e-3 -899 800 831 832 875 877 879 821 822 823 824
385 C      882 847 848 850 852      IMP:N,P=1 $ Enclosing surface
386 C
387 C  EXPERIMENT PACKAGES (1000's)
388 C 1001 702 -1.0245e-3 -1001 1002      IMP:N,P=1 $ 6 cm dia scoring sphere
389 C 1002 6 -8.902 -1002      IMP:N,P=1 $ Ni Foil
390 1001 702 -1.0245e-3 -1001      IMP:N,P=1 $ 6 cm dia scoring sphere
391 C
392 C
393 C
394 C
395 C  EXTERNAL WORLD
396 C
397 C
398 C
399 900 0      900      IMP:N,P=0 $Outside world
400
401 C  1      2      3      4      5      6      7      8
402 C 34567890123456789012345678901234567890123456789012345678901234567890
403 C *****
404 C *      SURFACE CARDS      *
405 C *****
406 C
407 C  Fuel Elements
408 C
409 10 RCC 0.000 0.000 23.32 0.000 0.000 52.25 0.2413 $Void
410 11 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.6840 $Fuel
411 14 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.72025 $Void
412 15 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.77125 $Niobium
413 16 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.82225 $Void gap
414 17 RCC 0.000 0.000 21.415 0.000 0.000 1.905 1.48700 $Lower plug
415 18 RCC 0.000 0.000 75.57 0.000 0.000 1.905 1.48700 $Upper plug
416 19 RCC 0.000 0.000 16.32 0.000 0.000 7.000 1.82225 $Lower plug
417 20 RCC 0.000 0.000 75.57 0.000 0.000 5.000 1.82225 $Upper plug
418 21 RCC 0.000 0.000 16.32 0.000 0.000 98.89 1.87325 $
419 22 RCC 0.000 0.000 16.32 0.000 0.000 98.89 5.00000 $Water
420 C
421 C  Control Rods
422 C
423 25 3 RCC 0.000 0.000 78.11 0.000 0.000 52.25 1.46050 $B4C poison
424 26 3 RCC 0.000 0.000 78.11 0.000 0.000 98.89 1.50495 $Void cap
425 27 3 RCC 0.000 0.000 78.11 0.000 0.000 98.89 1.74625 $poison sleeve
426 28 3 RCC 0.000 0.000 75.57 0.000 0.000 2.54 1.74625 $Magnaform plug

```

427	29 3 RCC 0.000 0.000 75.57 0.000 0.000 101.43 5.00000 \$Water
428	30 3 RCC 0.000 0.000 23.32 0.000 0.000 52.25 0.24130 \$Void
429	31 3 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.68400 \$Fuel
430	32 3 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.72025 \$Void
431	33 3 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.77125 \$Niobium
432	34 3 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.82225 \$Void gap
433	35 3 RCC 0.000 0.000 20.78 0.000 0.000 2.54 1.82225 \$BeO plug
434	36 3 RCC 0.000 0.000 -79.22 0.000 0.000 100.00 1.82225 \$Void
435	37 3 RCC 0.000 0.000 -79.22 0.000 0.000 154.79 1.87325 \$SS304
436	38 3 RCC 0.000 0.000 -79.22 0.000 0.000 154.79 5.00000 \$Water
437	C
438	C Safety Rods
439	C
440	39 4 RCC 0.000 0.000 78.11 0.000 0.000 52.25 0.57150 \$B4C poison
441	40 4 RCC 0.000 0.000 78.11 0.000 0.000 98.89 0.83185 \$Void cap
442	41 4 RCC 0.000 0.000 78.11 0.000 0.000 98.89 1.74625 \$Poison sleeve
443	42 4 RCC 0.000 0.000 75.57 0.000 0.000 2.54 1.74625 \$Magnaform plug
444	43 4 RCC 0.000 0.000 75.57 0.000 0.000 101.43 5.00000 \$Water
445	44 4 RCC 0.000 0.000 23.32 0.000 0.000 52.25 0.24130 \$Void
446	45 4 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.68400 \$Fuel
447	46 4 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.72025 \$Void
448	47 4 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.77125 \$Niobium
449	48 4 RCC 0.000 0.000 23.32 0.000 0.000 52.25 1.82225 \$Void gap
450	49 4 RCC 0.000 0.000 20.78 0.000 0.000 2.54 1.82225 \$BeO plug
451	50 4 RCC 0.000 0.000 -79.22 0.000 0.000 100.00 1.82225 \$Void
452	51 4 RCC 0.000 0.000 -79.22 0.000 0.000 154.79 1.87325 \$SS304
453	52 4 RCC 0.000 0.000 -79.22 0.000 0.000 154.79 5.00000 \$Water
454	C
455	C Transient Rods
456	C
457	53 5 RCC 0.000 0.0 -76.2762 0.000 0.0 73.1012 1.20000 \$Void
458	54 RCC 0.000 0.000 -79.22 0.000 0.000 200.00 1.27000 \$Al tubing
459	55 RCC 0.000 0.000 -79.22 0.000 0.000 200.00 1.49860 \$Water
460	56 RCC 0.000 0.000 -79.22 0.000 0.000 200.00 2.02438 \$Al guidex
461	57 RCC 0.000 0.000 -79.22 0.000 0.000 200.00 5.00000 \$Water
462	58 5 RCC 0.000 0.000 -3.175 0.000 0.000 3.174 1.20000
463	59 5 RCC 0.000 0.000 -0.001 0.000 0.000 76.201 0.88000 \$Poison
464	60 5 RCC 0.000 0.000 -0.001 0.000 0.000 76.201 1.20000 \$Inner sleeve
465	61 5 RCC 0.000 0.000 76.20 0.000 0.000 123.80 1.20000 \$Void
466	62 5 RCC 0.000 0.000 -100.0 0.000 0.00 23.7238 1.20000 \$End plug
467	C
468	C Aluminum Rods
469	C

470 90 RCC 0.000 0.000 15.41 0.000 0.000 66.14 1.77125 \$Void in Al rod
 471 C
 472 C Central Cavity Surfaces
 473 C
 474 100 RCC 0.000 0.000 -67.395 0.000 0.000 202.395 11.6450 \$Void
 475 101 RCC 0.000 0.000 -67.395 0.000 0.000 202.395 12.2800
 476 102 RCC 0.000 0.000 -67.395 0.000 0.000 202.395 13.9700
 477 C
 478 C Cavity Additions
 479 C
 480 110 RCC 0.000 0.000 -67.395 0.000 0.000 81.28 11.4300 \$32-in pedestal
 481 111 RCC 0.000 0.000 8.4748 0.000 0.000 2.8702 8.2550 \$32-in inset
 482 112 RPP -0.9525 0.9525 -8.255 8.255 11.345 13.885 \$Inset Notch
 483 113 RCC 0.000 0.000 13.885 0.000 0.000 1.270 10.3188 \$Bottom plate (8-in)
 484 114 RCC 0.000 0.000 32.935 0.000 0.000 1.270 10.3188 \$Top plate (8-in)
 485 115 RCC 0.000 0.000 15.155 0.000 0.000 17.78 5.7150 \$Center void (8-in)
 486 116 RCC 0.000 0.000 15.155 0.000 0.000 17.78 6.3500 \$Support tube (8-in)
 487 C
 488 C Top and Bottom Grid Plates
 489 C
 490 200 RCC 0.000 0.000 80.55 0.000 0.000 2.54 53.3500 \$Top plate
 491 201 PY -34.925 \$Cutoff of top plate
 492 202 RCC 0.000 0.000 11.33 0.000 0.000 5.08 47.0000 \$Bottom plate
 493 C
 494 C
 495 C Window to Radiography Lab
 496 C
 497 210 1 RPP 38.100 39.370 -26.670 26.670 16.41 80.55 \$Ni plate
 498 211 1 RPP 38.100 39.370 -38.100 38.100 16.41 80.55 \$Water
 499 212 1 RPP 48.895 100.00 -26.670 26.670 16.41 80.55 \$Void
 500 213 1 RPP 39.370 100.00 -38.100 38.100 16.41 80.55 \$Aluminum
 501 C
 502 C Nickel Plate near FREC-II
 503 C
 504 220 RPP -36.830 36.830 -36.195 -34.925 16.41 83.09 \$Nickel Plate
 505 C
 506 C Hexes for the lattice, inner and outer core, and core boundary
 507 C
 508 320 RHP 0.0 0.0 -132.0 0.0 0.0 400.0 2.0855 0.0 0.0 \$Lattice element
 509 300 1 RHP 0.0 0.0 16.41 0.0 0.0 64.14 42.7 0.0 0.0 \$Outer core bound
 510 310 1 RHP 0.0 0.0 -67.395 0.000 0.000 202.395 11.65 0.0 0.0 \$Inner liner of cavity
 511 311 1 RHP 0.0 0.0 -67.395 0.000 0.000 202.395 12.285 0.0 0.0 \$Outer liner of cavity
 512 C

513 C
 514 C Buckets (700's)
 515 C
 516 C Pb-B4C Bucket
 517 C
 518 700 7 RCC 0.0 0.0 6.35 0.0 0.0 85.09 6.27380 \$Void
 519 701 7 RCC 0.0 0.0 6.26872 0.0 0.0 85.17128 6.35508 \$0.032" Al liner
 520 702 7 RCC 0.0 0.0 3.65125 0.0 0.0 87.78875 9.76630 \$Pb layers
 521 703 7 RCC 0.0 0.0 3.01625 0.0 0.0 0.63500 9.17575 \$Boral
 522 704 7 RCC 0.0 0.0 1.90500 0.0 0.0 89.53500 10.1600 \$Al layer
 523 705 7 RCC 0.0 0.0 1.90500 0.0 0.0 89.53500 11.1125 \$B4C
 524 706 7 RCC 0.0 0.0 0.00000 0.0 0.0 91.44000 11.4300 \$Al layer
 525 707 7 RCC 0.0 0.0 3.65125 0.0 0.0 87.78875 9.84250
 526 708 7 RCC 0.0 0.0 1.90500 0.0 0.0 0.63500 7.62000 \$B4C bottom
 527 7091 7 RCC 0.0 -8.890 0.00 0.0 0.0 91.44000 0.31750 \$Dowel 1
 528 7092 7 RCC 0.0 8.890 0.00 0.0 0.0 91.44000 0.31750 \$Dowel 2
 529 C
 530 C 14" Aluminum Bucket
 531 C
 532 710 7 RCC 0.0 0.0 0.15875 0.0 0.0 35.40125 11.27125 \$Void
 533 711 7 RCC 0.0 0.0 0.00000 0.0 0.0 35.56000 11.43000 \$Al bucket
 534 C
 535 C USE these for a 24" Aluminum Bucket
 536 C
 537 C 710 7 RCC 0.0 0.0 0.15875 0.0 0.0 60.80125 11.27125 \$Void
 538 C 711 7 RCC 0.0 0.0 0.00000 0.0 0.0 60.96000 11.43000 \$Al bucket
 539 C
 540 C LP-1 Surfaces
 541 C
 542 720 7 RCC 0.0 0.0 5.74675 0.0 0.0 62.18825 7.46125 \$Inside
 543 721 7 RCC 0.0 0.0 5.58800 0.0 0.0 73.15200 7.62000 \$Al liner
 544 722 7 RCC 0.0 0.0 3.55600 0.0 0.0 64.37900 8.63600 \$Pb
 545 723 7 RCC 0.0 0.0 2.54000 0.0 0.0 65.39500 10.66800 \$HDPE
 546 724 7 RCC 0.0 0.0 3.55600 0.0 0.0 1.01600 7.62000 \$HDPE fill-in
 547 725 7 RCC 0.0 0.0 0.00000 0.0 0.0 78.74000 11.43000 \$Al Container
 548 726 7 RCC 0.0 0.0 67.9350 0.0 0.0 10.80500 7.46125
 549 C
 550 C Boom Box Surfaces
 551 C
 552 730 7 RCC 0.0 0.0 0.0 0.0 0.0 65.786 9.8425
 553 731 7 RCC 0.0 0.0 65.913 0.0 0.0 3.048 9.8425
 554 732 7 RCC 0.0 0.0 65.786 0.0 0.0 0.127 8.1280
 555 733 7 RCC 0.0 0.0 65.786 0.0 0.0 3.175 5.0800

556	734	7	RCC	0.0	0.0	65.786	0.0	0.0	0.127	9.8425	
557	735	7	RCC	0.0	0.0	59.436	0.0	0.0	6.350	6.6675	
558	736	7	RCC	0.0	0.0	59.436	0.0	0.0	6.350	6.7945	
559	737	7	RCC	0.0	0.0	2.540	0.0	0.0	54.864	7.9375	
560	738	7	RCC	0.0	0.0	57.404	0.0	0.0	2.032	5.0800	
561	C										
562	C										
563	C New 44" Pb-B4C Bucket										
564	C Base Plate w/ B4C Volume										
565	C From Ktech drawing labeled "PbB BASEII"										
566	C										
567	800	7	RCC	0.000	0.000	0.000	0.000	0.000	1.905	11.43	\$ Base Plate Bottom
568	801	7	RCC	0.000	0.000	1.905	0.000	0.000	1.905	7.9375	\$ Base Plate Top
569	802	7	RCC	0.000	0.000	2.2225	0.000	0.000	1.27	7.62	\$ B4C Cavity
570	C Bolts/Bolt Holes										
571	C Big Bolts/Bolt Holes										
572	C From McMaster-Carr catalog, Item # 98914A033, Threaded Rods and Studs										
573	803	7	RCC	8.890	0.000	0.000	0.000	0.000	1.905	0.53594	\$ All-Thread #1
574	804	7	RCC	8.890	0.000	1.524	0.000	0.000	0.381	0.65151	\$ All-Thread Hole #1
575	805	7	RCC	8.890	0.000	1.2954	0.000	0.000	0.2286	0.674688	\$ All-Thread Washer Void #1
576	806	7	RCC	8.890	0.000	1.2954	0.000	0.000	0.2286	1.27	\$ All-Thread Washer #1
577	807	7	RCC	8.890	0.000	0.50165	0.000	0.000	0.79375	0.9525	\$ All-Thread Nut #1
578	808	7	RCC	8.890	0.000	0.000	0.000	0.000	1.524	1.5875	\$ All-Thread Nut Hole #1
579	809	7	RCC	-8.890	0.000	0.000	0.000	0.000	1.905	0.53594	\$ All-Thread #2
580	810	7	RCC	-8.890	0.000	1.524	0.000	0.000	0.381	0.65151	\$ All-Thread Hole #2
581	811	7	RCC	-8.890	0.000	1.2954	0.000	0.000	0.2286	0.674688	\$ All-Thread Washer Void #2
582	812	7	RCC	-8.890	0.000	1.2954	0.000	0.000	0.2286	1.27	\$ All-Thread Washer #2
583	813	7	RCC	-8.890	0.000	0.50165	0.000	0.000	0.79375	0.9525	\$ All-Thread Nut #2
584	814	7	RCC	-8.890	0.000	0.000	0.000	0.000	1.524	1.5875	\$ All-Thread Nut Hole #2
585	C Small Bolts/Bolt Holes										
586	C From McMaster-Carr catalog, Item # 44705K334, Low-Pressure Aluminum Threaded Square-Socket Plug										
587	815	7	RCC	0.000	5.3975	0.000	0.000	0.000	2.2225	0.71374	\$ Small Hole #1
588	816	7	RCC	0.000	-5.3975	0.000	0.000	0.000	2.2225	0.71374	\$ Small Hole #2
589	817	7	RCC	0.000	5.3975	0.000	0.000	0.000	1.2192	0.71374	\$ Plug #1
590	818	7	RCC	0.000	-5.3975	0.000	0.000	0.000	1.2192	0.71374	\$ Plug #2
591	819	7	RPP	-0.357188	0.357188	5.040313	5.754688	0.000	0.862648		\$ Plug #1 9/32" Hole
592	820	7	RPP	-0.357188	0.357188	-5.754688	-5.040313	0.000	0.862648		\$ Plug #2 9/32" Hole
593	C										
594	C Containment Base										
595	C From Ktech drawing labeled "CONTAINMENT BASE II"										
596	C										
597	830	7	RCC	0.000	0.000	1.905	0.000	0.000	1.905	7.9883	\$ Inner void
598	831	7	RCC	0.000	0.000	1.905	0.000	0.000	1.905	9.779	\$ Inner Disc Region

599	832	7	RCC	0.000	0.000	1.905	0.000	0.000	0.635	11.43	\$	Outer Disc Region	
600	833	7	RCC	8.890	0.000	1.905	0.000	0.000	1.905	0.53594	\$	All-thread #1, General purpose steel	
601	834	7	RCC	-8.890	0.000	1.905	0.000	0.000	1.905	0.53594	\$	All-thread #2, General purpose steel	
602	C												
603	C Lead Bottom, Floor, and Rings (Items #4, 5, and 6 in DWG titled "LEAD BORON BUCKET ASSEMBLY II")												
604	C Drawn March 22, 2010 by S. Padias												
605	C Unchamfered lead components												
606	C												
607	840	7	RCC	0.000	0.000	3.810	0.000	0.000	2.540	9.7663	\$	Lead bottom disc	
608	C 841	7	RCC	0.000	0.000	6.350	0.000	0.000	100.33	6.477	\$	Inner lead void	
609	842	7	RCC	0.000	0.000	6.350	0.000	0.000	100.33	9.7663	\$	Lead ring	
610	843	7	RCC	8.89	0.000	3.810	0.000	0.000	106.68	0.65532	\$	Right-side big lead hole	
611	844	7	RCC	-8.89	0.000	3.810	0.000	0.000	106.68	0.65532	\$	Left-side big lead hole	
612	845	7	RCC	0.000	8.89	3.810	0.000	0.000	106.68	0.32639	\$	Top-side small lead hole	
613	846	7	RCC	0.000	-8.89	3.810	0.000	0.000	106.68	0.32639	\$	Bottom-side small lead hole	
614	C From McMaster-Carr catalog, Item # 98914A033, Threaded Rods and Studs												
615	847	7	RCC	8.89	0.000	3.810	0.000	0.000	113.665	0.53594	\$	Right-side All-Thread	
616	848	7	RCC	-8.89	0.000	3.810	0.000	0.000	113.665	0.53594	\$	Left-side All-Thread	
617	C General Purpose Aluminum Tubing												
618	C From McMaster-Carr catalog, Item # 89965K42, General Purpose Aluminum Tubing												
619	849	7	RCC	0.000	8.89	3.810	0.000	0.000	111.76	0.2286	\$	Top-side inner radius Al6061 Tubing	
620	850	7	RCC	0.000	8.89	3.810	0.000	0.000	111.76	0.3175	\$	Top-side outer radius Al6061 Tubing	
621	851	7	RCC	0.000	-8.89	3.810	0.000	0.000	111.76	0.2286	\$	Bottom-side inner radius Al6061 Tub	
622	852	7	RCC	0.000	-8.89	3.810	0.000	0.000	111.76	0.3175	\$	Bottom-side outer radius Al6061 Tub	
623	C Chamfered lead components												
624	853	7	TRC	0.000	0.000	106.68	0.000	0.000	3.81	6.477	7.62	\$	Chamfered lead ring void
625	854	7	RCC	0.000	0.000	106.68	0.000	0.000	3.81	9.7663	\$	Chamfered lead ring radius	
626	C												
627	C Inner Aluminum 6061 Sleeves (Items #14 and 15 in DWG titled "LEAD BORON BUCKET ASSEMBLY II")												
628	C												
629	860	7	RCC	0.000	0.000	6.35	0.000	0.000	0.08255	6.477	\$	Al6061 Base Plate	
630	861	7	RCC	0.000	0.000	6.43255	0.000	0.000	100.1776	6.39445	\$	Al6061 Sheet void	
631	862	7	RCC	0.000	0.000	6.43255	0.000	0.000	100.1776	6.477	\$	Al6061 Sheet	
632	863	7	RCC	0.000	0.000	106.6102	0.000	0.000	0.0698	6.477	\$	Void	
633	C												
634	C PbB base II Double Wall Weldment (Item #9 in DWG titled "LEAD BORON BUCKET ASSEMBLY II")												
635	C												
636	868	7	RCC	0.000	0.000	2.54	0.000	0.000	107.315	9.8425	\$	Inner Surface of Inner Skin II L1	
637	869	7	RCC	0.000	0.000	2.54	0.000	0.000	107.315	10.16	\$	Inner Skin II L1	
638	870	7	RCC	0.000	0.000	109.855	0.000	0.000	0.254	9.8425	\$	Inner Surface of Inner Skin II L2	
639	871	7	RCC	0.000	0.000	109.855	0.000	0.000	0.254	10.16	\$	Inner Skin II L2	
640	872	7	RCC	0.000	0.000	110.109	0.000	0.000	0.381	9.8425	\$	Inner Surface of Inner Skin II L3	
641	873	7	RCC	0.000	0.000	110.109	0.000	0.000	0.381	10.16	\$	Inner Skin II L3	

642	874	7	RCC	0.000	0.000	2.54	0.000	0.000	107.315	11.1125	\$ B4C Powder Region L1
643	875	7	RCC	0.000	0.000	2.54	0.000	0.000	107.315	11.43	\$ Outer Skin II L1
644	876	7	RCC	0.000	0.000	109.855	0.000	0.000	0.254	11.1125	\$ B4C Powder Region L2
645	877	7	RCC	0.000	0.000	109.855	0.000	0.000	0.254	11.43	\$ Outer Skin II L2
646	878	7	RCC	0.000	0.000	110.109	0.000	0.000	0.381	11.1125	\$ B4C Powder Region L3
647	879	7	RCC	0.000	0.000	110.109	0.000	0.000	0.381	11.43	\$ Outer Skin II L3
648	C										
649	C PbB Top: Top Plate (Item #1 in DWG titled "LEAD BORON BUCKET ASSEMBLY II")										
650	C										
651	880	7	RCC	0.000	0.000	110.49	0.000	0.000	0.4318	7.62	\$ Lower void
652	881	7	TRC	0.000	0.000	110.9218	0.000	0.000	0.8382	7.62	8.103935 \$ Upper void
653	882	7	RCC	0.000	0.000	110.49	0.000	0.000	1.27	11.43	\$ Al6061 Disc
654	883	7	RCC	0.000	0.000	110.49	0.000	0.000	-0.381	10.1727	\$ Lower B4C Cap void
655	884	7	RCC	0.000	0.000	110.49	0.000	0.000	-0.381	11.0998	\$ B4C Cap
656	885	7	TRC	0.000	0.000	109.855	0.000	0.000	0.254	10.4267	10.1727 \$ Inner chamfer
657	886	7	TRC	0.000	0.000	109.855	0.000	0.000	0.254	10.8458	11.0998 \$ Outer chamfer
658	C Top Plate holes										
659	887	7	RCC	-8.89	0.000	110.49	0.000	0.000	1.27	0.65151	\$ Left Large hole
660	888	7	RCC	8.89	0.000	110.49	0.000	0.000	1.27	0.65151	\$ Right Large hole
661	889	7	RCC	0.000	-8.89	110.49	0.000	0.000	1.27	0.65151	\$ Bottom Large hole
662	890	7	RCC	0.000	8.89	110.49	0.000	0.000	1.27	0.65151	\$ Top Large hole
663	891	7	RCC	-2.54	8.89	110.49	0.000	0.000	1.27	0.3175	\$ Small Hole #1
664	892	7	RCC	2.54	8.89	110.49	0.000	0.000	1.27	0.3175	\$ Small Hole #2
665	893	7	RCC	-2.54	-8.89	110.49	0.000	0.000	1.27	0.3175	\$ Small Hole #1
666	894	7	RCC	2.54	-8.89	110.49	0.000	0.000	1.27	0.3175	\$ Small Hole #2
667	C Modified Hex Head Plugs, 1/4 NPT, AL6061-T6 (Item #16 in DWG titled "LEAD BORON BUCKET ASSEMBLY")										
668	C From McMaster-Carr catalog, Item # 3867T65, High-Pressure Aluminum Pipe Fitting										
669	821	7	RCC	0.000	-8.89	111.76	0.000	0.000	-1.27	0.65151	\$ Bottom Hex Thread
670	822	7	RCC	0.000	-8.89	111.76	0.000	0.000	0.635	0.79375	\$ Bottom Hex Head
671	823	7	RCC	0.000	8.89	111.76	0.000	0.000	-1.27	0.65151	\$ Top Hex Thread
672	824	7	RCC	0.000	8.89	111.76	0.000	0.000	0.635	0.79375	\$ Top Hex Head
673	C										
674	C Enclosing surface for the 44" Pb-B4C bucket										
675	C										
676	899	7	RCC	0.000	0.000	0.000	0.000	0.000	117.475	11.43	\$ Enclosing surface
677	C										
678	C										
679	C EXPERIMENT SURFACES										
680	C										
681	1001	6	SO	3.							\$ 6 cm dia scoring sphere
682	1002	6	rcc	0.	-0.013301675	0.	0.	0.026603351	0.	0.635	\$ Nickel Foil
683	C										
684	C External Cutoff										

685 C
 686 900 RCC 0.000 0.000 -67.395 0.000 0.000 202.395 72.0000
 687
 688 C *****
 689 C * TRANSFORMATIONS *
 690 C *****
 691 C
 692 C TR1 rotates the hexes for the outer core bound and the cavity liner
 693 C
 694 *TR1 0 0 0 30 60 90 120 30 90
 695 C
 696 C TR3-->Movement of control rods -0.001 (full up) to -55.001 (full down)
 697 C
 698 C
 699 C Measured Up DC with 32-in pedestal is -39.731 (03/03/2004)
 700 C Measured Down DC is -30.851 (03/03/2004)
 701 C -->Up DC position of 1527 Rod Units
 702 C -->Down DC position of 2415 Rod Units
 703 C Measured Up DC with 8-in + 32-in pedestal is -40.421 (03/01/2004)
 704 C Measured Down DC with 8-in + 32-in pedestal is -31.291 (03/01/2004)
 705 C -->Up DC position of 1428 Rod Units
 706 C -->Down DC position of 2371 Rod Units
 707 C Measured Up DC with Pb-B4C on 32-in pedestal is -22.951 (03/09/2004)
 708 C Measured Down DC with Pb-B4C on 32-in pedestal is -10.741 (03/09/2004)
 709 C -->Up DC position of 3205 Rod Units
 710 C -->Down DC position of 4426 Rod Units
 711 C Measured DC with LP-1 on 32-in pedestal is -31.941 (03/11/2004)
 712 C Measured Down DC with LP-1 on 32-in pedestal is -23.721 (03/11/2004)
 713 C -->Up DC position of 2306 Rod Units
 714 C -->Down DC position of 3128 Rod Units
 715 C
 716 *TR3 0 0 -41.50
 717 C
 718 C TR4-->Movement of safety rods 0.001 (full up) to -54.999 (full down)
 719 C
 720 C Measured worth of safety rods: -\$2.12 (03/30/2004)
 721 C
 722 *TR4 0 0 0.001
 723 C
 724 C TR5-->Movement of transient rods 0 (full down) to 90 (full up)
 725 C
 726 C Measured worth of transient rods: -\$4.14 (03/30/2004)
 727 C

728 *TR5 0 0 90
729 C
730 C TR6-->Moves experiment package from origin (0 0 0) to fuel centerline
731 C
732 *TR6 0 0 49.445
733 C
734 C TR7-->Puts buckets on 8" (34.205) or 32" (13.885) pedestals
735 C Use 32" pedestal for LP-1
736 C Use 8" for Standard AI buckets
737 C
738 *TR7 0 0 13.885
739 C
740 C *****
741 C * MATERIAL CARDS *
742 C *****
743 C Materials cards use the latest available cross sections
744 C
745 C UO2-BeO fuel (3.3447 g/cc) (XSEC Temp - 293.6 K)
746 C
747 C
748 M1 4009.70c -0.2827602 8016.70c -0.5277690 92235.70c -0.0662957
749 92238.70c -0.1222844 92234.70c -0.0004547 92236.70c -0.0004358
750 MT1 beo.60t \$ S(alpha, beta) for UO2-BeO (Temp - 294 K)
751 C
752 C
753 C UO2-BeO fuel (3.0102 g/cc) -- This is the 90% fuel
754 C (XSEC Temp - 293.6 K)
755 C Included as a separate material to avoid warning message
756 C
757 C
758 M11 4009.70c -0.2827602 8016.70c -0.5277690 92235.70c -0.0662957
759 92238.70c -0.1222844 92234.70c -0.0004547 92236.70c -0.0004358
760 MT11 beo.60t \$ S(alpha, beta) for BeO (Temp - 294 K)
761 C
762 C NIOBIUM (8.4 g/cc)
763 C
764 M2 41093.70c 1.0000
765 C
766 C
767 C SS-304L from Ktech Materials Database Rev. 118
768 C Material Number: 3410
769 C Values are weight %
770 C Si: 0.0100 Cr: 0.1900 Mn: 0.0200 Fe: 0.6800 Ni: 0.1000

771 C
 772 C FM multiplier (neutrons): 1.76109641E-10 3410 -4 1
 773 C FM multiplier (photons): 1.76109641E-10 3410 -5 -6
 774 C
 775 C Density: 7.896 g/cc
 776 C
 777 M3 14028.70c -0.009187 14029.70c -0.000483 14030.70c -0.000329
 778 24050.70c -0.007930 24052.70c -0.159029 24053.70c -0.018380
 779 24054.70c -0.004661 25055.70c -0.020000 26054.70c -0.038390
 780 26056.70c -0.624930 26057.70c -0.014691 26058.70c -0.001989
 781 28058.70c -0.067198 28060.70c -0.026776 28061.70c -0.001183
 782 28062.70c -0.003834 28064.70c -0.001009
 783 C
 784 C
 785 C BeO (2.8 g/cc)
 786 C
 787 M4 4009.70c 0.5000 8016.70c 0.4998096 8017.70c 0.0001904
 788 MT4 beo.60t \$ S(alpha, beta) for BeO (Temp - 294 K)
 789 C
 790 C Water (1 g/cc)
 791 C
 792 M5 1001.70c 0.6665667 1002.70c 0.000100
 793 8016.70c 0.3332063 8017.70c 0.000127
 794 MT5 lwtr.60t \$ S(alpha, beta) for water (Temp - 294 K)
 795 C
 796 C
 797 C Ni reflector
 798 C Values are weight %
 799 C Ni-58: 67.19780 Ni-60: 26.77586 Ni-61: 1.18346
 800 C Ni-62: 3.83429 Ni-64: 1.00859
 801 C Converted Data from Nuclear Wallet Card to w/o with "Weight_Frac" program
 802 C Density: 8.9020 g/cc
 803 C
 804 C
 805 M6 28058.70c -0.6719780 28060.70c -0.2677586 28061.70c -0.0118346
 806 28062.70c -0.0383429 28064.70c -0.0100859
 807 C
 808 C
 809 C Al-6061 from Ktech Materials Database Rev. 118
 810 C Material Number: 3110
 811 C Values are weight %
 812 C Mg: 0.0110 Al: 0.9670 Si: 0.0080 Ti: 0.0007
 813 C Cr: 0.0020 Mn: 0.0013 Fe: 0.0056 Ni: 0.0004

814 C Cu: 0.0030 Zn: 0.0010
 815 C
 816 C FM multiplier (neutrons): 3.55249469E-10 3110 -4 1
 817 C FM multiplier (photons): 3.55249469E-10 3110 -5 -6
 818 C
 819 C Density: 2.704 g/cc
 820 C
 821 M7 12000.66c -0.011000 13027.70c -0.967000 14028.70c -0.007350
 822 14029.70c -0.000387 14030.70c -0.000263 22000.66c -0.000700
 823 24050.70c -0.000084 24052.70c -0.001674 24053.70c -0.000193
 824 24054.70c -0.000049 25055.70c -0.001300 26054.70c -0.000316
 825 26056.70c -0.005147 26057.70c -0.000121 26058.70c -0.000016
 826 28058.70c -0.000269 28060.70c -0.000107 28061.70c -0.000005
 827 28062.70c -0.000015 28064.70c -0.000004 29063.70c -0.002055
 828 29065.70c -0.000945 30000.42c -0.001000
 829 C
 830 C
 831 C B4C poison (2.48 g/cc)
 832 C Composition data taken from Jeff Wemple (KTech) Memo dated June 18, 2010
 833 C and titled "Re: Drawing of new Lead-boron bucket"
 834 C Manufacturer of powder is READE ADVANCED MATERIALS
 835 C Density of packed powder in the 44" Pb-B4C bucket is 1.2505 (half of 2.51 g/cc)
 836 C
 837 M8 6000.70c 0.20000 5010.70c 0.159200 5011.70c 0.640800
 838 C
 839 C
 840 C Natural Lead
 841 C True Weight %: Pb-204: 1.37808 Pb-206: 23.95550
 842 C Pb-207: 22.07430 Pb-208: 52.59212
 843 C Weight % based on Available MCNP XSEC:
 844 C Pb-206: 24.29024 Pb-207: 22.38275
 845 C Pb-208: 53.32701
 846 C
 847 C Converted Data from Nuclear Wallet Card to w/o with "Weight_Frac" program
 848 C Density is 11.35 g/cc from Nuclear Wallet Cards.
 849 C
 850 C
 851 M700 82206.70c -0.2429024 82207.70c -0.2238275
 852 82208.70c -0.5332701
 853 C
 854 C
 855 C Boral Plate Composition
 856 C

857 C Composition found in Nuclear Science and Engineering
 858 C Vol. 65, No. 1, pgs. 41-48, January 1978.
 859 C Values are weight %
 860 C B: 27.40 C: 7.61 Al: 63.68
 861 C Cu: 0.09 Zn: 0.16 Fe: 0.45
 862 C Cr: 0.10 Mn: 0.10 Mg: 0.05
 863 C Ti: 0.10 Li: 0.26
 864 C
 865 C Density: 2.53 g/cc
 866 C
 867 C
 868 M701 5010.70c -0.050242 5011.70c -0.223758 6000.70c -0.076100
 869 13027.70c -0.636800 29063.70c -0.000616 29065.70c -0.000284
 870 30000.42c -0.001600 26056.70c -0.004500 24050.70c -0.000042
 871 24052.70c -0.000837 24053.70c -0.000097 24054.70c -0.000024
 872 25055.70c -0.001000 12000.66c -0.000500 22000.66c -0.001000
 873 3006.70c -0.000171 3007.70c -0.002429
 874 C
 875 C
 876 C
 877 C Air
 878 C Standard Density: 1.205e-3 g/cc @ 20 deg C, 1 atm
 879 C Albuquerque: 1.0245e-3 g/cc in ABQ
 880 C See Attix p.531-532
 881 C
 882 M702 7014.70c -0.752308 7015.70c -0.002960 8016.70c -0.231687
 883 8017.70c -0.000094 6000.70c -0.000124 18000.42c -0.012827
 884 C
 885 C
 886 C HELIUM For Leak Test
 887 C @ 2 atm density = 3.57e-4 g/cc
 888 C
 889 M703 2003.70c 0.00000137 2004.70c 0.99999863
 890 C
 891 C
 892 C HDPE-> (C2H4)n -- --
 893 C | H H |
 894 C | | | |
 895 C -|- C -- C -|-
 896 C | | | |
 897 C | H H |
 898 C -- --
 899 C

900 M704 1001.70c 0.666667 6000.70c 0.333333
 901 MT704 poly.60t
 902 C
 903 C
 904 C A517 Carbon Steel (den = 7.28 g/cc)
 905 C Modified to match the mill test cert.
 906 C from Tubos de Acero de Mexico, S.A.
 907 C
 908 C Summary of MatMCNP Calculations:
 909 C
 910 C Isotope Number Fraction Weight Fraction Atoms/b-cm
 911 C C-12 0.0118115 0.0025688 0.0009385
 912 C C-13 0.0001326 0.0000312 0.0000105
 913 C Si-28 0.0048924 0.0024807 0.0003887
 914 C Si-29 0.0002484 0.0001305 0.0000197
 915 C Si-30 0.0001638 0.0000890 0.0000130
 916 C Cr-50 0.0000184 0.0000167 0.0000015
 917 C Cr-52 0.0003557 0.0003348 0.0000283
 918 C Cr-53 0.0000403 0.0000387 0.0000032
 919 C Cr-54 0.0000100 0.0000098 0.0000008
 920 C Mn-55 0.0078340 0.0078003 0.0006225
 921 C Fe-54 0.0568920 0.0556172 0.0045205
 922 C Fe-56 0.8930822 0.9053674 0.0709617
 923 C Fe-57 0.0206252 0.0212829 0.0016388
 924 C Fe-58 0.0027448 0.0028820 0.0002181
 925 C Cu-63 0.0007628 0.0008700 0.0000606
 926 C Cu-65 0.0003400 0.0004001 0.0000270
 927 C Mo-92 0.0000068 0.0000114 0.0000005
 928 C Mo-94 0.0000043 0.0000072 0.0000003
 929 C Mo-95 0.0000073 0.0000126 0.0000006
 930 C Mo-96 0.0000077 0.0000133 0.0000006
 931 C Mo-97 0.0000044 0.0000077 0.0000003
 932 C Mo-98 0.0000111 0.0000197 0.0000009
 933 C Mo-100 0.0000044 0.0000080 0.0000004
 934 C
 935 C The total compound atom density (atom/b-cm): 0.07945702
 936 C
 937 M765 06000.70c 0.0119440 14028.70c 0.0048924 14029.70c 0.0002484
 938 14030.70c 0.0001638 24050.70c 0.0000184 24052.70c 0.0003557
 939 24053.70c 0.0000403 24054.70c 0.0000100 25055.70c 0.0078340
 940 26054.70c 0.0568920 26056.70c 0.8930822 26057.70c 0.0206252
 941 26058.70c 0.0027448 29063.70c 0.0007628 29065.70c 0.0003400
 942 42000.66c 0.0000460

943 C
 944 C
 945 C
 946 C General Purpose Steel, Grade B7
 947 C 7 Comment Cards
 948 C
 949 C 1
 950 C 2 The weight fraction for elements of General Purpose Steel, Grade B7 is
 951 C 3 The weight fractions are used for each element.
 952 C 4 The density of natural cadmium is 7.83 g/cc,
 953 C 5 The MCNP material number is found after the material.
 954 C 6 The line below "7" gives the density.
 955 C 7
 956 C
 957 C Summary of MatMCNP Calculations:
 958 C
 959 C Isotope Number Fraction Weight Fraction Atoms/b-cm
 960 C C-12 0.0194064 0.0042483 0.0016694
 961 C C-13 0.0002178 0.0000517 0.0000187
 962 C Mn-55 0.0087306 0.0087500 0.0007510
 963 C P-31 0.0006194 0.0003500 0.0000533
 964 C S-32 0.0006498 0.0003790 0.0000559
 965 C S-33 0.0000051 0.0000031 0.0000004
 966 C S-34 0.0000288 0.0000178 0.0000025
 967 C S-36 0.0000001 0.0000001 0.0000000
 968 C Si-28 0.0045003 0.0022968 0.0003871
 969 C Si-29 0.0002285 0.0001208 0.0000197
 970 C Si-30 0.0001506 0.0000824 0.0000130
 971 C Cr-50 0.0004466 0.0004069 0.0000384
 972 C Cr-52 0.0086125 0.0081607 0.0007409
 973 C Cr-53 0.0009766 0.0009432 0.0000840
 974 C Cr-54 0.0002431 0.0002392 0.0000209
 975 C Mo-92 0.0001696 0.0002843 0.0000146
 976 C Mo-94 0.0001057 0.0001811 0.0000091
 977 C Mo-95 0.0001819 0.0003150 0.0000157
 978 C Mo-96 0.0001906 0.0003335 0.0000164
 979 C Mo-97 0.0001091 0.0001929 0.0000094
 980 C Mo-98 0.0002758 0.0004925 0.0000237
 981 C Mo-100 0.0001101 0.0002006 0.0000095
 982 C Fe-54 0.0557637 0.0548720 0.0047968
 983 C Fe-56 0.8753707 0.8932369 0.0753001
 984 C Fe-57 0.0202161 0.0209977 0.0017390
 985 C Fe-58 0.0026904 0.0028434 0.0002314

986 C
 987 C The total compound atom density (atom/b-cm): 0.08602087
 988 C
 989 C This material contains an isotope that is often
 990 C modified by an S(alpha,beta). Check MCNP
 991 C Manual Appendix G to see if an
 992 C S(alpha,beta) is required.
 993 C
 994 C MCNP Material 766
 995 C
 996 M766 06000.70c 0.0196242
 997 25055.70c 0.0087306
 998 15031.70c 0.0006194
 999 16000.66c 0.0006838
 1000 14028.70c 0.0045003
 1001 14029.70c 0.0002285
 1002 14030.70c 0.0001506
 1003 24050.70c 0.0004466
 1004 24052.70c 0.0086125
 1005 24053.70c 0.0009766
 1006 24054.70c 0.0002431
 1007 42000.66c 0.0011428
 1008 26054.70c 0.0557637
 1009 26056.70c 0.8753707
 1010 26057.70c 0.0202161
 1011 26058.70c 0.0026904
 1012 C
 1013 C Caution: The natural zaid is used for Carbon.
 1014 C
 1015 C Caution: The natural zaid is used for Sulfur.
 1016 C
 1017 C Caution: The natural zaid is used for Molybdenum.
 1018 C
 1019 C If the natural zaid is used for any element, the atom fractions of each isotope
 1020 C of that element are added together and listed with the natural zaid just once.
 1021 C
 1022 C To convert a particle flux to rad[Material]
 1023 C use FM 1.76023016E-10 766 -4 1 for neutrons
 1024 C or FM 1.76023016E-10 766 -5 -6 for photons.
 1025 C
 1026 C
 1027 C Carbon Steel
 1028 C 8 Comment Cards

1029 C

1030 C 1

1031 C 2 The weight fraction for elements of carbon steel is used.

1032 C 3 The weight fractions are used for each element.

1033 C 4 Data obtained from MCNP Primer by C.D. Harmon and R.D. Busch (1994)

1034 C 5 The density of natural cadmium is 7.82 g/cc,

1035 C 6 The MCNP material number is found after the material.

1036 C 7 The line below "8" gives the density.

1037 C 8

1038 C

1039 C Summary of MatMCNP Calculations:

1040 C

Isotope	Number Fraction	Weight Fraction	Atoms/b-cm
C C-12	0.0225772	0.0049399	0.0019386
C C-13	0.0002534	0.0000601	0.0000218
C Fe-54	0.0571155	0.0561733	0.0049043
C Fe-56	0.8965919	0.9144202	0.0769875
C Fe-57	0.0207062	0.0214957	0.0017780
C Fe-58	0.0027556	0.0029108	0.0002366

1048 C

1049 C The total compound atom density (atom/b-cm): 0.08586678

1050 C

1051 C This material contains an isotope that is often

1052 C modified by an S(alpha,beta). Check MCNP

1053 C Manual Appendix G to see if an

1054 C S(alpha,beta) is required.

1055 C

1056 C MCNP Material 767

1057 C

1058 M767 06000.70c 0.0228306

1059 26054.70c 0.0571155

1060 26056.70c 0.8965919

1061 26057.70c 0.0207062

1062 26058.70c 0.0027556

1063 C

1064 C Caution: The natural zaid is used for Carbon.

1065 C

1066 C If the natural zaid is used for any element, the atom fractions of each isotope

1067 C of that element are added together and listed with the natural zaid just once.

1068 C

1069 C To convert a particle flux to rad[Material]

1070 C use FM 1.75917531E-10 767 -4 1 for neutrons

1071 C or FM 1.75917531E-10 767 -5 -6 for photons.

```

1072 C
1073 C
1074 C *****
1075 C * TALLIES *
1076 C *****
1077 C
1078 C
1079 F24:N 1001
1080 FC24 neutron fluence n/cm**2/source neutron - 63 group
1081 E24 1.00000E-11 5.00000E-09 1.00000E-08
1082 1.50000E-08 2.00000E-08 2.50000E-08
1083 3.00000E-08 3.50000E-08 4.20000E-08
1084 5.00000E-08 5.80000E-08 6.70000E-08
1085 8.00000E-08 1.00000E-07 1.52000E-07
1086 2.51000E-07 4.14000E-07 6.83000E-07
1087 1.12500E-06 1.85500E-06 3.05900E-06
1088 5.04300E-06 8.31500E-06 1.37100E-05
1089 2.26000E-05 3.72700E-05 6.14400E-05
1090 1.01300E-04 1.67000E-04 2.75400E-04
1091 4.54000E-04 7.48500E-04 1.23400E-03
1092 2.03500E-03 2.40400E-03 2.84000E-03
1093 3.35500E-03 5.53100E-03 9.11900E-03
1094 1.50300E-02 1.98900E-02 2.55400E-02
1095 4.08700E-02 6.73800E-02 1.11100E-01
1096 1.83200E-01 3.02000E-01 3.88700E-01
1097 4.97900E-01 6.39279E-01 8.20850E-01
1098 1.10803E+00 1.35335E+00 1.73774E+00
1099 2.23130E+00 2.86505E+00 3.67879E+00
1100 4.96585E+00 6.06500E+00 1.00000E+01
1101 1.49182E+01 1.69046E+01 2.00000E+01
1102 2.50E+01
1103 C
1104 F44:N 1001
1105 FC44 total neutron fluence n/cm**2/source neutron
1106 C
1107 C
1108 MODE N
1109 C 20B
1110 KCODE 10000000 1.0 3 2000
1111 c KCODE 100000 1.0 3 2000
1112 KSRC 20 0 50 0 20 60 30 0 40 0 30 60
1113 C PRINT 10 60 100 110
1114 RAND GEN=2 SEED=19073486328125 STRIDE=152917

```

1115 PRDMP 50 50 0 1 0

APPENDIX B – Example CINDER Files

Input:

```
Thesis Example Run
1.0,100.0,1.0E-20,1.0E-20, , , ,0,0,2,-1,0,0, , ,1,1, ,1000,10000,1000,10000
Thesis Example Run
fluxname
maters
1 1.0
1 's'

1 0.0
3723 's'
```

Fluxes:

```
fluxname          fluxes File      63
                  2.00111E13
9.4200E09 3.0235E10 5.0838E10 6.4828E10 7.4435E10 8.0443E10 8.3643E10 1.1756E11
1.3080E11 1.2267E11 1.2753E11 1.5871E11 1.9902E11 3.1977E11 2.9680E11 2.6857E11
2.7631E11 2.7467E11 2.7916E11 2.7945E11 2.7651E11 2.7672E11 2.9405E11 3.0182E11
3.1505E11 2.7916E11 3.3811E11 3.4493E11 3.6870E11 3.0278E11 3.6467E11 3.6151E11
4.0255E11 1.1534E11 1.1489E11 1.3439E11 4.0025E11 3.6491E11 4.2993E11 2.2570E11
2.5442E11 3.7782E11 5.4444E11 6.2048E11 7.7693E11 9.7421E11 5.8161E11 5.2261E11
6.9985E11 8.2463E11 7.8493E11 6.1056E11 8.2111E11 7.1229E11 6.5644E11 3.9800E11
3.2268E11 1.2249E11 9.3896E10 4.8859E09 9.1226E07 1.8821E07 1.4959E06
```

Locate:

```
C:\Cinder2008\Data\c90lib0742
C:\Cinder2008\Data\cinderogl.dat
```

Material:

```
Thesis Example Run
maters282 0.091347904
0010010 0.04736832344
0020010 0.00000544861
0030020 0.00000000000
0040020 0.00000000000
0060030 0.00000000000
0070030 0.00000000000
0090040 0.00000000000
0100050 0.00000000000
0110050 0.00000000000
0120060 0.02324544377
0130060 0.00025147483
0140070 0.00000000000
0150070 0.00000000000
0160080 0.00000000000
0170080 0.00000000000
0180080 0.00000000000
```

0190090 0.01929183524
0200100 0.00000000000
0210100 0.00000000000
0220100 0.00000000000
0230110 0.00000000000
0240120 0.00000000000
0240120 0.00000000000
0240120 0.00000000000
0270130 0.67168639259
0280140 0.00000000000
0290140 0.00000000000
0300140 0.00000000000
0310150 0.00000000000
0320160 0.19512931252
0330160 0.00154065513
0340160 0.00032037774
0360160 0.00002051010
0350170 0.00000000000
0370170 0.00000000000
0360180 0.00000000000
0380180 0.00000000000
0400180 0.00000000000
0390190 0.00000000000
0400190 0.00000000000
0410190 0.00000000000
0400200 0.00943082372
0420200 0.00006293697
0430200 0.00001313331
0440200 0.00020330822
0460200 0.00000039028
0480200 0.00001818858
0450210 0.00000000000
0460220 0.00000000000
0470220 0.00000000000
0480220 0.00000000000
0490220 0.00000000000
0500220 0.00000000000
0500230 0.00000000000
0510230 0.00000000000
0500240 0.00000000000
0520240 0.00000000000
0530240 0.00000000000
0540240 0.00000000000
0550250 0.00000000000
0540260 0.00000000000
0560260 0.00000000000
0570260 0.00000000000
0580260 0.00000000000
0590270 0.00000000000
0580280 0.02138320562
0600280 0.00823764765
0600280 0.00035810643
0620280 0.00114175452
0640280 0.00029073073
0630290 0.00000000000
0650290 0.00000000000
0640300 0.00000000000
0660300 0.00000000000
0670300 0.00000000000
0680300 0.00000000000
0700300 0.00000000000
0690310 0.00000000000
0710310 0.00000000000
0700320 0.00000000000
0720320 0.00000000000
0730320 0.00000000000
0740320 0.00000000000
0760320 0.00000000000
0750330 0.00000000000
0740340 0.00000000000

0760340 0.0000000000
0770340 0.0000000000
0780340 0.0000000000
0800340 0.0000000000
0820340 0.0000000000
0790350 0.0000000000
0810350 0.0000000000
0780360 0.0000000000
0800360 0.0000000000
0820360 0.0000000000
0830360 0.0000000000
0840360 0.0000000000
0860360 0.0000000000
0850370 0.0000000000
0870370 0.0000000000
0840380 0.0000000000
0860380 0.0000000000
0870380 0.0000000000
0880380 0.0000000000
0890390 0.0000000000
0900400 0.0000000000
0910400 0.0000000000
0920400 0.0000000000
0940400 0.0000000000
0960400 0.0000000000
0930410 0.0000000000
0920420 0.0000000000
0940420 0.0000000000
0950420 0.0000000000
0960420 0.0000000000
0970420 0.0000000000
0980420 0.0000000000
1000420 0.0000000000
0960440 0.0000000000
0980440 0.0000000000
0990440 0.0000000000
1000440 0.0000000000
1010440 0.0000000000
1020440 0.0000000000
1040440 0.0000000000
1030450 0.0000000000
1020460 0.0000000000
1040460 0.0000000000
1050460 0.0000000000
1060460 0.0000000000
1080460 0.0000000000
1100460 0.0000000000
1070470 0.0000000000
1090470 0.0000000000
1060480 0.0000000000
1080480 0.0000000000
1100480 0.0000000000
1110480 0.0000000000
1120480 0.0000000000
1130480 0.0000000000
1140480 0.0000000000
1160480 0.0000000000
1130490 0.0000000000
1150490 0.0000000000
1120500 0.0000000000
1140500 0.0000000000
1150500 0.0000000000
1160500 0.0000000000
1170500 0.0000000000
1180500 0.0000000000
1190500 0.0000000000
1200500 0.0000000000
1220500 0.0000000000
1240500 0.0000000000
1210510 0.0000000000

1230510 0.0000000000
1200520 0.0000000000
1220520 0.0000000000
1230520 0.0000000000
1240520 0.0000000000
1250520 0.0000000000
1260520 0.0000000000
1280520 0.0000000000
1300520 0.0000000000
1270530 0.0000000000
1240540 0.0000000000
1260540 0.0000000000
1280540 0.0000000000
1290540 0.0000000000
1300540 0.0000000000
1310540 0.0000000000
1320540 0.0000000000
1340540 0.0000000000
1360540 0.0000000000
1330550 0.0000000000
1300560 0.0000000000
1320560 0.0000000000
1340560 0.0000000000
1350560 0.0000000000
1360560 0.0000000000
1370560 0.0000000000
1380560 0.0000000000
1380570 0.0000000000
1390570 0.0000000000
1360580 0.0000000000
1380580 0.0000000000
1400580 0.0000000000
1420580 0.0000000000
1410590 0.0000000000
1420600 0.0000000000
1430600 0.0000000000
1440600 0.0000000000
1450600 0.0000000000
1460600 0.0000000000
1480600 0.0000000000
1500600 0.0000000000
1440620 0.0000000000
1470620 0.0000000000
1480620 0.0000000000
1490620 0.0000000000
1500620 0.0000000000
1520620 0.0000000000
1540620 0.0000000000
1510630 0.0000000000
1530630 0.0000000000
1520640 0.0000000000
1540640 0.0000000000
1550640 0.0000000000
1560640 0.0000000000
1570640 0.0000000000
1580640 0.0000000000
1600640 0.0000000000
1590650 0.0000000000
1560660 0.0000000000
1580660 0.0000000000
1600660 0.0000000000
1610660 0.0000000000
1620660 0.0000000000
1630660 0.0000000000
1640660 0.0000000000
1650670 0.0000000000
1620680 0.0000000000
1640680 0.0000000000
1660680 0.0000000000
1670680 0.0000000000

1680680 0.0000000000
1700680 0.0000000000
1690690 0.0000000000
1680700 0.0000000000
1700700 0.0000000000
1710700 0.0000000000
1720700 0.0000000000
1730700 0.0000000000
1740700 0.0000000000
1760700 0.0000000000
1750710 0.0000000000
1760710 0.0000000000
1740720 0.0000000000
1760720 0.0000000000
1770720 0.0000000000
1780720 0.0000000000
1790720 0.0000000000
1800720 0.0000000000
1800730 0.0000000000
1810730 0.0000000000
1800740 0.0000000000
1820740 0.0000000000
1830740 0.0000000000
1840740 0.0000000000
1860740 0.0000000000
1850750 0.0000000000
1870750 0.0000000000
1840760 0.0000000000
1860760 0.0000000000
1870760 0.0000000000
1880760 0.0000000000
1890760 0.0000000000
1900760 0.0000000000
1920760 0.0000000000
1910770 0.0000000000
1930770 0.0000000000
1900780 0.0000000000
1920780 0.0000000000
1940780 0.0000000000
1950780 0.0000000000
1960780 0.0000000000
1980780 0.0000000000
1970790 0.0000000000
1960800 0.0000000000
1980800 0.0000000000
1990800 0.0000000000
2000800 0.0000000000
2010800 0.0000000000
2020800 0.0000000000
2040800 0.0000000000
2030810 0.0000000000
2050810 0.0000000000
2040820 0.0000000000
2060820 0.0000000000
2070820 0.0000000000
2080820 0.0000000000

APPENDIX C – DREAD C# Code

```
1      using System;
2      using System.Collections.Generic;
3      using System.ComponentModel;
4      using System.Data;
5      using System.Drawing;
6      using System.Linq;
7      using System.Text;
8      using System.Threading.Tasks;
9      using System.Windows.Forms;
10
11     namespace DREAD
12     {
13         public partial class DREAD : Form
14         {
15             public DREAD()
16             {
17                 InitializeComponent();
18             }
19
20             private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
21             {
22             }
23
24             private void Form1_Load(object sender, EventArgs e)
25             {
26             }
27
28             private void listView1_SelectedIndexChanged(object sender, EventArgs e)
29             {
30             }
31
32             private void label1_Click(object sender, EventArgs e)
33             {
34             }
35
36             private void label2_Click(object sender, EventArgs e)
37             {
38             }
39
40             private void numericUpDown2_ValueChanged(object sender, EventArgs e)
41             {
42             }
43
44             private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
45             {
46             }
47
48             private void dateTimePicker1_ValueChanged_1(object sender, EventArgs e)
49             {
50             }
51
52             private void numericUpDown3_ValueChanged(object sender, EventArgs e)
53             {
54             }
55
56             private void dateTimePicker1_ValueChanged_2(object sender, EventArgs e)
57             {
58             }
59
60             private void dateTimePicker1_ValueChanged_3(object sender, EventArgs e)
61             {
62             }
63
64             private void dateTimePicker1_ValueChanged_4(object sender, EventArgs e)
65             {
66             }
```

```

67     }
68
69     private void textBox2_TextChanged(object sender, EventArgs e)
70     {
71     }
72
73     }
74
75     private void button1_Click(object sender, EventArgs e)
76     {
77     }
78
79
80     private void files_Click(object sender, EventArgs e)
81     {
82         System.Diagnostics.Process.Start("c:\\cinder\\" + titleTextBox.Text);
83     }
84
85     private void checkedListBox1_SelectedIndexChanged(object sender, EventArgs e)
86     {
87     }
88
89
90     private void label7_Click(object sender, EventArgs e)
91     {
92     }
93
94
95     private void textBox2_TextChanged_1(object sender, EventArgs e)
96     {
97     }
98
99
100    private void execute_Click(object sender, EventArgs e)
101    {
102        System.IO.Directory.CreateDirectory("c:\\cinder\\" + titleTextBox.Text);
103        input_Click(null, null);
104        fluxes_Click(null, null);
105        materials_Click(null, null);
106        locate_Click(null, null);
107        run_Click(null, null);
108        // System.Threading.Thread.Sleep(10000);
109        //results_Click(null, null);
110    }
111
112    private void input_Click(object sender, EventArgs e)
113    {
114        var filestream = new System.IO.StreamWriter("c:\\cinder\\" + titleTextBox.Text + "\\input");
115        filestream.WriteLine(titleTextBox.Text);
116        filestream.Write((vol.Value).ToString("F1")); //volcc Volume in cubic centimeters of the region. 1.0
117        filestream.Write(",");
118        if (pulse.Checked)
119        {
120            filestream.Write(MJ.Value);
121        }
122        if (ss.Checked)
123        {
124            filestream.Write(reactorpower.Value);
125        } //flxmt Scaling factor to be applied at all times to the flux input from the fluxes file and the spallation
126        filestream.Write(","); //production rate of the splprods file. flxmt < 0 indicates that the constant power approximation is to be used. 1.0
127        filestream.Write("1.0E-20"); // flosig Parameter used in justifying termination of chains based on activity.
128        filestream.Write(","); See subsection Chain Termination for details. flosig can not be set to a value larger than the default. 10-12
129        filestream.Write("1.0E-20");
130        filestream.Write(",");
131        filestream.Write(" ");
132        filestream.Write(",");
133        filestream.Write(" ");
134        filestream.Write(",");

```

```

135     filestream.Write(" ");
136     filestream.Write(",");
137     filestream.Write(" ");
138     filestream.Write(",");
139     filestream.Write(0);
140     filestream.Write(",");
141     filestream.Write(0);
142     filestream.Write(",");
143     filestream.Write(2); //nfe
144     filestream.Write(",");
145     filestream.Write(-1);
146     filestream.Write(",");
147     filestream.Write(0);
148     filestream.Write(",");
149     filestream.Write(0);
150     filestream.Write(",");
151     filestream.Write(" ");
152     filestream.Write(",");
153     filestream.Write(" ");
154     filestream.Write(",");
155     filestream.Write(1);
156     filestream.Write(",");
157     filestream.Write(1);
158     filestream.Write(",");
159     filestream.Write(" ");
160     filestream.Write(",");
161     filestream.Write(1000);
162     filestream.Write(",");
163     filestream.Write(10000);
164     filestream.Write(",");
165     filestream.Write(1000);
166     filestream.Write(",");
167     filestream.WriteLine(10000);
168     filestream.WriteLine(titleTextBox.Text);
169     filestream.WriteLine("fluxname");
170     filestream.WriteLine("maters");
171     if (pulse.Checked)
172     {
173         filestream.WriteLine("1 1.0");
174         filestream.WriteLine("1 's'");
175     }
176     if (ss.Checked)
177     {
178         filestream.WriteLine("1 1.0");
179         filestream.Write(seconds.Value + minutes.Value * 60m + hours.Value * 3600m);
180         filestream.WriteLine(" 's'");
181         // filestream.Write(minutes.Value);
182         // filestream.WriteLine(" 'm'");
183         // filestream.Write(hours.Value);
184         // filestream.WriteLine(" 'h'");
185     }
186     filestream.WriteLine("");
187     filestream.WriteLine("1 0.0");
188     filestream.Write(secs.Value + mins.Value * 60m + hrs.Value * 3600m);
189     filestream.WriteLine(" 's'");
190     // filestream.Write(mins.Value);
191     // filestream.WriteLine(" 'm'");
192     //filestream.Write(hrs.Value);
193     //filestream.WriteLine(" 'h'");
194     filestream.Close();
195 }
196
197 private void fluxes_Click(object sender, EventArgs e)
198 {
199     var filestream = new System.IO.StreamWriter("c:\\cinder\\" + titleTextBox.Text + "\\fluxes");
200     filestream.WriteLine("fluxes File 63");
201     filestream.Write("fluxname");
202     if (lbb.Checked)
203     {
204         filestream.WriteLine("1.18547E13");

```

```

205         filestream.WriteLine("4.5874E06 1.6409E07 2.5971E07 3.9814E07 4.5155E07 5.1342E07
5.1096E07 7.2961E07 ");
206         filestream.WriteLine("8.2414E07 8.0008E07 7.4634E07 9.7301E07 1.1131E08 1.4882E08
1.0155E08 7.6311E07 ");
207         filestream.WriteLine("7.7003E07 9.6131E07 2.0502E08 6.2407E08 1.8018E09 4.4514E09
9.7984E09 1.8309E10 ");
208         filestream.WriteLine("3.1169E10 2.0502E08 6.8902E10 9.2876E10 1.1992E11 1.2393E11
1.7689E11 2.0276E11 ");
209         filestream.WriteLine("2.4172E11 7.4279E10 8.2663E10 9.1027E10 2.8699E11 2.8765E11
3.6069E11 2.1235E11 ");
210         filestream.WriteLine("2.3394E11 3.6439E11 5.2508E11 5.9132E11 7.5022E11 9.3830E11
5.3357E11 5.0295E11 ");
211         filestream.WriteLine("6.7080E11 7.5442E11 7.5433E11 4.9778E11 6.5335E11 5.4393E11
4.4518E11 2.4688E11 ");
212         filestream.WriteLine("1.8798E11 6.6964E10 5.1793E10 2.6944E09 4.8302E07 1.4841E07
1.3461E06 ");
213     }
214     if (plg.Checked)
215     {
216         filestream.WriteLine("2.17147E13");
217         filestream.WriteLine("5.6764E10 1.5277E11 2.0943E11 2.4936E11 2.6321E11 2.7426E11
2.7203E11 3.6487E11 ");
218         filestream.WriteLine("3.8463E11 3.4708E11 3.4037E11 4.0595E11 4.7467E11 6.7186E11
4.9556E11 3.7466E11 ");
219         filestream.WriteLine("3.4469E11 3.2664E11 3.1952E11 3.1553E11 3.1538E11 3.1714E11
3.2417E11 3.2847E11 ");
220         filestream.WriteLine("3.3369E11 3.1952E11 3.4211E11 3.4464E11 3.4925E11 3.4043E11
3.5225E11 3.5316E11 ");
221         filestream.WriteLine("3.6030E11 1.1703E11 1.1729E11 1.2071E11 3.6437E11 3.5767E11
3.8401E11 2.2058E11 ");
222         filestream.WriteLine("2.1562E11 3.7511E11 4.7976E11 5.3164E11 6.3792E11 8.0169E11
4.6710E11 4.7392E11 ");
223         filestream.WriteLine("5.7948E11 6.4301E11 6.8383E11 5.1246E11 6.8105E11 6.0354E11
5.3691E11 3.1820E11 ");
224         filestream.WriteLine("2.6449E11 1.0183E11 7.7623E10 4.1275E09 7.7138E07 1.7697E07
2.1680E06 ");
225     }
226     if (lp.Checked)
227     {
228         filestream.WriteLine("2.28739E13");
229         filestream.WriteLine("1.5149E11 4.0994E11 5.6798E11 6.7825E11 7.1438E11 7.3620E11
7.2450E11 9.6230E11 ");
230         filestream.WriteLine("9.9398E11 8.7480E11 8.3354E11 9.5091E11 1.0301E12 1.1948E12
5.7537E11 3.3082E11 ");
231         filestream.WriteLine("2.8871E11 2.6691E11 2.5547E11 2.4840E11 2.4379E11 2.4258E11
2.4225E11 2.4162E11 ");
232         filestream.WriteLine("2.4070E11 2.5547E11 2.3839E11 2.3695E11 2.3585E11 2.3315E11
2.3279E11 2.3058E11 ");
233         filestream.WriteLine("2.2972E11 7.5791E10 7.5668E10 7.5916E10 2.2881E11 2.2721E11
2.3343E11 1.3336E11 ");
234         filestream.WriteLine("1.2285E11 2.3961E11 2.7826E11 3.1877E11 3.8288E11 4.8510E11
2.8947E11 3.0901E11 ");
235         filestream.WriteLine("3.7372E11 4.1902E11 4.7512E11 3.5757E11 4.7920E11 4.3356E11
3.8991E11 2.3935E11 ");
236         filestream.WriteLine("2.0202E11 7.9012E10 6.2090E10 3.4373E09 6.3060E07 1.8563E07
9.3297E05 ");
237     }
238     if (freefield.Checked)
239     {
240         filestream.WriteLine("2.00111E13");
241         filestream.WriteLine("9.4200E09 3.0235E10 5.0838E10 6.4828E10 7.4435E10 8.0443E10
8.3643E10 1.1756E11 ");
242         filestream.WriteLine("1.3080E11 1.2267E11 1.2753E11 1.5871E11 1.9902E11 3.1977E11
2.9680E11 2.6857E11 ");
243         filestream.WriteLine("2.7631E11 2.7467E11 2.7916E11 2.7945E11 2.7651E11 2.7672E11
2.9405E11 3.0182E11 ");
244         filestream.WriteLine("3.1505E11 2.7916E11 3.3811E11 3.4493E11 3.6870E11 3.0278E11
3.6467E11 3.6151E11 ");
245         filestream.WriteLine("4.0255E11 1.1534E11 1.1489E11 1.3439E11 4.0025E11 3.6491E11
4.2993E11 2.2570E11 ");

```

```

246         filestream.WriteLine("2.5442E11 3.7782E11 5.4444E11 6.2048E11 7.7693E11 9.7421E11
5.8161E11 5.2261E11 ");
247         filestream.WriteLine("6.9985E11 8.2463E11 7.8493E11 6.1056E11 8.2111E11 7.1229E11
6.5644E11 3.9800E11 ");
248         filestream.WriteLine("3.2268E11 1.2249E11 9.3896E10 4.8859E09 9.1226E07 1.8821E07
1.4959E06 ");
249     }
250     filestream.Close();
251 }
252
253 private void materials_Click(object sender, EventArgs e)
254 {
255     nuclidesum.Value = (H1.Value + H2.Value + He3.Value + He4.Value + Li6.Value + Li7.Value +
Be9.Value + B10.Value + B11.Value + C12.Value + C13.Value + N14.Value + N15.Value + O16.Value + O17.Value +
O18.Value + F19.Value + Ne20.Value + Ne21.Value + Ne22.Value + Na23.Value + Mg24.Value + Mg25.Value +
Mg26.Value + Al27.Value + Si28.Value + Si29.Value + Si30.Value + P31.Value + S32.Value + S33.Value + S34.Value +
S36.Value + Cl35.Value + Cl37.Value + Ar36.Value + Ar38.Value + Ar40.Value + K39.Value + K41.Value + Ca40.Value +
Ca42.Value + Ca43.Value + Ca44.Value + Ca46.Value + Sc45.Value + Ti46.Value + Ti47.Value + Ti48.Value +
Ti49.Value + Ti50.Value + V51.Value + Cr50.Value + Cr52.Value + Cr53.Value + Cr54.Value + Mn55.Value + Fe54.Value
+ Fe56.Value + Fe57.Value + Fe58.Value + Co59.Value + Ni58.Value + Ni60.Value + Ni61.Value + Ni62.Value +
Ni64.Value + Cu63.Value + Cu65.Value + Zn64.Value + Zn66.Value + Zn67.Value + Zn68.Value + Zn70.Value +
Ga69.Value + Ga71.Value + Ge70.Value + Ge72.Value + Ge73.Value + Ge74.Value + As75.Value + Se74.Value +
Se76.Value + Se77.Value + Se78.Value + Se80.Value + Br79.Value + Br81.Value + Kr78.Value + Kr80.Value +
Kr82.Value + Kr83.Value + Kr84.Value + Kr86.Value + Rb85.Value + Sr84.Value + Sr86.Value + Sr87.Value + Sr88.Value
+ Y89.Value + Zr90.Value + Zr91.Value + Zr92.Value + Zr94.Value + Nb93.Value + Mo92.Value + Mo94.Value +
Mo95.Value + Mo96.Value + Mo97.Value + Mo98.Value + Ru96.Value + Ru98.Value + Ru99.Value + Ru100.Value +
Ru101.Value + Ru102.Value + Ru104.Value + Rh103.Value + Pd102.Value + Pd104.Value + Pd105.Value +
Pd106.Value + Pd108.Value + Pd110.Value + Ag107.Value + Ag109.Value + Cd106.Value + Cd108.Value + Cd110.Value
+ Cd111.Value + Cd112.Value + Cd114.Value + In113.Value + Sn112.Value + Sn114.Value + Sn115.Value +
Sn116.Value + Sn117.Value + Sn118.Value + Sn119.Value + Sn120.Value + Sn122.Value + Sn124.Value + Sb121.Value
+ Sb123.Value + Te120.Value + Te122.Value + Te123.Value + Te124.Value + Te125.Value + Te126.Value + I127.Value
+ Xe124.Value + Xe126.Value + Xe128.Value + Xe129.Value + Xe130.Value + Xe131.Value + Xe132.Value +
Xe134.Value + Xe136.Value + Cs133.Value + Ba132.Value + Ba134.Value + Ba135.Value + Ba136.Value + Ba137.Value
+ Ba138.Value + La139.Value + Ce136.Value + Ce138.Value + Ce140.Value + Ce142.Value + Pr141.Value +
Nd142.Value + Nd143.Value + Nd145.Value + Nd146.Value + Nd148.Value + Sm144.Value + Sm149.Value +
Sm150.Value + Sm152.Value + Sm154.Value + Eu151.Value + Eu153.Value + Gd154.Value + Gd155.Value +
Gd156.Value + Gd157.Value + Gd158.Value + Gd160.Value + Tb159.Value + Dy156.Value + Dy158.Value +
Dy160.Value + Dy161.Value + Dy162.Value + Dy163.Value + Dy164.Value + Ho165.Value + Er162.Value + Er164.Value
+ Er166.Value + Er167.Value + Er168.Value + Er170.Value + Tm169.Value + Yb168.Value + Yb170.Value + Yb171.Value
+ Yb172.Value + Yb173.Value + Yb174.Value + Yb176.Value + Lu175.Value + Hf176.Value + Hf177.Value + Hf178.Value
+ Hf179.Value + Hf180.Value + Ta180.Value + Ta181.Value + W180.Value + W182.Value + W183.Value + W184.Value +
W186.Value + Re185.Value + Os184.Value + Os187.Value + Os188.Value + Os189.Value + Os190.Value + Os192.Value
+ Ir191.Value + Ir193.Value + Pt192.Value + Pt194.Value + Pt195.Value + Pt196.Value + Pt198.Value + Au197.Value +
Hg196.Value + Hg198.Value + Hg199.Value + Hg200.Value + Hg201.Value + Hg202.Value + Hg204.Value + Tl203.Value
+ Tl205.Value + Pb204.Value + Pb206.Value + Pb207.Value + Pb208.Value + K40.Value+
256
257     var filestream = new System.IO.StreamWriter("c:\\cinder\\" + titleTextBox.Text + "\\material");
258     filestream.WriteLine(titleTextBox.Text);
259     filestream.Write("maters");
260     filestream.Write("282 "); //total nuclides written to file including 0 atom ones
261     filestream.WriteLine((nuclidesum.Value).ToString("F9")); //sum of all nuclides atom densities
262
263
264     filestream.Write("0010010 "); filestream.WriteLine((H1.Value / nuclidesum.Value).ToString("F11"));
265
266     filestream.Write("0020010 "); filestream.WriteLine((H2.Value / nuclidesum.Value).ToString("F11"));
267
268     filestream.Write("0030020 "); filestream.WriteLine((He3.Value / nuclidesum.Value).ToString("F11"));
269
270     filestream.Write("0040020 "); filestream.WriteLine((He4.Value / nuclidesum.Value).ToString("F11"));
271
272     filestream.Write("0060030 "); filestream.WriteLine((Li6.Value / nuclidesum.Value).ToString("F11"));
273
274     filestream.Write("0070030 "); filestream.WriteLine((Li7.Value / nuclidesum.Value).ToString("F11"));
275
276     filestream.Write("0090040 "); filestream.WriteLine((Be9.Value / nuclidesum.Value).ToString("F11"));
277
278     filestream.Write("0100050 "); filestream.WriteLine((B10.Value / nuclidesum.Value).ToString("F11"));
279
280     filestream.Write("0110050 "); filestream.WriteLine((B11.Value / nuclidesum.Value).ToString("F11"));

```

```

281
282     filestream.Write("0120060 "); filestream.WriteLine((C12.Value / nuclidesum.Value).ToString("F11"));
283
284     filestream.Write("0130060 "); filestream.WriteLine((C13.Value / nuclidesum.Value).ToString("F11"));
285
286     filestream.Write("0140070 "); filestream.WriteLine((N14.Value / nuclidesum.Value).ToString("F11"));
287
288     filestream.Write("0150070 "); filestream.WriteLine((N15.Value / nuclidesum.Value).ToString("F11"));
289
290     filestream.Write("0160080 "); filestream.WriteLine((O16.Value / nuclidesum.Value).ToString("F11"));
291
292     filestream.Write("0170080 "); filestream.WriteLine((O17.Value / nuclidesum.Value).ToString("F11"));
293
294     filestream.Write("0180080 "); filestream.WriteLine((O18.Value / nuclidesum.Value).ToString("F11"));
295
296     filestream.Write("0190090 "); filestream.WriteLine((F19.Value / nuclidesum.Value).ToString("F11"));
297
298     filestream.Write("0200100 "); filestream.WriteLine((Ne20.Value / nuclidesum.Value).ToString("F11"));
299
300     filestream.Write("0210100 "); filestream.WriteLine((Ne21.Value / nuclidesum.Value).ToString("F11"));
301
302     filestream.Write("0220100 "); filestream.WriteLine((Ne22.Value / nuclidesum.Value).ToString("F11"));
303
304     filestream.Write("0230110 "); filestream.WriteLine((Na23.Value / nuclidesum.Value).ToString("F11"));
305
306     filestream.Write("0240120 "); filestream.WriteLine((Mg24.Value / nuclidesum.Value).ToString("F11"));
307
308     filestream.Write("0240120 "); filestream.WriteLine((Mg25.Value / nuclidesum.Value).ToString("F11"));
309
310     filestream.Write("0240120 "); filestream.WriteLine((Mg26.Value / nuclidesum.Value).ToString("F11"));
311
312     filestream.Write("0270130 "); filestream.WriteLine((Al27.Value / nuclidesum.Value).ToString("F11"));
313
314     filestream.Write("0280140 "); filestream.WriteLine((Si28.Value / nuclidesum.Value).ToString("F11"));
315
316     filestream.Write("0290140 "); filestream.WriteLine((Si29.Value / nuclidesum.Value).ToString("F11"));
317
318     filestream.Write("0300140 "); filestream.WriteLine((Si30.Value / nuclidesum.Value).ToString("F11"));
319
320     filestream.Write("0310150 "); filestream.WriteLine((P31.Value / nuclidesum.Value).ToString("F11"));
321
322     filestream.Write("0320160 "); filestream.WriteLine((S32.Value / nuclidesum.Value).ToString("F11"));
323
324     filestream.Write("0330160 "); filestream.WriteLine((S33.Value / nuclidesum.Value).ToString("F11"));
325
326     filestream.Write("0340160 "); filestream.WriteLine((S34.Value / nuclidesum.Value).ToString("F11"));
327
328     filestream.Write("0360160 "); filestream.WriteLine((S36.Value / nuclidesum.Value).ToString("F11"));
329
330     filestream.Write("0350170 "); filestream.WriteLine((Cl35.Value / nuclidesum.Value).ToString("F11"));
331
332     filestream.Write("0370170 "); filestream.WriteLine((Cl37.Value / nuclidesum.Value).ToString("F11"));
333
334     filestream.Write("0360180 "); filestream.WriteLine((Ar36.Value / nuclidesum.Value).ToString("F11"));
335
336     filestream.Write("0380180 "); filestream.WriteLine((Ar38.Value / nuclidesum.Value).ToString("F11"));
337
338     filestream.Write("0400180 "); filestream.WriteLine((Ar40.Value / nuclidesum.Value).ToString("F11"));
339
340     filestream.Write("0390190 "); filestream.WriteLine((K39.Value / nuclidesum.Value).ToString("F11"));
341
342     filestream.Write("0400190 "); filestream.WriteLine((K40.Value / nuclidesum.Value).ToString("F11"));
343
344     filestream.Write("0410190 "); filestream.WriteLine((K41.Value / nuclidesum.Value).ToString("F11"));
345
346     filestream.Write("0400200 "); filestream.WriteLine((Ca40.Value / nuclidesum.Value).ToString("F11"));
347
348     filestream.Write("0420200 "); filestream.WriteLine((Ca42.Value / nuclidesum.Value).ToString("F11"));
349
350     filestream.Write("0430200 "); filestream.WriteLine((Ca43.Value / nuclidesum.Value).ToString("F11"));

```

```

351
352     filestream.Write("0440200 "); filestream.WriteLine((Ca44.Value / nuclidesum.Value).ToString("F11"));
353
354     filestream.Write("0460200 "); filestream.WriteLine((Ca46.Value / nuclidesum.Value).ToString("F11"));
355
356     filestream.Write("0480200 "); filestream.WriteLine((Ca48.Value / nuclidesum.Value).ToString("F11"));
357
358     filestream.Write("0450210 "); filestream.WriteLine((Sc45.Value / nuclidesum.Value).ToString("F11"));
359
360     filestream.Write("0460220 "); filestream.WriteLine((Ti46.Value / nuclidesum.Value).ToString("F11"));
361
362     filestream.Write("0470220 "); filestream.WriteLine((Ti47.Value / nuclidesum.Value).ToString("F11"));
363
364     filestream.Write("0480220 "); filestream.WriteLine((Ti48.Value / nuclidesum.Value).ToString("F11"));
365
366     filestream.Write("0490220 "); filestream.WriteLine((Ti49.Value / nuclidesum.Value).ToString("F11"));
367
368     filestream.Write("0500220 "); filestream.WriteLine((Ti50.Value / nuclidesum.Value).ToString("F11"));
369
370     filestream.Write("0500230 "); filestream.WriteLine((V50.Value / nuclidesum.Value).ToString("F11"));
371
372     filestream.Write("0510230 "); filestream.WriteLine((V51.Value / nuclidesum.Value).ToString("F11"));
373
374     filestream.Write("0500240 "); filestream.WriteLine((Cr50.Value / nuclidesum.Value).ToString("F11"));
375
376     filestream.Write("0520240 "); filestream.WriteLine((Cr52.Value / nuclidesum.Value).ToString("F11"));
377
378     filestream.Write("0530240 "); filestream.WriteLine((Cr53.Value / nuclidesum.Value).ToString("F11"));
379
380     filestream.Write("0540240 "); filestream.WriteLine((Cr54.Value / nuclidesum.Value).ToString("F11"));
381
382     filestream.Write("0550250 "); filestream.WriteLine((Mn55.Value / nuclidesum.Value).ToString("F11"));
383
384     filestream.Write("0540260 "); filestream.WriteLine((Fe54.Value / nuclidesum.Value).ToString("F11"));
385
386     filestream.Write("0560260 "); filestream.WriteLine((Fe56.Value / nuclidesum.Value).ToString("F11"));
387
388     filestream.Write("0570260 "); filestream.WriteLine((Fe57.Value / nuclidesum.Value).ToString("F11"));
389
390     filestream.Write("0580260 "); filestream.WriteLine((Fe58.Value / nuclidesum.Value).ToString("F11"));
391
392     filestream.Write("0590270 "); filestream.WriteLine((Co59.Value / nuclidesum.Value).ToString("F11"));
393
394     filestream.Write("0580280 "); filestream.WriteLine((Ni58.Value / nuclidesum.Value).ToString("F11"));
395
396     filestream.Write("0600280 "); filestream.WriteLine((Ni60.Value / nuclidesum.Value).ToString("F11"));
397
398     filestream.Write("0600280 "); filestream.WriteLine((Ni61.Value / nuclidesum.Value).ToString("F11"));
399
400     filestream.Write("0620280 "); filestream.WriteLine((Ni62.Value / nuclidesum.Value).ToString("F11"));
401
402     filestream.Write("0640280 "); filestream.WriteLine((Ni64.Value / nuclidesum.Value).ToString("F11"));
403
404     filestream.Write("0630290 "); filestream.WriteLine((Cu63.Value / nuclidesum.Value).ToString("F11"));
405
406     filestream.Write("0650290 "); filestream.WriteLine((Cu65.Value / nuclidesum.Value).ToString("F11"));
407
408     filestream.Write("0640300 "); filestream.WriteLine((Zn64.Value / nuclidesum.Value).ToString("F11"));
409
410     filestream.Write("0660300 "); filestream.WriteLine((Zn66.Value / nuclidesum.Value).ToString("F11"));
411
412     filestream.Write("0670300 "); filestream.WriteLine((Zn67.Value / nuclidesum.Value).ToString("F11"));
413
414     filestream.Write("0680300 "); filestream.WriteLine((Zn68.Value / nuclidesum.Value).ToString("F11"));
415
416     filestream.Write("0700300 "); filestream.WriteLine((Zn70.Value / nuclidesum.Value).ToString("F11"));
417
418     filestream.Write("0690310 "); filestream.WriteLine((Ga69.Value / nuclidesum.Value).ToString("F11"));
419
420     filestream.Write("0710310 "); filestream.WriteLine((Ga71.Value / nuclidesum.Value).ToString("F11"));

```

```

421
422     filestream.Write("0700320 "); filestream.WriteLine((Ge70.Value / nuclidesum.Value).ToString("F11"));
423
424     filestream.Write("0720320 "); filestream.WriteLine((Ge72.Value / nuclidesum.Value).ToString("F11"));
425
426     filestream.Write("0730320 "); filestream.WriteLine((Ge73.Value / nuclidesum.Value).ToString("F11"));
427
428     filestream.Write("0740320 "); filestream.WriteLine((Ge74.Value / nuclidesum.Value).ToString("F11"));
429
430     filestream.Write("0760320 "); filestream.WriteLine((Ge76.Value / nuclidesum.Value).ToString("F11"));
431
432     filestream.Write("0750330 "); filestream.WriteLine((As75.Value / nuclidesum.Value).ToString("F11"));
433
434     filestream.Write("0740340 "); filestream.WriteLine((Se74.Value / nuclidesum.Value).ToString("F11"));
435
436     filestream.Write("0760340 "); filestream.WriteLine((Se76.Value / nuclidesum.Value).ToString("F11"));
437
438     filestream.Write("0770340 "); filestream.WriteLine((Se77.Value / nuclidesum.Value).ToString("F11"));
439
440     filestream.Write("0780340 "); filestream.WriteLine((Se78.Value / nuclidesum.Value).ToString("F11"));
441
442     filestream.Write("0800340 "); filestream.WriteLine((Se80.Value / nuclidesum.Value).ToString("F11"));
443
444     filestream.Write("0820340 "); filestream.WriteLine((Se82.Value / nuclidesum.Value).ToString("F11"));
445
446     filestream.Write("0790350 "); filestream.WriteLine((Br79.Value / nuclidesum.Value).ToString("F11"));
447
448     filestream.Write("0810350 "); filestream.WriteLine((Br81.Value / nuclidesum.Value).ToString("F11"));
449
450     filestream.Write("0780360 "); filestream.WriteLine((Kr78.Value / nuclidesum.Value).ToString("F11"));
451
452     filestream.Write("0800360 "); filestream.WriteLine((Kr80.Value / nuclidesum.Value).ToString("F11"));
453
454     filestream.Write("0820360 "); filestream.WriteLine((Kr82.Value / nuclidesum.Value).ToString("F11"));
455
456     filestream.Write("0830360 "); filestream.WriteLine((Kr83.Value / nuclidesum.Value).ToString("F11"));
457
458     filestream.Write("0840360 "); filestream.WriteLine((Kr84.Value / nuclidesum.Value).ToString("F11"));
459
460     filestream.Write("0860360 "); filestream.WriteLine((Kr86.Value / nuclidesum.Value).ToString("F11"));
461
462     filestream.Write("0850370 "); filestream.WriteLine((Rb85.Value / nuclidesum.Value).ToString("F11"));
463
464     filestream.Write("0870370 "); filestream.WriteLine((Rb87.Value / nuclidesum.Value).ToString("F11"));
465
466     filestream.Write("0840380 "); filestream.WriteLine((Sr84.Value / nuclidesum.Value).ToString("F11"));
467
468     filestream.Write("0860380 "); filestream.WriteLine((Sr86.Value / nuclidesum.Value).ToString("F11"));
469
470     filestream.Write("0870380 "); filestream.WriteLine((Sr87.Value / nuclidesum.Value).ToString("F11"));
471
472     filestream.Write("0880380 "); filestream.WriteLine((Sr88.Value / nuclidesum.Value).ToString("F11"));
473
474     filestream.Write("0890390 "); filestream.WriteLine((Y89.Value / nuclidesum.Value).ToString("F11"));
475
476     filestream.Write("0900400 "); filestream.WriteLine((Zr90.Value / nuclidesum.Value).ToString("F11"));
477
478     filestream.Write("0910400 "); filestream.WriteLine((Zr91.Value / nuclidesum.Value).ToString("F11"));
479
480     filestream.Write("0920400 "); filestream.WriteLine((Zr92.Value / nuclidesum.Value).ToString("F11"));
481
482     filestream.Write("0940400 "); filestream.WriteLine((Zr94.Value / nuclidesum.Value).ToString("F11"));
483
484     filestream.Write("0960400 "); filestream.WriteLine((Zr96.Value / nuclidesum.Value).ToString("F11"));
485
486     filestream.Write("0930410 "); filestream.WriteLine((Nb93.Value / nuclidesum.Value).ToString("F11"));
487
488     filestream.Write("0920420 "); filestream.WriteLine((Mo92.Value / nuclidesum.Value).ToString("F11"));
489
490     filestream.Write("0940420 "); filestream.WriteLine((Mo94.Value / nuclidesum.Value).ToString("F11"));

```



```

491
492     filestream.Write("0950420 "); filestream.WriteLine((Mo95.Value / nuclidesum.Value).ToString("F11"));
493
494     filestream.Write("0960420 "); filestream.WriteLine((Mo96.Value / nuclidesum.Value).ToString("F11"));
495
496     filestream.Write("0970420 "); filestream.WriteLine((Mo97.Value / nuclidesum.Value).ToString("F11"));
497
498     filestream.Write("0980420 "); filestream.WriteLine((Mo98.Value / nuclidesum.Value).ToString("F11"));
499
500     filestream.Write("1000420 "); filestream.WriteLine((Mo100.Value / nuclidesum.Value).ToString("F11"));
501
502     filestream.Write("0960440 "); filestream.WriteLine((Ru96.Value / nuclidesum.Value).ToString("F11"));
503
504     filestream.Write("0980440 "); filestream.WriteLine((Ru98.Value / nuclidesum.Value).ToString("F11"));
505
506     filestream.Write("0990440 "); filestream.WriteLine((Ru99.Value / nuclidesum.Value).ToString("F11"));
507
508     filestream.Write("1000440 "); filestream.WriteLine((Ru100.Value / nuclidesum.Value).ToString("F11"));
509
510     filestream.Write("1010440 "); filestream.WriteLine((Ru101.Value / nuclidesum.Value).ToString("F11"));
511
512     filestream.Write("1020440 "); filestream.WriteLine((Ru102.Value / nuclidesum.Value).ToString("F11"));
513
514     filestream.Write("1040440 "); filestream.WriteLine((Ru104.Value / nuclidesum.Value).ToString("F11"));
515
516     filestream.Write("1030450 "); filestream.WriteLine((Rh103.Value / nuclidesum.Value).ToString("F11"));
517
518     filestream.Write("1020460 "); filestream.WriteLine((Pd102.Value / nuclidesum.Value).ToString("F11"));
519
520     filestream.Write("1040460 "); filestream.WriteLine((Pd104.Value / nuclidesum.Value).ToString("F11"));
521
522     filestream.Write("1050460 "); filestream.WriteLine((Pd105.Value / nuclidesum.Value).ToString("F11"));
523
524     filestream.Write("1060460 "); filestream.WriteLine((Pd106.Value / nuclidesum.Value).ToString("F11"));
525
526     filestream.Write("1080460 "); filestream.WriteLine((Pd108.Value / nuclidesum.Value).ToString("F11"));
527
528     filestream.Write("1100460 "); filestream.WriteLine((Pd110.Value / nuclidesum.Value).ToString("F11"));
529
530     filestream.Write("1070470 "); filestream.WriteLine((Ag107.Value / nuclidesum.Value).ToString("F11"));
531
532     filestream.Write("1090470 "); filestream.WriteLine((Ag109.Value / nuclidesum.Value).ToString("F11"));
533
534     filestream.Write("1060480 "); filestream.WriteLine((Cd106.Value / nuclidesum.Value).ToString("F11"));
535
536     filestream.Write("1080480 "); filestream.WriteLine((Cd108.Value / nuclidesum.Value).ToString("F11"));
537
538     filestream.Write("1100480 "); filestream.WriteLine((Cd110.Value / nuclidesum.Value).ToString("F11"));
539
540     filestream.Write("1110480 "); filestream.WriteLine((Cd111.Value / nuclidesum.Value).ToString("F11"));
541
542     filestream.Write("1120480 "); filestream.WriteLine((Cd112.Value / nuclidesum.Value).ToString("F11"));
543
544     filestream.Write("1130480 "); filestream.WriteLine((Cd113.Value / nuclidesum.Value).ToString("F11"));
545
546     filestream.Write("1140480 "); filestream.WriteLine((Cd114.Value / nuclidesum.Value).ToString("F11"));
547
548     filestream.Write("1160480 "); filestream.WriteLine((Cd116.Value / nuclidesum.Value).ToString("F11"));
549
550     filestream.Write("1130490 "); filestream.WriteLine((In113.Value / nuclidesum.Value).ToString("F11"));
551
552     filestream.Write("1150490 "); filestream.WriteLine((In115.Value / nuclidesum.Value).ToString("F11"));
553
554     filestream.Write("1120500 "); filestream.WriteLine((Sn112.Value / nuclidesum.Value).ToString("F11"));
555
556     filestream.Write("1140500 "); filestream.WriteLine((Sn114.Value / nuclidesum.Value).ToString("F11"));
557
558     filestream.Write("1150500 "); filestream.WriteLine((Sn115.Value / nuclidesum.Value).ToString("F11"));
559
560     filestream.Write("1160500 "); filestream.WriteLine((Sn116.Value / nuclidesum.Value).ToString("F11"));

```

```

561
562      filestream.Write("1170500 "); filestream.WriteLine((Sn117.Value / nuclidesum.Value).ToString("F11"));
563
564      filestream.Write("1180500 "); filestream.WriteLine((Sn118.Value / nuclidesum.Value).ToString("F11"));
565
566      filestream.Write("1190500 "); filestream.WriteLine((Sn119.Value / nuclidesum.Value).ToString("F11"));
567
568      filestream.Write("1200500 "); filestream.WriteLine((Sn120.Value / nuclidesum.Value).ToString("F11"));
569
570      filestream.Write("1220500 "); filestream.WriteLine((Sn122.Value / nuclidesum.Value).ToString("F11"));
571
572      filestream.Write("1240500 "); filestream.WriteLine((Sn124.Value / nuclidesum.Value).ToString("F11"));
573
574      filestream.Write("1210510 "); filestream.WriteLine((Sb121.Value / nuclidesum.Value).ToString("F11"));
575
576      filestream.Write("1230510 "); filestream.WriteLine((Sb123.Value / nuclidesum.Value).ToString("F11"));
577
578      filestream.Write("1200520 "); filestream.WriteLine((Te120.Value / nuclidesum.Value).ToString("F11"));
579
580      filestream.Write("1220520 "); filestream.WriteLine((Te122.Value / nuclidesum.Value).ToString("F11"));
581
582      filestream.Write("1230520 "); filestream.WriteLine((Te123.Value / nuclidesum.Value).ToString("F11"));
583
584      filestream.Write("1240520 "); filestream.WriteLine((Te124.Value / nuclidesum.Value).ToString("F11"));
585
586      filestream.Write("1250520 "); filestream.WriteLine((Te125.Value / nuclidesum.Value).ToString("F11"));
587
588      filestream.Write("1260520 "); filestream.WriteLine((Te126.Value / nuclidesum.Value).ToString("F11"));
589
590      filestream.Write("1280520 "); filestream.WriteLine((Te128.Value / nuclidesum.Value).ToString("F11"));
591
592      filestream.Write("1300520 "); filestream.WriteLine((Te130.Value / nuclidesum.Value).ToString("F11"));
593
594      filestream.Write("1270530 "); filestream.WriteLine((I127.Value / nuclidesum.Value).ToString("F11"));
595
596      filestream.Write("1240540 "); filestream.WriteLine((Xe124.Value / nuclidesum.Value).ToString("F11"));
597
598      filestream.Write("1260540 "); filestream.WriteLine((Xe126.Value / nuclidesum.Value).ToString("F11"));
599
600      filestream.Write("1280540 "); filestream.WriteLine((Xe128.Value / nuclidesum.Value).ToString("F11"));
601
602      filestream.Write("1290540 "); filestream.WriteLine((Xe129.Value / nuclidesum.Value).ToString("F11"));
603
604      filestream.Write("1300540 "); filestream.WriteLine((Xe130.Value / nuclidesum.Value).ToString("F11"));
605
606      filestream.Write("1310540 "); filestream.WriteLine((Xe131.Value / nuclidesum.Value).ToString("F11"));
607
608      filestream.Write("1320540 "); filestream.WriteLine((Xe132.Value / nuclidesum.Value).ToString("F11"));
609
610      filestream.Write("1340540 "); filestream.WriteLine((Xe134.Value / nuclidesum.Value).ToString("F11"));
611
612      filestream.Write("1360540 "); filestream.WriteLine((Xe136.Value / nuclidesum.Value).ToString("F11"));
613
614      filestream.Write("1330550 "); filestream.WriteLine((Cs133.Value / nuclidesum.Value).ToString("F11"));
615
616      filestream.Write("1300560 "); filestream.WriteLine((Ba130.Value / nuclidesum.Value).ToString("F11"));
617
618      filestream.Write("1320560 "); filestream.WriteLine((Ba132.Value / nuclidesum.Value).ToString("F11"));
619
620      filestream.Write("1340560 "); filestream.WriteLine((Ba134.Value / nuclidesum.Value).ToString("F11"));
621
622      filestream.Write("1350560 "); filestream.WriteLine((Ba135.Value / nuclidesum.Value).ToString("F11"));
623
624      filestream.Write("1360560 "); filestream.WriteLine((Ba136.Value / nuclidesum.Value).ToString("F11"));
625
626      filestream.Write("1370560 "); filestream.WriteLine((Ba137.Value / nuclidesum.Value).ToString("F11"));
627
628      filestream.Write("1380560 "); filestream.WriteLine((Ba138.Value / nuclidesum.Value).ToString("F11"));
629
630      filestream.Write("1380570 "); filestream.WriteLine((La138.Value / nuclidesum.Value).ToString("F11"));

```

```

631
632     filestream.Write("1390570 "); filestream.WriteLine((La139.Value / nuclidesum.Value).ToString("F11"));
633
634     filestream.Write("1360580 "); filestream.WriteLine((Ce136.Value / nuclidesum.Value).ToString("F11"));
635
636     filestream.Write("1380580 "); filestream.WriteLine((Ce138.Value / nuclidesum.Value).ToString("F11"));
637
638     filestream.Write("1400580 "); filestream.WriteLine((Ce140.Value / nuclidesum.Value).ToString("F11"));
639
640     filestream.Write("1420580 "); filestream.WriteLine((Ce142.Value / nuclidesum.Value).ToString("F11"));
641
642     filestream.Write("1410590 "); filestream.WriteLine((Pr141.Value / nuclidesum.Value).ToString("F11"));
643
644     filestream.Write("1420600 "); filestream.WriteLine((Nd142.Value / nuclidesum.Value).ToString("F11"));
645
646     filestream.Write("1430600 "); filestream.WriteLine((Nd143.Value / nuclidesum.Value).ToString("F11"));
647
648     filestream.Write("1440600 "); filestream.WriteLine((Nd144.Value / nuclidesum.Value).ToString("F11"));
649
650     filestream.Write("1450600 "); filestream.WriteLine((Nd145.Value / nuclidesum.Value).ToString("F11"));
651
652     filestream.Write("1460600 "); filestream.WriteLine((Nd146.Value / nuclidesum.Value).ToString("F11"));
653
654     filestream.Write("1480600 "); filestream.WriteLine((Nd148.Value / nuclidesum.Value).ToString("F11"));
655
656     filestream.Write("1500600 "); filestream.WriteLine((Nd150.Value / nuclidesum.Value).ToString("F11"));
657
658     filestream.Write("1440620 "); filestream.WriteLine((Sm144.Value / nuclidesum.Value).ToString("F11"));
659
660     filestream.Write("1470620 "); filestream.WriteLine((Sm147.Value / nuclidesum.Value).ToString("F11"));
661
662     filestream.Write("1480620 "); filestream.WriteLine((Sm148.Value / nuclidesum.Value).ToString("F11"));
663
664     filestream.Write("1490620 "); filestream.WriteLine((Sm149.Value / nuclidesum.Value).ToString("F11"));
665
666     filestream.Write("1500620 "); filestream.WriteLine((Sm150.Value / nuclidesum.Value).ToString("F11"));
667
668     filestream.Write("1520620 "); filestream.WriteLine((Sm152.Value / nuclidesum.Value).ToString("F11"));
669
670     filestream.Write("1540620 "); filestream.WriteLine((Sm154.Value / nuclidesum.Value).ToString("F11"));
671
672     filestream.Write("1510630 "); filestream.WriteLine((Eu151.Value / nuclidesum.Value).ToString("F11"));
673
674     filestream.Write("1530630 "); filestream.WriteLine((Eu153.Value / nuclidesum.Value).ToString("F11"));
675
676     filestream.Write("1520640 "); filestream.WriteLine((Gd152.Value / nuclidesum.Value).ToString("F11"));
677
678     filestream.Write("1540640 "); filestream.WriteLine((Gd154.Value / nuclidesum.Value).ToString("F11"));
679
680     filestream.Write("1550640 "); filestream.WriteLine((Gd155.Value / nuclidesum.Value).ToString("F11"));
681
682     filestream.Write("1560640 "); filestream.WriteLine((Gd156.Value / nuclidesum.Value).ToString("F11"));
683
684     filestream.Write("1570640 "); filestream.WriteLine((Gd157.Value / nuclidesum.Value).ToString("F11"));
685
686     filestream.Write("1580640 "); filestream.WriteLine((Gd158.Value / nuclidesum.Value).ToString("F11"));
687
688     filestream.Write("1600640 "); filestream.WriteLine((Gd160.Value / nuclidesum.Value).ToString("F11"));
689
690     filestream.Write("1590650 "); filestream.WriteLine((Tb159.Value / nuclidesum.Value).ToString("F11"));
691
692     filestream.Write("1560660 "); filestream.WriteLine((Dy156.Value / nuclidesum.Value).ToString("F11"));
693
694     filestream.Write("1580660 "); filestream.WriteLine((Dy158.Value / nuclidesum.Value).ToString("F11"));
695
696     filestream.Write("1600660 "); filestream.WriteLine((Dy160.Value / nuclidesum.Value).ToString("F11"));
697
698     filestream.Write("1610660 "); filestream.WriteLine((Dy161.Value / nuclidesum.Value).ToString("F11"));
699
700     filestream.Write("1620660 "); filestream.WriteLine((Dy162.Value / nuclidesum.Value).ToString("F11"));

```

```

701
702     filestream.Write("1630660 "); filestream.WriteLine((Dy163.Value / nuclidesum.Value).ToString("F11"));
703
704     filestream.Write("1640660 "); filestream.WriteLine((Dy164.Value / nuclidesum.Value).ToString("F11"));
705
706     filestream.Write("1650670 "); filestream.WriteLine((Ho165.Value / nuclidesum.Value).ToString("F11"));
707
708     filestream.Write("1620680 "); filestream.WriteLine((Er162.Value / nuclidesum.Value).ToString("F11"));
709
710     filestream.Write("1640680 "); filestream.WriteLine((Er164.Value / nuclidesum.Value).ToString("F11"));
711
712     filestream.Write("1660680 "); filestream.WriteLine((Er166.Value / nuclidesum.Value).ToString("F11"));
713
714     filestream.Write("1670680 "); filestream.WriteLine((Er167.Value / nuclidesum.Value).ToString("F11"));
715
716     filestream.Write("1680680 "); filestream.WriteLine((Er168.Value / nuclidesum.Value).ToString("F11"));
717
718     filestream.Write("1700680 "); filestream.WriteLine((Er170.Value / nuclidesum.Value).ToString("F11"));
719
720     filestream.Write("1690690 "); filestream.WriteLine((Tm169.Value / nuclidesum.Value).ToString("F11"));
721
722     filestream.Write("1680700 "); filestream.WriteLine((Yb168.Value / nuclidesum.Value).ToString("F11"));
723
724     filestream.Write("1700700 "); filestream.WriteLine((Yb170.Value / nuclidesum.Value).ToString("F11"));
725
726     filestream.Write("1710700 "); filestream.WriteLine((Yb171.Value / nuclidesum.Value).ToString("F11"));
727
728     filestream.Write("1720700 "); filestream.WriteLine((Yb172.Value / nuclidesum.Value).ToString("F11"));
729
730     filestream.Write("1730700 "); filestream.WriteLine((Yb173.Value / nuclidesum.Value).ToString("F11"));
731
732     filestream.Write("1740700 "); filestream.WriteLine((Yb174.Value / nuclidesum.Value).ToString("F11"));
733
734     filestream.Write("1760700 "); filestream.WriteLine((Yb176.Value / nuclidesum.Value).ToString("F11"));
735
736     filestream.Write("1750710 "); filestream.WriteLine((Lu175.Value / nuclidesum.Value).ToString("F11"));
737
738     filestream.Write("1760710 "); filestream.WriteLine((Lu176.Value / nuclidesum.Value).ToString("F11"));
739
740     filestream.Write("1740720 "); filestream.WriteLine((Hf174.Value / nuclidesum.Value).ToString("F11"));
741
742     filestream.Write("1760720 "); filestream.WriteLine((Hf176.Value / nuclidesum.Value).ToString("F11"));
743
744     filestream.Write("1770720 "); filestream.WriteLine((Hf177.Value / nuclidesum.Value).ToString("F11"));
745
746     filestream.Write("1780720 "); filestream.WriteLine((Hf178.Value / nuclidesum.Value).ToString("F11"));
747
748     filestream.Write("1790720 "); filestream.WriteLine((Hf179.Value / nuclidesum.Value).ToString("F11"));
749
750     filestream.Write("1800720 "); filestream.WriteLine((Hf180.Value / nuclidesum.Value).ToString("F11"));
751
752     filestream.Write("1800730 "); filestream.WriteLine((Ta180.Value / nuclidesum.Value).ToString("F11"));
753
754     filestream.Write("1810730 "); filestream.WriteLine((Ta181.Value / nuclidesum.Value).ToString("F11"));
755
756     filestream.Write("1800740 "); filestream.WriteLine((W180.Value / nuclidesum.Value).ToString("F11"));
757
758     filestream.Write("1820740 "); filestream.WriteLine((W182.Value / nuclidesum.Value).ToString("F11"));
759
760     filestream.Write("1830740 "); filestream.WriteLine((W183.Value / nuclidesum.Value).ToString("F11"));
761
762     filestream.Write("1840740 "); filestream.WriteLine((W184.Value / nuclidesum.Value).ToString("F11"));
763
764     filestream.Write("1860740 "); filestream.WriteLine((W186.Value / nuclidesum.Value).ToString("F11"));
765
766     filestream.Write("1850750 "); filestream.WriteLine((Re185.Value / nuclidesum.Value).ToString("F11"));
767
768     filestream.Write("1870750 "); filestream.WriteLine((Re187.Value / nuclidesum.Value).ToString("F11"));
769
770     filestream.Write("1840760 "); filestream.WriteLine((Os184.Value / nuclidesum.Value).ToString("F11"));

```

```

771
772     filestream.Write("1860760 "); filestream.WriteLine((Os186.Value / nuclidesum.Value).ToString("F11"));
773
774     filestream.Write("1870760 "); filestream.WriteLine((Os187.Value / nuclidesum.Value).ToString("F11"));
775
776     filestream.Write("1880760 "); filestream.WriteLine((Os188.Value / nuclidesum.Value).ToString("F11"));
777
778     filestream.Write("1890760 "); filestream.WriteLine((Os189.Value / nuclidesum.Value).ToString("F11"));
779
780     filestream.Write("1900760 "); filestream.WriteLine((Os190.Value / nuclidesum.Value).ToString("F11"));
781
782     filestream.Write("1920760 "); filestream.WriteLine((Os192.Value / nuclidesum.Value).ToString("F11"));
783
784     filestream.Write("1910770 "); filestream.WriteLine((Ir191.Value / nuclidesum.Value).ToString("F11"));
785
786     filestream.Write("1930770 "); filestream.WriteLine((Ir193.Value / nuclidesum.Value).ToString("F11"));
787
788     filestream.Write("1900780 "); filestream.WriteLine((Pt190.Value / nuclidesum.Value).ToString("F11"));
789
790     filestream.Write("1920780 "); filestream.WriteLine((Pt192.Value / nuclidesum.Value).ToString("F11"));
791
792     filestream.Write("1940780 "); filestream.WriteLine((Pt194.Value / nuclidesum.Value).ToString("F11"));
793
794     filestream.Write("1950780 "); filestream.WriteLine((Pt195.Value / nuclidesum.Value).ToString("F11"));
795
796     filestream.Write("1960780 "); filestream.WriteLine((Pt196.Value / nuclidesum.Value).ToString("F11"));
797
798     filestream.Write("1980780 "); filestream.WriteLine((Pt198.Value / nuclidesum.Value).ToString("F11"));
799
800     filestream.Write("1970790 "); filestream.WriteLine((Au197.Value / nuclidesum.Value).ToString("F11"));
801
802     filestream.Write("1960800 "); filestream.WriteLine((Hg196.Value / nuclidesum.Value).ToString("F11"));
803
804     filestream.Write("1980800 "); filestream.WriteLine((Hg198.Value / nuclidesum.Value).ToString("F11"));
805
806     filestream.Write("1990800 "); filestream.WriteLine((Hg199.Value / nuclidesum.Value).ToString("F11"));
807
808     filestream.Write("2000800 "); filestream.WriteLine((Hg200.Value / nuclidesum.Value).ToString("F11"));
809
810     filestream.Write("2010800 "); filestream.WriteLine((Hg201.Value / nuclidesum.Value).ToString("F11"));
811
812     filestream.Write("2020800 "); filestream.WriteLine((Hg202.Value / nuclidesum.Value).ToString("F11"));
813
814     filestream.Write("2040800 "); filestream.WriteLine((Hg204.Value / nuclidesum.Value).ToString("F11"));
815
816     filestream.Write("2030810 "); filestream.WriteLine((Tl203.Value / nuclidesum.Value).ToString("F11"));
817
818     filestream.Write("2050810 "); filestream.WriteLine((Tl205.Value / nuclidesum.Value).ToString("F11"));
819
820     filestream.Write("2040820 "); filestream.WriteLine((Pb204.Value / nuclidesum.Value).ToString("F11"));
821
822     filestream.Write("2060820 "); filestream.WriteLine((Pb206.Value / nuclidesum.Value).ToString("F11"));
823
824     filestream.Write("2070820 "); filestream.WriteLine((Pb207.Value / nuclidesum.Value).ToString("F11"));
825
826     filestream.Write("2080820 "); filestream.WriteLine((Pb208.Value / nuclidesum.Value).ToString("F11"));
827
828
829
830
831     filestream.Close();
832 }
833
834 private void locate_Click(object sender, EventArgs e)
835 {
836     var filestream = new System.IO.StreamWriter("c:\\cinder\\" + titleTextBox.Text + "\\locate");
837     filestream.WriteLine("C:\\Cinder2008\\Data\\c90lib0742");
838     filestream.WriteLine("");
839     filestream.WriteLine("C:\\Cinder2008\\Data\\cinderogl.dat");
840     filestream.Close();

```

```

841     }
842
843     private void dateTimePicker1_ValueChanged_3(object sender, EventArgs e)
844     {
845
846     }
847
848     private void textBox3_TextChanged(object sender, EventArgs e)
849     {
850
851     }
852
853     private void label3_Click(object sender, EventArgs e)
854     {
855
856     }
857
858     private void waittime_ValueChanged(object sender, EventArgs e)
859     {
860
861     }
862
863     private void checkedListBox2_ItemCheck(object sender, ItemCheckEventArgs e)
864     {
865
866     }
867
868     private void checkedListBox2_ItemCheck_1(object sender, ItemCheckEventArgs e)
869     {
870
871     }
872
873     private void checkedListBox2_SelectedIndexChanged(object sender, EventArgs e)
874     {
875
876     }
877
878     private void pulse_Click(object sender, EventArgs e)
879     {
880         if (pulse.Checked)
881         {
882             energy.Hide();
883             energyvalue.Hide();
884             radtimelabel.Hide();
885             powerlabel.Show();
886             MJ.Show();
887             percentpowerlabel.Hide();
888             percentpower.Hide();
889             reactorpower.Hide();
890             reactorpowerlabel.Hide();
891             hourlabel.Hide();
892             hours.Hide();
893             minutelabel.Hide();
894             minutes.Hide();
895             secondlabel.Hide();
896             seconds.Hide();
897         }
898     }
899
900     private void ss_Click(object sender, EventArgs e)
901     {
902         if (ss.Checked)
903         {
904             radtimelabel.Show();
905             reactorpower.Show();
906             reactorpowerlabel.Show();
907             powerlabel.Hide();
908             MJ.Hide();
909             percentpower.Show();
910             percentpowerlabel.Show();

```

```

911         hourlabel.Show();
912         hours.Show();
913         minutelabel.Show();
914         minutes.Show();
915         secondlabel.Show();
916         seconds.Show();
917         energy.Show();
918         energyvalue.Show();
919     }
920 }
921
922 private void pulse_CheckedChanged(object sender, EventArgs e)
923 {
924
925 }
926
927 private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
928 {
929
930 }
931
932 private void reactorpower_ValueChanged(object sender, EventArgs e)
933 {
934     percentpower.Value = reactorpower.Value * 10000 / 239;
935     energyvalue.Value = reactorpower.Value * (hours.Value * 3600 + minutes.Value * 60 + seconds.Value);
936
937 }
938
939 private void percentpower_ValueChanged(object sender, EventArgs e)
940 {
941     reactorpower.Value = percentpower.Value * 239 / 10000;
942     energyvalue.Value = reactorpower.Value * (hours.Value * 3600 + minutes.Value * 60 + seconds.Value);
943
944 }
945
946 private void percentpowerlabel_Click(object sender, EventArgs e)
947 {
948
949 }
950
951 private void textBox1_TextChanged(object sender, EventArgs e)
952 {
953
954 }
955
956 private void label3_Click_1(object sender, EventArgs e)
957 {
958
959 }
960
961 private void numericUpDown4_ValueChanged(object sender, EventArgs e)
962 {
963     energyvalue.Value = reactorpower.Value * (hours.Value * 3600 + minutes.Value * 60 + seconds.Value);
964 }
965
966 private void hours_ValueChanged(object sender, EventArgs e)
967 {
968     energyvalue.Value = reactorpower.Value * (hours.Value * 3600 + minutes.Value * 60 + seconds.Value);
969
970 }
971
972 private void minutes_ValueChanged(object sender, EventArgs e)
973 {
974     energyvalue.Value = reactorpower.Value * (hours.Value * 3600 + minutes.Value * 60 + seconds.Value);
975
976 }
977
978 private void seconds_ValueChanged(object sender, EventArgs e)
979 {
980     energyvalue.Value = reactorpower.Value * (hours.Value * 3600 + minutes.Value * 60 + seconds.Value);

```

```

981
982     }
983
984     private void reactorpowerlabel_Click(object sender, EventArgs e)
985     {
986
987     }
988
989     private void pulsetimelabel_Click(object sender, EventArgs e)
990     {
991
992     }
993
994     private void vScrollBar1_Scroll(object sender, ScrollEventArgs e)
995     {
996
997     }
998
999     private void checkBox1_CheckedChanged(object sender, EventArgs e)
1000    {
1001
1002    }
1003
1004    private void label11_Click(object sender, EventArgs e)
1005    {
1006
1007    }
1008
1009    private void panel2_Paint(object sender, PaintEventArgs e)
1010    {
1011
1012    }
1013
1014    private void numericUpDown24_ValueChanged(object sender, EventArgs e)
1015    {
1016
1017    }
1018
1019    private void panel3_Paint(object sender, PaintEventArgs e)
1020    {
1021
1022    }
1023
1024    private void DREAD_Load(object sender, EventArgs e)
1025    {
1026
1027    }
1028
1029    private void groupBox1_Enter(object sender, EventArgs e)
1030    {
1031
1032    }
1033
1034    private void nickelgram5mil_ValueChanged(object sender, EventArgs e)
1035    {
1036        AtomCalc();
1037    }
1038
1039    private void nickel10_CheckedChanged(object sender, EventArgs e)
1040    {
1041        if (nickel10.Checked)
1042        {
1043            nickelgram10mil.Show();
1044        }
1045        else
1046        {
1047            nickelgram10mil.Value = 0;
1048            nickelgram10mil.Hide();
1049        }
1050    }

```



```

1051
1052     private void nickelgram10mil_ValueChanged(object sender, EventArgs e)
1053     {
1054
1055         AtomCalc();
1056     }
1057
1058     private void checkBox8_CheckedChanged(object sender, EventArgs e)
1059     {
1060         if (copper63.Checked)
1061         {
1062             copper63gram.Show();
1063         }
1064         else
1065         {
1066             copper63gram.Value = 0;
1067             copper63gram.Hide();
1068         }
1069     }
1070
1071     private void checkBox9_CheckedChanged(object sender, EventArgs e)
1072     {
1073         if (cardboard.Checked)
1074         {
1075             cardboardgrams.Show();
1076         }
1077         else
1078         {
1079             cardboardgrams.Value = 0;
1080             cardboardgrams.Hide();
1081         }
1082     }
1083
1084
1085     private void checkBox10_CheckedChanged(object sender, EventArgs e)
1086     {
1087         if (pcbelectronics.Checked)
1088         {
1089             pcbelectronicsgrams.Show();
1090         }
1091         else
1092         {
1093             pcbelectronicsgrams.Value = 0;
1094             pcbelectronicsgrams.Hide();
1095         }
1096     }
1097
1098     private void checkBox11_CheckedChanged(object sender, EventArgs e)
1099     {
1100         if (circuitboard.Checked)
1101         {
1102             circuitboardgrams.Show();
1103         }
1104         else
1105         {
1106             circuitboardgrams.Value = 0;
1107             circuitboardgrams.Hide();
1108         }
1109     }
1110
1111     private void SS316_CheckedChanged(object sender, EventArgs e)
1112     {
1113         if (SS316.Checked)
1114         {
1115             SS316gram.Show();
1116         }
1117         else
1118         {
1119             SS316gram.Value = 0;
1120             SS316gram.Hide();

```

```

1121     }
1122 }
1123
1124 private void al6061_CheckedChanged(object sender, EventArgs e)
1125 {
1126     if (al6061.Checked)
1127     {
1128         Al6061gram.Show();
1129     }
1130     else
1131     {
1132         Al6061gram.Value = 0;
1133         Al6061gram.Hide();
1134     }
1135 }
1136
1137 private void tld_CheckedChanged(object sender, EventArgs e) //10 top pieces weigh 3.4078 g
1138 {
1139     if (tld.Checked)
1140     {
1141         tldgram.Show();
1142     }
1143     else
1144     {
1145         tldgram.Value = 0;
1146         tldgram.Hide();
1147     }
1148 }
1149
1150 private void sulfurlarge_CheckedChanged(object sender, EventArgs e)
1151 {
1152     if (sulfurlarge.Checked)
1153     {
1154         sulfurgramlarge.Show();
1155     }
1156     else
1157     {
1158         sulfurgramlarge.Value = 0;
1159         sulfurgramlarge.Hide();
1160     }
1161 }
1162
1163 private void sulfur_CheckedChanged(object sender, EventArgs e)
1164 {
1165     if (sulfur.Checked)
1166     {
1167         sulfurgramstandard.Show();
1168     }
1169     else
1170     {
1171         sulfurgramstandard.Value = 0;
1172         sulfurgramstandard.Hide();
1173     }
1174 }
1175
1176 private void checkBox7_CheckedChanged(object sender, EventArgs e)
1177 {
1178     if (poly.Checked)
1179     {
1180         polygram.Show();
1181     }
1182     else
1183     {
1184         polygram.Value = 0;
1185         polygram.Hide();
1186     }
1187 }
1188
1189 private void Ni60_ValueChanged(object sender, EventArgs e)
1190 {

```

```

1191
1192     }
1193
1194     private void ss_CheckedChanged(object sender, EventArgs e)
1195     {
1196
1197     }
1198
1199     private void MJ_ValueChanged(object sender, EventArgs e)
1200     {
1201
1202     }
1203
1204
1205     private void nickel5_CheckStateChanged(object sender, EventArgs e)
1206     {
1207         if (nickel5.Checked)
1208         {
1209             nickelgram5mil.Show();
1210         }
1211         else
1212         {
1213             nickelgram5mil.Value = 0;
1214             nickelgram5mil.Hide();
1215         }
1216     }
1217
1218
1219     private void SS316gram_ValueChanged(object sender, EventArgs e)
1220     {
1221         AtomCalc();
1222     }
1223
1224     public void AtomCalc()
1225     {
1226         //calc value from inputs for isotopes = total grams from all entries * weight percent for isotope of element
1227         * Avagadro's number * 1e-24 to convert cm^3 to barn-cm for cinder volume / isotopic mass
1228
1229         H1.Value = ((H.Value + circuitboardgrams.Value * 0.01776m + pcbelectronicsgrams.Value * .01456m +
1230         polygram.Value * .143736m + pvcgram.Value * .048402m + cardboardgrams.Value * .059m) * .99977m * 6.0221413e-1m
1231         / 1m);
1232         H2.Value = ((H.Value + circuitboardgrams.Value * 0.01776m + pcbelectronicsgrams.Value * .01456m +
1233         polygram.Value * .143736m + pvcgram.Value * .048402m + cardboardgrams.Value * .059m) * .00023m * 6.0221413e-1m
1234         / 2m);
1235         He3.Value = ((He.Value) * .000001m * 6.0221413e-1m / 3m);
1236         He4.Value = ((He.Value) * .999999m * 6.0221413e-1m / 4m);
1237
1238         Li6.Value = ((Li.Value) * .065785m * 6.0221413e-1m / 6m);
1239         Li7.Value = ((Li.Value) * .934215m * 6.0221413e-1m / 7m);
1240
1241         Be9.Value = ((Be.Value) * 1.00000m * 6.0221413e-1m / 9m);
1242
1243         B10.Value = ((B.Value) * .184309m * 6.0221413e-1m / 10m);
1244         B11.Value = ((B.Value) * .815691m * 6.0221413e-1m / 11m);
1245
1246         C12.Value = ((C.Value + SS316gram.Value * 0.0008m + circuitboardgrams.Value * 0.18192m +
1247         pcbelectronicsgrams.Value * .31539m + polygram.Value * .856164m + pvcgram.Value * .384141m + teflongram.Value *
1248         .24m + cardboardgrams.Value * .44m) * .988416m * 6.0221413e-1m / 12m);
1249         C13.Value = ((C.Value + SS316gram.Value * 0.0008m + circuitboardgrams.Value * 0.18192m +
1250         pcbelectronicsgrams.Value * .31539m + polygram.Value * .856164m + pvcgram.Value * .384141m + teflongram.Value *
1251         .24m + cardboardgrams.Value * .44m) * .011584m * 6.0221413e-1m / 13m);
1252
1253         N14.Value = ((N.Value + SS316gram.Value * 0.001m + cardboardgrams.Value * .003m) * .996102m *
1254         6.0221413e-1m / 14m);
1255         N15.Value = ((N.Value + SS316gram.Value * 0.001m + cardboardgrams.Value * .003m) * .003898m *
1256         6.0221413e-1m / 15m);
1257

```

1249 O16.Value = ((O.Value + circuitboardgrams.Value * 0.33856m + pcbelectronicsgrams.Value * .10074m
 + cardboardgrams.Value * .446m) * .997290m * 6.0221413e-1m / 16m);
 1250 O17.Value = ((O.Value + circuitboardgrams.Value * 0.33856m + pcbelectronicsgrams.Value * .10074m
 + cardboardgrams.Value * .446m) * .000404m * 6.0221413e-1m / 17m);
 1251 O18.Value = ((O.Value + circuitboardgrams.Value * 0.33856m + pcbelectronicsgrams.Value * .10074m
 + cardboardgrams.Value * .446m) * .002306m * 6.0221413e-1m / 18m);
 1252
 1253 F19.Value = ((F.Value + tldgram.Value * 0.0139m + teflongram.Value * .76m) * 1.00000m * 6.0221413e-
 1m / 19m);
 1254
 1255 Ne20.Value = ((Ne.Value) * .896388m * 6.0221413e-1m / 20m);
 1256 Ne21.Value = ((Ne.Value) * .002008m * 6.0221413e-1m / 21m);
 1257 Ne22.Value = ((Ne.Value) * .100803m * 6.0221413e-1m / 22m);
 1258
 1259 Na23.Value = ((Na.Value) * 1.00000m * 6.0221413e-1m / 23m);
 1260
 1261 Mg24.Value = ((Mg.Value + Al6061gram.Value * 0.01m + circuitboardgrams.Value * 0.00145m +
 pcbelectronicsgrams.Value * .00067m) * .779500m * 6.0221413e-1m / 24m);
 1262 Mg25.Value = ((Mg.Value + Al6061gram.Value * 0.01m + circuitboardgrams.Value * 0.00145m +
 pcbelectronicsgrams.Value * .00067m) * .102801m * 6.0221413e-1m / 25m);
 1263 Mg26.Value = ((Mg.Value + Al6061gram.Value * 0.01m + circuitboardgrams.Value * 0.00145m +
 pcbelectronicsgrams.Value * .00067m) * .117699m * 6.0221413e-1m / 26m);
 1264
 1265 Al27.Value = ((Al.Value + Al6061gram.Value * 0.96m + circuitboardgrams.Value * 0.04650m +
 pcbelectronicsgrams.Value * .04943m + tldgram.Value * 0.68773m) * 1.00000m * 6.0221413e-1m / 27m);
 1266
 1267 Si28.Value = ((Si.Value + Al6061gram.Value * 0.006m + SS316gram.Value * 0.0075m +
 circuitboardgrams.Value * 0.26201m + pcbelectronicsgrams.Value * .07890m) * .918665m * 6.0221413e-1m / 28m);
 1268 Si29.Value = ((Si.Value + Al6061gram.Value * 0.006m + SS316gram.Value * 0.0075m +
 circuitboardgrams.Value * 0.26201m + pcbelectronicsgrams.Value * .07890m) * .048336m * 6.0221413e-1m / 29m);
 1269 Si30.Value = ((Si.Value + Al6061gram.Value * 0.006m + SS316gram.Value * 0.0075m +
 circuitboardgrams.Value * 0.26201m + pcbelectronicsgrams.Value * .07890m) * .032999m * 6.0221413e-1m / 30m);
 1270
 1271 P31.Value = ((P.Value + SS316gram.Value * 0.00045m) * 1.00000m * 6.0221413e-1m / 31m);
 1272
 1273 S32.Value = ((sulfurgramstandard.Value * 0.250m + sulfurgramlarge.Value * 2.25m + S.Value +
 SS316gram.Value * 0.0003m + cardboardgrams.Value * .002m) * .9471530m * 6.0221413e-1m / 32m);
 1274 S33.Value = ((sulfurgramstandard.Value * 0.250m + sulfurgramlarge.Value * 2.25m + S.Value +
 SS316gram.Value * 0.0003m + cardboardgrams.Value * .002m) * .0077120m * 6.0221413e-1m / 33m);
 1275 S34.Value = ((sulfurgramstandard.Value * 0.250m + sulfurgramlarge.Value * 2.25m + S.Value +
 SS316gram.Value * 0.0003m + cardboardgrams.Value * .002m) * .0016523m * 6.0221413e-1m / 34m);
 1276 S36.Value = ((sulfurgramstandard.Value * 0.250m + sulfurgramlarge.Value * 2.25m + S.Value +
 SS316gram.Value * 0.0003m + cardboardgrams.Value * .002m) * .0001120m * 6.0221413e-1m / 36m);
 1277
 1278 Cl35.Value = ((Cl.Value + pcbelectronicsgrams.Value * .17018m + pvcgram.Value * .567457m) *
 .747256m * 6.0221413e-1m / 35m);
 1279 Cl37.Value = ((Cl.Value + pcbelectronicsgrams.Value * .17018m + pvcgram.Value * .567457m) *
 .252744m * 6.0221413e-1m / 37m);
 1280
 1281 Ar36.Value = ((Ar.Value) * .003004m * 6.0221413e-1m / 36m);
 1282 Ar38.Value = ((Ar.Value) * .000598m * 6.0221413e-1m / 38m);
 1283 Ar40.Value = ((Ar.Value) * .996399m * 6.0221413e-1m / 40m);
 1284
 1285 K39.Value = ((K.Value) * .929371m * 6.0221413e-1m / 39m);
 1286 K40.Value = ((K.Value) * .000120m * 6.0221413e-1m / 40m);
 1287 K41.Value = ((K.Value) * .070510m * 6.0221413e-1m / 41m);
 1288
 1289 Ca40.Value = ((Ca.Value + tldgram.Value * 0.0148m + pcbelectronicsgrams.Value * .00024m +
 cardboardgrams.Value * .03m) * .966575m * 6.0221413e-1m / 40m);
 1290 Ca42.Value = ((Ca.Value + tldgram.Value * 0.0148m + pcbelectronicsgrams.Value * .00024m +
 cardboardgrams.Value * .03m) * .006773m * 6.0221413e-1m / 42m);
 1291 Ca43.Value = ((Ca.Value + tldgram.Value * 0.0148m + pcbelectronicsgrams.Value * .00024m +
 cardboardgrams.Value * .03m) * .001447m * 6.0221413e-1m / 43m);
 1292 Ca44.Value = ((Ca.Value + tldgram.Value * 0.0148m + pcbelectronicsgrams.Value * .00024m +
 cardboardgrams.Value * .03m) * .022921m * 6.0221413e-1m / 44m);
 1293 Ca46.Value = ((Ca.Value + tldgram.Value * 0.0148m + pcbelectronicsgrams.Value * .00024m +
 cardboardgrams.Value * .03m) * .000046m * 6.0221413e-1m / 46m);
 1294 Ca48.Value = ((Ca.Value + tldgram.Value * 0.0148m + pcbelectronicsgrams.Value * .00024m +
 cardboardgrams.Value * .03m) * .002237m * 6.0221413e-1m / 48m);
 1295

1296 Sc45.Value = ((Sc.Value) * 1.00000m * 6.0221413e-1m / 45m);
 1297
 1298 Ti46.Value = ((Ti.Value + Al6061gram.Value * 0.0015m + circuitboardgrams.Value * 0.00010m) *
 .079201m * 6.0221413e-1m / 46m);
 1299 Ti47.Value = ((Ti.Value + Al6061gram.Value * 0.0015m + circuitboardgrams.Value * 0.00010m) *
 .072978m * 6.0221413e-1m / 47m);
 1300 Ti48.Value = ((Ti.Value + Al6061gram.Value * 0.0015m + circuitboardgrams.Value * 0.00010m) *
 .738451m * 6.0221413e-1m / 48m);
 1301 Ti49.Value = ((Ti.Value + Al6061gram.Value * 0.0015m + circuitboardgrams.Value * 0.00010m) *
 .055322m * 6.0221413e-1m / 49m);
 1302 Ti50.Value = ((Ti.Value + Al6061gram.Value * 0.0015m + circuitboardgrams.Value * 0.00010m) *
 .054049m * 6.0221413e-1m / 50m);
 1303
 1304 V50.Value = ((V.Value) * .002451m * 6.0221413e-1m / 50m);
 1305 V51.Value = ((V.Value) * .997549m * 6.0221413e-1m / 51m);
 1306
 1307 Cr50.Value = ((Cr.Value + Al6061gram.Value * 0.003m + SS316gram.Value * 0.18m +
 circuitboardgrams.Value * 0.00020m) * .041737m * 6.0221413e-1m / 50m);
 1308 Cr52.Value = ((Cr.Value + Al6061gram.Value * 0.003m + SS316gram.Value * 0.18m +
 circuitboardgrams.Value * 0.00020m) * .836994m * 6.0221413e-1m / 52m);
 1309 Cr53.Value = ((Cr.Value + Al6061gram.Value * 0.003m + SS316gram.Value * 0.18m +
 circuitboardgrams.Value * 0.00020m) * .096736m * 6.0221413e-1m / 53m);
 1310 Cr54.Value = ((Cr.Value + Al6061gram.Value * 0.003m + SS316gram.Value * 0.18m +
 circuitboardgrams.Value * 0.00020m) * .024534m * 6.0221413e-1m / 54m);
 1311
 1312 Mn55.Value = ((Mn.Value + Al6061gram.Value * 0.0015m + SS316gram.Value * 0.02m +
 circuitboardgrams.Value * 0.00015m + pcbelectronicsgrams.Value * .00015m) * 1.00000m * 6.0221413e-1m / 55m);
 1313
 1314 Fe54.Value = ((Fe.Value + Al6061gram.Value * 0.007m + SS316gram.Value * 0.68m +
 circuitboardgrams.Value * 0.00035m + pcbelectronicsgrams.Value * .02674m) * .056456m * 6.0221413e-1m / 54m);
 1315 Fe56.Value = ((Fe.Value + Al6061gram.Value * 0.007m + SS316gram.Value * 0.68m +
 circuitboardgrams.Value * 0.00035m + pcbelectronicsgrams.Value * .02674m) * .919015m * 6.0221413e-1m / 56m);
 1316 Fe57.Value = ((Fe.Value + Al6061gram.Value * 0.007m + SS316gram.Value * 0.68m +
 circuitboardgrams.Value * 0.00035m + pcbelectronicsgrams.Value * .02674m) * .021604m * 6.0221413e-1m / 57m);
 1317 Fe58.Value = ((Fe.Value + Al6061gram.Value * 0.007m + SS316gram.Value * 0.68m +
 circuitboardgrams.Value * 0.00035m + pcbelectronicsgrams.Value * .02674m) * .002925m * 6.0221413e-1m / 58m);
 1318
 1319 Co59.Value = ((Co.Value + pcbelectronicsgrams.Value * .00850m) * 1.00000m * 6.0221413e-1m /
 59m);
 1320
 1321 Ni58.Value = ((nickelgram5mil.Value * 0.140m + nickelgram10mil.Value * 0.280m + Ni.Value +
 SS316gram.Value * 0.14m + pcbelectronicsgrams.Value * .01450m) * .671878m * 6.0221413e-1m / 58m);
 1322 Ni60.Value = ((nickelgram5mil.Value * 0.140m + nickelgram10mil.Value * 0.280m + Ni.Value +
 SS316gram.Value * 0.14m + pcbelectronicsgrams.Value * .01450m) * .267759m * 6.0221413e-1m / 60m);
 1323 Ni61.Value = ((nickelgram5mil.Value * 0.140m + nickelgram10mil.Value * 0.280m + Ni.Value +
 SS316gram.Value * 0.14m + pcbelectronicsgrams.Value * .01450m) * .011834m * 6.0221413e-1m / 61m);
 1324 Ni62.Value = ((nickelgram5mil.Value * 0.140m + nickelgram10mil.Value * 0.280m + Ni.Value +
 SS316gram.Value * 0.14m + pcbelectronicsgrams.Value * .01450m) * .038349m * 6.0221413e-1m / 62m);
 1325 Ni64.Value = ((nickelgram5mil.Value * 0.140m + nickelgram10mil.Value * 0.280m + Ni.Value +
 SS316gram.Value * 0.14m + pcbelectronicsgrams.Value * .01450m) * .010080m * 6.0221413e-1m / 64m);
 1326
 1327 Cu63.Value = ((Cu.Value + Al6061gram.Value * 0.003m + circuitboardgrams.Value * 0.05100m +
 pcbelectronicsgrams.Value * .15000m + copper63gram.Value * 1.0m/0.684792m) * .684792m * 6.0221413e-1m / 63m);
 1328 Cu65.Value = ((Cu.Value + Al6061gram.Value * 0.003m + circuitboardgrams.Value * 0.05100m +
 pcbelectronicsgrams.Value * .15000m) * .315208m * 6.0221413e-1m / 65m);
 1329
 1330 Zn64.Value = ((Zn.Value + Al6061gram.Value * 0.0025m) * .480805m * 6.0221413e-1m / 64m);
 1331 Zn66.Value = ((Zn.Value + Al6061gram.Value * 0.0025m) * .279625m * 6.0221413e-1m / 66m);
 1332 Zn67.Value = ((Zn.Value + Al6061gram.Value * 0.0025m) * .041357m * 6.0221413e-1m / 67m);
 1333 Zn68.Value = ((Zn.Value + Al6061gram.Value * 0.0025m) * .191688m * 6.0221413e-1m / 68m);
 1334 Zn70.Value = ((Zn.Value + Al6061gram.Value * 0.0025m) * .006524m * 6.0221413e-1m / 70m);
 1335
 1336 Ga69.Value = ((Ga.Value) * .594205m * 6.0221413e-1m / 69m);
 1337 Ga71.Value = ((Ga.Value) * .405795m * 6.0221413e-1m / 71m);
 1338
 1339 Ge70.Value = ((Ge.Value) * .198044m * 6.0221413e-1m / 70m);
 1340 Ge72.Value = ((Ge.Value) * .271834m * 6.0221413e-1m / 72m);
 1341 Ge73.Value = ((Ge.Value) * .077816m * 6.0221413e-1m / 73m);
 1342 Ge74.Value = ((Ge.Value) * .371501m * 6.0221413e-1m / 74m);
 1343 Ge76.Value = ((Ge.Value) * .080806m * 6.0221413e-1m / 76m);

1344
1345 As75.Value = ((As.Value) * 1.00000m * 6.0221413e-1m / 75m);
1346
1347 Se74.Value = ((Se.Value) * .008332m * 6.0221413e-1m / 74m);
1348 Se76.Value = ((Se.Value) * .090092m * 6.0221413e-1m / 76m);
1349 Se77.Value = ((Se.Value) * .074329m * 6.0221413e-1m / 77m);
1350 Se78.Value = ((Se.Value) * .234563m * 6.0221413e-1m / 78m);
1351 Se80.Value = ((Se.Value) * .502114m * 6.0221413e-1m / 80m);
1352 Se82.Value = ((Se.Value) * .090570m * 6.0221413e-1m / 82m);
1353
1354 Br79.Value = ((Br.Value) * .500650m * 6.0221413e-1m / 79m);
1355 Br81.Value = ((Br.Value) * .499350m * 6.0221413e-1m / 81m);
1356
1357 Kr78.Value = ((Kr.Value) * .003301m * 6.0221413e-1m / 78m);
1358 Kr80.Value = ((Kr.Value) * .021801m * 6.0221413e-1m / 80m);
1359 Kr82.Value = ((Kr.Value) * .113323m * 6.0221413e-1m / 82m);
1360 Kr83.Value = ((Kr.Value) * .113787m * 6.0221413e-1m / 83m);
1361 Kr84.Value = ((Kr.Value) * .570642m * 6.0221413e-1m / 84m);
1362 Kr86.Value = ((Kr.Value) * .177146m * 6.0221413e-1m / 86m);
1363
1364 Rb85.Value = ((Rb.Value) * .717006m * 6.0221413e-1m / 85m);
1365 Rb87.Value = ((Rb.Value) * .282994m * 6.0221413e-1m / 87m);
1366
1367 Sr84.Value = ((Sr.Value) * .005363m * 6.0221413e-1m / 84m);
1368 Sr86.Value = ((Sr.Value) * .096679m * 6.0221413e-1m / 86m);
1369 Sr87.Value = ((Sr.Value) * .069435m * 6.0221413e-1m / 87m);
1370 Sr88.Value = ((Sr.Value) * .828524m * 6.0221413e-1m / 88m);
1371
1372 Y89.Value = ((Y.Value) * 1.00000m * 6.0221413e-1m / 89m);
1373
1374 Zr90.Value = ((Zr.Value) * .507061m * 6.0221413e-1m / 90m);
1375 Zr91.Value = ((Zr.Value) * .111809m * 6.0221413e-1m / 91m);
1376 Zr92.Value = ((Zr.Value) * .172781m * 6.0221413e-1m / 92m);
1377 Zr94.Value = ((Zr.Value) * .178911m * 6.0221413e-1m / 94m);
1378 Zr96.Value = ((Zr.Value) * .029438m * 6.0221413e-1m / 96m);
1379
1380 Nb93.Value = ((Nb.Value) * 1.00000m * 6.0221413e-1m / 93m);
1381
1382 Mo92.Value = ((Mo.Value + SS316gram.Value * 0.03m) * .139163m * 6.0221413e-1m / 92m);
1383 Mo94.Value = ((Mo.Value + SS316gram.Value * 0.03m) * .089541m * 6.0221413e-1m / 94m);
1384 Mo95.Value = ((Mo.Value + SS316gram.Value * 0.03m) * .156660m * 6.0221413e-1m / 95m);
1385 Mo96.Value = ((Mo.Value + SS316gram.Value * 0.03m) * .166604m * 6.0221413e-1m / 96m);
1386 Mo97.Value = ((Mo.Value + SS316gram.Value * 0.03m) * .096947m * 6.0221413e-1m / 97m);
1387 Mo98.Value = ((Mo.Value + SS316gram.Value * 0.03m) * .248845m * 6.0221413e-1m / 98m);
1388 Mo100.Value = ((Mo.Value + SS316gram.Value * 0.03m) * .102240m * 6.0221413e-1m / 100m);
1389
1390 Ru96.Value = ((Ru.Value) * .052573m * 6.0221413e-1m / 96m);
1391 Ru98.Value = ((Ru.Value) * .018115m * 6.0221413e-1m / 98m);
1392 Ru99.Value = ((Ru.Value) * .124874m * 6.0221413e-1m / 99m);
1393 Ru100.Value = ((Ru.Value) * .124553m * 6.0221413e-1m / 100m);
1394 Ru101.Value = ((Ru.Value) * .170331m * 6.0221413e-1m / 101m);
1395 Ru102.Value = ((Ru.Value) * .318120m * 6.0221413e-1m / 102m);
1396 Ru104.Value = ((Ru.Value) * .191433m * 6.0221413e-1m / 104m);
1397
1398 Rh103.Value = ((Rh.Value) * 1.00000m * 6.0221413e-1m / 103m);
1399
1400 Pd102.Value = ((Pd.Value) * .009768m * 6.0221413e-1m / 102m);
1401 Pd104.Value = ((Pd.Value) * .108771m * 6.0221413e-1m / 104m);
1402 Pd105.Value = ((Pd.Value) * .220131m * 6.0221413e-1m / 105m);
1403 Pd106.Value = ((Pd.Value) * .271985m * 6.0221413e-1m / 106m);
1404 Pd108.Value = ((Pd.Value) * .268301m * 6.0221413e-1m / 108m);
1405 Pd110.Value = ((Pd.Value) * .121044m * 6.0221413e-1m / 110m);
1406
1407 Ag107.Value = ((Ag.Value + circuitboardgrams.Value * 0.05000m + pcbelectronicsgrams.Value *
.03000m) * .513762m * 6.0221413e-1m / 107m);
1408 Ag109.Value = ((Ag.Value + circuitboardgrams.Value * 0.05000m + pcbelectronicsgrams.Value *
.03000m) * .486238m * 6.0221413e-1m / 109m);
1409
1410 Cd106.Value = ((Cd.Value) * .011777m * 6.0221413e-1m / 106m);
1411 Cd108.Value = ((Cd.Value) * .008543m * 6.0221413e-1m / 108m);

1412 Cd110.Value = ((Cd.Value) * .122113m * 6.0221413e-1m / 110m);
 1413 Cd111.Value = ((Cd.Value) * .126284m * 6.0221413e-1m / 111m);
 1414 Cd112.Value = ((Cd.Value) * .240208m * 6.0221413e-1m / 112m);
 1415 Cd113.Value = ((Cd.Value) * .122736m * 6.0221413e-1m / 113m);
 1416 Cd114.Value = ((Cd.Value) * .291113m * 6.0221413e-1m / 114m);
 1417 Cd116.Value = ((Cd.Value) * .077228m * 6.0221413e-1m / 116m);
 1418
 1419 In113.Value = ((In.Value) * .042185m * 6.0221413e-1m / 113m);
 1420 In115.Value = ((In.Value) * .957815m * 6.0221413e-1m / 115m);
 1421
 1422 Sn112.Value = ((Sn.Value) * .009144m * 6.0221413e-1m / 112m);
 1423 Sn114.Value = ((Sn.Value) * .006333m * 6.0221413e-1m / 114m);
 1424 Sn115.Value = ((Sn.Value) * .003291m * 6.0221413e-1m / 115m);
 1425 Sn116.Value = ((Sn.Value) * .141960m * 6.0221413e-1m / 116m);
 1426 Sn117.Value = ((Sn.Value) * .075631m * 6.0221413e-1m / 117m);
 1427 Sn118.Value = ((Sn.Value) * .240550m * 6.0221413e-1m / 118m);
 1428 Sn119.Value = ((Sn.Value) * .086040m * 6.0221413e-1m / 119m);
 1429 Sn120.Value = ((Sn.Value) * .329072m * 6.0221413e-1m / 120m);
 1430 Sn122.Value = ((Sn.Value) * .047545m * 6.0221413e-1m / 122m);
 1431 Sn124.Value = ((Sn.Value) * .060434m * 6.0221413e-1m / 124m);
 1432
 1433 Sb121.Value = ((Sb.Value) * .568078m * 6.0221413e-1m / 121m);
 1434 Sb123.Value = ((Sb.Value) * .431922m * 6.0221413e-1m / 123m);
 1435
 1436 Te120.Value = ((Te.Value) * .000846m * 6.0221413e-1m / 120m);
 1437 Te122.Value = ((Te.Value) * .024361m * 6.0221413e-1m / 122m);
 1438 Te123.Value = ((Te.Value) * .008572m * 6.0221413e-1m / 123m);
 1439 Te124.Value = ((Te.Value) * .046025m * 6.0221413e-1m / 124m);
 1440 Te125.Value = ((Te.Value) * .069205m * 6.0221413e-1m / 125m);
 1441 Te126.Value = ((Te.Value) * .185890m * 6.0221413e-1m / 126m);
 1442 Te128.Value = ((Te.Value) * .318150m * 6.0221413e-1m / 128m);
 1443 Te130.Value = ((Te.Value) * .346951m * 6.0221413e-1m / 130m);
 1444
 1445 I127.Value = ((I.Value) * 1.00000m * 6.0221413e-1m / 127m);
 1446
 1447 Xe124.Value = ((Xe.Value) * .000898m * 6.0221413e-1m / 124m);
 1448 Xe126.Value = ((Xe.Value) * .000853m * 6.0221413e-1m / 126m);
 1449 Xe128.Value = ((Xe.Value) * .018609m * 6.0221413e-1m / 128m);
 1450 Xe129.Value = ((Xe.Value) * .259205m * 6.0221413e-1m / 129m);
 1451 Xe130.Value = ((Xe.Value) * .040279m * 6.0221413e-1m / 130m);
 1452 Xe131.Value = ((Xe.Value) * .211694m * 6.0221413e-1m / 131m);
 1453 Xe132.Value = ((Xe.Value) * .270340m * 6.0221413e-1m / 132m);
 1454 Xe134.Value = ((Xe.Value) * .106434m * 6.0221413e-1m / 134m);
 1455 Xe136.Value = ((Xe.Value) * .091686m * 6.0221413e-1m / 136m);
 1456
 1457 Cs133.Value = ((Cs.Value) * 1.00000m * 6.0221413e-1m / 133m);
 1458
 1459 Ba130.Value = ((Ba.Value) * .001003m * 6.0221413e-1m / 130m);
 1460 Ba132.Value = ((Ba.Value) * .000970m * 6.0221413e-1m / 132m);
 1461 Ba134.Value = ((Ba.Value) * .023568m * 6.0221413e-1m / 134m);
 1462 Ba135.Value = ((Ba.Value) * .064758m * 6.0221413e-1m / 135m);
 1463 Ba136.Value = ((Ba.Value) * .077727m * 6.0221413e-1m / 136m);
 1464 Ba137.Value = ((Ba.Value) * .111976m * 6.0221413e-1m / 137m);
 1465 Ba138.Value = ((Ba.Value) * .720000m * 6.0221413e-1m / 138m);
 1466
 1467 La138.Value = ((La.Value) * .000882m * 6.0221413e-1m / 138m);
 1468 La139.Value = ((La.Value) * .999118m * 6.0221413e-1m / 139m);
 1469
 1470 Ce136.Value = ((Ce.Value) * .001794m * 6.0221413e-1m / 136m);
 1471 Ce138.Value = ((Ce.Value) * .002470m * 6.0221413e-1m / 138m);
 1472 Ce140.Value = ((Ce.Value) * .883173m * 6.0221413e-1m / 140m);
 1473 Ce142.Value = ((Ce.Value) * .112563m * 6.0221413e-1m / 142m);
 1474
 1475 Pr141.Value = ((Pr.Value) * 1.00000m * 6.0221413e-1m / 141m);
 1476
 1477 Nd142.Value = ((Nd.Value) * .267127m * 6.0221413e-1m / 142m);
 1478 Nd143.Value = ((Nd.Value) * .120616m * 6.0221413e-1m / 143m);
 1479 Nd144.Value = ((Nd.Value) * .237433m * 6.0221413e-1m / 144m);
 1480 Nd145.Value = ((Nd.Value) * .083316m * 6.0221413e-1m / 145m);
 1481 Nd146.Value = ((Nd.Value) * .173882m * 6.0221413e-1m / 146m);

1482 Nd148.Value = ((Nd.Value) * .059027m * 6.0221413e-1m / 148m);
 1483 Nd150.Value = ((Nd.Value) * .058600m * 6.0221413e-1m / 150m);
 1484
 1485 Sm144.Value = ((Sm.Value) * .029382m * 6.0221413e-1m / 144m);
 1486 Sm147.Value = ((Sm.Value) * .146459m * 6.0221413e-1m / 147m);
 1487 Sm148.Value = ((Sm.Value) * .110567m * 6.0221413e-1m / 148m);
 1488 Sm149.Value = ((Sm.Value) * .136868m * 6.0221413e-1m / 149m);
 1489 Sm150.Value = ((Sm.Value) * .073580m * 6.0221413e-1m / 150m);
 1490 Sm152.Value = ((Sm.Value) * .270263m * 6.0221413e-1m / 152m);
 1491 Sm154.Value = ((Sm.Value) * .232880m * 6.0221413e-1m / 154m);
 1492
 1493 Eu151.Value = ((Eu.Value) * .474814m * 6.0221413e-1m / 151m);
 1494 Eu153.Value = ((Eu.Value) * .525186m * 6.0221413e-1m / 153m);
 1495
 1496 Gd152.Value = ((Gd.Value) * .001932m * 6.0221413e-1m / 152m);
 1497 Gd154.Value = ((Gd.Value) * .021338m * 6.0221413e-1m / 154m);
 1498 Gd155.Value = ((Gd.Value) * .145808m * 6.0221413e-1m / 155m);
 1499 Gd156.Value = ((Gd.Value) * .202969m * 6.0221413e-1m / 156m);
 1500 Gd157.Value = ((Gd.Value) * .156173m * 6.0221413e-1m / 157m);
 1501 Gd158.Value = ((Gd.Value) * .249461m * 6.0221413e-1m / 158m);
 1502 Gd160.Value = ((Gd.Value) * .222318m * 6.0221413e-1m / 160m);
 1503
 1504 Tb159.Value = ((Tb.Value) * 1.00000m * 6.0221413e-1m / 159m);
 1505
 1506 Dy156.Value = ((Dy.Value) * .000537m * 6.0221413e-1m / 156m);
 1507 Dy158.Value = ((Dy.Value) * .000923m * 6.0221413e-1m / 158m);
 1508 Dy160.Value = ((Dy.Value) * .022921m * 6.0221413e-1m / 160m);
 1509 Dy161.Value = ((Dy.Value) * .187062m * 6.0221413e-1m / 161m);
 1510 Dy162.Value = ((Dy.Value) * .253852m * 6.0221413e-1m / 162m);
 1511 Dy163.Value = ((Dy.Value) * .249618m * 6.0221413e-1m / 163m);
 1512 Dy164.Value = ((Dy.Value) * .285086m * 6.0221413e-1m / 164m);
 1513
 1514 Ho165.Value = ((Ho.Value) * 1.00000m * 6.0221413e-1m / 165m);
 1515
 1516 Er162.Value = ((Er.Value) * .001346m * 6.0221413e-1m / 162m);
 1517 Er164.Value = ((Er.Value) * .015691m * 6.0221413e-1m / 164m);
 1518 Er166.Value = ((Er.Value) * .332368m * 6.0221413e-1m / 166m);
 1519 Er167.Value = ((Er.Value) * .228243m * 6.0221413e-1m / 167m);
 1520 Er168.Value = ((Er.Value) * .270866m * 6.0221413e-1m / 168m);
 1521 Er170.Value = ((Er.Value) * .151486m * 6.0221413e-1m / 170m);
 1522
 1523 Tm169.Value = ((Tm.Value) * 1.00000m * 6.0221413e-1m / 169m);
 1524
 1525 Yb168.Value = ((Yb.Value) * .001194m * 6.0221413e-1m / 168m);
 1526 Yb170.Value = ((Yb.Value) * .029282m * 6.0221413e-1m / 170m);
 1527 Yb171.Value = ((Yb.Value) * .139176m * 6.0221413e-1m / 171m);
 1528 Yb172.Value = ((Yb.Value) * .215400m * 6.0221413e-1m / 173m);
 1529 Yb173.Value = ((Yb.Value) * .160922m * 6.0221413e-1m / 173m);
 1530 Yb174.Value = ((Yb.Value) * .321897m * 6.0221413e-1m / 174m);
 1531 Yb176.Value = ((Yb.Value) * .132189m * 6.0221413e-1m / 176m);
 1532
 1533 Lu175.Value = ((Lu.Value) * .973865m * 6.0221413e-1m / 175m);
 1534 Lu176.Value = ((Lu.Value) * .026135m * 6.0221413e-1m / 176m);
 1535
 1536 Hf174.Value = ((Hf.Value) * .001559m * 6.0221413e-1m / 174m);
 1537 Hf176.Value = ((Hf.Value) * .051850m * 6.0221413e-1m / 176m);
 1538 Hf177.Value = ((Hf.Value) * .184393m * 6.0221413e-1m / 177m);
 1539 Hf178.Value = ((Hf.Value) * .271973m * 6.0221413e-1m / 178m);
 1540 Hf179.Value = ((Hf.Value) * .136552m * 6.0221413e-1m / 179m);
 1541 Hf180.Value = ((Hf.Value) * .353673m * 6.0221413e-1m / 180m);
 1542
 1543 Ta180.Value = ((Ta.Value) * .000119m * 6.0221413e-1m / 180m);
 1544 Ta181.Value = ((Ta.Value) * .999881m * 6.0221413e-1m / 181m);
 1545
 1546 W180.Value = ((W.Value) * .001175m * 6.0221413e-1m / 180m);
 1547 W182.Value = ((W.Value) * .262270m * 6.0221413e-1m / 182m);
 1548 W183.Value = ((W.Value) * .142406m * 6.0221413e-1m / 183m);
 1549 W184.Value = ((W.Value) * .306582m * 6.0221413e-1m / 184m);
 1550 W186.Value = ((W.Value) * .287567m * 6.0221413e-1m / 186m);
 1551

1552 Re185.Value = ((Re.Value) * .371482m * 6.0221413e-1m / 185m);
 1553 Re187.Value = ((Re.Value) * .628518m * 6.0221413e-1m / 187m);
 1554
 1555 Os184.Value = ((Os.Value) * .000193m * 6.0221413e-1m / 184m);
 1556 Os186.Value = ((Os.Value) * .015543m * 6.0221413e-1m / 186m);
 1557 Os187.Value = ((Os.Value) * .019263m * 6.0221413e-1m / 187m);
 1558 Os188.Value = ((Os.Value) * .130821m * 6.0221413e-1m / 188m);
 1559 Os189.Value = ((Os.Value) * .160425m * 6.0221413e-1m / 189m);
 1560 Os190.Value = ((Os.Value) * .262232m * 6.0221413e-1m / 190m);
 1561 Os192.Value = ((Os.Value) * .411523m * 6.0221413e-1m / 192m);
 1562
 1563 Ir191.Value = ((Ir.Value) * .370564m * 6.0221413e-1m / 191m);
 1564 Ir193.Value = ((Ir.Value) * .629436m * 6.0221413e-1m / 193m);
 1565
 1566 Pt190.Value = ((Pt.Value) * .000117m * 6.0221413e-1m / 190m);
 1567 Pt192.Value = ((Pt.Value) * .007694m * 6.0221413e-1m / 192m);
 1568 Pt194.Value = ((Pt.Value) * .326697m * 6.0221413e-1m / 194m);
 1569 Pt195.Value = ((Pt.Value) * .337579m * 6.0221413e-1m / 195m);
 1570 Pt196.Value = ((Pt.Value) * .253228m * 6.0221413e-1m / 196m);
 1571 Pt198.Value = ((Pt.Value) * .074685m * 6.0221413e-1m / 198m);
 1572
 1573 Au197.Value = ((Au.Value + circuitboardgrams.Value * 0.05000m + pcbelectronicsgrams.Value *
 .04000m) * 1.00000m * 6.0221413e-1m / 197m);
 1574
 1575 Hg196.Value = ((Hg.Value) * .001465m * 6.0221413e-1m / 196m);
 1576 Hg198.Value = ((Hg.Value) * .098392m * 6.0221413e-1m / 198m);
 1577 Hg199.Value = ((Hg.Value) * .167328m * 6.0221413e-1m / 199m);
 1578 Hg200.Value = ((Hg.Value) * .230274m * 6.0221413e-1m / 200m);
 1579 Hg201.Value = ((Hg.Value) * .132044m * 6.0221413e-1m / 201m);
 1580 Hg202.Value = ((Hg.Value) * .300641m * 6.0221413e-1m / 202m);
 1581 Hg204.Value = ((Hg.Value) * .069856m * 6.0221413e-1m / 204m);
 1582
 1583 Tl203.Value = ((Tl.Value) * .293190m * 6.0221413e-1m / 203m);
 1584 Tl205.Value = ((Tl.Value) * .706810m * 6.0221413e-1m / 205m);
 1585
 1586 Pb204.Value = ((Pb.Value) * .013781m * 6.0221413e-1m / 204m);
 1587 Pb206.Value = ((Pb.Value) * .239555m * 6.0221413e-1m / 206m);
 1588 Pb207.Value = ((Pb.Value) * .220743m * 6.0221413e-1m / 207m);
 1589 Pb208.Value = ((Pb.Value) * .525921m * 6.0221413e-1m / 208m);
 1590
 1591
 1592
 1593 nuclidesum.Value = (H1.Value + H2.Value + He3.Value + He4.Value + Li6.Value + Li7.Value +
 Be9.Value + B10.Value + B11.Value + C12.Value + C13.Value + N14.Value + N15.Value + O16.Value + O17.Value +
 O18.Value + F19.Value + Ne20.Value + Ne21.Value + Ne22.Value + Na23.Value + Mg24.Value + Mg25.Value +
 Mg26.Value + Al27.Value + Si28.Value + Si29.Value + Si30.Value + P31.Value + S32.Value + S33.Value + S34.Value +
 S36.Value + Cl35.Value + Cl37.Value + Ar36.Value + Ar38.Value + Ar40.Value + K39.Value + K41.Value + Ca40.Value +
 Ca42.Value + Ca43.Value + Ca44.Value + Ca46.Value + Sc45.Value + Ti46.Value + Ti47.Value + Ti48.Value +
 Ti49.Value + Ti50.Value + V51.Value + Cr50.Value + Cr52.Value + Cr53.Value + Cr54.Value + Mn55.Value + Fe54.Value
 + Fe56.Value + Fe57.Value + Fe58.Value + Co59.Value + Ni58.Value + Ni60.Value + Ni61.Value + Ni62.Value +
 Ni64.Value + Cu63.Value + Cu65.Value + Zn64.Value + Zn66.Value + Zn67.Value + Zn68.Value + Zn70.Value +
 Ga69.Value + Ga71.Value + Ge70.Value + Ge72.Value + Ge73.Value + Ge74.Value + As75.Value + Se74.Value +
 Se76.Value + Se77.Value + Se78.Value + Se80.Value + Br79.Value + Br81.Value + Kr78.Value + Kr80.Value +
 Kr82.Value + Kr83.Value + Kr84.Value + Kr86.Value + Rb85.Value + Sr84.Value + Sr86.Value + Sr87.Value + Sr88.Value
 + Y89.Value + Zr90.Value + Zr91.Value + Zr92.Value + Zr94.Value + Nb93.Value + Mo92.Value + Mo94.Value +
 Mo95.Value + Mo96.Value + Mo97.Value + Mo98.Value + Ru96.Value + Ru98.Value + Ru99.Value + Ru100.Value +
 Ru101.Value + Ru102.Value + Ru104.Value + Rh103.Value + Pd102.Value + Pd104.Value + Pd105.Value +
 Pd106.Value + Pd108.Value + Pd110.Value + Ag107.Value + Ag109.Value + Cd106.Value + Cd108.Value + Cd110.Value
 + Cd111.Value + Cd112.Value + Cd114.Value + In113.Value + Sn112.Value + Sn114.Value + Sn115.Value +
 Sn116.Value + Sn117.Value + Sn118.Value + Sn119.Value + Sn120.Value + Sn122.Value + Sn124.Value + Sb121.Value
 + Sb123.Value + Te120.Value + Te122.Value + Te123.Value + Te124.Value + Te125.Value + Te126.Value + I127.Value
 + Xe124.Value + Xe126.Value + Xe128.Value + Xe129.Value + Xe130.Value + Xe131.Value + Xe132.Value +
 Xe134.Value + Xe136.Value + Cs133.Value + Ba132.Value + Ba134.Value + Ba135.Value + Ba136.Value + Ba137.Value
 + Ba138.Value + La139.Value + Ce136.Value + Ce138.Value + Ce140.Value + Ce142.Value + Pr141.Value +
 Nd142.Value + Nd143.Value + Nd145.Value + Nd146.Value + Nd148.Value + Sm144.Value + Sm149.Value +
 Sm150.Value + Sm152.Value + Sm154.Value + Eu151.Value + Eu153.Value + Gd154.Value + Gd155.Value +
 Gd156.Value + Gd157.Value + Gd158.Value + Gd160.Value + Tb159.Value + Dy156.Value + Dy158.Value +
 Dy160.Value + Dy161.Value + Dy162.Value + Dy163.Value + Dy164.Value + Ho165.Value + Er162.Value + Er164.Value
 + Er166.Value + Er167.Value + Er168.Value + Er170.Value + Tm169.Value + Yb168.Value + Yb170.Value + Yb171.Value
 + Yb172.Value + Yb173.Value + Yb174.Value + Yb176.Value + Lu175.Value + Hf176.Value + Hf177.Value + Hf178.Value

```

+ Hf179.Value + Hf180.Value + Ta180.Value + Ta181.Value + W180.Value + W182.Value + W183.Value + W184.Value +
W186.Value + Re185.Value + Os184.Value + Os187.Value + Os188.Value + Os189.Value + Os190.Value + Os192.Value
+ Ir191.Value + Ir193.Value + Pt192.Value + Pt194.Value + Pt195.Value + Pt196.Value + Pt198.Value + Au197.Value +
Hg196.Value + Hg198.Value + Hg199.Value + Hg200.Value + Hg201.Value + Hg202.Value + Hg204.Value + Tl203.Value
+ Tl205.Value + Pb204.Value + Pb206.Value + Pb207.Value + Pb208.Value + K40.Value + Ca48.Value + V50.Value +
Ge76.Value + Se82.Value + Rb87.Value + Zr96.Value + Mo100.Value + Cd113.Value + Cd116.Value + In115.Value +
Te128.Value + Te130.Value + Ba130.Value + La138.Value + Nd144.Value + Nd150.Value + Sm147.Value +
Sm148.Value + Gd152.Value + Lu176.Value + Hf174.Value + Re187.Value + Os186.Value + Pt190.Value);
1594
1595     }
1596
1597     private void label8_Click(object sender, EventArgs e)
1598     {
1599
1600     }
1601
1602     private void listBox1_SelectedIndexChanged_1(object sender, EventArgs e)
1603     {
1604
1605     }
1606
1607     private void numericUpDown15_ValueChanged(object sender, EventArgs e)
1608     {
1609
1610     }
1611
1612     private void numericUpDown191_ValueChanged(object sender, EventArgs e)
1613     {
1614
1615     }
1616
1617     private void numericUpDown20_ValueChanged(object sender, EventArgs e)
1618     {
1619
1620     }
1621
1622     private void numericUpDown52_ValueChanged(object sender, EventArgs e)
1623     {
1624
1625     }
1626
1627     private void numericUpDown30_ValueChanged(object sender, EventArgs e)
1628     {
1629
1630     }
1631
1632     private void numericUpDown1_ValueChanged(object sender, EventArgs e)
1633     {
1634
1635     }
1636
1637     private void numericUpDown63_ValueChanged(object sender, EventArgs e)
1638     {
1639
1640     }
1641
1642     private void numericUpDown68_ValueChanged(object sender, EventArgs e)
1643     {
1644
1645     }
1646
1647     private void numericUpDown87_ValueChanged(object sender, EventArgs e)
1648     {
1649
1650     }
1651
1652     private void numericUpDown96_ValueChanged(object sender, EventArgs e)
1653     {
1654
1655     }

```

```

1656
1657     private void numericUpDown93_ValueChanged(object sender, EventArgs e)
1658     {
1659
1660     }
1661
1662
1663
1664     private void titleTextBox_TextChanged(object sender, EventArgs e)
1665     {
1666
1667     }
1668
1669     private void sN119_ValueChanged(object sender, EventArgs e)
1670     {
1671
1672     }
1673
1674     private void numericUpDown138_ValueChanged(object sender, EventArgs e)
1675     {
1676
1677     }
1678
1679     private void sulfurgramstandard_ValueChanged_1(object sender, EventArgs e)
1680     {
1681         AtomCalc();
1682     }
1683
1684     private void sulfurgramlarge_ValueChanged_1(object sender, EventArgs e)
1685     {
1686         AtomCalc();
1687     }
1688
1689     private void tldgram_ValueChanged(object sender, EventArgs e)
1690     {
1691         AtomCalc();
1692     }
1693
1694     private void Al6061gram_ValueChanged(object sender, EventArgs e)
1695     {
1696         AtomCalc();
1697     }
1698
1699     private void panel3_Paint_1(object sender, PaintEventArgs e)
1700     {
1701
1702     }
1703
1704     private void S_ValueChanged(object sender, EventArgs e)
1705     {
1706         AtomCalc();
1707     }
1708
1709     private void H_ValueChanged(object sender, EventArgs e)
1710     {
1711         AtomCalc();
1712     }
1713
1714     private void He_ValueChanged(object sender, EventArgs e)
1715     {
1716         AtomCalc();
1717     }
1718
1719     private void Li_ValueChanged(object sender, EventArgs e)
1720     {
1721         AtomCalc();
1722     }
1723
1724     private void Be_ValueChanged(object sender, EventArgs e)
1725     {

```

```

1726         AtomCalc();
1727     }
1728
1729     private void B_ValueChanged(object sender, EventArgs e)
1730     {
1731         AtomCalc();
1732     }
1733
1734     private void C_ValueChanged(object sender, EventArgs e)
1735     {
1736         AtomCalc();
1737     }
1738
1739     private void N_ValueChanged(object sender, EventArgs e)
1740     {
1741         AtomCalc();
1742     }
1743
1744     private void O_ValueChanged(object sender, EventArgs e)
1745     {
1746         AtomCalc();
1747     }
1748
1749     private void F_ValueChanged(object sender, EventArgs e)
1750     {
1751         AtomCalc();
1752     }
1753
1754     private void Ne_ValueChanged(object sender, EventArgs e)
1755     {
1756         AtomCalc();
1757     }
1758
1759     private void Na_ValueChanged(object sender, EventArgs e)
1760     {
1761         AtomCalc();
1762     }
1763
1764     private void Mg_ValueChanged(object sender, EventArgs e)
1765     {
1766         AtomCalc();
1767     }
1768
1769     private void Al_ValueChanged(object sender, EventArgs e)
1770     {
1771         AtomCalc();
1772     }
1773
1774     private void Si_ValueChanged(object sender, EventArgs e)
1775     {
1776         AtomCalc();
1777     }
1778
1779     private void P_ValueChanged(object sender, EventArgs e)
1780     {
1781         AtomCalc();
1782     }
1783
1784     private void Cl_ValueChanged(object sender, EventArgs e)
1785     {
1786         AtomCalc();
1787     }
1788
1789     private void Ar_ValueChanged(object sender, EventArgs e)
1790     {
1791         AtomCalc();
1792     }
1793
1794     private void K_ValueChanged(object sender, EventArgs e)
1795     {

```

```

1796         AtomCalc();
1797     }
1798
1799     private void Ca_ValueChanged(object sender, EventArgs e)
1800     {
1801         AtomCalc();
1802     }
1803
1804     private void Sc_ValueChanged(object sender, EventArgs e)
1805     {
1806         AtomCalc();
1807     }
1808
1809     private void Ti_ValueChanged(object sender, EventArgs e)
1810     {
1811         AtomCalc();
1812     }
1813
1814     private void V_ValueChanged(object sender, EventArgs e)
1815     {
1816         AtomCalc();
1817     }
1818
1819     private void Cr_ValueChanged(object sender, EventArgs e)
1820     {
1821         AtomCalc();
1822     }
1823
1824     private void numericUpDown230_ValueChanged(object sender, EventArgs e)
1825     {
1826         AtomCalc();
1827     }
1828
1829     private void Fe_ValueChanged(object sender, EventArgs e)
1830     {
1831         AtomCalc();
1832     }
1833
1834     private void Co_ValueChanged(object sender, EventArgs e)
1835     {
1836         AtomCalc();
1837     }
1838
1839     private void Ni_ValueChanged(object sender, EventArgs e)
1840     {
1841         AtomCalc();
1842     }
1843
1844     private void Cu_ValueChanged(object sender, EventArgs e)
1845     {
1846         AtomCalc();
1847     }
1848
1849     private void Zn_ValueChanged(object sender, EventArgs e)
1850     {
1851         AtomCalc();
1852     }
1853
1854     private void Ga_ValueChanged(object sender, EventArgs e)
1855     {
1856         AtomCalc();
1857     }
1858
1859     private void Ge_ValueChanged(object sender, EventArgs e)
1860     {
1861         AtomCalc();
1862     }
1863
1864     private void As_ValueChanged(object sender, EventArgs e)
1865     {

```

```

1866         AtomCalc();
1867     }
1868
1869     private void Se_ValueChanged(object sender, EventArgs e)
1870     {
1871         AtomCalc();
1872     }
1873
1874     private void Br_ValueChanged(object sender, EventArgs e)
1875     {
1876         AtomCalc();
1877     }
1878
1879     private void Kr_ValueChanged(object sender, EventArgs e)
1880     {
1881         AtomCalc();
1882     }
1883
1884     private void Rb_ValueChanged(object sender, EventArgs e)
1885     {
1886         AtomCalc();
1887     }
1888
1889     private void Sr_ValueChanged(object sender, EventArgs e)
1890     {
1891         AtomCalc();
1892     }
1893
1894     private void Y_ValueChanged(object sender, EventArgs e)
1895     {
1896         AtomCalc();
1897     }
1898
1899     private void Zr_ValueChanged(object sender, EventArgs e)
1900     {
1901         AtomCalc();
1902     }
1903
1904     private void Nb_ValueChanged(object sender, EventArgs e)
1905     {
1906         AtomCalc();
1907     }
1908
1909     private void Mo_ValueChanged(object sender, EventArgs e)
1910     {
1911         AtomCalc();
1912     }
1913
1914     private void Tc_ValueChanged(object sender, EventArgs e)
1915     {
1916         AtomCalc();
1917     }
1918
1919     private void Ru_ValueChanged(object sender, EventArgs e)
1920     {
1921         AtomCalc();
1922     }
1923
1924     private void Rh_ValueChanged(object sender, EventArgs e)
1925     {
1926         AtomCalc();
1927     }
1928
1929     private void Pd_ValueChanged(object sender, EventArgs e)
1930     {
1931         AtomCalc();
1932     }
1933
1934     private void Ag_ValueChanged(object sender, EventArgs e)
1935     {

```

```

1936         AtomCalc();
1937     }
1938
1939     private void Cd_ValueChanged(object sender, EventArgs e)
1940     {
1941         AtomCalc();
1942     }
1943
1944     private void In_ValueChanged(object sender, EventArgs e)
1945     {
1946         AtomCalc();
1947     }
1948
1949     private void Sn_ValueChanged(object sender, EventArgs e)
1950     {
1951         AtomCalc();
1952     }
1953
1954     private void Sb_ValueChanged(object sender, EventArgs e)
1955     {
1956         AtomCalc();
1957     }
1958
1959     private void Te_ValueChanged(object sender, EventArgs e)
1960     {
1961         AtomCalc();
1962     }
1963
1964     private void I_ValueChanged(object sender, EventArgs e)
1965     {
1966         AtomCalc();
1967     }
1968
1969     private void Xe_ValueChanged(object sender, EventArgs e)
1970     {
1971         AtomCalc();
1972     }
1973
1974     private void Cs_ValueChanged(object sender, EventArgs e)
1975     {
1976         AtomCalc();
1977     }
1978
1979     private void Ba_ValueChanged(object sender, EventArgs e)
1980     {
1981         AtomCalc();
1982     }
1983
1984     private void La_ValueChanged(object sender, EventArgs e)
1985     {
1986         AtomCalc();
1987     }
1988
1989     private void Ce_ValueChanged(object sender, EventArgs e)
1990     {
1991         AtomCalc();
1992     }
1993
1994     private void Pr_ValueChanged(object sender, EventArgs e)
1995     {
1996         AtomCalc();
1997     }
1998
1999     private void Nd_ValueChanged(object sender, EventArgs e)
2000     {
2001         AtomCalc();
2002     }
2003
2004     private void Pm_ValueChanged(object sender, EventArgs e)
2005     {

```

```

2006         AtomCalc();
2007     }
2008
2009     private void Sm_ValueChanged(object sender, EventArgs e)
2010     {
2011         AtomCalc();
2012     }
2013
2014     private void Eu_ValueChanged(object sender, EventArgs e)
2015     {
2016         AtomCalc();
2017     }
2018
2019     private void Gd_ValueChanged(object sender, EventArgs e)
2020     {
2021         AtomCalc();
2022     }
2023
2024     private void Tb_ValueChanged(object sender, EventArgs e)
2025     {
2026         AtomCalc();
2027     }
2028
2029     private void Dy_ValueChanged(object sender, EventArgs e)
2030     {
2031         AtomCalc();
2032     }
2033
2034     private void Ho_ValueChanged(object sender, EventArgs e)
2035     {
2036         AtomCalc();
2037     }
2038
2039     private void Er_ValueChanged(object sender, EventArgs e)
2040     {
2041         AtomCalc();
2042     }
2043
2044     private void Tm_ValueChanged(object sender, EventArgs e)
2045     {
2046         AtomCalc();
2047     }
2048
2049     private void Yb_ValueChanged(object sender, EventArgs e)
2050     {
2051         AtomCalc();
2052     }
2053
2054     private void Lu_ValueChanged(object sender, EventArgs e)
2055     {
2056         AtomCalc();
2057     }
2058
2059     private void Hf_ValueChanged(object sender, EventArgs e)
2060     {
2061         AtomCalc();
2062     }
2063
2064     private void Ta_ValueChanged(object sender, EventArgs e)
2065     {
2066         AtomCalc();
2067     }
2068
2069     private void W_ValueChanged(object sender, EventArgs e)
2070     {
2071         AtomCalc();
2072     }
2073
2074     private void Re_ValueChanged(object sender, EventArgs e)
2075     {

```



```

2076         AtomCalc();
2077     }
2078
2079     private void Os_ValueChanged(object sender, EventArgs e)
2080     {
2081         AtomCalc();
2082     }
2083
2084     private void Ir_ValueChanged(object sender, EventArgs e)
2085     {
2086         AtomCalc();
2087     }
2088
2089     private void Pt_ValueChanged(object sender, EventArgs e)
2090     {
2091         AtomCalc();
2092     }
2093
2094     private void Au_ValueChanged(object sender, EventArgs e)
2095     {
2096         AtomCalc();
2097     }
2098
2099     private void Hg_ValueChanged(object sender, EventArgs e)
2100     {
2101         AtomCalc();
2102     }
2103
2104     private void Tl_ValueChanged(object sender, EventArgs e)
2105     {
2106         AtomCalc();
2107     }
2108
2109     private void Pb_ValueChanged(object sender, EventArgs e)
2110     {
2111         AtomCalc();
2112     }
2113
2114
2115
2116     private void nuclidesum_ValueChanged(object sender, EventArgs e)
2117     {
2118
2119     }
2120
2121     private void panel4_Paint(object sender, PaintEventArgs e)
2122     {
2123
2124     }
2125
2126     private void hrs_ValueChanged(object sender, EventArgs e)
2127     {
2128
2129     }
2130
2131     private void run_Click(object sender, EventArgs e)
2132     {
2133         //System.Diagnostics.ProcessStartInfo info = new System.Diagnostics.ProcessStartInfo("cmd.exe");
2134         //info.WorkingDirectory = "c:\\cinder\\" + titleTextBox.Text;
2135         //info.Arguments = "/C C:\\Cinder2008\\cinder.exe";
2136         //System.Diagnostics.Process.Start(info);
2137         //System.Diagnostics.Process.
2138
2139
2140         String command = @"/C C:\\Cinder2008\\cinder.exe";
2141         System.Diagnostics.ProcessStartInfo cmdsi = new System.Diagnostics.ProcessStartInfo("cmd.exe");
2142         cmdsi.Arguments = command;
2143         cmdsi.WorkingDirectory = "c:\\cinder\\" + titleTextBox.Text;
2144         System.Diagnostics.Process cmd = System.Diagnostics.Process.Start(cmdsi);
2145         cmd.WaitForExit(); //wait indefinitely for the associated process to exit.

```

```

2146         results_Click(null, null);
2147
2148
2149
2150     }
2151
2152     private void circuitboardgrams_ValueChanged(object sender, EventArgs e)
2153     {
2154         AtomCalc();
2155     }
2156
2157     private void pcbelectronicsgrams_ValueChanged(object sender, EventArgs e)
2158     {
2159         AtomCalc();
2160     }
2161
2162     private void textBox1_TextChanged_1(object sender, EventArgs e)
2163     {
2164
2165     }
2166
2167     private void results_Click(object sender, EventArgs e)
2168     {
2169
2170         label5.Show();
2171         label6.Show();
2172         label14.Show();
2173         label15.Show();
2174         label16.Show();
2175         textBox1.Show();
2176         textBox2.Show();
2177         numericUpDown5.Show();
2178         numericUpDown3.Show();
2179         numericUpDown6.Show();
2180         double mRem;
2181
2182         // Read each line of the file into a string array. Each element
2183         // of the array is one line of the file.
2184         string[] lines = System.IO.File.ReadAllLines("c:\\cinder\\" + titleTextBox.Text + "\\tables_by_grp");
2185
2186         foreach (string line in lines)
2187         {
2188             if (line.StartsWith(" TOTAL GAMMAS/(CC-S)"))
2189             {
2190                 string[] substrings=line.Split(new char[] { '|' });
2191                 textBox1.Text = substrings[2];
2192                 numericUpDown1.Value = Decimal.Parse(textBox1.Text,
2193 System.Globalization.NumberStyles.Any);
2194             }
2195
2196             if (line.StartsWith(" GROUP MID POINTS:"))
2197             {
2198                 string[] substrings=line.Split(new char[] { '|' });
2199                 textBox2.Text = substrings[2];
2200                 numericUpDown2.Value = Decimal.Parse(textBox2.Text,
2201 System.Globalization.NumberStyles.Any);
2202                 break;
2203             }
2204         }
2205
2206         double E = Math.Log(System.Convert.ToDouble(numericUpDown2.Value));
2207
2208         if (numericUpDown2.Value <= 0.03m)
2209         {
2210             mRem = 1000.0 * System.Convert.ToDouble(numericUpDown1.Value) / (4.0 * Math.PI * 30.0 * 30.0) *
2211 Math.Exp(-20.477 + -1.7454 * E);
2212             numericUpDown3.Value = System.Convert.ToDecimal(mRem);
2213         }
2214     }

```

```

2213         if (numericUpDown2.Value <= 0.5m & numericUpDown2.Value > 0.03m)
2214         {
2215             mRem = 1000.0 * System.Convert.ToDouble(numericUpDown1.Value) / (4.0 * Math.PI * 30.0 * 30.0)
2216             * Math.Exp(-13.626 + -0.57117 * E + -1.0954 * E * E + -.024897 * E * E * E);
2217             numericUpDown3.Value = System.Convert.ToDecimal(mRem);
2218         }
2219         if (numericUpDown2.Value < 5.0m & numericUpDown2.Value > 0.5m)
2220         {
2221             mRem = 1000.0 * System.Convert.ToDouble(numericUpDown1.Value) / (4.0 * Math.PI * 30.0 * 30.0) *
2222             Math.Exp(-13.133 + 0.72008 * E + -0.033603 * E * E);
2223             numericUpDown3.Value = System.Convert.ToDecimal(mRem);
2224         }
2225         if (numericUpDown2.Value <= 15.0m & numericUpDown2.Value > 5.0m)
2226         {
2227             mRem = 1000.0 * System.Convert.ToDouble(numericUpDown1.Value) / (4.0 * Math.PI * 30.0 * 30.0) *
2228             Math.Exp(-12.791 + 0.28309 * E + 0.10873 * E * E);
2229             numericUpDown3.Value = System.Convert.ToDecimal(mRem);
2230         }
2231         numericUpDown5.Value = numericUpDown3.Value * 30m; // assume linear within 1-foot
2232         numericUpDown6.Value = numericUpDown3.Value * 0.09m; // l/l0 = r0^2/r^2 30^2/100^2
2233     }
2234 }
2235
2236 private void button4_Click_1(object sender, EventArgs e)
2237 {
2238     panel2.Show();
2239     label22.Show();
2240     label23.Show();
2241     button4.Hide();
2242 }
2243
2244 private void numericUpDown3_ValueChanged_1(object sender, EventArgs e)
2245 {
2246 }
2247
2248 private void numericUpDown13_ValueChanged(object sender, EventArgs e)
2249 {
2250     AtomCalc();
2251 }
2252
2253 private void cardboardgrams_ValueChanged(object sender, EventArgs e)
2254 {
2255     AtomCalc();
2256 }
2257
2258 private void panel1_Paint(object sender, PaintEventArgs e)
2259 {
2260 }
2261
2262 }
2263
2264 private void pvc_CheckedChanged(object sender, EventArgs e)
2265 {
2266     if (pvc.Checked)
2267     {
2268         pvcgram.Show();
2269     }
2270     else
2271     {
2272         pvcgram.Value = 0;
2273         pvcgram.Hide();
2274     }
2275 }
2276
2277 private void teflon_CheckedChanged(object sender, EventArgs e)
2278 {
2279     if (teflon.Checked)

```

```

2280         {
2281             teflongram.Show();
2282         }
2283         else
2284         {
2285             teflongram.Value = 0;
2286             teflongram.Hide();
2287         }
2288     }
2289
2290     private void polygram_ValueChanged(object sender, EventArgs e)
2291     {
2292         AtomCalc();
2293     }
2294
2295     private void pvcgram_ValueChanged(object sender, EventArgs e)
2296     {
2297         AtomCalc();
2298     }
2299
2300     private void teflongram_ValueChanged(object sender, EventArgs e)
2301     {
2302         AtomCalc();
2303     }
2304
2305     private void Ca42_ValueChanged(object sender, EventArgs e)
2306     {
2307     }
2308 }
2309
2310 }
2311 }
2312 }

```

Bibliography

Attix, Frank H. *Introduction to Radiological Physics and Radiation Dosimetry*. New York: Wiley, 1986. Print.

Chabot, George. "Relationship Between Radionuclide Gamma Emission and Exposure Rate." 13 Aug. 2014. Web. 4 Nov. 2014.

DePriest, K. Russell, and Karen C. Saavedra. "MatMCNP: A Code for Producing Material Cards for MCNP." *Sandia Report* SAND2014-17693 (2014). Print.

DePriest, K. Russell, Phillip J. Cooper, and Edward J. Parma. "MCNP/MCNPX Model of the Annular Core Research Reactor." *Sandia Report* Sand2006-3067 (2006). Print.

England, T.R. "cinder — A One-Point Depletion and Fission Product Program," Bettis Atomic Power Laboratory report WAPD-TM-334 (August 1962; Rev. June 1964).

Holloway, S. T. *CINDER2008*. Vers. 1.0. N.p.: Los Alamos National Laboratory, 2011. Computer software.

Lamarsh, John R. *Introduction to Nuclear Engineering*. 2nd ed. Reading, Mass.: Addison-Wesley, 1983. Print.

LANL. "A General Monte Carlo N-Particle (MCNP) Transport Code." *Los Alamos National Laboratory: MCNP Home Page*. Web. 18 Sept. 2015.

Martin, William J., "Cinder Tutorial." Sandia National Laboratories. July 16, 2014.

Microsoft Visual Studio Professional 2013. Vers. 12.0.21005.1 REL. N.p.: Microsoft Corporation, 2013. Computer software.

Parma, Edward J., Thomas J. Quirk, Lance L. Lippert, Patrick J. Griffin, Gerald E. Naranjo, and S. Michael Luker. "Radiation Characterization Summary: ACRR 44-Inch Lead-Boron Bucket Located in the Central Cavity on the 32-Inch Pedestal at the Core Centerline." *Sandia Report* SAND2013-3406 (2013). Print.

Shleien, B. *The Health Physics and Radiological Health Handbook*. Olney, MD: Nucleon Lectern Associates, 1984. Print.

Shultis, J. Kenneth, and Richard E. Faw. *Radiation Shielding*. Upper Saddle River, NJ: Prentice Hall PTR, 1996. Print.

Snoj, Luka, and Matjaž Ravnik. "Calculation of power density with MCNP in TRIGA reactor." *Proceedings of the International Conference Nuclear Energy for New Europe*. 2006.

Sørum, Lars. "Characterization of MSW for Combustion Systems." *Sintef Energy Research*. 22 Feb. 2001. Web.

Stellman, Andrew, and Jennifer Greene. *Head First C#*. Beijing: O'Reilly Media, 2008. Print.

Turner, J. E. *Atoms, Radiation, and Radiation Protection*. New York: Pergamon, 1986. Print.

Williams, John G., Patrick J. Griffin, Donald B. King, David W. Vehar, Tim Schnauber, S. Michael Luker, and K. Russell DePriest. "Simultaneous Evaluation of Neutron Spectra and 1-MeV-Equivalent (Si) Fluences at SPR-III and ACRR." *IEEE Transactions on Nuclear Science* (2007): 2296-302. Print.