

Efficient Space Radiation Computation with Parallel FPGA

by Liwen Shih

NASA HAS, ACCORDING TO RECENT MISSION OBJECTIVES, developed several long-range goals for human exploration of space, including missions to the moon and Mars in the near future. In addition, the deep space environment and the long-term duration of spaceflight require a thorough understanding of all hazards and dangers that both astronauts and spacecraft materials will encounter over such missions. Space radiation is among the foremost dangers of such missions. As such, Space Radiation Modeling is of paramount concern to NASA design engineers, scientists, and mission planners when considering and estimating radiation dosage to both human beings and spacecraft materials. Space radiation consists of galactic cosmic rays, solar particle events, trapped radiation, and ions over a wide energy range. These ions penetrate spacecraft materials producing nuclear fragments and secondary particles that damage materials, tissue, and electronics. See Fig. 1.

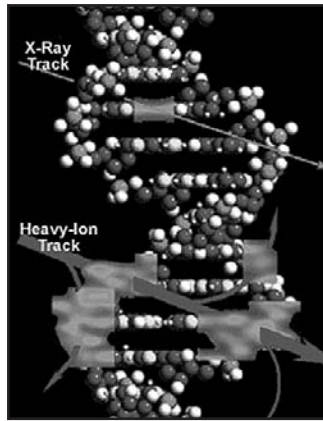


Figure 1. HZE particles cause more harm to DNA in living tissue than X-rays.


Among the various deep space particles, GCR particles modeled by HZETRN code are the most dangerous forms of known space radiation based on their relative biological effectiveness (RBE), an index of radiation dosage as it affects biological tissue.

In order to simulate the dosage effects of these High Charge and Energy Particles (HZE) NASA Langley Research Center (LaRC) scientists have created nuclear transport software to model these particles. This code is identified by its nomenclature, the High Charge and Energy Transport Code (HZETRN). It was developed in the early 1990s to provide dosage parameters to design engineers.

HZETRN was written in FORTRAN-77 for a VAX 4000 Mainframe platform. NASA LaRC made the source code available to UHCL in April 2005 hoping for urgently needed modernization and improvement in performance.

HZETRN Theory

A typical high-energy particle of radiation found in the space environment is ionized itself. As it passes through material such as human tissue, it disrupts the electronic clouds of the constituent molecules and leaves a path of ionization in its wake. These particles are either singly charged protons or more highly charged nuclei called “HZE” particles. HZETRN



ABSTRACT—Space radiation is likely to be the ultimate limiting factor for future deep space exploration by humans. NASA’s Radiation Dosage/Flux Software HZETRN is currently used to simulate high-energy nuclear transport across all materials being tested for a space mission. We are currently collaborating with NASA LaRC to establish an SAA to modernize and optimize radiation analysis.

Liwen Shih

models the dosage of these particles using deterministic models (as opposed to stochastic models such as the Monte Carlo method used in FLUKA code).

A fundamental summary of HZETRN has been provided by the physicists who developed the code as follows: HZETRN is high charge and energy transport code. This code calculates particle transport deterministically using approximate solutions to the Boltzmann transport equation. It is much faster than Monte Carlo codes and can be used to calculate the transport of the heavy ions found in the GCR as well as light ions and neutrons. The current version of HZETRN uses a straight-ahead approximation and models the 3-dimensional nature of radiation transport by calculating the transport along rays from a large number of different directions and averaging the fluencies.¹

The fundamental numerical algorithm used by HZETRN is a version of the time-independent Boltzmann Nuclear Transport Equation, which describes neutral and charged particle transport:

$$\vec{\Omega} \cdot \nabla \phi(\vec{x}, \vec{\Omega}, E) = \sum_k \int \sigma_{jk}(\vec{\Omega}, \vec{\Omega}', E, E') \phi_k(\vec{x}, \vec{\Omega}', E') d\vec{\Omega}' dE' - \sigma_j(E) \phi(\vec{x}, \vec{\Omega}, E) + O(\delta)$$

Although Monte Carlo simulation universally is viewed as the most accurate method for solving complex three-dimensional radiation transport problems, deterministic methods such as

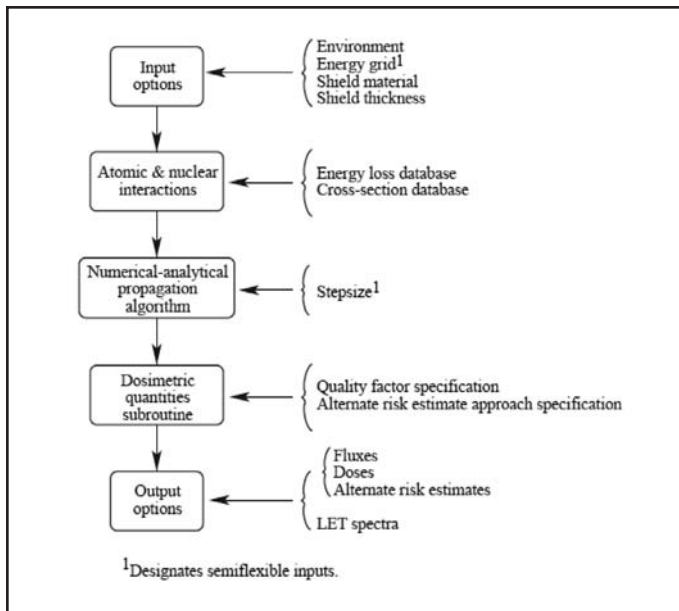


Figure 2. HZETRN Algorithm and Program Structure

HZETRN offer a faster alternative to Monte Carlo simulation. They often provide an accurate enough solution for engineering decisions or provide a ‘benchmark’ comparison.

HZETRN Algorithm and Program Structure

The HZETRN numerical algorithm involves interpolation, integration and extrapolation with the program controlling truncation and propagation errors. Flux and dosimetric results are output for six charge groups of HZE particles ranging from ($Z = 0, 1, 2, 3-10, 11-20$ and $21-28$). In addition, dose equivalent is also calculated for tissue material (which is approximated using water, since biological tissue is 80 percent water).

HZETRN code was developed in 1992 as a combination of previous nuclear transport codes by physicists at NASA-Langley and was written in FORTRAN-77 for the VAX platform. As such, it contains numerous inefficiencies in coding that increase the run time of the code and decrease performance.

Optimization of HZETRN code can have direct benefits in space radiation research. For example, HZETRN code is a major part of the MARIE (Martian Atmosphere Radiation Experiment) carried on the 2001 Mars Odyssey spacecraft. Current results indicate that HZETRN underestimates dosage data partly because of code inefficiency.

Methodology: High-performance/ Parallel HZETRN Code Optimization

Before the HZETRN source was available to UHCL for analysis in Spring 2005, HZETRN methodology was studied in search of run-time bottlenecks and code optimization possibilities. In particular, after communicating with scientists responsible for HZETRN development, we identified the numerical algorithms for interpolation and integration as being the optimal candidates for runtime bottlenecks, where integration also calls the interpolation routine. FPGA based numerical methods prototypes were developed for research into adapting the HZETRN numerical ker-

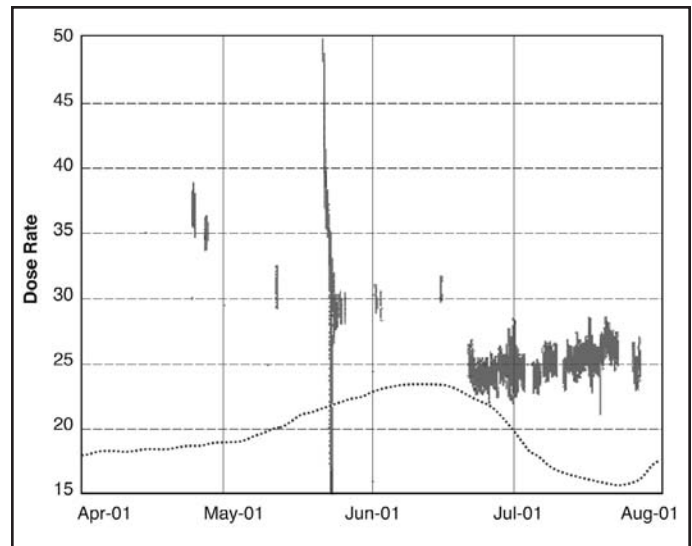


Figure 3. MARIE (gray spikes) vs. HZETRN (black curve) Data

nel onto an FPGA based computer, which showed an increased speed of up to 325 to 600 times for the bottleneck routines.

In April 2005, UHCL researchers were given approval to access the HZETRN1995 source code by NASA LaRC and were thus able to begin in-depth analysis of the code and undertake actual code optimization.

Steps toward Efficient Parallelization of HZETRN Code

The ultimate goal of our research involves speeding-up the runtime of the HZETRN code by parallelization of the code and by running it on a distributed computing environment

However, this process of parallelizing legacy code requires a number of intermediate steps: first, determining the static code structure, next evaluating runtime bottlenecks and inefficiencies/redundancies, then cleaning up the original code to prepare it for translation to a parallel environment, and, finally, conducting serial optimizations so that the code runs well even on a single processor. HZETRN, being written in FORTRAN-77, which is an outdated version of the FORTRAN, is not by itself directly suited to high-performance optimization:

1. Algorithm review (not well documented).
2. Analysis of source code and data files (Data files contain sparse matrices amenable to performance improvement).
3. Portability study (not portable to modern platforms even though it was designed to be platform-independent).
4. Static analysis (We generated a detailed HTML report documenting HZETRN source code functions and structure of subroutine calls).
5. Runtime analysis (Analysis revealed that the PHI interpolation function is the major bottleneck function. The natural logarithm intrinsic function is also a performance issue).
6. Serial optimization of code (FPGA to implement core routines in one instruction cycle).
7. Parallel optimization of code (Parallel FPGA and/or Cluster/Grid Computing needed).

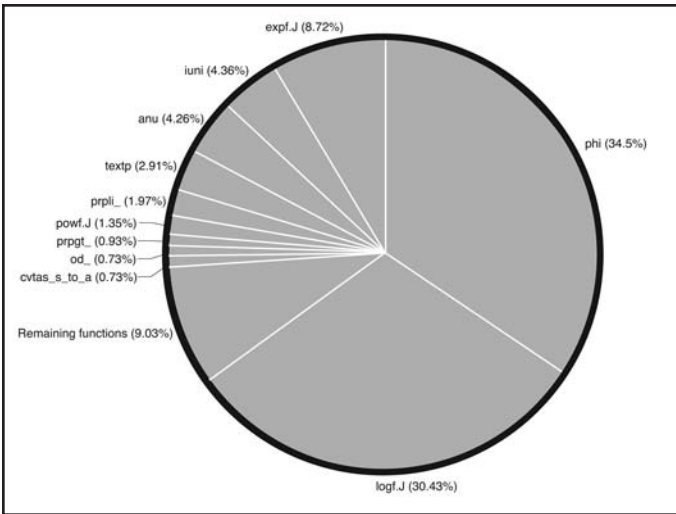


Figure 4. HZETRN Runtime Profile Analysis

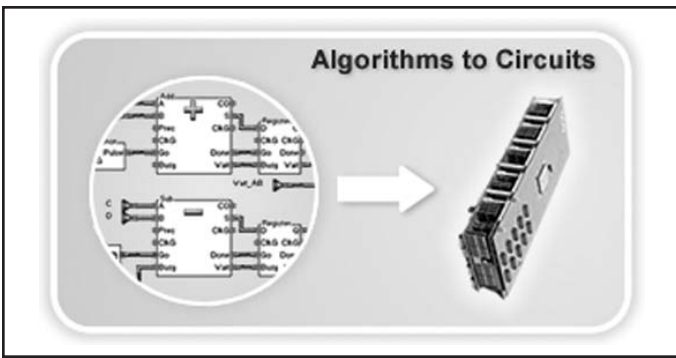


Figure 5. Algorithm-Specific FPGA

HZETRN Runtime Analysis

We examined the runtime performance of the standard HZETRN code. Statistics included average runtime and most called functions. Not surprisingly, analysis confirmed our previous study even before we received the source code, that interpolation and intrinsic math function calls consumed most of HZETRN runtime.

Function Bottlenecks: The PHI Interpolation Function

More than 70 percent of runtime is devoted to three functions with the PHI function taking up most of the runtime (approximately 34.5 percent time spent in calls to this interpolation function).

Therefore, we determined that the PHI interpolation function was the most attractive target to aim for in code optimization.

Starting with the PHI function, we removed extraneous function calls, cleaned up “messy code,” SAVE statements, and evaluated the use of faster intrinsic functions for *expf* and *logf* to replace the archaic ALOG function called in HZETRN. The effort resulted in a 10 percent overall increase in runtime performance improvement.

High-Performance Intrinsic Math Library (instead of standard math functions)

HZETRN code performance was evaluated using an Intel 9.0 FORTRAN compiler on an Intel Pentium platform. Since HZETRN functions make an extremely large number of intrinsic function calls (with the natural log being the major math function called), we considered using high performance versions of these intrinsic math functions.

For example, there are substantial performance advantages over scalar implementations with Intel® MKL 8.0 VML (Vector Math Library) functions that operate on real vector arguments versus LibM functions on Pentium 4. For example Exp and Log10 can be improved from ~90 to ~30 CPE, where Clocks Per Element (CPE) is the number of clock cycles that are required to execute a single element of the vector.²

HPC Technology

The performance profile of HZETRN code determined that 34.5 percent of the run time spent on the PHI/interpolation function confirms previous results. There is much inefficiency and redundancy in the code itself. By cleaning up minor programs, we obtained an overall speedup of about 10 percent. Much more can be improved if the PHI Function is updated. The current running environment (requiring changing parameters in the code, recompiling, then running again) is not user friendly. Next, we continue to explore parallel FPGA and parallel grid/clusters to improve the critical portion of the computation. NASA LaRC can arrange to provide Parallel FPGA hypercomputers for our study.

Parallel FPGA – Hypercomputing (supercomputing in a box): Xilinx’s radiation tolerant FPGAs are playing a critical role in the successful NASA/JPL Mars exploration mission for the Spirit Rover and the Opportunity MER rover. Cray XD1™ supercomputers will support the latest generation of Xilinx® Virtex-4 FPGAs, including the Virtex™-4 LX and SX platforms. The Starbridge Hypercomputer uses high-end Xilinx Virtex-II FPGAs, each containing more than six million gates. Various Hypercomputer configurations comprise between 7 and 22 Virtex-II FPGAs, providing from 36 million gates to 124 million gates, that can be employed simultaneously to implement massively parallel computational pipelines for powerful algorithm execution.

NASA – Scientists at NASA’s Langley Research Center use a Starbridge Hypercomputer for projects that push traditional computational boundaries, including radiation analyses, digital signal processing, and atmospheric studies. NASA nuclear physics engineers need our computer engineering expertise to help them apply these powerful emerging technologies effectively. LaRC has indicated the possibility of providing UHCL with a hypercomputer HC-38 with 10 Virtex-II FPGAs (60 million gates).

Parallel Clusters: Texas Learning & Computational Center (TLC²)

The computational resources of the Advanced Computing Laboratory at the UH Central Campus is considered the “flag-

ship” of the Texas Learning & Computational Center (TLC²). Three computer clusters are available: Two have Itanium2 CPUs as the workhorses (the Atlantis and Eldorado clusters); and the computational power of the third, the small Medusa cluster, utilizes the Opteron CPUs. The Atlantis and Eldorado feature distributed memory architecture with 4 GB of memory per node. With the Itanium2’s excellent floating point and good integer performance and a relatively fast network interconnect, the Atlantis/Eldorado systems are suitable for almost any kind of scientific computing that can be efficiently parallelized/partitioned on a Linux cluster. The Atlantis/Eldorado clusters can be “clustered” together also into one, resulting in 445 CPUs. Perhaps their only limitation for this sized cluster is the lack of a high-speed file system such as a GPFS/Lustre/PVFS file system.

The third cluster, Medusa, is a 21-node Opteron cluster that is partially owned by the Advanced Computing Research Laboratory and the Texas Learning and Computation Center. The small size of the Medusa cluster does not allow its use for jobs that require a large number of nodes, but the Opteron CPUs do well at both integer and floating point calculations. The Medusa Cluster would be effective for a problem that might be latency-sensitive and/or bandwidth-intensive, but does not require a large amount of CPUs. A job that the user might consider to be I/O bound may do well on Medusa, while using parallel I/O like MPI I/O or similar.

Preliminary Result – Parallel Space Radiation

Space radiation work at UHCL so far is summarized:

2004: Performed general space radiation problem study and made connection with LaRC researchers.

Spring 2005: FPGA prototype achieved up to 600 times speed up on HZETRN bottleneck routines. Obtained HZETRN1995 source code in 4/2005.

Fall 2005: HZETRN portability, performance profile study:

1. HZETRN is not portable, most encounter syntax error in PHTOD routine.
2. HZETRN Performance Profile: Bottleneck routines are PHI, then logf, then expf. (All three routines take up ~70 percent of HZETRN run time).
 - 2a. Hand optimized PHI function (10 percent speedup in HZETRN run time).
 - 2b. logf and expf: Table lookup did not help memory time increase.
3. Opportunities available for sparse data matrix techniques and HZETRN application-specific limited data value range.
4. Need modulization: We tried to modularize the code

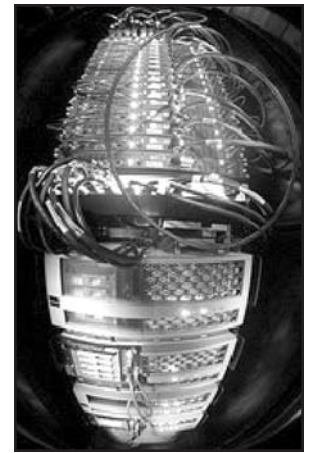


Figure 6. (l.) Atlantis Cluster (Gimli), (r.) Medusa at TLC²

for easier navigation of the source code. We also tried to organize the source in HTML and spreadsheet formats. (HZETRN2005 is trying to make the code better modularized.)

5. Need better user interface: A parameter change is required in the source code to recompile between runs. (NASA is also fixing this problem now.)

Spring 2006: After this detailed performance profiling and diagnosis of the HZETRN, we started to attack these bottlenecks by approaching the treatment of both top-down parallel grid computing among the particles tracking and bottom-up FPGA in time-consuming routines. Our UHCL HPC team continues to meet weekly to analyze the HZETRN code for improvement. A spreadsheet to categorize all routines is maintained. Large flow charts were generated by software analysis tools for inspection.

Summer 2006: Our UHCL HPC team used the parallel computer clusters to run the HZETRN in parallel. OpenMP was used first and MPI has just been started. A study of the more accurate but time consuming FLUKA Monte Carlos method was also performed during the summer.

Fall 2006: The FPGA developed in the spring of 2005 was specially designed for a reduced 8-bit floating point format to accommodate the limitation of pins. A general purposed IEEE-754 32-bit Floating Point Core has been designed and specified in VHDL (VHSIC Hardware Description Language). It is systematically designed to scale to future needs. It will be further tested and implemented on FPGA hardware. Eventually it will be extended to perform the bottleneck function of interpolation. During the FPGA redesign an error in the 2005 VHDL was found and corrected, where two variables were mistakenly swapped.

In other words, the HPC team at UHCL has completed most of the HZETRN “diagnosis” and has begun the parallel modernization “treatment” of the code. Support for this effort is still greatly needed to demonstrate this effort. To conduct syntax code thread analysis, we need to perform data flow analysis either from the code or algorithm design. To further optimize parallel mapping

and semantic application-, methodology- and algorithm-specific thread mapping optimization, a better understanding of the numerical models of nuclear physics theory is needed. Help is being sought from LaRC nuclear physics researchers.

Discussion

Since the performance profile revealed bottleneck routines such as PHI, we can try to model/simulate/estimate how much speedup can be achieved for various ways of improvement and determine the most promising direction for research:

- Replace PHI and other bottleneck routine with FPGA or updated algorithms/data-structure/operations.
- Produce data-flow and control flow charts of HZETRN.
- Perform data-flow/thread analysis to determine the best mapping (data/function partition) from HZETRN to either cluster, grid, parallel FPGA (Viva/Hyper-computing) or VHDL/Xilinx ML-310 with bottleneck code on FPGA and non-bottleneck code compiled to the built-in power-PC core on FPGA.

If we work more closely with NASA nuclear physics scientists to help us better understand the physics/math application of the radiation model, we can further improve the algorithm, data structure, and code design.

If the application-specific data range were known, we could replace some complex computations with simple tables (while controlling memory access time increase), or we could replace current methodology with faster equivalent calculation for the hardware.

The mode of execution can be improved also, such as conducting a multiple run in one or by reusing partial results from a previous run.

If we can better understand the radiation model, we can monitor/deduce the cumulative computation errors to improve the precision.

The current model predicting accuracy is low, and three days of data may require up to a day of CPU time to analyze. When the speed is improved, we might be able to improve the data size, resolution, to expand beyond straight-ahead simplification for better accuracy.

Conclusions

NASA LaRC is currently working on establishing a Space Act Agreement (SAA) between NASA and UHCL for a long-term collaboration to enable computer engineers to work with NASA nuclear scientists and engineers to modernize computer application and optimize space radiation computation. The SAA has been reviewed by UH System general counsel and is currently being revised at NASA Langley. After SAA is signed, we expect to gain access to the newest HZETRN2005 code for optimization within two weeks.

An essential step toward a more efficient and cost effective solution to the radiation-shielding problem is the development of accurate, efficient and fast tools for modeling radiation transport. We hope that HZETRN code improvement can benefit design and engineering of lighter and more cost-effective shielding material for use in NASA spacecraft.

As earth's ozone depletion continues, space radiation study

could lead to dual-use countermeasures that will, in turn, protect human health from radiation/aging effect in general, e.g., slowing down cataract development. Other evolving critical medical technology, e.g., proton cancer radiation treatment, is being realized for its cures.

Personnel

Computer Engineering Professor Liwen Shih specialized in optimizing computation-machine matching with AI and parallel techniques. Dr. Shih regularly teaches High-Performance Computer Architecture, Machine Intelligence, and Parallel Processing. Uniquely innovated approaches in knowledge delivery often result in full enrollment despite the national downtrend in computer science majors. Dr. Shih has supervised all or most of UHCL's recent computer engineering master's capstone research projects. The recent emphasis in critical medical applications has resulted in several presentations in the worldwide forum.

Dr. Shih and students successfully improved the performance of scientific/image modeling, including the speedup of Medical Image Texture Analysis from eight days CPU time on VAX per image to within 10 seconds on a PC platform (70,000 times speedup). Dr. Shih and students are very enthusiastic in applying the high-performance computing expertise to participate in the great space expedition to the moon and Mars.

Acknowledgments

NASA LaRC researcher Dr. Robert Singletery Jr. and the NASA JSC MARIE Team (Dr. Premkumar Saganti of Lockheed-Martin and physics professor at Prairie View A&M) worked with us to understand the space radiation modeling and analysis. NASA LaRC will provide UHCL researchers the parallel FPGA Hypercomputer STARBRIDGE HC-38 with the VIVA software. The School of SCE at UHCL provides faculty and students space and computer needs, including the Athena TxGrid Texas Education Grid and FPGA lab. Dr. Shih and students/associates have access to labs and clusters/grids at such resources as the Texas Education Grid, UH TLC², UT ACC, etc.

References

Singletery, R. C. et al. *Physica Medica* Vol. XVII, Suppl. 1: 90-93; 2001.

M. S. Cloudsley et al. NASA TP-2000-209856, 2000.

[MARIE 05] "Martian Radiation Environment Experiment" MARIE, Oct. 27, 2003, NASA-JSC, March 7, 2005 <<http://marie.jsc.nasa.gov/>>.

"Hypercomputer Parallel FPGA with VIVA Graphic Programming" <<http://www.starbridgesystems.com/hypercomputers.htm>>

"Starbridge Hypercomputers Complement Linux Clusters" <<http://www.linuxhpc.org/stories.php?story=04/05/24/7059836>>

Xilinx ML410 Embedded kit with Power PC core <http://www.xilinx.com/publications/magazines/emb_02/emb02-boards.htm>

Advance Computing Laboratory at Texas Learning & Computational Center (TLC2) <http://www.tlc2.uh.edu/Facilities/Computational_Resources>

Shih, L., T. Gilbert, A. Kadari and S. Kodali, "High-Performance Martian Space Radiation Mapping," NASA/UHCL/UH-ISSO Y2004 annual report, Spring 2005, PP. 145-149. <<http://www.issu.uh.edu/publications/A2004/2004-145LS.pdf>>

Shih, L., S. J. Larrondo, K. Katikaneni, A. Khan, T. Gilbert, S. Kodali, and ., Kadari, "High-Performance Martian Space Radiation Mapping" NASA/UHCL/UH-ISSO Y2005 annual report, Spring 2006, pp. 121-122 <http://www.issu.uh.edu/publications/A2005/2005_121_Shih.pdf>

Wilson, J. W., F. F. Badavi, F. A. Cucinotta, J. L. Shinn, G. D. Badhwar, R. Silberberg, C. H. Tsao, L. W. Townsend, and R. K. Tripathi, "HZETRN: Description of a Free-Space Ion and Nucleon Transport and Shielding Computer Program," <<http://techreports.larc.nasa.gov/ltrs/PDF/NASA-95-tp3495.pdf>>

Publications & Presentations

Gilbert, T. and L. Shih. "High-Performance Martian Space Radiation Mapping," *Proc. Computer Application Conference, IEEE/ACM/UHCL 2005*

Johnson, A. (supervised by Liwen Shih). "32-bit IEEE Compliant Floating Point FPGA Core Design," UHCL Master Capstone Project Report & Presentation, Fall 2006.

Kadari, A., S. Kodali, T. Gilbert and L. Shih. "High-Performance Space Radiation Analysis with FPGA," *Proc. Computer Application Conference, IEEE/ACM/UHCL 2005*

Kodali, S., A. Kadari, T. Gilbert, and L. Shih. "Space Radiation Analysis with FPGA" UHCL Master Capstone Project Report and Website, Spring 2005 <<http://dcm.cl.uh.edu/c4230s4kodalis/FPGA/Index.html>>.

Larrondo, S. and A. Johnson (supervised by Liwen Shih). "Space Radiation HZETRN Architecture and FPGA System Selection via Weighted Scores," UHCL Parallel Processing Project Report & Presentation, Fall 2006.

Shum, V., S. Strasser and R. Chua (supervised by Liwen Shih). "Space Radiation HZETRN on Parallel Cluster," UHCL Parallel Processing Project Report & Presentation, Spring 2005.

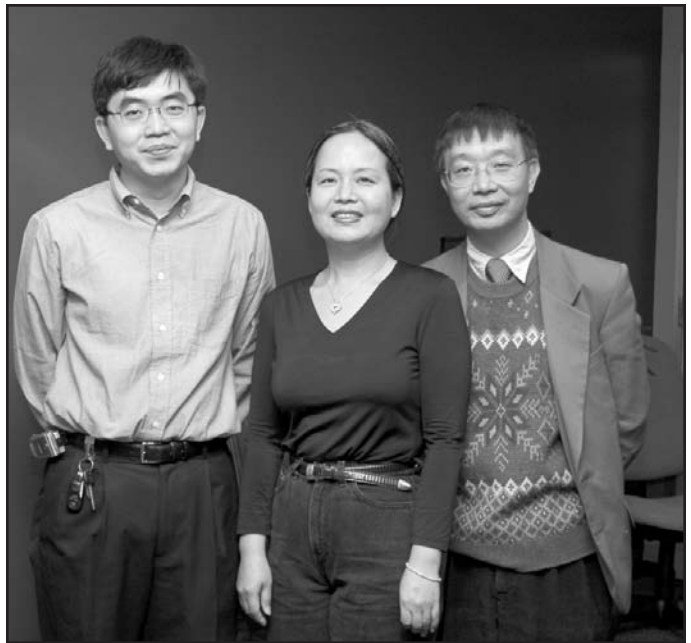
Funding

Shih, L., R. Singleterry, Jr. "Starbridge HC-38 (Viva FPGA machine)" and "HZETRN2005 (the newest version of NASA Space Radiation code)" – an on-going Space Act Agreement (SAA) between NASA-Langley and UHCL (Shih) to analyze/improve NASA's HZETRN2005 code that utilizes the newer computational devices. (*in progress*, currently being reviewed/revised by UH/NASA)

Shih, L. "Parallel Space Radiation Computation with Cluster and FPGA," NASA Langley Contract Statement of Work, Jan. 1–May 31, 2007, \$15,000 (*submitted*).

Shih, L., "Efficient Space Radiation Computation with Parallel FPGA" ISSO Mini-Grant 2006, \$16,950

Shih, L., "Partitioning Space Radiation Analysis/Simulation Code for High-Performance Execution with Parallel Computing Techniques," granted support for student RA, UHCL Faculty Research and Support Fund, June–December



FLEXIBILITY—Researchers seek to mathematically analyze and numerically simulate the solution of non-smooth mechanical problems seeking higher levels of freedom. With Dr. Shih are Romeo Chua from the Phillipines (*I.*), M.S. student in software engineering, and Victor Shum from Hong Kong, Ph.D. student in aerospace engineering system assembly, with the ARES Corp.