Atmospheric and
Environmental Research, Inc.

Final Technical Report

For the Project:

Enhancing Cloud Radiative Processes and Radiation Efficiency in the Advanced Research Weather Research and Forecasting (WRF) Model

Supported by:

U.S. Department of Energy
Office of Science
Office of Biological and Environmental Research
Climate and Environmental Sciences Division
Earth System Modeling Program
Regional & Global Climate Modeling Program
DOE Technical Contacts: Dr. Dorothy Koch, ESM Program Manager,
Dr. Renu Joseph, RGCM Program Manager

DOE Grant Number: DE-SC0007038 (ER65306)

Prepared by:

Michael J. Iacono, Principal Investigator
Atmospheric and Environmental Research
131 Hartwell Avenue, Lexington, Massachusetts 02421-3126
Tel: 781-761-2288; Fax: 781-761-2299; E-mail: miacono@aer.com

For the Project Period:
15 September 2011 - 14 December 2014

9 March 2015

Table of Contents

1. Overview

The objective of this research has been to evaluate and implement enhancements to the computational performance of the RRTMG (*Iacono et al.*, 2008; *Mlawer et al.*, 1997) radiative transfer option in the Advanced Research version of the Weather Research and Forecasting (WRF) model (*Skamarock et al.*, 2008). Efficiency is as essential as accuracy for effective numerical weather prediction, and radiative transfer is a relatively time-consuming component of dynamical models, taking up to 30-50 percent of the total model simulation time. To address this concern, this research has implemented and tested a version of RRTMG that utilizes graphics processing unit (GPU) technology (hereinafter RRTMGPU) to greatly improve its computational performance; thereby permitting either more frequent simulation of radiative effects or other model enhancements. Team members included the Principal Investigator, Michael J. Iacono (AER), who is expert in radiative transfer development and application to general circulation models (GCMs) and with dynamical model evaluation, Thomas Nehrkorn (AER), who is expert in the WRF modeling system, Dave Berthiaume (AER), who has developed RRTMGPU under separate funding, and John Michalakes (NOAA), who is working actively in the area of multi- and many-core acceleration for strong scaling of geophysical models (*Michalakes and Vachharajani*, 2008) and who has served as the contact for this project to the WRF Development Group. During the early stages of this project the development of RRTMGPU was completed at AER under separate NASA funding to accelerate the code for use in the Goddard Space Flight Center (GSFC) Goddard Earth Observing System GEOS-5 global model. It should be noted that this final report describes results related to the funded portion of the originally proposed work concerning the acceleration of RRTMG with GPUs in WRF.

As more accurate and sophisticated algorithms are developed to simulate physical processes in global models, it is critical that their efficiency also be considered. If used effectively, GPUs can provide a substantial improvement in speed by supporting the parallel computation of large numbers of independent radiative calculations (*Michalakes and Vachharajani*, 2008). As a k-distribution model, RRTMG (see Section 4.3) is especially well suited to this modification due to its relatively large internal pseudo-spectral (g-point) dimension (of 140 in the longwave and 112 in the shortwave) that, when combined with the horizontal grid vector in the dynamical model, can take great advantage of the GPU capability. RRTMG utilizes the Monte-Carlo Independent Column Approximation (McICA; *Barker et al.*, 2007; *Pincus et al.*, 2003), a statistical method for represent sub-grid cloud variability that also operates over the g-point dimension, and the sub-column generator required for McICA has also been accelerated. Thorough testing has been performed to ensure that RRTMGPU improves model run time while having no significant impact on calculated radiative fluxes and heating rates relative to RRTMG.

2. Enhancing RRTMG Radiation Efficiency: RRTMGPU

The great advantage of GPUs over parallel processing on modest CPU systems is the ability to process over a large number of elements simultaneously. In order to fully utilize this advantage, the program being accelerated must be parallelizable over multiple dimensions of as large a size as possible. RRTMG was developed to be a callable subroutine that essentially processes a single atmospheric column per call from within a larger global dynamical model. Figure 1 is a diagram of the basic structure of the subroutines in RRTMG_LW and SW, and the placement of the loop over atmospheric columns outside of most of the code is indicated. Only the vertical layer loop is utilized inside the subroutines shown in Figure 1. Often the vertical dimension in dynamical models is of modest length (30-90 layers), and parallelizing over this dimension alone would not be cost effective on a GPU. RRTMG is well suited to acceleration over a large number of elements, due to its internal pseudo-spectral g-point dimension, which is of length 140 in the longwave code and 112 in the shortwave code. G-points are the quadrature points that are used to integrate the k-distribution cumulative probability functions that represent the combined gaseous absorption from major and minor gas species within each spectral band and atmospheric layer. This vector of order 100 greatly expands the available number of elements over which the radiative transfer code can be parallelized. A more substantial benefit can be attained by the horizontal dimensions of a typical dynamical model, which can number of order 10000 or more over a geographic region or globally. In RRTMG, either one or both of the horizontal dimensions are brought into the code only at the interfacing level, and one column from the spatial dimension is passed into the radiative transfer at a time.

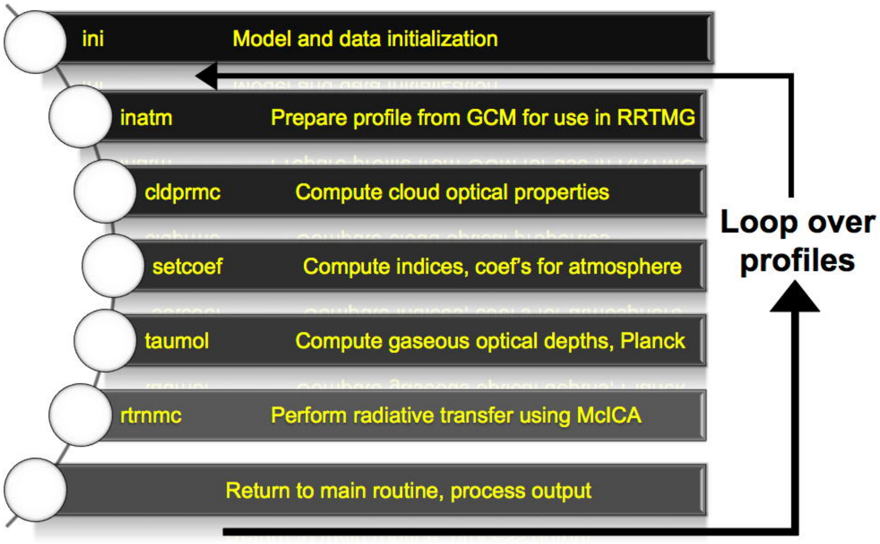
ini
Model and data initialization
inatm
Prepare profile from GCM for use in RRTMG
cldprmc
Compute cloud optical properties
setcoef
Compute indices, coef's for atmosphere
taumol
Compute gaseous optical depths, Planck
rtrnmc
Perform radiative transfer using McICA
Return to main routine, process output
Loop over profiles

Figure 1. Diagram showing the sequence of subroutine calls in RRTMG and the location of the outer loop over atmospheric columns. The vertical layer dimension is interior to each subroutine as needed.

Adapting RRTMG for use on GPU hardware required numerous changes to the code to optimize performance in this context. The essential modification was to refactor the code to allow parallelization over vertical layer, the g-point dimension, as well as both horizontal spatial dimensions so that an entire block of grid elements can be passed to the GPU for processing. Figure 2 illustrates the difference in the overall computational approach between RRTMG and RRTMGPU. It must be emphasized that the radiative physics represented by each model is identical, and it will be demonstrated in Section 4 that in additional to the improved performance the expectation of negligible to no impact on the calculated fluxes and heating rates of running on the GPU has been realized in the WRF application.

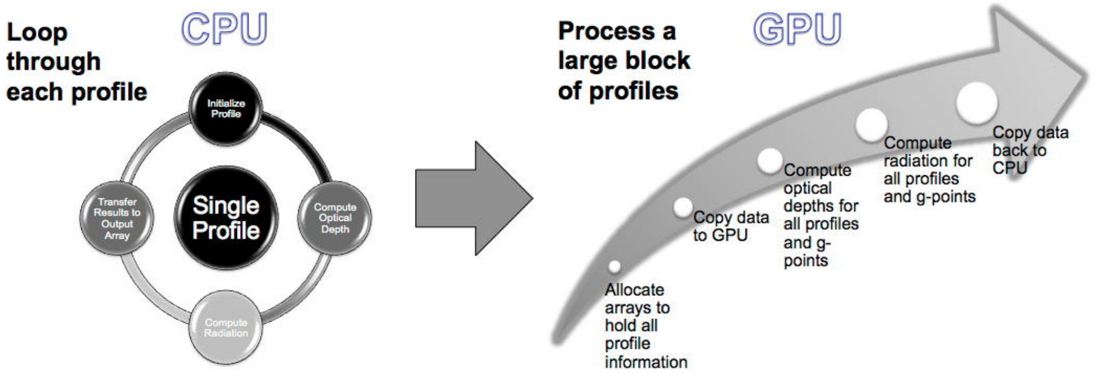
Loop through each profile
CPU
Initialize Profile
Single Profile
Compute Optical Depth
Compute Radiation
Transfer Results to Output Array
Process a large block of profiles
GPU
Allocate arrays to hold all profile information
Copy data to GPU
Compute optical depths for all profiles and g-points
Compute radiation for all profiles and g-points
Copy data back to CPU

Figure 2. Diagram showing the essential difference in processing approaches between the CPU version of the radiation code (RRTMG) and the GPU-accelerated version (RRTMGPU).

The specific code revisions include rewriting sections of the code (primarily the calculation of optical depths) so that the code can be parallelized over the g-point dimension. In addition, where necessary, arrays were padded to be multiples of 32, which is the size of a warp on a GPU, and were reordered so that the fastest changing dimension would coincide with the thread layout to enable efficient memory coalescing. In addition, several exponential lookup tables, which had been added to RRTMG to avoid the computational expense of performing exponentials, were removed for RRTMGPU, since the table lookup was more costly than the exponential in the GPU context, since it prevented parallelization. Also, profile partitioning was implemented using the MPI API and multiple streams to allow running on multiple GPUs in parallel. Despite these revisions, portions of the code cannot be parallelized, and thus RRTMGPU must be considered a transitional code. For example, the longwave code was prepared initially with CUDA Fortran, while the shortwave code utilized later capabilities including openACC, and thus each model is formatted somewhat differently. In addition, both the longwave and shortwave codes are currently restricted to being used with the PGI compilers and Nvidia GPU hardware. Future plans, detailed in Section 6, will result in a complete rewrite of the codes to produce more generalized and consistent accelerated models.

The NCAR computing system 'caldera' was utilized for testing RRTMGPU initially in stand-alone mode, since 'caldera' is one of the primary systems provided by NCAR to test and utilize GPU accelerated code. The NCAR system 'caldera' has 16 nodes with two eight-core Xeon (Sandybridge) processors and two GPGPUs per node. In the early stages of this project, the GPU hardware available in 'caldera' was the Nvidia Tesla M2070-Q model with compute capability 2.0. During 2014, the 'caldera' GPU hardware was upgraded to the Nvidia Tesla K20X with compute capability 3.5. The performance of RRTMGPU was tested in stand-alone mode using both hardware configurations for varying numbers of atmospheric columns.

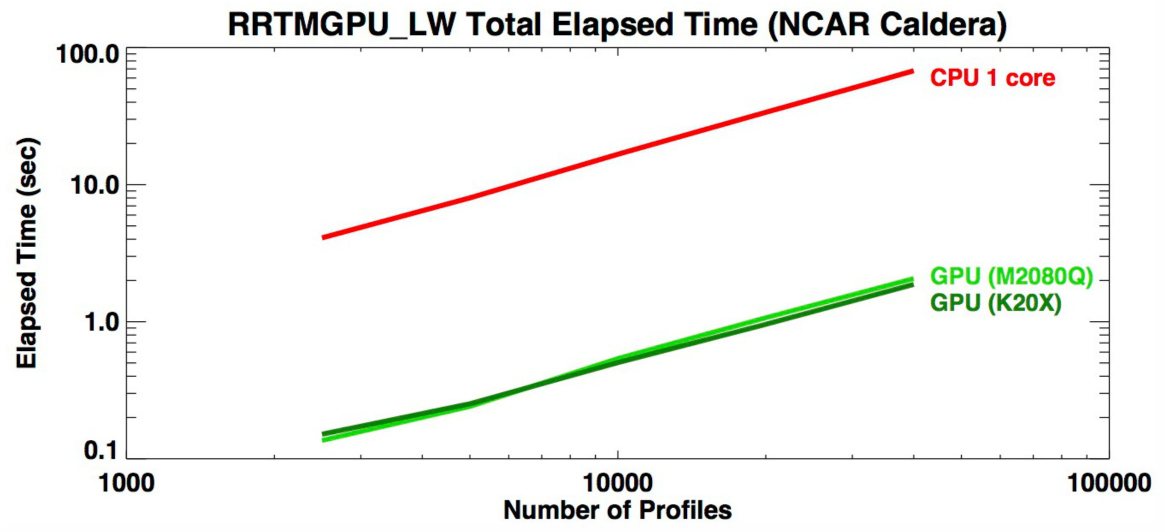

Figure 3. RRTMGPU_LW elapsed time as a function of the number of profiles (atmospheric columns) processed when running the code for 72 atmospheric layers on the NCAR system 'caldera' entirely on a single CPU processor (red), on the M2080-Q GPU (light green) and on the K20X GPU (dark green).

The result of testing the efficiency of the RRTMG codes in stand-alone mode on the 'caldera' system on both the CPU and on the GPU is summarized in Figure 3 for the longwave code and in Figure 4 for the shortwave code. The longwave code runs on the GPU more than an order of magnitude faster on either GPU relative to a single CPU for a few thousand profiles or up to 40,000 profiles, which is a typical grid size for a global or regional model. A small improvement in efficiency is noted with the newer K20X GPU hardware relative to the M2080-Q at higher numbers of profiles. The shortwave code was tested in a similar manner and also tested on the CPU using both eight and 16 processors as shown in Figure 4. As expected, the elapsed time on the CPU is considerably better when the shortwave model is run on eight or 16 processors rather than a single core, though the performance on either GPU is still an improvement by a factor of three or more compared to multiple CPU processors.

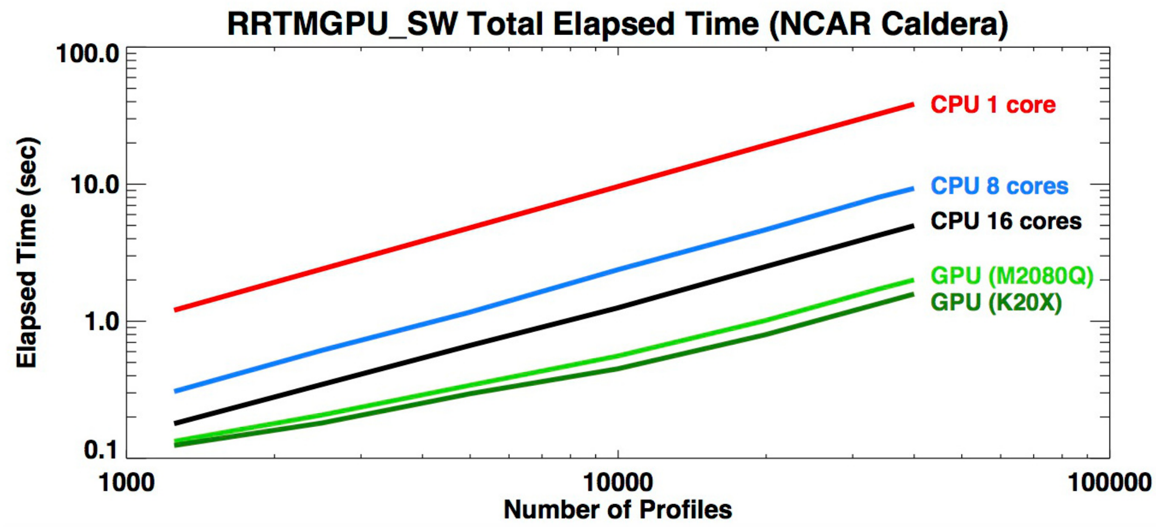

Figure 4. RRTMGPU_SW elapsed time as a function of the number of profiles (atmospheric columns) processed when running the code for 72 atmospheric layers on the NCAR system 'caldera' entirely on a single CPU processor (red), on 8 CPU processors (blue), on 16 CPU processors (black), on the M2080-Q GPU (light green), and on the K20X GPU (dark green).

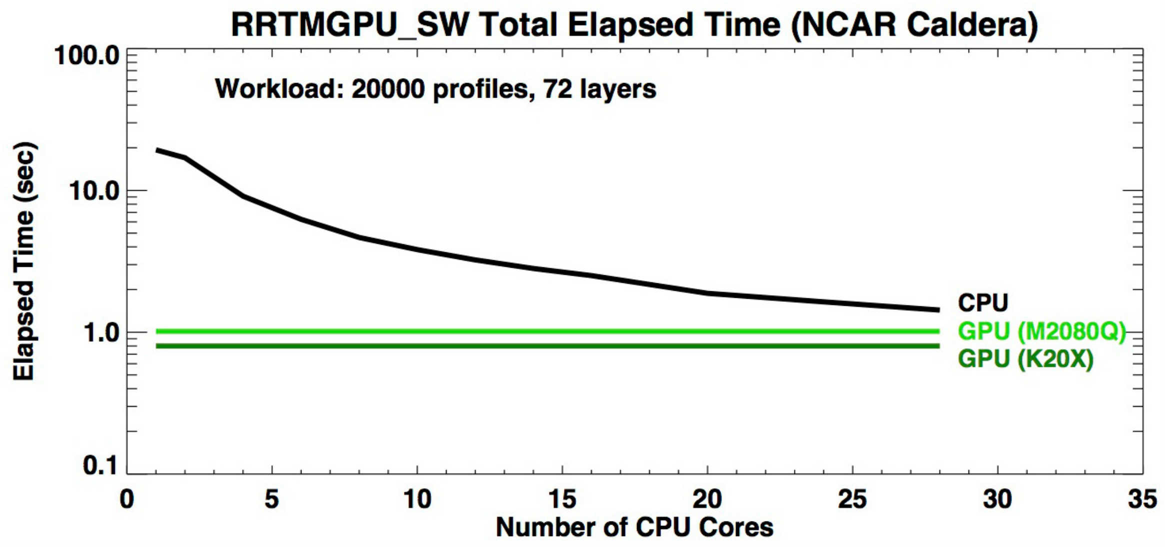

Figure 5. RRTMGPU_SW elapsed time as a function of the number of CPU processors used when running the code for 72 atmospheric layers on the NCAR system 'caldera' (black). The elapsed times using the M2080-Q GPU (light green) and the K20X GPU (dark green) for a workload of 20000 profiles are also shown for comparison.

The shortwave code efficiency was also tested on the CPU for a fixed workload (20000 profiles) for a varying number of processors from 1 to 28 as shown in Figure 5. Although ‘caldera’ has two processors on each of its 16 nodes, which suggests that the code could be tested on as many as 32 processors, it was found that model efficiency began to diminish when running with 30 or more processors. Figure 5 shows that the GPU performance has the advantage over the CPU when smaller numbers of processors are used, though the CPU can approach the GPU performance in this context on ‘caldera’ when nearly all the available cores are dedicated.

3. Application of RRTMGPU to WRF

Implementation of the GPU accelerated radiation into WRF required several steps. First, the numerous original source files and modules had to be consolidated into the two source modules utilized for RRTMG in WRF, one for the longwave code and another for the SW. These WRF source files include all radiation specific subroutines along with all customized interfacing required to run the codes in WRF. The latter executes the definition and preparation of the numerous input parameters needed by the radiation code including all associated unit conversions and other array and variable name conversions.

Since RRTMG was originally implemented in WRF_v3.1, several modifications to the interfacing and physics treatment were added by NCAR, and these changes had to be transferred into RRTMGPU to ensure no loss of functionality. Therefore, the next step required restoring these NCAR-sponsored code changes into the GPU radiation modules. Among these changes was the addition of the treatment of snow water path as a separate phase of cloud water (distinguished from liquid or ice cloud particles) for some configurations. The snow water path and associated snow particle size were passed into RRTMGPU through the existing interfacing and into the cloud property subroutine for inclusion of their cloud radiative effects.

4. Performance Impact of Accelerated Radiation in WRF

WRF_v3.6.1

Over the course of this project two relevant changes, one related to software and the other related to hardware, occurred that had to be accounted for when testing the GPU radiation codes in WRF in the NCAR computing environment. First, WRF itself was upgraded from v3.5.1 to v3.6 to v3.6.1 during 2014. Since RRTMGPU had originally been implemented and tested in WRF v3.5.1 during 2013, a second implementation was completed to integrate RRTMGPU into WRF_v3.6.1 during Fall 2014 to accommodate the transfer of the code to NCAR for application to the next version of WRF in 2015. This second implementation was not trivial, since several changes had been made by NCAR to the radiation interfacing for WRF_v3.6.1 that also had to be retained for the GPU-accelerated radiation.

The NCAR computing system ‘caldera’ was utilized for testing RRTMGPU in WRF, since ‘caldera’ is one of the primary systems provided by NCAR to test and utilize GPU accelerated code. In the early stages of this project, the GPU hardware available in ‘caldera’ was the Nvidia Tesla M2070-Q model. As note earlier, the ‘caldera’ GPU hardware was upgraded during 2014 to the Nvidia Tesla K20X. The performance of RRTMGPU in WRF was tested using both hardware configurations, with a notable improvement in efficiency seen with the newer GPU hardware within WRF.

WRF RRTMG_CPU/RRTMGPU Performance Examples on NCAR/Caldera (Tesla M-2080Q GPU)					
(1 Core; “serial”) WRF/CPU + RRTMG			**(8 Cores) WRF/CPU + RRTMG**		
Model	Elapsed Time (sec)	Time Fraction vs. WRF	Model	Elapsed Time (sec)	Time Fraction vs. WRF
LW	904.3	0.28	LW	116.6	0.23
SW	643.6	0.20	SW	90.2	0.18
LW+SW	1547.8	**0.48**	LW+SW	206.8	**0.40**
WRF	3210.7	1.00	WRF	512.6	1.00
(1 Core) WRF/CPU + RRTMGPU			**(8 Cores) WRF/CPU + RRTMGPU**		
Model	Elapsed Time (sec)	Time Fraction vs. WRF	Model	Elapsed Time (sec)	Time Fraction vs. WRF
LW	70.2	0.04	LW	70.2	0.16
SW	55.4	0.03	SW	55.4	0.13
LW+SW	125.6	**0.07**	LW+SW	125.6	**0.29**
WRF	1944.2	1.00	WRF	431.4 (estim.)	1.00
Model	**CPU/GPU Time Ratio**	**GPU/CPU Time Ratio**	Model	**CPU/GPU Time Ratio**	**GPU/CPU Time Ratio**
LW	12.9	0.08	LW	1.7	0.60
SW	11.6	0.09	SW	1.6	0.61
LW+SW	**12.3**	**0.08**	LW+SW	**1.7**	**0.61**
WRF	1.7	0.61	WRF	1.2	0.84

Table 1. Elapsed time for RRTMG_LW, RRTMG_SW, the LW and SW total, and the WRF total for all codes running on the CPU (top rows), and elapsed time for WRF running on the CPU and for RRTMGPU_LW, RRTMGPU_SW, and the total LW and SW all running on the M2080-Q GPU (center rows) using a single CPU (left columns) and using 8 CPU cores (right columns). Also shown are the fractions of time for the radiation components relative to the total WRF elapsed time (total radiation time in red), and the ratios of elapsed time for runs using only the CPU to those running the GPU radiation.

The GPU accelerated radiative transfer was tested in WRF both to quantify any changes in the model's computational performance and to demonstrate that any changes in the calculated radiative fluxes and heating rates were below a very small threshold. A simulation grid was prepared to extend over the entire continental United States with a horizontal grid resolution of 4 km (corresponding to a total of 33750 grid cells) and 27 layers in the vertical. A simulation length of 24 hours (18 UTC 9 January to 18 UTC 10 January 2014) was used to evaluate the model timing over an entire diurnal cycle, since usage of the shortwave code is dependent on time of day. Initial conditions for the WRF simulations were derived from forecast output generated by the NOAA Global Forecast System (GFS) model for 9-10 January 2014.

WRF RRTMG_CPU/RRTMGPU Performance Examples on NCAR/Caldera (Tesla K20X GPU)					
(1 Core) WRF/CPU + RRTMG			**(8 Cores) WRF/CPU + RRTMG**		
Model	Elapsed Time (sec)	Time Fraction vs. WRF	Model	Elapsed Time (sec)	Time Fraction vs. WRF
LW	866.4	0.23	LW	125.1	0.21
SW	644.6	0.17	SW	94.7	0.16
LW+SW	1511.0	**0.40**	LW+SW	219.8	**0.37**
WRF	3830.0	1.00	WRF	602.5	1.00
(1 Core) WRF/CPU + RRTMGPU			**(8 Cores) WRF/CPU + RRTMGPU**		
Model	Elapsed Time (sec)	Time Fraction vs. WRF	Model	Elapsed Time (sec)	Time Fraction vs. WRF
LW	65.6	0.03	LW	13.7	0.04
SW	47.6	0.02	SW	9.0	0.03
LW+SW	113.3	**0.05**	LW+SW	22.8	**0.07**
WRF	2337.4	1.00	WRF	335.0	1.00
Model	**CPU/GPU Time Ratio**	**GPU/CPU Time Ratio**	Model	**CPU/GPU Time Ratio**	**GPU/CPU Time Ratio**
LW	13.2	0.08	LW	9.1	0.11
SW	13.5	0.07	SW	10.6	0.10
LW+SW	**13.3**	**0.08**	LW+SW	**9.7**	**0.10**
WRF	1.6	0.61	WRF	1.8	0.56

Table 2. Elapsed time for RRTMG_LW, RRTMG_SW, the LW and SW total, and the WRF total for all codes running on the CPU (top rows), and elapsed time for WRF running on the CPU and for RRTMGPU_LW, RRTMGPU_SW, and the total LW and SW all running on the K20X GPU (center rows) using a single CPU (left columns) and using 8 CPU cores (right columns). Also shown are the fractions of time for the radiation components relative to the total WRF elapsed time (total radiation time in red), and the ratios of elapsed time for runs using only the CPU to those running the GPU radiation.

Timing results for the WRF simulations using the CPU version of the radiation (RRTMG) and the GPU version of the radiation (RRTMGPU) using the original Tesla M-2080Q GPU on 'caldera' are summarized in Table 1. A total of four, one-day WRF simulations are represented in Table 1. In two of these simulations, WRF was run in 'serial' mode on a single CPU processor with one using RRTMG (on the CPU) and the other utilizing the GPU to run RRTMGPU. The elapsed times for these runs are shown in the left columns of Table 1, with the pure CPU run in the top rows and the run including the GPU radiation code in the middle rows. Elapsed times are listed separately for the LW code, the SW code, the LW and SW total and the total WRF simulation time. The fraction of time for each of these components relative to the total WRF simulation time is also shown. The fraction of the total model time spent on the radiation (shown in red), which was 48% in the pure CPU simulation, dropped to just 7% when the radiation was running on the GPU. The total WRF elapsed time to simulate a model day with the radiation running on the GPU was 61% of the total elapsed time with all code running on the CPU. Of course, comparison of the GPU result to a single CPU is not a very representative test, since many WRF users do not run the model on a single CPU. Many utilize the distributed memory parallel processing ('dmpar') WRF configuration, so two additional simulations were performed in this configuration using eight CPU cores and each radiation model. The timing results of this pair of tests are shown in the right columns of Table 1. In this configuration, running the radiation on the GPU dropped to fraction of model time spent on the radiation calculation from 40% to 29% and the total elapsed time for WRF was reduced by about 20%.

Timing results for the WRF simulations using the CPU version of the radiation (RRTMG) and the GPU version of the radiation (RRTMGPU) using the newer Tesla K20X GPU on 'caldera' are summarized in Table 2. Once again, a total of four, one-day WRF simulations were completed. In two of these simulations, WRF was run in distributed memory ('dmpar') mode on a single CPU processor with one run using RRTMG (on the CPU) and the other utilizing the GPU to run RRTMGPU. The elapsed times for these runs are shown in the left columns of Table 2, with the pure CPU run in the top rows and the run including the GPU radiation code in the middle rows. The fraction of the total model time spent on the radiation (shown in red), which was 40% in the pure CPU simulation, dropped to just 5% when the radiation was running on the GPU. The total WRF elapsed time to simulate a model day with the radiation running on the GPU was 61% of the total elapsed time with all code running on the CPU. Two additional simulations were performed in the 'dmpar' configuration using eight CPU cores and each radiation model. The timing results of this pair of tests are shown in the right columns of Table 2. In this configuration, running the radiation on the GPU dropped to fraction of model time spent on the radiation calculation from 37% to 7% and the total elapsed time for

WRF was reduced almost by half. It should be noted that timing results are highly dependent on model configuration, compiler settings, as well as the GPU and CPU hardware used, so these figures should only be used as an indication of the performance improvement that can be attained in a few configurations. Establishing the optimal configuration for improving the radiation performance in WRF when utilizing the GPU will require some customization depending on the context in which it is used.

An additional consideration that somewhat affects code performance is the block size (that is, the number of grid points) that is sent to the GPU for simultaneous processing. The optimal selection of this value is largely controlled by the memory limitations of the hardware. RRTMGPU can either check the memory available in order to set this value accordingly, or it can be set manually. In all of the WRF simulations, which consisted of a single grid of 33,750 grid points, a block size of 4096 was used. The dependence of the model efficiency on block size was also tested and a slight reduction in elapsed time of 1-2% was seen when the block size was increased from 1024 to 2048 to 4096 in this hardware configuration, but increasing the block size further to 8192 increased the elapsed time of RRTMGPU by about a percent over the time using 4096, which illustrates the diminishing returns of overburdening the available memory resources.

The objective of running the radiation on the GPU was not only to improve its performance, but also to have negligible or no impact on the calculated fluxes and heating rates. To demonstrate effectively that flux differences between the CPU and GPU radiation codes are negligible, it was first necessary to ensure that each set of codes were functionally equivalent. Several layers of code differences had to be accounted for during the WRF implementation. As mentioned previously, physics changes added by NCAR to RRTMG in WRF were transferred into RRTMGPU to remove this difference. In addition, one remaining longwave bug fix that had been applied to RRTMG_LW_v4.85 (and RRTMGPU) by AER, but had not yet migrated into the codes in WRF had to be considered. Although the effect of this bug fix was expected to be negligible, its impact was tested first to account for its effect on radiative fluxes over the WRF CONUS test grid. A pair of WRF simulations was performed, entirely on the CPU, one using WRF_v361 as distributed, and another with the bug fix added to the longwave code. Outgoing longwave radiation and downward surface longwave flux as calculated by RRTMG in WRF_v361 at 18 UTC on 9 Jan 2014 are shown in the left panels in Figure 6. The right panels in Figure 6 show the differences in these parameters between the version of RRTMG in WRF_v361 and the version with the bug fix (RRTMG_LW_v4.85). As expected the flux differences are 0.01 Wm^{-2} or less over the entire CONUS test grid, and the output from the simulation that included the longwave bug fix can be used to compare to fluxes generated by RRTMGPU.

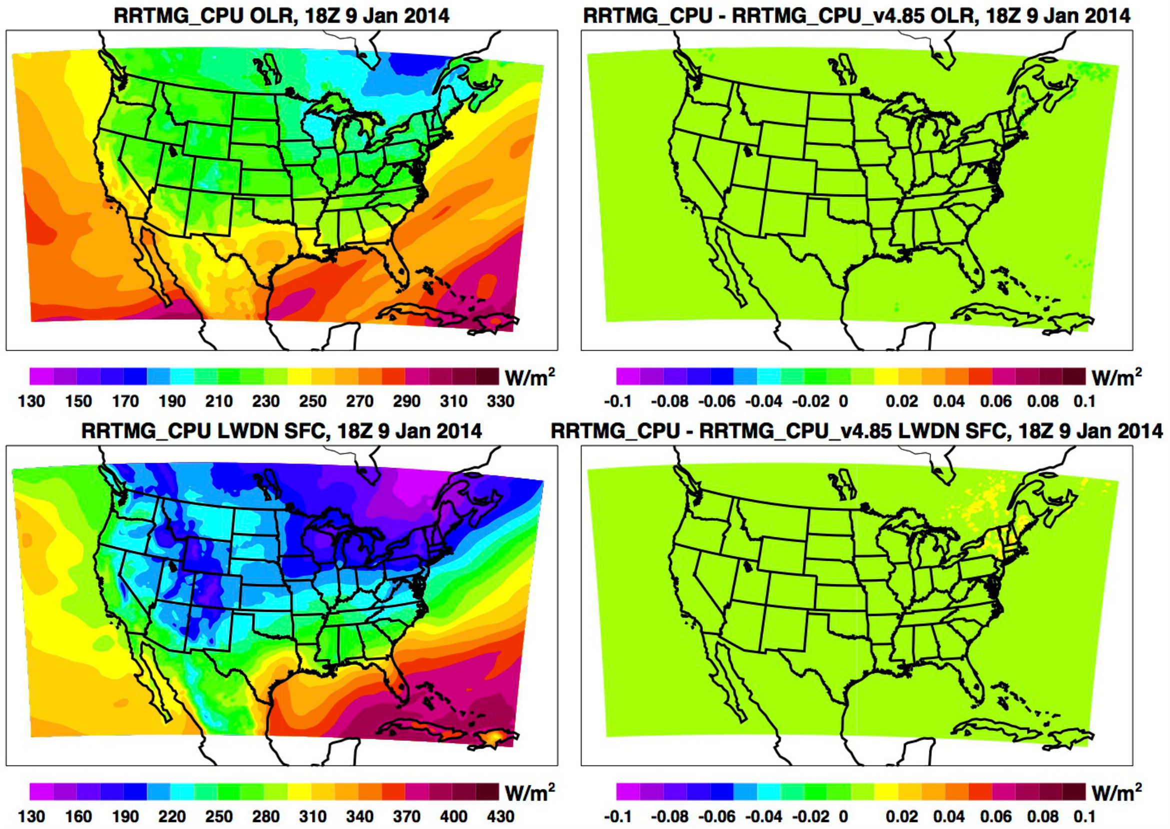

Figure 6. WRF generated longwave outgoing longwave radiation (OLR) at the top of the atmosphere at 18 UTC on 9 January 2014 calculated with RRTMG_LW running on the CPU (top left) and the OLR difference between the version of RRTMG_LW in WRF_v361 and another containing a minor bug fix that is not present in WRF_v361 (top right). WRF generated longwave downward surface flux (SFC) at 18 UTC on 9 January 2014 calculated with RRTMG_LW running on the CPU (bottom left) and the SFC difference between the version of RRTMG_LW in WRF_v361 and another containing a minor bug fix that is not present in WRF_v361 (bottom right).

An additional one-day WRF simulation covering 18 UTC 9 Jan to 10 Jan 2014 using RRTMGPU for the radiative calculation was next completed to demonstrate the impact on radiative fluxes of running on the GPU. It should be noted that this time is the initial radiation calculation at the beginning of each model run before any code differences could impact the atmospheric state. Outgoing longwave radiation and downward surface longwave flux as calculated by WRF_v361 with RRTMG_LW_v4.85 at 18 UTC on 9 Jan 2014 are shown in the left panels in Figure 7. The right panels in Figure 7 show the differences in these parameters between the CPU calculations (in the left panels) and the fluxes generated by RRTMGPU. In the OLR, very small negative differences generally less than 0.05 Wm^{-2} are seen over land, while over ocean differences up to 0.1 Wm^{-2} are seen. At the surface, very small positive differences of 0.03 Wm^{-2} or less are seen over both land and ocean areas.

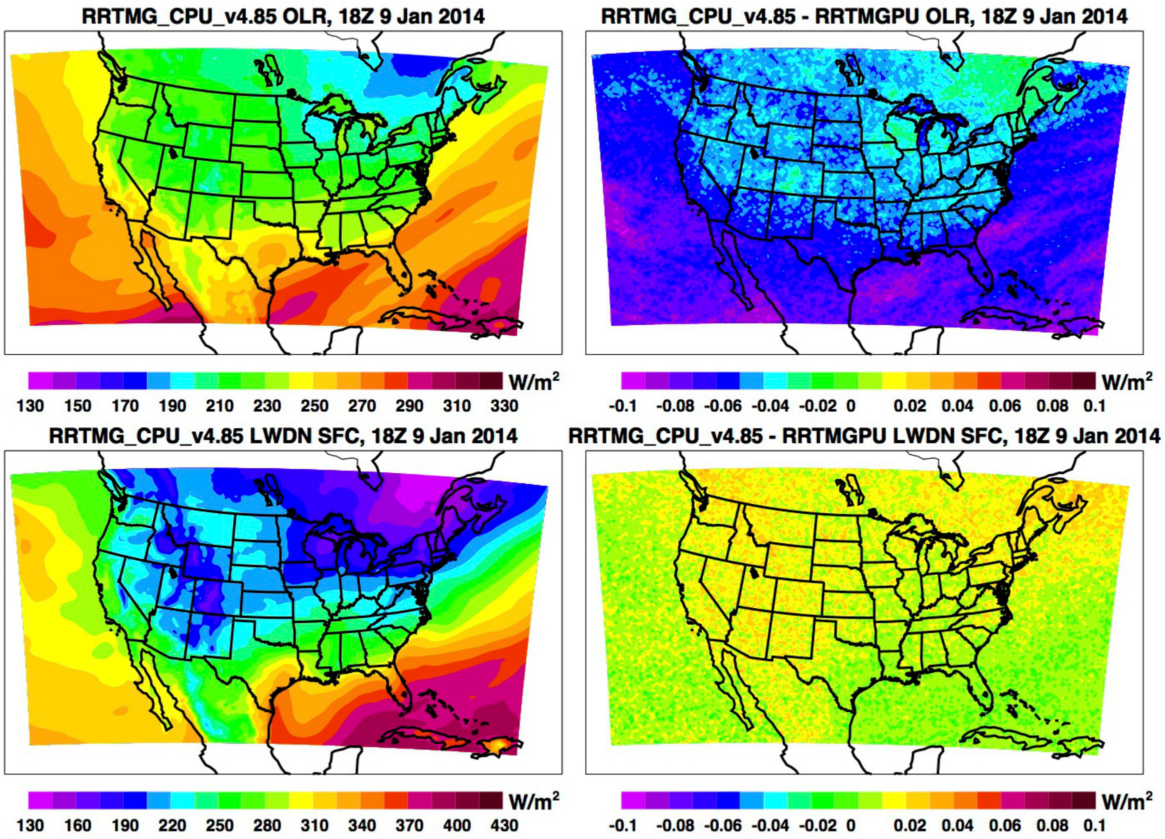

Figure 7. WRF generated longwave outgoing longwave radiation (OLR) at the top of the atmosphere at 18 UTC on 9 January 2014 calculated with RRTMG_LW_v4.85 running on the CPU (top left) and the OLR difference between the CPU calculation and fluxes generated by WRF running RRTMGPU (top right). WRF generated longwave downward surface flux (SFC) at 18 UTC on 9 January 2014 calculated with RRTMG_LW_v4.85 running on the CPU (bottom left) and the SFC difference between the CPU calculation and fluxes generated by WRF running RRTMGPU (bottom right).

The corresponding shortwave fluxes show an even smaller impact from running on the GPU. Upward shortwave top of the atmosphere flux and downward shortwave surface flux as calculated by WRF_v361 with RRTMG_SW at 18 UTC on 9 Jan 2014 are shown in the left panels in Figure 8. The right panels in Figure 8 show the differences in these parameters between the CPU calculations (in the left panels) and the fluxes generated by RRTMGPU. At both the top of the atmosphere and at the surface differences in shortwave fluxes remain less than 0.01 Wm^{-2} over the entire CONUS grid. Therefore, it can be concluded that running the radiation code on the GPU has inconsequential impacts on the calculated atmospheric radiative fluxes.

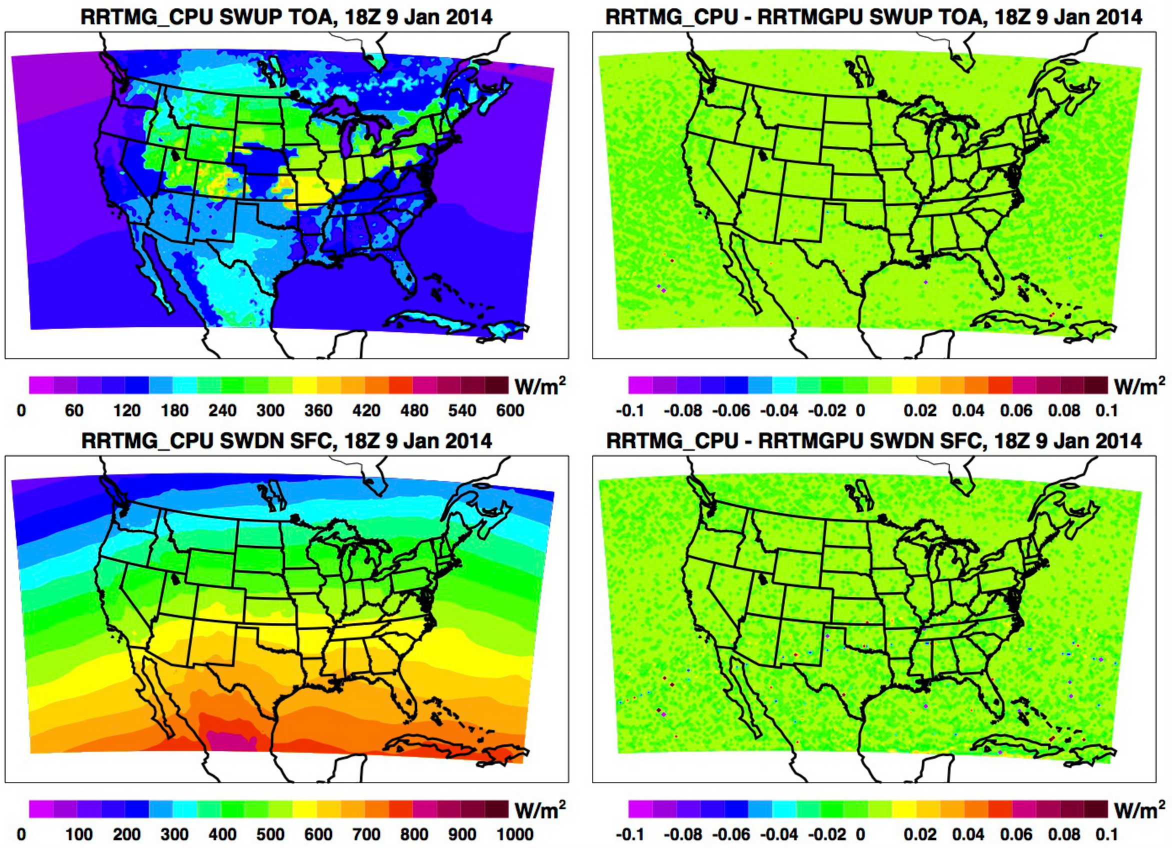

Figure 8. WRF generated shortwave upward flux at the top of the atmosphere (SWUP TOA) at 18 UTC on 9 January 2014 calculated with RRTMG_SW running on the CPU (top left) and the SWUP TOA difference between the CPU calculation and fluxes generated by WRF running RRTMGPU (top right). WRF generated shortwave downward surface flux (SWDN SFC) at 18 UTC on 9 January 2014 calculated with RRTMG_SW running on the CPU (bottom left) and the SWDN SFC difference between the CPU calculation and fluxes generated by WRF running RRTMGPU (bottom right).

WRF_v3.7beta

In early February 2015, a beta version of WRF_v3.7 was distributed by NCAR for testing to a small group of code developers and contributors including the PI prior to its expected public release in April 2015. The code was provided with sample initial boundary condition data files at 40-km resolution and at 4-km resolution over eastern North America for a test simulation of Hurricane Sandy over a 54-hour period from 12 UTC on 27 October 2012 through 18 UTC on 29 October 2012. For this version of WRF, the RRTMGPU code has been installed as a new radiative transfer option (‘ra_lw_physics’ = 24 and ‘ra_sw_physics’ = 24) that can be activated within the ‘namelist.input’ control file. New configure options have also been added to properly configure WRF for running the new radiation options on the GPU. The new accelerated radiation source files are named ‘module_ra_rrtmg_lwf.F’ and ‘module_ra_rrtmg_swf.F’ in WRF.

A simulation of Hurricane Sandy at 40-km resolution was completed as an additional test of the GPU radiation timing. Table 3 shows the elapsed time in seconds (following the format of Tables 1 and 2) for the longwave and shortwave codes separately, for both codes in combination, and for the full WRF calculation for one day of the 54-hour simulation. A matrix of four runs were completed over this period using WRF in its distributed memory ('dmpar') configuration. Two runs used a single CPU processor with either RRTMG running on the CPU or RRTMGPU running on the GPU, and the other pair of runs used eight CPU cores with each radiation code. This configuration of WRF uses a relatively modest grid of only 2500 grid points, which is not sufficiently large to utilize the GPU to full effect, but it is representative of a typical WRF simulation forecast grid. The timing results show the dramatic reduction in the fraction of time spent on the radiative transfer calculation in WRF when the radiation code runs on the GPU.

WRF RRTMG_CPU/RRTMGPU Performance Examples on NCAR/Caldera					
(1 Core) WRF/CPU + RRTMG			**(8 Cores) WRF/CPU + RRTMG**		
Model	Elapsed Time (sec)	Time Fraction vs. WRF	Model	Elapsed Time (sec)	Time Fraction vs. WRF
LW	99.87	0.17	LW	13.17	0.12
SW	99.44	0.17	SW	13.64	0.12
LW+SW	199.31	**0.34**	LW+SW	26.81	**0.24**
WRF	593.33	1.00	WRF	109.78	1.00
(1 Core) WRF/CPU + RRTMGPU			**(8 Cores) WRF/CPU + RRTMGPU**		
Model	Elapsed Time (sec)	Time Fraction vs. WRF	Model	Elapsed Time (sec)	Time Fraction vs. WRF
LW	9.35	0.02	LW	3.28	0.04
SW	7.72	0.02	SW	1.91	0.02
LW+SW	17.07	**0.04**	LW+SW	5.19	**0.06**
WRF	429.33	1.00	WRF	90.67	1.00
Model	**CPU/GPU Time Ratio**	**GPU/CPU Time Ratio**	Model	**CPU/GPU Time Ratio**	**GPU/CPU Time Ratio**
LW	10.7	0.09	LW	4.0	0.25
SW	12.9	0.08	SW	7.1	0.14
LW+SW	**11.7**	**0.09**	LW+SW	**5.2**	**0.19**
WRF	1.4	0.72	WRF	1.2	0.83

Table 3. Elapsed time for RRTMG_LW, RRTMG_SW, the LW and SW total, and the WRF total for all codes running on the CPU (top rows), and elapsed time for WRF_v3.7 running on the CPU and for RRTMGPU_LW, RRTMGPU_SW, and the total LW and SW all running on the K20X GPU (center rows) using a single CPU (left columns) and using 8 CPU cores (right columns) for a single forecast day from a 54-hour simulation at 40-km resolution. Also shown are the fractions of time for the radiation components relative to the total WRF elapsed time (total radiation time in red), and the ratios of elapsed time for runs using only the CPU to those running the GPU radiation.

Due to the much higher resolution of the 4-km grid, which has 250,000 grid points, and the much greater computational expense of running with this grid, only two 12-hour WRF simulations were completed each using eight CPU processors with either RRTMG or RRTMGPU. Since the larger grid size in this case allows for larger blocks of model grid points to be sent to the GPU, which greatly enhances its impact on timing, these experiments showed an even better boost in performance than the 40-km case, with RRTMGPU running at roughly 15-20 times faster than RRTMG on the 4-km grid. The primary conclusion of these tests is that running the radiation code on the GPU provides reductions in model elapsed time in all configurations, though the best improvement in performance will be seen for the largest, highest resolution grids.

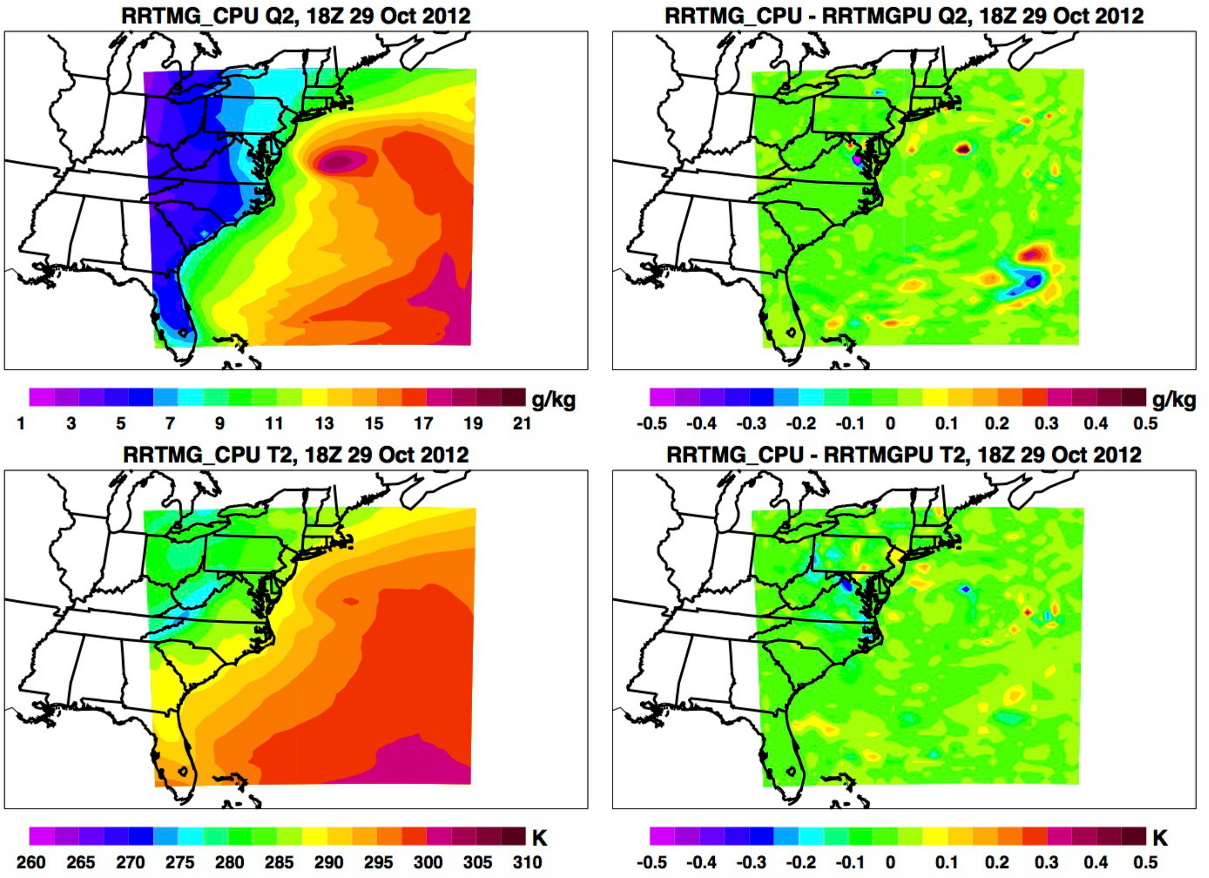

Figure 9. WRF generated 2-m specific humidity (Q2) at 18 UTC on 29 October 2012 calculated with RRTMG running on the CPU (top left) and the Q2 difference between the CPU calculation and values generated by WRF running RRTMGPU (top right). WRF generated 2-m temperature (T2) at 18 UTC on 29 October 2012 calculated with RRTMG running on the CPU (bottom left) and the T2 difference between the CPU calculation and values generated by WRF running RRTMGPU (bottom right). The weather feature off the mid-Atlantic coast is Hurricane Sandy prior to landfall.

As a final demonstration of the negligible impact on model output of running the radiation code on the GPU, several dynamical output parameters from the WRF_v3.7beta 40-km test simulations were examined. Figure 9 shows the 2-m specific humidity (Q2) output at the end of the 54-hour simulation at 18 UTC 29 October 2012 as generated by WRF with RRTMG running on the CPU in the upper left panel. The Q2 difference between this result and the Q2 generated by WRF with the radiation running on the GPU is shown in the upper right panel. The lower panels in Figure 9 show the 2-m temperature (T2) generated by WRF on the CPU and the temperature difference between the CPU and GPU simulations. It should be noted that unlike the flux differences shown in earlier figures, which were for the initial time step during the simulations, the differences in Figure 9 show the impact on Q2 and T2 at the end of a 54-hour forecast when the small initial flux perturbations have had a longer time to impact the dynamical

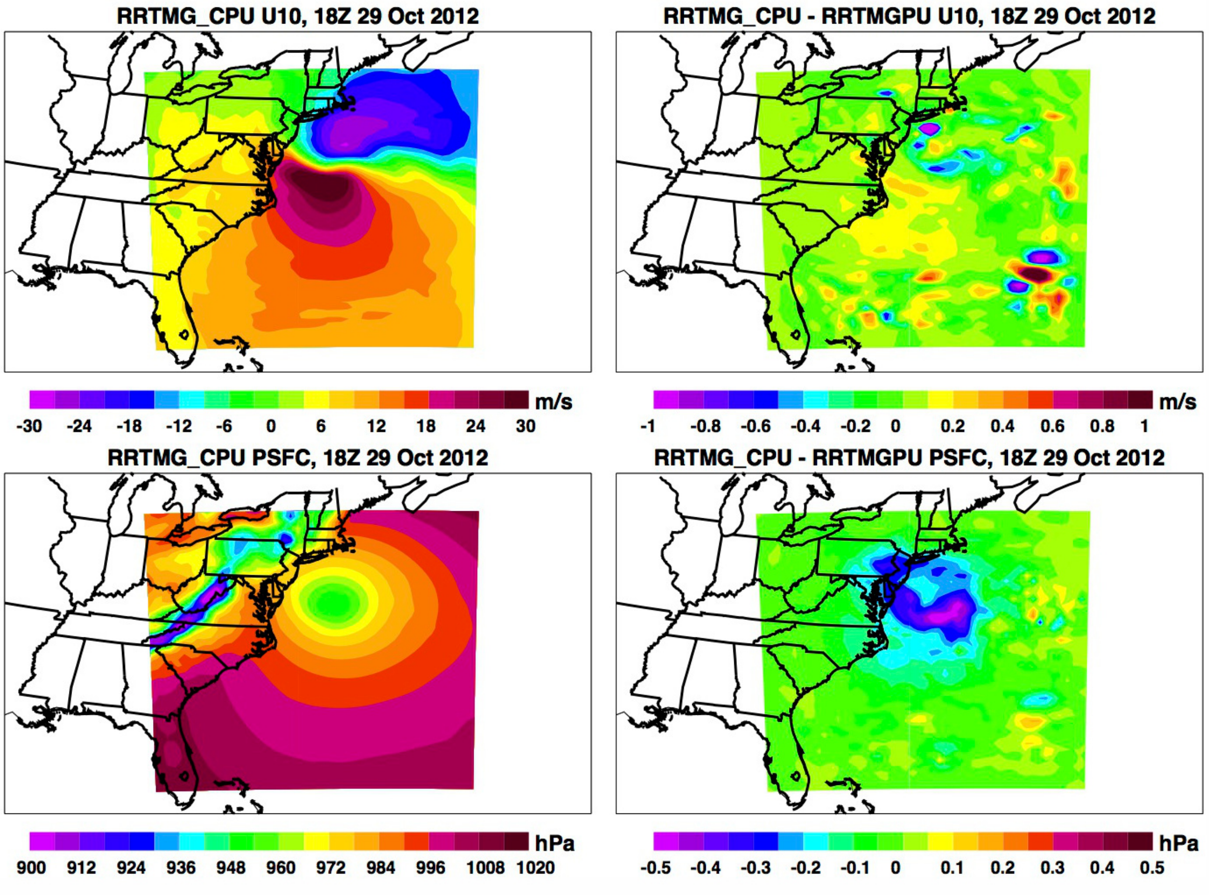

Figure 10. WRF generated 10-m east-west wind component (U10) at 18 UTC on 29 October 2012 calculated with RRTMG running on the CPU (top left) and the U10 difference between the CPU calculation and values generated by WRF running RRTMGPU (top right). WRF generated surface pressure (PSFC) at 18 UTC on 29 October 2012 calculated with RRTMG running on the CPU (bottom left) and the PSFC difference between the CPU calculation and values generated by WRF running RRTMGPU (bottom right). The weather feature off the mid-Atlantic coast is Hurricane Sandy prior to landfall.

fields. Furthermore, this simulation includes an extreme weather event to further highlight any potential field differences in such circumstances. The weather feature off the mid-Atlantic coast is Hurricane Sandy prior to its landfall in coastal New Jersey. Despite this, the differences seen in both the specific humidity and temperature are largely very small and generally less than one percent of the original field values with only a few instances of slightly larger differences.

Very minor differences are also seen in fields that are presumably more sensitive to small perturbations than temperature and moisture. Figure 10 shows the 10-m east-west wind component (U10) output at the end of the 54-hour simulation at 18 UTC 29 October 2012 as generated by WRF with RRTMG running on the CPU in the upper left panel. The U10 difference between this result and the U10 generated by WRF with the radiation running on the GPU is shown in the upper right panel. The lower panels in Figure 10 show the surface pressure (PSFC) generated by WRF on the CPU and the surface pressure difference between the CPU and GPU simulations. Figures 9 and 10 illustrate that the very small perturbations generated in the flux fields using the GPU radiation have a negligible effect even after a 54-hour forecast.

It is noted for completeness that collaborator John Michalakes has worked independently and with separate NOAA funding to further modify the new radiation options in WRF_v3.7beta so that in addition to being able to run on the GPU they are also able to run efficiently on Intel Many-Integrated-Core (MIC) CPU technology. This required modifying the code so that certain code processes such as memory allocation and array looping are performed in the most efficient way for the type of hardware in use. In this way, the new radiation options provide much better flexibility in enhancing performance both of the radiation codes and the WRF model overall.

5. Deliverables

Although not strictly a deliverable, since this work was proposed as a demonstration of utilizing the accelerated radiation in WRF, RRTMGPU has been provided to NCAR as of December 2014 for the purpose of making it available for operational use in WRF. John Michalakes is serving as the point of contact to David Gill and the WRF Developer's Committee at NCAR for this code contribution. The GPU radiation code will continue to undergo adjustment and testing within WRF to access its possible application to the next WRF release planned for Spring 2015. As of early March 2015, a beta version of WRF_v3.7, including the accelerated radiation codes as a new option, has been distributed to a small list of WRF developers (including the PI) and remains under review. As of this writing, NCAR has yet to announce formally which new features and options will be included in WRF_v3.7.

Other deliverables generated during the project include multiple poster and oral presentations during 2014 and 2015 directly related to this research:

- The PI attended the DOE Climate and Earth System Modeling Principal Investigator's Meeting in May 2014 and presented a poster with co-authors D. Berthiaume, E. Mlawer, and J. Michalakes titled *Enhancing Efficiency of the RRTMG Radiation Code with Graphics Processing Units in the Weather Research and Forecasting Model* (*Iacono et al.,* 2014a),
- The PI was a co-author on an oral presentation titled *Performance-Related Developments in WRF* and given by John Michalakes at the NCAR WRF User's Workshop in June 2014 (*Michalakes et al.,* 2014a),
- In July 2014, the PI attended the AMS Conference on Atmospheric Radiation and presented a poster titled *Enhancing Efficiency of the RRTMG Radiation Code with GPU and MIC Approaches for Numerical Weather Prediction Models* (*Iacono et al.,* 2014b),
- The PI was a co-author with D. Berthiaume on an oral presentation titled *Optimizing Weather Model Radiative Transfer Physics for the Many Integrated Core and GPGPU Architectures* given by John Michalakes at the NCAR Heterogeneous Multi-Core Workshop in September 2014 (*Michalakes et al.,* 2014b),
- Finally, in January 2015 the PI was a co-author on an oral presentation titled *Nest Generation of HPC and Forecast Model Application Readiness at NCEP* given by John Michalakes at the AMS First Symposium on High Performance Computing for Weather, Water, and Climate at the 95th AMS Annual Meeting (*Michalakes et al.,* 2015).

6. Future Direction: RRTMGP

RRTMGPU represents a transitional code that begins the process of improving the performance of the radiative transfer in parallel processing environments within global models. The next step is already in progress with the ongoing development of RRTMGP, which is being prepared (at AER with funding from the Office of Naval Research) as a redesigned and generalized version of RRTMG that will take optimum advantage of the various methods of parallelization available on modern supercomputers. RRTMGP will be unencumbered by the limitations of RRTMGPU, which requires specific GPU hardware and PGI compilers in order to be effective. Thus, application of RRTMGPU to WRF is only a first step in the direction of enhancing the radiation efficiency in WRF. The future availability of RRTMGP will make it possible to improve the performance of the radiative transfer in WRF and other dynamical models in a more generalized and comprehensive way.

7. References

Barker, H., J.N.S. Cole, J.-J. Morcrette, R. Pincus, P. Raisanen, Monte Carlo Independent Column Approximation (McICA): Up and running in North America and Europe, Talk presented at the 17th Atmospheric Radiation Measurement (ARM) Science Team Meeting, Monterey, CA, March 26-30, 2007.

Iacono, M.J., J.S. Delamere, E.J. Mlawer, M.W. Shephard, S.A. Clough, and W.D. Collins, Radiative forcing by long-lived greenhouse gases: Calculations with the AER radiative transfer models, *J. Geophys. Res.*, 113, D13103, doi:10.1029/2008JD009944, 2008.

Iacono, M.J., D. Berthiaume, E. Mlawer, and J. Michalakes, Enhancing efficiency of the RRTMG radiation code with graphics processing units in the Weather Research and Forecasting model, Poster presentation at the DOE Climate and Earth System Modeling Principal Investigator's Meeting, Potomac, Maryland, May 12-14, 2014a.

Iacono, M.J., D. Berthiaume, and J. Michalakes, Enhancing efficiency of the RRTMG radiation code with GPU and MIC approaches for numerical weather prediction models, Poster presentation at the 14th American Meteorological Society Conference on Atmospheric Radiation, Boston, Massachusetts, July 7-11, 2014b.

Michalakes, J., and M. Vachharajani. GPU Acceleration of Numerical Weather Prediction, *Parallel Processing Letters*, 18 No. 4, World Scientific, 531-548, 2008.

Michalakes, J., M.J. Iacono, D. Berthiaume, and I.M. Gokhale, Performance-related developments in WRF, Oral presentation at the 15th NCAR Weather Research and Forecasting (WRF) User's Workshop, Boulder, Colorado, June 23-27, 2014a.

Michalakes, J., M.J. Iacono, and D. Berthiaume, Optimizing weather model radiative transfer physics for the Many Integrated Core and GPGPU architectures, Oral presentation at the NCAR Heterogeneous Multi-Core Workshop, Boulder, Colorado, September 17-18, 2014b.

Michalakes, J., and M.J. Iacono, Next generation of HPC and forecast model application readiness at NCEP, Oral presentation at the First Symposium on High Performance Computing for Weather, Water, and Climate at the 95th American Meteorological Society Annual Meeting, Phoenix, Arizona, January 4-8, 2015.

Mlawer, E.J., S.J. Taubman, P.D. Brown, M.J. Iacono, and S.A. Clough, Radiative transfer for inhomogeneous atmospheres: RRTM, a validated correlated-k model for the longwave, *J. Geophys. Res.*, **102**, 16,663-16,682, 1997.

Pincus, R., H. W. Barker, J.-J. Morcrette, A fast, flexible, approximate technique for computing radiative transfer in inhomogeneous cloud fields, *J. Geophys. Res.,* **108**, D13, 2003.

Skamarock, W. C., J. B. Klemp, J. Dudhia, D. O. Gill, D. M. Barker, M. G. Duda, X.-Y. Huang, W. Wang, and J. G. Powers: A description of the Advanced Research WRF Version 3. NCAR Tech Notes-475+STR, 125 pp., 2008.