156 ENHANCING EFFICIENCY OF THE RRTMG RADIATION CODE WITH GPU AND MIC APPROACHES FOR NUMERICAL WEATHER PREDICTION MODELS

Michael J. Iacono * and David Berthiaume
Atmospheric and Environmental Research, Lexington, Massachusetts

John Michalakes
NOAA/National Centers for Environmental Prediction, College Park, Maryland

1. INTRODUCTION

Radiative transfer (RT) calculations are among the most computationally expensive components of global and regional weather and climate models, and radiation codes are therefore ideal candidates for applying techniques to improve the overall efficiency of such models. In many general circulation models (GCMs), a physically based radiation calculation can require as much as 30-50 percent of the total GCM execution time. This work examines two methods of enhancing the efficiency of the widely used RRTMG radiation codes by employing computing hardware designed for code acceleration such as graphics processing units (GPUs) and the Intel Xeon Many Integrated Core (MIC) architecture rather than traditional multi-core central processing units (CPUs).

In some dynamical models, the computational expense of RT in GCMs has been improved by reducing the spatial or temporal resolution of the radiation calculations, but this approach may degrade the calculated radiative fluxes and heating rates and possibly impair the simulation. One motivation for using high performance computing techniques is to allow RT calculations to be performed at appropriate spatial and temporal resolutions as well as making additional computational time available to improve other dynamical model components.

2. RADIATION MODEL MODIFICATIONS

The RRTMG longwave and shortwave broadband, correlated k-distribution radiation models (Iacono et al., 2008, Mlawer et al., 1997) used in this project are now in wide use by the weather and climate modeling community and running on a wide range of computer hardware. The improved accuracy of the RRTMG longwave (LW) and shortwave (SW) models, which were developed for application to GCMs for the Department of Energy (DOE) Atmospheric System Research program, is traceable to measurements through their comparison to higher resolution, data-validated line-by-line models (Clough et al., 2005) RRTMG also utilizes the Monte-Carlo Independent Column Approximation, McICA (Barker et al., 2002; Pincus et al., 2003), which is a statistical method for representing sub-grid scale cloud variability including cloud overlap. More information about these radiation codes is available at the AER RT web site (rtweb.aer.com).

The RRTMG radiative transfer options were originally implemented into the Weather Research and Forecasting (WRF) model as part of the WRF_v3.1 release in April 2009. For this work, these codes have been modified to demonstrate improvement in computational performance of the radiation physics that can be attained using GPUs and the Intel MIC technology. New versions of the longwave and shortwave RT codes (known as RRTMGPU) have been developed and tested in WRF on a GPU-enabled computer system at NCAR (Caldera) and in off-line mode on MIC-enabled workstations. The GPU system uses Intel Xeon (Sandybridge) processors, it has two NVIDIA Tesla M2070-Q GPUs per node, and it supports the PGI compilers that are currently necessary to compile the GPU-accelerated RT codes using CUDA Fortran. In order to fully utilize the potential of the GPU, the codes were transformed from operating over a single atmospheric column per call to running in parallel on multiple threads on the GPU over blocks of horizontal grid cells, vertical layers, and the RRTMG pseudo-spectral g-point dimension. Although the radiation models have been restructured for this application, the high accuracy of the radiative transfer is shown to be unaffected. It should be noted that the LW model tested here contains a minor improvement to the physics relative to the version currently available in WRF, though the impact of this change has a negligible effect on the resulting radiative fluxes, which will be demonstrated.

* *Corresponding author address*: Michael J. Iacono, Atmospheric and Environmental Research, 131 Hartwell Avenue, Lexington, Massachusetts 02421-3105; E-mail: miacono@aer.com.

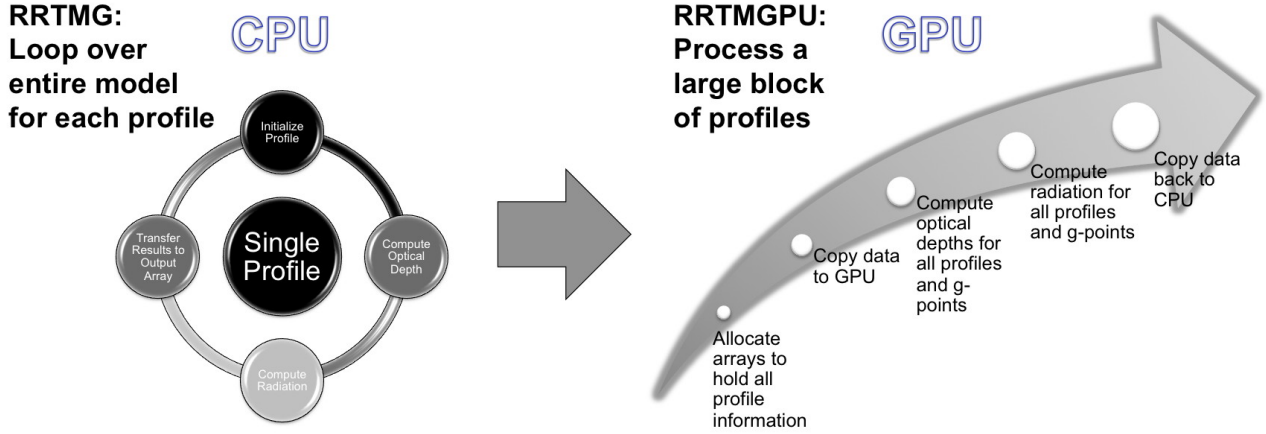

Figure 1. Schematic diagram of the radiative transfer treatment applied in the original version of RRTMG for use on CPUs (left) and the approach utilized for developing RRTMGPU, the GPU version of RRTMG (right).

A significant number of detailed code modifications were required to optimize the performance of RRTMGPU. In order for every atmospheric profile to run in parallel, arrays were padded to be multiples of 32, the size of a warp on a GPU, and reordered so that the fastest changing dimension coincides with the thread layout to enable efficient memory coalescing. The codes were restructured so that g-points can run in parallel, ensuring that even with a relatively low number of profiles, the GPU is always busy and therefore running efficiently. Several look-up tables, which had been added to RRTMG to improve its efficiency, were removed from RRTMGPU and calculations were restored within the main loops to avoid scattered memory access and to enable faster execution on the GPU. Profile partitioning was implemented using the MPI (Message Passing Interface) API (Advanced Message Passing Infrastructure) and multiple streams for running RRTMG on multiple GPUs in parallel. The main loop was restructured so that instead of running on a single profile at a time, the various subroutines for all of the profiles were designed to run in parallel.

In a parallel effort, the RRTMG codes have also been modified for testing on the Intel MIC architecture for application to the National Centers for Environmental Prediction (NCEP) Non-hydrostatic Multi-scale Model (NMM-B) and Global Forecast System (GFS) weather prediction models, continuing earlier work to improve the performance of WRF model physics (Michalakes, 2013). The RRTMG longwave radiation has been adopted without change in NMM-B, relying on the Intel compiler to identify and vectorize inner loops over the vertical dimension. The shortwave code has expensive vertical loops with recurrences that will not vectorize, so restructuring similar to the GPU implementation exposed inner loop vectorization horizontally over adjacent columns. Additional secondary changes were made to reorder the k-distribution look-up table indices to improve efficiency on both the MIC architecture and the host Xeon processor. Additionally, to allow direct performance comparison among devices, the RRTMGPU shortwave code was also optimized and ported back to run on the Sandybridge and MIC processors.

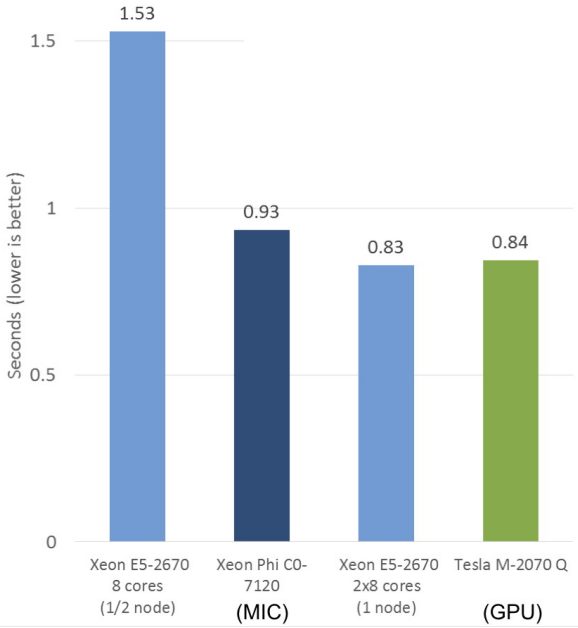

Figure 2. Total elapsed time in seconds for execution of 18819 profile calculations with RRTMG_SW on a Xeon CPU with 8 cores and 16 cores (light blue), on an MIC-enabled platform (dark blue) and on the GPU (green).

3. RADIATION PERFORMANCE ON MIC

A comparison of the performance of the RRTMG_SW code on an MIC-enabled workstation was run in column mode off-line from a GCM using a workload of 18819 columns and 60 atmospheric layers. Figure 2 shows the total elapsed time of the code (in seconds) for these runs on the Xeon Sandybridge CPU using both 8 cores and 16 cores (light blue), on the MIC Xeon Phi architecture (dark blue), and on the NVIDIA Tesla GPU (green). In this context, the elapsed times on the MIC and GPU are comparable to the CPU running with 16 cores.

4. RADIATION PERFORMANCE ON GPU

Assessment of the improved performance of the RT code on the GPU relative to the CPU was also tested on a GPU-enabled system at NCAR (Caldera) in off-line column mode. This CPU is a 2.6 GHz Intel Xeon E5-2670 and the GPU is the NVIDIA Tesla M2070-Q with a compute capability of 2.0. Compilation of the RRTMGPU codes requires v13.9 of the PGI (Portland Group) Fortran compiler with CUDA Fortran v5.0 using openACC, which have demonstrated good performance (Michalakes and Vachharajani, 2008). Tests were performed to evaluate performance on each type of processor relative to the number of atmospheric profiles (varying from 1250 to 40000) and to illustrate the GPU timing relative to the performance provided by varying numbers of CPU cores (from 1 to 28).

The top panel of Figure 3 shows the elapsed time (in seconds) required to run RRTMG_SW on the Xeon CPU using 1, 8 and 16 cores and the corresponding elapsed time to run RRTMGPU_SW on the GPU as a function of the number of profiles. The simulated profiles used consisted of 72 layers, varied in their specification of the atmosphere, and included clouds and aerosols to enhance the realism of the tests. For any number of profiles, the GPU processing is roughly a factor of two faster than the CPU running with 16 cores and more than ten times faster than the CPU using a single processor core.

The elapsed time required to run the SW codes over 20000 profiles with 72 layers on the CPU (black) as a function of the number of CPU cores is shown in the bottom panel of Figure 3. The elapsed time for running this code with the same number of profiles on the GPU (green) is shown for comparison. It should be noted that the CPU utilized hyper-threading on Caldera, which allows each core to access two threads simultaneously during processing and each core will appear to function essentially as two cores. Thus the maximum number of CPU threads available on this two-node (8 core per node) system was 32. This illustrates the improvement in performance realized for this code on this system as the number of processors is increased. Reduced performance was noted when the maximum number of threads (32) was requested.

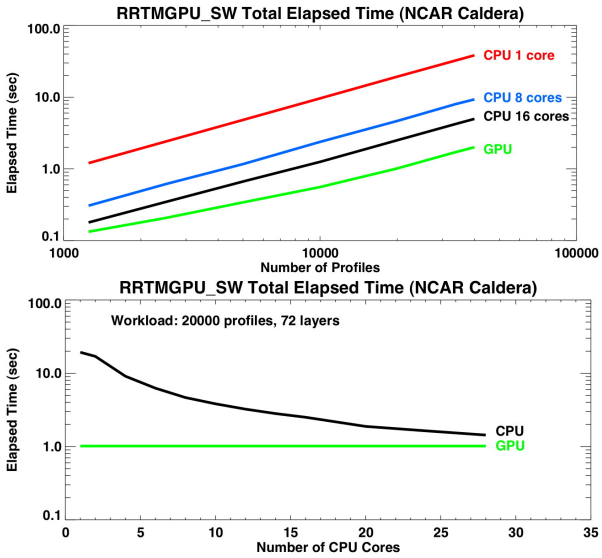

Figure 3. Elapsed time (in seconds) of RRTMG_SW relative to the number of atmospheric profiles for 1, 8 and 16 CPU cores and of RRTMGPU_SW on the GPU (top) and the elapsed time for the models on the GPU and CPU relative to the number of CPU cores (bottom).

5. RRTMGPU APPLICATION TO WRF

The RRTMGPU_LW and SW codes have been implemented into WRF_v3.5.1 to examine their impact on radiation efficiency and overall model efficiency in the context of a typical regional weather forecast. A single forecast domain over the continental United States and the adjacent ocean areas with a total of 33750 grid points and a corresponding spatial resolution of 30 km was used for the tests. The default vertical resolution of 29 layers was used. To establish model performance on the CPU, WRF was run in its default configuration along with the RRTMG longwave and shortwave radiation options. Timing was tested on the GPU by replacing the RRTMG modules with the RRTMGPU codes, which were repackaged into two source code modules for implementation into WRF. The default dynamical time step of three minutes was used along with a

30-minute radiation time step. Separate elapsed times for the longwave and shortwave calculations were measured over an entire forecast day initialized at 18 GMT on 9 January 2014. The timing test was conducted over a full forecast day to ensure a realistic representation of the timing of the SW code, which only runs during daylight.

A difference in the physics between the RRTMG and RRTMGPU codes must be noted that does not impact the efficiency, but it does impact the verification of the results as shown later in this section. The AER codes installed in WRF_v3.5.1 are equivalent to RRTMG_LW_v4.71 and to RRTMG_SW_v3.7 as described at the AER radiative transfer web site. The RRTMGPU codes tested for this work are equivalent to RRTMG_LW_v4.85 and RRTMG_SW_v3.8. While the shortwave codes are identical physically and produce the same results, the newer version of the longwave code includes a fix to the optical depth calculations (added in v4.82) that impacts the flux results to a very small degree. In atmospheric profiles with low temperature and low water vapor, downward fluxes are typically increased by this change by about 0.1 Wm^{-2} at the surface and by up to 0.5 Wm^{-2} in some layers. There are no changes in fluxes or heating rates in profiles with moderate to high water vapor.

Another difference between the CPU and GPU timings was necessary at this stage involving the shortwave codes. On the CPU, RRTMG_SW runs each grid point independently, and therefore any grid points not in daylight are easily skipped, which saves processing time. On the GPU, large blocks of grid points are processed simultaneously. Since this does not easily accommodate identification of individual daylight columns, the entire block is passed to the GPU whenever any of the grid points in the block is in daylight. Entire blocks that do not include any daylight grid points are not processed on the GPU. It is presumed that the benefit provided by the multiple processing is greater than the loss incurred by processing blocks that contain grid points that are not in daylight, though this will be investigated as a potential future enhancement.

WRF RRTMG_CPU/RRTMGPU Performance Examples on NCAR/Caldera					
WRF/CPU (1 Core; "serial")			**WRF/CPU (8 Cores; "dmpar")**		
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
WRF/CPU + RRTMGPU			**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		Model	**CPU/GPU** Time Ratio	
LW	12.9		LW	1.7	
SW	11.6		SW	1.6	
LW+SW	**12.3**		LW+SW	**1.7**	
WRF	1.7		WRF	1.2	

Table 1. Elapsed time (in seconds) and the fraction of time spent on each radiation component relative to the total WRF elapsed time for one-day simulations with WRF configured in 'serial' mode using a single CPU processor (at left) and with WRF configured in 'dmpar' mode using eight processors (at right). Timings are shown for tests with all code running on the CPU (top) and for tests in which the radiation codes ran on the GPU (middle). Timing ratios between the CPU and GPU runs are shown at bottom.

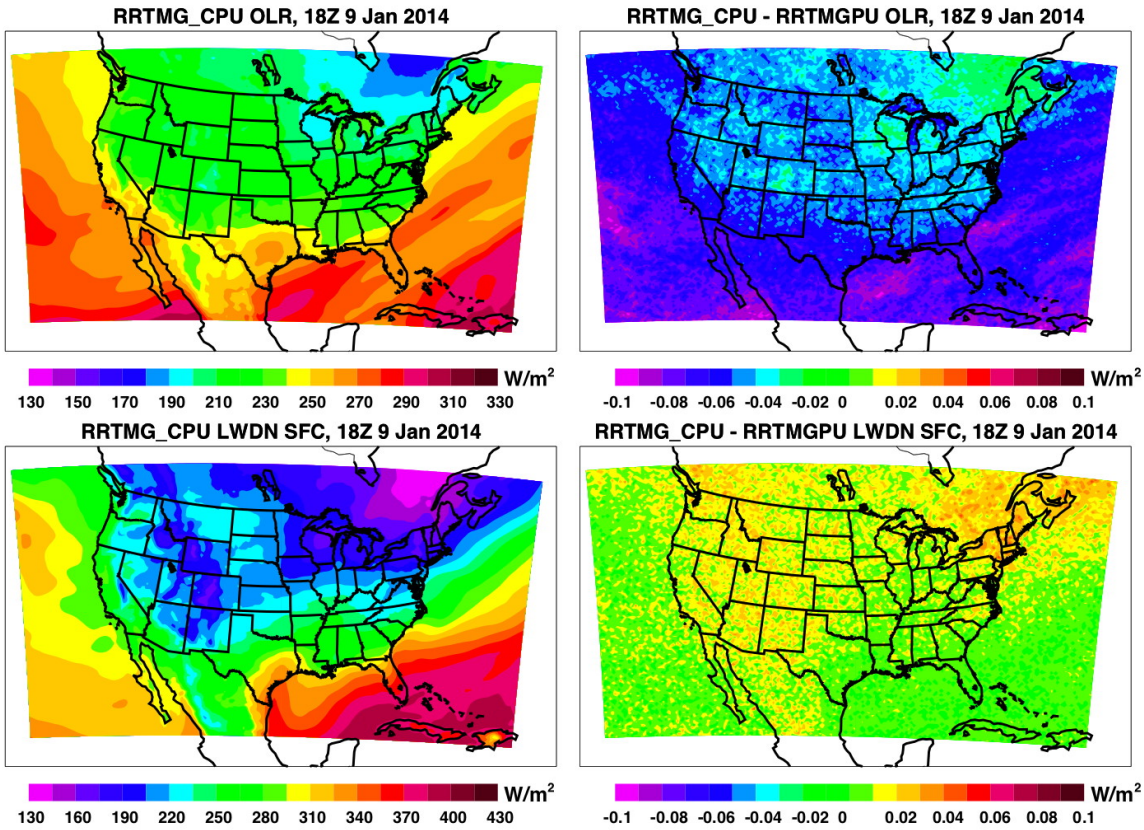

Figure 4. Outgoing longwave radiation (OLR, top left) and downward longwave surface flux (bottom left) as simulated by WRF at 18 GMT on 9 January 2014 using only CPU processing, and OLR (top right) and downward surface flux (bottom right) differences between simulations running only the CPU and running the radiation components on the GPU.

Radiation timing performance within WRF was tested with a set of four one-day WRF forecasts on the domain described earlier. For one pair of tests, WRF was configured to run in 'serial' mode on a single processor with the code running entirely on the CPU in one experiment and with RRTMGPU running on the GPU in the other experiment while the remainder of WRF used the CPU. For the other pair of tests, WRF was configured in 'dmpar' mode to use distributed memory parallel processing with eight cores while running the RT codes on the CPU in one simulation and on the GPU in the other.

The timing results for these four WRF simulations are summarized in Table 1. At left are the elapsed times (in seconds) for the single CPU tests for the entire code running on the CPU (at top) and with the RT running on the GPU (middle). Listed are the individual elapsed times for the longwave and shortwave components, the total LW and SW timing, and the total WRF timing. Also listed is the fractional time spent on each component relative to the total WRF time. While comparison of the GPU to a single CPU core is not an optimal comparison, running WRF in 'serial' mode is a valid model configuration, so timings based on this configuration are provided for reference. It is seen that the total radiation (LW+SW) timing is nearly half of the total WRF elapsed time when the code runs in full on a single core, while transferring the RT code to the GPU reduces the computational expense of the radiation to seven percent of the total WRF elapsed time. At right in Table 1 are comparable timing results for WRF configured to use parallel processing with eight CPU cores. In this context, the total RT cost is 40 percent of the total WRF elapsed time when only the CPUs are used, and this drops to 29 percent when the RT runs on the GPU. The figures at the bottom of Table 1 are the ratios of the fully CPU run to the test that ran the RT on the GPU. This shows that the radiation runs about twelve times faster on the GPU within WRF than on a single CPU and just under twice as fast on the GPU relative to eight CPU cores. It should

be emphasized that these are preliminary timing results that do not necessarily reflect the optimal improvement that can be attained on the GPU and are highly dependent on system configuration, compiler settings, GPU hardware, etc., but they are provided as an indication of the degree of timing improvement that is possible on the GPU.

Finally, an important objective in accelerating the performance of the RT codes is to accomplish the improvement while ensuring that the codes generate the same results within a small tolerance threshold. This is illustrated in Figure 4, which shows the impact on the outgoing longwave radiation (OLR) at the top of the atmosphere and the downward longwave flux at the surface. These parameters are shown for 18 GMT on 9 January 2014 in the left panels of Figure 4 for the simulation run entirely on the CPU. Flux differences for each quantity between the pure CPU run and the test in which the RT was run on the GPU are shown in the right panels of Figure 4. This result is not a pure verification of the impact of running the LW code on the GPU, since these differences also include the impact of the slight physics change added to the LW code running on the GPU. However, the magnitude of both the OLR and downward LW surface flux difference is generally less than 0.1 Wm^{-2}. Since this is the expected effect of the physics change alone, the impact of running the LW RT on the GPU are very likely of this magnitude or smaller.

Similar plots for the shortwave fluxes are shown in Figure 5, in which the left panels illustrate the upward shortwave flux at the top of the atmosphere and the downward shortwave flux at the surface simulated by WRF only on the CPU for 18 GMT on 9 January 2014. Flux differences for each quantity between the pure CPU run and the test in which the RT was run on the GPU are shown in the right panels of Figure 5. Here, since there is no change in the physics between the CPU and GPU versions of the shortwave RT code, the flux differences are within a small tolerance (i.e. close to zero) across the entire domain.

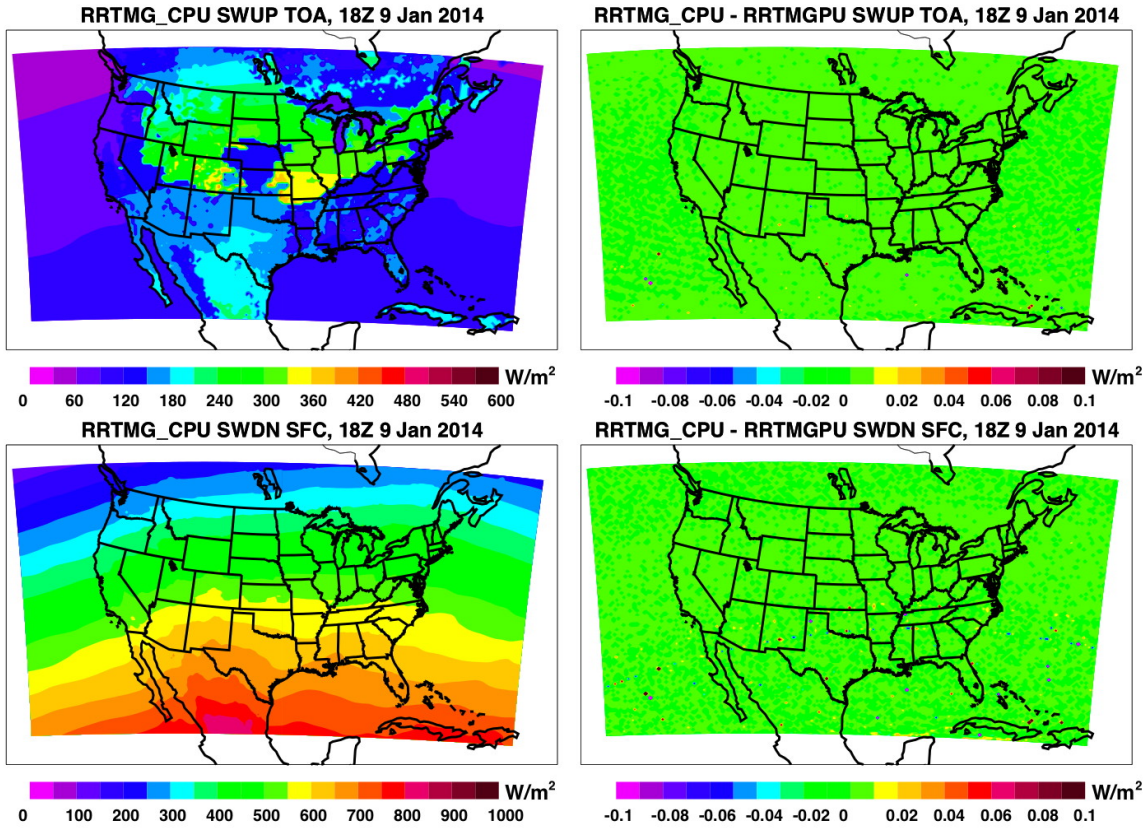

Figure 5. Upward shortwave radiation at the top of the atmosphere (TOA, top left) and downward shortwave surface flux (bottom left) as simulated by WRF at 18 GMT on 9 January 2014 using only CPU processing, and upward TOA (top right) and downward surface flux (bottom right) differences between simulations running only the CPU and running the radiation components on the GPU.

6. CONCLUSIONS

This work has demonstrated the impact on model efficiency of running modified versions of the RRTMG_LW and SW radiative transfer options in WRF on graphics processing units and in a stand-alone context on the Intel Xeon Phi MIC architecture. Multiple revisions were adopted to enhance the algorithmic efficiency of the codes for parallel processing. This includes restructuring loops to ensure parallelization over a larger set of dimensions, loop reordering (as necessary for GPU or MIC) and array padding to ensure efficient use of the hardware. Both approaches show improved performance over traditional multi-core CPU processors.

The modified codes (RRTMGPU) have been tested in a stand-alone context, and they have been implemented and tested in WRF_v3.5.1 to ensure negligible changes on the resulting calculated fluxes. The RRTMGPU codes currently require use of the PGI compiler (v13.9 or higher), a recent version of CUDA Fortran, openACC and NVIDIA GPU hardware. Within WRF, running the RT on the GPU significantly decreases the fraction of the total forecast model elapsed time spend on the radiation calculations. This will allow either more frequent radiation calculations or provide the opportunity to improve other model components. The performance enhancement demonstrated here is preliminary, and additional speed-up is possible with further configuration refinement. The specific improvement is also dependent on the specific GPU hardware; faster NVIDIA GPUs are available than the Tesla M-2070Q utilized for these tests.

The RRTMGPU code examined here is a transitional model that retains a number of limitations including a dependence on specific software and hardware. Future work will continue this development effort to produce versions of the RRTMG radiation codes that are optimally designed for efficient parallel processing as well as more general application to enhancing the computational performance of global climate and weather forecast models.

7. ACKNOWLEDGMENTS

Support for the GPU aspects of this work was provided through the Earth System Modeling (ESM) program funded by the U.S. Department of Energy, Office of Science, Biological and Environmental Research program under award number DE-SC0007038. The MIC-related code development and testing was supported by the U.S Department of Commerce through the National Oceanic and Atmospheric Administration, National Centers for Environmental Prediction.

8. REFERENCES

Barker, H. W., R. Pincus, and J.-J Morcrette, 2002: The Monte Carlo Independent Column Approximation: Application within Large-Scale Models. In Proceedings of the *GCSS-ARM Workshop on the Representation of Cloud Systems in Large-Scale Models*, May 2002, Kananaskis, AB, Canada.

Clough, S.A., M.W. Shephard, E.J. Mlawer, J.S. Delamere, M.J. Iacono, K. Cady-Pereira, S. Boukabara, P.D. Brown, 2005: Atmospheric radiative transfer modeling: a summary of the AER codes, *J. Quant. Spectrosc. Radiat. Transfer*, **91**, 233-244.

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

Michalakes, J., Code restructuring to improve performance in WRF model physics on Intel Xeon Phi, in Third NCAR Multi-core Workshop on Programming Earth System Models on Heterogeneous Multi-core Platforms, Boulder, Colorado, 2013, [http://data1.gfdl.noaa.gov/multi-core/2013/agenda.html].

Michalakes, J. and M. Vachharajani, GPU acceleration of numerical weather prediction, *Parallel Processing Letters*, Vol. 18, No. 4, pp. 531–548, 2008. [http://www.worldscientific.com/doi/abs/10.1142/S0129626408003557].

Mlawer, E.J., S.J. Taubman, P.D. Brown, M.J. Iacono, and S.A. Clough, 1997: RRTM, a validated correlated-k model for the longwave. *J. Geophys. Res.*, **103**, 16663–16682, doi: 10.1029/97JD00237.

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