Optimizing CESM for many-core

John Dennis, Chris Kerr, Youngsung Kim, Raghu Kumar, Amogh Simha
Application Scalability and Performance Group, CISL
Something more interesting than computers!

1931 Chevrolet Coupe

2012 Honda Fit
## Objective comparison

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>1931 Chevy Coupe</th>
<th>2012 Honda Fit</th>
<th>Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passengers</td>
<td>2</td>
<td>5</td>
<td>2.5x</td>
</tr>
<tr>
<td>Horsepower</td>
<td>50</td>
<td>117</td>
<td>2.3x</td>
</tr>
<tr>
<td>Max safe speed (MPH)</td>
<td>40-45</td>
<td>80-90</td>
<td>2.0x</td>
</tr>
<tr>
<td>Mileage (MPG)</td>
<td>15</td>
<td>38</td>
<td>2.5x</td>
</tr>
<tr>
<td>Cost in 2012 $</td>
<td>$8,700</td>
<td>$18,000</td>
<td>2.0x (\neq)</td>
</tr>
</tbody>
</table>

2012 Honda Fit is twice the car!
More then twice as nice?

- Automotive industry (81 years)
  - 1931 Chevrolet ➔ 2012 Honda Fit
- Computing capability (4 years)
  - Yellowstone to NWSC-2
- Your code (7 months)
  - MG2
  - Random Number Generator
  - Eulerian Advection in HOMME
Outline

• Motivation
• Group/Team
• Ongoing Efforts
  – MG2
  – RRTMG (RNG)
  – Other
• Coding best practices
• Conclusions
Outline

• Motivation

• Group/Team

• Ongoing Efforts
  – MG2
  – RRTMG (RNG)
  – Other

• Coding best practices

• Conclusions
Team

- Rich Loft, Division Director (NCAR)
- John Dennis, Scientist (NCAR)
- Chris Kerr, Software Engineer (U of Rhode Island), contractor
- Youngsung Kim, Software Engineer (NCAR) /Graduate Student (CU)
- Raghu Raj Prasanna Kumar, Associate Scientist (NCAR)
- Amogh Simha, Graduate Student (CU)
- Nitin Bhat, Graduate Student (Indian Institute for Science [IISC])
- Ravi Nanjundiah, Professor (IISC)
Related Activities

• Intel Parallel Computing Center (IPCC-WACS)
• NESAP (NERSC Exascale Science Application Program)
  – Bi-weekly: NERSC-Cray-NCAR telecon on CESM & HOMME performance (Feb 2015)
• Weekly Intel-TACC-NREL-NERSC-NCAR telecon
  – Concall focused on CESM/HOMME KNC performance
• Strategic Parallel Optimization of Computing
  – NCAR effort focused on Xeon architectures
  – SPOC currently focused on MPAS
Current optimization focus

Xeon and Xeon Phi based platforms

- Sandybridge (SNB) [i.e. Yellowstone]
- Ivybridge (IVB) [i.e. Edison]
- Haswell (HSW)
- Knights Corner (KNC) [i.e. Babbage]
CAM5-SE+MG2
SNB, ne=16 (8x2:120x2)
Intel 14.0.2, OPT=02, sec/day = 26.7
Outline

Motivation

Group/Team

Ongoing Efforts
  – MG2
  – RRTMG (RNG)
  – Other

Coding best practices

• Conclusions
Morrison Gettelman microphysics version 2 (MG2) kernel

- Need code to test optimization techniques
- Characteristics
  - Not too large/not too small
  - Non-trivial
  - Relevant to CESM science plan
  - Expensive (~10% of CAM)
  - Easily accessible
  - Well written
- Engage vendor experts
- Refine optimization techniques
MG2 kernel (con’t)

• Need to optimize kernel rapidly to minimize version skew
  – Given subversion URL: Nov 2014
  – Generate kernel using KGEN: Dec 2014
  – Public release of kernel: Jan 2015
  – Optimized version reintegrated: May 2015

• Used in NWSC-2 benchmark efforts
MG2 kernel timing results on Yellowstone

- Original version: 1075 usec
- Santos optimized version: 875 usec
- Current version: 541 usec

50% reduction in execution time through optimization!
# MG2 kernel timing results (usec)

<table>
<thead>
<tr>
<th></th>
<th>SNB (Intel)</th>
<th>SNB (PGI)</th>
<th>IVB (Intel)</th>
<th>IVB (Cray)</th>
<th>HSW (Intel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>original</td>
<td>1075</td>
<td>1103</td>
<td>843</td>
<td>656</td>
<td>1175</td>
</tr>
<tr>
<td>current</td>
<td>541</td>
<td>600</td>
<td>407</td>
<td>341</td>
<td>401</td>
</tr>
<tr>
<td>Reduction in cost</td>
<td>49.7%</td>
<td>45.6%</td>
<td>51.7%</td>
<td>48.0%</td>
<td>65.9%</td>
</tr>
</tbody>
</table>
How was MG2 optimized?

- Eliminated use of elemental functions
- Pushed vector loop into lower into call stack
- Eliminated divides where possible
- Avoided using slow vendor gamma function
- Rewrote code to minimized use of gamma functions (Santos)
- Rewrote the sedimentation loop (Santos)
- Remove initialization of variables that are overwritten
How was MG2 optimized (con’t)

- Eliminate use of automatic arrays
- Rearrange loops to allow alignment
- Use more aggressive compiler optimizations
Outline

Motivation
Group/Team
Ongoing Efforts
  – MG2
  – RRTMG (RNG)
  – Other
Coding best practices
  • Conclusions
Rapid Radiative Transport Model (RRTMG)

- Produced by Atmospheric & Environmental Research (AER)
- Multiple versions:
  - RRTMG-\{WRF, MPAS\}
  - RRTMG-fast-WRF [Michalakes]
  - RRTMG-CAM (PORT)
  - RRTMG-ECHAM (PSrad)
- Opportunity to examine different design choices
- Compare PORT & Psrad
- Focus on Longwave (LW)
Long-wave rad. in PORT and PSrad (sequential - one thread)

**PORT**

- **Configuration**
  - $\text{NX}=144, \text{NY}=96, \text{NLEV}=26, \text{NCOL}=16$

- **SUBROUTINE rrtmg_lw()**
  - `call mcica_subcol_lw()`
  - `do iplon = 1, ncol`
    - `call inatm()`
    - `call cldprmc()`
    - `call setcoef()`
    - `call taumol()`
    - `call rtrnmc()`
  - `rrtmg_lw()`

  * : Parallel Offline Radiative Transfer

**PSrad**

- **Configuration**
  - $\text{LAT}=96, \text{LON}=192, \text{NLEV}=47, \text{KPROMA}=16$

- **SUBROUTINE lrtm()**
  - `call sample_cld_state()`
  - `call lrtm_coeffs()`
  - `do ig = 1, n_gpts_ts`
    - `do jl = 1, kproma`
      - `call gas_optics_lw()`
      - `end do`
  - `end do`
  - `do ig = 1, n_gpts_ts`
    - `planckFunction()`
  - `end do`
  - `do ig = 1, n_gpts_ts`
    - `call lrtm_solver()`
  - `end do`
How to identify the performance problem?
SNB

- impact of L1 cache miss

![Graph showing the impact of L1 cache miss](image.png)
Analysis of RRTMG

- Three distinctive phases of longwave calculation
  - Random clouds
  - Gas-phase optics
  - Solver

- Optimization strategy for PORT
  - Push column loop down call tree
  - Optimize random number generator (Kerr)
  - Restructure branchy lookup table calls in gas-phase optics (Kumar & Loft)
  - Align and vectorize solver (TBD)
PSrad Performance Analysis on KNC
- Vectorization

Higher vectorization*
* NOTE: VPU_ELEMENTS_ACTIVE includes vectorized data movement including broadcast
RRTMG-RNG

• Add new function to CIME
  – Access to multiple RNG through single interface
  – Shr_rand????
  – KISSVEC (original)
  – Mersene Twister

• Optimization of shr_rand????
Random Number Generator (KISSLVEC) timing results (sec)

<table>
<thead>
<tr>
<th></th>
<th>IVB (Intel)</th>
<th>HSW (Intel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>original</td>
<td>3.64</td>
<td></td>
</tr>
<tr>
<td>current</td>
<td>1.24</td>
<td>1.15</td>
</tr>
<tr>
<td>Reduction in cost</td>
<td>65.9%</td>
<td></td>
</tr>
</tbody>
</table>
Other optimization efforts underway

- Eulerian Advection in HOMME (Dennis)
- Gas-optics \{SW,LW\} in RRTMG (IISC, Kumar, Loft)
- co2calc in POP (Dennis & Kim)
Conclusions

• Demonstrated ability to cut execution time of MG2 by 50-65%
  ▪ Significant improvements are possible
  ▪ Develop methodology/guidelines and pass them on to developers!
  ▪ Incremental improvements not necessary a BIG win from anyone thing
  ▪ MG2 excellent vehicle to engage vendors and explore
  ▪ MG2 will be used as NERSC optimization casestudy
Questions?

dennis@ucar.edu
PORT Performance Analysis on KNC
- Vectorization

PORT (KNC - VPU) - rad_rrtmg_lw

![Graph showing counter ratio per instruction and MIPS over time.

- VPU_INSTRUCTIONS_EXECUTED/ins
- VPU_ELEMENTS_ACTIVE/ins
- MIPS]
PORT Performance Analysis on SNB
- impact of L1 cache miss

PORT (SNB) - rad_rrtmg_lw

Counter ratio per instruction

Time (in ms)

PAPI_L2_DCM/ins  PAPI_L1_DCM/ins  PAPI_LD_INS/ins  MIPS

repeat rrtmg_lw for every column
mcica_subcol_lw
What code look like before and after changes

Before changes:

```fortran
 elemental function wv_sat_svp_to_qsat(es, p)
 result(qs)

   real(r8), intent(in) :: es   ! SVP
   real(r8), intent(in) :: p    ! Current pressure.
   real(r8) :: qs

   ! If pressure is less than SVP, set qs to maximum of 1.
   if ( (p - es) <= 0._r8 ) then
      qs = 1.0_r8
   else
      qs = epsilo*es / (p - omeps*es)
   end if

end function wv_sat_svp_to_qsat
```
What code look like before and after changes

After changes:

```fortran
function wv_sat_svp_to_qsat(es, p, mgncol) result(qs)
  integer,                intent(in) :: mgncol
  real(r8), dimension(mgncol), intent(in) :: es  ! SVP
  real(r8), dimension(mgncol), intent(in) :: p   ! Current pressure.
  real(r8), dimension(mgncol) :: qs
  integer :: i
  do i=1,mgncol
    ! If pressure is less than SVP, set qs to maximum of 1.
    if ( (p(i) - es(i)) <= 0._r8 ) then
      qs(i) = 1.0_r8
    else
      qs(i) = epsilo*es(i) / (p(i) - omeps*es(i))
    end if
  enddo
end function wv_sat_svp_to_qsat
```