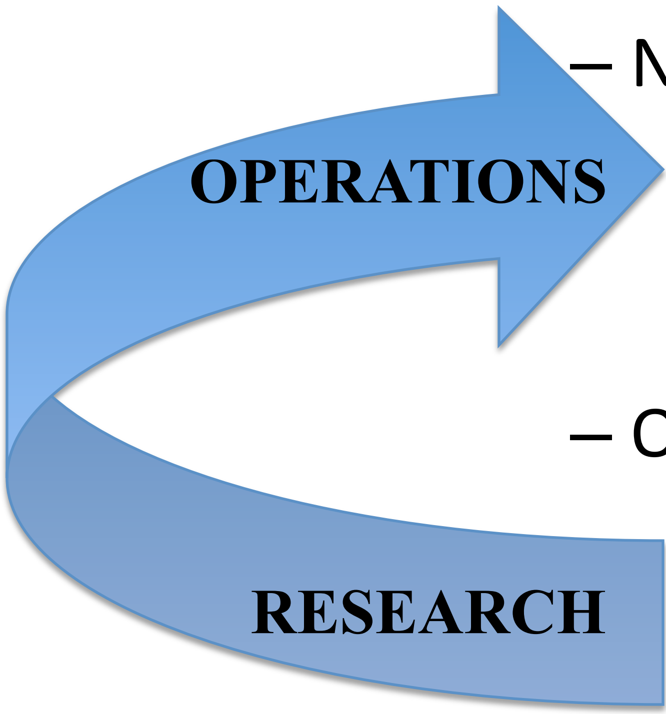
Using GPUs to Run Weather Prediction Models

Mark Govett
Tom Henderson, Jacques Middlecoff,
Paul Madden, Jim Rosinski

November 2010

Organizational Structure

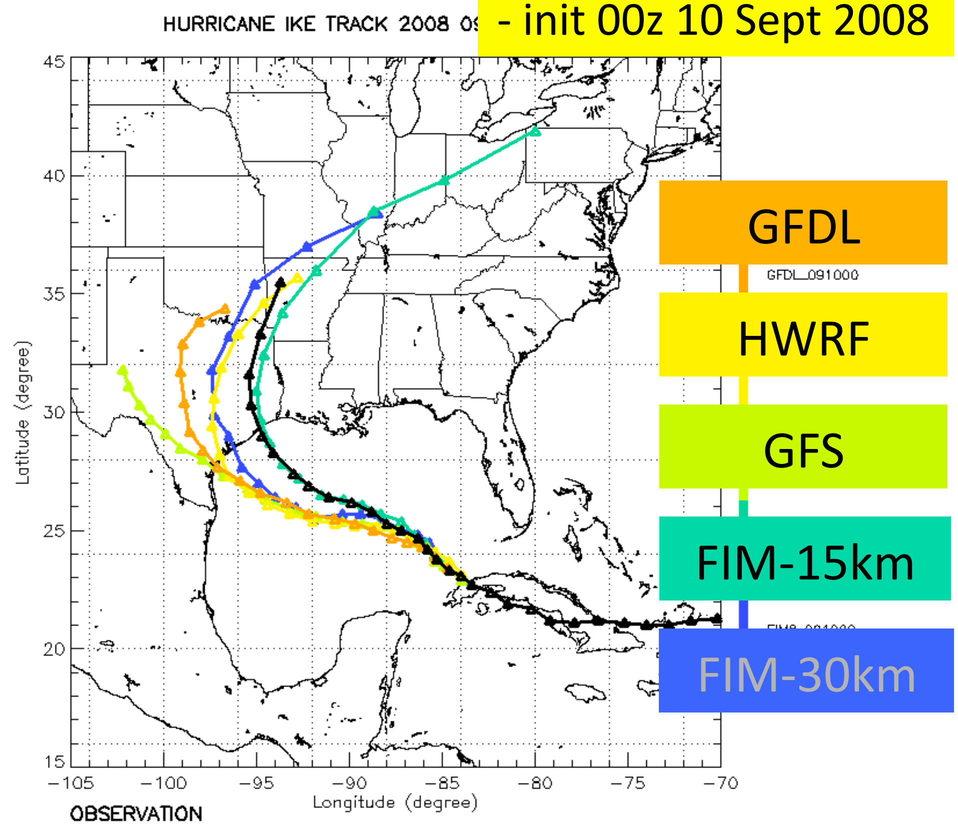


- NOAA
  - – National Weather Service
    - National Centers for Environmental Prediction (NCEP)
  - – Oceanic & Atmospheric Research
    - Earth System Research Laboratory
      - – Global Systems Division
        - » Advanced Computing Section

Model Development Activities

- Regional, Local Models (1-5 KM)
  - NOAA HRRR, WRF-ARW, WRF-NMM, etc
    - Hurricanes, Aviation, Fires, Chemistry / Ash
  - Ensembles (15-30KM)
- Global Models (10-30 KM)
  - NOAA FIM model
  - Improved hurricane forecasts

Hurricane Ike
Model tracks
- init 00z 10 Sept 2008

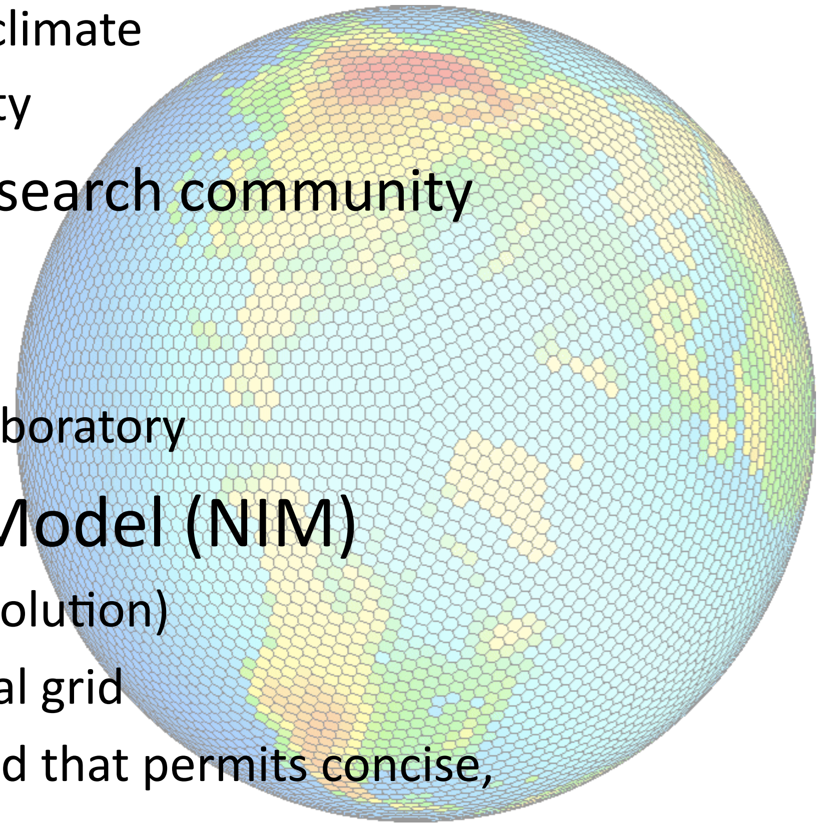


Computing Requirements

- 3000 cores:
  - 15KM Global FIM
- 126,000 cores
  - 21 member 30 KM ensemble

Cloud Resolving Models

- Benefits
  - Clouds have a major influence on weather and climate
  - Improvements in 5-20 day forecasts, climate
  - Improved Hurricane track and intensity
- Active developments within the research community
  - NICAM: University of Tokyo
  - GCRM: Colorado State University
  - NIM: NOAA Earth System Research Laboratory
- Non-hydrostatic Icosahedral Model (NIM)
  - Cloud Resolving Scale (target 2KM resolution)
  - Uniform, hexagonal-based, icosahedral grid
  - Novel indirect addressing scheme used that permits concise, efficient code
    - Used in the hydrostatic FIM model (Operational at NCEP in 2011)



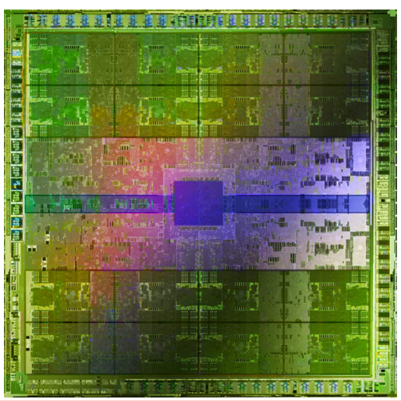
CPU Computing Requirements

- Research and Development
  - CSU's 4KM GCRM was run on 80,000 cores of DOE Jaguar
  - Simulations ran at ~50 percent of real-time
- Operations
  - Models must run at 1-2 percent of real-time
  - NIM performance & scaling study indicates about 200,000 cores would be needed to get to ~3 % real-time
    - System reliability, power requirements, uncertainties in model scaling are big concerns

GPU / Multi-core Technology

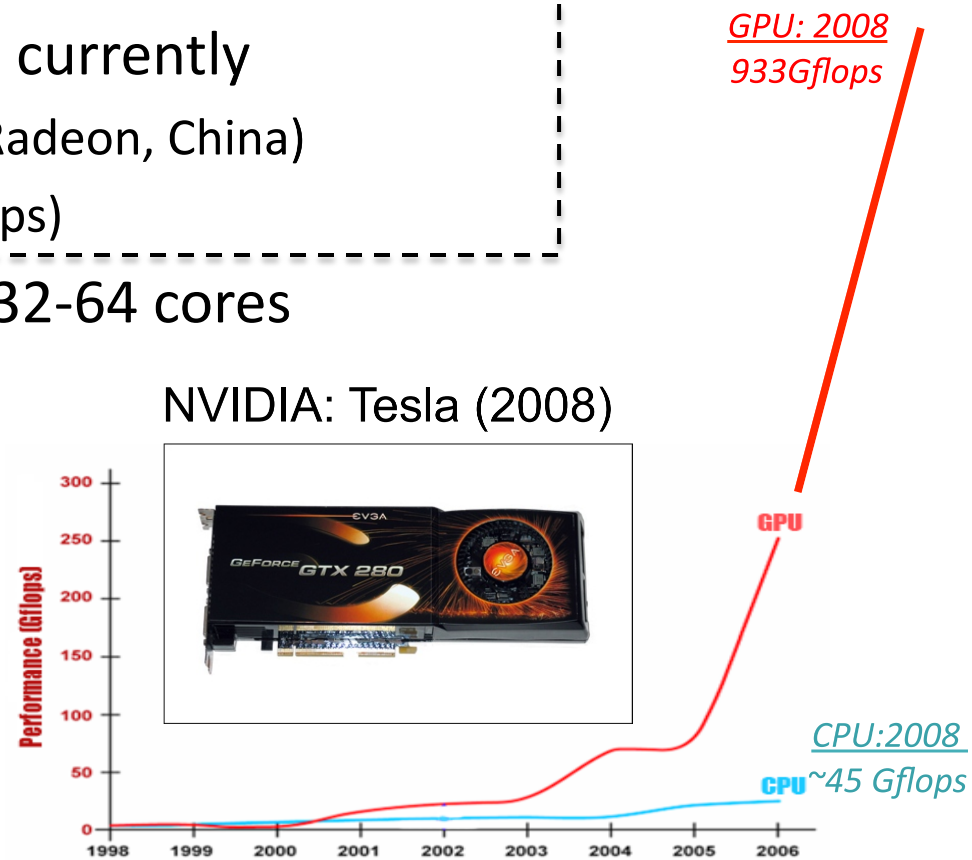
- NVIDIA: Fermi chip first to support HPC
  - Formed partnerships with Cray, IBM on HPC systems
  - #1, #3 systems on TOP500 (Fermi, China)
- AMD/ATI: Primarily graphics currently
  - #7 system on TOP500 (AMD-Radeon, China)
  - Fusion chip in 2011 (5 TeraFlops)
- Intel: Knights Ferry (2011), 32-64 cores

NVIDIA: Fermi (2010)



✧ 1.2 TeraFlops
✧ 8x increase in double precision
✧ 2x increase in memory bandwidth
✧ Error correcting memory

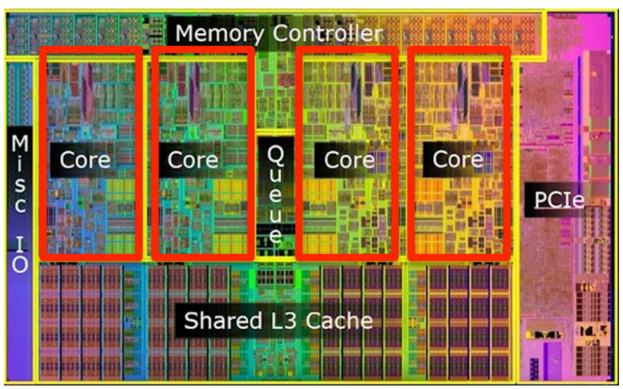
NVIDIA: Tesla (2008)



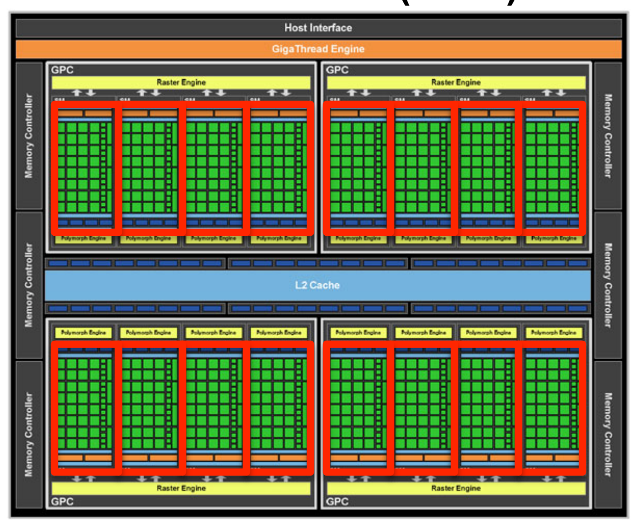
CPU – GPU Comparison

- CPUs focus on per-core performance
  - Chip real estate devoted to cache, speculative logic
  - Westmere: 6 cores, 140 Gflops, 130 Watts (~1 GFlop /Watt)
- GPUs focus on parallel execution
  - Fermi: 512 cores, 1100 Gflops, 220 Watts (~5 Gflops / Watt)

CPU: Nehalem (2009)



GPU: Fermi (2010)



Next Generation Weather Models

- Models being designed for global cloud resolving scales (3-4km)
- Requires PetaFlop Computers

DOE Jaguar System

- 2.3 PetaFlops
- 250,000 CPUs
- 284 cabinets
- 7-10 MW power
- ~ $100 million
- **Reliability in hours**

Equivalent GPU System

- 2.3 PetaFlop
- 2000 Fermi GPUs
- 20 cabinets
- 1.0 MW power
- ~ $10 million
- **Reliability in weeks**

- Large CPU systems (>100 thousand cores) are unrealistic for operational weather forecasting
    - Power, cooling, reliability, cost
    - Application scaling

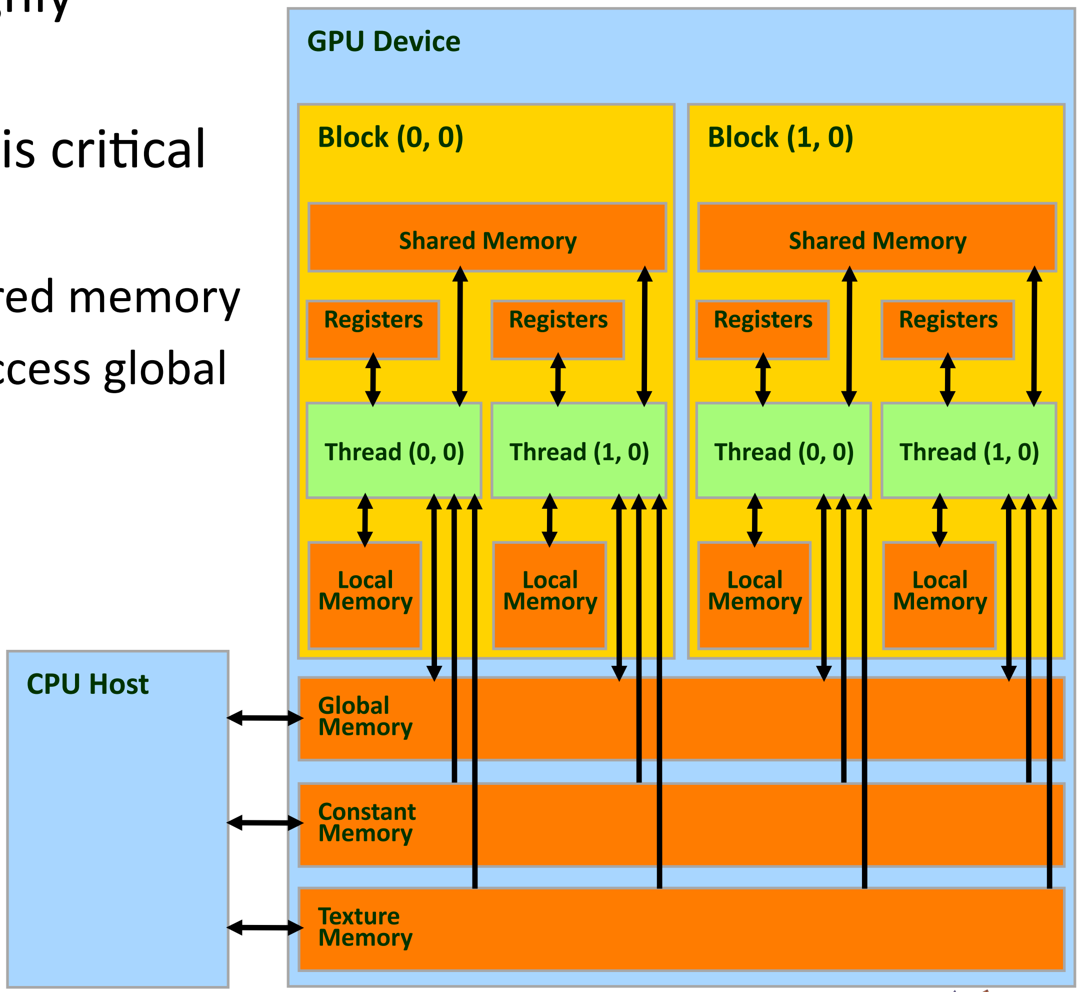
Valmont
Power Plant
~200 MegaWatts
Boulder, CO

Application Performance

- 20-50x is possible on highly scalable codes
- Efficient use of memory is critical to good performance
  - 1-2 cycles to access shared memory
  - Hundreds of cycles to access global memory

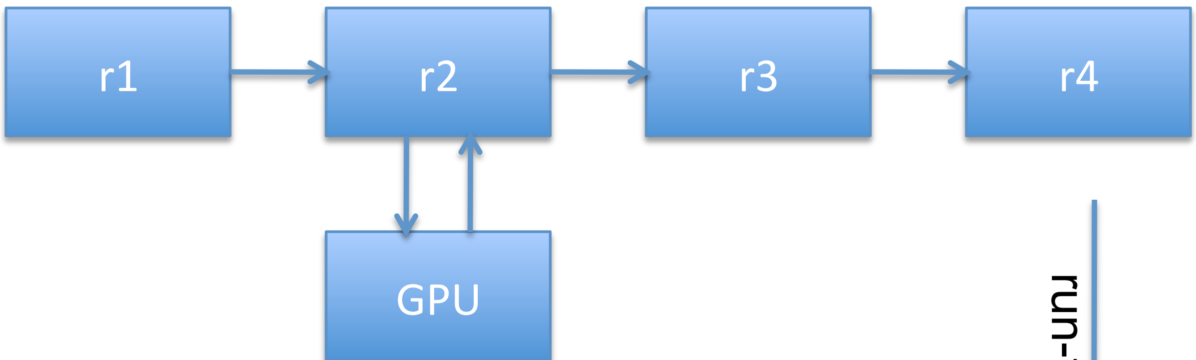
Memory	Tesla	Fermi
Shared	16K	64K
Constant	16K	64K
Global	1-2GB	4-6GB

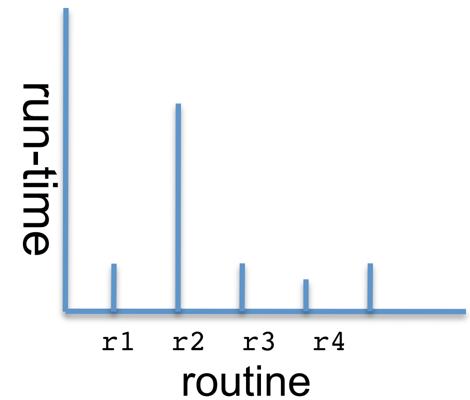
GPU Multi-layer Memory



Execution Flow-control

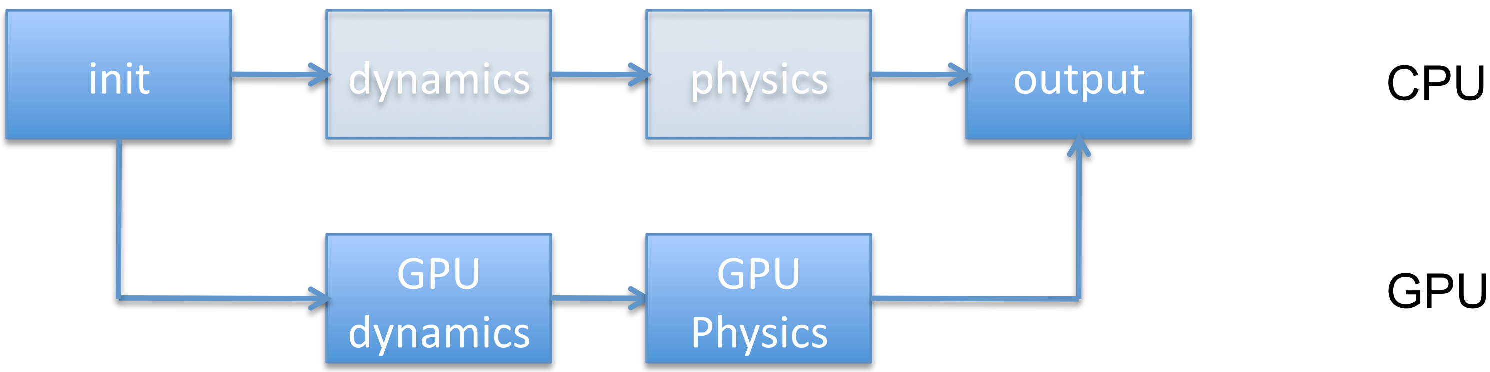
(Accelerator Approach)





– Copy between CPU and GPU is non-trivial
  - Performance benefits can be overshadowed by the copy
  - WRF demonstrated ~6x for one subroutine including data transfers (Michalakes, 2009)
    - – ~ 10x without data transfers

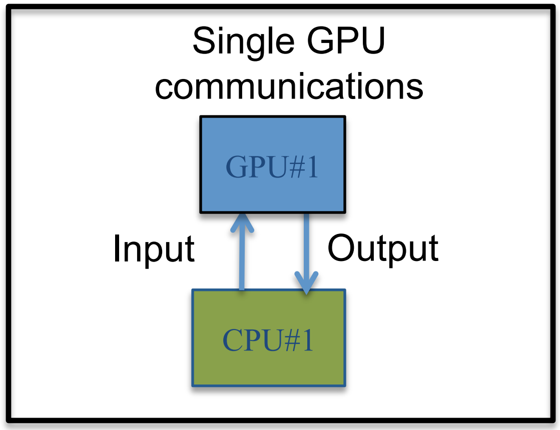
Execution Flow-control
(run everything on GPUs)



- Eliminates copy every model time step
- CPU-GPU copies only needed for input /output, inter-process communications
- JMA: ASUCA model, reported a 80x performance improvement
  - Rewrote the code in CUDA
  - SC2010 Paper: Tuesday 2:30 – 3:00 PM

Code Parallelization (2009)

- Developed the Fortran-to-CUDA compiler (F2C-ACC)
  - Commercial compilers were not available in 2008
  - Converts Fortran 90 into C or CUDA-C
  - Some hand tuning was necessary
- Parallelized NIM model dynamics
    - Tesla Chip, Intel Harpertown (2008)
    - Result for a single GPU
    - Communications only needed for I/O

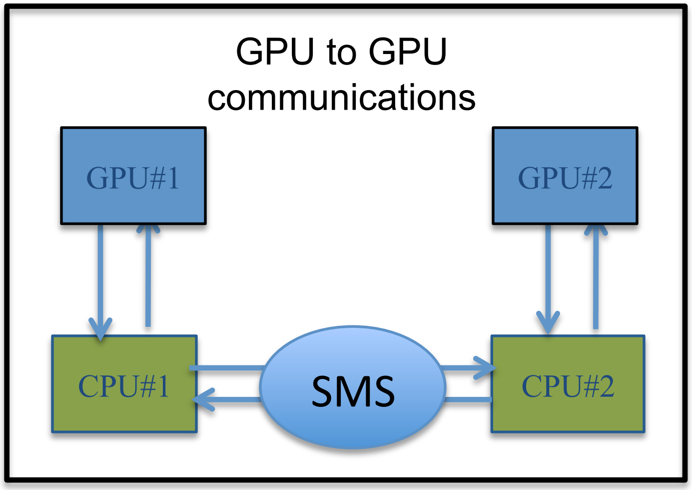


<u>NIM Dynamics (version 160)</u>

Resolution	HorizPts	Harpertown	Tesla	Nehalem	Fermi
G4-480km	2562	2.13	0.079 (26.9)	1.45	0.054 (26.7)
G5-240km	10242	8.81	0.262 (33.5)	5.38	0.205 (26.2)

Model Parallelization (2010)

- Updated NIM Model Parallelization
  - Active model development
  - Code optimizations on-going
- Evaluate Fortran GPU compilers
  - Use F2C results as benchmark
- Evaluate Fermi
- Run on Multiple GPUs
  - Modified F2C-ACC GPU compiler
  - Uses MPI-based Scalable Modeling System (SMS)
  - Testing on 10 Tesla & 10 Fermi GPUs



Fortran GPU Compilers

- General Features
  - Do not support all Fortran language constructs
  - Converts Fortran into CUDA for further compilation
- CAPS – HMPP
  - Extensive set of parallelization directives to guide compiler analysis and optimization
  - Optionally generates OpenCL
- PGI
  - **ACCELERATOR** – directive-based accelerator
  - **CUDA Fortran** – Fortran + language extensions to support Kernel calls, GPU memory, etc
- F2C-ACC
  - Developed at NOAA for our models
  - Requires hand tuning for optimal performance

Run Times for Single GPUs vs. Single Nehalem CPU Core

2652 points / GPU (NIM – G4)

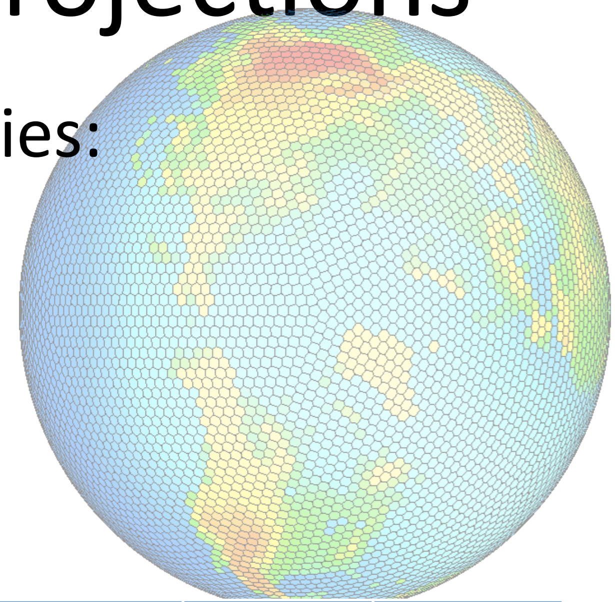
	Harpertown CPU Time	F2C-ACC CUDA-C Tesla GPU Time	HMPP Tesla GPU Time	PGI Tesla GPU Time	F2C-ACC CUDA-C Fermi GPU Time
vdmints	88.86	2.05	2.35	4.78	1.92
vdmintv	37.73	0.94	0.98	0.97	0.75
flux	17.97	0.55	1.05	2.51	0.30
vdn	12.77	0.56	0.73	--	0.53
diag	5.13	0.086	0.085	0.077	0.09
force	5.34	0.11	0.19		0.08
trisol	8.41	1.38	1.38	--	1.14
Total	190.26	6.54 (29.0)	8.12 (23.4)		

Run Times for Single GPUs vs. Single Nehalem

10242 points / GPU (NIM-G5)

	Nehalem CPU Time 10424 pts	F2C-ACC Fermi 10242 pts
vdmints	221.37	7.73 (28.6)
vdmintv	102.58	2.86 (35.9)
flux	56.84	1.17 (48.5)
vdn	17.67	2.02 (8.8)
diag	18.02	0.36 (50.2)
force	15.00	0.37 (40.0)
trisol	9.25	5.4 (1.7)
Total	467.38	21.5 (21.7)

Parallel Performance Projections

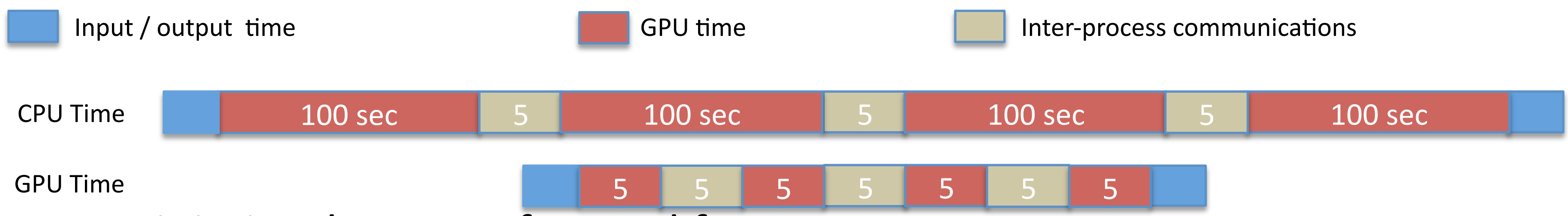


- A doubling of model resolution implies:
  - A 4x increase in horizontal points
  - 2x increase in model time step
  - 4x increase in memory
- GPU global memory limits scaling

	G4	G5	G6	G7	G8	G9	G10	G11
resolution	480KM	240KM	120KM	60KM	30 KM	15 KM	7 KM	3.5 KM
horizontal points	2.5K	10K	40K	160K	640K	2560K	10,000K	40,000K
memory	.25GB	1GB	4GB	16GB	64GB			
tesla	26x	33x						
fermi	26x	26x						
# GPUs	1	1	1	4	16	64	256	1024

Parallel Performance Considerations

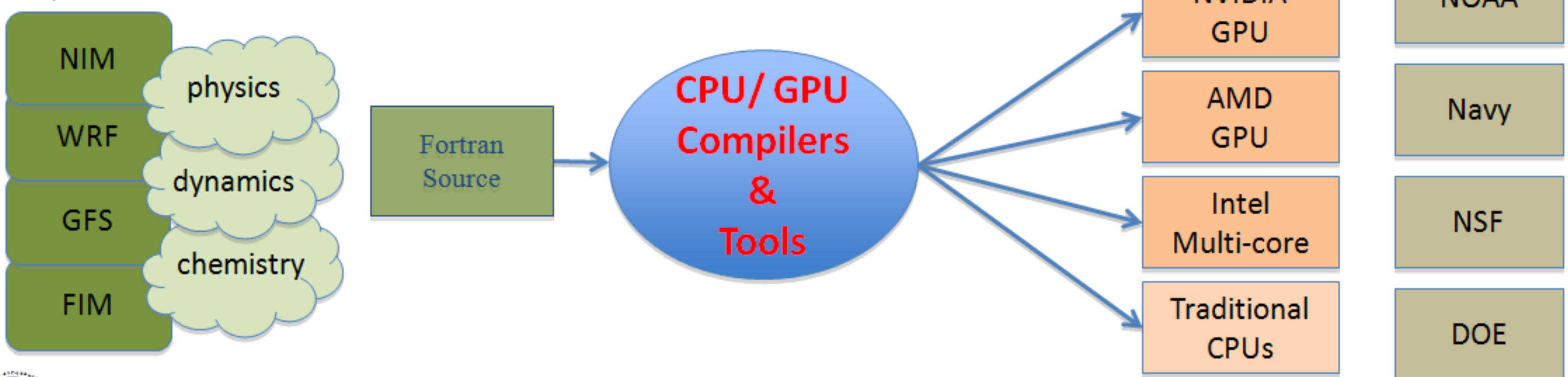
- Application scaling will be limited by the fraction of time spent doing inter-process communications
- Using GPUs, if we get a ~20x speedup in <u>computation</u> time, <u>communications</u> now becomes 50 percent of the runtime.



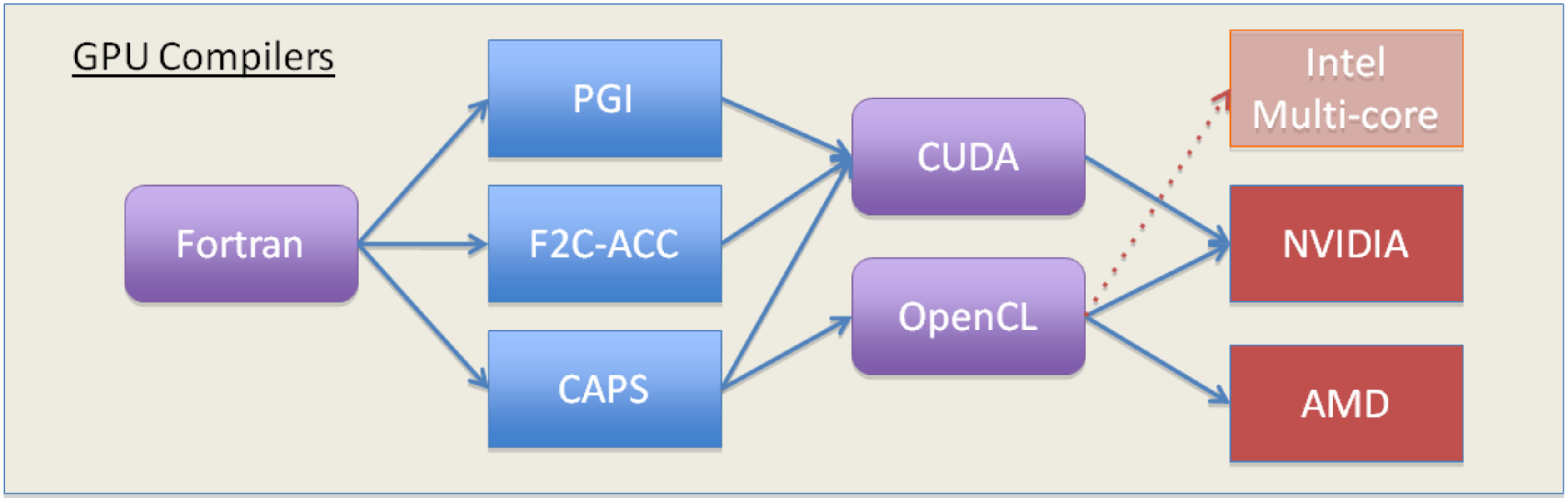
- Minimize data transfers and frequency
  - Trade communications for extra computations
    - GPU computes are "cheap"
- Overlap communications with computations
  - CPU is idle and available
  - Move inter-process communications from just <u>before data is needed</u> to just <u>after the data is updated</u>.

GPUs and the Challenges Ahead

- Performance and Portability
  - Models are becoming increasingly complex
  - Challenge to maintain a single source
    - Operations, research, collaboration
    - Especially for models under active development
- New codes are easier to parallelize
  - Models can be designed to run on GPUs, Multi-core
  - Collaboration between model developers, computer scientists
- Legacy codes will be harder to convert
  - Similar to transition from Vector to MPPs



GPU Performance Portability



- Reliance on NVIDIA, AMD compilers
  - Register allocation inefficient
  - Loop fusion, in-lining, data re-use optimizations are rudimentary
  - Commercial compilers overcome some performance issues
- Requires code changes to achieve good results
  - 2-3x performance benefit was observed (10x becomes 20x in NIM)
  - Different optimizations necessary on CPU than GPU
    - Cache, memory hierarchy
  - GPU architectures will affect performance