GPU tech conference

What We Learned at GTC 2013

- met main developer of espresso-GPU
- FD stencils: tricky, memory bandwidth limited, CPUs as good as GPUs if well written
- new cudaPython from continuum analytics
- more about GPUdirect, RDMA (waiting for driver software)
- can use UVA with our current hardware/software
- GPUdirect and RDMA broken by QPI
- MVAPICH support for above (more cutting edge than openMPI)
- LSF support for GPU clusters
- improved "metrics" for nvprof and racecheck analysis in cuda-memcheck in cuda 5.5
- jun won't have to take the sqrt in quite so ugly a way with new cublas
- no GPU scalapack (could implement by having scalapack call GPU lapack routines)
- upcoming multi-gpu cublas
- molecular dynamics easier than DFT on GPU
- GEMM is inefficient for "narrow" vectors (optimization in progress)
- "Datatype" idea in MPI for moving strided data between nodes

Quantum Espresso GPU Information (from filippo spiga at GTC 2013)

- 2-3x for small systems
- mpi+openmp+gpu
- tried on 2 GPUs per node
- for 8cpu+8gpu maybe run 8 openmpi (not done by anybody, may run into memory bottleneck)
- multiple jobs using same GPU OK
    - espresso enforces memory management
    - prints warning if it runs out of memory
- relies on faster kernels, slower kernels maybe use 1 mpi
- use magma for diagonalization
- matrices on order of 10000x10000
- if system is larger need to switch to scalapack instead of magma
- might have to disable some gpus to maximize bandwidth (pick up ones with most bandwidth)
- no p2p
- gpu code is a "plug-in"
- have support for "screenings"

future development possibility:

- spin magnetization
- potential long-term gain: split real gamma (breaks structure of code)
- porting "phonon" portion of the code
- for PW: people look at EXX but "closed source"?

people:
filippo spiga
girotto

Stencil Work with Samuli

- want to overlap stencil calc (with NO halo) with the halo transfer
- the halo transfer has to be done in X,Y,Z order to get the "corners" right (X,Y,Z transfers are not identical: Y transfer is larger then X, and Z transfer is larger than Y)
- x transmit and receive (for example) can be done at the same time
- stencil changes depending on the system, in particular uses 37 points for non-orthogonal unit cells, and 19 points for orthogonal unit cells?
- 19 point stencils are not sensitive to the halo corners being correct, while other stencils need the corners
- other fields (geophysics etc.) may not need non-orthogonal grids which require different stencils?
- current stencil algorithm optimized for points on axes (e.g. the 19-point stencil): points go into shared memory

Other work with Samuli

- general future of DFT on GPUs (terachem)
    - XC correlation the hard part for terachem
- how to determine bottlenecks (run profiler?)
    - synchronize option (--cuda-sync) that does cudaDeviceSynchronize on timer start/stop. since most of the code runs on the GPU (get very little overlap with CPU) this gives reasonable timings for the bottlenecks (preconditioner, poisson solver are stencil based, orthonormalization and subspace diagonalization is DGEMM based). two big matrix multiplcations: rotate_psi, calc_s_matrix. projections is the lfc integrate.
- lfc.c for rpa (with Jun)

What We Learned at GTC 2012

- how large blocks fit onto the SM (esp. wrt shared memory)
- nersc gpu cluster (dirac) usable?

- email addresses of fft/zher guy
- occupany spreadsheet
    - registers per thread
    - threads per block
    - shared memory per block
- access:
    - shared memory 10 clock ticks
    - global memory: 400*800 clock ticks
- l1*cache/shared memory size cnan be traded off (16/48kB)
- number of streams: 16 for 2.0 devices
- zher improves 50% in cuda 5 ("enable atomics")
- can try GEAM of ZHERK instead of ZHER
- blas2 functions memory bound, blas3 compute bound
- can maybe get access to zher source if we really need it
- 3d ffts might be good
- ucla gpu cluster available?
- openACC ony available for certain compilers (pgi, cray)
- new kepler/cuda5 features:
    - dynamic parallelism
    - hyper*Q for streams
    - slower clock speed more cores
    - mem 5x faster (maybe because of more reg space)
- startup time with multi*gpus is a known problem
- nvvp is dying: eclipse
    - much better information with nsight
- VASP person says we will win with 100x100x100 grid
- use single precision for pre-conditioning
- read the "best-practices" manual to understand nvvp profiling
- universal address space for transfering data
    - P2P for multi*gpus within a process
    - IPC for multi*gpus between processes