Notes

• OpenMP project due Monday
  • questions?

• MPI project grades out today
  • Visible in grades server, and a report by email

• Research project info by tomorrow
  • due dates
  • topics from previous semesters
NVIDIA Tesla P100 GPU

• Recent, but not most recent, NVIDIA high-end GPU (2016)
• Targeted at both HPC workloads and deep learning
  • Supports double precision (64-bit) FP all the way to half-precision (16-bit)
• Compute architecture (fully maxed out):
  • 6 Graphics Processing Clusters (GPCs), each with 10 Streaming Multiprocessors (SMs), 8 512-bit memory controllers, 4 stacks of HBM2 DRAM (16GB)
  • Each SM has 64 CUDA (SIMD) cores, partitioned into 2 32-core blocks, 4 texture units (mainly for graphics operations on bitmap images), 256KB registers
  • Each memory controller has 512KB L2 cache, 2 controllers for each HBM2 memory stack
P100 SM

• 2 blocks of 32 single-precision (FP32) cores (or 32 total double-precision (FP64)), each with instruction buffer, warp scheduler (warp is a set of SIMD threads), 2 dispatch units
  • And 64KB shared memory per SM plus an L1 cache – to gather data for all threads of a warp before loading into registers
  • 4MB L2 cache is shared across all SMs

• Atomic memory operations
  • For shared memory operations (synchronization) between threads/warps (even on different GPUs), using Unified Memory and NVLink
Other new features

• **RDMA in GPUDirect**
  • To allow other devices (e.g., Infiniband, SSD) to directly access memory on multiple GPUs – can help with MPI latency for sends/receives to/from GPU memory

• **HBM2 memory**
  • Provides very high bandwidth DRAM by directly connecting stacks of memory dies vertically, with vias (holes) through the dies to connect them to the GPU die
  • 4 (8 eventually) DRAM dies per stack, up to 8 Gb per die, up to 180GB/sec per stack, max 4 stacks per GPU
  • SECDED error correction
More features

• **NVLink high speed interconnect**
  • High speed bus connecting pairs of GPUs, much higher bandwidth than PCIe – 40GB/sec bidirectional bandwidth
  • Helps support shared memory across GPUs – full support for atomic operations across GPUs
  • For even higher bandwidth, can combine up to 4 links into 1 connection – 160GB/sec
  • Can also be used to connect to NVLink-enabled CPU – example is IBM Power8 (also works for Power9 in Summit)

• **Unified Memory**
  • Basically gives single virtual address space across GPU and CPU memory, so physical pages can be mapped from both sides
  • Helps limit copies, and with irregular memory accesses in warps
  • For performance, still need to maintain locality
  • Simplifies user programs, since no special memory allocator needed
  • Paging mechanism guarantees global coherency across GPU and CPU memory

• **Compute preemption**
  • To interrupt compute tasks (warps) before they complete
  • Same idea as in standard OS, swapping processes in and out of cores
  • Helps with debugging too!
GPUs vs. CPUs

• Study targeting throughput computing
  • Also called streaming applications sometimes, or data parallel

• Architectural limits to parallelism
  • CPUs have limited number of cores
  • GPUs have limited capabilities, e.g. no caches (not true now)

• End results, on a set of representative benchmarks, is that GPU performs 2.5X faster than CPU
  • Application kernels include linear algebra (SGEMM from BLAS), Monte Carlo, Convolution, FFT, SAXPY (from BLAS), Lattice Boltzman (CFD), Constraint Solver, Sparse Matrix/Vector Multiply, Collision Detection (virtual environments), Radix Sort, Ray Casting, Index Search, Histogram, Bilateral Filter (image processing)
  • Platforms are Intel Core i7 CPU (4 hyper-threaded cores, 4-wide SIMD units, and caches) and NVIDIA GTX280 GPU (array of 30 SMs, each with 8 scalar processing units and local memory)
GPUs vs. CPUs

- **Main advantage of CPU is caches**
  - For fast single thread performance, but also helps with multi-threaded apps
  - Disadvantage is complexity, limiting number of cores per chip
  - Also have fast synchronization

- **Main advantage of GPU is high throughput**
  - Each instruction for an SM executes on 8 scalar units (32 data elements)
  - Disadvantage is need to move data explicitly into (small) SM memory from large shared memory
  - Also have support for gather/scatter from memory and special functional units (e.g., texture sampling, math ops)

- Performance measurements for GPU assume data already in GPU memory (from other GPU computations)

- Overall performance of GPU (geometric mean) is 2.5X of CPU ($n^{th}$ root of product of speedups)
  - Why? Because they optimized both CPU and GPU versions of the kernels