Data Parallel Architectures - SIMD

Motivation
Vectors
Multimedia SIMD
GPUs
Graphics Processing Units (GPUs)

Graphics accelerators
   Heterogeneous computing: Host CPU + GPU (device)
   Great for graphics: exploit lots of data parallelism
   Can we use GPUs for other computing?

Multiple forms of parallelism
   MIMD, SIMD, ILP, Multithreading

How to program?
   2007: Nvidia developed a C like language
      Cuda: Compute Unified Device Architecture
   2009: Khronos group released OpenCL
   Recent: Heterogeneous System Architecture (HSA): unified virtual address space
Our Focus: Nvidia GPUs + CUDA

Hardware for Maxwell*

* Images from http://www.anandtech.com/show/8526/nvidia-geforce-gtx-980-review/3
CUDA Programming Model

Single Instruction Multiple Thread (SIMT)
CUDA thread is the unifying parallelism construct
Thread -> (Warp) -> Thread block -> Streaming multiprocessor (SM)
Each SM executes a thread block, one warp at a time
Each warp (32 threads) is executed in SIMD fashion

Computation structured in a grid of thread blocks and threads

    functionName<<dimGrid, dimBlock>>(parameter list)

Threads use blockIdx (which block), threadIdx (which thread in block),
blockDim (dimension of block) to determine which element to
compute on

Thread scheduling and management in hardware
Programming Model: Memory

Originally, separate memories for CPU/GPU: host vs. device or global
Device/global memory accessible by all SMs
Recent trend – shared virtual memory, integrated CPU+GPU
for (int i=0; i<n; i++)
    y[i] = a*x[i] + y[i]

CUDA:
E.g., n threads, one per vector element, 256 threads per thread block

_host_
int nblocks = (n+255)/256
daxpy<<<nblocks,256>>>(n,2.0,x,y)

device_
void daxpy(int n, double a, double *x, double *y)
{
    int i = (blockIdx * blockDim) + threadIdx;
    if (i < n) y[i] = a*x[i] + y[i]
}
Figure 4.14 Simplified block diagram of a Multithreaded SIMD Processor. It has 16 SIMD lanes. The SIMD Thread Scheduler has, say, 48 independent threads of SIMD instructions that it schedules with a table of 48 PCs.
Maxwell SM Details

64 concurrent warps
64K 32 bit registers
32 maximum thread blocks
64KB to 96KB scratchpad (shared memory)
2MB shared L2
Key Features and Challenges

Hardware managed thread/thread block scheduling
  - Thread blocks to SMs
  - Warps within SM

Multithreading for latency tolerance

Scratchpad aka Shared memory

Caches, Coherence, Consistency

Synchronization between SMs through atomics

SIMD Divergence

Future of accelerators/heterogeneous computing?