Introduction to GPU Computing
Add GPUs: Accelerate Science Applications
GPUs Accelerate Science
Small Changes, Big Speed-up

Application Code

Compute-Intensive Functions
Use GPU to Parallelize

Rest of Sequential CPU Code
3 Ways to Accelerate Applications

Applications

Libraries
“Drop-in” Acceleration

OpenACC Directives
Easily Accelerate Applications

Programming Languages
Maximum Performance
<table>
<thead>
<tr>
<th>Application</th>
<th>Description</th>
<th>Systems</th>
<th>Arch</th>
<th>Coding</th>
<th>Platform</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAYADA</td>
<td>Sequential mapping software</td>
<td>8, 16, 32, 64</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>CHARMM</td>
<td>Multi-scale molecular dynamics simulations of proteins, DNA, and lipids</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>LEAGIE</td>
<td>Analysis of molecular mechanics of biomolecules, including proteins, DNA, and RNA</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>AMBER</td>
<td>Simulation of molecular and chemical processes</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>DL_POLY</td>
<td>Simulation of polymers, proteins, and chemical reactions</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>CHARMM</td>
<td>Multi-scale molecular dynamics simulations of biomolecules</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>SPROCS</td>
<td>Simulation of biochemical reactions with comprehensive interactions</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>GROMACS</td>
<td>Classical molecular dynamics package</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>QM-PARVE</td>
<td>Quantum chemistry package for electronic structure calculations</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>LEAP</td>
<td>Quantum chemistry package for electronic structure calculations</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>GAMESS</td>
<td>Quantum chemistry package for electronic structure calculations</td>
<td>32, 64, 128</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
<tr>
<td>OpenKmer</td>
<td>Sequence mapping software</td>
<td>8, 16, 32, 64</td>
<td>X86</td>
<td>Yes</td>
<td>Linux</td>
<td>Single only</td>
</tr>
</tbody>
</table>

272 GPU-Accelerated Applications
www.nvidia.com/appscatalog
3 Ways to Accelerate Applications

Applications

Libraries
“Drop-in” Acceleration

OpenACC Directives
Easily Accelerate Applications

Programming Languages
Maximum Flexibility
Libraries: Easy, High-Quality Acceleration

- **Ease of use:** Using libraries enables GPU acceleration without in-depth knowledge of GPU programming
- **“Drop-in”:** Many GPU-accelerated libraries follow standard APIs, thus enabling acceleration with minimal code changes
- **Quality:** Libraries offer high-quality implementations of functions encountered in a broad range of applications
- **Performance:** NVIDIA libraries are tuned by experts
Some GPU-accelerated Libraries

NVIDIA cuBLAS
NVIDIA cuRAND
NVIDIA cuSPARSE
NVIDIA NPP

GPU VSIPL
CUDA tools
MAGMA
NVIDIA cuFFT

Vector Signal Image Processing
GPU Accelerated Linear Algebra
Matrix Algebra on GPU and Multicore

ROGUE WAVE SOFTWARE
IMSL Library
ArrayFire Matrix Computations
Sparse Linear Algebra

C++ STL Features for CUDA

ArrayFire Matrix Computations
Sparse Linear Algebra
C++ STL Features for CUDA

NVIDIA cuFFT
IMSL Library
Sparse Linear Algebra
C++ STL Features for CUDA
Explore the CUDA (Libraries) Ecosystem

CUDA Tools and Ecosystem described in detail on NVIDIA Developer Zone:

developer.nvidia.com/cuda-tools-ecosystem
3 Ways to Accelerate Applications

- Libraries
  - “Drop-in” Acceleration

- OpenACC Directives
  - Easily Accelerate Applications

- Programming Languages
  - Maximum Flexibility
OpenACC Directives

Program myscience
... serial code ...
!$acc kernels
do k = 1,n1
do i = 1,n2
... parallel code ...
enddo
enddo
!$acc end kernels
... End Program myscience

Simple Compiler hints

Compiler Parallelizes code

Works on many-core GPUs & multicore CPUs
2 Basic Steps to Get Started

Step 1: Annotate source code with directives:

```fortran
!$acc data copy(util1,util2,util3) copyin(ip,scp2,scp2i)
!$acc parallel loop
...
!$acc end parallel
!$acc end data
```

Step 2: Compile & run:

```
pgf90 -ta=nvidia -Minfo=accel file.f
```
OpenACC Directives Example

```c
$acc data copy(A,Anew)
iter=0
do while ( err > tol .and. iter < iter_max )
    iter = iter +1
    err=0._fp_kind

$acc kernels
    do j=1,m
        do i=1,n
            Anew(i,j) = .25_fp_kind *( A(i+1,j ) + A(i-1,j ) &
                         +A(i ,j-1) + A(i ,j+1))
            err = max( err, Anew(i,j)-A(i,j))
        end do
    end do
$acc end kernels
    IF(mod(iter,100)==0 .or. iter == 1)    print *, iter, err
A= Anew
end do
$acc end data
```

- **Copy arrays into GPU memory within data region**
- **Parallelize code inside region**
- **Close off parallel region**
- **Close off data region, copy data back**
3 Ways to Accelerate Applications

- Libraries
  - “Drop-in” Acceleration

- OpenACC Directives
  - Easily Accelerate Applications

- Programming Languages
  - Maximum Flexibility
GPU Programming Languages

**Numerical analytics**
- MATLAB, Mathematica, LabVIEW

**Fortran**
- OpenACC, CUDA Fortran

**C**
- OpenACC, CUDA C

**C++**
- OpenACC, Thrust, CUDA C++

**Python**
- PyCUDA, Copperhead

**C#**
- Cudafy

**Java, R, DSL’s, etc.**
Get Started Today

These languages are supported on all CUDA-capable GPUs.
You might already have a CUDA-capable GPU in your laptop or desktop PC!

CUDA C/C++

Thrust C++ Template Library
http://developer.nvidia.com/thrust

CUDA Fortran

PyCUDA (Python)
http://mathema.tician.de/software/pycuda

cudafy
http://cudafy.codeplex.com

MATLAB
http://www.mathworks.com/discovery/matlab-gpu.html

Mathematica
Where to get help?

- Sign up as a registered developer: https://developer.nvidia.com/
- Access the NVIDIA community forums: https://devtalk.nvidia.com/
- OpenACC: http://www.openacc-standard.org/
- StackOverflow:
  - CUDA: http://stackoverflow.com/questions/tagged/cuda
  - Thrust: http://stackoverflow.com/questions/tagged/thrust
  - OpenACC: http://stackoverflow.com/questions/tagged/openacc
Getting Putty

http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe
Thank you
developer.nvidia.com