OpenCL

Mike Bailey
mjb@cs.oregonstate.edu
Oregon State University
But, the problem is that you have to use a vendor-specific API.
OpenCL – Vendor-independent GPU Programming

Your OpenCL Code

CTM code
CTM Compiler and Linker
OpenCL for AMD/ATI Systems

or

or

CUDA code
CUDA Compiler and Linker
OpenCL for NVIDIA Systems

or

KC code
KC Compiler and Linker
OpenCL for Knights Corner Systems

This happens in the vendor-specific driver
OpenCL

- A C-like language, originally proposed by Apple, now an industry standard
- Like CUDA, OpenCL can share data with OpenGL
- You write one program, but designate a C/C++ part of it to run on the CPU and an OpenCL part to run on the GPU
- You can’t ask for threads in the OpenCL part, but the translation process might create them for you. (Also, you can use them in the CPU part via OpenMP, pthreads, etc.)
OpenCL wants you to break the problem down

```c
void mul( int n, float *a, float *b, float *c) {
    int i;
    for ( int i = 0;  i < n;  i++ )
        c[i] = a[i] * b[i];
}

kernel void mul( global float *a, global float *b, global float *c) {
    int id = get_global_id (0 );
    c[id] = a[id] * b[id];
}
```
OpenCL also supports vector parallelism

Part of OpenCL is vector-oriented, meaning that it can perform a single instruction on multiple data values at the same time (SIMD). Vector data types are: char\texttt{n}, int\texttt{n}, float\texttt{n}, where \texttt{n} = 2, 4, 8, or 16.

```cpp
float4 f, g;
f = (float4)(1.f, 2.f, 3.f, 4.f);

float16 a16, x16, y16, z16;
f.x = 0.;
f.xy = g.zw;
x16.s89ab = f;

float16 a16 = x16 * y16 + z16;
```