# Intel's Larrabee

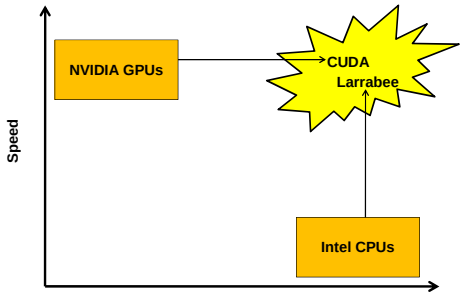
Mike Bailey

Oregon State University

Credit: Many of the figures in these notes came from Larry Seiler's *Larrabee Hardware Architecture* presentation



## Reaching the Promised Land






## Larrabee Characteristics

- x-86 instruction set
  - Compatible with all other Intel CPU chips
  - Have extra instructions for vector units (more on this later)
- 32 cores
  - Each core can have 4 threads per core (i.e., 4 sets of independent register state per core)
  - Each core has its own L1 and L2 cache
  - Each core has a vector unit
  - There are new instructions for vector units (more on this later)
- L2 Caches
  - Fully coherent among the cores
- Intel considers Larrabee a GPU and is marketing it as such
  - But, with much more convenient programmability than other GPUs
  - Doesn't have much fixed-function graphics hardware (texture sampling is about it)
  - Will do most of the graphics pipeline in multicore, vectored software

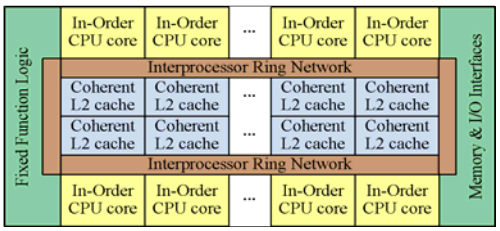


## Cache Behavior

- Independent 32K L1 Instruction and 32K L1 Data caches per core
  - 8-way, 64 Bytes/cache line
  - 64 sets of 8 lines per set
  - Non-Blocking: If 1 thread has a cache miss, other threads keep going
- 256K L2 Cache per Core
  - 4096 lines, 512 sets, 8-way, 64 Bytes/cache line
  - Architectural Central Tag Directory for Coherence
    - Makes programming easier because hardware ensures code always receives the most recent revision of data
- The caches are connected with a Ring Bus
  - 512 bits in each direction



## Larrabee Block Diagram

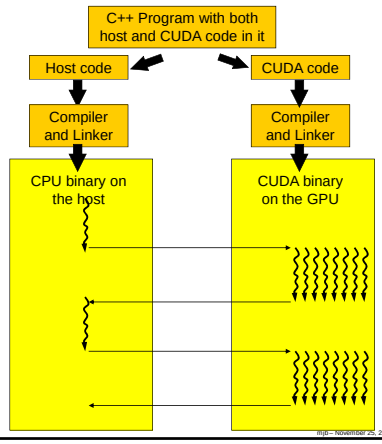





## Two Kinds of Parallelism

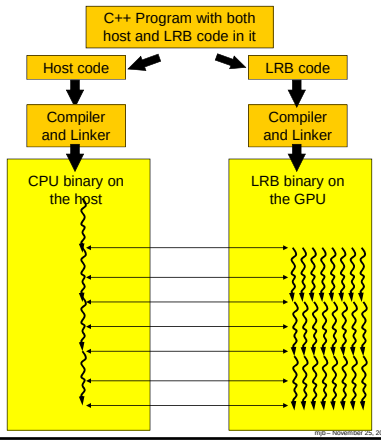
- Two kinds of parallelism:
  - Vectors:
    - Good when there is tight synchronization
    - Bad when data being processed follows different paths
  - Threads
    - Bad when there is tight synchronization
    - Fine when data being processed follows different paths

## The CUDA Paradigm



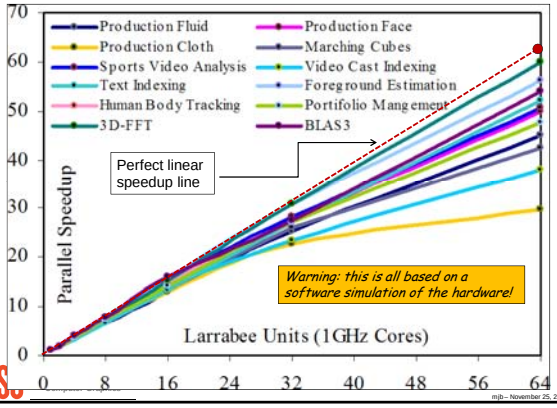



## The Larrabee Paradigm






## Good Behavior When Amdahl's Law is Working for You






## Larrabee and OSU

- We have a research project with Intel, and are using a Larrabee system remotely (one of only a handful of universities that have this access)
- We will be getting 4 or more Larrabee systems here next year
- Might be offering a Larrabee course in Fall 2010?



## Vector Unit

- Operates on 512 bits (=16 floating point numbers) at a time
  - All 16 operations happen in one clock
  - Larrabee's 32 cores can compute 512 floating point operations per clock
  - When fully kept busy, will produce a throughput of 1.5 teraflops
  - C language inline routines called instead of using the assembly language

**Examples:**

**Simple vector multiply:**

```
vmulps v0, v1, v2      ; v0 = v1 * v2
                       ; v's can be registers or memory locations
```

Multiply-add, destination can be the third source:

```
vmadd231ps v0, v1, v2  ; v0 = v1 * v2 + v0
```

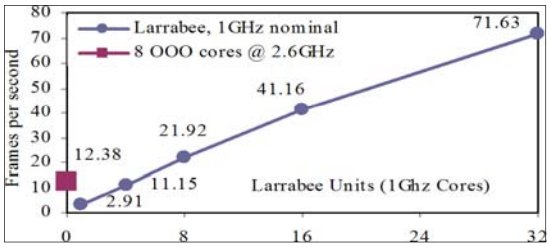
Mask the writing of the elements:

```
vmulps v0 {k1}, v1, v2 ; only some of the result is written to v0
```



## Realtime Ray-tracing (?)

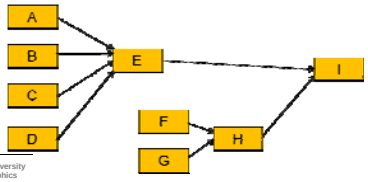
## Expected Ray-trace Performance






## Larrabee Native Multithreading

- pthreads and OpenMP are supported
- Larrabee's multitask library, called *XNTask*, is the "native" multithreading API
- Arrange tasks in a dependency graph
- When a node's required inputs are all available, that node can run
- Can support DLP, TLP, or the pipeline pattern
- Nodes can have different priorities






## Some References


http://en.wikipedia.org/wiki/Larrabee_(GPU)

Seiler, L., Carmean, D., et al., *Larrabee: A many-core x86 architecture for visual computing. SIGGRAPH 2008 Conference Proceedings*, August 2008.

A First Look at the Larrabee New Instructions: http://www.ddj.com/architect/216402188

Rasterization on Larrabee: http://www.ddj.com/architect/217200602

Game Physics Performance on the Larrabee Architecture: http://download.intel.com/technology/architecture-silicon/GamePhysicsOnLarrabee_paper.pdf