

# CS4803DGC Design Game Consoles

Spring 2010

Prof. Hyesoon Kim



**Georgia  
Tech**



College of  
Computing



# Lectures are from

- Reading assignment [LRB]
- <http://www.ece.neu.edu/groups/nucar/GPGPU/GPGPU-2/Seiler.pdf>



# Motivation for LRB

- Economic reason: Game market is getting bigger
- Technology reason: Power wall, memory wall. → Heterogeneous architecture shows promising results

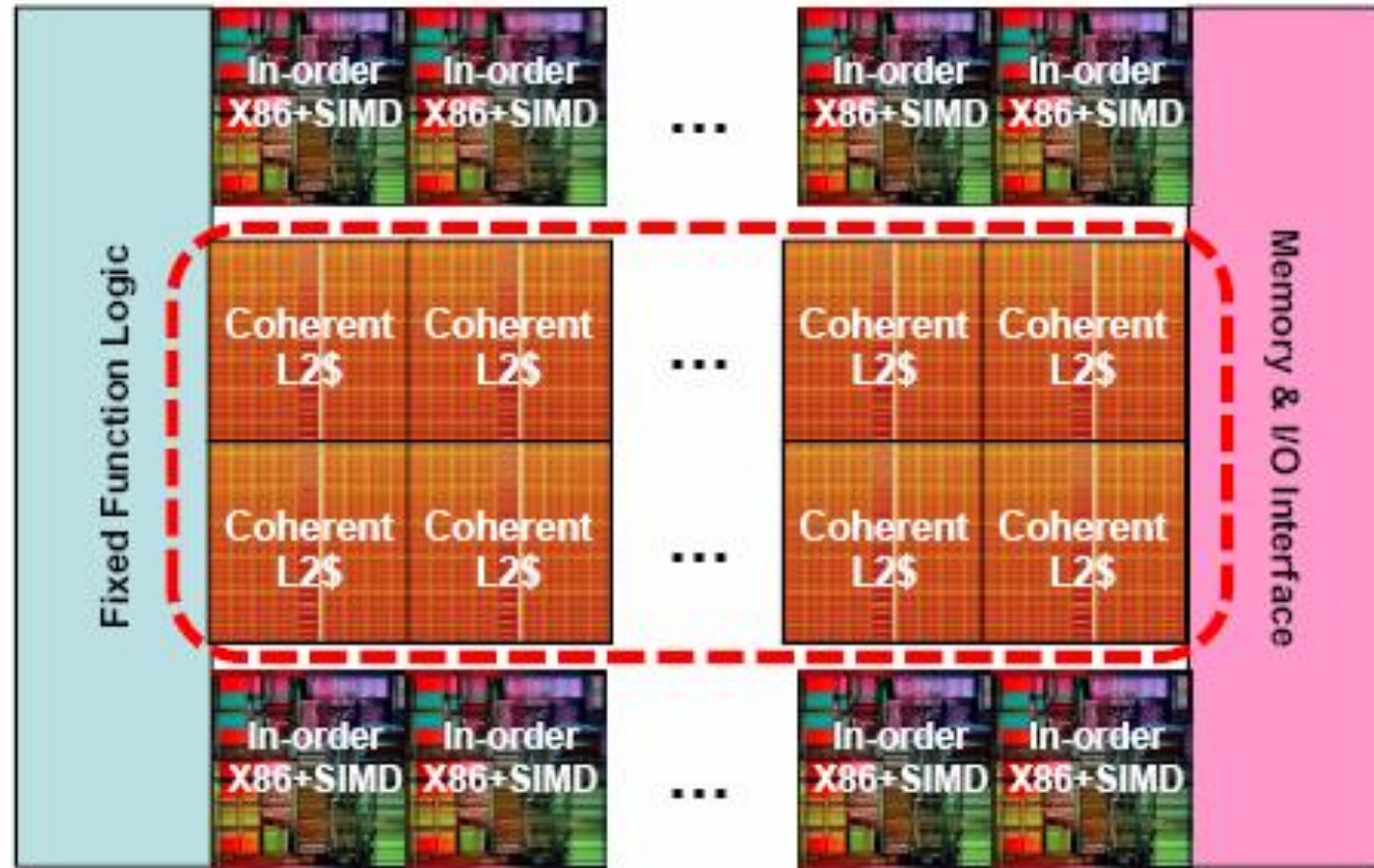


# In-order vs. OOO

# CPU cores:	2 out-of-order	10 in-order
Instruction issue:	4 per clock	2 per clock
VPU per core:	4-wide SSE	16-wide
L2 cache size:	4 MB	4 MB
Single-stream:	<b>4 per clock</b>	<b>2 per clock</b>
Vector throughput:	<b>8 per clock</b>	<b>160 per clock</b>

Detour: OOO

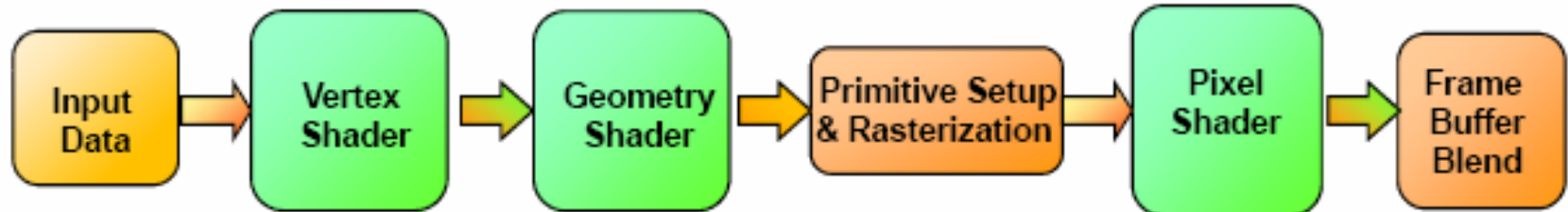
# Larrabee Architecture



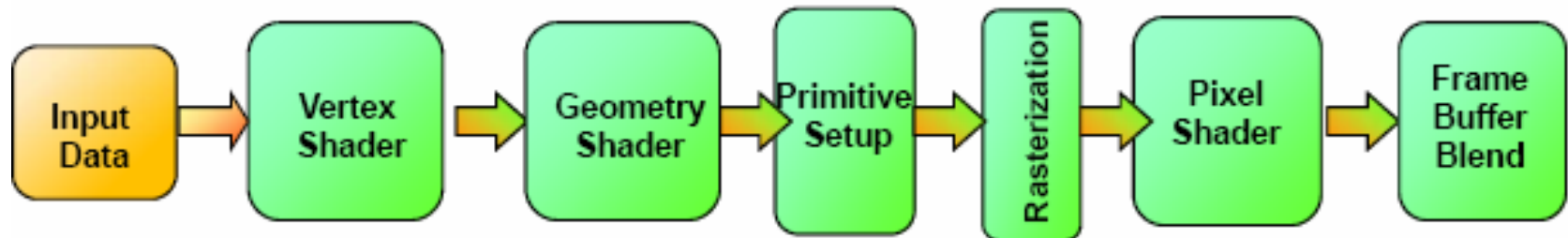
- 4-way SMT in order processor with cache coherence
- Extended X86 ISA
- Fixed functions: texture filtering



# Programmable Pipeline Comparison

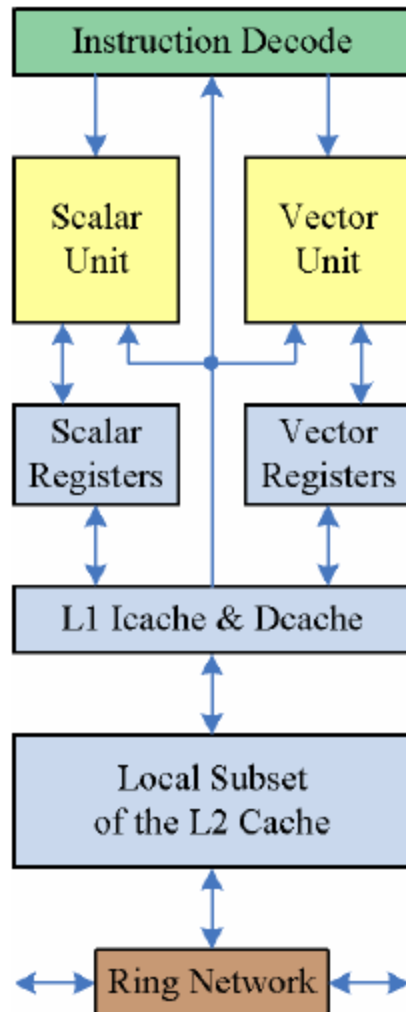


Conventional GPGPU pipeline (base on DirectX10)



Larrabee's fully programmable pipeline

# Core



- Pentium processor in-order
- Extended X86 (64bit, new instructions)
- 4-way SMT
- 32KB I-cache, 32KB D-cache (statically partitioned)
- 256KB Local L2 cache (subset of L2 cache)



# Dual issue

- U-pipe V-pipe
- Primary pipeline: All instructions (U-pipe)
- Secondary: Limited instructions (V-pipe)

loads, stores, simple ALU operations, branches, cache manipulation instructions, and vector stores.

- Reply on compiler's paring
  - VLIWish again



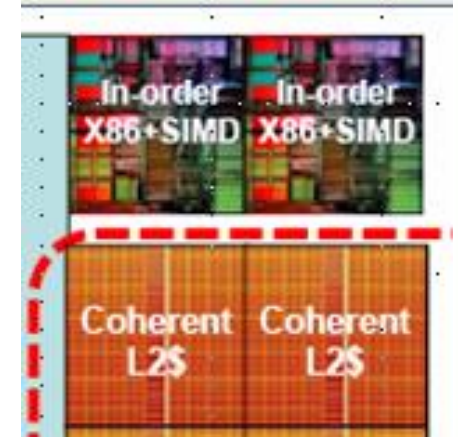


# Cache managements

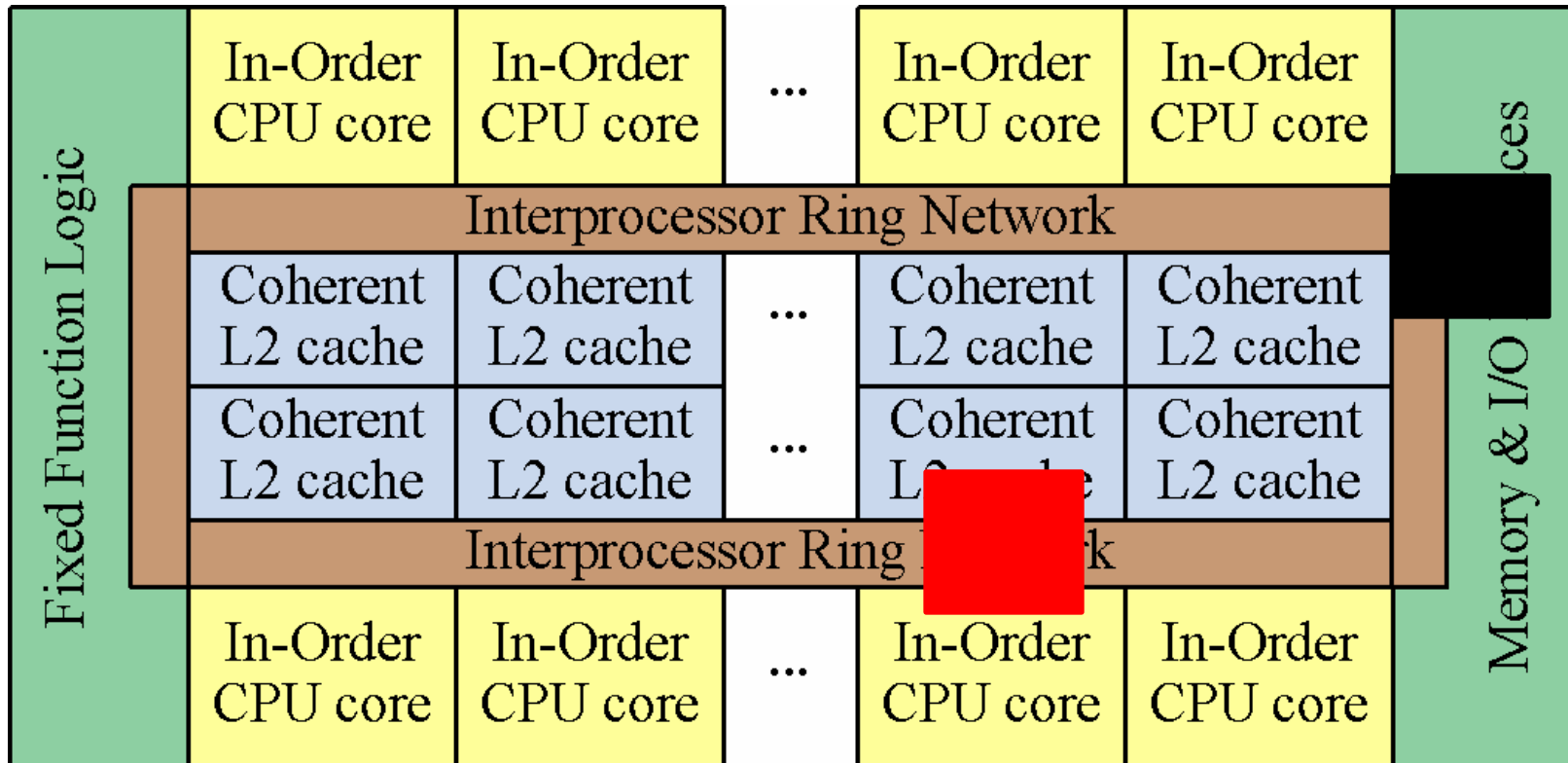
- Use cache as extended register file storage
- Target for Stream applications
- Each core can
  - Fast-access its local subset of L2 (256KB) – Access other's L2 shares too
  - Control for non-temporal streaming data (SSE)
  - Prefetch to L1, or L2 only
  - Mark a streaming cache line for early eviction
  - Render target kept in L2 (e.g., FB, ZB, SB, etc)

# L2-Cache and Ring network

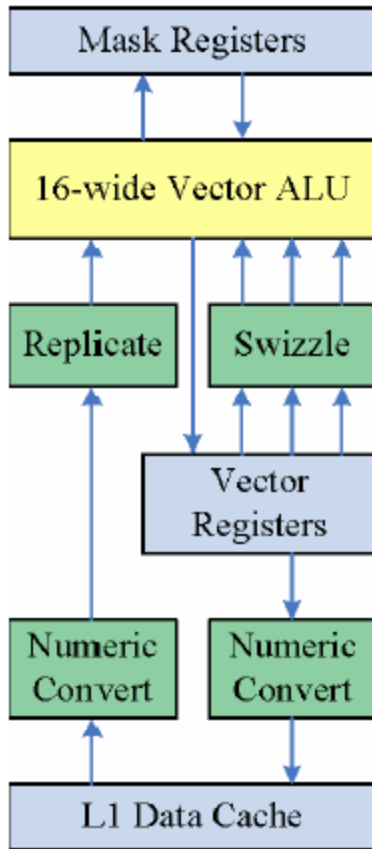
- Global L2 cache is divided into 256KB  
Local L2 per core
- Data written by a CPU core is stored in its own L2 cache subset and is flushed from other subsets, if necessary
- Bi-direction Ring network (<16)
  - Even cycle, odd cycle: one clock per one hop
- Each ring data-path is 512-bits wide per direction
- L2 cache Insertion requires cache coherence checking
- Memory and fixed function access



# Ring Network



# VPU (Vector Processor Units)

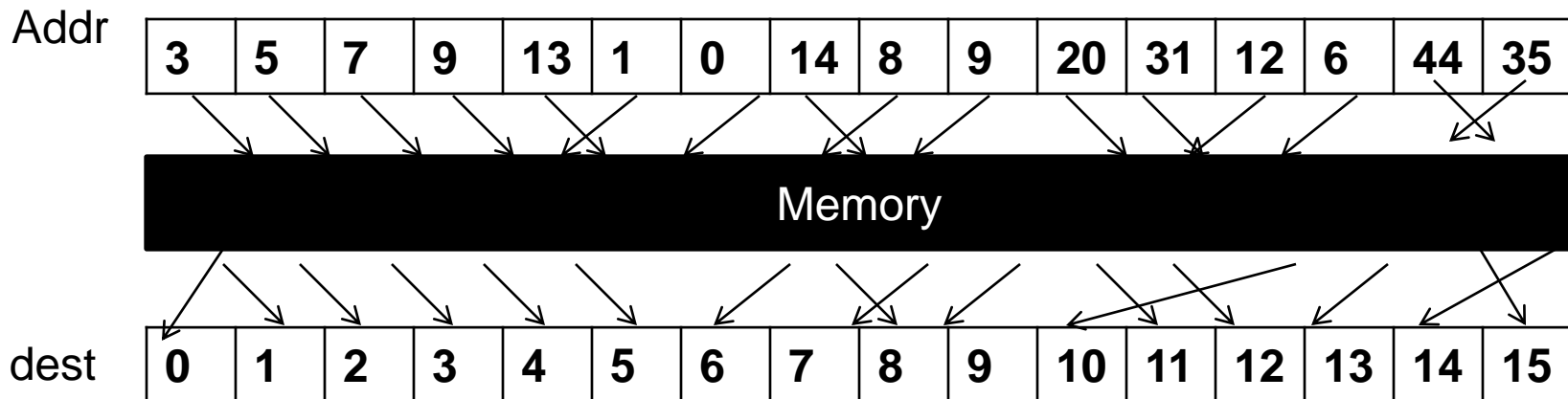


- 16-wide SIMD unit
  - 16 wide Single precision
  - 8-wide double precision
- Hardware scatter/gathering operations : 16 elements are loaded from or stored to up to 16 different addresses that are specified in another vector register.
- **New instructions:** fused multiply-add, and the standard logical operations, including instructions to extract non-byte-aligned
- Data can be replicated from L2 cache directly
- Free numeric type conversion and data replication while reading from memory
- Mask registers: predicated
- 3 source operands, one of them can come from L1 directly



# Gather and Scatter Operation Support

- Loads and stores from non-continuous addresses
- 16 data values can be loaded or stored from addresses in another vector register value.

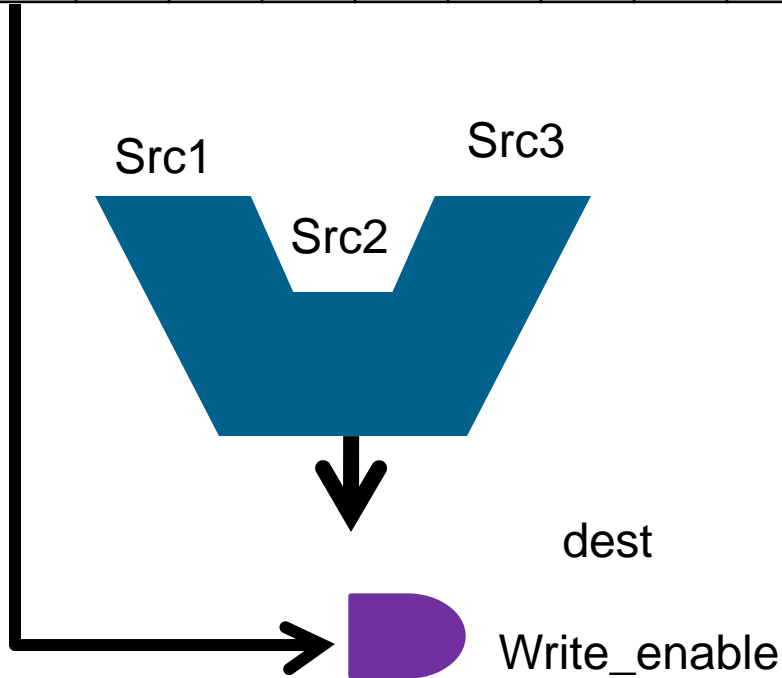




# Mask Bits



1	1	1	0	0	1	1	1	1	1	1	1	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

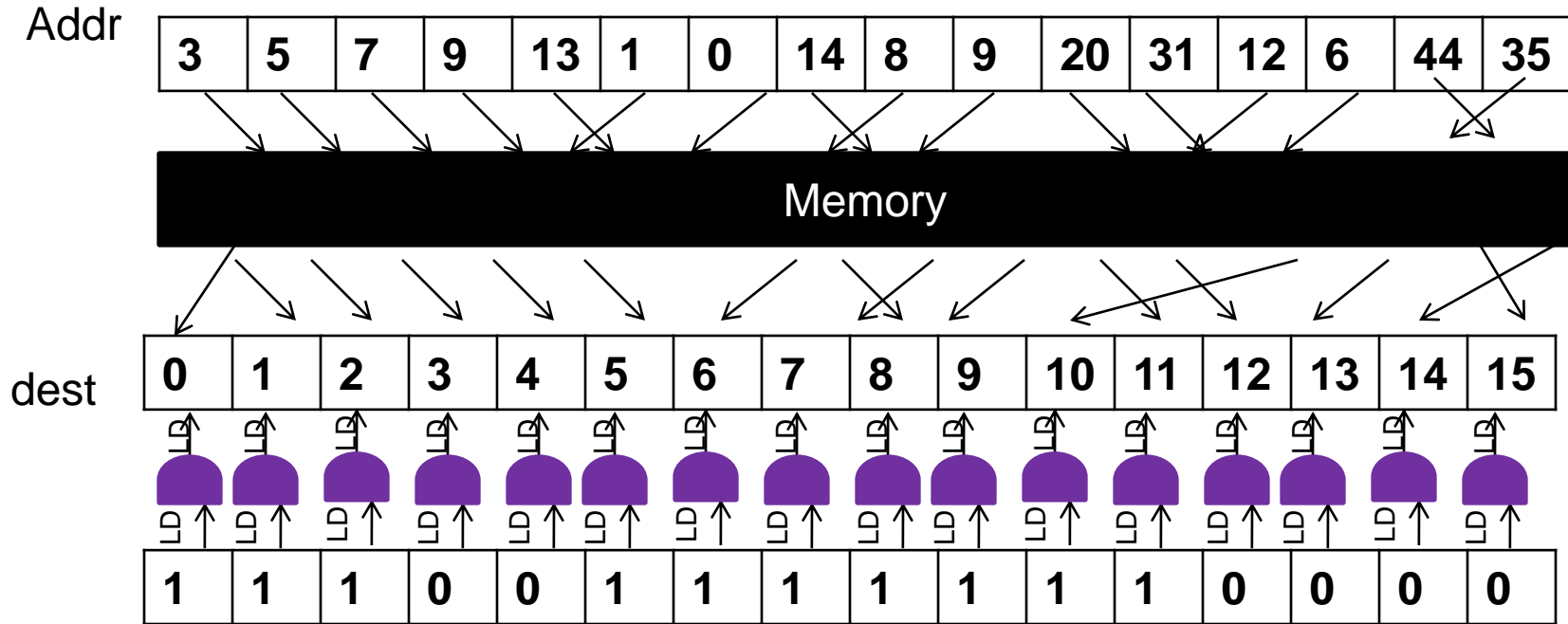


- All the operations participated the computation in vector units
- Mask bits decide write enable signal





# Predication for Load/Store

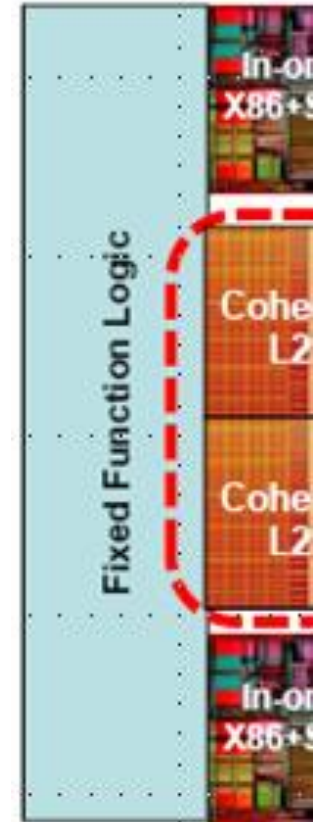


MASK\_bit



# Fixed Functions

- Use FIFO for load balancing
- No rasterizations
- Texture filtering
  - 32KB texture cache per each core
  - Core passes commands through L2 cache
- Texture units perform **virtual to physical page** translation



# Larrabee Data Parallelism: Threads

**Cores: Each runs multiple threads**

**Thread: HW-managed context (hide short unpredictable latencies)**

**Fiber: SW-managed context (hides long predictable latencies)**

Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand

16-wide vector unit

...

Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand  
Strand

16-wide vector unit

...

**More Fibers running (typically 2-10, depending on latency to cover)**

...

**More Threads (up to 4 per core, share memory via L1 & L2 caches)**



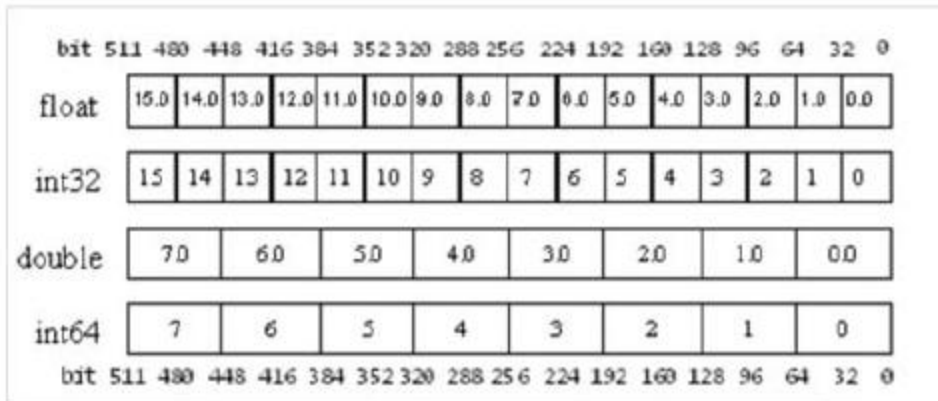
# Data Parallelism Support

- Implementation hierarchy (your names may vary)
  - Strand: runs a data kernel in one (or more) VPU lane(s)
  - Fiber: SW-managed group of strands, like co-routines
  - Thread: HW-managed, swaps fibers to cover long latencies
- Core: runs multiple threads to cover short latencies
  
- Comparison to GPU data parallelism
  - Same mechanisms as used in GPUs, except...
  - Larrabee allows SW scheduling (except for HW threads)



# LRB new instruction sets

- New Data types



- Vector arithmetic, logical, and
- Vector mask generation
- Vector load/store

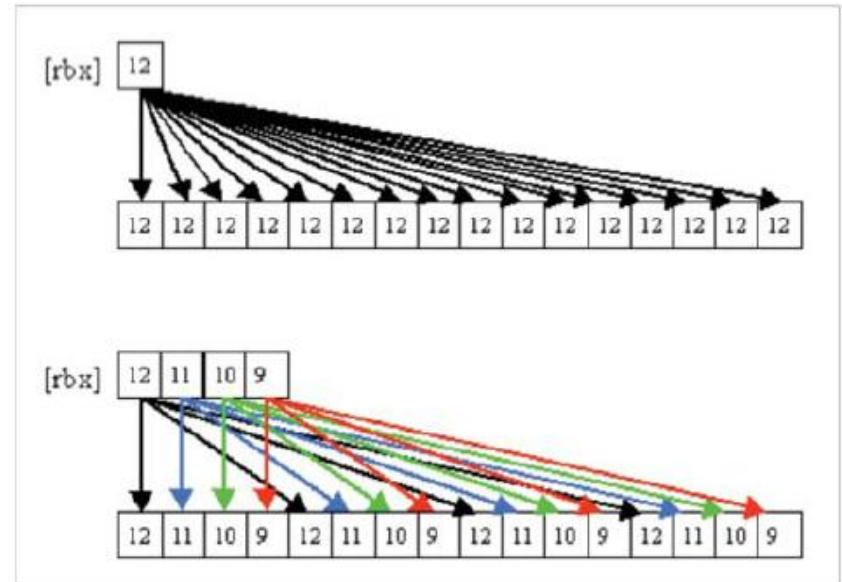
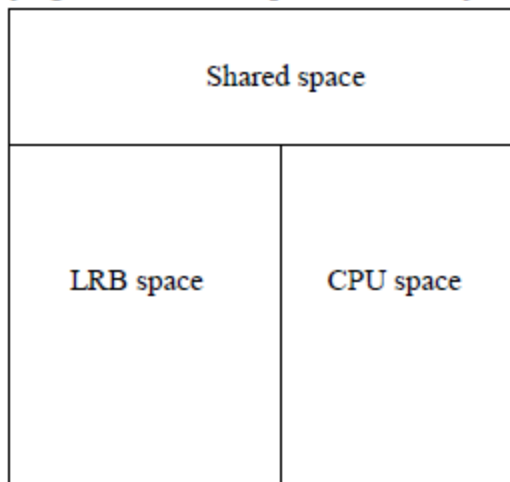


Figure 3: 1-to-16 [1to16] and 4-to-16 [4to16] broadcasts.



# LRB Programming Models

- Stream data handles
  - Cache control instructions
    - Prefetch into L1 or L2 caches
    - Early evictions
- CPU/GPU memory space







# Project

- Team members (1-2)
- Schedule
  - 4/12 (1 paragraph project proposal)
  - 4/14 proposal 1<sup>st</sup> feedback
  - 4/16 detailed proposal description meeting
  - 4/23 progress meeting
  - 4/28 final project presentation
    - 10 min for each team
  - 4/30 project submission



# Suggested Topics

- Programs
  - Game related programs
  - CUDA, Nintendo DS
- Architecture
  - Architecture survey
  - Hardware architecture simple models