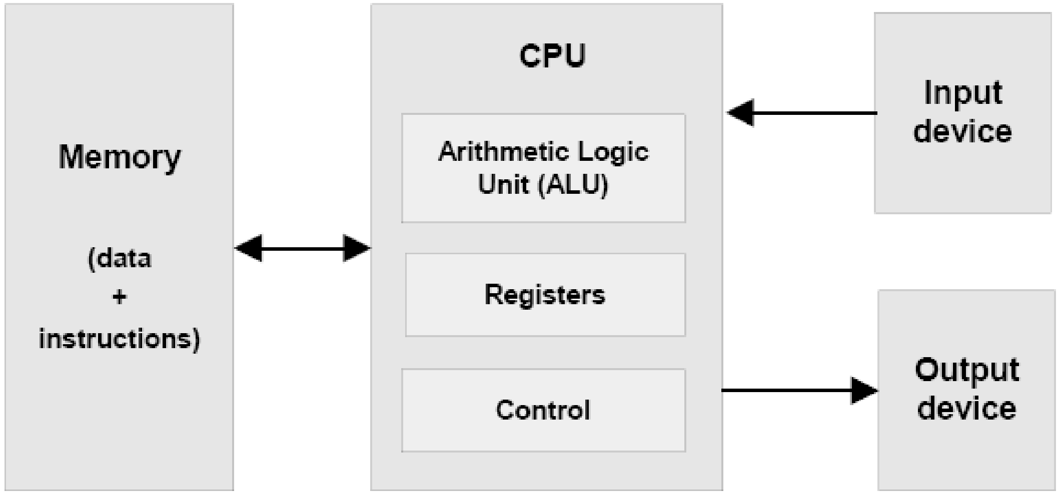
Streaming Processor

Computing Architectures

- General Central Processing Unit (CPU)
- Single Instruction Multiple Data (SIMD)
- Vector processors
- Stream processors

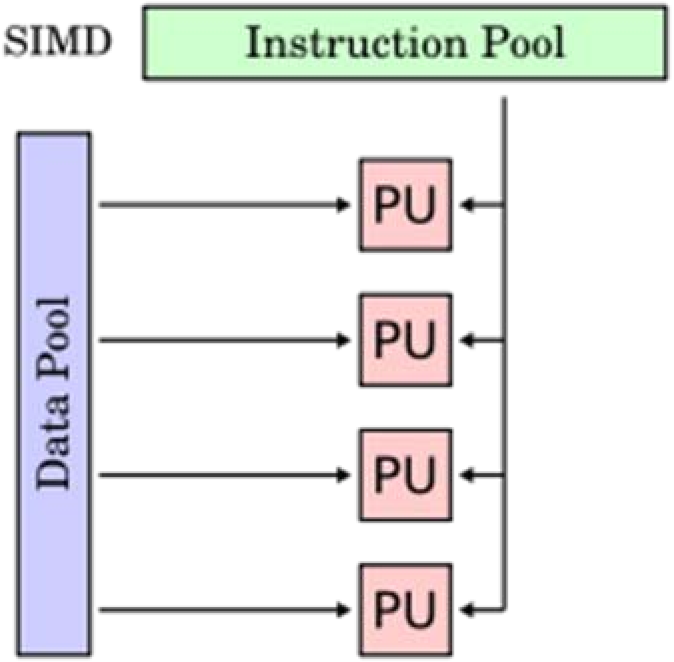
CPU



- Von Neumann processor
- Performance issue
  - ALUs faster than data communication
    - Between memory and ALU
  - Management and communication cost exceeds pure computation cost

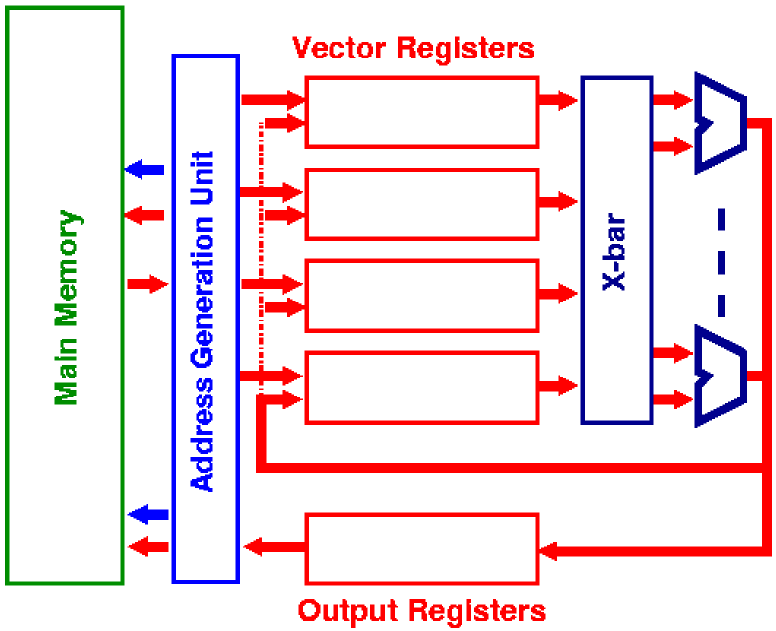
SIMD

- Single Instruction Multiple Data
  - a single set of instructions is executed by multiple processors
  - multiple data streams are processed at multiple processing units
  - Intel MMX,SSE
  - AMD 3D Now!
  - …



Vector processors

- Vector data for math computation
- Load vectors with a single instruction
- Operate on multiple data elements simultaneously
- Cray supercomputers



Why Use Stream Processors

- Fast computing by today's VLSI technology
  - thousands of arithmetic logic units operating at multiple GHz on $1cm^2$ die
- Communication and control are bottleneck
  - instructions and data management
- Example:
  - only 6.5% of the Intel Itanium die is devoted to its 12 integer and 2 floating point ALUs and their registers
  - remainder is used for communication, control, and storage

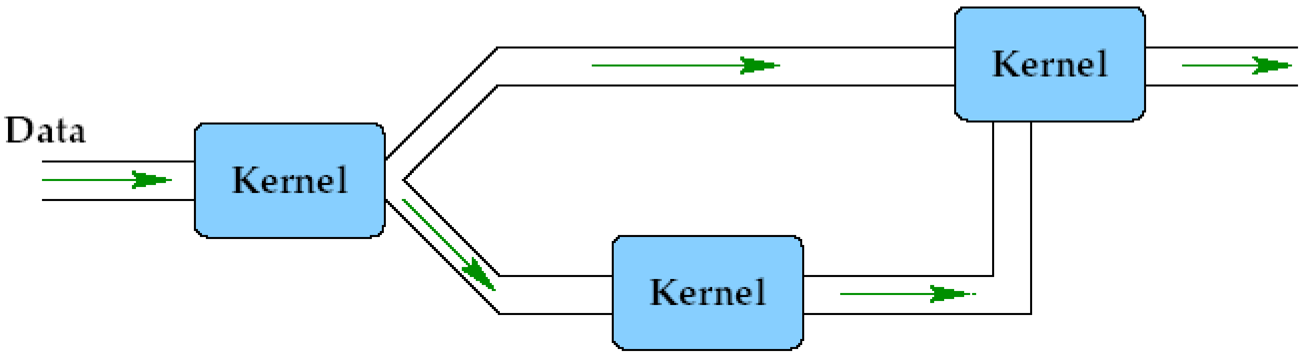
Why Use Stream Processors

- This is a drawback for general-purpose CPU
- On a GPU, such as a Nvidia GeForce4
  - Hundreds of ALUs
  - Efficient control and communication
  - Special purpose for media-processing (such as graphics pipelines)
  - Exposes abundant parallelism with little global communication and storage

Stream Processors

- Design such processors:
  - Expose these patterns of communication, control, and parallelism to more applications
  - Create a general purpose streaming architecture without compromising its advantages
- Existing implementations that come close
  - Nvidia FX, ATI Radeon GPUs …
  - enable GP-GPU (general purpose streaming, GP-GPU)

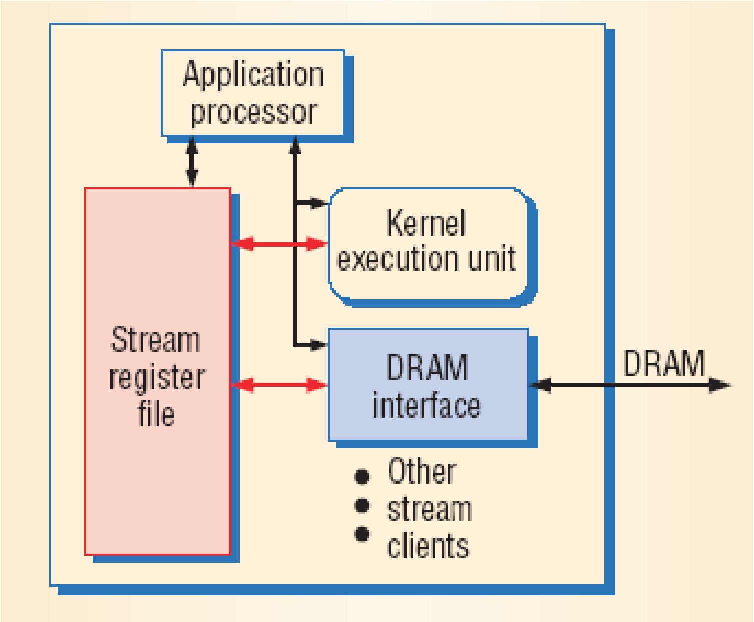
Stream Processing



- Organize an application into streams and kernels
  - inherent locality and concurrency
  - media-processing applications
- This creates the programming model for stream processors

Memory Hierarchy

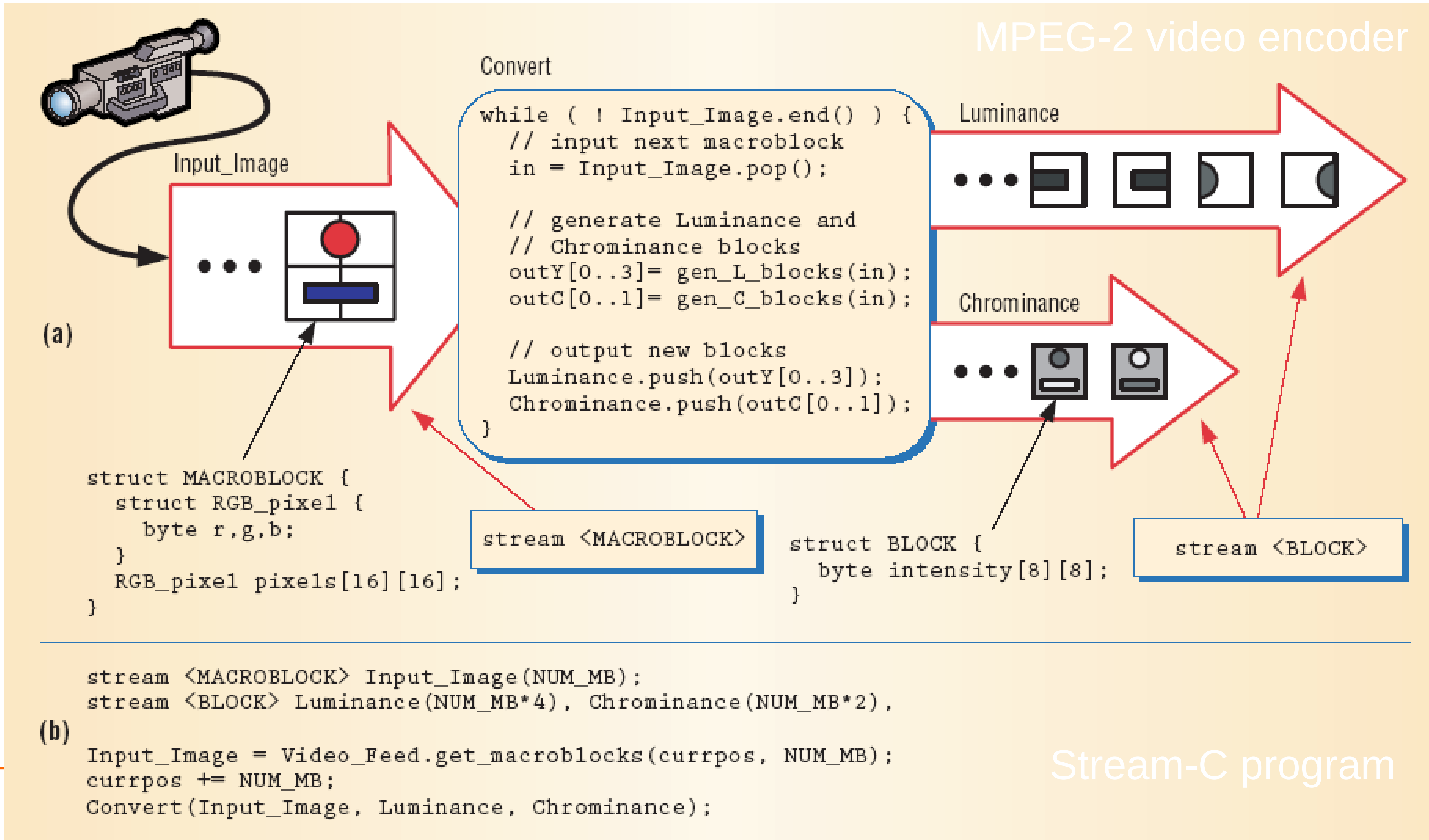
- Local register files (LRFs)
  - use for operands for arithmetic operations (similar to caches on CPUs)
  - exploit fine-grain locality
- Stream register files (SRFs)
  - capture coarse-grain locality
  - efficiently transfer data to and from the LRFs
- Off-chip memory
  - store global data
  - only use when necessary



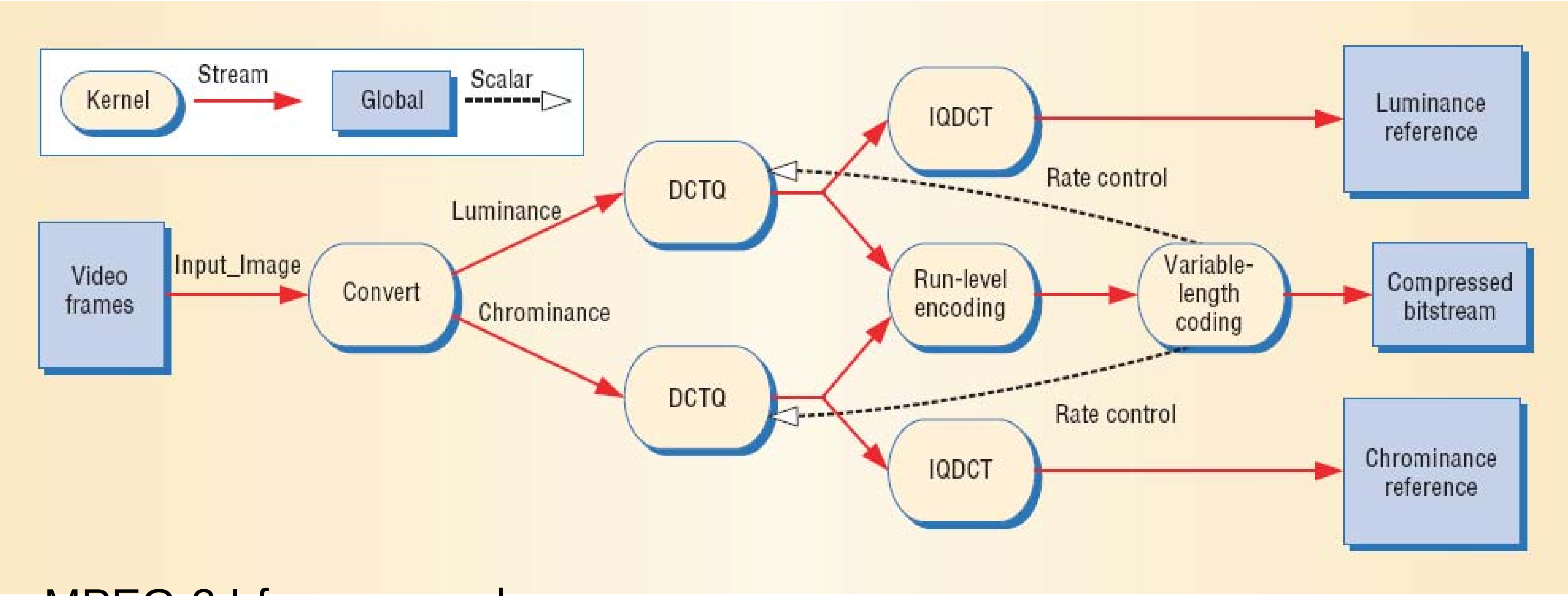
Bandwidth Hierarchy

- Together with the memory hierarchy
  - roughly an order of magnitude for each level
- From today's VLSI technology
- With the locality of operations within LRFs, hundreds of ALUs operate at peak rate

Streams and kernel for MPEG2 encoder



MPEG2 Encoder

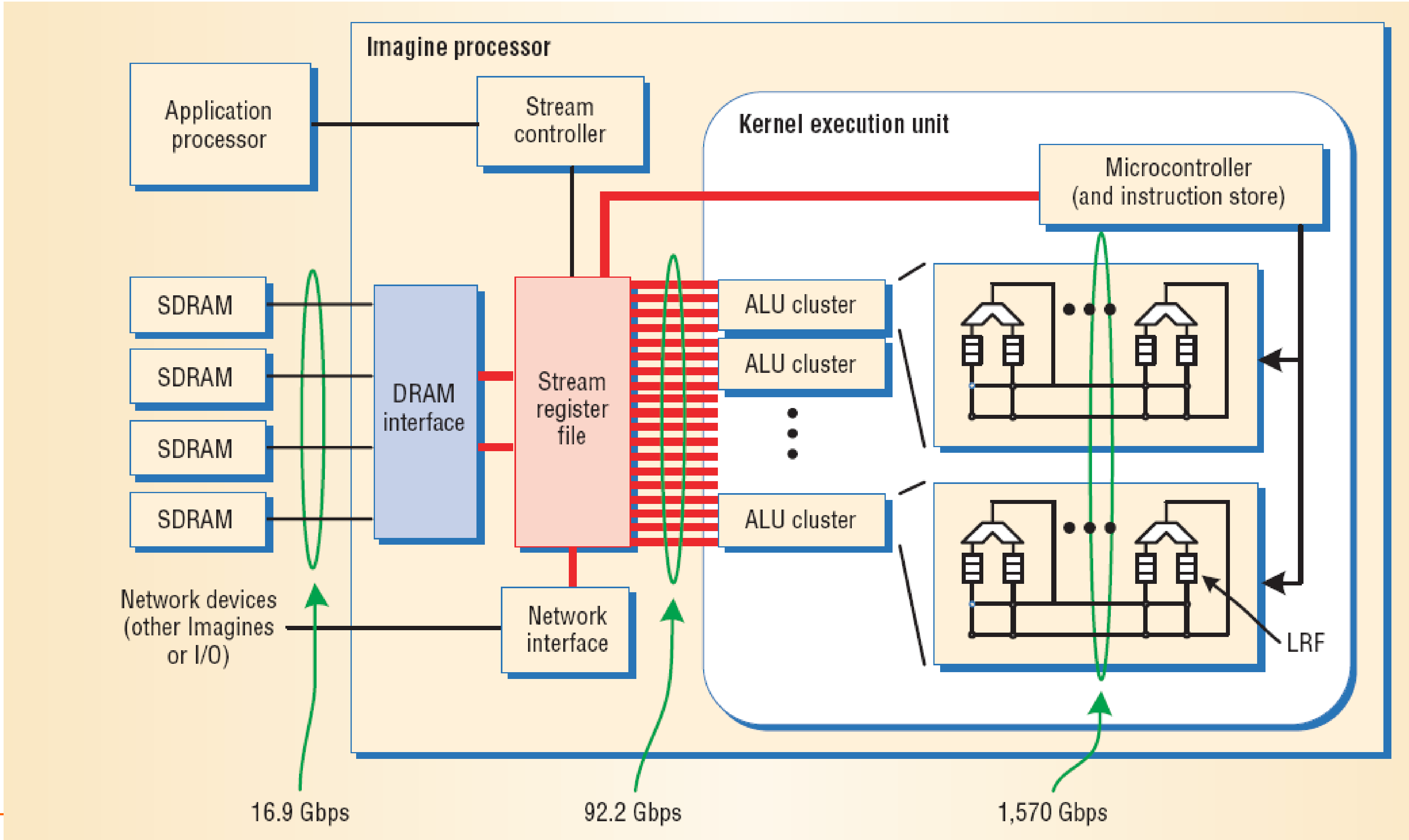


MPEG-2 I-frame encoder

Q: Quantization, IQ: Inverse Quantization, DCT: Discrete Cosine Transform

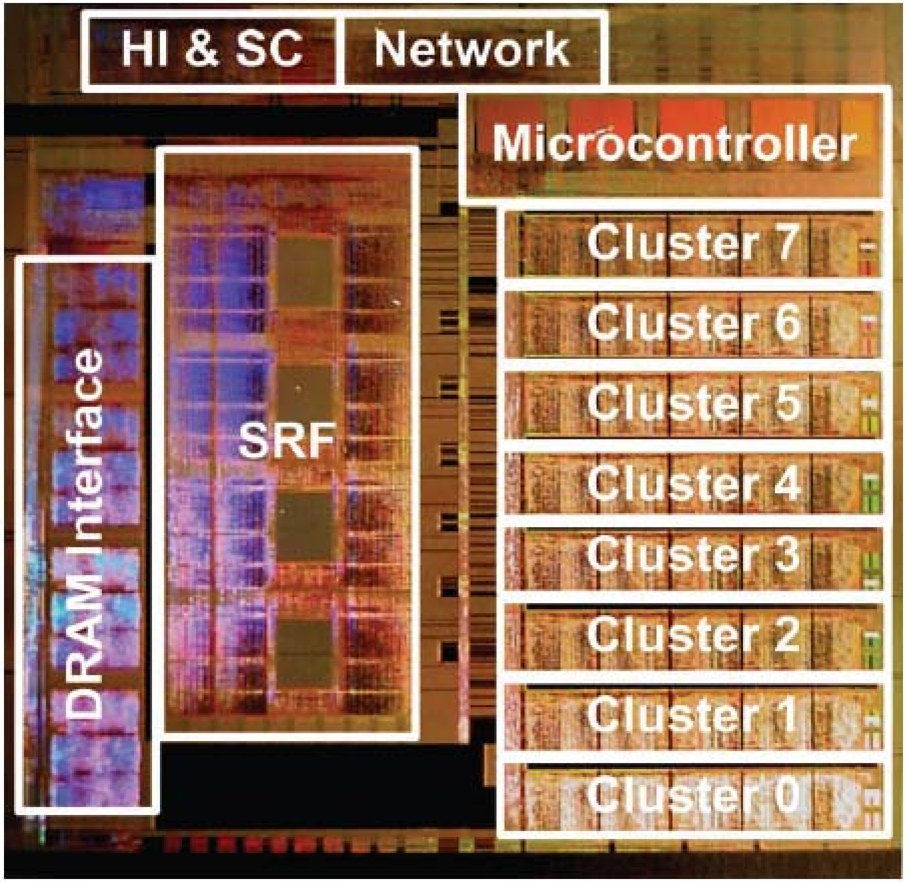
Global communication (from RAM) needed for the reference frames

Stanford's Imagine Processor

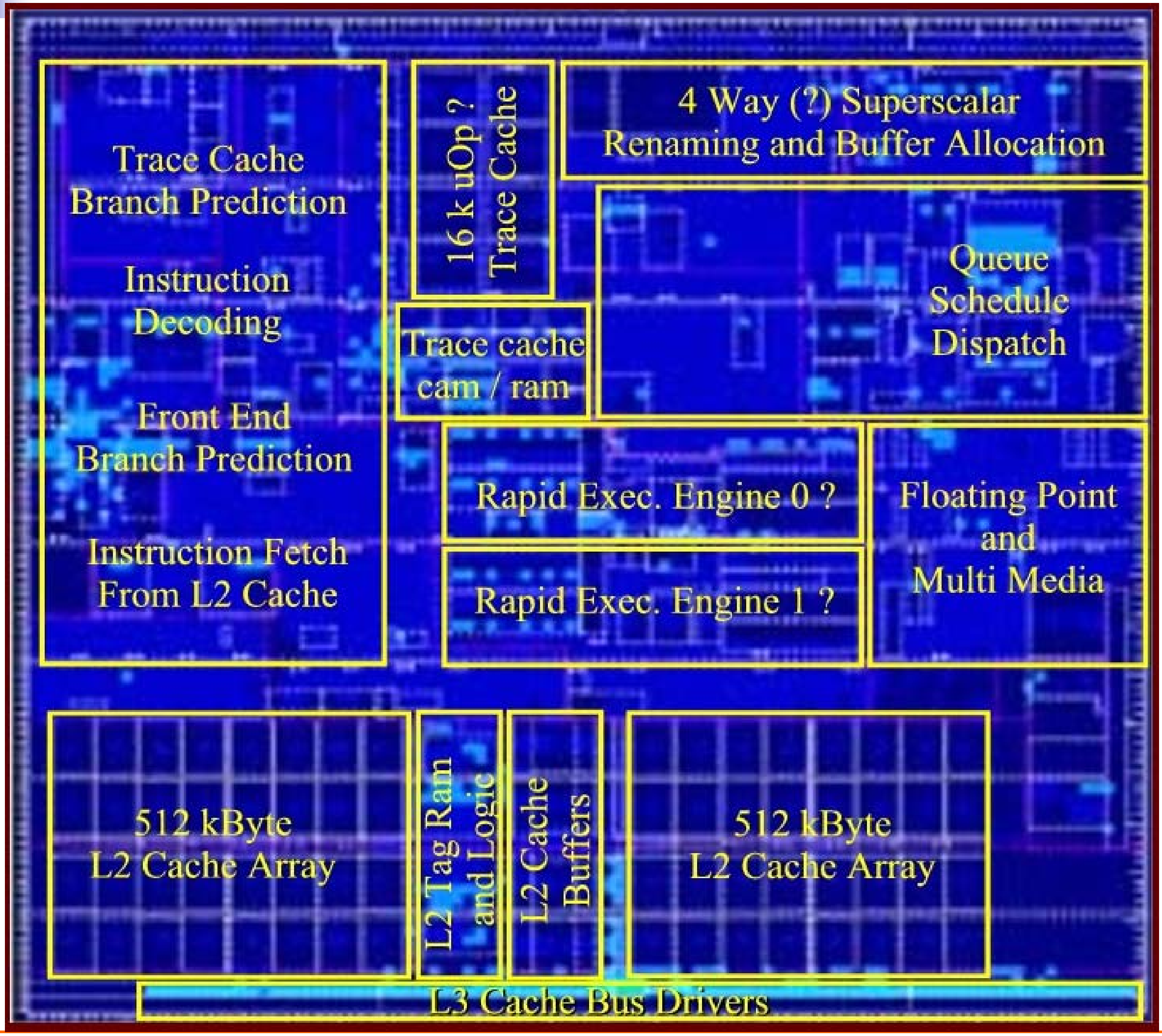


KENT STATE UNIVERSITY

Imagine's die layout



Intel’s Prescott die layout



Parallelism in Stream Processor

- Instruction-level
  - exploit parallelism in the scalar operations within a kernel
  - for example, gen_L_blocks, gen_C_blocks can occur in parallel
- Data-level
  - operate on several data items within a stream in parallel
  - for example, different blocks can be converted simultaneously
- Task parallelism
  - Multiple tasks runs concurrently
  - for example, the two DCTQ kernels could run in parallel

GPUs as Stream Processors

- Stream data elements
  - Points or vertices in vertex processing
  - fragments, essentially pixels in fragment processing
- Kernels
  - vertex and fragment shaders (computing unit)
- Memory
  - texture memory (SRFs)
  - not-exposed LRF
  - bandwidth to RAM (AGP and PCI-Express)

GPUs

- Data parallelism
  - fragments and points are processed in parallel
- Task parallelism
  - fragment and vertex shaders work in parallel
  - data transfer from RAM can be overlapped with computation

References

- U. Kapasi, S. Rixner, W. Dally et al. “Programmable stream processors,” IEEE Computer August 2003
- S.Venkatasubramanian, “The graphics card as a stream computer,“ SIGMOD DIMACS, 2003