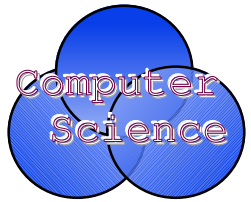
Asynchronous SIMD



Charles C. Weems

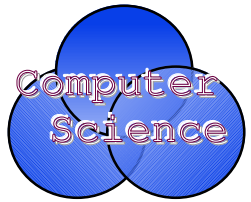
Computer Science Department

University of Massachusetts

Amherst, MA 01003-4610

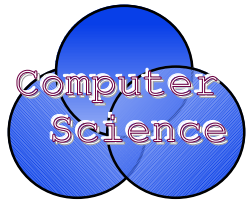
weems@cs.umass.edu

SIMD is Attractive



- Simple programming model
  - No multitasking or race conditions
- Delivers high performance
  - Unbeatable for dense fine-grained computation
- Scalable
  - Simple processors, easy to gang up into millions
- So why has it failed?

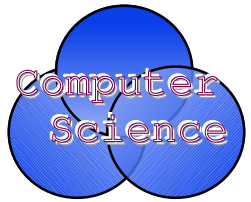
Scales Up But Not Down



Cost Issues

- Need many processor chips in a system
  - Individual chips give no boost in performance
  - Designed to work in groups of more than 16
  - Need to amortize cost of central controller
- Results in high entry-level system cost
  - No viable mass market
  - Can't recover development cost or follow technology curve

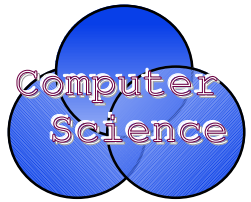
Synchronous



Performance Issues

- Must control clock skew across many chips
  - Limits clock rate
- Central control performs ~20 ops to deliver each instruction
  - Throttles issue rate
- Inter-chip communication
  - Requires high percentage of pin bandwidth
  - Intolerant of latency

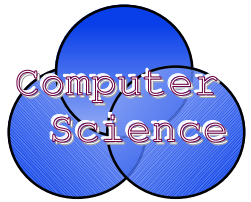
Inefficient



Cost/Performance Issues

- Never large enough
  - Virtualization adds unacceptable overhead
- Rarely the right size
  - Usually waste a high percentage of processors
- Poor memory bandwidth utilization
  - Bandwidth limited at low issue rates
  - Inactive processors still consume bandwidth

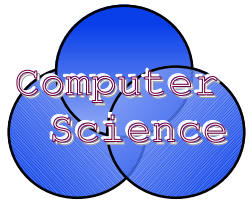
Inflexible



Usability Issues

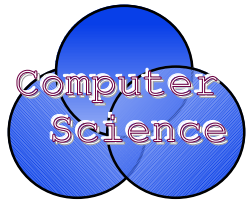
- One monolithic resource
  - High context-switch cost
  - Hard to partition, hard to share
- Custom interface via dedicated host
  - Not integral to user system
  - I/O limited (except for dedicated devices)
- Proprietary programming languages

Alternative Approach



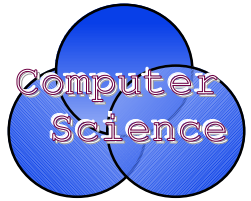
- Design a SIMD processor to scale *down*
- Accept the implied high virtualization factor
  - Treat it as an opportunity
- Internal virtual processor tiles provide
  - High communication latency tolerance and
  - Lower bandwidth requirement
  - Predictable access pattern in data memory
  - Can devote more bandwidth to memory
  - High degree of I-cache temporal locality

Embracing Virtualization



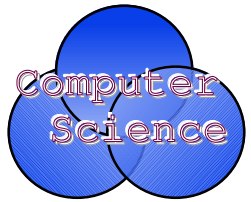
- More virtualization = better utilization
  - Oversize problems welcome!
  - Fewer unused processors
- Asynchronous communication
  - No need to synchronize clocks
- On-chip controller
  - Avoids costly separate device
  - Opens throttle on issue rate

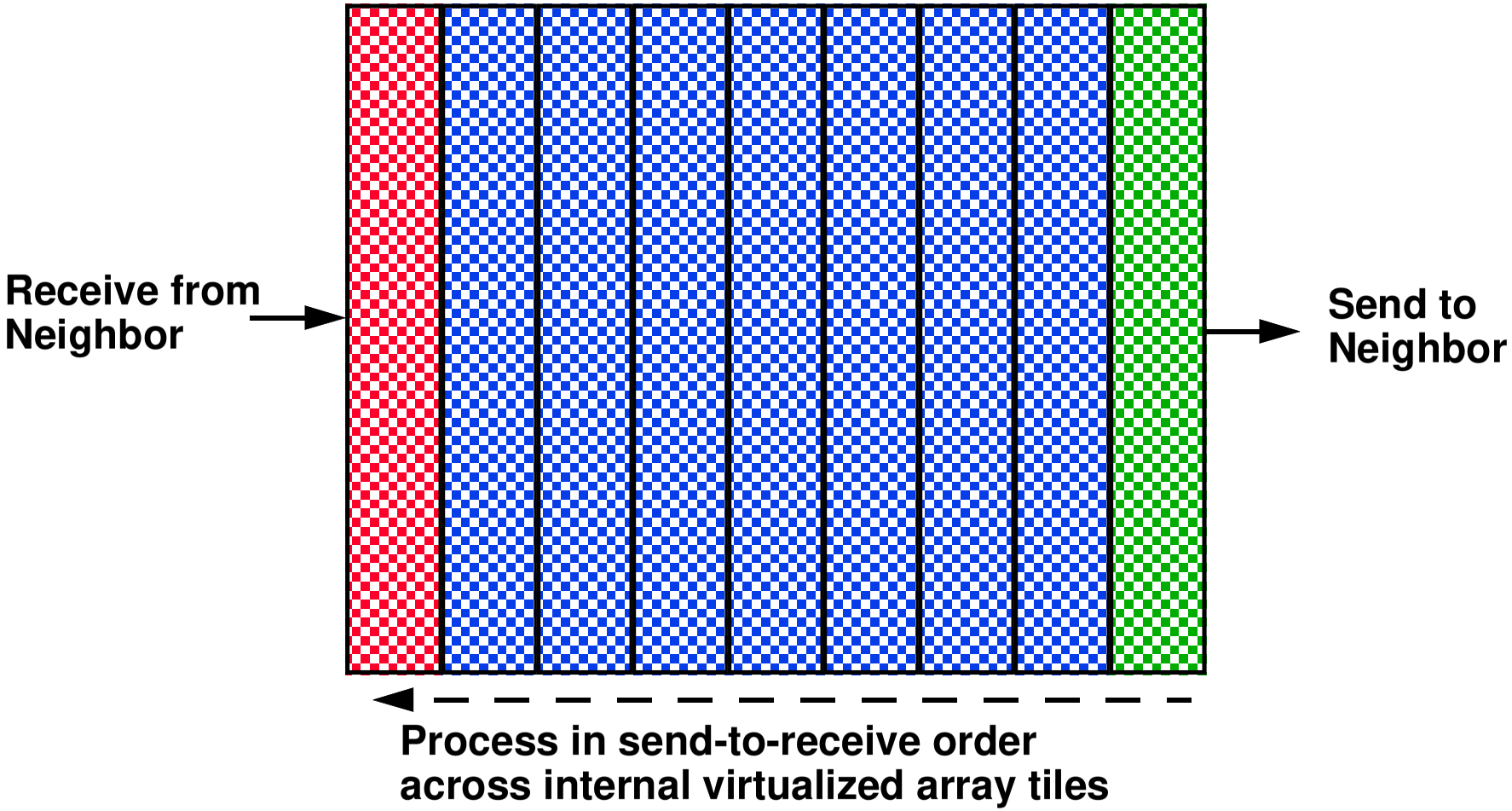
Asynchronous SIMD



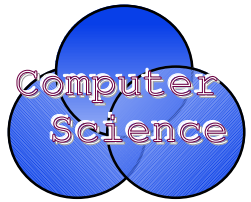
- High degree of fine-grained parallelism
- High communication latency tolerance
- Simple programming model like SIMD
- Flexible support for multitasking environments
- High clock rate (microprocessor speeds)
- High single-chip performance with scalability

Exploiting Virtualization



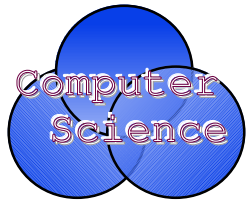


Implications



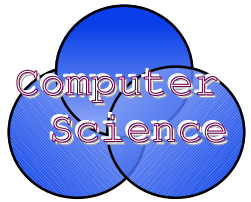
- Most access is to local memory
  - Predictable well in advance; regular streams
- Instruction blocks repeated many times
  - Can be cached predecoded
  - Can be prebroadcast by host
- Communication is asynchronous
  - Low bandwidth requirement, latency tolerant
  - Can use existing standard mechanisms

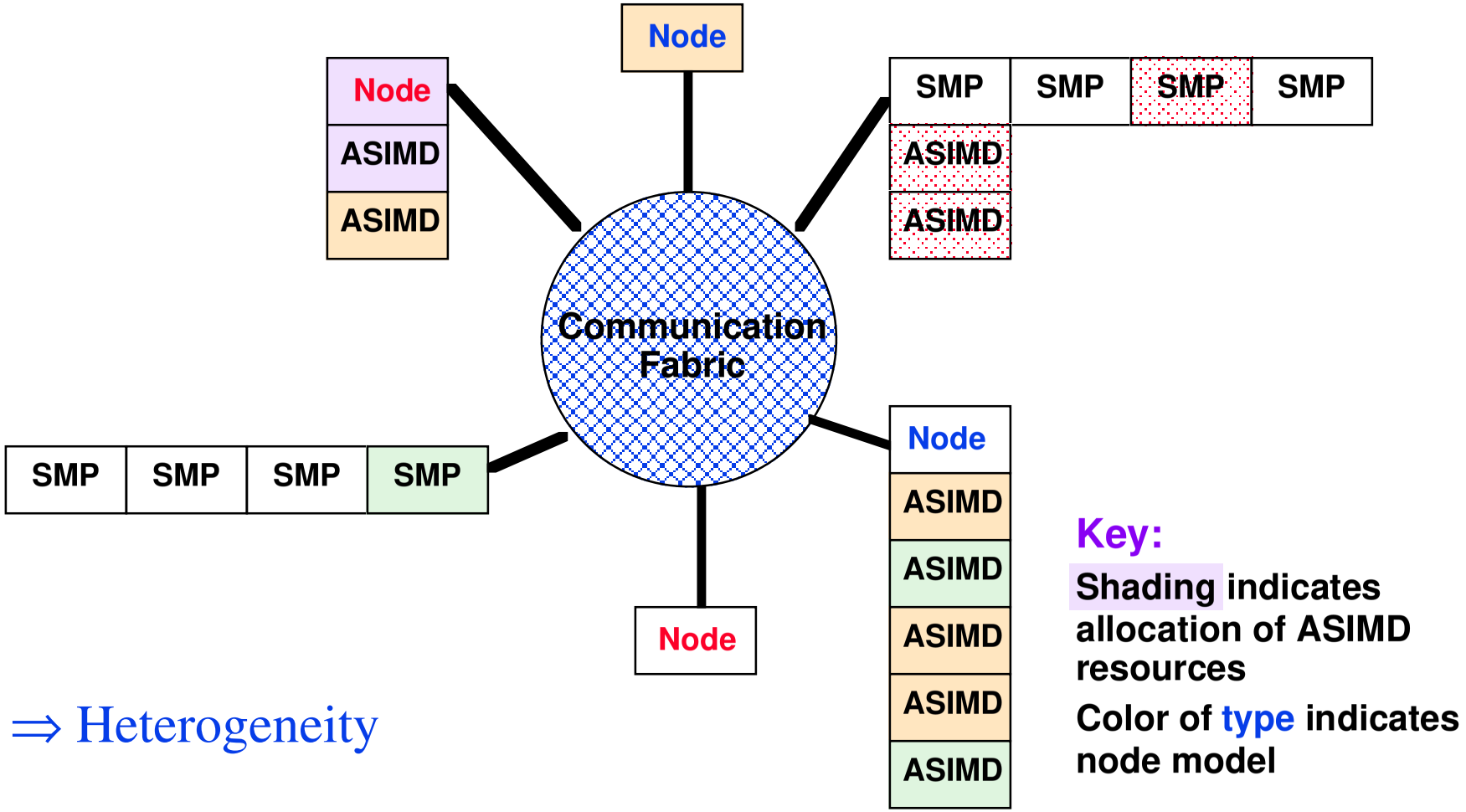
More Implications



- No global clock -- high local clock rate
- Powerful single chip -- e.g., PC accelerator
- Scales up to many chips -- supercomputing
  - As long as virtualization ratio remains high
- Multichip system can be partitioned
  - Allocable system resource, multiple hosts
  - Low context switch cost for time sharing

Example Configuration





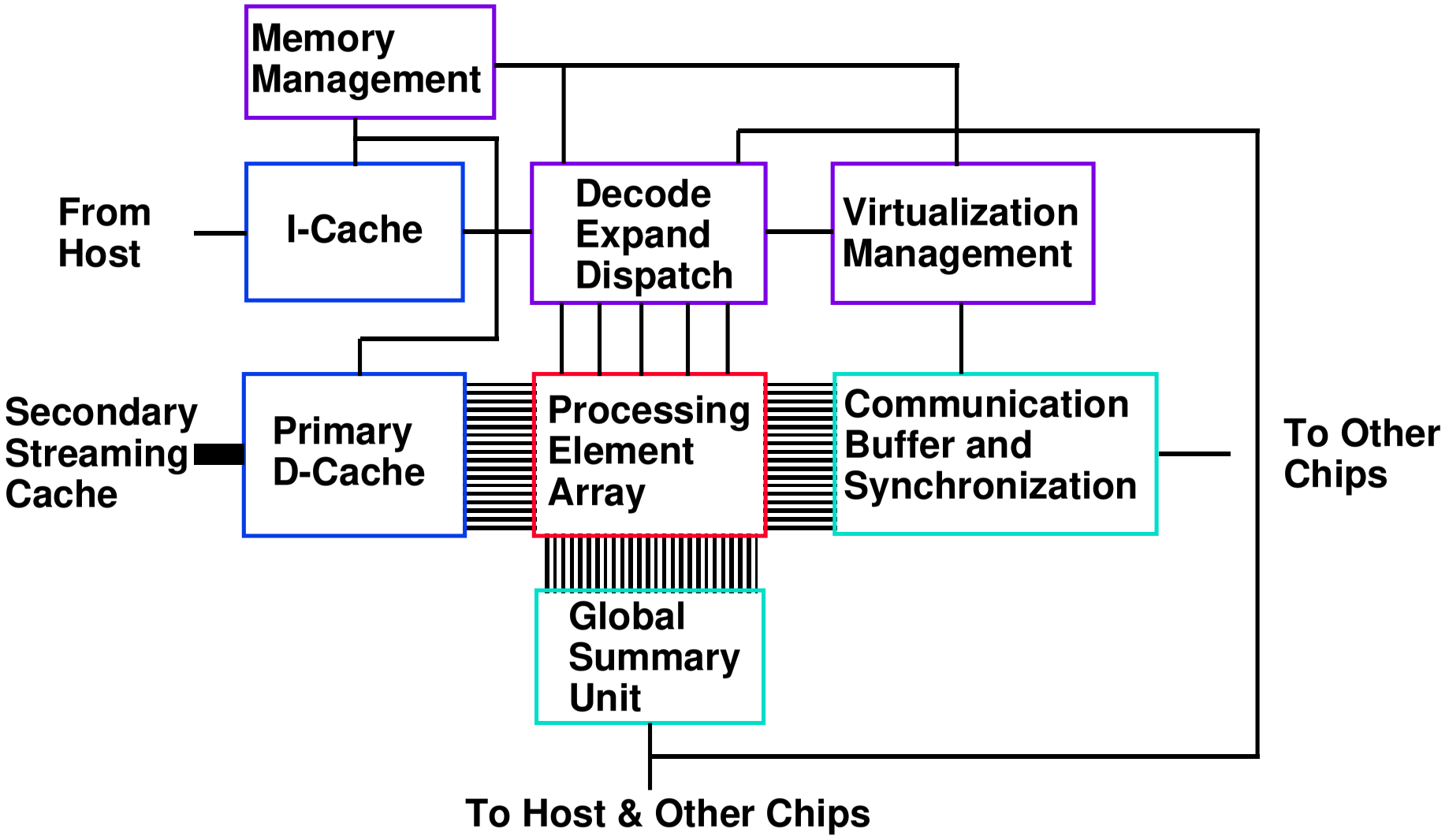
⇒ Heterogeneity

Key:

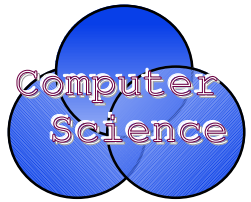
Shading indicates allocation of ASIMD resources

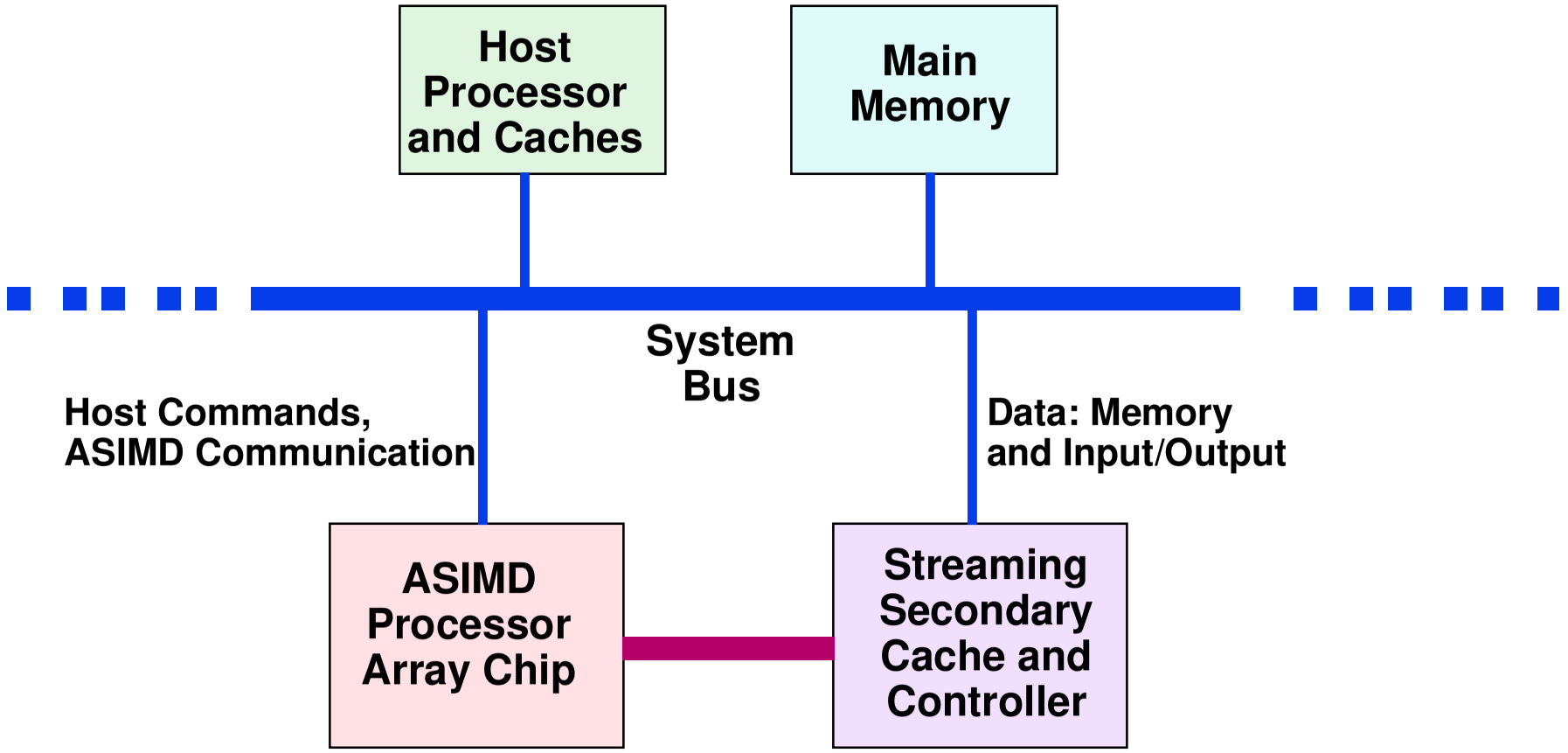
Color of type indicates node model

Chip Architecture

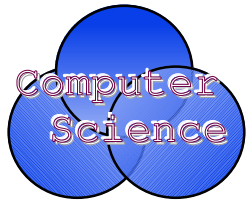


System Architecture





Summary



- ASIMD solves problems of traditional SIMD
  - High performance, low cost, highly usable
- Retains simple programming model
- Delivers flexible, scalable high performance
- Suitable for mass markets
- Plays well with others (PCs to Supercomputers)