Memory Bandwidth and System Balance in HPC Systems: 2021 Update

John D. McCalpin, PhD

mccalpin@tacc.utexas.edu

Outline

1. Changes in TOP500 Systems
   - System Architectures
   - Programming Models
   - System Sizes
2. Technology Trends & System Balances
   - Computation Rates vs Data Motion Latency and Bandwidth
   - Required Concurrency to Exploit available Bandwidths

Changes in TOP500 Systems

Part 1

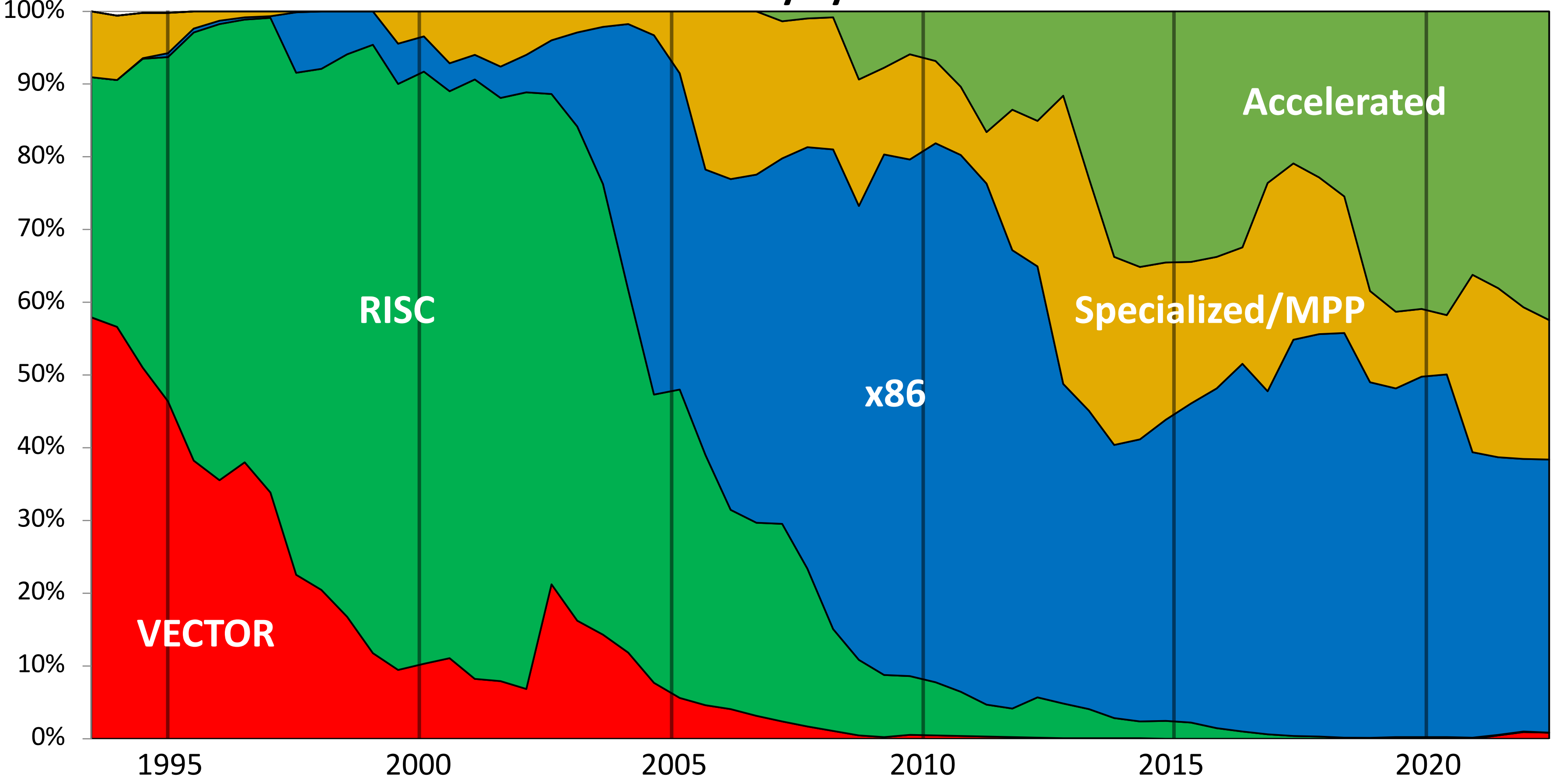
What is the TOP500 list?

- List of 500 fastest systems on ”High Performance Linpack” benchmark
- Only reported systems (e.g., missing “Blue Waters” at NCSA)
- Benchmark only needs to be run on **one** instance of any configuration
- System does not need to be used for HPC
  - Early 2000’s: lots of commercial/database clusters
  - Recently: lots of cloud systems (web services)
- Customers (including vendors) choose how to submit their results
  - 1 big system or more smaller systems
  - Heterogeneous vs homogeneous – also depends on available software

What is the TOP500 list?

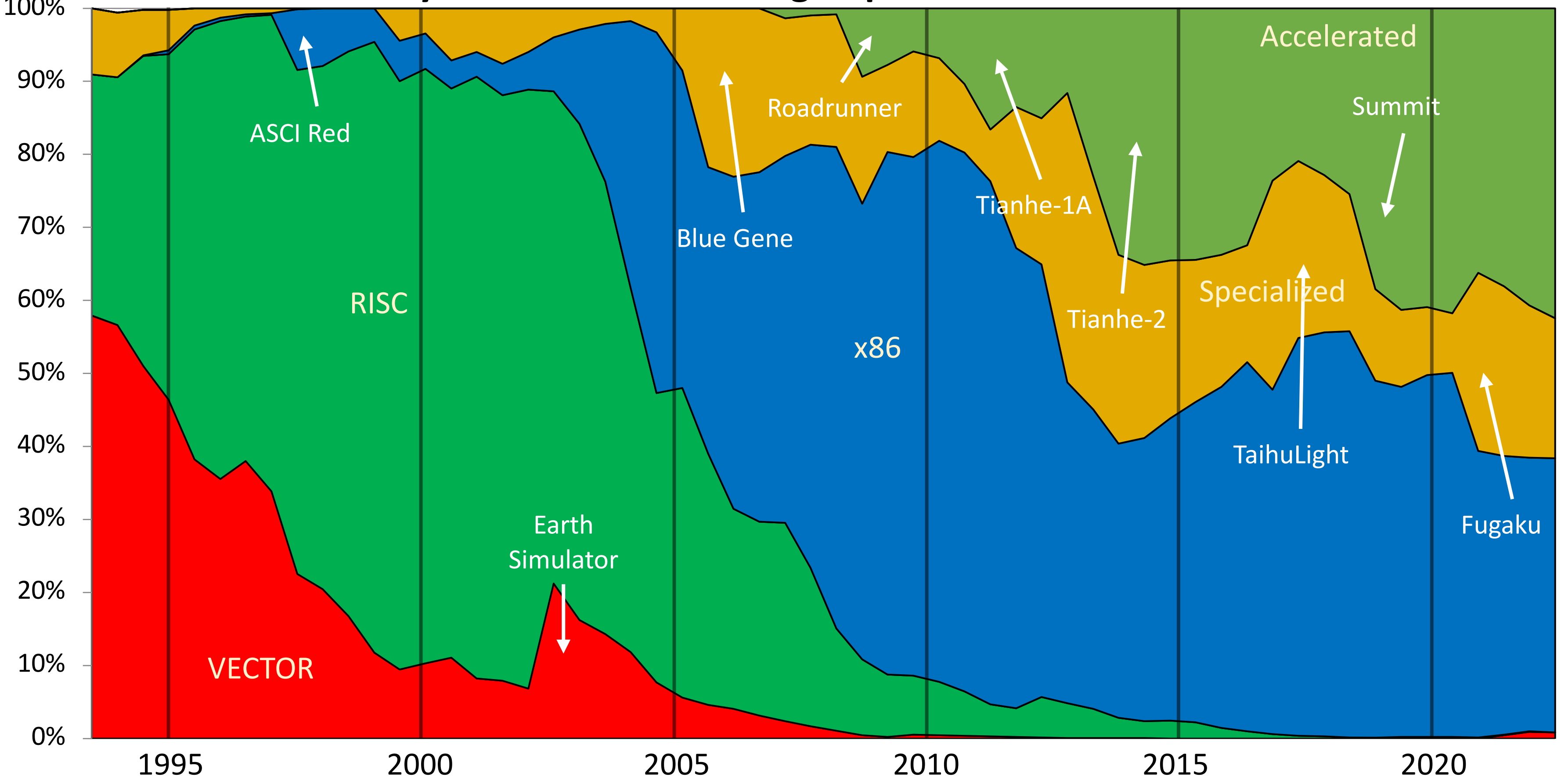
- Database
    - XML (without system configuration data)
    - Excel (15 different combinations of column headers)
        - Information about system configurations is minimal/misleading/incorrect
- Hundreds of hours spent
    - Fixing errors
    - Looking up system parameters
    - Reverse-engineering system configurations
- More systematic re-analysis in progress

TOP500 RMax Contributions by System Architecture



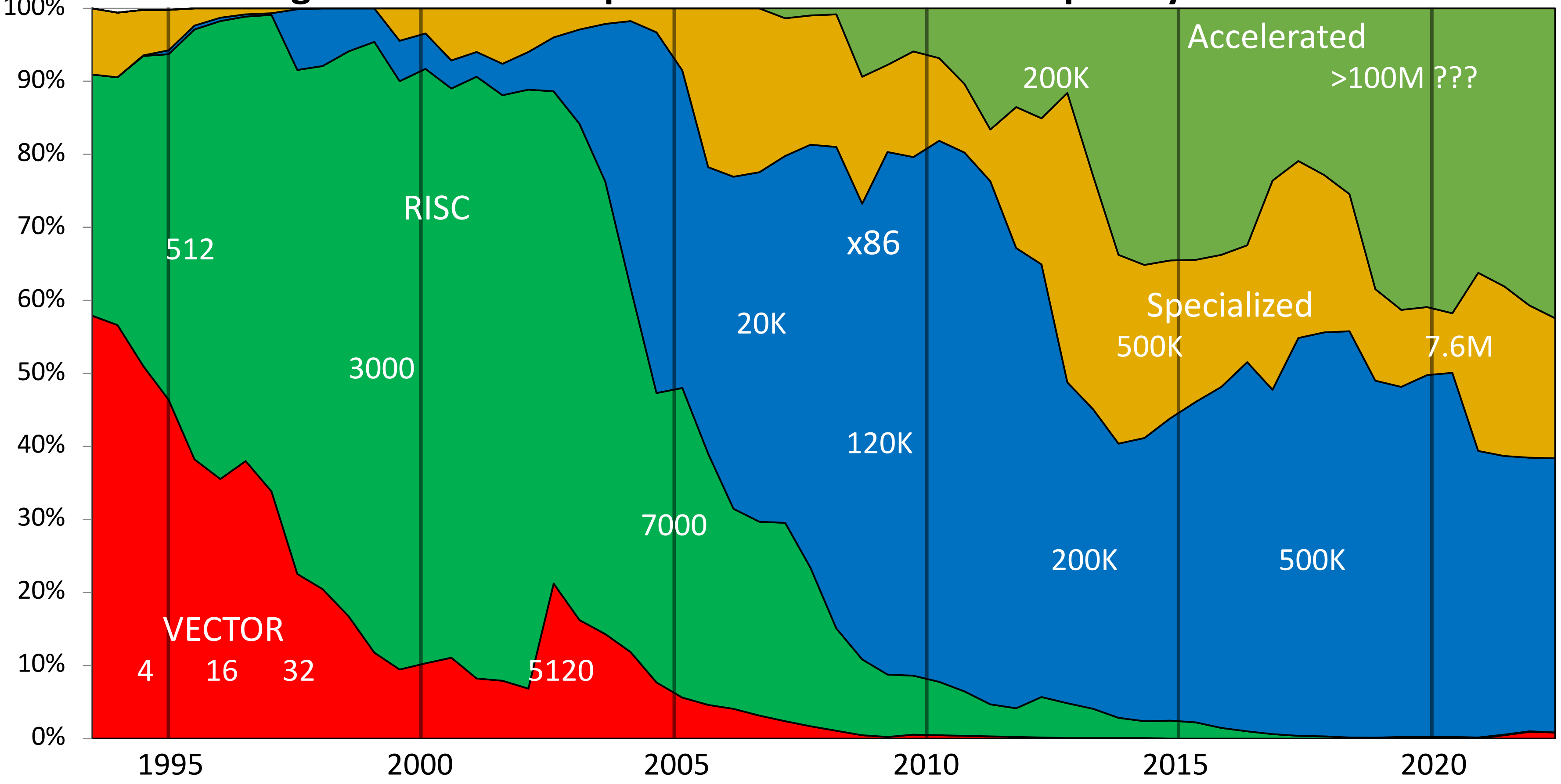
Note: a portion of the "Accelerated" systems Rmax is from x86 and other microprocessors

New #1 System often makes big impact on distribution

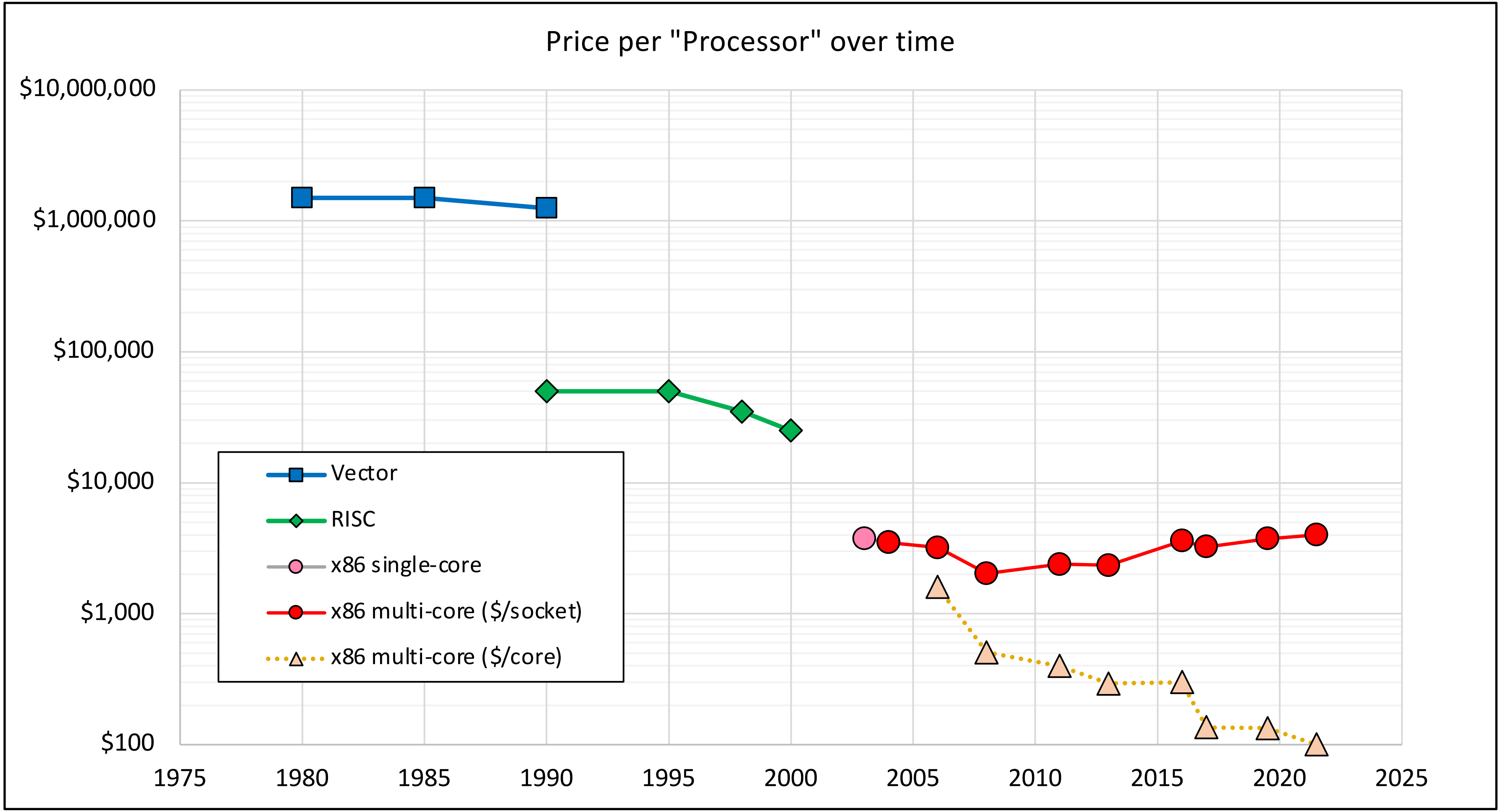


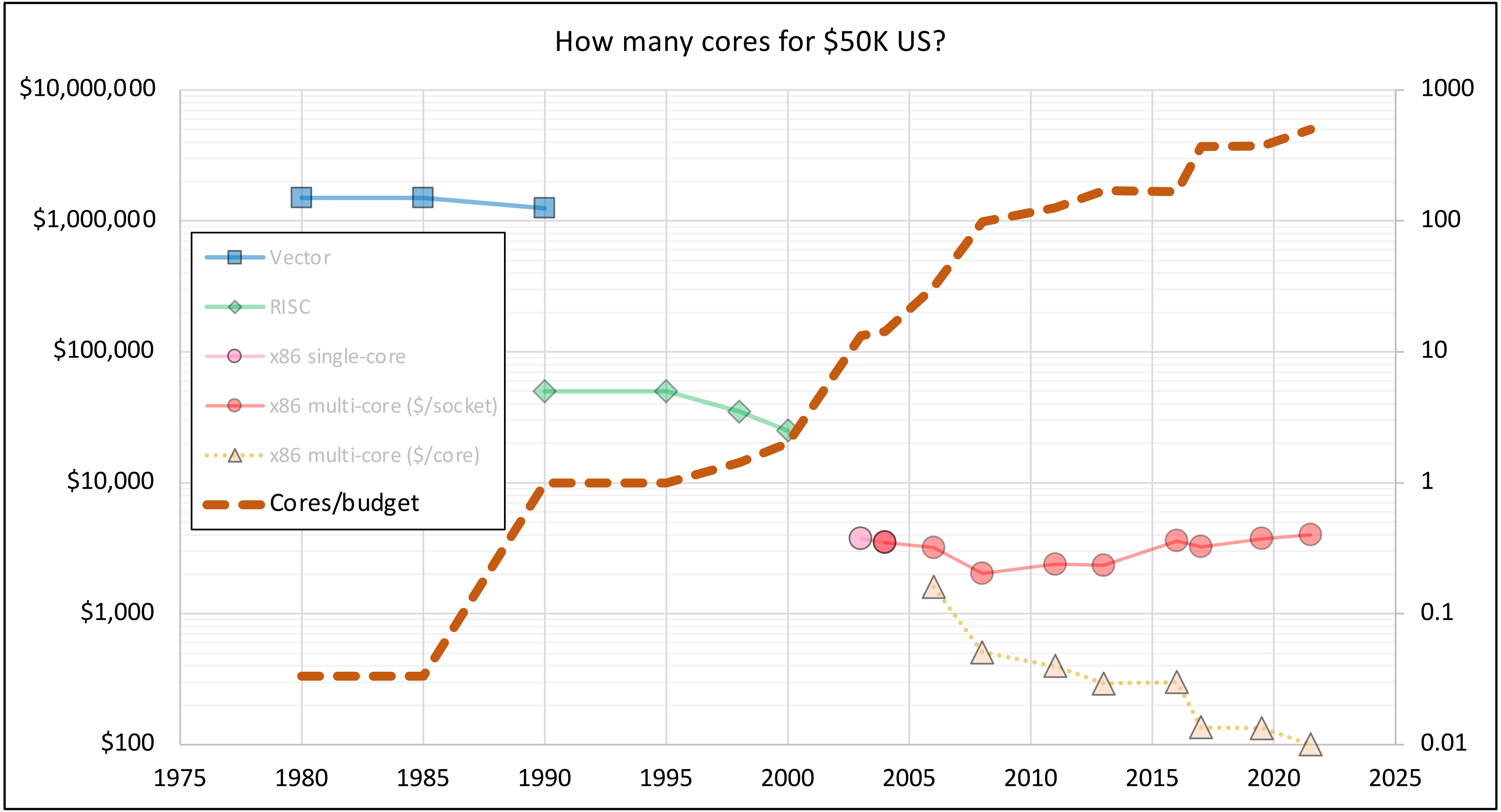
THE UNIVERSITY OF TEXAS AT AUSTIN
TEXAS ADVANCED COMPUTING CENTER

Huge increases in required thread counts for top 10 systems

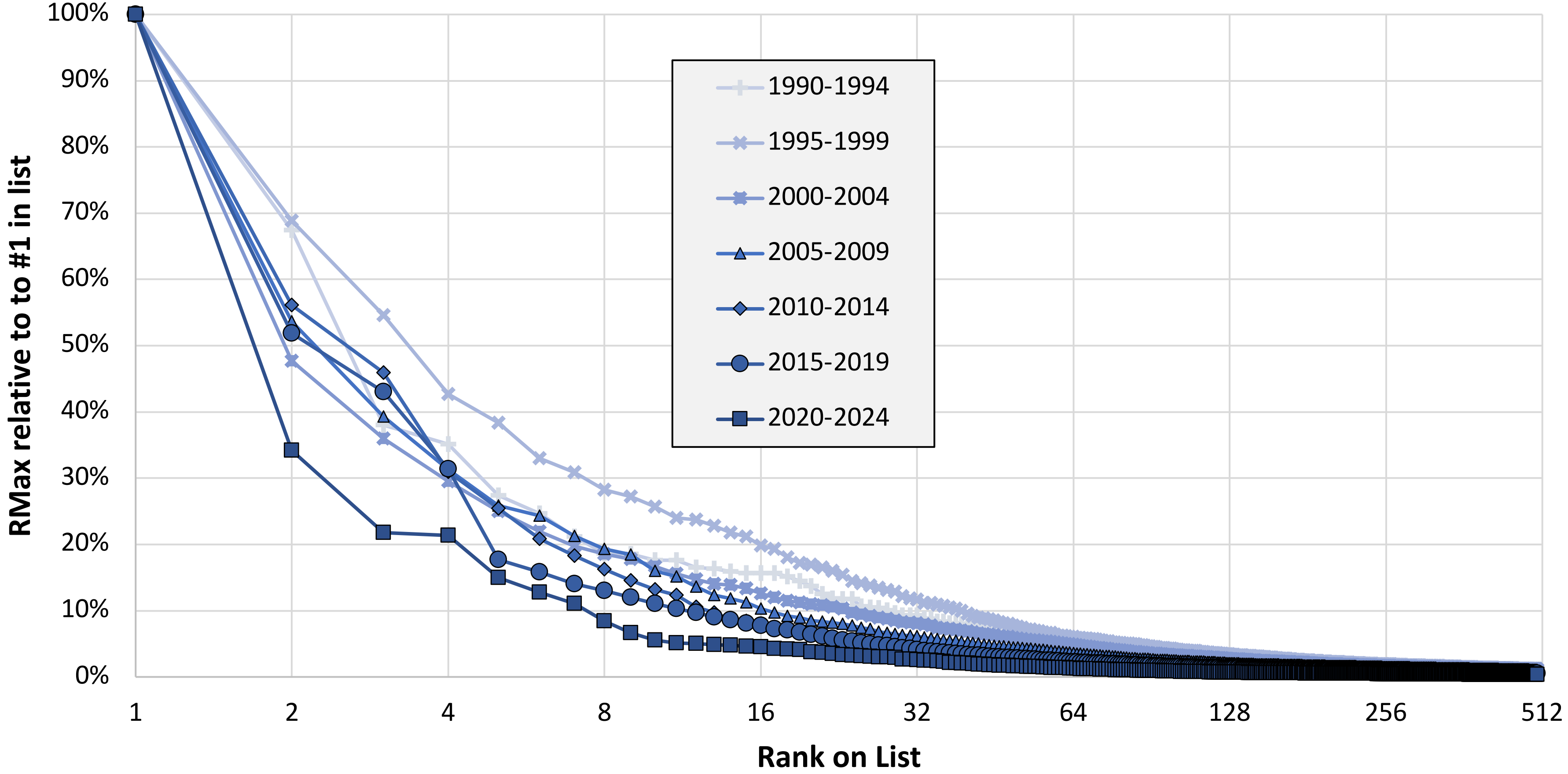


THE UNIVERSITY OF TEXAS AT AUSTIN
TEXAS ADVANCED COMPUTING CENTER


Price per "Processor" over time
$10,000,000
$1,000,000
$100,000
$10,000
$1,000
$100
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
2025
Vector
RISC
x86 single-core
x86 multi-core ($/socket)
x86 multi-core ($/core)

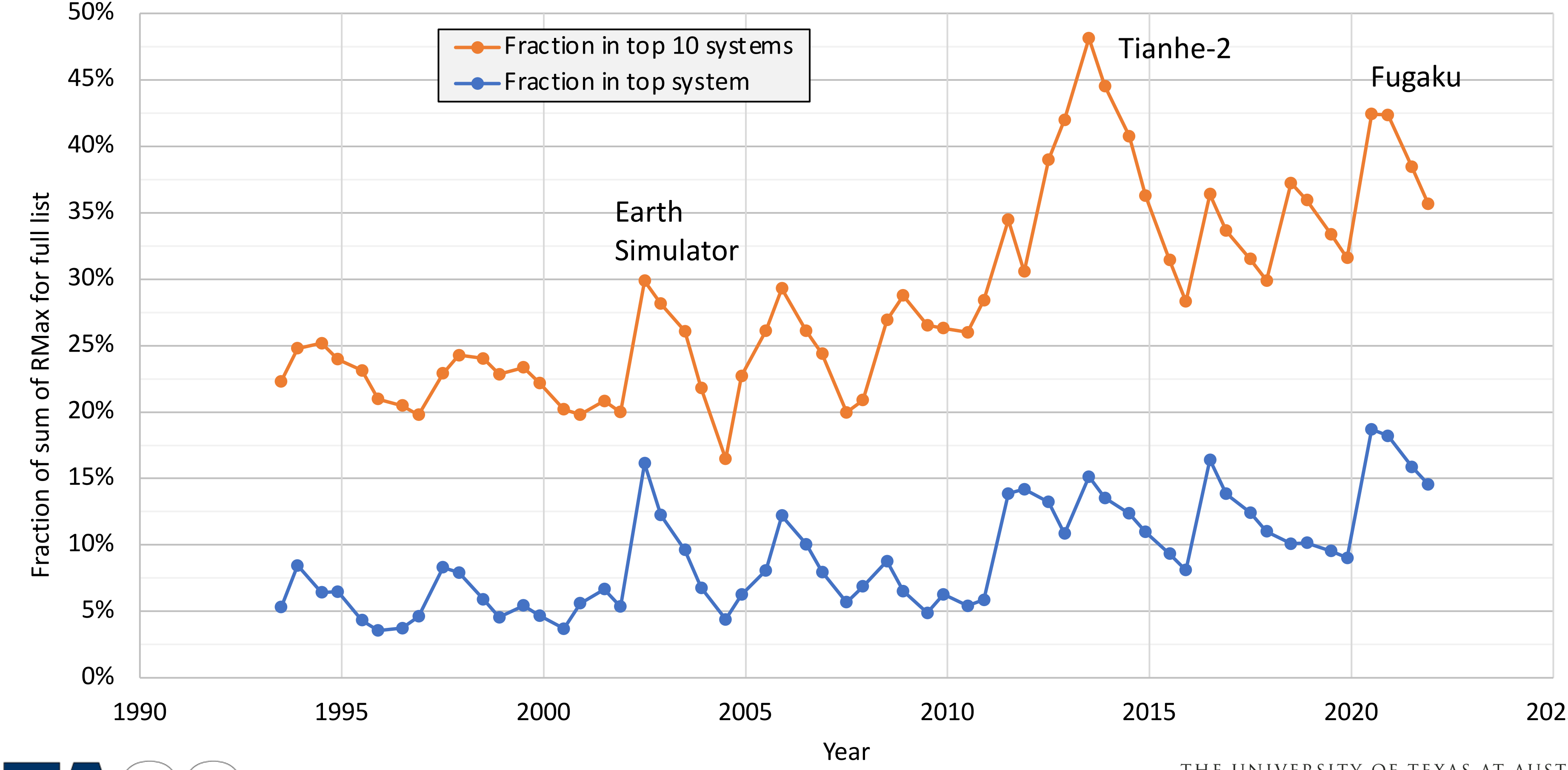

How many cores for $50K US?
$10,000,000
$1,000,000
$100,000
$10,000
$1,000
$100
1000
100
10
1
0.1
0.01
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
2025
Vector
RISC
x86 single-core
x86 multi-core ($/socket)
x86 multi-core ($/core)
Cores/budget

TOP500 Scaled RMax by Rank (averaged by half-decade)



THE UNIVERSITY OF TEXAS AT AUSTIN
TEXAS ADVANCED COMPUTING CENTER

The TOP500 list is becoming increasingly top-heavy



Part 2

Technology Trends & System Balances

What are "*System Balances*"?

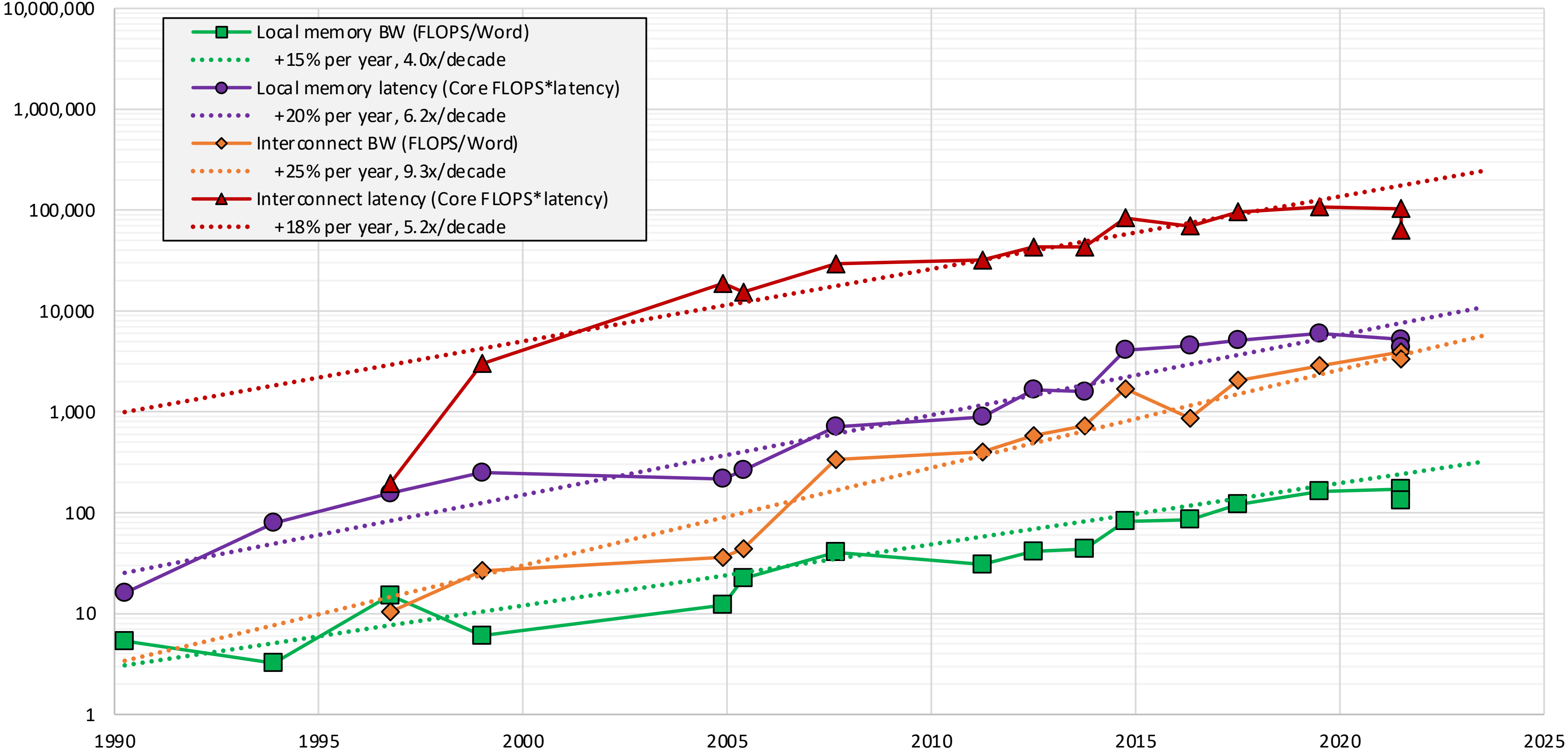
- "Performance" can be viewed as an N-dimensional vector of "mostly-orthogonal" components, e.g.:
  - Core performance (FLOPs) – *LINPACK*
  - Memory Bandwidth – *STREAM*
  - Memory Latency – *lmbench/lat_mem_rd*
  - Interconnect Bandwidth – *osu_bw, osu_bibw*
  - Interconnect Latency – *osu_latency*
- *System Balances* are the ratios of these components

Performance Component Trends

1. Peak FLOPS per socket increasing at **50%-60% per year**
2. Memory Bandwidth increasing at **~23% per year**
3. Memory Latency increasing at **~4% per year**
4. Interconnect Bandwidth increasing at **~20% per year**
5. Interconnect Latency decreasing at **~20% per year**

- *These ratios suggest that processors should be increasingly imbalanced with respect to data motion….*
  - *Today's talk focuses on (1), (2), and a bit of (3)*

Historical Balance Trends: Revised to 2021-12-14

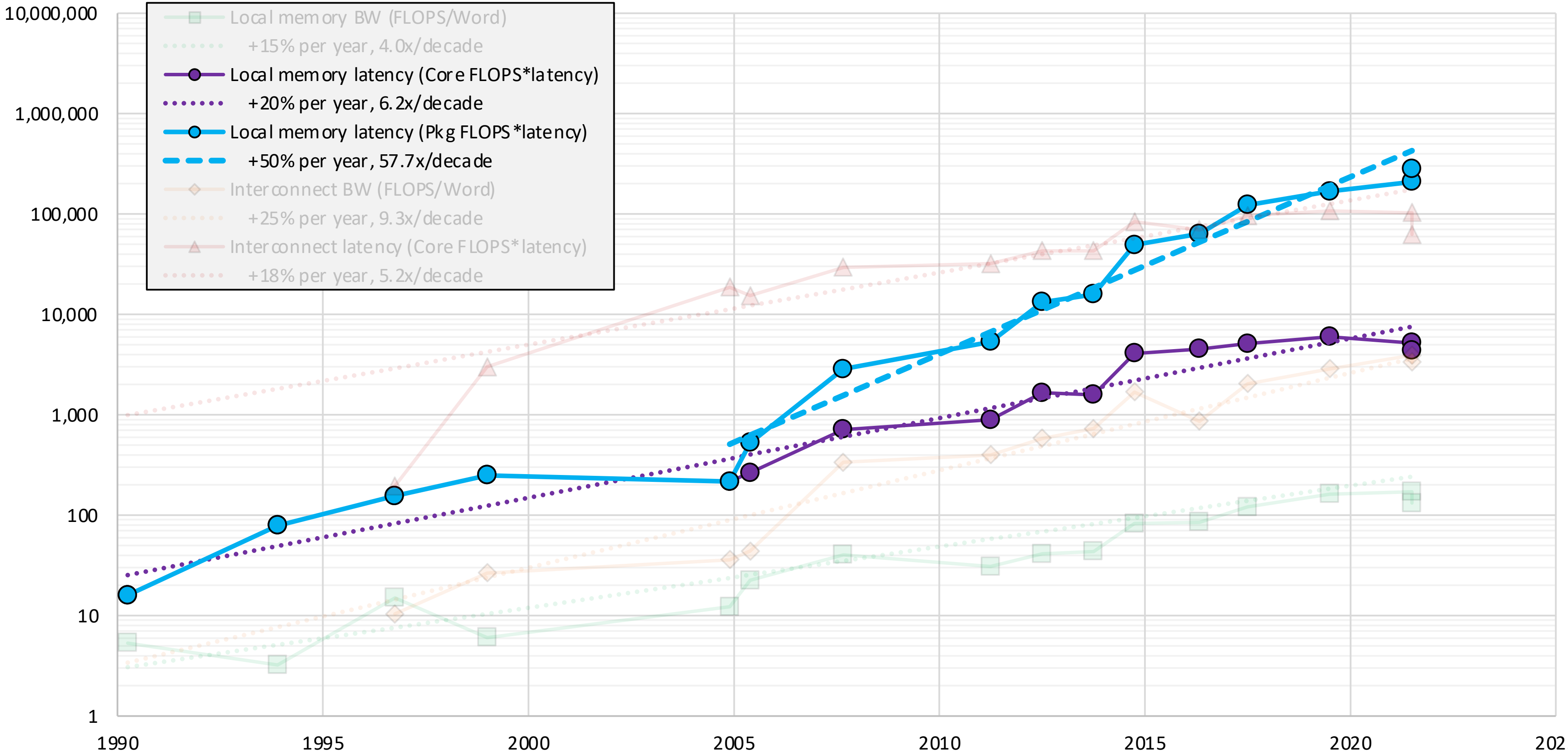


TACC

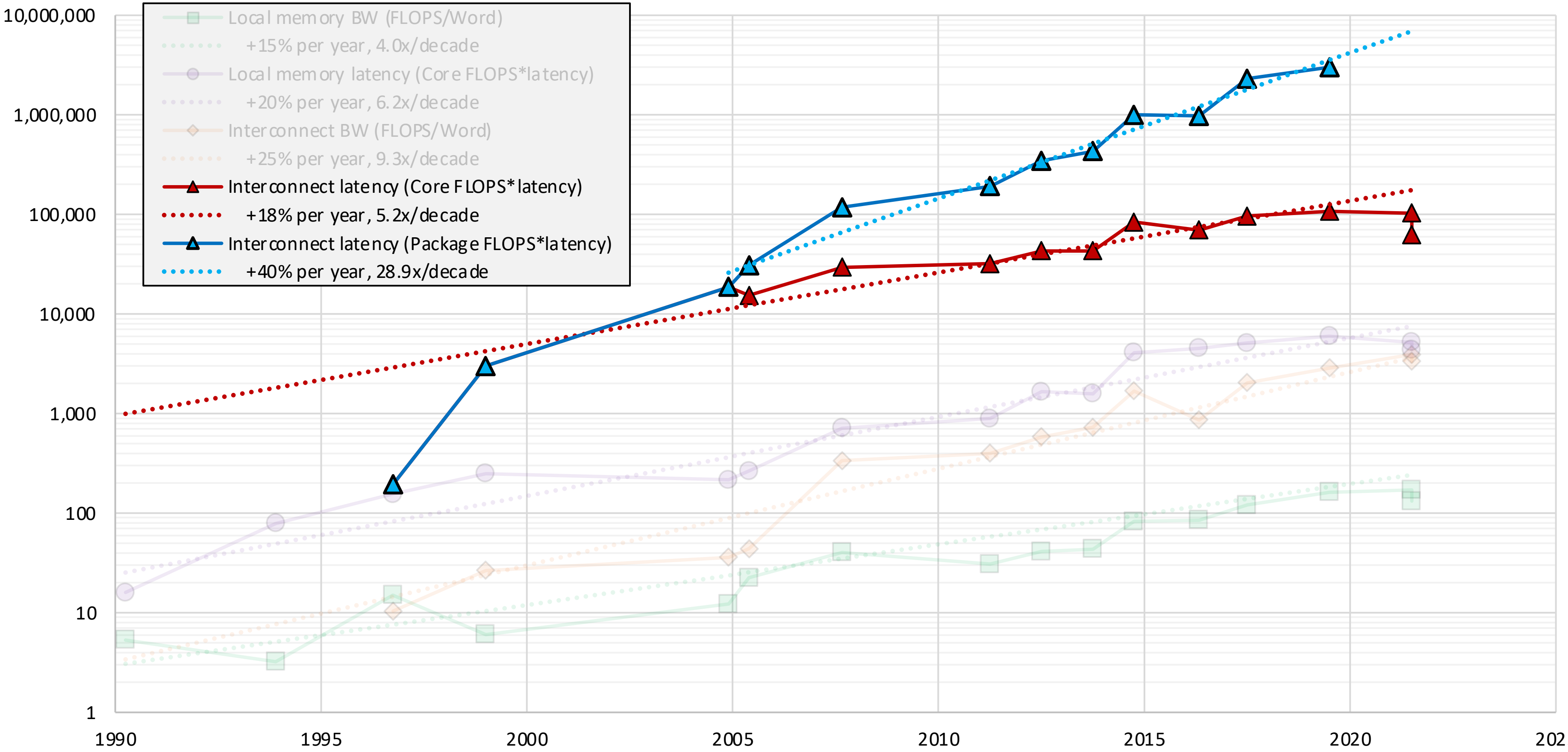
THE UNIVERSITY OF TEXAS AT AUSTIN

TEXAS ADVANCED COMPUTING CENTER

What if entire package stalls on local memory latency?



What if entire package stalls on interconnect latency?

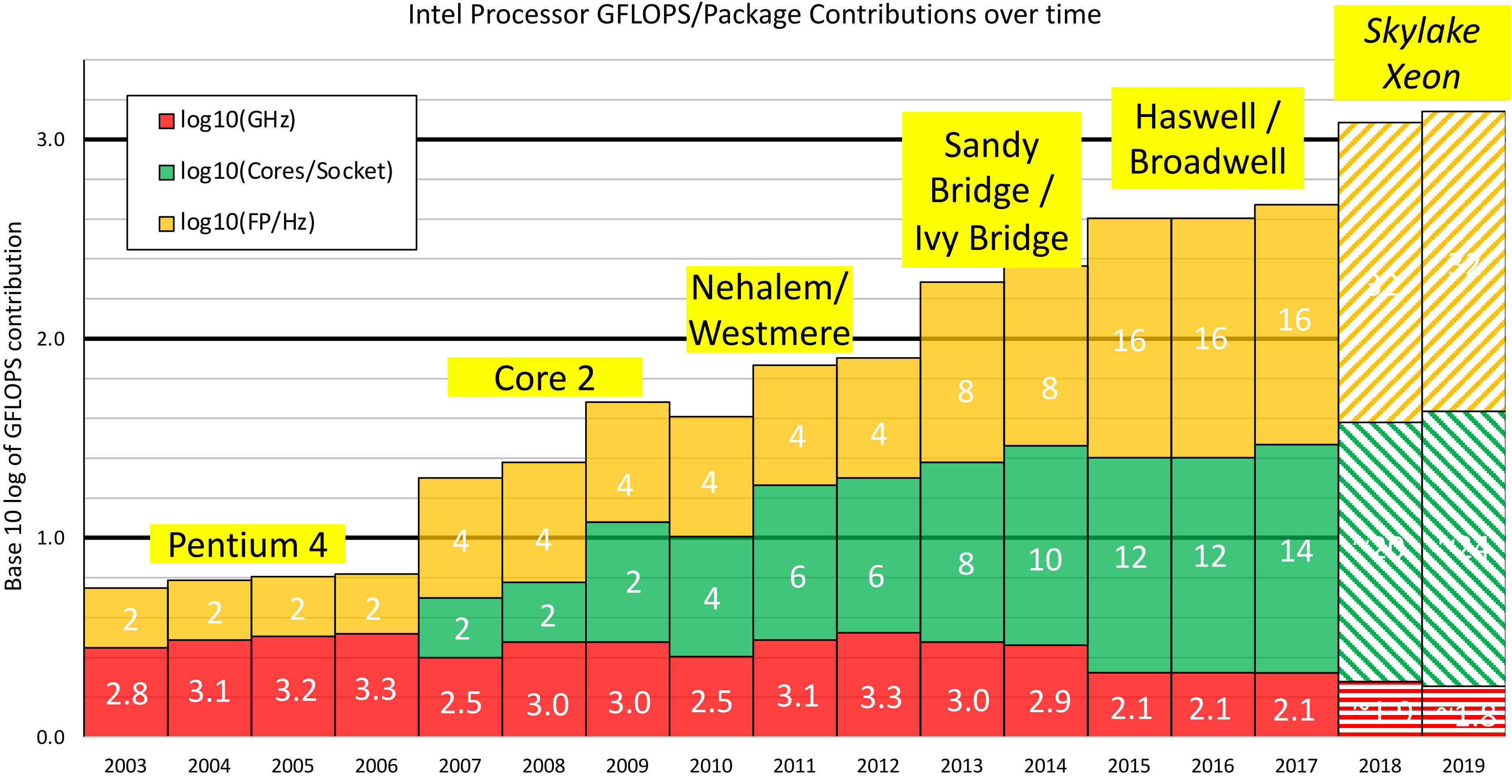


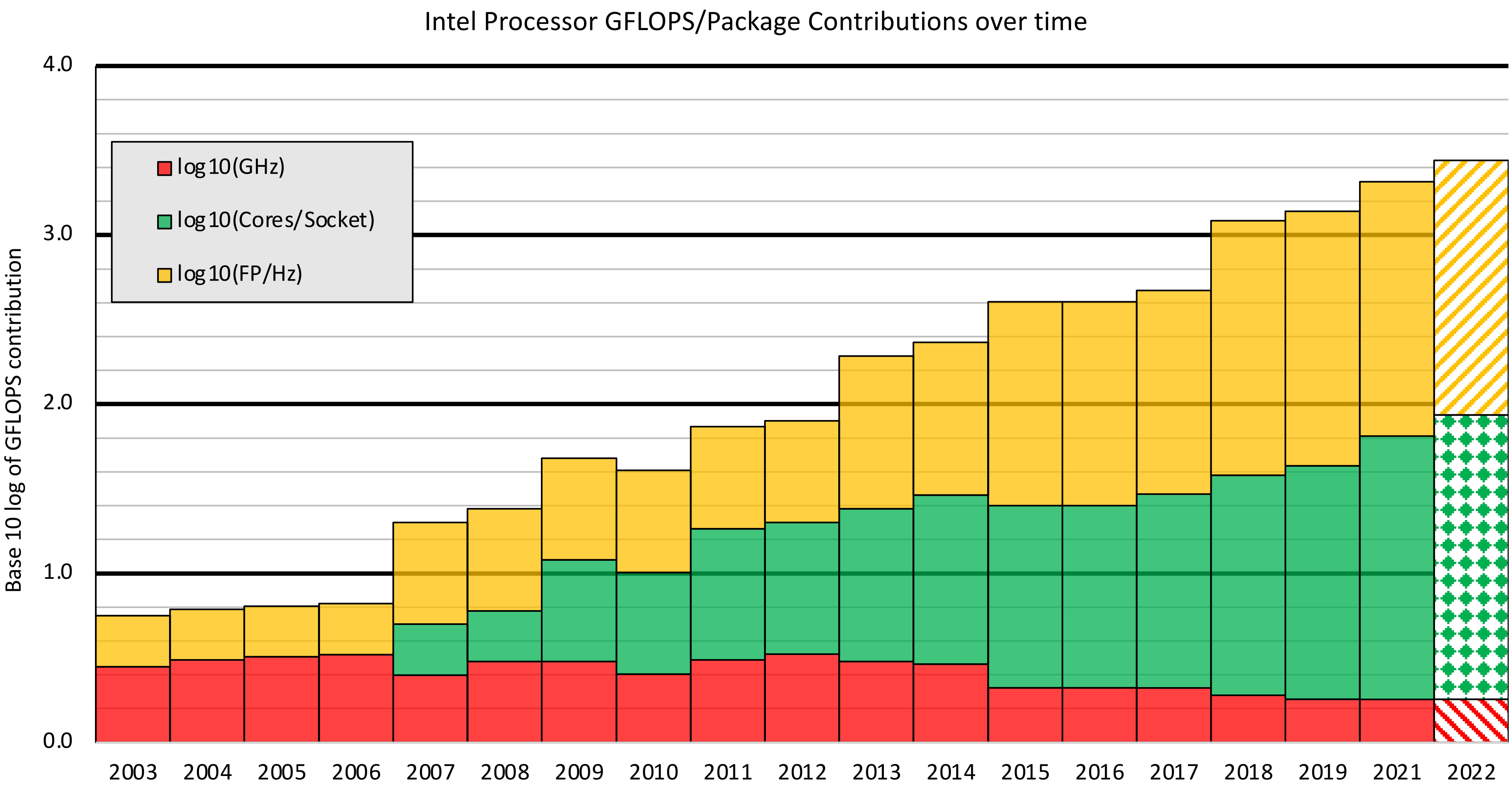
THE UNIVERSITY OF TEXAS AT AUSTIN
TEXAS ADVANCED COMPUTING CENTER

Why are FLOPS increasing so fast?

- Peak FLOPs per package is the product of several terms:
  - Frequency
  - FP operations per cycle per core
    - Product of #FP units, SIMD width of each unit, and complexity of FP instructions (e.g., separate ADD & MUL vs FMA)
  - Number of cores per package
- Low-level semiconductor technology tends to drive these terms at different rates…

Intel Processor GFLOPS/Package Contributions over time





THE UNIVERSITY OF TEXAS AT AUSTIN
TEXAS ADVANCED COMPUTING CENTER

Why is Memory Bandwidth increasing slowly?

- Slow rate of pin speed improvements
  - Emphasis has been on increasing capacity, not increasing bandwidth
  - Shared-bus architecture (multiple DIMMs per channel) is very hard at high frequencies
- DRAM cell cycle time almost unchanged in 20 years
  - Speed increases require increasing transfer sizes
  - DDR3/DDR4 have minimum 64 Byte transfers in DIMMs
- Slow rate of increase in interface width
  - Pins cost money!

Why is Memory Latency stagnant or growing?

- More levels in cache hierarchy
  - Many lookups serialized to save power
- More asynchronous clock domain crossings
  - Many different clock domains to save power
  - *Snoop (6):* Core -> Ring -> QPI -> Ring -> QPI -> Ring -> Core
  - *Local Memory (4):* Core -> Ring -> DDR -> Ring -> Core
  - *Remote Memory (8):*
    Core -> Ring -> QPI -> Ring -> DDR -> Ring -> QPI -> Ring -> Core
- More cores to keep coherent
  - Challenging even on a single mainstream server chip
  - Two-socket system latency typically dominated by coherence, not data
  - Manycore chips have much higher latency
- Decreasing frequencies!

Why is Interconnect Bandwidth growing slowly?

- Slow rate of pin speed improvements
  - About 20%/year
- Reluctance to increase interface width
  - Switch chips typically pin-limited – wider interfaces get fewer ports
  - Parallel links require more switches – too expensive and does not always provide improved real-world bandwidth

Why is Interconnect Latency improving slowly?

- Legacy IO architecture designed around disks, not communications
    - Control operations using un-cached loads/stores – hundreds of ns per operation and no concurrency
    - Interrupt-driven processing requires many thousands of cycles per transaction
- Mismatch between SW requirements and HW capabilities

Latency, bandwidth, and concurrency

A different implication of these technology trends

Latency, Bandwidth, and Concurrency

- "Little's Law" from queuing theory describes the relationship between latency (or occupancy), bandwidth, and concurrency.

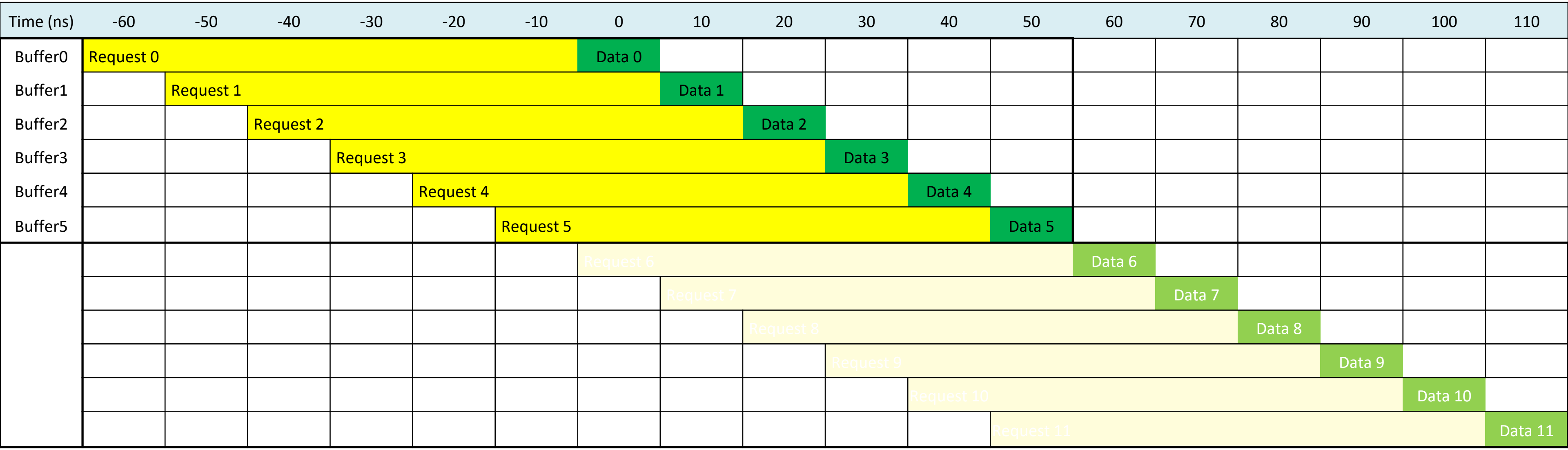
Latency * Bandwidth = Concurrency

- Flat Latency * Increasing Bandwidth → Increasing Concurrency

- Because these are exponential trends, these are not small changes…

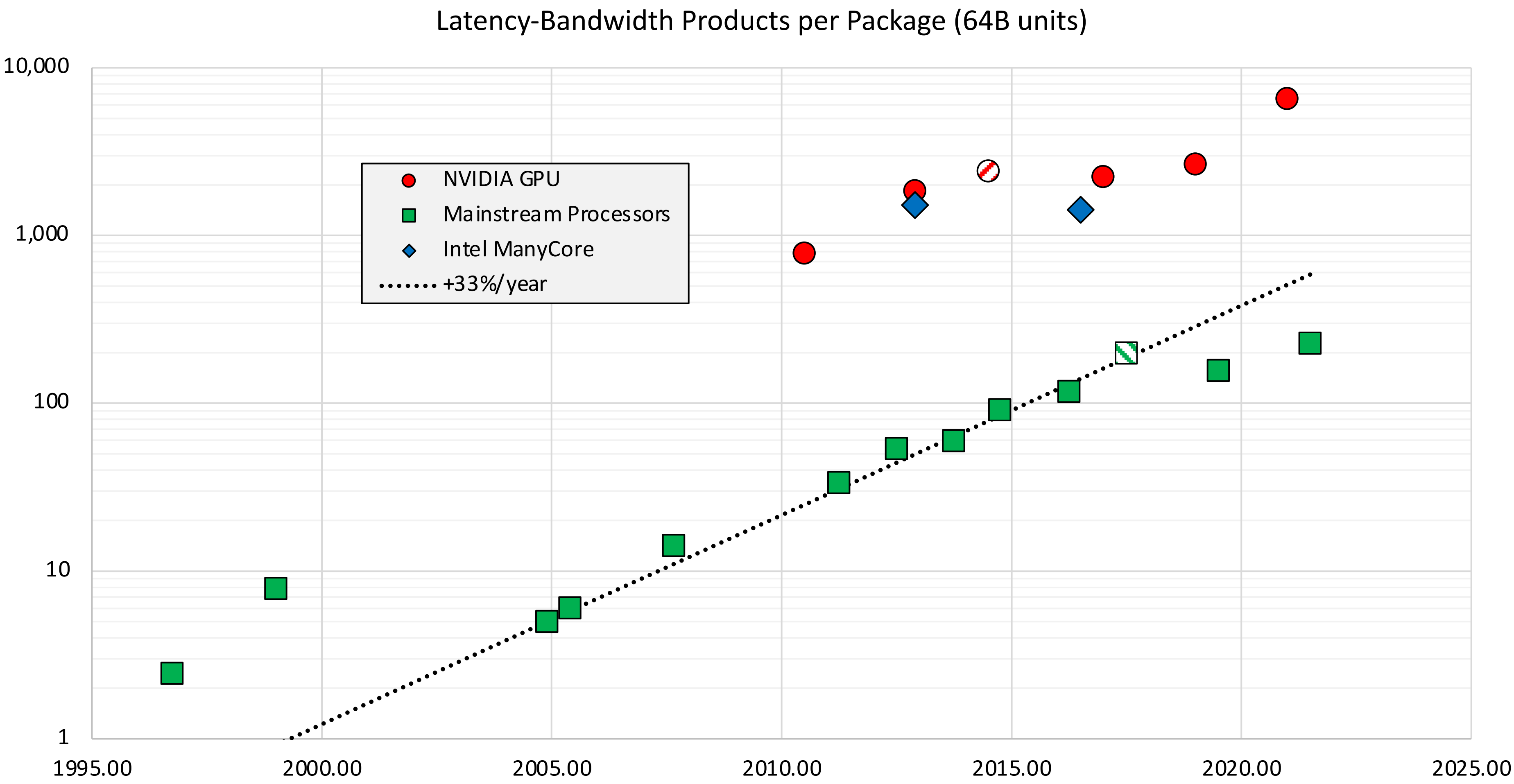
Little’s Law: illustration for 2005-era Opteron processor

60 ns latency, 6.4 GB/s (=10ns per 64B cache line)



Time (ns)	-60	-50	-40	-30	-20	-10	0	10	20	30	40	50	60	70	80	90	100	110
Buffer0	Request 0						Data 0											
Buffer1		Request 1						Data 1										
Buffer2			Request 2						Data 2									
Buffer3				Request 3						Data 3								
Buffer4					Request 4						Data 4							
Buffer5						Request 5						Data 5						
							Request 6						Data 6					
								Request 7						Data 7				
									Request 8						Data 8			
										Request 9						Data 9		
											Request 10						Data 10	
												Request 11						Data 11

- 60 ns * 6.4 GB/s = 384 Bytes = 6 cache lines
- To keep the pipeline full, there must always be 6 cache lines “in flight”
- Each request must be launched at least 60 ns before the data is needed


Latency-Bandwidth Products per Package (64B units)
NVIDIA GPU
Mainstream Processors
Intel ManyCore
+33%/year
10,000
1,000
100
10
1
1995.00
2000.00
2005.00
2010.00
2015.00
2020.00
2025.00

Why is Increasing Concurrency a Problem?

- Architectures are built assuming “flat” memory model
  - Location of data is invisible and uncontrollable
  - Caches and prefetchers are assumed to be “good enough” to cover latency and bandwidth differences
- Implementations support limited L1 Data Cache misses per core:
  - Xeon E5: 10 L1 misses (maximum)
  - L2 Hardware Prefetchers help, but are also “invisible” and not directly controllable

Increasing Concurrency (2)

- Many cores are needed just to generate concurrency, even if not needed to do computing
  - This costs a lot of energy in the cores!
- Large buffers and complex memory controllers are needed to handle the concurrent operations
  - DRAM page management requires memory schedule to be updated frequently as new transactions appear
  - DRAM open page hit rates still go down, so DRAM power increases too
  - Design cost up, power cost up, BW utilization down

Increasing Concurrency (3)

- More cores create more concurrent memory access streams, which requires more DRAM banks
- Examples:
  - 8-core Xeon E5 v1 with 2 streams per core needs >= 16 banks
    Requires 2 ranks of DDR3 DRAM (one dual-rank DIMM)
  - 12-core Xeon E5 v3 with 2 streams per core needs >= 24 banks
    Requires 2 ranks of DDR4 DRAM (one dual-rank DIMM)
- Problems:
  - Some codes generate many address streams per core – LBM >32
  - HyperThreading can double address streams per core
  - Adding more DIMMs can *decrease* performance due to rank-to-rank bus stalls

Power and energy

Another angle…

What about Power/Energy?

- Power density is important in processor implementations
  - Frequencies can be limited by small-scale (core-sized) hot spots
  - Multi-core frequencies are now limited by package cooling
  - E.g., Xeon E5 v3 (Haswell) can only run DGEMM or LINPACK on ½ of the cores before running out of power & needing to throttle frequency
- Power is not a first-order concern in operating cost!!!
  - Purchase price is $2500-$4000/socket
  - Socket draws 100-150 Watts & needs 40-50 Watts for cooling
  - At $0.10/kWh, this is 5%-7% of purchase price per year
  - This ratio is very hard to change!!!

What about Power/Energy later?

- If much cheaper processors become available, power would become a first-order cost
- Example 1: “client” multicore processors
  - Use the same core architecture, but at much lower price
  - Typical configuration needs 25% of purchase cost per year for power
  - (High performance interconnect solution not available at reasonable price)
- Example 2: “embedded” processors
  - Hypothetical $5 processor using 5 Watts requires $7/year for power
  - Not a problem for mobile – not credible for HPC
  - Response will be sociological and bureaucratic, as well as technical

John D. McCalpin, PhD

mccalpin@tacc.utexas.edu

512-232-3754

For more information:
www.tacc.utexas.edu