

# Data Processing on Modern Hardware

Viktor Leis

# What This Course is About

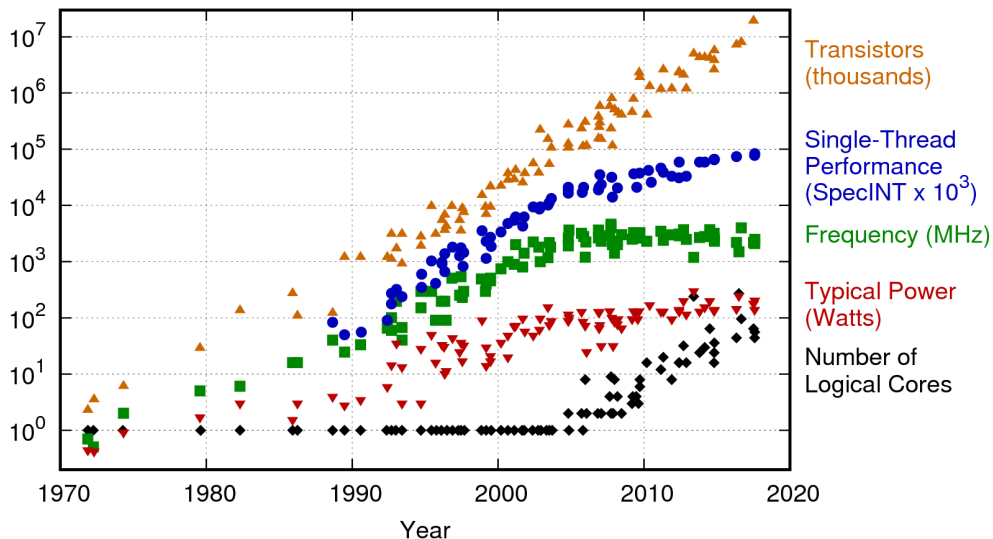
- make programs faster on modern multi-core CPUs using
  - ▶ data-parallel instructions (SIMD)
  - ▶ efficient synchronization of data structures
  - ▶ parallelization of algorithms
- many of the techniques and examples presented in this course are used in modern database systems, but are also applicable more generally whenever performance is important
- not part of this course:
  - ▶ number crunching with floating point numbers
  - ▶ distributed systems
  - ▶ GPU
  - ▶ FPGA
  - ▶ ...

# Programming Assignments

- this course is about skills, not just about abstract knowledge
- unless you do the assignments, this course is fairly pointless
- you will get a grade for the assignments  $h$ , which improves the grade of the exam  $e$ :  
 $\min(e, 0.6e + 0.4h)$
- (more information later)

# Hardware Trends

## 42 Years of Microprocessor Trend Data



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten  
New plot and data collected for 2010-2017 by K. Rupp

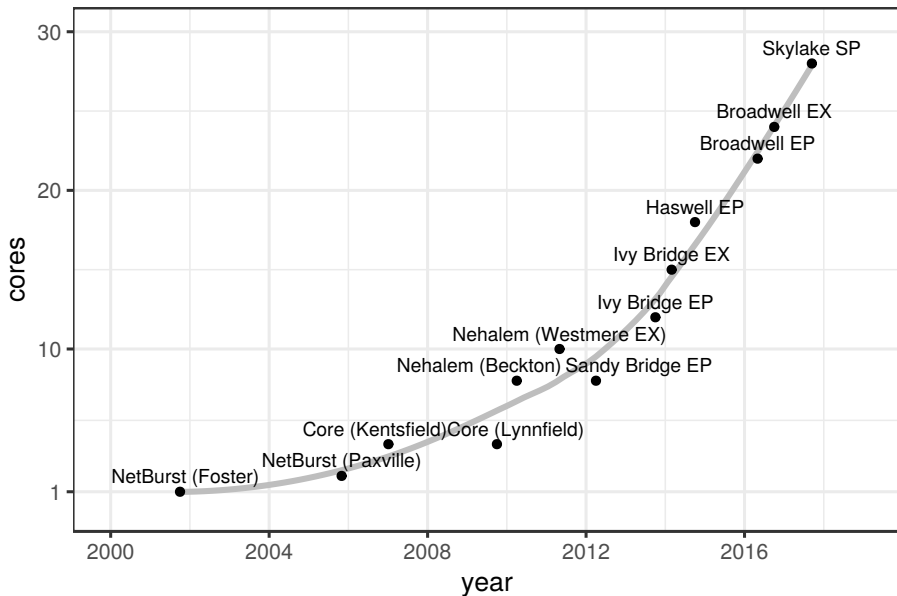
# Hardware Trends

- in the past, single-threaded performance doubled every 18-22 months
- now single-threaded performance is stagnating (single digit percentage growth)
- number of transistors is still growing quickly (“Moore’s law”)
- as a result, chips offer more and more parallelism (in particular on servers)
- to benefit from this parallelism, the software must generally be rewritten (“the free lunch is over”)
- software is becoming “a producer of performance” instead of a “consumer of performance” (Mark Hill)

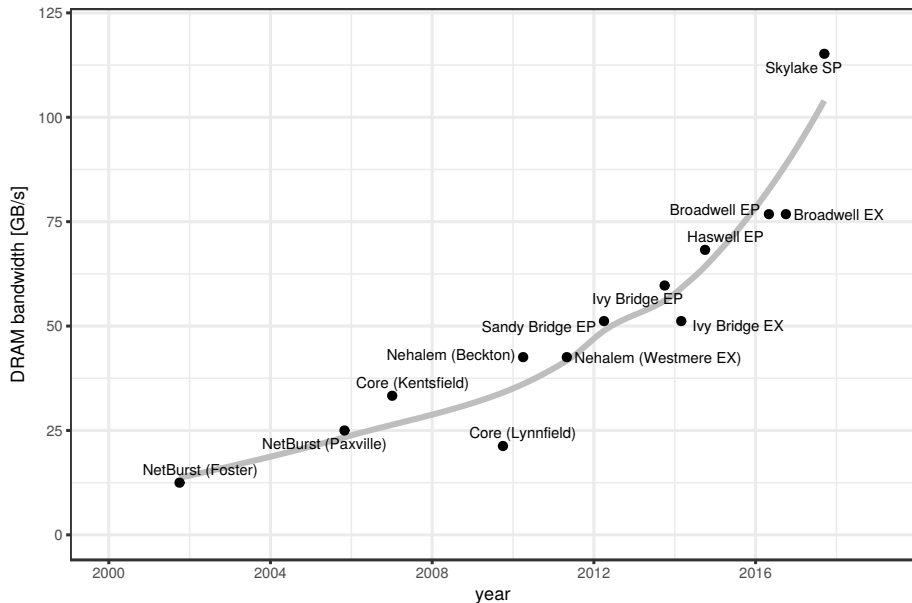
# Single Instruction, Multiple Data (SIMD) Width

- 1997: MMX 64-bit (Pentium 1)
- 1999: SSE 128-bit (Pentium 3)
- 2011: AVX 256-bit float (Sandy Bridge)
- 2013: AVX2 256-bit int (Haswell)
- 2017: AVX-512 512 bit (Skylake Server)

# Number of Cores

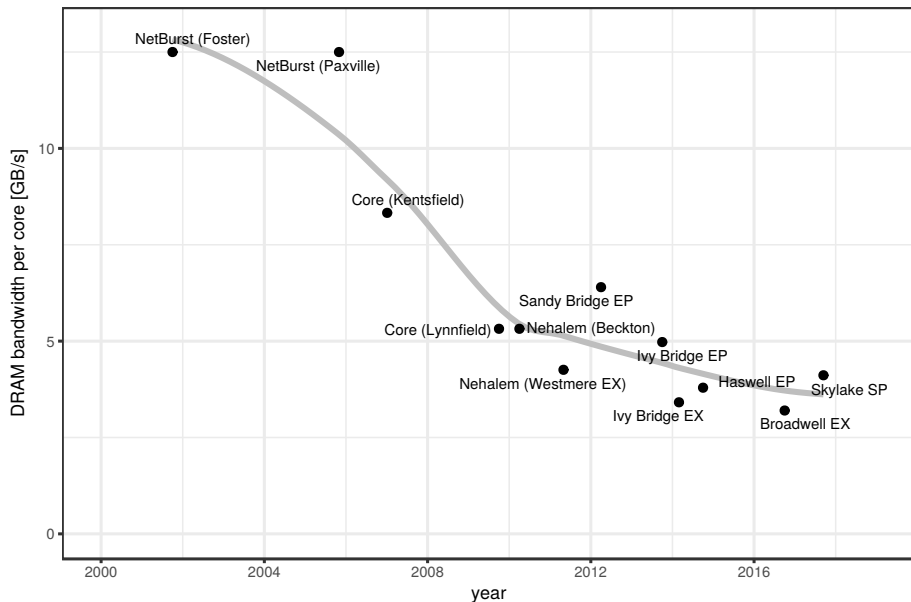


# Memory Bandwidth





# Memory Bandwidth Per Core



## Outlook: Dark Silicon

- heat dissipation is becoming a major problem
- modern CPUs already have close to 10 billion transistors<sup>1</sup>
- it is already impossible to power all available transistors at the same time
- SIMD code already runs at lower frequencies than sequential code
- one possible solution: more cores, but at low frequency (e.g., Intel's Many Integrated Core architecture: Xeon Phi)
- another possible solution: many specialized, heterogeneous cores and function units

---

<sup>1</sup>The Intel 8008, released in 1972, has 3500 transistors.

# Hardware Specialization

- special purpose instructions (e.g., AES encryption, video decoding)
- Graphics Processing Unit (GPU)
- Accelerated Processing Unit (APU, by AMD)
- Oracle Sparc T7, Data Analytics Accelerators (DAX): decompression
- Oracle RAPID (research project)<sup>2</sup>
- Google Tensor Processing Unit (TPU)<sup>3</sup>
- Field-Programmable Gate Array (FPGA): “software-defined hardware”
- Application-Specific Integrated Circuit (ASIC), e.g., for bitcoin mining
- historical: Gamma database machine, Lisp machine

---

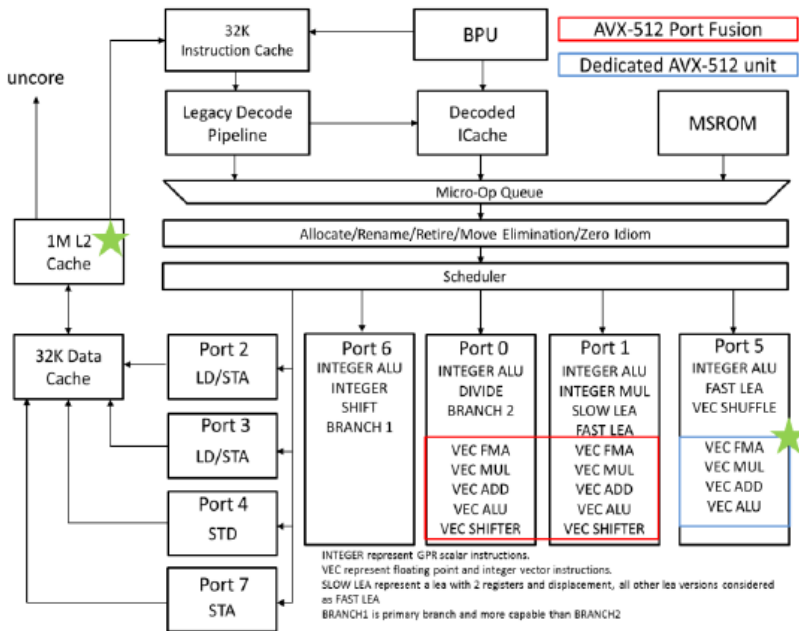
<sup>2</sup>[http://pages.cs.wisc.edu/~venkatv/MICRO-50\\_223-Camera-Ready.pdf](http://pages.cs.wisc.edu/~venkatv/MICRO-50_223-Camera-Ready.pdf)

<sup>3</sup><https://arxiv.org/abs/1704.04760>

# Skylake Server

- server variant of Intel Skylake microarchitecture, up to 28 cores
- available since August 2017
- our model: Intel Core i9-7900X (10 cores, 3.3–4.3GHz)

# Skylake Server Pipeline



## Skylake Server Caches and Memory

	L1	L2	L3
type	per core	per core	shared
size per core [KB]	32	1024	1408
latency [cycles]	4-6	14	50-70
max bandwidth [bytes/cycle]	192	64	16
sustained bandwidth [bytes/cycle]	133	52	15
associativity	8	16	-

- cache line size: 64 byte
- Skylake SP (up to 28 cores): 6 channels DDR4 2400  
(nominal bandwidth:  $6 * 2400 \text{ MHz} * 8 \text{ byte} = 112 \text{ GB/s}$ )
- Skylake X (up to 18 cores): 4 channels DDR4 2100  
(nominal bandwidth:  $4 * 2100 \text{ MHz} * 8 \text{ byte} = 66 \text{ GB/s}$ )

## Simultaneous Multithreading (SMT) aka Hyperthreading

- goal: improve utilization of execution units
- each core is exposed as 2, 4, or 8 hardware threads
- threads running on the same core share most resources (e.g., L1 cache, execution units)
- fully transparent to software and OS (looks like “real” cores)
- allows hiding latencies (from cache misses, expensive instructions, etc.)
- one cannot expect linear speed up from this, but often gives moderate performance boost at very little hardware cost
- Intel: 2-way Hyperthreading

## Low-Level Concepts/Terminology to Know

- pipelining, out-of-order execution, issue width
- branch prediction
- x86 uses little endian byte order