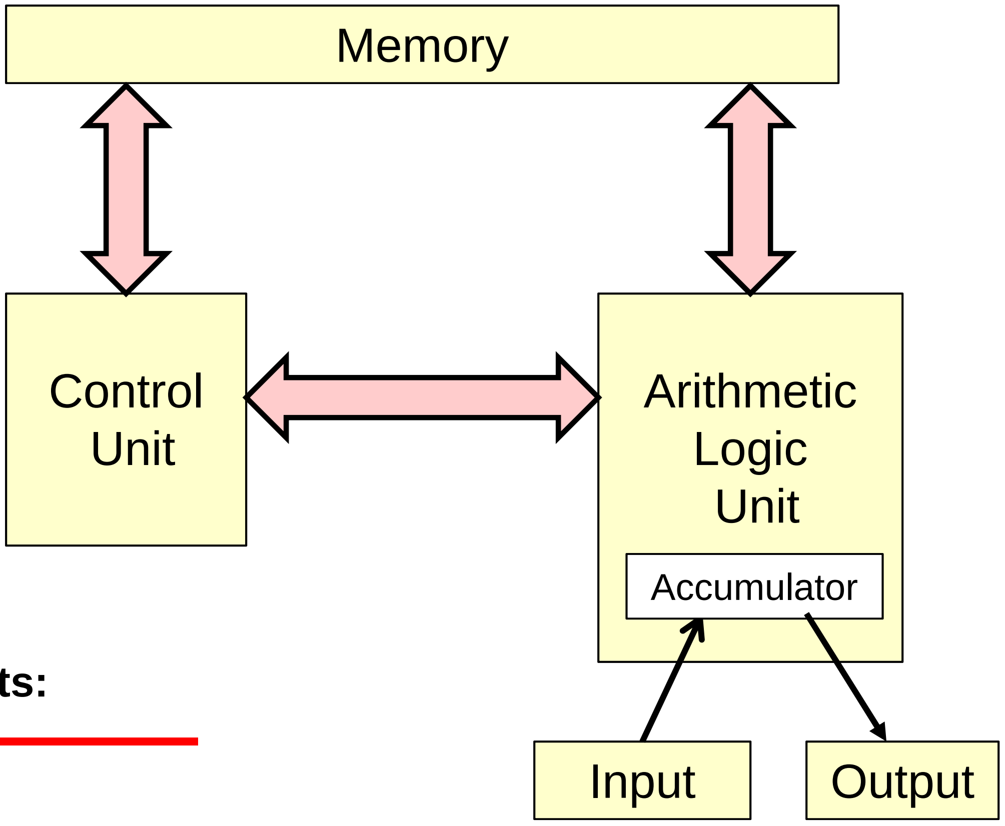
Parallel Programming: Moore's Law and Multicore

Mike Bailey

mjb@cs.oregonstate.edu

Oregon State University

Von Neumann Architecture: Basically the fundamental pieces of a CPU have not changed since the 1960s



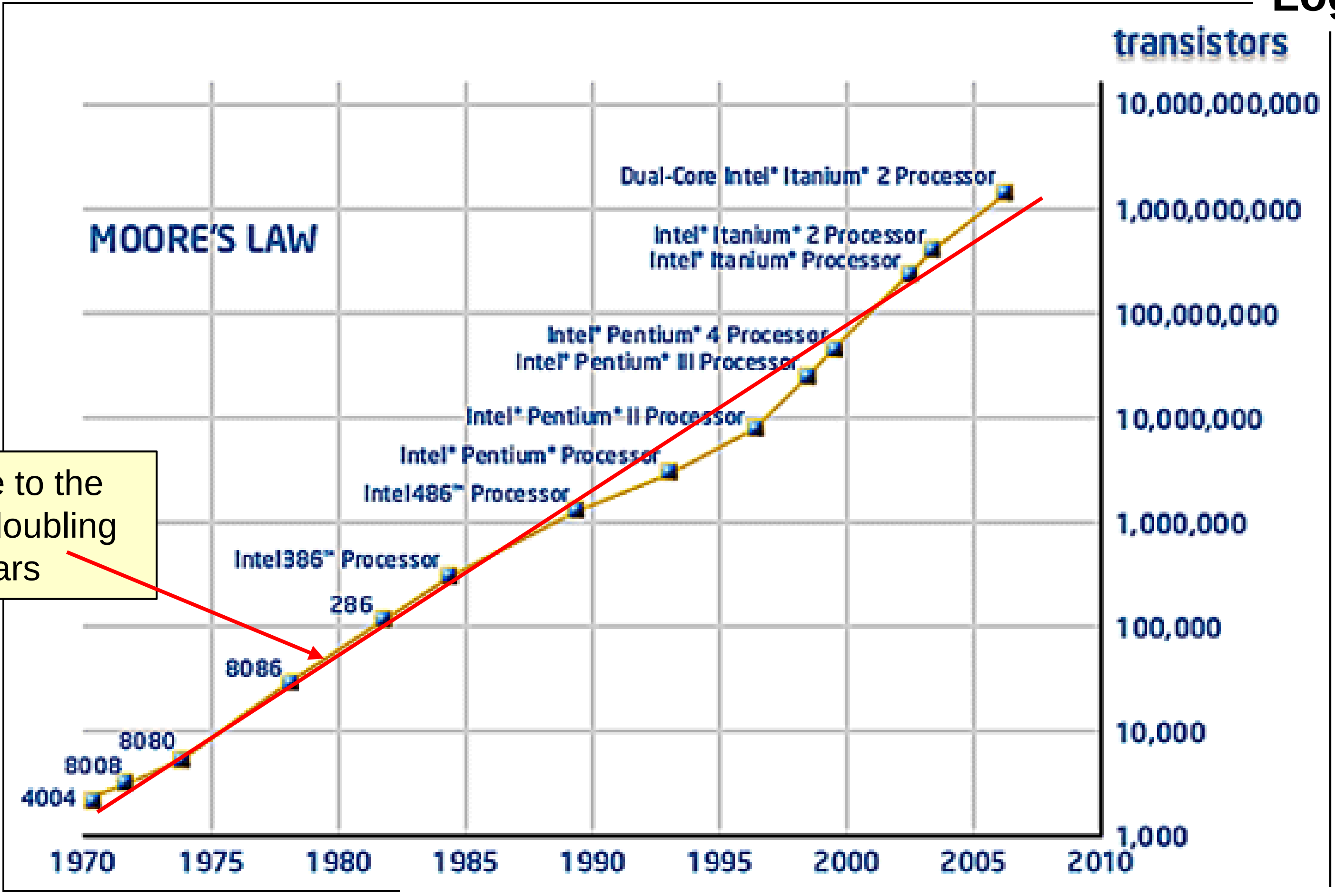
Other elements:

- **Clock**
- Registers
- Program counter
- Stack pointer

Increasing Transistor Density -- Moore's Law

"Transistor density doubles every 1.5 years."

Note:
Log scale!

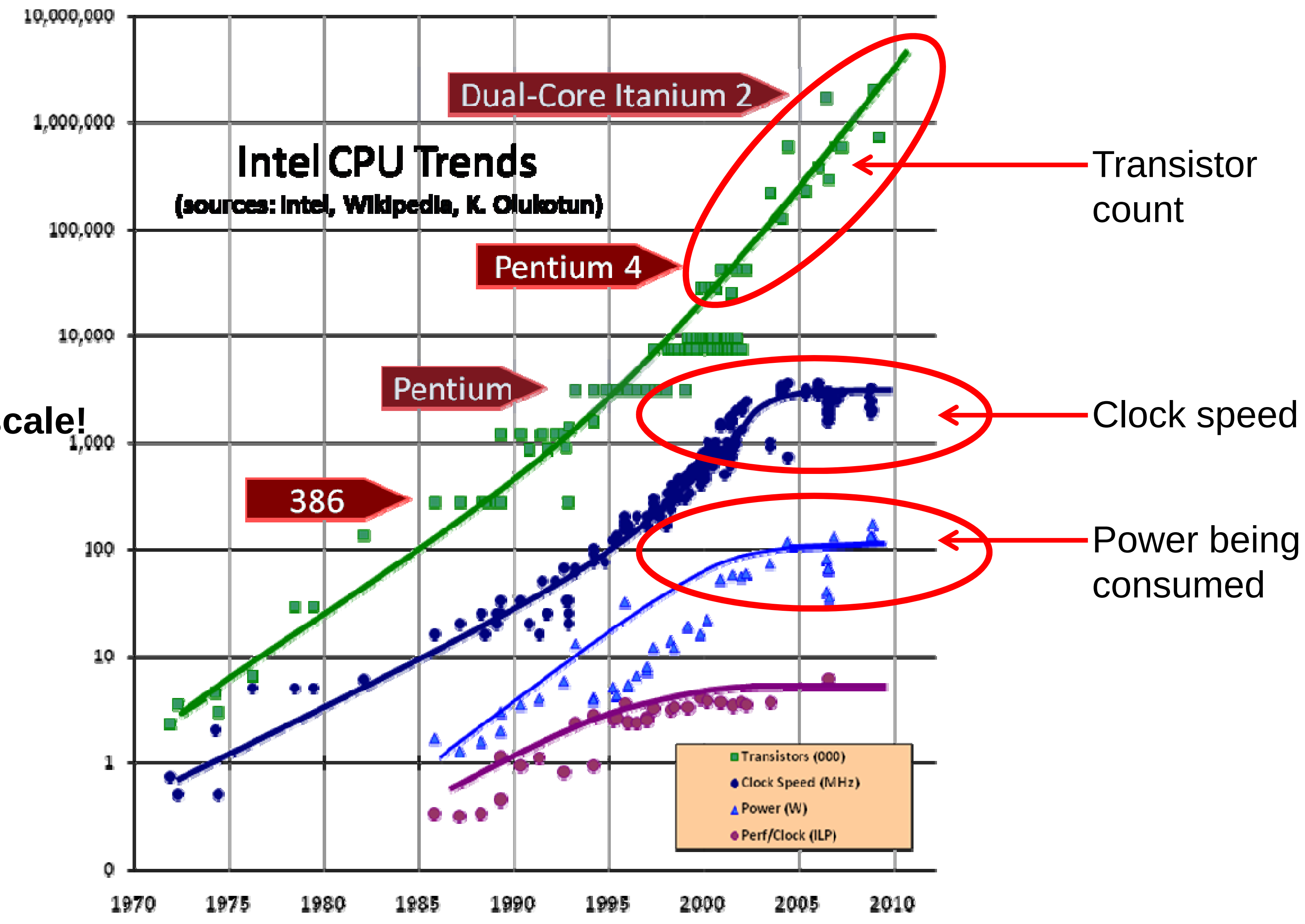


If I fit this line to the plot, I get a doubling every 1.6 years

Source: http://www.intel.com/technology/mooreslaw/index.htm

Oftentimes people have (*incorrectly*) equivalenced this to:
"Clock speed doubles every 1.5 years."

Increasing Clock Speed?



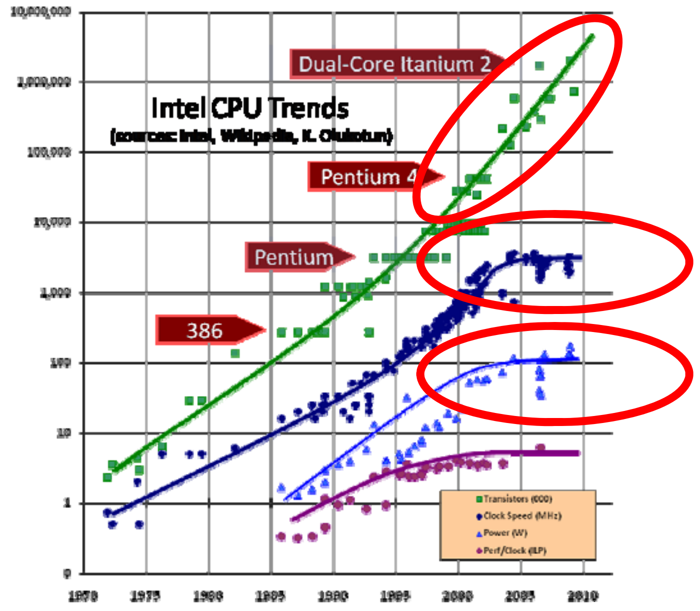
Note:
Log scale!

Source: Intel

Moore's Law

- Fabrication process sizes ("gate pitch") have fallen from 65 nm, to 45 nm, to 32 nm, to 22 nm, to 16 nm, to 11 nm. This translates to more transistors on the same size die.

- From 1986 to 2002, processor performance increased an average of 52%/year, but then virtually plateaued.



Clock Speed and Power Consumption

1981	IBM PC	5 MHz
1995	Pentium	100 MHz
2002	Pentium 4	3000 MHz (3 GHz)
2007		3800 MHz (3.8 GHz)
2009		4000 MHz (4.0 GHz)

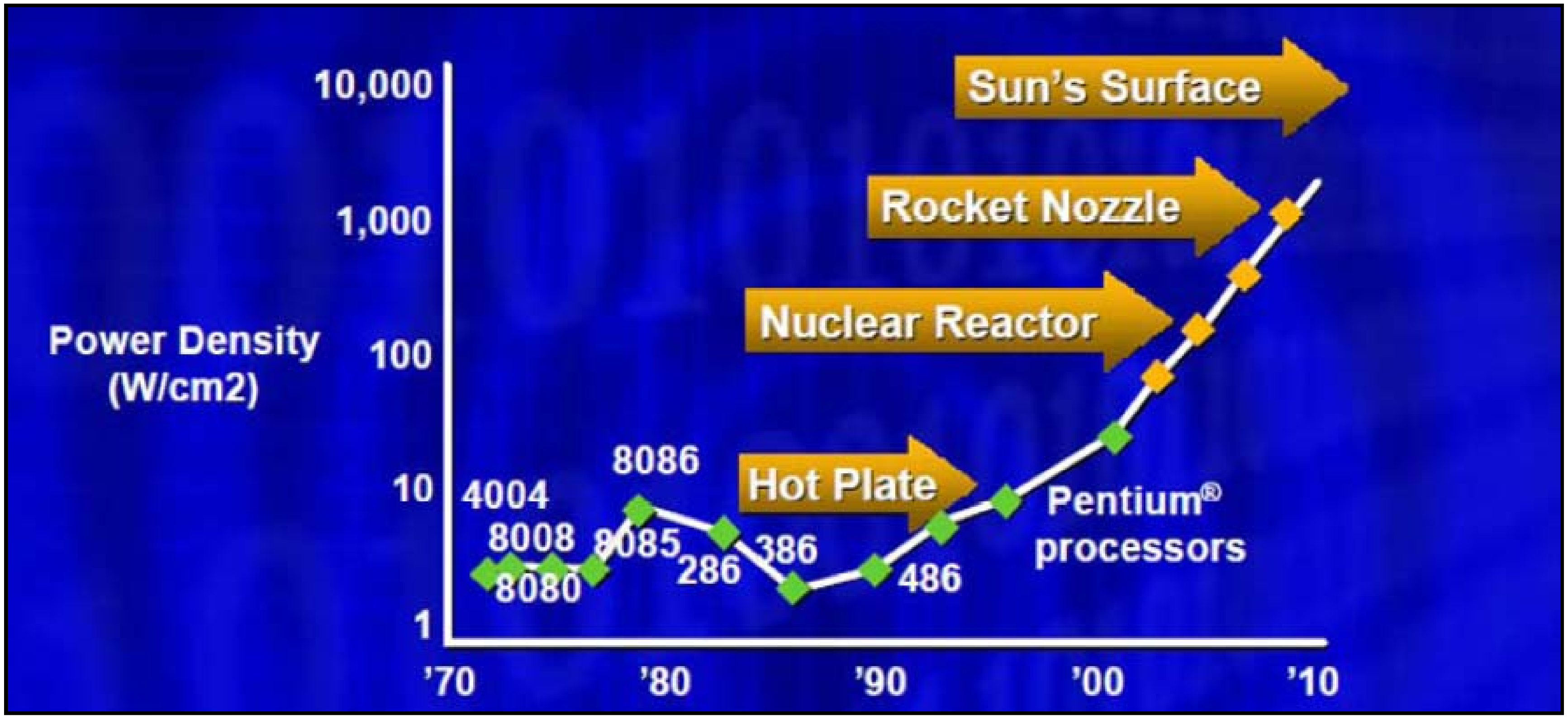
Clock speed has hit a plateau, largely because of power consumption.

$$\boxed{PowerConsumption \propto ClockSpeed^2}$$

is-proportional-to

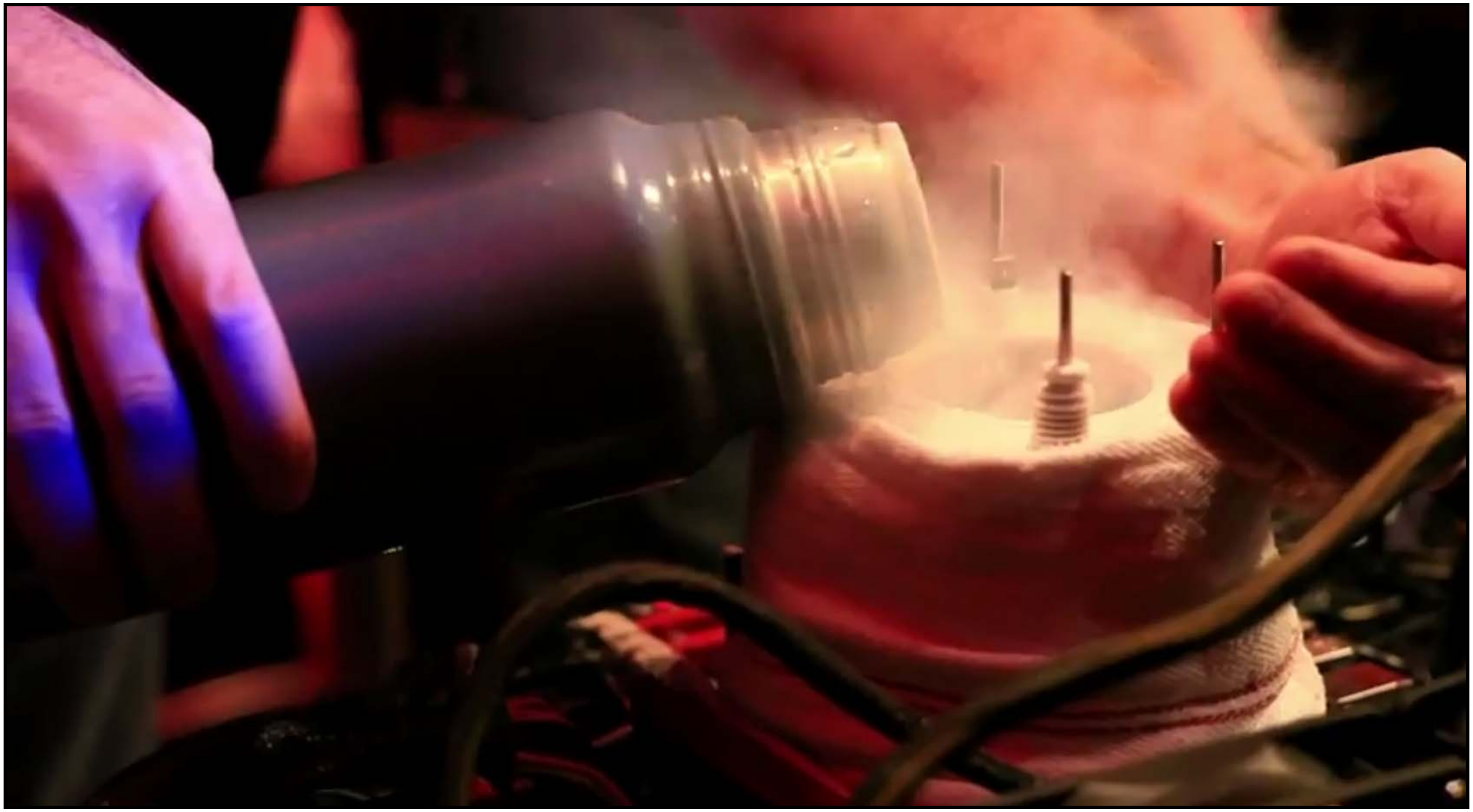
Yikes!

What Kind of Power Density Would it Have Taken to Keep up with Clock Speed Trends?



Source: Intel

Recently, AMD set the world record for clock speed (8.429 GHz) using a Liquid Nitrogen-cooled CPU



Source: ***AMD***

MultiCore -- Multiprocessing on a Single Chip

So, to summarize:

Moore's Law of transistor density is still going strong, but the "Moore's Law" of clock speed has hit a wall. Now what do we do?

We keep packing more and more transistors on a single chip, but don't increase the clock speed. Instead, we increase computational throughput by using those transistors to pack multiple processors onto the same chip.

This is referred to as ***multicore***.

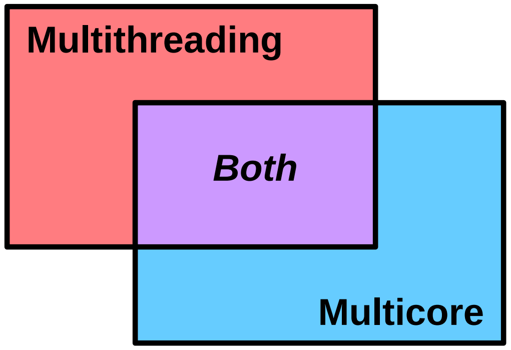
Vendors have also reacted by adding SIMD floating-point units on the chip as well.
We will get to that later.

MultiCore and Multithreading

Multicore, even without multithreading too, is still a good thing. It can be used, for example, to allow multiple programs on a desktop system to always be executing concurrently.

Multithreading, even without multicore too, is still a good thing. Threads can make it easier to logically have many things going on in your program at a time, and can absorb the dead-time of other threads.

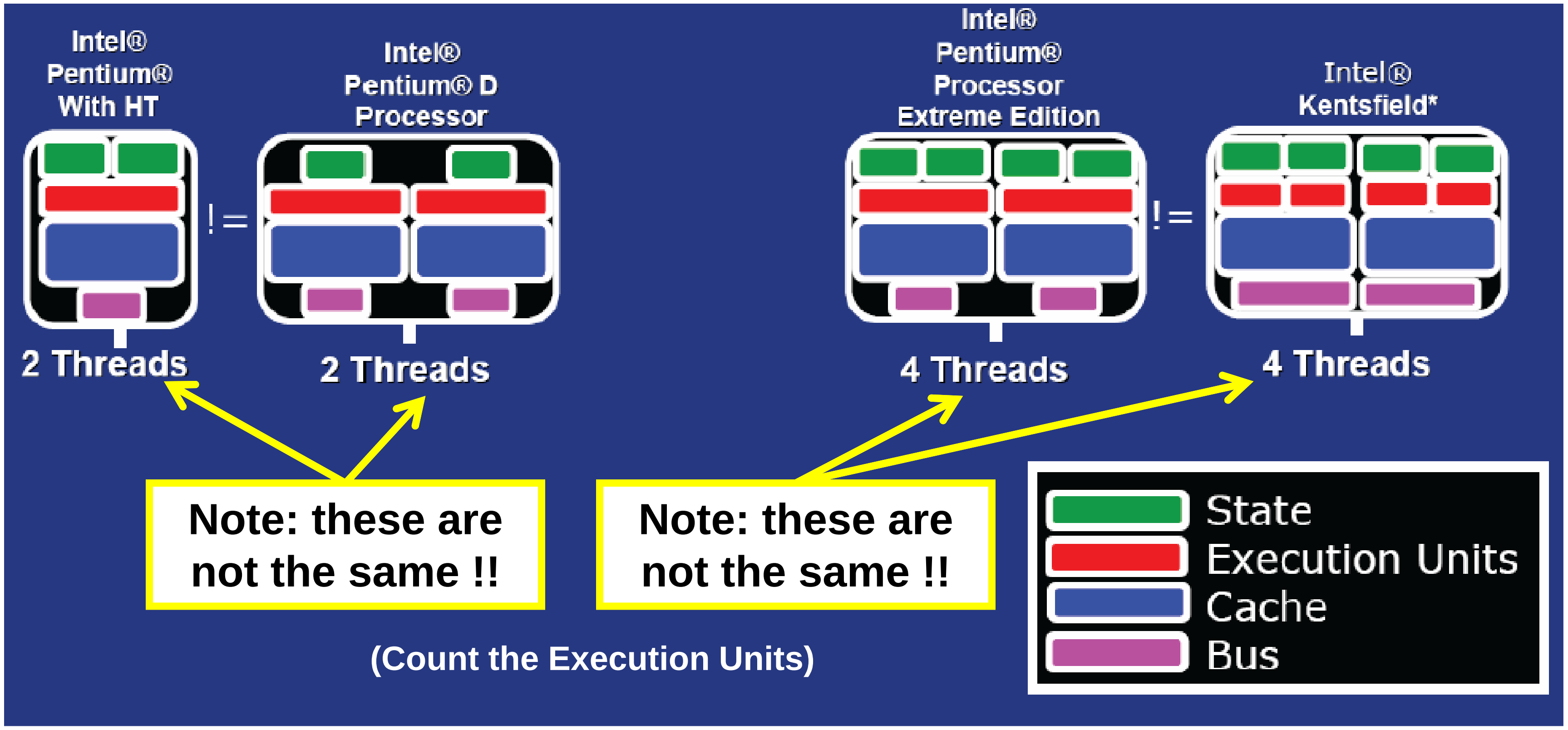
But, the big gain in performance is to use *both* to speed up a *single program*. For this, we need *a **combination of both multicore and multithreading***.



Multicore is a very hot topic these days. It would be hard to buy a CPU that doesn't have more than one core. We, as programmers, get to take advantage of that.

We need to be prepared to convert our programs to run on ***MultiThreaded Shared Memory Multicore*** architectures.

Intel's Approach to Multicore and Multithreading


Intel® Pentium® With HT
2 Threads
!=
Intel® Pentium® D Processor
2 Threads
Intel® Pentium® Processor Extreme Edition
4 Threads
!=
Intel® Kentsfield*
4 Threads
Note: these are not the same !!
Note: these are not the same !!
(Count the Execution Units)
State
Execution Units
Cache
Bus

Source: Intel

"HT" stands for "HyperThreading"