# Intel: from the i386 to today

Summer 2021

Possibly available from www.mjr19.org.uk/courses/

# CPU families

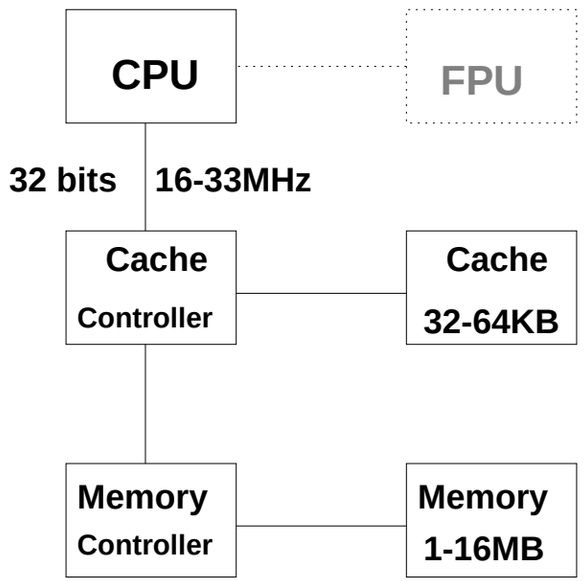
This discourse on the history of Intel CPUs serves two puposes. One is to be somewhat relevant – one cannot deny that you are almost certain to end up running programs on computers with Intel processors. The other is to demonstrate that not all CPUs are equal, and design choices need to be made.

# The 80386 (i386)

This CPU was launched in 1985, and first appeared in a PC in 1987. It is the oldest we will consider in detail.

It was Intel's first 32 bit processor, capable of 32 bit addressing and arithmetic operations on 32 bit integers. It had eight not quite general purpose integer registers, and no floating point support at all. For that one needed to procure a separate floating point unit.

It supported virtual memory, paging, and the concept of a privileged mode of operation. It was the first Intel processor capable of running UNIX (or any other modern operating system) in a sensible manner. It had around 275,000 transistors, and was initially made on a 1.5$\mu$m process.



# The FPU

The floating point unit was an optional separate chip, the 80387.

It supported the standard single (32 bit) and double (64 bit) precision IEEE floating point formats familiar today. However, its eight registers held data in an 80 bit extended format, and its functional units always used this extended precision. This 80 bit format was not unique to Intel. Motorola used it in its 68K range.

It was slow, not least because the protocol for transfering an instruction from the i386 to the FPU was slow, taking typically a dozen clock cycles. It supported the operations one would expect today, and also ones one would not, such as sine, tangent and arctangent, and logarithms.

Most instructions took a time which depended on their operands. An addition would be 23–34 clock-cycles, a multiplication around 50, a division around 90, a square root around 125, and the combined sincos instruction anything from about 200 to about 800 clock cycles!

# The i386: what was missing?

No cache.

No instruction re-ordering (OOO).

Not superscalar (but mostly pipelined, theoretically issuing one instruction every other clock cycle).

No register renaming.

Clock speed the same as its external databus (16MHz to 33MHz).

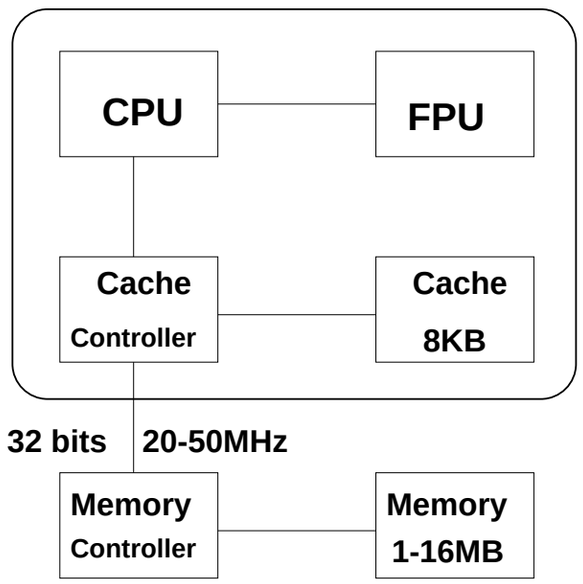
Optional floating point unit is pipelined at all, and very slow. It was incapable of reaching 1MFLOPS on the standard Linpack benchmark, which has approximately equal numbers of additions and multiplications.

# The 80486 (i486)

Released in 1989, depending on one's point of view, this was either a non-event, or a huge leap forwards. It offered no new capabilities, save that it combined a redesigned 80386, its optional floating point unit, its optional external write-through cache controller, and 8KB of cache, all on a single die, which contained 1.2 million transistors, over four times as many as the i386.

The integer unit was now fully pipelined, capable of issuing one instruction every clock cycle.

The floating point unit was still not pipelined, but simple operations now executed about twice as fast with addition being 8–20 clock cycles, and multiplication a constant 16. With clock speeds of 20MHz to 50MHz, it could just about achieve 1MFLOPS on Linpack.



# i486DX2 and DX4

Because the i486 had on-chip cache, it was sensible to increase its clock speed beyond that of the external bus. As long as the cache hit rate was good, then the slow speed of the external bus would not hold back the CPU.

So the i486DX2 ran at twice the speed of the external bus (25/50MHz or 33/66MHz), and the later i486DX4 at three times the speed of the external bus (25/75MHz or 33/100MHz). In order to improve its cache hit rate, the DX4 had a 16KB cache.

To ensure that contention between instruction and data accesses did not cause performance issues for the cache, a small prefetch buffer existed between the cache and the instruction fetcher. The bus to this prefetch buffer was 128 bits wide, enabling a single cache access to fetch multiple instructions.

For the first time heat dissipation became an issue. Even the DX2 at 5V (the traditional voltage for logic circuits) and 66MHz produced only about 6W, and barely needed a fan, but certainly needed a heatsink. The DX4 required a 3.3V supply, and, in many designs, a fan too.

# The Pentium

The Pentium was released in 1993. It was a complete redesign of the i486, although still used the same instruction set, so codes did not need to be recompiled.

The floating point unit was now fully pipelined, and the cache was now split into separate instruction and data caches.

It was the last Intel CPU to be introduced running at the same speed as its external bus (60 and 66MHz), and running at 5V. Its power dissipation was about 15W – it certainly needed a fan.

It was the first Intel CPU to support half integer clock multipliers, with the 90MHz and 100MHz version running at 1.5$\times$ the speed of their external bus, and the 200MHz variant at 2.5$\times$. All those running at more than 66MHz ran at 3.3V.

Not only was the bus speed doubled from the i486DX4 from 33MHz to 66MHz, but the bus width was doubled from 32 bits to 64 bits, where it would remain for a very long time.

# Superscalar

The Pentium had two separate instruction pipelines, called U and V. One could accept any instruction, the other a small subset of possible instructions. But, under the right conditions, instructions could pair and be issued to both pipes simultaneously.

So the Pentium was the first superscalar CPU in Intel's x86 range. To help maintain its pipeline, it had a branch predictor, which the i486 lacked.

The Pentium had 3.1 million transistors, over ten times as many as the i386. But even if run at the same clock speed it would offer a large performance increase over the i386, most significantly in floating point throughput, which might be up to twenty times higher, but even in integer instructions where the i386 could manage a theoretical peak of one instruction every other clock cycle, and the Pentium two instructions per clock cycle.

Not only that, but the first revision of the Pentium supported two processors in a single computer: SMP.

# A Name, not a Number

Intel discovered one could not trademark numbers, but could trademark names. So it is the Pentium™.

Also of political interest, both IBM and AMD had a licence which enabled them to fabricate and sell Intel processors from the 8086 to the i486 fairly freely. This licence did not extend to the Pentium.

Both AMD and IBM were quite good at taking Intel's designs, tweaking them slightly to improve them, and then making and selling their own versions.

# i386 to Pentium

| | i386 | i486 | Pentium |
|---|---|---|---|
| MIPS/MHz | 0.5 | 1 | 2 |
| MFLOPS/MHz | 0.02 | 0.05 | 1 |
| Cache | – | 8KB or 16KB | 8KB I + 8KB D |
| Transistors | 275,000 | 1.2m | 3.2m |
| MHz | 16–33 | 20–100 | 60–200 |
| bus width | 32 | 32 | 64 |

i386: supported up to Win95 and Win NT 3.51 inclusive, Linux up to 3.7
i486: supported up to Win98 and Win NT 4 inclusive, current Linux(!)
Pentium: supported up to Win XP inclusive

# MMX

The Pentium MMX (1997) was a minor revision to the Pentium which included Multi Media eXtensions. These reused the large floating-point registers to store vectors of integer data, and introduced operations to act on all elements of those vectors simultaneously.

The target applications were photographic filters (e.g. Photoshop), and graphics effects in games. The idea of add-with-saturate as a single instruction was introduced, which is very useful in graphics calculations. It simply says that if trying to combine two sources into a single 8 bit pixel, one of a brightness of 200, one of 120, then the answer is 255 (maximum possible 8-bit value), and not 64 (addition with conventional wrap-around).

A total of 57 new instructions were added, packing the floating point registers with eight 8 bit integers, four 16 bit integers, two 32 bit integers, or one 64 bit integer.

# MMX trouble

MMX presented Intel with two problems. Firstly, it introduced new instructions. Old code compiled with old compilers would make no reference to those instructions, not use them and go no faster.

Secondly, the new instructions were not ones which it was easy for a compiler to spot and issue.

```
unsigned char *a,*b,*c;

for (i=0;i<8;i++){
  c[i]=a[i]+b[i];
  if (c[i]<a[i]) c[i]=255;
}
```

That loop is something like a single MMX instruction, provided that the compiler spots it. Arguably

```
for (i=0;i<8;i++)
  a[i]=((a[i]+b[i])>a[i])?(a[i]+b[i]):255;
```

would be an even closer match.

# The Answer

‘Buy our new processor: your existing code will run no faster’ is a hard problem, even without the ‘P.S. Would you like to code in assembler, because life is really difficult for our compiler team?’

Intel’s solution was two-fold. Firstly, it encouraged some key applications to adopt MMX rapidly, and to publish wonderful benchmarks on the speedups of factors of three and more.

Secondly, it significantly improved the branch predictor and the L1 caches on the Pentium MMX. By moving from 8KB 2-way associative to 16KB 4-way associative, hit rates improved, and most code saw a speed improvement of 5-10% without recompilation, even if this improvement was nothing to do with the presence of the MMX instructions.

The 8KB cache on the old i486DX2 had been 4-way associative too. With the plain Pentium’s cache just 2-way associative, it was easy to write code which would run faster on the DX2 by deliberately using an access pattern cacheable with 4-way associativity, but not with two-way. The Pentium MMX stopped this trick.

# The Pentium Pro

The Pentium Pro, introduced at the end of 1995, is an often forgotten member of Intel's family. Its successor arrived only 18 months later, and in many ways the Pentium Pro was a step backwards from the Pentium MMX: it lacked the MMX instructions, and its L1 cache size went back to 8KB. Clock speeds varied from 150MHz to 200MHz.

However, it did make two significant advances which all future Intel processors would share.

The L2 cache was now integrated onto the chip (but not yet the die). This meant it could be much faster, as signals did not need to take a long, complicated path from chip through socket onto motherboard with all the potential for electrical interference that implies. It also kept L2 cache traffic off the bus from the processor to the motherboard. The L2 cache size was usually 256KB or 512KB.

# RISC translation

The second major advance of the Pentium Pro was that the core of the CPU was now a RISC core. The x86 instructions were translated into RISC instructions, called micro ops, and then these fed into a RISC core with its instruction issuing logic. The RISC core was a superscalar out-of-order core.

This design leads to a longer instruction pipeline than would be the case for a pure RISC design. The x86 instructions are first decoded into microcode instructions, and then these pass through a second decoding stage, followed by reordering, as expected for a standard RISC core. The Pentium Pro design can decode three x86 instructions per clock cycle, and issue five $\mu$-ops per clock cycle. A single x86 instruction typically produces between one and three $\mu$-ops, but can produce considerably more.

Pure RISC: Fetch, decode, queue/reorder, execute

PPro: Fetch, translate to $\mu$-ops, fetch, decode, queue/reorder, execute

# RISC vs CISC

The i386 had its origins in the 16 bit 8086, released in 1978 and found in the first IBM PC. The main 16 bit contempories of the 8086 were the Z80A (Sinclair ZX80, ZX81, ZX Spectrum, Amstrad PCW) and the 6502 (BBC Model B, Commodore PET and 64, Apple II).

Direct competition came from Motorola's 68K line. This was just a year younger than the 8086, but 32 bit from the outset. It was used in Apple Macs from 1984 to 1994, as well as the Commodore Amiga, Atari ST, Sun UNIX workstations and HP PostScript printers.

But Motorola abandonned the 68K line in favour of the RISC-based PowerPC. RISC CPUs abounded, with a clean, simple design and excellent performance per watt. The Acorn Archimedes (1987) was the based on an early ARM CPU, MIPS had a range of RISC CPUs, used in SGI workstations and DECstations, Sun had SPARC, HP had PA-RISC, IBM had Power, and IBM with Motorola developed PowerPC, used in Macs from 1994 to 2005. Arguably the best was DEC's Alpha CPU, introduced in 1992 as a pure 64 bit CPU.

When the DEC Alpha 21164a at 500MHz was launched in 1997, it was reasonably said that the fastest way to run Intel code was to run it under the FX!32 emulator on the new Alpha, not directly on Intel's fastest Pentium Pro. (Windows NT was available for the Alpha.)

# Pentium II

The Pentium II, introduced in 1997, was a relatively small change from the Pentium Pro. It added the MMX instructions and returned the L1 cache sizes to 16KB each, which removed any advantage that the Pentium MMX had.

It was packaged in a form which fitted into a slot, rather than a socket, which made it easier for Intel to make as it consisted of a processor die (of about 7.5m transistors) and again off-die L2 cache. This sat on what was effectively a tiny circuit-board. The Pentium II ran its L2 cache at just half the CPU speed, rather than full speed as the Pentium Pro had done. On the other hand, it typically had twice as much.

Variants ran at from 233MHz to 333MHz with a 66MHz bus, and later from 300MHz to 450MHz with a 100MHz bus. Core voltages ranged from 2.8V to 2.0V.

(MMX-like extensions were now widespread. The Alpha gained MVI (1996), and PowerPC gained AltiVec (1999).)

# Celeron and Xeon

Intel has used the name ‘Celeron’ for its budget CPUs since the days of the Pentium II. The first Celeron was a Pentium II running at 266MHz or 300MHz, but without any L2 cache.

This was slow, so 128KB of L2 was soon added (compared to 512KB on a real Pentium II). The Celeron line has continued to this day, generally with slight lower clock speeds (core, bus and memory), and smaller caches, than the corresponding mainstream CPUs.

The Pentium II also saw the introduction of the Xeon brand for CPUs aimed at servers.

# Pentium III

The Pentium III was another minor update to the Pentium Pro design. Initial versions continued with a 100MHz bus, and then later versions moved to 133MHz. Clock speeds ranged from 450MHz to 1.4GHz.

Improvements in process technology during the Pentium III's life enabled Intel to move the L2 cache on die. Most Pentium IIIs with on-die cache had just 256KB, which needed more transistors than the rest of the CPU (including its L1 caches).

The original Pentium had a 64 bit bus running at 66MHz, giving a theoretical peak of 528MB/s, and four times as much as the 32 bit 33MHz bus on the i486 or i486DX4. The Pentium III had improved this by just a factor of two.

# Pentium III: SSE

One important change with the Pentium III was the introduction of the SSE instructions. These added eight new 128-bit registers, each containing four single-precision floating point numbers. These new registers required OS support, as they needed to be saved when task switches occured, and the amount of saved state for a process is now 128 bytes bigger.

The instructions were designed for 3D games: they require floating point calculations to perform projections, but do not require the precision that most scientific calculations need. For code which could make use of them, they certainly increased performance, as a full vector of four adds could be started every other clock cycle, and independently four multiplies every other cycle.

# Pentium Pro to Pentium III

| | Pro | II | III |
|---|---|---|---|
| MMX | – | Y | Y |
| SSE | – | – | Y |
| bus MHz | 60–66 | 66–200 | 100–133 |
| core MHz | 150–200 | 233–450 | 450–1400 |

The Pentium Pro has no Celeron version.

All of the above are referred to as being ‘P6’ architecture, and offer a theoretical peak performance of 1 MFLOPS/MHz.

# Pentium 4

The Pentium 4 architecture, introduced in late 2000, was in many ways revolutionary.

Of most immediate interest to scientists, it introduced SSE2. This extended SSE to operate on vectors of two double precision numbers. At last useful for general purpose scientific computing! In a similar fashion to the Pentium III, its execution unit took two clock cycles to absorb a two element double precision add or multiply.

The changes in design were many. In no way was the Pentium 4 part of a continuous development from the Pentium Pro. The idea now was to move to high clock speeds partly by reducing the amount done per clock cycle.

As a minor change, the Pentium 4 returned to a traditional socket shape, rather than the slot of the Pentium II and III. Its L2 cache was on die, so the idea of a processor being a mini motherboard was no longer needed.

# Caching

The Pentium 4 cached not x86 instructions, but rather the $\mu$-ops generated from them. It could cache about 12,000 $\mu$-ops, so, in most loops the overhead of continuously decoding x86 instructions to $\mu$-ops was eliminated on all but the first iteration.

The length of the instruction pipeline, about 10 stages in the Pentium III, became 20 stages. Fortunately the branch predictor was improved, for a mispredicted branch was rather expensive. Later Pentium 4's, the so-called 'Prescott' revision, increased the pipeline length to 30 stages.

# Clock Speeds

The Pentium 4 was introduced at 1.4GHz, at which speed it was not very competitive with the faster Pentium IIIs. By the end of 2001 it reached 2GHz, then in 2002 3GHz versions were released. The final speed reached in 2004 was 3.8GHz, though at the cost of needing to dissipate 115W.

Given such high clock speeds, it is extraordinary that part of the integer unit ran at twice the main clock speed. This enabled two data-dependent integer operations to execute in a single clock cycle (for certain combinations of operation).

Writing a (pointless) benchmark which will execute faster on a Pentium 4 from 2004 than any current Intel processor is not hard.

The bus clock speed was confusing. Still 64 bit, now just 100MHz, but capable of four transfers per clock cycle, so 400MT/s (mega transfers), and known as QDR (quad data rate). At 3.2GB/s, over three times faster than the Pentium III's bus.

# AMD

Whilst the Pentium 4 was Intel's main processor, Intel's rival, AMD, produced the Opteron (and Athlon64) in 2003.

These produced two significant advantages over Intel's Pentium 4, and earlier AMD processors. Firstly the memory controller was integrated onto the CPU die, with the CPU connected directly to the memory DIMMs, and memory traffic no longer on the same bus as I/O (PCI devices). This produced a significant performance increase. The original memory interface on the Opteron was 6.4GB/s (128 bit, 400MT/s), similar to the shared I/O and memory interface on the Pentium 4 (64 bit, 400 to 1066MT/s), but with lower latency.

# AMD64

AMD's big change with the Opteron was to extend Intel's 32 bit architecture to 64 bits. This was done in a particularly wholesale manner, with many significant changes made which had nothing directly to do with the move from 32 bits to 64.

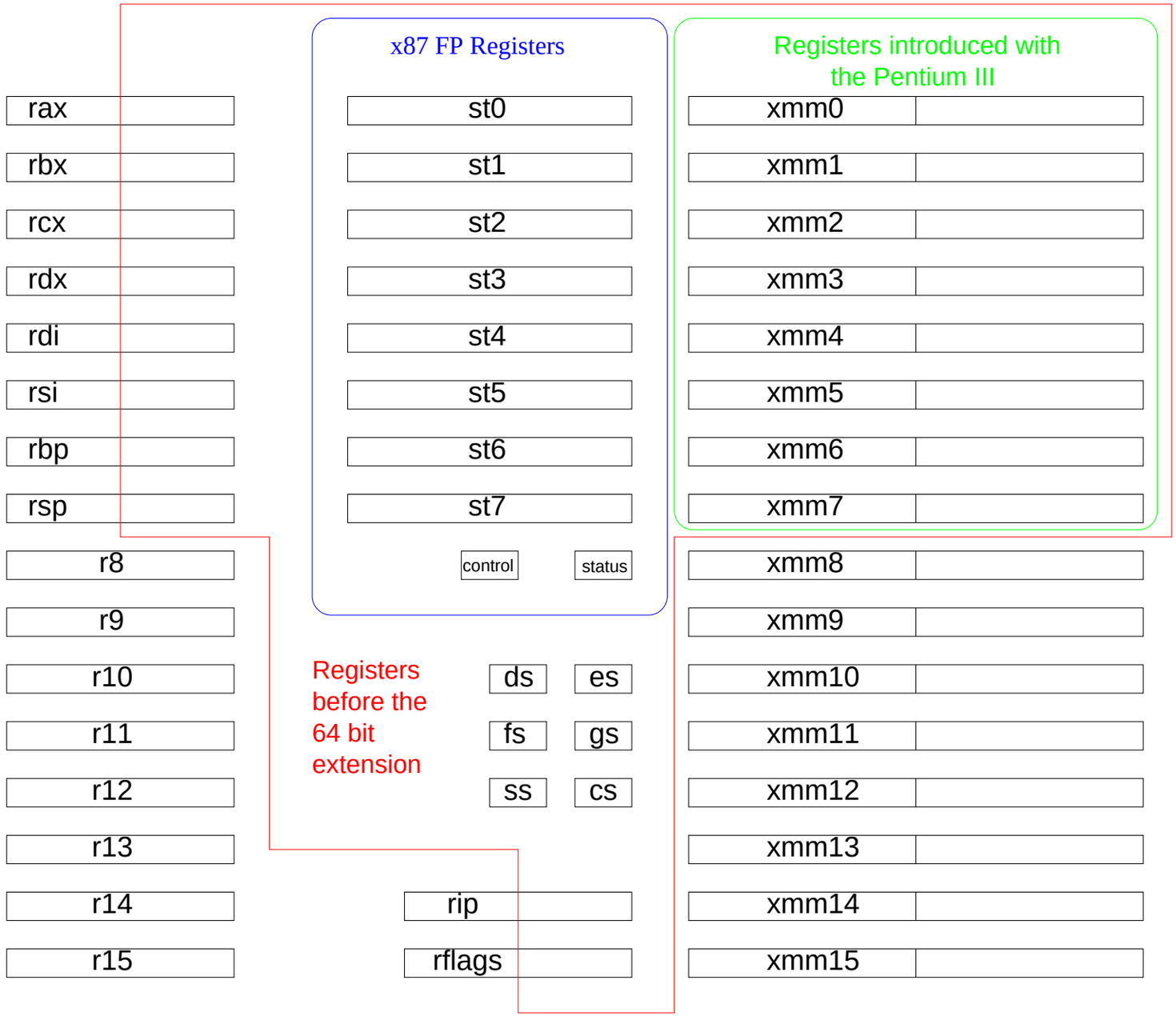
The eight 32-bit integer registers were extended to sixteen 64-bit ones.

The eight SSE registers were also doubled in number to sixteen.

The name was a problem. Although Intel adopted AMD's instruction set, it was never going to call it AMD64, preferring EM64T and then Intel 64. More neutral observers settled on x86-64 (or x86_64).

Intel's first response to AMD's 64 bit coup was to develop the Pentium4 EM64T, which supported the same 64 bit instructions. Intel also slightly extended SSE2 with SSE3, adding a few new instructions, notably "horizontal" operations, so rather than adding the vectors `v` and `u` returning a vector of `v[0]+u[0]` and `v[1]+u[1]` it would return `v[0]+v[1]` and `u[0]+u[1]`, and also a mixed addition and subtraction useful in complex multiplication. It returns `v[0]+u[0]` and `v[1]-u[1]`.

All x86_86 CPUs are guaranteed to have an FPU, and to support MMX, SSE and SSE2. Use of MMX and the old FPU instructions are deprecated in favour of SSE & SSE2.


x87 FP Registers
Registers introduced with the Pentium III
rax
rbx
rcx
rdx
rdi
rsi
rbp
rsp
r8
r9
r10
r11
r12
r13
r14
r15
st0
st1
st2
st3
st4
st5
st6
st7
control
status
Registers before the 64 bit extension
ds
es
fs
gs
ss
cs
rip
rflags
xmm0
xmm1
xmm2
xmm3
xmm4
xmm5
xmm6
xmm7
xmm8
xmm9
xmm10
xmm11
xmm12
xmm13
xmm14
xmm15

# Multiple Cores

Intel and AMD were both having difficulty increasing clock speeds, and yet no difficulty fitting more transistors on wafers. So the obvious answer was to move to dual core CPUs. Intel introduced the dual core Pentium D (based on the Pentium 4) in 2005, beating AMD Athlon64 X2 by a few weeks, but a few weeks behind AMD's dual core Opterons intended for the server market.

Intel's solution was less elegant – two dies in the same chip package, whereas AMD used a single die.

# Goodbye, Pentium 4

Intel's next processor, the Core, introduced in 2006, abandoned the architecture of the Pentium 4, and looked much more like modified Pentium III with SSE2 added. It did keep something very similar to the Pentium 4's bus.

The Pentium 4 had proven to be very power hungry, and Intel had not been able to get it up to the clock speeds it had hoped. The Pentium M, another development of the Pentium III, had been produced in 2004 for notebooks, offering better performance per watt than the Pentium 4, and the Core was an extension of this.

Surprisingly the Core remained 32 bit only – it did not support the EM64T extensions. More surprisingly, Apple switched from PowerPC to this CPU for the first Intel-based Macs.

# Core 2

The Core 2 was released only about seven months after the Core, and it regained the 64 bit extensions. It was the first Intel processor not to have a single core varient for desktops – it was introduced as a dual core processor, and soon moved to quad core.

Its clock speeds were more modest than the Pentium 4, ranging from about 1.8GHz to 3.3GHz.

In terms of theoretical peak performance, each core could now execute two element add and multiply instructions independently on every clock cycle, so four FLOPS per Hz – twice performance of the Pentium 4. It also increased the size of the L1 caches to 32KB each.

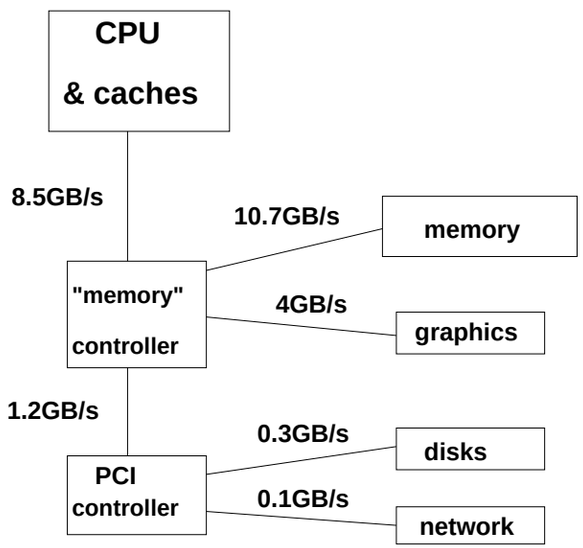
Late versions of the Pentium 4 had introduced Speedstep to the desktop, lowering frequency and voltage when idle, but not very aggressively. Their minimum frequency was 14$\times$ the bus frequency, generally at least 200MHz, so 2.8GHz. The Core 2 was more aggressive, dropping to 6$\times$ bus frequency, often meaning 1.6GHz.

# Slow buses

The single external bus on the Pentium 4 and Core / Core2, carried all memory and I/O traffic (disk, network, graphics card), and was 64 bits wide running at a range of disappointing speeds.

Disappointing? The dual core 2.13GHz Core 2 E6420 I have in front of me ran it at 1066MT/s. It took dual channel DDR2/667 memory, so the memory was capable of providing 10.67GB/s, but the bus carrying all memory and I/O traffic to the CPU was capable of just 8.5GB/s. Later varients increased the bus speed to 1333MT/s, but also increased the memory speed to 800MT/s.

The last Core 2s moved to using DDR3/1066 memory, with the CPU bus still running at just 1333MT/s. So 17GB/s of potential from the memory throttled to 10.67GB/s by the bus to the CPU, and this one bus shared by all CPU cores. (E.g. 2.83GHz quad core Q9550 Core 2 from 2008.)



# And Slower

Dual socket Core 2 designs were yet worse. Because both CPUs shared the same bus connecting them to each other and the combined memory and I/O controller, the bus was physically longer with more connections than on a single socket machine. So it tended to run slightly slower.

So a dual socket Core 2 based machine often had less total memory bandwidth than a single socket Core 2 based machine, which was in turn in general beaten by AMD's Athlon64 which had placed the memory controller on the processor die with the memory connecting directly to it, and a separate I/O bus.

A dual socket Core 2 machine I had access to at the time had four channels of DDR2/667 memory – 21.3GB/s – but feeding into a 1333MT/s bus capable of just 10.67GB/s. It is little wonder that a standard memory benchmark achieved just 6.4GB/s.

# Nehalem

Intel's next CPU, codenamed Nehalem, was introduced in 2008. Its big advance was to move the memory controller onto the CPU die, as AMD had done three years earlier. Many varients had a three channel memory controller, rather than the usual two channel.

The other feature of the separate 'memory' controller (also known as the north bridge) of the Core2 was to provide a fast PCIe bus to the graphics card. This feature was also incorporated directly into the Nehalem CPU. A PCIe controller with just 16 PCIe lanes is built into the CPU. There is still an external PCIe controller, connected via a different interface, for other PCIe things (disks controllers, network controllers, etc.)

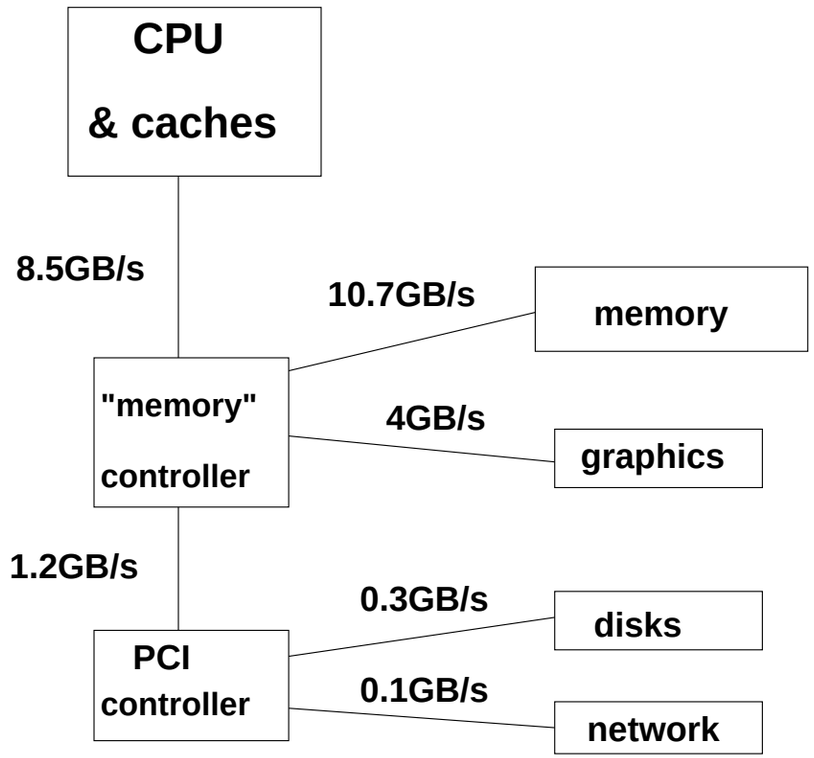
The biggest jump was for dual socket machines, which now had two independent memory buses. The Nehalem equivalent of the dual socket Core 2 machine on the previous slide had six channels of DDR3/800 memory. Theoretically 38.4GB/s, with a measured performance of 26GB/s, over four times what the Core 2 based machine achieved.

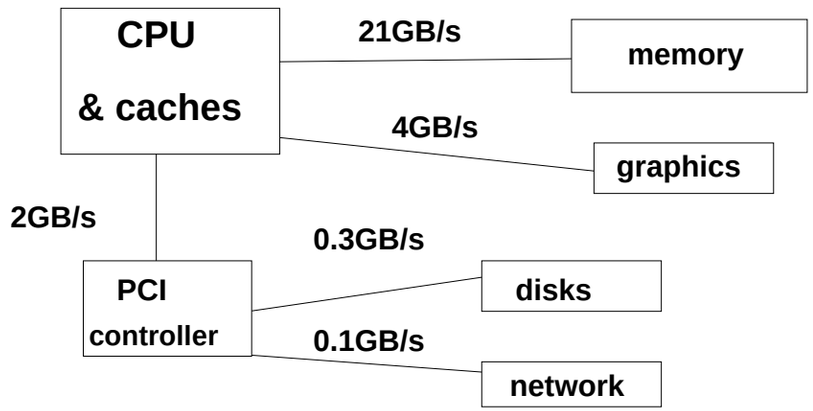
The peak floating point performance remained four FLOPS per Hz, but in general the Nehalem performed much better, mostly due to its increased memory bandwidth.

# Other Nehalem changes

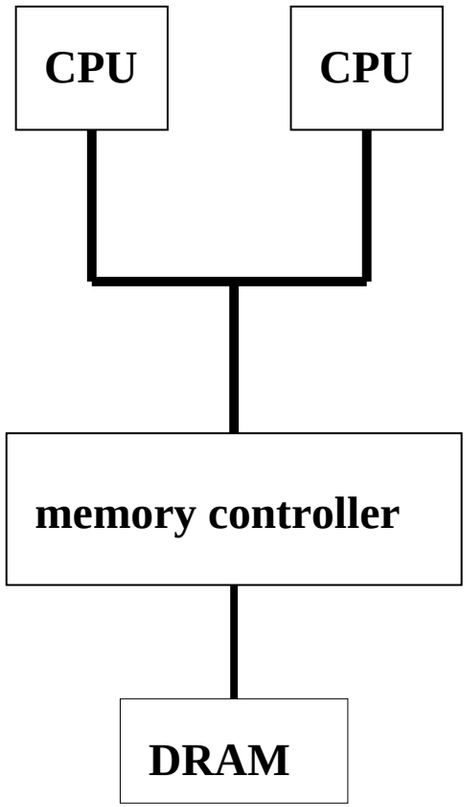
The Nehalem introduced the idea of an smallish (256KB) L2 cache for each core, unified between instruction and data, with the L3 cache being shared by all cores.

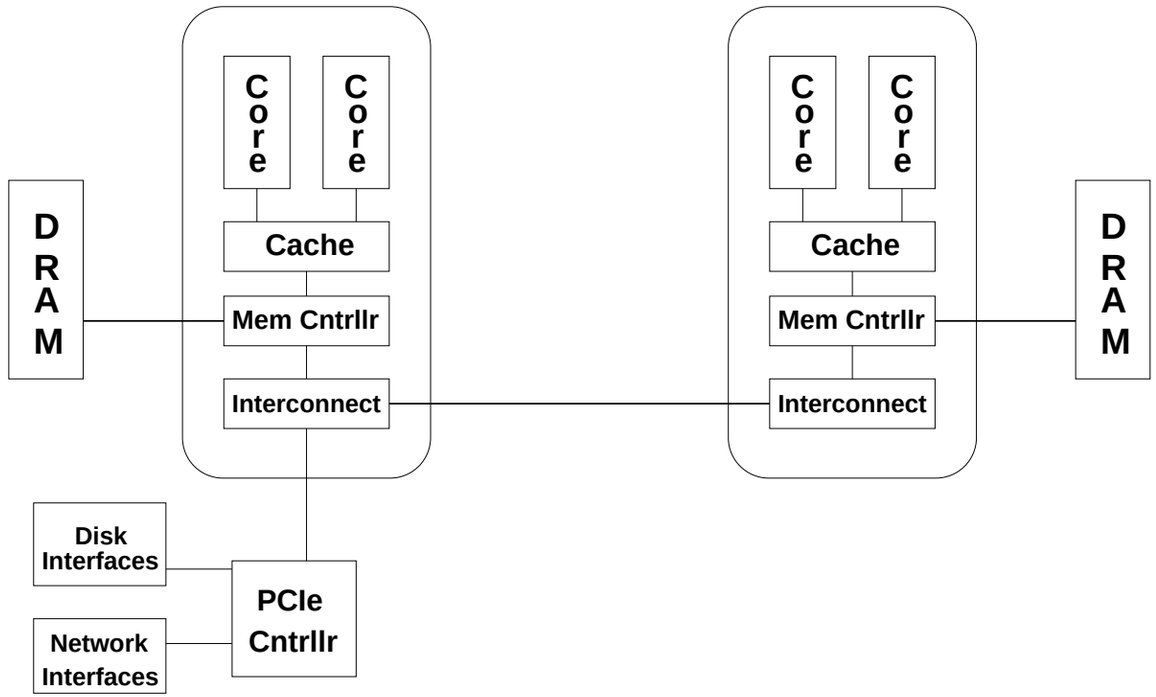
It introduced turbo clock speeds – raising the clock speed if busy but not too hot, particularly useful if just a single core is active.



# SMP


CPU
CPU
memory controller
DRAM


DRAM
Core
Core
Cache
Mem Cntrllr
Interconnect
Core
Core
Cache
Mem Cntrllr
Interconnect
DRAM
Disk Interfaces
Network Interfaces
PCIe Cntrllr

# Names

Nehalem also introduced the Core i5 and i7 names. The cheapest CPUs were still called Celeron, but the others were divided into i5, i7 and i7 Extreme. These names persist today, with i3 and i9 added.

It is very confusing. The original Core i7 Extreme was launched in 2008, and was a four-core 3.2GHz Nehalem with 8MB of LLC, three channels of DDR3/1066 memory and dissipating 130W.

A more recent Core i3, or even Celeron, would run rings around it, whilst using a fraction of the power.

The Pentium name was also revived, and fits between the Celeron and the bottom of the Core i series.

The Xeon name continued for server-class CPUs. Now that the memory controller was part of the CPU, not the motherboard, it was important to note that Xeons supported ECC memory, and non-Xeons did not. (In contrast, AMD CPUs do generally support ECC.) The core-count of Xeons started to increase significantly beyond the standard four of desktop CPUs. It reached ten with the Nehalem.

# AVX and AVX2

The next major change after Nehalem was the introduction of AVX. This doubled the size of the vector registers, and the associated functional units, so that they could operate on 256-bit vectors, rather than 128-bit ones. Floating point vector instructions could now operate on 1, 2 or 4 doubles, or 1, 2, 4 or 8 singles.

AVX was introduced with the Sandy Bridge, and Ivybridge was a fairly minor redesign. Assuming one used the new instructions, and could use a vector length of four, the peak performance was now eight FLOPS per Hz per core.

The next redesign, Haswell and Broadwell, introduced a fused multiply-add (FMA) instruction (`a*b+c` as a single instruction), and could issue two of these every clock cycle, so a peak of sixteen FLOPS per Hz per core. They also improved idle power management, with a lowest clock speed of 800MHz, compared to 1.6GHz as it had been since the Core 2.

# Lakes

Intel's current CPUs, Skylake, Kaby Lake, Coffee Lake, Cannon Lake, Whiskey Lake and Comet Lake are very confusing. Some, such as Skylake, existed in the full range of Celeron, Pentium, i3, i5, i7 and i9 brand-names.

Intel has again doubled the vector length to introduce AVX-512. But it has done so in a very fragmented fashion.

Laptop and most desktop CPUs simply do not support AVX-512 at all.

Some low-end server CPUs support the instructions, but have no dedicated AVX-512 units. Their peak FLOPS per Hz per core remains sixteen, and any AVX-512 instructions must pass through both AVX-256 units.

The high-end CPUs do have a pair of genuine AVX-512 execution units, and thus can achieve 32 FLOPS per Hz per core.

Most, but not all, of the CPUs with two AVX-512 units need to reduce their clock-speed to below their 'headline' speed when these are active. One of the worse offenders is the Xeon Scalable 6128, six cores and nominally 3.4GHz (3.7GHz turbo), but only 2.9GHz as a maximum turbo frequency when all cores are active with AVX-512 instructions, and just 2.3GHz as a non-turbo frequency.

# Core 2 to Lakes

| | Core 2 | Nehalem | Sandy/Ivy Bridge | Haswell | Lakes |
|---|---|---|---|---|---|
| Year | 2006 | 2008 | 2011 | 2013 | 2015 |
| Mem ctrl | – | Y | Y | Y | Y |
| Vector length | 2 | 2 | 4 | 4 | 4 or 8 |
| FMA | – | – | – | Y | Y |
| MFLOPS/MHz | 4 | 4 | 8 | 16 | 16 or 32 |
| Memory | – | DDR3/1066<br>DDR3/1333 | DDR3/1333<br>DDR3/1600 | DDR3/1600<br>DDR4/2133 | DDR4/2133<br>to DDR4/2933 |
| Cores (desktop) | 2–4 | 2–4 | 2–4 | 2–8 | 2–8 |
| Cores (server) | 2–4 | 2–10 | 2–15 | 2–18 | 2–56 |
| MHz | 1.8–3.3 | 1.8–3.4 | 1.8–3.6 | 2.0–4.0 | 2.0–4.2 |

All 64 bit, all with SSE2.

# Little changes

What makes a Coffee Lake faster than a Nehalem when executing non-AVX code?

Memory bandwidth and latency.

Cache bandwidth, latency and size.

Branch predictor improvements.

Number of x86_64 instructions reduced to RISC-like $\mu$-ops per cycle.

Number of $\mu$-ops issued per cycle.

Degree of speculative execution past unresolved jumps.

Degree of out of order execution to avoid dependencies.

# Compatibility

If one runs code compatible for the Nehalem (or earlier) on a Haswell, it will not achieve more than four FLOPS per Hz. The only way of achieving more is to use the extra length in the vector registers, and to use the new FMA instruction.

There are three choices: abandon performance, abandon compatibility with older processors, or embrace code which executes different instructions depending on which processor it is running on. The latter route is sanest, but can lead to different bugs appearing depending on the processor one executes on.

# Shared Libraries

Ideally everyone using Linux would agree on one set of libraries to use to cover the common, speed-critical, operations such as FFTs and Lapack. By linking these as shared libraries, one can then rely on the machine on which the code is executed having a version optimised for it which will be called automatically.

Unfortunately things don't quite work like this, especially as the maths libraries produced by Intel, which are reasonably fast, are not freely distributed, and have themselves had bugs in the past.

It is therefore quite likely that software purchased/installed a couple of years ago does not make full use of the latest processors. This is particularly true for AMD-based computers.

# AMD's dilemma

Much software does not use AVX / AVX2 / AVX512 for the above reasons.

Even more software fails to use AVX in any form on AMD CPUs, as libraries produced by Intel often deliberate fall back to plain SSE2 on any AMD CPU. How much should AMD worry about AVX support?

So AMD's Zen range of CPUs supported the AVX2 instruction set, but the functional units could process just two doubles at once. Instructions using the full width of an AVX2 register had to be issued in two halves, and its peak performance is 8 MFLOPS/MHz. This is the same trick that Intel does with AVX-512 on some CPUs.

AMD's Zen 2 range has full-width AVX units, and is 16 MFLOPS/MHz. This makes it much more competitive for scientific work. Zen 2 is the basis of Archer 2, UK academia's leading supercomputer being installed in 2021.

# Sockets

Intel's 'standard' uniprocessor CPUs currently use sockets of around 1150 pins. These have a 128 bit memory bus, and around 16 PCIe lanes into the lower-latency PCIe controller integrated onto the CPU.

Thus the Xeon E3-12xx and most Core-i7 processors.

There are rarer, high-end varients using sockets of around 2,000 pins. These have 256 bit memory buses, and 40 or more PCIe lanes. Thus the Xeon E5-16xx, Core i7 Extremes, i9, and a few others. This configuration is much more common with dual socket machines (Xeon E5 and E7).

The high number of PCIe lanes going to the low-latency on-CPU PCIe controller is wanted by gamers using two video cards, and HPC people wanting to use two GPU cards, or a GPU card and an Infiniband card. The doubled memory bandwidth is wanted by almost anyone once one has more than four cores per CPU, as most of these CPUs do.