CPU performance summary

Cache memory Vs RAM

RAM = Random access memory the main memory of a computer which is on flash chips usually external to the CPU on the motherboard. This memory is volatile so only retains data while power is flowing through it. Memory is needed to store the parts of the operating system, instructions and data of programs that is currently in use.

Data and instructions must be transferred to and from the CPU and memory during the fetch-decode-execute cycle via the data bus. Addresses are also sent over the address bus and control signals over the control bus. The transfer rate across such buses is far slower than that of the transfer rates that occur within a CPU.

Often the rate of the transfer means that the CPU is left stationary while waiting for data to be fetched. Moreover, instruction and data fetches share the same data bus so must be scheduled (they cannot occur synchronously). This is known as the **Von Neumann Bottleneck.**

Memory being separated from the CPU results in latency due to transfer of data over buses.

How do modern CPUs avoid such latency of the von Neumann bottleneck?

Caching = the storing of frequently used data in a special area (usually RAM) so that it is more readily accessible than if it were stored in the main memory.

Prefetching = moving some data into the Cache (local memory in the CPU) before it is requested so the access speed is faster (similar to the speed of CPU).

Cache = A small amount of random access memory located within the CPU's integrated chip (or on a separate nearby chip with a dedicated bus to CPU) so that the microprocessor can retrieve data at a fast rate. Frequently used data/instructions are stored here.

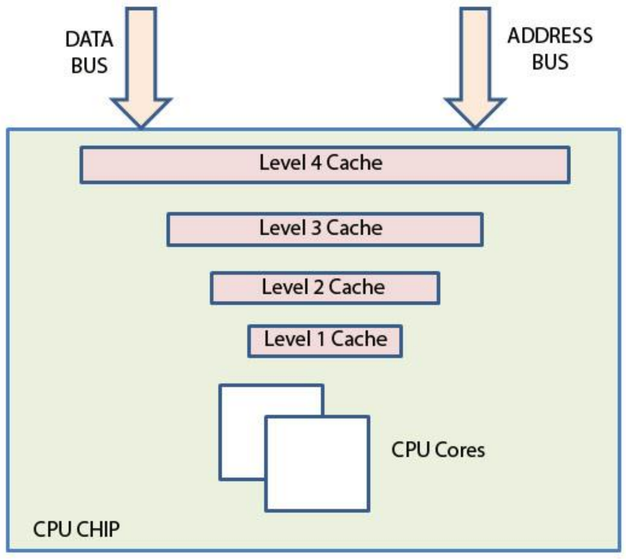
Since cache memory is so expensive to implement, generally only small amounts are incorporated into a CPU. Cache is managed using Harvard architecture to improve efficiency.

Arranging the Cache into a hierarchy of levels also helps separate the hierarchy of commands. Data that needs to be accessed faster can be stored in faster levels of cache with limited capacities.

The cache closest to the CPU cores is the fastest (also smallest capacity), this is level 1 cache.

Level 1 cache is serviced by level 2 cache, slightly slower but large capacity.

The idea is to match the speed of the instruction and the data flow to the speed of the CPU to maximise performance. The cache can be level 1, 2, 3 (or even 4). Higher levels are large in capacity but slower. Instructions are transitioned through the caches corresponding to their speed of retrieval requirement.



Clock speeds

Clock = The CPU is controlled by an internal `clock' signal. The clock is an A-stable operation so is an alternating square wave of logic 1 (true) and logic 0 (false) that cycles over a period of time. The period is called the frequency and is measured in Hz or cycles per second (or more commonly GHz).

One clock cycle is the equivalent of one fetch-decode-execute cycle. Therefore a high clock speed means more fetch-decode-execute cycles per second resulting in a higher computer performance. Data can be calculated and manipulated in less time.

Problem of the bottleneck

However, a higher clock speed is still bottlenecked by the Von Neumann Bottle architecture. It generally means that the CPU will spend more idle time waiting for retrieval of data over the buses.

Power consumption (heat/battery)

Higher clocks also result in a higher power consumption. Every time the clock cycles, power is consumed. An increased clock speed results in the transistors within the processor having to switch faster, which generates more heat and consequently uses more power.

Much beyond 4.0Ghz and the CPU would overheat melting the chip on a standard cooler.

More power also results in decreased battery life when referring to mobile devices or a larger annual energy consumption for desktops, more expensive bills.

Chip designers make portable CPUs with a lower clock speed to reduce power consumption.

Reliability

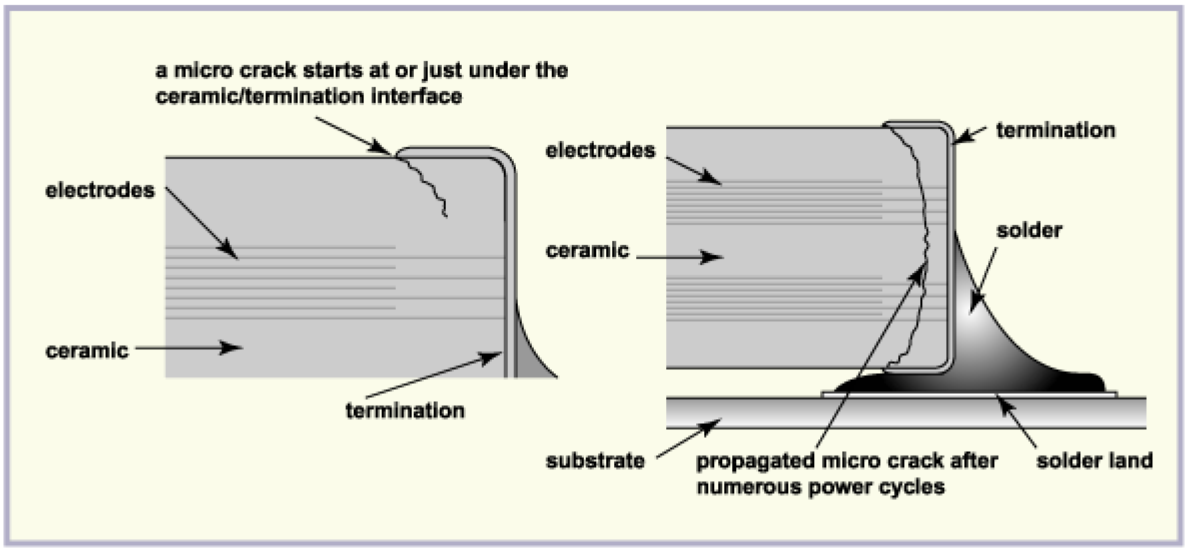
There is a fundamental relationship between heat and reliability. The hotter the chip gets, the lower the average life of the chip since stress on components low down is caused by temperature. A slight increase in temperature can have an exponential decrease in life expectancy.

Chip failure occurs due to the large temperature gradients by which the semiconductor must cool through. Higher temperatures cause more expansion of silicon wafers.
As they cool down to ambient temperature, they contract. An isotropic stress regime is set up creating microcracks in the silicon wafer. Over time, the cracks will get larger and eventually cause the semiconducting component to fail. For this reason, investing in good coolers and case fans to keep the overall temperature within the case lower (especially the motherboard) is a good idea – even if you have no plans of overclocking; increasing the life expectancy of components significantly.

Motherboards are most prone to heat dame. CPU failure is very rare nowadays since it is just a singular solid-state component with a high quality of manufacture on a nanoscale.

Graphics cards and motherboards consist of thousands of solid-state components and solder joints so a far more likely to fail. Hard drives being mechanical are most prone to failure but this is unrelated to temperature.

Micro cracks are common in ceramic components.


a micro crack starts at or just under the ceramic/termination interface
electrodes
ceramic
termination
electrodes
ceramic
termination
solder
substrate
propagated micro crack after numerous power cycles
solder land

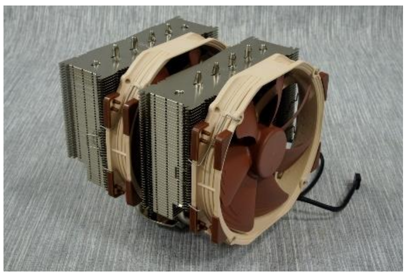
Chip protection (adaptive clocks and thermal throttling)

CPU manufacturers protect the chips from heat damage, increasing the CPUs reliability and reducing power consumption by using adaptive clocks and thermal throttling. When the CPU is handling less strenuous tasks, a lower clock speed is implemented creating fewer clock cycles per second, using less power.

When more data/harder tasks need to be accomplished, the clock speed is sped up. If the clock speed did not increase for such tasks then latency would become noticeable.

Conversely, when the CPU reaches a temperature too high, thermal throttling reduces the clock speed in an attempt to reduce the power consumption and so heat output.

CPUs will also have a large heat sync which has been thermodynamically engineered to dissipate heat as efficiently as possible. The structure has a large surface area and maximum contact with the CPU and the surrounding air. Metals such as aluminium or copper are used as they have a high thermal conductivity. Copper, being more expensive, only tends to be used in the most needed parts of the cooler such as the plate in direct contact with the CPU. Thermal paste is applied as it fills in the spaces that exist between the heatsink and CPU that may not be visible since they are on a microscopic scale. The plane of the heatsink base is not truly flat so is never in full contact with the CPU unless thermal paste is used. This increases heat transfer to the CPU, minimising micro convection currents.

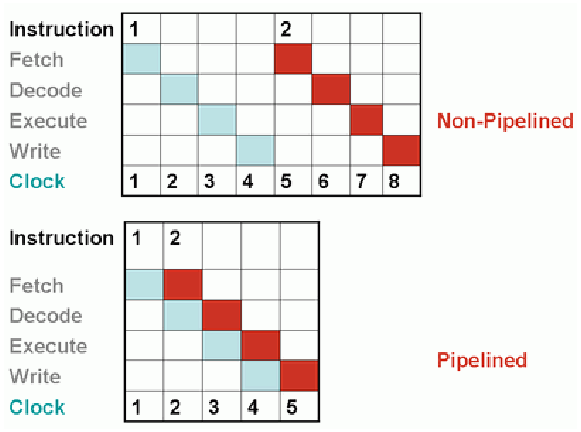


Pipelining

Pipelining is a technique that CPUs can use which allows them to handle multiple tasks more efficiently.

One instruction can begin its fetch-execute-decode cycle at the same time as another instruction is being processed and hasn't completely finished it's fetch-decode-execute (machine) cycle. This uses the CPU resources more effectively and so multiple instructions are split into different stages of their fetch-decode-execute cycle. This is common for Harvard architecture.

This results in a given amount of instructions being completed in a fewer amount of clock cycles. It does not speed up the time in which a single instruction is carried out.



Non-Pipelined

Instruction	1				2			
Fetch	🟦				🟥			
Decode		🟦				🟥		
Execute			🟦				🟥	
Write				🟦				🟥
Clock	1	2	3	4	5	6	7	8

Pipelined

Instruction	1	2			
Fetch	🟦	🟥			
Decode		🟦	🟥		
Execute			🟦	🟥	
Write				🟦	🟥
Clock	1	2	3	4	5

However, Pipelined CPUs are more expensive to manufacture and need more complex ALUs.

Multi-core CPUs

Traditionally, CPU performance was increased by finding new, ingenious ways of fitting more and more transistors onto the same size chip. There was even a rule called Moore's Law, which predicted that the number of transistors that could be placed on a chip would double every two years.

The multiple-core processor is the term given to a CPU containing two or more independent actual processing units called cores.

Manufacturers integrate multiple cores (independent processors) onto a single integrated chip, the CPU.

A CPU with multiple cores results in multi-tasking as independent processors can work simultaneously to process different instructions. Data can be fetched—decoded and executed simultaneously. Moreover, the operating system can make use of allocating instructions of applications to different cores so a workload can be completed in less time.

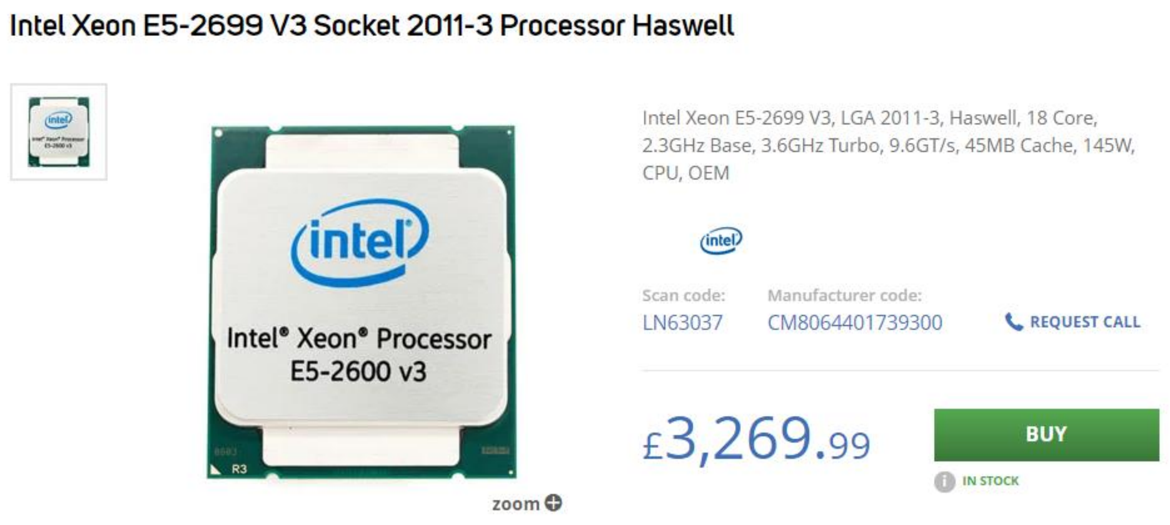
Compromises and problems

The performance increase to the number of cores is not linear. This is because even today, many applications are coded to make use of fewer cores. More complex programs and operating systems are required for multiple cores and so many programs can't make use of the extra resources as they can't keep up with the technologies of vast core numbers.

The power consumption and so heat output of the CPU is greater as a multi-core CPU has more cores (independent processors) each running inside the integrated chip, carrying out clock cycles that tick every second. Each tick results in power consumption so more cores result in more power being consumed.
Cores communicate to each other through different channels, this is allocated some of the clock speed so performance increase is not linear.

There is also a huge amount of additional complexity involved in implementing parallel processing, because when each separate core (processor) has completed its own task, the results from all the cores need to be combined to form the complete solution to the original problem.

In parallel processing, two or more processors work together to perform a single task. The task is split into smaller sub-tasks (threads). These tasks are executed simultaneously by all available processors (any task can be processed by any of the processors).



Intel Xeon E5-2699 V3 Socket 2011-3 Processor Haswell

Intel Xeon E5-2699 V3, LGA 2011-3, Haswell, 18 Core, 2.3GHz Base, 3.6GHz Turbo, 9.6GT/s, 45MB Cache, 145W, CPU, OEM

Scan code: LN63037

Manufacturer code: CM8064401739300

£3,269.99

IN STOCK