Asynchronous I/O

Asynchronous I/O is a form of input/output processing that allows the CPU to continue executing concurrently while communication is in progress.

Disk and network are orders of magnitude slower than the CPU. Rather than initiating an I/O operation and waiting for the result, the CPU can initiate the I/O operation as soon as it's known that the data will be needed, and continue executing other instructions which do not depend on the result of the I/O operation.

Some rough comparisons for illustration purposes:

Device Latency Bandwidth CPU L3 Cache 10 ns

> 400 GB/s Memory 100 ns 100 GB/s Disk (NVMe SSD) 400 us 380 MB/s Network 50 - 200 ms 1 Gb/s (125 MB/s) (actual disk stats as reported by fio for random reads of size 2KB - ~190k IOPS)

Fortunately, SSD drives can perform operations concurrently, so the CPU can initiate several requests at the same time, continue executing, and then receive the results of multiple operations around the same time.

Some databases (such as lmdb / mdbx) use memory-mapped storage to read and write to disk. Unfortunately, memory-mapped storage is implemented by the kernel (mmap) and is not asynchronous, so execution is blocked while waiting for the operation to complete.

More about asynchronous i/o can be read here .

Previous Pipelining Next Why blockchain? Last updated5 months ago

On this page