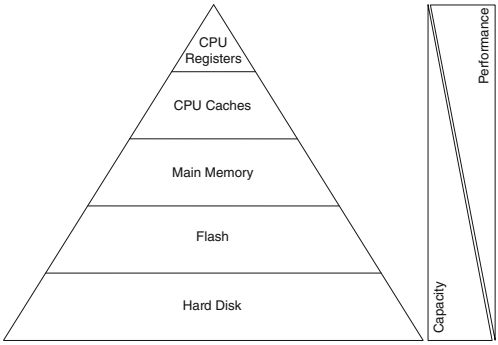


**Fig. 4.3** Conceptual view of the memory hierarchy

**Table 4.1** Latency numbers

| Action | Time (ns) | Time |
|---|---|---|
| L1 cache reference (cached data word) | 0.5 | |
| Branch mispredict | 5 | |
| L2 cache reference | 7 | |
| Mutex lock/unlock | 25 | |
| Main memory reference | 100 | 0.1 µs |
| Send 2,000 byte over 1 Gb/s network | 20,000 | 20 µs |
| SSD random read | 150,000 | 150 µs |
| Read 1 MB sequentially from memory | 250,000 | 250 µs |
| Disk seek | 10,000,000 | 10 ms |
| Send packet CA to Netherlands to CA | 150,000,000 | 150 ms |

The directly accessible main memory is located between Flash and the CPU caches L3, L2 and L1. The CPU registers, where data needs to be located to be used in actual calculations, form the top of the memory hierarchy. As every operation takes place inside the CPU and in turn the data has to be in the registers, there are usually four layers that are only used for transporting information when accessing data from disk.

Table 4.1 gives an overview of some of the latencies related to the memory hierarchy. Latency is the time delay that has to be taken into account for the system to load data from the respective storage medium into the CPU registers. The L1 cache latency is 0.5 ns. In contrast, accessing a main memory reference takes 100 ns and a simple disk seek accounts for a 10 ms delay.

In the end, there is nothing special about the main-memory based approach for database management systems. All computing ever done was in memory as actual calculations can only take place in the CPU. The performance of an application using large amounts of data is usually determined by how fast data can be transferred through the memory hierarchy to the CPU registers. To estimate the runtime of a typical database operator algorithm, it is therefore possible to roughly calculate