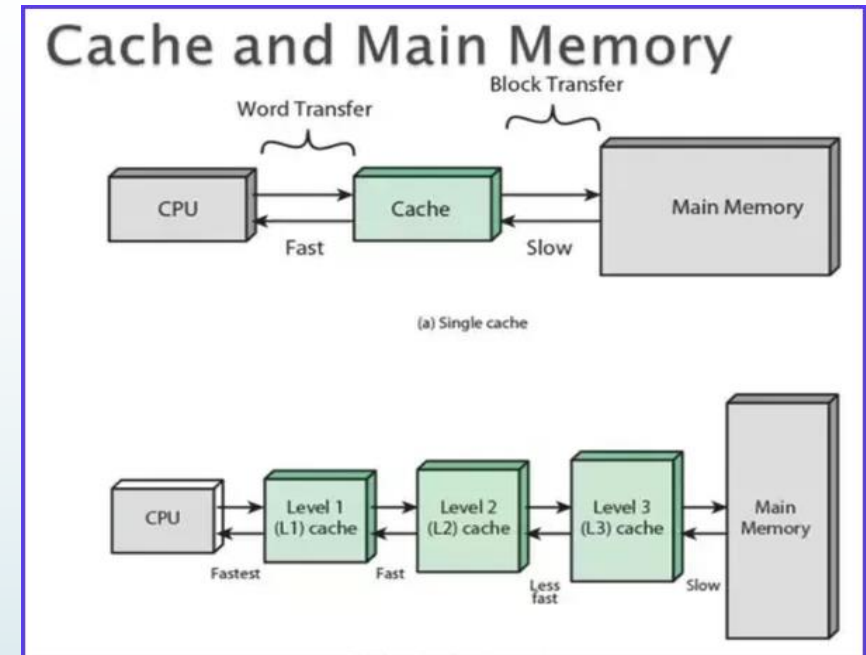



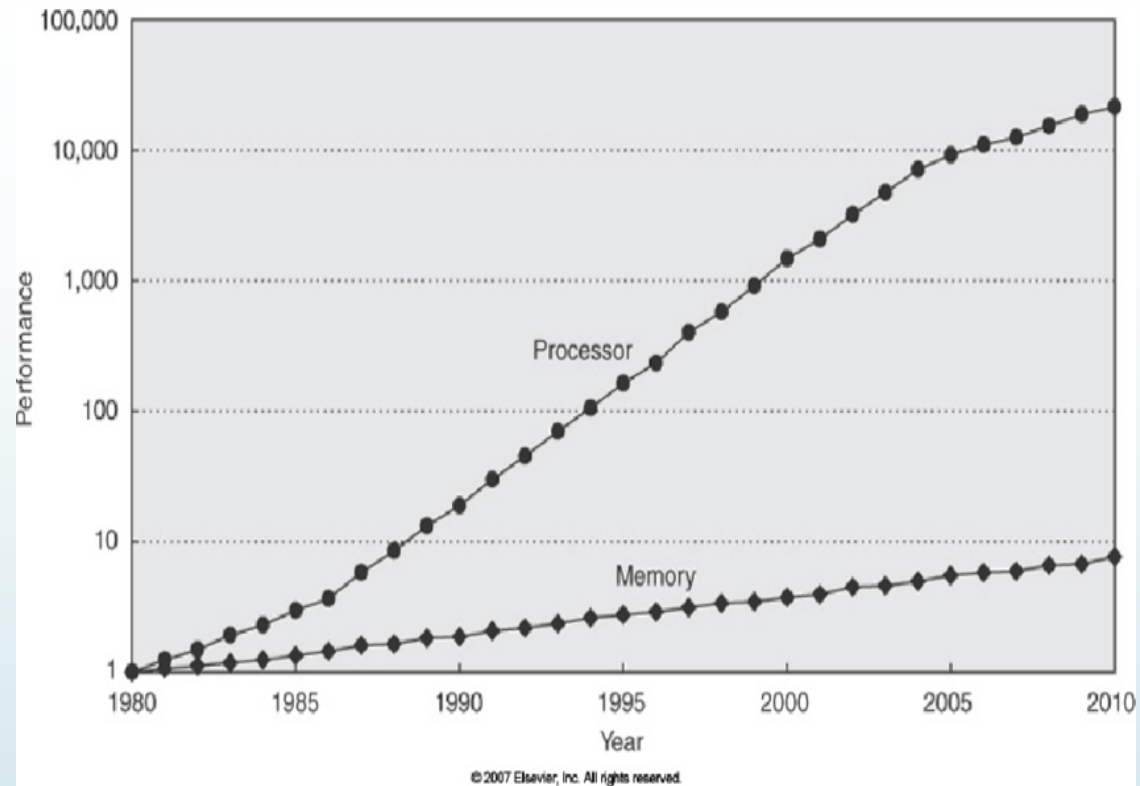
Earliest cache Design

- The cache memory is a very high speed, expensive piece of memory, which is used to speed up the memory retrieval process.
- Due to their higher costs, the CPU comes with a relatively small amount of cache compared with the main memory.
- Example-**The Intel 80486** microprocessor, contains an 8K memory cache, and **the Pentium** has a 16K cache.



- Memory cache was first used on PCs at the **386DX** timeframe. **The Intel 80386**, also known as **i386** or just **386**, is a 32-bit microprocessor introduced in 1985.
- The amount of available memory cache varied as well depending on the motherboard model and typical values for that time were 64 KB and 128 KB. At this time the memory cache controller used an architecture known as “write-through,” i.e., when the CPU wants to store data in memory – the memory cache controller updates the RAM memory immediately.

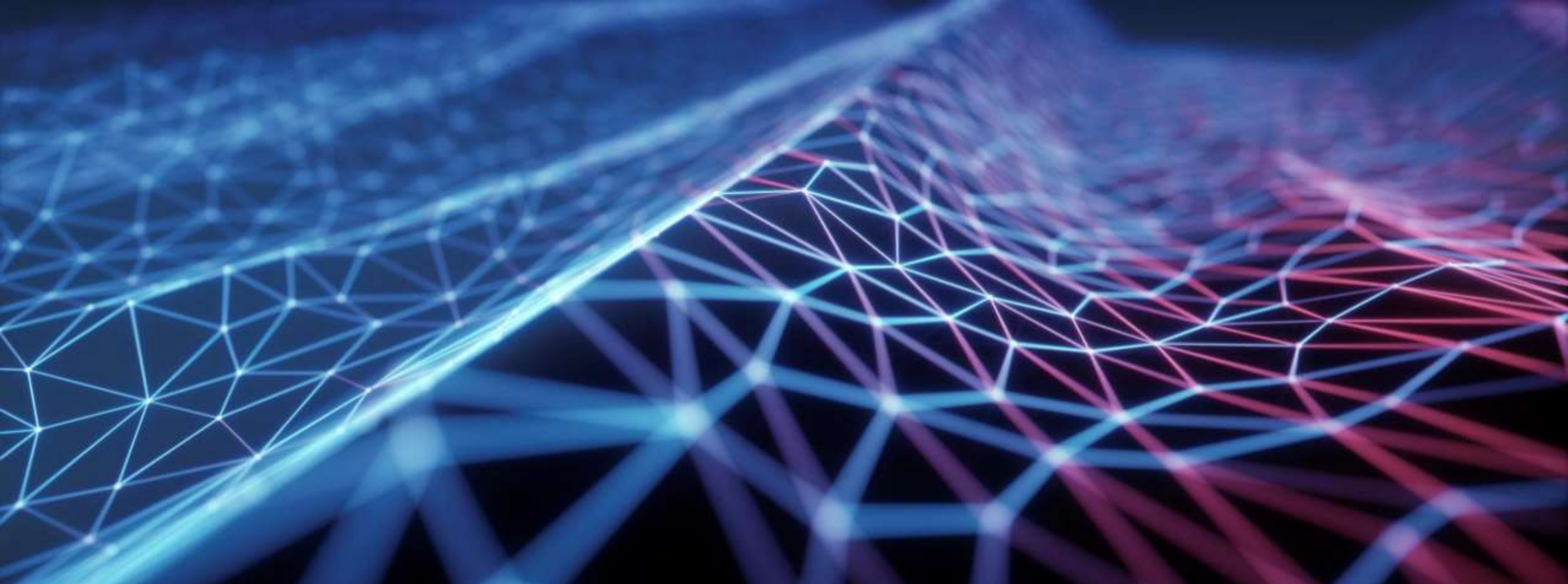
- 
- In 1989 when Intel first put memory on the CPU with the 80486, a small amount (8 KB) of memory cache inside the CPU. This internal memory cache was called L1 (level 1) or “internal,” while the external memory cache was called L2 (level 2) or “external.” Typical amounts for that time were 128 KB and 256 KB.
 - Later 486 models added the “write back” cache architecture, which is used until today, where for write operations the RAM memory isn’t updated immediately, the CPU stores the data on the cache memory and the memory controller updates the RAM memory only when a cache miss occurs.
 - With the first Pentium processor (**Pentium-200**), Intel created two separated internal memory caches, one for instructions and another for data (at the time with 8 KB each). This architecture is still used to date, and that is why you sometimes see the L1 memory cache being referred as 64 KB + 64 KB, for example – this is because there are one 64 KB instruction L1 cache and one 64 KB data L1 cache.
 - The problem with the L2 memory cache being external is that it is accessed with a lower clock rate, because since 486DX2 the CPU internal clock rate is different from the CPU external clock rate. While a Pentium-200 worked internally at 200 MHz, it accessed its L2 memory cache at 66 MHz, for example. Then with P6 architecture Intel moved the memory cache from the motherboard to inside the CPU – what allowed the CPU to access it with its internal clock rate .
 - This same architecture is used until today: both L1 and L2 memory caches are located inside the CPU running at the CPU internal clock rate.



Moore's Law states that we can expect the speed and capability of our computers to increase every couple of years, and we will pay less for them.



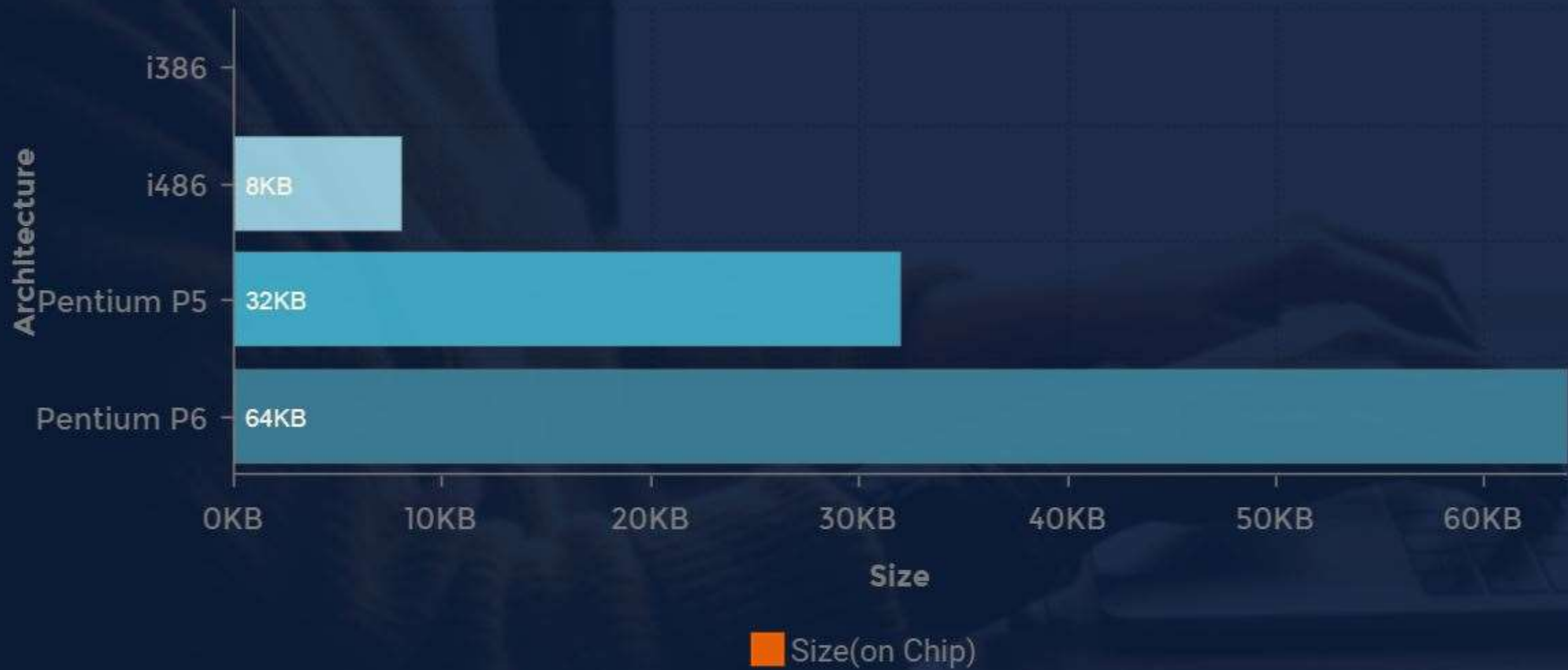
Thank You...



Early Cache designs

And Improvements

On chip Cache Size



Introduction of L2 Cache with P6

- P6's most noticeable addition was its on-package **L2 cache**, which ranged from 256 KB at introduction to 1 MB in 1997
- The cache was also "non-blocking", meaning that the processor could issue more than one cache request at a time (up to 4), reducing cache-miss penalties. These properties combined to produce an L2 cache that was immensely faster than the motherboard-based caches of older processors.
- This cache alone gave the CPU an advantage in input/output performance over older x86 CPUs.

Separate Data and Instruction Cache

- Separate data Cache and instruction Cache made it possible to fetch instructions and data in parallel.
- Reduced miss rate of data cache as compared to unified cache
- The split design is useful for pipelined processors where the instruction fetch unit and the memory access unit are physically located in different parts of the chip. With the unified design, it is impossible to place the cache both close to the instruction fetch unit and the memory unit, resulting in high cache access latency=



Thank You

Ishaan Yadav

CACHE EVOLUTION

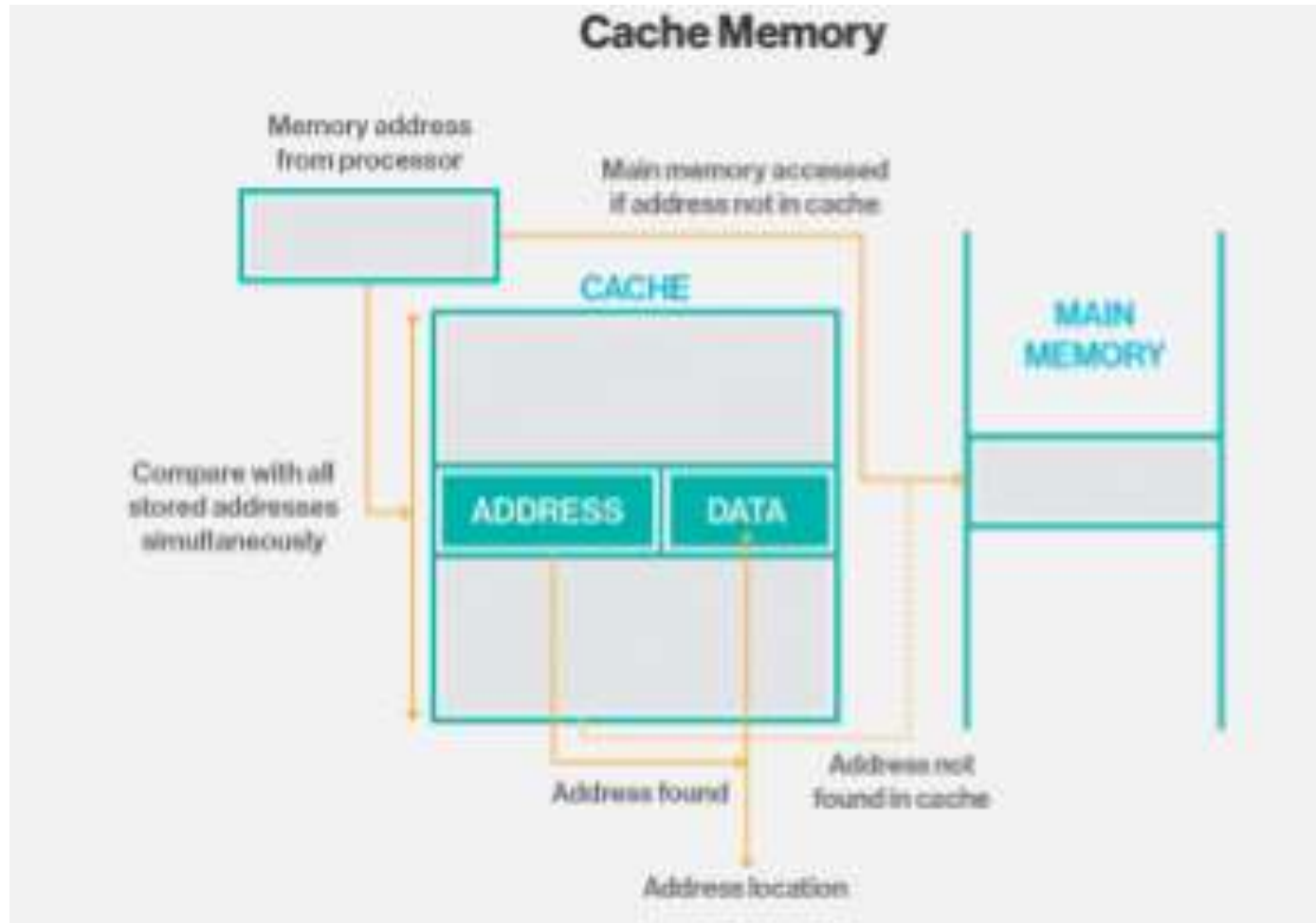
Current Cache Design

CURRENT CACHE DESIGN

- ◉ In computing, a cache is a hardware or software component that stores data so that future requests for that data can be served faster; the data stored in a cache might be the result of an earlier computation or a copy of data stored elsewhere. A cache hit occurs when the requested data can be found in a cache, while a cache miss occurs when it cannot. Cache hits are served by reading data from the cache, which is faster than recomputing a result or reading from a slower data store; thus, the more requests that can be served from the cache, the faster the system performs.



WORKING OF CACHE MEMORY





- ◉ The cache stores the frequently used data by the CPU. The CPU first checks the cache for the required data. Even though the RAM is fast, it is not as fast as the cache. Therefore, storing the commonly required data in the cache is beneficial to increase the computation speed.
- ◉ There are three types of cache. The level 1 cache is the smallest. It is located inside the CPU or the processor. So, it runs at the same speed as the CPU. Level 2 and level 3 caches are external. Level 2 cache is larger than level 1 cache. If the required data is not available in level 1 cache, the CPU checks the level 2 cache. If the required data is not available in both level 1 and level 2 caches, the CPU checks the level 3 cache. If the required data is not available in any of these caches, the CPU will access the RAM. Level 1 cache is the fastest cache of all. A CPU can have multiple cores. A core is the execution unit of the CPU. Each core can have separate level 1 and level 2 caches. The level 3 cache is shared among all cores

THANKYOU

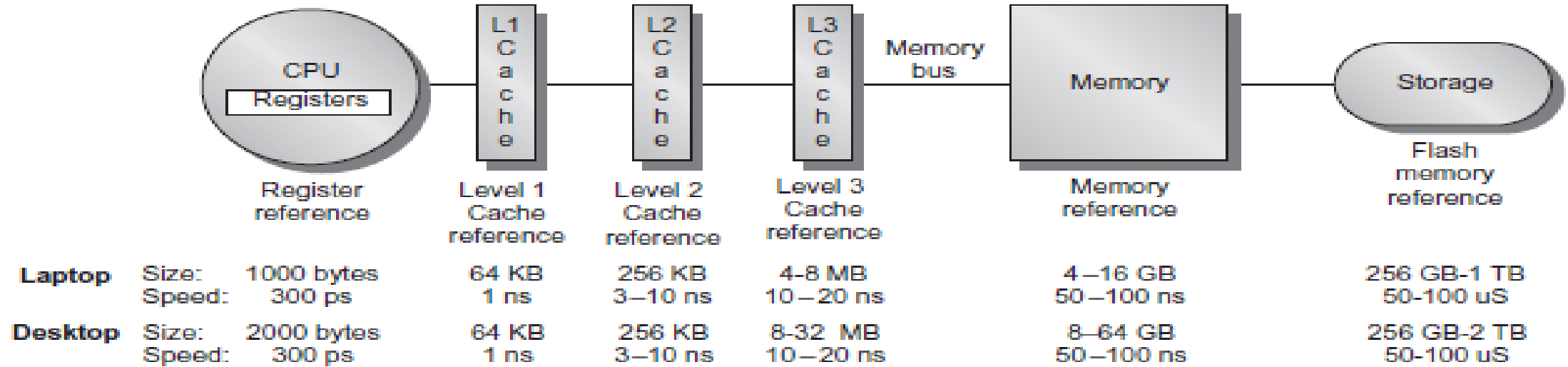
Sourabh Singh

CACHE EVOLUTION

Current cache design

Cache basic fact

- FACTS
 - BIG IS SLOW
 - FAST IS SMALL
- INCREASE PERFORMANCE BY HAVING “HIERARCHY” OF MEMORY SUBSYSTEMS
- “TEMPORAL LOCALITY” AND “SPATIAL LOCALITY” ARE BIG IDEAS



- L1 caches are usually on chip, and hence are area-constrained. L2 and L3 caches are implemented off chip, and hence have lesser stringent area constraints.
- L1 caches target lowering the access time, whereas lower level caches target to reduce the miss rates, due to an orders of magnitude higher miss penalties. Note that after the last level cache takes a miss, the processor needs to go to main memory, whose access time is a few hundreds of clock cycles for today's processors that clock GHz frequencies.
- On a multicore system, which is roughly all systems you encounter these days, L1 caches are private to each core, whereas lower level caches are shared. Hence the size difference. L1 size is reported on a per core basis, whereas the reported sizes of L2 and L3 are those of the entire shared chunk of cache memory at the respective levels.

we want to be able to access it quickly (low access time) and we'd also like to hit in the cache as much as possible. Another thing we want is good bandwidth

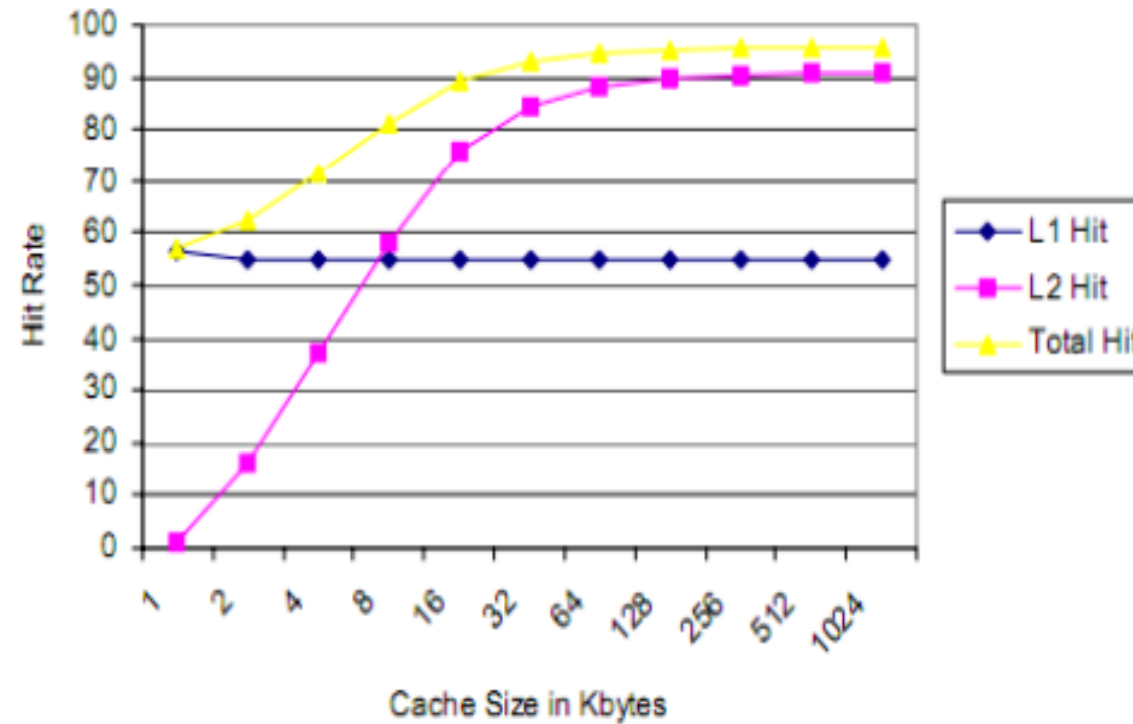
Memory Size and Distance are certainly key factors.

- Size of L3 cache > Size of L2 cache > Size of L1 cache,

which implies, the time to search and get a cache hit is inversely proportional to the size of the cache. If T is the time to search for a cache line and bring it into the processor,

- $T(L3) > T(L2) > T(L1)$

Hit Rates for Constant L1, Increasing L2



THANK YOU

Rohit Kumar





CACHE EVOLUTION

Current Cache Design

What is a Cache?

The cache is a very high speed, expensive piece of memory, which is used to speed up the memory retrieval process. Due to its higher cost, the CPU comes with a relatively small amount of cache compared with the main memory. Without cache memory, every time the CPU requests for data, it would send the request to the main memory which would then be sent back across the system bus to the CPU. This is a slow process. The idea of introducing cache is that this extremely fast memory would store data that is frequently accessed and if possible, the data that is around it. This is to achieve the quickest possible response time to the CPU.



Types of Cache Memory

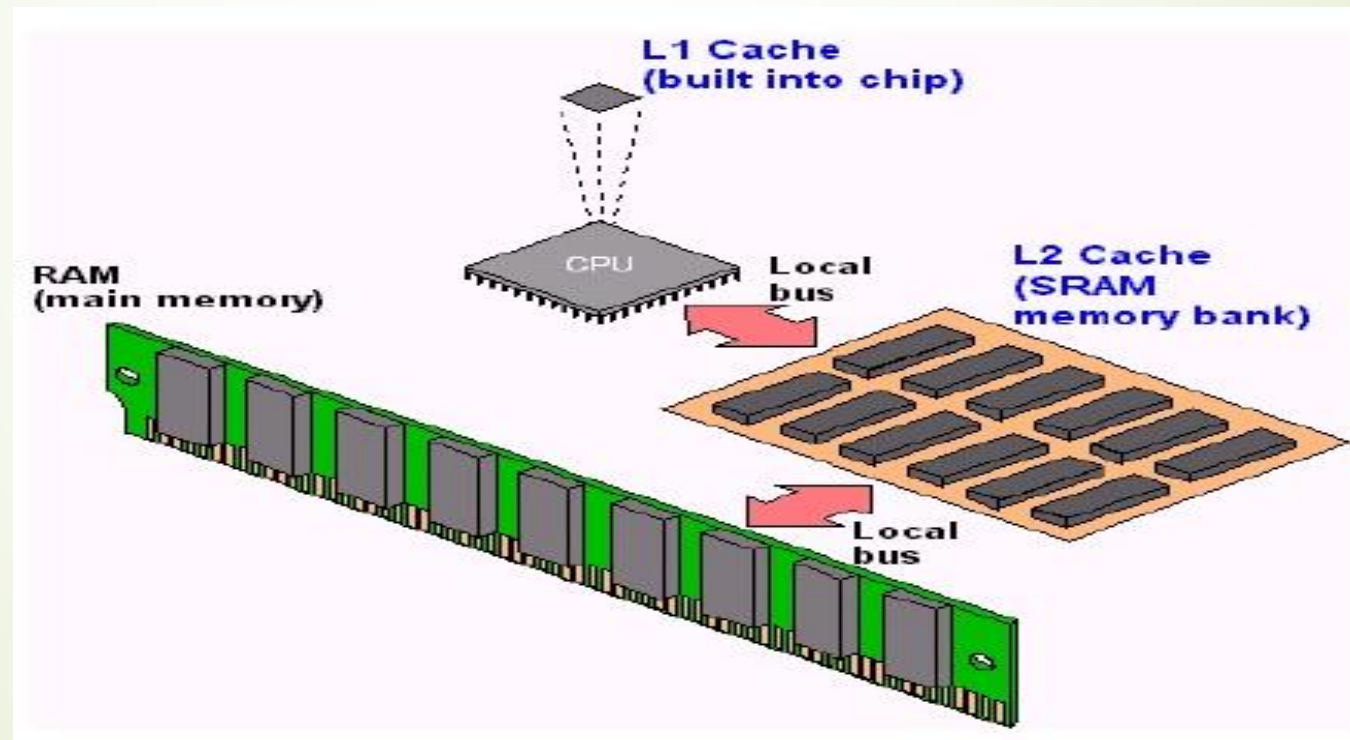
- **Memory Cache:** A memory cache, sometimes called a cache store or RAM cache, is a portion of memory made of high-speed static RAM (SRAM) instead of the slower and cheaper dynamic RAM (DRAM) used for main memory. Memory caching is effective because most programs access the same data or instructions over and over. By keeping as much of this information as possible in SRAM, the computer avoids accessing the slower DRAM.
- **Disk Cache:** Disk caching works under the same principle as memory caching, but instead of using high-speed SRAM, a disk cache uses conventional main memory. The most recently accessed data from the disk (as well as adjacent sectors) is stored in a memory buffer. When a program needs to access data from the disk, it first checks the disk cache to see if the data is there. Disk caching can dramatically improve the performance of applications, because accessing a byte of data in RAM can be thousands of times faster than accessing a byte on a hard disk.

Levels of Cache: Cache memory is categorized in levels based on its closeness and accessibility to the microprocessor. There are three levels of a cache.

□ **Level 1(L1) Cache:** This cache is inbuilt in the processor and is made of SRAM(Static RAM) Each time the processor requests information from memory, the cache controller on the chip uses special circuitry to first check if the memory data is already in the cache. If it is present, then the system is spared from time consuming access to the main memory. In a typical CPU, primary cache ranges in size from 8 to 64 KB, with larger amounts on the newer processors. This type of Cache Memory is very fast because it runs at the speed of the processor since it is integrated into it.

□ **Level 2(L2) Cache:** The L2 cache is larger but slower in speed than L1 cache. It is used to see recent accesses that is not picked by L1 cache and is usually 64 to 2 MB in size. A L2 cache is also found on the CPU. If L1 and L2 cache are used together, then the missing information that is not present in L1 cache can be retrieved quickly from the L2 cache. Like L1 caches, L2 caches are composed of SRAM but they are much larger. L2 is usually a separate static RAM (SRAM) chip and it is placed between the CPU & DRAM(Main Memory)


□ **Level 3(L3) Cache:** L3 Cache memory is an enhanced form of memory present on the motherboard of the computer. It is an extra cache built into the motherboard between the processor and main memory to speed up the processing operations. It reduces the time gap between request and retrieving of the data and instructions much more quickly than a main memory. L3 cache are being used with processors nowadays, having more than 3 MB of storage in it.





Thank You

G.Mahidhar.




Future
cache
design

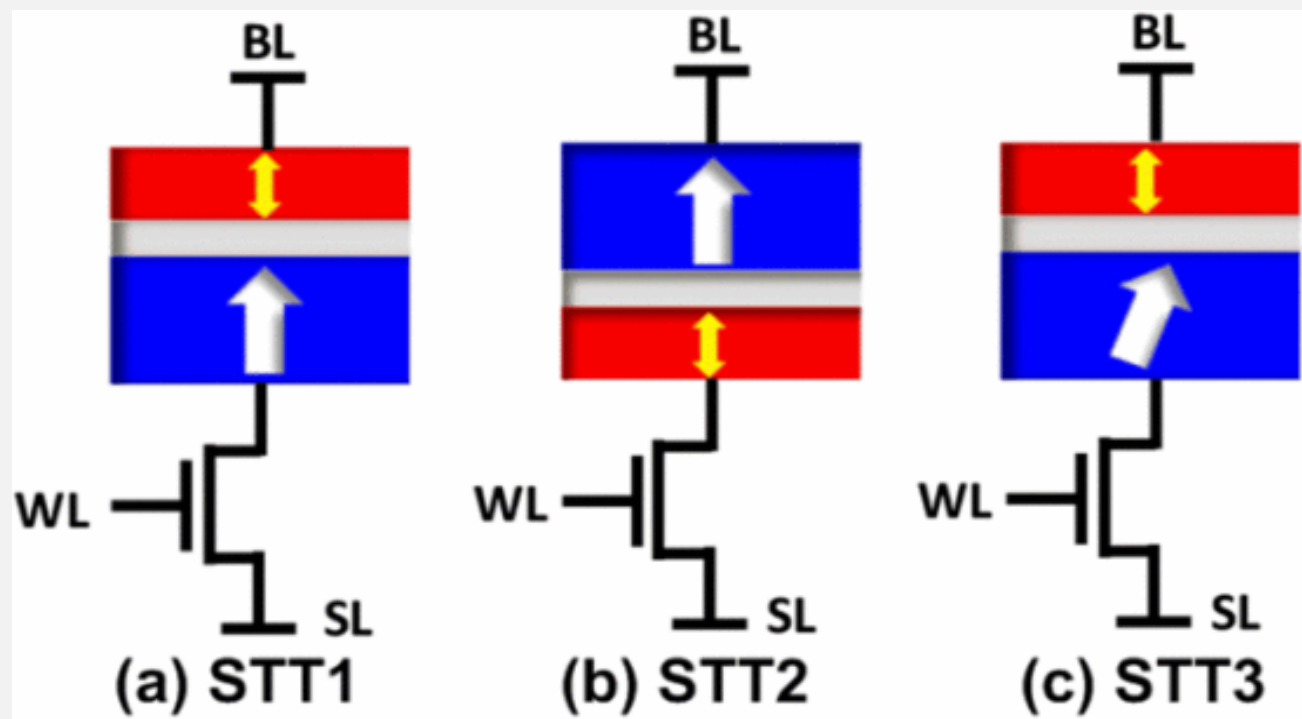
--STTMRAM



Introduction

- The ever-increasing gap between processor speed and main memory latency has driven the demand for larger on-chip caches in processors.
- Traditionally, on-chip caches in modern processors are implemented using static random access memories (SRAM).
- However, limited scalability, susceptibility to soft errors and high leakage power of SRAM pose challenges to high-density on-chip cache implementation. In order to address the limited scalability of SRAMs, several recent processors have adopted embedded dynamic RAM (EDRAM) in lower level caches.
- Among various candidates, spin-transfer torque magnetic RAM (STT MRAM) is considered as a promising technology that can offer desirable memory attributes such as high endurance, non-volatility, soft error immunity, zero standby power and high integration capability

- 
- A conventional STT MRAM cell comprises of a magnetic tunnel junction (MTJ) and an access transistor in series
 - The MTJ contains a pinned layer and a free layer separated by a dielectric layer (e.g. MgO).
 - A read operation is performed by sensing resistance difference of the two binary states
 - A write operation is performed by passing a current (I_w) through the bitcell that exceeds a critical current (I_c). The direction of (I_w) determines the final magnetization of the free layer (*i.e.*, parallel or anti-parallel states of the MTJ)



STT MRAM Cache vs. SRAM Cache

- A cache comprises of multiple arrays for storing tags and data bits. In conventional on-chip caches, both the tag and data arrays are implemented using SRAM. On the other hand, in the proposed STT MRAM caches, the tag arrays are implemented using SRAM and data arrays are implemented using STT MRAM.
- This is due to the fact that the write latency of STT MRAM may not be suitable for tag array operation, which requires frequent and fast updates of status bits and history bits
- From simulator It is shown that STT MRAM caches have a much higher integration density than SRAM cache.



Cache Utilization and Energy Consumption

- The contribution of active and leakage energy to total energy consumption is different for SRAM and STT MRAM-based caches.
- The leakage energy in an STT MRAM cache is smaller than an SRAM cache even with 4 times larger capacity (at iso-area).
- On the other hand, the dynamic energy for a write operation is higher in an STT MRAM cache compared to an SRAM cache.
- It is important to note that the total energy dissipation in a cache depends on factors such as cache access patterns (number of read and write operations) and cache utilization (number of times a processor accesses the cache per unit cycle).



THANK
YOU