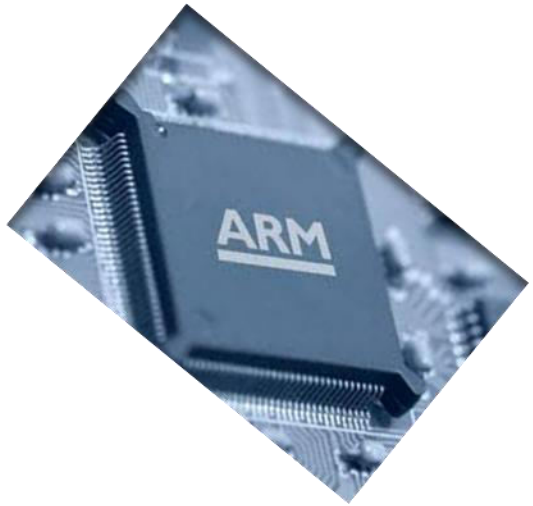
MICROPROCESSOR AND COMPUTER ARCHITECTURE

CACHE OPTIMIZATION



Credits:

MPCA Team

Cache Optimization

Sixth optimization:

Avoiding Address Translation during Indexing of the Cache to Reduce Hit Time.

Sixth Optimization: Avoiding Address Translation during indexing of the Cache to Reduce Hit Time

- Memory Management module of Operating System.
- Cache must cope with the translation of a virtual address to physical address to access memory.
- Since, processor treats main memory as just another memory in hierarchy, the address of the virtual memory that exists on the disk must be mapped on to the main memory.

Sixth Optimization: Avoiding Address Translation during indexing of the Cache to Reduce Hit Time

- Guidelines for cache:
  - Virtual addresses are used.
  - Hits are common than Misses.
- These caches are called **Virtual Caches.**

Sixth Optimization: Avoiding Address Translation during indexing of the Cache to Reduce Hit Time

- Two Tasks:
  - Indexing the cache.
  - Comparing Addresses.
    - A virtual or physical address is used
      - To index the cache
      - In tag Comparison.
  - Virtual addressing for both indexing & tags eliminates address translation time from cache hit.

Sixth Optimization: Avoiding Address Translation during indexing of the Cache to Reduce Hit Time

NOTE: Why doesn't everyone build virtually addressed cache?

Reason 1 : Protection: page level protection is checked as a part of physical address translation.

Solution : To copy the protection information from TLB on a miss, add a field to hold it, and check it on every access to the virtually addressed cache

Reason 2 : Every time a process is switched, the virtual address refer to the different physical address, requiring the cache to be flushed

Solution: To increase the width of the cache address tag with process identifier tag (PID).

Sixth Optimization: Avoiding Address Translation during indexing of the Cache to Reduce Hit Time

If OS assigns these tags to processes, it will need flush the cache when PID is recycled: i.e., PID distinguishes whether or not the data in the cache are for this program. Thus, shows improvements in miss rates by using PIDs to avoid cache flushes.

- Address translations are kept in a special cache.
- A memory access rarely requires a second access to translate the data.
- This special address translation is referred to as a translation lookaside buffer (TLB) also called Translation buffer(TB).

Virtual Address to Physical Address

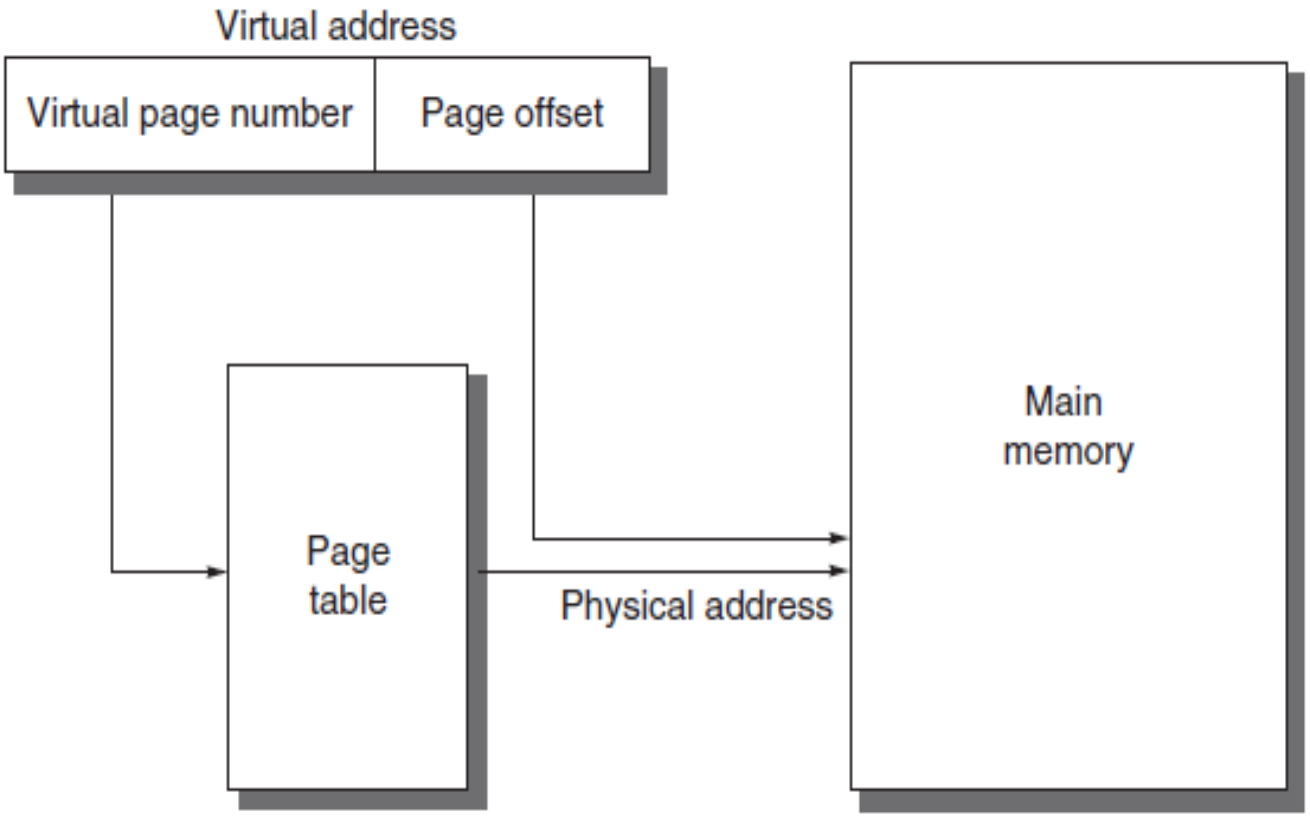


Figure B.23 The mapping of a virtual address to a physical address via a page table.

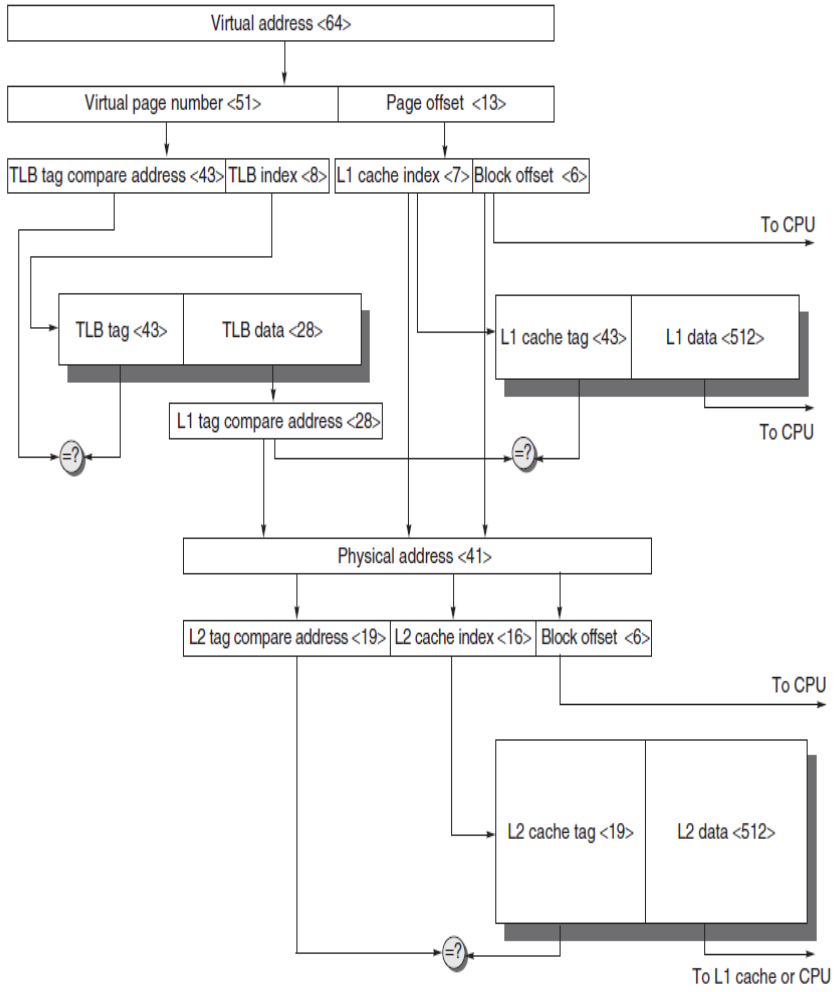


Figure B.25 The overall picture of a hypothetical memory hierarchy going from virtual address to L2 cache access. The page size is 8 KB. The TLB is direct mapped with 256 entries. The L1 cache is a direct-mapped 8 KB, and the L2 cache is a direct-mapped 4 MB. Both use 64-byte blocks. The virtual address is 64 bits and the physical address is 41 bits. The primary difference between this simple figure and a real cache is replication of pieces of this figure.

Virtual Address to Physical Address

- 64 bit virtual address logically divided into Virtual Page Number & Page Offset.
- Virtual page number sent to TLB to translate to physical address.
- Page number sent to L1 cache to act as index.
- If TLB match is a hit, physical page number is sent to L1 cache tag to check for the match.
- If it matches it is a hit in L1 cache.
- The block offset then selects the word for the processor.

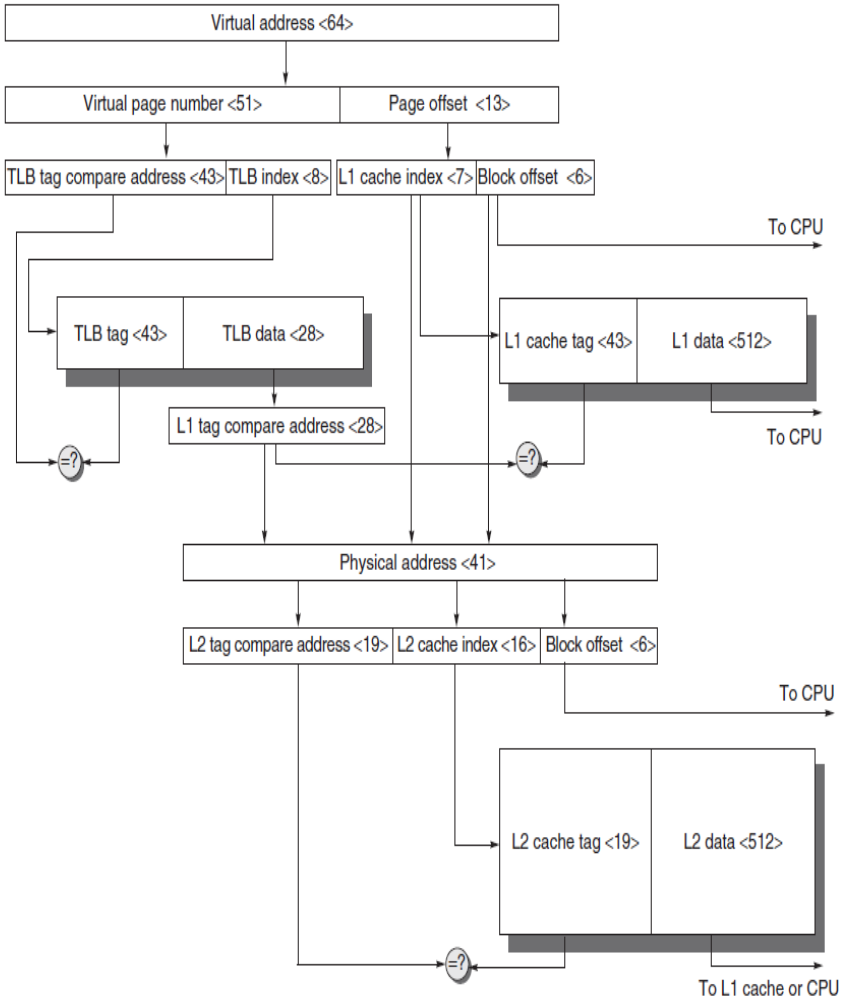


Figure B.25 The overall picture of a hypothetical memory hierarchy going from virtual address to L2 cache access. The page size is 8 KB. The TLB is direct mapped with 256 entries. The L1 cache is a direct-mapped 8 KB, and the L2 cache is a direct-mapped 4 MB. Both use 64-byte blocks. The virtual address is 64 bits and the physical address is 41 bits. The primary difference between this simple figure and a real cache is replication of pieces of this figure.

Translation Lookaside Buffer - TLB

- If L1 cache check is a miss, the physical address is then used to try the L2 cache.
- The middle portion the physical address is then used as an index.
- The resulting L2 cache tag is compared to the upper part of the physical address to check for the match.
  - If matches, it is a L2 cache hit.
  - Data is sent to the processor.
  - The block offset then selects the word for the processor.
- On a miss, the physical address is then used to get the block from the memory.

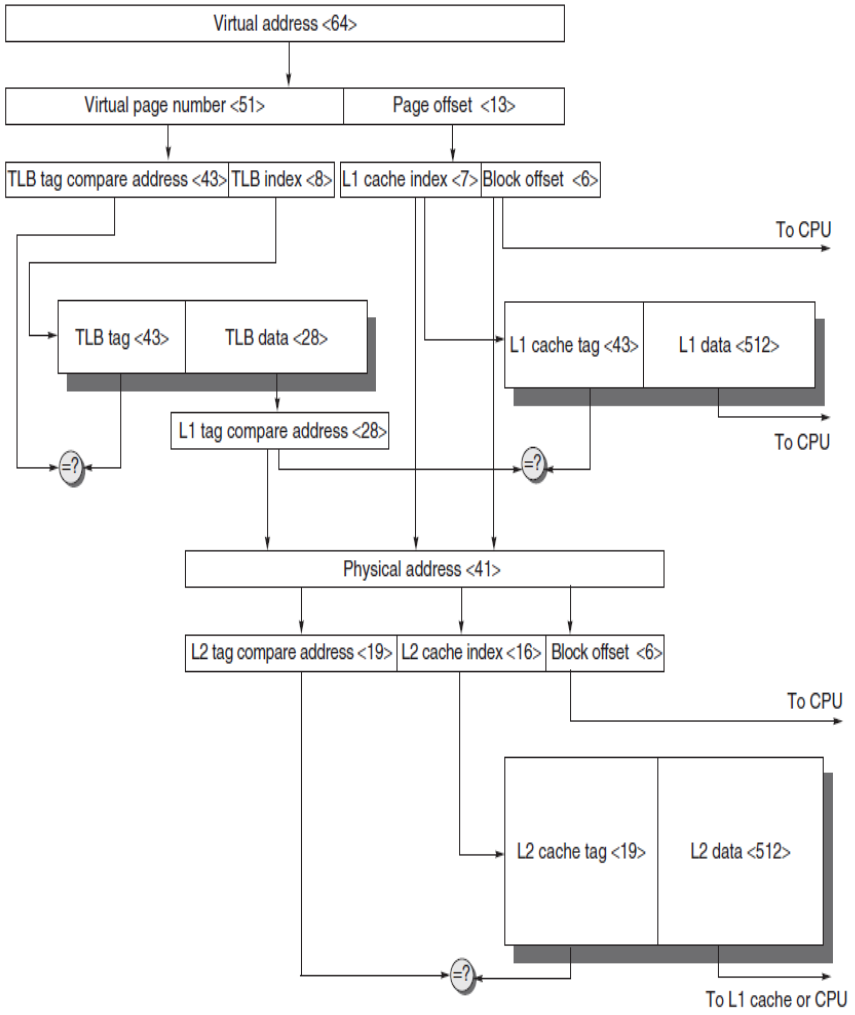


Figure B.25 The overall picture of a hypothetical memory hierarchy going from virtual address to L2 cache access. The page size is 8 KB. The TLB is direct mapped with 256 entries. The L1 cache is a direct-mapped 8 KB, and the L2 cache is a direct-mapped 4 MB. Both use 64-byte blocks. The virtual address is 64 bits and the physical address is 41 bits. The primary difference between this simple figure and a real cache is replication of pieces of this figure.

Translation Lookaside Buffer - TLB

First Simplification:

- L1 cache is Split cache, It requires, two TLBs.
  - One cache & TLB is for **instructions.**
    - Driven from the PC.
  - One cache & TLB is for **Data.**
    - Driven from the effective address.

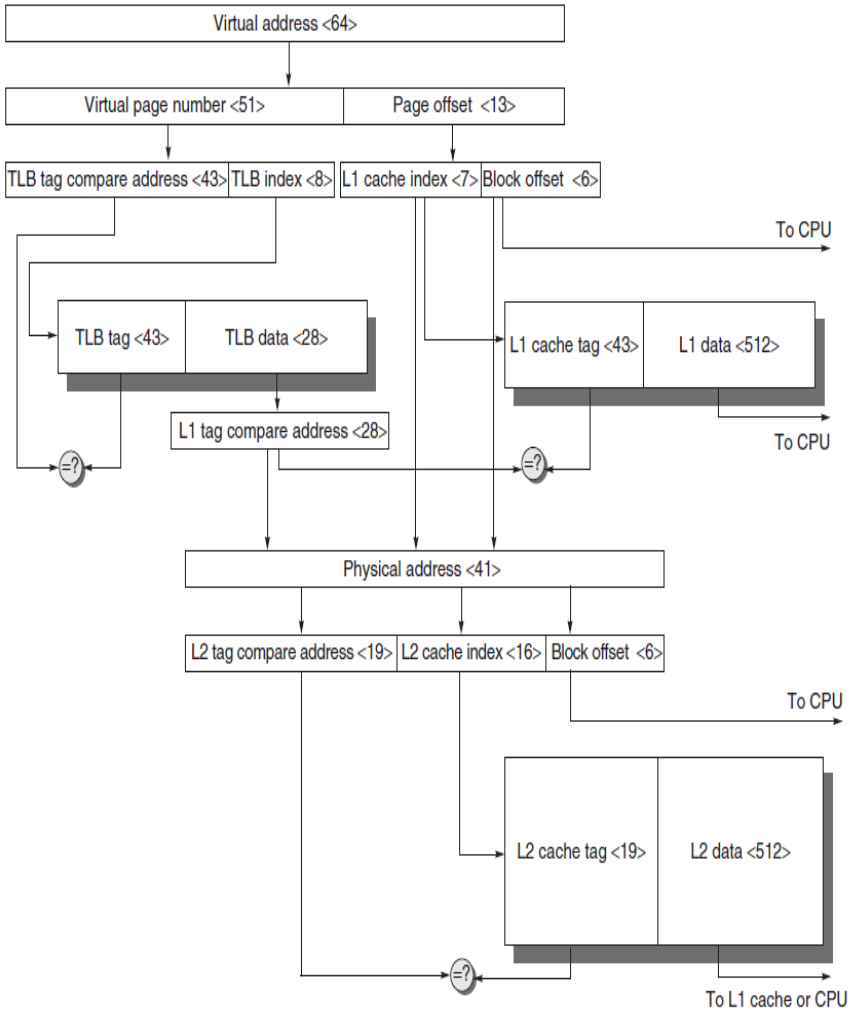


Figure B.25 The overall picture of a hypothetical memory hierarchy going from virtual address to L2 cache access. The page size is 8 KB. The TLB is direct mapped with 256 entries. The L1 cache is a direct-mapped 8 KB, and the L2 cache is a direct-mapped 4 MB. Both use 64-byte blocks. The virtual address is 64 bits and the physical address is 41 bits. The primary difference between this simple figure and a real cache is replication of pieces of this figure.

Translation Lookaside Buffer - TLB

Second Simplification:

All caches and TLBs are direct mapped.

If associativity is used it would replicate each set of comparators, tag memory and data memory.

Connect data memories with a n:1 MUX to select a hit.

If total cache size remained the same, the cache index would also shrink by log2n bits.

Q & A

Six Basic Cache Optimizations