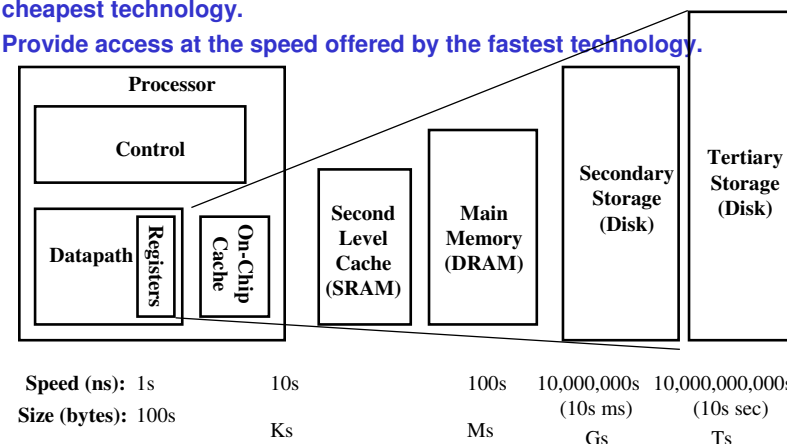


## Virtual Memory

Lecturer: Dr. Hui Annie Guo  
huig@cse.unsw.edu.au  
K17-501F (ext. 57136)

## Recap: Memory Hierarchy of a Modern Computer System

- By taking advantage of the principle of locality:
  - Present the user with as much memory as is available in the cheapest technology.
  - Provide access at the speed offered by the fastest technology.



COMP3211/9211

2011S1 wk10\_1 P2

## Virtual memory vs Physical memory

- Physical memory
  - Main memory in the hierarchical memory system
  - Its addresses are called *physical addresses*
- Virtual memory
  - A technique that
    - allows users/programs to reference memory larger than actually exists in the computer
    - Uses physical memory as a cache for secondary storage
    - The addresses in virtual memory are called *virtual addresses*.

## Virtual Memory: Key Ideas (1/2)

- Virtual address space, divided into **pages**, is much larger than physical address space, which is divided into similar sized blocks known as **frames**.
- A process, running on the processor, refers to data using virtual addresses
- The virtual addresses must be translated into physical addresses to access actual memory locations.
- Virtual page numbers (the high order address bits) are translated into physical frame numbers using a **page table** that is stored in the physical memory.
- The data cache is indexed/tagged using physical addresses.

## Virtual Memory: Key Ideas (2/2)

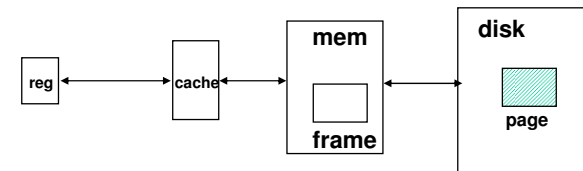
- To speed up translation, a translation lookaside buffer (TLB) – a small associative cache – is used to store recent page-frame translations
  - More will be covered later
- Hardware must trap to the operating system if a page is not resident in memory so that it can be loaded from disk – this may result in another page having to be written back to disk.

COMP3211/9211

2011S1 wk10\_1 P5

## Basic Issues in Virtual Memory System Design

- The size of the block that is transferred from secondary (Disk) to main storage ([M]emory)
  - E.g. virtual and physical address space partitioned into blocks of equal size
  - page size = frame size
- The load policy: when missing item fetched from secondary memory (e.g. disk)
  - only on the occurrence of a fault (demand load policy)
- Placement: which region of M is to hold the new block
- Replacement: when M is full, some region of M, must be released to make room for the new block



COMP3211/9211

2011S1 wk10\_1 P6

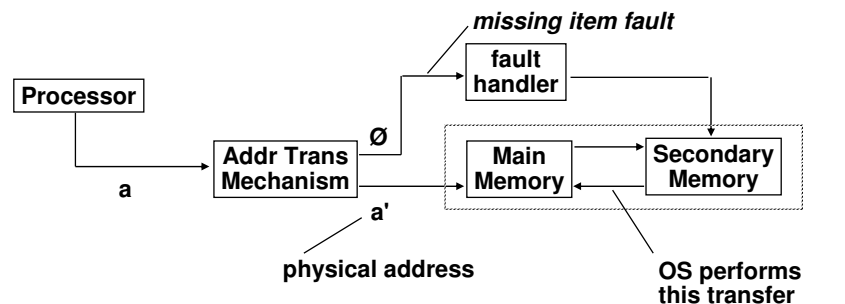
## Address Map ([B]lock [I]dentification)

$V = \{0, 1, \dots, n - 1\}$  virtual address space

$M = \{0, 1, \dots, m - 1\}$  physical address space,  $n > m$

MAP:  $V \rightarrow M \cup \{\emptyset\}$  address mapping function

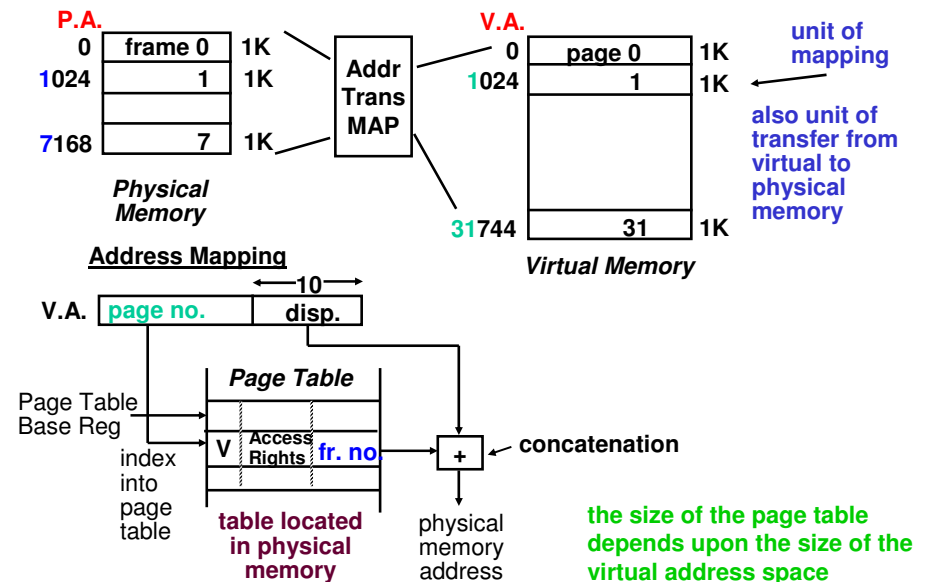
MAP(a) = a' if data at virtual address  $\underline{a}$  is present in physical memory (M) at address  $\underline{a}'$   
 =  $\emptyset$  if data at virtual address  $\underline{a}$  is not present in M



COMP3211/9211

2011S1 wk10\_1 P7

## Address Translation

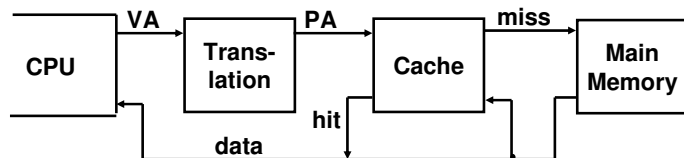


COMP3211/9211

2011S1 wk10\_1 P8

## Address and Cache Access

- Page table is implemented in the main memory. It takes an extra memory access to translate VA to PA
- This makes cache access very expensive, and this is the "innermost loop" that you want to have go as fast as possible!!

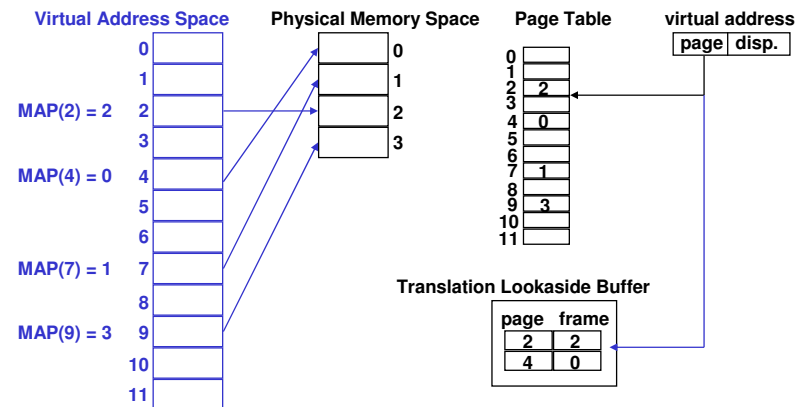


COMP3211/9211

2011S1 wk10\_1 P9

## Making address translation fast: TLB

- Translation Look-aside Buffer (TLB) provides a cache of recent translations



COMP3211/9211

2011S1 wk10\_1 P10

## Translation Look-Aside Buffers (TLB)

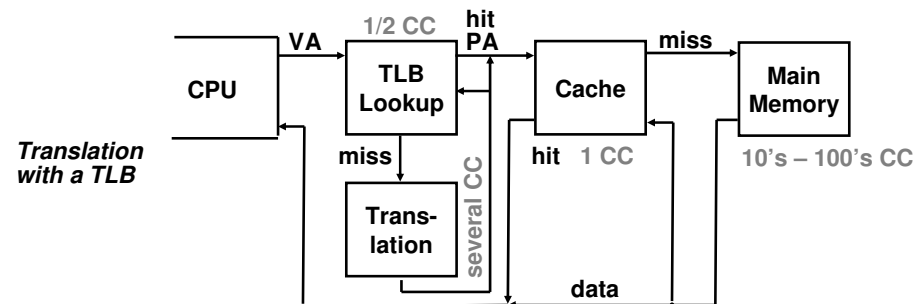
- Just like any other cache, the TLB can be organized as
  - fully associative,
  - set associative, or
  - direct mapped
- TLBs are usually small, typically not more than 128 - 256 entries
  - This permits fully associative lookup on high-end machines.
  - Most mid-range machines use small n-way set associative organizations.

COMP3211/9211

2011S1 wk10\_1 P11

## Datapath with TLB

- Access TLB requires 1/2 clock cycles as compared to multiple cycles to access the page table



COMP3211/9211

2011S1 wk10\_1 P12

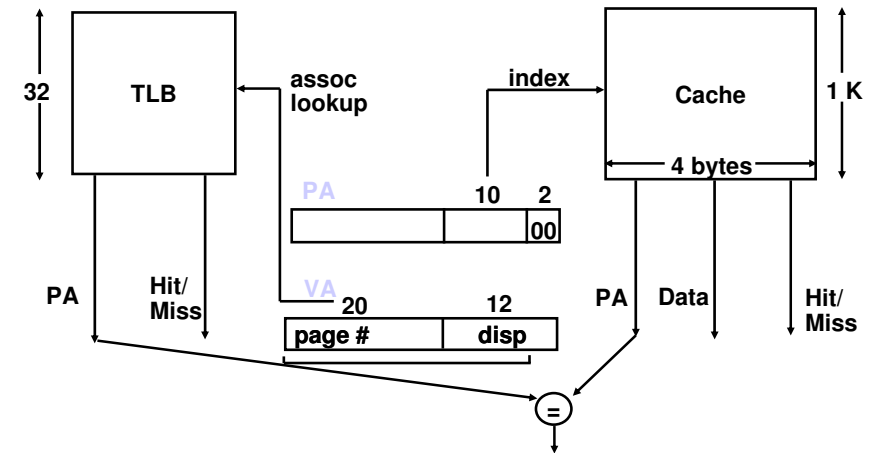
## Further Improvement

- **Reduce the effect of address translation on performance**
  - **Overlapping the cache access with the TLB access**
    - Works because high order bits of the VA are used to look in the TLB while low order bits are used as index into cache

MIPS R3000 Pipeline

Inst Fetch	Dcd/ Reg	ALU / E.A	Memory	Write Reg
TLB	RF	Operation		WB
I-Cache		E.A. TLB	D-Cache	

## Overlapped Cache & TLB Access



IF cache hit AND (cache tag = PA) THEN deliver data to CPU  
 ELSE IF cache miss AND TLB hit THEN  
 access memory with the PA from the TLB  
 ELSE do standard VA translation

## Optimal Page Size

- **Minimize wasted storage**
  - small page minimizes internal fragmentation
  - small page increase size of page table
- **Minimize transfer time**
  - large pages (multiple disk sectors) amortize access cost
  - sometimes transfer unnecessary info
  - sometimes prefetch useful data
  - sometimes discards useless data early
- **General trend toward larger pages because**
  - big cheap RAM
  - increasing mem / disk performance gap
  - larger address spaces