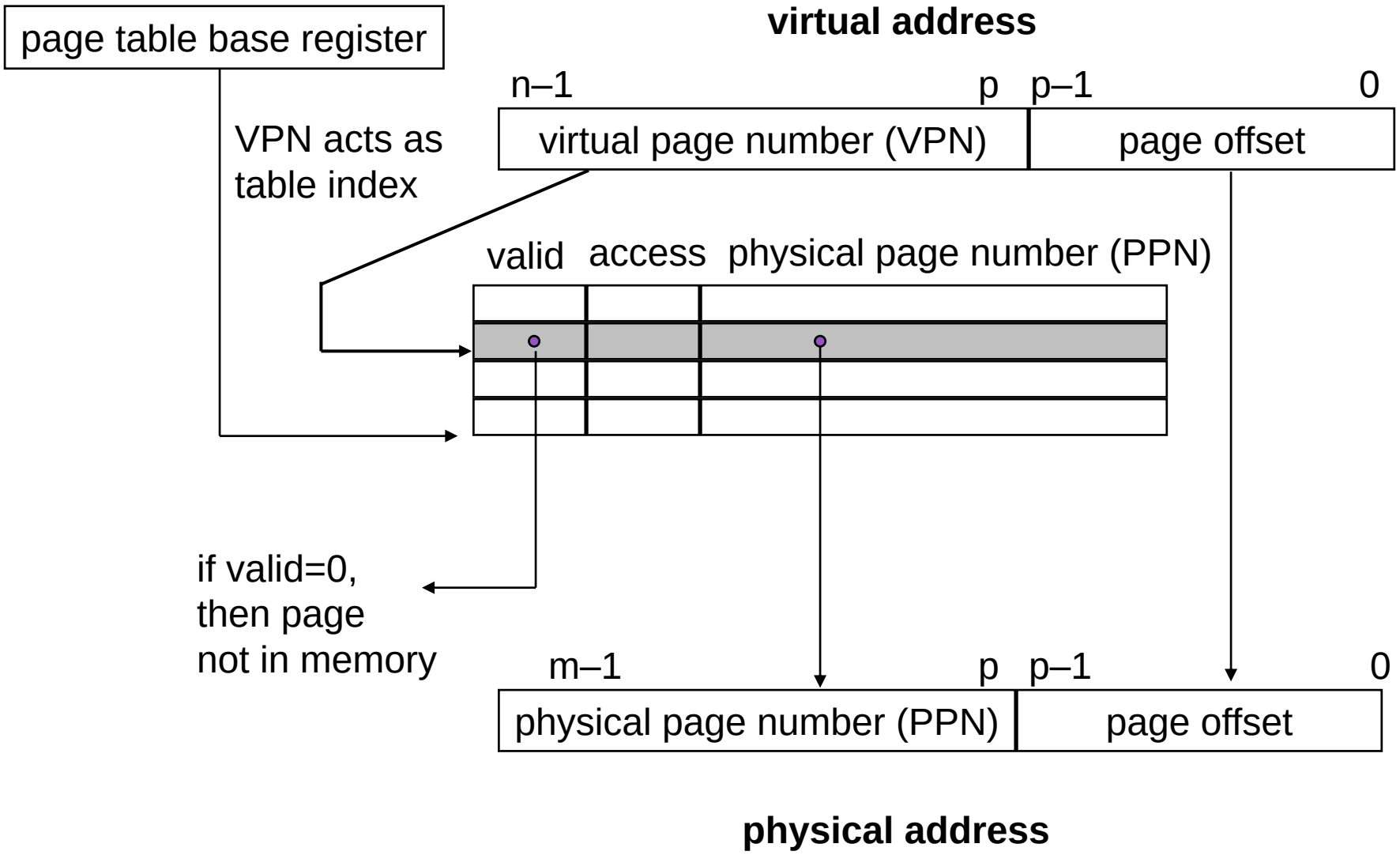
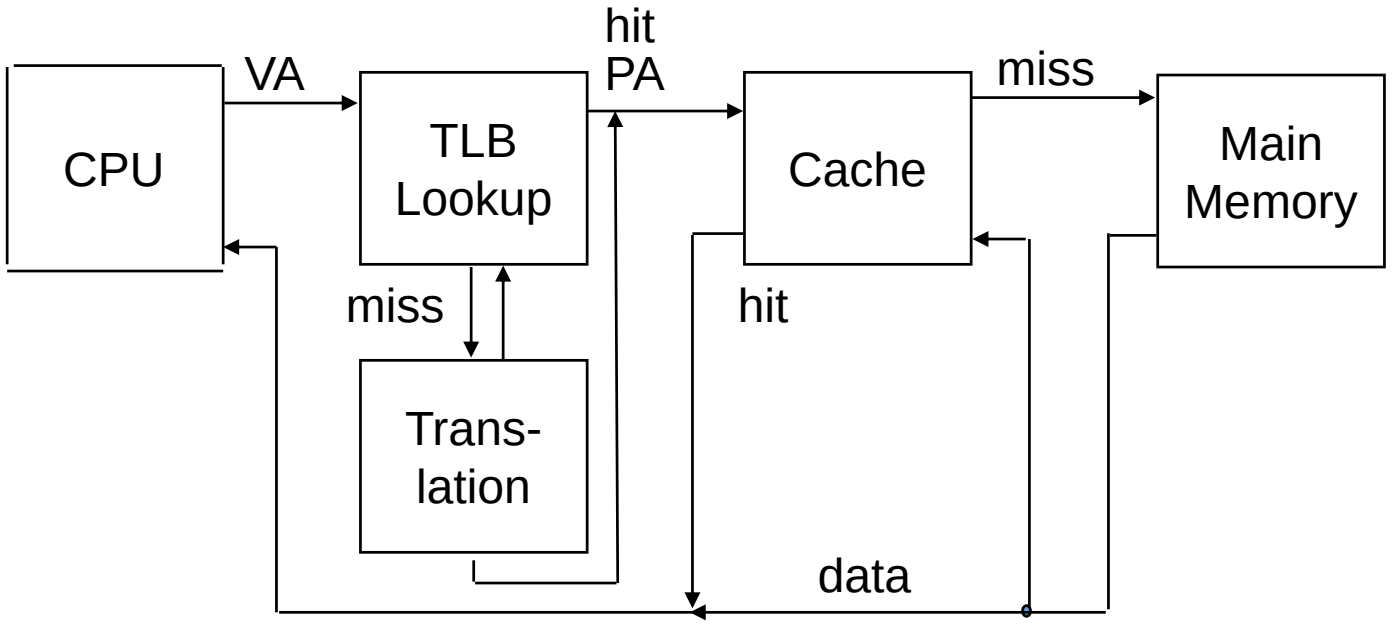
Address Translations

Address Translation via Page Table

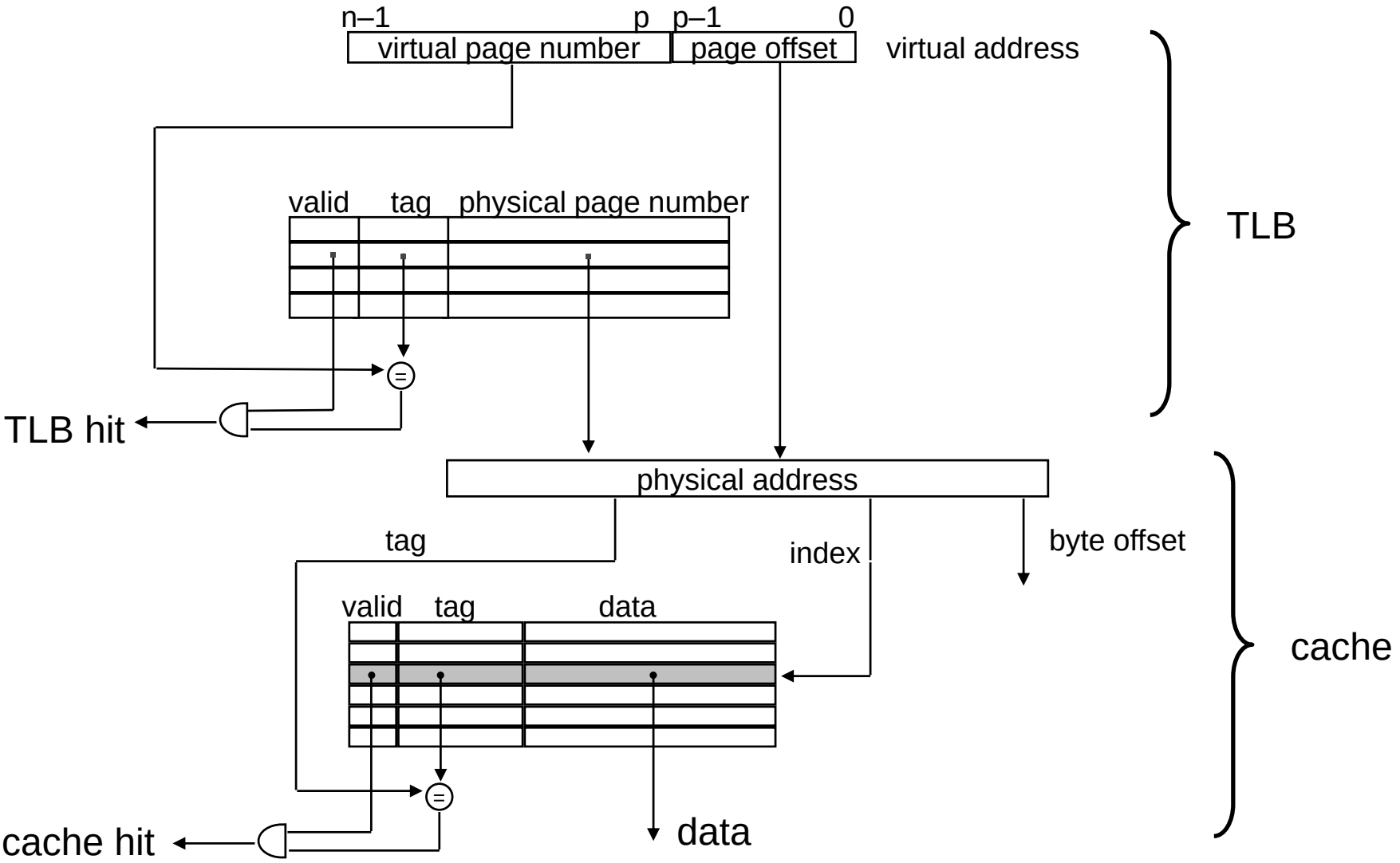


Faster Translations

- "Translation lookaside buffer" (TLB)
  - Small hardware cache in MMU
  - Maps virtual page numbers to physical page numbers
  - Contains complete page table entries for small number of pages

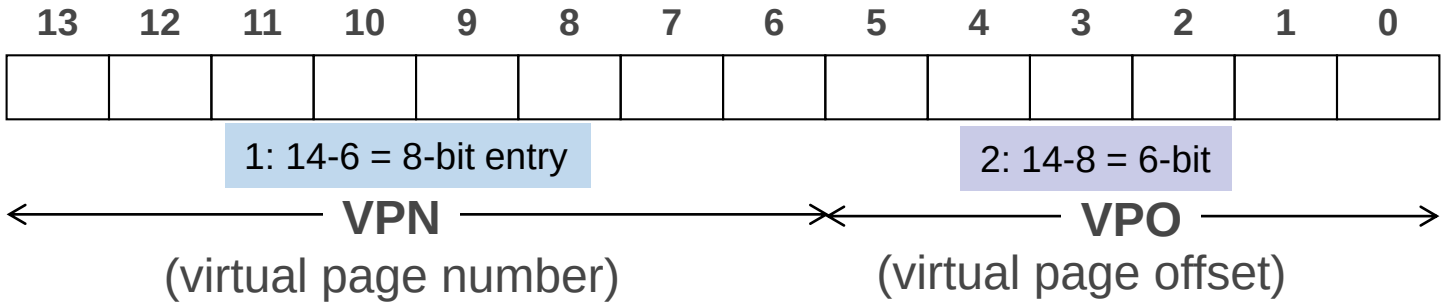


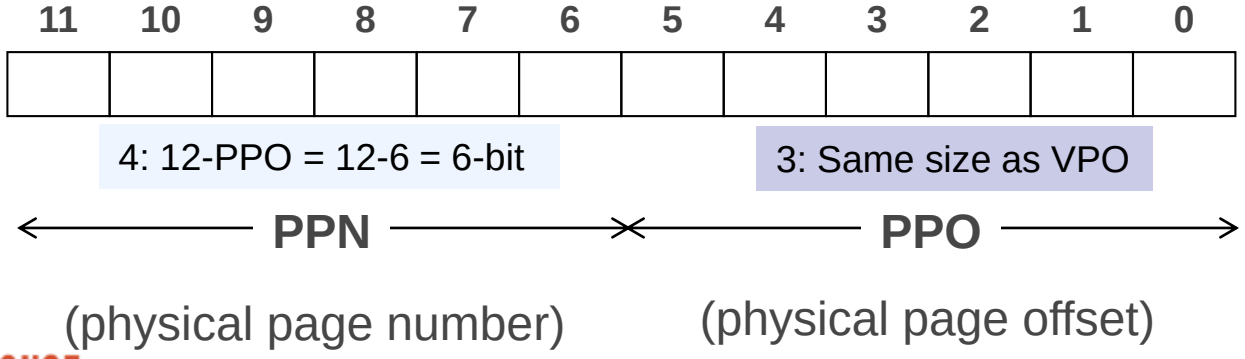
Address Translation with a TLB



Simple Memory System Example

- Addressing
  - 14-bit virtual addresses
  - 12-bit physical address
  - Page size = 64 bytes (6-bit)





Simple Memory System Page Table

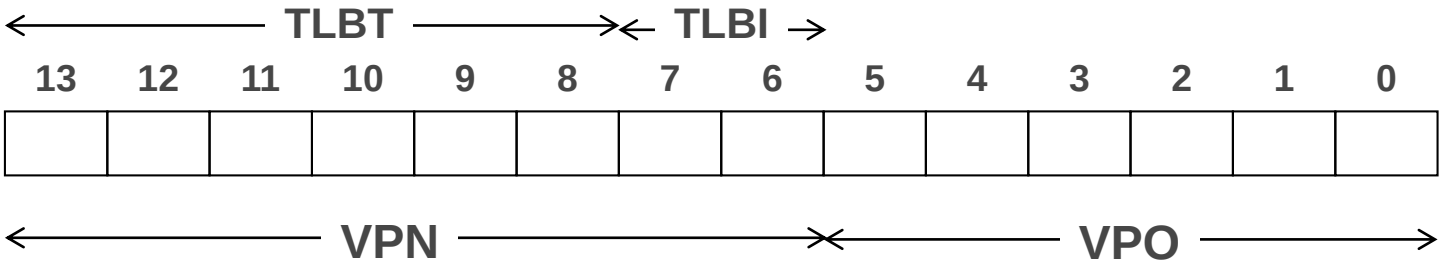
VPN	PPN	Valid	VPN	PPN	Valid
00	28	1	08	13	1
01	–	0	09	17	1
02	33	1	0A	09	1
03	02	1	0B	–	0
04	–	0	0C	–	0
05	16	1	0D	2D	1
06	–	0	0E	11	1
07	–	0	0F	0D	1

0D = 0000 1101

- Showing only first 16 entries

Simple Memory System TLB

- TLB
  - 16 entries
  - Four-way associative

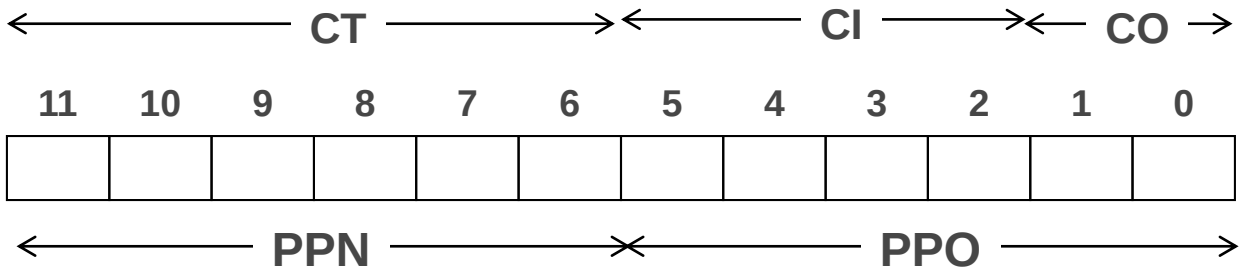


VPN = 0D = 000011 01 ➔ TAG=03, SET 1

Set	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid
0	03	–	0	09	0D	1	00	–	0	07	02	1
1	03	2D	1	02	–	0	04	–	0	0A	–	0
2	02	–	0	08	–	0	06	–	0	03	–	0
3	07	–	0	03	0D	1	0A	34	1	02	–	0

Simple Memory System Cache

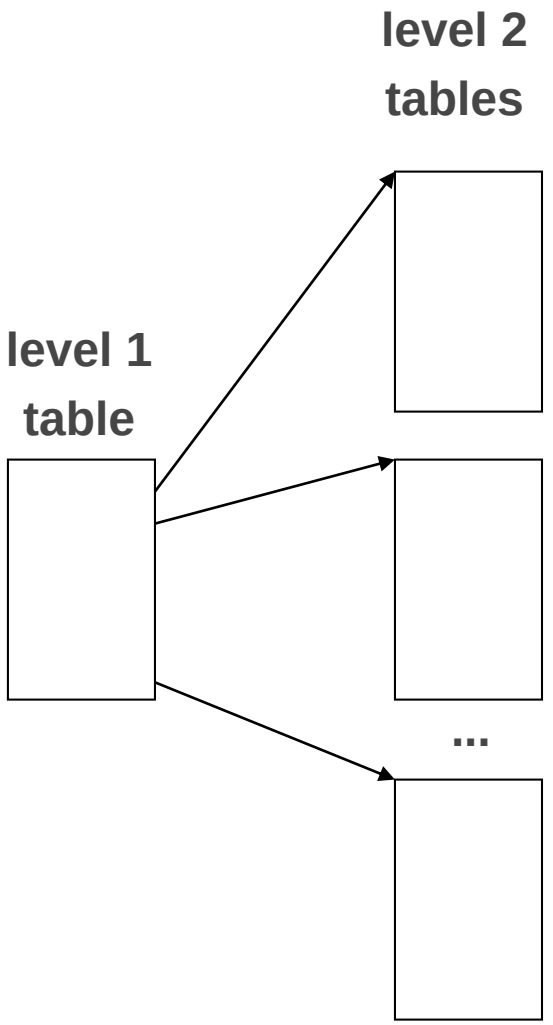
- Cache
  - 16 lines
  - 4-byte line size
  - Direct mapped



Idx	Tag	Valid	B0	B1	B2	B3	Idx	Tag	Valid	B0	B1	B2	B3
0	19	1	99	11	23	11	8	24	1	3A	00	51	89
1	15	0	–	–	–	–	9	2D	0	–	–	–	–
2	1B	1	00	02	04	08	A	2D	1	93	15	DA	3B
3	36	0	–	–	–	–	B	0B	0	–	–	–	–
4	32	1	43	6D	8F	09	C	12	0	–	–	–	–
5	0D	1	36	72	F0	1D	D	16	1	04	96	34	15
6	31	0	–	–	–	–	E	13	1	83	77	1B	D3
7	16	1	11	C2	DF	03	F	14	0	–	–	–	–

Multilevel Page Tables

- Given:
  - 4KB (2^{12}) page size
  - 32-bit address space
  - 4-byte PTE (to address memory)
- Problem:
  - Would need a 4 MB page table!
    - 2^{20} *4 bytes
- Common solution
  - Multilevel page tables
  - E.g., two-level table (P6)
    - Level 1 table: 1024 entries, each of which points to a level 2 page table.
    - Level 2 table: 1024 entries, each of which points to a page.



Virtual Memory

- Programmer's view
  - Large "flat" address space.
    - Can allocate large blocks of contiguous addresses
  - Processor "owns" machine.
    - Has private address space
    - Unaffected by behavior of other processes
- System view
  - User virtual address space created by mapping to set of pages.
    - Need not be contiguous
    - Allocated dynamically
    - Enforce protection during address translation
  - OS manages many processes simultaneously.
    - Continually switching among processes
    - Especially when one must wait for resource
      - E.g., disk I/O to handle page fault

	Paged VM	TLBs
Total size	16,000 to 250,000 words	16 to 512 entries
Block size	1000 to 16,000	1 to 8
Miss penalty (clocks)	10,000,000 to 100,000,000	10 to 1000
Miss rates	0.00001% to 0.0001%	0.01% to 2%

TLB Event Combinations

TLB	Page Table	Cache	Possible? Under what circumstances?
Hit	Hit	Hit	Yes – what we want!
Hit	Hit	Miss	Yes – although the page table is not checked if the TLB hits
Miss	Hit	Hit	Yes – TLB miss, PA in page table
Miss	Hit	Miss	Yes – TLB miss, PA in page table, but data not in cache
Miss	Miss	Miss	Yes – page fault
Hit	Miss	Miss/ Hit	Impossible – TLB translation not possible if page is not present in memory
Miss	Miss	Hit	Impossible – data not allowed in cache if page is not in memory

Question

- Match the memory hierarchy element with the closest phrase:

A. L1 cache

B. L2 cache

C. Main memory

D. TLB

1. A cache for a cache
2. A cache for disks
3. A cache for a main memory
4. A cache for page table entries

Exercise

- For a data cache with a 92% hit rate and a 2-cycle hit latency, calculate the average memory access latency. Assume that latency to memory and the cache miss penalty together is 124 cycles.

Exercise

- Calculate the performance of a processor taking into account stalls due to data cache and instruction cache misses. The data cache (for loads and stores) is the same as described previously and 30% of instructions are loads and stores. The instruction cache has a hit rate of 90% with a miss penalty of 50 cycles and hit time is 1 cycle. Assume the base CPI using a perfect memory system is 1.0. Calculate the CPI of the pipeline, assuming everything else is working perfectly. Assume the load never stalls a dependent instruction and assume the processor must wait for stores to finish when they miss the cache. Finally, assume that instruction cache misses and data cache misses never occur at the same time.
  - Calculate the additional CPI due to the icache stalls.
  - Calculate the additional CPI due to the dcache stalls.
  - Calculate the overall CPI for the machine.

Exercise

- Consider a 64-byte cache with 8 byte blocks, an associativity of 2 and LRU block replacement. Virtual addresses are 16 bits. The cache is physically tagged. The processor has 16KB of physical memory.
  - What is the total number of tag bits?
  - Assume each page is 64 bytes. How large would a single-level page table be given that each page requires 4 protection bits, and entries must be an integral number of bytes.
  - Assume the cache is initialized empty, the CPU needs to access memory with the following address sequence: 024_8, 130_8, 270_8, 022_8, 570_8, 124_8, 224_8, and 026_8. For each memory access, decide whether it is a cache hit or cache miss. If it is a cache miss, what is the type of cache miss (i.e. compulsory miss, capacity miss, or conflict miss) ?

Exercise

- The memory architecture of a machine is summarized in the following table.

Virtual Address	54 bits
Page size	16 K bytes
PTE size	4 bytes

  - Assume that there are 8 bits reserved for the OS functions (protection, replacement, valid, modified and Hit/miss) other than required by the hardware translation algorithm. Derive the largest physical memory size (in bytes) allowed by this PTE format.
  - How large (in bytes) is the page table?
  - If a 512 Byte TLB is created for the page table. How many page entries can this TLB store?

Exercise

- Assume a 5-bit virtual address and a memory system that uses 4 bytes per page. The physical memory has 16 bytes (four page frames). The page table used is a one-level scheme. Initially the table indicates that no virtual pages have been mapped. Implementing a LRU page replacement algorithm, show the contents of physical memory after the following virtual accesses: 10100, 01000, 00011, 01011, 01011,11111.
- Show the contents of memory and the page table information after each access successfully completes. Also indicate when a page fault occurs. Each page table entry (PTE) is 1 byte.

Question

- Why is miss rate not a good metric for evaluating cache performance? What is the appropriate metric?

- What is the reason for using a combination of first and second- level caches rather than using the same chip area for a larger first-level cache?

Exercise

- The Average Memory Access Time equation (AMAT) has three components: hit time, miss rate, and miss penalty. For each of the following cache optimizations, indicate which component of the AMAT equation is improved.
  - Using a second-level cache
  - Using a direct-mapped cache
  - Using a 4-way set-associative cache
  - Using a virtually-addressed cache
  - Performing hardware pre-fetching using stream buffers
  - Using a non-blocking cache
  - Using larger blocks

Exercise

- Design a 128KB direct-mapped data cache that uses a 32-bit address and 16 bytes per block.
- (a) How many bits are used for the byte offset?
- (b) How many bits are used for the set (index) field?
- (c) How many bits are used for the tag?

Question

- Why is miss rate not a good metric for evaluating cache performance? What is the appropriate metric?

- What is the reason for using a combination of first and second- level caches rather than using the same chip area for a larger first-level cache?

Review

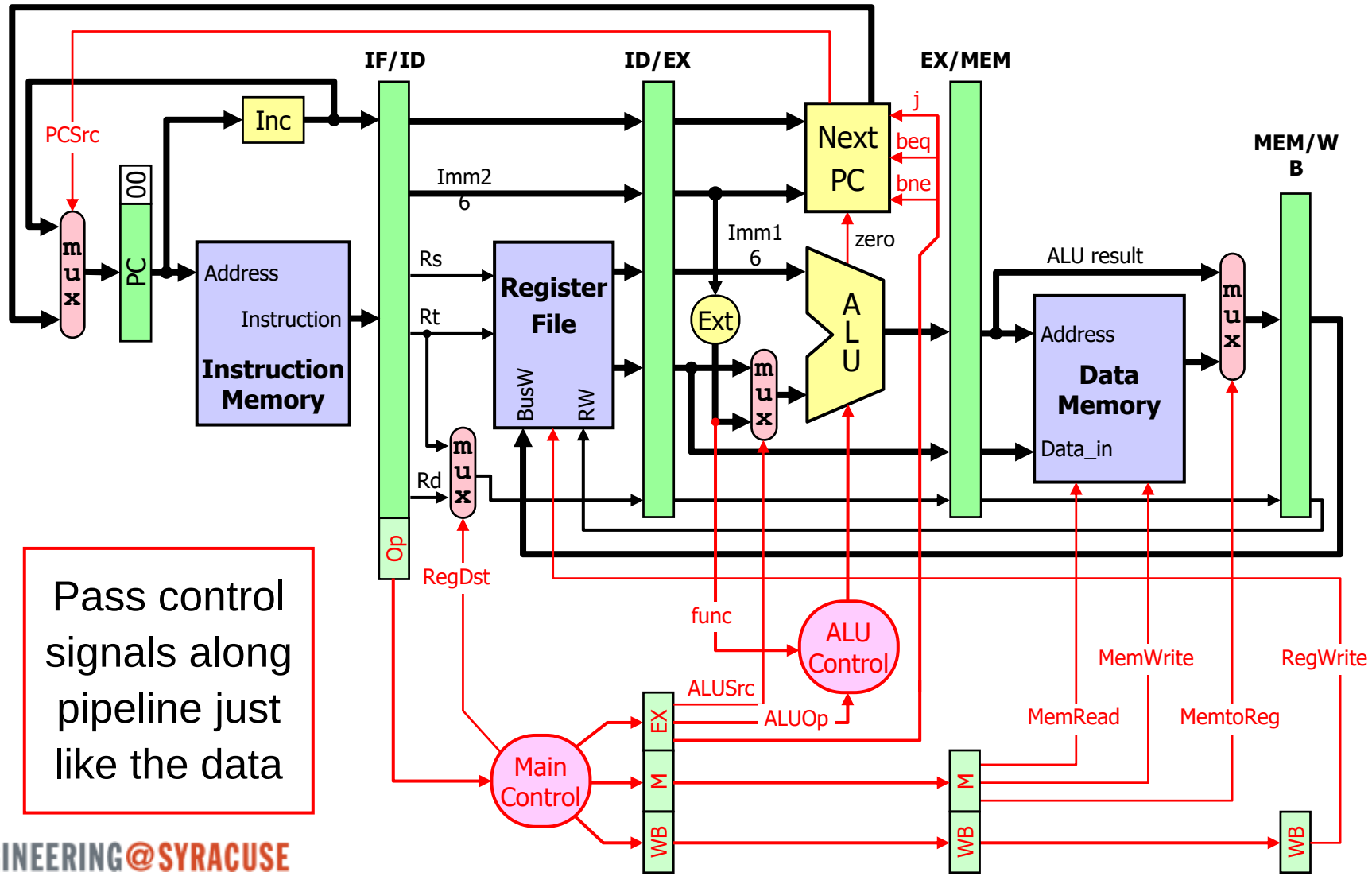
- The function of pipeline
  - Improve throughput not latency
  - Increase parallelism (by placing different hardware resource in different stages and using them simultaneously)

- Pipeline speedup
  - We may not achieve the ideal speed up
  - Due to unbalanced pipeline stage

Pipeline

- 5 Stage pipeline architecture
  - What hardware resource is involved in each stage?
  - Which control signal and data are propagated to each stage?
  - Pipeline diagram of instruction executions
- Hazard
  - Understand structure hazards
- Able to identify data dependency and data hazard
  - Able to resolve data hazard by
    - Pipeline stall
    - Data forwarding

Pipelined Control



Pass control signals along pipeline just like the data

Data hazards

- Hazard detection and data forwarding
  - Understand the concept
  - Identify the instructions that will trigger hazard detection and data forwarding
  - Able to fill pipeline diagram for processors with or without hazard detection and data forwarding (i.e. understand how differently these two processors work)

Control hazards

- Understand that when the branch condition is evaluated can impact performance
- Knowing how branch prediction works, why it improves performance
- Understand the function of delay slot

Memory hierarchy

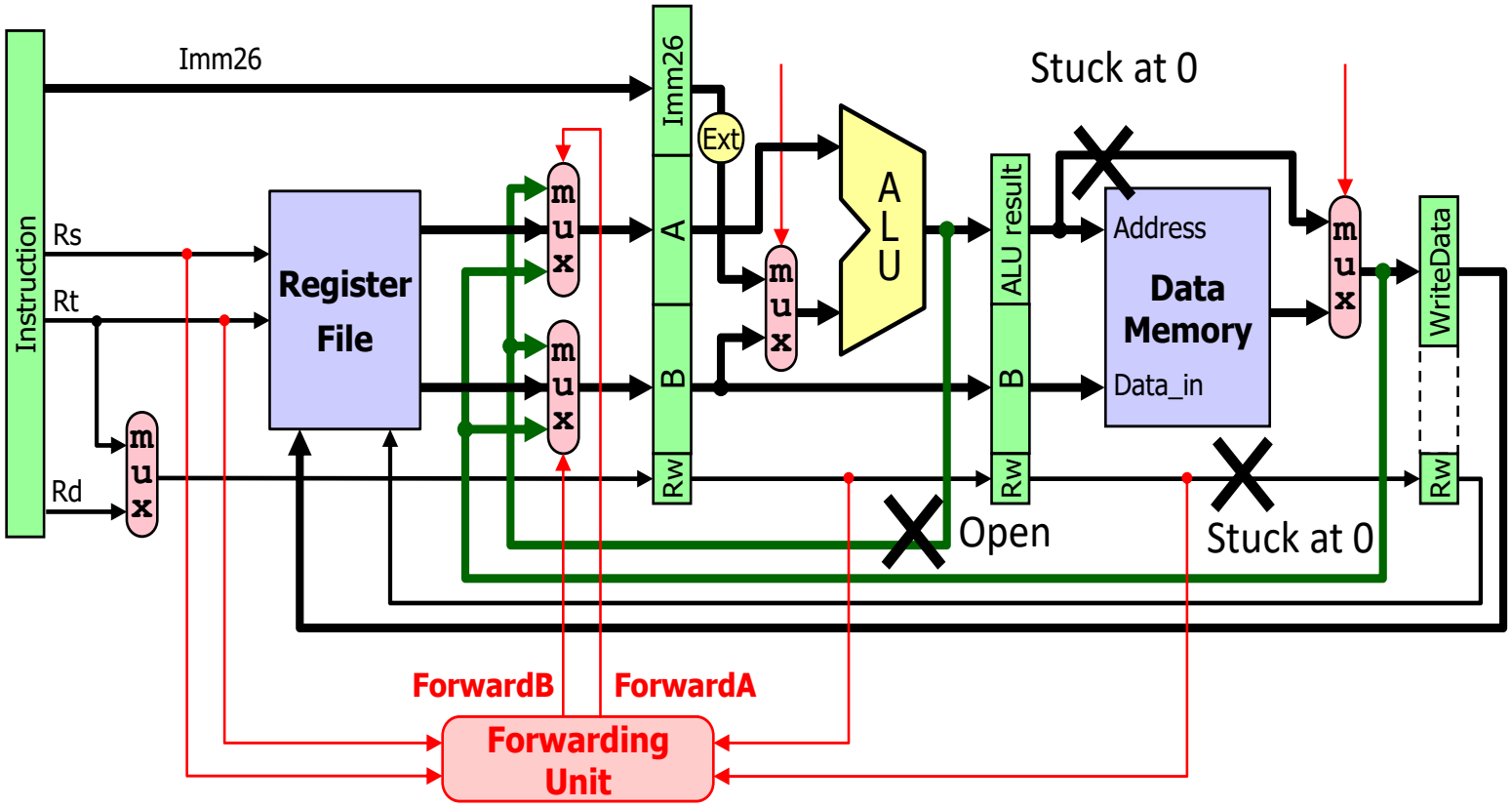
- Be able to compare DRAM (main memory), SRAM (cache) and hard disk (cost, performance, usage)
- Understand the purpose of creating memory hierarchy (fast and large), and how it works

Cache Structure

- Understand the difference between direct mapped, n-way associative and fully associative cache
- Given an address, identify the cache location
- Given cache structure, calculate the number of tag bits needed, and the size of the SRAM needed to implement the cache
- Understand write-through v.s. write back, write allocate v.s. non-write allocate, and be able to use them
- Understand LRU replacement policy and able to use it
- Able to calculate the performance impact (CPI, AMAT) of cache structure
- Able to analyze miss rate and hit rate

Virtual memory

- Understand the function of page, page table, TLB
- Understand the difference between page fault & cache miss
- Understand the virtual address translation procedure


Instruction
Rs
Rt
Rd
Imm26
Register File
mux
Imm26
A
B
Rw
Ext
ALU
ALU result
Stuck at 0
Address
Data Memory
Data_in
WriteData
Open
Stuck at 0
ForwardB
ForwardA
Forwarding Unit

ENGINEERING@SYRACUSE