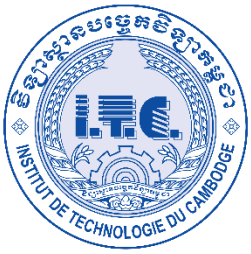


Institute of Technology of Cambodia
Telecommunication and Network

Computer Architecture
Group 10

TP 05 Memory

Name	ID	Score
Neath Morokot	e20220575	…….
Chhit Chantola	e20221273	…….
Kith Sereyvibol	e20221328	…….
An Vanneath	e20220208	…….

Lecture: Course: Mr.CHUN Thavorac
TP: Mr.OL Phearun

Academic Year: 2023-2024

Questions:

1) Which is faster, SRAM or DRAM?
2) What are the advantages of using DRAM for main memory?

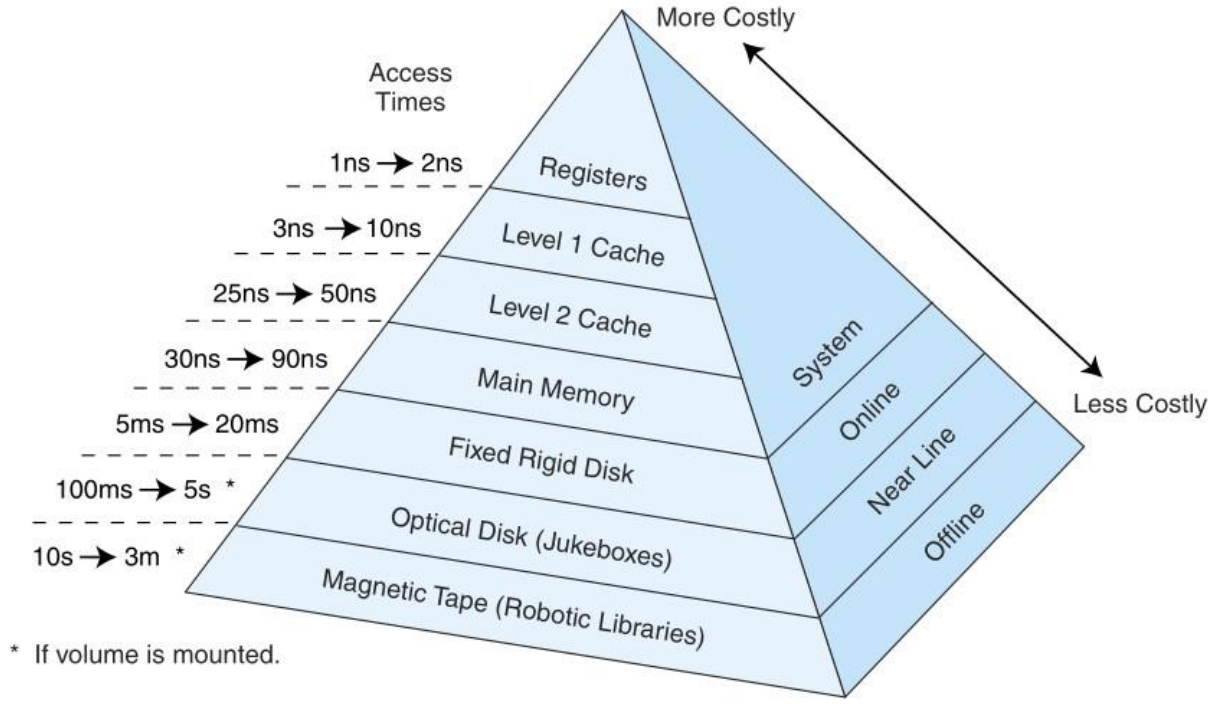


FIGURE 6.1 The Memory Hierarchy

3) Name three different applications where ROMs are often used.
4) Explain the concept of a memory hierarchy.
5) Which of L1 or L2 cache is faster? Which is smaller? Why is it smaller?
6) What is a page fault?
7) What is difference between a virtual memory address and physical memory address? Which is larger? Why?
8) What is a TLB and how does it improve EAT?
9) Discuss the pros and cons of paging.
10) What are the advantages and disadvantages of virtual memory?

Exercise:

1) Suppose a computer using direct mapped cache has 2^{20} words of main memory and a cache of 32 blocks, where each cache block contains 16 words.

a) How many blocks of main memory are there?
b) What is the format of a memory address as seen by the cache, that is, what are the sizes of the tag, block, and word fields?
c) To which cache block will the memory reference 0DB6316 map?

2) Suppose a computer using fully associative cache has 224words of main memory and a cache of 128 blocks, where each cache block contains 64 words.

a) How many blocks of main memory are there?
b) What is the format of a memory address as seen by the cache, that is, what are the sizes of the tag and word fields?
c) To which cache block will the memory reference 01D87216 map?

3) Consider a byte-addressable computer with 24-bit addresses, a cache capable of storing a total of 64KB of data, and blocks of 32 bytes. Show the format of a 24-bit memory address for:

a) direct mapped
b) associative
c) 4-way set associative

4) A direct-mapped cache consists of eight blocks. Main memory contains 4K blocks of eight words each. Access time for the cache is 22ns and the time required to fill a cache slot from main memory is 300ns. (This time allows us to determine the block is missing and bring it into cache.) Assume a request is always started in parallel to both cache and to main memory (so if it is not found in cache, we do not have to add this cache search time to the memory access). If a block is missing from cache, the entire block is brought into the cache and the access is restarted. Initially, the cache is empty.

a) Show the main memory address format that allows us to map addresses from main memory to cache. Be sure to include the fields as well as their sizes.
b) Compute the hit ratio for a program that loops 4 times from location 0 to 6710 in memory.
c) Compute the effective access time for this program.

5) A system implements a paged virtual address space for each process using a one level page table. The maximum size of virtual address space is 16MB. The page table for the running process includes the following valid entries (the → notation indicates that a virtual page maps to the given page frame, that is, it is located in that frame):

Virtual page 2 → Page frame 4
Virtual page 4 → Page frame 9
Virtual page 1 → Page frame 2
Virtual page 3 → Page frame 16
Virtual page 0 → Page frame 1

The page size is 1024 byte and the maximum physical memory size of the machine is 2MB.

a) How many bits are required for each virtual address?
b) How many bits are required for each physical address?
c) What is the maximum number of entries in a page table?
d) To which physical address will the virtual address 1524_{10} translate?
e) Which virtual address will translate to physical address 1024_{10}?

6) Given a virtual memory system with a TLB, a cache, and a page table, assume the following:

- A TLB hit requires 5ns.
- A cache hit requires 12ns.
- A memory reference requires 25ns.
- A disk reference requires 200ms (this includes updating the page table, cache, and TLB).
- The TLB hit ratio is 90%.
- The cache hit rate is 98%.
- The page fault rate is .001%.
- On a TLB or cache miss, the time required for access includes a TLB and/or cache update, but the access is not restarted.
- On a page fault, the page is fetched from disk, all updates are performed, but the access is restarted.
- All references are sequential (no overlap, nothing done in parallel).

For each of the following, indicate whether or not it is possible. If it is possible, specify the time required for accessing the requested data.

a) TLB hit, cache hit
b) TLB miss, page table hit, cache hit
c) TLB miss, page table hit, cache miss
d) TLB miss, page table miss, cache hit
e) TLB miss, page table miss

Write dow the equation to calculate the effective access time.

Answer:

1) Which is faster, SRAM or DRAM?

_ SRAM (Static Random-Access Memory) is generally faster than DRAM (Dynamic Random-Access Memory) in terms of access speed.

2) What are the advantages of using DRAM for main memory?

_ The advantages of using DRAM for main memory are:

+ It is much denser (can store many bits per chip).
+ Uses less power.
+ Generates less heat than SRAM.

3) Name three different applications where ROMs are often used.

_ The three names different applications where ROMs are often used:

+ *PROM (programmable read-only memory*) is a variation on ROM.
+ *EPROM* (*erasable PROM*) is programmable with the added advantage of being reprogrammable (erasing an EPROM requires a special tool that emits ultraviolet light).
+ *EEPROM* (*electrically erasable PROM*) removes many of the disadvantages of EPROM: no special tools are required for erasure (this is performed by applying an electric field) and you can erase only portions of the chip, one byte at a time.

4) Explain the concept of a memory hierarchy

The memory hierarchy in computing balances speed, cost, and size:

1.Registers:
Access Time: 1-2 ns
Description: Fastest, smallest memory inside the CPU.

2.Level 1 Cache (L1 Cache):
Access Time: 3-10 ns
Description: Small, fast memory close to the CPU.

3.Level 2 Cache (L2 Cache):
Access Time: 25-50 ns
Description: Larger than L1, buffering between L1 and RAM.

4.Main Memory (RAM):
Access Time: 30-90 ns
Description: Volatile memory for active data use.

5.Fixed Rigid Disk (Hard Disk Drive):
Access Time: 5-20 ms
Description: Non-volatile storage for large data.

6.Optical Disk:
Access Time: 100 ms-5 s
Description: Storage like CDs/DVDs.

7.Magnetic Tape:
Access Time: 10 s-3 m
Description: Archival and backup storage.

5) Which of L1 or L2 cache is faster? Which is smaller? Why is it smaller?

_ L1 cache resides on the processor, whereas L2 cache resides between the CPU and main memory. L1 cache is, therefore, faster than L2 cache.
_ A typical personal computer's level 2 (L2) cache is 256K or 512K. Level 1 (L1) cache is smaller, typically 8K or 16K.
_ L1 is smaller because:

- **Cost**: SRAM used in L1 cache is more expensive to manufacture compared to the DRAM used in main memory or L2 cache. The smaller size of L1 cache helps manage the overall cost of the CPU while still providing significant performance benefits.
- **Physical Space**: L1 cache needs to be physically close to the CPU cores to minimize latency. This proximity limits the available space on the CPU chip for L1 cache.

- **Speed Requirements**: L1 cache must maintain very fast access times to keep up with the CPU's demand for data and instructions. Smaller caches can be designed using faster SRAM technologies, which helps achieve the necessary speed.

6) What is a page fault?

_ A page fault is an exception or interrupt that occurs when a program or process attempts to access a page of memory that is currently not located in physical RAM (Random-Access Memory). In virtual memory systems, which are used by modern operating systems, programs access memory through virtual addresses that are mapped to physical addresses by the memory management unit (MMU) of the CPU.

7) What is difference between a virtual memory address and physical memory address? Which is larger? Why?

_ Difference between a virtual memory address and physical memory address:

• Virtual memory address:
  + **Purpose**: Virtual memory addresses are used by programs (processes) to access memory.
  + **Translation**: These addresses are translated by the Memory Management Unit (MMU) of the CPU into physical memory addresses.
  + **Size**: The size of a virtual memory address depends on the architecture and the operating system, but it's typically larger than a physical memory address.
  + **Management**: Virtual memory addresses are managed by the operating system, which ensures that each process has its own isolated virtual address space.

• Physical memory address:
  + **Purpose**: Physical memory addresses represent the actual locations in the computer's RAM where data is stored.
  + **Direct Access**: Unlike virtual addresses, physical addresses do not undergo translation by the MMU.
  + **Size**: The size of a physical memory address is determined by the computer's memory architecture. In most modern systems, physical addresses are typically the same size as the data bus width (e.g., 32-bit or 64-bit), allowing direct access to memory modules.

+ **Control**: Physical memory addresses are under the direct control of the hardware and are managed by the memory management hardware (MMU in conjunction with the memory controller).

8) What is a TLB and how does it improve EAT?

_ TLB stands for Translation Lookaside Buffer. It is a hardware cache used in modern computer systems, primarily by the Memory Management Unit (MMU) of the CPU, to improve the efficiency of virtual memory translation and thus enhance Effective Access Time (EAT) for memory operations.

_ Is improve EAT:
  + **Reduced Latency**: By storing frequently used virtual-to-physical address mappings in a small

and fast cache (the TLB), the MMU can quickly retrieve the necessary physical address for memory operations. This reduces the latency associated with memory accesses, improving the Effective Access Time (EAT).

+ **Efficient Use of Memory**: TLBs allow the system to efficiently manage virtual memory translations without having to access the slower main memory for every memory access. This is crucial for maintaining high performance in modern computer systems where multiple processes are running concurrently.
+ **Caching Mechanism**: The TLB acts as a cache for virtual-to-physical address translations, leveraging the principles of locality (both temporal and spatial). This means that recently accessed translations and translations from nearby addresses are likely to be found in the TLB, optimizing performance further.

9) Discuss the pros and cons of paging.

_ Paging in operating systems offers streamlined memory management but presents challenges like internal fragmentation and the overhead of managing page tables.

10) What are the advantages and disadvantages of virtual memory?

_ Advantages of virtual memory:

- Increased Memory Capacity:
  Virtual memory allows the system to run larger applications or multiple applications simultaneously by using disk space to extend the available RAM. This makes it possible to work with datasets larger than the physical memory.
- Memory Isolation and Protection:
  Virtual memory provides isolation between processes, ensuring that one process cannot interfere with the memory of another. This enhances security and stability by preventing accidental or malicious memory access violations.
- Efficient Use of Physical Memory:
  By swapping out less frequently used pages to disk, virtual memory ensures that the physical RAM is used more efficiently for active data and instructions, improving overall system performance.
- Simplified Memory Management:
  Virtual memory abstracts the physical memory layout from applications, allowing developers to work with a large, contiguous address space without worrying about the actual physical memory layout.
- Support for Multitasking:
  Virtual memory enables efficient multitasking by allowing multiple processes to run concurrently, each with its own virtual address space. This improves system responsiveness and utilization.
- Program Relocation and Flexibility:
  Virtual memory allows programs to be loaded into any available memory location without modification, facilitating program relocation and flexible memory usage.

_ Disadvantages of virtual memory:

- Performance Overhead:

Accessing data from disk (swap space) is much slower than accessing data from RAM. Frequent swapping, known as thrashing, can lead to significant performance degradation as the system spends more time swapping pages in and out of memory than executing actual tasks.

- Increased Complexity:
  Implementing virtual memory requires sophisticated hardware (MMU) and software support, including page tables, page replacement algorithms, and handling page faults. This adds complexity to the system.
- Disk Space Usage:
  Virtual memory uses disk space to store swapped-out pages. If the swap space is insufficient, the system may run out of virtual memory, leading to application crashes or degraded performance.
- Internal and External Fragmentation:
  Although virtual memory helps manage memory more efficiently, it can still lead to fragmentation issues. Internal fragmentation occurs when allocated memory is not fully used, and external fragmentation can occur in the swap space on disk.
- Overhead of Page Fault Handling:
  Handling page faults (when a required page is not in memory) introduces overhead, as the operating system must retrieve the page from disk and update the page tables. This process can be time-consuming and affect system performance.
- Resource Management:
  Managing virtual memory requires careful tuning of system resources, such as configuring appropriate swap space size and optimizing page replacement algorithms to balance performance and resource utilization.

Exercise:

1.

a) Find the number blocks of main memory

$Number\ of\ blocks\ of\ main\ memory = Total\ words\ of\ main\ memory\ /\ words\ per\ block$

We have:

● main memory has 2 words 20

● Each cache block contains 16 = 2 words. 4

So, $Number\ of\ blocks\ of\ main\ memory$ = 220/ 24 = 216 = 65536 $blocks$

b) Find size of the tag, block, words field.

We have:

● **Word** : Since each cache block contains 16 = 2 , so we need 4 bits to identify a 4 specific word within a block

● **Block** : since a cache has 32 = 25 blocks , so we need 5 bits to block into the cache

● **Tag** : The remaining bits will be used for the tag.

The total number of bits in the address is 20 (since we have 2 words of memory). 20

Tag bits = Total number of bits - (Block bits + Word bits)

= 20 - (5 + 4)

= 11 bits

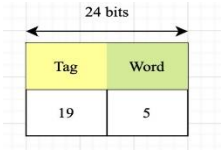
So, the memory address format is:

Tag	Block	Word
11	5	4

c) To which cache block will the memory reference 0DB6316 map?
We have:
● 0DB6316 = 1101 1011 0110 00112
This binary number should be 20 bits long 0DB6316 = 0000 1101 1011 0110 00112
● Tag (11 bits): 0000 1101 101
● Block (5 bits): 1 0110
● Word offset (4 bits): 0011
So, the Block 101102 = 2210
Thus, the memory reference 0DB6316 maps to cache block 22.
2.
a) Find the number blocks of main memory
$Number\ of\ blocks\ of\ main\ memory = Total\ words\ of\ main\ memory\ /\ words\ per\ block$
We have:
● main memory has 2 words 24
● Each cache block contains 64 = 2 words. 6
So, $Number\ of\ blocks\ of\ main\ memory$ = 224/ 26 = 218 = 262144 $blocks$
b) Find size of the tag, words field.
We have:
Since the cache is fully associative, the memory address format will include only a tag and a word field. The word field is needed to identify the word within a block, and the tag is used to identify the block itself.
● **Word** : Since each cache block contains 64 = 2 , so we need 6 bits to identify a 6 specific word within a block
● **Tag** : The remaining bits will be used for the tag.
The total number of bits in the address is 24 (since we have 2 words of memory). 24
Tag bits = Total number of bits - Word bits
= 24 - 6
= 18 bits
So, the memory address format is: Tag: 18 bits and Word: 6 bits.
c) To which cache block will the memory reference 0DB6316 map?
We have:
● 01D87216 = 11 1011 0000 1110 0102
The binary number should be 24 bits long 01D87216 = 0000 0011 1011 0000 1110 0102
● Tag (18 bits): 0000 0011 1011 0000 11
● Word (6 bits): 10 010
Since the cache is fully associative, any block can go into any cache block, so the memory reference 0DB6316 can be mapped to any of the 128 cache blocks depending on the cache's replacement policy.
3. Show the format of a 24-bit memory address for:

We have
● Total cache size = 64KB = 64 x 1024 = 65536 B
● Block size = 32 B
$Number\ block\ in\ cache = Total\ cache\ size\ /\ Block\ size$
= 65536 / 32 = 2048 $blocks$
a) direct mapped
● Block : Number of block = log2(2048) = log2 (211) = 11 bits
● Word : Since each block is 32 = 25 bytes, we need 5 bits to address each byte within a block
● Tag : The remaining bits will be used for the tag.
Tag bits = Total address size - (Block bits + Word bits)
= 24 bits - (11 + 5) = 8 bits
Therefore, the format of a 24-bit memory address for direct mapped is :

b) Associative
In fully associative, there is no block since any block can be placed anywhere in the cache.
● Word : number of words = log2 (32)=5 bits
● **Tag** : The remaining bits will be used for the tag.
The total number of bits in the address is 24 (since we have 2 words of memory). 24
Tag bits = Total address size - Word bits
= 24 - 5
= 19 bits
Thus, the format of a 24-bit memory address for associative is :

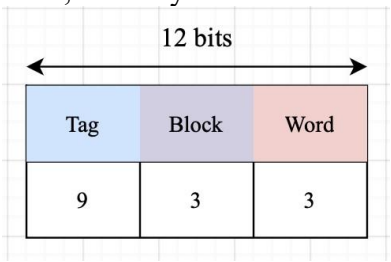


c) 4-way set associative
The number of sets is the total number of blocks divided by the associativity (4-way):
● Set : $Number\ of\ set = number\ of\ blocks\ in\ cache/number\ of\ way$
= 2048 / 4 = 512 $bits$
● Block : $block\ size = log2\ (\ number\ of\ set\ )$
= $log2\ (\ 512\ ) = log\ (\ 2\text{^}9\ ) = 9\ bits$
● Tag : Tag size = total address size - block size - word size
= 24 - 9 - 5 = 10 bits
Thus, the format of a 24-bit memory address for 4-way set associative is :

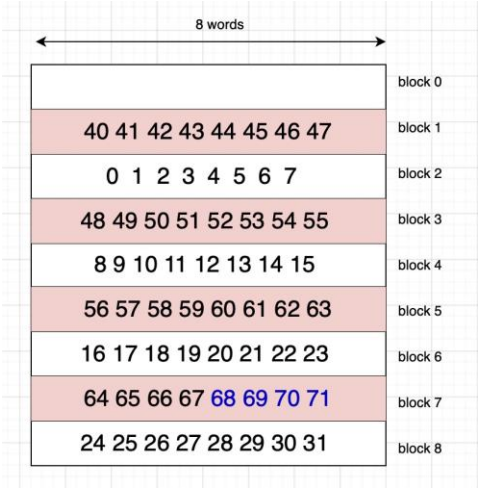
4. Given
   - Main memory contains 4K blocks of 8 words each.
   - Access time for the cache is 22ns
   - The time required to fill a cache slot from main memory is 300ns. ● A direct-mapped cache consists of eight blocks.

a) Show the main memory address format
   - 4K blocks of 8 words = 4 x 1024 x 8 = 2^{15} blocks
     So, total address size = log2 (4096) = log (2^{12}) = 12 bits
   - Word : Each block contains 8 words. So , word size = log2 (8) = 3 bits
   - Block : The cache has 8 blocks. So, block size = log2 (8) = 3 bits ● Tag: Tag size = Total address size - block size - word size
     = 15 - 3 - 3 = 9 bits

Thus, memory address format is:



b) Compute the hit ratio for a program that loops 4 times from location 0 to 67_{10} in memory.



   - Each block of 8 words is accessed.

- The program loops 4 times from location 0 to 67. Total addresses accessed in one loop = 68
- Total number of accessed = 68 x 4 = 272
- Since the cache is empty, the first loop will be missed.
- Loop1: hit = 7 + 7 + 7 + 7 +7 +7 + 7 + 7 + 7 + 3 = 59
  Miss = 1 + 1 + 1 + 1 + 1 + 1 + 1 +1 + 1 = 9
- Loop 2 : hit = 68
  Miss = 0
- Loop 3 : hit = 68
  Miss = 0
- Loop 4 : hit = 68
  Miss = 0

Number of hits = 59 + 68 + 68 + 68 = 263
The hit ratio = 253 / 272 = 0.96 = 96 %

c) Compute the effective access time for this program.

- Cache access time: 22ns
- Main memory access time (on a miss): 300ns
- Hit ratio = 0.96
- Miss ratio = 1 - hit ratio = 1 - 0.96 = 0.04

EAT = (Hit ratio×Cache access time)+(Miss ratio×Memory access time)
= (0.96 x 22ns) + (0.04 x 300ns)
= 32.12 ns

Therefore, the effective access time for this program is 32.12 ns.

5.Given

- Virtual page 2 → Page frame 4 Virtual page 4 → Page frame 9
- Virtual page 1 → Page frame 2 Virtual page 3 → Page frame 16
- Virtual page 0 → Page frame 1
- The page size is 1024 byte and the maximum physical memory size of the machine is 2MB.
- The maximum size of virtual address space is 16MB. a) How many bits are required for each virtual address?
- Virtual address space has 16MB = 16 x 2^{10} x 2^{10} = 2^{24} byte ⇒ $Log_2(2^{24}) = 24$ bits

There are 24 bits required for each virtual address.

b)How many bits are required for each physical address?

- The maximum physical memory size is 2MB with a page size of 1024 bytes

2MB = 2 x 2^{10} x 2^{10} = 2^{21} byte ⇒$\log_2(2^{21}) = 21$ bits

There are 21 bits required for physical address.

c)What is the maximum number of entries in a page table?

- Virtual page = 2^{24}
- Page size = 1024 = 2^{10}

$The\ maximum\ number\ of\ entries\ in\ a\ page\ table = 2^{24}_{14} / 2^{10}\ page / page\ size$

$= 2 = 16384$ pages

d)To which physical address will the virtual address 1524_{10} translate?

- Virtual address : $1524_{10} = 10111110100_2$
- Physical address = 21 bits
- Page size = 10 bits

=> offset = page size = 10 bits

⇒ Frame = physical address - page size = 21 - 10 = 11 bits

Fram (11 bits)	offset (10 bits)
00000000001	0111110100

- Fram = $00000000001_2 = 1_{10}$

So, Virtual page 1 → Page frame 2

- Offset = $0111110100_2 = 0+256+128+64+32+16+0+4+0+0 = 500$ ● Physical address = (Page frame number × Page size) + Offset

= 2 x 1024 +500 = 2548

The virtual address 1524 translates to the physical address **2548**.

e)Which virtual address will translate to physical address 1024?

- physical address 1024 = 10000000000_2 ● Virtual page = 24 bits

⇒ offset = page size = 10 bits

⇒ Frame = physical address - page size = 24 - 10 = 14 bits

Fram (11 bits)	offset (10 bits)
0001	0000000000

- Fram = $0001_2 = 1_{10}$

So, Virtual page 1 → Page frame 1

- Offset = 0
- Page frame 1→Virtual page 0
- Virtual address = (Virtual page number×Page size)+Offset Virtual address

= (0 x 1024) + 0 = 0

So, the physical address 1024 translates to virtual address 0.

6.Given a virtual memory system with a TLB, a cache, and a page table, assume the following:

- A TLB hit requires 5ns.
- A cache hit requires 12ns.
- A memory reference requires 25ns.
- A disk reference requires 200ms (this includes updating the page table, cache, and TLB).
- The TLB hit ratio is 90%.
- The cache hit rate is 98%.
- The page fault rate is .001%.
- On a TLB or cache miss, the time required for access includes a TLB and/or cache update, but the access is not restarted.
- On a page fault, the page is fetched from disk, all updates are performed, but the access is restarted.
- All references are sequential (no overlap, nothing done in parallel).

For each of the following, indicate whether or not it is possible. If it is possible, specify the time required for accessing the requested data.

f) TLB hit, cache hit

g) TLB miss, page table hit, cache hit

h) TLB miss, page table hit, cache miss

i) TLB miss, page table miss, cache hit

j) TLB miss, page table miss

Write down the equation to calculate the effective access time.