System Software, Architecture, Memory and Storage Practice Test (Sample)

Study Guide

BY EXAMZIFY

Everything you need from our exam experts!

Table of Contents

Introduction

Preparing for a certification exam can feel overwhelming, but with the right tools, it becomes an opportunity to build confidence, sharpen your skills, and move one step closer to your goals. At Examzify, we believe that effective exam preparation isn't just about memorization, it's about understanding the material, identifying knowledge gaps, and building the test-taking strategies that lead to success.

This guide was designed to help you do exactly that.

Whether you're preparing for a licensing exam, professional certification, or entry-level qualification, this book offers structured practice to reinforce key concepts. You'll find a wide range of multiple-choice questions, each followed by clear explanations to help you understand not just the right answer, but why it's correct.

The content in this guide is based on real-world exam objectives and aligned with the types of questions and topics commonly found on official tests. It's ideal for learners who want to:

- **Practice answering questions under realistic conditions,**
- **Improve accuracy and speed,**
- **Review explanations to strengthen weak areas, and**
- **Approach the exam with greater confidence.**

We recommend using this book not as a stand-alone study tool, but alongside other resources like flashcards, textbooks, or hands-on training. For best results, we recommend working through each question, reflecting on the explanation provided, and revisiting the topics that challenge you most.

Remember: successful test preparation isn't about getting every question right the first time, it's about learning from your mistakes and improving over time. Stay focused, trust the process, and know that every page you turn brings you closer to success.

Let's begin.

How to Use This Guide

This guide is designed to help you study more effectively and approach your exam with confidence. Whether you're reviewing for the first time or doing a final refresh, here’s how to get the most out of your Examzify study guide:

1. Start with a Diagnostic Review

Skim through the questions to get a sense of what you know and what you need to focus on. Your goal is to identify knowledge gaps early.

2. Study in Short, Focused Sessions

Break your study time into manageable blocks (e.g. 30 – 45 minutes). Review a handful of questions, reflect on the explanations.

3. Learn from the Explanations

After answering a question, always read the explanation, even if you got it right. It reinforces key points, corrects misunderstandings, and teaches subtle distinctions between similar answers.

4. Track Your Progress

Use bookmarks or notes (if reading digitally) to mark difficult questions. Revisit these regularly and track improvements over time.

5. Simulate the Real Exam

Once you're comfortable, try taking a full set of questions without pausing. Set a timer and simulate test-day conditions to build confidence and time management skills.

6. Repeat and Review

Don’t just study once, repetition builds retention. Re-attempt questions after a few days and revisit explanations to reinforce learning. Pair this guide with other Examzify tools like flashcards, and digital practice tests to strengthen your preparation across formats.

There’s no single right way to study, but consistent, thoughtful effort always wins. Use this guide flexibly, adapt the tips above to fit your pace and learning style. You've got this!

Questions

1. Which register holds the data fetched from memory or to be written to memory?

A. Memory Address Register

B. Arithmetic Logic Unit

C. Memory Data Register

D. Cache

2. What is the role of the write-ahead log in a journaling file system?

A. It records intended changes after applying to storage.

B. It logs user access patterns.

C. It records intended changes before applying; upon crash, logs are replayed to restore consistency.

D. It stores encryption keys for data blocks.

3. What are accumulators?

A. Stores the results of memory fetch

B. Holds the status flags

C. Keeps instruction history

D. Holds the results of calculations and logic operations that the ALU performs

4. Which of the following best describes the purpose of the Translation Lookaside Buffer (TLB) in memory management?

A. Total Load Buffer

B. Time Latency Buffer

C. Translation Lookaside Buffer; caches recent virtual-to-physical address translations.

D. Memory protection key

5. Which statement correctly describes Level 1, Level 2, and Level 3 caches?

A. Level 1 is the fastest and has the smallest capacity.

B. Level 2 is slower than L1 but holds more.

C. Level 3 is the slowest and has the highest capacity.

D. Level 2 is faster than Level 1 and has less capacity.

6. Describe the cache memory hierarchy and the MESI cache coherence protocol used in multi-core systems.

- **A. The cache hierarchy consists of L1, L2, and L3 caches between the CPU and memory.**
- **B. Modified, Exclusive, Shared, Invalid; coherence ensures all cores see consistent data; transitions handle writes and cache lines.**
- **C. MESI states are Modified, External, Shared, Internal.**
- **D. Cache coherence is unnecessary in multi-core systems.**

7. Which filesystem is known for copy-on-write with checksums, snapshots, and pooled storage?

- **A. ext4**
- **B. NTFS**
- **C. ZFS**
- **D. FAT32**

8. In the boot sequence, which component loads the kernel into memory after the bootloader has prepared the environment?

- **A. BIOS/UEFI firmware loads the kernel.**
- **B. The bootloader loads the kernel into memory.**
- **C. The kernel loads the bootloader.**
- **D. POST transfers control to the kernel.**

9. Which statement describes the CPU's purpose?

- **A. Memory management unit handles paging**
- **B. Graphics Processing Unit handles rendering**
- **C. Central Processing Unit is responsible for executing a sequence which involves taking inputs from an input device, processing the input and outputting the results.**
- **D. Network interface card handles data transfer**

10. Which statement best describes the distinction between kernel space and user space?

A. Kernel space has privileged access to hardware and all memory; user space runs with restricted privileges; Separation protects the OS from errant or malicious user code and enforces security boundaries; context switches via system calls.

B. User space has privileged access to hardware and all memory; kernel space runs with restricted privileges; Separation is mainly for organization.

C. Both spaces have unrestricted privileges; separation is only for debugging.

D. Kernel space executes in user mode while user space executes in supervisor mode.

Answers

1. C
2. C
3. D
4. C
5. A
6. B
7. C
8. B
9. C
10. A

Explanations

1. Which register holds the data fetched from memory or to be written to memory?

A. Memory Address Register

B. Arithmetic Logic Unit

C. Memory Data Register

D. Cache

The main idea is that there is a dedicated data buffer between the CPU and memory. The register that holds the data being transferred to or from memory is the Memory Data Register. When the CPU fetches something from memory, the address to access is placed in the Memory Address Register, memory retrieves the data and places it into the Memory Data Register so the CPU can use it. For writes, the value to be stored goes into the Memory Data Register, and the memory uses the address in the Memory Address Register to perform the write. The Arithmetic Logic Unit isn't used to hold memory data; it performs computations. Cache isn't the single register used in the transfer; it's a fast storage layer that holds copies of memory blocks to speed up access, not the immediate buffer for a single fetch or write operation.

2. What is the role of the write-ahead log in a journaling file system?

A. It records intended changes after applying to storage.

B. It logs user access patterns.

C. It records intended changes before applying; upon crash, logs are replayed to restore consistency.

D. It stores encryption keys for data blocks.

The main idea is crash resilience: a journaling file system uses a write-ahead log to record what will be changed before those changes actually go to disk, so the system can recover to a consistent state after a crash. When a write is issued, the filesystem first writes a log entry describing the pending modifications (which blocks or metadata will be updated and how) and ensures that this log is safely stored on disk. Only after the log is durable does the system apply the actual data or metadata changes. If the system crashes, the log is replayed on reboot to finish any pending updates or roll back incomplete ones, bringing the filesystem to a consistent state. This approach provides atomicity: every operation either fully happens or is undone, preventing partial, inconsistent updates. Some systems journal only metadata, while others journal both metadata and data for stronger guarantees. Choices about logging user access patterns or storing encryption keys don't fit this crash-recovery role, and logging changes after applying wouldn't protect against crashes in the middle of an update.

3. What are accumulators?

A. Stores the results of memory fetch

B. Holds the status flags

C. Keeps instruction history

D. Holds the results of calculations and logic operations that the ALU performs

An accumulator is a dedicated register in many CPU designs that holds the results produced by the arithmetic logic unit. After the ALU performs an arithmetic or logic operation, its result is written into the accumulator so that the most recent computation is readily available for the next operation or can be stored back to memory. This makes chaining calculations straightforward, since you keep the current result in one central place. It isn't used for memory fetch results or for history of instructions, which are handled by other parts of the system, and status flags live in a separate flag register. So the accumulator's role is to hold the results of calculations and logic operations that the ALU performs.

4. Which of the following best describes the purpose of the Translation Lookaside Buffer (TLB) in memory management?

A. Total Load Buffer

B. Time Latency Buffer

C. Translation Lookaside Buffer; caches recent virtual-to-physical address translations.

D. Memory protection key

A Translation Lookaside Buffer speeds memory access by caching recent virtual-to-physical address translations, so the processor doesn't have to walk the page table for every memory reference. When the CPU needs to access a memory location, it must translate the virtual address to a physical frame number. Doing this by repeatedly reading the page table from memory is slow, especially since page tables can be multi-level. The TLB sits between the CPU and memory and stores a small number of these translations. If the needed mapping is in the TLB (a hit), the physical address is obtained quickly and the access proceeds. If it's not in the TLB (a miss), the system must fetch the translation from the page table in memory, possibly perform a multi-level walk, and then refill the TLB with the new translation for future accesses. This cache of translations dramatically reduces latency for programs with locality, which is why virtual memory performance hinges on a well-functioning TLB. The other options don't describe this mechanism. They refer to concepts that aren't involved in translating virtual addresses to physical addresses or are unrelated features.

5. Which statement correctly describes Level 1, Level 2, and Level 3 caches?

A. Level 1 is the fastest and has the smallest capacity.

B. Level 2 is slower than L1 but holds more.

C. Level 3 is the slowest and has the highest capacity.

D. Level 2 is faster than Level 1 and has less capacity.

The main idea tested here is how CPU caches are arranged by speed and size. In a typical cache hierarchy, the closest cache to the processor (L1) is built to be extremely fast, but it has only a small capacity. As you move to L2 and then L3, the caches get larger to hold more data, but their access times increase, making them slower than the closer levels. This speed-versus-size trade-off is what makes the statement that L1 is the fastest and has the smallest capacity the most accurate descriptor of the hierarchy. L1 being the fastest and smallest captures the essential relationship across the levels: it serves the most frequently accessed data with minimal latency; L2 is slower but larger; L3 is even larger and typically slower, though still faster than main memory. The other statements mention aspects of the other levels, but the core characteristic that defines the hierarchy is the L1 fast-and-small property.

6. Describe the cache memory hierarchy and the MESI cache coherence protocol used in multi-core systems.

A. The cache hierarchy consists of L1, L2, and L3 caches between the CPU and memory.

B. Modified, Exclusive, Shared, Invalid; coherence ensures all cores see consistent data; transitions handle writes and cache lines.

C. MESI states are Modified, External, Shared, Internal.

D. Cache coherence is unnecessary in multi-core systems.

Understanding how multiple caches stay in sync is essential when designing or analyzing multi-core systems, and this hinges on how data moves through the cache hierarchy and how coherence is maintained. The MESI protocol manages the state of each cached line to coordinate access across cores, keeping everything consistent. The key states are Modified, Exclusive, Shared, and Invalid. Modified means the cache line has been written to and is dirty; it is the only copy and memory has stale data, so a write-back is needed when evicted. Exclusive means the line is clean (matches memory) and is the only copy in any cache. Shared means the line is clean and may be present in multiple caches, allowing concurrent reads by different cores. Invalid means the line in that cache is no longer valid for use. Coherence ensures all cores see the same value by controlling transitions when lines are read or written. A read miss brings the line into a valid state, often as Exclusive or Shared depending on whether other caches already hold it. A write typically requires upgrading to Modified and invalidating or updating copies in other caches. When a line is evicted, a dirty Modified copy is written back to memory. This answer is the best because it accurately names the four MESI states and succinctly explains how coherence maintains consistency across cores through the state transitions of cache lines. Other statements either misname the states or omit the coherence aspect entirely.

7. Which filesystem is known for copy-on-write with checksums, snapshots, and pooled storage?

A. ext4

B. NTFS

C. ZFS

D. FAT32

Copy-on-write with checksums, snapshots, and pooled storage are hallmark features of ZFS. With copy-on-write, updates don't overwrite existing blocks; instead, new blocks are written and only after they're safely stored is the old data considered updated, which helps guarantee data integrity even if a crash occurs during writes. Every block and metadata in ZFS carries a strong checksum, so the system can detect corruption and repair it using redundant copies within the storage pool. Snapshots are lightweight point-in-time captures that share unchanged data blocks, enabling easy rollback and cloning without duplicating data. Pooled storage means disks are combined into a single storage pool that ZFS manages, allowing flexible, efficient allocation, dynamic expansion, and built-in redundancy across the pool. These combined capabilities are what set ZFS apart from other filesystems.

8. In the boot sequence, which component loads the kernel into memory after the bootloader has prepared the environment?

A. BIOS/UEFI firmware loads the kernel.

B. The bootloader loads the kernel into memory.

C. The kernel loads the bootloader.

D. POST transfers control to the kernel.

The step of loading the kernel into memory is performed by the bootloader. After the firmware (BIOS/UEFI) initializes hardware and hands control to a bootloader, that bootloader takes responsibility for preparing a minimal runtime environment and locating the kernel image on disk. It then loads the kernel into RAM (often along with an initial RAM disk) and finally transfers execution to the kernel's entry point. The kernel itself does not load the bootloader, and POST is completed before the boot process reaches this stage. This separation—firmware starting the process, bootloader loading the kernel, then the kernel taking over—is what makes the boot sequence work correctly.

9. Which statement describes the CPU's purpose?

A. Memory management unit handles paging

B. Graphics Processing Unit handles rendering

C. Central Processing Unit is responsible for executing a sequence which involves taking inputs from an input device, processing the input and outputting the results.

D. Network interface card handles data transfer

The CPU’s job is to execute the sequence of instructions that drive the computer, coordinating how data moves and is transformed. It fetches instructions from memory, decodes what each instruction requires, and executes the operation—performing arithmetic or logic, moving data, and controlling other parts of the system. In practice, this means it takes input from devices (like a keyboard), processes that data according to the program, and outputs results to memory or an output device (like a display). That ability to run software and manage data flow between memory and I/O is what the CPU is all about. The other components handle different roles: memory management units deal with paging and virtual memory, GPUs render graphics, and network interface cards manage data transfer over networks.

10. Which statement best describes the distinction between kernel space and user space?

A. Kernel space has privileged access to hardware and all memory; user space runs with restricted privileges; Separation protects the OS from errant or malicious user code and enforces security boundaries; context switches via system calls.

B. User space has privileged access to hardware and all memory; kernel space runs with restricted privileges; Separation is mainly for organization.

C. Both spaces have unrestricted privileges; separation is only for debugging.

D. Kernel space executes in user mode while user space executes in supervisor mode.

The idea being tested is the privilege and isolation boundary between kernel space and user space. The kernel runs with full, privileged access to hardware and to the entire system memory. It operates in a protected mode that allows it to manage resources, handle interrupts, and execute trusted code. User-space processes, on the other hand, run with restricted privileges and are isolated from one another and from the kernel’s memory. This separation prevents a misbehaving or malicious program from directly touching hardware or the kernel’s data structures, which could crash the system or compromise security. When user code needs services from the kernel, the CPU performs a controlled transition: a context switch into kernel mode where the kernel handles the request (for example, performing I/O, memory management, or scheduling) and then switches back to user mode to resume the program. This mechanism enforces well-defined interfaces and security boundaries, since all access to hardware and critical resources must go through the kernel. Other statements either reverse the privileges (giving hardware access to user space) or claim both spaces have the same privileges, or incorrectly describe the mode in which each space runs. The described arrangement—kernel in privileged mode with direct hardware access and full memory control, user space in restricted mode with isolation and protected interfaces—best captures the distinction.

Next Steps

Congratulations on reaching the final section of this guide. You've taken a meaningful step toward passing your certification exam and advancing your career.

As you continue preparing, remember that consistent practice, review, and self-reflection are key to success. Make time to revisit difficult topics, simulate exam conditions, and track your progress along the way.

If you need help, have suggestions, or want to share feedback, we'd love to hear from you. Reach out to our team at hello@examzify.com.

Or visit your dedicated course page for more study tools and resources:

https://syssoftwarearchimemorystorage.examzify.com

We wish you the very best on your exam journey. You've got this!