Chat Mode

○ RAG Mode

Parameters

Max Length

1024

8 2048

Temperature

0.70

0.00 1.00

Actions

New Chat Clear History

Response Time (Non-RAG): 16.45 seconds

CPU Usage (Non-RAG): 20.90%

Memory Usage (Non-RAG): -0.10%

Local Chatbot

A chatbot powered by LlamaIndex and Ollama

Hello, I'm your assistant, how can I help you?

9. In the context of CPU cache design, which of the following cache mapping policies would result in the highest hit rate but potentially the most expensive implementation? a) Direct mapping b) Set associative mapping c) Fully associative mapping d) Sequential mapping

The answer is **(c) Fully associative mapping**. Here's why:

- **Fully Associative Mapping:** This policy allows any memory block to be placed in any cache line. This means the CPU can always find the most recently used data, leading to the highest possible hit rate. However, it requires complex hardware to check all cache lines for a match when a memory access occurs, making it the most expensive implementation.

Let's break down the other options:

- **Direct Mapping:** Each memory block has a single, fixed location in the cache. This is simple and cheap to implement but can lead to lower hit rates because if a block is needed that's not in its assigned spot, it's a miss.
- **Set Associative Mapping:** Memory blocks are divided into sets, and each set can hold multiple blocks. A memory access checks all blocks within the set. This offers a balance between performance and cost, generally better than direct mapping but not as good as fully associative.
- **Sequential Mapping:** This policy maps consecutive memory addresses to consecutive cache lines. It's simple but only effective for programs with predictable memory access patterns.

In summary: Fully associative mapping provides the best hit rate but comes at a significant cost in hardware complexity.

Ask a question: