**What Every Programmer Should Know About Memory**

**Ulrich Drepper**

**Red Hat, Inc.**

drepper@redhat.com

**November 21, 2007**

***Sobre:***

*Traduzi este artigo utilizando o ChatGPT, unicamente para fins de estudo.*

**Resumo**

À medida que os núcleos da CPU se tornam mais rápidos e numerosos, o fator limitante para a maioria dos programas agora, e será por algum tempo, é o acesso à memória. Os designers de hardware têm desenvolvido técnicas de manipulação e aceleração de memória cada vez mais sofisticadas, como caches de CPU, mas essas não podem funcionar de forma ideal sem alguma ajuda por parte do programador. Infelizmente, a estrutura e o custo de uso do subsistema de memória de um computador ou das caches em CPUs não são bem compreendidos pela maioria dos programadores. Este artigo explica a estrutura dos subsistemas de memória em uso em hardware de commodities moderno, ilustrando por que as caches de CPU foram desenvolvidas, como funcionam e o que os programas devem fazer para alcançar um desempenho ótimo ao utilizá-las.

**1 Introdução**

Nos primeiros dias, os computadores eram muito mais simples. Os vários componentes de um sistema, como a CPU, a memória, o armazenamento em massa e as interfaces de rede, foram desenvolvidos em conjunto e, como resultado, eram bastante equilibrados em seu desempenho. Por exemplo, a memória e as interfaces de rede não eram (muito) mais rápidas do que a CPU na disponibilização de dados.

Essa situação mudou uma vez que a estrutura básica dos computadores se estabilizou, e os desenvolvedores de hardware se concentraram na otimização de subsistemas individuais. De repente, o desempenho de alguns componentes do computador ficou significativamente atrasado e gargalos se desenvolveram. Isso foi especialmente verdadeiro para os subsistemas de armazenamento em massa e de memória, que, por razões de custo, melhoraram mais lentamente em relação a outros componentes.

A lentidão do armazenamento em massa foi em grande parte resolvida com técnicas de software: sistemas operacionais mantêm os dados mais frequentemente usados (e mais prováveis de serem usados) na memória principal, que pode ser acessada a uma taxa de ordens de grandeza mais rápida do que o disco rígido. O armazenamento em cache foi adicionado aos dispositivos de armazenamento em si, o que não requer alterações no sistema operacional para aumentar o desempenho¹. Para os fins deste artigo, não entraremos em mais detalhes sobre otimizações de software para o acesso ao armazenamento em massa.

Ao contrário dos subsistemas de armazenamento, remover a memória principal como um gargalo tem se mostrado muito mais difícil, e quase todas as soluções exigem alterações no hardware. Hoje, essas mudanças geralmente ocorrem principalmente nas seguintes formas:

* Design de hardware da RAM (velocidade e paralelismo).
* Designs de controladores de memória.
* Caches de CPU.
* Acesso direto à memória (DMA) para dispositivos.

Para a maior parte, este documento se concentrará em caches de CPU e em alguns efeitos do design do controlador de memória. No processo de explorar esses tópicos, também abordaremos o DMA e o incorporaremos à imagem geral. No entanto, começaremos com uma visão geral do design para hardware de commodities de hoje. Isso é um pré-requisito para entender os problemas e as limitações de usar eficientemente os subsistemas de memória. Também aprenderemos, em algum detalhe, sobre os diferentes tipos de RAM e ilustraremos por que essas diferenças ainda existem.

Este documento não é de forma alguma abrangente e definitivo. Ele está limitado ao hardware de commodities e ainda mais limitado a um subconjunto desse hardware. Além disso, muitos tópicos serão discutidos com detalhes apenas o suficiente para os objetivos deste artigo. Para tópicos desse tipo, recomenda-se que os leitores procurem documentação mais detalhada.

Quando se trata de detalhes e soluções específicas de sistemas operacionais, o texto faz referência exclusivamente ao Linux. Em nenhum momento conterá informações sobre outros sistemas operacionais. O autor não tem interesse em discutir as implicações para outros sistemas operacionais. Se o leitor acreditar que precisa usar um sistema operacional diferente, ele deve contatar seus fornecedores e exigir que eles escrevam documentos semelhantes a este. Um último comentário antes de começarmos. O texto contém várias ocorrências do termo "geralmente" e outros qualificadores semelhantes. A tecnologia discutida aqui existe em muitas variações no mundo real, e este artigo aborda apenas as versões mais comuns e convencionais. É raro que afirmações absolutas possam ser feitas sobre essa tecnologia, daí o uso dos qualificadores.

¹Mudanças são necessárias, no entanto, para garantir a integridade dos dados ao usar caches de dispositivos de armazenamento.

**Estrutura do Documento**

Este documento é principalmente destinado a desenvolvedores de software. Ele não entra em detalhes técnicos suficientes sobre o hardware para ser útil para leitores orientados para hardware. No entanto, antes de podermos entrar nas informações práticas para os desenvolvedores, é necessário estabelecer uma base sólida.

Com esse objetivo, a segunda seção descreve a memória de acesso aleatório (RAM) em detalhes técnicos. O conteúdo desta seção é bom saber, mas não é absolutamente essencial para entender as seções posteriores. Referências adequadas a essa seção são adicionadas nos locais onde o conteúdo é necessário, para que o leitor ansioso possa pular a maior parte desta seção inicialmente.

A terceira seção entra em muitos detalhes sobre o comportamento do cache da CPU. Gráficos são usados para tornar o texto menos árido do que seria de outra forma. Esse conteúdo é essencial para a compreensão do restante do documento.

A Seção 4 descreve brevemente como a memória virtual é implementada. Isso também é um trabalho preliminar necessário para o restante do documento.

A Seção 5 entra em muitos detalhes sobre sistemas de Acesso à Memória Não Uniforme (NUMA).

A Seção 6 é a seção central deste artigo. Ela reúne todas as informações das seções anteriores e fornece conselhos aos programadores sobre como escrever código que funcione bem nas várias situações. O leitor muito impaciente pode começar com esta seção e, se necessário, voltar às seções anteriores para atualizar o conhecimento sobre a tecnologia subjacente.

A Seção 7 introduz ferramentas que podem ajudar o programador a fazer um trabalho melhor. Mesmo com um entendimento completo da tecnologia, não é óbvio onde estão os problemas em um projeto de software não trivial. Algumas ferramentas são necessárias.

Na Seção 8, finalmente, apresentamos uma visão da tecnologia que pode ser esperada no futuro próximo ou que simplesmente pode ser boa de se ter.

**Relatando Problemas**

O autor pretende atualizar este documento por algum tempo. Isso inclui atualizações necessárias devido aos avanços na tecnologia, bem como correções de erros. Leitores dispostos a relatar problemas são incentivados a enviar um e-mail para o autor. Eles são solicitados a incluir informações exatas sobre a versão no relatório. As informações de versão podem ser encontradas na última página do documento.

**Agradeço**

Gostaria de agradecer a Johnray Fuller e à equipe da LWN (especialmente a Jonathan Corbet por assumir a tarefa desafiadora de transformar a forma de inglês do autor em algo mais tradicional. Markus Armbruster forneceu muitos insights valiosos sobre problemas e omissões no texto.

**Sobre este Documento**

O título deste artigo é uma homenagem ao clássico artigo de David Goldberg "What Every Computer Scientist Should Know About Floating-Point Arithmetic" [12]. Este artigo ainda não é amplamente conhecido, embora deva ser um pré-requisito para qualquer pessoa que ouse tocar em um teclado para programação séria.

Uma observação sobre o PDF: o xpdf desenha alguns dos diagramas de forma um tanto deficiente. É recomendável visualizá-lo com o evince ou, se realmente necessário, com programas da Adobe. Se você usar o evince, esteja ciente de que hiperlinks são usados extensivamente em todo o documento, embora o visualizador não os indique como outros fazem