República Federativa do Brasil
Ministério da Economia
Instituto Nacional da Propriedade Industrial

(21) BR 112020013505-4 A2

(22) Data do Depósito: 13/02/2019

(43) Data da Publicação Nacional: 01/12/2020

(54) Título: GRAVAÇÃO DE RASTREIO REGISTRANDO INFLUXOS PARA UM CACHE DE CAMADA INFERIOR COM BASE EM ENTRADAS EM UM CACHE DE CAMADA SUPERIOR

(51) Int. Cl.: G06F 11/34; G06F 11/36; G06F 11/30.

(30) Prioridade Unionista: 23/02/2018 US 15/904,072.

(71) Depositante(es): MICROSOFT TECHNOLOGY LICENSING, LLC.

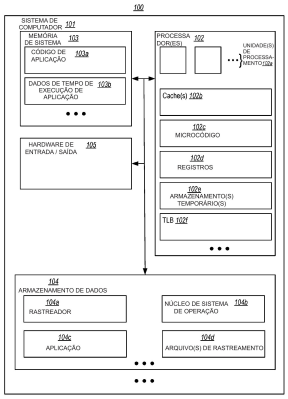
(72) Inventor(es): JORDI MOLA.

(86) Pedido PCT: PCT US2019017737 de 13/02/2019

(87) Publicação PCT: WO 2019/164710 de 29/08/2019

(85) Data da Fase Nacional: 01/07/2020

(57) Resumo: A presente invenção refere-se à gravação de rastreamento com base em gravar um influxo para um cache de nível inferior por referência a dados de registro anteriores, com base no conhecimento de um cache de nível superior. Um dispositivo de computação inclui uma pluralidade de unidades de processamento, uma pluralidade de caches de nível N e um cache de nível (N+i) que é um armazenamento de suporte para os caches de nível N. Com base na atividade de uma primeira unidade de processamento, o dispositivo de computação detecta um influxo de dados para um primeiro cache de nível N. O dispositivo de computação verifica o cache de nível (N+i) para determinar se os dados já foram registrados para uma segunda unidade de processamento. Com base na verificação, o dispositivo de computação (i) faz com que os dados sejam registrados para a primeira unidade de processamento por referência a dados de registro (isto é, quando os dados já foram registrados), ou faz com que os dados sejam registrados por valor para a primeira unidade de processamento (isto é, quando os dados ainda não foram registrados).



Relatório Descritivo da Patente de Invenção para **“GRAVAÇÃO DE RASTREIO REGISTRANDO INFLUXOS PARA UM CACHE DE CAMADA INFERIOR COM BASE EM ENTRADAS EM UM CACHE DE CAMADA SUPERIOR”.**

ANTECEDENTES

[0001] Quando escrevendo um código durante o desenvolvimento de aplicações de software, os desenvolvedores comumente dispendem uma significativa quantidade de tempo "depurando" o código para encontrar erros de tempo de execução e outros erros de códigos de fonte. Fazendo isto, os desenvolvedores podem adotar várias propostas para reproduzir e localizar um bug de código de fonte, tal como observando o comportamento de um programa com base em diferentes entradas, inserindo um código de depuração (por exemplo, para imprimir valores variáveis, para rastrear ramificações de execução; etc.), temporariamente removendo porções de código, etc. O rastreamento de erros de tempo de execução para identificar bugs de código pode ocupar uma porção significativa de tempo de desenvolvimento de aplicação.

[0002] Muitos tipos de aplicações de depuração ("depuradores") foram desenvolvidos de modo a ajudar os desenvolvedores com o processo de depuração de código. Estas ferramentas oferecem aos desenvolvedores a capacidade de rastrear, visualizar, e alterar a execução de código de computador. Por exemplo, os depuradores podem visualizar a execução de instruções de código, podem apresentar valores variáveis de código em vários tempos durante a execução de código, e podem permitir que os desenvolvedores alterem os percursos de execução de código, e/ou podem permitir que os desenvolvedores determinem "breakpoints" e/ou "watchpoints" em elementos de código de interesse (os quais, quando alcançados durante a execução, fazem com que a execução do código seja suspensa), entre outras coisas.

[0003] Uma forma emergente de aplicações de depuração permite

uma depuração "viagem no tempo", "reversa" ou "histórica". Com a depuração "viagem no tempo", a execução de um programa (por exemplo, entidades executáveis tais como encadeamentos) é gravada / rastreada por uma aplicação de rastreamento em um ou mais arquivos de rastreamento. Este(s) arquivo(s) de rastreamento podem então ser utilizados para reproduzir a execução do programa posteriormente, para análise tanto direta quanto reversa. Por exemplo, os depuradores de "viagem no tempo" podem permitir que um desenvolvedor determine breakpoints/watchpoints diretos (como depuradores convencionais), assim como breakpoints/watchpoints reversos.

[0004] Diversas considerações podem ser levadas em conta quando gravando arquivos de rastreamento. Mais proeminentemente, existe uma troca inerente entre a robustez dos dados de rastreamento gravados e os excessos incorridas pelo rastreamento de um programa. Estas trocas são manifestadas primariamente em tamanho de arquivo de rastreamento e impactos de desempenho sobre a execução do programa rastreado. Mais ainda, como o rastreamento pode ser executado com assistência de hardware (ou inteiramente em software), pode também existir um projeto de hardware e outras considerações de custo de hardware.

BREVE SUMÁRIO

[0005] As modalidades aqui descritas estão direcionadas para mecanismos para criar gravações de rastreamento de "viagem no tempo" precisas em bit utilizando assistência de hardware por um processador. Estes mecanismos estão baseados em rastrear os efeitos de execução através de uma pluralidade de unidades de processamento utilizando pelo menos duas filas ou camadas de caches de processador. Especificamente, estes mecanismos poderiam modificar um hardware e/ou microcódigo do processador de modo que este auxilie em (i) detectar um influxo (isto é, falta de cache) para um cache de processador interno ou

de "camada inferior" com base em atividade por uma unidade de processamento rastreada, e (ii) utilizar um cache de processador compartilhado externo ou de "camada superior" para determinar se os dados daquele influxo já foram registrados em nome de outra unidade de processamento rastreada. Se estes dados já foram registrados, então o influxo pode ser registrado por referência à entrada de registro anterior. Estas técnicas podem ser estendidas para "N" caches de nível. Gravar arquivos de rastreamento neste modo pode precisar somente de modestas modificações de processador e, quando comparado com propostas de gravação de rastreamento anteriores, pode reduzir por diversas ordens de magnitude tanto o impacto de desempenho de gravação de rastreamento assim como tamanho de arquivo de rastreamento.

[0006] As modalidades estão direcionadas a dispositivo(s) de computação que incluem uma pluralidade de unidades de processamento, uma pluralidade de caches de nível N, e um cache de nível (N+i). O cache de nível (N+i) está associado com dois ou mais da pluralidade de caches de nível N e está configurado como um armazenamento de apoio para a pluralidade de caches de nível N. Nestas modalidades, o(s) dispositivo(s) de computação incluem uma lógica de controle que configura o(s) dispositivo(s) de computação para detectar um influxo para um primeiro cache de nível N da pluralidade de caches de nível N e no qual o influxo compreende dados armazenados em uma localização de memória. A lógica de controle também configura o(s) dispositivo(s) de computação para verificar o cache de nível (N+i) para determinar se os dados para a localização de memória foram anteriormente registrados em nome de uma segunda unidade de processamento. A lógica de controle também configura o(s) dispositivo(s) de computação, com base nesta verificação, para executar um de (i) fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade

de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento (isto é, quando os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento), ou (ii) fazer com que os dados para a localização de memória sejam registrados por um valor em nome da primeira unidade de processamento (isto é, quando os dados para a localização de memória não foram anteriormente registrados em nome da segunda unidade de processamento).

[0007] As modalidades estão também direcionadas a métodos para gravação de rastreamento com base na gravação de um influxo para um cache de nível inferior por referência a dados de registro anteriores com base no conhecimento de um ou mais caches de nível superior. Estes métodos são implementados em um dispositivo de computação que inclui (i) uma pluralidade de unidades de processamento, (ii) uma pluralidade de caches de nível N, e (iii) um cache de nível (N+i) que está associado com dois ou mais da pluralidade de caches de nível N, e que está configurado como um armazenamento de apoio para a pluralidade de caches de nível N. O método inclui detectar um influxo para um primeiro cache de nível N da pluralidade de caches de nível N, o influxo compreendendo dados armazenados em uma localização de memória. O método também inclui com base na detecção do influxo para o primeiro cache de nível N, verificar o cache de nível (N+i) para determinar se os dados para a localização de memória foram anteriormente registrados em nome de uma segunda unidade de processamento. O método também inclui, com base nesta verificação, executar um de (i) quando os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro

que foram anteriormente registrados em nome da segunda unidade de processamento, ou (ii) quando os dados para a localização de memória não foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados por um valor em nome da primeira unidade de processamento.

[0008] As modalidades podem também ser incorporadas como instruções executáveis por computador (por exemplo, microcódigo de processador) armazenadas em um dispositivo de armazenamento de hardware, e que são executáveis para executar o método acima.

[0009] Este sumário está provido para introduzir uma seleção de conceitos em uma forma simplificada que são adicionalmente abaixo descritos na Descrição Detalhada. Este Sumário não pretende identificar características chave ou características essenciais do assunto reivindicado, nem este pretende ser utilizado como um auxílio na determinação do escopo do assunto reivindicado.

BREVE DESCRIÇÃO DOS DESENHOS

[0010] De modo a descrever o modo no qual as acima recitadas e outras vantagens e características da invenção podem ser obtidas, uma descrição mais específica da invenção brevemente acima descrita será prestada por referência a suas modalidades específicas as quais estão ilustradas nos desenhos anexos. Compreendendo que estes desenhos apresentam somente modalidades típicas da invenção e, portanto, não devem ser considerados limitativos de seu escopo, a invenção será descrita e explicada com especificidade e detalhes adicionais através da utilização dos desenhos acompanhantes:

[0011] Figura 1 ilustra um ambiente de computação exemplar que facilita a gravação de "rastreamentos de "precisão de bit" de execução através de uma pluralidade de unidades de processamento, utilizando pelo menos duas filas ou camadas de caches de processador - incluindo

detectar um influxo para um cache de processador interno ou de "camada inferior", e utilizar um cache de processador compartilhado externo ou de "camada superior" para determinar se este influxo pode ser registrado por referência a um valor anteriormente registrado;

[0012] Figura 2A ilustra um ambiente de computação exemplar que inclui caches de múltiplas camadas

[0013] Figura 2B ilustra um exemplo de um cache;

[0014] Figura 3 ilustra um fluxograma de um método exemplar para gravação de rastreamento com base na gravação de um influxo para um cache de nível inferior por referência a dados de registro anteriores com base no conhecimento de um ou mais caches de nível superior;

[0015] Figura 4A ilustra um cache compartilhado exemplar que estende cada uma de suas linhas de cache com um ou mais bits contábeis adicionais;

[0016] Figura 4B ilustra um exemplo de um cache compartilhado que inclui uma ou mais linhas de cache reservadas para armazenar bits contábeis que se aplicam a linhas de cache convencionais; e

[0017] Figura 5 ilustra um exemplo de mapeamento associativo de conjunto entre a memória de sistema e um cache.

DESCRIÇÃO DETALHADA

[0018] As modalidades aqui descritas estão direcionadas para mecanismos para criar gravações de rastreamento de "viagem no tempo" precisas em bit utilizando assistência de hardware por um processador. Estes mecanismos estão baseados em rastrear os efeitos de execução através de uma pluralidade de unidades de processamento utilizando pelo menos duas filas ou camadas de caches de processador. Especificamente, estes mecanismos poderiam modificar um hardware e/ou microcódigo do processador de modo que este auxilie em (i) detectar um influxo (isto é, falta de cache) para um cache de processador interno ou

de "camada inferior" com base em atividade por uma unidade de processamento rastreada, e (ii) utilizar um cache de processador compartilhado externo ou de "camada superior" para determinar se os dados daquele influxo já foram registrados em nome de outra unidade de processamento rastreada. Se estes dados já foram registrados, então o influxo pode ser registrado por referência à entrada de registro anterior. Estas técnicas podem ser estendidas para "N" caches de nível. Gravar arquivos de rastreamento neste modo pode precisar somente de modestas modificações de processador e, quando comparado com propostas de gravação de rastreamento anteriores, pode reduzir por diversas ordens de magnitude tanto o impacto de desempenho de gravação de rastreamento assim como tamanho de arquivo de rastreamento.

[0019] A Figura 1 ilustra um ambiente de computação exemplar 100 que facilita a gravação de rastreamentos de "precisão de bit" de execução através de uma pluralidade de unidades de processamento, utilizando pelo menos duas filas ou camadas de caches de processador - que inclui detectar um influxo para um cache de processador interno ou de "camada inferior", e utilizar um cache de processador compartilhado externo ou de "camada superior" para determinar se este influxo pode ser registrado por referência a um valor anteriormente registrado. Como apresentado, as modalidades podem compreender ou utilizar um sistema de computador de uso especial ou uso geral 101 que inclui um hardware de computador, tal como, por exemplo, um ou mais processador(es) 102, memória de sistema 103, um ou mais armazenamentos de dados 104, e/ou hardware de entrada / saída 105.

[0020] As modalidades dentro do escopo da presente invenção incluem meios físicos e outros legíveis por computador para carregar ou armazenar instruções executáveis por computador e/ou estruturas de dados. Tais meios legíveis por computador podem ser qualquer meio disponível que possa ser acessado pelo sistema de computador 101.

Os meios legíveis por computador que armazenam instruções executáveis por computador e/ou estruturas de dados são dispositivos de armazenamento de computador. Os meios legíveis por computador que carregam instruções executáveis por computador e/ou estruturas de dados são meios de transmissão. Assim, por meio de exemplo, e não limitação, as modalidades da invenção podem compreender pelo menos dois tipos distintamente diferentes de meios legíveis por computador: dispositivos de armazenamento de computador e meios de transmissão.

[0021] Os dispositivos de armazenamento de computador são dispositivos de hardware físico que armazenam instruções executáveis por computador e/ou estruturas de dados. Os dispositivos de armazenamento de computador incluem vários hardwares de computador, tal como RAM, ROM, EEPROM, unidades de estado sólido ("SSDs"), memória instantânea, memória de mudança de fase ("PCM"), armazenamento de disco ótico, armazenamento de disco magnético ou outros dispositivos de armazenamento magnético, ou qualquer outro dispositivo de hardware o qual possa ser utilizado para armazenar um código do programa na forma de instruções executáveis por computador ou estruturas de dados, e as quais podem ser acessadas e executadas pelo sistema de computador 101 para implementar a funcionalidade descrita da invenção. Assim, por exemplo, os dispositivos de armazenamento de computador podem incluir a memória de sistema apresentada 103, o armazenamento de dados apresentado 104 o qual pode armazenar instruções executáveis por computador e/ou estruturas de dados ou outro armazenamento tal como armazenamento em processador, como posteriormente discutido.

[0022] Os meios de transmissão podem incluir uma rede e/ou conexões de dados as quais podem ser utilizadas para carregar um código de programa na forma de instruções executáveis por computador ou estruturas de dados e as quais podem ser acessadas pelo sistema de

computador 101. Uma "rede" é definida como uma ou mais conexões de dados que permitem o transporte de dados eletrônicos entre sistemas de computador e/ou módulos e/ou outros dispositivos eletrônicos. Quando as informações são transferidas ou providas sobre uma rede ou outra conexão de comunicações (ou com fio, sem fio ou uma combinação de com fio ou sem fio) para um sistema de computador, o sistema de computador pode visualizar a conexão como meios de transmissão. Combinações dos acima devem também estar incluídas dentro do escopo de meios legíveis por computador. Por exemplo, o hardware de entrada / saída 105 pode compreender um hardware (por exemplo, um módulo de interface de rede (por exemplo, um "NIC")) que conecta uma rede e/ou conexão de dados a qual pode ser utilizada para carregar um código de programa na forma de instruções executáveis por computador ou estruturas de dados.

[0023] Ainda, quando alcançando vários componentes de sistema de computador, o código de programa na forma de instruções executáveis por computador ou estruturas de dados pode ser transferido automaticamente dos meios de transmissão para os dispositivos de armazenamento de computador (ou vice-versa). Por exemplo, instruções executáveis por computador ou estruturas de dados recebidas sobre uma rede ou conexão de dados podem ser armazenadas temporariamente em RAM dentro de um NIC (por exemplo, hardware de entrada / saída 105), e então eventualmente transferidas para a memória de sistema 103 e/ou para dispositivos de armazenamento de computador menos voláteis (por exemplo, armazenamento de dados 104) no sistema de computador 101. Assim, deve ser compreendido que os dispositivos de armazenamento de computador podem estar incluídos em componentes de sistema de computador que também (ou mesmo primariamente) utilizam os meios de transmissão.

[0024] As instruções executáveis por computador compreendem,

por exemplo, instruções e dados os quais, quando executados no(s) processador(es) 102, fazem com que o sistema de computador 101 execute uma certa função ou grupo de funções. As instruções executáveis por computador podem ser, por exemplo, binários, instruções de formato intermediário tal como linguagem assembly, ou mesmo código de fonte.

[0025] Aqueles versados na técnica apreciarão que a invenção pode ser praticada em ambientes de computação em rede com muitos tipos de configurações de sistema de computador, incluindo, computadores pessoais, computadores desktop, computadores laptop, processadores de mensagens, dispositivos portáteis, sistemas de múltiplos processadores, eletrônica de consumidor baseada em microprocessador ou programáveis, PCs de rede, minicomputadores, computadores mainframe, telefones móveis, PDAs, tablets, pagers, roteadores, comutadores, e similares. A invenção pode também ser praticada em ambientes de sistema distribuído onde sistemas de computador locais e remotos, os quais estão conectados (ou por conexão de dados com fio, conexão de dados sem fio, ou por uma combinação de conexões de dados com fio e sem fio) através de uma rede, ambos executam tarefas. Como tal, em um ambiente de sistema distribuído, um sistema de computador pode incluir uma pluralidade de sistemas de computador constituintes. Em um ambiente de sistema distribuído, módulos do programa podem estar localizados em dispositivos de armazenamento de memória tanto locais quanto remotos.

[0026] Aqueles versados na técnica também apreciarão que a invenção pode ser praticada em um ambiente de computação em nuvem. Os ambientes de computação em nuvem podem ser distribuídos, apesar disto não ser requerido. Quando distribuídos, os ambientes de computação em nuvem podem ser distribuídos internacionalmente dentro

de uma organização e/ou ter componentes possuídos através de múltiplas organizações. Nesta descrição e nas reivindicações seguintes, "computação em nuvem" é definida como um modelo para permitir acesso de rede sob demanda para um grupamento compartilhado de recursos de computação configuráveis (por exemplo, redes, servidores, armazenamentos, aplicações, e serviços). A definição de "computação em nuvem" não está limitada a qualquer das outras numerosas vantagens que podem ser obtidas de tal modelo quando apropriadamente implantado.

[0027] Um modelo de computação em nuvem pode ser composto de várias características, tal como autoatendimento sob demanda, amplo acesso à rede, grupamento de recursos, rápida elasticidade, serviço medido, e assim por diante. Um modelo de computação em nuvem pode também vir na forma de vários modelos de serviço tais como, por exemplo, Software como um Serviço ("SaaS"), Plataforma como um Serviço ("PaaS") e Infraestrutura como um Serviço ("laaS"). O modelo de computação em nuvem pode também ser implantado utilizando diferentes modelos de implantação tal como nuvem privada, nuvem de comunidade, nuvem pública, nuvem híbrida, e assim por diante.

[0028] Algumas modalidades, tal como um ambiente de computação em nuvem, podem compreender um sistema que inclui um ou mais hospedeiros que são cada um capazes de executar uma ou mais máquinas virtuais. Durante a operação, as máquinas virtuais emulam um sistema de computação operacional, suportando um sistema de operação e talvez uma ou mais outras aplicações também. Em algumas modalidades, cada hospedeiro inclui um hipervisor que emula recursos virtuais para as máquinas virtuais utilizando recursos físicos que são abstraídos da visualização das máquinas virtuais. O hipervisor também provê um isolamento apropriado entre as máquinas virtuais. Assim, da

perspectiva de qualquer dada máquina virtual, o hipervisor provê a ilusão que a máquina virtual está interfaceando com um recurso físico, mesmo que a máquina virtual somente interfaceie com a aparência (por exemplo, um recurso virtual) de um recurso físico. Exemplos de recursos físicos, incluem capacidade de processamento, memória, espaço em disco, largura de banda de rede, unidades de mídia, e assim por diante.

[0029] Como ilustrado, o armazenamento de dados 104 pode armazenar instruções executáveis por computador e/ou estruturas de dados que representam programas de aplicação tais como, por exemplo, um rastreador 104a, um núcleo de sistema de operação 104b, e aplicação 104c (por exemplo, a aplicação que é o assunto de rastreamento pelo rastreador 104a, e um ou mais arquivo(s) de rastreamento 104d). Quando estes programas estão executando (por exemplo, utilizando o(s) processador(es) 102), a memória de sistema 103 pode armazenar dados de tempo de execução correspondentes, tais como estruturas de dados de tempo de execução, instruções executáveis por computador, etc. Assim, a Figura 1 ilustra a memória de sistema 103 como incluindo código de aplicação de tempo de execução 103a e dados de tempo de execução de aplicação 103b (por exemplo, cada um correspondendo com a aplicação 104c).

[0030] O rastreador 104a é utilizável para gravar um rastreamento de precisão de bit de execução de uma aplicação, tal como a aplicação 104c, e para armazenar dados de rastreamento no(s) arquivo(s) de rastreamento 104d. Em algumas modalidades, o rastreador 104a é uma aplicação independente, enquanto que em outras modalidades o rastreador 104a está integrado em outro componente de software, tal como o núcleo de sistema de operação 104b, um hipervisor, uma fábrica de nuvem, etc. Apesar do(s) arquivo(s) de rastreamento 104d serem apresentados como estando armazenados no armazenamento de dados 104,

o(s) arquivo(s) de rastreamento 104d podem também ser gravados exclusivamente ou temporariamente na memória de sistema 103, ou em algum outro dispositivo de armazenamento. Como esclarecido posteriormente, o rastreador 104a pode interoperar com características específicas do(s) processador(es) 102 que permitem o rastreamento utilizando o protocolo de coerência de cache (CCP) do processador.

[0031] A Figura 1 inclui uma representação simplificada dos componentes de hardware internos do(s) processador(es) 102. Como ilustrado, cada processador 102 inclui uma pluralidade de unidade(s) de processamento 102a. Cada unidade de processamento pode ser física (isto é, um núcleo de processador físico) e/ou lógica (isto é, um núcleo lógico representado por um núcleo físico que suporta hiperencadeamento, no qual mais do que um encadeamento de aplicação executa no núcleo físico). Assim ,por exemplo, apesar de que o processador 102 pode em algumas modalidades incluir somente uma única unidade de processamento física (núcleo), este poderia incluir duas ou mais unidades de processamento lógicas 102a representadas por aquela única unidade de processamento física.

[0032] Cada unidade de processamento 102a executa instruções de processador que são definidas por aplicações (por exemplo, rastreador 104a, núcleo de operação 104b, aplicação 104c, etc.), e cujas instruções são selecionadas dentre uma arquitetura de conjunto de instruções do processador predefinida (ISA). A ISA específica de cada processador 102 varia com base no fabricante de processador e modelo de processador. As ISAs comuns incluem as arquiteturas IA-64 e IA-32 da INTEL, INC., a arquitetura AMD64 da ADVANCED MICRO DEVICES, INC. e várias arquiteturas de Máquina RISC Avançada ("ARM") da ARM HOLDINGS, PLC, apesar de que um grande número de outras ISAs existe e pode ser utilizado pela presente invenção. Em geral, uma "instrução" é

a menor unidade de código externamente visível (isto é, externa ao processador) que é executável por um processador.

[0033] Cada unidade de processamento 102a obtém instruções de processador de um ou mais cache(s) de processador 102b, e executa as instruções de processador com base em dados no(s) cache(s) 102b, com base em dados em registros 102d, e/ou sem dados de entrada. Em geral, cada cache 102b é uma pequena quantidade (isto é, pequena em relação à quantidade típica de memória de sistema 103) de memória de acesso randômico que armazena copias no processador de porções de um armazenamento de apoio, tal como a memória de sistema 103 e/ou outro cache no(s) cache(s) 102b. Por exemplo, quando executando o código de aplicação 103a, uma ou mais do()s) cache(s) 102b contêm porções dos dados de tempo de execução de aplicação 103b. Se a(s) unidade(s) de processamento 102a solicitarem dados ainda não armazenados em um cache específico 102b, então "falta de cache" ocorre, e estes dados são buscados da memória do sistema 103 ou outro cache, potencialmente "removendo" alguns outros dados daquele cache 102b.

[0034] Frequentemente, o(s) cache(s) de processador 102b estão dividido em filas, camadas, ou níveis separados - tal como camada 1 (L1), camada 2 (L2), camada 3 (L3), etc. Dependendo da implementação de processador, as filas poderiam fazer parte do próprio processador 102 (por exemplo, L1 e L2), e/ou poderiam ser separadas do processador 102 (por exemplo, L3). Assim, o(s) cache(s) 102b da Figura 1 podem compreender uma destas camadas (L1) ou podem compreender uma pluralidade destas camadas (por exemplo, L1 e L2 e mesmo L3). A Figura 2A ilustra um ambiente exemplar 200 que demonstra caches de múltiplas camadas. Na Figura 2A, existem dois processadores 201a e 201b (por exemplo, cada um correspondendo a um diferente processador 102 da Figura 1) e uma memória de sistema 202 (por exemplo, correspondendo à memória de sistema 103 da Figura 1). No ambiente

exemplar 200, cada processador 201 inclui quatro unidades de processamento físicas (isto é, unidades A1-A4 para o processador 201a e unidades B1-B4 para o processador 210b).

[0035] O ambiente exemplar 200 também inclui um cache de três camadas dentro de cada unidade de processamento 201. O ambiente 200 é um layout exemplar somente, e não limita as hierarquias de cache na qual as modalidades aqui podem operar. No ambiente 200, em uma camada mais baixa ou mais interna cada unidade de processamento está associada com o seu próprio cache L1 dedicado (por exemplo, cache L1 "L1-A1" no processador 201a para a unidade A1, cache L1 "L1-A2" no processador 201a para a unidade A2, etc.). Movendo uma camada cima, cada unidade de processamento 201 inclui dois caches L2 (por exemplo, cache L2 "L2-A1" no processador 201a que serve como um armazenamento de suporte para L1 caches L1-A1, e L1-A2, cache L2 "L1-A2" no processador 201a que serve como um armazenamento de suporte para L1 caches L1-A3 e L1-A4, etc.). Finalmente, na camada mais alta ou mais externa, cada unidade de processamento 201 inclui um único cache L3 (por exemplo, cache L3 "L3-A" no processador 201a que serve como um armazenamento de suporte para L2 caches L2-A1 e L2-A2, e cache L3 "L3-B" no processador 201b que serve como um armazenamento de suporte para L2 cache L2-B1 e L2-B2). Como mostrado, a memória de sistema 202 serve como um armazenamento de suporte para os L3 caches L3-A e L3-B.

[0036] Como demonstrado na Figura 2A, quando múltiplas camadas de cache são utilizadas, a(s) unidade(s) de processamento 102a tipicamente interagem diretamente com a camada mais baixa (L1). Na maioria dos casos, os fluxos de dados entre as camadas (por exemplo, em uma leitura um cache L3 interage com a memória de sistema 103 e serve dados para um cache L2, e o cache L2, por sua vez serve dados

para o cache L1). Quando uma unidade de processamento 102a executa uma escrita, os cache coordenam para assegurar que aqueles caches que afetaram dados que foram compartilhados entre a(s) unidade(s) de processamento 102a não os tenham mais. Esta coordenação é executada utilizando um CCP.

[0037] Os caches no ambiente 200 podem assim ser vistos como caches "compartilhados". Por exemplo, cada cache L2 e L3 serve múltiplas unidades de processamento dentro de um dado processador 201 e são assim compartilhados pelas unidades de processamento. Os caches L1 com um dentro de um dado processador 201, coletivamente, podem também ser considerados compartilhados - mesmo que cada um corresponda a uma única unidade de processamento - porque os caches L1 individuais podem coordenar uns com os outros (isto é, através de um CCP) para assegurar consistência (isto é, de modo que cada localização de memória em cache é vista consistentemente através de todos os caches L1). Os caches L2 dentro de cada processador 201 similarmente podem coordenar através de um CCP. Além disso, se o processador 201 suportar hiperencadeamento, cada cache L1 individual pode ser visto sendo compartilhado por duas ou mais unidades de processamento lógicas e são assim "compartilhados" mesmo em um nível individual.

[0038] Tipicamente, cada cache compreende uma pluralidade de "linhas de cache". Cada linha de cache armazena um pedaço de memória de seu armazenamento de suporte (por exemplo, memória de sistema 202 ou um cache de camada mais alta). Por exemplo, a Figura 2B ilustra um exemplo de pelo menos uma porção de um cache 203, a qual inclui uma pluralidade de linhas de cache 206, cada uma das quais compreende pelo menos uma porção de endereço 204 e uma porção de valor 205. A porção de endereço 204 de cada linha de cache 206 está configurada para armazenar um endereço na memória de sistema 202 para

a qual a linha de cache corresponde, e a porção de valor 205 armazena inicialmente um valor recebido da memória do sistema 202. A porção de valor 205 pode ser modificada pelas unidades de processamento, e eventualmente ser removidas de volta para o armazenamento de suporte. Como indicado pelas elipses, um cache 203 pode incluir um grande número de linhas de cache. Por exemplo, um processador INTEL de 64 bits contemporâneo pode conter caches L1 individuais que compreendem 512 ou mais linhas de cache. Em tal cache, cada linha de cache é tipicamente utilizável para armazenar um valor de 64 bytes (512 bits) em referência a um endereço de memória de 6 bytes (48 bits) até 8 bytes (64 bits). Como visualmente indicado na Figura 2A, os tamanhos de cache tipicamente aumentam com cada camada (isto é, caches L2 são tipicamente maiores do que caches L1, caches L3 são tipicamente maiores do que caches L2, etc.).

[0039] O endereço armazenado na porção de endereço 204 de cada linha de cache 206 pode ser um endereço físico, tal como o endereço de memória real na memória de sistema 202. Alternativamente, o endereço armazenado na porção de endereço 204 pode ser um endereço virtual, o qual é um endereço que é mapeado para o endereço físico para prover uma abstração (por exemplo, utilizando tabelas de páginas gerenciadas pelo sistema de operação). Tais abstrações podem ser utilizadas, por exemplo, para facilitar o isolamento de memória entre diferentes processos que executam no(s) processador(es) 102, incluindo isolamento entre processos de modo de usuário e processos de modo de núcleo associados com o núcleo de sistema de operação 104b. Quando endereços virtuais são utilizados, um processador 102 pode incluir um armazenamento temporário associativo de tradução (TLB) 102f (usualmente parte de uma unidade de gerenciamento de memória (MMU)), o qual mantém mapeamentos de endereços de memória recentemente utilizados entre endereços físicos e virtuais.

[0040] O(s) cache (s) 102b podem incluir porções de cache de código e porções de cache de dados. Quando executando o código de aplicação 103a, a(s) porção(ões) de código do(s) cache(s) 102b podem armazenar pelo menos uma porção das instruções de processador armazenadas no código de aplicação 103a e a(s) porção(ões) de dados do(s) cache(s) 102b pode armazenar pelo menos uma porção de estruturas de dados dos dados de tempo de execução de aplicação 103b. Além disso, os caches podem ser inclusivos, exclusivos, ou incluir comportamentos tanto inclusivos quanto exclusivos. Por exemplo, em um cache inclusivo, uma camada L3 armazenaria um superconjunto dos dados nas camadas L2 abaixo desta, e as camadas L2 armazenam um superconjunto das camadas L1 abaixo destas. Em caches exclusivos, as camadas podem ser separadas - por exemplo, se dados existirem em um cache L3 que um cache L1 precisa, estes podem trocar informações, tais como dados, endereço, e similares.

[0041] Retornando à Figura 1, cada processador 102 também inclui um microcódigo 102c, o qual compreende uma lógica de controle (isto é, instruções executáveis) que controla a operação do processador 102, e o qual geralmente funciona como um intérprete entre o hardware do processador e a ISA de processador exposta pelo processador 102 para executar aplicações. O microcódigo 102 está tipicamente incorporado em armazenamento no processador, tal como ROM, EEPROM, etc.

[0042] Os registros 102d são localizações de armazenamento baseadas em hardware que são definidas com base na ISA do(s) processador(es) 102 e que são lidos de e/ou escritos no pelas instruções de processador. Por exemplo, os registros 102d são comumente utilizados para armazenar valores buscados do(s) cache(s) 102b para utilização por instruções, para armazenar os resultados de instruções de execução, e/ou para armazenar status ou estado - tais como alguns dos efeitos colaterais de instruções de execução (por exemplo, o sinal de um

valor mudando, um valor atingindo zero, a ocorrência de um carregamento, etc.), uma contagem de ciclos de processador, etc. Assim, alguns registros 102d podem compreender "sinalizadores" que são utilizados para sinalizar alguma mudança de estado causada pela execução de instruções de processador. Em algumas modalidades, os processadores 102 podem também incluir registros de controle, os quais são utilizados para controlar diferentes aspectos de operação de processador. Apesar da Figura 1 apresentar registros 102d como uma caixa única, será apreciado que cada unidade de processamento 102a tipicamente inclui um ou mais conjuntos correspondentes de registros 102d que são específicos para aquela unidade de processamento.

[0043] Em algumas modalidades, o(s) processador(es) 102 podem incluir um ou mais armazenamentos temporários 102e. Como será posteriormente aqui discutido, o(s) armazenamento(s) temporário(s) 102e podem ser utilizados como localização de armazenamento temporário para dados de rastreamento. Assim, por exemplo, o(s) processador(es) 102 podem armazenar porções de dados de rastreamento no(s) armazenamento(s) temporário(s) 102e, e liberar estes dados para o(s) arquivo(s) de rastreamento 104d em momentos apropriados, tal como quando existe largura de banda de barramento de memória disponível e/ou ciclos de processador livres.

[0044] Como acima aludido, os processadores operam sobre o(s) cache(s) 102b de acordo com um ou mais CCPs. Em geral, um CCP define como uma consistência é mantida entre dados entre o(s) vário(s) cache(s) 102b, e como assegurar que as várias unidades de processamento 102a sempre leiam dados válidos de uma dada localização no(s) cache(s) 102b. Os PCCs estão relacionados com e permitem um modelo de memória definido pela ISA do processador 102.

[0045] Exemplos de CCPs comuns incluem o protocolo de MSI (isto é, Modificado, Compartilhado, e Inválido), o protocolo de MESI (isto é,

Modificado, Exclusivo, Compartilhado, e Inválido) e o protocolo de MOESI (isto é, Modificado, Próprio, Exclusivo, Compartilhado, e Inválido). Cada um destes protocolos define um estado para localizações individuais (por exemplo, linhas) no(s) cache(s) 102b. Uma localização de cache "modificada" contém dados que foram modificados no(s) cache(s) 102b e é portanto potencialmente inconsistente com os dados correspondentes no armazenamento de suporte (por exemplo, memória de sistema 103 ou outro cache). Quando uma localização que tem o estado "modificado" é removida do(s) cache(s) 102b, CCPs comuns requerem que o cache garanta que seus dados sejam escritos de volta no armazenamento de suporte, ou que outro cache assuma esta responsabilidade. Uma localização de cache "compartilhada" contém dados que não são modificados dos dados no armazenamento de suporte, existe em um estado somente de leitura e é compartilhado pela(s) unidade(s) de processamento 102a. O(s) cache(s) 102b podem remover estes dados sem escreve-los no armazenamento de suporte. Uma localização de cache "inválida" não contém dados válidos e pode ser considerada vazia e utilizável para armazenar dados de falta de cache. Uma localização de cache "exclusiva" contém dados que coincidem com o armazenamento de suporte e são utilizados por somente uma única unidade de processamento 102a. Este pode ser mudado para o estado "compartilhado" em qualquer tempo (isto é, em resposta a uma solicitação de leitura) ou pode ser mudado para o estado "modificado" quando escrevendo neste. Uma localização de cache "de propriedade" é compartilhada por duas ou mais unidades de processamento 102a, mas uma das unidades de processamento tem o direito exclusivo de fazer mudanças neste. Quando este processamento faz mudanças, este notifica as outras unidades de processamento - já que as unidades de processamento notificadas podem precisar invalidar ou atualizar com base na implementação de CCP.

[0046] Como aludido, as modalidades utilizam o(s) cache(s) 102b do processador 102 para eficientemente gravar um rastreamento preciso em bits de execução de uma aplicação 104c e/ou o núcleo de sistema de operação 104b. Estas modalidades são construídas sobre uma observação pelo inventor que o processador 102 (incluindo o(s) cache(s) 102b) forma um sistema semi ou quase fechado. Por exemplo, uma vez que porções de dados para um processo (isto é, dados de código e dados de aplicação de tempo de execução) são carregadas no(s) cache(s) 102b, o processador 102 pode executar por si próprio - sem qualquer entrada - como um sistema semi ou quase fechado por rajadas de tempo. Especificamente, uma vez que o(s) cache(s) 102b estão carregados com dados, uma ou mais das unidades de processamento 102a executam instruções da(s) porção(ões) de código do(s) cache(s) 102b, utilizando dados de tempo de execução armazenados na(s) porção(ões) de dados do(s) cache(s) 102b e utilizando os registros 102d.

[0047] Quando uma unidade de processamento 102a precisa de algum influxo de informações (por exemplo, porque uma instrução que esta está executando, executará, ou pode executar acessa um código ou dados de tempo de execução ainda não no(s) cache(s) 102b), uma "falta de cache" ocorre e estas informações são trazidas no(s) cache(s) 102b da memória de sistema 103. Por exemplo, se uma falta de cache de dados ocorrer quando uma instrução executada executa uma operação de memória em um endereço de memória dentro dos dados de tempo de execução da aplicação 103b, os dados daquele endereço de memória são trazidos em uma das linhas de cache da porção de dados do(s) cache(s) 102b. Similarmente, se uma falta de cache de código ocorrer quando uma instrução executa uma operação de memória em um código de aplicação de endereço de memória 103a armazenado na memória de sistema 103, o código daquele endereço de memória é trazido em uma das linhas de cache da(s) porção(ões) de código do(s)

cache(s) 102b. A unidade de processamento 102a então continua a execução utilizando as novas informações no(s) cache(s) 102b até que novas informações sejam novamente trazidas para dentro do(s) cache(s) 102b (por exemplo, devido a outra falta de cache ou uma leitura não em cache).

[0048] O inventor também observou que, de modo a gravar uma representação precisa em bit de execução de uma aplicação, o rastreador 104a pode gravar dados suficientes para ser capaz de reproduzir o influxo de informações no(s) cache(s) 102b conforme as unidades de processamento executam aquele(s) encadeamento(s) da aplicação. Por exemplo, uma proposta para gravas estes influxos opera em uma base por unidade de processamento e na camada de cache mais interna (por exemplo, L1). Esta proposta pode envolver gravar, para cada unidade de processamento que está sendo rastreada, todas as faltas de cache e leituras não em cache (isto é, leituras de componentes de hardware e memória não armazenável em cache) associadas com o cache L1 daquela unidade de processamento, juntamente com um tempo durante a execução no qual cada porção de dados foi trazida para aquele cache L1 da unidade de processamento (por exemplo, utilizando uma contagem de instruções executadas ou algum outro contador). Se existirem eventos que podem ser ordenados através das unidades de processamento (por exemplo, acessos a memória compartilhada), estes eventos podem ser registrados através dos fluxos de dados resultantes (por exemplo, utilizando um número que incrementa monotonicamente (MIN) através dos fluxos de dados).

[0049] No entanto, como uma camada de cache L1 pode incluir múltiplos caches L1 distintos que estão cada um associados a uma diferente unidade de processamento física (por exemplo, como mostrado na Figura 2A), gravar deste modo pode gravar dados duplicados - e as-

sim mais dados que são estritamente necessários para um rastreamento de "fidelidade total". Por exemplo, se múltiplas unidades de processamento físicas lerem da mesma localização de memória (o que pode ser uma frequente ocorrência em aplicações múltiplos encadeamentos), esta proposta pode registrar faltas de cache para a mesma localização de memória e dados para cada uma de múltiplas unidades de processamento físicas. Notadamente, como aqui utilizado, um rastreamento de "fidelidade total" é qualquer rastreamento que contenha informações suficientes para permitir uma total reprodução de uma entidade rastreada - mesmo que um rastreamento de "fidelidade total" específico possa realmente conter menos dados que encapsularam as mesmas informações que poderiam ser gravadas utilizando técnicas de rastreamento alternativas.

[0050] De modo a adicionalmente reduzir o tamanho de arquivo de rastreamento, o inventor desenvolveu técnicas de gravação aperfeiçoadas que utilizam um ou mais dos caches de camada superior para evitar gravar pelo menos uma porção destes dados duplicados. Ao invés, estas técnicas aperfeiçoadas podem registrar por referência a dados anteriormente registrados. Especificamente, as modalidades detectam um influxo (isto é, falta de cache) para um cache de processador interno ou de "camada inferior" (por exemplo, L1) com base em atividade por uma unidade de processamento, mas então utilizam um ou mais cache(s) de processador compartilhados externos ou de "camada superior" para registrar que este influxo por referência a um influxo já registrado em nome de outra unidade de processamento rastreada, quando possível.

[0051] De modo a compreender estas técnicas, é notado que, na maioria dos ambientes, um cache da camada superior é maior que os caches de camada inferior abaixo deste, e é frequentemente um armazenamento de suporte para múltiplos caches de camada inferior. Por

exemplo, no ambiente exemplar da Figura 2A, cada cache L2 é um armazenamento de suporte para dois caches L1, e cada cache L3 é um armazenamento de suporte para dois caches L2 (e, por extensão, quatro caches L1). Assim, um cache de camada superior pode reter conhecimento sobre múltiplos caches de camada inferior (por exemplo, na Figura 2A, o cache L2 L1-A1 pode reter conhecimento sobre caches L1 L1-A1 e L1-A2, o cache L2 L1-A2 pode reter conhecimento sobre caches L1 L1-A3 e L1-A4, e o cache L3 L3-A pode reter conhecimento sobre caches L2 L2-A1 e L2-A2, assim como caches L1 L1-A1, L1-A2 L1-A3 e L1-A4). Utilizando o conhecimento de uma ou mais camadas de cache superiores, as modalidades aqui permitem muitas oportunidades para registrar influxos causados por uma unidade de processamento por referência a um influxo já registrado em nome de outras unidades de processamento.

[0052] De acordo com estas modalidades, a Figura 3 ilustra um exemplo de um método 300 para gravação de rastreamento com base em gravar um influxo para um cache de nível inferior por referência a dados de registro anteriores com base no conhecimento de um ou mais caches de nível superior. A Figura 3 está agora descrita no contexto das Figuras 1 e 2.

[0053] Especificamente, a Figura 3 opera em ambientes, tal como um processador 102 ou 201a que inclui uma pluralidade de unidades de processamento, uma pluralidade de caches de nível N, e um cache de nível (N+i) que está associado com dois ou mais da pluralidade de caches de nível N, e que está configurado como um armazenamento de suporte para a pluralidade de caches de nível N. No método 300 (e nas reivindicações), N e i são inteiros positivos - isto é, $N \geq 1$, de modo que N é igual a 1, 2, 3, etc.; e $i \geq 1$,de modo que i é igual a 1, 2, 3, etc. Por exemplo, referindo ao processador 201a da Figura 2A, o processador inclui uma pluralidade de unidades de processamento A1, A2, etc. O

processador 201a também inclui uma pluralidade de caches de nível N L1-A1, L1-A2, etc. (isto é, onde N é igual a 1). O processador 201a também inclui um cache de nível (N+i) que está associado com dois ou mais da pluralidade de caches de nível N, e que está configurado como um armazenamento de suporte para a pluralidade de caches de nível N. Por exemplo, o processador 201a inclui um cache de nível (N+i) L2-A1 que é um armazenamento de suporte para os caches de nível N L1-A1 e L1-A2 (isto é, onde N é igual a 1, e i é igual a 1). Em outro exemplo, o processador 201a inclui um cache de nível (N+i) L3-A que é um armazenamento de suporte para caches de nível N L1-A1, L1-A2, etc. (isto é, onde N é igual a 1, e i é igual a 2). O processador 102/201a opera o método 300 com base em lógica de controle, tal como microcódigo 102c e/ou lógica de circuito.

[0054] Como mostrado, o método 300 inclui um ato 301 de, durante a execução em uma primeira unidade de processamento, detectar um influxo para um cache de nível N. Em algumas modalidades, o ato 301 compreende detectar um influxo para um primeiro cache de nível N de uma pluralidade de caches de nível N, o influxo compreendendo dados armazenados em uma localização de memória. Por exemplo, com base na atividade pela unidade de processamento A1, tal como um acesso de memória solicitado para a memória de sistema 202 (por exemplo, resultando de execução normal ou especulativa de um primeiro encadeamento da aplicação 104c), uma falta de cache pode ocorrer no cache L1-A1 (isto é, quando N é igual a 1). Como tal, uma linha de cache L1-A1 obtém um influxo de dados, que inclui o valor então corrente de localização de memória solicitada. Dependendo dos atributos de cache (por exemplo, quais camadas de nível superior existem, se a arquitetura de cache é inclusiva ou exclusiva, etc.) e estado de cache corrente, um influxo poderia ser originado da memória de sistema 202 ou de um cache de nível mais alto (por exemplo, L2-A1 e/ou L3-A).

[0055] O método 300 também inclui um ato 302 de verificar um cache de nível (N+i) para determinar se os dados do influxo já foram registrados com base na execução em uma segunda unidade de processamento. Em algumas modalidades, o ato 302 compreende, com base na detecção do influxo para o primeiro cache de nível N, verificar o cache de nível (N+i) para determinar se os dados para a localização da memória foram anteriormente registrados em nome de uma segunda unidade de processamento. Por exemplo, se i for igual a 1 - de modo que o cache de nível (N+i) compreende um cache de nível (N+1) - o processador 201 pode verificar um cache L2, tal como L2-A1 (o qual tem conhecimento do cache L1 -A2 e da unidade de processamento A2). Esta verificação pode ser utilizada para determinar se os dados da localização de memória foram anteriormente registrados em nome da unidade de processamento A2. Estes dados podem ter sido anteriormente registrados, por exemplo, com base na execução anterior de um segundo encadeamento da aplicação 104c na unidade de processamento A2 que causou a falta de cache no cache L1-A2. Em um exemplo alternativo, se i for igual a 2 - de modo que o cache de nível (N+i) compreenda um cache de nível (N+2) - então o processador 201 pode verificar um cache L2, tal como o cache L3- A (o qual tem conhecimento de todos os outros caches no processador 201). Esta verificação pode ser utilizada para determinar se os dados para a localização de memória foram anteriormente registrados em nome de quaisquer unidades de processamento A2-A4 (por exemplo, com base na execução anterior de um ou mais outro(s) encadeamento(s) da aplicação 104c em uma ou mais unidades de processamento A2-A4 que causaram falta de cache nos caches L1-A2 L1-A3 e/ou L1-A4). Note que neste segundo exemplo, o cache L2 pode ser pulado na verificação.

[0056] Como mostrado, o ato 302 poderia ser repetido qualquer número de vezes, enquanto incrementando o valor de i cada vez. Apesar

de i ser tipicamente incrementado por 1 cada vez, poderiam existir modalidades que o incrementam por um número inteiro positivo que é maior do que 1. O efeito de repetir o ato 302 enquanto incrementando i é verificar múltiplos caches de nível superior. Por exemplo, se i = 1, então, quando o ato 302 é inicialmente executado o processador 201 pode verificar uma camada de cache L2 (por exemplo, L2-A1 e/ou L2-A2). Se um conhecimento insuficiente sobre a localização de memória aplicável for encontrado no cache L2, então o processador 201 pode repetir o ato 302 com i = 2, por meio disto verificando uma camada de cache L3 (por exemplo, L3-A). Isto poderia ser continuado por tantos caches de nível quanto o ambiente de computação provê. Se i for alguma vez incrementado por um valor maior que 1, então uma ou mais camada(s) de cache poderiam ser puladas ao longo do caminho. Será apreciado que poderia ser benéfico verificar múltiplos caches de níveis em arquiteturas que proveem caches exclusivos ou que proveem caches que exibem comportamentos inclusivos / exclusivos híbridos. Isto é porque nestas arquiteturas pode não existir garantia que uma camada de cache externa contenha um superconjunto total dos dados na(s) camada(s) de cache interna(s).

[0057] Em vista do acima, será apreciado que o método 300 pode operar em ambientes, tal como um processador 102 ou 201a, nos quais i é igual a 1, de modo que o cache de nível (N+i) compreende um cache de nível (N+1), e no qual o processador também compreende um cache de nível (N+2) que está configurado como um armazenamento de suporte para o cache de nível (N+1). Nestes ambientes, verificar o cache de nível (N+1) para determinar se os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento (isto é, ato 302) pode compreender determinar que nenhuma linha de cache no cache de nível (N+1) corresponde à locali-

zação de memória. Ainda, verificar o cache de nível (N+2) para determinar se os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento.

[0058] Como mostrado, com base no resultado do ato 302, o método inclui um ato 303 de, quando os dados já foram registrados, registrar o influxo por referência; ou um ato 304 de, quando os dados ainda não foram registrados, registrar o influxo por valor.

[0059] Em algumas modalidades, o ato 303 compreende, quando os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento. Continuando os exemplos acima, por exemplo, se uma verificação do cache de nível (N+1) L2-A1 e/ou uma verificação do cache de nível (N+2) L3-A resultar em uma determinação que os dados / localização de a memória já foram registrados em nome da unidade de processamento A2 (com base em um influxo para o cache L1-A2), então, o processador 201a pode fazer com que o influxo para o cache L1-A1 seja registrado em nome da unidade de processamento A1 por referência à entrada de registro feita para a unidade de processamento A2. Exemplos de como registrar por referência pode ser executado são fornecidos posteriormente.

[0060] Observando o resultado alternativo do ato 302, em algumas modalidades, o ato 304 compreende, quando os dados para a localização da memória não foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados por valor em nome da primeira unidade de processamento. Por exemplo, se uma verificação do cache de nível (N+1) L2-A1 e/ou uma verificação do cache de nível (N+2) L3-

A resultar em uma determinação que os dados / localização de memória não foram ainda registrados em nome de outra unidade de processamento, então o processador 201a pode fazer com que o influxo para o cache L1-A1 seja registrado por valor em nome da unidade de processamento A1. Registrar por valor pode incluir, por exemplo, registrar o endereço de memória e o valor de memória em um pacote de dados para a unidade de processamento A1. Note que registrar por valor pode incluir qualquer número de técnicas de compressão para reduzir o número de bits necessários para executar o registro real.

[0061] Como foi descrito em conexão com a Figura 1, o(s) processador(es) 102 podem incluir armazenamento(s) temporário(s) 102d que podem ser utilizados para temporariamente armazenar dados de rastreamento. Assim, no método 300 "fazer com que" diferentes tipos de dados sejam registrados poderia compreender o processador 102 armazenar tais dados no(s) armazenamento(s) temporário(s) 102d. Além disso, ou alternativamente, poderia incluir o processador 102 comunicando tais dados para o rastreador 104a, escrever tais dados no(s) arquivo(s) de rastreamento 104d e/ou notificando o rastreador 104a que os dados estão disponíveis no(s) armazenamento(s) temporário(s) 102d. Em algumas situações, o(s) armazenamento(s) temporário(s) 102d poderiam compreender uma ou mais porções reservadas do(s) cache(s) 102b. Assim, utilizando armazenamentos temporários 102d, nos atos 304/304, fazer com que os dados para a localização de memória sejam registrados - ou por referência ou por valor - em nome da primeira unidade de processamento pode compreender retardar o registro com base na disponibilidade de recursos tais como ciclos de processador, localizações de memória, largura de banda de barramento, etc. Em modalidades nas quais o(s) armazenamento(s) temporário(s) 102d compreende uma ou mais porções reservadas do(s) cache(s) 102b, o registro retardado pode incluir invalidade uma linha de cache (no cache de

nível N e/ou no cache de nível (N+i)), ao invés de removê-lo, de modo a reter os dados para a localização de memória para propósitos de registro retardado.

[0062] A descrição do método 300 referiu a caches de camada superior tendo "conhecimento" sobre caches de camada inferior. A forma específica do "conhecimento" que um cache de camada superior retém sobre o caches de nível inferior pode variar, e exemplos agora seguem.

[0063] Em uma forma básica, este "conhecimento" poderia ser a mera presença de uma linha de cache em um cache de nível superior que corresponde a linha(s) de cache em cache(s) de nível inferior (isto é, linhas de cache que correspondem à mesma localização de memória e dados de memória). Como acima mencionado, em caches inclusivos a(s) camada(s) superior(es) armazenam um superconjunto dos dados na(s) camada(s) abaixo destas. Por exemplo, suponha que os caches na Figura 2A sejam inclusivos. Neste caso, quando a atividade pela unidade de processamento A2 faz com que uma localização da memória de sistema 202 seja importada para o cache L1-A2, esta mesma localização de memória é também colocada em cache nos caches L2-A1 e L3-A. Se a atividade da unidade de processamento A2 estiver sendo rastreada, as modalidades podem fazer com que a localização de memória e seu valor sejam registrados em nome da unidade de processamento A2. Posteriormente, se a atividade pela unidade de processamento A1 fazer com que esta mesma localização da memória de sistema 202 seja importada para o cache L1-A1, e esta localização ainda armazena os mesmos dados, estes são servidos do cache L2-A1, já que o cache L2-A1 já tem os dados. Técnicas anteriores podem novamente registrar estes dados para a unidade de processamento de A1 com base nestes sendo um influxo para o cache L2-A1. No entanto, as modalidades aqui podem ao invés reconhecer que a localização de memória e seu valor já existiam no cache L2-A1, e assim já existiam no cache L1-

A2. Como a unidade de processamento A2 está sendo registrada, as modalidades podem reconhecer que a localização de memória e seu valor já teriam sido registrados em nome da unidade de processamento A2, e, portanto, fazer com que esta nova atividade da unidade de processamento A1 seja registrada em referência aos dados de registro anteriormente gravados em nome de unidade de processamento A2.

[0064] Formas mais elaboradas de "conhecimento" por um cache de camada superior são também possíveis. Por exemplo, as modalidades podem estender as linhas de cache em uma ou mais camadas de cache com bits " contábeis" (ou de registro) adicionais que permitem o processador 102 identificar, para cada linha de cache que implementa bits contábeis, se esta linha de cache foi registrada (potencialmente com a identidade de unidade(s) de processamento que registraram a linha de cache). De modo a compreender estes conceitos, a Figura 4A ilustra um cache compartilhado exemplar 400a, similar ao cache compartilhado 203 da Figura 2B, que estende cada uma das suas linhas de cache 404 com um ou mais bit(s) contábeis adicionais 401. Assim, cada linha de cache 404 inclui bit(s) de contábeis 401 bits de endereço convencionais 402, e bits de valor 403.

[0065] Alternativamente, a Figura 4B ilustra um exemplo de um cache compartilhado 400b que inclui linhas de cache convencionais 405 que armazenam endereços de memória 402 e valores 403, assim como uma ou mais linha(s) de cache reservadas 406 para armazenar bits contábeis que aplicam às linhas de cache convencionais 405. Os bits da(s) linha(s) de cache reservadas 406 estão alocados em diferentes grupos de bits contábeis que cada um corresponde a uma diferente das linhas de cache convencionais 405.

[0066] Em uma variação da Figura 4B exemplar, a(s) linha(s) de cache reservadas 406 poderiam ser reservadas como uma (ou mais) modos em cada índice de um cache associativo de conjunto (o qual está

posteriormente discutido em mais detalhes). Por exemplo, em um cache associativo de conjunto de 8 vias uma via em um conjunto poderia ser reservada para bits contábeis que aplicam às outras sete vias no conjunto. Isto pode diminuir a complexidade de implementar linhas de cache reservadas e pode acelerar o acesso às linhas de cache reservadas já que todas as vias em um dado conjunto são tipicamente lidas em paralelo pela maioria dos processadores.

[0067] Independentemente de como os bits contábeis são realmente armazenados, o(s) bit(s) contábeis 401 de cada linha de cache poderiam compreender um ou mais bits que funcionam como um sinalizador (isto é, ligado ou desligado) utilizado pelo(s) processador(es) 102 para indicar se ou não o valor presente na linha de cache foi registrado em nome de uma unidade de processamento (ou, alternativamente, consumido por uma unidade de processamento que participa no registro). Assim, a verificação no ato 302 pode incluir utilizar este sinalizador para determinar se a linha de cache foi registrada por uma unidade de processamento que participa no registro.

[0068] Alternativamente, os bits contábeis 401 de cada linha de cache poderiam compreender uma pluralidade de bits. Pluralidades de bits poderiam ser utilizadas em diversos modos. Utilizando uma proposta, aqui referida como "bits unitários", os bits contábeis 401 de cada linha de cache podem incluir um número de bits unitários igual a um número de unidades de processamento 102a do processador 102 (por exemplo, o número de unidades de processamento lógicas se o processador 102 suportar hiperencadeamento ou o número de unidades de processamento físicas se o hiperencadeamento não for suportado). Estes bits unitários podem ser utilizados pelo processador 102 para rastrear qual ou mais unidade(s) de processamento específicas registraram na linha de cache (se alguma). Assim, por exemplo, um cache que é compartilhado por duas unidades de processamento 102a poderia associar dois

bits unitários com cada linha de cache.

[0069] Em outra proposta para utilizar pluralidades de bits contábeis 401, referidos aqui como "bits de índice", os bits contábeis 401 de cada linha de cache podem incluir um número de bits de índice suficientes para representar um índice para cada uma das unidades de processamento 102a de um processador 102 do sistema de computador 101 que participam em registro, potencialmente juntamente com um valor "reservado" (por exemplo, -1). Por exemplo, se o processador 102 incluir 128 unidades de processamento 102a, estas unidades de processamento podem ser identificadas por um valor de índice (por exemplo, 0-127) utilizando somente sete bits de índice por linha de cache. Em algumas modalidades, um valor de índice é reservado (por exemplo, "inválido") para indicar que nenhum processador registrou uma linha de cache. Assim, isto significaria que os sete bits de índice seriam realmente capazes de representar 127 unidades de processamento 102a, mais o valor reservado. Por exemplo, valores binários 0000000 - 1111110 poderiam corresponder a localizações de índice 0-126 (decimal), e o valor binário 1111111 (por exemplo, -1 ou 127 decimal, dependendo da interpretação) poderia corresponder a "inválido", para indicar que nenhum processador registrou a linha de cache correspondente - apesar de esta notação poderia variar, dependendo da implementação. Assim, bits unitários podem ser utilizados pelo processador 102 para indicar se a linha de cache foi registrada (por exemplo, um valor outro que -1), e como um índice para uma unidade de processamento específica que registrou a linha de cache (por exemplo, a unidade de processamento que mais recentemente a consumiu). Esta segunda proposta para utilizar pluralidades de bits contábeis 401 tem a vantagem de suportar um grande número de unidades de processamento com pouca sobrecarga no cache 102b, com a desvantagem de menos granularidade do que a pri-

meira proposta (isto é, somente uma unidade de processamento é identificada de cada vez).

[0070] Em vista do acima, será apreciado que, no ato 302, verificar o nível de cache (N+i) para determinar se os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento poderia compreende determinar se uma linha de cache no cache de nível (N+i) que corresponde à localização de memória tem um ou mais conjuntos de bits contábeis.

[0071] Outro mecanismo que pode ser utilizado para determinar se uma linha de cache foi registrada é utilizar caches associativos por conjunto e bloqueio de via. Como um cache de processador 102b é geralmente muito menor do que a memória de sistema 103 (frequentemente por ordem de magnitude), assim existem usualmente muito mais localizações de memória na memória de sistema 103 do que existem linhas em qualquer dada camada do cache 102b. Como tal, alguns processadores definem mecanismos para mapear múltiplas localizações de memória de memória de sistema para linhas(s) de uma ou mais camadas de cache. Os processadores geralmente empregam uma de duas técnicas gerais: mapeamento direto e mapeamento associativo (ou associativo de conjunto). Utilizando o mapeamento direto, diferentes localizações de memória na memória de sistema 103 são mapeadas para apenas uma linha em uma camada de cache, de modo que cada localização de memória pode somente ser colocada em cache em uma linha específica naquela camada.

[0072] Utilizando o mapeamento associativo por conjunto, por outro lado, diferentes localizações na memória de sistema 103 podem ser colocadas em cache para uma de múltiplas linhas em uma camada de cache. A Figura 5 ilustra um exemplo 500 do mapeamento associativo por conjunto entre a memória de sistema e um cache. Aqui, as linhas de

cache 504 de uma camada de cache 502 estão logicamente particionadas em diferentes conjuntos de duas linhas de cache cada, incluindo um primeiro conjunto de duas linhas de cache 504a e 504b (identificadas como índice 0), e um segundo conjunto de dois linhas de cache 504c e 504d (identificadas como índice 1). Cada linha de cache em um conjunto está identificada como uma diferente "via", de modo que a linha de cache 504a é identificada pelo índice 0, via 0, a linha de cache 504b é identificada pelo índice 0, via 1, e assim por diante. Como ainda apresentado, as localizações de memória 503a, 503c, 503e e 503g (índices de memória 0, 2, 4, e 6) são mapeadas para o índice 0. Como tal, cada uma destas localizações na memória de sistema pode ser colocada em cache em qualquer linha de cache dentro do conjunto no índice 0 (isto é, linhas de cache 504a e 504b). Os padrões específicos dos 30 mapeamentos apresentados são para propósitos ilustrativos e conceituais somente, e não devem ser interpretados como um único modo no qual os índices de memória podem ser mapeados para linhas de cache.

[0073] Os caches associativos por conjunto são geralmente referidos como sendo caches associativos por conjuntos de N vias, onde N é o número de "vias" em cada conjunto. Assim, o cache 500 da Figura 5 seria referido como um cache associativo por conjunto de duas vias. Os processadores comumente implementam caches de N vias onde N é uma potência de dois (por exemplo, 2, 4, 8, etc.), com N valores de 4 e 8 sendo comumente escolhidos (apesar de que as modalidades aqui não estão limitadas a quaisquer valores de N ou subconjuntos de valores de N). Notadamente, um cache associativo por conjunto de uma via é geralmente equivalente a um cache mapeado direto, já que cada conjunto contém somente uma linha de cache. Além disso, se N for igual ao número de linhas no cache, este é referido como um cache totalmente associativo, já que este compreende um único conjunto que contém todas as linhas no cache. Em caches totalmente associativos, qualquer

localização de memória pode ser colocada em cache em qualquer linha no cache.

[0074] É notado que a Figura 5 representa uma vista simplificada de memória de sistema e caches, de modo a ilustrar os princípios gerais. Por exemplo, apesar da Figura 5 mapear localizações de memória individuais para linhas de cache, será apreciado que cada linha em um cache pode armazenar dados relativos a múltiplas localizações endereçáveis na memória de sistema. Assim, na Figura 5, cada localização (503a-503h) na memória de sistema (501) pode realmente representar uma pluralidade de localizações de memória endereçáveis. Além disso, os mapeamentos podem ser entre endereços físicos reais na memória de sistema 501 e as linhas no cache 502, ou podem utilizar uma camada intermediária de endereços virtuais.

[0075] Os caches associativos por conjunto podem ser utilizados para determinar se uma linha de cache foi registrada através da utilização de bloqueio de via. O bloqueio de via bloqueia ou reserva uma ou mais vias em um cache para algum propósito. Especificamente, as modalidades aqui utilizam o bloqueio de via para reservar uma ou mais vias para uma unidade de processamento que está sendo rastreada, de modo que as vias bloqueadas / reservadas são utilizadas exclusivamente para armazenar faltas de cache relativas à execução naquela unidade. Assim, referindo de volta à Figura 5, se "via 0" fosse bloqueado para uma unidade de processamento rastreada, então as linhas de cache 504a e 504c (isto é, índice 0, via 0 e índice 1, via 0) seriam utilizadas exclusivamente para faltas de cache relativas à execução daquela unidade, e as linhas de cache restantes seriam utilizadas para todas as outras faltas de cache. Assim, de modo a determinar se uma linha de cache específica foi registrada, o processador 102 precisa somente determinar se a linha de cache armazenada em uma camada de cache de

"N+1" faz parte de uma via que foi reservada para uma unidade de processamento rastreada.

[0076] Em vista do acima, será apreciado que, no ato 302, verificar o cache de nível (N+i) para determinar se os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento poderia compreender determinar se uma linha de cache no cache de nível (N+i) que corresponde à localização de memória está armazenada em uma via que corresponde a uma unidade de processamento registrada.

[0077] Como foi anteriormente explicado, os caches operam de acordo com um CCP, o que define como a consistência é mantida entre vários caches conforme as unidades de processamento leem de e escrevem em dados em cache, e como assegurar que as unidades de processamento sempre leiam dados válidos de uma dada localização em um cache. Como tal, em conexão com a operação de um cache, um processador 102 mantém e armazena dados de estado de CCP. A granularidade com a qual diferentes processadores e/ou diferentes CCPs rastreiam estado de coerência de cache e tornam estes dados de coerência de cache disponíveis para um rastreador 104a pode variar. Por exemplo, em uma extremidade do espectro, alguns processadores / CCPs rastreiam coerência do cache por linha de cache assim como por unidade de processamento. Estes processadores / CCPs podem, portanto, rastrear o estado de cada linha de cache conforme esta refere-se a cada unidade de processamento. Isto significa que uma única linha de cache pode ter informações sobre seu estado já que esta refere a cada unidade de processamento 102a. Outros processadores / CCPs são menos granulares, e rastreiam coerência de cache no nível de linha de cache somente (e faltam informações por unidade de processamento). Na outra extremidade do espectro, os fabricantes de processadores podem escolher por rastrear a coerência de cache no nível da linha de

cache somente para eficiência, já que somente um processador pode possuir uma linha exclusivamente (exclusiva, modificada, etc.) de cada vez. Como um exemplo de granularidade média, um processador / CCP pode rastrear coerência de cache por linha de cache, assim como um índice (por exemplo, 0, 1, 2, 3 para um processador de quatro unidades de processamento) para a unidade de processamento que tem estado de linha de cache corrente.

[0078] Qualquer que seja a granularidade com a qual os dados do estado de CCP sejam mantidos em um dado processador, estes dados do estado de CCP podem ser incluídos no "conhecimento" que um cache de nível (N+i) tem sobre dados em cache. Especificamente, os dados de estado de CCP associados a com uma dada linha de cache em um cache de nível (N+i) podem ser utilizados para determinar se aquela linha de cache foi registrada por uma das unidades de processamento. Por exemplo, se os dados de estado de CCP indicarem que uma unidade de processamento específica tomou uma dada linha de cache como "compartilhada", estes dados podem, por sua vez, ser utilizados para determinar que a unidade de processamento registrou uma leitura da linha de cache. Assim, será apreciado que, no ato 302, verificar o cache de nível (N+i) para determinar se os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento poderiam compreender determinar se uma linha de cache no cache de nível (N+i) que corresponde à localização de memória tem dados de estado de CCP associados que são utilizáveis para determinar que a linha de cache foi registrada.

[0079] No ato 303, um influxo de dados pode ser registrado por referência a dados anteriormente registrados (tipicamente dados registrados por uma diferente unidade de processamento que aquela que causou o presente influxo). Registrar por referência pode ser executado uti-

lizando um ou mais de uma variedade de métodos (incluindo suas combinações), alguns dos quais são agora descritos.

[0080] Um primeiro método registra por referência a um endereço de memória anteriormente registrado. Por exemplo, suponha que a unidade de processamento A2 na Figura 2A registrou dados que representam um endereço de memória específico (isto é, na memória de sistema 202) e dados específicos armazenados naquele endereço de memória. Posteriormente, se este endereço de memória específico / dados específicos para um influxo para a unidade de processamento A1, a unidade de processamento A1 poderia armazenar uma entrada de registro que identifica (i) endereço de memória específico e (ii) unidade de processamento A2. Aqui, a unidade de processamento A1 evitou registrar novamente os dados reais armazenados no endereço de memória (os quais podem ser de tamanho considerável). Algumas variantes deste primeiro método poderiam também armazenar dados de ordenação, tal como um MIN de uma série que incrementa através dos fluxos de dados para processar as unidades A1 e A2. Este MIN posteriormente poderia ser utilizado para ordenar este influxo pela unidade de processamento A1 contra um ou mais eventos na unidade de processamento A2 (por exemplo, aqueles que estão também associados com um MIN da mesma série). Consequentemente, no ato 303, fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento poderiam compreender um ou mais de registrar um endereço da localização de memória, ou registrar um endereço da localização de memória e dados de ordenação, tal como um MIN.

[0081] Um segundo método registra por referência a um proprietário anterior de uma linha de cache que armazena os dados. Por exemplo, suponha que a unidade de processamento A2 na Figura 2A registrou

um primeiro influxo de dados. Suponha também que o primeiro influxo fez com que os dados fossem colocados em cache em uma linha de cache de um cache de nível (N+i) (por exemplo, cache L2-A1) - com a unidade de processamento A2 sendo identificada como proprietária da linha de cache. Posteriormente, se a unidade de processamento A1 causar um segundo influxo dos mesmos dados, a unidade de processamento A1 poderia tornar-se a proprietário desta linha de cache no cache de nível (N+i). A unidade de processamento A1 poderia então armazenar uma entrada de registro que identifica o proprietário anterior da linha de cache (isto é, a unidade de processamento A2), de modo que a entrada de registro de A2 possa ser utilizada posteriormente para obter os dados. Isto significa que registrar por referência pode envolver gravar a identidade de uma linha de cache juntamente com um proprietário anterior de uma linha de cache (por exemplo, potencialmente evitando gravar de endereços de memória e valores de memória). Consequentemente, no ato 303, fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento poderiam compreender registrar a segunda unidade de processamento como um proprietário anterior de uma linha de cache que corresponde à localização de memória.

[0082] Um terceiro método registra por referência a dados de CCP. Por exemplo, como mencionado, os CCPs podem armazenar o estado de coerência de cache sobre cada linha de cache conforme diferentes unidades de processamento a levam para leitura e escrita. A granularidade destes dados pode variar dependendo a implementação de processador, mas poderia, por exemplo, rastrear o estado de coerência de cache de cada linha de cache conforme esta refere-se a cada unidade de processamento, rastrear o estado de coerência de cache de cada

linha de cache juntamente com um índice (por exemplo, 0, 1, 2, 3 etc.) para a unidade de processamento que possui o estado de linha de cache corrente etc. O terceiro método utiliza dados de CCP disponíveis para rastrear qual(is) unidade(s) de processamento anteriormente possuíam o estado de coerência de cache para uma linha de cache, qual estado de coerência de cache pode então ser utilizado para identificar qual(is) unidade(s) de processamento registraram o valor de uma linha de cache. Isto significa que registrar por referência pode envolver gravar dados de CCP para uma linha de cache (por exemplo, novamente potencialmente evitando gravar endereços de memória e valores de memória). Consequentemente, no ato 303, fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento poderia compreender registrar dados de CCP referenciando a segunda unidade de processamento.

[0083] Um quarto método registra por referência a uma via de cache. Como mencionado, caches associativos por conjunto podem ser utilizados para determinar se uma linha de cache foi registrada através da utilização de bloqueio de via. Por exemplo, suponha que bloqueio de via é utilizado para reservar uma ou mais vias para a unidade de processamento P2, e que P2 registra um primeiro influxo de dados. O primeiro influxo também resulta em um cache de nível (N+i) (por exemplo, cache L2-A1) que armazena dados do primeiro influxo em uma linha de cache associada com aquela via. Quando outra unidade de processamento (por exemplo, P1) tem um segundo influxo dos mesmos dados, a presença desta linha de cache no cache de nível (N+i) indica que P2 já registrou os dados. As modalidades podem registrar uma referência a dados de registro de P2 com base em notar um modo no qual a linha de cache está armazenada e podem novamente potencialmente evitar

registrar endereços e valores de memória. Esta modalidade pode também ser utilizada em conexão com gravar informações de ordenação (por exemplo, MINs) para ordenar eventos entre PI e P2. Consequentemente, no ato 303, fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento poderia compreender um ou mais registrar a uma referência para uma via de cache, ou registrar uma referência para uma via de cache e dados de ordenação.

[0084] Além de registrar um influxo para uma primeira unidade de processamento com base em um influxo anterior por uma segunda unidade de processamento, as modalidades também incluem otimizações para reduzir (e até eliminar) registro quando existem múltiplos influxos dos mesmos dados por uma única unidade de processamento. Por exemplo, referindo à Figura 2A, a unidade de processamento A1 poderia causar uma falta de cache em um cache de nível N (por exemplo, o cache L1-A1) para dados específicos em uma localização de memória. Em resposta, a hierarquia de cache pode importar estes dados para o cache L1-A1, e potencialmente também para o cache de nível (N+i) (por exemplo, o cache L2-A1 e/ou o cache L3-A). Além disso, o influxo pode ser registrado por valor para a unidade de processamento A1. Posteriormente, estes dados poderiam ser removidos do cache L1-A1. Em ambientes de cache típicos, isto poderia resultar nos dados também sendo proativamente removidos do cache L2-A1 e/ou do cache L3-A. No entanto, ao invés de causar remoção(ões) nos caches L2-A1 e/ou L3-A, as modalidades poderiam ao invés reter a(s) linha(s) de cache apropriadas em um ou mais destes caches de nível (N+i). Consequentemente, o método 300 pode compreender remover uma primeira linha de cache

no primeiro cache de nível N que corresponde à localização de memória, enquanto retendo uma segunda linha de cache no cache de nível (N+i) que corresponde à localização de memória.

[0085] Posteriormente, se a unidade de processamento A1 causar uma subsequente falta de cache no cache L1-A1 para os mesmos dados, a(s) linha(s) de cache retidas em um cache de nível (N+i) (por exemplo, os caches L2-A1 e/ou L3-A) podem ser utilizadas para determinar que estes dados já foram registrados em nome da unidade de processamento A1. Assim, em algumas modalidades, esta subsequente falta de cache é registrada em referência à entrada de registro anterior pela unidade de processamento A1. Em outras modalidades, uma entrada de registro poderia ser omitida inteiramente para esta subsequente falta de cache - porque a unidade de processamento A1 já tem os dados em seu rastreamento. Consequentemente, o método 300 pode compreender, com base em detectar um subsequente influxo para o primeiro cache de nível N, o subsequente influxo também compreendendo os dados armazenados na localização de memória, fazendo com que o subsequente influxo seja registrado por referência com base na presença da segunda linha de cache. Além disso, ou alternativamente, o método 300 pode compreender (i) detectar um subsequente influxo para o primeiro cache de nível N com base em execução de código adicional na primeira unidade principal de processamento, o subsequente influxo também compreendendo os dados armazenados na localização de memória, e (ii) com base em pelo menos detectar o subsequente influxo para o primeiro cache de nível N, e com base pelo menos na presença da segunda linha de cache, determinar que o subsequente influxo não precisa ser registrado.

[0086] Consequentemente, as modalidades aqui criam gravações de rastreamento de "viagem no tempo" precisas em bit sobre rastrear

os efeitos de execução através de uma pluralidade de unidades de processamento utilizando pelo menos duas filas ou camadas de caches de processador. Isto poderia incluir modificações no hardware de processador e/ou microcódigo que auxiliam em (i) detectar influxos (isto é, faltas de cache) para um cache de processador interno ou de "camada inferior" com base na atividade por uma unidade de processamento rastreada, e (ii) utilizar um cache de processador compartilhado externo ou de "camada superior" para determinar se dados de um dado influxo já foram registrados no nome de outra unidade de processamento rastreada. Se estes dados já foram registrados, então o influxo pode ser registrado por referência à entrada de registro anterior. Estas técnicas podem ser estendidas para "N" níveis de caches. Gravar arquivos de rastreamento neste modo pode precisar somente modestas modificações de processador e, quando comparado com as propostas de gravação de rastreamento anteriores, isto pode reduzir por diversas ordens de magnitude tanto o impacto de desempenho de gravação de rastreamento assim como tamanho de arquivo de rastreamento.

[0087] A presente invenção pode ser incorporada em outras formas específicas sem afastar de seu espírito ou características essenciais. As modalidades descritas devem ser consideradas em todos os aspectos somente como ilustrativas e não restritivas. O escopo da invenção está, portanto, indicado pelas reivindicações anexas ao invés da descrição acima. Todas as mudanças que vêm dentro do significado e a faixa de equivalência das reivindicações devem ser abrangidas dentro de seu escopo.

REIVINDICAÇÕES

1. Dispositivo de computação, caracterizado pelo fato de que compreende:

uma pluralidade de unidades de processamento;

uma pluralidade de caches de nível N;

um cache de nível (N+i) que está associado com dois ou mais da pluralidade de caches de nível N, e que está configurado como um armazenamento de suporte para a pluralidade de caches de nível N; e

uma lógica de controle que configura o dispositivo de computação para executar pelo menos o seguinte:

detectar um influxo para um primeiro cache de nível N da pluralidade de caches de nível N, o influxo compreendendo dados armazenados em uma localização de memória; e

com base na detecção do influxo para o primeiro cache de nível N, verificar o cache de nível (N+i) para determinar se os dados para a localização de memória foram anteriormente registrados em nome de uma segunda unidade de processamento, e executar um de:

quando os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento; ou

quando os dados para a localização de memória não foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados por um valor em nome da primeira unidade de processamento.

2. Dispositivo de computação de acordo com a reivindicação

1, caracterizado pelo fato de que verificar o cache de nível (N+i) para determinar se os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento compreende um ou mais de:

determinar se uma linha de cache no cache de nível (N+i) que corresponde à localização de memória tem um ou mais conjuntos de bits contábeis;

determinar se uma linha de cache no cache de nível (N+i) que corresponde à localização de memória está armazenado em um modo que corresponde a uma unidade de processamento registrada; ou

determinar se uma linha de cache no cache de nível (N+i) que corresponde à localização de memória tem associado dados de estado de protocolo de coerência de cache (CCP) que são utilizáveis para determinar que a linha de cache foi registrada.

3. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que i é igual a 1, de modo que o cache de nível (N+i) compreende um cache de nível (N+1).

4. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que i é igual a 2, de modo que o cache de nível (N+i) compreende um cache de nível (N+2).

5. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que:

i é igual a 1, de modo que o cache de nível (N+i) compreende um cache de nível (N+1);

o dispositivo de computação também compreende um cache de nível (N+2) que está configurado como um armazenamento de suporte para o cache de nível (N+1); e

verificar o cache de nível (N+1) para determinar se os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento compreende:

determinar que nenhuma linha de cache no cache de nível (N+1) corresponde à localização de memória; e

verificar o cache de nível (N+2) para determinar se os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento.

6. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento compreende um ou mais de:

registrar um endereço da localização de memória;

registrar um endereço da localização de memória e dados de ordenação;

registrar uma referência para uma via de cache;

registrar uma referência para uma via de cache e dados de ordenação;

registrar a segunda unidade de processamento como um proprietário anterior de uma linha de cache que corresponde à localização de memória; ou

registrar dados de protocolo de coerência de cache (CCP) referenciando a segunda unidade de processamento.

7. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento compreende retardar o registro com base em disponibilidade de um ou ambos do processador ou recursos de memória.

8. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que retardar o registo compreende invalidar uma linha de cache para reter os dados para a localização de memória

para o registro retardado.

9. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que a lógica de controle também configura o dispositivo de computação para executar pelo menos o seguinte:

remover uma primeira linha de cache no primeiro cache de nível N que corresponde à localização de memória, enquanto retendo uma segunda linha de cache no cache de nível (N+i) que corresponde à localização de memória; e

com base em detectar um subsequente influxo para o primeiro cache de nível N, o subsequente influxo também compreendendo os dados armazenados na localização de memória, fazer com que o subsequente influxo seja registrado por referência com base na presença da segunda linha de cache.

10. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que a lógica de controle também configura o dispositivo de computação para executar pelo menos o seguinte:

remover uma primeira linha de cache no primeiro cache de nível N que corresponde à localização de memória, enquanto retendo uma segunda linha de cache no cache de nível (N+i) que também corresponde à localização de memória,

detectar um subsequente influxo para o primeiro cache de nível N com base em execução de código adicional na primeira unidade de processamento, o subsequente influxo também compreendendo os dados armazenados na localização de memória; e

com base pelo menos em detectar o subsequente influxo para o primeiro cache de nível N, e com base pelo menos na presença da segunda linha de cache, determinar que o subsequente influxo não precisa ser registrado.

11. Método para gravação de rastreamento com base na gravação de um influxo para um cache de nível inferior por referência a dados de registro anteriores com base no conhecimento de um ou mais caches de nível superior, o método sendo implementado em um dispositivo de computação que inclui (i) uma pluralidade de unidades de processamento, (ii) uma pluralidade de caches de nível N, e (iii) um cache de nível (N+i) que está associado com dois ou mais da pluralidade de caches de nível N, e que está configurado como um armazenamento de suporte para a pluralidade de caches de nível N, caracterizado pelo fato de que o método compreende:

detectar um influxo para um primeiro cache de nível N da pluralidade de caches de nível N, o influxo compreendendo dados armazenados em uma localização de memória; e

com base em detectar o influxo para o primeiro cache de nível N, verificar o cache de nível (N+i) para determinar se os dados para a localização de memória foram anteriormente registrados em nome de uma segunda unidade de processamento, e executar um de:

quando os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento; ou

quando os dados para a localização de memória não foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados por um valor em nome da primeira unidade de processamento.

12. Método de acordo com a reivindicação 11, caracterizado pelo fato de que verificar o cache de nível (N+i) para determinar se os

dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento compreende um ou mais de:

determinar se uma linha de cache no cache de nível (N+i) que corresponde à localização de memória tem um ou mais conjuntos de bits contábeis;

determinar se uma linha de cache no cache de nível (N+i) que corresponde à localização de memória está armazenado em um modo que corresponde a uma unidade de processamento registrada; ou

determinar se uma linha de cache no cache de nível (N+i) que corresponde à localização de memória tem associado dados de estado de protocolo de coerência de cache (CCP) que são utilizáveis para determinar que a linha de cache foi registrada.

13. Método de acordo com a reivindicação 11, caracterizado pelo fato de que i é igual a 1, de modo que o cache de nível (N+i) compreende um cache de nível (N+1).

14. Método de acordo com a reivindicação 11, caracterizado pelo fato de que i é igual a 2, de modo que o cache de nível (N+i) compreende um cache de nível (N+2).

15. Produto de programa de computador, caracterizado pelo fato de que compreende um ou mais meios legíveis por computador que têm armazenados nos mesmos instruções executáveis por computador que configuram um dispositivo de computação que inclui (i) uma pluralidade de unidades de processamento, (ii) uma pluralidade de caches de nível N, e (iii) um cache de nível (N+i) que está associado com dois ou mais da pluralidade de caches de nível N, e que está configurado como um armazenamento de suporte para a pluralidade de caches de nível N, para executar pelo menos o seguinte:

detectar um influxo para um primeiro cache de nível N da

pluralidade de caches de nível N, o influxo compreendendo dados armazenados em uma localização de memória; e

com base em detectar o influxo para o primeiro cache de nível N, verificar o cache de nível (N+i) para determinar se os dados para a localização de memória foram anteriormente registrados em nome de uma segunda unidade de processamento, e executar um de:

quando os dados para a localização de memória foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados em nome da primeira unidade de processamento por referência a dados de registro que foram anteriormente registrados em nome da segunda unidade de processamento; ou

quando os dados para a localização de memória não foram anteriormente registrados em nome da segunda unidade de processamento, fazer com que os dados para a localização de memória sejam registrados por um valor em nome da primeira unidade de processamento.

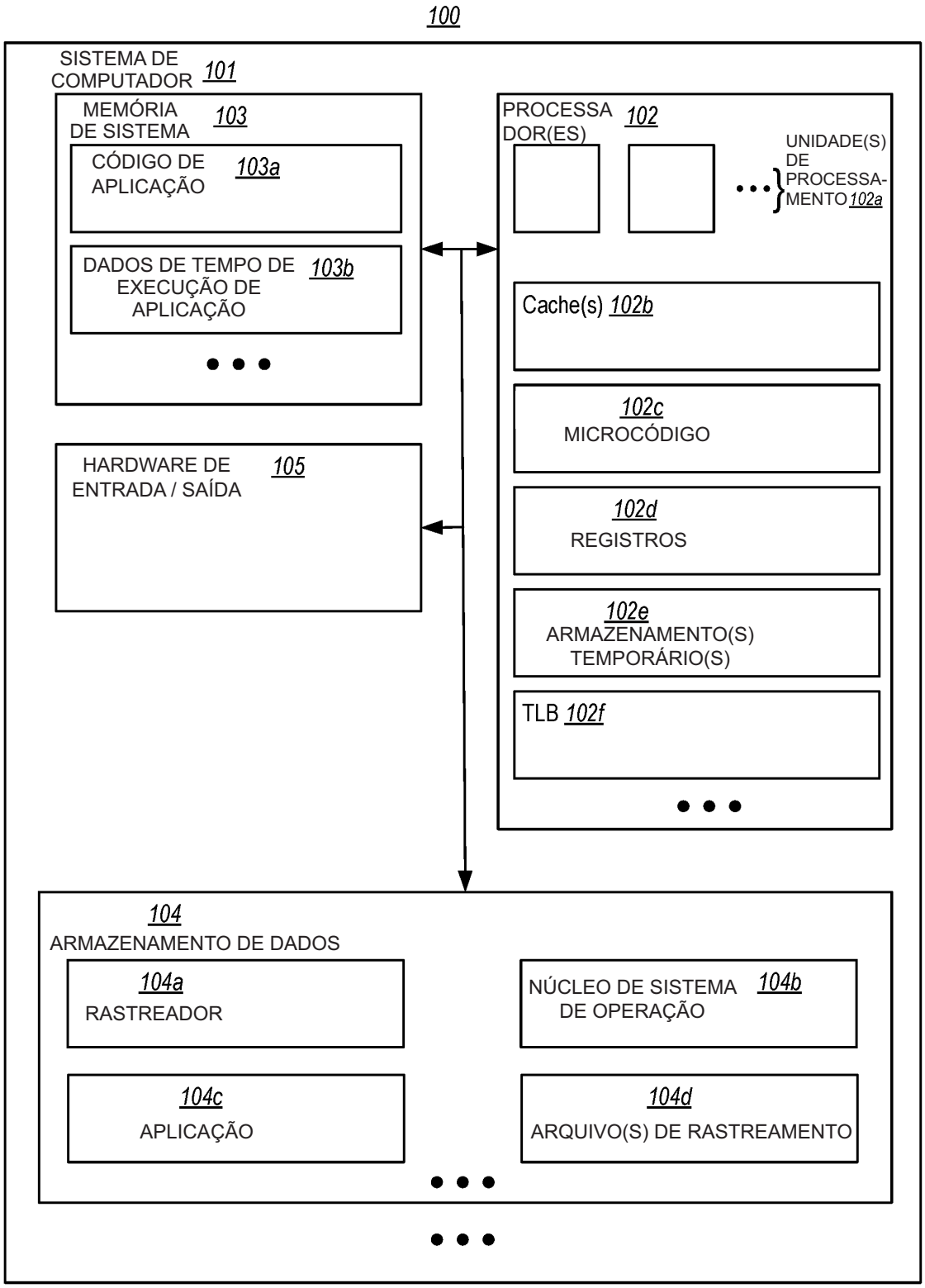

100
SISTEMA DE COMPUTADOR 101
MEMÓRIA DE SISTEMA 103
CÓDIGO DE APLICAÇÃO 103a
DADOS DE TEMPO DE EXECUÇÃO DE APLICAÇÃO 103b
PROCESSADOR(ES) 102
UNIDADE(S) DE PROCESSAMENTO 102a
Cache(s) 102b
102c MICROCÓDIGO
102d REGISTROS
102e ARMAZENAMENTO(S) TEMPORÁRIO(S)
TLB 102f
HARDWARE DE ENTRADA / SAÍDA 105
104 ARMAZENAMENTO DE DADOS
104a RASTREADOR
NÚCLEO DE SISTEMA DE OPERAÇÃO 104b
104c APLICAÇÃO
104d ARQUIVO(S) DE RASTREAMENTO

FIG. 1

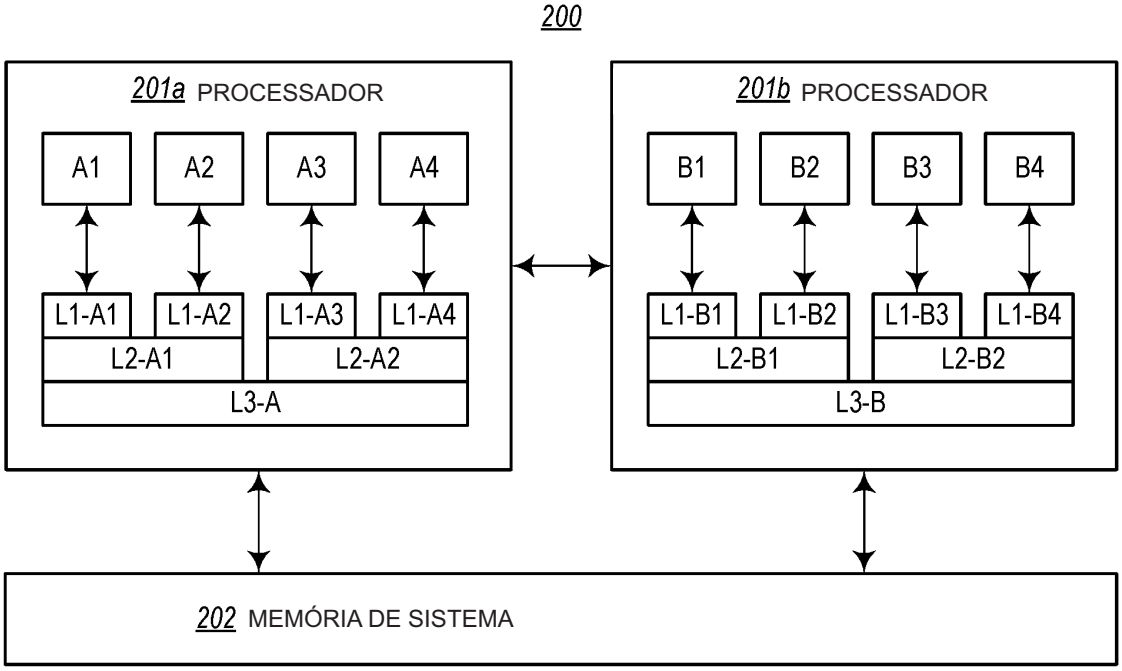

200
201a PROCESSADOR
A1
A2
A3
A4
L1-A1
L1-A2
L1-A3
L1-A4
L2-A1
L2-A2
L3-A
201b PROCESSADOR
B1
B2
B3
B4
L1-B1
L1-B2
L1-B3
L1-B4
L2-B1
L2-B2
L3-B
202 MEMÓRIA DE SISTEMA

FIG. 2A

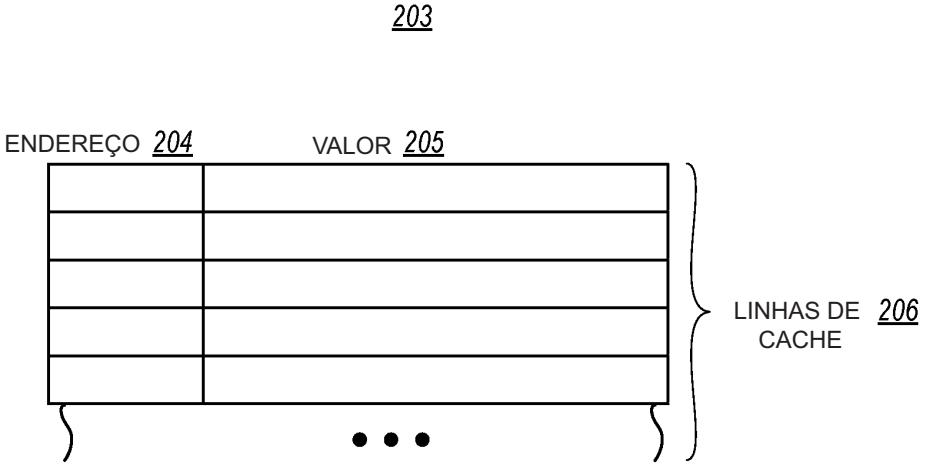

203
ENDEREÇO 204
VALOR 205
LINHAS DE CACHE 206

FIG. 2B

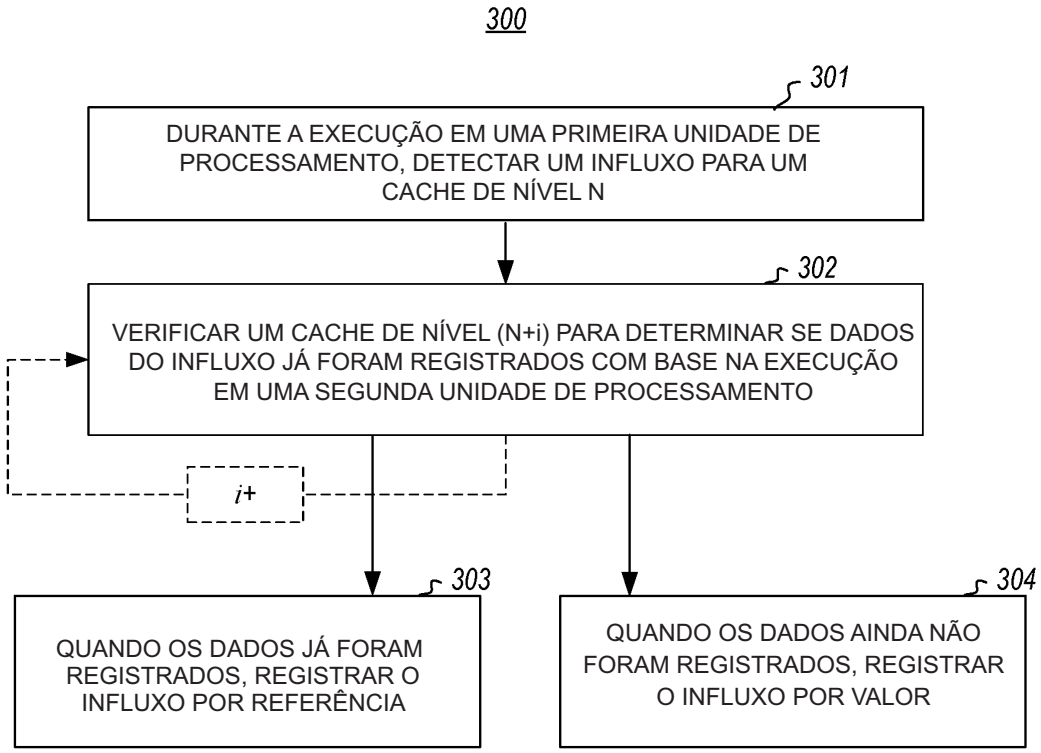

300
301
DURANTE A EXECUÇÃO EM UMA PRIMEIRA UNIDADE DE PROCESSAMENTO, DETECTAR UM INFLUXO PARA UM CACHE DE NÍVEL N
302
VERIFICAR UM CACHE DE NÍVEL (N+i) PARA DETERMINAR SE DADOS DO INFLUXO JÁ FORAM REGISTRADOS COM BASE NA EXECUÇÃO EM UMA SEGUNDA UNIDADE DE PROCESSAMENTO
i+
303
QUANDO OS DADOS JÁ FORAM REGISTRADOS, REGISTRAR O INFLUXO POR REFERÊNCIA
304
QUANDO OS DADOS AINDA NÃO FORAM REGISTRADOS, REGISTRAR O INFLUXO POR VALOR

FIG. 3

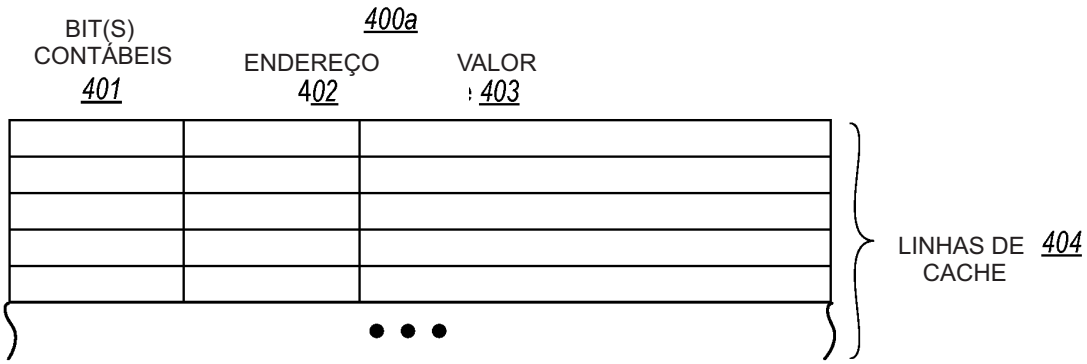

400a
BIT(S) CONTÁBEIS
401
ENDEREÇO
402
VALOR
403
LINHAS DE CACHE
404

FIG. 4A

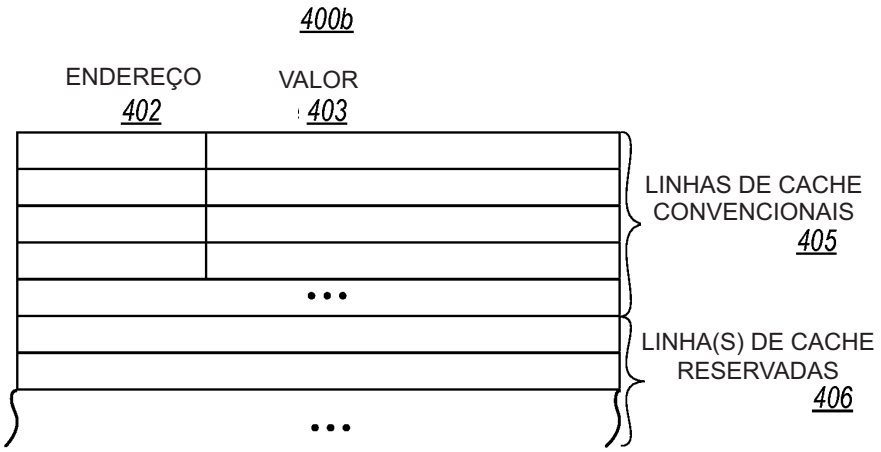

400b
ENDEREÇO
402
VALOR
403
LINHAS DE CACHE CONVENCIONAIS
405
LINHA(S) DE CACHE RESERVADAS
406

FIG. 4B

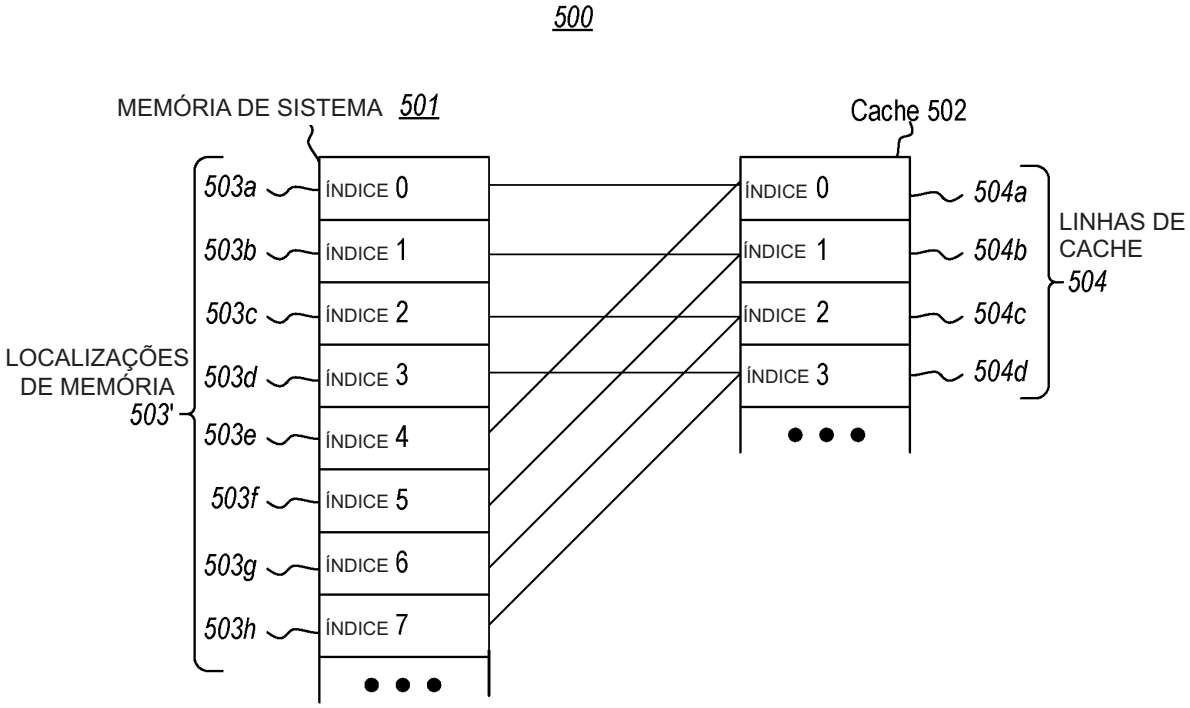

500
MEMÓRIA DE SISTEMA 501
Cache 502
503a
503b
503c
503d
503e
503f
503g
503h
LOCALIZAÇÕES DE MEMÓRIA 503'
ÍNDICE 0
ÍNDICE 1
ÍNDICE 2
ÍNDICE 3
ÍNDICE 4
ÍNDICE 5
ÍNDICE 6
ÍNDICE 7
504a
504b
504c
504d
LINHAS DE CACHE 504

Figure 5

RESUMO

Patente de Invenção: **“GRAVAÇÃO DE RASTREIO REGISTRANDO INFLUXOS PARA UM CACHE DE CAMADA INFERIOR COM BASE EM ENTRADAS EM UM CACHE DE CAMADA SUPERIOR”.**

A presente invenção refere-se à gravação de rastreamento com base em gravar um influxo para um cache de nível inferior por referência a dados de registro anteriores, com base no conhecimento de um cache de nível superior. Um dispositivo de computação inclui uma pluralidade de unidades de processamento, uma pluralidade de caches de nível N e um cache de nível (N+i) que é um armazenamento de suporte para os caches de nível N. Com base na atividade de uma primeira unidade de processamento, o dispositivo de computação detecta um influxo de dados para um primeiro cache de nível N. O dispositivo de computação verifica o cache de nível (N+i) para determinar se os dados já foram registrados para uma segunda unidade de processamento. Com base na verificação, o dispositivo de computação (i) faz com que os dados sejam registrados para a primeira unidade de processamento por referência a dados de registro (isto é, quando os dados já foram registrados), ou faz com que os dados sejam registrados por valor para a primeira unidade de processamento (isto é, quando os dados ainda não foram registrados).