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

(21) BR 112020003342-1 A2

(22) Data do Depósito: 22/06/2018

(43) Data da Publicação Nacional: 18/08/2020

(54) Título: GRAVAÇÃO DE RASTREIO BASEADO EM CACHE UTILIZANDO DADOS DE PROTOCOLO DE COERÊNCIA DE CACHE

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

(30) Prioridade Unionista: 08/03/2018 US 15/915,930; 18/09/2017 US 62/559,780.

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

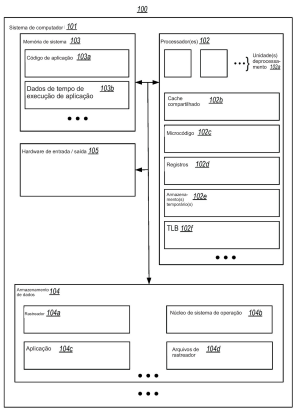
(72) Inventor(es): JORDI MOLA.

(86) Pedido PCT: PCT US2018038875 de 22/06/2018

(87) Publicação PCT: WO 2019/055094 de 21/03/2019

(85) Data da Fase Nacional: 18/02/2020

(57) Resumo: A presente invenção refere-se a executar uma gravação de rastreamento baseada em cache utilizando dados de protocolo de coerência de cache (CCP). As modalidades detectam que uma operação que causa uma interação entre uma linha de cache e um armazena-mento de apoio ocorreu, que o registro é habilitado para a unidade de processamento que causou a operação, que a linha de cache é uma participante no registro, e que o CCP indica que existem dados a serem registrados em um rastreamento. As modalidades então fazem com que os dados sejam registrados no rastreamento, cujos dados são utilizáveis para reproduzir a operação.



Relatório Descritivo da Patente de Invenção para **"GRAVAÇÃO DE RASTREIO BASEADO EM CACHE UTILIZANDO DADOS DE PROTOCOLO DE COERÊNCIA DE CACHE".**

FUNDAMENTOS

[001] Quando escrevendo um código durante o desenvolvimento de aplicações de software, os desenvolvedores comumente dispendem uma quantidade significativa de tempo "depurando" o código para encontrar erros de tempo de execução e outros de código de fonte. Fazendo isto, os desenvolvedores podem adotar diversas propostas para reproduzir e localizar um erro 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. Rastrear de erros de tempo de execução para identificar erros de código pode ocupar uma significativa porção de tempo de desenvolvimento de aplicação.

[002] Muitos tipos de aplicações de depuração ("depuradores") foram desenvolvidos de modo a auxiliar 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 do 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, podem permitir que os desenvolvedores alterar os percursos de execução de código, e/ou podem permitir os desenvolvedores determinem "pontos de interrupção" e/ou "pontos de observação" sobre elementos de interesse de código (os quais, quando alcançados durante a execução, fazem com que a execução do código seja suspensa), entre outras coisas.

[003] Uma forma emergente de aplicações de depuração permite

a depuração "viagem no tempo", "reversa" ou "histórica". Com a depuração de "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 mais tarde, para análise tanto para frente quanto para trás. Por exemplo, os depuradores "viagem no tempo" podem permitir um desenvolvedor determinar pontos de interrupção/pontos de observação para frente (como os depuradores convencionais), assim como pontos de interrupção/pontos de observação reversos.

BREVE SUMÁRIO

[004] As modalidades aqui melhoram as gravações de depuração de "viagem no tempo", utilizando um cache compartilhado de processador, juntamente com seu protocolo de coerência de cache (CCP), de modo a determinar quais dados devem ser registrados em um arquivo de rastreamento. Fazendo isto pode reduzir o tamanho do arquivo de rastreio por diversas ordens de magnitude quando comparado com propostas anteriores, por meio disto significativamente reduzindo o excesso de gravação de rastreamento.

[005] Em algumas modalidades, são implementados em ambientes de computação que incluem (i) uma pluralidade de unidades de processamento, e (ii) uma memória de cache que compreende uma pluralidade de linhas de cache que são utilizadas para colocar em cache dados de um ou mais armazenamentos de apoio e que são compartilhadas pela pluralidade de unidades de processamento. A consistência entre dados na pluralidade de linhas de cache e os um ou mais armazenamentos de apoio é gerenciada de acordo com um protocolo de coerência de cache.

[006] Estas modalidades incluem executar uma gravação de ras-

treamento baseada em cache utilizando dados de CCP. Estas modalidades incluem determinar que uma operação causou uma interação entre uma linha de cache específica da pluralidade de linhas de cache e os um ou mais armazenamentos de apoio, que um registro é permitido para uma unidade de processamento específica da pluralidade de unidades de processamento que causou a operação, que a linha de cache específica é uma participante no registro, e que o CCP indica que existem dados a serem registrados em um rastreamento. Com base pelo menos nestas determinações, as modalidades fazem com que estes dados sejam registrados no rastreamento. Os dados são utilizáveis para reproduzir a operação.

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

BREVE DESCRIÇÃO DOS DESENHOS

[008] De modo a descrever um modo no qual as vantagens e características acima recitadas e outras da invenção podem ser obtidas, uma descrição mais específica da invenção brevemente acima descrita será criada 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 limitantes de seu escopo, a invenção será descrita e explicada com especificidade e detalhes adicionais através da utilização dos desenhos acompanhantes nos quais:

[009] Figura 1 ilustra um ambiente de computação exemplar que facilita a gravação de rastreamentos "precisos em bit" de execução de código através de caches compartilhados utilizando dados de protoco-

lo de coerência de cache (CCP);

[0010] Figura 2 ilustra um exemplo de um cache compartilhado;

[0011] Figura 3 ilustra um fluxograma de um método exemplar para executar uma gravação de rastreamento baseada em cache utilizando dados de CCP;

[0012] Figura 4A ilustra um cache compartilhado exemplar que estende cada uma de suas linhas de cache com um ou mais bits de contagem adicionais;

[0013] Figura 4B ilustra um exemplo de um cache compartilhado que inclui uma ou mais linhas de cache reservadas para armazenar bits de contagem que aplicam a linhas de cache convencionais;

[0014] Figura 5 ilustra um exemplo de mapeamentos de cache associativos;

[0015] Figura 6A ilustra uma tabela que mostra uma atividade de leitura e escrita exemplar por quatro unidades de processamento sobre uma única linha em um cache compartilhado;

[0016] Figura 6B ilustra uma tabela que mostra um estado de coerência de cache rastreado exemplar com base na atividade de leitura e escrita mostrada na Figura 6A;

[0017] Figura 6C ilustra uma tabela que mostra dados exemplares armazenados em bits de contagem (isto é, bits de unidades, bits de índice, e/ou bits de sinalização) de um cache compartilhado com base na atividade de leitura e escrita mostrada na Figura 6A;

[0018] Figura 6D ilustra uma tabela que mostra registro de dados exemplares que poderiam ser escritos em arquivos de rastreamento em conexão com a atividade de leitura e escrita mostrada na Figura 6A;

[0019] Figura 7A ilustra um exemplo no qual algumas transições de leitura -> leitura poderiam ser omitidas de um rastreamento dependendo de como os processadores são rastreados;

[0020] Figura 7B ilustra um exemplo de registro de dados que omitem as transições de leitura -> leitura destacadas na Figura 7A;

[0021] Figura 7C ilustra uma tabela que mostra registro de dados exemplares que poderiam ser gravados se "bits de índice" forem utilizados e índices forem atualizados nas leituras;

[0022] Figura 8A ilustra um ambiente de computação exemplar que inclui dois processadores, cada um incluindo quatro unidades de processamento, e caches L1-L3;

[0023] Figura 8B ilustra uma tabela que mostra operações de leitura e escrita exemplares executadas por algumas das unidades de processamento da Figura 8A;

[0024] Figura 9A ilustra uma tabela que mostra leituras e escritas exemplares por duas unidades de processamento;

[0025] Figura 9B ilustra um exemplo de uma tabela que compara quando as entradas de registro poderiam ser feitas um ambiente que provê informações de unidade de CCP mais um bit de sinalização de linha de cache, versus um ambiente que provê informações de índice de CCP mais um bit de sinalização de linha de cache;

[0026] Figura 10A ilustra um exemplo de diferentes partes de um endereço de memória, e sua relação com caches associativos; e

[0027] Figura 10B ilustra um exemplo de erros de cache de registro e remoções de cache em um cache associativo.

DESCRIÇÃO DETALHADA

[0028] As modalidades aqui melhoram as gravações de depuração de "viagem no tempo", utilizando um cache compartilhado de processador, juntamente com seu protocolo de coerência de cache, de modo a determinar quais dados devem ser registrados em um arquivo de rastreamento. Fazendo isto pode reduzir o tamanho de arquivo de rastreamento por diversas ordens de magnitude quando comparado com propostas anteriores, por meio disto significativamente reduzindo o ex-

cesso de gravação de rastreamento.

[0029] A Figura 1 ilustra um ambiente de computação exemplar 100 que facilita a gravação de rastreamentos "precisos em bit" de código de execução através de caches compartilhados utilizando dados de protocolo de coerência de cache. Como apresentado, as modalidades podem compreender ou utilizar um sistema de computador de utilização especial ou de utilização 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.

[0030] As modalidades dentro do escopo da presente invenção incluem meios físicos e outros meios legíveis por computador para executar ou armazenar instruções e/ou estruturas de dados executáveis por computador. Tal meio legível por computador pode ser qualquer meio disponível que possa ser acessado pelo sistema de computador 101. Os meios legíveis por computador que armazenam instruções e/ou estruturas de dados executáveis por computador são dispositivos de armazenamento de computador. Os meios legíveis por computador que carregam instruções e/ou estruturas de dados executáveis por computador 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.

[0031] Os dispositivos de armazenamento de computador são dispositivos de hardware físicos que armazenam instruções e/ou estruturas de dados executáveis por computador. 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"), armaze-

namento de disco ótico, armazenamento de disco magnético ou outros dispositivos de armazenamento magnético, ou quaisquer outro dispositivo(s) de hardware os quais podem ser utilizados para armazenar um código de programa na forma de instruções ou estruturas de dados executáveis por computador, 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 e/ou estruturas de dados por computador, ou outro armazenamento tal como armazenamento em processador, como posteriormente discutido.

[0032] O meio de transmissão pode incluir uma rede e/ou conexão de dados as quais podem ser utilizadas para carregar um código de programa na forma de instruções ou estruturas de dados executáveis por computador, 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 um meio de transmissão. Combinações dos acima devem também estar incluídas dentro do escopo de meio legível por computador. Por exemplo, o hardware de entrada/saída 105 pode compreender um hardware (por exemplo, um módulo de interface de rede (por exemplo, uma "NIC")) que conecta uma rede e/ou conexão de dados os quais podem ser utilizados para carregar um código de programa na forma de instruções ou estruturas de dados executáveis por computador.

[0033] Além disso, quando alcançando vários componentes de sistema de computador, o código de programa na forma de instruções ou estruturas de dados executáveis pode computador pode ser transferido automaticamente dos meios de transmissão para dispositivos de armazenamento de computador (ou vice-versa). Por exemplo, as instruções ou estruturas de dados executáveis por computador recebidas sobre uma rede ou conexão de dados podem ser armazenadas temporariamente em RAM dentro de uma 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.

[0034] 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 em formato intermediário tal como linguagem assembly, ou mesmo código de fonte.

[0035] Aqueles versados na técnica apreciarão que a invenção pode ser praticada em ambientes de computação de rede com muitos tipos de configurações de sistema de computador, incluindo, computadores pessoais, computadores desktop, computadores laptops, processadores de mensagens, dispositivos portáteis, sistemas de múltiplos processadores, eletrônica de consumidor baseado em microprocessador ou programável, 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 distribuídos onde sistemas de computador localizações e remotos, os quais estão conectados (ou por conexões de dados com fio, conexões 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.

[0036] Aqueles versados na técnica também apreciarão que a invenção possa ser praticada em um ambiente de computação de nuvem. Os ambientes de computação de nuvem podem ser distribuídos, apesar disto não ser requerido. Quando distribuídos, os ambientes de computação de nuvem podem ser distribuídos internacionalmente dentro de uma organização e/ou terem componentes possuídos através de múltiplas organizações. Nesta descrição e nas reivindicações seguintes, "computação de nuvem" é definida como um modelo para permitir acesso da rede sob demanda a um grupamento compartilhado de recursos de computação configuráveis (por exemplo, redes, servidores, armazenamento, aplicações e/ou serviços). A definição de "computação de nuvem" não está limitada a qualquer uma das outras numerosas vantagens que podem ser obtidas de tal modelo quando apropriadamente implantadas.

[0037] Um modelo de computação de nuvem pode ser composto de várias características, tal como autosserviço sob demanda, amplo acesso de rede, grupamento de recursos, rápida elasticidade, serviço medido, e assim por diante. Um modelo de computação de nuvem pode também vir na forma de vários modelos de serviço, tal como, por exemplo, Software como um Serviço ("SaaS"), Plataforma como um

Serviço ("PaaS") e Infraestrutura como um Serviço ("IaaS "). O modelo de computação de nuvem pode também ser distribuído utilizando diferentes modelos de distribuição tal como nuvem privada, nuvem de comunidade, nuvem pública, nuvem híbrida, e assim por diante.

[0038] Algumas modalidades, tal como um ambiente de computação de nuvem, podem compreender um sistema que inclui um ou mais hospedeiro 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 um 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 vista 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, apesar de que a máquina virtual somente interfaceia com a aparência (por exemplo, um recurso virtual) de um recurso físico. Exemplos de recursos físicos que incluem capacidade de processamento, memória, espaço de disco, largura de banda de rede, unidades de mídia, e assim por diante.

[0039] Como ilustrado, o armazenamento de dados 104 pode armazenar instruções e/ou estruturas de dados executáveis por computador 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 objeto 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, tal como es-

truturas de dados de tempo de execução, instruções executáveis por computador, etc. Assim, a Figura 1 ilustra a memória de sistema 103 como incluindo um código de aplicação de tempo de execução 103a e dados de tempo de execução de aplicação 103b (por exemplo, cada um corresponde com a aplicação 104c).

[0040] O rastreador 104a é utilizável para gravar um rastreamento preciso de bits de execução de uma aplicação tal como a aplicação 104c, e 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, um tecido de nuvem, etc. Apesar do(s) arquivo(s) de rastreamento 104d serem apresentados como 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.

[0041] 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 unidades 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 apresentado por um núcleo físico que suporta hiperencadeamento, no qual mais do que um encadeamento de aplicação executa em um núcleo físico). Assim, por exemplo, mesmo que o processador 102 possa em algumas modalidades incluir somente uma única unidade de processamento físico (núcleo), este poderia incluir duas ou mais unidades de processamento lógico 102a apresentadas por aquela única unidade de processamento físico.

[0042] 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 de processador predefinida (ISA). A ISA específica de cada processador 102 varia com base no fabricante de processador e modelo de processador. ISAs comuns incluem as arquiteturas IA-64 e IA-32 da INTEL, INC., a arquitetura MD64 da ADVANCED MICRO DEVICES, INC., e várias arquiteturas de Máquina RISC Avançada ("ARM") da ARM HOLDINGS, PLC, apesar de um grande número de outras ISAs existirem e poderem ser utilizadas 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.

[0043] Cada unidade de processamento 102a obtém instruções de processador de um cache compartilhado 102b, e executa as instruções de processador com base em dados no cache compartilhado 102b, com base em dados em registros 102d, e/ou sem dados de entrada. Em geral, o cache compartilhado 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 cópias no processador de porções de um armazenamento de apoio, tal como a memória de sistema 103 e/ou outro cache. Por exemplo, quando executando o código de aplicação 103 a, o cache compartilhado 102b contém porções dos dados de tempo de execução de aplicação 103b. Se a(s) unidade(s) de processamento 102a requererem dados ainda não armazenados no cache compartilhado 102b, então "erro de cache" ocorre, e estes dados serão buscados da memória de sistema 103 (potencialmente "removendo" alguns outros dados do cache compartilhado 102b).

[0044] Tipicamente, um cache compartilhado 102b compreende

uma pluralidade de "linhas de cache", cada uma das quais armazena um pedaço de memória do armazenamento de apoio. Por exemplo, a Figura 2 ilustra um exemplo de pelo menos uma porção de um cache compartilhado 200, o qual inclui uma pluralidade de linhas de cache 203, cada uma das quais inclui uma porção de endereço 201 e uma porção de valor 202. A porção de endereço 201 de cada linha de cache 203 pode armazenar um endereço no armazenamento de apoio (por exemplo, memória de sistema 103) para o qual a linha corresponde, e a porção de valor 202 pode inicialmente armazenar um valor recebido do armazenamento de apoio. A porção de valor 202 pode ser modificada pelas unidades de processamento 102a, e eventualmente serem removidas de volta para o armazenamento de apoio. Como indicado pelas elipses, um cache compartilhado 200 pode incluir um grande número de linhas de cache. Por exemplo, um processador INTEL contemporâneo pode conter um cache de camada 1 que compreende 512 ou mais linhas de cache. Neste 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 8 bytes (64 bits).

[0045] O endereço armazenado na porção de endereço 201 de cada linha de cache 203 pode ser um endereço físico, tal como o endereço de memória real na memória de sistema 103. Alternativamente, o endereço armazenado na porção de endereço 201 de cada linha de cache 203 pode ser um endereço virtual, o qual é um endereço que é atribuído ao endereço físico para prover uma abstração. Tais abstrações podem ser utilizadas, por exemplo, para facilitar isolamento de memória entre diferentes processos que executam no(s) processador(es) 102. Quando endereços virtuais são utilizados, um processador 102 pode incluir um armazenamento de lookaside de tradução (TLB) 102f (usualmente parte de uma unidade de gerenciamento de memória (MMU)), o qual mantém mapeamentos entre o endereço de

memória físico e virtual.

[0046] Um cache compartilhado 102b pode incluir uma porção de cache de código e uma porção de cache de dados. Por exemplo, quando executando o código de aplicação 103a, a porção de código do cache compartilhado 102b armazena pelo menos uma porção das instruções de processador armazenadas no código de aplicação 103 e a porção de dados do cache compartilhado 102b armazena pelo menos uma porção de estruturas de dados dos dados de tempo de execução de aplicação 103b. Frequentemente, um cache de processador é dividido em fileiras/camadas separadas (por exemplo, camada 1 (L1), camada 2 (L2) e camada 3 (L3)), com algumas fileiras (por exemplo, L3) potencialmente existindo separados do(s) processador(es) 102. Assim, o cache compartilhado 102b pode compreender uma destas camadas (L1), ou pode compreender uma pluralidade destas camadas.

[0047] Quando múltiplas camadas de cache são utilizadas, a(s) unidade(s) de processamento 102a interagem diretamente com a camada mais baixa (L1). Na maioria dos casos, os dados fluem entre as camadas (por exemplo, em uma leitura de 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 precisa executar uma escrita, os caches coordenam para assegurar que estes caches que afetaram dados que foram compartilhados entre a(s) unidade(s) de processamento 102a não mais os têm. Esta coordenação é executada utilizando um protocolo de coerência de cache (posteriormente discutido).

[0048] Os caches podem ser inclusivos, exclusivos, ou incluem comportamentos tanto inclusivo quanto exclusivo. Por exemplo, em um cache inclusivo uma camada L3 armazenaria um superconjunto dos dados na camada L2 abaixo desta, e as camadas L2 armazenam um

superconjunto das camadas L1 abaixo destas. Em caches exclusivos, as camadas podem ser desunidas - por exemplo, se dados existem em um cache L3 que um cache L1 necessita, estes podem trocar informações, tais como dados, endereço e similares.

[0049] Cada unidade de processamento 102 também inclui um microcódigo 102c, o qual compreende uma lógica de controle (isto é, instruções executáveis) que controlam a operação do processador 102, e a 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 pode estar incorporado em um armazenamento no processador, tal como ROM, EEPROM, etc.

[0050] Os registros 102d são localizações de armazenamento baseadas em hardware que são definidos com base na ISA do(s) processador(es) 102 e que são lidos de e/ou escritos pelas instruções de processador. Por exemplo, os registros 102d são comumente utilizados para armazenar valores buscados do cache compartilhado 102b para utilização por instruções, para armazenar os resultados de instruções de execução e/ou para armazenar status ou estado - tal como alguns dos efeitos colaterais de executar instruções (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 alguma modalidade, os processadores 102 podem também incluir registros de controle, os quais são utilizados para controlar diferentes aspectos de operação de processador.

[0051] Em algumas modalidades, o(s) processador(es) 102 podem incluir um ou mais armazenamentos temporários 102e. Como será

aqui posteriormente discutido, o(s) armazenamento(s) temporário(s) 102e podem ser utilizados como uma localização de armazenamento temporária 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 descarregam estes dados para o(s) arquivo(s) de rastreamento 104d em tempos apropriados, tal como quando existe largura de banda de barramento de memória disponível. Em algumas implementações, o(s) armazenamento(s) temporário(s) 102e poderiam ser parte do cache compartilhado 102b.

[0052] Como aludido acima, os processadores que possuem um cache compartilhado 102b operam o cache de acordo com um protocolo de coerência de cache ("CCP"). Especificamente, os CCPs definem como a consistência é mantida entre dados no cache compartilhados 102b e o armazenamento de dados de apoio (por exemplo, memória de sistema 103 ou outro cache) já que as várias unidades de processamento 102a leem e de escrevem dados no cache compartilhados 102b, e como assegurar que as várias unidades de processo 102a sempre leem dados válidos de uma dada localização no cache compartilhamento 102b. Os CCPs estão tipicamente relacionados com e permitem um modelo de memória definido pelo ISA do processador 102.

[0053] Exemplos de CCPs comuns incluem o protocolo MSI (isto é, Modificado, Compartilhado e Inválido), o protocolo MESI (isto é, Modificado, Exclusivo, Compartilhado e Inválido) e o protocolo MOESI (isto é, Modificado, Possuído, Exclusivo, Compartilhado e Inválido). Cada um destes protocolos define um estado para localizações individuais (por exemplo, linhas) no cache compartilhado 102b. Uma localização de cache "modificada" contém dados que foram modificados no cache compartilhado 102b, e é portanto potencialmente inconsistente

com os dados correspondentes no armazenamento de apoio (por exemplo, memória de sistema 103 ou outro cache). Quando uma localização que tem o estado "modificado" é removida do cache compartilhado 102b, CCPs comuns requerem que o cache garanta que seus dados sejam escritos de volta no armazenamento de apoio, ou que outro cache assuma esta responsabilidade. Uma localização de cache "compartilhada" contém dados que não são modificados dos dados no armazenamento de apoio, existe no estado somente de leitura, e é compartilhada pela(s) unidade(s) de processamento 102a. O cache compartilhado 102b pode remover estes dados sem escreve-los no armazenamento de apoio. Uma localização de cache "inválida" não contém dados válidos, e pode ser considerada vazio e utilizável para armazenar dados de erro de cache. Uma localização de cache "exclusiva" contém dados que coincidem com o armazenamento de apoio, e é utilizada somente por uma única unidade de processamento 102a. Esta pode ser mudada para o estado "compartilhado" em qualquer tempo (isto é, em resposta a uma solicitação de leitura) ou pode ser mudada para o estado "modificado" quando escrevendo nesta. Uma localização de cache "possuída" é compartilhada por duas ou mais unidades de processamento 102a, mas uma das unidades de processamento tem um direito exclusivo de fazer mudanças nestas. Quando este processamento faz mudanças, este diretamente ou diretamente notifica as outras unidades de processamento - já que as unidades de processamento notificadas podem precisar invalidar ou atualizar com base na implementação de CCP.

[0054] A granularidade com a qual diferentes CCPs rastreiam coerência de cache e tornam estes dados de coerência de cache disponíveis para o rastreador 104a pode variar. Por exemplo, em uma extremidade do espectro, alguns CCPs rastreiam coerência do cache por linha de cache assim como por unidade de processamento. Estes

CCPs podem, portanto, rastrear o estado de cada linha de cache conforme esta se relaciona com cada unidade de processamento. Como será demonstrado no exemplo que segue em conexão com as Figuras 6A-6D, isto significa que uma única linha de cache pode ter informações sobre seu estado já que estas se relacionam a cada unidade de processamento 102a. Outros CCPs são menos granulares, e rastreiam coerência de cache no nível da linha de cache somente (e não possuem informações por unidade por processamento). Na outra extremidade do espectro, os fabricantes de processador podem escolher rastrear a coerência de cache no nível da linha de cache somente para eficiência, já que somente um processador pode possuir uma linha exclusivamente (exclusivo, modificado, etc.) de cada vez. Como um exemplo de granularidade média, um 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 o estado da linha de cache corrente.

[0055] As modalidades utilizam o cache compartilhado do processador 102b para eficientemente gravar um rastreamento preciso de bits de execução de uma aplicação 104c e/ou do núcleo de sistema operação 104b. Estas modalidades são construídas sobre uma observação que o processador 102 (incluindo o cache compartilhado 102b) formam 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 cache compartilhado 102b, o processador 102 pode ser funcionar por si mesmo - sem qualquer entrada - como um sistema semi ou quase fechado por rajadas de tempo. Especificamente, uma ou mais das unidades de processamento 102a executam instruções da porção de código do cache compartilhado 102b, utilizando dados de tempo de exe-

cução armazenados nas porções de dados do cache compartilhado 102b e utilizando os registros 102d.

[0056] Quando uma unidade de processamento 102a precisa de algum fluxo de informações (por exemplo, porque uma instrução que este está executando, executará, ou pode executar código de acesso ou dados de tempo de execução que ainda não estão no cache compartilhado 102b), um "erro de cache" ocorre e estas informações são trazidas para o cache compartilhado 102b da memória do sistema 103. Por exemplo, se um erro no cache de dados ocorre quando uma instrução executada executa uma operação de memória em um endereço de memória dentro dos dados de tempo de execução de aplicação 103b, dados daquele endereço de memória são trazidos em uma das linhas de cache da porção de dados do cache compartilhado 102b. Similarmente, se um erro no cache de código quando uma instrução executa uma operação de memória em um código de aplicação de endereço de memória 103a armazenado na memória do sistema 103, o código daquele endereço de memória é trazido para uma das linhas de cache da porção de código do cache compartilhado 102b. A unidade de processamento 102a então continua a execução utilizando as novas informações no cache compartilhado 102b até que novas informações sejam novamente trazidas para o cache compartilhado 102b (por exemplo, devido a outra erro de cache ou uma leitura não em cache).

[0057] O inventor observou que, de modo a gravar uma representação precisa em bits de execução de uma aplicação, o rastreador 104a pode gravar dados suficientes para ser capaz de reproduzir o influxo de informações para o cache compartilhado 102b durante a execução do(s) encadeamento(s) daquela aplicação). Uma primeira proposta para fazer isto é gravar todos os dados trazidos para o cache compartilhado 102b registrando de todos os erros de cache e leituras não em cache (isto é, leituras de componentes de hardware e memória

não em cache), juntamente com um tempo durante a execução no qual cada porção de dados foi trazida para o cache compartilhado 102b (por exemplo, utilizando uma contagem de instruções executadas ou algum outro contador).

[0058] Uma segunda proposta - a qual resulta em arquivos de rastreamento significativamente menores do que a primeira proposta - é rastrear e gravar as linhas de cache que foram "consumidas" por cada unidade de processamento 102a. Como aqui utilizado, uma unidade de processamento "consumiu" uma linha de cache quando esta está ciente de seu valor presente. Isto poderia ser porque a unidade de processamento é uma que escreveu o presente valor da linha de cache, ou porque a unidade de processamento executou uma leitura na linha de cache. Esta segunda proposta envolve extensões para o cache compartilhado 102b que permitem o processador 102 identificar, para cada linha de cache, uma ou mais unidades de processamento 102a que consumiram a linha de cache.

[0059] De acordo com as modalidades aqui, uma terceira proposta é utilizar o CCP do processador para determinar um subconjunto de linhas de cache "consumidas" para gravar no(s) arquivo(s) 104d, e o qual ainda permitirá que a atividade do cache compartilhado 102b seja reproduzida. Esta terceira proposta resulta em arquivos de rastreamento significativamente menores - e assim excessos de rastreamento significativamente mais baixos - do que tanto a primeira quanto a segunda propostas.

[0060] Algumas modalidades aqui gravam fluxos de dados de rastreamento que correspondem a unidades de processamento/encadeamentos. Por exemplo, o(s) arquivo(s) de rastreamento 104 poderiam incluir um ou mais fluxos de dados de rastreamento separados para cada unidade de processamento. Nestas modalidades, os pacotes de dados em cada fluxo de dados de rastreamento podem não ter

identificação da unidade de processamento à qual os pacotes de dados se aplicam, já que estas informações são inerentes com base no próprio fluxo de dados de rastreamento. Nestas modalidades, se o sistema de computador 101 incluir múltiplos processadores 102 (isto é, dentro de diferentes soquetes de processador), o(s) arquivo(s) de rastreamento poderiam ter um ou mais diferentes fluxos de dados de rastreamento para cada unidade de processamento 102a nos diferentes processadores 102. Fluxos de dados plurais poderiam até ser utilizados para um único encadeamento. Por exemplo, algumas modalidades poderiam associar um fluxo de dados com uma unidade de processamento utilizada por um encadeamento, e associar um ou mais fluxos de dados adicionais com cada cache compartilhado utilizado pelo encadeamento.

[0061] Em outras modalidades, o(s) arquivo(s) de rastreamento 104 poderiam incluir um único fluxo de dados de rastreamento para o processador 102, e poderiam identificar em cada pacote de dados a qual unidade de processamento o pacote de dados se aplica. Nestas modalidades, se o sistema de computador 101 incluir múltiplos processadores 102, o(s) arquivo(s) de rastreamento 104 poderiam incluir um fluxo de dados de rastreamento separado para cada um dos múltiplos processadores 102. Independentemente do layout do(s) arquivo(s) de rastreamento, pacotes de dados para cada unidade de processamento 102a são geralmente gravados independentes de outras unidades de processamento, permitindo que diferentes encadeamentos que executaram nas diferentes unidades de processamento 102a sejam reproduzidos independentemente. Os arquivos de rastreamento podem, no entanto, incluir algumas informações - sejam estas expressas ou inerentes - que proveem uma ordenação parcial entre os diferentes encadeamentos.

[0062] A Figura 3 ilustra um fluxograma de um método 300 para

executar uma gravação de rastreamento gaseado em cache utilizando dados de CCP. O método 300 pode incluir atos que são executados pelo processador 102 conforme o rastreador 104a rastreia a aplicação 104c e/ou o núcleo de sistema de operação 104b. As ações feitas pelo processador 102 podem ser baseadas em lógica codificada no processador 102, lógica codificada em software (isto é, microcódigo 102c) e/ou outra aplicação de software tal como o rastreador 104a, o núcleo de sistema operação 104b, ou um hipervisor. Apesar da Figura 3 ilustrar uma sequência de atos, será apreciado que modalidades poderiam executar muitos destes atos em qualquer ordem, com alguns até sendo executados em paralelo. Como tal, a sequência de atos mostrada no método 300 é não limitante.

[0063] Como apresentado, o método 300 inclui um ato 301 de detectar uma interação entre um cache e um armazenamento de apoio. Em algumas modalidades, o ato 301 compreende detectar uma operação que causa uma interação entre uma linha de cache específica de uma pluralidade de linhas de cache e um ou mais armazenamentos de apoio. Por exemplo, enquanto executando um encadeamento da aplicação 104c ou do núcleo de sistema de operação 104b em uma das unidades de processamento 102a, a unidade de processamento pode causar uma interação entre uma linha no cache compartilhado 102b e um armazenamento de apoio (por exemplo, memória de sistema 103, ou outro cache). A detecção pode ser executada, por exemplo, pelo processador 102 com base na execução de seu microcódigo 102c.

[0064] O método 300 também inclui um ato 302 de identificar uma unidade de processamento que causou a interação. Em algumas modalidades, o ato 302 compreende identificar uma unidade de processamento específica da pluralidade de unidades de processamento que causou a operação. Por exemplo, com base em executar o microcódigo 102c, o processador 102 pode identificar qual das unidades de pro-

cessamento 102a causou a operação detectada no ato 301.

[0065] O método 300 também inclui um ato 303 de determinar se um registro é habilitado para a unidade de processamento. Em algumas modalidades, o ato 303 compreende utilizar um ou mais bits de controle de registro para determinar que o registro é habilitado para a unidade de processamento específica. Por exemplo, o processador 102 pode determinar se a unidade de processamento identificada no ato 302 tem registro habilitado, com base em um ou mais bits de controle de registro. A utilização de registro de bit(s) de controle de registro permite que o registro de diferentes unidades de processamento possa ser dinamicamente habilitado e desabilitado. Assim, utilizando bit(s) de controle de registro, o rastreador 104a pode dinamicamente controlar qual(is) encadeamento(s) estão sendo rastreados, e/ou qual(is) porção(ões) de execução de diferentes encadeamentos estão sendo rastreadas.

[0066] A forma e função específicas do(s) bit(s) de controle de registro podem variar. Em algumas modalidades, por exemplo, o(s) bit(s) de controle de registro é/são parte de um dos registros 102d, tal como um registro de controle. Nestas modalidades, um único bit de controle de registro poderia corresponder a uma unidade de processamento 102a ou a uma pluralidade de unidades de processamento 102a. Assim, um registro 102d inclui um único bit de controle de registro (por exemplo, que corresponde a todas as unidades de processamento ou uma unidade de processamento específica ou subconjunto de unidades de processamento), ou poderia potencialmente incluir uma pluralidade de bits de controle de registro (por exemplo, cada um correspondendo a uma ou mais unidades de processamento). Em outras modalidades, o(s) bit(s) de controle de registro compreendem, ou estão de outro modo associados com, um identificador de espaço de endereço (ASID) e/ou um identificador de contexto de processo (PCID) que cor-

responde a uma instrução que causou a interação entre o cache e o armazenamento de apoio. Assim, por exemplo, o método 300 poderia rastrear uma unidade de processamento somente quando esta está executando instruções associadas com um ou mais ASIDs/PCIDs específicos. Neste modo, o método 300 pode gravar somente espaço(s) de endereço especificados e/ou contexto(s) de processo específicos. Combinações são também possíveis. Por exemplo, o(s) bit(s) de controle de registro poderiam ser armazenados em um ou mais dos registros 102d, mas podem ser definidos/apagados com base em valores de ASID/PCID correntes. Independentemente da forma do(s) bit(s) de controle de registro, algumas modalidades podem ser capazes de definir/apagar o(s) bit s) de controle de registro em comutadores de contexto, permitindo que o método 300 rastreie somente encadeamentos específicos.

[0067] O método 300 também inclui um ato 304 de determinar se uma linha de cache participa em registro. Em algumas modalidades, o ato 304 compreende, com base pelo menos no registro sendo habilitado para a unidade de processamento específica, determinar se a linha de cache específica é uma participante no registro. Por exemplo, o processador 102 pode determinar se a linha de cache envolvida na operação detectada no ato 301 está envolvida em registro. Como será posteriormente discutido em maiores detalhes, existem diversos mecanismos que podem ser utilizados para a detecção, tal como a utilização de bits dentro do cache compartilhado 102b, ou a utilização de travamento de modo do cache.

[0068] O método 300 também inclui um ato 305 que utiliza um CCP para identificar que existem dados a serem registrados em um rastreamento. Por exemplo, o processador 102 pode consultar seu CCP para determinar quais transições em estado de cache ocorreram como um resultado da operação, e se estas transições garantem regis-

tro de dados. Exemplos detalhados de utilização de um CCP para identificar dados de rastreamento são dados posteriormente em conexão com as Figuras 6A-9B.

[0069] O método 300 também inclui um ato 306 de registrar dados apropriados para um rastreamento utilizando um CCP. Em algumas modalidades, o ato 306 compreende fazer com que os dados sejam registrados para o rastreamento, os dados utilizáveis para reproduzir a operação. Quando dados devem ser registrados para o(s) arquivo(s) de rastreamento, um ou mais pacotes de dados podem ser adicionados aos fluxos de dados de rastreamento apropriados - tal como um fluxo de dados de rastreamento que corresponde à unidade de processamento específica, ou um fluxo de dados de rastreamento que corresponde ao processador 102 geralmente. Se o fluxo de dados de rastreamento apropriado corresponder ao processador 102 geralmente, os um ou mais pacotes de dados podem identificar a unidade de processamento específica. Note que, se o fluxo de dados de rastreamento corresponder ao processador 102 geralmente, a ordem inerente dos pacotes de dados no próprio fluxo de dados provê algumas informações de ordenação adicionais que podem não ser disponíveis se múltiplos fluxos de dados forem utilizados.

[0070] É notado que quando o cache compartilhado 102b compreende múltiplos níveis de cache, em algumas modalidades o método 300 opera no nível de cache que interage com a memória de sistema 103, já que é este nível de cache que processa os erros de cache. Operar neste nível permite que a atividade de cache de cada unidade de processamento 102a seja representada, sem ser redundante (isto é, representando a atividade de uma unidade mais de uma vez). Assim, por exemplo, se o sistema de computador 101 incluir dois processadores 102 (isto é, dois soquetes de processador) e compreender um cache L3 "inclusivo" por soquete, assim como caches L2 "inclusivos"

abaixo do cache L3, em algumas modalidades o método 300 opera sobre os caches L3. O método 300 pode também operar em múltiplos níveis de cache. Por exemplo, se o sistema de computador 101 incluir um processador 102 (isto é, um soquete de processador) e compreender um cache L3 "exclusivo" para o soquete, assim como caches L2 "inclusivos" abaixo do cache L3, são ambos os caches L3 e L2 sobre os quais o método 300 pode operar. Exemplos adicionais de registro dentro de caches que exibem comportamentos inclusivos/exclusivos misturados estão abaixo discutidos.

[0071] Como acima mencionado em conexão com o ato 304, existem diversos mecanismos que podem ser utilizados pelo processador 102 para determinar se uma linha de cache é uma "participante em registro". Deve-se estender cada linha do cache compartilhado 102b com um ou mais "bits de contagem" adicionais que podem ser utilizados como um sinalizador, como identificadores de unidade de processamento ou como um índice de processador. A lógica para controlar estes "bits de contagem" pode fazer parte do microcódigo 102c do processador.

[0072] Para ilustrar esta modalidade, a Figura 4A ilustra um cache compartilhado exemplar 400a, similar ao cache compartimento 200 da Figura 2, que estende cada uma de suas linhas de cache 404 com um ou mais bits de contagem adicionais 401. Assim, cada linha de cache 404 inclui bit(s) de contagem 401, bits de endereço convencionais 402, e bits de valor 403.

[0073] Em algumas implementações, o(s) bit(s) de contagem de cada linha de cache 401 compreendem um único bit que funciona como um sinalizador (isto é, ligado ou desligado) utilizado pelo processador 102 para indicar se ou não a linha de cache está participando no registro de registro. Se o CCP do processador tiver granularidade suficiente (por exemplo, se o CCP rastreia o estado de coerência para ca-

da linha de cache ou como esta se refere a cada unidade de processamento, ou em referência a um índice para uma unidade de processamento que possui o estado de coerência da linha de cache), este único bit pode ser suficiente para facilitar gravar um rastreamento robusto totalmente determinístico (isto é, um que garanta total capacidade de reconstrução da execução rastreada).

[0074] Em outras implementações, o(s) bit(s) de contagem de cada linha 401 inclui uma pluralidade de bits. Pluralidades de bits poderiam ser utilizadas em diversos modos. Utilizando uma proposta, aqui referida como "bits de unidade", o(s) bit(s) de contagem de cada linha de cache 401 podem incluir um número de bits de unidade 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 de unidade podem ser utilizados pelo processador 102 para rastrear quais uma ou mais unidades de processamento específicas consumiram a linha de cache (ou se a linha de cache não foi consumida, para notar que nenhuma das unidades de processamento a consumiu). Assim, por exemplo, um cache compartilhado 102b que é compartilhado por duas unidades de processamento 102a poderia incluir dois bits de unidade para cada linha de cache. Em conexão com estes bits de unidade adicionados a cada linha de cache, as modalidades estendem o microcódigo 102c do processador para utilizar estes bits de unidade para rastrear se ou não o valor corrente na linha de cache foi registrado (isto é, no arquivo de rastreamento 104d) em nome de cada unidade de processamento, ou é de outro modo conhecido para a unidade de processamento. Se o CCP do processador tiver uma granularidade mais grossa (por exemplo, se o CCP rastreia o estado de coerência no nível da linha de cache somente), estes bits de

unidade podem prover informações adicionais para facilitar um rastreamento robusto. Por exemplo, se uma linha de cache for marcada como compartilhada ou exclusiva pelo CCP, os bits de unidade podem ser utilizados para identificar qual(is) unidade(s) de processamento compartilham a linha de cache ou qual unidade de processamento tem a exclusividade.

[0075] Utilizando outra proposta, referida aqui como "bits de índice", os bits de contagem de cada linha de cache 401 podem incluir um número de bits de índice suficientes para representar um índice para cada uma das unidades de processamento 102a do(s) processador(es) 102 do sistema de computador 101 que participam no registro, juntamente com um valor "reservado" (por exemplo, -1). Por exemplo, se o(s) processador(es) 102 do sistema de computador 101 incluírem 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, os 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 a corresponder a "inválido", para indicar que nenhum processador registrou a linha de cache correspondente - apesar desta notação poder variar, dependendo da implementação. Assim, os bits de unidade podem ser utilizados pelo processador 102 para rastrear se a linha de cache está participando em registro de rastreamento (por exemplo, um valor outro que -1), e como um índice para uma unidade de processamento específica que consumiu a

linha de cache (por exemplo, a unidade de processamento que mais recentemente a consumiu). Esta segunda proposta tem a vantagem de suportar um grande número de unidades de processamento com pouco excesso no cache compartilhado 102b, com a desvantagem de menos granularidade que a primeira proposta (isto é, somente uma unidade de processamento é identificada de cada vez). Novamente, se o CCP do processador tiver uma granularidade mais grossa (por exemplo, se o CCP rastreia um estado de coerência no nível da linha de cache somente), estes bits de índice podem prover informações adicionais para facilitar um rastreamento robusto. Por exemplo, se uma linha de cache for marcada como compartilhada ou exclusiva pelo CCP, os bits de índice podem ser utilizados para identificar pelo menos uma unidade de processamento que compartilha a linha de cache, ou qual unidade de processamento tem a exclusividade.

[0076] Outro mecanismo que pode ser utilizado pelo processador 102 para determinar se uma linha de cache é um participante em registro pode empregar os conceitos discutidos em conexão com a Figura 4A, mas sem estender cada linha de cache com bit(s) de contagem de bits de contagem adicionais 401. Ao invés, este mecanismo reserva uma ou mais das linhas de cache 404 para armazenar bits de contagem. 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 linhas de cache reservadas 406 para armazenar bits de contagem que aplicam às linhas de cache convencionais 405. Os bits da(s) linha(s) de cache reservadas 406 são alocados em diferentes grupos de bits de contagem que cada um corresponde a uma diferente das linhas de cache convencionais 405. Estes grupos de bits de contagem poderiam funcionar como um bit de sinalização, bits de unidade, ou bits de índice, dependendo da implementação.

[0077] Outro mecanismo que pode ser utilizado pelo(s) processador(es) 102 para determinar se uma linha de cache é uma participante em registro é utilizar caches associativos e bloqueio de modo. Como um cache compartilhado 102b do processador é geralmente muito menor do que a memória do sistema 103 (frequentemente por ordem de magnitude), e assim existem usualmente muito mais localizações de memória na memória de sistema 103 do que existem linhas no cache compartilhado 102b. Como tal, cada processador define um mecanismo para mapear múltiplas localizações de memória de memória de sistema para linha(s) em um cache. Os processadores geralmente empregam uma de duas técnicas gerais: mapeamento direto e mapeamento associativo. Utilizando o mapeamento direto, diferentes localizações de memória na memória de sistema 103 são mapeadas para apenas uma linha no cache, de modo que cada localização de memória pode somente ser colocada em cache em uma linha específica no cache.

[0078] Utilizando mapeamento associativo, por outro lado, diferentes localizações na memória de sistema 103 podem ser colocadas em cache para uma de múltiplas linhas no cache compartilhado 102b. A Figura 5 ilustra um exemplo 500 de mapeamentos de cache associativos. Aqui, as linhas de cache 504 de um cache 502 estão logicamente particionadas em diferentes grupos de endereços de duas linhas de cache cada, incluindo um primeiro grupo de duas linhas de cache 504a e 504b (identificado como índice 0), e um segundo grupo de endereços de duas linhas de cache 504c e 504d (identificado como índice 1). Cada linha de cache em um grupo de endereços está associada com um diferente "modo", de modo que a linha de cache 504a é identificada pelo índice 0, modo 0, a linha de cache 504b é identificada pelo índice 0, modo 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 para qualquer linha de cache dentro do grupo no índice 0 (isto é, linhas de cache 504a e 504b). Os padrões específicos dos mapeamentos apresentados são para propósitos ilustrativos e conceituais somente, e não devem ser interpretados como o único modo no qual os índices de memória podem ser mapeados para linhas de cache.

[0079] Os caches associativos são geralmente referidos como sendo caches associativos de modo N, onde N é o número de "modos" em cada grupo de endereços. Assim, o cache 500 da Figura 5 poderia ser referido como um cache associativo de modo 2. Os processadores comumente implementam caches de modo N onde N é uma potência de dois (por exemplo, 2, 4, 8, etc.), com N valores de 4 e 8 sendo comumente escolhidos (apesar das modalidades aqui não serem limitadas a nenhum valor de N específico ou subconjuntos de valores de N). Notadamente, um cache associativo de modo 1 é geralmente equivalente a um cache mapeado direto, já que cada grupo de endereços 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 de associativo, já que este compreende um único grupo de endereços que contém todas as linhas no cache. Em caches totalmente associativos qualquer localização de memória pode ser colocada em cache para qualquer linha do cache.

[0080] É notado que a Figura 5 representa uma vista simplificada de memória de sistema e caches, de modo a ilustrar 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 geralmente armazena 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 re-

presentar uma pluralidade de localizações de memória endereçáveis. Além disso, os mapeamentos podem ser entre endereços físicos reais na memória de sistema 501 e linhas no cache 502, ou pode utilizar uma camada intermediária de endereços virtuais.

[0081] Os caches associativos podem ser utilizados para determinar se uma linha de cache é uma participante em registro através da utilização de bloqueio de modo. O bloqueio de modo bloqueia ou reserva certos modos em um cache para algum propósito. Especificamente, as modalidades aqui utilizam bloqueio de modo para reservar um ou mais modos para um encadeamento que está sendo rastreado, de modo que os modos bloqueados/reservados são utilizados exclusivamente para armazenar erros de cache relativos à execução daquele encadeamento. Assim, referindo de volta à Figura 5, se "modo 0" fosse bloqueado para um encadeamento rastreado, então as linhas de cache 504a e 504c (isto é, índice 0, modo 0 e índice 1, modo 0) seriam utilizadas exclusivamente para erros de cache relativos à execução daquele encadeamento, e as linhas de cache restantes seriam utilizadas para todos os outros erros de cache. Assim, de modo a determinar se uma linha de cache específica é uma participante em registro, o processador 102 precisa somente determinar se a linha de cache faz parte de modo que está reservado para o encadeamento que está sendo rastreado.

[0082] As Figuras 6A-6D ilustram um exemplo concreto 600 de aplicação do método 300 da Figura 3, no contexto das Figuras 1, 2, 4A, 4B, e 5. A Figura 6A ilustra uma primeira tabela 600a que mostra atividade de leitura e escrita por quatro unidades de processamento 102a (isto é, P0-P3) em uma única linha no cache compartilhado 102b. A Figura 6B ilustra uma segunda tabela 600b que indica uma modalidade de estado de coerência de cache rastreado (por exemplo, como rastreado utilizando o CCP do processador) com base nestas leituras

e escritas. A Figura 6C ilustra uma terceira tabela 600c que mostra o que poderia estar armazenado em bits de contagem do cache compartilhado 102b (como descrito em conexão com as Figuras 4A e 4B), se bits de contagem forem utilizados. Apesar de somente um tipo de bits de contagem seria tipicamente utilizado (isto é, bits de unidade por linha, bits de índice por linha ou um bit de sinalização por linha), para integridade na descrição a tabela 600c mostra cada um dos bits de unidade 603, bits de índice 604, e bit de sinalização 605. Finalmente, a Figura 6D ilustra uma quarta tabela 600d que mostra tipos exemplares de dados de registro 606 que poderiam ser escritos no(s) arquivo(s) de rastreamento 104d em conexão com cada operação.

[0083] Para simplicidade em descrição, a tabela 600a apresenta operações por somente uma única unidade de processamento 102a de cada vez, mas será apreciado que os princípios aqui se aplicam em situações nas quais existem uma atividade concorrente (por exemplo, leituras concorrentes por duas ou mais unidades de processamento da mesma linha de cache). Além disso, os exemplos descritos em conexão com as Figuras 6A-6D assumem que o rastreamento está habilitado para as unidades de processamento P0-P2, e está desabilitado para a unidade de processamento P3. Por exemplo, como acima discutido, isto poderia ser controlado por um bit que corresponde a cada unidade de processamento, tal como uma pluralidade de bits de um registro de controle.

[0084] Inicialmente, para facilidade em descrição, este exemplo utilizará estados de linhas de cache simplificados que são derivados dos estados de linha de cache (isto é, Modificado, Possuído, Exclusivo, Compartilhado, e Inválido) utilizadas nos CCPs acima discutidos (isto é, MSI, MESI, e MOESI). Nesta simplificação, estes estados mapeiam para ou um estado de "leitura" (isto é, a linha de cache foi lida de) ou um estado de "escrita" (isto é, a linha de cache foi escrita em).

A Tabela 1 abaixo mostra um exemplo destes mapeamentos. Note que estes mapeamentos são utilizados como um exemplo somente, e não são limitantes. Por exemplo, podem existir CCPs e estados outros que aqueles aqui discutidos, e alguém versado na técnica reconhecerá, em vista da descrição aqui, que mapeamentos similares podem ser feitos com muitos diferentes CCPs.

Estado de Protocolo	**Estado Mapeado**
Modificado	Escrita
Possuído	Leitura
Exclusivo	Escrita
Compartilhado	Leitura
Inválido	Sem mapeamento - linha de cache considerada vazia

Tabela 1

[0085] Notadamente, as modalidades poderiam registrar dados de CCP em níveis variáveis, dependendo de quais dados estão disponíveis do processador 102 e/ou com baseados em escolhas de implementação. Por exemplo, dados de CCP poderiam ser registrados com base em estados de CCP "mapeados" (tais como aqueles mostrados na Tabela 1), com base em estados de CCP reais (por exemplo, Modificado, Possuído, Exclusivo, Compartilhado, e/ou Inválido) tornados visíveis pelo processador 102, e/ou mesmo baseado em dados de CCP "brutos" de nível mais baixo que podem não tipicamente ser tornados visíveis pelo processador 102.

[0086] Retornando às Figuras 6A-6D, a tabela 600a inclui uma primeira coluna 601 que mostra um identificador (ID), o qual é utilizado para especificar uma ordem global entre as operações. A Tabela 600a também inclui quatro colunas adicionais 602a-602d que cada uma corresponde a uma das unidades de processamento. Apesar de que, para simplicidade, este exemplo utiliza um ID global, será apreciado que na prática cada unidade de processamento normalmente ordenaria operações utilizando seus próprios conjuntos de identificadores independentes. Estes IDs poderiam compreender uma contagem de instruções

(IC) ou qualquer outro mecanismo apropriado para especificar uma ordenação entre operações, tal como uma "contagem de saltos" mais contador de programa. Note que que o exemplo utiliza a memória em modo que é consistente com os CCPs de MSI, MESI e MOESI, mas para simplicidade, este utiliza somente os estados "modificado", "compartilhado" e "e" inválido". É notado, no entanto, que alguns CCPs poderiam prover seus próprios IDs únicos e/ou monotonicamente incrementados que poderiam também ser gravados em um rastreamento (por exemplo, em cada pacote ou em pacotes ocasionais) para fortemente ordenar as entradas de rastreamento. Mesmo se o CCP não prover tal ID, o valor de um temporizador de soquete (por exemplo, TSC) ou outro ID ordenável poderia potencialmente ser utilizado.

[0087] Como mostrado na tabela 600a, na unidade de processamento P0 de identificador ID[0] executa uma leitura, o que causa um erro de cache que traz dados DATA[1] para a linha de cache. Correspondentemente, a tabela 600b mostra que o CCP do processador nota que a linha de cache é agora "compartilhada" por P0. A Tabela 600c mostra que se bits de unidade 603 forem utilizados, estes indicam que a unidade de processamento P0 consumiu (isto é, leu) a linha de cache (e que as unidades de processamento P1-P3 não), que se os bits de índice 604 forem utilizados estes indicam que P0 consumiu a linha de cache e que se um bit de sinalização 605 for utilizado este indica que alguma unidade de processamento consumiu a linha de cache. Dado este status, no ato 303 o processador 102 determinaria que o registro está habilitado para P0, e no ato 304 determinaria que a linha de cache participa no registro (isto é, utilizando bits de unidade 603, bits de índice 604, bit de sinalização 605, ou travamento de modo). Assim, no ato 306, o processador 102 utilizaria CCP para registrar dados apropriados no(s) arquivo(s) de rastreamento, se necessário. Aqui, como a linha de cache está indo de um status inválido (vazio) para um

estado lido (tabela 600a)/compartilhado (tabela 600b), os dados devem ser registrados. Como mostrado nos dados de registro 606 da tabela 600d, o processador 102 poderia notar a unidade de processamento (P0) se necessário (isto é, dependendo se os pacotes de dados estão sendo registrados para fluxos de dados separados por unidade de processamento, ou para um único fluxo de dados); o endereço de linha de cache (@); a contagem de instruções ou alguma outra contagem; e os dados (DATA[1]) que foram trazidos para a linha de cache. Apesar de que, como acima discutido, a contagem de instruções tipicamente será um valor específico de unidade de processamento, para simplicidade a tabela 600d refere-se a contagens de instruções em referência ao ID global correspondente (isto é, IC[0], neste caso).

[0088] É notado que o endereço de linha de cache (@) e os dados (por exemplo, DATA[1]) poderiam, em algumas modalidades, ser comprimidos dentro do(s) arquivo(s) de rastreamento 104d. Por exemplo, os endereços de memória podem ser comprimidos impedindo de gravar os bits "altos" de um endereço de memória referenciando (ou expressamente ou implicitamente) os bits "altos" em um endereço de memória gravado anterior. Os dados podem ser comprimidos agrupando bits de um valor de dados em uma pluralidade de grupos que compreende uma pluralidade de bits cada, e associando cada grupo com um bit "sinalizador" correspondente. Se um grupo for igual a um padrão específico (por exemplo, todos 0's, todos 1's, etc.), o bit de sinalização pode ser determinado, e este grupo de bits não precisa ser armazenado no rastreamento.

[0089] A seguir, a tabela 600a mostra que em ID[1] a unidade de processamento P1 executa uma leitura sobre a linha de cache, lendo os dados DATA[1], a Tabela 600b mostra que o CCP do processador nota que a linha de cache é agora "compartilhada" por P0 e P1. A Tabela 600c mostra que estas unidades de processamento P0 e P1 con-

sumiram a linha de cache (bits de unidade 603), que P1 consumiu a linha de cache (bits de índice 604) ou que alguma unidade de processamento consumiu a linha de cache (bit de sinalização 605). Note que que também seria correto que os bits de índice 604 ainda referenciariam P0 ao invés de P1. A tabela 600d mostra que, utilizando o CCP, o processador 102 determina que uma gravação da operação deve ser registrada. Como mostrado, o processador 102 poderia notar a unidade de processamento (P1); o endereço de linha de cache (@); a contagem de instruções (IC[1]); que a linha de cache foi de um estado de escrita (compartilhado) para um estado de escrita (compartilhado); e que P0 teve acesso anterior à linha de cache anterior, mas agora P0 e P1 têm acesso.

[0090] A seguir, a tabela 600a mostra que em ID[2] a unidade de processamento P0 executa uma escrita na linha de cache, escrevendo dados DATA[2], a Tabela 600b mostra que o CCP do processador nota que a linha de cache está agora "modificada" "por P0 e" inválida "para P1. A Tabela 600c mostra que apenas a unidade de processamento P0 consumiu (isto é, atualizou o valor da) linha de cache (bits de unidade 603), que P0 consumiu a linha de cache (bits de índice 604) ou que alguma unidade de processamento consumiu a linha de cache (bit de sinalização 605). A Tabela 600d mostra que, utilizando o CCP, o processador 102 determina que uma gravação da operação precisa ser registrada, já que a linha de cache foi escrita/modificada. Como mostrado, o processador 102 poderia notar a unidade de processamento (P0); o endereço de linha de cache (@); a contagem de instruções (IC[2]); que a linha de cache foi de um estado de leitura (compartilhado) para um estado de escrita (modificado); e que P0 e P1 tiveram acesso anterior à linha de cache anterior, mas agora somente P0 tem acesso.

[0091] Note que as informações sobre quais unidade(s) de pro-

cessamento tiveram acesso anterior a uma linha de cache poderiam ser encontradas utilizando o estado do CCP mostrado na tabela 600b. No entanto, é notado que alguns CCPs podem não manter informações suficientes para fazê-lo (por exemplo, um CCP que rastreia estado de coerência em um nível da linha de cache somente). Alternativamente, se os bits de unidade 603 forem utilizados, estas informações podem ser derivadas dos bits de unidade. Consequentemente, os dados de registro 606 mostrados na Figura 6D assumem ou um CCP robusto que mantém estas informações ou utilizam bits de unidade 603.

[0092] Se nenhum destes for utilizado (por exemplo, se o CCP não for tão robusto, e se utilizados bits de índice 604, bit de sinalização 605 ou bloqueio de modo for utilizado ao invés de bits de unidade 603), os dados de registro 606 podem ser menos completos ou maiores. Como um primeiro exemplo, se o CCP rastreia o estado de coerência no nível da linha de cache somente, e se bits de índice 604 forem utilizados, os dois podem ser utilizados para identificar que o estado de linha de cache é inválido (para todas as unidades de processamento), que está modificado (juntamente com o índice da unidade de processamento que o modificou), que é exclusivo (juntamente com o índice da unidade de processamento que o possui exclusivo) ou que é compartilhado (e todas as unidades de processamento têm acesso). Isto pode resultar em uma implementação de hardware mais simples, com a desvantagem que quando for o tempo de mudar linha de cache de compartilhada para modificada ou exclusiva, todas as unidades de processamento devem ser notificadas, ao invés de somente aquelas seriam conhecidas por um CCP mais granular para compartilhar a linha de cache. Como um segundo exemplo, bits de índice 604 poderiam ser utilizados para identificar a última unidade de processamento que acessou a linha de cache. Então, se o cache for inclusivo (isto é, tantas leituras estão ocultas atrás de acessos em níveis de cache L2

ou L1) então mesmo se as unidades de processamento estiverem lendo a mesma linha de cache, um cache L3 pode ver relativamente poucas solicitações repetidas das mesmas unidades de processamento. O registro de cada mudança de índice para uma leitura -> leitura e então tendo a leitura -> escrita, escrita -> escrita e escrita -> leitura registrar o índice também fornece os mesmos dados que a utilização de bits de unidade 603, ao custo de um rastreamento potencialmente ligeiramente maior. Como um terceiro exemplo, cada linha de cache poderia incluir um único bit de sinalização, mas o CCP poderia rastrear o estado de coerência para cada linha de cache em referência a um índice de uma unidade de processamento que possui o estado de coerência da linha de cache. Aqui, o rastreamento pode gravar mais movimentos da linha de cache do que se bits de unidade fossem utilizados ou o CCP rastreasse unidades de processamento individuais, mas o rastreamento pode ser ainda totalmente determinístico. Uma breve comparação de tamanho do arquivo de rastreamento quando tendo informações sobre cada unidade de processamento, versus somente informações sobre índice de processador, aparece daqui em diante em conexão com as Figuras 9A e 9B.

[0093] Retornando à Figura 6A, a tabela 600a mostra que em ID[3] a unidade de processamento P1 executa uma leitura da linha de cache, lendo dados DATA[2], a Tabela 600b mostra que o CCP do processador nota que a linha cache é agora "compartilhada" por P0 e P1. A Tabela 600c mostra que as unidades de processamento P0 e P1 consumiram a linha de cache (bits de unidade 603), que P1 consumiu a linha de cache (bits de índice 604), ou que alguma unidade de processamento consumiu a linha de cache (bit de sinalização 605). Note que seria também correto para os bits de índice 604 ainda referenciassem P0 ao invés P1. A Tabela 600d mostra que, utilizando o CCP, o processador 102 determina que uma gravação da operação precisa

ser registrada, já que a linha de cache passou de um estado de escrita (modificado) para um status de leitura (compartilhado). Como mostrado, o processador 102 poderia notar a unidade de processamento (P1); o endereço de linha de cache (@); a contagem de instruções (IC[3]); que a linha de cache passou de um estado de escrita (modificado) para um estado de leitura (compartilhado); e que P0 teve acesso anterior à linha de cache anterior, mas agora P0 e P1 têm acesso.

[0094] A seguir, a tabela 600a mostra que em ID[4] a unidade de processamento P0 novamente executa uma escrita na linha de cache, desta vez escrevendo dados DATA[3]. A Tabela 600b mostra que o CCP do processador nota que a linha de cache está novamente "modificada" por P0 e "inválida" para P1. A Tabela 600c mostra que somente a unidade de processamento P0 consumiu a linha de cache (bits de unidade 603), que P0 consumiu a linha de cache (bits de índice 604) ou que alguma unidade de processamento consumiu a linha de cache (bit de sinalização 605). A Tabela 600d mostra que, utilizando o CCP, o processador 102 determina que uma gravação da operação precisa ser registrada, já que a linha de cache foi escrita/modificada. Como mostrado, o processador 102 poderia notar a unidade de processamento (P0); o endereço de linha de cache (@); a contagem de instruções (IC[4]); que a linha de cache passou do estado de leitura (compartilhado) para o estado de escrita (modificado); e que P0 e P1 tiveram acesso anterior à linha de cache anterior, mas agora somente P0 tem acesso.

[0095] A seguir, a tabela 600a mostra que em ID[5] a unidade de processamento P2 executa uma leitura da linha de cache, lendo dados DATA[3]. A Tabela 600b mostra que o CCP do processador nota que a linha de cache agora é "compartilhada" por P0 e P2. A Tabela 600c mostra que as unidades de processamento P0 e P2 consumiram a linha de cache (bits de unidade 603), que P2 consumiu a linha de cache

(bits de índice 604) ou que alguma unidade de processamento consumiu a linha de cache (bit de sinalização 605). Note que também seria correto que os bits de índice 604 ainda referenciassem P0 ao invés de P2. A Tabela 600d mostra que, utilizando o CCP, o processador 102 determina que uma gravação da operação precisa ser registrada, já que a linha de cache passou de um estado de escrita (modificado) para um estado de leitura (compartilhado). Como mostrado, o processador 102 pode notar a unidade de processamento (P2); o endereço de linha de cache (@); a contagem de instrução (IC[5]); que a linha de cache passou de um estado de escrita (modificado) para um estado de leitura (compartilhado); e que P0 teve acesso anterior à linha de cache anterior, mas agora P0 e P2 têm acesso.

[0096] A seguir, a tabela 600a mostra que em ID[6] a unidade de processamento P1 executa uma leitura da linha de cache, também lendo dados DATA[3]. A Tabela 600b mostra que o CCP do processador nota que a linha de cache está agora "compartilhada" por P0, P1 e P2. A Tabela 600c mostra que as unidades de processamento P0, P1 e P2 consumiram a linha de cache (bits de unidade 603), que P1 consumiu a linha de cache (bits de índice 604) ou que alguma unidade de processamento consumiu a linha de cache (bit de sinalização 605). Note que também seria correto que os bits de índice 604 ainda referenciassem P0 ou P2 ao invés de P1. A Tabela 600d mostra que, utilizando o CCP, o processador 102 determina que uma gravação da operação deve ser registrada. Como mostrado, o processador 102 poderia notar a unidade de processamento (P1); o endereço de linha de cache (@); a contagem de instruções (IC[6]); que a linha de cache passou de um estado de leitura (compartilhado) para um estado de leitura (compartilhado); e que P0 e P2 tiveram acesso anterior à linha de cache anterior, mas agora P0, P1 e P2 têm acesso.

[0097] A seguir, a tabela 600a mostra que em ID[7] a unidade de

processamento P3 executa uma leitura da linha de cache, também lendo dados DATA [3]. A Tabela 600b mostra que o CCP do processador nota que a linha de cache está agora "compartilhada" por P0, P1, P2 e P3. A Tabela 600c mostra que nenhum dos bits de unidade 603, bits de índice 604 ou bit de sinalização 605 foram atualizados. Isto é porque o registro está desabilitado para P3, e, para propósitos de rastreamento, este assim não "consumiu" a linha de cache executando a leitura. A tabela 600d mostra que nenhum dado foi registrado. Isto é porque no ato 303 o processador 102 determinaria que o registro não está habilitado para P3.

[0098] A seguir, a tabela 600a mostra que em ID[8] a unidade de processamento P3 executa uma escrita na linha de cache, escrevendo dados DATA[4]. A Tabela 600b mostra que o CCP do processador nota que a linha de cache está agora "inválida" para P0, P1, e P2, e "modificada" por P3. A Tabela 600c mostra que os bits de unidade 603, os bits de índice 604 e bit de sinalização 605 todos refletem a linha de cache como sendo não consumida por nenhuma unidade de processamento. Isto é porque o registro está desabilitado para P3, assim, para os propósitos de rastreamento, este não "consumiu" a linha de cache quando este executou a escrita; mais ainda, a escrita invalidou o valor na linha de cache para as outras unidades de processamento. A tabela 600d mostra que nenhum dado foi registrado. Novamente, isto é porque no ato 303 o processador 102 determinaria que o registro não está habilitado para P3.

[0099] A seguir, a tabela 600a mostra que em ID[9] a unidade de processamento P0 executa uma escrita na linha de cache, escrevendo dados DATA[5]. A Tabela 600b mostra que o CCP do processador nota que a linha de cache está agora "modificada" por P0 e "inválida" para P3. A Tabela 600c mostra que nenhuma unidade de processamento consumiu a linha de cache. Isto é porque nenhuma entrada de registro

foi feita em conexão com esta operação - como refletido na tabela 600d. Nenhuma entrada de registro precisa ser feita porque os dados escritos seriam reproduzidos através de execução normal das instruções do encadeamento de P0. No entanto, qualquer entrada poderia opcionalmente ser escrita no rastreamento nesta circunstância (isto é, uma escrita a uma linha de cache que não está registrada por uma unidade de processamento com registro habilitado) para prover dados extras para um consumidor do rastreamento. Nesta circunstância, uma entrada de registro poderia ser tratada como uma leitura do valor de linha de cache, mais a escrita de DATA[5].

[00100] A seguir, a tabela 600a mostra que em ID[10] a unidade de processamento P2 executa uma leitura da linha de cache, lendo dados DATA[5]. A Tabela 600b mostra que o CCP do processador nota que a linha de cache está agora "compartilhada" por P0 e P2. A Tabela 600c mostra que a unidade de processamento P2 consumiu a linha de cache (bits de unidade 603), que P2 consumiu a linha de cache (bits de índice 604) ou que alguma unidade de processamento consumiu a linha de cache (bit de sinalização 605). A tabela 600d mostra que, utilizando o CCP, o processador 102 determina que uma gravação da operação precisa ser registrada, já vez que o valor de linha de cache não foi registrado anteriormente (isto é, não foi registrado em ID[9]). Como mostrado, o processador 102 pode notar a unidade de processo (P2); o endereço de linha de cache (@); a contagem de instruções (IC[10]); que os dados (DATA[5]) que foram trazidos para dentro da linha de cache; e que P2 tem acesso à linha de cache. Pode ser possível também registrar que P0 também tem acesso à linha de cache, dependendo de quais informações o CCP específico e os bits de contagem proveem.

[00101] A seguir, a tabela 600a mostra que em ID[11] a unidade de processamento P1 executa uma leitura da linha de cache, também

lendo dados DATA[5]. A Tabela 600b mostra que o CCP do processador nota que a linha de cache está agora "compartilhada" por P0, P1 e P2. A Tabela 600c mostra que as unidades de processamento P1 e P2 consumiram a linha de cache (bits de unidade 603), que P1 consumiu a linha de cache (bits de índice 604) ou que alguma unidade de processamento consumiu a linha de cache (bit de sinalização 605). Note que também seria correto que os bits de índice 604 ainda referenciassem P2 ao invés de P1. A Tabela 600d mostra que, utilizando o CCP, o processador 102 determina que uma gravação da operação deve ser registrada. Como mostrado, o processador 102 poderia notar a unidade de processamento (P1); o endereço de linha de cache (@); a contagem de instruções (IC[11); que a linha de cache passou de um estado de leitura (compartilhado) para um estado de leitura (compartilhado); e que P2 teve acesso anterior à linha de cache anterior, mas agora P1 e P2 têm acesso. Note que que o valor (DATA[5]) não precisa ser registrado, já que este foi registrado por P2 em ID[10].

[00102] A seguir, a tabela 600a mostra que em ID[12] a unidade de processamento P0 executa uma leitura da linha de cache, também lendo dados DATA[5]. A Tabela 600b mostra que o CCP do processador ainda nota que a linha de cache está agora "compartilhada" por P0, P1 e P2. A Tabela 600c mostra que as unidades de processamento P0, P1 e P2 consumiram a linha de cache (bits de unidade 603), que P0 consumiu a linha de cache (bits de índice 604), ou que alguma unidade de processamento consumiu a linha de cache (bit de sinalização 605). Note que também seria correto que os bits de índice 604 ainda referenciassem P1 ou P2 ao invés de P0. A Tabela 600d mostra que, utilizando o CCP, o processador 102 poderia determinar que uma gravação da operação deve ser registrada. Neste caso, o processador 102 pode notar a unidade de processamento (P0); o endereço de linha de cache (@); a contagem de instrução (IC[12]); que a linha de cache

passou de um estado de leitura (compartilhado) para um estado de leitura (compartilhado); e que P1 e P2 tiveram acesso anterior à linha de cache anterior, mas agora P0, P1 e P2 têm acesso. Nenhum valor (DATA[5]) é registrado, já que este está disponível em P2.

[00103] Alternativamente, poderia ser possível para o processador 102 referenciar P0 somente em ID[12], já que P0 já tem o valor da linha de cache (isto é, porque este escreveu este valor em ID[9]). Poderia mesmo ser possível abster-se de qualquer registro em ID[12], já que heurística poderia ser utilizada na reprodução para recuperar o valor (isto é, DATA[5]) sem informações referenciando P0 estando no rastreamento. No entanto, estas técnicas podem ser computacionalmente dispendiosas e reduzem a capacidade de sistema de detectar quando a reprodução "não viabilizou". Um exemplo de heurística é reconhecer que o acesso de memória através de unidades de processamento é geralmente fortemente ordenado (com base nos dados de CCP), de modo que a reprodução poderia utilizar o último valor através destas unidades para localização de uma dada memória.

[00104] A seguir, a tabela 600a mostra que em ID[13] a linha de cache é removida. Como um resultado, a tabela 600b mostra que as entradas de CCP estão vazias, a tabela 600c mostra que os bits de contagem não refletem nenhuma unidade de processamento como tendo consumido a linha de cache, e a tabela 600d mostra que nenhum dado está registrado.

[00105] Note que apesar de que, para integralidade, os dados de registro 606 listam todos os estados de acesso finais (isto é, quais unidades de processamento agora têm acesso à linha de cache), estas informações são potencialmente implícitas e o tamanho de arquivo de rastreamento pode ser reduzido omitindo-as. Por exemplo, em uma transição de uma escrita -> leitura, a lista de unidades de processamento que têm acesso após a leitura é sempre a unidade de proces-

samento que teve acesso anterior, mais a unidade de processamento que executou a leitura. Em uma transição de uma leitura -> escrita ou uma transição de uma escrita -> escrita, a lista de unidades de processamento que têm acesso de escrita após a escrita é sempre a unidade de processamento que executou a escrita. Em uma transição de uma leitura -> leitura, a lista de unidades de processamento que têm acesso após a leitura é sempre as unidades de processamento que têm acesso antes da transição, mais a unidade de processamento que executou a leitura.

[00106] Em geral, de modo a gerar um arquivo rastreamento totalmente determinístico, um CCP ditaria que todas as transições (isto é, escrita -> leitura, escrita -> escrita, leitura -> escrita, e leitura -> leitura) através de unidades de processamento (por exemplo, de P0 para P1) sejam registradas. No entanto, as transições com a mesma unidade de processamento (por exemplo, de P0 para P0) não precisam ser registradas. Estas não precisam ser registradas porque estas serão reproduzidas através de execução normal do encadeamento que executou naquela unidade de processamento.

[00107] Será apreciado que, utilizando dados tais como aqueles os quais são registrados no exemplo acima, e com conhecimento adicional do CCP utilizado pelo processador 102 no qual a gravação foi feita, uma ordenação total das operações que ocorreram em cada encadeamento pode ser reconstruída, e pelo menos uma ordenação parcial das operações entre as diferentes unidades de processamento pode ser reconstruída. Assim, ou um processo de indexação e/ou através de uma reprodução do arquivo de rastreamento, cada uma das operações acima pode ser reconstruída - mesmo que estas não tenham todas sido expressamente gravadas no(s) arquivo(s) de rastreamento 104d.

[00108] Em algumas modalidades, o rastreador 104a pode gravar pacotes de dados adicionais n (s) arquivo(s) de rastreamento 104d, de

modo a melhorar o registro da ordenação de operações através de unidades de processamento. Por exemplo, o rastreador 104a poderia gravar com alguns eventos informações de ordenamento tal como monotonicamente incrementando números (MINs) (ou algum outro contador/temporizador) de modo a prover uma total ordenação dos eventos que têm um MIN (ou outro contador/temporizador) através dos encadeamentos. Estes MINs poderiam ser utilizados para identificar pacotes de dados que correspondem a eventos que são definidos serem "ordenáveis" através de encadeamentos. Estes eventos poderiam ser definidos com base em um "modelo de memória de rastreamento" que define como os encadeamentos podem interagir através de memória compartilhada e sua utilização compartilhada de dados na memória. Como outro exemplo, o rastreador 104a poderia (periodicamente ou randomicamente) gravar um hash de estado de processador com base em um algoritmo determinístico definido e um conjunto de registros definido (por exemplo, contador de programas, pilha, registros de uso geral, etc.). Como ainda outro exemplo, o rastreador 104a poderia (periodicamente ou randomicamente) forçadamente registrar dados de linha de cache. Como ainda outro exemplo, o rastreador 104a poderia incluir no rastreamento pacotes de "transição" que registram um hash de todos ou uma porção (por exemplo, alguns bits) de dados que estes implicitamente carregam. Assim, quando estes dados implícitos são reconstruídos na reprodução, porções apropriadas dos dados implícitos podem ser colocadas em hash e coincididas com estes pacotes de transição para ajudar a identificar sua ordenação. Isto pode ser útil, por exemplo, se o CCP não puder rastrear os índices de processador associados as linhas de cache se as linhas de cache estiverem no estado compartilhado.

[00109] Quando o rastreador 104a grava pacotes de dados adicionais no(s) arquivo(s) de rastreamento 104d, de modo a melhorar a or-

denação, pode ser possível omitir gravar algumas das transições através de unidades de processamento. Por exemplo, pode ser possível omitir gravar algumas transições de leitura -> leitura através dos encadeamentos. Isto poderia resultar em um rastreamento não determinístico "enfraquecido" em algumas situações - já que o ordenamento de algumas leituras pode não ser capaz de ser totalmente reconstruído com base no rastreamento e no CCP - mas informações de ordenação adicionais (por exemplo, MINs, hashes de estado de processador, dados de linha de cache extras) podem ajudar a reduzir o espaço de pesquisa durante a reprodução para encontrar ordenações válidas das leituras que não "inviabilizam" a reprodução do rastreamento. Benefícios de omitir algumas das transições de leitura -> leitura através de encadeamentos incluem tamanho de rastreamento e modificações potencialmente simplificadas no processador 101 para facilitar o rastreamento.

[00110] A Figura 7A ilustra um exemplo no qual algumas transições de leitura -> leitura poderiam ser omitidas do rastreamento dependendo de como os processadores são rastreados. Similar à Figura 6A, a Figura 7A inclui uma tabela 700a com um ID global 701, e três colunas (702a-702c) que correspondem a três unidades de processamento (P0-P2). Omitir algumas transições de leitura -> leitura baseia-se em duas observações. Primeiro, as escritas precisam ser ordenadas; no entanto, todas as leituras entre duas escritas consecutivas (por exemplo, as leituras em ID[3] -ID[7]) lerão o mesmo valo de modo que a ordem entre estas leituras é irrelevante (e assim um rastreamento que omite estas transições de leitura -> leitura pode ser determinístico). Segundo, tendo uma leitura "cruzar" uma escrita quanto reproduzindo (isto é, uma leitura e uma escrita para a mesma linha de cache sendo reproduzidas na ordem incorreta) significa que os dados corretos não estão sendo utilizados para reprodução; no entanto, tendo dados (por

exemplo, MINs etc.) para evitar fazer este erro ajudará a identificar as ordenações válidas.

[00111] No exemplo mostrado na tabela 700a, a unidade de processamento P2 somente executa leituras para dados compartilhados, e estas leituras compartilhadas somente "roubam" de outras leituras (por exemplo, assumindo que ID[9] deixou a linha de cache compartilhada). Se nenhuma entrada de registro for feita de qualquer uma das transições de leitura -> leitura (isto é, ID[4]-ID[7] e ID[10]), não existirão informações no rastreamento para apropriadamente colocar as leituras de P2. Com base nas escritas pode ser concluído que P2 nunca leu o valor DATA[1] (já que a escrita em ID [2] não roubou de P2), e a falta de entradas de registro para as transições de leitura -> leitura de P2 (isto é, ID[4], ID[7] e ID[10]), tudo que se pode concluído para P2 é que houve pelo menos uma leitura por P2 entre ID[2] e ID[8]. Se, no entanto, houveram entradas de registro para ID[4] e ID[10], as leituras restantes que podem não precisar serem registradas (isto é, ID[5]-ID[7], como mostrado na Figura 7B) podem ser localizadas. Cada uma destas leituras pertence à mesma seção de interleitura como a última leitura registrada (isto é, em ID[4]). Estas leituras podem, portanto, ser localizadas com base do que as escritas roubam (e se nenhuma operação rouba de uma leitura então não existe uma escrita após esta até o próximo pacote registrado).

[00112] Em vista da tabela 700a, a Figura 7B ilustra uma tabela 700b que mostra de dados de registro - omitindo as transições de leitura -> leitura destacadas na Figura 7A, que poderiam ser gravadas se "bits de unidade" fossem utilizados. A Figura 7C ilustra uma tabela 700c que mostra dados de registro que poderiam ser gravados se "bits de índice" fossem utilizados e os índices são atualizados nas leituras.

[00113] Como brevemente acima mencionado, alguns caches incluem camadas tanto inclusivas quanto exclusivas (isto é, um cache

não totalmente inclusivo). As técnicas de registo aqui descritas são aplicáveis a estes caches, assim como caches puramente inclusivos ou exclusivos. Como um exemplo, a Figura 8A ilustra um ambiente de computação 800a que inclui dois processadores 801a/801b (por exemplo, dois processadores em soquetes correspondentes). Cada processador 801 inclui quatro unidades de processamento 802a/802b (por exemplo, unidades de processamento físicas ou lógicas). Cada processador 801 também inclui um cache de três camadas, incluindo uma camada L1 803a/803b, uma camada L2 804a/804b e uma camada L3 805a/805b. Como mostrado, cada cache inclui quatro caches L1 803 - cada um correspondendo a uma das unidades de processamento 802. Além disso, cada cache inclui dois caches L2 804 - cada um correspondendo a duas das unidades de processamento 802. Além disso, cada cache inclui um cache L3 805 para todas as unidades de processamento 802 no processador 801. As unidades de processamento e alguns dos caches estão individualmente identificados - por exemplo, as unidades de processamento 802a do processador 801a são identificadas como A0- A3, os caches L2 são identificados como A4 e A5, e o cache L3 é identificado como A6. Identificadores similares são utilizados para componentes correspondentes no processador 801b. Os asteriscos (*) associados como as unidades de processamento A0, A1, A2, B0 e B1 indicam que o registro está habilitado para estas unidades de processamento.

[00114] No ambiente de computação 800a, os caches poderiam exibir uma mistura de comportamentos inclusivos e exclusivos. Por exemplo, pode ser ineficiente para o cache L3 de A6 do processador 801a armazenar uma linha de cache quando somente a unidade de processamento A0 está utilizando-a. Ao invés, neste caso a linha de cache poderia ser armazenada no cache L1 do A0 e no cache L2 de A4, mas não no cache L1 do A1 ou no cache L2 de A5 ou caches infe-

riores. Para liberar espaço, alguns caches podem permitir que o cache L3 de A6 remova esta linha de cache nesta situação. Quando isto acontece, A1 poderia obter a linha de cache do cache L2 de A4 como seria normal em um cache inclusivo. No entanto, como a linha de cache não existe no cache L3 de A6 ou no cache L2 de A5, algumas implementações de cache podem também permitir um movimento lateral da linha de cache tal como do cache L1 do A0 para os caches L1 do A2 ou A3. Isto pode apresentar alguns desafios para o rastreamento utilizando os CCPs. Os exemplos abaixo ilustram como o rastreamento utilizando CCPs pode ser executado em tais situações.

[00115] A Figura 8B ilustra inclui uma tabela 800b que mostra operações de leitura e escrita exemplares executadas por algumas das unidades de processamento 802. O formato da tabela 800b é similar ao formato da tabela 600a. Em vista do ambiente de computação 800a e da tabela 800b, três diferentes exemplos de registro são agora fornecidos, cada um utilizando diferentes comportamentos de cache. Estes exemplos estão descritos no contexto dos seguintes princípios para registro utilizando um CCP:

(1) Geralmente, dados de registro quando um endereço (linha de cache) vai de "não registrado" para "registrado" (isto é, com base na determinação que a linha de cache participa no registro no ato 304);

(2) Geralmente, evita registrar quando uma linha de cache vai de "registrado" para "não registrado" ou "removido" (apesar do registro ser ainda válido se estes dados forem registrados). No entanto, este é válido para remoções de registro. Fazendo isto aumenta o tamanho de rastreamento, mas provê informações adicionais que podem ajudar a identificar a ordenação entre fluxos de dados de rastreamento, podem ajudar a identificar quando a reprodução de um rastreamento "não viabilizou", pode prover análise de rastreamento adicional.

Com relação à análise de rastreamento, as remoções de registro podem prover mais informações sobre como o cache foi utilizado, podem ser utilizadas para identificar características de desempenho do código executado, e podem ajudar a identificar uma janela de tempo durante a qual uma dada linha de cache armazenou um valor específico. Modalidades para registrar remoções estão posteriormente discutidas em conexão com as Figuras 10A e 10B;

(3) Movimento de registro quando a linha de cache move através de núcleos ou status de coerência de cache em um modo que provê novas informações;

(4) Quando uma unidade de processamento faz uma escrita, invalidar a linha de cache para todas as outras unidades de processamento. Se a linha de cache já não foi registrada para a unidade de processamento, o sistema pode ou não registrar a linha de cache, ou tratar a escrita como: (i) uma leitura (isto é, que registra a linha de cache e liga o rastreamento de registro), juntamente com (ii) uma escrita, assumindo que a memória na linha de cache é legível para a unidade de processamento. Pode ser legal, mas menos eficiente, que o processador desligue o registro e não registre a escrita - mas isto perde informações que precisam ser reconstruídas na reprodução e, na média, pode ser ineficiente em tamanho de rastreamento já que é mais econômico registrar uma referência do que registrar uma linha de cache inteira de dados posteriormente;

(5) É válido sobrerregistrar (por exemplo, como no princípio 2 acima) para ajudar com a reconstrução do rastreamento posteriormente. Apesar disto aumentar o tamanho de rastreamento, isto não impacta a exatidão. Por exemplo, algumas transições de leitura -> leitura poderiam ser omitidas (como acima descrito em conexão com as Figuras 7A-7C), mas qualquer transição de núcleo cruzado que começa ou termina com uma escrita deve ser explicitamente ou implicita-

mente registrada. Em outro exemplo, as modalidades podem adicionar pacotes de dados adicionais (por exemplo, que proveem informações de ordenação e/ou hashes adicionais) para o rastreamento em qualquer tempo. Em ainda outro exemplo, modalidades podem registrar quando uma escrita é primeiro comprometida para uma linha de cache após sua transição de CCP para um estado de escrita (isto é, já que uma execução especulativa pode fazer com que uma linha de cache transicione para um estado de escrita, mas não realmente comprometa nenhuma escrita a esta). O registro destas escritas pode facilitar a separação de fluxos de rastreamento por núcleo posteriormente. Em ainda outro exemplo, modalidades podem registrar saltos indiretos, ou outras informações que ajudam rapidamente reduzir o espaço de pesquisa quando separando fluxos de dados de rastreamento; e

(6) Um registro não completo (isto é, um sem todas as transições registradas) pode ainda ser utilizado para reproduzir o rastreamento. Isto pode criar um custo computacional extra no tempo de reprodução para calcular as porções faltantes.

[00116] Em um primeiro exemplo, mostrado na Figura 8C, o CCP rastreia o status de linha de cache por unidade de processamento (isto é, cada núcleo tem o seu próprio status de leitura e escrita). Neste exemplo, o cache se comporta similar a um cache inclusivo, exceto que podem existir dados que movem através de cache ou através de soquete que não estão disponíveis no tempo do registro. Para brevidade, nestes exemplos, as unidades de processamento 802 são referidas como "núcleos" e os processadores 801a e 801b são referidos como processadores A e B ou soquetes A e B. Além disso, uma notação de registro simplificada de "ID:Núcleo:From:Transition (isto é, de -> para)" é utilizada para representar tipos de dados que poderiam ser registrados. Esta notação é explicada em mais detalhes em linha. Para o primeiro exemplo, o registro poderia incluir:

[00117] Em ID[0], "0:A0:R[DATA] -> [1]"- isto é, em ID[0], registrando que o núcleo A0 lê DATA[1], pelo princípio 1 acima.

[00118] Em ID[1], "1:B0:R[DATA] -> [1]"- isto é, em ID[1], registrando que o núcleo B0 lê DATA[1], também pelo princípio 1 acima. Se o cache no processador B não estiver ciente que A0 já tem os dados registrados, então o processador B registra-os ele mesmo. Alternativamente, se o cache no processador B estiver ciente que A0 tem DATA[1] registrado, então a entrada de registro poderia incluir "1:B0:R[A0]- >R".

[00119] Em ID[2], "2:A1:R[A0]->R"- isto é, em ID[2], registrando que o núcleo A1 fez uma transição de leitura -> leitura, e que A0 teve acesso. Como o estado de linha de cache é compartilhado com o processador B, a entrada poderia ser "2:A1:R:[A0,B0]->R"- isto é, em ID[2], registrando que A1 fez uma transição de leitura -> leitura, e que A0 e B0 tiveram acesso. Como cruzar soquetes é tipicamente mais dispendioso do que registrar dentro de um soquete, a primeira entrada de registro pode ser preferida para transições de leitura -> leitura. Quando registrando para/de escritas que cruzam soquetes, no entanto, o registro também cruza soquetes.

[00120] Em ID[3], algumas modalidades não registram nada. Alternativamente, como o núcleo A2 não registrou nada ainda, e a primeira coisa que este faz é uma escrita, isto poderia ser registrado como leitura -> escrita. De qualquer modo, como uma escrita ocorreu, os outros núcleos têm seu estado de linha de cache invalidado. O custo (por exemplo, em termos de dados de rastreamento) de registrar a leitura -> escrita em ID[3] tipicamente seria menor do que o registrar dados reais em ID[4], assim pode ser benéfico registrar aqui. Neste caso, a entrada de registro poderia incluir "3:A2:R[A0,B1,B0] ->W" - isto é, o núcleo A2 fez uma transição de leitura -> escrita e núcleos A0, B1, e B0 tiveram acesso.

[00121] O que acontece em ID[4] depende no que foi registrado em ID[3]. Se nada foi registrado em ID[3], então os dados são registrados (isto é, "4:A2:R[DATA]->[2]"). Por outro lado, se um pacote foi registrado em ID[3], então não há nada para registrar.

[00122] Em ID[5] existe uma leitura que atravessa núcleos. No entanto, se o núcleo A2 ainda tiver a linha de cache como modificada (ou equivalente) então a linha de cache serve a solicitação (esta não pode ser servida da memória). Neste caso, o soquete B saberá que isto veio do soquete A e um re-registro dos dados pode ser evitado; isto poderia ser registrado como "5:B0:W[A2]->R". Se o cache obteve os dados da memória principal (este poderia ser o caso se o soquete A fosse capaz de atualizar a memória principal e compartilhar seu estado de coerência de cache para a linha), então a entrada poderia ser "5:B0:R[DATA]->2".

[00123] Em ID[6] a operação é uma leitura normal. Como a leitura em ID[2], o soquete B poderia saber sobre os dados do soquete A ou não. Se sim, a entrada de registro poderia incluir "6:B1:R[B0,A2]->R"; de outro modo poderia incluir "6:B1:R[B0]->R".

[00124] Em ID[7], se a linha de cache para B0 não foi removida não há nada para registrar. Se tiver esta foi removida, o processador B registraria os dados como vindo de outro núcleo, ou registraria os dados de linha de cache. Esta remoção de um núcleo, mas não outros no soquete, geralmente não acontece em caches totalmente inclusivos. Em um cache totalmente inclusivo, se qualquer núcleo no soquete tiver a linha de cache em seu cache L1, o então L3 tem a linha de cache, de modo que a linha de cache não pode ser removida de um núcleo, mas não de outro.

[00125] Em ID[8], como o núcleo A0 não tem nada registrado desde então e a primeira operação para registrar é uma escrita, isto é similar à operação em ID[3]. O processador A pode registrar isto como uma

leitura -> escrita; alternativamente, mas talvez menos de preferência, o processador A não poderia registrar nada. Se o pacote for registrado, seu conteúdo variaria dependendo se o soquete A poder ver o soquete B. Se este não puder, o pacote poderia incluir "8:A0:R[A2]->W", mas se este puder o pacote poderia incluir "8:A0:R[B0,B1,A2]->W".

[00126] Em ID[9] não há nada para registrar se um pacote foi registrado em ID[8] (já que é uma escrita em um cache já registrado), apesar de que o estado de linha de cache para os outros núcleos é tipicamente invalidado se já não foi.

[00127] Em ID[10], o registro depende do que foi registrado em ID[8]. Se nenhum dado foi registrado em ID[8] então precisa ser feito aqui, de modo que o pacote poderia incluir" 10:A1 :R[DATA]- >[4]". Se um pacote foi registrado em ID[8], este é um pacote de escrita-> leitura normal (por exemplo, "10:A1:W[A0]->R").

[00128] Em ID[11] a transição de leitura -> leitura é registrada. Se um pacote foi registrado em ID[8] então A0 está na lista de fonte de núcleos (por exemplo, "11:A2:R[A0,A1]->R); de outro modo, A0 não está na lista (por exemplo, "11:A2:R[A1]->R").

[00129] Em ID[12] se o soquete B puder ver o soquete A, este é um pacote de leitura -> leitura (por exemplo, "12:B0:R[A0,A1,A2]->R"). Se este não puder então é um registro de dados total (por exemplo, "12:B0:R[DATA]->[4]").

[00130] Em ID[13] os dados vêm de B0, mais soquete A se este for visível (por exemplo, "13:B1:R[A0,A1,A2,B0]->R"). A lista pode omitir o núcleo A0 se a escrita não foi registrada em ID[8].

[00131] Em ID[14] nada precisa ser registrado se um pacote foi registrado em ID[8] já foi registrado. De outro modo A0 obterá os dados A1 & A2, mais potencialmente o soquete B se puder ser visto. Como tal, o pacote poderia incluir "14:A0:R[A1,A2,B0,B1]->R".

[00132] Note que apesar deste este exemplo ter registrados os so-

quetes juntos, seria correto registrar cada soquete em isolamento, similar ao modo que os encadeamentos podem ser registrados em isolamento. Isto poderia resultar em rastreamentos maiores, mas tem a vantagem de não precisar mudar nenhum mecanismo de comunicação de soquete cruzado no processador.

[00133] Também em qualquer momento no tempo a linha de cache pode ser removida, o que significaria que os dados precisam ser reunidos de outro núcleo ou re-registrados. Por exemplo, se antes de ID[11], A0 tivesse sua linha de cache removida, então A2 obteria o valor de A1. Se tanto A1 quanto A0 fossem removidos, então o processador A pode precisar registrar o valor de linha de cache no rastreamento para A2.

[00134] Finalmente, alguns processadores podem saber que os dados vêm de outro soquete, mas não sabem qual núcleo naquele soquete. Nestes casos, o processador poderia registrar precedência (fonte) como um ID de soquete, registrar os dados este mesmo, ou registrar o ID de soquete e um hash dos dados (isto é, para ajudar ordenar acessos de soquete cruzado, mas não precisa registrar os dados inteiros para o rastreamento).

[00135] Em um segundo exemplo, mostrado na Figura 8D, o CCP utiliza índices ao invés de rastrear coerência de cache de cada núcleo separadamente. Neste ambiente, o índice poderia ser rastreado em soquete cruzado ou intra-soquete. Devido ao desempenho de comunicações de soquete cruzado versus intra-soquete, o último caso (intra-soquete) pode ser mais prático. Quando o índice é rastreado intra-soquete, o rastreamento pode precisar registrar algo quando os dados movem em soquete cruzado. Isto poderia incluir registrar o índice do outro soquete (mas isto pode não necessariamente ser único o suficiente para um rastreamento determinístico), registrar um hash de uma ou mais porções do valor de linha de cache, ou registrar um pacote no

rastreamento de soquete de envio para indicar que os dados foram enviados.

[00136] Quando rastreando índices de núcleo quando utilizando um cache não totalmente inclusivo, uma complicação surge quando um cache L1 pode ter dados que não estão no cache L3. Assim, por exemplo, assuma a seguinte sequência de eventos: (i) A0 obtém uma linha (assim os bits de índice referem a A0) em seu cache L1; (ii) A1 obtém a linha (assim os bits de índice referem a Al) em seu cache L1; (iii) o cache L3 remove a linha; (iv) A1 remove a linha de cache L1; e (v) A2 obtém a linha de cache de A0 em seu cache L1. Aqui, apesar de A2 obter a linha de cache de A0, o índice não refere-se a A0. Isto complica os mapeamentos de registro no rastreamento. Algumas soluções poderiam incluir adicionar informações extras (como acima descrito), tal como um hash de uma ou mais porções dos dados de linha de cache, periódica adicionando informações redundantes como um hash dos registros de uso geral, etc. Registrar as remoções poderia também ajudar, mas isto pode significativamente aumentar o tamanho de arquivo de rastreamento e complicar o registro (por exemplo, registrar remoções de cache L1 que não estão nos caches L2 ou L3, mas não registrar remoções de cache L1 que estão nos caches L2 ou L3).

[00137] Em algumas modalidades, quando os dados movem de um cache L3 para um cache L2 ou L1 filho, uma entrada de registro é somente feita se o índice mudar. Por exemplo, suponha que A0 tem a linha em seu cache L1 (assim os bits de índice referem a A0), então A1 obtém a linha em seu cache L1 (índice em A1), então ambos removem a linha de cache mas o L2 (ou L3) comum ainda a tem. Se o cache L2 serve A1, então não há nada para registrar. Se o cache L2 serve A0 então nenhuma entrada de registro precisa ser feita se for sabido que A0 já tem os dados; mas se não for sabido (ou não puder ser determinado) se A0 já tem os dados, então o processador pode preci-

sar registrar uma leitura -> leitura.

[00138] A Tabela 800d apresenta um registro das operações da tabela 800b, assumindo que os soquetes registram independentemente, que o rastreamento é executado por índice, que não existem remoções ocultas extras, e que todas as escritas que impactam o CCP e que acontecem quando o registro é ligado são registradas (por exemplo, uma escrita precisa ser registrada se existirem escritas consecutivas pelo mesmo núcleo e não existem acesso entre as escritas por outro núcleo ou outra entidade externa). Para o segundo exemplo, o registro poderia incluir:

[00139] Para ID[0], "0:A0:R[DATA]->[1]".

[00140] Para ID[1], "1:B0:R[DATA]->[1]" - isto é, lembre-se de que cada soquete é registrado separadamente.

[00141] Para ID[2], "2:A1 :R[A0]->R".

[00142] Para ID[3], "3:A2:R[A1]->W".

[00143] Para ID[4], nada.

[00144] Para ID[5], "5:B0:R[DATA]->[2]". Isto porque a escrita em ID[3] invalidou a linha através de todos os soquetes, e os soquetes estão sendo rastreados independentemente (como acima indicado).

[00145] Para ID[6], "6:B1:R[B0]->R".

[00146] Para ID[7], se a linha de cache para B0 não foi removida não existe nada para registrar.

[00147] Para ID[8]: "8:A0:R[A2]->W", como o bit de registro está ligado (e apesar deste núcleo não ter registrado os dados antes). Esta entrada demonstra como, com índices, existe somente o conhecimento do último proprietário no soquete.

[00148] Para ID[9], não há nada para registrar.

[00149] Para ID[10], "10:A1:W[A0]->R".

[00150] Para ID[11], "11:A2:R[A1]->R".

[00151] Para ID[12], " 12:B0:R[DATA]->[4]". Isto é porque a linha de

cache foi invalidada através de todos os soquetes em ID[8],

[00152] Para ID[13], "13:B1:R[B0]->R".

[00153] Para ID[14], "14:A0:R[A2]->R". Note que em ID[11] o índice foi atualizado para ser A2. Note também que não seria sabido que este núcleo já tinha os dados (isto é, ID[9]) já que o índice não carrega estas informações, enquanto antes do estado por processador (bits de unidade) era capaz de carregar as informações.

[00154] Em um terceiro exemplo, os caches no ambiente 800a são incapazes de acompanhar qual núcleo tem o último acesso compartilhado (leitura) para uma linha de cache. Assim, neste exemplo, o índice do último leitor não pode ser rastreado, já não existem bits para fazê-lo. Aqui, o CCP pode utilizar um valor de índice (que não mapeia para nenhum núcleo) para sinalizar uma linha compartilhada, outro valor de índice para sinalizar uma linha inválida, e o índice de processador para um estado "modificado" (por exemplo, utilizando um protocolo MSI). Neste terceiro exemplo, o registro poderia incluir registrar o índice do cache em um pacote, ao invés do índice do núcleo. Os movimentos de pai para filho não precisam ser registrados, mas poderiam ser registrados como dados extras. Se os movimentos de pai para filho não forem registrados, então a hierarquia de cache pai para filho pode precisar ser provida para o registro para ser interpretado.

[00155] Como acima mencionado, em alguns ambientes cada linha de cache de um cache poderia incluir um único bit de sinalização, mas o CCP do processador poderia rastrear estado de coerência para cada linha de cache, em referência a um índice para uma unidade de processamento que possui o estado de coerência da linha de cache. Como mencionado, isto produz rastreamentos totalmente determinísticos, mas pode resultar em maiores rastreamentos do que em casos que têm informações por unidade de processamento (por exemplo, um CCP que rastreia por unidade de processamento, em combinação com

um bit de sinalização por linha de cache). As Figuras 9A e 9B ilustram como o registro pode diferir nestas duas situações (isto é, informações de unidade de CCP mas bit de sinalização de linha de cache versus índice de CCP mais bit de sinalização de linha de cache). A Figura 9A ilustra uma tabela 900a que mostra leituras e escritas por duas unidades de processamento (P0 e P1), e a Figura 9B ilustra uma tabela 900b que compara quando as entradas de registro poderiam ser feitas nestes dois ambientes. Nestes exemplos, assuma que o bit de sinalização começa liberado, e que os bits de unidade/índice indicam que nenhuma unidade de processamento tem acesso à linha de cache.

[00156] Inicialmente, se o CCP rastrear informações de unidade e a linha de cache utilizar um bit de sinalização, o registro poderia prosseguir como segue. Como mostrado na tabela 900b, em ID[0] nada precisa ser registrado, já que é uma escrita em uma linha de cache que não foi registrada (alternativamente, o valor antes da escrita poderia ser registrado e o bit de sinalização poderia ser pulado). Neste ponto, o CCP pode notar que nem P0 nem P1 acessam a linha de cache. Em ID[1], os dados da linha de cache poderiam ser registrados para P1. O bit de sinalização poderia ser ligado e o CCP poderia notar que P1 tem acesso à linha de cache. Em ID[2], um pacote de leitura -> leitura poderia ser registrado, com P0 tomando a linha de cache de P1 (isto está registrado já que o bit de sinalização estava ligado, e o CCP é utilizado para determinar que P0 não teve acesso). O bit de sinalização já estava ligado, e o CCP nota que P0 agora também tem acesso ao estado da linha de cache. Em ID [3], nada precisa ser registrado (a linha de cache já está no registro para este núcleo). Isto é determinado porque o bit de sinalização está ligado, e o CCP indica que P1 já teve acesso à linha de cache. Em ID[4] um pacote de leitura -> escrita poderia ser registrado para P0. Isto é porque o bit de sinalização está ligado, e P0 já tem acesso à linha de cache. Como isto era uma escrita, o CCP po-

deria invalidar a linha de cache para todos os outros processadores (isto é, P0 tem acesso e P1 não). Em ID[5], um pacote de escrita -> leitura poderia ser registrado para P1. Isto é porque o bit de sinalização está ligado, mas P1 não tem os dados no rastreamento (como indicado pelo CCP). Note que os dois pacotes de referência em ID[4] e ID[5] são menores do que registrar nada em ID[4], e então precisando registrar os dados em ID[5]. O CCP nota que P1 agora tem acesso à linha de cache, além de P0.

[00157] Agora, se o CCP rastrear informações do índice somente e a linha de cache utilizar o bit de sinalização, o registro poderia prosseguir como segue. Como mostrado na tabela 900b, em ID[0] nada precisa ser registrado já que o bit de sinalização está desligado e isto é uma escrita. Como antes, isto pode ser alternativamente registrado como uma leitura mais uma escrita, se a memória for legível por P0. Em ID[1] os dados de linha de cache poderiam ser registrados para P1. O bit de sinalização poderia ser ligado, e o CCP e atualizar o índice para o ponto para P1. Em ID[2] um pacote de leitura -> leitura poderia ser registrado para P0. Isto é porque o bit de sinalização já está ligado e o índice está em P1. O CCP pode atualizar o índice para P0. Em ID[3] um pacote de leitura -> leitura poderia ser registrado para P1. Note que este caso é agora indistinguível de ID[2] já que em ambos os casos o índice no outro processador, o bit de sinalização está ligado e a linha de cache está em um estado compartilhado. O CCP pode atualizar o índice para P1. Em ID[4], um pacote de leitura -> escrita poderia ser registrado para P0. O bit de sinalização está ligado, de modo que o pacote pode registrar por referência. Isto atualiza o índice do CCP para P0. Em ID[5] um pacote de escrita -> leitura poderia ser registrada para P1. O bit de sinalização está ligado, o pacote registra por referência. A linha de cache move para um estado compartilhado, de modo que o CCP atualiza o índice para P1. Como mostrado na tabela 900b,

o caso de índice resulta em um arquivo de rastreamento maior do que o caso de unidade, mas ainda produz um rastreamento totalmente determinístico.

[00158] Algumas modalidades aqui indicaram que pode ser benéfico em termos de tamanho de arquivo de rastreamento gravar pacotes de dados que referenciam dados possuídos por outra unidade de processamento (quando possível), ao invés de gravar os dados de linha de cache posteriormente (por exemplo, ID[4] em cada um dos exemplos precedentes). Outros benefícios podem também fluir de gravar por referência. Por exemplo, na reprodução, quando existe uma série de entradas de registro que são por referência, pode ser inferido que nenhuma intervenção externa aconteceu nos dados de linha de cache. Isto é porque quando os dados de uma linha de cache total são re-registrados isto significa que ou a linha de cache foi removida ou invalidada. Assim, incluindo entradas de registro por referência, mesmo em uma situação quando uma entrada de registro pode estritamente não ser necessária, pode prover informações implícitas sobre a ausência de intervenções externas que podem ser informações úteis na reprodução ou para depuração.

[00159] Em algumas implementações, os endereços que são gravados nas entradas de rastreamento (por exemplo, as entradas "@" acima) compreendem endereços de memória físicos. Nestas implementações, o processador 102 pode gravar uma ou mais entradas do TLB 102f no(s) arquivo(s) de rastreamento 104d. Isto pode ser como parte dos fluxos de dados de rastreamento para as diferentes unidades de processamento ou como parte de um mais fluxo de dados de rastreamento adicionais. Isto permitirá reproduzir o software para mapear estes endereços físicos para endereços virtuais posteriormente.

[00160] Além disso, como os endereços físicos podem as vezes ser considerados informações "secretas" (por exemplo, quando gravando

no nível do modo de usuário), algumas modalidades gravam alguma representação dos endereços físicos reais, ao invés dos próprios endereços físicos. Esta representação poderia ser qualquer representação que unicamente mapeia seus identificadores para endereços físicos, sem revelar o endereço físico. Um exemplo poderia ser um hash de cada endereço físico. Quando estas representações são utilizadas, e entradas do TLB 102f são gravadas no(s) arquivo(s) de rastreamento 104d, o processador 102 grava um mapeamento entre estas representações e endereços virtuais, ao invés de endereços físicos para endereços virtuais.

[00161] Como mencionado, o processador 102 pode incluir um ou mais armazenamentos temporários 102e. Estes armazenamentos temporários podem ser utilizados como uma localização de armazenamento temporária para entradas de arquivo de rastreamento, antes de realmente escrever estas entradas no(s) arquivo(s) de rastreamento 104d. Assim, quando o ato 305 faz com que dados sejam registrados para o rastreamento, o ato 305 poderia compreender escrever os dados no(s) armazenamento(s) temporário(s) 102e. Em algumas modalidades, o processador 102 emprega técnicas de registro deferidas de modo a reduzir o impacto de escrever dados de rastreamento no processador 102 e no barramento de memória. Nestas modalidades, o processador 102 pode armazenar dados de rastreamento no(s) armazenamento(s) temporário(s)102e e deferir escrever o(s) arquivo(s) de rastreamento 102f até que exista largura de banda disponível no barramento de memória ou o(s) armazenamento(s) temporário(s)102e está/estão cheios.

[00162] Como foi também mencionado, algumas modalidades podem registrar remoções de cache. As Figuras 10A e 10B ilustram algumas modalidades de como a remoção de cache pode ser registrada em um modo eficiente (isto é, em termos de tamanho de arquivo de

rastreamento) alavancando propriedades de caches associativos. Inicialmente a Figura 10A ilustra um exemplo de 1000 de diferentes partes de um endereço de memória, e sua relação com caches associativos. Como mostrado, os endereços de memória incluem uma primeira pluralidade de bits 1001 que são os bits baixos do endereço, e que são tipicamente zero. A primeira pluralidade de bits 1001 é zero porque os endereços de memória são tipicamente alinhados com um tamanho do endereço de memória (por exemplo, 32 bits, 64 bits, etc.). Assim, o número da primeira pluralidade de bits 1001 é dependente do tamanho do endereço de memória. Por exemplo, se um endereço de memória for 32 bits (isto é, 2^5 bits), então a primeira pluralidade de bits 1001 compreende cinco bits (de modo que os endereços de memória sejam múltiplos de 32), se um endereço de memória for 64 bits (isto é, 2^6), então a primeira pluralidade de bits 1001 compreende seis bits (de modo que os endereços de memória sejam múltiplos de 64), etc. Os endereços de memória também incluem uma segunda pluralidade de bits 1002 que pode ser utilizada por um processador 102 para determinar um grupo de endereços específico em um cache associativo no qual os dados do endereço de memória devem ser armazenados. No exemplo 1000 da Figura 10A, por exemplo, a segunda pluralidade de bits 1002 compreende três bits, os quais corresponderia a um cache associativo que tem oito grupos de endereços. O número da segunda pluralidade de bits 1002 é portanto dependente da geometria específica do cache associativo. Os endereços de memória também incluem uma terceira pluralidade de bits 1003 que compreende os bits altos restantes do endereço de memória.

[00163] No contexto da Figura 10A, a Figura 10B ilustra um exemplo 1004 de registrar erros de cache e remoções de cache em um cache associativo. Inicialmente, o exemplo 1004 mostra três endereços de memória 1005 (isto é, endereço 1024), 1006 (isto é, endereço @

2112) e 1007 (isto é, endereço @ 2048). A Figura 10B também ilustra um cache associativo 1010 que tem oito grupos cada um compreendendo quatro modos. A identidade binária destes grupos e modos está mostrada nas colunas 1008 (grupos) e 1009 (modos), juntamente com uma representação decimal correspondente entre parênteses. Assim, por exemplo, a linha de cache (0, 0) - isto é, grupo 0, modo 0 - no cache 1010 está mostrada em binário como grupo '000' (coluna 1008) e modo '00' (coluna 1009); a linha de cache (0, 1) - grupo 0, modo 1 - no cache 1010 está mostrada em binário como grupo '000' (coluna 1008) e modo '01' (coluna 1009); e assim por diante até a linha de cache (8, 3) - 30 isto é, o grupo 8, o modo 3 - no cache 1010 está mostrada em binário como grupo '111' (coluna 1008) e modo '11' (coluna 1009).

[00164] Agora, suponha que exista um primeiro erro de cache no endereço 1005 (isto é, @ 1024). Aqui, como sua segunda pluralidade de bits 1002 é '000' o processador 102 pode determinar que deve armazenar os dados que correspondem ao endereço 1005 no grupo 0 do cache 1010. O modo específico no grupo 0 é tipicamente escolhido pela lógica específica de processador. Para propósitos do exemplo 1004, no entanto, suponha que os dados estão armazenados no modo 0 (como mostrado pela seta 1011a). Em conexão com este erro de cache, os dados registrados gravados pelo rastreador 104a poderiam incluir o endereço de memória (isto é, @ 1024) e o modo (isto é, modo 0) no qual os dados foram armazenados. Note que qualquer número de técnicas de compressão poderia ser utilizado para reduzir o número de bits necessários para armazenar o endereço de memória no rastreamento. O grupo (isto é, grupo 0) não precisa ser registrado porque este pode ser obtido da segunda pluralidade de bits 1002 do endereço de memória.

[00165] A seguir, suponha que exista um segundo erro de cache no endereço 1006 (isto é, @ 2112). Desta vez, a segunda pluralidade de

bits 1002 é '010' de modo que o processador 102 pode determinar que deve armazenar os dados que correspondem ao endereço 1006 no grupo 2 do cache 1010. Novamente, o modo específico no grupo 2 é tipicamente escolhido pela lógica específica de processador. Para propósitos do exemplo 1004, no entanto, suponha que os dados estão armazenados no modo 0 (como mostrado pela seta 1011b). Em conexão com este erro de cache, os dados de registro gravados pelo rastreador 104a poderiam incluir o endereço de memória (isto é, @ 2112) e o modo (isto é, modo 0) no qual os dados foram armazenados. Novamente, o grupo (isto é, grupo 2) não precisa ser registrado porque este pode ser obtido da segunda pluralidade de bits 1002 do endereço de memória.

[00166] Agora, suponha que exista um terceiro erro de cache no endereço 1007 (isto é, @ 2048). A segunda pluralidade de bits 1002 é novamente '000', de modo que o processador 102 pode determinar que deve armazenar os dados que correspondem ao endereço 1007 no grupo 0 do cache 1010. O modo específico é novamente escolhido por lógica específica de processador, mas suponha que o processador escolha o modo 0 (como mostrado pela seta 1011c). Em conexão com este erro de cache, os dados de registro gravados pelo rastreador 104a poderiam incluir o endereço de memória (isto é, @ 2048) e o modo (isto é, modo 0) no qual os dados foram armazenados. Novamente, o grupo (isto é, grupo 0) não precisa ser registrado porque este pode ser obtido da segunda pluralidade de bits 1002 do endereço de memória.

[00167] Como esta linha de cache (0,0) correntemente corresponde ao endereço 1005, este terceiro erro de cache no endereço 1007 faz com que o endereço 1005 seja removido do cache 1010. No entanto, as modalidades podem abster-se de gravar qualquer dado de rastreamento que documente esta remoção. Isto é porque a remoção pode

ser inferida de dados já no rastreamento - isto é, o primeiro erro de cache no endereço 1005 no modo 0, juntamente com o segundo erro de cache no endereço 1007 no modo 0. Apesar do grupo (isto é, grupo 0) poder não ser expressamente registrado no rastreamento, este pode ser inferido a destes endereços. Como tal, a reprodução destes dados de rastreamento pode reproduzir a remoção.

[00168] Algumas remoções resultam de eventos outros que um erro de cache. Por exemplo, um CCP pode fazer com que uma remoção ocorra de modo a manter uma consistência entre diferentes caches. Suponha, por exemplo, que o endereço 1006 é removido da linha de cache (2,0) do cache 1010 devido a um evento de CCP. Aqui, a remoção pode ser expressamente registrada gravando o grupo (isto é, '010') e o modo (isto é, '00') da remoção. Notadamente, o endereço que foi removido não precisa ser registrado, já que este já foi capturado quando registrando o segundo erro de cache que trouxe o endereço 1006 para dentro da linha de cache (2,0). Consequentemente, neste exemplo, a remoção pode ser totalmente capturada no(s) arquivo(s) de rastreamento 104d com meros cinco bits de dados de registro (antes de qualquer forma de compressão).

[00169] Algumas modalidades são também capazes de seguramente rastrear atividade de uma unidade de processamento, mesmo quando um encadeamento que executa nesta unidade de processamento interage com um enclave seguro. Como será apreciado por aqueles versados na técnica, os enclaves são características de segurança baseadas em hardware que podem proteger informações confidenciais (por exemplo, chaves criptográficas, credenciais, dados biométricos, etc.) de potencialmente mesmo o software de nível mais baixo que executa em um processador 102. Assim, além de proteger informações confidenciais de processos de modo de usuário, os enclaves podem até proteger informações confidenciais de núcleos e/ou hi-

pervisores. Em muitas implementações, os enclaves aparecem para um processo em execução como porção(ões) criptografada(s) de memória mapeadas no espaço de endereço do processo. Isto pode ser implementado, por exemplo, utilizando diferentes tabelas de páginas de memória para o processo em execução e o enclave. Quando um processo interage com um enclave, o processo pode ler da/escrever na sua própria memória mapeada e o enclave pode ler da/escrever na sua própria memória mapeada e/ou a memória mapeada do processo.

[00170] As primeiras modalidades de rastreamento ciente de enclave rastreiam um processo de execução, enquanto abstendo-se de rastrear um enclave com o qual o processo interage, enquanto ainda permitindo que o processo rastreado seja totalmente reproduzido. Nestas modalidades, as leituras de memória pelo processo em execução para seu espaço de endereço são rastreadas/registradas utilizando um ou mecanismos já aqui descritos. Quando existe uma troca de contexto para o enclave, no entanto, as modalidades podem rastrear quaisquer localizações de memória que foram previamente lidas pelo processo rastreado, e que são escritas pelo enclave durante a sua execução. Quando o processo rastreado novamente executa após a troca para o enclave, esta(s) localização(ões) de memória são tratados como não tendo sido registradas pelo processo rastreado. Deste modo, se o processo rastreado novamente lê desta(s) localização(ões) de memória (potencialmente lendo dados que foram colocados nesta(s) localização(ões) pelo enclave), estas leituras são registradas no rastreamento. Efetivamente, isto significa que quaisquer efeitos colaterais de execução do enclave que são visíveis para o processo rastreado são capturados no rastreamento, sem precisar rastrear a execução do enclave. Neste modo, o processo rastreado pode ser posteriormente reproduzido utilizando estes efeitos colaterais, sem realmente precisar (ou mesmo ser capaz de) reproduzir a execução do enclave. Existem

diversos mecanismos (anteriormente descritos) que podem ser utilizados para rastrear localização(ões) de memória que foram anteriormente lidas pelo processo rastreado e que são escritas pelo enclave durante a sua execução, tais como bits de contagem (por exemplo, bits de sinalização, bits de unidade, bits de índice), travamento de percurso, utilização de dados de CCP, etc.

[00171] As segundas modalidades de rastreamento ciente de enclave rastreiam o processo de execução (por exemplo, com base em acessos, tais como leituras, a seu próprio espaço de endereço), enquanto também rastreando o enclave (por exemplo, com base em acessos a seu próprio espaço de endereço e/ou acessos ao espaço de endereço do processo rastreado). Estas modalidades poderiam ser implementadas quando existe um nível de confiança necessário entre o núcleo/hipervisor e o enclave. Nestas modalidades, os dados de rastreamento relativos à execução do enclave poderiam ser registrados em um fluxo de dados de rastreamento separado e/ou criptografados, de modo que qualquer entidade que executa uma reprodução seja incapaz de reproduzir o enclave sem acesso ao fluxo de dados de rastreamento separado do enclave e/ou chave(s) criptográfica(s) que podem ser utilizadas para decifrar os dados de rastreamento relativos à execução do enclave.

[00172] As terceiras modalidades de rastreamento ciente de enclave combinam a primeira e segunda modalidades. Assim, estas terceiras modalidades podem gravar um rastreamento de um processo em execução que inclui os efeitos colaterais desta utilização do processo de um enclave (isto é, a primeira modalidade), juntamente com um rastreamento do próprio enclave (isto é, a segunda modalidade). Isto permite que a execução do processo rastreado seja reproduzida por um usuário que não possui um nível de privilégio necessário e/ou chave(s) criptográfica(s), enquanto permite um usuário que tem o nível de

privilégio necessário e/ou chave(s) criptográfica(s) também reproduzir a execução do próprio enclave.

[00173] Cada uma destas modalidades de rastreamento de enclave é aplicável além de enclaves, e a qualquer situação na qual uma entidade rastreada interage com outra entidade cuja execução precisa ser protegida durante o rastreamento (referida agora como uma entidade protegida). Por exemplo, qualquer uma destas modalidades poderia ser utilizada quando rastreando um processo de modo usuário que interage com um processo de modo de núcleo - aqui, o processo de modo núcleo poderia ser tratado similarmente a um enclave. Em outro exemplo, qualquer destas modalidades poderia ser utilizado quando rastreamento um processo de modo núcleo que interage com um hipervisor - aqui, o hipervisor poderia ser tratado similarmente a um enclave.

[00174] Podem existir ambientes nos quais não é prático (por exemplo, devido a considerações de desempenho ou segurança), não possível (por exemplo, devido à falta de suporte de hardware), ou não desejável rastrear qual(is) localização(ões) de memória que foram anteriormente lidas por um processo rastreado e escritas por uma entidade protegida durante sua execução. Isto pode impedir a utilização das modalidades de rastreamento de enclave acima descritas. No entanto, existem também técnicas para rastrear nestas situações.

[00175] Uma primeira técnica é tratar o cache de processador como tendo sido invalidado após a troca de contexto da entidade protegida. Tratar o cache de processador como tendo sido invalidado faz que com as leituras pela entidade rastreada após o retorno da entidade protegida para causar erros de cache - os quais podem ser registrados. Estes erros de cache incluirão quaisquer valores que foram modificados no espaço de endereço da entidade rastreada pela entidade protegida, e que foram subsequentemente lidos pela entidade rastrea-

da. Apesar desta técnica poder gerar mais dados de rastreamento do que as três modalidades acima descritas, esta realmente captura os efeitos de execução da entidade protegida em que a entidade rastreada se baseou. Em algumas modalidades, esta primeira técnica poderia também gravar um ou mais quadros chave (por exemplo, incluindo um instantâneo de registros do processador) quando de um retorno para uma entidade rastreada de uma entidade protegida. O(s) quadro(s) chave permitem que a reprodução da entidade rastreada seja começada após o retorno da entidade protegida, apesar de existir uma falta em continuidade em dados de rastreamento (isto é, durante a execução da entidade protegida).

[00176] Uma segunda técnica é registrar os erros de cache relativos a leituras por uma entidade protegida do espaço de endereço da entidade rastreada, assim como escritas executadas pela entidade protegida no espaço de endereço da entidade rastreada. Isto permite que uma reprodução do rastreamento reproduza as escritas da entidade protegida sem precisar ter acesso às instruções da entidade protegida que as produziu. Isto também fornece acesso de reprodução para os dados (no espaço de endereço da entidade rastreada) que a entidade protegida leu e os quais a entidade rastreada acessou posteriormente. Propostas híbridas são possíveis (se informações de contabilidade suficientes, tais como dados de CCP, forem disponíveis) que poderiam registrar as escritas da entidade protegida (no espaço de endereço da entidade rastreada), mas não suas leituras - se estas leituras fossem registradas mais tarde devido ao tratamento do cache como invalidado.

[00177] 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 in-

venção é, portanto, indicado pelas reivindicações anexas ao invés de pela descrição acima. Todas as mudanças as quais vêm dentro do significado e faixa de equivalência das reivindicações devem ser abrangidas dentro de seu escopo.

REIVINDICAÇÕES

1. Dispositivo de computação, caracterizado pelo fato de compreender:

uma pluralidade de unidades de processamento;

uma memória de cache que compreende uma pluralidade de linhas de cache que são utilizadas para colocar em cache dados de um ou mais armazenamentos de apoio e que são compartilhadas pela pluralidade de unidades de processamento, em que consistência entre dados na pluralidade de linhas de cache e os um ou mais armazenamentos de apoio é gerenciada de acordo com um protocolo de coerência de cache (CCP); e

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

determinar que pelo menos as seguintes condições tenham sido atendidas:

(i) uma operação causou uma interação entre uma linha de cache específica da pluralidade de linhas de cache e os um ou mais armazenamentos de apoio;

(ii) o registro é habilitado para uma unidade de processamento específica da pluralidade de unidades de processamento que causou a operação;

(iii) a linha de cache específica é uma participante no registro; e

(iv) o CCP indica que existem dados a serem registrados em um rastreamento com base na operação; e

com base pelo menos na determinação que as condições foram atendidas, fazer com que os dados sejam registrados no rastreamento, os dados utilizáveis para reproduzir a operação.

2. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que a lógica de controle armazenada

também configura o dispositivo de computação para atualizar um ou mais bits de contagem associados com a linha de cache específica para indicar se a linha de cache específica permanece uma participante em registro após a operação.

3. Dispositivo de computação de acordo com a reivindicação 2, caracterizado pelo fato de que os um ou mais bits de contagem associados com a linha de cache específica compreendem um de (i) um único bit, (ii) uma pluralidade de bits que cada um corresponde a uma da pluralidade de unidades de processamento, ou (iii) uma pluralidade de bits que armazena um valor de índice de processador.

4. Dispositivo de computação de acordo com a reivindicação 2, caracterizado pelo fato de que os um ou mais bits de contagem associados com a linha de cache específica são armazenados em uma ou mais linhas de cache reservadas que são separadas das linhas de cache que são utilizadas para colocar em cache dados de um ou mais armazenamentos de apoio.

5. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que fazer com que os dados a serem registrados no rastreamento compreende escrever os dados em um armazenamento temporário, e em que descarregar dados do armazenamento temporário para o arquivo de rastreamento é deferido com base em atividade de barramento de memória.

6. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que a lógica de controle armazenada também configura o dispositivo de computação para registrar pelo menos uma remoção de cache por referência a um grupo e um modo em um cache associativo.

7. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que os dados registrados compreendem transições entre diferentes estados de CCP.

8. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que os dados registrados compreendem pelo menos uma de: uma transição de um estado de escrita para um estado de leitura, uma transição de um estado de escrita para um estado de escrita, ou uma transição de um estado de leitura para um estado de escrita.

9. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que utilizar o CCP para identificar que existem dados a serem registrados a um rastreamento compreende identificar que uma transição de um estado de leitura para um estado de leitura não precisa ser registrada no rastreamento.

10. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que os dados para cada unidade de processamento são registrados em pelo menos um fluxo de dados separado.

11. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que os dados para duas ou mais unidades de processamento são registrados no mesmo fluxo de dados, mas identificados com um identificador de unidade de processamento.

12. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que os dados a serem registrados no rastreamento compreendem informações de ordenação.

13. Dispositivo de computação de acordo com a reivindicação 1, caracterizado pelo fato de que os dados a serem registrados compreendem dados escritos na linha de cache específica por um enclave, e em que fazer com que os dados sejam registrados no rastreamento compreende:

quando a operação que causou a interação entre a linha de cache específica e os um ou mais armazenamentos de apoio corresponde a um encadeamento que interage com o enclave, fazer com

que os dados a serem registrados em fluxo de dados de rastreamento que corresponde ao encadeamento; ou

quando a operação que causou a interação entre a linha de cache específica e os um ou mais armazenamentos de apoio corresponde ao enclave, fazer com que os dados sejam registrados para serem separados do fluxo de dados de rastreamento que corresponde ao encadeamento.

14. Método implementado em um ambiente de computação, caracterizado por incluir uma pluralidade de unidades de processamento e uma memória de cache que compreende uma pluralidade de linhas de cache que são utilizadas para colocar em cache dados de um ou mais armazenamentos de apoio e que são compartilhadas pela pluralidade de unidades de processamento, em que consistência entre dados na pluralidade de linhas de cache e os um ou mais armazenamentos de apoio é gerenciada de acordo com um protocolo de coerência de cache, o método para executar uma gravação de rastreamento baseada em cache utilizando dados de protocolo de coerência de cache (CCP), o método compreendendo:

determinar que pelo menos as seguintes condições tenham sido atendidas:

(i) uma operação causou uma interação entre uma linha de cache específica da pluralidade de linhas de cache e os um ou mais armazenamentos de apoio;

(ii) o registro é habilitado para uma unidade de processamento específica da pluralidade de unidades de processamento que causou a operação;

(iii) a linha de cache específica é uma participante no registro; e

(iv) o CCP indica que existem dados a serem registrados em um rastreamento com base na operação; e

com base pelo menos na determinação que as condições foram atendidas, fazer com que os dados sejam registrados no rastreamento, os dados utilizáveis para reproduzir a operação.

15. Produto programa de computador para utilização em um dispositivo de computação, caracterizado pelo fato de compreender uma pluralidade de unidades de processamento e uma memória de cache que compreende uma pluralidade de linhas de cache que são utilizadas para colocar em cache dados de um ou mais armazenamentos de apoio e que são compartilhadas pela pluralidade de unidades de processamento, em que a consistência entre dados na pluralidade de linhas de cache e os um ou mais armazenamentos de apoio é gerenciada de acordo com um protocolo de coerência de cache (CCP), o produto do programa de computador compreendendo um meio legível por computador que tem armazenado neste instruções executáveis por computador que são executáveis por uma ou mais unidades de processamento para fazer com que o dispositivo de computação execute pelo menos o seguinte:

determinar que pelo menos as seguintes condições tenham sido atendidas:

(i) uma operação causou uma interação entre uma linha de cache específica da pluralidade de linhas de cache e os um ou mais armazenamentos de apoio;

(ii) o registro é habilitado para uma unidade de processamento específica da pluralidade de unidades de processamento que causou a operação;

(iii) a linha de cache específica é uma participante no registro; e

(iv) o CCP indica que existem dados a serem registrados em um rastreamento com base na operação; e

com base pelo menos na determinação que as condições

foram atendidas, fazer com que os dados sejam registrados no rastreamento, os dados utilizáveis para reproduzir a operação.

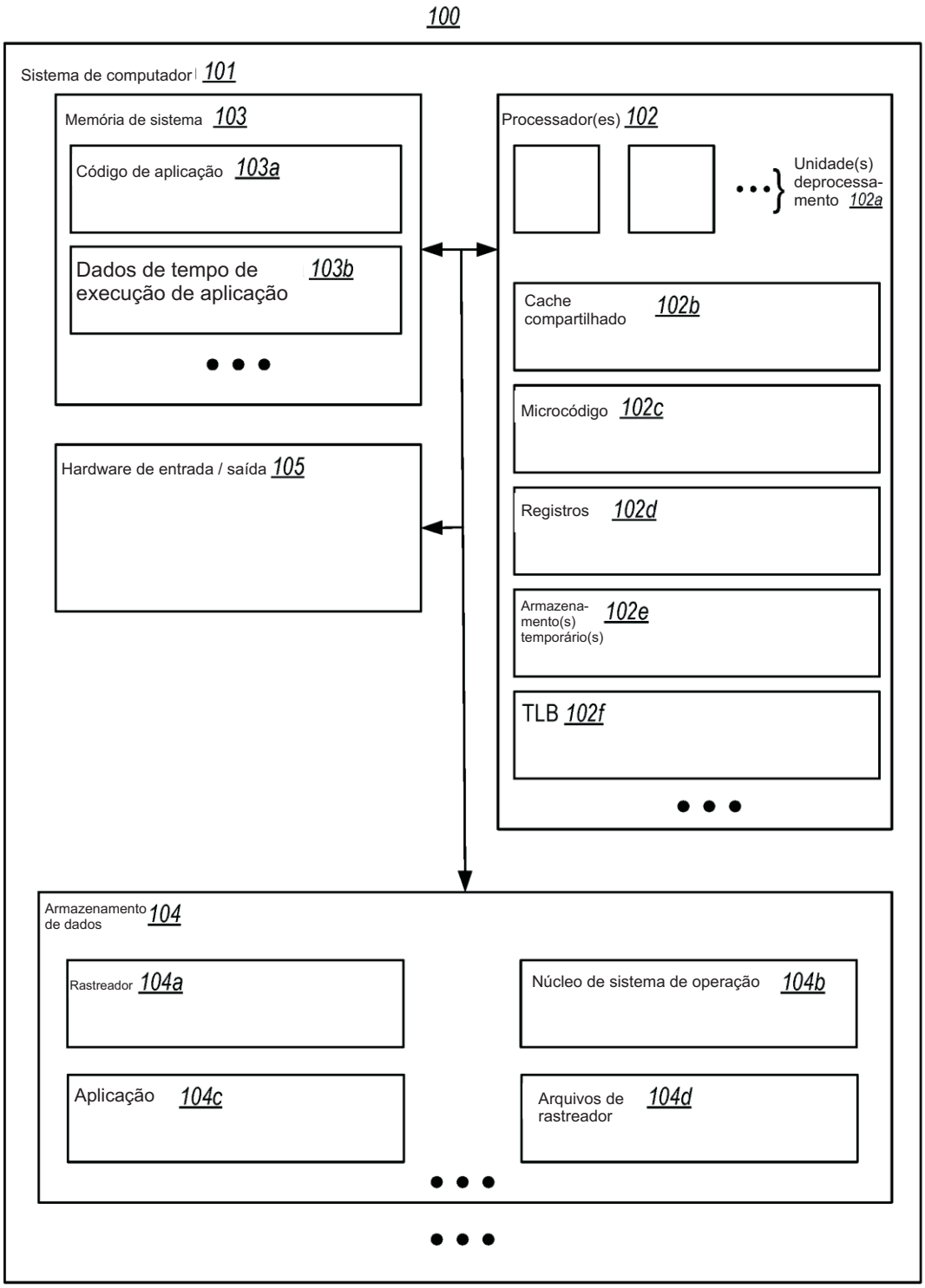

100
Sistema de computador 101
Memória de sistema 103
Código de aplicação 103a
Dados de tempo de execução de aplicação 103b
Processador(es) 102
Unidade(s) deprocessa-mento 102a
Cache compartilhado 102b
Microcódigo 102c
Registros 102d
Armazena-mento(s) temporário(s) 102e
TLB 102f
Hardware de entrada / saída 105
Armazenamento de dados 104
Rastreador 104a
Núcleo de sistema de operação 104b
Aplicação 104c
Arquivos de rastreador 104d

FIG. 1

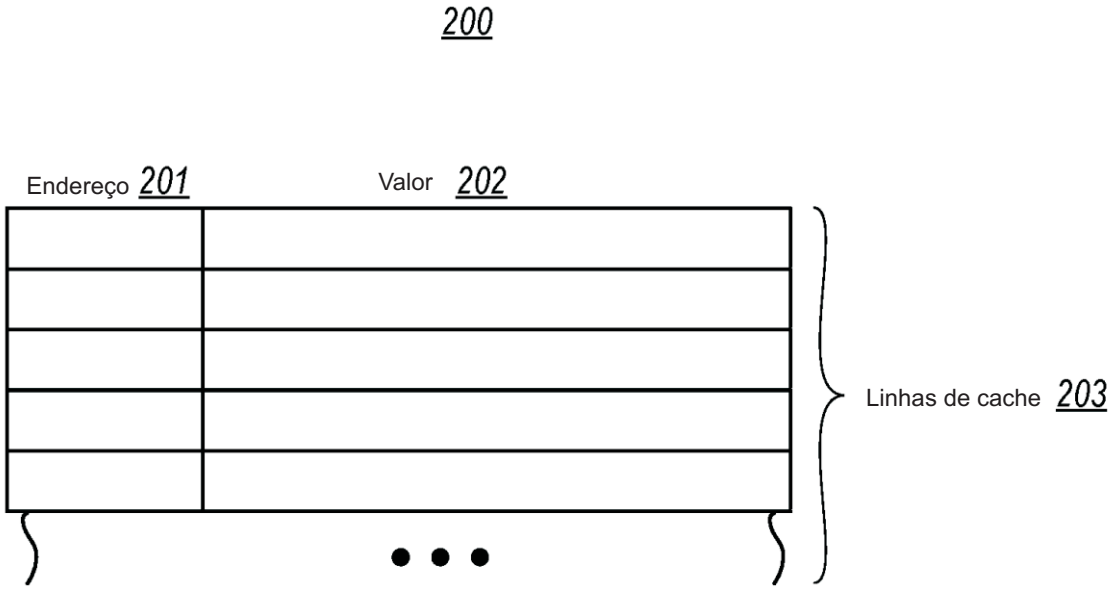


FIG. 2

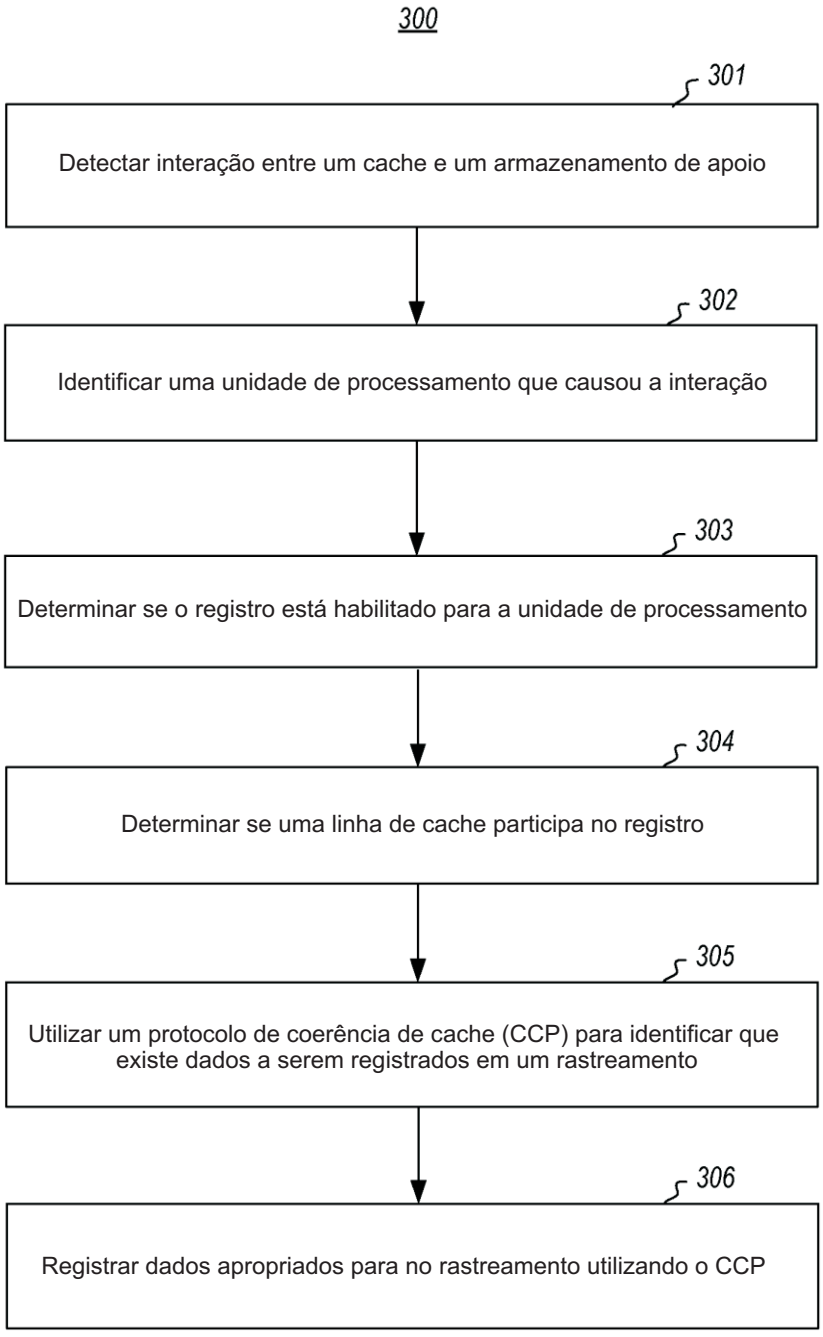

300
301
Detectar interação entre um cache e um armazenamento de apoio
302
Identificar uma unidade de processamento que causou a interação
303
Determinar se o registro está habilitado para a unidade de processamento
304
Determinar se uma linha de cache participa no registro
305
Utilizar um protocolo de coerência de cache (CCP) para identificar que existe dados a serem registrados em um rastreamento
306
Registrar dados apropriados para no rastreamento utilizando o CCP

FIG. 3

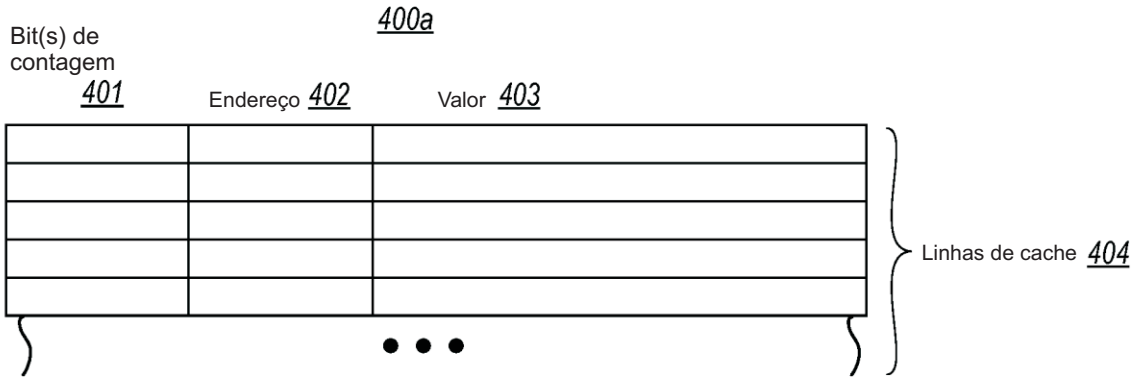

400a
Bit(s) de contagem
401
Endereço 402
Valor 403
Linhas de cache 404

FIG. 4A

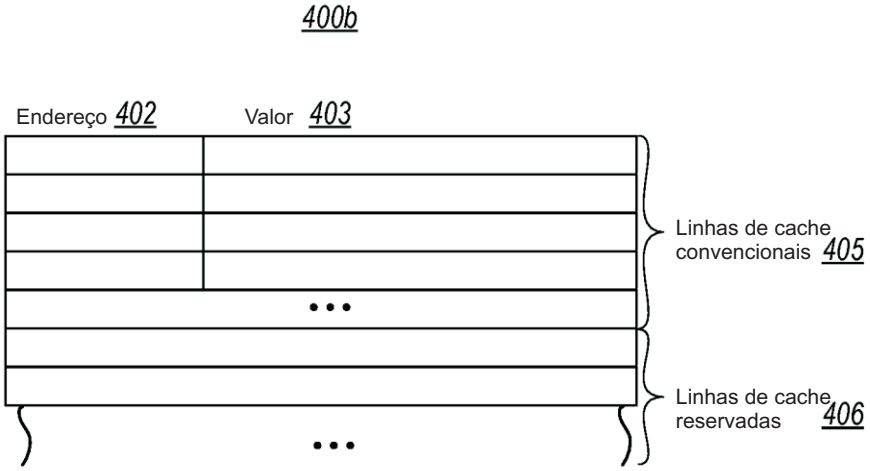

400b
Endereço 402
Valor 403
Linhas de cache convencionais 405
Linhas de cache reservadas 406

FIG. 4B

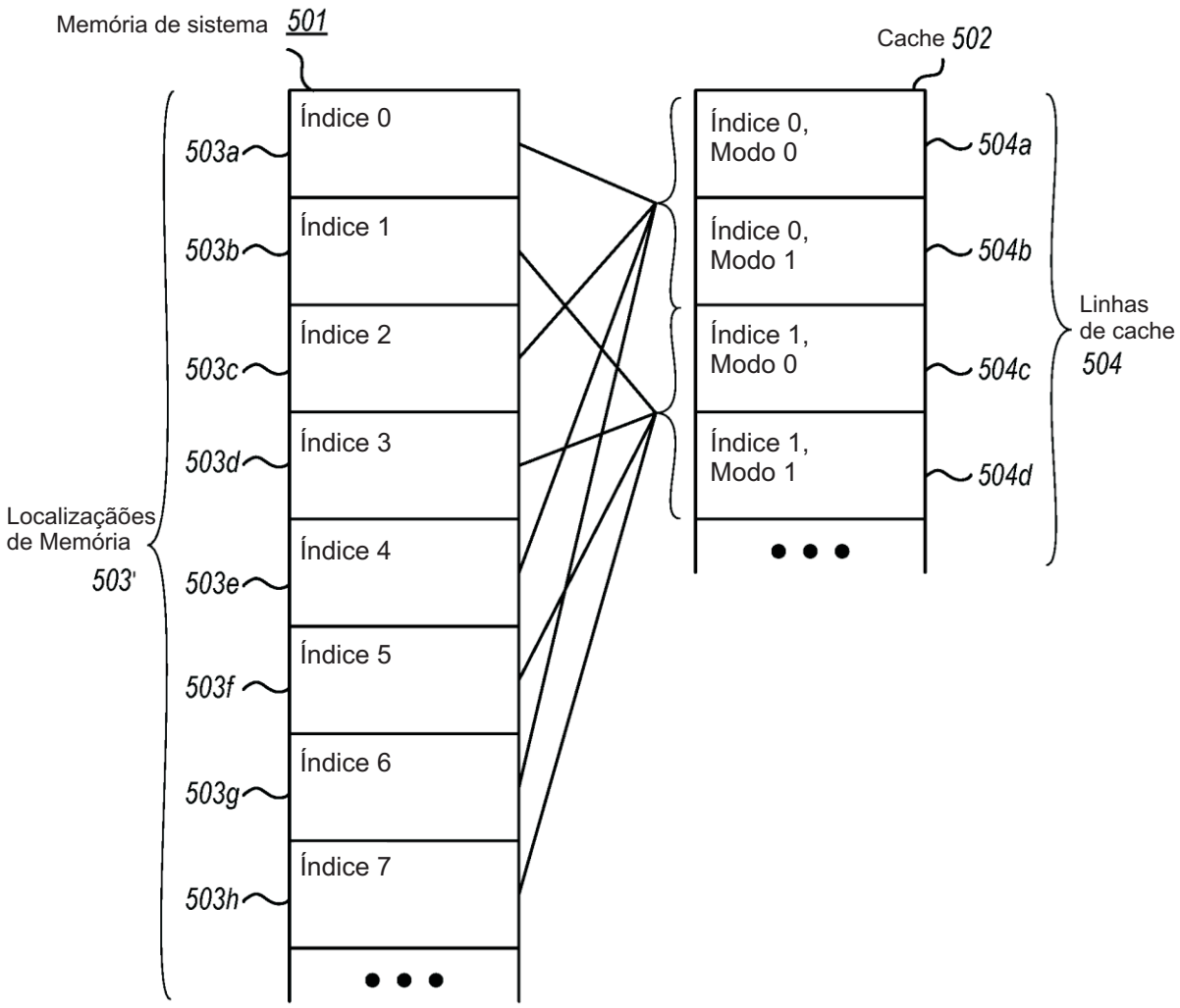

Memória de sistema 501
Cache 502
Índice 0
Índice 1
Índice 2
Índice 3
Índice 4
Índice 5
Índice 6
Índice 7
503a
503b
503c
503d
503e
503f
503g
503h
Localizaçãões de Memória 503'
Índice 0, Modo 0
Índice 0, Modo 1
Índice 1, Modo 0
Índice 1, Modo 1
504a
504b
504c
504d
Linhas de cache 504

FIG. 5

600a

601 ID	602a P0	602b P1	602c P2	602d P3
0	R[1]			
1		R[1]		
2	W[2]			
3		R[2]		
4	W[3]			
5			R[3]	
6		R[3]		
7				R[3]
8				W[4]
9	W[5]			
10			R[5]	
11		R[5]		
12	R[5]			
13	Linha de remoção			

FIG. 6A

600b

601 ID	602a P0	602b P1	602c P2	602d P3
0	S			
1	S	S		
2	M	I		
3	S	S		
4	M	I		
5	S		S	
6	S	S	S	
7	S	S	S	S
8	I	I	I	M
9	M			I
10	S		S	
11	S	S	S	
12	S	S	S	
13	-	-	-	-

FIG. 6B

600c

601	603				604	605
ID	Bits de unidade				Índice	Sinalizador
0	**1**	**0**	**0**	**0**	**0**	**1**
1	**1**	**1**	**0**	**0**	**1**	**1**
2	**1**	**0**	**0**	**0**	**0**	**1**
3	**1**	**1**	**0**	**0**	**1**	**1**
4	**1**	**0**	**0**	**0**	**0**	**1**
5	**1**	**0**	**1**	**0**	**2**	**1**
6	**1**	**1**	**1**	**0**	**1**	**1**
7	**1**	**1**	**1**	**0**	**1**	**1**
8	**0**	**0**	**0**	**0**	**-1**	**0**
9	**0**	**0**	**0**	**0**	**-1**	**0**
10	**0**	**0**	**1**	**0**	**2**	**1**
11	**0**	**1**	**1**	**0**	**1**	**1**
12	**1**	**1**	**1**	**0**	**0**	**1**
13	**0**	**0**	**0**	**0**	**-1**	**0**

FIG. 6C

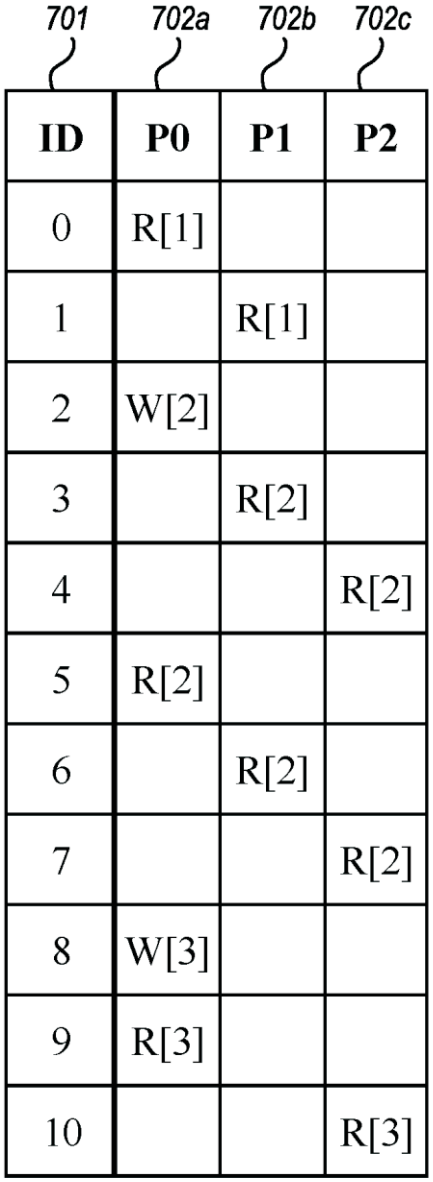
600d

601 606

ID	Dados de registro
0	P0; @; IC[0]; DATA [1]
1	P1; @; IC[1]; R->R; P0->P0, P1
2	P0; @; IC[2]; R->W; P0, P1->P0
3	P1; @; IC[3];W->R; P0->P0, P1
4	P0; @; IC[4]; R->W; P0, P1->P0
5	P2; @; IC[5]; W->R; P0->P0, P2
6	P1; @; IC[6]; R->R; P0, P2->P0, P1, P2
7	
8	
9	
10	P2; @; IC[10]; DATA[5], P2
11	P1; @; IC[11]; R->R; P2->P1, P2
12	P0; @; IC[12]; R->R; P1, P2->P0, P1, P2
13	

FIG. 6D

700a



ID	P0	P1	P2
0	R[1]		
1		R[1]	
2	W[2]		
3		R[2]	
4			R[2]
5	R[2]		
6		R[2]	
7			R[2]
8	W[3]		
9	R[3]		
10			R[3]

FIG. 7A

700b

0	P0; @; IC[0]; DATA [1]
1	P1; @; IC[1]; R->R; P0->P0, P1
2	P0; @; IC[2]; R->W; P0, P1->P0
3	P1; @; IC[3];W->R; P0->P0, P1
4	P2; @; IC[4]; R->R; P0, P1->P0, P1, P2
5	
6	
7	
8	P0; @; IC[8]; R->W; P0, P1, P2->P0
9	
10	P2; @; IC[10]; R->R; P0->P0, P2

FIG. 7B

700c

0	P0; @; IC[0]; DATA [1]
1	P1; @; IC[1]; R->R; P0-> P1
2	P0; @; IC[2]; R->W; P1->P0
3	P1; @; IC[3];W->R; P0->P1
4	P2; @; IC[4]; R->R; P1->P2
5	P0; @; IC[5]; R->R; P2->P0
6	P1; @; IC[6]; R->R; P0->P1
7	P2; @; IC[7]; R->R; P1->P2
8	P0; @; IC[8]; R->W; P2->P0
9	
10	P2; @; IC[10]; R->R; P0->P2

FIG. 7C

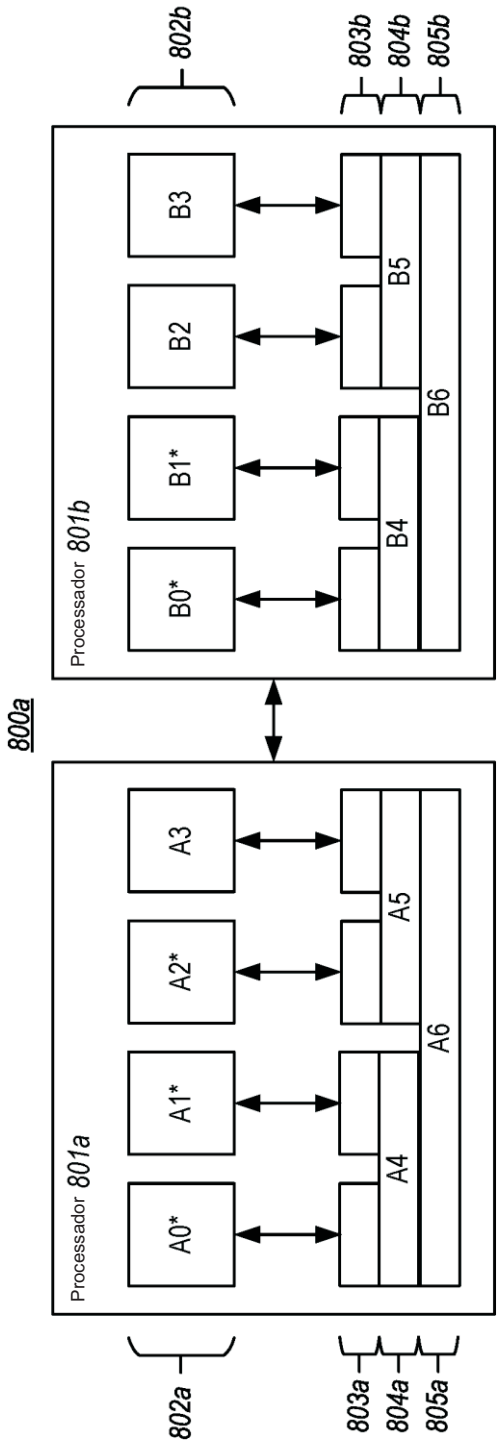

800a
Processador 801a
A0*
A1*
A2*
A3
A4
A5
A6
Processador 801b
B0*
B1*
B2
B3
B4
B5
B6
802a
803a
804a
805a
802b
803b
804b
805b

FIG. 8A

800b

ID	A0	A1	A2	A3	B0	B1	B2	B3
0	R[1]							
1					R[1]			
2		R[1]						
3			W[2]					
4			R[2]					
5					R[2]			
6						R[2]		
7					R[2]			
8	W[3]							
9	W[4]							
10		R[4]						
11			R[4]					
12					R[4]			
13						R[4]		
14	R[4]							

FIG. 8B

900a

ID	P0	P1
0	W[1]	
1		R[1]
2	R[1]	
3		R[1]
4	W[2]	
5		R[2]

FIG. 9A

900b

ID	Unidades de CCP + sinalizador	Índice de CCP + sinalizador
0		
1	Registrar DATA	Registrar DATA
2	Registrar R->R	Registrar R->R
3		Registrar R->R
4	Registrar R->W	Registrar R->W
5	Registrar W->R	Registrar W->R

FIG. 9B

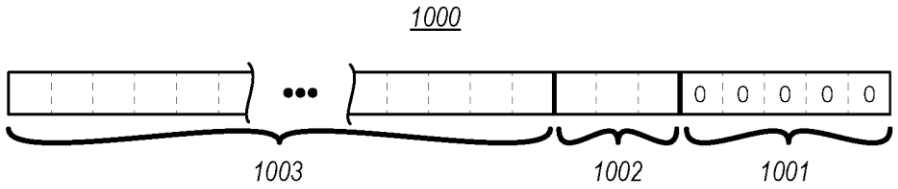


FIG. 10A

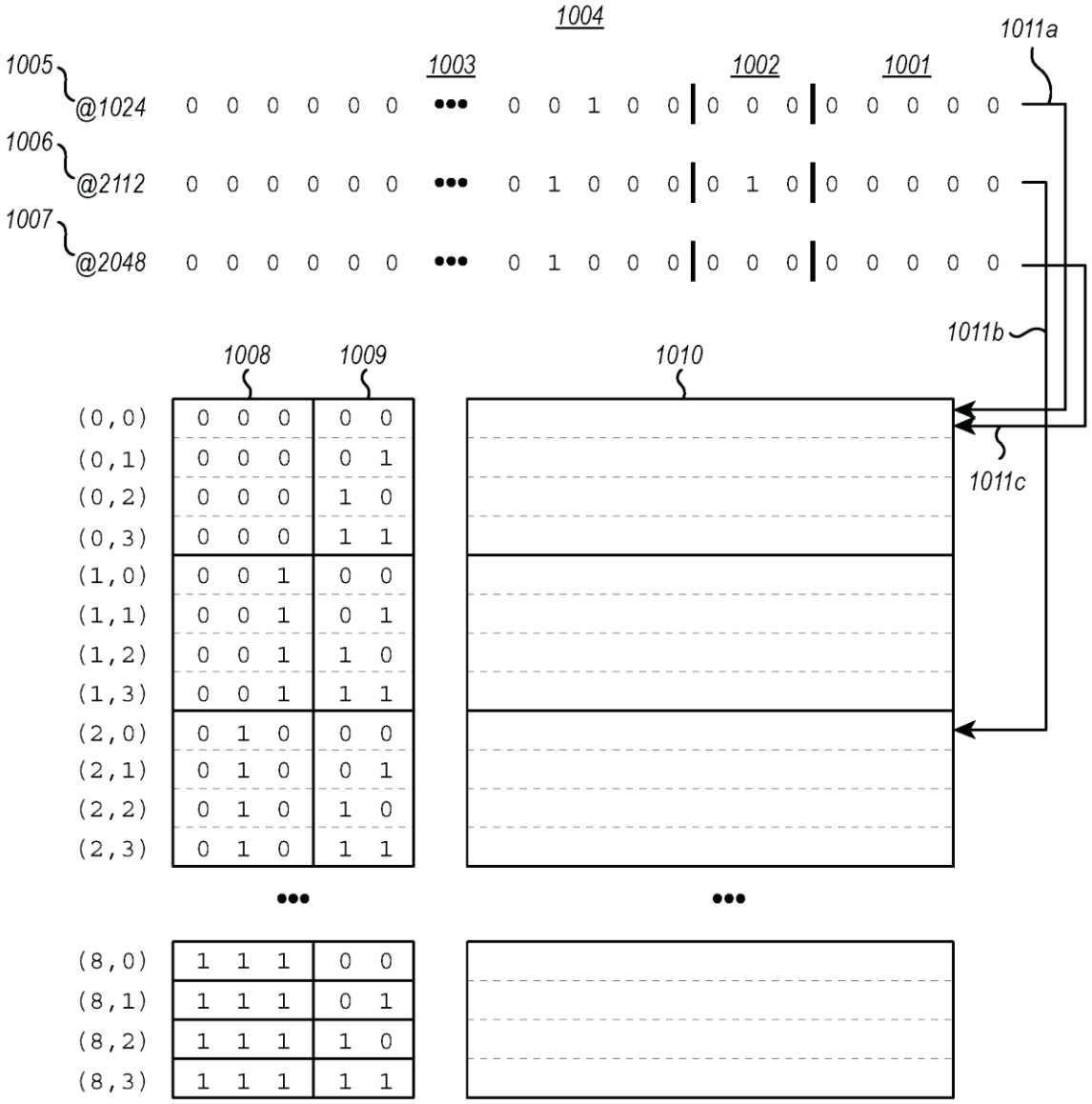


FIG. 10B

RESUMO

Patente de Invenção: **"GRAVAÇÃO DE RASTREIO BASEADO EM CACHE UTILIZANDO DADOS DE PROTOCOLO DE COERÊNCIA DE CACHE".**

A presente invenção refere-se a executar uma gravação de rastreamento baseada em cache utilizando dados de protocolo de coerência de cache (CCP). As modalidades detectam que uma operação que causa uma interação entre uma linha de cache e um armazenamento de apoio ocorreu, que o registro é habilitado para a unidade de processamento que causou a operação, que a linha de cache é uma participante no registro, e que o CCP indica que existem dados a serem registrados em um rastreamento. As modalidades então fazem com que os dados sejam registrados no rastreamento, cujos dados são utilizáveis para reproduzir a operação.