(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2017 年 1 月 19 日(19.01.2017)

WIPO | PCT

(10) 国際公開番号
WO 2017/010004 A1

(51) 国際特許分類:
G06F 12/08 (2006.01)

(21) 国際出願番号: PCT/JP2015/070438

(22) 国際出願日: 2015 年 7 月 16 日(16.07.2015)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(71) 出願人: 株式会社東芝 (KABUSHIKI KAISHA TOSHIBA) [JP/JP]; 〒1058001 東京都港区芝浦一丁目１番１号 Tokyo (JP).

(72) 発明者: 城田 祐介(SHIROTA, Yusuke); 〒1058001 東京都港区芝浦一丁目１番１号 株式会社東芝 知的財産室内 Tokyo (JP). 金井 達徳(KANAI, Tatsunori); 〒1058001 東京都港区芝浦一丁目１番１号 株式会社東芝 知的財産室内 Tokyo (JP). 樽家 昌也(TARUI, Masaya); 〒1058001 東京都港区芝浦一丁目１番１号 株式会社東芝 知的財産室内 Tokyo (JP).

(74) 代理人: 特許業務法人酒井国際特許事務所 (SAKAI INTERNATIONAL PATENT OFFICE); 〒1000013 東京都千代田区霞が関３丁目８番１号 虎の門三井ビルディング Tokyo (JP).

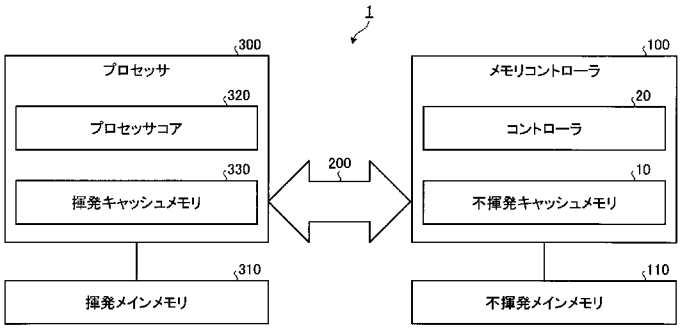
(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

添付公開書類:
— 国際調査報告（条約第 21 条(3)）

(54) **Title**: MEMORY CONTROLLER, INFORMATION PROCESSING DEVICE, AND PROCESSING DEVICE

(54) 発明の名称 : メモリコントローラ、情報処理装置および処理装置



10 Non-volatile cache memory
20 Controller
100 Memory controller
110 Non-volatile main memory
300 Processor
310 Volatile main memory
320 Processor core
330 Volatile cache memory

(57) **Abstract**: According to the present invention, a high-performance and highly-reliable data perpetuating method is realized. A memory controller of an embodiment includes a non-volatile cache memory and a controller. The non-volatile cache memory stores therein one portion of data stored in a non-volatile main memory connected to the memory controller. The controller controls writing into the non-volatile cache memory. The memory controller is connected to a processing device via an interconnector that guarantees a protocol indicating a procedure for preventing mismatch of data among a plurality of cache memories. After detecting that data stored in any region of the non-volatile main memory has been updated by the processing device, the controller transmits the updated data to the memory controller and writes the updated data into the non-volatile cache memory by using the protocol.

(57) 要約:

[続葉有]

WO 2017/010004 A1

高性能・高信頼なデータ永続化処理方式を実現する。実施形態のメモリコントローラは、不揮発キャッシュメモリとコントローラとを有する。不揮発キャッシュメモリは、メモリコントローラに接続された不揮発メインメモリに記憶されたデータの一部を記憶する。コントローラは、不揮発キャッシュメモリへの書き込みを制御する。メモリコントローラは、複数のキャッシュメモリ間のデータの不一致が起こらないようにするための手順を示すプロトコルを保証するインターコネクトを介して、処理装置と接続される。コントローラは、プロトコルを利用して、処理装置が、不揮発メインメモリの何れかの領域に記憶されたデータを更新したことを検出した後に、その更新済みのデータをメモリコントローラへ送信させ、その更新済みのデータを不揮発キャッシュメモリに書き込む。

明　細　書

発明の名称 ： メモリコントローラ、情報処理装置および処理装置

技術分野

[0001] 本発明の実施形態は、メモリコントローラ、情報処理装置および処理装置に関する。

背景技術

[0002] ＭＲＡＭ（Magnetoresistive Random Access Memory）、ＰＣＭ（Phase Change Memory）、Ｍｅｍｒｉｓｔｏｒなどのような、プロセッサのメモリバスに直結できるバイトアドレッサブルで高速な不揮発メモリをメインメモリ（永続メモリ）として使用するコンピュータシステムは、高信頼・高性能が要求されるオンライントランザクション処理システムなどのアプリケーションに利用すると、重要なデータをメインメモリに保存すれば該データを永続化できる。そのため、従来システムのように、ハードディスクやＳＳＤ（Solid State Drive）にデータを保存する場合と比較して、高速にデータを永続化できるので、オンライントランザクション処理データベースアプリケーションなどのアプリケーションを高性能化できる可能性を有している。

[0003] また、トランザクション処理システムは高信頼性も要求されるため、プロセッサ上で実行するプログラムによる不揮発メモリへの書き込みのアトミシティ（書き込み途中の状態で終わらない性質）やオーダリング（順序性）を保証するデータの永続的な更新処理方式が必要である。

先行技術文献

特許文献

[0004] 特許文献1：特開２００４－３６２５７０号公報

発明の概要

発明が解決しようとする課題

[0005] しかしながら、従来技術では、十分に高性能・高信頼なデータ永続化処理方式を実現することは困難であった。

課題を解決するための手段

[0006] 実施形態のメモリコントローラは、不揮発キャッシュメモリとコントローラとを有する。不揮発キャッシュメモリは、メモリコントローラに接続された不揮発メインメモリに記憶されたデータの一部を記憶することなどに利用される。コントローラは、不揮発キャッシュメモリへの書き込みや読み出しを制御する。メモリコントローラは、複数のキャッシュメモリ間のデータの不一致が起こらないようにキャッシュの一貫性を維持・管理するための手順を示すプロトコルを保証するインターコネクトを介して処理装置と接続される。バスやネットワークで接続されてもよい。コントローラは、プロトコルを利用して、処理装置が、不揮発メインメモリの何れかの領域に記憶されたデータを更新したことを検出した後に、その更新済みのデータをメモリコントローラへ送信させ、その更新済みのデータを不揮発キャッシュメモリに書き込む。

図面の簡単な説明

[0007] [図1]第１の実施形態の情報処理装置の構成の一例を示す図。

[図2]対比例の情報処理装置の構成の一例を示す図。

[図3]変形例の情報処理装置の構成の一例を示す図。

[図4]実施形態のアトミシティを説明するための図。

[図5]実施形態のアトミシティを説明するための図。

[図6]第１の実施形態の永続データの更新を行う処理の一例を示す図。

[図7]第１の実施形態の永続データが永続化される手順の一例を示す図。

[図8]第１の実施形態の永続データが永続化される手順の詳細を示す図。

[図9]第１の実施形態のメモリコントローラの動作例を示す図。

[図10]変形例の永続データが永続化される手順の一例を示す図。

[図11]第１の実施形態の永続データの永続化の完了確認手順の一例を示す図。

[図12]第１の実施形態の情報処理装置の動作例を示す図。

[図13]第２の実施形態の永続データが永続化される手順の一例を示す図。

[図14]第２の実施形態の情報処理装置の動作例を示す図。

[図15]第２の実施形態の情報処理装置の動作例を示す図。

発明を実施するための形態

[0008] 以下、添付図面を参照しながら、本発明に係るメモリコントローラ、情報処理装置および処理装置の実施形態を詳細に説明する。

[0009] （第１の実施形態）

具体的な内容を説明する前に、従来技術と併せて本実施形態の概要を説明する。ＭＲＡＭやＰＣＭやＭｅｍｒｉｓｔｏｒのような、プロセッサのメモリバスに直結できるバイトアドレッサブルで高速な不揮発メモリをメインメモリとして使用するコンピュータシステムは、高信頼・高性能が要求されるオンライントランザクション処理システムなどに利用すると、重要なデータをメインメモリに保存すれば該データを永続化できる。そのため、従来システムのように、ハードディスクやＳＳＤにデータを保存する場合と比較して、データベースのデータを永続化する処理（コミット処理）を高速化できることが期待されている。

[0010] 本実施形態では、これらのメインメモリ向けの高速な不揮発メモリよりもさらに高速に動作する不揮発キャッシュメモリを、不揮発メインメモリと接続されたメモリコントローラに内蔵させ、不揮発キャッシュメモリに書き込まれたデータの永続性を保証することで、コミット処理をさらに高速化する。不揮発キャッシュメモリは、例えばＳＲＡＭ並みの速度で動作するＭＲＡＭ、ＰＣＭ、Ｍｅｍｒｉｓｔｏｒなどの不揮発メモリである。

[0011] 図１に示すように、本実施形態では、不揮発キャッシュメモリ１０を内蔵したメモリコントローラ１００が、ＭＥＳＩＦプロトコル、後述のＱＰＩで実装されているプロトコル（ＱＰＩプロトコル）、ＭＥＳＩプロトコル、ＭＯＥＳＩプロトコルなどの複数のプロセッサ（あるいはプロセッサコア）のキャッシュメモリ間のデータの不一致が起こらないようにする（キャッシュメモリ間の一貫性を維持する）ための手順を示すキャッシュコヒーレンスプロトコルを保証するＱＰＩ（Intel QuickPath Interconnect）などの高速

なインターコネクト（キャッシュコヒーレント・インターコネクト）２００を介して、プロセッサ（典型的にはＣＰＵ、「処理装置」に対応）３００と接続される。図１の詳細な構成については後述する。

[0012] これらのキャッシュコヒーレンスプロトコルは、一般的にはプロセッサ同士を接続するために設計されているが、本実施形態ではプロセッサにメモリコントローラを接続し、メモリコントローラ側からも制御するためにこれを用いる。また、これらのキャッシュコヒーレンスプロトコルは、一般的には揮発キャッシュ同士のコヒーレンス維持のために設計されているが、本実施形態では揮発キャッシュと不揮発キャッシュ（あるいは不揮発キャッシュ同士）のコヒーレンス維持のためにこれを用い、揮発キャッシュに対する更新を不揮発キャッシュに反映させて更新を永続化する。

[0013] 図１の構成は、例えば図２に示すような、ＱＰＩなどのインターコネクト２００で２つのプロセッサ（第１プロセッサ４００、第２プロセッサ５００）を直接接続したマルチプロセッサシステム（例えばＩｎｔｅｌ　ＸＥＯＮプロセッサシステム）に含まれる２つのプロセッサのうちの一方を、不揮発キャッシュメモリ１０を内蔵したメモリコントローラ１００に置き換えた構成であると捉えることもできる。

[0014] 図２の例では、第１プロセッサ４００は、例えばＤＲＡＭ（Dynamic Random Access Memory）などの第１揮発メインメモリ４１０に接続され、第２プロセッサ５００は、例えばＤＲＡＭなどの第２揮発メインメモリ５１０に接続されている。第１プロセッサ４００は、第１揮発メインメモリ４１０または第２揮発メインメモリ５１０に対するデータの読み書きを行う１以上のプロセッサコア４２０（説明の便宜上、１つのプロセッサコア４２０のみを例示、他も同様）と、第１揮発メインメモリ４１０または第２揮発メインメモリ５１０に記憶されたデータの一部を記憶する第１揮発キャッシュメモリ４３０とを有する。また、第２プロセッサ５００は、第１揮発メインメモリ４１０または第２揮発メインメモリ５１０に対するデータの読み書きを行う１以上のプロセッサコア５２０と、第１揮発メインメモリ４１０または第２

揮発メインメモリ５１０に記憶されたデータの一部を記憶する第２揮発キャッシュメモリ５３０とを有する。

[0015] ここでは、メモリコントローラ１００さえ用意すれば、ＱＰＩなどのインターコネクト２００を介して、ＸＥＯＮプロセッサシステムのようなマルチプロセッサシステムを構成する既存のプロセッサと該メモリコントローラ１００とを接続するだけで、本実施形態の構成を実現することも可能である。

[0016] また、図３は、他の構成例として３つのプロセッサ（第１プロセッサ４００、第３プロセッサ６００、第４プロセッサ７００）、および、不揮発キャッシュメモリ１０を内蔵したメモリコントローラ１００の各々がＱＰＩなどのインターコネクト２００を介して相互に接続される構成を示しているが、これに限られるものではなく、例えば複数のプロセッサと複数のメモリコントローラ１００とが互いに接続される構成であってもよく、スケーラブルなコンピュータシステムの構築が可能である。

[0017] 図３の例では、第１プロセッサ４００は、例えばＤＲＡＭなどの第１揮発メインメモリ４１０に接続され、第３プロセッサ６００は、例えばＤＲＡＭなどの第３揮発メインメモリ６１０に接続され、第４プロセッサ７００は、例えばＤＲＡＭなどの第４揮発メインメモリ７１０に接続され、メモリコントローラ１０は、例えばＭＲＡＭなどの不揮発メインメモリ１１０に接続されている。

[0018] 第１プロセッサ４００は、何れかのメインメモリ（第１揮発メインメモリ４１０、不揮発メインメモリ１１０、第３揮発メインメモリ６１０、第４揮発メインメモリ７１０）に対するデータの読み書きを行う１以上のプロセッサコア４２０と、何れかのメインメモリに記憶されたデータの一部を記憶する第１揮発キャッシュメモリ４３０とを有する。また、第３プロセッサ６００は、何れかのメインメモリに対するデータの読み書きを行う１以上のプロセッサコア６２０と、何れかのメインメモリに記憶されたデータの一部を記憶する第３揮発キャッシュメモリ６３０とを有する。また、第４プロセッサ７００は、何れかのメインメモリに対するデータの読み書きを行う１以上の

プロセッサコア７２０と、何れかのメインメモリに記憶されたデータの一部を記憶する第４揮発キャッシュメモリ７３０とを有する。メモリコントローラ１００の具体的な構成については後述する。

[0019] また、トランザクション処理システムは高信頼性も要求されるため、プロセッサ上で実行するプログラムによる不揮発メモリへの書き込みのアトミシティ（書き込み途中の状態で終わらない性質）やオーダリング（順序性）を保証するデータの永続的な更新処理方式が必要である。

[0020] アトミシティを保証する方式として、例えば、ＷＡＬ（Write Ahead Logging）やシャドウページなどと呼ばれるデータの複数バージョンを管理する方式が広く知られている。これらデータ永続化を実現するプログラミング技法では、最初はメインメモリに記憶されたオリジナルのデータを更新せずに、まずはメインメモリの別の領域に確保したデータの値を更新してデータの新しいバージョンを作成することで明示的にデータの複数のバージョンを生成し、メモリバリアを挟んで、その後でオリジナルのデータを更新する、または、オリジナルのデータへの参照を新しいバージョンのデータへの参照に切り替える等の処理を行う。しかし、この方法では、図２の第１揮発メインメモリ４１０と第２揮発メインメモリ５１０を不揮発メモリに置き換えただけのシステムにおいては、新しいバージョンを作成するために不揮発メインメモリ（永続メモリ）へのアクセスが必要になるため、十分に高速化できない。

[0021] また、上述の複数バージョンを管理する方式の場合、メモリバリア以前の書き込みが不揮発メインメモリのセルに到着した後に、メモリバリア後の書き込みが行われるようにタイミングを制御する必要がある。不揮発メインメモリへの書き込みのオーダリングを保証する上で問題になるのが、プロセッサコアと不揮発メインメモリの間に存在する各種キャッシュやライトバッファやライトコンバインバッファなどのメモリへの書き込みタイミングを調整するためのバッファなどの揮発メモリである。この存在により、プロセッサコアがライト命令でデータを書く場合において、不揮発メインメモリに到達

するデータの順番が入れ替わる、あるいは、直ちに不揮発メインメモリには書き込まれずにデータがキャッシュに留まっているといった事態が起こる。そのため、順序性を保証するためには、例えばキャッシュフラッシュ命令を利用して、データをキャッシュから順番に追い出しながら処理を進める必要がある。しかし、キャッシュフラッシュ命令はレイテンシが大きいため、性能面で課題がある。

[0022] 本実施形態では、ＱＰＩなどのインターコネクト２００を介して、不揮発キャッシュメモリ１０を内蔵したメモリコントローラ１００と、プロセッサ３００とを接続し、アトミシティとオーダリングを保証しつつ高速なコミット処理を実現するための手段を提供する。

[0023] 一つ目の手段は、キャッシュコヒーレンスプロトコルを利用して、コミット処理の対象となるデータ（永続データ）をアプリケーションなどが書き込んだ直後に高速に、メモリコントローラ１００に内蔵された不揮発キャッシュメモリ１０に転送して追い出して永続化するための手段である。より具体的な内容については後述するが、本実施形態では、複数のプロセッサ（プロセッサコア）のキャッシュメモリの状態が矛盾しないように制御するキャッシュコヒーレンスプロトコルの特性を利用して、プロセッサコアで永続データに対して直接書き込み処理（インプレース更新）を行うと、即座に更新内容がメモリコントローラ１００に転送され、その転送された更新済みのデータが不揮発キャッシュメモリ１０に書き込まれて永続化される。

[0024] メモリコントローラ１００側から見ると、不揮発キャッシュメモリ１０を内蔵するメモリコントローラ１００は、キャッシュを持つプロセッサ３００とキャッシュコヒーレンスプロトコルで両キャッシュの一貫性を保証するインターコネクトで接続され、接続されたプロセッサ３００が永続データに対して書き込みを行うと、プロセッサ３００内にある書き込み後のデータをメモリコントローラ１００側に転送させるために、キャッシュコヒーレンスプロトコルで決められた、更新したデータを送付する手順を引き起こす要求リクエストをプロセッサ３００に対して送付し、送られてきたデータを不揮発

キャッシュメモリ１０に対して書き込み永続化する。

[0025] 図４および図５を用いて、本実施形態におけるアトミシティについて説明する。プロセッサ３００のプロセッサコア３２０は、明示的な複数バージョン管理方式を使わずに、不揮発メインメモリ１１０の領域（アドレス）Ｘに記憶されたデータ（値Ａ）のインプレース更新を実施しようとする。領域Ｘに記憶されたデータ（値Ａ）は不揮発メインメモリ１１０より読み出され、ＱＰＩなどのインターコネクト２００を介してプロセッサ３００側に送られ、プロセッサコア３２０でインプレース更新すると、更新されたデータ（値Ａ’）は、キャッシュコヒーレンスプロトコルに従って、インターコネクト２００を介してプロセッサ３００と接続されたメモリコントローラ１００へ追い出される。そして、更新されたデータ（値Ａ’）は、不揮発キャッシュメモリ１０に書き込まれる。更新前のデータ（値Ａ）は、まだ不揮発メインメモリ１１０上に存在するので、この時点で、領域Ｘに対応するデータとして、不揮発メインメモリ１１０上のデータ（値Ａ）と、不揮発キャッシュメモリ１０上のデータ（値Ａ’）とが同時に存在する状態を、明示的に別の領域（アドレス）などに複数バージョンを作ることなく実現でき、アトミシティを保証できる。永続データの複数のバージョンが永続メモリ上に同時に存在しているので、この時点でコミット処理を完了できる状態になっている。すなわち、更新されたデータを、不揮発メインメモリ１１０まで書き込むことなくより高速な不揮発キャッシュメモリ１０に書き込むだけで、コミット処理を完了できるので、より高速なコミット処理が可能になる。

[0026] 二つ目の手段は、例えばトランザクションのコミット処理の完了を確定するために、更新された永続データの書き込みの完了（言い換えれば、更新された永続データの永続化の完了）を確認するメモリバリアのための手段である。図４のように、状態（ｂ）にしてメモリバリアを実行した後に、状態（ｃ）にする必要があり、この順序を保証することがオーダリングの保証である。一般に、プロセッサコアが書き込み処理をしたと認識しても、更新されたデータは、プロセッサコアとメインメモリの間にあるキャッシュやライト

バッファなどの揮発メモリに留まっていて、更新されたデータがどのタイミングで実際にメインメモリに書き込まれたかをプロセッサコア（つまりプロセッサコアで動作するアプリケーション）が知ることは難しい。本実施形態では、キャッシュコヒーレンスプロトコルを利用して、プロセッサ３００（アプリケーション）が永続データを更新すると、更新されたデータを即座にメモリコントローラ１００に内蔵された不揮発キャッシュメモリ１０へ追い出すための手段を提供しており、この手段が、オーダリングを保証する役割の一つを担っている。ただし、オーダリングを保証するためには、更新されたデータが不揮発キャッシュメモリ１０に到達して実際の書き込み処理が完了したのかを確認する手段も必要である。より具体的な内容については後述するが、本実施形態では、キャッシュコヒーレンスプロトコルを利用して、プロセッサ３００（アプリケーション）が、メモリコントローラ１０へ追い出したデータ（更新されたデータ）の書き込み処理の完了（永続化の完了）を確認するための手段を提供することで、オーダリングを保証している。

[0027] 以上のように、本実施形態では、キャッシュコヒーレンスプロトコルを利用した上記２つの手段でアトミシティとオーダリングを保証しつつ、高速なコミット処理を実現する。ＱＰＩなどのインターコネクト２００は、ＸＥＯＮプロセッサなどのプロセッサが接続されるように設計されており、インターコネクト２００を介してメモリコントローラ１００とプロセッサを接続しても、プロセッサは、インターコネクト２００を介した接続先も自身と同様のプロセッサであるとみなして、キャッシュコヒーレンスプロトコルに従ったメッセージのやり取りを行う。これを利用して、メモリコントローラ１００は、更新された永続データや更新されるタイミングの情報をプロセッサから得ることを特徴の一つとする。

[0028] 以下、本実施形態の具体的な内容を詳細に説明する。一例として、図１に示す情報処理装置１を例に挙げて説明する。図１に示すように、情報処理装置１は、不揮発メインメモリ１１０が接続されたメモリコントローラ１００と、揮発メインメモリ３１０が接続されたプロセッサ３００と、を備え、こ

れらはインターコネクト２００を介して接続される。インターコネクト２００は、複数のキャッシュメモリ間のデータの不一致が起こらないようにするための手順を示すプロトコル（キャッシュコヒーレンスプロトコル）を保証するインターコネクトであり、例えばＱＰＩなどで構成される。メモリコントローラ１００とプロセッサ３００は、バスやネットワークなどで接続されてもよい。

[0029] プロセッサ３００は、例えばＸＥＯＮプロセッサなどのＣＰＵであり、インターコネクト２００が保証するキャッシュコヒーレンスプロトコルに従って動作する。図１に示すように、プロセッサ３００は、１以上のプロセッサコア３２０（説明の便宜上、１つのプロセッサコア３２０のみを例示）と、揮発キャッシュメモリ３３０とを有する。プロセッサコア３２０は、揮発メインメモリ３１０または不揮発メインメモリ１１０に対するデータの読み書きを行う。揮発キャッシュメモリ３３０は、揮発メインメモリ３１０または不揮発メインメモリ１１０に記憶されたデータの一部を記憶する。

[0030] この例では、１以上のプロセッサコア３２０ごとに、Ｌ１データキャッシュ、Ｌ１命令キャッシュ、Ｌ２キャッシュなどのプライベートキャッシュを有している（不図示）。また、例えばＬ３キャッシュなどのキャッシュ階層における最下位に位置するキャッシュであるラストレベルキャッシュ（ＬＬＣ）は各プロセッサコア３２０で共有されるシェアドキャッシュである。これらはＳＲＡＭ（Static Random Access Memory）などの揮発性のキャッシュメモリで構成されることを前提としており、ここでは、これらを揮発キャッシュメモリ３３０と称する。以下では、このような構成を前提として説明するが、これに限られるものではない。また、プロセッサ３２０は、プロセッサコア３２０の指示に従ってデータの読み書きを制御するＤＲＡＭコントローラなどのメモリコントローラ（不図示）を内包しており、これを介して揮発メインメモリ３１０が接続されている。また、ラストレベルキャッシュ（ＬＬＣ）などは不揮発キャッシュであってもよい。

[0031] 揮発メインメモリ３１０は、例えばＤＲＡＭなどの揮発メモリで構成され

るメインメモリ（主記憶装置）である。プロセッサ３００に接続されるメインメモリとしては、例えばＤＲＡＭ以外の揮発メモリであってもよいし、例えばＭＲＡＭなどの不揮発メモリであってもよい。

[0032] また、図１に示すように、メモリコントローラ１００は、不揮発キャッシュメモリ１０と、コントローラ２０とを有する。不揮発キャッシュメモリ１０は、不揮発メインメモリ１１０に記憶されたデータの一部を記憶するなどに利用される。コントローラ２０は、不揮発キャッシュメモリ１１０へのデータの書き込み及び読出しなどのメモリアクセスを制御する。プロセッサ３００は、このメモリコントローラ１００を介して不揮発メインメモリ１１０と接続されている。

[0033] 本実施形態における不揮発メインメモリ１１０は、ＭＲＡＭで構成されるが、これに限られるものではない。例えば不揮発メインメモリ１１０は、ＰＣＭ、ＲｅＲＡＭ（Resistive Random Acess Memory）、ＦｅＲＡＭ（Ferroelectric Random Access Memory）、Ｍｅｍｒｉｓｔｏｒなどで構成されてもよいし、ＤＲＡＭとＮＡＮＤ Ｆｌａｓｈを組み合わせたＮＶＤＩＭＭ（Non‐volatile DIMM）であってもよい。また、不揮発メインメモリ１１０は、バッテリでバックアップされる（電源がオフになってもバッテリからの電力供給が継続される）揮発メモリ（ＤＲＡＭやＳＲＡＭなど）であってもよい。

[0034] コントローラ２０は、インターコネクト２００が保証するキャッシュコヒーレンスプロトコルを利用して、プロセッサ３００が、不揮発メインメモリ１１０の何れかの領域に対応するデータを更新したことを検出した後に、その更新済みのデータをメモリコントローラ１００へ送信させ、その更新済みのデータを不揮発キャッシュメモリ１０に書き込む。

[0035] より具体的には、コントローラ２０は、プロセッサ３００から、不揮発メインメモリ１１０の何れかの領域に対応するデータを要求する第１の要求を受信した後に、該領域に対応するデータと、該データを不揮発キャッシュメモリ１０に保持していたことを示す情報とを含む第１応答メッセージを、イ

ンターコネクト２００を介してプロセッサ３００へ送信する。コントローラ２０は、第１応答メッセージを送信した後に、プロセッサ１０から、該領域に対応するキャッシュラインを無効にすることを要求する第２の要求を受信したときに、プロセッサ３００（アプリケーション）が、該領域に対応するデータを更新したことを検出する。なお、キャッシュラインとは、キャッシュメモリに保持される単位情報であり、例えば不揮発メインメモリ１１０上の特定の領域に対応するキャッシュラインとは、該特定の領域に対応するデータのコピーである。ここでは、コントローラ２０は、第１の要求により要求された、不揮発メインメモリ１１０の何れかの領域に対応するデータを不揮発キャッシュメモリ１０に保持していない場合であっても、不揮発メインメモリ１１０の該領域から読み出したデータと、該データを不揮発キャッシュメモリ１０に保持していたことを示す情報とを含む第１応答メッセージをプロセッサ３００へ送信する。

[0036] また、コントローラ２０は、上記第２の要求を受信した後に、該領域に対応するキャッシュラインを無効にする処理は行わずに、第２の要求に対する応答として、該領域に対応するキャッシュラインを無効にしたことを通知するための第２応答メッセージを、インターコネクト２００を介してプロセッサ３００へ送信する。そして、コントローラ２０は、第２応答メッセージを送信した後に、該領域に対応するデータを要求する第３の要求を、インターコネクト２００を介してプロセッサ３００へ送信し、第３の要求に対する応答として、プロセッサ３００により更新済みのデータを受信する。そして、コントローラ２０は、その受信した更新済みのデータを不揮発キャッシュメモリ１０に書き込む。ここでは、上記第３の要求は、プロセッサ３００が更新した該領域に対応するデータの更新を行うために該領域に対応するデータを要求する情報であり、メモリコントローラ１００側に第２のプロセッサが接続されていると仮定した場合に第２のプロセッサで該領域に対応するデータを参照しようとして失敗（ライトミスやリードミス）した場合に送信される情報である。また、コントローラ２０は、上記第３の要求に対する応答と

して受信した更新済みのデータの更新は行わない。要するに、コントローラ２０は、更新済みのデータを送付する手順を引き起こす第３の要求をプロセッサ３００に対して送信し、第３の要求に対する応答として送られてきたデータを不揮発キャッシュメモリ１０に対して書き込む。更新済みのデータを送付する手順は、プロセッサ３００内の更新済みのデータをメモリコントローラ１００側に転送させるために、インターコネクト２００が保証するプロトコル（キャッシュコヒーレンスプロトコル）で決められた手順である。以上の詳細な動作例については後述する。

[0037] ここで、プロセッサ３００とメモリコントローラ１００とを接続するインターコネクト２００で保証しているキャッシュコヒーレンスプロトコルは、ＱＰＩで保証しているＱＰＩプロトコルやＭＥＳＩＦプロトコルのようなキャッシュコヒーレンスプロトコルであるが、これに限られるものではなく、例えばＭＳＩプロトコル、ＭＥＳＩプロトコル、ＭＯＥＳＩプロトコル、ＭＥＳＩＦプロトコルそのものでもよいし、これらを拡張・変更した任意のプロトコルであってもよい。例えばＭＥＳＩＦプロトコルにＭＯＥＳＩプロトコルのＯｗｎｅｄ状態を追加したプロトコルであってもよい。他の実施形態についても同様である。例えば既存のＸＥＯＮプロセッサとＱＰＩを用いて、独自のメモリコントローラ１００を設計・実装してＱＰＩ経由で接続する構成であってもよい。本実施形態のインターコネクト２００でサポートしているＱＰＩプロトコルやＭＥＳＩＦプロトコルのようなキャッシュコヒーレンスプロトコルでは、Ｍｏｄｉｆｉｅｄ状態、Ｅｘｃｌｕｓｉｖｅ状態、Ｓｈａｒｅｄ状態、Ｉｎｖａｌｉｄ状態、Ｆｏｗａｒｄ状態というキャッシュラインの状態を設け、基本動作はＱＰＩプロトコルやＭＥＳＩＦプロトコルに準ずる。詳細については、後述の詳細な動作例の中で随時説明していく。

[0038] 以上の構成の情報処理装置１において、インプレース更新により図４の永続データの複数バージョンを暗黙的に作成してコミット処理する例を説明する。まず、事前準備として、永続データを保存・管理するホームノード（Ｈｏｍｅ　ｎｏｄｅ、例えばＸＥＯＮプロセッサシステムの場合はＣＰＵが配

置される）が、メモリコントローラ１００が接続されるノードになるように配置する。例えば永続データを管理するホームノードの番号を指定するＡＰＩ（アプリケーションプログラムインタフェース）を利用することで、プログラムが設定することができる。例えば図３のように、情報処理装置のシステム構成上、第２プロセッサ５００が存在してよい位置にメモリコントローラ１００を配置し、このノードをホームノードにする場合において、ＡＰＩは、例えば永続データが記憶される領域（メモリ）を確保する際に、メモリ確保の関数の引数にホームノード番号の「２」を指定できるようなものであってもよい。これにより、図５のように、永続データが不揮発メインメモリ１１０の領域Ｘに配置される。要するに、本実施形態の情報処理装置１は、永続データの配置先となるメインメモリを設定するための設定手段を備える形態であればよい。この設定手段の一例として、ＡＰＩが挙げられる。あるいは、永続データを記憶するためにメモリを確保するための専用の関数を使うと、図５のように、永続データが不揮発メインメモリ１１０の領域Ｘに自動的に配置されるようなものであってもよい。

[0039] 図６は、アプリケーションが不揮発メインメモリ１１０の領域Ｘに対応するデータの更新（永続化）を行う処理の一例を示す図である。まず、プロセッサコア３２０（アプリケーション）は、不揮発メインメモリ１１０の領域Ｘに記憶されたデータ（この例では値Ａを示すデータ）をリード命令で読み出して揮発キャッシュメモリ３３０にキャッシュし、そのキャッシュライン（第１キャッシュライン）に対して、値Ａ’をライト命令で書き込んでインプレース更新する（ステップＳ７０１）。後述する「永続データを高速に不揮発キャッシュメモリ１０に追い出して永続化する手段」により、プロセッサ３００の揮発キャッシュメモリ３３０に書き込まれた更新済みのデータ（値Ａ’）は揮発キャッシュメモリ３３０上に留まることなく即座に不揮発キャッシュメモリ１０へ追い出される。そして、プロセッサコア３２０（アプリケーション）は、インターコネクト２００を介して接続された外部装置（この例ではメモリコントローラ１００）による更新済みのデータの永続化が

完了したのかを確認する永続化確認命令を実行する（ステップＳ７０２）。この命令を実行すると、後述する「永続データの永続化の完了を確認する手段」により、プロセッサ３００（アプリケーション）側で、外部装置（この例ではメモリコントローラ１００）による更新済みのデータの永続化が完了したのかを検出（確認）することが可能となる。以下、「永続データを高速に不揮発キャッシュメモリ１０に追い出して永続化する手段」と「永続データの永続化の完了を確認する手段」の動作の詳細を説明する。

[0040] まず、「永続データを高速に不揮発キャッシュメモリ１０に追い出して永続化する手段」で永続データが永続化される手順の一例を説明する。図７のステップＳ１に示すように、まず、プロセッサコア３２０は、不揮発メインメモリ１１０の領域Ｘに記憶されたデータ（値Ａ）を更新するために、領域Ｘに記憶されたデータ（値Ａ）を読み出して揮発キャッシュメモリ３３０に保存する。以下では、揮発キャッシュメモリ３３０のキャッシュラインのうち、不揮発メインメモリ１１０の領域Ｘに対応するキャッシュラインを「第１キャッシュライン」と称する場合がある。次に、図７のステップＳ２に示すように、プロセッサコア３２０は、第１キャッシュラインに対して、値Ａ’をライト命令で書き込んで第１キャッシュラインのデータ（領域Ｘに対応するデータ）をインプレース更新する。次に、図７のステップＳ３に示すように、メモリコントローラ１００が、不揮発キャッシュメモリ１０のキャッシュラインのうち、不揮発メインメモリ１１０の領域Ｘに対応するキャッシュライン（「第２キャッシュライン」と称する場合がある）への書き込みを行うように振る舞う（実際には書き込まない）と（ステップＳ３－１）、プロセッサ３００は更新済みのデータをメモリコントローラ１００へ追い出す（ステップＳ３－２）。そして、メモリコントローラ１００は、プロセッサ３００から受信した更新済みのデータ（領域Ｘに対応するデータ）を不揮発キャッシュメモリ１０に書き込む。

[0041] 図８は、図７の各ステップ（Ｓ１～Ｓ３）の詳細な手順の一例を示す図である。まず、ステップＳ１に対応する内容を説明する。まず、プロセッサコ

ア３２０が、不揮発メインメモリ１１０の領域Ｘのアドレスを指定して、領域Ｘに記憶されたデータを読み出すリード命令を実行すると、この例では、揮発キャッシュメモリ３３０は、領域Ｘに対応するキャッシュラインをキャッシュしていないＩＮＶＡＬＩＤ状態になっているので、キャッシュミスが発生し、領域Ｘのアドレスのホームノードに対して、キャッシュミスの第１スヌープリクエスト（領域Ｘに対応するデータを要求する情報）が送られる。第１スヌープリクエストは上述の第１の要求に相当する。この例では、ホームノードの位置には、プロセッサの代わりに本実施形態のメモリコントローラ１００が接続されており、メモリコントローラ１００に第１スヌープリクエストが到着する（他のプロセッサが存在する場合はそれらにも届く）。

[0042] この際、仮にメモリコントローラ１００の代わりに、キャッシュコヒーレンスプロトコルに従って動作する第２のプロセッサが存在し、第２のプロセッサで、領域Ｘに対応するキャッシュラインをキャッシュしていないと仮定すると、第２のプロセッサは、不揮発メインメモリ１１０の領域Ｘに記憶されたデータ（値Ａ）が入ったキャッシュラインを不揮発メインメモリ１１０から読み出し、インターコネクト２００を介してプロセッサ３００へ転送する。そして、プロセッサ３００は、転送されたキャッシュライン（領域Ｘに対応するキャッシュライン）を揮発キャッシュメモリ（ラストレベルキャッシュ）３３０へ保存する。そうすると、プロセッサ３００では、この領域Ｘに対応するキャッシュライン（第１キャッシュライン）は、キャッシュラインの内容が不揮発メインメモリ１１０から読み出されたときと変更がなく、かつ、他のどのプロセッサのキャッシュメモリにも同じアドレスのキャッシュラインが入っていない状態であるＥｘｃｌｕｓｉｖｅ状態になってしまう。

[0043] 後述する理由により、本実施形態では、プロセッサ３００の第１のキャッシュラインは、他のプロセッサも、領域Ｘに対応するキャッシュラインをキャッシュしている状態であるＳｈａｒｅｄ状態、あるいは、他のプロセッサも、領域Ｘに対応するキャッシュラインをキャッシュしていて、かつ、領域

Ｘに対応するキャッシュラインを要求された場合は唯一コピーを転送する形で応答することができる状態であるＦｏｗａｒｄ状態になるように、メモリコントローラ１００から制御することを狙いとする。そこで、メモリコントローラ１００のコントローラ２０は、第１スヌープリクエストが到着すると、領域Ｘに対応するキャッシュラインを唯一キャッシュしているＥｘｃｌｕｓｉｖｅ状態、あるいは、Ｆｏｗａｒｄ状態であるように振る舞い、領域Ｘに対応するキャッシュライン（不揮発メインメモリ１１０の領域Ｘに記憶されたデータ（値Ａ）を含む）と、領域Ｘに対応するデータ（キャッシュライン）をキャッシュしている旨を示す情報とを含む第１応答メッセージを、プロセッサ３００へ返す。この例では、実際にはメモリコントローラ１００の不揮発キャッシュメモリ１０には、領域Ｘに対応するキャッシュラインはキャッシュされていないので、コントローラ２０は、不揮発メインメモリ１１０の領域Ｘに記憶されたデータを読み出して不揮発キャッシュメモリ１０にキャッシュし（キャッシュしなくてもよい）、領域Ｘから読み出したデータと共に、領域Ｘに対応するデータをキャッシュしている旨を示す情報（実際にはキャッシュしていなくても）とを含む第１応答メッセージをプロセッサ３００へ返す。

[0044] この第１応答メッセージを受信したプロセッサ３００は、インターコネクト２００を介して接続されたプロセッサ（第２のプロセッサ）が領域Ｘに対応するキャッシュラインを保持していると判断するので、第１応答メッセージに含まれるキャッシュラインを第１キャッシュラインとして揮発キャッシュメモリ３３０に保持する。これで、プロセッサコア３２０は、領域Ｘに記憶されたデータ（値Ａ）をリードすることができる。この段階で、プロセッサ３００の第１キャッシュラインはＦｏｗａｒｄ状態になっていて、プロセッサ３００からは、インターコネクト２００を介して接続されたプロセッサ（第２のプロセッサ）が存在（実際に存在するのはメモリコントローラ１００）し、第２のプロセッサのキャッシュラインのうち、領域Ｘに対応するキャッシュライン（第２キャッシュライン）はＳｈａｒｅｄ状態になっている

と見えている。

[0045] 次に、ステップＳ２に対応する内容を説明する。まず、プロセッサコア３２０は、領域Ｘに対応するデータに対して、ライト命令でライトする（領域Ｘのアドレスを指定してライト命令を実行する）。領域Ｘに対応するキャッシュライン（第１キャッシュライン）は揮発キャッシュメモリ３３０に存在するので、第１キャッシュラインに対して値Ａ’をライトすることになる。ここで、第１キャッシュラインの状態がＦｏｗａｒｄ状態で、他のプロセッサが有する同じアドレスのキャッシュライン（領域Ｘに対応するキャッシュライン）を無効化する必要があるので、プロセッサ３００は、キャッシュコヒーレンスプロトコルに基づき、他のプロセッサに対して、領域Ｘに対応するキャッシュラインを無効にすることを要求するインバリデートリクエストをブロードキャストする。このインバリデートリクエストは、上述の第２の要求に相当する。上記で第１キャッシュラインをＦｏｗａｒｄ状態にするようにメモリコントローラ１００側から制御したのは、このインバリデートリクエストを引き起こすためである。つまり、このインバリデートリクエストが発行されることにより、領域Ｘに対応するデータを更新するライト命令が実行されることをメモリコントローラ１００が知ることができる。仮にメモリコントローラ１００の代わりに、キャッシュコヒーレンスプロトコルに従って動作する第２のプロセッサが存在する場合は、第２のプロセッサは、インバリデートリクエストを受信すると、その要求に従ってキャッシュラインのインバリデート処理を行う必要がある。ただし、本実施形態では、メモリコントローラ１００が領域Ｘに対応するデータの更新のタイミングを知るために、メモリコントローラ１００側がキャッシュコヒーレンスプロトコルを利用して意図的に発行させているインバリデートリクエストなので、メモリコントローラ１００はインバリデート処理を実際に行う必要は必ずしもない。そのため、メモリコントローラ１００のコントローラ２０は、インバリデートリクエストを受け取ると、インバリデート処理は行わずに、インバリデートリクエストに対する応答として、領域Ｘに対応するキャッシュラインを

キャッシュコヒーレンスプロトコルに従って無効化したように振る舞うために、無効にしたことを通知するための第２応答メッセージをプロセッサ３００に返す。この段階で、メモリコントローラ１００は、領域Ｘに対応するデータの更新のタイミングを知ることが出来たことになる。インバリデートリクエストの応答である第２応答メッセージを受け取ったプロセッサ３００は、他のプロセッサが有している同じアドレスのキャッシュライン（領域Ｘに対応するキャッシュライン）は無効化されたと判断し、第１キャッシュラインに対して実際のライト処理を行い、第１キャッシュラインの状態をＭｏｄｉｆｉｅｄ状態に変更する。

[0046] 次に、ステップＳ３に対応する内容を説明する。メモリコントローラ１００のコントローラ２０は、インバリデートリクエストの応答である第２応答メッセージを返した後、メモリコントローラ１００の代わりに第２のプロセッサが存在すると仮定した場合に、第２のプロセッサが第２キャッシュラインに対する書き込みを行おうとして失敗した時に送る、ライトミスによる第２スヌープリクエスト（領域Ｘに対応するデータを要求する情報）を送る。これは、ステップＳ２でプロセッサ３００により更新された領域Ｘに対応するデータを、インターコネクト２００を介して即座にメモリコントローラ１００側に引き寄せることを目的としている。第２スヌープリクエストは、上述の第３の要求に相当している。

[0047] 第２スヌープリクエストを受信したプロセッサ３００は、第２スヌープリクエストに対する応答として、第１キャッシュラインの更新済みのデータ（値Ａ’）を含む第３応答メッセージを送信し、第１キャッシュラインをＩＮＶＡＬＩＤ状態に変更する。仮に、メモリコントローラ１００の代わりに第２のプロセッサが存在する場合は、第２のプロセッサは、第２スヌープリクエストの応答である第３応答メッセージを受信したら、第３応答メッセージに含まれるデータ（値Ａ’）で不揮発メインメモリ１１０の内容を更新するが、ここでは、メモリコントローラ１００は、プロセッサ３００により更新済みの領域Ｘに対応するデータを、インターコネクト２００を介して即座に

メモリコントローラ１００側に引き寄せるために、意図的に第２スヌープリクエストを発行しているので、第３応答メッセージに含まれる更新済みのデータ（値Ａ’）で不揮発メインメモリ１１０の内容を更新することはしない（ライトミスを引き起こしたがそのライト自体も行わない）。そもそも、更新してしまうと、領域Ｘに対応するデータとして、複数のバージョンを作成することができなくなってしまう。ここで第３応答メッセージに含まれるデータこそが領域Ｘに対応するデータの更新内容なので、メモリコントローラ１００のコントローラ２０は、このデータを不揮発キャッシュメモリ１０に書き込む。より具体的には、例えば、不揮発キャッシュメモリ１０のうち第２キャッシュラインが保持される領域（不揮発キャッシュメモリ１０のうち領域Ｘに対応する領域）に書き込む。これにより、プロセッサ３００により更新済みの領域Ｘに対応するデータが不揮発キャッシュに書き込まれたことになる。これで図７に示したように、領域Ｘに対応するデータとして、値Ａを示す旧バージョンと、値Ａ’を示す最新バージョンが存在する状態になり、コミット処理の完了が可能な状態になる。本実施形態の情報処理装置１は、不揮発メインメモリ１１０上の第１データ（例えば領域Ｘに記憶されたデータ（値Ａ））をプロセッサ３００に読み出して更新し、更新済みのデータ（値Ａ’）を不揮発キャッシュメモリ１０に第２データとして保存した時点で、データを永続化するコミット処理を完了する。

[0048] 図９は、以上のメモリコントローラ１００のコントローラ２０の動作例を示すフローチャートである。コントローラ２０は、プロセッサ３００から上述の第１スヌープリクエストを受信すると、上述のインバリデートリクエストをプロセッサに発行させるために、第１スヌープリクエストに対する応答として、上述の第１応答メッセージを送信する（ステップＳ９０１）。次に、プロセッサ３００から上述のインバリデートリクエストを受信すると、コントローラ２０は、上述の第１スヌープリクエストで要求されたデータの更新がプロセッサ３００側で行われたことを検出する。そして、コントローラ２０は、実際にはインバリデート処理を行わずに、インバリデートリクエス

トに対する応答として、上述の第２応答メッセージを送信する（ステップＳ９０２）。次に、コントローラ２０は、更新済みのデータを即座にメモリコントローラ１００側に引き寄せるために、上述の第２のスヌープリクエストをプロセッサ３００へ送信する（ステップＳ９０３）。次に、コントローラ２０は、第２スヌープリクエストに対する応答として受信した上述の第３応答メッセージに含まれるデータ（プロセッサ３００により更新済みの領域Ｘに対応するデータ）を不揮発キャッシュメモリ１０に書き込む（ステップＳ９０４）。

[0049] また、図１０に示すように、ステップＳ３でライトミスによる第２スヌープリクエストを送る代わりに、リードミスによる第２スヌープリクエストを送ってもよい。

[0050] 次に、「永続データの永続化の完了を確認する手段」で永続データの永続化の完了を確認する手順の一例を説明する。上述の第２のスヌープリクエストに対する応答として、第３応答メッセージを送信した後、図１１に示すように、プロセッサ３００は、第１キャッシュラインに対する読み出しを行おうとして失敗した時に送る、リードミスによる第３スヌープリクエスト（領域Ｘに対応するデータ（キャッシュライン）を要求する情報）を送る。ここでは、上述したように第１キャッシュラインはＩＮＶＡＬＩＤ状態になっているので第３スヌープリクエストが発行され、その発行された第３スヌープリクエストはインターコネクト２００を経由してメモリコントローラ１００に到着する。第３スヌープリクエストを受信したコントローラ１０は、上述の第３応答メッセージに含まれるデータ（値Ａ’）の不揮発キャッシュメモリ１０への書き込みが完了した後に（つまり永続化が完了した後に）、第３スヌープリクエストに対する応答として、値Ａ’を示すデータを含む第４応答メッセージを送信する。プロセッサ３００が第４応答メッセージを受信すると、第１キャッシュラインは例えばＳｈａｒｅｄ状態になり、値Ａ’を示すデータの読み込みが完了すると、メモリコントローラ１００側で永続データの永続化が完了したことが保証されるので、コミット処理を完了すること

ができる。

[0051] 要するに、本実施形態のプロセッサ３００は、インターコネクト２００を介して接続された外部装置（ここではメモリコントローラ１００）からの、プロセッサ３００が更新した、不揮発メインメモリ１１０の何れかの領域Ｘに対応するデータの参照を行うために該領域Ｘに対応するデータを要求する第３の要求（この例では上述の第２スヌープリクエスト）に対するプロトコル（インターコネクト２００が保証するキャッシュコヒーレンスプロトコル）に基づいた応答として、キャッシュメモリ（この例では揮発キャッシュメモリ３３０）に記憶されたデータのうち該領域Ｘに対応するデータを送信した後に、その送信したデータの永続化が完了したことを確認するための確認要求を外部装置へ送信し、確認要求に対する応答に基づいて、その送信したデータの永続化が完了したことを確認することができる。ここでは、上記確認要求の一例として、上述の第３スヌープリクエストを挙げて説明したが、これに限られるものではない。

[0052] また、「Intel Architecture Instruction Set Extensions Programming Reference (319433-022, OCTOBER 2014) のCHAPTER 11: MEMORY INSTRUCTIONS」という非特許文献にある、メモリ書き込みの永続化を確認するＩｎｔｅｌのＰｃｏｍｍｉｔ命令を本実施形態で実装する場合には、図１１に示すような方法で実現することができる。

[0053] 図１２は、永続データの永続化の完了を確認する場合の情報処理装置１の動作例を示すフローチャートである。プロセッサ３００は、上述の第２のスヌープリクエストに対する応答として、上述の第３応答メッセージを送信した後、上述の第３スヌープリクエストを、インターコネクト２００を介してメモリコントローラ１００へ送信する（ステップＳ１２０１）。次に、第３スヌープリクエストを受信したメモリコントローラ１００のコントローラ２０は、第３スヌープリクエストを受信する前にプロセッサ３００から受信した第３応答メッセージに含まれるデータ（値Ａ’）の不揮発キャッシュメモリ１０への書き込みの完了を確認した後に、上述の第４応答メッセージを、

インターコネクト２００を介してプロセッサ３００へ送信する（ステップＳ１２０２）。第４応答メッセージを受信したプロセッサ３００は、読み出し可能な状態になった第１キャッシュラインからデータ（値Ａ’）の読み出しを完了すると、揮発キャッシュメモリ３３０からメモリコントローラ１００へ追い出したデータ（上述の第３応答メッセージに含まれるデータ）の永続化が完了したことを確認する（ステップＳ１２０３）。

[0054] また、永続データの永続化の完了を確認する別の手段として、上述の第２のスヌープリクエストに対する応答として、第３応答メッセージを送信した後、数サイクル～数十サイクルという決められた一定時間待つことで、不揮発キャッシュメモリ１０への書き込みが完了したことを確認する手段もある。つまり、第３応答メッセージを送信した後、一定時間が経過したときに、不揮発キャッシュメモリ１０への書き込みが完了したことを確認する手段であってもよい。一定時間とは、プロセッサ３００からメモリコントローラ１００へデータを送るのに必要な時間と、メモリコントローラ１００が不揮発キャッシュメモリ１０にデータを書き込む処理（永続化）を完了するのに必要な時間との合計よりも長い時間である。この時間は予め測定や設計時などに決定されてアプリケーションなどに提供される値である。この方法は後述の第２の実施形態についても同様に適用することができる。また、前述のＰｃｏｍｍｉｔ命令はこの方法で実装されてもよく、後述の第２の実施形態についても同様である。

[0055] 以上に説明したように、本実施形態のメモリコントローラ１００は、複数のキャッシュメモリ間のデータの不一致が起こらないようにするための手順を示すキャッシュコヒーレンスプロトコルを保証するインターコネクト２００を介してプロセッサ３００と接続される。そして、メモリコントローラ１００のコントローラ２０は、インターコネクト２００が保証するキャッシュコヒーレンスプロトコルを利用して、プロセッサ３００が、メモリコントローラ１００に接続された不揮発メインメモリ１１０の何れかの領域Ｘに対応するデータを更新したことを検出した後に、その更新済みのデータをメモリ

コントローラ１００へ送信させ、プロセッサ３００から受信した更新済みのデータを不揮発キャッシュメモリ１０に書き込む。更新前のデータは、まだ不揮発メインメモリ１１０上に存在するので、この時点で、領域Ｘに対応するデータとして、不揮発メインメモリ１１０上のデータと、不揮発キャッシュメモリ１０上のデータとが同時に存在する状態を、明示的に複数バージョンを作ることなく実現でき、アトミシティを保証できる。また、更新済みのデータを、不揮発メインメモリ１１０まで書き込むことなく不揮発キャッシュメモリ１０に書き込むだけで、コミット処理を完了できるので、より高速なコミット処理が可能になる。

[0056] また、上述したように、本実施形態のプロセッサ３００は、メモリコントーラ３００からの、プロセッサ３００が更新した、不揮発メインメモリ１１０の領域Ｘに対応するデータの参照を行うために領域Ｘに対応するデータを要求する第３の要求（この例では上述の第２スヌープリクエスト）に対するプロトコルに基づいた応答として、揮発キャッシュメモリ３３０に記憶されたデータのうち領域Ｘに対応するデータを送信した後に（この例では上述の第３応答メッセージを送信した後に）、その送信したデータの永続化が完了したことを確認するための確認要求（この例では上述の第３スヌープリクエスト）を送信し、確認要求に対する応答に基づいて、その送信したデータの永続化が完了したことを確認する。

[0057] すなわち、本実施形態では、キャッシュコヒーレンスプロトコルを利用して、プロセッサ３００が永続データを更新すると、更新されたデータを即座にメモリコントローラ１００に内蔵された不揮発キャッシュメモリ１０へ追い出す手段を提供するとともに、プロセッサ３００が、メモリコントローラ１００へ追い出したデータ（更新されたデータ）の永続化の完了を確認するための手段を提供することで、オーダリングを保証している。

[0058] 以上より、本実施形態によれば、アトミシティとオーダリングを保証しつつ、高速なコミット処理を実現することができる。したがって、高性能・高信頼なデータ永続化処理方式を実現することができる。

[0059] なお、上述の本実施形態では、コヒーレンスをとる単位としてキャッシュラインを用いたが、この限りではなく、ＯＳのページ単位や、ブロック単位などの任意の粒度であってもよいのは言うまでもない。また、プロセッサ３００はＣＰＵだけに限らず、ＧＰＵなどのアクセラレータなどであってもよいのも言うまでもない。後述の第２の実施形態でも同様である。

[0060] （第２の実施形態）

次に、第２の実施形態について説明する。上述の第１の実施形態と共通する部分については適宜に説明を省略する。本実施形態では、図１３に示すように、プロセッサ３００とメモリコントローラ１００は、既存のキャッシュコヒーレンスプロトコルとは異なる、コミット処理向けに最適化したキャッシュコヒーレンスプロトコル（以下、「最適化プロトコル」と称する場合がある）を保証するインターコネクト８００を介して接続される点で上述の第１の実施形態と相違する。インターコネクト８００は、プロセッサ３００の揮発キャッシュメモリ３３０と、メモリコントローラ１００の不揮発キャッシュメモリ１０の一貫性を維持するための最適化プロトコルを保証する。プロセッサ３００およびメモリコントローラ１００の基本的な構成は第１の実施形態と同様であるが、プロセッサ３００およびメモリコントローラ１００は、最適化プロトコルに従って動作する。

[0061] 本実施形態では、最適化プロトコルを利用して、上述の第１の実施形態と同様に、「永続データを高速に不揮発キャッシュメモリ１０に追い出して永続化する手段」と「永続データの永続化の完了を確認する手段」を提供する。

[0062] まず、「永続データを高速に不揮発キャッシュメモリ１０に追い出して永続化する手段」で永続データが永続化される手順の一例を説明する。図１３のステップＳ１１に示すように、まず、プロセッサ３００は、不揮発メインメモリ１１０の領域Ｘに記憶されたデータ（値Ａ）を更新するために、領域Ｘに記憶されたデータを読み出して揮発キャッシュメモリ３３０に保存する。より具体的には以下のとおりである。まず、プロセッサコア３２０が、不

揮発メインメモリ１１０の領域Ｘのアドレスを指定して、領域Ｘに記憶されたデータ（値Ａ）を読み出すリード命令を実行すると、この例では、揮発キャッシュメモリ３３０は、領域Ｘに対応するキャッシュラインをキャッシュしていないＩＮＶＡＬＩＤ状態になっているので、キャッシュミスが発生し、領域Ｘのアドレスのホームノード（この例ではメモリコントローラ１００）に対して、領域Ｘに対応するデータを要求する第４の要求が送られる。第４の要求を受信したメモリコントローラ１００のコントローラ２０は、不揮発メインメモリ１１０の領域Ｘに記憶されたデータ（値Ａ）を読み出して不揮発キャッシュメモリ１０にキャッシュする（第２キャッシュラインをキャッシュする）。なお、この例ではキャッシュしているがキャッシュはしなくてもよい。そして、コントローラ２０は、第４の要求に対する応答として、領域Ｘに対応するキャッシュライン（領域Ｘから読み出したデータ（値Ａ）を含む第２キャッシュラインの内容）を含む第５応答メッセージをプロセッサ３００へ返す。第５応答メッセージを受信したプロセッサ３００のプロセッサコア３２０は、第５応答メッセージに含まれるキャッシュラインを第１キャッシュラインとして揮発キャッシュメモリ３３０に保持する。これで、プロセッサコア３２０は、領域Ｘに記憶されたデータ（値Ａ）をリードすることができる。以上のステップＳ１１の終了時点で、第１キャッシュラインの状態は、次に永続化を行う書き込みが行われたらホームノードへ書き込み内容を転送する状態（FORWARD TO HOMENODE）になる。なお、永続化を行う必要がないデータは書き込み内容を転送する状態にする必要はなく、Ｅｘｃｌｕｓｉｖｅ状態などの、キャッシュラインの内容がメインメモリから読み出されたときと変更がない一般的な状態（転送が不要な状態）になる。

[0063] 次に、図１３のステップＳ１２に示すように、プロセッサ３００（アプリケーション）は、第１キャッシュラインに対して、値Ａ’をライト命令で書き込んで第１キャッシュラインに記憶されたデータをインプレース更新し、更新済みのデータをメモリコントローラ１００へ追い出す。より具体的には以下のとおりである。まず、プロセッサコア３２０は、領域Ｘに対応するデ

ータに対して、ライト命令でライトする（領域Ｘのアドレスを指定してライト命令を実行する）。領域Ｘに対応するキャッシュライン（第１キャッシュライン）は揮発キャッシュメモリ３３０に存在するので、第１キャッシュラインに対して値Ａ’をライトすることになる。ここでは、第１キャッシュラインの状態は「FORWARD TO HOMENODE」なので、第１キャッシュラインに対する書き込み内容が即座に揮発キャッシュメモリ３３０からメモリコントローラ１００に転送される。転送後、第１キャッシュラインの状態はＩＮＶＡＬＩＤ状態になる。そして、メモリコントローラ１００のコントローラ２０は、プロセッサ３００から受信した更新済みのデータを不揮発キャッシュメモリ１０に書き込む。より具体的には、コントローラ２０は、プロセッサ３００から受信した更新済みの第１キャッシュラインのコピーを、第２キャッシュラインに書き込んで反映させる。このときには、不揮発メインメモリ１１０の更新は行わない。これにより、領域Ｘに対応するデータとして、値Ａを示す旧バージョンと、値Ａ’を示す最新バージョンが同時に存在する状態になり、コミット処理の完了が可能な状態になる。

[0064] 図１４は、以上の情報処理装置１の動作例を示すフローチャートである。各ステップの具体的な内容は上述したとおりであるが、まずプロセッサ３００のプロセッサコア３２０は、上述の第４の要求を、インターコネクト８００を介してメモリコントローラ１００へ送信する（ステップＳ１３０１）。次に、第４の要求を受信したメモリコントローラ１００のコントローラ２０は、第２キャッシュラインをキャッシュし、第４の要求に対する応答として上述の第５応答メッセージを返す（ステップＳ１３０２）。次に、第５応答メッセージを受信したプロセッサ３００のプロセッサコア３２０は、第５応答メッセージに含まれるキャッシュライン（第２キャッシュラインの内容）を第１キャッシュラインとしてキャッシュする（ステップＳ１３０３）。次に、プロセッサコア３２０は、第１キャッシュラインに対するライトを実行し（ステップＳ１３０４）、更新済みのデータ（更新済みの第１キャッシュラインのコピー）を、インターコネクト８００を介してメモリコントローラ

１００へ転送する（ステップＳ１３０５）。次に、メモリコントローラ１００のコントローラ２０は、プロセッサ３００からインターコネクト８００を介して受信した更新済みのデータを、第２キャッシュラインに書き込んで反映させる（ステップＳ１３０６）。

[0065] 要するに、本実施形態では、プロセッサ３００とメモリコントローラ１００は、上述の最適化プロトコルを保証するインターコネクト８００を介して接続される。そして、プロセッサ３００が不揮発メインメモリ１１０の何れかの領域Ｘに対応するデータの更新を行うと、更新済みのデータが上述の最適化プロトコルに従ってメモリコントローラ１００へ転送され、コントローラ２０は、プロセッサ３００から受信した更新済みのデータを不揮発キャッシュメモリ１０に書き込む。

[0066] 次に、「永続データの永続化の完了を確認する手段」で永続データの永続化の完了を確認する手順の一例を説明する。プロセッサコア３２０（アプリケーション）は、以上のようにして第１キャッシュラインに対する書き込み内容をメモリコントローラ１００へ転送した後、その転送したデータの永続化の完了を確認するための確認要求を、インターコネクト８００を介してメモリコントローラ１００へ送信し、確認要求に対する応答に基づいて、その転送したデータの永続化が完了したことを確認する。この例では、プロセッサコア３２０は、第１キャッシュラインに対する書き込み内容をメモリコントローラ１００へ転送した後、第１キャッシュラインに対する読み出しを試み、失敗し、読み出しを失敗した時に送られる第５の要求（領域Ｘに対応するデータ（キャッシュライン）を要求する情報）を送る。ここでは、上述したように第１キャッシュラインはＩＮＶＡＬＩＤ状態になっているので第５の要求が発行され、その発行された第５の要求はインターコネクト８００を経由してメモリコントローラ１００に到着する。第５の要求を受信したコントローラ１０は、第５の要求を受信する前にプロセッサ３００から受信した更新済みの領域Ｘに対応するデータ（値Ａ’）の不揮発キャッシュメモリ１０への書き込みが完了した後に（つまり永続化が完了した後に）、第５の要

求に対する応答として、値Ａ’を示すデータを含む第６応答メッセージを送信する。プロセッサ３００が第６応答メッセージを受信すると、第１キャッシュラインは読み込むことが可能な状態になり、値Ａ’を示すデータの読み込みを完了することで、メモリコントローラ１００側で永続データの永続化が完了したことが保証されるので、コミット処理を完了することができる。この例では、永続化の確認用に領域Ｘを利用しているが、この限りではなく、例えば領域Ｘとは別の領域を確認用に利用してもよい。その場合は第１キャッシュラインに対する書き込み内容をメモリコントローラ１００に転送後に行われた第１キャッシュラインの状態をＩＮＶＡＬＩＤ状態にする処理は必要ない。

[0067] 図１５は、永続データの永続化の完了を確認する場合の情報処理装置１の動作例を示すフローチャートである。プロセッサ３００は、以上のようにして第１キャッシュラインに対する書き込み内容をメモリコントローラ１００へ転送した後、上述の第５の要求を、インターコネクト８００を介してメモリコントローラ１００へ送信する（ステップＳ１４０１）。次に、第５の要求を受信したメモリコントローラ１００のコントローラ２０は、第５の要求を受信する前にプロセッサ３００から受信した更新済みの領域Ｘに対応するデータ（値Ａ’）の不揮発キャッシュメモリ１０への書き込みの完了を確認した後に、上述の第６応答メッセージを、インターコネクト８００を介してプロセッサ３００へ送信する（ステップＳ１４０２）。第６応答メッセージを受信したプロセッサ３００は、読み出し可能な状態になった第１キャッシュラインからデータ（値Ａ’）の読み出しを完了すると、揮発キャッシュメモリ３３０からメモリコントローラ１００へ追い出したデータ（更新済みの領域Ｘに対応するデータ）の永続化が完了したことを確認する（ステップＳ１４０３）。永続データの永続化の完了を確認する別の方法として、上述の第１の実施形態で述べたように、第１キャッシュラインに対する書き込み内容をメモリコントローラ１００へ転送した後、上述の第５の要求を送信するかわりに、一定時間待つことで永続化を確認する方法を適用してもよい。

[0068] なお、前述のＩｎｔｅｌのＰｃｏｍｍｉｔ命令もこのように実装されてもよい。

[0069] 以上、本発明の実施形態を説明したが、上述の各実施形態は、例として提示したものであり、発明の範囲を限定することは意図していない。これら新規な実施形態は、その他の様々な形態で実施されることが可能であり、発明の要旨を逸脱しない範囲で、種々の省略、置き換え、変更を行うことができる。これら実施形態やその変形は、発明の範囲や要旨に含まれるとともに、請求の範囲に記載された発明とその均等の範囲に含まれる。

請求の範囲

[請求項1] メモリコントローラであって、

前記メモリコントローラに接続された不揮発メインメモリに記憶されたデータの一部を記憶する不揮発キャッシュメモリと、

前記不揮発キャッシュメモリへのデータの書き込みを制御するコントローラと、を有し、

前記メモリコントローラは、

複数のキャッシュメモリ間のデータの不一致が起こらないようにするための手順を示すプロトコルを保証するインターコネクトを介して処理装置と接続され、

前記コントローラは、

前記プロトコルを利用して、前記処理装置が、前記不揮発メインメモリの何れかの領域に対応するデータを更新したことを検出した後に、その更新済みのデータを前記メモリコントローラへ送信させ、その更新済みのデータを前記不揮発キャッシュメモリに書き込む、

メモリコントローラ。

[請求項2] 前記コントローラは、

前記処理装置から、前記不揮発メインメモリの何れかの領域に対応するデータを要求する第１の要求を受信した後に、前記領域に対応するデータと、該データを前記不揮発キャッシュメモリに保持していたことを示す情報とを含む第１応答メッセージを送信し、

前記第１応答メッセージを送信した後に、前記処理装置から、前記領域に対応するキャッシュラインを無効にすることを要求する第２の要求を受信したときに、前記処理装置が、前記領域に対応するデータを更新したことを検出する、

請求項１に記載のメモリコントローラ。

[請求項3] 前記コントローラは、

前記第１の要求により要求された前記領域に対応するデータを前記

不揮発キャッシュメモリに保持していない場合であっても、前記不揮発メインメモリの前記領域から読み出したデータと、該データを前記不揮発キャッシュメモリに保持していたことを示す情報とを含む前記第１応答メッセージを送信する、

請求項２に記載のメモリコントローラ。

[請求項4] 前記コントローラは、

前記第２の要求を受信した後に、前記領域に対応するキャッシュラインを無効にする処理は行わずに、前記第２の要求に対する応答として、前記領域に対応するキャッシュラインを無効にしたことを通知するための第２応答メッセージを送信し、

前記第２応答メッセージを送信した後に、前記領域に対応するデータを要求する第３の要求を送信し、

前記第３の要求に対する応答として、前記処理装置により更新済みのデータを受信し、その受信した更新済みのデータを前記不揮発キャッシュメモリに書き込む、

請求項２または３に記載のメモリコントローラ。

[請求項5] 前記第３の要求は、前記処理装置が更新した前記領域に対応するデータの更新を行うために前記領域に対応するデータを要求する情報であり、

前記コントローラは、前記第３の要求に対する応答として受信した更新済みのデータの更新は行わない、

請求項４に記載のメモリコントローラ。

[請求項6] 前記コントローラは、更新済みのデータを送付する手順を引き起こす第３の要求を前記処理装置に対して送信し、前記第３の要求に対する応答として送られてきたデータを前記不揮発キャッシュメモリに対して書き込み、

前記更新済みのデータを送付する手順は、前記処理装置内の更新済みのデータを前記メモリコントローラ側に転送させるために、前記プ

ロトコルで決められた手順である、

請求項１に記載のメモリコントローラ。

[請求項7] 前記プロトコルは、ＭＥＳＩＦプロトコル、ＱＰＩプロトコル、ＭＥＳＩプロトコル、ＭＯＥＳＩプロトコル、ＭＳＩプロトコルの何れかである、

請求項１に記載のメモリコントローラ。

[請求項8] メモリコントローラと、処理装置とを備える情報処理装置であって、

前記メモリコントローラは、

前記メモリコントローラに接続された不揮発メインメモリに記憶されたデータの一部を記憶する不揮発キャッシュメモリと、

前記不揮発キャッシュメモリへのデータの書き込みを制御するコントローラと、を有し、

複数のキャッシュメモリ間のデータの不一致が起こらないようにするための手順を示すプロトコルを保証するインターコネクトを介して前記処理装置と接続され、

前記コントローラは、

前記プロトコルを利用して、前記処理装置が、前記不揮発メインメモリの何れかの領域に対応するデータを更新したことを検出した後に、その更新済みのデータを前記メモリコントローラへ送信させ、その更新済みのデータを前記不揮発キャッシュメモリに書き込む、

情報処理装置。

[請求項9] 前記不揮発メインメモリ上の第１データを前記処理装置に読み出して更新し、更新済みのデータを前記不揮発キャッシュメモリに第２データとして保存した時点で、データを永続化するコミット処理を完了する、

請求項８に記載の情報処理装置。

[請求項10] 永続データの配置先となるメインメモリを設定するための設定手段

を備える、

請求項８に記載の情報処理装置。

[請求項11] 処理装置であって、

外部装置に接続された不揮発メインメモリに記憶されたデータの読み書きを行うプロセッサコアと、

前記不揮発メインメモリに記憶されたデータの一部を記憶するキャッシュメモリと、を有し、

前記処理装置は、

複数のキャッシュメモリ間のデータの不一致が起こらないようにするための手順を示すプロトコルを保証するインターコネクトを介して前記外部装置と接続され、

前記外部装置からの、前記処理装置が更新した、前記不揮発メインメモリの何れかの領域に対応するデータの参照を行うために前記領域に対応するデータを要求する第３の要求に対する前記プロトコルに基づいた応答として、前記キャッシュメモリに記憶されたデータのうち前記領域に対応するデータを送信した後に、その送信したデータの永続化が完了したことを確認するための確認要求を送信し、前記確認要求に対する応答に基づいて、その送信したデータの永続化が完了したことを確認する、

処理装置。

[請求項12] 処理装置であって、

外部装置に接続された不揮発メインメモリに記憶されたデータの読み書きを行うプロセッサコアと、

前記不揮発メインメモリに記憶されたデータの一部を記憶するキャッシュメモリと、を有し、

前記処理装置は、

複数のキャッシュメモリ間のデータの不一致が起こらないようにするための手順を示すプロトコルを保証するインターコネクトを介して

前記外部装置と接続され、

前記外部装置からの、前記処理装置が更新した、前記不揮発メインメモリの何れかの領域に対応するデータの参照を行うために前記領域に対応するデータを要求する第３の要求に対する前記プロトコルに基づいた応答として、前記キャッシュメモリに記憶されたデータのうち前記領域に対応するデータを送信した後、一定時間を経過したときに、その送信したデータの永続化が完了したことを確認する、

処理装置。

[請求項13] 処理装置と、前記処理装置とインターコネクトで接続されるメモリコントローラと、を備え、

前記処理装置は、

前記メモリコントローラに接続された不揮発メインメモリに記憶されたデータの読み書きを行い、前記不揮発メインメモリに記憶されたデータの一部を記憶するキャッシュメモリを有し、

前記メモリコントローラは、

前記不揮発メインメモリに記憶されたデータの一部を記憶する不揮発キャッシュメモリと、前記不揮発キャッシュメモリへのデータの書き込みを制御するコントローラと、を有し、

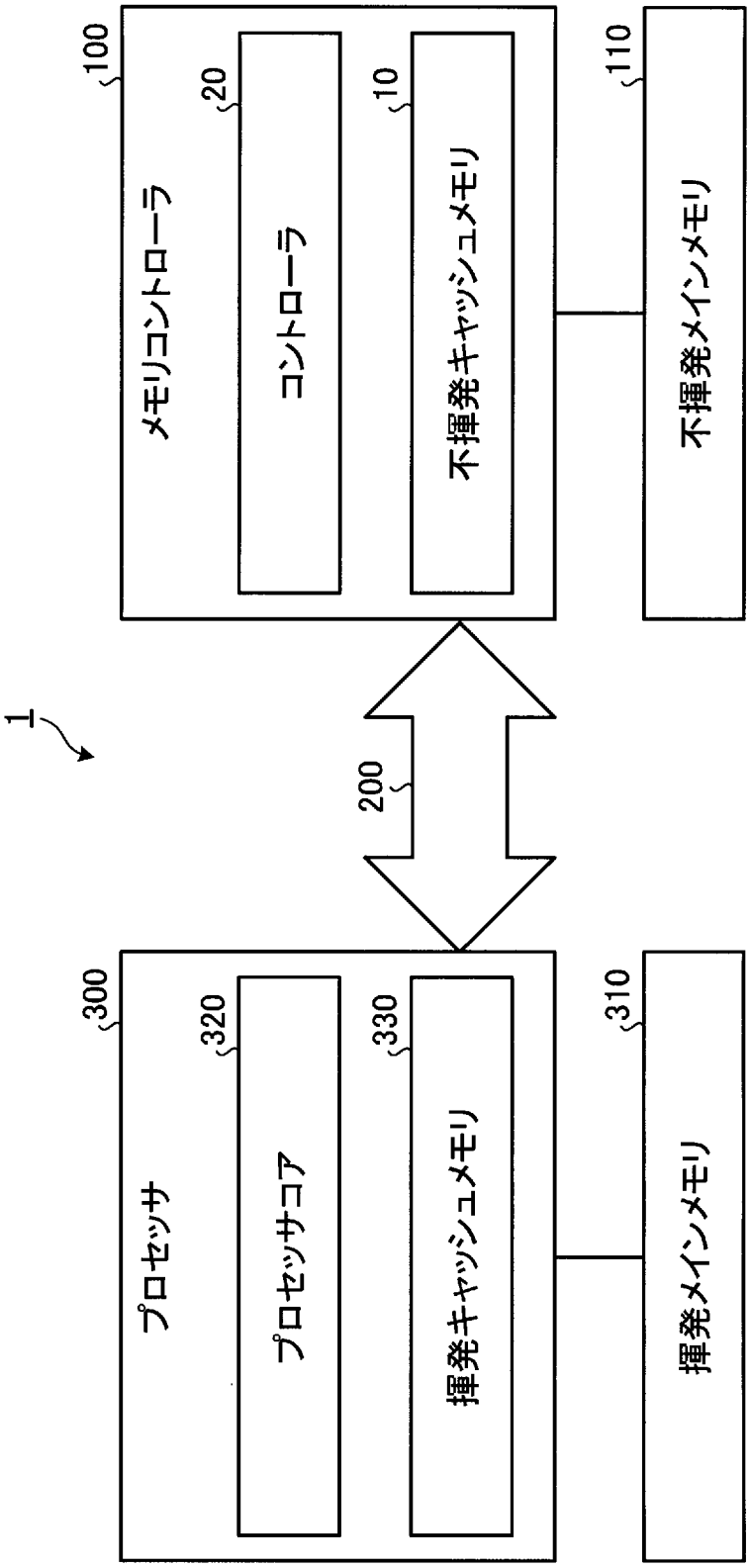
前記インターコネクトは、前記処理装置の前記キャッシュメモリと、前記メモリコントローラの前記不揮発キャッシュメモリの一貫性を維持するためのプロトコルを保証し、

前記処理装置が前記不揮発メインメモリの何れかの領域に対応するデータの更新を行うと、更新済みのデータが前記プロトコルに従って前記メモリコントローラへ転送され、

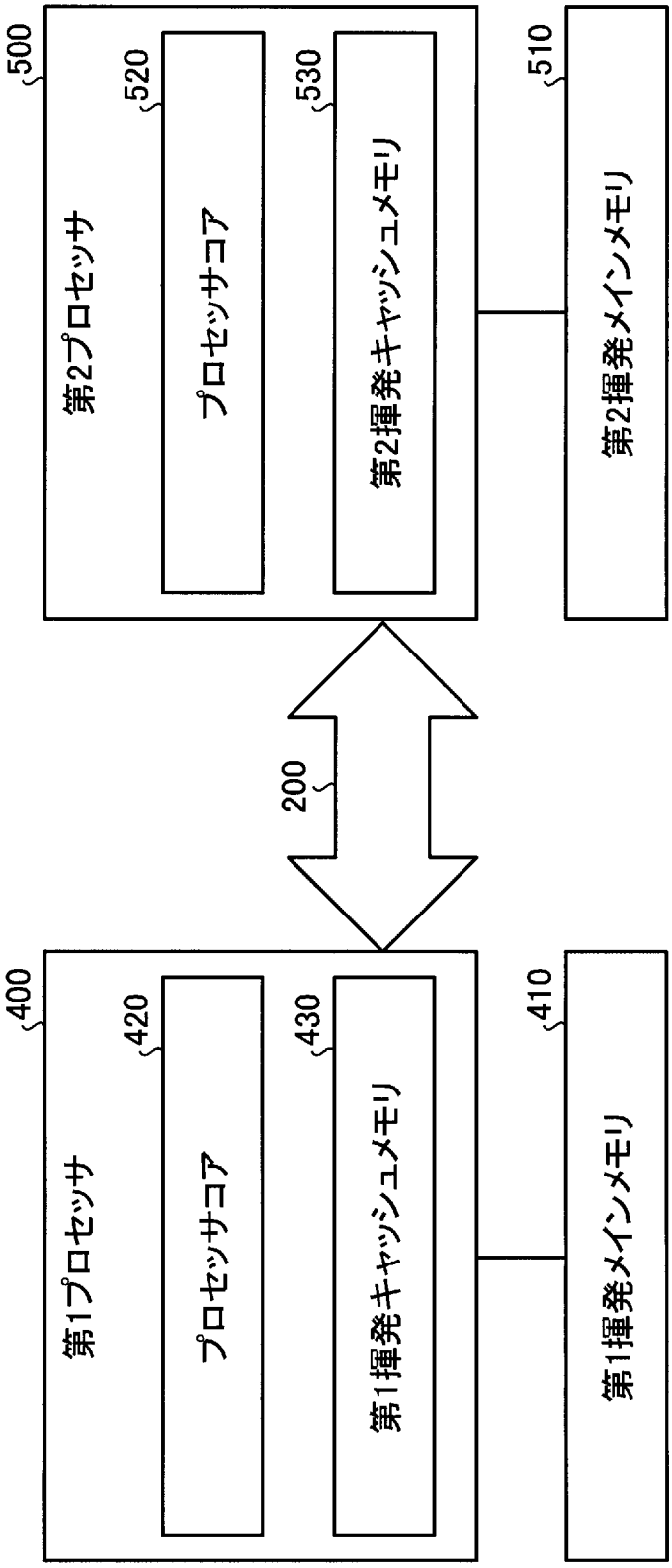
前記コントローラは、前記処理装置から受信した更新済みのデータを前記不揮発キャッシュメモリに書き込む、

情報処理装置。

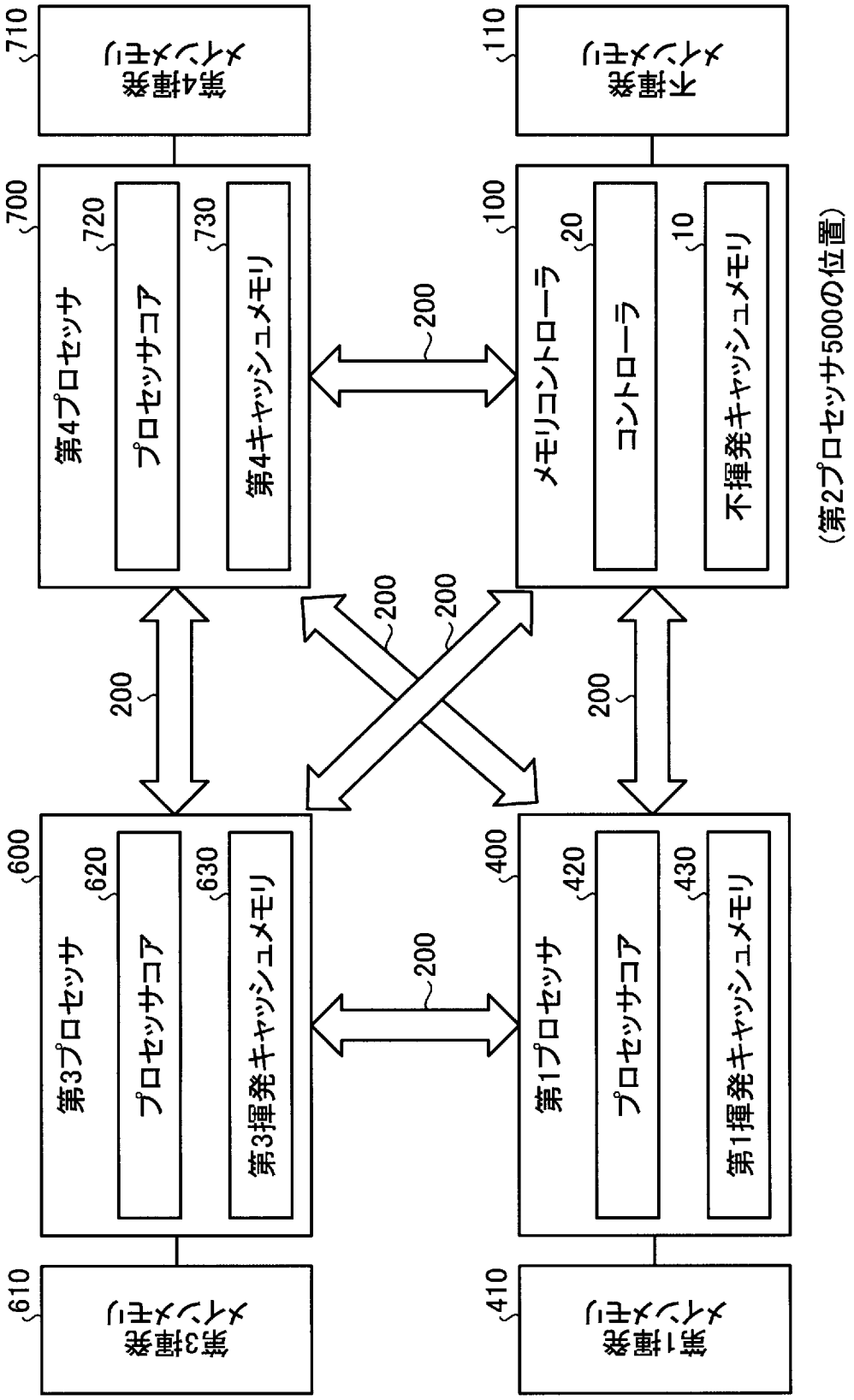
[図1]


1
100
メモリコントローラ
20
コントローラ
10
不揮発キャッシュメモリ
110
不揮発メインメモリ
200
300
プロセッサ
320
プロセッサコア
330
揮発キャッシュメモリ
310
揮発メインメモリ

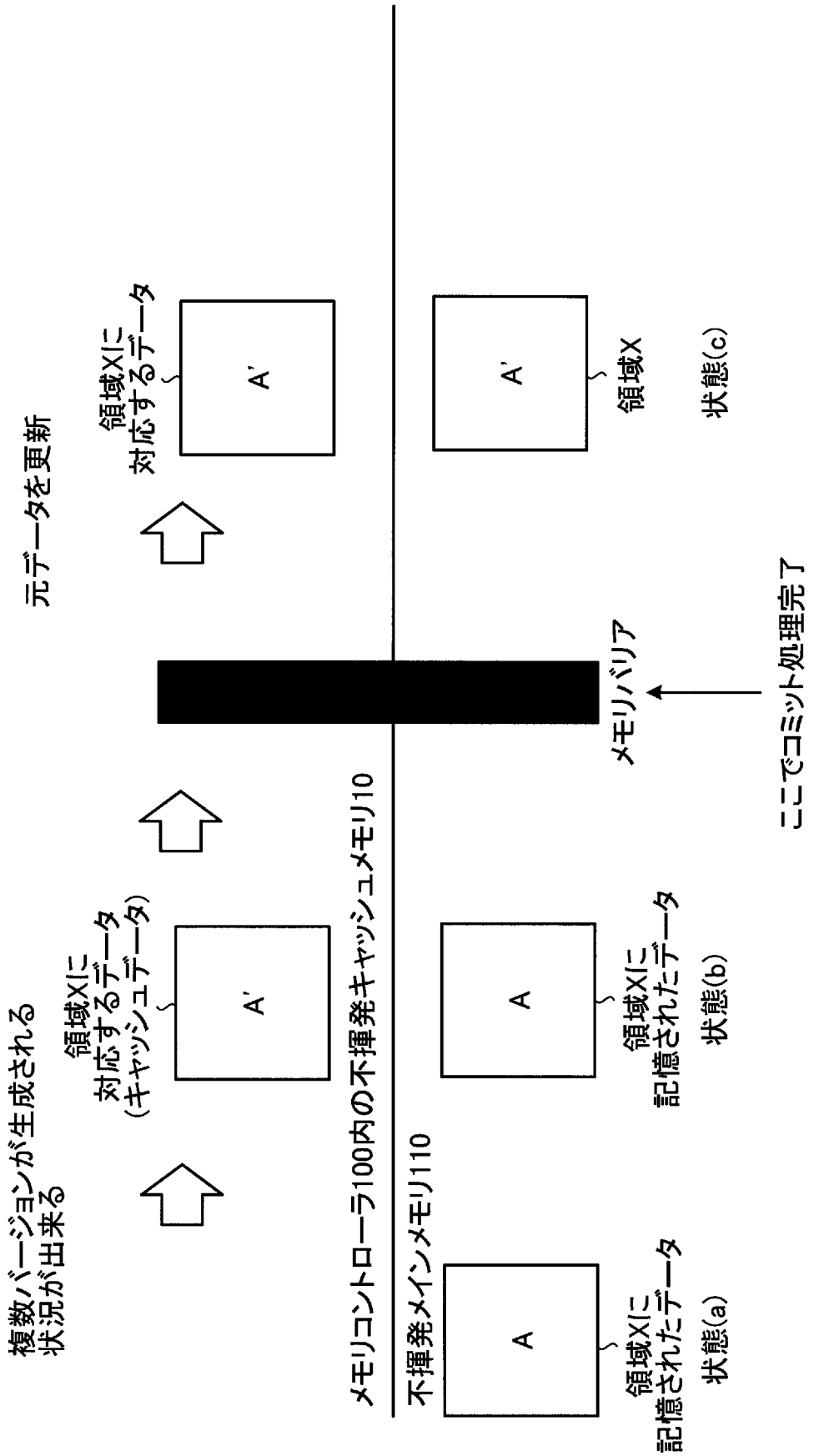
[図2]


500
第2プロセッサ
520
プロセッサコア
530
第2揮発キャッシュメモリ
510
第2揮発メインメモリ
200
400
第1プロセッサ
420
プロセッサコア
430
第1揮発キャッシュメモリ
410
第1揮発メインメモリ

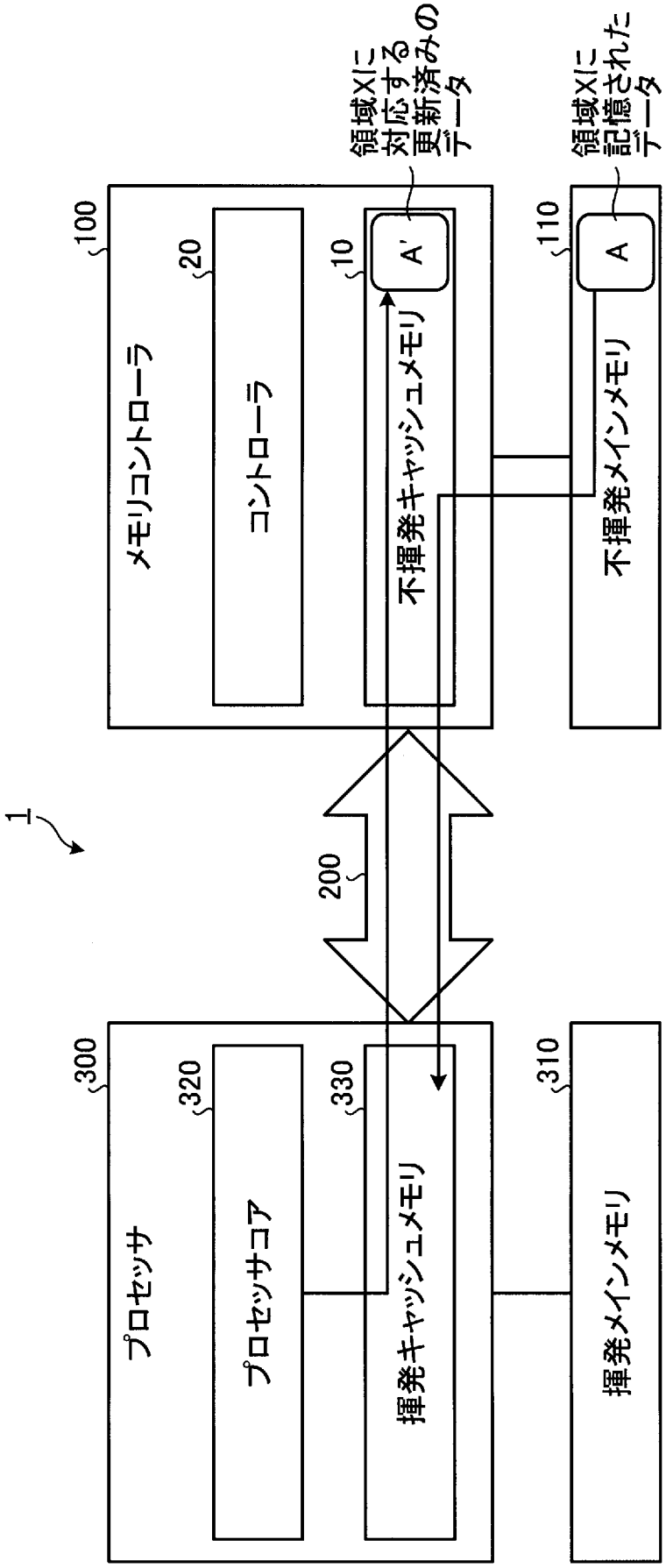
[図3]


710
第4揮発メモリ
110
不揮発メモリ
700
第4プロセッサ
720
プロセッサコア
730
第4キャッシュメモリ
200
100
メモリコントローラ
20
コントローラ
10
不揮発キャッシュメモリ
(第2プロセッサ500の位置)
200
200
200
200
600
第3プロセッサ
620
プロセッサコア
630
第3揮発キャッシュメモリ
200
400
第1プロセッサ
420
プロセッサコア
430
第1揮発キャッシュメモリ
610
第3揮発メモリ
410
第1揮発メモリ

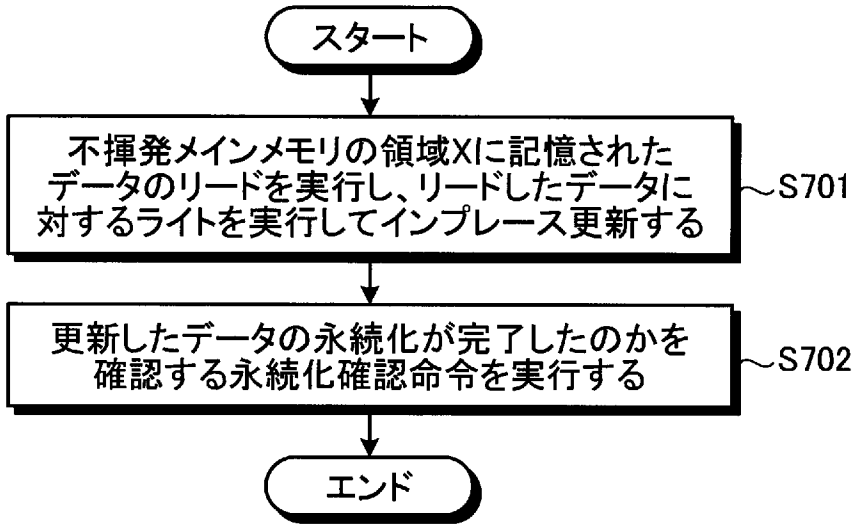
[図4]


複数バージョンが生成される
状況が出来る
領域Xに
対応するデータ
(キャッシュデータ)
A'
元データを更新
領域Xに
対応するデータ
A'
メモリコントローラ100内の不揮発キャッシュメモリ10
不揮発メインメモリ110
A
領域Xに
記憶されたデータ
状態(a)
A
領域Xに
記憶されたデータ
状態(b)
メモリバリア
ここでコミット処理完了
A'
領域X
状態(c)

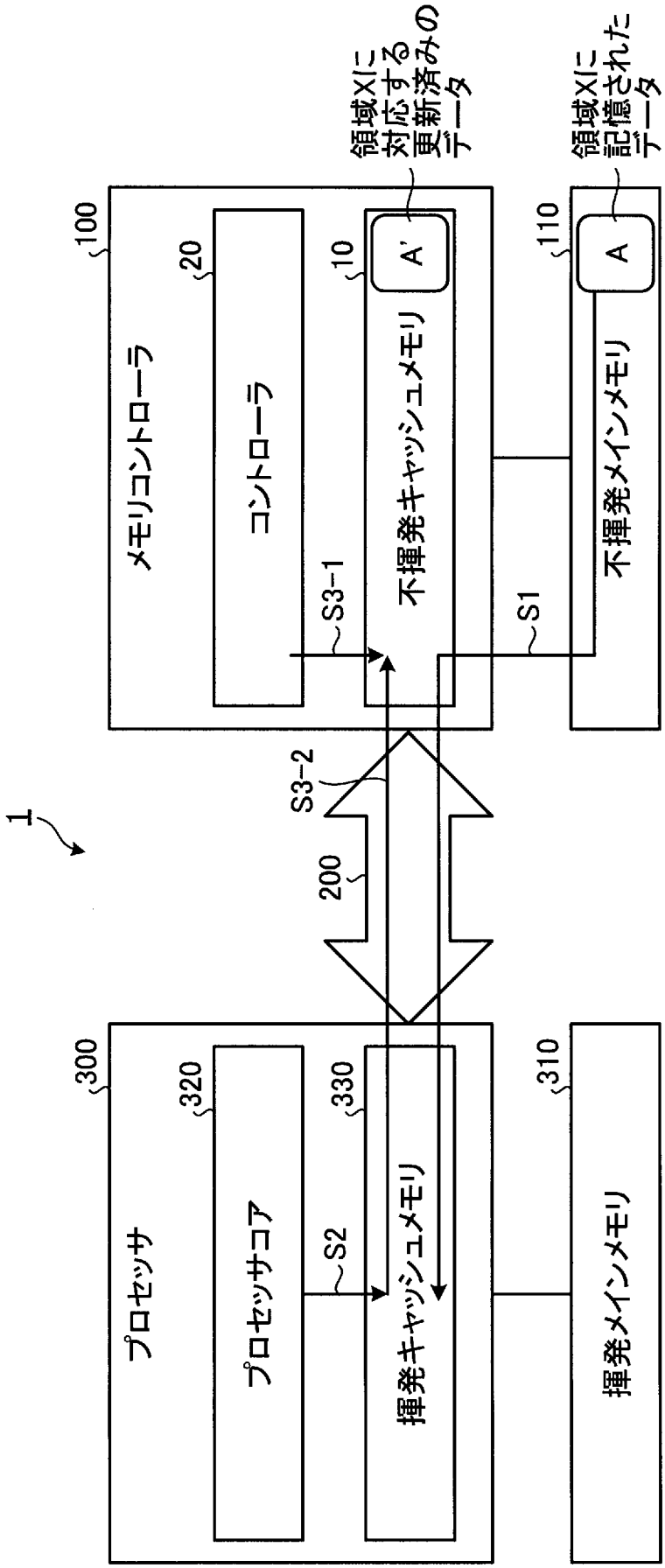
[図5]


1
100
メモリコントローラ
20
コントローラ
10
不揮発キャッシュメモリ
A'
領域Xに対応する更新済みのデータ
110
不揮発メインメモリ
A
領域Xに記憶されたデータ
200
300
プロセッサ
320
プロセッサコア
330
揮発キャッシュメモリ
310
揮発メインメモリ

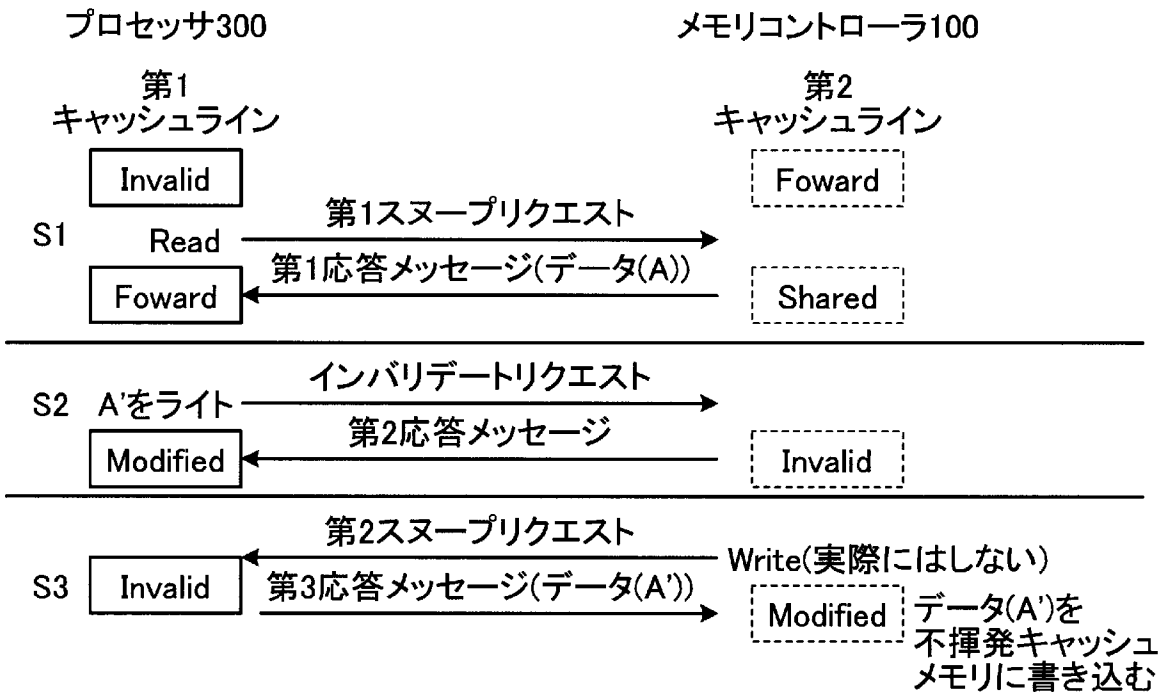
[図6]


スタート
不揮発メインメモリの領域Xに記憶されたデータのリードを実行し、リードしたデータに対するライトを実行してインプレース更新する
S701
更新したデータの永続化が完了したのかを確認する永続化確認命令を実行する
S702
エンド

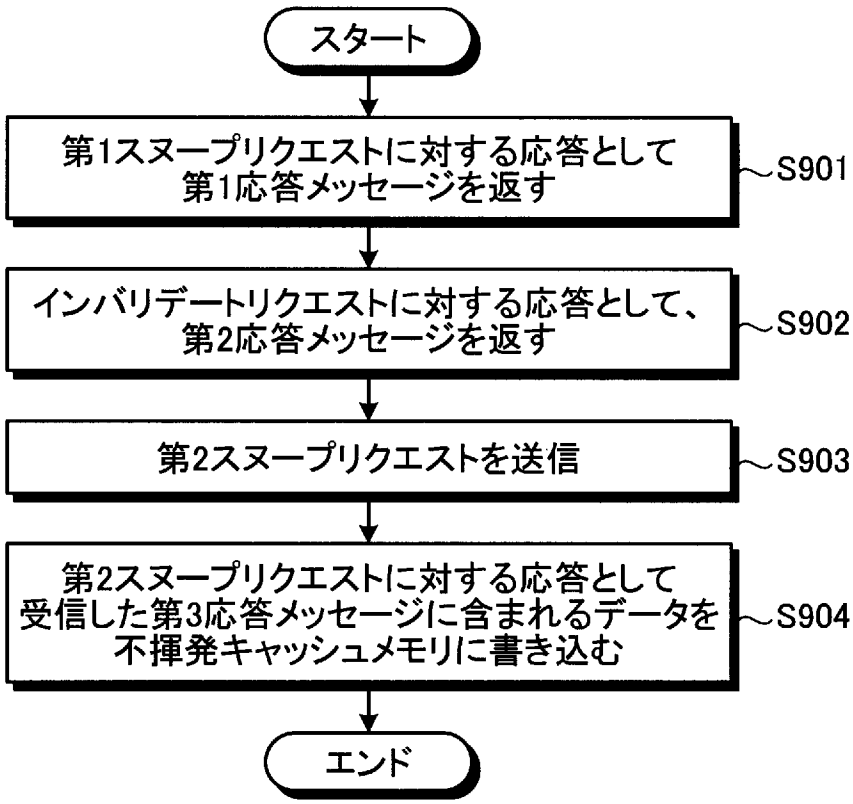
[図7]


1
100
メモリコントローラ
20
コントローラ
10
不揮発キャッシュメモリ
A'
領域Xに対応する更新済みのデータ
110
不揮発メインメモリ
A
領域Xに記憶されたデータ
S3-1
S1
S3-2
200
300
プロセッサ
320
プロセッサコア
330
揮発キャッシュメモリ
S2
310
揮発メインメモリ

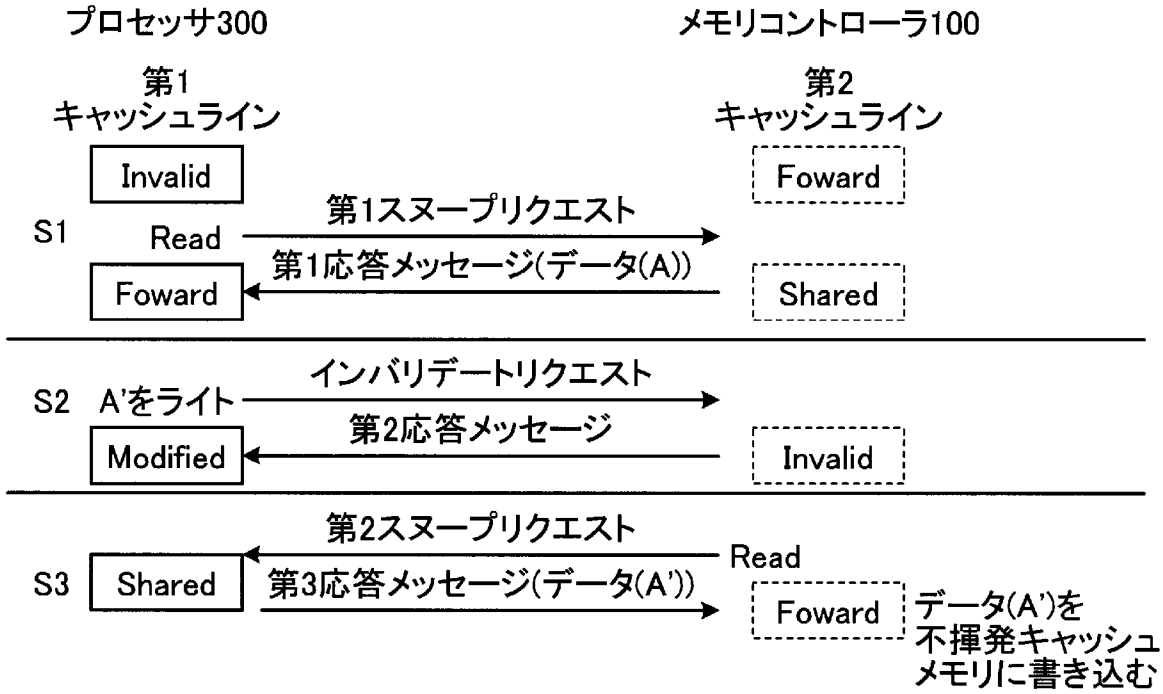
[図8]


プロセッサ300
メモリコントローラ100
第1
キャッシュライン
第2
キャッシュライン
Invalid
Foward
S1
Read
第1スヌープリクエスト
第1応答メッセージ(データ(A))
Foward
Shared
S2
A'をライト
インバリデートリクエスト
第2応答メッセージ
Modified
Invalid
第2スヌープリクエスト
Write(実際にはしない)
S3
Invalid
第3応答メッセージ(データ(A'))
Modified
データ(A')を
不揮発キャッシュ
メモリに書き込む

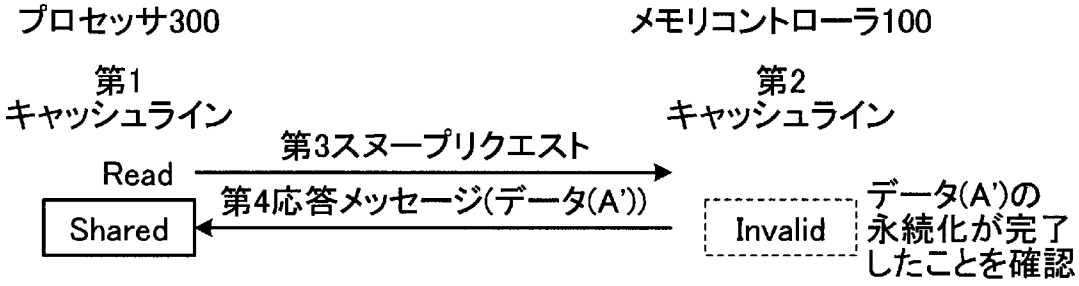
[図9]


スタート
第1スヌープリクエストに対する応答として
第1応答メッセージを返す
S901
インバリデートリクエストに対する応答として、
第2応答メッセージを返す
S902
第2スヌープリクエストを送信
S903
第2スヌープリクエストに対する応答として
受信した第3応答メッセージに含まれるデータを
不揮発キャッシュメモリに書き込む
S904
エンド

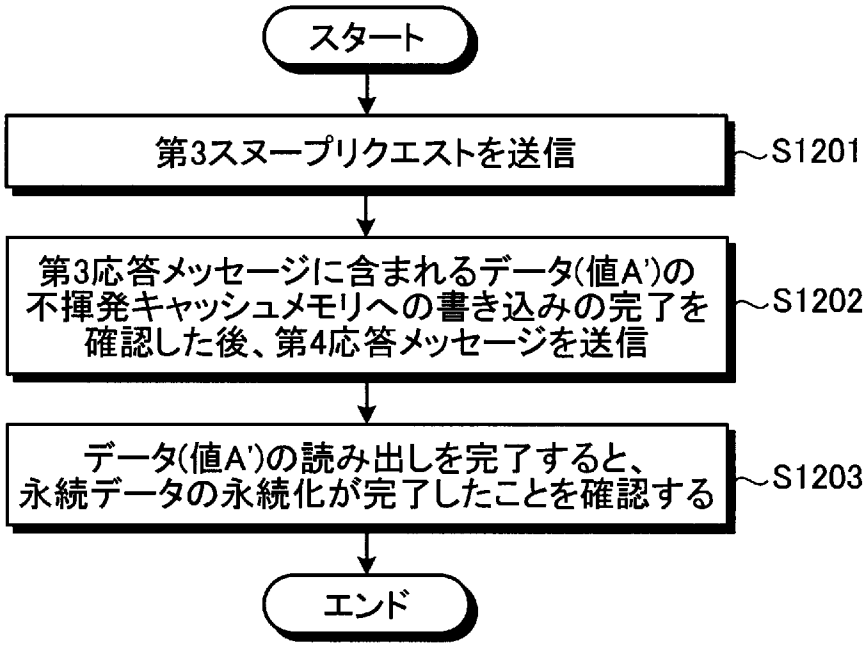
[図10]


プロセッサ300
メモリコントローラ100
第1
キャッシュライン
第2
キャッシュライン
Invalid
Foward
S1
Read
第1スヌープリクエスト
第1応答メッセージ(データ(A))
Foward
Shared
S2
A'をライト
インバリデートリクエスト
第2応答メッセージ
Modified
Invalid
S3
第2スヌープリクエスト
Read
Shared
第3応答メッセージ(データ(A'))
Foward
データ(A')を
不揮発キャッシュ
メモリに書き込む

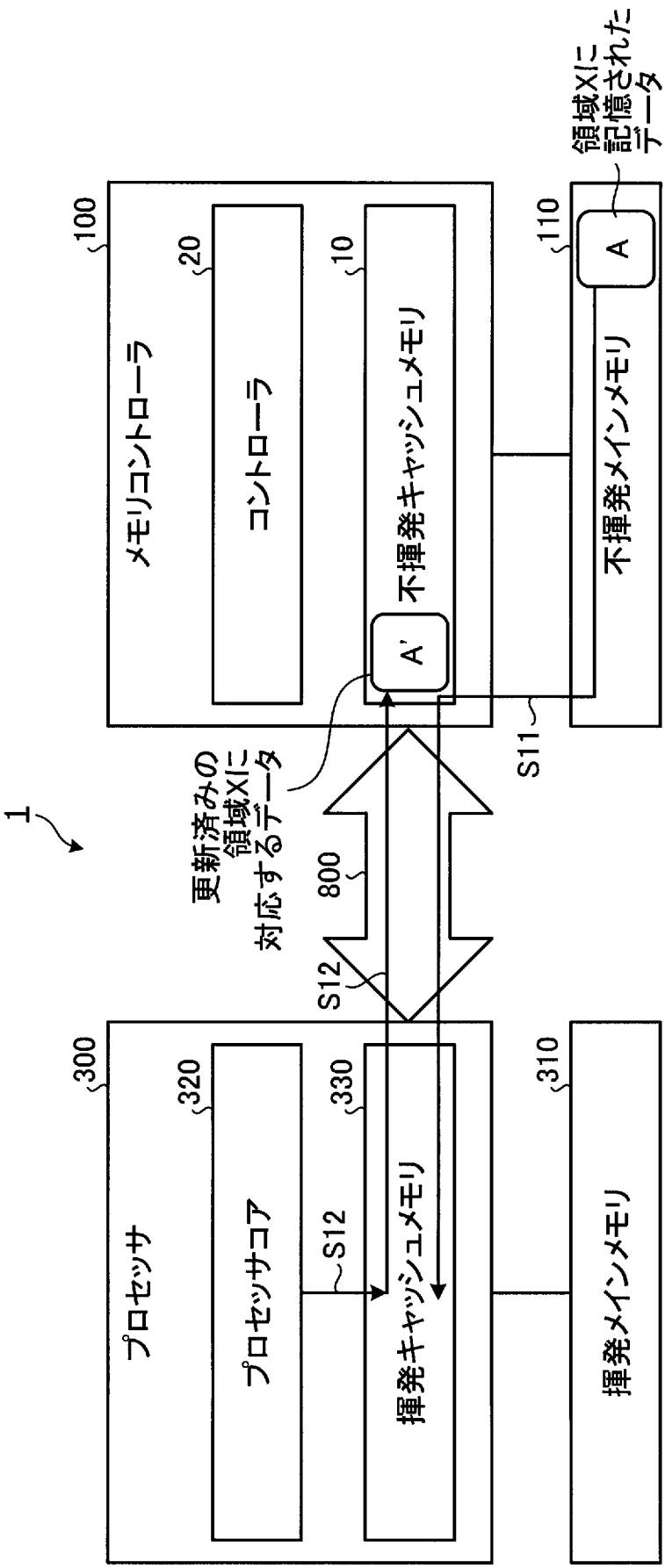
[図11]


プロセッサ300
メモリコントローラ100
第1
キャッシュライン
第2
キャッシュライン
第3スヌープリクエスト
Read
第4応答メッセージ(データ(A'))
Shared
Invalid
データ(A')の
永続化が完了
したことを確認

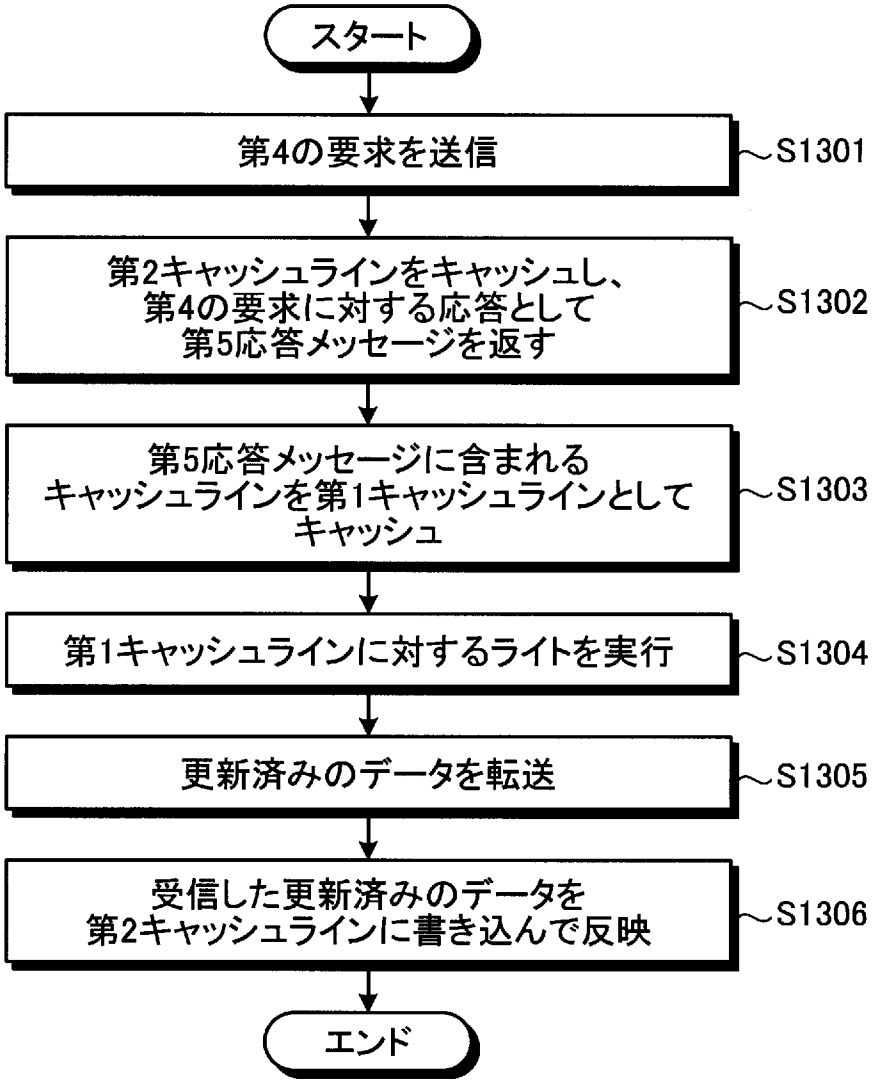
[図12]


スタート
第3スヌープリクエストを送信
S1201
第3応答メッセージに含まれるデータ(値A')の不揮発キャッシュメモリへの書き込みの完了を確認した後、第4応答メッセージを送信
S1202
データ(値A')の読み出しを完了すると、永続データの永続化が完了したことを確認する
S1203
エンド

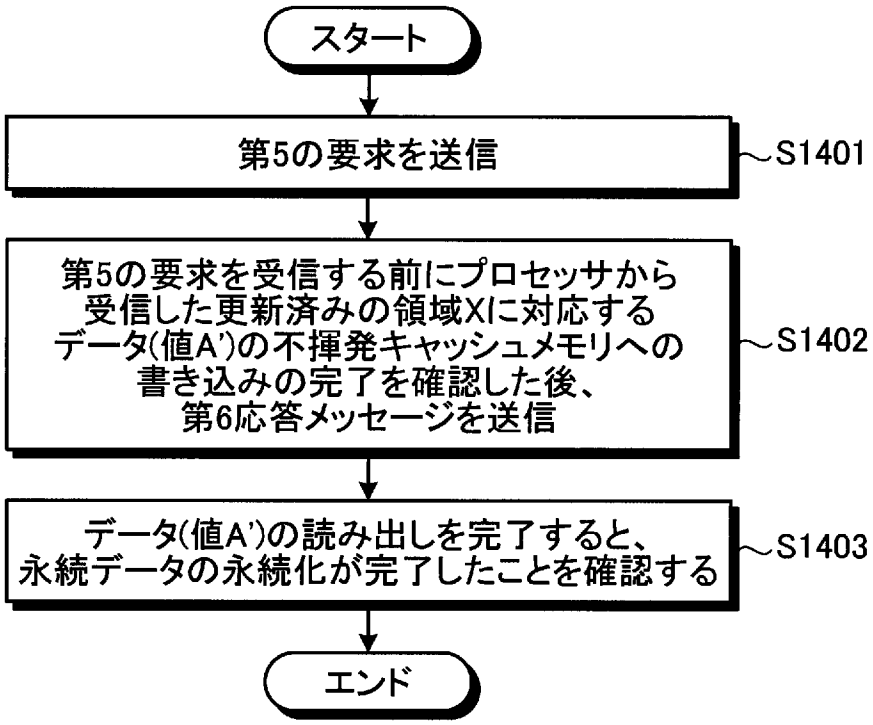
[図13]


1
100
メモリコントローラ
20
コントローラ
10
不揮発キャッシュメモリ
A'
110
不揮発メインメモリ
A
領域Xに記憶されたデータ
S11
更新済みの領域Xに対応するデータ
800
S12
300
プロセッサ
320
プロセッサコア
330
揮発キャッシュメモリ
S12
310
揮発メインメモリ

[図14]


スタート
第4の要求を送信
S1301
第2キャッシュラインをキャッシュし、
第4の要求に対する応答として
第5応答メッセージを返す
S1302
第5応答メッセージに含まれる
キャッシュラインを第1キャッシュラインとして
キャッシュ
S1303
第1キャッシュラインに対するライトを実行
S1304
更新済みのデータを転送
S1305
受信した更新済みのデータを
第2キャッシュラインに書き込んで反映
S1306
エンド

[図15]


スタート
第5の要求を送信
S1401
第5の要求を受信する前にプロセッサから
受信した更新済みの領域Xに対応する
データ(値A')の不揮発キャッシュメモリへの
書き込みの完了を確認した後、
第6応答メッセージを送信
S1402
データ(値A')の読み出しを完了すると、
永続データの永続化が完了したことを確認する
S1403
エンド

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2015/070438

A. CLASSIFICATION OF SUBJECT MATTER
G06F12/08(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F12/08

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Jitsuyo Shinan Toroku Koho	1996-2015
Kokai Jitsuyo Shinan Koho	1971-2015	Toroku Jitsuyo Shinan Koho	1994-2015

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2014/142908 A1 (HEWLETT-PACKARD DEVELOPMENT CO., L.P.), 18 September 2014 (18.09.2014), paragraphs [0032] to [0039]; fig. 1B (Family: none)	1-13
A	Sayuri SHIOHIRA, Tomaranai Server o Jitsugen suru Intel Itanium no Technology, Monthly ASCII. Technologies, 24 July 2010 (24.07.2010), vol.15, no.9, pages 50 to 61	1-13

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
13 August 2015 (13.08.15)	25 August 2015 (25.08.15)

Name and mailing address of the ISA/	Authorized officer
Japan Patent Office 3-4-3,Kasumigaseki,Chiyoda-ku, Tokyo 100-8915,Japan	Telephone No.

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２０１５／０７０４３８

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F12/08(2006.01)i

Ｂ．　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int.Cl.　G06F12/08

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２０１５年
日本国実用新案登録公報	１９９６－２０１５年
日本国登録実用新案公報	１９９４－２０１５年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

Ｃ．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求項の番号
A	WO 2014/142908 A1 (HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.) 2014.09.18, 段落［００３２］－［００３９］、図１Ｂ (ファミリーなし)	1-13
A	潮平さゆり，止まらないサーバーを実現する Intel Itanium のテクノロジー，月刊アスキードットテクノロジーズ，2010.07.24，第 15 巻，第９号，p.50-61	1-13

☐　Ｃ欄の続きにも文献が列挙されている。

☐　パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
１３．０８．２０１５	２５．０８．２０１５

国際調査機関の名称及びあて先
日本国特許庁（ＩＳＡ／ＪＰ）
郵便番号１００－８９１５
東京都千代田区霞が関三丁目４番３号

特許庁審査官（権限のある職員）　５Ｕ　４２２６
後藤　彰
電話番号　０３－３５８１－１１０１　内線　３５６５