(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5573829号
(P5573829)
(45)発行日　平成26年8月20日(2014.8.20)　(24)登録日　平成26年7月11日(2014.7.11)

(51)Int.Cl.　　F I
G06F 12/06 (2006.01)　　G06F 12/06 530A
G06F 12/08 (2006.01)　　G06F 12/08 531E

請求項の数 8　(全 37 頁)

(21)出願番号　特願2011-279022(P2011-279022)
(22)出願日　平成23年12月20日(2011.12.20)
(65)公開番号　特開2013-130976(P2013-130976A)
(43)公開日　平成25年7月4日(2013.7.4)
審査請求日　平成24年7月20日(2012.7.20)

前置審査

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100089118
弁理士　酒井　宏明
(72)発明者　鯉沼　秀之
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
(72)発明者　岡田　誠之
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
(72)発明者　杉崎　剛
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
最終頁に続く

(54)【発明の名称】情報処理装置およびメモリアクセス方法

(57)【特許請求の範囲】
【請求項１】
それぞれがプロセッサ及び記憶装置を備える複数のノードと、前記複数のノード間を接続するインターコネクトとを有する情報処理装置であって、
前記ノードの各々は、
各ノードに備えられたプロセッサを識別するプロセッサ識別情報と、当該プロセッサ識別情報が示すプロセッサを備えるノードの記憶装置に割当てられた物理アドレスとを対応付けて記憶する記憶部と、
論理アドレスと物理アドレスとの変換を行う第１変換部と、
物理アドレスを、前記記憶部に当該物理アドレスと対応付けて記憶されたプロセッサ識別情報に変換する第２変換部と、
前記物理アドレス及び前記プロセッサ識別情報を含み、当該プロセッサ識別情報が識別するプロセッサを備えるノードの記憶装置からデータの読出しを指示する読出要求、または、前記物理アドレス及び前記プロセッサ識別情報を含み、当該プロセッサ識別情報が識別するプロセッサを備えるノードの記憶装置へのデータの書込みを指示する書込要求を送信する送信部と、
他のノードから前記インターコネクトを介して送信された読出要求または書込要求を受信する受信部と、
前記受信部により受信された前記読出要求または前記書込要求に含まれる物理アドレスに基づいて、自ノードの記憶装置のデータ格納領域のうち、自ノード内のアクセスに用い

られるローカル領域と、他のノードからもアクセス可能な共有領域とのいずれの領域へのアクセスであるかを判定するローカル判定部と、
各ノードへの前記プロセッサまたは前記記憶装置の追加や削除に応じて、前記記憶部に記憶されたプロセッサ識別情報と物理アドレスとの対応付けを書換える制御部と
を有することを特徴とする情報処理装置。
【請求項２】
前記ノードの各々は、
前記ノードの各々が備える記憶装置の物理アドレスのうち、所定の位置のビットが同一の値である物理アドレスを前記共有領域に割当てるとともに、前記所定の位置のビットが前記共有領域に割当てた物理アドレスとは異なる値である物理アドレスを前記ローカル領域に割当し、
前記ローカル判定部は、前記読出要求または前記書込要求に含まれる物理アドレスのうち、前記所定の位置のビットの値に応じて、前記ローカル領域と前記共有領域とのいずれの領域へのアクセスであるかを判定する
ことを特徴とする請求項１に記載の情報処理装置。
【請求項３】
前記ノードの各々は、
前記ノードの各々が備える記憶装置の全物理アドレスを前記ローカル領域と前記共有領域とに分けて割り当て、
前記ローカル判定部は、前記読出要求または前記書込要求に含まれる物理アドレスのうち、最上位のビットの値に応じて、前記ローカル領域と前記共有領域とのいずれの領域へのアクセスであるかを判定する
ことを特徴とする請求項１または２に記載の情報処理装置。
【請求項４】
前記ノードの各々は、
前記ローカル判定部が前記ローカル領域へのアクセスであると判定した場合には、前記読出要求または前記書込要求の送信元となるノードに対してアクセスを許可しない旨の否定応答を送信することを特徴とする請求項１－３のいずれか１つに記載の情報処理装置。
【請求項５】
前記ノードの各々は、前記他のノードから前記読出要求または前記書込要求を受信した場合には、当該読出要求または当該書込要求の対象となるデータにつき、自ノードが備える記憶装置のデータをキャッシュした他のノードを示すディレクトリを用いて、自ノードが備える記憶装置上のデータと他のノードがキャッシュしたデータとの同一性を保持するディレクトリ制御部を有することを特徴とする請求項１－４のいずれか１つに記載の情報処理装置。
【請求項６】
前記ノードの各々は、
前記記憶装置からデータをキャッシュするキャッシュメモリと、
キャッシュミスが発生した場合は、キャッシュミスした物理アドレスが他のノードが有する記憶装置の物理アドレスであるか否かを判別する判別部とをさらに有し、
前記第２変換部は、前記キャッシュミスした物理アドレスが他のノードが有する記憶装置の物理アドレスであると前記ローカル判定部が判定した場合は、当該物理アドレスをプロセッサ識別情報に変換することを特徴とする請求項１－５のいずれか１つに記載の情報処理装置。
【請求項７】
前記ノードの各々が備えるプロセッサにより実行される各ＯＳは、アプリケーションから前記共有領域の獲得が要求された場合には、当該アプリケーションが使用する論理アドレスと、前記共有領域に割当てられる物理アドレスとの変換を行うように前記第１変換部を設定することを特徴とする請求項１－６のいずれか１つに記載の情報処理装置。
【請求項８】

それぞれがプロセッサと、記憶装置と、各ノードに備えられたプロセッサを識別するプロセッサ識別情報と当該プロセッサ識別情報が示すプロセッサを備えるノードの記憶装置に割当てられた物理アドレスとを対応付けて記憶する記憶部とを備え、インターコネクトを介して他ノードと接続されるノードが実行するメモリアクセス方法であって、

アクセス対象の論理アドレスと物理アドレスとの変換を行い、

前記物理アドレスを前記記憶部に当該物理アドレスと対応付けて記憶されたプロセッサ識別情報に変換し、

前記物理アドレス及び前記プロセッサ識別情報を含み、当該プロセッサ識別情報が識別するプロセッサを備えるノードの記憶装置からデータの読出しを指示する読出要求、または、前記物理アドレス及び前記プロセッサ識別情報を含み、当該プロセッサ識別情報が識別するプロセッサを備えるノードの記憶装置へのデータの書込みを指示する書込要求を送信し、

他のノードから前記インターコネクトを介して送信された読出要求または書込要求を受信した場合は、当該読出要求または書込要求に含まれる物理アドレスに基づいて、自ノードの記憶装置のデータ格納領域のうち、自ノード内のアクセスに用いられるローカル領域と、他ノードからもアクセス可能な共有領域とのいずれの領域へのアクセスであるかを判定し、

各ノードへの前記プロセッサまたは前記記憶装置の追加や削除に応じて、前記記憶部に記憶されたプロセッサ識別情報と物理アドレスとの対応付けを書換える

処理を実行することを特徴とするメモリアクセス方法。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、情報処理装置およびメモリアクセス方法に関する。

【背景技術】

【０００２】

従来、複数の演算処理装置が主記憶装置を共有するＳＭＰ（Ｓｙｍｍｅｔｒｉｃ　ＭｕｌｔｉＰｒｏｃｅｓｓｏｒ）の技術が知られている。このようなＳＭＰの技術が適用された情報処理システムの一例として、演算処理装置と主記憶装置とを有する複数のノードを同一のバスで接続し、バスを介して、各演算処理装置が各主記憶装置を共有する情報処理システムが知られている。

【０００３】

このような情報処理システムでは、例えばスヌープ方式を用いて、各ノードの演算処理装置がキャッシュしたデータのコヒーレンシを保持する。しかし、スヌープ方式では、各演算処理装置がキャッシュしたデータの更新状況をバスを介してやり取りするので、ノード数が増加するに従って、バスがボトルネックとなり、メモリアクセスの性能が悪化する。

【０００４】

このようなバスのボトルネックを回避するため、インターコネクトを用いて、複数のノードを接続し、各ノードの演算処理装置が各ノードの主記憶装置を共有するＮＵＭＡ（Ｎｏｎ　Ｕｎｉｆｏｒｍ　Ｍｅｍｏｒｙ　Ａｃｃｅｓｓ）の技術が知られている。

【０００５】

このようなＮＵＭＡの技術が適用された情報処理システムでは、各ノードの主記憶装置の記憶領域が共通の物理アドレス空間に対して一意にマップされる。そして、各ノードの演算装置は、アクセス対象の物理アドレスが示す記憶領域が存在するノードを識別し、インターコネクトを介して、識別したノードの主記憶装置にアクセスする。

【先行技術文献】

【特許文献】

【０００６】

【特許文献１】特開２０００－２３５５５８号公報

【非特許文献】
【０００７】
【非特許文献１】Computer Architecture: A Quantitative Approach, Second Edition , John L. Hennessy ,David A. Patterson ,§8.4
【発明の概要】
【発明が解決しようとする課題】
【０００８】
ここで、上述したＮＵＭＡの技術では、各ノードの演算処理装置がキャッシュしたデータのコヒーレンシを保つことができない。そこで、各ノードの演算処理装置がキャッシュしたデータのコヒーレンシを保つ機構を備えるｃｃＮＵＭＡ（Ｃａｃｈｅ Ｃｏｈｅｒｅｎｔ ＮＵＭＡ）を採用することが考えられる。
【０００９】
しかしながら、ｃｃＮＵＭＡを適用した情報処理システムでは、各ノードがアクセス対象となる記憶領域が存在するノードを識別するので、アドレス変換を効率良く行う必要がある。また、各ノードは、主記憶装置を自ノードのみが利用する記憶領域と、他ノードと共用する記憶領域とに分割する場合がある。このような場合には、各ノードは、アクセス対象記憶領域が、他のノードと共用する記憶領域であるかを効率良く行う必要がある。
【００１０】
本発明は、１つの側面では、各演算処理装置によるメモリアクセスを効率的に行うことができる。
【課題を解決するための手段】
【００１１】
１つの側面では、それぞれがプロセッサ及び記憶装置を備える複数のノードと、各ノード間を接続するインターコネクトとを有する情報処理装置である。このような情報処理装置が有する各ノードは、各ノードに備えられたプロセッサを識別するプロセッサ識別情報と、当該プロセッサ識別情報が示すプロセッサを備えるノードの記憶装置に割当てられた物理アドレスとを対応付けて記憶する記憶部を有する。また、各ノードは、論理アドレスと物理アドレスとの変換を行う。また、各ノードは、物理アドレスを、前記記憶部に当該物理アドレスと対応付けて記憶されたプロセッサ識別情報に変換する。また、各ノードは、第１変換部によって論理アドレスから変換された物理アドレスと、前記第２変換部により物理アドレスから変換されたプロセッサ識別情報を含み、当該プロセッサ識別情報が識別するプロセッサを備えるノードの記憶装置からデータの読出しを指示する読出要求、または、前記物理アドレス及び前記プロセッサ識別情報を含み、当該プロセッサ識別情報が識別するプロセッサを備えるノードの記憶装置へのデータの書込みを指示する書込要求を送信する。また、各ノードは、他のノードからインターコネクトを介して送信された読出要求または書込要求を受信すると、読出要求または書込要求に含まれる物理アドレスに基づいて、自ノード内のアクセスに用いられるローカル領域と、他ノードからもアクセス可能な共有領域とのいずれの領域へのアクセスであるかを判定する。また、前記ノードは、各ノードへの前記プロセッサまたは前記記憶装置の追加や削除に応じて、前記記憶部に記憶されたプロセッサ識別情報と物理アドレスとの対応付けを書換える制御部を有する。
【発明の効果】
【００１２】
１実施形態によれば、各演算処理装置によるメモリアクセスを効率的に行うことができる。
【図面の簡単な説明】
【００１３】
【図１】図１は、実施例１に係る情報処理システムの一例を説明するための図である。
【図２】図２は、実施例１に係るビルディングブロックの機能構成を説明するための図である。
【図３】図３は、実施例１に係るビルディングブロックのメモリに振り分けられる物理ア

ドレスの範囲を説明するための図である。
【図４】図４は、実施例１に係る情報処理システムが各メモリに振り分ける物理アドレスを説明するための図である。
【図５】図５は、物理アドレスの振り分けのバリエーションを説明するための第１の図である。
【図６】図６は、物理アドレスの振り分けのバリエーションを説明するための第２の図である。
【図７】図７は、実施例１に係るＣＰＵの機能構成を説明するための図である。
【図８】図８は、実施例１に係るノードマップが記憶する情報の一例を説明するための図である。
【図９】図９は、ノードマップが記憶する情報のバリエーションの一例を説明するための第１の図である。
【図１０】図１０は、ノードマップが記憶する情報のバリエーションの一例を説明するための第２の図である。
【図１１Ａ】図１１Ａは、キャッシュタグの一例を説明するための図である。
【図１１Ｂ】図１１Ｂは、実施例１に係るＣＰＵが送信するパケットを説明するための図である。
【図１２】図１２は、実施例１に係るＣＰＵがリクエストを送信する処理の一例を説明するための図である。
【図１３】図１３は、実施例１に係るＣＰＵがパケットを受信した際に実行する処理の一例を説明するための図である。
【図１４】図１４は、ノードマップを設定する処理の流れを説明するためのフローチャートである。
【図１５】図１５は、共有領域を制御する処理の流れを説明するためのフローチャートである。
【図１６】図１６は、共有メモリの割当処理を説明するためのフローチャートである。
【図１７】図１７は、共有メモリアタッチ処理を説明するためのフローチャートである。
【図１８】図１８は、アプリケーションが共有メモリを使用する処理を説明するためのフローチャートである。
【図１９】図１９は、ノード間の共有メモリデタッチ処理を説明するためのフローチャートである。
【図２０】図２０は、ノード間共有メモリの解放処理を説明するためのフローチャートである。
【図２１】図２１は、リクエストを発行する処理の流れを説明するためのフローチャートである。
【図２２】図２２は、リクエストを受信した際に実行する処理の流れを説明するためのフローチャートである。
【図２３】図２３は、ＣＰＵが応答を受信した際に実行する処理の流れを説明するためのフローチャートである。
【図２４】図２４は、実施例２に係る情報処理システムを説明するための図である。
【図２５】図２５は、パーティションの一例を説明するための図である。
【図２６Ａ】図２６Ａは、パーティション＃ＡのＣＰＵが記憶するノードマップの一例を説明するための図である。
【図２６Ｂ】図２６Ｂは、パーティション＃Ａを示すノードマップの一例を説明するための図である。
【図２６Ｃ】図２６Ｃは、パーティション＃Ｂを示すノードマップの一例を説明するための図である。
【発明を実施するための形態】
【００１４】
以下に添付図面を参照して本願に係る情報処理装置及びメモリアクセス方法について説

明する。
【実施例１】
【００１５】
まず、本願に係る実施例の説明の前に、従来の情報処理システムが有する問題の具体例について説明する。例えば、従来の情報処理システムは、ＣＰＵ（Central Processing Unit）が共有メモリ領域にアクセスするために出力した論理アドレスを共有メモリ空間アドレスに変換する。そして、情報処理システムは、共有メモリ空間をアドレスを物理アドレスに変換することで、ＣＰＵのアクセス対象となる記憶領域を識別する。
【００１６】
しかし、このように論理アドレスを共有メモリ空間アドレスに変換し、変換後の共有メモリ空間アドレスを物理アドレスに変換する手法では、アドレス変換に要するハードウェアの物量が多くなってしまう。また、論理アドレスを共有メモリ空間アドレスに変換し、変換後の共有メモリ空間アドレスを物理アドレスに変換する手法では、アドレス変換に要する時間が増加してしまう。
【００１７】
また、従来の情報処理システムは、ＣＰＵが共有メモリ空間のデータをキャッシュする際に、全てのＣＰＵに対してキャッシュ情報を送信することで、コヒーレンシを保持する。しかし、このように、全てのＣＰＵに対してキャッシュ情報を送信する手法では、ボトルネックが発生し、メモリアクセスの性能が悪化してしまう。また、従来の情報処理システムは、ＣＰＵの増設を行った場合は、ＣＰＵの個数の増加に比例してバストラフィックが増加するので、ボトルネックが発生し、メモリアクセスの性能が悪化してしまう。
【００１８】
また、例えば、ノードは、自ノードのみがアクセスするローカル領域にカーネルデータやユーザデータを格納する。このため、各ノードは、ローカル領域に格納したデータのセキュリティを確保し、ソフトウェアバグに対する耐性を高くするため、アクセス対象となる記憶領域が他ノードからアクセス可能である共有メモリ域かアクセス不能なローカルメモリ域かの判断を要する。
【００１９】
このため、従来の情報処理システムにおいては、ローカル領域に記憶されたデータはキャッシュ可能とし、共有領域に記憶されたデータはキャッシュ不能とする。しかし、このように共有領域に記憶されたデータをキャッシュ不能とする手法では、メモリアクセスにおけるレイテンシが増大してしまう。なお、他のノードからメモリアクセスが行われる度にアクセス対象が共有領域であるかローカル領域であるかを判断する場合には、判断を行うための回路規模が増大するとともに、アクセスにおけるレイテンシが増大してしまう。
【００２０】
また、従来の情報処理システムにおいては、ノードは、他のノードが有するメモリにアクセスする度に、特殊なチャネル装置やＤＭＡ（Direct Memory Access）エンジンのプログラムの実行を要するため、メモリアクセスの性能が劣化してしまう。また、従来の情報処理システムでは、メモリが有する記憶領域のどの領域を共有領域とするかを固定的に設定する。このため、例えば、従来の情報処理システムでは、システムを停止させることなく、ノードを追加することで共有領域を追加することができない。
【００２１】
また、従来の情報処理システムにおいては、チャネルやＤＭＡ経路を介したメモリアクセスを行うためのハードウェアが追加される。このため、従来の情報処理システムにおいては、ノード間でメモリを共有しないシステムと比較して、設置されるハードウェアが大きく異なる。この結果、従来の情報処理システムにおいては、ノード間でメモリを共有する場合には、ＯＳ（Operating System）等のプログラムを大幅に変更しなければならない。
【００２２】
以下の説明では、実施例１として、上述した問題を解決する情報処理システムの一例に

ついて説明する。まず、図１を用いて、情報処理システムの構成例について説明する。図１は、実施例１に係る情報処理システムの一例を説明するための図である。図１に示す例では、情報処理システム１は、ＸＢ（クロスバスイッチ）２と複数のビルディングブロック１０～１０ｅとを有する。また、各ビルディングブロック１０～１０ｅは、管理用ネットワークを介して管理端末３と接続されている。また、ＸＢ２は、サービスプロセッサ２ｂを有する。

【００２３】

ビルディングブロック１０は、複数のＣＰＵ２１～２１ｃと複数のメモリ２２～２２ｃとサービスプロセッサ２４とを有する。また、他のビルディングブロック１０～１０ｅも、ビルディングブロック１０と同様の構成を有するものとし、以下の説明を省略する。なお、図１に示す例では、ＣＰＵ２１ｂ、２１ｂおよびメモリ２２ｂ、２２ｃについては、記載を省略した。

【００２４】

ＸＢ２は、各ビルディングブロック１０～１０ｅを相互に接続するクロスバスイッチである。また、ＸＢ２が有するサービスプロセッサ２ｂは、各ビルディングブロック１０～１０ｅが有するサービスプロセッサを管理するサービスプロセッサ、すなわち、マスタとなるサービスプロセッサである。また、管理端末３は、管理用ネットワークを介して、各ビルディングブロック１０～１０ｅが有するサービスプロセッサの設定や制御を行う端末である。なお、少数のノードが接続される小規模構成の場合、ＸＢ２を介さずにビルディングブロック同士を直接接続してもよい。

【００２５】

各ビルディングブロック１０～１０ｅは、それぞれ独立してＯＳを動作させる。すなわち、各ビルディングブロック１０～１０ｅが実行するＯＳは、ビルディングブロック毎に異なるパーティションで動作する。ここで、パーティションとは、同一のＯＳが動作し、動作しているＯＳから見て１つのシステムとして動作するビルディングブロックの群を示す。

【００２６】

例えば、ビルディングブロック１０～１０ａがパーティション＃Ａとして動作し、ビルディングブロック１０ｂ～１０ｄがパーティション＃Ｂとして動作する。このような場合には、ビルディングブロック１０が動作させるＯＳは、ビルディングブロック１０、１０ａが１つのシステムとして動作していると識別し、ビルディングブロック１０ｂが動作させるＯＳは、ビルディングブロック１０ｂ～１０ｄが１つのシステムとして動作していると識別する。

【００２７】

次に、図２を用いて、ビルディングブロックの構成例について説明する。図２は、実施例１に係るビルディングブロックの機能構成を説明するための図である。図２に示す例では、ビルディングブロック１０は、ノード２０、サービスプロセッサ２４、ＸＢ接続部２７、２７ａ、ＰＣＩｅ（Peripheral Component Interconnect Express）接続部２８を有する。

【００２８】

また、ノード２０は、複数のＣＰＵ２１～２１ｃと複数のメモリ２２～２２ｃと通信部２３とを有する。また、サービスプロセッサ２４は、制御部２５と通信部２６とを有する。また、図２に示す例では、各ＣＰＵ２１～２１ｃは、相互に直接接続されるとともに、通信部２３と接続されている。また、各メモリ２２～２２ｃは、各ＣＰＵ２１～２１ｃと接続されている。

【００２９】

また、各ＣＰＵ２１～２１ｃは、ＸＢ接続部２７またはＸＢ接続部２７ａと接続されている。なお、ＸＢ接続部２７、２７ａは、同一のＸＢ接続部であってもよい。また、各ＣＰＵ２１～２１ｃは、ＰＣＩｅ接続部２８と接続されている。また、通信部２３は、サービスプロセッサ２４が有する通信部２６と接続されている。なお、制御部２５、通信部２

６、通信部２３、各ＣＰＵ２１～２１ｃは、例えば、ＪＴＡＧ（Joint Test Action Group）やＩ２Ｃ（Inter-Integrated Circuit）で接続されている。

【００３０】

例えば、図２に示す例では、ＣＰＵ２１～２１ｃは、演算処理を実行する演算処理装置である。また、各ＣＰＵ２１～２１ｃには、それぞれ独立したメモリ２２～２２ｃが接続されている。また、各ＣＰＵ２１～２１ｃは、メモリ２２～２２ｃや、他のビルディングブロック１０ａ～１０ｅが有するメモリを共有メモリとして利用する。また、各ＣＰＵ２１～２１ｃは、後述するように、物理アドレスと、物理アドレスが割り振られたメモリと接続されたＣＰＵの識別子であるＣＰＵＩＤ（identification）とを対応付けたノードマップを有する。

【００３１】

そして、例えば、ＣＰＵ２１は、アクセス対象となる物理アドレスと対応付けられたＣＰＵＩＤが、ノード２０とは異なるノードが有するＣＰＵを示す場合には、ＸＢ接続部２７およびＸＢ２を介して他のノードにメモリアクセスのリクエストを送信する。また、ＣＰＵ２１は、アクセス対象となる物理アドレスと対応付けられたＣＰＵＩＤが、ＣＰＵ２１ａ～２１ｃを示す場合には、ＣＰＵ間の直接接続を介して、メモリアクセスのリクエストを送信する。すなわち、ＣＰＵ２１は、アクセス対象となる物理アドレスと対応付けられたＣＰＵＩＤが、ＣＰＵ２１以外のＣＰＵであって、自身と同じノード２０に存在するＣＰＵを示す場合には、ＣＰＵ間の直接接続を介して、メモリアクセスのリクエストを送信する。

【００３２】

また、ＣＰＵ２１は、自身と接続されたメモリに対するリクエストを他のノードから受信した場合には、リクエストの対象となるデータを自身と接続されたメモリ２２から読出し、リクエスト元へ送信する。

【００３３】

なお、各ＣＰＵ２１～２１ｃは、実行中のアプリケーションが共有メモリの割り当てを要求した場合には、相互に通信を行い、アプリケーションが使用する共有メモリの割当てを行う機能を有する。また、各ＣＰＵ２１～２１ｃは、ＴＬＢを用いたアドレス変換を行うとともに、ＴＬＢミスが発生した際に、トラップ処理を実行するなど、従来のＣＰＵと同様の処理を実行する機能を有するものとする。

【００３４】

メモリ２２～２２ｃは、情報処理システム１が有する全てのＣＰＵが共用するメモリである。また、情報処理システム１においては、全てのビルディングブロック１０～１０ｅが有するメモリに対して、各ビルディングブロック１０～１０ｅのサービスプロセッサが、同一の物理アドレス空間にマッピングされる物理アドレスを振分ける。すなわち、情報処理システム１が有する全てのメモリには、重複しない値の物理アドレスが割当てられている。

【００３５】

また、メモリ２２～２２ｃは、記憶領域の一部を、情報処理システム１が有する全てのＣＰＵが共用する共有領域とし、他の部分を、自身にアクセスするＣＰＵ２１～２１ｃがカーネルデータやユーザデータを格納するローカル領域とする。また、メモリ２２～２２ｃには、情報処理システム１が用いる物理アドレス空間のうち、ある位置のビットが同一の値となる範囲の物理アドレスが有領域に振り分けられる。また、メモリ２２～２２ｃには、ある位置のビットが共有領域に振り分けた物理アドレスとは異なる値となる範囲の物理アドレスがーカル領域に振り分けられる。

【００３６】

例えば、メモリ２２～２２ｃには、４６ビット目のビットが「０」となる物理アドレスがローカル領域に振り分けられ、４６ビット目のビットが「１」となる物理アドレスが共有領域に振り分けられる。詳細な例を挙げると、メモリ２２～２２ｃのローカル領域に対しては、物理アドレス空間のうち、「０」～「０ｘ６３ｆｆ　ｆｆｆｆ　ｆｆｆｆ」に含

まれる物理アドレスが振り分けられる。また、メモリ２２～２２ｃの共有領域に対しては、物理アドレス空間のうち、「０ｘ６４００　０００　００００」～「０ｘ１　２７ｆｆ　ｆｆｆｆ　ｆｆｆｆ」に含まれる物理アドレスが振り分けられる。

【００３７】

なお、情報処理システム１においては、各ビルディングブロック１０～１０ｇごとに、異なる範囲に含まれる物理アドレスをメモリに振り分ける。以下、図面を用いて、情報処理システム１において、各ビルディングブロック１０～１０ｅごとに、メモリに振り分ける物理アドレスの範囲を説明する。

【００３８】

図３は、実施例１に係るビルディングブロックのメモリに振り分けられる物理アドレスの範囲を説明するための図である。なお、図３に示す例では、各ビルディングブロックをＢＢ（Bilding Block）と記載した。また、ＢＢ＃０とは、ビルディングブロック１０を示し、ＢＢ＃１とは、ビルディングブロック１０ａを示し、ＢＢ＃１５とは、ビルディングブロック１０ｅを示す。すなわち、図３に示す例では、情報処理システム１は、１６個のビルディングブロックを有するものとする。

【００３９】

また、図３に示す例では、各ビルディングブロックには、最大で４ＴＢ（Terabyte)のメモリを搭載可能であるものとする。また、以下の説明においては、メモリアドレスの表記を簡易化するため、例えば「2^{42}」となるアドレス番地を「４ＴＢ」と記載する。

【００４０】

図３に示す例では、ビルディングブロック１０が有するメモリ２２～２２ｃにおいては、物理アドレス空間のうち、「０」～「４ＴＢ－１」までの範囲に含まれる物理アドレスがローカル領域に振り分けられる。また、ビルディングブロック１０が有するメモリ２２～２２ｃにおいては、物理アドレス空間のうち、「６４ＴＢ」～「６８ＴＢ－１」までの範囲に含まれる物理アドレスが共有領域に振り分けられる。

【００４１】

また、ビルディングブロック１０ａが有するメモリにおいては、物理アドレス空間のうち、「４ＴＢ」～「８ＴＢ－１」までの範囲に含まれる物理アドレスがローカル領域に振り分けられる。また、ビルディングブロック１０ａが有するメモリにおいては、物理アドレス空間のうち、「６８ＴＢ」～「７２ＴＢ－１」までの範囲に含まれる物理アドレスが共有領域に振り分けられる。

【００４２】

また、ビルディングブロック１０ｅが有するメモリにおいては、物理アドレス空間のうち、「６０ＴＢ」～「６４ＴＢ－１」までの範囲に含まれる物理アドレスがローカル領域に振り分けられる。また、ビルディングブロック１０ａが有するメモリにおいては、物理アドレス空間のうち、「１２４Ｂ」～「１２８ＴＢ－１」までの範囲に含まれる物理アドレスが共有領域に振り分けられる。

【００４３】

この結果、情報処理システム１は、図４に示すように、物理アドレス空間を各ビルディングブロック１０～１０ｅが有する全てのメモリに対して振り分けることとなる。図４は、実施例１に係る情報処理システムが各メモリに振り分ける物理アドレスを説明するための図である。

【００４４】

具体的には、図４に示す例では、情報処理システム１は、「０」～「２５６ＴＢ－１」までの物理アドレスのうち、「０」～「６４ＴＢ－１」までの範囲をローカル領域に振り分ける物理アドレスとする。また、情報処理システム１は、「６４ＴＢ」～「１２８ＴＢ－１」までの範囲を共有領域に振り分ける物理アドレスとする。

【００４５】

すなわち、情報処理システム１は、最下位のビットを０ビット目として４６ビット目のビットが「０」の範囲をローカル領域に振り分け、「１」の範囲を共有領域に振り分ける

。なお、情報処理システム１は、「１２８ＴＢ」～「２５６ＴＢ－１」までの範囲をＩ／Ｏ空間として用いる。

【００４６】

なお、図３、４に示す例は、あくまで一例であり、情報処理システム１は、異なる振り分け方を採用してもよい。以下、情報処理システム１が、物理アドレスを振り分けるバリエーションの例について図を用いて説明する。

【００４７】

図５は、物理アドレスの振り分けのバリエーションを説明するための第１の図である。図５に示す例では、各ビルディングブロック１０～１０ｅが有するメモリにおいては、「０」～「４ＴＢ－１」までの範囲に含まれる物理アドレスがローカル領域に振り分けられる。また、図５に示す例では、ビルディングブロック１０が有するメモリ２２においては、「４ＴＢ」～「８ＴＢ－１」までの範囲に含まれる物理アドレスが共有領域に振り分けられる。

【００４８】

また、図５に示す例では、ビルディングブロック１０ａが有するメモリにおいては、「８ＴＢ」～「１２ＴＢ－１」までの範囲に含まれる物理アドレスが共有領域に振り分けられる。また、図５に示す例では、ビルディングブロック１０ｅが有するメモリにおいては、「６４ＴＢ」～「６８ＴＢ－１」までの範囲に含まれる物理アドレスが共有領域に振り分けられる。

【００４９】

この結果、図５に示す例では、情報処理システム１は、物理アドレス空間のうち、「０」～「４ＴＢ－１」までの範囲の物理アドレスをローカル領域に振り分け、「４ＴＢ」～「１２８ＴＢ－１」までの範囲の物理アドレスを共有領域に振り分ける。また、図５に示す例では、情報処理システム１は、「１２８ＴＢ」～「２５６ＴＢ－１」までの範囲をＩ／Ｏ空間として用いる。すなわち、情報処理システム１は、最下位のビットを０ビット目として４２ビット目のビットが「０」の範囲をローカル領域に振り分け、「１」の範囲を共有領域に振り分ける。

【００５０】

また図６は、物理アドレスの振り分けのバリエーションを説明するための第２の図である。図６に示す例では、各ビルディングブロック１０～１０ｅが有するメモリにおいては、「０」～「４ＴＢ－１」までの範囲に含まれる物理アドレスがＩ／Ｏ空間用に保存される。また、図６に示す例では、各ビルディングブロック１０～１０ｅが有するメモリにおいては、「４ＴＢ」～「８ＴＢ－１」までの範囲に含まれる物理アドレスがローカル領域に振り分けられる。

【００５１】

また、図６に示す例では、ビルディングブロック１０が有するメモリ２２～２２ｃにおいては、「８ＴＢ」～「１２ＴＢ－１」までの範囲に含まれる物理アドレスが共有領域に振り分けられる。また、図６に示す例では、ビルディングブロック１０ａが有するメモリにおいては、「１２ＴＢ」～「１６ＴＢ－１」までの範囲に含まれる物理アドレスが共有領域に振り分けられる。また、図６に示す例では、ビルディングブロック１０ｅが有するメモリにおいては、「６８ＴＢ」～「７２ＴＢ－１」までの範囲に含まれる物理アドレスが共有領域に振り分けられる。

【００５２】

この結果、図６に示す例では、情報処理システム１は、物理アドレス空間のうち、「０」～「４ＴＢ－１」までの範囲をＩ／Ｏ空間とし、「４ＴＢ」～「８ＴＢ－１」までの範囲の物理アドレスをローカル領域に振り分けることとなる。また、図５に示す例では、情報処理システム１は、「８ＴＢ」～「２５６ＴＢ－１」までの範囲の物理アドレスを共有領域に振り分けることとなる。すなわち、情報処理システム１は、最下位のビットを０ビット目として４３ビット目のビットが「０」の範囲をローカル領域に振り分け、「１」の範囲を共有領域に振り分ける。

【００５３】
図２に戻って、制御部２５は、ビルディングブロック１０の制御を行う。例えば、制御部２５は、ビルディングブロック１０の電源管理や、ビルディングブロック１０内の異常の監視や制御等を実行する。また、制御部２５は、管理用ネットワークを介して、管理端末３や他のビルディングブロック１０～１０ｅが有するサービスプロセッサの制御部とも接続されており、管理端末３によって指示された制御や、各ビルディングブロック１０～１０ｅ間で連係した制御を実行できる。また、制御部２５は、各ＣＰＵ２１～２１ｃが実行するＯＳと通信を行うことができる。
【００５４】
なお、実施例１では、各ビルディングブロック１０～１０ｅが有するサービスプロセッサは、管理用ネットワークを介して接続されているが、実施例はこれに限定されるものではない。たとえば、各ビルディングブロック１０～１０ｅを接続するＸＢを介して相互に通信しても良い。
【００５５】
また、制御部２５は、通信部２６と通信部２３を介して、各ＣＰＵ２１～２１ｃにアクセスする。そして、制御部２５は、後述するように、各ビルディングブロック１０～１０ｅが有するノードマップの更新や制御等を実行する。
【００５６】
なお、通信部２３は、サービスプロセッサ２４が有する通信部２６を介して、制御部２５による制御信号を各ＣＰＵ２１～２１ｃに伝達する。また、通信部２６は、制御部２５による制御信号をノード２０が有する通信部２３に伝達する。また、ＸＢ接続部２７、２７ａは、各ＣＰＵ２１～２１ｃをＸＢ２と接続し、各ビルディングブロック１０～１０ｅが有するＣＰＵ間の通信を中継する。また、ＰＣＩｅ接続部２８は、各ＣＰＵ２１～２１ｃによるＩ／Ｏ（Input　Output）装置へのアクセスを中継する。
【００５７】
次に、図７を用いて、各ＣＰＵ２１～２１ｃが有する機能構成について説明する。図７は、実施例１に係るＣＰＵの機能構成を説明するための図である。なお、ＣＰＵ２１ａ～２１ｃは、ＣＰＵ２１と同様の機能を発揮するものとして、説明を省略する。また、図７に示す例では、サービスプロセッサ２４とＣＰＵ２１とを接続する接続部２３、２６については、記載を省略した。
【００５８】
図７に示す例では、ＣＰＵ２１は、演算処理部３０、ルータ４０、メモリアクセス部４１、ＰＣＩｅ制御部４２を有する。また、演算処理部３０は、演算部３１、Ｌ１（Level 1）キャッシュ３２、Ｌ２キャッシュ３３、ノードマップ３４、アドレス変換部３５、キャッシュディレクトリ管理部３６、パケット制御部３７を有する。また、パケット制御部３７は、リクエスト生成部３８、リクエスト受信部３９を有する。また、ＰＣＩｅ制御部４２は、リクエスト生成部４３、ＰＣＩｅバス制御部４４を有する。
【００５９】
まず、演算処理部３０が有するノードマップ３４について説明する。ノードマップ３４は、物理アドレスと、物理アドレスが示す記憶領域を有するメモリと接続されたＣＰＵのＣＰＵＩＤとを対応付けて記憶する。以下、ノードマップ３４が記憶する情報の例を図面を用いて説明する。
【００６０】
図８は、実施例１に係るノードマップが記憶する情報の一例を説明するための図である。図８に示す例では、ノードマップ３４は、アドレス、バリッド、ノードＩＤ、ＣＰＵＩＤとを対応付けたエントリを記憶する。ここで、各エントリのアドレスには、連続する複数の物理アドレスを含むアドレス域を示す情報が格納される。
【００６１】
例えば、情報処理システム１は、全てのメモリに対して振り分けた物理アドレス空間を均等な大きさのアドレス域に分割し、各アドレス域に＃０、＃１、＃２等の識別子を付与

する。そして、情報処理システム１は、各アドレス域を示す識別子を、ノードマップ３４が有する各エントリのアドレスに格納する。

【００６２】

また、各エントリのバリッドには、物理アドレスが示す記憶領域にアクセスすることができるか否かを示すバリッドビットが格納される。例えば、物理アドレスが示す記憶領域が、各ＣＰＵで共有される共有領域である場合には、アクセスを行う事ができる旨のバリッドビット（例えば「１」）が格納される。

【００６３】

また、ノードＩＤとは、物理アドレスが振り分けられたメモリが存在するノードを示す識別子である。また、ＣＰＵＩＤとは、物理アドレスが振り分けられたメモリと接続されたＣＰＵを示す識別子である。すなわち、ノードマップ３４は、アクセス対象となる物理アドレスが、どのＣＰＵと接続されたメモリの物理アドレスであるかを示す情報を記憶する。

【００６４】

例えば、図８に示す例では、ノードマップ３４は、識別子が「＃０」のアドレス域が、ノードＩＤ「０」で示されるノードに存在し、ＣＰＵＩＤが「０」のＣＰＵがアクセスを行う旨を示す。また、ノードマップ３４は、識別子が「＃１」のアドレス域が、ノードＩＤ「０」で示されるノードに存在し、ＣＰＵＩＤが「１」のＣＰＵがアクセスを行う旨を示す。また、ノードマップ３４は、識別子が「＃２」のアドレス域が、ＣＰＵ２１がアクセスを行わない、又は、マッピングされていないアドレス域であるため、ノードＩＤとＣＰＵＩＤとが設定されていない旨を示す。

【００６５】

なお、ノードマップ３４は、アクセス対象となる物理アドレスがどのＣＰＵと接続された物理アドレスであるかを示すことができれば、本実施例以外の任意の形式で情報を登録することとしてよい。以下、ノードマップ３４のバリエーションの例について、図９および図１０を用いて説明する。

【００６６】

図９は、ノードマップが記憶する情報のバリエーションの一例を説明するための第１の図である。図９に示す例では、ノードマップ３４は、バリッド、スタートアドレス、アドレスマスク、ノードＩＤ、ＣＰＵＩＤを対応付けてエントリを記憶する。ここで、スタートアドレスとは、アドレス域に含まれる物理アドレスのうち、最若番の物理アドレスが格納される。

【００６７】

また、アドレスマスクには、ＣＰＵが管理する物理アドレスの範囲を示すアドレスマスクが格納される。例えば、あるエントリは、アドレスマスクが「０ｘｆｆｆｆ　ｆｆｆｆ　ｆｆｆｆ　００００」である場合には、同一エントリのスタートアドレスと上位４８ビットが一致するアドレス領域を、同一エントリのＣＰＵＩＤが示すＣＰＵが管理することを示す。

【００６８】

例えば、図９に示す例では、ノードマップ３４は、最初のエントリとして、アドレス「０ｘ０００００」からアドレスマスク「０ｘ３ｆｆｆ」でマスクされる範囲、すなわち「０ｘ０３ｆｆｆ」までの範囲が１つのアドレス域である旨を示す。また、ノード３４は、「０ｘ０００００」から「０ｘ０３ｆｆｆ」のアドレス域が、ノードＩＤ「０」で示されるノードに存在し、ＣＰＵＩＤが「０」のＣＰＵがアクセスするアドレス域である旨を示す。

【００６９】

同様に、ノードマップ３４は、「０ｘ１００００」から「０ｘ１３ｆｆｆ」のアドレス域が、ノードＩＤ「１」で示されるノードに存在し、ＣＰＵＩＤが「４」のＣＰＵがアクセスするアドレス域である旨を示す。また、ノードマップ３４は、「０ｘ１４０００」から「０ｘ１７ｆｆｆ」のアドレス域が、ノードＩＤ「１」で示されるノードに存在し、Ｃ

ＰＵＩＤが「５」のＣＰＵがアクセスするアドレス域である旨を示す。また、ノードマップ３４は、「０ｘ２００００」から「０ｘ２１ｆｆｆ」のアドレス域が、ノードＩＤ「２」で示されるノードに存在し、ＣＰＵＩＤが「８」のＣＰＵがアクセスするアドレス域である旨を示す。

【００７０】

なお、ノードマップ３４は、図９に示すように、アドレス域をスタートアドレスとアドレスマスクとで表現した場合には、物理アドレスが各アドレス域に含まれるか否かを論理和と論理積の組合せで実行することができるため、回路構成が容易となる。

【００７１】

また、図１０は、ノードマップが記憶する情報のバリエーションの一例を説明するための第２の図である。図１０に示す例では、ノードマップ３４は、バリッド、スタートアドレス、レングス、ノードＩＤ、ＣＰＵＩＤを対応付けたエントリを記憶する。ここで、レングスとは、アドレス域の大きさを設定する情報である。

【００７２】

例えば、スタートアドレスが「０ｘ１２　００００」で、レングスが「０ｘ１　ｆｆｆｆ」とすると、同一エントリのＣＰＵＩＤが示すＣＰＵは、管理するメモリに対して、物理アドレス「０ｘ１２　００００」から「０ｘ１３　ｆｆｆｆ」を割当てることとなる。

【００７３】

例えば、図１０に示す例では、ノードマップ３４は、最初のエントリとして、アドレス「０ｘ０００００」からレングスが「０ｘ３ｆｆｆ」に含まれる範囲、すなわち「０ｘ０３ｆｆｆ」までの範囲が１つのアドレス域である旨を示す。また、ノード３４は、「０ｘ０００００」から「０ｘ０３ｆｆｆ」のアドレス域が、ノードＩＤ「０」で示されるノードに存在し、ＣＰＵＩＤが「０」のＣＰＵがアクセスするアドレス域である旨を示す。

【００７４】

同様に、ノードマップ３４は、「０ｘ１００００」から「０ｘ１３ｆｆｆ」のアドレス域が、ノードＩＤ「１」で示されるノードに存在し、ＣＰＵＩＤが「４」のＣＰＵがアクセスするアドレス域である旨を示す。また、ノードマップ３４は、「０ｘ１４０００」から「０ｘ１７ｆｆ」のアドレス域が、ノードＩＤ「１」で示されるノードに存在し、ＣＰＵＩＤが「５」のＣＰＵがアクセスするアドレス域である旨を示す。また、ノードマップ３４は、「０ｘ２００００」から「０ｘ２０２ｅｆ」のアドレス域が、ノードＩＤ「２」で示されるノードに存在し、ＣＰＵＩＤが「８」のＣＰＵがアクセスするアドレス域である旨を示す。

【００７５】

なお、ノードマップ３４は、図１０に示すように、アドレス域をスタートアドレスとレングスとで表現した場合には、各アドレス域の長さを柔軟に設定することができる。すなわち、ノードマップ３４は、アドレス域をスタートアドレスとアドレスマスクとで表現した場合は、ＬＳＢ（Least　Significant　Bit）から１が連続する範囲のアクセス域を指定することとなる。一方、各アドレス域をスタートアドレスとレングスとで表現した場合には、各アドレス域の長さを任意の長さに設定することができる。

【００７６】

図７に戻って、演算部３１は、演算処理を実行し、ＯＳやアプリケーションを実行する演算装置のコアである。また、演算部３１は、データの読み込みを行う場合には、読み込み対象となるデータが格納された記憶領域の論理アドレスをアドレス変換部３５に出力する。

【００７７】

Ｌ１キャッシュ３２は、データやディレクトリのうち頻繁に利用されるデータを一時的に記憶するキャッシュメモリである。Ｌ２キャッシュ３３は、Ｌ１キャッシュ３２と同様に、データやディレクトリのうち頻繁に利用されるデータを一時的に記憶するが、Ｌ１キャッシュ３２よりも記憶容量が大きく、データを読み書きする速度が低速なキャッシュメモリである。なお、ディレクトリとは、メモリ２２の各記憶領域に記憶されたデータをキ

ャッシュしたＣＰＵや、キャッシュされたデータの更新状況を示す情報である。

【００７８】

アドレス変換部３５は、ＴＬＢ（Translation Lookaside Buffer）を用いて、演算部３１が出力した論理アドレスを物理アドレスに変換する。例えば、アドレス変換部３５は、論理アドレスと物理アドレスとを対応付けたエントリを記憶するＴＬＢを有し、演算部３１から取得した論理アドレスと対応付けて記憶する物理アドレスをキャッシュディレクトリ管理部３６に出力する。なお、アドレス変換部３５は、ＴＬＢミスが発生した場合は、トラップ処理を実行し、ＴＬＢミスした物理アドレスと論理アドレスの組をＴＬＢに登録する。

【００７９】

また、アドレス変換部３５は、ＣＰＵ２１が実行するアプリケーションから共有メモリへの割当てを要求された場合には、以下の処理を実行する。すなわち、アドレス変換部３５は、各ＣＰＵ２１～２１ｃが共用する共有領域にアクセスする際にアプリケーションが用いる論理アドレスと、共有領域に割当てられる範囲の物理アドレスとを対応付けたエントリをＴＬＢに設定する。

【００８０】

また、アドレス変換部３５は、アプリケーションやＯＳからローカル領域の割当を要求された場合は、以下の処理を実行する。すなわち、アドレス変換部３５は、アプリケーションやＯＳがＣＰＵ２１専用のローカル領域にアクセスするための論理アドレスと、ローカル領域に割当てられる範囲の物理アドレスとを対応付けたエントリをＴＬＢに設定する。

【００８１】

キャッシュディレクトリ管理部３６は、キャッシュデータおよびディレクトリの管理を行う。具体的には、キャッシュディレクトリ管理部３６は、アドレス変換部３５から、演算部３１が出力した論理アドレスを変換した物理アドレスを取得する。

【００８２】

そして、キャッシュディレクトリ管理部３６は、アドレス変換部３５から物理アドレスを取得した場合には、ディレクトリをチェックし、物理アドレスが示すデータの状態が正常かチェックする。また、物理アドレスが示すデータをＬ１キャッシュ３２またはＬ２キャッシュ３３がキャッシュしている場合には、キャッシュしているデータを演算部３１に出力する。

【００８３】

一方、キャッシュディレクトリ管理部３６は、物理アドレスが示すデータをＬ１キャッシュ３２またはＬ２キャッシュ３３がキャッシュしていない場合は、物理アドレスが示す記憶領域が、メモリ２２に存在するか否かを判別する。そして、キャッシュディレクトリ管理部３６は、物理アドレスが示す記憶領域がメモリ２２に存在しない場合には、ノードマップ３４を参照する。

【００８４】

また、キャッシュディレクトリ管理部３６は、ノードマップ３４を参照し、取得した物理アドレスを含む範囲のエントリを識別する。そして、キャッシュディレクトリ管理部３６は、識別したエントリのＣＰＵＩＤがＣＰＵ２１のＣＰＵＩＤであるか否かを判別する。その後、キャッシュディレクトリ管理部３６は、識別したエントリのＣＰＵＩＤがＣＰＵ２１のＣＰＵＩＤである場合は、メモリアクセス部４１に、物理アドレスを出力する。

【００８５】

また、キャッシュディレクトリ管理部３６は、識別したエントリのＣＰＵＩＤがＣＰＵ２１のＣＰＵＩＤではない場合には、以下の処理を実行する。すなわち、キャッシュディレクトリ管理部３６は、識別したエントリのＣＰＵＩＤとノードＩＤとを取得する。そして、キャッシュディレクトリ管理部３６は、パケット制御部３７に対して、取得したＣＰＵＩＤと物理アドレスとを出力する。

【００８６】

なお、キャッシュディレクトリ管理部３６は、出力した物理アドレスが示す記憶領域に格納されているデータをメモリアクセス部４１やパケット制御部３７から取得した場合には、取得したデータをＬ１キャッシュ３２およびＬ２キャッシュ３３に格納する。そして、キャッシュディレクトリ管理部３６は、Ｌ１キャッシュ３２にキャッシュさせたデータを演算部３１に出力する。

【００８７】

また、キャッシュディレクトリ管理部３６は、パケット制御部３７から物理アドレスを取得した場合、すなわち、他のＣＰＵからのメモリアクセスのリクエストの対象となる物理アドレスを取得した場合には、以下の処理を実行する。すなわち、キャッシュディレクトリ管理部３６は、取得した物理アドレスのうち、所定の位置のビットが「０」であるか「１」であるかに応じて、取得した物理アドレスがローカル領域に振り分けられた物理アドレスであるか否かを判別する。

【００８８】

例えば、キャッシュディレクトリ管理部３６は、情報処理システム１の各メモリに対して、図３、図４に例示した範囲の物理アドレスが振り分けられている場合には、最下位のビットを０ビット目として４６ビット目が「０」であるか「１」であるかを判別する。そして、キャッシュディレクトリ管理部３６は、４６ビット目が「０」である場合には、取得した物理アドレスがローカル領域に振り分けられた物理アドレスであると判別する。このような場合には、キャッシュディレクトリ管理部３６は、パケット制御部３７に対して、リクエスト元に否定応答（アクセスエラー）を送信するように指示する。

【００８９】

また、キャッシュディレクトリ管理部３６は、４６ビット目が「１」である場合には、取得した物理アドレスが共有領域に振り分けられた物理アドレスであると判別する。このような場合には、キャッシュディレクトリ管理部３６は、取得した物理アドレスが示す記憶領域に記憶されたデータを取得し、取得したデータをパケット制御部３７に出力し、リクエスト元へ送信するよう指示する。

【００９０】

なお、キャッシュディレクトリ管理部３６は、メモリ２２に格納されたデータにアクセスする場合には、物理アドレスが示す記憶領域のデータと、キャッシュされたデータとのコヒーレンスを保持する処理を行う。例えば、キャッシュディレクトリ管理部３６は、キャッシュエントリごとにキャッシュデータの状態を示すキャッシュタグと、ディレクトリとを参照する。そして、キャッシュディレクトリ管理部３６は、キャッシュタグとディレクトリとに基づいて、キャッシュコヒーレンスを保持する処理、および、メモリアクセス処理を実行する。

【００９１】

ここで、図１１Ａは、キャッシュタグの一例を説明するための図である。図１１Ａに示す例では、キャッシュタグは、縮退フラグ、ＥＣＣ（Error Check and Correct memory）チェックビット、ＩＦ（Instruction Fetch）／オプコード、Ｌ１キャッシュステート、Ｌ２キャッシュステート、ＡＡとを有する。

【００９２】

ここで、縮退フラグとは、縮退するか否かを示すキャッシュライン縮退情報である。また、ＥＣＣチェックビットとは、冗長化のために付加されるチェックビットである。ＩＦ／オプコードとは、データがインストラクションであるかデータであるかを示す情報である。

【００９３】

また、ＡＡとは、アドレス情報であり、詳細には、物理アドレスのフレームアドレスが格納される。また、Ｌ１キャッシュステート、および、Ｌ２キャッシュステートとは、Ｌ１キャッシュ３２およびＬ２キャッシュ３３に格納されたデータの状態を示す情報である。

【００９４】

例えば、Ｌ１キャッシュステートや Ｌ２キャッシュステートには、「Ｍ（Modified）」、「Ｅ（Exclusive）」、「Ｓ（Shared）」、「Ｉ（Invalid）」のいずれかを示すビットが格納される。ここで、Ｍｏｄｉｆｉｅｄとは、いずれか１つのＣＰＵがデータをキャッシュしており、かつ、キャッシュされたデータが更新されている状態を示す。なお、キャッシュされたデータの状態がＭｏｄｉｇｉｅｄである場合には、ライトバックを実行する必要がある。

【００９５】

また、Ｅｘｃｌｕｓｉｖｅとは、いずれか１つのＣＰＵがデータをキャッシュしており、かつ、キャッシュされたデータが更新されていない状態を示す。また、Ｓｈａｒｅｄとは、複数のＣＰＵがデータをキャッシュしており、かつ、キャッシュされたデータが更新されていないことを示す。なお、Ｉｎｖａｌｉｄとは、キャッシュのステータスが登録されていないことを示す。

【００９６】

一方、ディレクトリは、２ビットのＣＫビット、６３ビットのＰＲＣ、４ビットのＵＥを管理する。ここで、ＣＫビットとは、キャッシュされたデータの状態をコード化した情報である。また、ＰＲＣとは、当該キャッシュラインのデータをキャッシュしたＣＰＵの位置をビットマップで示す情報である。また、ＵＥとは、ディレクトリの異常と要因とを示す情報である。

【００９７】

キャッシュディレクトリ管理部３６は、取得した物理アドレスに格納されたデータをキャッシュするＣＰＵや、キャッシュされたデータの状態等を識別する。そして、キャッシュディレクトリ管理部３６は、キャッシュされたデータの状態に基づいて、スヌープを発行してメモリのデータを更新する等の処理を行い、キャッシュされたデータとメモリのデータとのコヒーレンスを保持する。その後、キャッシュディレクトリ管理部３６は、データを要求元に出力する。

【００９８】

ここで、キャッシュディレクトリ管理部３６がキャッシュコヒーレンスを保持する処理の一例について説明する。例えば、キャッシュディレクトリ管理部３６は、ステータスがＭ（Modified）であるデータをキャッシュしたＣＰＵに対してライトバックを指示する命令を送信するようパケット生成部３８に指示する。そして、キャッシュディレクトリ管理部３６は、データのステータスを更新し、更新後のステータスに応じた処理を実行する。なお、キャッシュディレクトリ管理部３６が送受信するリクエストや命令の種別については、後述する。

【００９９】

リクエスト生成部３８は、キャッシュディレクトリ管理部３６から物理アドレスと、ＣＰＵＩＤとを取得した場合には、取得した物理アドレスと、ＣＰＵＩＤとを格納したパケット、すなわち、メモリアクセスのリクエストとなるパケットを生成する。そして、リクエスト生成部３８は、生成したパケットをルータ４０に送信する。

【０１００】

ここで、図１１Ｂは、実施例１に係るＣＰＵが送信するパケットを説明するための図である。なお、図１１Ｂに示す例では、物理アドレスをＰＡ（Physical Address）と記載した。図１１Ｂに示す例では、リクエスト生成部３８は、ＣＰＵＩＤと物理アドレスと、リクエストの内容を示すデータとが格納されたリクエストを生成し、生成したリクエストをルータ４０に出力する。このような場合には、ルータ４０は、リクエスト生成部３８が生成したリクエストをＸＢ接続部２７を解してＸＢ２に出力する。すると、ＸＢ２は、リクエストに格納されたＣＰＵＩＤが示すＣＰＵへとリクエストを転送する。

【０１０１】

なお、リクエスト生成部３８は、キャッシュディレクトリ管理部３６からコヒーレンシを保持するためのリクエストや命令の発行の指示を受付けた場合には、指示されたリクエストや命令を生成する。そして、リクエスト生成部３８は、生成したリクエストや命令を

ルータ４０、ＸＢ接続部２７、ＸＢ２を介して、指示されたＣＰＵに送信する。なお、リクエスト生成部３８は、Ｉ／Ｏ装置からデータを取得する場合は、Ｉ／Ｏに対するアクセス要求をルータ４０に出力する。

【０１０２】

図７に戻って、リクエスト受信部３９は、ＸＢ２、ＸＢ接続部２７、ルータ４０を介して、他のＣＰＵが出力したパケットを受信すると、受信したパケットに含まれる物理アドレスを取得する。そして、リクエスト受信部３９は、取得した物理アドレスをキャッシュディレクトリ管理部３６に出力する。また、リクエスト受信部３９は、他のＣＰＵが送信したデータを受信した場合には、受信したデータをキャッシュディレクトリ管理部３６に出力する。

【０１０３】

なお、リクエスト受信部３９は、コヒーレンシを保持するためのリクエストや命令を受信した場合には、受信したリクエストや命令をキャッシュディレクトリ管理部３６に出力する。また、リクエスト受信部３９は、Ｉ／Ｏに対するアクセス要求の応答やデータをルータ４０から受信した場合は、受信した応答やデータをキャッシュディレクトリ管理部３６に出力する。このような場合には、キャッシュディレクトリ管理部３６は、例えば、取得したデータをメモリアクセス部４１に出力し、メモリ２２に格納する処理を行う。

【０１０４】

ルータ４０は、パケット制御部３７が有するリクエスト生成部３８が出力したパケットを受信した場合には、受信したリクエストをＸＢ接続部２７に出力する。また、ルータ４０は、ＸＢ接続部２７を介して、他のＣＰＵが送信したパケットやデータをリクエスト受信部３９に出力する。また、ルータ４０は、パケット制御部３７がＩ／Ｏ等に対して出力したパケットをＰＣＩｅ制御部４２に出力する。また、ルータ４０は、Ｉ／Ｏからの応答等をＰＣＩｅ制御部４２から受信した場合には、受信した応答等をパケット制御部３７に出力する。

【０１０５】

メモリアクセス部４１は、いわゆるＭＡＣ（Memory Access Controller）であり、メモリ２２に対するアクセスの制御を行う。例えば、メモリアクセス部４１は、キャッシュディレクトリ管理部３６から物理アドレスを受信した場合には、受信した物理アドレスに格納されたデータをメモリ２２から取得し、取得したデータをキャッシュディレクトリ管理部３６に出力する。なお、メモリアクセス部４１は、メモリーミラー機能を用いて、共有領域を冗長化してもよい。

【０１０６】

ＰＣＩｅ制御部４２が有するリクエスト生成部４３は、ルータ４０を介してＩ／Ｏに対するアクセス要求を取得した場合には、アクセス要求の対象となるＩ／Ｏ装置に送信するリクエストを生成し、生成したリクエストをＰＣＩｅバス制御部４４に出力する。ＰＣＩｅバス制御部４４は、リクエスト生成部４３が生成したリクエストを取得した場合には、ＰＣＩｅ接続部２８を介して、Ｉ／Ｏ装置にリクエストを送信する。

【０１０７】

次に、図１２を用いて、ＣＰＵ２１が他のＣＰＵに対してリクエストを送信する処理の一例について説明する。図１２は、実施例１に係るＣＰＵがリクエストを送信する処理の一例を説明するための図である。例えば、図１２中（Ａ）に示すように、サービスプロセッサ２４からノードマップ３４に対して、物理アドレスが振り分けられるメモリにアクセスするＣＰＵのＣＰＵＩＤと物理アドレスとを対応付けたエントリの設定が行われる。

【０１０８】

また、演算部３１は、演算処理を実行し、図１２中（Ｂ）に示すように、アクセス対象となる論理アドレスをアドレス変換部３５に出力する。すると、アドレス変換部３５は、論理アドレスを物理アドレスに変換し、変換した物理アドレスを図１２中（Ｃ）に示すように、キャッシュディレクトリ管理部３６に出力する。

【０１０９】

ここで、キャッシュディレクトリ管理部３６は、アドレス変換部３５から物理アドレスを取得すると、図１２中（Ｄ）に示すように、ノードマップ３４を参照し、取得した物理アドレスと対応付けられたＣＰＵＩＤを取得する。そして、キャッシュディレクトリ管理部３６は、取得したＣＰＵＩＤがＣＰＵ２１のＣＰＵＩＤではない場合には、図１２中（Ｅ）に示すように、取得したＣＰＵＩＤと物理アドレスとをパケット制御部３７に出力する。

【０１１０】

このような場合には、リクエスト生成部３８は、キャッシュディレクトリ管理部３６から取得した物理アドレスとＣＰＵＩＤとを格納したパケットを生成し、図１２中（Ｆ）に示すように、生成したパケットをルータ４０に出力する。すると、図１２中（Ｇ）に示すように、ルータ４０は、リクエスト生成部３８から取得したパケットをＸＢ接続部２７に出力する。その後、図１２中（Ｈ）に示すように、ＸＢ接続部２７は、取得したパケットをＸＢ２に出力する。すると、ＸＢ２は、パケットに格納されたＣＰＵＩＤが示すＣＰＵへパケットを伝達することとなる。

【０１１１】

次に、図１３を用いて、ＣＰＵ２１が他のＣＰＵからパケットを受信した際に実行する処理の一例について説明する。図１３は、実施例１に係るＣＰＵがパケットを受信した際に実行する処理の一例を説明するための図である。例えば、図１３中（Ｉ）に示すようにリクエスト受信部３９は、他のＣＰＵからＣＰＵ２１のＣＰＵＩＤとメモリ２２に振り分けられた物理アドレスとが格納されたパケットを受信する。

【０１１２】

このような場合には、リクエスト受信部３９は、受信したパケットから物理アドレスを取得し、図１３中（Ｊ）に示すように、取得した物理アドレスをキャッシュディレクトリ管理部３６に出力する。すると、キャッシュディレクトリ管理部３６は、取得した物理アドレスの４６ビット目が「０」であるか「１」であるかを判別する。

【０１１３】

すなわち、キャッシュディレクトリ管理部３６は、情報処理システム１が図３、図４に示すように、共有領域とローカル領域に振り分ける物理アドレスを設定している場合には、物理アドレスの全ビットを識別せずともよい。すなわち、キャッシュディレクトリ管理部３６は、４６ビット目が「０」であるか「１」であるかを判別するだけで、物理アドレスが示す記憶領域が、共有領域であるかローカル領域であるかを正確に判別することができる。

【０１１４】

そして、キャッシュディレクトリ管理部３６は、受信した物理アドレスの４６ビット目が「１」である場合には、共有領域に対するアクセスであると判別する。このような場合には、キャッシュディレクトリ管理部３６は、図１３中（Ｋ）に示すように、物理アドレスが示す記憶領域のデータがＬ１キャッシュ３２およびＬ２キャッシュ３３にキャッシュされているか判別する。

【０１１５】

また、キャッシュディレクトリ管理部３６は、データがキャッシュされていないと判別した場合には、図１３中（Ｌ）に示すように、物理アドレスをメモリアクセス部４１に出力する。すると、図１３中（Ｍ）に示すように、メモリアクセス部４１は、メモリ２２から物理アドレスが示す記憶領域のデータを取得し、キャッシュディレクトリ管理部３６に出力する。

【０１１６】

そして、キャッシュディレクトリ管理部３６は、Ｌ１キャッシュ３２、Ｌ２キャッシュ３３、またはメモリアクセス部４１からデータを取得した場合には、取得したデータをパケット制御部３７に出力し、リクエスト元のＣＰＵに送信するよう指示する。

【０１１７】

例えば、ＣＰＵ２１～２１ｃ、通信部２３、サービスプロセッサ２４、制御部２５、通

信部２６、ＸＢ接続部２７、ＰＣＩｅ接続部２８とは、電子回路である。また、演算部３１、アドレス変換部３５、キャッシュディレクトリ管理部３６、パケット制御部３７、リクエスト生成部３８、リクエスト受信部３９とは、電子回路である。

【０１１８】

また、ルータ４０、メモリアクセス部４１、ＰＣＩｅ制御部４２、リクエスト生成部４３、ＰＣＩｅバス制御部４４とは、電子回路である。ここで、電子回路の例として、ＡＳＩＣ（Application Specific Integrated Circuit）やＦＰＧＡ（Field Programmable Gate Array）などの集積回路、またはＣＰＵ（Central Processing Unit）やＭＰＵ（Micro Processing Unit）などを適用する。

【０１１９】

また、メモリ２２～２２ａとは、ＲＡＭ（Random Access Memory）、ＲＯＭ（Read Only Memory）、フラッシュメモリ（flash memory）などの半導体メモリ素子である。また、Ｌ１キャッシュ３２、Ｌ２キャッシュ３３は、ＳＲＡＭ（Static Random Access Memory）等の高速な半導体メモリ素子である。

【０１２０】

次に、各ＣＰＵ２１～２１ｃがキャッシュコヒーレンスを保持する処理について簡単に説明する。なお、以下の説明においては、情報処理システム１の各ＣＰＵはイリノイプロトコルを用いて、キャシュコヒーレンスを保持するものとする。

【０１２１】

なお、以下の説明においては、情報処理システム１が有する各メモリは、全てのＣＰＵからキャッシュ可能な空間を有するメモリとして識別されるものとする。また、以下の説明においては、キャッシュ対象となるデータを記憶するメモリに、そのＣＰＵ内のＭＡＣを介して物理的に直接接続されているＣＰＵをホームＣＰＵとし、キャッシュを要求したＣＰＵをローカルＣＰＵと記載する。

【０１２２】

また、ホームＣＰＵに対して既にリクエストを送信し、データをキャッシュ済みであるＣＰＵをリモートＣＰＵと記載する。なお、ローカルＣＰＵとホームＣＰＵが同一のＣＰＵとなる場合や、ローカルＣＰＵとリモートＣＰＵとは同一のＣＰＵとなる場合も存在する。

【０１２３】

例えば、ローカルＣＰＵは、自身のノードマップを参照し、アクセス対象となる物理アドレスがホームＣＰＵがアクセスするメモリに振り分けられていると判別する。そして、ローカルＣＰＵは、物理アドレスを格納したリクエストをホームＣＰＵに対して発行する。なお、ローカルＣＰＵが発行するリクエストには、複数の種別のリクエストが存在する。このため、ホームＣＰＵが有するキャッシュディレクトリ管理部は、取得したリクエストの種別に応じたキャッシュコヒーレンス制御を実行することとなる。

【０１２４】

例えば、ローカルＣＰＵが発行するリクエストの種別としては、共有型フェッチアクセス、排他型フェッチアクセス、キャッシュ無効化要求、キャッシュリプレース要求等が存在する。共有型フェッチアクセスとは、ＭｏｖｅＩｎ ｔｏ Ｓｈａｒｅの実行要求であり、ホームＣＰＵがアクセスするメモリからデータの読出しを行う際に発行されるリクエストである。

【０１２５】

また、排他型フェッチアクセスとは、例えばＭｏｖｅＩｎ Ｅｘｃｌｕｓｉｖｅｌｙの実行要求であり、ホームＣＰＵがアクセスするメモリへデータストアを行う際の、キャッシュへのデータロードを行う際に発行される。また、キャッシュ無効化要求とは、例えばＭｏｖｅＯｕｔの実行要求であり、キャッシュラインの無効化をホームＣＰＵに対して要求する際に発行される。なお、ホームＣＰＵは、キャッシュ無効化要求を受信すると、リモートＣＰＵに対してキャッシュ無効化要求を発行する場合や、キャッシュをＩｎｖａｌｉｄａｔｉｏｎとさせる命令を発行する場合がある。

【０１２６】
キャッシュリプレース要求とは、例えばＷｒｉｔｅＢａｃｋの実行要求であり、更新されたキャッシュデータ、すなわちＭｏｄｉｆｉｅｄ状態のキャッシュデータをホームＣＰＵがアクセスするメモリに書き戻す際に発行される。なお、キャッシュリプレース要求には、例えば、ＦｌｕｓｈＢａｃｋの実行要求であり、更新されていないキャッシュデータ、すなわち、Ｓｈａｒｅｄ又はＥｘｃｌｕｓｉｖｅ状態のキャッシュの破棄を行う際に発行される。
【０１２７】
ホームＣＰＵは、上述したリクエストをローカルＣＰＵから受信した場合には、リクエストを処理するために、ローカルＣＰＵやリモートＣＰＵに対して、命令を発行する。ここで、ホームＣＰＵは、取得したリクエストの種別に応じてたキャッシュコヒーレンス制御を実行するため、複数の種別の命令を発行することとなる。例えば、ホームＣＰＵは、リモートＣＰＵがキャッシュしているデータをローカルＣＰＵにロードさせるＭｏｖｅＯｕｔ ａｎｄ Ｂｙｐａｓｓ ｔｏ Ｓｈａｒｅを発行する。
【０１２８】
また、例えば、ホームＣＰＵは、ローカルＣＰＵ以外のすべてのリモートＣＰＵのキャッシュを無効化し、その後、ホームＣＰＵがローカルＣＰＵにデータを送信するためのＭｏｖｅＯｕｔ ａｎｄ Ｂｙｐａｓｓ Ｅｘｃｌｕｓｉｖｅｌｙを発行する。また、ホームＣＰＵは、リモートＣＰＵにキャッシュの無効化を要求するＭｏｖｅＯｕｔ ＷＩＴＨ Ｉｎｖａｌｉｄａｔｉｏｎを発行する。なお、ホームＣＰＵがＭｏｖｅＯｕｔ ＷＩＴＨ Ｉｎｖａｌｉｄａｔｉｏｎを発行した場合には、全てのＣＰＵのキャッシュが、対象となるアドレスについてＩｎｖａｌｉｄａｔｅ状態となる。
【０１２９】
また、ホームＣＰＵは、リモートＣＰＵにキャッシュラインの無効化を要求するＭｏｖｅＯｕｔ ｆｏｒ Ｆｌｕｓｈを発行する。なお、ホームＣＰＵがＭｏｖｅＯｕｔ ｆｏｒ Ｆｌｕｓｈを発行した場合には、対象となるデータを、ホームＣＰＵのみがキャッシュした状態となる。また、ホームＣＰＵは、対象となるデータの状態がＳｈａｒｅｄであるときに、リモートＣＰＵにキャッシュの破棄を要求するＢｕｆｆｅｒ Ｉｎｖａｌｉｄａｔｉｏｎを発行する。
【０１３０】
ホームＣＰＵは、リクエストの種別に応じて、上述した命令を発行し、各ＣＰＵがキャッシュしたデータのステートを遷移させる。また、ローカルＣＰＵやリモートＣＰＵは、命令を受信した場合には、命令が示す処理を実行し、自身がキャッシュしたデータのステートを遷移させる。
【０１３１】
その後、ローカルＣＰＵやリモートＣＰＵは、命令に対する完了応答やデータ付の完了応答をホームＣＰＵに送信する。また、ホームＣＰＵやリモートＣＰＵは、命令処理を実行した後に、ローカルＣＰＵに対して、データ付のリクエスト応答を送信することとなる。
【０１３２】
［ＣＰＵの処理の流れ］
次に、図１４を用いて、情報処理システム１において各ＣＰＵが有するノードマップ３４を設定する処理の流れについて説明する、図１４は、ノードマップを設定する処理の流れを説明するためのフローチャートである。なお、以下の説明においては、１つのＣＰＵとＣＰＵがアクセスするメモリとの組をノードとして記載する。また、以下の説明においては、新たなノードを情報処理システム１に追加する例について説明する。
【０１３３】
まず、情報処理システム１のオペレータは、ノードの新規増設を行う（ステップＳ１０１）。次に、各ビルディングブロック１０～１０ｅのサービスプロセッサが追加されたノードのハードウェアの構成の読み取りを行う（ステップＳ１０２）。次に、情報処理シス

テム１のオペレータは、新たなノードが有するメモリの共有領域の割り当てをサービスプロセッサに指示する（ステップＳ１０３）。

【０１３４】

次に、情報処理システム１のオペレータは、新たなノードのサービスプロセッサに電源投入を指示する（ステップＳ１０４）。すると、各ビルディングブロック１０～１０ｅのサービスプロセッサは、読み取った構成の情報を元に、各ビルディングブロック１０～１０ｅが有するＣＰＵのノードマップ３４をＩ２Ｃを用いて設定する（ステップＳ１０５）。その後、情報処理システム１は、各ビルディングブロック１０～１０ｅの電源投入を行い（ステップＳ１０６）、処理を終了する。

【０１３５】

次に、図１５を用いて、情報処理システム１が共有領域を制御する処理の流れについて説明する。図１５は、共有領域を制御する処理の流れを説明するためのフローチャートである。まず、情報処理システム１は、アプリケーションの要求に応じて、ノード間の共有メモリの割当処理を実行する（ステップＳ２０１）。次に、情報処理システム１は、ノード間で共有する共有メモリのアタッチ処理を実行する（ステップＳ２０２）。

【０１３６】

その後、情報処理システム１が有する各ＣＰＵによって実行されるアプリケーションが各メモリを使用する（ステップＳ２０３）。次に、情報処理システム１は、共有メモリのデタッチ処理を実行する（ステップＳ２０４）。その後、情報処理システム１は、共有メモリの解放処理を実行し（ステップＳ２０５）、処理を終了する。なお、ステップＳ２０１、および、ステップＳ２０５は、その共有メモリのホームノード上のアプリケーションのみが実施してもよいし、実際の処理は、ｎｏｐとなるものの、その共有メモリのホームノード以外のノード上のアプリケーションも実施するものとしても良い。

【０１３７】

次に、図１６を用いて、図１５中ステップＳ２０１で示した共有メモリの割当処理を実行する処理の流れについて説明する。図１６は、共有メモリの割当処理を説明するためのフローチャートである。図１６に示す例では、例えば、ＣＰＵ２１が実行するアプリケーションがＯＳに対して、ノード間の共有メモリ割当処理の実行を要求する（ステップＳ３０１）。

【０１３８】

すると、ＣＰＵ２１が実行するＯＳが共有領域用の物理アドレスの領域から要求されたサイズのメモリ割当を行う（ステップＳ３０２）。次に、ＯＳが割り当てた共有メモリの管理用ＩＤをアプリケーションに引渡し（ステップＳ３０３）、共有メモリの割当処理を終了する。

【０１３９】

次に、図１７を用いて、図１５中ステップＳ２０２で示したノード間の共有メモリアタッチ処理の流れについて説明する。図１７は、共有メモリアタッチ処理を説明するためのフローチャートである。まず、アプリケーションは、ＯＳに対して管理用ＩＤを引渡し、ノード間の共有メモリのアタッチ処理を要求する（ステップＳ４０１）。このような場合には、ＯＳは、他のノードで実行されているＯＳと通信を行い、管理用ＩＤに対応する物理アドレスを獲得する（ステップＳ４０２）。

【０１４０】

ここで、ＯＳが他のノードで実行されているＯＳと通信を行う場合には、ＬＡＮ（Local Area Network）などによる通信、サービスプロセッサ２４を介した各ノード間の通信等を用いる。また、例えば、各ノードで実行されるＯＳは、特定の共有領域を、ノード間通信に用いる領域として設定し、設定した領域に対する情報の格納や読み取りを行う事で、通信を行うこととしても良い。

【０１４１】

次に、ＯＳは、物理アドレスに対応する論理アドレス（Virtual Address）を決定し、割当を行う（ステップＳ４０３）。例えば、ＣＰＵ２１で実行されるＯＳは、物理アドレ

スと論理アドレスとのＴＬＢをアドレス変換部３５に設定する。
【０１４２】
なお、各ＣＰＵ２１～２１ｃが用いる論理アドレスは、重複する範囲であっても良く、また、ＣＰＵごとに異なる範囲でもよい。また、各ＣＰＵ２１～２１ｃが用いる論理アドレスは、アプリケーションがＯＳに指定できるようにしてもよい。その後、ＯＳは、論理アドレスの値をアプリケーションに引渡し（ステップＳ４０４）、処理を終了する。
【０１４３】
次に、図１８を用いて、図１５中ステップＳ２０３で示したアプリケーションがノード間の共有メモリを使用する処理の流れについて説明する。図１８は、アプリケーションが共有メモリを使用する処理を説明するためのフローチャートである。例えば、ＣＰＵ２１が実行するアプリケーションは、論理アドレスを発行し、論理アドレスが示す記憶領域へのアクセスを行う（ステップＳ５０１）。
【０１４４】
すると、ＣＰＵ２１は、ＴＬＢミスが発生したか否かを判別する（ステップＳ５０２）。そして、ＣＰＵ２１は、ＴＬＢミスが発生した場合は（ステップＳ５０２肯定）、トラップ処理を実行し、ＴＬＢに論理アドレスと物理アドレスとの組のエントリを設定する（ステップＳ５０３）。
【０１４５】
次に、アプリケーションは、再度論理アドレスを発行し、ＴＬＢによる物理アドレスへの変換を経て、正常に共有メモリに対するアクセスを実行する（ステップＳ５０４）。一方、ＴＬＢミスが発生しなかった場合は（ステップＳ５０２否定）、正常に共有メモリに対するアクセスが実行され（ステップＳ５０５）、処理が終了する。
【０１４６】
次に、図１９を用いて、図１５中ステップＳ２０４で示したノード間の共有メモリデタッチ処理の流れについて説明する。図１９は、ノード間の共有メモリデタッチ処理を説明するためのフローチャートである。例えば、ＣＰＵ２１が実行するアプリケーションは、ＯＳに対して、ノード間共有メモリの論理アドレス、または管理用ＩＤを指定して、デタッチ処理を要求する（ステップＳ６０１）。
【０１４７】
すると、ＣＰＵ２１が実行するＯＳは、キャッシュのフラッシュを行う（ステップＳ６０２）。すなわち、ＯＳは、共有メモリの割り当て解除後、再度共有メモリとして割当てを行った場合に、共有メモリとして割当てが行われていない際に共有メモリの実メモリにアクセスするＣＰＵがリブートすると、キャッシュと実メモリの状態が食い違う恐れがある。このため、ＯＳは、キャッシュのフラッシュを行い、キャッシュと実メモリの状態とが食い違う状態を防止する。
【０１４８】
そして、ＯＳは、ノード間共有メモリ、すなわち、アプリケーションが利用していた範囲の論理アドレスの割当を解除し、解除した論理アドレスに関連するＴＬＢのエントリを削除する（ステップＳ６０３）。また、ＯＳは、ノード間で通信を行い、本アプリケーションが対象ＰＡの使用を完了したことを通知する（ステップＳ６０４）。そして、ＯＳは、ノード間通信により、解放済みの共有メモリについて、最後の利用者がデタッチを行ったことをホームノードが認識した場合、指定された共有メモリ用のメモリ割当て解除を行う（ステップＳ６０５）。なお、ステップＳ６０５の処理は、図２０に示すステップＳ７０２の処理と関連する。
【０１４９】
なお、ステップＳ６０３以降は、ＯＳは、本ノード上で、デタッチが完了しているメモリアドレスについてＴＬＢミス（ステップＳ５０２肯定）が発生しても、デタッチが完了している論理アドレスに対応する物理アドレスをＴＬＢに設定しない。このような場合には、ステップＳ５０４の処理は、正常に終了せず、アクセスエラーとなる。また、デタッチ完了後、ステップＳ４０２と逆に、ＯＳがノード間で通信し、本アプリケーションがこ

の共有メモリのＰＡに対してアクセスを完了したことを通知する。もし、この共有メモリがホームノード上で解放済みで、かつ、このアプリケーションがこの共有メモリの最後の利用者であった場合は、ホームノードに解放処理を依頼する。

【０１５０】

次に、図２０を用いて、図１５中ステップＳ２０５で示したノード間共有メモリの解放処理の流れについて説明する。図２０は、ノード間共有メモリの解放処理を説明するためのフローチャートである。例えば、ＣＰＵ２１が実行するアプリケーションは、ＯＳに対してノード間共有メモリの解放処理を要求する（ステップＳ７０１）。すると、ＯＳは、指定された共有領域の使用者が全てでタッチしていた場合は、割当てを解放し（ステップＳ７０２）、処理を終了する。もし、デタッチが完了していなければ、割当ての解放処理は行わず、処理を終了する。なお、実際の割当ての完了処理は、ステップＳ６０５で行われる。

【０１５１】

次に、図２１を用いて、ＣＰＵ２１が他のＣＰＵに対して、メモリアクセスのリクエストを送信する処理の流れについて説明する。図２１は、リクエストを発行する処理の流れを説明するためのフローチャートである。例えば、ＣＰＵ２１の演算部は、論理アドレスを発行する（ステップＳ８０１）。

【０１５２】

すると、アドレス変換部３５において、論理アドレスから物理アドレスへの変換が行われる（ステップＳ８０２）。次に、キャッシュディレクトリ管理部３６が、物理アドレスを取得し、キャッシュディレクトリ管理を実行する（ステップＳ８０３）。すなわち、キャッシュディレクトリ管理部３６は、取得した物理アドレスが示す記憶領域についてのキャッシュステートを遷移させる。

【０１５３】

次に、キャッシュディレクトリ管理部３６は、ノードマップ３４を参照し、取得した物理アドレスが他ノードのメモリに振り分けられた物理アドレスであるか否かを判別する（ステップＳ８０４）。そして、キャッシュディレクトリ管理部３６は、取得した物理アドレスが他ノードのメモリに振り分けられた物理アドレスではないと判別した場合には（ステップＳ８０４否定）、取得した物理アドレスを用いてメモリアクセスを実行する（ステップＳ８０５）。

【０１５４】

一方、キャッシュディレクトリ管理部３６は、取得した物理アドレスが他ノードのメモリに振り分けられた物理アドレスである場合には（ステップＳ８０４肯定）、ノードマップ３４から物理アドレスと対応付けられたＣＰＵＩＤを取得する（ステップＳ８０６）。そして、パケット送信部が、ＣＰＵＩＤと物理アドレスとを格納したパケット、すなわち、メモリアクセスのリクエストを生成し、ＸＢ２に送出し（ステップＳ８０７）、処理が終了する。

【０１５５】

次に、図２２を用いて、ＣＰＵ２１が他のＣＰＵからメモリアクセスのリクエストを受信した際に実行する処理の流れについて説明する。図２２は、リクエストを受信した際に実行する処理の流れを説明するためのフローチャートである。なお、図２２に示す例では、ＣＰＵ２１が、他のＣＰＵからＭｏｖｅＩｎ ｔｏ ＳｈａｒｅやＭｏｖｅＩｎ Ｅｘｃｌｕｓｉｖｅｌｙを受信した際に実行する処理の流れについて説明する。例えば、ＣＰＵ２１は、他のＣＰＵからＸＢ２を介してリクエストを受信する（ステップＳ９０１）。

【０１５６】

このような場合には、ＣＰＵ２１は、リクエストの対象となる物理アドレスの所定のビットが「１」であるか否かを判別することで、リクエストの対象となる物理アドレスがローカル領域であるか否かを判別する（ステップＳ９０２）。そして、ＣＰＵ２１は、リクエストの対象となる物理アドレスがローカル領域であると判別した場合には（ステップＳ９０２肯定）、リクエスト元のＣＰＵに否定応答を返信し（ステップＳ９０３）、処理を

終了する。

【０１５７】

また、ＣＰＵ２１は、リクエストの対象となる物理アドレスがローカル領域でない場合には（ステップＳ９０２否定）、コヒーレンスを保持するキャッシュディレクトリ管理を実行する（ステップＳ９０４）。また、ＣＰＵ２１は、物理アドレスが示す記憶領域のステータスを判定する（ステップＳ９０５）。

【０１５８】

そして、ＣＰＵ２１は、判定したステータスに応じた命令を他のＣＰＵに対して発行し（ステップＳ９０６）、ステータスを遷移させる（ステップＳ９０７）。その後、ＣＰＵ２１は、物理アドレスが示す記憶領域のデータをリクエスト元のＣＰＵに送信する応答を行い（ステップＳ９０８）、処理を終了する。

【０１５９】

次に、図２３を用いて、ＣＰＵ２１が応答を受信した際に実行する処理の流れについて説明する。図２３は、ＣＰＵが応答を受信した際に実行する処理の流れを説明するためのフローチャートである。例えば、ＣＰＵ２１は、応答を受信する（ステップＳ１００１）。このような場合には、ＣＰＵ２１は、応答の内容が正常な応答であるか否かを判別する（ステップＳ１００２）。

【０１６０】

そして、ＣＰＵ２１は、応答の内容が正常である場合、すなわち、リクエスト対象となるデータを受信した場合には（ステップＳ１００２肯定）、データを用いた正常な処理を実行し（ステップＳ１００３）、処理を終了する。一方、ＣＰＵ２１は、否定応答を受信した場合は（ステップＳ１００２否定）、否定応答の理由がアクセスエラーであるか否かを判別する（ステップＳ１００４）。

【０１６１】

そして、ＣＰＵ２１は、否定応答の理由がアクセスエラーではない場合には（ステップＳ１００４否定）、通常のエラー処理を実行し（ステップＳ１００５）、処理を終了する。一方、ＣＰＵ２１は、否定応答の理由がアクセスエラーではない場合には（ステップＳ１００４肯定）、エラーが発生した物理アドレスをエラーレジスタに設定して、トラップ処理を実行し（ステップＳ１００６）、処理を終了する。

【０１６２】

［実施例１の効果］

上述したように、情報処理システム１は、ＣＰＵ２１～２１ｃとメモリ２２～２２ｃと、各ＣＰＵ２１～２１ｃを接続するＸＢ２とを有する。また、ＣＰＵ２１は、論理アドレスと物理アドレスとの変換を行うアドレス変換部と、物理アドレスとＣＰＵＩＤとを変換するノードマップ３４を有する。

【０１６３】

そして、ＣＰＵ２１は、物理アドレスとＣＰＵＩＤとを有するリクエストのパケットを送信する。また、ＣＰＵ２１は、他のＣＰＵからリクエストのパケットを受信した場合には、受信したパケットに格納された物理アドレスに基づいて、アクセス対象となる記憶領域が、共有領域であるかローカル領域であるかを判別する。

【０１６４】

このようにすることで、情報処理システム１は、効率的、かつ小さなハードウェア物量でノード間共有メモリに対するメモリアクセスを行うことができる。すなわち、情報処理システム１は、ＣＰＵ２１が物理アドレスとＣＰＵＩＤとを変換するノードマップ３４を用いてアドレス変換を行うので、効率的なメモリアクセスを行うことができる。

【０１６５】

また、ＣＰＵ２１は、他のＣＰＵがアクセスするメモリの共有領域にアクセスする場合には、物理アドレスとＣＰＵＩＤを格納したパケットをＸＢ２に送出するのみでよい。このため、情報処理システム１は、効率的なメモリアクセスを行うことができる。

【０１６６】

また、情報処理システム１は、ＣＰＵ２１が他のＣＰＵからリクエストのパケットを受信した場合には、受信したパケットに格納された物理アドレスに基づいて、アクセス対象となる記憶領域が、共有領域であるかローカル領域であるかを判別する。このため、情報処理システム１は、ローカル領域に格納するカーネルデータやユーザデータのセキュリティレベルを高く保つことができる。また、情報処理システム１は、全てのメモリをキャッシュ可能とするので、メモリアクセスにおけるレイテンシを容易に隠蔽することができる。

【０１６７】

また、ＣＰＵ２１は、他のＣＰＵがアクセスするメモリの共有領域に対して、メモリ２２にアクセスする場合と同様の方法でアクセスする。すなわち、ＣＰＵ２１が有する演算部３１は、アクセス対象となる記憶領域がメモリ２２上に存在する場合にも、他のメモリ上に存在する場合にも、論理アドレスを出力するだけでよい。

【０１６８】

このため、情報処理システム１は、Ｉ／Ｏの排他制御等の処理やプログラミング等を実行せずとも、容易に共有領域にアクセスできるため、メモリアクセス性能を向上させることができる。また、ＣＰＵ２１は、実行するプログラムやＯＳに改変を行わずとも、共有メモリを適切に利用することができる結果、プリフェッチ処理の実行を従来と同様に行う事ができるため、メモリアクセスの性能を向上させることができる。

【０１６９】

また、情報処理システム１は、所定のビットが「１」となる物理アドレスを共有領域に割り当て、所定のビットが「０」となる物理アドレスをローカル領域に割り当てる。このため、ＣＰＵ２１は、物理アドレスのうち、所定の１ビットが「１」であるか否かを判別するだけで、アクセス対象の物理アドレスが共有領域の物理アドレスであるか否かを容易に判別することができる。この結果、情報処理システム１は、効率的なメモリアクセスを行うことができる。

【０１７０】

また、ＣＰＵ２１は、他のＣＰＵからのメモリアクセスの対象がローカル領域へのアクセスであると判定した場合には、否定応答を返信する。このため、情報処理システム１は、共有領域以外へのアクセスを防止する結果、エラーを防ぐことができる。

【０１７１】

また、キャッシュディレクトリ管理部３６は、ノードマップ３４を用いて、物理アドレスをノードマップ３４に対応付けて記憶されたＣＰＵＩＤに変換する。このため、ＣＰＵ２１は、アクセス対象となる物理アドレスが振り分けられたメモリにアクセスするＣＰＵを識別することができる。

【０１７２】

また、各ビルディングブロック１０～１０ｅは、ノードマップ３４の書き換えを行うサービスプロセッサを有する。このため、情報処理システム１は、メモリ２２～２２ｃごとに、ローカル領域と共有領域とを自由に割り当てることができる。例えば、情報処理システム１は、メモリ２２が４ＴＢの容量を有する場合に、ローカル領域に１ＴＢを割り当て、共有領域に３ＴＢを割り当てるというように、任意の容量の記憶領域をノード間で共有することができる。

【０１７３】

また、情報処理システム１は、新たなＣＰＵとメモリを追加した場合やＣＰＵやメモリの削除を行った場合にも、サービスプロセッサを介して容易にローカル領域と共有領域との割り当てを行うことができる。

【０１７４】

また、ＣＰＵ２１は、メモリ２２に記憶されたデータをキャッシュしたＣＰＵを管理するディレクトリを用いて、キャッシュコヒーレンスの制御を行う。このため、情報処理システム１は、情報処理システム１が有するＣＰＵの数が増加した場合にも、ＸＢ２のトラフィックを増加させることなく、効率的にキャッシュコヒーレンスを保持することができ

る。
【０１７５】
具体的には、情報処理システム１においては、各ＣＰＵ間の通信が、リモートＣＰＵとホームＣＰＵ間、または、リモートＣＰＵとホームＣＰＵと更新したデータをキャッシュするローカルＣＰＵ間に限定される。このため、情報処理システム１は、効率的にキャッシュコヒーレンスを保持することができる。
【０１７６】
また、ＣＰＵ２１は、キャッシュミスが発生した場合に、キャッシュミスした物理アドレスが他のＣＰＵがアクセスするメモリに割り当てられた物理アドレスであるか否かを判別する。そして、ＣＰＵ２１は、キャッシュミスした物理アドレスが他のＣＰＵがアクセスするメモリに割り当てられた物理アドレスであると判別した場合には、物理アドレスをＣＰＵＩＤに変換し、物理アドレスとＣＰＵＩＤとを格納したパケットの生成および送出を行う。このため、ＣＰＵ２１は、無駄なアドレス変換処理を行うことなくメモリアクセスを行うことができる。
【０１７７】
また、ＣＰＵ２１は、実行するアプリケーションが共有領域の獲得を要求した場合には、アプリケーションが利用する論理アドレスと、共有領域に割り当てられる物理アドレスとを変換するＴＬＢを設定する。このため、ＣＰＵ２１は、実行するアプリケーションやＯＳに共有領域やローカル領域へのアクセスを意識した改変を加えずとも、メモリアクセスを行うことができる。
【実施例２】
【０１７８】
これまで本発明の実施例について説明したが実施例は、上述した実施例以外にも様々な異なる形態にて実施されてよいものである。そこで、以下では実施例２として本発明に含まれる他の実施例を説明する。
【０１７９】
（１）ビルディングブロックについて
上述した情報処理システム１は、４つのＣＰＵを有するビルディングブロック１０～１０ｅを有していた。しかし、実施例はこれに限定されるものではなく、ビルディングブロック１０～１０ｅは、任意の数のＣＰＵ及び各ＣＰＵがアクセスするメモリを有することができる。また、ＣＰＵとメモリは、１対１で対応している必要はなく、メモリに直接アクセスするＣＰＵは全体の一部であってもよい。
【０１８０】
（２）共有領域とローカル領域の割り当てについて
上述した共有領域とローカル領域に対する物理アドレスの割り当ては、あくまで一例であり、情報処理システム１は、任意の物理アドレスを各領域に割当てることができる。
【０１８１】
例えば、情報処理システム１は、物理アドレスの最下位１ビットが「０」となる物理アドレスを共有領域に割り当て、物理アドレスの最下位１ビットが「１」となる物理アドレスを共有領域に割当てることとしてもよい。このような場合には、各ＣＰＵは、物理アドレスの最下位１ビットが「０」であるか「１」であるかを判別することで、アクセス対象が共有領域であるかを容易に判別できる。
【０１８２】
また、情報処理システム１は、物理アドレス空間の前半に含まれる任意の物理アドレスを共有領域に割り当て、物理アドレス空間の後半に含まれる任意の物理アドレスをローカル領域に割当ててもよい。このような場合には、各ＣＰＵは、物理アドレスの最上位１ビットが「０」であるか「１」であるかを判別することで、アクセス対象が共有領域であるかを容易に判別できる。なお、情報処理システム１は、物理アドレス空間の前半に含まれる任意の物理アドレスをローカル領域に割当て、後半に含まれる任意の物理アドレスを共有領域に割当ててもよい。

【０１８３】
すなわち、情報処理システム１は、任意の物理アドレスを共有領域とローカル領域に割当てることができるが、所定のビットが同一の値となる物理アドレスを共有領域に割り当て、所定のビットが共有領域とは異なる値の物理アドレスをローカル領域に割当てることで、アクセス対象が共有領域であるかローカル領域であるかを容易に判別できる。
【０１８４】
（３）ＣＰＵが送信するパケットについて
上述したＣＰＵ２１は、ＣＰＵＩＤとＰＡとを有するパケットをメモリアクセスのリクエストとして送信した。しかし、実施例は、これに限定されるものではない。すなわち、ＣＰＵ２１は、アクセス対象となるメモリにアクセスするＣＰＵを一意に識別できるのであれば、任意の情報を格納したパケットを出力してよい。
【０１８５】
また例えば、ＣＰＵ２１は、ＣＰＵＩＤからＶＣ（Virtual Connection）ＩＤに変換し、ＶＣＩＤを格納することとしてもよい。また、ＣＰＵ２１は、パケットに、データ長を示すレングス等の情報を格納することとしてもよい。
【０１８６】
（４）ＣＰＵが発行する命令について
上述したように、各ＣＰＵ２１～２１ｃは、リクエストや命令を発行して、キャッシュのコヒーレンスを保持した。しかし、上述したリクエストや命令は、あくまで一例であり、例えばＣＰＵ２１～２１ｃは、ＣＡＳ（Ｃｏｍｐａｒｅ ＡｎｄＳｗａｐ）命令を発行してもよい。
【０１８７】
このように、ＣＰＵ２１～２１ｃがＣＡＳ命令を発行した場合には、排他制御のコンテンションが複数のＣＰＵ間で頻発しても、各ＣＰＵのキャッシュ上で処理が行われる。この結果、ＣＰＵ２１～２１ｃは、メモリアクセスの発生による遅延を防止するとともに、各ＣＰＵ間のドランザクションが混雑するのを防ぐことができる。
【０１８８】
（５）ハイパーバイザを経由した制御について
上述した情報処理システム１では、ＯＳによってハードウェアであるアドレス変換部３５にアクセスを行う例について説明した。しかし、実施例はこれに限定されるものではなく、たとえは、仮想マシンを動作させるハイパーバイザ（ＨＰＶ：Hypervisor）がアドレス変換部３５にアクセスを行っても良い。
【０１８９】
すなわち、ハイパーバイザが動作するノードにおいては、ＯＳは、キャッシュやＭＭＵなどのＣＰＵ２１～２１ｃのハードウェア資源に対して直接の操作を行わず、操作をハイパーバイザに依頼することとなる。このように、各ＣＰＵ２１～２１ｃは、ハイパーバイザを介した制御を受付ける場合には、仮想アドレスを実アドレス（ＲＡ：Real Address）に変換し、その後、実アドレスを物理アドレスに変換することとなる。
【０１９０】
また、ハイパーバイザが動作するノードにおいては、割り込み処理は、ＯＳには直接割り込まず、ＨＰＶに対して割り込みを行う。このような場合には、ハイパーバイザが、ＯＳの割り込み処理ハンドラを読出すことで割り込みを行う。なお、上述したハイパーバイザが実行する処理は、仮想マシンを動作させるために実行される公知な処理である。
【０１９１】
（６）パーティションを用いた処理について
上述した情報処理システム１では、各ＣＰＵ２１～２１ｃは、１つのノードマップを用いてメモリアクセスを送信していた。しかし、実施例はこれに限定されるものではない。例えば、各ビルディングブロック１０～１０ｅは、複数のノード群として動作し、各ノード群毎に、同一のファームウェア（ハイパーバイザ）を動作させる１つの論理パーティションを構成しても良い。

【０１９２】
このような場合には、各ＣＰＵ２１～２１ｃは、アクセス先のＣＰＵを示すノードマップと、同一論理パーティション内のＣＰＵを示すノードマップとを有する。このように、各ＣＰＵ２１～２１ｃは、同一論理パーティション内に含まれるＣＰＵを示すノードマップを有することで、エラー発生通知、ダウン要求、リセット要求パケット等の、論理パーティションを超えて転送すべきではない特殊パケットの転送範囲を識別することができる。

【０１９３】
以下、同一論理パーティション内に含まれるＣＰＵを示すノードマップを有するＣＰＵについて説明する。図２４は、実施例２に係る情報処理システムを説明するための図である。図２４に示すように、ビルディングブロック１０、１０ａは、論理パーティション＃Ａを動作させ、ビルディングブロック１０ｂ～１０ｄは、論理パーティション＃Ｂを動作させる。

【０１９４】
ここで、論理パーティション＃Ａでは、複数のドメイン＃Ａ～＃Ｃと、ファームウェア＃Ａが動作する。また、論理パーティション＃Ｂでは、複数のドメイン＃Ｄ～＃Ｇとファームウェア＃Ｂが動作する。なお、ファームウェア＃Ａおよびファームウェア＃Ｂとは、例えばハイパーバイザである。また、ドメイン＃Ａでは、アプリケーションとＯＳとが動作しており、他のドメイン＃Ｂ～＃Ｇもドメイン＃Ａと同様に、アプリケーションとＯＳとが動作する。

【０１９５】
つまり、各ドメイン＃Ａ～＃Ｇは、それぞれ独立してアプリケーションとＯＳが動作する仮想マシンである。ここで、ビルディングブロック１０が有する各ＣＰＵ２１～２１ｃは、パーティション＃Ａに含まれる各ＣＰＵに対して上述した特殊パケットを送信してもよいが、パーティション＃Ｂに含まれる各ＣＰＵに対しては特殊パケットを送信すべきではない。

【０１９６】
このため、各ビルディングブロック１０～１０ｄのＣＰＵは、同一の論理パーティションに含まれるＣＰＵのＣＰＵＩＤを示すノードマップを有する。例えば、ＣＰＵ２１は、物理アドレスと、物理アドレスが示す記憶領域を有するメモリと接続されたＣＰＵのＣＰＵＩＤとを対応付けて記憶するノードマップ３４を有する。また、ＣＰＵ２１は、ＣＰＵ２１と同一のパーティション、すなわち、パーティション＃Ａに含まれるＣＰＵのＣＰＵＩＤを記憶するノードマップ３４ａを有する。なお、ノードマップ３４ａは、ノードマップ３４と同様に、サービスプロセッサ２４によって設定されるものとする。

【０１９７】
以下、図面を用いて、同一の論理パーティションに含まれるＣＰＵのＣＰＵＩＤを示すノードマップの一例について説明する。図２５は、パーティションの一例を説明するための図である。例えば、図２５に示す例では、パーティション＃Ａは、ビルディングブロック＃０を有する。また、ビルディングブロック＃０は、ＣＰＵ＃０とアドレス域「＃０」が割当てられたメモリとを有する。

【０１９８】
また、パーティション＃Ｂは、ビルディングブロック＃１とビルディングブロック＃２とを有する。また、ビルディングブロック＃１は、ＣＰＵ＃４、ＣＰＵ＃５、アドレス域「＃１」が割当てられたメモリ、アドレス域「＃２」が割当てられたメモリを有する。なお、アドレス域「＃１」が割当てられたメモリには、ＣＰＵ＃４がアクセスし、アドレス域「＃２」が割当てられたメモリには、ＣＰＵ＃５がアクセスする。また、ビルディングブロック＃２は、ＣＰＵ＃８とアドレス域「＃３」が割当てられたメモリを有する。

【０１９９】
次に、図２６Ａ～２６Ｃを用いて、図２５に示すＣＰＵ＃０が有するノードマップと、ＣＰＵ＃４が有するノードマップとについて説明する。まず、図２６Ａおよび図２６Ｂを

用いて、パーティション＃ＡのＣＰＵが記憶するノードマップについて説明する。なお、図２６Ａは、パーティション＃ＡのＣＰＵが記憶するノードマップの一例を説明するための図である。また、図２６Ｂは、パーティション＃Ａを示すノードマップの一例を説明するための図である。

【０２００】

なお、以下の説明では、ノードＩＤ「０」は、ビルディングブロック＃０を示し、ノードＩＤ「１」は、ビルディングブロック＃１を示す、ノードＩＤ「２」は、ビルディングブロック＃２を示す。また、ＣＰＵＩＤ「０」は、ＣＰＵ＃０のＣＰＵＩＤであり、ＣＰＵＩＤ「４」は、ＣＰＵ＃４のＣＰＵＩＤであり、ＣＰＵＩＤ「５」は、ＣＰＵ＃５のＣＰＵＩＤであり、ＣＰＵＩＤ「８」は、ＣＰＵ＃８のＣＰＵＩＤであるものとする。

【０２０１】

例えば、図２６Ａに示す例では、ノードマップ３４は、アドレス域「＃０」が、ビルディングブロック＃０に存在し、ＣＰＵ＃０がアクセスを行う旨を示す。また、ノードマップ３４は、アドレス域「＃１」が、ビルディングブロック＃１に存在し、ＣＰＵ＃４がアクセスを行う旨を示す。また、ノードマップ３４は、アドレス域「＃２」が、ビルディングブロック＃１に存在し、ＣＰＵ＃５がアクセスを行う旨を示す。また、ノードマップ３４は、アドレス域「＃３」がビルディングブロック＃２に存在し、ＣＰＵ＃８がアクセスを行う旨を示す。

【０２０２】

また、図２６Ｂには、パーティション＃Ａを示すノードマップを示した。図２６Ｂに示すように、パーティション＃Ａを示すノードマップは、各エントリに、バリッドと、ノードＩＤとＣＰＵＩＤとを有する。例えば、図２６Ｂに示す例では、ノードマップは、パーティション＃Ａにビルディングブロック＃０のＣＰＵ＃０が含まれる旨を示す。

【０２０３】

例えば、図２５に示す例では、ＣＰＵ＃０は、図２６Ａおよび図２６Ｂに示すノードマップを有する。そして、ＣＰＵ＃０は、メモリアクセスを行う場合には、図２６Ａに示すノードマップを用いて、アクセス先のＣＰＵを識別する。一方、ＣＰＵ＃０は、同一パーティション内のＣＰＵのみに特殊パケットを送信する場合には、図２６Ｂに示すノードマップを用いて、送信先のＣＰＵを識別する。すなわち、ＣＰＵ＃０は、図２６Ｂに例示するノードマップが示すパーティション＃Ａ内のＣＰＵに対して、特殊パケットを送信する。

【０２０４】

一方、ＣＰＵ＃４は、メモリアクセスを行うために、図２６Ａに示すノードマップと、図２６Ｃに示すノードマップとを有する。ここで、図２６Ｃは、パーティション＃Ｂを示すノードマップの一例を説明するための図である。図２６Ｃに示す例では、パーティション＃Ｂを示すノードマップは、パーティション＃Ｂに、ビルディングブロック＃１のＣＰＵ＃４およびＣＰＵ＃５、ビルディングブロック＃２のＣＰＵ３８が存在することを示す。ＣＰＵ＃４は、図２６Ｃに例示するノードマップが示すパーティション＃Ｂ内のＣＰＵに対して、特殊パケットを送信する。

【０２０５】

このように、ＣＰＵ＃１およびＣＰＵ＃４は、アドレス域とＣＰＵＩＤとを対応付けたノードマップと、パーティションを示すノードマップとを記憶する。そして、ＣＰＵ＃１およびＣＰＵ＃４は、アドレス域とＣＰＵＩＤとを対応付けたノードマップを用いて、他のノードが有するメモリに対して直接メモリアクセスを行う。また、ＣＰＵ＃１は、パーティション＃Ａを示すノードマップを用いて、特殊パケットの送信を行う。また、ＣＰＵ＃４は、パーティション＃Ｂを示すノードマップを用いて、特殊パケットの送信を行う。

【０２０６】

このように、各ＣＰＵは、自身を含むパーティションごとに、異なる値を有するノードマップを有してもよい。また、各ＣＰＵは、自身を含むパーティションごとに異なる値を有するノードマップを有する場合は、特殊パケットをパーティションを超えた送信を行う

ことを防ぐことができる。

【０２０７】

なお、各ＣＰＵは、実施例１と同様、スタートアドレスとアドレスマスク、又は、スタートアドレスとレングスでアクセス対象となるアドレス域を示しても良い。すなわち、ＣＰＵ＃１とＣＰＵ＃４とは、スタートアドレスとアドレスマスク、又は、スタートアドレスとレングスとを用いて、アクセス対象となるアドレス域を示すノードマップを用いて、アクセス対象となるノードを識別する。また、ＣＰＵ＃１とＣＰＵ＃４とは、それぞれ異なるパーティションを示すノードマップを用いて、特殊パケットの送信を行う。

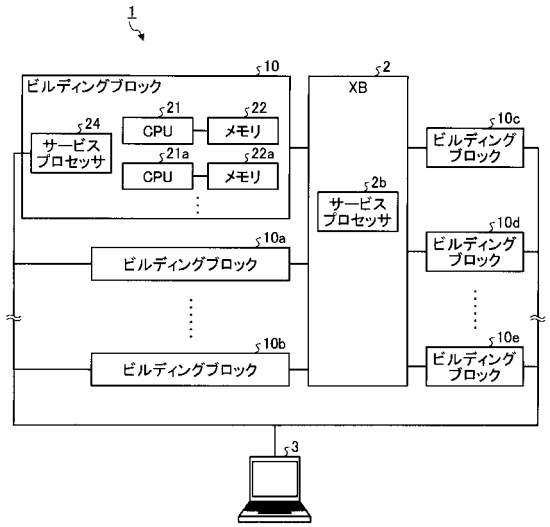
【符号の説明】

【０２０８】

１　情報処理システム
２　ＸＢ
１０～１０ｅ　ビルディングブロック
２０　ノード
２１～２１ｃ　ＣＰＵ
２２～２２ｃ　メモリ
２３、２６　通信部
２４　サービスプロセッサ
２５　制御部
２７、２７ａ　ＸＢ接続部
２８　ＰＣＩｅ接続部
３０　演算処理部
３１　演算部
３２　Ｌ１キャッシュ
３３　Ｌ２キャッシュ
３４　ノードマップ
３５　アドレス変換部
３６　キャッシュディレクトリ管理部
３７　パケット制御部
３８　リクエスト生成部
３９　リクエスト受信部
４０　ルータ
４１　メモリアクセス部
４２　ＰＣＩｅ制御部
４３　リクエスト生成部
４４　ＰＣＩｅバス制御部

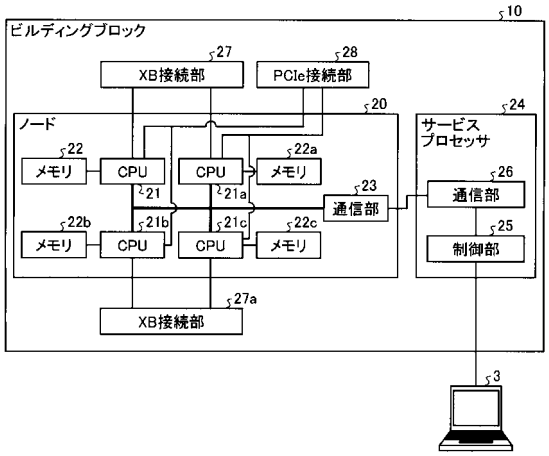
【図１】

実施例1に係る情報処理システムの一例を説明するための図



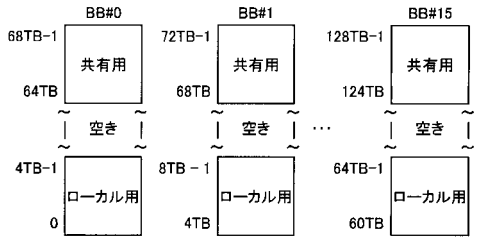
【図２】

実施例1に係るビルディングブロックの機能構成を説明するための図



【図３】

実施例1に係るビルディングブロックのメモリに振り分けられる物理アドレスの範囲を説明するための図

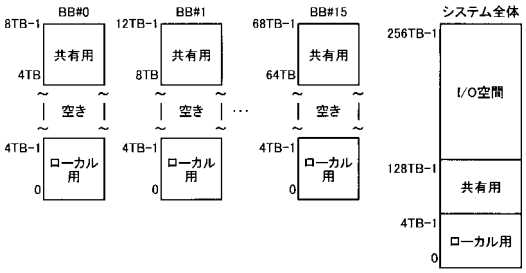


【図４】

実施例1に係る情報処理システムが各メモリに振り分ける物理アドレスを説明するための図

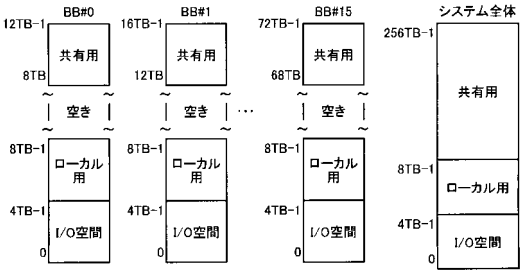
【図５】

物理アドレスの振り分けのバリエーションを説明するための第１の図



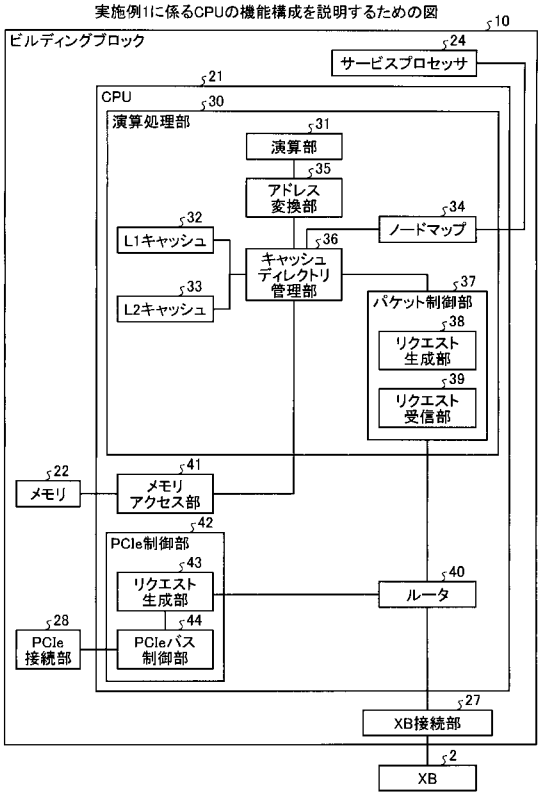
【図６】

物理アドレスの振り分けのバリエーションを説明するための第２の図



【図７】

実施例1に係るCPUの機能構成を説明するための図



【図８】

実施例1に係るノードマップが記憶する情報の一例を説明するための図

アドレス	バリッド	ノードID	CPUID
#0	1	0	0
#1	1	0	1
#2	0		

⋮

【図９】

ノードマップが記憶する情報のバリエーションの一例を説明するための第1の図

バリッド	スタートアドレス	アドレスマスク	ノードID	CPUID
1	0x00000	0x3fff	0	0
1	0x10000	0x3fff	1	4
1	0x14000	0x3fff	1	5
1	0x20000	0x1fff	2	8

⋮

【図１０】

ノードマップが記憶する情報のバリエーションの一例を説明するための第2の図

バリッド	スタートアドレス	レングス	ノードID	CPUID
1	0x00000	0x3fff	0	0
1	0x10000	0x3fff	1	4
1	0x14000	0x3fff	1	5
1	0x20000	0x2ef	2	8

⋮

【図１１Ａ】

キャッシュタグの一例を説明するための図

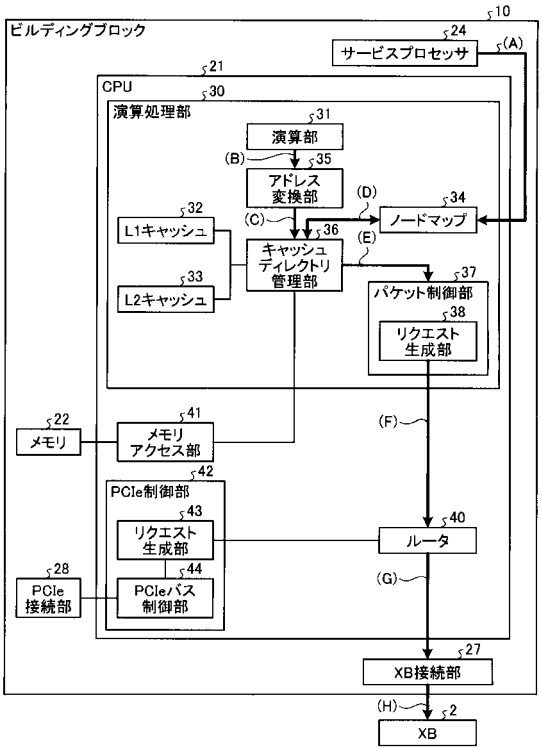
【図１１Ｂ】

実施例1に係るCPUが送信するパケットを説明するための図

CPUID	PA	データ

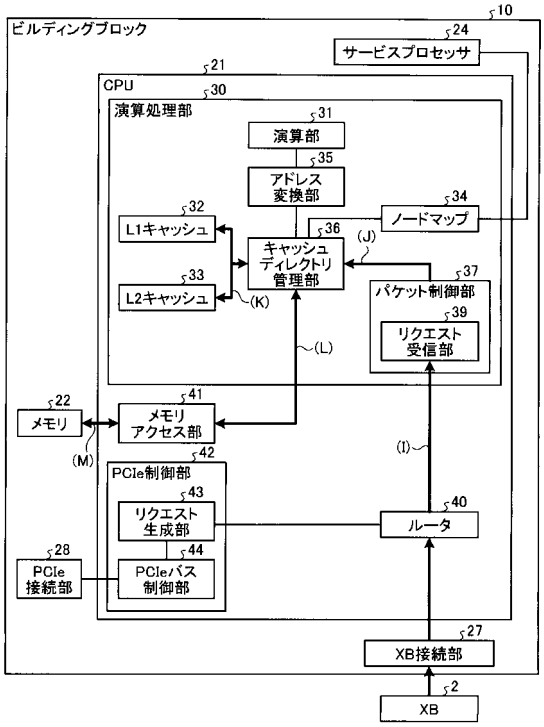
【図１２】

実施例1に係るCPUがリクエストを送信する処理の一例を説明するための図



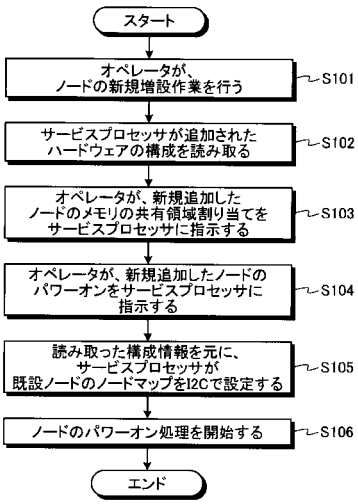
【図１３】

実施例1に係るCPUがパケットを受信した際に実行する処理の一例を説明するための図



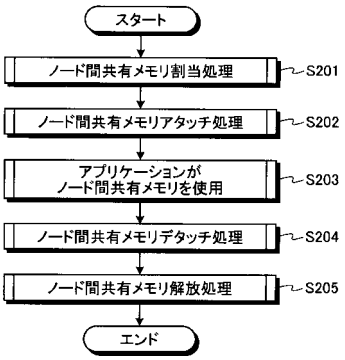
【図１４】

ノードマップを設定する処理の流れを説明するためのフローチャート



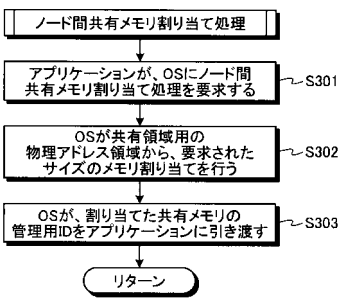
【図１５】

共有領域を制御する処理の流れを説明するためのフローチャート

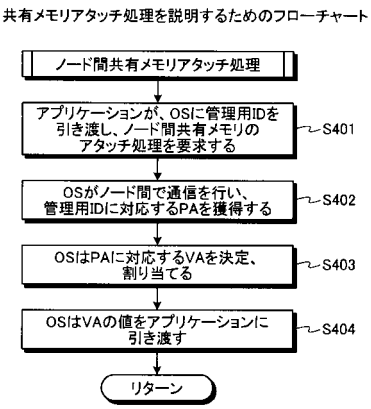


【図１６】

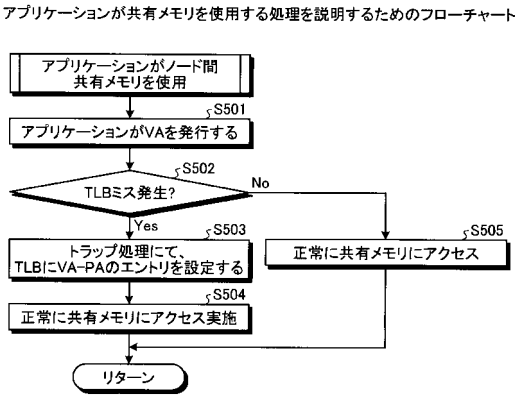
共有メモリの割当処理を説明するためのフローチャート



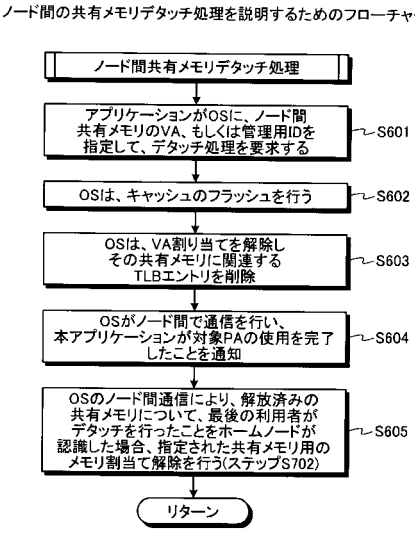
【図１７】



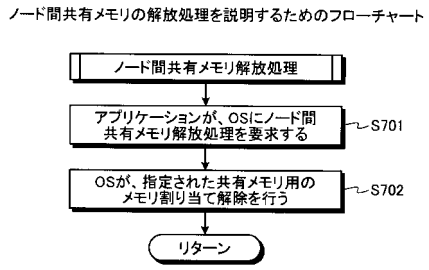
【図１８】



【図１９】

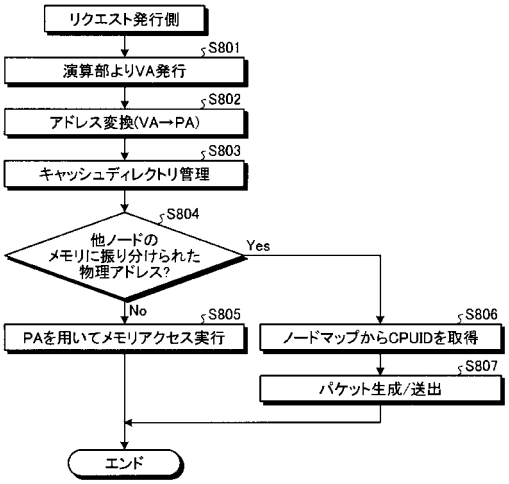


【図２０】



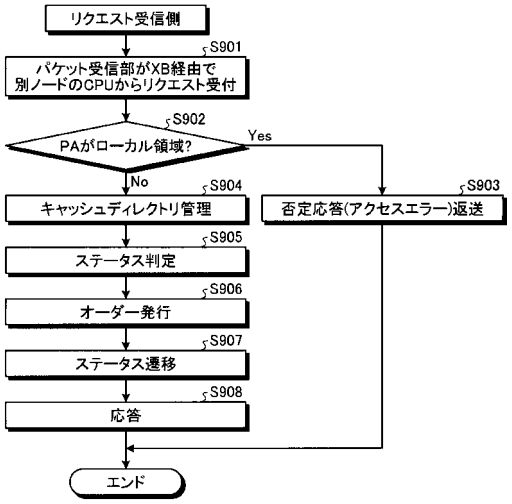
【図２１】

リクエストを発行する処理の流れを説明するためのフローチャート



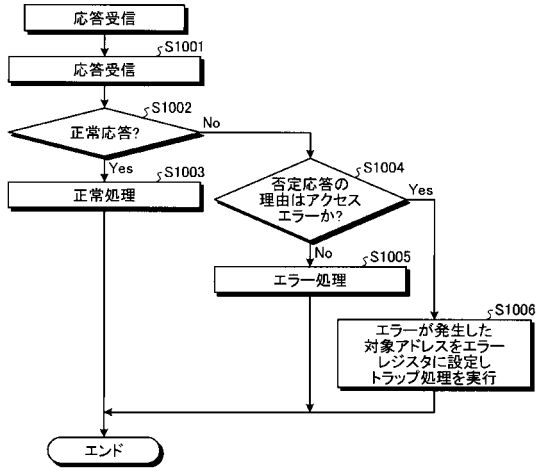
【図２２】

リクエストを受信した際に実行する処理の流れを説明するためのフローチャート



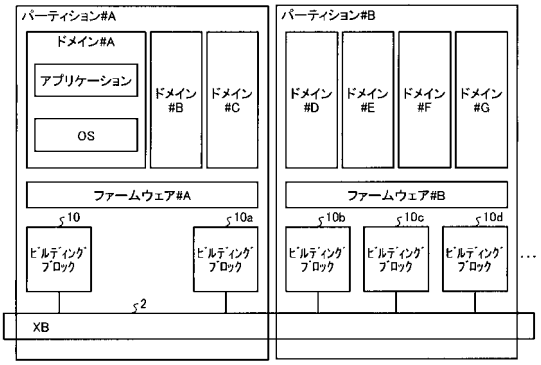
【図２３】

CPUが応答を受信した際に実行する処理の流れを説明するためのフローチャート



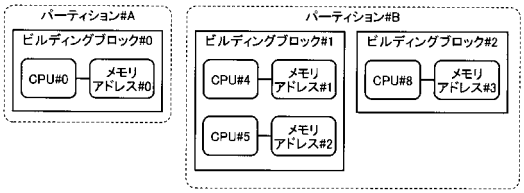
【図２４】

実施例2に係る情報処理システムを説明するための図



【図２５】

パーティションの一例を説明するための図



【図２６Ａ】

パーティション＃ＡのCPUが記憶するノードマップの一例を説明するための図

アドレス	バリッド	ノードID	CPUID
#0	1	0	0
#1	1	1	4
#2	1	1	5
#3	1	2	8

⋮

【図２６Ｂ】

パーティション＃Ａを示すノードマップの一例を説明するための図

エントリ	バリッド	ノードID	CPUID
#0	1	0	0
#1	0		
#2	0		
#3	0		

⋮

【図２６Ｃ】

パーティション＃Ｂを示すノードマップの一例を説明するための図

エントリ	バリッド	ノードID	CPUID
#0	1	1	4
#1	1	1	5
#2	1	2	8
#3	0		

⋮

フロントページの続き

審査官　野田　佳邦

(56)参考文献　特開平１０－２４０７０７（ＪＰ，Ａ）
特開２０００－０６７００９（ＪＰ，Ａ）
特開平０２－２４４２５３（ＪＰ，Ａ）
特開平０８－０３０５６８（ＪＰ，Ａ）
特開２００７－１９９９９９（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１２／００－１２／０６
Ｇ０６Ｆ　１２／０８－１２／１２