

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第5151559号
(P5151559)

(45) 発行日 平成25年2月27日(2013.2.27)

(24) 登録日 平成24年12月14日(2012.12.14)

(51) Int.Cl. F I
G 0 6 F 9/52 (2006.01) G O 6 F 9/46 4 7 2 Z
G 0 6 F 15/167 (2006.01) G O 6 F 15/167 6 1 0 B

請求項の数 8 (全 23 頁)

(21) 出願番号	特願2008-50953 (P2008-50953)	(73) 特許権者	000005223
(22) 出願日	平成20年2月29日(2008.2.29)		富士通株式会社
(65) 公開番号	特開2009-211167 (P2009-211167A)		神奈川県川崎市中原区上小田中4丁目1番1号
(43) 公開日	平成21年9月17日(2009.9.17)	(74) 代理人	100070150
審査請求日	平成22年10月18日(2010.10.18)		弁理士 伊東 忠彦
		(72) 発明者	上方 輝彦
			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
		審査官	井上 宏一

最終頁に続く

(54) 【発明の名称】 プログラム実行システム

(57) 【特許請求の範囲】

【請求項1】

複数のプロセッサを含み、前記複数のプロセッサ上で動作可能な中間コードを実行するプログラム実行環境を備えたプログラム実行システムであって、

前記複数のプロセッサ夫々に専用の複数のプロセッサ専用メモリと、

前記複数のプロセッサ間で共有される前記中間コードによって操作される共有オブジェクトを格納するプロセッサ間共有メモリとを備え、

前記複数のプロセッサの各々は、

前記プロセッサ専用の前記中間コードを実行する前記プログラム実行環境と、

前記共有オブジェクトを指定する名前と前記プロセッサ間共有メモリ内の該共有オブジェクトの実体との対応に基づいて、前記複数のプロセッサ専用メモリの各々と該プロセッサ間共有メモリ間にて該共有オブジェクトを読み出すルーチン及び書き込むルーチンを呼び出すためのAPIによってプロセッサ間中間コード実行環境を提供するプロセッサ間共有メモリ用ライブラリとを有するようにしたプログラム実行システム。

10

【請求項2】

前記プロセッサ間共有メモリは、前記共有オブジェクトの名前と該共有オブジェクトの実体とを対応させた対応テーブルを備え、

前記複数のプロセッサの各々は、前記共有オブジェクトの名前と該共有オブジェクトの実体との対応を前記対応テーブルに書き込むことにより該共有オブジェクトを登録し、また該登録された共有オブジェクトを該対応テーブルから削除することにより該共有オブジ

20

ェクトを削除する登録削除機能を実行するようにした請求項 1 記載のプログラム実行システム。

【請求項 3】

前記複数のプロセッサの各々は、

前記プロセッサ間共有メモリのメモリ領域の割り当て及び開放を管理する共有メモリ管理機能と、

前記複数のプロセッサ間の前記プロセッサ間共有メモリへのアクセスに対してセマフォ又はロック機能を用いて排他制御を行うプロセッサ間排他制御機能と、

前記プロセッサ間共有メモリのヒープ領域に前記共有オブジェクトを格納するための共有メモリヒープ機能とを実行するようにした請求項 1 又は 2 記載のプログラム実行システム。

10

【請求項 4】

前記プロセッサ間共有メモリのメモリ領域と前記複数のプロセッサ専用メモリの各々のメモリ領域との間のダイレクトメモリアccessによるデータ転送を制御するDMAコントローラを更に備え、

前記複数のプロセッサの各々において、前記DMAコントローラを用いて前記高速データ転送を行うルーチンと該ルーチンを前記プログラム実行環境から呼び出すためのインターフェイスを備えるようにした請求項 1 乃至 3 のいずれか一項記載のプログラム実行システム。

【請求項 5】

前記プロセッサ間排他制御機能を有するハードウェアを備え、

前記複数のプロセッサの各々において、前記ハードウェアのプロセッサ間排他制御機能を用いて排他制御を行うルーチンと該ルーチンを前記プログラム実行環境から呼び出すためのインターフェイスを備えるようにした請求項 1 乃至 3 のいずれか一項記載のプログラム実行システム。

20

【請求項 6】

複数の異種プロセッサ毎に専用メモリを備えたヘテロジニアスマルチプロセッサであって、

前記複数の異種プロセッサ間で共有される該各異種プロセッサ上で動作可能な中間コードによって操作される共有オブジェクトを格納するプロセッサ間共有メモリを備え、

前記各専用メモリは、

前記共有オブジェクトを指定する名前と前記プロセッサ間共有メモリ内の該共有オブジェクトの実体との対応に基づいて、前記各プロセッサ専用メモリと該プロセッサ間共有メモリ間にて該共有オブジェクトを読み出すルーチン及び書き込むルーチンを呼び出すためのAPIによってプロセッサ間中間コード実行環境を提供するプロセッサ間共有メモリ用ライブラリを格納し、

30

前記各異種プロセッサは、

前記異種プロセッサ専用の前記中間コードを実行する前記プログラム実行環境と、

自プロセッサの前記専用メモリに格納されたプロセッサ間共有メモリ用ライブラリを用いて前記読み出すルーチン及び前記書き込むルーチンを呼び出すことによって、前記共有オブジェクトへの操作を行う前記プロセッサ間中間コード実行環境を有するようにしたヘテロジニアスマルチプロセッサ。

40

【請求項 7】

前記各専用メモリは、

前記共有オブジェクトの名前と該共有オブジェクトの実体との対応を前記プロセッサ間共有メモリが保持する対応テーブルに書き込むことにより該共有オブジェクトを登録し、また該登録された共有オブジェクトを該対応テーブルから削除することにより該共有オブジェクトを削除する登録削除機能を前記異種プロセッサに実行させるプログラムを格納するようにした請求項 6 記載のヘテロジニアスマルチプロセッサ。

【請求項 8】

50

前記各専用メモリは、

前記プロセッサ間共有メモリのメモリ領域の割り当て及び開放を管理する共有メモリ管理機能と、

前記複数のプロセッサ間の前記プロセッサ間共有メモリへのアクセスに対してセマフォ又はロック機能を用いて排他制御を行うプロセッサ間排他制御機能と、

前記プロセッサ間共有メモリのヒープ領域に前記共有オブジェクトを格納するための共有メモリヒープ機能とを前記異種プロセッサに実行させるプログラムを格納するようにした請求項6又は7記載のヘテロジニアスマルチプロセッサ。

【発明の詳細な説明】

【技術分野】

10

【0001】

本発明は、組み込みヘテロマルチプロセッサシステム上でのプロセッサ間で連携動作する中間コードを用いるアプリケーションプログラムを高速に実行可能とするプログラム実行システムに関する。

【背景技術】

【0002】

近年、情報処理装置を組み込んだ電子機器の多様化及びそれら電子機器間を通信接続するネットワーク化に伴い、多数の異種プロセッサが開発されると共に、ネットワーク化もあらゆる電子機器間で実現されるようになってきた。多様化した電子機器の開発において、プロセッサに依存したプログラム開発を行うのでは開発者の負担が大きい。そのため、ハードウェアやOS (Operation System) に依存しない仮想マシンを構築し、ハードウェアが解釈するネイティブコードより抽象度の高い中間コードで表現されるプログラムを仮想マシン上で実行することで、ネットワークを介してプログラムの授受を可能とし、かつプロセッサ毎のプログラム開発の負担を低減することができるようになった。

20

【0003】

更に、仮想マシンを備えないプロセッサへネットワークを介してプログラムを転送するために、仮想マシンを備えたプロセッサが仮想マシンを備えないプロセッサ向けのネイティブコードに変換することが提案されている(例えば、特許文献1及び特許文献2参照。)。また、プロセッサ毎のネイティブコードへ変換する負荷を低減するために、各プロセッサがネイティブトランスレータを備えることが提案されている(例えば、特許文献3参照。)。)

30

【特許文献1】特開2000-172509号公報

【特許文献2】特開2005-108126号公報

【特許文献3】特表2003-508844号公報

【発明の開示】

【発明が解決しようとする課題】

【0004】

従来の疎結合のヘテロジニアスマルチプロセッサによるシステム1は、例えば図1に示すように、異なるアーキテクチャを持つプロセッサA11及びB15を備え、プロセッサA11にはプロセッサA専用LAN-IF12とプロセッサA専用メモリ13とが各々バスBで接続され、同様にプロセッサB15にはプロセッサB専用LAN-IF16とプロセッサB専用メモリ17とが各々バスで接続される。プロセッサA及びB間のデータ転送は、例えば、LAN19を介して行われる。

40

【0005】

また、このようなシステム1は、例えば図2に示すような階層構造を成す。プロセッサA専用メモリ13には、アプリケーションプログラム、アプリケーションデータ、プロセッサA用実行環境、RMI (Remote Method Invocation)、プロセッサA用JIT (Just In Time) コンパイラ、LANドライバなどがプロセッサA用メモリ領域13aに格納されている。プロセッサB専用メモリ17においても同様に、アプリケーションプログラム、アプリケーションデータ、プロセッサB用実行環境、RMI、プロセッサB用JIT

50

Tコンパイラ、LANDライバなどがプロセッサB用メモリ領域17aに格納されている。アプリケーションプログラムを上位層としLANDライバを下位層とした階層構造に従った機能構成を成す。各プロセッサA11及びプロセッサB15は、それぞれのプロセッサA専用メモリ13及びプロセッサB専用メモリ17からコード又はデータを読み出して所定機能を実行する。

【0006】

アプリケーションプログラムがJava(登録商標)言語によるプログラムである場合、図2に示すプロセッサA用実行環境及びプロセッサB用実行環境は、図3に示すようにJava(登録商標)VM(Virtual Machine)として機能する。Java(登録商標)VMは、クラスローダー、クラス・キャッシュ機能、ヒープメモリ管理機能、ガーベジ・コレクション(GC)機能を備え、Java(登録商標)プログラムの実行時には、PCレジスタ、メソッド領域、ヒープ領域、Java(登録商標)VMスタック、実行時定数プール、Nativeメソッドスタック等の領域がメモリ内に展開される。メソッド領域にはクラスローダーによってクラスがロードされ、ヒープ領域にはヒープメモリ管理機能によってクラスがインスタンス化したオブジェクトが格納される。Java(登録商標)VMスタックには、関数が1つ呼ばれるたびにフレームがスタックされ、フレーム内にはローカル変数及びオペランドスタックの領域が構成される。このような実行環境にて実行されるプログラムを中間コードで表現されたプログラムという。

10

【0007】

同種のプロセッサ間で、かつ対称型マルチプロセッシング(Symmetric Multi-Processing(SMP))ならば、POSIX(Portable Operating System Interface for UNIX(登録商標))スレッドによってスレッド化してメモリを共有できるが、図1に示すようなシステム1では、異種のプロセッサA11及びB15間ではスレッド化してメモリを共有することができない。

20

【0008】

Java(登録商標)言語、C#言語などのプロセッサ非依存の中間コードによる処理では、複数のJava(登録商標)VM間のメモリ空間は共有されない仕組みであるため、プロセッサAがプロセッサBのJava(登録商標)VMが管理するメモリにはアクセスできない。

【0009】

ヘテロジニアスマルチプロセッサによるシステム1での負荷分散のために各プロセッサ用の実行形式モジュールを持たせると、アプリケーションプログラムのROM容量が多く必要となる。また、異種プロセッサA11及びB15毎にプログラムを作成し管理するのは、開発者にとって負担となる。

30

【0010】

ヘテロジニアスマルチプロセッサ間のデータ転送にLAN通信層を使うとデータコピーが多くなり内部バス間転送より時間を要する。また、Java(登録商標)やC#言語などで書かれたアプリケーションプログラムを用いた場合、ヘテロジニアスマルチプロセッサ間では、RMIなどのようにリモートオブジェクトとして登録して、LAN等を介して別のJava(登録商標)のオブジェクトのメソッドを呼ぶ仕組みであるため、ヘテロジニアスマルチプロセッサ間のLAN等を用いた(TCP/IPによる)データ転送には多くのデータ転送回数が必要になる。そのため、ヘテロジニアスマルチプロセッサを1チップ化したとしても、それ自身の性能を生かすことができないという問題があった。

40

【0011】

更に、非対称型マルチプロセッシング(Asymmetric Multi-Processing(AMP))での異種プロセッサ上のOSからは、異種PE間での共有ファイルシステムをNFSなどのように片方のOSに依頼するなどによって行われるため、高速にアクセスできない。

【0012】

よって、本発明の目的は、ヘテロジニアスマルチプロセッサでアプリケーションプログラムを共通化してプログラム開発の負荷軽減を可能とし、ヘテロジニアスマルチプロセ

50

サ上で負荷分散できるようにしつつ、ヘテロマルチプロセッサ間のデータ転送を削減することで、ヘテロジニアスマルチプロセッサ間で協調動作するプログラムを高速化することを提供することである。

【課題を解決するための手段】

【0013】

上記課題を解決するため、本発明は、複数のプロセッサを含み、前記複数のプロセッサ上で動作可能な中間コードを実行するプログラム実行環境を備えたプログラム実行システムであって、前記複数のプロセッサ夫々に専用の複数のプロセッサ専用メモリと、のプロセッサ間で共有される前記中間コードによって操作される共有オブジェクトを格納するプロセッサ間共有メモリとを備え、前記複数のプロセッサの各々は、前記中間コードを実行するプロセッサ専用の実行環境と、前記共有オブジェクトを指定する名前と前記プロセッサ間共有メモリ内の該共有オブジェクトの実体とを対応に基づいて、前記複数のプロセッサ専用メモリの各々と該プロセッサ間共有メモリ間にて該共有オブジェクトを読み出すルーチン及び書き込むルーチンと呼び出すためのAPIによってプロセッサ間中間コード実行環境を提供するプロセッサ間共有メモリ用ライブラリとを有するように構成される。

10

【0014】

このようなプログラム実行システムにおいて、前記プロセッサ間共有メモリは、前記共有オブジェクトの名前と該共有オブジェクトの実体とを対応させた対応テーブルを備え、前記各プロセッサは、前記共有オブジェクトの名前と該共有オブジェクトの実体との対応を前記対応テーブルに書き込むことにより該共有オブジェクトを登録し、また該登録された共有オブジェクトを該対応テーブルから削除することにより該共有オブジェクトを削除する登録削除機能を実行するように構成してもよい。

20

【発明の効果】

【0015】

本願発明は、ヘテロジニアスマルチプロセッサでアプリケーションプログラムを共通化してプログラム開発の負荷軽減を可能とし、ヘテロジニアスマルチプロセッサ上で負荷分散できるようにしつつ、ヘテロマルチプロセッサ間のデータ転送を削減することで、ヘテロジニアスマルチプロセッサ間で協調動作するプログラムを高速化することができる。

【発明を実施するための最良の形態】

【0016】

以下、本発明の実施の形態を図面に基づいて説明する。

30

【0017】

図4は、本発明の第一実施例に係る密結合のヘテロジニアスマルチプロセッサによるシステム例を示す図である。図4において、システム100は、密結合されたヘテロジニアスマルチプロセッサのシステムであり、プロセッサA21と、プロセッサA専用LAN-IF22と、プロセッサA専用メモリ23と、プロセッサB25と、プロセッサB専用LAN-IF26と、プロセッサB専用メモリ27と、プロセッサ間共有メモリ31とを有し、それらはバス30で互いに接続される。プロセッサA21とプロセッサB25とは異種プロセッサであり、ハードウェアに依存しない中間コードで書かれたアプリケーションプログラムを実行できる実行環境を提供する。例えば、中間コードはJava（登録商標）言語であり、実行環境はJava（登録商標）VMである。

40

【0018】

プロセッサA専用LAN-IF22とプロセッサA専用メモリ23とは、プロセッサA21によってのみ使用されるデバイスである。同様に、プロセッサB専用LAN-IF26と、プロセッサB専用メモリ27とは、プロセッサB25によってのみ使用されるデバイスである。

【0019】

プロセッサ間共有メモリ31は、プロセッサA21とプロセッサB25との間でアクセス可能な共有メモリである。プロセッサA21は、プロセッサ間共有メモリ31をアクセスすることによって、プロセッサA21とプロセッサB25とで共有に使用されるオブジ

50

エクトをプロセッサ間共有メモリ31からプロセッサA専用メモリ23へと読み込んだり、プロセッサA専用メモリ23からプロセッサ間共有メモリ31へと書き込んだりする操作を高速に行うことができる。同様に、プロセッサB25は、プロセッサ間共有メモリ31をアクセスすることによって、プロセッサA21とプロセッサB25とで共有に使用されるオブジェクトをプロセッサ間共有メモリ31からプロセッサB専用メモリ27へと読み込んだり、プロセッサB専用メモリ27からプロセッサ間共有メモリ31へと書き込んだりする操作を高速に行うことができる。

【0020】

プロセッサ間共有メモリ31は、図5に示すようにメモリ領域を構成する。図5は、プロセッサ間共有メモリの領域構成例を示す図である。

10

【0021】

図5において、プロセッサ間共有メモリ31は、主に、プロセッサ間共有メモリ領域32とを備える。更に、プロセッサ間共有メモリ領域32は、プロセッサ間ヒープ領域33と、プロセッサ間排他制御用領域34と、プロセッサ間共有オブジェクト名テーブル35とを備える。

【0022】

プロセッサ間ヒープ領域33は、プロセッサ間共有オブジェクト名テーブル35に登録されているオブジェクトが格納され、プロセッサA21又はプロセッサB25によってオブジェクトが書き込まれたり、読み出されたりする領域である。プロセッサ間排他制御用メモリ領域34では、プロセッサ間ヒープ領域33へのアクセスに係る排他制御を行うためのプロセッサ間セマフォ34a又はプロセッサ間ロック34bが管理される。

20

【0023】

プロセッサ間共有メモリ31に、プロセッサ間排他制御用領域34を備えて、各プロセッサA21及びプロセッサB25からアトミックアクセスさせることで、LAN-IFやLANDライバを経由しないで、排他制御を行うことでプロセス間共有メモリ領域32を使用するときの排他制御を高速化することができる。

【0024】

図6は、ヘテロジニアスマルチプロセッサによるシステムの構成例を示す図である。図6に示すシステム100では、プロセッサA専用メモリ23には、アプリケーションプログラム232と、アプリケーションデータ233と、プロセッサA用実行環境234と、プロセッサA用JIT(Just In Time)コンパイラ235と、プロセッサ間共有メモリ用ライブラリ236とがプロセッサA用メモリ領域231に格納され、プロセッサB専用メモリ27には、アプリケーションプログラム272と、アプリケーションデータ273と、プロセッサB用実行環境274と、プロセッサB用JITコンパイラ275と、プロセッサ間共有メモリ用ライブラリ276とがプロセッサB用メモリ領域271に格納されている。アプリプログラムを上位層とし実行環境の一部及びJITコンパイラを下位層とした階層構造に従った機能構成を成す。

30

【0025】

各プロセッサA21及びプロセッサB25は、それぞれのプロセッサA専用メモリ23及びプロセッサB専用メモリ27からコード又はデータを読み出して所定機能を実行する。

40

【0026】

プロセッサ間共有メモリ用ライブラリ236とプロセッサ間共有メモリライブラリ276とは、図7に示すようにプロセッサ間共有メモリ管理機能36及びプロセッサ間排他制御機能37とを備え、それらを用いて共有メモリ管理及び共有メモリ排他制御とを提供する種々ルーチンを備えると共に、それらルーチンを呼び出すためのAPI(Application Program Interface)を提供する。

【0027】

プロセッサ間共有メモリ管理機能36は、自身のプロセッサA21又はプロセッサB25からのアクセスに応じて、プロセッサ間共有メモリ31のプロセッサ間ヒープ領域33

50

への割り当て及び開放等のメモリ管理を行う。プロセッサ間排他制御機能37は、自身のプロセッサA21又はプロセッサB25からのプロセッサ間共有メモリ31のプロセッサ間ヒープ領域33へのアクセスに対して排他制御を行うためには、プロセッサ間ヒープ領域33の全体に対して1以上のプロセッサ間セマフォ34a又はプロセッサ間ロック34bをプロセッサ間排他制御用領域34に設ける。

【0028】

プロセッサ間共有メモリ用ライブラリ236はプロセッサA用実行環境234にAPIを提供し、プロセッサA用実行環境234はプロセッサ間共有メモリ用ライブラリ236から提供される共有メモリ管理及び共有メモリ排他制御を実行する種々ルーチンを用いて、プロセッサA21によるプロセッサ間共有メモリ31へのアクセスを高速に行うことを可能とする。同様に、プロセッサ間共有メモリライブラリ276はプロセッサB用実行環境274にAPIを提供し、プロセッサB用実行環境274はプロセッサ間共有メモリライブラリ276から提供される共有メモリ管理及び共有メモリ排他制御を実行する種々ルーチンを用いて、プロセッサB21によるプロセッサ間共有メモリ31へのアクセスを高速に行うことを可能とする。

10

【0029】

プロセッサ間共有メモリ領域32では、プロセッサ間共有メモリ用ライブラリ236とプロセッサ間共有メモリライブラリ276とが夫々のプロセッサA用実行環境234とプロセッサB用実行環境274とに提供する種々ルーチンによって、プロセッサ間ヒープ領域33と、プロセッサ間排他制御用領域34と、プロセッサ間共有オブジェクト名テーブル35とがプロセッサA21とプロセッサB25とに共有可能となる。

20

【0030】

図8は、各プロセッサ用実行環境とプロセッサ間共有メモリ用ライブラリとによる機能構成を示す図である。図8において、例えば、各プロセッサ用実行環境234及び274がJava(登録商標)VM54を提供する場合、Java(登録商標)VM54は、クラスファイルをロードしてメソッド領域63に格納するクラスローダー54aと、クラスをキャッシュして高速に実行するクラス・キャッシュ機能54bと、ヒープメモリ管理機能54cと、ヒープ領域64の割り当て及び開放を管理するガーベジ・コレクション(GC)機能54dとを備える。Java(登録商標)VM54は、実行時データ領域61に示される。

30

【0031】

実行時データ領域61には、Java(登録商標)VM54によって、実行中の命令アドレスを示すプログラムカウンタとして使用されるPCレジスタ62と、クラスファイルが格納されるメソッド領域63と、クラスがインスタンス化されたオブジェクトを格納するヒープ領域64と、一つのスレッドを実行するための各メソッドに対応するフレームを格納するJava(登録商標)VMスタック65と、各クラスで使用される定数を格納する実行時定数プール66と、ネイティブコードのメソッドを実行するためのNativeメソッドスタック67とが展開される。

【0032】

このように、システム100の構成において、プロセッサA21側でプロセッサA21専用のJava(登録商標)プログラムを実行する場合には、プロセッサA用実行環境234は、プロセッサA専用メモリ23を用いてJava(登録商標)VM54を提供する。プロセッサB25側でプロセッサB25専用のJava(登録商標)プログラムを実行する場合においても同様である。

40

【0033】

プロセッサA21側のアプリケーションプログラム232及びプロセッサB25側のアプリケーションプログラム272がプロセッサ間で共有される共有クラスからインスタンス化したプロセッサ間共有オブジェクトを実行するために、各プロセッサ用実行環境234及び274は、夫々のプロセッサ間共有メモリ用ライブラリ236及び237を用いて、更にプロセッサ間中間コード実行環境70が提供される。プロセッサ間中間コード実行

50

環境 7 0 において、共有メモリ 3 1 と専用メモリ 2 3 1 又は 2 7 1 (以下、単に専用メモリと言う)間のプロセッサ間データ転送機能 7 9 が行われる。

【 0 0 3 4 】

プロセッサ間中間コード実行環境 7 0 は、プロセッサ間共有メモリ・クラス 7 1 と、プロセッサ間共有中間コードのヒープメモリ 7 2 と、プロセッサ間共有オブジェクト操作機能 7 3 と、プロセッサ間共有クラスローダー 7 4 と、プロセッサ間共有クラス・キャッシュ機能 7 5 とを有する。

【 0 0 3 5 】

プロセッサ間共有メモリ・クラス 7 1 は、インスタンス化されるとプロセッサ間共有オブジェクトとなるプロセッサ間共有クラスである。プロセッサ間共有中間コードのヒープメモリ 7 2 は、プロセッサ間共有メモリ 3 1 のプロセッサ間共有メモリ領域 3 2 内にあるプロセッサ間ヒープ領域 3 3 をプロセッサ間共有メモリ管理機能 3 6 を用いて制御するための機能を備え、J a v a (登録商標) V M 5 4 のヒープメモリ管理機能 5 4 c 及びガーベジ・コレクション (G C) 機能 5 4 d とは別にプロセッサ間共有メモリ 3 1 用として、ヒープメモリ操作・管理機能 7 2 a とプロセッサ間共有ガーベジ・コレクション (G C) 機能 7 2 b とを有する。

【 0 0 3 6 】

プロセッサ間共有オブジェクト操作機能 7 3 は、プロセッサ間共有メモリ 3 1 が備えるプロセッサ間共有メモリ管理機能 3 6 及びプロセッサ間排他制御機能 3 7 と、プロセッサ間データ転送機能 7 9 とを呼び出すルーチンを使用して共有オブジェクトの操作を行う。これらルーチンは、プロセッサ間共有ライブラリ 2 3 6 及び 2 7 6 によって提供される。プロセッサ間共有オブジェクト操作機能 7 3 は、専用メモリ又はプロセッサ間共有メモリ 3 1 から読み出すためのリード 7 3 a と、専用メモリ又はプロセッサ間共有メモリ 3 1 へと書き込むためのライト 7 3 b と、プロセッサ間共有オブジェクトを登録するための登録 7 3 c と、登録したプロセッサ間共有オブジェクトを削除するための削除 7 3 d と、図 5 に示すプロセッサ間共有オブジェクト名テーブル 3 5 を用いてプロセッサ間共有オブジェクトとその名前とを対応させるオブジェクト対応機能 7 3 f とを有する。

【 0 0 3 7 】

プロセッサ間共有クラスローダー 7 4 は、プロセッサ間共有クラスをロードするための機能であり、J a v a (登録商標) V M 5 4 のクラスローダー 5 4 a とは別に備えられる。プロセッサ間共有クラス・キャッシュ機能 7 5 とは、プロセッサ間共有クラスをキャッシュするための機能であり、J a v a (登録商標) V M 5 4 のクラス・キャッシュ機能 5 4 b とは別に備えられる。

【 0 0 3 8 】

プロセッサ間共有メモリ 3 1 には、図 5 及び図 6 で説明したように、プロセッサ間共有メモリ管理機能 3 6 と、プロセッサ間排他制御機能 3 7 とを有する。プロセッサ間共有メモリ管理機能 3 6 にはプロセッサ間共有オブジェクト 3 3 a に対するプロセッサ間ヒープ領域 3 3 のメモリ割り当て / 開放部 3 6 a が備えられる。

【 0 0 3 9 】

プロセッサ間データ転送機能 7 9 は、プロセッサ間共有メモリ 3 1 と専用メモリ間のデータ転送を L A N - I F や L A N ドライバを用いることなく実現する機能であり、引数 / 戻り値のデータ転送部 7 9 a 及びデータ転送を高速に行うための高速データ転送部 7 9 b とを備える。

【 0 0 4 0 】

上述したようなプロセッサ間中間コード実行環境 7 0 及びプロセッサ間データ転送機能 7 9 が実現する種々の機能は、各プロセッサ間共有メモリライブラリ 2 3 6 及び 2 7 6 においてルーチン化され提供される。後述されるように D M A (Direct Memory Access) コントローラ (D M A C) を備えるシステム構成の場合、D M A C の D M A 転送機能を用いたより高速なデータ転送を行うルーチンが提供される。

【 0 0 4 1 】

10

20

30

40

50

図9は、本発明の第二実施例に係る密結合のヘテロジニアスマルチプロセッサによるシステム構成例を示す図である。図9に示すシステム1002は、図4に示すシステム1000における構成に加えてDMAC41を備える。システム1002は、更に排他制御機構42を備えるようにしてもよい。

【0042】

DMAC41を使用して高速にデータ転送を行う場合、高速データ転送部79b(図8)によってプロセッサ間共有メモリ31からプロセッサA専用メモリ23又はプロセッサB専用メモリ27へと、又は、プロセッサA専用メモリ23又はプロセッサB専用メモリ27からプロセッサ間共有メモリ31へとプロセッサ間共有オブジェクトに係る情報が転送される。プロセッサ間共有オブジェクトに係る情報は、アプリケーションプログラム232又は272のプロセッサ間共有オブジェクトへの入力データ、プロセッサ間共有オブジェクトを実行した結果データ等である。高速データ転送部79bは、プロセッサ間共有メモリ用ライブラリ236及び276が提供するプロセッサ間データ転送機能79にDMAC41にDMAデータ転送を実行させるためのルーチンとして組み込まれる。

10

【0043】

また、図5に示すプロセッサ間共有メモリ31のプロセッサ間排他制御機能37とプロセッサ間排他制御用領域34とを、別のハードウェアとして構成される排他制御機構42に備えるようにしてもよい。この場合、プロセッサ間共有メモリ用ライブラリ236及び276が提供するプロセッサ間共有オブジェクト操作機能73(図8)に、排他制御機構42に備えられたプロセッサ間排他制御機能を実行するルーチンが組み込まれる。従って、プロセッサ間共有メモリ31には、プロセッサ間共有メモリ管理機能36とプロセッサ間共有メモリ領域32とを備えるのみでよい。

20

【0044】

プロセッサ間排他制御を専用に行う排他制御機構42を備えることによって、プロセッサ間共有メモリ31への操作に係る排他制御を、ハードウェアのプロセッサ間のセマフォ又はロック機能を用いて高速に行うことができる。

【0045】

次に、図8に示すプロセッサ間共有オブジェクト操作機能73での処理について説明する。先ず、リード73aが行うプロセッサ間共有オブジェクト33aに係る情報の読み出し操作の処理について図10で説明する。図10中、プロセッサA21が行う処理として説明するが、異種プロセッサであるプロセッサB25においても同様に実行可能である。

30

【0046】

図10は、プロセッサ間共有オブジェクトに係る情報の読み出し操作の処理フローを示す図である。図10(A)において、プロセッサ間中間コード実行環境70において、プロセッサA21は、参照用プロセッサ間共有オブジェクトを作成する(ステップS10r)。そして、プロセッサA21は、アプリケーションプログラム232からプロセッサ間共有オブジェクトを参照可能とする処理を行って(ステップS20r)、プロセッサ間共有オブジェクトに係る情報の読み出し操作を行う(ステップS30r)。

【0047】

図10(B)では、図10(A)のステップS20rでのプロセッサ間共有オブジェクト参照の処理フローを説明する。図10(B)において、プロセッサA21は、参照用プロセッサ間共有オブジェクトを呼び出して(ステップS21r)、オブジェクト対応機能73fによって、アプリケーションプログラム232が指定した名前に対応付けられるプロセッサ間共有オブジェクト33aの実体をプロセッサ間共有メモリ31のプロセッサ間共有オブジェクト名テーブル35を参照して特定する(ステップS22r)。そして、対応付けられるプロセッサ間共有オブジェクト33aを戻り値として取得し(ステップS23r)、参照用プロセッサ間共有オブジェクトにプロセッサ間共有オブジェクト33aを返却する(ステップS24r)。参照用プロセッサ間共有オブジェクトがプロセッサ間共有オブジェクト33aとして振る舞うようになる。

40

【0048】

50

図10 (C) では、図10 (A) のステップ S 3 0 r でのプロセッサ間共有オブジェクトの読み出し操作の処理フローを説明する。図10 (C) において、プロセッサ A 2 1 は、アプリケーションプログラム 2 3 2 からのリード 7 3 a の呼び出しによって引数を受け付けてその引数を解析し (ステップ S 3 1 r 及び S 3 2 r) 、プロセッサ間共有メモリ 3 1 又は排他制御機能 4 2 に備えられるプロセッサ間排他制御機能 3 7 を使用して、プロセッサ A 2 1 がプロセッサ間共有メモリ 3 1 に格納されているプロセッサ間共有オブジェクト 3 3 a に係る情報を読み出すためのセマフォ又はロックを取得する (ステップ S 3 3 r)

【 0 0 4 9 】

そして、高速データ転送部 7 9 b を使用して、プロセッサ間共有メモリ領域 3 2 のプロセッサ間ヒープ領域 3 3 に格納されている、プロセッサ間共有オブジェクト名テーブル 3 5 を参照して名前と対応付けしたプロセッサ間共有オブジェクト 3 3 a から、プロセッサ A 2 1 の専用メモリ領域 (プロセッサ A 用メモリ領域 2 3 1 に展開されたヒープ領域 6 4) の参照用プロセッサ間共有オブジェクトへと、プロセッサ間共有オブジェクト 3 3 a に係る情報をデータ転送によってデータコピーする (ステップ S 3 4 r) 。

【 0 0 5 0 】

データコピーが終了すると、プロセッサ間共有メモリ排他制御によるセマフォ又はロックを解放する (ステップ S 3 5 r) 。そして、戻り値を作成して (ステップ S 3 6 r) 、戻り値をアプリケーションプログラム 2 3 2 へ返却することによって (ステップ S 3 7 r) 、リード 7 3 a の処理を終了する。

【 0 0 5 1 】

図11 は、読み出しプログラムの例を示す図である。図11 に示す読み出しプログラム 3 1 0 は、アプリケーションプログラム 2 3 2 及び 2 7 2 においてプロセッサ間共有オブジェクト 3 3 a に係る情報の読み出し操作を行うためのプログラムの例である。読み出しプログラム 3 1 0 において、p r g 3 1 2 「CPUSharedMemory_xxx obj ;」の実行によって、図10 (A) のステップ S 1 0 r による参照用プロセッサ間共有オブジェクトが作成される。

【 0 0 5 2 】

また、p r g 3 1 4 「obj = (CPUSharedMemory_xxx) CPUSharedMemoryNaming.lookup (" SharedObjectName_XXX ");」の実行によって、図10 (B) のプロセッサ間共有オブジェクトへの参照処理が行われ、指定した名前 (SharedObjectName_XXX) と参照用プロセッサ間共有オブジェクト (CPUSharedMemory_xxx) との対応がなされる。指定した名前によって、プロセッサ間共有オブジェクト名テーブル 3 5 に登録されているプロセッサ間共有オブジェクト 3 3 a の実体が参照用プロセッサ間共有オブジェクトから参照可能となり、プロセッサ間共有オブジェクト 3 3 a として振る舞うことができる。

【 0 0 5 3 】

更に、p r g 3 1 6 「data = obj.read();」の実行によって、図10 (C) のプロセッサ間共有オブジェクトの読み出し操作が行われ、プロセッサ間共有メモリ 3 1 から読み出したプロセッサ間共有オブジェクト 3 3 a に係る情報が値dataに設定される。

【 0 0 5 4 】

次に、ライト 7 3 b が行うプロセッサ間共有オブジェクト 3 3 a に係る情報の書き込み操作の処理について図12 で説明する。図12 中、プロセッサ A 2 1 が行う処理として説明するが、異種プロセッサであるプロセッサ B 2 5 においても同様に実行可能である。

【 0 0 5 5 】

図12 は、プロセッサ間共有オブジェクトに係る情報の書き込み操作の処理フローを示す図である。図12 (A) において、プロセッサ間中間コード実行環境 7 0 において、プロセッサ A 2 1 は、参照用プロセッサ間共有オブジェクトを作成する (ステップ S 1 0 w) 。そして、プロセッサ A 2 1 は、アプリケーションプログラム 2 3 2 からプロセッサ間共有オブジェクトを参照可能とする処理を行って (ステップ S 2 0 w) 、プロセッサ間共有オブジェクトに係る情報の読み出し操作を行う (ステップ S 3 0 w) 。

【 0 0 5 6 】

図12 (B) では、図12 (A) のステップ S 2 0 w でのプロセッサ間共有オブジェクト参照の処理フローを説明する。図12 (B) において、プロセッサ A 2 1 は、参照用プロセッサ間共有オブジェクトを呼び出して (ステップ S 2 1 w)、オブジェクト対応機能 7 3 f によって、アプリケーションプログラム 2 3 2 が指定した名前に対応付けられるプロセッサ間共有オブジェクト 3 3 a の実体をプロセッサ間共有メモリ 3 1 のプロセッサ間共有オブジェクト名テーブル 3 5 を参照して特定する (ステップ S 2 2 w)。そして、対応付けられるプロセッサ間共有オブジェクト 3 3 a を戻り値として取得し (ステップ S 2 3 w)、参照用プロセッサ間共有オブジェクトにプロセッサ間共有オブジェクト 3 3 a を返却する (ステップ S 2 4 w)。参照用プロセッサ間共有オブジェクトがプロセッサ間共有オブジェクト 3 3 a として振る舞うようになる。

10

【 0 0 5 7 】

図12 (C) では、図12 (A) のステップ S 3 0 w でのプロセッサ間共有オブジェクトの読み出し操作の処理フローを説明する。図12 (C) において、プロセッサ A 2 1 は、アプリケーションプログラム 2 3 2 からのライト 7 3 b の呼び出しによって引数を受け付けてその引数を解析し (ステップ S 3 1 w 及び S 3 2 w)、プロセッサ間共有メモリ 3 1 又は排他制御機能 4 2 に備えられるプロセッサ間排他制御機能 3 7 を使用して、プロセッサ A 2 1 がプロセッサ間共有メモリ 3 1 に格納されているプロセッサ間共有オブジェクト 3 3 a に係る情報を書き出すためのセマフォ又はロックを取得する (ステップ S 3 3 w)。

20

【 0 0 5 8 】

そして、高速データ転送部 7 9 b を使用して、プロセッサ A 2 1 の専用メモリ領域 (プロセッサ A 用メモリ領域 2 3 1 に展開されたヒープ領域 6 4) の参照用プロセッサ間共有オブジェクトから、プロセッサ間共有メモリ領域 3 2 のプロセッサ間ヒープ領域 3 3 に格納されている、プロセッサ間共有オブジェクト名テーブル 3 5 を参照して名前と対応付けしたプロセッサ間共有オブジェクト 3 3 a へと、プロセッサ間共有オブジェクト 3 3 a に係る情報をデータ転送によってデータコピーする (ステップ S 3 4 w)。

【 0 0 5 9 】

データコピーが終了すると、プロセッサ間共有メモリ排他制御によるセマフォ又はロックを解放する (ステップ S 3 5 w)。そして、戻り値を作成して (ステップ S 3 6 w)、戻り値をアプリケーションプログラム 2 3 2 へ返却することによって (ステップ S 3 7 w)、ライト 7 3 b の処理を終了する。

30

【 0 0 6 0 】

図13 は、書き込みプログラムの例を示す図である。図13 に示す書き込みプログラム 3 2 0 は、アプリケーションプログラム 2 3 2 及び 2 7 2 においてプロセッサ間共有オブジェクト 3 3 a に係る情報の書き込み操作を行うためのプログラムの例である。書き込みプログラム 3 2 0 において、p r g 3 2 2 「CPUSharedMemory_xxx obj ;」の実行によって、図12 (A) のステップ S 1 0 w による参照用プロセッサ間共有オブジェクトが作成される。

【 0 0 6 1 】

また、p r g 3 2 4 「obj = (CPUSharedMemory_xxx) CPUSharedMemoryNaming.lookup(“SharedObjectName_XXX”);」の実行によって、図12 (B) のプロセッサ間共有オブジェクト参照の処理が行われ、指定した名前 (SharedObjectName_XXX) と参照用プロセッサ間共有オブジェクト (CPUSharedMemory_xxx) との対応がなされる。指定した名前によって、プロセッサ間共有オブジェクト名テーブル 3 5 に登録されているプロセッサ間共有オブジェクト 3 3 a の実体が参照用プロセッサ間共有オブジェクトから参照可能となり、プロセッサ間共有オブジェクト 3 3 a として振る舞うことができる。

40

【 0 0 6 2 】

更に、p r g 3 2 6 「obj.write(data);」の実行によって、図12 (C) のプロセッサ間共有オブジェクトの書き込み操作が行われ、プロセッサ間共有メモリ 3 1 のプロセッサ

50

間共有オブジェクト 3 3 a にプロセッサ間共有オブジェクト 3 3 a に係る情報が書き込まれる。

【 0 0 6 3 】

図14は、プロセッサ間共有オブジェクトを登録するための処理フローを示す図である。図14中、プロセッサ A 2 1 が行う処理として説明するが、異種プロセッサであるプロセッサ B 2 5 においても同様に実行可能である。

【 0 0 6 4 】

図14において、プロセッサ間中間コード実行環境 7 0 において登録 7 3 c が実行されると、プロセッサ A 2 1 は、プロセッサ間共有メモリ・クラス 7 1 (以下、単にクラスと言う) が登録されているかを確認し (ステップ S 4 1)、登録済みであるか否かを判断する (ステップ S 4 2)。登録済みである場合、ステップ S 4 9 へと進む。一方、登録済みでない場合、プロセッサ A 専用メモリ 1 3 上のメソッド領域 6 3 (図 8) にクラスを生成して (ステップ S 4 3)、生成したクラスを登録する (ステップ S 4 4)。

【 0 0 6 5 】

そして、プロセッサ A 2 1 は、登録したクラスを関連するクラスとリンクした後 (ステップ S 4 5)、準備処理として初期化する (ステップ S 4 6)。更に、プロセッサ A 2 1 は、アクセス制御をクラスに設定し (ステップ S 4 7)、クラスを初期化する (ステップ S 4 7)。

【 0 0 6 6 】

次に、プロセッサ A 2 1 は、メソッド領域 6 3 に生成したクラスのインスタンスをプロセッサ間共有メモリ 3 1 のプロセッサ間ヒープ領域 3 3 (図 6) にプロセッサ間共有オブジェクト 3 3 a として生成する (ステップ S 4 9)。プロセッサ間共有オブジェクト 3 3 a がプロセッサ間ヒープ領域 3 3 (図 6) 内に生成される際には、プロセッサ間共有メモリ管理機能 3 6 を用いたヒープメモリ操作・管理機能 7 2 a (図 8) が実行され、メモリ割り当て / 開放部 3 6 a によってプロセッサ間共有メモリ 3 1 のプロセッサ間ヒープ領域 3 3 へのメモリ割り当てが行われる。

【 0 0 6 7 】

更に、プロセッサ A 2 1 は、オブジェクト対応機能 7 3 f によって、プロセッサ間共有オブジェクト 3 3 a の名前と実体との対応をプロセッサ間共有メモリ 3 1 のプロセッサ間共有オブジェクト名テーブル 3 5 に登録する (ステップ S 5 0)。

【 0 0 6 8 】

図15は、登録プログラムの例を示す図である。図15に示す登録プログラム 3 3 0 は、プロセッサ間共有メモリ・クラス 7 1 をインスタンス化してプロセッサ間共有オブジェクト 3 3 a を生成し、登録するためのプログラムの例である。登録プログラム 3 3 0 において、`prg 3 3 2 「CPUSharedMemory_xxx obj = new CPUSharedMemory_xxx ()」` の実行によって、新たにプロセッサ間共有オブジェクト 3 3 a がプロセッサ間共有メモリ 3 1 上に生成される。そして、`prg 3 3 4 「CPUShaedMemoryNaming.rebind(" SharedObjectName_XXX ", obj);」` の実行によって、プロセッサ間共有オブジェクト 3 3 a の名前と実体との対応がプロセッサ間共有オブジェクト名テーブル 3 5 に登録される。

【 0 0 6 9 】

図16は、プロセッサ間共有オブジェクトを削除するための処理フローを示す図である。図16中、プロセッサ A 2 1 が行う処理として説明するが、異種プロセッサであるプロセッサ B 2 5 においても同様に実行可能である。

【 0 0 7 0 】

図16において、プロセッサ間中間コード実行環境 7 0 において削除 7 3 d が実行されると、プロセッサ A 2 1 は、オブジェクト対応機能 7 3 f によって、プロセッサ間共有オブジェクト 3 3 a の名前と実体との対応をプロセッサ間共有メモリ 3 1 のプロセッサ間共有オブジェクト名テーブル 3 5 から削除する (ステップ S 6 1)。

【 0 0 7 1 】

次に、プロセッサ A 2 1 は、プロセッサ間共有メモリ 3 1 上にあるインスタンス (プロ

10

20

30

40

50

セッサ間共有オブジェクト 33a) を削除する (ステップ S62)。この場合、ヒープメモリ捜査・管理機能 72a によって、プロセッサ間共有メモリ 31 のメモリ割り当て / 開放部 36a が実行されてプロセッサ間共有オブジェクト 33a の領域が開放される。

【0072】

そして、プロセッサ A21 は、プロセッサ A 専用メモリ 23 のメソッド領域 63 にロードされたプロセッサ間共有メモリ・クラス 71 が削除可能な場合に削除する (ステップ S63)。

【0073】

図 17 は、削除プログラムの例を示す図である。図 17 に示す削除プログラム 340 は、`prg 312` 「`obj = (CPUSharedMemory_xxx) CPUShaedMemoryNaming.lookup(" SharedObjectName_XXX ");`」の実行によって、オブジェクト対応機能 73f により、プロセッサ間共有オブジェクト名テーブル 35 が参照され、指定した名前 (SharedObjectName_XXX) に対応するプロセッサ間共有オブジェクト 33a (CPUSharedMemory_xxx) の実体を取得する。

10

【0074】

そして、`prg 314` 「`CPUShaedMemoryNaming.finalize (" SharedObjectName_XXX, obj ");`」の実行によって、オブジェクト対応機能 73f により、指定した名前 (SharedObjectName_XXX) と対応するプロセッサ間共有オブジェクト 33a (CPUSharedMemory_xxx) の実体とがプロセッサ間共有オブジェクト名テーブル 35 から削除される。更に、`prg 316` 「`System.runFinalize();`」の実行によって、クラスを削除する。

20

【0075】

上述したように、本発明では、下位にある OS やハードウェアの違いを吸収し、異種プロセッサ上であっても上位にあるアプリケーションプログラムのソースをコンパイルした中間コードを共通の実行環境で動作可能とするプログラム実行環境を備えたシステムにおいて、2 以上の異種プロセッサによってアクセス可能とするプロセッサ間共有メモリを備えるようにする。

【0076】

本発明に係るプログラム実行環境は、一のプロセッサのみで実行されるアプリケーションプログラムを実行可能する他、プロセッサ間共有メモリを介して中間コードによって実行されるオブジェクトをプロセッサ間で共有するために、各プロセッサ専用メモリに作成した共有オブジェクトを、その共有オブジェクトを指定する名前とプロセッサ間共有メモリ内に作成した共有オブジェクトの実体とを対応させることによって、各プロセッサは読み出し及び書き込みの操作をプロセッサ間共有メモリ内の共有オブジェクトの実体に対して行うことができる。従って、各プロセッサ上で動作するアプリケーションプログラムを高速に連携動作させることができる。

30

【0077】

また、このようなプログラム実行環境では、プロセッサ間共有メモリに対して共有オブジェクトの実体を格納するためのメモリ領域の割り当て / 開放する管理機能と、プロセッサ間のプロセッサ間共有メモリに対する排他制御のためのセマフォ又はロック機能と、プロセッサ間共有メモリ内の共有オブジェクトの実体をプロセッサ専用メモリ上に割り当てたメモリ領域へ読み出し、また、プロセッサ専用メモリ内の共有オブジェクトの実体をプロセッサ間共有メモリ上に割り当てたメモリ領域へ書き出す機能と、プロセッサ間共有メモリに対するヒープ機能とを備えることによって、異種プロセッサ間でプロセッサ間共有メモリへのアクセスをプロセッサ専用メモリへのアクセスと同様に制御することができる。

40

【0078】

よって、本発明に係るシステムでは、LAN-IF を介することなく、また LAN ドライバを不要とし、高速に異種プロセッサ間でオブジェクトを共有することができる。

【0079】

本発明は、複数の異種プロセッサ (ヘテロジニアスマルチプロセッサ) で成るシステム

50

に限定されることなく、例えば、中間コードを実行可能なプログラム実行環境を備えた非対称型マルチプロセッシング方式の複数の同種のマルチプロセッサで成るシステムにも適応可能である。

【0080】

更に、複数の同種のマルチプロセッサで成るシステムが、均一メモリアクセス (Uniform Memory Access/Architecture: UMA) 又は非均一メモリアクセス (Non-Uniform Memory Access/Architecture: NUMA) を実装、集中共有メモリ (Centralized Shared Memory: CSM) 又は分散共有メモリ (Distributed Shared Memory: DSM) を搭載等による共有メモリ型マルチプロセッサシステムであっても、非リモートメモリアクセス (No Remote memory Access: NORA) を実装、分散非共有メモリ (Distributed Non-Shared Memory) を搭載等による非共有メモリ型マルチプロセッサシステムであっても、すべてのプロセッサからアプリケーションプログラムが格納されているメモリを参照できない場合に、プロセッサ間共有メモリを備え、本発明に係るプログラム実行環境を適応することによってアプリケーションプログラムを高速に連携動作させることができる。

10

【0081】

均一メモリアクセスは、すべてのプロセッサが同じスピードでメインメモリにアクセスできる方式であり、メモリアクセスコストの均一性を保つことにより、対称型マルチプロセッシング (Symmetric Multi-Processing (SMP)) システムに実装される。非均一メモリアクセスは、すべてのプロセッサが全てのメモリにアクセスできる方式であるが、プロセッサからメインメモリへのアクセスのコストがアクセスするメモリ領域とプロセッサに依存して均一ではないメモリ・アーキテクチャである。また各ノードのメモリを全プロセッサに共有の物理アドレス空間にマップできなければならない等の制約がある。また、非リモートメモリアクセスは、各ノードのメモリを全プロセッサに共通の物理アドレスにマップしない。

20

【0082】

上記実施例において、オブジェクト指向プログラミング言語の Java (登録商標) 言語を一例として説明した。Java (登録商標) 言語 (プログラムソース) は、中間コード (バイトコード) にコンパイルされ、Java (登録商標) 仮想マシンとしての OS やハードウェアに依存しないプログラム実行環境で実行される。Java (登録商標) 仮想マシンでは、Java (登録商標) の中間コード (バイトコード) を JIT コンパイラ等によってハードウェアに依存するネイティブコードに変換して実行される。

30

【0083】

本発明は、このようなプログラム実行環境を備えるシステムであれば適用可能であり、例えば、オブジェクト指向プログラミング言語の C# 言語、共通中間言語 (Common Intermediate Language: CIL) 等によるアプリケーションプログラムを実行するシステムであってもよい。例えば、この場合、共通言語基盤 (Common Language Infrastructure: CLI) がプログラム実行環境に相当する。

【0084】

以上の説明に関し、更に以下の項を開示する。

(付記1)

40

複数のプロセッサを含み、前記複数のプロセッサ上で動作可能な中間コードを実行するプログラム実行環境を備えたプログラム実行システムであって、

前記複数のプロセッサ夫々に専用の複数のプロセッサ専用メモリと、

前記複数のプロセッサ間で共有される前記中間コードによって操作される共有オブジェクトを格納するプロセッサ間共有メモリとを備え、

前記複数のプロセッサの各々は、

前記共有オブジェクトを指定する名前と前記プロセッサ間共有メモリ内の該共有オブジェクトの実体とを対応させることによって、前記複数のプロセッサ専用メモリの各々と該プロセッサ間共有メモリ間にて該共有オブジェクトを読み出し及び書き込みする読出書込機能を実行するようにしたプログラム実行システム。

50

(付記 2)

前記プロセッサ間共有メモリは、前記共有オブジェクトの名前と該共有オブジェクトの実体とを対応させた対応テーブルを備え、

前記複数のプロセッサの各々は、前記共有オブジェクトの名前と該共有オブジェクトの実体との対応を前記対応テーブルに書き込むことにより該共有オブジェクトを登録し、また該登録された共有オブジェクトを該対応テーブルから削除することにより該共有オブジェクトを削除する登録削除機能を実行するようにした付記 1 記載のプログラム実行システム。

(付記 3)

前記複数のプロセッサの各々は、

前記プロセッサ間共有メモリのメモリ領域の割り当て及び開放を管理する共有メモリ管理機能と、

前記複数のプロセッサ間の前記プロセッサ間共有メモリへのアクセスに対してセマフォ又はロック機能を用いて排他制御を行うプロセッサ間排他制御機能と、

前記プロセッサ間共有メモリのヒープ領域に前記共有オブジェクトを格納するための共有メモリヒープ機能とを実行するようにした付記 1 又は 2 記載のプログラム実行システム。

(付記 4)

前記プロセッサ間共有メモリのメモリ領域と前記複数のプロセッサ専用メモリの各々のメモリ領域との間のダイレクトメモリアccessによるデータ転送を制御するDMAコントローラを更に備え、

前記複数のプロセッサの各々において、前記DMAコントローラを用いて前記高速データ転送を行うルーチンと該ルーチンを前記プログラム実行環境から呼び出すためのインターフェイスを備えるようにした付記 1 乃至 3 のいずれか一項記載のプログラム実行システム。

(付記 5)

前記プロセッサ間排他制御機能を有するハードウェアを備え、

前記複数のプロセッサの各々において、前記ハードウェアのプロセッサ間排他制御機能を用いて排他制御を行うルーチンと該ルーチンを前記プログラム実行環境から呼び出すためのインターフェイスを備えるようにした付記 1 乃至 3 のいずれか一項記載のプログラム実行システム。

(付記 6)

複数の異種プロセッサ毎に専用の専用メモリを備えたヘテロジニアスマルチプロセッサであって、

前記複数の異種プロセッサ間で共有される該各異種プロセッサ上で動作可能な中間コードによって操作される共有オブジェクトを格納するプロセッサ間共有メモリを備え、

前記各専用メモリは、

前記共有オブジェクトを指定する名前と前記プロセッサ間共有メモリ内の該共有オブジェクトの実体とを対応させることによって、前記各プロセッサ専用メモリと該プロセッサ間共有メモリ間にて該共有オブジェクトを読み出し及び書き込みする読出書込機能を前記異種プロセッサに実行させるプログラムを格納するようにしたヘテロジニアスマルチプロセッサ。

(付記 7)

前記各専用メモリは、

前記共有オブジェクトの名前と該共有オブジェクトの実体との対応を前記プロセッサ間共有メモリが保持する対応テーブルに書き込むことにより該共有オブジェクトを登録し、また該登録された共有オブジェクトを該対応テーブルから削除することにより該共有オブジェクトを削除する登録削除機能を前記異種プロセッサに実行させるプログラムを格納するようにした付記 6 記載のヘテロジニアスマルチプロセッサ。

(付記 8)

10

20

30

40

50

前記各専用メモリは、

前記プロセッサ間共有メモリのメモリ領域の割り当て及び開放を管理する共有メモリ管理機能と、

前記複数のプロセッサ間の前記プロセッサ間共有メモリへのアクセスに対してセマフォ又はロック機能を用いて排他制御を行うプロセッサ間排他制御機能と、

前記プロセッサ間共有メモリのヒープ領域に前記共有オブジェクトを格納するための共有メモリヒープ機能とを前記異種プロセッサに実行させるプログラムを格納するようにした付記6又は7記載のヘテロジニアスマルチプロセッサ。

【0085】

本発明は、具体的に開示された実施例に限定されるものではなく、特許請求の範囲から逸脱することなく、種々の変形や変更が可能である。

【図面の簡単な説明】

【0086】

【図1】疎結合のヘテロジニアスマルチプロセッサのシステム例を示す図である。

【図2】従来のヘテロジニアスマルチプロセッサによるシステムの構成例を示す図である。

。

【図3】従来の中間コード実行環境の構成例を示す図である。

【図4】本発明の第一実施例に係る密結合のヘテロジニアスマルチプロセッサによるシステム例を示す図である。

【図5】プロセッサ間共有メモリの領域構成例を示す図である。

【図6】ヘテロジニアスマルチプロセッサによるシステムの構成例を示す図である。

【図7】プロセッサ間共有メモリ用ライブラリの構成例を示す図である。

【図8】各プロセッサ用実行環境とプロセッサ間共有メモリ用ライブラリとによる機能構成を示す図である。

【図9】本発明の第二実施例に係る密結合のヘテロジニアスマルチプロセッサによるシステム構成例を示す図である。

【図10】プロセッサ間共有オブジェクトに係る情報の読み出し操作の処理フローを示す図である。

【図11】読み出しプログラムの例を示す図である。

【図12】プロセッサ間共有オブジェクトに係る情報の書き込み操作の処理フローを示す図である。

【図13】書き込みプログラムの例を示す図である。

【図14】プロセッサ間共有オブジェクトを登録するための処理フローを示す図である。

【図15】登録プログラムの例を示す図である。

【図16】プロセッサ間共有オブジェクトを削除するための処理フローを示す図である。

【図17】削除プログラムの例を示す図である。

【符号の説明】

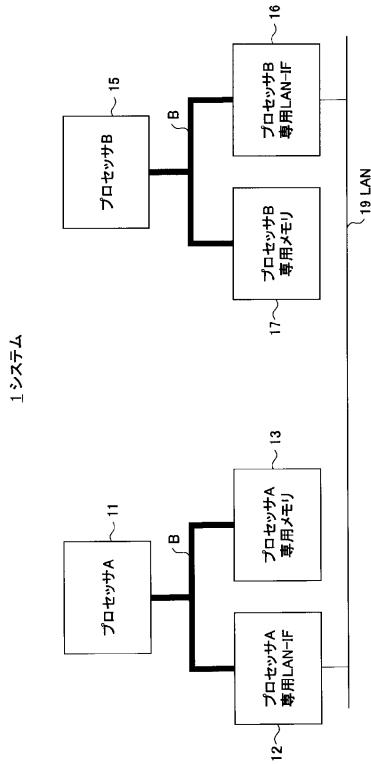
【0087】

2 1	プロセッサ A	
2 2	プロセッサ A 専用 LAN - I F	40
2 3	プロセッサ A 専用メモリ	
2 5	プロセッサ B	
2 6	プロセッサ B 専用 LAN - I F	
2 7	プロセッサ B 専用メモリ	
3 1	プロセッサ間共有メモリ	
3 2	プロセッサ間共有メモリ領域	
3 3	プロセッサ間ヒープ領域	
3 4	プロセッサ間排他制御用領域	
3 4 a	プロセッサ間セマフォ	
3 4 b	プロセッサ間ロック	50

3 5	プロセッサ間共有オブジェクト名テーブル	
3 6	プロセッサ間共有メモリ管理機能	
3 6 a	メモリ割り当て / 開放部	
3 7	プロセッサ間排他制御機能	
5 4	J a v a (登録商標) V M	
5 4 a	クラスローダー	
5 4 b	クラス・キャッシュ機能	
5 4 c	ヒープメモリ管理機能	
5 4 d	ガーベジ・コレクション (G C) 機能	
6 1	実行時データ領域	10
6 2	P C レジスタ	
6 3	メソッド領域	
6 4	ヒープ領域	
6 5	J a v a (登録商標) V M スタック	
6 5 2	フレーム	
6 5 3	ローカル変数	
6 5 4	オペランドスタック	
6 6	実行時定数プール	
6 7	N a t i v e メソッドスタック	
7 0	プロセッサ間中間コード実行環境	20
7 1	プロセッサ間共有メモリ・クラス	
7 2	プロセッサ間共有中間コードのヒープメモリ	
7 2 a	ヒープメモリ操作・管理機能	
7 2 b	プロセッサ間共有ガーベジ・コレクション (G C) 機能	
7 3	プロセッサ間共有オブジェクト操作機能	
7 3 a	リード	
7 3 b	ライト	
7 3 c	登録	
7 3 d	削除	
7 3 f	オブジェクト対応機能	30
7 4	プロセッサ間共有クラスローダー	
7 5	プロセッサ間共有クラス・キャッシュ機能	
7 9	プロセッサ間データ転送機能	
7 9 a	引数 / 戻り値のデータ転送部	
7 9 b	高速データ転送部	
1 0 0	システム	
2 3 1	プロセッサ A 用メモリ領域	
2 3 2	アプリケーションプログラム	
2 3 3	アプリケーションデータ	
2 3 4	プロセッサ A 用実行環境	40
2 3 5	プロセッサ A 用 J I T コンパイラ	
2 3 6	プロセッサ間共有メモリ用ライブラリ	
2 7 1	プロセッサ B 用メモリ領域	
2 7 2	アプリケーションプログラム	
2 7 3	アプリケーションデータ	
2 7 4	プロセッサ B 用実行環境	
2 7 5	プロセッサ B 用 J I T コンパイラ	
2 7 6	プロセッサ間共有メモリ用ライブラリ	

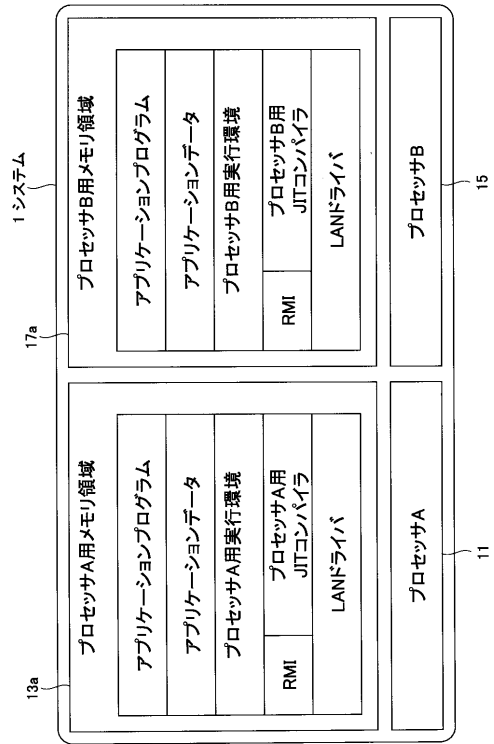
【 図 1 】

疎結合のヘテロジニアスマルチプロセッサのシステム例を示す図



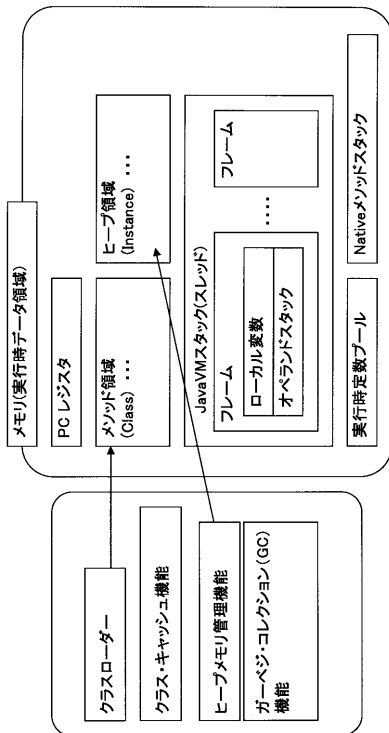
【 図 2 】

従来のヘテロジニアスマルチプロセッサによるシステムの構成例を示す図



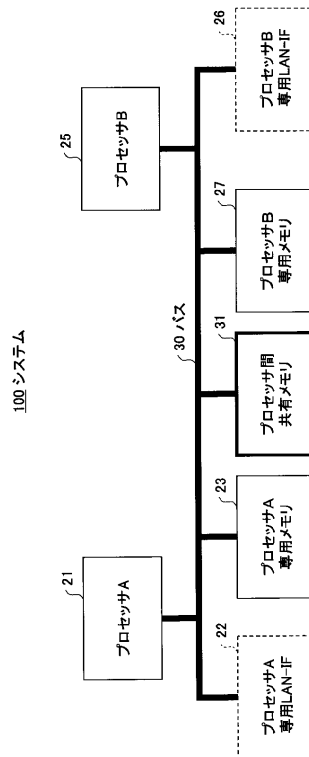
【 図 3 】

従来の中間コード実行環境の構造例を示す図



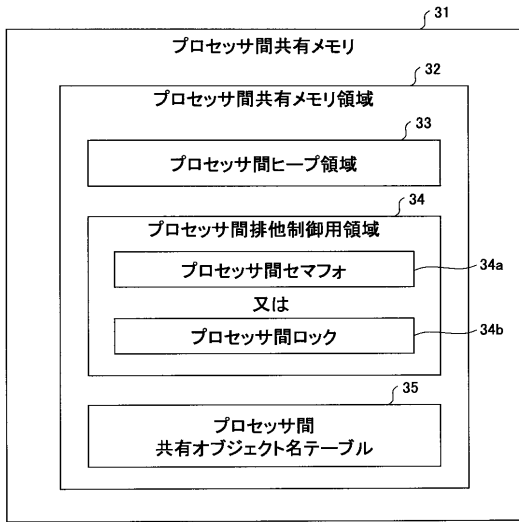
【 図 4 】

本発明の第一実施例に係る密結合のヘテロジニアスマルチプロセッサによるシステム例を示す図



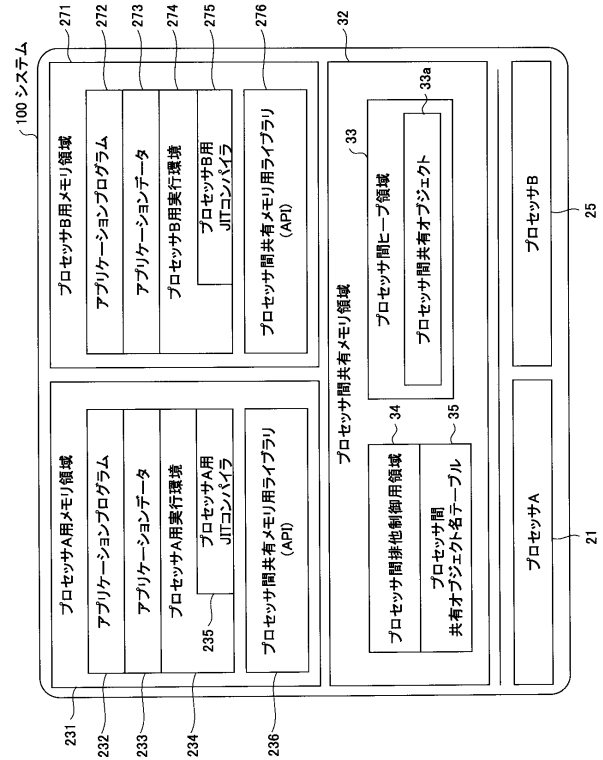
【図5】

プロセッサ間共有メモリの領域構成例を示す図



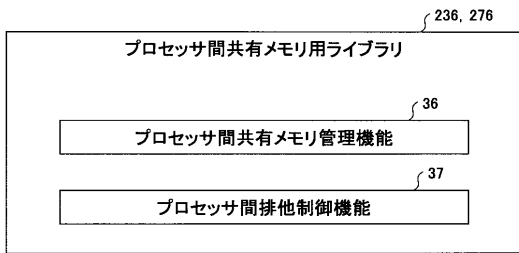
【図6】

ヘテロニアスマルテプロセッサによるシステムの構成例を示す図



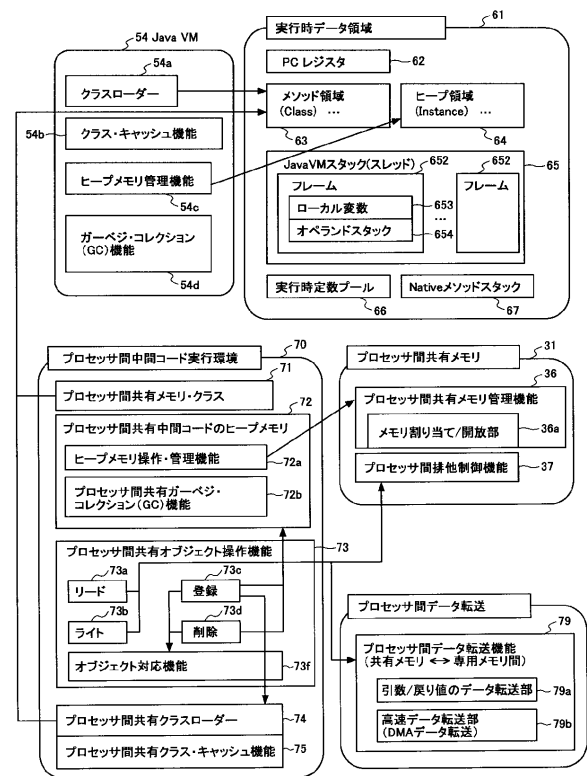
【図7】

プロセッサ間共有メモリ用ライブラリの構成例を示す図



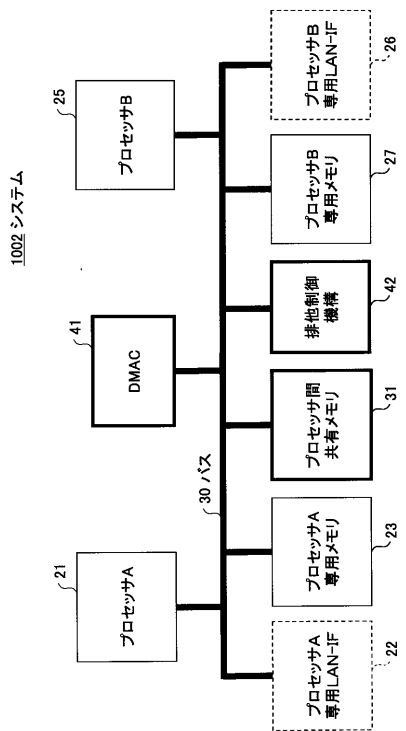
【図8】

各プロセッサ用実行環境とプロセッサ間共有メモリ用ライブラリによる機能構成を示す図



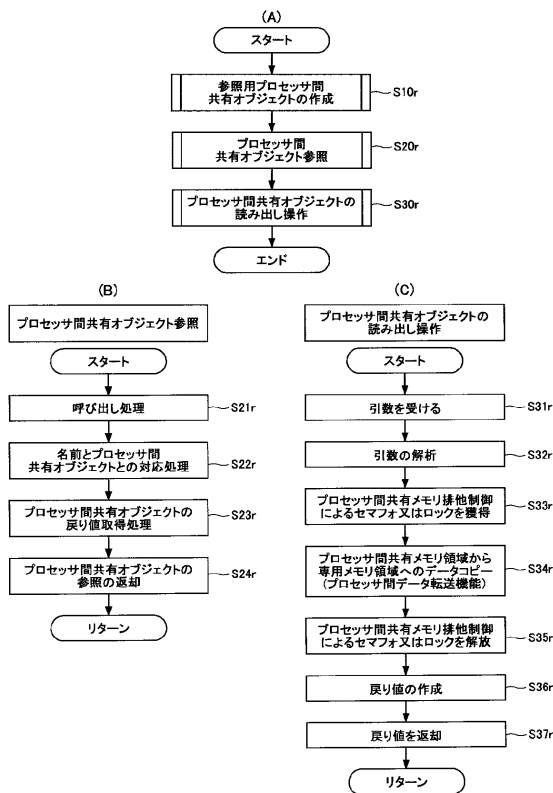
【図9】

本発明の第二実施例に係る
密結合のヘテロニアスマルチプロセッサによるシステム構成例を示す図



【図10】

プロセッサ間共有オブジェクトに係る
情報の読み出し操作の処理フローを示す図



【図11】

読み出しプログラムの例を示す図

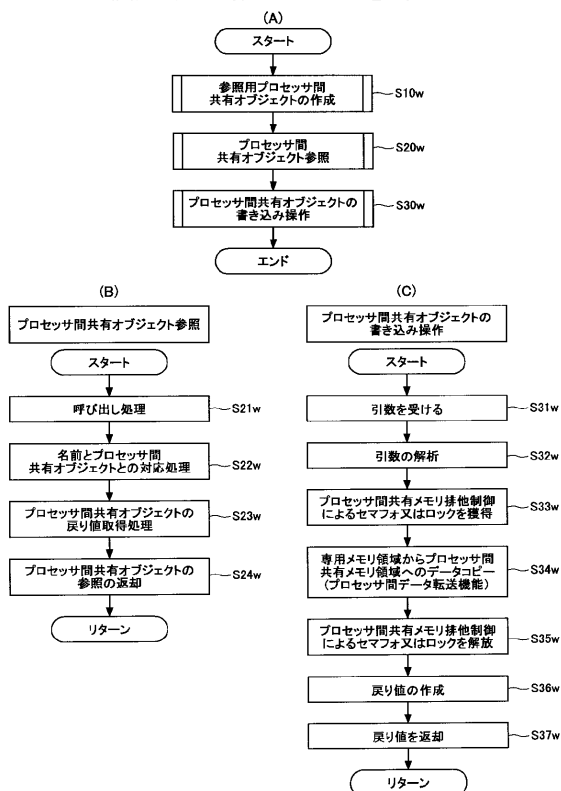
```

310 読み出しプログラム
Public class ProgramName {
  Public static void main () {
    String data;
    CPUSharedMemory xxx obj; ~ prg312
    Try {
      obj = (CPUSharedMemory xxx)
        CPUSharedMemoryNaming.lookup("SharedObjectName_XXX");
      data = obj.read(); ~ prg316
    }
  }
}

```

【図12】

プロセッサ間共有オブジェクトに係る
情報の書き込み操作の処理フローを示す図



【図 13】

書き込みプログラムの例を示す図

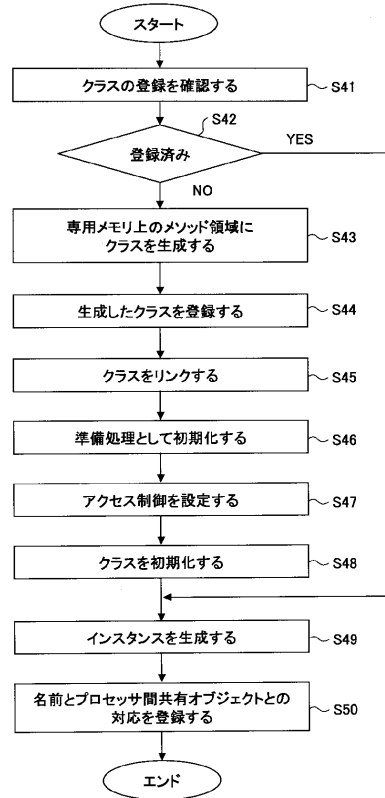
```

320 書き込みプログラム
Public class ProgramName {
Public static void main () {
String data, tmp1;
CPUSharedMemory_XXX obj; ~ prg322
Try {
    prg324
obj = (CPUSharedMemory_XXX)
CPUSharedMemoryNaming.lookup("SharedObjectName_XXX");
obj.write(data); ~ prg326
}
}

```

【図 14】

プロセッサ間共有オブジェクトを登録するための処理フローを示す図



【図 15】

登録プログラムの例を示す図

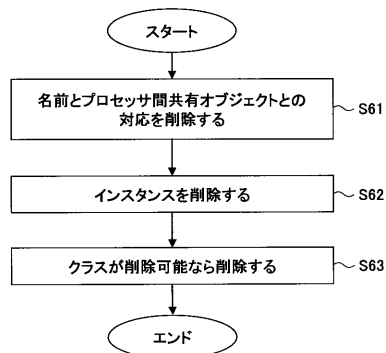
```

330 登録プログラム
Public class ProgramName {
Public static void main () {
String data, tmp1;
CPUSharedMemory_XXX obj;
Try {
    prg332
CPUSharedMemory_XXX obj = new CPUSharedMemory_XXX (); ~ prg332
CPUSharedMemoryNaming.rebind("SharedObjectName_XXX", obj); ~ prg334
}
}

```

【図 16】

プロセッサ間共有オブジェクトを削除するための処理フローを示す図



【 図 17 】

削除プログラムの例を示す図

```

} 340 削除プログラム
Public class ProgramName {
Public static void main () {
String data, tmp1;
CPUSharedMemory_xxx obj ;
Try {
obj = (CPUSharedMemory_xxx)
CPUShaedMemoryNaming.lookup("SharedObjectName_XXX");
CPUShaedMemoryNaming.finalize ("SharedObjectName_XXX", obj); ~ prg344
System.runFinalize(); ~ prg346
}
}
}
```

フロントページの続き

(56)参考文献 特表2007-538324(JP,A)
特表2008-500633(JP,A)
特開2007-122741(JP,A)
特開2006-268123(JP,A)
特開2000-172509(JP,A)

(58)調査した分野(Int.Cl., DB名)

G06F 9/46 - 9/54
G06F 15/16 - 15/177