(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
2012 年 10 月 4 日(04.10.2012)

WIPO | PCT

(10) 国際公開番号
WO 2012/131884 A1

(51) 国際特許分類:
G06F 21/24 (2006.01) *G06F 12/14* (2006.01)

(21) 国際出願番号: PCT/JP2011/057715

(22) 国際出願日: 2011 年 3 月 28 日(28.03.2011)

(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 (米国を除く全ての指定国について): 富士通株式会社(FUJITSU LIMITED) [JP/JP]; 〒2118588 神奈川県川崎市中原区上小田中４丁目１番１号 Kanagawa (JP).

(72) 発明者; および

(75) 発明者/出願人 (米国についてのみ): 山下 浩一郎(YAMASHITA, Koichiro) [JP/JP]; 〒2118588 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 Kanagawa (JP). 山内 宏真(YAMAUCHI, Hiromasa) [JP/JP]; 〒2118588 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 Kanagawa (JP). 鈴木 貴久(SUZUKI, Takahisa) [JP/JP]; 〒2118588 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 Kanagawa (JP). 栗原 康志(KURIHARA, Koji) [JP/JP]; 〒2118588 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 Kanagawa (JP).

(74) 代理人: 酒井 昭徳(SAKAI, Akinori); 〒1006020 東京都千代田区霞が関３丁目２番５号 霞が関ビルディング２０階 酒井総合特許事務所 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, 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, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, 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.

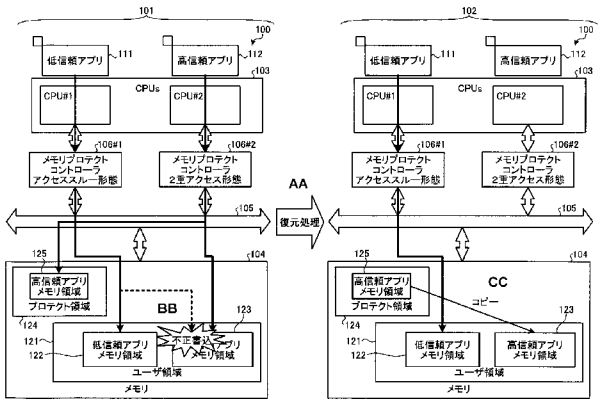
(84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM,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

[続葉有]

(54) **Title**: MULTICORE PROCESSOR SYSTEM

(54) 発明の名称 : マルチコアプロセッサシステム

[図1]



104 Memory
111 Low-reliability application
112 High-reliability application
106#1 Memory protection controller access-through mode
106#2 Memory protection controller dual-access mode
121 User region
122 Low-reliability application memory region
123, 125 High-reliability application memory region
124 Protection region
AA Restoration process
BB Unauthorized writing
CC Copy

(57) **Abstract**: A multicore processor system (100) operates in a state in which a high-reliability application (112) and a low-reliability application (111) are jointly loaded. With the multicore processor system (100) a CPU (#1) executes the low-reliability application (111), and a CPU (#2) executes the high-reliability application (112). A memory protection controller (106#2) accesses a high-reliability application memory region (123) within a user region (121), and accesses a high-reliability application memory region (125) within a protection region (124). When the data in the high-reliability application memory region (123) is destroyed due to unauthorized writing by the low-reliability application (111), the CPU (#2) restores the data by copying the data in high-reliability application memory region (125) to high-reliability application memory region (123).

(57) 要約: マルチコアプロセッサシステム（１００）を、高信頼アプリ（１１２）と低信頼アプリ（１１１）が混載した状態で運用する。マルチコアプロセッサシステム（１００）は、ＣＰＵ（＃１）が低信頼アプリ（１１１）を実行し、ＣＰＵ（＃２）が高信頼アプリ（１１２）を実行する状態である。メモリプロテクトコントローラ（１０６＃２）は、ユーザ領域（１２１）内の高信頼アプリメモリ領域（１２３）にアクセスするとともに、プロテクト領域（１２４）内の高信頼アプリメモリ領域（１２５）にアクセスする。低信頼アプリ（１１１）による不正書込によって、高信頼アプリメモリ領域（１２３）のデータが破壊された場合、ＣＰＵ（＃２）は、高信頼アプリメモリ領域（１２５）のデータを高信頼アプリメモリ領域（１２３）にコピーすることでデータを復元する。

WO 2012/131884 A1

(BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

添付公開書類:

— 国際調査報告（条約第 21 条(3)）

明　細　書

発明の名称 ： マルチコアプロセッサシステム

技術分野

[0001] 本発明は、ソフトウェアを保護するマルチコアプロセッサシステムに関する。

背景技術

[0002] 近年、１つのシステム内に、複数のコアを有するマルチコアプロセッサシステムの形態を採用する機器が増加している。１つのシステム内に１つのコアを有するシングルコアプロセッサシステム用のアプリケーションソフトウェア（以下、「アプリ」と称する）をマルチコアプロセッサシステムで動作させた場合、多くの障害が発生する可能性が存在する。また、新たにマルチコアプロセッサシステム向けに開発されたソフトウェアであっても、複数のコアが同時に動作する並列動作状態での検証について、試験者が全てのタイミングを検証することは困難であった。次に、マルチコアプロセッサシステムでどのような障害が起きるかということを図１０、図１１を参照して説明する。

[0003] 図１０は、シングルコアプロセッサシステム向けのソフトウェアをマルチコアプロセッサシステム上で動作する場合の動作例を示す説明図である。符号１００１で示す説明図では、アプリ１００３をシングルコアプロセッサシステムで動作する場合について説明し、符号１００２で示す説明図では、アプリ１００３をマルチコアプロセッサシステムで動作する場合について説明する。

[0004] 初めに、符号１００１で示す説明図にて、アプリ１００３内のメインスレッド１００４は、関数ｆ（ｘ）を呼び出し、ｆ（ｘ）の結果を利用する。そこで、メインスレッド１００４は、ｆ（ｘ）を実行するオフロード処理用のスレッド１００５を起動し、スレッド１００５の終了後、ｆ（ｘ）の結果を利用し、処理を続行する。なお、スレッドとはＣＰＵで行う処理を管理する

ための基本単位である。

[0005] アプリ１００３がシングルコアプロセッサシステムで動作する場合、メインスレッド１００４は、時刻ｔ１でスレッド１００５を起動する。続けて、スレッド１００５が時刻ｔ２にてｆ（ｘ）の処理を終了した後、時刻ｔ３にて、メインスレッド１００４は、結果を利用する。

[0006] 次に、符号１００２で示す説明図にて、アプリ１００３がマルチコアプロセッサシステムで動作する場合について説明する。アプリ１００３が、マルチコアプロセッサシステム環境で動作検証されておらず安全でない場合、符号１００６で示す動作となり、マルチコアプロセッサシステム環境で動作検証されており安全である場合、符号１００７で示す動作となる。なお、以下の説明にて、マルチコアプロセッサシステム環境で動作検証されておらず安全でないアプリを低信頼アプリと呼称し、マルチコアプロセッサシステム環境で動作検証されており安全であるアプリを高信頼アプリと呼称する。なお、マルチコアプロセッサシステムで対象のアプリに信頼性があるか否かを検証する方法としては、様々なタイミングに対して検証を行う方法が存在する。

[0007] アプリ１００３が低信頼アプリである場合、ＣＰＵ（Ｃｅｎｔｒａｌ Ｐｒｏｃｅｓｓｉｎｇ Ｕｎｉｔ）＃１で実行するメインスレッド１００４は、時刻ｔ１でスレッド１００５をＣＰＵ＃２上で起動する。ＣＰＵ＃２は、スレッド１００５と他スレッドが割り当てられている状態を想定している。スレッド１００５は、時刻ｔ３にてｆ（ｘ）の処理を終了する。

[0008] また、ＣＰＵ＃１はメインスレッド１００４のみを実行しているため、メインスレッド１００４の割当時間がシングルコアプロセッサシステム時より多くなる。結果、時刻ｔ３より早い時刻ｔ２にて、メインスレッド１００４は、ｆ（ｘ）の結果を利用する。時刻ｔ２では、まだｆ（ｘ）の処理が終了していないため、メインスレッド１００４は、予期せぬ値を読み込むことになり、障害が発生する。

[0009] アプリ１００３が高信頼アプリである場合、メインスレッド１００４は、

ｆ（ｘ）の結果を利用する前に同期処理が挿入されている。これにより、メインスレッド１００４は、時刻ｔ２にて、スレッドｆ（ｘ）の結果を利用する前で同期処理を実行する。同期処理が実行されたメインスレッド１００４は、スレッド１００５が処理終了するまで待機し、安全に処理を実行することができる。

[0010] 図１１は、シングルコアプロセッサシステム上で呼び出されていたライブラリがマルチコアプロセッサシステム上で呼び出される場合の説明図である。符号１１０１で示す説明図は、ライブラリＡがシングルコアプロセッサシステム上で動作する場合を示しており、符号１１０２で示す説明図は、ライブラリＡがマルチコアプロセッサシステム上で動作する場合を示している。

[0011] 符号１１０１で示す説明図にて、アプリＸは、ライブラリＡを呼び出している。また、アプリＹは、ライブラリＡとライブラリＢを呼び出している。このように、シングルコアプロセッサシステム上で呼び出されるライブラリＡは、特別な管理を行わなくてよい。

[0012] 符号１１０２で示す説明図にて、アプリＸはＣＰＵ＃１で実行されており、アプリＹはＣＰＵ＃２で実行されている。ライブラリＡが共有リソースを有する場合、ライブラリＡは、ライブラリＡのコンテキスト１１０３に対して排他制御を行うといった対策が取られる。具体的な対策として、ＣＰＵ＃１上のライブラリＡは、インターフェース１１０４＃１を通じ、ＩＰＣ（Ｉｎｔｅｒ－Ｐｒｏｃｅｓｓｏｒ　Ｃｏｍｍｕｎｉｃａｔｉｏｎ）１１０５＃１、ＩＰＣ１１０５＃２を用いてＣＰＵ＃２のライブラリＡと排他制御を行う。このような対策が行われているライブラリは高信頼のライブラリとなり、対策が行われていないライブラリは低信頼のライブラリとなる。

[0013] このような、高信頼アプリ、低信頼アプリを実行する場合の技術として、たとえば、システムを、ドメイン分割し、安全な動作をする高信頼アプリ群となる安全ドメインと、安全な動作をする保証がない低信頼アプリ群となる非安全ドメインとに分割する。安全ドメインは、メモリ内の非安全ドメインがアクセスできない安全データにアクセスする技術が開示されている（たと

えば、下記特許文献１、２を参照。）。

[0014] また、データを保護する技術として、たとえば、フラッシュメモリの例として、未使用の予備ブロックを探し出し、保護領域用のエラーブロックとする技術が開示されている。また、データを保護する他の技術として、たとえば、メモリを２重化する領域を設け、重要なプログラムやデータの破壊を防ぐ技術が開示されている（たとえば、下記特許文献３、４を参照。）。

先行技術文献

特許文献

[0015] 特許文献1：特表２００６－５０６７５４号公報

特許文献2：特開２００４－１７１５６３号公報

特許文献3：特開２００４－３１０７７０号公報

特許文献4：特開平５－１０８４９３号公報

発明の概要

発明が解決しようとする課題

[0016] 上述した従来技術を適用し、高信頼アプリ、低信頼アプリを実行する場合、マルチコアプロセッサシステムは、高信頼アプリ、低信頼アプリを分割して運用することになる。したがって、従来技術を適用したマルチコアプロセッサシステムは、高信頼アプリと低信頼アプリを混載して運用することが困難であるという問題があった。

[0017] 本発明は、上述した従来技術による問題点を解消するため、高信頼アプリと低信頼アプリを混載して運用できるマルチコアプロセッサシステムを提供することを目的とする。

課題を解決するための手段

[0018] 上述した課題を解決し、目的を達成するため、本発明の一側面によれば、複数のＣＰＵと、メモリと、複数のＣＰＵとメモリとの間に配置されるメモリプロテクトコントローラと、を含み、メモリプロテクトコントローラは、アプリケーション実行時に複数のＣＰＵのアクセス要求によって第１メモリ

領域にアクセスするとともに、システムブート時に確保された第２メモリ領域にアクセスするマルチコアプロセッサシステムが提案される。

発明の効果

[0019] 本発明の一側面によれば、高信頼アプリと低信頼アプリを混載して運用できるという効果を奏する。

図面の簡単な説明

[0020] [図1]図１は、本実施の形態にかかるマルチコアプロセッサシステム１００の障害発生時の動作例と復元処理の動作例を示す説明図である。

[図2]図２は、マルチコアプロセッサシステム１００のハードウェア例を示すブロック図である。

[図3]図３は、マルチコアプロセッサシステム１００の機能例を示すブロック図である。

[図4]図４は、メモリプロテクトコントローラ１０６の内部の機能例を示す説明図である。

[図5]図５は、メモリプロテクトコントローラ１０６の登録例を示す説明図である。

[図6]図６は、マルチコアプロセッサシステム１００の起動時の処理手順の一例を示すフローチャートである。

[図7]図７は、マルチコアプロセッサシステム１００のアプリ切替時の処理手順の一例を示すフローチャート（その１）である。

[図8]図８は、マルチコアプロセッサシステム１００のアプリ切替時の処理手順の一例を示すフローチャート（その２）である。

[図9]図９は、マルチコアプロセッサシステム１００のエラー時の処理手順の一例を示すフローチャートである。

[図10]図１０は、シングルコアプロセッサシステム向けのソフトウェアをマルチコアプロセッサシステム上で動作する場合の動作例を示す説明図である。

[図11]図１１は、シングルコアプロセッサシステム上で呼び出されていたラ

イブラリがマルチコアプロセッサシステム上で呼び出される場合の説明図である。

発明を実施するための形態

[0021] 以下に添付図面を参照して、開示のマルチコアプロセッサシステムの実施の形態を詳細に説明する。

[0022] 図１は、本実施の形態にかかるマルチコアプロセッサシステム１００の障害発生時の動作例と復元処理の動作例を示す説明図である。図１では、符号１０１で示す説明図にて、マルチコアプロセッサシステム１００の障害発生時の動作例を示し、符号１０２で示す説明図にて、マルチコアプロセッサシステム１００の復元処理の動作例を示す。

[0023] 図１におけるマルチコアプロセッサシステム１００は、複数のコアとなるＣＰＵｓ１０３と、メモリ１０４とを含む。マルチコアプロセッサシステム１００は、携帯電話といった携帯端末を想定している。ＣＰＵｓ１０３には、ＣＰＵ＃１とＣＰＵ＃２が含まれる。以下、接尾記号“＃ｎ”が付随された記号は、ｎ番目のＣＰＵに対応する記号であることを示している。ＣＰＵｓ１０３とメモリ１０４は、バス１０５で接続されている。また、ＣＰＵ＃１、ＣＰＵ＃２は、それぞれ、メモリプロテクトコントローラ１０６＃１、メモリプロテクトコントローラ１０６＃２と通信可能である。

[0024] メモリプロテクトコントローラ１０６は、２重アクセス形態とアクセススルー形態という、２つの形態のうち指示された形態に応じた動作をする装置である。２重アクセス形態の指示がある場合、メモリプロテクトコントローラ１０６は、アプリ本来の要求先アドレスにアクセスしつつ、保護されたメモリ領域内のアドレスにアクセスする機能を有している。また、アクセススルー形態の指示がある場合、メモリプロテクトコントローラ１０６は、アプリ本来の要求先アドレスにアクセスする。

[0025] また、マルチコアプロセッサシステム１００は、低信頼アプリ１１１と高信頼アプリ１１２を実行する。具体的には、ＣＰＵ＃１が低信頼アプリ１１１を実行し、ＣＰＵ＃２が高信頼アプリ１１２を実行する。また、マルチコ

アプロセッサシステム１００は、低信頼アプリ１１１のコンテキストが格納される領域として、ユーザ領域１２１内に、低信頼アプリメモリ領域１２２を確保する。同様に、マルチコアプロセッサシステム１００は、高信頼アプリ１１２のコンテキストが格納される領域として、ユーザ領域１２１内に、高信頼アプリメモリ領域１２３を確保する。なお、コンテキストとは、ＣＰＵのレジスタの値となるプログラムカウンタ、スタックポインタなどといったアプリが使用するデータである。

[0026] さらに、マルチコアプロセッサシステム１００は、ユーザ領域１２１とは別の領域となるプロテクト領域１２４を確保し、プロテクト領域１２４内に、高信頼アプリメモリ領域１２５を確保する。高信頼アプリメモリ領域１２５は、高信頼アプリメモリ領域１２３と同一のデータが格納される。同一のデータの格納方法として、ＣＰＵ＃２は、高信頼アプリ１１２が実行するタイミングで、メモリプロテクトコントローラ１０６＃２に２重アクセス形態の登録を行う。２重アクセス形態となっているメモリプロテクトコントローラ１０６＃２は、ライトアクセスが発生した場合、書き込み内容を高信頼アプリメモリ領域１２３に書き込みつつ、同内容を高信頼アプリメモリ領域１２５にも書き込む。

[0027] 符号１０１で示す説明図では、このように高信頼アプリメモリ領域１２３が保護されている状態で、低信頼アプリ１１１が不正書込を行い、高信頼アプリメモリ領域１２３のデータを破壊した場合を想定している。ＣＰＵ＃２は、高信頼アプリ１１２を実行する際に、高信頼アプリメモリ領域１２３を読み込んで実行しようとするが、高信頼アプリメモリ領域１２３のデータは正しい値を示しておらず、高信頼アプリ１１２が強制終了する。たとえば、高信頼アプリメモリ領域１２３のプログラムカウンタが不正なアドレスに書き換えられた場合、ＣＰＵ＃２は、不正なアドレスの値を実行コードとして実行しようとし、高信頼アプリ１１２が強制終了する。

[0028] なお、不正書込が行われることによって高信頼アプリ１１２が強制終了する場合以外に、ＯＳがハングする場合もある。ＯＳがハングする場合の動作

については、図９にて説明する。

[0029] 次に、符号１０２で示す説明図では、データ破壊後の復元処理を示している。ＣＰＵ＃２は、高信頼アプリ１１２に対してエラーが発生したことを検出すると、高信頼アプリ１１２が実行中であったか否かを判定する。高信頼アプリ１１２が実行中であった場合、ＣＰＵ＃２は、高信頼アプリメモリ領域１２３と高信頼アプリメモリ領域１２５を比較し、差分が存在する場合、高信頼アプリメモリ領域１２３を高信頼アプリメモリ領域１２５で上書きする。これにより、不正書込が行われたデータが正常なデータに復元し、ＣＰＵ＃２は、高信頼アプリ１１２を続行することができるため、高信頼アプリ１１２を保護することができる。

[0030] （マルチコアプロセッサシステム１００のハードウェア）

図２は、マルチコアプロセッサシステム１００のハードウェア例を示すブロック図である。図２において、マルチコアプロセッサシステム１００は、ＣＰＵを複数搭載するＣＰＵｓ１０３と、ＲＯＭ（Ｒｅａｄ-Ｏｎｌｙ Ｍｅｍｏｒｙ）２０１と、ＲＡＭ（Ｒａｎｄｏｍ Ａｃｃｅｓｓ Ｍｅｍｏｒｙ）２０２と、を含む。また、マルチコアプロセッサシステム１００は、フラッシュＲＯＭ２０３と、フラッシュＲＯＭコントローラ２０４と、フラッシュＲＯＭ２０５と、を含む。なお、メモリ１０４は、ＲＡＭ２０２の全てであってもよいし、または一部であってもよい。さらに、メモリ１０４は、ＲＯＭ２０１、フラッシュＲＯＭ２０３、フラッシュＲＯＭ２０５を含んでいてもよい。

[0031] また、マルチコアプロセッサシステム１００は、ユーザやその他の機器との入出力装置として、ディスプレイ２０６と、Ｉ／Ｆ（Ｉｎｔｅｒｆａｃｅ）２０７と、キーボード２０８と、を含む。また、各部はバス１０５によってそれぞれ接続されている。

[0032] ここで、ＣＰＵｓ１０３は、マルチコアプロセッサシステム１００の全体の制御を司る。ＣＰＵｓ１０３は、シングルコアのプロセッサを並列して接続した全てのＣＰＵを指している。ＣＰＵｓ１０３は、ＣＰＵ＃１～ＣＰＵ

＃ｘを含む。なお、ｘは２以上の整数である。また、マルチコアプロセッサシステムとは、コアが複数搭載されたプロセッサを含むコンピュータのシステムである。コアが複数搭載されていれば、複数のコアが搭載された単一のプロセッサでもよく、シングルコアのプロセッサが並列されているプロセッサ群でもよい。なお、本実施の形態では、シングルコアのプロセッサであるＣＰＵが並列されている形態を例にあげて説明する。

[0033] また、ＣＰＵ＃１～ＣＰＵ＃ｘは、それぞれ、メモリプロテクトコントローラ１０６と通信可能である。さらに、ＣＰＵ＃１～ＣＰＵ＃ｘは、それぞれ専用のキャッシュメモリを有してもよい。

[0034] ＲＯＭ２０１は、ブートプログラムなどのプログラムを記憶している。ＲＡＭ２０２は、ＣＰＵｓ１０３のワークエリアとして使用される。フラッシュＲＯＭ２０３は、ＯＳ（Ｏｐｅｒａｔｉｎｇ Ｓｙｓｔｅｍ）などのシステムソフトウェアやアプリケーションソフトウェアなどを記憶している。たとえば、ＯＳを更新する場合、マルチコアプロセッサシステム１００は、Ｉ／Ｆ２０７によって新しいＯＳを受信し、フラッシュＲＯＭ２０３に格納されている古いＯＳを、受信した新しいＯＳに更新する。

[0035] フラッシュＲＯＭコントローラ２０４は、ＣＰＵｓ１０３の制御にしたがってフラッシュＲＯＭ２０５に対するデータのリード／ライトを制御する。フラッシュＲＯＭ２０５は、フラッシュＲＯＭコントローラ２０４の制御で書き込まれたデータを記憶する。データの具体例としては、マルチコアプロセッサシステム１００を使用するユーザがＩ／Ｆ２０７を通して取得した画像データ、映像データや、本実施の形態にかかるソフトウェア保護方法を実行するプログラムが格納されていてもよい。フラッシュＲＯＭ２０５は、たとえば、メモリカード、ＳＤカードなどを採用することができる。

[0036] ディスプレイ２０６は、カーソル、アイコンあるいはツールボックスをはじめ、文書、画像、機能情報などのデータを表示する。たとえば、ディスプレイ２０６は、ＴＦＴ液晶ディスプレイなどを採用することができる。

[0037] Ｉ／Ｆ２０７は、通信回線を通じてＬＡＮ（Ｌｏｃａｌ Ａｒｅａ Ｎｅ

ｔｗｏｒｋ）、ＷＡＮ（Ｗｉｄｅ　Ａｒｅａ　Ｎｅｔｗｏｒｋ）、インターネットなどのネットワーク２０９に接続され、ネットワーク２０９を介して他の装置に接続される。そして、Ｉ／Ｆ２０７は、ネットワーク２０９と内部のインターフェースを司り、外部装置からのデータの入出力を制御する。Ｉ／Ｆ２０７には、たとえばモデムやＬＡＮアダプタなどを採用することができる。

[0038] キーボード２０８は、数字、各種指示などの入力のためのキーを有し、データの入力を行う。また、キーボード２０８は、タッチパネル式の入力パッドやテンキーなどであってもよい。

[0039] （マルチコアプロセッサシステム１００の機能例）

次に、マルチコアプロセッサシステム１００の機能例について説明する。図３は、マルチコアプロセッサシステム１００の機能例を示すブロック図である。なお、マルチコアプロセッサシステム１００は、各機能からアクセスされる記憶領域として、高信頼ホワイトリスト３０１にアクセス可能である。

[0040] マルチコアプロセッサシステム１００は、確保部３１１と検出部３１２と比較部３１３と、通知部３１４と、登録部３１５と、検出部３１６と、検出部３１７と、比較部３１８と、復元部３１９と、を含む。この制御部となる機能（確保部３１１～復元部３１９）は、記憶装置に記憶されたプログラムをＣＰＵ＃１～ＣＰＵ＃ｘが実行することにより、その機能を実現する。記憶装置とは、具体的には、たとえば、図２に示したＲＯＭ２０１、ＲＡＭ２０２、フラッシュＲＯＭ２０３、フラッシュＲＯＭ２０５などである。または、Ｉ／Ｆ２０７を経由して他のＣＰＵが実行することにより、その機能を実現してもよい。

[0041] また、図３では、確保部３１１は、高信頼アプリ１１２を実行するＣＰＵ＃２の機能となっているが、ＣＰＵ＃１、ＣＰＵ＃３～ＣＰＵ＃ｘのうちいずれか一つのＣＰＵの機能であってもよい。また、検出部３１２～復元部３１９は、ＣＰＵ＃１～ＣＰＵ＃ｘの全てのＣＰＵに含まれる機能であっても

よい。

[0042] なお、ＣＰＵ＃２は、ハイパーバイザ＃２、ＯＳ＃２、カーネル＃２を実行する。ハイパーバイザ＃２は、ＣＰＵ＃２などのハードウェア上で直接動作するプログラムである。ハイパーバイザ＃２は、ＣＰＵ＃２内のレジスタを直接参照したり、ＣＰＵ＃２内のレジスタの情報を読み出したり、ＣＰＵ＃２内のレジスタの情報を書き換えたりする特権命令を実行することができるプログラムである。カーネル＃２は、ＯＳ＃２の中核の機能であり、たとえば、マルチコアプロセッサシステム１００のリソースを管理し、スレッドなどのソフトウェアがハードウェアにアクセスできるようにする。

[0043] また、図示していないが、ＯＳ＃２は、ＣＰＵ＃２を制御するプログラムである。たとえば、マルチコアプロセッサシステム１００内のリソースにアクセスするライブラリ、ＡＰＩ（Ａｐｐｌｉｃａｔｉｏｎ Ｐｒｏｇｒａｍｍｉｎｇ Ｉｎｔｅｒｆａｃｅ）等をアプリに提供する。

[0044] なお、図示していないが、ＣＰＵ＃１、ＣＰＵ＃３～ＣＰＵ＃ｘも、ハイパーバイザ、ＯＳ、カーネルを実行する。確保部３１１～通知部３１４、検出部３１６は、カーネル＃２の機能に含まれ、登録部３１５、検出部３１７～復元部３１９はハイパーバイザ＃２の機能に含まれる。

[0045] 高信頼ホワイトリスト３０１は、所定のアプリを登録するリストである。登録される所定のアプリとは、マルチコアプロセッサシステム１００での動作検証が行われた高信頼アプリ１１２である。たとえば、高信頼アプリ１１２は、マルチコアプロセッサシステム１００に予めプリインストールされているアプリである。また、マルチコアプロセッサシステム１００を製造するメーカ、ネットワーク２０９を提供する通信キャリア等が提供するアプリを高信頼アプリとして登録してもよい。たとえば、マルチコアプロセッサシステム１００は、ネットワーク２０９からアプリをダウンロードした際に、メーカまたは通信キャリアが提供することを示す識別情報が付随していた場合、ダウンロードしたアプリを高信頼ホワイトリスト３０１に登録してもよい。

[0046] なお、高信頼ホワイトリスト３０１に登録される内容は、たとえば、アプリの名称であってもよいし、またファイルシステムが存在する場合、アプリのプログラムが格納されたファイルパスであってもよい。

[0047] 確保部３１１は、システムブート時に、アプリ実行時に複数のＣＰＵがアクセスする第１メモリ領域に対応するとともに、複数のＣＰＵが第１メモリ領域にアクセスするときに同様にアクセスされる第２メモリ領域を確保する機能を有する。なお、第１メモリ領域は、実行中アプリのアクセス先となるメモリ領域であり、図３で示すユーザ領域１２１となる。また、第２メモリ領域は、ユーザ領域１２１のデータを保護する領域となるプロテクト領域１２４となる。また、第２メモリが確保されるタイミングとしては、システムブート時であってもよいし、アプリが起動されるときであってもよい。

[0048] また、確保部３１１は、マルチコアプロセッサシステム１００での同時に実行するアプリの数がＮであり、アプリ実行時にて割り当てられるメモリ領域がＭ［バイト］である場合、第２メモリ領域の大きさをＮ×Ｍ［バイト］で確保してもよい。たとえば、確保部３１１は、Ｎ＝５、Ｍ＝１［Mバイト］である場合、Ｎ×Ｍ＝５×１＝５［Mバイト］の領域をユーザ領域１２１として確保する。

[0049] また、Ｎの値として、本実施の形態にかかるマルチコアプロセッサシステム１００は、携帯電話等を想定しており、多数のアプリを同時起動することを想定していない。Ｎの最大値としては、たとえば、８、１６といった値になる。また、Ｎは、ＣＰＵｓ１０３の数であってもよい。なお、確保された領域のアドレスは、ＣＰＵ＃２のレジスタ、キャッシュメモリ、ＲＡＭ２０２等の記憶領域に記憶される。

[0050] 検出部３１２は、アプリが起動したというイベント、アプリが終了したというイベント、またはアプリが切り替えられたというイベントのうち、いずれかのイベントを検出する機能を有する。たとえば、検出部３１２は、高信頼アプリ１１２が起動されたというイベントを検出する。なお、検出したイベントは、ＣＰＵ＃２のレジスタ、キャッシュメモリ、ＲＡＭ２０２等の記

憶領域に記憶される。

[0051] 比較部３１３は、検出部３１２によってイベントが検出された場合、実行されるアプリと所定のアプリのリストに登録されたアプリとを比較する機能を有する。たとえば、比較部３１３は、実行される高信頼アプリ１１２と高信頼ホワイトリスト３０１に登録されたアプリを比較する。なお、比較結果は、ＣＰＵ＃２のレジスタ、キャッシュメモリ、ＲＡＭ２０２等の記憶領域に記憶される。

[0052] 通知部３１４は、比較部３１３による比較結果が一致を示すとき、実行されるアプリからのアクセスについて、メモリプロテクトコントローラ１０６＃２が第１メモリ領域と第２メモリ領域とにアクセスする指示をハイパーバイザ＃２に通知する機能を有する。たとえば、高信頼アプリ１１２と高信頼ホワイトリスト３０１に登録されたアプリが一致を示す場合を想定する。このとき、通知部３１４は、高信頼アプリ１１２からのアクセスについて、メモリプロテクトコントローラ１０６＃２が高信頼アプリメモリ領域１２３と高信頼アプリメモリ領域１２５とにアクセスする指示をハイパーバイザ＃２に通知する。

[0053] また、通知部３１４は、比較部３１３による比較結果が不一致を示すとき、実行されるアプリからのアクセスについて、メモリプロテクトコントローラ１０６＃２が第１メモリ領域にアクセスする指示をハイパーバイザ＃２に通知してもよい。

[0054] また、通知部３１４は、アプリが起動され、起動されたアプリの比較部３１３による比較結果が一致を示すとき、第２メモリ領域内に起動されたアプリ用の領域を追加する指示を通知してもよい。また、通知部３１４は、アプリが終了し、終了したアプリの比較部３１３による比較結果が一致を示すとき、第２メモリ領域内の終了したアプリ用の領域を解放する指示を通知してもよい。なお、通知された内容は、ＣＰＵ＃２のレジスタ、キャッシュメモリ、ＲＡＭ２０２等の記憶領域に記憶されてもよい。

[0055] 登録部３１５は、通知部３１４によって通知された指示をメモリプロテク

トコントローラ１０６＃２に登録する機能を有する。たとえば、登録部３１５は、第１アクセス形態である２重アクセス形態の指示が通知された場合、メモリプロテクトコントローラ１０６＃２に２重アクセス形態を登録する。具体的に、登録部３１５は、メモリプロテクトコントローラ１０６＃２内のレジスタに登録する。

[0056] また、登録部３１５は、第２アクセス形態であるアクセススルー形態の指示が通知された場合、メモリプロテクトコントローラ１０６＃２にアクセススルー形態を登録する。なお、具体的な登録内容は、図４にて説明する。なお、登録を行ったという情報は、ＣＰＵ＃２のレジスタ、キャッシュメモリ、ＲＡＭ２０２等の記憶領域に記憶されてもよい。

[0057] 検出部３１６は、自ＣＰＵ内のアプリが強制終了したことを検出する機能を有する。たとえば、検出部３１６は、アクセス不可能なメモリにアクセスしたことを示すセグメントエラー等が発生し、高信頼アプリ１１２が強制終了したことを検出する。なお、アプリが強制終了したという情報は、ＣＰＵ＃２のレジスタ、キャッシュメモリ、ＲＡＭ２０２等の記憶領域に記憶される。

[0058] 検出部３１７は、ＯＳ＃２がハングしたことを検出する機能を有する。たとえば、検出部３１７は、ＯＳ＃２を周期監視し、ＯＳ＃２からの応答がない場合、ＯＳ＃２がハングしたことを検出する。なお、ハング中にあるＯＳは、異常状態となり、要求に対する応答が行えない状態である。ＯＳのハング状態は、カーネルパニックとも呼ばれる。なお、ＯＳ＃２がハングしたという情報は、ＣＰＵ＃２のレジスタ、キャッシュメモリ、ＲＡＭ２０２等の記憶領域に記憶される。

[0059] 比較部３１８は、検出部３１６によるアプリの強制終了が検出された場合、または検出部３１７によるＯＳ＃２のハングが検出された場合、第１メモリ領域と第２メモリ領域とを比較する機能を有する。たとえば、比較部３１８は、高信頼アプリメモリ領域１２３と高信頼アプリメモリ領域１２５のデータを比較する。また、比較部３１８は、実行中のアプリが高信頼ホワイト

リスト３０１に登録されている場合に、第１メモリ領域と第２メモリ領域とを比較してもよい。なお、比較結果は、ＣＰＵ＃２のレジスタ、キャッシュメモリ、ＲＡＭ２０２等の記憶領域に記憶される。

[0060] 復元部３１９は、比較部３１８による比較結果に基づいて、第１メモリ領域を復元する機能を有する。たとえば、復元部３１９は、比較部３１８による比較結果が不一致を示すとき、比較結果の差分を第１メモリ領域に上書きすることで、第１メモリ領域を復元する。なお、復元を行ったという結果は、ＣＰＵ＃２のレジスタ、キャッシュメモリ、ＲＡＭ２０２等の記憶領域に記憶されてもよい。

[0061] 図４は、メモリプロテクトコントローラ１０６の内部の機能例を示す説明図である。図４では、メモリプロテクトコントローラ１０６＃１を例にして説明を行うが、メモリプロテクトコントローラ１０６＃２～メモリプロテクトコントローラ１０６＃ｘも、メモリプロテクトコントローラ１０６＃１と同様の機能を含む。メモリプロテクトコントローラ１０６＃１は、記憶部４０１＃１、判定部４０２＃１、変換部４０３＃１を含む。

[0062] 記憶部４０１は、メモリプロテクトコントローラ１０６のアクセス形態とアプリに対応するプロテクト領域を記憶する。具体的に、記憶部４０１＃１は、制御レジスタ４０４＃１と格納レジスタ４０５＃１＿１～格納レジスタ４０５＃１＿ｙを含む。ｙは１以上の整数である。

[0063] なお、具体的なｙの値としては、ＣＰＵ＃１が同時実行可能なアプリの数の最大値でよい。本実施の形態にかかるマルチコアプロセッサシステム１００は、携帯電話等を想定しており、パーソナル・コンピュータのように多数のアプリを同時実行することを想定していない。したがって、各ＣＰＵで同時に起動される際の最大のアプリ数も、多数のアプリを同時起動することを想定していない数となる。さらに、ｙは、Ｎ以下の値となる。たとえば、ｙは４、または８といった値となる。

[0064] 制御レジスタ４０４＃１は、アクセス形態フラグ、プロテクト領域管理番号という２つのフィールドを含む。アクセス形態フラグフィールドには、２

箇所のアドレスにアクセスを行う２重アクセス形態、または通常のアクセスを行うアクセススルー形態、のうちいずれかのアクセス形態情報を示す識別子が格納される。たとえば、具体的な識別子として、“１”が２重アクセス形態を意味してもよい。プロテクト領域管理番号フィールドには、２箇所のアドレスにアクセスを行う場合、格納レジスタ４０５＃１＿１～格納レジスタ４０５＃１＿ｙのうち、どの格納レジスタ４０５＃１を適用するかを示す値が格納される。たとえば、プロテクト領域管理番号フィールドに“１”が格納された場合、変換部４０３＃１は、格納レジスタ４０５＃１＿１の設定を用いて変換を行う。

[0065] 格納レジスタ４０５＃１＿１～格納レジスタ４０５＃１＿ｙは、プロテクト領域をアプリごとに格納するレジスタである。また、格納レジスタ４０５＃１＿１～格納レジスタ４０５＃１＿ｙは、それぞれ、管理番号＿１～管理番号＿ｙに対応している。続けて、格納レジスタ４０５＃１の各フィールドについて説明を行う。格納レジスタ４０５＃１は、使用中ビット、アプリＩＤ、マスクアドレス、プロテクトアドレスという４つのフィールドを含む。

[0066] 使用中ビットフィールドには、該当の格納レジスタ４０５＃１が使用中か否かを示すビットが格納される。アプリＩＤフィールドには、ＣＰＵ＃１にて実行中のアプリの識別情報が格納される。マスクアドレスフィールドには、アプリのアクセスするメモリ範囲を示すアドレスが格納される。具体的なメモリ範囲の大きさは、ＯＳが有するメモリ管理機構によって決定される。プロテクトアドレスフィールドには、ＣＰＵ＃１にて実行中のアプリのプロテクト領域へのアドレスが格納される。なお、制御レジスタ４０４＃１、格納レジスタ４０５＃１の具体的な設定例は、図５にて後述する。

[0067] 判定部４０２＃１は、記憶部４０１＃１に記憶されたアクセス形態情報に応じてアドレス変換を行うか否かを判定する機能を有する。具体的には、判定部４０２＃１は、アクセス形態フラグフィールドが２重アクセス形態であれば、２箇所のアドレスにアクセスを行い、アクセス形態フラグフィールドがアクセススルー形態であればアドレス変換を行わず、アドレススルーする

。

[0068] 変換部４０３＃１は、判定部４０２＃１により２重アクセス形態であると判定された場合、一方のアクセス先についてアドレス変換を行う機能を有する。具体的には、変換部４０３＃１は、プロテクト領域管理番号フィールドに設定された管理番号に対応する格納レジスタ４０５＃１のマスクアドレスフィールドと、プロテクトアドレスフィールドとを参照してアドレス変換を行う。たとえば、変換部４０３は、変換後のアドレスを、下記（１）式によって変換する。

[0069] 変換後のアドレス＝変換前のアドレス＆マスクアドレス＋プロテクトアドレス…（１）

[0070] 具体的に、変換前のアドレスとなるアプリからのアクセス先が、０ｘ０５０００１００であり、マスクアドレスフィールドが０ｘ０００００ｆｆｆｆであり、プロテクトアドレスフィールドが０ｘ０１０００００００である場合を想定する。このとき、変換部４０３は、（１）にしたがって、以下のように変換する。

[0071] 変換後のアドレス＝０ｘ０５０００１００＆０ｘ０００００ｆｆｆｆ＋０ｘ０１０００００００

⇔変換後のアドレス＝０ｘ０１０００１００

[0072] なお、メモリプロテクトコントローラ１０６＃１は、ＣＰＵ＃１からのリードリクエストアクセスとライトリクエストアクセスのうち、ライトリクエストアクセスについて２重アクセス形態を行ってもよい。理由として、エラーが発生していない状態で、リードリクエストアクセスに関する２重アクセス形態を行っても同一の値が返ってくるので、変換を行わないアドレスに対するリードアクセスで十分なためである。

[0073] 図５は、メモリプロテクトコントローラ１０６の登録例を示す説明図である。図５で示すマルチコアプロセッサシステム１００は、高信頼ホワイトリスト３０１に登録されており、高信頼アプリとなるアプリＡ、アプリＣと、高信頼ホワイトリスト３０１に登録されておらず低信頼アプリとなるアプリ

Ｂ、アプリＤ、アプリＥを実行している。アプリＡとアプリＢは、ＣＰＵ＃１に割り当てられており、アプリＣ～アプリＥは、ＣＰＵ＃２に割り当てられている。さらに、ＣＰＵ＃１はアプリＡを実行しており、ＣＰＵ＃２はアプリＤを実行している。なお、アプリＡ～アプリＥのアプリＩＤは、それぞれ、０ｘ０００１、０ｘ０００２、０ｘ０００３、０ｘ０００４、０ｘ０００５となる。

[0074] カーネル＃１は、ＯＳがブートする際に、０ｘ００００００００～０ｘ０５０００００となるメモリ領域をカーネル占有領域５０１として確保する。続けて、カーネル＃１は、アプリの動作空間となるユーザ領域１２１を確保する前に、プロテクト領域１２４を確保する。たとえば、マルチコアプロセッサシステム１００のアプリ同時実行数Ｎ＝５であり、１つのアプリに対して許容するメモリサイズＭを１［Ｍバイト］と想定する。

[0075] このとき、カーネル＃１は、Ｎ×Ｍ＝５×１＝５［Ｍバイト］となる０ｘ０１０００００～０ｘ０１０４ｆｆｆｆをプロテクト領域１２４として確保する。続けて、カーネル＃１は、０ｘ０５０００００～０ｘｆｆｆｆｆｆｆｆをユーザ領域１２１として確保する。また、カーネル＃１は、ユーザ領域１２１からプロテクト領域１２４へのオフセットアドレスとして、０ｘ０５０００００－０ｘ０１０００００＝０ｘ０４０００００を記憶する。なお、Ｍ＝１［Ｍバイト］であるため、１つのアプリがアクセスするアドレス範囲となるマスクアドレスは、０ｘ０００ｆｆｆｆとなる。本実施の形態では、１つのアプリがアクセスするアドレス範囲は、説明の簡略化のため、常に固定であることを想定する。

[0076] このとき、プロテクト領域１２４は、アプリからはアクセスできない領域となる。このため、たとえば、アプリＡ、アプリＢがＯＳ＃１を通じてプロテクト領域１２４にアクセスしようとした場合、カーネル＃１は、ＯＳ＃１の管理外領域にアクセスすることを示すセグメントエラーを発行することになる。このように、マルチコアプロセッサシステム１００は、プロテクト領域１２４をユーザ領域１２１以外の領域とすることで、低信頼アプリによっ

てプロテクト領域１２４の不正書込を防ぐことができる。

[0077] なお、プロテクト領域１２４となる領域は、連続範囲かつ固定でなくてもよく、メモリプロテクトコントローラ１０６が管理可能であれば、分散して確保されてもよいし、マルチコアプロセッサシステム１００の実行中にダイナミックに変化してもよい。本実施の形態では、説明を簡略化するため、プロテクト領域１２４となる領域が連続範囲かつ固定で説明する。

[0078] また、マルチコアプロセッサシステム１００は、アプリＡ～アプリＥが起動される際に、各アプリのコンテキストを格納する領域をユーザ領域１２１内に確保する。具体的に、カーネル＃１は、アプリＡの起動時にアプリＡメモリ領域５０２を確保し、アプリＢの起動時にアプリＢメモリ領域５０３を確保する。また、カーネル＃２は、アプリＣの起動時にアプリＣメモリ領域５０４を確保し、アプリＤの起動時にアプリＤメモリ領域５０５を確保し、アプリＥの起動時にアプリＥメモリ領域５０６を確保する。アプリＡメモリ領域５０２～アプリＥメモリ領域５０６の先頭アドレスは、それぞれ、０ｘ０５０００００、０ｘ０５０１００００、０ｘ０５０２００００、０ｘ０５０３００００、０ｘ０５０４００００となる。

[0079] また、マルチコアプロセッサシステム１００は、高信頼アプリが起動する場合、プロテクト領域１２４内にもメモリ領域を確保する。具体的に、カーネル＃１から指示を受けたハイパーバイザ＃１が、アプリＡの起動時にアプリＡメモリ領域５０７を確保する。

[0080] アプリＡメモリ領域５０７のアドレスの設定方法として、たとえば、ハイパーバイザ＃１は、アプリＡメモリ領域５０２の先頭アドレスから、ユーザ領域１２１へのオフセットアドレスを減じたアドレスをアプリＡメモリ領域５０７の先頭アドレスに設定する。具体的に、ハイパーバイザ＃１は、アプリＡメモリ領域５０２の先頭アドレス０ｘ０５０００００－オフセットアドレス０ｘ０４０００００＝０ｘ０１０００００をアプリＡメモリ領域５０７の先頭アドレスに設定する。

[0081] 続けて、ハイパーバイザ＃１は、格納レジスタ４０５＃１__１に値を設定

する。具体的に、ハイパーバイザ＃１は、使用中ビットフィールドに使用中を設定し、アプリＩＤフィールドにアプリＡのアプリＩＤである０ｘ０００１を設定する。続けて、ハイパーバイザ＃１は、マスクアドレスフィールドに０ｘ００００ｆｆｆｆを設定し、プロテクトアドレスフィールドにアプリＡメモリ領域５０７の先頭アドレスである０ｘ０１０００００を設定する。

[0082] 同様に、ハイパーバイザ＃２は、アプリＣの起動時にアプリＣメモリ領域５０８を確保する。さらに、ハイパーバイザ＃２は、格納レジスタ４０５＃２＿１に値を設定する。具体的に、ハイパーバイザ＃２は、使用中ビットフィールドに使用中を設定し、アプリＩＤフィールドにアプリＣのアプリＩＤである０ｘ０００３を設定する。続けて、ハイパーバイザ＃２は、マスクアドレスフィールドに０ｘ００００ｆｆｆｆを設定し、プロテクトアドレスフィールドにアプリＣメモリ領域５０８の先頭アドレスである０ｘ０１０２００００を設定する。

[0083] 上述の段階で、マルチコアプロセッサシステム１００はアプリの起動の段階を終え、実際に動作する状態に遷移する。ＣＰＵ＃１は、アプリＡ、アプリＢ、アプリＡ、…、の順でアプリを実行する。アプリ切替イベントが発生し、ＣＰＵ＃１がアプリＡを実行するときに、アプリＡは高信頼ホワイトリスト３０１に登録されているため、カーネル＃１は、アクセス形態を２重アクセス形態に設定するようにハイパーバイザ＃１に指示する。このとき、カーネル＃１は、切替後のアプリＩＤも指示内容に含める。

[0084] 指示を受けたハイパーバイザ＃１は、メモリプロテクトコントローラ１０６＃１に２重アクセス形態を登録する。具体的に、ハイパーバイザ＃１は、メモリプロテクトコントローラ１０６＃１のアクセス形態フラグを２重アクセス形態に設定する。また、ハイパーバイザ＃１は、指示されたアプリＩＤと一致する格納レジスタ４０５＃１の管理番号を、プロテクト領域管理番号フィールドに設定する。このような動作により、マルチコアプロセッサシステム１００は、ＯＳ＃１にてアプリＡ、アプリＢ、…、と切り替えられる動

作と同期して、メモリプロテクトコントローラ１０６＃１にて２重アクセス形態、アクセススルー形態、…、と動作する。

[0085] なお、この動作により、アプリＡが実行中の場合、マルチコアプロセッサシステム１００は、０ｘ０５０００００００および０ｘ０１０００００００の２つのアドレス空間にアクセスを行うことになる。２重アクセスによって、バスアクセスのパフォーマンスが劣化する可能性があるが、一般的に、６４［ビット］系のＣＰＵに対し、１２８［ビット］帯域のバス設定が行われるなど、ＣＰＵアクセスの帯域はバスの帯域を全て使用しない粗の状態である。したがって、バス１０５内にて、２つの６４［ビット］のアクセスを１つの１２８［ビット］にまとめるデータパッキング機能と連動することにより、バスアクセスのパフォーマンスの劣化を避けることができる。

[0086] 同様に、ＣＰＵ＃２は、アプリＣ、アプリＤ、アプリＥ、アプリＣ、…、の順でアプリを実行する。たとえば、ＣＰＵ＃２がアプリＤを実行するとき、アプリＤは高信頼ホワイトリスト３０１に登録されていないため、カーネル＃２は、アプリ切替イベントが発生し、アクセス形態をアクセススルー形態に設定するようにハイパーバイザ＃２に指示する。このとき、カーネル＃２は、切替後のアプリＩＤも指示内容に含める。指示を受けたハイパーバイザ＃２は、メモリプロテクトコントローラ１０６＃２にアクセススルー形態を登録する。具体的に、ハイパーバイザ＃２は、メモリプロテクトコントローラ１０６＃２のアクセス形態フラグをアクセススルー形態に設定する。

[0087] 次に、図５で説明した動作を行うフローチャートを図６～図９にて説明する。なお、図６で示す起動時の処理手順の一例を実行する実行主体は、ＣＰＵ＃１～ＣＰＵ＃ｘのうちいずれのＣＰＵであってもよいが、説明の簡略化のため、ＣＰＵ＃１が実行する場合を想定して説明を行う。また、図７～図９で示すアプリ切替時の処理手順とエラー時の処理手順を実行する実行主体は、ＣＰＵ＃１～ＣＰＵ＃ｘの全てのＣＰＵで実行されるが、説明の簡略化のため、ＣＰＵ＃１が実行する場合を想定して説明を行う。

[0088] 図６は、マルチコアプロセッサシステム１００の起動時の処理手順の一例

を示すフローチャートである。カーネル＃１は、ブート開始する（ステップＳ６０１）と、メモリ管理機構を起動する（ステップＳ６０２）。カーネル＃１は、メモリ管理機構により、Ｎ×Ｍ［バイト］のアドレス空間を確保する（ステップＳ６０３）。

[0089] 確保後、カーネル＃１は、確保されたアドレス空間をハイパーバイザ＃１に通知し（ステップＳ６０４）、通常運用を行い（ステップＳ６０５）、起動時の処理を終了する。なお、通常運用としては、ユーザ領域１２１の確保を行ったり、ブートの初めに起動されるアプリを起動したりする。アドレス空間の通知を受けたハイパーバイザ＃１は、通知されたアドレス空間をプロテクト領域１２４として設定し（ステップＳ６０６）、起動時の処理を終了する。これにより、マルチコアプロセッサシステム１００は、高信頼アプリのメモリ保護領域を確保することができる。

[0090] 図７、図８では、マルチコアプロセッサシステム１００のアプリ切替時の処理手順の一例を示す。図７では、カーネル＃１の処理手順の一例を示し、図８では、ハイパーバイザ＃１の処理手順の一例を示す。

[0091] 図７は、マルチコアプロセッサシステム１００のアプリ切替時の処理手順の一例を示すフローチャート（その１）である。カーネル＃１は、イベントが発生したか否かを判断する（ステップＳ７０１）。イベントが発生していない場合（ステップＳ７０１：イベント発生なし）、カーネル＃１は、一定時間後に、ステップＳ７０１を再度実行する。なお、カーネル＃１は、一定時間にカーネル＃１の他の処理を実行してもよい。

[0092] アプリ起動イベントが発生した場合（ステップＳ７０１：アプリ起動イベント）、カーネル＃１は、起動したアプリが高信頼ホワイトリスト３０１に登録されているか否かを判断する（ステップＳ７０２）。登録されている場合（ステップＳ７０２：Ｙｅｓ）、カーネル＃１は、ハイパーバイザ＃１に管理番号の追加指示を通知する（ステップＳ７０４）。なお、ステップＳ７０４の処理にて、カーネル＃１は、起動したアプリのアプリＩＤを指示内容に含める。

[0093] 通知後、カーネル＃１は、通常時の処理を実行し（ステップＳ７０３）、ステップＳ７０１の処理に移行する。登録されていない場合（ステップＳ７０２：Ｎｏ）、カーネル＃１は、ステップＳ７０３の処理に移行する。なお、アプリ起動イベントの通常時の処理としては、たとえば、起動したアプリのコンテキストを格納する領域をユーザ領域１２１内に確保する処理等である。

[0094] アプリ終了イベントが発生した場合（ステップＳ７０１：アプリ終了イベント）、カーネル＃１は、終了したアプリが高信頼ホワイトリスト３０１に登録されているか否かを判断する（ステップＳ７０５）。登録されている場合（ステップＳ７０５：Ｙｅｓ）、カーネル＃１は、ハイパーバイザ＃１に管理番号の解放指示を通知する（ステップＳ７０６）。なお、ステップＳ７０６の処理にて、カーネル＃１は、終了したアプリのアプリＩＤを指示内容に含める。

[0095] 通知後、または、登録されていない場合（ステップＳ７０５：Ｎｏ）、カーネル＃１は、ステップＳ７０３の処理に移行する。なお、アプリ終了イベントの通常時の処理としては、たとえば、ユーザ領域１２１内に確保されている、終了したアプリのコンテキストを格納する領域を解放する処理等である。

[0096] アプリ切替イベントが発生した場合（ステップＳ７０１：アプリ切替イベント）、カーネル＃１は、切替後のアプリが高信頼ホワイトリスト３０１に登録されているか否かを判断する（ステップＳ７０７）。登録されている場合（ステップＳ７０７：Ｙｅｓ）、カーネル＃１は、ハイパーバイザ＃１に２重アクセス形態指示を通知する（ステップＳ７０８）。なお、ステップＳ７０８の処理にて、カーネル＃１は、切替後のアプリのアプリＩＤを指示内容に含める。通知後、カーネル＃１は、ステップＳ７０３の処理に移行する。登録されていない場合（ステップＳ７０７：Ｎｏ）、カーネル＃１は、ハイパーバイザ＃１にアクセススルー形態指示を通知する（ステップＳ７０９）。通知後、カーネル＃１は、ステップＳ７０３の処理に移行する。

[0097] なお、アプリ切替イベントの通常時の処理としては、たとえば、ディスパッチ処理がある。ディスパッチ処理とは、ＣＰＵのレジスタ等を切替前のアプリのコンテキストを格納する領域に退避し、切替後のアプリのコンテキストを格納する領域に退避されていたレジスタの値をＣＰＵのレジスタに設定する。

[0098] これにより、マルチコアプロセッサシステム１００は、アプリの起動、終了、切替といったイベントに対して、メモリプロテクトコントローラ１０６への設定変更を指示することができる。

[0099] 図８は、マルチコアプロセッサシステム１００のアプリ切替時の処理手順の一例を示すフローチャート（その２）である。なお、図８では、ハイパーバイザ＃１がメモリプロテクトコントローラ１０６＃１内の制御レジスタ４０４＃１、格納レジスタ４０５＃１を設定することを想定している。したがって、各ステップの説明において、説明の簡略化のため、設定先のレジスタがメモリプロテクトコントローラ１０６＃１内のレジスタであるという記述を省略する。

[0100] ハイパーバイザ＃１は、指示内容を確認する（ステップＳ８０１）。指示内容が管理番号の追加指示である場合（ステップＳ８０１：管理番号の追加指示）、ハイパーバイザ＃１は、格納レジスタ４０５＃１＿１～格納レジスタ４０５＃１＿ｙのうち、未使用の格納レジスタ４０５＃１が存在するか否かを判断する（ステップＳ８０２）。

[0101] 未使用の格納レジスタ４０５＃１が存在する場合（ステップＳ８０２：Ｙｅｓ）、ハイパーバイザ＃１は、発見された格納レジスタ４０５＃１の使用中ビットフィールドを使用中に設定する（ステップＳ８０３）。続けて、ハイパーバイザ＃１は、発見された格納レジスタ４０５＃１のアプリＩＤフィールドを指示内容に含まれていたアプリＩＤに設定する（ステップＳ８０４）。さらに、ハイパーバイザ＃１は、発見された格納レジスタ４０５＃１のプロテクトアドレスフィールドに未割当のアドレスを設定し（ステップＳ８０５）、ステップＳ８０１の処理に移行する。格納レジスタ４０５＃１が存

在しない場合（ステップＳ８０２：Ｎｏ）、ハイパーバイザ＃１は、ステップＳ８０１の処理に移行する。

[0102] なお、ステップＳ８０５の処理での未割当のアドレスとは、プロテクト領域１２４内で、他の高信頼アプリのメモリ領域として割り当てられていないメモリ領域のアドレスである。また、ステップＳ８０３～ステップＳ８０５の処理にて、マスクアドレスフィールドの記述がないが、本実施の形態におけるマスクアドレスは常に固定値であることを想定している。したがって、マスクアドレスフィールドの設定箇所として、たとえば、前述のステップＳ６０６にて、ハイパーバイザ＃１は、格納レジスタ４０５＃１＿１～格納レジスタ４０５＃１＿ｙのマスクアドレスフィールドに０ｘ０００００ｆｆｆｆを設定する。

[0103] 指示内容が管理番号の解放指示である場合（ステップＳ８０１：管理番号の解放指示）、ハイパーバイザ＃１は、格納レジスタ４０５＃１＿１～格納レジスタ４０５＃１＿ｙのうち、アプリＩＤフィールドが指示内容に含まれていたアプリＩＤと一致する格納レジスタ４０５＃１を選択する（ステップＳ８０６）。選択後、ハイパーバイザ＃１は、選択された格納レジスタ４０５＃１のプロテクトアドレスフィールドの値を未割当のアドレスに設定する（ステップＳ８０７）。続けて、ハイパーバイザ＃１は、選択された格納レジスタ４０５＃１の使用中ビットフィールドを未使用に設定し（ステップＳ８０８）、ステップＳ８０１の処理に移行する。

[0104] 指示内容が２重アクセス形態指示である場合（ステップＳ８０１：２重アクセス形態指示）、ハイパーバイザ＃１は、アプリＩＤフィールドが指示内容に含まれていたアプリＩＤと一致する格納レジスタ４０５＃１の管理番号を、プロテクト管理番号フィールドに設定する（ステップＳ８０９）。続けて、ハイパーバイザ＃１は、アクセス形態フラグを２重アクセス形態に設定し（ステップＳ８１０）、ステップＳ８０１の処理に移行する。

[0105] 指示内容がアクセススルー形態指示である場合（ステップＳ８０１：アクセススルー形態指示）、ハイパーバイザ＃１は、アクセス形態フラグをアク

セススルー形態に設定し（ステップＳ８１１）、ステップＳ８０１の処理に移行する。このように、マルチコアプロセッサシステム１００は、アプリの起動、終了、切替といったイベントに対応してメモリプロテクトコントローラ１０６への設定を変更するため、アプリの各イベントに同期してアクセス形態を変更することができる。

[0106] 続けて、図９では、エラー時の処理手順の一例を説明する。図９で示すフローチャートでは、ＣＰＵ＃２で実行している低信頼アプリの動作検証が不十分であったため、低信頼アプリが他のメモリ領域へ書き込んだ結果、障害が発生した場合を想定している。不具合によって引き起こされる状態には２種類ある。

[0107] １つ目の状態は、単純なプログラムエラーで高信頼アプリが終了する状態である。たとえば、ＯＳ＃１で実行している高信頼アプリに対してエラーが発生した場合、ＯＳ＃１はエラーの発生を検出し、ハイパーバイザ＃１に通知することが可能である。２つ目の状態は、ＯＳ＃１がハングする場合である。ＯＳ＃１がハングすると高信頼アプリの障害状態を検出できない。したがって、ＯＳ＃１と独立で動作しているハイパーバイザ＃１が、周期監視によりＯＳ＃１がハングしているかどうかを検出する。

[0108] 図９は、マルチコアプロセッサシステム１００のエラー時の処理手順の一例を示すフローチャートである。カーネル＃２は、低信頼アプリを実行する（ステップＳ９０１）。この実行により、ユーザ領域１２１に不正書込が行われ、障害が発生した場合を想定する。

[0109] 不正書込により、アプリが強制終了するなどの軽微な障害が発生した場合、カーネル＃１は、セグメントエラーを検出する（ステップＳ９０２）。検出後、カーネル＃１は、マルチコアプロセッサシステム１００を一時停止し（ステップＳ９０３）、ハイパーバイザ＃１へリカバリ通知を指示する（ステップＳ９０４）。

[0110] ＯＳハング等の重大な障害が発生し、周期監視より検出した場合、または、リカバリ通知を指示された場合、ハイパーバイザ＃１は、実行中アプリが

高信頼ホワイトリスト３０１に登録されているか否かを判断する（ステップＳ９０５）。登録されている場合（ステップＳ９０５：Ｙｅｓ）、ハイパーバイザ＃１は、プロテクト領域１２４内の実行中アプリのメモリ領域とユーザ領域１２１内の実行中アプリのメモリ領域のデータを比較する（ステップＳ９０６）。

[0111] 続けて、ハイパーバイザ＃１は、比較結果が一致したか否かを判断する（ステップＳ９０７）。一致しない場合（ステップＳ９０７：Ｎｏ）、ハイパーバイザ＃１は、ユーザ領域１２１内の実行中アプリのデータを復元する（ステップＳ９０８）。具体的に、ハイパーバイザ＃１は、比較結果の差分をユーザ領域１２１内の実行中アプリのメモリ領域に上書きする。これにより、不正書込が行われたデータが正常なデータに復元することになる。

[0112] 高信頼ホワイトリスト３０１に登録されていない場合（ステップＳ９０５：Ｎｏ）、比較結果が一致した場合（ステップＳ９０７：Ｙｅｓ）、または、データ復元後、ハイパーバイザ＃１は、ＯＳ＃１がハング中か否かを判断する（ステップＳ９０９）。ハング中である場合（ステップＳ９０９：Ｙｅｓ）、ハイパーバイザ＃１は、ＯＳ＃１のウォームスタートを実行し（ステップＳ９１０）、エラー時の処理を終了する。具体的なウォームスタートの方法としては、チェックポイントリスタート技術や、ハイバネーション技術を適用することができる。

[0113] ハング中でない場合（ステップＳ９０９：Ｎｏ）、ハイパーバイザ＃１は、カーネル＃１にマルチコアプロセッサシステム１００の一時停止解除を通知し（ステップＳ９１１）、エラー時の処理を終了する。一時停止解除を受け付けたカーネル＃１は、マルチコアプロセッサシステム１００の一時停止を解除し（ステップＳ９１２）、エラー時の処理を終了する。このように、マルチコアプロセッサシステム１００は、不正書込によるメモリ破壊が原因で高信頼アプリにエラーが発生した場合、データを復元し動作を続行することができる。

[0114] 以上説明したように、マルチコアプロセッサシステムによれば、高信頼ア

プリがアクセスする第１メモリ領域とは異なる第２メモリ領域を確保し、同一の内容を書き込む。これにより、マルチコアプロセッサシステムは、低信頼アプリが第１メモリ領域を破壊し障害が発生しても第２メモリ領域を使用してリカバリできるため、高信頼アプリと低信頼アプリを混載して実行できる。

[0115] また、本実施の形態にかかるマルチコアプロセッサシステムが実行するソフトウェア保護方法は、シングルコアプロセッサシステムでのソフトウェア資産を、安全にマルチコアプロセッサシステム環境に継承することができる。また、本実施の形態にかかるマルチコアプロセッサシステムは、通常動作状態において、たとえば、高信頼アプリ実行中に低信頼アプリを実行させないなどといった、バイアスのかかった不利なスケジューリングが発生することなく運用することができる。また、本実施の形態にかかるマルチコアプロセッサシステムは、障害が発生した場合にリカバリ処理が発生し、通常動作状態では負荷のかかる処理を実行しないため、全体のパフォーマンスが劣化することはない。

[0116] また、マルチコアプロセッサシステムは、所定のアプリを格納するリストにアクセス可能であり、実行されるアプリとリストに格納されたアプリとを比較してもよい。これにより、マルチコアプロセッサシステムは、実行されるアプリが高信頼アプリか否かを判断することができる。

[0117] また、マルチコアプロセッサシステムは、実行されるアプリとリストに格納されたアプリが一致する場合、第１メモリ領域と第２メモリ領域とにアクセスするようにアプリを実行するＣＰＵに対応するメモリプロテクトコントローラに登録してもよい。これにより、マルチコアプロセッサシステムは、高信頼アプリのデータを保護することができる。

[0118] また、マルチコアプロセッサシステムは、実行されるアプリとリストに格納されたアプリが一致しない場合、第１メモリ領域にアクセスするようにアプリを実行するＣＰＵに対応するメモリプロテクトコントローラに登録してもよい。これにより、マルチコアプロセッサシステムは、データの保護を行

わなくてよい低信頼アプリについてはデータを保護せず、データ保護用に確保したプロテクト領域を有効に使用することができる。

[0119] また、マルチコアプロセッサシステムは、自システムの異常検出に応答して、第１メモリ領域と第２メモリ領域との比較結果に基づいて第１メモリ領域を復元してもよい。これにより、マルチコアプロセッサシステムは、障害があったアプリをリカバリすることができる。

[0120] また、マルチコアプロセッサシステムは、マルチコアプロセッサシステム環境での動作検証が行われたアプリを高信頼アプリとして保護してもよい。低信頼のアプリは、第１メモリ領域を自ら破壊してしまう場合があり、このとき、メモリプロテクトコントローラによって第２メモリ領域にデータを保護していても、第２メモリ領域も破壊する結果となる。この状態で、第１メモリ領域のデータを復元しても、破壊されたデータで上書きされてしまい、マルチコアプロセッサシステムは、リカバリすることができない。したがって、マルチコアプロセッサシステムは、不具合がないと検証された高信頼アプリを対象とすることで、低信頼アプリの動作によって発生した高信頼アプリの障害をリカバリすることができる。

[0121] なお、本実施の形態で説明したマルチコアプロセッサシステムが実行するソフトウェア保護方法では、対象がアプリである場合に対して説明していたが、対象がライブラリであっても本実施の形態を適用することができる。

[0122] なお、本実施の形態で説明したマルチコアプロセッサシステムが実行するソフトウェア保護方法は、予め用意されたプログラムをパーソナル・コンピュータやワークステーション等のコンピュータで実行することにより実現することができる。また、ソフトウェア保護方法を実行するソフトウェア保護プログラムは、ハードディスク、ＣＤ－ＲＯＭ、ＤＶＤ、メモリカード等のコンピュータで読み取り可能な記録媒体に記録され、コンピュータによって記録媒体から読み出されることによって実行される。またソフトウェア保護プログラムは、インターネット等のネットワークを介して配布してもよい。

[0123] また、本実施の形態で説明したメモリプロテクトコントローラ１０６は、

スタンダードセルやストラクチャードＡＳＩＣ（Ａｐｐｌｉｃａｔｉｏｎ Ｓｐｅｃｉｆｉｃ Ｉｎｔｅｇｒａｔｅｄ Ｃｉｒｃｕｉｔ）などの特定用途向けＩＣ（以下、単に「ＡＳＩＣ」と称す。）やＦＰＧＡなどのＰＬＤ（Ｐｒｏｇｒａｍｍａｂｌｅ Ｌｏｇｉｃ Ｄｅｖｉｃｅ）によっても実現することができる。具体的には、たとえば、上述したメモリプロテクトコントローラ１０６の機能（記憶部４０１～変換部４０３）をＨＤＬ記述によって機能定義し、そのＨＤＬ記述を論理合成してＡＳＩＣやＰＬＤに与えることにより、メモリプロテクトコントローラ１０６を製造することができる。

符号の説明

[0124] ＃１、＃２ ＣＰＵ

１００ マルチコアプロセッサシステム

１０４ メモリ

１０５ バス

１０６ メモリプロテクトコントローラ

１１１ 低信頼アプリ

１１２ 高信頼アプリ

１２１ ユーザ領域

１２２ 低信頼アプリメモリ領域

１２３ 高信頼アプリメモリ領域

１２４ プロテクト領域

１２５ 高信頼アプリメモリ領域

３０１ 高信頼ホワイトリスト

３１１ 確保部

３１２ 検出部

３１３ 比較部

３１４ 通知部

３１５ 登録部

３１６ 検出部

３１７　検出部

３１８　比較部

３１９　復元部

請求の範囲

[請求項1] 複数のＣＰＵと、

メモリと、

前記複数のＣＰＵと前記メモリとの間に配置されるメモリプロテクトコントローラと、

を含み、

前記メモリプロテクトコントローラは、

アプリケーション実行時に前記複数のＣＰＵのアクセス要求によって第１メモリ領域にアクセスするとともに、システムブート時に確保された第２メモリ領域にアクセスすること

を特徴とするマルチコアプロセッサシステム。

[請求項2] 所定のアプリケーションのリストを格納するテーブルと、

実行されるアプリケーションと前記リストのアプリケーションとを比較する比較部と、

を含むことを特徴とする請求項１に記載のマルチコアプロセッサシステム。

[請求項3] 比較結果が一致を示すとき、前記実行されるアプリケーションについて、前記第１メモリ領域と前記第２メモリ領域とをアクセスする指示を登録すること

を特徴とする請求項２に記載のマルチコアプロセッサシステム。

[請求項4] 比較結果が不一致を示すとき、前記実行されるアプリケーションについて、前記第１メモリ領域のみをアクセスする指示を登録すること

を特徴とする請求項２に記載のマルチコアプロセッサシステム。

[請求項5] 自システムの異常の検出に応答して、前記実行されるアプリケーションが実行中であるときに前記第１メモリ領域と前記第２メモリ領域との比較結果に基づいて前記第１メモリ領域を復元すること

を特徴とする請求項１乃至請求項４の何れか一に記載のマルチコアプロセッサシステム。

[請求項6] 前記所定のアプリケーションは、マルチコアプロセッサシステム環境での動作検証が行われたアプリケーションであること

を特徴とする請求項１乃至請求項５の何れか一に記載のマルチコアプロセッサシステム。

[請求項7] 前記複数のＣＰＵの数がＮ（Ｎは２以上の整数）であり、アプリケーション実行時に割り当てられる前記メモリのメモリ領域がＭ（Ｍは１以上の整数）バイトであるとき、前記第２メモリ領域の大きさはＮ×Ｍであること

を特徴とする請求項１乃至請求項６の何れか一に記載のマルチコアプロセッサシステム。

[請求項8] 複数のＣＰＵと、

メモリと、

前記複数のＣＰＵと前記メモリとの間に配置されるメモリプロテクトコントローラと、

を含み、

前記メモリプロテクトコントローラは、

アクセス形態情報が第１アクセス形態を示す場合には、アプリケーション実行において第１メモリ領域へのアクセスと同様に第２メモリ領域をアクセスし、

アクセス形態情報が第２アクセス形態を示す場合には、アプリケーション実行において前記第１メモリ領域のみをアクセスすること

を特徴とするマルチコアプロセッサシステム。

[請求項9] 前記アプリケーションとリストに登録されるアプリケーションとの比較結果に基づいて前記アクセス形態情報が設定されること

を特徴とする請求項８に記載のマルチコアプロセッサシステム。

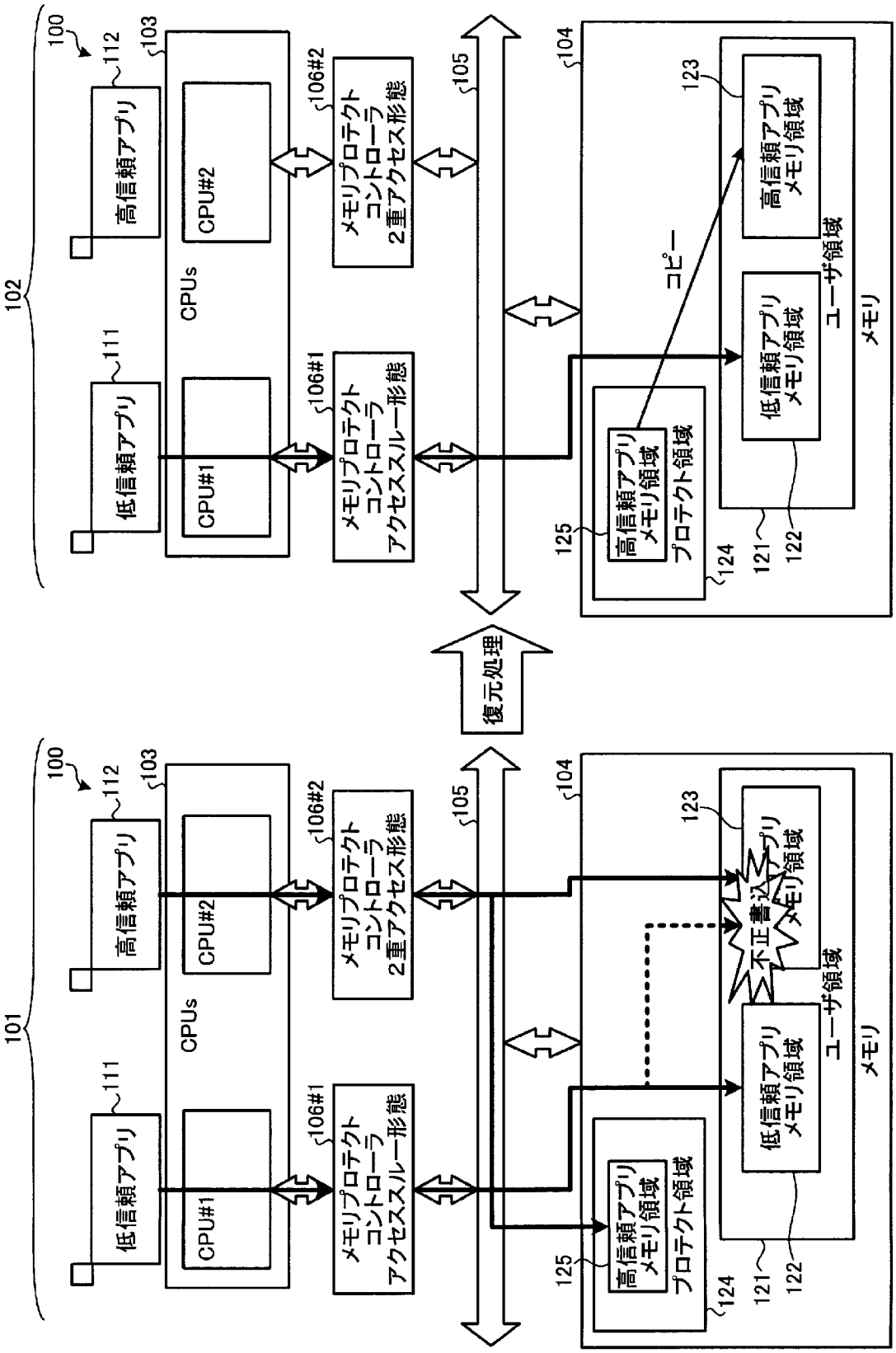
[請求項10] 前記比較結果が一致を示すとき、前記アクセス形態情報が第１アクセス形態に設定され、

前記比較結果が不一致を示すとき、前記アクセス形態情報が第２ア

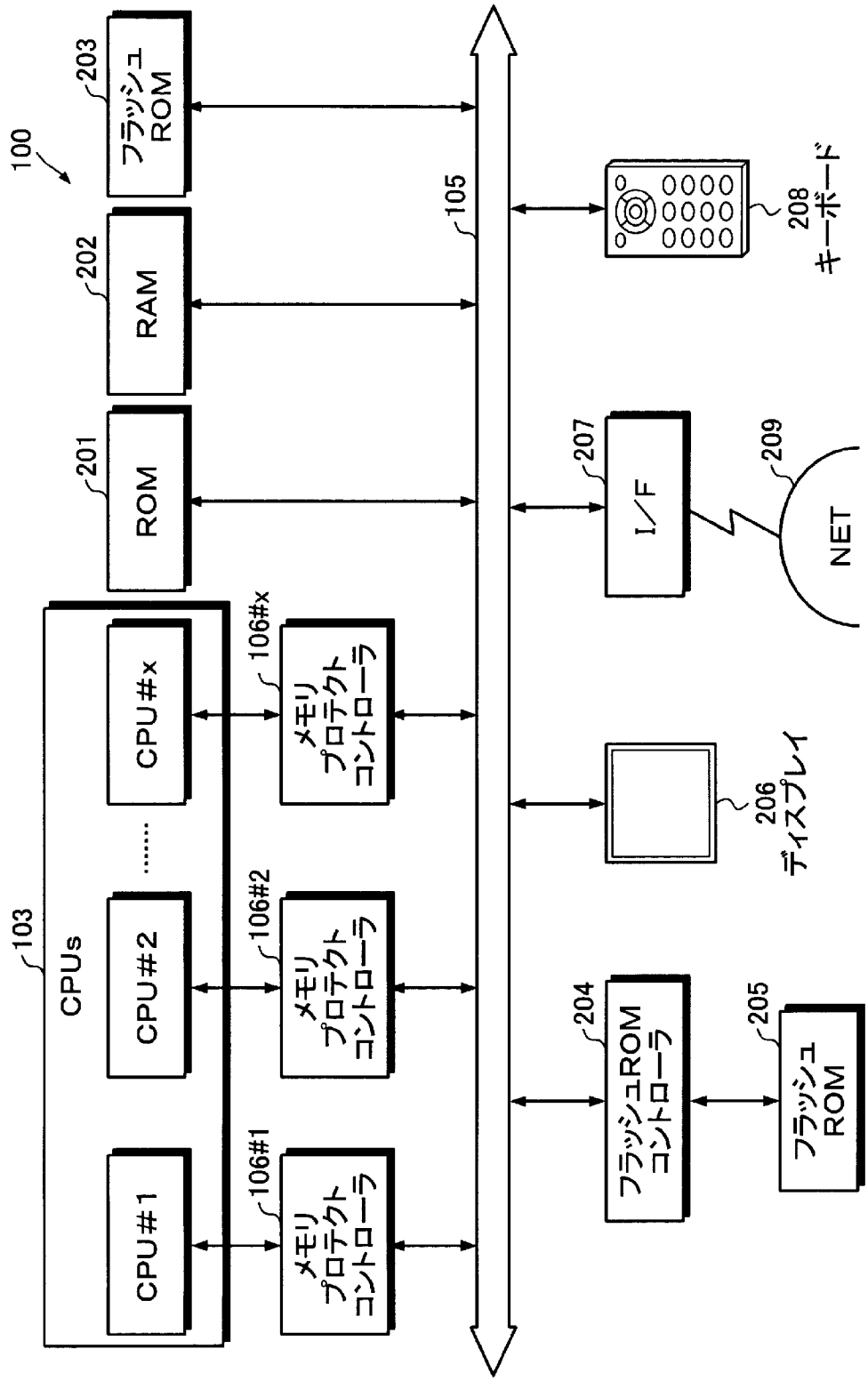
クセス形態に設定されること

を特徴とする請求項９に記載のマルチコアプロセッサシステム。

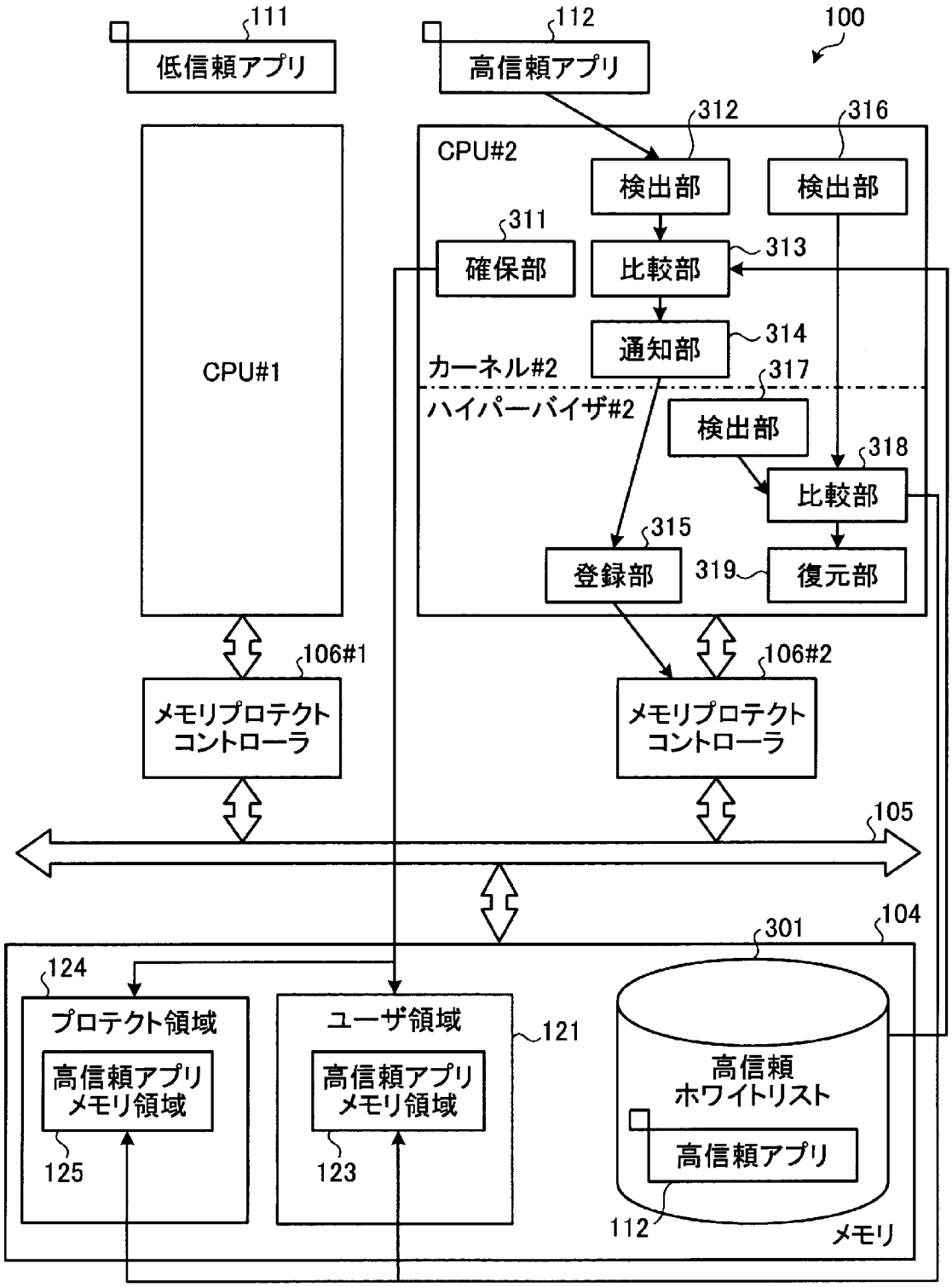
[図1]


100
101
102
103
104
105
106#1
106#2
111
112
121
122
123
124
125
低信頼アプリ
高信頼アプリ
CPU#1
CPU#2
CPUs
メモリプロテクト
コントローラ
アクセススルー形態
メモリプロテクト
コントローラ
2重アクセス形態
メモリ
ユーザ領域
低信頼アプリ
メモリ領域
高信頼アプリ
メモリ領域
高信頼アプリ
メモリ領域
プロテクト領域
コピー
不正書込
復元処理

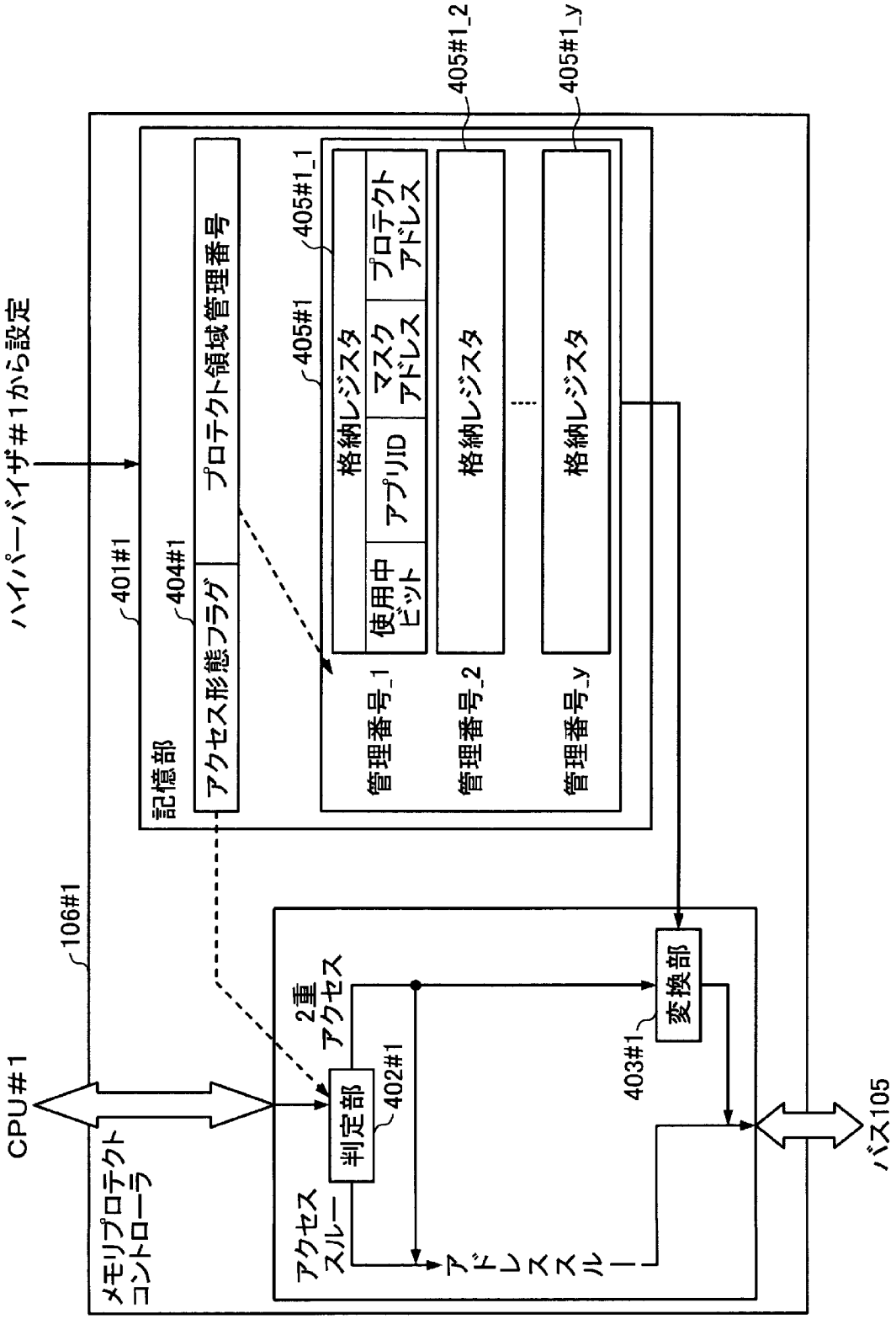
[図2]


100
203
フラッシュ ROM
202
RAM
201
ROM
105
208
キーボード
207
I/F
209
NET
206
ディスプレイ
204
フラッシュROM コントローラ
205
フラッシュ ROM
103
CPUs
CPU#1
CPU#2
CPU#x
106#1
106#2
106#x
メモリ プロテクト コントローラ

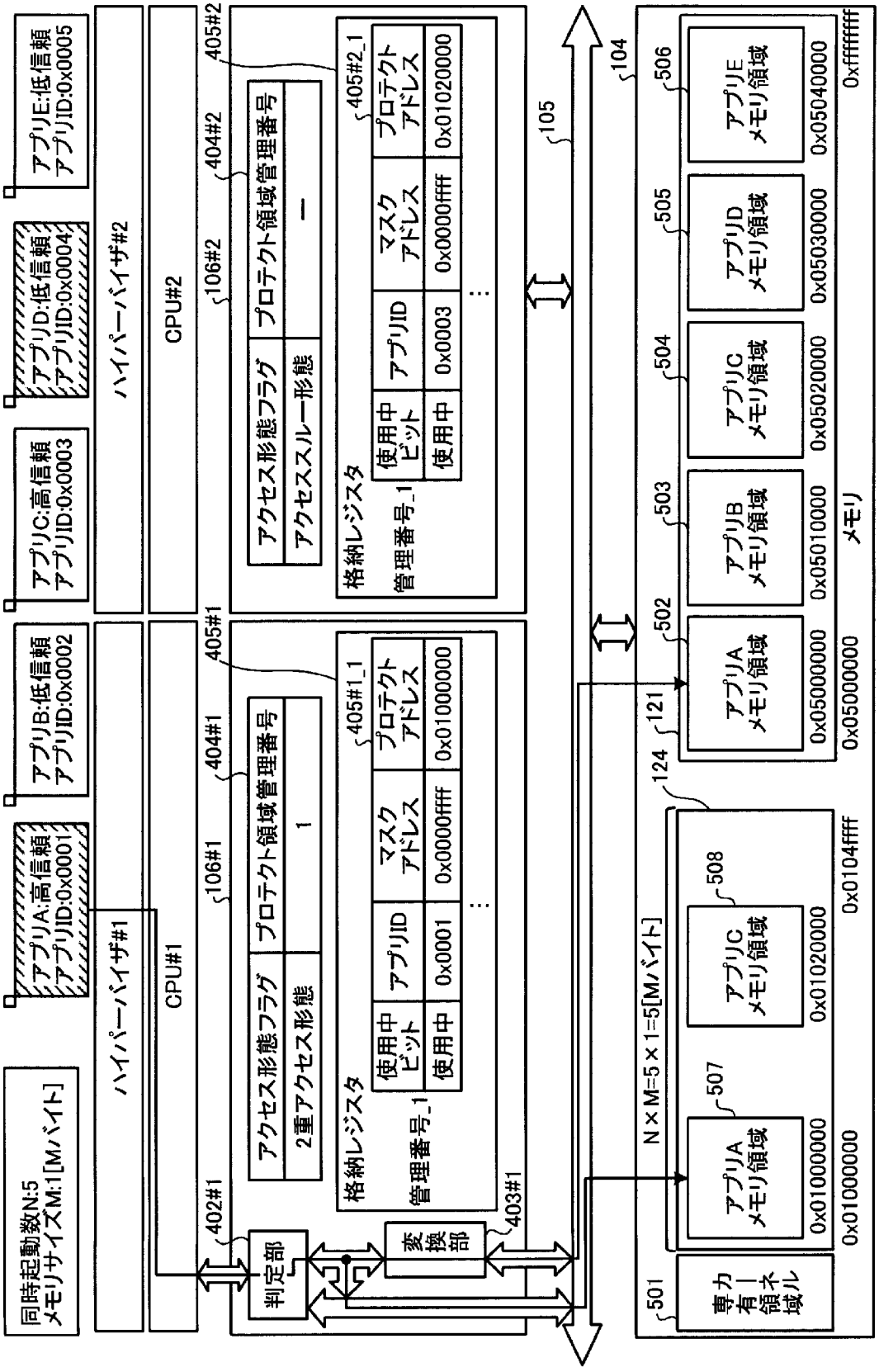
[図3]


111
低信頼アプリ
112
高信頼アプリ
100
312
316
CPU#2
検出部
検出部
311
確保部
比較部
313
通知部
314
CPU#1
カーネル#2
317
ハイパーバイザ#2
検出部
318
比較部
315
登録部
319
復元部
106#1
106#2
メモリプロテクト
コントローラ
メモリプロテクト
コントローラ
105
301
104
124
プロテクト領域
高信頼アプリ
メモリ領域
125
ユーザ領域
121
高信頼アプリ
メモリ領域
123
高信頼
ホワイトリスト
高信頼アプリ
112
メモリ

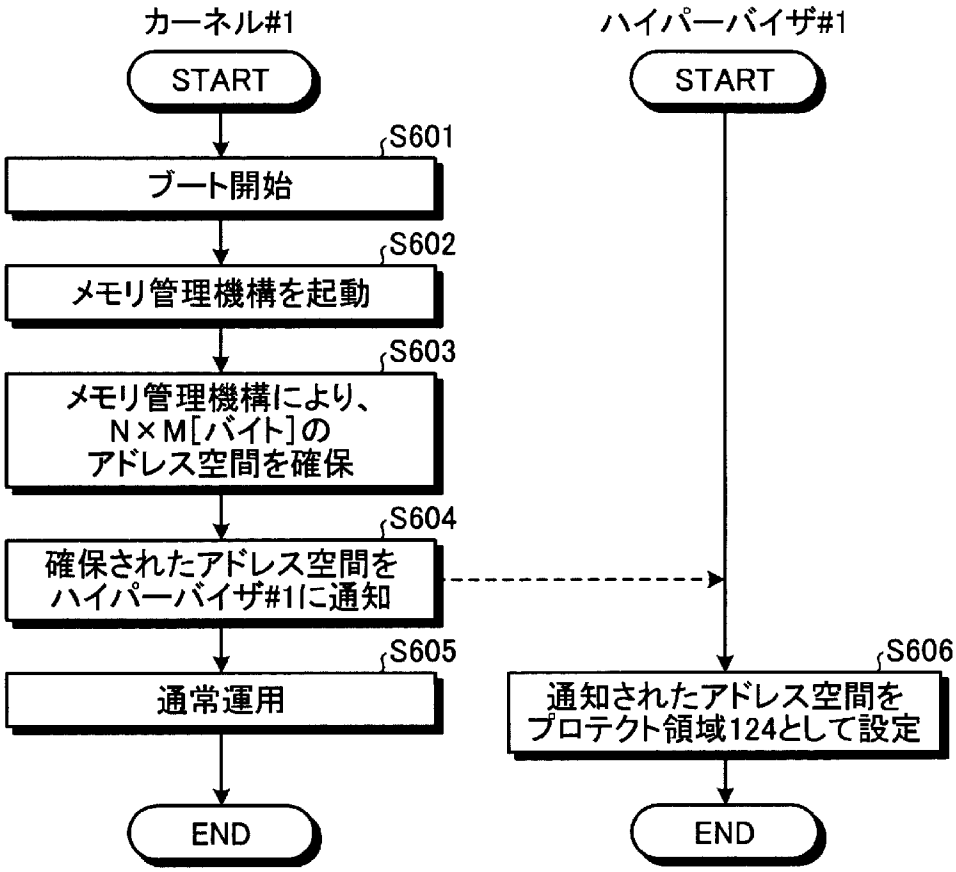
[図4]


ハイパーバイザ#1から設定
405#1_2
405#1_y
405#1_1
405#1
プロテクトアドレス
マスクアドレス
アプリID
使用中ビット
格納レジスタ
格納レジスタ
格納レジスタ
管理番号_1
管理番号_2
管理番号_y
401#1
404#1
プロテクト領域管理番号
アクセス形態フラグ
記憶部
106#1
CPU#1
メモリプロテクトコントローラ
2重アクセス
判定部
402#1
403#1
変換部
アクセススルー
アドレススルー
バス105

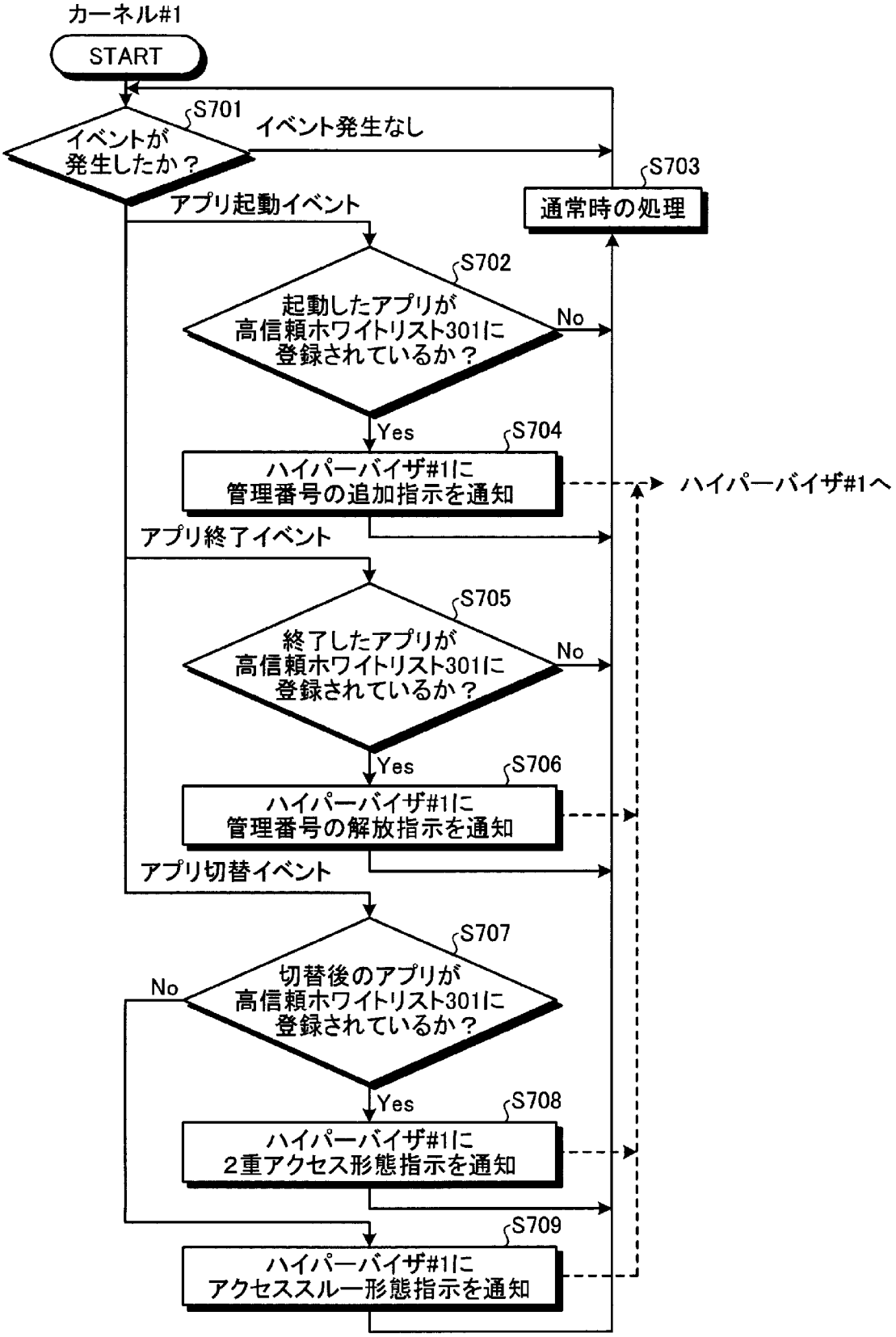
[図5]


同時起動数N:5
メモリサイズM:1[Mバイト]
アプリA:高信頼
アプリID:0x0001
アプリB:低信頼
アプリID:0x0002
アプリC:高信頼
アプリID:0x0003
アプリD:低信頼
アプリID:0x0004
アプリE:低信頼
アプリID:0x0005
ハイパーバイザ#1
ハイパーバイザ#2
CPU#1
CPU#2
402#1
403#1
404#1
404#2
405#1
405#2
106#1
106#2
判定部
変換部
アクセス形態フラグ
2重アクセス形態
プロテクト領域管理番号
1
アクセス形態フラグ
アクセスルー形態
プロテクト領域管理番号
ー
格納レジスタ
管理番号_1
405#1_1
使用中ビット
使用中
アプリID
0x0001
マスクアドレス
0x0000ffff
プロテクトアドレス
0x01000000
405#2_1
使用中ビット
使用中
アプリID
0x0003
マスクアドレス
0x0000ffff
プロテクトアドレス
0x01020000
105
104
N×M=5×1=5[Mバイト]
124
121
501
502
503
504
505
506
507
508
専有領域
アプリAメモリ領域
アプリCメモリ領域
アプリAメモリ領域
アプリBメモリ領域
アプリCメモリ領域
アプリDメモリ領域
アプリEメモリ領域
0x01000000
0x01000000
0x01020000
0x01104ffff
0x05000000
0x05000000
0x05010000
0x05020000
0x05030000
0x05040000
0xffffffff
メモリ

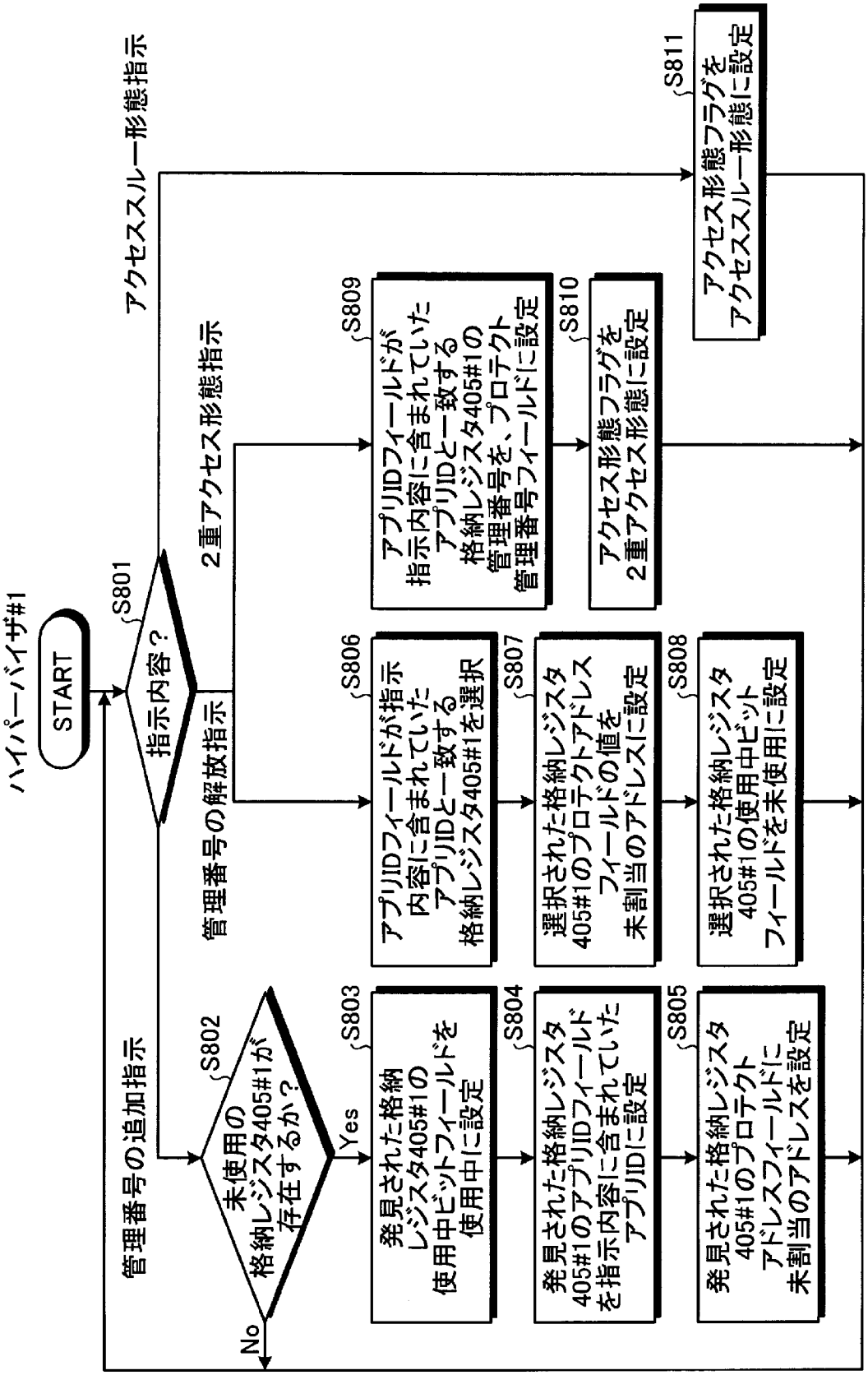
[図6]


カーネル#1
START
S601
ブート開始
S602
メモリ管理機構を起動
S603
メモリ管理機構により、
N×M[バイト]の
アドレス空間を確保
S604
確保されたアドレス空間を
ハイパーバイザ#1に通知
S605
通常運用
END
ハイパーバイザ#1
START
S606
通知されたアドレス空間を
プロテクト領域124として設定
END

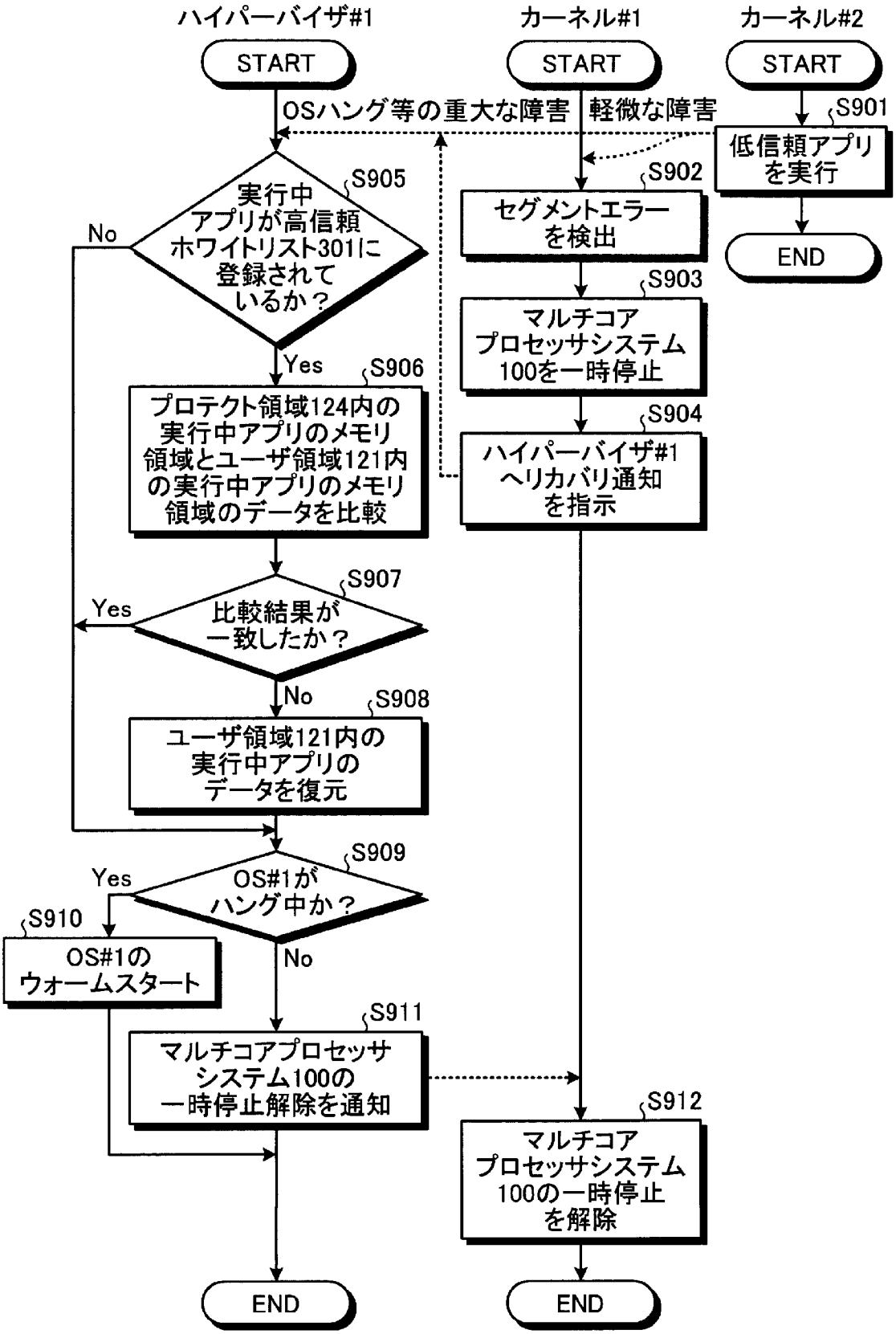
[図7]


カーネル#1
START
S701
イベントが発生したか？
イベント発生なし
S703
通常時の処理
アプリ起動イベント
S702
起動したアプリが高信頼ホワイトリスト301に登録されているか？
No
Yes
S704
ハイパーバイザ#1に管理番号の追加指示を通知
ハイパーバイザ#1へ
アプリ終了イベント
S705
終了したアプリが高信頼ホワイトリスト301に登録されているか？
No
Yes
S706
ハイパーバイザ#1に管理番号の解放指示を通知
アプリ切替イベント
S707
切替後のアプリが高信頼ホワイトリスト301に登録されているか？
No
Yes
S708
ハイパーバイザ#1に2重アクセス形態指示を通知
S709
ハイパーバイザ#1にアクセススルー形態指示を通知

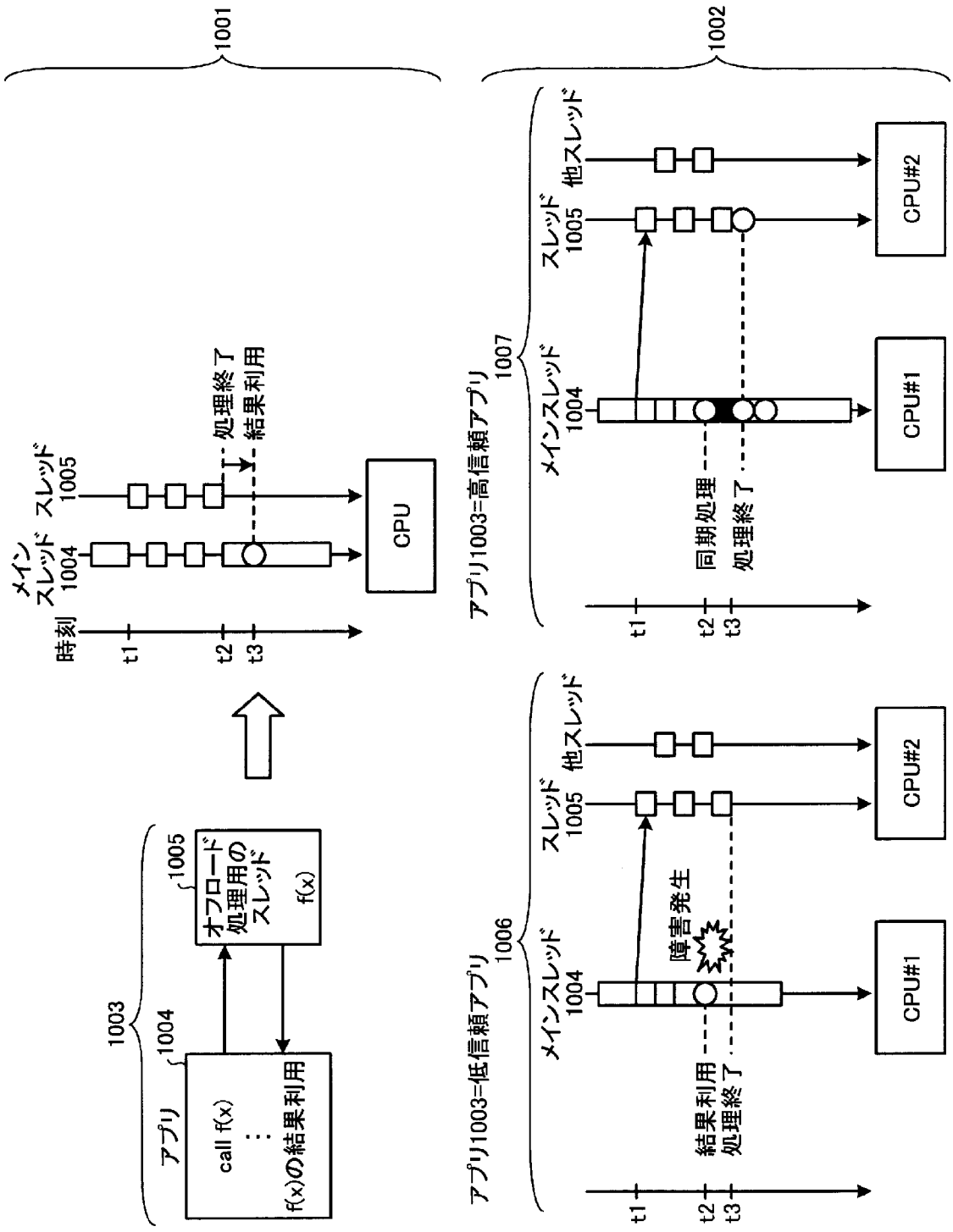
[図8]


ハイパーバイザ#1
START
S801
指示内容？
管理番号の追加指示
S802
未使用の格納レジスタ405#1が存在するか？
Yes
No
S803
発見された格納レジスタ405#1の使用中ビットフィールドを使用中に設定
S804
発見された格納レジスタ405#1のアプリIDフィールドを指示内容に含まれていたアプリIDに設定
S805
発見された格納レジスタ405#1のプロテクトアドレスフィールドに未割当のアドレスを設定
管理番号の解放指示
S806
アプリIDフィールドが指示内容に含まれていたアプリIDと一致する格納レジスタ405#1を選択
S807
選択された格納レジスタ405#1のプロテクトアドレスフィールドの値を未割当のアドレスに設定
S808
選択された格納レジスタ405#1の使用中ビットフィールドを未使用に設定
2重アクセス形態指示
S809
アプリIDフィールドが指示内容に含まれていたアプリIDと一致する格納レジスタ405#1の管理番号を、プロテクト管理番号フィールドに設定
S810
アクセス形態フラグを2重アクセス形態に設定
アクセススルー形態指示
S811
アクセス形態フラグをアクセススルー形態に設定

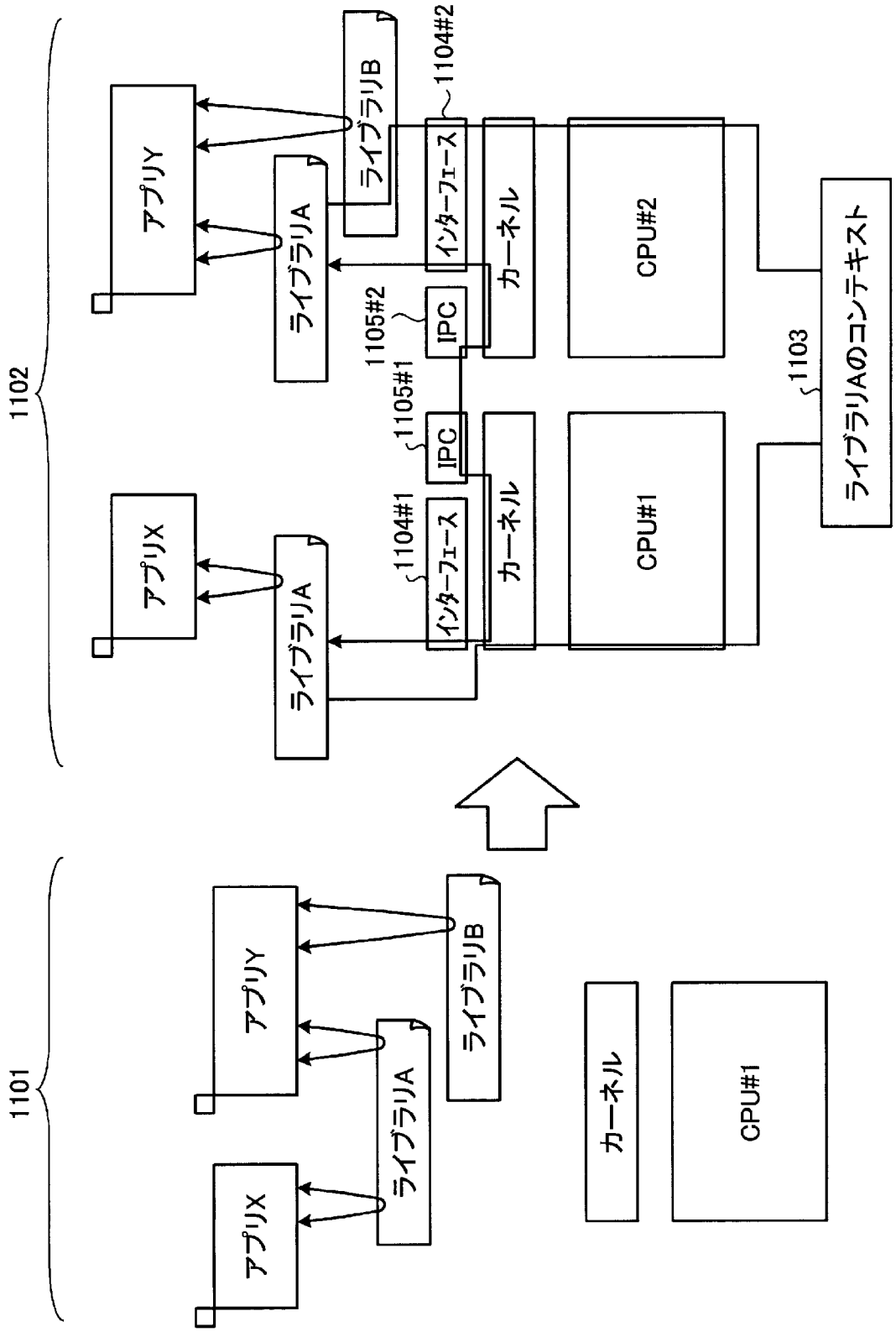
[図9]


ハイパーバイザ#1
カーネル#1
カーネル#2
START
START
START
OSハング等の重大な障害
軽微な障害
S901
低信頼アプリを実行
END
S902
セグメントエラーを検出
S903
マルチコアプロセッサシステム100を一時停止
S904
ハイパーバイザ#1へリカバリ通知を指示
S905
実行中アプリが高信頼ホワイトリスト301に登録されているか？
No
Yes
S906
プロテクト領域124内の実行中アプリのメモリ領域とユーザ領域121内の実行中アプリのメモリ領域のデータを比較
S907
比較結果が一致したか？
Yes
No
S908
ユーザ領域121内の実行中アプリのデータを復元
S909
OS#1がハング中か？
Yes
No
S910
OS#1のウォームスタート
S911
マルチコアプロセッサシステム100の一時停止解除を通知
S912
マルチコアプロセッサシステム100の一時停止を解除
END
END

[図10]


1001
1002
1003
1004
1005
1006
1007
アプリ
call f(x)
:
f(x)の結果利用
オフロード処理用のスレッド
f(x)
メインスレッド
スレッド
他スレッド
CPU
CPU#1
CPU#2
時刻
t1
t2
t3
処理終了
結果利用
同期処理
障害発生
アプリ1003=高信頼アプリ
アプリ1003=低信頼アプリ

[図11]


1101
アプリX
アプリY
ライブラリA
ライブラリB
カーネル
CPU#1
1102
アプリX
アプリY
ライブラリA
ライブラリB
1104#1
インターフェース
1104#2
インターフェース
1105#1
IPC
1105#2
IPC
カーネル
カーネル
CPU#1
CPU#2
1103
ライブラリAのコンテキスト

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2011/057715

A. CLASSIFICATION OF SUBJECT MATTER
G06F21/24(2006.01)i, *G06F12/14*(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)
G06F12/14, 21/00-21/24, 9/46, 9/48, 9/50-9/52, 9/54, 12/08-12/12, 15/16-15/177

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	1996-2011
Kokai Jitsuyo Shinan Koho	1971-2011	Toroku Jitsuyo Shinan Koho	1994-2011

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.
Y	JP 2009-271597 A (Toshiba Corp.), 19 November 2009 (19.11.2009), paragraphs [0012], [0035]; fig. 1 (Family: none)	1-5,7-10
Y	JP 2003-316752 A (NEC Corp.), 07 November 2003 (07.11.2003), paragraphs [0017], [0020], [0039] to [0044]; fig. 1, 3 (Family: none)	1-5,7-10
Y	JP 5-143467 A (Mitsubishi Electric Corp.), 11 June 1993 (11.06.1993), paragraph [0018]; fig. 1 (Family: none)	1-5,7-10

☒ 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 06 April, 2011 (06.04.11)	Date of mailing of the international search report 19 April, 2011 (19.04.11)
Name and mailing address of the ISA/ Japanese Patent Office Facsimile No.	Authorized officer Telephone No.

Form PCT/ISA/210 (second sheet) (July 2009)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2011/057715

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y A	JP 2009-251967 A (Toyota Motor Corp.), 29 October 2009 (29.10.2009), claims 1, 2 (Family: none)	2-4,9,10 6
Y A	JP 2010-118010 A (Nomura Research Institute, Ltd.), 27 May 2010 (27.05.2010), abstract; paragraphs [0027] to [0029] (Family: none)	2-4,9,10 6
Y	JP 5-108493 A (NEC Software Shikoku, Ltd.), 30 April 1993 (30.04.1993), abstract; paragraph [0018] (Family: none)	5

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２０１１／０５７７１５

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F21/24(2006.01)i, G06F12/14(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.　G06F12/14, 21/00-21/24, 9/46, 9/48, 9/50-9/52, 9/54, 12/08-12/12, 15/16-15/177

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	１９７１－２０１１年
日本国実用新案登録公報	１９９６－２０１１年
日本国登録実用新案公報	１９９４－２０１１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
Y	JP 2009-271597 A (株式会社東芝) 2009.11.19, 段落[0012], [0035], 図１ (ファミリーなし)	1-5, 7-10
Y	JP 2003-316752 A (日本電気株式会社) 2003.11.07, 段落[0017], [0020], [0039]-[0044], 図 1, 3 (ファミリーなし)	1-5, 7-10
Y	JP 5-143467 A (三菱電機株式会社) 1993.06.11, 段落[0018], 図１ (ファミリーなし)	1-5, 7-10
Y	JP 2009-251967 A (トヨタ自動車株式会社) 2009.10.29, 請求項 1, 2 (ファミリーなし)	2-4, 9, 10
A		6

☑　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
０６．０４．２０１１	１９．０４．２０１１

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）　５Ｓ　２９５７
戸島　弘詩
電話番号　０３－３５８１－１１０１　内線　３５４６

Ｃ（続き）．　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
Y	JP 2010-118010 A (株式会社野村総合研究所) 2010.05.27, 要約, 段落[0027]-[0029] (ファミリーなし)	2-4, 9, 10
A		6
Y	JP 5-108493 A (四国日本電気ソフトウエア株式会社) 1993.04.30, 要約, 段落[0018] (ファミリーなし)	5