(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
2019年4月4日(04.04.2019)

WIPO | PCT

(10) 国際公開番号
WO 2019/065302 A1

(51) 国際特許分類:
G06F 15/167 (2006.01) *F02D 45/00* (2006.01)

(21) 国際出願番号： PCT/JP2018/034137

(22) 国際出願日： 2018年9月14日(14.09.2018)

(25) 国際出願の言語： 日本語

(26) 国際公開の言語： 日本語

(30) 優先権データ：
特願 2017-185977 2017年9月27日(27.09.2017) JP

(71) 出願人: 日立オートモティブシステムズ株式会社(**HITACHI AUTOMOTIVE SYSTEMS, LTD.**) [JP/JP]; 〒3128503 茨城県ひたちなか市高場２５２０番地 Ibaraki (JP).

(72) 発明者: 三宅 淳司(**MIYAKE Junji**); 〒3128503 茨城県ひたちなか市高場２５２０番地 日立オートモティブシステムズ株式会社内 Ibaraki (JP).

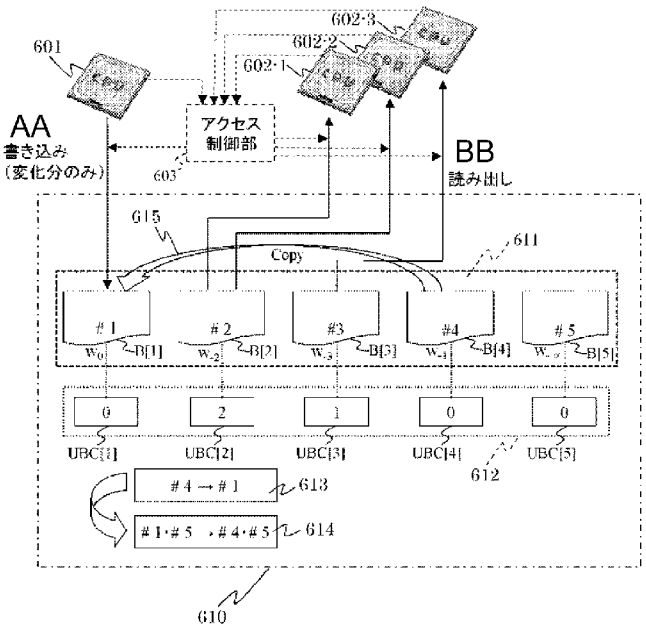
(74) 代理人: 戸田 裕二(**TODA Yuji**); 〒1008220 東京都千代田区丸の内一丁目６番１号 株式会社日立製作所内 Tokyo (JP).

(81) 指定国(表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ,

(54) **Title:** DATA TRANSMISSION DEVICE FOR IN-VEHICLE MULTI-CORE CONTROL AND ELECTRONIC CONTROL DEVICE

(54) 発明の名称： 車載マルチコア制御用データ伝達装置および電子制御装置

図６



603 Access control unit
AA Writing (changed portion only)
BB Reading

(57) **Abstract:** In order to enable a data set for which synchronicity has been maintained to be transferred between cores in a multi-core system while improving the real-time characteristic of processing by the cores, even when the cores operate asynchronously, the present invention is provided with bank memories B(1) – B(5), and a writing core 601 and reading cores 602-1, 602-2, and 602-3 capable of accessing these bank memories B(1) – (B5). Furthermore, an access control unit 603 assigns only the one writing memory 601 to the bank memories B(1) – B(5) to which writing is being carried

［続葉有］

WO 2019/065302 A1

NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

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

添付公開書類:

一 国際調査報告(条約第21条(3))

out, assigns one or more of the reading cores 602-1, 602-2, and 602-3 to the bank memories B(1) – B(5) from which reading is being carried out, and controls access to the bank memories B(1) – B(5) in an asynchronous manner such that the bank memories B(1) – B(5) to which writing is being carried out and the bank memories B(1) – B(5) from which reading is being carried out are not the same.

(57) 要約:マルチコアシステムにおけるコアが非同期に動作する場合おいても、コアによる処理のリアルタイム性を向上させつつ、同時性が確保されたデータセットをコア間で伝達する。バンクメモリＢ［1］～Ｂ［5］と、これらのバンクメモリＢ［1］～Ｂ［5］にアクセス可能な書き込みコア６０１および読み出しコア６０２－１、６０２－２、６０２－３が設けられ、アクセス制御部６０３は、書き込みが行われるバンクメモリＢ［1］～Ｂ［5］には、１個の書き込みコア６０１のみを割り当て、読み出しが行われるバンクメモリＢ［1］～Ｂ［5］には、１以上の読み出しコア６０２－１、６０２－２、６０２－３読を割り当て、書き込みが行われているバンクメモリＢ［1］～Ｂ［5］と読み出しが行われているバンクメモリＢ［1］～Ｂ［5］とが同一にならないようにバンクメモリＢ［1］～Ｂ［5］へのアクセスを排他的に制御する。

明　細　書

発明の名称 ：

車載マルチコア制御用データ伝達装置および電子制御装置

技術分野

[0001] 本発明は、エンジンの燃料制御などに適用可能な車載マルチコア制御用データ伝達装置および電子制御装置に関する。

背景技術

[0002] 高性能化への要求に応えるため、組込みシステムではこれまでＣＰＵ（Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）のクロック周波数を高めることで処理能力の向上を図ってきた。しかしその結果、ＣＰＵの消費電力、発熱量およびノイズなどの増加を招いていた。これに対して近年注目されているのが、ＣＰＵのコアをチップ内に複数実装し、各コアの負荷を減少させるマルチコアシステムである。

[0003] マルチコアシステムでは、今までシングルコアで実行してきた制御をコアごとに分割・分散し、並列処理を行うことによって処理スループットを向上させる。しかしながら、シングルコアの制御で行われていたＯＳ（Ｏｐｅｒａｔｉｏｎｇ　Ｓｙｓｔｅｍ）によるタスク間排他制御および割込み禁止によるクリティカル区間の割込み禁止制御がマルチコアにおいては成立しない。マルチコアシステムにおいて、同時性を確保しつつデータの組（データセット）を送り手ＣＰＵから受け手ＣＰＵに伝達する場合がある。この時、何らかの保護機構を設定しないと、受け手のＣＰＵがデータセットの一部のデータを読みだしている最中に、送り手のＣＰＵがそのデータセットの他のデータを書き換える現象が起こり得る。これは複数のＣＰＵが各々独自の実行コードを保有し、基本的に非同期で動作を行うためである。

[0004] このような現象を排除する手法として、従来からロック変数によるスピンロック制御が知られている。これはテスト・アンド・セット命令のようなアトミック（不可分）な変数チェック＆書き込み命令を用いて、特定のロック

変数に使用中を示す鍵をかけ、鍵をかけ得たＣＰＵのみが特定メモリ領域にアクセスできるプロセッサ間排他制御システムである。

[0005] しかしながら、スピンロック制御でプロセッサ間排他制御を行う資源に関わる実行区間は、マルチコアにとって並列実行区間ではなく、直列実行区間となる。従って、この直列実行時間が長くなればなるほど、アムダールの法則によってマルチコア全体として見た場合の並列実行性能が低下する。なお、アムダールの法則とは、複数のプロセッサを使い、並列計算によってプログラムの高速化を図る場合、そのプログラムの中で逐次的に実行しなければならない部分の時間によって、高速化が制限される現象を記述したものである。

[0006] スピンロックを行うメモリ粒度を大きく取り、複数のＣＰＵが関与すればするほど、ロック待ちに要する時間が増加（スケーラブル性に考慮したスピンロック機構の場合はＣＰＵ数に比例）し、ＣＰＵによる処理のリアルタイム性を損なう。

[0007] 一方、メモリ粒度を下げ、ロック変数（および該当排他資源）の個数を増加させると、管理が煩雑になって各ＣＰＵの制御に対する最悪実行時間を見積もること（および仕様変更に対する波及効果を予測すること）が困難になる。

[0008] 実際にはメモリ粒度を下げる方法はモチベーションが逆で、システムやマルチコアＯＳの制約上、設定できるロック変数の上限数が決められていることと、同一変数で異なるロック向けに値をコピーする等の管理が煩雑になるため、設計的にロック粒度が大きくなりがちである。極論すれば、細粒度ロックを多用するより、グローバル変数を丸ごとロック単位として扱った方が設計の見通し良くなる。（これをジャイアントロックと称する。）従って、この手法の帰結としてロックを取り合うＣＰＵの個数が増え、ロック待ち時間が見過ごせなくなる傾向にある。

[0009] 以上の観点からハードリアルタイム性を重視する車載制御用のマルチコアでは、スピンロック制御を採用しないことが好ましい。特に、マルチコアシ

ステムでは、ＣＰＵ数を増やすと、ロック待ち時間が程度の良い方式（これはスケーラブルな方式と呼ばれる）においても比例的に増加する。

[0010] そのためには、以下の先行技術で示すようなスピンロックを用いないＣＰＵ間データ伝達方式が考案されている。

先行技術文献

特許文献

[0011] 特許文献1：特開２００９－２１７７２１号公報

発明の概要

発明が解決しようとする課題

[0012] しかしながら、特許文献１に開示された技術は、データの送り手のＣＰＵをサーバ、受け手のＣＰＵをクライアントとみなして、サーバ－クライアント・モデルによるデータの授受を図るものである。

[0013] 受け手のＣＰＵは通知領域（共有メモリ）にデータ要求の指令を書き、送り手のＣＰＵは、その指令を解釈して受け手の割り当て領域（共有メモリ）に要求されたデータを書込み、その指令を完了済みの値に書き換える。

[0014] すなわち、完全なハンドシェーク動作であって、受け手のＣＰＵの要求頻度と送り手のＣＰＵのデータ生成頻度が完全に同期しないと、方式が破綻する。受け手の要求頻度が大きい時に適度に同一データを重ね読みしたり、送り手のデータ生成頻度が大きい時に受け手は適度にデータを読み飛ばしたりすることができない。これは、非同期のＣＰＵ間データ授受システムおいて負担になる。（データの授受が疎結合でない。）

[0015] また、サーバ－クライアント・モデルは、操作対象のオブジェクトが所属するＣＰＵ（送り手のＣＰＵ）に対する遠隔呼び出し法である。このため、データを生成するＣＰＵ（送り手のＣＰＵ）の動作を阻害し、操作を依頼したＣＰＵ（受け手のＣＰＵ）も操作完了通知を待つ必要があるため、送り手のＣＰＵも受け手のＣＰＵもリアルタイム性が悪くなる。

[0016] 本発明は、上記事情に鑑みなされたものであり、その目的は、マルチコア

システムにおけるコアが非同期に動作する場合おいても、コアによる処理のリアルタイム性を向上させつつ、同時性が確保されたデータセットをコア間で伝達することが可能な車載マルチコア制御用データ伝達装置および電子制御装置を提供することにある。

課題を解決するための手段

[0017] 上記目的を達成するため、第１の観点に係る車載マルチコア制御用データ伝達装置は、Ｍ（Ｍは２以上の整数）個のバンクメモリと、前記バンクメモリにアクセス可能なＮ（Ｎは２以上の整数）個のプロセッサとが設けられた車載マルチコア制御用データ伝達装置であって、前記Ｍ個のバンクメモリが配置され、前記Ｎ個のプロセッサからアクセス可能な共有メモリと、前記バンクメモリを特定する識別情報に基づいて、前記プロセッサから前記バンクメモリへのアクセスを制御するアクセス制御部とを備える。

発明の効果

[0018] 本発明によれば、マルチコアシステムにおけるコアが非同期に動作する場合おいても、コアによる処理のリアルタイム性を向上させつつ、同時性が確保されたデータセットをコア間で伝達することができる。

図面の簡単な説明

[0019] [図1]図１は、実施形態に係るマルチコア制御装置のハードウェア構成を示すブロック図である。

[図2]図２は、実施形態に係る電子制御装置が適用されるエンジンの燃料噴射系の構成例を示すブロック図である。

[図3]図３の（ａ）は、実施形態に係るデータ伝達装置に適用可能な単精度浮動小数点数のデータ構造を示す図、図３の（ｂ）は、実施形態に係るデータ伝達装置に適用可能な倍精度浮動小数点数のデータ構造を示す図、図３の（ｃ）は、図３の（ａ）および図３の（ｂ）のデータ構造のビットの割り当て方法を示す図である。

[図4]図４は、実施形態の比較例に係るスピンロック制御方法を示すフローチャートである。

[図5]図５は、実施形態の比較例に係るスピンロック制御と割込み禁止の関係を示すフローチャートである。

[図6]図６は、実施形態に係るデータ伝達装置の構成例を示すブロック図である。

[図7]図７は、実施形態に係るデータ伝達装置の読み出し手続きを示すフローチャートである。

[図8]図８は、実施形態に係るデータ伝達装置の書き込み手続きを示すフローチャートである。

[図9]図９は、実施形態に係るデータ伝達装置がエンジンの燃料噴射系に適用された時のデータ伝達手順を示すフローチャートである。

[図10]図１０は、実施形態に係るデータ伝達装置がエンジンの燃料噴射系に適用された時のデータ伝達方法を時系列的に示すシーケンス図である。

[図11]図１１（ａ）は、実施形態に係るデータ伝達装置の書き込みコアとスクラッチパッド・メモリとの占有関係を示す図、図１１（ｂ）は、実施形態に係るデータ伝達装置のスクラッチパッド・メモリごとのコアへの関数の割り当て方法を示す図である。

[図12]図１２は、実施形態に係るデータ伝達装置のスクラッチパッド・メモリのバンク数とコア数との関係を示す図である。

発明を実施するための形態

[0020] 実施形態について、図面を参照して説明する。なお、以下に説明する実施形態は特許請求の範囲に係る発明を限定するものではなく、また実施形態の中で説明されている諸要素及びその組み合わせの全てが発明の解決手段に必須であるとは限らない。また、実施例開示分野の一例としてエンジン制御を挙げているが、エンジン制御に限らず変速機制御などのドライブトレイン制御、防振やダンパーに関連するシャーシ制御、アンチロックブレーキシステムや衝突安全などのアクティブセーフティ、トラクションコントロールシステムなどの駆動力制御、および自動走行制御など車載用リアルタイム制御が要求される各分野に適用可能であることは言うまでもない。

[0021] 図１は、実施形態に係るマルチコア制御装置のハードウェア構成を示すブロック図である。なお、図1では、エンジンの燃料制御装置を２コアにて実現した構成を例にとった。

[0022] 図１において、この燃料制御装置には、ＣＰＵ（Ｃｅｎｔｒａｌ Ｐｒｏｃｅｓｓｉｎｇ Ｕｎｉｔ）１００、１１０、実行コード１０１、１１１が格納された実行コード格納領域、ローカルＲＡＭ（Ｒａｎｄｏｍ Ａｃｃｅｓｓ Ｍｅｍｏｒｙ）１０２、１１２および共有ＲＡＭ１２０が設けられている。

[0023] ＣＰＵ１００にはセンサ入力として、エアフローメータ１０３、圧力センサ１０４、スロットルセンサ１０５、回転数センサ１０６が接続されている。エアフローメータ１０３は、空気吸入量Ｑを計算するためのセンサ値を出力する。圧力センサ１０４は、吸気管圧力Ｐを計算するためのセンサ値を出力する。スロットルセンサ１０５は、スロットル開度αを計算するためのセンサ値を出力する。回転数センサ１０６は、エンジン回転数Ｎｅを計算するためのセンサ値を出力する。

[0024] ＣＰＵ１１０にはアクチュエータ出力として、燃料噴射量指示装置１１３が接続されている。燃料噴射量指示装置１１３は、エンジンの燃料噴射量ｑを出力する。

[0025] ＣＰＵ１００は主にセンサ入力の処理を担当し、実行コード１０１およびローカルＲＡＭ１０２を用いて処理を実行する。ローカルＲＡＭ１０２は、ＣＰＵ１００のみからアクセスできるメモリ資源である。

[0026] ＣＰＵ１１０は主に数値演算と出力アクチュエータ操作を担当し、実行コード１１１およびローカルＲＡＭ１１２を用いて処理を実行する。ローカルＲＡＭ１１２は、ＣＰＵ１１０のみからアクセスできるメモリ資源である。

[0027] ＣＰＵ１００、１１０には、共通してアクセス可能な資源として共有ＲＡＭ１２０がバス１０７を介して接続されている。共有ＲＡＭ１２０上には、全体としての制御を成立させる上で一方のＣＰＵ１００、１１０から他方のＣＰＵ１１０、１００に開示することが必要なデータの格納領域が設けられ

ている。この格納領域には、ＣＰＵ開示データ１２１、１２２が格納される。ＣＰＵ開示データ１２１には、空気吸入量Ｑ、吸気管圧力Ｐ、スロットル開度αおよびエンジン回転数Ｎｅを設定することができる。ＣＰＵ開示データ１２２には、燃料噴射量ｑを設定することができる。

[0028] 図２は、実施形態に係る電子制御装置が適用されるエンジンの燃料噴射系の構成例を示すブロック図である。図２では、燃料噴射系で主に用いられている３方式を示した。図２の（ａ）ではマスフロー方式、図２の（ｂ）ではスピードデンシティ方式、図２の（ｃ）ではα－Ｎ方式を示した。

[0029] 図２の（ａ）～（ｃ）において、エンジン２０３には吸気管２０２が接続され、吸気管２０２には燃料噴射装置２０５が接続されている。吸気管２０２の上流には、スロットルチャンバ２０１が設けられている。エンジン２０３には回転数センサ１０６が設けられている。さらに、燃料噴射系には、燃料制御を行うＥＣＵ（Ｅｌｅｃｔｒｏｎｉｃ Ｃｏｎｔｒｏｌ Ｕｎｉｔ）２０４が設けられている。ＥＣＵ２０４は、図１のＣＰＵ１００、１１０、実行コード１０１、１１１が格納された実行コード格納領域、ローカルＲＡＭ１０２、１１２および共有ＲＡＭ１２０を備えることができる。

[0030] また、図２の（ａ）に示すように、マスフロー方式では、エアフローメータ１０３がスロットルチャンバ２０１の上流に設けられる。

そして、ＥＣＵ２０４のＣＰＵ１００には、吸入空気量Ｑを計算するためのセンサ値がエアフローメータ１０３から入力されるとともに、エンジン回転数Ｎｅを計算するためのセンサ値が回転数センサ１０６から入力される。そして、ＣＰＵ１００において、これらのセンサ値から吸入空気量Ｑおよびエンジン回転数Ｎｅが計算され、共有ＲＡＭ１２０に格納される。

[0031] 次に、ＣＰＵ１１０において、吸入空気量Ｑおよびエンジン回転数Ｎｅが共有ＲＡＭ１２０から読み出される。そして、燃料噴射量ｑがｑ∝Ｑ／Ｎｅとして計算される。

[0032] 一方、図２の（ｂ）に示すように、スピードデンシティ方式では、圧力センサ１０４が吸気管２０２に設けられる。

そして、ＥＣＵ２０４のＣＰＵ１００には、吸気管圧力Ｐを計算するためのセンサ値が圧力センサ１０４から入力されるとともに、エンジン回転数Ｎｅを計算するためのセンサ値が回転数センサ１０６から入力される。そして、ＣＰＵ１００において、これらのセンサ値から吸気管圧力ＰおよびエンジンＮｅが計算され、共有ＲＡＭ１２０に格納される。

[0033] 次に、ＣＰＵ１１０において、吸気管圧力ＰおよびエンジンＮｅが共有ＲＡＭ１２０から読み出される。そして、燃料噴射量ｑがｑ∝f_1（Ｐ，Ｎｅ）として計算される。なお、f_1（）はエンジン特性より決定される関数である。

[0034] 一方、図２の（ｃ）に示すように、α－Ｎ方式では、スロットルセンサ１０５がスロットルチャンバ２０１に設けられる。

そして、ＥＣＵ２０４のＣＰＵ１００には、スロットル開度αを計算するためのセンサ値がスロットルセンサ１０５から入力されるとともに、エンジン回転数Ｎｅを計算するためのセンサ値が回転数センサ１０６から入力される。そして、ＣＰＵ１００において、これらのセンサ値からスロットル開度αおよびエンジン回転数Ｎｅが計算され、共有ＲＡＭ１２０に格納される。

[0035] 次に、ＣＰＵ１１０において、スロットル開度αおよびエンジン回転数Ｎｅが共有ＲＡＭ１２０から読み出される。そして、燃料噴射量ｑがｑ∝f_2（α，Ｎｅ）として計算される。なお、f_2（）はエンジン特性より決定される関数である。

[0036] ここで、センサ入力処理を行うＣＰＵ１００で計算されたデータセット｛Ｑ，Ｐ，α，Ｎｅ｝が、燃料噴射量ｑを計算するＣＰＵ２で使われる場合、データセット｛Ｑ，Ｐ，α，Ｎｅ｝の各値が、ほとんど同時刻に計測されたセンサ値から得られたデータであることが重要である。この関係をデータセットの同時性と言う。このデータセット内のデータ間の同時性を巨視的な同時性と言う。このデータセットの同時性が崩れると、データセット｛Ｑ，Ｐ，α，Ｎｅ｝に応じた燃料噴射量ｑの算出精度が低下する。

[0037] 例えば、図２の（ａ）のマスフロー方式において、ＣＰＵ１１０がＣＰＵ

開示データ１２１の吸入空気量Ｑを読み出している間に、ＣＰＵ１００がＣＰＵ開示データ１２１のエンジン回転数Ｎｅを別時刻の値に勝手に書き換えると、ＣＰＵ１１０にとって吸入空気量Ｑとエンジン回転数Ｎｅの同時性が崩れる。この時、ｑ∝Ｑ／Ｎｅとして計算された燃料噴射量ｑは間違った制御データとなる。このため、マルチコアシステムでは、データセットの同時性を確保し、データセットの同時性が崩れないように注意する必要がある。

[0038] 図３の（ａ）は、実施形態に係るデータ伝達装置に適用可能な単精度浮動小数点数のデータ構造を示す図、図３の（ｂ）は、実施形態に係るデータ伝達装置に適用可能な倍精度浮動小数点数のデータ構造を示す図、図３の（ｃ）は、図３の（ａ）および図３の（ｂ）のデータ構造のビットの割り当て方法を示す図である。なお、図３の（ａ）および図３の（ｂ）では、アドレス昇順を縦軸にとった。

[0039] 図３の（ａ）および図３の（ｂ）の例では、ビッグエンディアン形式のデータ格納順とし、ＩＥＥＥ７５４規格（ＩＥＥＥ Ｓｔａｎｄａｒｄ ｆｏｒ Ｆｌｏａｔｉｎｇ－Ｐｏｉｎｔ Ａｒｉｔｈｍｅｔｉｃ （ＡＮＳＩ／ＩＥＥＥ Ｓｔｄ ７５４－２００８））に準拠した。

[0040] 図３の（ａ）には、４バイトの単精度データ３０１を示した。図３の（ｂ）には、８バイトの倍精度データ３０２を示した。単精度データ３０１および倍精度データ３０２において、ｓは符号ビットを、ｅは指数部のビットを、ｆは仮数部のビットを示す。

[0041] 図３の（ｃ）に示すように、単精度データ３０１では、符号ビットに１ビット、指数部に８ビット、仮数部に２３ビットが割り当てられる。倍精度データ３０２では、符号ビットに１ビット、指数部に１１ビット、仮数部に５２ビットが割り当てられる。

[0042] 単精度データ３０１（または倍精度データ３０２）が１つの数値であるといっても、その数値の書き換えはアドレスにマッピングされたバイトごとに行われる。このため、単精度データ３０１（または倍精度データ３０２）の一部が部分的に書き換わっている最中に、その単精度データ３０１（または

倍精度データ３０２）にアクセスすると、データとして意味をなさない数値が取り込まれる。

[0043] 複数バイトにまたがるデータ構造のデータは、すべてのバイトのデータの書き換えが終わってから取り込まなければならない。また、読み手がデータを読んでいる最中に、そのデータの一部のバイトを書き換えてはならない。複数バイトにまたがるデータ構造を持つ１つのデータの中のバイト間の一貫した同時性を微視的な同時性と言う。

[0044] マルチコアシステムにおいて取り込まれるデータが意味を持つためには、巨視的な同時性だけでなく、微視的な同時性も確保する必要がある。

[0045] 以上述べたようなデータの同時性の破壊は、複数のＣＰＵによる同一メモリ領域への読み書きの競合に起因する。このような競合を回避するために、複数のＣＰＵが同一メモリ領域に同時にアクセスできないＣＰＵ間排他制御を行うことがある。

[0046] 図４は、実施形態の比較例に係るスピンロック制御方法を示すフローチャートである。

図４では、ＣＰＵ間排他制御を行うために、スピンロック制御方法を用いる場合を示した。図４において、ＣＰＵはロック変数を読み込む（Ｓ４０１）。ロック変数は、排他制御される共有資源と同様に共有メモリ上に確保される。

[0047] 次に、このロック変数がテストされ、解放されているか（値がゼロか）がチェックされる（Ｓ４０２）。ロック変数が解放されていれば、ＣＰＵはロック変数を取得し、ロック変数へ値を書き込む（Ｓ４０３）。ロック変数が解放されていなければ、Ｓ４０１へ戻り、ＣＰＵはロック変数を再び読み込み、テストを繰り返す。

[0048] 次に、ＣＰＵはロック変数を取得すると、共有資源にアクセスする（Ｓ４０４）。その後、ＣＰＵはロック変数を解放する（Ｓ４０５）。

[0049] この動作は、ロック変数が解放されるまで、ループ操作によって変数を読み込み、テストを続行しつつ待ちを続けるのでスピンロックと呼ばれる。

Ｓ４０１、Ｓ４０２およびＳ４０３の一連の動作はテスト・アンド・セット命令のようなアトミック命令Ｂ４００を用いて実装される。アトミック命令とは、あるＣＰＵが命令実行中に他のＣＰＵが干渉できない不可分な複合命令セットのことである。

[0050] 図５は、実施形態の比較例に係るスピンロック制御と割込み禁止の関係を示すフローチャートである。図５では、図４で示したスピンロック制御と割込み禁止区間とのネスト関係を示した。

図５において、プロセッサ内排他制御区間Ｂ５００には、割込み禁止（Ｓ５０１）と割込み許可（Ｓ５０５）とが設けられている。割込み禁止（Ｓ５０１）と割込み許可（Ｓ５０５）とで挟まれた区間は、割込み禁止区間である。プロセッサ間排他制御区間Ｂ５１０は、この割込み禁止区間に設けられている。プロセッサ間排他制御区間Ｂ５１０では、ＣＰＵがロックを取得すると（Ｓ５０２）、そのＣＰＵが共有資源へアクセスし（Ｓ５０３）、その後、ＣＰＵはロックを開放（Ｓ５０４）する。

[0051] この時、あるＣＰＵが共有資源へアクセスしている間は、他のＣＰＵから共有資源へのアクセスが禁止される。また、プロセッサ内排他制御Ｂ５００の中にプロセッサ間排他制御Ｂ５１０が入り、このネスト関係が維持される。

このネスト関係を維持することにより、あるＣＰＵがロックを取得してからロックを開放するまでは、そのＣＰＵは割込み制御ができなくなる。割込み制御は、そのＣＰＵ独自の都合で行われるので、複数ＣＰＵ間のロック実行時間・待ち時間の量的関係に含ませるべきではないからである。

[0052] したがって、あるＣＰＵのロック取得中の割込み制御を禁止することにより、その割込み制御によって、そのＣＰＵからロックが解放されるまでの時間が延びるのを防止することができる。このため、他のＣＰＵのロック待ち時間の増加を抑制することができ、リアルタイム制御の実効性を確保することができる。また、ロック待ちのＣＰＵは、ロック待ち時間の最悪値を正しく見積もることができる。逆に、ＣＰＵのロック取得中の割込み制御を許す

と、ロック待ち時間を管理することが困難になってしまう。

[0053] 図５のネスト関係を維持したままスピンロックを行う方法は、ロック待ちをするＣＰＵが増加すると、ロック待ち時間が増加する。このロック待ち時間は、最善手法でロック待ちをするＣＰＵ数に対して比例的に増加し、最悪手法で指数関数的に増加する。ロック待ち時間の増加をＣＰＵ数に比例的にする手法は、ＣＰＵ数に対してスケーラブルな手法と呼ばれる。この手法では、ロック待ちをするＣＰＵの要求が待ち行列で記憶され、要求順にロック権利が与えられる。

[0054] ここで、図５に示すように、ロック待ちを行うプロセッサ間排他制御Ｂ５１０は、割込みを禁止するプロセッサ内排他制御Ｂ５００の中に位置している。このため、ロック待ち時間（Ｓ５０２～Ｓ５０４）が増加すると、割込み禁止時間（Ｓ５０１～Ｓ５０５）も必然的に増加する。

[0055] すなわち、ＥＣＵ２０４の処理性能を上げるために、マルチコア化されたＣＰＵ１００、１１０を用いると、スピンロック制御では、ＣＰＵ数の増加に従って割込み禁止時間も増加する。割込み禁止時間の増加は、車載用途におけるハードリアルタイム制御の破綻を招く。ハードリアルタイム制御の破綻を防止するため、ＣＰＵ数を制限すると、ＥＣＵ２０４の処理性能を上げるのが困難となる。

[0056] ＣＰＵ数を制限することなく、ハードリアルタイム制御の破綻を防止するため、ロック変数を用いた待ち合わせが生じないロックレス方式が望ましい。

[0057] 図６は、実施形態に係るデータ伝達装置の構成例を示すブロック図である。なお、以下の説明では、５個のバンクメモリと、これらのバンクメモリにアクセス可能な４個のＣＰＵとが設けられた構成を例にとる。ＣＰＵは、プロセッサとして用いることができる。プロセッサは、ＭＰＵ（Ｍｉｃｒｏ Ｐｒｏｃｅｓｓｉｎｇ Ｕｎｉｔ）またはＤＳＰ（Ｄｉｇｉｔａｌ Ｓｉｇｎａｌ Ｐｒｏｃｅｓｓｏｒ）などであってもよい。

[0058] 図６において、データ伝達装置には、５個のバンクメモリＢ［１］～Ｂ［

５］と、これらのバンクメモリＢ［１］～Ｂ［５］にアクセス可能な書き込みコア６０１および読み出しコア６０２－１、６０２－２、６０２－３が設けられている。書き込みコア６０１は、書き込みに使われるＣＰＵに設けることができる。読み出しコア６０２－１、６０２－２、６０２－３は、読み出しに使われるＣＰＵに設けることができる。

[0059] バンクメモリＢ［１］～Ｂ［５］は共有メモリ６１０に設けることができる。共有メモリ６１０は、書き込みコア６０１および読み出しコア６０２－１、６０２－２、６０２－３からアクセス可能である。バンクメモリＢ［１］～Ｂ［５］は、同一の変数内容および先頭から同一オフセットアドレスを持つように構成することができる。バンクメモリＢ［１］～Ｂ［５］には、バンクメモリＢ［１］～Ｂ［５］を特定する識別情報としてバンク番号＃１～＃５がそれぞれ付加されている。

[0060] バンクメモリＢ［１］～Ｂ［５］をまとめることでスクラッチパッド・メモリ６１１を構成することができる。スクラッチパッド・メモリ６１１は、データの一時的な受け渡しに使うことができる。スクラッチパッド・メモリ６１１は、バンクメモリ群すなわちバンクメモリＢ［１］～Ｂ［５］をまとめて指し示す時の呼称である。

[0061] さらに、データ伝達装置には、アクセス制御部６０３が設けられている。アクセス制御部６０３は、バンクメモリＢ［１］～Ｂ［５］を特定するバンク番号＃１～＃５に基づいて、書き込みコア６０１および読み出しコア６０２－１、６０２－２、６０２－３からバンクメモリＢ［１］～Ｂ［５］へのアクセスを制御する。この時、アクセス制御部６０３は、対象のバンクメモリＢ［１］～Ｂ［５］を対応付けたバンク番号＃１～＃５を用いて管理することができる。

[0062] 具体的には、アクセス制御部６０３は、書き込みが行われるバンクメモリＢ［１］～Ｂ［５］には、１個の書き込みコア６０１のみを割り当てる。読み出しが行われるバンクメモリＢ［１］～Ｂ［５］には、１以上の読み出しコア６０２－１、６０２－２、６０２－３を割り当てる。そして、書き込み

が行われているバンクメモリＢ［１］～Ｂ［５］と読み出しが行われているバンクメモリＢ［１］～Ｂ［５］とが同一にならないようにバンクメモリＢ［１］～Ｂ［５］へのアクセスを排他的に制御することができる。

[0063] なお、アクセス制御部６０３は、書き込みコア６０１または読み出しコア６０２－１、６０２－２、６０２－３が設けられたＣＰＵがアクセス制御に関するプログラムを実行することにより、各々のＣＰＵのソフトウェア上またはファームウェア上で分散して実現することができる。

[0064] この時、書き込みコア６０１が設けられたＣＰＵに対し、書き込み操作を実現する実行コードおよびこの実行コードを含む関数がローカルに保持される。そして、書き込みコア６０１が設けられたＣＰＵが、書き込み操作を実現する実行コードおよびこの実行コードを含む関数を実行することでアクセス制御部６０３を実現することができる。

[0065] また、読み出しコア６０２－１、６０２－２、６０２－３が設けられたＣＰＵに対し、読み出し操作を実現する実行コードおよびこの実行コードを含む関数がローカルに保持される。そして、読み出しコア６０２－１、６０２－２、６０２－３が設けられたＣＰＵが、読み出し操作を実現する実行コードおよびこの実行コードを含む関数を実行することでアクセス制御部６０３を実現することができる。

[0066] さらに、共有メモリ６１０には、カウンタ配列領域６１２およびレジスタ６１３、６１４が設けられている。カウンタ配列領域６１２は、読み出しコア６０２－１、６０２－２、６０２－３からバンクメモリＢ［１］～Ｂ［５］への現在のアクセス数のカウント値ＵＢＣ［１］～ＵＢＣ［５］をバンクメモリＢ［１］～Ｂ［５］ごとに対応して記憶する。カウンタ配列領域６１２のＵＢＣ［１］～ＵＢＣ［５］をバンク使用中カウンタと呼ぶ。

[0067] レジスタ６１３は、書き込みコア６０１によって最後に書き込みが行われたバンクメモリＢ［１］～Ｂ［５］を特定するバンク番号＃１～＃５を記憶する。レジスタ６１３で示されるバンクメモリＢ［１］～Ｂ［５］を最新データバンクと言う。レジスタ６１４は、最後に書き込みが行われたバンクメ

モリＢ［１］～Ｂ［５］（すなわちレジスタ６１３で示されるバンクメモリ）以外で、かついずれの読み出しコア６０２－１、６０２－２、６０２－３からもアクセスされていないバンクメモリＢ［１］～Ｂ［５］を特定するバンク番号＃１～＃５を記憶する。レジスタ６１４で示されるバンクメモリＢ［１］～Ｂ［５］をフリーバンクと言う。

[0068] この時、書き込みコア６０１および読み出しコア６０２－１、６０２－２、６０２－３は、カウンタ配列領域６１２に記憶されたカウント値ＵＢＣ［１］～ＵＢＣ［５］およびレジスタ６１３、６１４に記憶されたバンク番号＃１～＃５を参照したり、変更したりすることができる。

[0069] カウンタ配列領域６１２に記憶されたカウント値ＵＢＣ［１］～ＵＢＣ［５］およびレジスタ６１３、６１４に記憶されたバンク番号＃１～＃５を変更する場合、他の書き込みコア６０１および読み出しコア６０２－１、６０２－２、６０２－３からの干渉（競合）を排除できるようにアトミック命令を使用することができる。

[0070] 読み出しコア６０２－１、６０２－２、６０２－３からの読み出し要求があると、アクセス制御部６０３は、レジスタ６１３に記憶されたバンク番号＃１～＃５を参照する。そして、その参照結果に基づいて、最後に書き込みが行われたバンクメモリＢ［１］～Ｂ［５］を特定し、そのバンク番号＃１～＃５を読み出しコア６０２－１、６０２－２、６０２－３に通知する。さらに、アクセス制御部６０３は、レジスタ６１３に記憶されたバンク番号＃１～＃５に対応するバンクメモリＢ［１］～Ｂ［５］のカウント値ＵＢＣ［１］～ＵＢＣ［５］を１だけインクリメントする。

[0071] 次に、読み出しコア６０２－１、６０２－２、６０２－３からバンクメモリＢ［１］～Ｂ［５］の返却要求があると、アクセス制御部６０３は、そのバンクメモリＢ［１］～Ｂ［５］のカウント値ＵＢＣ［１］～ＵＢＣ［５］を１だけデクリメントする。そして、そのバンクメモリＢ［１］～Ｂ［５］のカウント値ＵＢＣ［１］～ＵＢＣ［５］が０、かつバンク番号＃１～＃５がレジスタ６１３に記憶されたバンク番号＃１～＃５と不一致の場合、その

バンクメモリＢ［１］～Ｂ［５］のバンク番号＃１～＃５をレジスタ６１４に登録する。

[0072] 一方、書き込みコア６０１からの書き込み要求があると、アクセス制御部６０３は、レジスタ６１４に記憶されたバンク番号＃１～＃５を参照する。そして、その参照結果に基づいて、最後に書き込みが行われたバンクメモリＢ［１］～Ｂ［５］（すなわちレジスタ６１３で示されるバンクメモリ）以外で、かついずれの読み出しコア６０２－１、６０２－２、６０２－３による読み出しにも使用されていないバンクメモリＢ［１］～Ｂ［５］を特定する。そして、アクセス制御部６０３は、その特定されたバンクメモリＢ［１］～Ｂ［５］のバンク番号＃１～＃５を書き込みコア６０１に通知する。さらに、アクセス制御部６０３は、書き込みコア６０１に通知したバンク番号＃１～＃５をレジスタ６１４から削除する。

[0073] 書き込みが行われるバンクメモリＢ［１］～Ｂ［５］のバンク番号＃１～＃５が書き込みコア６０１に通知される前に、最新のデータを保持しているバンクメモリＢ［１］～Ｂ［５］（すなわちレジスタ６１３で示されるバンクメモリ）のデータが、書き込みコア６０１に通知されるバンク番号＃１～＃５に対応したバンクメモリＢ［１］～Ｂ［５］にコピーされる。この時、書き込みコア６０１は、前回の書き込み時から値が変化した差分のみを書き換えることができる。

[0074] 書き込みコア６０１が最新のデータを書いたバンクメモリＢ［１］～Ｂ［５］を公開する場合、レジスタ６１３に今まで記憶されたバンク番号＃１～＃５が、最新のデータが書かれたバンクメモリＢ［１］～Ｂ［５］のバンク番号＃１～＃５に書き換えられる。レジスタ６１３に今まで記憶されたバンク番号＃１～＃５に対応するバンクメモリＢ［１］～Ｂ［５］のカウント値ＵＢＣ［１］～ＵＢＣ［５］が０の場合、そのバンクメモリＢ［１］～Ｂ［５］のバンク番号＃１～＃５がレジスタ６１４に登録される。

[0075] 以下、書き込みコア６０１および読み出しコア６０２－１、６０２－２、６０２－３の動作について時系列順に説明する。各バンクメモリＢ［１］～

Ｂ［５］の左下に表記した添え字付きのＷの数字の順番でバンクメモリＢ［１］～Ｂ［５］の値が更新された（書き込まれた）ものとする。この時、バンクメモリＢ［１］～Ｂ［５］の読み出しコアからの被参照数に対応してカウント値ＵＢＣ［１］～ＵＢＣ［５］が設定される。

[0076] W_{-2}で２回前に書き込まれたバンクメモリＢ［２］のカウント値ＵＢＣ［２］は２である。この時、２個の読み出しコア６０２－１、６０２－２がバンクメモリＢ［２］にアクセス中である。W_{-3}で３回前に書き込まれたバンクメモリＢ［３］のカウント値ＵＢＣ［３］は１である。この時、１個の読み出しコア６０２－３がバンクメモリＢ［３］にアクセス中である。

[0077] W_{-1}で前回に最新のデータとして書き込まれたバンクメモリＢ［４］のカウント値ＵＢＣ［４］は０であり、まだどの読み出しコア６０２－１～６０２－３も読み出していない。この時、バンクメモリＢ［４］のバンク番号＃４はレジスタ６１３に登録される。

[0078] $W_{-\infty}$で少なくとも注目している時間帯では、書き込まれていないバンクメモリＢ［５］のカウント値ＵＢＣ［５］は０であり、どの読み出しコア６０２－１～６０２－３からもアクセスされていない。この時、バンクメモリＢ［５］はフリーバンクとなり、そのバンク番号＃５はレジスタ６１４に登録される。

[0079] ここで、書き込みコア６０１からの書き込み要求があると、それまでレジスタ６１４に登録されていたバンク番号＃１のバンクメモリＢ［１］が書き込み用に予約バンクとして確保され、書き込みコア６０１に提供される。この時、バンクメモリＢ［１］のカウント値ＵＢＣ［１］は０である。バンクメモリＢ［１］が書き込みコア６０１に提供する前に、それまでの最新のデータが記憶されていたバンクメモリＢ［４］（すなわちレジスタ６１３で示されるバンクメモリ）のデータがバンクメモリＢ［１］にコピーされる（６１５）。

[0080] 書き込みコア６０１からバンクメモリＢ［１］への書き込みが終わり、そのバンクメモリＢ［１］が読み出しコア６０２－１～６０２－３に公開され

ると、レジスタ６１３にそれまで登録されていたバンク番号＃４がバンク番号＃１に書き換えられる。前回までの（一段古い）最新のデータが格納されていたバンクメモリＢ［４］のカウント値ＵＢＣ［４］は０であり、どの読み出しコア６０２－１～６０２－３からも読み出し中ではない。このため、バンクメモリＢ［４］はフリーバンクとなり、バンクメモリＢ［４］のバンク番号＃４がレジスタ６１４に登録される。すなわち、レジスタ６１４に登録されたバンク番号は｛＃１，＃５｝から｛＃４，＃５｝に変化する。

[0081] ここで、読み出しコア６０２－１～６０２－３は、バンクメモリＢ［１］～Ｂ［５］にアクセスする時に、レジスタ６１３に登録されたバンク番号＃１～＃５を参照することにより、書き込みコア６０１による書き込みと競合することなく、バンクメモリＢ［１］～Ｂ［５］から最新のデータを読み出すことができる。書き込みコア６０１は、バンクメモリＢ［１］～Ｂ［５］にアクセスする時に、レジスタ６１４に登録されたバンク番号＃１～＃５を参照することにより、読み出しコア６０２－１～６０２－３による読み出しと競合することなく、バンクメモリＢ［１］～Ｂ［５］に最新のデータを書き込むことができる。

[0082] この時、バンクメモリＢ［１］～Ｂ［５］に書き込める書き込みコア６０１を１個のみに制限することにより、バンクメモリＢ［１］～Ｂ［５］に書きこまれたデータセットの同時性を確保しつつ、最新のデータが書き込まれたバンクメモリＢ［１］～Ｂ［５］を正確に指定することが可能となる。

[0083] これにより、ロックレスで書き込みコア６０１と読み出しコア６０２－１～６０２－３との間でのデータ伝達の最新性および同時性を保証することができる。すなわち、マルチコアシステムによる処理のリアルタイム性を向上させつつ、同時性が確保された最新データセットをコア間で伝達することが可能となる。読み出しコアの個数が増加しても、スクラッチパッド・メモリ６１１内のメモリバンクの個数を増加させることで対処できる。したがって、スピンロックによる待ち時間を必要としない１対ｎの同報データ伝達装置を実現することができる。

[0084] 書き込みコア６０１と読み出しコア６０２－１～６０２－３の動作タイミングは非同期である。読み出し周期より書き込み周期の方が早ければ、読み出し側で適宜データを読み飛ばすことができ、書き込み周期より読み出し周期の方が早ければ、同一データを適宜重ね読みすることができる。このため、マルチコアシステムの書き込み周期と読み出し周期とを一致させる必要がなくなり、各コアに対するデータ入出力期間の柔軟性を向上させることができる。

[0085] 図７は、実施形態に係るデータ伝達装置の読み出し手続きを示すフローチャートである。

図７において、読み出し手続きＳ７００が開始されると、読み出しコア６０２－１～６０２－３は、Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０を呼び、戻り値よりアクセス可能なバンク番号ｉｄを取得する（ステップＳ７１０）。なお、図６の例では、バンク番号ｉｄは、＃１～＃５のいずれかの値に設定される。

[0086] 次に、読み出しコア６０２－１～６０２－３は、ｒｅａｄ（ｉｄ，ｏｆｆｓｅｔ）を実行する。この時、そのバンク番号ｉｄで指定されるバンクメモリＢ［ｉｄ］の先頭よりのオフセットアドレスｏｆｆｓｅｔで所望の変数値ｄａｔａを読み出す（ステップＳ７２０）。ステップＳ７２０は、所望の変数値ｄａｔａをすべて読み出すまで繰り返し実行することができる。

[0087] 次に、読み出しコア６０２－１～６０２－３は、Ｒｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０を呼び出し、Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０で確保したバンクの返却を行う（ステップＳ７３０）。Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０のコールとＲｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０のコールとの間では、Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０で確保されたバンクメモリＢ［ｉｄ］のデータはキャプチャされ、他の要因で変更されることはない。このため、演算処理などに使われるデータセットの同時性を確保することができる。

[0088] 上述した実施形態では、一連の読み出し手続きＳ７００が、コアごとに１か所で使われる前提で説明したが、同一コアのプリエンプション可能な別タ

スクの複数個所で使われてもよい。その場合は、読み出し複数個所に相当する分読み出しコア数が増えたものと見なし、バンクメモリの数を増加させて対応することができる。ただし、メモリ資源の節約の観点から、同一コアの主たる１か所のタスクがアクセスし、値はローカル変数にコピーして複数のタスクで使いまわす方がよい。この［１外部入力インターフェース／１コア］の手法により、管理上の煩雑さの低減、デバッグおよび変更の容易化を図ることができる。

[0089] Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０では、レジスタ６１３より最新のデータが存在するバンクメモリＢ［１］～Ｂ［５］のバンク番号＃１～＃５を取得する。この時、最新のデータが存在するバンクメモリＢ［１］～Ｂ［５］のバンク番号＃１～＃５をＬＩＢＲ（Ｌａｔｅｓｔ Ｉｎｆｏｒｍａｔｉｏｎ Ｂａｎｋ Ｒｅｇｉｓｔｅｒ）とすると、変数ｘにＬＩＢＲを設定する。

[0090] そして、そのバンクメモリＢ［１］～Ｂ［５］のカウント値ＵＢＣ［ｘ］を１だけインクリメントする（ステップＳ７１１）。レジスタ６１３およびカウント値ＵＢＣ［１］～ＵＢＣ［５］は共有メモリ６１０上に存在しているため、これらの操作はアトミック命令で実行される。

[0091] 次に、最新のデータが存在するバンクメモリＢ［１］～Ｂ［５］のバンク番号＃１～＃５を特定する変数ｘを戻り値として、メインルーチンにリターンを行う（ステップＳ７１２）。

[0092] Ｒｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０では、バンク番号ｉｄで指定されるバンクメモリＢ［ｉｄ］の返却に先立って、そのバンクメモリＢ［ｉｄ］のカウント値ＵＢＣ［ｉｄ］を１だけデクリメントする（ステップＳ７３１）。カウント値ＵＢＣ［ｉｄ］は共有メモリ６１０上に存在しているため、この操作はアトミック命令で実行される。

[0093] 次に、カウント値ＵＢＣ［ｉｄ］が０となり、かつバンク番号ｉｄがレジスタ６１３に登録されたバンク番号＃１～＃５とが一致しない（すなわち最新データではない）場合、バンク番号ｉｄで指定されるバンクメモリＢ［ｉｄ］をフリーバンクとみなし、そのバンク番号ｉｄをレジスタ６１４に登録

する（ステップＳ７３２）。レジスタ６１３、６１４およびカウント値ＵＢＣ［１］～ＵＢＣ［５］は共有メモリ６１０上に存在しているため、これらの操作はアトミック命令で実行される。

[0094] 次に、メインルーチンにリターンし、一連の読み出し手続きＳ７００を終了する（ステップＳ７３３）。

[0095] 図８は、実施形態に係るデータ伝達装置の書き込み手続きを示すフローチャートである。

図８において、書き込み手続きＳ８００が開始されると、書き込みコア６０１は、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０を呼び、戻り値よりアクセス可能なバンク番号ｉｄを取得する（ステップＳ８１０）。なお、図６の例では、バンク番号ｉｄは、＃１～＃５のいずれかの値に設定される。

[0096] 次に、書き込みコア６０１は、ｗｒｉｔｅ（ｉｄ，ｏｆｆｓｅｔ，ｄａｔａ）を実行する。この時、そのバンク番号ｉｄで指定されるバンクメモリＢ［ｉｄ］の先頭よりのオフセットアドレスｏｆｆｓｅｔで所望の変数値ｄａｔａを書き換える（ステップＳ８２０）。ステップＳ８２０は、要求する変数値ｄａｔａをすべて書き換えるまで繰り返し実行することができる。

[0097] 値に変化のない変数は、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０で前回の最新のデータがあらかじめコピーされており、何も操作を行う必要がない。

[0098] 次に、Ｐｕｔ＿Ｂａｎｋ関数ｆ８３０を呼び出し、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０で確保したバンクメモリＢ［ｉｄ］の公開を行う（ステップＳ８３０）。Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０のコールとＰｕｔ＿Ｂａｎｋ関数ｆ８３０のコールとの間では、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０で確保したバンクメモリＢ［ｉｄ］のデータは、読み出しコア６０２－１～６０２－３のアクセスから保護され、他の要因で参照されることはない。このため、読み出しコア６０２－１～６０２－３の視点で、演算処理などに使われるデータセットの同時性を確保することができる。

[0099] Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０では、レジスタ６１４よりフリーバンクのバンク番号＃１～＃５を取得し、そのバンク番号＃１～＃５をレジ

スタ６１４から削除する（ススステップＳ８１１）。図８では、レジスタ６１４をＦＢＲ（Ｆｒｅｅ　Ｂａｎｋ　Ｒｅｇｉｓｔｅｒ）として表し、ＦＢＲからバンク番号を1つ選択して変数ｘに設定している。ＦＢＲから選択して取り出したバンク番号は、ＦＢＲ自体からは削除される。レジスタ６１４は共有メモリ６１０上に存在しているため、これらの操作はアトミック命令で実行される。

[0100] 次に、最新データが確保されているバンクメモリＢ［ＬＩＢＲ］から、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０で確保されたバンクメモリＢ［ｘ］にデータのコピーを行う（ステップＳ８１２）。

[0101] 次に、フリーバンクのバンク番号＃１～＃５を特定する変数ｘを戻り値として、メインルーチンにリターンを行う（ステップＳ８１３）。

[0102] Ｐｕｔ＿Ｂａｎｋ関数ｆ８３０では、バンク番号で指定されるバンクメモリＢ［ｉｄ］の公開に先立って、今までのＬＩＢＲの値を呼び出し、変数ｏｌｄに一旦保存する（ステップＳ８３１）。変数ｏｌｄは、書き込みコア６０１のローカルＲＡＭ（１０２、１１２）に格納するため、アトミック命令を使わなくてよい。

[0103] 次に、レジスタ６１３に登録されているＬＩＢＲの値を今回公開するバンクメモリＢ［ｉｄ］のバンク番号ｉｄで書き換える（ステップＳ８３２）。レジスタ６１３は共有メモリ６１０上に位置しているが、値を書き換える関数はこのＰｕｔ＿Ｂａｎｋ関数ｆ８３０しか存在しないため、アトミック命令を使わなくてもよい。また、この時点では、最新のデータの準備が完了しているため、任意の読み出しコア６０２－１～６０２－３から値を参照することができる。

[0104] 次に、一つ前の最新データが確保されていたバンクメモリＢ［ｏｌｄ］のカウント値ＵＢＣ［ｏｌｄ］がチェックされる（ステップＳ８３３）。このカウント値ＵＢＣ［ｏｌｄ］が０であった場合、そのバンクメモリＢ［ｏｌｄ］は、どの読み出しコア６０２－１～６０２－３からも参照されておらず、データが古くなっている。このため、バンクメモリＢ［ｏｌｄ］をフリー

バンクとみなし、この変数ｏｌｄをレジスタ６１４（図８ではＦＢＲと表記）に登録する。レジスタ６１４およびカウント値ＵＢＣ［１］～ＵＢＣ［５］は共有メモリ６１０上に存在しているため、これらの操作はアトミック命令で実行される。

[0105] 次に、メインルーチンにリターンし、一連の書き込み手続きＳ８００を終了する（ステップＳ８３４）。

[0106] 以下、図６から図８で説明した実施形態をマルチコア組み込み制御ソフトに適用した例を示す。この適用例として、図１のエンジンの燃料噴射系を例にとる。この時、図１のＣＰＵ１００には、図６の書き込みコア６０１を設けることができる。図１のＣＰＵ１１０には、図６の読み出しコア６０２－１を設けることができる。図１の共有ＲＡＭ１２０には、図６のバンクメモリＢ［１］～Ｂ［３］（なぜ３つのバンクメモリが必要かは図１２で示す）およびレジスタ６１３、６１４を設けるとともに、カウント値ＵＢＣ［１］～ＵＢＣ［３］を記憶することができる。バンクメモリＢ［１］～Ｂ［３］を束ねることでスクラッチパッド・メモリ６１１を構成することができる。

[0107] Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０およびＰｕｔ＿Ｂａｎｋ関数ｆ８３０は、実行コード１０１が格納された実行コード格納領域に実装することができる。Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０およびＲｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０は、実行コード１１１が格納された実行コード格納領域に実装することができる。

[0108] 図９は、実施形態に係るデータ伝達装置がエンジンの燃料噴射系に適用された時のデータ伝達手順を示すフローチャートである。なお、図９では、図１のエンジンの燃料噴射系に図２の（ａ）のマスフロー方式が適用された場合を示した。

[0109] 図９において、ＣＰＵ１００には、リアルタイムＯＳ（Ｏｐｅｒａｔｉｎｇ Ｓｙｓｔｅｍ）９００が実装されている。ＣＰＵ１００は、このＯＳ９００の指揮下のもとに５ｍｓと１０ｍｓのタスクを周期的に実行することができる。ＯＳ９００は、図１の実行コード１０１に実装することができる。

なお、５ｍｓというタスクの実行時間は、吸入空気量Ｑの計算９０１に用いられるデジタルフィルタのサンプリング周期に基づいて設定することができる。１０ｍｓというタスクの実行時間は、エンジン回転数Ｎｅの計算９０２に用いられるデジタルフィルタのサンプリング周期に基づいて設定することができる。

[0110] ＣＰＵ１００は、主にセンサからの入力処理を担当する。５ｍｓのタスクでは、吸入空気量Ｑの計算９０１を行う。１０ｍｓのタスクでは、エンジン回転数Ｎｅの計算９０２を行う。この計算９０１、９０２の結果として、｛Ｑ，Ｎｅ｝のデータセットの瞬時値が確定する。吸入空気量Ｑの計算９０１およびエンジン回転数Ｎｅの計算９０２は、実行コード１０１を用いてローカルＲＡＭ１０２のメモリ資源を利用しつつ行われる。

[0111] 次に、ＣＰＵ１００は、図８の書き込み手続きＳ８００を実行することにより、共有ＲＡＭ１２０に設けられたバンクメモリＢ［１］～Ｂ［３］に最新の｛Ｑ，Ｎｅ｝のデータセットを書き込むことができる（９０３）。

[0112] この時、ＣＰＵ１００は、書き込み手続きＳ８００においてＲｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０およびＰｕｔ＿Ｂａｎｋ関数ｆ８３０を呼び出すことにより、カウント値ＵＢＣ［１］～ＵＢＣ［３］およびレジスタ６１３、６１４の値の参照や更新を行いつつ、書き込みバンクメモリＢ［１］～Ｂ［３］を確保したり、バンクメモリＢ［１］～Ｂ［３］を公開したりすることができる。Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０およびＰｕｔ＿Ｂａｎｋ関数ｆ８３０の処理は、実行コード１０１を用いて、カウント値ＵＢＣ［１］～ＵＢＣ［３］およびレジスタ６１３、６１４の値の参照や更新を行いつつ、共有ＲＡＭ１２０上のスクラッチパッド・メモリ６１１に対して行われる。最新の｛Ｑ，Ｎｅ｝のデータセットの準備時間および送信時間（書き込み手続きＳ８００にかかる時間）を延長しても、他の読み出しコアの動作遅延を引き起こすことはない。

[0113] ＣＰＵ１１０には、リアルタイムＯＳ９１０が実装されている。ＣＰＵ１１０は、このＯＳ９１０の指揮下のもとに１５ｍｓのタスクを周期的に実行

することができる。ＯＳ９１０は、図１の実行コード１１１に実装することができる。なお、１５ｍｓというタスクの実行時間は、出力アクチュエータの動作時間（要求される時間分解能）に基づいて設定することができる。

[0114] ＣＰＵ１１０は、主に演算と出力アクチュエータの指示値計算を担当する。１５ｍｓのタスクでは、スクラッチパッド・メモリ６１１の読み出しと、燃料噴射量ｑの計算９１２を行う。燃料噴射量ｑの計算９１２は、実行コード１１１を用いてローカルＲＡＭ１１２のメモリ資源を利用しつつ行われる。

[0115] 燃料噴射量ｑの計算９１２を行うために、ＣＰＵ１１０は、図７の読み出し手続きＳ７００を実行することにより、共有ＲＡＭ１２０に設けられたバンクメモリＢ［１］～Ｂ［３］から最新の｛Ｑ，Ｎｅ｝のデータセットを読み出すことができる（９１１）。

[0116] この時、ＣＰＵ１１０は、読み出し手続きＳ７００においてＧｅｔ＿Ｂａｎｋ関数ｆ７１０およびＲｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０を呼び出すことにより、カウント値ＵＢＣ［１］～ＵＢＣ［３］およびレジスタ６１３、６１４の値の参照や更新を行いつつ、読み出しバンクメモリＢ［１］～Ｂ［３］を確保したり、バンクメモリＢ［１］～Ｂ［３］を返却したりすることができる。Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０およびＲｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０の処理は、実行コード１１１を用いて、カウント値ＵＢＣ［１］～ＵＢＣ［３］およびレジスタ６１３、６１４の値の参照や更新を行いつつ、共有ＲＡＭ１２０上のスクラッチパッド・メモリ６１１に対して行われる。

[0117] Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０の処理と、Ｒｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０の処理との間は、バンクメモリＢ［１］～Ｂ［３］の内容がそのまま保持されるので、そのバンクメモリＢ［１］～Ｂ［３］上の任意の変数を任意の回数だけ読み出すことができる。

また、この保持時間（読み出し手続きＳ７００にかかる時間）をどれだけ延長しても、他の書き込みコアおよび読み出しコアの動作遅延を引き起こす

ことはない。

[0118] 図１０は、実施形態に係るデータ伝達装置がエンジンの燃料噴射系に適用された時のデータ伝達方法を時系列的に示すシーケンス図である。

図１０において、ＣＰＵ１００のＱ計算９０１では、吸入空気量Q_1～Q_{10}が５ｍｓごとに計算される。ＣＰＵ１００のＮｅ計算９０２では、エンジン回転数Ne_1～Ne_5が１０ｍｓごとに計算される。バンク出力９０３では、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０およびＰｕｔ＿Ｂａｎｋ関数ｆ８３０が１０ｍｓごとに呼び出され、バンクメモリＢ［１］～Ｂ［３］に｛Ｑ，Ｎｅ｝のデータセットが書き込まれる。

[0119] スクラッチパッド・メモリ６１１のバンクメモリＢ［１］～Ｂ［３］に最新の｛Ｑ，Ｎｅ｝のデータセットが書き込まれると、この最新の｛Ｑ，Ｎｅ｝のデータセットが書き込まれたバンクメモリＢ［１］～Ｂ［３］のバンク番号＃１～＃３がレジスタ６１３に登録される。図１０ではレジスタ６１３がＬＩＢＲと表現されている。この最新データによるバンク内容のデータ変化を１０００で示す。

[0120] Ｑ計算９０１は５ｍｓごとであり、Ｎｅ計算９０２は１０ｍｓごとである。このため、バンク出力９０３からは、Ｑ計算９０１による吸入空気量Q_1～Q_{10}は２回に１回読み飛ばしが起こっているように見える。しかしながら、実際には、Ｑ計算９０１の出力はデジタルフィルタの出力値である。このため、デジタルフィルタのサンプリングタイムでＱ計算９０１の計算周期が決定され、Ｑ計算９０１による吸入空気量Q_2、Q_4、Q_6、Q_8、Q_{10}はバンク出力９０３として採用されなくても、バンク出力９０３として採用される吸入空気量Q_1、Q_3、Q_5、Q_7、Q_9に影響を与えるサンプル値である。このため、吸入空気量Q_2、Q_4、Q_6、Q_8、Q_{10}のＱ計算９０１は無駄ではない。

[0121] バンク入力９１１では、Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０およびＲｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０が１５ｍｓごとに呼び出され、バンクメモリＢ［１］～Ｂ［３］から最新の｛Ｑ，Ｎｅ｝のデータセットが読み出される。ＣＰＵ１１０のｑ計算９１２では、最新の｛Ｑ，Ｎｅ｝のデータセットに基

づいて燃料噴射量q_1～q_3が１５ｍｓごとに計算される。

[0122] 例えば、ＣＰＵ１００が吸入空気量Q_1およびエンジン回転数Ne_1を計算した時にバンクメモリＢ［３］がフリーバンクであるものとすると、ＣＰＵ１００は、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０にてバンクメモリＢ［３］を確保し、$\{Q_1, Ne_1\}$のデータセットをバンクメモリＢ［３］に書き込む。そして、Ｐｕｔ＿Ｂａｎｋ関数ｆ８３０にてバンクメモリＢ［３］を公開する。この時、最新のデータが登録されたバンクメモリＢ［３］のバンク番号＃３がＬＩＢＲ（レジスタ６１３）で示される。

[0123] 次に、ＣＰＵ１１０は、Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０にてバンクメモリＢ［３］を確保し、$\{Q_1, Ne_1\}$のデータセットをバンクメモリＢ［３］から読み出す。そして、ＣＰＵ１１０は、$\{Q_1, Ne_1\}$のデータセットに基づいて燃料噴射量q_1を計算し、Ｒｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０にてバンクメモリＢ［３］を返却する。

[0124] ＣＰＵ１１０がバンクメモリＢ［３］を返却する前に、ＣＰＵ１００は吸入空気量Q_3およびエンジン回転数Ne_2を計算する。この時、バンクメモリＢ［２］がフリーバンクであるものとすると、ＣＰＵ１００は、バンクメモリＢ［３］の返却を待つことなく、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０にてバンクメモリＢ［２］を確保し、$\{Q_3, Ne_2\}$のデータセットをバンクメモリＢ［２］に書き込む。そして、Ｐｕｔ＿Ｂａｎｋ関数ｆ８３０にてバンクメモリＢ［２］を公開する。この時、最新のデータが登録されたバンクメモリＢ［２］のバンク番号＃２がＬＩＢＲ（レジスタ６１３）で示される。

[0125] 次に、ＣＰＵ１１０は、Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０にてバンクメモリＢ［２］を確保し、$\{Q_3, Ne_2\}$のデータセットをバンクメモリＢ［２］から読み出す。そして、ＣＰＵ１１０は、$\{Q_3, Ne_2\}$のデータセットに基づいて燃料噴射量q_2を計算し、Ｒｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０にてバンクメモリＢ［２］を返却する。

[0126] ＣＰＵ１１０がバンクメモリＢ［２］を返却する前に、ＣＰＵ１００は吸

入空気量Q_5およびエンジン回転数Ne_3を計算する。この時、バンクメモリＢ［１］がフリーバンクであるものとすると、ＣＰＵ１００は、バンクメモリＢ［２］の返却を待つことなく、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０にてバンクメモリＢ［１］を確保し、｛Q_5，Ne_3｝のデータセットをバンクメモリＢ［１］に書き込む。そして、Ｐｕｔ＿Ｂａｎｋ関数ｆ８３０にてバンクメモリＢ［１］を公開する。この時、最新のデータが登録されたバンクメモリＢ［１］のバンク番号＃１がＬＩＢＲ（レジスタ６１３）で示される。

[0127] この時、ＣＰＵ１１０は、最新のデータが登録されたバンクメモリＢ［１］がＬＩＢＲで示されている場合においても、ＣＰＵ１１０は、バンクメモリＢ［２］を返却する前に、それより古い｛Q_3，Ne_2｝のデータセットに基づいて燃料噴射量q_2の計算を続行することができる。

[0128] 次に、ＣＰＵ１００が吸入空気量Q_7およびエンジン回転数Ne_4を計算した時にバンクメモリＢ［３］がフリーバンクであるものとすると、ＣＰＵ１００は、Ｒｅｓｅｒｖｅ＿Ｂａｎｋ関数ｆ８１０にてバンクメモリＢ［３］を確保し、｛Q_7，Ne_4｝のデータセットをバンクメモリＢ［３］に書き込む。そして、Ｐｕｔ＿Ｂａｎｋ関数ｆ８３０にてバンクメモリＢ［３］を公開する。この時、最新のデータが登録されたバンクメモリＢ［３］のバンク番号＃３がＬＩＢＲ（レジスタ６１３）で示される。

[0129] 次に、ＣＰＵ１１０は、Ｇｅｔ＿Ｂａｎｋ関数ｆ７１０にてバンクメモリＢ［３］を確保し、｛Q_7，Ne_4｝のデータセットをバンクメモリＢ［３］から読み出す。そして、ＣＰＵ１１０は、｛Q_7，Ne_4｝のデータセットに基づいて燃料噴射量q_3を計算し、Ｒｅｌｅａｓｅ＿Ｂａｎｋ関数ｆ７３０にてバンクメモリＢ［３］を返却する。

[0130] この時、バンクメモリＢ［１］に書き込まれた｛Q_5，Ne_3｝のデータセットは、バンクメモリＢ［３］に書き込まれた｛Q_7，Ne_4｝のデータセットより古くなっている。このため、ＣＰＵ１１０は、バンクメモリＢ［１］に書き込まれた｛Q_5，Ne_3｝のデータセットを読み出すことなく、｛Q_7，

$Ｎｅ_4$｝のデータセットをバンクメモリＢ［３］から読み出すことができる。この時、｛$Ｑ_5$，$Ｎｅ_3$｝のデータセットの読み飛ばし１００１を行うことができる。

[0131] すなわち、ＣＰＵ１００がスクラッチパッド・メモリ６１１に対して行うバンク出力９０３は１０ｍｓの周期で実行される。スクラッチパッド・メモリ６１１からＣＰＵ１１０へのバンク入力９１１は１５ｍｓの周期で実行される。この時、スクラッチパッド・メモリ６１１のデータ変化１０００に着目すると、バンク出力９０３が３回だけ行われるごとに、バンク入力９１１に対するデータセットの読み飛ばし１００１を１回だけ行うことができる。

[0132] このように、送り手と受け手が非同期でデータの授受を可能とすることで、受け手は要求すれば少なくとも最新のデータセットを常時取得することができる。また、送り手は受け手の都合を考慮せずに最新値の更新に専念できる。したがって、データの送り手と受け手とが疎結合であり、一方の動作のタイミングに他方の動作が引きずられることがないので、ロバスト性を確保することができる。

[0133] 図１１（ａ）は、実施形態に係るデータ伝達装置の書き込みコアとスクラッチパッド・メモリとの占有関係を示す図である。

[0134] 図１１（ａ）において、４個のスクラッチパッド・メモリＰ１～Ｐ４と、これらのスクラッチパッド・メモリＰ１～Ｐ４にアクセス可能な４個のコアＣ１～Ｃ４が設けられている。この時、スクラッチパッド・メモリＰ１～Ｐ４の個数と、コアＣ１～Ｃ４の個数とは等しくすることができる。各スクラッチパッド・メモリＰ１～Ｐ４には、５個のバンクメモリを設けることができる。例えば、スクラッチパッド・メモリＰ１は、図６のスクラッチパッド・メモリ６１１と同様に構成することができる。各スクラッチパッド・メモリＰ１～Ｐ４のバンク数は、総コア数＋１に設定することができる。スクラッチパッド・メモリＰ１～Ｐ４は、共有メモリ６１０に配置することができる。

[0135] コアＣ１～Ｃ４は、書き込みにおよび読み出し使われるＣＰＵに設けるこ

とができる。この時、各コアＣ１～Ｃ４は、アクセスするスクラッチパッド・メモリＰ１～Ｐ４に応じて書き込みコアとして動作したり、読み出しコアとして動作したりすることができる。

[0136] ここで、コアＣ１は、スクラッチパッド・メモリＰ１に対しては書き込みコアとして動作し、スクラッチパッド・メモリＰ２～Ｐ４に対しては読み出しコアとして動作することができる。コアＣ２は、スクラッチパッド・メモリＰ２に対しては書き込みコアとして動作し、スクラッチパッド・メモリＰ１、Ｐ３、Ｐ４に対しては読み出しコアとして動作することができる。コアＣ３は、スクラッチパッド・メモリＰ３に対しては書き込みコアとして動作し、スクラッチパッド・メモリＰ１、Ｐ２、Ｐ４に対しては読み出しコアとして動作することができる。コアＣ４は、スクラッチパッド・メモリＰ４に対しては書き込みコアとして動作し、スクラッチパッド・メモリＰ１～Ｐ３に対しては読み出しコアとして動作することができる。

[0137] すなわち、各コアＣ１～Ｃ４は、自分がライト・オーナーであるスクラッチパッド・メモリＰ１～Ｐ４に関しては書き込みコア６０１であり、他のコアＣ１～Ｃ４がライト・オーナーであるスクラッチパッド・メモリＰ１～Ｐ４に関しては読み出しコア６０２－１～６０２－３の一つである関係が成立する。

[0138] 図１１（ｂ）は、実施形態に係るデータ伝達装置の各コアごとのスクラッチパッド・メモリへの操作関数の割り当て方法を示す図である。これらの操作関数は、各コアのローカルな実行コード格納領域に格納することができる。

[0139] 自分がライト・オーナーであるスクラッチパッド・メモリに関しては書き込み系関数Ｒｅｓｅｒｖｅ__Ｂａｎｋ__Ｐ（）およびＰｕｔ__Ｂａｎｋ__Ｐ（）が実装され、その他のスクラッチパッド・メモリに関しては読み出し系関数Ｇｅｔ__Ｂａｎｋ__Ｐ（）およびＲｅｌｅａｓｅ__Ｂａｎｋ__Ｐ（）が実装される。ただし、Ｐはスクラッチパッド・メモリＰ１～Ｐ４のメモリ番号である。

[0140] ここで、スクラッチパッド・メモリＰ１～Ｐ４を複数設け、スクラッチパッド・メモリＰ１～Ｐ４の個数とコアＣ１～Ｃ４の個数とを等しくすることにより、一つのスクラッチパッド・メモリに書き込みコアが一つしか割り当てられないシステムを構成すると、複数のコアＣ１～Ｃ４が同時に書き込み動作を実行することが可能となる。

[0141] また、自分がライト・オーナーであるスクラッチパッド・メモリＰ１～Ｐ４以外についての書き込み系関数Ｒｅｓｅｒｖｅ＿Ｂａｎｋ＿Ｐ（）およびＰｕｔ＿Ｂａｎｋ＿Ｐ（）を保持しないことにより、ライト・オーナーでないスクラッチパッド・メモリに間違って書き込むことを防ぐことができる。

[0142] また、自分がライト・オーナーであるスクラッチパッド・メモリＰ１～Ｐ４については読み出し系関数Ｇｅｔ＿Ｂａｎｋ＿Ｐ（）およびＲｅｌｅａｓｅ＿Ｂａｎｋ＿Ｐ（）を保持しないことによって、誤読み出しを防止する（読み出しバンクの勘違いを防ぐ）ことができる。

[0143] 以上のように、上述した実施形態によれば、スクラッチパッド・メモリＰ１～Ｐ４を用いてコアＣ１～Ｃ４間でデータ伝達可能とすることで、マルチコアにおいても、非同期でコア間のデータの授受を図ることができる。スクラッチパッド・メモリＰ１～Ｐ４の操作においては、スピンロック方式を用いる必要がなくなり、ロック待ち時間なしにデータ伝達の高速化を図ることができる。

[0144] また、例示したコア数より、コア数をもっと増加させたマルチコアに関しても、スクラッチパッド・メモリを増加させるだけで対応可能であり、実行遅延の発生を防止することができる。

[0145] スクラッチパッド・メモリ内のバンク数を設計する指針として、スクラッチパッド・メモリにアクセス可能なコアの個数がＮ個（そのうち１個がライト・オーナーである）であるとすると、合計Ｎ＋１個のバンクメモリを設けることができる。

[0146] 図１２は、実施形態に係るデータ伝達装置のスクラッチパッド・メモリのバンク数とコア数との関係を示す図である。なお、図１２は、図６の書き込

みコア６０１と読み出しコア６０２－１、６０２－２、６０２－３とで、バンクメモリＢ［１］～Ｂ［５］が最大限使用されている状態を示した。

[0147] 図１２において、読み出しコア６０２－１、６０２－２、６０２－３からバンクメモリＢ［３］～Ｂ［５］に対してＧｅｔ＿Ｂａｎｋ（）関数がそれぞれ呼び出され、読み出しコア６０２－１、６０２－２、６０２－３がそれぞれ別個のバンクメモリＢ［３］～Ｂ［５］を使用中である。この時、読み出しだけで最大で（Ｎ－１）個のバンクメモリＢ［３］～Ｂ［５］が占有される。

[0148] この状態で、書き込みコア６０１が連続して二度書きを行う場合を想定する。すなわち、読み出しコア６０２－１、６０２－２、６０２－３がそれぞれ別個のバンクメモリＢ［３］～Ｂ［５］を使用中に、書き込みコア６０１によるバンクメモリＢ［２］への書き込みが終了したものとする。この時、レジスタ６１３には、最新のデータが書き込まれたバンクメモリＢ［２］のバンク番号＃２が登録される。

[0149] ここで、コアの個数よりも、バンクメモリの個数を１個だけ多くすることにより、バンクメモリＢ［２］に最新のデータが書き込まれ、かつ読み出しコア６０２－１、６０２－２、６０２－３がバンクメモリＢ［３］～Ｂ［５］を使用中である場合においても、バンクメモリＢ［１］をフリーバンクとして登録することがきる。

[0150] このため、書き込みコア６０１は、バンクメモリＢ［２］への書き込みの終了後に、読み出しコア６０２－１、６０２－２、６０２－３からのバンクメモリＢ［３］～Ｂ［５］の返却を待つことなく、バンクメモリＢ［１］に対してＲｅｓｅｒｖｅ＿Ｂａｎｋ（）関数を呼び出し、バンクメモリＢ［１］に書き込みを行うことができる。バンクメモリＢ［１］に書き込み中に、読み出しコア６０２－１、６０２－２、６０２－３がバンクメモリＢ［３］～Ｂ［５］のいずれかを返却し、新たな読み出し要求をした場合は、最新のデータバンクＢ［２］を提供することができる。

[0151] この時、前回と今回の書き込み用に１バンクずつ必要なため、書き込み用

に２バンクが必要となる。すなわち、読み出しだけで最大で（Ｎ－１）個のバンクが必要なので、全体として合計で（Ｎ－１）＋２＝（Ｎ＋１）バンクが必要であることが分かる。

[0152] 書き込みコア６０１がバンクメモリＢ［１］に書き込んでいる間に、新たに読み出し要求が来た時に備えて、バンクメモリＢ［１］への書き込みが終了し、バンクメモリＢ［１］が公開される（Ｐｕｔ＿Ｂａｎｋ（）関数が呼び出される）まで、バンクメモリＢ［２］の状態を保持する。これ以降、読み出しコア６０２－１、６０２－２、６０２－３のバンクメモリＢ［３］～Ｂ［５］の占有状態に変化がなく、新たな書き込み要求が発生した場合には、レジスタ６１３にて示されるバンクメモリと、書き込みコア６０１がＲｅｓｅｒｖ＿Ｂａｎｋ（）関数で占有するバンクメモリとを交互に切り替えて、読み出し用最新データの保持と書き込み要求の実行とを両立させることができる。

[0153] 以上述べたように、上述した実施形態によれば、ロック待ち時間なく、かつデータセットの同時性を破綻させることなく、コア間のデータ伝達を行うことができる。また、コア数を増加させたマルチコアにおいても、バンク数を増やすことで、２コアの時とほとんど同等な伝達時間を維持しつつ、コア間のデータ伝達を行うことができ、遅延時間増加によるマルチコアシステムの破綻を防止することができる。

[0154] 以上、本発明に係る実施形態について具体的に説明したが、本発明は、上述した実施形態に限定されるものではなく、その要旨を逸脱しない範囲で種々変更可能である。特に、ＣＰＵのコア数は、実施形態で述べたような２個および４個に限定するものではなく、Ｎ（Ｎは２以上の整数）個のマルチコアであってもよい。

[0155] また、上記各構成、機能、処理部などは、それらの全部または一部を、例えば集積回路で設計することによりハードウェアとして実現することもできるし、プロセッサがそれぞれの機能を実現するプログラムを実行することによりソフトウェアとして実現することもできるし、集積回路と密接に関連し

たデバイス制御言語もしくはファームウェアとして実現することもできる。

符号の説明

[0156] １００、１１０…ＣＰＵ、１０１、１１１…実行コード、１０２、１１２…ローカルＲＡＭ、１２０…共有ＲＡＭ、６０１…書き込みコア、６０２－１、６０２－２、６０２－３…読み出しコア、６１０…共有メモリ、６１１…スクラッチパッド・メモリ、６１２…カウンタ配列領域、ＵＢＣ［１］～ＵＢＣ［５］…カウント値、６１３、６１４…レジスタ、Ｂ［１］～Ｂ［５］…バンクメモリ

請求の範囲

[請求項1] Ｍ（Ｍは２以上の整数）個のバンクメモリと、前記バンクメモリにアクセス可能なＮ（Ｎは２以上の整数）個のプロセッサとが設けられた車載マルチコア制御用データ伝達装置であって、

前記Ｍ個のバンクメモリが配置され、前記Ｎ個のプロセッサからアクセス可能な共有メモリと、

前記バンクメモリを特定する識別情報に基づいて、前記プロセッサから前記バンクメモリへのアクセスを制御するアクセス制御部とを備える車載マルチコア制御用データ伝達装置。

[請求項2] 前記アクセス制御部は、

書き込みが行われるバンクメモリには、書き込みに使われる１個のプロセッサのみを割り当て、

読み出しが行われるバンクメモリには、読み出しに使われるＫ（Ｋは１以上Ｎ－１以下の整数）個のプロセッサを割り当て、

書き込みが行われているバンクメモリと読み出しが行われているバンクメモリとが同一にならないように前記バンクメモリへのアクセスを排他的に制御する請求項１に記載の車載マルチコア制御用データ伝達装置。

[請求項3] 前記アクセス制御部は、

前記プロセッサによる読み出し時には、最後に書き込みが行われたバンクメモリの識別情報を返し、

前記プロセッサによる書き込み時には、最後に書き込みが行われたバンクメモリ以外で、かついずれのプロセッサによる読み出しにも使用されていないバンクメモリの識別情報を返す請求項２に記載の車載マルチコア制御用データ伝達装置。

[請求項4] 前記プロセッサによる書き込みが行われるバンクメモリは、前記プロセッサに前記バンクメモリが割り当てられる前に、最後に書き込みが行われたバンクメモリの内容がコピーされる請求項２に記載の車載

マルチコア制御用データ伝達装置。

[請求項5] 書き込みに使われるプロセッサの個数は１、読み出しに使われるプロセッサの個数はＮ－１、前記バンクメモリの個数ＭはＮ＋１に設定される請求項２に記載の車載マルチコア制御用データ伝達装置。

[請求項6] 前記バンクメモリをＭ個のごとにまとめたＮ個のスクラッチパッド・メモリを備え、

書き込みに使われるＮ個のプロセッサと前記Ｎ個のスクラッチパッド・メモリとは１対１に対応づけられている請求項１に記載の車載マルチコア制御用データ伝達装置。

[請求項7] 前記スクラッチパッド・メモリに対して行われる書き込み操作および読み出し操作を実現する実行コードおよびこの実行コードを含む関数は、各プロセッサに対してローカルに設けられた実行コード格納領域に保持される請求項６に記載の車載マルチコア制御用データ伝達装置。

[請求項8] 前記スクラッチパッド・メモリに対して対応づけられた書き込みに使われるプロセッサに関しては書き込みコードおよび書き込み関数のみを実装し、前記スクラッチパッド・メモリに対して対応づけられた読み出しに使われるプロセッサに関しては読み出しコードおよび読み出し関数のみを実装する請求項７に記載の車載マルチコア制御用データ伝達装置。

[請求項9] 前記共有メモリに配置されたカウンタ配列領域と第１レジスタと第２レジスタを備え、

前記カウンタ配列領域は、読み出しに使用されるプロセッサから前記バンクメモリへのアクセス数を前記バンクメモリごとに記憶し、

前記第１レジスタは、最後に書き込みが行われたバンクメモリを特定する識別情報を記憶し、

前記第２レジスタは、前記カウンタ配列領域および前記第１レジスタの参照結果に基づいて、最後に書き込みが行われたバンクメモリ以

外で、かつどのプロセッサからもアクセスされていないバンクメモリを特定する識別情報を記憶し、

前記アクセス制御部は、前記カウンタ配列領域に記憶されたアクセス数および前記第１レジスタに記憶された識別情報および前記第２レジスタに記憶された識別情報に基づいて、前記プロセッサから前記バンクメモリへのアクセスを制御する請求項１に記載の車載マルチコア制御用データ伝達装置。

[請求項10] Ｍ（Ｍは２以上の整数）個のバンクメモリと、前記バンクメモリにアクセス可能なＮ（Ｎは２以上の整数）個のプロセッサとが設けられた電子制御装置であって、

前記Ｍ個のバンクメモリが配置され、前記Ｎ個のプロセッサからアクセス可能な共有メモリと、

前記バンクメモリを特定する識別情報に基づいて、前記プロセッサから前記バンクメモリへのアクセスを制御するアクセス制御部とを備え、

前記アクセス制御部は、

書き込みが行われるバンクメモリには、書き込みに使われる１個のプロセッサのみを割り当て、

読み出しが行われるバンクメモリには、読み出しに使われるＫ（Ｋは１以上Ｎ－１以下の整数）個のプロセッサを割り当て、

書き込みが行われているバンクメモリと読み出しが行われているバンクメモリとが同一にならないように前記バンクメモリへのアクセスを排他的に制御し、

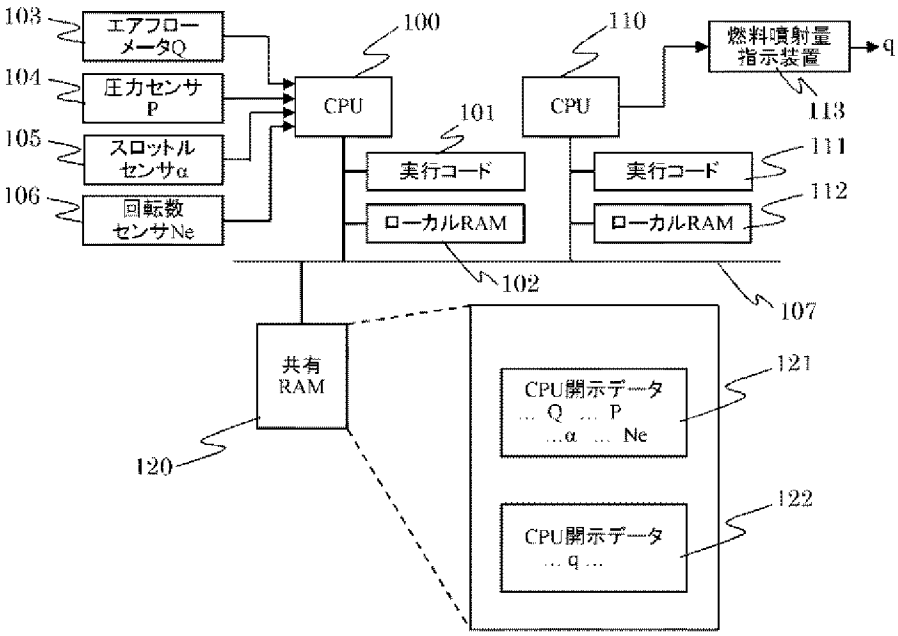
前記書き込みに使われるプロセッサは、車載制御の制御量の算出に用いられる1以上複数の入力値を計算し、その入力値を前記バンクメモリに書き込み、

前記読み出しに使われるプロセッサは、前記書き込みが行われたバンクメモリから前記入力値を読み出し、前記入力値に基づいて車載制

御の制御量を算出する電子制御装置。

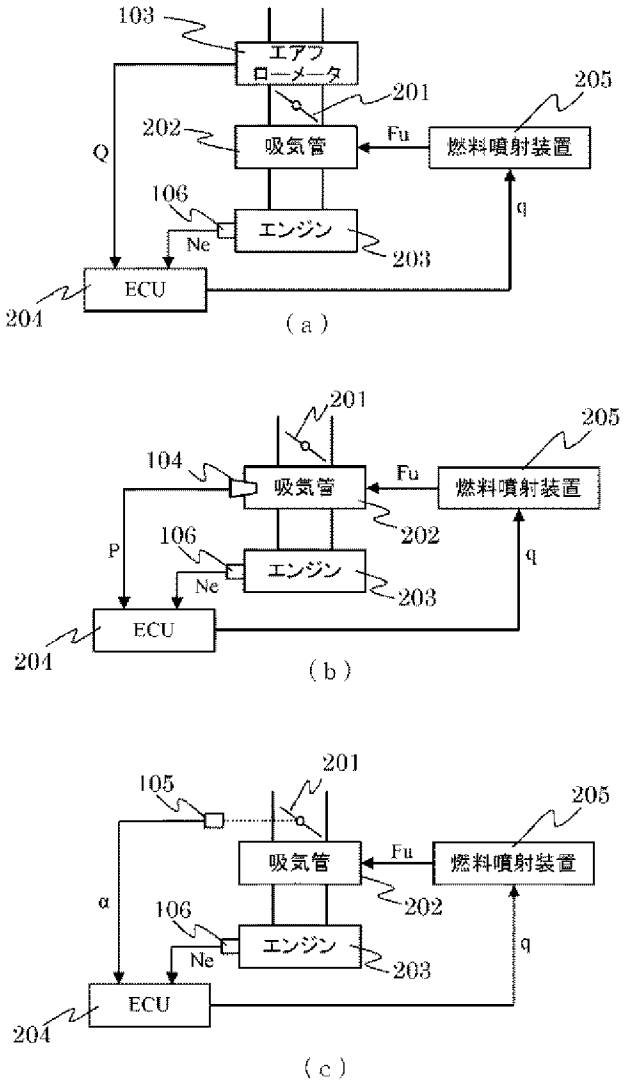
[図1]

図 1


103
エアフローメータQ
104
圧力センサP
105
スロットルセンサα
106
回転数センサNe
100
CPU
101
実行コード
ローカルRAM
102
110
CPU
実行コード
ローカルRAM
111
112
燃料噴射量指示装置
q
113
107
共有RAM
120
CPU関示データ
… Q … P
…α … Ne
121
CPU関示データ
… q …
122

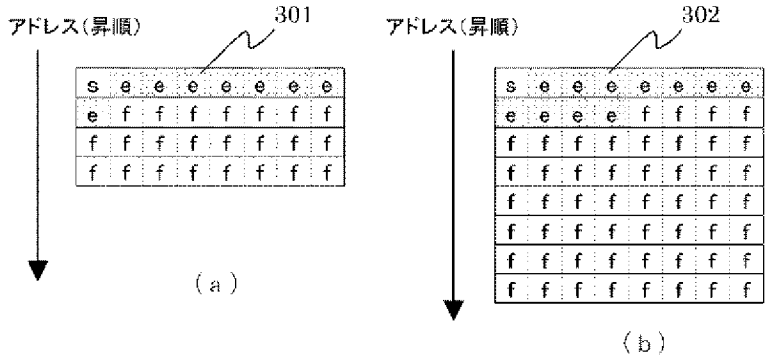
[図2]

図２


103
エアフローメータ
201
205
202
吸気管
Fu
燃料噴射装置
Q
106
q
Ne
エンジン
203
ECU
204
（ａ）
201
205
104
吸気管
Fu
燃料噴射装置
202
P
106
q
エンジン
Ne
203
ECU
204
（ｂ）
201
105
205
吸気管
Fu
燃料噴射装置
202
α
106
q
Ne
エンジン
203
ECU
204
（ｃ）

[図3]

図 3

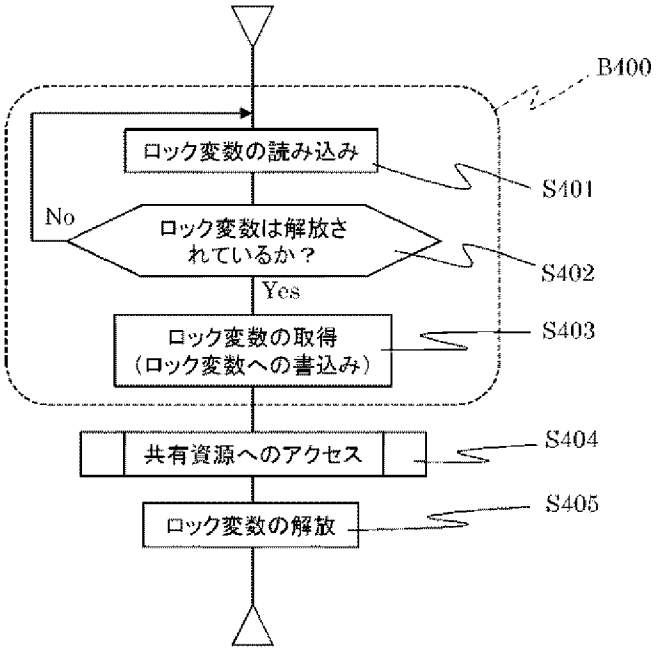


精度	sign:符号ビット	exponent:指数部	fraction:仮数部	総バイト数
単精度	1 bit	8 bits	23 bits	4 bytes
倍精度	1 bit	11 bits	52 bits	8 bytes

(c)

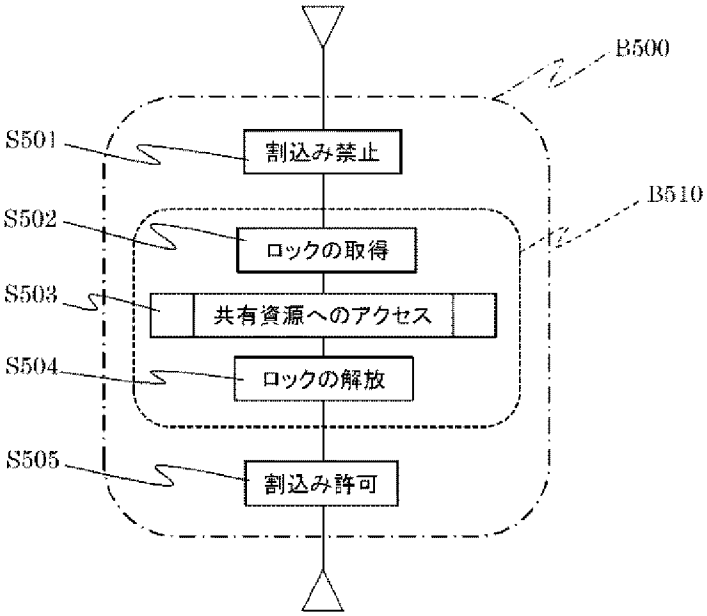
[図4]

図4


B400
ロック変数の読み込み
S401
No
ロック変数は解放されているか？
S402
Yes
ロック変数の取得
（ロック変数への書込み）
S403
共有資源へのアクセス
S404
ロック変数の解放
S405

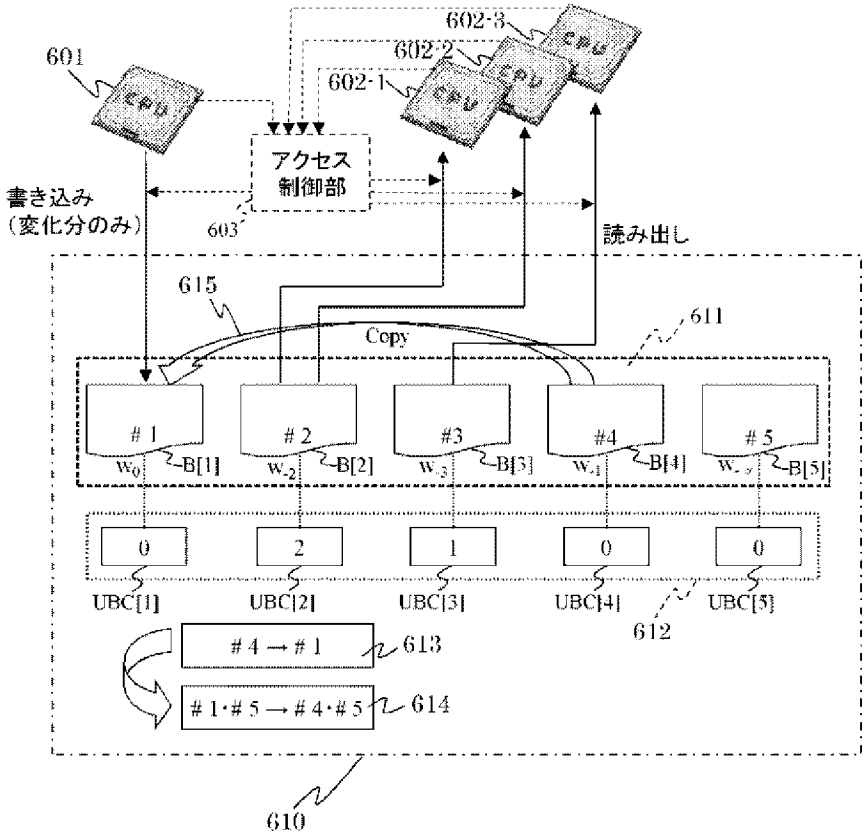
[図5]

図 5


B500
S501
割込み禁止
B510
S502
ロックの取得
S503
共有資源へのアクセス
S504
ロックの解放
S505
割込み許可

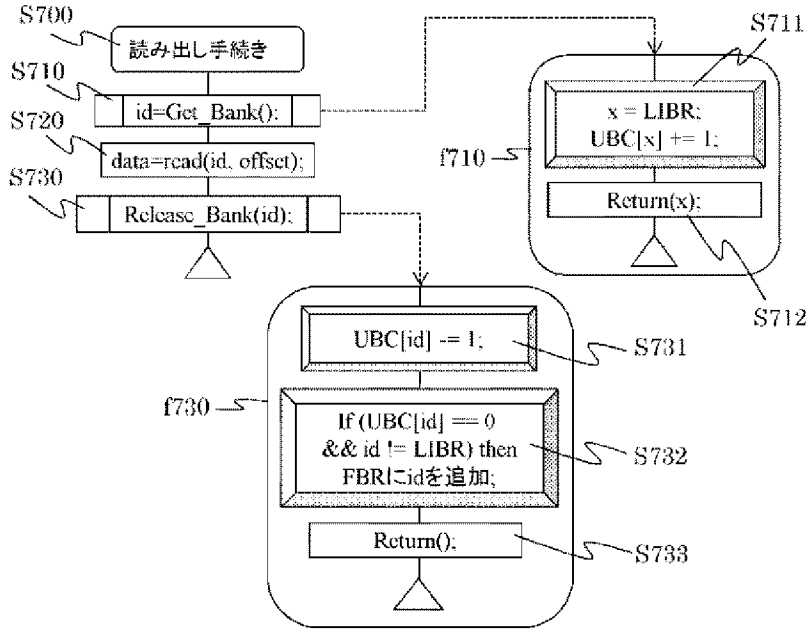
[図6]

図 6


601
602-3
602-2
602-1
CPU
アクセス
制御部
603
書き込み
（変化分のみ）
読み出し
615
Copy
611
1
2
#3
#4
5
B[1]
B[2]
B[3]
B[4]
B[5]
0
2
1
0
0
UBC[1]
UBC[2]
UBC[3]
UBC[4]
UBC[5]
612
4 → # 1
613
1・# 5 → # 4・# 5
614
610

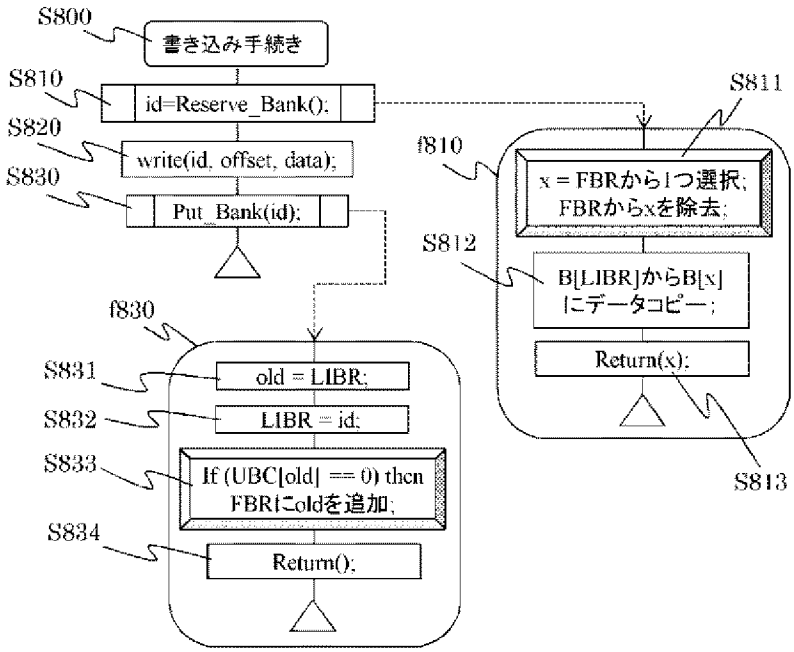
[図7]

図 7


S700
読み出し手続き
S710
id=Get_Bank();
S720
data=read(id, offset);
S730
Release_Bank(id);
S711
x = LIBR;
UBC[x] += 1;
f710
Return(x);
S712
UBC[id] -= 1;
S731
f730
If (UBC[id] == 0
&& id != LIBR) then
FBRにidを追加;
S732
Return();
S733

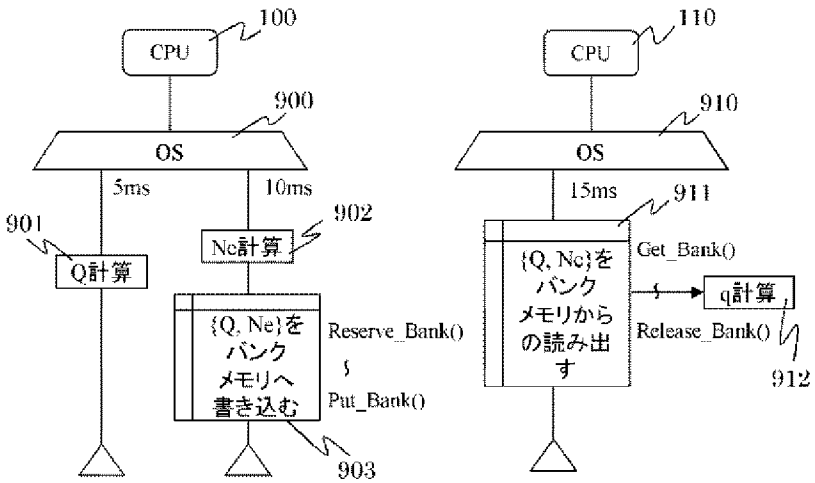
[図8]

図８


S800
書き込み手続き
S810
id=Reserve_Bank();
S820
write(id, offset, data);
S830
Put_Bank(id);
S811
f810
x = FBRから1つ選択;
FBRからxを除去;
S812
B[LIBR]からB[x]
にデータコピー;
Return(x);
S813
f830
S831
old = LIBR;
S832
LIBR = id;
S833
If (UBC[old] == 0) then
FBRにoldを追加;
S834
Return();

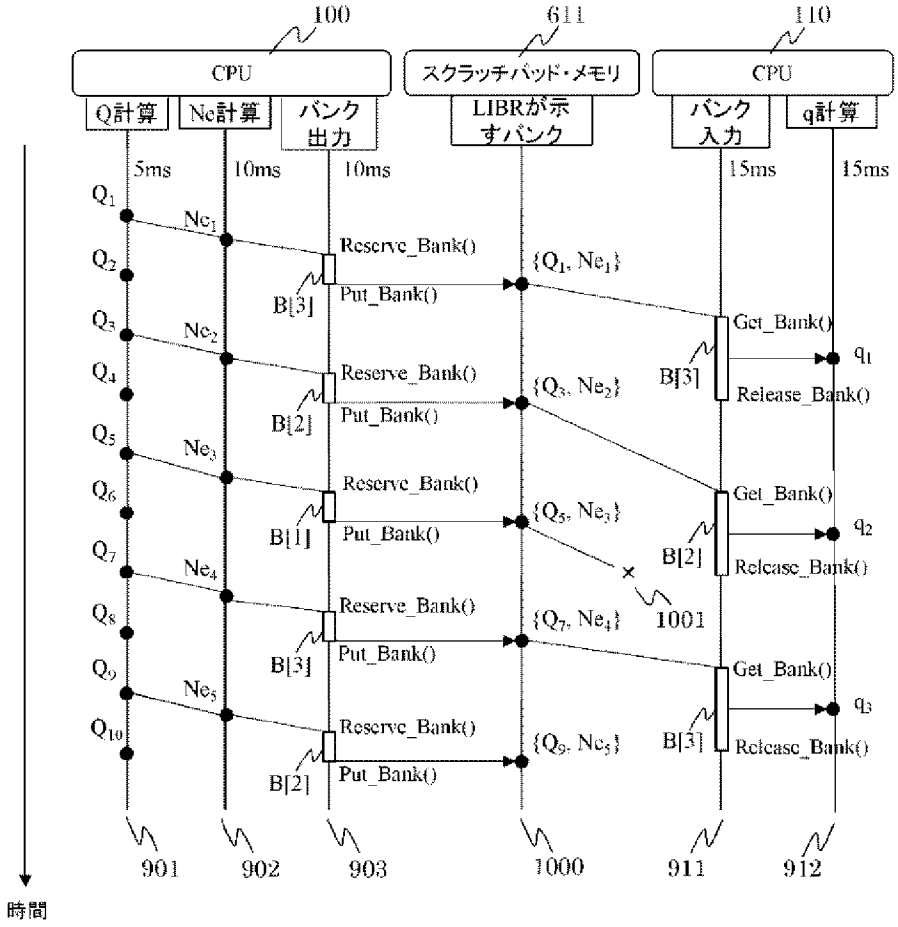
[図9]

図 9


100
CPU
900
OS
5ms
10ms
901
Q計算
902
Ne計算
{Q, Ne}を
バンク
メモリへ
書き込む
Reserve_Bank()
Put_Bank()
903
110
CPU
910
OS
15ms
911
{Q, Ne}を
バンク
メモリから
の読み出
す
Get_Bank()
q計算
Release_Bank()
912

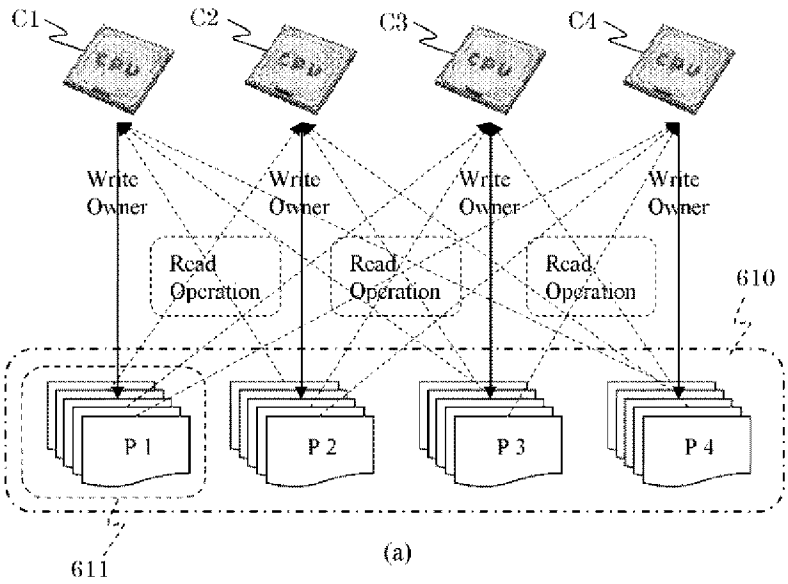
[図10]

図１０


100
611
110
CPU
スクラッチパッド・メモリ
CPU
Q計算
Ne計算
バンク出力
LIBRが示すバンク
バンク入力
q計算
5ms
10ms
10ms
15ms
15ms
Reserve_Bank()
Put_Bank()
Get_Bank()
Release_Bank()
Release_Bank()
1001
901
902
903
1000
911
912
時間

[図11]

図１１



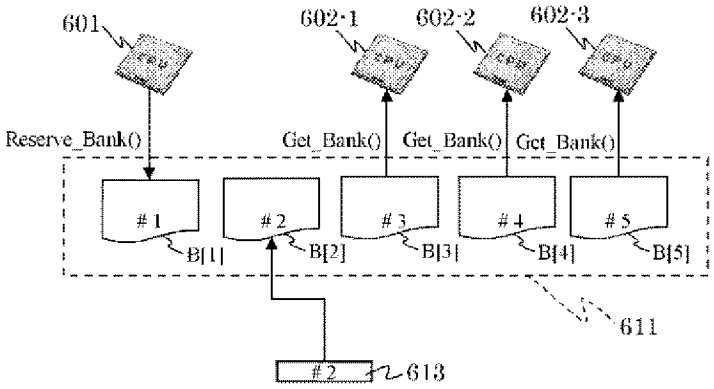
(a)

スクラッチパッド・メモリ	コア保有関数			
	コアC1	コアC2	コアC3	コアC4
P1	Reserve_Bank_1() Put_Bank_1()	Get_Bank_1() Release_Bank_1()	Get_Bank_1() Release_Bank_1()	Get_Bank_1() Release_Bank_1()
P2	Get_Bank_2() Release_Bank_2()	Reserve_Bank_2() Put_Bank_2()	Get_Bank_2() Release_Bank_2()	Get_Bank_2() Release_Bank_2()
P3	Get_Bank_3() Release_Bank_3()	Get_Bank_3() Release_Bank_3()	Reserve_Bank_3() Put_Bank_3()	Get_Bank_3() Release_Bank_3()
P4	Get_Bank_4() Release_Bank_4()	Get_Bank_4() Release_Bank_4()	Get_Bank_4() Release_Bank_4()	Reserve_Bank_4() Put_Bank_4()

(b)

[図12]

図１２


601
602-1
602-2
602-3
Reserve_Bank()
Get_Bank()
Get_Bank()
Get_Bank()
#1
#2
#3
#4
#5
B[1]
B[2]
B[3]
B[4]
B[5]
611
#2
613

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2018/034137

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl. G06F15/167(2006.01)i, F02D45/00(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)
Int.Cl. G06F15/167, F02D45/00

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

Published examined utility model applications of Japan	1922-1996
Published unexamined utility model applications of Japan	1971-2018
Registered utility model specifications of Japan	1996-2018
Published registered utility model applications of Japan	1994-2018

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.
X Y A	JP 2010-501951 A (ROBERT BOSCH GMBH) 21 January 2010, paragraphs [0037]-[0047], fig. 3-5 & US 2010/0242051 A1, paragraphs [0036]-[0045], fig. 3-5 & WO 2008/095548 A1 & EP 1956484 A1 & CN 101601016 A	1-2, 5, 10 6-8 3-4, 9
Y	JP 2006-301894 A (NEC ELECTRONICS CORP.) 02 November 2006, paragraphs [0016]-[0020], fig. 3 & US 2006/0242257 A1, paragraphs [0033]-[0037], fig. 3	6-8

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

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

Date of the actual completion of the international search	Date of mailing of the international search report
06.11.2018	20.11.2018

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

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2018/034137

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	JP 2014-78239 A (SAMSUNG ELECTRONICS CO., LTD.) 01 May 2014, paragraph [0019] & US 2014/0109069 A1, paragraph [0039] & KR 10-2014-0046897 A & CN 103729228 A	6-8

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２０１８／０３４１３７

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F15/167(2006.01)i, F02D45/00(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.　G06F15/167, F02D45/00

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	１９７１－２０１８年
日本国実用新案登録公報	１９９６－２０１８年
日本国登録実用新案公報	１９９４－２０１８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
X Y A	JP 2010-501951 A（ローベルト　ボッシュ　ゲゼルシャフト　ミット　ベシュレンクテル　ハフツング）2010.01.21, 段落[0037]-[0047], 第3-5図 & US 2010/0242051 A1, 段落[0036]-[0045], 第3-5図 & WO 2008/095548 A1 & EP 1956484 A1 & CN 101601016 A	1-2, 5, 10 6-8 3-4, 9
Y	JP 2006-301894 A（ＮＥＣエレクトロニクス株式会社）2006.11.02, 段落[0016]-[0020], 第3図 & US 2006/0242257 A1, 段落[0033]-[0037], 第3図	6-8

☑ 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
０６．１１．２０１８	２０．１１．２０１８

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）　５Ｂ　９３６６
漆原　孝治
電話番号　０３－３５８１－１１０１　内線　３５４５

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 2014-78239 A (三星電子株式会社) 2014.05.01, 段落[0019] & US 2014/0109069 A1, 段落[0039] & KR 10-2014-0046897 A & CN 103729228 A	6-8