(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2020-4108
(P2020-4108A)
(43)公開日　令和2年1月9日(2020.1.9)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 15/167	(2006.01)	G06F	15/167	615A	5B034
G06F 11/18	(2006.01)	G06F	11/18	640	5B045

審査請求 未請求　請求項の数 20　ＯＬ　（全 30 頁）

(21)出願番号　特願2018-123381 (P2018-123381)
(22)出願日　平成30年6月28日 (2018.6.28)

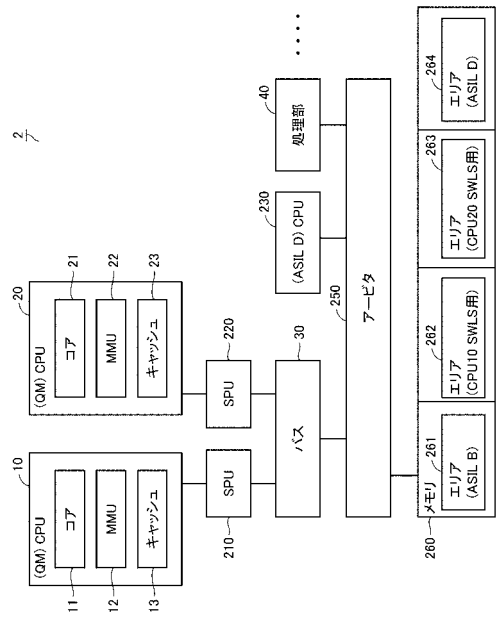
(71)出願人　302062931
ルネサスエレクトロニクス株式会社
東京都江東区豊洲三丁目２番２４号
(74)代理人　110001195
特許業務法人深見特許事務所
(72)発明者　早川　雄貴
東京都江東区豊洲三丁目２番２４号　ルネサスエレクトロニクス株式会社内
(72)発明者　加谷　俊之
東京都江東区豊洲三丁目２番２４号　ルネサスエレクトロニクス株式会社内
(72)発明者　芝原　真一
東京都江東区豊洲三丁目２番２４号　ルネサスエレクトロニクス株式会社内

Ｆターム(参考)　5B034 AA02　CC01　CC03
5B045 BB12　BB32　DD03

(54)【発明の名称】半導体装置、制御システムおよび半導体装置の制御方法

(57)【要約】
【課題】ロックステップのためのハードウェアを追加することなく半導体装置のソフトウェアロックステップを実現する。
【解決手段】半導体装置２は、ＣＰＵ１０，２０と、スヌープ動作を制御するＳＰＵ２１０，２２０と、機能安全規格の最高レベル（ＡＳＩＬ　Ｄ）のＣＰＵ２３０と、アービタ２５０と、メモリ２６０と、比較器とを備え、ＣＰＵ２３０は、通常動作が実行される場合にスヌープ動作を許可し、ソフトウェアロックステップが実行される場合にスヌープ動作を禁止し、ＣＰＵ１０は、ソフトウェアロックステップのための第１のソフトウェアを実行してメモリ２６０に確保された第１の領域２６２に実行結果を書き込み、ＣＰＵ２０は、ソフトウェアロックステップのための第２のソフトウェアを実行し、第２の領域２６３に実行結果を書き込み、比較器は、第１の領域に書き込まれた実行結果と、第２の領域に書き込まれた実行結果とを比較する。
【選択図】図２

図2



【特許請求の範囲】
【請求項１】
半導体装置であって、
キャッシュを有し、ソフトウェアロックステップと通常動作とを実行するための第１および第２のプロセッサと、
メモリと、
前記メモリに接続され、アドレスプロテクト機能を有するアービタと、
前記アービタに接続されたバスと、
前記第１のプロセッサと前記バスとに接続され、前記第１のプロセッサによる前記第２のプロセッサのキャッシュへのスヌープ動作を制御するための第１のスヌープ制御回路と、
前記第２のプロセッサと前記バスとに接続され、前記第２のプロセッサによる前記第１のプロセッサのキャッシュへのスヌープ動作を制御するための第２のスヌープ制御回路と、
前記ソフトウェアロックステップと前記通常動作とを切り替えるための制御装置とを備え、
前記制御装置は、
前記通常動作が実行される場合に、前記第１のスヌープ制御回路および前記第２のスヌープ制御回路に対して、各前記スヌープ動作を許可し、
前記ソフトウェアロックステップが実行される場合に、前記第１のスヌープ制御回路および前記第２のスヌープ制御回路に対して、各前記スヌープ動作を禁止し、
前記第１のプロセッサは、
前記ソフトウェアロックステップのための第１のソフトウェアを実行し、
前記第１のプロセッサのために前記メモリに確保された第１の領域に、前記第１のソフトウェアの実行結果を書き込み、
前記第２のプロセッサは、
前記ソフトウェアロックステップのための第２のソフトウェアを実行し、
前記第１の領域と異なる第２の領域に、前記第２のソフトウェアの実行結果を書き込み、
前記半導体装置は、前記第１の領域に書き込まれた実行結果と、前記第２の領域に書き込まれた実行結果とを比較する比較器をさらに備える、半導体装置。
【請求項２】
前記第１のスヌープ制御回路または前記第２のスヌープ制御回路は、所定の条件を満たさない異常なスヌープ動作を禁止し、
前記アービタは、前記制御装置によってアクセス可能なメモリ領域に対する前記第１のプロセッサまたは前記第２のプロセッサによるアクセスを禁止する、請求項１に記載の半導体装置。
【請求項３】
前記アービタに接続された第１の処理部および第２の処理部をさらに備え、
前記第１のプロセッサが前記第１のソフトウェアを実行することは、
前記第１の処理部による演算結果に基づいて、ソフトウェアロックステップのための第３のソフトウェアを実行することと、
前記第３のソフトウェアの実行結果に基づいて前記第１のソフトウェアを実行することとを含み、
前記第２のプロセッサが前記第２のソフトウェアを実行することは、
前記第２の処理部による演算結果に基づいて、ソフトウェアロックステップのための第４のソフトウェアを実行することと、
前記第４のソフトウェアの実行結果に基づいて前記第２のソフトウェアを実行することとを含む、請求項１に記載の半導体装置。
【請求項４】

前記第１のプロセッサは、前記第１のソフトウェアの実行結果に基づいてＣＲＣ（Cyclic Redundancy Code）を生成するように構成されており、
前記制御装置は、
前記第２のソフトウェアの実行結果に基づいてＣＲＣを生成し、
前記第１のプロセッサによって生成されたＣＲＣと、前記制御装置によって生成されたＣＲＣとを比較するように構成されている、請求項１に記載の半導体装置。
【請求項５】
前記比較器は、前記アービタに含まれており、前記メモリに格納されているデータの読出要求に基づいて、前記メモリに格納されている各実行結果を比較するように構成されている、請求項１に記載の半導体装置。
【請求項６】
前記メモリは、データを格納するための複数の領域を含み、
前記第１のプロセッサに割り当てられる領域は、前記第２のプロセッサによる前記実行結果の書き込みから保護されており、
前記第２のプロセッサに割り当てられる領域は、前記第１のプロセッサによる前記実行結果の書き込みから保護されており、
前記制御装置に割り当てられる領域は、前記第１のプロセッサおよび前記第２のプロセッサによる書き込みから保護されている、請求項１に記載の半導体装置。
【請求項７】
前記制御装置の機能安全レベルは、前記第１のプロセッサおよび前記第２のプロセッサの機能安全レベルよりも高い、請求項１に記載の半導体装置。
【請求項８】
信号の入力を受ける入力インターフェイスと、
前記信号に基づいて演算を実行する半導体装置と、
前記演算の実行結果を出力するための出力インターフェイスとを備え、
前記半導体装置は、
キャッシュを有し、ソフトウェアロックステップと通常動作とを実行するための第１および第２のプロセッサと、
メモリと、
前記メモリに接続され、アドレスプロテクト機能を有するアービタと、
前記アービタに接続されたバスと、
前記第１のプロセッサと前記バスとに接続され、前記第１のプロセッサによる前記第２のプロセッサのキャッシュへのスヌープ動作を制御するための第１のスヌープ制御回路と、
前記第２のプロセッサと前記バスとに接続され、前記第２のプロセッサによる前記第１のプロセッサのキャッシュへのスヌープ動作を制御するための第２のスヌープ制御回路と、
前記ソフトウェアロックステップと前記通常動作とを切り替えるための制御装置とを備え、
前記制御装置は、
前記通常動作が実行される場合に、前記第１のスヌープ制御回路および前記第２のスヌープ制御回路に対して、各前記スヌープ動作を許可し、
前記ソフトウェアロックステップが実行される場合に、前記第１のスヌープ制御回路および前記第２のスヌープ制御回路に対して、各前記スヌープ動作を禁止し、
前記第１のプロセッサは、
前記ソフトウェアロックステップのための第１のソフトウェアを実行し、
前記第１のプロセッサのために前記メモリに確保された第１の領域に、前記第１のソフトウェアの実行結果を書き込み、
前記第２のプロセッサは、
前記ソフトウェアロックステップのための第２のソフトウェアを実行し、

前記第１の領域と異なる第２の領域に、前記第２のソフトウェアの実行結果を書き込み、
前記半導体装置は、前記第１の領域に書き込まれた実行結果と、前記第２の領域に書き込まれた実行結果とを比較する比較器をさらに備える、制御システム。
【請求項９】
前記第１のスヌープ制御回路または前記第２のスヌープ制御回路は、所定の条件を満たさない異常なスヌープ動作を禁止し、
前記アービタは、前記制御装置によってアクセス可能なメモリ領域に対する前記第１のプロセッサまたは前記第２のプロセッサによるアクセスを禁止する、請求項８に記載の制御システム。
【請求項１０】
前記アービタに接続された第１の処理部および第２の処理部をさらに備え、
前記第１のプロセッサが前記第１のソフトウェアを実行することは、
前記第１のプロセッサが、
前記第１の処理部による演算結果に基づいて、ソフトウェアロックステップのための第３のソフトウェアを実行することと、
前記第３のソフトウェアの実行結果に基づいて前記第１のソフトウェアを実行することとを含み、
前記第２のプロセッサが前記第２のソフトウェアを実行することは、
前記第２のプロセッサが、
前記第２の処理部による演算結果に基づいて、ソフトウェアロックステップのための第４のソフトウェアを実行することと、
前記第４のソフトウェアの実行結果に基づいて前記第２のソフトウェアを実行することとを含む、請求項８に記載の制御システム。
【請求項１１】
前記第１のプロセッサは、前記第１のソフトウェアの実行結果に基づいてＣＲＣ（Cyclic Redundancy Code）を生成するように構成されており、
前記制御装置は、
前記第２のソフトウェアの実行結果に基づいてＣＲＣを生成し、
前記第１のプロセッサによって生成されたＣＲＣと、前記制御装置によって生成されたＣＲＣとを比較するように構成されている、請求項８に記載の制御システム。
【請求項１２】
前記比較器は、前記アービタに含まれており、前記メモリに格納されているデータの読出要求に基づいて、前記メモリに格納されている各実行結果を比較するように構成されている、請求項８に記載の制御システム。
【請求項１３】
前記メモリは、データを格納するための複数の領域を含み、
前記第１のプロセッサに割り当てられる領域は、前記第２のプロセッサによる前記実行結果の書き込みから保護されており、
前記第２のプロセッサに割り当てられる領域は、前記第１のプロセッサによる前記実行結果の書き込みから保護されており、
前記制御装置に割り当てられる領域は、前記第１のプロセッサおよび前記第２のプロセッサによる書き込みから保護されている、請求項８に記載の制御システム。
【請求項１４】
前記制御装置の機能安全レベルは、前記第１のプロセッサおよび前記第２のプロセッサの機能安全レベルよりも高い、請求項８に記載の制御システム。
【請求項１５】
キャッシュを有し、ソフトウェアロックステップと通常動作とを実行するための第１および第２のプロセッサを準備するステップと、
前記第１のプロセッサによる前記第２のプロセッサのキャッシュへのスヌープ動作を制

御するステップと、
前記第２のプロセッサによる前記第１のプロセッサのキャッシュへのスヌープ動作を制御するステップと、
前記通常動作が実行される場合に、各前記スヌープ動作を許可するステップと、
前記ソフトウェアロックステップが実行される場合に、各前記スヌープ動作を禁止するステップと、
前記第１のプロセッサが、前記ソフトウェアロックステップのための第１のソフトウェアを実行するステップと、
前記第１のプロセッサのためにメモリに第１の領域を確保するステップと、
前記第２のプロセッサのために前記第１の領域と異なる第２の領域を確保するステップと、
前記第１のプロセッサが、前記第１の領域に、前記第１のソフトウェアの実行結果を書き込むステップと、
前記第２のプロセッサが、前記ソフトウェアロックステップのための第２のソフトウェアを実行するステップと、
前記第２のプロセッサが、前記第２の領域に、前記第２のソフトウェアの実行結果を書き込むステップと、
制御装置が、前記第１の領域に書き込まれた実行結果と、前記第２の領域に書き込まれた実行結果とを比較するステップとを含む、半導体装置の制御方法。
【請求項１６】
所定の条件を満たさない異常なスヌープ動作を禁止するステップと、
メモリ領域に対する前記第１のプロセッサまたは前記第２のプロセッサによるアクセスを禁止するステップとをさらに含む、請求項１５に記載の制御方法。
【請求項１７】
第１の処理部および第２の処理部を準備するステップをさらに含み、
前記第１の処理部による演算結果に基づいて、ソフトウェアロックステップのための第３のソフトウェアを実行することと、
前記第３のソフトウェアの実行結果に基づいて前記第１のソフトウェアを実行することとを含み、
前記第２のプロセッサが、前記第２のソフトウェアを実行するステップは、
前記第２の処理部による演算結果に基づいて、ソフトウェアロックステップのための第４のソフトウェアを実行することと、
前記第４のソフトウェアの実行結果に基づいて前記第２のソフトウェアを実行することとを含む、請求項１５に記載の制御方法。
【請求項１８】
前記第１のソフトウェアの実行結果に基づいてＣＲＣ（Cyclic Redundancy Code）を生成するステップと、
前記第２のソフトウェアの実行結果に基づいてＣＲＣを生成するステップとをさらに含み、
前記比較するステップは、各前記生成されたＣＲＣを比較することを含む、請求項１５に記載の制御方法。
【請求項１９】
前記メモリに格納されているデータの読出要求に基づいて、前記メモリに格納されている各実行結果を比較するステップとをさらに含む、請求項１５に記載の制御方法。
【請求項２０】
前記メモリは、データを格納するための複数の領域を含み、
前記第１のプロセッサに割り当てられる領域は、前記第２のプロセッサによる前記実行結果の書き込みから保護されており、
前記第２のプロセッサに割り当てられる領域は、前記第１のプロセッサによる前記実行結果の書き込みから保護されており、

前記制御装置に割り当てられる領域は、前記第１のプロセッサおよび前記第２のプロセッサによる書き込みから保護されている、請求項１５に記載の制御方法。

【発明の詳細な説明】

【技術分野】

【０００１】

本開示は半導体装置に関し、より特定的には、半導体装置におけるロックステップに関する。

【背景技術】

【０００２】

昨今、より高い機能安全のレベルが求められている。ＩＳＯ２６２６２の機能安全として、ＡＳＩＬ（Automotive Safety Integrity Level）が規定されている。安全レベルが低いものから、ＱＭ（Quality Management）／ＡＳＩＬ Ａ／ＡＳＩＬ Ｂ／ＡＳＩＬ Ｃ／ＡＳＩＬ Ｄと規定されている。ＡＳＩＬ Ｂレベルでは、動作中に発生する故障の９０％以上が検知されなければならず、ＡＳＩＬ Ｄレベルでは、９９％以上の故障が検知されなければならない。例えば、ＩＳＯ２６２６２に準拠した車載電子システムでは、ＡＳＩＬ Ｄが要求される。具体的には、ＡＳＩＬ Ｄレベルの機能安全には、ロックステップと呼ばれる、２つ以上のハードウェアの演算で得られた結果を比較一致させるような高いレベルの故障検出性能が求められる。

【０００３】

ロックステップに関し、例えば、特開２０１４－５６３９６号公報（特許文献１）は、「複数のプロセッサコアが正常に同じプログラムを実行しているか否かを判定するとともに、メモリチェックの実行機会を確保する電子制御装置」を開示している（［要約］）。

【先行技術文献】

【特許文献】

【０００４】

【特許文献１】特開２０１４－５６３９６号公報

【発明の概要】

【発明が解決しようとする課題】

【０００５】

特許文献１に開示された技術によれば、機能安全レベルは高いが１つの動作しかできないデュアルコア・ロックステップのモードと、機能安全レベルは通常であるが２つのコアがそれぞれ動作できるモードとを切り替えることができる。しかしながら、デュアルコア・ロックステップのモードへの切り替えの際には、２つのコアが同期をとるためにリセットが必要となり、数ミリ秒程度の処理停止期間が生じ、性能が低下する場合がある。したがって、機能安全レベルを維持しつつ性能が低下しない技術が必要とされている。

【０００６】

本開示は、上述のような問題点を解決するためになされたものであって、ある局面において、機能安全レベルを維持しつつ性能が低下しない半導体装置が開示される。他の局面において、機能安全レベルを維持しつつ性能が低下しない制御システムが開示される。さらに他の局面において、機能安全レベルを維持しつつ性能が低下しないように半導体装置を制御する方法が開示される。

【課題を解決するための手段】

【０００７】

ある実施の形態に従う半導体装置は、キャッシュを有し、ソフトウェアロックステップと通常動作とを実行するための第１および第２のプロセッサと、メモリと、メモリに接続され、アドレスプロテクト機能を有するアービタと、アービタに接続されたバスと、第１のプロセッサとバスとに接続され、第１のプロセッサによる第２のプロセッサのキャッシュへのスヌープ動作を制御するための第１のスヌープ制御回路と、第２のプロセッサとバスとに接続され、第２のプロセッサによる第１のプロセッサのキャッシュへのスヌープ動作を制御するための第２のスヌープ制御回路と、ソフトウェアロックステップと通常動作

とを切り替えるための制御装置とを備える。制御装置は、通常動作が実行される場合に、第１のスヌープ制御回路および第２のスヌープ制御回路に対して、各スヌープ動作を許可し、ソフトウェアロックステップが実行される場合に、第１のスヌープ制御回路および第２のスヌープ制御回路に対して、各スヌープ動作を禁止し得る。第１のプロセッサは、ソフトウェアロックステップのための第１のソフトウェアを実行し、第１のプロセッサのためにメモリに確保された第１の領域に、第１のソフトウェアの実行結果を書き込み得る。第２のプロセッサは、ソフトウェアロックステップのための第２のソフトウェアを実行し、第１の領域と異なる第２の領域に、第２のソフトウェアの実行結果を書き込み得る。半導体装置は、第１の領域に書き込まれた実行結果と、第２の領域に書き込まれた実行結果とを比較する比較器をさらに備える。

【０００８】

ある局面において、半導体装置の機能安全レベルを維持しつつ性能の低下を防止し得る。

【０００９】

その他の課題と新規な特徴は、本明細書の記述および添付図面から明らかになるであろう。

【図面の簡単な説明】

【００１０】

【図１】半導体装置１のハードウェア構成を表わすブロック図である。

【図２】ソフトウェアロックステップを実行可能な半導体装置２の構成の概略を表わすブロック図である。

【図３】ＳＰＵ２１０，２２０の構成の詳細を説明するための図である。

【図４】半導体装置２の動作を表わすフローチャート（その１）である。

【図５】半導体装置２の動作を表わすフローチャート（その２）である。

【図６】半導体装置２の動作を表わすフローチャート（その３）である。

【図７】半導体装置２の動作を説明するための図（その１）である。

【図８】半導体装置２の動作を説明するための図（その２）である。

【図９】半導体装置２の動作を説明するための図（その３）である。

【図１０】半導体装置２の動作を説明するための図である。

【図１１】ソフトウェアロックステップが行なわれない場合の設定を説明するための図である。

【図１２】スヌープが誤って他のＣＰＵに対して行なわれた場合の保護を説明する図である。

【図１３】メモリ２６０内においてアドレスエリアを跨ぐアクセスのプロテクトについて説明する図である。

【図１４】スヌープを許可する場合の動作を説明する図である。

【図１５】スヌープが許可されている場合の動作を表わす図である。

【図１６】ソフトウェアロックステップが行なわれていない場合の半導体装置２の動作を説明する図である。

【図１７】ある局面に従う半導体装置３におけるデータの流れを表わす図である。

【図１８】ある局面に従う半導体装置４の構成の概略を表わすブロック図である。

【図１９】半導体装置４におけるデータの流れを表わす図である。

【図２０】半導体装置４が実行する処理の一部を表わすフローチャートである。

【図２１】半導体装置５におけるデータフローを表わす図である。

【図２２】半導体装置５におけるＣＰＵのプログラムの実行結果の出力を説明するための図である。

【図２３】半導体装置５におけるソフトウェアロックステップのプログラムの実行結果の比較を説明するための図である。

【図２４】半導体装置６の構成を表わすブロック図である。

【図２５】半導体装置６におけるデータのフローを説明するための図である。

【図２６】半導体装置６の制御構造を説明するためのフローチャートである。
【図２７】半導体装置６における処理の一部を表わすフローチャートである。
【図２８】対象エリアに格納されているデータの読み出しを説明するための図である。
【図２９】比較器２５２による比較の動作を説明するための図である。
【発明を実施するための形態】
【００１１】
以下、図面を参照しつつ、本開示に係る技術思想の実施の形態について説明する。以下の説明では、同一の部品には同一の符号を付してある。それらの名称および機能も同じである。したがって、それらについての詳細な説明は繰り返さない。
【００１２】
［第１の実施の形態］
図１を参照して、ある局面に従ってソフトウェアロックステップ（ＳＷＬＳ）を行なう半導体装置１の構成について説明する。図１は、半導体装置１のハードウェア構成を表わすブロック図である。本実施の形態において、ソフトウェアロックステップとは、複数のコアが同一の結果になるプログラムを実行し、これらのコア以外の比較装置によってその結果が同一であることを確認することで、これらのコアの故障の検知を行なう方法をいう。より具体的には、機能安全レベルが通常である二つのコアが、クロックサイクル単位で同期をとることなく、アプリケーション単位あるいは関数単位でプログラムを実行し、実行結果が比較される。この時、各コアへの入力がクロックサイクル単位で同じであることが求められるデュアルコアロックステップと異なり、本実施の形態に従うソフトウェアロックステップでは、各コアへの入力タイミングは任意でよく、各コアの同期をとる必要はない。
【００１３】
半導体装置１は、ＣＰＵ（Central Processing Unit）１０，２０と、バス３０と、処理部４０と、アービタ５０と、メモリ６０とを備える。ＣＰＵ１０は、コア１１と、ＭＭＵ（Memory Management Unit）１２と、キャッシュ１３とを含む。ＣＰＵ２０は、コア２１と、ＭＭＵ２２と、キャッシュ２３とを含む。ＣＰＵ１０，２０は、それぞれ、バス３０に接続されている。バス３０と、処理部４０と、メモリ６０とが、アービタ５０に接続されている。
【００１４】
ＣＰＵ１０，２０は、機能安全を適用しなくてもよい通常の品質管理（Quality Management（ＱＭ））クラスのプロセッサである。ＣＰＵ１０，２０は、データをキャッシュに保存し得る。ある局面において、ＣＰＵ２０は、データをキャッシュ２３に格納して、メモリ６０に当該データを書き込んでいない場合がある。そのような場合において、ＣＰＵ２０以外のプロセッサ、例えばＣＰＵ１０が、当該データを扱いたい時、ＣＰＵ１０は、所謂スヌープと呼ばれる処理、すなわち、ＣＰＵ２０のキャッシュ２３にアクセスして当該データを読み込む等の処理を行う必要がある。スヌープを行なうことにより、コア１１，２１間で高速にデータをやり取りできるので、スヌープがない場合と比べて、演算速度等の性能が大きく向上し得る。
【００１５】
コア１１，２１は、演算処理回路を含む。ＭＭＵ１２は、メモリのアドレス変換、メモリ６０へのアクセスの保護、キャッシュ１３へのアクセスを制御する。ＭＭＵ２２は、メモリのアドレス変換、メモリ６０へのアクセスの保護、キャッシュ２３へのアクセスを制御する。
【００１６】
バス３０は、キャッシュコヒーレントインターコネクト（ＣＣＩ）であり、例えば、ＣＰＵ１０によるキャッシュへのアクセスを扱う。バス３０は、ＡＳＩＬ　Ｄクラスに対応する（以下、適宜、「ＡＳＩＬ　Ｄクラス」は「ＡＳＩＬ　Ｄ対応」あるいは「ＡＳＩＬ　Ｄレベル」と表わすこともあり、他の場合も同様である）。
【００１７】

処理部４０は、キャッシュを有さない演算器であり得る。処理部４０は、ＡＳＩＬ　Ｂクラスおよびａｓｉｌ　Ｄクラスのいずれであってもよい。

【００１８】

アービタ５０は、アドレスの保護機能を有し、ＣＰＵ１０，２０、バス３０、処理部４０によるメモリ６０へのアクセスを調停する。

【００１９】

メモリ６０は、例えば、ＤＤＲ（Double Data Rate）タイプのメモリである。

ある局面に従う半導体装置１によれば、ＣＰＵ１０，２０間のスヌープを防ぐためには、例えば、ＭＭＵ１２，２２がスヌープを保護することが考えられる。しかしながら、ＭＭＵ１２，２２自体がロックステップの対象であるため、スヌープを防ぐためにＭＭＵ１２，２２を使用することはできない。あるいは、ＣＰＵ１０，２０がスヌープをできないように構成することも考えられる。この場合、スヌープを行なっても問題がない場合でもスヌープができなくなるため、処理の遅延のように半導体装置１の性能が悪化する恐れがある。

【００２０】

なお、本実施の形態において、ソフトウェアやハードウェアのバグによる故障をシステマティックフォールトと呼び、ハードウェアが宇宙線や経年劣化で故障することをランダムハードウェアフォールトと呼ぶ。ＣＰＵがＡＳＩＬ　Ｄ対応であるとは、単独で（ソフトウェアロックステップを使用せずに）システマティック・フォールトとランダムハードウェア・フォールトの両方において、ＡＳＩＬ　Ｄクラスに対応するＣＰＵであることをいう。また、ソフトウェアロックステップに使用されるハードウェアは、システマティック・フォールトの観点ではＡＳＩＬ　Ｄクラスとし、ランダムハードウェア・フォールトの観点では、単体では、ＡＳＩＬ　Ｄクラスではないものを指すとする。例えば、ソフトウェアロックステップが行なわれない場合には、当該ハードウェア（例えば、ＣＰＵ）は、ＱＭクラスであってもよい。

【００２１】

図１に示される半導体装置１においてソフトウェアロックステップが行なわれる場合、以下の様な問題が生じ得る。より具体的には、ＣＰＵ１０のＭＭＵ１２が故障して不要なスヌープが開始され、ＣＰＵ２０のキャッシュ２３へのアクセスが行なわれる場合があり得る。この場合、当該スヌープによって、ＣＰＵ１０の故障がＣＰＵ２０に広がり、ＣＰＵ１０，２０による誤った演算結果が一致し、ＣＰＵ１０の故障が検知されない可能性があり、結果として、ＡＳＩＬ　Ｄクラスを達成することができない。

【００２２】

既存の回路構成では、スヌープはＭＭＵ１２，２２において保護されることになっているが、ＭＭＵ１２，２２自体がソフトウェアロックステップの対象となっており、スヌープを保護することができない。また、ＱＭクラスのＣＰＵ１０，２０のうち、ＭＭ１２，２２までを２重化する方法も考えられるが、その場合、面積のオーバヘッドが大きくなるという問題が生じ得る。さらに、近年では、ＣＰＵ、画像処理回路、メモリ等のＬＳＩ（Large Scale Integrated Circuit）を構成する機能ブロック（所謂ＩＰ（Intellectual Property））は、自社製ではなく社外製であることが多く、ＣＰＵ自体に変更を加えられないこともある。また、プロトコルの複雑化により、ＭＭＵなどの面積が増加してきており、それを別途追加するのは、従来のメリットが損なわれるうえ、レイテンシも増大する。

【００２３】

また、ＩＳＯ２６２６２では、ＡＳＩＬの異なるクラス間での故障の伝播等の干渉があってはならないとされており、不干渉（以下、ＦＦＩ（Freedom From Interface）ともいう。）が求められている。ソフトウェアロックステップによって達成されるＡＳＩＬ　ＤクラスとＣＰＵ、ソフトウェアロックステップを行なわないＱＭクラスのＣＰＵ１０，２０との間で、ＦＦＩの考慮が必要になるため、単体でＡＳＩＬ　ＤクラスのＣＰＵのように、ＱＭクラスのＣＰＵ１０，２０を制御するためのプロセッサが必要となる。

【００２４】
そこで、図２を参照して、ある局面に従う半導体装置２について説明する。図２は、ソフトウェアロックステップを実行可能な半導体装置２の構成の概略を表わすブロック図である。半導体装置２は、半導体装置１の構成に加えて、スヌープ動作を制御するための回路として、スヌープ保護部（以下、ＳＰＵ（Snoop Protection Unit）と表す）２１０，２２０と、ＣＰＵ２３０とを備える。さらに、半導体装置２は、半導体装置１の構成に対して、アービタ５０に代えてアービタ２５０を備え、メモリ６０に代えてメモリ２６０を備える。
【００２５】
ＳＰＵ２１０は、ＣＰＵ１０によるスヌープを保護する。ＳＰＵ２２０は、ＣＰＵ２０によるスヌープを保護する。ＳＰＵ２１０，２２０は、ＡＳＩＬ Ｄクラスである。
【００２６】
ＣＰＵ２３０は、ＡＳＩＬ Ｄクラスのプロセッサである。半導体装置２において、ＦＦＩの観点から、ＡＳＩＬ Ｄクラスを達成するための制御は、ＡＳＩＬ Ｄクラスのプロセッサで行なわれる必要がある。そこで、ＣＰＵ２３０は、ＡＳＩＬ Ｄクラスであることが望ましい。
【００２７】
アービタ２５０は、メモリ２６０へのアクセスを調停する。例えば、アービタ２５０は、アドレスが異常であることを検知すると、当該アドレスへのアクセスを禁止する。あるいは、アービタ２５０は、メモリ２６０内において、アドレスエリアをまたぐようなアクセスをプロテクトし得る。例えば、アービタ２５０は、ＡＳＩＬ Ｄ対応のプロセッサに割り当てられたメモリ領域に対するＡＳＩＬ Ｂ対応のプロセッサによるデータの書き込みを禁止する。
【００２８】
メモリ２６０は、ＤＤＲタイプのメモリであり得る。メモリ２６０は、エリア２６１，２６２，２６３，２６４を含む。エリア２６１は、ＡＳＩＬ Ｂクラスのハードウェアのためのエリアである。エリア２６２は、ＣＰＵ１０のソフトウェアロックステップためのエリアである。エリア２６３は、ＣＰＵ２０のソフトウェアロックステップためのエリアである。エリア２６４は、ＡＳＩＬ Ｄクラスのハードウェアのためのエリアである。
【００２９】
ソフトウェアロックステップのための複数のエリア２６２，２６３は、それぞれのＣＰＵ１０，２０が排他的に当該エリアにアクセスできるように構成されている。ＡＳＩＬ Ｂクラスのハードウェア（例えば、処理部４０）は、ＦＦＩの要件を満たすため、エリア２６２，２６３，２６４にアクセスできないように構成される。また、ソフトウェアロックステップの対象となるＣＰＵ１０，２０は、同様の理由により、ＡＳＩＬ Ｄクラスのハードウェアによって使用されるエリア２６４にアクセスできないように構成される。なお、ＡＳＩＬ Ｄクラスの機能安全では、バスのように半導体装置２において他の回路等と共通して使用される部品に関しても、ＡＳＩＬ Ｄ対応であることが求められる。これらの部品においては、デュアルコア・ロックステップやＥＤＣ（Error Detection Code）機能若しくはＥＣＣ（Error Correction Code）等でＡＳＩＬ Ｄ対応であるものとする。
【００３０】
［スヌープ保護部］
図３を参照して、ある局面に従うＳＰＵ２１０，２２０の構成について説明する。図３は、ＳＰＵ２１０，２２０の構成の詳細を説明するための図である。ある局面において、ＳＰＵ２１０，２２０は、デュアルコア・ロックステップ等の仕組みにより、ＡＳＩＬ Ｄクラスに対応している。
【００３１】
ＳＰＵ２１０，２２０は、スヌープｂｉｔ強制無効化ブロック３１０と、プロテクトＯＮ／ＯＦＦレジスタ３２０とを含む。ＣＰＵ１０の出力は、ＳＰＵ２１０のスヌープｂｉ

ｔ強制無効化ブロック３１０に、ＣＰＵ２０の出力は、ＳＰＵ２２０のスヌープｂｉｔ強制無効化ブロック３１０に、それぞれ入力され得る。スヌープｂｉｔ強制無効化ブロック３１０の出力は、バス３０に入力され得る。

【００３２】

ある局面において、プロテクトＯＮ／ＯＦＦレジスタ３２０がオフに設定されている場合には、ＳＰＵ２１０，２２０は、マスタ（例えば、ＡＳＩＬ　ＤクラスのＣＰＵ２３０）からの信号をそのまま通す。別の局面において、プロテクトＯＮ／ＯＦＦレジスタ３２０がオンに設定されている場合には、ＳＰＵ２１０，２２０は、ＣＰＵ１０，２０からのリクエストに含まれるスヌープを示す信号を監視し、その信号を無効にして、バス３０に送る。

【００３３】

本実施の形態に係る半導体装置２では、ＳＰＵ２１０，２２０がスヌープの保護を行なう。ＳＰＵ２１０，２２０には、スヌープのアクセス保護以外の機能を持たせないことで、レイテンシの増大を低減できる。また、ソフトウェアロックステップが行なわれない場合には、ＳＰＵ２１０，２２０によるスヌープを許可することで、半導体装置２の性能を高めることができる。

【００３４】

［制御構造］

図４～図６を参照して、半導体装置２の制御構造について説明する。図４～図６は、半導体装置２の動作を表わすフローチャートである。

【００３５】

図４に示されるように、ステップＳ４１０にて、ＡＳＩＬ　ＤクラスのＣＰＵ２３０は、ソフトウェアロックステップのために、ＳＰＵ２１０，２２０をスヌープ禁止に設定する。例えば、ＣＰＵ２３０は、プロテクトＯＮ／ＯＦＦレジスタ３２０の設定をオフにする。また、ＣＰＵ２３０は、アービタ２５０に、禁止アドレスを設定する。例えば、ＣＰＵ２３０は、ＣＰＵ１０がエリア２６２にアクセスできるようにアドレスを設定し、ＣＰＵ２０によるアクセスを禁止する。ＣＰＵ２３０は、ＣＰＵ２０がエリア２６３にアクセスできるようにアドレスを設定し、ＣＰＵ１０によるアクセスを禁止する。エリア２６４は、ＣＰＵ２３０によるアクセスが可能にアドレス設定され、他のＣＰＵ１０，２０によるアクセスは禁止される。

【００３６】

ステップＳ４１５にて、ＣＰＵ２３０は、ソフトウェアロックステップ用のプログラムおよびデータをメモリ２６０のエリア２６４に書き込む。

【００３７】

ステップＳ４２０にて、ＣＰＵ２３０は、メモリ２６０への書き込みを完了したことに基づいて、ソフトウェアロックステップを行なうＣＰＵ１０，２０を起動する。例えば、ＣＰＵ２３０は、ソフトウェアロックステップの開始指令を、アービタ２５０、バス３０を介して、ＣＰＵ１０，２０に送信する。ステップＳ４２５にて、ＣＰＵ１０，２０は、当該開始指令の受信を検知すると、起動する。

【００３８】

ステップＳ４３０にて、ＣＰＵ１０は、メモリ２６０にアクセスして、ソフトウェアロックステップのためのプログラムおよびデータを読み出して、そのプログラムを実行する。同様に、ステップＳ４３１にて、ＣＰＵ２０は、メモリ２６０にアクセスして当該プログラムおよびデータを読み出して、そのプログラムを実行する。なお、ステップＳ４３０における読み出しおよび／または実行のタイミングと、ステップＳ４３１における読み出しおよび／または実行のタイミングとは、同一である必要はなく、異なっていてもよい。

【００３９】

ステップＳ４３５にて、ＣＰＵ１０は、当該プログラムの実行結果を、メモリ２６０に確保されたエリア２６２に書き込む。ステップＳ４３６にて、ＣＰＵ２０は、当該プログラムの実行結果を、メモリ２６０に確保されたエリア２６３に書き込む。なお、ステップ

Ｓ４３５における書き込みのタイミングと、ステップＳ４３６における書き込みのタイミングとは、同一である必要はなく、異なっていてもよい。

【００４０】

ステップＳ４４０にて、ＣＰＵ２３０は、ＣＰＵ１０，２０による各処理が終了したことを検知するまで待機する。ＣＰＵ１０，２０は、実行結果の書き込みを終了すると、その旨を、バス３０及びアービタ２５０を介して、ＣＰＵ２３０に通知する。

【００４１】

ステップＳ４４５にて、ＣＰＵ２３０は、ＣＰＵ１０，２０による実行結果の書き込みが終了した旨の通知を受信すると、エリア２６２，２６３にアクセスして実行結果を読み出し、ＣＰＵ１０による実行結果と、ＣＰＵ２０による実行結果とを比較する。ある局面において、比較の単位は、例えば、アプリケーション単位または関数単位である。

【００４２】

ステップＳ４５０にて、ＣＰＵ２３０は、これらの実行結果が一致しているか否かを判断する。ＣＰＵ２３０は、これらの実行結果が一致していると判断すると（ステップＳ４５０にてＹＥＳ）、ステップＳ４５５にて、ＣＰＵ２３０は、実行結果がＡＳＩＬ　Ｄレベルに対応していると判定し、所定の出力先に当該実行結果を出力し、ソフトウェアロックステップを終了する。当該実行結果が一致していることに基づいて、所定の処理が実行される。そうでない場合には（ステップＳ４５０にてＮＯ）、ステップＣＰＵ２３０は、実行結果がＡＳＩＬ　Ｄレベルに対応していないと判定し、エラーを出力する。

【００４３】

図５は、アービタ２５０によるプロテクト動作を表わす図である。

ステップＳ５１０にて、アービタ２５０は、トランザクションを待機している。

【００４４】

ステップＳ５２０にて、アービタ２５０は、トランザクションの受信に基づいて、トランザクションの情報および指定されたアドレスと、マスタ情報（ＣＰＵ２３０によって登録されている情報）およびアドレスとを比較する。

【００４５】

ステップＳ５３０にて、アービタ２５０は、ＣＰＵ２３０による設定内容に基づいて、メモリ２６０におけるアドレスのプロテクトが必要であるか否かを判断する。例えば、ＣＰＵ２１０は、ＣＰＵ１０に対して許可されていないアドレスへのアクセスをＣＰＵ１０が要求しているか否かを判断する。アービタ２５０は、アドレスのプロテクトが必要であると判断すると（ステップＳ５３０にてＹＥＳ）、制御をステップＳ５４０に切り替える。そうでない場合には（ステップＳ５３０にてＮＯ）、アービタ２５０は、制御をステップＳ５１０に戻す。

【００４６】

ステップＳ５４０にて、アービタ２５０は、エラーレスポンスをＣＰＵ１０に返し、当該トランザクションを通過させない。また、アービタ２５０は、所定のエラー信号をＣＰＵ２３０に出力し得る。

【００４７】

図６は、ＳＰＵ２１０が実行する処理の一部を表わすフローチャートである。なお、ＳＰＵ２２０も同様の処理を実行し得る。

【００４８】

ステップＳ６１０にて、ＳＰＵ２１０は、トランザクションを待機する。

ステップＳ６２０にて、ＳＰＵ２１０は、トランザクションがＣＰＵ１０によるスヌープであるか否かを判断する。この判断は、例えば、トランザクションに含まれる信号（例えば、後述するＡＲＳＮＯＯＰ信号）の値に基づいて行なわれる。

【００４９】

ＣＰＵ２１０は、当該トランザクションがスヌープであると判断すると（ステップＳ６３０にてＹＥＳ）、ステップＳ６４０にて、ＳＰＵ２１０は、当該トランザクションをスヌープではないトランザクションに変換し、当該トランザクションを通過させて、バス３

０に伝送する。これにより、ＣＰＵ１０が誤ったスヌープ動作を要求するトランザクションを発した場合には、当該トランザクションは、スヌープ動作を要求しないトランザクションに変換されるので、ＣＰＵ２０のキャッシュに対する誤ったスヌープが防止され得る。

【００５０】

他方、当該トランザクションがスヌープではないと判断すると（ステップＳ６３０にてＮＯ）、ＳＰＵ２１０は、制御をステップＳ６１０に戻す。その後、処理は繰り返される。

【００５１】

図７～図１０は、半導体装置２の動作を説明するための図である。図７を参照して、ある局面において、ＣＰＵ２３０は、ソフトウェアロックステップを実行するために、ＳＰＵ２１０，２２０によるスヌープを禁止する。例えば、ＣＰＵ２３０は、スヌープｂｉｔ強制無効化ブロック３１０において、ＳＰＵ２１０，２２０によるスヌープを禁止することを示すフラグを立てる。

【００５２】

ＣＰＵ２３０は、アービタ２５０のプロテクトアドレスを設定する。より具体的には、ＣＰＵ２３０は、メモリ２６０において、エリア２６２，２６３，２６４を確保し、各エリアへのアクセスを規定する。たとえば、エリア２６２は、ＣＰＵ１０およびＣＰＵ２３０によるアクセスが可能に設定され、ＣＰＵ２０によるアクセスが禁止される。エリア２６３は、ＣＰＵ２０およびＣＰＵ２３０によるアクセスが可能に設定され、ＣＰＵ１０によるアクセスが禁止される。エリア２６４は、ＣＰＵ２３０によるデータの書き込みが可能に設定され、ＣＰＵ１０，２０によるデータの書き込みが禁止される。さらに、ＣＰＵ２３０は、ソフトウェアロックステップに必要なプログラムおよびデータをメモリ２６０の所定のエリア２６２，２６３にそれぞれ格納する。このような構成により、ある局面において、あるプロセッサによって生成された誤ったデータが他のプロセッサによって使用され、比較結果が同一になることにより誤りが検出されなくなることが防止され得る。

【００５３】

図８を参照して、ＣＰＵ２３０は、ソフトウェアロックステップを実行するＣＰＵ１０，２０をそれぞれ起動する（ステップＳ４２０）。ある局面において、ＣＰＵ２３０は、アービタ２５０、バス３０およびＳＰＵ２１０，２２０を介して、起動命令をＣＰＵ１０，２０にそれぞれ送信する。別の局面において、ＣＰＵ２３０がＣＰＵ１０，２０に起動命令を送信するためのバスが別途設けられてもよい。

【００５４】

図９を参照して、ＣＰＵ１０，２０は、それぞれ、同一の結果をもたらすプログラムを実行し、実行結果を所定のエリア２６２，２６３にそれぞれ書き込む（ステップＳ４３０，Ｓ４３１）。

【００５５】

図１０を参照して、ＣＰＵ２３０は、エリア２６２，２６３から実行結果を読み出して、これらの実行結果を比較する（ステップＳ４４５）。ＣＰＵ２３０は、実行結果が一致すると判断すると、その旨を示すデータをエリア２６４の所定のアドレスに格納し得る。実行結果が一致しない場合には、ＣＰＵ２３０は、ＣＰＵ１０またはＣＰＵ２０が故障していることその他エラーを通知するためのデータを、エリア２６４の所定のアドレスに格納し得る。

【００５６】

図１１は、ソフトウェアロックステップが行なわれない場合の設定を説明するための図である。別の局面において、ＣＰＵ１０，２０がソフトウェアロックステップを行なわない場合には、ＣＰＵ２３０は、ＳＰＵ２１０，２２０の設定を非ソフトウェアロックアップに変更することにより、ＳＰＵ２１０，２２０によるスヌープを許可する。

【００５７】

［異常なスヌープのプロテクト］

図１２は、スヌープが誤って他のＣＰＵに対して行なわれた場合の保護を説明する図である。ある局面に従う半導体装置２は、ＡＸＩ（Advanced eXtensible Interface）規格に従ってバスが拡張されたＡＣＥ（ＡＸＩ Coherency Extensions）プロトコルを使用する場合がある。ＡＸＩ規格に従うインターフェイスを構成する５つのチャネルのうち、リードデータチャネルには、ＡＲＤＯＭＡＩＮ信号とＡＲＳＮＯＯＰ信号とが追加される。ＡＲＳＮＯＯＰ信号は、リードチャネルおよびライトチャネルでシェアされるトランザクションのためのスヌープトランザクションのタイプを示す。ＡＲＤＯＭＡＩＮ信号は、いずれのマスタがスヌープトランザクションにおいてスヌープされるべきであるか、と、いずれのマスタがバリアートランザクションを命令するために考慮されるべきであるか、とを示す。

【００５８】

スヌープ時には、トランザクション内のＡＲＤＯＭＡＩＮ信号は、２進数で０１または１０となっているか、ＡＲＳＮＯＯＰ信号が０以外になっている。そのような信号が来た時には、ＡＲＳＮＯＯＰ信号を強制的に０に設定する。このようにすることで、そのトランザクションがキャッシュを利用しないトランザクションとなり、スヌープの発生を防ぐことができる。

【００５９】

ライトトランザクションの場合も同様である。スヌープのためのアクセスは、通常のアクセスとなり、アドレスが異常であれば、アービタ２５０でプロテクトされる。データがおかしくなれば、ＣＰＵ２３０は、ソフトウェアロックステップの処理の実行結果を比較する処理において、当該比較の結果に基づいて異常を検知し得る。

【００６０】

［アドレス異常のプロテクト］

図１３は、メモリ２６０内においてアドレスエリアを跨ぐアクセスのプロテクトについて説明する図である。ある局面において、アービタ２５０は、メモリ２６０内におけるアドレスエリアを跨ぐようなアクセスを検知すると、そのアクセスを禁止し、当該アクセスにつながるトランザクションを与えたプロセッサに、エラーレスポンスを返す。より具体的には、アービタ２５０は、トランザクションのアドレスを監視し、そのトランザクションの発行元が、当該アドレスにアクセスしてはいけないことを検知すると、当該アドレスへのアクセスをプロテクトする。発行元は、トランザクションに付加されている識別情報で特定され得る。識別情報は、例えば、ＡｘＩＤその他の値であり得る。当該アドレスへのアクセスが許可されているか否かは、例えば、ＣＰＵ２３０において予め設定されている。

【００６１】

［スヌープの許可の設定］

図１４は、スヌープを許可する場合の動作を説明する図である。ある局面において、ＣＰＵ２３０は、ＳＰＵ２１０，２２０によるスヌープを許可する設定のための信号を、アービタ２５０およびバス３０を介して、ＳＰＵ２１０，２２０に送信する。ＳＰＵ２１０，２２０は、この信号を受信すると、プロテクトＯＮ／ＯＦＦレジスタ３２０のプロテクトをオフに設定する（図３）。プロテクトがオフに設定されると、スヌープｂｉｔ強制無効化ブロック３１０は、ＣＰＵ１０，２０からのリクエストに含まれるスヌープを表わす信号（例えば、ＡＲＳＮＯＯＰ）を検知した場合、そのまま、当該信号をアービタ２５０に送るので、スヌープが実行される。

【００６２】

図１５は、スヌープが許可されている場合の動作を表わす図である。ある局面において、スヌープが許可されている場合、ＣＰＵ１０は、キャッシュ２３に対するスヌープを、また、ＣＰＵ２０は、キャッシュ１３に対するスヌープを行なうことができる。また、ＣＰＵ１０，２０は、それぞれ、メモリ２６０のうち、ソフトウェアロックステップに使用されないエリア２６１のみへのアクセスが許可され、他のエリア２６２，２６３，２６４へのアクセスは禁止される。

【００６３】
図１６は、ソフトウェアロックステップが行なわれていない場合の半導体装置２の動作を説明する図である。ある局面において、ソフトウェアロックステップが行なわれていない時、アービタ２５０は、エリア２６２，２６３，２６４のアドレス保護を行なっている。したがって、ＣＰＵ１０は、エリア２６２，２６３，２６４にはアクセスできない。
【００６４】
以上のようにして、本実施の形態に従う半導体装置２は、異常なスヌープをプロテクトするためのＳＰＵ２１０，２２０と、アドレスをプロテクト可能なアービタ２５０と、ソフトウェアロックステップを制御するためのＡＳＩＬ　ＤクラスのＣＰＵ２３０とを備える。このような構成により、ソフトウェアロックステップを用いたＡＳＩＬ　Ｄが半導体装置２で実現できる。本実施の形態に従うＳＰＵ２１０，２２０は、図３に示されるように、簡易な構成で実現されるため、半導体装置２の大幅な面積増大を必要とせず、また、レイテンシの増大もない。また、ソフトウェアロックステップはＣＰＵ２３０によって制御されるため、ＣＰＵ１０，２０を変更できない場合であっても、ソフトウェアロックステップを導入することができる。
【００６５】
［第２の実施の形態］
以下、第２の実施の形態について説明する。本実施の形態に従う半導体装置は、所謂ＩＰコアの一例であるＣＰＵとして、二種類のＣＰＵを有する場合にソフトウェアロックステップを実行できる点で、一種類のＩＰコアを有する第１の実施の形態と異なる。
【００６６】
図１７は、ある局面に従う半導体装置３におけるデータの流れを表わす図である。半導体装置３において、ＣＰＵ２３０は、メモリ２６０のエリア６１に対してデータを読み書きする。ＣＰＵ１０，２０は、それぞれ、エリア６１からデータを読み出す。ＣＰＵ１０は、ソフトウェアロックステップを行う際、メモリ２６０のエリア６２に対してデータを読み書きし、メモリ２６０のエリア６４にデータを書き込む。ＣＰＵ２０は、ソフトウェアロックステップを行う際、メモリ２６０のエリア６３に対してデータを読み書きし、メモリ２６０のエリア６５にデータを書き込む。ＣＰＵ２３０は、比較器２３１として、エリア６４，６５に書き込まれたデータを比較し、これらのデータが一致した場合には、ＡＳＩＬＤクラスのデータとして、当該データを使用する。
【００６７】
図１７は、一種類のＩＰコアに対してソフトウェアロックステップを適用する場合を表わしている。二種類のＩＰコアに対してソフトウェアロックステップを連続して適用する場合、二度の比較処理が必要になり、より具体的には、二度のソフトウェアロックステップのためのデータ通信のための帯域および比較時間が必要になる。そこで、本実施の形態においては、二種類のＩＰコアに対してソフトウェアロックステップを行なう場合に帯域および比較時間の増大が抑制される構成を説明する。
【００６８】
図１８は、ある局面に従う半導体装置４の構成の概略を表わすブロック図である。なお、前述の構成要素と同一の構成要素には同一の番号を付してある。当該構成要素の機能も同じである。したがって、当該構成要素の説明は繰り返さない。
【００６９】
半導体装置４は、図２に示される構成要素に加えて、画像処理部７０，７１をさらに備える。画像処理部７０，７１は、それぞれ、アービタ２５０に接続されている。画像処理部７０，７１は、ＡＳＩＬＤ非対応のＩＰ（例えば、ＡＳＩＬ　Ｂクラスの画像処理部）の一例であるが、ＡＳＩＬＤ非対応のＩＰは、画像処理部に限られない。
【００７０】
本実施の形態に係る半導体装置４は、ソフトウェアロックステップにおいて、一組目のＩＰ（すなわち、画像処理部７０，７１）による各計算結果を比較することなく、二組目のＩＰ（すなわち、ＣＰＵ１０，２０）に計算結果を渡すことで、一回の比較処理が省略

される。
【００７１】
図１９は、半導体装置４におけるデータの流れを表わす図である。ＣＰＵ２３０がデータをメモリ２６０のエリア６１に書き込むと、画像処理部７０，７１は、それぞれエリア６１からデータを読み出す。画像処理部７０は、エリア６２に対して、データの書き込みおよび読み出しを行なう。画像処理部７１は、エリア６３に対して、データの書き込みおよび読み出しを行なう。
【００７２】
さらに、画像処理部７０は、ソフトウェアロックステップのためのプログラムの実行結果をエリア６４に書き込む。画像処理部７０は、書込みが完了するとその旨をＣＰＵ１０に通知し、ソフトウェアロックステップのためのプログラムを実行させるためにＣＰＵ１０を起動させる。同様に、画像処理部７１は、ソフトウェアロックステップのためのプログラムの実行結果をエリア６５に書き込む。画像処理部７１は、書込みが完了するとその旨をＣＰＵ２０に通知し、ソフトウェアロックステップのためのプログラムを実行させるためにＣＰＵ２０を起動させる。
【００７３】
ＣＰＵ１０は、エリア６４からデータを読み出す。ＣＰＵ１０は、エリア６２に対してデータの書き込みおよび読み出しを行なう。さらに、ＣＰＵ１０は、画像処理部７０による実行結果を用いて、ソフトウェアロックステップのプログラムを実行し、実行結果をエリア６４に書き込む。ＣＰＵ２０は、エリア６５からデータを読み出す。ＣＰＵ２０は、エリア６３に対してデータの書き込みおよび読み出しを行なう。さらに、ＣＰＵ２０は、画像処理部７１による実行結果を用いて、ソフトウェアロックステップのプログラムを実行し、実行結果をエリア６５に書き込む。
【００７４】
ＣＰＵ２３０は、比較器２３１として、エリア６４，６５に書き込まれた各実行結果を比較し、これらが一致する場合には、ＡＳＩＬ　Ｄクラスのデータとして上記データを使用する。
【００７５】
なお、エリア６２、６４は、画像処理部７０とＣＰＵ１０とによって排他的に使用され得る。また、エリア６３、６５は、画像処理部７１とＣＰＵ２０とによって排他的に使用され得る。
【００７６】
ここで、図２０を参照して、半導体装置４が使用される局面の一例について説明する。図２０は、半導体装置４が実行する処理の一部を表わすフローチャートである。半導体装置４は、車両の周辺を撮影するためのカメラに接続されている。
【００７７】
ステップＳ２０１０にて、半導体装置４は、カメラからの画像信号を受信する。ステップＳ２０２０にて、画像処理部７０（または画像処理部７１）は、歪補正その他の画像処理を実行する。画像処理の結果は、エリア６４（またはエリア６５）に格納される。
【００７８】
ステップＳ２０３０にて、ＣＰＵ１０（またはＣＰＵ２０）は、人を検出したか否かを判断する。人の検出は、顔認識、特徴量算出その他の公知の技術を用いて実現される。検出結果は、エリア６４，６４に書き込まれる。ＣＰＵ１０（またはＣＰＵ２０）が人を検出したと判断すると（ステップＳ２０３０にてＹＥＳ）、ステップＳ２０４０にて、ＡＳＩＬ　ＤレベルのＣＰＵ２３０は、車両のブレーキを作動するための信号を、ブレーキの制御装置に送信する。その後、車両は停止する。
【００７９】
他方、ＣＰＵ１０（またはＣＰＵ２０）が人を検出しないと判断した場合には（ステップＳ２０３０にてＮＯ）、ステップＳ２０５０にて、ＣＰＵ１０、ＣＰＵ２０またはＣＰＵ２３０のいずれかは、予め定められたその他の処理を実行する。

【００８０】
以上のようにして、本実施の形態によれば、ソフトウェアロックステップの対象となるＣＰＵ、画像処理部その他のＩＰが二種類ある場合には、一組目のＩＰによるプログラムの実行結果は比較されることなく二組目のＩＰによるソフトウェアロックステップのプログラムの実行のために使用される。これにより、比較処理を１回に留めることができるので、一回目の実行結果の比較のためのデータ読み込み、データ書き込み、および比較処理のための時間が不要となり、半導体装置４の処理時間の増大化が防止され得る。
【００８１】
［第３の実施の形態］
以下、第３の実施の形態について説明する。前述の各実施の形態では、ソフトウェアロックステップのプログラムの実行結果が比較される。これに対して、本実施の形態は、実行結果以外のデータが比較される点で、前述の実施の形態と異なる。以下、実行結果に基づくＣＲＣ（Cyclic Redundancy Code）が比較される場合を説明するが、比較対象は、ＣＲＣに限られず、例えば、ＡＳＩＬ Ｄクラスの故障検出率を実現した上でバイト数が削減されるハッシュ等であってもよい。
【００８２】
図２１は、半導体装置５におけるデータフローを表わす図である。ある局面において、ＣＰＵ２３０は、データをエリア６１に書き込む。ＣＰＵ１０は、エリア６１から当該データを読み出し、エリア６２に対してデータの書き込みと読み出しとを行なう。ＣＰＵ２０は、エリア６１から当該データを読み出し、エリア６３に対してデータの書き込みと読み出しとを行なう。
【００８３】
ＣＰＵ１０は、エリア６２から読み出したデータを用いてソフトウェアロックステップのプログラムを実行し、その実行結果からＣＲＣを生成し、エリア６４にその生成したＣＲＣを書き込む。ＣＰＵ２０は、エリア６３から読み出したデータを用いてソフトウェアロックステップのプログラムを実行し、実行結果をエリア６５に書き込む。
【００８４】
ＣＰＵ２３０は、比較器２３１として、エリア６５から実行結果を読み出し、その読み出した実行結果からＣＲＣを生成する。さらに、比較器２３１は、エリア６４からＣＲＣを読み出し、その読み出したＣＲＣと、生成したＣＲＣとを比較する。これらのＣＲＣが一致すると、ＣＰＵ２３０は、ＡＳＩＬ Ｄ対応として上記データを使用する。
【００８５】
なお、上記のＣＲＣのデータサイズは、例えば、２５６バイトのデータについて２バイトであるが、データサイズは、これに限られない。
【００８６】
図２２および図２３を参照して、半導体装置５の動作について説明する。図２２は、半導体装置５におけるＣＰＵのプログラムの実行結果の出力を説明するための図である。ある局面において、いずれかのＣＰＵ（例えば、ＣＰＵ１０）は、ソフトウェアロックステップのプログラムを実行し、その実行結果からＣＲＣを生成し、エリア２６２（エリア６４に相当）に書き込む。他方のＣＰＵ（例えば、ＣＰＵ２０）は、ソフトウェアロックステップのプログラムを実行し、その実行結果をエリア２６３（エリア６５に相当）に書き込む。
【００８７】
図２３は、半導体装置５におけるソフトウェアロックステップのプログラムの実行結果の比較を説明するための図である。ＣＰＵ２３０は、エリア２６２からＣＲＣを読み出し、エリア２６３からデータを読み出し、その読み出したデータからＣＲＣを計算する。ＣＰＵ２３０は、比較器２３１として、読み出したＣＲＣと計算したＣＲＣとを比較する。
【００８８】
以上のようにして、本実施の形態によれば、二つのＣＰＵから出力される実行結果のうちのいずれかからＣＲＣが計算され、メモリ２６０の所定のエリアに書き込まれる。これ

により、メモリ２６０へのアクセスとして、ＣＲＣおよび実行結果の書き込みと、当該ＣＲＣおよび実行結果の読み出しと、当該ＣＲＣと、実行結果に基づいて計算されるＣＲＣとの比較の後におけるエリア６５からのデータの読出とが行なわれる。このような構成によれば、ＡＳＩＬ　Ｄクラスの故障検出率を保ったまま、ＣＲＣを用いない場合に比べて、帯域を１．５倍の増加に留めることができる。

【００８９】

［第４の実施の形態］

以下、第４の実施の形態について説明する。第４の実施の形態に従う半導体装置６は、ソフトウェアロックステップの実行結果の比較が、アービタで行なわれる点で、前述の実施の形態と異なる。

【００９０】

図２４は、半導体装置６の構成を表わすブロック図である。半導体装置６は、半導体装置２に示される構成に対して、アービタ２５０に代えてアービタ２５１を、処理部４０に代えて処理部４１を備える。アービタ２５１は、比較器２５２を含む。比較器２５２は、比較対象アドレス、範囲および比較用データベースアドレスのためのレジスタを有する。処理部４１は、アービタ２５１に接続されている。処理部４１は、例えば、ＡＳＩＬ　Ｄクラスであり、キャッシュを含み得る。ある局面において、処理部４１は、ＣＰＵであってもよい。

【００９１】

図２５は、半導体装置６におけるデータのフローを説明するための図である。ある局面において、ＣＰＵ２３０がデータをエリア６１に書き込むと、ＣＰＵ１０，２０は、それぞれエリア６１にアクセスし、データを読み出す。ＣＰＵ１０は、そのデータをエリア６２に書き込み、ＣＰＵ２０は、そのデータをエリア６３に書き込む。

【００９２】

ＣＰＵ１０は、エリア６２のデータを読み出して、ソフトウェアロックステップのプログラムを実行し、実行結果をエリア６４に書き込む。ＣＰＵ２０は、エリア６３のデータを読み出して、ソフトウェアロックステップのプログラムを実行し、実行結果をエリア６５に書き込む。

【００９３】

比較器２５２は、処理部４１からのリードリクエストを受信すると、当該リードリクエストから二つのリードリクエストを生成する。比較器２５２は、各リードリクエストに基づいて、エリア６４，６５にアクセスして、実行結果をそれぞれ読み出す。さらに、比較器２５２は、読み出した実行結果を比較し、比較結果が一致していることを確認すると、読み出した実行結果を処理部４１に送信する。

【００９４】

図２６は、半導体装置６の制御構造を説明するためのフローチャートである。

ステップＳ２６１０にて、ＣＰＵ１０，２０は、それぞれ、ソフトウェアロックステップのプログラムを実行し、実行結果を、メモリ２６０の所定のエリア６２，６３にそれぞれ書き込む。

【００９５】

ステップＳ２６２０にて、ＣＰＵ２３０は、アービタ２５２にアクセスして、比較器２５２の設定を行う。例えば、ＣＰＵ２３０は、比較対象アドレス、範囲、および比較用データベースアドレスをレジスタに設定する。

【００９６】

ステップＳ２６３０にて、ＡＳＩＬ　Ｄクラスの処理部４１は、データを所定の領域から読み出す。

【００９７】

ステップＳ２６４０にて、比較器２５２は、処理部４１からのリクエストに基づいて、エリア６４，６５から、それぞれデータを読み出す。例えば、比較器２５２は、エリア６４からＣＲＣを読み出し、エリア６５から実行結果を読み出す。ＣＲＣは、ＣＰＵ１０に

よるソフトウェアロックステップのプログラムの実行により生成されたものである。
【００９８】
ステップＳ２６５０にて、比較器２５２は、読み出したデータ（実行結果）からＣＲＣを計算し、計算で得られたＣＲＣと、読み出したＣＲＣとを比較する。
【００９９】
ステップＳ２６６０にて、比較器２５２は、これらのＣＲＣが一致するか否かを判断する。比較器２５２は、これらのＣＲＣが一致すると判断すると（ステップＳ２６６０にてＹＥＳ）、ステップＳ２６７０にて、比較器２５２は、上記リクエストの発信元である処理部４１に、実行結果を送信する。そうでない場合には（ステップＳ２６６０にてＮＯ）、ステップＳ２６８０にて、比較器２５２は、エラーを通知するために予め定められたエラー出力処理を実行する。
【０１００】
図２７を参照して、半導体装置６の動作についてさらに説明する。図２７は、半導体装置６における処理の一部を表わすフローチャートである。
【０１０１】
ステップＳ２７１０にて、半導体装置６は、図２６に示される比較器の処理を実行する。例えば、比較器２５２は、ＣＰＵ２３０から送られる設定データに基づき、メモリ２６０へのアクセスを管理するためのアドレス設定を行なう。
【０１０２】
ステップＳ２７１５にて、比較器２５２は、ＣＰＵ２３０からのリクエストトランザクションを待機する。
【０１０３】
ステップＳ２７２０にて、比較器２５２は、受信したリクエストトランザクションに含まれているアドレスと、ＣＲＣの比較のために設定されたアドレスとを比較する。ステップＳ２７２５にて、比較器２５２は、これらのアドレスが一致するか否かを判断する。比較器２５２は、これらのアドレスが一致すると判断すると、比較器２５２は、ステップＳ２６４０以降の処理を実行する。これらのアドレスが一致しない場合には、比較器２５２は、当該トランザクションの対象となるリクエストが通常のデータ読出リクエストであると判定し、ステップＳ２７３５にて、比較器２５２は、通常のデータ読出処理を実行する。
【０１０４】
図２８は、対象エリアに格納されているデータの読み出しを説明するための図である。ある局面において、半導体装置６の処理部４１は、比較器２５２にリードリクエストを送信する。比較器２５２は、当該リードリクエストから、メモリ２６０の所定の領域にアクセスするための２つのリードリクエストを生成する。さらに、比較器２５２は、二つのリードリクエストに基づいて、エリア２６２（エリア６４に相当）と、エリア２６３（エリア６５に相当）とにそれぞれアクセスする。
【０１０５】
図２９は、比較器２５２による比較の動作を説明するための図である。ある局面において、比較器２５２は、エリア２６２から読み出したデータと、エリア２６３から読み出したデータとに基づいて比較処理を実行する。ある局面において、比較器２５２は、ＣＰＵ１０によるソフトウェアロックステップのプログラムの実行結果と、ＣＰＵ２０によるソフトウェアロックステップのプログラムの実行結果とを比較し得る。別の局面において、いずれか一方ＣＰＵが当該実行結果からＣＲＣを生成している場合には、比較器２５２は、そのＣＲＣをメモリ２６０から読み出し、他方のＣＰＵによる実行結果からＣＲＣを生成し、その読み出したＣＲＣと、生成したＣＲＣとを比較し得る。比較器２５２は、データの一致を確認すると、当該データを処理部４１に送信する。
【０１０６】
以上のようにして、本実施の形態によれば、アービタ２５０が比較器２５２を備える。比較器２５２がメモリ２６０にアクセスしてデータの書き込みおよび読み出しを行なうこ

とにより、メモリ２６０にアクセスするために使用される帯域の使用量（２ライト＋２リード）を、ソフトウェアロックステップを行なわない通常の動作のために使用される帯域の使用量（２ライト＋２リード）よりも削減することができるので、比較処理によるＣＰＵリソースの減少を回避できる。

【０１０７】

なお、上記の実施の形態は、第３の実施の形態と組み合わされてもよい。このようにすると、メモリの帯域は、実質的に１倍まで削減でき、処理部４１が、ＣＲＣの復号を行う必要がなくなるので、処理速度が向上し得る。

【０１０８】

上記開示された技術的特徴は、以下のように要約され得る。

（構成１） ある局面に従う半導体装置は、キャッシュを有し、ソフトウェアロックステップと通常動作とを実行するための第１のプロセッサ（例、ＣＰＵ１）および第２のプロセッサ（ＣＰＵ２０）と、メモリ２６０と、当該メモリに接続され、アドレスプロテクト機能を有するアービタ２５０と、当該アービタに接続されたバス３０と、当該第１のプロセッサと当該バスとに接続され、当該第１のプロセッサによる当該第２のプロセッサのキャッシュへのスヌープ動作を制御するための第１のスヌープ制御回路（例えば、ＳＰＵ２１０）と、当該第２のプロセッサと当該バスとに接続され、当該第２のプロセッサによる当該第１のプロセッサのキャッシュへのスヌープ動作を制御するための第２のスヌープ制御回路（例えば、ＳＰＵ２２０）と、当該ソフトウェアロックステップと当該通常動作とを切り替えるための制御装置（例えば、ＣＰＵ２３０）とを備える。当該制御装置は、当該通常動作が実行される場合に、当該第１のスヌープ制御回路および当該第２のスヌープ制御回路に対して、各当該スヌープ動作を許可し、当該ソフトウェアロックステップが実行される場合に、当該第１のスヌープ制御回路および当該第２のスヌープ制御回路に対して、各当該スヌープ動作を禁止する。当該第１のプロセッサは、当該ソフトウェアロックステップのための第１のソフトウェアを実行し、当該第１のプロセッサのために当該メモリに確保された第１の領域に、当該第１のソフトウェアの実行結果を書き込む。当該第２のプロセッサは、当該ソフトウェアロックステップのための第２のソフトウェアを実行し、当該第１の領域と異なる第２の領域に、当該第２のソフトウェアの実行結果を書き込む。当該半導体装置は、当該第１の領域に書き込まれた実行結果と、当該第２の領域に書き込まれた実行結果とを比較する比較器をさらに備える。

【０１０９】

（構成２） ある局面に従うと、当該第１のスヌープ制御回路または当該第２のスヌープ制御回路は、所定の条件を満たさない異常なスヌープ動作を禁止する。当該アービタは、当該制御装置によってアクセス可能なメモリ領域に対する当該第１のプロセッサまたは当該第２のプロセッサによるアクセスを禁止する。

【０１１０】

（構成３） ある局面に従う半導体装置は、当該アービタに接続された第１の処理部および第２の処理部をさらに備える。当該第１のプロセッサが当該第１のソフトウェアを実行することは、当該第１の処理部による演算結果に基づいて、ソフトウェアロックステップのための第３のソフトウェアを実行することと、当該第３のソフトウェアの実行結果に基づいて当該第１のソフトウェアを実行することとを含む。当該第２のプロセッサが当該第２のソフトウェアを実行することは、当該第２の処理部による演算結果に基づいて、ソフトウェアロックステップのための第４のソフトウェアを実行することと、当該第４のソフトウェアの実行結果に基づいて当該第２のソフトウェアを実行することとを含む。

【０１１１】

（構成４） ある局面に従うと、当該第１のプロセッサは、当該第１のソフトウェアの実行結果に基づいてＣＲＣ（Cyclic Redundancy Code）を生成するように構成されている。当該制御装置は、当該第２のソフトウェアの実行結果に基づいてＣＲＣを生成し、当該第１のプロセッサによって生成されたＣＲＣと、当該制御装置によって生成されたＣＲＣとを比較するように構成されている。

【０１１２】
（構成５）　ある局面に従うと、当該比較器は、当該アービタに含まれており、当該メモリに格納されているデータの読出要求に基づいて、当該メモリに格納されている各実行結果を比較するように構成されている。
【０１１３】
（構成６）　ある局面に従うと、当該メモリは、データを格納するための複数の領域（例えば、エリア２６１，２６２，２６３，２６４）を含む。当該第１のプロセッサに割り当てられる領域は、当該第２のプロセッサによる当該実行結果の書き込みから保護されている。当該第２のプロセッサに割り当てられる領域は、当該第１のプロセッサによる当該実行結果の書き込みから保護されている。当該制御装置に割り当てられる領域は、当該第１のプロセッサおよび当該第２のプロセッサによる書き込みから保護されている。
【０１１４】
（構成７）　ある局面に従うと、当該制御装置の機能安全レベルは、当該第１のプロセッサおよび当該第２のプロセッサの機能安全レベルよりも高い。
【０１１５】
（構成８）　他の実施の形態に従うと、半導体装置を備える制御システムが提供される。この制御システムは、信号の入力を受ける入力インターフェイスと、当該信号に基づいて演算を実行する半導体装置と、当該演算の実行結果を出力するための出力インターフェイスとを備える。当該半導体装置は、キャッシュを有し、ソフトウェアロックステップと通常動作とを実行するための第１のプロセッサ１０および第２のプロセッサ２０と、メモリ２６０と、当該メモリに接続され、アドレスプロテクト機能を有するアービタ２５０と、当該アービタに接続されたバス３０と、当該第１のプロセッサと当該バスとに接続され、当該第１のプロセッサによる当該第２のプロセッサのキャッシュへのスヌープ動作を制御するための第１のスヌープ制御回路（例えば、ＳＰＵ２１０）と、当該第２のプロセッサと当該バスとに接続され、当該第２のプロセッサによる当該第１のプロセッサのキャッシュへのスヌープ動作を制御するための第２のスヌープ制御回路（例えば、ＳＰＵ２２０）と、当該ソフトウェアロックステップと当該通常動作とを切り替えるための制御装置とを備える。当該制御装置は、当該通常動作が実行される場合に、当該第１のスヌープ制御回路および当該第２のスヌープ制御回路に対して、各当該スヌープ動作を許可し、当該ソフトウェアロックステップが実行される場合に、当該第１のスヌープ制御回路および当該第２のスヌープ制御回路に対して、各当該スヌープ動作を禁止する。当該第１のプロセッサは、当該ソフトウェアロックステップのための第１のソフトウェアを実行し、当該第１のプロセッサのために当該メモリに確保された第１の領域に、当該第１のソフトウェアの実行結果を書き込む。当該第２のプロセッサは、当該ソフトウェアロックステップのための第２のソフトウェアを実行し、当該第１の領域と異なる第２の領域に、当該第２のソフトウェアの実行結果を書き込む。当該半導体装置は、当該第１の領域に書き込まれた実行結果と、当該第２の領域に書き込まれた実行結果とを比較する比較器をさらに備える。
【０１１６】
（構成９）　さらに他の局面に従うと、半導体装置の制御方法が提供される。この制御方法は、キャッシュを有し、ソフトウェアロックステップと通常動作とを実行するための第１および第２のプロセッサを準備するステップと、当該第１のプロセッサによる当該第２のプロセッサのキャッシュへのスヌープ動作を制御するステップと、当該第２のプロセッサによる当該第１のプロセッサのキャッシュへのスヌープ動作を制御するステップと、当該通常動作が実行される場合に、各当該スヌープ動作を許可するステップと、当該ソフトウェアロックステップが実行される場合に、各当該スヌープ動作を禁止するステップと、当該第１のプロセッサが、当該ソフトウェアロックステップのための第１のソフトウェアを実行するステップと、当該第１のプロセッサのためにメモリに第１の領域を確保するステップと、当該第２のプロセッサのために当該第１の領域と異なる第２の領域を確保するステップと、当該第１のプロセッサが、当該第１の領域に、当該第１のソフトウェアの実行結果を書き込むステップと、当該第２のプロセッサが、当該ソフトウェアロックステ

ップのための第２のソフトウェアを実行するステップと、当該第２のプロセッサが、当該第２の領域に、当該第２のソフトウェアの実行結果を書き込むステップと、制御装置が、当該第１の領域に書き込まれた実行結果と、当該第２の領域に書き込まれた実行結果とを比較するステップとを含む。

【０１１７】

＜実施の形態の効果＞

上記の各実施の形態に従うソフトウェアロックステップによれば、機能安全レベルが通常レベルのＣＰＵ１０，２０は、互いに同期をとることなく、アプリケーションあるいは関数単位で処理を実行し、ＡＳＩＬ　ＤクラスのＣＰＵ２３０が、当該処理結果を比較する。このようにすると、通常の動作モードからデュアルコアロックステップを行なうモードへの切り替えに必要とされるコア同士の同期をとるためのリセット処理等が不要になり、数ミリ秒の処理停止時間が生じなくなる。

【０１１８】

なお、上述の実施の形態では、主として、ＡＳＩＬ　Ｄレベルあるいは ＡＳＩＬ　Ｂレベルの例を用いているが、本開示に係る技術思想は、他の機能安全レベル（例えば、ＡＳＩＬ　Ａレベル、あるいは、ＡＳＩＬ　Ｃレベル）にも適用可能である。

【０１１９】

以上、本発明者によってなされた発明を実施の形態に基づき具体的に説明したが、本発明は上記実施の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ことはいうまでもない。

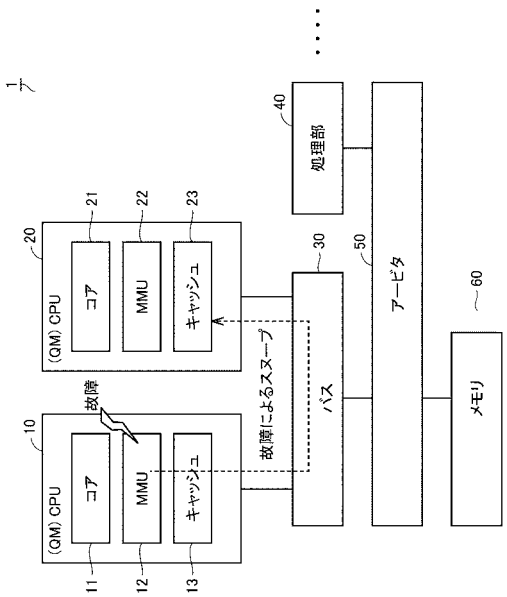
【符号の説明】

【０１２０】

１，２，３，４，５，６　半導体装置、１１，２１　コア、１３，２３　キャッシュ、３０　バス、４０，４１　処理部、５０，２５０　アービタ、６０，２６０　メモリ、６１，６２，６３，６４，６５，２５２，２６１，２６２，２６３，２６４　エリア、７０，７１　画像処理部、２３１，２５２　比較器、３１０　強制無効化ブロック、３２０　レジスタ。

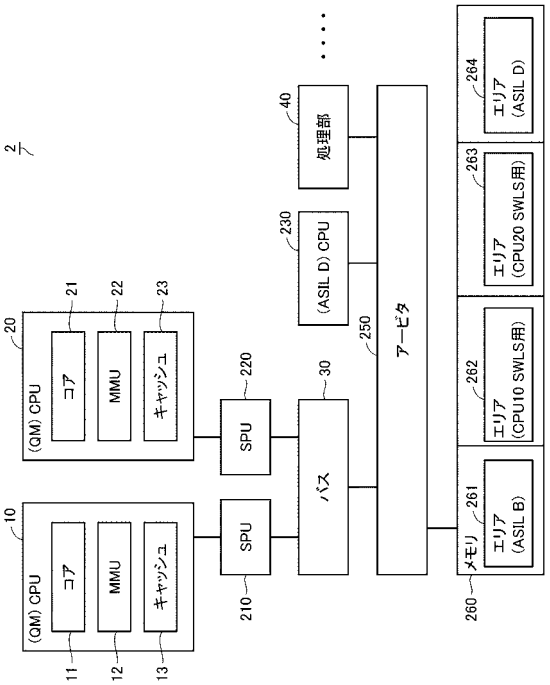
【図１】

図1



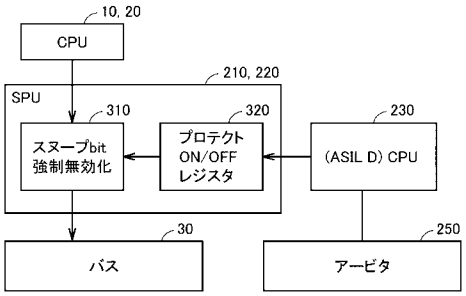
【図２】

図2



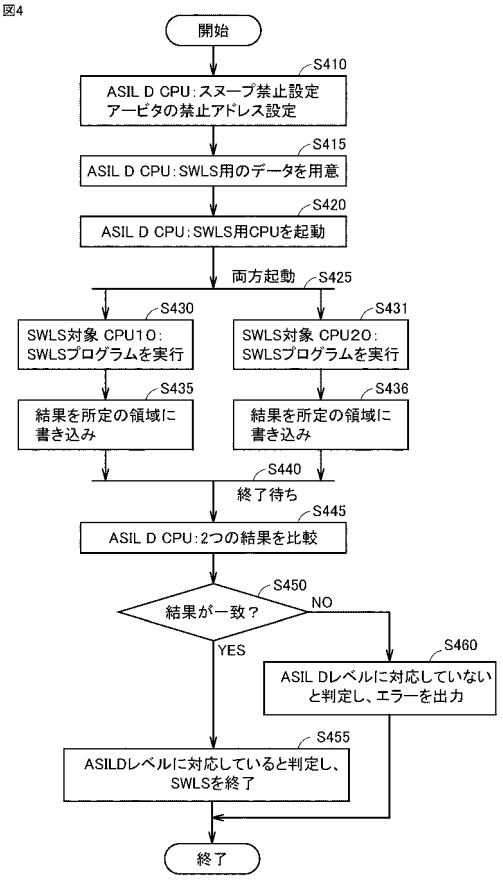
【図３】

図3



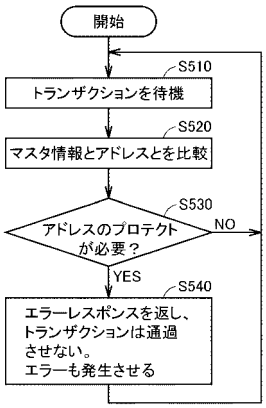
【図４】

図4



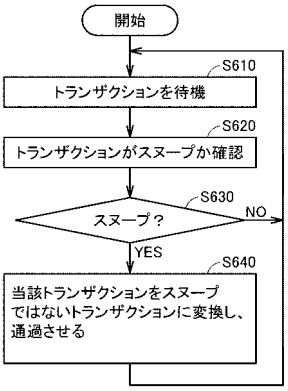
【図５】

図5



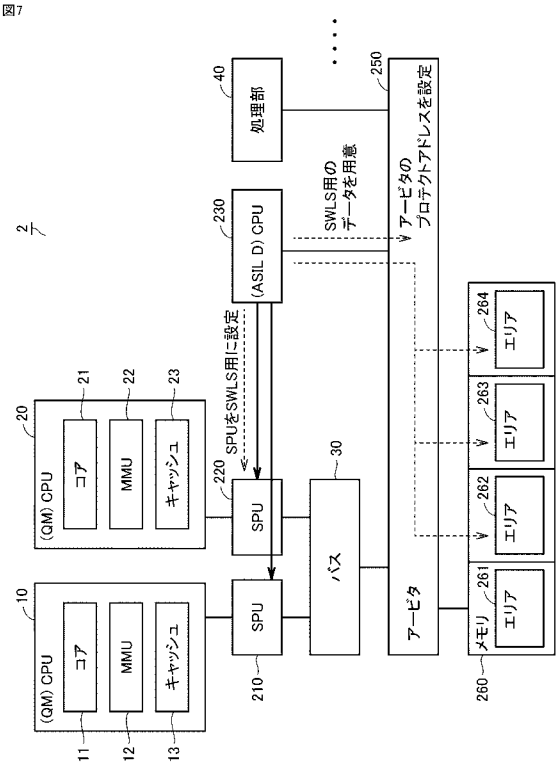
【図６】

図6



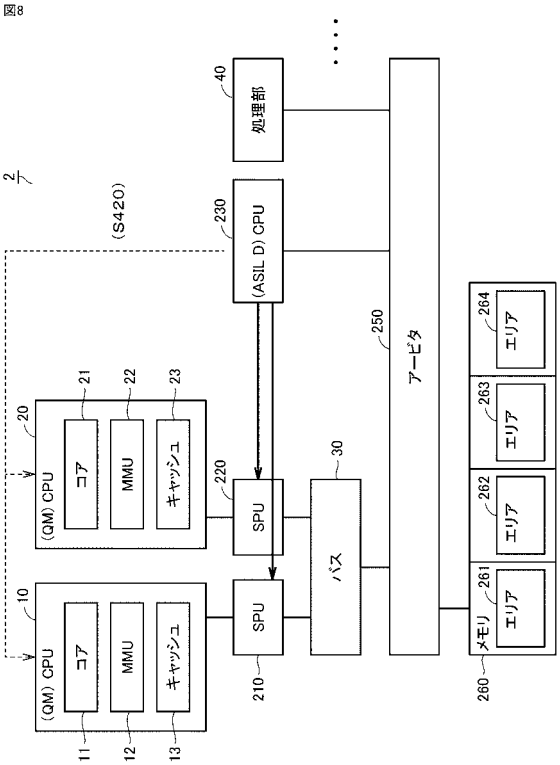
【図７】

図7



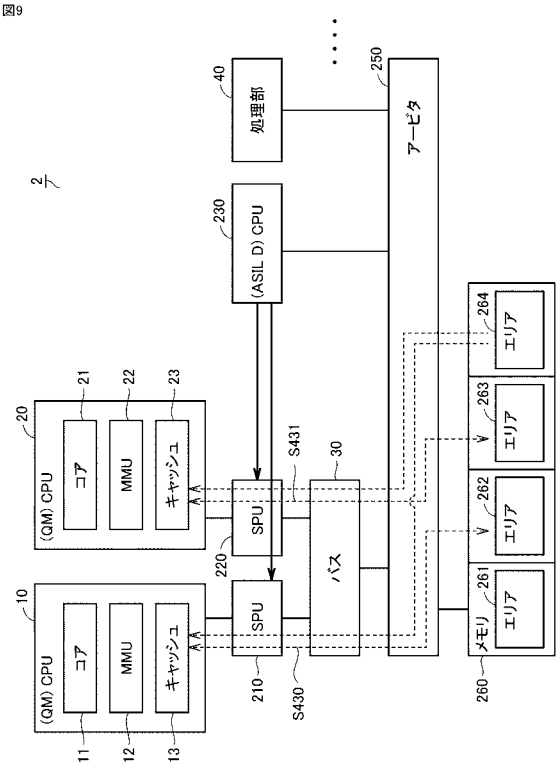
【図８】

図8



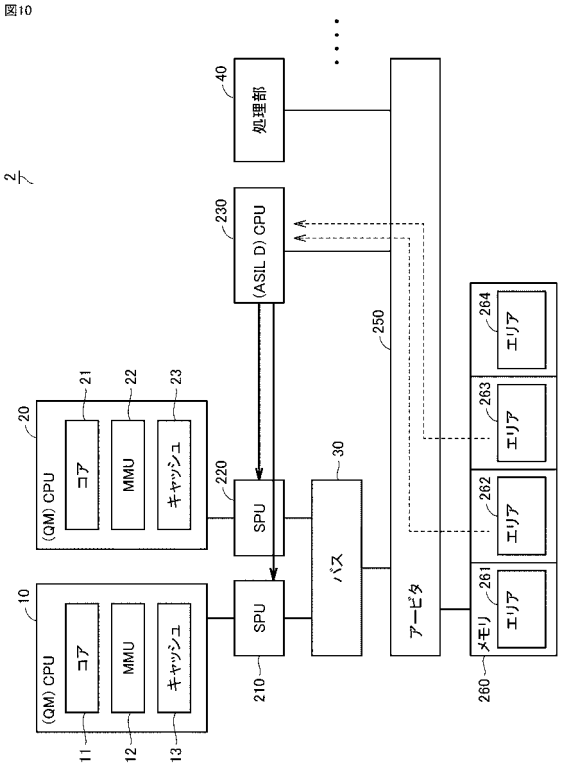
【図９】

図9



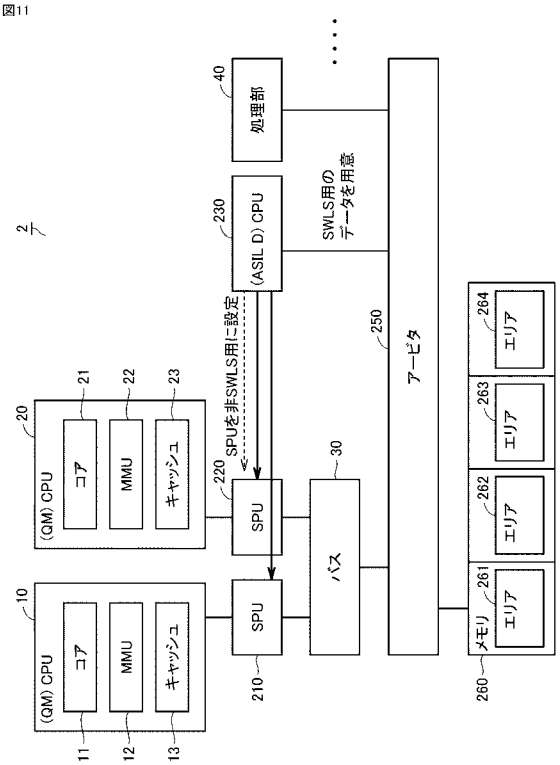
【図１０】

図10



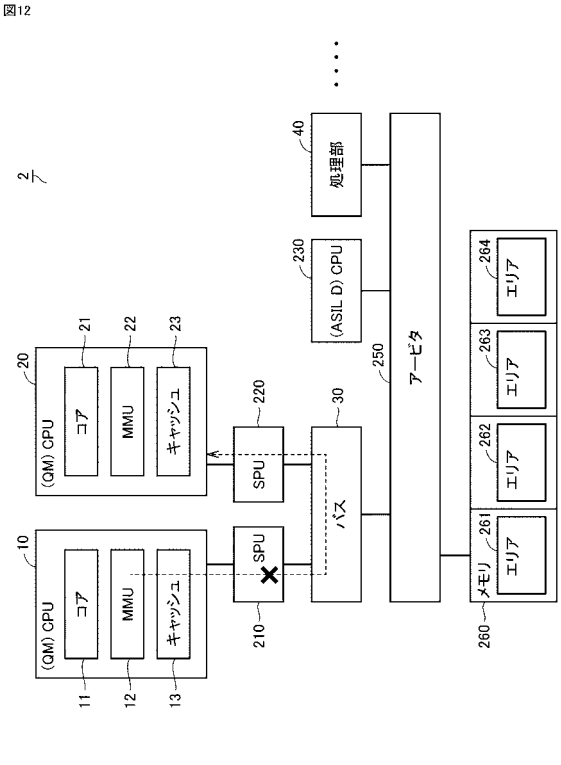
【図１１】

図11



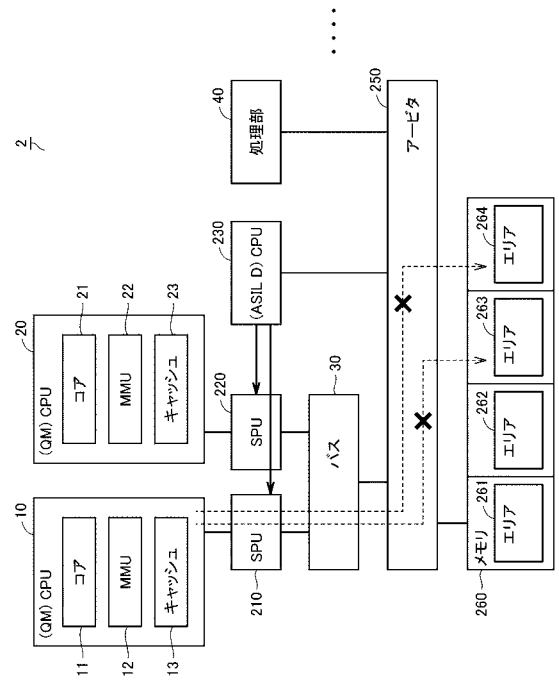
【図１２】

図12



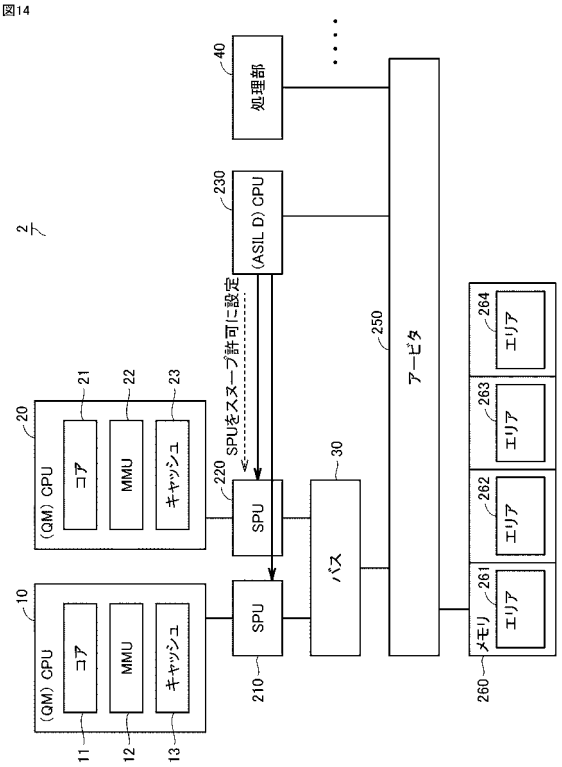
【図１３】

図13



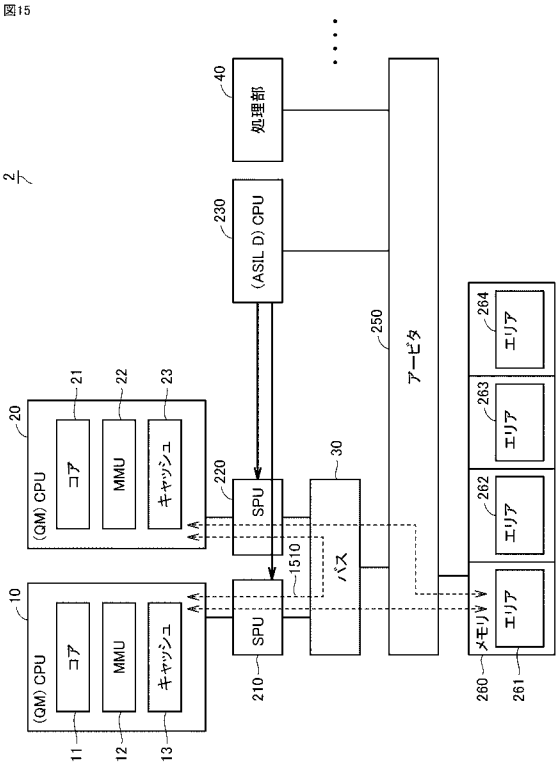
【図１４】

図14



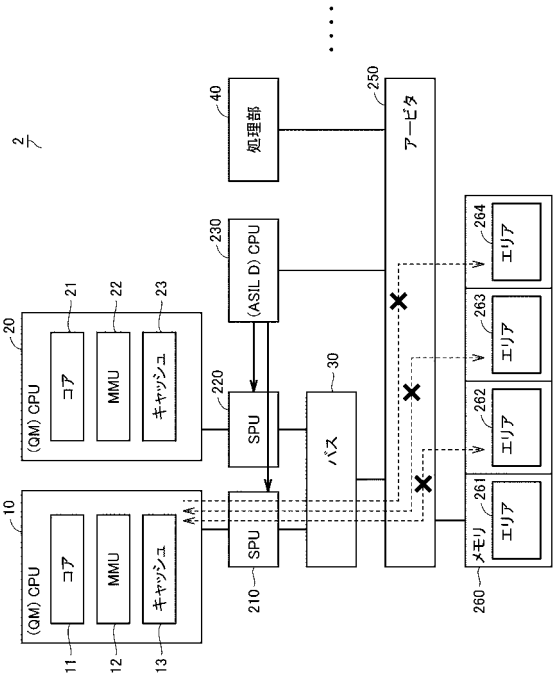
【図１５】

図15



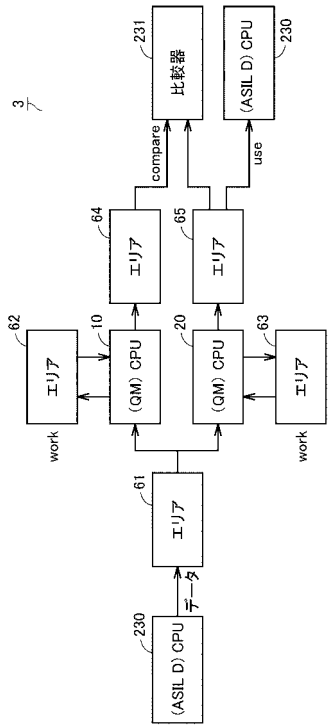
【図１６】

図16



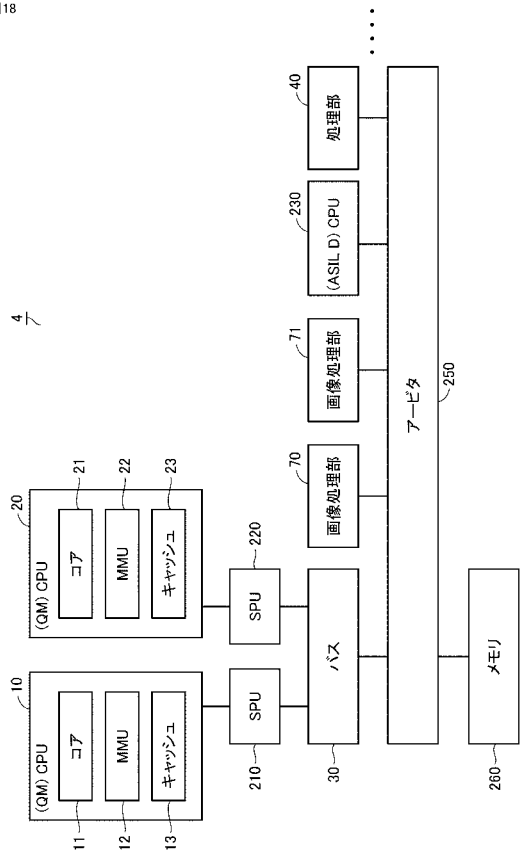
【図１７】

図17



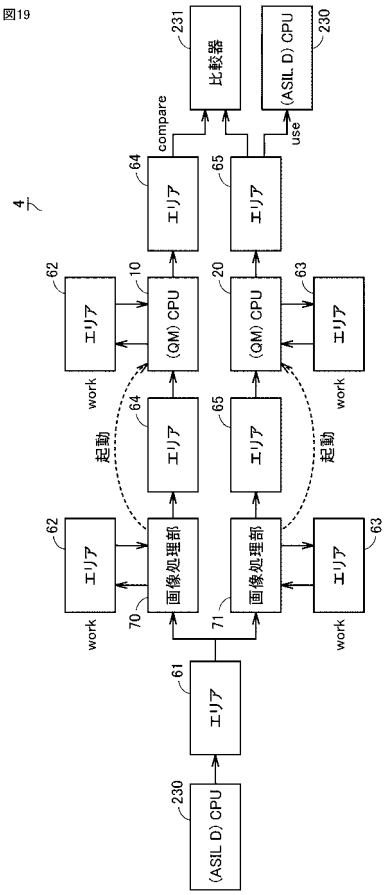
【図１８】

図18



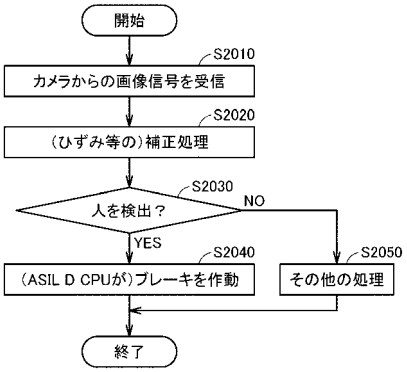
【図１９】

図19



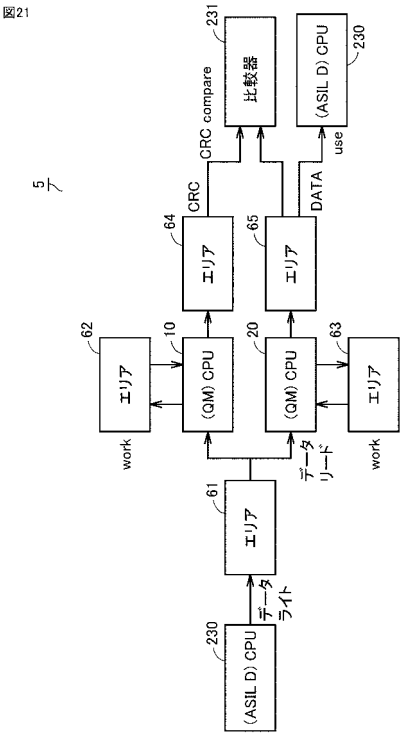
【図２０】

図20



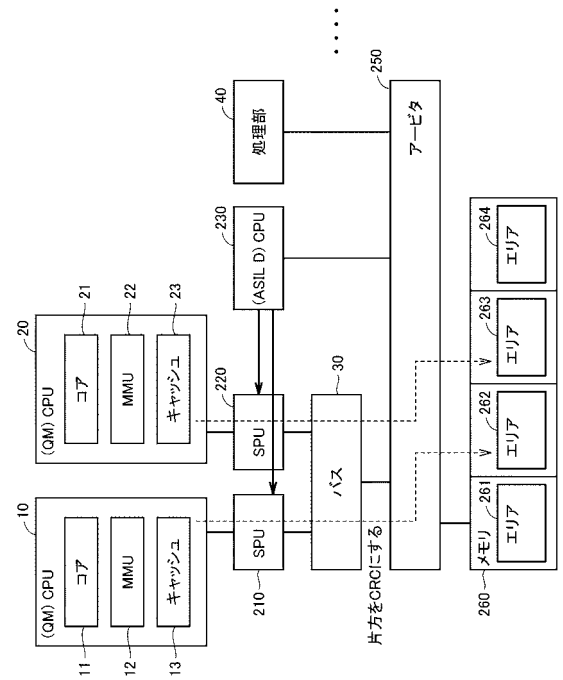
【図２１】

図21



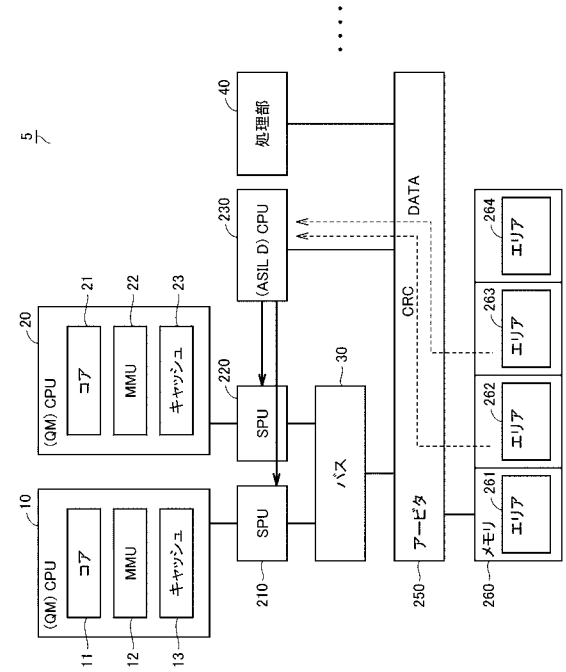
【図２２】

図22



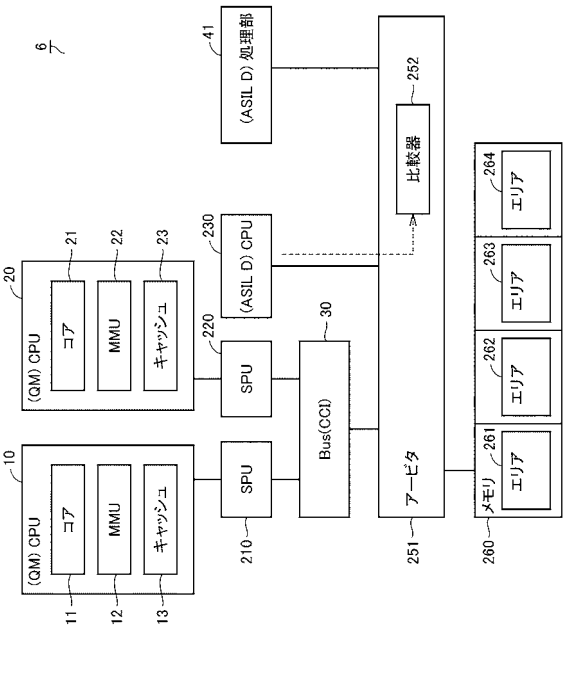
【図２３】

図23



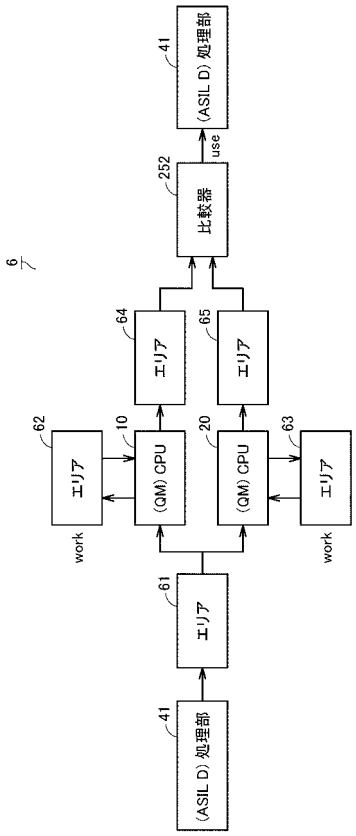
【図２４】

図24



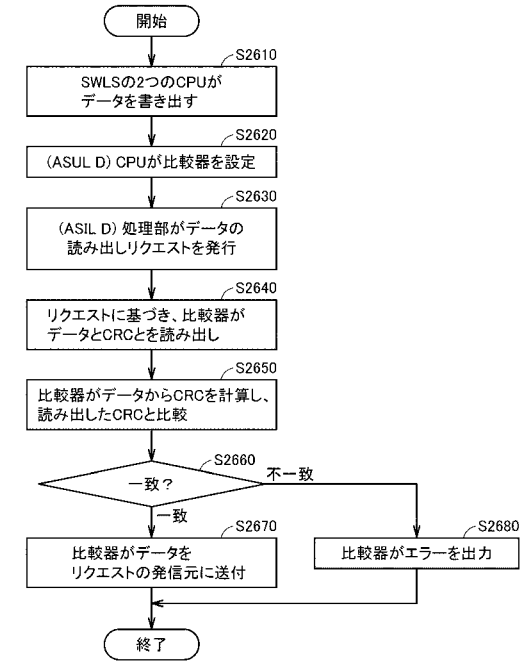
【図２５】

図25



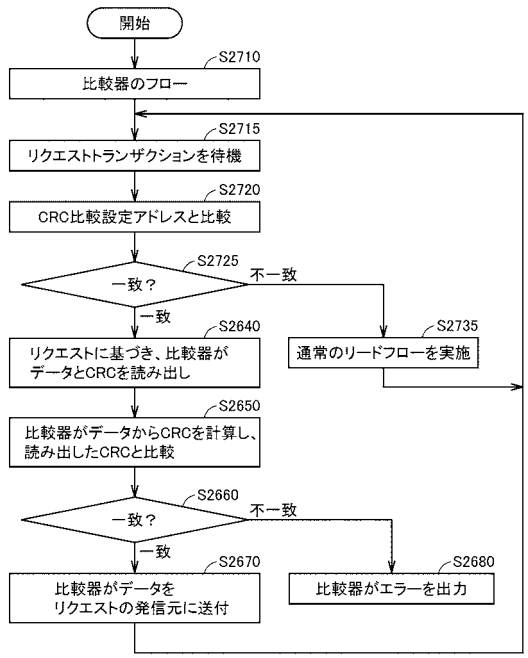
【図２６】

図26



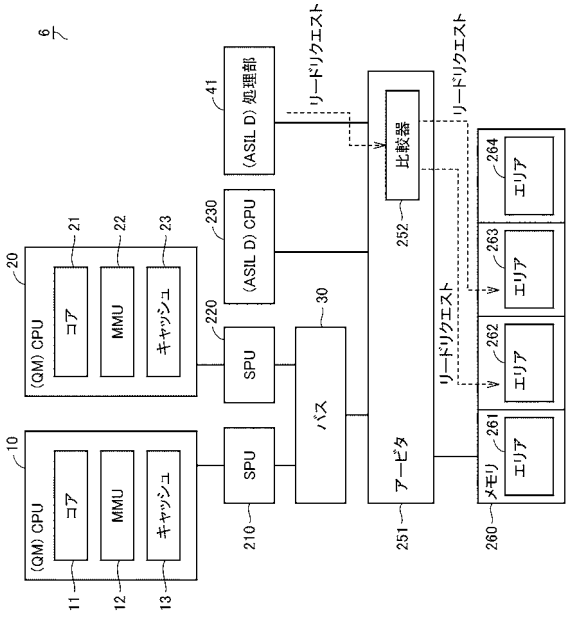
【図２７】

図27



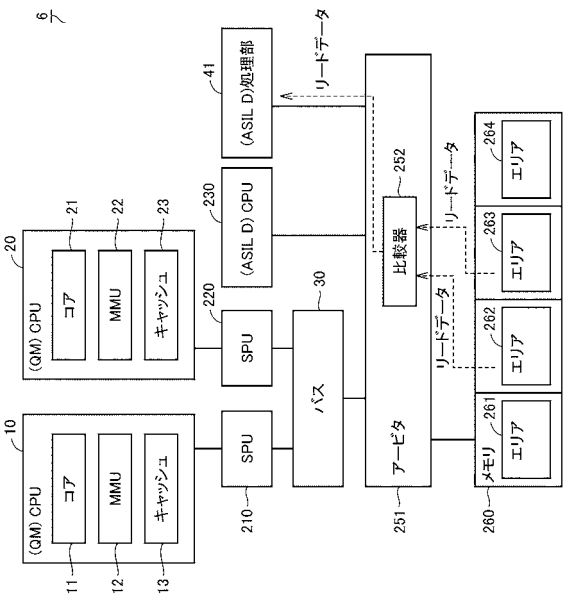
【図２８】

図28



【図２９】

図29


6
(QM) CPU
コア
MMU
キャッシュ
11
12
13
10
20
21
22
23
SPU
210
220
(ASIL D) CPU
230
(ASIL D)処理部
41
バス
30
アービタ
251
比較器
252
リードデータ
メモリ
260
エリア
261
262
263
264