(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7380406号
(P7380406)

(45)発行日　令和5年11月15日(2023.11.15)

(24)登録日　令和5年11月7日(2023.11.7)

(51)国際特許分類

		FI		
G06F	*9/48 (2006.01)*	G06F	9/48	300C
G06F	*9/38 (2018.01)*	G06F	9/38	370X
G06F	*15/167 (2006.01)*	G06F	15/167	615Z

請求項の数　4　（全24頁）

(21)出願番号　特願2020-79203(P2020-79203)
(22)出願日　令和2年4月28日(2020.4.28)
(65)公開番号　特開2021-174365(P2021-174365A)
(43)公開日　令和3年11月1日(2021.11.1)
審査請求日　令和4年8月10日(2022.8.10)

(73)特許権者　000004260
株式会社デンソー
愛知県刈谷市昭和町１丁目１番地
(74)代理人　110000567
弁理士法人サトー
(72)発明者　田中 伸幸
愛知県刈谷市昭和町１丁目１番地 株式会社デンソー内
審査官　漆原 孝治

最終頁に続く

(54)【発明の名称】　リアルタイム演算処理装置

(57)【特許請求の範囲】

【請求項１】

外部から演算イベントが発行されると時間軸及び優先度に基づいて実行順序を判断し前記演算イベントに係るタスクを実行指令すると共に前記タスクの管理を行うことに特化したオペレーティングシステム処理部（３）と、

前記オペレーティングシステム処理部（３）の実行指令に従って演算処理を実行することに特化した演算処理部（６）とを、

同一のＩＣチップ内にそれぞれハードウェア上で独立して構成され、

前記タスクに紐付いた開始アドレス及び終了アドレスを記憶するメッセージリスト（４）と、

演算命令セットを記憶する演算命令テーブル（５）と、を備え、

前記演算処理部は、プログラムカウンタ（４１）及び演算命令デコーダ（３２）を備え、前記オペレーティングシステム処理部（３）から前記演算イベントに係る前記タスクの実行指令を受付けると、前記タスクに紐付いた前記開始アドレス及び前記終了アドレスをメッセージリスト（４）から受領し、前記開始アドレスから前記終了アドレスにかけて前記プログラムカウンタ（４１）のカウンタ値の示すアドレスにおける前記演算命令テーブル（５）の演算命令セットを参照して演算処理を実行するものであり、

前記オペレーティングシステム処理部（３）と前記演算処理部（６）との間のインタフェースとして、

前記タスクの非実行状態（ＳＴ１、ＳＴ３、ＳＴ４）、実行状態（ＳＴ２、ＳＴ５）の状

態の間の予め定められた遷移を、前記オペレーティングシステム処理部（３）及び前記演算処理部（６）から共に認識可能にするタスクトレイ（２５）を備えるリアルタイム演算処理装置。
【請求項２】
前記タスクトレイは、前記オペレーティングシステム処理部（３）及び前記演算処理部の双方から書込可能なタスクトレイ状態（TTRAY__ST）の格納領域を備え、
前記タスクトレイ状態の書込みアクセス権は、前記オペレーティングシステム処理部（３）及び前記演算処理部（６）の間で競合を避けるように設定され、
前記演算処理部による前記タスクの非実行状態では、前記オペレーティングシステム処理部（３）にアクセス権が付与され、
前記演算処理部による前記タスクの実行状態では、前記演算処理部にアクセス権が付与される請求項１記載のリアルタイム演算処理装置。
【請求項３】
前記演算処理部（６）が最優先となる第１優先度の前記タスクを実行中に、前記オペレーティングシステム処理部（３）が前記第１優先度よりも低い第２優先度のタスクに係る演算イベントが発生した場合、
前記オペレーティングシステム処理部が前記第２優先度の前記タスクを前記演算処理部に対して実行指令に移さないようにした請求項２記載のリアルタイム演算処理装置。
【請求項４】
前記演算処理部（６）が最優先よりも低い第３優先度の前記タスクを実行中に、前記オペレーティングシステム処理部（３）に前記第３優先度よりも高い第４優先度のタスクに係る演算イベントが発生した場合、
前記オペレーティングシステム処理部が前記第４優先度の前記タスクを前記第３優先度よりも優先して前記演算処理部に実行指令するようにした請求項２記載のリアルタイム演算処理装置。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、リアルタイム演算処理装置に関する。
【背景技術】
【０００２】
従来より、広く使用されているマイコンは、主にＣＰＵ、メモリ、及び周辺回路を用いて構成されている。従来、開発者は、所望の処理内容をソフトウェアとしてメモリに実装し、マイコンがメモリに記憶されたソフトウェアを実行することで各種処理を実現できる。この手法はソフトウェアをアップデートすることで比較的簡単に処理を変更でき、フレキシビリティ性を確保できる。
【０００３】
従来の一般的なマイコンの場合、ＲＯＭにアプリケーション、ＯＳ（オペレーティングシステム）、割込みハンドラといったソフトウェアであったり、割込み時のジャンプ先割込みハンドラのジャンプアドレスがマシンコード化されて実装される。このときマイコンに構成されるＣＰＵは、プログラムカウンタに基づいて命令系アドレスバスが指し示すＲＯＭ内の命令コードを命令系データバスを経由して読み出すと共にＣＰＵ内命令レジスタに格納する。ＣＰＵ内命令デコーダ部が命令レジスタに格納された命令コードを各種制御信号に置換し、ＣＰＵ内汎用レジスタ、ＡＬＵ（Arithmetic Logic Unit：算術論理演算装置）や、ＲＡＭ、その他の周辺機能を制御するようにしている。
【０００４】
特許文献１記載のように、ＣＰＵのマルチコア技術が提案されており、処理能力改善の手法を提供している。しかし、特許文献１記載の構成では、マルチコア構成を採用しているため、逆に回路規模が大きくなってしまうという事情がある。
【先行技術文献】

【特許文献】
【０００５】
【文献】特許５４５３８２５号公報
【発明の概要】
【発明が解決しようとする課題】
【０００６】
本開示の目的は、回路規模の増大を極力抑制しつつ演算イベント割込に迅速に対応できるようにしたリアルタイム演算処理装置を提供することにある。
【課題を解決するための手段】
【０００７】
請求項１記載の発明によれば、オペレーティングシステム処理部（３）と、演算処理部（６）とを同一のＩＣチップ内にそれぞれハードウェア上で独立して構成している。オペレーティングシステム処理部（３）及び演算処理部（６）は独立に処理可能になる。
【０００８】
ここで、オペレーティングシステム処理部（３）は、外部から演算イベントが発行されると時間軸及び優先度に基づいて実行順序を判断し演算イベントに係るタスクを実行指令すると共にタスクの管理を行うことに特化している。また演算処理部（６）は、オペレーティングシステム処理部（３）の実行指令に従って演算処理を実行することに特化している。
【０００９】
また請求項１記載の発明は、タスクに紐付いた開始アドレス及び終了アドレスを記憶するメッセージリスト（４）と、演算命令セットを記憶する演算命令テーブル（５）とを備えている。
【００１０】
演算処理部は、プログラムカウンタブロック（３１）及び演算命令デコーダ（３２）を備え、オペレーティングシステム処理部（３）から前記演算イベントに係るタスクの実行指令を受付けると、タスクに紐付いた開始アドレス及び前記終了アドレスをメッセージリスト（４）から受領し、開始アドレスから終了アドレスにかけてプログラムカウンタ（４１）のカウンタ値の示すアドレスにおける演算命令テーブル（５）の演算命令セットを参照して演算処理を実行する。
【００１１】
このため、オペレーティングシステム処理部（３）及び演算処理部（６）は各ハードウェア構成毎に処理内容を分担して並行処理できるようになり、回路規模の増大を極力抑制しつつ演算イベント割込に迅速に対応できるようになる。
【００１２】
また、オペレーティングシステム処理部（３）と前記演算処理部（６）との間のインタフェースとして、演算タスクの非実行状態（ＳＴ１、ＳＴ３、ＳＴ４）、実行状態（ＳＴ２、ＳＴ５）の状態の間の予め定められた遷移を、オペレーティングシステム処理部（３）及び演算処理部（６）から共に認識可能にするタスクトレイ（２５）を備える。
【００１３】
請求項１記載の発明によれば、タスクトレイは、演算タスクの非実行状態、実行状態の状態の間の予め定められた遷移を、オペレーティングシステム処理部及び演算処理部から共に認識可能にしている。このため、オペレーティングシステム処理部及び演算処理部は、タスクトレイにおける演算タスクの非実行状態、実行状態の間の遷移を認識でき、演算タスクの非実行状態／実行状態を互いにハンドシェークできる。
【図面の簡単な説明】
【００１４】
【図１】第１実施形態に係るリアルタイム演算処理装置のハードウェア構成図
【図２】ＦＰＵＯＳ機能ブロックのハードウェア構成を概略的に説明する図
【図３】タスクトレー状態における状態遷移図

【図４】ＦＰＵＯＳ機能ブロックとプログラムカウンタブロックとのインターフェース構造を概略的に説明する電気的構成図
【図５】演算タスク処理の流れを説明するタイミングチャートのその１
【図６】演算タスク処理の流れを説明するタイミングチャートのその２
【図７】演算タスク処理の流れを説明するタイミングチャートのその３
【図８】第２実施形態に係るリアルタイム演算処理装置のハードウェア構成図
【図９】ウォッチドッグタイマのカウント値の変化を概略的に示すタイミングチャート
【発明を実施するための形態】
【００１５】
以下、幾つかの実施形態について図面を参照しながら説明する。以下の説明では、各実施形態で説明した構成と同一又は類似機能を備えた構成について同一符号又は類似符号を付し、第２実施形態以降では、必要に応じて説明を省略する。
【００１６】
（第１実施形態）
図１から図７に第１実施形態の説明図を示す。まず図１を参照し、リアルタイム演算処理装置１の全体のハードウェア構成を説明する。リアルタイム演算処理装置１は、ＦＰＵＯＳ機能ブロック３（以下、ＦＰＵＯＳ３と略す：オペレーティングシステム処理部相当）、メッセージリスト４、演算命令テーブル５、ＷＯＲＫＥＲ６（演算処理部相当）、ＦＰＵ第１引数汎用レジスタ群７、ＦＰＵ第２引数汎用レジスタ群８、及びＦＰＵ９（浮動小数点演算処理部相当）を、同一のＡＳＩＣ内、すなわちＩＣチップ内に構成している。なお、ハードウェア構成を監視するためのウォッチドッグタイマ２（図８参照）も設けられているが、後述の第２実施形態にて説明する。
【００１７】
リアルタイム演算処理装置１は、外部の上位のスケジューラ１０から演算イベントを受領すると、演算命令テーブル５の中に予め設定された２項演算命令に従って浮動小数点演算処理を実行する。
【００１８】
メッセージリスト４は、演算イベントに係るタスクの識別符号（ＩＤ）であるタスクＩＤと、当該タスクＩＤと紐付けられた開始アドレス及び終了アドレスとをＦＰＵＯＳ３の外にリスト化して格納している（図２も参照）。ＦＰＵＯＳ３が、タスクＩＤを指定すると（図１のタスクＩＤ指定Ｓ８）、指定したタスクＩＤに対応した開始アドレス及び終了アドレスをメッセージリスト４の選択結果Ｓ９として返す。
【００１９】
また図１に示す演算命令テーブル５には演算命令テーブルアドレスに対応して演算命令セットが展開されている。ＷＯＲＫＥＲ６が、プログラムカウンタブロック３１から演算命令テーブル５に演算命令テーブルアドレスＳ１０を指定すると、当該演算命令テーブルアドレスＳ１０に対応した演算命令セットを演算命令セット選択結果Ｓ１１として返す。
【００２０】
ＦＰＵＯＳ３は、ＦＰＵ９に係るオペレーティングシステム機能をＡＳＩＣの中のハードウェアブロックにより実現したブロックである。外部のスケジューラ１０が演算イベントを発行すると、ＦＰＵＯＳ３はスケジューラ１０から演算イベントを受領する。
【００２１】
ＦＰＵＯＳ３は、演算イベントに係るタスクを時間軸及び優先度に基づいて実行順序を判断する。ＦＰＵＯＳ３は、タスクに付与されたタスクＩＤと紐付けてメッセージリスト４から演算開始アドレス及び演算終了アドレスを読出し、ＷＯＲＫＥＲ６にタスクを実行指令する。図１に示すタスク実行指令Ｓ５参照。なおＷＯＲＫＥＲ６は、タスク実行指令Ｓ５を受領し、当該タスクの実行を完了するとタスク完了Ｓ６として返す。
【００２２】
またＦＰＵＯＳ３は、前述のようにタスクを実行指令すると共に、ＷＯＲＫＥＲ６により実行されているタスクの管理を行うことに特化した機能ブロックである。ＦＰＵＯＳ３

は、オペレーティングシステム処理部相当として機能する。またＷＯＲＫＥＲ６は、ＦＰＵＯＳ３からの実行指令に従って演算処理を実行することに特化した機能ブロックである。

【００２３】

次に、図２を参照してＦＰＵＯＳ３の構成例を説明する。ＦＰＵＯＳ３は、パワーオンリセット２１（ＰＯＲ）、ＯＳステートマシン２２、メッセージトレイ２３、メッセージキュー２４、及びタスクトレイ２５を備え、またメッセージリスト４との間のインタフェース２６～２９を備える。

【００２４】

メッセージトレイ２３には、タスクＩＤと優先度とのパラメータをタスクメッセージＭＳＧとしてｎ対格納可能とされていると共に、トレイ格納メッセージ数のパラメータの格納領域が確保されている。タスクＩＤは、入力した時間順に割り振られるタスクメッセージＭＳＧの識別番号を示す。タスクＩＤは、メッセージリスト４に格納される開始アドレス及び終了アドレスの情報量より少ない情報量により表現可能な識別番号である。タスクメッセージＭＳＧにはそれぞれ優先度が設定される。

【００２５】

トレイ格納メッセージ数は、メッセージトレイ２３に格納したタスクメッセージＭＳＧの総数を示す。このトレイ格納メッセージ数に従ってタスクメッセージＭＳＧはメッセージキュー２４に転送される。

転送方法は、メッセージトレイ２３にタスクＩＤと優先度のパラメータが格納される度に、ＦＰＵＯＳ３はトレイ格納メッセージ数をアップカウントする時分割転送を行っても良いし、また並列に１サイクルで転送しても良い。また、タスクメッセージＭＳＧが、メッセージトレイ２３からメッセージキュー２４に転送されると、ＦＰＵＯＳ３はトレイ格納メッセージ数をダウンカウントする。

【００２６】

メッセージキュー２４は、優先度とタスクＩＤと状態とのパラメータの組合せのタスクメッセージＭＳＧをｎ対格納可能に構成される。タスクトレイ２５には、タスク中断フラグ、タスクトレイ状態TTRAY__ST、通常タスクトレイNTASK__TRAYの通常開始アドレスNTT__STA__ADDR及び通常終了アドレスNTT__STP__ADDR、及び、優先タスクトレイPTASK__TRAYの優先開始アドレスPTT__STA__ADDR及び優先終了アドレスPTT__STP__ADDR、のパラメータの格納領域が確保されている。タスクトレイ２５は、ＦＰＵＯＳ３とＷＯＲＫＥＲ６との間のインタフェースとして設けられるもので、タスクトレイ２５に確保される各パラメータは、ＦＰＵＯＳ３の側、及び、ＷＯＲＫＥＲ６の側から共に認識可能（読取可能）にされている。

【００２７】

タスク中断フラグは、ＦＰＵＯＳ３からセットされることでタスクを中断することを表すフラグであり、ＦＰＵＯＳ３及びＷＯＲＫＥＲ６の双方から認識可能なフラグである。タスク中断フラグは、演算タスクを中断するための中断ハンドシェーク期間（図６のｔ１７～ｔ１８）の間に用いられるフラグであり、タスク中断フラグをセット及びクリアする権限は、ＷＯＲＫＥＲ６には与えられておらず、ＦＰＵＯＳ３の側に与えられている。

【００２８】

またタスクトレイ状態TTRAY__STは、ＷＯＲＫＥＲ６がタスクを実行していない状態を表す非実行状態、又は、ＷＯＲＫＥＲ６がタスクを実行中である状態を表す実行状態、の何れかを表す。

【００２９】

具体的に、タスクトレイ状態TTRAY__STは、図３に例示したように、「タスク空状態ＳＴ１」、「通常タスク実行状態ＳＴ２」、「通常タスク完了状態ＳＴ３」、「通常タスク中断状態ＳＴ４」、「優先タスク実行状態ＳＴ５」、のうち何れかの状態に設定される。

【００３０】

この中で、タスク空状態ＳＴ１、通常タスク完了状態ＳＴ３、通常タスク中断状態ＳＴ４では、ＷＯＲＫＥＲ６が何れのタスクも実行していないタスク非実行状態になっており

、通常タスク実行状態ＳＴ２、通常タスク中断状態ＳＴ４では、ＷＯＲＫＥＲ６が何らかのタスクを実行しているタスク実行状態となっている。

【００３１】

またタスクトレイ状態TTRAY__STの書込みアクセス権は、ＦＰＵＯＳ３及びＷＯＲＫＥＲ６の双方に与えられており、タスクトレイ状態TTRAY__STはＦＰＵＯＳ３及びＷＯＲＫＥＲ６の双方から書込可能・書換可能に設定されている。

【００３２】

ただし、タスクトレイ状態TTRAY__STの書込みアクセス権は、ＦＰＵＯＳ３及びＷＯＲＫＥＲ６の間で競合を避けるように設定されている。タスクトレイ状態TTRAY__STの書込みアクセス権は、タスク非実行状態ではＦＰＵＯＳ３の側にあり、タスク実行状態ではＷＯＲＫＥＲ６の側にある。

【００３３】

ＦＰＵＯＳ３は、イベントドリブン型で処理を実行するオペレーティングシステム演算に係るハードウェアブロックであり、演算イベントを管理する外部の上位ブロックとなるスケジューラ１０から演算イベントの発生をインターフェース１２（図１参照）を経由して受付ける。

【００３４】

ＦＰＵＯＳ３は、図２に示すメッセージトレイ２３にタスクＩＤ、優先度、トレイ格納メッセージ数が書込まれることで演算イベントに係るタスクメッセージＭＳＧを受領する。

【００３５】

ＯＳステートマシン２２は、電源投入後にパワーオンリセット２１が解除された後、基本動作として下記の４つの状態をサイクリックに遷移するようにハードウェア構成されている。４つの状態は、Send message（以降、Sendと略す）状態、Peek message（以降、Peekと略す）状態、Dispatch message（以降、Dispと略す）状態、Post processing（以降、Postと略す）状態である。

【００３６】

以下、ＯＳステートマシン２２によるサイクリックの状態遷移に伴い、ＷＯＲＫＥＲ６が実行する通常タスク処理の基本的流れを説明する。

＜Send状態＞

ＯＳステートマシン２２は、Send状態に遷移すると、外部からメッセージトレイ２３に受領したタスクメッセージＭＳＧをメッセージキュー２４に転送する。このとき、メッセージキュー２４が空状態であれば、タスクメッセージＭＳＧは、メッセージキュー２４の先頭から順に格納される。また、他のタスクメッセージＭＳＧが、メッセージキュー２４に既に存在する場合には、既に格納されているタスクＩＤの次の番号にタスクメッセージＭＳＧが時系列的に順に書き込まれる。メッセージキュー２４は、このような先入れ先出し型のキュー構造に構成されている。

【００３７】

＜Peek状態＞

ＯＳステートマシン２２は、メッセージトレイ２３からメッセージキュー２４への転送を終えると、Peek状態に遷移する。ＯＳステートマシン２２は、Peek状態において、起動有無を判定すると共に、メッセージキュー２４の中から起動するべきタスクメッセージＭＳＧを選択する。

【００３８】

ＯＳステートマシン２２は、Peek状態において、メッセージキュー２４の中にタスクメッセージＭＳＧが無いと判定すると、スルーしてタスクメッセージＭＳＧの起動無しとする。

【００３９】

逆に、タスクメッセージＭＳＧがメッセージキュー２４の中に格納されている場合、ＯＳステートマシン２２は、Peek状態において、メッセージキュー２４の優先度のパラメータと、状態のパラメータと、メッセージキュー２４のキュー番号と、に基づいて起動する

べきタスクメッセージＭＳＧを選択する。同一優先度であってもキュー番号が若い番号であるほど、タスクメッセージＭＳＧは優先的に選択される。

【００４０】

優先度のパラメータは、低優先度又は高優先度を含む複数段階の優先度を示すパラメータを示す。本実施形態において、優先度は低優先度又は高優先度の２段階に設定される例を説明するが、３段階以上であっても良い。状態のパラメータは、メッセージキュー２４の状態を示し、準備状態（以下Ready状態）、初動状態又は起動状態（以下Peek状態）、実行状態（以下Run状態）、待機状態（以下Wait状態）、終了状態（以下Nothing状態）の何れかに設定される。Run状態以外の状態では、ＷＯＲＫＥＲ６の側でタスクメッセージＭＳＧを実行していない非実行状態を示している。

【００４１】

通常、ＯＳステートマシン２２は、Ready状態となっているメッセージキュー２４のタスクメッセージＭＳＧを選択し、選択したメッセージキュー２４の状態をReady状態からPeek状態に遷移させる。ＯＳステートマシン２２は、タスクＩＤを選択することで起動するべきタスクメッセージＭＳＧを選択する。

【００４２】

＜Disp状態＞

ＯＳステートマシン２２は、起動するべきタスクメッセージＭＳＧの選択を終えると、Disp状態に遷移し、Peek状態で選定されたタスクメッセージＭＳＧをタスクトレイ２５に転送し、ＷＯＲＫＥＲ６にタスクメッセージＭＳＧを実行指令する。

【００４３】

ＦＰＵＯＳ３は、Disp状態において、メッセージキュー２４の中の状態のパラメータがPeek状態に該当しているタスクＩＤをメッセージリスト４の中で検索し、照合されたタスクＩＤに紐づけられた開始アドレス及び終了アドレスを決定する。

【００４４】

そしてＯＳステートマシン２２は、メッセージリスト４の中で決定された開始アドレス／終了アドレスを、インタフェース２７を経由してタスクトレイ２５の中の通常タスクトレイNTASK__TRAYの通常開始アドレスNTT__STA__ADDR／通常終了アドレスNTT__STP__ADDRにそれぞれ格納する。

【００４５】

そしてＯＳステートマシン２２は、メッセージキュー２４の状態をPeek状態からRun状態に遷移させ、タスクトレイ状態TTRAY__STを「通常タスク実行状態ＳＴ２」に遷移させる。ＷＯＲＫＥＲ６は、タスクトレイ状態TTRAY__STをポーリングすることでタスク実行指令Ｓ５を受領する。

【００４６】

＜Post状態＞

ＦＰＵＯＳ３が、ＷＯＲＫＥＲ６へのタスク実行指令Ｓ５を終えると、ＯＳステートマシン２２は、Post状態に遷移する。ＦＰＵＯＳ３は、Post状態において、後処理としてメッセージキュー２４を整理整頓する。具体的に、ＦＰＵＯＳ３は、メッセージキュー２４にNothing状態となっているタスクメッセージＭＳＧがある場合にはメッセージキュー２４を空にする。

【００４７】

他方、ＷＯＲＫＥＲ６が、演算タスクを実行中にはメッセージキュー２４に格納されるタスクメッセージＭＳＧがNothing状態になることはない。ＷＯＲＫＥＲ６が、演算タスクを実行中の場合には、ＯＳステートマシン２２は、Post状態を通過しSend状態に戻ってサイクリックに状態遷移する。

【００４８】

ＷＯＲＫＥＲ６が、演算タスクを実行終了すると、ＷＯＲＫＥＲ６はタスクトレイ状態TTRAY__STを「通常タスク完了状態ＳＴ３」に遷移させる。ＯＳステートマシン２２は、Disp状態に遷移し、実行中のメッセージキュー２４の状態をRun状態からNothing状態

に遷移させる。ＯＳステートマシン２２は、Post状態に遷移する。ＯＳステートマシン２２は、Post状態にてNothing状態のメッセージキュー２４がある場合、当該メッセージキュー２４を空にする。

【００４９】

＜タスクＩＤを用いる技術的意義の説明＞

通常、メッセージリスト４に格納される開始アドレス／終了アドレスは、２＾ｎビット、例えば１６ビット、３２ビットなど多ビット表現される。アドレス番地が比較的大きな値になると、このアドレス番地を他の情報と紐付けて記憶保持するためのレジスタなどのハードウェア記憶部を必要以上に多く必要とし、特にメッセージトレイ２３及びメッセージキュー２４のハードウェア回路の規模が大きくなりやすい。

【００５０】

しかし本実施形態では、ＯＳステートマシン２２が、取り扱うデータを開始アドレス／終了アドレスそのものにするのではなく、タスクＩＤを用いている。タスクＩＤは、１ビット又は２ビット程度で表現可能であり、少なくとも開始アドレス／終了アドレスのアドレス番地よりも少ない情報量で表現可能となっている。このため、メッセージトレイ２３及びメッセージキュー２４のハードウェア記憶部に必要な記憶容量を少なくでき、メッセージトレイ２３及びメッセージキュー２４の回路規模を抑制できる。

【００５１】

＜バンク切替インタフェース２８を用いる技術的意義の説明＞

前述の構成では、タスクＩＤは１ビット又は２ビット程度の例を挙げたが、仮にタスクＩＤの種類が多くなると、タスクＩＤの情報量が多くなり、この場合、メッセージトレイ２３やメッセージキュー２４のハードウェア回路規模の増加に繋がる虞がある。

【００５２】

このような場合、図２に示すように、スケジューラ１０とメッセージリスト４との間を仲介し外部からバンク切替えするためのバンク切替インタフェース２８を設けることが望ましく、またメッセージリスト４には、当該メッセージリスト４の中の情報をバンク切替えするバンク情報保持領域を備えることが望ましい。これにより、外部のスケジューラ１０が、メッセージリスト４に紐づけられたタスクＩＤと開始アドレス／終了アドレスとの関係性をバンク切替えできるようになる。

【００５３】

メッセージリスト４のバンク情報保持領域にはバンク切替情報が記憶される。バンク切替情報は例えば１ビット程度の少数の情報量によるもので、少数の情報によりタスクＩＤと開始アドレス／終了アドレスとの関係性を大きく変更できる。このため、外部からバンク切替えするためのバンク切替インタフェース２８を設けることで、記憶すべきタスクＩＤの情報量を削減でき、この結果、メッセージトレイ２３及びメッセージキュー２４を構成するハードウェア回路を小規模化できる。

【００５４】

＜ＷＯＲＫＥＲ６の詳細説明＞

図１に例示したように、ＷＯＲＫＥＲ６は、プログラムカウンタブロック３１、演算命令デコーダ３２、及びセレクタ３３を備える。ＦＰＵＯＳ３とＷＯＲＫＥＲ６とは、ハードウェア上、独立してＡＳＩＣ内に構成されている。

【００５５】

ＷＯＲＫＥＲ６は、ＦＰＵＯＳ３から演算イベントに係るタスクメッセージＭＳＧに紐付いたメッセージリスト４、演算命令テーブル５の開始アドレスと終了アドレス（通常開始アドレスNTT__STA__ADDR／通常終了アドレスNTT__STP__ADDR、又は、優先開始アドレスPTT__STA__ADDR／優先終了アドレスPTT__STP__ADDR）を受付けると、プログラムカウンタブロック３１を直接的に制御する。ＦＰＵＯＳ３とプログラムカウンタブロック３１とのインタフェース構造は、その詳細を後述する。

【００５６】

ＷＯＲＫＥＲ６は、ＦＰＵＯＳ３から演算タスクに係る実行指令を受付けると、演算タ

スクの実行指令に紐付いて予め設定されたメッセージリスト４の開始アドレスから終了アドレスに至るまで、演算命令デコーダ３２により演算命令テーブル５の中の演算命令セットを参照する。

【００５７】

ＷＯＲＫＥＲ６は、演算命令デコーダ３２にて参照した演算命令セットを、ＦＰＵ演算命令Ｓ１７、引数選択Ｓ２７～Ｓ２８に分解し、ＦＰＵ第１引数汎用レジスタ群７、ＦＰＵ第２引数汎用レジスタ群８、及び、ＦＰＵ９に出力する。

【００５８】

ＦＰＵ第１引数汎用レジスタ群７は、複数の引数を記憶可能なレジスタの集合と、レジスタの集合の後段に接続されるレジスタ選択セレクタと（何れも図示せず）により構成されている。レジスタ選択セレクタは、セレクタ３３の出力Ｓ１５も入力するように接続されている。

【００５９】

ＦＰＵ第２引数汎用レジスタ群８もまた、複数の引数を記憶可能なレジスタの集合と、レジスタの集合の後段に接続されるレジスタ選択セレクタと（何れも図示せず）により構成されている。レジスタ選択セレクタは、セレクタ３３の出力Ｓ１６も入力するように接続されている。

【００６０】

前記したセレクタ３３は、外部メモリ１１から外部データＳ１３を入力可能に接続されている。演算命令デコーダ３２は、外部アドレス信号Ｓ１２により外部メモリ１１のアドレスを指定することで、入力する外部データＳ１３を指定できる。またセレクタ３３は、ＦＰＵ９による浮動小数点演算結果、ＦＰＵ第１引数汎用レジスタ群７の記憶値、ＦＰＵ第２引数汎用レジスタ群８の記憶値、を入力可能に接続されている。

【００６１】

演算命令デコーダ３２は、入力される演算命令テーブル５の演算命令セットに基づいて、セレクタ３３の入力元及び出力先を選択する。セレクタ３３は、演算命令デコーダ３２から入力した演算命令デコーダ選択Ｓ２９に基づいて、前述の入力値を、ＦＰＵ第１引数汎用レジスタ群７にＦＰＵ第１引数汎用レジスタ入力選択結果として出力Ｓ１５を振り分けたり、ＦＰＵ第２引数汎用レジスタ群８にＦＰＵ第２引数汎用レジスタ入力選択結果として出力Ｓ１６を振り分ける。

【００６２】

これにより、ＦＰＵ第１引数汎用レジスタ群７の何れかのレジスタや、ＦＰＵ第２引数汎用レジスタ群８の何れかのレジスタには、外部メモリ１１の外部データＳ１３、又は、ＦＰＵ９による浮動小数点演算結果を保持させることができる。

【００６３】

演算命令デコーダ３２は、引数選択Ｓ２７に基づいてＦＰＵ第１引数汎用レジスタ群７の何れかのレジスタに記憶された記憶値をＦＰＵ９に出力したり、セレクタ３３の出力Ｓ１５を直接ＦＰＵ９に出力できる。

【００６４】

演算命令デコーダ３２は、引数選択Ｓ２８に基づいてＦＰＵ第２引数汎用レジスタ群８の何れかのレジスタに記憶された記憶値をＦＰＵ９に出力したり、セレクタ３３の出力Ｓ１６を直接ＦＰＵ９に出力できる。

【００６５】

ＦＰＵ９は、ＦＰＵ第１引数汎用レジスタ群７、ＦＰＵ第２引数汎用レジスタ群８の出力引数を用いて、ＦＰＵ演算命令Ｓ１７により指定される四則演算や比較、型変換等の演算を行うことができ、これにより浮動小数点の演算処理を実行できる。ＦＰＵ９は、浮動小数点の演算結果をＦＰＵ演算結果Ｓ１４として出力する。ＦＰＵ９は、多種多様な入力データを引数として浮動小数点の演算タスクを実行できる。

【００６６】

ＷＯＲＫＥＲ６は、演算毎にプログラムカウンタブロック３１のプログラムカウンタ４

１（図４参照）を更新しつつ、２項演算処理を順次実行し、終了アドレスまで到達すると、ＷＯＲＫＥＲ６は、ＦＰＵＯＳ３にタスク完了通知を出力することで、一連の浮動小数点演算タスクを完了させることができる。

【００６７】

＜＜プログラムカウンタブロック３１の構成及び基本動作＞＞

以下、プログラムカウンタブロック３１の構成及び基本動作について、図４を参照しながら説明する。プログラムカウンタブロック３１は、プログラムカウンタ４１、一致比較部４２、遷移検出部４３、中断判定部４４、開始アドレスセレクタ４５、及び、終了アドレスセレクタ４６を備える。

【００６８】

開始アドレスセレクタ４５は、タスクトレイ状態TTRAY__STに記憶された状態に基づいて、通常タスクトレイNTASK__TRAYの通常開始アドレスNTT__STA__ADDR、又は、優先タスクトレイPTASK__TRAYの優先開始アドレスPTT__STA__ADDRの何れかを選択し、プログラムカウンタ４１に何れかの開始アドレスを出力する。

【００６９】

終了アドレスセレクタ４６は、タスクトレイ状態TTRAY__STに記憶された状態に基づいて、通常タスクトレイNTASK__TRAYの通常終了アドレスNTT__STP__ADDR又は優先タスクトレイPTASK__TRAYの優先終了アドレスPTT__STP__ADDRの何れかを選択し、一致比較部４２に何れかの終了アドレスを出力する。

【００７０】

遷移検出部４３は、タスクトレイ状態TTRAY__STが「通常タスク空状態ＳＴ１」から「通常タスク実行状態ＳＴ２」へ遷移したこと、「通常タスク中断状態ＳＴ４」から「優先タスク実行状態ＳＴ５」へ遷移したことを検出し、これらのタイミングでロード指令Ｓ３３をプログラムカウンタ４１に出力する。

【００７１】

プログラムカウンタ４１は、ロード指令Ｓ３３を入力することで開始アドレスセレクタ４５を通じて開始アドレスをロードし、当該開始アドレスから順にカウント開始し、一致比較部４２にカウントしたアドレスカウンタ値を出力すると共に、演算命令テーブル５にカウントしたアドレスカウンタ値を演算命令テーブルアドレスＳ１０として出力する。これにより、演算命令デコーダ３２は、演算命令テーブルアドレスＳ１０に対応した演算命令セット選択結果Ｓ１１を参照し、当該演算命令をデコードできる。

【００７２】

他方、一致比較部４２は、プログラムカウンタ４１のアドレスカウンタ値が終了アドレスセレクタ４６により選択された終了アドレスと一致しているか否かを判定し、終了アドレス一致信号Ｓ３２を出力したときにタスクトレイ状態TTRAY__STの状態を遷移させる。

【００７３】

このとき、図３に例示したように、タスクトレイ状態TTRAY__STが「通常タスク実行状態ＳＴ２」であれば一致比較部４２の指令によりタスクトレイ状態TTRAY__STは「通常タスク完了状態ＳＴ３」に遷移する。またタスクトレイ状態TTRAY__STが「優先タスク実行状態ＳＴ５」であれば一致比較部４２の指令によりタスクトレイ状態TTRAY__STは「通常タスク中断状態ＳＴ４」に遷移する。

【００７４】

また中断判定部４４は、タスク中断フラグのセット／リセット状態に基づいて、現在実行中のタスクを継続するか中断するかを判定する。中断判定部４４は、タスク中断フラグがセットされたタイミング以降、演算命令テーブル５から出力される演算命令セット選択結果Ｓ１１を確認する。

【００７５】

また中断判定部４４は、確認した演算命令セットに基づいて中断アドレスを決定し、プログラムカウンタ４１のアドレスカウンタ値が中断アドレスに到達した後に、通常開始アドレスNTT__STA__ADDRを格納する格納領域に中断アドレスを退避させる。中断アドレ

スを退避させる格納領域は、通常開始アドレスNTT__STA__ADDRの格納領域に限られるものではなく、例えば、プログラムカウンタブロック３１の内部に別途レジスタを設けて中断アドレスを退避させるようにしても良い。

【００７６】

そして中断判定部４４は、ＷＯＲＫＥＲ中断完了Ｓ３１を出力しタスクトレイ２５のタスクトレイ状態TTRAY__STを「通常タスク実行状態ＳＴ２」から「通常タスク中断状態ＳＴ４」に遷移させる。

【００７７】

遷移検出部４３は、タスクトレイ状態TTRAY__STが「通常タスク中断状態ＳＴ４」から「通常タスク実行状態ＳＴ２」へ遷移する時に再開ロード信号Ｓ３０をトリガ出力する。遷移検出部４３は、この再開ロード信号Ｓ３０のトリガ出力にて中断時に通常開始アドレスNTT__STA__ADDRを格納する格納領域に退避した中断アドレスをプログラムカウンタ４１に再ロードさせる。

【００７８】

すると、演算命令デコーダ３２は、プログラムカウンタ４１のカウント値の示す通常タスクに係る演算命令コードをデコード再開し、ＦＰＵ９が通常タスクに係る演算処理を実行再開する。その後の流れは、前述と同様である。

【００７９】

次に、ＦＰＵＯＳ３によるオーバーヘッド削減効果について比較例と共に説明する。

＜比較例の説明＞

従来より用いられるマイコンでは、アプリケーション未処理中に割込みが発生すると、割込みベクタにジャンプし、割込みハンドラ処理にジャンプし、割込みハンドラ処理の先頭で、汎用レジスタやステータスレジスタ、ジャンプ元のプログラムカウンタ等、中断元に戻るための情報を退避する割込み時にコンテキストスイッチが行われる。

【００８０】

その後、割込みハンドラが、メイン処理として各種割込みに応じたタスク選定を行い、ＯＳに向けて選定したタスクをメッセージ送信し、割込みハンドラの後処理として復帰時コンテキストスイッチを行うことで中断元に復帰する。その後、ＯＳ内にてメッセージ受信し、アプリケーションタスクを起動する。

【００８１】

システムがアプリケーションタスクを実行するときに、その先頭で起動元に戻るための情報を退避するアプリ起動時コンテキストスイッチを行い、アプリケーションタスクのメイン処理を実行する。

【００８２】

高優先度タスク実行中に割込みが発生すると、前述同様に割込みハンドラ処理にジャンプすると共に、割込み時コンテキストスイッチが行われ、割込みハンドラのメイン処理として、各種割込みに応じたタスク選定を行い、ＯＳに向けて選定したタスクをメッセージ送信し、割込みハンドラの後処理として復帰時コンテキストスイッチを行って中断元に復帰する。

【００８３】

一般的なマイコンでは、割込みが発生すると、割込み時及び復帰時におけるコンテキストスイッチのオーバーヘッド時間と、割込みハンドラ処理時間分、アプリケーションタスク処理が中断される構造となっている。このため、割込発生時にアプリケーションのタスク処理が中断されるため、タスク処理のスピードが劣ってしまう。

【００８４】

＜本実施形態の動作説明＞

これに対し、本実施形態では、ＦＰＵＯＳ３とＷＯＲＫＥＲ６とがハードウェア的に独立して構成されている。このため、実行タスクを中断させることなくアプリケーションタスクを処理でき、割込み時及び復帰時におけるコンテキストスイッチのオーバーヘッド時間分だけ短縮できる。

【００８５】
以下、この処理動作に係る詳細説明を行う。図５は、ＷＯＲＫＥＲ６が最優先となる高優先度（第１優先度相当）の演算タスクを実行している最中に低優先度（第２優先度相当）の演算イベントが発生した場合のケースを示している。
【００８６】
まずＷＯＲＫＥＲ６が演算タスクを実行待機している最中に、最優先の高優先度の演算イベントが発生すると、ＦＰＵＯＳ３は、ＯＳステートマシン２２によりメッセージトレイ２３にタスクＩＤと優先度とを紐づけてタスクメッセージＭＳＧ１を格納する。ここでは、高優先度のタスクメッセージＭＳＧをタスクメッセージＭＳＧ１と定義している。
【００８７】
図５に示すタスクメッセージＭＳＧ１の格納タイミングｔ１は、ＯＳステートマシン２２のSend message状態（以降、Send状態と略す）を過ぎていることを想定している。この場合、ＯＳステートマシン２２は、Peek message状態（以降、Peek状態と略す）、Dispatch message状態（以降、Disp状態と略す）、Post processing状態（以降、Post状態と略す）を遷移後の次のSend状態のタイミングｔ２にて、メッセージトレイ２３の第１トレイＮｏ１からメッセージキューＭＱ１にタスクメッセージＭＳＧ１を転送すると共に、メッセージトレイ２３の第１トレイＮｏ１を空状態にクリーニングする。
【００８８】
ＯＳステートマシン２２は、タイミングｔ２においてメッセージキューＭＱ１の状態をReadyから次ステートのPeek状態に遷移させ、起動タスクを決定する。ＯＳステートマシン２２は、Peek状態においてメッセージキュー２４の中から起動するべきタスクメッセージＭＳＧ１を選択する。ここでは、タスクメッセージＭＳＧ１を選択したものとして説明する。ＯＳステートマシン２２は、タイミングｔ３においてDisp状態に遷移し、Peek状態で選択された起動待機中のタスクメッセージＭＳＧ１を起動する。
【００８９】
ＯＳステートマシン２２は、タイミングｔ４において、メッセージキューＭＱ１に係るタスクメッセージＭＳＧ１の状態をPeek状態からRun状態に遷移させると共に、タスクメッセージＭＳＧ１に紐づいたタスクＩＤの演算命令テーブル５の開始アドレス及び終了アドレスをメッセージリスト４の中で検索する。
【００９０】
そしてＯＳステートマシン２２は、開始アドレス及び終了アドレスをインタフェース２７を経由して読み出し、ＦＰＵＯＳ３のタスクトレイ２５の中の通常タスクトレイNTASK__TRAYの通常開始アドレスNTT__STA__ADDR及び通常終了アドレスNTT__STP__ADDRに格納し、タスクトレイ状態TTRAY__STを「通常タスク空状態ＳＴ１」から「通常タスク実行状態ＳＴ２」に遷移させる。
【００９１】
ＷＯＲＫＥＲ６の側では、タスクトレイ状態TTRAY__STが「通常タスク空状態ＳＴ１」から「通常タスク実行状態ＳＴ２」へ遷移することをもって、通常開始アドレスNTT__STA__ADDRから通常終了アドレスNTT__STP__ADDRにかけた演算内容を演算命令テーブル５から読出しながら、ＦＰＵ９、ＦＰＵ第１引数汎用レジスタ群７、及び、ＦＰＵ第２引数汎用レジスタ群８を用いて浮動小数点の２項演算を順次処理する。
【００９２】
ＷＯＲＫＥＲ６が高優先度の演算タスクを実行中に、タイミングｔ５にて低優先度の演算イベントが発生すると、ＦＰＵＯＳ３の中のメッセージトレイ２３の第１トレイＮｏ１に低優先度のタスクメッセージＭＳＧ２を格納させる。ここでは、低優先度のタスクメッセージＭＳＧをタスクメッセージＭＳＧ２としている。
【００９３】
その後、ＯＳステートマシン２２がSend状態にされたとしても、メッセージキューＭＱ１が空状態ではない。このため、ＯＳステートマシン２２は、タイミングｔ６においてタスクメッセージＭＳＧ２をメッセージキューＭＱ２に転送すると共に、メッセージトレイ

２３の第１トレイＮｏ１を空状態にクリーニングする。続いて、ＯＳステートマシン２２がPeek状態になると、起動タスクを選択しようとするが、すでに高優先度演算タスクが実行されている状態であるため、Peek状態では何もせずDisp状態に移行する。

【００９４】

このときＦＰＵＯＳ３の側では、タスクトレイ状態TTRAY__STが通常タスク実行状態ＳＴ２に維持されていることから、ＷＯＲＫＥＲ６の側で実行しているタスクメッセージＭＳＧが、タスクメッセージＭＳＧ１であることを把握できる。

【００９５】

またさらに、ＦＰＵＯＳ３が、ＷＯＲＫＥＲ６とは独立したハードウェアにより構成されている。このため、ＷＯＲＫＥＲ６が高優先度の演算タスクを実行している最中に、低優先度の演算イベントが発生したとしても、ＷＯＲＫＥＲ６は、タイミングｔ５以降において高優先度の演算タスクの実行を中断することなく、高優先度の演算タスクを集中的に処理できる。

【００９６】

ＷＯＲＫＥＲ６の側で特に最優先となる高優先度の演算タスクを実行しているときには、ＦＰＵＯＳ３が低優先度の演算タスクをＷＯＲＫＥＲ６に対して実行指令に移さないようにしているため、ＷＯＲＫＥＲ６は当該高優先度の演算タスクを中断することなく、当該演算タスクを最速処理できる。

【００９７】

続いて、他の実行例を挙げて、ＦＰＵＯＳ３による処理オーバーヘッド時間抑制効果を説明する。以下では、図６に例示したように、最優先よりも低い低優先度（第３優先度相当）の演算タスクを実行中に、この演算タスクよりも高優先度（第４優先度相当）の演算イベントが発生した時の挙動を例に挙げて動作概要を説明する。

【００９８】

ＷＯＲＫＥＲ６が、未処理状態にて待機している最中に、スケジューラ１０から低優先度の演算イベントが発行されると、低優先度の演算イベントにて実行したいタスクのタスクＩＤと優先度がメッセージトレイ２３に格納される。

【００９９】

図６に示すように、低優先度の演算イベントに係るタスクメッセージＭＳＧ１のメッセージトレイ２３への格納タイミングｔ１１は、ＯＳステートマシン２２のSend状態を過ぎている。このため、ＯＳステートマシン２２は、Peek状態、Disp状態、Post状態を遷移した後の次のSend状態の終了タイミングにて、メッセージトレイ２３からメッセージキューＭＱ１にタスクメッセージＭＳＧ１を転送すると共に、メッセージトレイ２３の第１トレイＮｏ１を空状態にクリーニングする。

【０１００】

ＯＳステートマシン２２は、タイミングｔ１２においてメッセージキューＭＱ１の状態をReadyから次ステートのPeek状態に遷移させ、起動タスクを決定する。ＯＳステートマシン２２は、Peek状態においてメッセージキュー２４の中から起動タスクを選択する。ＯＳステートマシン２２は、タイミングｔ１３においてDisp状態に遷移し、Peek状態で選択された起動待機中のタスクを起動する。

【０１０１】

ＯＳステートマシン２２は、タイミングｔ１４においてメッセージキューＭＱ１のタスクメッセージＭＳＧ１の状態をPeek状態からRun状態に遷移させると共に、メッセージキューＭＱ１のタスクＩＤに紐づいた演算命令テーブル５の開始アドレス及び終了アドレスを検索する。

【０１０２】

そしてＯＳステートマシン２２は、開始アドレス及び終了アドレスをインターフェース２７を経由して読み出し、ＦＰＵＯＳ３のタスクトレイ２５の通常タスクトレイNTASK__TRAYの通常開始アドレスNTT__STA__ADDR及び通常終了アドレスNTT__STP__ADDRに格納し、タスクトレイ状態TTRAY__STを「通常タスク空状態ＳＴ１」から「通常タスク

実行状態ＳＴ２」に遷移させる。
【０１０３】
ＷＯＲＫＥＲ６の側では、タスクトレイ状態TTRAY__STが「通常タスク空状態ＳＴ１」から「通常タスク実行状態ＳＴ２」へ遷移することをもって、通常開始アドレスNTT__STA__ADDRから通常終了アドレスNTT__STP__ADDRにかけた演算内容を演算命令テーブル５から読出しながら、ＦＰＵ９、ＦＰＵ第１引数汎用レジスタ群７、ＦＰＵ第２引数汎用レジスタ群８を用いて浮動小数点の２項演算を順次処理する。
【０１０４】
＜プログラムカウンタブロック３１の動作概要＞
ＯＳステートマシン２２がDisp状態とされている時、図４に示すプログラムカウンタブロック３１は、遷移検出部４３によるロード指令Ｓ３３の出力のタイミングにおいて通常開始アドレスNTT__STA__ADDRをプログラムカウンタ４１にロードする。
【０１０５】
このロードしたタイミングにおいて、タスクトレイ状態TTRAY__STは、「通常タスク実行状態ＳＴ２」とされている。このため、開始アドレスセレクタ４５は、通常開始アドレスNTT__STA__ADDRの格納領域を選択し、プログラムカウンタ４１に通常開始アドレスNTT__STA__ADDRをロードする。他方、終了アドレスセレクタ４６は、通常タスクトレイNTASK__TRAYに格納される通常終了アドレスNTT__STP__ADDRを選択し、一致比較部４２に通常終了アドレスNTT__STP__ADDRを入力させる。
【０１０６】
演算命令デコーダ３２は、演算命令テーブル５の中でプログラムカウンタ４１の指し示すアドレスを指定し演算命令コードをデコードする。プログラムカウンタ４１は、クロック入力に伴いカウントを進行させることで演算命令テーブル５を参照しながら演算命令コードを順次デコードし、ＦＰＵ９は、通常終了アドレスNTT__STP__ADDRまで演算命令を実行する。
【０１０７】
＜低優先度の演算イベントを実行中に高優先度の演算イベントを発行＞
他方、ＷＯＲＫＥＲ６が、タイミングｔ１４以降にて低優先度の演算タスクを実行中に、タイミングｔ１５にて高優先度の演算イベントを発生すると、ＦＰＵＯＳ３は、当該高優先度の演算イベントに係るタスクを低優先度の演算イベントに係るタスクよりも優先してＷＯＲＫＥＲ６に実行指令する。
【０１０８】
このとき、外部のスケジューラ１０がメッセージトレイ２３の第１トレイＮｏ１に高優先度のタスクメッセージＭＳＧ２を格納させる。その後、ＯＳステートマシン２２はSend状態になると、メッセージキューＭＱ１が空状態ではないため、タイミングｔ１６においてタスクメッセージＭＳＧ２をメッセージキューＭＱ２に転送すると共に、メッセージトレイ２３を空状態にクリーニングする。
【０１０９】
続いて、ＯＳステートマシン２２がPeek状態になると、起動するタスクメッセージＭＳＧを選択しようとするが、ＷＯＲＫＥＲ６が低優先度のタスクメッセージＭＳＧ１を実行中の状態であるため、メッセージキューＭＱ２はReady状態をキープし、ＦＰＵＯＳ３は、ＯＳステートマシン２２の状態がPeek状態のときにタスク中断フラグをセットする。
【０１１０】
ＦＰＵＯＳ３は、タスク中断フラグをセットすることで、ＷＯＲＫＥＲ６に実行中の低優先度のタスクメッセージＭＳＧ１の中断指令を行い、低優先度のタスクメッセージＭＳＧ１の中断ハンドシェークを行う。サイクリックに遷移していたＯＳステートマシン２２は、中断ハンドシェーク期間ｔ１７～ｔ１８においてDisp状態に保持される。
【０１１１】
＜ＷＯＲＫＥＲ６による演算実行＞
ＦＰＵＯＳ３がタスク中断フラグをセットするタイミングｔ１７以降、ＷＯＲＫＥＲ６

がタスク中断フラグを認識する。ＷＯＲＫＥＲ６は、演算命令テーブル５を参照して演算命令セット選択結果Ｓ１１を確認する。そして、ＷＯＲＫＥＲ６は、タイミングｔ１８において演算命令を一時中断可能な区切りのよいタイミングで中断アドレスを判断し、処理中断を完了させると共に、判断した中断アドレスを退避させる。このとき、ＷＯＲＫＥＲ６は、中断アドレスをタスクトレイ２５の通常開始アドレスNTT__STA__ADDRの格納領域に退避させる。

【０１１２】

また中断判定部４４は、ＷＯＲＫＥＲ中断完了Ｓ３１を経由してタスクトレイ状態TTRAY__STを「通常タスク実行状態ＳＴ２」から「通常タスク中断状態ＳＴ４」に遷移させる。

【０１１３】

ＦＰＵＯＳ３の側では、当該ＦＰＵＯＳ３とは独立して動作するＷＯＲＫＥＲ６の演算処理中断の応答を待機している。ＦＰＵＯＳ３は、タスクトレイ状態TTRAY__STが「通常タスク実行状態ＳＴ２」から「通常タスク中断状態ＳＴ４」へ遷移したことをもって、ＷＯＲＫＥＲ６が演算実行を中断したと判断する。

【０１１４】

ＦＰＵＯＳ３は、タイミングｔ１８にてメッセージキューＭＱ１の状態をRun状態からWait状態に遷移させると共に、Disp状態からサイクリック遷移を再開し、Post状態に遷移する。

【０１１５】

ＦＰＵＯＳ３とＷＯＲＫＥＲ６との間の中断ハンドシェーク期間ｔ１７～ｔ１８の間、ＯＳステートマシン２２は、Disp状態に留まっている。ＯＳステートマシン２２は、サイクリックに状態を遷移させずDisp状態に留まているため、ＦＰＵＯＳ３は、ＷＯＲＫＥＲ６による演算タスクの実行に係る中断応答に対し即時対応できる。これにより、タスク中断オーバーヘッド時間の増加を抑制できる。

【０１１６】

その後、ＯＳステートマシン２２がサイクリック状態遷移を再開し、Peek状態まで遷移すると、ＦＰＵＯＳ３は、タイミングｔ１９においてタスク中断フラグをクリアする。

【０１１７】

またＯＳステートマシン２２は、高優先度のタスクメッセージＭＳＧ２が格納されているメッセージキューＭＱ２の状態をReady状態からPeek状態に遷移させ、タスク起動待ち状態にした後、Disp状態に遷移させる。

【０１１８】

通常、ＯＳステートマシン２２は、Disp状態においてタスクトレイ２５に、メッセージリスト４の開始アドレス及び終了アドレスを格納する。しかし、現状態では、タスクトレイ状態TTRAY__STが「通常タスク中断状態ＳＴ４」であるため、通常タスクトレイNTASK__TRAYに中断アドレスを残留したまま保持しておく。

【０１１９】

ＦＰＵＯＳ３は、メッセージキュー２４のPeek状態に紐付いたタスクＩＤの演算命令テーブル５の開始アドレス／終了アドレスを、メッセージリスト４の中で検索し、検索された開始アドレス／終了アドレスをインターフェース２７を経由して読み出す。

【０１２０】

ＦＰＵＯＳ３は、読み出した開始アドレス／終了アドレスを、タイミングｔ２０にて優先タスクトレイPTASK__TRAYの優先開始アドレスPTT__STA__ADDR及び優先終了アドレスPTT__STP__ADDRの格納領域に格納し、タスクトレイ状態TTRAY__STを「通常タスク中断状態ＳＴ４」から「優先タスク実行状態ＳＴ５」に遷移させる。ＯＳステートマシン２２は、Disp状態に留まるように保持させると共に、タスクトレイ状態TTRAY__STが「通常タスク中断状態ＳＴ４」に戻ることをポーリングする。

【０１２１】

＜ＷＯＲＫＥＲ６によるタスク実行＞

ＷＯＲＫＥＲ６の側では、タスクトレイ状態TTRAY__STが「優先タスク実行状態ＳＴ５」に遷移すると、開始アドレスセレクタ４５が優先開始アドレスPTT__STA__ADDRを選択すると共に、終了アドレスセレクタ４６が優先終了アドレスPTT__STP__ADDRを選択する。

【０１２２】

遷移検出部４３は、タスクトレイ状態TTRAY__STが「通常タスク中断状態ＳＴ４」から「優先タスク実行状態ＳＴ５」へ遷移する時にロード指令Ｓ３３を出力する機能を有しており、このロード指令Ｓ３３のトリガ出力により優先開始アドレスPTT__STA__ADDRをプログラムカウンタ４１にロードする。

【０１２３】

このロード時点では、タスクトレイ状態TTRAY__STは「優先タスク実行状態ＳＴ５」に遷移している。開始アドレスセレクタ４５は、優先開始アドレスPTT__STA__ADDRの側を選択しているため、プログラムカウンタ４１は、優先開始アドレスPTT__STA__ADDRをロードする。その後、演算命令デコーダ３２は、プログラムカウンタ４１の指し示す演算命令テーブル５の中の演算命令コードをデコードする。

【０１２４】

従ってＷＯＲＫＥＲ６は、「優先タスク実行状態ＳＴ５」への遷移をもって、優先開始アドレスPTT__STA__ADDRから優先終了アドレスPTT__STP__ADDRまでのアドレスに従った演算内容について、演算命令テーブル５を参照しながら、ＦＰＵ９、ＦＰＵ第１引数汎用レジスタ群７、及びＦＰＵ第２引数汎用レジスタ群８を用いて、浮動小数点の２項演算を順次処理する。これにより、最小限のレイテンシにて高優先度演算タスクを起動できる。

【０１２５】

高優先度演算タスクが、優先終了アドレスPTT__STP__ADDRまで実行されると、プログラムカウンタ４１が優先終了アドレスPTT__STP__ADDRに一致することで一致比較部４２がアクティブとなる。一致比較部４２は、終了アドレス一致信号Ｓ３２を出力することで、タスクトレイ状態TTRAY__STを「優先タスク実行状態ＳＴ５」から「通常タスク中断状態ＳＴ４」に遷移させる。

【０１２６】

図７は図６に続くタイミングチャートである。ＷＯＲＫＥＲ６が、優先開始アドレスPTT__STA__ADDRから優先終了アドレスPTT__STP__ADDRまで実行を完了すると、図７のタイミングｔ２１にてタスクトレイ状態TTRAY__STを「優先タスク実行状態ＳＴ５」から「通常タスク中断状態ＳＴ４」に遷移させる。

【０１２７】

他方、ＯＳステートマシン２２は、少なくともＷＯＲＫＥＲ６が高優先度演算タスクを実行中にDisp状態に留まっている。このため、タスクトレイ状態TTRAY__STが「通常タスク中断状態ＳＴ４」に遷移したことを確認することで、ＷＯＲＫＥＲ６が高優先度演算タスクを実行終了したことを即時把握できる。

【０１２８】

ＯＳステートマシン２２は、タスクトレイ状態TTRAY__STが「通常タスク中断状態ＳＴ４」に遷移したことを確認した後、メッセージキューＭＱ２の状態をRun状態からNothing状態に遷移させる。ＯＳステートマシン２２は、サイクリック遷移を再開し、Post状態に遷移する。

【０１２９】

ＦＰＵＯＳ３は、メッセージキューＭＱ２をRun状態からNothing状態に遷移させる機能を、ＯＳステートマシン２２の任意の固定状態、ここではDisp状態に限定している。

【０１３０】

なお仮に、ＯＳステートマシン２２がDisp状態にて待機することなくサイクリック遷移を継続し、Disp状態以外のタイミングでＷＯＲＫＥＲ６がタスク処理を完了すると、次回のDisp状態までサイクリック遷移を待機する必要があり、リアルタイム応答性に劣ること

になる。

【０１３１】

本実施形態では、ＯＳステートマシン２２は、ＷＯＲＫＥＲ６が高優先度演算タスクを実行している最中に、Disp状態に留まっているため、ＦＰＵＯＳ３は、ＷＯＲＫＥＲ６による高優先度演算タスク処理を終了したタイミングｔ２１にてこの状態遷移を受付けることができ、即時対応できる。

【０１３２】

このように、ＷＯＲＫＥＲ６が高優先度演算タスクを実行中には、ＯＳステートマシン２２の状態をDisp状態に保持しているため、ＷＯＲＫＥＲ６が高優先度演算タスクを実行完了したときに、ＯＳステートマシン２２はＷＯＲＫＥＲ６が実行完了したことを即時認識できる。この結果、ＯＳステートマシン２２は即時対応でき、タスク完了オーバーヘッド時間の増加を抑制できる。

【０１３３】

その後、ＯＳステートマシン２２が、サイクリック遷移を再開して再度Peek状態まで遷移すると、タイミングｔ２３において、ＯＳステートマシン２２は、中断していたメッセージキューＭＱ１の状態をWait状態からPeek状態に遷移させることで、タスク起動待ち状態にした後、ＯＳステートマシン２２のサイクリック遷移をDisp状態に遷移させる。ＯＳステートマシン２２は、タイミングｔ２３のDisp状態においてメッセージキューＭＱ１の状態をPeek状態からRun状態に遷移させる。

【０１３４】

通常、ＯＳステートマシン２２が、Disp状態に遷移すると、通常タスクトレイNTASK__TRAYにメッセージリスト４から開始アドレス及び終了アドレスを格納するが、タイミングｔ２３においては、タスクトレイ状態TTRAY__STが「通常タスク中断状態ＳＴ４」であり、タスクトレイ２５の通常開始アドレスNTT__STA__ADDRには中断アドレスが記憶されている。

【０１３５】

このためＯＳステートマシン２２は、メッセージリスト４におけるアドレス検索処理をスキップし、タスクトレイ状態TTRAY__STを「通常タスク中断状態ＳＴ４」から「通常タスク実行状態ＳＴ２」に遷移させ、ＯＳステートマシン２２がサイクリック遷移を再開する。

【０１３６】

タスクトレイ２５のタスクトレイ状態TTRAY__STが、「通常タスク中断状態ＳＴ４」から「通常タスク実行状態ＳＴ２」へ遷移すると、ＷＯＲＫＥＲ６の側では、遷移検出部４３が「通常タスク中断状態ＳＴ４」から「通常タスク実行状態ＳＴ２」への遷移を検出し、このタイミングにて再開ロード信号Ｓ３０を出力する。

【０１３７】

再開ロード信号Ｓ３０がトリガ出力されると、中断時に退避した中断アドレスがプログラムカウンタ４１にロードされる。再開ロードする時点において、タスクトレイ状態TTRAY__STは「通常タスク実行状態ＳＴ２」となっている。このため開始アドレスセレクタ４５は、通常開始アドレスNTT__STA__ADDRの側を選択し、通常開始アドレスNTT__STA__ADDRの格納領域に退避された中断アドレスをプログラムカウンタ４１にロードする。プログラムカウンタ４１に中断アドレスがロードされると、中断アドレスから演算命令テーブル５の中の演算命令コードが順次実行される。

【０１３８】

一方、終了アドレスセレクタ４６も、通常終了アドレスNTT__STP__ADDRの側を選択しているため、演算命令デコーダ３２が、プログラムカウンタ４１の指示する演算命令テーブル５を読み取り、通常終了アドレスNTT__STP__ADDRまで順次実行する。

【０１３９】

一致比較部４２は、プログラムカウンタ４１の示すアドレス値が通常終了アドレスNTT__STP__ADDRに一致したと判定すると、タスクトレイ状態TTRAY__STを「通常タスク実

行状態ＳＴ２」から「通常タスク完了状態ＳＴ３」に遷移させる。これにより、低優先度演算タスク処理を再開、完了できる。

【０１４０】

ＷＯＲＫＥＲ６は、タイミングｔ２４以降において、中断アドレスから通常終了アドレスNTT__STP__ADDRまでの実行アドレスに従った演算内容を演算命令テーブル５から読出し、ＦＰＵ９、ＦＰＵ第１引数汎用レジスタ群７、ＦＰＵ第２引数汎用レジスタ群８を用いて、浮動小数点の２項演算を順次処理できる。

【０１４１】

ＷＯＲＫＥＲ６が高優先度の演算タスクを実行中には、ＯＳステートマシン２２をDisp状態に保持しているため、ＷＯＲＫＥＲ６が高優先度演算タスクを実行完了したときに、ＯＳステートマシン２２はＷＯＲＫＥＲ６が実行完了したことを即時認識できる。この結果、ＯＳステートマシン２２は即時対応でき、タスク完了オーバーヘッド時間の増加を抑制できる。

【０１４２】

＜比較例＞

現在、一般化されているマルチタスクＯＳでは、複数のプロセスのタスクが衝突したときにコンテキストスイッチ等を発生することを考慮に入れると、処理オーバーヘッドを生じることから高速演算には不向きである。

【０１４３】

＜本実施形態のまとめ＞

本実施形態では、ＦＰＵＯＳ３とＷＯＲＫＥＲ６とがハードウェア的に分離して独立して構成されている。このため、高優先度の実行タスクを中断させることなくアプリケーションタスクを処理でき、割込み時及び復帰時におけるコンテキストスイッチのオーバーヘッド時間分だけ短縮できる。

また、ＷＯＲＫＥＲ６の側で特に最優先となる高優先度の演算タスクを実行しているときには、ＦＰＵＯＳ３が低優先度の演算タスクをＷＯＲＫＥＲ６に対して実行指令に移さないようにしているため、ＷＯＲＫＥＲ６は当該高優先度の演算タスクを中断することなく、当該演算タスクを最速処理できる。

また、本実施形態においては、ＦＰＵＯＳ３とＷＯＲＫＥＲ６との間の中断ハンドシェーク期間ｔ１７～ｔ１８の間、低優先度の演算タスクに係るメッセージキューＭＱ１をRun状態に保持したまま、ＯＳステートマシン２２はDisp状態に留まっている。ＯＳステートマシン２２は、サイクリックに状態を遷移させずDisp状態に留まっており、メッセージキュー２４をRun状態に保持しているため、ＦＰＵＯＳ３は、ＷＯＲＫＥＲ６による演算タスクの実行に係る中断応答に対し、メッセージキュー２４をWait状態に即時変更でき、即時対応できる。これにより、タスク中断オーバーヘッド時間の増加を抑制できる。

【０１４４】

また本実施形態において、ＷＯＲＫＥＲ６が、高優先度の演算タスクに係るタスクメッセージＭＳＧ２を実行開始するタイミング以降、「通常タスク中断状態ＳＴ４」にタスクトレイ状態TTRAY__STを遷移させるまでの期間ｔ２０～ｔ２１には、ＦＰＵＯＳ３は、高優先度のタスクメッセージＭＳＧ２に係るメッセージキューＭＱ２をRun状態に保持したまま、ＯＳステートマシン２２がDisp状態に留まるステート制御を実行している。このため、ＷＯＲＫＥＲ６が高優先度の演算タスクを実行完了したときに、ＯＳステートマシン２２はＷＯＲＫＥＲ６が実行完了したことを即時認識できる。この結果、ＯＳステートマシン２２は即時対応でき、タスク完了オーバーヘッド時間の増加を抑制できる。

【０１４５】

（第３実施形態）

図８及び図９に第３実施形態の追加説明図を示している。従来のマイコンが、ＯＳ処理もアプリケーション処理も１つのハードウェアを用いて順次処理する場合、デッドロックしたか否かはＯＳが定期的に所定の状態に遷移するか否かを監視することで判定できる。この場合、ウォッチドッグタイマを１つ設ければデッドロックしたか否かを判定できる。

【０１４６】
しかし、前述した第１実施形態の構成では、ＦＰＵＯＳ３及びＷＯＲＫＥＲ６がハードウェア上独立して構成されているため、ウォッチドッグタイマが一つだけ設けられていても、一方のハードウェア構成（例えば、ＦＰＵＯＳ３）を監視可能になるものの、他方のハードウェア構成（例えば、ＷＯＲＫＥＲ６）は監視不能になるといった問題を生じる。
【０１４７】
そこで図８に示すように、動作監視用のウォッチドッグタイマ２としては、ＦＰＵＯＳ３の動作を監視するためのＦＰＵＯＳ監視ウォッチドッグタイマ２ａ（オペレーティングシステム処理監視ＷＤＴ相当）、及び、ＷＯＲＫＥＲ６の動作を監視するためのＷＯＲＫＥＲ監視ウォッチドッグタイマ２ｂ（演算処理監視ＷＤＴ相当）を独立に設け、ＦＰＵＯＳ３及びＷＯＲＫＥＲ６をそれぞれ監視するように構成することが望ましい。これにより、ＦＰＵＯＳ３のハードウェア構成、ＷＯＲＫＥＲ６のハードウェア構成のそれぞれの動作を独立して監視できる。
【０１４８】
ＦＰＵＯＳ監視ウォッチドッグタイマ２ａ（以下、ＦＰＵＯＳ監視ＷＤＴ２ａ）はフリーランカウンタにより構成され、カウント値をフリーランカウントする。ＯＳステートマシン２２が、Send状態からPost状態までサイクリックに遷移する度に、ＦＰＵＯＳ監視ＷＤＴ２ａはカウント値をクリアする。
【０１４９】
ＯＳステートマシン２２がデッドロックし、ＦＰＵＯＳ監視ＷＤＴ２ａのカウント値がＦＰＵＯＳ監視ＷＤＴ閾値を超えると、ＦＰＵＯＳ監視ＷＤＴ２ａはＷＤＴエラーを外部に出力する。
【０１５０】
他方、ＷＯＲＫＥＲ監視ＷＤＴ２ｂはフリーランカウンタにより構成され、「通常タスク実行状態ＳＴ２」にてフリーランカウントされる。ＷＯＲＫＥＲ監視ＷＤＴ２ｂは、ＦＰＵＯＳ３のタスクトレイ状態TTRAY__STが「通常タスク実行状態ＳＴ２」以外になるとクリアされる。ＷＯＲＫＥＲ監視ＷＤＴ２ｂのカウント値が、所定のＷＯＲＫＥＲ監視ＷＤＴ閾値を超えると、ＷＯＲＫＥＲ監視ＷＤＴ２ｂはＷＤＴエラーを出力する。
【０１５１】
図９には、ＷＯＲＫＥＲ６が通常の演算タスクを実行中であることを示す「通常タスク実行状態ＳＴ２」から、ＷＯＲＫＥＲ６が演算タスクを正常終了するケースを実線により示している。また、「通常タスク実行状態ＳＴ２」から、ＷＯＲＫＥＲ６が演算タスクを異常終了するケースを破線により示している。
【０１５２】
図９に示す「通常タスク実行状態ＳＴ２」が正常終了する実線ケースでは、ＷＯＲＫＥＲ６が暴走することなく正常動作し続ける。この際、ＷＯＲＫＥＲ６及びＦＰＵ９は、通常開始アドレスNTT__STA__ADDRから通常終了アドレスNTT__STP__ADDRまで演算タスクを終了する。すると、ＷＯＲＫＥＲ６の中のプログラムカウンタブロック３１が、タスクトレイ状態TTRAY__STを「通常タスク実行状態ＳＴ２」から「通常タスク完了状態ＳＴ３」へ書き換える。このとき、ウォッチドッグタイマ２は、タイミングｔ３１においてＷＯＲＫＥＲ監視ＷＤＴ２ｂをクリアする。このため、ＷＤＴエラーが出力されることはない。
【０１５３】
しかし、「通常タスク実行状態ＳＴ２」にて正常終了しないと、タスクトレイ状態TTRAY__STには「通常タスク実行状態ＳＴ２」と保持され続けるため、ＷＯＲＫＥＲ監視ＷＤＴ２ｂのカウント値がＷＯＲＫＥＲ監視ＷＤＴ閾値をいずれ超える。するとウォッチドッグタイマ２は、タイミングｔ３２においてＷＤＴエラーを出力する。これにより、ＷＯＲＫＥＲ６のデッドロックを回避できると共に、ＷＯＲＫＥＲ６の暴走を監視できる。
【０１５４】
なお、ＦＰＵＯＳ監視ＷＤＴ閾値は、ＦＰＵＯＳ３がタスクトレイ２５にタスク中断フ

ラグをセットしたタイミングｔ１７から、ＷＯＲＫＥＲ６の側でタスクトレイ状態TTRAY__STを演算タスクを中断する「通常タスク中断状態ＳＴ４」に遷移させるタイミングｔ１８までの時間Ｔ１（図６参照）、を少なくとも超える時間をカウント可能な閾値に設定することが望ましい。すなわち、ＦＰＵＯＳ監視ＷＤＴ閾値は、時間Ｔ１に所定の第１マージン時間を加算した時間以上をカウント可能な閾値に設定することが望ましい（第１条件）。

【０１５５】

しかも、ＦＰＵＯＳ監視ＷＤＴ閾値は、ＷＯＲＫＥＲ６が高優先度の演算タスクを実行開始してから実行完了するまでの時間Ｔ２（図６及び図７参照）、を少なくとも超える時間をカウント可能な閾値に設定することが望ましい。すなわち、ＦＰＵＯＳ監視ＷＤＴ閾値は、時間Ｔ２に所定の第２マージン時間を加算した時間をカウント可能な値に設定することが望ましい（第２条件）。ＦＰＵＯＳ監視ＷＤＴ閾値は、前記した第１条件及び第２条件の双方を満たす閾値とすることが望ましい。

【０１５６】

本実施形態によれば、ウォッチドッグタイマ２が、ＦＰＵＯＳ３、ＷＯＲＫＥＲ６を監視する機能を備えているため、ＦＰＵＯＳ３もＷＯＲＫＥＲ６もそれぞれ独立して監視できる。ＦＰＵＯＳ３の動作を監視できると共に、ＷＯＲＫＥＲ６のデッドロック及び暴走を監視でき異常を検知できる。

【０１５７】

（他の実施形態）

前述実施形態に限定されるものではなく、例えば、以下に示す変形又は拡張が可能である。

浮動小数点の演算は、単精度、倍精度等に限定されるものではない。また、浮動小数点演算処理を例示して演算処理する形態を説明したが、これに限定されるものではなく、例えば固定小数点演算処理に適用しても良い。

【０１５８】

前述した複数の実施形態の構成、機能を組み合わせても良い。前述実施形態の一部を、課題を解決できる限りにおいて省略した態様も実施形態と見做すことが可能である。また、特許請求の範囲に記載した文言によって特定される発明の本質を逸脱しない限度において考え得るあらゆる態様も実施形態と見做すことが可能である。

【０１５９】

本発明は、前述した実施形態に準拠して記述したが、当該実施形態や構造に限定されるものではないと理解される。本発明は、様々な変形例や均等範囲内の変形をも包含する。加えて、様々な組み合わせや形態、さらには、それらに一要素、それ以上、あるいはそれ以下、を含む他の組み合わせや形態をも、本発明の範疇や思想範囲に入るものである。

【符号の説明】

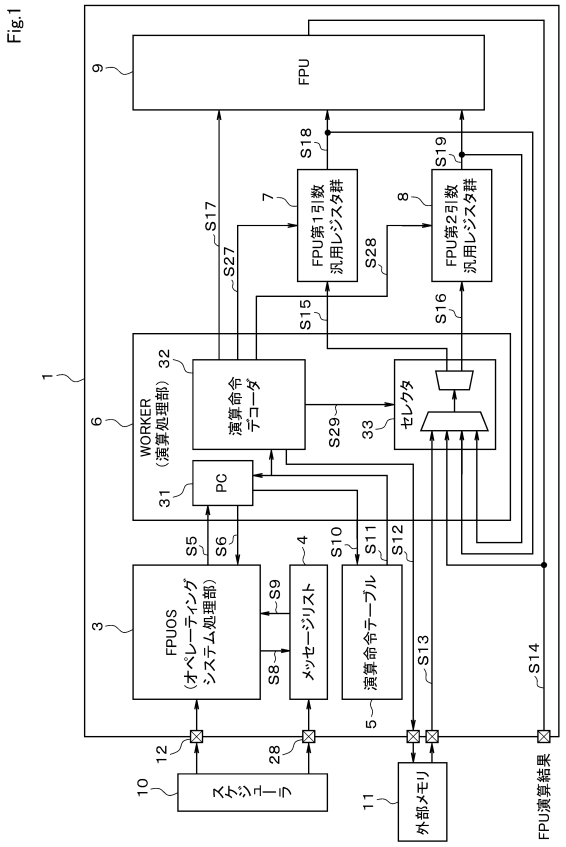
【０１６０】

図面中、１はリアルタイム演算処理装置、３はＦＰＵＯＳ（オペレーティングシステム処理部）、４はメッセージリスト、５は演算命令テーブル、６はＷＯＲＫＥＲ（演算処理部）、２５はタスクトレイ、TTRAY__STはタスクトレイ状態、３１はプログラムカウンタブロック、３２は演算命令デコーダ、４１はプログラムカウンタ、を示す。

【図面】

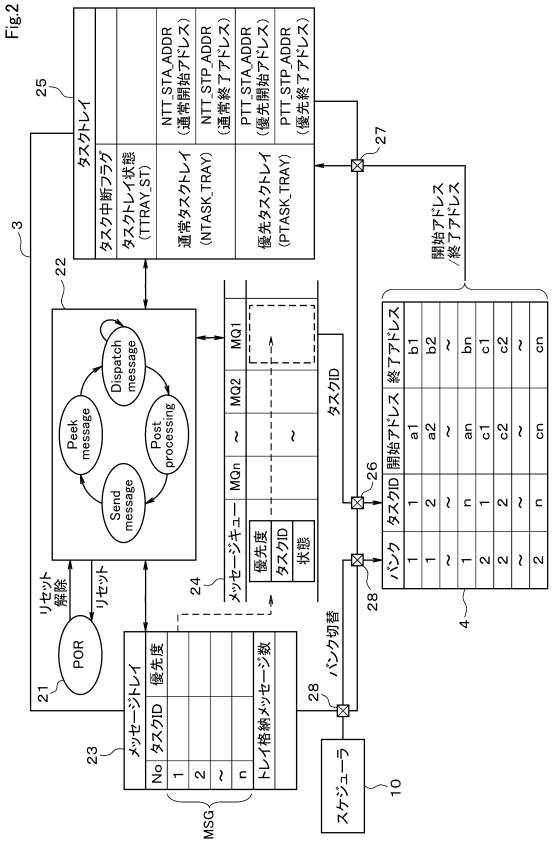
【図１】

Fig.1



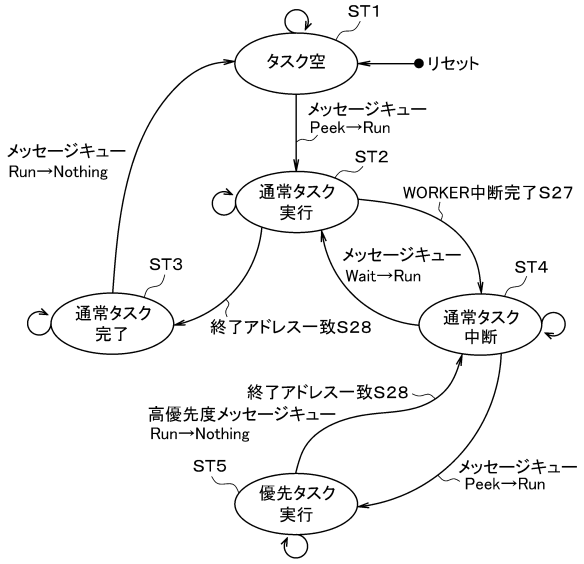
【図２】

Fig.2



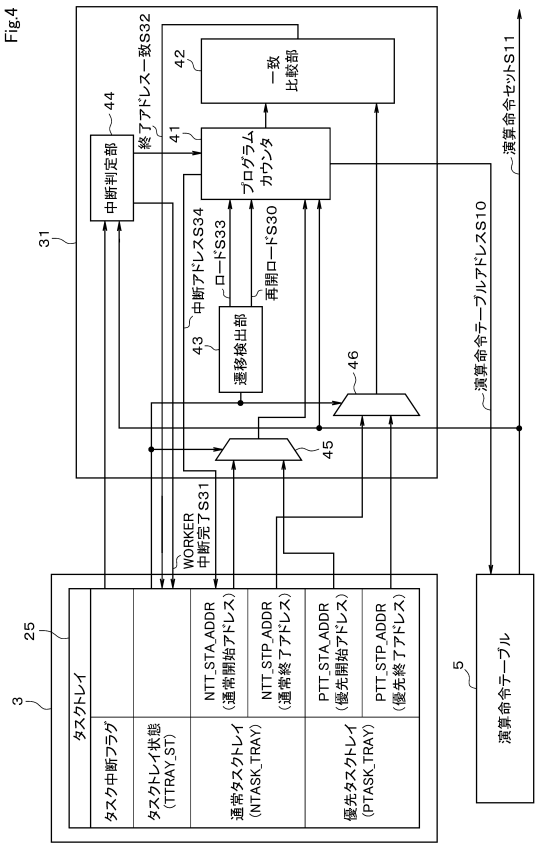
【図３】

Fig.3



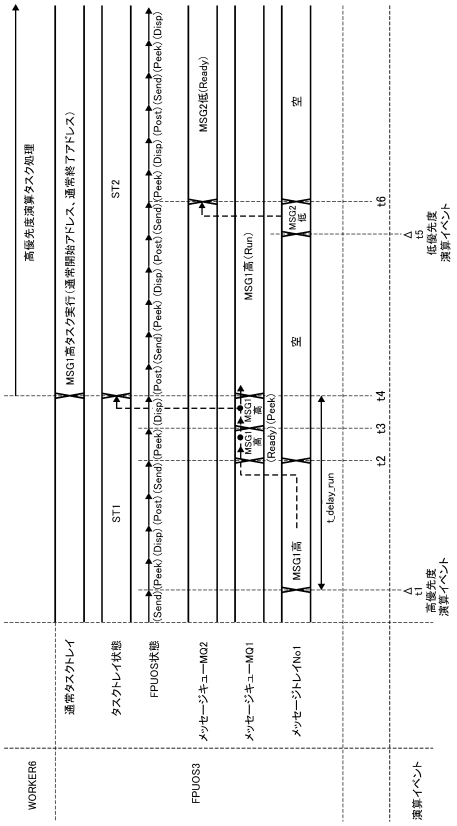
【図４】

Fig.4



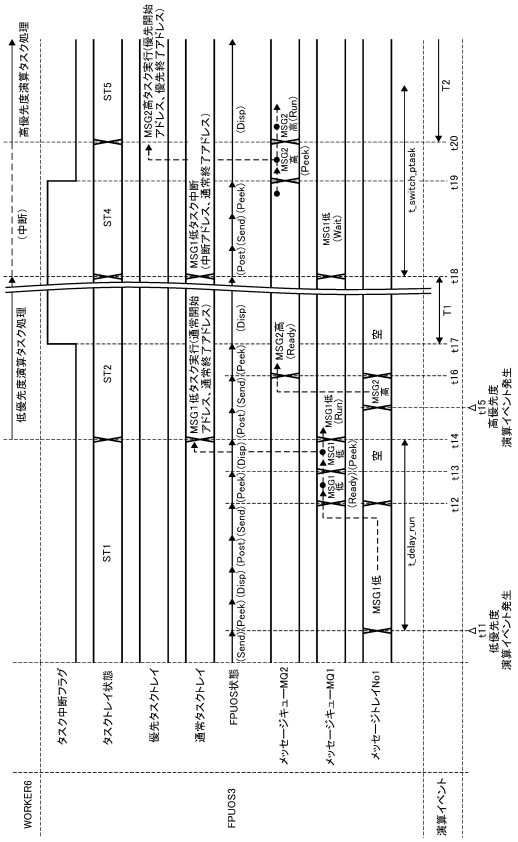
【図５】

Fig.5



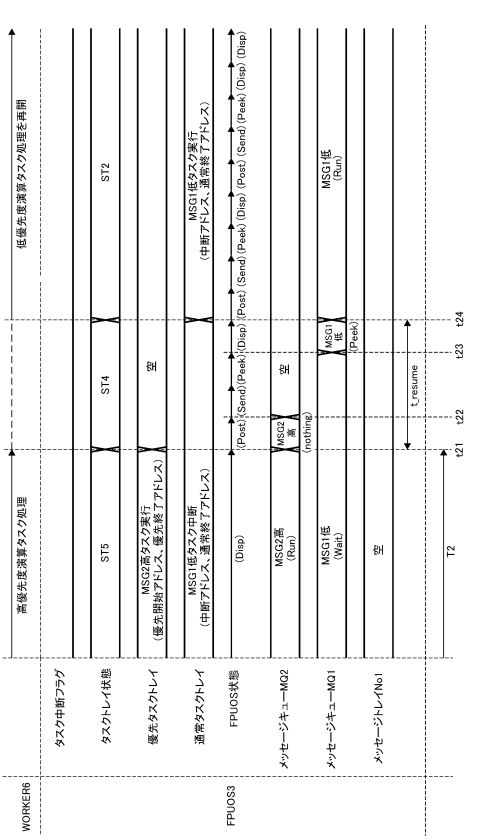
【図６】

Fig.6



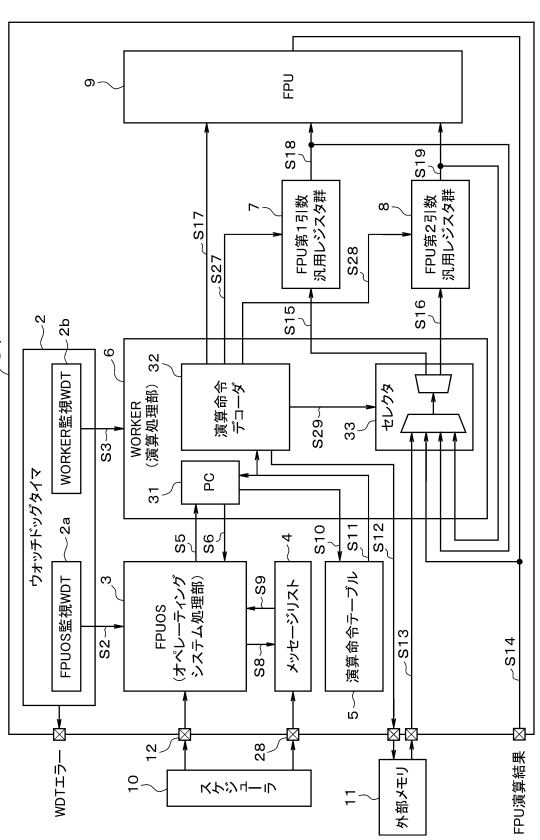
【図７】

Fig.7



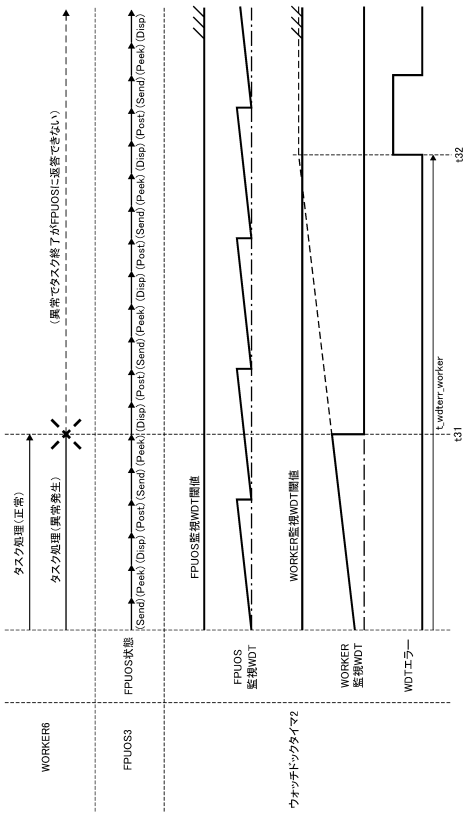
【図８】

Fig.8



【図９】

Fig.9


タスク処理（正常）
タスク処理（異常発生）
（異常でタスク終了がFPUOSに返答できない）
WORKER6
FPUOS3
FPUOS状態
(Send) (Peek) (Disp) (Post) (Send) (Peek) (Disp) (Post) (Send) (Peek) (Disp) (Post) (Send) (Peek) (Disp) (Post) (Send) (Peek) (Disp)
FPUOS監視WDT閾値
WORKER監視WDT閾値
ウォッチドッグタイマ2
FPUOS
監視WDT
WORKER
監視WDT
WDTエラー
t32
t31
t_waiter_worker

フロントページの続き

(56)参考文献　特開昭６０－１１８９３９（ＪＰ，Ａ）
特開２００１－３１８７９６（ＪＰ，Ａ）
特開平０９－１９８２５６（ＪＰ，Ａ）

(58)調査した分野　(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　９／４８
Ｇ０６Ｆ　９／３８
Ｇ０６Ｆ　１５／１６７