

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第6220863号  
(P6220863)

(45) 発行日 平成29年10月25日(2017.10.25)

(24) 登録日 平成29年10月6日(2017.10.6)

(51) Int.Cl. F I  
**G06F 11/36 (2006.01)**  
 G06F 11/36 1 3 6  
 G06F 11/36 1 5 6

請求項の数 20 (全 24 頁)

(21) 出願番号	特願2015-511620 (P2015-511620)	(73) 特許権者	397050741
(86) (22) 出願日	平成25年5月7日(2013.5.7)		マイクロチップ テクノロジー インコーポレイテッド
(65) 公表番号	特表2015-516637 (P2015-516637A)		MICROCHIP TECHNOLOGY INCORPORATED
(43) 公表日	平成27年6月11日(2015.6.11)		アメリカ合衆国 85224-6199
(86) 国際出願番号	PCT/US2013/039934		アリゾナ チャンドラー ウェスト チャンドラー ブルヴァード 2355
(87) 国際公開番号	W02013/169766	(74) 代理人	100078282
(87) 国際公開日	平成25年11月14日(2013.11.14)		弁理士 山本 秀策
審査請求日	平成28年5月6日(2016.5.6)	(74) 代理人	100113413
(31) 優先権主張番号	61/643,725		弁理士 森下 夏樹
(32) 優先日	平成24年5月7日(2012.5.7)		
(33) 優先権主張国	米国 (US)		

最終頁に続く

(54) 【発明の名称】 リセット条件トレース能力を伴うプロセッサデバイス

## (57) 【特許請求の範囲】

## 【請求項 1】

デバッグ能力を有するプロセッサデバイスであって、前記プロセッサデバイスは、  
中央処理ユニットと、

トレースモジュールを含むデバッグ回路と、

内部クロック信号を提供するためのシステムクロックモジュールであって、処理デバイスをリセットすることは、前記システムクロックモジュールが最小リセット期間中にリセットされることを要求する、システムクロックモジュールと、

デバッグモードの間、前記システムクロックモジュールがリセット信号を受信することを防止するリセット検出ユニットと

を備え、

前記トレースモジュールは、リセット信号が検出された後、トレース情報を記録するように動作可能であり、前記トレース情報を記録した後、前記システムクロックモジュールをリセットする、前記最小リセット期間を有する信号を生成する、プロセッサデバイス。

## 【請求項 2】

前記トレースモジュールは、前記システムクロックモジュールがリセットされるときに、前記トレースモジュールの中のデータを損失せずに中断するように構成されている、請求項 1 に記載のプロセッサデバイス。

## 【請求項 3】

前記トレース情報は、リセットソース情報を含む、請求項 1 に記載のプロセッサデバイ

ス。

【請求項 4】

前記トレースモジュールは、実行された命令に関する情報を含むトレースストリームを生成するように構成されており、前記トレースストリームは、外部インターフェースを通して出力される、請求項 1 に記載のプロセッサデバイス。

【請求項 5】

前記トレースストリームは、パケットベースである、請求項 4 に記載のプロセッサデバイス。

【請求項 6】

トレースパケットは、トリガソースに関する情報を含む、請求項 5 に記載のプロセッサデバイス。

【請求項 7】

前記情報は、条件付きで提供され、前記条件は、ユーザ定義されることができ、請求項 6 に記載のプロセッサデバイス。

【請求項 8】

リセット信号は、同期リセット信号または非同期リセット信号であることができる、請求項 1 に記載のプロセッサデバイス。

【請求項 9】

同期リセット信号は、ウォッチドッグタイマ WDT、前記中央処理ユニットによって実行される R E S E T 命令、スタックオーバーフロー / アンダーフローリセットによって生成される、請求項 8 に記載のプロセッサデバイス。

【請求項 10】

非同期リセット信号は、前記プロセッサデバイスの外部ピンを通して受信される、請求項 8 に記載のプロセッサデバイス。

【請求項 11】

内部クロック信号を提供するためのシステムクロックモジュールを備えるプロセッサデバイス内で実行されたコードをデバッグする方法であって、処理デバイスをリセットすることは、前記システムクロックモジュールが最小リセット期間中にリセットされることを要求し、前記方法は、

中央処理ユニット (CPU) によってコードを実行することと、

リセットの決定に応じて、前記システムクロックモジュールを除く、マイクロコントローラの内部ユニットにリセット信号を自動転送することにより、トレースモジュールのさらなる動作を可能にすることと、

前記トレースモジュールによって、前記リセットの受信後、トレース情報を記録することと、

前記トレース情報を記録した後、前記システムクロックモジュールをリセットする、前記最小リセット期間を有する信号を生成することと

を含む、方法。

【請求項 12】

前記システムクロックモジュールがリセットされる間、前記トレースモジュールの中のデータを損失せずに前記トレースモジュールを中断することをさらに含む、請求項 11 に記載の方法。

【請求項 13】

前記トレース情報は、リセットソース情報を含む、請求項 12 に記載の方法。

【請求項 14】

前記トレースモジュールは、実行された命令に関する情報を含むトレースストリームを生成し、前記トレースストリームは、外部インターフェースを通して出力される、請求項 11 に記載の方法。

【請求項 15】

前記トレースストリームは、パケットベースである、請求項 14 に記載の方法。

10

20

30

40

50

**【請求項 16】**

トレースパケットは、トリガソースに関する情報を含む、請求項 15 に記載の方法。

**【請求項 17】**

前記情報は、条件付きで提供され、前記条件は、ユーザ定義されることができる、請求項 16 に記載の方法。

**【請求項 18】**

リセット信号は、同期リセット信号または非同期リセット信号であることができる、請求項 11 に記載の方法。

**【請求項 19】**

同期リセット信号は、ウォッチドッグタイマ WDT、前記中央処理ユニットによって実行される RESET 命令、スタックオーバーフロー / アンダーフローリセットによって生成される、請求項 18 に記載の方法。

**【請求項 20】**

非同期リセット信号は、前記プロセッサデバイスの外部ピンを通して受信される、請求項 18 に記載の方法。

**【発明の詳細な説明】****【技術分野】****【0001】**

(関連出願の引用)

本願は、米国仮特許出願第 61 / 643, 725 号 (2012 年 5 月 7 日出願、名称「PROCESSOR DEVICE WITH INSTRUCTION TRACE CAPABILITIES」) の利益を主張し、該出願は、その全体が本明細書に引用される。

**【0002】**

(技術分野)

本開示は、プロセッサデバイスに関し、特に、統合されたデバッグ能力を伴う、マイクロコントローラデバイスに関する。

**【背景技術】****【0003】**

現代のマイクロプロセッサおよびマイクロコントローラは、いわゆる回路内デバッガまたはエミュレータデバイスを用いて、起動中のプログラムを分析することを効率的に可能にする回路を含む。この目的を達成するために、マイクロコントローラまたはマイクロプロセッサは、例えば、デバッグインターフェースとして動作するように、デバイスの複数の機能ピンをプログラムすることによってアクティブ化されることができるデバッグ機能および特定のインターフェースをサポートする内部回路を提供する。そのようなインターフェースは、通常、高速シリアルインターフェースとして、実際のデバイスと外部デバッガまたはエミュレータとの間の高速通信を可能にするように構成されることができる。したがって、デバイス自体は、このインターフェース、および他の目的のために使用されることができる関連付けられたピンをアクティブ化させない通常動作モードと、このインターフェースを使用して、外部ホストコンピュータからおよびそれによって動作され得る、デバッガまたはエミュレータ等のそれぞれの外部デバイスとデータを交換するデバッグ動作モードとにおいて動作されることができる。デバッガまたはエミュレータはまた、プログラマとして動作されることができ、プログラムは、同一のデバッグインターフェースを介して、標的デバイスに転送される。ホストコンピュータ、外部デバッガ、またはエミュレータは、したがって、安価な分析およびデバッグシステムを形成する。

**【0004】**

現代のプロセッサおよびマイクロコントローラは、それぞれのデバイスの内側に拡張された一式のデバッグ機能を提供する。例えば、いくつかのブレイクポイントが、デバイス内に設定され、デバイスが、実際に、リアルタイムで起動することを可能にすることができ、これは、高速シリアルインターフェースだけを使用するとき、外部デバッガでは不可

10

20

30

40

50

能であり、したがって、ボンダアウトチップおよび高価なデバッグ回路を要求するであろう。しかしながら、これらの内部デバッグ回路網の機能性は、限定されたシリコン専有面積および他の理由のため、必然的に、幾分、制限される。例えば、トレースバック機能は、多くの場合、外部回路内デバッガによってサポートされず、さらにより複雑な回路内エミュレータは、デバッグモードにあるとき、特に、デバイスが、リセットを実行するとき、トレース機能を十分にサポートしない場合がある。

【発明の概要】

【課題を解決するための手段】

【0005】

したがって、プロセッサまたはマイクロコントローラデバイス内の改良された回路内デバッグユニットの必要性が、存在する。例えば、非同期および同期リセットの両方を通した命令トレースが、種々の実施形態によって、可能にされる。

【0006】

ある実施形態によると、デバッグ能力を有するプロセッサデバイスは、中央処理ユニットと、トレースモジュールを含むデバッグ回路と、内部クロック信号を提供するためのシステムクロックモジュールと、デバッグモードの間、システムクロックモジュールがリセット信号を受信することを防止するリセット検出ユニットとを備え得る。

【0007】

さらなる実施形態によると、トレースモジュールは、リセット信号が検出された後、トレース情報を記録し、トレース情報を記録した後、システムクロックモジュールをリセットする信号を生成するように動作可能であることができる。さらなる実施形態によると、トレース情報は、リセットソース情報を含み得る。さらなる実施形態によると、トレースモジュールは、実行された命令に関する情報を含むトレースストリームを生成し得、トレースストリームは、外部インターフェースを通して出力される。さらなる実施形態によると、トレースストリームは、パケットベースであることができる。さらなる実施形態によると、トレースパケットは、トリガソースに関する情報を含み得る。さらなる実施形態によると、情報は、条件付きで提供されることができ、条件は、ユーザ定義されることができる。さらなる実施形態によると、リセット信号は、同期リセット信号または非同期リセット信号であることができる。さらなる実施形態によると、同期リセット信号は、ウォッチドッグタイマWDT、中央処理ユニットによって実行されるRESET命令、スタックオーバーフロー/アンダーフローリセットによって生成されることができる。さらなる実施形態によると、非同期リセット信号は、プロセッサデバイスの外部ピンを通して受信されることができる。

【0008】

別の実施形態によると、プロセッサデバイス内で実行されるコードをデバッグする方法は、中央処理ユニット(CPU)によってコードを実行することと、リセットの決定に応じて、システムクロックモジュールを除く、マイクロコントローラの内部ユニットにリセット信号を自動転送することにより、トレースモジュールのさらなる動作を可能にすることと、リセットの受信後、トレース情報を記録することとを含み得る。

【0009】

本方法のさらなる実施形態によると、本方法はさらに、トレース情報が記録された後、システムクロックモジュールをリセットすることを含み得る。本方法のさらなる実施形態によると、トレース情報は、リセットソース情報を含み得る。本方法のさらなる実施形態によると、トレースモジュールは、実行された命令に関する情報を含むトレースストリームを生成し得、トレースストリームは、外部インターフェースを通して出力される。本方法のさらなる実施形態によると、トレースストリームは、パケットベースであることができる。本方法のさらなる実施形態によると、トレースパケットは、トリガソースに関する情報を含み得る。本方法のさらなる実施形態によると、情報は、条件付きで提供されることができ、条件は、ユーザ定義されることができる。本方法のさらなる実施形態によると、リセット信号は、同期リセット信号または非同期リセット信号であることができる。本

10

20

30

40

50

方法のさらなる実施形態によると、同期リセット信号は、ウォッチドッグタイマWDT、中央処理ユニットによって実行されるRESET命令、スタックオーバーフロー/アンダーフローリセットによって生成されることができ、本方法のさらなる実施形態によると、非同期リセット信号は、プロセッサデバイスの外部ピンを通して受信されることができ、

本願明細書は、例えば、以下の項目も提供する。

(項目1)

デバッグ能力を有するプロセッサデバイスであって、  
中央処理ユニットと、  
ソーストレースモジュールを含むデバッグ回路と、  
内部クロック信号を提供するためのシステムクロックモジュールと、  
デバッグモードの間、前記システムクロックモジュールがリセット信号を受信すること  
を防止するリセット検出ユニットと  
を備えている、プロセッサデバイス。

10

(項目2)

前記トレースモジュールは、リセット信号が検出された後、トレース情報を記録するよ  
うに動作可能であり、前記トレース情報を記録した後、前記システムクロックモジュール  
をリセットする信号を生成する、項目1に記載のプロセッサデバイス。

(項目3)

前記トレース情報は、リセットソース情報を含む、項目2に記載のプロセッサデバイス  
。

20

(項目4)

前記トレースモジュールは、実行された命令に関する情報を含むトレースストリームを  
生成し、前記トレースストリームは、外部インターフェースを通して出力される、項目1  
に記載のプロセッサデバイス。

(項目5)

前記トレースストリームは、パケットベースである、項目4に記載のプロセッサデバイ  
ス。

(項目6)

前記トレースパケットは、トリガソースに関する情報を含む、項目5に記載のプロセッ  
サデバイス。

30

(項目7)

前記情報は、条件付きで提供され、前記条件は、ユーザ定義されることができ、項目  
6に記載のプロセッサデバイス。

(項目8)

リセット信号は、同期リセット信号または非同期リセット信号であることができ、項  
目1に記載のプロセッサデバイス。

(項目9)

同期リセット信号は、ウォッチドッグタイマWDT、前記中央処理ユニットによって実  
行されるRESET命令、スタックオーバーフロー/アンダーフローリセットによって生  
成される、項目8に記載のプロセッサデバイス。

40

(項目10)

非同期リセット信号は、前記プロセッサデバイスの外部ピンを通して受信される、項目  
8に記載のプロセッサデバイス。

(項目11)

プロセッサデバイス内で実行されたコードをデバッグする方法であって、  
中央処理ユニット(CPU)によってコードを実行することと、  
リセットの決定に応じて、システムクロックモジュールを除く、前記マイクロコントロ  
ーラの内部ユニットにリセット信号を自動転送することにより、トレースモジュールのさ  
らなる動作を可能にすることと、

50

前記リセットの受信後、トレース情報を記録することとを含む、方法。

(項目 1 2 )

前記トレース情報が記録された後、前記システムクロックモジュールをリセットすることをさらに含む、項目 1 1 に記載の方法。

(項目 1 3 )

前記トレース情報は、リセットソース情報を含む、項目 1 2 に記載の方法。

(項目 1 4 )

前記トレースモジュールは、実行された命令に関する情報を含むトレースストリームを生成し、前記トレースストリームは、前記外部インターフェースを通して出力される、項目 1 1 に記載の方法。

(項目 1 5 )

前記トレースストリームは、パケットベースである、項目 1 4 に記載の方法。

(項目 1 6 )

前記トレースパケットは、トリガソースに関する情報を含む、項目 1 5 に記載の方法。

(項目 1 7 )

前記情報は、条件付きで提供され、前記条件は、ユーザ定義されることができ、項目 1 6 に記載の方法。

(項目 1 8 )

リセット信号は、同期リセット信号または非同期リセット信号であることができる、項目 1 1 に記載の方法。

(項目 1 9 )

同期リセット信号は、ウォッチドッグタイマ W D T、前記中央処理ユニットによって実行される R E S E T 命令、スタックオーバーフロー / アンダーフローリセットによって生成される、項目 1 8 に記載の方法。

(項目 2 0 )

非同期リセット信号は、前記プロセッサデバイスの外部ピンを通して受信される、項目 1 8 に記載の方法。

**【図面の簡単な説明】**

**【 0 0 1 0 】**

**【図 1】**図 1 は、種々の実施形態による、統合されたデバッグモジュールのブロック図を示す。

**【図 2】**図 2 は、図 1 のハードウェアブレイクポイントブロック図をより詳細に示す。

**【図 3】**図 3 は、図 1 のイベントコンバイナブロック図をより詳細に示す。

**【図 4】**図 4 は、トレースクロックを取り扱うためのブロック図を示す。

**【図 5】**図 5 は、図 4 において使用されるような例示的トレース Q - ジェネレータを示す。

**【図 6】**図 6 は、図 4 において使用されるような例示的 I C D アナログリセットユニットをより詳細に示す。

**【図 7】**図 7 は、図 1 のトレースブロック図をより詳細に示す。

**【図 8】**図 8 は、典型的トレース信号タイミング図を示す。

**【図 9】**図 9 は、簡略化されたブロック図のトレースサブシステムを示す。

**【図 1 0】**図 1 0 は、種々の実施形態による、内蔵デバッグユニットを伴うマイクロコントローラと、外部回路内デバッグ ( I C D ) コントローラとを使用するシステムのブロック図を示す。

**【図 1 1】**図 1 1 は、システムクロック c l k i n および関連直交信号の例示的タイミング図を示す。

**【図 1 2】**図 1 2 は、種々の実施形態による、内蔵デバッグユニットを伴う、マイクロコントローラと、外部回路内デバッグ ( I C D ) コントローラとを使用する、システムのブロック図を示す。

10

20

30

40

50

## 【発明を実施するための形態】

## 【0011】

典型的マイクロコントローラデバイスは、いくつかの非同期および同期システムリセットソースを有し得る。マイクロコントローラユニット(MCU)システムが、種々の実施形態に従って、そのようなリセットの発生に応じて、動作を停止しても、デバイスのデバッグユニット内の命令トレースモジュールは、リセット発生およびソースを記録するであろう。この目的を達成するために、種々の実施形態は、外部非同期および内部同期デバイスリセットを通して、命令トレースを提供する。

## 【0012】

種々の実施形態によると、トレースシステムは、システムクロック構成モジュールに対するリセットをブロックし、リセットを同期させ(非同期である場合)、リセットを記録する。システムは、したがって、リセットに対するタイミング感度を有していない。

## 【0013】

図1は、ある実施形態による、マイクロコントローラ内に統合され得る回路内デバッグモジュールのブロック図を示す。しかしながら、前述のような一般的概念は、他のタイプのオンチップデバッグ回路内に実装されることもできる。ブロック図は、例えば、マイクロコントローラ内に実装されることができ、以下の5つの基本ブロックから成り得るモジュールを示す。

- ・ブレイクポイント比較論理 135
- ・ストップウォッチサイクルカウンタ論理 150
- ・制御および状態機械論理 145
- ・トレース論理 120
- ・イベントコンバイナ論理 125

CPU110は、内部バスを通して、ハードウェアブレイクポイントユニット135と、イベント検出ユニット140と、制御論理ユニット145と、バックグラウンドインターフェース155とに連結される。マルチプレクサ160は、専用デバッグクロックおよびデータピン165を通して、外部デバッグとのデバッグユニットの連結を可能にするために使用される。制御論理145は、1つ以上の外部ピンと連結され得る。図1は、例示的TRIGINピン185を示す。そのようなピンは、他の機能を伴わない専用ピンであり得る。しかしながら、特に、低ピンデバイスでは、そのようなピンは、プログラム制御下、異なる周辺デバイスに割り当てられることができ、したがって、そのプログラムされた割当てに従って、異なる機能を行うことができるマルチ機能ピンであり得る。例えば、そのようなピンは、構成レジスタを用いて、トレーストリガ入力機能に加え、シリアルインターフェースクロックピン、デジタルI/Oピン、アナログ入力ピン等として作用するようにプログラムされ得る。図1に示されるように、制御論理はまた、ピン185と同様に、マルチ機能ピンであり得るトリガ出力ピン190を提供する。トレースモジュール120は、トレース機能停止入力ピン115と、トレースクロックおよびデータ出力ピン175と連結される。図1はまた、制御論理145を通して構成可能であり得る、パルス除去フィルタ170および180を示す。そのようなフィルタを通した信号ルーティングは、図1には示されない。いくつかの実施形態によると、ブレイクポイントデバッグは、命令が実行される前に、実行が一時停止されるように実装される(いわゆる「ゼロスキッド」動作)。他のデバッグ実施形態によると、これは、該当せず、コードが、停止または「スキッド」し、プロセッサが一時停止される前に、命令が実行することを可能にする場合、問題を生じさせ得る。外部イベントは、(定義上)命令実行ストリームと非同期である。したがって、その動作は、ゼロスキッド概念に匹敵しない。

## 【0014】

内部信号debug\_en=1であるとき、モジュールは、イネーブルにされ、全「一時停止」イベントを監視し、イベントを生成し、データ捕捉等を行う。内部信号debug\_en=0である場合、全デバッグ論理は、ディスエーブルにされ、モジュールは、最小電力モードを消費するように構成される。

10

20

30

40

50

## 【 0 0 1 5 】

低侵襲性デバッグを行うために、データをデバイスからリアルタイムでストリーミングする手段を有することが有用であり得る。トレースモジュール 1 2 0 は、読み取られる、または特定のアドレスに書き込まれるデータを取り出し、それをトレースポートから伝送する方法をサポートする。これは、リアルタイムウォッチポイントとして説明されることができる。デバイスの通常動作は、ウォッチポイントを使用するとき、インタラプトされない。

## 【 0 0 1 6 】

データ捕捉は、ハードウェアブレークポイントを生成するために使用されるものと同一のハードウェアを使用して行われ得る。一時停止を生成することと同時のデータ捕捉のためにブレークポイントを使用することは、両動作が生じることを可能にするであろう。しかしながら、データ捕捉およびデータマッチングを同時にイネーブルにすることは、特に、2 以上のブレークポイントカウンタに対して、予想外の結果を生成し得る。データ捕捉およびデータ比較は、同一の物理的レジスタを使用するので、比較値は、捕捉毎に更新されるであろう。

## 【 0 0 1 7 】

図 2 は、ハードウェアブレークポイントユニット 1 3 5 をより詳細に示す。ハードウェアブレークポイントは、プログラムまたはデータメモリのいずれか内のアドレスをマッチングすることにおいてブレークするように構成されることができる。この目的を達成するために、それぞれのコンパレータ 2 2 0 および 2 3 0 とデコーディングユニット 2 4 0 とが、図 2 に示されるように、提供される。ブレークポイントが、データメモリ上で動作するように選択されると、ブレークポイントは、加えて、データ値およびマスクを伴って認定され、ある値のみがブレークポイントイベントを生成することを可能にすることができる。データブレークポイントはまた、随意に、読み取りまたは書き込みサイクルにおいてのみブレークするように設定されることができる。あらゆる場合において、ブレークポイントは、ブレークポイントイベントが生成される前に、特定のイベントが、N 回、生じなければならないようなカウンタ 2 1 0 を有する。これは、例えば、いくつかの実施形態によると、1 ~ 2 5 6 回の任意の値に設定されることができる。

## 【 0 0 1 8 】

図 2 のブロック図は、単一ブレークポイントに関して示される。しかしながら、実装されるブレークポイントの数は、種々の実施形態に従って、可変であり、多くのブレークポイントが、存在し得る。図 2 は、種々のパラメータが、ブレークポイントのためのトリガ要件を定義するようにプログラムされることができる、例示的实施形態を示す。他の実施形態によると、より少ないまたはより多いそのようなパラメータが、使用され得る。例えば、ブレークポイントを生成するために必要なブレークポイント発生数は、カウンタ 2 1 0 内の  $BP \times CNT$  パラメータによって設定されることができる。各ブレークポイントモジュールは、同じレジスタを有し得る。

## 【 0 0 1 9 】

ブレークポイントは、イベントチャネル定義の中にリストされ、サイクルカウンタ 1 5 0 を開始または停止するために、イベントコンバイナステージ 1 2 5 を設定またはリセットするために、トレースユニット 1 2 0 を開始または停止するために、あるいはスタックスナップショットを撮影するために使用されることができる。

## 【 0 0 2 0 】

一実施形態によると、ブレークポイントがイネーブルにされるために、制御レジスタ  $ICDBP \times CON$  のビット  $BPEN$  が、設定されなければならない。このビットが、消去される場合、この特定のブレークポイントに対する全回路は、ディスエーブルにされ、ブレークポイントイベントは、生成されないであろう。ブレークポイントは、認定条件の N 回目の発生毎にのみ、アクションをトリガするように構成されることができる。例えば、3 回目の発生毎にトリガするようにブレークポイントを設定するために、カウンタ 2 1 0 は、 $BP \times CNT = 2$  に設定される。それぞれの制御レジスタが、その値をリロードし、

10

20

30

40

50



および／または現在の状態を監視するために、カウンタ 2 1 0 と組み合わせて使用され得る。

#### 【 0 0 2 1 】

ブレークポイントはまた、例えば、関連付けられた構成レジスタ内にそれぞれのビットを設定することによって、実行コンテキスト（メインラインコード、インタラプトハンドラ、または両方）に基づいて認定され得る。ブレークポイントは、したがって、プログラムが選択されたコンテキストから実行するときのみ、生じ得る。

#### 【 0 0 2 2 】

さらに別のブレークポイントパラメータは、構成レジスタ内に、プログラムカウンタ（PC 実行アドレス）を監視することを可能にする、それぞれのビットを設定することによって設定され得る。プログラムメモリーブレークは、ゼロスキッドであり、動作が実行される前に生じる。PC は、トリガ命令のアドレスを示すであろう。

#### 【 0 0 2 3 】

それぞれの制御ビットが、消去されている、例えば、BPAREN = ' 0 ' であるとき、PC が所定のアドレスに等しくなると、ブレークがトリガされる。BPAREN = ' 1 ' であるとき、PC がアドレスの所定の包含範囲内にあると、ブレークがトリガされる。

#### 【 0 0 2 4 】

いくつかの実施形態によると、実行されている命令のみ、ブレークポイントを生成することができる。PC が、実行されていないアドレスにあるとき、ブレークポイントは、トリガしない。これは、以下を含む。

- ・フロー変更命令（CALL、RETURN 等）、
- ・スキップされた命令（BTFS、BTFS、BTFS に従って）、または
- ・PCL、FSR、または他の 2 つのサイクル命令後の次のフェッチ。

#### 【 0 0 2 5 】

別の制御ビットフィールドが、制御レジスタ内で 0 1、1 0、または 1 1 に設定されると、ブレークポイントは、データアクセス、すなわち、アドレスおよび値の両方を監視する。関連付けられたビットの 3 つの状態は、読み取りまたは書き込みサイクルが、ブレークポイントを決定するために使用されるかどうかを選択する。

#### 【 0 0 2 6 】

データブレークポイントは、必然的に、データが読み取られた、または書き込まれた後、命令実行終了時にブレークを生じさせる（規定通りに）。あらゆる場合において、命令は、完了するまで実行する。故に、「ブレーク」は、実際には、次の命令実行時に生じ、PC は、トリガ命令後に命令を示すであろう。ブレークはまた、メモリアドレスおよびデータ値認定子の両方が満たされるときにトリガされ得る。

#### 【 0 0 2 7 】

サイクルカウンタ 1 5 0 は、ユーザコードがプロファイル化され得るようなストップウォッチ機能を提供するために使用されるカウンタである。サイクルカウンタは、それぞれの制御レジスタによって制御される。カウンタ 1 5 0 は、4 つの 8 - ビットカウンタ / レジスタから成り得る。カウンタ 1 5 0 は、CPU の Q - サイクルの終了毎にインクリメントされ得る。すなわち、マルチサイクル命令（例えば、GOTO）は、複数回、カウントを行う。

#### 【 0 0 2 8 】

複数の機能が特定のイベントによって制御されることを可能にするために、可能なソースの全てが、1 つのイベントバスに組み合わせられ得る。これは、サイクルカウンタ 1 5 0、トレースユニット 1 2 0、およびイベントコンバイナユニット 1 2 5 が、同一の設定を使用して、そのアクションを選択することを可能にする。

#### 【 0 0 2 9 】

イベントコンバイナ 3 0 0 は、図 3 に示されるように、複数のイベント入力 3 2 0 を監視し、それらの入力の組み合わせおよびシーケンスに基づいて、一時停止またはトリガ出力 1 9 0 を生成することができる。イベントコンバイナ 3 0 0 は、対応する制御ビットが

10

20

30

40

50

設定されると、イネーブルにされる。ディスエーブルにされたコンバイナ 300 は、出力イベントをもたらさない。イベントコンバイナ 300 は、イベントチャネル定義の中にリストされ、サイクルカウンタ 150 を開始または停止するために、イベントコンバイナステージ 310 を設定またはリセットするために、トレースユニット 120 を開始または停止するために、またはスタックスナップショットを撮影するために使用されることができる。イベントコンバイナステージ 310 は、そのステージに対するそれぞれの制御ビットが、関連付けられた制御レジスタ内に設定されると、独立して、イネーブルにされる。ステージの現在の出力は、関連付けられたステータスビット内に反映されるであろう。ステージ 310 は、図 3 に示されるような暗示される順序を有し、以下のいくつかの方法で組み合わせられることができる。

10

- ・ステージは、個々に、イベントによってアクティブ化されることができる。
- ・ステージは、次の下位ステージがアクティブである間、イベントによってアクティブ化されることができる。
- ・ステージは、個々に、イベントによって非アクティブ化されることができる。
- ・ステージは、イベントによって、または次の下位ステージが非アクティブ化されると、非アクティブ化されることができる。

#### 【0030】

それぞれの制御ビットを設定することによって、組み合わせられたイベントの (N + 1) 回目の発生のみが、出力イベントを信号伝達するであろう。N は、0 ~ 255 に設定されることができる。組み合わせられたトリガ条件が満たされる場合、レジスタは、1 だけデクリメントされる。組み合わせられたトリガ条件が満たされる場合、イベントコンバイナイベントが、生成され、カウンタに、既定値をリロードされる。また、新しいカウント値がそれぞれの制御レジスタに書き込まれると、随時、カウンタ内の値が、リロードされる。例えば、3 回目の発生時にトリガするようにブレイクポイントを設定するために、それぞれのカウンタ値は、2 に設定されるべきである。

20

#### 【0031】

加えて、いくつかの実施形態によると、ピン TRGIN185 にフィードされる外部信号は、ユーザ入力、トレースストリーム内に挿入されるべきトレースパケットを生成し、一時停止を生成し、随意に、TRGOUT 信号をトリガすることを可能にする。「極性」= 0 (図 6) であるとき、トリガ入力、アクティブ高であり、立ち上がりエッジがイベントを生じさせる。「極性」= 1 であるとき、トリガ入力、アクティブ低であり、立ち下がりエッジがイベントを生じさせる。別の制御ビットは、フィルタを制御するために使用され、例えば、認識されるための最小時間の間、入力がアクティブ状態でなければならないことを定義するために使用され得る。より短いパルスは、したがって、無視される。

30

#### 【0032】

TRGIN イベントが、イベントチャネル定義の中にリストされることができ、サイクルカウンタ 150 を開始または停止するために、イベントコンバイナステージ 310 を設定またはリセットするために、トレースユニット 120 を開始または停止するために、またはスタックスナップショットを撮影するために使用されることができる。トリガ入力の変化は、トレースがイネーブルにされている場合、トレースパケットを生成するであろう。

40

#### 【0033】

ブレイクポイント等のイベントが、イネーブルにされたトリガに伴って生じると、TRGOUT ピン 195 上にパルスが、生成される。基本トリガ出力信号動作は、それぞれの制御ビットを設定することによって構成される。これらの制御ビットは、例えば、トリガ出力が、ほぼトリガイイベントの持続時間の間、アサートされるように制御し得る。隣接したイベントまたは重複したイベントは、信号をアサートされた状態に保持し得る。制御ビットはまた、出力が最小期間まで引き伸ばされるどうか制御し得る。TRGOUT ワンショットがトリガされると、そのタイミング期間内に発生するさらなるイベントは、無視さ

50

れるであろう。ワンショットがタイムアウトし、TRGOUTがゼロに戻った後、再び、別のイベントによってトリガされ得る。ワンショットは、エッジトリガされ、イベント信号が持続する場合でも、所定の期間後、消えてなくなるであろう。

#### 【0034】

ソフトウェアは、それぞれの制御ビットを設定することによって、トリガ出力を生じさせ得る。デバイスが、ウェーク状態である場合、ビットは、1サイクル後、ハードウェアによって消去される。TRGOUTもまた、それぞれの制御ビットを書き込むことによって、消去され得、またはデバイスがウェークアップすると、自動的に、消去されるであろう。

#### 【0035】

外部非同期リセットは、典型的には、プロセッサデバイス全体に影響を及ぼす。そのようなイベントが発生したことを適切に記録するために、命令トレースモジュールは、リセットのタイミングに関係なく、リセットによって影響を受けないように設計される。種々の実施形態は、したがって、オンチップデバッグ能力を向上させ、従来のマイクロコントローラデバイス上で以前は利用可能ではなかった特徴を提供する。したがって、通常、高価な専用回路内デバッガからしか知られていない高度なデバッグ能力が、「標準的」マイクロコントローラまたはマイクロプロセッサデバイス内に実装されることができる。

#### 【0036】

図4は、種々の実施形態による、高レベル概念を示す。システムクロック構成モジュール410は、システムクロックicd\_\_sys\_\_clkを生成する。システムクロックは、4相のtrc\_\_q\_\_clk信号、すなわち、trc\_\_q1\_\_clk、trc\_\_q2\_\_clk、trc\_\_q3\_\_clk、およびtrc\_\_q4\_\_clkを生成する、トレースQ-ジェネレータ420によって使用される。図11は、システムクロックclk\_inおよび命令の実行に関する関連直交信号の例示的タイミング図を示す。本実施形態では、clk\_inから派生した4つのクロックq1、q2、q3、q4が、1つの命令を実行するために使用される。トレースクロックは、内部クロックq1、q2、q3、q4から派生され得るか、または同じこれらのクロックであり得る。しかしながら、他の実施形態によると、異なる中央処理アーキテクチャが実装され、例えば、単一サイクルにおいて、命令を実行することを可能にし得る。

#### 【0037】

図5は、トレースQ-ジェネレータ420の図の詳細を示す。これは、例えば、従来の様式において、4つのフリップフロップ510、520、530、および540によって形成される4ビット循環シフトレジスタであることができる。リセットソースのいくつかの実施例は、ウォッチドッグタイマWDTRIセット、中央処理ユニットによって実行されるRESET命令、スタックオーバーフロー/アンダーフローリセット、およびマスタクリアMCLRリセットである。しかしながら、他のリセットソースおよび信号が、適用され得る。

#### 【0038】

図4に戻ると、全体的リセットが生じると、リセットアナログユニット430は、一般的リセットが、システムクロック構成モジュール410を除き、チップ全体にアサートされることを可能にする。これは、システムクロックが、起動し続けることを可能にし、トレースQ-ジェネレータ420、したがって、また、トレースデータジェネレータ120も、動作を継続することができる。マイクロコントローラ中央処理ユニット110が、リセットに保持されている間、トレースデータジェネレータは、リセット関連トレース情報を記録し、次いで、トレースQ-ジェネレータに、記録が終了したことを通知する。

#### 【0039】

全体的リセット条件もまた、例えば、フリップフロップ450および460によって、それぞれ、trc\_\_q2\_\_clk信号およびtrc\_\_q4\_\_clk信号を使用して同期され、トレースQ-ジェネレータ420が停止すると、リセットアナログユニット430は、例えば、フリップフロップ480によって通知され、システムクロック構成モジュール

10

20

30

40

50

430 がリセットすることを可能にする。

【0040】

リセットアナログユニット430の例示的实施形態は、図6に示される。リセットアナログユニット430は、システムクロック構成モジュール410が、そうするように通知された後、リセットすることを可能にする。システムクロック構成モジュール430がリセットすると同時に、ワンショットパルス `trc_reset_os1_pulse` が、生成され、回路全体が、別のリセット発生検出のために再度備えていることができるように、シンクロナイザレジスタ要素450、460、480をリセットする。図6に示されるように、論理は、種々の制御信号を生成するために提供されることができる。他の論理回路も、デバイスの一般的設計に応じて、使用および適合されることができる。

10

【0041】

このような種々の実施形態は、したがって、レベル高感度リセットおよびエッジ高感度リセットの両方に対処する能力を有し、リセットが生じるとすぐにマイクロコントローラシステム全体（システムクロック構成モジュールを除く）が、リセットすることを可能にし、したがって、命令トレースが進行中であってもネイティブ動作により近い。

【0042】

種々の実施形態は、モジュールが、チップリセットが生じるとすぐにリセットすることを防止された場合でも、システムクロック構成モジュールをリセットするために、最小パルス幅持続時間が保証されることを確実にする。

【0043】

以下のセクションは、デバイスと外部デバッグツールとの間のトレースデータインターフェースについて詳述する。例えば、図7に示されるようなトレースサブシステムは、デバッグツールによって捕捉および分析され得る、命令実行ストリームのリアルタイム記録を提供する。トレース動作は、デバッグツールが、ソースコードおよびプログラムメモリコンテンツへのアクセスを有し、CPU動作のいくつかの側面を推測することができる

20

【0044】

命令が、PCLに書き込むと（直接またはINDxを通してのいずれか）、新しいPCは、プログラムメモリ内の任意の場所にあり得る。この場合、命令パケットは、いわゆるフルプログラムカウンタ（FPC）パケットによって取って代わられる。FIFOは、図7に示されるように、コアデータレートをデバッグツールデータレート（TRSTALLによって統制される）と一致させるために使用されることができる。FIFOは、図7に図示されるように、CPUエンコーダから充填され、データエンコーダによって排出される。FIFOは、最大4096命令パケットを保持するが、有効動作サイズは、それぞれの制御ビットによって選択され得る。WATCHイベントパケットもまた、FIFO内に置かれ、したがって、処理中（in-flight）の命令パケットの実際数は、通常、それより少ないであろう。TRCLK出力は、トグルすると、各クロックエッジに伴って、1データワードを出力する。出力クロックレートは、常時、CPU命令レートにリンクされ、ソフトウェアがSYCLKを変更する場合、またはリセットがSYCLKを変更する場合、変化するであろう。それぞれの構成によると、出力レートは、常時、命令期間あたり2つのトレースワードであることができる。レートは、命令あたり1および1/2まで減少されることができるが、これは、FIFOオーバーフローにつながる可能性が高くなるであろう。TRCLK出力は、データ変化に伴って、位相が一致するか、またはTRCPSビットの設定に応じて、位相がずれるかのいずれかとなるように選択されることができる。FIFO内エンコーディングは、ラインエンコーディングとは異なり、したがって、各FIFO場所は、TRDATインターフェースにおいて3データワードも表示し得る。

30

40

【0045】

トレース機能停止機能が、実装されることができ、構成ビットが、TRSTALL入力が有効であるかどうか決定し得る。例えば、制御ビット `TRXSE = 1` および信号 `TRS`

50

T A L L = 1 であるとき、トレース F I F O は、ペイロード境界における排出を停止し、クロックを停止するであろう。T R S T A L L ( t r \_ s t a l l \_ p i n ) が ' 0 ' に戻ると、クロッキングは、再開し、F I F O は、データを再び排出し始めるであろう。トレースがイネーブルにされ、F I F O が空であるとき、I D L E または S L E E P パケットが、デバイスのスリープ状態に従って送信される。F I F O はまた、いくつかの実施形態によると、空にされることができ、トレースが、非アクティブ化されることができる。F I F O が満杯になる場合、システム応答は、それぞれの制御設定に依存し得る。いずれの場合も、オーバーフロー状態（機能停止される、またはデータをポストしない）は、それぞれの制御レジスタによって選択されるように、F I F O が 2 5 % または 7 5 % 満杯のいずれかになるまで、持続するであろう。トレース実装に関する他の機能が、いくつかの実施形態によると、追加され得、説明されるようないくつかの機能は、実装されなくともある。典型的トレース信号タイミング図は、図 8 に示される。

10

#### 【 0 0 4 6 】

トレースペイロードパケットは、C P U コアが実行する命令および選択されたデータイベントをエンコードし、また、トレースストリーム同期を提供する。ある実施形態によると、ほとんどのパケットは、1 つまたは 2 つの 7 ビットワード、あるいは F P C パケットの場合、3 ワードから成り得る。概して、パケットは、「ワード 1 」および随意的「ワード 2 」から成る。しかしながら、他の形式も、異なる実施形態に従って使用され得る。特定の実施形態によると、「ワード 1 」の値は、パケットを識別し、「ワード 2 」が存在するかどうかを暗示する。命令実行と同期するパケット（E X 、E X D、および R E S E T 等）、および W A T C H、R E S E T、および T R I G G E R を含む非同期「イベント」パケットは、以下により詳細に説明される。

20

#### 【 0 0 4 7 】

同期パケットは、実行の順序で送られる。イベントパケットは、イベントとほぼ同時にストリーム中に現れるが、いくつかのイベントが同時に生じる場合、いくつかのレポートは、遅延されるであろう。ある場合には、F P C は、同期レポートであるが、他の場合には、イベントである。トランスポート層（T R ）パケット R E S Y N C、I D L E、および E N D は、インターフェースデータストリームの種々の状態を管理および識別するために、要求に応じて挿入される。T R パケット R E S Y N C および I D L E は、命令トレースを分析すると破棄され得る。

30

#### 【 0 0 4 8 】

R E S Y N C パケットは、受信機が、正しく同期されていることを検証し得るように、それぞれの制御ビットによって規定されるように、周期的に挿入される。所定の時間間隔にほぼ対応する時に、他の F P C がその間隔内で送信されなかった場合のみ、F P C は、ストリームに追加されるであろう。これは、受信機が命令ストリームを正しく追跡していることのチェックを提供する。挿入された F P C は、常時、追従する命令のアドレスを示す。パケット内のワード数は、パケットワード 1 の値によって決定される。R E S Y N C が、「ワード 2 」に対する正しくない値を伴うワード 1 として受信される場合、ストリームは、同期からずれ、エラーがフラグされるはずである。

40

#### 【 0 0 4 9 】

表 1 は、2 ワードパケットを伴う、最悪の場合の状況を図示し、2 番目のワードは、0 x 7 D であり、その後、R E S Y N C が続く。受信機が適切に同期されている場合、受信されたワード # 3 は、ワード 1 となり、ワード # 4 は、ワード 2 となり、パケット # 2 として示される完全な R E S Y N C 対を形成するであろう。パケット # 3 は、ワード # 5 から開始し、正しく解釈されるであろう。

#### 【 0 0 5 0 】

【表 1】

表 1

受信ワード		パケット	
#	値	タイプ／データ	#
1	0x71	EXD 0x7D	1
2	0x7D		
3	0x7D	RESYNC	2
4	0x7D		
5	0x74	EX	3

10

## 【 0 0 5 1 】

表 2 は、第 1 のワードを複製し、受信機を同期から外す、クロックバウンスを伴う同一のデータを示す。ワード # 1 および # 2 は、データ = 0 x 7 1 (但し、これは、正しくない解釈である) を伴う EXD パケットとして受信され、ワード # 3 および # 4 は、RESYNC 対として現れる。ワード # 5 は、新しい RESYNC パケットのワード 1 として見なされるが、ワード # 6 は、0 x 7 D ではなく、同期から外れた状態を表す。ワード # 6 は、新しいパケットを開始する。

20

## 【 0 0 5 2 】

【表 2】

表 2

受信ワード		パケット	
#	値	タイプ／データ	#
1	0x71	EXD 0xF1	1
2	0x71 <sup>(1)</sup>		
3	0x7D	RESYNC	2
4	0x7D		
5	0x7D	Error	3
6	0x74		

30

注記 1: 複製ワードの受信は、クロックケーブルにおけるインピーダンス不整合の典型である。

2: 任意の 0x7D パケットが続く、第 1 の非 0x7D (ワード 1 = 0x7D) は、常時、ワード 1 (または、FPC ワード # 3) であると十分言える。受信機は、ワード # 6 を新しいパケット内の第 1 のワードとして直ちに再解釈しなければならない。

40

## 【 0 0 5 3 】

トレースが開始する場合、またはデバッグ実行のために一時中断された後にトレースが再開する場合に送信される第 1 のパケットは、常時、FPC であろう。フルプログラムカウンタ (FPC) パケットは、ストリーム内に現れる次の命令の絶対アドレスをレポートする。FPC は、以下の状況においてレポートする。

50

- トレースの開始、
- オーバーフロー後の再開、および
- デバッグ後の再開、単に、次の命令のアドレスを示す。

**【 0 0 5 4 】**

F P Cの全ての他の使用は、命令が実行され、ある場合には、その命令に対してレポートされたであろうパケットを置換することを示す。F P Cが、分岐またはプログラムカウンタ変更命令の実行を表す場合、レポートされる値は、分岐標的アドレスである。F P Cに続く命令パケットは、F P Cが指し示す命令の実行を表す。

**【 0 0 5 5 】**

G O T OおよびC A L L命令は、プログラムカウンタP C [ 1 0 : 0 ]の最下位ビットが、デバッグ環境（アセンブリコード内）に対して既知であり、新しいP Cの上位4ビットのみ、上位部分プログラムカウンタ（U P C）パケットにおいてレポートされると仮定する。しかしながら、他の実施形態は、より多いまたはより少ない情報をレポートし得る。レポートされる値は、 $0 \times 0 F \& (P C L A T H > > 3)$ であることができ、P C L A T Hは、プログラムカウンタの上位ビットにラッチする、実装固有のレジスタを表す。相対分岐（B R A）の宛先は、ソースコードに既知であるので、命令は、単に、E Xとしてレポートされる。種々のトレースペイロードが、実装され得る。表3は、異なるペイロード信号の例を示す。

**【 0 0 5 6 】**

【表 3】

表3

ニーモニック	説明	グループ	トレースエンコーディング		送信される7ビットワードの数	TRIEN=0の場合送信される
			ワード1 (ワード3)	ワード2		
WATCH	データトレースウォッチポイント	ヒューズマスタ	0 ccccc d	ddd dddd	2	はい
FPC	フル新PC、P=PC[14:0] 現在の命令の実行を暗示する 2パケットが連続して送信される		100 pppp (PC[14:11])	ppp pppp (PC[10:4])	3	RESYNC間隔内に1 FPC
			101 pppp (PC[3:0])			
UPC	上位部分PC、P=PC[14:11] 現在の命令の実行を暗示する		110 pppp (PC[14:11])		1	いいえ
EXD	命令を実行する、D=記憶されたデータ		111 000d	ddd dddd	2	いいえ
EX	命令を実行する		111 0100		1	いいえ
STALL	命令が実行されない(NOPに強制される) PCは、変更されない(セクション3. 19. 3. 5)		111 0101		1	いいえ
SKIP	命令が実行されない(NOPに強制される) PCは、 インクリメントされる		111 0110		1	いいえ
OVERFLOW	FIFOがオーバーフロー; データが損失された		111 0111	111 0111	2 <sup>(3)</sup>	はい
INT	インタラプトベクトル インタラプトベクトルNに誘導する		111 1000	nnn nnnn Table 3-8	2	いいえ
RESET	CPUがリセットされている PCは、今、RSTVECに等しい		111 1010	v00 nnnn Figure 3-4	2	はい
ERROR	内部エラーが認められる; 詳細については、ハード ウェアドキュメンテーションを参照		111 1010	xlx eeee Figure 3-4	2	はい
TRIGGER	TRGTR=1およびトリガ入力変化		111 1011		1	はい
SLEEP	TREN=2' b1X、FIFOは空、スリープ状態	トランスポート	111 1100		1	はい
RESYNC	周期的再同期 FPTは、同一の間隔で送信されるであろう		111 1101	111 1101	2 <sup>(3)</sup>	はい
IDLE	TREN=2' b1X、FIFOは空 (スリープ状態ではない、TRSTALLではない)		111 1110		1	はい
END	TREN=2' b00		111 1111	111 1111	2 or 3 <sup>(2,3)</sup>	はい
保留			111 001x 111 1001		1	

注記1: 全てのフィールドは、MSB先出で送信される。  
2: ENDパケットは、TRCLKが低状態で停止するように、少なくとも2回、おそらく3回送信される。  
3: 受信機は、OVERFLOW、RESYNC、およびENDを1ワード/パケットとして取り扱うべきである。付録A. 2. 3の議論を参照。

凡例: c=ウォッチポイントのためのチャンネル p=プログラムカウンタ  
d=書き込みデータ

【 0 0 5 7 】

表 4 は、実際のトレース実施例を示す。

【 0 0 5 8 】



【表 4】

表4

サイクル	命令	オペコード	タイプ/データ
	PC(0x)		
1	123	MOVLW HIGH(2300)	EX
2	124	MOVWF PCLATH	EXD 8'h23
3	125	MOVLW #3	EX
4	125	CALL 200	
5			UPC-4 <sup>(1)</sup>
6	2200	BTFSS W,7	EX
7	2201	BRA \$+4	
8			FPC
9	2205	NOP (注記 2)	EX
10	2206	BRW	
11			FPC 220A <sup>(1)</sup>
11	220A	RETLW #77	
12			FPC
13	126	NOP	EX
14	127	CALLW	
15			FPC 2377 <sup>(1)</sup>
16	2377	NOP	EX
17	2378	GOTO 500	
18			UPC-4 <sup>(1)</sup>
19	2500	NOP	EX
20	2501	RETURN	
21			FPC
22	128	NOP	EX

注記1:TRFPCB=1の場合、この命令は、FPCをレポートする

2:PCLATH=8'h23(サイクル2から)であるため、サイクル8のためのPC値は、15'h2205ではない。UIは、このエラーをフラグすべきである。

## 【0059】

トリガ入力の変化は、それぞれの制御ビットが設定されている場合、トレースパケットを生成するであろう。極性ビット = 0 である場合、イベントは、立ち上がりエッジでトリガするであろう。極性ビット = 1 である場合、イベントは、立ち下がりエッジでトリガするであろう。命令が、直接または間接的のいずれかにおいて、プログラムカウンタ PCL に書き込むと、新しい PC は、プログラムメモリ内の任意の場所にあり得る。この場合、命令パケットは、FPC パケットによって置換される。

## 【0060】

FIFO は、コアデータレートをデバッグツールデータレート (TRSTALL によって統制される) と一致させるために使用される。FIFO は、図 6 によって図示されるように、CPU エンコーダから充填され、データエンコーダによって排出される。FIFO は、最大 4096 命令パケットを保持するが、有効動作サイズは、制御レジスタ内のそれぞれのビットによって選択される。WATCH イベントパケットもまた、FIFO 内に置かれ、したがって、処理中の実際の命令パケットの数は、通常、より少ないであろう。

## 【0061】

TRCLK 出力は、トグルすると、各クロックエッジに伴って、1 データワードを出力する。出力クロックレートは、常時、CPU 命令レートにリンクされ、ソフトウェアが SYSCLOCK を変更する場合、またはリセットが SYSCLOCK を変更する場合、変化するで

10

20

30

40

50

あろう。関連付けられた制御レジスタ内のそれぞれのビットフィールドが設定されると、出力レートは、常時、命令期間あたり2トレースワードとなり得る。レートは、いくつかの実施形態によると、命令あたり1および1/2まで減少されることができ、これは、FIFOオーバーフローにつながる可能性が高くなるであろう。トレースクロックTRCLK出力は、データ変化に伴って、位相が一致するか、またはTRCPS制御ビットの設定に応じて、位相がずれるかのいずれかとなるように選択されることができ、FIFO内エンコーディングは、ラインエンコーディングとは異なり、したがって、各FIFO場所は、TRDATインターフェースにおいて3データワードも表し得る。

#### 【0062】

図9は、トレースサブシステム700の簡略化されたブロック図を示す。サブシステム700は、命令コードをコアおよびWATCHイベント信号から受信し、TRDAT信号上における配信のために、このデータをフォーマットする。シーケンスコントローラ710は、命令およびWATCHデータをFIFOにロードすることに関与する。各データパケットは、単一16-ビットワードとしてエンコードされる。q34の間、WATCH信号は、サンプリングされ、最高優先順位の信号が、エンコード、ロード、およびリセットされる。2つ以上の信号が、アサートされる場合、最高優先順位の信号のみ、ロードされ、他は、後続q34の機会を待たなければならない。レポート優先順位は、ブレークポイント番号に基づくので、イベントは、順序外でレポートされ得る。q12の間、以前の命令からのデータは、エンコードおよびロードされる（オブコードは、q3においてエンコードされ、バスデータは、安定しq3-q3、全て、立ち上がりq1において有効である）。概して、これは、命令期間毎に生じる。分岐および呼び出し命令ならびにインタラプトサイクルの場合、第1のサイクルの間、何もロードされず、パケットは、2回目のサイクルの間、エンコードされ（いわゆる「強制NOP」）、UPCおよびFPCが、正しいPC値とともに送られることを可能にする。その結果、2つのWATCHパケットが、分岐の間、ロードされ得る。SKIPおよびSTALLは、現在のコア動作に基づいてエンコードされる。

#### 【0063】

シーケンスコントローラ710は、命令サイクルあたり2回ロードすることができ、ストリームマネージャ730は、命令サイクルあたり2回アンロードすることができ、命令サイクルあたり最大4メモリサイクルを要求する。FIFOコントローラ720は、シーケンスコントローラ710によって提供されるデータを管理する。データは、ストリームマネージャ730によって要求されると、同一の順序で配信される。ストリームマネージャ730は、16-ビットFIFOワードをTRDAT信号で伝送されるデータワードに再フォーマットする。いくつかのパケット（例えば、EX）は、FIFOワード毎に単一TRDATワードをもたらす一方、他（例えば、FPC）は、より多くをもたらす。要求に応じて、トランスポート管理パケットRESYNC、IDLE、およびENDが、ストリーム内に挿入され、読み出し動作は、TRSTALL入力に従って中断される。2つのTRDATワードが、各命令サイクル期間の間に伝送される。TRCLK信号は、システムリセットの間、中断する（サイクルの一部を引き伸ばす）であろう（データは、損失されないであろう）。

#### 【0064】

図10は、トレース受信機として動作するデバッグツール820と連結される、種々の実施形態による、マイクロコントローラ810を伴うシステム800を示す。デバッグツールは、例えば、出願人によって製造された回路エミュレータ内のReal-ICEであることができる。受信機820は、トレース同期を行い、全IDLEおよびトランスポートパケットを破棄し、残りのパケットのバンドルを遠隔ホスト830、例えば、パケットストリーム解釈が行われる、パーソナルコンピュータまたはワークステーションに伝送する。

・ワード1分析 - パケットを3ワード幅データバスに変換する。

・ダブレット（RESYNC、OVERFLOW等）を検証し、（a）ダブレットが連続

10

20

30

40

50

ワード中にないとき、または (b) 非実装ワード 1 コード値が認められるとき、同期エラーをフラグする。

- ・ I D L E および他のトランスポートパケットを破棄し、残りの値を F I F O にスタックする。

- ・ パケット全体を遠隔ホストに伝送する。

#### 【 0 0 6 5 】

T R S T A L L をアサートすると、受信機 8 2 0 は、最大もう 6 つの T R D A T ワード ( 2 つの連続 F P C パケットの均等物 ) に加え、再同期パイプラインの待ち行列に追加され得る 2 ワードを受け取る準備ができる。T R S T A L L を解除すると、パケットワード 1 アラインメント ( a l i g n m e n t ) が、保証される。I D L E パケットのストリングから、大部分、または時として、全部が、破棄され、遠隔ホストに送られず、帯域幅を削減することができる。非実装オブコードおよび F P C ワード 3 ( 7 ' h 5 x ) もまた、同期エラーとしてフラグされ、1 ワードパケットとして取り扱われるべきである。

#### 【 0 0 6 6 】

F P C の 3 番目のワードは、単一ワードパケットのスタイルでエンコードされる ( 例えば、コード 7 ' h 5 x ) 。同期しているかどうかにかかわらず、例えば、7 ' h 5 x の任意のワードに続くワードは、新しいパケットのワード 1 であると仮定され得る。ワード 1 を追跡する目的のために、O V E R F L O W 、 R E S Y N C 、 S L E E P 、および E N D パケットは、1 ワードパケットとして取り扱われるべきである。7 ' h 7 D が、1 ワードパケットとして取り扱われるとき、別の 7 ' h 7 D ( 対にされた R E S Y N C ワード ) である場合でも、続くパケットは、常時、ワード 1 値となるであろう。ワード 1 がアラインされた ( a l i g n e d ) ( 疑似 ) データは、ダブルット分析に通され、2 つの R E S Y N C がストリーム内で連続的に現れないとき、同期失敗が認識される。同様に、O V E R F L O W ( 7 ' h 7 F ) は、受信機が同期から外れているときに現れ得、2 番目のワードが一致しない場合でも、正確に解釈されなければならない。これもまた、同期エラーをフラグし得る。受信機が、同期から外れている場合、1 つのみの 7 ' h 7 C または 7 ' h 7 F が、最後に現れ得、第 2 の値を待つ間、ハングアップすることは受信機にとって好適ではないであろうため、同様の論法は、S L E E P および E N D パケットにも適用されることができる。その一方で、3 つの同じワードが、現れる場合もあり、これも受信機を混乱させるべきではない。

#### 【 0 0 6 7 】

完全分析のために、ホスト 8 3 0 は、トレースデータを、マイクロプロセッサをプログラムするために使用されたオリジナルソースコードと比較しなければならない。大部分の命令に対して、トレースデータは、実行が生じたことを示すが、動作詳細は、含まれない。分岐が、E X パケット ( T R F P C B = 0 ) のみをもたらすとき、分岐先は、ソースコードを検証することによってのみ決定されることができる。同様に、特定の実施形態によると、P C L A T H への書き込みは、部分的データのみを送り、評価を完了するために、ソース知識を要求する。動作に応じて、S T A L L パケットは、影響を受ける命令に先行するか、または続き得る。S T A L L パケットは、ユーザのディスプレイ上にハイライトされた表記によって、非典型的動作 ( 例えば、不揮発性メモリへのファイル選択レジスタ ( F S R ) 書き込み ) が生じたことの示唆として見られ得る。W A T C H レポートは、多くのパケットによって、トリガ命令を遅延させ得る。高密度に取り込まれたウォッチは、同一のウォッチが、読み出しが生じる前にトリガする場合、実際には損失され得る。ウォッチポイントは、データアドレスのみを識別するので、トレース分析は、直接アドレスモード ( B S R およびオブコードの知識を要求する ) から、または間接モード ( F S R の知識を要求する ) からかにかかわらず、データアクセスポイント値を再構成することができる。ウォッチデータは、常時、ウォッチイベントの最新の発生に由来する。いくつかの F P C パケットは、命令が実行された ( 例えば、R E T U R N 命令 ) ことを暗示するが、他のインスタンスは、単なる参考情報であることに留意されたい。

#### 【 0 0 6 8 】

図12は、開発プログラムを起動し、例えば、USBインターフェースを介して、外部デバッガ/プログラミングユニット520と接続される、パーソナルコンピュータ等のホストを伴う、典型的デバッグ/プログラミングシステム500を示す。外部デバッガプログラミングユニット520は、デバッガ/プログラマ520内で生成された電源電圧を供給し得る専用インターフェースを提供する。しかしながら、他の実施形態は、専用電源を介して供給電圧を供給し得、標的システムは、自己給電され得る。実際のデバッグ/プログラミングインターフェースは、デバッガ/プログラミングユニット520によって提供される単方向性クロック信号ICD<sub>CLK</sub>および双方向性データラインICD<sub>Data</sub>を伴う、同期シリアルインターフェースによって提供され得る。したがって、最低限でも、3つの接続ライン、ICD<sub>CLK</sub>、ICD<sub>Data</sub>、および基準電位(GND)が、デバッガ/プログラミングユニット520と、最低限として、前述のように、種々の実施形態による、デバッグ/プログラミングインターフェースを伴うマイクロコントローラであり得る、標的システム510を連結するために使用され得る。

10

#### 【0069】

そのようなシステムは、ユーザが、ホスト上で起動するデバッグプログラムを、プログラムがリアルタイムで実行している間、前述されたような条件を有する種々のブレークポイントを設定し、随意に、追加の接続ラインを通して、トレース情報を受信するようにプログラムすることを可能にする。デバッグソフトウェアは、ソースコード内のその位置に関して、種々のブレークポイントを追跡する間、デバッガ/プログラマ520は、それぞれのブレークポイントを設定し、その関連付けられたレジスタを構成する標的デバイスに、ブレークポイント情報を通信する。また、トレース能力の設定および構成は、デバッガ/プログラマ520によって、標的デバイス510に通信される。例えば、メモリ内に記憶されたデータ値の一致によってトリガされる特定のブレークポイントが、設定され得る。ユーザは、次いで、ホストPC530上で起動するデバッグソフトウェアを通して、標的デバイス510のソフトウェアの実行を開始する。標的ソフトウェアの実行は、ブレークポイントが検出されるときのみ停止される。しかしながら、トレース情報は、標的プログラムの実行の間、継続して自動転送され得る。ホストコンピュータ530は、このトレースデータを評価し、テキスト形態において、またはグラフィカルディスプレイを使用してのいずれかにおいて、それを利用可能にする。前述のように、リセットイベントはまた、種々の実施形態によると、トレースサブシステムが、依然として、クロックされているため、完全にトレースされることができる。

20

30

#### 【0070】

本実施形態は、前述のような特徴を実装することによって、トレースストリームのより優れた分析を可能にする。種々の実施形態は、したがって、オンチップデバッグ能力水準を高め、高度デバッグ能力を多くの異なるユーザにもたらすことを可能にするであろう。

【図 1】

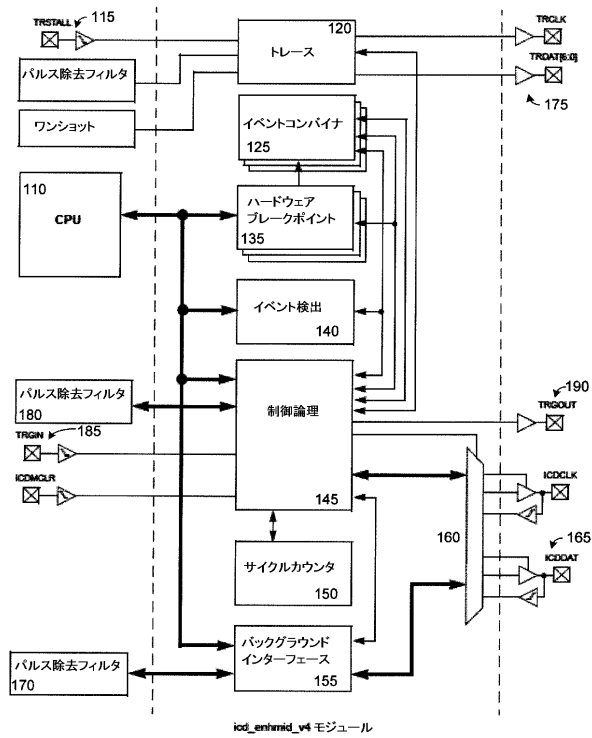


Figure 1

【図 2】

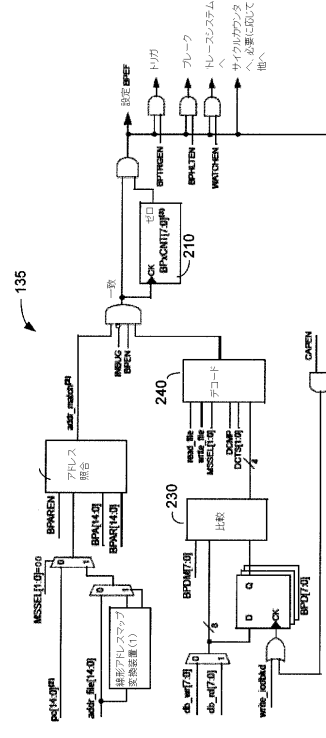


Figure 2

【図 3】

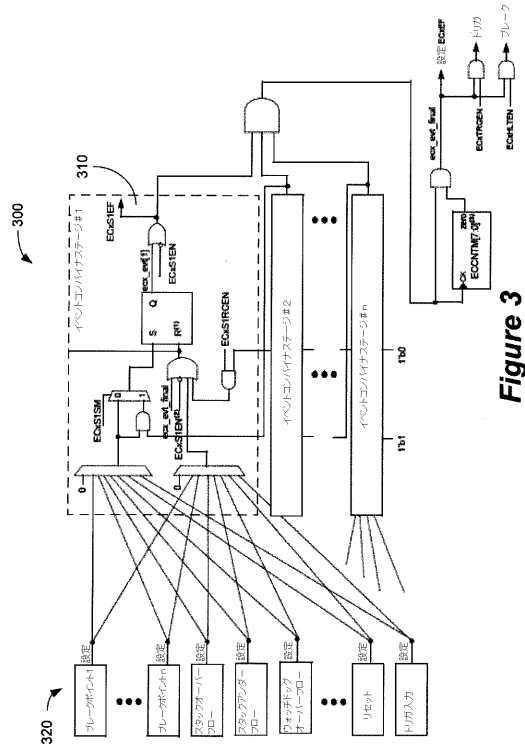


Figure 3

【図 4】

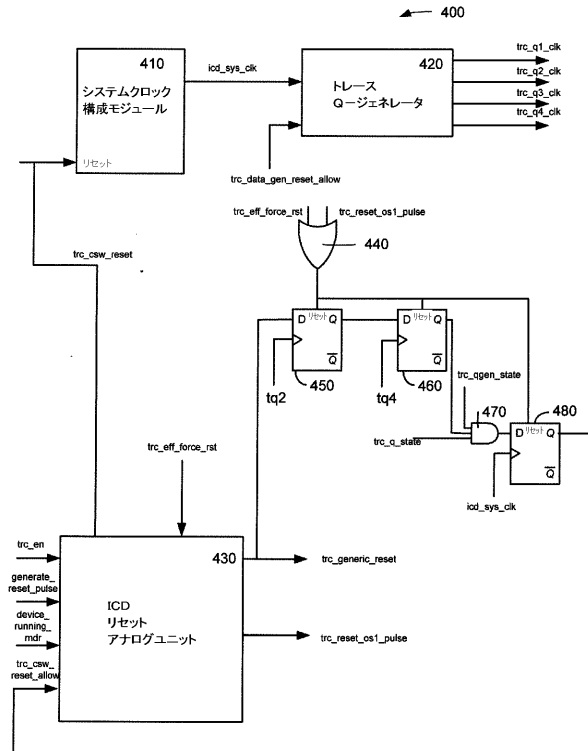


Figure 4

【図 5】

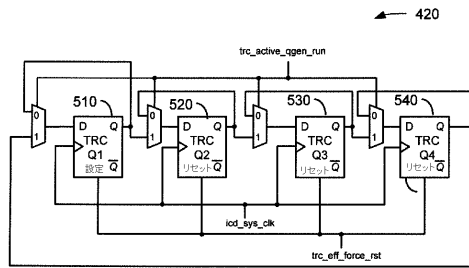


Figure 5

【図 6】

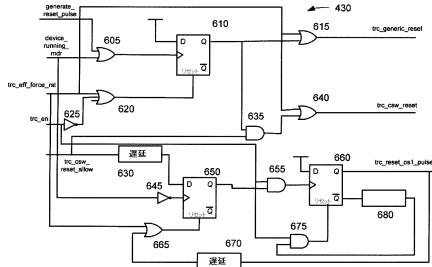


Figure 6

【図 7】

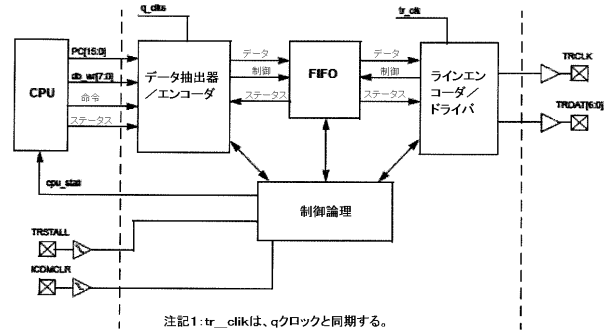


Figure 7

【図 8】

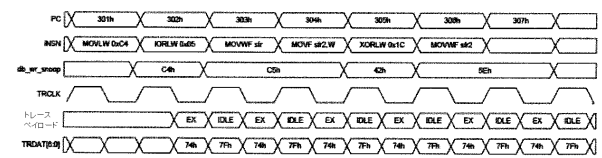


Figure 8

【図 9】

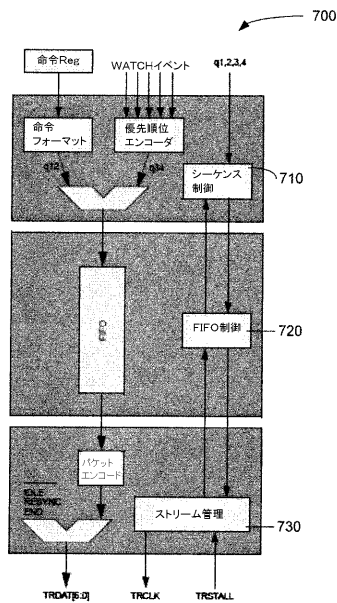


Figure 9

【図 10】

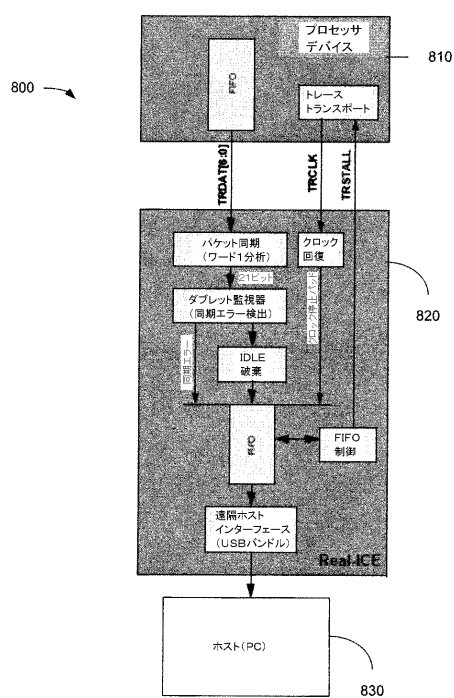


Figure 10

【図 11】

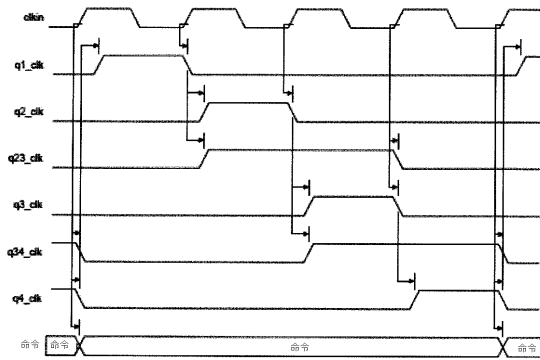


Figure 11

【図 12】

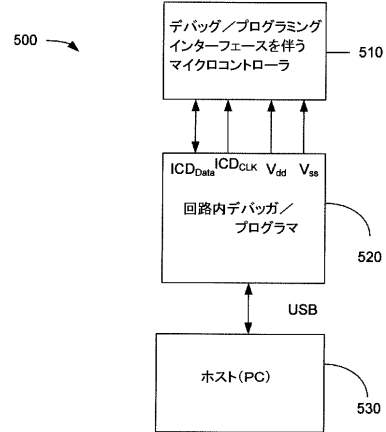


Figure 12

## フロントページの続き

- (72)発明者 ミルクス, ジャスティン  
アメリカ合衆国 アリゾナ 85286, チャンドラー, イースト ウッズマン プレイス  
630
- (72)発明者 パーム, トーマス エドワード  
アメリカ合衆国 アリゾナ 85225, チャンドラー, エス. ユーカリプス プレイス  
222
- (72)発明者 バラスブラマニアン, スンダール  
アメリカ合衆国 アリゾナ 85286, チャンドラー, ダブリュー. ブルーバード ドラ  
イブ 1163
- (72)発明者 ジャバガル, クシャラ  
アメリカ合衆国 アリゾナ 85044, フェニックス, イー. レイ ロード 3625,  
アパートメント 2050

審査官 多賀 実

- (56)参考文献 特開2006-011991(JP,A)  
特開2011-197952(JP,A)  
特開2008-129669(JP,A)  
特開2008-276324(JP,A)  
特開平07-036735(JP,A)  
特開平11-265311(JP,A)

## (58)調査した分野(Int.Cl., DB名)

G06F 11/07  
G06F 11/28 - 11/36  
G06F 15/78