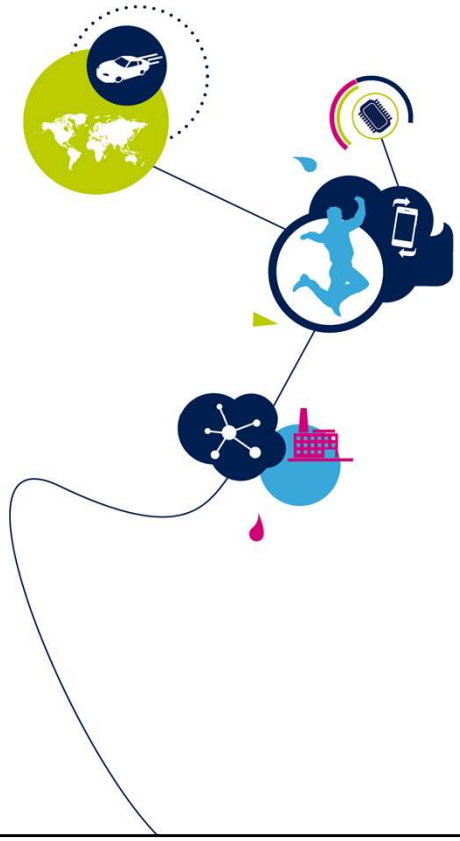
STM32G4 - DBG

デバッグとトレース

1.0 版



life.augmented

STM32 デバッグとトレース用インタフェースのプレゼンテーションにようこそ。STM32G4 デバイスが備えているデバッグの機能について説明します。

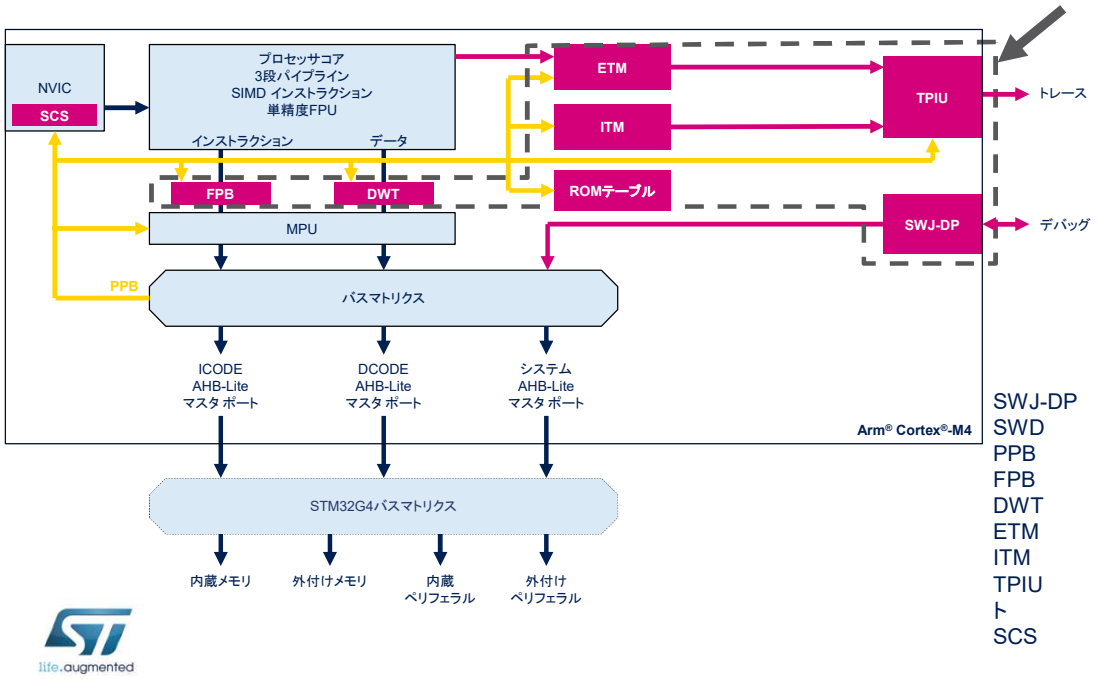
概要

- STM32G4はデバッグのための充実したサポートを提供
  - プログラムを RAM または Flashメモリにダウンロード
  - メモリとレジスタの内容を確認
  - ブレーク･ポイントを挿入してプロセッサを停止
  - ブログラムを実行またはシングル･ステップ実行
- ARM® CoreSight™アーキテクチャ･ベース
  - 広範囲な互換ツール
  - STリンクなど、プローブ･デバッグするための標準のシリアル･ワイヤ･デバッグ･インタフェース

life.augmented

STM32G4 には、STM32 ファミリ MCU が提供するお馴染みのデバッグ機能（Flash ダウンロード、ブレークポイントデバッグ、レジスタ表示とメモリ表示）がすべて内蔵されています。デバッグとトレースのインフラストラクチャは、ほとんどのツールプロバイダ によってサポートされている ARM® CoreSight™規格を採用しています。

デバッグ･アーキテクチャ

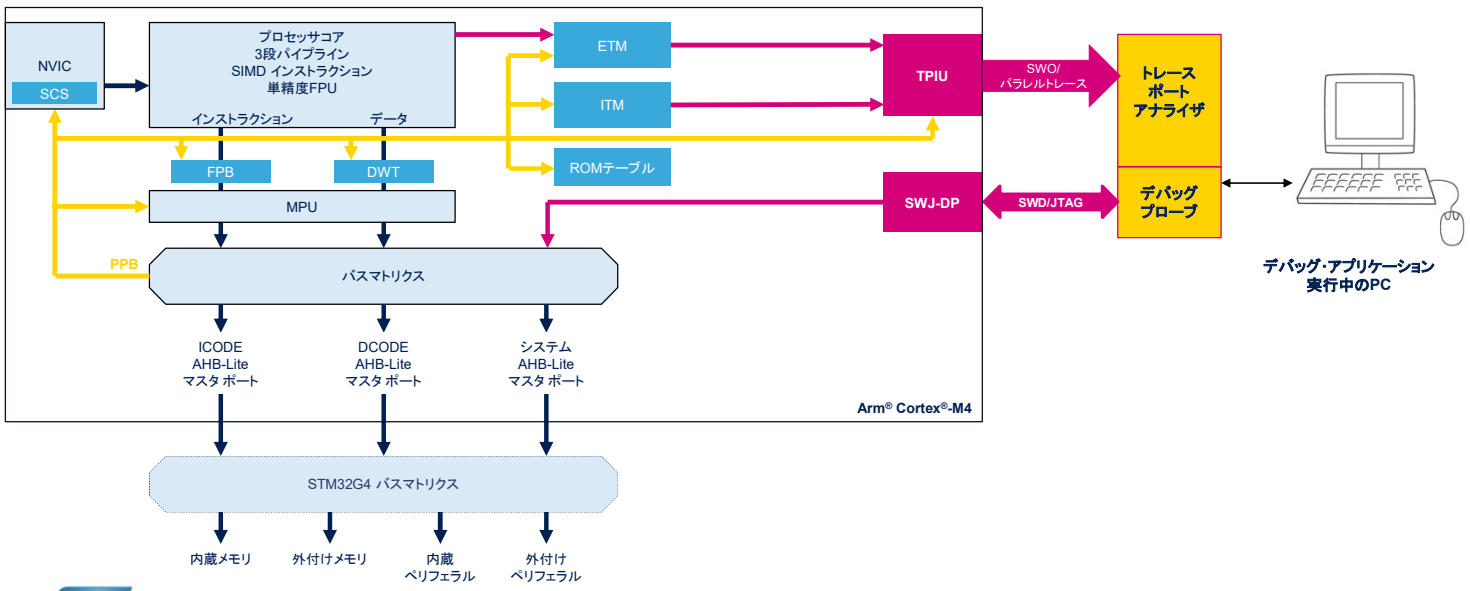


SWJ-DP	= シリアルワ･イヤJTAGデバッグ･ポート
SWD	= シリアル･ワイヤ･デバッグ
PPB	= プライベート･ペリフェラル･バス
FPB	= Flashパッチとブレーク･ポイント
DWT	= データ･ウォッチポイントとトレース
ETM	= 組込みトレース･マクロセル
ITM	= 計装トレース･マクロセル
TPIU	= トレース･ポート･インタフェース･ユニット
SCS	= システム制御スペース

life.augmented

デバッグプロセスに関与するすべてのユニットは、コアとSWJ-DPの両方からプライベートペリフェラルバス(PPB)を介してアクセス可能なメモリマップレジスタを持っています。
デバッガは、プロセッサの実行中にメモリにマップされたリソースにアクセスできます。たとえば、プロセッサが命令を実行している間に、プライベートペリフェラルバスに接続されている FPB ユニットにアクセスし、デバッガによってブレークポイントを設定できます。

SWJ-DP とトレース･ピン



SWJ-DP を使用すると、外部デバッグ プローブは、Cortex®-M4 コアからもアクセス可能なメモリにマップされたリソースにアクセスできます。
メモリ保護ユニット(MPU)は、SWJ-DP によって開始されたリクエストをインタセプトしないことに注意してください。
SWJ-DP は、デバッグ プローブとデータを交換する 2 つのプロトコルをサポートします。

- 2 線式シリアル ワイヤ デバッグ(SWD)プロトコル
- または5線式JTAGプロトコル。

SWJ-DPは、使用されているプロトコルを自動的に検出します。
トレース出力に関しては、TPIU は 2 つの可能性を提供します。

- シリアル ワイヤ出力(SWO)と呼ばれる非同期の 1 線トレースポート
- または、クロック信号と1、2または4ビットのデータを含む同期5線トレースポート。

SWO は JTAG JTDO 信号で多重化されます。したがって、JTAGプロトコルと同時に使用することはできません。SWD を選択する必要があります。

柔軟なSWJ-DPとトレース･ピンの割り当て

- デバッグが必要ではない場合、すべてのデバッグ･ピンは機能用として再割当て可能

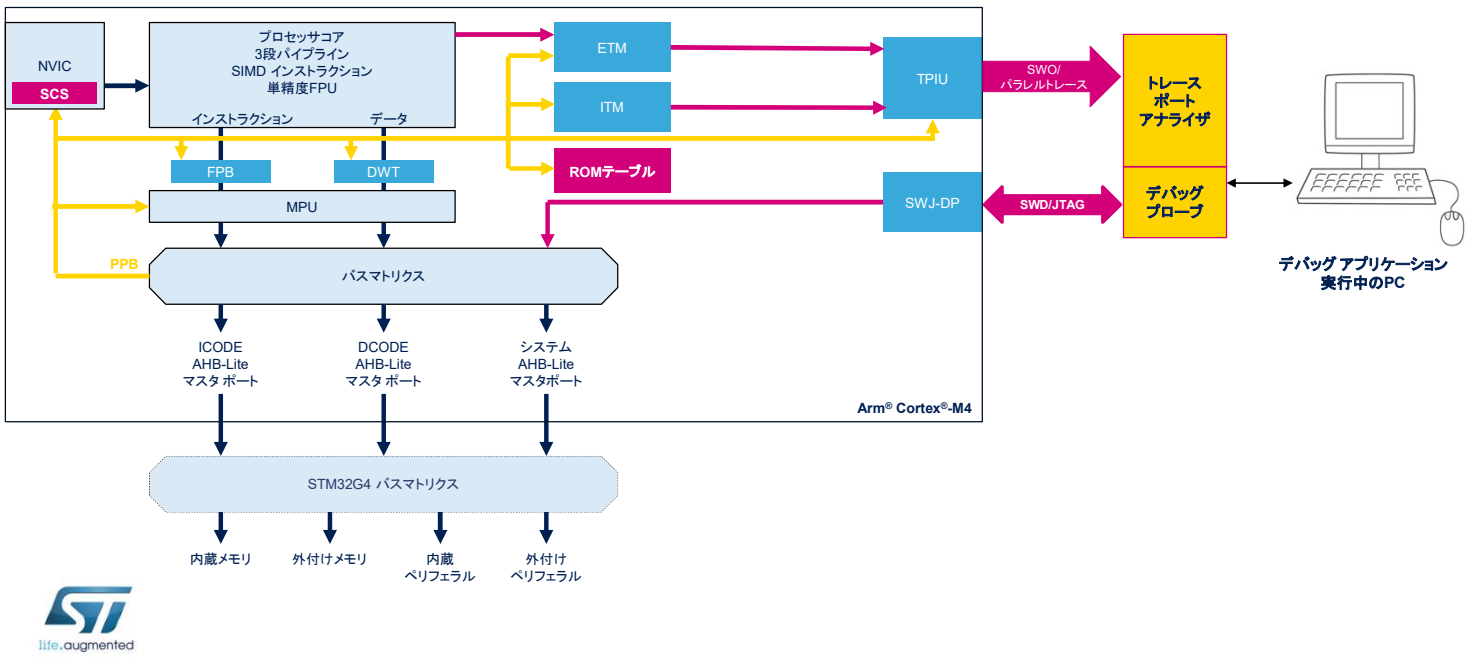
Pinout	PA13	PA14	PA15	PB3	PB4	PE2	PE3	PE4	PE5	PE6
SWD	SWDIO	SWCLK								
JTAG	JTMS	JTCK	JTDI	JTDO	JTRST					
SWO				TRACESWO						
トレース･ポート						TRACECK	TRACED0	TRACED1	TRACED2	TRACED2

- コントロールされていない I/Oのレベルを回避するため、デバイスは JTAG 入力ピンに内部プルアップとプルダウンを設定

SWJ-DPのSTM32G4からの出力として、5本のピンが汎用I/Oの代替機能として使用されます。
これらのピンは、すべてのパッケージで使用できます。
リセット解除後、SWJ-DP に使用される 5 つのピンはすべて、デバッガ ホストとして使用できる専用ピンとしてすぐに割り当てられます。
ただし、STM32G4 MCU は、SWJ-DP ポートの一部またはすべてを無効にする可能性があるため、接続を切断したままにできるが、デバッガ接続を失うことなく汎用 GPIO として使用できない NJTRST を除き、汎用 IO(GPIO)の使用のために関連ピンを解放する可能性があります。
注意：デバッガ ホストによって明示的にプログラムされている場合を除き、パラレルトレース ポートは割り当てられません。
JTAG ピンには内部プルアップとプルダウンがあり、デフォルトはアクティブです。

- プルアップ オン NJTRST、JTDI、JTMS/SWDIO
- プルダウン オン TCK/SWCLK

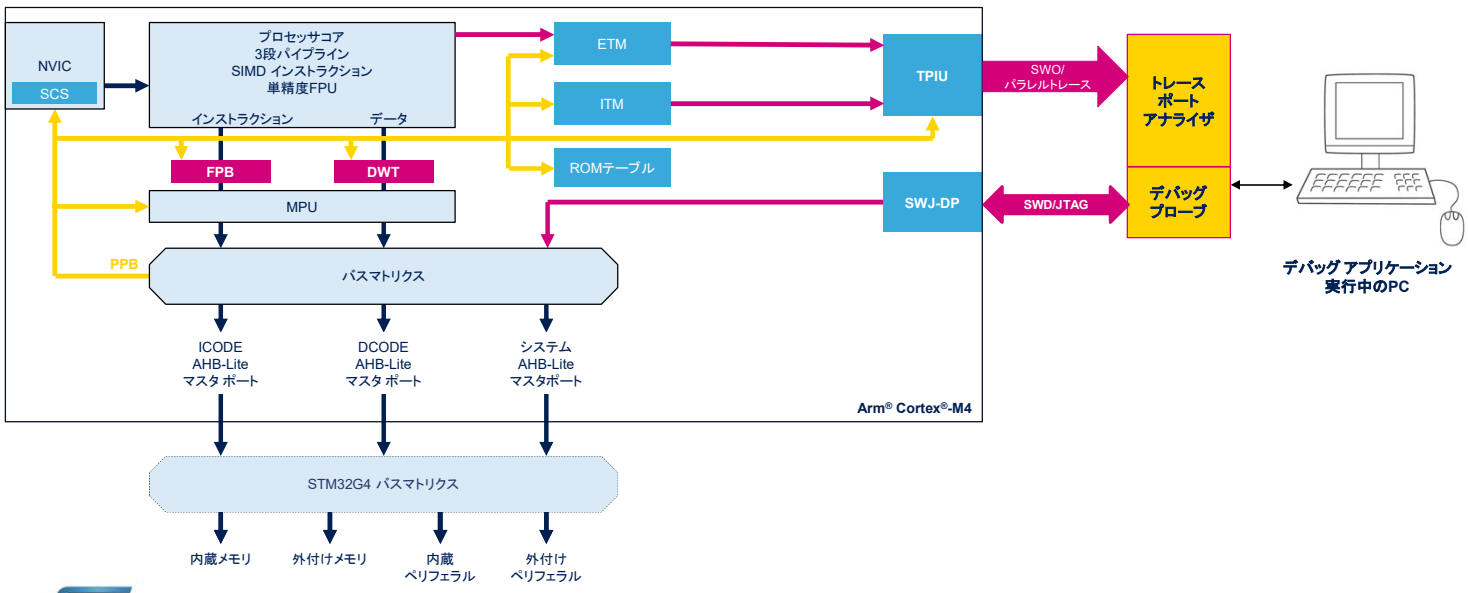
CoreSight™トポロジ検出



ROM テーブルには、コアから見ることが出来る各デバッグ コンポーネントのベース アドレスへのポインターが含まれています。一部のデバッグ ツールでは、ターゲットの CoreSight™ インフラストラクチャのトポロジを自動的に検出するために使用されます。SWJ-DP には ROM テーブルを指す読み取り専用レジスタが含まれています。

SCS (システム制御スペース) は、CPU の参照とリビジョンを示す CPUID レジスタを含むレジスタのブロックであり、デバッガが Halt モードへの入り口と終了を制御するために使用します。

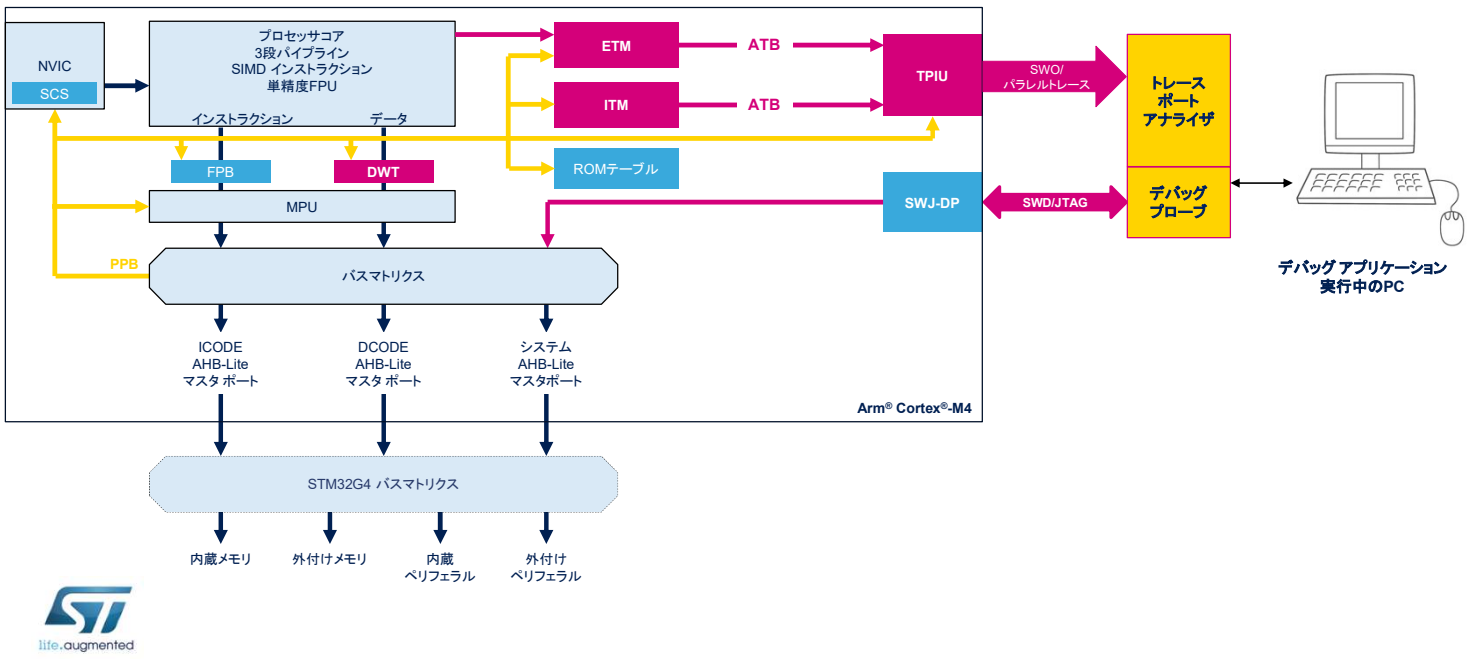
侵略性デバッグ



侵略性デバッグは、デバッグ イベントが発生したときにプロセッサを停止します。
2つのユニットは、侵襲性デバッグに関与しています:

- フラッシュパッチとブレイクポイント(FPB)
- データウォッチポイントとトレース(DWT)

非侵襲性デバッグ



非侵襲性デバッグは、プロセッサのパフォーマンスに影響を与えることなく、プロセッサで何が起こるかをトレースすることを目的としています。これはリアルタイムトレースに基づいています。
ETM および ITM モジュールは、トレース パケットを合成し、AMBA トレース バス(ATB)を介して発行する機能を備えています。
TPIU はトレース パケットを受信し、外部インタフェイスを介して出力するため、トレース ポート アナライザ (TPA) はパケットをキャプチャできます。
通常、トレース ポート アナライザはデバッグ プローブと同じデバイスに含まれています。
TPA にトレース パケットを転送するために 2 つのインタフェイスがサポートされています。

- 非同期式シングルワイヤ出力(SWO)
- または同期式パラレルトレースポート

TPIU で競合が検出されると、オーバーフロー状態が発生します。これはトレース ポートの帯域幅が、すべてのトレース パケットをエクスポートするのに十分な大きさではないことを意味します。
トレース パケットは ITM モジュールと ETM モジュールでタイムスタンプ付きであることに注意してください。

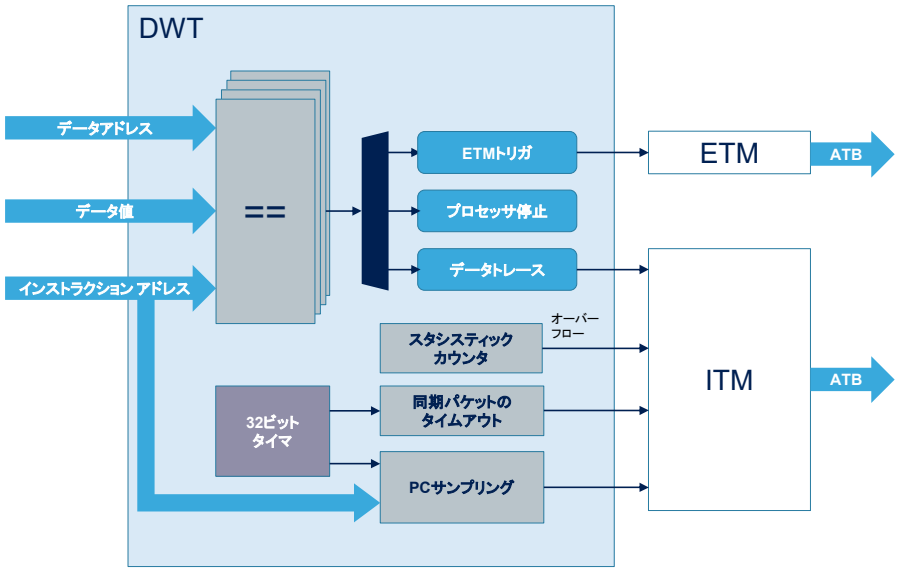
フラッシュ パッチとブレークポイント(FPB)

- FPB はハードウェア のブレークポイントを設定可能
  - これは、不揮発性メモリにブレークポイントを設定できる6つのコンパレータが含まれる
  - 2 つのコンパレータがデータ･アクセスをモニタリング
- FPB は、STM32G4 のFlashメモリの 0 ~ 0x1FFFFFFF の範囲内のアドレスコンペアのみをサポート
  - アドレス 0x20000000 以上では、コードは RAM から実行しDWT を使用してデータ･アクセスを停止する場合に、ソフトウェア･ブレークポイントが使用可能
- これらの 8 つのコンパレータのどれでも、CPU を停止する代わりにインストラクションの変更またはデータを出力するように設定可能

life.augmented

フラッシュ パッチとブレークポイント ユニット (FPB) では、ハードウェアのブレークポイントを設定できます。
これは、不揮発性メモリにブレークポイントを設定する6つのコンパレータが含まれています。一致が検出されると、命令フェッチ アドレスを監視し、ブレークポイント命令を返します。ブレークポイント命令が実行されると、プロセッサはデバッグ モードで停止します。
2 つのコンパレータは、不揮発性メモリに存在する特定の定数がアクセスされたときにプロセッサを停止するために使用されるデータ アクセスを監視します。
FPB は、STM32G4 のフラッシュ メモリに対応する 0 ~ 0x1FFFFFFF の範囲内のアドレスコンペアのみをサポートします。
アドレス 0x20000000 以上では、RAM からコードを実行し、DWT を使用してデータ アクセスを停止する場合に、ソフトウェア ブレークポイントを使用できます。
ソフトウェア ブレークポイントを使用すると、デバッガは、ユーザーが停止する命令を BKPT という専用の命令に置き換えます。
これらの 8 つのコンパレータのどれでも、CPU を停止する代わりにインストラクションの変更、または定数データを出力するように設定できます。

データ･ウォッチポイントとトレース･ユニット



- DWT は自律的にデータ･ウォッチポイントを設定、およびトリガするために使用可能
- ETMトリガ信号をアサートするようにコンパレータを設定できる
- DWTモジュールはITMと連携し、さまざまなリアルタイム･トレース機能をサポート:
  - データ･トレース
  - スタティスティック･カウンタ･オーバーフロー
  - 定期的な PCサンプリング

DWT は、単にデータウォッチポイントを実装するよりも多くの機能をサポートします。
図の左側にある 4 つのコンパレータは、データ アドレスまたは命令アドレスの一致条件を検出します。
コンパレータ1 は、データ値の一致条件を検出するように構成することもできます。
一致条件が発生した場合、次のアクションがサポートされます。

- コアを停止、これはウォッチポイント機能
- ETM へのトリガ信号のアサート
- データ アクセスを報告し、それを ITM に渡してエクスポートできるようにトレース パケットを生成します。トレース情報を補完するために、現在のデータと PCの 値をキャプチャすることもできます。

複雑な条件、混合アドレスとデータ比較はプログラム可能です。
比較ロジックに加えて、DWT には 8 ビットスタティスティックカウンタが含まれています。それらのいずれかにオーバーフロー条件が発生すると、DWT は、このオーバーフローをデバッガに報告するトレース パケットを発行するように ITM に通知します。
DWT は、トレース ポート アナライザとの同期を維持するために必須の同期パケットの定期的な生成をトリガも担っています。
これは 32 ビット タイマによって実現され、トレース パケットに格納するために ITM に渡される PCの値の定期的なサンプリングをトリガするためにも使用できます。

データ･ウォッチポイントとトレース･ユニット

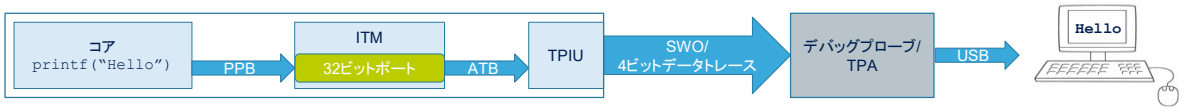
- ウォッチポイントの機能
  - DWT は命令アドレスとデータ･アドレスを比較できる 4 つのコンパレータを提供
    - 選択可能な方向(読取り、書込み、またはその両方)で特定のデータにアクセスしようとする試みを検出するのに非常に便利
    - 命令アドレスの比較では、ウォッチポイント･イベントはフェッチ時に取られる
- 統計カウンタ
  - DWTは、プロセッサのシステム･プロファイリングを提供
  - カウンタが含まれている:
    - クロックサイクル数、フォールドされた命令数、ロード･ストア･ユニット(LSU)の動作数、スリープ･サイクル数、命令当たりのサイクル数、例外マイクロコード実行数

life.augmented

DWT は、現在のデータまたは命令アドレスと、デバッガによってプログラムされたコンパレータの内容との一致によって発生するウォッチポイント イベントをトリガします。
アドレスの照合については、アドレスの範囲に照合出来るように、マスクを使用できます。
照合が成立すると、コンパレータは PC 値またはアクセスされたデータアドレスに対してウォッチポイントデバッグイベントを生成します。
データ値との一致は、アドレス一致と組み合わせることが出来ます。
FPB は、命令が実行ユニットに入ろうとしたときにブレークポイント イベントが発生するため、命令ブレークポイントの実装に適しています。
したがって、フェッチされた命令が、分岐によって破棄された場合、ウォッチポイント イベントが発生する間、ブレークポイント イベントは発生しません。
R0 などのコア レジスタを読み書きするには、デバッガはコアを停止する必要があります。
DWT には、プロセッサのシステム プロファイリングを容易にする統計カウンタが含まれています。
たとえば、カウンタは、例外の取得と終了を担当するマイクロコードがアクティブになっているクロック数をカウントします。

計装トレース･マクロセル (ITM)

- ITM トレース･パケットの 4 つのソースは:
  - ソフトウェア･トレース
    - 32 個のスティムラス･レジスタのいずれかへのソフトウェア書込み



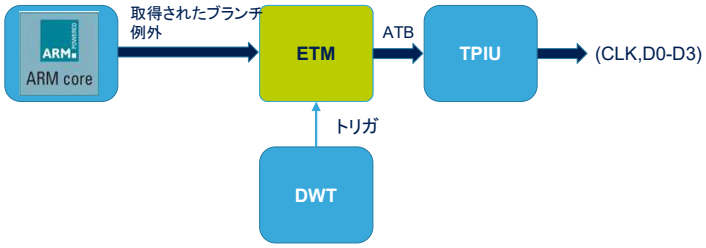
  - ハードウェア･トレース
    - DWT はパケットを生成し、ITM はパケットを出力
  - ローカル･タイムスタンプ
    - ITM トレース･パケットのタイミング情報を提供
  - グローバル･タイムスタンプ
    - タイムスタンプ生成コンポーネントから取得されるシステム全体の 64bitカウント値を使用して生成

2 種類のトレース情報が ITM によって処理されます。
- DWT モジュールで説明されているハードウェア･トレース
- ソフトウェア トレースは、デバッガー ウィンドウに表示されるデータをソフトウェアが送信できるようにします。

32 の ITM 32 ビット ポートのいずれかでソフトウェアによって書き込まれたデータは、トレース パケットのペイロードでホスト デバッガに転送されます。
printf 関数は、コードインストルメンテーションを容易にするために ITM ポートにリダイレクトできます。
ITM ユニットでは、2 つの連続したローカルタイムスタンプ パケットとグローバルタイムスタンプの相対時間を示すローカルタイムスタンプという 2 種類のタイムスタンプがサポートされています。
グローバルタイムスタンプは、ETMユニットでも使用される専用の48ビットから来ています。

組み込みトレース･マクロセル (ETM)

- ETMは、プロセッサの命令トレースを提供するリアルタイム･トレース･モジュール
  - ETM は、クロックと最大 4 つのデータ出力 TRACECLK、および 4 つのデータ出力 TRACEDATA(3:0) からなる TPIU で使用される



Index	Frame	Address	Opcode	Trace	
				}	
012960	000420	0x0803024C	0xbcf0	POP	{R4-R7}
012961	000420	0x0803024E	0x4770	BX	LR
				SysTick->LOAD=SysTick_LOAD_RELO...	
012962	000421	0x0802FFA8	0x487a	LDR.N	R0, ??DataTable1_10 ...
012963	000421	0x0802FFAA	0xf07f	MVNS.W	R1, #-16777216 ...
012964	000421	0x0802FFAE	0x6001	STR	R1, [R0]

SWO は十分な帯域幅を提供しないため、ETM は 4 ビット のデータ トレース ポートを必要とします。
ITM とは異なり、コードインストルメンテーションは必要ありません。
取得されたブランチと例外イベントは、ETM ユニットに対して Cortex®-M4 コアによって通知され、ATB ポートに出力されるトレース パケットに変換されます。
トレースがキャプチャされると、デバッガはトレースを解凍して、実行されたコードの完全な逆アセンブリ（シンボル付き）を提供します。デバッガはまた、これを元の高レベルソースコードにリンクして、ターゲットでコードがどのように実行されたかを視覚化します。
ETM は Cortex®-M4 によって実行されるコードをプロファイリングし、複雑なソフトウェアバグを調査するのに役立ちます。

DBGMCU

- 「MCUデバッグ」ブロックはデバイス固有のデバッグ機能を有効化
  - デバイスID
    - デバイスIDコード･レジスタを読み取るための標準の場所
  - 低電力モードのエミュレーション
    - デバイスが低電力モード（SLEEP、STOP、STANDBY）になったときに、デバッグ･アクセスが引き続 き可能となるように電力とクロックを維持
  - デバッグモード時のペリフェラルクロックの「停止」
    - プロセッサが停止している間は、RTC、TIM、LPTIM、およびウォッチドッグ（IWDG、WWDG）タイマ のカウンタ、ならびに SMBUS および FDCAN タイムアウト･カウンタを停止
  - トレース･ピンの割当ての制御

life.augmented

MCU デバッグ コンポーネント以下を提供します。

- MCU ID
- 低電力モード
- ブレークポイント時のタイマ、ウォッチドッグ、I2C、および bxCAN のクロック制御
- トレース ピンの割り当ての制御

DBG_IDCODEレジスタは、STM32 標準フォーマットのデバイス ID およびリビジョン コードが格納されています。この情報は、SWJ-DPまたはユーザーソフトウェアからアクセスできます。

低電力モードのエミュレーションは、低電力モードに入ってもデバッガとの接続が失われないことを意味します。この機能により、低電力に入力されるコマンド (WFI/WFE など) を while() ループに置き換える必要がなくなります。終了時に、デバイスはエミュレーションがアクティブでなかった場合と同じ状態になります (低電力モードエミュレーション中にデバッガーが行った変更を除く)。

ペリフェラルクロックのフリーズは、デバッガを用いてウォッチドッグを再設定をする必要が無く、デバッグ中にウォッチドッグ タイムアウトがデバイスをリセットするのを防ぐのに特に便利です。また、タイマ値の検査と、対応する割り込みを「通常の」操作が再開するまでサスペンドすることが可能です。

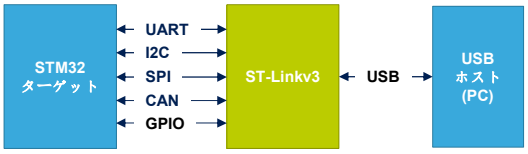
DBGMCU_CRレジスタには、トレースポートを構成する制御フィールド (非同期 SWO モードまたは TRACEDATA サイズが 1、2、または 4 の同期モード) を設定するフィールドが含まれています。

ST-LinkV3の主な機能

- パフォーマンス向上

	ST-Link V2 (KHz)	ST-Link V3 (KHz)
SWD	4000	24000
JTAG	9000	21333

- ブリッジ機能の概要



- STLINK-V3SETは複数のプロトコルのSTM32ターゲットとの通信を可能にする独自のUSBインタフェースを提供: SPI、I²C、CAN、UART、GPIO
- このインタフェースは、ターゲット·ブートローダと通信するために使用可能

life.augmented

ST-Link は ST によって開発されたデバッガで、SW および JTAG デバッグ プロトコルと SWO トレース ポートをサポートします。
ST-Linkの第3世代は、SWDおよびJTAGの早い通信速度をサポートしています。
ST-LinkV3 は、UART、I2C、SPI、CAN、または GPIO を使用してターゲット上のデータを送受信するために使用できます。同時にアクティブにできるのは、これらのインタフェイスの 1 つだけです。
たとえば、STM32G4のUARTは、ST-LinkV3経由でホストPCに接続しテスト可能です。キャラクタストリームは、USB 経由で透過的に転送されます。
さらに、ST-LinkV3 では、ホスト PC からの UART または CAN リンクを使用してユーザ イメージのダウンロードがサポートされ、STM32G4 のリセットデアサーションでブーストラップ モードが選択されます。

STM32CubeProgrammer

- STM32CubeProgrammerは、既存のプログラミング･ソフトウェア･ツールを1つのソリューションに統合します
- コマンドラインおよびグラフィカル･ユーザ･インタフェースで利用可能
- 主な特長
  - マルチプラットフォーム (Windows®, Linux®, macOS®)
  - デバッグ･インタフェースおよびブートローダ･インタフェースのサポート
    - JTAG/SWD (ST-LinkV2、V3)
    - ブートローダ･インタフェース(UART、USB デバイス･ファームウェア･アップグレード･クラス)
    - ST-LinkV3ブリッジ
  - STM32 内蔵 / 外付けFlashのイレーズ、プログラム(セキュア、またはノン･セキュア)
  - オプション･バイト･プログラミング

life.augmented

STM32CubeProgrammerはSTM32のマイクロコントローラをプログラミングするためのオールインワンマルチOSソフトウェアツールです。
STM32CubeProgrammerはGUI(グラフィカルユーザインタフェース)およびCLI(コマンドラインインタフェース)バージョンで提供されます。
デバッグインタフェイス (JTAG と SWD) とブートローダ インタフェイス (UART と USB) の両方を使用して、デバイス メモリの読み取り、書き込み、およびベリファイを行うための、簡単で効率的な環境を提供します。
追加された機能は、:

- マルチOSサポート: Windows®, Linux®, macOS®
- デバッグインタフェイスおよびブートローダ インタフェイスのサポート
- ST-LinkV3 ブリッジ

STM32CubeProgrammerは、USBデバイスにファームウェアイメージをダウンロードするように設計されたデバイスファームウェアアップグレード(DFU)USBクラスをサポートしています。