

STM32H7 - DBG

デバッグとトレース
1.0版



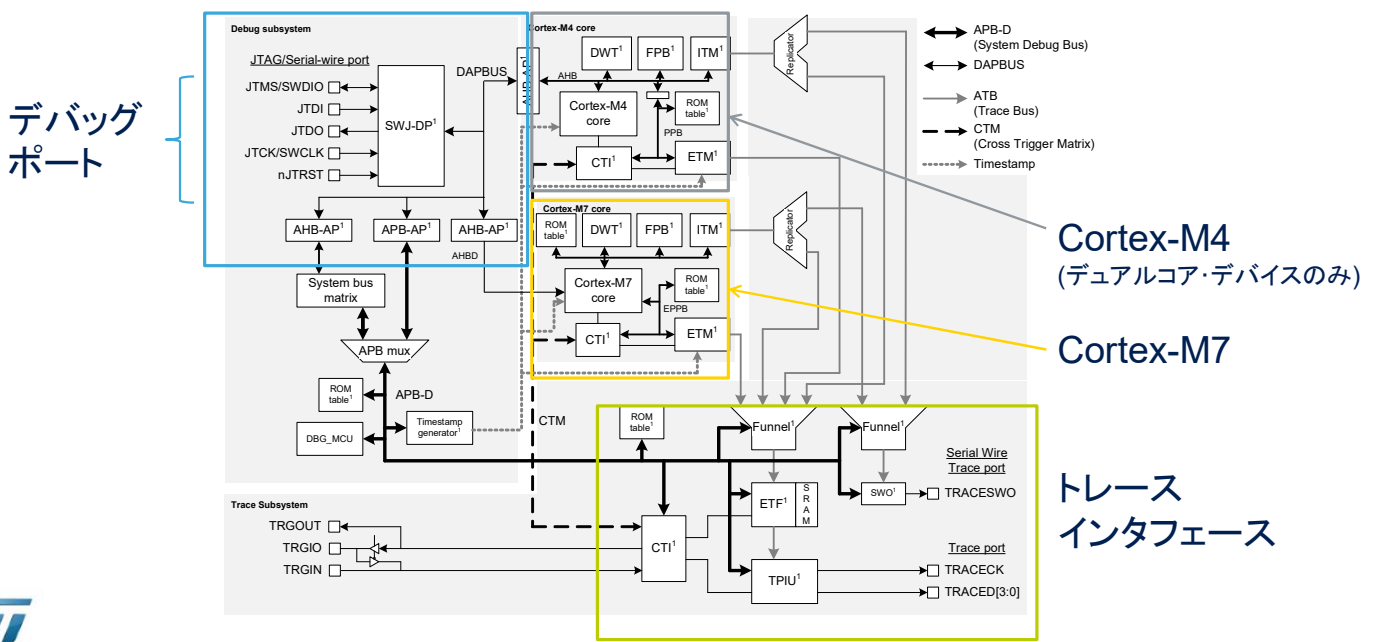
STM32デバッグとトレース用インタフェースのプレゼンテーション
によろこそ。STM32H7デバイスが備えているデバッグとトレース
機能について説明を行います。



- STM32H7はデバッグとトレースに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
 - プログラムを実行またはシングルステップ実行
 - プログラムの実行をトレース
- Arm® CoreSight™アーキテクチャベース
 - 広範囲な互換ツール
 - 標準インターフェース(JTAG/シリアルワイヤ)

STM32H7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、シリアルワイヤトレース)がすべて内蔵されていますが、さらに、マルチコア・バージョンのSTM32H7ファミリでは、高帯域幅の命令トレースやクロス・トリガ機能が追加されています。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



デバッグとトレースのインフラストラクチャは、次の4つの異なる機能ドメインで構成されています。

- ・ デバッグアクセスインフラストラクチャ - デバッグポート (SWJ-DP) およびアクセスポート (AP) を含み、外部のデバッガがターゲットのトレースおよびデバッグ機能にアクセスできるようにします。
- ・ トレースインフラストラクチャ: シリアル (SWO) およびパラレル (TPIU) のトレースポート、トレースフローを平滑化するためのトレースFIFO (ETF)、各ソースからのトレースを1つのフローにまとめるためのトレースファンネルなどがあります。
- ・ Cortex-M7コア-プロセッサと関連するトレースおよびデバッグユニット (DWT、FPB、ITM、およびETM) を含みます。
- ・ Cortex-M4コア (デュアルコアデバイスのみ)

さらに、次のものを含むシステムデバッグ機能もあります。

- ・ クロストリガインターフェイスとマトリックス (CTI、CTM) - これらは、両方のコアの同時停止、トレースのトリガなどを可能にします。
- ・ グローバル・タイムスタンプ・ジェネレーター: 異なるトレース・ソースに共通の時間基準を提供
- ・ DBG_MCU - デバッグ中のタイマのフリーズのような独自機能を提供します。
- ・ 外部トリガ入出力 - 外部信号によるデバッグまたはトレースのトリガを可能としたり、外部の装置やコンポーネントを同期するためのトリガパルスを生成したりします。

- デバッガはJTAG/SWDデバッグ・ポート経由でSTM32H7にアクセス
 - 標準の5ピンJTAGポートもバウンダリスキャンとDFTに使用
 - シリアル・ワイヤ・デバッグ (SWD) ポートはJTAGポートのうち2ピンだけ使用
 - 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

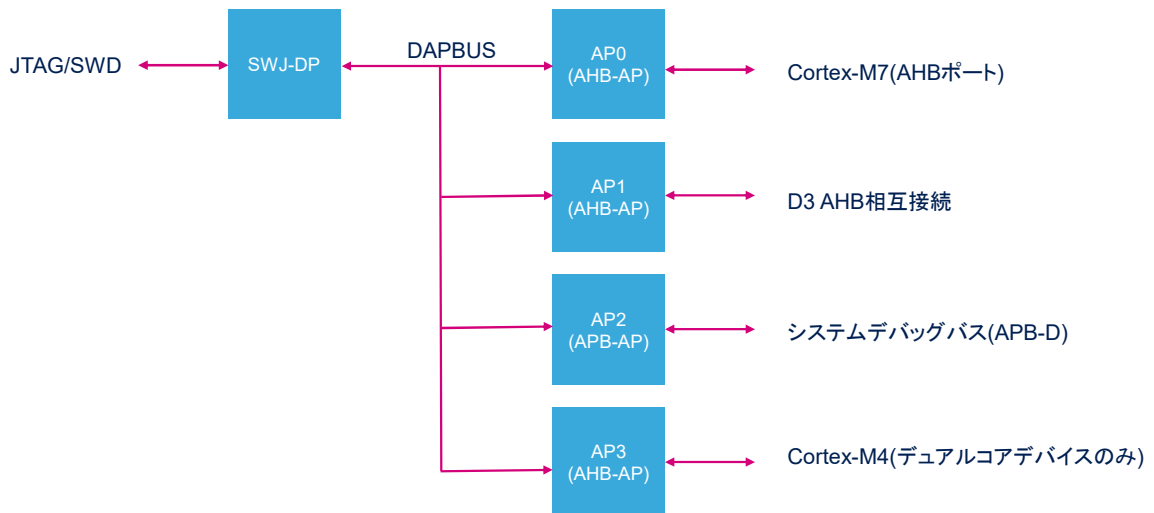
使用可能なデバッグポート	PA13	PA14	PA15	PB3	PB4
JTAG/SWD*すべて	JTMS	JTCK	JTDI	JTDO	NJTRST
nJTRSTを除くJTAG/SWDすべて	JTMS	JTCK	JTDI	JTDO	
JTAG-DP無効、SW-DP有効	SWDIO	SWCLK			
JTAG-DPとSW-DPがともに無効					

*リセット状態



デバッグ用の最小構成には、ピンPA13とPA14をシリアルワイヤデバッグ (SWDIOとSWCLK) に割り当てる必要があります。シリアルワイヤデバッグには、SWDIO (JTMS) 入力に接続されたデバッガにより入力される特殊シリアルコードを使用します。このコードは、SWDモード (リセット後にデフォルトでJTAGモードに設定) に切り換わってSWJ-DPによって認識されます。ST-Linkならびに大半のサードパーティーデバッグアダプタ (例えばUlinkなど) は、シリアルワイヤデバッグに対応しています。

- 4つのアクセス・ポート(AP)がバス・マスタとして機能し、メモリとレジスタの読出し／書込みトランザクションをデバッガが実行可能



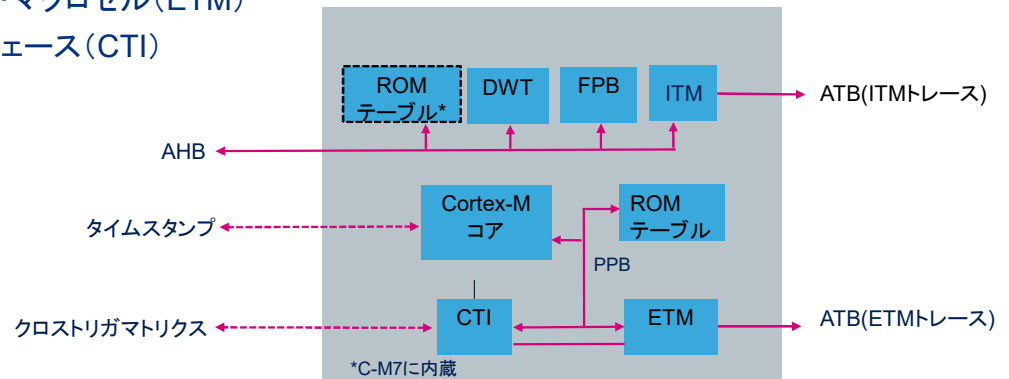
AP0: Cortex-M7プロセッサコアに搭載されているデバッグ機能やトレース機能に、プロセッサのAHBDポートに接続されたAHB-Liteバスを介してアクセスすることができます。

AP1: D3ドメインのAHBバスマトリックスへのアクセスを可能にします。これにより、D1およびD2ドメインがオフの場合でも、D3ドメインのメモリやペリフェラルを見ることができます。

AP2: システムAPBデバッグバス上のデバッグおよびトレース機能へのアクセスを許可します。すなわち、いずれかのプロセッサコアに含まれないすべてのコンポーネントが対象となります。

AP3: (デュアルコアデバイスのみ) Cortex-M4プロセッサコアに統合されたデバッグおよびトレース機能に、その内部AHBバスを介してアクセスできます。

- Cortex-Mコアには以下のデバッグ・コンポーネントが含まれる
 - システム制御空間(SCS)
 - データウォッチ・ポイントおよびトレース・ユニット(DWT)
 - ブレークポイント・ユニット(FPB)
 - 計装トレース・マクロセル(ITM)
 - エンベデッド・トレース・マクロセル(ETM)
 - クロス・トリガ・インタフェース(CTI)
 - ROMテーブル



ETM

Cortex-M7コアのデバッグ関連レジスタには、専用AHBアクセスポートAP0経由でアクセスします。

ROMテーブルには、APから見え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が含まれています。これらは、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検出するデバッグツールによって使用されます。

SCS(システム制御空間)には、デバッグモードでプロセッサコアを制御するレジスタが含まれています。

その他のユニットについては、以降のスライドで説明します。

データウォッチポイントおよびトレースユニット

7

- DWTには4つのコンパレータがあり、それぞれ以下の役割に使用可能
 - ウォッチポイント
 - ETMトリガ
 - PCサンプリング・トリガ
 - データアドレス・サンプリング・トリガ
 - データ・コンパレータ
 - クロック・サイクル・カウンタ・コンパレータ
- ソフトウェア・プロファイリング用のカウンタも内蔵しています。
 - 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
 - 割込みオーバーヘッドの回数



データウォッチポイント(DWT)コンパレータは、以下の項目のうちの1つを、DWT_COMPレジスタに保持されている値と比較します。

- データアドレス
- 命令アドレス
- データ値
- サイクルカウント値(コンパレータ0のみ)

アドレス照合の場合、コンパレータはマスクを使用することができるため、アドレスの範囲を照合することができます。

照合が成立すると、コンパレータは以下のうちの1つを生成します。

- 1つ以上のDWTデータトレースパケットで、以下を1つ以上含むもの。
 - データアクセスを伴う命令のアドレス
 - アドレスオフセット(データアクセスアドレスのビット[15:0])
 - 一致したデータ値。
- PC値またはアクセスしたデータアドレスのいずれかで発生するウォッチポイントデバッグイベント。
- DWTユニット外での一致を信号で伝えるCMPMATCH[N]イベント。

- FPBを使用することでハードウェアのブレークポイントを設定することが可能
 - このクラスには、命令フェッチアドレスを監視し、一致が検出されたときにブレークポイント命令を返す8つのコンパレータが含まれる
 - 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



life.augmented

デュアルコアデバイスでは、Cortex-M4のブレークポイントユニット(FPB)は、Flashメモリのパッチもサポートしています。この機能は、指定されたアドレスの揮発性メモリに実行を移すことで、誤ったコードをパッチすることを目的としています。

Cortex-M7では、FPBはフラッシュメモリのパッチングをサポートしていません。

- ITMは次の4種類のソースからトレースパケットを生成
 - ソフトウェアトレース:
ソフトウェアは、32本のスティムラス・レジスタのいずれかに書込みを行う
 - DWTからのハードウェア・トレース・パケット
これはデータトレース・イベント、PCサンプル、カウンタの折り返しの可能性がある
 - ローカル・タイムスタンプ
ITMの21ビット・カウンタが、前のパケットと比較したタイムスタンプを各トレースパケットに供給
 - グローバル・タイムスタンプ
タイムスタンプは、TSGENのシステム全体の64ビット・タイムスタンプを使って生成することも可能
- トレースパケットをATBのトレースバスで出力



ソフトウェアは、32 x 32ビットの軽装トレースマクロセル(ITM)スティムラスレジスタのいずれかに直接書込みを行って、パケットを生成します。各ポートの許可レベルはプログラムで設定できます。ソフトウェアが有効なスティムラスポートに書き込むと、ITMはFIFOに書き込むパケットの中に、ポートのID、書込みアクセスのサイズ、および書き込まれたデータを統合します。ITMはFIFOからトレースバスにパケットを出力します。スティムラスポートレジスタを読み出すと、ビット0のスティムラスレジスタのステータス(エンptyまたはペンディング)を返します。

パケットが複数のソースから同時に生成される場合、ITMはパケットの出力順番についてアービトレーションを行います。このシートには、優先度の高いものから順にソースがリストアップされています。

タイムスタンプジェネレータ(TSGEN)は、すべてのトレースパケットのタイムスタンプに64ビットの共通のタイムベースを提供します。これにより、異なるソースから送られてきたトレースを、トレースが生成された時刻に応じて揃えることができます。ローカルのタイムスタンプは同期しておらず、異なる周波数で動作しているため、トレースが生成された正確なタイミングを知ることはできません。

注: Cortex-M4は、グローバルタイムスタンプの48LSBのみを使用します。Cortex-M7では、64ビットすべてを使用します。

- ETMはソフトウェアの実行を観察できるトレースパケットを生成
トレース情報には次の情報が含まれる
 - 同一サイクルで実行される命令の数
 - プログラムフローの変更
 - プロセッサの現在の命令ステート
 - ロードおよびストア命令によってアクセスされるメモリ位置のアドレス
 - 転送のタイプ、方向、およびサイズ
 - 条件コードに関する情報
 - 例外情報
 - 割り込み待ちステートに関する情報
- トレースパケットをATBのトレースバスに出カ

Time	Address / PC	Instruction / Data	Src Code / Trigger Addr	Function
75.600 827 180	0x40000738	LDRH #0x00000000		main
75.600 827 181	0x40000738	CEZ #0x40000738		main
75.600 827 182	0x40000738	MOVW #0x00000000		main
75.600 827 183	0x40000738	LDH #0x40000738		main
75.600 827 184	0x40000738	LDH #0x40000738		main
75.600 827 185	0x40000738	LDH #0x40000738		main
75.600 827 186	0x40000738	LDH #0x40000738		main
75.600 827 187	0x40000738	LDH #0x40000738		main
75.600 827 188	0x40000738	LDH #0x40000738		main
75.600 827 189	0x40000738	LDH #0x40000738		main
75.600 827 190	0x40000738	LDH #0x40000738		main
75.600 827 191	0x40000738	LDH #0x40000738		main
75.600 827 192	0x40000738	LDH #0x40000738		main
75.600 827 193	0x40000738	LDH #0x40000738		main
75.600 827 194	0x40000738	LDH #0x40000738		main
75.600 827 195	0x40000738	LDH #0x40000738		main
75.600 827 196	0x40000738	LDH #0x40000738		main
75.600 827 197	0x40000738	LDH #0x40000738		main
75.600 827 198	0x40000738	LDH #0x40000738		main
75.600 827 199	0x40000738	LDH #0x40000738		main
75.600 827 200	0x40000738	LDH #0x40000738		main
75.600 827 201	0x40000738	LDH #0x40000738		main
75.600 827 202	0x40000738	LDH #0x40000738		main
75.600 827 203	0x40000738	LDH #0x40000738		main
75.600 827 204	0x40000738	LDH #0x40000738		main
75.600 827 205	0x40000738	LDH #0x40000738		main
75.600 827 206	0x40000738	LDH #0x40000738		main
75.600 827 207	0x40000738	LDH #0x40000738		main
75.600 827 208	0x40000738	LDH #0x40000738		main
75.600 827 209	0x40000738	LDH #0x40000738		main
75.600 827 210	0x40000738	LDH #0x40000738		main
75.600 827 211	0x40000738	LDH #0x40000738		main
75.600 827 212	0x40000738	LDH #0x40000738		main
75.600 827 213	0x40000738	LDH #0x40000738		main
75.600 827 214	0x40000738	LDH #0x40000738		main
75.600 827 215	0x40000738	LDH #0x40000738		main
75.600 827 216	0x40000738	LDH #0x40000738		main
75.600 827 217	0x40000738	LDH #0x40000738		main
75.600 827 218	0x40000738	LDH #0x40000738		main
75.600 827 219	0x40000738	LDH #0x40000738		main
75.600 827 220	0x40000738	LDH #0x40000738		main
75.600 827 221	0x40000738	LDH #0x40000738		main
75.600 827 222	0x40000738	LDH #0x40000738		main
75.600 827 223	0x40000738	LDH #0x40000738		main
75.600 827 224	0x40000738	LDH #0x40000738		main
75.600 827 225	0x40000738	LDH #0x40000738		main
75.600 827 226	0x40000738	LDH #0x40000738		main
75.600 827 227	0x40000738	LDH #0x40000738		main
75.600 827 228	0x40000738	LDH #0x40000738		main
75.600 827 229	0x40000738	LDH #0x40000738		main
75.600 827 230	0x40000738	LDH #0x40000738		main
75.600 827 231	0x40000738	LDH #0x40000738		main
75.600 827 232	0x40000738	LDH #0x40000738		main
75.600 827 233	0x40000738	LDH #0x40000738		main
75.600 827 234	0x40000738	LDH #0x40000738		main
75.600 827 235	0x40000738	LDH #0x40000738		main
75.600 827 236	0x40000738	LDH #0x40000738		main
75.600 827 237	0x40000738	LDH #0x40000738		main
75.600 827 238	0x40000738	LDH #0x40000738		main
75.600 827 239	0x40000738	LDH #0x40000738		main
75.600 827 240	0x40000738	LDH #0x40000738		main
75.600 827 241	0x40000738	LDH #0x40000738		main
75.600 827 242	0x40000738	LDH #0x40000738		main
75.600 827 243	0x40000738	LDH #0x40000738		main
75.600 827 244	0x40000738	LDH #0x40000738		main
75.600 827 245	0x40000738	LDH #0x40000738		main
75.600 827 246	0x40000738	LDH #0x40000738		main
75.600 827 247	0x40000738	LDH #0x40000738		main
75.600 827 248	0x40000738	LDH #0x40000738		main
75.600 827 249	0x40000738	LDH #0x40000738		main
75.600 827 250	0x40000738	LDH #0x40000738		main
75.600 827 251	0x40000738	LDH #0x40000738		main
75.600 827 252	0x40000738	LDH #0x40000738		main
75.600 827 253	0x40000738	LDH #0x40000738		main
75.600 827 254	0x40000738	LDH #0x40000738		main
75.600 827 255	0x40000738	LDH #0x40000738		main
75.600 827 256	0x40000738	LDH #0x40000738		main
75.600 827 257	0x40000738	LDH #0x40000738		main
75.600 827 258	0x40000738	LDH #0x40000738		main
75.600 827 259	0x40000738	LDH #0x40000738		main
75.600 827 260	0x40000738	LDH #0x40000738		main
75.600 827 261	0x40000738	LDH #0x40000738		main
75.600 827 262	0x40000738	LDH #0x40000738		main
75.600 827 263	0x40000738	LDH #0x40000738		main
75.600 827 264	0x40000738	LDH #0x40000738		main
75.600 827 265	0x40000738	LDH #0x40000738		main
75.600 827 266	0x40000738	LDH #0x40000738		main
75.600 827 267	0x40000738	LDH #0x40000738		main
75.600 827 268	0x40000738	LDH #0x40000738		main
75.600 827 269	0x40000738	LDH #0x40000738		main
75.600 827 270	0x40000738	LDH #0x40000738		main
75.600 827 271	0x40000738	LDH #0x40000738		main
75.600 827 272	0x40000738	LDH #0x40000738		main
75.600 827 273	0x40000738	LDH #0x40000738		main
75.600 827 274	0x40000738	LDH #0x40000738		main
75.600 827 275	0x40000738	LDH #0x40000738		main
75.600 827 276	0x40000738	LDH #0x40000738		main
75.600 827 277	0x40000738	LDH #0x40000738		main
75.600 827 278	0x40000738	LDH #0x40000738		main
75.600 827 279	0x40000738	LDH #0x40000738		main
75.600 827 280	0x40000738	LDH #0x40000738		main
75.600 827 281	0x40000738	LDH #0x40000738		main
75.600 827 282	0x40000738	LDH #0x40000738		main
75.600 827 283	0x40000738	LDH #0x40000738		main
75.600 827 284	0x40000738	LDH #0x40000738		main
75.600 827 285	0x40000738	LDH #0x40000738		main
75.600 827 286	0x40000738	LDH #0x40000738		main
75.600 827 287	0x40000738	LDH #0x40000738		main
75.600 827 288	0x40000738	LDH #0x40000738		main
75.600 827 289	0x40000738	LDH #0x40000738		main
75.600 827 290	0x40000738	LDH #0x40000738		main
75.600 827 291	0x40000738	LDH #0x40000738		main
75.600 827 292	0x40000738	LDH #0x40000738		main
75.600 827 293	0x40000738	LDH #0x40000738		main
75.600 827 294	0x40000738	LDH #0x40000738		main
75.600 827 295	0x40000738	LDH #0x40000738		main
75.600 827 296	0x40000738	LDH #0x40000738		main
75.600 827 297	0x40000738	LDH #0x40000738		main
75.600 827 298	0x40000738	LDH #0x40000738		main
75.600 827 299	0x40000738	LDH #0x40000738		main
75.600 827 300	0x40000738	LDH #0x40000738		main

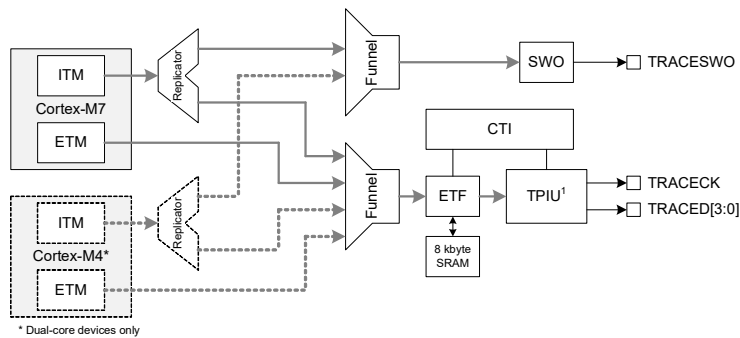
STM32H7では、組み込みトレースマクロセル(ETM)は命令トレース専用設定されているため、トレース情報にデータアクセスは含まれません。

Cortex-M7のETMは、Cortex-M3/M4のETMとは全く異なります。

トレース・インフラ・ストラクチャ

11

- ITMおよびETMからのトレース情報は、AMBAトレースバス(ATB)を經由して、1つまたは複数のトレースシンクに送られる
- ルーティングは、2つのATBインフラ・ストラクチャ・コンポーネントによって実行
 - レプリケータ: 2つのATBブランチでトレースパケットを複製
 - ファネル: 複数のATBブランチを1つにまとめたもの



ITMとETMは共にトレースストリームを生成し、トレースファネルを用いて結合します。例えば、ある入力から別の入力に切り替わるまでに受信するバイト数を変更することができます。また、トレースをフィルタリングすることも可能です。例えば、ITMトレースをTPIUから削除することができます(代わりにSWOに出力することができます)。

ETMではなくITMからのトレースをシングルワイヤートレースポートに導くことができます。デュアルコアデバイスでは、両方のコアからのITMトレースをSWOに導き、SWOのトレースファネルで結合することができます。しかし、SWOにはフォーマットがないため、トレースポートアナライザでトレースストリームを分離することはできません。そのため、ファネルを使って一度に1つのITMを手動で選択し、SWOに出力することをお勧めします。両方が必要な場合はTPIUを使用してください。

• **トレースパケットは次の3種類の転送先(シンク)の1つに送られる**

- **組み込みトレースFIFO(ETF)**
 - トレースパケットをサーキュラバッファに格納できる
8KByteのメモリ
トレースは、ソフトウェアやデバッガで読み出すことが可能
- **トレースポート・インタフェース(TPIU)**
 - トレースパケットは、同期クロック信号とともに、4ピン
パラレル・ポートを通じてデバイスから出力される
これには、ULINKproやDStreamなどのトレースポート
アナライザ・プローブを接続する必要がある
- **シングルワイヤ・トレースポート(SWO)**
 - ITMトレースは、SWOに導くことが可能であり、非同期
プロトコル(NRZまたはマンチェスタ)を用いて出力される
この出力は、ST-Linkその他のアダプタと多くの市販デバッガ
ツールを用いて読み込み可能



ETFは、オンチップでトレースを保存するためのトレースバッファとして使用できます。トレースは、ソフトウェアやデバッガで読み取ることができ、トレースポートを介して削除することもできます。サーキュラバッファとして構成されている場合、トレースは継続的に保存されるため、最新のトレースが最も古いトレースを上書きします。また、FIFOフルフラグを使用すると、バッファが一杯になった時点でトレースを停止し、特定の時点でのトレースをキャプチャすることができます。

ETFは、(ハードウェアモードでは)TPIUへのトレースの流れをスムーズにする役割も果たします。トレースストリームはバースト的な性質を持ち、瞬間的な帯域幅はトレースポートの帯域幅よりもはるかに高いため、バッファはピークを吸収し、トレースポートの最大連続帯域幅にフローを調整します。

- TPIUパラレル・ポート
 - 1~4本のデータ・ピンとクロックをトレースに割当て可能(デフォルトはGPIO)
 - TRACECLKは最高133MHzのDDRモードで動作可能
 - ✓ 最大帯域幅1Gbps(ULINKproでは800Mbps)
- SWOシングル・ワイヤ・シリアル・ポート
 - 1本の非同期データピンがJTDOとマルチプレクス
 - 帯域幅100Mbps(マンチェスタ符号化)



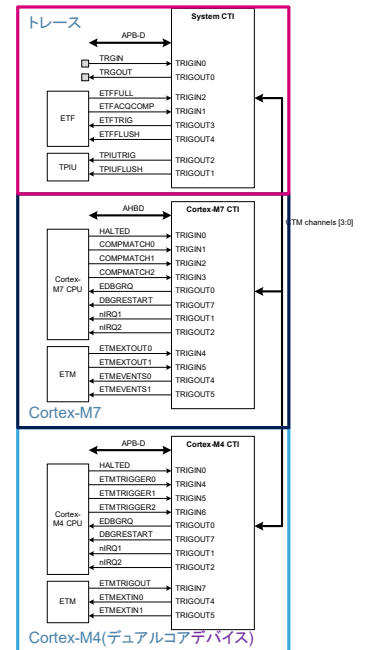
トレースポートの幅は1~4ピンにプログラム可能です。帯域幅は、ピン数とTRACECLK周波数(RCCの分周器で選択可能)に比例して変化します。

最大クロック周波数でのフルデュアルコア命令トレースは、多くの場合 最大のバンド幅を必要とします。

トレースソース(特にETM)にフィルタとトリガを適用することにより、トレースデータの平均量を減らすことが可能となり、クロックレートの低減やピン数の削減が可能です。

TRACESWOピンには、JTAGインタフェースの一部であるJTDO信号がマルチプレクスされています。したがって、シングルワイヤトレースは、シリアルワイヤデバッグ(SWD)インタフェースが有効化されている場合にしか使用できません。

- CTIは、その他のデバッグ・コンポーネントと
トレース・コンポーネントに対するトリガ・イベントを伝播
- トリガ・イベント・ソースの例：
 - データ・ウォッチポイント
 - ハードウェア・ブレークポイント
 - プロファイリング・カウンタ・イベント
 - トレースバッファのフル／エンプティ
 - 外部トリガ信号
 - プロセッサの停止／再起動
- トリガ・イベントにより転送先で発生する事象：
 - トレースの開始／停止
 - トレースバッファのフラッシュ
 - プロセッサの停止／再起動
 - 外部トリガ信号出力
 - プロセッサ割込み



クロストリガは、デュアルコアデバイスで使用して両方のコアを同時に停止させることができます。片方のコアがブレークポイントをヒットすると、その「停止状態」出力(デバッグモードに入ったことを示します)がもう一方のコアに伝搬して、そちらもデバッグモードに入ります。同様に、両方のコアが同時に再起動することもできます。

クロストリガ機能は、外部トリガ信号 (IOピン1本のエッジなど) でプロセッサを停止させるためにも使用できます。

各Cortex-Mプロセッサ専用のCTI(クロストリガインタフェース)と、トレースコンポーネント(ETF、TPIU)および外部トリガ信号に接続されたシステムCTIがあります。

いずれかのクロストリガ機能を使用するには、デバッガによりCTIをしかるべくプログラムする必要があります。必要なトリガ入力信号(TRIGINn)とトリガ出力信号(TRIGOUTn)は、クロストリガマトリックス(CTM)に接続する必要があります。CTMは最大4チャンネルから構成されており、4つの異なるイベントを並列に伝搬可能です。トリガ入力はCTIに束ねることが可能ですので、束ねられた入力のどの1本によっても接続されたチャンネルにイベントが発生します。同様に、1つのチャンネルを複数のトリガ出力に接続できますので、1つのイベントで複数の動作をトリガ可能です。

• 「MCUデバッグ」ブロックによりデバイス専用のデバッグ機能が実現

- デバイスID
 - 標準的に配置されており、デバイスIDコードレジスタを読み出す
- 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)に入った場合にもデバッグアクセスが可能であるように、電源とクロックを維持
- デバッグモードにおけるペリフェラル・クロックの「フリーズ」
 - プロセッサが停止している間に、RTC、TIM、LPTIM、ウォッチドッグ (IWDG、WWDG) タイマカウンタ、SMBUSやFDCANタイムアウト・カウンタをフリーズ
- ドメイン・デバッグ・クロックのイネーブル
 - 各パワードメイン内のデバイスをデバッグするためのクロックを、必要のないときは無効にし、電力を節約
- 外部トリガ方向
 - 双方向TRGIO外部トリガピンの方向(入力/出力)を制御



DBGMCUは、デバッグ用APBバス上にあり、APBアクセスポートAP2を介してデバッガからアクセスできます。また、デバッグ用APBアドレス空間のプロセッサからもアクセス可能です。DBGMCU_IDC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を提供します。この情報は、デバッグポート(DP_TARGETIDレジスタ-外部デバッガのみアクセス可能)およびシステムデバッグROMテーブルレジスタ(SYSROM_PIDR[2:0]-ソフトウェアでもアクセス可能)でも利用可能です。

低電力エミュレーションとは、低電力モードに入ってもデバッガとの接続が切れないことを意味します。これにより、低電力に入るコマンド(例えばWFI/WFE)をwhile()ループで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有効ではなかった場合と同じ状態となります(低電力モードエミュレーション中にデバッガにより行われた変更を除く)。

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する必要なく、デバッグ中にウォッチドッグタイムアウトによってデバイスがリセットするのを防ぐために特に便利です。また、それによってタイマ値の検査と、「通常」動作が再開されるまで対応する割り込みのサスペンドが可能です。デバッグクロックイネーブルビットにより、デバッグブロックは必要なときだけクロックが供給されます。これにより、DAP以外のすべてのブロックは、ゲートされていないドメインクロックでクロック供給されるため、不要な電力消費を避けることができます。

特定のパッケージでは、TRGINピンとTRGOUTピンが存在せずに双方向ピンのみが使用され、その方向はTRGOENビットを用いて機能選択する必要があります。