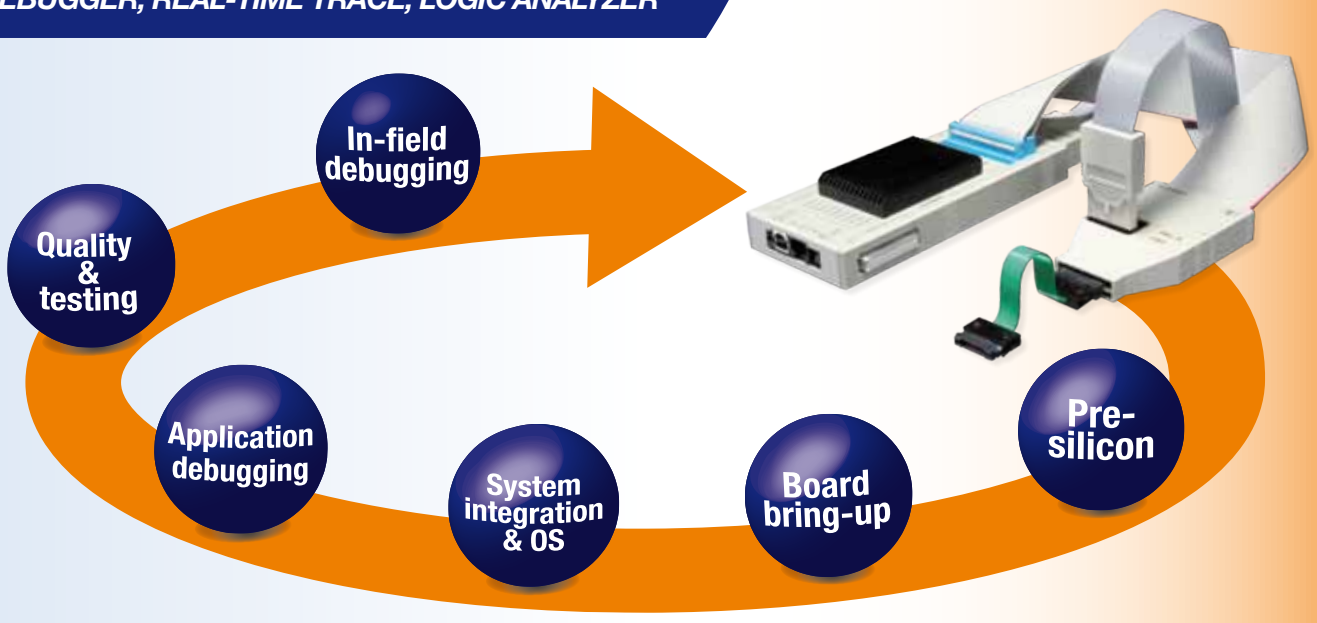
# NEWS 2012

LAUTERBACH
DEVELOPMENT TOOLS

DEBUGGER, REAL-TIME TRACE, LOGIC ANALYZER




## プロジェクトのあらゆるフェーズで使用されるデバッガ

組み込み設計はこれまで以上に複雑さを増し、開発期間はますます短縮される傾向にあります。この課題に対処するために、現在、多くのプロジェクトマネージャは、プロジェクトのあらゆるフェーズで開発者をサポートできる、デバッグおよびトレースツールを利用しています。

ローターバッハが提供するデバッグおよびトレースツールファミリーであるTRACE32は、一貫した概念および環境を提供します（これは、ユーザーがカスタマイズできるスクリプトでさらに拡張可能です）。TRACE32を使用することで、習熟に要する工程が短縮され、実際の開発作業に時間を割くことができるようになります。10年以上TRACE32を使用することで、実用的な知識を身につけた開発者も珍しくありません。ここでは、TRACE32のどのような点が他の製品と異なるかについて紹介します。

- ハードウェアベース/ソフトウェアベースのツール
- 新しいプロセッサ向けの初期サポート
- 幅広いラインナップのサポートプロセッサ
- 多くのテストおよび解析機能
- 組み込みツールチェーンへのシームレスな統合

### ハードウェアツールおよびソフトウェアツール

ローターバッハの主力事業は、ハードウェアベースデバッグツールおよびトレースツールの設計、製造です。さらに、ローターバッハは20年以上ロジックアナライザを提供し続けています。TRACE32ロジックアナライザの主な特長は、ハードウェアベースのデバッグツール、トレースツールにシームレスに統合できる点です。PowerTrace　IIに統合されたロジックアナライザを使用した代表的なアプリケーションについては、6ページの記事「JTAG信号のチェック」を参照してください。

高速で効率の良いコンピュータでは、PCおよびワークステーションで、多くのシミュレーションと検証が実行されています。現在、組み込み業界では、仮想ターゲットでのプリシリコンソフトウェア開発が当たり前になっています。ローターバッハでは、プロジェクトのこのフェーズに対して、純粋なソフトウェアソリューションを提供できます。 »

### コンテンツ

## 仮想ターゲット

現在では、最初のハードウェアプロトタイプが使用可能になるよりもずっと以前に、仮想ターゲットを使用してソフトウェア開発に着手する事例がますます増えています。仮想ターゲットが使用可能になれば、直ちにドライバ、オペレーティングシステム、およびアプリケーションのデバッグを開始できます。

多くの仮想ターゲットでは、デバッグおよびトレースに独自のAPIが用意されています。独自のAPIを使用しない場合は、標準MCD-API（http://www.lauterbach.com/mcd_api.html）を使用できます。現在では、多くの新しいプロジェクトにマルチコアチップが使用されています。そのため、2011年以降、ローターバッハは仮想ターゲットのマルチコアデバッグサポートを拡充しています。

### プリシリコン検証

半導体製造会社にとっては、プロセッサやSystem-on-Chip（SoC）の生産を実際に開始する前に、その設計を検証することが重要です。JTAGインタフェース、コア全体、およびコアとペリフェラル間の相互動作などについて、個々のセクションが徹底的にテストされます。

従来、このテストには、ハードウェアベースのTRACE32デバッグツールに接続した、シリコン用エミュレータ（Palladiumなど）またはFPGAプロトタイプが使用されてきました。この実行速度は、実際のプロセッサよりもかなり低速になります。

現在では、VerilogまたはSystemCモデルの最初の検証を、PCまたはワークステーション上で直接実行できます。純粋なソフトウェア検証では、デバッグハードウェアは使用できません。そのため、2011年に、ローターバッハはソフトウェアにVerilogバックエンドを追加しました。これは、信号レベルでのJTAGインタフェースのシミュレーションを実行します（図1を参照）。

TRACE32ツールをプリシリコン検証に統合することで、最新のプロセッサおよびSoCに対する重要な初期サポートが実現します:

- シリコンの初回出荷前に、テスト済みツールの準備を完了
- 新しいプロセッサ/SoCの専門知識が利用でき、お客様が自由にアクセス可能
- TRACE32デバッガの起動スクリプトを使用可能

## 60以上のプロセッサアーキテクチャをサポート

ローターバッハでは、組み込み開発市場において一般的なすべてのプロセッサおよびSoCで使用できるツールを用意しています。実際に、複数のコアに対応できるツールを提供しているプロバイダは、ローターバッハ以外にはありません。TRACE32ツールを使用することで、標準的なコントローラ、DSP、FPGAソフトコア、コンフィギャラブルコアなどの、あらゆるプロセッサ/SoCをマルチコアチップに統合してデバッグできます。

また、2011年には、ローターバッハはサポート対象に多数の新しいプロセッサおよびマルチコアチップを追加しました。概要については、4ページの表を参照してください。

## テストおよび解析機能

プロジェクトの各フェーズでは、独自のテストおよび解析機能が必要になります。これらの機能を提供するために、TRACE32 PowerView GUIには、非常に多くのコマンドおよびメニューが用意されています。バウンダリスキャンコマンド（図2を参照）、コア検出コマンド、およびJTAGコマンド

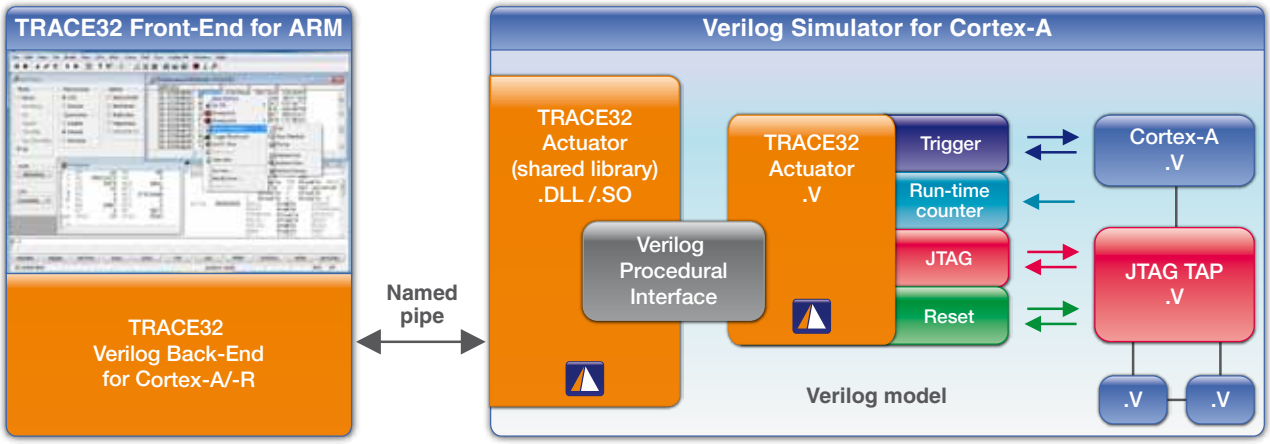



図1: TRACE32フロントエンドの各ユーザー入力に対して、Verilogバックエンドによりモデル検証用のJTAG信号が生成される

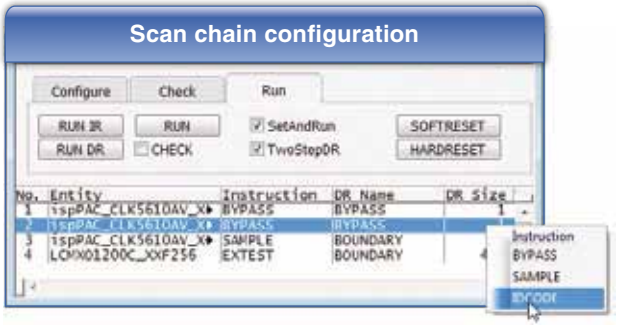



図2: ハードウェアの試用にバウンダリスキャンコマンドを使用可能

の操作コマンドは、低水準コマンドの例の一部です。品質/テストフェーズでは、開発者へのサポートは高水準コマンドにより提供されます。また、通常はトレースデータの解析は高水準コマンドで対処されます。たとえば、関数ルーチンの測定、エネルギープロファイリング、コードカバレッジの詳細などです。

2011年初頭から、ローターバッハは主要なプロセッサアーキテクチャからホストコンピュータへの、トレース情報のストリーミングを可能にしています。これにより、これまでよりも大幅に多くの診断データを集めることができるようになったため、品質保証がより容易になりました。詳細については、7ページの記事「コードカバレッジの簡略化」を参照してください。

## 組み込みツールチェーンへの統合

TRACE32ソフトウェアは、組み込み設計における、あらゆる一般的な基本コンポーネントに使用できるオープン設計になっています。たとえば、次のようなコンポーネントに対応しています:

- ホストオペレーティングシステム
- プログラミング言語およびコンパイラ
- ターゲットオペレーティングシステム
- Android VM Dalvikなどの仮想マシン

TRACE32のオープンなAPIを使用することで、多くのサードパーティ製ツール(たとえば、Eclipseなどの専用IDE、グラフィカルプログラミングツール、および外部プロファイリングツールなど)とのシームレスな相互動作が可能になります。2011年には、この領域に複数の新しい技術開発が追加されました。

スコットランド企業であるCriticalBlue社製並列化ツール「Prism」は、シングルコアコードをマルチコアチップで実行できるように移行する際に、開発者をサポートするツールです。このツールを使用することで、機能コードを変更せずに別の並列化戦略を試すことができます。最適な戦略が決定したら、段階的に並列化を進めることができます。この際にも、Prismによるサポートが提供されます。

2011年7月以降、ローターバッハはPrism形式でトレース情報をエクスポートするオプションを含めることで、実際のコードオペレーションにより記録されたトレースをCriticalBlueツールで使用できるようにしています。

14ページの記事「近づくシミュレーションと実動作」では、もう1つの技術革新であるMATLAB　Simulink®とTRACE32との統合を詳しく説明しています。

## 長い製品寿命

ローターバッハは、新しいテクノロジへの移行には、長期の過渡フェーズが存在するという哲学を持っています。重要なプロジェクトの途中で、テクノロジの変更を受け入れるようにお客様に強要することはありません。

たとえば、ローターバッハは2012年5月から、TRACE32 PowerViewのグラフィカルユーザーインタフェースで、QTバージョンの導入を予定しています(図3を参照)。QTの導入により、Linux、Mac　OS　X、およびその他のホストオペレーティングシステムで最新のGUIを使用できるようになります。

ローターバッハは、お客様が最も都合の良い移行時期を決定できるように、TRACE32 PowerViewのMotifバージョンのサポートを継続します。

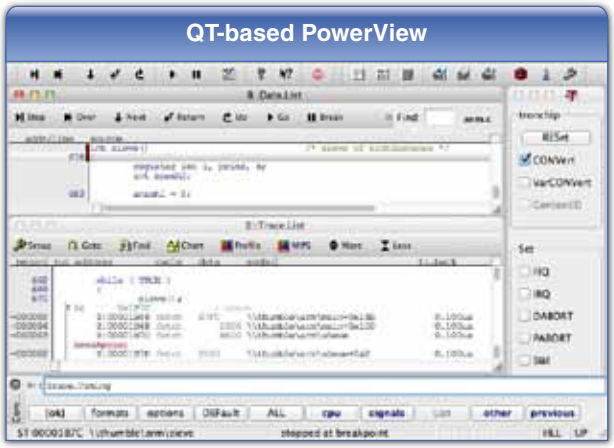



図3: Linux、Mac　OS　X、およびその他のオペレーティングシステム向けQTベースGUI

本書の次ページ以降では、現在および将来的なプロジェクトに役立つ、詳しい情報が提供されています。ここで、お客様のプロジェクトを成功に導く機能が見つかることを期待します。弊社では、「ESC　Silicon　Valley」(3月26日～29日、開催地:　サンノゼ)、および年間を通して米国で開催されるその他のショーで、このうちのいくつかのデモを実演する予定です。.

# 新サポートプロセッサ

| New Derivatives | |
|---|---|
| Altera | Cortex-A / -R<br>• FPGA with Cortex-A9 MPCore as Hardcore<br>MIPS32<br>• MP32 |
| AppliedMicro | PPC44x<br>• 86290 / 491 / 791 Q2/2012 |
| ARM | Cortex-A / -R<br>• Cortex-A7/Cortex-A7 MPCore<br>• Cortex-A15<br>• Cortex-A15 MPCore<br>• Cortex-R5/Cortex-R5 MPCore<br>• Cortex-R7/Cortex-R7 MPCore |
| Beyond Semiconductor | Beyond<br>• BA22 |
| Broadcom | MIPS32<br>• BCM35230<br>• BCM63168, BCM63268<br>• BCM7231, BCM7358 |
| Cavium | MIPS64<br>• CN61XX / CN62XX / CN66XX<br>• CN67XX / CN68XX |
| Ceva | CEVA-X<br>• CEVA-XC |
| CSR | ARM11<br>• QUATRO 4500 |
| Cypress | ARM9<br>• EZ-USB FX3 |
| Energy Micro | Cortex-M<br>• Giant Gecko |
| Freescale | MCS12X<br>• MC9S12VR, MC9S12XS<br>• MM912F634<br>Cortex-A / -R<br>• i.MX 6 Series<br>MPC55xx/56xx<br>• MPC5604E, MPC5675K,<br>• MPC5676R<br>Power QUICC III<br>• P1010, P1020<br>• P2040, P2041<br>• P3041, P4040, P4080<br>• PSC9131<br>QorIQ 64-Bit<br>• P5010, P5020 |
| Fujitsu | Cortex-A / -R<br>• MB9DF126, MB9EF126 |
| IBM | PPC44x<br>• 476FP Q2/2012 |
| Ikanos | MIPS32<br>• Fusiv Vx185 |
| Infineon | TriCore<br>• TriCore Multi-Core Architecture |
| Intel® | Atom™/x86<br>• Atom D2500, Atom N550<br>• Core i3/i5/i7 2nd Generation |
| Lantiq | MIPS32<br>• XWAY xRX100<br>• XWAY xRX200 |
| LSI | PPC44x<br>• ACP344x Q2/2012 |
| Marvell | ARM9 Debug-Cabel<br>• 88E7251<br>ARM11 Debug-Cabel<br>• 88AP610-V6, MV78460-V6<br>Cortex-A / -R Debug-Cabel<br>• 88AP610-V7, MV78460-V7 |
| Nuvoton | Cortex-M<br>• NuMicro |
| NXP | Cortex-M<br>• LPC12xx<br>Beyond<br>• JN5148 |
| Qualcomm | MIPS32<br>• AR7242<br>Cortex-A / -R<br>• Krait |
| Renesas | V850<br>• V850E2/Fx4: 70F3548..66 70F4000..70F4011<br>• V850E2/Fx4-L: 70F3570..89<br>• V850E2/Px4: 70F3503 / 05 70F3507 / 08 / 09<br>78K0R / RL78<br>• 78K0R/Kx3-C/L<br>• RL78/G14, RL78/G1A<br>• RL78/F12, RL78/I1A<br>SH<br>• SH708x with AUD/Onchip-Trace<br>• SH7147 |

| New Derivatives | |
|---|---|
| Samsung | ARM7<br>• S3F4<br>Cortex-A / -R<br>• S5PV310<br>Cortex-M<br>• S3FM, S3FN |
| ST-Ericsson | Cortex-A / -R<br>• A9500, A9540, M7400<br>MMDSP<br>• A9500, A9540 |
| STMicro-electronics | MPC55xx/56xx<br>• SPC56A80, SPC56HK<br>Cortex-M<br>• STM32F2xx, STM32F4xx |
| Synopsys | ARC<br>• ARC EM4, ARC EM6 |
| Tensilica | Xtensa<br>• BSP3, LX4, SSP16 |
| Texas Instruments | MSP430<br>• CC430Fxxx, MSP430FR5xxx<br>• MSP430x1xx..MSP430x6xx |
| Texas Instruments (Cont.) | ARM9<br>• AM38xx<br>• OMAP4460 / 4470<br>• TMS320C6A81xx<br>• TMS320DM81xx<br>Cortex-A / -R<br>• AM335x, AM38xx<br>• OMAP4460 / 4470 / 543x<br>• RM48L950<br>• TMS320C6A81xx<br>• TMS320DM81xx<br>• TMS570LS3xxx<br>Cortex-M<br>• AM335x<br>• OMAP4460 / 4470 / 543x<br>• TMS470MFxxx<br>TMS320C28X<br>• TMS320C28346 / F28069<br>TMS320C6x00<br>• OMAP4460 / 4470 / 543x<br>• TMS320C6A81xx<br>• TMS320DM81xx<br>• TMS320TCI6616 / 18 |
| Xilinx | Cortex-A / -R<br>• Zynq7000 |

# 小型パッケージ形式のコア向けNexus-Trace

Freescale社製MPC560xB/Cファミリーのコントローラ、またはST社のSPC560B/Cコントローラに統合されているNexusセルは、コアにより実行される命令のトレースデータを生成できます。オペレーティングシステムが使用されている場合は、タスクスイッチングについての情報も生成されます。

マイクロコントローラには、TRACE32などの外部トレースツールでこのトレースデータが記録されるように、トレースインタフェースが備わっている必要があります。しかし、MPC560xB/Cファミリーメンバーの標準パッケージには、このインタフェースは搭載されていません。プログラム実行に関するこの有益なデータに、開発フェーズでアクセスできるようにするために、シリコン互換のマイクロコントローラが208ピンBGA開発パッケージで提供されています。このマイクロコントローラでは、MDO（メッセージデータ出力）4個を備えたNexusインタフェースを使用しています。

2011年中盤以降、ローターバッハは、MPC560xB/Cのアダプタの提供を開始しました。このアダプタを使用することで、ターゲットハードウェアの元々のコントローラを、Nexusインタフェースを備えた208ピンコントローラに交換できます。MPC560xB/Cアダプタは、208ピンBGA開発パッケージに入った適切なMPC560xB/Cコントローラと、TRACE32トレースツールの接続に使用するNexusインタフェースを備えたMictorプラグで構成されています（図4の青で示された部分）。これに加えて、Tokyo　Eletech社製ソケットアダプタが必要です。

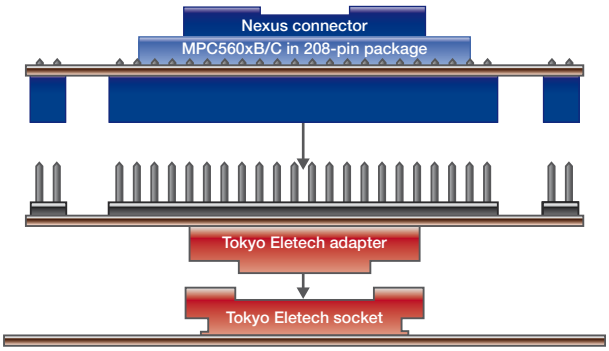



図4: MPC560xB/Cアダプタを使用することで、元々のコントローラの代わりにNexusインタフェースを備えた開発パッケージを使用できる

# JTAG信号のチェック

ローターバッハ製PowerTrace　IIには統合ロジックアナライザが備わっており、標準のデジタルプローブに同梱されています。この製品は、最大200MHzのサンプリングレートで、17個のデジタルチャネルを記録できます。このロジックアナライザを使用することで、1024Kサンプル分のメモリを節約できます。使用例としては、プリシリコン検証中のJTAG信号のテストなどがあります（図6および7を参照）。

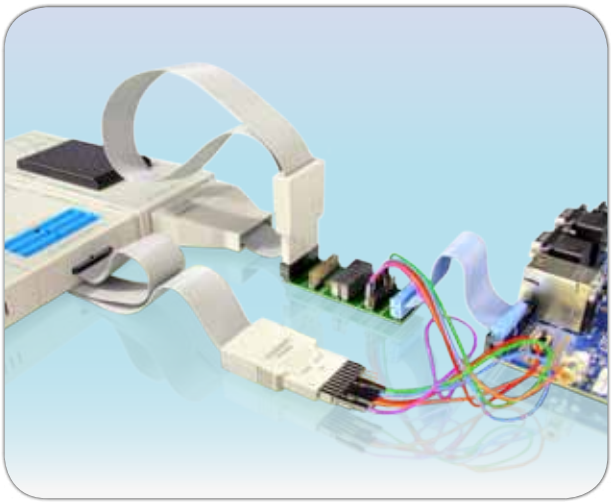


図5: JTAG信号記録の測定装置

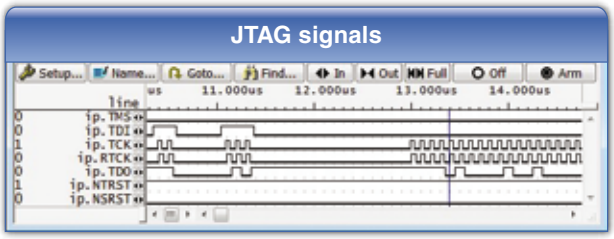



図6: 記録されたJTAG信号

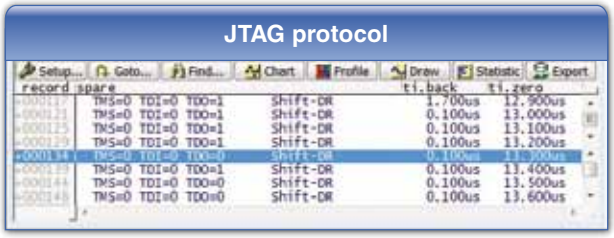



図7: JTAG信号のプロトコル表示

# ターゲットOS対応機能の拡張

次のバージョンへの対応が実施されています:

- eCos 3.0
- Linux v3.0
- SMX v4
- embOS 3.80
- MQX 3.6
- FreeRTOS v7
- RTEMS 4.10

- QNXトレースロガーの内容は、TRACE32　QNX　OS対応機能を使用して表示できる。TRACE32コマンドグループのLOGGER.Chartを使用して、タスクスイッチのグラフィカル表示も可能。

- 場所に依存しない実行可能ファイルを使用できるように、TRACE32 QNX OS対応機能が修正されている。

| 新サポートRTOS | |
|---|---|
| µC/OS-II for Andes | available |
| Elektrobit tresos (OSEK/ORTI) | available |
| Erika (OSEK/ORTI) | available |
| FreeRTOS für AVR32 | available |
| Linux for Beyond | planned |
| MQX for ARC | available |

| | |
|---|---|
| OSEK/ORTI SMP | planned |
| PikeOS | available |
| PXROS-HR Run Mode Debugging | available |
| RTEMS for Nios II | available |
| Sciopta 2.x | available |
| SYS/BIOS for ARM | available |
| VxWorks SMP | available |

# コードカバレッジの簡略化

**2011年3月より、実行中のターゲットからホストのハードディスクにTRACE32のトレース情報をストリーミングできるようになりました。この方法で取得できる大量のプログラムフローデータにより、コードカバレッジが大幅に簡略化されます。**

## トレースベースのコードカバレッジ

医療や自動車のような業界では、多くの場合、ステートメントカバレッジおよび条件カバレッジの証明がシステムの品質仕様を満たしている必要があります。

- **ステートメントカバレッジは、システムのテスト中に各コード行が実行されたことを証明する。**
- **条件カバレッジは、各条件命令に対して、パス分岐とフェイル分岐の両方が1回以上実行されたことを証明する。**

多くの組み込みシステムでは、高度に最適化されたコードをリアルタイムでテストする必要があります。このような場合では、コードインストルメンテーションの代替手法および非リアルタイムオペレーションは使用できません。

これらの要件を満たすには、ターゲットプロセッサ/SoCは、次の前提条件を満たす必要があります:

1. 実装されているコアには、コアトレースロジックが搭載されている必要があります(図8を参照)。このロジックは、コアにより実行される命令についての情報を生成します。トレースロジックの動作に応じて、タスクスイッチおよびリード/ライト操作も出現します。

2. 情報を損失せずに外部ツールでトレース情報を記録するために、プロセッサ/SoCには十分な帯域幅のトレースポートが備わっている必要があります。

## 従来型の測定プロセス

従来、TRACE32のコードカバレッジ解析は、次の手順で実行されていました:

1. プログラムの実行を開始します。これは、トレースメモリが満杯になると自動的に停止されます。
2. トレースメモリの内容をコードカバレッジデータベースに転送します。
3. プログラムの実行を継続します。

各測定ステップで、収集できるデータ量はトレースツール内の使用可能なメモリサイズによって制限されていました。コードカバレッジ解析の結果は、すべての測定が完了した後、または必要に応じて各中間ステップの後にチェックできました。

## 新しい手法: ストリーミング

記録時に、トレースデータがホストコンピュータのドライブに転送されると、全ソフトウェアルーチンが、1つの測定ステップで記録できます。ストリーミングされたデータは、ハードディスク上のファイルに保存されます。TRACE32は、ハードディスクすべてがトレースデータで満杯になってしまうことを防ぐために、残りの空きメモリが1ギガバイト未満になると直ちにストリーミングを停止します。

ストリーミングを可能にするには、次の技術的な前提条件を満たす必要があります:

- 64ビットのホストコンピュータと64ビットのTRACE32実行可能ファイル
- トレースツールとホストコンピュータ間のインタフェースは可能な限り高速であること
- トレースソースとトレースツールの最適な設定

»

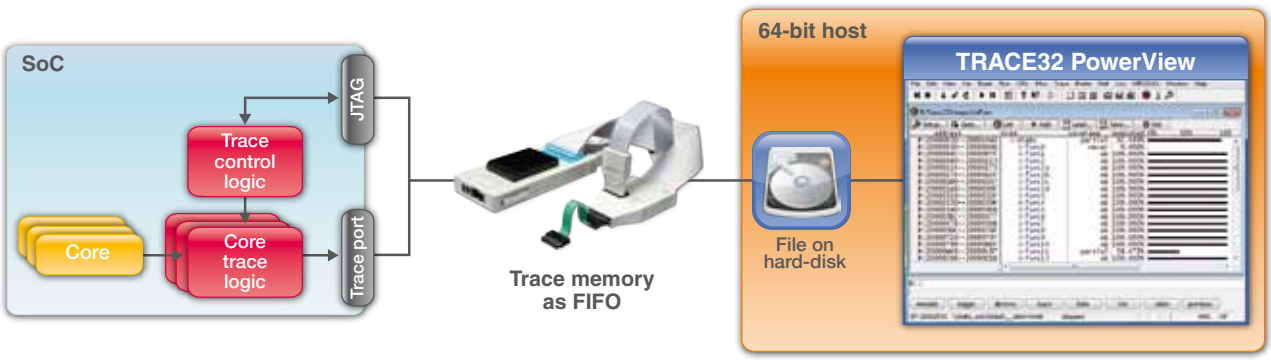



図8: コードカバレッジ解析の場合、最大1テラバイトのトレースデータをホストコンピュータにストリーミング可能

### 高速ホストインタフェース

トレースポートを経由してエクスポートされるトレースデータの量は、ターゲットシステムのハードウェアに依存します。コアの数、トレースポートピンの数、およびトレースクロックの速度はすべて、重要なパラメータです。また、コアトレースロジックの使用するプロトコルも、重要な役割を担います。たとえば、ARM PTMプロトコルは、ARM ETMv3プロトコルよりもコンパクトです（図9を参照）。

組み込みソフトウェアは、もう1つの主要な変数です。主にキャッシュでデータや命令を何度もジャンプ/検索するようなソフトウェアプログラムのほうが、連続する多数の命令を処理し、かつデータや命令が利用可能になるまで待機することの多いプログラムのよりも、1秒あたりに生成されるトレースパッケージは多くなります。

データの容量は変化しますが、常に大きいことには変わりません。ストリーミングは、ツールとホストコンピュータ間の転送速度が、データを損失せずにトレースポートからホストコンピュータにすべてのデータを転送するのに十分な速度である場合に限り、正常に機能します。PowerTrace IIの推奨インタフェースは、1ギガビットイーサネットインタフェースだけです。

チップのトレースロジックをプログラムすることで、生成されているトレースデータの量に直接影響を与えることができます。ロジックは、コードカバレッジ解析に関連するトレース情報のみが生成されるようにプログラムする必要があります。次に、この点を説明する例を示します。

## ETM/PTM: 最適な設定

ETMとPTMは、ARM/Cortexアーキテクチャの異なるコアトレースロジック実装です。
ETMは、プログラムにより実行された命令のトレース情報だけが生成されるように設定できます。コードカバレッジには、リード/ライト操作の情報は不要です。デフォルトでは、PTMはプログラムフローの情報しか生成しません。そのため、PTMを設定する必要はありません。

どちらのトレースソースも、仮想アドレス命令をエンコードします。組み込み設計で、Linuxまたは組み込みWindowsなどのオペレーティングシステムを使用している場合、仮想アドレスを明確に物理アドレスにマップすることはできません。また、内部に命令が格納される仮想アドレスを定義する情報が生成されるように、トレースソースを設定する必要もあります。

### PowerTrace vs. PowerTrace II

TRACE32トレースツールは2種類の設計で使用できます。この2つの設計は、特に機能の点で異なります。

#### PowerTrace

- 256メガバイトまたは512メガバイトのトレースメモリ
- USB 2.xおよび100メガビットイーサネット
- ホストコンピュータへの最大転送速度は80メガビット/秒
- トレースデータのソフトウェア圧縮（係数3）
- 100MHzでのメモリインタフェース

#### PowerTrace II

- 1/2/4ギガバイトのトレースメモリ
- USB 2.xおよび1ギガビットイーサネット
- ホストコンピュータへの最大転送速度は500メガビット/秒
- ETMv3およびPTMのトレースデータのソフトウェア圧縮（係数6）
- 233MHzでのメモリインタフェース

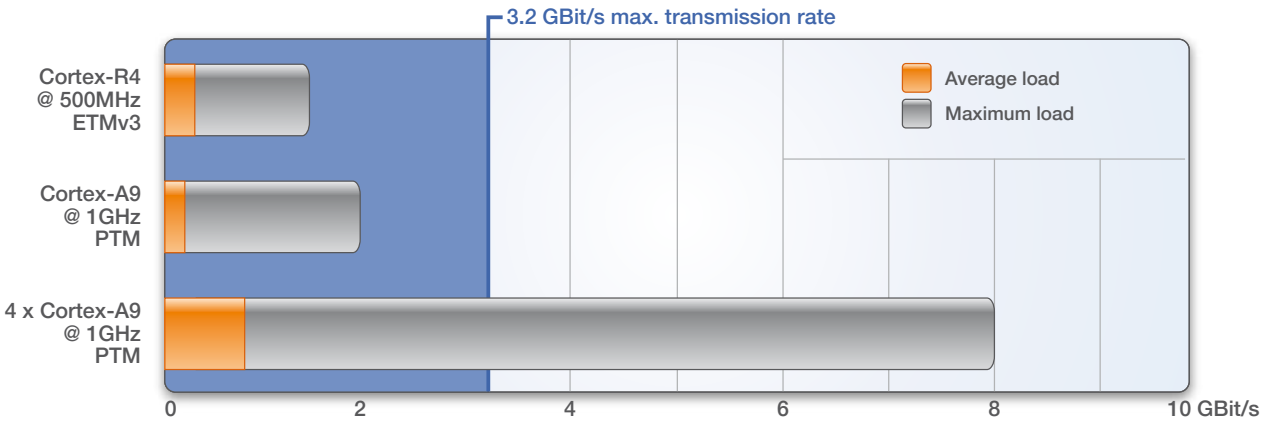



図9: ホストのプログラムシーケンス情報のストリーミングには、通常は3.2GB/秒の転送速度が適正

ARM　ETM/PTMの場合、トレースデータの量をさらに削減できます:

- コードカバレッジ解析では、時間の情報は解析せず、また、必要でもありません。そのため、TRACE32トレースツールは、トレースデータはタイムスタンプなしでホストに転送されるように設定することをお勧めします。これにより、データ量は3分の1に削減されます。

- また、PowerTrace　IIでは、FPGAベースでトレースデータをハードウェア圧縮する機能が提供されています。この機能を使用することで、3.2ギガビット/秒でトレースデータをホストコンピュータに転送できます。図9は、この転送速度がデータ損失なしでETM/PTMデータをストリーミングする場合に、通常は十分であることを示しています。

### Nexus: 最適な設定

MPC5xxx/SPC5xxファミリーのプロセッサでは、コアトレースロジックはNexus標準に実装されています。Nexusクラス2トレースセルは、個々のコアのプログラムシーケンス詳細しか必要としないため、コードカバレッジ解析の実行に適しています。分岐履歴メッセージングが使用されている場合は、トレースデータは大幅に小さくなります。標準のトレースデータと比較して、係数10での削減が現実的です。Nexusトレースポートからのストリーミングをサポートしているのは、PowerTrace IIのみです。

また、ストリーミングは、TRACE32でサポートされており、かつトレースポートを備えているその他すべてのプロセッサ/SoCに対しても機能します。

## SMPシステムのコードカバレッジ

TRACE32は、SMP(対称型マルチプロセッシング)システムのコードカバレッジもサポートしています。コードカバレッジでは、命令が実行されたことを証明する必要はありますが、どのコアがコードを実行したかは重要ではありません。図10に、2つのCortex-A9　MPCoreのコードカバレッジの結果を示します。

ステートメントカバレッジおよび条件カバレッジでは、条件文のフェイル分岐しか実行されなかった場合、ステートメントは黄色でハイライトされ、「not　exec」とマーキングされます。カバレッジ詳細には、各ステートメントまたはステートメントの各分岐がどの位の頻度で実行されたかについての詳細がリストされます。.

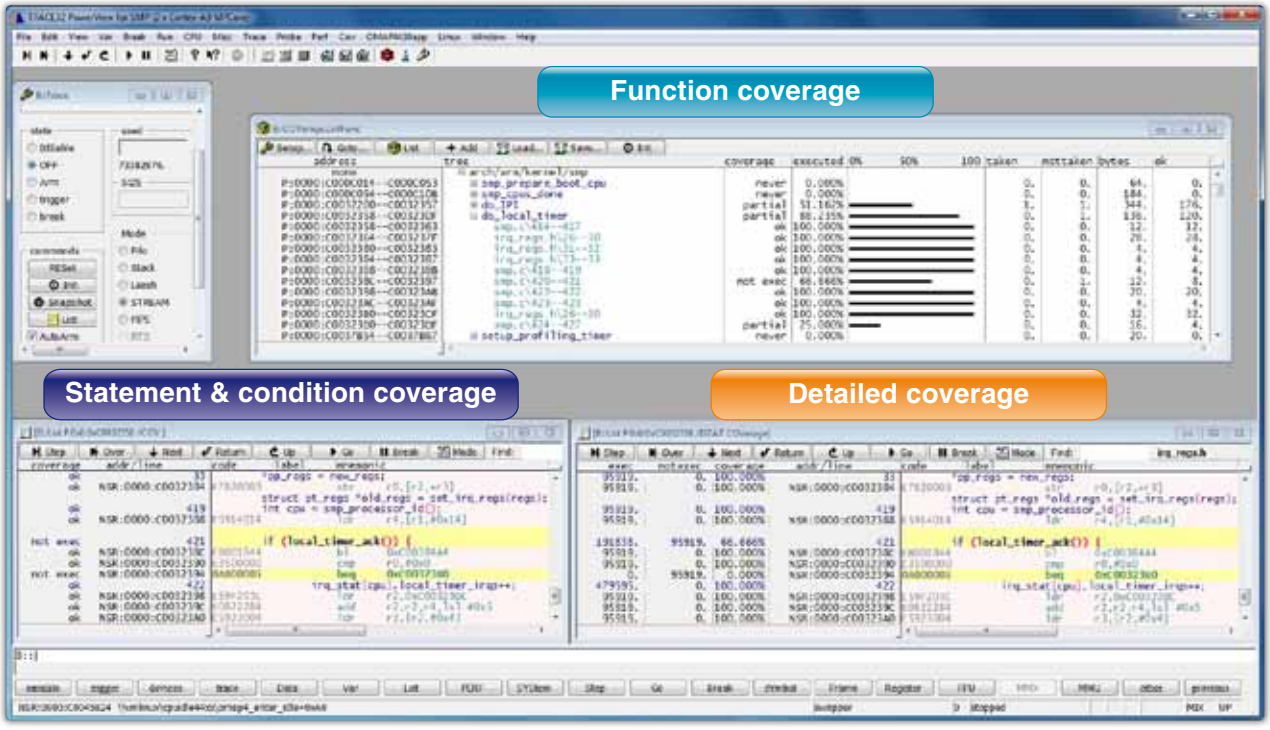



図10: SMPシステムのコードカバレッジ解析

# CoreSightトレースメモリコントローラ

**新しいCoreSightトレースメモリコントローラは、より多くのトレースインフラストラクチャ向け設計オプションをSoC設計者に提供します。TRACE32は、TMCを使用する初期の設計をサポートしています。**

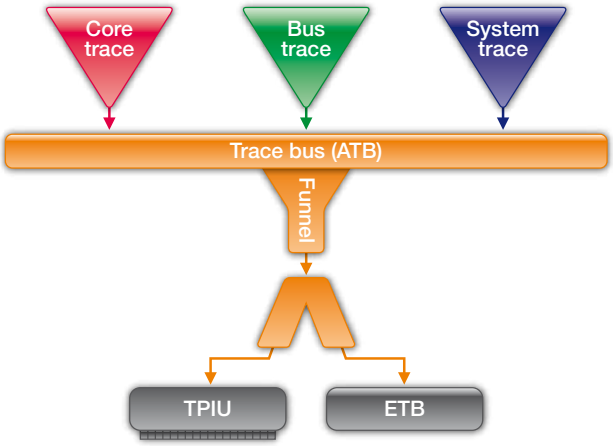



図11: CoreSight Funnelは、トレースマクロセルによって生成されたすべてのトレースデータを単一のデータストリームに組み入れる

CoreSightを通じて、SoC内部プロセスの解析に必要な診断データが、「トレースマクロセル」により生成されます。トレースマクロセルには、次の3種類があります:

- **コアに割り当てられ、そのコアによって処理される命令についての情報を生成するコアトレースマクロセル。プロセススイッチおよびロード/保存操作についての情報は、トレースセルの設計に応じて生成される。**
- **バスに厳密に割り当てられ、バスで発生するデータ転送についてのトレース情報を生成するバストレースマクロセル。**
- **ハードウェアトレーストリガーのトレース情報を生成したり(システムイベントトレース)、アプリケーションソフトウェアのコードインストルメンテーションによって生成された診断情報を提供したりする、システムトレースマクロセル**

CoreSight Funnelは、すべてのトレースデータを単一のデータストリームに組み入れます(図11参照)。このトレースデータストリームは、その後オンチップメモリバッファに保存されるか(ETB)、またはトレースポートを使用して外部ツールにエクスポートされます(TPIU)。現在実装が進められているCoreSightトレースのIPは、多数のトレースマクロセルを含む複雑なマルチコアSoCを対処するときに、限界に達する場合があります。

## ARM CoreSight

CoreSightを使用することで、ARMで多数のIPブロックのセットを使用できるようになります。これにより、SoC設計者はカスタムのデバッグおよびトレースインフラストラクチャを構築できます。

単一のデバッグインタフェースで、SoCのすべてのコアの制御と調整、およびすべてのメモリへのアクセスに十分対応できます。

また、SoC内で発生しているプロセスについての診断データを、リアルタイムのパフォーマンスに影響を与えることなく提供することも、単一のトレースインタフェースで十分対応できます。

- **ETB:** 多くの場合、オンチップトレースメモリは、意味のある将来的な解析を行うためのトレースデータを記録するには小さすぎます。それでも、ETBの一般的なサイズは4～16キロバイトです。
- **TPIU:** トレースポートの出力能力を超えるトレースデータが生成されようとしている場合、システムステートが発生する場合があります。CoreSightは、トレースデータがTPIUによってエクスポート可能な場合のみ、トレースマクロセルからのトレースデータが引き継がれるように設計されています。生成されたトレースデータが長期間トレースマクロセル内に留まっていると、その場所のFIFOがオーバーフローして、重要なデータが損失する可能性があります。

新しいCoreSightトレースメモリコントローラは、上記の両方のシナリオに対するソリューションを提供します。

## 組み込みトレースバッファとしてのTMC

チップの製造会社は、後で分析するためにより多くのオンチップトレースデータを保存できるようにするために、最大4ギガバイトのSRAMをトレースメモリコントローラに理論的に接続できます(図12を参照)。

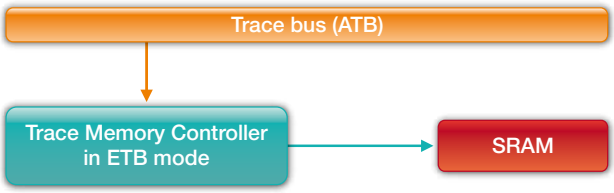



図12: ETBモードでは、トレースメモリコントローラは最大4ギガバイトのオンチップトレースメモリを使用可能にできる

## 組み込みトレースFIFOとしてのTMC

TPIUによるエクスポート中のトレースデータストリームの検査により、ほとんどのトレースポートの帯域幅は、通常の動作には十分であることがわかっています。オーバーロードや、それによって起こるトレースデータの損失は、ピークの発生時のみ起こります。

トレースメモリコントローラは、SoCのトレースインフラストラクチャに統合できます。統合することで、トレースメモリコントローラは組み込みトレースFIFOとして動作し、TPIUでのロードにおけるピークを和らげます(図13を参照)。このETFは、トレースデータの損失が起きないように設計されていま

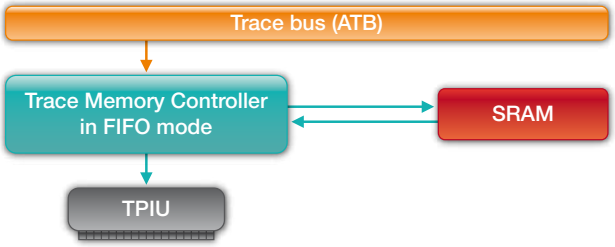



図13: FIFOモードでは、トレースメモリコントローラでTPIUのロードピークを和らげることができる。これにより、データの損失を回避できる。

す。ETFのサイズは、512バイト～4ギガバイトの間で自由に定義できます。
上記で説明したトレースインフラストラクチャのトレースメモリコントローラへの統合は、両方とも簡単な例です。もちろん、より複雑かつ柔軟性の高い方法で、CoreSightシステムにTMC IPブロックを構築することもできます。

## TRACE32の変更

ご想像の通り、ローターバッハは、トレースメモリコントローラの設定および操作のために、TRACE32ソフトウェアを変更する必要があります。これは特に、SoCに統合されているトレースメモリコントローラが、以前はサポートされていなかった新しい方法を使用している場合に当てはまります。TRACE32のユーザーに必要とされる操作は、TMCの基本アドレス設定だけです。その後、すべての実績のあるトレース表示および解析機能を、通常どおり使用できます。

## 高速リンクへのルーターとしてのTMC

専用トレースポートから脱却するという考えは、組み込みコミュニティで長らく検討されてきました。この移行についての議論の一部は、確かに有意義なものです。

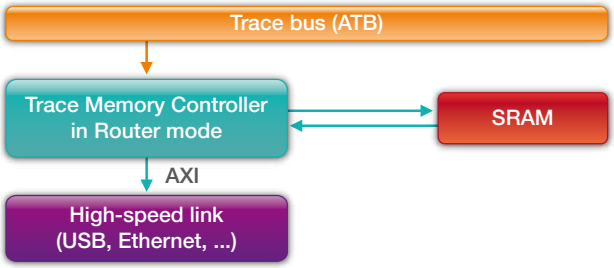



図14: ルーターモードでは、トレースメモリコントローラにより、エクスポート対象のトレースデータが高速標準インタフェースに転送される

今回初めて、トレースメモリコントローラを使用して、CoreSightトレースが高速標準インタフェースに接続できるようになりました。USBまたはイーサネットインタフェースは、特に多くの最終製品で使用可能なため、一般に好まれています。外部トレースツールとその他の接続デバイスがインタフェースを共有することが理想です。

TMCは、SoC内では組み込みトレースルーターとして動作し、高速インタフェースのIPにエクスポートするために、AXIバスを経由してトレースデータを受け渡すタスクを持っています(図14を参照)。

この新しいトレースのエクスポート方法では、全く新しいトレースツールが必要です。現在、ローターバッハは大手半導体製造会社と密接に連携して、このテクノロジの転換に適したツールを開発しています。

### TRACE32 CoreSight 対応機能

- CoreSightに統合できるすべてのコアでオープンに使用可能。ローターバッハは、すべてのARM/Cortexコア、多数のDSP、およびコンフィギャラブルコア用のデバッグソリューションを提供
- 非対称型マルチプロセッシング(AMP)および対称型マルチプロセッシング(SMP)をサポート
- JTAGインタフェースと2ピンシリアルワイヤデバッグを使用したデバッグ
- すべてのコアの同期デバッグ
- CoreSight Cross Trigger Matrixをサポート
- あらゆるタイプのトレースマクロセル(ETM、PTM、HTM、ITM、STM、その他)をサポート
- パラレルおよびシリアルトレースインタフェース用ツール
- マルチコアトレース

# Cortex-M3/M4向けのインテリジェントなトレース解析

**適切なトレース解析が提供されている場合、トラブルシューティング、パフォーマンス調整、コードカバレッジはすべて、組み込みシステム上ですばやく正確に実行できます。2011年、ローターバッハは、Cortex-M3/M4プロセッサの最適化されたトレース解析を有効にする新しい道を開拓しました。**

## ETMとITMの組み合わせ

Cortex-M3/M4プロセッサの場合、トレース情報は2つの異なるソースから生成されます(図17を参照)。ETMv3は、実行された命令についての情報を生成します。ITMは、Data Watchpoint and Trace Unit(DWT)によってサポートされた、実行されたリード/ライトアクセスについての情報を生成します。

リード/ライトアクセス用のITMトレースパッケージには、データアクセス、データ値、およびプログラムカウンタの情報が含まれます。

プログラムカウンタを解析することにより、別途生成されたデータアクセスをプログラムシーケンスにシームレスに統合することができます(図15を参照)。これにより、エラーの場所を非常に簡単に特定できるようになります。また、全体的なプログラムトレースにライトアクセスが組み込まれていれば、エラーの原因(アドレスに不正なデータ値が書き込まれているなど)を簡単に特定できます。

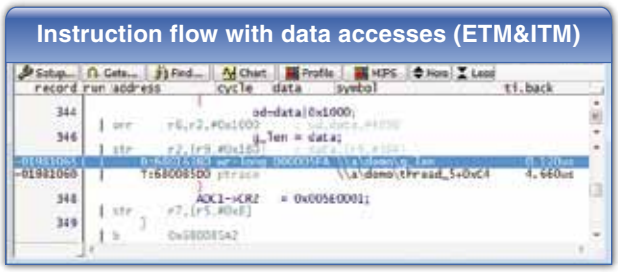



図15: ETMとITMトレースデータを組み合わせることにより、リード/ライトアクセスをプログラムシーケンスにシームレスに統合できる。

## OS対応のトレース

オペレーティングシステムがCortex-M3/　M4で実行されている場合、トレース解析にはタスクスイッチ情報が必須になります。
タスクスイッチについての情報を受信するには、次の方法を使用できます: カーネルによって、対応するOS変数に現在

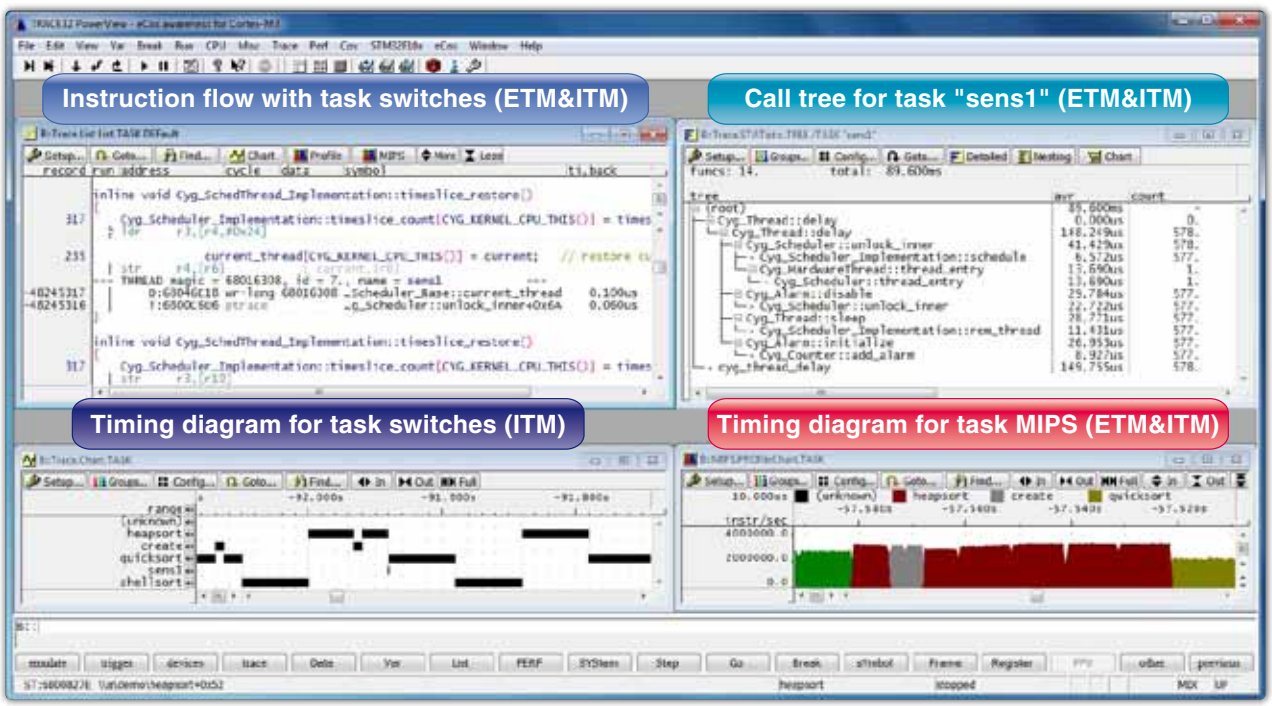



図16: ETMとITMトレースデータを組み合わせることにより、eCosオペレーティングシステム用に幅広いトレース解析が提供できる

のタスクの識別子が書き込まれる書き込みサイクルのトレース情報は、ITMを使用して生成できます。前述のように、ライトアクセス情報は、プログラムフロートレースにシームレスに統合できます。これにより、トレースリストの可読性が向上します（図16を参照）。また、図16に示すように、タスクスイッチのプログラムシーケンスへの統合は、ランタイム解析の基礎にもなります。

## 3つの記録モード

Cortex-M3/M4プロセッサにより生成されるトレース情報を記録するために、ローターバッハは次の3つのモードをサポートしています:

- **FIFOモード: TRACE32 CombiProbeの128メガバイトのメモリに情報を保存する。**
- **STREAMモード:　ホストコンピュータのハードディスクに情報をストリーミングする。**
- **リアルタイムプロファイリング: ランタイム中にトレース情報をホストコンピュータにストリーミングして解析する。**

最初の2つの記録モードでは、記録の完了後にトレース情報が収集されて解析されます。

各記録モードには、独自の特長があります。最もよく使用されているモードはFIFOです。このモードは処理が早く、必要となるのはエラーの場所の特定とランタイム解析のみです。

Cortex-M3/M4プロセッサに実装されているETMv3には、トレーストリガーもトレースフィルターもありません。トラブルシューティングに必要なプログラムセグメントのみの記録を選択することはできません。これは、解析に必要な領域をカバーするために、比較的長い期間トレースデータを収集する必要がある可能性があることを意味しています。この場合は、STREAMモードが最適な選択です。ただし、STREAMモードでは、デバッグ環境に高い要件があります:

- ストリーミングの結果取得される大量のデータには、64ビットTRACE32実行可能ファイルが必要です。これは、収集される多数のトレースエントリのアドレスレンジを許可するために必要となります。
- CombiProbeとホストコンピュータ間の転送速度は、データ損失なしにすべてのトレースデータをストリーミングするのに十分な速度である必要があります。CombiProbeの128メガバイトのメモリは、トレースポートからのロードピークを和らげるために使用されます（TPIU）。

リアルタイムプロファイリングは、特にステートメントカバレッジおよび条件カバレッジの実行に適しています。カバレッジ解析は、画面上でリアルタイムに追うことができ、テスト結果は直ちに表示されます（図17を参照）。「OK」とマーキングされた行は、すでにカバーされています。


図17: リアルタイムプロファイリングでは、コードカバレッジ解析を画面上でリアルタイムに追うことができる

# 近づくシミュレーションと実動作

現在では、ハードウェアにコミットする前に設計のシミュレーションおよび検証を実行することが一般的です。これは、制御エンジニアリングの市場に、MATLAB®およびSimulink®などのツールが開発ソフトウェアとして浸透したためです。設計を最終決定する前に、多くの変数の影響について制御ループをテストすれば、多くの時間と手間を省くことができます。

シミュレーションを通じて制御アルゴリズムが特定された後の、次のステップは何でしょうか。また、このソリューションはどのように制御ソフトウェアに統合されるのでしょうか。この点について、Simulinkを使用すれば、コードを自動的に生成できます。しかし、プログラムがシミュレーションと同様に制御ハードウェア上で動作するのか、という疑問は残ります。

## 検証方法

ミュンヘン工科大学のInstitute of Flight System Dynamicsでは、Diamond DA42の飛行制御システムの開発中に興味深いソリューションを発見しました(図20を参照)。
Simulinkで制御アルゴリズムを作成し、機能をテストした後は、制御ハードウェアのプロセッサに対応するプログラムコードが、組み込みコーダーを使用して制御ブロックから生成されました。生成されたコードは、TRACE32デバッガを使用して制御ハードウェアにロードされ、その機能はin-situでテストされました。

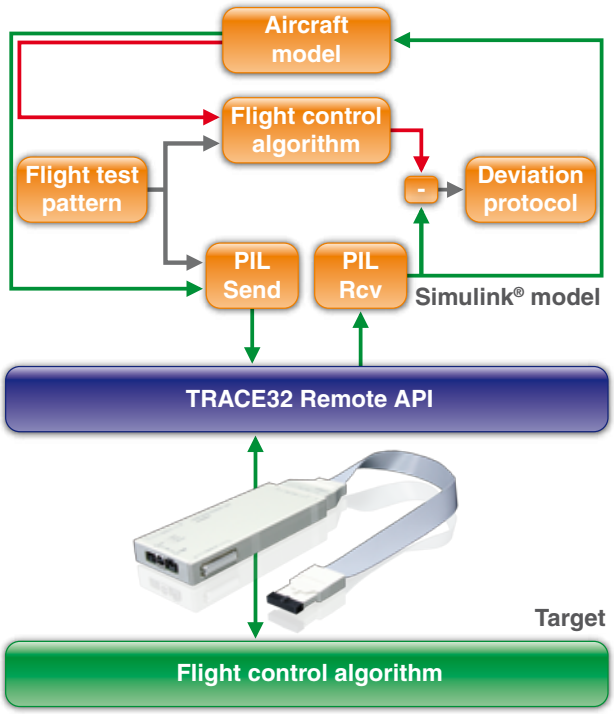



図18: 実際の制御動作(緑の線)とシミュレーションされた制御動作(赤の線)の比較

シミュレーションされた制御動作(赤の線)と実際の制御動作(緑の線)との逸脱の程度を特定するため、またそれよりも制御ハードウェアの数値的精度を確認するために、Processor-In-the-Loop(PIL)シミュレーションが選択されました(図18参照)。基本的に、PILシミュレーションは特別に開発されたSimulinkブロックの「PIL Send」および「PIL Receive」に基づいています。これらのブロックは、SimulinkとTRACE32リモートAPI間の通信を実装するために設計されました。

実行ごとに、飛行制御アルゴリズムは、ターゲットハードウェア上で離散時間飛行制御の1つの計算ステップを実行します。必要な入力パラメータは、Simulinkモデルにより提供されます。計算された値はSimulinkモデルに返され、そこで航空機モデルを提供します。並列計算では、シミュレーションされた飛行制御アルゴリズムにより同じ値が算出されます。その後、2つの結果を比較するために差が使用されます。

スタンドでのテストの結果は、絶対偏差10-13になりました。この方法で、高い整合性が的確かつ簡単に証明されたことになります。

ミュンヘン工科大学のInstitute of Flight System Dynamicsのプロジェクトの詳細情報については、www.lauterbach.com/intsimulink.htmlを参照してください。

## Simulink®とTRACE32の統合化

ローターバッハは、ドイツ/ニュルンベルクで2012年2月に開催されるEmbedded Worldショーで、Simulinkとローターバッハ製TRACE32デバッガの密接な組み合わせについてのプレゼンテーションを行います。

ローターバッハは、Simulinkコード生成のプロパティを使用していますが、これはコードブロックが常にブロックの名前とモデルパスが含まれるコメント行で始まります。これらのコメント行は、生成されたコードがTRACE32デバッガにロードされてから使用可能になります。これらのラインを使用することで、Simulinkブロックとソースコードの行との単純相関が許可されます。

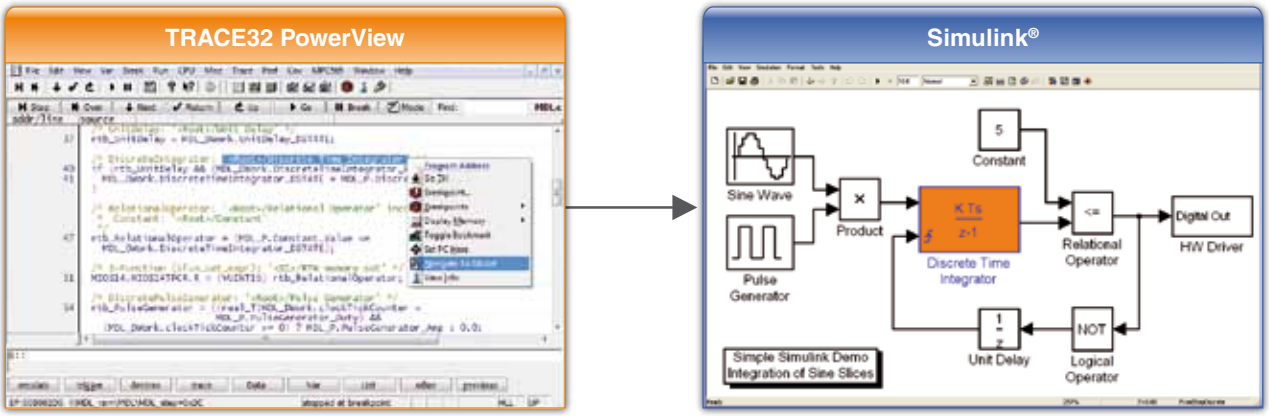



図19: 選択されたソースコード行に属するブロックがSimulinkでマークされる

### Simulink® から TRACE32へのナビゲーション

TRACE32のグローバルメニューおよびブロック/信号用TRACE32メニューは、「Simulinkカスタマイズメニュー」としてSimulinkに統合されます。TRACE32デバッガは、これらのメニューを使用してSimulinkから制御できます。次の機能を使用できます:

- TRACE32のコード表示
- 信号用のTRACE32[Variable Watch]ウィンドウの起動
- TRACE32デバッガに組み込まれたSimulinkのロード
- ブロック/信号ブレークポイントの設定および管理
- 制御ハードウェアでのプログラムの起動と停止

### TRACE32 から Simulink® へのナビゲーション

TRACE32デバッガでソースコードのセクションを選択すると、Simulinkで対応するブロックがマーキングされます(図19を参照)。

## 今後の予定

Simulink Release 2012aがリリースされると、Simulinkでより多くのTRACE32の機能を利用できるようになります。ローターバッハは、Simulink rtiostream APIの実績のある機能を使用して、PILシミュレーション、データロギング、およびパラメータ調整を統合します。

MATLAB® および Simulink® はThe MathWorks, Inc.の登録商標です。.


図20: Diamond DA42(出典: www.diamond-air.at)

# TRACE32を使用したUEFI BIOSのデバッグ

**Atom™デバッガ向けのTRACE32の新拡張機能は、Insyde社製H2O UEFI BIOSのすべてのデバッグ機能提供します。**

UEFIは、従来のPC BIOSの後継です。これは、ファームウェアとブートプロセスを管理するオペレーティングシステム間のインタフェースとして機能します。電源オンからオペレーティングシステムへの管理の移行までの間に、UEFIは様々な、明確に区分されたフェーズを実行していきます（図21を参照）。TRACE32はJTAGベースのツールであるため、リセットベクターから開始するデバッグが可能です。

ブートプロセスの各フェーズで、PowerViewユーザーインタフェースには、UEFI独自の情報を示す専用ウィンドウが表示されます。関数および準備されたスクリプトにより、ダイナミックにロードされたドライバのデバッグを、最初の命令から開始できます。新しいUEFI拡張機能の詳細については、www.lauterbach.com/uefi.htmlを参照してください。.

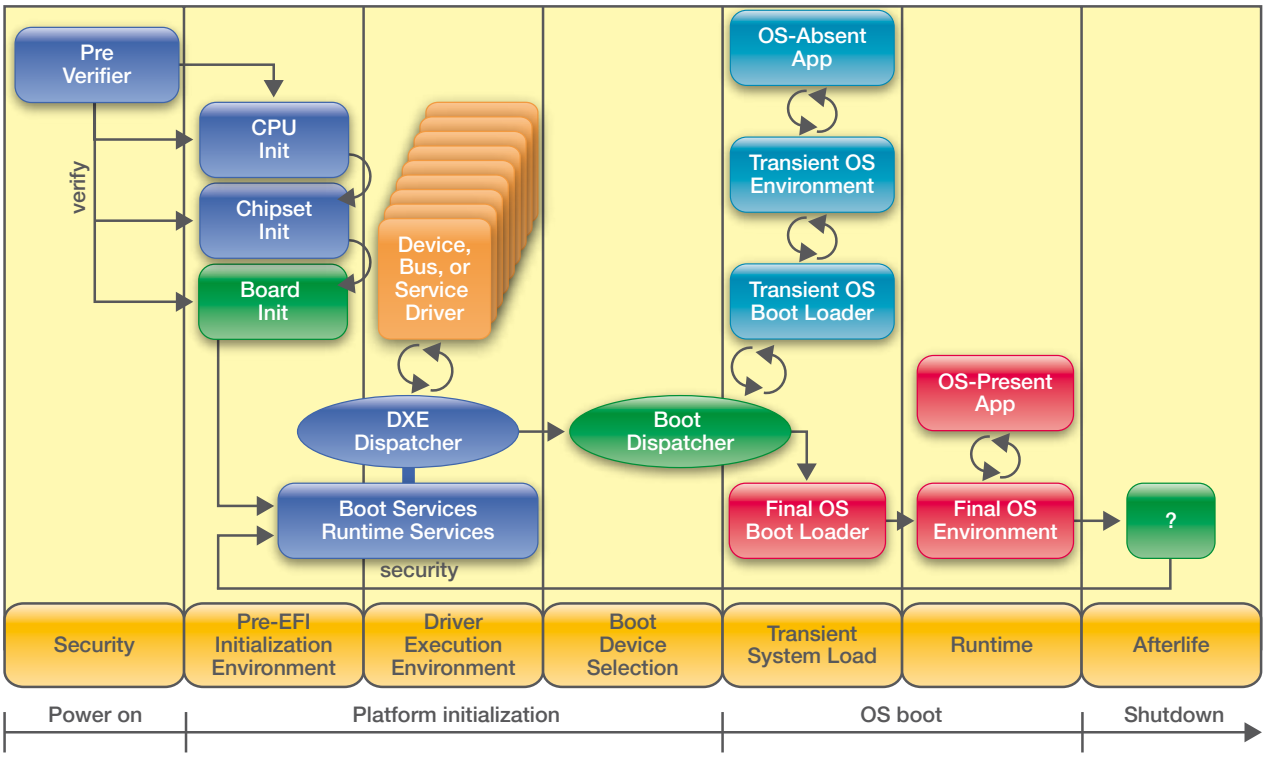



図21: UEFIでのシステムブートプロセス

## 世界の支社

- 日本
- ドイツ
- フランス
- イギリス
- イタリア
- 中国
- アメリカ

その他の国々でも経験豊富な販売代理店が対応させていただきます

## 連絡先更新のお願い

アドレスの変更、またはメーリングリストからの退会をご希望の場合には、下記へご連絡ください。


info@lauterbach.co.jp