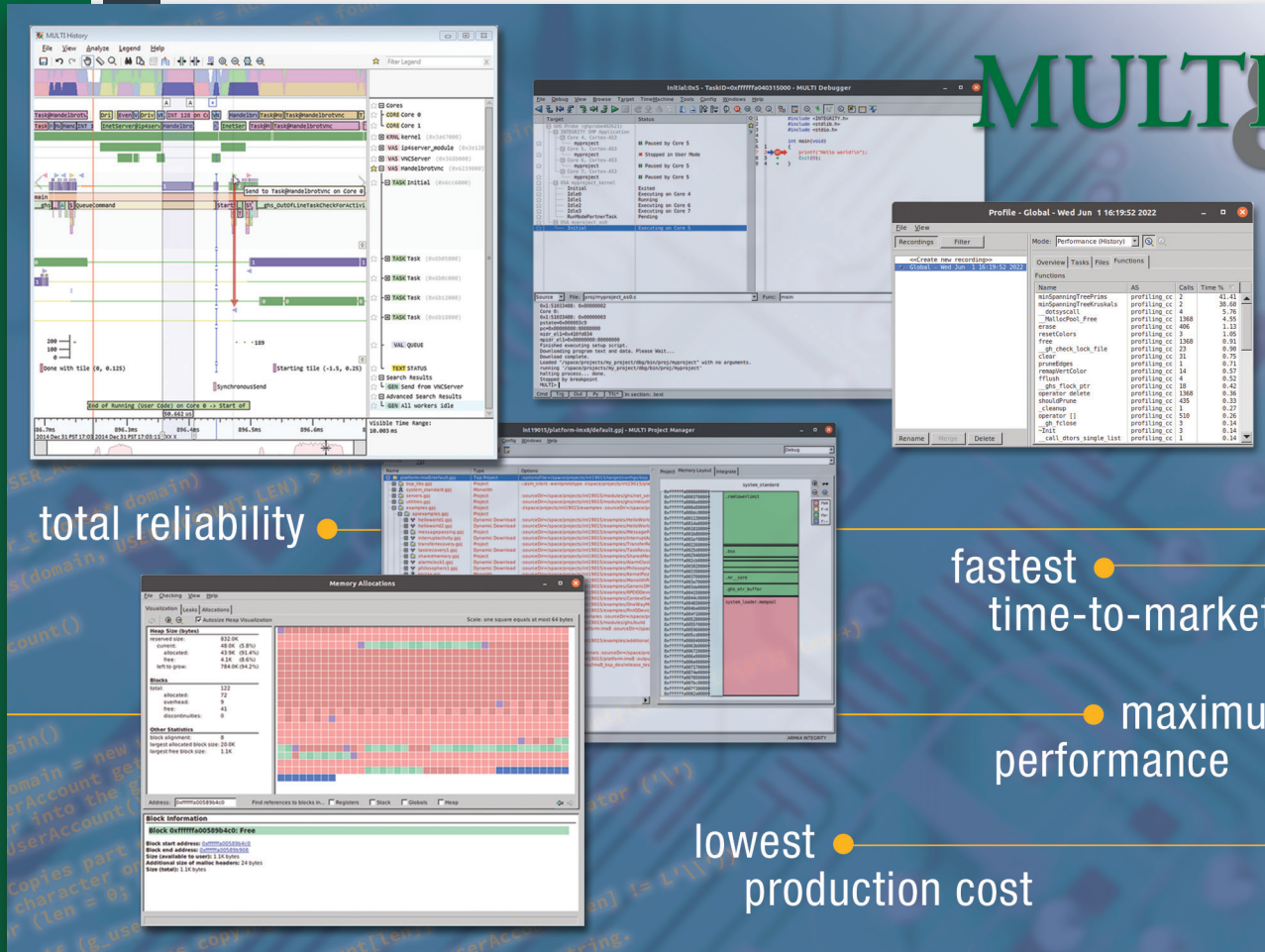


MULTI⁸



total reliability

fastest
time-to-market

maximum
performance

lowest
production cost

MULTI⁸ 統合開発環境

Green Hills[®]
SOFTWARE

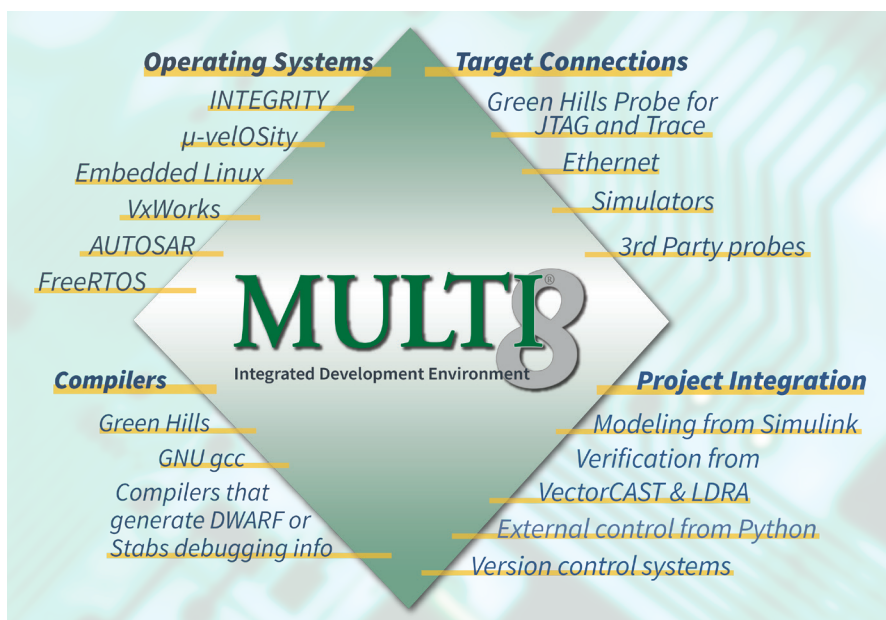
MULTIに含まれる一連のツールは、Green Hillsの開発者により、組み込みソフトウェア開発者のニーズに合った完全な環境を実現するために作られました。

- ▲ 安全認証済 C/C++ コンパイラとランタイムライブラリ
- ▲ マルチコアデバッグ
- ▲ アドバンスドシステムビューアー
- ▲ リバースプレイデバッグ
- ▲ OS認識
- ▲ 静的コード解析
- ▲ ランタイムエラー検知
- ▲ メモリリーク検知
- ▲ パフォーマンスプロファイラ
- ▲ エディタ
- ▲ 命令セットシミュレータ
- ▲ コードカバレッジ
- ▲ プロジェクトマネージャ
- ▲ バージョン管理インテグレーション
- ▲ フラッシュプログラマ
- ▲ Pythonインターフェース
- ▲ MATLAB/Simulinkインテグレーション

組み込みソフトウェア開発者の生産性を向上させるツール

30年以上にわたって何千人もの開発者に使用されてきたMULTIは、組み込みシステムのコードの作成、デバッグ、最適化に特化されており、業界において比類のない統合開発環境です。MULTIを使用することで、開発者は難しいバグの検知と修正、パフォーマンスのボトルネックの特定、そして将来の問題の発生を防止できます。主な利点として、納期内にソフトウェアをリリースするためのコストの削減、低いソフトウェアの品質による費用のかかるリコールからの回避、そして効率的なコードによって可能となる省メモリ、低CPU使用率によるプロセッサとメモリのコスト削減です。

- ▲ バグの早期発見と修正
- ▲ リプレイによる問題特定
- ▲ 実行前にバグを発見
- ▲ 複雑なシステムの解析と最適化
- ▲ パフォーマンスのボトルネックを特定・修正
- ▲ コードの安全性と信頼性を向上
- ▲ パフォーマンスの最適化
- ▲ プロセッサのコスト削減
- ▲ 機能安全要求への対応
- ▲ Simulink®やその他のモデルの実行



MULTIは、一般的なプロセッサ・アーキテクチャをすべてサポートし、LinuxまたはWindowsデスクトップまたはクラウド環境で動作し、複数のコンパイラからのコードをデバッグすることができます。様々なリンクを介してターゲットに接続し、シリコン、シミュレータ、エミュレータ上で動作する複数のオペレーティングシステムをサポートします。

バグの発見と修正のスピードアップ

複雑なリアルタイムシステムでは、散発的で深く埋もれたバグの探索やパフォーマンスのボトルネックの特定に予想外の時間がかかり、時には数週間から数ヶ月消費してしまう可能性があり、これらは予定通りの製品発売へのリスクとなりうるものです。

MULTIでは、これまで数日、数週間、数ヶ月かかっていた問題が、数時間、あるいは数分で発見でき、修正できます。複雑なシステムにおいて、最も時間のかかるバグは、散発的で隠れたものであり、コードを見ても発見困難・不可能なものです。MULTIを使用することで、開発者はメモリ破壊、意図しない依存関係、予期しない割り込み、マルチコアの複雑さ、タスク間の破損、ソースコードの欠落や仮想化による不透明さ、リアルタイム要件の見落とし、非同期ハードウェアイベントなどが原因の問題を迅速に特定できます。

以下は、MULTI Debuggerの有効性と、より低い性能のツールでは達成できない困難な問題を解決する能力を強調するお客様からの体験談です。

- ▲ MULTIを使用することで、2人月かけて探したバグを2時間で発見し、修正することに成功しました。
- ▲ MULTIを数分間使用しただけで、起動コードに予期せぬ割り込みが入り、ターゲットが頻繁にハングアップする原因となっていた3ヶ月前のバグを発見しました。

- ▲ 車載用オーディオ・システムで散発的に発生するポップノイズの原因を6カ月も突き止められずにいました。量産期限が迫っていく中、ソースコードがない状態でも、Green Hillsのシステム可視化ツールHistory® viewerは、予期せぬ実行フローを瞬時に表示し、数時間の分析の結果、グラフィックアプリケーション内の根本原因を突き止めました。エンジニアへ原因を指摘後、この問題は数分で解決されました。

MULTI Debuggerは以下の3つの強力な機能を備えています：

- ▲ **History viewer** プログラムがどのように現状に至ったのか、またシステムが何を実行していたかを表示します。
- ▲ **TimeMachine®** トレースデータを利用し、デバッグを通して時間を巻き戻し実行、ステップ、および分析することができます。
- ▲ **Debug Snapshot** デバッグセッションを保存し、世界中の同僚と簡単にデバッグセッションを共有することができます。

複雑なシステムを理解する

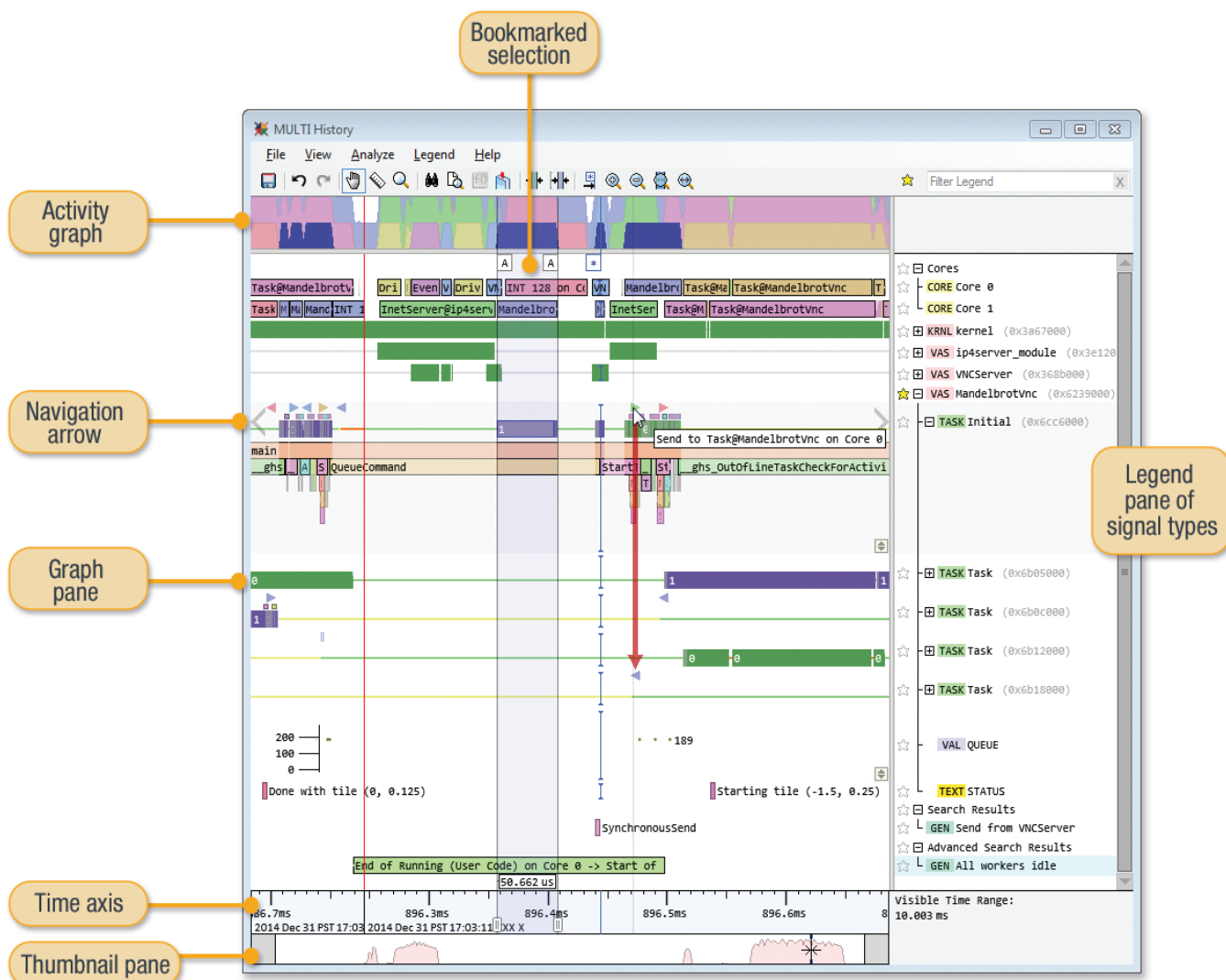
- ▲ プログラムはどのようにしてここにたどり着いたのか？
- ▲ どこに時間がかかっているのか？
- ▲ プログラムは予期せぬ動作をしていないか？

この3つの質問は、コードのデバッグやシステムの高速化の基礎となるものです。History viewerは、これらの問いに答えるものです。

History viewerは、複雑なヘテロジニアス・マルチコア・システム上で実行されたプログラムの過去数秒、数分あるいは数日にわたる記録を、自然で直感的なGUIで表示します。これは不透明なハードウェアおよびソフトウェアシステムを明確に、完全に把握することができる初めてのシステムです。プロセッサの動作をマイクロ秒レベルで深く掘り下げること、数分、数日にわたるシステムの動作をズームアウトして見ることも可能です。この新しい可視化機能により、困難なバグを瞬時に発見し、隠れたボトルネックや依存関係を確認し、実行時間を分析することができます。

History viewerの機能

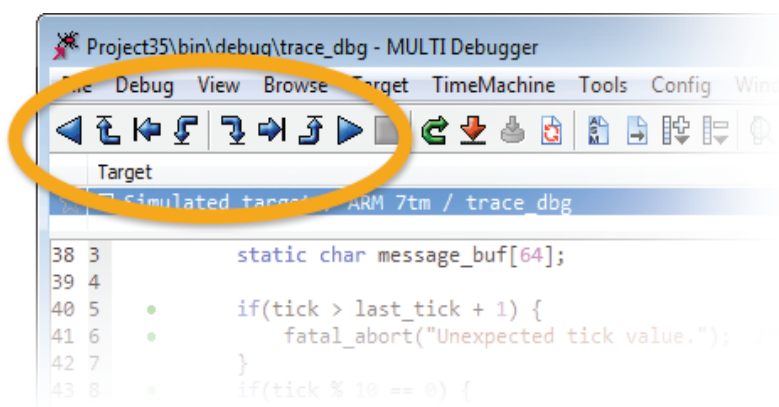
- ▲ MULTI Debuggerを停止させ、任意の時点までのシステム履歴をHistory viewerで表示することができます。
- ▲ タスクのコールスタックを表示することにより、プログラムの流れを把握することができます。プログラムがどのようにしてここに至ったのか、コードが予測される経路からどこで逸脱したのかを確認することができます。
- ▲ 予期しない実行フロー、誤ったタスクの優先順位指定、過剰な割り込みレベル処理、OSへの過剰なシステムコール、異常な割り込みレイテンシなど、マルチコアシステムでのバグやボトルネックの原因を明確に把握することができます。
- ▲ ズームイン機能でマイクロ秒単位のOSイベントと依存関係を調査・測定したり、ズームアウト機能でシステムレベルの実行パターンを確認することができます。
- ▲ History viewer画面内をクリックすると、対応するソースコードの行を素早くデバッグで表示します。
- ▲ カーネル実行、仮想アドレス空間、マルチコアCPUの使用率、コンポーネント間通信、正確なタイミング、ユーザ定義イベントなどを視覚的に確認することで、複雑なプログラムの理解に新たな視点を与えます。
- ▲ シミュレータやボード上で実行されている数テラバイトのトレースデータやインストルメントドコードを取り込むことができます。
- ▲ Linux ftraceやCOVESA DLT (診断ログ&トレース) など、サードパーティのログデータも表示可能です。
- ▲ 低オーバーヘッドなユーザ定義変数をほぼ無制限に注入し、History viewerでログを表示することができます。
- ▲ 高度な検索オプションにより、複雑な条件を満たす実行を見つけ出し、分析することができます。
- ▲ 分析上重要な箇所にブックマークや注釈を付け加えて、システムのHistory viewer snapshotファイルを保存し、それを他の開発者、テストエンジニア、テクニカルサポートへ送信することで、システム全体の情報を即座に共有し、デバッグを継続することができます。
- ▲ 非常に高い解像度でイベントの時間を測定します。
- ▲ ソースコードがない場合や、仮想マシン上でソフトウェアが動作している場合にも対応できます。
- ▲ LinuxやAndroidの内部を深く見るすることができます。
- ▲ プロセッサのオーバーヘッドを最小限に抑え、ハードウェアを追加する必要がありません。



History viewer は、複雑なヘテロジニアス・マルチコア・システムにおいて、過去数秒、数分、数日間のプログラム実行を表示することにより、これまでにないシステムの可視性を提供します。

リバースプレイデバッガ

TimeMachineは、実際のプログラム実行データを自動的に取得することで、デバッガがHistory viewerに表示された問題箇所まで遡ってコードを実行、ステップ実行、デバッグすることを可能にします。また、Profiler などの他のツールも強化されています。



TimeMachineは、MULTI Debuggerを使用して、プログラムを時間的に前方および後方に実行、ステップ、アライメントし、問題の根本原因を突き止めることができます。

バグの発生を抑える

DoubleCheck はGreen Hillsの統合静的ソースコードアナライザで、プログラムを実行する前からプログラミングエラーを特定することで、時間とコストを節約します。DoubleCheckは Green Hillsの最適化コンパイラの一部であるため、セットアップの手間がかからず、使用するためのオーバーヘッドも最小限です。オプションを設定するだけで、コンパイルされるたびにDoubleCheckが自動的にソースコードをレビューします。コンパイラの一部であるため、DoubleCheckは従来の静的解析ツールよりもはるかに高速に動作し、すべての開発者が常にオンにしておくことが可能です。

さらに、DoubleCheckは従来のコードレビューよりも信頼性が高く、通常のシステムテストでは見つかるのが困難なバグも発見することができます。

Run-time Error Checkは、静的解析だけでは特定できないバグを見つけ出すことで、DoubleCheckを補完します。特にメモリ破壊によるバグは凶悪で発見が難しく、長い間コード内に潜むことがあります。Run-time Error Checkを使えば、問題が発生した時点でその原因を警告されます。Run-time Error Checkは、ポインタによる無効なメモリアクセス、範囲外の値の割り当て、switch文での未処理のケース、ゼロ除算などの問題をランタイム時にを見つけ出します。

クリーンで簡潔なコードは、エラーを含む可能性が低く、テスト、理解、変更が容易です。これらの要素は全て、バグの減少や信頼性の向上に寄与します。Green Hillsの最適化コンパイラは、安全なプログラミングのため100以上のルールを含む**MISRA 2012および2004**コーディングスタンダードなどの業界基準によって定義されたクリーンコーディング規約を実施することができます。また、お客様の要件に合わせて規約の一部のみを適用することも可能です。

GHS Standard Modeは、通常のCやC++よりも厳しいコーディング基準を適用するコンパイラの警告やエラーの集合体です。これは、コードの信頼性を高めるためにGreen Hills が長年にわたって培ってきた経験に基づいています。Green Hills 社内でも利用されており、誤検出をあまり起こさずに、一般的なミスや悪いコーディングスタイルからコードを保護します。

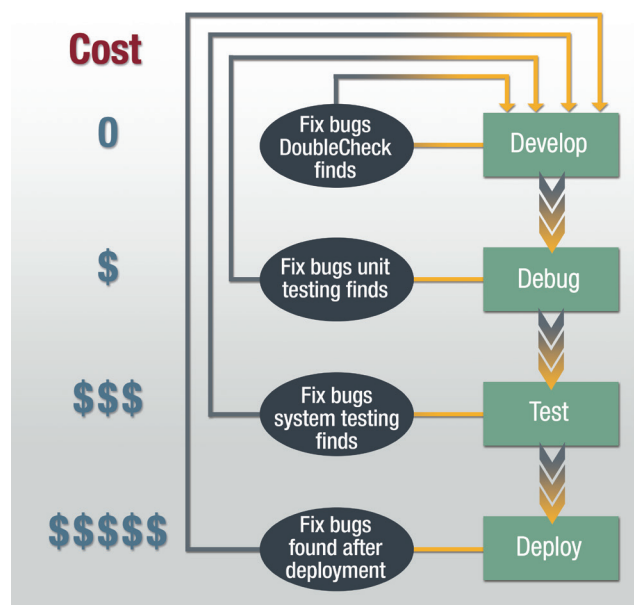
高度なデバッグ

MULTI Debuggerは、複雑なヘテロジニアス・マルチコアターゲットプロセッサやシミュレータ上で実行しながら、ソースコードを調査、監視、変更するための強力なツールです。TimeMachineを使用すれば、時間を巻き戻して実行することも可能です。デバッグはMULTIの他のツールとシームレスに統合されており、History viewerなどの様々なMULTIツール内でクリックすることで呼び出すことが可能です。

- ▲ 様々なOSやボード上のアプリケーション、カーネル、デバイスドライバコードをデバッグ
- ▲ 仮想OSカーネルとそのアプリケーションをデバッグ
- ▲ 対称型(SMP)および非対称型マルチコア(AMP)のランモードおよびフリーズモードのデバッグとランコントロールをひとつのデバッグで実現
- ▲ コールスタック、キャッシュ、オブジェクト、メモリ、そしてレジスタの閲覧
- ▲ **OSA Explorer**を通して、カーネルオブジェクト、タスク、リソースのステータスを確認
- ▲ タスクのインタラクションをソースレベルでデバッグ
- ▲ 強力な実行機能とデータブレークポイントを駆使して、データポイントを特定
- ▲ 仮想アドレス空間内で動作するコードをデバッグ

MULTI Debuggerを使えば、マルチタスク、マルチコアシステムのデバッグがシングルコアシステムと同じように簡単にできます。

INTEGRITY®やLinuxなどのOS上でマルチタスクアプリケーションをデバッグする場合、MULTIはランモード、フリーズモード、または両方のモードで同時に複数のタスクに操作することができます。ランモードでは、OSカーネルを実行し続けた状態で、個々のタスクを停止してデバッグすることができます。フリーズモードでは、タスクを調査する際にターゲットシステム全体を停止します。

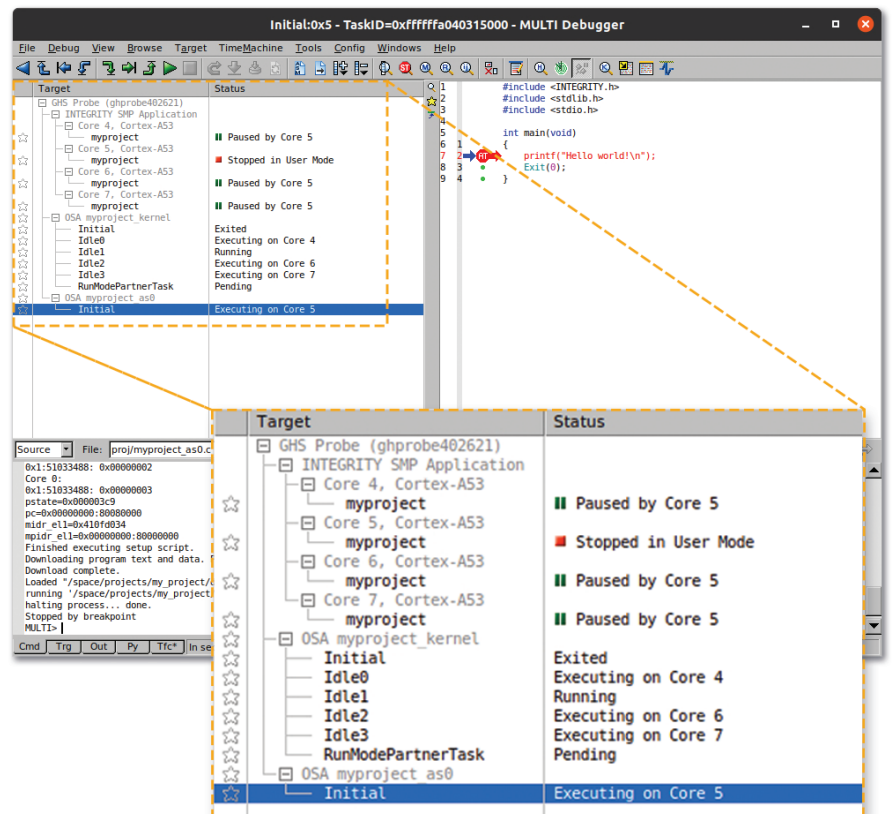


DoubleCheckはシステムの開発段階で問題を早期に発見し、時間とリソースを節約することで、開発コストを削減します。

その中でも特に重要なのが同期実行制御(synchronous run control)で、いずれかのコアがデバッグ状態になったときに、すべてのコアを一括して停止させます。例えば、あるコアがブレークポイントに到達すると、ターゲットリストは下記の情報を明確に表示します：

- ▲ ブレークポイントに到達したコア
- ▲ その時に実行されていたスレッド
- ▲ システムの他のコアが実行していたタスク

通常のツールでは、マルチコアターゲットへのコードのダウンロードは複雑で冗長なプロセスであるケースが多々あります。MULTIは、これらのターゲットへのINTEGRITY RTOSのダウンロードを自動的に処理することができます。他のオペレーティングシステムやアプリケーション向けには、シンプルな設定ファイルを使用して、ダウンロードとセットアップのプロセスを自動化することができます。



MULTI同期実行制御 (synchronous run control)は、マルチコアシステム内のすべてのコアを同時に停止させます。これにより、他のコアで実行中の操作が共有メモリに影響を与えることを心配することなく、1つのコアをデバッグすることができます。

組み込みLinuxのデバッグ

MULTI for Linuxは、組み込みLinuxソフトウェアを開発するエンジニアに高度なデバッグ機能を提供します。これにより、エンジニアの生産性が劇的に向上し、より信頼性が高く、より高性能な製品をより早く市場に投入することができます。

従来のLinux用デバッグツールは、限られた視認性と有用性しか提供しません。GDBやEclipseは、セットアップが複雑で、使いにくいという反応が鈍く、最適化された組み込みコードをデバッグする際には必ずしも信頼できるものではありません。その結果、開発者はしばしば効果に当たり外れのあるprintfやprintkコマンドに頼らざるをえません。しかし、残念ながら、性能に大きな影響を与えることもあり、これらのテクニックは独自の問題を引き起こし、複雑なシステムを理解するには取るに足らない情報しか得ることができません。

MULTI for Linuxと Green Hills のコンサルティングサービスにより、Linuxカーネル、カーネルスレッド、割り込みサービスルーチン、ユーザモードスレッド、プロセスなど、組み込みLinuxシステムのあらゆる側面を、これまで以上に迅速かつ容易に可視化し、デバッグすることが可能となります。

Linux デバッグ機能

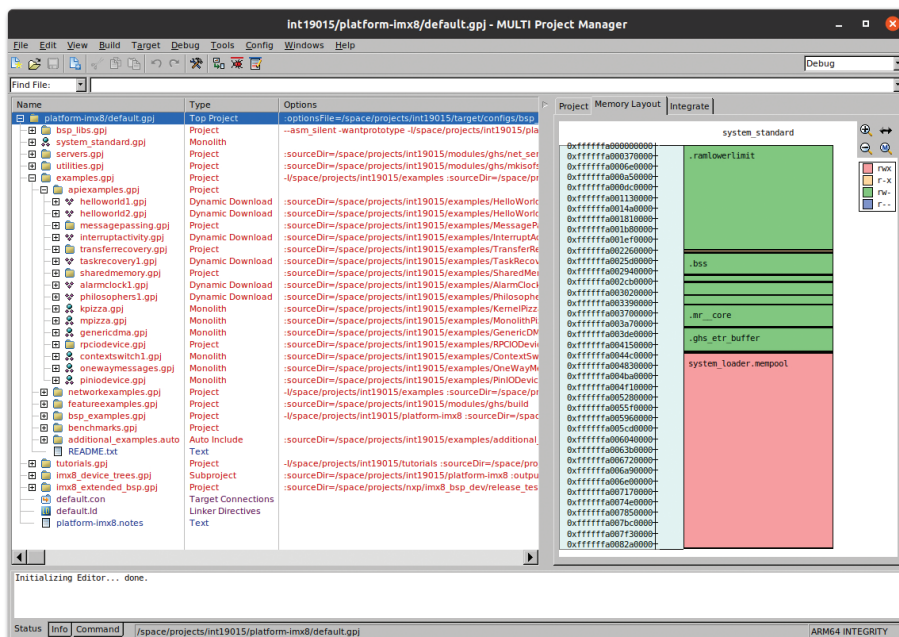
- ▲ 共有メモリ領域、セマフォ、メッセージキュー、カーネルモジュールなど、Linuxカーネルを包括的に把握
- ▲ SMPマルチコアシステムのデバッグをサポート
- ▲ カーネルコード、ドライバ、ユーザスペースアプリケーション、仮想化コードのデバッグを1台のデバッグで実現
- ▲ ターゲット上の組み込みLinux、INTEGRITY Multivisor®仮想化ソリューション上で実行している仮想化Linux、そしてデスクトップ上のネイティブLinuxもデバッグ可能
- ▲ 仮想化LinuxとINTEGRITYのイベントをHistory viewerで閲覧
- ▲ ロード可能なカーネルモジュールやデバイスドライバのデバッグが可能
- ▲ Linuxのプロセス起動コードのデバッグの簡易化
- ▲ カーネルコードにスレッド毎のブレークポイントが設置可能
- ▲ 仮想アドレス空間内の共有オブジェクトをデバッグ
- ▲ システムのパフォーマンスへの影響を最小限に抑えたブレークポイント処理
- ▲ プロセス間のマルチスレッドインタラクションのデバッグ

プロジェクト管理に役立つIDE

MULTI は、プロジェクト管理を簡素化することで、貴重な時間をより多く製品開発のために割くことができます。Builderを使えば、複雑なビルドインフラを維持する必要がなく、ビルド構成を簡単に変更することができます。Builderは、プロジェクト全体の依存関係を自動的に分析し、最新のマルチコアシステムを最大限に活用し、できるだけ多くのファイルを並行してコンパイルおよびリンクします。

Builderに加え、シームレスに統合された Project Manager、エディタ、フラッシュプログラマ、命令セットシミュレータにより、ビルド管理ではなく、コードの開発に時間をかけることができます。

MULTIは、CI/CDプロセスの一環として、コンパイラ、静的ソースコードチェック、シミュレータをローカルまたはクラウド環境で実行できるため、お客様のソフトウェア開発ライフサイクルプロセスを問わずに、コンパイルおよびテストフェーズにおけるチームの効率を向上させます。



MULTI Project Managerは、アプリケーションの構築と更新を簡素化します。タブ付きブロック図とメモリレイアウトビューは、アプリケーションの構造を一目で理解することができます。ここで、緑は読み書き可能なメモリ、赤は読み取り専用メモリを意味します。

パフォーマンスの最大化

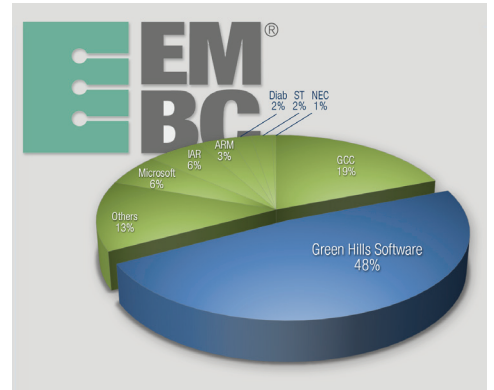
最高のパフォーマンスを必要とするアプリケーションのために、MULTI は、最適なコンパイラと最高のパフォーマンス解析ツールを兼ね備えています。最高のパフォーマンスを実現するために、Green Hills のパフォーマンス チューニングのエキスパートにお問い合わせください。お客様のデバイスに合わせてカスタマイズしたソリューションを実現します。

より速く、より小さなコードを生成

優れたコンパイラは、プロセッサの性能を最大限に発揮するのに不可欠な要素であり、C/C++最適化コンパイラである Green Hills Compiler は業界で最も優れたものです。組み込み業界で広く受け入れられているベンチマークであるEEMBCベンチマークにおいて、Green Hills Compilerは常に競合他社のコンパイラを凌駕し、32ビット・64ビットプロセッサ用では、最高速かつ最小のサイズのコードを生成します。

Green Hills Compiler は、何百もの最先端のコンパイラ最適化を使用して、プログラムの実行速度を大きく向上させ、サイズを削減しています。

Green Hills Compiler は、デフォルトの最適化と設定の最も優れた基準となっていますが、アプリケーションにはそれぞれ独自性があることも理解しています。さまざまな最適化を組み合わせ、プロジェクト単位、ファイル単位、関数単位、さらにはソース行単位で適用することで、コンパイラの出力を細かく調節できます。Green Hills のコンサルティングサービスチームは、お客様がどのようなプロジェクトでも最高のパフォーマンスを発揮できるよう、柔軟性の高いGreen Hills Compilerを活用するお手伝いをします。今までの多くのプログラムにおいて、Green Hills のコンサルティング・サービス・チームはGreen Hills Compilerを使用することでプログラムを最適化し、競合コンパイラよりも20%もの高速化を実現しています。

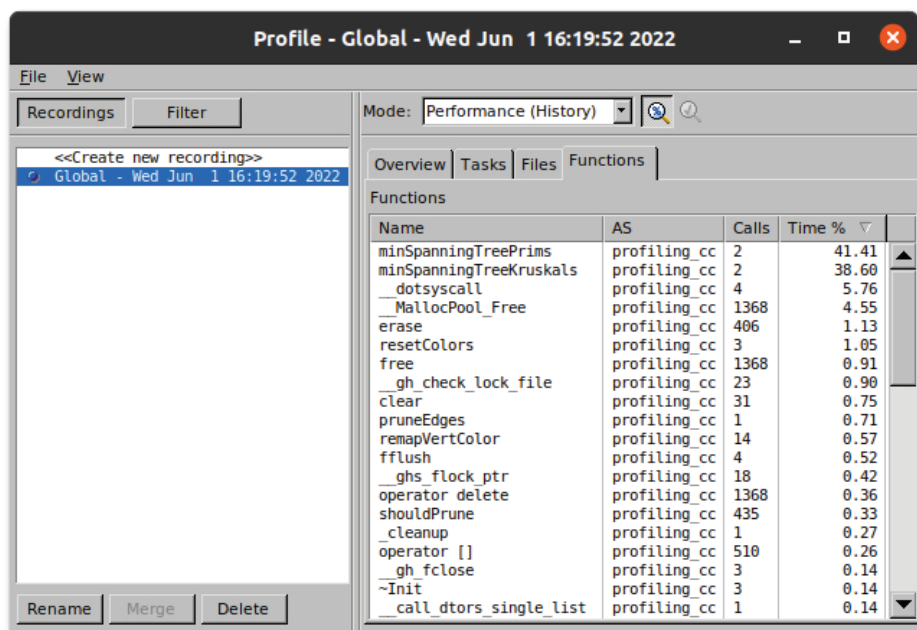


プロセッサメーカーは、実世界のアプリケーションにおけるプロセッサの性能を示すために、EEMBC認定のベンチマーク結果を公開しています。プロセッサメーカーの多くは Green Hills Compilerを使用したものを最高スコアとして公開しており、その比率は他社の2倍以上にもなります。

パフォーマンスのボトルネックを特定

プログラムのパフォーマンスを最大化させるための2つ目の重要な要素は最高のパフォーマンス解析ツールを使用することです。History viewer と Profiler ツールを活用すれば、コード内のパフォーマンスのボトルネックになっている部分を簡単に特定して排除できます。その結果、より速く、より効率的なアプリケーションを実現することができます。

最高性能を求めるなら、時間が最も重視すべき要素となります。また、これはほとんどのプロファイリングツールが投出しがちな要素でもあります。革新的なHistory viewerは、プログラム内で実行される各関数を時系列で分かりやすく表示でき、プログラムフローのパターン特定や、時間がかかっている場所の見極めを簡単に行うことができます。



Profilerのインターフェースはさまざまなタスク、関数、命令の実行頻度および時間を表示することで、解析を効率化し、どの部分のコードに最適化やテスト作業を集中して行うべきかを特定することができます。

Profilerの性能

強力なProfilerは、トレースまたはインスツルメンテーションされたコードから収集されたデータを使用して、プログラムの動作に関するさらなる分析結果を提供します。各タスク、関数、ソース行、アセンブリ言語命令ごとに、プロセッサ時間の消費量を示し、以下が容易に行えるようになります。

- ▲ パフォーマンスの問題点の特定
- ▲ カバレッジギャップを見つけ出し、特定のコードのテストカバレッジを追跡
- ▲ ユーザ間でのプロファイルデータ共有
- ▲ 複数のテストまたプロファイルの記録からのプロファイルデータ統合

History viewer のグラフペイン、アクティビティグラフペイン、サムネイルペインでは、プログラムがどこに時間を費やしているのか、これまででない視点で見ることができます。最適化作業でパフォーマンスの問題を特定する場合、History viewer は数日から数週間を節約することができます。

- ▲ RTOSイベントの長さや頻度を表示し、どのような操作に最も時間がかかり、どこに最適化を図るべきかを明らかに
- ▲ タスクの優先順位の誤り、過度の割り込み処理、オペレーティングシステムへの過剰な呼び出し、予期せぬタスクコンテキストスイッチなどの発見
- ▲ コアの実行内容やタスク間の関係を直感的に表示



30年にわたって、ツールの開発と使用を通じてお客様の問題を解決してきた経験から、熟練のエキスパートが、あなたのアプリケーション性能と信頼性を最大限高めるお手伝いをします。

豊富な知識によるメリット

可能な限り最高のパフォーマンスを達成するには、Green Hillsのパフォーマンスチューニングのエキスパートにお任せください。Green Hills のカスタムサービスでは、開発ツールやコンパイラを最大限に活用し、設計を評価し、さらにお客様の特定の目標に合わせたカスタムコンパイラの最適化を作成することができます。

生産コストの削減

大量生産においては、単価を数円下げるだけでも全体的なコスト削減額は莫大なものになります。MULTIは、メモリとプロセッサのコストを大幅に削減し、収益を向上させることができます。

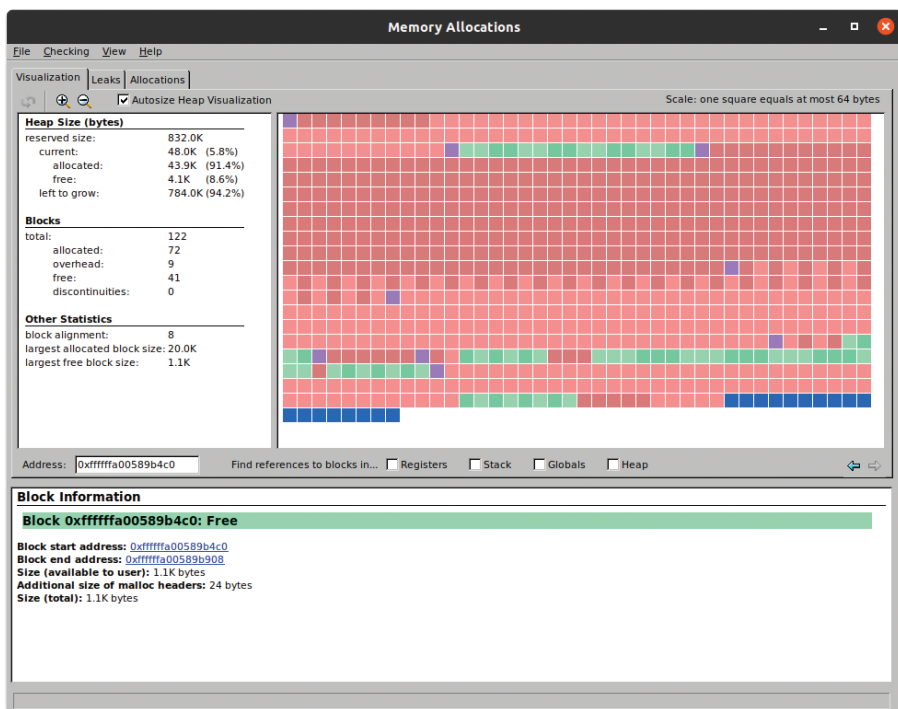
プロセッサコストの削減

増大する性能要件を満たすために、必ずしもより高速なプロセッサが必要というわけではありません。Green Hillsの優れたC/C++最適化コンパイラは、使用するプロセッサの性能を最大限に引き出すことができます。さらに、History viewerやProfilerなどの性能分析ツールによって、プログラムがどこに時間を費やしているかを鳥瞰図的に把握できるので、最適化の作業に集中することができます。同じリアルタイム要件を満たすために、より低価格のプロセッサを使用することができれば、どれだけのコスト削減が可能でしょうか？

コードのパフォーマンスを向上させることで、設計のCPUヘッドルームに余裕が生まれ、より多くの機能を追加できるようになります。さらに、消費電力や冷却機能を抑えた低コストのプロセッサを選択することも可能となります。

メモリコストの削減

Green Hills のコンパイラによる最適化は、実行ファイルのサイズを小さくすることでメモリコストを削減します。ほとんどのプログラムでは、競合コンパイラと比較して少なくとも10%の改善が見られます。さらに、DoubleCheckとMULTI DebuggerのMemory Allocations ツールは、メモリをより効率的に使用し、組み込みシステムに必要なメモリ量をさらに削減するのに役立ちます。



MULTI DebuggerのMemory Allocationsツールを使えば、ランタイム時にメモリリークを簡単に検出することができます。

機能安全への自信

MULTI と C/C++コンパイラツールチェーンは、最高レベルの安全性を満たしていることが証明されています：

- ▲ ISO 26262 (自動車)、IEC 61508:2010 (産業)、EN 50128:2011 および EN 50657 (鉄道) 機能安全準拠
- ▲ TÜV NORDとexidaの両社から証明書を取得
- ▲ 最高到達レベルSIL 4 (安全度水準)、ASIL D (自動車安全度水準) を満たし、C/C++ランタイムライブラリも認証済
- ▲ 幅広いターゲットプロセッサに対応

そのメリットは多岐にわたります：

- ▲ コスト削減および認証取得・量産までの時間短縮
- ▲ 製品開発後の認証メンテナンスの軽減
- ▲ ASIL/SILの最高レベルまでコードをターゲットにした、より高い品質と信頼性

Green Hillsのトレーニングとコンサルティング

Green Hills エキスパートのトレーニングとコンサルティングのサービスは、MULTI の高いポテンシャルを理解するための一番の近道であり、短時間で生産性を最大化することができます。

実装プログラムの管理

このプログラムは、開発チームが素早くGreen Hills の製品環境に慣れ親しんで生産性を向上できるように設計されています。提供されるサービスは、内容も期間も完全にカスタマイズ可能で、以下のようなものがあります。

- ▲ エキスパートトレーニング
- ▲ 環境の立ち上げや設定
- ▲ アプリケーション設計コンサルティング

設計段階から、Green Hills のエキスパートが、お客様のニーズに最も適した製品の開発に集中できるようお手伝いをします。カスタマイズされたトレーニングに加えて、マネージド・インプリメンテーション・プログラムでは、Green Hillsのエキスパートと定期的に連絡を取りながら、お客様に最適な環境の使い方を理解していただくために、積極的かつ継続的なコンサルティングを行います。

カスタマイズされたトレーニング

トレーニングカリキュラムには、INTEGRITYプログラミングから高度なHistory viewerデバッグに至るまで広範囲の教材が用意されています。コースはGreen Hillsのツールを使った開発の実地経験を持つエキスパートが担当します。クラスはお客様の施設でお客様のスケジュールに合わせて行うことができ、スケジュールへの影響を最小限に抑えて最大限の価値が得られます。

オープンエンrollment・トレーニング

少人数のエンジニア、予算が無い場合等、世界中の予定された場所で開催される人気のオープンエンrollmentコースに参加することができます。また、すでにGreen Hillsのトレーニングクラスを修了したチームに新たに加わった新入社員にも最適なクラスです。

MULTI エコシステム

新しいプロセッサやオペレーティングシステムの最新のサポートについては、Green Hillsにお問い合わせください。

ターゲットプロセッサ

- ▲ Arm (AArch32 & AArch64)
- ▲ TriCore
- ▲ Intel (x86 & x64)
- ▲ MIPS
- ▲ Power Architecture
- ▲ RH850/V850
- ▲ RISC-V

開発ホスト

- ▲ Windows
- ▲ Linux

オペレーティングシステム

- ▲ INTEGRITY
- ▲ Linux
- ▲ μ-velOSity™
- ▲ AUTOSAR
- ▲ VxWorks
- ▲ FreeRTOS

JTAG・トレースプローブ

- ▲ Green Hills Probe
- ▲ 厳選されたサードパーティプローブ

シミュレーション プラットフォーム

- ▲ Arm Fast Models
- ▲ VLAB Works
- ▲ MULTI命令セットシミュレータ
- ▲ Synopsys VDK
- ▲ Cadence VSP

言語

- ▲ C
- ▲ C++
- ▲ GNU C/C++ 拡張機能
- ▲ Ada (with AdaMULTI)
- ▲ EC++

サードパーティとの統合

コンパイラ

- ▲ GCCおよびDWARFやStabsのデバッグ情報を生成するもの

デザイン・テストツール

- ▲ Rhapsody
- ▲ SCADE
- ▲ VectorCAST
- ▲ VLAB Works
- ▲ Simulink/MATLAB®
- ▲ LDRA
- ▲ Python
- ▲ Synopsys VDK



Corporate Headquarters

30 West Sola Street ▲ Santa Barbara, CA 93101
ph: 805.965.6044 ▲ fax: 805.965.6343 ▲ email: info@ghs.com ▲ www.ghs.com

Green Hills Software GK

〒150-0001 東京都渋谷区神宮前1-5-8 神宮前タワービルディング 13階
電話番号: 03-6741-7168 ▲ お問い合わせ: jpsales@ghs.com



〒101-0045 東京都千代田区神田鍛冶町3-4 oak神田鍛冶町
電話番号: 03-3251-3170(代) ▲ www.adac.co.jp/