

CoFluent Studio

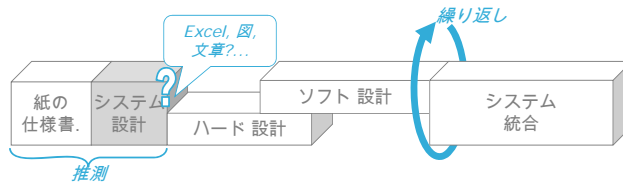
アーキテクチャ探索とパフォーマンス解析のための ESL ツール

CoFluent Studio™は、独自のマッピング技術で、複雑なハードウェア/ソフトウェア・システムをモデリングし、システムのパフォーマンス解析を可能にする、アーキテクチャ探索ツールである。

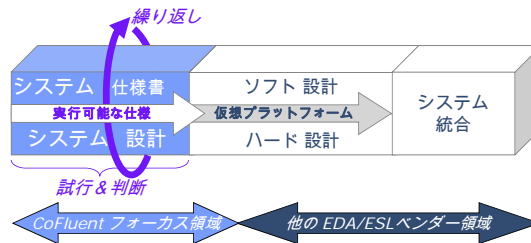
静的なドキュメントではなく、正確で動的なシステム仕様書を生成し、早期の判断を可能にする。プラットフォーム検証の為、SystemC テストベンチを生成する。

プロジェクト納期はハードウェア及びソフトウェア設計に大きく依存している。その為、アーキテクチャに対するパフォーマンスの問題を検出する為に必要な、サイクル精度の協調シミュレーションが、いつ使えるかが重要となっている。開発工程の後半でこのような問題に対処する従来の工程では、システムの検証にかかる時間と労力は膨大になる。CoFluentStudio は必要十分な抽象レベルでの検討を可能にし、早期段階で、短い時間と少ない労力で、技術的課題の解決を可能にする。

・ 従来：仕様内容が不明確のまま後工程にすすむ



・ CoFluent はインタラクティブ的に予測値を提供し早期判断をサポートする



モデルベース設計による革新

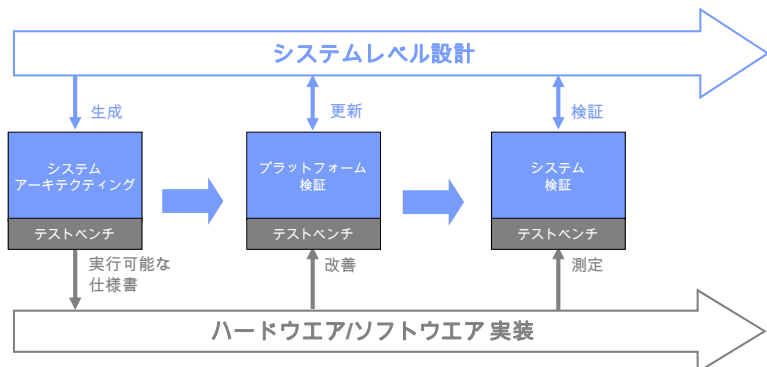
電子機器やシステム LSI 開発において、開発の中止、遅れ、またマーケットの要求に合わないというケースが増えている。これはアーキテクチャに対するパフォーマンスが十分に検討されていない事に起因することが多い。

要求機能及び実行プラットフォームが複雑化し、且つ、開発期間とコスト/パフォーマンス要求が厳しくなっている為、ますます機能とアーキテクチャの検証が重要となってきている。

マイクロ・アーキテクチャ・プラットフォームを定義する前、若しくはハードウェア/ソフトウェア設計に入る前の準備段階にシステムアーキテクティングフェーズがある。それは与えられた実時間制約やパフォーマンス制約項目（リソース、負荷、消費電力、メモリ使用、コスト、面積、等）のもとで要求される機能を実現するアーキテクチャ探索のプロセスである。このステージは、開発初期段階である為、必要情報の一部しか提供されていない状態である。つまり全体のソフトウェア若しくは完全なハードウェアというものはまだ存在していない。

しかし、この時期に設計上重要な 80% の項目への判断が必要で、これを怠ると開発コストに多大な影響を与える。モデルベース設計では、その機能、時間プロパティ、アーキテクチャそしてパフォーマンスなどの特性を得ることが可能である。それらが得られれば、ターゲットシステムの青写真とそのテストベンチが全開発サイクルで使用可能となり、後段のシステムレベルのアーキテクティング、テスト・検証が詳細化され、洗練されていく。

すべてのステージにおいて、検証しながら継続的に並列に ESL フローを実施する事で、実装作業を効率化させる。



Y-設計フローとマッピング

CoFluent Studio は最適なパフォーマンスを実現するハードウェア/ソフトウェア分割とシステムコミュニケーショントポロジーの生成を可能にする SystemC ベースのアーキテクチャ探索及びパフォーマンス解析ツール群である。

CoFluent Studio のアプローチ手法の特徴は、システムのアプリケーションと実行プラットフォームを分離してモデリングする事にある。

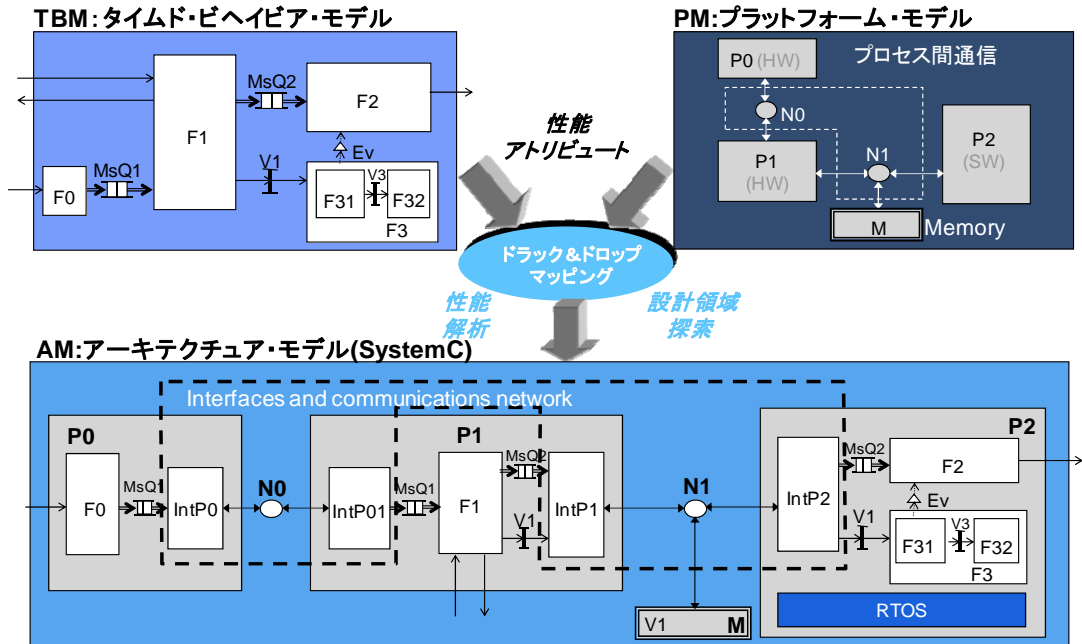
簡単な図形入力、C コード入力により、実装情報を含まないタイムド・ビヘイビアモデルを生成する。一方、汎用的なハードウェア部品で、プラットフォーム・モデルが生成される。

最後に、仮定のハードウェア/ソフトウェア・システムのアーキテクチャ・モデルを得るために、Y 設計フローを使い、アプリケーション・モデルとプラットフォーム・モデルが結合される。これはドラッグ&ドロップのマッピング操作によって簡単に行える。

CoFluent Studio は、図形表記のタイムド・ビヘイビアもしくはアーキテクチャ記述から、SystemC を自動生成する。

- ⇒ ハードウェア IP 不要
- ⇒ 組み込みソフトウェア不要
- ⇒ ファームウェア/OS 不要
- ⇒ ISS 不要

- 評価するパフォーマンス項目:
- ✓ リソース負荷
 - ✓ 消費電力
 - ✓ メモリー使用領域
 - ✓ コスト/シリコン面積



SystemC を記述することなく、また SystemC の知識が無くても、実行可能 SystemC モデルを得ることができる。

システムのメモリアドレス・マップを不要とするスレッド間で、簡単なメッセージ送受信をベースとしたトランザクションレベルで、モデルが記述される。

CoFluent Studio は、早期かつ迅速なシステム・マイクロアーキテクチャ探索、パフォーマンス解析を可能にする。これは実ソフトウェア・コードを持つハードウェア・モデルの協調シミュレーションの代わりに、ジェネリック・プラットフォームのパフォーマンス・モデルのビヘイビアをモデル化することによって可能となる。

CoFluent Studio は、システムマイクロアーキテクチャ開発と検証に対して使われる SystemC テストベンチの自動生成も可能にする。

CoFluent モデルは、プロジェクト全メンバーが使える実行可能な仕様書により、プラットフォーム開発及びハードウェア/ソフトウェア実装に対しての青写真を提供する。

CoFluent モデルは、資産蓄積及び再利用の目的のために、システムレベル機能及びアーキテクチャ IP のライブラリ化を可能にする。

Value Proposition

他の ESL ツールは、詳細なハードウェア及びソフトウェアモデルから結果を生み出している。

CoFluent Studio は、一部のハードウェア及びソフトウェア記述でシミュレーションが可能 (サイクル精度の仮想プラットフォーム不要、組み込みソフト不要) CoFluent Studio は以下の価値を提供する

革新

新しい機能と従来の手法が混在したシステム開発において、既に作成したモデルを再利用する事により、最小限の開発労力でアーキテクチャを検討

最適化

HW・SW の制限のない検討スペースで、最適なアーキテクチャや高効率な消費電力化が探索可能

検証

リアルタイムビヘイビアの検証、性能予測、実インプリ用のテストケースを生成する為に、動作環境・シナリオを定義

オールインワンのアプリケーション・モデリング環境

- 並列実行の関数（若しくは複数プロセス）のプロセス間通信を、モデリングする。
- 技術的視点と物理的視点を切り離して、システムの様々な動的振る舞いを表現する。
- 全プロジェクト関係者にわかりやすいグラフィック・モデル・シンボルを使いモデリングする。
- 同一モデル内に、システム構造、制御フロー、データフロー及び時間プロパティの全てを取り込む。
- テストベンチとして使われる環境に沿って、完全なシステムを記述する。
- 外部ソースファイル、オブジェクト C/C++、SystemC コードを統合する。
- 作成したモデルをライブラリとしてエクスポートし、次期プロジェクトで再利用する。ライブラリのインポート時はモードコントロールする。

システム・ビヘイビアと時間的制約項目における綿密な検証

- 実時間ビヘイビア・モデリングで、時間を演算ブロックと入力/出力ブロックに割り当てる。
- グラフィカル・モデルから、トランザクショナルな SystemC を自動生成する。
- スレッド間（メモリーマップ不要）でのメッセージ送受信をベースにした高速な SystemC シミュレーションをする。
- 多くの視覚化された解析ツールにおける自動計測機能からシミュレーション結果を表示する。
- システムのビヘイビア、タイミング、コミュニケーションそしてパフォーマンス・データを観察する。

視覚的なシステム・アーキテクチャ・モデリング環境

- ターゲットシステムに要求される機能・ビヘイビアと、そのビヘイビアを実行する HW プラットフォームを分離してモデリングする。
- 演算ユニットとメモリ・ユニットのネットワークとして、HW プラットフォームをモデリングする。
- シンプルで汎用的な HW 部品（演算ユニット、通信ノード、共有メモリ）のグラフィック・シンボルでモデリングする。特定の IP モデルや ISS を必要としない。
- 短時間に、必要な数の HW プラットフォーム・モデルが定義できる：例えばシングル、マルチ・プロセッサ、複数のプロセッサ間通信ノードによる接続
- 仮定のマクロ・アーキテクチャ・モデルを得るために、ドラッグ&ドロップ操作によってアプリケーションの機能ブロックを HW プラットフォームの各部品にマッピングする。
- シミュレーション目的の為に、各機能と実時間ソフトウェア・タスクとハードウェア・ブロックを 1 対 1 マッピングさせる。

システム・レベル検証環境

- 1 行のコードを書かなくても、異なるプラットフォームと交換したり、マッピングを変更して、“what-if” シナリオを実行できる。
- 時間精度とパフォーマンス精度を持たせたシミュレーション結果を得ることができる。
- 実装の青写真として使われるレファレンス仮想システム・モデルを提供する
- 全プロジェクト工程で使われる、SystemC テストケースを生成する。

製品パッケージ

機能	製品パッケージ	Timed-Behavioral Modeling	System Architecting
アプリケーション・モデリング		✓	✓
TLM/TL3 SystemC 生成&シミュレーション		✓	✓
機能検証		✓	✓
リアルタイム解析		✓	✓
プラットフォーム・モデリング			✓
アプリケーションとプラットフォームのマッピング (HW/SW の分割)			✓
ソフトウェア・アーキテクチャ探索 (タスク・スケジュール解析)			✓
アーキテクチャ探索			✓
パフォーマンス・プロファイリング			✓

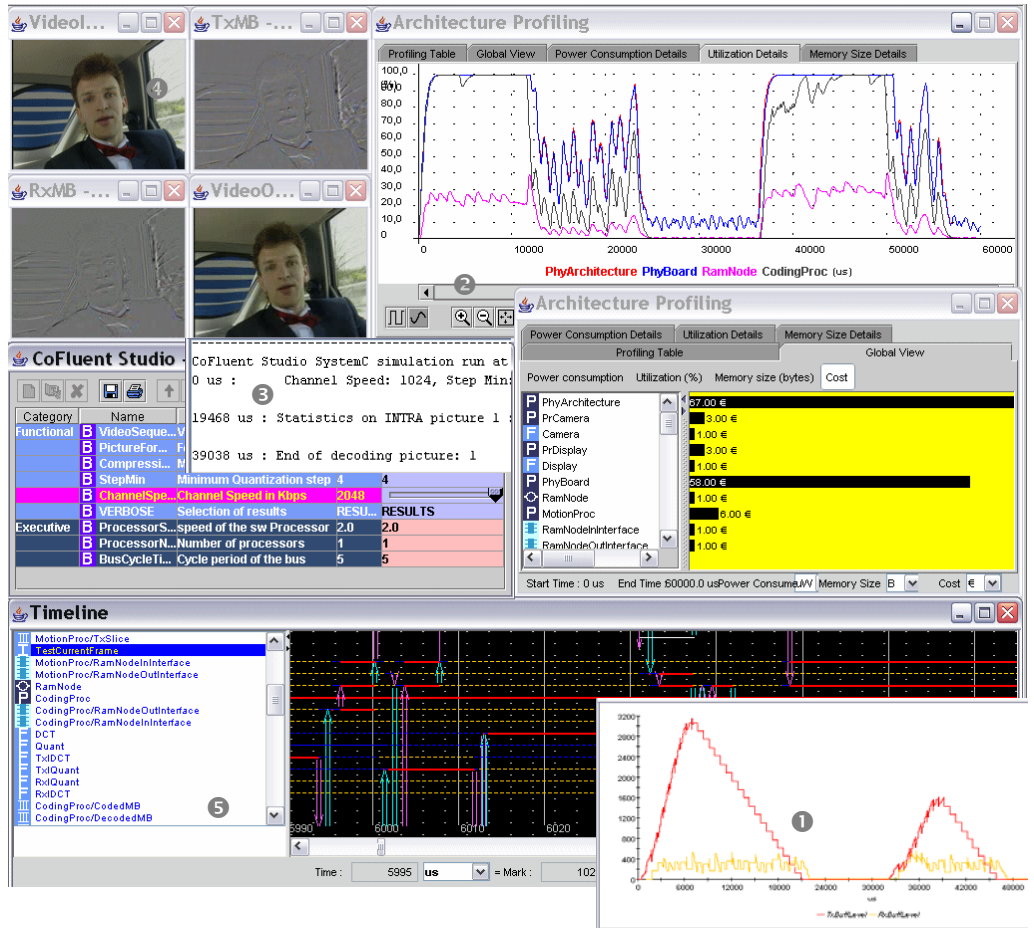
● Supported platforms	<i>Host OS:</i>	Microsoft Windows
● Supported SystemC platforms		OSCI SystemC 2.2 & TLM-2.0, CoWare Platform Architect, Synopsys Innovator
● Supported C++ compilers	<i>For Windows-based simulation:</i>	Microsoft Visual Studio 2005 & 2008 MinGW GCC 3.4 & 4.1
	<i>For Linux-based simulation:</i>	GNU GCC 3.4 & 4.1

性能予測の可視化 デバックツール

シミュレーションで様々な視点の情報を得ることが出来る：
ビヘイビア、コミュニケーション、リアルタイム、パフォーマンス（リソース使用状況、メモリ・サイズ、消費電力、コスト等）

性能予測の可視化：

- ① 各種性能値をダイナミックにプロファイリング
- ② 各種性能指標：関数/プロセッサ内部関数/プロセッサ関係（プロセッサとバスの負荷）動作状況
メモリ使用領域、消費電力
コスト見積り
- ③ デバック結果
データ依存アルゴリズムの実行プロファイル
- ④ 画像処理結果
- ⑤ タイムライン
完全なタイムド・シーケンス・ダイアグラム
タスク間通信・同期を含む
マルチ・タスク処理の観察が可能



メリット

開発時間の短縮

- ▶ SystemCの自動生成によって、コーディングとデバック時間を短縮
- ▶ グラフィック記述とシステム・レベルのモデル化で、複雑なシステムに対応し、開発効率が向上
- ▶ システム分解とハードウェア/ソフトウェア分割によって、実装の立ち上げ時間を短縮
- ▶ 共通のHW/SW仕様書とテストベンチが、サードパーティとの共同作業を円滑にし、決定を迅速化

プロジェクトの遅れ、中止のリスクを低減

- ▶ タイムド・ビヘイビア・モデリングと検証によって、機能上の問題やタイミングエラーを早期に発見できる
- ▶ HW/SW設計に入る前の、早期段階で迅速な設計領域探索とパフォーマンス解析をする事によって、最適な設計指標を得る事ができる。
- ▶ HW/SWの協調モデリング手法によって、HW/SWの結合を容易にし、インテグレーション時の問題を減らす。
- ▶ 開発チーム間で、仮想システムモデルを実行可能仕様書として共有する事で、設計方針の曖昧性を排除し、円滑なコミュニケーションを図り、その結果チームワークによる開発効率を向上させる。

最終製品のコストダウン

- ▶ アーキテクチャ探索とパフォーマンス解析によって、アーキテクチャの最適化を可能にする
- ▶ モデルに挿入された様々な情報は研究開発で得られた成果をデータベース化する際重要な構成要素となる。

新規製品開発の促進

- ▶ アプリケーションとプラットフォームを分離してモデリングする手法が、自由な創造性をもたらす。モデルを容易に変更できる為、柔軟な検証が可能となる。
- ▶ 作成したモデルをライブラリ化する事により、IP蓄積として再利用できる。



CoFluent Design Japan
Shinagawa Intercity A Tower 28F
2-15-1 Kounan, Minato-ku
Tokyo 108-6028 – Japan
Phone: +81-3-6717-4224

info-jp@cofluentdesign.com

CoFluent Design Europe
24, rue Jean Duplessis
78150 Le Chesnay – France
Phone: +33-139-661-697

info@cofluentdesign.com

CoFluent Design, Inc.
2290 N. First Street
Suite 304
San Jose, CA 95131 – USA
Phone: +1-408-573-6172

info-us@cofluentdesign.com