In-Circuit-Emulatorの過去、現在、未来

京都マイクロコンピュータ株式会社
代表取締役社長
山本彰一
jp-info@kmckk.co.jp
http://www.kmckk.co.jp/

自己紹介

- 岡山大学工学部卒業
- 1985年に京都マイクロコンピュータ(株)を設立し代表取締役社長に就任(設立20年)。
- 以降一貫して開発支援装置の開発に携わる
  - PARTNERシリーズというデバッグ支援装置
  - exeGCC:GNU CのNative Windows Compiler
  - 評価ボード
- H/WとS/Wの境界面に存在する現役エンジニア

Agenda

- In-Circuit-Emulatorとは
- In-Circuit-Emulator歴史(過去)
- トレンド(現在)
- これから(未来)

In-Circuit-Emulator（ICE）とは

マイクロコンピュータシステム(マイコン基板)を開発する際に使うデバッガ。ソフトウェアのデバッグとハードウェアの動作確認を行なうことができる。

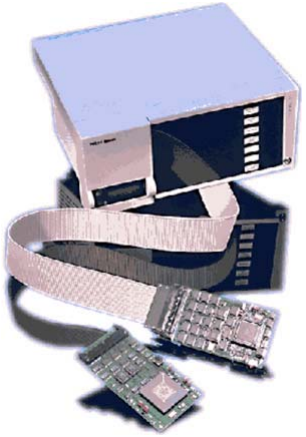
ICEのデバッガとしての機能には、任意のアドレスで実行を停止させるブレークポイント機能や、プログラムの特定の命令を実行する度に指定されたメモリの内容を出力するシングルステップ機能、アセンブル機能・逆アセンブル機能などがある。

また、実行時間を実時間で確認できるリアルタイムトレース機能、レジスタへのデータ設定機能なども搭載されている。エミュレーションメモリ機能を使えば、アプリケーションをエミュレータのメモリ上に置いてプログラムの動作確認ができるため、効率よく不具合の修正を行なうことができる。

ICEの歴史(過去)

ICE誕生の世代

- 8080,Z80,6800,8086,68000など
  - CPUクロック：～十数MHz
- **ハードウェ構成**：実際CPUの周りにバッファを入れた、ICEプローブ
- **接続**：ターゲットのCPUを抜いてICEプローブで接続
- **特徴**：実CPUの周りに回路をいれて、そのCPUのターゲットと接続するため、不安定な動作やターゲットとの相性の問題が多く発生。CPUが高速クロックになるに従って、機能実装が不可能となる。
- **機能**：エミュレーションRAM、リアルタイムトレース(バストレース)
- **価格**：ハード構成が巨大でかつ、高価



Eva Chipの世代

- 64180, H8, NEC V_{86}シリーズなど
  - CPUクロック：～数十MHz前半
- **ハードウェ構成**：エバチップを使ったICEプローブ
- **接続**：ターゲットのCPUを抜いてICEプローブで接続
- **特徴**：CPUが多ピンとなってきたのでCPUとターゲットとの接続が非常にデリケートとなる。不安定な動作やターゲットとの相性の問題が多く発生。エバチップが特別なチップのため、供給の問題が発生した。
- **機能**：エミュレーションRAM、リアルタイムトレース(バストレース)
- **価格**：ハード構成が巨大でエバッチプ自体も高価なためさらに高価

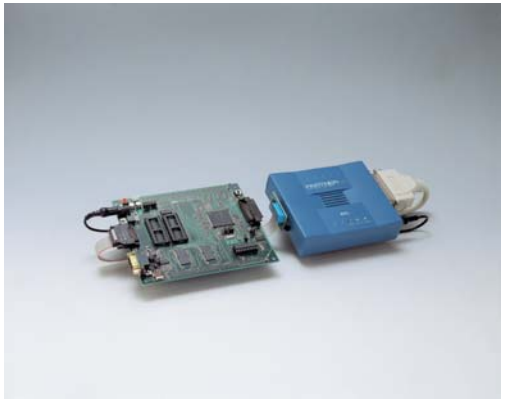
ROM ICE

- Eva chipからJTAGへの進化の穴を埋めたテクノロジ
  - On Chip Debug以前で、
    - 高クロック
    - キャッシュON

  などが実現された時に、効果を発揮
  （初期のMIPSなど）

JTAGの世代

- ARM,SH,MIPS,PowerPCなど
  - 32bit RISC 数十MHz後半～
- **ハードウェ構成**:ターゲットとの接続の最小構成では、JTAGのみのなる。トレース用の専用ピンがあるCPUも存在。ターゲットCPU内にデバッグの為の基本機能を入れてあるため、接続用のハードウェアは比較的簡単。
- **接続**:JTAG＋トレース信号で接続
- **特徴**:実チップそのものため、極めて安定性が高い。(ICE接続時の問題が少ない)。JTAGの接続とCPU自体のクロックは、異なるため高クロックCPUにも対応可能。
- **機能**:CPUの機能を使ったリアルタイムトレース。
- **価格**:ハード構成が簡単で比較的安価



JTAGでの on chip デバッグ

- 高速CPUは、全てJTAG ICEとなる。
  特にユーザインターフェィスソフトウェアの規模が大きい、携帯電話、テレビ、カーナビなどでは、これらのCPUを使用する。

ICEに求められる事

- ハードウェア面
  - 透過性(ICEでの動作と実ターゲットとの動作が同じ)
    →ツールとしての安定性(ツールを信頼できるか?)
  - メモリやIOコントロール
    →IOの初期化なしに動作できる。
    →不完全なターゲットでもある程度動作できる。
- ソフトウェア面
  - ツールとしての安定性(ツールを信頼できるか？)
  - 対応言語やOSのサポート(C, C++, iTRON, Linux･･･)
  - デバッガの使いやすさ(手になじむツール)
  - 高速な動作(高速ダウンロード、高速ステップ実行)

トレンド(現在)

組み込みシステムの進化

- 大規模なソフトウェア開発
  - 携帯電話、カーナビ、デジタルTV/レコーダなど




- 大規模化するソフトにあわせて、プラットフォームの変化
  - ソフトウェア　大型OSの採用
  - ハードウェア　高機能、高性能半導体

組み込みシステムの進化

- 仮想記憶をサポートしたOS(LINUX,WinCE,T-Kernelなど)
  - ソフトウェアの巨大化
  - ネットワーク接続
  - オープンソースの台頭
- SoC全盛
  - エンコーダなど多くの機能が一つのシリコンに入る時代で当然CPUもその中に入る。
- マルチコア(パソコン、MP211,MPCoreなど)
  - 高機能CPUの開発は、より困難になってきた。
  - 高クロックの限界

ハードウェアの進化

- 演算性能を強力に
  - 高クロック、大容量キャッシュ
    ↓ それも厳しくなり…
  - **マルチコアの採用**
- 多機能を低コストに実現するために
  - 動画デコーダなど各種機能を搭載した**SoC**化

ソフトウェアの進化

- ソフトウェアの大規模化
  - プロセス、ドライバのモデルを導入
    - 開発効率化、システムの堅牢性向上
  - ネットワーク接続機能の採用
    - TCP/IPスタックなどの標準搭載
  - ストレージの巨大化
    - 高機能なファイルシステムの搭載

- Linuxなどの仮想記憶を採用した大型OSが増える

いままでのICEの限界

- LinuxはICEでデバッグ不可能といわれた
  - デマンドページング、仮想空間、多重空間など
- マルチコアをどうデバッグするのか？
- 実現するために、ICEを大きく進化させる

動的なICEへ

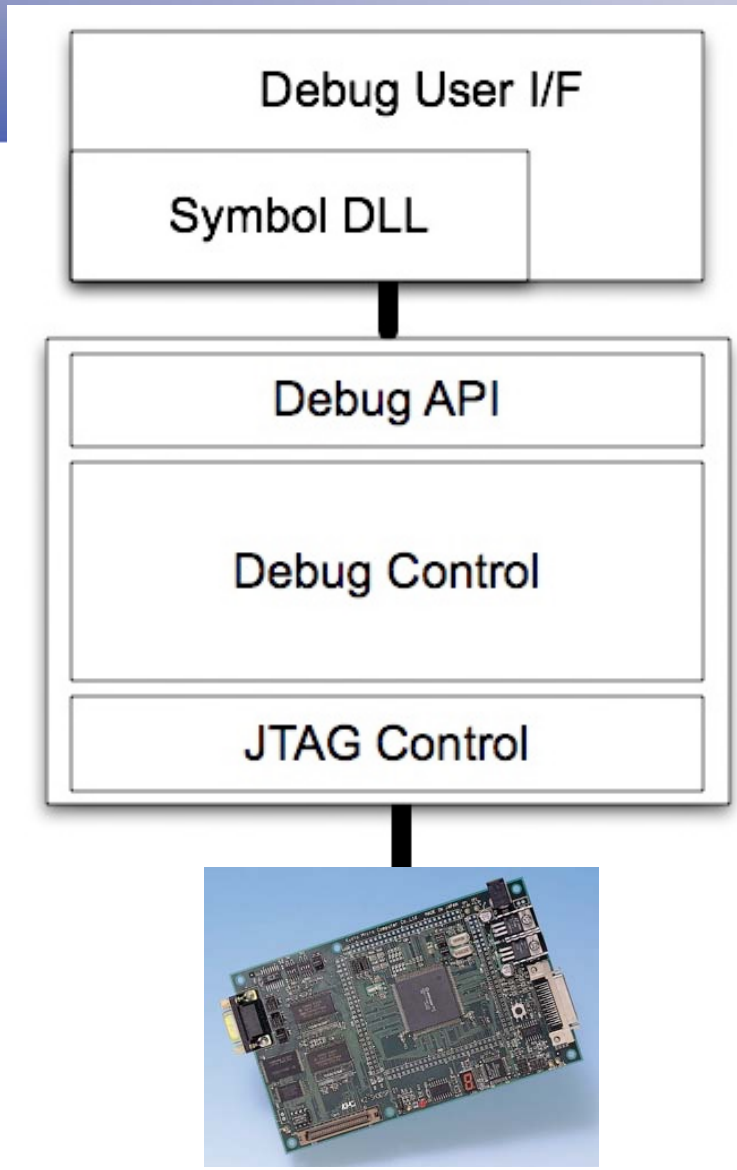
- ICEが静的から動的に進化
  - デマンドページング解決をICEが行う
  - ICEがプロセスのメモリ表示するために、カーネルとCPUを自分で調査する
  - マルチコアでのブレークポイントの制御

- 上記のように、多くの処理をユーザー処理のバックグラウンドで実行

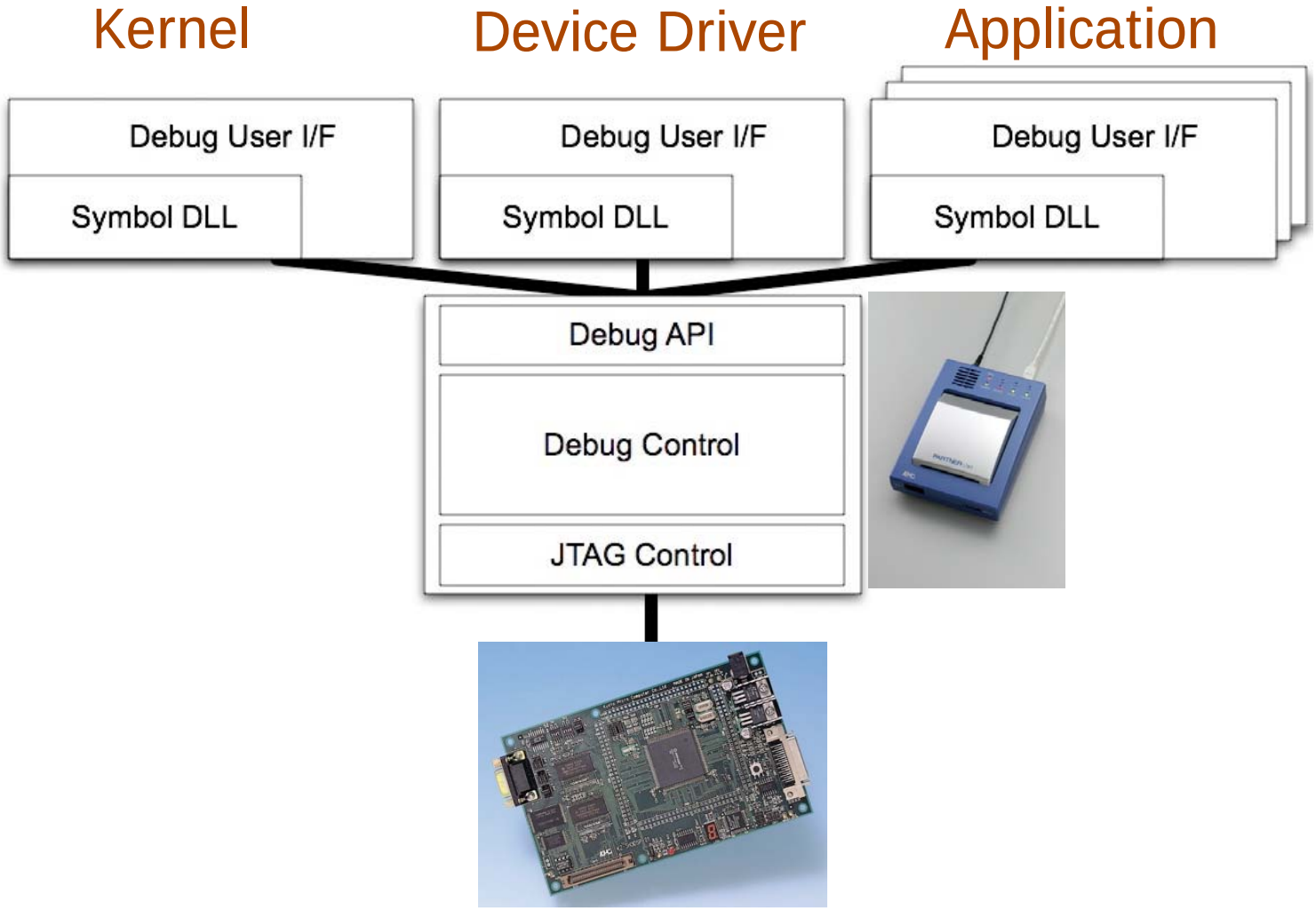
KMCの仮想化テクノロジ

- 動的なICEを開発するために、ICEの各レイヤの機能を徹底して仮想化
  - デバッガソフト
    - ユーザインターフェース
    - Symbol
  - デバッグAPI
  - JTAGコントロール
- 仮想化による処理速度の低下を発生させないよう、ICE BOX側とデバッガUIの処理の工夫（多数の処理はICE BOXとターゲット間の通信で完結する）

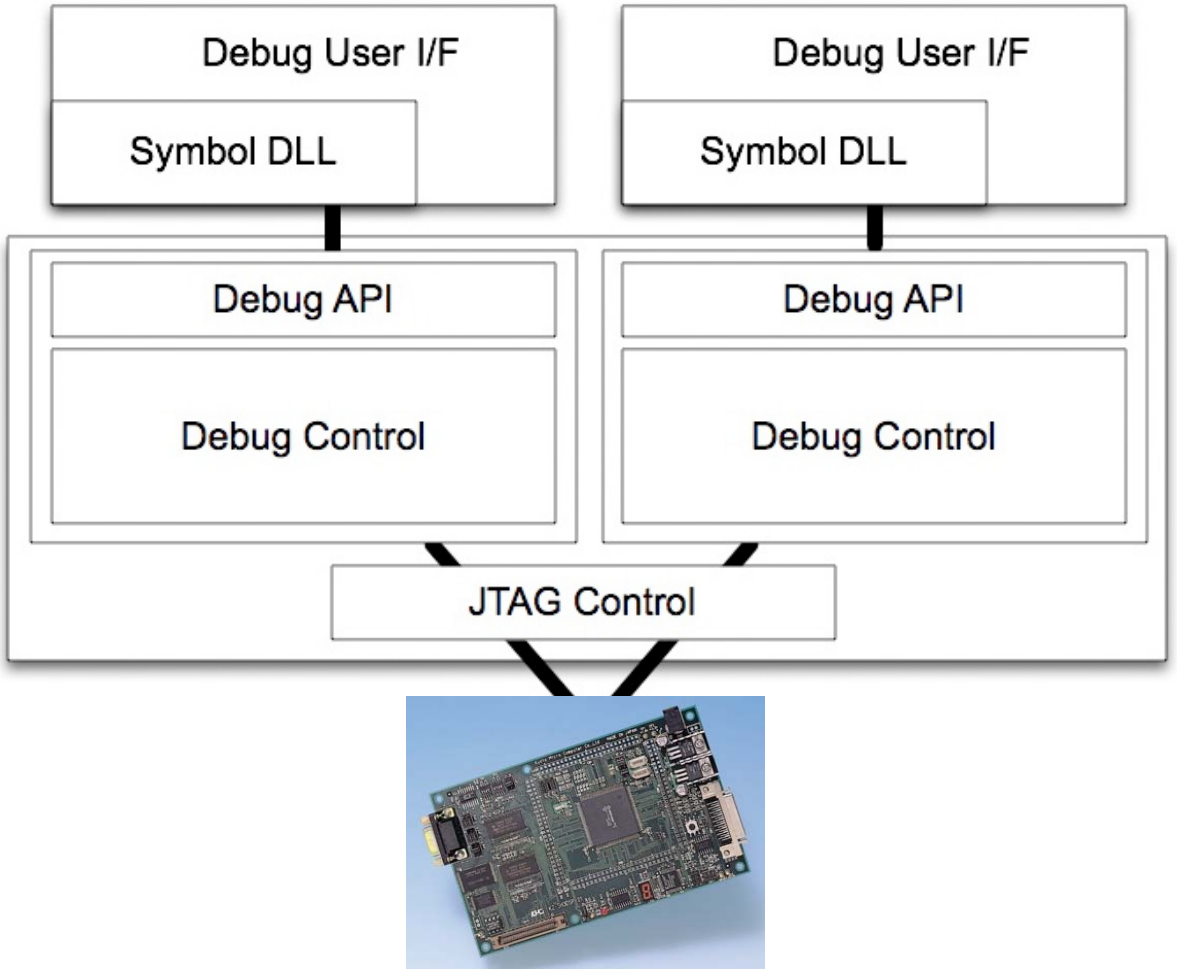
PARTNERのアーキテクチャ



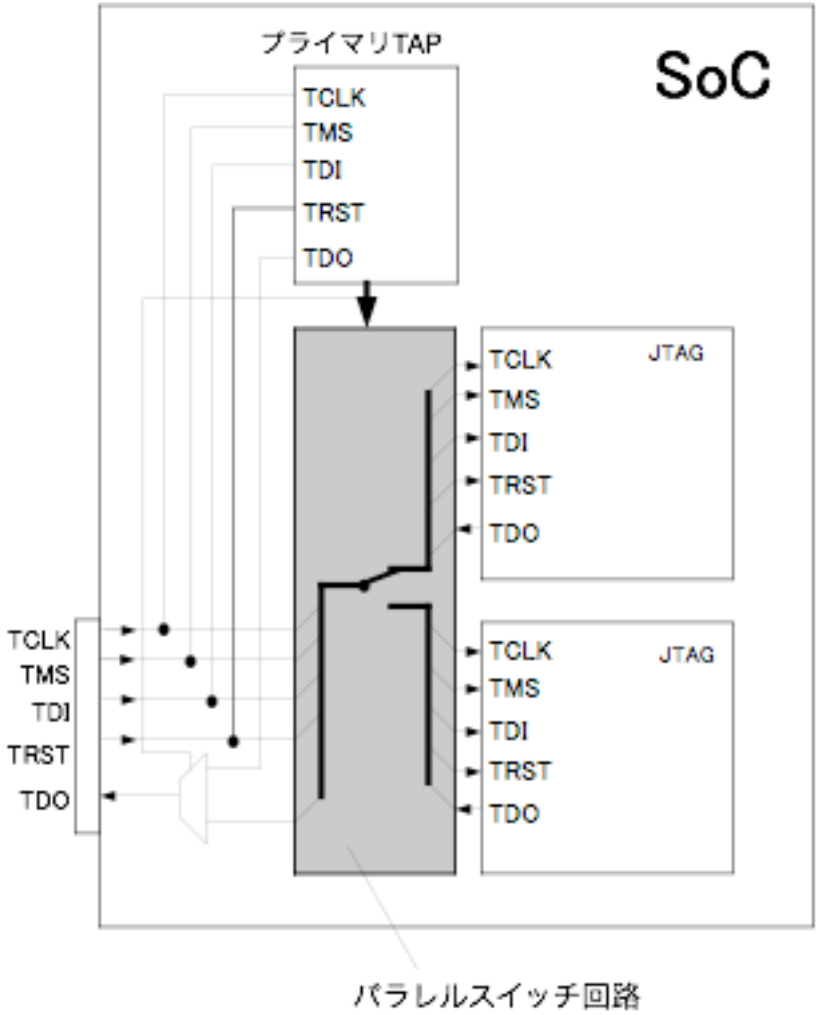
Linuxのデバッグ


Kernel
Device Driver
Application
Debug User I/F
Symbol DLL
Debug User I/F
Symbol DLL
Debug User I/F
Symbol DLL
Debug API
Debug Control
JTAG Control

Multi-coreのデバッグ対応



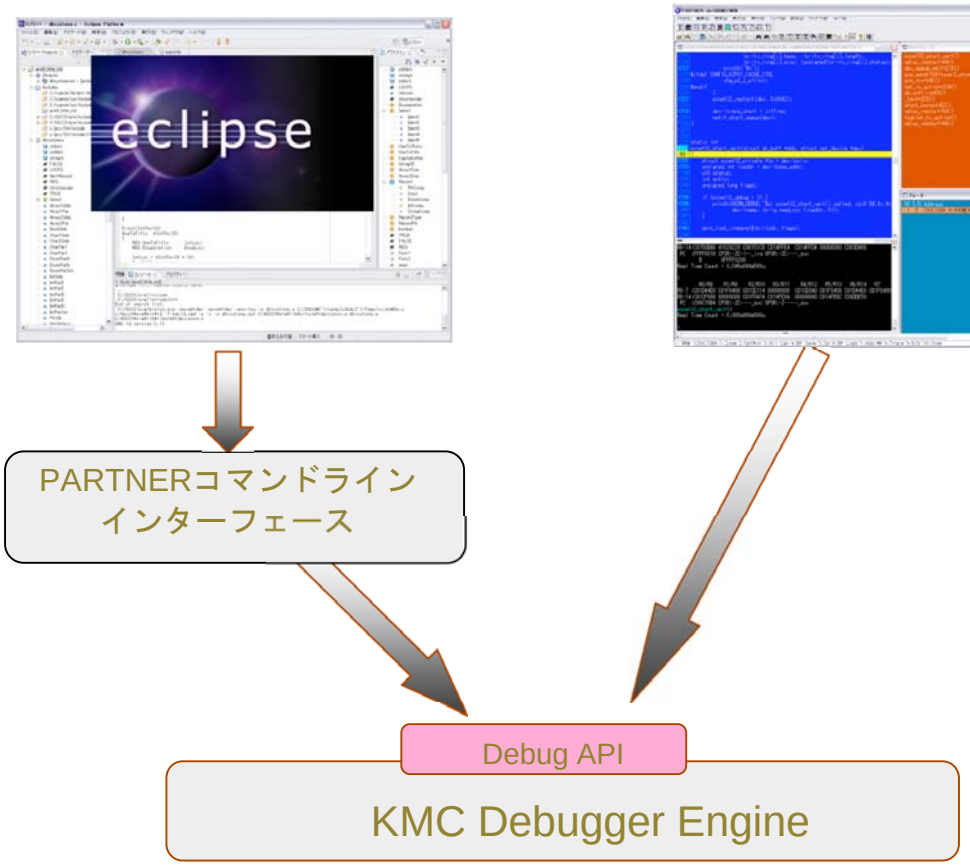
マルチコア – JTAGの実装例



仮想化(レイヤ化)によるメリット

- サードパーティやユーザ連携
- ICEのデバッガ以外の使用の可能性
- Eclipseとの連携も実現

EclipseとPATNERの協調動作



これから(未来)

進化する組み込みのデバッグへの対応

- SMP
- メイニイコア（many core）
- ボトルネックを探す技術 – プロファイル
  - （バスのモニタなど、SoCでのサポートなど）
  - マルチコアやSMPはバスの状態も大事

セキリュティとJTAG ICE

- 全ての組み込み機器は、ネットワークでつながる
- 組み込みでもセキュリティ
- JTAG　ICEは、強力なハッキングツール？
- 半導体メーカとの連携

デバッガ以外への応用

- Virtual Link
- パフォーマンスモニタ
- テスティングツール
- フィールドメンテナンスツール
- UMLツールなどとの連携
- シュミレータとの連携

提案

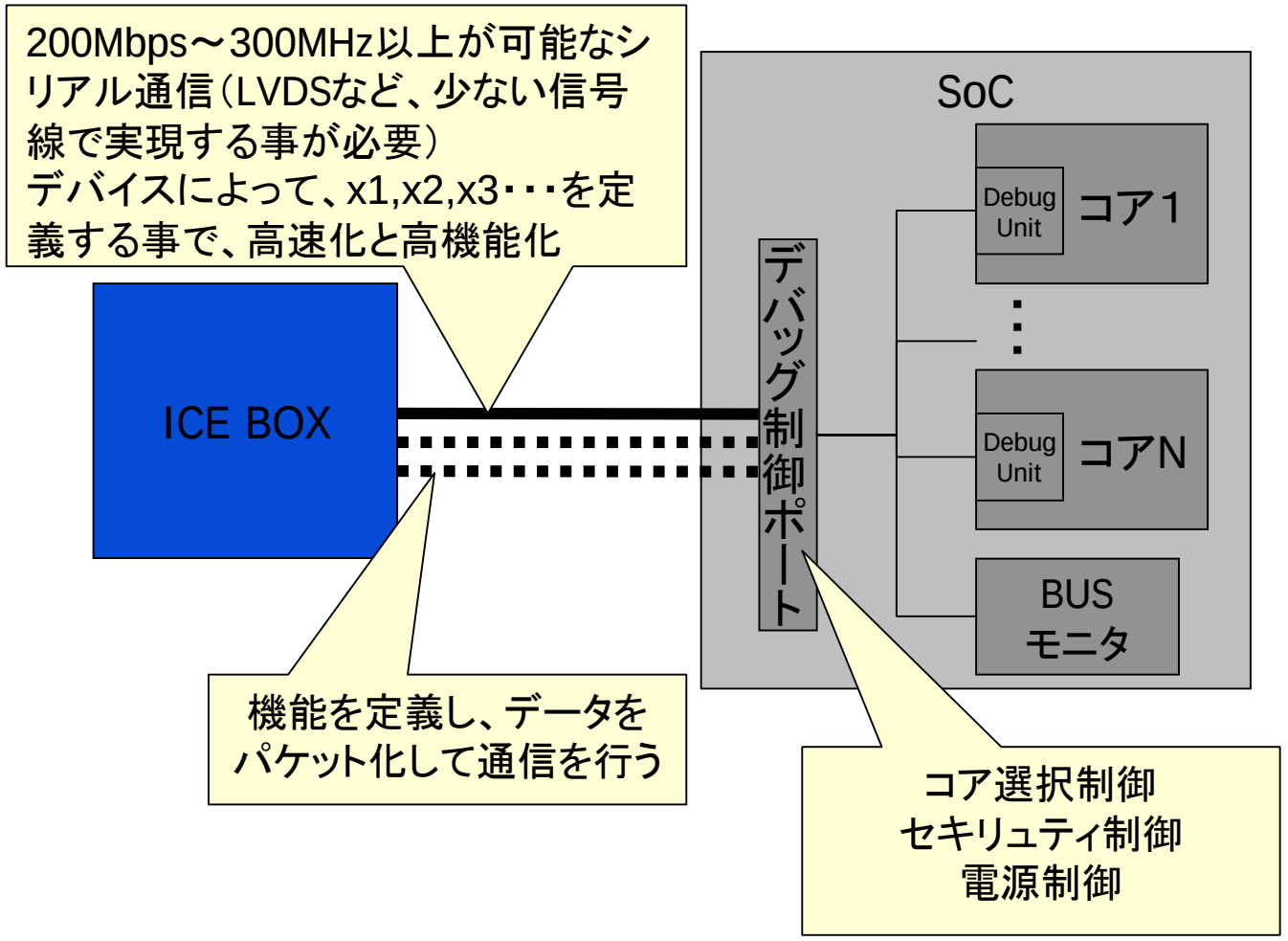
- 組み込みシステムの構成が、いま大きく変化している
  - 高クロック、Soc、マルチコア、大型OS採用など
- 今のJTAGによるOn Chip Debugは、これらの事があまり考慮されていない（登場したのは変化前）
- 今のOn Chip Debugの方式を、変化に合わせて見直す時期では？

JTAGに置き換わる仕様

- 少ない信号線で200Mbps～300Mbpsを実現するデバッグ専用ポート
- 信号線をx1,x2,x3・・・とする事で、帯域を大きくする事が可能な仕様
- マルチコアデバッグ、トレース、プロファイル、メモリエミュレーション、汎用通信などをデバッグ専用ポートで実現

JTAGに変わるデバッグシステム

- 高機能なマルチコア/SMPデバッグ
- 高速な実行、データトレース
- メモリのエミュレーション
- 高速汎用通信ポート
- OSサポート



まとめ

On Chip Debug 機能は、これからも主役