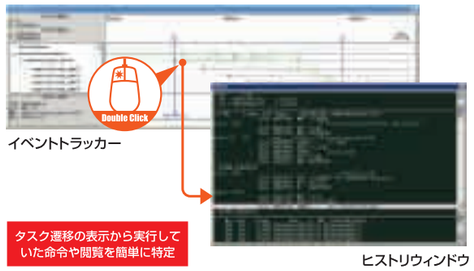
Embedded Linux

数十億命令の保存が可能な PARTNER-Jet Model40「ギガトレース」新登場!

大容量の実行トレースとプロセス遷移が連携

プロセス遷移を解析するイベントトラッカーは、大容量の実行トレースと連携して動作します。これにより、イベントトラッカーによるマクロな解析から、トレース結果を参照するミクロな解析まで、シームレスに行うことが可能で、システムの性能ボトルネックの抽出などを効率的に行えます。



イベントトラッカー

ヒストリウィンドウ

ギガトレース対応、Model40

- 数十億の命令を保存可能
- 大容量記録をいかすための、統計解析機能や、高速表示機能

各種RTOSに対応した、イベントトラッカー機能

- 任意のユーザーイベントの記録と表示にも対応
- すべてのPARTNER-Jetに標準添付

Linux	Windows CE	T-Engine
SMP-Linux	Multicore CPU	Eclipse
Visual Studio 2005 for Windows CE	PARTNER	イベントトラッカー

- JTAGによる超高速ダウンロード
  3MByte/secに対応した余裕の高速ダウンロード ※1
- 全機種USB2.0/1.1対応
  LAN(100Base-TX/10Base-T)対応(Model 30 以上)
- 大容量トレースメモリ **最大8GBit**
- 高速トレースクロックに対応 **400MHz** ※2
- CPU汎用対応(Model 20 以上)
- マルチコアCPUに対応
- オプションエミュレーションメモリ、
  ROMエミュレーション対応(Model 20 以上)
- VLINK対応:転送速度1MByte/sec以上 ※1
- Linux、T-Engine、Windows CEほか各種組み込みOS対応
- SMP-Linux対応
- 統合開発環境Eclipse対応
- FLASHメモリ書き込み対応(NORフラッシュ)
- デバッガソフトは業界で定評のある"PARTNER"が標準で付属

対応CPU ARM / MIPS / SH / V850 / MN103

※1 ターゲット、CPUの条件により速度は異なります。
※2 ARM CPUでトレースクロック200MHz,デュアルエッジでの場合。実際の動作可能周波数は、CPUやターゲットを含んだ電気的条件により制限されます。

京都マイクロコンピュータ株式会社
本　　社:〒610-1104 京都市西京区大枝中山町2-44　Tel.075-335-1050　Fax.075-335-1051
東京オフィス:〒105-0004 東京都港区新橋2-14-4 Rビル5F　Tel.03-5157-4530　Fax.03-5157-4531

※記載の社名、製品名は、各社の商標または登録商標です。記載内容は予告なしに変更する場合があります。

2008.11

The Debugging Evolution

Linux

PARTNERはデバッグ環境を創造する

カーネルからアプリケーションまで

PARTNER-Jetで Embedded Linuxをデバック

MMUを使った仮想空間に対応

カーネルやモジュールだけでなく、ユーザーランドのプロセスもデバッグ可能。

マルチコア、SMPにも対応

一台のPARTNER-Jetで複数のコアのデバッグが可能。
複数のコアを利用したSMPカーネルでも、Linuxデバッグ機能の全てが利用可能。

強力なデバッグ機能

ETMなどのCPUのトレース履歴を、ユーザーランド含めて対応。
実行中プロセスの、動的アタッチによるデバッグ開始共有ライブラリの自動読み込み、自動リロケーションイベントトラッカー機能による、実行状況解析や性能解析機能など、豊富なデバッグ機能を用意。

カーネルデバッグ
ローダブルモジュールデバッグ
アプリケーションデバッグ
共有ライブラリデバッグ

JTAG接続

Embedded Linux

- **PARTNER-Jet対応Linux**
- **カーネルバージョン** kernel2.4、2.6系
- **ディストリビューション** MontaVista PRO、CEE、PREVIEW-KIT、Lineo uLinux、TimeSys Linux、axLinuxなど
- **対応CPU** PARTNER-Jetのサポートする ARM、MIPS、SH、MN103

Linuxデバッグを使うためには、別売の"Linuxサポートパック"(ソフトウェアおよびドキュメント)が必要です。価格や購入方法については、弊社までお問い合わせ下さい。

京都マイクロコンピュータ株式会社

http://www.kmckk.co.jp/

カーネルからアプリケーションまで

PARTNER-*Jet* で Embedded Linuxをデバッグ

●デバッガから見たLinuxのプログラムの種別と特徴

プログラム種別	空間	アドレス	ページング	デバッグ
ブートローダ	非MMU空間	固定番地	なし	既存の組み込みデバッグに同じ
カーネル	MMU上の単一カーネル空間	固定番地	（ほぼ）なし	既存の組み込みデバッグに（ほぼ）同じ
ローダブルモジュール	MMU上の単一カーネル空間	リロケータブル	デマンドページング	リロケーション解決、ページングへの対応が必要
アプリケーション	MMU上の論理多重仮想空間	固定番地	デマンドページング	論理多重空間、ページングへの対応が必要
共有ライブラリ	MMU上の論理多重仮想空間	リロケータブル	デマンドページング	リロケーション解決、論理多重空間、ページングへの対応が必要

上記のようにLinuxのデバッグをICEで行うためには、デマンドページングやリロケータブルフォーマット、そしてMMUを用いた仮想空間などに対応する必要があります。PARTNER-Jet/JetMはこれら全てに対応しているので、Linuxの全てをデバッグする事が可能です。今までLinuxアプリケーションが動作する論理多重仮想空間にICEで対応する事は難しいと言われてきました。KMCでは業界でいち早くこの問題を解決し、MMUフルサポートを行う事で、ICEによるLinux開発を可能にしました。PARTNER-JetのLinux対応機能は、多くのLinux開発者の意見を頂いて作られています。

●Linuxデバッグ機能比較

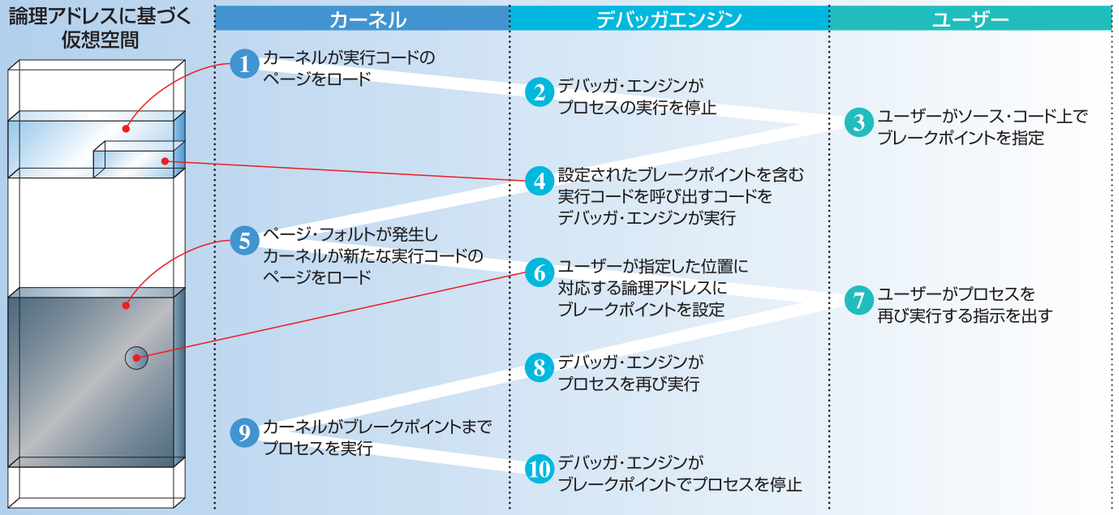
		PARTNER-Jet	一般的なICE	gdb（ptrace）	kgdb
デバッグ対象	カーネル	○	○	×	○
	モジュール	○	△ ※3	×	○
	アプリケーション	○	×	○	×
	特別なアプリ（init=プロセスID1番）	○	×	×	×
	XIPアプリケーション ※1	○	×	△ ※2	×
	マルチプロセス	○	×	○	×
	マルチスレッド	○	×	○	×
デバッグ機能（カーネル空間）	ソフトウェアブレーク	○	○		○
	ハードウェアブレーク（命令実行）	○	○		△ ※6
	ハードウェアブレーク（データアクセス）	○	○		△ ※6
	実行トレース（分岐トレース）	○	△ ※4		×
	ソースレベルデバッグ	○	○		○
	ローダブルモジュールの自動リロケーション	○	△ ※5		未調査
	デマンドページングの解決	○	×		○
	カーネルブレーク中のプロセスデバッグ	○	×		×
デバッグ機能（ユーザー空間）	ソフトウェアブレーク	○		○	
	ハードウェアブレーク（命令実行）	○		△ ※6	
	ハードウェアブレーク（データアクセス）	○		△ ※6	
	実行トレース（分岐トレース）	○		×	
	ソースレベルデバッグ	○		○	
	デマンドページングの解決	○		○	
	共有ライブラリの自動リロケーション	○		○	
	実行中プロセスへのアタッチ	○		○	
	プロセスブレーク中のカーネルデバッグ	○		×	
	プロセスブレーク中のカーネルや他のプロセスの実行	○ ※7		○	

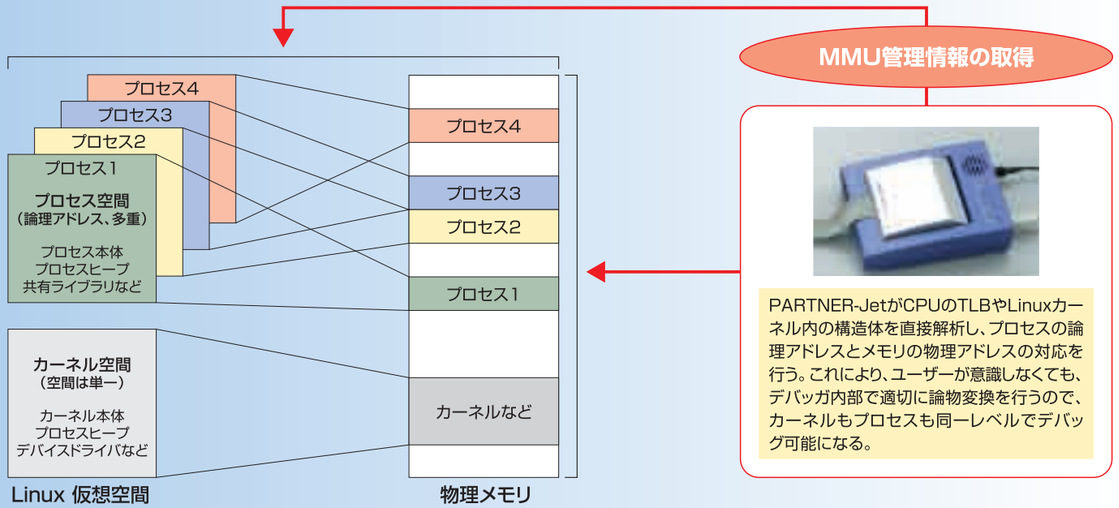
※1　XIPアプリケーションとは、アプリケーションのtextセクションがフラッシュメモリなどのROMデバイス上に配置されるアプリケーション
※2　カーネルにXIPアプリケーションデバッグの機能を実装すればgdbでもデバッグ可能です
※3　ICEにリロケーション機能の実装が必要です。またページアウトしている部分はデバッグ出来ません
※4　ユーザー空間の実行トレースが存在すると、正しく解析できない可能性があります
※5　デバッグ情報をリロケーションする機能が必要です
※6　カーネルのデバッグ機能に、CPU固有のハードウェアブレークポイント機能の実装が必要です
※7　仮想ICE機能で実現します

Linuxデバッグ技術解説

デマンドページングへの対応

Linux上で動作するカーネル以外のプログラムは（ローダブルモジュールやユーザーアプリケーション）、初期化された状態では全てのプログラムがメモリに存在する訳ではありません。実行時に「ページフォルト」を発生させながら、カーネルが必要な部分だけをメモリに読み込む「デマンドページング」が採用されています。そのため、ICEからS/Wブレークポイントを設定したり、また変数を参照したりする時に、そのメモリがまだメモリに読み込まれていない場合には、その処理を ICEは継続できません。そこでPARTNER-Jetは、必要な時に必要な部分だけ意図的にページフォルトを発生させるプログラムをバックグラウンドで実行し、メモリ上にプログラムのデバッグに必要な部分を読み込ませる処理をしています。



MMUへの完全対応

Linuxのユーザー空間は MMUによって仮想化された多重空間になっています。この多重空間では、たとえば 0x1000番地というアドレスも複数の仮想空間に存在する事になり、そのアドレスだけでは「どのプロセスの 0x1000番地なのか?」という事が分かりません。これらの問題や前述のデマンドページングなどの、Linuxには複雑な問題があるために、従来の ICEでは Linuxのデバッグは困難と言われていました。PARTNER-JetはMMU上の多重仮想空間に対しても、PARTNER-Jetが自分でカーネルのプロセス管理・ファイル管理・メモリ管理などの構造体情報を調べて、必要なアドレス変換を行いながら正しくデバッグを行えるよう、デバッグするユーザーのバックグラウンドで処理を行っています。