TRACE32®

Linuxデバッグ

- ARM®/Cortex™
- Intel® Atom™
- MIPS® Architecture
- Power Architecture®
- and others

LAUTERBACH
DEVELOPMENT TOOLS

Linux デバッグ環境

TRACE32 は、標準 Linux カーネルを使用するすべての Linux ディストリビューション、バンドル、およびプラットフォーム（Android など）をサポートします。以下の機能が提供されます。

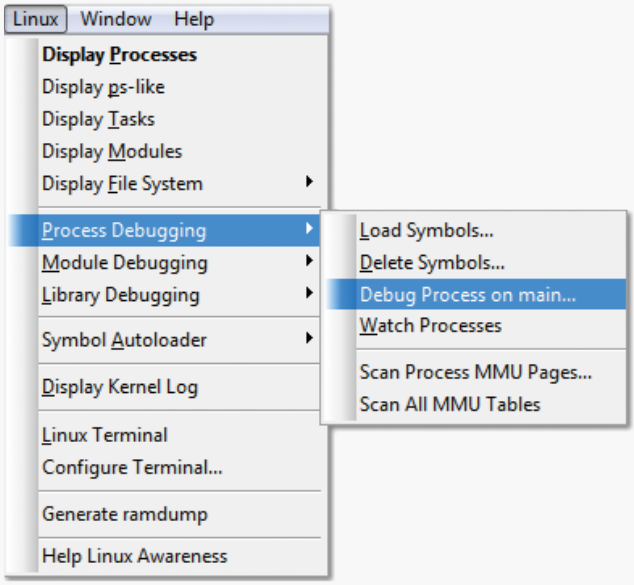
- シングルコアシステムの Linux 認識デバッグ
- SMP システムの Linux 認識デバッグ

ストップモードデバッグ

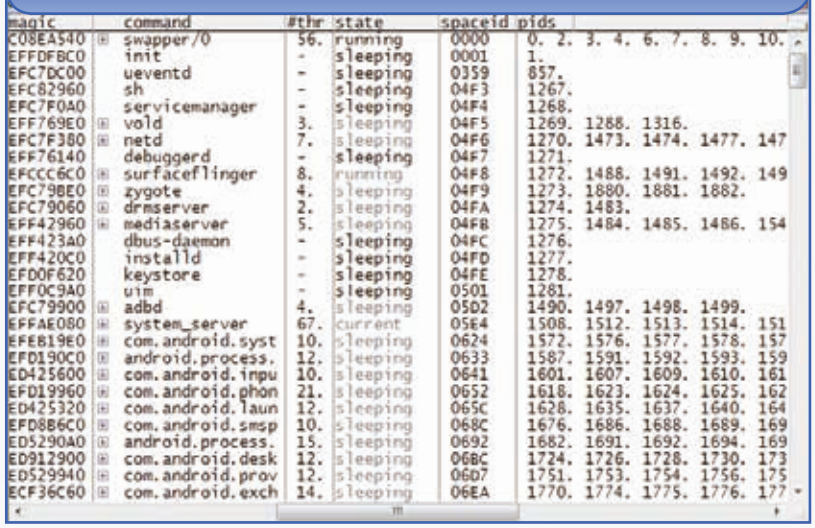
JTAG デバッガを使用して、ストップモードデバッグを行えます。ブレークポイントにヒットするたびに、プロセッサ、およびその結果としてシステム全体が停止します。これにより、特定時点におけるシステム全体の状態を解析できるようになります。

利点

- JTAG インタフェース接続でストップモードデバッグを行うことができ、リセットベクタからのデバッグも可能。
- TRACE32 デバッガでは Linux および MMU に対するサポートが提供されるため、カーネルおよびユーザランドのデバッグが可能。
- ソフトウェアが反応しない場合は、プロセッサを停止させて、コード内でプロセッサがクラッシュした箇所を見つけ出すことができる。
- プロセッサを停止することにより、カーネルまたは他のプロセスによる干渉を受けずに解析可能。

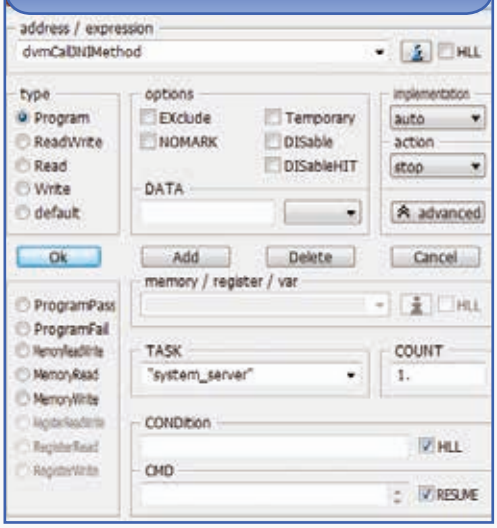


Linuxプロセスリスト表示



magic	command	#thr	state	spaceid	pids
C08EA540	swapper/0	56.	running	0000	0. 2. 3. 4. 6. 7. 8. 9. 10.
EFFDFBC0	init	-	sleeping	0001	1.
EFC7DC00	ueventd	-	sleeping	0359	857.
EFC82960	sh	-	sleeping	04F3	1267.
EFC7F0A0	servicemanager	-	sleeping	04F4	1268.
EFF769E0	vold	3.	sleeping	04F5	1269. 1288. 1316.
EFC7F380	netd	7.	sleeping	04F6	1270. 1473. 1474. 1477. 147
EFF76140	debuggerd	-	sleeping	04F7	1271.
EFCCC6C0	surfaceflinger	8.	running	04F8	1272. 1488. 1491. 1492. 149
EFC79BE0	zygote	4.	sleeping	04F9	1273. 1880. 1881. 1882.
EFC79060	drmserver	2.	sleeping	04FA	1274. 1483.
EFF42960	mediaserver	5.	sleeping	04FB	1275. 1484. 1485. 1486. 154
EFF423A0	dbus-daemon	-	sleeping	04FC	1276.
EFF420C0	installd	-	sleeping	04FD	1277.
EFD0F620	keystore	-	sleeping	04FE	1278.
EFF0C9A0	uim	-	sleeping	0501	1281.
EFC79900	adbd	4.	sleeping	05D2	1490. 1497. 1498. 1499.
EFFAE080	system_server	67.	current	05E4	1508. 1512. 1513. 1514. 151
EFEB19E0	com.android.syst	10.	sleeping	0624	1572. 1576. 1577. 1578. 157
EFD190C0	android.process.	12.	sleeping	0633	1587. 1591. 1592. 1593. 159
ED425600	com.android.inpu	10.	sleeping	0641	1601. 1607. 1609. 1610. 161
EFD19960	com.android.phon	21.	sleeping	0652	1618. 1623. 1624. 1625. 162
ED425320	com.android.laun	12.	sleeping	065C	1628. 1635. 1637. 1640. 164
EFD8B6C0	com.android.smsp	10.	sleeping	068C	1676. 1686. 1688. 1689. 169
ED5290A0	android.process.	15.	sleeping	0692	1682. 1691. 1692. 1694. 169
ED912900	com.android.desk	12.	sleeping	06BC	1724. 1726. 1728. 1730. 173
ED529940	com.android.prov	12.	sleeping	06D7	1751. 1753. 1754. 1756. 175
ECF36C60	com.android.exch	14.	sleeping	06EA	1770. 1774. 1775. 1776. 177

プロセス認識ブレークポイント



ランモードデバッグ

ランモードデバッグは、GDB プロトコルを介したデバッグです。選択したプロセスのみを停止させるため、カーネルや他のプロセスは影響を受けません。

Linuxプロセスリスト表示

name	id	space		sel	stop
OMAP_UART3:	403.	403.	0x0193		
kpsmoused:	489.	489.	0x01E9		
irq/363-rtc0:	495.	495.	0x01EF		
kworker/u2:	513.	513.	0x0201		
sieve:	545.	545.	0x0221		●
gdbserver:	546.	546.	0x0222		
hello:	549.	549.	0x0225	√	●

利点

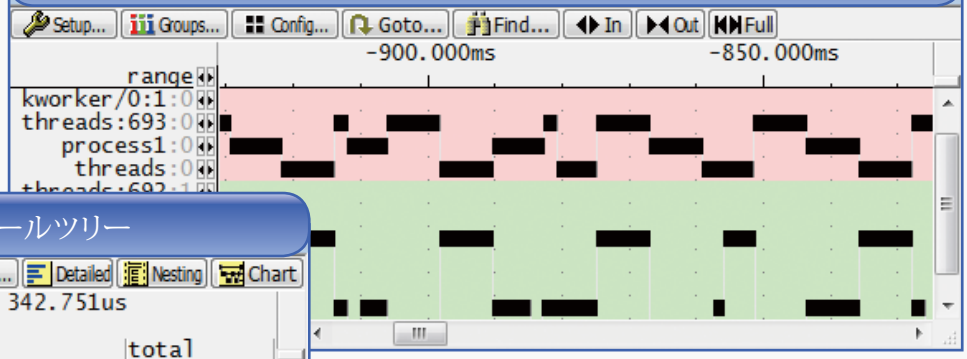
- 純粋なアプリケーションプロセスデバッグに最適。
- 通信インタフェースはアクティブなまま（Ethernet、RS232 など）。
- TRACE32 では、ランモードデバッグとストップモードデバッグを同時に実行できる特別な機能も提供。

Linux認識トレース

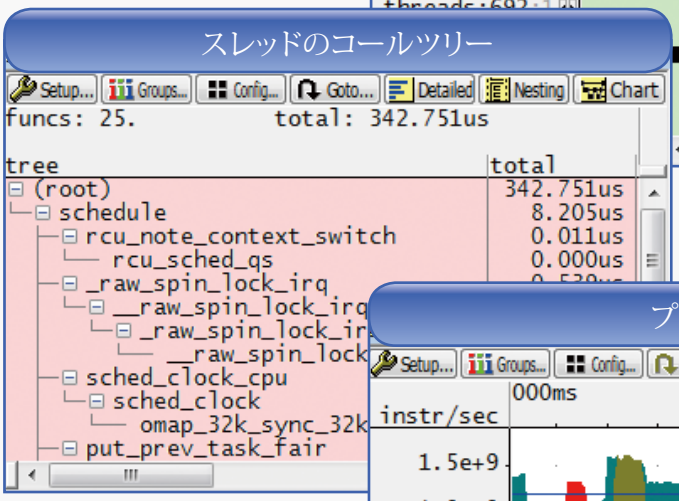
トレーステクノロジにより、組み込みシステムの動作およびタイミング特性を詳細まで解析できるようになります。コアトレースモジュールでは、関連コアの命令実行および実行プロセスに関する情報を生成します。

パラレルまたはシリアルのオフチップトレースポートにより、TRACE32 トレースツールで、この情報を Linux 対応トレース解析用に記録できるため、効果的なトラブルシューティング、包括的なプロファイリング、品質保証が可能になります。

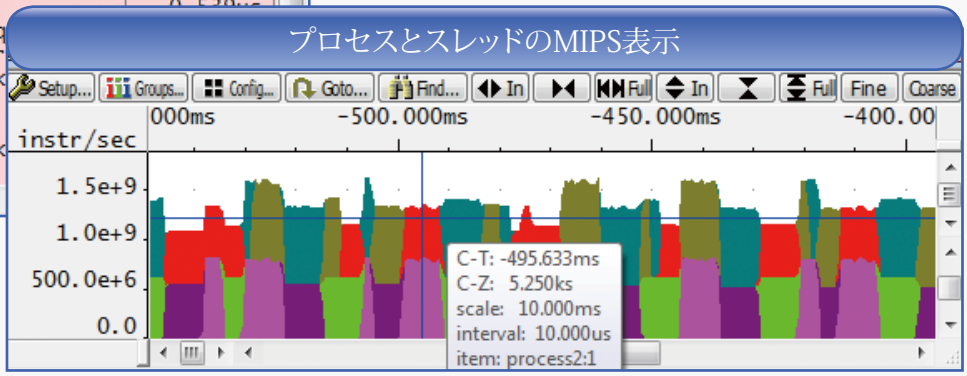
プロセスとスレッドのタイムチャート



スレッドのコールツリー



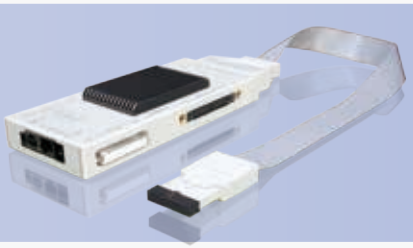
プロセスとスレッドのMIPS表示



ハードウェアベースデバッグツール

Power Debug

JTAG経由のデバッガ
主にストップモードデバッグ
ランモードデバッグとストップモードデバッグの同時実行も可能



リアルタイムトレース
Linux認識でのシングルコアおよびマルチコアトレース
ストップモードデバッグのみ

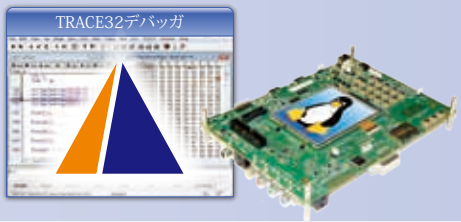
Power Trace

ソフトウェアのみの製品

全製品で共通のユーザインタフェース

Front-End

仮想ターゲット用のデバッガ
主にストップモードデバッグ
ランモードデバッグも可能



GDBプロトコルを介したデバッガ
ランモードデバッグ
GDBおよびKGDB

Front-End

詳しい情報はホームページをご覧ください。http://www.jp.lauterbach.com/rtoslinux.html

日本ローターバッハ
DEVELOPMENT TOOLS
日本ローターバッハ株式会社
〒222-0033 横浜市港北区新横浜3-8-8 日総第16ビル4F TEL.045-477-4511 FAX.045-477-4519