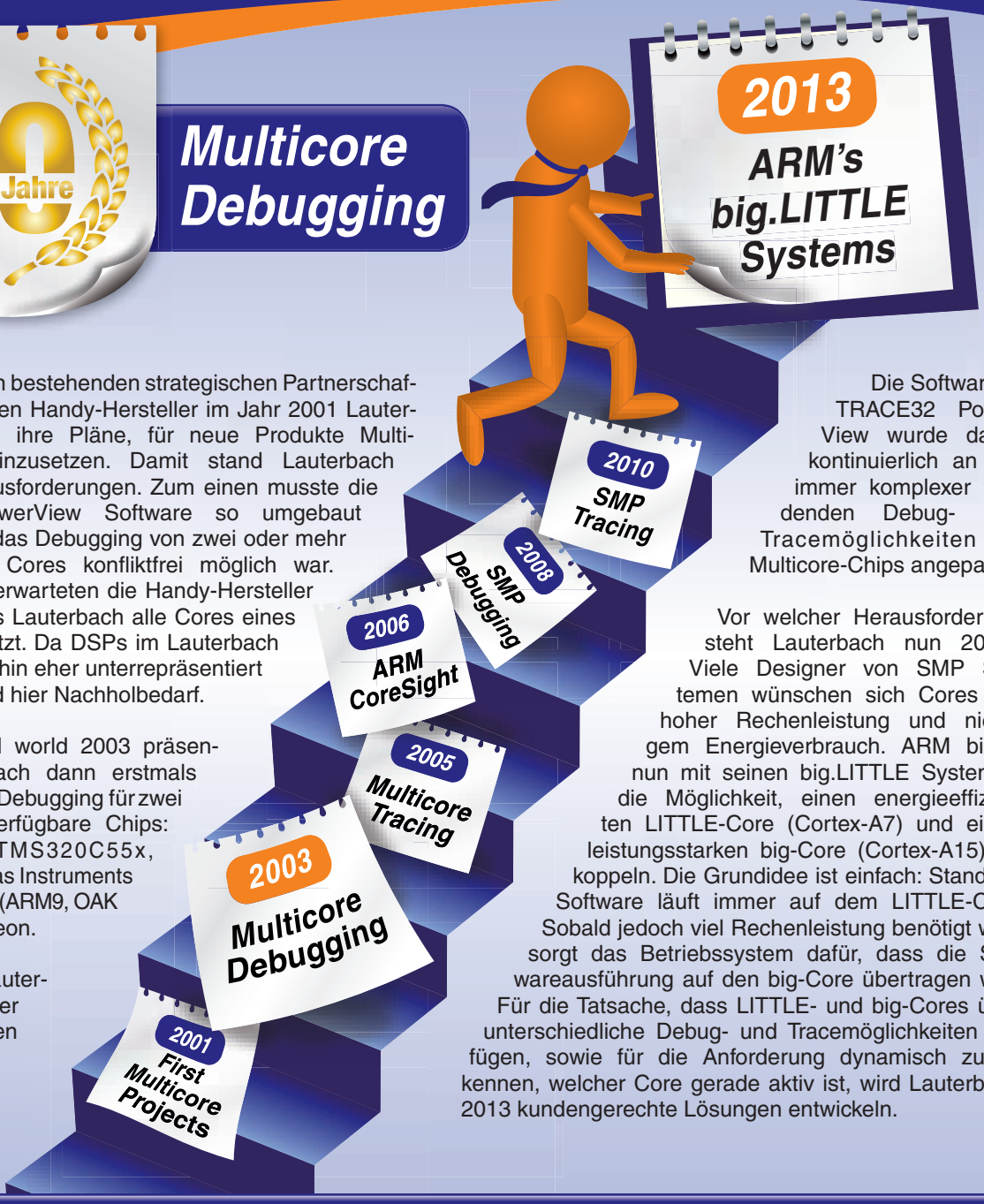
LAUTERBACH
DEVELOPMENT TOOLS

NEWS 2013

10 Jahre

Multicore Debugging



Im Rahmen von bestehenden strategischen Partnerschaften präsentierten Handy-Hersteller im Jahr 2001 Lauterbach erstmals ihre Pläne, für neue Produkte Multicore-ASICs einzusetzen. Damit stand Lauterbach vor zwei Herausforderungen. Zum einen musste die TRACE32 PowerView Software so umgebaut werden, dass das Debugging von zwei oder mehr daisy-chained Cores konfliktfrei möglich war. Zum anderen erwarteten die Handy-Hersteller ganz klar, dass Lauterbach alle Cores eines Chips unterstützt. Da DSPs im Lauterbach Portfolio bis dahin eher unterrepräsentiert waren, bestand hier Nachholbedarf.

Zur embedded world 2003 präsentierte Lauterbach dann erstmals sein Multicore-Debugging für zwei kommerziell verfügbare Chips: OMAP1510 (TMS320C55x, ARM9) von Texas Instruments sowie S-GOLD (ARM9, OAK DSP) von Infineon.

Seitdem hat Lauterbach viele seiner Kunden bei ihren Multicore-Projekten begleitet.

Die Software TRACE32 PowerView wurde dabei kontinuierlich an die immer komplexer werdenden Debug- und Tracemöglichkeiten der Multicore-Chips angepasst.

Vor welcher Herausforderung steht Lauterbach nun 2013? Viele Designer von SMP Systemen wünschen sich Cores mit hoher Rechenleistung und niedrigem Energieverbrauch. ARM bietet nun mit seinen big.LITTLE Systemen die Möglichkeit, einen energieeffizienten LITTLE-Core (Cortex-A7) und einen leistungsstarken big-Core (Cortex-A15) zu koppeln. Die Grundidee ist einfach: Standard-Software läuft immer auf dem LITTLE-Core. Sobald jedoch viel Rechenleistung benötigt wird, sorgt das Betriebssystem dafür, dass die Softwareausführung auf den big-Core übertragen wird. Für die Tatsache, dass LITTLE- und big-Cores über unterschiedliche Debug- und Tracemöglichkeiten verfügen, sowie für die Anforderung dynamisch zu erkennen, welcher Core gerade aktiv ist, wird Lauterbach 2013 kundengerechte Lösungen entwickeln.

INHALTSVERZEICHNIS NEWS 2013

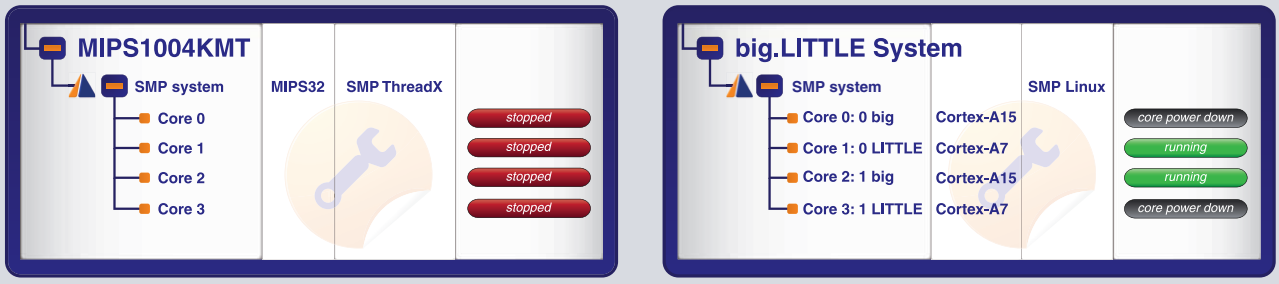
TRACE32 Multicore-Strategie

Seit mehr als 10 Jahren unterstützt Lauterbach das Debuggen und Tracen von Multicore-Chips.

Flexibilität

Ein bewährter Lauterbach Anspruch ist, seine TRACE32 Hard- und Software flexibel zu gestalten. Jede Core-Kombination, jede Multicore-Topologie und jede beliebig komplexe Debug- und Traceinfrastruktur kann von TRACE32 unterstützt werden. Flexibilität bedeutet in diesem Rahmen aber auch, dass TRACE32 das Debuggen und Tracen sowohl von AMP als auch von SMP Systemen unterstützt. Eine Übersicht über die wichtigsten Unterscheidungsmerkmale zwischen beiden Systemen bieten die Tabellen auf Seite 2 und 3.

SMP multicore configurations



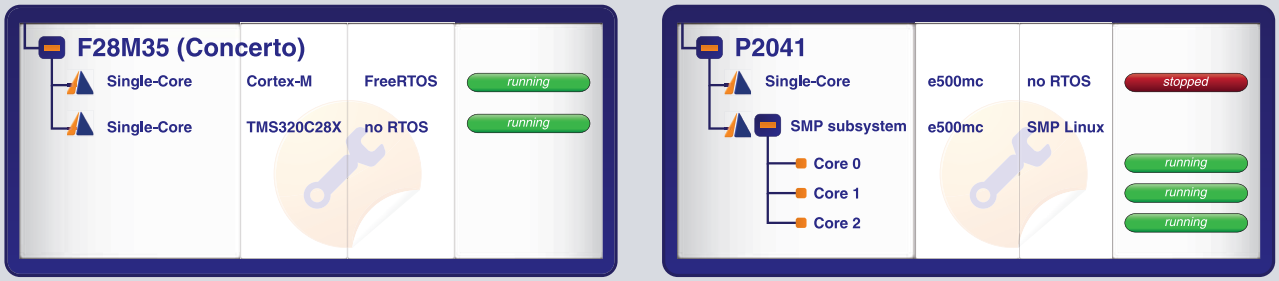
SMP System — Symmetric MultiProcessing	
Systemaufbau	Ein SMP System besteht aus zwei oder mehreren Cores. Diese sind in der Regel identisch oder zumindest Instruction Set kompatibel.
Aufgabenzuteilung/ Betriebssystem	Ein einzelnes SMP-Betriebssystem weist den Cores ihre Aufgaben zu (dynamisch oder statisch).
Anzahl der TRACE32 Instanzen	Für das Debugging eines SMP Systems wird nur eine TRACE32 Instanz gestartet. Diese kontrolliert alle Cores und visualisiert alle Informationen.
Synchrones Starten und Stoppen der Cores	Alle Cores werden synchron gestartet und gestoppt.
Onchip Breakpoints	Onchip Breakpoints werden parallel in die Debug-Register aller Cores programmiert.
Trace-Filter und Trigger	Trace-Filter und Trigger werden parallel in die Trace-Register aller Cores programmiert.
Tracedarstellung	Traceinformationen lassen sich für alle Cores gemeinsam oder pro Core darstellen.
Profiling	Laufzeitmessungen lassen sich pro Core oder für das Gesamtsystem durchführen.

70+ unterstützte Prozessorarchitekturen

Ein ebenfalls wichtiger Lauterbach Grundsatz ist, eine Vielzahl von Prozessorarchitekturen zu unterstützen: Standardcores, DSPs, FPGAs mit Embedded Softcores, konfigurierbare Cores. Dabei wird heute jeder neue Core zuerst so in TRACE32 integriert, dass er als Single-Core Subsystem in einem AMP System debugbar ist.

Das SMP Debugging und Tracing für eine Architektur wird nachgezogen, sobald der erste SMP-fähige Chip angekündigt ist. Der Anpassung der TRACE32 OS-Awareness kommt dabei eine besondere Bedeutung zu. Ob das SMP-Betriebssystem die Prozesse den Cores zur Laufzeit dynamisch zuordnet oder ob diese Zuordnung teilweise oder vollständig statisch erfolgt, wird dabei berücksichtigt.

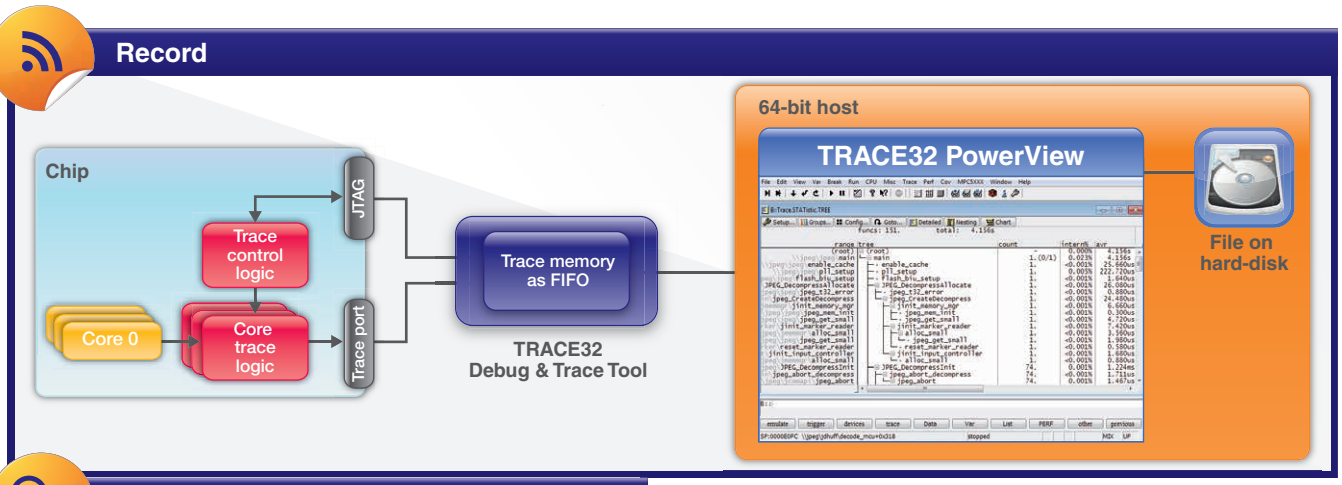
AMP multicore configurations

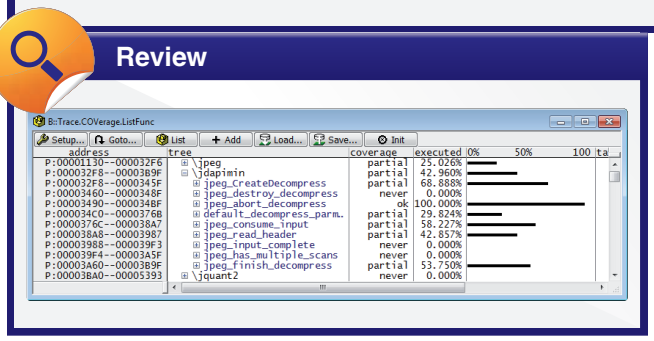


AMP System — Asymmetric MultiProcessing	
Systemaufbau	Ein AMP System besteht aus mehreren Subsystemen: aus einzelnen Cores und/oder SMP Systemen.
Aufgabenzuteilung/ Betriebssystem	In der Designphase werden die Aufgaben den Subsystemen fest zugeordnet. Ein Betriebssystem kontrolliert höchstens ein Subsystem.
Anzahl der TRACE32 Instanzen	Für das Debugging eines AMP Systems werden zwei oder mehr TRACE32 Instanzen gestartet. Jede TRACE32 Instanz kontrolliert ein komplettes Subsystem und visualisiert dessen Informationen.
Synchrones Starten und Stoppen der Cores	Alle Subsysteme können synchron gestartet und gestoppt werden (konfigurierbar).
Onchip Breakpoints	Onchip Breakpoints werden für jedes Subsystem unabhängig programmiert.
Trace-Filter und Trigger	Trace-Filter und Trigger werden für jedes Subsystem unabhängig programmiert.
Tracedarstellung	In einer TRACE32 Instanz werden die Traceinformationen für all die Cores dargestellt, die diese Instanz kontrolliert.
Profiling	In einer TRACE32 Instanz lassen sich Laufzeitmessungen für all die Cores durchführen, die diese Instanz kontrolliert. Ein globaler Zeitstempel ermöglicht die direkte Darstellung des zeitlichen Zusammenhangs zwischen den Subsystemen.

Code Coverage: Ergebnisse dokumentieren

Seit November 2012 bietet TRACE32 PowerView neue Möglichkeiten die Ergebnisse der Code Coverage Analyse zu dokumentieren. Neu hinzu gekommen sind eine Kommentierfunktion sowie ein XML-Export.





Trace-basiertes Code Coverage

Für die Qualitätssicherung von sicherheitsrelevanten Produkten ist ein Nachweis von *Statement Coverage* und *Decision Coverage* oft zwingend vorgeschrieben. Für viele Embedded Systeme ist zudem gefordert, dass hochoptimierter Code in Echtzeit getestet wird. Code-instrumentierung und Laufzeitverfälschung sind hier tabu. Um seinen Kunden einen Nachweis von *Statement Coverage* und *Decision Coverage* zu ermöglichen, bietet Lauterbach ein so genanntes Trace-basiertes Code Coverage. Dazu muss der verwendete Prozessor bzw. Multicore-Chip jedoch folgende Voraussetzungen erfüllen:

Die eingesetzten Cores müssen eine Tracelogik besitzen, die Informationen über die Instruktionsausführung auf den Cores generiert. Gleichzeitig muss der Prozessor bzw. Multicore-Chip über einen Traceport mit ausreichender Bandbreite verfügen, damit die Traceinformationen vollständig durch ein externes Tool aufgezeichnet werden können.

Bei durchschnittlichen Traceport-Datenraten von bis zu 60 MByte/s lassen sich die Tracedaten zur Aufzeichnungszeit auf den Hostrechner streamen. Damit können pro Testlauf mehrere TByte Traceinformationen erfasst werden.

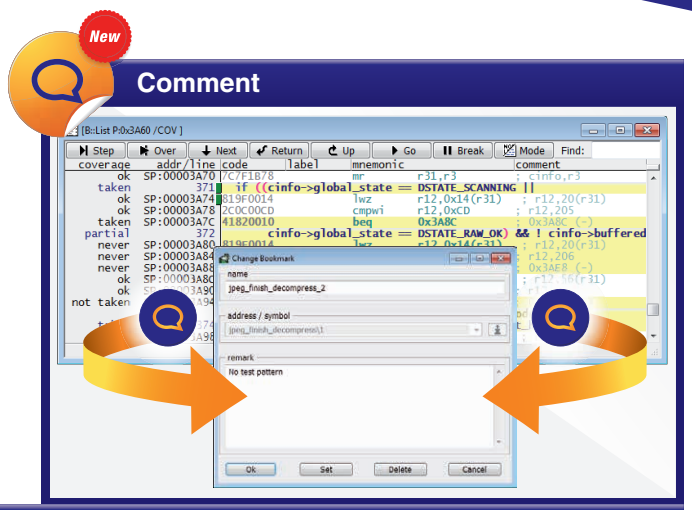
Da die Traceinformationen auf Assemblerebene vorliegen, können folgende Nachweise erfolgen:

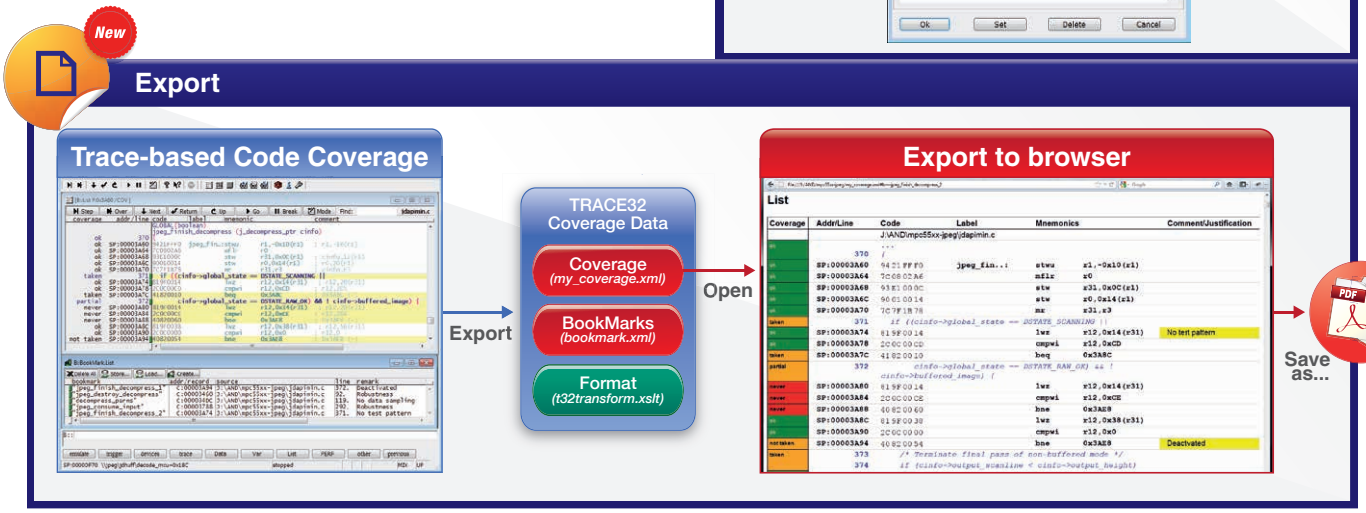
- **Object Statement Coverage** weist nach, dass jede Assemblerzeile während des Systemtests mindestens einmal ausgeführt wurde.
- **Object Branch Coverage** weist für bedingte Sprünge nach, dass jeder Sprung mindestens einmal genommen (taken) sowie mindestens einmal nicht genommen (not taken) wurde.

Für die Hochsprachen-Codezeilen lassen sich *Statement Coverage* und *Decision Coverage* einfach aus dieser Analyse ableiten und darstellen.

Kommentierfunktion

Um nachzuweisen, dass ein Embedded System alle Requirements vollständig und korrekt umsetzt, werden in der Regel Testfälle entwickelt. Diese bilden die Basis für den Systemtest. Um die Daten für die Code Coverage Analyse zu sammeln, zeichnet





das Tracetool während des gesamten Systemtests alle Informationen zur Instruktionsausführung auf (**Record**).

Die aufgezeichneten Traceinformationen werden von TRACE32 PowerView in einer Code Coverage Datenbank verwaltet.
Über diese werden zahlreiche Funktionen zur Überprüfung der Codeabdeckung angeboten (**Review**).

Für alle Instruktionen, die nach dem Testlauf als „nicht vollständig ausgeführt“ markiert sind, muss der Tester Folgendes entscheiden:

- Soll der nicht ausgeführte Codebereich ein Requirement abdecken? Dann muss ein passender Testfall für den nächsten Durchlauf des Systemtests bereitgestellt werden.

- Soll der nicht ausgeführte Codebereich ein Requirement abdecken, das in der aktuellen Systemkonfiguration jedoch nicht testbar ist, kann die neue TRACE32 Kommentierfunktion verwendet werden, um die Nicht-Ausführung dieses Codebereichs zu rechtfertigen (**Comment**).

- Handelt es sich um „toten Code“? Dann muss dieser aus der Software entfernt werden.

Export

Nach Abschluss des Systemtests müssen die Ergebnisse der Code Coverage Analyse natürlich dokumentiert werden. Neu unterstützt TRACE32 PowerView dazu einen Export im XML-Format. Exportiert werden:

1. Der Assembler- und Hochsprachencode zusammen mit den Markierungen der Code Coverage Analyse (my_coverage.xml).

2. Die Kommentare, die begründen, warum einzelne Codeabschnitte zulässig sind – obwohl sie während des Tests nicht ausgeführt wurden (bookmark.xml).

Für eine intuitive Darstellung der Ergebnisse der Code Coverage Analyse in einem Web-Browser stellt Lauterbach ein Formatierungsfile bereit (t32transform.xslt). Bei Bedarf können die Ergebnisse aus dem Browser heraus auch als PDF-File abgespeichert werden.

New

Prozessoren/Chips

Altera	**Cortex-A/-R** • Cyclone V SoC
Analog Devices	**Cortex-M** • ADuCM36x
AppliedMicro	**PPC40x** • PPC405EX, PPC405EXr **PPC44x** • SMP für APM PacketPro
ARM	**Cortex-A5x (ARMv8)** • Cortex-A53 • Cortex-A57
Atmel	**Cortex-M** • ATSAM4
Axis	**MIPS32** • ARTPEC-4
Broadcom	**MIPS32** • BCM47186 • BCM6318, BCM6828 • BCM7346, BCM7356 • BCM7418, BCM7425
BroadLight	**MIPS32** • BL25580
CEVA	**CEVA-X** • CEVA-XC323 **TeakLite-III** • CEVA-TeakLite-4
Energy Micro	**Cortex-M** • EFM32LGxxx, EFM32WGxxx • EFM32ZGxxx
Freescale	**ColdFire+/V1** • MCF51AC/AG/CN/EM • MCF51JE/JM/MM/QE • MCF51JF/JU/QM/QU **Cortex-A/-R** • Vybrid F Series **Cortex-M** • Kinetis L • Vybrid Series **MPC85XX/QorIQ e500** • P1010, P1012, P1014 • P1017, P1021, P1023 **QorIQ 32-Bit** • P2040, P2041 **QorIQ 64-Bit** • B4220, B4420, B4860 • P5021, P5040, T10XX • T2080, T2081, T4160, T4240
Freescale (Forts.)	**PX-Series** • PXD1005, PXD1010, PXD2020 • PXN2020, PXN2120, PXR40xx • PXS2005, PXS2010, PXS30xx **Qorivva MPC5xxx** • MPC5743K, MPC5744K • MPC5744P, MPC5746M, • MPC5748G, MPC5777M **S12Z** • S12ZVH, S12ZVM **StarCore** • B4220, B4420, B4860
Hilscher	**ARM9** • NETX 51
Infineon	**Cortex-M** • XMC4000 Family • TC2D5T/D7T, TC2D5TE/D7TE • TC275T/277T, TC275TE/277TE **TriCore** • TC2D5T/D7T, TC2D5TE/D7TE • TC275T/277T, TC275TE/277TE
Intel®	**Atom™/x86** • Atom Z2460/CE2600/N2800 • Core i3/i5/i7 3rd Generation
Marvell	**ARM11** • MV78130v6, MV78160v6 • MV78230v6, MV78260v6 **Cortex-A/-R** • MV78130v7, MV78160v7 • MV78230v7, MV78260v7
Mobileye	**MIPS32** • EyeQ3
NEC	**MIPS32** • EMMA3 Series
NVIDIA	**Cortex-A/-R** • TEGRA 3
NXP	**Beyond** • JN5168 **Cortex-M** • LPC43xxx, LPC800
Renesas	**Cortex-A/-R** • R-Car H1 **MIPS32** • RT3352 **RH850** • RH850/E1x, RH850/F1x **RL78** • RL78D1A/F1x/G1x/I1A/Lxx **RX** • RX630, RX631, RX63N **SH** • SH7267

New Prozessoren/Chips

Renesas (Forts.)	**V850** • V850E2/Fx4-L • V850E2/Mx4 Multicore
Samsung	**Cortex-A/-R** • Exynos 4212, Exynos 4412 • Exynos 5250 • S5PV210
Sigma Designs	**MIPS32** • SMP8634, SMP8654
ST-Ericsson	**Cortex-A/-R** • DB8540 **MMDSP** • DB8540
STMicro-electronics	**Cortex-A/-R** • SPEAr1310, SPEAr1340 **Cortex-M** • STM32 F3, STM32 F4
STMicro-electronics (Forts.)	**SPC5xx** • SPC56AP60, SPC56AP64 • SPC560P54, SPC560P60 • SPC574K70, SPC574K72 • SPC574L74, SPC57EM80 • SPC57HM90
Synopsys	**ARC** • ARC-EM 1.1
Texas Instruments	**Cortex-A/-R** • RM4 Series **Cortex-M** • F28M35 Concerto • LM4F Series **MSP430** • MSP430FR5xx **TMS320C28X** • C28346 • F28022, F28027, F28M35 **TMS320C55X** • C5535 **TMS320C6x00** • C6655, C6657, C6713

UEFI-Debugging für ARM

Lauterbach hat 2012 seinen Support für das Debugging von UEFI BIOS weiter ausgebaut. Unterstützt werden jetzt:

- UEFI BIOS InsydeH2O für Atom und x86
- UEFI BIOS Intel BLDK für Atom und x86 New
- UEFI BIOS TianoCore für ARM/Cortex New

Aktiviert wird das UEFI-Debugging durch das Nachladen einer so genannten TRACE32 Extension. Ausführliche Informationen zum UEFI-Debugging finden Sie unter: **www.lauterbach.com/uefi.html**

Target-OS Erweiterungen

- FreeRTOS für Beyond und ColdFire
- Linux für Beyond und x86 64-bit
- OSEK/ORTI SMP
- QNX für x86
- Quadros für CEVA-X
- RTX-ARM v4
- SMX für ColdFire
- SYS/BIOS für TMS320C6x00
- VxWorks für x86
- µC/OS-II für TMS320C28X
- µC/OS-III für SH

New Target-OS

DEOS für PowerPC	verfügbar
Linux für ARMv8 (64-Bit)	geplant
OKL4 5.0 für ARM	verfügbar
Windows Standard (XP, Vista, Win7, Win8) für x86 32/64-Bit	geplant
µT-Kernel für ARM	verfügbar

embedded world 2013
Exhibition & Conference
Halle 4, Stand 210
26.-28. Februar 2013

µTrace für Cortex™-M Familie

Ab Mai 2013 bietet Lauterbach einen kostengünstigen Debugger für die Cortex-M Familie an. Grund für die Entwicklung eines separaten Tools ist der hohe Marktdurchsatz der Cortex-M Prozessoren. Angeboten wird eine **All-in-One Lösung** mit den folgenden Features:

Kenndaten µTrace

- Support für mehr als 1000 verschiedene Cortex-M Prozessoren
- USB 3 Schnittstelle zum Hostrechner
- Standard JTAG, Serial Wire Debug und cJTAG
- 256 MByte Tracespeicher
- 34-Pin Half-Size Stecker zur Zielhardware, Adapter für viele andere Stecker verfügbar
- Spannungsbereich 0.3 V bis 3.3 V, 5 V tolerant

Debug Features

- C/C++ Debugging
- Einfache und komplexe Breakpoints
- Speicher schreiben und lesen, während das Programm läuft.
- Flash Programmierung
- OS-aware Debugging
- Multicore Debugging von zwei oder mehr Cortex-M Cores

Trace Features

- 4-Bit ETMv3 im Continuous Mode
- ITM über TPIU und Serial Wire Output
- Multicore Tracing
- Endlos Trace durch Streamen der Traceinformation auf den Hostrechner
- Analyse von Task- und Funktionslaufzeiten
- Code Coverage Analyse
- Trace-Auswertung schon zur Aufzeichnungszeit (Real-time Profiling)
- Energiemessung über Analog Probe

Bedient wird der µTrace wie alle anderen Lauterbach Produkte über die TRACE32 PowerView GUI.

ARM CONNECTED

BENACHRICHTIGEN SIE UNS

Falls sich Ihre Adresse geändert hat oder Sie kein Mailing mehr von uns erhalten möchten, schicken Sie bitte einfach eine kurze E-Mail an:
mailing@lauterbach.com