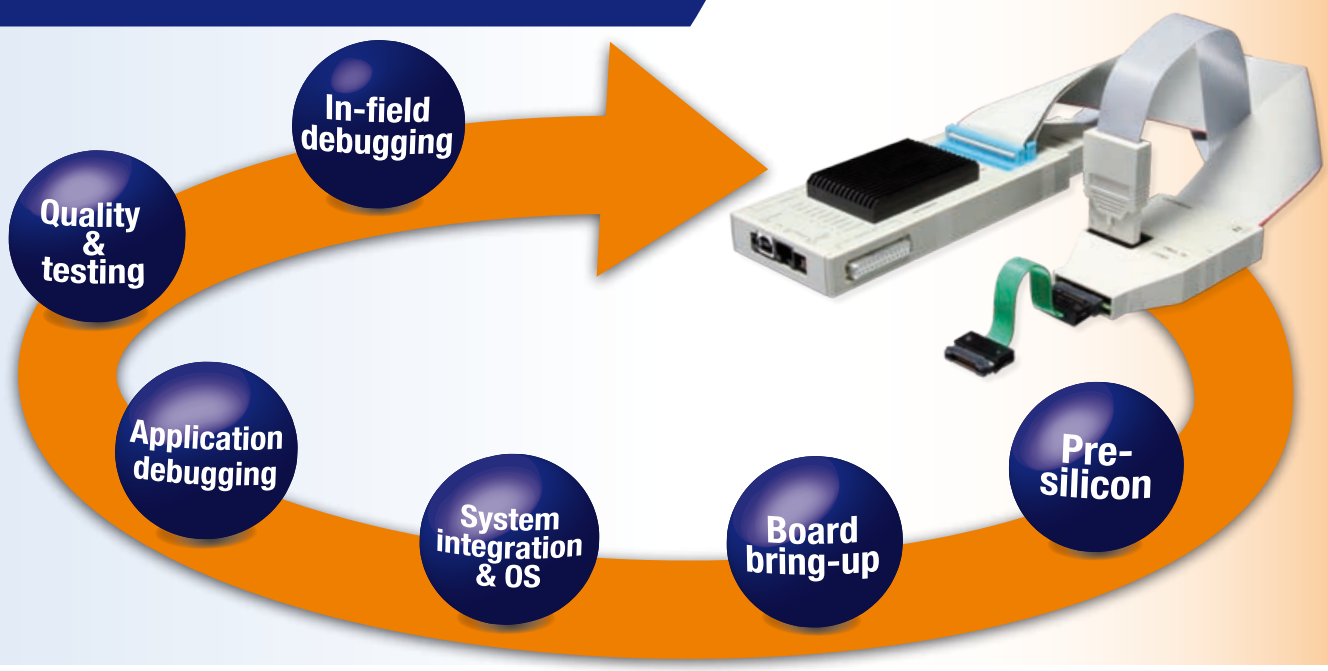
NEWS 2012

LAUTERBACH DEVELOPMENT TOOLS

DEBUGGER, REAL-TIME TRACE, LOGIC ANALYZER



Un Debugger per tutte le fasi di progetto

I progetti embedded stanno diventando sempre più complessi e si riduce sempre più il time-to-market. Per contrastare queste sfide, molti project manager si affidano a strumenti di debug e trace in grado di supportare gli sviluppatori lungo tutte le fasi di progetto.

TRACE32, la famiglia di sistemi di debug e trace di Lauterbach, garantisce un approccio coerente al mondo del debug e fornisce un ambiente operativo che può essere esteso mediante script personalizzabili dall'utente. In questo modo si riduce il tempo di apprendimento e aumenta la quantità di lavoro effettivo sul progetto. Ci sono ormai molti sviluppatori che hanno acquisito un'esperienza pratica su TRACE32 in oltre 10 anni di lavoro. E dunque, cosa rende TRACE32 così diverso dagli altri debugger?

- Sistema di sviluppo hardware e software
- Supporto ai nuovi processori sin dal primo rilascio
- Ampia gamma di processori supportati
- Innumerevoli funzionalità di test e analisi
- Continuità di integrazione con la tool-chain di sviluppo embedded

Sistemi di sviluppo Hardware e Software

L'attività principale di Lauterbach consiste nel progettare e produrre strumenti di debug e trace basati sull'hardware. Inoltre Lauterbach offre anche analizzatori di stati logici da più di 20 anni. La caratteristica principale dei logic analyzer TRACE32 è la loro capacità di integrarsi senza soluzione di continuità con i sistemi hardware di debug e trace. Un'applicazione tipica che utilizza il logic analyzer integrato in PowerTrace II è riportata a pagina 6 "Verifica dei segnali JTAG".

Avere a disposizione computer veloci ed efficienti permette di svolgere la maggior parte delle simulazioni e delle verifiche su PC e workstation. Nel mondo embedded è ormai diventato normale sviluppare software su Virtual Targets prima della disponibilità del silicio. Per questa fase di progetto Lauterbach è in grado di fornire soluzioni puramente software. »

INDICE

Virtual Targets

Al giorno d'oggi i virtual targets sono sempre più utilizzati per iniziare lo sviluppo software molto prima che sia disponibile il primo prototipo hardware. Con la disponibilità di un virtual target, si può già iniziare il debug dei driver, del sistema operativo e dell'applicazione.

La maggior parte dei virtual targets hanno API specifiche per debug e trace. Altrimenti si può usare la MCD-API standard (**http://www.lauterbach.com/mcd_api.html**). Molti progetti oggi usano chip multicore. Di conseguenza, sin dal 2011 Lauterbach ha esteso su virtual targets il supporto al debug multicore.

Validazione Pre-Silicio

Per i produttori di semiconduttori è importante validare il progetto dei loro processori o SoCs prima che vengano effettivamente prodotti. Le singole parti devono essere testate accuratamente, per esempio: l'interfaccia JTAG, il core nella sua globalità o l'interazione fra il core e la periferia.

Per questo tipo di test, si usa tradizionalmente un emulatore per silicio (ad esempio Palladio) o prototipi di FPGA collegati a sistemi di debug hardware TRACE32. Ne risulta una velocità di esecuzione molto più bassa rispetto ai processori reali.

Oggi si può eseguire una prima validazione di modelli Verilog o SystemC direttamente su PC o workstation. Non si possono usare debugger hardware in una pura validazione software. Per questo motivo Lauterbach nel 2011 ha aggiunto al suo software un Back-End Verilog, che simula un'interfaccia JTAG a livello dei segnali (vedere Figura 1).

L'integrazione dei sistemi TRACE32 per la validazione pre-silicio costituisce una parte importante del supporto ai più nuovi processori e SoCs sin dal primo rilascio:

- Sistemi di sviluppo già testati sono disponibili ben prima che il primo silicio lasci la fabbrica.
- È disponibile e accessibile al cliente una conoscenza approfondita del nuovo processore / SoC.
- Sono disponibili script di Startup per il debugger TRACE32.

60+ architetture di processori supportate

Lauterbach supporta tutti i più diffusi processori / SoCs del mercato embedded. Di fatto, Lauterbach è l'unica azienda a rilasciare sistemi di sviluppo per molti diversi core. Standard controllers, DSP, softcore FPGA, core configurabili: qualsiasi cosa può essere integrata in un chip multicore e debuggata con un sistema TRACE32.

Nel 2011 Lauterbach ha aggiunto il supporto di un gran numero di nuovi processori e chip multicore. Per una presentazione generale, vedere la tabella a pagina 4.

Funzionalità di Test e Analisi

Ogni fase di un progetto richiede specifiche funzionalità di test e analisi. Per questo la GUI PowerView di TRACE32 comprende un'ampia gamma di comandi e menu. Alcuni esempi di comandi di basso livello sono i comandi di Boundary Scan (vedere Figura 2), i comandi di identificazione dei core e quelli per manipolare i pin JTAG.

Nelle fasi di test e verifica della qualità, i comandi di alto livello forniscono supporto al progettista. Si tratta

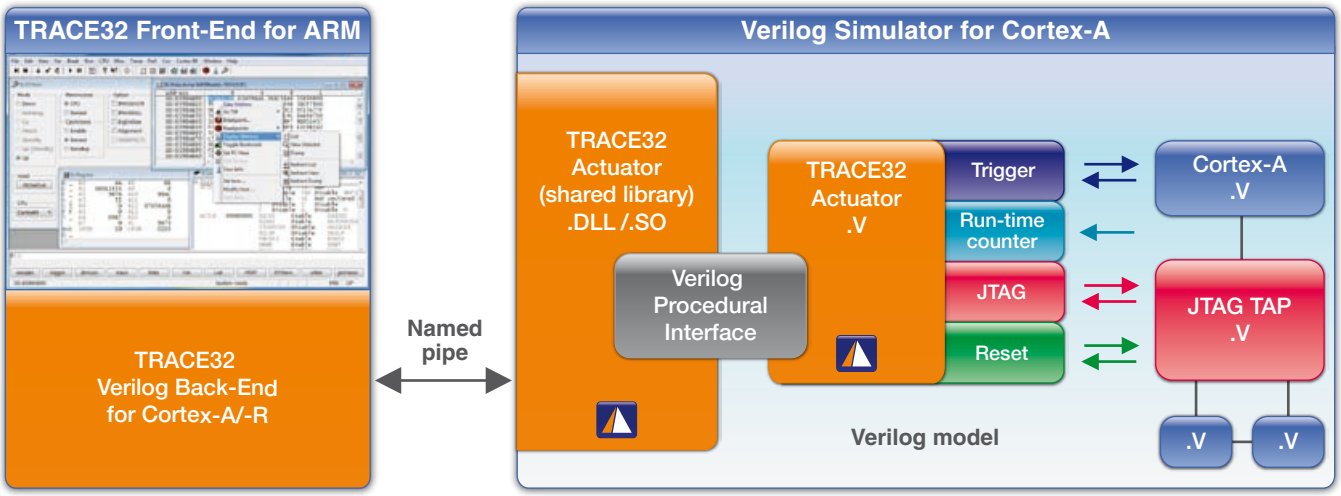


Fig. 1: Per ogni comando nel Front-End TRACE32, il Back-End Verilog genera i segnali JTAG per la validazione del modello.

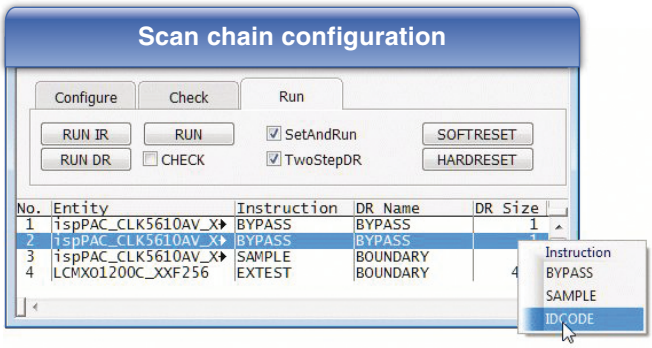


Fig. 2: I comandi di Boundary Scan sono disponibili per il commissioning dell'hardware.

tipicamente di analisi dei dati di trace. Alcuni esempi sono: misure di durata di una funzione, energy profiling o informazioni sul code coverage.

Sin dall'inizio del 2011, per la maggior parte delle principali architetture di processori, Lauterbach ha introdotto la gestione dello stream dei dati di trace in tempo reale su host computer. Ciò permette di raccogliere una gran quantità di dati diagnostici e semplifica il processo di verifica della qualità. Per maggiori informazioni, vedere l'articolo "Code Coverage semplificato" a pagina 7.

Integrazione nella Tool Chain Embedded

Il software TRACE32 ha un'architettura di tipo "open", per cui si integra perfettamente con i più comuni componenti di base di un progetto embedded. Questi comprendono:

- Sistemi operativi lato Host
- Linguaggi di programmazione e compilatori
- Sistemi operativi lato Target
- Virtual Machines come Dalvik VM Android

Le open-API TRACE32 rendono possibile l'integrazione con numerosi tools di terze parti, ad esempio IDE speciali come Eclipse, tools di programmazione grafica e di profiling esterno. Diversi nuovi sviluppi in quest'area sono stati aggiunti nel 2011.

Prism, il tool di parallelizzazione dell'azienda scozzese CriticalBlue, supporta gli sviluppatori che devono migrare il codice single-core su chip multi-core. Il tool permette di valutare diverse strategie di migrazione senza fare alcuna modifica al codice. Una volta definita la strategia ottimale, si può effettuare la parallelizzazione step by step, ancora con il supporto di Prism.

A partire da luglio 2011, Lauterbach ha introdotto la possibilità di esportare l'informazione di trace in formato Prism, permettendo così ai tools CriticalBlue di analizzare il trace registrato nell'effettivo flusso di codice.

L'articolo "Simulazione e realtà ora più vicine", a pagina 14, descrive accuratamente un'altra innovazione: l'integrazione fra Simulink® di MATLAB e TRACE32.

Lunga durata del Ciclo di vita

Quando introduce una nuova tecnologia, Lauterbach ha l'abitudine di garantire una lunga fase di transizione. Il cliente non è mai forzato ad accettare un cambio tecnologico mentre si trova nel mezzo di un progetto importante.

Per esempio, a partire da maggio 2012, Lauterbach introdurrà una versione QT della sua interfaccia grafica PowerView TRACE32 (vedere Figura 3). Con QT, sarà disponibile una GUI aggiornata per Linux, Mac OS X e altri sistemi operativi host.

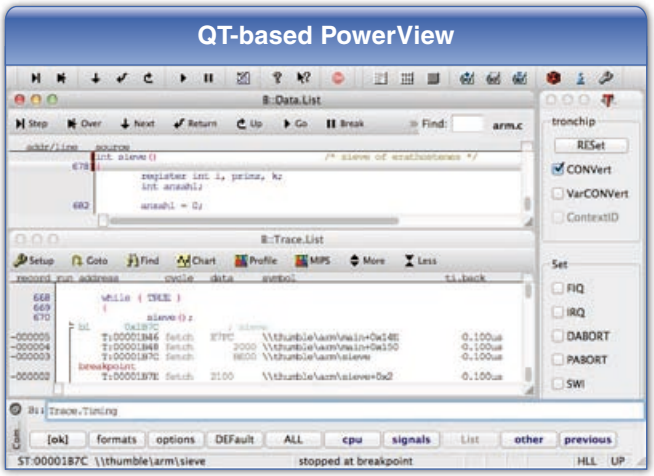


Fig 3: La nuova GUI basata su QT per Linux, Mac OS X e altri sistemi operativi.

Lauterbach continuerà a supportare la versione Motif di PowerView TRACE32 in modo che i clienti possano decidere il momento migliore per passare da una versione all'altra.

Embedded Software Engineering Kongress

Abendseminars freescale

MathWorks AUTOMOTIVE CONFERENCE 2012

In queste pagine delle NEWS 2012 troverete molte altre informazioni che potrebbero servirvi per i vostri progetti attuali o futuri. Speriamo che possiate trovare qualche prestazione che contribuisca al successo dei vostri progetti. Nel corso dell'anno ne mostreremo alcune dal vivo in eventi organizzati in Italia e in Europa (visitare **www.lauterbach.it**).

Nuovi Processori supportati

Nuovi derivati

Altera	**Cortex-A/-R** • FPGA Cortex-A9 MPCore con Hardcore **MIPS32** • MP32
AppliedMicro	**PPC44x** • 86290/491/791 Q2/2012
ARM	**Cortex-A/-R** • Cortex-A7/Cortex-A7 MPCore • Cortex-A15 • Cortex-A15 MPCore • Cortex-R5/Cortex-R5 MPCore • Cortex-R7/Cortex-R7 MPCore
Beyond Semiconductor	**Beyond** • BA22
Broadcom	**MIPS32** • BCM35230 • BCM63168, BCM63268 • BCM7231, BCM7358
Cavium	**MIPS64** • CN61XX/CN62XX/CN66XX • CN67XX/CN68XX
Ceva	**CEVA-X** • CEVA-XC
CSR	**ARM11** • QUATRO 4500
Cypress	**ARM9** • EZ-USB FX3
Energy Micro	**Cortex-M** • Giant Gecko
Freescale	**MCS08** • S08LG **MCS12X** • MC9S12VR, MC9S12XS • MM912F634 **Cortex-A/-R** • Serie i.MX 6 **MPC55xx/56xx** • MPC5604E, MPC5675K, • MPC5676R **QorIQ 32-Bit** • P1010, P1020 • P2040, P2041 • P3041, P4040, P4080 • PSC9131
Freescale (cont.)	**QorIQ 64-Bit** • P5010, P5020
Fujitsu	**Cortex-A/-R** • MB9DF126, MB9EF126
IBM	**PPC44x** • 476FP Q2/2012
Ikanos	**MIPS32** • Fusiv Vx185
Infineon	**TriCore** • Architettura TriCore Multi-Core
Intel®	**Atom™/x86** • Atom D2500, Atom N550 • Core i3/i5/i7 2° Generazione
Lantiq	**MIPS32** • XWAY xRX100
LSI	**PPC44x** • ACP344x Q2/2012
Marvell	**ARM9 Debug Cable** • 88E7251 **ARM11 Debug Cable** • 88AP610-V6, MV78460-V6 **Cortex-A/-R Debug Cable** • 88AP610-V7, MV78460-V7
Nuvoton	**Cortex-M** • NuMicro
NXP	**Cortex-M** • LPC12xx **Beyond** • JN5148
Qualcomm	**MIPS32** • AR7242 **Cortex-A/-R** • Krait
Renesas	**V850** • V850E2/Fx4: 70F3548..66 70F4000..70F4011 • V850E2/Fx4-L: 70F3570..89 • V850E2/Px4: 70F3503/05 70F3507/08/09 **78K0R/RL78** • 78K0R/Kx3-C/L • RL78/G14, RL78/G1A • RL78/F12, RL78/I1A **H8SX** • H8SX1725

Nuovi derivati

Renesas (Cont.)	**SH** • SH708x con AUD/Onchip-Trace • SH7147
Samsung	**ARM7** • S3F4 **Cortex-A/-R** • S5PV310 **Cortex-M** • S3FM, S3FN
ST-Ericsson	**Cortex-A/-R** • A9500, A9540, M7400 **MMDSP** • A9500, A9540
STMicro-electronics	**MPC55xx/56xx** • SPC56A80, SPC56HK **Cortex-M** • STM32F2xx, STM32F4xx
Synopsys	**ARC** • ARC EM4, ARC EM6
Tensilica	**Xtensa** • BSP3, LX4, SSP16
Texas Instruments	**MSP430** • CC430Fxxx, MSP430FR5xxx • MSP430x1xx..MSP430x6xx **ARM9** • AM38xx • OMAP4460/4470 • TMS320C6A81xx • TMS320DM81xx **Cortex-A/-R** • AM335x, AM38xx • OMAP4460/4470/543x • RM48L950 • TMS320C6A81xx • TMS320DM81xx • TMS570LS3xxx **Cortex-M** • AM335x • OMAP4460/4470/543x • TMS470MFxxx **TMS320C28X** • TMS320C28346/F28069 **TMS320C6x00** • OMAP4460/4470/543x • TMS320C6A81xx • TMS320DM81xx • TMS320TCI6616/18
Xilinx	**Cortex-A/-R** • Zynq7000

Trace Nexus per Cores a Package ridotto

La cella Nexus, integrata nei controllori delle famiglie Freescale MPC560xB/C o nei controllori ST SPC560B/C, può generare dati di trace delle istruzioni eseguite dal core. Se si utilizza un sistema operativo, vengono anche prodotte informazioni sullo switch dei task.

Per permettere a un trace-tool esterno, come TRACE32, di registrare i dati di trace, un microcontrollore deve avere un'interfaccia di trace. Tuttavia i membri della famiglia MPC560xB/C non hanno questa interfaccia nei package standard. Per consentire l'accesso a questi importanti dati sul flusso di esecuzione del programma, durante la fase di sviluppo vengono proposti microcontrollori compatibili a livello di silicio in package BGA 208-pin, che hanno un' interfaccia Nexus con 4 pin MDO (Message Data Out).

Sin dalla metà del 2011, Lauterbach ha reso disponibili adattatori per MPC560xB/C in grado di rimpiazzare il controllore originale sulla scheda hardware con controllori a 208 pin con interfaccia Nexus.

L'adattatore MPC560xB/C è formato da un opportuno controllore MPC560xB/C in package BGA 208 pin e da un plug Mictor con interfaccia Nexus per collegare i sistemi di trace TRACE32 (in blu in Figura 4). Occorre inoltre un socket adapter Tokyo Eletech.

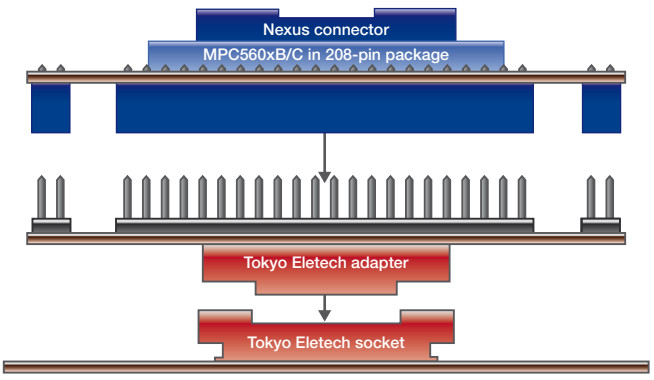


Fig. 4: L'adattatore MPC560xB/C permette di usare un package di sviluppo con interfaccia Nexus al posto del controllore originale.

Verifica dei segnali JTAG

PowerTrace II di Lauterbach è equipaggiato con un logic analyzer e fornito di un probe digitale standard, che permette di registrare 17 canali digitali con una velocità di campionamento fino a 200MHz.

Il logic analyzer ha una capacità di memorizzazione pari a 1024K campioni. Un possibile esempio di utilizzo potrebbe essere il test dei segnali JTAG durante la validazione pre-silicio (vedere Figure 6 e 7).

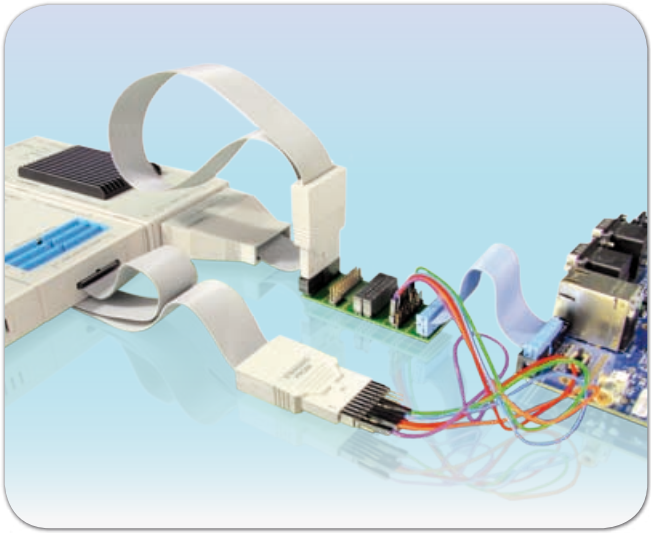


Fig. 5: Schema delle connessioni per registrare i segnali JTAG.

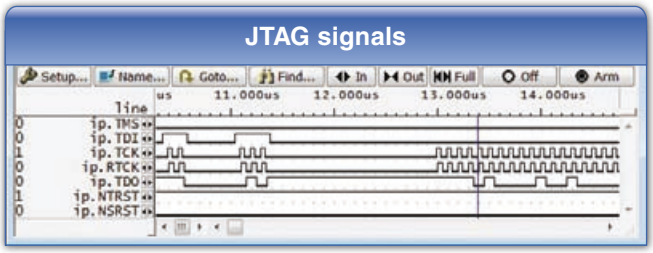


Fig. 6: I segnali JTAG registrati.

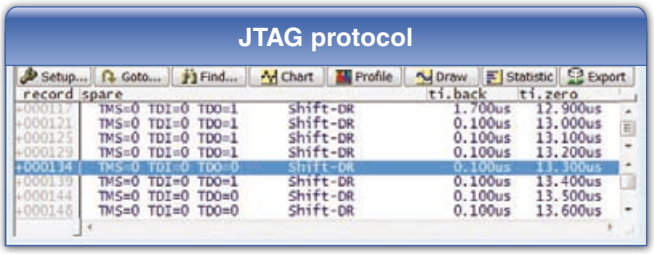


Fig. 7: Rappresentazione dei segnali JTAG sotto forma di protocollo.

Miglioramenti al supporto dei sistemi operativi

Sono state realizzate versioni di supporto per:

- eCos 3.0
- embOS 3.80
- FreeRTOS v7
- Linux v3.0
- MQX 3.6
- RTEMS 4.10
- SMX v4

- I contenuti del tracelogger QNX possono essere visualizzati usando il supporto OS QNX TRACE32. È anche possibile una rappresentazione grafica dei task switch mediante il gruppo di comandi LOGGER.Chart.

- Il supporto OS QNX TRACE32 è stato predisposto per l'utilizzo con eseguibili position-independent.

Nuovi sistemi operativi supportati su Target	
µC/OS-II per Andes	disponibile
µC/OS-II per Blackfin	disponibile
Elektrobit tresos (OSEK/ORTI)	disponibile
Erika (OSEK/ORTI)	disponibile
FreeRTOS per AVR32	disponibile
Linux per Beyond	pianificato
MQX per ARC	disponibile
OSEK/ORTI SMP	pianificato
PikeOS	disponibile
PXROS-HR per TriCore	disponibile
PXROS-HR Run Mode Debugging	disponibile
RTEMS per Nios II	disponibile
Sciopta 2.x	disponibile
SYS/BIOS per ARM	disponibile
VxWorks SMP	disponibile

Code Coverage semplificato

A partire da marzo 2011, in TRACE32 l'informazione di trace può essere inviata in stream dal target in esecuzione ad un hard-disk host. La grande quantità di dati sul flusso di programma che ne deriva, comporta una significativa semplificazione del code-coverage.

Code Coverage basato sul trace

In ambiti industriali come il medicale o l'automotive, è spesso richiesta una verifica sulla copertura delle istruzioni e sulla copertura dei punti decisionali per garantire le specifiche di qualità del sistema.

- La **copertura delle istruzioni** mostra che ogni linea di codice è stata eseguita durante il test del sistema.
- La **copertura dei punti decisionali** mostra che, per ogni istruzione condizionale, sono stati eseguiti almeno una volta sia il ramo di pass sia il ramo di fail**.**

Per molti sistemi embedded occorre testare in real-time del codice altamente ottimizzato. In questi casi non è possibile usare come alternative la strumentazione del codice e l'esecuzione non real-time.

Per essere in grado di raggiungere questi obiettivi, il processore / SoC su target deve soddisfare i seguenti prerequisiti:

1. I core presenti devono avere una logica di trace (vedere Figura 8). Questa logica produce informazioni sulle istruzioni eseguite dal core. In base alle caratteristiche della logica di trace, possono essere prodotte anche informazioni sui task switch e sui cicli di lettura-scrittura.

2. Il processore / SoC deve avere una porta di trace con larghezza di banda sufficiente perché l'informazione di trace possa essere registrata da un sistema esterno senza alcuna perdita di informazione.

Il processo di misura tradizionale

Finora l'analisi di code-coverage in TRACE32 è stata eseguita secondo i seguenti passi:

1. Far partire il programma e fermarlo automaticamente quando la memoria di trace è piena.
2. Trasferire il contenuto della memoria di trace al database di code-coverage.
3. Continuare l'esecuzione del programma.

Per ogni passo del processo di misura, la quantità di dati raccolti era limitata dalla capacità di memoria disponibile nel sistema di trace. I risultati dell'analisi di code-coverage potevano essere controllati dopo che l'intera misura era stata completata oppure, se si voleva, dopo ogni passo intermedio.

Novità: lo Streaming

Se i dati di trace vengono trasferiti su un disco dell'host computer durante la registrazione, tutta l'esecuzione del software può essere registrata in un singolo ciclo di misura. I dati in stream vengono memorizzati su un file dell'hard-disk.
Per evitare di riempire l'hard-disk con i dati di trace, TRACE32 ferma lo streaming non appena rimane solo 1 GByte di memoria libera.

Per far funzionare lo stream, occorre soddisfare i seguenti prerequisiti:

- Host computer a 64-bit ed eseguibile TRACE32 a 64-bit.
- Interfaccia molto veloce fra il sistema di trace e l'host computer.
- Configurazione ottimale della sorgente di trace e del sistema di trace. »

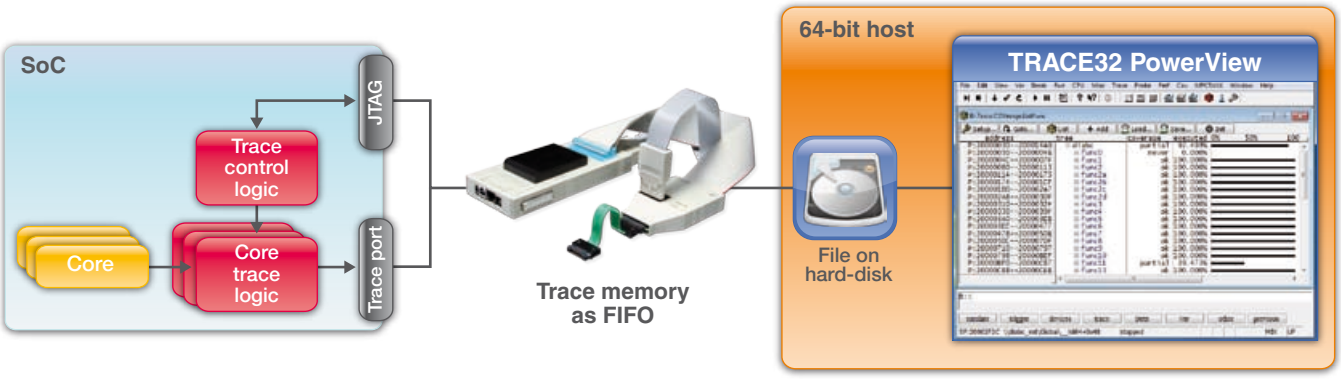


Fig. 8: Nell'analisi di code-coverage si può inviare in stream fino a 1TByte di dati su host computer.

Interfaccia Host veloce

La quantità di dati di trace in uscita dalla trace-port dipende dalle caratteristiche hardware del sistema target. Il numero di core, il numero di pin della trace-port e la velocità del trace clock sono tutti parametri importanti. Anche il protocollo utilizzato dalla logica di trace del core gioca un ruolo importante. Per esempio, il protocollo ARM PTM è più compatto del protocollo ARM ETM v3 (vedere Figura 9).

Il software embedded è un'altra variabile di primaria importanza. Un programma che esegue molti salti e recupera dati o istruzioni principalmente dalla cache, produce più dati di trace al secondo rispetto a un altro programma che esegue molte istruzioni in sequenza e deve attendere spesso la disponibilità di dati o istruzioni.

La quantità di dati è variabile ma è sempre molto elevata. Lo streaming funziona bene solo se la velocità di trasferimento fra il sistema di trace e l'host computer è sufficientemente alta da trasferire tutti i dati dalla trace-port all'host computer senza perdita di informazione. L'interfaccia 1 GBit Ethernet è l'unica raccomandata per il PowerTrace II. La programmazione della logica di trace sul chip può essere usata per influenzare direttamente la quantità di dati di trace generati. La logica deve essere programmata in modo che sia generata solo l'informazione di trace rilevante per l'analisi di code-coverage. Per illustrare questo concetto, vengono forniti i due esempi seguenti.

ETM/PTM: configurazione ottimale

ETM e PTM sono implementazioni diverse della logica di trace del core sulle architetture ARM/Cortex. L'ETM può essere configurata in modo da produrre informazione di trace solo per le istruzioni eseguite dal programma. Per il code-coverage, non sono richieste informazioni sui cicli di lettura e scrittura. Per definizione, PTM genera solo informazioni sul flusso di programma; pertanto non richiede alcuna configurazione.

Entrambe le sorgenti di trace codificano le istruzioni a indirizzi virtuali. Se un progetto embedded utilizza un sistema operativo, come Linux o Windows Embedded, gli

PowerTrace vs. PowerTrace II

I sistemi TRACE32 sono disponibili in due varianti, che differiscono specialmente per le loro prestazioni.

PowerTrace

- Memoria di trace 256 o 512 MByte
- USB 2.x e 100 MBit Ethernet
- Massima velocità di trasferimento verso host-computer pari a 80 MBit/s
- Compressione software dei dati di trace (fattore 3)
- Interfaccia di memoria a 100 MHz

PowerTrace II

- Memoria di trace 1/2/4 Gbyte
- USB 2.x e 1 Gbit Ethernet
- Massima velocità di trasferimento verso host-computer pari a 500 MBit/s
- Compressione hardware dei dati di trace per ETM v3 e PTM (fattore 6)
- Interfaccia di memoria a 233 MHz

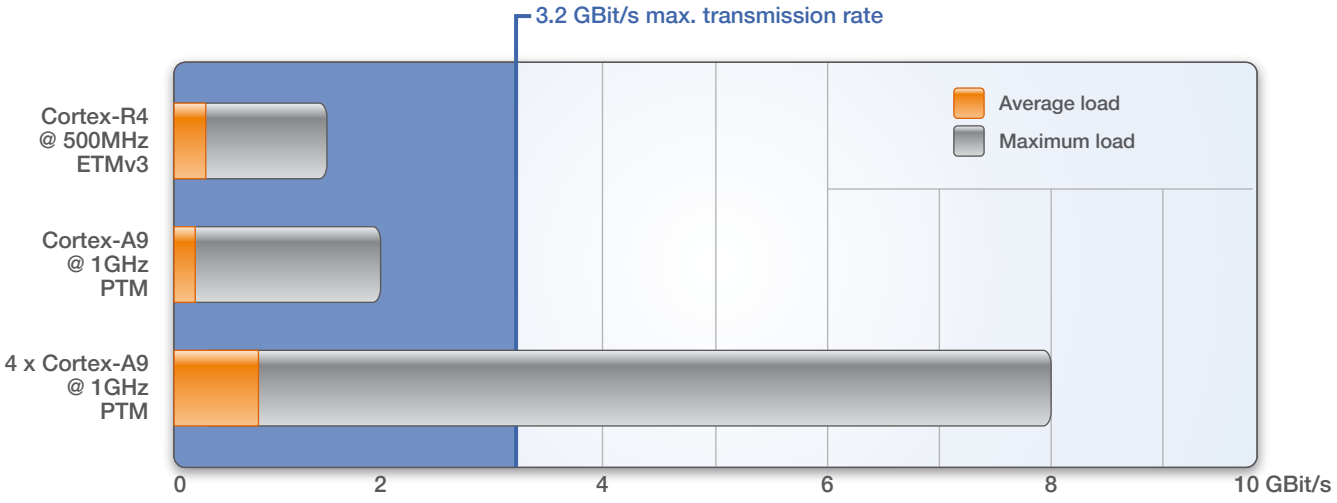


Fig. 9: Una velocità di trasmissione di 3.2 GB/s è generalmente adeguata per lo stream verso host di informazioni sul flusso di programma.

indirizzi virtuali non possono essere mappati sugli indirizzi fisici in modo banale. La sorgente di trace deve essere configurata in modo da produrre informazioni sullo spazio di indirizzamento virtuale in cui si trova un'istruzione.

Per ARM ETM/PTM, la quantità di dati di trace può essere ulteriormente ridotta:

- L'analisi di code-coverage non utilizza e non richiede informazioni temporali. Raccomandiamo dunque di configurare il sistema di trace di TRACE32 in modo che i dati di trace siano trasferiti all'host senza riferimenti temporali (time-stamps). Ciò permette di ridurre di un terzo la quantità di dati.

- PowerTrace II fornisce inoltre una compressione hardware dei dati di trace basata su FPGA. In questo modo si possono trasferire fino a 3.2 GBit/s di dati di trace verso host computer. La Figura 9 mostra come tale velocità di trasferimento sia generalmente sufficiente per lo streaming di dati ETM/PTM senza perdita alcuna.

Nexus: configurazione ottimale

Sui processori delle famiglie MPC5xxx/SPC5xx, la logica di trace del core è implementata con lo standard Nexus. Per poter svolgere un'analisi di code-coverage è sufficiente una cella trace Nexus di classe 2, dal momento che basta avere il flusso di programma sui singoli core. Se si usa il Branch History Messaging, si può avere un trace dati molto compatto. Rispetto al trace dati standard, è realistico stimare una riduzione di un fattore 10. Solo il PowerTrace II supporta lo streaming dalla porta di trace Nexus.

Lo streaming funziona anche su tutti gli altri processori / SoC supportati da TRACE32 e che abbiano una porta di trace.

Code-Coverage per sistemi SMP

TRACE32 supporta anche l'analisi di code-coverage su sistemi SMP (Symmetric MultiProcessing). Per avere la copertura del codice, occorre mostrare che un'istruzione è stata eseguita, mentre è irrilevante sapere quale core l'abbia eseguita. La Figura 10 mostra i risultati del code-coverage per due Cortex-A9 MPCore.

Circa la copertura delle istruzioni e dei punti decisionali, se è stato eseguito soltanto il ramo fail di un'istruzione condizionale, l'istruzione è evidenziata in giallo e marcata con "not exec". Il "detailed coverage" mostra in dettaglio quante volte ogni istruzione o ogni salto nell'istruzione è stato eseguito.

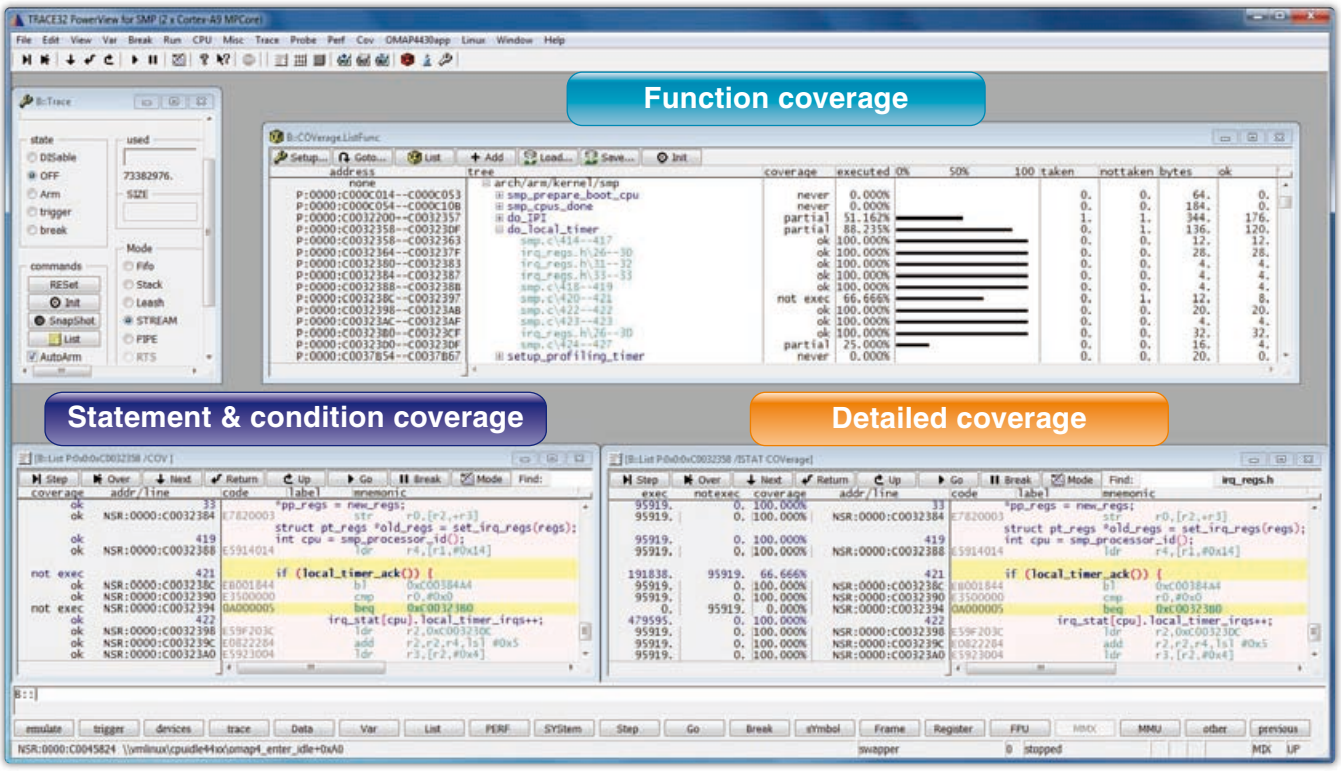


Fig. 10: Analisi di code-coverage per un sistema SMP.

CoreSight Trace Memory Controller

Il nuovo CoreSight Trace Memory Controller fornisce ai progettisti SoC più opzioni di progettazione per l'infrastruttura di trace. TRACE32 supporta già il primo componente che utilizza il TMC.

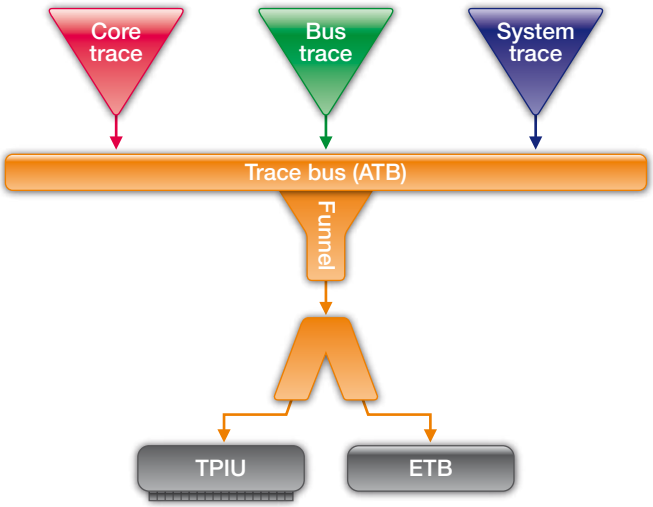


Fig. 11: Il CoreSight Funnel combina in un singolo flusso dati tutti i dati di trace prodotti dalle diverse trace macrocells.

Attraverso CoreSight, i dati necessari per l'analisi dei processi interni a un SoC vengono prodotti da "trace macrocells". Ci sono tre tipi di trace macrocells:

- **Core trace macrocells** sono assegnate ad un core e generano informazioni di trace relative alle istruzioni eseguite dal core stesso. Le informazioni sullo switch dei processi e sulle letture/scritture di dati sono generate in base al design della trace macrocell.
- **Bus trace macrocells** sono assegnate ad un bus e generano informazioni di trace sui trasferimenti dati che avvengono su questo bus.
- **System trace macrocells** generano informazioni di trace per trigger hardware (system event tracing) o forniscono informazioni diagnostiche generate dalla instrumentazione del programma applicativo.

Il CoreSight Funnel combina tutti i dati di trace in un singolo flusso dati (vedere Figura 11). Questo flusso dati può essere quindi registrato in una memoria buffer on-chip (ETB) oppure trasmesso a uno strumento esterno attraverso una trace port (TPIU).
La IP per il trace CoreSight attualmente implementato viene a volte spinta al limite con SoC multicore complessi che contengono molte trace macrocells.

ARM CoreSight

Con CoreSight, ARM mette a disposizione una vasta serie di blocchi che consentono ai progettisti SoC di costruire un'infrastruttura debug e trace personalizzata.

Una singola interfaccia di debug è sufficiente per controllare e coordinare tutti i core del SoC, così come l'accesso a tutta la memoria.

Un'interfaccia trace è sufficiente per fornire dati diagnostici sui processi che eseguono all'interno del SoC senza alcun impatto sulle prestazioni real-time.

- **ETB:** la memoria dell'on-chip trace è spesso troppo piccola per registrare dati sufficienti per un'analisi significativa. La dimensione tipica dell'ETB varia tra i 4 e i 16 Kbyte.
- **TPIU:** in alcuni stati il sistema può generare più dati di trace di quanti la porta trace ne possa trasmettere. CoreSight è progettato per prelevare i dati dalle trace macrocells solo se possono essere trasmessi dalla TPIU. Se i dati di trace generati rimangono nelle trace macrocells per troppo tempo, le loro FIFO possono riempirsi e si possono perdere dati importanti.

Il nuovo CoreSight Trace Memory Controller dovrebbe fornire una soluzione per entrambi gli scenari sopra descritti.

TMC come Embedded Trace Buffer

Per poter memorizzare più dati di trace on-chip per una successiva analisi, i produttori di chip possono teoricamente collegare fino a 4 Gbyte di SRAM al Trace Memory Controller (vedere Figura 12).

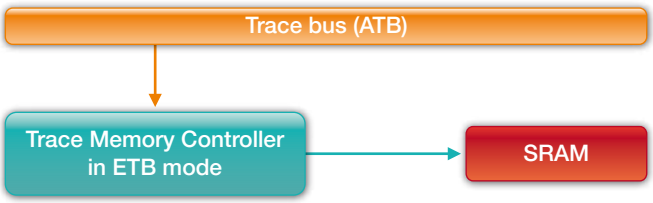


Fig. 12: In modo ETB, il Trace Memory Controller può rendere disponibili fino a 4Gbyte di memoria trace on-chip.

TMC come Embedded Trace FIFO

Le analisi del flusso dati trasmesso dalla TPIU mostrano che la banda passante della maggior parte delle trace port è sufficientemente ampia per le normali operazioni. Il sovraccarico, e la conseguente perdita di dati di trace, capita solo quando si verificano dei picchi.

Il Trace Memory Controller può essere integrato nell'infrastruttura trace del SoC, in modo da funzionare come una Embedded Trace FIFO per assorbire i picchi di carico della TPIU (vedere Figura 13). Questa ETF è progettata in modo che i dati di trace non possano essere perduti. La dimensione della ETF può essere liberamente definita tra 512 Byte e 4 Gbyte.

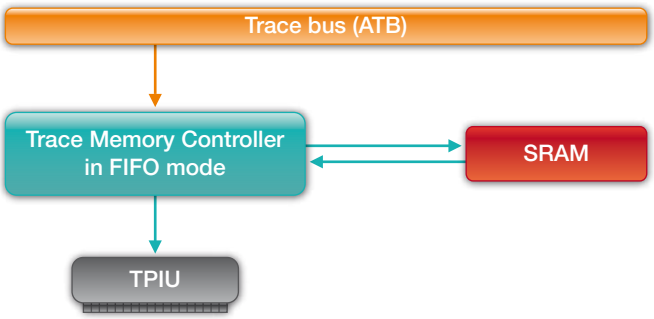


Fig. 13: In modo FIFO, il Trace Memory Controller assorbe i picchi di carico sulla TPIU. In questo modo si può evitare la perdita di dati di trace.

Entrambe le integrazioni del Trace Memory Controller nella infrastruttura trace sopra descritta sono semplici esempi. Naturalmente è possibile configurare il blocco TMC nel sistema CoreSight in modi più complessi e flessibili.

Modifiche in TRACE32

Come ci si può aspettare, Lauterbach ha dovuto modificare il software TRACE32 per la configurazione e la gestione del Trace Memory Controller. Questo vale in particolare quando il Trace Memory Controller è integrato nel SoC usando nuovi modi prima non supportati. L'utente TRACE32 deve solo configurare l'indirizzo base del TMC. Poi tutte le funzionalità di visualizzazione del trace e di analisi possono essere utilizzate come di consueto.

TMC come Router per High-Speed Link

L'idea di allontanarsi dalle trace port dedicate è stata a lungo discussa nella comunità embedded. Ci sono certamente diverse buone ragioni per questa mossa.

Per la prima volta i trace CoreSight possono essere collegati ad un'interfaccia standard ad alta velocità usando

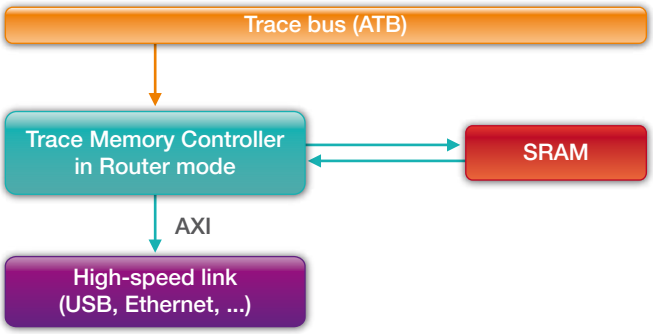


Fig. 14: In modo Router, il Trace Memory Controller inoltra i dati di trace per la trasmissione ad una interfaccia standard ad alta velocità.

il Trace Memory Controller. Le interfacce USB o Ethernet sono fra le più comuni, specialmente perché sono disponibili in molti prodotti finali. Idealmente, lo strumento di trace esterno condivide l'interfaccia con altri dispositivi connessi. All'interno del SoC, il TMC funziona come Embedded Trace Router e ha il compito di trasmettere i dati di trace attraverso il bus AXI per l'esportazione all'IP dell'interfaccia ad alta velocità (vedere Figura 14).

Questo nuovo metodo di esportazione del trace richiederà strumenti di trace completamente nuovi. Lauterbach è in stretto contatto con i principali produttori di semiconduttori per sviluppare gli strumenti appropriati per questo cambio di tecnologia.

Funzionalità CoreSight di TRACE32

- Aperto all'utilizzo con tutti i core che possono essere integrati in CoreSight; Lauterbach offre soluzioni di debug per tutti i core ARM/Cortex e per numerosi DSP e core configurabili.
- Supporto multi processing asimmetrico (AMP) e simmetrico (SMP)
- Debug via JTAG e Serial Wire Debug a 2-pin
- Debug sincronizzato di tutti i core
- Supporto per la Cross Trigger Matrix di CoreSight
- Supporto per tutti i tipi di macrocelle trace (ETM, PTM, HTM, ITM, STM e altre)
- Strumenti per trace port parallele e seriali
- Trace multicore
- Profiling completo per sistemi multi-core

Analisi trace intelligente per Cortex-M3/M4

Risoluzione dei problemi, ottimizzazione delle prestazioni e analisi di copertura del codice – tutte queste attività possono essere eseguite con rapidità e precisione in un sistema embedded se è disponibile un'adeguata analisi del trace. Nel 2011 Lauterbach ha esplorato nuove vie per permettere l'analisi del trace per i processori Cortex-M3/M4.

Uso congiunto di ETM e ITM

Nei processori Cortex-M3/M4, si può generare informazione di trace da due diverse sorgenti (vedere Figura 17). L'ETMv3 genera informazioni sulle istruzioni eseguite. L'ITM genera informazioni sugli accessi read/write eseguiti, mediante l'unità Data Watchpoint and Trace (DWT).

Le strutture di trace ITM per gli accessi read/write contengono le seguenti informazioni: indirizzi dei dati, valori e program counter.

Attraverso l'analisi del program counter, gli accessi dati generati in modo separato possono essere rapportati alla sequenza di programma (vedere Figura 15), che a sua volta permette di localizzare gli errori in modo molto più semplice. La causa di un errore, ad esempio la scrittura di un dato sbagliato a un certo indirizzo, si può trovare facilmente se gli accessi in scrittura sono mappati sull'intero trace di programma.

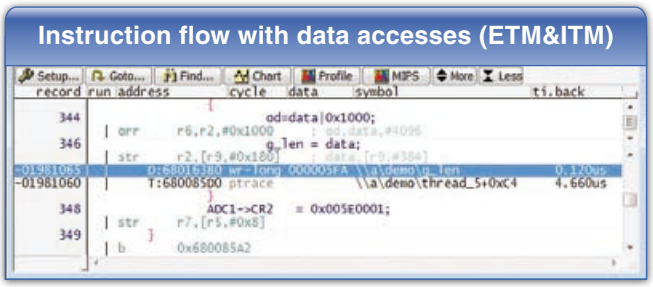


Fig. 15: Unendo le analisi di data-trace ETM e ITM, gli accessi read/write possono essere rapportati alla sequenza di programma.

OS-Aware Tracing

Se un sistema operativo sta girando su Cortex-M3/M4, l'informazione sui task-switch diventa essenziale per l'analisi di trace.

Per ricevere informazioni sui task-switch si può usare il seguente metodo: l'informazione di trace sul ciclo di scrittura con cui il kernel scrive l'identificatore del task corren-

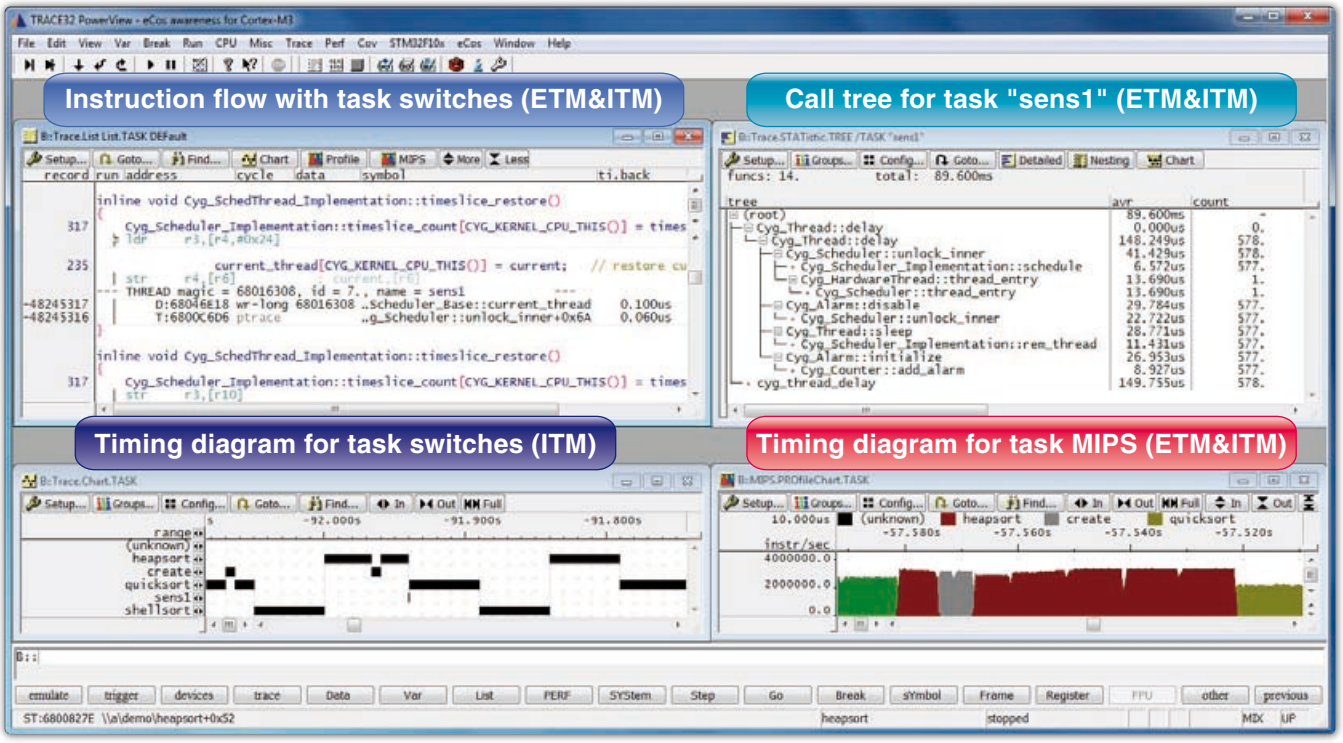


Fig. 16: Attraverso l'unione dei data-trace ETM e ITM, si può fornire un'ampia analisi di trace per il sistema operativo eCos.

te nella corrispondente variabile di OS, può essere generata usando l'ITM. Come descritto sopra, l'informazione di accesso in scrittura si può rapportare al trace del flusso di programma. In questo modo migliora la leggibilità del listato di trace (vedere Figura 16). L'integrazione del task-switch nella sequenza di programma è anche alla base delle analisi di runtime mostrate in Figura 16.

Tre modalità di registrazione

Per registrare l'informazione di trace prodotta dai processori Cortex-M3/M4, Lauterbach supporta tre modalità:

- **modalità FIFO:** l'informazione è memorizzata nella memoria di 128MByte del CombiProbe TRACE32.
- **modalità STREAM:** l'informazione è inviata in streaming su un hard-disk dell'host computer.
- **real-time Profiling:** l'informazione di trace è inviata in streaming sull'host computer e analizzata in tempo reale.

Nelle prime due modalità di registrazione, si raccoglie prima l'informazione di trace; l'analisi si svolge dopo aver concluso la registrazione.

Ogni modalità di registrazione ha delle caratteristiche specifiche. Il metodo più comunemente usato è quello FIFO. È molto veloce e tipicamente basta per localizzare gli errori e per svolgere analisi runtime.

L'ETMv3 implementata sui processori Cortex-M3/M4 non ha né trigger né filtri sul trace. Non è possibile selezionare per la registrazione solo quei segmenti di programma che sono richiesti per risolvere un problema. Questo può comportare che i dati di trace debbano essere raccolti per un periodo di tempo relativamente lungo, al fine di coprire l'area richiesta per l'analisi. In questo caso, la modalità STREAM può essere l'opzione migliore. Tuttavia la modalità STREAM è molto esigente nei confronti dell'ambiente di debug.

- La grande quantità di dati ottenuti con lo streaming richiede un eseguibile TRACE32 a 64-bit, necessario per gestire l'indirizzamento del gran numero di dati di trace che saranno raccolti.

- La velocità di trasferimento fra il CombiProbe e l'host computer deve essere sufficientemente elevata per consentire lo stream di tutti i dati di trace senza perdita di informazione. I 128 MByte di memoria del CombiProbe sono usati per tamponare i picchi di carico dalla porta trace (TPIU).

Il real-time profiling è particolarmente adatto per la copertura delle istruzioni e dei punti decisionali. L'analisi di copertura può essere seguita in tempo reale sul monitor e i risultati dei test sono visibili immediatamente (vedere Figura 17). Le linee marcate con "ok" sono quelle coperte.

Fig. 17: Il real-time profiling permette di seguire in tempo reale sul monitor l'analisi di code-coverage.

Simulazione e realtà ora più vicine

È ormai abitudine comune eseguire simulazioni e verifiche del progetto prima di commissionare l'hardware. Questo è il motivo per cui prodotti come MATLAB® e Simulink® hanno fatto irruzione nel mercato dell'ingegneria dei controlli, come software di sviluppo. Si può risparmiare molto tempo e molto lavoro se il ciclo di controllo può essere testato rispetto agli effetti di molte variabili prima di chiudere la fase di design.

E dunque, qual è il passo successivo, dopo aver definito l'algoritmo di controllo attraverso la simulazione? Come si integra questa soluzione nell'hardware di controllo? Per rispondere a queste domande, Simulink permette la generazione automatica del codice. Ma si può essere certi che il programma si comporterà sull'hardware di controllo proprio come nella simulazione?

Un approccio alla verifica

L'Istituto "Dinamica dei Sistemi di Volo" alla Technische Universität di Monaco ha prodotto un'interessante soluzione durante lo sviluppo di un sistema di controllo di volo per il Diamond DA42 (vedere Figura 20).

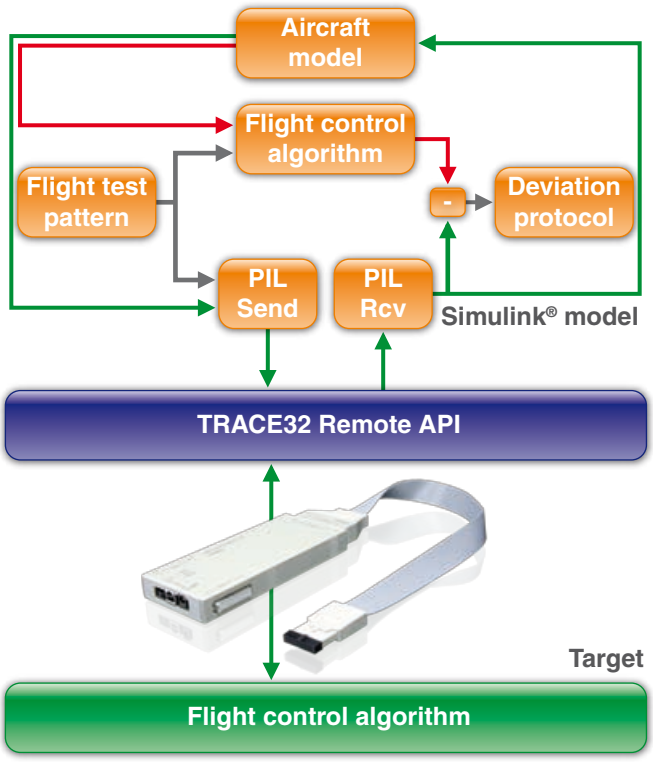


Fig. 18: Confronto fra il comportamento nel controllo reale (righe verdi) e nel controllo simulato (righe rosse).

Dopo aver creato gli algoritmi di controllo e averli testati funzionalmente con Simulink, il corrispondente codice di programma per il processore dell'hardware di controllo è stato generato a partire dai blocchi di controllo, usando il modulo Embedded Coder.
Mediante un debugger TRACE32, il codice generato è stato caricato nell'hardware di controllo e testato funzionalmente in-situ.

Per determinare lo scostamento fra controllo simulato (righe rosse) e reale (righe verdi), ma soprattutto per confermare l'accuratezza numerica del controllo hardware, è stata scelta una simulazione Processor-In-the-Loop (PIL) (vedere Figura 18). Essenzialmente, la simulazione PIL si basa sui blocchi specializzati di Simulink "PIL Send" e "PIL Receive". Questi blocchi sono stati progettati per implementare una comunicazione fra Simulink e le API remote di TRACE32.

In ogni esecuzione, l'algoritmo di controllo del volo esegue un singolo passo di calcolo del controllo di volo a tempo discreto sull'hardware target. Il modello Simulink fornisce i necessari parametri d'ingresso. I valori calcolati vengono restituiti al modello Simulink e qui sono forniti al modello del velivolo. In un sistema di calcolo parallelo, l'algoritmo di controllo del volo simulato calcola gli stessi valori. La differenza è poi usata per confrontare i due risultati.

Il test locale ha prodotto come risultato una deviazione assoluta pari a 10^{-13} – un livello di consistenza molto accurato, dimostrato elegantemente e facilmente con questo approccio.

Per maggiori informazioni sul progetto dell'Istituto "Dinamica dei Sistemi di Volo" alla Technische Universität di Monaco, visitare il sito
www.lauterbach.com/intsimulink.html.

Integrazione di TRACE32 con Simulink®

All' Embedded World show di febbraio 2012 a Norimberga in Germania, Lauterbach ha presentato un'integrazione ancora più stretta fra Simulink e i debuggers TRACE32 Lauterbach.

Lauterbach ha utilizzato la proprietà del generatore di codice di Simulink di far iniziare un blocco di codice sempre con una linea di commento che contiene il nome e il path del modello per quel blocco. Le linee di commento restano disponibili dopo che il codice generato è stato caricato nel debugger TRACE32. Queste linee permettono una semplice correlazione fra i blocchi di Simulink e le linee di codice sorgente.

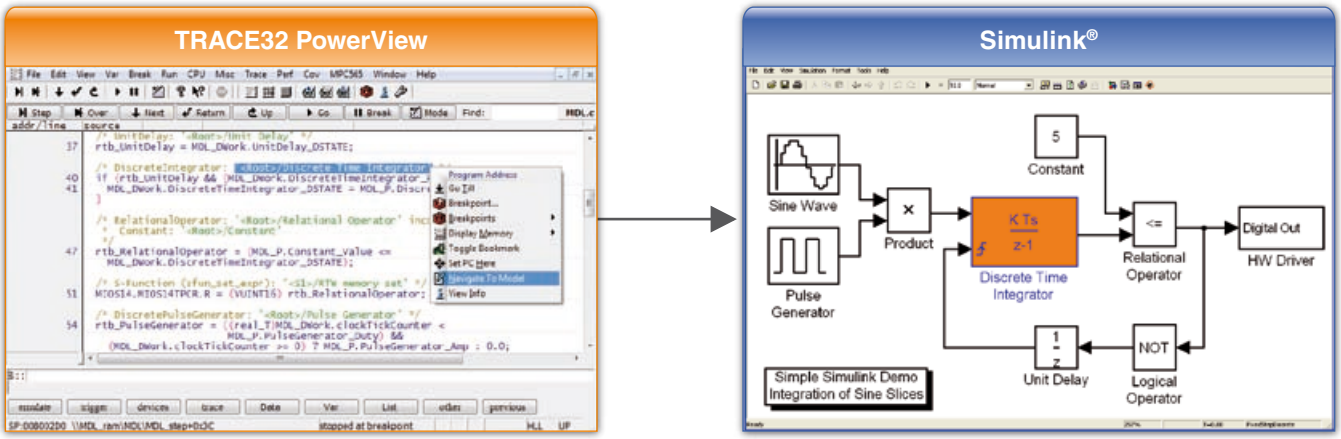


Fig. 19: Il blocco che appartiene al codice sorgente selezionato è evidenziato in Simulink.

Navigare da Simulink® a TRACE32

Nei 'Simulink Customization Menus' di Simulink sono integrati un menu generale TRACE32 e dei menu TRACE32 per i blocchi e i segnali. Il debugger TRACE32 può essere controllato da Simulink con l'aiuto di questi menu. Sono disponibili le seguenti funzioni:

- mostrare blocchi di codice in TRACE32
- aprire la finestra Variable Watch di TRACE32 per i segnali
- caricare un build Simulink nel debugger TRACE32
- impostare e gestire breakpoint programma sui blocchi in Simulink
- impostare e gestire breakpoint su segnali o dati
- avviare e fermare il programma sull'hardware di controllo

Navigare da TRACE32 a Simulink®

La selezione di una parte di codice sorgente nel debugger TRACE32 evidenzia il blocco corrispondente in Simulink (vedere Figura 19).

Sviluppi futuri

Quando sarà disponibile la release 2012a di Simulink, saranno utilizzabili in Simulink ulteriori funzioni di TRACE32. Lauterbach utilizzerà le nuove funzionalità dell'API rtiostream di Simulink per integrare una simulazione PIL, il data logging e il tuning dei parametri.

MATLAB® e Simulink® sono marchi registrati di The MathWorks, Inc.

Fig. 20: Diamond DA42 (fonte: www.diamond-air.at)

Debug del BIOS UEFI con TRACE32

Una nuova estensione di TRACE32 per il Debugger Atom™ permette una piena capacità di debug del BIOS UEFI H2O di Insyde.

UEFI è il successore del tradizionale BIOS per PC. Funziona come interfaccia fra il firmware e il sistema operativo, gestendo il processo di boot. Dall'accensione fino al passaggio di controllo al sistema operativo, UEFI esegue diverse fasi ben distinte (vedere Figura 21).

Essendo un sistema basato su JTAG, TRACE32 permette il debug a partire dal vettore di reset.
In ogni fase del processo di boot, l'interfaccia utente PowerView rende disponibili finestre speciali che mostrano informazioni specifiche per UEFI. Le funzioni e gli script predisposti permettono il debug dei driver caricati dinamicamente fin dalla prima istruzione. Per maggiori informazioni sulla nuova estensione UEFI, visitare **www.lauterbach.com/uefi.html.**

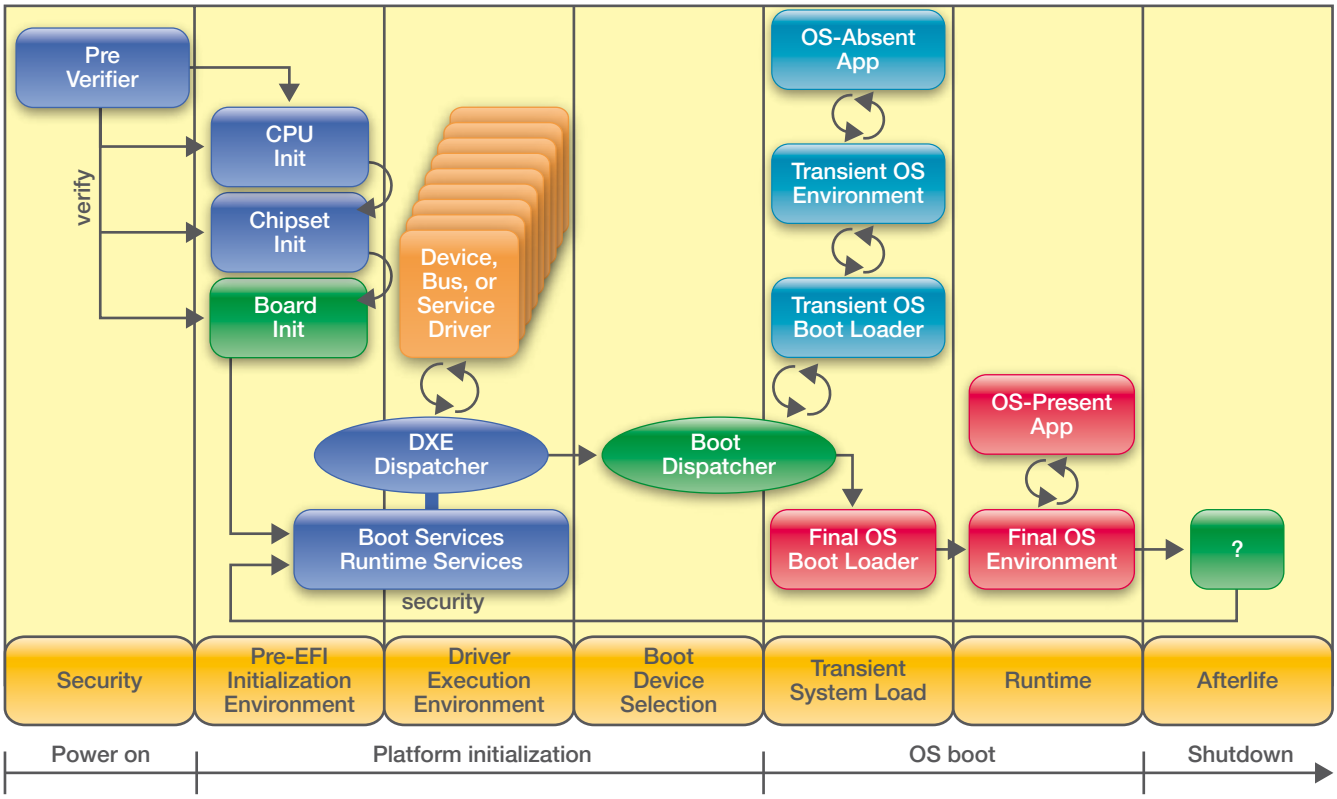


Fig. 21: Il processo di boot del sistema con UEFI.

I.P.

LE NOSTRE SEDI IN TUTTO il MONDO

- **Italia**
- USA
- Germania
- Francia
- Regno Unito
- Cina
- Giappone

Siamo rappresentati da partners esperti in tutte le altre nazioni.

TENETECI INFORMATI

Se il vostro indirizzo è cambiato o se non volete più rimanere nella nostra mailing list, mandateci un e-mail a:

info_it@lauterbach.it