Neue Konzepte für das Multicore Debugging

Seit 2002 unterstützt Lauterbach mit seinen In-Circuit Debuggern TRACE32-ICD das Debugging von Multicore SoCs. Aufgrund der Erfahrungen in zahlreichen Kundenprojekten wurde das Lauterbach Multicore Debugging Konzept nun weiterentwickelt.

Um eine optimale Funktionalität und Performance zu erreichen, werden immer häufiger mehrere Cores zu einem System-on-Chip (SoC) integriert. Weit verbreitet ist das Zusammengehen eines RISC Prozessors mit einem oder mehreren DSPs, aber auch andere Kombinationen kommen zum Einsatz. Bei Multicore SoC Designs wird in der Regel, um Pins und Kosten zu sparen, nur eine Debug-Schnittstelle für alle Cores zur Verfügung gestellt.

Ansteuerung mehrerer Cores über eine gemeinsame Debug-Schnittstelle

Mit dem neuen Konzept genügt für das Debugging aller Cores eines SoC ein Power Debug Module und ein Debug Kabel. Dabei muss das Debug Kabel Lizenzen für alle eingesetzten Cores bzw. eine Multicore-Lizenz enthalten.

Für das Debugging jedes Cores läuft auf dem Host eine separate TRACE32 Applikation. Die gemeinsame TRACE32-Systemsoftware im Power Debug Module sorgt nun dafür, dass die von einer Applikation abgesetzten Debug-Kommandos an den zugehörigen Core weitergeleitet werden.

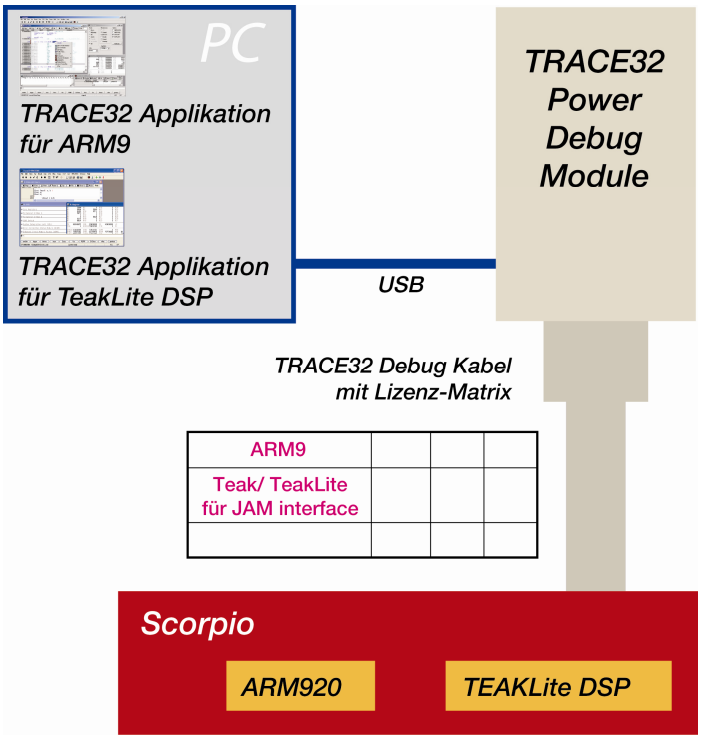

Bild 1 Debugging des ARM920 und des TeakLite DSP im Scorpio über eine gemeinsame TRACE32-ICD Hardware

Zur Veranschaulichung dieses Konzepts zwei Beispiele:

Scorpio: Der Scorpio von Samsung ist ein SoC, der einen ARM920 und einen TeakLite DSP enthält. Für das gleichzeitige Debugging beider Cores muss das Debug Kabel eine Lizenz für den ARM9 und eine Lizenz für den TeakLite enthalten (siehe Bild 1).

MSC8102: Der MSC8102 ist ein SoC von Motorola, der vier StarCore DSPs enthält. Für das gleichzeitige Debugging aller vier DSPs genügt es, dass im Debug Kabel eine Lizenz für den StarCore sowie eine Multicore-Lizenz eingetragen ist.

Start-/Stopp-Synchronisation

Ein wichtiges Thema beim gleichzeitigen Debugging mehrerer Cores ist natürlich die Start- und Stopp-Synchronisation. Beides funktioniert nur dann, wenn sowohl das synchrone Starten und als auch das synchrone Stoppen vom SoC unterstützt wird. Sind die einzelnen Cores beispielsweise über einen Breakswitch miteinander gekoppelt, lässt sich ein nahezu clocksynchrones Anhalten aller Cores realisieren. Ein programmierbarer Breakswitch erlaubt dem Anwender zudem selbst zu konfigurieren, welche Cores des SoC sich gegenseitig synchron stoppen sollen.

Unterstützt der SoC selbst kein synchrones Anhalten und gibt es auch im Zielsystem physikalisch keine Möglichkeit, dass sich die Cores gegenseitig stoppen, kann auch der Debugger nur eine zeitnahe Stopp-Synchronisation realisieren. Gleiches gilt auch für die Start-Synchronisation: Lassen sich die einzelnen Cores des SoC über ihre Debugging-Logik gemeinsam starten (beispielsweise über ein spezielles JTAG Kommando), kann der Debugger eine Start-Synchronisation realisieren. Andernfalls kann er nur versuchen, die einzelnen Cores schnellstmöglich nacheinander zu starten.

Gemeinsamer Traceport

Viele Entwickler wollen auch bei einem Multicore SoC Design nicht auf die Vorteile eines Echtzeit-Traces verzichten. Natürlich wird auch hier aus Kostengründen an den Pins für den Traceport gespart. Inzwischen gibt es die ersten SoC Designs, bei denen sich mehrere Cores einen Traceport teilen. In den aktuellen Designs kann der Entwickler einstellen, welcher Core den Traceport exklusiv nutzen darf. Im Gespräch sind aber auch SoCs bei denen mehrere Cores gleichzeitig einen Traceport bedienen (z.B. im CoreSight Konzept von ARM oder bei NEXUS).

Andrea Martin, Februar 2005

Altlaufstraße 40 • D-85635 Höhenkirchen-Siegertsbrunn • Tel. +49 8102 9876-0 • Fax +49 8102 9876-999 • sales@lauterbach.com