Debugging von AMP- und SMP-Systemen

Viele Multicore-Prozessoren können als AMP- oder SMP-Systeme betrieben werden. Angepasst an die Betriebsart kommen unterschiedliche Debug- und Trace-Konzepte zur Anwendung. Am Beispiel des ARM Cortex-A9 MPCore werden diese TRACE32-Konzepte vorgestellt.

Debug-Konzepte

Im Gespräch mit Kunden stellen wir immer wieder fest, dass es für die beiden Begriffe

- **AMP – Asymmetrisches Multiprocessing**
- **SMP – Symmetrisches Multiprocessing**

recht unterschiedliche Interpretationen gibt. Deshalb beschreiben wir im Folgenden, wie die Firma Lauterbach diese Begriffe verwendet und welche Auswirkungen unsere Sichtweise auf die Konfiguration und die Bedienung des TRACE32-Debuggers hat.

Wie der Begriff „Multiprocessing“ schon vermuten lässt, arbeiten mehrere Cores in einem Embedded System zusammen. Entscheidend für das Debugging ist die Frage, wie die Systemaufgaben auf die einzelnen Cores verteilt werden.

Debug-Konzept für AMP-Systeme

In AMP-Systemen werden jedem Core bestimmte Aufgaben fest zugeteilt. Die Entscheidung über die Zuteilung wird in der Designphase des Systems getroffen. Dabei werden neben Cores für allgemeine Aufgaben oft zusätzlich Spezialcores ausgewählt, die für bestimmte Funktionen optimiert sind, z.B. DSPs.

Beim Debugging von AMP-Systemen wird für jeden Core eine eigene TRACE32-Instanz gestartet (siehe Bild 4). Dies hat zwei Gründe:

1. Ein AMP-System kann unterschiedliche Core-Architekturen enthalten.
2. Jeder Core arbeitet einen separaten Teil der Applikation ab. Das heißt, die Symbol- und Debug-Informationen sind größtenteils exklusiv dem jeweiligen Core zugeordnet.

Da die Cores jedoch nicht unabhängig voneinander arbeiten, sondern parallel und gemeinsam alle Applikationsaufgaben lösen, muss es möglich sein, alle Cores synchron zu starten und zu stoppen. Nur so lässt sich das Zusammenspiel der Cores und damit die Gesamtapplikation umfassend testen.

Das synchrone Starten und Stoppen aller Cores kann unterschiedlich realisiert sein. Idealerweise unterstützt der Multicore-Prozessor dies durch eine interne Synchronisierungslogik. Fehlt diese, übernimmt TRACE32 die Synchronisation. Ein spezieller Algorithmus berechnet dabei JTAG-Sequenzen, um alle Cores möglichst zeitnah zu steuern. »

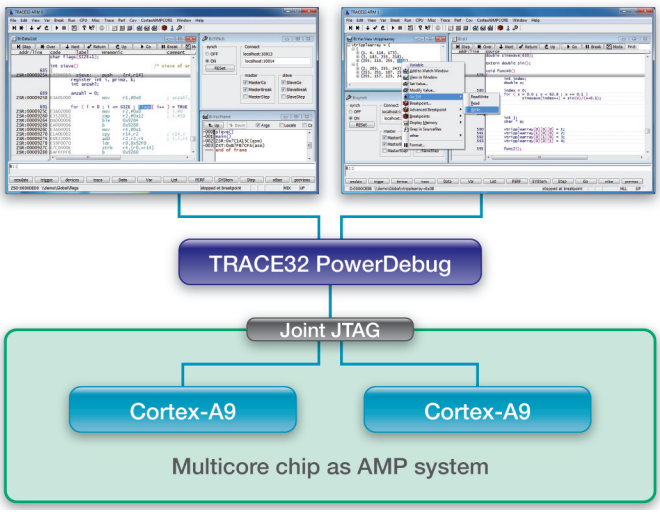

Bild 4: Beim Debugging von AMP-Systemen wird für jeden Core eine eigene TRACE32-Instanz gestartet.

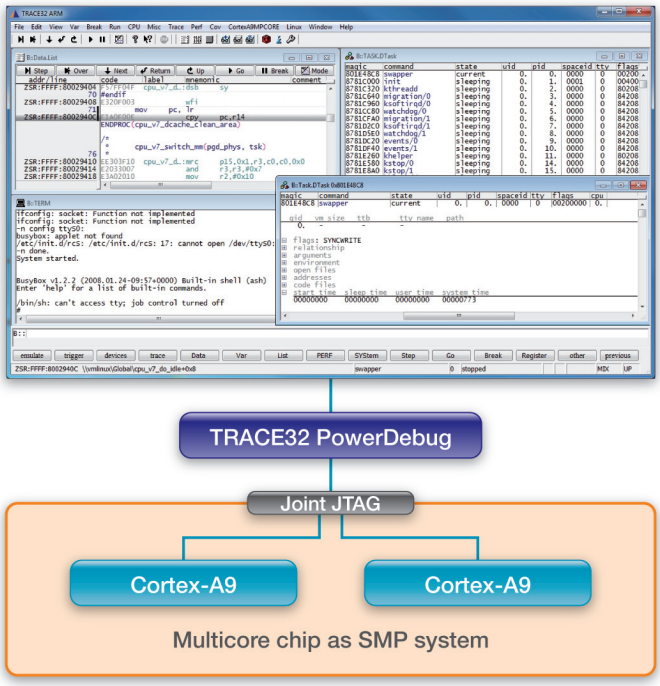

Bild 5: Beim Debugging von SMP-Systemen wird für alle Cores nur eine einzige TRACE32-Instanz gestartet.

Debug-Konzept für SMP-Systeme

Im Gegensatz zu AMP-Systemen, bei denen genau definiert ist, welche Aufgaben von welchem Core bearbeitet werden, wird diese Zuordnung bei SMP-Systemen nicht mehr vom Systemdesigner, sondern von einem SMP-Betriebssystem übernommen. Damit die Aufgaben frei jedem Core zugeteilt werden können, müssen alle Cores vom gleichen Typ sein.

Die Aufgabenzuteilung erfolgt dynamisch, das heißt abhängig vom aktuellen Systemzustand. Eine Aufgabeneinheit, die ein Betriebssystem zuteilen kann, wird als Task oder Thread bezeichnet. Ein zur Bearbeitung anstehender Task wird, vereinfacht gesagt, dem Core zugeteilt, der gerade frei ist.

Für das Debugging von SMP-Systemen wird nur **eine** TRACE32-Instanz geöffnet, von der aus alle Cores kontrolliert werden (siehe Bild 5 auf der vorigen Seite). Da für den Entwickler das Debugging eines einzelnen Tasks im Vordergrund steht, visualisiert die TRACE32-Bedienoberfläche den Zustand des Gesamtsystems immer aus Sicht eines einzelnen Tasks bzw. aus Sicht des Cores, auf dem dieser Task gerade läuft. Selbstverständlich lässt sich diese Visualisierung auch auf andere Tasks bzw. Cores umschalten.

TRACE32 übernimmt eine ähnliche Funktion wie das SMP-Betriebssystem. Es organisiert das Debuggen aller Cores, ohne dass sich der Entwickler um die Details des SMP-Systems kümmern muss. Wird beispielsweise ein Breakpoint gesetzt, so sorgt TRACE32 dafür, dass dieser in allen Cores eingetragen wird. Dies ist notwendig, da zum Zeitpunkt des Breakpoint-Setzens noch nicht klar ist, welcher Core den Programmteil mit dem Breakpoint ausführen wird. Hält ein Core dann am Breakpoint, werden automatisch auch alle anderen Cores gestoppt. Die Anzeige in TRACE32 wechselt auf den Task bzw. den Core, bei dem der Breakpoint zugeschlagen hat. Wird das Programm wieder gestartet, laufen alle Cores gemeinsam los.

Das Debuggen von SMP-Systemen mit TRACE32 ist einfach. Nachdem eine TRACE32-Instanz gestartet und für das SMP-System konfiguriert wurde, arbeitet man damit im Wesentlichen genauso, als würde man nur einen Core debuggen.

Trace-Konzepte

Auch die Darstellung und Auswertung von Traceinformation wird von TRACE32 unterschiedlich gehandhabt, abhängig davon, ob die Tracedaten von einem AMP- oder einem SMP-System erzeugt werden. Für AMP-Systeme wird die Traceauswertung für die einzelnen Cores weitgehend eigenständig durchgeführt. Die Traceinformationen für ein SMP-System hingegen lassen sich, abhängig von der Fragestellung, für einen einzelnen Task, für einen einzelnen Core bzw. für das Gesamtsystem auswerten.

Trace-Konzept für AMP-Systeme

Da das Debugging der einzelnen Cores eines AMP-Systems über separate TRACE32-Instanzen durchgeführt wird, erfolgt auch die Darstellung der Traceinformationen in den einzelnen Bedienoberflächen. AMP-Systeme können aus verschiedenartigen Cores bestehen, deshalb kommen unter Umständen unterschiedliche Traceprotokolle zum Einsatz. Die einzelnen Protokolle lassen sich durch ihre Aufteilung in die einzelnen Bedienoberflächen separat dekodieren und für die Auswertung aufbereiten.

Um die Zusammenarbeit der Cores zu testen und komplexe Systemfehler schnell zu finden, ist es möglich, die einzelnen Tracedarstellungen in ihrem zeitlichen Bezug zueinander darzustellen. TRACE32-PowerTrace stellt dafür eine gemeinsame Zeitbasis zur Verfügung. Diese erlaubt es, in einer Bedienoberfläche einen Zeitpunkt in der Tracedarstellung auszuwählen und in allen anderen Bedienoberflächen genau den Befehl zu sehen, der dort zum etwa gleichen Zeitpunkt ausgeführt wurde (siehe Bild 6). »

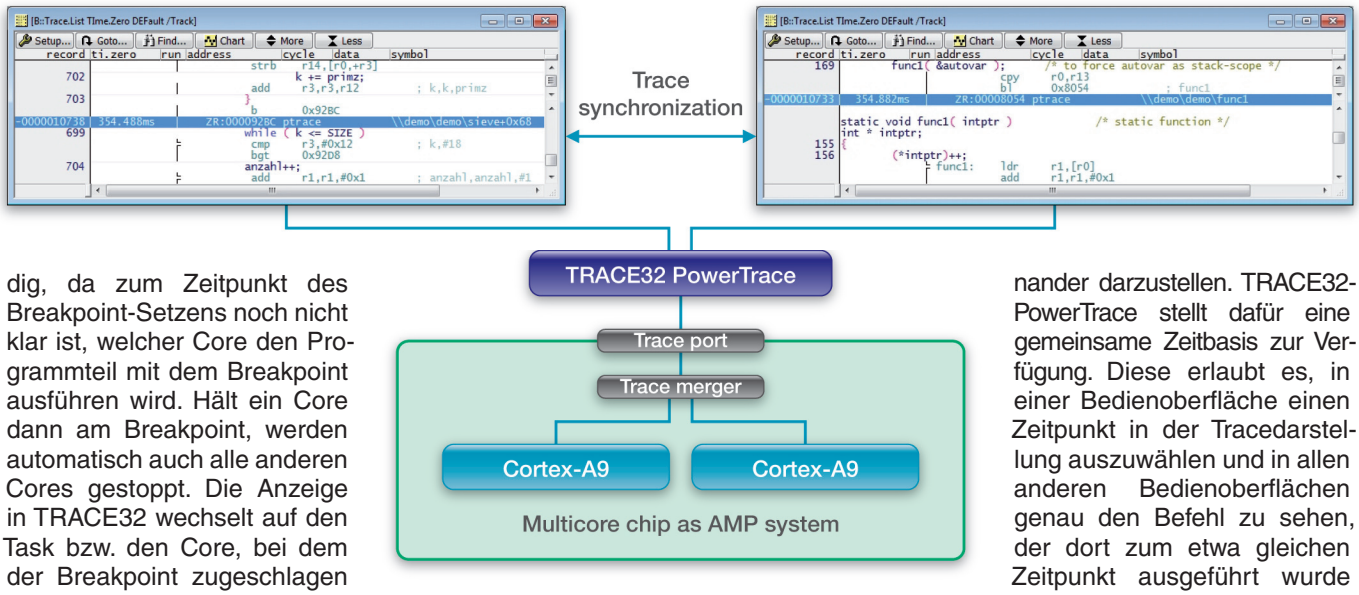

Bild 6: Beim Tracen von AMP-Systemen wird die Traceinformation für jeden Core in seiner Bedienoberfläche dargestellt. Eine zeitliche Synchronisierung zwischen den Bedienoberflächen ist möglich.

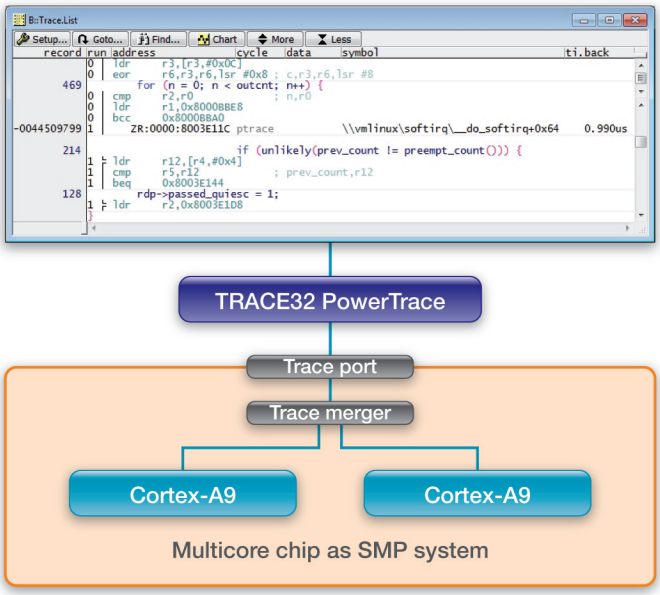

Bild 7: Beim Tracen von SMP-Systemen werden die Informationen für alle Cores in einem gemeinsamen Tracespeicher abgelegt.

Trace-Konzept für SMP-Systeme

Obwohl die Informationen zum Programmablauf in einem SMP-System für alle Cores in einem gemeinsamen Tracespeicher abgelegt werden (siehe Bild 7), ist es von Vorteil, dass TRACE32 unterschiedliche Sichtweisen auf diese Information anbietet.

Für die Fehlersuche in einem Task bzw. für task-spezifische Laufzeitmessungen lässt sich die Traceinformation gezielt für einen einzelnen Task darstellen.

Stehen Fragestellungen wie „Von welchen Cores wurde mein Task bearbeitet?“ oder „Wie sieht die Laufzeit-Auslastung meiner Cores aus?“ im Vordergrund, ist es zweckdienlich, die Traceinformation für alle Cores gemeinsam darzustellen. Bild 8 zeigt eine solche Darstellung. Die Core-Nummer (0 oder 1) gibt hier an, auf welchen Cores die einzelnen Programmabschnitte gelaufen sind.

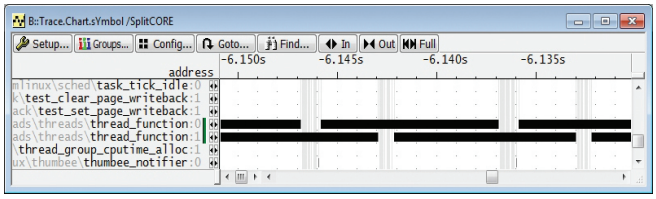

Bild 8: Die Traceauswertung zeigt, von welchen Cores die einzelnen Programmabschnitte bearbeitet wurden.

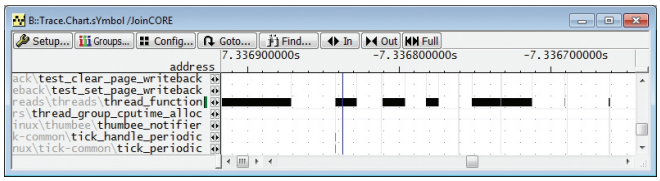

Bild 9: Die Traceauswertung analysiert das SMP-System als Ganzes; von welchem Core welcher Programmabschnitt bearbeitet wurde, spielt keine Rolle.

Untersucht man das SMP-System als Ganzes, hat man unter Umständen kein Interesse daran, zu erfahren von welchem Core welcher Task bzw. Programmabschnitt bearbeitet wurde. Auch für diese Sichtweise auf das SMP-System bietet TRACE32 Darstellungsoptionen (siehe Bild 9).

Bei der Aufbereitung und Auswertung der Traceinformation für SMP-Systeme gibt es noch viel Spielraum für zukünftige Erweiterungen. Lauterbach wird auch 2010, unter Berücksichtigung der Anforderungen seiner Kunden, neue Darstellungs- und Auswertefunktionen hinzufügen.