Germany	Lauterbach GmbH
France	Lauterbach S.A.R.L.
Great Britain	Lauterbach Ltd.
Italy	Lauterbach Srl
USA	Lauterbach Inc.
China	Suzhou Lauterbach Technologies Co. Ltd.
Japan	Lauterbach Japan Ltd.

LAUTERBACH
DEVELOPMENT TOOLS

DEBUGGER, REAL-TIME TRACE, LOGIC ANALYZER

Debugging with ARM CoreSight

ARM CoreSight is a good example of the debug and trace concepts for heterogeneous multicore processors.

To process the many tasks within an embedded system, processors that contain different core types are increasingly used. To be able to properly debug such a system, two conditions must be met:

1. The multicore processor must have suitable on-chip debug and trace logic.

2. The development environment must support debugging of the individual cores and also the overall system with intelligent test and analysis functions.

This article describes how the TRACE32 development environment meets these requirements in conjunction with the CoreSight on-chip debug and trace technology.

What Is CoreSight?

CoreSight is the name of the on-chip debug and trace technology provided specially by ARM for multicore processors. However, CoreSight is not designed as a fixed logic block but rather, like a construction kit it provides many different components. In this way, the designer of the multicore processor can define the scope of the functions provided for debugging and tracing. CoreSight offers great freedom of configuration. Integrating suitable debug and trace options on the processor often requires the specialist knowledge of the tool manufacturer. Our experts have been advising developers worldwide on this subject for many years during the design phases of the latest generations of processors.

The construction kit concept of CoreSight naturally has an effect on the development tool used. If the tool knows the processor and its CoreSight component configuration, debugging is very simple. For new processors the construction kit concept requires the tool to have a high degree of flexibility. Although CoreSight configuration information can be read from the processor, it is still often necessary to clarify details of the implementation with the processor's designer.

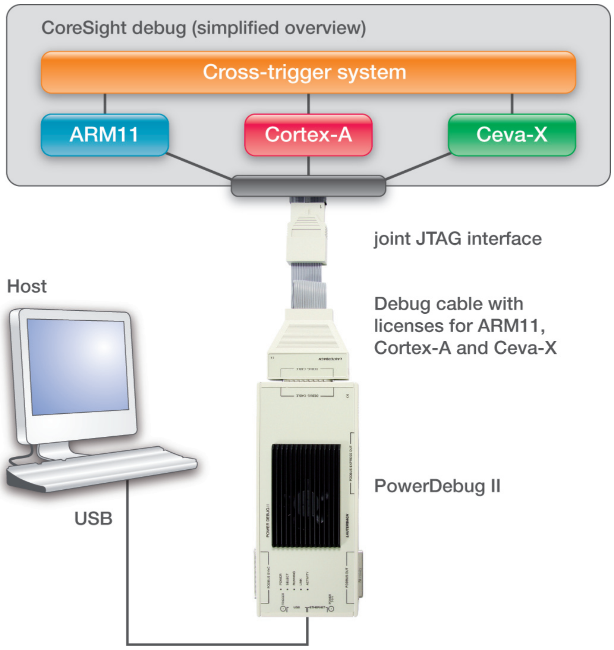
A heterogeneous multicore processor consisting of the RISC cores ARM11 and Cortex-A as well as a Ceva-X DSP was chosen for the following examples.

CoreSight Debug

For processors with CoreSight, all cores are debugged over a joint JTAG interface. A development environment for our specimen processor consists of the following TRACE32 products (see Figure 1):

- A universal PowerDebug module connected to the host over a USB or an Ethernet interface

- A debug cable with licenses for the ARM11, Cortex-A, and Ceva-X architectures



CoreSight debug (simplified overview)
Cross-trigger system
ARM11
Cortex-A
Ceva-X
joint JTAG interface
Host
Debug cable with licenses for ARM11, Cortex-A and Ceva-X
PowerDebug II
USB

Fig. 1: A TRACE32 development environment for CoreSight Debug

In heterogeneous multicore processors, the individual cores usually work on their tasks relatively independently of each other. It therefore makes sense to start a separate TRACE32 instance to debug each core (see Figure 2 on the next page).

»

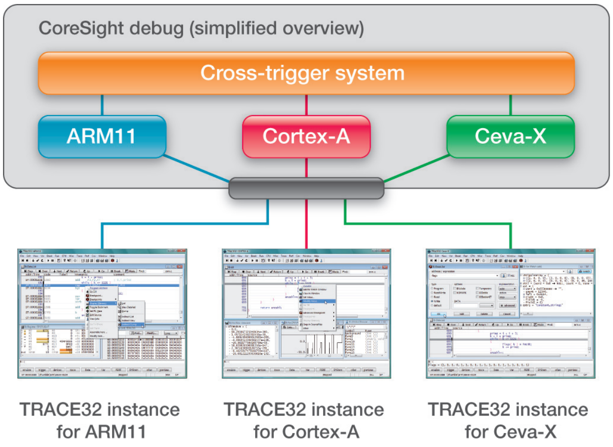

CoreSight debug (simplified overview)
Cross-trigger system
ARM11
Cortex-A
Ceva-X
TRACE32 instance for ARM11
TRACE32 instance for Cortex-A
TRACE32 instance for Ceva-X

Fig. 2: A separate TRACE32 instance is started to debug each core

However, to test that the cores are working properly together, it must be possible to run debugging across the cores. For this purpose, CoreSight provides a cross-trigger system that enables synchronous debugging of all cores: If a core stops at a breakpoint, the other cores are also stopped synchronously. This means the user can easily visualize the context of the individual cores at any selected place in the program.

In addition to this basic function for multicore debugging, TRACE32 can provide other useful debug functions depending on the CoreSight configuration. See the box on the right for a summary of all TRACE32 features for CoreSight Debug.

CoreSight Trace

A common interface is also provided for the trace information from all cores. Under CoreSight, a component for generating trace information can be assigned to each core. For our specimen processor, these are the following components:

- ARM ETM for the ARM11 and the Cortex-A
- Ceva-X ETM for the Ceva-X (see also Figure 3)

Every trace component generates information about the instructions its core has executed and the data accesses that have been made. To provide this trace information at the joint interface, the Funnel combines the trace data into a single data stream. This is then output at a trace port or stored in an on-chip trace memory.

TRACE32-Features for CoreSight-Debug

- Flexible support for multicore processors with CoreSight; Lauterbach offers debuggers for all ARM/Cortex cores as well as a wide range of DSPs
- Debugging over the JTAG interface and the Serial Wire Debug Port
- Runtime access to the physical memory and the peripherals register
- Synchronous debugging of all cores and the peripherals
- The power-down mode of a core has no effect on the debugging of the other cores

Off-Chip Trace Port

Using 18 processor pins (16 pins for the actual trace information and two for control signals), the trace data of all cores can be output to an external trace tool. For »

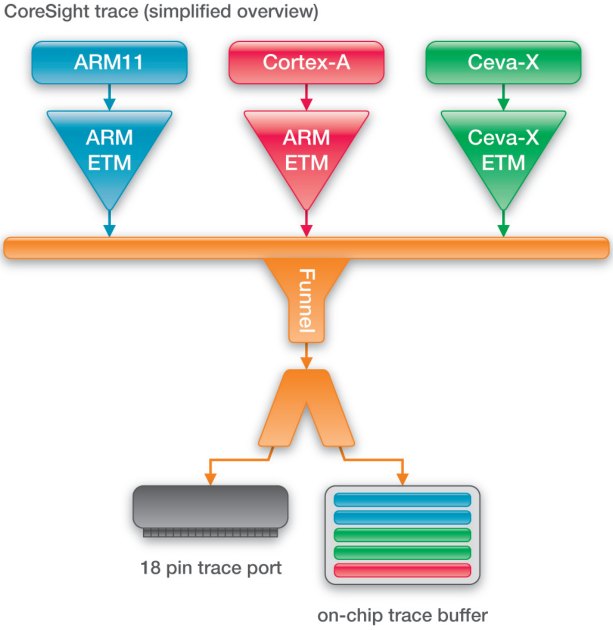


Fig. 3: Every core generates its own trace information

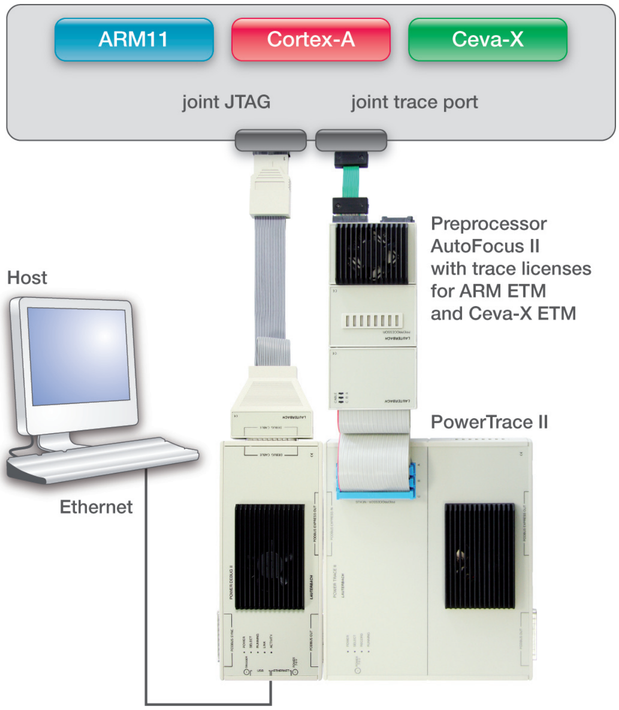


Fig. 4: A TRACE32 development environment for CoreSight Debug and CoreSight Trace

off-chip recording and analysis by TRACE32, the following products must be added to the development environment in Figure 1 (see Figure 4):

- A universal “PowerTrace II” module that provides up to 4 GBytes of trace memory
- An “Preprocessor AutoFocus II” for accessing the trace data at the trace port. In this case, the Preprocessor AutoFocus II must contain trace licenses for the ARM ETM and the Ceva-X ETM.

On-Chip Trace Memory ETB

A pin-saving alternative to the trace port is the on-chip trace memory known as the CoreSight *Embedded Trace Buffer* (ETB). However, its capacity is much smaller than an external trace tool – normally only 2 to 8 KB.

If the trace data is saved in the ETB and then read over the JTAG interface, the debug cable in Figure 1 must also contain trace licenses for the ARM ETB and the Ceva-X ETB.

Trace Analysis

After recording, the developer can display and analyze the trace data for each individual core. For this purpose, each TRACE32 instance reads its trace data from the common trace memory.

To analyze the interaction of the cores, their trace displays can be configured to show the trace entries of all cores in a direct time relationship. For example, if a trace entry is selected in the ARM11 instance, the other two TRACE32 instances mark the instruction that was executed by their core at that moment.

Similar to the debug options, the trace options available in TRACE32 depend on the current CoreSight configuration. The trace options ease systematic trouble shooting and allow an overall analysis of system performance. For a summary of the trace features, see the box below.

TRACE32-Features for CoreSight-Trace

- Flexible support for multicore processors with CoreSight; TRACE32 supports the analysis of trace information for the ARM ETM and a wide range of DSP ETMs
- Trace of bus cycles of the AMBA AHB bus
- Trace of data output of the application with the help of the *Instrumentation Trace Macrocell* (ITM)
- Output of trace data at a trace port or storage in the on-chip trace memory
- The trace data generation components can activate each other by means of the cross-trigger system
- Time-correlated visualization of trace data for the individual cores
- Code coverage and comprehensive runtime analyses