µTrace®

Intelligent Debugging and Tracing for ARM® Cortex™-M

LAUTERBACH
DEVELOPMENT TOOLS

Intelligent Debugging and Tracing for ARM® Cortex™-M

A new all-in-one debug and trace solution has been developed by Lauterbach in response to the breakthrough of Cortex-M processors into the embedded market. This lower cost system called µTrace specifically targets the Cortex-M family.

µTrace® Characteristics

- Support for more than 1000 different Cortex-M based chips
- USB 3 interface to the host computer
- Support for standard JTAG, Serial Wire Debug, and cJTAG (IEEE 1149.7)
- 256 MByte trace memory
- 10- / 20- / 34-pin half-size connector for target hardware and a wide variety of adapters to work with other connectors
- Voltage range 0.3V to 3.3V (5V tolerant inputs)

Debug Features

- C/C++ debugging
- Simple and complex breakpoints
- Read and write memory during program runs
- Flash programming support for FLASH memory on microcontrollers and external flash on the target system.
- OS-aware debugging provides the user with symbolic debugging capability in operating system. With the specification of the OS name a menu is added to TRACE32 PowerView that allows easy access to task lists and other OS specific information.
- The LPC43xx from NXP is the first chip that contains a Cortex-M4 and a Cortex-M0. It goes without saying that the µTrace includes the approved TRACE32 AMP multicore debugging technology.

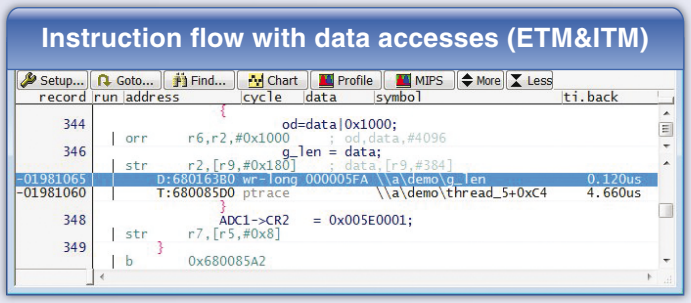

Fig. 1: By combining ETM and ITM trace data, read/write accesses can be integrated seamlessly into the instruction flow.

Extract from supported vendors and chip families	
Actel / Microsemi	SmartFusion, SmartFusion2
Analog Devices	ADuCM3
Atmel	SAM3, SAM4
Cypress	PsoC 5
Energy Micro	EFM32
Freescale	Kinetis, Kinetis-L
Fujitsu	FM3
Infineon	XMC1000, XMC4000
Nuvoton	NuMicro
NXP	LPC800, LPC1xxx, LPC4300
Samsung	S3FM02G
STMicroelectronics	STM32
Texas Instruments	Stellaris
Toshiba	TX03

Trace Features

Cortex-M processors may include an Instrumentation Trace Macrocell and an Embedded Trace Macrocell. In this case the µTrace can provide the following trace features:

- ITM over Serial Wire Output
- 4-bit ETMv3 in Continuous mode for Cortex-M3/M4.
- ITM over TPIU for Cortex-M3/M4
- Three recording modes: FIFO, STREAM and Real-time Profiling (for details refer to page 4)
- Combining ETM and ITM trace data allows a seamless integration of read/write accesses into the instruction flow and OS-Aware tracing (for details refer to page 3).
- Analysis of task and function run times
- Code coverage analysis
- Energy measurement using TRACE32 Analog Probe

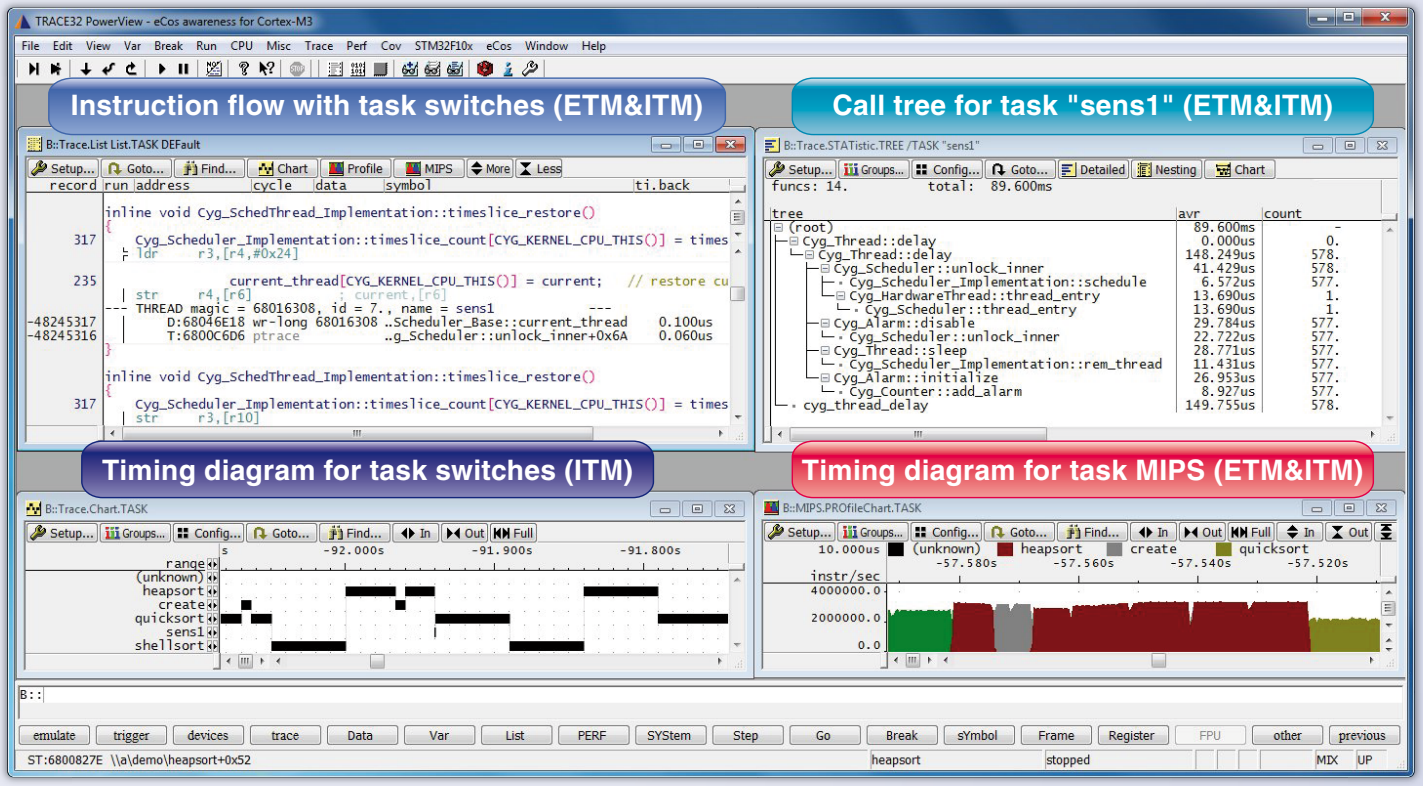

Fig. 2: Through the combination of ETM and ITM trace data, extensive trace analysis can be provided for the eCos operating system.

Combining ETM and ITM Trace Data

For Cortex-M3/M4 processors, trace information can be generated from two different sources (see Figure 3).

The **ETMv3** generates information about the executed instructions. The **ITM** generates information about the performed read/write accesses assisted by the Data Watchpoint and Trace Unit (DWT).

The ITM trace packets for read/write accesses contain the following information: data address, data value, program counter. Through analysis of the program counter, these separately generated data accesses can be seamlessly integrated into the instruction flow (see Figure 1). This in turn leads to significantly faster error location. The cause of an error such as an incorrect data value being written to an address can be easily found if the write accesses are embedded into the overall instruction flow.

OS-Aware Tracing

If an operating system is running on the Cortex-M3/M4, task switch information becomes essential for the trace analysis.

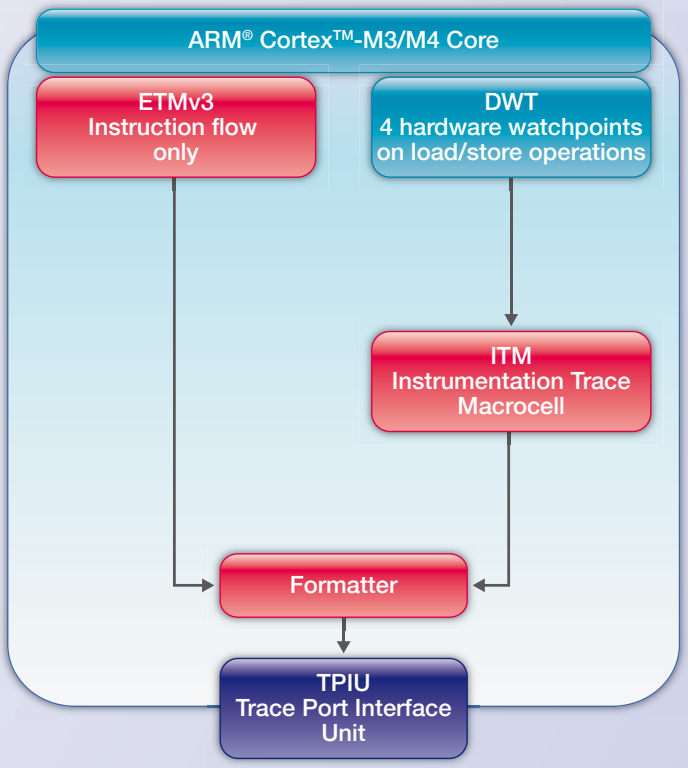

Fig. 3: Block diagram of the trace logic in a Cortex-M3/4 processor.

The following method can be used to include information about task switches:
The ITM can generate trace information for the kernel's write cycle of the current task identifier to the corresponding OS variable.

As described above the write access information can be integrated seamlessly into the instruction flow trace. This improves the readability of the trace listing (see Figure 2 on page 3). The integration of the task switch into the instruction flow trace also forms the basis for the run-time analysis shown in this figure.

Three Recording Modes

To record the trace information generated by the Cortex-M3/M4 processors, µTrace supports three modes:

- **FIFO mode**
  Storing the information in the 256 MByte memory of the TRACE32 µTrace.

- **STREAM mode**
  Streaming the information to a hard-disk on the host computer through USB 3 with up to 100MBytes/s.

- **Real-time Profiling**
  The trace information is streamed to the host computer and analyzed during run-time.

For the first two recording modes, the trace information is collected and the trace analysis is undertaken after recording is completed.

Each recording mode has its specific strengths. **FIFO** is the most commonly used mode. It is quick and often all that is needed for error location and run-time analysis.

STREAM mode:
The ETMv3 implemented on Cortex-M3/M4 processors has no built-in trigger or trace filter, therefore it is not possible for the user to record only those program segments required for troubleshooting. This can mean trace data might have to be collected for a relatively long period in order to cover the area needed for analysis. In this case STREAM mode is the best option.

The STREAM mode, however, places high demands on the debug environment:

- The large amount of data that results from streaming requires a 64-bit TRACE32 executable. This is needed to handle the large number of trace entries that will be collected.

- The transfer rate between µTrace and host computer must be fast enough to stream all trace data without data loss. The 256 MByte memory of the µTrace is used to cushion load peaks from the trace port (TPIU).

Real-time Profiling is particularly suitable for performing object statement and object branch coverage. The coverage analysis can be followed live on the screen and the test results are visible immediately (see Figure 4). Lines marked as “ok” have already been covered, lines marked as "not exec" may require additional test stimuli.

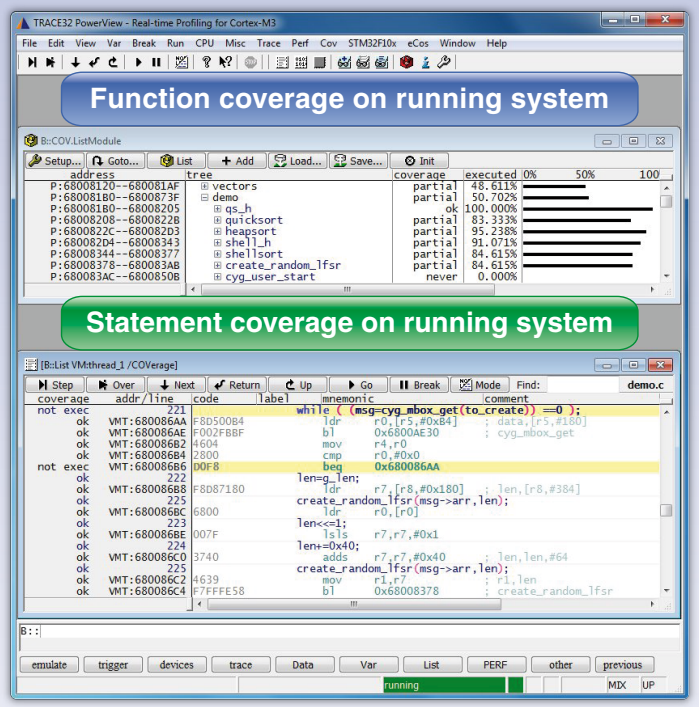

Fig. 4: Real-time profiling enables code coverage analysis to be followed live on the screen.

For more information: *www.lauterbach.com/1658*

LAUTERBACH DEVELOPMENT TOOLS
Lauterbach GmbH
Altlaufstraße 40 • 85635 Höhenkirchen (Germany) • Tel: +49 8102 9876 0 • sales@lauterbach.com