Tools Correlate Code and Current

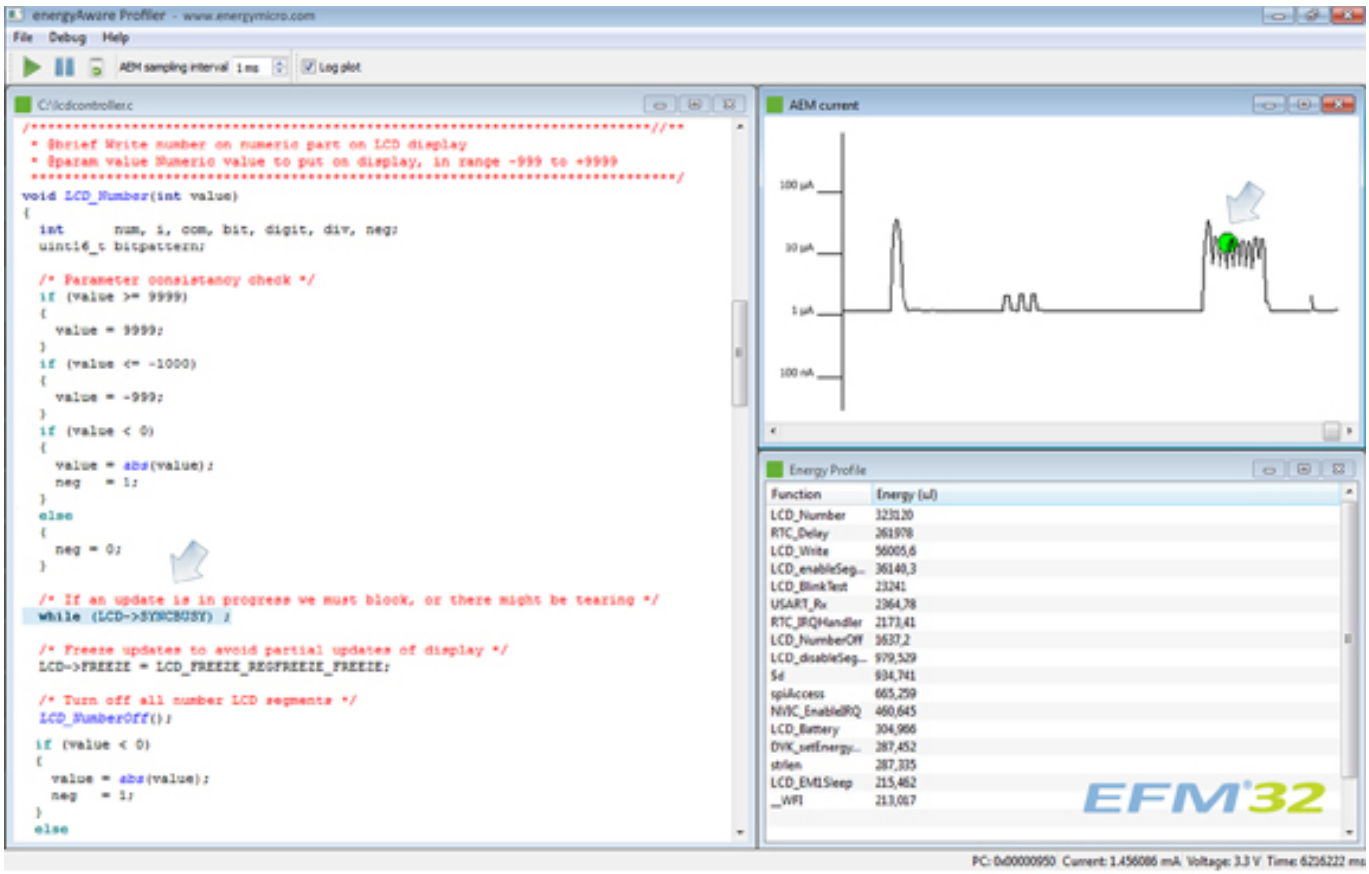
Jon Titus, Senior Technical Editor

The emphasis on low-power embedded systems can give engineers heartburn. They must strive to enhance a system's performance, add capabilities, and make its battery last for as many as 10 years. A combination of hardware and software tools now help engineers understand power consumption without the antacids.

The ARM Cortex-M3 processors include sophisticated and flexible debug and monitoring capabilities that software-development tool suppliers use to let engineers trace program flow, set breakpoints, watch variables, and so on. Energy Micro, based in Oslo, Norway, has combined that information with electric-current measurements so engineers can correlate power use with code execution. To find out more, I talked with Øyvind Janbu, chief technology officer at Energy Micro.

"In our EFM32 Gecko Development Kit and EFM32 Gecko Starter Kit we provide hardware outside the ARM Cortex-M3 microcontroller to measure current and voltage," said Janbu. "Current measurements have a high dynamic range; from about 0.1 ?A up to 100 mA. So when you run the energyAware Profiler on a host PC connected to the kit's board you can monitor a wide range of current consumption."

Energy Micro uses the Cortex-M3 serial wire output (SWO) pin on the MCU to obtain program-counter information from the CPU. The SWO pin can provide other test and debug information, too. And the Cortex-M3 has even more test and debug capabilities that tool developers take advantage of. But in most cases, engineers leave that aspect of software to tool suppliers.



Art 1. The energyAware Profiler displays current use, C/C++ code, and energy use by function. The green dot (blue arrow, upper right) selects part of a current plot that the profiler relates to a line of code (blue arrow, lower left). Courtesy of Energy Micro.

"Engineers don't think in terms of program-counter values," noted Janbu. "They think about lines of assembly-language or C/C++ code. So, they give the energyAware Profiler tool their object code with debug information in it. The tool then correlates the program-counter value to the machine code. And because the tool also knows about the functions in the C/C++ source program, it can indicate how power-use changes as various functions run. The object code can come from development software from companies such as Keil and IAR, or from the GCC [GNU Compiler Collection]."

"We obtain the program-counter value at a maximum of once every 12 clock cycles," continued Janbu. "We can get the value faster at lower MCU clock frequencies, though. Current measurements in the latest software occur at 10 Ksamples/sec, where each sample provides an average measurement over the sample period. You cannot get a current measurement for each instruction because you would have to deal with pipelined instructions. And you would get more electrical noise from your system at higher sampling rates."

But what happens when engineers create prototype circuits and no longer need an Energy Micro board? "Each kit has a measured-current and an unmeasured-current power section," explained Janbu. "So they power their prototype board or circuit from our board's measured-current section and connect their MCU's SWO pin to our board. They might want to measure only the power the MCU uses, or they might

choose to measure the power for the entire prototype circuit."

The Energy Micro boards can isolate the SWO signal from the on-board Cortex-M3 MCU and a second on-board MCU measures current and handles USB communications with a host PC.

Engineers can quickly locate a problem, such as too much time spent in a power-hungry loop. They watch a windowed display of power versus time, select a high-current point, and immediately see the related code, as shown in the nearby figure.

"The profiler can catch obscure things, too," said Janbu. "You might send a burst of data to an LCD and discover the display controller spends a lot of time waiting for data that already arrived. So rather than wait in a long loop and waste power, you use an interrupt-driven display function. And you can annotate each interrupt event in the current graph, to give additional information on what causes a particular change in current consumption."

You might set the profiler to run over a long period. When you return you notice the the MCU stayed on for 10 seconds rather than for a millisecond. You can go back to the current data and uncover what happened. Perhaps you turned off an interrupt too soon or used a too-large timing value.

Energy Micro isn't along in the energy-to-code correlation arena. At the Embedded Systems Conference in April 2010, IAR Systems announced a similar type of power-code correlation tool within the company's Embedded Workbench tools. The company's demonstration showed current measurement with a J-Link debug pod. IAR expects to offer the new software later in 2010, but declined to make a spokesperson available now to answer questions.

For further reading

Download the free energyAware Profiler at:
www.energymicro.com/software/efm32-software [1].

Yiu, Joseph, "The Definitive Guide to the ARM Cortex-M3," Elsevier. ISBN: 978-1-85617-963-8.

"CoreSight Technology, System Design Guide," ARM Ltd. tinyurl.com/2fm2sbo.

Source URL (retrieved on *12/09/2013 - 10:19am*):
http://www.ecnmag.com/articles/2010/06/tools-correlate-code-and-current?qt-recent_content=0

Links:
[1] http://www.energymicro.com/software/efm32-software