Embedded Systems - Optimizing Energy Consumption in ARM® Cortex™-M-based Low Power Applications with µTrace

Everyone is aware of them - embedded systems in cars, telephones, and a wide range of electronic devices that influence and change everyday life and human behavior. Many of these devices contain small computers. In addition to functionality and computing power, low energy consumption is frequently a key factor in determining the market success of a product, especially for mobile and battery-operated systems. Also power supply and high temperatures put a strain on semi-conductor components and accelerate the ageing process; therefore, it is particularly important for systems in temperature-critical environments to reduce power loss and related heat generation. Slowing down the ageing process extends the service life and increases the quality of the end products.

Modern semi-conductors are often designed with intelligent power supply concepts, enabling you to independently control the power supply to applications. You can vary the frequency, switch off unused modules, and place the entire chip in hibernation by using power down modes. The main task of the software is to effectively implement these concepts, and as a result, reduce energy consumption.

Electrical energy is a product of three parameters; current, voltage and time. These are determined by many factors in the design. With the right tools to monitor the system the designer can understand the causes of changes in voltage and current.

This can only be successful if the power consumption values can be dynamically compared with the actual operation of the system software and hardware. Using this correlation you can take corrective action to reduce excessive power consumption by the optimization of software and hardware resources.
This used to be a difficult task but has become much easier with the new generation of TRACE32 debuggers as they have the correct interfaces to monitor and record both the software at run time and the power consumption of areas of interest. This data is recoded in the internal storage of the tools for analysis by the developer.

Energy profiling describes the method of gathering current and voltage values with the debuggers from Lauterbach and analyzing and displaying the results using the graphical user interface (GUI) PowerView.

Energy Profiling with µTrace

The µTrace is Lauterbach's newest product, which integrates debugging and trace functions into one device. Optimized for ARM® Cortex™-M processors, the µTrace supports a wide variety of derivatives from different manufacturers.

Figure 1 shows the measurements for energy profiling with the µTrace. In this example, the debugger is connected to the target using 2 cables and to the host computer with a USB 3.0 interface. The 20-pin debug cable connects to 2 combined interfaces on the processor debug port. Debugging commands are sent using JTAG/cJTAG/SWD to control the program flow and access the system resources. The processor internal trace module (4 bit ETM) is also configured by this interface and when in operation outputs details of the program flow and data to the tools for storage and analysis. An analog probe from Lauterbach uses a second cable to read the current and voltage measurements.

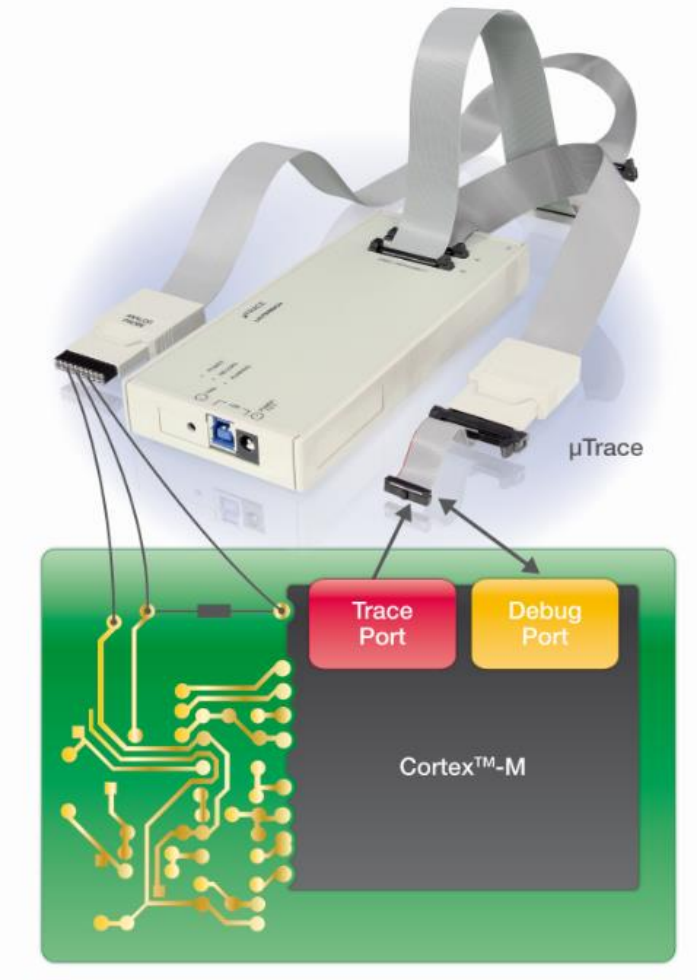

Figure 1: Energy profiling with µTrace

There are 4 channels for measuring voltage and 3 channels for measuring current. A low resistance shunt resistor in the supply line to any section of the system that is of interest is necessary to allow recording of the current values.

With a measurement of the supply voltage and a simultaneous measurement of the voltage drop across the shunt resistor the instantaneous power consumption can easily be calculated.

The µTrace has an internal capacity of 256 MB for storing trace data measurements, as well as current and voltage values. This can be expanded by streaming the trace data during run time via the tools to a hard drive on the host computer.

PowerView is the user interface that provides all the control and configuration functions for configuring the tools both for debugging and power data collection and analysis. The settings can be configured graphically using the respective dialog boxes or by using commands. For complex settings all configurations can be saved into a script file so they can easily be repeated

Energy profiling only requires a few settings. The settings for the debug and trace interfaces of the processor must be configured together with those of the debugger and the analog probe.

As a prerequisite you must have a functioning debug channel so the µTrace and the processor can communicate with each other over the same JTAG/cJTAG/SWD interface.

In the second step, the trace functions are configured. With ARM® Cortex™-M, the trace data to be collected and the port width are defined by the combined settings of the Embedded Trace Macrocell (ETM), Instrumentation Trace Macrocell (ITM), and Trace Port Interface Unit (TPIU). By configuring macros you can display program trace and data trace, including time stamps. For a limited trace port width (maximum 4bit with this processor architecture), you should choose settings for the content and the amount of trace data being generated to fit within the available bandwidth and at the same time remember the program trace should be as detailed as possible to provide the most accurate correlation.

Data collection with the analog probe, as shown in figure 2, is configured using a separate menu in which the current and voltage channels are activated and the values of the shunt resistors are entered. You can optimize the use of available data storage for each channel with dedicated trigger settings. You can also activate three power channels whose values come

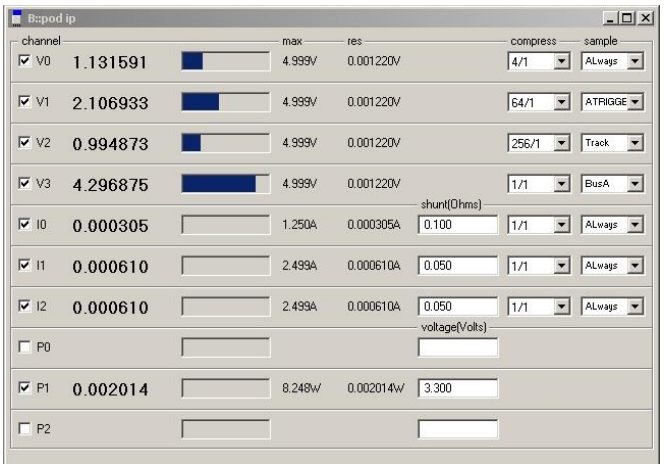

Figure 2: Configuration of the analog probe

from the calculation made on the associated current and voltage channels.

To simplify demands on the measurement system for a power channel, the voltage can be assumed to be constant; however, this reduces the accuracy of performance calculations. When set up the debugger will record trace and power values on execution of the application or from defined trigger conditions.

PowerView - Correlation of Measurements and Evaluation of Results

While the current and voltage measurements are being read at regular intervals, the trace data is transmitted by the core in compressed packets with a time stamp from the target. After uncompressing the trace data, both series of measurements are converted to a common time axis with the help of the internal µTrace system clock and each power measurement is assigned a time stamp. Both the power and trace files can now be correlated to the high level language source files.

The power measurements can now be presented with the high level language in either list or graphical formats. Figure 3 shows the results of a measurement where the use of external storage causing a peak in the energy consumption is being investigated. In this example, while an application loop is performed multiple times and the cache is repeatedly activated and deactivated, the power supply for the memory interface and the core is being measured.

The upper window in figure 3 shows the run time as the nested functions are called. This shows how the cache being activated makes accesses quicker. The middle diagram shows current flow to the external storage interface.

The higher current values are a result of the external storage being accessed during the second run at the point when the cache is deactivated and before the third run at the point when the cache is reloaded. The use of the cache is also apparent in the power supply to the core, whose timeline is visible in the bottom window. All the windows are synchronized so that changing the time period in one all will update all of the others. It is therefore easy to track individual power measurement to the code running at that time.

Further information can be calculated from the measurements, or statistically deduced and presented including the calculation of minimum and maximum values and power consumption by function. Finally, all of the recorded data can be saved in files and then reloaded and analyzed offline within the TRACE32 simulator.

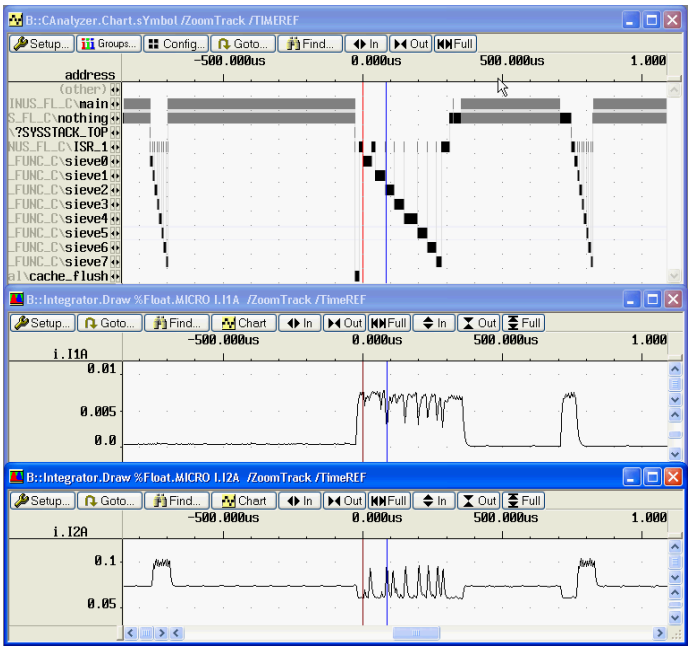

Figure 3: Presentation of results with PowerView

The µTrace, analog probe, and PowerView create a compact system for determining consumption that can be managed and operated by any developer. However, as with any system there are some technical limitations. Depending on the time stamp provided by the core in the trace macro, the accuracy of the correlation between the measured values and the trace data can be limited. This does not cause problems when assigning values at the function level; however, assigning values on the command level is not practical. Due the design of command pipeline and multicore architectures several commands can be processed at the same time so a power consumption value cannot be assigned to an individual command. Also activating the trace functionality also causes an increase in power consumption which can lead to recording a false value if you are undertaking a high grade optimization.

Energy profiling with the µTrace is an extremely effective method for power optimization that can be easily implemented into the development program helping with the program and compiler optimization and chip commissioning.
The µTrace is supplied as a system with software, debug cable, power supply and USB cables. The analog probe and a range of adaptors are also available as accessories.

Lauterbach GmbH
Altlaufstrasse 40
85635 Hoehenkirchen, Germany
Phone: +49 8102 9876-0
Fax +49 8102 9876-187
info@lauterbach.com

September 2013