

TN08001

Power consumption and system performance testing using hardware and software floating point math

Rev. 01 — 18 June 2008

Technical note

Document information

Info	Content
Keywords	EnergyBench, AutoBench, LPC3180

Revision history

Rev	Date	Description
01	20080618	Initial version.

Contact information

For additional information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

Designing an embedded system is usually a challenging task. Almost every such system has a relatively unique hardware configuration. Custom code must often be written for a specific embedded operating system. On top of this there are usually very tight energy constraints.

Early in the project's design phase, an engineer can apply a full spectrum of methods to evaluate the performance of various microcontrollers. Reviewing and compiling information from datasheets represents one end of the spectrum. The other end of the spectrum involves custom performance testing and power measurement methods using some type of evaluation board. Both methods have their disadvantages. Relying on datasheet comparisons is risky, while testing multiple hardware options is usually impractically time-consuming and expensive. In this article, we examine a mid-point solution using industry-standard benchmark data as a means to evaluate performance and energy consumption during the early stages of product design, when key components are being selected.

Our goal was to investigate system performance of NXP's LPC3180 microcontroller under several different test criteria and relate this collected data to energy consumption. We needed to measure performance and power consumption simultaneously to accurately determine the overall energy consumed during a specific workload. We used three steps in the evaluation process. First, we characterized the system by running various system benchmarks while varying different system parameters. Second, we interpreted the collected characterization data to establish the behavior of the system. Third, we determined from the behavior of the system how to set control parameters that make the system exhibit the desired behavior.

2. Characterization

In theory, performance testing is the qualified or quantified validation of an operational specification. In practice, a system specification may not be detailed enough to define a through qualification test, or creating the test may be too expensive to warrant its development. A good compromise to characterize a system is to use benchmarks as a test or series of tests executed in software that provide metric data one can use to compare the characteristics of different systems.

To characterize the LPC3180 microcontroller, we selected a set of performance tests from the AutoBench suite of the Embedded Microprocessor Benchmark Consortium (EEMBC). These benchmarks help predict the performance of microcontrollers in automotive, industrial, and general-purpose applications. We ran each benchmark test through multiple consecutive iterations to remove the effects of some setup code that ran once at the beginning of each test. An advantage of using this industry-standard benchmark suite is the utility of comparing the resulting data with test data from other microcontrollers of similar architecture to gauge overall system performance.

The LPC3180 is an ARM926-EJS based microcontroller with a hardware vector floating point coprocessor and a 32k instruction cache (I-cache). Our testing characterized the floating point (FP) coprocessor and instruction cache performance. We ran the AutoBench benchmarks at different microcontroller operating frequencies, while measuring the energy consumed during the execution of each benchmark using EnergyBench. EnergyBench is another EEMBC tool that measures the amount of energy a processor consumes during a benchmark workload run. The data gathered from EnergyBench gives insight into the energy efficiency of the microcontroller under various

workloads. Having selected the tools to characterize the microcontroller, the next step was to determine the behavior of the microcontroller under various operating conditions.

3. Evaluating a microcontroller's behavior

To analyze the microcontroller's behavior we needed to determine the aggregate system response under various conditions. In our test case, we wanted to evaluate the behavior of the FP coprocessor and the I-cache in the LPC3180 microcontroller. We ran the AutoBench benchmark suite while changing four parameters: the operating frequency, the CPU core voltage, the state of the I-cache, and the state of the FP coprocessor. [Fig 1](#) shows a diagram of the AutoBench/EnergyBench test setup. It consists of three parts: the data acquisition system (DAC), the software development environment, and the target under test. The DAC, from National Instruments, is connected to a PC running the EnergyBench power and energy measurement software. The software development environment used the Keil Integrated Development tools to compile, download, and run the AutoBench benchmarks. The target under test was the LPC3180 integrated into a phyCORE-LPC3180 board. We isolated the three power supply voltages to the LPC3180 microprocessor so EnergyBench could measure the power consumed during the AutoBench benchmark testing and calculate the total energy consumed during each test.

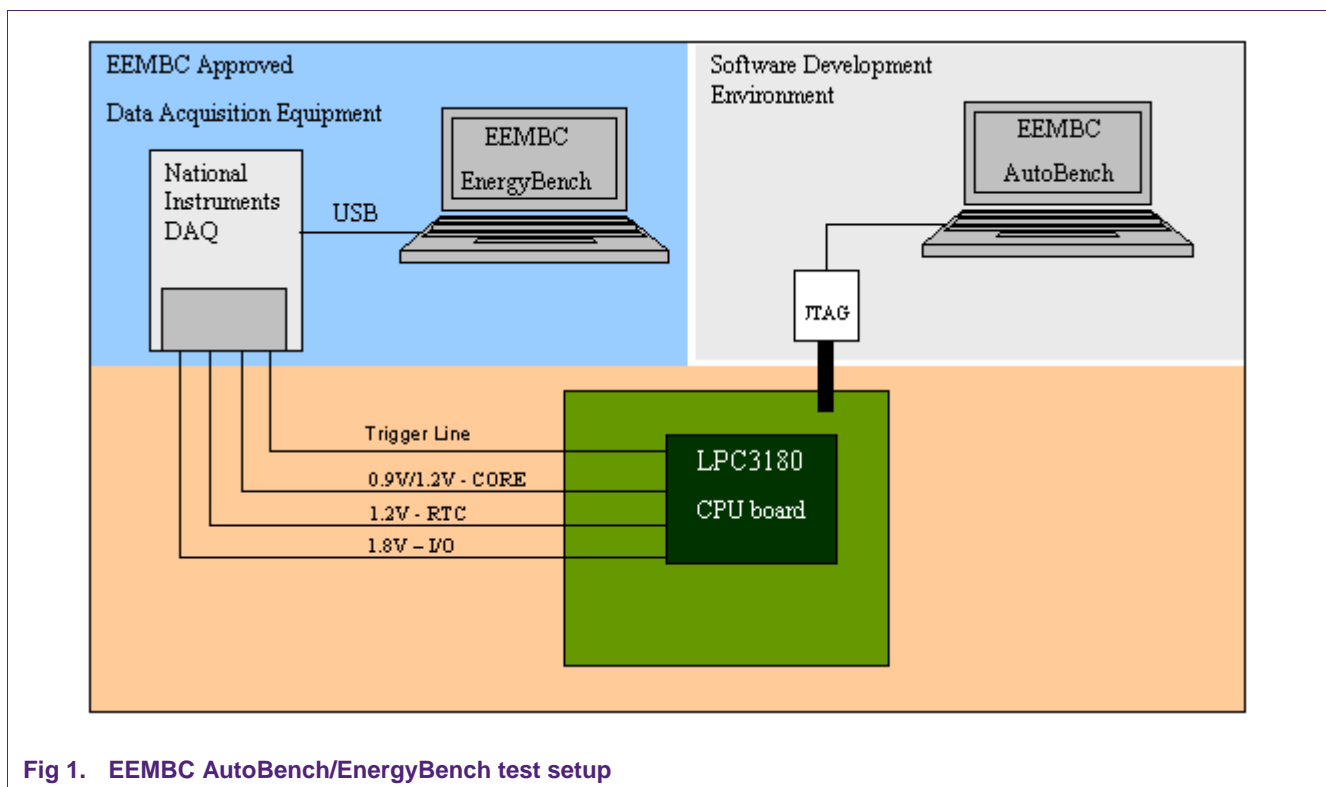


Fig 1. EEMBC AutoBench/EnergyBench test setup

We ran the AutoBench benchmarks at 4 different frequencies (13, 52, 104, and 208 MHz), in combinations that included turning on or off the floating point coprocessor, and turning on and off the instruction cache. The floating-point coprocessor was pseudo-disabled by forcing the compiler to use software floating point for any required floating point calculations.

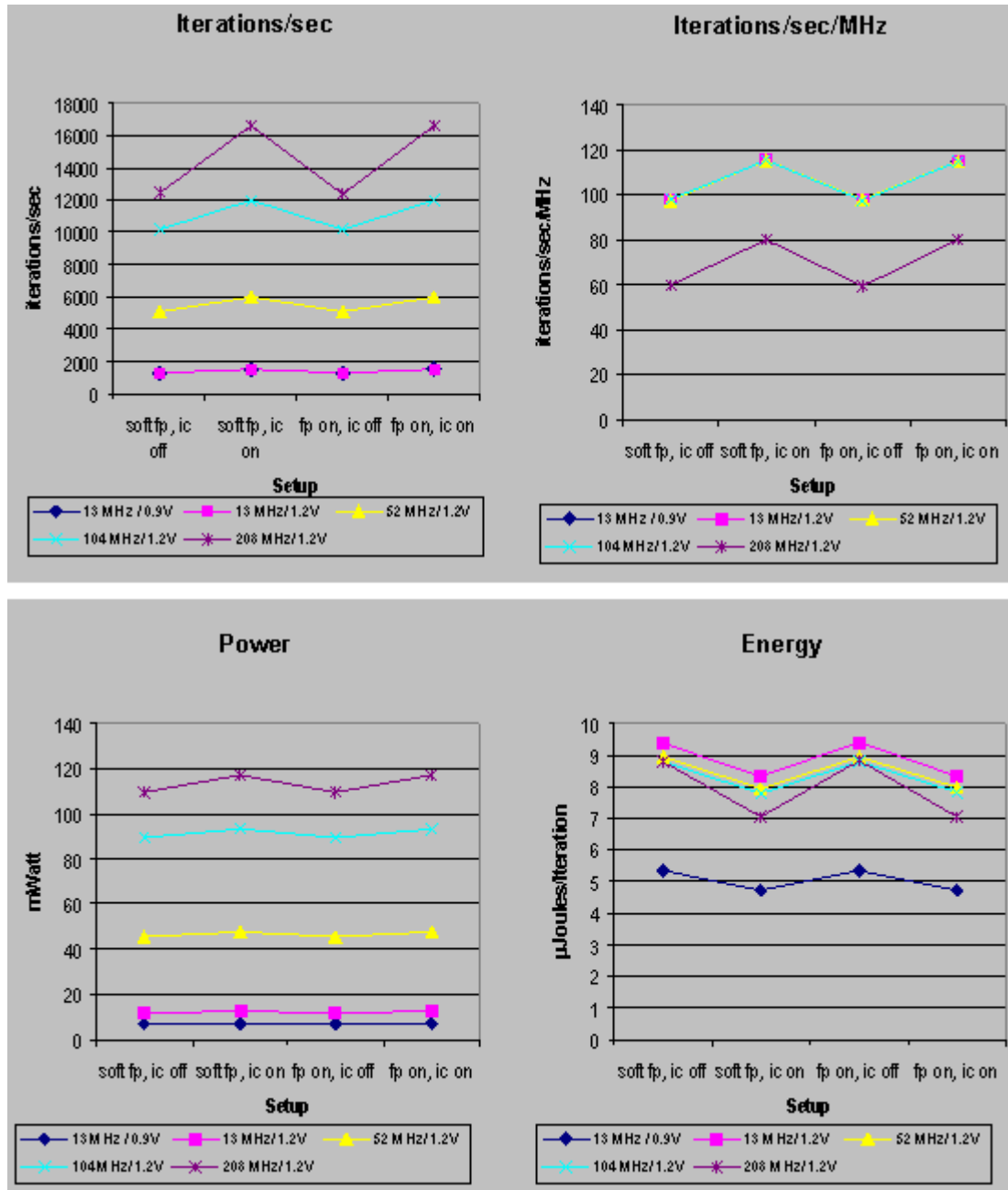


Fig 2. AutoBench test – Finite impulse response filter (aifirf01)

We collected more data than can easily be presented in this overview so we present two representative cases to show how collecting characterization data can determine the behavior of a system. [Fig 2](#) graphically shows the results of the test data for EEMBC's "Finite Impulse Response Filter" (FIR). [Fig 3](#) graphically shows the results for data collected for EEMBC's "Basic Integer Floating Point." We ran two different benchmark tests at 13 MHz, changing the CPU core voltages between 0.9 V and 1.2 V. Benchmark runs taken with the CPU clock set at 208 MHz had the AHB clock set to its limit of 104 MHz. At all other test frequencies the CPU clock and AHB clock were the same.

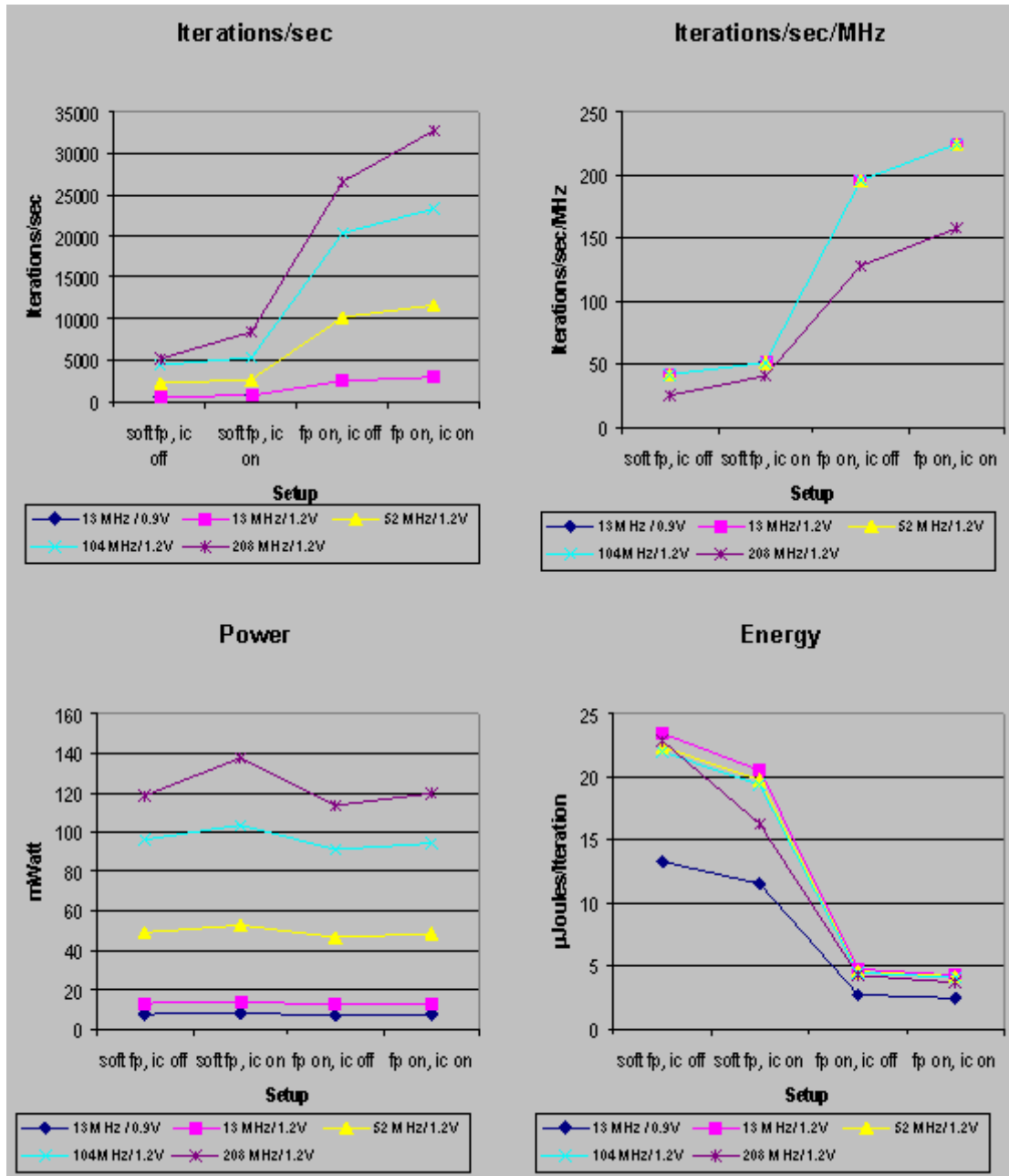


Fig 3. AutoBench Test – Basic integer floating point (basefp01)

We selected the FIR benchmark as a baseline test because it contains no floating-point calculations and it provides useful data when compared to the Basic Integer Floating Point benchmark. The data from these two benchmarks gives us the necessary information needed to determine the behavior of the I-cache and the FP coprocessor.

Let's look at I-cache performance first by observing [Fig 2](#) and the graph labeled Iterations/sec. The data shows that at all frequencies, the absolute performance of the microcontroller is better with the I-cache enabled. Second, even though the I-cache

provides greater absolute performance as the CPU clock is increased, the relative magnitude of the improvement is not linear. The reader can verify this behavior by looking at the graph labeled Iterations/sec/MHz that shows that performance increases linearly at approximately 100 Iterations/sec/MHz for all CPU clock frequencies except for the run at 208 MHz where the performance falls off to 60, or 80 Iterations/sec/MHz depending on whether the I-cache is enabled or disabled.

Obviously, the system performs faster with I-cache enabled because the CPU makes fewer accesses to the AHB RAM when it executes instructions from the I-cache.

The non-linear performance behavior is the result of the maximum limit of 104 MHz for the AHB clock. When the AHB clock is slower than the CPU clock, the CPU must wait longer to read instructions from the RAM on the AHB bus, and the result is a smaller relative performance increase per MHz.

Let's look at the I-cache effects on energy consumption. If we were to take into account only the absolute power consumption in the Power graph in [Fig 2](#), we might conclude that turning off the I-cache would result in energy savings to the overall system. However, the EnergyBench data shows that when the I-cache is enabled, the energy consumed per benchmark iteration is actually lower than when the I-cache is disabled. A more detailed look at the energy graph shows that with the I-cache enabled, the energy consumed per iteration at 208 MHz is even less than the other operating frequencies tested at 1.2 V; in fact, there is an improvement of 10 % to 12 %. In other words, it is more energy efficient to run with the I-cache enabled at high speed (208 MHz) for a short period than to run with the I-cache enabled at a lower frequency (52 MHz or 104 MHz) for a longer period to execute the same benchmark.

Look at the effects performance and energy consumption of using the FP coprocessor by starting with [Fig 3](#) and the iterations/sec graph. This graph is quite dramatic in demonstrating the performance effects of an integrated FP coprocessor. At 208 MHz and with the I-cache enabled, the microcontroller runs about 8500 iterations/sec using software floating point, but this value jumps to over 32500 iterations/sec using the FP coprocessor – for a better than 280 % performance improvement.

To examine the effects of the FP coprocessor on energy consumption, look at the energy graph in [Fig 3](#). With the I-cache enabled, the energy per benchmark workload at 208 MHz shows the microcontroller consuming about 16 μ Joules/iteration using software floating point, but less than 4 μ Joules/iteration with the FP coprocessor - an energy savings of more than 75 % while performing the same amount of work.

As mentioned earlier, we also collected data at 13 MHz using supply voltages of 0.9 V and 1.2 V. [Fig 2](#) and the iterations/sec graph show that at 13 MHz the LPC3180 performance benchmark data is identical for both the 0.9 V and 1.2 V runs. However, the power graph shows the power consumed at 1.2 V is about 75 % greater than the power consumed at 0.9 V.

4. System control parameters

In our test example, we used EEMBC characterization tools to determine the behavior of the I-cache and FP coprocessor in our target test system. From this behavior, we can select general configuration parameters that provide the best condition for a system performance with low energy consumption.

Here are some parameter choices for controlling a system for power utilization and performance in environments similar to those tested by the EEMBC AutoBench benchmark suite:

1. Performance is better with the I-cache enabled.
2. Floating point performance and energy consumption is significantly better when using hardware FP rather than software.
3. Energy consumption is better with the I-cache enabled at 208 MHz than at lower frequencies.
4. For low power operation at 13 MHz, it is much better to run with the core voltage at 0.9 V than at 1.2 V.

The statements above are general guidelines. However, even more important is the fact that we have determined system behavior from data derived from industry-standard performance and energy benchmarks that are publicly available and validated by an independent authority. Using the EEMBC AutoBench benchmarks and EnergyBench benchmarks result in consistent behavior analysis that is easily demonstrable to others, and more importantly repeatable and verifiable.

5. Conclusion

If you want to compare the behavior of several different systems, then using test tools with certifiable results to characterize the behavior is very useful. Even more important, the certification data helps a system evaluator compare the same performance characteristics even though the embedded system under test may be very different.

In our test setup, we used characterization tools from EEMBC to determine the behavior of the NXP LPC3180 microcontroller. We then used the behavior information to select the best control parameters for a specific operational environment. Our test example quantified the system behavior of the I-cache and FP coprocessor in our evaluation system using the LPC3180. The collected characteristic data let us define system behavior and gave us a methodology for selecting operational parameters to control performance and energy consumption.

6. Legal information

6.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

6.2 Disclaimers

General — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of a NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is for the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

6.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

7. Contents

1.	Introduction	3
2.	Characterization	3
3.	Evaluating a microcontroller's behavior	4
4.	System control parameters	8
5.	Conclusion.....	9
6.	Legal information	10
6.1	Definitions.....	10
6.2	Disclaimers.....	10
6.3	Trademarks	10
7.	Contents.....	11

Please be aware that important notices concerning this document and the product(s) described herein, have been included in the section 'Legal information'.

© NXP B.V. 2008. All rights reserved.

For more information, please visit: <http://www.nxp.com>
For sales office addresses, email to: salesaddresses@nxp.com

Date of release: 18 June 2008
Document identifier: TN08001_1

