Power estimation on functional level for programmable processors

M. Schneider, H. Blume, and T. G. Noll

Lehrstuhl für Allgemeine Elektrotechnik und Datenverarbeitungssysteme, RWTH Aachen, Schinkelstraße 2, 52062 Aachen, Germany

Abstract. In diesem Beitrag werden verschiedene Ansätze zur Verlustleistungsschätzung von programmierbaren Prozessoren vorgestellt und bezüglich ihrer Übertragbarkeit auf moderne Prozessor-Architekturen wie beispielsweise Very Long Instruction Word (VLIW)-Architekturen bewertet. Besonderes Augenmerk liegt hierbei auf dem Konzept der sogenannten Functional-Level Power Analysis (FLPA). Dieser Ansatz basiert auf der Einteilung der Prozessor-Architektur in funktionale Blöcke wie beispielsweise Processing-Unit, Clock-Netzwerk, interner Speicher und andere. Die Verlustleistungsaufnahme dieser Blöcke wird parameterabhängig durch arithmetische Modellfunktionen beschrieben. Durch automatisierte Analyse von Assemblercodes des zu schätzenden Systems mittels eines Parsers können die Eingangsparameter wie beispielsweise der erzielte Parallelitätsgrad oder die Art des Speicherzugriffs gewonnen werden. Dieser Ansatz wird am Beispiel zweier moderner digitaler Signalprozessoren durch eine Vielzahl von Basis-Algorithmen der digitalen Signalverarbeitung evaluiert. Die ermittelten Schätzwerte für die einzelnen Algorithmen werden dabei mit physikalisch gemessenen Werten verglichen. Es ergibt sich ein sehr kleiner maximaler Schätzfehler von 3%.

In this contribution different approaches for power estimation for programmable processors are presented and evaluated concerning their capability to be applied to modern digital signal processor architectures like e.g. Very Long Instruction Word (VLIW) -architectures. Special emphasis will be laid on the concept of so-called Functional-Level Power Analysis (FLPA). This approach is based on the separation of the processor architecture into functional blocks like e.g. processing unit, clock network, internal memory and others. The power consumption of these blocks is described by parameter dependent arithmetic model functions. By application of a parser based automized analysis of assembler codes of the systems to be estimated the input parameters of the arithmetic functions like e.g. the achieved degree of parallelism or the kind and number of memory accesses can be computed. This approach is exemplarily demonstrated and evaluated applying two modern digital signal processors and a variety of basic algorithms of digital signal processing. The resulting estimation values for the inspected algorithms are compared to physically measured values. A resulting maximum estimation error of 3% is achieved.

1 Introduction

In the course of increasing complexity of digital signal processing applications, especially in the field of mobile applications, low power techniques are of crucial importance. Therefore, it is desirable to estimate the power consumption of a system at a very early stage in the design flow. By this means it is possible to predict whether a system will meet a certain power budget before it is physically implemented. Necessary changes in the system partitioning or the underlying architecture will then be much less time and money consuming, because no physical implementation of the system is required to determine its power dissipation.

Another important design criteria of modern electronic systems is the demand for flexibility, e.g. the ability to adapt a system to changing specifications or standards. This fact along with the continuous growth of their computational power makes programmable digital signal processor (DSP)-kernels a very attractive component for heterogeneous Systems-on-Chip.

Like any other architecture block the power consumption of a DSP (-kernel) depends on several factors like the switching activity of the input data, the clock frequency and of course the executed algorithm itself. Besides these dependencies there are many more DSP-specific influencing factors like the type and rate of memory accesses, the usage of specific architecture elements like DMA controllers or dedicated co-processors, different compiler optimization settings, pipeline stalls and cache misses but also different

Correspondence to: H. Blume
(blume@eecs.rwth-aachen.de)
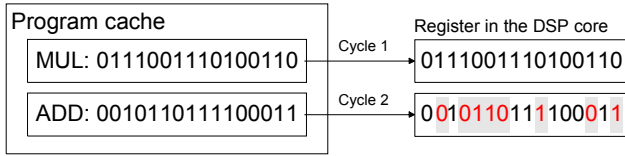
Fig. 1. Sequential execution of two different DSP instructions.
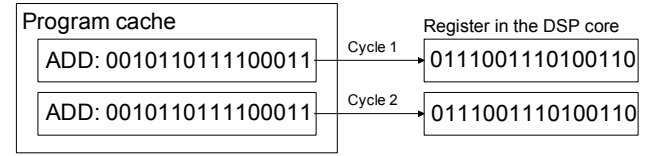
Fig. 2. Sequential execution of two identical DSP instructions.

programming styles or the choice of algorithmic alternatives which all strongly influence the power consumption of an algorithm that is executed on a DSP.

For this reason it is desirable to consider methodologies for power estimation that cover all significant influencing factors and provide a sufficient accuracy at moderate complexity. Such a methodology is presented in this paper and verified using several exemplary vehicles. The paper is organized as follows: Sect. 2 shortly reviews and discusses several existing power estimation techniques in terms of their portability to modern DSP architectures. The following section describes the so-called Functional-Level Power Analysis (FLPA) approach in detail. Section 4 lists some results concerning the application of the FLPA methodology for estimating the power of a variety of basic algorithms. A conclusion of the paper is given in Sect. 5.

2 Classical approaches for power estimation

One possible straight forward power estimation approach on DSPs is the so-called Physical-Level Power Analysis methodology. This approach is based on the analysis of the switching activity of all transistors of the DSP architecture. The requirement of this methodology is the availability of a description of the processor architecture on the transistor level, which is rarely given for modern DSPs. But the main disadvantage is the extremely high computational effort that makes approaches like this inapplicable for digital signal processors. Architectural-Level approaches like (Brooks et al., 2000) reduce this computational effort by modelling typical architecture elements like registers, functional units or load/store queues. These models are not based on physical measurements and require still exact knowledge of the processors architecture. Therefore, these two methodologies can be mainly found in the field of microprocessor development.

Another possibility for power estimation for DSPs is the so-called Instruction-Level Power Analysis (Tiwari et al., 1996). By means of physical measurements or low level simulations the energy consumption of each instruction out of the instruction set of a given processor is determined. By analysis of the assembler code of a program it is then possible to estimate the specific power consumption for this program performed on a certain processor. The advantage of this approach is the ability to cover a specific part of power consumption of DSPs: the so-called inter-instruction effects. In general, the energy consumption of a DSP instruction de-

pends on the previously executed instructions, what can be explained by means of Figs. 1 and 2.

At a certain stage of a processors pipeline, instruction words are transferred from the program cache into a register in the DSP core for further processing. Figure 1 shows the situation that an ADD (addition) instruction word replaces a MUL (multiplication) instruction word in cycle 2. The numbers shaded with gray boxes show the bits in the register that switch their state in this case. In this example a Hamming distance (number of different bits of these two instruction words) of eight ($H_d=8$) is resulting. As can be seen in Fig. 2 the sequence of two identical instructions causes no switching activity ($H_d=0$). Effects like this occur in many stages of a processors pipeline and as a result of these effects the energy consumption of a DSP instruction obviously depends on the previously executed instruction (Marwedel, 2003). The Instruction-Level Power Analysis methodology allows to cover such inter-instruction effects by measuring the energy consumption of groups of DSP instructions, but that makes this approach very complex due to the huge number of possible combinations. The effort will even grow, if Very-Long-Instruction-Word (VLIW) architectures shall be modeled due to their increasing word length and their ability to issue several operations in parallel.

A more attractive approach for power estimation is the Functional-Level Power Analysis (FLPA) methodology. This methodology has been introduced in (Qu et al., 2000) and was first applied in (Senn et al., 2002) to a digital signal processor. Here, a refined extension of this methodology is presented in order to model complete DSP cores including the modeling of separate units like cache, internal RAM, EDMA and integrated co-processors, different types of memory accesses etc. The following section will demonstrate this methodology applying an exemplary vehicle – the TMS320C6416 DSP.

3 Functional-Level Power Analysis (FLPA)

The basic principle of the FLPA methodology is depicted in Fig. 3.

In a first step the DSP architecture is divided into functional blocks like fetch unit, processing unit, internal memory and others like the clocking system. By means of measurements it is possible to find an arithmetic function for each block that determines its power consumption in dependency of certain parameters. These parameters are for example the clock frequency, the degree of parallelism or the rate with
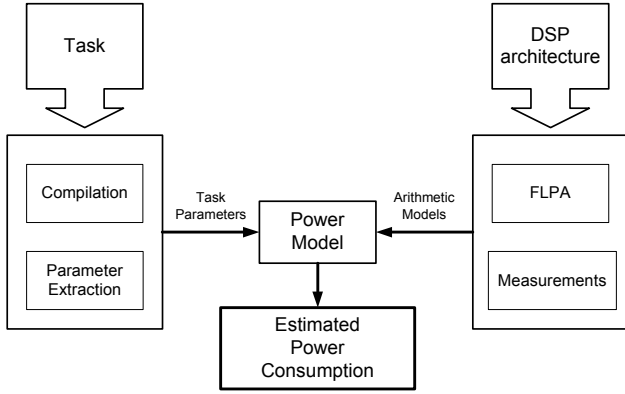
Task

Compilation

Parameter Extraction

Task Parameters

Power Model

Estimated Power Consumption

DSP architecture

Arithmetic Models

FLPA

Measurements

Fig. 3. The basic FLPA principle.

Interfaces, Peripherals

EDMA

Level 2 SRAM, Level 2 Cache

Level 1 Program Cache

Fetch VLIW instruction word
Dispatch atomic instructions

Datapath A

L1 S1 M1 D1

Register bank A
A31 - A0

Datapath B

D2 S2 M2 L2

Register bank B
B31 - B0

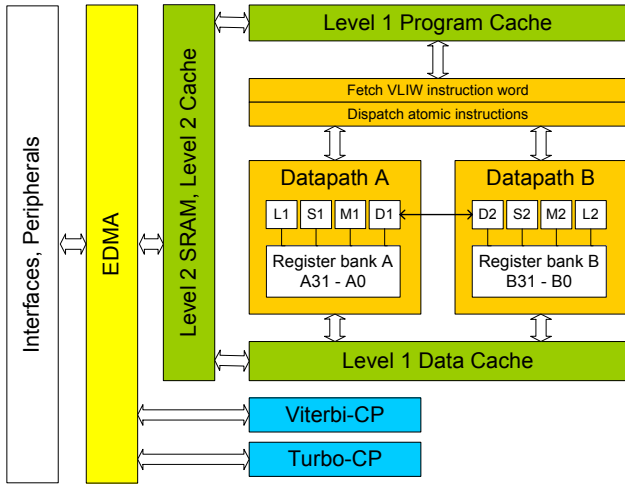Level 1 Data Cache

Viterbi-CP

Turbo-CP

Fig. 4. The TMS320C6416 architecture.

which the internal memory is accessed. Most of these parameters can be automatically determined by a parser which analyzes the assembler file of a program code. The total power consumption is then given as the sum of the power consumption of each functional block:

$$P_{\text{total}} = \sum_i P_{\text{block } i}. \tag{1}$$

The left side of Fig. 3 depicts the process of extracting parameters from a program which implements a task. After compilation it is possible to extract the task parameters from the assembler code. Further parameters can be derived from a single execution of the program (e.g. the number of required clock cycles). These parameters are the input values for the previously determined arithmetic model functions. Thus, an estimation for the algorithms power consumption can be computed. This approach is applicable to all kinds of processor architectures. Further on, FLPA can be applied to a processor with moderate effort and no detailed knowledge of the processors architecture is necessary.
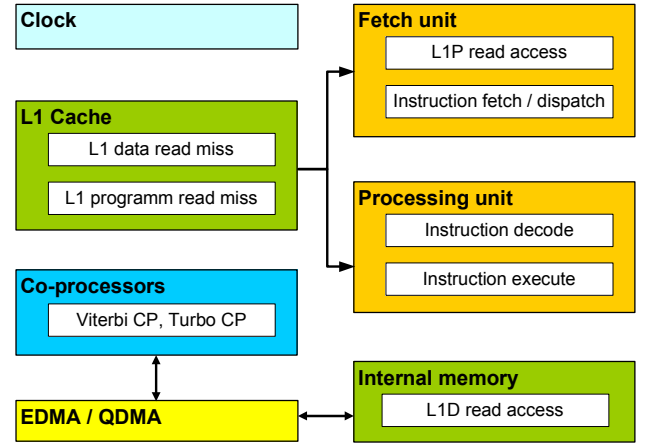
Clock

L1 Cache

L1 data read miss

L1 programm read miss

Co-processors

Viterbi CP, Turbo CP

EDMA / QDMA

Fetch unit

L1P read access

Instruction fetch / dispatch

Processing unit

Instruction decode

Instruction execute

Internal memory

L1D read access

Fig. 5. Separation of the TMS320C6416 architecture into functional blocks.

worst-case
complex ops (mul)
32 bit data

typical-case
16 bit data

best-case
simple ops (add)
8 bit data

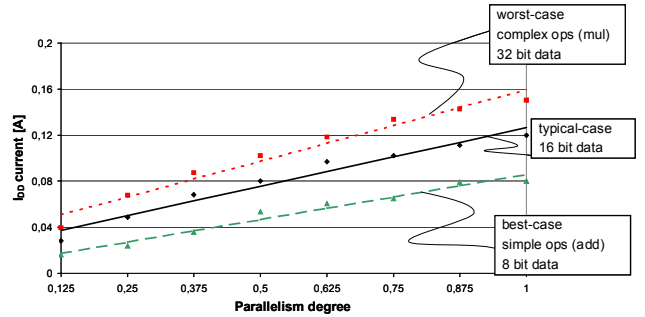I_{DD} current [A]

Parallelism degree

Fig. 6. Model function of the TMS320C6416 processing unit.

3.1 An exemplary vehicle: The TMS320C6416 DSP

The TMS320C6416 is a state-of-the-art VLIW DSP aiming for multimedia applications. Figure 4 depicts a block diagram of the DSP architecture.

It is based on a VLIW-architecture with two parallel data paths each including four issue-slots. Furthermore, this processor includes a couple of interfaces (ATM, PCI, etc.), an Enhanced DMA-controller (EDMA) and two dedicated co-processors (Viterbi and Turbo decoder co-processor). For this work the integrated software development environment Code Composer Studio (CCS) and the hardware test and evaluation board (TEB) including the C6416 have been utilized. For further details of this architecture see (TMS320C6416, SPRS164C documentation set).

This architecture can be divided into seven functional blocks as depicted in Fig. 5.

Arithmetic model functions describing the power consumption of a functional block can be found by means of measurements. Therefore, it is necessary to stimulate each block separately. This can be achieved by executing different parts of assembler code, that will be called scenarios according to (Senn et al., 2002).

A determination of a model function applying such scenarios will be described here considering the processing unit and the fetch unit as example.
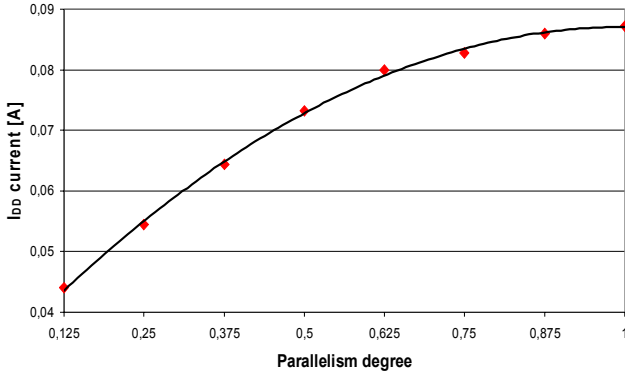
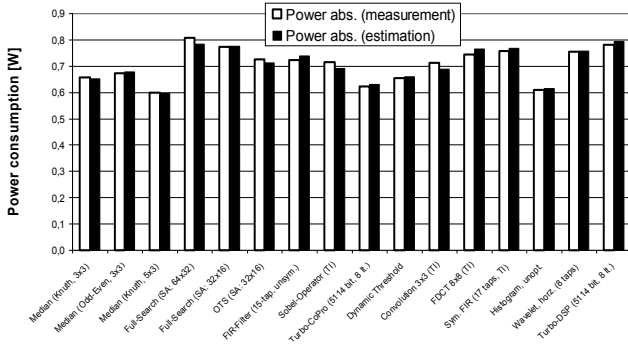**Fig. 7.** Model function of the TMS320C6416 fetch unit.

Fig. 8. FLPA power estimation results and measurements for the TMS320C6416 (absolute power consumption).

The power consumption of the processing unit has three significant parameters:

– the degree of parallelism α (percentage of parallel working functional units),

– the number of executed instructions,

– the type of input data.

The scenarios belonging to the processing unit vary these parameters separately.

In Fig. 6 the current drawn by the processing unit is depicted over the degree of parallelism. The applied test scenario includes a loop where within each loop iteration 1000 instructions are executed. The dotted line of Fig. 6 represents the worst-case power consumption of the processing unit, in which complex instructions (e.g. multiplications) with the maximum word length of the input data (32 bit) are executed. In contrast to that, the dashed line represents the best-case power consumption with simple instructions (e.g. additions) and a small word length of the input data (8 bit). The arithmetic function belonging to the straight line (typical-case: instruction mix, medium word length of the input data (16 bit)) is chosen as model function for the FLPA model of the TMS320C6416 processing unit and modeled by

$$P_{\text{processing unit}} = (1.02 \cdot 10^{-1} \cdot \alpha + 2.46 \cdot 10^{-2}) \cdot V_{\text{core}}$$
$$= I_{DD,\text{ processing unit}} \cdot V_{\text{core}}. \tag{2}$$

Table 1. Model functions of the functional blocks of the TMS320C6416 and belonging list of parameters.

functional block	block specific power consumption function
clock system	$P_{\text{clock system}} = (a \cdot F + b) \cdot V_{\text{Core}}$
fetch unit	$P_{\text{fetch unit}} = (c \cdot \alpha^2 + d \cdot \alpha + e) \cdot F \cdot (1\text{-}PSR) \cdot V_{\text{Core}}$
processing unit	$P_{\text{proc. unit}} = (f \cdot \alpha + g) \cdot F \cdot (1\text{-}PSR) \cdot V_{\text{Core}}$
internal memory	$P_{\text{internal memory}} = (h \cdot \beta + i \cdot \gamma) \cdot F \cdot (1\text{-}PSR) \cdot V_{\text{Core}}$
level-1 cache	$P_{\text{level-1 cache}} = (j \cdot \delta + k \cdot \varepsilon) \cdot F \cdot (1\text{-}PSR) \cdot V_{\text{Core}}$
EDMA/QDMA	$P_{\text{EDMA/QDMA}} = (m \cdot \zeta) \cdot F \cdot (1\text{-}PSR) \cdot V_{\text{Core}}$
co-processors (Turbo, Viterbi)	$P_{\text{copro}} = (n \cdot \eta + p \cdot \theta) \cdot F \cdot (1\text{-}PSR) \cdot V_{\text{Core}}$

parameter	description	parameter	description
α	parallelism degree	θ	Turbo co-processor activity rate
β	memory access rate (read)	η	Viterbi co-processor activity rate
γ	memory access rate (write)	V_{Core}	Core Voltage of the processor
δ	L1P cache miss rate	F	clock frequency
ε	L1D cache miss rate	PSR	pipeline stall rate
ζ	EDMA activity rate	$a,b,c,d,e,f,$ $g,h,i,j,k,m,$ n,p	coefficients for polynomials

Here, V_{Core} denotes the core-voltage of the processing unit and α the achieved degree of parallelism. The error of the estimated power consumption for algorithms with either extremely complex or extremely simple instructions will be examined in the next section.

The architecture of the fetch unit of the TMS320C6416 has the task to control the flow of VLIW instruction words to the DSP core and to dispatch the atomic instructions to the functional units. Though the architecture of the fetch unit of the TMS320C6416 is not known in detail it is possible to model this functional block. In a test scenario the only parameter having a strong impact on the power consumption of the fetch unit, the parallelism degree α, is varied and some working points are measured. Figure 7 depicts the current consumption drawn by the fetch unit.

According to the measured working points a polynomial function (here, a quadratic function) can be found which describes the power consumption of the fetch unit

$$P_{\text{fetch unit}} =$$
$$(-5.67 \cdot 10^{-2} \cdot \alpha^2 + 1.14 \cdot 10^{-1} \cdot \alpha + 3.02 \cdot 10^{-2}) \cdot V_{\text{core}}$$
$$= I_{DD,\text{ fetch unit}} \cdot V_{\text{core}}. \tag{3}$$

All the other FLPA blocks depicted in Fig. 5 can be modeled similarly. The complete FLPA power model of the TMS320C6416 including the complete list of required parameters is shown in Table 1.

4   Results

For the evaluation of the FLPA methodology the power consumption was measured as well as estimated for a variety of digital signal processing algorithms. The comparison of estimated and measured values shows a maximum error of 3%, as can be seen in Fig. 8. All algorithms which are marked
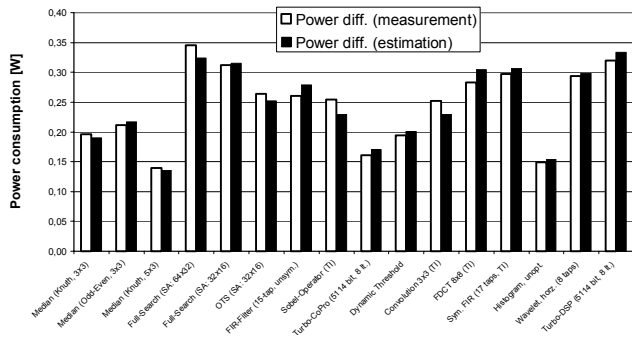
Fig. 9. FLPA power estimation results and measurements for the TMS320C6416 (differential power consumption).

Fig. 10. FLPA power estimation results and measurements for the TMS320C6711 (differential power consumption).

with (TI) have been taken from the TI code library in order to apply the methodology also for DSP code which was optimized by the processor manufacturer himself.

Obviously, the part of the total power consumption according to the clock system is a constant offset for each algorithm which is performed on the processor. Therefore, for a fair comparison differential power consumption values (without the clock system) should also be regarded. The comparison depicted in Fig. 9 yields a maximum error of 10%. It should be noticed that according to the program to be performed on the processor the differential power consumption varies by more than 200 mW. This dynamics is much larger than the maximum estimation error of about ten to twenty mW.

The FLPA approach has also been applied to the C6711 processor which is a floating point processor providing no further co-processors. The C6711 FLPA model comprises seven model functions. Compared to the set of algorithms which have been taken as benchmarking set for the C6416 the benchmarking set for the C6711 also included dedicated floating point applications like floating point matrix multiplications. The maximum power consumption of the C6711 within the experiments amounted to 1.1 W and the dynamics concerning the power consumption of the different algorithms amounted to 350 mW. A comparison between the FLPA power estimation and physical measurements yields a maximum error of less than 5% for the absolute power consumption and less than 10% (40 mW) for the differential power consumption (see Fig. 10). Again this comparison proves that the FLPA methodology provides sufficient accuracy for a power estimation in an early stage of the design flow.

5 Conclusion

Different approaches for power estimation for programmable processors have been described and evaluated concerning their capability to be applied to modern digital signal processor (DSP) architectures like e.g. Very Long Instruction Word (VLIW)-architectures. The concept of so-called Functional-Level Power Analysis (FLPA) has been extended and refined and the belonging separation of the processor architecture
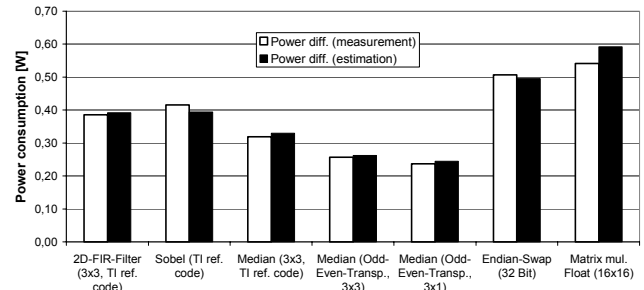
into functional blocks has been shown. The power consumption of these blocks has been described in terms of parameterized arithmetic model functions. A parser which allows to analyze automatically the assembler codes has been implemented. This parser yields the input parameters of the arithmetic functions like e.g. the achieved degree of parallelism or the kind and number of memory accesses. A demonstration and evaluation of this approach has been performed applying the DSPs TMS320C6416 and TMS320C6711 and a variety of basic algorithms of digital signal processing. Resulting estimation values for the inspected algorithms are compared to measured values. A resulting maximum estimation error of 3% for the absolute power consumption and 10% for the differential power consumption is achieved. The application of this methodology allows to evaluate efficiently different parameter settings of a programmable processor like different coding styles, compiler settings, algorithmic alternatives etc. concerning the resulting power consumption. Therefore, it is a valuable methodology for a system designer to explore the design space of programmable processors concerning the power aspect.

References

Brooks, D., Tiwari, V., and Martonosi, M.: Wattch: A Framework for Architectural-Level Power Analysis and Optimizations, Proceedings of the ISCA, 83–94, 2000.

Marwedel, P.: Fast, predictable and low-energy memory references through memory-architecture aware compilation, Proceedings of the DSP Design Workshop 2003, Dresden, 2003.

Qu, G., Kawabe, N., Usami, K., and Potkonjak, M.: Function Level Power Estimation Methodology for Microprocessors, Proc. of the Design Automation Conference 2000, 810–813, 2000.

Senn, E., Julien, N., Laurent, J., and Martin, E.: Power Consumption Estimation of a C Program for Data-Intensive Applications, Proc. of the PATMOS Conference 2002, 332–341, 2000.

Tiwari, V., Malik, S., and Wolfe, A.: Instruction Level Power Analysis and Optimization of Software, Journal of VLSI Signal Processing, 1–18, 1996.

TMS320C6416: Fixed Point Digital Signal Processor, Texas Instruments, SPRS164C, 2001.

TMS320C6711: datasheets, http://www.ti.com.