Power Estimation Methodology for VLIW Digital Signal Processors

Mostafa E. A. Ibrahim[(1+2)], Markus Rupp[(1)], and Hossam A. H. Fahmy[(2)]
[(1)]Institute of Communications and RF Engineering-Vienna University of Technology, Austria
[(2)] Electronics and Communication Department Faculty of Engineering-Cairo University, Egypt
Email: {mhalas,mrupp}@nt.tuwien.ac.at, hfahmy@arithmetic.stanford.edu

***Abstract*— In this contribution the modeling of power consumption for the VLIW processor TMS320C6416T is presented taking into account typical software algorithms in signal and image processing. The modeling is performed at the functional level making this approach distinctly different from other modeling approaches in low level technique. This means that the power consumption can be identified at an early stage in the design process, enabling the designer to explore different hardware architectures and software algorithms. Some typical signal and image processing algorithms are used for the purpose of validating the proposed model. The estimated power consumption is compared to the physically measured power consumption, achieving a very low resulting average estimation error of** 1.05% **and a maximum estimation error of only** 3.3%

I. Introduction

Many applications in special areas such as hand-held computation, tiny robots, and guidance systems in automated vehicles are powered by batteries of low rating. In order to avoid frequent recharging or replacement of the batteries, there is significant interest in low-power system design. Very Long Instruction Word (VLIW) Digital Signal Processors (DSP) are the most worthy choice for such an application domain because of their optimal performance at low power [1].

Research efforts have focused primarily on optimizing speed to realize computationally intensive real-time tasks such as video compression and speech recognition. As a result, many systems have successfully integrated various complex signal processing modules to meet users computation and entertainment demands. While these solutions have provided answers to the real-time problem, they have not addressed the rapidly increasing demand for portable operation. The strict limitation on power dissipation which portability imposes, must be met by the designer while still meeting ever higher computational requirements. This has led to a significant research effort in power estimation and low power design [2], [3].

The existing Integrated Development Environments (IDE) are focusing on the speed and code size optimization. Which by no means optimize the power consumption. In the past decade, many researchers did put more efforts trying to introduce efficient power consumption estimation methodologies as a first step toward power dedicated compiler.

In this paper, an approach for modeling the power consumption of a VLIW DSP, from the software point of view, is presented. The contribution of this work aims to precisely estimate the power consumption of the core processor while running a software algorithm at an early stage in the design process. The targeted DSP is the TMS320C6416T (for the rest of the paper it is referred to as C6416T for brevity) from Texas Instrument. This processor features the highest-performance among the fixed-point DSPs of the C6000 DSP platforms.

The rest of the paper is organised as follows: Section II presents an overview of several existing power consumption modeling techniques for general purpose processors. A general overview of the target architecture is presented in Section III. It is followed by a detailed description of the functional level analysis for the targeted architecture in Section IV. The proposed power consumption model is verified in Section V and the reliability of the estimation is demonstrated. Finally, Section VI summarizes the main contributions of this paper.

This work has been funded by the Christian Doppler Laboratory for Design Methodology of Signal Processing Algorithms.

II. Related work

Recent approaches to model the power consumption of DSPs can be separated into two main categories: hardware level models and instruction level models [4]. Hardware level models calculate power and energy from detailed electrical descriptions, comprising circuit level, gate level, register transfer (RT) level [5], [6] or micro-architectural level simulation [7], [8]. While providing excellent accuracy; these methodologies are slow and impractical for analyzing the power consumption at an early design stage. Moreover, these methodologies require the availability of lower level circuit details or a complete hardware description language (HDL) design of the targeted processor, which is not available for most of commercial off-the-shelf processors [8]. Instruction level models can be classified into Instruction Level Power Analysis (ILPA) and Functional Level Power Analysis (FLPA).

A. Instruction Level Power Analysis

An instruction level power model for individual processors was first proposed by V. Tiwari [9]. Power is modeled as a base cost for each instruction plus a circuit state overhead that depends on neighboring instructions. The base cost of an instruction can be considered as the cost associated with the basic processing needed to execute the instruction. An experimental method is proposed by the authors of [9] to empirically determine the base and the circuit overhead costs. In this experimental method, a program containing an infinite

loop consisting of several instances of the given instruction is used. The average current drawn by the processor core during the execution of this loop is measured by a standard off-the-shelf, dual-slope integrating digital multimeter.

Much more accurate measuring environments have been proposed to precisely monitor the instantaneous current drawn by the processor instead of the average current [10]. Another approach, to reduce the spatial complexity of instruction-level power models, is presented in [11].

The ILPA based methods exhibit usually a small margin of error, typically 2 to 4 percent. However, these methods have some drawbacks. One of these drawbacks is that the number of current measurements is directly related to the number of instructions in the instruction set architecture (ISA), and also the number of parallel instructions composing the very long instruction in the VLIW processor [12]. Also they do not provide any insight on the instantaneous causes of power consumption within the processor core, which is seen as a black-box model.

B. Function Level Power Analysis

FLPA was first introduced by J. Laurent et al. in [13]. The basic idea behind the FLPA is the distinction of the processor architecture into functional blocks like processing unit (PU), instruction management unit (IMU), internal memory and others [13]. Figure 1 presents the main steps for FLPA, firstly, a functional analysis of these blocks is performed to specify and then discard the non-consuming blocks (those with negligible impact on the power consumption). The second step is to figure out the parameters that affect the power consumption of each of the power consuming blocks. For instance, the IMU is affected by the instructions dispatching rate which in turn is related to the parallelism degree. In addition to these parameters, there are some parameters that affect the power consumption of all functional blocks in the same manner such as operating frequency and word length of input data [14].

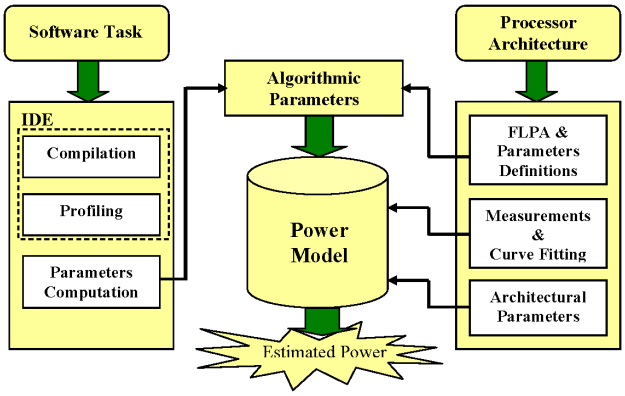

Fig. 1. Basic FLPA Concept.

By means of simulations or measurements it is possible to find an arithmetic function for each block that determines its power consumption depending on a set of parameters. For the determination of these arithmetic functions for each functional block, the average supply current of the processor core is measured in relation with the variation of each parameter. These variations are achieved by a set of small programs, called scenarios. Such scenarios are short programs written in assembly language and consisting of unbounded loops with a body of several hundreds of certain instructions that individually invoke each block. The power consumption rules are finally obtained by curve-fitting the measurements values [14].

The parameters that affect the power consumption for each functional block can be extracted from the assembly code generated by the integrated development environment (IDE). Some parameters cannot be extracted directly from the assembly code, such as the execution time and the data cache miss rate. Therefore, at least one simulation is required to obtain these parameters with the aid of the profiler.

The functional level power modeling approach is applicable to all types of processor architectures. Furthermore, FLPA-modeling can be applied to a processor with moderate effort and no detailed knowledge of the processors architecture is necessary [15].

III. Target Architecture

The $C6416T$ CPU contains a program fetch unit, an instruction dispatch unit, an instruction decode unit, two data paths each of four functional units, as well as 64 32-bit registers. The program fetch, instruction dispatch, and instruction decode units can deliver up to eight 32-bit instructions to the functional units every CPU clock cycle. The processing of the instructions occurs in each of the two data paths (A and B). The CPU also has a 32-bit, byte-addressable address space. However, the internal L2 memory is unified for data and program, the L1 memory is organized into separate data and program caches. The block digaram as well as much more details about the $C6416T$ DSP can be found in [16].

This DSP is considered as a complex processor architecture since it features a deep pipeline (11 stages) and can execute up to eight parallel instructions per cycle.

IV. Modeling Methodology for C6416T

After applying the FLPA, the C6416T architecture is subdivided into six distinct functional blocks (clock tree, instruction management unit, processing unit, internal memory, L1 data cache and L1 program cache) as shown in Fig. 2 [17]. The parameters that affect the power consumption for the determined functional blocks are also shown in Fig. 2. The C6416T fetches instructions from memory in fixed bundles of 8 instructions, known as fetch packets. The instructions are decoded and separated into bundles of parallel-issue instructions known as execute packets.

The dispatching rate α represents the average number of execution packets per fetch packet. The processing rate β stands for the average number of active processing units per cycle. The internal memory read/write access rates express the number of memory accesses divided by the number of required clock cycles for executing the code segment under investigation. Finally the data cache miss rate λ corresponds to

the number of data cache misses divided by the total memory accesses.

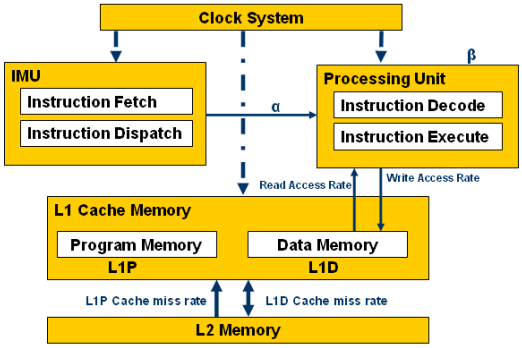

Fig. 2. Functional level power analysis for C6416T.

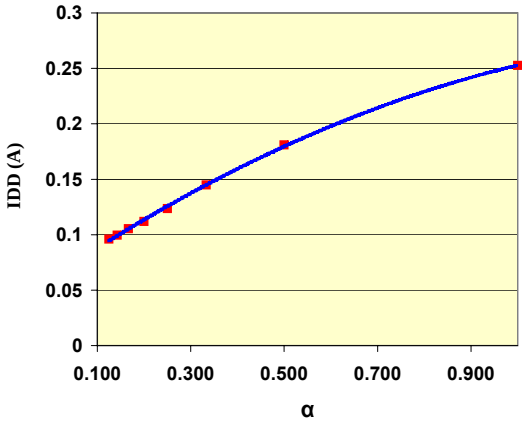

Fig. 3. Model function of the C6416T IMU.

A. Experimental setup

In our setup, the processor core voltage is 1.2 V and the operating frequency ranges from 600 MHz to 1200 MHz. The arithmetic functions in the following sections IV-B, IV-C, IV-D, describe the current drawn by the core processor at operating frequency of 1000 MHz. All measurements are carried out on the DSP Starter Kit (DSK) of the C6416T manufactured by Spectrum Digital Inc. The code composer studio (CCS3.1) from Texas Instruments is used as the IDE.

Several assembly language scenarios have been developed to separately stimulate each of the functional blocks. All scenarios consist of unbounded loops with a body of more than 1000 instructions, to avoid the effect of branching instructions on the measured current and to guarantee accurate measurement due to the long averaging period. First of all, the effect of the clock tree on the power consumption is determined. The operating frequency linearly affects the current drawn by the core processor and hence, also linearly affects the power consumption of the processor.

For demonstration purposes the process of determining the power consumption rules for IMU, PU and L1 data cache functional blocks is presented next.

B. IMU power consumption model

The IMU of the C6416T processor fetches eight instructions per cycle as one fetch packet. The dispatch unit then subdivides this fetch packet into execution packets. Since the C6416T has eight functional units, it is capable of simultaneously executing up to eight instructions. Consequently, the dispatch unit can divide the fetch packet into one (maximum parallelism) to eight (sequential) execution packets. Therefore, it is obvious that the dispatch rate is the only parameter that affects the power consumption of the IMU.

The proposed scenario to invoke the IMU is composed of an unbounded loop with more than 1000 NOPs. Since the NOP instruction does not require any processing unit for its execution. The scenario varies the dispatch rate (number of fetch packets divided by the number of execution packets) from 0.125 to 1.0.

Figure 3 indicates the characteristics of the current drawn by the core processor with a varying dispatch rate. By curve fitting the measurement values in Fig. 3 the arithmetical function in (1) is obtained.

$$IDD_{\mathrm{IMU}} = -0.0918\alpha^2 + 0.284\alpha + 0.0603, \qquad (1)$$

The quality of the fitting process is measured by the value R-squared (R^2): A number from 0 to 1, which is the square of the residuals of the data after the fit. This value expresses what fraction of the variance of the data is explained by the fitted trend line. It reveals how closely the estimated values for the trend line correspond to the actual data. A trend line is most reliable when its R^2 value is at or close to 1.0 [18]. Since the R^2 value for the arithmetic function in (1) equals 0.9994 that means (1) is an excellent fit for the curve values in Fig. 3.

The arithmetic function in (1) does not consider the effect of pipeline stalls. Many reasons cause the pipeline to stall. For instance, one data cache miss stalls the pipeline for at least 6 cycles. Hence, the arithmetic function in (2) is presented to account for the pipeline stall effect.

$$IDD_{\mathrm{IMU}} = (-0.0918\alpha^2 + 0.284\alpha + 0.0603)(1 - PSR), \qquad (2)$$

where PSR stands for Pipeline Stall Rate which can be expressed as the number of pipeline stall cycles divided by the total cycles required for executing the code segment under investigation.

C. PU power consumption model

The data path of the C6416T consists of eight functional units. These functional units can work simultaneously, if the dispatch unit succeeds to compose an execution packet with eight instructions. Unlike the model in [15] that uses the parallelism degree as the affecting parameter for the processing unit model, the fact that the NOP does not require any PU for its execution convinced us that another parameter yields a better description of the PUs. The new parameter is the processing unit rate which expresses the average number of active processing units per cycle. Figure 4 ilustrates the difference between the dispatch rate and the processing unit rate. Another important parameter that affects the processing unit power consumption is the word length of the data operands.

In the C6416T the word length varies from 8 bits to 32 bits. Thus, in our model 16 bit word length has been chosen to be the typical word length.

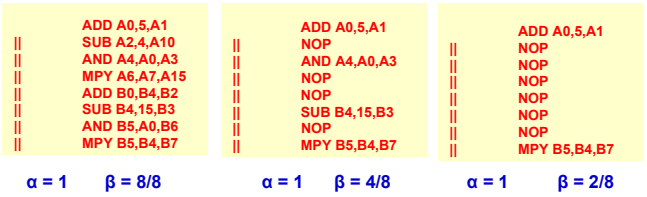

Fig. 4. Difference between β and α.

More than 1000 different instructions compose the scenario that varies the processing unit rate. That is to account for the inter-instructions effect. The current measured from the DSK is the sum of the clock tree, IMU, and the PU currents. To attain only the current drawn by the PU, the IMU and clock tree currents are subtracted from the measured current.

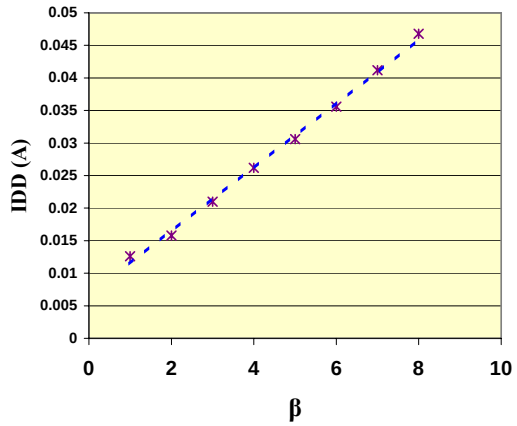

Fig. 5. Model function of the C6416T PU.

Figure 5 depicts the effect of varying the number of active PU per cycle on the current drawn by the core processor. The arithmetic function that best fits the curve in Fig. 5 is a linear equation as shown in (3)

$$IDD_{\mathrm{PU}} = (-0.0049\beta + 0.0065)(1 - PSR), \qquad (3)$$

The arithmetic function in (3) results in an excellent R^2 value of 0.9982, fit for the curve values in Fig. 5.

Compared to other functional units such as clock tree or the IMU, it is clear that the PU does not significantly contribute to the total power consumption of the core processor. It is important to mention that the scenario for invoking the PU does not include any memory instructions. The internal memory operations are handled in a separate scenario.

D. L1 data cache power consumption model

The L1 data cache functional block represents the flow of data from the L1 data cache to L2 memory and vice versa. Different scenarios are prepared to stimulate the effect of the data cache miss.

The data cache miss rate is used as the affecting parameter for the L1 data cache functional block. Taking into account the fact that L1 data cache is a two-way associative cache, a scenario that varies the number of data cache misses over a fixed number of memory accesses has been developed. In this scenario, arbitrary data are pre-loaded into both blocks of set 0. To force a data cache miss, data from certain addresses in the L2 memory, which must be mapped into set 0 blocks, are loaded to the L1 data cache. The addresses of the new data to be loaded are different from those already in set 0. Hence, a data cache miss occurs.

Figure 6 shows the effect of varying the data cache miss rate on the current drawn by the core processor. The best arithmetic function that fit the measured values in Fig. 6 is obtained as indicated in (4) with R^2 value of 0.9909.

$$IDD_{\mathrm{L1D}} = (-2 \cdot 10^{-5}\lambda^2 + 0.0041\lambda)(1 - PSR), \qquad (4)$$

where λ is the L1 data cache miss rate.

The arithmetic function in (4) is a quadratic-polynomial. This arithmetic function differs from the corresponding linear function that was proposed in [15] for the cache functional block. The squared-function yields better description for the L1 data cache block due to the fact that L1 data cache pipelines the cache misses, to decrease the resulting pipeline stalls. The proposed model in [14] did not separately investigate the effect of data cache misses instead it is included in the processing unit functional block.

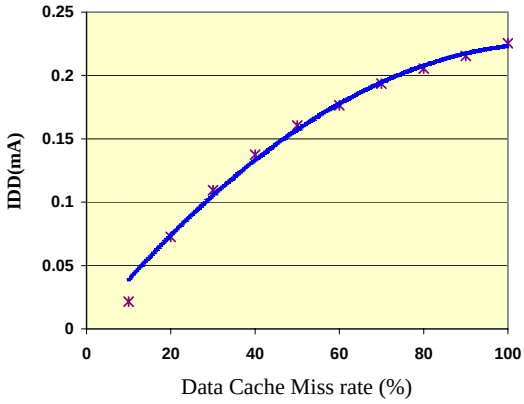

Fig. 6. Model function of the C6416T L1 data cache.

V. Validation

For purpose of validating the proposed power consumption model of the VLIW processors some common signal and image processing benchmarks from Texas Instruments libraries are presented. The input data for all investigated benchmarks are located in the internal data memory.

First of all, all optimization options which are included in the CCS3.1 are turned off because these optimization options affect the speed or the code size only and are not dedicated to power optimization. The second step is to compile the benchmarks. From the generated assembly files the required parameters for the model are calculated with the aid of the CCS3.1 profiler for the parameters that cannot be estimated statically such as the data cache miss rate. For instance, the processing unit rate which is defined as the average number of active processing units per cycle is calculated from the

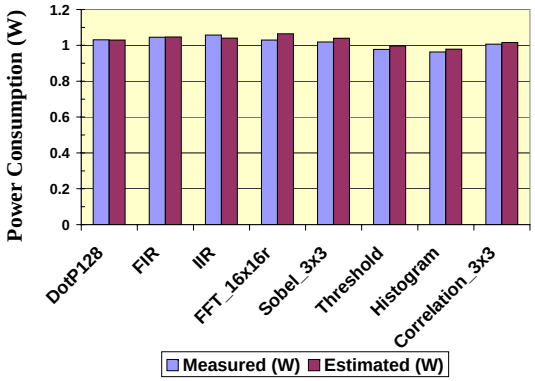

Fig. 7. Estimated vs. Measured power consumption of the C6416.

assembly code. The parameter β is the result of dividing the number of processing units (equals the number of instructions excluding the NOP) by the number of cycles per code iteration.

Figure 7 presents the result of the estimated power consumption versus the measured one for the above mentioned benchmarks. The average estimation error is 1.05% and in the worst case is 3.3%.

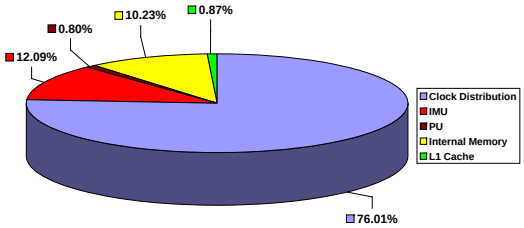

Fig. 8. Average functional units contribution to the processor core power consumption.

Figure 8 illustrates the percentage of the contributions of the different functional blocks of the processor to the power consumption. It is clear that the clock distribution is the largest contributor while the processing unit is the smallest contributor. The clock distribution contribution percentage to the total power consumption is expected to be much lower when estimating the power consumption of much more complex algorithms. This for sure increases the opportunity for more power oriented optimization efforts.

VI. Conclusion

In the presented paper different power consumption approaches that are applied on various levels of abstraction have been recapitulated. A functional level power analysis technique has been applied to the commercial off-the-shelf VLIW processor C6416T. The processor architecture has been divided into several functional blocks. The parameters that affect the power consumption of each functional block have been determined. These parameters have been computed from the generated assembly code of the IDE. The inter-instructions as well as the pipeline stall effects have been investigated in our proposed model. The power consumption has been estimated for several signal and image processing benchmarks. The estimated power consumption is compared with the physically measured power consumption and a very low resulting average error of 1.05% is realized. A maximum estimation error of only 3.3% is achieved.

References

[1] N. Zafar Azeemi and M. Rupp. Multicriteria low energy source level optimization of embedded programs. In *Tagungsband zur Informationstagung Mikroelektronik 06 IEEE Austria*, pages 150–158, Vienna, Austria, October 2006.

[2] Anantha P. Chandrakasan, Miodrag Potkonjak, Renu Mehra, Jan M. Rabaey, and Robert W. Brodersen. Optimizing power using transformations. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 14(1):12–31, 1995.

[3] William Fornaciari, Paolo Gubian, Donatella Sciuto, and Cristina Silvano. Power estimation of embedded systems: A hardware/software codesign approach. *IEEE Trans. Very Large Scale Integr. Syst.*, 6(2):266–275, 1998.

[4] Chris J. Bleakley, Miguel Casas-Sanchez, and Jose Rizo-Morente. Software level power consumption models and power saving techniques for embedded dsp processors. *Journal of Low Power Electronics*, 2(2):281–290, 2006.

[5] T. Chou and K. Roy. Accurate estimation of power dissipation in CMOS sequential circuits. *IEEE Trans. Very Large Scale Integr. Syst.*, 4:369–380, September 1996.

[6] Charlie X. Huang, Bill Zhang, An-Chang Deng, and Burkhard Swirski. The design and implementation of PowerMill. In *ISLPED '95: Proceedings of the 1995 international symposium on Low power design*, pages 105–110, New York, NY, USA, 1995. ACM.

[7] W. Ye, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin. The design and use of SimplePower: a cycle-accurate energy estimation tool. In *DAC'2000: Proceedings of the 37th conference on Design automation*, pages 340–345, New York, NY, USA, 2000. ACM.

[8] David Brooks, Vivek Tiwari, and Margaret Martonosi. Wattch: a framework for architectural-level power analysis and optimizations. *SIGARCH Comput. Archit. News*, 28(2):83–94, 2000.

[9] Vivek Tiwari, Sharad Malik, and Andrew Wolfe. Power analysis of embedded software: a first step towards software power minimization. *IEEE/ACM International Conference on Computer-Aided Design,Digest of Technical Papers.*, pages 384–390, November 1994.

[10] S. Nikolaidis, N. Kavvadias, P. Neofotistos, K. Kosmatopoulos, T. Laopoulos, and L. Bisdounis. Instrumentation set-up for instruction level power modeling. In *PATMOS '02: Proceedings of the 12th International Workshop on Integrated Circuit Design. Power and Timing Modeling, Optimization and Simulation*, pages 71–80, London, UK, 2002. Springer-Verlag.

[11] B. Klass, D. E. Thomas, H. Schmit, and D. F. Nagle. Modeling inter-instruction energy effects in a digital signal processor. In *Power Driven Microarchitecture Workshop in conjunction with International Syymposism Computer Architecture*, June 1998.

[12] Mariagiovanna Sami, Donatella Sciuto, Cristina Silvano, and Vittorio Zaccaria. An instruction-level energy model for embedded VLIW architectures. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 21(9):998–1010, 2002.

[13] J. Laurent, E. Senn, N. Julien, and E. Martin. High level energy estimation for DSP systems. In *PATMOS'01: Proceedings International Workshop on Power And Timing Modeling and Optimization and Simulation*, pages 311–316, September 2001.

[14] Eric Senn, Nathalie Julien, Johann Laurent, and Eric Martin. Power consumption estimation of a C program for data-intensive applications. In *PATMOS'02: Proceedings of the 12th International Workshop on Integrated Circuit Design. Power and Timing Modeling, Optimization and Simulation*, pages 332–341, London, UK, 2002. Springer-Verlag.

[15] M. Schneider, H. Blume, and T. G. Noll. Power estimation on functional level for programmable processors. In *Advances in Radio Science*, volume 2, pages 215–219, May 2005.

[16] Texas Instruments. *TMS320C6416T, Fixed Point Digital Signal Processor, Datasheet*, November 2003. SPRS226J.

[17] Mostafa Ibrahim, Markus Rupp, and Serag Habib. Power consumption model at functional level for VLIW digital signal processor. In *Conference on Design and Architectures for Signal and Image Processing DASIP08*, November 2008.

[18] Norman R. Draper and Harry Smith. *Applied Regression Analysis*. Wiley Series in Probability and Mathematical Statistics. John Wiley and Sons, Inc., New York, NY, second edition, 1981.