A Power Estimation Method for ASIPs Considering Data Types of Variables in Application Programs

Tsutomu Kimura(*), Shin-ichi Shibahara(**), Yoshinori Takeuchi(**), Masaharu Imai(**), Akira Kitajima(**), and Michiaki Muraoka(**)(***)

(*) Dept. of Information and Computer Engineering,
Toyota College of Technology

2-1 Eisei-cho, Toyota,
Aichi 471-8525 Japan
Tel: +81-565-36-5869
Fax: +81-565-36-5926

(**) Dept. of Informatics and Mathematical Sciences Graduate School of Engineering Science,
Osaka University
1-3 Machikaneyama, Toyonaka,
Osaka 560-8531 Japan
Tel: +81-6-6850-6625
Fax: +81-6-6850-6627

(***) Corporate Semiconductor Development Division,
Matsushita Electric Industrial Co., Ltd.

3-1-1 Yagumo-nakamachi, Moriguchi,
Osaka 570-8501 Japan
Tel: +81-6-6906-4925
Fax: +81-6-6906-1458

E-mail: peasv@vlsilab.ics.es.osaka-u.ac.jp

Abstract

This paper proposes an efficient and accurate power estimation method for Application Specific Instruction set Processors (ASIPs). Proposed method takes advantage of the data types of variables in application program to be executed on the ASIP. According to the experimental results, the efficiency of proposed method was more than 1000 times as high as that of conventional RTL based power estimation method, and the estimation error was within 10% compared to a conventional gate-level accurate power estimation method.

Keywords: power estimation, ASIP, power macromodeling, data types, profiling

1. Introduction

In the SoC (System-on-a-Chip) design, architectural exploration should be performed repeatedly in order to find an optimal candidate of architecture. Then, such estimation method that can be applied in an early stage of design will make it possible for designers to explore a huge design space and find a better architecture in a shorter design time.

Accurate and efficient power estimation method is essential to design a less power consuming SoC, such as used in cellar phone and PDA (Personal Digital Assistant) driven by dry cell batteries. Power estimation of ASIPs is much more difficult than of ASICs because the power consumption of an ASIP does not simply depend on the sum of power consumption of internal components but also on the application programs and data items executed on the ASIP.

Many power consumption estimation methods have been proposed so far as shown in Refs. [1][2][3]. Conventional power estimation methods that can be applied to processor design can be classified into three types. Please note that gate level and transistor level simulation methods are not included in the following discussion because these methods require too long simulation time to estimate power consumption for architectural exploration.

(1) RTL (Register Transfer Level) simulation based method
(2) PSL (Pipeline Stage Level) simulation based method
(3) ISL (Instruction Set Level) simulation based method

The limitations of these conventional estimation methods are that they are time consuming and/or less accurate. While RTL simulation based method provides the most accurate estimation results compared to PSL and ISL simulation based methods, but it requires much longer simulation time than others. ISL simulation based method is the most efficient but less accurate than others.

Another limitation of PSL and ISL simulation based methods is that the power macromodels have to be modified when target architecture is changed, otherwise less accurate results will be obtained. In these conventional methods, however, considerable amount of effort is required to modify a power macromodel according to the architecture.

This paper proposes a new power estimation method for ASIPs that takes data types into accounts. Proposed method is efficient in terms of computation time as well as accurate in the estimation results.

The rest of this paper is organized as follows. In section 2, the power macromodeling defined in this paper is introduced. In section 3, the proposed method using power macromodeling in section 2 is explained. The experimental results are illustrated in section 4. In section 5, conclusion and future work are presented.

2. Power Macromodeling Based on Data Types of Variables

2.1 Data Types and Power Consumption

Power consumption of a component heavily depends on the number of bits whose value are changed. In other words, effective bit range will affects the power consumption. Therefore, data types of variables can be used as a very important information to estimate the power consumption. For example, in C language, there are some data types like character, short integer, integer, and so on. These data types are utilized in our proposed methods to estimate power consumption more precisely.

2.2 States of Component

In order to analyze the power consumption of a component, it is assumed that the component is one of the following three states: *standby*, *stop*, and *running* and the next state can be decided by the current state of the component and the input data.

(1) *Standby*: The state where the clock is the only active signal and other signals are the same as in the previous state, disabled state by the disable signal of the component, or the initial state.
(2) *Stop*: The state where all input signals do not change.
(3) *Running*: The state except for the standby and the sates.

2.3 Power Macromodeling

Every component used in ASIP must be modeled using Power Macromodel. The Power Macromodel in the proposed method consists of following elements.

(1) Power consumption $P_{comp}(C_{type}, D_{type})$ [W/MHz]: Power consumption of the component C_{type} when its input data type is D_{type}.
(2) Power consumption $P_{conn}(C_{type}, F_{out}, D_{type})$ [W/MHz]: Power consumption of the wires from component C_{type} when its input data-type is D_{type} and the number of fan-out is F_{out}.
(3) Delay time [ns]: Delay time of the component C_{type}.

2.4 Structure of Power Macromodel

Gate level simulation is required to estimate the wiring delay inside the component. Power consumption of the component C_{type} is measured by a gate level power consumption estimation tool. Power consumption data for all data type D_{type} must be measured in the proposed method.

The power consumption $P_{conn}(C_{type}, F_{out}, D_{type})$ of the connection wires is estimated by extracted wiring capacity and the transition density $D_{out}(i, C_{type}, D_{type})$, where i denotes the output bit number. Where transition density of wire number i is defined as the possibility of output bit transition probability for randomly generated input data. It is also assumed that the capacitance of wires is proportional to the fanout of the wire.

3. Estimation of Power Consumption

3.1 Outline

The computation flow of the proposed power consumption estimation method is shown in Figure 1. First, statistics on the execution counts of each basic block is measured by executing instruction level simulation. In the proposed method, power consumption is calculated for each basic blocks. Instruction level simulation requires the application program and its input data. Secondly, the power consumption of each instruction is computed using the HDL description of the processor, the operating information of each component of all instructions, and the clock frequency of the processor. Thirdly, the power consumption of each Basic Block is computed using instruction set power consumption. Finally, the power consumption when an application program is executed is estimated.

3.2 Assumptions

Following assumptions are made in the proposed method:

(1) A processor consist of components such as functional units, register files and the control unit.
(2) Application programs are described in the high level language where the data types of variables are explicitly declared.
(3) Each component in the processor is in one of the three states: *Standby*, *Stop*, or *Running*, for each instruction.
(4) Power consumption of the control unit is small enough compared to that of other components in a processor.

3.3 Input

The input to the proposed method is as follows:

(1) Application program described in the high level language. (C language was used in the experiment)
(2) Input data set for the application program.
(3) HDL description of the processor.

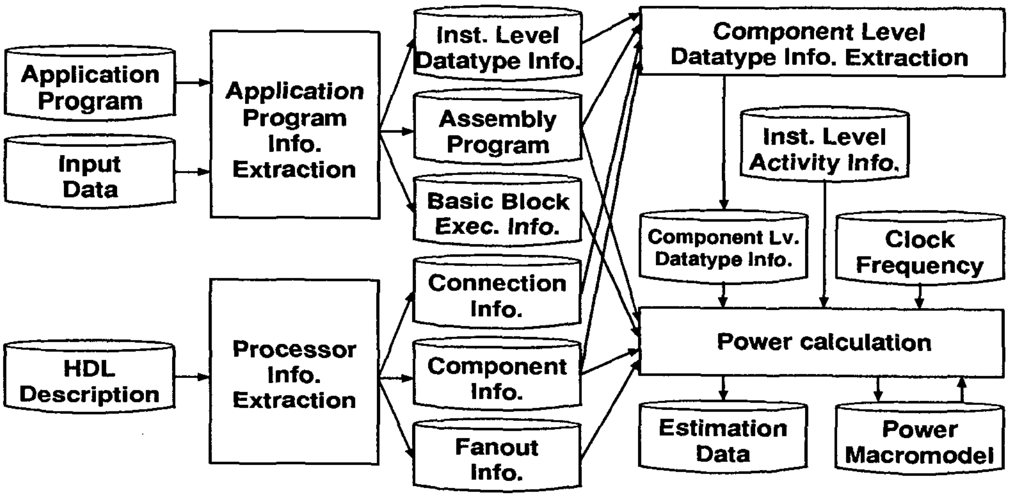

Figure 1. Estimation Flow

(4) The operating information of each component at every processor instructions.
(5) Clock frequency of the processor.

3.4 Estimation Procedure

The computation process to estimate power consumption is as follows.

Step 1: Analysis of Processor

Analyze the HDL description of the processor and extract the functional units and their connection information. Then calculate the fun-out of each component. Let $F_{out}(C_i)$ denote the fun-out of component *#i*.

Step 2: Analysis of Application Program

(1) Extract data types
Extract the data type associated to each variable in the program
(2) Extract Basic Blocks
Translate the given application program into the assembly program of the processor. Then, divide the assembly program into basic blocks.

Step 3: Instruction Level Profiling

Analyze the execution count of each basic block. The program is simulated at the instruction level using the assembly program and its input data. Let $N_{ex}(BBi)$ denote the execution count of basic block *#i*.

Step 4: Input Data Type Analysis

Decide the "Input data-type" for each component from the assembly instruction analysis. After the input data type $D_{type}(C_{type}(C_i),\ A_j)$ of the component is decided, operation information of each component of each processor instruction, the data type of the operand of the assembly instruction A_j, the connection information of each component C_i can be acquired.

Step 5: Power Estimation of Assembly Instruction

Compute the power consumption when each assembly instruction as follows. Let $P(A_j)$ denote the power consumption of the assembly instruction A_j. $P(A_j)$ can be calculated by the following expression. $P_{macro}(\ )$ is the power macromodel of component C_i. Where $C_{type}(C_i)$ denotes the type of the functional unit, $D_{type}(C_{type}(C_i), A_j)$ denotes the data type used in the assembly instruction A_j, and $Fout(C_i)$ denotes the funout of component *#i* used in A_j, and F_{proc} denotes the clock frequency.

$$P(A_j) = \sum_{\forall i} P_{macro}(C_{type}(C_i),\ D_{type}(C_{type}(C_i),\ A_j),\ Fout(C_i),\ F_{proc})$$

Step 6: Computation of the power consumption of *BB*

Power consumption $P(BB_k)$ of the Basic Block BB_k can be computed by the following equation. Where N_{inst} (BB_k) denotes the number of assembly instructions contained in basic block BB_k.

$$P(BB_k) = \frac{\sum_{\forall A_j \in BB_k} P(A_j)}{N_{inst}(BB_k)}$$

Step 7: Power Consumption of the Program

Let $P_{program}$ denote the power consumption of the whole program. $P_{program}$ can be calculated by the following equation, where *#BB* denote the number of basic blocks in the program.

$$P_{program} = \frac{\sum_{k=1}^{\#BB} P(BB_k) \times N_{ex}(BB_k)}{\sum_{k=1}^{\#BB} N_{ex}(BB_k)}$$

3.5 Comparison with Other Methods

The feature of the proposed method is as follows:

(1) Power consumption of each component is modeled and categorized by the data types of variables declared in the application program.
(2) Power consumption of each instruction is defined by the sum of power consumed by the components used by the instruction, which can be easily calculated using the power macromodels.
(3) Total power consumption can be estimated taking advantage of a profile of application programs, such as the number of basic blocks and execution counts of these basic blocks.

One of the advantages of the proposed method is the efficiency in terms of computation time because the profiling of application program can be performed at instruction set level. Another advantage of this method is the higher accuracy because the RTL simulation based method is utilized.

The disadvantage of the proposed method is that the power macromodeling of each component takes long time to construct. However, this effort is worth to invest because once the power macromodel of each component is constructed, it is reusable for other processors with different instruction sets. Thus, the proposed method is suitable for design space exploration in ASIP design.

4. Experiments

The effectiveness of the proposed power estimation method has been examined using several application programs and the processors which were designed to have specific instruction set and the specific architecture suitable for the application programs, respectively.

4.1 Sample Programs

Following three programs were used in the experiments.

(1) **INNER**: Inner product calculation of two vectors with 200 elements each
(2) **MATRIX**: Matrix product calculation of two matrixes of size 10x10

(3) **MERGE**: Merge sort of 20 elements

4.2 Processor Model

The basic instruction set of the processors examined in the experiments is that of MIPS R3000 processors, but some specific instructions were added according to sample programs. These processors were designed by using an ASIP design workbench PEAS-III [4]. In this experiment, common features of designed processors for all three sample programs were as follows:

(1) 32 bit RISC processor of Harvard architecture,
(2) 5-pipeline stages (IF, ID, EX, MEM, WB),
(3) One delay slot.

Number of instructions, area and delay of designed processors are summarized in Table 2. The environment used in the experiments is shown in Table 3.

Table 2. Features of Designed Processors

Program	Instructions	Area[gates]	Delay [ns]
INNER	20	31,897	42.48
MATRIX	20	31,897	42.48
MERGE	20	19,628	12.64

Table 3. Experimental environment

Machine	Sun Ultra Enterprise 450
CPU	UltraSparc 296MHz×2
Memory	1.6GB
Technology	CMOS 0.6um (VDD=5V)
Place & Route	Avant! Milkyway, Apollo
RTL Simulation	Synopsys VSS
Gate Level Simulation	Synopsys VSS
Gate Level Power Estimation	Synopsys DesignPower
Processor Design	PEAS-III
Compiler	GCC
Inst. Set Simulator	SPIM[5]
HDL	VHDL
Application Lang.	C

4.3 Experimental Results

4.3.1 Estimation Error

The estimated results by the proposed method were compared with that of a power estimation tool DesignPower by Synopsys, Inc. The reported values by DesignPower was assumed to be the true values, because it is one of the most reliable industrial standard gate level power estimation tools.

Table 4 shows estimated power and the errors when proposed method and conventional method were used. Conventional method does not utilize the information of the data types of variables. From Table 4, following observations can be made.

(1) Estimation error of the conventional method is very large.
(2) Estimation error of the proposed method is drastically reduced.

Table 4. Power consumption estimate results

Program	True Value [mW]	Error [%] in proposed method	Error [%] in conventional method
INNER	617.1696	-6.41	151.98
MATRIX	582.7245	-4.92	166.88
MERGE	515.4261	-5.42	39.71

True Value: Report form DesignPower by Synopsys

4.3.2 Performance

The execution time of the program to estimate the power consumption for sample programs is shown in Table 5. The proposed method can be classified as an Instruction Level Profiling method. From this table, it is known that the proposed method is at least 1,000 times faster than other profiling method.

Table 5. Execution Time of Profiling

Program	Inst. Level Profiling [sec]	RT Level [sec]	Gate Level [sec]
INNER	0.07	511.10	1,681.70
MATRIX	0.09	1,575.10	12,827.40
MERGE	0.05	89.80	747.20

5. Conclusion

In this paper, a very efficient and accurate power estimation method was proposed and the effectiveness of the proposed method was evaluated through experiments.

While the proposed method performs instruction level profiling instead of RT level profiling, the estimated power consumption is much more accurate as RT level estimation method. Therefore, the proposed method is suitable for designing huge space exploration of ASIPs.

Our future work includes the development of a more accurate estimation method taking more detailed behaviors of processor, such as pipeline stall, into account.

6. Reference

[1] F. Najm, "A survey of power estimation techniques in VLSI circuits," IEEE Transactions on VLSI Systems, pp. 446-455, Dec. 1994.

[2] P. Landman, "High-Level Power Estimation," Int. Symp. Low Power Electronics and Design, pp. 29-35, Aug. 1996.

[3] M. Pedram, "Power Simulation and Estimation in VLSI Circuits," The VLSI Handbook, The CRC Press and the IEEE Press, 1999.

[4] M. Itoh, Y. Takeuchi, M. Imai, and A. Shiomi, "Synthesizable HDL Generation for Pipelined Processors from Micro-Operation Description," IEICE Trans. on Fundamentals., Vol. E83-A, No.3, pp.394-400, Mar. 2000.

[5] J. R. Larus, "SPIM S20: A MIPS R2000 Simulator," University of Wisconsin-Madison, 1990.