System-Level Leakage Power Estimation Model for ASIC Designs

Abhishek Narayan TRIPATHI, Arvind RAJAWAT

Department of Electronics & Communication Engineering, Maulana Azad National Institute of Technology Bhopal, Link Road Number 3, Near Kali Mata Mandir, Bhopal, 462003 Madhya Pradesh, India

abhi.sanutripathi@gmail.com, rajawata@manit.ac.in

DOI: 10.15598/aeee.v16i3.2947

Abstract. *Nowadays, with advances in CMOS technology and sub-micron processes, the leakage power dissipation is becoming a critical design metric. The designs are getting complex to incorporate more functions, thereby increasing the leakage power dissipation. The low power design objective requires early exploration and estimation. This paper presents the power estimation model for ASIC (Application Specific Integrated Circuit) based designs at the C-level of abstraction. The method includes analysis and extraction of the application specific information from the LLVM (Low-Level Virtual Machine) bit-code and then training of the neural network. The trained model provides the estimation of the leakage power. The estimated leakage power of designs is compared with the implemented power to demonstrate the accuracy. In addition, the model provides fast estimates and eliminates the need for synthesis based exploration.*

Keywords

Artificial neural network (ANN), ASIC, LLVM IR (Intermediate representation), power estimation, system-level.

1. Introduction

Modern applications require large on-chip functions which results in increased design complexity. Currently, leakage power by such complex designs, makes power budget a critical issue. In the ASIC design, the leakage power of cells, such as AND, NAND, MUX increases as technology node decreases. Complex functional design and implementation require a number of combinational and sequential cells. The sub-threshold leakage, gate-oxide tunneling leakage, and reverse bias drain-substrate leakage are the primary sources of leakage power dissipation [1]. The leakage power remains present in active as well as in the idle time. Thus, it is a primary concern for the designers. Since changes in the designs at a later stage is difficult.

The leakage power can be determined very quickly at the system-level. This reduces the design cycle time. Modeling of designs at the system-level provides an opportunity to explore the design metrics at the initial phase. Modeling of HW and SW was carried out at the cycle accurate level [2], instruction level [3] and functional level [4] to get the power estimates. However, there is a need of a high-level model that can provide quick as well as accurate estimates of design power at the high-level of abstraction. Therefore, this work presents a model to get an accurate early power estimate at the system-level. The LLVM IR and ANN are used for the profiling and the modeling, respectively.

The rest of the paper is organized as follows. Section 2. discusses related work. Section 3. discusses the profiling and modeling methodology. Section 4. presents the implementation and the results and finally, Sec. 5. presents the conclusion.

2. Related Work

Sources of the leakage current and reduction techniques were discussed in [1]. These techniques are proposed at the circuit-level and process technology level. Thus, to decrease the design time, estimation should be done at the high level of abstraction.

The leakage power for ASIC designs was estimated in [5]. The linear regression model, with a maximum error of 12 %, has been reported for the HDL description of the circuits.

In [6], early power estimation model was presented for the FPGA based designs with better accuracy. However, it does not provide any model for ASIC based designs.

In [2], power was evaluated by observing the cycle by cycle operations and exchanges. However, this work models the static power at the cost of the accuracy.

A SystemC class library, PowerSim, was proposed in [7], to calculate the energy consumption of hardware described at the system-level. It was based on monitoring the C++ operators when called on SystemC data types at the time of simulation. However, PowerSim requires a change in SystemC library and recompilation of source code.

The IO ports power consumption of OpenRISC architecture was modeled in [3]. IO ports power was modeled using the types of instructions such as load and store, and the cache miss rate. Since the effect of the adjacent instruction was not considered, the maximum error of 15 % was reported.

Open-PEOPLE (Open Power and Energy Optimization and Estimator), an estimator, was presented in [8] with the integration of estimation tools from the functional level to real board measurements, and a maximum error of 22.5 % was reported.

The HLS based, a fast system-level power estimation method was presented in [9]. However, a maximum error of 9 % was reported for the DSP based designs. Similarly, a neural network based linear and nonlinear system-level power estimation models were presented in [10]. However, the maximum error of 31 % and 4.78 % were reported for the linear and nonlinear models, respectively.

A System-level power estimation model was proposed in [11] for the wireless communication systems, implemented on the FPGA. However, it takes a significant amount of time because of the low-level characterization.

In [4], learning methods based on high-level energy models were used for the processors. Host compiled modeling approach instead of instruction level simulation was used. The observed maximum estimation error is 10 % for considered applications GEMM, DCT, and HDR.

Nowadays, with the help of high-level synthesis [12], [13] and [14] tools, it is possible to start the design space exploration at the system-level and synthesize the behavioral description into the RTL model. The RTL model of any given application simulated with a test bench provides a power estimate. PETS, a simulationbased tool, was presented in [15]. The power was estimated by running an application on the embedded platform. However, it does not generalize the power model and requires a new power model for a new architecture.

PK tool 2.0, a power estimation model, was introduced in [16] for the executable designs. It directly extracts the energy consumption from SystemC sc_module, which contains the system to examine. Since the SystemC sc_module was divided into the different power states representing the operative conditions and the set of instructions, it was required to provide the power states information to the tool.

To summarize, all of the aforementioned approaches are time-consuming, inaccurate, and complex. The power estimation is difficult at the system-level due to the lack of the information and may lead to the inferior precision. Consequently, it prevents a fast and an efficient design space exploration.

Therefore, this paper presents a system-level power estimation model to overcome the aforementioned challenges.

3. Profiling and Modeling Methodology

Figure 1 shows the profiling and modeling flow.

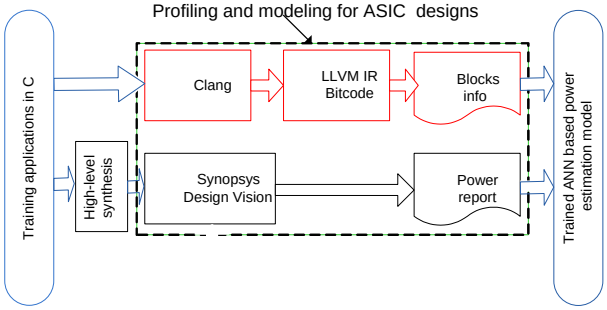

Fig. 1: Profiling and Modeling flow for estimation.

The profiling phase uses the analysis of the LLVM IR (bit-code) format of the applications. The LLVM IR is a static single assignment [17] and an easier representation to analyze than the machine level representation of applications. This leads us to collect the application specific information from LLVM IR through static code analysis.

LLVM bit-code files were generated by the Clang compiler. Bit-code analysis flow is shown in Fig. 2. The bit-stream files contain tags and nested structures. Blocks in a bit-stream denote nested regions of the stream and are identified by a content-specific ID number. The block IDs 0-7 are reserved and the IDs greater than 7 provide the application specific information. The information from these blocks used as the inputs to the proposed model, is shown in Tab. 1.

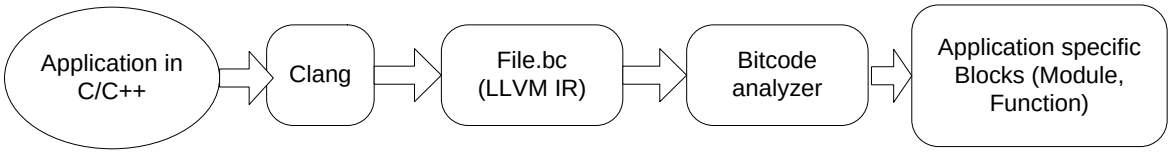

Fig. 2: Bit-code analysis flow.

Tab. 1: Type of LLVM IR Blocks.

Type of Block	Block ID	Content
Module Block	8	Top-level block, contains the entire module information
Parameter Block	9	Parameter attributes
Constant Block	11	Constants for a module or function
Function Block	12	Describes function body
Type Block	17	Describes all of the types in the module

The training data set was prepared to train the ANN-based model after profiling of the applications [18]. The feed-forward-back-propagation neural network is selected for the training. Input dataset with corresponding target data is applied to the network for the supervised learning and training. The errors are generated by comparing the training output with initially applied target data to the network. The weights of each connection get updated according to these errors. Then the network is trained with the updated weights. The *trainlm* and *learngdm* are the selected training algorithm and transfer function, respectively. *Tansig* and *purelin* are activation functions for the input and the output layer neuron, respectively. The network configuration and training parameters are shown in Tab. 2.

Tab. 2: Network configuration for leakage power estimation.

Parameters [18]	
Network type	**Feed-forwardback propagation**
Training function	*trainlm*
Learning function	*learngdm*
Activation function	*tansig*, *purelin*
Number of layers	2
Number of neurons	7

4. Implementation and Result

In this section, estimation results are presented for ASIC based designs. A total of 74 applications were used for the training. After that, the trained model was validated for the different benchmark applications. The information obtained from the LLVM IR was applied to the proposed neural network model. The proposed neural network model is shown in Fig. 3.

Input layers have 7 inputs and 7 neurons. Transfer function for input neurons is *tansig*. Transfer function for the second layer, which is the output layer, is *purelin*. The following equations represent the output of this neural network configuration.

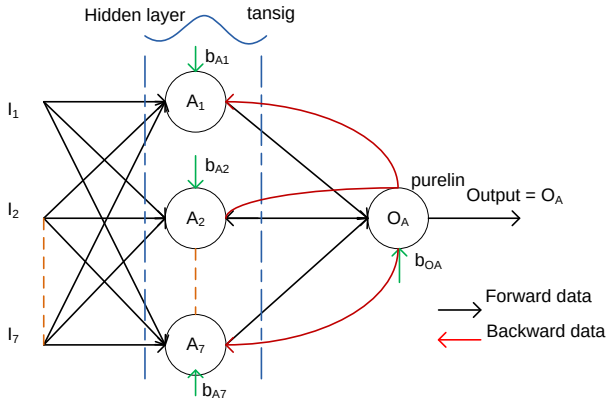

Fig. 3: Proposed ANN model.

$$A_n = tansig\left(\sum_{n=1}^{n=7} w_{na} I_n + b_{Aa}\right), \tag{1}$$

$$Output = purelin\left(\sum_{a=1}^{a=7} A_n w_{AaO} + b_{OA}\right). \tag{2}$$

The output of this model is given by Eq. (2). In the Eq. (1) and Eq. (2), I_n are the inputs, w_{na} are the weights from the inputs to the hidden layer neurons, b_{Aa} are the biases to the hidden layer neurons, w_{AaO} are the weights of hidden layer neurons to the output layer neuron, and b_{OA} is the bias to the output neuron.

Tab. 3: Benchmark/Application description.

Benchmarks	Description
gsm	Linear predictive coding analysis of global system for mobile communications
dfsin	It implements IEC/IEEE standard double precision floating point sine function using 64-bit integer numbers
dfdiv	It implements IEC/IEEE standard double precision floating point division using 64-bit integer numbers

The proposed ANN-based model is applied to the CHStone [19] benchmark applications, mentioned in Tab. 3.

The size of the blocks is shown in Tab. 4 for the benchmark applications [19].

Tab. 4: Type and size of LLVM IR Blocks.

Type of Block	Size of Blocks		
	gsm	dfsin	dfdiv
Module Block	85	154	114
Parameter Block	9	3	3
Constants Block	399	441	343
Function Block	1557	1773	916
Type Block	33	43	37
Constant block file	16.3	14.5	18
Function block file	63.8	58.2	48

4.1. Validation and Comparison

In this section, we will be validating the estimated power against the power obtained from the commercial tool, in order to demonstrate the accuracy of the proposed methodology. Figure 4 shows the result validation and comparison flow. The output of the proposed model, estimated leakage power, is compared with the power obtained from the Synopsys Design Compiler. Designs are compiled using Synopsys Design Compiler $SAED\_90nm$ library. The error in evaluated power amends the weights of the input for the network in the training phase. Now, the trained network is ready to estimate the power for any given application.

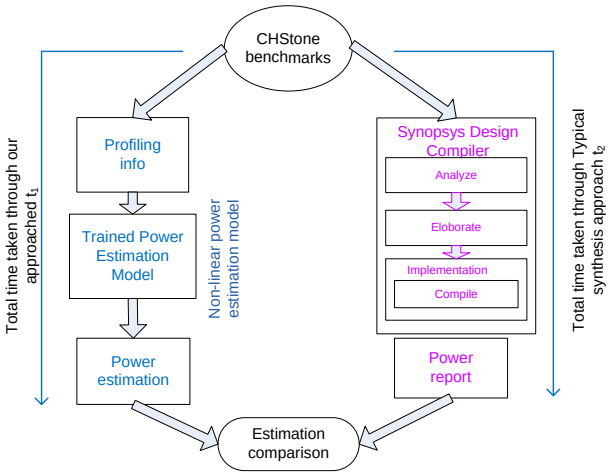

Fig. 4: Validation and Comparison.

The estimated results are compared using the following equation.

$$error_i = \left(\frac{e_i - p_i}{p_i}\right) \cdot 100. \tag{3}$$

In the above expression, $error_i$ is the percentage error in the estimate, e_i is estimated power obtained from the presented methodology, and p_i is power obtained from Synopsys DC, for an application i. The estimated results and comparisons are shown in Tab. 5 and Tab. 6, respectively. The time comparison is shown in Tab. 7. The platform used for the time comparison is the Intel Core i3 @ 3.3 GHz processor. Since the LLVM profiling output comes in less than 1 second, only the training time of the ANN was considered. The total time, t_2, taken by Synopsys DC, for the benchmark applications in this paper is 4 minutes and 53 seconds. However, through the proposed model, estimation time (t_1) is 58 times faster than the typical synthesis tool.

Tab. 5: Comparison of Estimated Leakage power with Implemented power.

	Through Synopsys DC (mW)	Proposed model (mW)	Relative Error %
gsm	3.90	3.70	5.10 %
dfsin	3.17	3.26	2.80 %
dfdiv	2.30	2.40	4.30 %

Tab. 6: Comparison with other works.

S. No.	Works	Error %
1.	[4]	3 % to 10 %
2.	[9]	3 % to 9 %
3.	This work	2.8 % to 5.1 %

Tab. 7: Time taken by Synopsys Design Compiler (DC) and proposed model to get power estimate for Intel Core i3 @ 3.3 GHz processor platform.

Benchmarks	Through Design Compiler t_2	Through Presented model t_1
gsm	2 min 13 sec	–
dfsin	1 min 35 sec	–
dfdiv	1 min 5 sec	–
Total Time	4 min 53 sec	5 sec

5. Conclusion

A system-level power estimation model was presented for the ASIC designs. The application-specific information was obtained through the analysis of the LLVM bit-code, then further employed to train the ANN. The estimation errors of 2.8 % to 5.1 % were observed. Moreover, the model was found 58 times quicker than the Synopsys Design Compiler. In addition, it is possible to extend this work for an early estimate of the area of the ASIC designs.

References

[1] ROY, K., S. MUKHOPADHYAY and H. MAHMOODI-MEIMAND. Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits. *Proceedings of the IEEE*. 2003, vol. 91, iss. 2, pp. 305–327. ISSN 0018-9219. DOI: 10.1109/JPROC.2002.808156.

[2] LI, S., J. H. AHN and R. D. STRONG. Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In: *MICRO, IEEE/ACM International Symposium on Microarchitecture*. New York: IEEE, 2009, pp. 469–480. ISBN 978-1-60558-798-1.

[3] SULTAN, S. and S. MASUD. Rapid software power estimation of embedded pipelined processor through instruction level power model. In: *International Symposium on Performance valuation of Computer and Telecommunication Systems*. Istanbul: IEEE, 2009, pp. 27–34. ISBN 978-1-4244-4165-5.

[4] LEE, D., L. K. JOHN and A. GERSTLAUER. Dynamic Power and Performance Back-Annotation for Fast and Accurate Functional Hardware Simulation. In: *Design Automation of Test in Engineering*. Grenoble: IEEE, 2015, pp. 1126–1131. ISBN 978-3-9815-3705-5.

[5] KUMAR, R. and C. P. RAVIKUMAR. Leakage power estimation for deep submicron circuits in an ASIC design environment. In: *Proceedings of Asia and South Pacific Design Automation Conference*. Bangalore: IEEE, 2002, pp. 45–50. ISBN 0-7695-1441-3. DOI: 10.1109/ASPDAC.2002.994883.

[6] TRIPATHI, A. N. and A. RAJAWAT. Fast and efficient power estimation model for FPGA based designs. *Microprocessors and Microsystems*. 2018, vol. 59, iss. 1, pp. 37–45. ISSN 0141-9331. DOI: 10.1016/j.micpro.2018.03.005.

[7] GIAMMARINI, M., M. CONTI and S. ORCIONI. System-level energy estimation with Powersim. In: *International Conference on Electronics, Circuits, and Systems*. Beirut: IEEE, 2011, pp. 723–726. ISBN 978-1-4577-1846-5. DOI: 10.1109/ICECS.2011.6122376.

[8] ATITALLAH, R., E. SENN, D. CHILLET, M. LANOE and D. BLOUIN. An Efficient Framework for Power-Aware Design of Heterogeneous MPSoC. *IEEE Transactions on Industrial Informatics*. 2013, vol. 9, iss. 1, pp. 487–501. ISSN 1551-3203. DOI: 10.1109/TII.2012.2198657.

[9] AHUJA, S., D. A. MATHAIKUTTY, G. SINGH, J. STETZER, S. K. SHUKLA and A. DINGANKAR. Power estimation methodology for a high-level synthesis framework. In: *International Symposium on Quality of Electronic Design*. San Jose: IEEE, 2009, pp. 541–546. ISBN 978-1-4244-2952-3. DOI: 10.1109/ISQED.2009.4810352.

[10] DENG, C., L. LIU, Y. LIU, S. YIN and S. WEI. PMCC: Fast and Accurate System-Level Power Modeling for Processors on Heterogeneous SoC. *IEEE Transactions on Circuits and Systems II: Express Briefs*. 2017, vol. 64, iss. 5, pp. 540–544. ISSN 1549-7747. DOI: 10.1109/TCSII.2016.2615930.

[11] LORANDEL, J., J. PREVOTET and M. HELARD. Fast Power and Performance Evaluation of FPGA-Based Wireless Communication Systems. *IEEE Access*. 2016, vol. 64, iss. 1, pp. 2005–2018. ISSN 2169-3536. DOI: 10.1109/ACCESS.2016.2559781.

[12] NAVARRO, D., O. LUCIA, L. A. BARRAGAN, I. URRIZA and O. JIMENEZ. High-Level Synthesis for Accelerating the FPGA Implementation of Computationally Demanding Control Algorithms for Power Converters. *IEEE Transactions on Industrial Informatics*. 2013, vol. 9, iss. 3, pp. 1371–1379. ISSN 1551-3203. DOI: 10.1109/TII.2013.2239302.

[13] CONG, J., B. LIU, S. NEUENDORFFER, J. NOGUERA, K. VISSERS and Z. ZHANG. High-Level Synthesis for FPGAs: From Prototyping to Deployment. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 2011, vol. 30, iss. 4, pp. 473–491. ISSN 0278-0070. DOI: 10.1109/TCAD.2011.2110592.

[14] HLS, X. Vivado High level synthesis. In: *Xilinx* [online]. 2018. Available at: `http://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html`.

[15] RETHINAGIRI, S., O. PALOMAR, O. UNSAL, A. CRISTAL and R. ATITALLAH. PETS: Power and energy estimation tool at system-level. In: *Fifteenth International Symposium on Quality Electronic Design*. Santa Clara: IEEE, 2014, pp. 535–542. ISBN 978-1-4799-3946-6. DOI: 10.1109/ISQED.2014.6783373.

[16] VECE, G. B., M. CONTI and S. ORCIONI. Transaction-level power analysis of VLSI digital systems. *Integration, the VLSI Journal*. 2015, vol. 50, iss. 1, pp. 116–126. ISSN 0167-9260. DOI: 10.1016/j.vlsi.2015.02.003.

[17] The llvm compiler infrastructure project. In: *LLVM* [online]. 2010. Available at: http://www.llvm.org.

[18] Neural Network Toolbox. In: *MathWorks* [online]. 2018. Available at: http://www.mathworks.com.

[19] HARA, Y., H. TOMIYAMA, S. HONDA and H. TAKADA. Proposal and Quantitative Analysis of the CHStone Benchmark Program Suite for Practical C-based High-level Synthesis. *Journal of Information Processing*. 2009, vol. 17, iss. 1, pp. 242–254. ISSN 1882-6652. DOI: 10.2197/ipsjjip.17.242.

About Authors

Abhishek Narayan TRIPATHI received the B.E. degree in Electronics and Communication engineering from the Rajiv Gandhi Proudyogiki Vishwavidyalaya (RGPV) University, Bhopal, India, in 2008, and the M.Tech degree in microelectronics and embedded technology from the Jaypee Institute of Information Technology, Noida, India, in 2012. He is currently pursuing the Ph.D. degree in electronics and communication engineering with the Maulana Azad National Institute of Technology Bhopal, Bhopal, India. His research interests include System-level design, VLSI Design.

Arvind RAJAWAT received his Bachelor of Engineering in Electronics and Communication Engineering from Government Engineering College Ujjain (MP, India) in 1989, Master of Engineering in Computer Engineering from Shri Govindram Seksaria Institute of Technology and Science (SGSITS) Indore (MP, India) in 1991 and Ph.D. from Maulana Azad National Institute of Technology Bhopal (MP, India) in 2009. He is currently working as Professor of Electronics and Communication Engineering with the department of Electronics and Communication Engineering at MANIT Bhopal (MP, India). His areas of interest are Hardware Software Co-Design, Embedded Systém Design, Digital System Design.