PowerCount
Measuring the Power at the VHDL Netlist Level

A.Th. Schwarzbacher, P.A. Comiskey and J.B. Foley
Department of Electronic and Electrical Engineering, Trinity College
Trinity College, Dublin 2, Ireland

Abstract: Up until recently area and speed were the main factors concerning integrated circuit (IC) design. Traditionally portable applications such as notebooks and cellular phones where the only applications where the power consumption was of interest to the designer. But higher integration densities and increased speed have enabled the implementation of real time DSP applications.
To implement such DSP structures it is important to monitor the power consumption at the earliest possible stages in order to avoid time consuming and expensive redesigning. This paper will present PowerCount, a tool developed to estimate the power consumption by means of standard VHDL netlists.

1 Introduction

Traditionally power consumption is estimated at the lowest possible level by simulating at the layout level in order to guarantee a high accuracy. This is usually done be means of SPICE simulations. Even if these simulations are very accurate, they not only have the drawback that they are slow, but also that they do not analyse the design until its final stages. This makes redesigning time intensive and therefore uneconomic.

Other approaches to determine the power consumption such as those taken by Powermill [1] or IRSIM [2], estimate the power at the RTL by means of a netlist specially created for this purpose. Even if these tools are much quicker and provide accuracy better than 90%, they still have the drawback that a special netlist has to be generated from a routing tool. These tools depend on a special design library and therefore on a particular vendor. Figure 1 shows the traditional approach taken to calculate the power consumption of IC's.

High Level Power Estimation Tools are relatively new and usually make some generalisations in order to compute the power consumption. Instead of using the actual value of the node capacitance, they use the average node capacitance. The second idea used is to replace the switching activity factor $n_{(0,1)}$ with the probability factor p_t. This factor reflects the probability that a power consuming event occurs. This probability factor is calculated for a uniform distributed white noise (UWN) input signal and stored with the library information. This probability factor does not reflect the actual switching activity at the node. However, this method is very fast since no simulations with large input data have to be made. But this method also has its drawbacks. The generalisations assume that each node regardless of its load or output capacitance has the same node capacitance. This is not true in real life. Therefore, a path with a high switching activity but low node capacitance or vice-versa, will have a negative influence on the result. Furthermore, it is highly unlikely that the input signal is of random nature. Internally, due to the connections between the different units and the distribution of the signal, the switching activity is highly correlated and is not random. These methods do not take this into account. Therefore, different techniques which use the probability

factor in conjunction with the real node capacitance are introduced. Other techniques first simulate the input probability of a module and overlay this with data from a UWN simulation [3]. This method has the drawback that a design has to be described at a software level. Then, a full simulation with actual input data is performed in order to obtain the actual input probabilities at each block. With these probabilities, the input vectors are generated and the design is resimulated. Recently, Landman purposed a new method, called the Dual Bit Type model, which accounts for input correlation [4]. This model relies on a specially prepared netlists, libraries and look up tables. The model also needs calibration for each type of design.

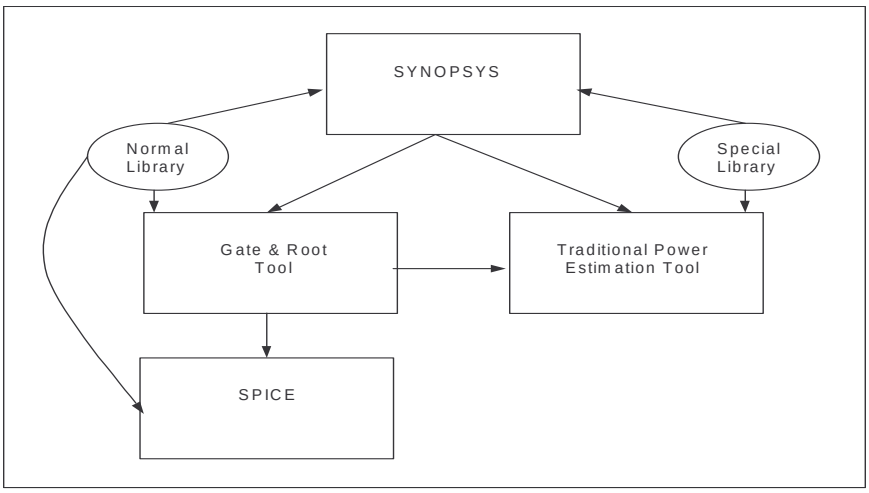

Figure 1: Hierarchy of Traditional Power Estimation

All of these methods rely on special design libraries, which include the switching probability for each cell. Most estimation methods also use a zero delay model. These models do not take delay times into account and neglect glitches and hazards. These glitches depend mainly on the implementation of the algorithm and can only be accounted for by simulating with real input data.

Finally, it should be noted that most of these methods only work for combinational logic [5]. When estimating the power consumption of sequential logic the state probabilities have to be taken into account. These were the various reasons why PowerCount was designed, in order to overcome these problems.

2 PowerCount

One of the main features of PowerCount is that it works as an add-on tool to Synopsys. Normally the designer writes the algorithm to implement in a HDL. Than the correct logic function is of the code is verified before the code is synthesised. After synthesis timing simulations are performed. These simulations use the node capacitance's estimated by the synthesis tool in order to determine rise and fall times. Therefore, these simulations include glitches and hazards. The basic idea of PowerCount is to analyse these timing simulations. The result of each node is than multiplied with the actual node capacitance. Finally the active capacitance is calculated by summing the individual node capacitances.

All the above steps are automatically performed by PowerCount. The only additional file needed when compared with a pure timing simulation is the node capacitance report. This file can be automatically generated within the Synopsys Design Compiler environment. A file for more than 3000 nodes is typically generated in under 10s on a Sparc5 workstation. The great advantage of PowerCount is that the designer does not need to prepare special netlists. He also does not need to prepare special input vectors. PowerCount uses the normal HDL testbench. Therefore, the same input vectors can be used as for the logic verification and the timing simulations. The power simulation is controlled via the VSS control language used for all Synopsys simulations. Therefore,

the designer does not need to learn a new complex language and is also not restricted to a specific HDL.

Unfortunately is not possible to generate a random seed within the HDL testbench. Therefore, PowerCount supplies the control script with a 64 bit random number. This number can then be assigned to a variable in the testbench.

In order to ensure accurate results PowerCount operates on the Monte Carlo principle [6]. This means that first a number of simulations are carried out. Then the variance is calculated. If the variance is under a specified value the simulation terminates. Otherwise it continues until the error constraint is met.

3 Conclusions

A tool, PowerCount, was developed to estimate the power consumption of CMOS circuits, at the netlist level. This tool has the general advantages over existing tools. It is an add-on tool to the Synopsys Design Environment. It uses only one additional file, when compared to timing analysis. This additional file can be generated in seconds in the Design Compiler environment. Figure 2 shows the interaction between PowerCount and Synopsys.

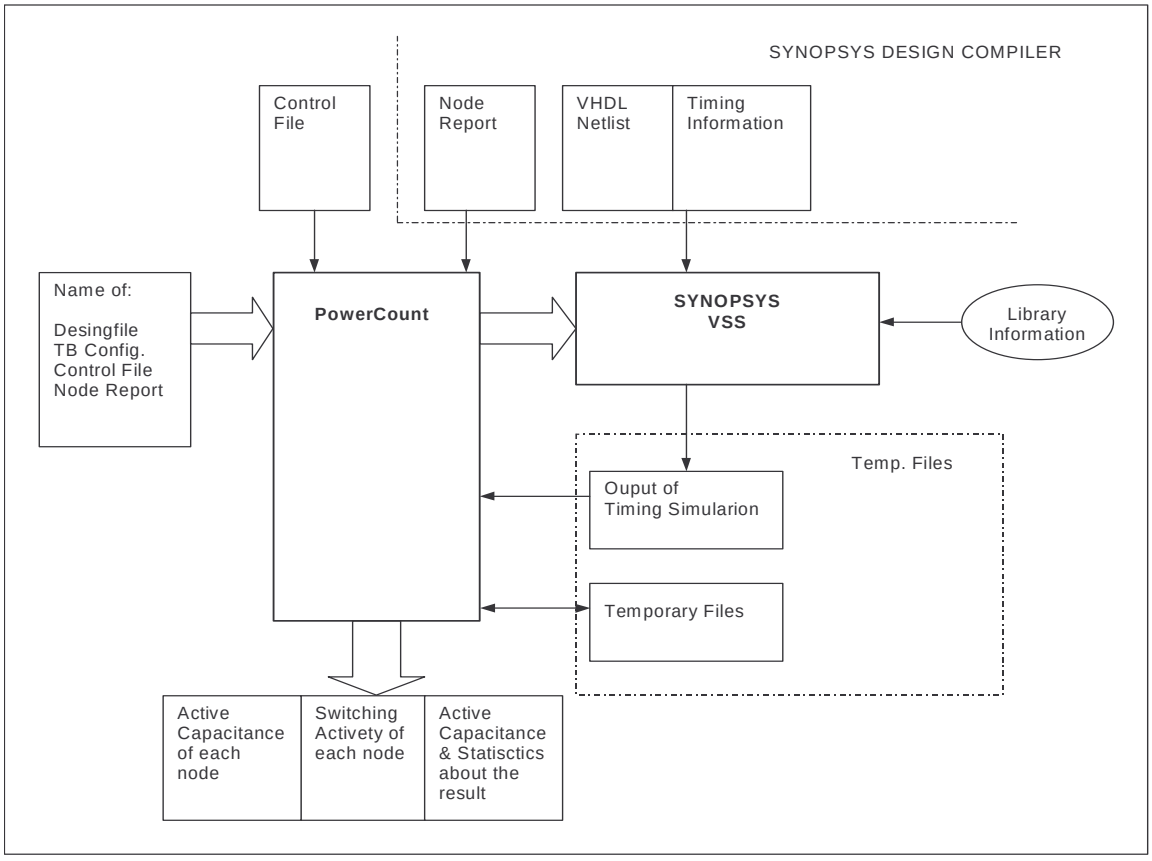

Figure 2: PowerCount in the Synopsys Environment

PowerCount computes the active capacitance. It uses the most accurate node capacitance information available at this level. This node information includes the capacitance of the interconnect, as estimated by Synopsys. Thus real timing simulation with the Synopsys System Simulator is used in order to compute the actual switching of each node. These simulations include the glitching activity. Any kind of input signal can be applied to the circuit. This gives the designer the freedom to use real input data instead of UWN.

In order to reduce the simulation time and ensure maximum accuracy, the simulation uses a Monte Carlo technique. Simulations have shown that even large designs with more than one

thousand nodes can be simulated in under one hour. Extensive tests with a variety of benchmark circuits have been carried out. Figure 3 represents the complete breakdown of power consumption of the sub-architectures in a 40 MHz image processing chip. Figure 4 shows the active capacitance of a 64-bit adder as estimated by PowerCount, over 30 simulations.

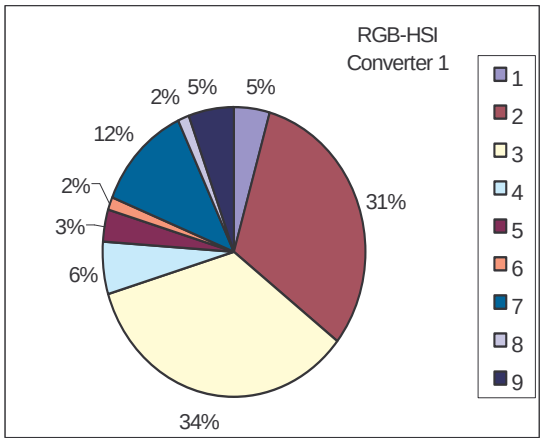

Figure 3: A Image Processing Module

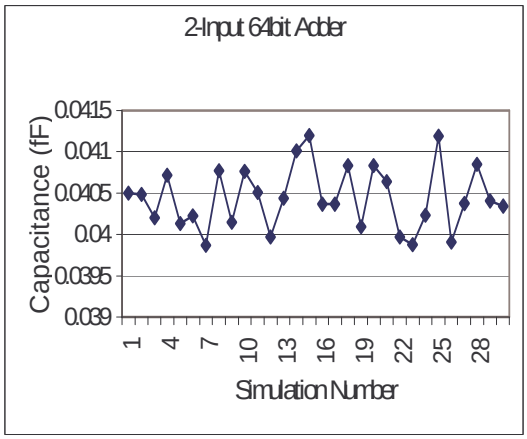

Figure 4: Simulation of a 64-bit Adder

Acknowledgements

The authors wish to thank the Dublin Institute of Technology, in particular Mr. Chris Cowley for access to their research facilities.

References

[1]
[2] A. Satlz and M. Horowitz, “IRSIM: An Incremental MOS Switch-Level Simulator,” *Proceedings of the 26th DAC*, pp173-178, 1989.
[3] T. Burd and B. Peters, "A Power Analysis of a Microprocessor: A study of the MIPS R3000 architecture," May 1994.
[4] P. Landman and J. Rabaey, "Black-Box capacitance models for architectural power analysis," *Proc. of 1994 International Workshop on Low-Power Design*, April 1994.
[5] C. Tsui, J. Monteiro, M. Pedram, *et al.* "Power estimation for sequential logic circuits," *unpublished. Dept. of EE, University of California, Los Angeles.* 1994.
[6] Burch, Kang et al., “McPower: A Monte Carlo Approach to Power Estimation”, IEEE/ACM Int. Conf. Computer Aided Design, pp.90-97, November 8-12, 1992.