EmpowerSoC: An Open-Source Power Analysis Engine based on Qflow

Akshat Jain
Department of Electronics and Communication Engineering
Netaji Subhas University of Technology
Delhi, India
jakshat2312@gmail.com

Naveen Dugar
Department of Electronics and Communication Engineering
Netaji Subhas University of Technology
Delhi, India
naveen.dugar1010@gmail.com

Sagar Yadav
Department of Electronics and Communication Engineering
Netaji Subhas University of Technology
Delhi, India
13sagaryadav@gmail.com

Kunwar Singh
Department of Electronics and Communication Engineering
Netaji Subhas University of Technology
Delhi, India
kunwar.singh@nsut.ac.in

***Abstract*—The paper introduces EmpowerSoC which is an open-source power analysis engine based on the Qflow tool chain. Both active and standby power consumption can be estimated using this tool. Qflow is used to run the RTL to GDSII flow a target design and is a silicon proven flow [1]. Power estimation in EmpowerSoC is done by extracting transistor-level post-layout netlists for various building blocks/cells on an automated basis. It has a user-friendly GUI through which users can set input bit patterns and other simulation parameters. It has been tested on various digital blocks and the obtained results have been presented.**

***Index Terms*—EmpowerSoC, Open-Source, Qflow, Power Analysis Engine, EDA**

I. INTRODUCTION

Power analysis is an important consideration while designing an SoC. Manual estimation of power dissipation for Very Large Scale Integrated (VLSI) Circuits is far beyond human ability because of their complexity. Therefore, in order to analyze power in these circuits one has to use professional computer-aided tools. Most of such tools are proprietary which makes them less accessible to the vast majority of students. EmpowerSoC is a Qflow based power analysis engine that is built to fulfill the same purpose. The proprietary software has to maintain cutting-edge performance, which justifies their cost. EmpowerSoC on the other hand provides an open-source alternative that can be used by students and independent researchers to estimate power consumption for building blocks of an system on chip (SoC).

Power dissipation is the major driving force behind the development of the EmpowerSoC tool. With the advent of transistors, the number of transistors being used to fabricate SoCs has been exponentially increasing. With such a huge increase in the number of transistors on chip , power dissipation is very high and represents a critical challenge. The increasing use of an SoC is compelling the designers to increase its functionality and at the same time reduce the power consumption. As the circuit becomes more complex, the power calculation becomes increasingly difficult and tiresome for the designers. Nowadays, power dissipation has become a major design entity that is kept in mind before designing and fabricating a circuit. Keeping a check on power dissipation becomes even more important owing to the rise in the number of portable electronic devices. Using the EmpowerSoC tool, one can estimate the power consumption of the digital blocks and comment on the feasibility of the design early in the design cycle.

The Qflow tool chain consists of various open-source tools namely Yosys[2], Graywolf, Qrouter[3], Opentimer/OpenSTA[4], and Magic[5]. One can extract information about area and timing/performance for a design from these tools but it lacks a power analysis engine. With EmpowerSoC we have tried to bridge this gap so that open-source designers have the option to predict the power consumption of their designs along with the chip area and performance.

Rest of the paper is organized as follows. Section II provides a brief overview of the various types of power dissipation encountered in CMOS technology and the methodology used by EmpowerSoC to calculate power consumption. This is followed by a description of the tool flow in Section III. The methodology used for power estimation is then applied on various RTL designs and presented in section IV. Finally, conclusion and future work is summarized in Section V.

II. POWER CONSUMPTION

This section talks briefly about the various sources of power dissipation in CMOS circuits as well as the methodology used by EmpowerSoC for power estimation. Power dissipation in CMOS circuits can be thought of as consisting of two components: dynamic power dissipation and static power dissipation. Dynamic power dissipation is is considered when the circuit is in active state. It consists of switching power dissipation, short circuit power sissipation, and glitching power dissipation

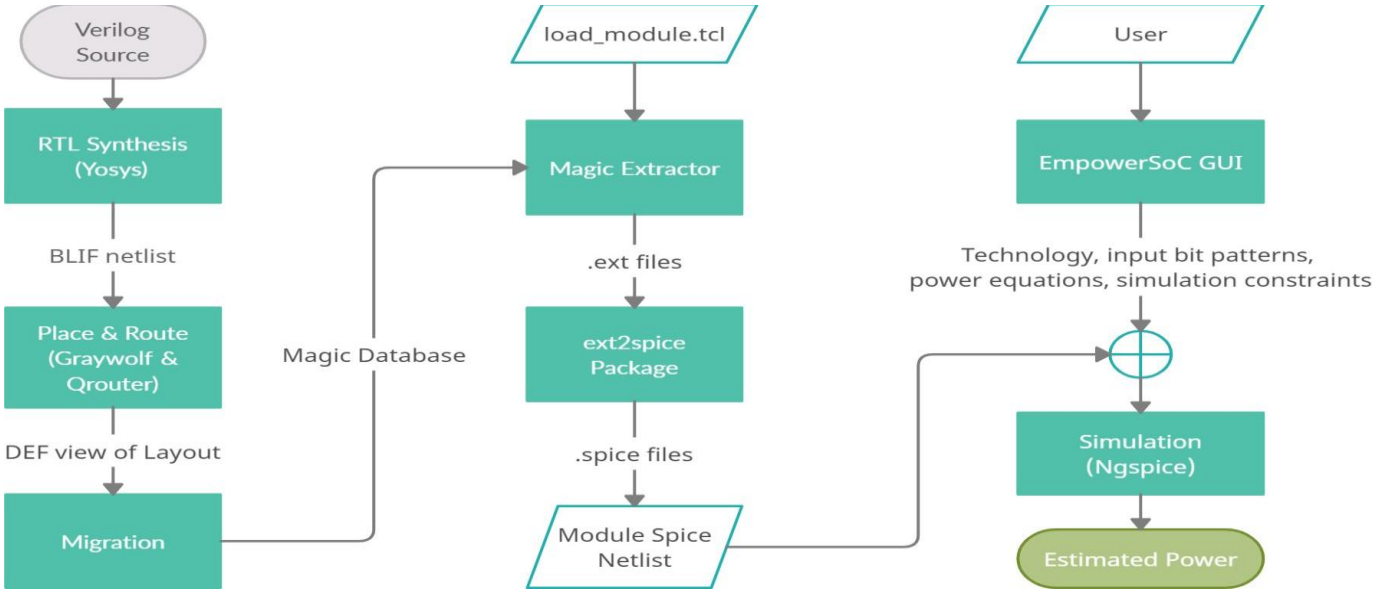

Fig. 1. EmpowerSoC Flow

Switching power dissipation consists of the power that is dissipated in the parasitic capacitors and resistors while charging and discharging. Short circuit Power dissipation consists of the power that is dissipated during switching when a path exists from VDD to Ground for current to flow. Glitching power dissipation occurs when the input signals to a particular logic block reach the inputs of a gate at different times resulting in glitches. Static power dissipation mainly consists of the power dissipated due to leakage current. Static power dissipation is a constant factor in CMOS circuits and is not dependent on switching i.e. it mainly occurs when the transistor is off. Below 90nm technology, it has become an important contributor to power dissipation. The main causes of static power dissipation are sub-threshold leakage, gate leakage, reverse-biased current, and junction leakage [6].

Another way of thinking about power consumption in CMOS circuits is through modes of operation, namely active mode and standby mode. Active power is the power consumed while the chip is active, the inputs and outputs are changing, and is doing useful work. It mainly consists of the switching power. Whereas, standby power is the power consumed when the device is in sleep mode or when the input and output nodes are stable and no useful work is being done. This is mainly dominated by leakage power. EmpowerSoC can be used for estimating both active and standby power consumption.

EmpowerSoC first calculates the total charge supplied to the design by integrating the instantaneous current drawn from the supply voltage over the simulation time frame. This total charge is then used to calculate the average energy supplied by taking its product with the supply voltage. Finally, this average energy supplied is used to obtain the value of average power consumption. During active mode, the input pin voltages are allowed to change through controlled pulses whose properties can be defined by the user. Whereas, in standby mode input pin voltages are held constant to keep the device in sleep mode.

III. TOOL FLOW

This section provides a brief introduction to the EmpowerSoC tool flow. Figure 1 illustrates the basic tool flow. The entire tool flow can be divided into three steps. Below is a summarized breakdown of the three stages.

A. Run RTL to GDSII flow using Qflow.

The first step is to run the RTL to GDSII flow using the Qflow tool chain. This digital synthesis flow takes as input a Verilog description of a circuit design and creates its physical implementation(layout) for a particular fabrication technology. The very capable open-source synthesis tool Yosys is being used to carry out the front-end synthesis. Synthesis includes mapping a logic circuit to a standard cell library. Here, the Oklahoma State University's open-source standard cell library is being used[7]. The circuit description in .blif format (Berkeley Logic Interchange Format) is also obtained after synthesis. The BLIF file is used to convert any digital circuit in text format, as any digital circuit can be visualized as a directed graph of combinational and sequential logic nodes. EmpowerSoC uses this to get useful information about the module. The next step in the flow is placement. Yale university's Graywolf tool takes care of placement. It is used for placing and assigning locations to various circuit subsystems within the chip. After placement comes routing, which is done by Qrouter. This step is essential for determining the detailed wiring between the placed subsystems within the chip. Opentimer or OpenSTA can be used for performing the static timing analysis (STA). The popular and easy-to-use MAGIC tool written by John Ousterhout is the open-source VLSI layout tool that is being used. It can perform design rule checks (DRC) and parasitic extraction. After verification, MAGIC can convert the data into the industry standard GDSII format. It must be noted that Qflow might not be suitable for carrying out the physical synthesis for the new generation complex microprocessors, but it is very well capable of creating the digital subsystems used in various chips.

B. Obtain transistor-level netlists for the sub-cells

The next step after running Qflow is to obtain transistor-level netlists for the sub-cells to eventually create the spice netlist for the complete circuit. After placement and routing, the layout is obtained in DEF file format. Migration step is done to convert this DEF view of the layout into a MAGIC layout database. This MAGIC database is loaded into the magic tool using a Tcl script. Then the Magic circuit extractor is invoked to extract the main cell and all sub-cells. This extraction computes information about the lumped node resistance, lengths, and widths of the transistors as well as information about the other parasitics involved, all of which is needed for carrying out the simulation. Various kinds of inter-nodal parasitic capacitances, as well as substrate capacitances, are computed during this process. This step ensures that our power estimation includes the parasitics component. After extraction, the .ext files for all the sub-components are obtained. Next, this extracted information in the form of .ext files is converted into spice files for the subcomponents using the ext2spice package in MAGIC. These Spice files are essential for carrying out the simulation in the final step. Once the spice definitions for all the subcells are extracted, the spice netlist for the complete cell can be created.

C. Calculate power consumption as per user-defined simulation constraints

After including the sub-circuit definitions and transistor-level netlist in a spice file, the complete spice netlist for target module was obtained. The next step is to include the model parameters for the transistors. EmpowerSoC's user-friendly GUI gives the user freedom to specify the simulation parameters based on their requirement. The user has the flexibility to use their own technology or use the default TSMC 180nm technology parameters provided with the tool. After specifying the technology, the input bit patterns specified by the user are used to add input pin excitations in the form of spice commands to the main netlist. Users can specify the excitations for the input pins in the form of controlled voltage pulses, or constant voltage signals. Thereafter, any external load capacitance as specified by the user are included in the main netlist. Finally, the relevant power calculation equations in the form of spice commands are included in the main netlist and the simulation is run in a circuit simulator like Ngspice[8] for the specified simulation period to calculate the power consumption. Both the active power consumption as well as standby power consumption can be calculated by selecting appropriate options in the GUI.

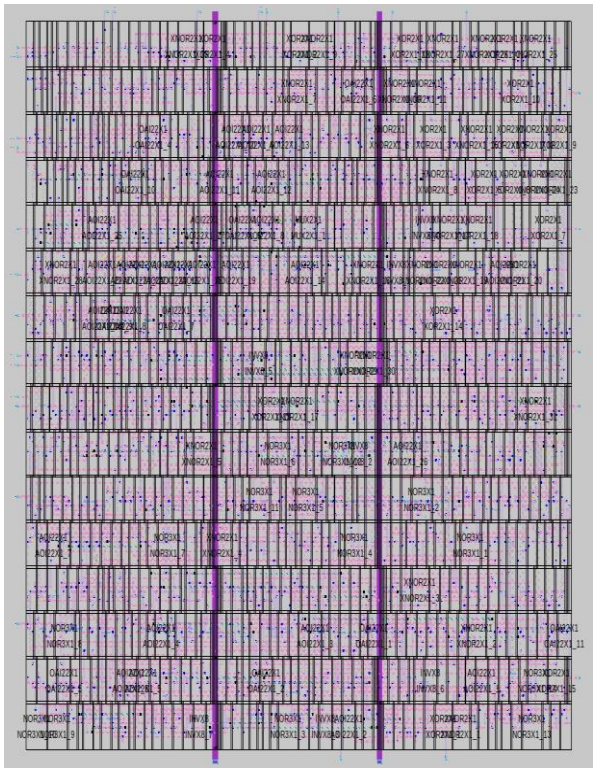

Fig. 2. ALU Layout

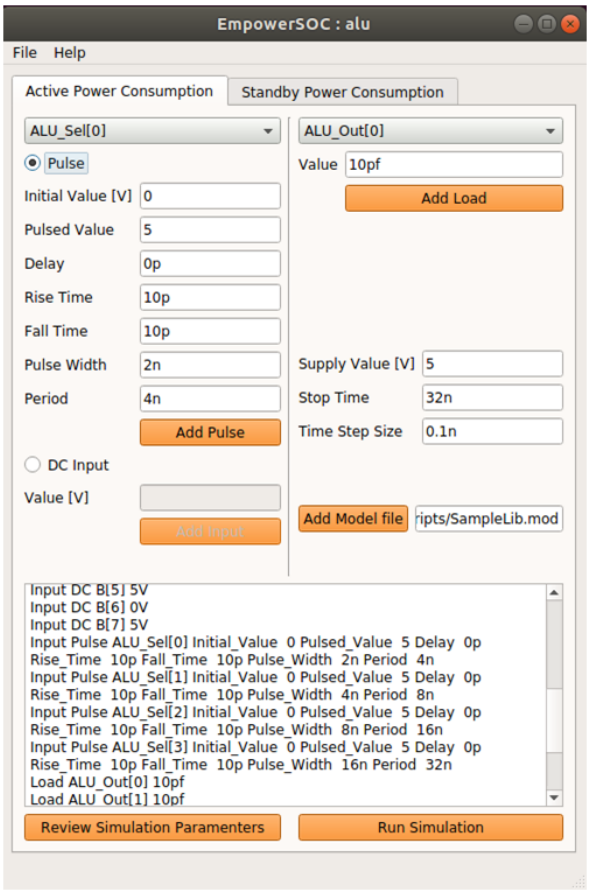

Fig. 3. EmpowerSoC GUI

TABLE I. ACTIVE AND STANDBY POWER CONSUMPTION VALUES FOR VARIOUS MODULES

Module Type	**Gate Count**	**Active Power Consumption**		**Standby Power Consumption**	
		Average Energy(fJ)	**Average Power(μW)**	**Average Energy(fJ)**	**Average Power(μW)**
Inverter	2	1.908×10^{2}	5.490×10^{3}	6.779×10^{-3}	3.389×10^{-4}
3-bit Up-counter	15	6.253×10^{5}	3.126×10^{4}	1.125×10^{-1}	5.625×10^{-3}
LFSR	76	1.519×10^{6}	7.593×10^{4}	5.699×10^{-1}	2.849×10^{-2}
Booth Multiplier	205	2.113×10^{6}	1.057×10^{5}	1.083	5.413×10^{-3}
Jtag	212	1.809×10^{6}	9.048×10^{4}	1.559×10^{4}	7.798×10^{2}
8 bit ALU	855	2.096×10^{6}	6.551×10^{4}	4.721	2.360×10^{-1}
16x16 RAM	1091	6.846×10^{6}	3.423×10^{5}	1.118×10^{1}	5.592×10^{-1}

IV. RESULTS

EmpowerSoC was tested for an 8 bit Arithmetic Logic Unit (ALU) comprising of 855 logic gates and capable of performing 16 possible operations. The RTL description of the ALU was written in Verilog which was then used to run RTL to GDSII flow using Qflow. Fig. 2 shows the obtained layout. Next, EmpowerSoC was invoked which utilized Magic for extracting the parasitics from the layout, creating the final transistor level netlist from the extracted sub-circuits, and finally running the simulation in Ngspice as per the defined constraints. Using EmpowerSoC's GUI (Fig. 3) in active power mode, the ALU was made to perform all the 16 possible operations in a sequential manner for constant operands. This ensured the average active power consumption which was calculated accounted for all the possible use cases. Similarly, using the GUI in standby mode, the ALU was operated in sleep state by keeping the input data pins and select line voltages constant thereby performing no operations. This leads to estimation of average standby power consumption.

In addition to the 8-bit ALU, the tool has been used to calculate active and standby power consumption for various other modules like Joint Test Action Group (JTAG), 32 Byte Random Access Memory, Linear Feedback shift Register (LFSR), and Booth's multiplier with logic gates count up to 1091. Table I shows all the obtained values for gate counts, average energy consumed in femto Joules, and average power consumed in micro Watts for both active and standby modes.

V. CONCLUSION AND FUTURE WORK

This paper introduces EmpowerSoc which is an open source power analysis engine based on the Qflow toolchain. Power estimation is done by extracting transistor-level post-layout netlist for building blocks on an automated basis. The flow is technology independent and the user has the flexibility to select their own technology parameters. Also, the tool can be used to calculate both active and standby power consumption. The tool is user-friendly and is backed by comprehensive documentation that will enable beginners to use the tool easily. Future work includes improving the user interface of the tool so that it becomes more interactive such that users with different degrees of knowledge can adapt to the tool and use it effectively. Also, the tool at present only supports digital circuits. The power estimation of mixed-signal circuits would be soon inculcated in the tool so that many more designs can be supported. Furthermore, in future we wish to expand EmpowerSoC to support OpenLane[10] tool chain based designs in addition to Qflow.

REFERENCES

[1] Qflow : Digital Synthesis Flow. http://opencircuitdesign.com/qflow/

[2] Yosys Open Synthesis Suite. http://bygone.clairexen.net/yosys/

[3] Qrouter. http://opencircuitdesign.com/qrouter/

[4] OpenSTA : Timing Analysis Tool. https://github.com/The-OpenROAD-Project/OpenSTA

[5] Magic : VLSI Layout Tool. http://opencircuitdesign.com/magic/

[6] Weste, N. H. E., Harris, D. M.,Weste, N. H. E. (2005). CMOS VLSI design: A circuits and systems perspective. Boston: Pearson/Addison-Wesley.

[7] OSU Standard Cell Library. https://vlsiarch.ecen.okstate.edu/flows/

[8] Ngspice - Open Source Spice Simulator. http://ngspice.sourceforge.net

[9] JTAG Test Access Port. http://www.opencores.org/projects/jtag/

[10] OpenLane : RTL to GDSII Flow. https://github.com/The-OpenROAD-Project/OpenLane

[11] D. Shah, E. Hung, C. Wolf, S. Bazanski, D. Gisselquist and M. Milanovic, "Yosys+nextpnr: An Open Source Framework from Verilog to Bitstream for Commercial FPGAs," 2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), 2019, pp. 1-4.

[12] T. -W. Huang and M. D. F. Wong, "OpenTimer: A high-performance timing analysis tool," 2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2015, pp. 895-902.

[13] T. -W. Huang, C. -X. Lin, G. Guo and M. D. F. Wong, "INVITED: Essential Building Blocks for Creating an Open-source EDA Project," 2019 56th ACM/IEEE Design Automation Conference (DAC), 2019, pp. 1-4.

[14] M. Chupilko, A. Kamkin and S. Smolov, "Survey of Open-source Flows for Digital Hardware Design," 2021 Ivannikov Memorial Workshop (IVMEM), 2021, pp. 11-16.

[15] Wolf, Clifford, Johann Glaser and Johannes Kepler. "Yosys-A Free Verilog Synthesis Suite." (2013).

[16] O. G. Berkes and R. W. Williams, "Magic as a PC layout tool for small budget VLSI circuit design," in IEEE Transactions on Education, vol. 34, no. 1, pp. 52-55, Feb. 1991.

[17] H. Martin and R. Ram, "A CAD tool for circuit diagram extraction from VLSI layout cells," Proceedings of the 32nd Midwest Symposium on Circuits and Systems,, 1989, pp. 817-820 vol.2.