Acquiring real-time heating of cells in standard cell designs

András Timár, Márta Rencz
Budapest University of Technology and Economics
Department of Electron Devices
Budapest, Hungary 1111
Email: timar|rencz@eet.bme.hu

***Abstract*—In today's digital electronic integrated circuits device heating is one of the most critical issues. Overheating can cause failures in functionality and device malfunction. In certain circumstances overheating of ICs can cause physical destruction of the device itself. This paper introduces a solution to determine cell and gate heating curves across the standard cell IC's surface. The presented methodology and toolset is tightly integrated into standardized logic simulator engines thus providing digital circuit designers a low-level, cell-resolution temperature distribution map during logic simulations. Actual temperatures of each consisting cell of the design can be monitored throughout the whole logic simulation. By being able to monitor temperatures of digital cells during initial simulations, it allows us to detect hot-spots and overheating caused malfunctions far before manufacture. By using the spatial location and temperature magnitude of hot-spots acquired from the presented methodology, place and route (P&R) tools can be driven to change cell placement and routing in order to avoid heating caused failures. Additionally, cooling solutions can be developed using the simulated temperature maps of the IC's surface.**

***Index Terms*—logi-thermal simulation, temperature distribution, hot-spot detection**

I. Introduction

The methodology presented in this paper approaches the ever-growing thermal issues of an integrated circuit from the logic, functional simulation side. This paper demonstrates a tool and a methodology that can shed light on evolving temperatures and possible hot-spots in a standard cell digital integrated circuit. The presented solution treats the standard cells as fundamental elements that produce heat during operation proportional to their switching activity. The greatest peculiarity of the presented application is that the actual temperatures can be continuously monitored during simulation and from the evolving temperature gradients delay and timing alterations can be accomplished.

II. Related work

Krencker et al. introduces a method in [1] where 3D stacked structures are simulated electro-thermally. They use the Verilog-A language to connect the electrical and thermal behavior and determine a temperature map of the 3D IC. The simulator is integrated into the Cadence® framework that allows electrical and thermal simulations can be coupled. In their work they do low-level electro-thermal simulations by generating a variable thermal mesh of the IC. In contrast, our proposed method realizes logi-thermal simulation where digital gates are the lowest-level building block of the design and simulation is done with standard-compilant logic simulators.

Ref. [2] presents a simulator which allows taking the thermal aspects accurately into account through the coupling of an automatically generated thermal network with the electronic circuit schematic. The resulting electro-thermal simulator can be classified as a direct simulator which uses standard CAD tools.

In [3] the thermal behavior of the entire stack is considered by integration of thermal field parameters into a cost function during floorplanning, this means during positioning and interconnecting of all functional blocks. Standard floorplanners are not aware of thermal issues but only try to satisfy timing constraints given by the system, which could create thermal problems. The work of Reitz et al. proposes a method to solve this issue.

Ref. [4] attempts to show that there is a significant peak temperature reduction potential in managing lateral heat spreading through floorplanning. As a demonstration, it uses a wire delay model and floorplanning algorithm based on simulated annealing to present a profile-driven, thermal-aware floorplanning scheme that significantly reduces peak temperature with minimal performance impact that is quite competitive with Dynamic Thermal Management (DTM). The floorplanning tool *HotFloorplan* is part of the *HotSpot* software [5] that is developed at DCS, University of Virginia.

A pre-RTL temperature-aware design methodology is presented in [6], where a fast, yet accurate architectural thermal model that is able to explore large regions of the design space is proposed.

In addition, simulating the self-heating of the circuit in the early phase of the design before manufacture would make cooling issues less problematic. Self-heating simulations may also eliminate the need for design back-annotation after manufacture. An example of temperature-aware ASIC design flow can be found in [7].

III. Solution to hot-spot detection

Our proposed method involves using a logic simulator to simulate the logic behavior of the digital circuit. In our simulations and experiments we used QuestaSim from Mentor Graphics®. As part of our solution we augmented the logic

simulator with a custom C++ PLI [8] code that registers switching activity, reads the actual layout of the circuits and reads the power characteristics of the consisting cells. The HDL source of the synthesized circuit is read into the simulator upon startup. A short testbench was written to create stimuli for the circuit. Our test case was a 4-bit counter circuit synthesized on a TSMC $0.35\mu\mathrm{m}$ technology.

Using the layout of the circuit, the corresponding power characteristics and the switching activity information, the incorporated thermal solver engine calculates cell temperatures during the simulation. The built-in thermal solver engine is derived from Ref. [9] and has been completely rewritten, ported to various platforms (Linux, Unix, Windows) and tightly integrated into the PLI application. In its former version, the thermal solver engine was a standalone program. The new, rewritten version is integrated into the framework as a shared library (.so or .dll), thus providing faster simulations. Further work will be done to eliminate unnecessary I/O overhead that slows down execution. The thermal solver constructs a thermal impedance matrix that contains all the self- and cross-impedances of all the dissipating shapes on the surface. These shapes are the same as the digital cells in the design (see section IV for the actual layout). The resulting data-matrix contains thermal RC-ladders for each cell's self-impedances and cross-impedances to each of the other cells. The RC-ladders' element values are calculated by the thermal solver engine and the underlying topology is the Foster RC-ladder which can be seen in Fig. 1. The initial number of poles in the RC-ladder is 12 but this can also be changed according to the user's need.

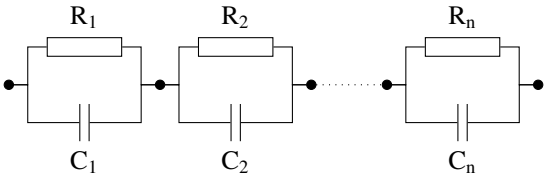

Figure 1. Foster topology of thermal RC ladder

The digital timestep of the circuit (usually the clock signal) and the usual thermal time constants of the circuit differ in more than 9 orders of magnitude. Assuming that the fastest signal (the clock signal) in the digital circuit has a period of 1ns, the thermal time constants of a few $\mu\mathrm{m}^2$ area chip can be in the 1-10 seconds range. Simulating a digital design for a few seconds (in simulation time) with 1ns resolution can take hours to days in realtime. It is not even necessary to run thermal solving in each logic timestep because temperatures barely change in that short period.

In our experiments we used a coarser time resolution for thermal solver runs. The thermal solver engine was invoked in every $1\mu\mathrm{s}$ and this timing resolution proved to be very accurate but simulation time (in real time) has been reduced significantly. The thermal solver timestep can be freely set to other values according to the user's intention. The thermal solver is only called once during the *thermal simulation time window* and therefore simulation time is reduced significantly without sacrificing accuracy.

In the initial phase of the simulation the PLI application sets up its thermal solver engine from the data files pointed to in the Verilog description. After the setup phase, the logic simulator calls the corresponding thermal solver function regularly, once within a thermal time window.

IV. Layout

The layout of the design can be seen in Fig. 2. The logic circuit contains 13 digital cells (multiplexers, xor, xnor gates, D-type flip-flops, etc.) and one feedthru cell. The feedthru cell has been inserted by the Place&Route tool in order to maintain continuity of VDD and GND rails between cells. For the feedthru cell was inserted by the P&R tool *after* synthesis, obviously the synthesized HDL netlist generated by the synthesis tool does not contain such *feedthru* cells because layout placement is not known by that time. Fig. 3 shows the

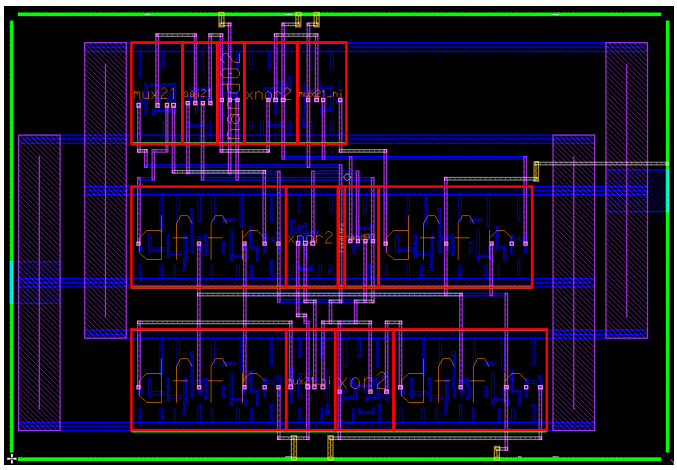

Figure 2. Layout of the synthesized design

schematic layout of the design with the cell instance names for easier identification.

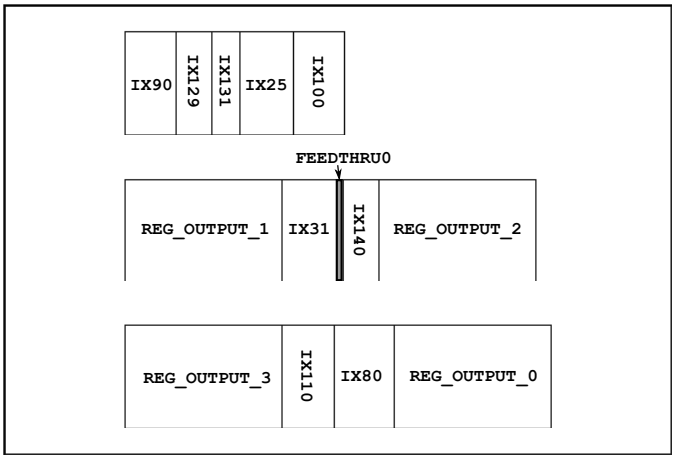

Figure 3. Schematic layout of the synthesized design

V. Verilog extensions

Our proposed method involves using custom Verilog-PLI codes that are written in C/C++ and is responsible for processing power, layout and thermal data inside the logic simulator. After obtaining the synthesized Verilog code from the synthesizer application, a custom testbench is needed to drive the HDL description with a real life stimuli. This testbench includes a few initial setup commands that make the logic simulator cooperate with the third party C/C++ code. Also, the synthesized Verilog description is completed with a few parameters that specifies

- the thermal simulation time window width,
- the layout XML file
- and the power characterization XML file.

A. Input design files

Besides the structural Verilog description of the design our presented application needs to be aware of the actual layout and the power characteristics of the cells. This is accomplished by passing in the required files as XML documents.

During our experiments and development we used *Design Architect IC*™ schematic capture application *LeonardoSpectrum*™ synthesizer and *ICstation*™ layout editor from Mentor Graphics®. ICstation is able to export the custom place & routed layout to GDSII and XML textfile formats. We have choosen to use the XML description of the layout considering its human-readability. Our toolset could be prepared for GDSII, CIF or OASIS layout formats with writing parsers for those formats. The final data that must be extracted from the layout files are eventually the standard cell's boundaries and location.

Listing 1 shows a skeleton layout XML file for our test circuit.

Listing 1. XML layout file

```
<design>
<cell name="dffr"/>
        <boundary>
                <coords>0 0 184000 120000</coords>
        </boundary>
<cell name="mux21"/>
        <boundary>
                ...
        </boundary>
<cell name="oai21"/>
<cell name="nand02"/>
<cell name="mux21_ni"/>
<cell name="xnor2"/>
<cell name="nand03"/>
<cell name="xor2"/>
<cell name="feedthru"/>
<cell name="count4">
        <instance cellname="xor2" inst_name="ix80">
        ...
        <instance cellname="nand03" inst_name="ix140">
        ...
        <instance cellname="dffr" inst_name="reg_output_3">
                <translate>144000 34000</translate>
</design>
```

Using XML descriptions of the layout and the power characteristics proved to be manageable as our third-party C++ application contains a complete XML parser framework and handling XML documents is quite easy.

VI. Results

Fig. 4 shows the temperature functions with respect to simulation time. Each curve stands for one cell in the design including the feedthru cells, if any. A zoomed section can be observed in Fig. 5. The curves run from 0 second to 4 second where the steady state of the temperatures can be observed. Fig. 6 shows the steady-state thermal distribution on the IC

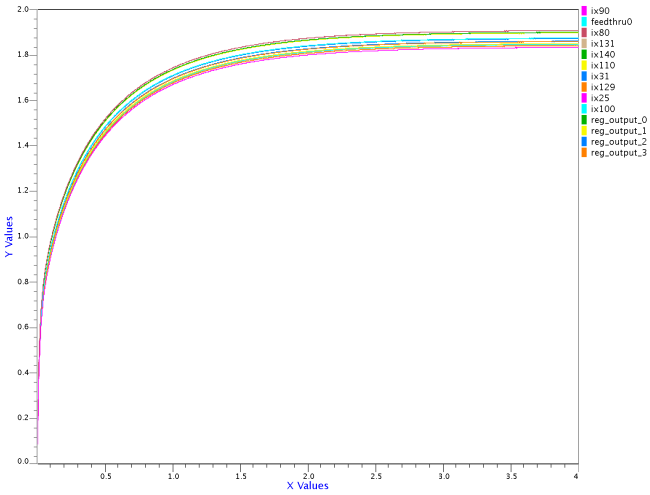

Figure 4. Temperature vs. time functions for every cell

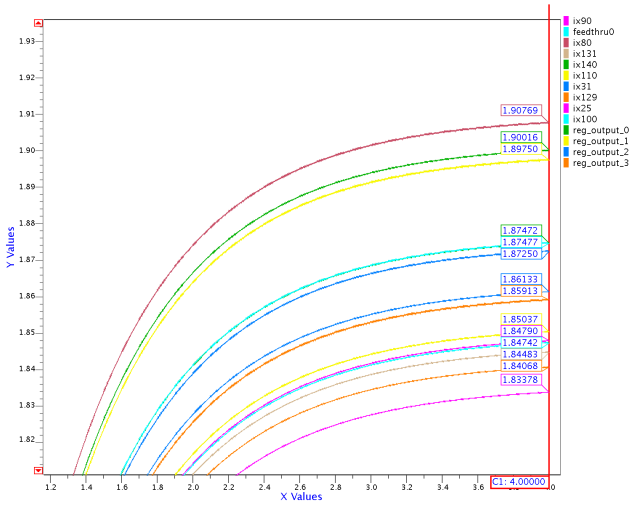

Figure 5. Zoomed temperature functions

surface. It can be observed that the hottest point on the surface is near the least significant bit of the counter (*reg_output_0*) and the consecutive output bits have proportional temperatures. This is in concert with the operation of the counter logic since the LSB bit toggles twice as fast as the next bit and so on.

Table I summarizes the steady-state temperatures and the toggle count as well for every cell in the design. The feedthru

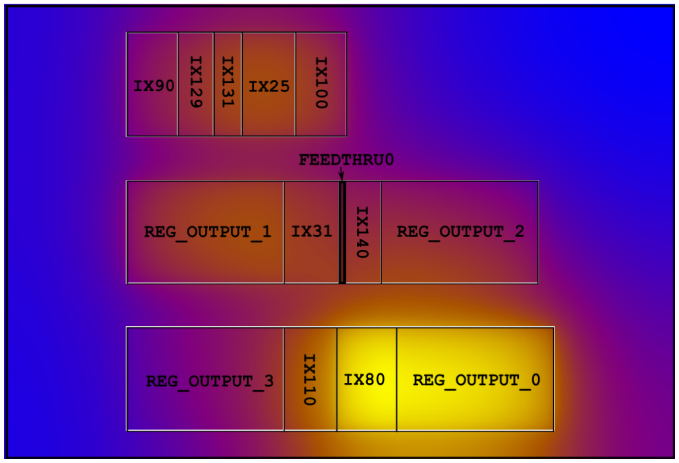

Figure 6. Temperature distribution in steady state

cell's temperature is also monitored altough it does not contain any logic and thus its toggle count is zero.

Toggle counts are relative figures of switching activity during one *thermal timestep*. In the end of the simulation a few files are written that contain the simulation data:

- a text file that contains the simulated temperatures and additional information of each cell for each thermal evaluation:
  - Simulation time
  - Instance name
  - Temperature
  - Power dissipated
  - Toggle count
  - X,Y coordinates of dissipating shape
- A *backward SAIF* [10] file generated by the logic simulator

The Switching Activity Interchange Format (SAIF) is designed to assist in the extraction and storing of the switching activity information generated by EDA (Electronic Design Automation) tools.

A SAIF file containing switching activity information can be generated using an HDL simulator and then the switching activity can be back-annotated into the power analysis/optimization tool. This type of SAIF file is called a backward SAIF file.

The SAIF file contains switching activity information for the whole simulation run in contrast to the 'toggle count' measured in one thermal timestep.

The output file of the simulation was further processed by several scripts in order to generate temperature maps of the surface and produce heating curves in function of simulation time. The temperature functions are written in .csv (Comma Separated Values) file and can be displayed in either as a spreadsheet or with a waveform viewer. Certain *GNUplot* automatization scripts were written in order to automate the display process.

Table I. Steady-state temperatures in the 4th second

Cell type	Instance name	ΔT [$\circ$C]	Toggle count
MUX21	ix90	1.833780	250
OAI21	ix129	1.840678	250
NAND02	ix131	1.844825	250
MUX21_NI	ix100	1.847418	250
XNOR2	ix25	1.847896	250
DFFR	reg_output_1	1.850369	250
DFFR	reg_output_3	1.859125	62
DFFR	reg_output_2	1.861329	125
XNOR2	ix31	1.872500	124
NAND03	ix140	1.874717	124
FEEDTHRU	feedthru0	1.874771	0
MUX21_NI	ix110	1.897503	124
DFFR	reg_output_0	1.900164	500
XOR2	ix80	1.907687	500

VII. Summary

In our presented work a fast and efficient methodology was introduced to overcome thermal heating issues of digital standard cell circuits. With the use of a logic simulator, the power characterized standard cells and the actual layout of the circuit we are able to continuously monitor each cell's temperature and detect evolving hot-spots. Our solution to determining hot-spots uses a self-contained Verilog-PLI application that is called by the logic simulator thus being very fast and efficient. Furthermore, the PLI application is compatible with the IEEE Verilog standard thus can be used by any standard logic simulator engine.

The thermal data acquired from our solution can be used to create temperature maps of IC's surface during regular operation. With the thermal data Place&Route tools can be controlled to change placement schemes in order to eliminate thermal irregularities.

VIII. Acknowledgement

This work was partly supported by the IP 248603 THERMINATOR FW7 project of the European Union and by the Hungarian Government through TÁMOP-4.2.1/B-09/1/KMR-2010-0002.

References

[1] J.-C. Krencker, J.-B. Kammerer, Y. Herve, and L. Hebrard, "3D electro-thermal simulations of analog ICs carried out with standard CAD tools and Verilog-A," in *Proceedings of the 17th Therminic Workshop*, Paris, France, 27–29 September 2011, pp. 19–22.

[2] J.-C. Krencker, J.-B. Kammerer, Y. Herve, and L. Hebrard, "Direct electro-thermal simulation of integrated circuits using standard CAD tools," in *Proceedings of the 16th Therminic Workshop*, Barcelona, Spain, 6–8 October 2010, pp. 61–64.

[3] S. Reitz, A. Heinig, R. Martin, J. Stolle, and A. Wilde, "Thermal modeling of 3D stacks for floorplanning," in *Proceedings of the 17th Therminic Workshop*, Paris, France, 27-29 September 2011, pp. 153–158.

[4] K. Sankaranarayanan, S. Velusamy, M. Stan, and K. Skadron, "A case for thermal-aware floorplanning at the microarchitectural level," *Journal of Instruction-Level Parallelism*, vol. 8, no. 1–16, 2005.

[5] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "Hotspot: A compact thermal modeling methodology for early-stage vlsi design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 5, May 2005.

[6] W. Huang, K. Sankaranarayanan, K. Skadron, R. J. Ribando, and M. R. Stan, "Accurate pre-RTL temperature-aware design using a parameterized, geometric thermal model," *IEEE Transactions on Computers*, vol. 57, no. 8, August 2008.

[7] W. Huang, M. R. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusamy, "Compact thermal modeling for temperature-aware design," *41st Design Automation Conference (DAC), San Diego, CA*, June 2004.

[8] I. C. Society, *IEEE Standard Verilog ® Hardware Description Language*, Design Automation Standards Committee, 28 September 2001, IEEE Standard.

[9] V. Székely, A. Poppe, M. Rencz, M. Rosental, and T. Teszére, "THERMAN: a thermal simulation tool for ic chips, microstructures and PW boards," *Microelectronics Reliability*, vol. 40, no. 3, pp. 517–524, 2000.

[10] Accellera, Ed., *Unified Power Format (UPF) Standard*, 22 February 2007, ch. 7, pp. 59–88.