Limits of gate-level power estimation considering real delay effects and glitches

Michael Meixner, Tobias G. Noll
Chair of Electrical Engineering and Computer Systems, RWTH Aachen University, Germany
{meixner, tgn}@eecs.rwth-aachen.de

***Abstract*—Gate-level power estimation based on foundry-supplied standard cell libraries is a common analysis step during digital design. Surprisingly little is known about the accuracy of this approach and the suitability for different circuit types. At the same time, commercial tools implementing this approach are employed broadly and often regarded as the reference when comparing estimation methodologies on higher levels of abstraction. This work evaluates the suitability and accuracy of gate level power estimators for combinatorial circuits of different logic depths in order to test the ability of handling real gate delay effects. While the basic methodology leads to estimation errors of up to 32 % for the tested circuits, by various improvements in the work flow the accuracy can be improved at the cost of longer runtimes. Apart from recommendations on improving accuracy this work identifies shortcomings in the established approach and highlights circuit characteristics that tend to influence estimation accuracy.**

I. Introduction

Increased transistor density as well as the constantly growing demand for mobile computing performance have made power efficiency and optimization the prime design objective in many applications of SoCs. Depending on the design phase and availability of detailed physical information various methods are routinely applied to estimate the power consumption of circuit implementations ranging from coarse spreadsheet calculations on system level to highly accurate circuit simulations on the physical transitor level. Due to their long runtimes, physical-level circuit simulations are only feasible for selected subcircuits of a complex design. Post-layout power analysis of larger circuits and SoCs is typically performed using established tool suites that work on gate-level netlists and pre-characterized cell libraries.

These estimation tools typically are several orders of magnitude faster than physical-level circuit simulations since at most logic gate-level simulations are required. It is obvious that this speedup will result in decreased accuracy compared to circuit simulations but the individual error for specific circuits is hard to predict. The descriptions of methodologies recommended by the vendors of these tools typically do not give numbers on the reliability to be expected of the power estimations and there are few works that try to validate the results obtained this way. Griffith [1] tried to correlate results from a power estimation of an SoC centered around a LEON processor to silicon measurements and gives recommendations which decreased the initial estimation error of 30 % down to 10 %. Unfortunately, as only the top-level power consumption was analyzed, compensation of errors of individual circuit components is not considered and generalization of the error bound is not possible. Other publications focus solely on problems regarding register transfer level power estimation while using gate-level results generated with the same tools as the reference [2], [3].

In general, power consumption of digital circuits can be separated into static and dynamic power consumption. Estimation of static power consumption due to leakage effects of transistors can be modeled quite accurately using steady state node probabilities that can easily be calculated using zero-delay simulations or static boolean calculations. The dynamic component of power consumption on the other hand is highly dependent on accurate modeling of gate delays in order to capture spurious transitions, known as glitches, that can potentially contribute a significant portion of the power consumption. There has been extensive work on estimation of dynamic energy consumption considering glitches but many publications limit themselves to estimating accurate switching probabilities on gate level (e.g. [4]–[6]). As will be described in the next sections, this switching information could be used as an input to actual power estimation methods but does not guarantee a good match to physical-level circuit simulations.

In this contribution we will evaluate the estimation accuracy of commercial gate-level power estimation suites for several benchmark circuits by comparing the estimation results with SPICE-level circuit simulation. Based on these results we will identify potential error contributions and look for circuit characteristics that influence estimation accuracy. Finally we will recommend mitigation methodologies for selected deficiencies. As the main focus is on dynamic power consumption in order to evaluate the handling of glitching effects, the selected benchmarks are combinatorial circuits featuring varying logic depths and different subsets of logic gates. In the next section we will describe the selected benchmark circuits and highlight their differences. In section III the basic work flow using commercial gate-level estimation tools is outlined. This work flow is subsequently applied to the benchmark circuits in section IV. Based on these results, we identify sources of error and outline approaches to mitigate them in section V.

II. Benchmark circuits

The twelve circuits selected as benchmarks as well as some of their characteristics are listed in Table I. The first ten

978-1-4799-6890-9/14/$31.00

TABLE I
BENCHMARK CIRCUITS

circuit name	number of gates	number of inputs	number of outputs	max. logic level
c432	79	36	7	20
c499	195	41	32	12
c880	164	60	26	18
c1355	195	41	32	12
c1908	185	33	25	24
c2670	248	233	140	19
c3540	446	50	22	30
c5315	601	178	123	20
c6288	608	32	32	72
c7552	697	207	108	29
add8w16	115	128	19	20
mult32x32	2157	64	64	73

circuits are taken from the ISCAS85 benchmark suite. These circuits are purely combinatorial designs that mostly perform arithmetic functions or error detection and correction. Additionally, the circuit *add8w16*, which performs the summation of eight 16-bit words, was included as well as the circuit *mult32x32*, that implements a 32-bit multiplier and forms by far the largest circuit in the benchmark set. By today's standards all benchmark circuits are relatively small but as we are interested in dynamic power consumption, the main challenge is to correctly handle timing effects due to varying delay paths that can only propagate between pipeline registers in larger sequential designs. The reasoning behind the selection of the benchmark set is to model single pipeline stages and identify problems which decrease accuracy. The error bound of estimated power consumption for a large sequential design is then directly correlated with that of its combinatorial sections.

All circuits were synthesized, placed and routed using a commercial 28-nm CMOS standard cell library. The number of gates reported in Table I corresponds to the actual standard cells needed to implement the given function. Due to the optimization during synthesis these numbers can differ significantly from the number of basic gates that appear in the original descriptions of the ISCAS85 circuits. The left-hand side of Fig. 1 shows a breakdown of the cell types used during synthesis for each benchmark circuit. The maximum logic level given in Table I is defined as the maximum number of gates that are connected in series between primary inputs and outputs. It provides some information about the logic depth of each circuit even though the complexity of the individual gates is not considered.

Parasitic information required for simulations and estimations is extracted from the routed layout. The reference results against which the power estimations are compared were generated by simulation of these extracted netlists at SPICE-level accuracy. All results reported in this paper were generated in the slow corner at 0 °C. Since no realistic testbenches for the ISCAS85 circuits are available, all primary inputs are assumed to be uncorrelated with a steady state probability of being in the high state of 50 % and a switching probability of 50 % resulting in switching events in every second cycle on

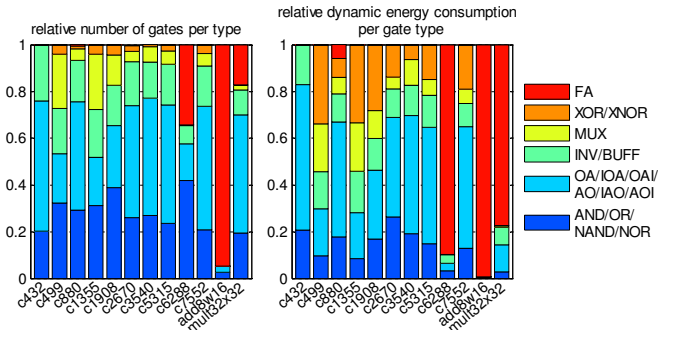


Fig. 1. Cell type composition of synthesized benchmark circuits and simulated energy consumption per cell type.

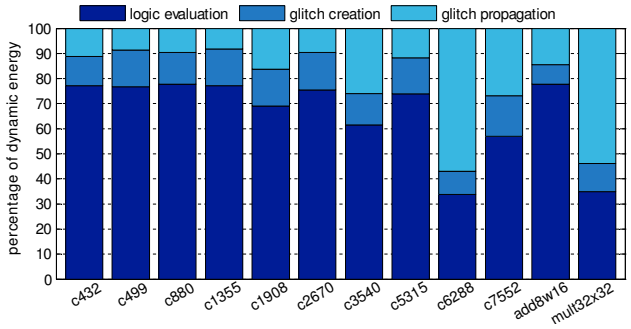


Fig. 2. Dynamic energy consumption of benchmark circuits separated by source.

average. Simulation was performed for 1000 switching cycles for all circuits in order to obtain approximately average energy consumption results.

The individual contribution of different cell types to the overall consumption of dynamic energy is displayed in Fig. 1 next to the relative number of instantiated cell types. Not surprisingly, the simple NAND/NOR-type gates that amount to almost a quarter of the gates used in most circuits consume relatively little energy in many cases. Other more complex gates with few instances in the synthesized circuit can be responsible for a significant fraction of the total energy like the 4-input XNOR gates found in *c499*, *c1355* and *c1908* or the full adders (FA) in *c6288* or *mult32x32*.

In order to confirm the assumption that the selected benchmark set poses a challenge to power estimation methodologies regarding delay effects, a preliminary analysis was conducted to separate the energy consumed by each circuit according to the source of switching. This analysis is based on careful modifications of the circuit simulation but the details are not relevant for the context of this paper. The resulting breakdown is shown in Fig. 2. The major part of energy consumption is caused by the evaluation of the combinatorial logic in most circuits. This corresponds to the minimum energy that would be required for operation of the circuits if all gate inputs would switch simultaneously (i.e. when zero gate delay could be achieved). Misaligned transitions on different inputs of the same gate resulting from unbalanced delay paths lead to the

creation of glitches on the output of gates. This undesired switching can increase energy consumption significantly as shown in the energy breakdown. Finally, once glitches are generated, they cause additional switching activity when they are propagated through downstream gates. This effect is gaining in importance with growing logic depth as can been seen in Fig. 2 (e.g. *c6288*, *mult32x32*).

III. Standard estimation flow

Two industry-standard gate-level power estimation tools, denoted *Tool A* and *Tool B* throughout this paper, will be evaluated in the following section. Both tools are supplements to gate-based statistical timing analysis (STA) tools by the same vendors and feature almost identical work flows.

The minimum inputs required for power estimation are a logic netlist (in Verilog or VHDL) of the design to evaluate and a library that contains pre-characterized data for all standard cells (most commonly in Liberty CCS or NLDM/NLPM format). If no specific values are supplied, some standard switching activity and signal probabilities are assumed at the primary inputs and latch outputs that can be propagated through the logic gates to arrive at toggle counts on all gate inputs. Using this switching information the average power for each cell can be looked up from the library and accumulated to obtain a power estimation for the whole circuit.

This basic estimation flow can be extended in several ways depending on what information is available about the circuit of interest. If available, a significant improvement in accuracy can be achieved by supplying the parasitics of the interconnects between standard cells, which allows accurate calculation of load capacitances. The next important modification is to supply better switching activity information to the estimation tool as the internal propagation mechanisms do not use realistic gate delays. This switching activity is normally supplied in the form of a *Value Change Dump (VCD)* that is generated by a logic simulation of gate level netlist. In order to obtain realistic gate delays for the logic simulation, the STA function of the respective tool suite can be employed to estimate delay paths for all gates in the circuit based on the parasitic information and the timing data from the library file. These gate delays described in *Standard Delay Format (SDF)* can be annotated during logic simulation to generate improved switching information that can finally be used for power estimation. This leads to the work flow displayed in Fig. 3 that corresponds to the methodology recommended by the vendors of the tools for gate-level power estimation.

Apart from the static flow, that uses the supplied VCD file only to extract the toggle count per gate input, both power estimation tools support a dynamic mode that takes the actual switching events and their temporal relationship into account. As the dynamic mode generally produces more accurate results, it will be used in the following analysis whenever the required VCD switching information is available.

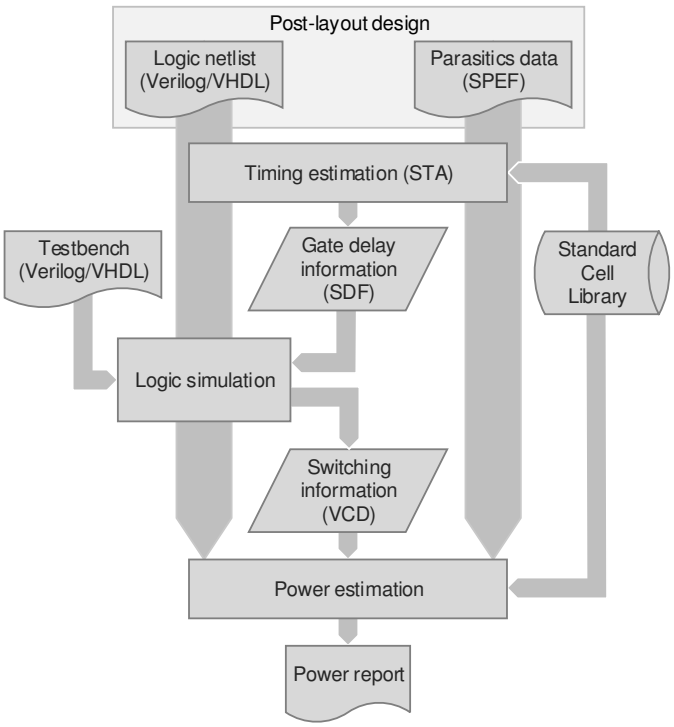
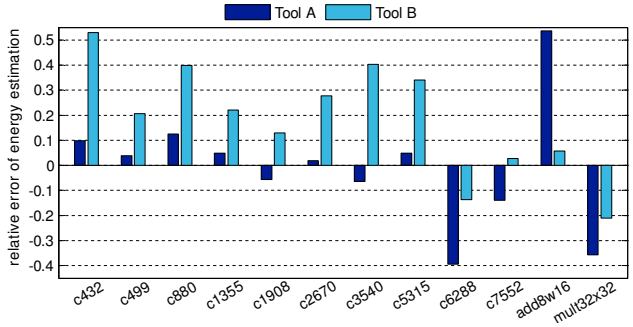


Fig. 3. Standard gate-level power estimation flow

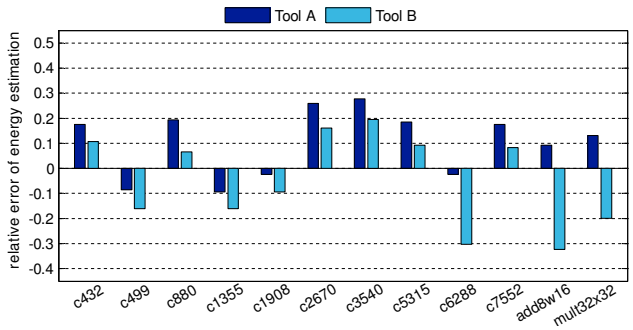
IV. Power estimation results

When applied to the circuits from the selected benchmark set, the power estimation flow described in the previous section leads to relative errors in estimated dynamic energy compared to physical-level circuit simulation as summarized in Fig. 4. The first results, labeled *internal propagation*, do not use switching activity information from VCD files but rely on the statistics propagation engines integrated in the estimation tools. In order to correspond to the circuit simulations, the switching activity was initialized to 0.5 for all primary inputs. The results labeled *SDF annotation* were generated as described in Fig. 3 by annotation of estimated gate delays during gate-level logic simulation in order to arrive at more realistic switching activity data on all circuit nodes. The testbench used during logic simulation is the same that was used to generate the stimuli for the circuit simulation resulting in identical switching sequences.

For the smaller benchmark circuits Tool A's internal estimation is actually closer to the reference results than the recommended methodology using delay-annotated logic simulation. This effect is reversed for more complex circuits where energy consumption due to glitches becomes more pronounced. This is caused by the fact that internal propagation is based on zero-delay gate models with extensions that try to reintroduce some delay variation. In contrast to Tool A, the internal propagation method included in Tool B seems to exaggerate switching activity for the smaller circuits. The large discrepancies between both estimation methods for the circuits *c6288*, *add8w16* and *mult32x32* are caused by a flaw regarding the usage of the



(a) Internal propagation



(b) SDF annotation

Fig. 4. Relative error of dynamic energy estimation using different approaches for generation of switching activity information.

full-adder cell which is employed in the three circuits. This effect will be discussed in detail in section IV-B.

The last noticeable fact in Fig. 4 is that the estimated energy using Tool B is smaller than that using Tool A for all circuits when using SDF annotation. This discrepancy seems to be caused by differences in the calculation of load capacitances of all cells in the netlist even though both tools were supplied with identical input files. The differing assumptions for the load capacitances lead to unequal estimation for gate delay which results in other switching activities reported by logic simulation. Additionally, dynamic energy estimation due to switching of nodes is directly proportional to load capacitance leading to further mismatch of the estimation results.

The mean error is as high as 15.7 % or 16.2 % for estimation using Tool A and Tool B respectively when relying on delay-annotated logic simulation to supply switching activity. More importantly, the error per circuit varies between -32 % and +27 %, so the energy reported cannot even be regarded as a pessimistic estimation.

Fig. 4b is based on energy estimations that combine the energy consumption due to logic evaluation and that due to glitch creation and propagation. As the focus of this paper is on handling of real delay effects, the accuracy of the contribution that is only due to glitch pulses can also be analyzed separately. This is achieved by additionally performing the basic gate-level estimation flow using zero-delay gate models during logic simulation, which results in the energy consumed by pure logic evaluation. The difference of both estimation results,

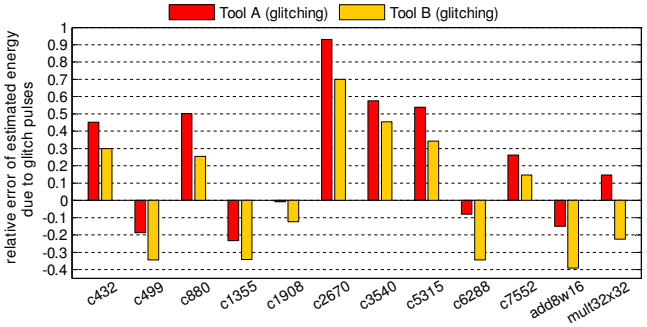


Fig. 5. Relative error of dynamic energy only due to glitch creation and propagation for standard estimation flow using SDF annotation compared to circuit simulation.

which represents the energy contribution caused by creation and propagation of glitches, is compared to the corresponding energy components from circuit simulation in Fig. 2. Because the functional energy consumption due to logic evaluation that is relatively easy to estimate is excluded from this analysis, the relative estimation error increases significantly.

In order to investigate the source of these errors we need to separate the contributions of the different tools employed during estimation.

A. *Errors due to logic simulation*

As the estimation tools basically accumulate energy values related to the switching activity on circuit nodes, the estimation result can only be as accurate as the switching information supplied as an input. As described in section III this switching activity can be estimated using logic simulation of gate-level descriptions with annotated gate delays. Alternatively, probabilistic methods like *tagged probability waveforms* [6] or *transition waveforms* [7], [8] could be employed to calculate the switching activities. As the main benefit of these approaches compared to simulative methods is speed-up and simulation runtime for the selected benchmark circuits is in the order of seconds, these methods are not included in this analysis.

The main contribution of errors during logic simulation is the handling of glitches in the waveforms. The default setting for all common logic simulators which was used so far is to employ inertial gate delays which means that pulses at the output of a gate that are shorter than the gate delay are suppressed while all other pulses are propagated unchanged. This approach prevents excessive switching in logic simulation that would not be physically possible due to limited signal slopes. Unfortunately the default behavior has multiple drawbacks. The first and most obvious one is the fact that short partial-swing glitches that might not propagate through downstream gates but nevertheless contribute to energy consumption [7] are removed from the switching activity output and are therefore not considered during energy estimation. Moreover, this inertial filtering fails completely for complex multi-stage gates like the 4-input XNOR gate shown in Fig. 6 because the total delay depends on the sum of stage

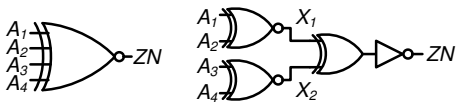


Fig. 6. Symbol and schematic of a 4-input XNOR gate consisting of multiple internal stages.

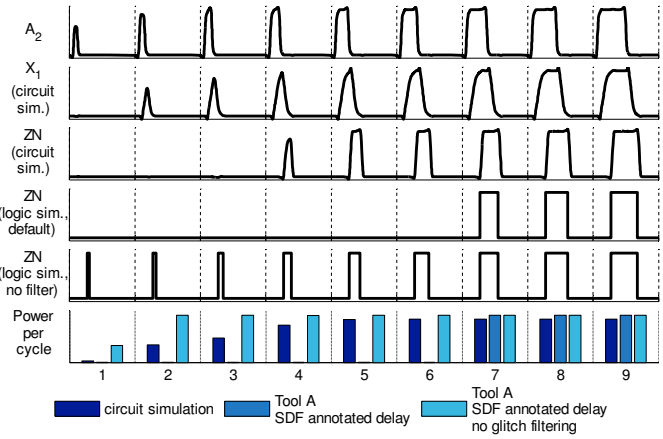


Fig. 7. Estimated glitch energy for propagated pulses on a single input of a 4-input XNOR gate ($A_1 = 1$, $A_3 = A_4 = 0$)

delays while the shortest pulse that can propagate through the whole gate according to the inertial delay model is determined by the delay of the slowest stage.

The estimation tools try to handle the first problem of partial swing transitions by recognizing glitches as pulses that are shorter than the sum of rising and falling transition times and scaling the dynamic power accordingly. For this approach to work, no glitch filtering should be performed during logic simulation. While this can be achieved easily using global pulse control settings of the simulator, the resulting switching information leads to gross overestimations of 31 - 218 % for the selected benchmark circuits. Fig. 7 demonstrates this shortcoming by comparing cycle-accurate estimates of power consumption for a single 4-input XNOR gate when one input features a glitch of increasing pulse width while the other inputs remain constant. While the pulse in the second cycle already appears on the internal node X_1 in physical-level circuit simulation, the forth pulse is the first to propagate to the output ZN. The default logic simulation using inertial delay filtering is clearly rejecting too many valid pulses which leads to underestimation of dynamic energy. Logic simulation without any glitch filtering on the other hand results in overestimation since gate delay is much longer than transition time for this gate.

While for complex gates glitch filtering using inertial gate delays is often too aggressive, for small gates the opposite can be the case. Whereas gate delay is commonly calculated based on intersects of 50 % thresholds, the actual slopes show exponential characteristics due to loading and unloading of the load capacitance. Therefore, the times required for a full signal swing are normally much longer than the gate delays annotated

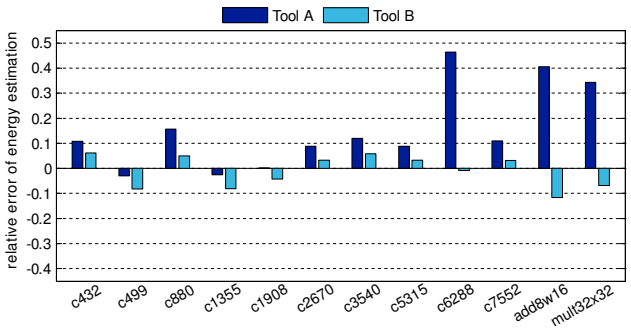


Fig. 8. Relative error of dynamic energy estimation for switching activity information extracted from physical-level circuit simulation.

during logic simulation. Glitches which pass the inertial delay filtering might actually be too small to propagate though the gate.

B. Errors due to energy lookup

While we cannot hope to generate perfect switching activity estimates in a realistic work flow, for evaluation of the tools' accuracy we can supply inputs that match the node activities encountered in the reference circuit simulation exactly. This perfect switching activity replaces the VCD file generated by logic simulation in Fig. 3. Of course, this is no viable way for generating switching activity information in a real-life work flow. If time-consuming physical-level circuit simulation were used to generate switching data, it could just as well record the power consumption itself at the same time. In the course of the analysis this approach ensures that the remaining estimation error is exclusively caused by the lookup and the accumulation of energy values from the characterization library.

The resulting errors plotted in Fig. 8 still show deviations of up to 16 % in energy estimation for the majority of benchmark circuit and even up to 46 % for the circuits containing a significant number of full-adders. These errors can in parts be explained by deficiencies of the Liberty library format, which is the de-facto standard format for timing/power/noise cell libraries. These deficiencies are caused by simplifications that were introduced to reduce characterization effort.

The most fundamental simplification is the inability to characterize power consumption of multiple-input events in the library format [9] which means that no power values are characterized for simultaneous or quasi-simultaneous switching on multiple inputs. This does not reduce accuracy too much for simple gates where switching energy of the output node dominates dynamic energy consumption. However, for complex gates incorporating multiple stages the internal energy contribution can differ significantly between single-input and multi-input events. Directly related to the missing characterization of multi-input events is the inability to accurately model energy consumption of quasi-simultaneous switching events that do not result in switching of the output while the individual events on the inputs would have toggled the output. Both effects are demonstrated in Fig. 9 showing cycle-accurate power

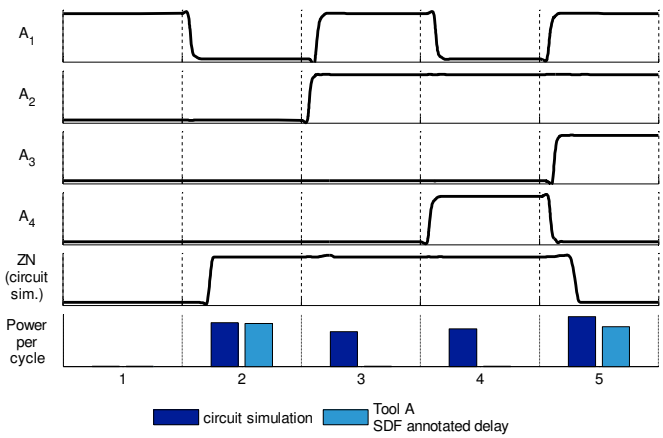


Fig. 9. Estimated glitch energy for multi-event switching of 4-input XNOR gate.

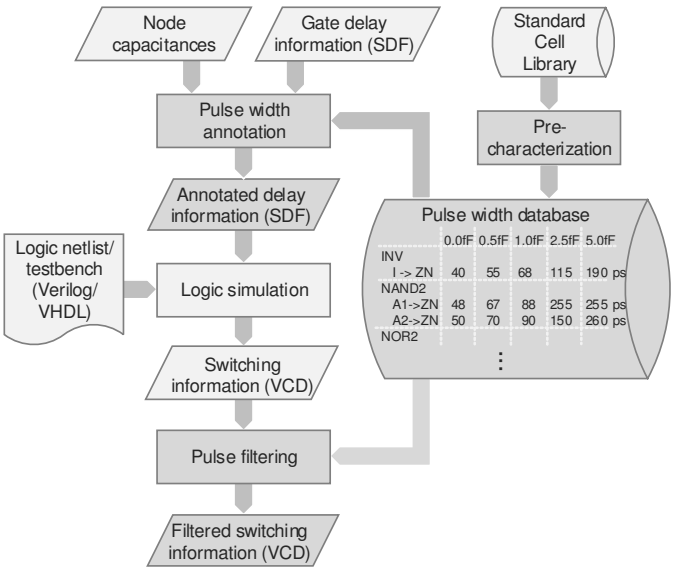


Fig. 10. Pre- and post-processing steps added to logic simulation to improve handling of glitch pulses.

estimations of the 4-input XNOR gate from Fig. 6. The energy consumption of the single-input event in cycle 2 is estimated perfectly whereas energy consumption corresponding to the multi-input event in cycle 5 is underestimated by 20 %. The events in cycles 3 and 4 that involve two inputs each but do not result in switching of the output are erroneously estimated to cause no energy consumption at all.

The last problem that we identified affects only cells with more than one output. In typical standard cell libraries this concerns almost exclusively full- and half-adder cells apart from sequential gates with inverted outputs. As can be seen in Fig. 8 the circuits *c6288*, *add8w16* and *mult32x32* show particularly bad energy estimations when Tool A is used even for perfect switching activity data. This is directly related to the fact that these are the circuits in the benchmark set that contain full-adder cells. Apart from the general problems of complex multi-stage gates described above, the poor estimation accuracy for these circuits ultimately stems from the fact that the foundry-supplied characterization library used during analysis contains separate power tables for both the carry and the sum output for input events that result in toggling of both outputs. As both output signals are closely related and share a large part of the internal interconnects and devices, this has to result in miscalculations during power estimation. As the Liberty library format actually supports look-up tables related to multiple switching outputs, this source of error could easily have been avoided. Interestingly, Tool B seems to handle this flawed characterization library much better.

V. Improvements to the Standard Work Flow

In the previous section, multiple shortcomings of the recommended standard estimation work flow have been identified when dynamic energy estimation of multiple levels of combinatorial logic is important:

- Propagation and filtering of glitch pulses during logic simulation is handled poorly when using default settings:
  - For complex (multi-stage) gates, filtering is too aggressive which results in underestimation of switching energy.
  - For simple gates, filtering is too weak resulting in overestimation of switching activity and energy.
- Library format restrictions result in missing look-up values for certain transitions:
  - Multi-input events that result in switching of the output cannot be characterized.
  - Multi-input events that do not lead to switching of the output cannot be estimated if each single-input event involved would have resulted in toggling of the output.
  - When applicable, cells with multiple outputs should be characterized with shared look-up tables.

While the last point could be fixed by re-characterization of the library if feasible, the problems related to multi-input events cannot be alleviated when relying on the commercial estimation tools. These limitations are inherent parts of the library standard. In contrast, the errors of switching activity due to glitch handling in logic simulations can be mitigated somewhat by careful pre- and post-processing of simulation data.

As described in section IV-A, glitch filtering using inertial delay overestimates or underestimates switching activity at the output of gates depending on the complexity and internal structure of the gate. Pure post-processing of the simulated logic waveforms cannot be the solution as the transitions removed due to excessive filtering are already lost at this stage. Global pulse control settings of the logic simulator are no help either because they only allow filtering to be lowered to a fixed percentage of the total gate delay. This relative setting cannot reflect the varying filtering effects of the individual gates. Instead, it is possible to extend the gate delay information in SDF format that is annotated during logic simulation by pulse

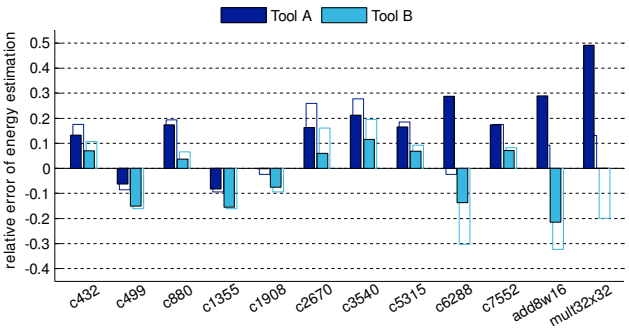


Fig. 11. Estimation error resulting from improved logic simulation based on characterized pulse widths. Empty boxes represent errors of standard estimation approach as shown in Fig. 4b for comparison.

filtering commands (using the *PATHPULSE* keyword) that specify absolute pulse widths. Unfortunately, the maximum pulse width specified this way can only be as long as the gate delay [10]. Therefore, we propose the two-stage approach shown in Fig. 10 that both appends pulse filtering commands to the SDF file and additionally introduces a post-processing step of the switching information in VCD format that is generated during logic simulation in order to handle glitch behavior of simple cells correctly.

The glitch width limits required for filtering are automatically calculated from a database that was created in a pre-characterization process. This characterization database contains the minimum pulse widths at the different inputs of each standard cell that result in a pulse featuring a specific minimum signal swing at the output. As these pulse widths strongly depend on the capacitive load at the output of a gate, the database contains look-up tables characterized for multiple output loads from which the effective pulse widths can be interpolated using the estimated net capacitances calculated during STA.

As shown in Fig. 11, the estimation error is reduced to a maximum of 21 % when excluding *c6288*, *add8w16* and *mult32x32*. Except for these three circuits, estimation accuracy has been improved compared to the results obtained by the standard work flow using unmodified logic simulation that were shown in Fig. 4b. While the circuits *c6288*, *add8w16* and *mult32x32* seem to feature worse estimation accuracy than before, the increasing error can actually be attributed to the flawed characterization of the dominating full-adder cell as described in section IV-B. This corresponds to the high estimation error in Fig. 8 that was based on perfect switching activity information.

The seemingly good estimation accuracy for these circuits in Fig. 4b are actually caused by two different error contributions that have opposite effects and cancel each other out for these specific circuits. The flawed characterization of the full-adder cell would have resulted in overestimation of dynamic energy consumption as demonstrated in Fig. 8. On the other hand, switching activity has been severely underestimated in the unmodified logic simulation of these circuits due to excessive pulse filtering at the output of the full-adder cells in the inertial delay model. As shown in Fig. 2, pulse propagation is the major cause of dynamic energy consumption in *c6288* and *mult32x32*.

The increase in estimation accuracy resulting from the improved handling of glitch pulses comes at the price of growing runtime. In general the speedup of gate-level estimation compared to circuit simulation is higher for large circuits. For *mult32x32*, the largest circuit in the benchmark set, the basic gate-level estimation work flow requires a runtime of around 80 seconds which is approximately 5000 times faster than circuit simulation. The pre- and post-processing steps described in Fig. 10 decrease this speedup significantly. While pulse width annotation to the SDF file is quite fast, the current implementation of pulse filtering of the VCD waveforms increases the overall runtime of the estimation flow by a factor of 30 due to the large number of internal signals in this circuit.

VI. Conclusion

Commercial gate-level estimation tools that have been adopted in many design flows need to be employed with care in order to obtain accurate power estimations. We have demonstrated several deficiencies in the work flow some of which are inherent to the estimation methodology. Other shortcomings that are caused by overly simplified handling of glitch pulses during logic simulation can be mitigated by trading of increased simulation effort for improved estimation accuracy. In general, the higher the logic depth of a circuit, the more errors can be expected due to simplified use of gate delays. Other significant contributions to estimation error stem from missing characterization of quasi-simultaneous switching which is most noticeable in circuits containing complex gates and well-balanced delay paths. Particular care has to be taken when characterizing library cells with multiple outputs in order to avoid huge estimation errors.

References

[1] S. Griffith, “Power Correlation with Silicon - A PrimeTime PX Evaluation,” in *Synopsys User Group San Jose*, 2010.

[2] D. Galbi and K. Kannan, “Measuring Active Power Using PT PX - A User Perspective,” in *Synopsys User Group Boston*, 2010.

[3] S. Patel, B. Patra, S. Rangan, S. Madhava, and A. Bhat, “A novel approach to get optimal power estimate for 45nm large SoC using PTPX Shibashish Patel,” in *Synopsys User Group India*, 2011.

[4] L. Wang, M. Olbrich, E. Barke, T. Buchner, M. Buhler, and P. Panitz, “A theoretical probabilistic simulation framework for dynamic power estimation,” in *Int. Conference on Computer-Aided Design*, Nov. 2011.

[5] J. D. Alexander and V. D. Agrawal, “Algorithms for Estimating Number of Glitches and Dynamic Power in CMOS Circuits with Delay Variations,” in *Symposium on VLSI*, 2009.

[6] F. Hu and V. D. Agrawal, “Enhanced dual-transition probabilistic power estimation with selective supergate analysis,” in *International Conference on Computer Design*, 2005.

[7] Q. Dinh, D. Chen, and M. D. F. Wong, “Dynamic power estimation for deep submicron circuits with process variation,” in *Asia and South Pacific Design Automation Conference*, Jan. 2010.

[8] Z. Hao, R. Shen, S. X.-D. Tan, B. Liu, G. Shi, and Y. Cai, “Statistical full-chip dynamic power estimation considering spatial correlations,” in *International Symposium on Quality Electronic Design*, Mar. 2011.

[9] Synopsys, “CCS Power Liberty Syntax,” 2006.

[10] IEEE, “IEEE Standard for SystemVerilog - Unified Hardware Design, Specifiction, and Verification Language,” IEEE Std 1800-2009, 2009.