An efficient method of leakage optimisation in timing critical designs

Shrikrishna Pundoor
Digital Signal Processing Systems,
Texas Instruments, Bangalore India.
Email:spundoor@ti.com

R.Venkatraman
Digital Signal Processing Systems,
Texas Instruments, Bangalore India.
Email:rvenkat@ti.com

***Abstract*— Power optimisation is one of the most important care-abouts in current-day integrated- circuit (IC) design. The effort spans from design architecture to timing closure and physical design. Use of multiple-threshold transistors in the design cell library (multi-Vt) is one of the common options for power reduction. This paper describes some methods for optimizing leakage power by sizing/swapping across threshold classes of standard cells without impacting the quality of results (QoR) for other design goals. While the intent is to swap existing standard cells in the design with lower-leakage cells across different Vt types, the sequencing algorithms presented here are intended to maximise the number of such swaps, and hence result in optimal leakage power. The algorithms also comprehend timing constraints on the design, and do not worsen the design timing. Three sequencing algorithms are presented in this paper. In this patent pending optimization method, we also explore the option of having transition time limits based on timing slack ranges, and present results from several real-life designs.**

I. Introduction

In the current deep-submicron (DSM) era of IC design, power dissipation has become one of the key care-abouts, much like performance (speed) has been over the generations of chip-design eras. The criticality of power reduction has been accentuated by the ever-growing demand for hand-held battery operated devices, which impose power dissipation constraints from battery-life and heat-generation aspects. Power management in design circuits has been taken care of at various phases of design - during the architecture definition phase, the logic design and implementation phase, and the design optimisation phase.

Power dissipation in digital circuits has two components: active power dissipated when logic ciruits transition from one logic level to another and leakage power, which is largely due to currents through the gate of a device as also the sub-threshold conduction. In this paper, we present methods for leakage power reduction in standard-cell based digital designs, without deteriorating existing design timing and area .

The rest of the paper is organised as follows: in the next section, we highlight prior work in this area. Section-III details the scope for power optimisation in designs implemented using multi-Vt standard-cell libraries. Section-IV explains the optimisation technique used for leakage power reduction. And the subsections continue on explaining the sequencing techniques that have been developed for resulting in optimal design changes. We present some results in subsequent section, and we conclude with some future work in last section, Section-V.

II. Power Optimisation Approaches

There have been several publications in the area of leakage power optimisation, from architecture to implementation, we highlight some of them across different domains of design.

A. Leakage optimisation by input vector control, behavioural synthesis

Traditional methods involve designing sleep circuits. Recent research topics include input vector control optimisation techniques,application of minimum leakage producting input vectors to the sleeping circuits. Leakage power aware behavioural synthesis, and algorithms for gate replacement with input vector simulations are cited in [1]. These are extensively pursued in [2], [3]. Leakage power driven behavioural synthesis of datapaths is discussed in [4].

B. Leakage optimisation by multi-supply, multi-threshold voltages

To reduce total power, lower supply voltages are used. To maintain the desired circuit speeds, threshold of transistors are reduced. Thus, implementation of speed criticial functionality needs either high supply-high threshold voltage islands or low supply-low threshold combination. Optimisation with dual supply, dual threshold transistors have been discussed in [5]. In [5] the proposals for the second-Vdd as well as Vt are discussed in context of variability for a variety of process conditions.

C. Gate length biasing, Sleep transistor insertion, body biasing techniques

Popular methods of tweaking the transistor for better leakage include increasing the gate length of low threshold transistor, body biasing techniques as well sleep transistor techniques.Gate length biasing, sleep transistor insertion and body biasing are to name a few techniques which invlove either circuit topology changes or post-mortem approaches to lower leakage power. Discussions on gate length biasing are in [6],sleep transistor insertion in [7] and body biasing techniques in [8].

III. Multi threshold libraries and optimisation

Semi-custom integrated circuit design involves assembling combination of pre-designed circuits, called standard cells. Each cell performs a specific function and has a defined set of inputs and outputs. By circuit topology and fabrication process techniques for the reduction of threshold voltage, the speed of such a standard cell is optimized. A collection of standard cells for a technology node is called a standard cell library for that node. The lower the threshold, the faster but leakier are the standard cells. A standard cell library can contain multiple classes of cells, having differences in circuit topology as well as threshold values.

A multi-threshold standard cell library contains standard cells with multiple drive strengths as well as thresholds. Out of three leakage currents - p-n junction reverse bias leakage, subthreshold leakage and gate oxide tunneling leakage, sub-threshold leakage is more prominent as technology scales. The subthreshold leakage current is expressed as :

$$I_{sub} = \mu_0 C_{ox} \frac{W}{L} (\frac{kT}{q}) 2e^{\frac{q}{\eta' kT}(V_{gs} - V_t)} (1 - e^{\frac{-qV_{DS}}{kT}})$$

where V_t is the thrshold voltage, kT/q is the thermal voltage, C_{ox} is the gate oxide capacitance per unit area and μ_0 is the zero bias mobility. From this formula, we could deduce that the subthreshold leakage increases exponentially with decrease in V_t and V_{gs}. The drain current of a MOS in saturation region is

$$I_{drain} = \mu_n \frac{C_{ox}}{2} \frac{W}{L} (V_{gs} - V_t)^2$$

where W is width of the channel, L is length. From the equation, the drain current has a quadratic relationship with the threshold voltage. For a given load and supply, The drain current increases as the threshold voltage drops, and hence the ability to drive the load faster. The most popular way so is to use the low threshold transistors in timing critical paths, high threshold transistors otherwise. Hence the standard cell library usually will have mixed threshold class standard cells, and depending on the criticality of timing paths, a mix of threshold classes can be used. Striking a balance between the speed and leakage power using appropriate threshold classes is one of the goals of design closure optimisation.

Usually at physical design implementation, a global slew limit is applied to the entire design. The global slew limit is mainly decided and applied as a avoidance technique to limit the short circuit current and crosstalk victims. But timing critical nets would be normally driven much faster than this slew limit. At the same time, this slew limit is expected to be adhered to for the nets which fall in high slack region. But in reality, we will see scope for optimisation in this region, where slacks are enough and slews are sharper than the expected limits. The picture below (Fig.1) illustrates this, where we have a plot of node slew versus node slack on a real-life design.

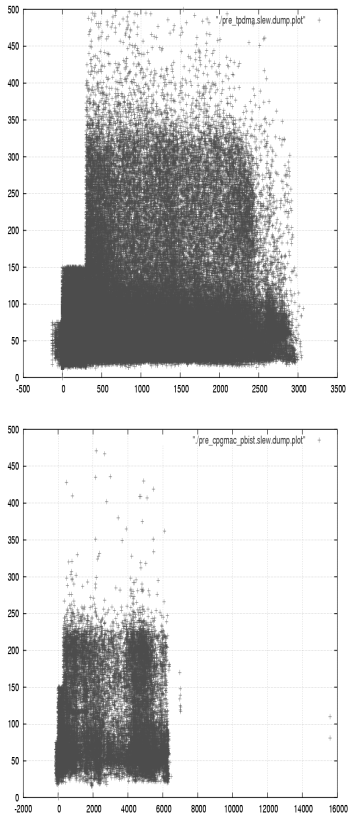


Fig. 1 Node slew rates versus node slacks with global maximum slew constraint of 500 (All measurements in pico seconds)

As we can see, the slews (Y) are tightly maintained even in the high slack (X) region. That means, by swapping or sizing a standard cell in the high slack region and thus shaping the curve, by defining a relationship of slew versus slack, we can optimise the leakage power of the design. This approach also can result in improved design timing and area, as is seen in some of our results presented in a later section.

IV. Optimisation techniques

Considering the dependency between slew and slack,the objective is to swap/size the maximum number of standard cells. This requires structural study of logic cones, the standard cell under consideration can be a parent standard cell in a diverging logic cone. The objective of increasing the number of swaps will be defied thus by a random swap, as the delay propagates through the logic to reach all end points. Apart from this, as a standard cell sizing or swapping affects timing, we have to make sure we update the timing graph, to see the effect of a swap. It would not be a viable solution of updating the timing information for every swap/size operation, as it would be computationally expensive. We developed three sequencing algorithms, namely forward, backward and dependency tracing to help identify the optimal cell candidates for a change. The idea is to increase the number of swaps, perform maximum amount of swaps per timer update and reduce leakage power without sacrificing on existing timing quality of results.

A. Sequencing Algorithms

Sequencing algorithms generate a database from the timing graph of the design. It consists of identifying how many nodes of logic a node would have to traverse reach its timing start point or end point. A timing graph can consists of three types of logic cones, as shown in Fig.2 a diverging cone, a converging and a pipeline (not shown, is just a cascaded buffer structure). A timing graph can have one of these, or may have all of these. Once the sequence database is generated, per cell basis, identification of best sequence method to be used for that cell is made, and then the cell is sized or swapped during the course of implementation. The sequence graphs make sure to consider the stem cells first, to increase the number of sizing/swapping, the fan-out index makes sure right graph is chosen for a standard cell, depending on the logic cone the cell is part of.

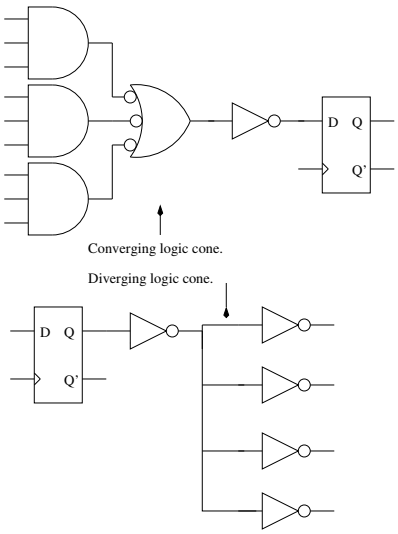


Fig.2 Structure of logic cones.

B. Forward sequencing algorithm

Forward sequencing graph generation involves calculation of worst number of logic stages a node is from its timing start points. Thus a node having a forward sequence number of n means that there are n levels of logic stages from its timing start point, considering all timing arcs through that node. This is depicted in Fig.3.

C. Backward sequencing algorithm

Backward sequencing graph generation is same as forward, except that the number of stages is from the timing end point, instead of start point. Thus a node is at level n means that at worst, the node is n logic stages away from the timing end point. This is depicted in Fig.3.

D. Dependency sequencing algorithm

Dependency sequencing graph generation involves calculation of number of timing arcs passing through a node. Thus a node is at level n means that there are total of n number of timing arcs, and is the dependency metric for the cell associated with the node. This is depicted in Fig.3.

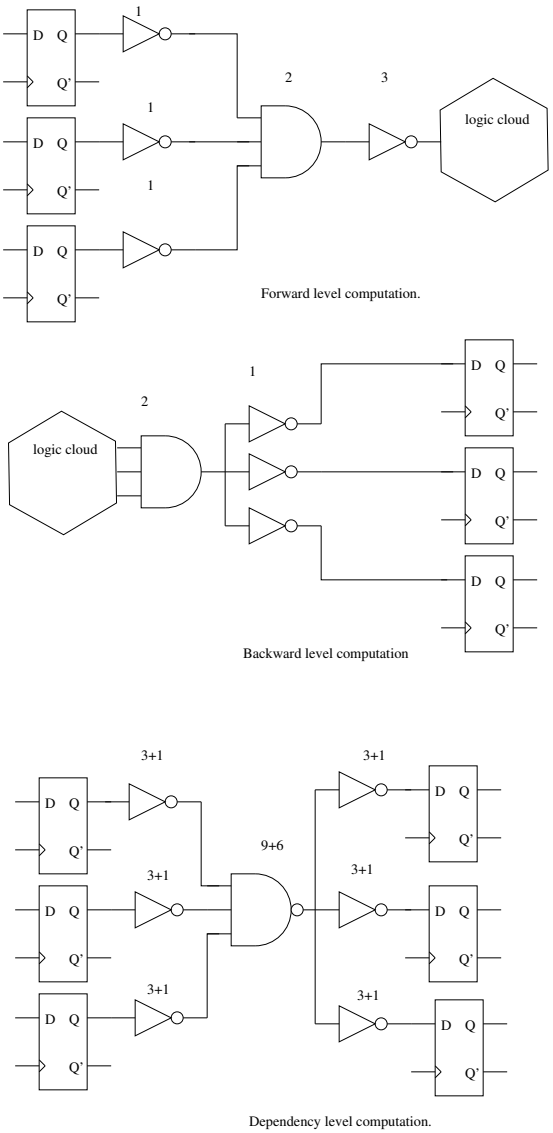


Fig. 3 Forward, backward and dependency level computation depicted respectively.

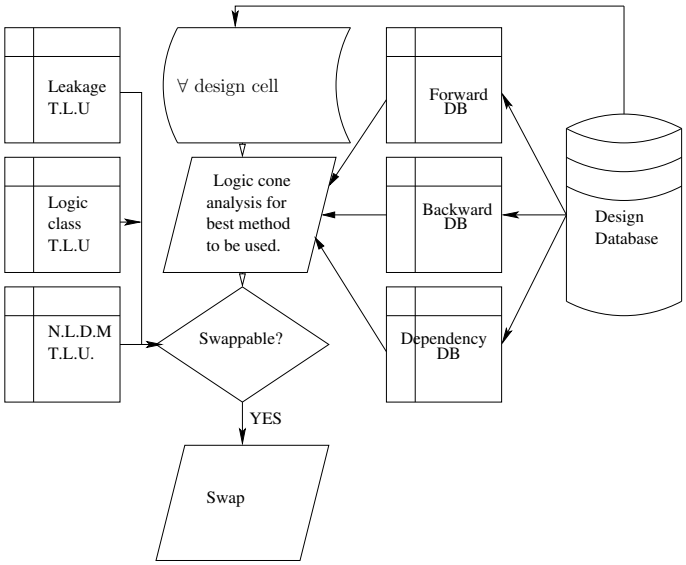
E. Application of algorithms

The algorithms are applied to maximise the number of swaps per timing graph generation. Usually, timing graph re-generation for a large design is very timing consuming, and as already discussed updation of this for every single cell swap is not computationally affordable. Thus, for every stage of the database generated by these sequencing algorithms, basically it is intended to have standard cells from unique timing paths. This is needed as swapping two or more cells in a single timing path is not intended without a timing graph generation after the first swap. The decision of which sequencing database would work best for a given database, infact needs further analysis of logic cones. Discussed below is the some details on implementation of the optimisation process, with data flow chart.

F. Flow chart

The implementation flow chart is depicted in Fig.4. There are static look up tables, which capture library as well as design data. Amongst them, leakage power,area, logic class and delay look up table generation are from library, and the sequence graph database from design data. The procedure looks at every design cell instantiation of a master standard cell, analyzes the best method to use for

swapping. Once method has been identified,every cell in a method is taken and tested for worst timing slack and worst driving slews. If there is a scope for downsizing or Vt swapping, then the worst QoR impact interms of timing, area and power are compared for every standard cell falling in to the same logic class. This process invloves application of realistic input slew rates and output capacitive loads to every standard cell in the same logic class. The outcome can be a single element, or a list. If it is a list, then least leaky standard cell master is picked up for swapping the candidate cell. Thus, leakage power is optimised without affecting the overall timing. It is obvious from circuit design constraints that timing is emphasized over leakage power, and hence the leakage power comparison is pushed down the entire process, thus optimisation in a timing critical/sensitive but closed design.



High level flow chart of the optimisation process.

Fig.3 Flow chart of the leakage optimisation process, implementation.

V. Results

The different sequencing algorithms were tested on several designs in Texas Instruments. We have seen good reduction in leakage power with these approaches, and in some of the designs, the optimisation scheme helped improve timing as well as area results. These results are from 90nm production database. As it is quite obvious from the graphs that in the high slack region the slew rates are better fine-tuned now, and they are well within the maximum slew rate, which is global to the design. Lower down in the slack region (X) there is no change in the density of nodes.Though ideally a timing closed design was expected, to bench mark the algorithms we used them on designs which still have negative slack.

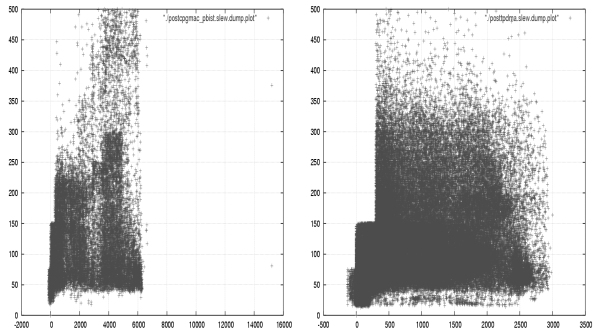


Fig. 4 Slew vs Slack histogram after running leakage optimisation algorithms.(in pico seconds)

A. Results in figure

TABLE I
QUALITY OF RESULTS. DESIGN A.

PARAMETER	PRIOR VALUE	VALUE
Power	170mW	163mW
Area reduction	0	2%
TNS	-2.012n	-1.918n
Number of swaps	-	14196

TABLE II
QUALITY OF RESULTS. DESIGN B.

PARAMETER	PRIOR VALUE	VALUE
Power	2.1W	2W
Area reduction	0	1.1%
TNS	-3.444n	-3.464n
Number of swaps	-	72807

TABLE III
QUALITY OF RESULTS. DESIGN C.

PARAMETER	PRIOR VALUE	VALUE
Power	2.4W	2.3W
Area reduction	0	2.5%
TNS	-22.7n	-15.9n
Number of swaps	-	105400

TABLE IV
QUALITY OF RESULTS. DESIGN D : MULTI-VDD.

PARAMETER	PRIOR VALUE	VALUE
Power	498mW	486mW
Area reduction	0	1.3%
TNS	-192.7n	-192.5n
Number of swaps	-	14757

VI. Future Work and Conclusion

We have proposed a leakage optimisation methodology which is proven to be working well in several designs. This approach looks purely at leakage power reduction incrementally, in a design which otherwise meets other design care-abouts. A similar approach can be used for overall power optimisation. This does not obviate the need for a full-fledged power optimisation as part of logic implementation, but the approach can definitely be used when trying to optimise for design power without impacting other design goals.

VII. Acknowledgment

The authors would like to thank their team members at Texas Instruments for their support in this power optimisation work

REFERENCES

[1] H. J. N. F.N., *A gate-level leakage power reduction method for ultra-low-power CMOS circuits*. Custom Integrated Circuits Conference, 1997., Proceedings of the IEEE 1997, May 1997.

[2] X. C. D. F. Y. H. Z. Zhang and X. Li, *Fast algorithm for leakage power reduction by input vector control*, volume:1 ed. ASIC, 2005. ASICON 2005. 6th International Conferencen, 2005.

[3] A. A. F. Fallah and M. Pedram, *Leakage Current Reduction in CMOS VLSI Circuits by Input Vector Control*, vol. 12,no. 2, ed. IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, February 2004.

[4] R. G. C. Gopalakrishnan and S. Katkoori, *Leakage Power Driven Behavioral Synthesis Of Pipelined Datapaths*, pp-167-172 ed. VLSI, 2005. Proceedings. IEEE Computer Society Annual Symposium, 2005.

[5] A. Oruganti and N. Ranganathan, *Leakage Power Reduction in Dual-Vdd and Dual-Vth Designs through Probabilistic Analysis of Vth Variation*. VLSI Design, 2006. Held jointly with 5th International Conference on Embedded Systems and Design., 19th International Conference, January 2006.

[6] A. S. P. S. D. Gupta, P. Kahng, *Gate-length biasing for runtime-leakage control*, volume 25, issue 8, aug. 2006 page(s):1475 - 1485 ed. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions, August 2006.

[7] B. P. B. L. M. A. and M. E., *Post-Layout Leakage Power Minimization Based on Distributed Sleep Transistor Insertion*, pp 138 - 143. ed. Low Power Electronics and Design, 2004. ISLPED '04. Proceedings of the 2004 International Symposium, August 2004.

[8] K. V. S. A, *Active mode leakage reduction using fine-grained forward body biasing strategy*, pp-150-154. ed. Low Power Electronics and Design, 2004. ISLPED '04. Proceedings of the 2004 International Symposium, August 2004.