Tiago Reimann – Cliff Sze – Ricardo Reis

Gate Sizing and Threshold Voltage Assignment for High Performance Microprocessor Designs

A grain of rice has the price of more than a 100 thousand transistors

Source: The Economist, September 6, 2010

A transistor is cheap

BUT

Energy is expensive

Outline

- Background and Motivation
- Physical Synthesis Flows and Power-driven Gate Sizing
- Timing Quality of Results
- Our Approach
- Motivating Results
- Conclusions

Power Reduction at Physical Level

Gate Sizing

Reduction on the amount of transistors

Power Reduction at Physical Level

Gate Sizing

Continuous Gate Sizing

Needs a Tool for Automatic Layout Generation, like ASTRAN

Discrete Gate Sizing

When using an Standard Cell Approach

Background – UFRGS

Why/When did we start working on discrete gate sizing?

Previous work on continuous (transistor) sizing **(Gracieli Posser)**

ISPD 2012 Gate Sizing Contest (organized by Intel)

Based on work:

M. M. Ozdal, S. Burns, and J. Hu, "Gate sizing and device technology selection algorithms for high-performance industrial designs," in Proc. ICCAD, Nov. 2011.

Simple timing model and only leakage power to stimulate participation

Only lumped capacitance (no wire delay)
Realistic technology library
Design size ranging from 10K to 900K
All designs with zero violation solution

ISPD 2013 Gate Sizing Contest (organized by Intel)

More realistic timing model for wires (RC tree)
More challenging benchmarks

UFRGS – 2012 - Second and First Place Award

Simulated Annealing based method

UFRGS – 2013 - First Place Award

Lagrangian Relaxation based method

ISPD - International Symposium on Physical Design
Discrete Gate Sizing Contest 2012

organized by Intel

Second Place in one ranking (result metric)
First Place in the second ranking (that included running time)

Tiago Reimann, Guilherme Flach, Gracieli Posser
Jozeanne Belomo, Marcelo Johann, Ricardo Reis

ISPD - International Symposium on Physical Design
Discrete Gate Sizing Contest 2013

organized by Intel

First Place in the Primary Metric Ranking

Motivation

Why gate-by-gate heuristics are used early in the optimization flow?

Global algorithms are computationally prohibitive

need to be performed thousands of times.

Simple timing models.

Not possible to use signoff timer early in the flow

slew/capacitance/fanout violations

missing parasitics extraction information, etc.

Hidden library cells with particular threshold voltages.

only the most critical paths can use the low-V_t cell options in late optimization

Motivation

Applying LR-based gate sizing algorithms in a industrial flow

ISPD 2013 Contest winner uses LR-based gate sizing algorithm.

Previous literature works fail to handle two issues in the late physical synthesis stage:

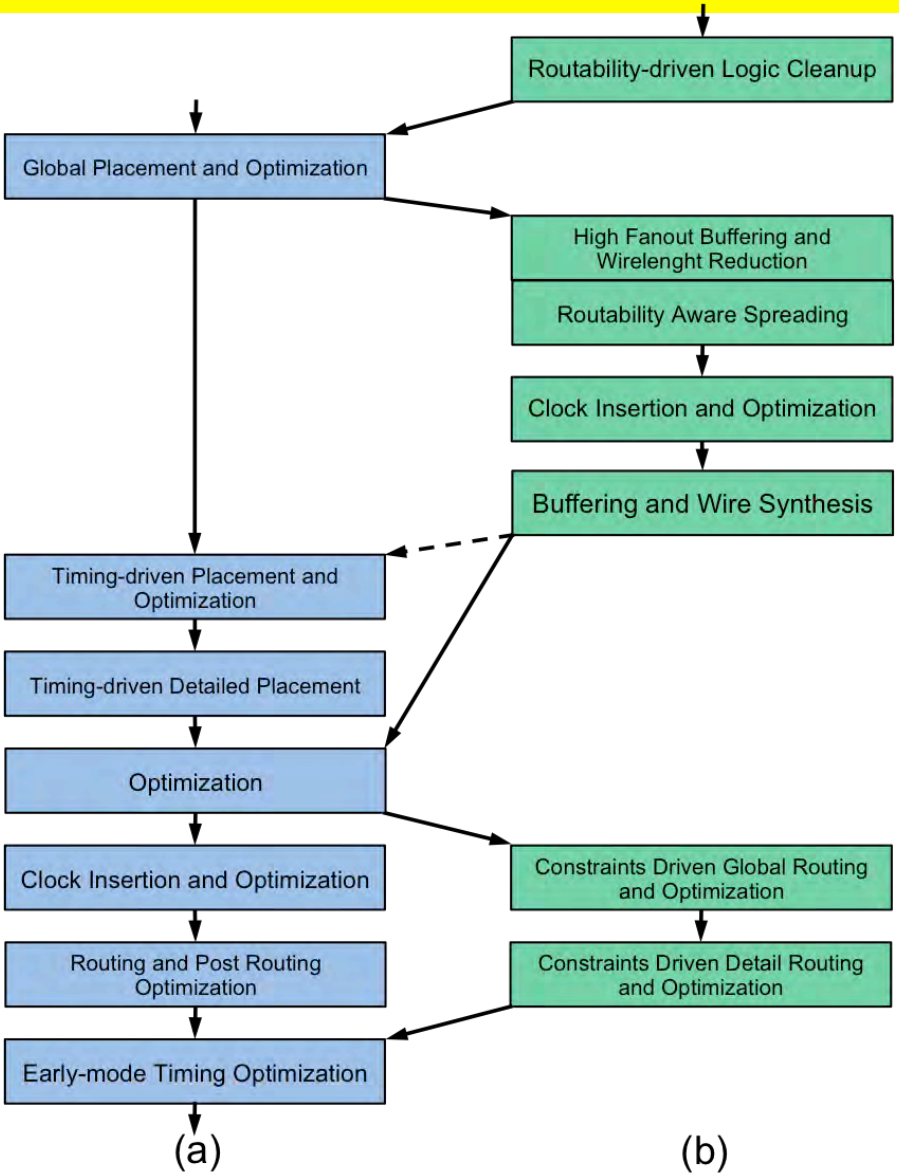
- incremental optimization capability;
- support for different negative-slack constraints.

We focus in the practical challenges of applying LR-based algorithm for power-reduction at the late stage of physical synthesis.

The objective is to minimize both the leakage and dynamic power while making sure that timing is not degraded.

Physical Synthesis Flows

Where global cell selection best fits in the flow?



Power-driven Gate Sizing

Why apply cell selection late in the flow?

LR-based cell selection algorithms require signoff timing engine.

Does not fit in the runtime budget of global and timing-driven placement/optimization steps.

Timing optimization has a higher priority earlier in the flow, and normally power-driven optimization algorithms are applied after timing optimization is converged.

Physical synthesis flows are invoked by tool users and designers in different design stages.

Timing Quality of Results

We have to formulate the problem so that the timing quality of results is not degraded by power minimization algorithms.

Setting the timing constraints to the worst slack for all endpoints is not a good idea

Positive (or less negative) paths will have timing degraded, delivering a wrong perception.

Other flow steps, such as, logic changes, floorplanning updates and other efforts will be made in order to bring the worst slack to zero.

Timing Quality of Results

How can we set timing constraints in designs not closed?

Set the timing constraints of each endpoint to its slack at the end of timing optimization.

Also restricts the TNS (Total Negative Slack).

Timing constraints along side paths (which cannot be observed at any endpoint) will be relaxed leading to timing degradation.

A metric is needed to truly capture the timing quality of results including the non-critical paths with negative slack.

Timing Quality of Results

How can we evaluate timing quality?

Our proposal

True Total Negative Slack (TTNS).

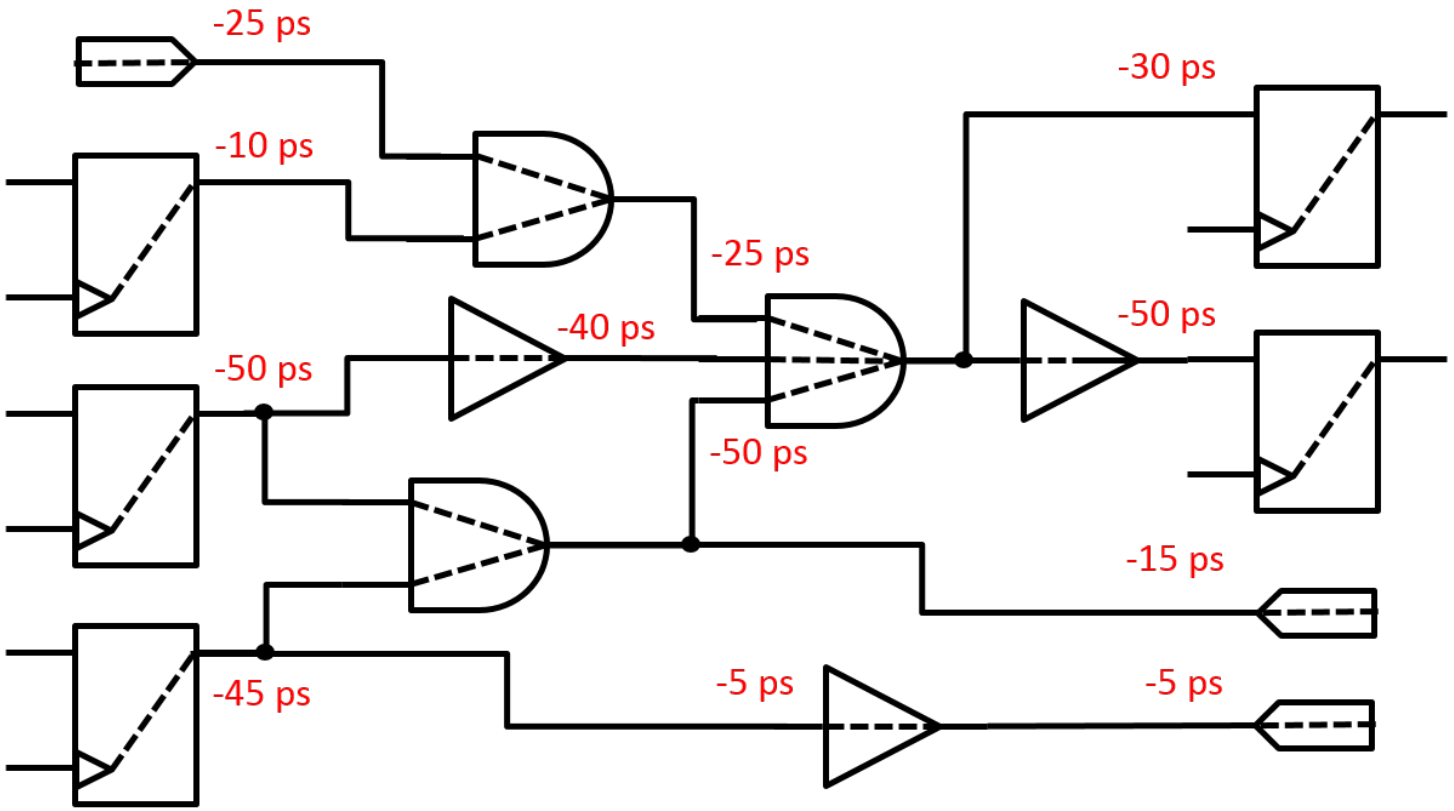
Includes non-critical paths with negative slack into the calculation of total negative slack.

TTNS displays a much better picture of timing quality of results than worst slack and TNS.

TTNS only records one slack value for each subpath

Timing Quality of Results

Example of TTNS:

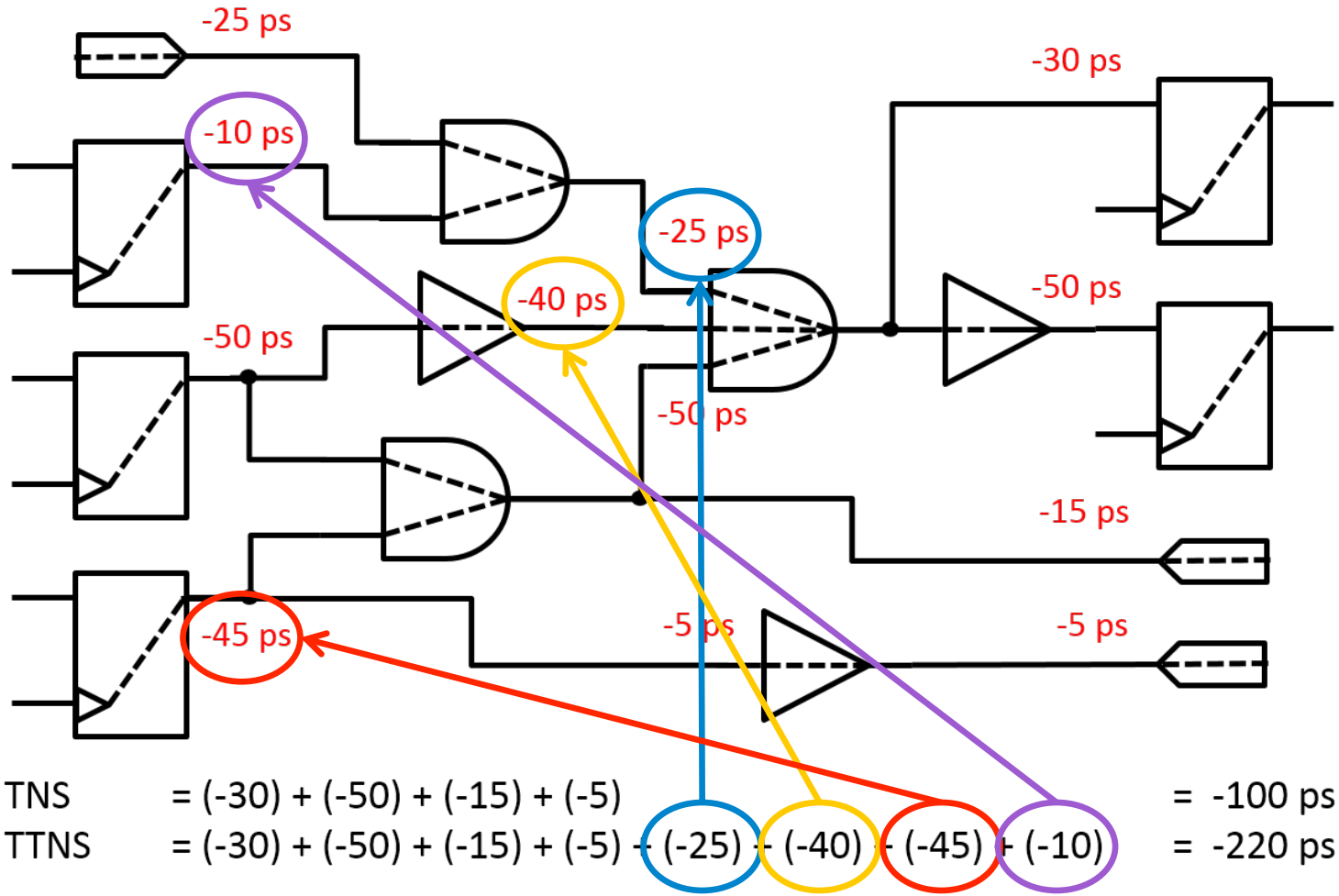


TNS = (-30) + (-50) + (-15) + (-5) = -100 ps

TTNS = (-30) + (-50) + (-15) + (-5) + (-25) + (-40) + (-45) + (-10) = -220 ps

Timing Quality of Results

Example of TTNS:



Our Approach

Applying cell selection algorithm in industrial flow

Algorithm based on the winning team at the ISPD2013 contest.

- LR-based method with greedy local cell selection
- Followed by Timing Recovery and Power Reduction greedy methods

22nm library with core clock period of 174ps

- 2 to 3 V_t levels used
- Around 40 cell library choices in average.

14 high performance microprocessor blocks

- Different characteristics.

G. Flach, T. Reimann, G. Posser, M. Johann, and R. Reis. Effective Method for Simultaneous Gate Sizing And **V**th Assignment using Lagrangian Relaxation, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, April 2014.

Our Approach

Changing the LR formulation to handle negative slacks

Existing timing information is used as constraints of the problem instead of modeling it in the objective function.

Slack for every pin in the design is stored and used as the slack target (instead of zero slack as target)

Modified lambda update aims at preserving the timing of the input state

$$\lambda_{i\to j} \leftarrow \lambda_{i\to j} \times \begin{cases} \left(1 + \frac{a_j - q_j + slk_{init}}{T}\right)^{+1/k} & a_j \geq q_j - slk_{init} \\ \left(1 + \frac{q_j - a_j - slk_{init}}{T}\right)^{-k} & a_j < q_j - slk_{init} \end{cases}$$

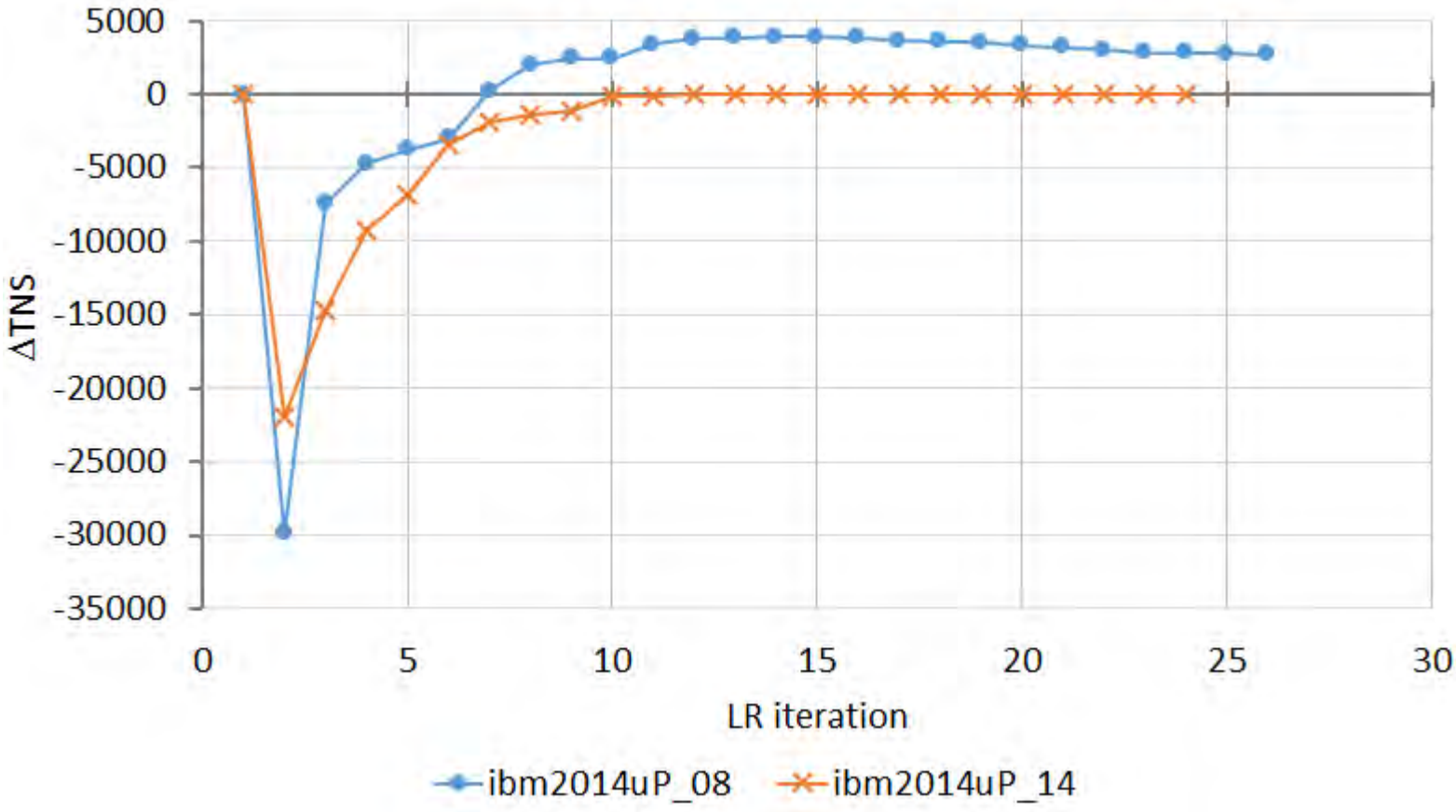
Test Setup

Set of 14 microprocessor blocks

Design	Sizeable Gates	Worst Slack	TNS	Leakage Power	Dynamic Power	Total Power
ibm2014uP_01	88521	-72.57	-138492	79.08	13.54	92.62
ibm2014uP_02	7346	-142.43	-3004	1.07	1.28	2.35
ibm2014uP_03	7940	8.76	-18	2.73	51.37	54.1
ibm2014uP_04	5848	-8.38	-552	1.59	1.35	2.94
ibm2014uP_05	10448	-87.06	-42914	18.92	45.21	64.13
ibm2014uP_06	60768	-142.67	-37685	37.4	112.03	149.43
ibm2014uP_07	63724	-41.5	-52367	60.51	12.6	73.11
ibm2014uP_08	14372	-72.89	-38053	16.29	68.06	84.35
ibm2014uP_09	15270	-30.74	-13234	14.41	33.03	47.44
ibm2014uP_10	117011	-42.43	-65276	85.01	304.35	389.36
ibm2014uP_11	20929	-164.94	-188713	35.97	21.65	57.62
ibm2014uP_12	14529	-421.29	-363751	4.26	20.47	24.73
ibm2014uP_13	16481	-46.36	-27261	19.9	61.22	81.12
ibm2014uP_14	5595	-61.86	-6525	8.18	9.68	17.86

Preliminary Results

Timing convergence w.r.t. LR iterations



Preliminary Results

Design	Worst Slack		TNS		Leakage Power		
	pre-GS	post-GS	pre-GS	post-GS	pre-GS	post-GS	% diff
ibm2014uP_01	-72.57	-72.55	-138492	-133799	79.08	58.75	-25.7%
ibm2014uP_02	-142.43	-141.34	-3004	-2480	1.07	0.99	-7.3%
ibm2014uP_03	8.76	6.99	-18	-14	2.73	2.70	-1.0%
ibm2014uP_04	-8.38	-7.40	-552	-513	1.59	1.58	-0.4%
ibm2014uP_05	-87.06	-86.91	-42914	-42213	18.92	17.53	-7.4%
ibm2014uP_06	-142.67	-142.67	-37685	-34145	37.40	34.64	-7.4%
ibm2014uP_07	-41.50	-41.78	-52367	-50054	60.51	50.57	-16.4%
ibm2014uP_08	-72.89	-72.89	-38053	-33914	16.29	13.68	-16.1%
ibm2014uP_09	-30.74	-30.66	-13234	-12408	14.41	11.55	-19.9%
ibm2014uP_10	-42.43	-41.96	-65276	-62411	85.01	72.85	-14.3%
ibm2014uP_11	-164.94	-164.94	-188713	-182711	35.97	28.98	-19.4%
ibm2014uP_12	-421.29	-421.21	-363751	-352693	4.26	3.74	-12.1%
ibm2014uP_13	-46.36	-46.39	-27261	-24594	19.90	16.69	-16.1%
ibm2014uP_14	-61.86	-60.89	-6525	-6430	8.18	8.12	-0.7%
		Total	-977845	-938379		Average	-11.7%

Preliminary Results

Design	Dynamic Power		TTNS			CPU(s)
	pre-GS	post-GS	pre-GS	post-GS	diff	
ibm2014uP_01	13.54	13.54	-882785	-956814	-74029	37068
ibm2014uP_02	1.28	1.23	-24295	-30079	-5784	3064
ibm2014uP_03	51.37	50.90	-39	-67	-27	3875
ibm2014uP_04	1.35	1.34	-560	-519	41	1021
ibm2014uP_05	45.21	45.15	-77630	-80474	-2845	2548
ibm2014uP_06	112.03	111.97	-64667	-66497	-1830	19120
ibm2014uP_07	12.60	12.60	-390841	-415402	-24561	26886
ibm2014uP_08	68.06	67.29	-201562	-200360	1202	5314
ibm2014uP_09	33.03	33.11	-65844	-77650	-11806	6083
ibm2014uP_10	304.35	295.83	-292779	-306952	-14173	44659
ibm2014uP_11	21.65	21.60	-1020419	-1023425	-3006	6848
ibm2014uP_12	20.47	20.01	-795556	-830203	-34648	5663
ibm2014uP_13	61.22	60.98	-137055	-143574	-6519	5656
ibm2014uP_14	9.68	9.68	-11213	-11134	78	1033
		Total	-3965245	-4143151	-177906	

Preliminary Results

11.7% average leakage power improvement

Up to 25.7% improvement

Improvements obtained after a power-driven flow run

TNS and worst slack show same input state quality or even improvements

TTNS presents significant degradation.

A better formulation for the greedy methods is still needed

Characteristics needed in a power-driven cell selection algorithm

Runtime scalability

Typical runtime of sizing algorithms using signoff timing engines is too long for practical use.

Preserve timing quality of results

It is unacceptable that TTNS gets degraded significantly during power reduction.

Incremental optimization

The algorithm has to be able to recognize the existing cell type and timing quality of results, especially for non-critical subpaths with negative slacks.

Conclusions

Need of new timing-constrained cell selection algorithms for power reduction, where the focus is at the integration into a physical synthesis flow.

Experimental results show promising power saving based on a contest-winning LR-based algorithm. **11.7%** average leakage power improvement with up to 25.7% improvement.

There is much room to improve the power dissipation of our state-of-the-art physical synthesis flow.

We detailed our experience in adopting the ISPD2013 winner algorithm while discussed real concerns and issues which have not been seen in the literature.

Tiago Reimann – Cliff Sze – Ricardo Reis

Gate Sizing and Threshold Voltage Assignment for High Performance Microprocessor Designs