

(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Low Power Multi Bit Flip Flops Design for VLSI Circuits

L.Angel Prabha¹, S.Joy²

M.E Applied Electronics, V.S.B Engineering College, Karur, Tamilnadu, India^{1, 2}

ABSTRACT: In this paper we present a power optimization technique to reduce clock power by using multi bit flip flop method. We have proposed the several techniques to overcome the problems of flip- flops replacement without timing and placement capacity constraints violation. First we perform the co –ordinate transformation to convert the diamond shape legal region into rectangular region and Building the Combination table to identify the Mergeable Flip-flops' Moreover, by judiciously merging and placing the MBFFs, the total wire length is also significantly reduced.

KEYWORDS— Clock power reduction, merging, multi-bit flip-flop, replacement, wirelength.

I. INTRODUCTION

Due to the popularity of portable electronic products, Low power system has attracted more attention in recent years. Reducing the power consumption not only can enhance battery life but also can avoid the overheating problem, which would increase the difficulty of packaging or cooling. Moreover, in modern VLSI designs, power consumed by clocking has taken a major part of the whole design.. Given a design that the locations of the cells have been determined, the power consumed by clocking can be reduced further by replacing several flip-flops with multi-bit flip-flops. During clock tree synthesis, less number of flip-flops means less number of clock sinks. Thus, the resulting clock network would have smaller power consumption and uses less routing resource. Besides, once more smaller flip-flops are replaced by larger multi-bit flip-flops; device variations in the corresponding circuit can be effectively reduced. Fig. 1 shows the clock diagrams of 1- and 2-bit flip-flops. If we replace the two 1-bit flip-flops as shown in Fig. 1(a) by the 2-bit flip-flop as shown in Fig. 1(b), the total power consumption can be reduced because the two 1-bit flip-flops can share the same clock buffer. However, the locations of some flip-flops would be changed after this replacement, and thus the wirelengths of nets connecting pins to a flip-flop are also changed. To avoid violating the timing constraints, we restrict that the wirelengths of nets connecting pins to a flip-flop cannot be longer than specified values after this process. Besides, to guarantee that a new flipflop can be placed within the desired region, we also need to be considering the area capacity of the region. Given a design that the locations of the cells have been determined, the power consumed by clocking can be reduced further by replacing several flip-flops with multi-bit flip-flops. During clock tree synthesis, less number of flip-flops means less number of clock sinks. Thus, the resulting clock network would have smaller power consumption and uses less routing resource.

A.RELATED WORK

They use the graph-based approach. In a graph, each node represents a flip-flop. If two flip-flops can be replaced by a new flip-flop without violating timing and capacity constraints, they build an edge between the corresponding nodes. After the graph is built, the problem of replacement of flip-flops can be solved by finding an m-clique in the graph. The flip-flops corresponding to the nodes in an m-clique can be replaced by an m-bit flip-flop. They use the branch-and-bound and backtracking algorithm to find all m-cliques in a graph. Because one node (flip-flop) may belong to several m-cliques (m-bit flip-flop), they use greedy heuristic algorithm to find the maximum independent set of cliques, which every node only belongs to one clique, while finding m-cliques groups. However, if



(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

some nodes correspond to k-bit flip-flops that k 1, the bit width summation of flip-flops correspond- ing to nodes in an m-clique, j, may not equal m. If the type of a j-bit flip-flop is not supported by the library, it may be time-wasting in finding impossible combinations of flip -flops.

B.OUR CONTRIBUTIONS

The difficulty of this problem has been illustrated in the above descriptions. To deal with this problem, the direct way is to repeatedly search a set of flip-flops that can be replaced by a new multi-bit flip-flop until none can be done. However, as the number of flip-flops in a chip increases dramatically, the complexity would increase exponentially, which makes the method impractical. To handle this problem more efficiently and get better results, we have used the following approaches.

1) To facilitate the identification of mergeable flip-flops, we transform the coordinate system of cells. In this way, the memory used to record the feasible placement region can also be reduced.

2) To avoid wasting time in finding impossible combinations of flip-flops, we first build a combination table before actually merging two flip-flops. For example, if a library only provides three kinds of flip-flops, which are 1-, 2-, and 3-bit, we first separate the flip-flops into three groups. Therefore, the combination of 1- and 3-bit flip-flops is not considered since the library does not provide the type of 4-bit flip-flop.

3) We partition a chip into several sub regions and perform replacement in each sub region to reduce the complexity. However, this method may degrade the solution quality. To resolve the problem, we also use a hierarchical way to enhance the result.

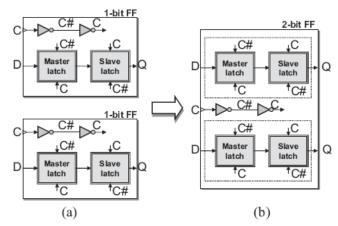


Fig. 1. Example of merging two 1-bit flip-flops into one 2-bit flip-flop. (a) Two 1-bit flip-flops (before merging). (b) 2-bit flip-flop (after merging).

II. PROPOSED METHOD

DESIGN FLOW

Our design flow can be roughly divided into three stages . In the beginning, we have to identify a legal placement region for each flip-flop fi. First, the feasible placement region of a flip-flop associated with different pins are found based on the timing constraints defined on the pins. Then, the legal placement region of the flip-flop fi can be obtained by the overlapped area of these regions. However, because these regions are in the diamond shape, it is not easy to identify the overlapped area. Therefore, the overlapped area can be identified more easily if we can transform the coordinate system of cells to get



(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

rectangular regions. In the second stage, we would like to build a combination table, which defines all possible combinations of flip-flops in order to get a new multi-bit flip-flop provided by the library.

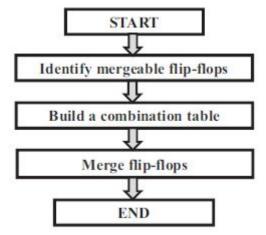


FIG. 2. FLOW CHART OF OUR ALGORITHM

The flip-flops can be merged with the help of the table. After the legal placement regions of flip-flops are found and the combination table is built, we can use them to merge flip-flops. To speed up our program, we will divide a chip into several bins and merge flip-flops in a local bin. However, the flip-flops in different bins may be mergeable. Thus, we have to combine several bins into a larger bin and repeat this step until no flip-flop can be merged anymore.

III. BUILD A COMBINATION TABLE

we may add pseudo types, which denote those flip-flops that are not provided by the library. For example, assume that a library only supports two kinds of flip-flops whose bit widths are 1 and 4, respectively. In order to use a binary tree to denote a combination whose bit width is 4, there must exist flip-flops whose bit widths are 2 and 3.

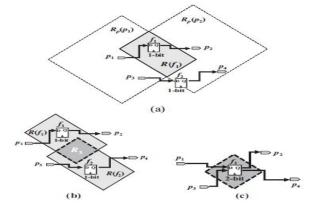


Fig. 3. (a) Feasible regions Rp(p1) and Rp(p2) for pins p1 and p2 which are enclosed by dotted lines, and the legal region R(f1) for f1 which is enclosed by solid lines. (b) Legal placement regions R(f1) and R(f2) for f1 and f2, and the feasible area R3 which is the overlap region of R(f1) and R(f2). (c) New flip-flop f3 that can be used to replace f1 and f2 without violating timing constraints for all pins p1, p2, p3 and p4

Copyright @ IJIRCCE

www.ijircce.com



(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Let *b*max and *b*min denote the maximum and minimum bit width of flip-flops in L. In InsertPseudoType, it inserts all flipflops whose bit widths are larger than *b*min and smaller than *b*max into *L* if they are not provided by *L* originally. After this procedure, all combinations in *L* are sorted according to their bit widths in the ascending order. At present, all combinations are represented by binary trees with 0-level. Thus, we would assign NULL to its right and left child. Finally, for every two kinds of combinations in *T*, we try to combine them to create a new combination. If the new combination is the flip-flop of a feasible type (this can be checked by the function Type Verify), we would add it to the table *T*. In the function Type Verify, we first add the bit widths of the two combinations together and store the result in *b*sum. Then, we will add a new combination *n* to *T* with bit width *b*sum if *L* has such kind of a flip-flop. After these procedures, there may exist some duplicated or unused combinations in *T*. Thus, we have to delete them from the table and the two functions Duplicate Combination Delete and Unused Combination Delete are called for the purpose In DuplicateCombinationDelete, it checks whether the duplicated combinations exist or not. If the duplicated combinations exist, only the one with the smallest height of its corresponding binary tree is left and the others are deleted. In Unused Combination Delete, it checks the combinations whose corresponding type is pseudo in L. If the combination is not included into any other combinations, it will be deleted.

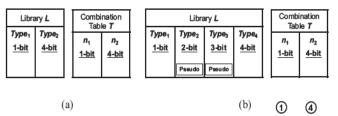
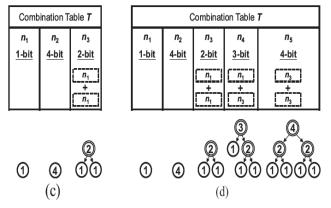


Fig. 4. Example of building the combination table(a) Initialize the library L and the combination table T. (b) Pseudo types are added into L, and the corresponding binary tree is also build for each combination in T.



(c) New combination n3 is obtained from combining two n1s. (d) New combination n4 is obtained from combining n1 and n3, and the combination n5 is obtained from combining two n3s.

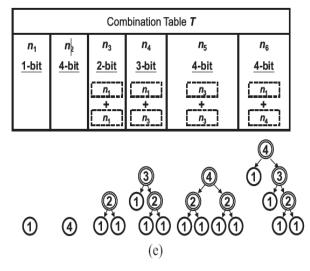


(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14) Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014



⁽e) New combination *n*6 is obtained from combining *n*1 and *n*4.

IV. MERGE FLIP-FLOPS

In merging of flip flops to reduce the complexity, we first divide the whole placement region into several subregions, and use the combination table to replace flip-flops in each subregion. Then, several subregions are combined into a larger subregion and the flip-flops are replaced again so that those flip-flops in the neighboring subregions can be replaced further. Finally, those flip-flops with pseudo types are deleted in the last stage because they are not provided by the library. Region Partition

To speed up our problem, we divide the whole chip into several subregions. By suitable partition, the computation complexity of merging flip-flops can be reduced significantly we divide the region into several subregion, each subregion contains sixbins.bin is a small unit of subregion.

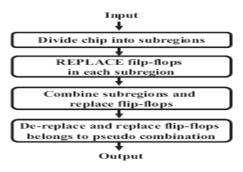


Fig 5 detailed flow to merge flipflops



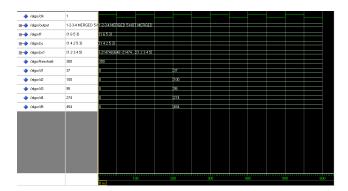
(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014



V. SIMULATION RESULTS Flip flop Merging output

Proposed Power

Tablescore Total sector Total sector <th></th> <th></th> <th></th> <th></th>					
Variation of the second seco	Voltage (V) Current (mA Power (mW) Coint 1.0 Datasetter 18.81 33.85	quiescent numbers in XPower are based on measurements of real designs with active functional elements reflecting real world design scenarios.			
Generation Output Solution Generation Decomposition Organization Organization The Second Part of	2.2 2.3	Power summary:	I(mA)	P(mW)	
Control Control Control	Dynamic 0.00 0.00	Total estimated power consumption:		67	
United Data (Section 2000) United Data (Section 20000) United Data (Section 20000) <	otal Powel 87.46				
Veces3 3.300; 2 7 Due Vame Image: State of the stat	artup Curre 500.00	Vecint 1.80V:	34	61	
Under Series Display	ttery Capacity (mó Hours) 0.00	Veco33 3:30V:	2	7	
Constant Sector Constant Sector					
Data Asses Logr: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 <td>ummary Power b Current b Therma</td> <td>Clocks:</td> <td>13</td> <td>23</td>	ummary Power b Current b Therma	Clocks:	13	23	
Output: Output: Output: Output: Image: Constraint of the second of the sec	8	Inputs:	6	11	
Image: Construction of the second s	Data Views	Logic:	0	0	
a) The Theorem VerseX3 0 0 b) Doppeda Signala 0 0 Figure Verse 0 0 0 Prover Name Ourse cent Vecint LBOY: 15 27 ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id 1/0 Forgers and Rose ANNIO: Prover 1000 - Loval.id <td>Types Types</td> <td>Outputs:</td> <td></td> <td></td>	Types Types	Outputs:			
Constraint of the second	ix 🛄 Inpute	Vcco33	0	0	
Option of the second seco	te 🛄 Logio	Signals:	0	0	
Total View Total View Quescent Vecnt 180V: 15 27 Quescent Vecnt 180V: 2 7 Quescent 180V: 2 7 Qu					
Come Report Come Repo	N Peport Views	Quiexcent Vccint 1.80V:	15	2.7	
AANING FORME 1889 - Invalid I/O Forgram Rode AANING FORME 1889 - Invalid I/O Forgram Rode MANING FORME 1889 - Invalid I/O Forgram Rode Maning Formation - Invalid I/O Forgram Rode Maning Formation - Invalid I/O Forgram Rode Maning Formation - Invalid I/O Forgram Rode Maning Forget - Invalid I/O Forgram Rode Maning Forget - Invalid I/O Forgram Rode Maning Forget - Invalid I/O Forget - Invalid I/O Forget - Invalid Maning Forget - Invalid I/O Forget	 Power Report (HTML) 	Ouiescent Vcco33 3.30V:	2	7	
MANING: Jower: 060 - Invalid I/O Program Mode MANING: Jower: 060 - Invalid I/O Program Mode Mann rank ison - Invalid V or Commence Current differences. This is due to the fact that the subscent numbers in I/Over and seed on measurements of real designs with active monthlong, active Tool Conference of the support active Tool Section 2000 and active monthlong active Tool Conference of the support active Tool Section 2000 and active Destination of the section of the section of the support active Tool Section 2000 and active Destination of the section of the section of the support active Tool Section 2000 active Destination of the section 2000 active Tool Section 2000 active Tool Section 2000 active Tool Section 2000 active Destination 2000 active Tool Section 2000 active Tool Section 2000 active Tool Section 2000 active Destination 2000 active Tool Section 2000 active Tool Section 2000 active Tool Section 2000 active Destination 2000 active Tool Section 2000 active Tool Section 2000 active Tool Section 2000 active Destination 2000 active Tool Section 2000 ac					
functional elements reflecting real world design scenarios. WARNINGIPOURCIFGO - Conju 0% of the register output signals have been set.	WARNING:Power:989 - Invalid I/O Progra WARNING:Power:989 - Invalid I/O Progra XPower and Datasheet may have some Qu	m Mode m Mode .escent Current differences.			
	functional elements reflecting real wo	vid design scenarios.	zal designs with a	ctive	
MARNING: Power: 763 - Only Dt of the design signals toggle.					

Existing Power

Total semantary Total semantary Non 100 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00 1 0.00 0.00 0.00 0.00		76 69 7 23
Image: State of the state o		69 7 23
Normal state Normal state<	Normal Book	7 23
Name O No	Image: Construction Image: Construction Image: Construction Image: Construction	7 23
All from Used of the second seco	All money South of the second state of the sec	23
Terminal and		23
Big (dir. Decision) Big (dir. Decision)<	Big (dr. provide from a first of the second secon	
The second sec	Dispute Dispute 11 Light Image: Stress of the str	
Contract of the second se	Bard Mark Outputs 0 Bard Mark Outputs	
Contract View View View View View View View View	Out Verm Verm3 0 Out Verm3 0 0 Out Verm4 0 0 Out	0
Manual O O Image: Second Sec	Name Name O Or Data O O O Or Data O O O O Or Data O O O O O Or Data O <t< td=""><td></td></t<>	
Addition Device 1000 - Involution 200 - Device Network at Forenames. Name and the set of th	Bigstards O Description Contractions 0 Description	0
Contract View LBOV: 18 Contract View LB	Contraction View 1 100V; 15 Contraction View 1 100V;	0
Outcome for the second secon	Online control Versity 1400V: 15 ••••••••••••••••••••••••••••••••••••	
Output	Contract of the second se	27
Providence of the second state of the second s	Provide and a second seco	
Part Registricity	Paus Registricity	
ANNUEL CONTINUES - LIVELIA US FORMATINA AND AND AND AND AND AND AND AND AND A	SANIHOI, Forme 1000 - 1002114 5/5 Former Mana SANIHOI, Former 1000 - 1002114 5/5 Former Mana SANIHOI, Former 1000 - 1002114 5/5 Former Mana Martin - Andre Former 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 Martin - Andre Former 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 Martin - Andre Former 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 Martin - Andre Former 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 - 1000 Martin - Andre Former 1000 - 1000	
AMMINGIPOWECING - Invalia I/O Feogram Bode Dever and Database have have a new Collegence Courses differences. The four and to have been been been and the second of the second on behaviorements of real designs with active AMMING Power 2010 - Colleging of the register output of the second on been set.	AMPHINO(PSWEL1989 - Invalid I/O Program Hode PDFM and Debmesh may have emmo Greenent Curent differences. This is due to the fact that the quiescent humbers in SPOWEr are based on measurements of real designs with (involtions) elements calledring real vocil design permities. When men, etc.	
AMBINDIPOPERCIPUS - LIVELIA 100 FEOREma Mode Martinologiane and Antonio Characteria Contenti Carente differences. Funct and a lander text for an of characteria content of the stand on measurements of real designs with active AMBINDIPOPERCIPUS - CALLYON of the registeria content of the stand of measurements of real designs with active AMBINDIPOPERCIPUS - CALLYON of the registeria content of the stand of measurements of the standard	CAMPHRONFROWER (909) - Invalid I/O Program Mode CAMPHRONFROME TO A Second Secon	
AANTHOIPower1969 - Invalid I/O Programs Mode Dynams and Decahamst may have manual concerned algorizations. Dynams Langer and the second second second second second second second on measurements of Yeal designs with active functional elements reflecting teal voild design scenarios. MarMING Prower 700 - Only Or of the registers subject actionals have been set.	IANING/Fower:1909 - Invalid I/O Program Mode Fours and Database texp have some Guiescent Current differences. This is due to the fact that the quiescent humbers in Prover are based on measurements of real designs with (IANING/Fower:760 - Childy O' of the registers output signals have been set.	
his is due to the fact that the quiescent numbers in XPower car based on measurements of real designs with active unctional elements reflecting real world design scenarios. JANNING(Power/760 - Only Ok of the registers output signals have been set. JANNING(Power/762 - Only Ok of the design signals bave been set.	his is due to the fact that the quiescent numbers in XPower are based on measurements of real designs with (unctional elements reflecting real world design scenarios, (ANINN):Puwer:760 - Only O's the register subput signals have been set.	
Sunctional elements reflecting real world design scenarios. ANNING hower/780 - Only OK of the register submit signals have been set.	functional elements reflecting real world design scenarios. ARNING/Power/760 - Only 05 of the register output signals have been set.	
MAINING/Power/760 - Only 0% of the register output signals have been set. ANNING/Power/760 - Only 0% of the design signals have been set.	(ABNING: Power: 760 - Only 0% of the register output signals have been set.	GLIVE
ARNING Power 782 - Only 0% of the design signals have been set.		
AADNING:Power:765 - Onlý Ot of the design signals toggle.		
	APNING: Power: 263 - Only 0% of the design signals toggle.	
	72227 Robert and State and Sta	

VI. CONCLUSION

This paper has proposed an algorithm for flip-flop replacement for power reduction in digital integrated circuit design. The procedure of flip-flop replacements is depending on the combination table, which records the relationships among the flip -flop types. The concept of pseudo type is, introduced to help to enumerate all possible combinations in the combination table. By the guidelines of replacements from the combination table, the impossible combinations of flip-flops will not be considered that decreases execution time. Besides power reduction, the



(An ISO 3297: 2007 Certified Organization)

Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

objective of minimizing the total wirelength also is considered to the cost function. The experimental results show that our algorithm can achieve a balance between power reduction and wirelength reduction.

REFERENCES

[1] P. Gronowski, W. J. Bowhill, R. P. Preston, M. K. Gowan, and R. L.Allmon, "High-performance microprocessor design," *IEEE J. Solid-StateCircuits*, vol. 33, , May 1998.

[2]. D. Duarte, V. Narayanan, and M. J. Irwin, "Impact of technology scaling in the clock power," in *Proc. IEEE VLSI Comput. Soc. Annu. Symp.*, Pittsburgh, PA, Apr. 2002, pp. 52–57.

[3] H. Kawagachi and T. Sakurai, "A reduced clock-swing flip-flop (RCSFF) for 63% clock power reduction," in VLSI Circuits Dig. Tech. PapersSymp., Jun. 1997, pp. 97–98.

[4] Y. Cheon, P.-H. Ho, A. B. Kahng, S. Reda, and Q. Wang, "Power-aware placement," in Proc. Design Autom. Conf., Jun. 2005, pp. 795-800.

[5] Y.-T. Chang, C.-C. Hsu, P.-H. Lin, Y.-W. Tsai, and S.-F. Chen, "Post-placement power optimization with multi-bit flip-flops," in Proc.IEEE/ACM Comput.-Aided Design Int. Conf., San Jose, CA, Nov. 201pp. 218–223.