Blockage and Voltage Island-Aware Dual-VDD Buffered Tree Construction

Bruce Tseng
Faraday Technology Cor.
Hsinchu, Taiwan

Hung-Ming Chen
Dept of EE
National Chiao Tung U.
Hsinchu, Taiwan

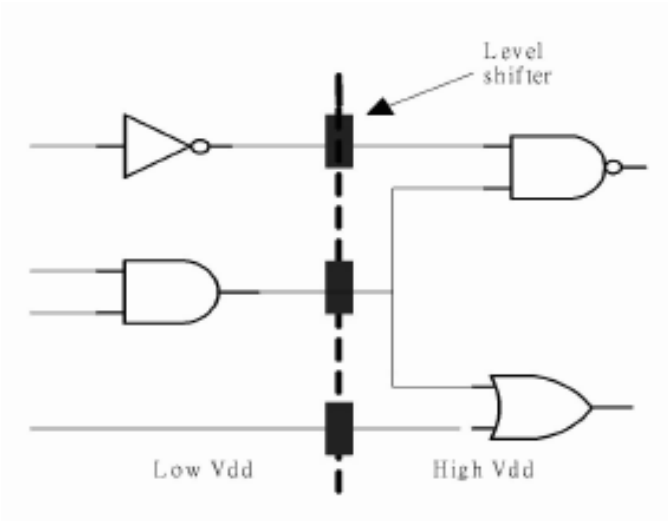
April 14, 2008 ISPD

Outline

- Introduction
- Modified RMP Algorithm
- Voltage Island Aware Buffered Tree Construction (ViaBuf)
- Experimental Results
- Conclusions

Motivation of This Work

- Voltage island architecture is getting popular, however corresponding EDA tools development is still very few.
- We develop approaches to solving the buffer insertion and level converter assignment problem in the presence of voltage island in a low-power design.



Our Contributions

- We have modified the RMP approach[1] so that it can be applied on those designs which contain voltage islands.
- Our method *ViaBuf* has provided massive speedup over modified RMP, and even produced lower power buffered trees.
- As the number of sinks increases, our approach can effectively find feasible solutions within reasonable runtime

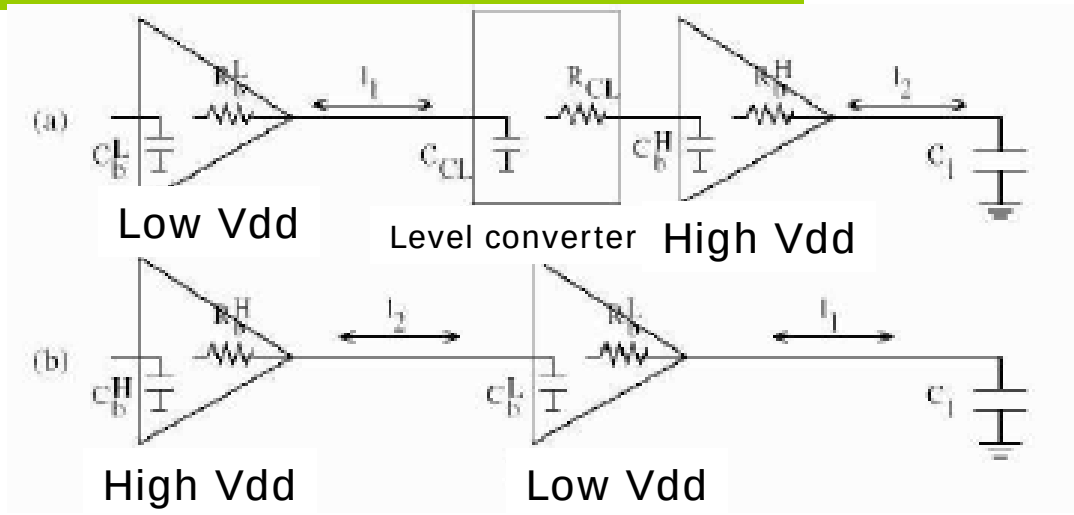
1. K. H. Tam and L. He, “Power optimal dual vdd buffered tree considering buffer stations and blockages” in Proc. of the Design Automation Conf., pp. 497-502, 2005.

Previous Work: DVB Algorithm

- First in depth study on applying dual Vdd buffers in buffer insertion. (DAC'05[1])
  - With restrictions on the ordering of buffers, DVB neglects the necessity of level converter.
  - But DVB can't fit a design with voltage island because of the restrictions.
- DVB is realized on a tree based VG's style buffer insertion and a graph based RMP algorithm.
  - Compared with single voltage, it reduces 18%~26% power consumption.
- With RMP algorithm, DVB uses long time to complete both routing and buffer insertion for a net with less than 10 sinks.

1. K. H. Tam and L. He, "Power optimal dual vdd buffered tree considering buffer stations and blockages" in Proc. of the Design Automation Conf., pp. 497-502, 2005.

Previous Work: DVB Algorithm

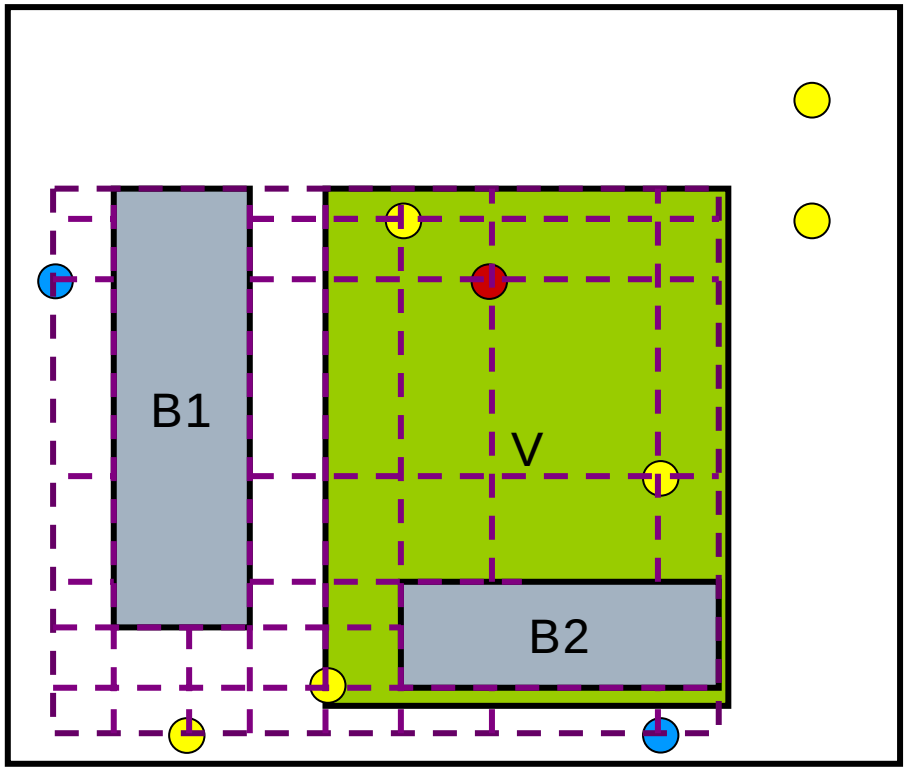


- It is not practical to have no level converters (LCs) presented in the Dual-Vdd designs
  - If C_l is a high Vdd device, we still need LC
  - DVB inserts both kinds of buffers anywhere, which makes P/G routing very difficult

Problem Formulation

- Given a design with voltage island(s), a net with:
  - A source node
  - Multiple sink node with RAT (required arrival time) at each sink
  - Feasible buffer locations
  - Buffer library
  - Wire obstacles (such as hard IPs)
- We want to construct buffered routing tree with buffer insertion and level converter assignment under the following constraints:
  - RAT at each sink should be met.
  - The design works during power saving mode.
  - Signal levels are maintained for all devices.

Modified RMP Algorithm: Routing Grid Construction



Partition the graph into a grid graph by using the vertical and horizontal lines intersect at:

- Source and sink nodes
- Buffer locations
- 4 corners of the wiring blockages

:Source :Sink :Buffer location :Blockage :voltage island

Modified RMP Algorithm: Initial Solution Fill

- There are ten items (cap, rat, pow, rn, rs, B, signalV, Cbl, bend, totLength) in each solution
  1. **cap**: capacitive load
  2. **rat** : require arrival time
  3. **pow**: power consumption
  4. **rn**: reachable nodes (preventing from traversing the same path)
  5. **rs**: reachable sinks (the farthest sink contained in solutions)
  6. **B**: buffer type and corresponding location
  7. **signalV**: signal voltage level
  8. **Cbl**: extra load capacitance that the buffer needs to drive (when solutions merged at buffer location)
  9. **bend**: The accumulated number of bending (solution pruning)
  10. **totLength**: The accumulated wirelength

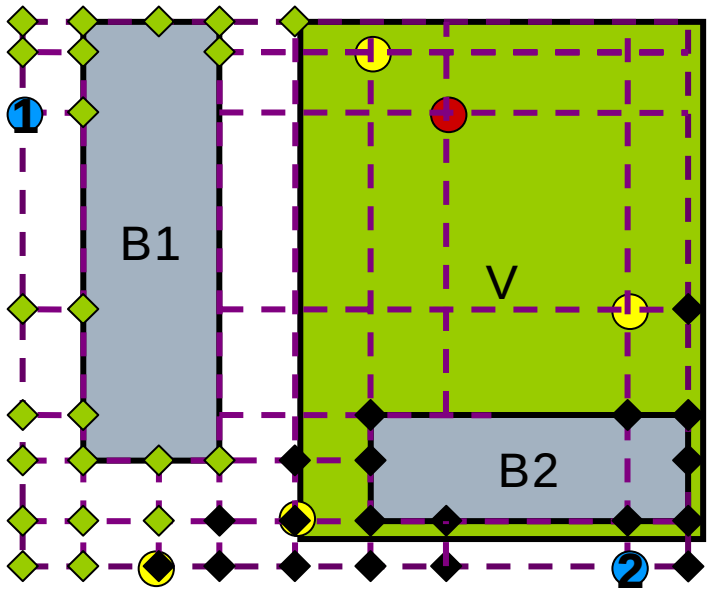
Modified RMP Algorithm: Initial Solution Fill (cont'd)

1. For a sink p, there is only one solution that states a buffer routing tree with zero wirelength.
2. For a source p, there is only one solution that models a driver as a specialized buffer.
3. For other kinds of node p:

   (Assume there are n_H high V_{dd} buffers, n_L low V_{dd} buffers, m voltage level converters)

   a. If it is not a feasible buffer location, there is only one solution.
   b. If it is a feasible buffer location and within voltage island (low V_{dd} region), fill $1+n_L$ solutions.
   c. If it is a feasible buffer location and outside the voltage island, fill $1+n_H+m$ solutions.

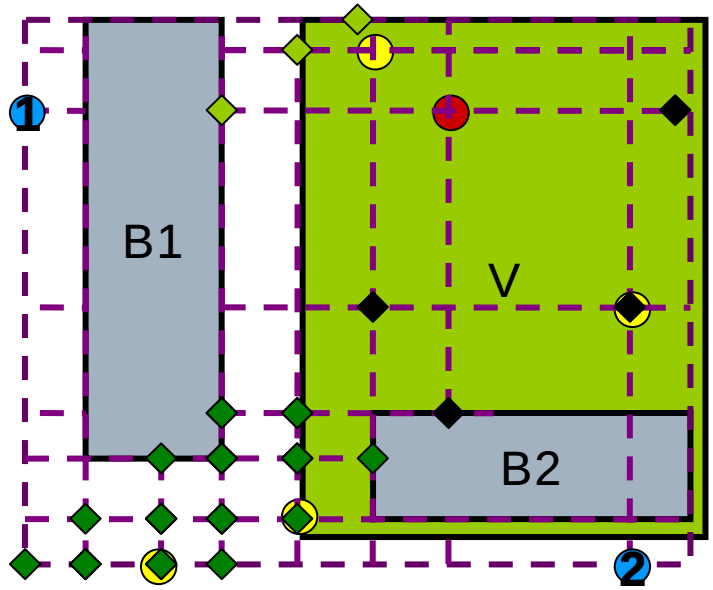
Modified RMP Algorithm: Solution Propagation (1/5)



◆ : solution with rs={1}
◆ : solution with rs={2}
◆ : solution with rs={1,2}

● :Source ● :Sink ● :Buffer feasible location

Modified RMP Algorithm: Solution Propagation (2/5)



: solution with rs={1}

: solution with rs={2}

: solution with rs={1,2}

:Source :Sink :Buffer feasible location

Modified RMP Algorithm: Solution Propagation (3/5)

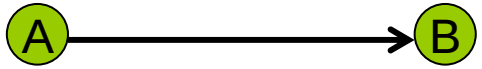
- Use the wave propagation style to propagate the solutions from sink nodes to source node
- Some restrictions:
  1. If both source and sink nodes are out of island, buffer can not be placed within island.

     (in case voltage island turns off)
  2. If ***signalV*** (signal voltage level) is high, low Vdd buffer can not be placed at target node. (otherwise large leakage will occur)
  3. $rn_A \cap rn_B = \emptyset$ (solutions propagating from A to B)

     (to avoid path overlapping)

Modified RMP Algorithm: Solution Propagation (4/5)

- We propagate a solution within node A to its neighbor node B



- If B_B=0, (No buffer placed at node B):

  $cap_{new}= cap_B+cap_A+C_W$

  $rat_{new}= \min(rat_B, rat_A-D_W)$

  $pow_{new}= pow_A+pow_B+E_w$

  $rn_{new}= rn_A \cup rn_B$

  $rs_{new}= rs_A \cup rs_B$

  $B_{new}= B_A \cup B_B$

  $signalV_{new}= signalV_A$

  $CbI_{new}= 0$

  $bend_{new}= bend_A+bend_B+((\text{turn direction})?1:0)$

  $totLength_{new}= totLength_A+totLength_B+(\text{Length between A, B})$

Modified RMP Algorithm: Solution Propagation (5/5)

- If $B_B \neq 0$, (Assume buffer B_B placed at node B)

  cap_{new}= buffer B_B's input capacitance

  rat_{new}= min(D_1, D_2) where $D_1 = rat_B - R_w \cdot (C_w + cap_A)$; $D_2 = rat_A - (D_w + D_B + R_w \cdot Cbl_{new})$

  pow_{new}= pow_A+E_w (Vdd bases on driver)+E_B

  rn_{new}= $rn_A \cup rn_B$

  rs_{new}= $rs_A \cup rs_B$

  B_{new}= B_B

  $signalV_{new}$= (B_B is a level converter) ? low : (V_A || V_B)

  Cbl_{new}=cap_A+C_w+Cbl_B

  $bend_{new}$= $bend_A$+$bend_B$+((turn direction)?1:0)

  $totLength_{new}$= $totLength_A$+$totLength_B$+(Length between A, B)

Modified RMP Algorithm: Solution Pruning

- For two solutions s_A and s_B
  - Prune with VG approach:
    - If $signalV_A = signalV_B$, $pow_A > pow_B$, $cap_A \geq cap_B$, $rat_A \leq rat_B$, then s_A is dominated and can be pruned.
  - Prune with bends and wirelength:
    - If $bend_A > bend_B$, $totLength_A \geq totLength_B$, $rat_A \leq rat_B$, then s_A is dominated and can be pruned

Modified RMP Algorithm: Complexity Analysis

- Almost all the nodes in the graph could be a Steiner point for merging two buffered routing subtree with non-overlap reachable sink

- Assume that a net with n sinks, a grid graph has size M*N and each node has K solutions, then the modified RMP has $O(2^n MNK)$ solutions during propagation, which grows exponentially

Voltage Island Aware Buffered Tree Construction (ViaBuf)

- Perform modified RMP to deal with one sink only during each iteration.
- Erase the useless solutions besides the following solutions:
  - Initial solutions

    (to propagate solution from sink to source)
  - For the node on the desired path, keep solution with
    1. *rs={sinks that were processed}*
    2. *Solutions with different buffer insertion solutions on the desired path.*

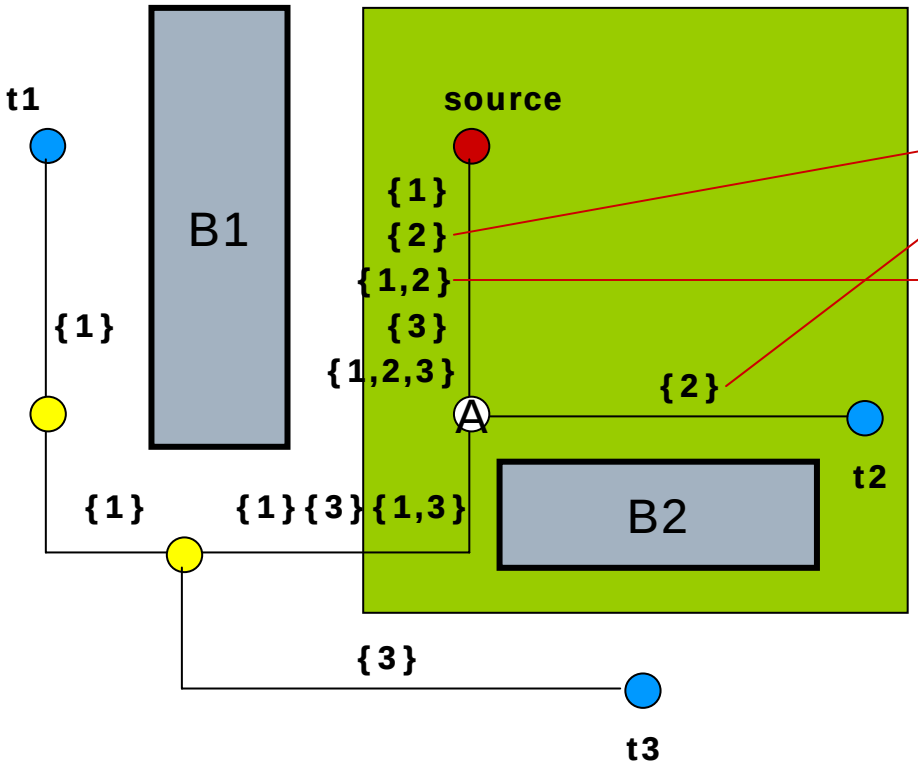
    (useful Steiner points!)

Algorithm Voltage Island Aware Buffered Tree Construction (VIABuf)

Input: A routing grid graph and a wave pool W
Output: Solutions at source node, each one corresponds to a buffered routing tree topology

```
1 While (W is not empty) {
2    get a wave w with sink nearest to source node
3    for each node ni in w {
4       for each solution si in ni {
5          for each node nk which is a neighbor of ni {
6             propagate si to the solutions at neighbor node nk
7             store new generated solutions in temporary container Q
8             prune redundant solutions in Q
9             if Q is not empty {
10               store new generated solution from Q to nk
11               put nk to a temporary wave wtemp
12   }}}}
13   if wtemp contains source node {
14      choose a desired solution with least power consumption
15      erase useless solutions in the routing grid graph
16   } else {
17      W = W [ wtemp
18   }
19 }
```

ViaBuf (cont.)



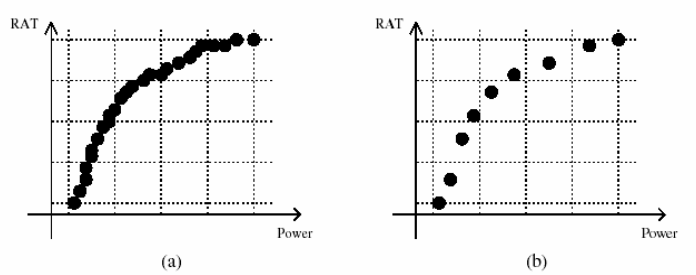
- Keep the following solutions in our approach:

rs={sinks that were processed}

Solutions can be used when the path is possibly shared by handling next sink.

- Besides the above solutions and initial solutions, all the solutions of each node on the grid graph can be pruned.

Comparisons Between Approaches

Differences / Algorithms	Key steps	Solutions keeping and pruning
RMP	Pop solution with maximum RAT during each iteration	1. Keep exact one solution with the smallest cap for each reachable sink set
DVB	The same as RMP	1. Solution sampling. 2. Store solutions with a balanced tree. 
Modified RMP	Classify solutions with the same reachable sink set as a wave, pop a wave during each iteration.	1. Prune with bends
ViaBuf	The same as modified RMP	1. Prune with bends 2. Greedy heuristic

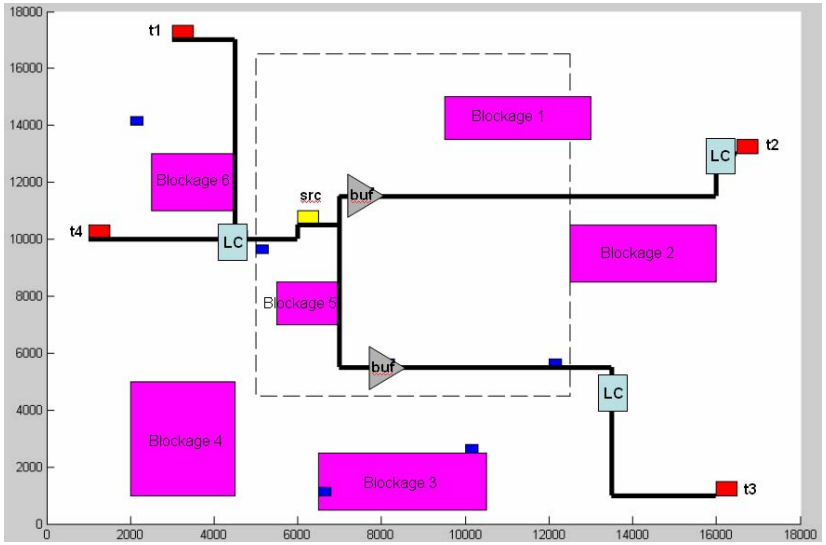
Experimental Results

- Each of these cases has the 6 obstacles, 1 voltage island, 10 buffer locations, and grid graph is about 25*25 nodes on a 17*17mm design.
- A massive speed up over modified RMP could be obtained, while RATs are met
- Our approach also achieves lower power with slightly worse phase delay.

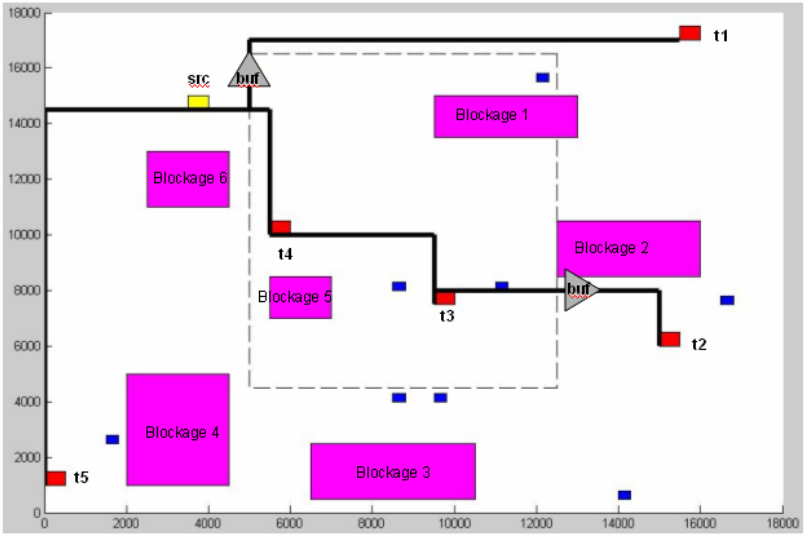
Benchmarks	source out of voltage island	modified RMP			ViaBuf		
		delay(ps)	power(fJ)	CPU time(sec)	delay(ps)	power(fJ)	CPU time(sec)
net4	no	1162	9253	66.7	1200	9253	0.1(606X)
net5	no	903	8918	420.6	982	9328	0.1(3500X)
net6	yes	1199	12504	342.1	1199	9003	1.2(282X)
net10	no	-	-	>6hr.	1306	13957	281
net15	yes	-	-	>6hr.	1631	16882	18.3

Experimental Results (cont'd)

- Instead of MRST, our algorithm intends to find a buffered routing tree meeting timing requirement and also signal integrity.



Source is within voltage island



Source is outside voltage island

Conclusions

- We have implemented modified RMP algorithm to deal with the designs in the presence of voltage island
-
- *ViaBuf* is much faster than modified RMP algorithm and can deal with multiple sinks net as the number of sinks increases
- With RAT constraints, we can produce lower power buffered routing tree suitable for voltage island designs

Thank you!

References (cont.)

[1] C. J. Alpert, A. Devgan, and S.T. Quay, “Buffer insertion with accurate gate and interconnect delay computation” in Proc. of the Design Automation Conf., pp. 479-484, 1999.

[2] W. Chen, M. Pedram, and P. Buch, “Buffered routing tree construction under buffer placement blockages” Asia and South Pacific Design Automation Conference (ASP-DAC), page 381, 2002

[3] J. Cong and X. Yuan, “Routing tree construction under fixed buffer location” in Proc. of the Design Automation Conference, pp. 379-384, 2000.

[4] A. Dechu, C. Shen, and C. Chu, “An efficient routing tree construction algorithm with buffer insertion, wire sizing, and obstacle considerations” IEEE Trans. On Computer-Aidede Design, vol. 24, no. 4, April 2005, pp. 600-608.

[5] van Ginneken, “Buffer placement in distributed RC-tree networks for minimal Elmore delay” in Proc. of IEEE Int. Symp. Circuits Systems, May 1990, pp. 865-868.

[6] J. Hu, Y. Shin, N.Dhanwada, and R. Marculescu, “Architecting voltage islands in core-based system-on-a-chip designs” IEEE International Symposium on Low Power Electronics and Design, pp. 180-185, 2004

[7] W. Hwang, “New trends in low power SoC design technologies” IEEE International SOC Conference, p. 422, 2003

[8] D. Lackey, P. Zuchowski, T. Bednar, D.Stout, S. Gould, and J. Cohn, “Managing power and performance for system-on-chip designs using voltage islands” in IEEE International Conference on Computer Aided Design, pp. 195-202, 2002.

[9] M. Lai and D.F. Wong, “Maze routing with buffer insertion and wire sizing” in Proc. of the Design Automation Conference, pp. 374-378, 2000.

[10] W.-P. Lee, H.-Y. Liu, and Y.-W. Chang, “Voltage island aware floorplanning for power and timing optimization” in IEEE International Conference on Computer Aided Design, pp. 389-394, 2006.

[11] Z. Li and W. Shi, “A fast algorithm for optimal buffer insertion” IEEE Trans. on Computer-Aidede Design, vol. 24, no. 6, June 2005, pp. 879-891.

[12] Z. Li, C. N. Sze, C. J. Alpert, J. Hu and W. Shi, “Making fast buffer insertion even faster via approximation techniques” Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 13-18, Jan. 2005

References (cont.)

[13] J. Lillis, C. K. Cheng, and T. Lin, “Optimal wire sizing and buffer insertion for low power and a generalized delay model” in IEEE/ACM International Conference on Computer Aided Design, pp. 138-143, 1995.

[14] M.-C. Lu, M.-C. Wu, H.-M. Chen, and H.-R. Jiang, “Performance Constraints Aware Voltage Island Generation in SoC Floorplan Design” in IEEE International SOC Conference, pp. 211-214, 2006.

[15] Y. Peng and X. Liu, “Freeze: Engineering a fast repeater insertion solver for power minimization using the Ellipsoid method” in Proc. of the Design Automation Conf., pp.813-818, 2005.

[16] K. H. Tam and L. He, “Power optimal dual vdd buffered tree considering buffer stations and blockages” in Proc. of the Design Automation Conf., pp. 497-502, 2005.

[17] X. Tang, R. Tian, H. Xiang, and D. F. Wong, “A new algorithm for routing tree construction with buffer insertion and wire sizing under obstacle constraints” in IEEE International Conference on Computer Aided Design, pp. 49-56, 2001.

[18] H. Wu, I-Min. Liu, Martin D. F. Wong, and Y. Wang, “Post-placement voltage island generation under performance requirement” in IEEE International Conference on Computer Aided Design, pp. 309-316, 2005.

[19] H. Wu, Martin D. F. Wong, and I-Min Liu, “Timing-constrained and voltage island aware voltage assignment” in Proc. of the Design Automation Conference, pp. 429-432, 2006

[20] A. Youssef, M. Anis, and M. Elmasry, “POMR: A power aware interconnect optimization methodology” IEEE Transaction on Very Large Scale Integration Systems, vol. 13, pp. 297-307, March 2005