Activity and Register Placement Aware Gated Clock Network Design

Weixiang Shen, Yici Cai, Xianlong Hong
EDA Lab, CS Dept., Tsinghua University, China

Jiang Hu
ECE Dept., A&M University, USA
Outline

- Introduction
- Motivation
- Our Contributions
- Preliminaries
  - Top-down Min-Cut Placement
  - Power Model
- Our Algorithm
  - Initial Placement
  - Gated Clock Tree Construction
  - Incremental Placement
- Experimental Results
- Conclusions and Future Work
Introduction

- **Power becomes a primary concern**
- **Clock network is the major power consumer**
Introduction

- **Clock gating**
  - Not all of the cells are active in a particular cycle
  - Shut down the idle cells could save the power
  - Clock gating is the most effective method to do this by controlling the clock signal fed into the idle cells
Introduction

● **Previous work on clock gating**

  - Activity-driven clock tree, TCAD’ 01
  - Zero-skew gated clock routing, ASPDAC’98, TCAD’01
  - Activity-sensitive clock tree, ISLPED’02
  - Power-aware clock tree planning, DAC’03, ISPD’04
  - Activity-aware register placement, ISVLSI’07

● **The drawbacks of these methods**

  - Focus on clock tree construction after placement or at RTL
  - Previous placers are not designed for gated clock tree
Outline

- Introduction
- Motivation
- Our Contributions
- Preliminaries
  - Top-down Min-Cut Placement
  - Power Model
- Our Algorithm
  - Initial Placement
  - Gated Clock Tree Construction
  - Incremental Placement
- Experimental Results
- Conclusions and Future Work
Motivation

- The parent’s activity is OR-ing of its children
- Place the registers with similar activity patterns closely
  - decrease the clock tree wirelength and capacitance
  - decrease the parent’s activity
  - shut off the clock signal more periods
- Activity rate is minimal != transition of control signal is optimal
- To place the gating logics other than removing overlap by ECO
- To control the clock skew
Motivation

- The relationship between activity and transition
  
  reg1: 1010001  reg2: 1010101  reg3: 1100001
  
  reg1-reg2: 1010101  idle: 3  transitions: 6
  reg1-reg3: 1110001  idle: 3  transitions: 2

- Different merging strategies:
  
  Blue: activity similar first
  Pink: activity & transition
  Black: transition minimal first

Aug.1, 2008  IC-DFN Workshop
Outline

- Introduction
- Motivation
- Our Contributions
- Preliminaries
  - Top-down Min-Cut Placement
  - Power Model
- Our Algorithm
  - Initial Placement
  - Gated Clock Tree Construction
  - Incremental Placement
- Experimental Results
- Conclusions and Future Work
Our Contributions

- **Our design flow**

- Gate level netlist
  - Activity and transition analysis
  - Initial placement
  - Gated clock tree construction and routing
  - Revised netlist including clock network and gate logic
  - Incremental placement
Our Contributions

- Propose a more reasonable design flow, taking the gating logics into placement, avoiding the optimal positions of the gates are occupied by other cells

- In the initial placement, we consider both the logical and physical information. Besides considering the activity similarity of the registers, we also try to minimize the resultant transitions of the control signals

- Optimize the clock skew by implementing zero skew gated clock routing after topology construction and optimization
Outline

- Introduction
- Motivation
- Our Contributions
- Preliminaries
  - Top-down Min-Cut Placement
  - Power Model
- Our Algorithm
  - Initial Placement
  - Gated Clock Tree Construction
  - Incremental Placement
- Experimental Results
- Conclusions and Future Work
Preliminaries

- Cut-based placement paradigm – Capo
  - A min-cut placer recursively partitions each bin and its associated hyper-graph at current level, its objective is to minimize the total weighted net cuts

- Power model
  \[ [(c_0 l_i + C_i) p(i) + 0.5 \times (c_0 l_g + C_g) p_{tr}] fV_{dd}^2 \]
Outline

- Introduction
- Motivation
- Our Contributions
- Preliminaries
  - Top-down Min-Cut Placement
  - Power Model
- Our Algorithm
  - Initial Placement
  - Gated Clock Tree Construction
  - Incremental Placement
- Experimental Results
- Conclusions and Future Work
Objective:
- Clump the registers into a small area to decrease the clock wirelength
- Pull the registers with the similar activity patterns closely to reduce its parent’s activity and the transitions of the control signal

Strategy:
- Capo is a minimal cut-based placer, so we can add some pseudo edges between the registers we want to pull closely
The pseudo edge’s weight is determined based on

- $w_l$, with respect to the partition level
- $w_s$, with respect to the similarity of the activity patterns
- $w_t$, with respect to the transitions of the control signal
Initial Placement

- The weight of $w_l$

$$w_l(\text{level}) = -3.0 \times \text{level}/20.0 + W \text{ if level} > T$$

- $T$: the partition level we start to add pseudo edge
- $W$: control the scope of $w_l$
- $T$ is determined based on the circuit scale
- For the beginning level, $w_l$ should be assigned a relatively large value to avoid partitioning the registers with the similar activity patterns to different bins
- For the last levels, the area of the bin is relatively small
Initial Placement

- The weight of $w_s$

$$w_s(\text{reg}1, \text{reg}2) = \frac{(C_{\text{reg}1} + C_{\text{reg}2})}{C_{\text{average}}} \times p(\text{reg}1, \text{reg}2)$$

$$p(\text{reg}1, \text{reg}2) = P(\text{reg}1.\text{act}(i) = 0, \text{reg}2.\text{act}(i) = 0) \quad (0 \leq i \leq \text{act.size})$$

$$\text{Average(bin)} = \frac{2}{n*(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_s(\text{reg}i, \text{reg}j)$$

- If reg1 and reg2’s activity patterns are similar, their parent’s activity rate will reduce, and also they can share a common gate

- Add a pseudo edge between every pair of registers will affect the run time of Capo, and it is no need if $w_s$ is small

- If the similarity between regi and regj is larger than $C_0 \times \text{Average(bin)}$, we add a pseudo edge
The weight of $w_t$

\[ w_t(regi, regj) = \frac{\text{trans}(\text{parAct}(regi, regj))}{\text{act.size()} - 1} \]

- \( \text{parAct}(regi, regj) \) gets the parent’s activity pattern of \( regi \) and \( regj \)
- \( \text{trans}(\text{act}) \) gets the transition of the control signal

e.g. \( reg1: 10110001 \) \quad \( reg2: 00010010 \)

\[ \Rightarrow \text{parent}(reg1, reg2): 10110011 \]

\[ \Rightarrow \text{transition probability of the control signal: } 4/7 \]

The resultant weight \( w \)

\[ w(\text{edge}(regi, regj)) = w_t(L) \times (\alpha \text{w}_s(regi, regj) + \beta(1 - w_t(regi, regj))) \]
Gated Clock Tree Construction

- **Node merging**
  
  \[ \text{dis}(i, j) = \alpha f(D(i, j)) + \beta g(L(i, j)) \]
  
  \[ = \alpha \left[ \frac{1}{\text{dim}_{\max}} (| x_i - x_j | + | y_i - y_j |) \right] \rightarrow \text{Physical information} \]
  
  \[ + \beta \left[ 1 - \frac{1}{C_{\text{tot}}} (C_i + C_j) p(i, j) \right] \rightarrow \text{Activity information} \]

- **Gate moving**
Algorithm: Clock Tree Construction

Require: a set of sinks \( \{n_i\} \)
repeat
  for each pair of \( n_i, n_j \) do
    compute \( \text{dis}(n_i, n_j) \)
  end for
  pick the pair \( n_i, n_j \) whose \( \text{dis}(n_i, n_j) \) is minimum
  merge these two nodes and generate parent \( n_k \)
  remove node \( n_i, n_j \)
  push \( n_k \) to nodes set
until only one node is left in nodes set
Gated Clock Tree Construction

- Zero skew clock routing based on the DME algorithm
Incremental Placement

- Integrate the gated clock network into previous database
  - change the +Placed attribute of the registers to +Fixed
  - add the gating logics into the database and fix them

- The key of our incremental placement
  - try to preserve the gated clock network we optimized before and
    remove the overlaps of the gating logics with the other cells
  - minimize the overheads of signal nets wirelength and power against
    the ECO method
Outline

- Introduction
- Motivation
- Our Contributions
- Preliminaries
  - Top-down Min-Cut Placement
  - Power Model
- Our Algorithm
  - Initial Placement
  - Gated Clock Tree Construction
  - Incremental Placement

- Experimental Results

- Conclusions and Future Work
## Experimental Results

### Comparison results of our algorithm with ISVLSI’07

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Algorithm</th>
<th>Clock Wirelength</th>
<th>Clock Power</th>
<th>#Gate</th>
<th>Clock Skew (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1488</td>
<td>ref</td>
<td>11362.50</td>
<td>8.17E-5</td>
<td>2</td>
<td>1.62E-11</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>9464.29</td>
<td>7.70E-5</td>
<td>1</td>
<td>&lt;1.0E-12</td>
</tr>
<tr>
<td>s15850</td>
<td>ref</td>
<td>1.85E6</td>
<td>0.005795</td>
<td>38</td>
<td>2.02E-8</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>1.68E6</td>
<td>0.005399</td>
<td>44</td>
<td>&lt;1.0E-12</td>
</tr>
<tr>
<td>s35932</td>
<td>ref</td>
<td>5.61E6</td>
<td>0.007507</td>
<td>99</td>
<td>9.57E-8</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>5.23E6</td>
<td>0.007071</td>
<td>105</td>
<td>&lt;1.0E-12</td>
</tr>
<tr>
<td>s38417</td>
<td>ref</td>
<td>5.48E6</td>
<td>0.027230</td>
<td>121</td>
<td>5.17E-8</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>5.20E6</td>
<td>0.025969</td>
<td>103</td>
<td>&lt;1.0E-12</td>
</tr>
<tr>
<td>s38584</td>
<td>ref</td>
<td>4.99E6</td>
<td>0.012993</td>
<td>103</td>
<td>8.09E-8</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>4.69E6</td>
<td>0.012288</td>
<td>98</td>
<td>&lt;1.0E-12</td>
</tr>
<tr>
<td>Ave ratio</td>
<td></td>
<td>0.91</td>
<td>0.94</td>
<td>0.90</td>
<td></td>
</tr>
</tbody>
</table>
## Experimental Results

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Algorithm</th>
<th>HPWL</th>
<th>Signal Power</th>
<th>Slack (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1488</td>
<td>ref</td>
<td>1.90E6</td>
<td>0.003160</td>
<td>0.891</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>1.93E6</td>
<td>0.003280</td>
<td>0.840</td>
</tr>
<tr>
<td>s15850</td>
<td>ref</td>
<td>2.53E6</td>
<td>0.018373</td>
<td>0.348</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>2.66E6</td>
<td>0.019017</td>
<td>0.638</td>
</tr>
<tr>
<td>s35932</td>
<td>ref</td>
<td>4.81E6</td>
<td>0.014851</td>
<td>1.48</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>5.16E6</td>
<td>0.015409</td>
<td>3.59</td>
</tr>
<tr>
<td>s38417</td>
<td>ref</td>
<td>5.63E6</td>
<td>0.064018</td>
<td>-12.80</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>5.98E6</td>
<td>0.066258</td>
<td>-9.10</td>
</tr>
<tr>
<td>s38584</td>
<td>ref</td>
<td>6.48E6</td>
<td>0.036635</td>
<td>-5.83</td>
</tr>
<tr>
<td></td>
<td>our</td>
<td>6.82E6</td>
<td>0.037526</td>
<td>-2.48</td>
</tr>
<tr>
<td>Average ratio</td>
<td></td>
<td>1.05</td>
<td>1.03</td>
<td></td>
</tr>
</tbody>
</table>

Aug.1, 2008 IC-DFN Workshop
Experimental Results

- Clock nets and signal nets trade off by pseudo edges

![Graph showing experimental results]

- Clock nets wirelength and power
- HPWL and signal nets power

Aug.1, 2008
Initial Placement

- The weight of $w_1$

$$w_1(\text{level}) = -3.0 \times \text{level}/20.0 + W \text{ if level} > T$$

- $T$: the partition level we start to add pseudo edge
- $W$: control the scope of $w_1$
- $T$ is determined based on the circuit scale
- For the beginning level, $w_1$ should be assigned a relatively large value to avoid partitioning the registers with the similar activity patterns to different bins
- For the last levels, the area of the bin is relatively small
Initial Placement

- The weight of $w_s$

\[
  w_s(\text{reg}1, \text{reg}2) = \frac{(C_{\text{reg}1} + C_{\text{reg}2})}{C_{\text{average}}} \cdot p(\text{reg}1, \text{reg}2)
\]

\[
p(\text{reg}1, \text{reg}2) = P(\text{reg}1.\text{act}(i) = 0, \text{reg}2.\text{act}(i) = 0) \quad (0 \leq i \leq \text{act.size})
\]

\[
  \text{Average(bin)} = \frac{2}{n \cdot (n - 1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_s(\text{reg}i, \text{reg}j)
\]

- If reg1 and reg2’s activity patterns are similar, their parent’s activity rate will reduce, and also they can share a common gate.

- Add a pseudo edge between every pair of registers will affect the run time of Capo, and it is no need if $w_s$ is small.

- If the similarity between regi and regj is larger than $C_0 \cdot \text{Average(bin)}$, we add a pseudo edge.
Outline

• Introduction
• Motivation
• Our Contributions
• Preliminaries
  - Top-down Min-Cut Placement
  - Power Model
• Our Algorithm
  - Initial Placement
  - Gated Clock Tree Construction
  - Incremental Placement
• Experimental Results
• Conclusions and Future Work
Conclusions and Future Work

● Conclusions

- Activity pattern plays an important role in gated clock tree construction
- Activity aware register placement is effective for gated clock tree to reduce power
- Transition of the control signal could not be ignored
- Clock nets power and wirelength Vs. signal nets power and HPWL

● Future work

- Gating logic planning during placement
- Power Vs. performance (delay, skew, timing, et al.)
Thank you!

E-mail: cwx04@mails.tsinghua.edu.cn
reg1: 1000011100
reg2: 0111000010
reg3: 1100011100
reg4: 0110000011

a) n1: 1111011110
n2: 1110011111

b) n1: 1100011100
n2: 0111000011