Timing Driven Power Gating in High-Level Synthesis

Shih-Hsu Huang and Chun-Hua Cheng

Department of Electronic Engineering
Chung Yuan Christian University, Taiwan
Power Gating Technique (1/2)

- In standby mode: (the sleep transistor is turned off)
  - The standby leakage current of the functional unit is proportional to the size of the sleep transistor.
  - Small sleep transistor to reduce leakage
In active mode: (the sleep transistor is turned on and works as a resistor)
- The sleep transistor produces a voltage drop that degrades the speed of the functional unit.
Pervious Works

- Up to now, the impact of high-level synthesis on the maximum allowable delays of functional units (for a target clock period) has not been studied.
  - Since the clock skew is assumed to be zero, the maximum allowable delay of each functional unit is definitely the target clock period; thus, there is no need to study this problem.

- However, in modern high-speed circuit design, the clock skew is often intentionally utilized to improve the circuit performance.
Our Contributions

In this paper, we present the first work to formally draw up the timing driven power gating problem in the high-level synthesis of non-zero clock skew circuits.

- Given a target clock period and design constraints, our objective is to derive the minimum-standby-leakage-current resource binding solution.

Our work includes the following two aspects:

- First, we propose an MILP (mixed integer linear programming) approach to guarantee obtaining the optimal solution.
- Second, we also propose a heuristic approach to deal with the same problem in polynomial time complexity.
Outline

- Introduction
- Motivation
- Our Approach
  - MILP
  - Heuristic Algorithm
- Experimental Result
- Conclusion
Suppose we are given two multipliers, called $\text{mul}_1$ and $\text{mul}_2$, and two adders, called $\text{add}_1$ and $\text{add}_2$:

<table>
<thead>
<tr>
<th>Type of Functional Unit</th>
<th>Transistor Size</th>
<th>Delay (min,max)</th>
<th>Leakage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiplier (mul)</td>
<td>Small (S)</td>
<td>(34,40)</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Medium (M)</td>
<td>(33,38)</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Large (L)</td>
<td>(28,34)</td>
<td>35</td>
</tr>
<tr>
<td>Adder (add)</td>
<td>Small (S)</td>
<td>(10,12)</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Large (L)</td>
<td>(8,10)</td>
<td>5</td>
</tr>
</tbody>
</table>

If the power gating implementation selection is $\text{mul}_1(\text{fast})$, $\text{mul}_2(\text{fast})$, $\text{add}_1(\text{fast})$, and $\text{add}_2(\text{fast})$

Total standby leakage current is 80 (due to 35+35+5+5)
Suppose we are given two multipliers, called \textit{mul}_1 and \textit{mul}_2, and two adders, called \textit{add}_1 and \textit{add}_2:
Suppose we are given four registers, called R1, R2, R3, and R4:

- Step 1
- Step 2
- Step 3
- Step 4

R1 \{a\} R2 \{c, f\}
R3 \{b, e\} R4 \{d, g\}
If the clock skew is zero, the clock period cannot be less than 34.

mul₁(fast) = \{o₂,o₇\},
mul₂(fast) = \{o₄\},
add₁(fast) = \{o₁,o₅,o₈\},
add₂(fast) = \{o₃,o₆\},
R₁ = \{a\}, R₂ = \{b,e\},
R₃ = \{c,f\}, and R₄ = \{d,g\}.  

Resource Binding
By properly scheduling the clock arrival time of registers, the clock period can be shorter than the longest combinational delay.

Several graph-based algorithms use the constraint graph to solve the optimal clock skew scheduling.
DRAWBACK OF EXISTING FLOW

Resource Binding

mul₁(fast) = \{o₂, o₇\},
mul₂(fast) = \{o₄\},
add₁(fast) = \{o₁, o₅, o₈\},
add₂(fast) = \{o₃, o₆\},
R₁ = \{a\}, R₂ = \{b, e\},
R₃ = \{c, f\}, and R₄ = \{d, g\}.

Circuit Graph

The smallest feasible clock period is 22.
Total standby leakage current is 80.
The smallest feasible clock period is only 22.
Total standby leakage current is only 49.
Outline

- Introduction
- Motivation
- Our Approach
  - MILP
  - Heuristic Algorithm
- Experimental Result
- Conclusion
Objective Function

- Our objective function is to minimize total standby leakage current.

- Minimize

\[
\sum_{z \in Q} \sum_{w \in h(e(z))} f_{z,e(z),w} \cdot I_{e(z),w}. 
\]

Minimize

\[
\begin{align*}
&f_{\text{mul1},<\text{mul},L>} \times 35 + f_{\text{mul1},<\text{mul},M>} \times 20 + f_{\text{mul1},<\text{mul},S>} \times 5 \\
&f_{\text{mul2},<\text{mul},L>} \times 35 + f_{\text{mul2},<\text{mul},M>} \times 20 + f_{\text{mul2},<\text{mul},S>} \times 5 \\
&f_{\text{add1},<\text{add},L>} \times 5 + f_{\text{add1},<\text{add},S>} \times 4 \\
&f_{\text{add2},<\text{add},S>} \times 4.
\end{align*}
\]
Each functional unit must select one implementation. Thus, for each functional unit $z$, we have the following constraint:

$$\sum_{w \in \text{h}(e(z))} f_{z,<e(z),w>} = 1.$$ 

In this example, we have the following constraints:

$$f_{\text{mul1},<\text{mul},L>} + f_{\text{mul1},<\text{mul},M>} + f_{\text{mul1},<\text{mul},S>} = 1; \text{ and so on.}$$
If the functional unit $z$ is not implemented by $<e(z),w>$, then the value of binary variable $y_{j,z,<e(z),w>}$ is definitely to be 0. Therefore, we have the following constraint: $y_{j,z,<e(z),w>} \leq f_{z,<e(z),w>}$.

In this example, we have the following constraints:

$y_{o1,add1,<add,L>} \leq f_{add1,<add,L>}$;

and so on.

The speed of $O_1$ is slow.
Each operation must be assigned to one functional unit. Therefore, for each operation $o_j$, we have the following constraint:

$$\sum_{z \in c(g(j))} \sum_{w \in h(g(j))} y_{j,z, <e(z),w>} = 1.$$ 

In this example, we have the following constraints:

$$y_{o1, \text{add1}, \langle \text{add}, \text{L} \rangle} + y_{o1, \text{add1}, \langle \text{add}, \text{S} \rangle} + y_{o1, \text{add2}, \langle \text{add}, \text{L} \rangle} + y_{o1, \text{add2}, \langle \text{add}, \text{S} \rangle} = 1; \text{ and so on.}$$
If two operations have overlapping lifetimes, they cannot share the same functional unit. Thus, we have the following constraint:

\[ y_{j,z,<e(z),w>} + y_{k,z,<e(z),w>} \leq 1. \]

In this example, we have the following constraints:

\[ y_{o1,add1,<add,L>} + y_{o3,add1,<add,L>} \leq 1; \]
\[ y_{o1,add1,<add,S>} + y_{o3,add1,<add,S>} \leq 1; \]
and so on.

**O₁, O₃ Life Time conflict**
Let $P$ be a constant that denotes the target clock period. For the input variable $u$ and the output variable $v$ of operation $o_j$, the maximum allowable delay must satisfy the setup constraint:

$$
\sum_{z \in \epsilon(g(j))} \sum_{w \in h(g(j))} y_{j,z,<e(z),w>} \cdot D_{<e(z),w>} \leq P - T_u + T_v.
$$

In this example, we have the following constraints:

$$
y_{o1,\text{add1},<\text{add},L>} \times 10 + y_{o1,\text{add1},<\text{add},S>} \times 12 + \
y_{o1,\text{add2},<\text{add},L>} \times 10 + y_{o1,\text{add2},<\text{add},S>} \times 12 \
\leq P - T_{\text{host}} + T_a; \quad (P = 22)
$$
and so on.
For the input variable \( u \) and the output variable \( v \) of operation \( o_i \), the minimum allowable delay must satisfy the hold constraint:

\[
T_v - T_u \leq \sum_{z \in c(g(j))} \sum_{w \in h(g(j))} y_{j,z,<e(z),w>} \cdot d_{<e(z),w>}. \]

In this example, we have the following constraints:

\[
T_a - T_{\text{host}} \leq y_{o1,\text{add1},<\text{add},L>} \times 8 + y_{o1,\text{add1},<\text{add},S>} \times 10 + y_{o1,\text{add2},<\text{add},L>} \times 8 + y_{o1,\text{add2},<\text{add},S>} \times 10; \]

and so on.

Formula 8~11 (Use the Register Binding Approach in [7] )
Outline

- Introduction
- Motivation
- Our Approach
  - MILP
  - Heuristic Algorithm
- Experimental Result
- Conclusion
Heuristic Approach

- We provide a method to estimate the maximum allowable delay of each operation before the resource binding.
- We try to assign the operations that have similar maximum allowable delay to the same functional unit.
- As a result, we can have more non-critical functional units.
Step 1: Estimate the Maximum Allowable Delay

- The estimated maximum allowable delays of operations $o_1$, $o_2$, $o_3$, $o_4$, $o_5$, $o_6$, $o_7$, and $o_8$ are 30, 46, 30, 34, 22, 10, 34, and 10, respectively.

$$P = 12$$

The maximum allowable delays $P-E_u+L_v$

$$E_v = -8, \quad L_v = 8$$
We consider the multiplier type. The estimated maximum allowable delays of operations $o_2$, $o_4$, and $o_7$ are 46, 34, and 34, respectively. ($o_4$, $o_7$ high priority)
Suppose operation $o_4$ is chosen. We assign operation $o_4$ to $mul_1$ (fast). The estimated maximum allowable delays of operation $o_7$ is 34.
We assign operation $o_7$ to $mul_1(fast)$. The estimated maximum allowable delays of operation $o_7$ is 34.
Step 2: Power Gating Implementation (4/5)

- Because of the lifetime constraint, we find operation o₂ cannot be assigned to the functional unit mul₁. We assign operation o₂ to mul₂(slow).

\[ \text{o₂ maximum allowable delays} = 46. \]
Similarly, we consider the adder type. We have $\text{add}_1(\text{fast}) = \{o_3, o_6, o_8\}$, $\text{add}_2(\text{slow}) = \{o_1, o_5\}$. 
Step 3: Use the Register Binding Approach in [7]

- We use [7] to perform register binding for clock period minimization. We have R1 = \{a, e\}, R2 = \{b\}, R3 = \{c, f\}, and R4 = \{d, g\}. The smallest feasible clock period is 22.
Step 4: Adjust the Power Gating Implementation

- Now, the actual maximum allowable delay of each functional unit can be calculated based on the target clock period and the clock skew schedule derived in the third step.

- We adjust the power gating implementation of each functional unit according to its actual maximum allowable delay.

The total standby leakage current is only 49.
Outline

- Introduction
- Motivation
- Our Approach
  - MILP
  - Heuristic Algorithm
- Experimental Result
- Conclusion
Experimental Environment

- Our platform is Windows XP operating system running on AMD K8-4200+ processor.
  - We use Extended-LINGO Release 10.0 as the mixed integer linear program solver and use C programming language to implement the process of solution space reduction.
- We compare our approach with existing design flow.

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Existing Flow</th>
<th>Our MILP</th>
<th>Our MILP</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>HAL</td>
<td>1.764</td>
<td>0.484</td>
<td>0.484</td>
<td>72.56%</td>
</tr>
<tr>
<td>AR</td>
<td>2.644</td>
<td>0.940</td>
<td>1.152</td>
<td>64.45%</td>
</tr>
<tr>
<td>BF</td>
<td>1.336</td>
<td>0.484</td>
<td>0.484</td>
<td>63.77%</td>
</tr>
<tr>
<td>EWF</td>
<td>1.350</td>
<td>0.498</td>
<td>0.738</td>
<td>63.11%</td>
</tr>
<tr>
<td>IDCT1</td>
<td>3.585</td>
<td>1.458</td>
<td>1.956</td>
<td>59.33%</td>
</tr>
<tr>
<td>Motion</td>
<td>7.376</td>
<td>2.607</td>
<td>4.118</td>
<td>54.62%</td>
</tr>
</tbody>
</table>

Improvement:
- 64.65%
- 54.62%
This paper presents the first work to formally formulate the timing driven power gating in the high-level synthesis of a non-zero clock skew circuit.

Given a target clock period and design constraints, our goal is to find a resource binding solution so that the total standby leakage current is minimized.

Compared with the existing design flow, our approach has a significant improvement without any overhead.
Thank you