Gate Sizing and Buffer Insertion Algorithm to Reduce Glitch Power Dissipation

Sungjae Kim, Juho Kim, and Sunyoung Hwang

Abstract

In this paper, we propose an efficient algorithm to reduce glitch power dissipation in CMOS logic circuits. The proposed algorithm takes path balancing approach that is achieved by gate sizing and buffer insertion methods. The gate sizing technique reduces not only glitches, but also effective capacitance in the circuit. After gate sizing, buffer insertion is performed for those remaining unbalanced paths that are not covered by gate sizing. Buffers are inserted between the gates where power reduction achieved by glitch reduction is larger than the power consumed by the inserted buffers. The ILP (Integer Linear Program) has been employed to determine the location of inserted buffers, which is a very difficult problem because the power reduction by buffer insertion is closely related to other inserted buffers. The proposed algorithm has been tested on LGSynth91 benchmark circuits. Experimental results show that 61.5% of glitch reduction and 30.4% of power reduction are achieved without increasing the critical path delay.

Keywords: Low Power Design, Glitch, Gate Sizing, Buffer Insertion, ILP

I. Introduction

Due to the growth of portable electronics, low power design techniques are required with the high density integration techniques. The total power consumed by CMOS digital circuits is composed of dynamic power caused by signal transition, and the power due to short-circuit current and leakage current [1]. Of these power dissipation sources, the dynamic power is dominant. The average dynamic power consumed by a CMOS gate $i$ can be represented as by Equation (1).

$$P_i = \frac{1}{2} C_i \cdot V_{dd}^2 \cdot f \cdot D(i)$$

(1)

where $C_i$ is the load capacitance, $V_{dd}$ is the supply voltage, $f$ is the clock frequency and $D(i)$ is the transition density of gate $i$ which denotes the average number of signal transitions per clock cycle. The signal transitions can be classified into functional transitions and spurious transitions (glitches). The spurious transitions are not related to the functionality of a circuit, however it is known to consume about 20% - 70% of the total power dissipation in CMOS circuits [2][3].

Recently, researches toward dynamic power optimization have been focused on gate sizing techniques [4]-[11]. These methods try to achieve power reduction by minimizing chip area [4]-[6] or capacitance under given timing constraints [7]-[10]. Since load capacitance is proportional to gate size, the power dissipation in Equation (1) can be reduced by minimizing gate size. But these approaches did not consider the glitches caused by input signals which have different arrival times. The power due to glitches may in turn offset the power reduction achieved by gate sizing.

We propose an algorithm that maximizes path balancing and reduces load capacitance at the same time. The algorithm embodies two processes - gate sizing and buffer insertion. As the first step, gate sizing algorithm is applied prior to buffer insertion. Signal arrival time, required time and slack are computed for each gate as proposed in [8]. Then, the minimum input signal required time and the size of a gate are determined such that glitches and load capacitance can be reduced. During the gate sizing process, those signal paths that cannot be balanced are identified and marked. These are considered as candidate locations for buffer insertion. For balancing the remaining paths, buffers

---

Manuscript received January 28, 2000; accepted May 2, 2000.
Sungjae Kim is with the SiliconCraft Inc., CAD Laboratory, Ever Building 600-3, 31-157 Nogangdong, Mapo-gu, Seoul 121-100, Korea.
Juho Kim is with the Department of Computer Science, Sogang University, Shinsu-dong, Mapo-gu, Seoul 121-742, Korea.
Sunyoung Hwang is with the Department of Electronic Engineering, Sogang University, Shinsu-dong, Mapo-gu, Seoul 121-742, Korea.
are inserted. Among the candidate locations identified during gate sizing, we select the locations and insert the buffers such that power reduction achieved by reducing glitches can be larger than the power consumed by added buffers. Finding the buffer locations for optimized power is an NP-complete problem. In the proposed system, we formulate the buffer insertion problem into the ILP-based approach to find the solution.

The rest of the paper is organized as follows. In the next section, we describe a glitch model and a power estimation method. Section III describes gate sizing and buffer insertion techniques that are proposed to achieve path balancing. Experimental results are presented in Section IV, and conclusions are drawn in Section V.

II. Glitch Model

In a CMOS digital circuit, signal transition from logic level 0 to logic level 1 or vice versa is the main source of power dissipation. A glitch is a spurious transition and also consumes power just as a functional transition does. An example of the glitch is given in Fig. 1.

![Fig. 1. Glitches. (a) Static glitch, (b) Dynamic glitch.](image)

A glitch can be classified into a generated glitch and a propagating glitch depending on its precedence [11]. A glitch is generated when non-glitch input signals arrive at different times (larger than the inertial delay of a gate) and the input signals make transitions to introduce a glitch. A propagating glitch is caused by an input signal which contains glitches. It can be propagated through gates or suppressed by a side-input signal.

Glitches can be removed by path balancing. Because the main cause for a glitch is the difference in input signal transition times, glitch reduction is achievable by making the transitions of input signals occur at the same time. Even though it is impossible to achieve exact synchronization, it is possible to bring them within the range of the inertial delay of a gate.

The probabilistic approaches to estimate glitches have been proposed in [11][12]. But these approaches do not take the spatial and temporal correlations into consideration. So, we adopt the event-driven logic simulation method to compute spurious transitions. The width of a glitch is also a factor in determining whether a glitch is propagated or suppressed. The event-driven logic simulation can exactly determine propagated or suppressed glitches according to the width of glitches. A glitch makes even number of signal transitions at a node. Since just one of the transitions occurred at a node is a functional transition, If the number of signal transitions at a node for a given input vector obtained by simulation is \(N_{res}\), the number of transitions by glitches can be represented as Equation (2).

\[
N_{glitch_{res}} = \left[ \frac{N_{res}}{2} \right] \times 2
\]  

(2)

If we suppose that a input vector is supplied once per a clock period, the transition density including glitches at gate \(i\), \(D(i)\) can be computed using \(N_{res}\) as Equation (3), where \(T\) represents the clock period.

\[
D(i) = \frac{N_{res}}{T}
\]  

(3)

This equation can be used for Equation (1) to estimate the dynamic power.

III. Glitch Reduction Algorithm

The proposed algorithm applies gate sizing and buffer insertion to achieve path balancing. In gate sizing, signal arrival times, required times and slacks are computed. Although gate downsizing can reduce load capacitance, it may produce new glitches. The proposed algorithm reduces load capacitance and glitches at the same time. Buffer insertion reduces the glitches that are not suppressed during gate sizing.

1. Gate Sizing Algorithm

To apply our gate sizing algorithm the arrival times, required times and slacks are computed for all the gates. The topological delay analysis method is applied for safety. Fig. 2 shows an example circuit showing the computed slacks. The delay of a gate can be increased through downsizing by the amount of its slack without violating timing constraints. This can reduce load capacitance, but it may produce new glitches.
Fig. 2. The arrival times, required times, and slacks of an example circuit. (The number within the braces denotes the delay of a gate.)

To determine the size of a gate (hence the delay of a gate) to reduce load capacitances and glitches, we define minimum input signal required time of a gate as follows.

**Definition 1:** The minimum input signal required time of a gate $G$ ($\tau_{\text{min, req}}(G)$) is the minimum of the input signal arrival times. For a gate $G$ has $n$ fansin,

$$\tau_{\text{min, req}}(G) = \min(t_r(v_1), t_r(v_2), \ldots, t_r(v_n))$$

(4)

where $t_r(v_i)$ is the required time of signal $v_i$.

Fig. 3. Determination of gate delay. If the delay of Ga is increased to 4, path balancing is maximized.

Downsizing a gate to increase the delay by the amount of the slack can reduce the capacitance of the sized gate. But it may introduce new glitches and make other gates cannot be downsized. Downsizing a gate as amount of $\text{Sizable delay}$ makes more gates be downsized. As the number of sized gate increases, the path balancing can be achieved.

$$\text{Sizable delay} = \min_{G \in \text{level}(G)} (\tau_{\text{min, req}}(G)) - t_r$$

(5)

It is very important to determine the sequence of gates to be sized. As mentioned earlier, glitches can be classified into generated glitches and propagating glitches. A generated glitch can occur at any gate of a circuit, but a propagating glitch is the propagation of an already existing glitch. A propagating glitch never occurs when there are not any glitches in the input signal of a gate. Therefore, it is preferred to select gates near the primary inputs first. After levelizing the gates using Equation (6), sizing is performed starting from the gates with lower levels.

$$\text{level}(G) = \max_{G \in \text{level}(G)} (\text{level}(G) + 1)$$

(6)

The proposed gate sizing algorithm is given in Fig. 4. During gate sizing, we can find that some paths are never be balanced just by gate sizing alone.

```
GadgetSizing()
Levelize the circuit;
Compute the arrival time, required time, slack for each gate;
for each gate $G_i$, Compute $\tau_{\text{min, req}}(G_i)$;
for $i = 1; 1 < \text{max level}; i++$ |
  for each gate $G_i$ in Level $i$ |
    $D_i = \text{Sizable Delay of } G_i$;
    if ($D_i > 0$) |
      Downsize gate $G_i$;
      Update timing information by downsizing gate $G_i$;
    |
    /* for */
  Find candidate locations for buffer insertion;
  /* for */
}
```

Fig. 4. Gate sizing algorithm to reduce glitches.

In such a case, locations for the buffer insertion are marked and delays which can achieve path balancing are determined.

The example circuit in Fig. 2 is changed after gate sizing as shown in Fig. 5. For simplicity, we assume in the example that the delay of all gates is in the range from 1 to 5. By the proposed algorithm, the delay of gate G1 is
increased from 1 to 5, G4 and G5 from 1 to 4, G8 from 2 to 5, G11 from 1 to 5, and G20 is from 1 to 4. The critical path delay which amounts to 11 is not increased after gate sizing.

![Circuit Diagram]

**Fig. 5.** The example circuit after gate sizing. The candidate buffer locations are shown in dotted lines. The delay of all gates is in the range from 1 to 5.

2. Buffer Insertion Method Based on ILP

To determine the optimal locations for buffer insertion, all combinations of candidate buffers must be taken into consideration. However, when there are n candidate buffers, 2^n power estimations are needed, thus categorizing the problem as NP-complete. The difficulties in determining the buffer locations arise from the spatial correlation between buffers as shown in Fig. 6.

![Spatial Correlation Diagram]

**Fig. 6.** The spatial correlation between buffers.

Suppose that the achievable power reduction is \( \Delta P(b) \), when inserting buffer \( b \) at G1, and \( \Delta P(b_2) \) when inserting buffer \( b_2 \) at G2. If we insert both buffers, the given circuit is completely balanced and glitches do not exist anymore. Let's further suppose the achievable power reduction is \( \Delta P(b_1,b_2) \) when both \( b_1 \) and \( b_2 \) are inserted. Through extensive experimentation, we found that \( \Delta P(b_1,b_2) > \Delta P(b_1) \), \( \Delta P(b_1,b_2) > \Delta P(b_2) \) in most cases. To solve the spatial correlation, we formulate buffer insertion problem as follows:

maximize Power reduction thru buffer insertion (7)
subject to The number of buffers in a path \( \leq 1 \)

To formulate the mathematical model, two binary decision variables \( x_i \) and \( a_i \) are defined, as shown in Equations (8) and (9).

\[
x_i = \begin{cases} 
1 & \text{if candidate buffer } i \text{ is to be inserted} \\
0 & \text{if candidate buffer } i \text{ is not to be inserted} 
\end{cases} 
\]

\[
a_i = \begin{cases} 
1 & \text{if candidate buffer } i \text{ is on path } i \\
0 & \text{if candidate buffer } i \text{ is not on path } i 
\end{cases} 
\]

The gain function \( G \) can be written as follows:

\[ G = x_1 \Delta P(b_1) + x_2 \Delta P(b_2) + \ldots + x_n \Delta P(b_n) + x_1 x_2 \Delta P(b_1,b_2) + \ldots + x_1 x_n \Delta P(b_1,b_n) + \ldots + x_n x_1 \Delta P(b_n,b_1) \]

This equation is not linear. Because one location is selected per a path, a buffer is likely to be inserted to the location close to the primary input. It makes the spatial correlation between candidate buffers trivial. Then the ILP formulation can be rewritten as Equation (10).

\[
\text{maximize } x_1 \Delta P(b_1) + x_2 \Delta P(b_2) + \ldots + x_n \Delta P(b_n) + \ldots \\
\text{subject to }
\]

\[ a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq 1 \\
 a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq 1 \\
 a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq 1 \\
\ldots \\
 a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq 1 
\]

![Buffer Insertion Diagram]

**Fig. 7.** The result of buffer insertion. Buffers b1, b2, b3, b4 are inserted.

The power reduction achieved by inserting buffer \( b_n \), \( \Delta P(b_n) \) is computed prior to the evaluation of the gain function. After finding the solution of Equation (10), the
buffer \( b_i \) is inserted into the path if \( r_i \) is equal to 1. For the candidate buffers which are not inserted, the ILP algorithm is repeated. Program is stopped when the gain function becomes positive. Fig. 7 shows the results of buffer insertion for the circuit in Fig. 2. In this example, we assume that \( \Delta P(b_1) = 10, \Delta P(b_2) = 10, \Delta P(b_3) = 7, \Delta P(b_4) = 9, \Delta P(b_5) = 9, \Delta P(b_6) = 9 \).

IV. Experimental Results

The algorithm described in this paper has been implemented in C on an Ultra Sparc workstation. The library we used contains AND, OR, NAND, NOR, BUFFER, and INVERTER gates. Each type of gate has five different implementations, each of which has different area-delay ratio. For simplicity, we assumed that the gate capacitance and gate area are proportional to each other. The inertial delay model was used in our experiment. For the accurate estimation of transition density by glitches, event-driven simulation was employed. To estimate the power dissipation, capacitance values of nodes were achieved by the method similar to SIS [13]. Benchmarking has been performed on several combinational circuits from the LGSynth91 benchmark suite.

As a result, the gate count, the percentage of power reduction obtained by gate sizing, and that obtained by buffer insertion together with gate sizing are illustrated in Table 1. The gate sizing alone achieved an average of 25.7% power reduction. An average of 30.4% of power reduction was achieved by buffer insertion with gate sizing. The insertion of buffers takes unwanted power dissipation. However, greater overall power savings can be obtained by compensating the power losses due to inserted buffers.

Table 2 reports results of glitch reduction. The first column of the table represents the percentage of glitches in the total switching activities of benchmark circuits. The percentage of glitches after gate sizing is given in the third column. The C1908 circuit is composed of 880 gates and 35.3% of the signal transitions are glitches. Before sizing, 482 gates were balanced. After sizing, 542 gates were balanced and 18.1% of the capacitance was reduced with glitch reduction of 25.8%. The power reduction by gate sizing was 27.2% and the glitch ratio of sized circuit was 29.5%. During the sizing process, among 356 candidate buffers 98 buffers were selected by the proposed-ILP based buffer insertion method. As a result, 33.0% of power reduction was achieved by reducing 55.8% of the glitches. The circuits with low glitch ratio, such as s1 or s6, have few candidate buffers and buffer insertion was rarely required. By gate sizing, an average of 31.9% of glitch reduction was achieved as shown in the fourth column of the table. After buffer insertion, glitches were reduced to

| Circuit Name | # Gates | Gate Sizing | | # Sized Gates | Power Reduction (%) | | # Inserted Buffers | Power Reduction (%) |
|--------------|---------|-------------|----------|-------------------|------------------|-------------------|-------------------|
| z4ml         | 20      | 12          |          | 33.3              | 4                | 34.7              |
| decod        | 22      | 10          |          | 15.0              | 1                | 18.5              |
| x2           | 42      | 15          |          | 28.2              | 1                | 28.6              |
| vda          | 585     | 417         |          | 32.8              | 22               | 33.1              |
| scx          | 91      | 29          |          | 15.5              | 8                | 16.0              |
| unreg        | 97      | 77          |          | 40.5              | 0                | 40.5              |
| lal          | 114     | 66          |          | 30.6              | 3                | 30.8              |
| l7           | 471     | 346         |          | 38.6              | 2                | 38.8              |
| l6           | 340     | 209         |          | 27.6              | 0                | 27.6              |
| C43          | 27      | 11          |          | 8.5               | 7                | 21.3              |
| C432         | 160     | 88          |          | 29.4              | 21               | 32.1              |
| C499         | 202     | 168         |          | 13.0              | 35               | 32.3              |
| C880         | 383     | 272         |          | 27.2              | 7                | 27.5              |
| C1355        | 546     | 80          |          | 3.8               | 97               | 28.6              |
| C1908        | 880     | 556         |          | 27.2              | 98               | 33.0              |
| C2670        | 1193    | 959         |          | 38.5              | 290              | 41.2              |
| C3540        | 1669    | 1126        |          | 28.4              | 202              | 32.8              |
| Avg          | -       | -           |          | 25.7              | -                | 30.4              |
7.9%. As a result, an average of additional 29.6% was obtained. This shows that buffer insertion combined with gate sizing are effective for glitch reduction. There are some cases where the glitch reduction is salient but the power reduction is marginal. Since a buffer itself consumes power, the power reduction ratio can be smaller than the glitch reduction ratio.

V. Conclusions

In this paper, an algorithm of combined gate sizing and buffer insertion to reduce glitches in CMOS circuits has been presented. The proposed gate sizing algorithm reduces load capacitance and glitches at the same time by path balancing. An ILP-based buffer insertion algorithm is employed to reduce glitches that cannot be removed by gate sizing. With our algorithm, 61.5% of glitch reduction and 30.4% of total power reduction are achieved without violating timing constraints.

Because the proposed algorithm does not consider the power dissipation due to short-circuit current, short-circuit power consumed by inserted small buffers is neglected. If the estimation of short-circuit power is performed at the gate level, more accurate results can be obtained.

References

Table 2. Result of glitch reductions.

<table>
<thead>
<tr>
<th>Circuit Name</th>
<th>Glitches (%)</th>
<th>Gate Sizing</th>
<th>Gate Sizing + Buffer Insertion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Glitches (%)</td>
<td>Glitch Reduction (%)</td>
<td>Glitches (%)</td>
</tr>
<tr>
<td>z4ml</td>
<td>31.2</td>
<td>30.5</td>
<td>3.1</td>
</tr>
<tr>
<td>decod</td>
<td>13.5</td>
<td>4.0</td>
<td>73.3</td>
</tr>
<tr>
<td>x2</td>
<td>12.0</td>
<td>0.4</td>
<td>64.9</td>
</tr>
<tr>
<td>vda</td>
<td>16.4</td>
<td>11.3</td>
<td>30.9</td>
</tr>
<tr>
<td>act</td>
<td>9.5</td>
<td>7.5</td>
<td>22.8</td>
</tr>
<tr>
<td>unreg</td>
<td>8.0</td>
<td>2.4</td>
<td>71.4</td>
</tr>
<tr>
<td>lal</td>
<td>6.6</td>
<td>5.2</td>
<td>23.1</td>
</tr>
<tr>
<td>j7</td>
<td>10.0</td>
<td>3.8</td>
<td>66.7</td>
</tr>
<tr>
<td>i6</td>
<td>6.3</td>
<td>2.3</td>
<td>65.4</td>
</tr>
<tr>
<td>C43</td>
<td>22.8</td>
<td>21.9</td>
<td>5.1</td>
</tr>
<tr>
<td>C432</td>
<td>31.1</td>
<td>19.2</td>
<td>47.5</td>
</tr>
<tr>
<td>C499</td>
<td>20.3</td>
<td>19.6</td>
<td>6.8</td>
</tr>
<tr>
<td>C880</td>
<td>33.7</td>
<td>32.5</td>
<td>5.5</td>
</tr>
<tr>
<td>C1555</td>
<td>28.7</td>
<td>27.6</td>
<td>2.9</td>
</tr>
<tr>
<td>C1908</td>
<td>35.3</td>
<td>29.5</td>
<td>25.8</td>
</tr>
<tr>
<td>C2670</td>
<td>40.3</td>
<td>38.7</td>
<td>10.8</td>
</tr>
<tr>
<td>C3540</td>
<td>37.6</td>
<td>35.7</td>
<td>17.6</td>
</tr>
<tr>
<td>Avg.</td>
<td>21.3</td>
<td>17.1</td>
<td>31.9</td>
</tr>
</tbody>
</table>


Sungjae Kim received the B.S. and M.S. degree in Computer Science from Sogang University, Seoul, Korea, in 1998 and 2000, respectively. In 1999, he joined the CAD Laboratory of SiliconCraft Inc., where he is now a R&D engineer. His current work involves static timing analysis and circuit optimization.

Sun-Young Hwang received the B.S. degree in electronic engineering from Seoul National University, Seoul, Korea, in 1976, the M.S. degree from Korea Advanced Institute of Science in 1978, and the Ph.D. degree in electrical engineering from Stanford University, California, USA., in 1986. Since 1986, he has been with the Center for Integrated Systems at Stanford, working on design of a high-level synthesis and simulation system. In 1986 and 1987, he held a consulting position at Palo Alto Research Center of Fairchild Semiconductor Corporation. In 1989, he joined the Department of Electronic Engineering at Sogang University, where he is now a professor. His current research interests include hardware/software co-design, and DSP/VLSI system design.