Crosstalk Noise Optimization by Post-Layout Transistor Sizing

Masanori HASHIMOTO and Hidetoshi ONODERA, Members

SUMMARY This paper proposes a post-layout transistor sizing method for crosstalk noise reduction. The proposed method downsize the drivers of aggressor wires for noise reduction, utilizing the precise interconnect information extracted from the detail-routed layouts. We develop a transistor sizing algorithm for crosstalk noise reduction under delay constraints, and construct a crosstalk noise optimization method utilizing an analytic crosstalk noise model and a transistor sizing framework that have been developed. Our method exploits the transistor sizing framework that can vary transistor widths inside cells with interconnects unchanged. Our optimization method therefore never causes a new crosstalk noise problem, and does not need iterative layout optimization. The effectiveness of the proposed method is experimentally examined using 2 circuits. The maximum noise voltage is reduced by more than 50% without delay violation. These results show that the risk of crosstalk noise problems can be considerably reduced after detail-routing.

key words: crosstalk noise, capacitive coupling noise, transistor sizing, gate sizing, post-layout optimization

1. Introduction

Crosstalk noise problem heavily depends on interconnect structure, i.e. coupling length, spacing between adjacent wires, and coupling position, and hence many techniques of routing and interconnect optimization for crosstalk noise reduction are proposed [1]–[3]. Buffer insertion is also effective for noise reduction, and some methods are proposed [4], [5]. References [6]–[8] discuss the effectiveness of transistor sizing for crosstalk noise reduction, but practical implementations are not shown. Recently, Refs. [9]–[11] propose transistor sizing methods for crosstalk noise reduction. Reference [9] expresses the influence of crosstalk noise as the amount of coupling capacitance, and optimizes noise as well as area, delay and power by gate and wire sizing. Reference [10] proposes a driver sizing algorithm for noise reduction using a crosstalk noise estimation tool [12]. Reference [11] estimates crosstalk noise by Ref. [8], and circuit area is minimized under delay and crosstalk noise constraints. The authors do not mention layout modification after optimization. When optimization results are applied to layout, a certain number of interconnects are changed, which may spoil the optimization result, or may cause a new crosstalk noise problem. More recently, a gate sizing method to reduce crosstalk induced delay is proposed [13]. This method is based on a crosstalk noise aware static timing analysis. Although this method changes circuits considerably, layout modifications are not taken into consideration. Reference [14] can not solve the problem that iterative gate sizing does not necessarily converge due to layout modification, either.

This paper proposes a post-layout transistor sizing method for crosstalk noise reduction. The proposed method optimizes detail-routed circuits and reduces peak noise voltage as much as possible while preserving interconnects entirely. The interconnect information required for crosstalk noise estimation can be accurately obtained after detail-routing. The optimization result of transistor sizing can be completely applied to the layout because the proposed method utilizes the transistor sizing framework that can downsize the transistors inside cells preserving interconnects [15], [16]. This framework is originally developed for power reduction. In this paper, we use this framework for crosstalk noise reduction. In this framework, various driving-strength cells are generated on the fly according to the optimization result, and hence transistor-level optimization can be executed in cell-base design. Cell layouts are generated by a layout generation system called VARDS [17], [18]. VARDS is based on a symbolic layout method and is enhanced to produce cell layouts with variable driving strength. Exploiting this framework, the proposed method reduces crosstalk noise efficiently after detail-routing. As for crosstalk noise estimation, our method utilizes a 2-π noise model with improved aggressor modeling [19]. This model can consider the location of coupling, the effect of distributed RC networks, and the slew of input signal. Reference [19] also mentions a transformation method that can apply all types of RC trees to the 2-π noise model, which enables crosstalk noise optimization of practical circuits. In this paper, we develop an optimization algorithm for crosstalk noise reduction that explores solution space effectively under delay constraints, and construct a crosstalk noise reduction method using the noise estimation method [19] and the transistor sizing framework [15], [16]. Thanks to the framework, the estimation method and the algorithm, our method can estimate and optimize crosstalk noise with interconnects unchanged. The proposed method can be applied to the circuits optimized by other methods, such as interconnect optimization and buffer insertion, and it further reduces the risk of crosstalk noise.

Upsizing and downsizing drivers have different mer-
Its and demerits although both approaches can optimize crosstalk noise. The upsizing approach does not degrade the robustness against random manufacturing variation but increases power dissipation and temperature, where it is known that random manufacturing variation becomes smaller as transistor size becomes larger. On the other hand, our approach decreases power dissipation and temperature, although it may degrade robustness against random manufacturing variability.

This paper is organized as follows. Section 2 explains an estimation method of crosstalk noise. Section 3 shows an optimization algorithm for crosstalk noise reduction. Section 4 demonstrates some experimental results. Finally, Sect. 5 concludes the discussion.

2. Crosstalk Noise Estimation

This section discusses crosstalk noise estimation. The proposed method utilizes the 2-π noise model for crosstalk estimation [19], and we explain it briefly. We next discuss a noise estimation method for a net with multiple aggressors based on superposition considering timing window.

2.1 Overview of Crosstalk Noise Estimation

In practical circuits, many interconnects couple with multiple aggressors, i.e., with multiple aggressors. We estimate the peak noise voltage caused by each aggressor respectively, and calculates the maximum noise voltage at the sink by superposition. The superposition of the noise voltage is discussed in Sect. 2.2.

The victim net with one aggressor is represented as two partially-coupled interconnects (Fig. 1). The partially-coupled interconnects in Fig. 1 are modeled as an equivalent circuit shown in Fig. 2. \( R_{a1} \) is the effective driver resistance of the victim net. The node \( n_{v2} \) corresponds to the center of the coupling portion of the victim interconnect, and \( n_{a2} \) similarly corresponds to the center of the coupling portion of the aggressor interconnect. \( R_{v2} \) is the resistance between the source and \( n_{v2} \), and \( R_{a3} \) is the resistance between \( n_{a2} \) and the sink. \( C_{c} \) is the coupling capacitance between the victim and the aggressor. The capacitances \( C_{a1} \), \( C_{v2} \) and \( C_{a3} \) are represented as \( C_{1}/2 \), \( (C_{1}+C_{2})/2 \), and \( C_{2}/2+C_{l} \) respectively, where \( C_{1} \) is the wire capacitance from the source to \( n_{a2} \), \( C_{2} \) is the wire capacitance from \( n_{a2} \) to the sink, and \( C_{l} \) is the capacitance of the receiver. The parameters of the aggressor wire, \( R_{a1} \), \( R_{a2} \), \( R_{a3} \), \( C_{a1} \), \( C_{a2} \), \( C_{a3} \), are determined similarly.

Reference [19] also develops a method that can apply interconnects with branches into the model circuit of Fig. 2. We here omit the explanation of this application method.

In the circuit of Fig. 2, the peak voltage \( V_{\text{peak}} \) is expressed as follows [19].

\[
V_{\text{peak}} = \frac{(R_{a1} + R_{a2})C_{c}V_{dd}}{\tau_v} \left( \frac{\tau_v}{\tau_a} \right)^{-\frac{n_{a2}}{n_{v2}}},
\]

where

\[
\tau_v = R_{a1}(C_{a1} + C_{v2} + C_{c} + C_{a3}) + R_{v2}C_{v2},
\]

\[
\tau_a = R_{a1}(C_{a1} + C_{a2} + C_{c} + C_{a3eff}) + R_{a2}(C_{a2} + C_{c} + C_{a3eff}),
\]

\[
C_{a3eff} = C_{a3} \left( 1 - e^{-T/R_{a1}C_{a3}} \right),
\]

\[
T = R_{a1}(C_{a1} + C_{a2} + C_{c} + C_{a3}) + R_{a2}(C_{a2} + C_{c} + C_{a3}).
\]
a transition may occur and is defined as the range between \( EAT_i \) and \( LAT_i \). \( EAT_i \) is the earliest time of signal arrival at the output of cell \( l \), and \( LAT_i \) is the latest arrival time. \( EAT_i \) and \( LAT_i \) are calculated as follows.

\[
EAT_i = \min_{m \in FI(l)} \{ EAT_{m} + d_{m,l}^{\text{min}} \}, \tag{6}
\]

\[
LAT_i = \max_{m \in FI(l)} \{ LAT_{m} + d_{m,l}^{\text{max}} \}, \tag{7}
\]

where \( FI(l) \) is the set of the fan-in cells of cell \( l \). \( d_{m,l}^{\text{min}} \) represents the minimum delay between the output of cell \( m \) to the output of cell \( l \) in the case that the aggressors and the victim change in the same transition direction simultaneously. Similarly, \( d_{m,l}^{\text{max}} \) is the maximum delay in the case that the transition direction of the victim is opposite to those of the aggressors.

The maximum noise voltage at the \( i \)-th sink of the victim net at time \( t \) is represented as follows.

\[
V_{\text{max},i}(t) = \sum_{j} k(t) \cdot V_{\text{peak},j \rightarrow i}, \tag{8}
\]

\[
k(t) = \begin{cases} 
1 & \text{if } EAT_j \leq t \leq LAT_j \\
0 & \text{otherwise}
\end{cases} \tag{9}
\]

where \( n \) is the number of the aggressors, and \( V_{\text{peak},j \rightarrow i} \) is the noise voltage at the \( i \)-th sink caused by the \( j \)-th aggressor. We sweep time \( t \), and find the maximum noise voltage at each victim net.

We here use a simple method [20] for estimating the maximum delay \( d_{m,l}^{\text{max}} \) and the minimum delay \( d_{m,l}^{\text{min}} \). Other methods, such as Ref. [21], can be also used. When we have to estimate timing window more tightly considering the dependence of \( d_{m,l}^{\text{max}} \) and \( d_{m,l}^{\text{min}} \) on the timing window, we should calculate timing window iteratively [22]. In this paper, this iterative calculation is not executed simply because of an implementation matter. There are no technical limitations. Reference [20] indicates that the upper bound of \( d_{m,l}^{\text{min}} \) can be estimated as follows; all coupling capacitances are converted into the 3X capacitances to the ground, and then cell delay and wire delay are calculated. As for \( d_{m,l}^{\text{max}} \), coupling capacitances are replaced with the \(-1\)X capacitances to the ground. We utilize those upper and lower bound for \( d_{m,l}^{\text{max}} \) and \( d_{m,l}^{\text{min}} \) in this paper. Other sophisticated gate delay model [23], [24] also can be used, as long as the computational cost permits.

3. Optimization Algorithm

In this section, an optimization algorithm for crosstalk noise reduction is discussed. The proposed algorithm reduces crosstalk noise under delay and transition time constraints. First, an optimization algorithm for a localized problem that includes one victim net and its adjacent nets is explained. This section then shows the overall algorithm that builds and solves the local optimization problems, considering the global optimality under delay constraints.

3.1 Optimization Algorithm in Each Victim Net

First, the noise reduction algorithm for each victim net is explained. The proposed method downsizes the drivers of the adjacent aggressor wires in order to reduce the amount of crosstalk noise at the victim wire. When the driving strength of the aggressor wire becomes weak, i.e. the driver resistance \( R_{a} \) becomes large, the time constant of the aggressor voltage source \( \tau_a \) increases (Eq. (4)). Then the maximum noise voltage \( V_{\text{peak}} \) (Eq. (1)) at the victim net consequently decreases. This relationship can be revealed from the partial derivative of \( V_{\text{peak}} \) respect to \( R_{a} \) as follows.

\[
\frac{\partial V_{\text{peak}}}{\partial R_{a}} = \frac{\tau_a(C_a + C_e)}{(\tau_e - \tau_a)^2} \left( \log \frac{\tau_e}{\tau_a} - \frac{\tau_a}{\tau_v} + 1 \right) V_{\text{peak}} \leq 0. \tag{10}
\]

In order to choose the driver of the aggressor wire to be downsized efficiently, a measure \( \text{priority} \) is devised.

\[
\text{priority}_i = \text{slack}_i \cdot \sum_{j} V_{\text{peak},j \rightarrow i}, \tag{11}
\]

where \( V_{\text{peak},j \rightarrow i} \) is the peak noise voltage at the \( j \)-th sink caused by the \( i \)-th aggressor net, and \( m \) is the number of sinks. The value \( \text{slack}_i \) represents the timing margin at the \( i \)-th aggressor net, and is defined as the time difference between the required time and the arrival time [25]. The measure \( \text{priority}_i \) becomes large in the case that the \( i \)-th adjacent net causes a large amount of noise and the timing constraint at the \( i \)-th aggressor net is not tight. Using this measure, the proposed algorithm can find the aggressor net efficiently that has strong influence on the crosstalk noise at the victim net yet has little influence on the circuit delay.

One of the difficulties in crosstalk noise optimization is that each victim net also becomes an aggressor from the opposite standpoint. When the driver of an aggressor is downsized for reducing noise at the victim net, the noise at the aggressor may increase intolerably. We therefore calculate the peak noise voltages at both the victim and the aggressor wires, and find a proper driver size of the aggressor. For this purpose, we here minimize the sum of the squared noise voltage at the aggressor and the squared noise voltage at the victim.

**Step 1:** Calculate \( \text{priority} \) (Eq. (11)) for each adjacent aggressor net, and put all the aggressor nets into list \( L_i \).

**Step 2:** Choose the aggressor net with the maximum \( \text{priority} \) from list \( L_i \).

**Step 3:** Downsize the driver of the chosen aggressor net within the limit that the delay constraints and the transition time constraints are satisfied. The best size of the driver is decided such that the value of \( (V_e^2 + V_a^2) \) becomes the smallest, where \( V_e \) is the noise voltage at the victim net, and \( V_a \) is the noise voltage at the aggressor net. In the practical implementation, we try several driver sizes and calculate the value of \( (V_e^2 + V_a^2) \) and the
circuit delay. We then choose the best size from those
types without delay violation. Remove the aggressor
net from \( L_i \).

**Step 4:** If the noise voltage becomes smaller than the target
value \( V_{\text{target}} \), or if the list \( L_i \) becomes empty, finish the
optimization procedure. Otherwise go back to **Step 2**.
The value \( V_{\text{target}} \) is explained in the following section.

3.2 Overall Optimization Algorithm

Section 3.1 discusses the optimization algorithm for the lo-
icalized problem that contains one victim net and its adja-
cent aggressor nets. Next, the overall algorithm is discussed.
This algorithm aims to reduce both the maximum noise volt-
age in a circuit and the number of nets whose noise is large.

The optimization iterates the following procedure from
**Stage 1** to **Stage 4** for several times, as parameter \( \text{threshold} \)
gradually decreases. The parameter \( \text{threshold} \) is used for
selecting the nets to be optimized, and it ranges from 0 to 1.

**Stage 1:** Calculate the crosstalk noise at each net in the cir-
cuit.

**Stage 2:** Find the maximum voltage of crosstalk noise \( V_{\text{max}} \)
in the circuit, and put the nets whose noise voltages are
larger than \( V_{\text{target}} \), which is the product of \( \text{threshold} \) and the maximum noise voltage in the circuit \( V_{\text{max}} \), are chosen as the optimization can-
didates. In the beginning, \( \text{threshold} \) is set close to 1 in order
to reduce the maximum noise voltage intensively. In the end, \( \text{threshold} \) is set close to 0, and the most of the nets in
the circuit are optimized.

**Stage 3:** Choose the net with the maximum noise voltage
in the list \( L_o \), and execute the optimization explained in
Sect. 3.1. The value of \( V_{\text{target}} \) is given to the optimization
as the target value. Remove the net from the list
\( L_o \), and update the information of timing window.

**Stage 4:** If the list \( L_o \) becomes empty, finish the optimiza-
tion procedure. Otherwise go back to **Stage 3**.

When the timing constraints are given, the timing mar-
gin at each net should be utilized efficiently for reducing
the crosstalk noise. Therefore the sequence of the nets to
be optimized is critical and essential to obtain high-quality
circuits. In order to reduce the maximum noise voltage,
the proposed algorithm gives priority to the nets with large
noise. **Stage 2** excludes the nets whose noise voltages are
smaller than \( V_{\text{target}} \) from the the optimization candidates. In
**Stage 3**, the nets are optimized in order of the amount of
noise voltage.

In **Stage 3**, the target noise value \( V_{\text{target}} \) is given to the
localized optimization problem, in order to control the local
optimization from the viewpoint of global optimality. The
optimization result that the noise voltage is minimized in
the localized problem may incur a bad local-minimum solu-
tion globally. This is because the timing margins, which
may be utilized for reducing the noise at other nets, are
wasted. The proposed algorithm hence stops the local op-
timization when the noise voltage becomes smaller than the
target value in **Step 4**. Thanks to the good sequence of the
net to be optimized and setting the target noise value, the
proposed method can reach a good solution under the delay
constraints.

4. Experimental Results

This section shows the optimization results for crosstalk
noise reduction. The circuits used for the experiments are
an ALU in a DSP for mobile phone [26] (dsp_alu) and the
circuits included LGSynth93 benchmark sets (des). These
circuits are synthesized by a commercial logic synthesis
tool. The circuit scale of dsp_alu is 12547 cells, and the
number of cells in des is 3414. The placement and rout-
ing are performed by a commercial tool with an option that
minimizes interconnect length and without any options for
crosstalk noise reduction. The layout area of dsp_alu is 5.3
(2.3 × 2.3) mm², and the area of des is 0.64 (0.8 × 0.8) mm².

RC trees of interconnects are extracted from the layouts by
a quasi-3D RC extract tool [28]. The coupling capacitances
below 10 fF are extracted as the capacitance to the ground,
where the coupling capacitance of 10 fF corresponds to the
length of 230 \( \mu \)m. The supply voltage is 3.3 V.

Cell layouts are generated using VARDS [17], [18] in
a 0.35 \( \mu \)m process with three metal layers. The layout gen-
eration system VARDS can vary transistor widths in a cell
while keeping the location of each pin. Exploiting this fea-
ture, the proposed method optimizes a detail-routed circuit
without any wire modifications [15], [16]. The height of the
generated cells is 13 interconnect-pitches. In transistor siz-
ing, MOSFETs are downsized within the range that VARDS
can generate cell layouts. The maximum transistor width of
standard driving-strength (×1) cells is 6.2 \( \mu \)m. The transistor
width can be reduced to 0.9 \( \mu \)m. We characterize the resis-
tance of a CMOS gate as 4 values; \( R_{Dp} \) and \( R_{Dn} \) are the driv-
ing resistances of the pull-up PMOS part and the pull-down
NMOS part respectively, and \( R_{Hp} \) and \( R_{Hn} \) are the ho-
ling resistances. \( R_{Dp} \) and \( R_{Dn} \) are decided such that the propaga-
tion delay becomes the same with circuit simulation results
[29]. \( R_{Hp} \) and \( R_{Hn} \) are evaluated by the operating condition
analysis of circuit simulation.

The initial circuits used for the experiments are de-
signed for minimizing the circuit delay in a usual cell-base
design style. The circuits are optimized such that the circuit
delay does not increase. The given constraint of the transi-
tion time is 1.0 ns and it is the same with the constraint given
for the initial circuit design. The \( \text{threshold} \) value used in the
optimization algorithm of Sect. 3.2 is varied from 0.98 at the
beginning to 0.5 at the end of the optimization. The iteration
number of the optimization of Sect. 3.2 is 6 for both circuits.

In the current implementation, when we execute **Step 3** of
the proposed algorithm explained in Sect. 3.1, we try five
transistor widths between the current transistor width and the
minimum transistor width.
#Table 1: Noise optimization results.

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Maximum Noise (V)</th>
<th>CPU Time (s)</th>
<th>#Pairs†</th>
<th># Cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial</td>
<td>Optimized</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>des</td>
<td>0.40</td>
<td>0.19</td>
<td>12</td>
<td>1018</td>
</tr>
<tr>
<td>dspalu</td>
<td>1.00</td>
<td>0.50</td>
<td>604</td>
<td>82368</td>
</tr>
</tbody>
</table>

†: number of pairs of an aggressor and a victim in a circuit.

Table 1 demonstrates the crosstalk noise optimization results. Figures 3 and 4 show the distributions of the maximum noise voltage before and after the optimization. In des circuit, the maximum noise voltage is reduced from 0.40 V to 0.19 V by 53%. In dspalu circuit, the maximum noise is reduced from 1.00 V to 0.50 V by 50%. The distribution is also shifted in the direction that the noise voltage decreases. The number of nets whose noise voltages are over 0.5 V is decreased from 59 to 3. After the detailed-routing, the crosstalk noise can be reduced considerably by downsizing the transistors inside cells while keeping the interconnects without delay violation. The proposed method reduces crosstalk noise as much as possible after detail-routing, and hence the number of pairs whose slack is small may cause a delay violation when delay variation exists. If this problem is significant, we should use statistical static timing analysis for timing verification inside the optimization like Ref. [31].

The CPU times required for the optimization on an Alpha Station (600 MHz) are 12 seconds in des (3.4k cells), and 604 seconds in dspalu (13k cells). Here the CPU time for reading circuit and interconnect information files is excluded. The basic optimization of the proposed algorithm in Sect. 3.1 is executed for each pair of an aggressor and a victim, and hence the required computational time is basically proportional to the number of pairs of an aggressor and a victim, although it also depends on circuit structure, timing constraints and so on.

Figure 5 shows the slack distribution before and after the optimization in dspalu circuit. The proposed method downsizes drivers whose timing constraint is not tight for crosstalk noise reduction, and hence the number of gates whose slack is small increases. Reference [30] indicates that increasing the number of gates whose slack is small may cause a delay violation when delay variation exists. If this problem is significant, we should use statistical static timing analysis for timing verification inside the optimization like Ref. [31].

Figure 6 shows two peak noise distributions after the optimizations with different setup. In Case 1, the number of iterative optimization of Sect. 3.2 is 6 and threshold at each iteration is set as shown in Table 2. As for Case 2, the number of iteration is 1, and threshold is 0.5. In Case 2, the number of nets whose peak voltage is above 0.4 V is 116, whereas it is 8 in Case 1. The iteration with gradual decrease of threshold is important to obtain a good opti-
Table 2  Noise optimization progress at each iteration count.

<table>
<thead>
<tr>
<th>Iteration Count</th>
<th>threshold</th>
<th>Maximum Noise (V)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>des</td>
<td>dsp</td>
</tr>
<tr>
<td>Initial</td>
<td>-</td>
<td>1.00</td>
</tr>
<tr>
<td>1</td>
<td>0.98</td>
<td>0.93</td>
</tr>
<tr>
<td>2</td>
<td>0.95</td>
<td>0.76</td>
</tr>
<tr>
<td>3</td>
<td>0.90</td>
<td>0.68</td>
</tr>
<tr>
<td>4</td>
<td>0.80</td>
<td>0.54</td>
</tr>
<tr>
<td>5</td>
<td>0.70</td>
<td>0.50</td>
</tr>
<tr>
<td>6</td>
<td>0.50</td>
<td>0.50</td>
</tr>
</tbody>
</table>

Fig. 7  Accuracy of peak noise estimation for each pair of an aggressor and a victim (des).

Fig. 8  Histogram of estimation error (des).

5. Conclusion

This paper proposes a method to reduce crosstalk noise as much as possible by transistor sizing after detail-routing. The proposed method optimizes the detail-routed circuits such that MOSFETs inside cells are downsized with interconnects unchanged. The effectiveness of the proposed method is experimentally verified using 2 benchmark circuits. The maximum noise voltage is reduced by more than 50% without delay increase after detail-routing, which reduces the failure risk of crosstalk noise and contributes to high-reliability LSI design.

Acknowledgement

This work is supported in part by the 21st Century COE Program (Grant No. 14213201).

References

We discuss

\[ V_{\text{peak}} = \frac{A}{\tau_v} \left( \frac{\tau_v}{\tau_a} \right)^{1-\frac{1}{n}}. \]  

(A·1)

where \( A \) is \((R_d + R_c)C_v V_{dd}\). We replace \( \frac{\tau_v}{\tau_a} \) with 1 + \( \delta \), and then Eq. (A·1) becomes

\[ V_{\text{peak}} = \frac{A}{\tau_v} \left( 1 + \delta \right)^{-\frac{1}{2}}. \]  

(A·2)

We discuss \( V_{\text{peak}} \) when \( \tau_a \) becomes equal to \( \tau_v \), i.e. we examine the following equation.

\[ V_{\text{peak}}|_{\tau_a=\tau_v} = \frac{A}{\tau_v} \cdot \lim_{\delta \to 0} (1 + \delta)^{-\frac{1}{2}}, \]  

(A·3)

Here, the base of natural logarithms e is defined as follows.

\[ e = \lim_{n \to \infty} \left( 1 + \frac{1}{n} \right)^n, \]  

(A·5)

\[ e = \lim_{\delta \to 0} (1 + m)^\delta. \]  

(A·6)

Therefore Eq. (A·4) becomes

\[ V_{\text{peak}}|_{\tau_a=\tau_v} = \frac{A}{\tau_v} \cdot \frac{1}{e}. \]  

(A·7)

Equation (1) can be calculated even when \( \tau_a = \tau_v \).

Appendix

We prove that Eq. (1) is valid when \( \tau_a = \tau_v \). Equation (1) is rewritten as

\[ V_{\text{peak}} = \frac{A}{\tau_v} \left( \frac{\tau_v}{\tau_a} \right)^{1-\frac{1}{n}}. \]  

(A·1)

Masanori Hashimoto received the B.E., M.E. and Ph.D. degrees in Communications and Computer Engineering from Kyoto University, Kyoto, Japan, in 1997, 1999, and 2001, respectively. Since 2001, he was an Instructor in Department of Communications and Computer Engineering, Kyoto University. Since 2004, he has been an Associate Professor in Department of Information Systems Engineering, Graduate School of Information Science and Technology, Osaka University. His research interest includes computer-aided-design for digital integrated circuits, and high-speed circuit design. He is a member of IEEE, ACM and IPSJ.

Hidetoshi Onodera received the B.E., M.E., and Dr.Eng. degrees in Electronic Engineering from Kyoto University, Kyoto, Japan, in 1978, 1980, 1984, respectively. He joined the Department of Electronics, Kyoto University, in 1983, and currently a Professor in the Department of Communications and Computer Engineering, Graduate School of Informatics, Kyoto University. His research interests include design technologies for Digital, Analog, and RF LSIs, with particular emphasis on high-speed and low-power design, design and analysis for manufacturability, and SoC architectures. Dr. Onodera has been the Program Chair and General Chair of the ACM/IEEE International Conference on Computer-Aided Design (ICCAD) in 2003 and 2004, respectively, and served on the technical program committees for international conferences including DAC, DATE, ASPDAC, CICC, etc. He was the Chairman of the IEEE Kansai SSCS Chapter from 2001 to 2002, and the Chairman of the Technical Group on VLSI Design Technologies, IEICE, Japan, from 2000 to 2001.