Modelling of Guardband Reduction on Design Area

Ning Jin

Advisor: Professor Puneet Gupta

Department of Electrical Engineering, University of California, Los Angeles

jinning@ucla.edu, puneet@ee.ucla.edu

***Abstract* — Considering fabrication variations, timing guardband is introduced to ensure design reliability. However, trade-off exists between guardband and design performance, especially the chip area discussed in this paper. To predict the variation of total chip area with guardband reduction, we construct the model from the basic Elmore delay theory and optimize the gate sizing and buffering to meet the setup timing constrains. To determine the coefficients in the model, we utilize the empirical results from the Synthesis, Placement and Routing implementation flow and employ nonlinear fitting method. Furthermore, the testing on our model exhibits high accuracy and capability of predicting.**

I. Introduction

When technology scales beyond 90 nm, higher levels of device parameter variations introduced during fabrication are posing a major challenged to the future high performance VLSI design [1]. To ensure proper functionality of the design, timing guardbands are introduced [2]. However, the pessimism coming along with timing guardbands is becoming a big concern in industry. To tackle the problem, various design methodologies have been proposed or implemented, such as improving manufacturability [3], advanced lithography techniques [4], and variation reduction approaches [5]. Nevertheless, every technique has not only pros but also cons, which is at the expense of performance degradation, such as area and power increase. Thus, it is important to quantify the trade-off between guardband reduction and performance degradation.

Previous work has been done on quantification of guardband reduction based on experiment results [6]. They proposed to run synthesis, place and route (SPR) simulations on given design with cell libraries with different guardband while making sure that the setup and hold time can be met. Then they assessed the influence of model guardband reduction on design performance metrics from the SPR implementation flow, such as area, dynamic power, leakage power and routed wirelength. However, the shortcoming of this approach is that the results are highly design-dependent, which means lack of prediction and extensive run time. Facing the problem, another group [7] from UCLA proposed an idea that a more general model can be derived based on the basic Elmore delay model [8]. Stage by stage, area is optimized by estimating the number and size of buffers while ensuring that setup time constrains can be met. Isolating guardband model from real design gives this model the superiority of generalization and simplicity. Whereas, it's idealization leads to deviation from reality and causes unnecessary errors, such as noise from EDA tool or technology dependent factors.

In this work, a model is proposed to quantify area benefit of delay change of individual logic gate based on both derivation from Elmore delay model and input from SPR implementation flow at one corner. As a combination, this model has both solid physical origin and input from real technology and design. So it can provide us with prediction for the area benefit of other designs with confidence while saving time.

II. Modeling

In this section, the method used to derive the relationship between critical path (CP) (defined as the paths with slack value within 5% of clock cycle) area and delay of individual logic gate is described. Thus, the guardband reduction model is further built up to predict the design area change.

A. Delay Modelling

We assume the CP delay is the sum of the delay of individual logic stage in the path. Similar to the analysis in R. S. Ghaida's paper [7], each logic stage is modelled as a series of interconnected RC π-circuits as shown in **Fig. 1** [9]. Then, based on Elmore's delay model, the following expression can be derived as the delay for one logic stage:

$$\begin{aligned} T_{ls} = 0.69 \Big[& \frac{C_{int}R_{int}}{2(k+1)} + \frac{k}{k+1} C_{int} \frac{R_o}{h} + kR_oC_{o,out} \\ & + (k-1)R_oC_{o,in} + R_gC_{g,out} \\ & + \frac{R_o}{h} C_{g,in} + R_g h C_{o,in} \\ & + \frac{k}{k+1} R_{int} h C_{o,in} + \frac{C_{int}R_g}{k+1} \\ & + \frac{R_{int}C_{g,in}}{k+1} \Big] \end{aligned} \quad (1)$$

where k and h are the number of repeaters and scaling factor w.r.t. minimum size inverter, R_o, $C_{o,in}$ and $C_{o,out}$ are the output resistance, input and output capacitance of the minimum size inverter in the library (technology-dependent); R_g, $C_{g,in}$ and $C_{g,out}$ are output resistance, input and output capacitance of logic gate under examination (design-dependent); while R_{int} and C_{int} are interconnect resistance and capacitance (both technology and design dependent), and R_{int} is given by:

$$R_{int} = \frac{\rho L}{W \times T} \quad (2)$$

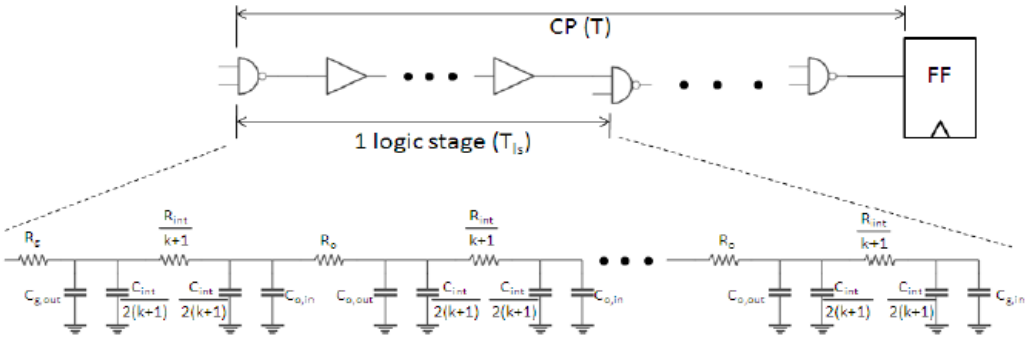


Fig. 1 Illustration of critical path delay model.

where ρ is the resistivity of copper, and L, W and T are interconnect length, width and thickness respectively, which are shown in **Fig. 2**.

For C_{int}, the three-metal lines and one-ground plane model is introduced as illustrated in **Fig. 2** [10].

$$C_{int} = (2C_m + C_g) \times L \tag{3}$$

where C_m and C_g are line-to-line and line-to-ground capacitances per length assuming one-ground plane. And they are calculated using the empirical model from [10]:

$$\begin{aligned}\frac{C_m}{\epsilon} &= 1.064\left(\frac{T}{S}\right)\left(\frac{T+2H}{T+2H+0.5S}\right)^{0.695} \\ &+ \left(\frac{W}{W+0.8S}\right)^{1.4148}\left(\frac{T+2H}{T+2H+0.5S}\right)^{0.804} \\ &+ 0.831\left(\frac{W}{W+0.8S}\right)^{0.055}\left(\frac{2H}{2H+0.5S}\right)^{3.542}\end{aligned} \tag{4}$$

$$\frac{C_g}{\epsilon} = \frac{W}{H} + 1.086\left(1 + 0.685e^{\frac{-T}{1.3435}} - 0.9964\,e^{\frac{-S}{1.421H}}\right) \tag{5}$$

where S and H are the spacing between lines and height of lines from ground separately, also as shown in **Fig. 2 below**.

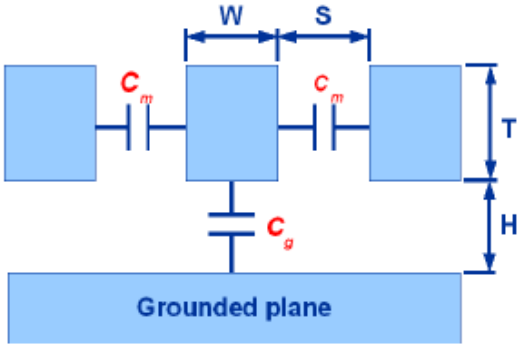


Fig. 2 Interconnect resistance and capacitance approximation model.

Now, we assume that during SPR, the automation tool takes two steps to meet the setup time requirement: first sizing the original logic gate and then adding buffers in-between logic gates to further improve timing, which are shown below separately (experimental foundation can be seen in part IV):

1) Sizing of Logic Gates: Now, we assume that no buffer added, so *k* and *h* are fixed at 0 and 1 separately. Substituting *k* and *h* into Eq. (1) gives:

$$\begin{aligned}T_{ls} = 0.69\Big[&\frac{C_{int}R_{int}}{2} - R_oC_{o,in} + R_gC_{g,out} + R_oC_{g,in} \\ &+ R_gC_{o,in} + C_{int}R_g + R_{int}C_{g,in}\Big]\end{aligned} \tag{6}$$

To calculate the optimum sizing factor, we take first order derivative of T_{ls} with regards to $C_{g,in}$, noticing the relationships below:

- Gate delay equation:

$$T_{gate} = 0.69 \times R_g \times C_{g,out} \tag{7}$$

- Fanout equation:

$$F = \frac{C_{g,out}}{C_{g,in}} \tag{8}$$

where F is the fanout load of the given logic gate, which is equal to the logic fanout by assuming that all gates in CPs are of equal size; T_{gate} is the delay of the logic gate. We use T_{gate} value corresponding to the minimum input signal slew and maximum C_L for more accurate approximation of R_g. Since T_{gate} is provided for rise and fall transitions, we use the average value as approximation. The method we use to extract parameter T_{gate} will be described in detail in part III.

Combining Eq. (7) and (8) gives:

$$R_g = \frac{T_{gate}}{0.69 \times C_{g,out}} = \frac{T_{gate}}{0.69 \times F \times C_{g,in}} \tag{9}$$

Taking first order derivative of R_g with regards to $C_{g,in}$ gives:

$$\frac{dR_g}{dC_{g,in}} = -\frac{T_{gate}}{0.69 \times F \times C_{g,in}^{\,2}} \tag{10}$$

Now, we can get:

$$\begin{aligned}\frac{dT_{ls}}{dC_{g,in}} = 0.69 \times \Big[&(R_o + R_{int}) \\ &-(C_{g,out} + C_{o,in} \\ &+ C_{int})\frac{T_{gate}}{0.69 \times F \times C_{g,in}^{\,2}}\Big]\end{aligned} \tag{11}$$

Setting Eq. (11) equal to zero gives:

$$(R_o + R_{int}) = (F \times C_{g,in(opt)} + C_{o,in} + C_{int}) \times \frac{T_{gate}}{0.69 \times F \times C_{g,in(opt)}^{\ 2}} \Rightarrow \frac{0.69 \times F \times (R_o + R_{int})}{T_{gate}} \times C_{g,in(opt)}^{\ 2} - F \times C_{g,in(opt)} - (C_{o,in} + C_{int}) = 0 \quad \Rightarrow \quad C_{g,in(opt)} = \frac{F + \sqrt{F^2 + 4 \times \frac{0.69 \times F \times (R_o + R_{int})}{T_{gate}} \times (C_{o,in} + C_{int})}}{2 \times \frac{0.69 \times F \times (R_o + R_{int})}{T_{gate}}} \tag{12}$$

where $C_{g,in(opt)}$ is the optimized input capacitance of the logic gate considering setup time constrains.

2) Buffering Between Logic Gates: Similar to the previous method used to optimize the size of logic gates, we can further improve timing by adding buffers of optimal number (k_{opt}) and scaling factor (h_{opt}). By setting $\frac{\delta T_{ls}}{\delta k}$ and $\frac{\delta T_{ls}}{\delta h}$ to zero, k_{opt} and h_{opt} are calculated to be:

$$k_{opt} \approx \sqrt{\frac{1}{R_o C_o}\left(\frac{C_{int}R_{int}}{2} C_{int}R_g + R_{int}C_{g,in(opt)}\right)} = \sqrt{\frac{1}{R_o C_o}\left(\frac{C_{int}R_{int}}{2} C_{int}\frac{T_{gate}}{0.69 \times F \times C_{g,in(opt)}} + R_{int}C_{g,in(opt)}\right)} \tag{13}$$

$$h_{opt} = \sqrt{\frac{\frac{k_{opt}}{k_{opt}+1} C_{int}R_o + R_o C_{g,in(opt)}}{\frac{k_{opt}}{k_{opt}+1} R_{int}C_{o,in} + R_g C_{o,in}}} = \sqrt{\frac{\frac{k_{opt}}{k_{opt}+1} C_{int}R_o + R_o C_{g,in(opt)}}{\frac{k_{opt}}{k_{opt}+1} R_{int}C_{o,in} + \frac{T_{gate}}{0.69 \times F \times C_{g,in(opt)}} C_{o,in}}} \tag{14}$$

Eq. (13) and (14) are calculated for all the CPs in the design to optimize delay of each logic stage and meet setup time requirement. After that, area of CPs can be estimated as a function of T_{gate} as following.

B. Area Modelling

1) Modelling of CP Area vs. Gate Delay: Based on the two parts mentioned above, we can model the area of CP vs. gate delay as:

$$A_{CP} = c_1 \times \frac{\sum_{stages} C_{g,in(opt)}}{C_{unit}} + c_2 \times A_{INV} \sum_{stages} (k_{opt} \times h_{opt}) + c_3 \tag{15}$$

where c_1 , c_2 *and* c_3 are coefficients to be fitted from the experiment results. A_{INV} is the area of minimum size inverter.

For $\sum C_{g,in(opt)}$, first of all we calculate the average $C_{g,in(opt)}$ of all the gates in CPs at the slowest gate delay library based on Eq. (12) and then sum up among all stages in CPs. The reason why we pick the input from the slowest corner is that it can make sure that all CPs needed to be considered are included during the modelling. The definition of each parameter is summarized in Form 1 below:

FORM I
PARAMETER DEFINATION

Symbol	Definition
F	Average logic fanout of all the instances in CPs
R_o	Output resistance of minimum size inverter
$C_{o,in}$	Average input capacitance of minimum size inverter
R_{int}	Average resistance of interconnect wires
C_{int}	Interconnect capacitance [11]
C_{unit}	Unit capacitance per area
T_{gate}	Average gate delay of all instances in CPs

Specifically, $T_{gate} = T_{gate(ave)} \times \frac{S}{f_{slow}}$, where *S* is the scaling factor of delay change and f_{slow} is the scaling factor of the slowest corner with positive slack for specific clock period (will be explained more in part III).

For $\sum k_{opt} \times h_{opt}$, it is calculated in a similar way as that of $\sum C_{g,in(opt)}$ mentioned above.

Substituting Eq. (12)-(14) into (15), we have

$$A_{CP} = c_1 \times \frac{F + \sqrt{F^2 + 4 \times \frac{0.69 \times F \times (R_o + R_{int})}{T_{gate}} \times (C_{o,in} + C_{int})}}{C_{unit} \times 2 \times \frac{0.69 \times F \times (R_o + R_{int})}{T_{gate}}} \times N_{CPI} + c_2 \times A_{INV} \times \sqrt{\frac{1}{R_o C_{o,in}}\left(\frac{C_{int}R_{int}}{2} C_{int}\frac{T_{gate}}{0.69 \times F \times C_{g,in(opt)}} + R_{int}C_{g,in(opt)}\right)} \times \sqrt{\frac{\frac{k_{opt}}{k_{opt}+1} C_{int}R_o + R_o C_{g,in(opt)}}{\frac{k_{opt}}{k_{opt}+1} R_{int}C_{o,in} + \frac{T_{gate}}{0.69 \times F \times C_{g,in(opt)}} C_{o,in}}} \times N_{CPI} + c_3 \tag{16}$$

where N_{CPI} is number of instances in all CPs at the slowest corner. In Eq. (16), CP area is implicitly expressed as a

function of L, F, N_{CPI}, f_{slow} and T_{gate}. Because L, F, N_{CPI}, f_{slow} and $T_{gate(ave)}$ are all input data from experiments at the slowest corner, now we successfully model CP area vs. the delay of individual logic gate.

To further investigate the relationship between A_{CP} and S, we can expand Eq. (16) by inserting the expression of R_o, R_{int}, C_{int} and T_{gate}:

$$A_{CP} = c_1 \times \frac{F + \sqrt{F^2 + 4 \times \frac{0.69 \times F \times \left(\frac{T_{gate(INV)} \times S}{0.69 \times C_{o,out}} + \frac{\rho L}{W \times T}\right)}{T_{gate(ave)} \times \frac{S}{f_{slow}}} \times (C_{o,in} + (2C_m + C_g) \times L)}}{2 \times \frac{0.69 \times F \times \left(\frac{T_{gate(INV)} \times S}{0.69 \times C_{o,out}} + \frac{\rho L}{W \times T}\right)}{T_{gate(ave)} \times \frac{S}{f_{slow}}}} \times N_{CPI} + c_2 \times A_{INV} \times \sqrt{\frac{1}{\frac{T_{gate(INV)} \times S}{0.69 \times C_{o,out}} \times C_{o,in}} \left(\frac{(2C_m + C_g) \times L \times \frac{\rho L}{W \times T}}{2} \times (2C_m + C_g) \times L \times \frac{T_{gate(ave)} \times \frac{S}{f_{slow}}}{0.69 \times F \times C_{g,in(opt)}} + \frac{\rho L}{W \times T} \times C_{g,in(opt)} \right)} \times \sqrt{\frac{\frac{k_{opt}}{k_{opt}+1} \times (2C_m + C_g) \times L \times \frac{T_{gate(INV)} \times S}{0.69 \times C_{o,out}} + \frac{T_{gate(INV)} \times S}{0.69 \times C_{o,out}} \times C_{g,in(opt)}}{\frac{k_{opt}}{k_{opt}+1} \times \frac{\rho L}{W \times T} \times C_{o,in} + \frac{T_{gate(ave)} \times \frac{S}{f_{slow}}}{0.69 \times F \times C_{g,in(opt)}} C_{o,in}}} \times N_{CPI} + c_3 \quad (17)$$

where $C_{g,in(opt)}$ and k_{opt} are given by Eq. (12) and (13):

$$C_{g,in(opt)} = \frac{F + \sqrt{F^2 + 4 \times \frac{0.69 \times F \times \left(\frac{T_{gate(INV)} \times S}{0.69 \times C_{o,out}} + \frac{\rho L}{W \times T}\right)}{T_{gate(ave)} \times \frac{S}{f_{slow}}} \times (C_{o,in} + (2C_m + C_g) \times L)}}{2 \times \frac{0.69 \times F \times \left(\frac{T_{gate(INV)} \times S}{0.69 \times C_{o,out}} + \frac{\rho L}{W \times T}\right)}{T_{gate(ave)} \times \frac{S}{f_{slow}}}} \quad (18)$$

$$k_{opt} = \sqrt{\frac{1}{\frac{T_{gate(INV)} \times S}{0.69 \times C_{o,out}} \times C_{o,in}} \left(\frac{(2C_m + C_g) \times L \times \frac{\rho L}{W \times T}}{2} \times (2C_m + C_g) \times L \times \frac{T_{gate(ave)} \times \frac{S}{f_{slow}}}{0.69 \times F \times C_{g,in(opt)}} + \frac{\rho L}{W \times T} \times C_{g,in(opt)} \right)} \quad (19)$$

Thus, we have the full relationship between A_{CP} and S.

2) Modelling of Chip Area vs. Gate Delay: Now, we assume that when we scale the library from the slowest corner to faster ones, the chip area change only comes from area change of CPs (sizing and buffering), while the non-CPs area is kept unchanged. So in that case, we have the relationship

$$A - A_o = A_{CP} - A_{CPO} \quad (20)$$

where A is the total chip area after SPR, A_o is the total chip area at the slowest library corner, A_{CP} is the CP area from Eq. (17) and A_{CPO} is the CP area from Eq. (17) with S = 1.4.

Dividing both sides of Eq. (20) by A_o, we have

$$\frac{A - A_o}{A_o} = \frac{A_{CP} - A_{CPO}}{A_o} \Rightarrow \frac{A}{A_o} = \frac{A_{CP} - A_{CPO}}{A_o} + 1 \quad (21)$$

By substituting Eq. (19) in to Eq. (21), we will have the relationship between $\frac{A}{A_o}$ and scaling factor S (which is inside A_{CP}), while all the other parameters are from the experiment results at the slowest library corner.

III. IMPLEMENTATION FLOW AND TESTCASES

A. Liberty Model Scaling

In our experiments, ARM 45nm 12S SOI standard-cell library is used at the P/V/T corner of ss/0.9V/-40C with maximum history effect, nominal extraction and maximum overlay. To simplify the problem, we model the effect of delay change instead of actual guardband reduction because guardband reduction is equivalent to keeping one corner fixed while changing the delay in the other library corner. So, starting from the original library, we scale the cell delay value in the library by a factor from 0.5 up to 1.4 and get 9 new library corners. Then the parameter T_{gate} is modelled as $T_{gate} = T_{gate(ave)} \times \frac{S}{f_{slow}}$, where $T_{gate(ave)}$ is the average gate delay of the design at the slowest possible library corner for given clock cycle, f_{slow} is the scaling factor of the slowest possible library corner with positive slack and S is the scaling factor with the range of from 0.5 to f_{slow}.

B. Timing-driven Implementation Flow

Fig. 3 illustrates the timing-driven SPR implementation flow used in our experiments. We start from the RTL codes of the design, do physical-driven synthesis with 9 library corners in parallel. Based on the results from synthesis, we execute the physical design steps with four routing layers and check timing at each step. At the event of timing violations, timing optimization will be performed to ensure setup time constrains are met. Because in the synthesis stage, different library corners are used, gate level netlists are generated with different total standard-cell areas. In that case, to get the best utilization ratio, iteration over the entire physical design steps is implemented until zero DRC error is ensured.

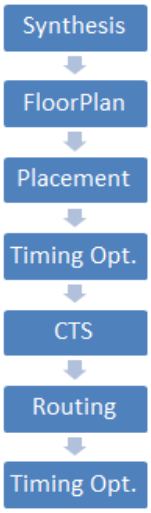


Fig. 3 Timing-driven SPR Implementation Flow.

C. Testcases and Tools

In our experiments, five benchmark designs are taken to run through the flow, which are *ae18*, *mips*, *spi*, *tv80* and *usb* obtained from the open-source site opencores.org [12]. Three

of them (*ae18*, *tv80* and *spi*) are taken to fit the coefficients in the model, while the other two (*usb* and *ae18*) are used to verify the accuracy of the model. The design description, approximate number of instances in each design and the clock period used in synthesis (Clk_s) and place and route (Clk_{pr}) are shown in Table 1. The rule of thumb to pick the clock period is to make sure that the timing slack is positive at the slowest library corner while the slack is not too large at the faster corners, in which case the data from different library corners can be comparable.

TABLE I
DESCRIPTION OF THE BENCHMARKS USED IN THE EXPERIMENT

	Design Description	# of total instances	Clk_s (ps)	Clk_{pr} (ps)
ae18	Clean room implementation of the Microchip PIC18 series CPU core	~4,000	1000	1000
mips	Soft processor core with five pipeline stages	~7,600	800	800
spi	SPI IP	~1,300	500	500
tv80	TV80 8-Bit Microprocessor Core	~2,800	800	700
usb	USB function core	~6,000	550	500

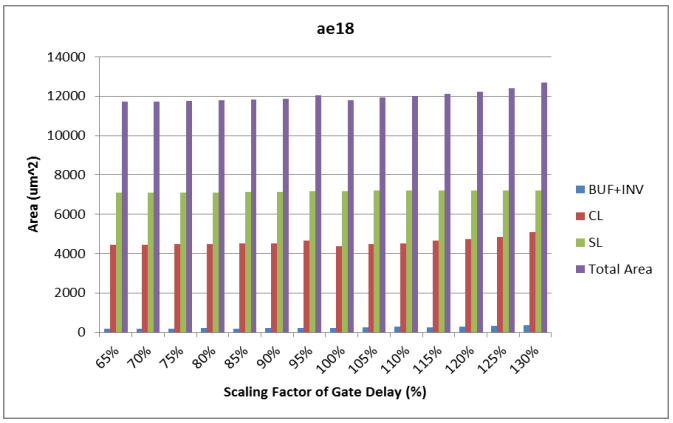
The EDA tools used in our experiments include Blaze v2010.1.0 for library scaling, *Cadence RTL Compiler* for synthesis and *Cadence SOC Encounter* to execute physical design flow.

IV. EXPERIMENT RESULTS AND DISCUSSION

In our experiments, for all five testcases (*ae18*, *mips*, *spi*, *tv80* and *usb*), we run the entire SPR implementation flow with 9 library corners of scaling factor from 0.5 to 1.4. So, in total, we have $9 \times 5 = 45$ sets of SPR runs.

A. Empirical Analysis of Chip Area Change

From the synthesis results, the total design area is decomposed into three categories: area of buffers and inverters, area of combinational logic (CL) and sequential logic (SL), while the interconnect area is not considered at this stage of study. To investigate the origin of the total design area change, the contribution and variation tendency of each part are plotted in Fig. 4 below as histogram.



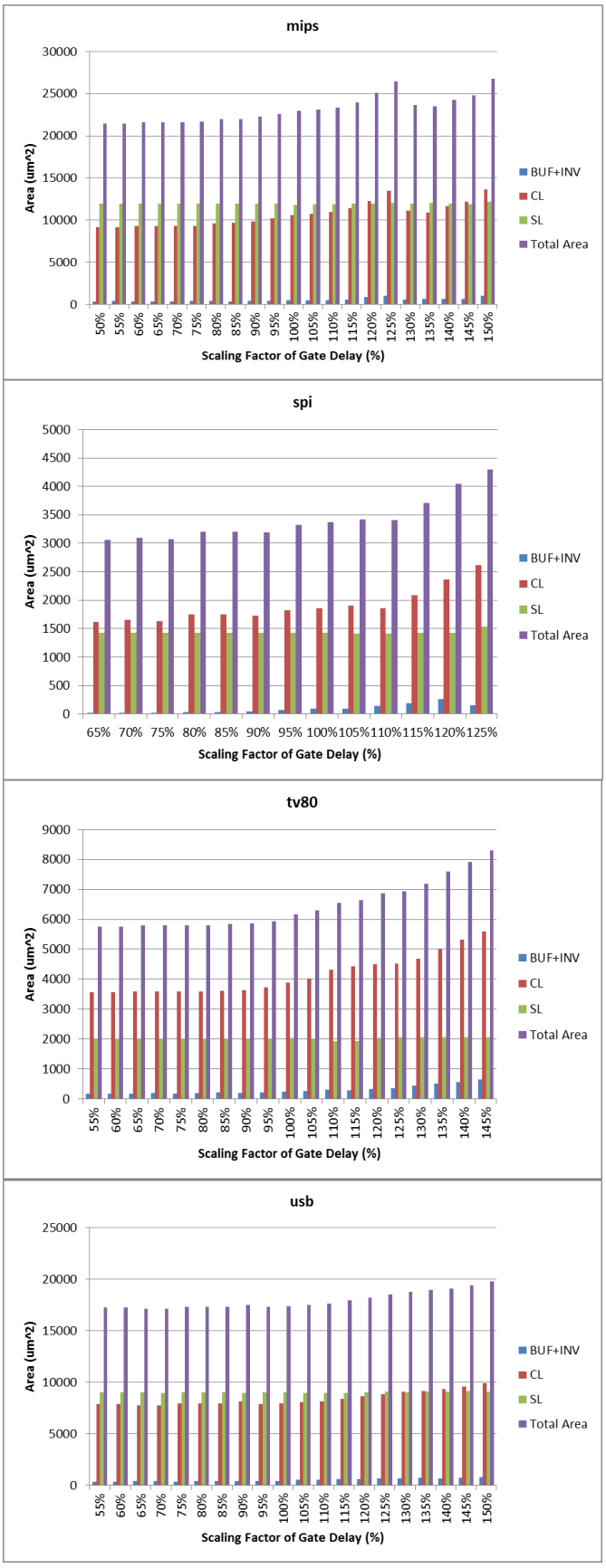


Fig. 4 Histogram of area contribution of 5 designs.

From Fig. 4, it can be seen that the total area change mostly comes from the area change of CL and inverter and buffer, while SL area change can be reasonably ignored. Thus the experiment results support the assumptions and model built up as described in part II.

B. Impact of Gate Delay on Design Area

Based on the implementation flow described in Part III, we can obtain the total chip area from *Encounter* after SPR corresponding to 9 library corners and 5 designs except design *spi*, the slowest corner of which is at scaling factor equals 1.35 so we have 8 corners for that design.

Fig. 5 and Fig. 6 below show the plot of the normalized (by chip area at the slowest library corner A_o) total area versus gate delay scaling factor *S* from synthesis and SPR implementation flow separately. For synthesis, only data with zero slack are considered valid and we normalize the total area by the one at the nominal corner, which means synthesis at the original library.

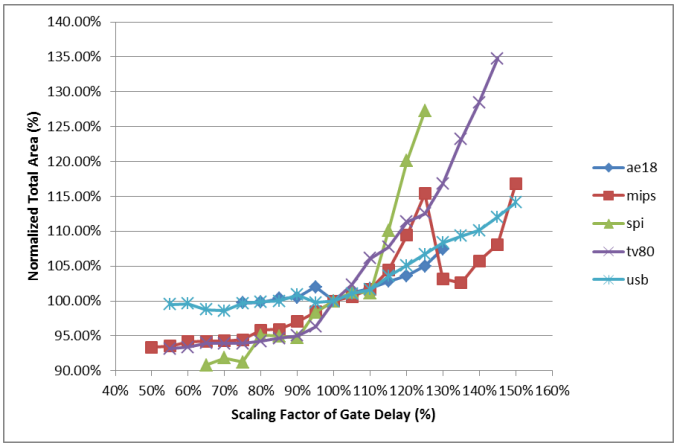


Fig. 5 Total chip area versus scaling factor of gate delay (from synthesis).

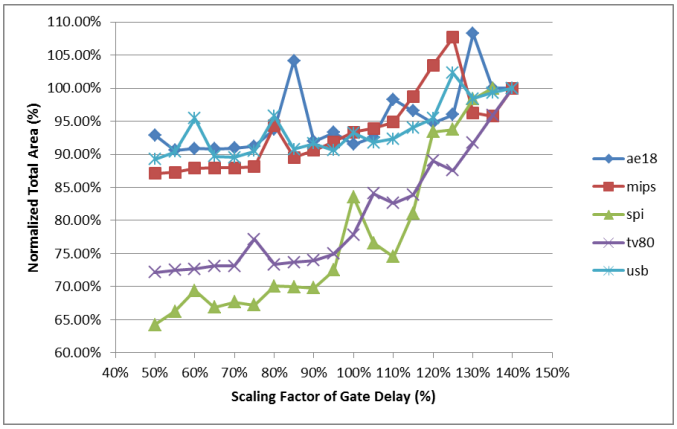


Fig. 6 Total chip area versus scaling factor of gate delay (from SPR).

It is obvious that when the scaling factor increases, which means the gate delay in the standard-cell library increases, the total chip area increases too if we ignore the noise obseved from the EDA tool. It can be explained that when the individual gate delay increases, the tool will automatically do sizing and buffering to help improve timing and fix setup violations. However, it does not necessarily size the gates up because sometimes smaller gates help with timing as well, which interprets the "noise" obseved in the plot.

C. Mathematical Pre-Modeling

First of all, from observation of the curves in Fig. 5, we notice that roughly the normalized total area increases exponentially with gate delay. So, mathematically, we fit the experiment data to see the possibility of modeling. To simplify the problem, we use only one parameter from the synthesis results to count for the difference of designs, which is the CPs area at the nominal library corner (named CPA).

We assume that the model can be in two forms:

$$\frac{A}{A_o} = (a_1 \times CPA^m + a_2) \times e^{(a_3 \times CPA \times S^n)} \tag{22}$$

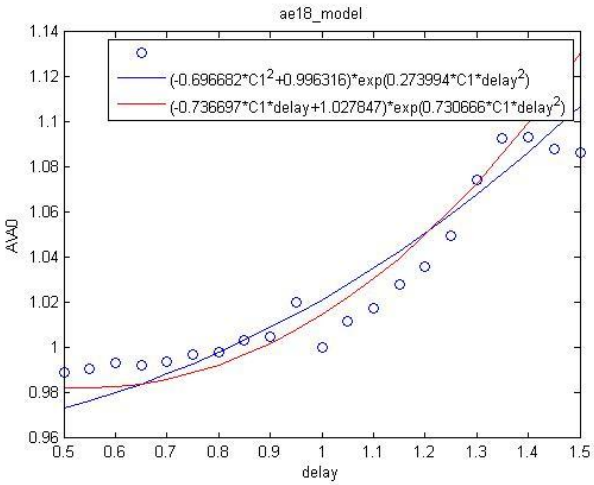
$$\frac{A}{A_o} = (b_1 \times CPA \times S^p + b_2) \times e^{(b_3 \times CPA \times S^q)} \tag{23}$$

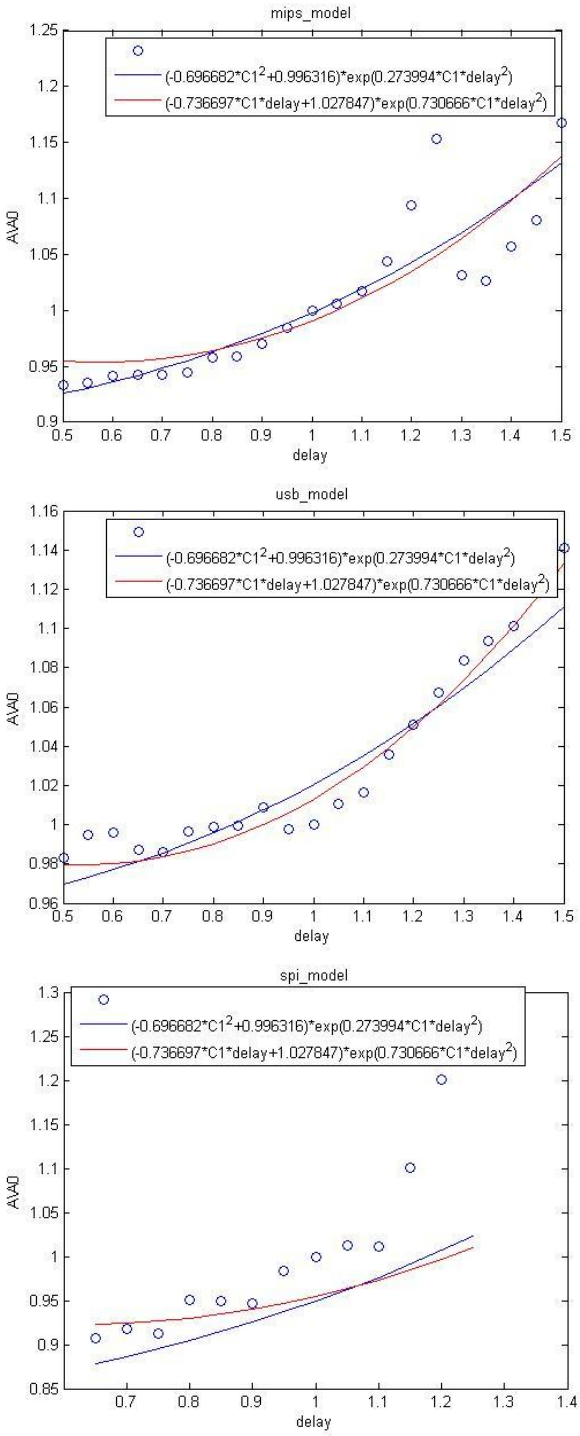
where a_1, a_2, a_3, b_1, b_2 and b_3 are coefficients to be fitted from the experiment results; *m*, *n*, *p* and *q* are the exponents to be optimized by testing. In this work, Newton method is applied to find the best nonlinear models, which are

$$\frac{A}{A_o} = (-0.6967 \times CPA^2 + 0.9963) \times e^{(0.2740 \times CPA \times S^2)} \tag{24}$$

$$\frac{A}{A_o} = (-0.7367 \times CPA \times S + 1.0278) \times e^{(0.7307 \times CPA \times S^2)} \tag{25}$$

Here, designs *ae18*, *mips* and *usb* are used to fit the coefficients while the model is tested by *spi* and *tv80*. The fitting curves are shown below:





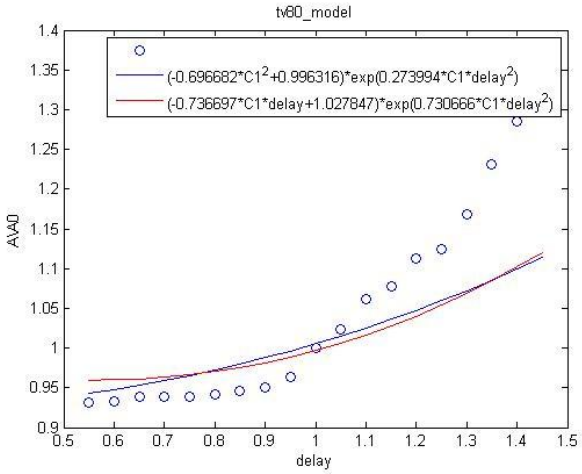


Fig. 7 Mathematical model fitting.

From Fig. 7, it can be seen that this simplified mathematical model roughly shows the trend of the area change, which projects the possibility of modeling from physical derivations.

D. Physical Model

Now, guardband model can be built based on the derivation from part II. As the model is dependent on data input (L, F, N_{CPI}, f_{slow} and $T_{gate(ave)}$) from the slowest corner, we summarize all those parameters for five designs in Table 2 shown below.

TABLE II
PARAMETERS AT SLOWEST LIBRARY CORNER

	$T_{gate(ave)}(s)$	F	$L\ (m)$	N_{CPI}	f_{slow}
ae18	0.1685e-9	4.2054	0.9218e-6	367	1.40
mips	0.1656e-9	4.2965	0.8567e-6	357	1.40
spi	0.1636e-9	3.5800	0.5853e-6	494	1.35
tv80	0.1753e-9	2.9372	0.5269e-6	656	1.40
usb	0.1674e-9	4.2870	0.8406e-6	320	1.40

To figure out the three coefficients in the model (Eq. (19)), we selected data from *ae18*, *tv80* and *spi* and applied the nonlinear fitting method. The nonlinear method used is "nlinfit" function in Matlab, which is implemented based on Levenberg-Marquardt algorithm [13]. Levenberg-Marquardt algorithm is an iterative technique that locates the minimum of a function that can be expressed as the sum of squares of nonlinear functions. This method can also be viewed as a combination of steepest descent and Gauss-Newton method and has become a standard technique for solving nonlinear least-squares problems. The termination tolerance is fixed at 1e-8 in this case.

In addition, roubust option is added, which means it iteratively refits a weighted nonlinear regression, where the weights at each iteration are based on each observation's residual from the previous iteration. In other word,at each iteration the nonlinear regression is a weighted version of the Levenberg-Marquardt algorithm which "nlinfit" uses for non-robust fits.

Based on the nonlinear fitting method presented above, the three coefficients c_1, c_2, and c_3 are calculated to be -2.4870e3, -0.0608e3 and -0.2420e3 m^2 respectively.

Fig. 8 below exhibits the comparison between the fitting curves and experiment data points:

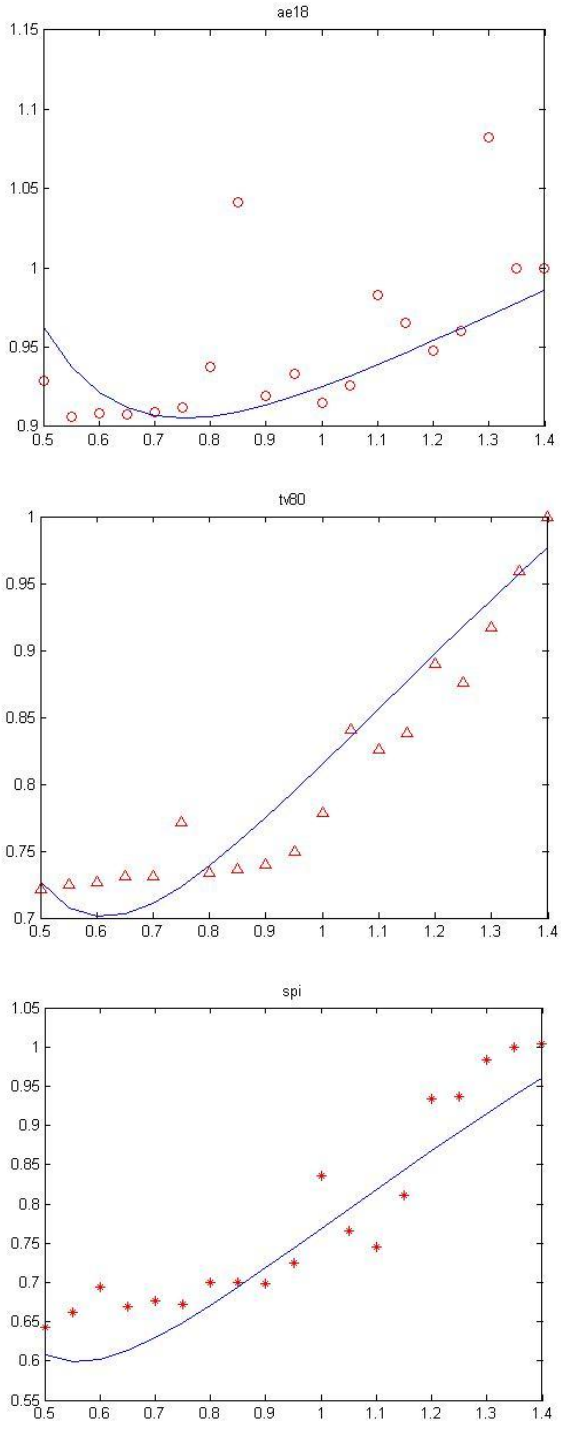


Fig. 8 Comparison between the fitting model and experiment results.

After getting the three coefficients from the three designs, we use the other two designs (*usb* and *mips*) to verify the accuracy of the model, which is shown in Fig. 9 as following:

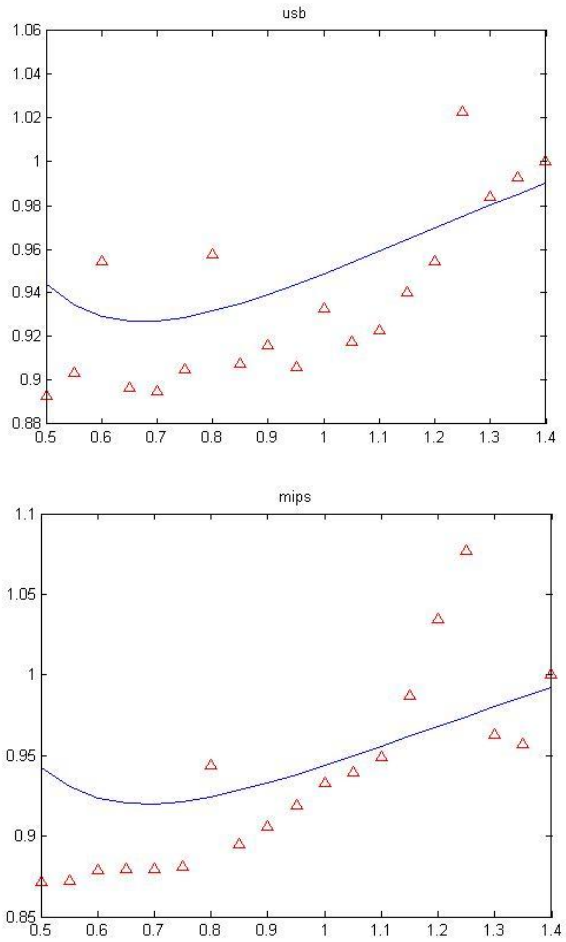


Fig. 9 Testing of the fitted model.

From the above figures, it can be concluded that the fitted model based on sizing and buffering theory has a good prediction for the trend of normalized total chip area change with gate delay scaling upon information at one corner. This model combines physical theory with real experiment results, which provides a robust and time-efficient reference for guardband reduction during design stage.

V. Conclusions

In this paper, a mathematical model is established to predict the chip area change with the scaling of the gate delay in the standard-cell library. From the modeling perspective of view, we start from the most classical Elmore delay theory, optimize the gate size, buffer size and number step by step and finally arrive at a model relating chip area to the gate delay scaling factor. However, this model is not totally isolated from real design characteristics. Design-dependent parameters from the SPR implementation flow are introduced to consider the variation among different designs, which makes it more comprehensive than previous works.

References

[1] Bowman, K., et al. 2002. "Impact of die-to-die and within-die parameter fluctuation on the maximum clock frequency distribution for

gigascale integration," *IEEE Journal of Solid-State Circuits*, vol. 37, pp.183-190, Feb 2002.
[2] S.R. Nassif, "Modeling and forecasting of manufacturing variations," *International Workshop on Statistical Metrology*, pp.2-10, 2000.
[3] F. Worm, P. Thiran, G. de Micheli, P. Ienne, "Self-calibrating networks-on-chip," *IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 3, pp. 2361-2364, 23-26, May 2005.
[4] M. C. Smayling et al, "Low k1 Logic Design using Gridded Design Rules," *SPIE*, 2008.
[5] P. Gupta, A. B. Kahng, Y. Kim and D. Sylvester, "Self-Compensating Design for Reduction of Timing and Leakage Sensitivity to Systematic Pattern Dependent Variation," *IEEE Transactions on CAD*, 2007.
[6] K. Jeong, A. B. Kahng and K. Samadi, "Impact of Guardband Reduction On Design Outcomes: A Quantitative Approach," *IEEE Transaction on Semiconductor Manufacturing*, vol. 22, no. 4, Nov. 2009.
[7] R. S. Ghaida, "Area Benefit of Guardband-Reduction."
[8] W. C. Elmore, "The Transient Response of Damped Linear Networks With Particular Regard to Wideband Amplifiers," *Journal of Applied Physics*, vol 19, 55,
[9] H. B. Bakoglu and J. D. Meindl "Optimal interconnection circuits for VLSI," *IEEE Trans. Electron Devices*, vol. ED-32, no. 5, May 1985.
[10] J. Chern, J. Huang, L. Arledge, P. Li, and P. Yang, "Multilevel metal capacitance models for CAD design synthesis systems," *IEEE Electron Device Letters*, vol. 13, no. 1, January 1992.
[11] International Technology Roadmap for Semiconductors [Online]. Available: http://public.itrs.net/
[12] OPENCORES.ORG [Online]. Available: http://www.opencores.org/
[13] Seber, G. A. F., and C. J. Wild. *Nonlinear Regression. Hoboken*, NJ: Wiley-Interscience, 2003.