Pulsed-Latch Aware Placement for Timing-Integrity Optimization *

Yi-Lin Chuang[1], Sangmin Kim[2], Youngsoo Shin[2], and Yao-Wen Chang[1,3]
[1]Graduate Institute of Electronics Engineering, National Taiwan University, Taiwan
[2]Department of Electrical Engineering, KAIST, Korea
[3]Department of Electrical Engineering, National Taiwan University, Taiwan
nicky@eda.ee.ntu.edu.tw; smkim@dtlab.kaist.ac.kr; youngsoo@ee.kaist.ac.kr; ywchang@cc.ee.ntu.edu.tw

ABSTRACT

Utilizing pulsed latches in a circuit is one emerging solution to timing improvements. Pulsed latches, driven by a brief clock signal generated from pulse generators, possess superior design parameters over flip-flops. If pulse generators and pulsed latches are not placed properly, however, pulse-width degradations at pulsed latches and thus timing violations might occur. In this paper, we *introduce* the pulsed-latch aware placement problem for timing integrity and present a unified placement framework to tackle this problem. Our new placer has the following distinguished features: (1) a multi-level pulsed-latch aware analytical placement framework to effectively prevent the potential pulse-width distortion problem, (2) a physical-information aware latch grouping algorithm to identify each desired group of a pulse generator and pulsed latches, and (3) a new optimization gradient for global placement to consider the impact of load capacitance of generators. Experimental results show that our placement flow can effectively consider pulse-width integrity and thus achieve much smaller total/worst negative slacks with marginal wirelength overheads, compared to a leading commercial and an academic placement flows.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids—*Placement and Routing*; J.6 [**Computer-Aided Engineering**]: Computer-Aided Design

General Terms

Algorithms, Performance

Keywords

Physical Design, Placement, Pulsed latch

*This work was partially supported by ITRI, SpringSoft, Synopsys, TSMC, and NSC of Taiwan under Grant No's. NSC 98-2622-E-002-005-A2, NSC 98-2221-E-002-119-MY3, NSC 97-2221-E-002-237-MY3, NSC 96-2628-E-002-249-MY3, and NSC 96-2628-E-002-248-MY3.

DAC'10, June 13-18, 2010, Anaheim, California, USA

1. INTRODUCTION

Flip-flops are commonly used as sequential components to store data in datapath circuits. Because flip-flop synchronization with the clock edge matches with static timing analysis (STA), flip-flop based sequential circuit has the advantage of easier timing verification. As a result, edge-triggered sequential circuits, which consist of combinational blocks that lie between edge-triggered D flip-flops, are the most common form of sequential circuits in ASICs. However, since a conventional flop-flop is composed of two latches (master and slave) triggered by a clock signal; flip-flops have significant overheads than latches in terms of delay, clock load, and area. In the simulation report under the $45nm$ process technology from our experiments, a flip-flop requires 1.25X setup time and 1.55X area than a latch. Specifically, the delay of a flip-flop is one of many reasons why ASICs are slower than custom designs under the same technology by a factor of six or more [4].

Level-sensitive latch designs are relatively simple and consume much less power than that of flip-flops. However, it is harder to perform timing verification on latch designs due to their data transparent nature. On the other hand, because of the transparency, latches allow a combinational block to have delay larger than the clock period, commonly called time borrowing or cycle stealing; clock skew can be tolerated if the transparency window, shifted by the skew, can still capture the data [10]. For this reason, they are widely used in high-performance microprocessor designs.

Pulsed latches are latches driven by a pulse clock waveform. A latch is synchronized with the clock similarly to an edge-triggered flip-flop because the rising and falling edges of the pulse clock are almost identical in terms of timing. In addition to its better design parameters (e.g., area, delay, etc.), therefore, a pulsed latch itself also offers easier timing verification/optimization just like a flip-flop. In practical applications, pulsed latches have been mostly adapted in high-performance microprocessor designs. In recent research, by selecting appropriate pulse widths for each latch, we can effectively improve circuit timing [10]. Therefore, pulsed-latch based designs have become a promising solution for modern circuit designs.

Triggered by a clock signal, pulsed latches require a pulse generator to generate a clock waveform. Different types of generators generate pulses of different widths. Figure 1 shows our pulsed-latch design scheme and a pulse generator structure. After receiving the clock waveform from the clock source, a pulse generator with the structure illustrated in Figure 1(b) generates a brief clock signal to each connected latch. A similar pulse-generator structure is also applied in [10, 16], and its pulse width is controlled by the delay cell. In this methodology, latches with the same pulse width would share the same generator. In this work, we call the latches and their corresponding generator as pulse-generator-latches

group (PGL group, for short).

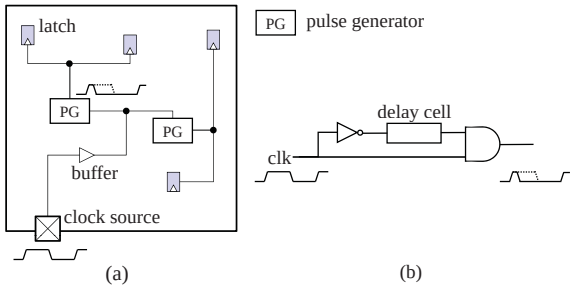

Figure 1: (a) Pulsed-latch circuit and (b) pulse generator structure, which consists of an inverter, a delay cell, and an AND gate.

However, since a pulse generator itself is a combinational circuit (see Figure 1). The delay and driving capability of a pulse generator would also be affected by the (output) load capacitance. If a pulse generator and latches are not placed properly, the wirelength among them might become too long and thus make the generated pulse width distorted, which might cause serious timing violations. As the simulation shown in Figure 2, as load capacitance keeps increasing, the pulse width shrinks dramatically. Therefore, after circuit placement, the pulse-width degradation might cause timing constraint violations, e.g., setup/hold time violations. Even if we apply the same width to every latch, the load capacitance of each PGL group also needs to be considered. To further explore the pulse-generator characteristics, we conducted a set of experiments to explore the corresponding effects in Section 2.2.

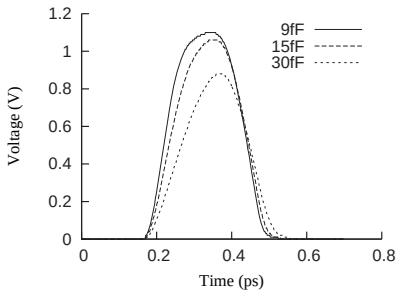

Figure 2: Pulse-width simulation. The pulse width might be distorted as load capacitance increases.

Due to the pulse-width degradation property shown in Figure 2, the pulsed-latch placement is essentially different from traditional placement problems. In traditional placement problems, the main objective is to minimize the total wirelength; however, minimizing the total wirelength does not mean that the placement result will have smaller load capacitance at a generator. Since traditional placement algorithms do not honor the load capacitance of a generator, this could cause longer wirelength between the generator and latches and thus degrade the pulse width. One simple solution is to increase the net weight between a pulse generator and its corresponding latches, like traditional clock-gating aware or power aware placement. However, it is very difficult to determine the magnitude of increased weight for each PGL group; especially, each PGL group has different pulse-width profiles. Applying a large weight may maintain the pulse width during placement; however, it could cause unstability problems on signal wirelength or density. Therefore, the traditional placement techniques cannot directly apply to the pulsed-latch placement problem well.

In this paper, we *introduce* the pulsed-latch aware placement problem for timing integrity and present a unified analytical placement framework to tackle this problem. We summarize the key features of our approach as follows.

- Our proposed multilevel placement framework utilizes the relations between pulse generators and pulsed latches to derive a placement result with the load capacitance consideration of each PGL group. Therefore, our proposed placement framework can prevent from serious timing violations that might occur to the traditional sequential-circuit placement flows.
- Unlike the previous work [10] that determines PGL groups only by logic specifications from the scheduled netlist, we develop a better PGL-group identification algorithm which can effectively reduce the wirelength overhead by the physical relations among latches.
- With the proposed optimization gradient, the barrier forces, integrated with the analytical placer, we can effectively consider the interactions among PGL groups and other logic cells during placement and thus optimize each element *simultaneously and globally*, leading to better tradeoff between total wirelength and density.
- Experimental results show that the pulsed-latch based designs placed by our proposed methods result in superior pulse-width integrity. This allows us to obtain significant improvements on timing with comparable wirelength, compared to traditional sequential circuit placement flows.

2. PRELIMINARIES

2.1 Analytical Placement Framework

Since we adopt NTUplace3 [6] to demonstrate our placement flow, we shall first review the NTUplace3 algorithm. For analytical placement, we can model a circuit using a hypergraph $H = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ denote cell/macro blocks, and hyperedges $E = \{e_1, e_2, \ldots, e_m\}$ represent nets. Let x_i and y_i be the respective x and y coordinates of the center of block i. The analytical placement optimizes wirelength under the cell density constraint which is modelled with uniform non-overlapping bins. Consequently, the global placement problem can be viewed as a constrained minimization problem as follows:

$$\begin{aligned} \min \quad & W(\mathbf{x}, \mathbf{y}) \\ s.t. \quad & D_b(\mathbf{x}, \mathbf{y}) \leq M_b, \qquad \text{for each bin b,} \end{aligned} \tag{1}$$

where $W(\mathbf{x}, \mathbf{y})$ is the wirelength function, $D_b(\mathbf{x}, \mathbf{y})$ is the potential function which represents the area function of movable blocks in bin b, and M_b is the maximum potential which denotes the total area of movable blocks allowed in the bin. The wirelength $W(\mathbf{x}, \mathbf{y})$ is defined as the total half-perimeter wirelength (HPWL). Since function $W(\mathbf{x}, \mathbf{y})$ is not smooth, it is hard to optimize it directly. In our placement framework, we use the log-sum-exp function [12] to smooth the wirelength function, and then apply gradient search to find the optimal global placement. The potential function can be computed by the multiplication of the horizontal overlap and the vertical overlap. We use the bell-shaped function [9] to obtain the smoothed potential function $\hat{D}_b(\mathbf{x}, \mathbf{y})$. Nevertheless, the bell-shaped function might generate high "mountains" and deep "valleys" in the chip density map, which makes the gradient search difficult and inefficient in finding a desired solution. As a result, we use the Gaussian function to further smooth the base potential [6].

To solve Equation (1), we utilize the quadratic-penalty method to convert the original formulation into a sequence of unconstrained minimization problem of the following form:

$$\min \quad W(\mathbf{x},\mathbf{y}) + \lambda \sum_b (\hat{D}_b(\mathbf{x},\mathbf{y}) - M_b)^2 \qquad (2)$$

by introducing the multiplier λ. As the value of λ increases, the cells spread over the chip gradually until the density constraint for each bin is near-satisfied.

2.2 Pulse-Generator Characteristics

As Section 1 introduces, we consider the load capacitance of generators to reduce the possibility of pulse-width degradation. To understand the impact of capacitance and resultant pulse width, we use HSPICE [7] to find out the relations between these two factors. For each type of generators, we derive the corresponding pulse width under different load capacitances. The relations are summarized in Figure 3. Figure 3 shows the capacitance-width relations derived from five pulse generators adopted in this work. From the figure, we can see that as load capacitance increases, the resultant width keeps decreasing. If the pulse width is too small, the latch allocated by this width may not function in a proper time, which might increase the clock period (slower design) and cause timing violations.

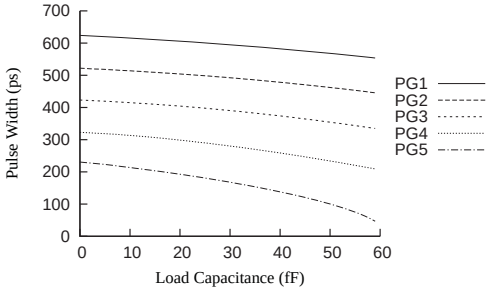

Figure 3: Relations between load capacitance and pulse width under different types of generators.

2.3 Problem Formulation

In this paper, we would like to reduce potential timing violations incurred by pulse-width degradation of generators. Given a user specified maximum pulse-width bound of each type of generators, we compute the corresponding maximum tolerable load capacitance from the simulation shown in Figure 3. To consider the capacitance impact, we transform it into a wirelength constraint (the maximum accumulated wirelength constraint, or MAWC for short) in the placement stage by our proposed algorithm. Accompanied by a scheduled netlist, we can determine a set of PGL groups with logical consideration and physical correlations. We formulate our problem as follows:

Pulsed-Latch Aware Placement: Given a pulsed-latch-width scheduled netlist, the maximum tolerable load capacitance of each type of generators, determine pulse-generator-latches (PGL) groups and find a placement for blocks such that the total wirelength is minimized and specified maximum tolerable capacitance is also satisfied.

3. MULTILEVEL PULSED-LATCH-AWARE PLACEMENT FRAMEWORK

In this paper, we propose a pulsed-latch aware multilevel analytical placement framework. Figure 4 summarizes the flow. Before placement, we determine a set of PGL groups by the physical locations from an initial placement. After this process, we update the input netlist with generators connected to each set of latches. Due to increasing complexity of modern circuit designs, multilevel framework is usually applied on analytical placement to improve the scalability. In general, the generator and latches of each PGL group should be placed closer to each other due to the maximum tolerable capacitance constraint. We propose a PGL-macro-like clustering technique to provide close distribution of latches in each PGL group before solving the analytical placement formulation. Then in the finest level, with known block distributions, we apply PGL-group aware global placement techniques including PGL-group compression and the proposed barrier force. In this level, our proposed optimization scheme provides a correlation mechanism among the traditional wire force, the density spreading force, and the barrier force to derive a global placement result with the MAWC consideration. Finally, after the legalization and detailed placement, we can obtain a pulsed-latch aware placement result. In the following subsections, we will detail the proposed algorithms.

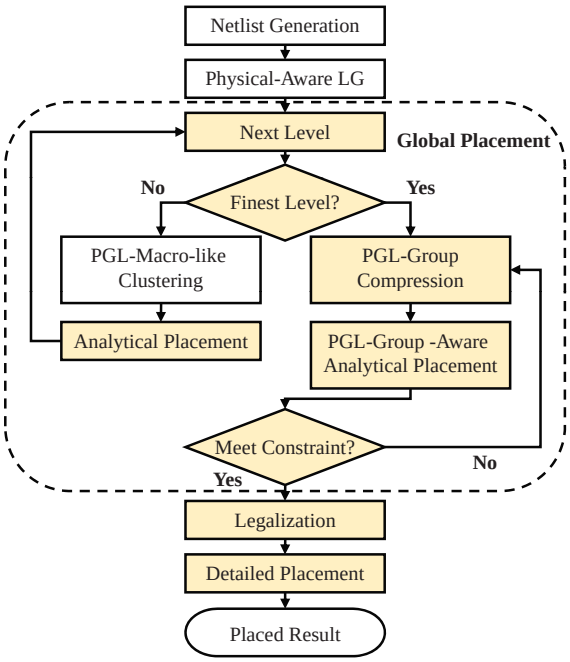

Figure 4: The proposed flow of our pulsed-latch aware multilevel analytical placement framework.

3.1 Physical-Aware Latch Grouping

Given a pulse-width scheduled netlist and an initial placement, we first determine each PGL group by the physical locations of latches. In [10], Lee *et al.* determined the PGL groups only by the logic relations and skew specifications, which may not be sufficient for pulsed-latch placement. To obtain a better trade-off between wirelength and the maximum capacitance constraint, we resort to a new grouping method with physical information in circuit placement. We summarize our physical aware latch grouping (PLG) algorithm in Figure 5.

There are two major issues for the PLG algorithm: PGL-group and MAWC specification. The PLG algorithm checks each type of pulse-width latches and identifies an appropriate group to include the latches, without exceeding the maximum capacitance constraint of the corresponding type of generators. Based on an initial placement, we transform the capacitance into the impact of the MAWC within each PGL group. After this transformation, satisfying the MAWC means considering the maximum capacitance constraint implicitly.

Each group size is determined by the accumulated capaci-

tance. For each grouping iteration, we consider the following inequality (line 6 in Figure 5).

$$C_w + C_l \leq C_t, \tag{3}$$

where C_w and C_l are the capacitance of wire and latches, respectively. C_t is the maximum tolerable capacitance of the generator with pulse width t. C_w can be obtained by the unit capacitance of metal wire and the corresponding wirelength, which can briefly expressed as $C_w = W_l(\mathbf{x}, \mathbf{y}) \times C_u$, where $W_l(\mathbf{x}, \mathbf{y})$ and C_u are the wirelength function of a latch group and unit capacitance of a metal wire, respectively. In this process, we use the half-perimeter wirelength (HPWL) to estimate the interconnect length among latches. We continue to group a latch if Equation (3) is satisfied.

Algorithm: **Physical-Aware Latch Grouping**
Input:
H: a pulse-width scheduled netlist by [10]
C_t: maximum tolerable capacitance of generator type t
l_i: a set of latches
Output:
H': updated netlist including pulse generators connected to the corresponding set of latches with the same pulse width

01. **for** each pulse-width type $t \in H$
02. un-group latches $U = \{l_i\}$
03. **while** $U \neq \phi$
04. find the starting point l in the most congested region
05. $G = \{l\}, U = U - \{l\}$
06. **while** $total\_capacitance(G) \leq C_t$
07. find a latch $l_i \in t$ with minimum $distance(l_i, l)$
08. $G = G + \{l_i\}$ and $U = U - \{l_i\}$
09. generate a group G
10. update latch density regions
11. generate a set of groups G_i with type t
12. generate all sets of groups G_i with all types t
13. insert corresponding generators
14. update the netlist H to H'
15. output H'

Figure 5: Our physical aware latch grouping (PLG) algorithm.

After all latches are grouped, we have the grouping result. As a result, the MAWC for each PGL group is the final value of $W_l(\mathbf{x}, \mathbf{y})$. Then we update the netlist by adding the generators and connect the generator to a set of designated latches with the same pulse width. Based on the proposed grouping algorithm, we can effectively reduce the wirelength and fully utilize the pulse generators, which provides a much better PGL-grouping result compared with the previous work.

3.2 PGL-Macro-Like Clustering

The multilevel framework adopts a two-stage technique of bottom-up coarsening followed by top-down uncoarsening. During the coarsening stage, the blocks are clustered level by level to reduce the number of movable blocks. The clustering process continues until the number of blocks is reduced below a given threshold. After clustering, the analytical placement problem is solved at each level of the uncoarsening stage. However, as we mentioned in Section 2.3, we intend to derive a placement satisfying the MAWC. If the multilevel framework is not aware of this requirement before solving the analytical formulation, after declustering, the blocks of a PGL group may spread all over the placement region, which makes the analytical placer difficult to meet the MAWC.

Moreover, since in the uncoarsening stage, each cluster contains multiple movable blocks, and the exact placement within a cluster remains unknown, it is relatively difficult to optimize PGL-group locations within each cluster directly. Therefore, in the upper level of uncoarsening, we cluster the blocks of each PGL group to a single macro and update the corresponding nets connecting to the original latches or generators. Then in the multilevel framework, each PGL group can be regarded as a little macro in the placement region. As the analytical placer gradually optimizes the clustered blocks level by level, the placer implicitly provides a better location for each of PGL groups.

In addition, to consider the cluster size effectively, we adopt the best-choice clustering [1] to obtain the clusters during each level. Therefore, in the finest level after each PGL macro is declustered, the latches belonging to the same PGL group would have relatively closer distances, which provides a good initial input for analytical placer.

3.3 PGL-Group-Aware Placement

3.3.1 *Barrier Method for Pulsed-Latch-Aware Global Placement*

When the uncoarsening stage reaches the finest level, the optimization process of analytical placement framework is directly applied on blocks of the circuit instead of clusters. Therefore, with known latch positions, we can optimize the MAWC in global placement. To make each PGL group satisfy the MAWC and maintain wirelength quality, in the finest level, we propose a new barrier force to provide the third gradient for the optimization.

Before we introduce our proposed barrier force, we shall briefly introduce how the barrier method works. Consider the following inequality constrained minimization problem:

$$\begin{aligned} \min \quad & c^T x \\ s.t. \quad & a_i^T x \leq b_i, \qquad i = 1, \cdots, m \end{aligned} \tag{4}$$

To solve this constrained optimization problem, one popular approach is the barrier method. By defining the logarithmic barrier, we solve a sequence of unconstrained minimization problems as follows:

$$\min c^T x + \sum_{i=1}^{m} -\log(-(a_i^T x - b_i)), \tag{5}$$

where the logarithmic barrier $\phi(x)$ is defined as [2]

$$\phi(x) = \begin{cases} \sum_{i=1}^{m} -\log(-(a_i^T x - b_i)), & Ax < b, \\ +\infty, & \text{otherwise.} \end{cases} \tag{6}$$

With the above definition, we can find that $\phi(x)$ will approach to ∞ as x approaches to the boundary of $Q = \{x | Ax < b\}$, which can effectively confine the solution to lie inside the constraint. Since we want to ensure that the wirelength of a PGL group will not exceed the corresponding MAWC, this property is very useful for this problem. In addition, theoretically, according to the log-barrier method, if we start at a feasible initial solution $\mathbf{x_0}$ ($\mathbf{x_0}$ satisfies the constraint), we can always stay strictly in the interior of the constraint space [2]. This feasiblity property is desirable. Therefore, by adopting the log-barrier method, we can have the following two theorems.

THEOREM 1. *(Feasibility of the Log-Barrier Method) By the log-barrier method, starting at a feasible $\mathbf{x_0}$ in the interior of Q, we can always stay strictly in the interior of Q.*

THEOREM 2. *(Smoothness and Convexity Properties of the Barrier Function) According to the 1st- and 2nd-order derivatives, the barrier function $\phi(x)$ is a smooth function on Q, and the function value y is a convex function for all $y \in \mathbf{R}^n$.*

The log-barrier method and its application is extensively discussed in the nonlinear optimization field [2]; due to the page limit, therefore, we omit the proofs for these theorems.

To effectively consider the MAWC of each PGL group, we apply the log-barrier method to optimize the pulsed-latch aware placement. For the MAWC of each PGL group, the corresponding Q can be defined as $W_g(\mathbf{x},\mathbf{y}) \leq MAWC_g$, where $W_g(\mathbf{x},\mathbf{y})$ is the wirelength function of a PGL group g, and $MAWC_g$ is the corresponding MAWC value of the group. Obviously, according to the transformation shown in Equation (5), we can transform our pulsed-latch aware placement problem into a unconstrained formulation as

$$\min \quad W(\mathbf{x},\mathbf{y}) + \lambda_d \sum_b (\hat{D}_b(\mathbf{x},\mathbf{y}) - M_b)^2 \\ -\lambda_r \sum_g log(MAWC_g - W_g(\mathbf{x},\mathbf{y})), \qquad (7)$$

where λ_d and λ_r are density and barrier weights, respectively. As λ_r gradually decreases, it can eventually converge to a desired placement result with good wirelength, density overflow, and also satisfy the MAWC. For $W_g(\mathbf{x},\mathbf{y})$, we use the log-sum-exp wire model. To solve this unconstrained form, we apply the conjugate gradient method. By Theorem 2, the log barrier in Equation (7) is also a smooth and differentiable function; consequently, we directly compute its gradient in the analytical placement framework.

To apply the log-barrier method, mathematically we need to start at a feasible initial solution, implying that the wirelength of each PGL group should not exceed its MAWC. We apply the PGL-group compression to attain this requirement. We first derive the corresponding vectors from a generator to a latch, and directly move the latch close to the generator according to the derived vector. After the mathematical formula derivation, basically we perform the compression by the following equation:

$$X' = diag(1-\alpha_i)X + x_g, \qquad (8)$$

where X is the original x-coordinate matrix of latches, $diag$ is the diagonal matrix with diagonal entries $(1-\alpha_i)$, and X' is the resultant x-coordinate. α_i is a user specified parameter to control how much to compress the latch i along the vector direction, and x_g is the x-coordinate of generator. A similar equation is also applicable to the y-direction.

By applying the barrier force, as shown in Figure 6, each PGL group will receive three different forces simultaneously. Imagine that we have a MAWC feasible region (of course, it is hard to find this region exactly in practice), the latch locations are determined by the summarization of wire forces, spreading forces, and barrier forces. Basically, the barrier gradient provides a confined force trying to keep the current MAWC region feasible; the latch locations will be guided by the wire and spreading forces. By the proposed optimization scheme, the locations of latches and the generator in a PGL group are determined while considering the impacts of other logic cells simultaneously, which can have higher flexibility

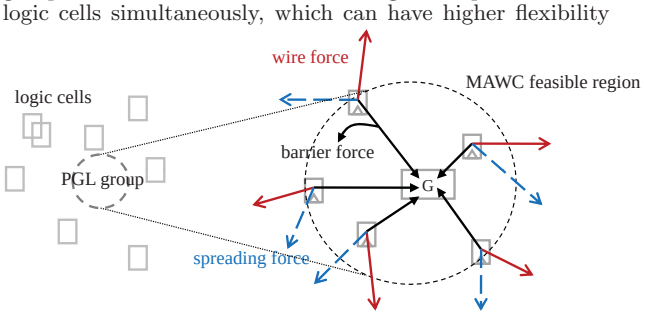

Figure 6: Force concept of barrier forces among other two traditional forces.

4. EXPERIMENTAL RESULTS

We conducted experiments to verify the effectiveness of our proposed flow and placement algorithm. Our placement algorithm was integrated into NTUplace3 [6], a leading academic placer which is available to the public. It should be noted that the proposed algorithm is flexible and can also easily be integrated into other placers based on the similar framework. We mainly compare our proposed pulsed-latch aware placement flow with two placers, Cadence SOC Encounter [3] (a leading commercial tool) and DCTB [17] (a state-of-the-art academic clock-tree aware placer with its name taken from the paper title). Most experiments were performed on the same machine with four AMD Opteron CPUs and 16 GB memory. We tested on the six OpenCores [13] circuits in the IWLS2005 benchmark suite [8]. The circuit statistics are given in Table 1. The columns "#Latches", "#Gates", "#Nets", "P_{pwcs}", and "Util (%)" list the number of latches, the number of gates, the number of nets, the improved clock period by running the binary from [10], and the utilization rate of the circuit, respectively. Our comparisons for the timing results are based on the clock period P_{pwcs}.

We synthesized the circuits and added pulse generators using the Nangate $45nm$ Open Cell Library [11]. A set of five pulse generators were constructed to provide pulse widths. The widths were 230ps, 322ps, 423ps, 522ps, and 623ps. Each generator has the similar structure/layout with [10, 16]. In this work, the pulse width is controlled by the delay cell.

Table 1: Statistics of the OpenCores circuits and clock periods after pulse-width scheduling by the method in [10].

Circuit	#Latches	#Gates	#Nets	$P_{pwcs}(ps)$	Util(%)
ac97_ctrl	2199	10205	10371	439	74.83
aes_core	530	15796	16055	1208	69.21
des_perf	8808	74011	74224	878	72.90
mem_ctrl	1045	11672	11782	1552	72.28
pci_bridge32	3134	20248	20743	1006	74.29
wb_conmax	770	30629	31721	924	70.72

In the experiment, we compared three different flows: (A) LC (latch-clustering) followed by SOC Encounter, (B) LC followed by DCTB, and (C) our proposed flow introduced in Section 3. The input netlists used in Flows A and B were generated by the LC method proposed in [10]. For Flow C, we used the PLG proposed in this work to determine the PGL groups. Then, based on the generated netlist (with generators), we performed circuit placement by Encounter with timing-driven mode, DCTB, and our pulsed-latch aware placer, respectively. To compare with the flow integrated with DCTB, we also implemented the DCTB algorithm into NTUplace3 for fair comparison.

After circuit placement, we used FLUTE [5] to estimate the clock Steiner wirelength and derived the wire capacitance based on the clock wirelength. By the simulation results shown in Figure 3, we computed the corresponding pulse width by interpolation, and applied the derived pulse widths to latches for timing verification. Given a clock period after logic scheduling, we explored the internal impact of pulse-width degradation after circuit placement. To quantify this metric, we proposed the "Pulse Width Degradation Ratio" (PWDR) to model the width degradation of each latch. Given the allocated pulse width $P_{i,opt}$ of latch i in a scheduled netlist (which does not consider any physical information and can thus be regarded as the optimal period), the PWDR is defined as

$$PWDR(P_{opt}, P_{placed}) = \frac{\sum_{i \in L} \left(\frac{P_{i,opt} - P_{i,placed}}{P_{i,opt}} \right)}{|N_L|}, \qquad (9)$$

where L is the set of latches, N_L is the total number of

latches, and $P_{i,placed}$ is the pulse width of latch i after placement. PWDR basically is the timing index of pulse widths. A smaller PWDR implies that the placement result can maintain the scheduled pulse widths more easily and can thus potentially keep the clock period smaller.

The experimental results for the three different flows are shown in Table 2. The columns "Vio.", "WNS", "TNS", "SW", "CW", and "TW" are the total number of timing path violations, the worst negative slack, the total negative slack, the signal wirelength, the clock wirelength, and the total wirelength, respectively. The WNS and TNS were computed from SIS [15] by setting the clock period to the original optimal clock period, and adding all the slacks of the timing paths with violations.

As Table 2 shows, compared with Flow A, our proposed flow achieves 19X, 13X, and 47X improvements on PWDR, WNS, and TNS, respectively. In addition, compared with Flow B, we can also achieve 18X, 13X, and 27X improvements on PWDR, WNS, and TNS. Moreover, our flow has the minimum violations compared to others. The reasons for these significant improvements are two-fold. First, the LC algorithm does not consider physical relations among latches; therefore, the quality of its placement result is limited. Second, the placement algorithms of SOC Encounter and DCTB might not capture the characteristics of pulse generators, leading to inferior performance for pulsed latch placement.

With our barrier method, we can reduce the clock wirelength by almost 5X, compared with other flows. With the effective interaction among wire forces, in addition, our respective signal wirelength overheads over Flows A and B are only 2% and 1%. Due to the space limit, we do not list the area and runtime in the table and instead summarize the comparison as follows. Since our PLG algorithm considers load capacitance based on the physical correlation among latches, we need more generators than the LC method, ranging from 28 to 397 generators which sum up to a 3.36% area overhead than Flows A and B on average. In addition, according to the reported runtimes, Flows A and B incur respective 55% and 61% longer runtimes than our flow. Note that the commercial tool (Flow A) is installed in a 64-bit Intel Xeon machine with 8 GB memory, so the reported runtime is just for readers' reference. For Flow B, DCTB would impose additional attracting forces to the pairs of topological clock tree nodes during global placement, which could cause solution oscillation during global placement. Therefore, it might incur redundant loops and thus degrade the solution quality and efficiency.

In the second experiment, we compared individual placement algorithms. The input netlist consists of two different PGL-group configurations, derived from LC and PLG. We applied Encounter, DCTB, and our pulsed-latch aware placer to place the circuits. Comparing the results listed in Tables 2, our PLG algorithm can find better PGL groups than the previous work. Due to the page limit, we just summarize the corresponding results. For the LC scheme, our placement algorithm can achieve respectively 2X, 2–3X, and 3–5X improvements over PWDR, WNS, and TNS with much fewer violations. For the PLG scheme, in addition, our placer can outperform the other approaches (ranging from 1.3X to 6X improvements on different metrics); the results show that our placement algorithm can also improve the solution quality, besides the proposed latch-clustering algorithm.

Table 2: Comparisons among the three placement flows. Note that the values in row "Norm." are all normalized to those of Flow C.

	Flow A: LC + Encounter						
	Vio.	PWDR	WNS	TNS	Wirelength ($\times e^7$)		
Circuit		(%)	(ps)	(ps)	SW	CW	TW
ac97_ctrl	18	5.54	16.80	152.35	12.94	2.36	15.30
aes_core	133	27.41	91.99	1203.10	32.53	1.56	34.09
des_perf	56	15.98	50.12	586.70	132.28	15.53	147.81
mem_ctrl	300	16.11	71.98	1034.41	18.08	2.14	20.22
pci_bridge32	289	18.71	122.93	3776.37	31.72	7.68	39.40
wb_conmax	743	21.17	202.10	8313.09	109.14	3.37	112.51
Norm.	-	**19.28**	**12.58**	**46.84**	**0.98**	**5.63**	**1.07**
	Flow B: LC + DCTB						
	Vio.	PWDR	WNS	TNS	Wirelength ($\times e^7$)		
Circuit		(%)	(ps)	(ps)	SW	CW	TW
ac97_ctrl	35	5.56	12.51	170.13	12.79	2.29	15.08
aes_core	81	24.04	56.47	539.57	32.61	1.40	34.02
des_perf	72	15.18	138.85	708.05	128.37	15.87	144.24
mem_ctrl	155	15.35	45.97	318.52	18.80	2.06	20.86
pci_bridge32	147	16.40	166.85	1641.01	29.41	7.06	36.47
wb_conmax	722	18.81	190.84	8000.48	124.37	1.75	126.12
Norm.	-	**17.62**	**13.33**	**26.98**	**0.99**	**4.51**	**1.07**
	Flow C: Our flow (PLG + our pulsed-latch aware placer)						
	Vio.	PWDR	WNS	TNS	Wirelength ($\times e^7$)		
Circuit		(%)	(ps)	(ps)	SW	CW	TW
ac97_ctrl	10	0.86	1.95	8.04	13.56	0.92	14.48
aes_core	7	1.10	5.12	13.20	31.91	0.23	32.15
des_perf	13	0.69	5.57	17.45	132.92	4.94	137.86
mem_ctrl	5	0.80	3.62	10.01	19.01	0.50	19.51
pci_bridge32	74	0.79	8.41	382.89	29.35	1.33	30.68
wb_conmax	29	1.23	37.63	345.10	122.83	0.30	123.13
Norm.	-	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**

5. CONCLUSION

We have introduced the pulsed-latch aware placement problem for timing integrity. To tackle this problem, we have proposed a better latch-group determination algorithm considering physical information and have extended the analytical placement framework by a provably-good optimization force to reduce the potential load capacitance of pulse generators. Experimental results have shown the effectiveness and efficiency of our approach for pulsed-latch placement.

6. REFERENCES

[1] C. Alpert, A. Kahng, G.-J. Nam, S. Reda, and P. Villarrubia. A semi-persistent clustering technique for VLSI circuit placement. In *Proc. of ISPD*, 2005.

[2] S. Boyd and L. Vandenberghe. Convex Optimation. Cambridge University Press, 2004.

[3] *Cadence Design Systems.* http://www.cadence.com.

[4] D. Chinnery and K. Keutzer. Closing the Gap between ASIC & Custom. Kluwer Academic Publishers, 2002.

[5] C. Chu. FLUTE: fast lookup table based wirelength estimation technique. In *Proc. of ICCAD*, 2004.

[6] T.-C. Chen, Z.-W. Jiang, T.-C. Hsu, H.-C. Chen, and Y.-W. Chang. NTUplace3: An analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints. In *IEEE TCAD*, Vol. 27, No. 7, pp. 1228–1240, July 2008.

[7] HSPICE. http://www.synopsys.com/community/interoperability/pages/hspice.aspx.

[8] *IWLS 2005 Benchmarks.* http://iwls.org/iwls2005/benchmarks.html.

[9] A. B. Kahng and Q. Wang. Implementation and extensibility of an analytic placer. *IEEE TCAD*, 2005.

[10] H. Lee, S. Paik, and Y. Shin. Pulse width allocation with clock skew scheduling for optimizing pulsed latch-based sequential circuits In *Proc. of ICCAD*, 2008.

[11] Nangate 45nm Open Cell Library. http://www.nangate.com/.

[12] W. C. Naylor, R. Donelly, and L. Sha. Non-linear optimization system and method for wire length and delay optimization for an automatic electric circuit placer. U.S. Patent 6 301 693, 2001.

[13] *OpenCores.* http://www.opencores.org.

[14] M. Pan and C. Chu. A step to integrate global routing into placement. In *Proc. of ICCAD*, 2006.

[15] E. Sentovich, K. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. Stephan, R. Brayton, and A. Sangiovanni-Vincentelli. SIS: a system for sequential circuit synthesis. Tech. Rep., 1992.

[16] S. Shibatani and A. H.C. Li. Pulse-latch approach reduces dynamic power. *EETimes Online*, 2006.

[17] Y. Wang, Q. Zhou, X. Hong, and Y. Cai. Clock-tree aware placement based on dynamic clock-tree building. In *Proc. of ISCAS*, 2007.

[18] J.-S. Yim, S.-O. Bae, and C.-M. Kyung. A floorplan-based planning methodlogy for power and clock distribution in ASICs. In *Proc. of DAC*, 1999.