Post-Layout Logic Optimization of Domino Circuits

Aiqun Cao and Cheng-Kok Koh
School of Electrical and Computer Engineering
Purdue University, West Lafayette, IN 47907–1285
{caoa,chengkok}@ecn.purdue.edu

ABSTRACT

Logic duplication, a commonly used synthesis technique to remove trapped inverters in reconvergent paths of Domino circuits, incurs high area and power penalties. In this paper, we propose a synthesis scheme to reduce the duplication cost by allowing inverters in Domino logic under certain timing constraints. In order to guarantee the robustness of such Domino circuits, we perform the reduction of logic duplication at the physical level. Experimental results show significant reduction in duplication cost, which translates into significant improvements in area, power, and/or delay.

Categories and Subject Descriptors

B.6.3 [**Logic Design**]: Design Aids—*Automatic synthesis, Optimization*; B.7.2 [**Integrated Circuits**]: Design Aids—*Layout*

General Terms

Algorithms

Keywords

Domino logic, synthesis, optimization, layout

1. INTRODUCTION

Domino logic is the most popular dynamic logic. Owing to its high performance characteristic, Domino logic is widely used in designs that demand high speed. However, Domino logic circuits can implement only non-inverting logic; the synthesis of a Domino logic circuit typically involves the conversion to a unate representation from the original binate logic network. That is accomplished by pushing inverters (bubble pushing) [4] in the logic network to the primary inputs using DeMorgan's law. However, some inverters may be trapped at some intermediate fanout nets. Figure 1 shows two possible scenarios of the trapped inverters. In order to eliminate the trapped inverters, most approaches duplicate

the fan-in cones of the nets where the inverters are trapped. In the worst case, the size of the circuit doubles [7].
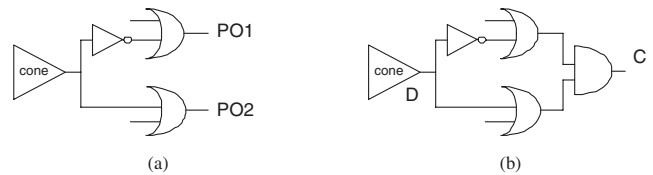
Figure 1: Inverters trapped in: (a) optimizable logic region, (b) logic reconvergent paths (non-optimizable region).
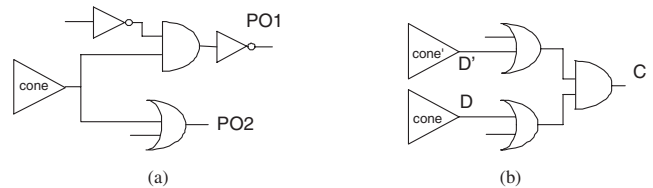
Figure 2: Trapped inverters eliminated by: (a) output phase assignment, (b) logic duplication.

There have been several studies on reducing the duplication cost. The trapped inverter in Figure 1(a) for example, can be eliminated without duplicating the fan-in cone by proper output phase assignment [6, 8]. As shown in Figure 2 (a), *PO1* has negative polarity (phase) whereas *PO2* has positive polarity (phase). The circuit in Figure 1(a) is called an optimizable logic region whose logic duplication cost can be minimized using phase assignment. The logic reconvergent paths in Figure 1(b) is categorized as a duplicated logic region; the fan-in cone is typically duplicated so as to remove the trapped inverter, as shown in Figure 2(b).

The output phase assignment problem was formulated as a vertex covering problem in [6], whereas [8] formulated the phase assignment problem as a 0-1 integer linear programming problem. The power consumption of Domino circuits could also be minimized by output phase assignment [3]. In general, the achievable area and power reduction is limited for the output phase assignment approaches due to the presence of logic reconvergent paths.

To reduce the duplication cost caused by logic reconvergent paths, [2] proposed a hybrid circuit composing of both Domino logic and static CMOS logic, with the latter trailing the former in the circuit. The inverters were allowed in the

logic network by synthesizing the transitive fan-outs of the inverters into static CMOS logic. An ATPG-based transformation was proposed to remove or relocate the inverters in order to minimize the static CMOS logic. The performance of the circuit however, is compromised. Moreover, as noted in [2], the static CMOS logic may cause functional faults in the presence of transparent latches as the trailing static CMOS logic may be non-monotonic.

In this paper, we observe that trapped inverters could be allowed in the reconvergent paths if a certain timing constraint, called the *Early-Late Delay Difference Bound*, is satisfied. (We will elaborate on this in Section 2.) Consequently, logic duplication cost can be reduced substantially. As accurate timing information is crucial to our proposed Domino circuit synthesis solution, we perform logic optimization of Domino circuits as a post-layout step, where accurate timing information can be obtained from post-layout simulation that takes into account accurate device and interconnect parasitics. The main contribution of this paper is the logic optimization and physical compaction of placed-and-routed Domino circuits with lower area and power overheads. We are also able to maintain or improve the timing performance of the Domino circuits.

2. EARLY-LATE DELAY DIFFERENCE BOUND

In Domino logic, correct operation is guaranteed (ignoring charge sharing and leakage) if every input of each gate can make only a single $0\to1$ transition during the evaluation phase; a $1\to0$ transition is not allowed as it would cause a false discharge that is unrecoverable. However, [5] proposed that $1\to0$ transitions may be tolerated in mixed Static-Domino circuit if the timing constraint to be presented below is satisfied. We observe that the trapped inverter can also be allowed in the reconvergent paths of pure Domino circuits if that timing constraint is satisfied.
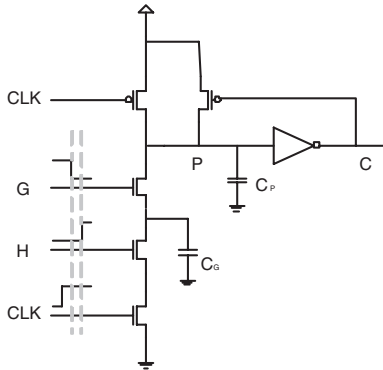
Figure 3: A correct operating Domino gate with $1\to0$ transition input with the timing constraint that signal G arrives earlier than signal H.

Consider the inputs to a Domino AND gate. During the evaluation phase, each input signal makes at most one transition, $0\to1$ or $1\to0$. We denote the earliest $0\to1$ transition among all inputs as $D^e_{0\to1}$ and the latest $1\to0$ transition as $D^l_{1\to0}$. To prevent the dynamic node from being falsely discharged, the pull down serial NMOS transistors in the Domino AND gate cannot be on simultaneously. That can

be achieved if the latest $1\to0$ transition always arrives earlier than the earliest $0\to1$ transition. In other words, the Domino AND gate would operate correctly if the following timing constraint is satisfied:

$$D^e_{0\to1} - D^l_{1\to0} \geq B, \qquad (1)$$

where $B > 0$ is the delay difference bound. We refer to B as the *Early-Late Delay Difference Bound (ELDDB)*. Note that the ELDDB should be large enough to accommodate delay variations due to crosstalk noise, process and voltage variations, etc. Figure 3 shows an example of a Domino two-input AND gate. The constraint is that G must always switch earlier than H if both of them switch. Obviously, this constraint implies that the fan-out of the trapped inverter cannot be an OR gate (unless we are considering clock-delayed Domino logic). If the ELDDB constraint is satisfied, the inverter may be allowed in the reconvergent paths, as Figure 4(a) shows.

In Figure 3, the transistor connected to G is put on top of that connected to H (closer to the precharged node P). It helps to alleviate the charge sharing problem in Domino circuit as follows: The internal capacitance C_G is charged to $V_{dd} - V_{th}$ (V_{th} being the threshold voltage) during the precharge phase as G is high. Therefore, there is no charge redistribution between C_P and C_G during the evaluation phase. Although the opposite transistor ordering (H on top of G) could have resulted in a speed-up, we do not consider that option in this paper as the robustness of the circuit is the main concern here.

3. LOGIC OPTIMIZATION

3.1 Definition

Consider the logic reconvergent paths in Figure 4(a). In this paper, we refer to logic reconvergent paths with one trapped inverter after bubble pushing as "*reconvergent loops*" (although there is really no feedback in terms of signal propagation). We define node D as the divergent node, C as the convergent node, and E, F, G, H, I as the intermediate nodes of the reconvergent loop in Figure 4(a). The fan-in cone of the reconvergent loop is defined as the divergent node and its fan-in cone. The two paths starting from the divergent node and ending at the convergent node are called two branches of the reconvergent loop. We also define node C, which is an AND gate and satisfies the ELDDB constraint, as a potential *candidate* that could allow $1\to0$ transition input so as to avoid logic duplication. (We shall discuss how the timing information can be obtained in Section 4.)

Consider a reconvergent loop whose convergent node is an AND gate. Moreover, the trapped inverter is one of the direct fan-ins of the convergent node. If G always switches earlier than H, as in Figure 4(a), the convergent node C is a potential candidate. If G always switches later than H (possible if W arrives late), we can relocate the trapped inverter from one branch to the other branch such that it becomes the other direct fan-in of C, as shown in Figure 4(c). Therefore, the convergent node is still a potential candidate. Note that the inverter can be relocated to locations in front of other AND gates within the reconvergent loop, for example, as shown in Figure 4(b). However, there is no guarantee that D' switches earlier than W'. On the contrary, the convergent node of a reconvergent loop with uneven branches is more likely to be a potential candidate. Moreover, different
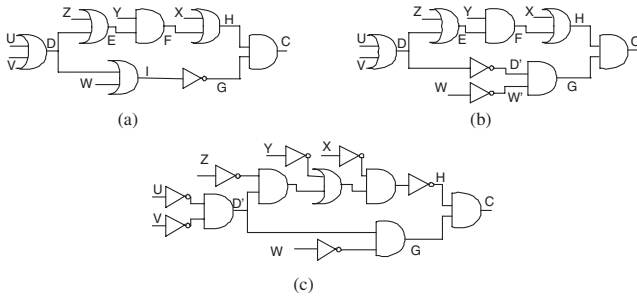
Figure 4: Trapped inverters allowed in the reconvergent loops.

potential candidates may conflict with each other due to incompatible phase assignments (as illustrated in Section 3.2). Consequently, we consider only convergent nodes as potential candidates in this paper.

3.2 Output phase assignment

Suppose we have the timing information to identify potential candidate gates. In order to reduce the duplication cost as much as possible, we require that the convergent nodes of all reconvergent loops be transformed into AND gates during the bubble pushing. However, there may be conflicts among those convergent nodes. Figure 5 shows an example of that. $C1$ and $C2$ are two convergent nodes of different reconvergent loops and $PO1$ and $PO2$ are primary outputs. From Figure 5(b) and (c), we can see that no matter how different phases are assigned to $PO1$ and $PO2$, only one of the convergent nodes can be transformed into an AND gate. $C1$ and $C2$ are incompatible with each other as they require different phase assignments to the primary outputs to be transformed into AND gates. Therefore, the problem is to search for the optimal output phase assignment such that the avoidance of logic duplication for potential candidates is maximized, i.e., the potential cost of logic duplication is minimized.
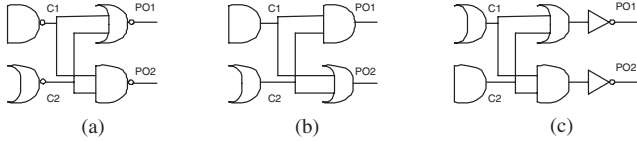
Figure 5: Conflicts among convergent nodes.

The problem can be formulated as follows: $G(V, E)$ is an incompatibility graph with a vertex v in V corresponding to a convergent node of a reconvergent loop. The duplication cost of the fan-in cone of the corresponding reconvergent loop is assigned as the weight to each vertex. An edge e in E connects two vertices in V if the two corresponding convergent nodes are incompatible with each other. Searching for the optimal output phase assignment is equivalent to searching for the least weight vertex cover set $Q \subset V$ that contains at least one endpoint of every edge in E. The fan-in cones of the reconvergent loops corresponding to vertices in Q still have to be duplicated. $V - Q$ contains the maximum weighted compatible candidates. The fan-in cones of the reconvergent loops corresponding to vertices in $V - Q$ can avoid duplication.

To solve the problem formulated above, the branch and bound algorithm based on BDD is applied similar to [6]. The incompatibility graph can be represented as a Boolean function:

$$F = \prod_{e_{ij} \in E} (v_i + v_j),$$

where $v_i \in Q$ if v_i is assigned the Boolean value TRUE. Therefore, an vertex cover Q of the incompatibility graph G corresponds to a SAT solution of the Boolean function F. The BDD can be built for the Boolean function F and an optimal SAT solution of least weight can be obtained by a branch-and-bound search on the BDD.

4. POST-LAYOUT OPTIMIZATION

At the logic level, no accurate timing information can be obtained to directly implement what we propose in the previous section. Therefore, a post-layout optimization is presented in this section. However, the physical layout step is also dependent on the synthesis, which provides the synthesized netlist for place-and-route tools. There is an interdependency between logic synthesis and physical layout: the final layout cannot be determined without knowing the circuit netlist; the netlist in turn cannot be determined as the candidate gates are not identified unless timing information of the layout is obtained.

We present a flow to solve this dilemma, as shown in Figure 6. At the logic level, as a preprocessing step, we use estimated delay values based on logic level information to predict the potential candidates. The synthesis scheme proposed in Section 3.2 is applied to produce maximum (weighted) potential candidates. Accurate timing information obtained after post-layout simulation will be used to verify and filter among the potential candidates to obtain real candidates later. After marking these potential candidates, the fan-in cones of their corresponding reconvergent loops are also duplicated. In other words, the physical design tools will handle the circuit with the fan-in cones of all reconvergent loops duplicated.

After mapping to Domino cells, standard-cell placement and routing are carried out to obtain an initial layout. Timing information are collected for the potential candidate cells after post-layout simulation. By setting the proper ELDDB value B, only those satisfying the ELDDB constraint remain as the candidate cells.

After that, we try two different approaches for the post-layout optimization as indicated by the two branches shown in Figure 6. One approach is to eliminate the duplication for all candidates at the same time, without compacting the layout. The other approach eliminate the duplication followed by a layout compaction iteratively. The two approaches are discussed in the next two subsections respectively.

4.1 Simultaneous elimination without compaction

Consider one candidate cell whose corresponding reconvergent loop has its original fan-in cone $CONE$ and duplicated fan-in cone $CONE'$ driving two branches respectively. $CONE'$ will be eliminated and $CONE$ will have to drive two branches to rejoin the reconvergent loop. In order to ensure that the ELDDB constraint is met, we perform only the elimination and rejoining and do not change the layout of the remaining cells. In other words, the empty space
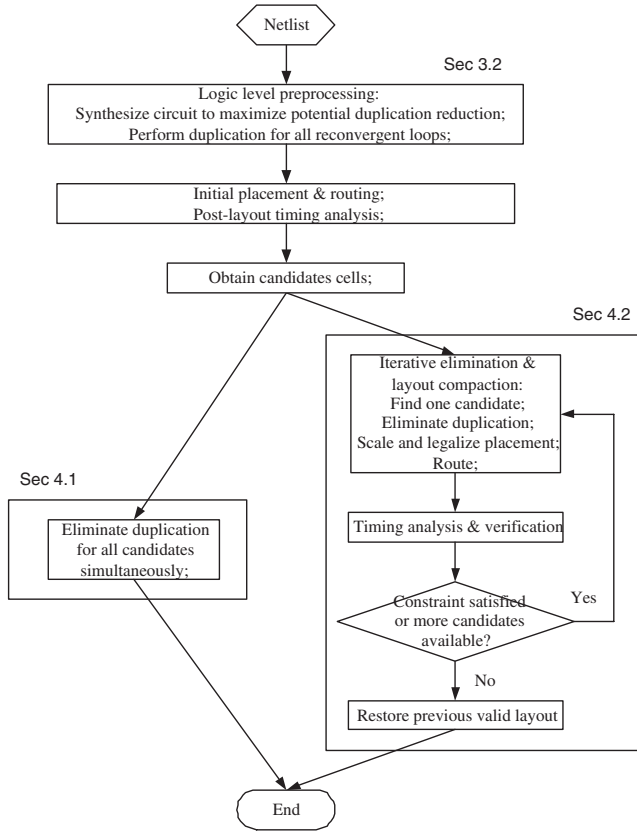
Figure 6: Post-layout optimization flow.

caused by elimination will be left as part of the layout. In this case, the timing behavior for the majority of the circuit does not change. Note that to rejoin the reconvergent loop, an input of the candidate cell is redirected to the internal dynamic node of the fan-in cell with proper transformation. This actually improves the robustness of the design as the $1 \rightarrow 0$ transition arrives even earlier by skipping the inverter in the Domino gate. In other words, it may be easier to meet the ELDDB constraint.

As the branch that was driven by $CONE'$ is now driven by $CONE$, the delay difference between $CONE$ and $CONE'$ may affect the ELDDB constraint. Therefore, either the ELDDB value B is set to take that into account or buffers are inserted to make up for the delay difference between $CONE$ and $CONE'$. Fortunately there is enough empty space for buffer insertion as we do not compact the layout after eliminating the duplicated fan-in cone. Moreover, the performance of the circuit is affected only if the delay of $CONE$ is larger than that of $CONE'$ and $CONE$ happens to be part of the critical path. Experimental results show that in general the delay difference between $CONE$ and $CONE'$ is very small. Moreover, to minimize the effect of crosstalk introduced by wires with $1 \rightarrow 0$ transition, shielding on such wires by power or ground lines is performed.

As the duplication elimination for each candidate does not affect other part of the layout, the elimination can be performed for all candidates simultaneously without violating the ELDDB constraints (see the left branch in the flow in Figure 6). Although the area of the layout does not reduce,

the power dissipation reduces significantly due to fewer number of gates in the circuit.

4.2 Iterative elimination and compaction

Leaving empty space in the layout is wasteful to the area. Moreover, it does not take advantage of shorter interconnects if layout compaction is performed; both performance and power dissipation may improve. However, to perform layout compaction is not trivial because the compaction may drastically change the timing behavior of the circuit.

In this subsection, we present an iterative flow (see the right branch of the flow in Figure 6) to perform both logic and physical level optimization. In each iteration, we eliminate the duplication for one particular candidate, followed by a layout compaction to get rid of the empty space caused by elimination. The order in which the candidates are chosen is based on the duplication cost (the area of the duplicated fan-in cone). The iterative algorithm starts with the candidate with the highest duplication cost.

The ELDDB constraints for the candidates considered in the current and all previous iterations have to be satisfied after compaction. A new post-layout simulation after each compaction is performed to verify that the constraints are still satisfied. If one of the constraints is not satisfied, the layout is restored and some other candidate is tried. For each successful compaction, we also perform an update on the remaining candidates based on the new timing information.

During the layout compaction, the routing information from the original layout is ignored as many cells will be relocated. Compaction is realized by scaling the placement and legalizing it (see Section 4.2.1 for details). A new routing solution is obtained after the placement compaction. The routing step is performed by a commercially available software, and the timing information is obtained from post-layout simulation using dynamic timing analysis tool. In the rest of this section, we present the only missing component of the iterative algorithm: placement scaling and legalization.

4.2.1 Placement scaling and legalization

Without loss of generality, assume that the left bottom corner of the layout is the origin and the position of a cell i is represented by its center (x_i, y_i). The layout area before the nth iteration is A_{n-1}. The candidate cell under consideration is C and the area of the duplicated fan-in cone that is eliminated is A_n^d. Therefore, the new layout area after compaction can be approximated as $A_n = A_{n-1} - A_n^d$ if all the empty space due to elimination can be removed completely.

A simple placement scaling is performed to relocate cell i to a new location (x_i', y_i'):

$$x_i' = x_i \times \sqrt{\frac{A_n}{A_{n-1}}}, y_i' = y_i \times \sqrt{\frac{A_n}{A_{n-1}}}.$$

The placement scaling can keep the topology of the circuit, i.e., the relative positions of and hopefully, the connections between cells. By keeping the circuit topology, we can potentially minimize the changes of delay differences between signals, although the scaling may reduce the absolute delays. Therefore, the ELDDB constraint is likely to remain valid.

After scaling the placement, however, most cells are not aligned to the row boundaries of the standard cell layout

any more. Furthermore, there are overlaps between cells. Therefore, a legalization step is required to relocate cells to align to the row boundaries of the standard cell layout and eliminate the overlaps between cells.

In the legalization step, the objective is to minimize the displacement for all cells. We place a higher priority on cells whose timing behaviors are crucial to meeting the ELDDB constraints. The fan-in cones of the candidates considered in the current and previous iterations are those critical cells. We assign a higher weight to these critical cells in the legalization step, which aims to optimize the following function:

$$DIS = \sum_i W_i \times ((x_i'' - x_i')^2 + (y_i'' - y_i')^2), \qquad (2)$$

where (x_i'', y_i'') is the new legal position of cell i, and W_i is the weight assigned to cell i.

We adopt the placement legalization approach proposed in [1]. In [1], a dynamic programming based row-by-row legalization is used to legalize a placement with overlap. The only difference between our legalizer and that in [1] is that we optimize weighted displacement for cells whereas [1] minimized total wire length without weight assignment.

After legalization, the layout topology does not change much and the total wire length reduces substantially. Therefore, routability is not an issue here, as verified by the experiments. In general, the critical wire length will reduce even with possible adverse effect due to legalization as the scaling of layout can more than compensate for that.

If the ELDDB constraint is violated to a certain extent, buffers can be inserted to increase slightly the delay of the slower path in order to satisfy the ELDDB constraint. The performance of the circuit may be marginally degraded if the path happens to be part of the critical path.

5. EXPERIMENTAL RESULTS

We have performed the experiments on ten ISCAS benchmark circuits on a Sun ultra80 workstation with two 450MHz CPUs and 1Gigabytes memory. Commercial placement and routing tools *QPLACE* and *WROUTE* are used for initial placement and routing. Row utilization is set to 90% for placement. The algorithms proposed in Sections 3 and 4 are implemented in C++ language. The flow in Figure 6 is automated by running a PERL script using LEF/DEF data formats for data transfer. The post-layout simulations are performed by using NanoSim and PathMill.

The ELDDB value B is conservatively set to $2ns$ for all circuits, which approximately equals three times the delay of a gate. The weight values in Eqn.(2) are assigned empirically. Experiments show that a weight of 1.3 is suitable for critical cells whereas other cells have a weight of 1.

A small library of Domino cells is constructed with $0.35\mu m$ technology for the experiments. The library consists of upto-three-input AND gates, up-to-six-input OR gates, Inverters, Buffers, and other filler cells. Each Domino cell in the library has a footer and a half-latch keeper. An output pin for the internal dynamic node of each Domino cell is added, so that each Domino cell has an additional output with $1 \rightarrow 0$ transition that may be connected to a candidate cell.

The experimental results are reported in Table 1. The first line 'ORI' lists the gate counts of the original binate networks of the benchmark circuits. Each original circuit is synthsized according to the flow shown in Figure 6. The subtable labeled 'SYN' corresponds to the circuit optimized us-

ing the pre-layout technique (i.e., output phase assignment) presented in Section 3.2. The two post-layout optimization approaches, simultaneous elimination without compaction (labeled as 'SE' in the table) and iterative elimination and compaction (labeled as 'IEC'), are applied on each synthesized circuit in 'SYN'. For comparison, each original circuit is also synthesized without any output phase assignment. The sub-table labeled 'DUP' corresponds to these circuits. For 'DUP', 'SYN', 'SE', and 'IEC', we report the gate number, area, power, and delay for each (physically) implemented circuit.

From Table 1, we observe that applying the pre-layout logic optimization (presented in Section 3.2) does not degrade the area, timing, and power performance of the Domino circuits; 'DUP' and 'SYN' are comparable in all aspects. Most important, the pre-layout logic optimization allows significant reductions in area, power, and delay after postlayout optimization. 'SE' has 30% lower power when compared with 'DUP'. Moreover, it achieves a modest 2% reduction in delay, while maintaining the same overall area. 'IEC' has less area (81%) than 'DUP', 'SYN', and 'SE'. Moreover, 'IEC' achieves more delay reduction (9%) than 'SE' over both 'DUP' and 'SYN'. However, 'IEC' achieves less power reduction (21%) than 'SE' as 'SE' eliminates more fan-in cones and cells (without compaction). Number of iterations and run time for 'IEC' are also reported in Table 1, where 'prog cpu' is the run time for logic level preprocessing and layout compaction whereas 'total cpu' is for the whole flow. To evaluate the effectiveness of the output phase assignment algorithm presented in Section 3.2, we also apply 'SE' and 'IEC' on each circuit in 'DUP'. We observe that such flows ('DUP' followed by 'SE' and 'DUP' followed by 'IEC') result in 25% and 19% more gates than the flows that have 'SYN' followed by 'SE' and 'SYN' followed by 'IEC', respectively. Such degradations are also reflected in the area and power consumptions.

To the best of our knowledge, all previous work on Domino logic synthesis reported results without physical implementation. Therefore, no direct comparison can be made with previous work. [6] reported 14% area reduction using output phase assignment and [8] had similar results. In [2], gate area was reduced by 25% whereas the delay increased by 3%. In contrast, our results from 'SE' indicate that the gate area (not placement area) can be reduced by 24%, and the delay by 2%. Our post-layout optimization may be applied to the synthesized netlist using the output phase assignment in [6] and [8]. Further improvements on their results are possible.

Spice simulations are also performed to the final circuits to verifty the robustness of the circuits. Figure 7 shows the simulation results on the inputs to a candidate gate (2-input AND) in each implementation ('SYN', 'SE', and 'IEC') for the benchmark circuit C2670. As indicated by the simulation results, the ELDDB constraint for the candidate cell is still satisfied after 'SE' and 'IEC'; the delay differences between the early and late transitions (see Eqn. (1)) are 2.52ns, 2.50ns, and 2.59ns for 'SYN', 'SE', and 'IEC', respectively.

6. CONCLUSION AND FUTURE WORK

We propose a synthesis scheme to reduce the duplication cost caused by logic reconvergent paths in Domino logic. Two post-layout optimization approaches are presented to guarantee the robustness of our idea. Experimental results show that significant reductions in power, area, and delay

	Circuit	C432	C499	C880	C1355	C1908	C2670	C3540	C5315	C6288	C7552	ratio
ORI	#gates	258	563	434	548	632	1278	1252	2050	2432	3095	1
DUP	#gates	472	990	702	979	1105	2232	2488	3403	4705	5899	1.80
	Area (x1000μm^2)	72.1	205.9	145.9	207.4	257.0	428.2	547.1	749.3	942.2	1190.5	1
	Power (mW)	40.0	102.4	93.2	112.7	114.7	186.3	225.4	258.8	271.3	396.1	1
	delay (ns)	18.8	15.6	15.1	16.9	21.0	18.8	30.4	27.9	31.9	25.2	1
SYN	#gates	468	998	693	968	1110	2203	2392	3371	4743	5712	1.78
	Area (x1000μm^2)	72.5	207.0	144.7	206.1	258.6	424.8	532.4	746.5	950.9	1162.7	0.99
	Power (mW)	40.4	106.8	91.2	112.0	116.5	188.1	219.2	255.2	279.7	380.3	1.00
	delay (ns)	18.7	15.3	15.2	17.0	20.8	18.8	30.5	27.0	32.3	24.6	0.99
SE	#gates	352	776	505	792	854	1577	1813	2640	4144	4195	1.37
	Area (x1000μm^2)	72.5	207.0	144.7	206.1	258.6	424.8	532.4	746.5	950.9	1162.7	0.99
	Power (mW)	32.8	69.5	58.2	82.6	75.7	123.3	152.0	176.5	193.4	293.4	0.70
	delay (ns)	18.0	16.0	14.8	16.4	20.1	19.0	29.5	26.8	32.9	24.0	0.98
IEC	#gates	395	848	571	833	904	1881	1984	2813	4301	4866	1.51
	Area (x1000μm^2)	58.2	168.4	119.1	160.8	209.3	337.0	448.9	605.0	839.5	931.8	0.81
	Power (mW)	35.0	82.9	69.2	90.4	85.5	151.3	166.2	189.6	212.5	328.5	0.79
	delay (ns)	16.6	14.0	16.0	14.2	17.0	16.9	29.1	25.5	29.4	22.5	0.91
	#iterations	3	3	3	4	3	4	3	4	2	3	-
	prog cpu (m)	1	3	2	4	4	9	10	16	17	23	-
	total cpu (m)	9	16	12	23	20	42	37	103	70	108	-

Table 1: Comparison of results of original circuits (ORI), duplicated circuits without phase assignment (DUP), synthesized circuits (SYN), after simultaneous elimination without compaction (SE), and after iterative elimination and compaction (IEC).

are achieved by our approach. The gap between 'SE' and 'IEC' in terms of power also indicates that further improvements in area, power, and performance for the 'IEC' approach are possible. We will continue to explore the idea of iterative elimination and compaction in the future.

7. ACKNOWLEDGMENTS

8. REFERENCES

[1] A. Agnihotri, M. C. Yildiz, A. Khatkhate, A. Mathur, S. Ono, and P. H. Madden. Fractional cut: improved recursive bisection placement. In *Proc. Int. Conf. on Computer Aided Design*, Nov. 2003.

[2] K. W. Kim, T. Kim, C. L. Liu, and S. M. Kang. Domino logic synthesis based on implication graph. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 21(2):232–240, Feb. 2002.

[3] P. Patra and U. Narayanan. Automated phase assignment for the synthesis of low power Domino circuits. In *Proc. Design Automation Conf*, pages 379–384, June 1999.

[4] M. R. Prasad, D. Kirkpatrick, R. K. Brayton, and A. Sangiovanni-Vincentelli. Domino logic synthesis and technology mapping. In *Proc. Int. Workshop on Logic Synthesis*, May 1997.

[5] R. Puri. Design issues in mixed Static-Domino circuit implementations. In *Proc. IEEE Int. Conf. on Computer Design*, pages 270–275, Oct. 1998.

[6] R. Puri, A. Bjorksten, and T. E. Rosser. Logic optimization by output phase assignment in dynamic logic synthesis. In *Proc. Int. Conf. on Computer Aided Design*, pages 2–8, Nov. 1996.

[7] S. M. Reddy. Complete test sets for logic functions. *IEEE Trans. on Computers*, C-22(11):1016–1020, Nov. 1973.

[8] M. Zhao and S. S. Sapatnekar. Dual-monotonic domino gate mapping and optimal output phase assignment of domino logic. In *Proc. IEEE Int. Symp. on Circuits and Systems*, pages 309–312, May 2000.
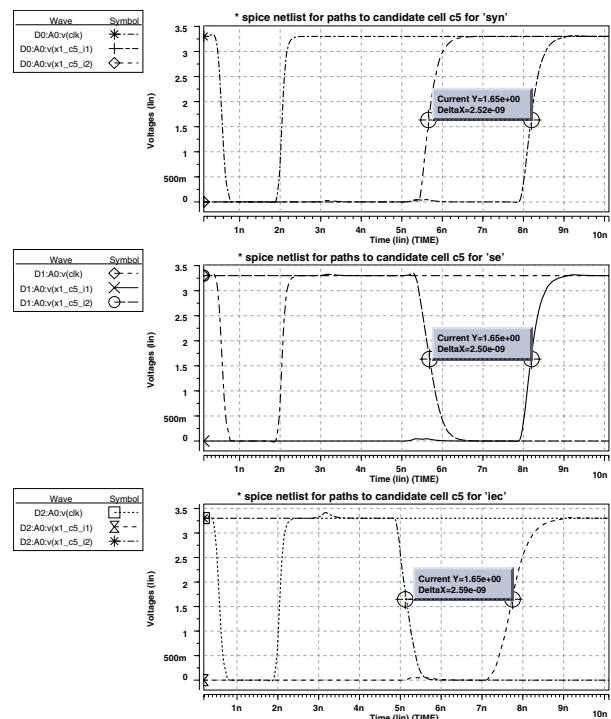
Figure 7: Spice simulation on a candidate in C2670.