INTEGRATION the VLSI journal 58 (2017) 236–244

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Minimizing detection-to-boosting latency toward low-power error-resilient circuits

Chih-Cheng Hsu[a], Masanori Hashimoto[b], Mark Po-Hung Lin[a,*]

[a] Department of Electrical Engineering and AIM-HI, National Chung Cheng University, Taiwan
[b] Department of Information Systems Engineering, Osaka University, Japan

ARTICLE INFO

Keywords:
Latch clustering
Flip-flop clustering
Time borrowing
Local boosting
Dynamic voltage scaling
Reliability
Resilient circuits

ABSTRACT

Dynamic voltage scaling (DVS) has become one of the most effective approaches to achieve ultra-low-power SoC. To eliminate timing errors arising from DVS, several error-resilient circuit design techniques were proposed to detect and/or correct timing violations. The most recently proposed time-borrowing-and-local-boosting (TBLB) technique has the advantage of lower power consumption and less performance degradation due to the needlessness of pipeline stalls. On the other hand, to make the best use of the TBLB technique, the latency from error detection to voltage boosting for TBLB latches must be carefully considered, especially during physical design. To address this issue, this paper first introduces the behavior of TBLB circuits, and then presents two major design styles of TBLB latches, including TBLB macros and multi-bit TBLB latches, for reducing detection-to-boosting latency. The corresponding physical synthesis methodologies for both design styles are further proposed. Experimental results based on the IWLS benchmarks show that the proposed physical synthesis approach for resilient circuits with multi-bit TBLB latches is very effective in reducing the delay of both combinational and error-detection circuits, which indicates better circuit reliability. To our best knowledge, this is the first work in the literature which introduces the physical synthesis methodologies for TBLB resilient circuits.

1. Introduction

Dynamic voltage scaling (DVS) has become one of the most effective approaches to achieve ultra-low power SoC design. To eliminate the timing errors arising from DVS, several timing error resilient circuits or error detection latch/latch design techniques were proposed to dynamically detect timing violations and to control the supply voltage based on in situ circuit operations. In addition to DVS, applying timing error resilient circuits can also prevent the timing violation induced by dynamic variations, such as process variations, soft errors, and transistor aging degradation. Existing timing error resilient circuit techniques can be classified into four major categories: (1) canary latches [1,2]; (2) delay monitors [3,4]; (3) razor latches [5–10], and (4) time-borrowing-and-local-boosting (TBLB) latches [11,12].

The canary latch [1] consists of a main latch and a shadow latch. As the shadow latch can discover timing error earlier than that of main latch in the data path due to extra timing margin, the predicted timing error can be corrected by scaling either voltage or frequency. The delay monitor [3] consists of a delay chain and a phase detector. The delay is directly measured by the delay chain, which replicates the critical path of the design with an additional delay margin. Once the delay is predicted by the phase detector, either voltage or frequency scaling is applied to shorten delay. However, these two techniques do not allow timing error occurrence and require a timing margin/guardband to ensure correct circuit functionalities. The circuit performance may be degraded due to the overly conservative operation.

To further reduce the overestimated timing margin leading to energy-efficient operation, the Razor latch [5] was proposed. A shadow pulsed latch is incorporated to detect timing errors on the main latch in a data path. Once the timing error is detected, an extra cycle is required to perform instruction replay for error correction. Although cycle overheads can be suppressed by extra hardware design, such as pipeline stalls, they significantly complicate the whole circuit. In addition, the throughput of the whole system may be greatly reduced when replaying a large number of instructions, and thus degrades the system performance.

Instead of applying instruction replay, a novel TBLB resilient circuit with TBLB latches [11,12] was proposed to detect and correct timing violations, as shown in Fig. 1(a). It consists of three major components: a transition detector (TD), a level-converting pulse-latch (PL) driven by

* Corresponding author.
E-mail address: marklin@ccu.edu.tw (M.P.-H. Lin).

http://dx.doi.org/10.1016/j.vlsi.2017.01.002

Available online 10 January 2017
0167-9260/

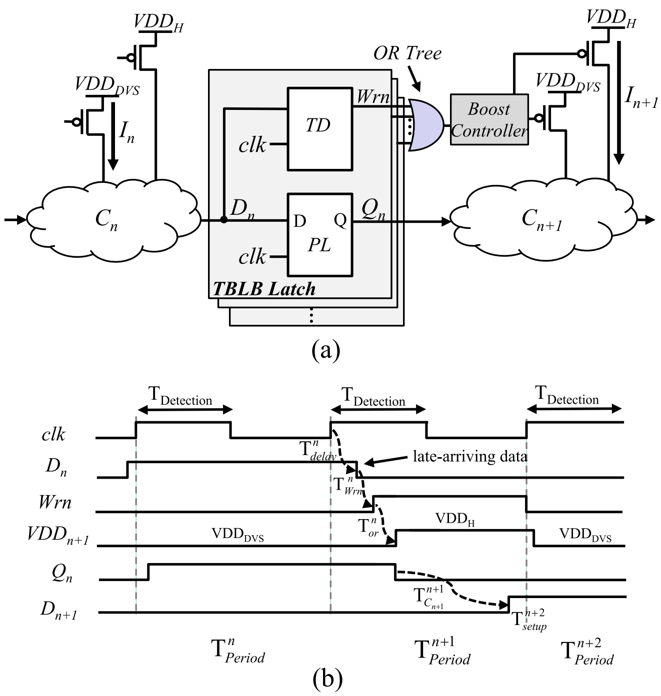

Fig. 1. The architecture of a TBLB resilient circuit and its timing diagram [11].

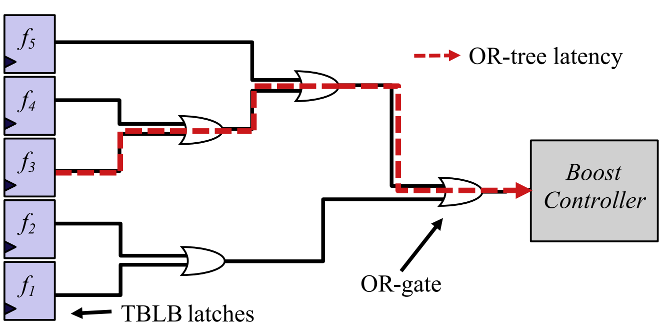

Fig. 2. An OR-tree with five TBLB latches, four OR-gates, and a boost controller.

a pulse generator (PG), and a boost controller. During the normal operation, the system is expected to operate at a lower supply voltage, VDD_{DVS}, for lower power consumption. If the delay of the nth combinational logic, C_n, is longer than the clock period, T_{period}, as seen in Fig. 1(b), where D_n arrives late, due to circuit aging and other reliability effects, the transition detector will flag the warning signal, Wrn. The warning signals are then transferred to the boost controller through a large or multi-level OR gate (i.e. an OR tree), as shown in Fig. 2. Since the combinational logic delay at C_n exceeds its cycle limit and requires time borrowing from the next stage, C_{n+1}, to ensure correct data, C_{n+1} is required to speed up immediately by boosting the local voltage to a higher supply voltage, VDD_H, to prevent timing error propagation. Note that similar error correction can also be implemented with body biasing, where speed boosting is achieved by forward body bias. Consequently, timing violations can be rescued without extra-cycle or performance overhead. Fig. 3 collapses the timing diagram in Fig. 1(b), and details the timing information at the stage, C_{n+1}, where $T^n{}_{delay}$ is the late-arriving delay from C_n, $T_{Wrn}{}^n$ is the warning detection delay of transition detector, $T_{or}{}^n$ is the propagation delay through the OR-tree, $T^{n+1}_{C_{n+1}}$ is the combinational logic delay of C_{n+1}, and T^{n+2}_{setup} is the setup time for C_{n+2}.

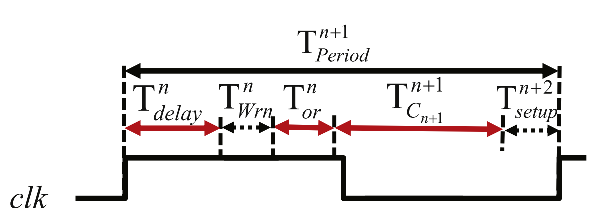

Fig. 3. Detailed timing information at the stage, C_{n+1}, when it is borrowed by the previous stage, C_n, because the combinational logic delay at C_n exceeds the clock period.

Because of the elimination of extra-cycle overhead with TBLB resilient circuits, the delay margin of both error detection and correction must be strictly limited within a clock period. If a timing delay occurs at C_n, the timing constraint at C_{n+1}, as seen in Eq. (1), must have to be satisfied such that the timing violation at C_n can be rescued without data error and performance overhead

$$T^n_{delay} + T^n_{Wrn} + T^n_{or} + T^{n+1}_{C_{n+1}} + T^{n+2}_{setup} \le T^{n+1}_{Period}. \tag{1}$$

In Eq. (1), both $T_{Wrn}{}^n$ and T^{n+2}_{setup} are constants, which were determined when a TBLB latch is designed. We shall minimize $T_{delay}{}^n$, $T_{or}{}^n$ and $T^{n+1}_{C_{n+1}}$ during logic and physical synthesis such that the timing constraint is satisfied. With circuit aging, the delay of combinational logic cells becomes much longer, and hence the delay margin for error-correction is even more stringent. Therefore, it is essential to minimize the detection-to-boosting latency in TBLB error-resilient circuits.

It should be noted that such TBLB technique might not be suitable for ultra-high-performance design due to some overhead. However, it is applicable for lower-speed and ultra-low-power applications. A real-chip implementation [11] for the application to digital hearing aids has confirmed the feasibility of the TBLB technique. According to [11], given a performance specification with fixed VDD_H, the TBLB technique is applied together with dynamic voltage scaling for lower supply voltage, VDD_{DVS}, resulting in even lower power consumption.

Recent physical synthesis approaches [9,13] dealt with timing error resilient circuits. However, they did not consider the special timing requirement for TBLB error-resilient circuits. These works mainly focused on reducing hold buffer penalties arising from short paths instead of shortening the error-detection delay, or the detection-to-boosting latency, for larger delay margin of error correction in TBLB error-resilient circuits. In addition, due to the reliability issue caused by circuit aging, the delay of combinational logic cells of a pipeline stage may degrade more than 9% in circuit speed over ten years, as shown in Fig. 4. With circuit aging, the delay of combinational logic cells becomes much longer such that the delay margin for error-correction is even smaller.

In this paper, we investigate new design methodologies for TBLB low-power error-resilient circuits. The contributions of this paper are summarized in the following:

- We present the behavior, design challenge, and required physical

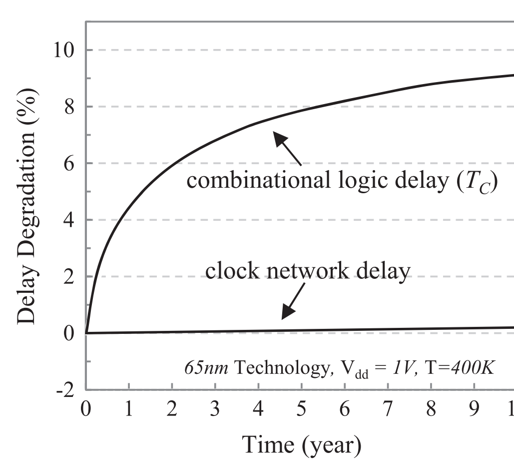

Fig. 4. Analysis of delay degradation of a single pipeline circuit containing combinational logic cells and clock network with latches induced by circuit aging over time [14] based on ISCAS89 circuit benchmarks [15].

design styles of TBLB error-resilient circuits.

- Different from the previous works which do not consider the special timing requirement for TBLB latches, we propose a novel physical synthesis flow and algorithms for TBLB resilient circuits. Our approach can simultaneously minimize the delay of error-detection circuits and that of ordinary data paths.
- In order to reduce the delay of TBLB error-detection circuits and consequently increase the margin for TBLB error-correction, we propose a novel OR-tree-latency-aware TBLB latches clustering to minimize both OR-tree wirelength and latency with Hamiltonian path and dynamic-programming (DP) formulations.
- Experimental results based on the IWLS-2005 benchmark show that the proposed approach applying multi-bit TBLB latches is very effective in reducing the delay of both combinational and error-detection circuits compared with TBLB macro based approach.
- To our best knowledge, this is the first work in the literature which studies the physical synthesis methodologies while minimizing detection-to-boosting latency for TBLB resilient circuits.

The remainder of this paper is organized as follows. Section 2 investigates some design styles for TBLB resilient circuits. Section 3 details the proposed physical design flow and the corresponding algorithms. Section 4 reports the experimental results. Finally, Section 5 concludes this paper.

2. Design styles for TBLB resilient circuits

2.1. Physical design styles of TBLB latches

Due to the aforementioned critical and stringent timing constraint, it is essential to investigate better design styles and design methodologies for TBLB resilient circuits. Different physical design styles of TBLB latches may have great impact on the circuit performance. We first introduce two major physical design styles, including *TBLB macros* and *multi-bit TBLB latches*, as shown in Fig. 5, which can effectively reduce detection-to-boosting latency of TBLB error-resilient circuits, and then demonstrate the impacts on detection-to-boosting latency and signal-path delay.

- *TBLB macros*: A TBLB macro contains a boost controller, an OR tree, several pulse generators, and the whole TBLB latches of the same pipeline stage. Such design style has the advantages of integrated and compacted TBLB latches at each pipeline stage of a TBLB resilient circuit, which makes the whole circuit easy to design and debug. However, it may introduce more critical signal paths in the combinational circuits among different pipeline stages because of longer interconnections.
- *Multi-bit TBLB latches*: A multi-bit TBLB latch cell consists of only one pulse generator and several 1-bit TBLB latches. Both OR tree and boost controller are not included in the cell. Such design style has more flexibilities in optimizing the combinational logic cells in each data path together with the logic cells in OR trees, boost controllers, and multi-bit TBLB latch cells among different pipeline stages. However, it requires more sophisticated design methodologies and algorithms for achieving higher circuit performance/reliability and lower power consumption.

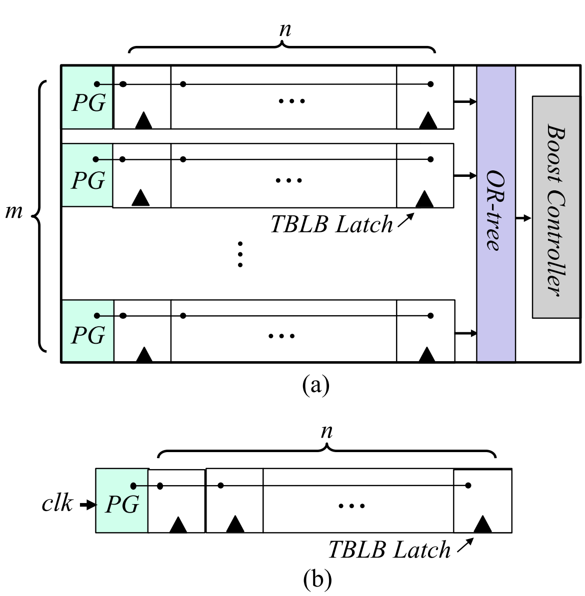

Fig. 5. Two design styles of TBLB latches in TBLB error-resilient circuits for reducing detection-to-boosting latency: (a) A TBLB macro and (b) a multi-bit TBLB latch.

Fig. 6 further shows three different physical implementations of TBLB resilient circuits with different design styles of TBLB latches, which are discrete 1-bit TBLB latches, integrated TBLB macros, and distributed multi-bit TBLB latches. The design style with discrete 1-bit TBLB latches may have more gates in the OR-tree, which results in much larger T_{or}. Although the design style with integrated TBLB macros will result in the smallest T_{or}, it may introduce more critical paths in the combinational circuits among different pipeline stages because of longer interconnections. Compared with discrete 1-bit TBLB latches and integrated TBLB macros, the design style with distributed multi-bit TBLB latches is expected to achieve the best tradeoff among T_{delay}, T_{or}, and $T_{C_{n+1}}$ during physical synthesis.

The physical implementations with discrete 1-bit TBLB latches and integrated TBLB macros can be automatically generated by modern physical synthesis tools with given TBLB latch cells or macros in the library. For the physical implementation with distributed multi-bit TBLB latches, although recent studies have explored some latch/latch merging and multi-bit latch/latch generation methods [16–26] during physical synthesis, all of them tried to merge as many latches as possible while satisfying general timing and physical design constraints. None of them consider the delay of both combinational and error-detection circuits as the first-order design objective, whereas it is essential in TBLB resilient circuits.

3. Physical synthesis flow and algorithms for TBLB resilient circuits with multi-bit TBLB latches

Given a TBLB error-resilient circuit, which contains combinational logic cells, sequential logic cells including TBLB latches and their pipeline stages, maximum capacitance loading of a pulse-generator, and multi-bit TBLB latches with different bit numbers, we want to generate a legalized non-overlapped placement for the TBLB resilient circuit with multi-bit TBLB latches such that the delay of combinational and error-detection circuits, T_{delay}, $T_{C_{n+1}}$, and T_{or}, is minimized while satisfying the maximum loading constraint of all pulse generators (i.e. the maximum bit number of multi-bit TBLB latches), and other common physical design rules and/or constraints.

Based on the problem formulation, we propose a novel physical synthesis flow for TBLB error-resilient circuits, as shown in Fig. 7, which consists of five major steps: (1) Initial placement, (2) OR-tree-latency-aware TBLB latch clustering, (3) PG-group-aware incremental placement, (4) multi-bit TBLB latch replacement, and (5) OR-tree synthesis. At the beginning, all TBLB latches are one-bit. The initial placement produces a good solution in terms of wirelength, density, and other placement constraints. Based on the initial placement, the TBLB latches are then clustered according to the construction of the OR-tree with latency minimization, which is followed by PG group extraction for all TBLB latches. The incremental placement is further performed according to PG groups of TBLB latches for timing optimization. The multi-bit TBLB latches are finally generated, and

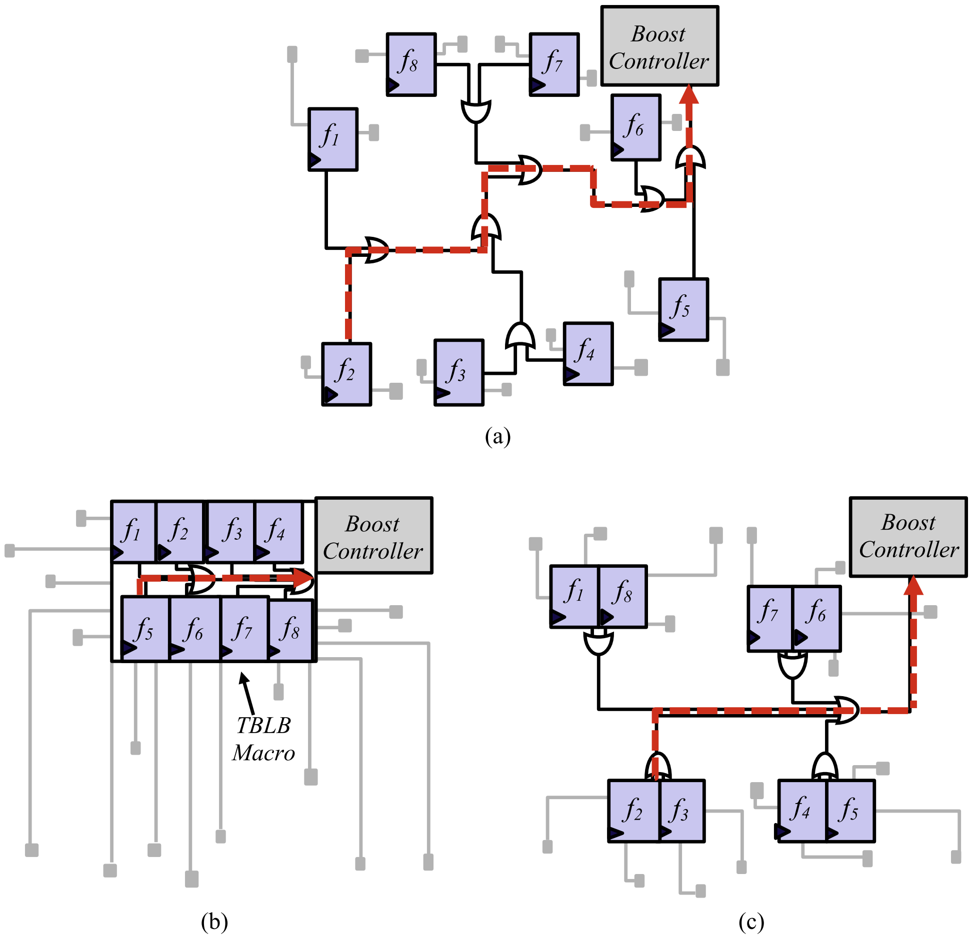

Fig. 6. Physical implementations of TBLB resilient circuits with different design styles of TBLB latches. (a) Discrete 1-bit TBLB latches resulting in longer propagation delay of an OR tree. (b) An integrated TBLB Macro leading to many critical paths in combination circuits. (c) Distributed multi-bit TBLB latches achieving better tradeoff between the propagation delay of OR trees and that of combination circuits.

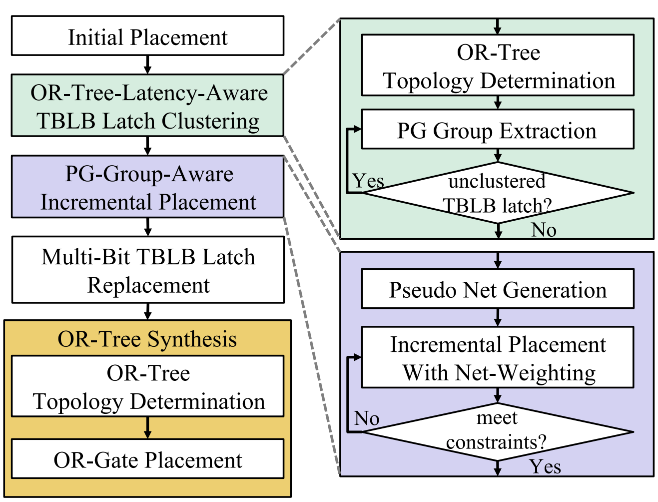

Fig. 7. The proposed physical synthesis flow for TBLB error-resilient circuits.

the OR-trees are re-synthesized to achieve the shortest OR-tree delay with multi-bit TBLB latches.

3.1. Initial placement

Since minimizing signal net wirelength and placement density are the most important objectives for a general global placement problem, we consider both objectives and try to find the best tradeoff between the two objectives at the beginning stage. Inputting a design netlist, we first perform initial placement based on the analytical placer [27], to obtain the initial locations of all cells. The initial placement is formulated with an unconstrained minimization problem as follows:

$$\min \widehat{W}(\mathbf{x}, \mathbf{y}) + \lambda_d \sum (\widehat{D}_{b_i}(\mathbf{x}, \mathbf{y}) - D_{MAX})^2, \quad (2)$$

where $\widehat{W}(\mathbf{x}, \mathbf{y})$ is the log-sum-exponential (LSE) wirelength function for all signal nets, $\widehat{D}_{b_i}(\mathbf{x}, \mathbf{y})$ is a smoothed density function for each bin, D_{MAX} is the maximum allowable placement density, and λ_d is a Lagrange multiplier, which controls the weighting of the density. We solve a series of the unconstrained optimization problem in Eq. (2) based on the conjugate gradient method with increasing λ_d until the cells are evenly distributed throughout the chip area. Similar to [24], we integrate our analytical placer with a timer, and apply a net-weighting method to enlarge the wirelength costs of the timing critical nets in the objective function during the last few iterations.

After performing the initial placement, we can capture more accurate physical information to optimize the locations of all TBLB error-detection latches for reducing the delay of error-detection circuits in the following steps.

3.2. OR-tree-latency-aware TBLB latch clustering

Once the cells are evenly distributed with minimized wirelength, we then perform OR-tree-latency-aware TBLB latch clustering to reduce the delay of error-detection circuits and clock sinks without degrading circuit performance. The proposed OR-tree-latency-aware TBLB latch clustering consists of two major steps: (1) OR-tree topology determina-

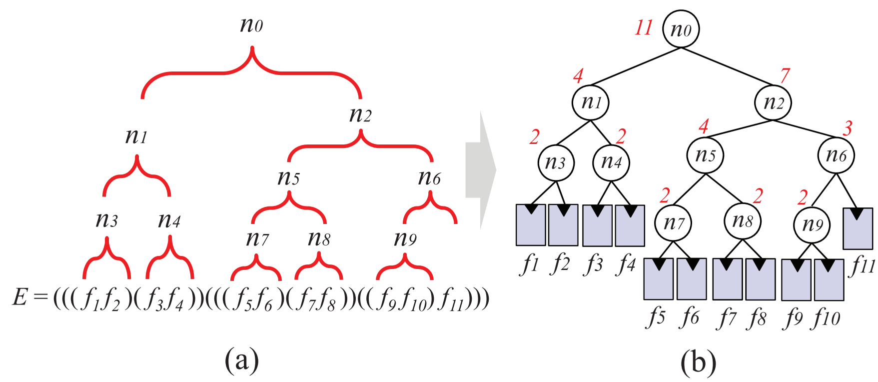

Fig. 8. (a) An example for OR-tree topology determination after dynamic programming. (b) The corresponding OR-tree topology with node weights.

tion and (2) PG group extraction:

$$D[i, j] = \begin{cases} d_{f_i} & \text{if } i = j, \\ \min\limits_{i \le k < j} \{\max(D[i, k], D[k+1, j]) + d_{OR} + d_{wire}\} & \text{if } i < j. \end{cases} \quad (3)$$

3.2.1. OR-tree topology determination

Since we want to construct an OR-tree topology with minimized wirelength, we first construct a TBLB latch chain to represent the adjacency relationship among different TBLB latches with respect to their physical locations. In order to minimize the total distance of the TBLB latch chain, we model the TBLB latch chain construction problem as a Hamiltonian path problem, and find an optimal TBLB latch chain by searching the shortest Hamiltonian path [28]. The closer TBLB latches in a TBLB latch chain will have higher opportunity to be clustered into the same branch or neighboring branches of an OR-tree topology. In addition, minimizing the total distance of a TBLB latch chain can help to reduce total wirelength of the OR-tree when performing OR-tree synthesis.

After obtaining the TBLB latch chain by searching the shortest Hamiltonian path, we formulate the problem of OR-tree topology determination as a dynamic programming problem by inputting the TBLB latch chain. The objective, $D[i, j]$, is to parenthesize the sub-chain of TBLB latches, $f_i \ldots f_j$, in order to minimize the latency of OR-tree, which can be defined in Eq. (3). d_{f_i} is the negative slack of f_i from C_n. d_{OR} is the intrinsic delay of OR-gate, and d_{wire} is the estimated delay of the wire. By using d_{f_i} of each f_i as the weight, we can estimate the locations of all OR-gates based on the force-directed method and calculate the latency of each sub-path of OR-tree as its solutions during our algorithms. In Algorithm 1, $D[i, j]$ is the shortest OR-tree latency between $f_i \ldots f_j$ and $S[i, j]$ is the value of k such that the optimal parenthesization of $f_i \ldots f_j$ splits between f_k and f_{k+1}, as calculated in Lines 6–18. We can derive the shortest OR-tree latency by recursive referring the tables of D and S in a bottom-up fashion. In addition, the optimal parenthesization of the TBLB latch chain, $f_i \ldots f_j$, can also be obtained by recursive computing *OptimalParenthesization*(S, i, j) in Algorithm 2.

Algorithm 1. OR-tree topology determination.

Require: A TBLB latch chain, E.
1: $n \leftarrow E.size()$;
2: Let $D[1 \ldots n, 1 \ldots n]$ and $S[1 \ldots n-1, 2 \ldots n]$ be new tables;
3: **for all** $i \leftarrow 1$ to n **do**
4: $D[i, i] = d_{f_i}$;
5: **end for**
6: **for all** $l \leftarrow 2$ to n **do**
7: **for all** $i \leftarrow 1$ to $n - l + 1$ **do**
8: $j = i + l - 1$;
9: $D[i, j] = \infty$;
10: **for all** $k \leftarrow i$ to $j - 1$ **do**
11: $q = \max(D[i, k], D[k+1, j]) + d_{OR} + d_{wire}$;
12: **if** $q < D[i, j]$ **then**
13: $D[i, j] = q$;
14: $S[i, j] = k$;
15: **end if**
16: **end for**
17: **end for**
18: **end for**
19: **return** D and S;

Algorithm 2. Optimal Parenthesization(S, i, j).

if $j - i = 0$ **then**
print "f"$_i$;
else
print "(";
Optimal Parenthesization$(S, i, S[i, j])$;
Optimal Parenthesization$(S, S[i, j] + 1, j)$;
print ")";
end if

Once the optimal parenthesization of the TBLB latch chain is obtained, we then construct the corresponding OR-tree topology for PG grouping extraction. The input for the OR-tree topology construction is a set of nodes, which represent the corresponding TBLB latches, respectively, and the initial weight of each node is set to 1. We first trace the TBLB latch chain according to the parentheses from inner to outer, and add the nodes to the sub-chain of TBLB latches when there is a pair of parentheses. The weight of each node is then assigned by summing up the weight of its child nodes. Fig. 8 shows an example of eleven TBLB latches, $f_1, f_2 \ldots f_{11}$ in chain, E, with optimal parenthesization and the corresponding weighted OR-tree topology. Based on the weighted OR-tree topology, as seen in Fig. 8(b), the nodes, whose weights are more than two, will correspond to either one multi-input OR gate or several 2-input OR gates, which is determined by PG group extraction.

3.2.2. PG group extraction

After constructing the weighted OR-tree topology, we further extract the PG groups from root to leaves of the OR-tree topology according to the maximum capacitance loading constraint. Intuitively, grouping the TBLB latches having the same branch or the nearest branches in the OR-tree topology can help to reduce the total wirelength of the OR-tree as well as the OR-tree latency. In addition to the capacitance loading constraint, we estimate the total signal net wirelength of each candidate of PG group, g_i, and select the candidate of PG groups which contain total signal net wirelength of g_i within 3× of $W_{New}^{g_i}/W_{Ori}^{g_i}$, where $W_{Ori}^{g_i}$ and $W_{New}^{g_i}$ are the estimated total signal net wirelengths of g_i before and after PG grouping, respectively. With this constraint, the total signal net wirelength of selected g_i can be prevented from large increase and timing quality can be maintained when grouping g_i during PG-group-aware incremental placement. The algorithm of PG group extraction, as shown in Algorithm 3, iteratively clusters the TBLB latches based on the result of weighted OR-tree topology until all the nodes of the weighted OR-tree topology are traced or all the TBLB latches are grouped.

Algorithm 3. PG group extraction.

```
require: An OR tree, T;
Set all v ∈T as unvisited;
Sort all v ∈T in the ascending order with respect to its tree level;
for all v ∈T with the sorted order do
  if v is unvisited then
    if the subtree of v satisfies the constraints of capacitance loading and wirelength increment ratio, then
      Cluster all vertices in the subtree of v;
      Set all vertices in the subtree of v as visited;
    end if
  end if
end for
```

3.3. PG-group-aware incremental placement

After applying OR-tree-latency-aware TBLB latch clustering according to OR-tree latency and physical locations of TBLB latches, PG-group-aware incremental placement is performed to progressively place TBLB latches of the same group close to each other for reducing the delay of error-detection circuits. In addition to the placement adjustment among TBLB latches, the locations of all the other cells can also be refined such that the placement density constraints can be met without degrading circuit performance.

To achieve this, we first calculate the target location of each PG groups by the force-directed method according to Eqs. (4) and (5), where (x_i, y_i) corresponds to the original location of each TBLB latch, f_i, in a PG group, and the force of f_i is denoted by d_{f_i}. Since the larger d_{f_i} implies that the data path related to f_i is more critical than other data paths, we would like to locate the target location of the corresponding PG group closer to f_i such that the wirelength of the path related to f_i after moving f_i to a new location does not increase too much during the PG-group-aware incremental placement. Fig. 9 gives an example of a PG group with four TBLB latches. In Fig. 9(a), the four TBLB latches, f_1, f_2, f_3, and f_4, have different force values, 3 ns, 4 ns, 2 ns, and 4 ns, respectively. Since the force values of the TBLB latches, f_2 and f_4, are larger than the other two, the target location of this PG group is located closer to f_2 and f_4, but farther from f_1 and f_3:

$$x_{target} = \frac{\sum_i d_{f_i} x_i}{\sum_j d_{f_i}} \tag{4}$$

$$y_{target} = \frac{\sum_i d_{f_i} y_i}{\sum_j d_{f_i}} \tag{5}$$

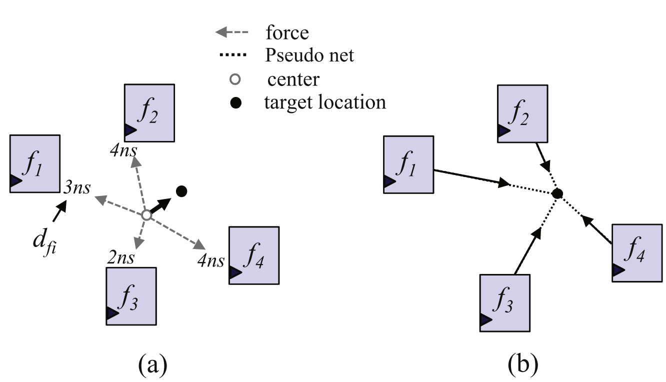

Fig. 9. An example of a PG-group-aware incremental placement for a PG group containing four TBLB latches. (a) Determination of the target location of the PG group based on the force-directed method. (b) Incremental placement of the four latches in the same PG group with pseudo nets and attracting force.

We repeatedly applied the placer [27] with additional pseudo nets during incremental placement. In order to place all TBLB latches of the same group closer to each other, the pseudo nets are introduced. Each pseudo net connects the target location and one of the TBLB flip-flops in the same group such that the delay of error-detection circuits can be reduced. To strengthen the attractions, the weight of each pseudo net should be greater than the weight of the ordinary signal nets, which is about 10× according to our experimental study. Fig. 9(b) shows that the four TBLB latches in the same group are connected and attracted to the target location by the generated pseudo nets with strengthened attractions.

3.4. Multi-Bit TBLB latch replacement and OR-tree synthesis

Once all TBLB latches of the same PG group are closed enough to the target location, the TBLB latches in each PG group are replaced with a multi-bit TBLB latch. We reconstruct the OR-tree topology because the original OR-tree topology might be slightly changed after multi-bit TBLB latch replacement. Based on the reconstructed OR-tree topology, we can calculate the optimal OR gate locations. Consequently, the resulting OR-tree with optimized wirelength and latency can be obtained.

4. Experimental results

We implemented our algorithms in C/C++ programming languages on a 2.26 GHz Intel Xeon machine under the Linux operating system, and integrated with the placer based on NTUplace3 [27]. We experimentally tested our algorithms on the five OpenCores [29] circuits in the IWLS-2005 benchmark suite [30] with the Nangate 45 nm Open cell Library [31]. Based on the library, a pulse generator can drive at most 10 TBLB latches, and the available multi-bit TBLB latches range from 1 to 10 bits. Table 1 lists the names of the circuits (Circuit), the numbers of combinational logic cells (# of Comb. Logic Cells), the numbers of sequential logic cells (# of Seq. Logic Cells), the numbers of nets (# of Nets), and the clock cycle time (T_{Period}).

We conducted three sets of experiment to show the effectiveness of our approach. The first set of experiment will compare different design styles with multi-bit TBLB latches and TBLB macros. The second set of experiment will further compare the proposed approach with multi-bit pulsed-latch generation [16]. The third set of experiment will finally demonstrate the importance of considering OR-tree topology during OR-tree-latency-aware TBLB latch clustering.

4.1. Comparison of the design styles with TBLB macros and multi-bit TBLB latches

In the first set of experiment, we compared the design style with

Table 1
Five OpenCores circuits [29] in IWLS-2005 benchmark [30].

Circuit	# of Comb. Logic Cells	# of Seq. Logic Cells	# of Nets	T_{Period} (ns)
ac97ctrl	9656	2199	11,637	0.44
aescore	20,265	530	20,626	1.21
memctrl	10,357	1083	11,280	1.55
pcibridge32	13,457	3359	16,726	1.01
wbconmax	28,264	770	29,675	0.92

Table 2
Comparisons of total signal net wirelength (WL), OR-tree latency (T_{or}), clock wirelength (CWL), worst negative slack (WNS), total negative slack (TNS), and runtime (Time) based on both design styles, TBLB macros and multi-bit TBLB latches.

Circuit	The design style with TBLB macros					
	WL × 10^8 (nm)	T_{or} (ns)	CWL × 10^7 (nm)	WNS (ns)	TNS (ns)	Time (s)
ac97ctrl	8.10	0.31	1.91	1.28	598.65	787
aescore	5.90	0.15	2.84	1.32	72.99	127
memctrl	6.32	0.22	1.44	1.68	170.45	216
pcibridge32	15.05	0.36	3.23	2.10	609.73	1905
wbconmax	11.38	0.19	3.03	1.80	357.11	514
Comp.	1	1	1	1	1	1

Circuit	The design style with multi-bit TBLB latches resulting from the proposed approach					
	WL × 10^8 (nm)	T_{or} (ns)	CWL × 10^7 (nm)	WNS (ns)	TNS (ns)	Time (s)
ac97ctrl	2.98	0.33	2.38	0.92	266.14	836
aescore	4.15	0.25	2.89	0.86	37.56	127
memctrl	2.67	0.26	1.60	0.84	89.54	231
pcibridge32	4.80	0.46	3.90	0.66	113.26	2395
wbconmax	9.10	0.33	3.18	1.58	303.60	588
Comp.	0.522	1.392	1.125	0.612	0.504	1.107

TBLB macros, as seen in Fig. 6(b), resulting from the analytical placer [27], and the design style with multi-bit TBLB latches, as seen in Fig. 6(c), resulting from the proposed approach. After obtaining a legal placement with either TBLB macros or multi-bit TBLB latches, hold buffer insertion/short path padding [13] should be further performed to fix hold violations.

Table 2 lists the names of the benchmark circuits (Circuit), total signal net wirelength (WL), OR-tree delay (T_{or}), clock wirelength (CWL), worst negative slack (WNS), total negative slack (TNS), and runtime (Time) for the two approaches based on different design styles of TBLB resilient circuits. The clock wirelength was obtained based on [32], while the worst negative slack and the total negative slack were obtained based on Encounter Digital Implementation System [33].

The total signal net wirelength resulting from the design style with multi-bit TBLB latches is 48% shorter than that resulting from the design style with TBLB macros. Since the design style with TBLB macros compacts all the TBLB latches without considering any physical information of the combinational circuits, the interconnections from TBLB latches to combinational circuits are substantially increased.

The OR-tree latency resulting from the design style with multi-bit TBLB latches is 39% larger than that resulting from the design style with TBLB macros. It is because the design style with TBLB macros has the advantages of integrated TBLB resilient circuits, including all TBLB latches, in each pipeline stage. The OR-tree latency can be minimized due to the compacted layout of TBLB resilient circuits.

The clock wirelength resulting from the design style with multi-bit TBLB latches is 12% larger than that resulting from the design style with TBLB macros. Similar to OR-tree latency, the design style with TBLB macros has the advantages of integrated all TBLB latches. The clock wirelength can be reduced due to much less clock sinks.

The worst negative slack and total negative slack resulting from the design style with multi-bit TBLB latches are 39% and 50% smaller than those resulting from the design style with TBLB macros. Since the design style with TBLB macros may introduce longer signal net wirelength and more critical paths in the combinational circuits among different pipeline stages. The circuit performance may also be degraded.

The runtime resulting from the design style with multi-bit TBLB latches is 10% larger than that resulting from the design style with TBLB macros because the design style with multi-bit TBLB latches resulting from the proposed approach additionally performs TBLB latch clustering, incremental placement, and OR-tree synthesis, which require more sophisticated computations.

To sum up, the design style with multi-bit TBLB latches resulting from the proposed physical synthesis flow and the corresponding algorithms based on the IWLS-2005 benchmark are very effective in reducing the delay of both combinational and error-detection circuits, which indicates better circuit reliability due to circuit aging.

4.2. Comparison of multi-bit pulse-latch generation method and the proposed approach

In the second set of experiment, we compared the proposed approach with the multi-bit pulse-latch generation method [16]. The multi-bit pulse-latch generation method performs global placement followed by multi-bit pulse-latch generation for multi-bit TBLB latch clustering at the post-placement stage.

Table 3 lists the names of the benchmark circuits (Circuit), total signal net wirelength (WL), OR-tree delay (T_{or}), clock wirelength (CWL), worst negative slack (WNS), total negative slack (TNS), and runtime (Time) for the multi-bit pulse-latch generation method [16] and our approach. The clock wirelength was obtained based on [32], while the worst negative slack and the total negative slack were obtained based on [33].

The total signal net wirelength, worst negative slack, and total negative slack resulting from our proposed approach are 9%, 9%, and 8% shorter than those resulting from multi-bit pulse-latch generation method [16], respectively. Since the multi-bit pulse-latch generation method [16] only tries to optimize the number of clock sinks after placement, the total signal net wirelength may not be considered during TBLB latch clustering. In addition, the legalization of multi-bit TBLB latches may hurt the placement result leading to worse signal net wirelength, worst negative slack, and total negative slack.

The OR-tree latency resulting from our proposed approach is 6% shorter than that resulting from multi-bit pulse-latch generation method [16]. It is clear that the proposed approach which considers OR-tree topology can help to reduce OR-tree latency during TBLB latch clustering.

The clock wirelength resulting from our approaches is 3% larger than that resulting from multi-bit pulse-latch generation method [16], and the runtime resulting from our proposed approach is 22% longer than that resulting from multi-bit pulse-latch generation method [16].

Table 3
Comparisons of total signal net wirelength (WL), OR-tree latency (T_{or}), clock wirelength (CWL), worst negative slack (WNS), total negative slack (TNS), and runtime (Time) for the multi-bit pulse-latch generation method [16] and our proposed approach.

Circuit	Multi-bit pulse-latch generation [16]					
	WL × 10^8 (nm)	T_{or} (ns)	CWL × 10^7 (nm)	WNS (nm)	TNS (ns)	Time (s)
ac97ctrl	2.96	0.37	2.41	0.97	269.59	505
aescore	4.55	0.28	2.79	0.95	44.46	119
memctrl	2.91	0.28	1.50	0.92	93.86	185
pcibridge32	6.30	0.47	3.73	0.77	136.54	1561
wbconmax	9.10	0.34	3.15	1.68	302.09	508
Comp.	1.099	1.063	0.974	1.098	1.089	0.772

Circuit	The proposed approach					
	WL × 10^8 (nm)	T_{or} (ns)	CWL × 10^7 (nm)	WNS (ns)	TNS (ns)	Time (s)
ac97ctrl	2.98	0.33	2.38	0.92	266.14	836
aescore	4.15	0.25	2.89	0.86	37.56	127
memctrl	2.67	0.26	1.60	0.84	89.54	231
pcibridge32	4.80	0.46	3.90	0.66	113.26	2395
wbconmax	9.10	0.33	3.18	1.58	303.60	588
Comp.	1	1	1	1	1	1

It is because our approach additionally constructs OR-tree topology with dynamic programming, which require more computation time.

4.3. Comparison of OR-tree-unaware and OR-aware TBLB latch clustering approaches

In the third set of experiment, we compared the proposed approach with and without OR-tree TBLB latch clustering to show the importance of considering OR-tree topology during TBLB latch clustering. The OR-tree-unaware TBLB latch clustering approach is implemented based on the proposed approach without constructing OR-tree topology during OR-tree-latency-aware TBLB latch clustering. In this flow, we only search the shortest TSP tour, which can be treated as the TBLB latch clustering order, during TBLB latch clustering instead of OR-tree construction to minimize the total distance of a TBLB latch chain. In addition, only maximum capacitance loading constraint is considered when grouping TBLB latch during PG-group extraction.

According to the results in Table 4, the total signal net wirelength, worst negative slack, and total negative slack resulting from the OR-tree-aware TBLB latch clustering approach are 13%, 11%, and 9% shorter than those resulting from OR-tree-unaware TBLB latch clustering approach, respectively. Since OR-tree-unaware TBLB latch clustering approach only considers maximum capacitance loading constraint during TBLB latch clustering which may adopt much more multi-bit TBLB latches with larger bit numbers compared with the OR-tree-aware TBLB latch clustering approach, it also leads to more TBLB latch displacement as well as total signal net wirelength compared with the OR-tree-aware TBLB latch clustering approach.

The OR-tree latency resulting from the OR-tree-aware TBLB latch clustering approach is 6% shorter than that resulting from OR-tree-

Table 4
Comparisons of total signal net wirelength (WL), OR-tree latency (T_{or}), clock wirelength (CWL), worst negative slack (WNS), total negative slack (TNS), and runtime (Time) for the OR-tree-unaware and OR-tree-aware TBLB latch clustering approaches.

Circuit	OR-Tree-unaware TBLB latch clustering					
	WL × 10^8 (nm)	T_{or} (ns)	CWL × 10^7 (nm)	WNS (ns)	TNS (ns)	Time (s)
ac97ctrl	3.42	0.37	2.26	1.01	285.33	506
aescore	4.29	0.27	2.86	0.96	45.14	110
memctrl	2.89	0.29	1.62	0.93	91.49	154
pcibridge32	6.83	0.47	4.00	0.81	136.42	1504
wbconmax	9.08	0.33	3.16	1.64	294.07	516
Comp.	1.138	1.062	0.995	1.115	1.094	0.729

Circuit	OR-Tree-aware TBLB latch clustering					
	WL × 10^8 (nm)	T_{or} (ns)	CWL × 10^7 (nm)	WNS (ns)	TNS (ns)	Time (s)
ac97ctrl	2.98	0.33	2.38	0.92	266.14	836
aescore	4.15	0.25	2.89	0.86	37.56	127
memctrl	2.67	0.26	1.60	0.84	89.54	231
pcibridge32	4.80	0.46	3.90	0.66	113.26	2395
wbconmax	9.10	0.33	3.18	1.58	303.60	588
Comp.	1	1	1	1	1	1

unaware TBLB latch clustering approach. It is clear that the proposed approach which considers OR-tree topology can help to reduce OR-tree latency during TBLB latch clustering.

The clock wirelength resulting from both approaches is similar, and the runtime resulting from the OR-tree-aware TBLB latch clustering approach is 27% larger than that resulting from OR-tree-unaware TBLB latch clustering approach because the OR-tree-aware TBLB latch clustering approach additionally constructs OR-tree topology with dynamic programming formulation, which require more computation time.

Consequently, the proposed approach which considers OR-tree topology based on the IWLS-2005 benchmark is very effective in reducing the delay of both combinational and error-detection circuits to improve circuit performance when performing TBLB flip-flop clustering.

5. Conclusions

In this paper, we have introduced the problem of multi-bit TBLB latch replacement for the state-of-the art TBLB resilient circuits. We have also proposed a novel timing-driven multi-bit latch replacement method for low-power TBLB resilient circuits, which simultaneously minimizes the delay of error-detection circuits and that of ordinary data paths. Experimental results based on the IWLS-2005 benchmark have shown that the proposed approach is very effective in reducing the delay of both combinational and error-detection circuits without degrading circuit performance, which indicates better circuit reliability due to circuit aging.

Acknowledgments

We would like to thank Prof. Jinn-Shyan Wang, Prof. Tay-Jyi Lin, and their students from National Chung Cheng University, Taiwan, for the insightful discussions on various design techniques of resilient circuits. This work was partially supported by the Ministry of Science and Technology of Taiwan, under Grant nos. MOST 104-2628-E-194-001-MY3, NSC 102-2220-E-194-006, and NSC 102-2221-E-194-065-MY2, and FY2015 JSPS Invitation Fellowship for Research in Japan.

References

[1] B.H. Calhoun, A.P. Chandrakasan, Standby power reduction using dynamic voltage scaling and canary flip-flop structures, IEEE J. Solid-State Circuits 39 (2004) 1504–1511.
[2] T. Sato, Y. Kunitake, A simple flip-flop circuit for typical-case designs for DFM, in: Proceedings of 2007 IEEE/ACM International Symposium on Quality of Electronic Design, pp. 539–544.
[3] M. Nomura, Y. Ikenaga, K. Takeda, Y. Nakazawa, Y. Aimoto, Y. Hagihara, Delay and power monitoring schemes for minimizing power consumption by means of supply and threshold voltage control in active and standby modes, IEEE J. Solid-State Circuits 41 (2006) 805–814.
[4] X. Wang, M. Tehranipoor, R. Datta, Path-RO: a novel on-chip critical path delay measurement under process variations, in: Proceedings of 2008 IEEE/ACM International Conference on Computer-Aided Design, pp. 640–646.
[5] S. Das, D. Roberts, S. Lee, S. Pant, D. Blaauw, T. Austin, K. Flautner, T. Mudge, A self-tuning DVS processor using delay-error detection and correction, IEEE J. Solid-State Circuits 41 (2006) 792–804.
[6] M. Kurimoto, H. Suzuki, R. Akiyama, T. Yamanaka, H. Ohkuma, H. Takata, H. Shinohara, Phase-adjustable error detection flip-flops with 2-stage hold driven optimization and slack based grouping scheme for dynamic voltage scaling, in: Proceedings of 2008 ACM/IEEE Design Automation Conference, pp. 884–889.
[7] S. Das, C. Tokunaga, S. Pant, W.-H. Ma, S. Kalaiselvan, K. Lai, D.M. Bull, D.T. Blaauw, RazorII: in situ error detection and correction for PVT and SER tolerance, IEEE J. Solid-State Circuits 44 (2009) 32–48.
[8] K.A. Bowman, J.W. Tschanz, N.S. Kim, J.C. Lee, C.B. Wilkerson, S.-L.L. Lu, T. Karnik, V.K. De, Energy-efficient and metastability-immune resilient circuits for dynamic variation tolerance, IEEE J. Solid-State Circuits 44 (2009) 49–63.
[9] M. Kurimoto, H. Suzuki, R. Akiyama, T. Yamanaka, H. Ohkuma, H. Takata, H. Shinohara, Phase-adjustable error detection flip-flops with 2-stage hold-driven optimization, slack-based grouping scheme and slack distribution control for dynamic voltage scaling, ACM Trans. Des. Autom. Electron. Syst. 15 (2010) 17:1–17:17.
[10] K.A. Bowman, J.W. Tschanz, S.-L.L. Lu, P.A. Aseron, M.M. Khellah, A. Raychowdhury, B.M. Geuskens, C. Tokunaga, C.B. Wilkerson, T. Karnik, V.K. De, A 45 nm resilient microprocessor core for dynamic variation tolerance, IEEE J. Solid-State Circuits 46 (2011) 194–208.
[11] J.-S. Wang, K.-J. Chang, T.-J. Lin, R.W. Prasojo, C. Yeh, A 0.36 V, 33.3 uW 18-band ANSI S1.11 1/3 -octave filter bank for digital hearing aids in 40 nm CMOS, in: 2013 IEEE Symposium on VLSI Circuits, pp. C254–C255.
[12] J.-S. Wang, Dynamic Voltage Scaling System having Time Borrowing and Local Boosting Capability, US Patent No. 8,933,726, 2014.
[13] Y.-M. Yang, I.H.-R. Jiang, S.-T. Ho, PushPull: short-path padding for timing error resilient circuits, IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst. 33 (2014) 558–570.
[14] W. Wang, S. Yang, S. Bhardwaj, R. Vattikonda, S.B. K. Vrudhula, F. Liu, Y. Cao, The impact of NBTI on the performance of combinational and sequential circuits, in: Proceedings of 2007 ACM/IEEE Design Automation Conference, pp. 364–369.
[15] ISCAS89 Benchmarks, ⟨https://people.engr.ncsu.edu/brglez/CBL/benchmarks/ISCAS89/⟩
[16] C.-L. Chang, I.H.-R. Jiang, Pulsed-latch replacement using concurrent time borrowing and clock gating, IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst. 32 (2013) 242–246.
[17] Y.-T. Chang, C.-C. Hsu, M.P.-H. Lin, Y.-W. Tsai, S.-F. Chen, Post-placement power optimization with multi-bit flip-flops, in: Proceedings of 2010 IEEE/ACM International Conference on Computer-Aided Design, pp. 218–223.
[18] M.P.-H. Lin, C.-C. Hsu, Y.-T. Chang, Post-placement power optimization with multi-bit flip-flops, IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst. 30 (2011) 1870–1882.
[19] S.-H. Wang, Y.-Y. Liang, T.-Y. Kuo, W.-K. Mak, Power-driven flip-flop merging and relocation, IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst. 31 (2012) 180–191.
[20] I.H.-R. Jiang, C.-L. Chang, Y.-M. Yang, INTEGRA: fast multi-bit flip-flop clustering for clock power saving, IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst. 31 (2012) 192–204.
[21] Y.-T. Shyu, J.-M. Lin, C.-P. Huang, C.-W. Lin, Y.-Z. Lin, S.-J. Chang, Effective and efficient approach for power reduction by using multi-bit flip-flops, IEEE Trans. Very Large Scale Integr. Syst. 21 (2013) 624–635.
[22] S.-Y.S. Liu, W.-T. Lo, C.-J. Lee, H.-M. Chen, Agglomerative-based flip-flop merging and relocation for signal wirelength and clock tree optimization, ACM Trans. Des. Autom. Electron. Syst. 18 (2013) 40.
[23] Z.-W. Chen, J.-T. Yan, Routability-constrained multi-bit flip-flop construction for clock power reduction, Integration, VLSI J. 46 (2013) 290–300.
[24] M.P.-H. Lin, C.-C. Hsu, Y.-C. Chen, Clock-tree aware multibit flip-flop generation during placement for power optimization, IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst. 34 (2015) 280–292.
[25] C.-C. Hsu, M.P. Lin, Y.-T. Chang, Crosstalk-aware multi-bit flip-flop generation for power optimization, Integration, VLSI J. 48 (2015) 146–157.
[26] C. Xu, P. Li, G. Luo, Y. Shi, I.H.-R. Jiang, Analytical clustering score with application to post-placement multi-bit flip-flop merging, in: Proceedings of 2015 ACM International Symposium on Physical Design, pp. 93–100.
[27] T.-C. Chen, Z.-W. Jiang, T.-C. Hsu, H.-C. Chen, Y.-W. Chang, NTUplace3: an analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints, IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst. 27 (2008) 1228–1240.
[28] K. Helsgaun, General k-opt submoves for the Lin–Kernighan TSP heuristic, Math. Program. Comput. 1 (2009) 119–163.
[29] OpenCores, ⟨http://www.opencores.org⟩, 2016.
[30] IWLS 2005 Benchmarks, ⟨http://iwls.org/iwls2005/benchmarks.html⟩, 2005.
[31] Nangate 45 nm Open Cell Library, ⟨http://www.nangate.com⟩, 2016.
[32] D.J.-H. Huang, A.B. Kahng, C.-W.A. Tsao, On the bounded-skew clock and Steiner routing problems, in: Proceedings of 1995 ACM/IEEE Design Automation Conference, pp. 508–513.
[33] Encounter Digital Implementation System, Cadence, Inc., ⟨http://www.cadence.com⟩, 2016.