Going Cooler With Timing-Constrained TeSHoP: A Temperature Sensing-Based Hotspot-Driven Placement Technique for FPGAs

Weina Lu, Yu Hu, *Member, IEEE*, Jing Ye, *Member, IEEE*, and Xiaowei Li, *Senior Member, IEEE*

***Abstract*— The continuous shrinking of the feature size in CMOS technology has significantly increased the power densities of integrated circuits, leading to severe temperature issues. However, the previous offline simulation-based thermal optimization works cast large deviations with the reality, while online sensing-based thermal managements usually incur significant performance overhead. Therefore, it is crucial to propose a method that could achieve fine-grained optimization with accurate temperature profiles. In this paper, we propose a timing-constraint temperature sensing-based hotspot-driven placement technique for field-programmable gate arrays (FPGAs). The hotspot optimization issue is modeled as a hyper minimum bipartite matching problem and is solved by a place adjustment with the input of an online sensed temperature profile. We propose an open-source/commercial hybrid design flow to implement the whole optimization in Xilinx Virtex-6 FPGA. Experimental results demonstrate a significant reduction in peak temperature and a great improvement on thermal uniformity, with slight performance overhead under timing constraints.**

***Index Terms*— Computer-aided design flow, field-programmable gate arrays (FPGAs), hotspot optimization, performance.**

I. Introduction

FIELD-PROGRAMMABLE GATE ARRAY (FPGA) devices have been expanding their fields in many commercial applications for high performance and rapid hardware implementation. However, along with transistor technology shrinking at a high pace, the power density of FPGAs increased rapidly and caused the thermal issue to be one of the severest challenges on reliability and performance. It is reported that the failure rate doubles with every 10 °C rises in temperature [1] and thus, the appearance of hotspots could significantly degrade the reliability. On the other hand, excessive temperatures could also increase the delay of circuits and exert a tight constraint on the performance budget. Therefore, the hotspots have become an increasingly pressing concern for FPGAs.

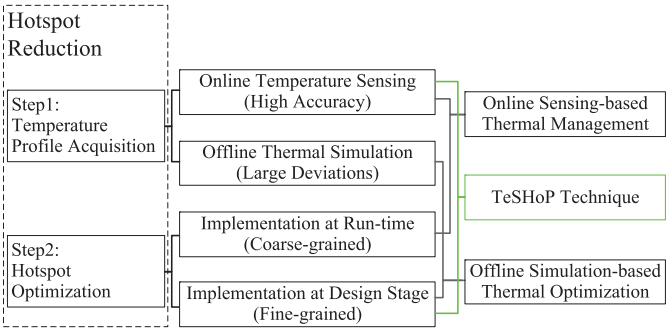


Fig. 1. Key steps and methods of hotspot reduction.

The key reason behind hotspots is that the junction temperature increases with the dramatic growth of power density, while it in turn exponentially aggravates the leakage power as well [2]. This mutual effect gives rise to a positive feedback loop. Especially when circuits are designed in clusters, which leads to inadequate heat emission on FPGAs, this temperature issue will cause a partial thermal explosion [3] and make hotspots come into being.

Methods for hotspot reduction mainly contain two steps: temperature profile acquisition and hotspot optimization, as seen in Fig. 1. The temperature profile of the FPGA describes the hotspot distribution of the target circuit on chip, which is used as a reference to guide hotspot optimization. Temperature profile acquisition techniques can be broadly classified into online temperature sensing and offline thermal simulation, while hotspot optimization could be implemented at either run-time or the design stage.

Existing works mainly have two kinds of hotspot reduction methods: the online sensing-based thermal management and the offline simulation-based thermal optimization.

The online sensing-based thermal management methods [4]–[9] dynamically monitor and give response to temperature violations. To acquire accurate temperature profile, they use programmable thermal sensors in their reactive systems for online sensing. The hotspot optimizations are implemented at run-time, including dynamic voltage/frequency scaling [8], [9] and task migration [5], [6]. The optimization of these methods is coarse-grained, for it takes the module as the minimum unit, which is composed of a large number of configurable

Manuscript received December 17, 2016; revised April 13, 2017; accepted May 18, 2017. Date of publication June 13, 2017; date of current version August 23, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61274030, Grant 61532017, Grant 61376043, and Grant 61521092. *(Corresponding author: Yu Hu.)*

W. Lu, Y. Hu, and X. Li are with the State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China and also with the Graduate University of Chinese Academy of Sciences, Beijing 100080, China (e-mail: luweina@ict.ac.cn; huyu@ict.ac.cn; lxw@ict.ac.cn).

J. Ye is with the State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: yejing@ict.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2017.2707120

logic blocks (CLBs). Coarse-grained optimizations give rise to the significant performance overhead, such as 142.86% in [7]. Not merely the execution time of the controller but also the scaling of voltage/frequency will increase the latency and further degrade the performance. Therefore, it is important to perform fine-grained optimization to reduce hotspots and simultaneously consider protecting the performance and the area from being disturbed.

The offline simulation-based thermal optimization methods [10]–[16] are implemented in the design flow. Their temperature profile is acquired by power estimation and thermal simulation. The simulation models are usually established by the analysis of temperature and power, such as the finite difference method-based model [10], [12], [14] and the electrostatic-based model [15]. The hotspot optimizations are usually implemented by treating the thermal issue as one of the optimization objectives at the design stage. These optimization methods have the superiority of fine granularity. But the simulation-based temperature profile always casts large deviations from the reality and makes them less effective. Amrouch *et al.* [17] have studied the accuracy of simulation-based methods and reached a conclusion that the difference between simulation and actual temperature is up to 13 °C.

In contrast with the conventional methods presented above, the temperature sensing-based hotspot-driven placement (TeSHoP) technique proposed in this paper obtains the temperature profile by online temperature sensing, and the hotspot optimization is implemented at the design stage to achieve a fine-grained CLB-level optimization. To implement the TeSHoP technique in real-life scenarios and evaluate the optimizations on a real FPGA, there are several challenges to be overcome. First, to implement a general online temperature sensing method, we need to design a flexible sensor network which is convenient in inserting to various circuits. Second, to keep the performance from being disturbed as much as possible, the proposed method for hotspot optimization needs to have the capability of reducing hotspots, meanwhile, effectively protecting the performance. Furthermore, to implement the timing-constrained optimizations on a real FPGA, we need to pioneer ways between an open-source and a commercial tool to complete an open-source/commercial hybrid design flow which supports the commercial timing constraint.

With overcoming all the barriers presented above, the key contributions of this paper are as follows.

1) We propose a flexible sensing network to obtain the online temperature profile for design optimization, which reflects a much more accurate temperature distribution of the FPGA than the profile generated by simulation.
2) We propose a high-level formulation of the hotspot-driven placement problem with the hyper minimum bipartite matching model.
3) We propose an efficient heuristic algorithm for solving the problem, which is capable of keeping the performance from being disturbed as much as possible during the hotspot optimization.
4) We propose an open-source/commercial hybrid design flow to implement the hotspot reduction. The proposed design flow supports the commonly used physical timing constraints of the commercial tool. All the experiments are implemented in a real FPGA through this flow.

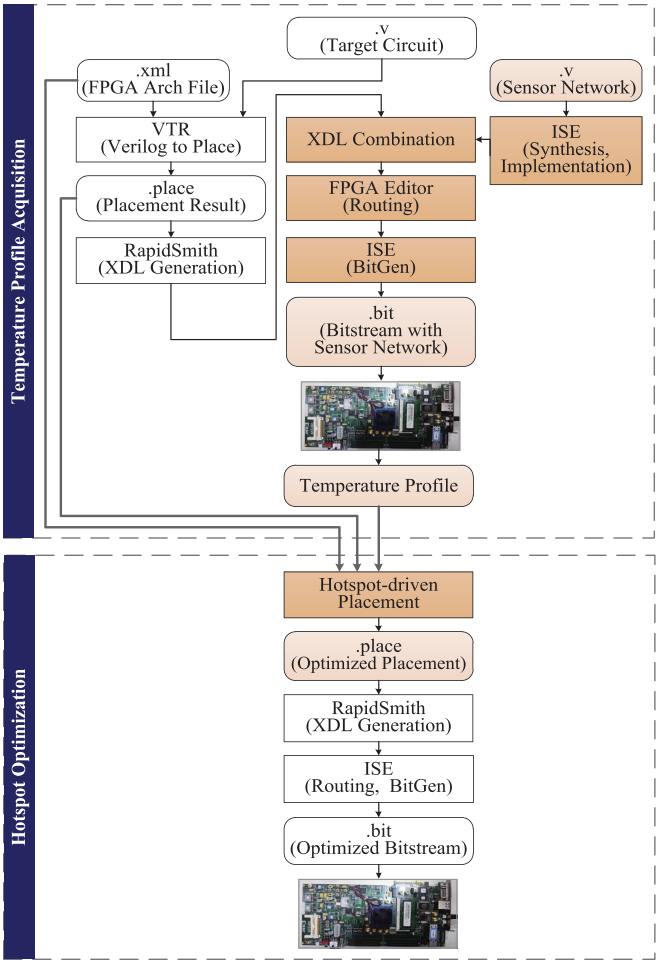


Fig. 2. TeSHoP framework overview.

The remainder of this paper is organized as follows. Section II depicts the framework overview of our TeSHoP technique. Method for temperature profile acquisition is shown in Section III. The strategy of hotspot optimization is detailed in Section IV. The experimental results are analyzed in Section V. We conclude and present future work in Section VI.

II. TeSHoP Framework Overview

To implement the TeSHoP technique in real-life scenarios, we come up with an open-source/commercial hybrid design flow, as shown in Fig. 2. The design flow of TeSHoP is mainly composed of temperature profile acquisition and hotspot optimization. The temperature profile is obtained by online sensing method. This part mainly contains the original area-and-timing-aware implementation of the target circuit, the insertion of the sensor network, and the online temperature sensing for the FPGA. It provides the original placement result and the temperature profile for hotspot optimization. With this original placement result as the start point, the hotspot optimization adjusts the placement according to the temperature profile and reduces hotspots under a specific upper bound of area

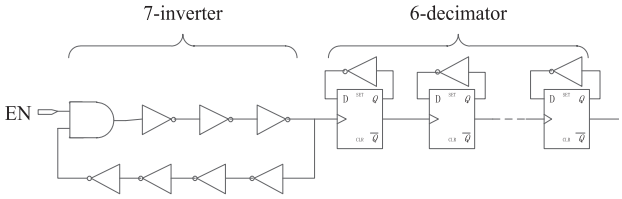


Fig. 3. RO-based sensor structure.

and timing overhead. The bitstream is then generated for the hotspot-optimized design to run on real FPGAs.

The highlighted orange parts in Fig. 2 are the key implementation of our TeSHoP technique. The temperature sensing network is implemented by integral of squared error (ISE), a commercial design suit of Xilinx, to achieve a flexible and region-restricted network. The hotspot-driven placement is implemented in VTR, an open-source Verilog-to-Routing design tool [18]. With the extension of VTR-to-bitstream design flow [19], the placement result of VTR is converted to the ISE for routing, bitstream generation and finally run on real FPGAs.

III. Temperature Profile Acquisition

In this section, we first design and implement a flexible programmable temperature sensor network for online measurement. Then, we insert the sensor network into the circuit during the circuit design flow.

A. Design of Temperature Sensor Network

The programmable temperature sensor network contains a series of ring oscillator-based (RO) sensors, a counter, and a sensor controller.

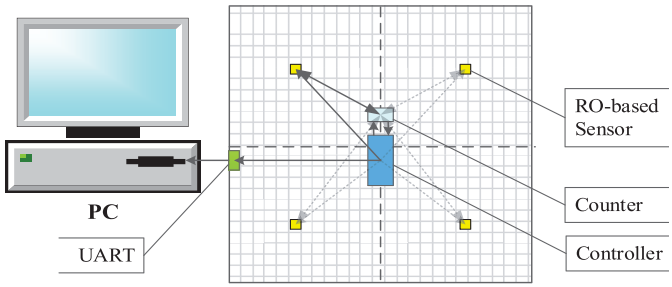
The RO has an odd number of inverters and the frequency is computed as in

$$f_{\text{ro}} = \frac{1}{2N \cdot t_d} \tag{1}$$

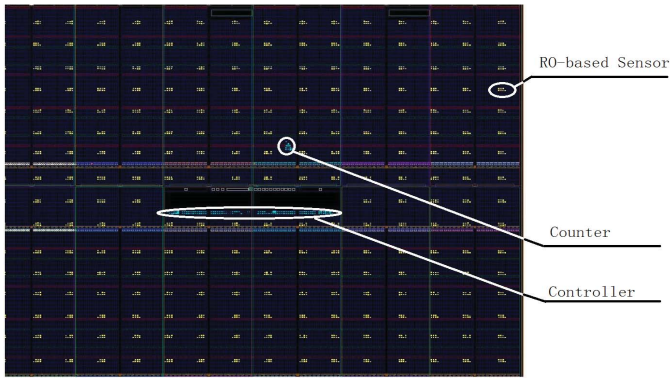
where N is the number of inverters in an RO, t_d is the delay of the inverter. Its frequency is inversely proportional to the temperature [20]. Owing to its flexibility and conciseness, RO is widely adopted for temperature measurement in various applications of integrated circuits.

We design our RO-based sensor on the foundation of [21]. As shown in Fig. 3, it contains 7 inverters for oscillation and 6 decimators for the oscillating frequency to be captured by the clock under Nyquist sampling theory. To improve the accuracy of measurement, the sensor network is calibrated from 25 °C to 95 °C with the aid of the system monitor [22]. The measurement accuracy of this method is proven to be within 0.5 °C [21] and is accurate enough for our technique.

The RO-based sensor network is distributed in arrays and the sensor controller is located in the center of the chip. We divide the FPGA into $r_1 \times r_2$ grids and put an RO-based sensor in the center of each grid. The temperature of all the logic blocks in a grid is treated with the same value which is measured by the sensor in that grid. Fig. 4(a) gives out a simple



(a) A Simple 2 × 2 Grid Structure



(b) Floorplan of a Sensor Network

Fig. 4. Temperature sensor network on an FPGA. (a) Simple 2 × 2 grid structure. (b) Floorplan of a sensor network.

example of the component structure in a 2 × 2 grid partition and Fig. 4(b) is the physical floorplan of a 10 × 10 sensor network from a real FPGA device. The blocks highlighted in yellow are the RO-based sensors and the blocks in blue are the controller and counter, respectively.

Since the RO-based sensor oscillates at a high frequency which could generate heat as well, it would disturb the online temperature profile if the RO-based sensor is in the working state for a long time. Therefore, the sensor network is initially set to be idle. When the temperature sensing is required, the sensor controller activates and shuts off the RO-based sensors in pipeline, as shown in Fig. 5. For each RO-based sensor, when it is activated by the controller, it takes 2^{16} cycles for stabilization because the oscillation of the RO may not be stable at the beginning. Then it takes 2^{16} cycles for counting the oscillation, which reflects the frequency of this RO. Meanwhile, the next RO-based sensor is activated for stabilization and prepares for counting. Each RO-based sensor is immediately set back to the idle state after the counting to keep the influence induced by the sensor network to be negligible. The counter module keeps counting for each RO-based sensor and sequentially sends the frequency value back to the controller. All the frequency value would be sent back to the PC through the UART port. And the temperature profile is then generated by the frequency of each RO-based sensor.

With the online temperature profile, we can detect the hotspots for optimization. Apparently, the accuracy of the sensor network increases along with the number of grids rising. However, overuse of sensors not only requires more

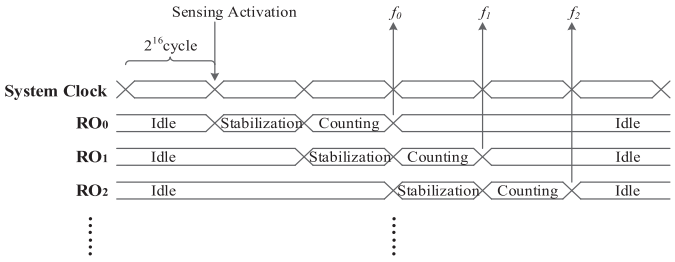


Fig. 5. Working state of ROs in the sensor network.

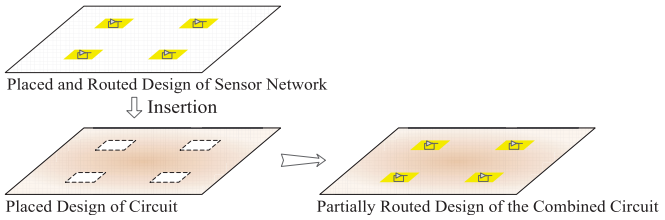


Fig. 6. XDL Combination flow for the sensor insertion.

resource utilization for sensing, but also the heat produced by oscillating can no longer be neglected as well. This effect will in turn influence the accuracy of the sensor network and further degrade the optimization. Taking the adopted FPGA as an example, we discuss different grid partitions which are detailed in Section V.

All the components of the sensor network are constrained to the specific regions and the normalized resource utilization for sensing is less than 2% under the most reasonable grid partition, which is discussed in Section V. Notably, this resource requirement only takes place when the sensor network is in need for obtaining the temperature profile. In the later hotspot optimization, the sensor network is no longer needed and will not be contained in the final circuit design. Therefore, no area overhead will be brought into the circuit design after using the TeSHoP technique.

B. Insertion of Sensor Network

Existing works on programmable sensor-based managements mainly insert the sensor in their Verilog Hardware Description Language. Since the calibrated function between temperature and frequency varies with different routing structure, the sensors need to be recalibrated for different implementations. In contrast, our TeSHoP technique proposes a general method where the sensor network only needs to be calibrated once and then applicable for different circuits thereafter. The process of the insertion is as follows.

1) Calibration (Only Once): For a given FPGA architecture, we design a suitable $r_1 \times r_2$ grid partition for the sensor network and implement it with a commercial tool to achieve a region-restricted implementation. Then the calibration of the sensor network is executed. Afterwards, the calibrated sensor network could be used for different circuits for temperature sensing without any recalibration. This step is required only when a different FPGA architecture is applied for the first time.

2) Insertion: During the design flow of the target circuit, we insert the post-route design of the calibrated sensor network which contains the complete place and route information into the circuit. Specifically, to realize it in Xilinx FPGAs, we adopt the human readable Xilinx Design Language (XDL) and propose an $XDL\_Combine$ tool for the design combination. Since the segmentation information is hard to be extracted from a commercial FPGA and routing conflict is not avoidable for two individual routing designs, we combine the post-place file (.xdl) of the circuit and the post-route file (.xdl) of the sensor network into a partial-routed design through our tool, which could ensure them sharing the same system signal, such as the system clock, and avoiding placement conflict, as shown in Fig. 6.

3) Routing: On account that the commonly used *par* routing tool in the ISE is entitled to change the existing routing information and disturb the routed structure of the RO-based sensors, we adopt the $FPGA\ Editor$ tool to automatically route the unrouted nets [23] and meanwhile, ensure the routed ROs untouched after the insertion. While for the design after the TeSHoP technique, since it is unnecessary to insert the sensor network to the circuit for temperature sensing, we adopt the *par* tool in the ISE for routing, which supports the physical timing constraint by adding them to the pcf file and make it a way for the circuit meeting their timing requirements. The fully routed design is finally sent to the $bitgen$ tool in the ISE for subsequent bitstream generation.

Therefore, the original structure of the RO will not be touched and the RO calibration will not be required for different circuits anymore. In addition, the insertion of the sensor network has slight effect on the original circuit placement because the resource utilization in most applications is within 80%, which means there would always be at least 20% resource left unused at run-time. Since the resource utilization of our flexible sensor network is less than 2%, the unused resource is sufficient for inserting a sensor network. In addition, since the sensor network is no longer needed after the hotspot optimization, no area and power influence will be exerted on the final optimized circuit design.

IV. Hotspot Optimization

SRAM-based FPGAs are the island-style structure with an array of CLBs, some heterogeneous blocks between them, input–output blocks on the periphery and routing segments in the channels [24]. All the components are arranged in arrays and the task of placement is to work out an appropriate match between the circuit blocks and the available components, such as the CLBs on the FPGA. As can be seen, the TeSHoP technique is implemented by taking hotspot reduction as one of the optimization objectives during this process.

In this section, we first give out a high-level problem formulation of the TeSHoP. Then we detail the hotspot-driven algorithm flow for solving the problem.

A. Problem Formulation

For an arbitrary circuit, assume it contains m circuit blocks to be placed and there are n ($n \geq m$) logic blocks available

in the target FPGA. Then we model the circuit blocks as a set $B = \{b_0, b_1, \ldots, b_{n-1}\}$ and model the available CLBs, taking CLBs as an example, as a set $C = \{c_0, c_1, \ldots c_{n-1}\}$. The attributes of set B are as follows.

1) m nodes of B are corresponding to the circuit blocks while the rest $n - m$ nodes are defined as $NULL$ and are mutually equivalent.
2) Each b_i can map with one and only one node c_j in set C.
3) Each b_i has a specific temperature extracted from the temperature profile.

And for set C, we have the following.

1) Each c_j is corresponding to a unique coordinate $p(x_j, y_j)$ on the FPGA.
2) Each c_j can map with one and only one node b_i.

All the possible mapping relations between sets B and C are modeled as an edge set E, where $E = \{e_{ij}, i \subset B, j \subset C\}$. For an arbitrary circuit block b_i, different e_{ij} indicates that the b_i is placed in a different position $p(x_j, y_j)$ of the FPGA. For example, edges of b_5 in Fig. 7(a) and (b) are $e_{5,5}$ and $e_{5,10}$ which means b_5 is placed in c_5 and c_{10} is surrounded by different blocks, respectively. We define the terms as follows.

Definition: The range of thermal effect of a hotspot b_i is a square that is centered on b_i. The vertical/horizontal radius of the square is equal to a nonnegative integer $r (r \geq 0)$. The *neighbors* of e_{ij} is the set of blocks N who are in the range of thermal effect with b_i when b_i is mapping to c_j. Taking Fig. 7(a) as an example, the range of thermal effect is $r = 1$ and the *neighbors* of $e_{5,5}$ is $N = \{b_0, b_1, b_2, b_4, b_6, b_8, b_9, b_{10}\}$. Apparently, there are $(2r + 1)^2 - 1$ *neighbors* for an edge.

The w_{ij} of set $W = \{w_{ij}, i \subset B, j \subset C\}$ denotes the thermal weight for the edge e_{ij} in E, which represents the potential temperature when b_i maps to c_j. As depicted in (2), weight w_{ij} is composed of two parts, the temperature value of b_i itself and the thermal effect that its *neighbors* exert on it. The thermal propagation of hotspot is set to be $(1/d^3)$ to conform to the propagation of point heat source [25]

$$w_{ij} = T_i + \sum_{k=0}^{(2r+1)^2-2} \frac{T_{N(k)} - T_i}{d^3_{(c_{N(k)}, c_j)}} \tag{2}$$

where T_i and $T_{N(k)}$ are the temperature of b_i and its *neighbor* $b_{N(k)}$, respectively. The $d_{(c_{N(k)}, c_j)}$ is the distance between c_j and $c_{N(k)}$, as shown in

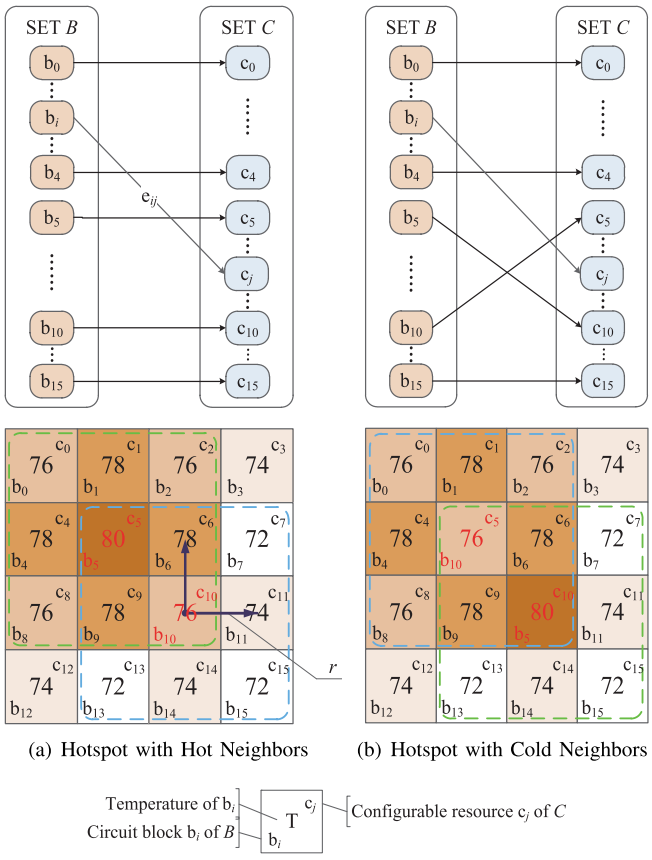
$$d_{(c_{N(k)}, c_j)} = \begin{cases} \sqrt{(\Delta x + 1)^2 + (\Delta y + 1)^2}, & \Delta x \cdot \Delta y \neq 0 \\ |\ \Delta x\ | + |\ \Delta y\ | + 1, & \Delta x \cdot \Delta y = 0. \end{cases} \tag{3}$$

The distance of c_j to itself is defined as 1 for solving the problem that the denominator cannot be 0. And the coordinates are concomitantly increased by 1 for its neighbors. For example, the weight of $e_{5,5}$ in Fig. 7(a) is $w_{5,5} = 80 + 4 \times ((78 - 80)/2^3) + 4 \times ((76 - 80)/(2\sqrt{2})^3) = 78.3$.

The hotspot-driven placement problem can be described as a hyper weighted bipartite graph $G = (V, E, W)$, $V = B \cup C$. The objective of our TeSHoP technique is to find a minimum weight match in these perfect match for graph G.

Traditional weighted bipartite graph has a fixed weight for each edge thus finding minimum weight match for the graph is known as the assignment problem, which could be solved in polynomial time. However, G is equivalent to the traditional weighted bipartite graph only when the range of thermal effect $r = 0$. As in our problem $r \geq 1$, weights in W are dynamically changed with every perfect match of G. Taking Fig. 7 as an example, Fig. 7(a) and (b) demonstrates two scenarios where only b_5 and b_{10} have different edge. The temperature is enumerated in the center of each block. The range of thermal effect is assumed to be $r = 1$. Therefore, the weights of $e_{5,5}$, $e_{10,10}$, and $e_{4,4}$ in Fig. 7(a) are $w_{5,5} = 78.3$, $w_{10,10} = 75.7$, and $w_{4,4} = 77.8$, respectively. As for Fig. 7(b), since the edge of b_5 and b_{10} has changed, weights of b_5, b_{10} and their *neighbors* such as b_4, b_{11} are affected simultaneously. The weights of $e_{5,10}$, $e_{10,5}$, and an example *neighbor* $e_{4,4}$ in Fig. 7(b) are $w'_{5,10} = 76.8$, $w'_{10,5} = 77.2$, $w'_{4,4} = 77.3$, respectively. Apparently, any changing in edges would give rise to different weight w_{ij} even for the unchanged edges, which increases the complexity of the problem. Therefore, our TeSHoP technique adopts a heuristic algorithm to solve it by finding an optimal perfect match that approaches to the minimum weight match in graph G.

Conventional area/timing-driven placement algorithms would locate the blocks as close as possible in order to minimize the area or timing cost. However, hotspots in



NOTE: A square in a sub-figure represents a circuit block of set B. The block distribution of a sub-figure represents the placement implementation of the corresponding perfect match.

Fig. 7. Hyper minimum weight match process of TeSHoP. (a) Hotspot with hot neighbors. (b) Hotspot with cold neighbors.

clusters would aggravate the positive feedback loop of temperature and power, and lead to excessive damage. Moreover, nonuniformity of on-chip temperature would cause many other problems such as reliability and aging [26]. Thus during the placement process, we try to decrease the peak temperature and improve the thermal uniformity. Lower peak temperature can directly mitigate hotspots on the present temperature profile, while better thermal uniformity indicates a more reasonable thermal distribution and benefits system health as well.

Therefore, for the hyper weighted bipartite graph G, the maximum and minimum weight in W, denoted as $w_{\max}$ and $w_{\min}$, are much meaningful to us. For example, comparing with Fig. 7(a), the highest weight induced by the hotspot b_5 is significantly decreased from 78.3 to 77.8 in Fig. 7(b), which makes this hotspot surrounded by a larger number of colder neighbors. The highest thermal gap is decreased from 5.6 to 4.9 as well. Apparently, mapping with lower weight is more conductive to heat emission and thus it is more preferred by hotspot reduction.

We take $w_{\max}$ as the evaluation metric for the potential peak temperature of this perfect match and take $w_{\max} - w_{\min}$ as the evaluation metric for the potential highest thermal gap. Hence, the goal of the hotspot-driven placement is to minimize these two values as much as possible.

1) *Given:* A set of circuit blocks B, a set of FPGA available logic blocks C, a temperature profile of the original circuit design.
2) *Tasks:* Finding a perfect match that approaches to the minimum weight match of the hyper weighted bipartite graph G.
3) *Objective:* Reducing the hotspots by minimizing $w_{\max}$ and $w_{\max} - w_{\min}$.

B. Problem Solving

We propose an improved simulated annealing (ISA) algorithm for solving the problem, as depicted in Fig. 8.

The initial perfect match of graph G is established from the given original placement file and the temperature of each circuit block in B is extracted from the temperature profile. Then the optimal perfect match is obtained by iteratively exchanging the edges of G and evaluating the adjustment. The hotspot is taken as one of the optimization objectives for evaluation during this process. With multitudes of iterations, the optimal perfect match with as few hotspots as possible is worked out when the annealing temperature T_{SA} or the iteration reaches the threshold.

The given original placement file is an area-and-timing optimized placement result. Thus during the process of the hotspot-driven placement, the selected edge pair should satisfy the following prerequisites: (Prerequisite 1) being capable of effectively reducing hotspots; (Prerequisite 2) not disturbing the optimized area and timing as possible as it could.

Prerequisite 1 (Increasing the Efficiency of Hotspot Reduction): For a perfect match of G, it is unexpected if hotspots in B map to the positions of C in clusters which would increase the peak temperature $w_{\max}$ and the temperature gap $w_{\max} - w_{\min}$. Thus, operations of exchanging the edge between hot and cold blocks which indicate remapping hot blocks to positions with cold *neighbors* are preferred in our algorithm. We define a temperature difference lower bound δ as one of the metrics for choosing suitable pairs of edges in G for adjustment. For a pair of blocks $\{b_i, b_j\}$ whose temperature are $\{T_i, T_j\}$, it could be taken as a pair of candidate for edge exchange unless it satisfies

$$| T_i - T_j | \geq \delta \tag{4}$$

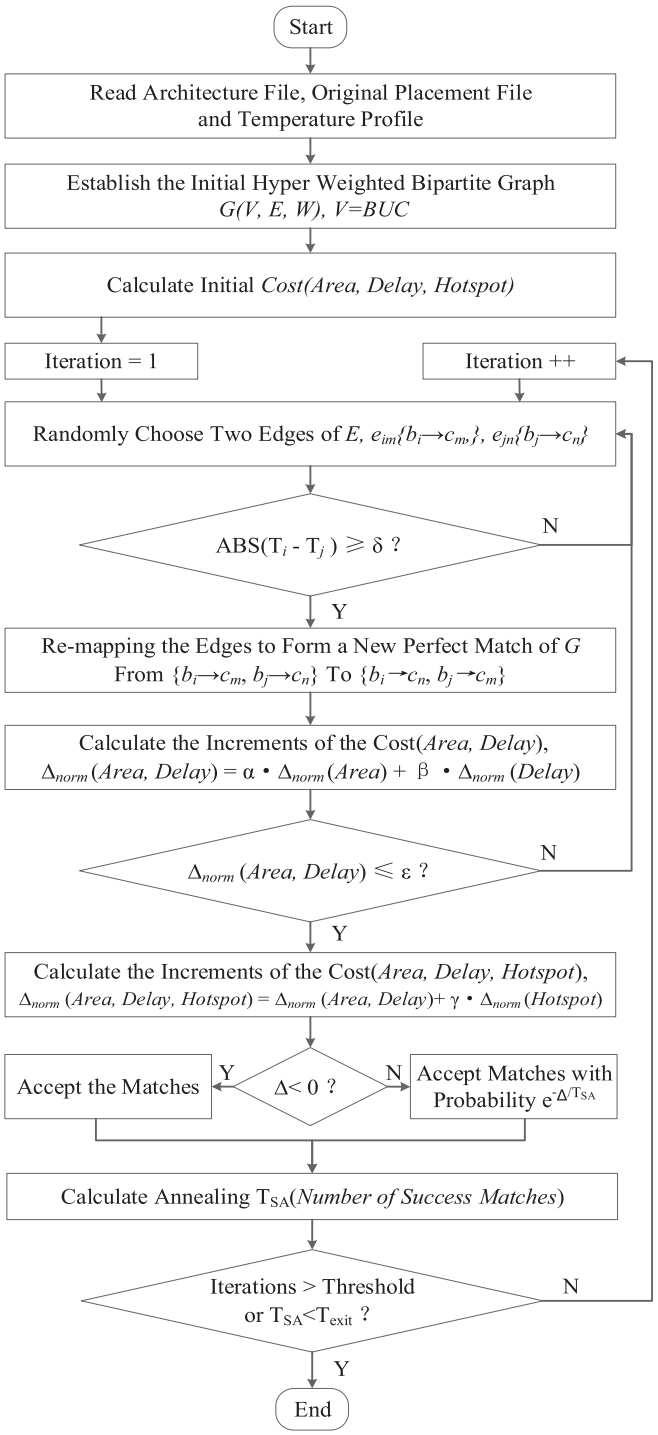


Fig. 8. TeSHoP algorithm flow.

where the positive real number δ is the user-defined lower bound of the temperature difference.

Prerequisite 2 (Keeping the Optimized Area and Timing From Being Disturbed as Much as Possible): The area and timing are intuitively considered to be taken as part of the optimization objectives. However, even though the cost of area, timing, and hotspot are jointly taken for optimization, the optimized area and timing would still be disturbed because the start point of the placement is an area-and-timing optimized result where the normalized cost of area and timing is magnitudes lower than that of unoptimized hotspot at the beginning of the optimization. Even with the premier work of parameters trading off, the disturbance on area and timing would still be masked by the significant improvement of hotspot. In consequence, the area and timing are disturbed and need a long time to converge to a new optimized result along with the hotspot optimization which degrades the efficiency.

Therefore, to decrease the disturbance on the optimized area and timing, we define an overhead threshold ε for the area and timing cost. For a pair of candidate blocks $\{b_i, b_j\}$, the edge exchange would be considered only when the area and timing degradation induced by this change is under the overhead threshold, as in

$$\alpha \cdot \Delta_{\text{norm}}(\text{Area}) + \beta \cdot \Delta_{\text{norm}}(\text{Delay}) \leq \varepsilon \tag{5}$$

where α, $\beta \epsilon (0, 1)$ are the tradeoff parameter of area and timing cost, respectively. $\varepsilon \epsilon [0, 1]$ is the user-defined overhead threshold. The calculation of the normalized increment of area and timing cost is shown in

$$\Delta_{\text{norm}}(\text{Area}) = \frac{\text{cost(Area)} - \text{cost}_{\text{prev}}(\text{Area})}{\text{cost}_{\text{prev}}(\text{Area})} \tag{6}$$

$$\Delta_{\text{norm}}(\text{Delay}) = \frac{\text{cost(Delay)} - \text{cost}_{\text{prev}}(\text{Delay})}{\text{cost}_{\text{prev}}(\text{Delay})} \tag{7}$$

$$\text{cost(Area)} = \sum_{n=1}^{N_{\text{nets}}} q(n) \left[\frac{bb_x(n)}{C_{av,x}(n)} + \frac{bb_y(n)}{C_{av,y}(n)} \right]. \tag{8}$$

The cost of area cost (Area) and timing cost (Delay) follow the computation in the versatile placement and routing (VPR) tool of VTR [27], [28]. The area cost (Area) is computed by the linear congestion function as shown in (8) [27]. It is the sum of all the fan-out which is denoted as N_{nets} in the circuit. For each fan-out n, bb_x, and bb_y denote the horizontal and vertical spans of its bounding box, respectively. The $q(n)$ is the compensation factor. Cav, $x(n)$ and Cav, $y(n)$ are the average channel capacities in the x and y directions over the bounding box of fan-out n. The timing cost (Delay) is computed by the timing model in [28]. It is computed with the RC parameters extracted from the architecture file.

For an arbitrary pair of blocks, on condition that both (4) and (5) are satisfied, the cost of the hotspot would then be calculated and the overall cost would finally be worked out for assessing the adjustment. The hotspot cost [cost(Hotspot)] is defined as follows:

$$\text{cost(Hotspot)} = \text{w}_{\max} \times (\text{w}_{\max} - \text{w}_{\min}). \tag{9}$$

$$T = \begin{bmatrix} \text{T}_0 \\ \vdots \\ \text{T}_i \\ \vdots \\ \text{T}_{n-1} \end{bmatrix} \quad M(\Delta T) = \begin{bmatrix} \text{T}_{(N0(0),0)} & \text{T}_{(N0(1),0)} & \cdots & \text{T}_{(N0((2r+1)^2-2),0)} \\ \vdots & \vdots & & \vdots \\ \text{T}_{(Ni(0),i)} & \text{T}_{(Ni(1),i)} & \cdots & \text{T}_{(Ni((2r+1)^2-2),i)} \\ \vdots & \vdots & & \vdots \\ \text{T}_{(Nn\text{-}1(0),n\text{-}1)} & \text{T}_{(Nn\text{-}1(1),n\text{-}1)} & \cdots & \text{T}_{(Nn\text{-}1((2r+1)^2-2),n\text{-}1)} \end{bmatrix}$$

$$D = \begin{bmatrix} \frac{1}{d_0^3} & \frac{1}{d_1^3} & \cdots\cdots & \frac{1}{d^3_{(2r+1)^2-2}} \end{bmatrix}$$

Fig. 9. Matrix T, $M(\Delta T)$, and D.

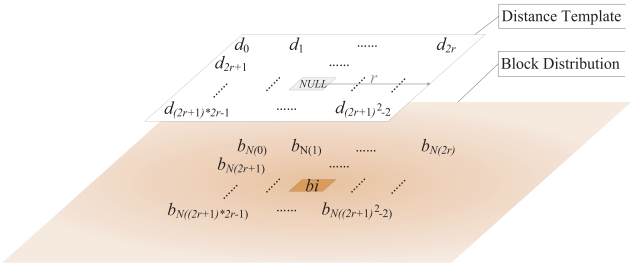


Fig. 10. Distance template.

And the overall cost function for hotspot optimization is

$$\text{cost} = \alpha \cdot \text{cost(Area)} + \beta \cdot \text{cost(Delay)} + \gamma \cdot \text{cost(Hotspot)} \tag{10}$$

where α, β are the same tradeoff parameter as in (5), $\gamma \epsilon (0, 1)$ is the tradeoff parameter of hotspot. The three parameters satisfy $\alpha + \beta + \gamma = 1$.

During the iteration process of ISA, the weight in W is intuitively supposed to be recalculated because weights are dynamically changed with different perfect match of G. Since there are n edges in a perfect match and $(2r+1)^2$ computation items in a weight, the time complexity of calculating W is $O(n \times r^2)$, which is time-consuming. Therefore, we parse the weights of W to a temperature matrix T, a temperature difference matrix $M(\Delta T)$ for each perfect match, and a distance template D for a specific r, as shown in Fig. 9. The $n \times 1$ matrix T lists the temperature of each b_i in order which corresponds to the first computation item in w_{ij}. The $M(\Delta T)$ is an $n \times [(2r+1)^2 - 1]$ matrix, where each row i lists the temperature difference of the rest $(2r+1)^2 - 1$ items in weight w_{ij}. A $1 \times [(2r+1)^2 - 1]$ distance template D contains the inversely relative distances from each of the $(2r+1)^2 - 1$ *neighbors* to the center, as shown in Fig. 10. Thus, we have

$$W = T + M(\Delta T) \times D^T. \tag{11}$$

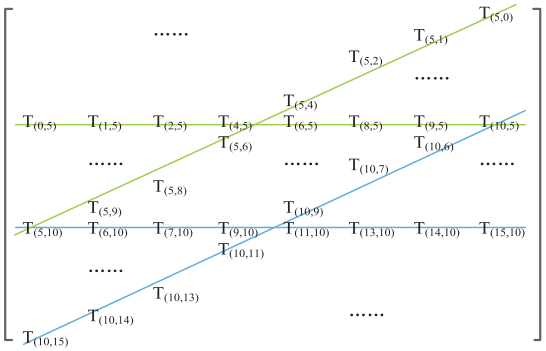
Taking Fig. 7(a) for example, the *neighbors* of block b_5 are bordered by a 3×3 dotted template highlighted in green. Thus the weight of $e_{5,5}$ can be written as

$$M_{w_{5,5}} = [T_{(0,5)} \quad T_{(1,5)} \quad \cdots \quad T_{(10,5)}]$$

$$D = \left[\frac{1}{(2\sqrt{2})^3} \quad \frac{1}{2^3} \quad \cdots \quad \frac{1}{(2\sqrt{2})^3} \right]$$

$$w_{5,5} = T_5 + M_{w_{(5,5)}} \times D^T.$$

Similarly, the temperature difference matrix $M(\Delta T)$ of Fig. 7(a) is

$M(\Delta T) =$ 

Apparently, w_{ij} changes only if e_{ij} or any of its *neighbor* remap, just as the items on green line for b_5 or the items on blue line for b_{10}. We establish a $4 \times [(2r+1)^2 - 1]$ matrix $M_{\text{affected}}(\Delta T)$ to temporarily record the changes where the two rows are recorded with the computation items of two selected blocks and the rest two rows are recorded with the affected items in the *neighbors*. Therefore, we only need to calculate the $M_{\text{affected}}(\Delta T)$ in each iteration for updating $M(\Delta T)$. And thus, the time complexity of each iteration is significantly reduced to $O(r^2)$.

V. Experimental Evaluation

We performed the experiments through the flow as described in Section II. For evaluation of our TeSHoP technique, we additionally inserted the sensor network into the circuits after the hotspot optimization to obtain the optimized temperature, as shown in Fig. 11. While it is worth noting that the insertion after the optimization is unnecessary for practical applications.

A. Benchmarks and Parameter Setting

The design flow of TeSHoP is the extension of VTR-to-bitstream who only supports Xilinx Virtex-6 xc6vlx240t-ff1156 FPGA with ISE design suit and a limited number of benchmark circuits at present. In consequence, we adopt *ml*605 as our target FPGA board and apply the compatible benchmarks for our experiments. Since the heterogeneous blocks of FPGAs are also supported by the VTR currently [29], our optimization supports all the available resources on chip. The original physical information of the benchmark circuits is outlined in Table I. Since it is recommended to keep the CLBs utilization under 80% for a design on chip [30], we categorize the utilization below 30%, between 30%–60%, and above 60% as small, medium, and large size design, respectively.

In the TeSHoP algorithm, we set the range of thermal effect $r = 14$, because the thermal influence out of this range is negligible even for the worst temperature distribution [25]. The ambient temperature is 25 °C during the experiments and pseudorandom vectors are continuously input to the circuit during the run-time.

The available CLBs of *ml*605 are arranged in 100×240 array and the RO-based sensor network is set with 5×5, 10×10, and 15×15 grids for experiments. The resource utilization of the three different grid partitions is listed in Table II. For each grid partition, we set 3 different values for the parameter γ, which are 0.01, 0.03, 0.05. And the other two parameters, α and β, are set equivalent and simultaneously satisfy $\alpha + \beta + \gamma = 1$.

We use a typical benchmark *stereovision*2 to discuss the parameter γ and the grid partition of the sensor network under different values. The experimental results are depicted in Table III. We evaluate hotspot optimization with the aspects of peak temperature reduction and thermal uniformity improvement. And the peak temperature is evaluated by the maximum value in the temperature profile while the thermal uniformity is calculated with the standard deviation of grid temperatures on chip. As shown in the table, optimizations of 5×5 grids are much worser than that of 10×10 and 15×15, while 15×15 grids are comparable with 10×10 for both peak temperature and thermal uniformity, but 15×15 grids take

TABLE I
Resource Utilization of Benchmark Circuits

Benchmarks	Utilization	Design Parameters	Size	Application
stereovision3	CLB: 0.11%	Multiplier: 0	Small	Image Processing
	BRAM: 0%	Adder: 31		
	DSP: 0%	Max Adder Size: 12		
LU8PEEng	CLB: 17.59%	Multiplier: 8	Small	Matrix Decomposition
	BRAM: 10.10%	Adder: 6083		
	DSP: 2.08%	Max Adder Size: 47		
stereovision2	CLB: 25.17%	Multiplier: 564	Small	Image Processing
	BRAM: 0%	Adder: 15096		
	DSP: 73.44%	Max Adder Size: 35		
mcml	CLB: 51.32%	Multiplier: 130	Medium	Image Processing
	BRAM: 38.22%	Adder: 29611		
	DSP: 22.92%	Max Adder Size: 130		
LU32PEEng	CLB: 62.73%	Multiplier: 32	Large	Matrix Decomposition
	BRAM: 36.10%	Adder: 17819		
	DSP: 8.3%	Max Adder Size: 47		

TABLE II
Resource Utilization of Different Grids

Resource Type	5*5 Grids	10*10 Grids	15*15 Grids
LUT	0.37%	1.14%	2.73%
FF	0.13%	0.35%	0.80%
BRAM	0.12%	0.12%	0.12%
Others	0.05%	0.37%	1.48%
Total	0.67%	1.98%	5.13%

TABLE III
Parameter Setting and Thermal Improvement

Grids	Hotspot Weight(γ)	Peak Temperature Reduction(°C)	Thermal Uniformity Improvement(σ)	Execution times (Normalized)
5*5	0.01	0.83	0.31	1
	0.03	0.51	0.61	1.012
	0.05	-0.6	0.06	1.048
10*10	0.01	1.08	0.48	1.071
	0.03	1.72	0.53	1.080
	0.05	1.77	0.46	1.088
15*15	0.01	1.16	0.55	1.059
	0.03	2.06	0.67	1.068
	0.05	1.94	0.69	1.071

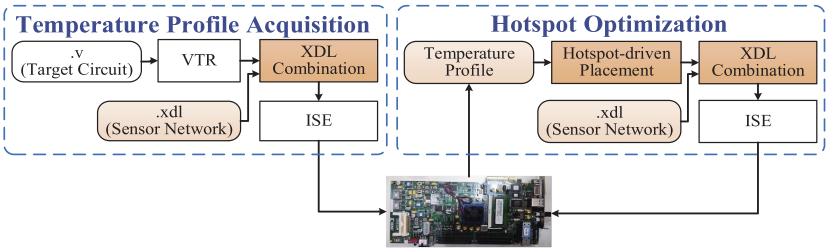


Fig. 11. Experimental flow of TeSHoP technique.

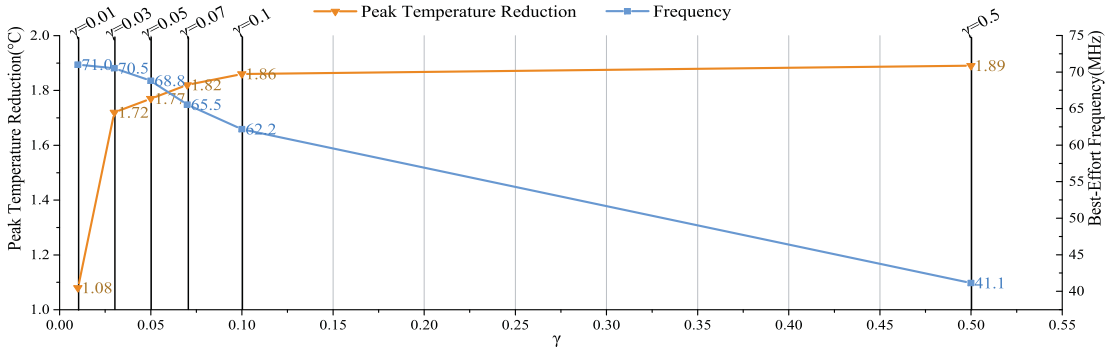


Fig. 12. Optimization and overhead under various γ.

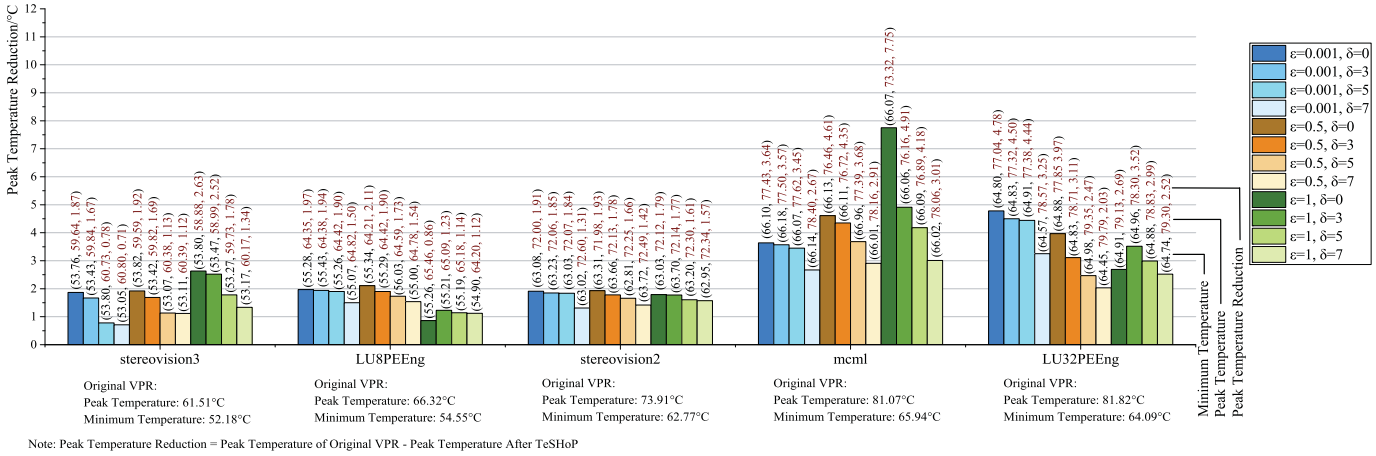


Fig. 13. Peak temperature reductions under various ε and δ.

longer execution times and require 3.15% more resource utilization for online temperature sensing, which may be inapplicable for circuits of high resource requirement. As for the tradeoff parameters, hotspot weight of 0.03 and 0.05 achieves significantly better optimizations than that of 0.01 for both peak temperature and thermal uniformity. Since the peak temperature reduction and performance are the two key concerns in this paper, the corresponding optimization and overhead for $\gamma \geq 0.01$ under the 10×10 grid partition are shown in Fig. 12. The peak temperature has a significant reduction and then a little improvement for $\gamma \geq 0.03$. And the frequency shows that the performance has a little degradation for $\gamma \leq 0.03$, however, it decreases rapidly along with the increase of γ.

With the experiments and analysis above, we set the tradeoff factors to be $\alpha = 0.485$, $\beta = 0.485$, $\gamma = 0.03$ in the TeSHoP and partition the Virtex-6 FPGA with 10×10 grids for RO-based sensor network in our experiments.

The temperature difference lower bound δ and the overhead threshold ε are two user-defined parameters. Higher δ could increase the possibility of hotspots mitigation and lower ε could reduce the disturbance on optimized area and timing. Since the feasible range of the two parameters are $\delta\epsilon[0, 1], \varepsilon\epsilon(0, +\infty)$, we take a series of parameter pairs in their feasible range for evaluation.

B. Results and Analysis of Hotspot Reduction

Since the average online temperature is empirically at least 5 °C lower than the peak temperature, we take the δ with 0,3,5,7 and ε with 0.001,0.5,1 for experiments.

The peak temperature reductions under various parameter settings are depicted in Fig. 13. Above each bar, the data

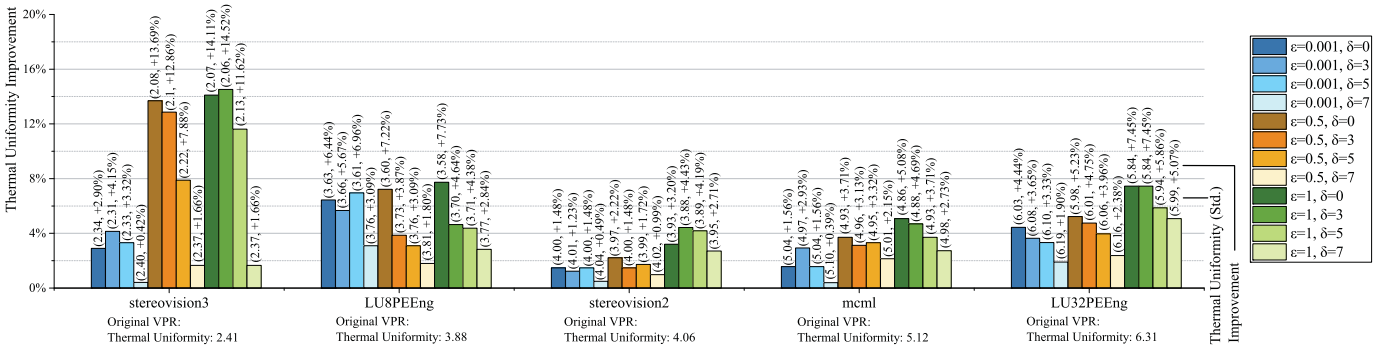


Fig. 14. Thermal uniformity improvement under various ε and δ.

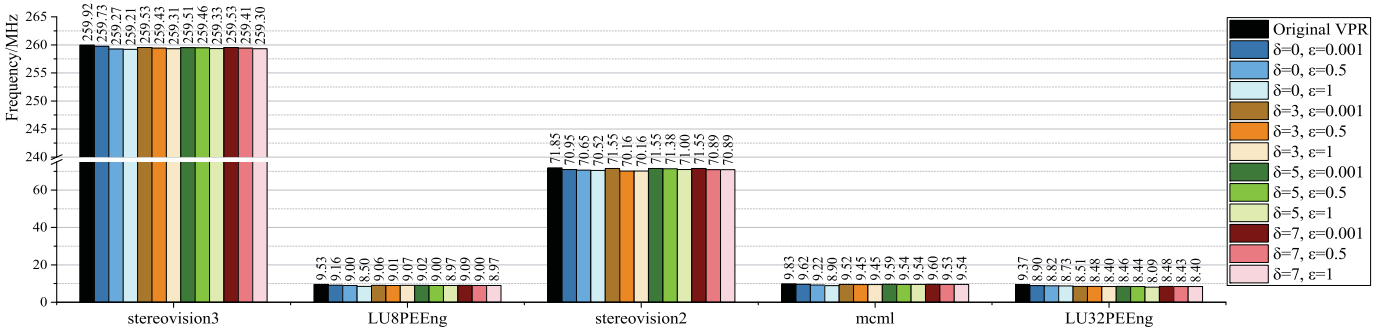


Fig. 15. Best-effort performance under various ε and δ.

from bottom to top represents the minimum temperature, the peak temperature, and the peak temperature reduction under the corresponding parameter setting. Bars with the same color scheme have the same ε. The peak temperature reduction exhibits a negative relation with the parameter δ for most benchmark circuits, which can be seen from most groups of bars with the same color scheme in each circuit.

The thermal uniformity under various parameter settings is depicted in Fig. 14. Above each bar, the data from bottom to top represents the thermal uniformity and the improvement ratio. Bars with same color shade have the same δ. The thermal uniformity exhibits a positive relation with the parameter ε for most benchmark circuits, which can be learnt from most groups of bars with the same color shade in each circuit.

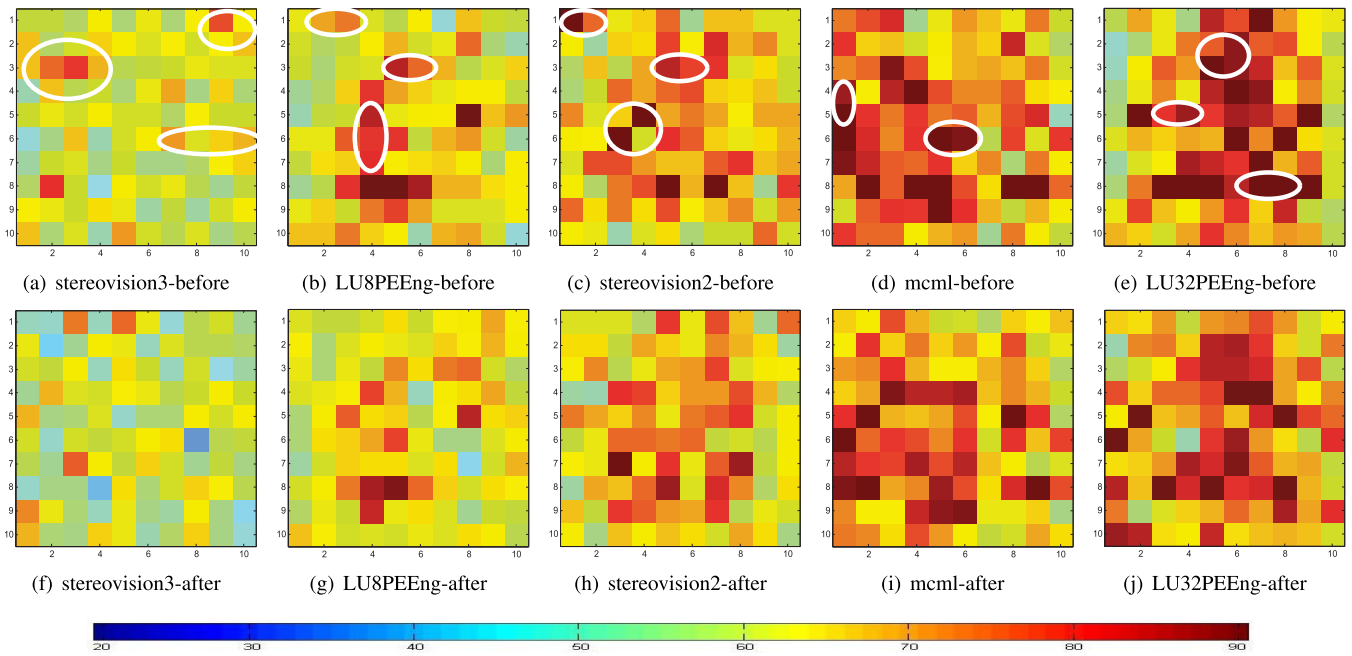
In the adopted benchmark circuits, the small size circuits achieve the most significant improvement on the thermal uniformity with up to 14.52%. Since the small size circuits occupy a minority of blocks with a small proportion of hotspots, the reduction on hotspots would prominently improve the thermal uniformity. However, the peak temperature reduction in the small size circuits is less impressive than the medium or large size circuits, because the difference between the peak and minor temperature is less severe, which indicates a smaller optimization space on hotspots. The medium size circuit achieves the most significant reduction on peak temperature with up to 7.75 °C. The medium size circuit has a larger amount of hotspots than the small size circuit and a greater space for hotspot migration than the large size circuit. Therefore, the hotspots can be effectively reduced in the medium size circuits. The large size circuit achieves a comprehensive optimization on both of the peak temperature reduction and thermal uniformity improvement. Since a majority of blocks are occupied on the FPGA, the hotspots are extremely serious for the large size circuits. Even though it has less space for hotspots migration, the distinct optimization on the peak temperature and thermal uniformity demonstrate the effectiveness of the TeSHoP technique.

C. Results and Analysis of Performance

To evaluate the performance influence of the TeSHoP technique, we extract the critical path delay of the optimized circuit-only designs in the post-route stage via the timing analyzer tool of ISE design suit.

To evaluate the performance of the TeSHoP with the best-effort timing constraints, we add a physical timing constraint at the routing stage of ISE. The physical timing constraint is tightened step by step to explore the upper bound of the constraint. The best-effort frequencies of each benchmark circuit under various parameter settings are illustrated in Fig. 15. The bars in black exhibit the frequencies of original VPR. Bars with the same color scheme have the same δ. As can be seen that the frequencies of the design after the TeSHoP are still lying in the same level as the original ones, with slight performance degradation. In addition, for the same δ, most circuits demonstrate a slight degradation along with the increase in ε. All the benchmark circuits degrade distinctly on the frequency for $\varepsilon > 0.001$.

Our technique can optimize hotspots up to 7.75 °C reduction in peak temperature and 14.52% improvement in thermal uniformity. With the consideration of performance conservation, our technique can achieve up to 4.78 °C peak temperature reduction and 6.96% thermal uniformity improvement with slight performance degradation. Even though the



NOTE: A square in a sub-figure represents a grid unit of an online-measured temperature profile.

Fig. 16. Temperature profile before and after the TeSHoP with $\delta = 7$, $\varepsilon = 0.001$. (a) Stereovision3-before. (b) LU8PEEng-before. (c) Stereovision2-before. (d) mcml-before. (e) LU32PEEng-before. (f) Stereovision3-after. (g) LU8PEEng-after. (h) Stereovision2-after. (i) mcml-after. (j) LU32PEEng-after.

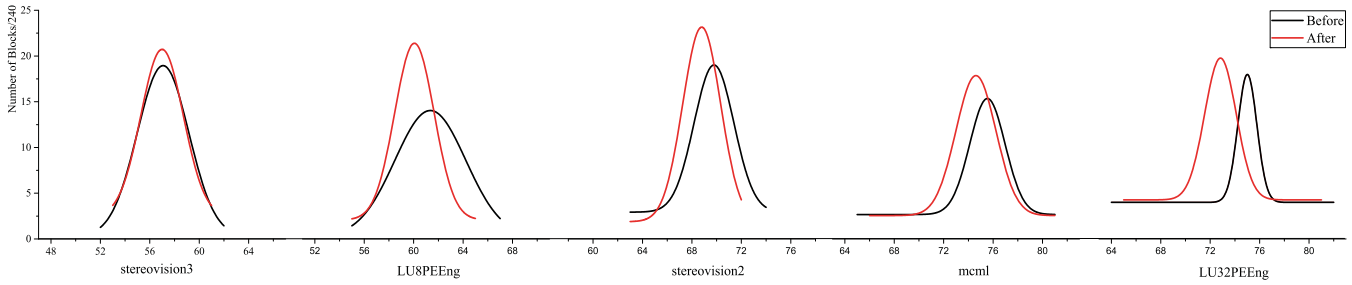


Fig. 17. Gaussian fit of temperature distribution before and after the TeSHoP with $\delta = 7$, $\varepsilon = 0.001$.

hotspot optimization degrades slightly with the consideration of performance, the hotspot reduction and thermal uniformity improvement are still significant for all the benchmark circuits. As an illustration, the TeSHoP with $\delta = 7$, $\varepsilon = 0.001$ is adopted as a representation. It achieves better performance conservation with less significant peak temperature reduction among the parameter settings. Fig. 16 shows the temperature profile before and after the TeSHoP with $\delta = 7$, $\varepsilon = 0.001$. Hotspots are effectively mitigated after the optimization, just as some examples highlighted in white circles. More specifically, Fig. 17 depicts the Gaussian fit of temperature distributions before and after the TeSHoP with $\delta = 7$, $\varepsilon = 0.001$ for each circuit. The curve in leftward means lower temperature for the whole FPGA. And the narrower the shape of curve is, the better temperature uniformity the FPGA has. We can see that all the benchmarks have a significant improvement on both temperature reduction and thermal uniformity, which is particularly distinct in larger circuits.

The execution time of an example implementation, the TeSHoP with $\delta = 5$ and $\varepsilon = 0.001$, is listed in Table IV.

TABLE IV
EXECUTION TIME OF TeSHoP WITH $\delta = 5$, $\varepsilon = 0.001$

Benchmark Circuits	Execution Time(hour)		
	Original Placement (VTR)	Hotspot-driven Placement	TeSHoP($\delta = 5$, $\varepsilon = 0.001$) (Verilog to Bitstream)
stereovision3	0.05	0.05	0.39
LU8PEEng	0.34	0.42	1.21
stereovision2	0.54	0.89	1.87
mcml	4.87	2.65	8.47
LU32PEEng	2.78	3.29	6.97

The ISA starts from the area and timing optimized placement and the execution efficiency is effectively enhanced by the algorithm improvement. Thus, the execution time of the hotspot-driven placement is comparable with the original one and even gains improvement for some benchmark circuits. The execution time of the whole TeSHoP implementations is from 0.39 to 6.97 h, where the prolonged time mainly results from the double implementation of the placement and routing.

VI. Conclusion

We have presented a deeper exploration of TeSHoP technique for FPGAs. First, we propose an on-chip sensing method to obtain the temperature profile for fine-grained design optimization, which reflects a much more accurate online temperature distribution of the FPGA. Second, we provide a high-level formulation of the hotspot-driven placement problem with the hyper minimum bipartite matching model and propose a heuristic algorithm for solving the problem. The proposed algorithm is efficient in hotspot optimization and, meanwhile, capable of keeping the performance from being disturbed as much as possible. Furthermore, we extend the VTR-to-bitstream to an open-source/commercial hybrid design flow to implement the hotspot reduction. The proposed design flow supports the commonly used physical timing constraints of the commercial tool. All the experimental statistics come from the results measured on a real FPGA. The results show that our technique can optimize hotspots up to 7.75 °C reduction in peak temperature and 14.52% improvement in thermal uniformity. With the setting of the user-defined parameters, our technique is adaptive to performance-sensitive circuits with up to 4.78 °C peak temperature reduction and 6.96% thermal uniformity improvement under specific timing constraint.

References

[1] Xilinx. (Jan. 2015). *Leveraging Power Leadership at 28 nm With Xilinx 7 Series FPGAs*. [Online]. Available: http://www.xilinx.com

[2] S. Wei, J. Zheng, and M. Potkonjak, "Aging-based leakage energy reduction in FPGAs," in *Proc. Int. Conf. Field Program. Logic Appl. (FPL)*, 2013, pp. 1–4.

[3] S. Heo, K. Barr, and K. Asanović, "Reducing power density through activity migration," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, 2003, pp. 217–222.

[4] S. Golshan, E. Bozorgzadeh, and B. C. Schafer, "Exploiting power budgeting in thermal-aware dynamic placement for reconfigurable systems," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, 2010, pp. 49–54.

[5] H. Khdr *et al.*, "mDTM: Multi-objective dynamic thermal management for on-chip systems," in *Proc. Conf. Design Autom. Test Eur. (DATE)*, 2014, pp. 1–6.

[6] D. Pagano *et al.*, "Thermal-aware floorplanning for partially-reconfigurable FPGA-based systems," in *Proc. Conf. Design, Autom. Test Eur. (DATE)*, 2015, pp. 920–923.

[7] R. Chafi, P, M. Moradi, and N. Rahmanikia, "A platform for dynamic thermal management of FPGA-based soft-core processors via dynamic frequency scaling," in *Proc. IEEE Iranian Conf. Electr. Eng. (ICEE)*, Sep. 2015, pp. 1093–1097.

[8] I. Christoforakis *et al.*, "Dithering-based power and thermal management on FPGA-based multi-core embedded systems," in *Proc. IEEE Int. Conf. Embedded Ubiquitous Comput. (EUC)*, Apr. 2015, pp. 173–177.

[9] H. Shen and Q. Qiu, "An FPGA-based distributed computing system with power and thermal management capabilities," in *Proc. Int. Conf. Comput. Commun. Netw. (ICCCN)*, 2011, pp. 1–6.

[10] C. C. N. Chu and D. F. Wong, "A matrix synthesis approach to thermal placement," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 17, no. 11, pp. 1166–1174, Nov. 1998.

[11] C.-H. Tsai and S.-M. Kang, "Standard cell placement for even on-chip thermal distribution," in *Proc. Int. Symp. Phys. Design (ISPD)*, 1999, pp. 179–184.

[12] C.-H. Tsai and S.-M. Kang, "Cell-level placement for improving substrate thermal distribution," *IEEE Trans. Comput.-Aided Design Integr.*, vol. 19, no. 2, pp. 253–266, Feb. 2000.

[13] P. Sundararajan, A. Gayasen, N. Vijaykrishnan, and T. Tuan, "Thermal characterization and optimization in platform FPGAs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Jun. 2006, pp. 443–447.

[14] S. Bhoj and D. Bhatia, "Thermal modeling and temperature driven placement for FPGAs," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Sep. 2007, pp. 1053–1056.

[15] J. Jaffari and M. Anis, "Thermal-aware placement for FPGAs using electrostatic charge model," in *Proc. Int. Symp. Quality Electron. Design (ISQED)*, 2007, pp. 666–671.

[16] B. C. Schafer and T. Kim, "Hotspots elimination and temperature flattening in VLSI circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 11, pp. 1475–1487, Nov. 2008.

[17] H. Amrouch *et al.*, "Analyzing the thermal Hotspots in FPGA-based embedded systems," in *Proc. Int. Conf. Field Program. Logic Appl. (FPL)*, 2013, pp. 1–4.

[18] J. Rose *et al.*, "The vtr project: Architecture and cad for fpgas from verilog to routing," in *Proc. ACM/SIGDA Int. Symp. Field Program. Gate Arrays (FPGA)*, 2012, pp. 77–86.

[19] E. Hung, F. Eslami, and S. Wilton, "Escaping the academic sandbox: Realizing VPR circuits on Xilinx devices," in *Proc. Int. Symp. Field-Program. Custom Comput. Mach. (FCCM)*, 2013, pp. 45–52.

[20] E. Boemo and S. López-Buedo, "Thermal monitoring on FPGAs using ring-oscillators," in *Proc. Int. Conf. Field Program. Logic Appl. (FPL)*, 1997, pp. 69–78.

[21] G. Tarawneh, T. Mak, and A. Yakovlev, "Intra-chip physical parameter sensor for FPGAS using flip-flop metastability," in *Proc. Int. Conf. Field Program. Logic Appl. (FPL)*, 2012, pp. 373–379.

[22] M. Happe, A. Agne, and C. Plessl, "Measuring and predicting temperature distributions on FPGAs at run-time," in *Proc. Int. Conf. Reconfigurable Comput. FPGAs (ReConFig)*, 2011, pp. 55–60.

[23] Xilinx. (Oct. 1999). *FPGA Editor Guide*. [Online]. Available: http://www.xilinx.com

[24] D. Chen, J. Cong, and P. Pan, "FPGA design automation: A survey," *Found. Trends Electron. Design Autom.*, vol. 1, no. 3, pp. 139–169, 2006.

[25] D. S. Chandrasekharaiah and K. S. Srinath, "Thermoelastic interactions without energy dissipation due to a point heat source," *J. Elasticity*, vol. 50, no. 2, pp. 97–108, 1998.

[26] P. Mangalagiri, S. Bae, R. Krishnan, Y. Xie, and V. Narayanan, "Thermal-aware reliability analysis for platform FPGAs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Oct. 2008, pp. 722–727.

[27] V. Betz and J. Rose, "VPR: A new packing, placement and routing tool for FPGA research," in *Proc. Int. Conf. Field Program. Logic Appl. (FPL)*, 1997, pp. 213–222.

[28] V. Betz, J. Rose, and A. Marquardt, Eds., *Architecture and CAD for Deep-Submicron FPGAs*. Norwell, MA, USA: Kluwer, 1999.

[29] A. Somerville and K. B. Kent, "Improving memory support in the VTR flow," in *Proc. Int. Conf. Field Program. Logic Appl. (FPL)*, 2012, pp. 197–202.

[30] Xilinx. Oct. 2011. *Getting Started With the Xilinx Virtex-6 FPGA ML605 Evaluation Kit*. [Online]. Available: http://www.xilinx.com

Weina Lu received the B.S. degree in automatic science from Beihang University, Beijing, China, in 2012. She is currently pursuing the Ph.D. degree with the State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing.

Her current research interests include FPGA power optimization and thermal managements.

Yu Hu (M'06) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 1997, 1999, and 2003, respectively.

She is currently a Professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. Her current research interests include autonomous navigation systems, architectural-level and circuit-level reliable design, and fault diagnosis techniques.

Dr. Hu is a member of Association for Computing Machinery (ACM), The Institute of Electronics, Information and Communication Engineers (IEICE), and China Computer Federation (CCF).

Jing Ye (M'13) received the B.S. degree in electronics engineering and computer science from Peking University, Beijing, China, in 2008, and the Ph.D. degree from the State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2014.

He is currently an Assistant Professor with the State Key Laboratory of Computer Architecture. His current research interests include very large scale integration testing and diagnosis, physical unclonable function, hardware Trojan, and hardware security.

Xiaowei Li (SM'04) received the B.Eng. and M.Eng. degrees in computer science from the Hefei University of Technology, Hefei, China, in 1985 and 1988, respectively, and the Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China, in 1991.

He was an Assistant Professor from 1991 to 1993 and an Associate Professor from 1993 to 2000 with the Department of Computer Science and Technology, Peking University, Beijing. In 2000, he joined the ICT, CAS, as a Professor, where he is currently the Deputy (Executive) Director with the State Key Laboratory of Computer Architecture, ICT, CAS. He has authored or co-authored over 300 papers in journals and international conferences. He holds 52 patents and 42 software copyrights. His current research interests include very large scale integration testing, design for testability, design verification, dependable computing, and wireless sensor networks.

Prof. Li is an Editorial Board Member of the *Journal of Computer Science and Technology (JCST), Journal of Low Power Electronics (JOLPE),* and *Journal of Electronic Testing - Theory and Applications (JETTA)*. He served as the Chair of the Technical Committee on Fault Tolerant Computing, the China Computer Federation from 2008 to 2012, the Vice Chair of the IEEE Asian Pacific Regional Test Technology Technical Council since 2004, and the Steering Committee Chair of the IEEE Asian Test Symposium from 2011 to 2013). He is on the Technical Program Committee of several IEEE and ACM conferences, including International Test Conference (ITC), VLSI Test Symposium (VTS), Design, Automation & Test in Europe Conference (DATE), Asia and South Pacific Design Automation Conference (ASP-DAC), and Pacific Rim International symposium on Dependable Computing (PRDC).