Thermal-Aware Design and Flow for FPGA Performance Improvement

Behnam Khaleghi and Tajana Šimunić Rosing
CSE Department, UC San Diego, La Jolla, CA 92093, USA
Email: {bkhaleghi, tajana}@ucsd.edu

***Abstract*—To ensure reliable operation of circuits under elevated temperatures, designers are obliged to put a pessimistic timing margin proportional to the worst-case temperature (T_{worst}), which incurs significant performance overhead. The problem is exacerbated in deep-CMOS technologies with increased leakage power, particularly in *Field-Programmable Gate Arrays* (FPGAs) that comprise an abundance of leaky resources. We propose a two-fold approach to tackle the problem in FPGAs. For this end, we first obtain the performance and power characteristics of FPGA resources in a temperature range. Having the temperature-performance correlation of resources together with the estimated thermal distribution of applications makes it feasible to apply minimal, yet sufficient, timing margin. Second, we show how optimizing an FPGA device for a specific thermal corner affects its performance in the operating temperature range. This emphasizes the need for optimizing the device according to the target (range of) temperature. Building upon this observation, we propose thermal-aware optimization of FPGA architecture for foreknown field conditions. We performed a comprehensive set of experiments to implement and examine the proposed techniques. The experimental results reveal that thermal-aware timing on FPGAs yields up to 36.5% performance improvement. Optimizing the architecture further boosts the performance by 6.7%.**

I. Introduction

The continuous shrinking of transistor size has been accompanied with exacerbated reliability degradations, e.g., thermal challenges caused by intensified power density due to the failure of Dennard scaling [1]. In particular, elevated temperature exponentially increases the leakage power, which contributes to a substantial ratio of total chip power in deep-nano era, especially in *Field-Programmable Gate Arrays* (FPGAs) that comprise an abundance of leaky resources [2]–[6]. The increase in power density, in turn, further raises the temperature, forming a power-temperature positive feedback loop.

To ensure reliable operation of the circuits under elevated temperatures, in addition to setting back increased packaging costs, designers are obliged to put a pessimistic timing margin proportional to the worst-case temperature (T_{worst}). That is because, as temperature rises, transistors slow down [7]; hence, the circuit needs to be clocked according to the slowest (worst) case to meet timing constraint across the entire temperature range. Such a one-size-fits-all policy suppresses the performance as it overestimates the real timing margin. A straightforward approach to alleviate the problem could be employing more sophisticated cooling to throttle the worst-case temperature. This, however, is not always affordable in terms of cost or power requirement.

Previous studies have coped with the temperature-induced performance inefficiency generally by diminishing the required margin through decreasing the peak temperature T_{worst} [8], [9], or by online adapting (i.e., dynamic scaling) of the frequency, equivalent to narrowing the temperature-induced margin, according to the circuit temperature [10]–[13]. While the former approaches can be orthogonally employed with the frequency adapting methods, they have demonstrated limited efficacy in the scope of FPGAs. The latter approaches have yielded promising power or performance gains, however, as we will discuss in Section II, they need sophisticated techniques to obtain temperature-performance correlation and do not account for on-chip temperature variation in FPGAs [14].

In this paper, we propose a two-fold approach to enhance the performance of FPGAs by tackling the pessimistic temperature-induced delay margin and designing (i.e., fabricating) considering field thermal condition. For this end, first, we obtain the performance and power characteristics of each type of resources under a specific range of temperatures and estimate the thermal distribution of applications using thermal simulation, which if conducted precisely, is shown to provide an accuracy of 1°C [14]. Having the temperature-performance correlation of resources together with the thermal distribution of applications makes it feasible to apply small, yet sufficient, timing margin. We call this approach *thermal-aware guardbanding*. Second, we investigate the impact of operating corner on the efficient design of FPGAs. Typically, FPGAs allow mapping applications using a few temperature corners, e.g., lowest and highest supported junction temperatures [15]. Therefore, whether an FPGA fabric is optimized (fabricated) for, e.g., 0°C or 100°C may play a determinant role on its performance in each of these temperatures. This becomes further complicated when, by using the first proposed approach, the frequency of applications can also be determined based on intermediate temperatures (i.e., multiple corners). Therefore, it becomes crucial to investigate the optimum design corner that yields highest performance in a given temperature range. Building upon our investigation, we propose thermal-aware optimization of FPGA architecture according to the thermal condition of the field application. We examine the proposed method using specific commercial-like FPGA architecture and toolset, however, the approach can be generalized to different FPGA architecture and flows.

II. Related Work

The previous research coping with FPGA temperature include studies that insert thermal sensors or exploit simulation-based approaches to characterize the thermal profile, methods that propose design (mapping) flow to reduce or balance the temperature variation, and studies that aim to leverage the available temperature headroom to provide performance efficiency.

Thermal Estimation: Arguing for the inefficiency of fabrication-time insertion of thermal sensors as the thermal profile of the device widely varies across applications, the early

978-3-9819263-2-3/DATE19/©2019 EDAA

studies propose dynamic insertion and elimination of sensor circuits, e.g., *Ring Oscillator* (RO), to correlate the captured frequency of the sensor circuit with application temperature [16], [17]. These approaches utilize the unused resources to sensor sensors, which might be located distant from the temperature hotspots. Furthermore, the accuracy of such sensors is contingent upon inter-die variations, e.g., voltage fluctuations. In general, it is shown that sensor circuits may not precisely replicate the behavior of circuits *critical paths* (CPs) [11].

The study in [14] proposes a more accurate simulation-based thermal profiling augmented by measurements from a thermal camera. In this work, iteratively, the initial leakage power of each block is estimated by factorizing the reported total leakage power based on the temperature of each block which is obtained using HotSpot simulator [18]. Thereafter, the temperature-leakage convergence is considered to reflect the real operation scenario. The factorization and temperature-leakage loop parameters are calibrated by validating the simulation temperatures with camera measurements.

Thermal Mitigation: In [8], the authors characterize the thermal distribution of designs mapped on Xilinx FPGAs by importing their block-level power consumption information obtained from XPower tool [19] into a thermal simulator. They integrate the power-thermal model in the placement algorithm to isolate the hot blocks by constraining to prohibit the utilization of their adjacent blocks. This technique showed limited effectiveness. The study in [20] proposes a temperature-aware placement and routing which attempts to balance the temperature distribution by minimizing the difference of switching activities among neighbor blocks. This approach linearly correlates the activity with temperature and does not consider heat flow. In [21], the authors propose and calibrate a two-layer thermal model by measuring a time-series of temperature distributions by employing RO-based sensors. The tuned model is used to predict the runtime temperature. Despite a one-time adjustment of device parameters is required, the heat dissipation parameters for each new application should be learned, which is cumbersome.

Thermal-Aware Boosting: The work in [22] proposes timing analysis of FPGAs in real operating conditions rather than the corner cases through measuring the delay sensitivity of look-up table (LUT) chain with respect to temperature. It shows an insignificant variation of delay over a large temperature range, which is in contrast with previous studies. This work also does not propose a systematic approach to obtain the thermal profile of an application. In [10] the authors propose an online timing slack measurement circuit to be inserted in the CPs to achieve the available timing margin. This approach utilizes additional logic and clock resources which can be excessive accounting the significant number of (near-) CPs in large applications. In addition, it needs to add additional shadow register at the end of each CP, which is not possible for certain cores such as Block RAMs. Moreover, detection of timing mismatch in the proposed circuitry depends on the input and might not be triggered during operation. Another analogous approach [12] proposes a two-step method to find the frequency-voltage correlation in different temperatures. In this approach, before mapping the actual application, its CPs are extracted via timing analysis and mapped to the FPGA fabric, along with an error-checking and heating circuit. For different voltages and under a range of temperatures, the frequency of the paths is increased until they violate the timing. This approach assumes the same temperature across the entire chip (and the entire CP) while the temperature variation can reach above 20°C [14]. Therefore, still, a pessimistic temperature needs to be considered, which is inefficient. In addition, this approach will be cumbersome for large applications with an excessive number of CPs, especially CP of a design changes with temperature [11], and hence, the same CPs may not represent the worst delay at different temperatures, which has not been considered in this study.

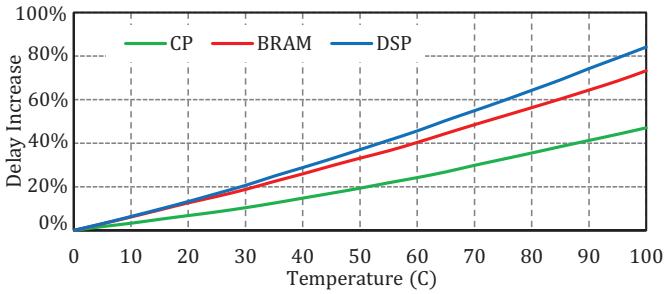


Fig. 1. Impact of temperature on the delay of FPGA resources.

Distinction from previous works:
(1) The proposed thermal-aware guardbanding leverages the temperature-delay correlation obtained accurately in the fabrication stage of FPGA.
(2) Our thermal-aware guardbanding performs offline thermal analysis. According to the obtained thermal profile, it specifies the minimum timing guardband (i.e., maximum frequency) based on the temperature (and associated delay) of the blocks.
(3) We examine the performance of an FPGA fabric when optimized for, and running in, different field temperatures. According to our investigation, we propose thermal-aware optimization of FPGA architectures that enhances their performance under a foreknown field condition.

III. Proposed Method

We begin this section with constructing the proposed guardbanding technique upon investigating the impact of temperature on FPGA building blocks. Afterwards, we represent the efficiency gain obtained by calibrating the FPGA fabric for certain operating condition. Finally, by leveraging the first two techniques, we discuss thermal-aware optimization of FPGA architectures.

A. Thermal-Aware Guardbanding

As a motivational example towards thermal-aware guardbanding in FPGAs, in Fig. 1 we have measured the impact of temperature on different components of these devices. In this figure, CP indicates the *representative critical path* delay of FPGA as a weighted average over the delay of different soft (configurable) components, e.g., LUT and routing switch boxes (SBs), according to their occurrence probability in a real design critical path [23]. The setup of the experiments is detailed in Section IV-A. According to Fig. 1, for a design merely composed of soft-fabric, the overhead associated with worst-case thermal guardbanding might reach 47% while this value for a DSP-hungry application can reach up to 84%. In addition,

Fig. 1 also advocates for the ineffectiveness of sensor circuits in predicting the timing behavior of applications under temperature variations as different resources have shown different sensitivity to temperature. Thus, a set of pre-specified circuits may not exhibit the exact behavior of the real critical path(s). Notice that even the components forming the representative CP denoted in Fig. 1 have different timing behaviors whereby the delay of LUT might increase up to 69% (for 0°C $\rightarrow$ 100°C) while the delay of switch box rises by 39%[1]. These make the need for an accurate thermal-aware guardbanding, which precisely accounts for the temperature and sensitivity of each individual resource, indispensable.

Operating temperature of an FPGA device depends on ambient temperature and power consumption of the design, whereby the latter itself depends on the operating temperature and frequency. Hence, an efficient guardbanding requires correlation between temperature, frequency, and power consumption. Algorithm 1 presents our proposed guardbanding method. First, it calculates an initial frequency for the design, assuming a base junction temperature equal to ambient temperature T_{amb} for each of FPGA tiles[2]. Based on the obtained frequency and activity factor, the dynamic power of each tile is calculated, while the leakage power of each tile is calculated based on its temperature. Note that both $p_{dyn}-f$ and $p_{lkg}-T$ relation for each type of resource are pre-identified. As for the temperature, the powers of tiles are stored in a vector with n (equal to the number of tiles) elements. Afterwards, the obtained power vector is fed to a thermal simulator (e.g., HotSpot [18]) to estimate the temperature of each tile. Notice that the dynamic power varies among the tiles because of the type (e.g., soft-fabric or hard-core) and activities of the tiles. Likewise, leakage power also varies because of the type and temperature of the tiles (though, initially, similar tiles consume equal leakage power because their initial temperature is equal). With the updated temperature, the algorithm (line 4) performs static timing analysis again considering the temperature of each tile. That is, the delay of each resource is calculated according to the temperature-delay relation of the resource (as shown in parts by

Algorithm 1: Thermal-Aware Guardbanding

Input: $netlist$: Placed and routed design
Input: T_{amb}: Ambient temperature
Input: $\overrightarrow{\alpha}$: Activity of resources
Output: f: Design frequency

1 $\overrightarrow{T}_{1\times n} = [T_{amb}, \cdots, T_{amb}]$ // n : Number of FPGA tiles
2 $\overrightarrow{\Delta T}_{1\times n} = [\infty, \cdots, \infty]$
3 **while** $\|\overrightarrow{\Delta T}\|_\infty > \delta_T$ **do**
4 $\quad f = \mathcal{T}(netlist, \overrightarrow{T})$ // Timing analysis using $\overrightarrow{T}$
5 $\quad \overrightarrow{p} = \overrightarrow{p}_{dyn}(netlist, \overrightarrow{\alpha}, f) + \overrightarrow{p}_{lkg}(\overrightarrow{T})$
6 $\quad \overrightarrow{T}_{old} = \overrightarrow{T}$
7 $\quad \overrightarrow{T} = HotSpot(\overrightarrow{p})$
8 $\quad \overrightarrow{\Delta T} = \overrightarrow{T} - \overrightarrow{T}_{old}$
9 $f = \mathcal{T}(netlist, \overrightarrow{T} + \delta_T)$

[1]For the sake of simplicity, we have not shown the individual delays.
[2]As shown in Fig. 4, an FPGA tile comprises a logic cluster (or other hard-cores) and its neighboring routing resources.

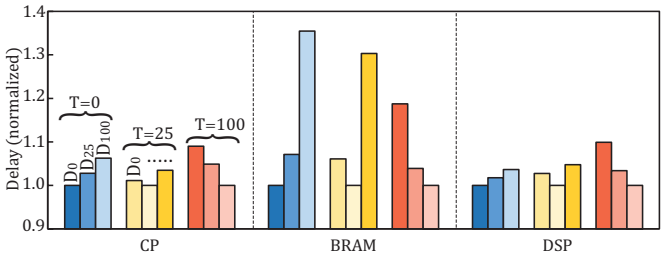


Fig. 2. Delay of differently optimized FPGA fabrics on different temperatures.

Fig. 1). Thus, the same resource will exhibit different delays in different tiles, based on the temperature of the residing tile. It is noteworthy that for accurate timing analysis, in each iteration, the entire netlist should be probed since the critical path might change at different temperatures. The updated temperature and frequency is then used to update the power. This procedure repeats until the temperature of every tile converges[3], which often takes a few (less than ten) iterations. Eventually, the operating frequency is recalculated by assuming a small margin of δ_T to compensate the convergence error. We provide further details of timing analysis, power measurement, and thermal simulation with an overview of the guardbanding flow in Section IV-A.

B. Thermal-Aware Design

In spite of the design of *Application-Specific Integrated Circuits* (ASICs) whereby the synthesis algorithm tries to optimize the constraints by choosing the most-suitable gates, the structure of FPGA fabric is foreknown. Thereby, the design tool mainly optimizes the fabric by efficient sizing of the constituting transistors of each kind of resource. In other words, optimizing an FPGA deals with efficient sizing of the resources. The target of optimization is a specific operating corner (e.g., 100°@0.8V). Whilst the operating frequency of an application can be tuned with the proposed thermal-aware guardbanding technique, a device optimized for a certain corner will not be necessarily optimum in the other temperatures.

To demonstrate this concept, we optimized (i.e., synthesized) a specific FPGA architecture for three different operating temperatures, i.e., 0°C, 25°C, and 100°C. Thereafter, we examined the timing of each synthesized FPGA in all these temperatures using our first proposed technique. That is, for a device optimized for 100°C, we measure its performance in 25°C by performing static timing analysis under 25°C; hence, *no extra margin is assumed.* Fig. 2 illustrates the results. The timing of different components, i.e., CP (a soft-fabric path comprising the configurable building blocks of FPGA), BRAM, and DSP block is shown separately. As clarified in the figure, each chunk of bars indicates a specific operating temperature. Analogously, each bar in a chunk represents a device optimized for a certain temperature. That is, D_0, D_{25}, and D_{100} denote the devices optimized for 0°C, 25°C, and 100°C, respectively. To facilitate comparison, the delays in each chunk (operating temperature) are normalized to the minimum delay of that chunk.

As it is clear from Fig. 2, all components of a device optimized for a specific temperature afford comparatively minimum delay when running in that temperature. This observation is

[3]$\|\overrightarrow{\Delta T}\|_\infty$ denotes the maximum absolute value through the vector $\overrightarrow{\Delta T}$.

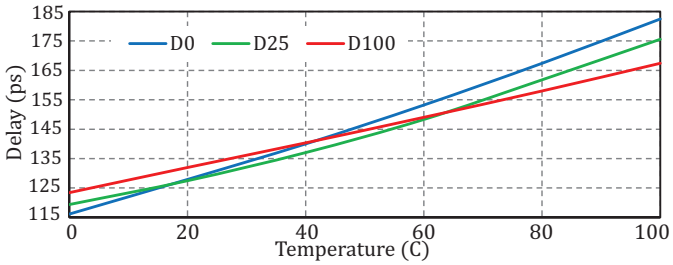


Fig. 3. Comparing the temperature-delay relation of the representative critical path in differently optimized FPGA fabrics.

intensified in the Block RAM wherein delay of the device optimized for 100°C is 1.35× of the device optimized for 0°C, when both running at 0°C. Likewise, running in 100°C, BRAM delay in the device optimized for 0°C, is 1.19× of that optimized for 100°C. As shown in the figure, a similar trend (though with less intensity) holds for the soft-fabric and DSP block, as well. In Fig. 3, we elaborate this observation for FPGA soft-fabric (CP) by comparing the delay of FPGAs optimized for 0°C (D_0), 25°C (D_{25}), and 100°C (D_{100}) in the entire operating temperature range. According to the shown measurements, the D_0 device provides 6.3% higher performance than D_{100} device when both operate in 0°C. However, as temperature elevates, D_{100} device becomes more efficient, which eventually affords 9.0% better performance in 100°C. The D_{25} device lies in between and is optimal for medium temperatures ($T \in [20°C, 65°C]$). It can be inferred from Fig. 3 that as the operating temperature converges the target optimized temperature of the device, it provides better performance. This is valid for BRAM and DSP cases shown in Fig. 2, as well. For instance, compared to D_0, the BRAM optimized for 25°C shows an *inefficiency* of only 6% when running in 0°C, while this rises to 35% for the 100°C–optimized device, D_{100}. Analogously, operating in 100°C, the BRAM of D_0 is 19% less efficient than D_{100} BRAM, while D_{25}'s BRAM is only 4% slower. It affirms that a device optimized for a certain temperature exhibits minimum delay in the target temperature, while it still supplies near-optimum delay in neighboring temperatures.

C. Thermal-Aware Architecture

A deduction of Section III-B might be devising an omnipotent FPGA fabric through optimizing a set of devices for different temperatures, and choosing the one that yields higher overall performance by studying their timing characteristics under varying temperatures. Nevertheless, our investigation shows that a single device cannot provide all-embracing superiority. Assuming the operating temperature is uniformly distributed in $[T_{min}, T_{max}]$, we define the expected delay of an FPGA fabric as:

$$E[d] = \frac{\int_{T_{min}}^{T_{max}} d(T)dT}{T_{max} - T_{min}} \quad (1)$$

As FPGA devices are usually employed in foreknown field condition, we propose thermal-aware architectures according to the field condition. For instance, while Intel Arria 10 devices support a junction temperature range of 0°C to 100°C [24], in certain cases, the field operating condition is known. A terrestrial example is the state-of-the-art datacenters whereby the heat generated by CPUs reaches 68°C, which can raise the temperature of the embedded FPGA accelators (which are gaining traction) up to 100°C [25]. For such applications, a new FPGA grade optimized for high temperatures might be determined. Notice that defining different grades is common in FPGAs, e.g., each class of Arria 10 FPGAs already offers different speed grades. Obviously, a device architecture optimized for, e.g., 70°C can still operate under low temperatures though it yields higher efficiency in elevated temperatures.

TABLE I
ARCHITECTURAL PARAMETERS USED IN COFFE

Parameter	Value	Parameter	Value
K	6	SB_{mux}	12
N	10	CB_{mux}	64
Channel tracks	320	$local_{mux}$	25
Wire segment length	4	$V_{dd}, V_{low\ power}$	$0.8V, 0.95V$
Cluster global inputs	40	BRAM	$1024 \times 32\,bit$

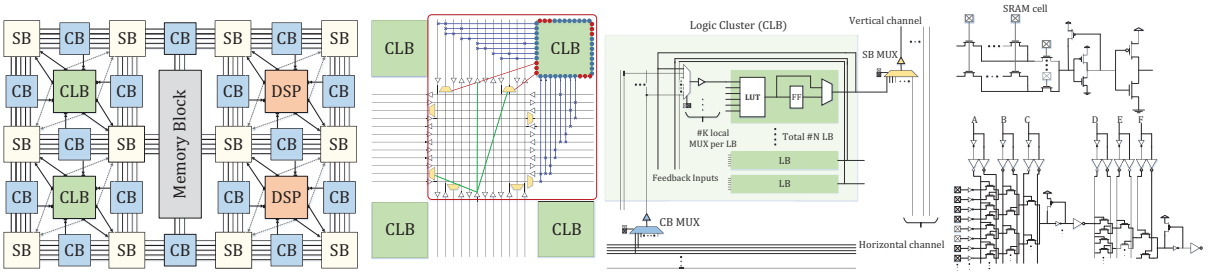
IV. EXPERIMENTAL SETUP AND RESULTS

In this section we first elaborate the general setup and flow of the experiments conducted in this work, and then present the results of the proposed method. Notice that while we have used a particular FPGA architecture and design tools, the proposed flow incorporates any arbitrary FPGA architecture and characterization tools.

A. General Setup and Flow

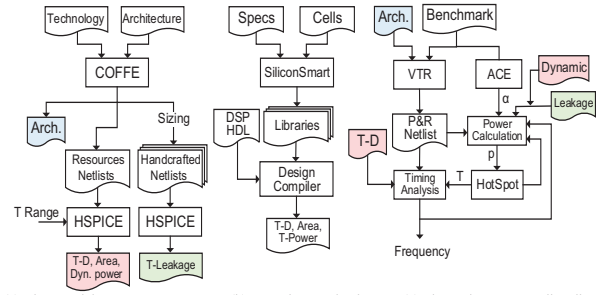
Netlists: As shown in Fig. 5(a), to generate and obtain the delay of FPGA soft-fabric resources and Block RAM, we utilized the latest version of COFFE [23]. Given the architectural description, COFFE models the FPGA resources and performs automated transistor sizing to attain an area-delay optimum architecture. It requires a transistor technology to generate and utilize the SPICE netlist of components, for which we fed 22nm high-performance PTM process model [26] for the soft-fabric, while we used its low power (high V_{th}) transistors for the BRAM core. The architecture and components structure generated by COFFE, partially shown in Fig. 4, conforms with that of commercial devices [27], [28]. Table I summarizes the target architectural parameters considered in our experiments, mostly following COFFE defaults. Note that BRAM optimization in COFFE requires to provide it with the leakage current of the weakest SRAM cell in the target temperature, which we obtained by carrying out Monte Carlo simulations over V_{th} variations in various temperatures, as suggested by [29].

Performance: To obtain the temperature-delay relation of resources (previously represented in Section III), we generated the netlist of soft-fabric and BRAM resources in a base temperature (e.g., 25°C) using COFFE, and then swept the temperature over 0°C → 100°C, which concurs with the supported junction temperature range of commercial devices [24]. As shown in Fig. 5(b), for timing characterization of the DSP block under varying temperature, we created multiple standard cell libraries with the aim of Synopsys SiliconSmart 2016. Each library targets a specific temperature within [0°C, 100°C] range. The tool automatically characterizes a given SPICE description of cells into a *liberty* file format, which can be utilized by synthesis tools such as Design Compiler. For SPICE description of the required cells, we exploited NanGate Open Cell Library that provides post-layout netlist of various combinational and sequential cells [30]. Eventually, we synthesized a Stratix-like



(a) An island-style FPGA architecture (b) A representative tile with SB and CB cons. (c) Detailed view of a tile (d) Multiplexer and LUT structure

Fig. 4. Overview of the architecture [2] and building elements [23] of island-style FPGAs.



(a) Characterizing FPGA resources (b) DSP characterization (c) Thermal-aware guardbanding

Fig. 5. General overview of implementation of the proposed method.

DSP block [31] using the created cell library by Synopsys Design Compiler 2013.12 for a base temperature and then performed static timing analysis on different temperatures by sweeping the libraries over the synthesized design.

Area and power: Area of the resources are directly reported by COFFE and Design Compiler. COFFE also reports the dynamic power of the soft-fabric and BRAM elements for a fixed frequency, which can be linearly scaled to other frequencies and activity factors. To measure the leakage power, however, we handcrafted the full SPICE netlist of the entire resources as COFFE only models the critical path of each resource. As it is required for thermal simulations, we conducted the leakage measurements over the whole temperature range. We obtained the dynamic power report of DSP from Design Compiler. Similarly, we obtained the leakage power of the DSP under different temperatures by sweeping the library.

Table II summarizes the area (μm^2), delay (ps) and power (uW) characterization of a device optimized for 25°C which, as discussed in the following, we use to examine the proposed guardbanding method. Delay and leakage power are shown as a function of temperature, T, for which we measured them with the step of 1°C and then obtained the best fitting function. Dynamic powers are measured under 100MHz and switching probability of 1 ($\alpha = 1$) which, following $p_{dyn} = \frac{1}{2}\alpha CV^2 f$, can be scaled linearly to other frequencies and activities. The area of an entire soft-fabric tile is $\sim 1196 \mu m^2$.

Placement and routing: The implementation flow of the proposed method along with the employed toolset is demonstrated in Fig. 5(c). We used open source *Verilog-to-Routing* (VTR) 7.0 [32] which enables a full FPGA-based design stack from logic synthesis to placement, routing, and timing analysis and supports wide range of architectures. For its input architecture description, we have provided COFFE-generated architecture file. Benchmarks consist of the 19 designs of the VTR repository that comprise an average (maximum) of 17K (89K) 6-input LUTs, 39 (334) BRAMs, and 19 (213) DSP blocks. Afterwards, according to Algorithm 1, we use the placed and routed output netlist (with the reported frequency) of VTR along with the signals activity estimated by ACE 2.0 [33] as well as the previously characterized dynamic power and leakage-temperature information of the resources (Fig. 5(a) and Fig. 5(b)) to estimate power using an in-house script. Notice that, in addition to the tiles placement, the routing information is also required for accurate estimation of the dynamic power distribution in routing resources. The estimated power vector is given to HotSpot simulator [18] to estimate the temperature of the tiles. Though the absolute value of the measured powers are not necessarily equal to commercial FPGAs, we cross-validated the thermal simulations exploiting Xilinx Power Estimator spreadsheet [19] by having similar temperature sensitivity with respect to power density, i.e., $\Delta T \simeq 0.7 \frac{p_{design}}{p_{base}}$ in which p_{design} is the estimated total power of the design and p_{base} is the device base (leakage) power. With the updated temperature and pre-characterized temperature-delay information of resources, we recalculate the timing information of the placed and routed netlist. For timing analysis, we leverage and modify timing analyzer of VTR to update the nodes delay according to the residing tile temperature. As explained by Algorithm 1, this procedure repeats until that temperature shows trivial change in consecutive iterations.

TABLE II
PRIMARY AREA, DELAY, AND POWER RESULTS OF RESOURCES

Parameter	$Area,\ Delay,\ P_{dyn},\ P_{lkg}$	Parameter	$Area,\ Delay,\ P_{dyn},\ P_{lkg}$
SB_{mux}	$2.8\|166+0.67T\|5.74\|0.28e^{0.014T}$	$output_{mux}$	$0.6\|31+0.17T\|0.3\|0.24e^{0.014T}$
CB_{mux}	$5.7\|112+0.70T\|0.64\|0.26e^{0.014T}$	LUT_A	$33\|163+1.4T\|1.6\|2.5e^{0.015T}$
$local_{mux}$	$1.2\|65+0.350T\|0.15\|0.06e^{0.015T}$	BRAM	$7811\|902+6.74T\|6.85\|6.2+(\frac{T}{70})^2$
$feedback_{mux}$	$0.9\|100+0.54T\|0.63\|0.23e^{0.014T}$	DSP	$5338\|547+4.42T\|879\|24.4e^{0.01T}$

B. Experimental Results

The experiments target both the thermal-aware guardbanding and optimization approaches. As the effectiveness of thermal-aware guardbanding is dependent on the ambient (hence, application) temperature, we considered different ambient temperatures, i.e., 25°C and 70°C. Fig. 6 and Fig. 7 demonstrate the performance improvement obtained by thermal-aware guardbanding whereby, respectively, 36.5% and 14% frequency increase can be observed. For the baseline, we assumed $T_{worst} =$ 100°C, hence timing guardband of the baseline is conventionally considered assuming maximum operating temperature. As expected, with ambient temperature of 25°C there is more

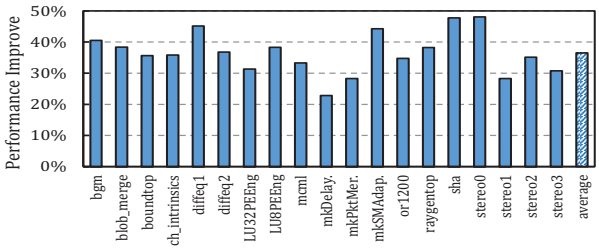


Fig. 6. Performance gain of thermal-aware guardbanding at $T_{amb} = 25°C$.

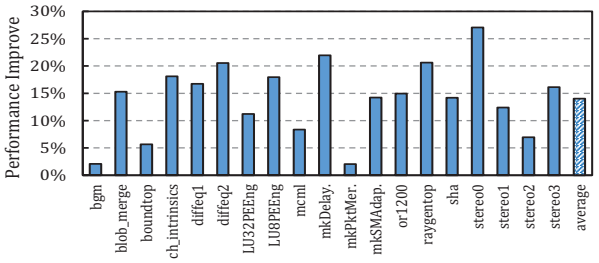


Fig. 7. Performance gain of thermal-aware guardbanding at $T_{amb} = 70°C$.

room to increase the frequency before the application violates timing. In both cases, due to relatively low switching rate, the temperature converged after $\sim 2°C$ increase.

To examine the efficiency of the thermal-aware architectural optimization, we considered high temperature operational environment and set the optimization temperature to 70°C (similarly, low or mid temperature conditions can be considered, as well). We compared the frequency of benchmarks mapped to the 70°C–optimized device with a typical device (synthesized to $25°C@0.8V$). Both devices also employ thermal-aware guardbanding (rather than wort-case guardbanding), hence they operate with maximum performance. As shown by Fig. 8, thermal-aware architectural optimization further boosts the performance by 6.7%. The variation in performance gains depends on the resources forming the critical path as BRAM and certain soft-fabric resources are more sensitive to the size (optimization) of the transistors. Note that, as discussed in Section III-B, the 70°C–optimized device is expected to supply near-optimum performance in a range of neighboring temperatures.

V. Conclusion

In this paper, we proposed design and timing flow for boosting the FPGA devices. The proposed method aims on obtaining the temperature-delay correlation for FPGA resources to select accurate, non worst-case, timing margin based on the application thermal profile. It showed 36.5% performance boosting when the FPGA operates at ambient temperature of $25°C$. In addition, we proposed architectural optimization of the device according to foreknown field temperature conditions. A device optimized for high temperature (70°C) afforded 6.7% performance gain over the typical device.

Acknowledgements

This work was partially supported by CRISP, one of six centers in JUMP, an SRC program sponsored by DARPA, and also NSF grants #1730158 and #1527034. We thank Sadegh Yazdanshenas from University of Toronto for his valuable help in COFFE simulations.

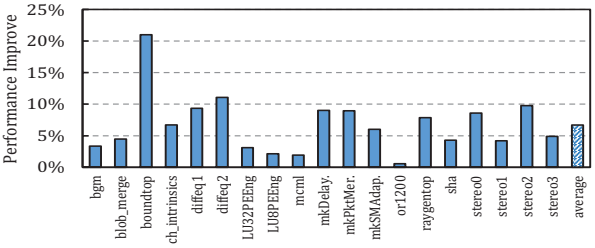


Fig. 8. Performance improvement of thermal-aware architecture optimized for $T_{amb} = 70°C$ over the baseline (both employ thermal-aware guardbanding).

References

[1] M. Horowitz, E. Alon, D. Patil, S. Naffziger, R. Kumar, and K. Bernstein, "Scaling, power, and the future of cmos," in *Electron Devices Meeting, 2005. IEDM Technical Digest. IEEE International*, pp. 7–pp.

[2] A. A. Bsoul and S. J. Wilton, "An fpga architecture supporting dynamically controlled power gating," in *Field-Programmable Technology, International Conference on*. IEEE, 2010, pp. 1–8.

[3] Z. Seifoori, B. Khaleghi, and H. Asadi, "A power gating switch box architecture in routing network of sram-based fpgas in dark silicon era," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2017, pp. 1342–1347.

[4] Z. Ebrahimi, B. Khaleghi, and H. Asadi, "Peaf: A power-efficient architecture for sram-based fpgas using reconfigurable hard logic design in dark silicon era," *IEEE Transactions on Computers*, vol. 66, no. 6, pp. 982–995, 2017.

[5] Z. Seifoori, Z. Ebrahimi, B. Khaleghi, and H. Asadi, "Introduction to emerging sram-based fpga architectures in dark silicon era," *Advances in Computers*, 2018.

[6] S. Yazdanshenas and H. Asadi, "Fine-grained architecture in dark silicon era for sram-based reconfigurable devices," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 61, no. 10, pp. 798–802, 2014.

[7] D. Wolpert and P. Ampadu, "Temperature effects in semiconductors," in *Managing temperature effects in nanoscale adaptive systems*. Springer, 2012, pp. 15–33.

[8] P. Sundararajan, A. Gayasen, N. Vijaykrishnan, and T. Tuan, "Thermal characterization and optimization in platform fpgas," in *Computer-Aided Design, 2006. ICCAD'06. IEEE/ACM International Conference on*, 2006, pp. 443–447.

[9] S. Bhoj and D. Bhatia, "Thermal modeling and temperature driven placement for fpgas," in *Circuits and Systems, IEEE International Symposium on*, 2007, pp. 1053–1056.

[10] J. M. Levine, E. Stott, and P. Y. Cheung, "Dynamic voltage & frequency scaling with online slack measurement," in *Proceedings of the 2014 ACM/SIGDA international symposium on Field-programmable gate arrays*, pp. 65–74.

[11] H. Amrouch, B. Khaleghi, and J. Henkel, "Optimizing temperature guardbands," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2017, pp. 175–180.

[12] S. Zhao, I. Ahmed, C. Lamoureux, A. Lotfi, V. Betz, and O. Trescases, "A universal self-calibrating dynamic voltage and frequency scaling (dvfs) scheme with thermal compensation for energy savings in fpgas," in *Applied Power Electronics Conference and Exposition (APEC)*. IEEE, 2016, pp. 1882–1887.

[13] H. Amrouch, B. Khaleghi, and J. Henkel, "Voltage adaptation under temperature variation," in *2018 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*. IEEE, 2018, pp. 57–60.

[14] A. Amouri, H. Amrouch, T. Ebi, J. Henkel, and M. Tahoori, "Accurate thermal-profile estimation and validation for fpga-mapped circuits," in *Field-Programmable Custom Computing Machines (FCCM), 21st Annual International Symposium on*. IEEE, 2013, pp. 57–60.

[15] "Timing analyzer user guide," User Guide, Intel, May 2018.

[16] S. Lopez-Buedo, J. Garrido, and E. I. Boemo, "Dynamically inserting, operating, and eliminating thermal sensors of fpga-based systems," *IEEE Transactions on components and packaging technologies*, vol. 25, no. 4, pp. 561–566, 2002.

[17] S. Velusamy, W. Huang, J. Lach, M. Stan, and K. Skadron, "Monitoring temperature in fpga based socs," in *Computer Design: VLSI in Computers and Processors, IEEE International Conference on*, 2005, pp. 634–637.

[18] R. Zhang, M. R. Stan, and K. Skadron, "Hotspot 6.0: Validation, acceleration and extension," *University of Virginia, Tech. Rep*, 2015.

[19] "Xilinx power estimator user guide," User Guide, Xilinx, October 2013.

[20] K. Siozios and D. Soudris, "A novel methodology for temperature-aware placement and routing of fpgas," in *VLSI, IEEE Computer Society Annual Symposium on*, 2007, pp. 55–60.

[21] M. Happe, A. Agne, and C. Plessl, "Measuring and predicting temperature distributions on fpgas at run-time," in *Reconfigurable Computing and FPGAs (ReConFig), International Conference on*. IEEE, 2011, pp. 55–60.

[22] M. A. Kacou, F. Ghaffari, O. Romain, and B. Condamin, "Fpga static timing analysis enhancement based on real operating conditions," in *Industrial Electronics Society, IECON 2017-43rd Annual Conference of the IEEE*, pp. 3556–3561.

[23] S. Yazdanshenas and V. Betz, "Automatic circuit design and modelling for heterogeneous fpgas," in *Field Programmable Technology (ICFPT), International Conference on*. IEEE, 2017, pp. 9–16.

[24] "Intel arria 10 device datasheet," Datasheet, Intel, June 2018.

[25] A. Putnam, A. M. Caulfield, E. S. Chung, D. Chiou, K. Constantinides, J. Demme *et al.*, "A reconfigurable fabric for accelerating large-scale datacenter services," *ACM SIGARCH Computer Architecture News*, vol. 42, no. 3, pp. 13–24, 2014.

[26] Predictive technology model. [Online]. Available: http://ptm.asu.edu/

[27] D. Lewis, E. Ahmed, G. Baeckler, V. Betz, M. Bourgeault, D. Cashman *et al.*, "The stratix ii logic and routing architecture," in *Proceedings of the 2005 ACM/SIGDA 13th international symposium on Field-programmable gate arrays*, pp. 14–20.

[28] I. Kuon, R. Tessier, J. Rose *et al.*, "Fpga architecture: Survey and challenges," *Foundations and Trends® in Electronic Design Automation*, vol. 2, no. 2, pp. 135–253, 2008.

[29] S. Yazdanshenas, K. Tatsumura, and V. Betz, "Don't forget the memory: Automatic block ram modelling, optimization, and architecture exploration," in *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pp. 115–124.

[30] Nangate open cell library. [Online]. Available: http://nangate.com/

[31] A. Boutros, S. Yazdanshenas, and V. Betz, "Embracing diversity: Enhanced dsp blocks for low-precision deep learning on fpgas," in *Field Programmable Logic and Applications, 2018, International Conference on*. IEEE, 2018, pp. 1–8.

[32] J. Luu, J. Goeders, M. Wainberg, A. Somerville, T. Yu, K. Nasartschuk *et al.*, "Vtr 7.0: Next generation architecture and cad system for fpgas," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 7, no. 2, p. 6, 2014.

[33] J. Lamoureux and S. J. Wilton, "Activity estimation for field-programmable gate arrays," in *Field Programmable Logic and Applications, 2006. FPL'06. International Conference on*. IEEE, 2006, pp. 1–8.