Energy-aware Placement for SRAM-NVM Hybrid FPGAs

Seongsik Park, Jongwan Kim, and Sungroh Yoon*
Department of Electrical and Computer Engineering, ASRI, INMC, and Institute of Engineering Research
Seoul National University, Seoul 08826, Korea

***Abstract*—Field-programmable gate arrays (FPGAs) have been widely used in many applications due to their reconfigurability. Notably, the short development time makes the FPGAs one of the promising reconfigurable architectures for emerging applications, such as deep learning. As CMOS technology advances, however, conventional SRAM-based FPGAs have reached their limitations. To overcome these obstacles, NVM-based FPGAs have been introduced. Although NVM-based FPGAs have the features of high area density, low static power consumption, and non-volatility, they are struggling to reduce energy consumption. Their challenge is mainly caused by the access speed of NVM, which is relatively slower than SRAM. In this paper, for compensating this limitation, we suggest SRAM-NVM hybrid FPGA architecture with SRAM- and NVM-based CLBs. In addition, we propose an energy-aware placement for utilizing the SRAM-NVM hybrid FPGAs. As a result of our experiments, we were able to reduce the average energy consumption of SRAM-NVM hybrid FPGA by 22.23% and 21.94% compared to SRAM-based FPGA on the MCNC and VTR benchmarks, respectively.**

***Index Terms*—SRAM-NVM hybrid FPGA, energy-aware placement, VTR, MCNC**

I. Introduction

Field-programmable gate arrays (FPGAs) have been receiving more attention as a time-to-market of digital circuit design has become more critical. Recently, FPGAs have been widely used in various fields as not only prototypes of application-specific integrated circuits (ASICs), but also application-specific accelerators [1].

Conventional island-style FPGAs consist of configurable logic blocks (CLBs), switching blocks (SBs), and connection blocks (CBs) [2]. The CLBs are logic components that contain look-up tables (LUTs) for storing truth tables of logic functions. The SBs and CBs are routing components that take a considerable amount of area and power consumption in conventional FPGAs [3].

Most commercialized FPGAs have static random access memory (SRAM) for storing configuration and data in CLBs, SBs, and CBs. As complementary metal oxide semiconductor (CMOS) technology node continues to scale down, the SRAM-based FPGAs have faced many challenges [4]. Among the challenges, large area and static power consumption are important issues as in Fig. 1.

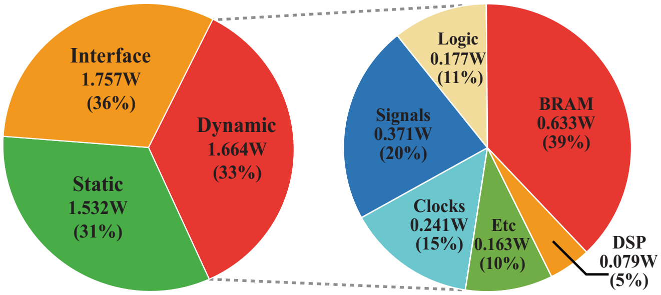

Fig. 1. Power breakdown of a recent deep neural network accelerator on Xilinx Ultrascale [1]

To address the problems, non-volatile memory (NVM) based-FPGAs have been proposed [3], [5], [6], [7], [8]. The main principle of the NVM-based FPGAs is replacing SRAM with NVM. The emerging NVMs, including resistive RAM (RRAM), phase-change RAM (PCRAM), and spin-transfer torque magnetic RAM (STTRAM), have features of high area density, low static power consumption, and non-volatility [9]. In addition, the fabrication process of these NVMs is compatible to that of CMOS by adopting the back-end-of-line (BEOL) process [7], which provides feasibility to the NVM-based FPGAs.

The research of NVM-based FPGAs can be mainly divided into two parts: architectural [3], [5], [6], [7], [8], [10], [11] and algorithmic perspectives [12], [13], [14], [15], [16], [17], [18], [19], [20]. The studies on NVM-based FPGA architecture have successfully reduced the area and static power consumption compared to the SRAM-based FPGAs. Most previous research related to placement algorithms has focused on improving the inefficiency of the NVMs in writing. However, these studies did not evaluate whether the FPGAs' energy consumption could be reduced.

SRAM-NVM hybrid FPGAs have been proposed to utilize both advantages of SRAM and NVM [10], [11], [20]. In [10], the hybrid LUTs with volatile and non-volatile inputs can reduce power consumption. In [20], SRAM-NVM dual-aware placement algorithms were proposed to improve the write endurance. However, these studies did not consider energy consumption of the FPGAs, including dynamic and static energy consumption, as well as area and delay.

In this work, we propose a placement for reducing the energy consumption on SRAM-NVM hybrid FPGAs. The proposed energy-aware placement is based on simulated annealing (SA) and exploits the slacks of each mapped block to maximize the utilization of NVMs maintaining the critical

[*] corresponding author: Sungroh Yoon (sryoon@snu.ac.kr)

978-3-9819263-4-7/DATE20/©2020 EDAA

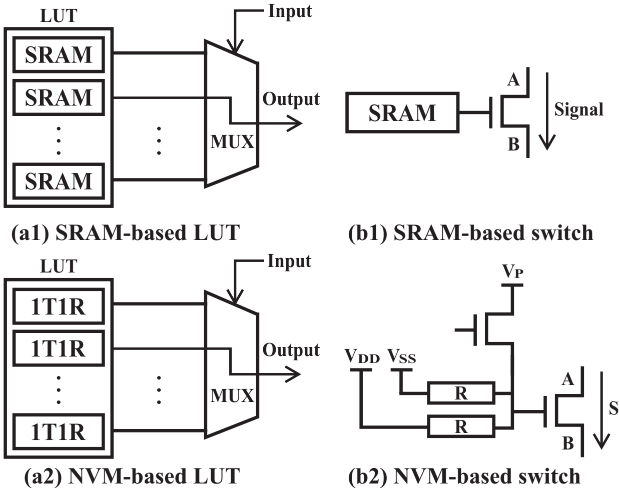

Fig. 2. NVM integration methods [4]

path. The placement method consists of SRAM-first initialization, slack-aware block selection for the swapping, and cost function considering energy consumption on the hybrid FPGA. Through the proposed placement, we were able to achieve lower energy consumption on the SRAM-NVM hybrid FPGAs than that on the SRAM-based FPGAs. The contributions of this paper can be summarized as follows:

- We suggest a SRAM-NVM hybrid FPGA as a low energy-consumption FPGA architecture.
- We propose an energy-efficient placement considering the SRAM-NVM hybrid FPGA.
- We implement and evaluate the proposed methods on the verilog-to-routing (VTR) CAD tool, which is a well-known CAD tool for FPGA.

II. Background and Related Work

A. SRAM-NVM Hybrid FPGAs

Island-style FPGAs have been widely used for commercialized products [2]. Typically, FPGAs of this type consist of many logic and routing components. The logic component has CLBs with several LUTs. The routing component is composed of SB and CB, which is known to take up a significant area of the FPGAs [3]. Each component has storage for its configuration.

Currently, most commercial FPGAs have adopted SRAM as their storage element for configurations and data. However, the SRAM-based FPGAs have suffered from large static power consumption and low area density. To overcome the limitations of SRAM-based FPGAs, NVM-based FPGAs have been proposed [3], [5], [6], [7], [8], [10], [11] with the development of emerging NVMs [9].

NVM-based FPGAs are considered one of the next-generation FPGAs for their low static power consumption, high area density, and non-volatility. NVM-based FPGAs can be divided into three categories depending on how the NVM is integrated into the FPGA. The first method is replacing the SRAM with NVM which is the most intuitive way to integrate NVM into an FPGA [5]. As shown in Fig. 2-(a1) and (a2), SRAM can be replaced with an NVM-based 1T1R (one transistor and one resistor) cell. This method is suitable for memory element arrays such as BRAM and LUT because of its high area density and high scalability.

The second method is to replace a SRAM and a transistor with an NVM-based switch to efficiently replace SRAM in routing components, such as SB and CB with NVM. As depicted in Fig. 2-(b1) and (b2), the SRAM and pass transistors in SB and CB are replaced by an NVM-based switch (2T2R). Although scalability and area density are relatively low, the switch has an advantage in delivering signals without significant speed degradation compared to that of SRAM.

The last method is using both SRAM and NVM together so that CLBs can take advantage of SRAM and NVM at the same time [10], [11], [20]. In [11], the authors introduced a framework for hybrid FPGAs including a SRAM-STTRAM hybrid FPGA architecture. [20] proposed a hybrid FPGA containing SRAM- and RRAM-based CLBs with a placement algorithm for efficient dynamic reconfiguration. [10] proposed a hybrid LUT supporting both volatile and non-volatile input based on magnetic domain-wall racetrack memory.

To date, there have been many studies on FPGAs using various types of NVM, such as RRAM [3], [5], [6], [7], [10], STTRAM [11], and racetrack memory [10]. Among them, RRAM-based FPGAs have been actively researched because of the features of RRAM, such as, CMOS-compatible process, small area, and low latency. Thus, we adopted RRAMs as NVMs in this paper.

B. Placement Algorithms

The placement is a process of mapping a netlist to physical blocks in the target FPGAs. The general objectives of the placement algorithms are minimizing total interconnect length and delay for minimizing the mapped area and critical path delay while guaranteeing a successful routing [12]. The placement is a well-known NP-complete problem. SA has been widely used for solving the optimization problem. The cost function of SA for the placement can be formulated as follows:

$$\min C = \min\{\alpha C_{\mathrm{b}} + \beta C_{\mathrm{t}}\}, \tag{1}$$

where C_{b} is the bounding box cost, C_{t} is the timing cost. α and β are scaling parameters for the bounding box and delay cost, respectively.

Due to the flexibility of cost function in SA, various objectives have been studied based on Eq. 1. A fault-tolerant placement was proposed to increase the probability of recovery using a spare block [13] . In [14], a hotspot-driven placement was introduced to lower the peak temperature and make thermal distribution uniform. The placement method in [15] considered the process variation. [16] reduced dynamic power consumption using the accurate net capacitance model. These placement methods successfully achieved their specific goals on SRAM-based FPGAs by modifying a cost function accordingly.

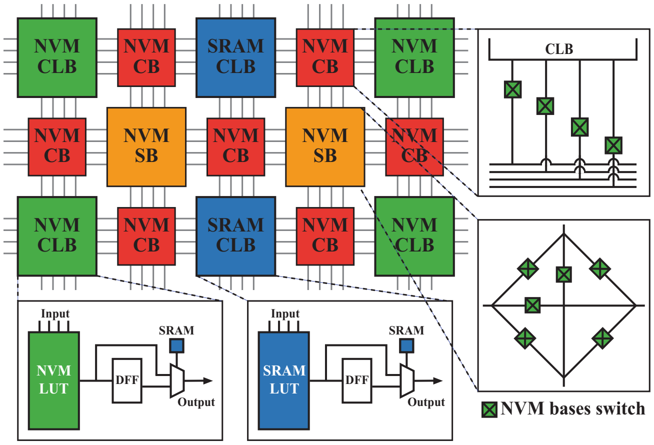

Fig. 3. SRAM-NVM hybrid FPGA architecture

As NVM technology advances, many studies about the placement algorithms of NVM-based FPGAs have been published. Unlike SRAM-based placement, placement algorithms for NVM-based FPGAs have mainly focused on improving the endurance of NVMs. Placement methods were proposed to increase the write endurance of NVM-based CLBs when reconfiguration is frequently performed at runtime [17], [18]. They improved the life time of FPGAs by utilizing the write count for each CLB and performing placement to manage the write endurance. [19] was able to improve the endurance of NVM-based BRAMs as well as NVM-based CLBs [19] without significantly increasing the critical path.

Furthermore, SRAM-NVM hybrid FPGA placement has been proposed in [20] to take advantage of both SRAM and NVM. Their methods have successfully reduced the reconfiguration time by considering both critical path and reconfiguration time in the hybrid FPGAs with SRAM- and NVM-based CLBs. The shortcoming of their method, however, is that they failed to take full advantage of hybrid FPGAs such as reduction of energy consumption.

III. Methods

In this paper, we suggest SRAM-NVM hybrid FPGA architecture which can exploit the advantages of both SRAM and NVM. In addition, we propose energy-aware placement algorithm for the SRAM-NVM hybrid FPGAs.

A. SRAM-NVM Hybrid FPGA Architecture

The operating speed of the FPGA is determined by the critical path, which has a significant impact on the energy required for a certain application. Thus, it is important to reduce power consumption while maintaining the critical path delay for reducing energy consumption. In the previous NVM-based FPGA research, static power consumption was reduced due to the low static power of NVM. NVMs on the critical path, however, increased the critical path delay, making it difficult to improve energy consumption. Thus, in this paper, we used SRAM-NVM hybrid FPGA architecture to reduce the energy consumption with the low power consumption of NVM and the high speed of SRAM.

As conventional island-style FPGAs, the hybrid FPGA architecture mainly consists of CLB, SB, and CB as in Fig. 3. The logic component, represented by the CLBs, implements the desired logic function by storing the Boolean function into the LUTs. The routing component, represented by SB and CB, connects the CLBs by programmable switches. These logic and routing components have storage elements for the configuration of the FPGA. NVM-based FPGAs are implemented by using NVM for these storage elements.

In this paper, different NVM integration methods were used for the hybrid FPGAs depending on routing and logic components as in [5]. For the routing components, such as CB and SB, we replaced SRAM and the pass transistor with an NVM-based switch. For the flexibility of connection, the routing elements have many SRAMs, which takes significant amounts of area, energy consumption, and delay [4]. The delay of routing elements, naturally, has a significant effect on the critical path delay. Thus, for the routing elements, the delay is more significant than area density for reducing energy consumption. In this regard, we replaced the SRAM and pass transistor with an NVM-based switch (2T2R) as [6] in the routing elements, which is depicted in Fig. 3.

For the logic components, such as CLBs, NVM-based LUTs were used to integrate NVM into logic components. In addition, to exploit the advantages of both SRAM and NVM, we used SRAM- and NVM-based CLBs as in [20]. As depicted in Fig. 3, SRAM- and NVM-based CLBs were arranged in columns. We can reduce the power consumption without significantly increasing the critical path delay by placing logic elements in critical and non-critical paths to SRAM- and NVM-based CLBs, respectively. Using the SRAM-NVM based FPGA as shown in Fig. 3, the fast access speed of SRAM and the low power characteristic of the NVM can be utilized to run the FPGA with low energy consumption. In order to utilize the hybrid FPGA, however, a placement method is required that can consider the hybrid FPGA architecture and energy consumption simultaneously.

B. Energy-aware Placement

The placement of SRAM- or NVM-based FPGAs with homogeneous CLBs is mainly required to minimize the critical path delay. However, in SRAM-NVM hybrid FPGAs with heterogeneous CLBs, we need to consider the type of mapped CLBs for reducing the energy consumption. At the same time, the placement for the hybrid FPGAs should not increase the critical path delay, which has a significant impact on the performance and energy consumption of FPGAs. In addition, we can represent a netlist as a directed acyclic graph $G = (V, E)$, where BLEs and nets in the netlist correspond to the vertex set V and edge set E, respectively. Thus, we can set an objective of placement for minimizing the energy consumption as follows:

$$\min \sum_{v \in V} P_v = \min \sum a_{x,y} m_{x,y}, \qquad (2)$$

$$\text{subject to} \quad d(e) \leq T \quad \text{for all } e \in E, \qquad (3)$$

Algorithm 1: Energy-aware Placement (EAP)

```
Input: Hybrid FPGA layout, netlist, $r_{\text{sel}}$: random
  selection probability for swapping candidate block
Output: Placement result $M$
$C_{\text{b}}(M)$, $C_{\text{t}}(M)$, and $C_{\text{e}}(M)$: bounding box, timing,
  and energy cost of placement $M$, respectively

// initial placement
$M \leftarrow$ SRAM-first initial placement
$T \leftarrow$ initial temperature of simulated annealing

while $!\text{exit}(\text{T})$ do
    for $i = 1\,\text{to}\,num\_swap\_limit$ do
        // select and swap blocks
        $r$ = uniform_random(0,1)
        if $r < r_{sel}$ then
            $A \leftarrow$ randomly selected block in $M$
        else
            $A \leftarrow$ block with largest slack in $M$
        $A' \leftarrow$ randomly selected block not in $M$
        $M' \leftarrow M$ with swapping $A$ and $A'$

        // cost calculation
        $\Delta C_{\text{b}} = C_{\text{b}}(M') - C_{\text{b}}(M)$
        $\Delta C_{\text{t}} = C_{\text{t}}(M') - C_{\text{t}}(M)$
        $\Delta C_{\text{e}} = C_{\text{e}}(M') - C_{\text{e}}(M)$
        $\Delta C = \alpha \Delta C_{\text{b}} + \beta \Delta C_{\text{t}} + \gamma \Delta C_{\text{e}}$

        $r$ = uniform_random(0,1)
        if $\Delta C < 0$ or $r < e^{-\Delta C/T}$ then
            $M \leftarrow M'$ // update placement
            update $C_{\text{b}}$, $C_{\text{t}}$, $C_{\text{e}}$, and $C$
    update $T$
return $M$
```

where P_v is a power consumption of vth logic element in the netlist, (x, y) is a coordinate on 2D FPGA (integer), $a_{x,y}$ is a power consumption of CLBs, $m_{x,y}$ is a placement result, d is a delay function between CLBs, and T is a given critical path delay.

The power consumption of CLBs $a_{x,y}$ is defined depending on the type of storage element in the CLBs. The placement result $m_{x,y}$ can be represented in binary form depending on whether the physical logic block of the coordinate is mapped or not. If the physical block is mapped, then $m_{x,y}$ is set to 1, otherwise 0. The distance function between two blocks (m_{x_1,y_1} and m_{x_2,y_2}) is defined as $d(m_{x_1,y_1}, m_{x_2,y_2}) = d_x|x_1 - x_2| + d_y|y_1 - y_2|$, where d_x and d_y are delay constants of each coordinates. Based on these functions, we can reduce integer linear programming (ILP) to the optimization problem (Eq. 2), which indicates that the placement for minimizing energy consumption is NP-complete problem.

For a practical solution for reducing the energy consumption on the SRAM-NVM hybrid FPGAs, we proposed energy-aware placement (EAP) based on SA (Algorithm 1. The objective of EAP is mapping logic blocks on a critical and a non-critical path to SRAM- and NVM-based CLBs, respectively, maintaining the critical path delay. EAP consists of mainly three parts: 1) SRAM-first placement initialization, 2) slack-aware block selection, and 3) cost function considering the energy consumption of the hybrid CLBs.

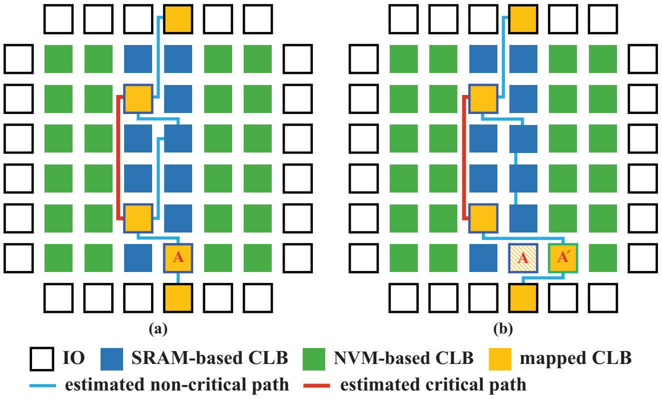

Fig. 4. Energy-aware placement on SRAM-NVM hybrid FPGAs (a) SRAM-first initial placement (b) slack-aware block selection

We assumed that a critical path would be on SRAM-based CLBs for guaranteeing a short critical path delay after placement. Based on this assumption, we introduced a SRAM-first initial placement method that can reduce the amount of computation in cost function and find a better optimization solution. This approach maps logic blocks in a netlist to SRAM-based CLBs as preferentially as possible.

With the initialization method, we proposed a slack-aware block selection scheme for the block swapping. The proposed method selects the mapped block with the largest slack, which is the difference between the critical path delay and the delay required for each mapped block, as the candidate block for swapping. This method exploits slack for mapping the logic blocks in a netlist to NVM-based CLBs as much as possible without increasing the critical path delay. With only this approach, we cannot explore the search space for minimizing the critical path. In order to reduce power consumption and the critical path simultaneously, we introduced the stochastic method that selects randomly mapped blocks with a probability r_{sel} during swapping block selection.

As a determining criterion of the swapping, we proposed an energy cost function as follows:

$$C_{\text{e}} = (P_S N_S + P_N N_N)/(N_S + N_N), \tag{4}$$

where P_S and P_N are power consumption of SRAM- and NVM-based CLBs, respectively, and N_S and N_N are the number of mapped SRAM- and NVM-based CLBs, respectively. Under the general assumption that NVM typically consumes less power than SRAM considering both dynamic and static power, this cost function encourages the mapped CLB of a logic block to be swapped from SRAM- to NVM-based CLB.

The cost function of EAP should consider critical path delay and routability as well as energy consumption. Thus, we included the energy cost to Eq. 1 as follows:

$$C = \alpha C_{\text{b}} + \beta C_{\text{t}} + \gamma C_{\text{e}}, \tag{5}$$

where γ is a scaling parameter of energy cost. An example of EAP is depicted in Fig. 4. The SRAM-first initial placement

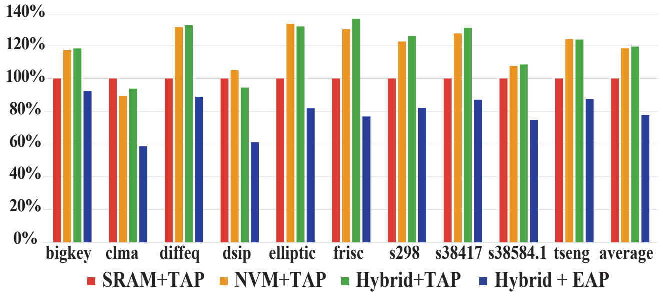

Fig. 5. Normalized energy consumption of various configurations on the MCNC benchmarks

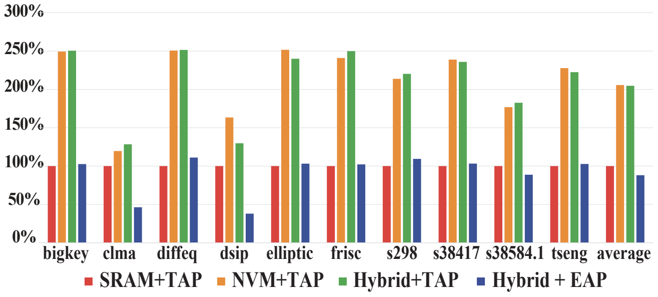

Fig. 6. Normalized critical path of various configuration on the MCNC benchmarks

is illustrated in Fig. 4-(a). As a result of the initialization method, the logic blocks in the netlist (yellow box) are mapped to SRAM-based CLBs (blue box). After the initialization, the block with the largest slack A (light yellow box) in the placement M is selected as the swapping candidate block (Fig. 4-(b)). If another candidate block A' (yellow box) is selected as in Fig. 4-(b), the energy cost and total cost (Eqs. 4 and 5) can be decreased. Consequently, the block swapping (A and A') is accepted without changing the critical path (red line), which results in a placement with improved energy consumption.

IV. Experiments

A. Experimental Methodology

To evaluate EAP on the SRAM-NVM hybrid FPGAs, we implemented the proposed placement algorithm and the hybrid FPGAs in VTR [21], which is a well-known CAD flow simulator. The SRAM-NVM hybrid FPGAs are based on the Altera Stratix IV in the VTR. We set the width and height of the hybrid FPGAs to 128. To minimize the area of hybrid FPGAs, we set the number of SRAM-based CLBs to 10% of the total number of CLBs in the FPGAs.

For the routing element (SBs and CBs), we adopted an NVM-based switch (2T2R) as in [6]. To model the NVM-based LUTs in the hybrid FPGA, we used the parameters simulated by NVSIM [22]. Table I shows the parameters used in SRAM- and NVM-based LUTs with six inputs.

We evaluated the proposed placement method on Microelectronics Center of North Carolina (MCNC) and VTR benchmarks [21]. We measured the critical path delay and energy consumption, including dynamic and static energy. We empirically set α, β to 0.5, and γ to 0.01 in Eq. 5. We set the random selection probability for swapping candidate block r_{sel} to 0.1. For accurate experimental results, we averaged ten experiments on each benchmark.

B. Experimental Results

We measured the energy consumption and critical path delay of MCNC and VTR benchmarks on SRAM-based, NVM-based, and SRAM-NVM hybrid FPGAs. We applied the commonly used timing-aware placement (TAP) to these FPGA architectures as done in [21]. The proposed placement (EAP) considering SRAM-NVM hybrid CLBs was applied to the hybrid FPGA. The experimental results on MCNC benchmarks are depicted in Fig. 5 and 6.

TABLE I
Parameters of SRAM- and RRAM-based LUTs

Parameters	SRAM	RRAM
Read latency (ns)	0.16671	0.86445
Read dynamic energy (pJ)	0.28160	1.01252
Static power (mW)	1.65865	0.03585
Area (μm^2)	0.00195	0.00012

The NVM-based FPGA can reduce power consumption, which may not lead to decreasing the energy consumption because of the slow access speed (Table I). As illustrated in Fig. 5, the average energy consumption of NVM-based FPGA was increased by 18.38% on the MCNC benchmarks compared to that of SRAM-based FPGA. Despite the low power consumption of the NVM, this increase in energy consumption is due to a dramatic increase in the critical path delay. As described in Fig. 6, the critical path delay of NVM-based FPGA was increased more than twice on average compared to that of SRAM-based FPGA. The energy consumption was reduced only in the *clma* benchmark with about 20% increase in the critical path delay.

SRAM-NVM hybrid FPGAs can exploit the advantages of both the low power consumption of NVM and the fast access speed of SRAM. However, the hybrid FPGA was not able to achieve energy reduction with the conventional TAP. Even though the critical path delay was decreased by approximately 1% (Fig. 6), the energy consumption of SRAM-NVM hybrid FPGA was increased by roughly 1% on average compared to that of NVM-based FPGA (Fig. 5). This result reveals that the TA placement is not suitable for the hybrid FPGAs. Compared to the SRAM-based FPGA, the hybrid FPGA consumed 19.48% more energy on average.

In contrast to TAP, the proposed EAP was able to reduce the energy consumption on the hybrid FPGA. We achieved 34.90% reduction in energy consumption compared to that of TAP on the hybrid FPGA. The critical path delay was reduced by 57% compared to that of TAP, which indicates that the EAP can utilize the heterogeneity of the hybrid FPGA. Compared to the case of SRAM-based FPGA with TAP, we were able to reduce the energy consumption and the critical path delay by 22.23% and 11.94%, respectively. The experimental results showed that the proposed EAP found a better optimization solution, reducing not only power consumption but also delay.

TABLE II
EXPERIMENTAL RESULTS ON VTR BENCHMARKS

	Critical path delay (ns)				Energy (nJ/cycle)				Area (gain)[a]		Slack[b]
FPGA architecture	SRAM	NVM	Hybrid	Hybrid	SRAM	NVM	Hybrid	Hybrid	Hybrid	Hybrid	Hybrid
Placement algorithm	TAP	TAP	TAP	EAP	TAP	TAP	TAP	EAP	TAP	EAP	EAP
blob_merge	**10.10**	28.76	29.79	10.47	4.24	6.30	6.64	**3.21**	**6.21**	**6.21**	0.45
boundtop	6.68	9.49	7.95	**2.16**	2.73	2.67	2.49	**1.62**	7.68	**11.84**	0.42
ch_intrinsics	5.68	8.70	9.53	**2.49**	2.62	2.62	2.78	**1.70**	7.42	**16.25**	0.37
diffeq1	**23.31**	39.88	40.59	23.39	6.63	7.29	7.51	**5.02**	**4.59**	3.38	0.54
diffeq2	**19.02**	34.52	33.29	19.13	5.44	6.39	6.38	**4.21**	10.52	**16.25**	0.57
mkDelayWorker32B	13.51	19.81	17.87	**8.10**	9.59	9.97	9.60	**7.37**	6.25	**15.25**	1.40
mkPktMerge	4.78	5.98	5.66	**4.52**	4.12	3.98	3.95	**3.72**	7.63	**16.25**	0.76
mkSMAdapter4B	8.08	13.20	11.84	**6.68**	4.00	4.17	3.93	**3.11**	5.46	**8.80**	0.53
spree	10.41	26.81	24.28	**10.40**	4.25	5.88	5.49	**3.34**	**16.25**	5.33	0.58
raygentop	5.90	12.56	12.79	**5.44**	2.79	3.24	3.33	**2.22**	**6.36**	5.73	0.54
sha	14.27	34.60	34.66	**13.94**	5.05	7.02	7.04	**3.59**	**6.22**	5.59	0.42
or1200	15.35	42.64	41.61	**15.09**	6.54	9.16	9.24	**4.44**	**6.83**	6.66	0.45
bgm	**18.80**	41.47	41.14	31.52	**8.43**	10.82	10.84	8.85	6.68	**7.38**	0.62
LU8PEEng	**81.95**	221.78	212.56	85.70	38.37	65.31	64.33	**29.38**	7.19	**7.52**	0.81
average	**16.99**	38.58	37.40	17.07	7.48	10.34	10.25	**5.84**	7.52	**9.46**	0.66

[a] compared to the area of SRAM (based on Table I)

[b] normalized to the total slack of Hybrid + TAP

The experimental results on the VTR benchmarks are depicted in Table II. The results showed that the EAP on the hybrid FPGA was able to reduce the energy consumption by 21.94% compared to that of TAP on the SRAM-based FPGA on average. The area gain of TAP and EAP on the hybrid FPGA are 7.52 and 9.46 times compared to that of SRAM-based FPGA, respectively. In addition, the slack of EAP on the hybrid FPGA was reduced to 0.66 of that of TAP on the hybrid FPGA. Considering both the gain of area and slack, we validated the effectiveness of the proposed EAP on the SRAM-NVM hybrid FPGA.

V. CONCLUSION

In this paper, we suggested the SRAM-NVM hybrid FPGAs as low energy consumption architectures. In addition, for exploiting the heterogeneity of SRAM and NVM on the hybrid FPGAs, we proposed an energy-aware placement on the hybrid FPGAs. The proposed placement encourages logic blocks in a netlist to be mapped to NVM-based CLBs rather than SRAM-based ones. We validated the effectiveness of the proposed methods by various experiments results on the MCNC and VTR benchmarks. We expect that the proposed placement method can be applied to other heterogeneous FPGAs to reduce the energy consumption.

ACKNOWLEDGMENT

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korean government (Ministry of Science and ICT) [2016M3A7B4911115, 2018R1A2B3001628], the Brain Korea 21 Plus Project in 2019, and Samsung Electronics (DS and Foundry).

REFERENCES

[1] S. Park *et al.*, "Energy-efficient inference accelerator for memory-augmented neural networks on an fpga," in *DATE*, 2019, pp. 1587–1590.

[2] V. Betz *et al.*, *Architecture and CAD for deep-submicron FPGAs*. Springer Science & Business Media, 2012, vol. 497.

[3] P. Gaillardon *et al.*, "Design and architectural assessment of 3-d resistive memory technologies in fpgas," *IEEE TNANO*, vol. 12, no. 1, pp. 40–50, 2012.

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

[5] W. Wang *et al.*, "Fpga based on integration of memristors and cmos devices," in *ISCAS*, 2010, pp. 1963–1966.

[6] S. Tanachutiwat, M. Liu, and W. Wang, "Fpga based on integration of cmos and rram," *IEEE TVLSI*, vol. 19, no. 11, pp. 2023–2032, 2010.

[7] J. Cong and B. Xiao, "Fpga-rpi : A novel fpga architecture with rram-based programmable interconnects," *IEEE TVLSI*, vol. 22, no. 4, pp. 864–877, 2013.

[8] K. Huang *et al.*, "High-density and high-reliability nonvolatile field-programmable gate array with stacked 1d2r rram array," *IEEE TVLSI*, vol. 24, no. 1, pp. 139–150, 2015.

[9] S. Yu and P.-Y. Chen, "Emerging memory technologies: Recent trends and prospects," *IEEE J. Solid-State Circuits*, vol. 8, no. 2, pp. 43–56, 2016.

[10] K. Huang *et al.*, "Racetrack memory based hybrid look-up table (lut) for low power reconfigurable computing," *J. Parallel and Distributed Computing*, vol. 117, pp. 127–137, 2018.

[11] S. Paul *et al.*, "A circuit and architecture codesign approach for a hybrid cmos–sttram nonvolatile fpga," *IEEE TNANO*, vol. 10, no. 3, pp. 385–394, 2010.

[12] A. Marquardt *et al.*, "Timing-driven placement for fpgas," in *FPGA*, 2000, pp. 203–213.

[13] A. Agarwal *et al.*, "Fault tolerant placement and defect reconfiguration for nano-fpgas," in *ICCAD*, 2008, pp. 714–721.

[14] W. Lu *et al.*, "Teshop: A temperature sensing based hotspot-driven placement technique for fpgas," in *FPL*, 2016, pp. 1–4.

[15] G. Lucas *et al.*, "Variation-aware placement with multi-cycle statistical timing analysis for fpgas," *IEEE TCAD*, vol. 29, no. 11, pp. 1818–1822, 2010.

[16] K. Vorwerk *et al.*, "A technique for minimizing power during fpga placement," in *FPL*, 2008, pp. 233–238.

[17] H. Zhang *et al.*, "Strap: Stress-aware placement for aging mitigation in runtime reconfigurable architectures," in *ICCAD*, 2015, pp. 38–45.

[18] Y. Xue *et al.*, "Age-aware logic and memory co-placement for rram-fpgas," in *DAC*, 2017, p. 1.

[19] S. Huai *et al.*, "Performance-aware wear leveling for block ram in nonvolatile fpgas," in *DAC*, 2019, p. 157.

[20] Q. Lou *et al.*, "Runtime and reconfiguration dual-aware placement for sram-nvm hybrid fpgas," in *NVMSA)*, 2017, pp. 1–6.

[21] J. Rose *et al.*, "The vtr project: architecture and cad for fpgas from verilog to routing," in *FPGA*, 2012, pp. 77–86.

[22] X. Dong *et al.*, "Nvsim: A circuit-level performance, energy, and area model for emerging nonvolatile memory," *IEEE TCAD*, vol. 31, no. 7, pp. 994–1007, 2012.