ARTICLES

Journal of Semiconductors
(2020) 41, 022406
doi: 10.1088/1674-4926/41/2/022406

Optimizing energy efficiency of CNN-based object detection with dynamic voltage and frequency scaling

Weixiong Jiang[1, 2, 3], Heng Yu[4], Jiale Zhang[1, 2, 3], Jiaxuan Wu[1, 2, 3], Shaobo Luo[5], and Yajun Ha[1, 2, 3, †]

[1]School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China

[2]Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai 200050, China

[3]University of Chinese Academy of Sciences, Beijing 100049, China

[4]University of Nottingham Ningbo China, Ningbo 315100, China

[5]Universite Paris-Est, Paris 93162, France

Abstract: On the one hand, accelerating convolution neural networks (CNNs) on FPGAs requires ever increasing high energy efficiency in the edge computing paradigm. On the other hand, unlike normal digital algorithms, CNNs maintain their high robustness even with limited timing errors. By taking advantage of this unique feature, we propose to use dynamic voltage and frequency scaling (DVFS) to further optimize the energy efficiency for CNNs. First, we have developed a DVFS framework on FPGAs. Second, we apply the DVFS to SkyNet, a state-of-the-art neural network targeting on object detection. Third, we analyze the impact of DVFS on CNNs in terms of performance, power, energy efficiency and accuracy. Compared to the state-of-the-art, experimental results show that we have achieved 38% improvement in energy efficiency without any loss in accuracy. Results also show that we can achieve 47% improvement in energy efficiency if we allow 0.11% relaxation in accuracy.

Key words: CNN; FPGA; DVFS; object detection

Citation: W X Jiang, H Yu, J L Zhang, J X Wu, S B Luo, and Y J Ha, Optimizing energy efficiency of CNN-based object detection with dynamic voltage and frequency scaling[J]. *J. Semicond.*, 2020, 41(2), 022406. http://doi.org/10.1088/1674-4926/41/2/022406

1. Introduction

FPGA has become a promising platform for edge computing in recent years, given its energy efficiency compared to GPU and flexibility compared to ASICs[1]. There are many previous works such as[2–13] that present various methods of optimizing architecture or design flow. However, none of them specifically optimize energy efficiency. Dynamic voltage and frequency scaling (DVFS) has been extensively applied as a system-level methodology to optimize the system execution. Through judiciously scaling up or down the execution voltage and speed of the processing units, DVFS effectively achieves throughput maximization, temperature management, application quality maximization, and energy minimization. Although the delay of the combinational logic will increase as the voltage decreases, which may bring a certain chance of error during operation, the CNN itself is robust to errors[4–17]. Even if a small number of neurons get wrong values in the process of inference, it does not affect the final accuracy. In this context, combining CNN and DVFS is an ideal method to extrude the potential of FPGAs to optimize either performance or energy efficiency.

To add the DVFS support to CNN accelerators, a flexible DVFS platform is needed. Previous works have focussed on adding the DVFS support to commercial FPGAs, but their solutions for DVFS have various limitations. For example, the DVFS module consumes too much logic resources and thus affects timing and power[18]. The scaling resolution is not high enough, or the scaling time is too long[19]. To solve these issues, we develop a novel DVFS framework with high resolution and flexibility as well as low area overhead and scaling time to implement a DVFS system quickly. Our main innovative technical contributions are as follows:

(1) We propose a framework for fine-grained DVFS on state-of-the-art FPGA devices.

(2) We combine the framework with SDSoC and then apply it to SkyNet, a lightweight CNN for object detection.

(3) We analyze the impact of DVFS on performance, power, energy efficiency, and accuracy.

(4) We achieve 54% improvement in performance, 38% improvement in energy efficiency, and 106% improvement in unified energy efficiency (UEE) without any loss in accuracy compared to the original SkyNet. If we relax the requirement on the accuracy, we can achieve 56% improvement in performance, 47% improvement in energy efficiency, and 121% improvement in UEE at the cost of 0.11% decrease in intersection over union (IoU).

(5) We develop a DVFS policy basing on the measured metrics targeting on real-time applications in realistic scenarios. With this DVFS policy, the average power has been reduced by 30% compared to the original design.

Section 2 presents the related works on CNN and DVFS. Section 3 defines a power optimization problem in the scenario of edge computing. Section 4 introduces the DVFS framework as well as the DVFS policy for energy efficiency optimization. Section 5 describes how to combine the DVFS framework with SDSoC and the architecture of the system after applying DVFS to CNN accelerators. Section 6 gives the experi-

Correspondence to: Y J Ha, hayj@shanghaitech.edu.cn
Received 17 SEPTEMBER 2019; Revised 13 NOVEMBER 2019.

mental results and analyzes the impact of DVFS on the original CNN accelerator. It also presents the experimental results of the DVFS policy. Finally, Section 7 concludes the paper.

2. Related work

2.1. CNN

For edge applications, there are many previous works on optimizing the architecture of the FPGA-based CNN accelerator. Ma[4] presents an RTL-level CNN compiler that automatically generates customized FPGA hardware for the inference tasks of various CNNs, in order to enable high-level fast prototyping of CNNs from software to FPGA. A programmable and flexible CNN accelerator architecture together with a data quantization strategy and compilation tool is introduced in Ref. [8]. The authors of Ref. [6] present an efficient hardware accelerator design of deep residual learning algorithms. In Ref. [5], they quantitatively analyze and optimize the design objectives of the CNN accelerator based on multiple design variables. In Ref. [7], an architecture named tile-grained pipeline architecture (TGPA) for low latency CNN inference is proposed. Wei[20] proposes a layer conscious memory management framework for FPGA-based CNN hardware accelerators.

Recently, software–hardware co-design has gained more and more attention. Ding[21] proposes REQ-YOLO, a resource-aware, systematic weight quantization framework for object detection, considering both algorithm and hardware resource aspects in object detection. Zhang[22] and Hao[23] raise a novel and practical bi-directional co-design approach, including a bottom–up DNN model design strategy together with a top–down flow for DNN accelerator design. It enables a joint optimization of both DNN models and their deployment configurations on FPGAs. Also, Hao[23] builds an automatic co-design flow, including an Auto-DNN engine to perform a hardware-oriented DNN model search, as well as an Auto-HLS engine to generate synthesizable C code of the FPGA accelerator for explored DNNs. However, all of the works mentioned above ignore specific energy efficiency optimization.

Nunez-Yanez[24] proposes an energy proportional framework with adaptive voltage and frequency scaling and apply it to binary neural networks (BNN) for classification. However, BNN has very limited precision and is almost unavailable in practical applications. What is more, Nunez-Yanez[24] lacks exploration of actual scenes in an edge computing context. In this paper, we combine our fine-grained DVFS framework with CNN based object detection accelerators to perform specific optimization on energy efficiency. Besides, we define two kinds of actual scenarios in edge computing and perform specific optimization. To demonstrate the effectiveness of DVFS, we choose SkyNet[25] as a case study. SkyNet won the championship in Design and Automation Conference-System Design Contest 2019 (DAC-SDC2019), a low power object detection challenge in images captured by unmanned aerial vehicles (UAVs). SkyNet delivers 0.716 IoU and 25.05 FPS on an Ultra96 FPGA. The reason why we choose SkyNet as a case study is that SkyNet represents the highest level of CNN implementation on FPGAs considering performance and energy efficiency. We want to develop a DVFS based method to further tap its potential and achieve better improvement in both performance and energy efficiency.

2.2. DVFS

DVFS has been proven to be a popular and efficient system-level methodology to optimize system execution metrics, such as throughput, power, energy, temperature, reliability, and QoS. For example, Weissel[26] proposes to adjust the frequency in response to application behavior changes, to optimize energy efficiency for general-purpose CPU systems. The authors of Ref. [27] empirically reveal that on a commercial smart phone platform, energy consumption strongly correlates its CPU frequency, and exhibits an optimal operating frequency for energy minimization. Huang *et al.*[28] propose to maximize throughput and prevent system overheating by carefully interleaving the hot and cool tasks and adjusting the task execution frequencies. QoS maximization through efficient DVFS can be found in Refs. [29, 30]. By leveraging task adaptability, the output quality can be dynamically adjusted under temperature constraints, and judiciously applying DVFS can maximize the quality. To enhance system reliability, the authors propose to carefully tune the frequency of CPUs to control temperature overflow to minimize the thermal cycling that stresses chips[31]. While the evaluation of DVFS efficacy on those works is largely based on simulations, a realistic platform like ours that enables quick system synthesis and DVFS algorithm evaluation is highly desirable.

There have been existing works combining CNN and DVFS on ASICs, CPU, or GPU. Motamedi[12] develops a principled approach and a data-driven analytical model with DVFS to optimize the granularity of threads during CNN software synthesis. Bong[32] proposes a low-power CNN-based face recognition system for user authentication in smart devices with DVFS. Santoro[33] uses a performance-power analytical model fitted on a parameterized implementation of a CNN accelerator in a 28-nm FDSOI technology to explore large design space and to obtain the Pareto points that maximize the effectiveness of DVFS in the sub-space of throughput and energy efficiency. Our work is the first one to combine CNN-based object detection with DVFS on FPGA to our best knowledge.

On the hardware aspect, many DVFS supported logic and systems have been developed. Widely adopted frequency scaling approaches include phase-locked-loop[34] and delay-locked-loop[35]. Brynjolfson and Zilic propose a clock management scheme named DPCP on FPGA platforms[36]. To produce dynamically scaled frequency, state-of-the-art FPGA platforms (Xilinx 7 Series or later) adopt a hybrid mixed-mode clock manager (MMCM) + PLL architecture. The authors of Ref. [37] propose performance scaling on Virtex-7, where DFS is realized by employing an external programmable oscillator controlled by a Picoblaze processor. In addition to works on traditional FPGAs, there exist other works that implement system-level optimization on ZYNQ using DVFS. Beldachi[38] raises a method for accurate power control and monitoring on ZYNQ device, Hosseinabady[39] investigates the viability of physical power gating FPGA devices that incorporate a hardened processor in a different power domain, but they only implemented several computing units such as fp32mult, IIR, DCT, etc., rather than an accelerator. In order to solve these problems, in this paper we propose a DVFS framework with high scaling range and resolution as well as low scaling time. What is more, we combine the DVFS framework with

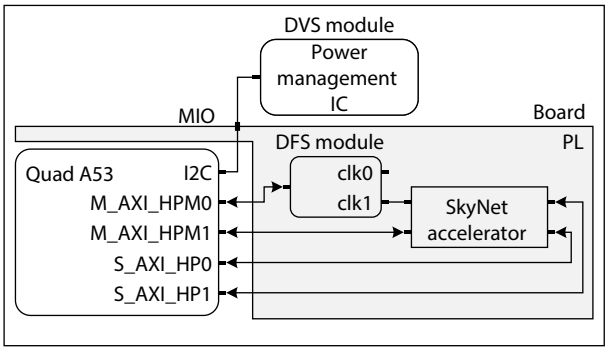

Fig. 1. System architecture of CNN accelerator.

high-level synthesis tool, SDSoC, making it possible to build a system with DVFS support in a complete software development flow.

3. Problem definition

Edge computing refers to an open platform that integrates network, computing, storage, and application core capabilities on the side close to the data source, providing near-end services. The edge computing platform's power supply conditions are far less than the cloud computing platform, which put higher requirements on the energy efficiency ratio of edge computing devices. For the object detection task in the edge computing scenario, there are generally two cases. The first application scenario pursues the least amount of energy required to process each frame of data. For example, in a home security scenario, only when the sensor detects a moving object, will the camera take a picture and wake up the edge computing platform to process the data. Otherwise, the edge computing platform can work in a sleep state. In the second application scenario, data is generated at a fixed rate, and the goal is to minimize the average power consumption throughout the work period. For instance, in autonomous driving, the camera produces images at a fixed rate, and the edge computing platform has to be running at all times. In this paper, we want to develop a DVFS method that significantly improve the energy efficiency of object detection on FPGA-based edge computing platform.

4. DVFS framework

Fig. 1 shows the system architecture of our DVFS framework, where DVS module is implemented by a power management IC (PMIC) that is responsible for controlling and monitoring the switching regulators. The switching regulators are connected to different components on the FPGA, such as PS, PL, IO ports, and BRAM. The DFS module is a clock generator that generates clock signals for the accelerators on the programmable logic. Our platform provides the capability of dynamic frequency and voltage scaling with high resolution, wide range, and low scaling time. Users can scale frequency from 20 to 400 MHz at a step size of 1 MHz in 3 μs and scale voltage from 650 to 850 mV at the step size of 10 mV in 2 ms. Besides, power monitoring APIs for various power rails on FPGA are also provided in our framework. The proposed DVS framework is coarse-grained because modern FPGAs do not support multiple voltages in different domains. The power supply of all the slices is connected together. Therefore it is impossible to implement the module-level DVS method on current FPGA devices up till now. In this section, we will introduce how the DVS and DFS are implemented in hardware and then discuss how to access them in Linux OS. The framework supports a series of Zynq 7000 and Zynq UltraScale+ series FPGA boards including ZC702, ZC706, ZCU102, etc. at the moment. To be specific, we use ZCU104 as an example in this paper.

4.1. Dynamic frequency scaling

The Zynq 7000 and Zynq UltraScale+ series FPGA mainly has two kinds of clock sources, FCLK on the PS and MMCM on the PL. The FCLK has the advantage of low area overhead and existing driver in Linux. However, it can only provide limited frequency points. Therefore, we choose MMCM as the clock generator in our design since as it is armed with a larger range and higher resolution. As shown in Fig. 2, the input clock is firstly multiplied and then divided by VCO in MMCM. The VCO output is further divided to generate different frequencies for different components. The output frequency can be calculated by the following equation,

$$F_{OUT} = F_{CLKIN} \times \frac{M}{D \times O}, \tag{1}$$

where M, D, and O can be configured through dynamic scaling port available in the MMCM blocks and new frequencies are generated at run-time. As shown in Fig. 3, the configuration information and the reset signal are sent to the MMCM via an AXI4-Lite interface and then saved in the register map. MMCM retrieves the configuration from the register map and resets to desired frequency accordingly.

Fig. 4 shows the driver for DFS. The clock generator is mapped to a device firstly so that it can be accessed in Linux userspace, then M, D, and O are calculated according to the target frequency. The clock input of the MMCM is connected to the FCLK on the PS, and the FCLK is set to 100 MHz. As shown in Eq. (1), the M can be the fractional number that is a multiple of 0.25. Thus we can divide the target frequency by 4.0. As shown in Eq. (1), the output frequency is $F_{CLKIN} \times M \div D \div O = 100 \times F \div 4.0 \div 5 \div 5 = F$ (MHz) and the resolution is 1 MHz. Benefit from high bandwidth and low latency on-chip bus between the PS and the PL, it takes only 3 μs for each frequency scaling operation.

4.2. Dynamic voltage scaling

The power management bus (PMBus) is an open standard power-management protocol, which is compatible with the I2C protocol at the physical level. This flexible and highly versatile standard allows for communication between devices based on both analog and digital technologies and provides true interoperability, which will reduce design complexity and shorten the time to market for power system designers. The-state-of-the-art FPGA boards usually adopt power regulators and a PMBus-compliant system controller to supply core and auxiliary voltages as shown in Fig. 5. For instance, ZC702 and ZC706 use the Texas Instrument (TI) UCD92x, ZCU100 and ZCU102 use the TI INA226, and ZCU104 uses the Infineon IRPS5401. All of the PMICs mentioned follow the same power management protocol, thus our DVS framework is compatible with a wide variety of FPGA boards with minimal changes. The power management IC(PMIC) gets configuration information from the FPGA through the I2C bus and con-

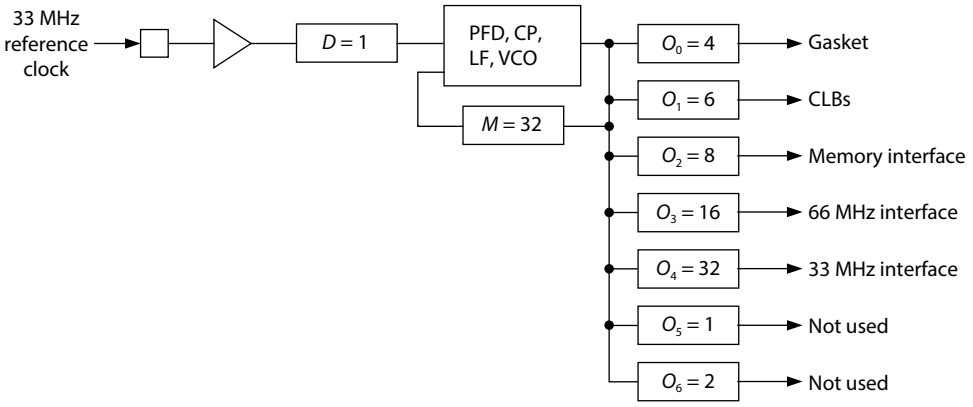

Fig. 2. MMCM.

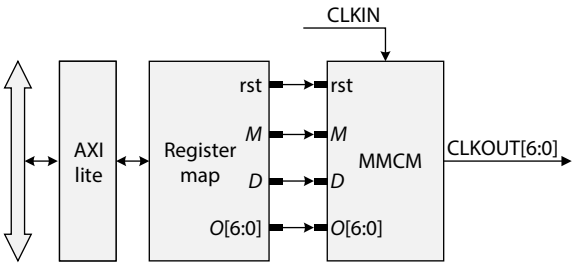

Fig. 3. Dynamically reconfiguring MMCM using AXI4-Lite interface.

```
#include <i2c-dev.h>
void set_voltage(int voltage)
{
    i2c_write(POWER_GOOD_OFF);
    i2c_write(POWER_GOOD_ON);
    i2c_write(VOUT_UV_FAULT_LIMIT);
    i2c_write(VOUT_UV_WARN_LIMIT);
    i2c_write(VOUT_MARGIN_LOW);
    i2c_write(VOUT_COMMAND);
    i2c_write(VOUT_MARGIN_HIGH);
    i2c_write(VOUT_OV_WARN_LIMIT);
    i2c_write(VOUT_MAX);
}
```

Fig. 4. Pseudocode of the driver for dynamic frequency scaling.

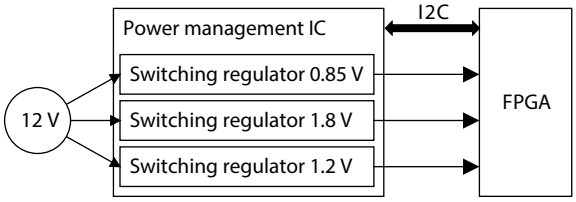

Fig. 5. Power Management Framework on FPGA.

trols the output voltage of each switching regulator. In the meantime, the PMIC monitors the voltage and the current of each switching regulator and reports the information back to the FPGA through the I2C bus.

As shown in Fig. 6, there are two IRPS5401s on the ZCU104 board, both of which are connected to PCA9548, which is an I2C 1-to-8 bus switch external to the device. Then PCA9548 is connected to the PS I2C controller via MIO. IRPS5401 can control and monitor up to ten voltage rails on board, including VCC12 and VCCINT. The former is the power input of the whole board and the latter is the PL side supply voltage. This gives us the possibility to scale the accelerator's

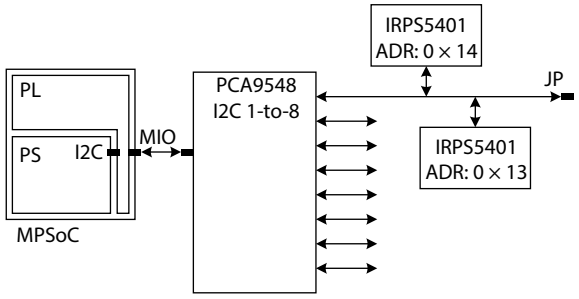

Fig. 6. Power monitoring system topology on ZCU104.

```
#include <i2c-dev.h>
void set_voltage(int voltage)
{
    i2c_write(POWER_GOOD_OFF);
    i2c_write(POWER_GOOD_ON);
    i2c_write(VOUT_UV_FAULT_LIMIT);
    i2c_write(VOUT_UV_WARN_LIMIT);
    i2c_write(VOUT_MARGIN_LOW);
    i2c_write(VOUT_COMMAND);
    i2c_write(VOUT_MARGIN_HIGH);
    i2c_write(VOUT_OV_WARN_LIMIT);
    i2c_write(VOUT_MAX);
}
```

Fig. 7. Pseudocode of the driver for dynamic voltage scaling.

voltage and frequency in real-time. Besides, IRPS5401 can also monitor the power consumption of each power rail and feedback the information through PMBus. There are two alternatives to communicate with the PMIC internally, as both the PL and the PS banks have access to the PMIC. The hardware method is to implement an AXI I2C controller in PL, while the software method connects the MIO pins directly to the PS. We chose the software method because not only does it not have any area overhead but it can also take advantage of the mature I2C driver embedded in the Linux kernel.

Fig. 7 demonstrates the driver for DVS. Since the PMBus protocol is compatible with the I2C protocol at the physical level, we can leverage the integrated I2C driver in Linux to communicate with the PMIC. The output voltage can be modified by writing PMBus commands to the PMIC in a fixed order according to the PMBus protocol. Because of the low speed of I2C and more commands to transfer, the voltage scaling takes about 2 ms each time. On the other hand, the power consumption can also be accessed in Linux in a similar way in which the PMIC is configured.

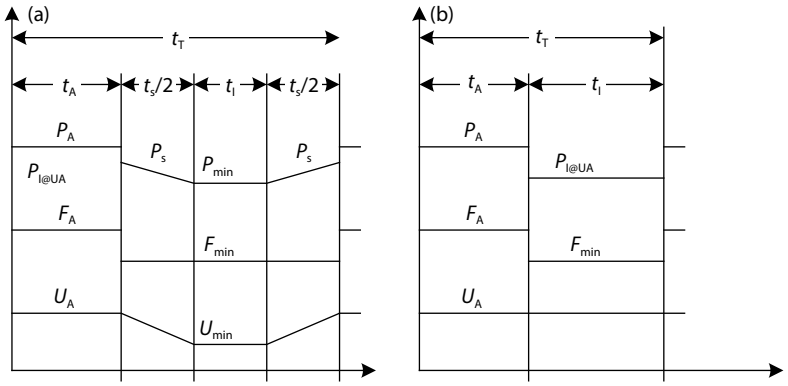

Fig. 8. (a) Timing diagram illustrating the DVFS policy with enough time to perform DVS. (b) Timing diagram illustrating the DVFS policy without enough time to perform DVS.

In summary, at the hardware level, the proposed DVFS framework takes advantage of the MMCM module on modern FPGAs, as well as the PMBus compliant power supply system. In the software level, we adapt the provided memory map and I2C drivers in Linux to control the DVFS module, making it easier to use. We take ZCU104 as an example in this paper, however, the DVFS framework can be easily ported to Zynq 7000 devices or even non-Zynq FPGA devices as long as the board is equipped with a configurable PMIC. For the non-Zynq FPGA devices, we can inherit the drivers for Zynq devices if a MicroBlaze, a RISC Harvard architecture soft processor provided by Xilinx, is adapted as the DVFS module controller and running Linux on it.

4.3. DVFS policy

For practical applications like real-time object detection, there is a constant frame rate of images obtained from a camera. In this context, it is reasonable to let the processing frame rate of the accelerator being consistent with the frame rate of the camera to save power consumption. If the peak processing frame rate of the accelerator is higher than that of the camera, there are two strategies, namely "long active, short idle" and "short active, long idle," to meet the performance requirements. The former is to set the accelerator to low frequency thus it takes a long time for the accelerator to process one frame and thus the idle time is short, while the latter is to set the accelerator to a high frequency and thus has a shorter active time.

To maximize energy savings, we can scale both the voltage and the frequency to a minimum when the accelerator is idle. And this makes it impossible to analyze which solution is the optimal one in the qualitative method. To quantitatively illustrate the problem, we define E_T, t_T, t_S, t_I, T_A, P_S, P_I, P_A, and P_M, where E stands for energy, P stands for power, t stands for time, T stands for total, S stands for scaling, I stands for idle, A stands for active and M stands for mean. Their relationship is shown in the following equations:

$$t_T = t_I + t_S + t_A, \tag{2}$$

$$E_T = t_I P_I + t_S P_S + t_A P_A, \tag{3}$$

$$P_M = E_T / t_T. \tag{4}$$

Fig. 8 shows the detailed steps of our DVFS policy. The two sub-figures show that the voltage, frequency, and power consumption vary with time differently under different t_T. In each period, the accelerator first runs at F_A and U_A and then scales to the minimum supported frequency F_{min} and the minimum supported voltage U_{min} to save static power once the computation is finished. According to our DVFS framework, frequency scaling takes 3 μs while voltage scaling takes 2 ms. This means that when the t_T is long enough, both frequency scaling and voltage scaling are performed. On this occasion, the chip's power consumption first drops to power-idle-at-active-voltage ($P_{I@UA}$) when frequency scaling is performed and then gradually decreases to P_{min} as the voltage gradually decreases to U_{min}. The power when the voltage is scaling (P_S) is given by the average of P_{min} and $P_{I@UA}$. However, if the t_T is short, only the frequency can be adjusted as shown in the right sub-figure. Our goal is to find a voltage/frequency combination that has a minimum of P_M with our DVFS policy.

5. Design flow

5.1. System architecture

Fig. 1 shows the topology details of the system. It is implemented on the latest Zynq UltraScale+ device that integrates processing system (PS) and programmable logic (PL). The PS is a Quad-Core A53 processor running at 1.2 GHz. It provides three M_AXI buses and six S_AXI buses for high throughput data transfer between the PL and the PS. What's more, the PS is also embedded with common interfaces for various peripherals, such as I2C, USB, Ethernet, Display Port, etc. The I2C interface is connected to the power management IC on the FPGA board via MIO. Two high-performance M_AXIs are enabled in this design, one of them is connected to the DFS module, and the other one is connected to the SkyNet accelerator. The PS is running Linux OS and is responsible for loading images and sends control signals such as target frequency of the DFS module and the configuration parameters for the SkyNet accelerator. All the clock signals related to the SkyNet accelerator are under the DFS module/clk1 domain including the accelerator's clock input as well as that of M_AXI_HPM1, S_AXI_HP0, S_AXI_HP1. All the input feature maps and weights are transferred through S_AXI_HP0 and all the output feature maps are transferred through S_AXI_HP1. The frequency and the voltage can only be modified during the iteration interval of the hardware calls.

5.2. SDSoC support

To build an ARM + FPGA system with DVFS support in

Table 1. Resource utilization of the system.

Resource	LUT	LUTRAM	FF	BRAM	DSP
SkyNet total	54639	1984	65196	209	333
Accelerator	49934	921	57101	209	333
AXI bus	4691	1062	8030	0	0
System reset	14	1	65	0	0
SkyNet DVFS total	56902	2084	66781	209	333
Accelerator	49910	921	56095	209	333
AXI bus	5799	1161	9127	0	0
DFS module	1164	0	1492	0	0
System reset	29	2	67	0	0
Total	230400	101760	460800	312	1728

the traditional design flow, designers should first add the DVFS module in PL, then prepare the drivers for controlling DVFS module in Linux. After the DVFS function is ready, the accelerators are added to the system and connected with the DVFS modules, which is quite exhausting. What is more, preparing a test bench that is able to cover a sufficient number of test cases is also time-consuming. To solve this problem, we combine the DVFS framework with Xilinx's latest tool for developing ARM + FPGA system in C/C++/OpenCL, SDSoC, to further enhance the efficiency of system development.

The starting point of an SDSoC-based design is an SDSoC hardware platform, where all the hardware components such as DDR and DVFS modules are defined. Then the boot loaders and target operating systems are built to bring-up the hardware platform. After that, the drivers for the DVS and the DFS module are embedded in the operating system. The boot loaders, target operating system, and the drivers together are called SDSoC software platform, and the hardware platform and software platform together are called SDSoC platform. With this platform, users can develop an embedded system with hardware accelerators in a total software design flow. The user's C/C++/OpenCL code will be translated into Verilog/VHDL code and package the Verilog/VHDL into IPs by SDSoC compiler. After that, SDSoC will connect the accelerator to the PS and the DFS module automatically and then generate the bitstream. Using this platform, we can focus on the accelerator design rather than paying attention to the DVFS module.

6. Experimental results and analysis

6.1. Experimental setup

The SkyNet is written in high-level synthesis (HLS) C++ and then synthesized in SDSoC 2019.1. The Quad Cortex-A53 cores in ZCU104 is running Ubuntu18.04 modified by Xilinx. We test the system using the sample dataset of DAC-SDC2019 that is composed of 1000 images and corresponding ground truth. We record the total time and energy consumption for processing all the images and calculated the average IoU under different voltage & frequency combinations. We sweep the VCCINT from 680 to 840 mV at the step size of 20 mV and sweep the frequency from 200 to 400 MHz at the step size of 1 MHz. Table 1 gives the resource consumption of the original SkyNet and SkyNet with DVFS.

6.2. Performance analysis

Fig. 9 shows the relationship between the frequency and the achieved frames per second (FPS). The figure shows that the highest performance of SkyNet in Zynq Ultrascale+ is at 371 MHz with 38.3 FPS and that the achieved FPS performs a linear relation with frequency as expected.

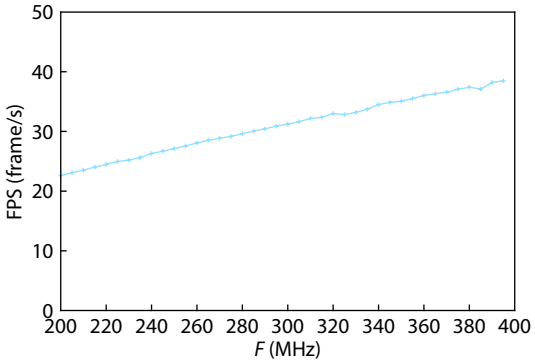

Fig. 9. Total time change with with frequency respectively.

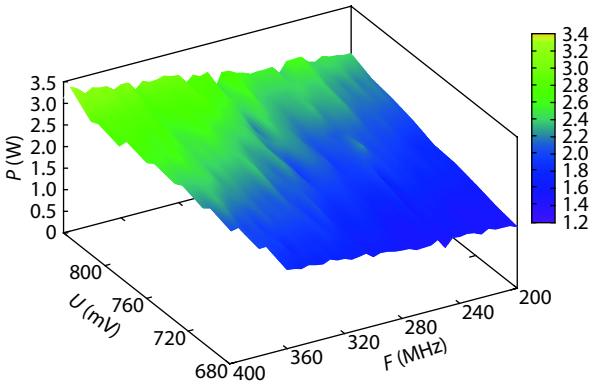

Fig. 10. (Color online) PL side power change with voltage and frequency respectively.

6.3. Power analysis

To better illustrate the impact of DVFS on power consumption, in this section, we focus on the power of the PL that is supplied by the VCCINT power rail. Other power rails include VCCAUX which charges the clock managers, IOs among the other blocks and VCCBRAM used by BRAMs. The power drawn from these additional power rails is considerably lower than VCCINT. Also, the PS of the device where the ARM processor resides is not included in the calculations in that scaling VCCINT does not affect the power of the ARM processor.

Fig. 10 shows the measured power in function of the clock frequency and the voltage when the SkyNet accelerator operates on ZCU104. The highest frequency at which the accelerator can operate increases with the voltage of the PL. For instance, if VCCINT is set to 680 mV, the highest frequency of the accelerator is 360 MHz, above which the system is unstable. Whether the calculation result is correct or not is not considered here, and the impact of DVFS on accuracy will be discussed later. As expected, power has a linear relation with frequency and that the configurations of voltage scaled reduce power significantly since voltage affects both dynamic and static power. The minimum power measured is 1.42 W at 200 MHz/680 mV for the Ultrascale+ device. Therefore, these experiments confirm that significant performance and power margins are available that can be exploited by the DVFS framework.

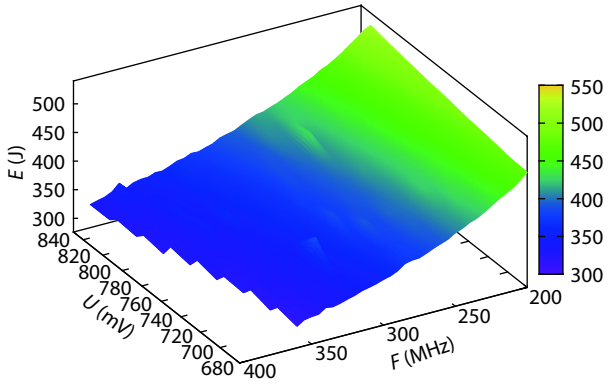

Fig. 11. (Color online) Total energy changes with voltage and frequency respectively.

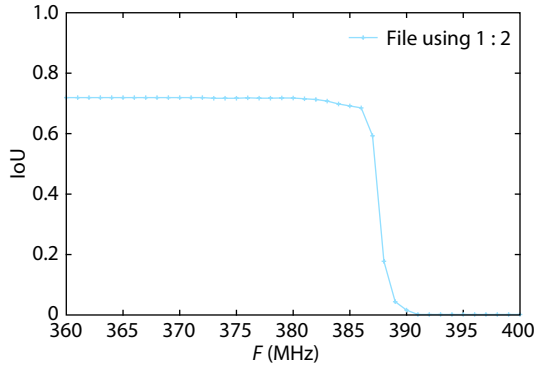

Fig. 12. (Color online) IoU changes with frequency at 840 mV.

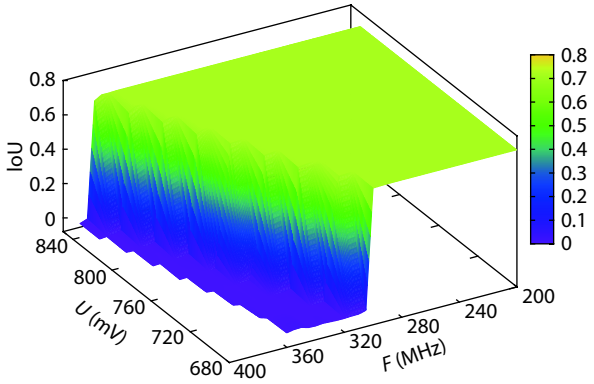

Fig. 13. (Color online) Average of IoU changes with voltage and frequency respectively.

6.4. Energy analysis

In the previous section, we only considered the power of the PL part to make the result look more obvious, but considering the actual edge computing scenario, we should also take the power consumption of the CPU into account. For example, when the data arrives, the CPU wakes up and processes the images, and then the CPU immediately enters the sleep state. Although the power of the PL side goes up as the frequency increases, it is also considered that the time for processing each picture as the frequency increases is also reduced. In such a scenario, the computational time reduction caused by the frequency increase may bring energy savings in the CPU part.

Fig. 11 gives the relationship between the total energy consumption for processing all the 1000 images in the sample dataset and voltage/frequency obtained by our DVFS framework. As shown in Fig. 11, at each voltage point, the total energy consumption goes down as the frequency increases. The reason is that when the accelerator's frequency runs at a low frequency, the dynamic power decreases but the static power remains. What is more, although the PS side's power consumption decreases as the operating frequency of accelerator decreases because the data throughput is reduced, the difference can be neglected compared to the total power of the PS. At the meantime, at each frequency point, the energy consumption performs a linear relationship with voltage. The highest energy consumption is 518.8 J achieved at 200 MHz/840 mV while the least energy consumption is 354.1 J achieved at 371 MHz/840 mV. About 32% of energy can be saved with our DVFS framework. It should be noted that all the data mentioned above is given under strict constraint that the result should be 100% correct. In the next section, we will relax the constraint and discuss the impact of DVFS on accuracy.

6.5. Accuracy analysis

All the results presented so far have used the neural network at full accuracy, that is 71.9% IoU on the whole sample dataset. As previously mentioned, it is possible to relax this constraint and let errors affect the user logic. As shown in Fig. 12, there is a threshold frequency at 371 MHz, above which there may be errors in the final results and the accuracy will change. However, the IoU does not go from 71.9% sharply to zero, but goes through four different stages, namely jitter period, slow descent period, quick descent period and zero period. In the first period, the IoU jitter around 71.9% within +1/−1% until 382 MHz. In the second period, the IoU goes down gradually from 383 to 386 MHz (70.8% to 68.5%). Only until the quick descent period (387 to 388 MHz), we can see a sharp decrease in IoU. We regard the region where the IoU is below 10% as the zero period. This means that a further increase in performance is possible if we relax the IoU requirement to the jitter period and quick descent period.

It is foreseeable that the threshold frequency is related to voltage and Fig. 13 shows the IoU varies concerning both voltage and frequency. The first error comes at 285 MHz when VCCINT is set to 680 mV and the threshold frequency comes later in an approximately linear relationship with the voltage. What is more, the jitter period, slow descent period and quick descent period narrow as the voltage goes up (285–301–304 MHz at 680 mV versus 371–382–386 MHz at 840 mV).

Through the above analysis, we can draw the conclusion that, CNNs are robust to errors. Even if errors are generated by the accelerator during operation, it have a very limited effect on the final result when the number of errors is small. Through this feature, we can sacrifice accuracy and performance to some extent.

6.6. Comprehensive evaluation metric

Previous works usually compare performance, energy efficiency and the achieved accuracy respectively. However, these three metrics are not completely independent but can be transformed into each other through various means. For instance, network compression and quantization can be the method to balance between accuracy and performance. Our

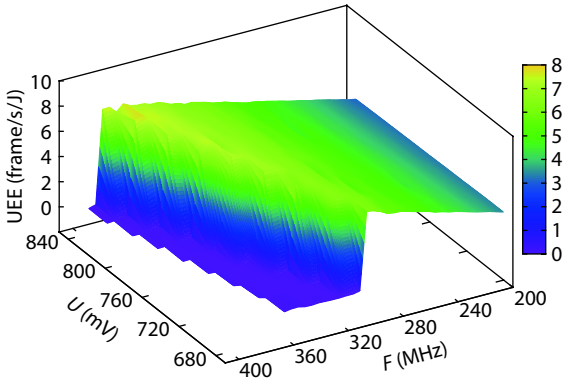

Fig. 14. (Color online) UEE changes with voltage and frequency respectively.

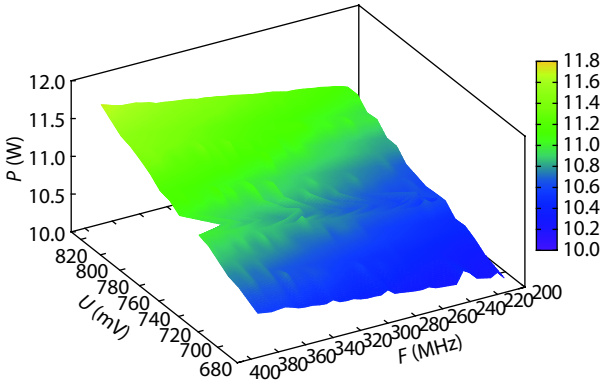

Fig. 15. (Color online) Idle power changes with voltage and frequency respectively.

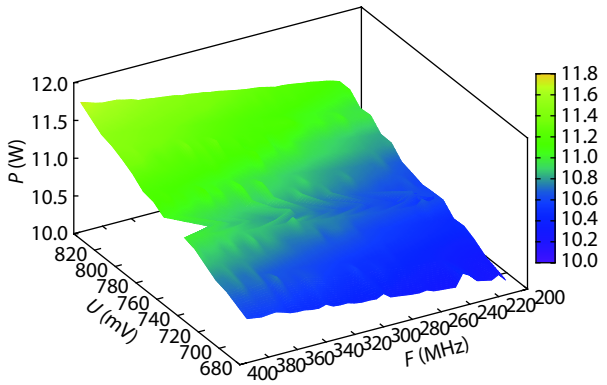

Fig. 16. (Color online) Average power changes with voltage and frequency respectively.

DVFS framework is a method to trade-off between energy efficiency and performance as well as accuracy. Therefore, a comprehensive evaluation metric that takes performance, energy efficiency and accuracy into account is necessary to evaluate a system design. In this paper, the performance is given by the achieved FPS, energy efficiency is given by the energy consumption for each frame and the accuracy is given by IoU. Under this context, we proposed a rough metric called unified energy efficiency (UEE):

$$\text{UEE} = \frac{\text{Performance} \times \text{Accuracy}}{\text{Energy}}. \tag{5}$$

The UEE is given by:

$$\text{UEE} = \frac{\text{FPS} \times \text{Average IoU}}{\text{Total energy}}. \tag{6}$$

There are previous works using frames per second per watt (FPS/W) as a metric. However, the watt is given by Joule per second and thus FPS/W equals to frames per Joule (FPJ), it does not take the performance into account. As a result, in this paper, we adapt energy rather than power as the numerator.

Fig. 14 shows UEE changes with voltage and frequency respectively. As shown in the figure, UEE first goes up with frequency at each voltage since the accuracy remains and the performance and energy efficiency increase and then goes down sharply as IoU drops to zero. For each frequency point under 290 MHz, the threshold frequency of 680 mV, UEE decreases as voltage goes up since lower voltage has higher energy efficiency, However, higher voltage provides higher threshold frequency and thus can get higher performance and energy efficiency. The highest UEE, 7.71, is achieved at 840 mV/379 MHz and is beyond 7.22 achieved at 840 mV/370 MHz which is the threshold voltage. It also shows that the DVFS can further improve UEE.

6.7. DVFS for real-time application

As mentioned in the previous section, in realistic scenarios, the needed frame rate is bounded by the camera. In this paper, we assume that the frame rate of the camera is 24 FPS, the standard movie frame rate, and try to find the lowest average power solution that meets this performance requirement with our proposed DVFS policy. Fig. 15 gives the relationship between idle power and voltage/frequency. As can be seen from the figure, the idle power performs a linear relationship between both the voltage and the frequency. Thus slowing down the clock, which can be regarded as a variant of clock gating technology, can be an effective method to save idle power. Furthermore, scaling down the voltage combined with clock gating can save idle power to the utmost. P_{Imin} is 9.81 W, achieved when the frequency is scaled to 20 MHz and the voltage is scaled to 680 mV.

Fig. 16 shows the achieved P_{M} at different voltage/frequency combinations. As shown in the figure, the minimum average power is 10.90 W and is achieved at 680 mV/285 MHz, we get a 15% reduce in power consumption compared to 12.86 W of the original design.

6.8. Comparison with other work

We compare the performance, energy efficiency, accuracy as well as UEE with the original SkyNet, where the energy efficiency is given by frames per Joule (FPJ). The original SkyNet is implemented on Ultra96, we transfer the design to ZCU104 without any modification. We choose the data achieved at the threshold frequency of 840 mV/370 MHz as the first candidate and that achieved at 840 mV/379 MHz as the second candidate. As shown in Table 2, our work achieves 54% improvement in performance, 38% improvement in energy efficiency and 106% improvement in UEE without any degradation in accuracy compared to the original SkyNet. If we relax the requirement on accuracy, we can achieve 56% improvement in performance, 47% improvement in energy efficiency and 121% improvement in UEE at the cost of 0.11% decrease in IoU.

Table 2. Comparison with other work.

Parameter	Performance	Energy efficiency	IoU	UEE
SkyNet[25]	23.93 FPS	2.02 FPJ	71.91%	3.49
Candidate 1	37.04 FPS, 1.54 ×	2.79 FPJ, 1.38 ×	71.91%	7.22, 2.06 ×
Candidate 2	37.42 FPS, 1.56 ×	2.97 FPJ, 1.47 ×	71.80%	7.71, 2.21 ×

7. Conclusion

FPGA has been proven as a favorable platform to accelerate convolution neural network (CNN) on the edge-computing paradigm given its high flexibility and energy efficiency. However, the energy efficiency of the FPGA platform can be further improved with dynamic voltage and frequency scaling (DVFS). Although an overly aggressive voltage and frequency combination can lead to errors, the high robustness of the CNN to errors makes it possible to combine DVFS with it. In this paper, we first introduce a framework for fine-grained DVFS on the state-of-the-art FPGA device and then apply the framework to SkyNet, a state-of-the-art neural network targeting on object detection. Third, we analyze the impact of DVFS on performance, power, energy efficiency and accuracy. We verify the possibility of sacrificing accuracy for performance or energy efficiency. In order to evaluate the entire system comprehensively, we propose a new metric, called unified energy efficiency (UEE), that takes performance, energy efficiency as well as accuracy into account. Finally, we achieve 54% improvement in performance, 38% improvement in energy efficiency and 106% improvement in UEE without any loss in accuracy compared to the original SkyNet. If we relax the requirement on the accuracy, we can achieve 56% improvement in performance, 47% improvement in energy efficiency and 121% improvement in UEE at the cost of 0.11% decrease in accuracy.

References

[1] Nurvitadhi E, Venkatesh G, Sim J, et al. Can FPGAs beat GPUs in accelerating next-generation deep neural networks. Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2017

[2] Mantovani P, Cota E G, Tien K, et al. An FPGA-based infrastructure for fine-grained DVFS analysis in high-performance embedded systems. Proceedings of the 53rd Annual Design Automation Conference, 2016

[3] Bai L, Zhao Y, Huang X. A CNN accelerator on FPGA using depthwise separable convolution. IEEE Trans Circuits Syst II, 2018, 65(10), 1415

[4] Ma Y, Cao Y, Vrudhula S, et al. An automatic RTL compiler for high-throughput FPGA implementation of diverse deep convolutional neural networks. 27th International Conference on Field Programmable Logic and Applications (FPL), 2017

[5] Ma Y, Cao Y, Vrudhula S, et al. Optimizing loop operation and dataflow in FPGA acceleration of deep convolutional neural networks. Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2017

[6] Ma Y, Kim M, Cao Y, et al. End-to-end scalable FPGA accelerator for deep residual networks. 2017 IEEE International Symposium on Circuits and Systems (ISCAS), 2017

[7] Wei X, Liang Y, Li X, et al. TGPA: tile-grained pipeline architecture for low latency CNN inference. 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2018

[8] Guo K, Sui L, Qiu J, et al. Angel-eye: A complete design flow for mapping CNN onto embedded FPGA. IEEE Trans Comput-Aid Des Integr Circuits Syst, 2018, 37(1), 35

[9] Ma Y, Cao Y, Vrudhula S, et al. Performance modeling for cnn inference accelerators on FPGA. IEEE Trans Comput-Aid Des Integr Circuits Syst, 2019

[10] Qiu J, Wang J, Yao S, et al. Going deeper with embedded FPGA platform for convolutional neural network. Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2016

[11] Zhang X, Wang J, Zhu C, et al. Dnnbuilder: an automated tool for building high-performance DNN hardware accelerators for FPGAs. Proceedings of the International Conference on Computer-Aided Design, 2018

[12] Motamedi M, Fong D, Ghiasi S. Machine intelligence on resource-constrained IoT devices: The case of thread granularity optimization for CNN inference. ACM Trans Embedded Comput Syst, 2017, 16(5s), 151

[13] Xiao Q, Liang Y, Lu L, et al. Exploring heterogeneous algorithms for accelerating deep convolutional neural networks on FPGAs. 2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC), 2017

[14] Dutta S, Bai Z, Low T M, et al. Codenet: Training large scale neural networks in presence of soft-errors. arXiv preprint arXiv: 190301042, 2019

[15] Nie B, Tiwari D, Gupta S, et al. A large-scale study of soft-errors on GPUs in the field. 2016 IEEE International Symposium on High Performance Computer Architecture (HPCA), 2016

[16] Chen Y, Zhu Y, Qiao F, et al. Evaluating data resilience in CNNs from an approximate memory perspective. Proceedings of the on Great Lakes Symposium on VLSI, 2017, 89

[17] Qiao A, Aragam B, Zhang B, et al. Fault tolerance in iterative-convergent machine learning. arXiv preprint arXiv: 1810.07354, 2018

[18] Nunez-Yanez J L. Adaptive voltage scaling with in-situ detectors in commercial FPGAs. IEEE Trans Comput, 2014, 64(1), 45

[19] Nabina A, Nunez-Yanez J L. Adaptive voltage scaling in a dynamically reconfigurable FPGA-based platform. ACM Trans Reconfig Technol Syst, 2012, 5(4), 20

[20] Wei X, Liang Y, Cong J. Overcoming data transfer bottlenecks in FPGA-based DNN accelerators via layer conscious memory management. DAC, 2019, 125

[21] Ding C, Wang S, Liu N, et al. Req-yolo: A resource-aware, efficient quantization framework for object detection on FPGAs. Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2019

[22] Zhang X, Hao C, Li Y, et al. A bi-directional co-design approach to enable deep learning on IoT devices. arXiv preprint arXiv: 190508369, 2019

[23] Hao C, Zhang X, Li Y, et al. FPGA/DNN co-design: An efficient design methodology for IoT intelligence on the edge. Proceedings of the 56th Annual Design Automation Conference, 2019

[24] Nunez-Yanez J L. Energy proportional neural network inference with adaptive voltage and frequency scaling. IEEE Trans Comput, 2018, 99(99), 1

[25] Zhang X, Hao C, Lu H, et al., Skynet: A champion design for DAC-SDC on low power object detection. arXiv preprint arXiv: 190610327, 2019

[26] Weissel A, Bellosa F, Process cruise control: event-driven clock scaling for dynamic power management. Proceedings of the 2002 International Conference on Compilers, Architecture, and Synthes-

is for Embedded Systems, 2002
[27] De Vogeleer K, Memmi G, Jouvelot P, et al. The energy/frequency convexity rule: Modeling and experimental validation on mobile devices. International Conference on Parallel Processing and Applied Mathematics, 2013
[28] Huang H, Chaturvedi V, Quan G, et al. Throughput maximization for periodic real-time systems under the maximal temperature constraint. ACM Trans Embed Comput Syst, 2014, 13(2s), 70
[29] Yu H, Syed R, Ha Y. Thermal-aware frequency scaling for adaptive workloads on heterogeneous MPSoCs. Proceedings of the Conference on Design, Automation & Test in Europe, 2014
[30] Yu H, Ha Y, Wang J. Quality optimization of resilient applications under temperature constraints. Proceedings of the Computing Frontiers Conference, 2017
[31] Ma Y, Chantem T, Dick R P, et al. Improving system-level lifetime reliability of multicore soft real-time systems. IEEE Trans Very Large Scale Integr Syst, 2017, 25(6), 1895
[32] Bong K, Choi S, Kim C, et al. Low-power convolutional neural network processor for a face-recognition system. IEEE Micro, 2017, 37(6), 30
[33] Santoro G, Casu M R, Peluso V, et al. Design-space exploration of pareto-optimal architectures for deep learning with DVFS. 2018 IEEE International Symposium on Circuits and Systems (ISCAS), 2018
[34] Hsieh G C, Hung J C. Phase-locked loop techniques. A survey. IEEE Trans Indust Electron, 1996, 43(6), 609
[35] Kim J H, Kwak Y H, Kim M, et al. A 120-MHz–1.8-GHz CMOS dll-based clock generator for dynamic frequency scaling. IEEE J Solid-State Circuits, 2006, 41(9), 2077
[36] Brynjolfson I, Zilic Z. Dynamic clock management for low power applications in FPGAs. Proceedings of the IEEE 2000 Custom Integrated Circuits Conference, 2000
[37] Beldachi A F, Nunez-Yanez J L. Run-time power and performance scaling in 28 nm FPGAs. IET Comput Digit Tech, 2014, 8(4), 178
[38] Beldachi A F, Nunez-Yanez J L. Accurate power control and monitoring in zynq boards. 2014 24th International Conference on Field Programmable Logic and Applications (FPL), 2014
[39] Hosseinabady M, Nunez-Yanez J L. Run-time power gating in hybrid arm-FPGA devices. 2014 24th International Conference on Field Programmable Logic and Applications (FPL), 2014