Exploring Slice-Energy Saving on an Video Processing FPGA Platform with Approximate Computing

Yunxiang Zhang, Xiaokun Yang, Lei Wu, Jiang Lu,
Kewei Sha, Archit Gajjar, Han He
University of Houston-Clear Lake
2700 Bay Area Blvd, Houston, TX77058, USA
{ZhangY0552, YangXia, WuL, Luj, Sha, GajjarA7402}@uhcl.edu

ABSTRACT

This paper proposes a scalable video processing platform on Field Programmable Gate Array (FPGA), providing several slice-energy cost solutions corresponding to different application constrains. Specifically three approximations of multipliers and two approximations of adders, along with the exact designs as well, are presented and integrated as twelve benchmarks to implement RGB to grayscale conversion as a case study. Experimental results show that the minimum slice-energy cost, integrated with approximate#2 adder and approximate#3 multiplier, achieves 25.17% slice-energy saving compared with the exact design by sacrificing the quality of results as 5.69% error for multiplier and 2.85% for adder.

CCS Concepts

• **Hardware→ Application specific integrated circuit**

Keywords

Approximate design, Field Programmable Gate Array (FPGA), slice-energy cost.

1. INTRODUCTION

In order to reduce the computational cost and improve the energy efficiency, approximate design on Field Programmable Gate Array (FPGA) platforms has been widely used in many application domains, such as artificial intelligent [1], edge computing [2], and Internet-of-Things (IoT) security [3] [4].

The inaccurate implementation potentially provides an opportunity to find the minimum FPGA cost in terms of slice number and energy consumption corresponding to different quality of constrains. The main challenges are: 1) the combinational circuit design on FPGA is mapped to look-up-tables (LUTs), leading to uncertainty to determine the power consumption and the maximum operational frequency (MOF); 2) different from the improvement on one sub-components, the combination of multiple approximations of computational components affects the energy cost and slice utilization, making the tradeoff between accuracy and slice-energy reduction difficult.

Under this background, this paper proposes several approximations of adders and multipliers, and then applies all the components on an FPGA platform for converting color image into grayscale image. More specifically, the main contributions are:

- We presented a fixed-point design on RGB to grayscale converter (RGB2Grayscale) with a Spartan-6 FPGA platform. Then three approximations of fixed-point multipliers, and two approximations of adders are proposed and applied in this prototype. The tradeoff between quality of results and resource cost is statically analyzed with different implementations.
- We implemented twelve RGB2Grayscale converters with different approximations of register-transfer-level (RTL) design, and synthesized the design-under-test (DUT) with a Spartan-6 xc6slx45-3csg484. The performance in terms of slice count and dynamic energy consumption were estimated using the performance evaluation methodology [5][6].
- In order to evaluate the cost saving on weighted slice count and energy dissipation, the slice-energy metric is presented in our work. Experimental results show that the minimum slice-energy reduction can reach 25.17% compared with the exact design.

The organization of this paper is as follows: section 2 briefly reviews the related works of FPGA cost saving on approximate computing, and section 3 introduces our proposed work. In section 4, the system performance are statically formulated and analyzed and in section 5, the experimental results are estimated demonstrated on an FPGA platform. Finally, section 6 concludes this paper.

2. PRIOR WORKS

To find the tradeoff between quality of the results and energy cost on FPGA is one of the important points driving the research of approximate computing. Previous works in such field have mainly focused on combinational circuit design [7], and some of the researchers concentrated on optimizing the flow chart to reduce the energy consumption [8]. In this paper we show the prototype in a sequential circuit with twelve approximations of designs, providing a wide range of quality-cost tradeoffs.

Additionally the earlier researches on improving FPGA design focused on either the low-energy technology [9][10] or the low-cost architecture [11]. However, most embedded chips are operating in cost-limited and energy-constrained environments such as energy harvesting powered platforms and microcontrollers, in which the increasing of resource cost and power consumption

ICACS '18, July 27–29, 2018, Beijing, China

ACM ISBN 978-1-4503-6509-3/18/07…$15.00
https://doi.org/10.1145/3242840.3242852

will villainously shorten the lifetime of the systems. To overcome this issue, a prior work in [12] proposed four approximations of addition models and evaluated the quality of results on a histogram equalization algorithm. The idea was simulated on Matlab so the hardware performance was not estimated. This paper focuses on proposing several approximate multipliers and employing the approximate adders as well, and more important, the slice-energy savings are estimated and demonstrated on an FPGA platform.

3. APPROXIMATE DESIGN

3.1 Energy Consumption on FPGA

Generally energy can be computed by running time and power as Energy = Power × Time. Thus to reduce the energy cost we can either increasing the maximum operating frequency (MOF) or decreasing the power consumption. However, decreasing the clock frequency can lower the system speed and cost more time on each cycle, resulting an increasing of energy dissipation. Thus the power reduction is mainly considered in this work to save energy. Since the static power on FPGAs is dependent on the specific designs, in this work we focus on dynamic power estimation, which can be expressed as

$$P_{dyn} = (V^2) \times \sum_{i=1}^{n} C_{eff-i} \times U_i \times f_i \qquad (1)$$

where the total switching capacitance is the product of its effective capacitance C_{eff-i}, the number of instances in the design U_i, and the average switching frequency across all the instances f_i including the logic, signals, and IOs. The dynamic power of switching all instances of resource i is the product of (V^2) and its switching capacitance. From Eq. (1), one of the most effective ways to lower the dynamic power is reducing number of logic, IOs, and signals employed in a digital system design.

3.2 A Case Study of RGB2Grayscale Converter

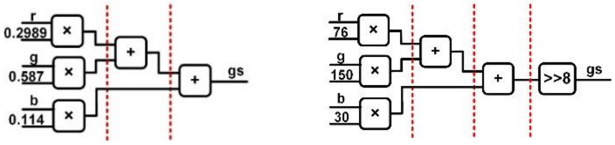
As a case study, the RGB2Grayscale converter is implemented as different approximations of designs in our work. Basically a grayscale pixel can be computed as a summation of 29.89% of the red pixel, 58.7% of green, and 11.4% of blue, which is written as below

$$Grayscale = o.2989 \times r + 0.587 \times g + 0.114 \times b \text{ .(2)}$$

Fig. 1(a) shows the design structure with three floating-point multipliers and two adders. Although the floating-point multiplier is precise in results but it costs much more slices compared to the fixed-point designs. Therefore, this paper presents a fixed-point design shown in Fig. 1(b) by multiplying 2^8 for the three floating-point constants then rounding the fractions up, and finally dividing by 2^8 after the addition.

$$Grayscale = (76 \times r + 150 \times g + 30 \times b)/256. \qquad (3)$$

Furthermore, we present and apply several approximations of multipliers and adders in the design, in order to trade the accuracy for the energy efficiency. Theoretically the higher approximation of the multiplier, the higher energy efficiency can be achieved.



(a) floating-point (b) fixed-point

Figure 1. Design Structures of RGB2Grayscale Converter

3.3 Proposed Approximate Multiplier

In this section, four approximations of multiplier are presented. Notice that any multi-bit multiplication, denoted as $W \times W - bit$, can be integrated with four $\frac{W}{2} \times \frac{W}{2} - bit$ designs. As shown in Fig. 2, two inputs A and B are represented as (A_H A_L) and (B_H B_L) with the most significant bit (MSB) A_H and B_H, and the least significant bit A_L and B_L (LSB). Then the sum of four partial products, denoted as $A_L \times B_L$, $A_H \times B_L$, $A_L \times B_H$ and $A_H \times B_H$, is the final product of $W \times W - bit$ multiplication.

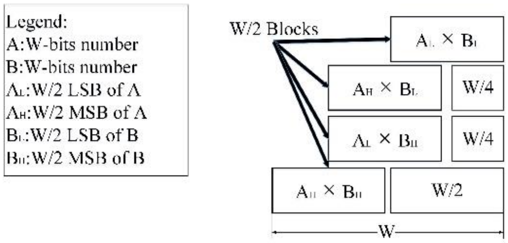


Figure 2. Multi-bit Multiplier Design

Before discussing the approximate design, the exact $2 \times 2 - \text{bit}$ multiplication can be implemented using the K-map shown in Fig. 3 and written as

$$\text{Mul}[3] = \text{A}[1] \cdot \text{A}[0] \cdot \text{B}[1] \cdot \text{B}[0].$$

$$\text{Mul}[2] = \text{A}[1] \cdot \text{B}[1] \cdot \text{B}[0]' + \text{A}[1] \cdot \text{A}[0]' \cdot \text{B}[1].$$

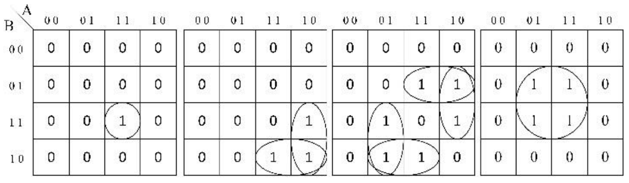
$$\text{Mul}[1] = \text{A}[1] \cdot \text{A}[0]' \cdot \text{B}[0] + \text{A}[0] \cdot \text{B}[1] \cdot \text{B}[0]' + \text{A}[1] \cdot \text{B}[1]' \cdot \text{B}[0] + \text{A}[1]' \cdot \text{A}[0] \cdot \text{B}[1].$$

$$\text{Mul}[0] = \text{A}[0] \cdot \text{B}[0]. \qquad (4)$$

where Mul[3], Mul[2], Mul[1], and Mul[0] are the four bits of the products, from the MSB to LSB. And A[1: 0] and B[0] and the 2-bit input of the multiplier.

The corresponding design structure of the exact multiplier is shown in Fig. 4, requiring sixteen AND gates and four OR gates. More specifically, the computation of the MSB bit takes three AND gates and the LSB takes one AND gate. The middle bit Mul[2] needs four AND gates and one OR gate, and the Mul[1] bit requires eight AND gates and three OR gates. Since the LSB only uses one AND gate, it doesn't need to be simplified.

$$AP\_\text{Mul}[3] = \text{A}[1] \cdot \text{A}[0] \cdot \text{B}[0] \qquad (5)$$



B A	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	1	0
10	0	0	0	0

(a) Mul[3]

B A	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	1
10	0	0	1	1

(b)Mul[2]

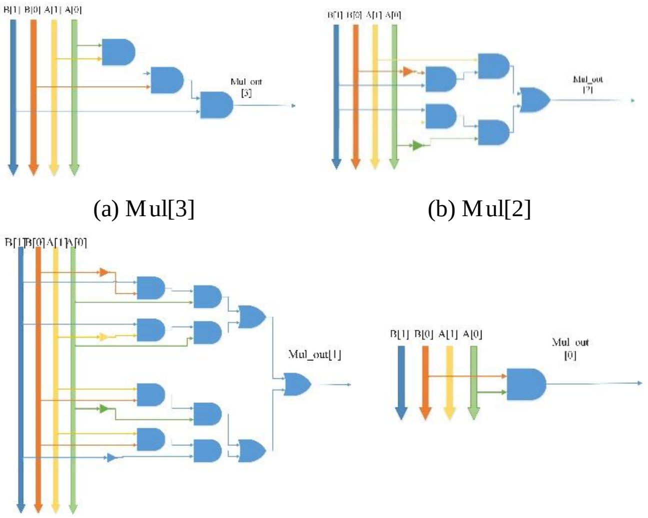
B A	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	0	1	0	1
10	0	1	1	0

(c) Mul[1]

B A	00	01	11	10
00	0	0	0	0
01	0	1	1	0
11	0	1	1	0
10	0	0	0	0

(d) Mul[0]

Figure 3 K-map Simplification of $2 \times 2 -$ bit Exact Multiplier



(a) Mul[3]

(b) Mul[2]

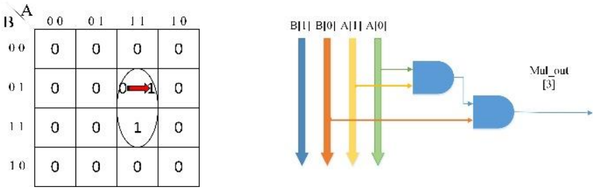
(c) Mul[1]

(d) Mul[0]

Figure 4 Exact Multiplier Design

To reduce the gate count for Mul[3:1], we present some approximations by modifying three bits from 0 and 1 in the K-map. For example shown in Fig. 5, we change 0 to 1 for the case A[1:0]=11 and B[1:0]=01, leading to a simple boolean expression as

Comparing to Fig. 4 (a) with the exact Mul[3] bit computation, the approximate design in Fig. 5 reduces one AND gate within the criticial path and one AND gate for the totoal gate count as well, which theoritically would achieve a higher MOF and lower power dissipation.



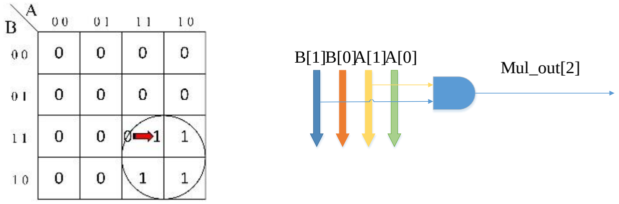
B A	00	01	11	10
00	0	0	0	0
01	0	0	0→1	0
11	0	0	1	0
10	0	0	0	0

Figure 5 Approximate Design for Mul[3]

Similarly, the exact computation of Mul[2] in Fig. 4(b) can be improved in resource cost by modifying 0 to 1 at A[1:0]=11 and B[1:0]=11, as shown in Fig. 6. The boolean expression is rewritten as

$$AP\_\mathrm{Mul}[2] = \ \mathrm{A}[1] \cdot \mathrm{B}[1]. \tag{6}$$

After the optimization, the total gate cost is reduced from five to one gate. In other words, the approximation sacrifices one bit error for saving 80% gate numbers compared to the eact design.



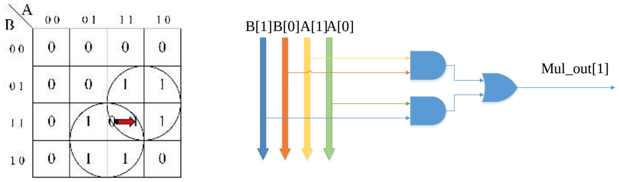
B A	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0→1	1
10	0	0	1	1

Figure 6 Approximate Design for Mul[2]

Finally, the Mul[1] is simplifed as the boolean expression below with changing 0 to 1 for the case of A[1:0]=11 and B[1:0]=11.

$$AP\_\mathrm{Mul}[1] = \ \mathrm{A}[1] \cdot \mathrm{B}[0] + \mathrm{A}[0] \cdot \ \mathrm{B}[1]. \tag{7}$$

Comparing to the exact design on Mul[1] shown in Fig. 4(c), the approximate design shown in Fig. 7 significantly reduces the gate count by 72.7% with one bit error tallerance.



B A	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	0	1	0→1	1
10	0	1	1	0

Figure 7 Approximate Design for Mul[1]

In conclusion, we present three different approximations of 2×2-bit multiplier design combining the three approximate bit computations. In Eq. 4, Mul[1] is replaced by AP_Mul[1] as the first approximation (AP #1), Mul[2:1] is substituted with AP_Mul[2:1] as the second approximation (AP #2), and Mul[3:1] is replaced by with AP_Mul[3:1] as the third approximation (AP #3).

Therefore, the total gate counts for the exact design and AP#1~AP#3 designs can be summarized in Table 1. It can be observed that the resource cost is saved by 40%, 60%, 65%, respectively, for AP#1, AP#2, AP#3, compared with the exact design, leading to a significant slice and energy saving by employing inexact computing. Notice that the combinational design on FPGA is based on LUT not logic gate, so the results might be a little difference, which is proved in Section 5.

Table 1. Gate Cost of 2×2-bit Multiplier

Designs	Hardware cost
EX MUL	16 AND + 4 OR
AP#1 MUL	10 AND + 2 OR
AP#2 MUL	7 AND + 1 OR
AP#3 MUL	6 AND + 1 OR

3.4 Approximate Adder

Generally, the multi-bit adders can be simply integrated with several single-bit adders. The exact single-bit adder can be expressed as

$$Cout = (a \cdot b) + (b \cdot c) + (a \cdot c)$$
$$Sum = (a' \cdot b \cdot c') + (a \cdot b' \cdot c') + (a' \cdot b' \cdot c) + (a \cdot b \cdot c) \tag{8}$$

Likewise, two approximate single-bit adders can be rewritten as

$$AP1\_Cout = a + (b \cdot c)$$

$$AP1\_Sum = (a' \cdot b \cdot c') + (a' \cdot b' \cdot c) + (a \cdot b \cdot c) \qquad (9)$$

and

$$AP2\_Cout = a$$

$$AP2\_Sum = (a' \cdot c) + (b \cdot c) + (a' \cdot c) \qquad (10)$$

where a and b are the two inputs, and c is the carry in bit. AP1_cout and AP1_sum are the carry out bit and summation bit for the first approximate adder, and AP2_cout and AP2_sum are bits the second approximate adder.

The static analysis of the approximate adder is shown in Table 2. Compared to the exact adder design, it can be observed that the gate counts are saved by 77.8% and 44.4%, respectively, for using AP#1 and AP#2.

Table 2. Single-Bits Adder Resource Cost

Designs	Hardware cost
EX Adder	7AND+2OR
AP#1 Adder	4AND+2OR
AP#2 Adder	3AND+1OR

4. FIGURES/CAPTIONS

This section evaluates the quality of the results by using exact design and approximate implementations.

In Fig. 8, six grayscales image converted by different approximations of RGB2Grayscale designs are depicted. Fig. 8(a) shows the quality of result using the exact design. Fig. 8(b), Fig. 8(c) and Fig. 8(d) depict the results employing the AP#1, AP#2, and AP#3 multipliers respectively. And Fig. 8(e) and Fig. 8(f) show the results with AP#1 and AP#2 adders respectively.

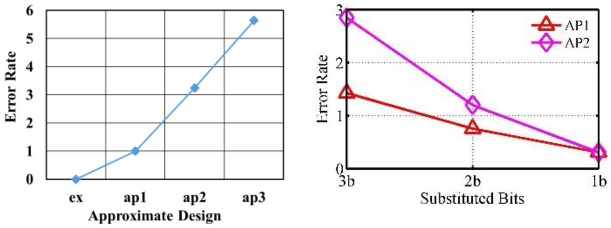
(a) Exact (b) AP#1 Mul (b) AP#2 Mul

(c) AP#3 Mul (d) AP#1 Add (e) AP#2 Add

Figure 8 Grayscale Images with Approximations of RGB2Grayscale Designs

It is not clear to see the difference between images in Fig. 8 by human eyes, so the error rate is further graphed in Fig. 9. As depicted in Fig. 9 (a), the horizontal axis represents the different approximates of the multipliers, and the vertical axis indicates the error rate for each specific implementation. It can be observed that the error rates for converting the color image into grayscale image are 0.999%, 3.243%, and 5.64%, respectively, by using AP#1, AP#2, and AP#3 multipliers.

In Fig. 9 (b), the error rate decreases by replacing less number of addition bits, which is represented by the horizontal axis from the third bit (3b) to the least bit (1b). It is obvious that the higher bits have more error effect compared to the lower bits. To keep the error tolerance acceptable (less than 4%), therefore, only the least three bits are considered in this benchmark. When replacing all the three bits with AP#1 and AP#2 adders, the error rates are 1.43% and 2.85%, respectively. The error rates drop to around 1% with replacing the least signifiant two bits for both AP#1 and AP#2 adders, and the error rates are less than 1% when replacing the least significant bit.



(a) AP Mul (b) AP Add

Figure 9 Error rate VS. Approximations

5. FPGA Implementation and Simulation

In this section, first the register transfer level (RTL) design and verification with Verilog hardware description language (HDL) is discussed. The Mentor Graphic ModelSim is used as the simulator, and the Xilinx ISE 14.7 with the target device Spartan-6 xc6slx45-3csg484 is employed as the synthesis tool.

In our work, the FPGA performance evaluation methodology is adopted to estimate the system performance in terms of slice count, speed, and power dissipation [5] [6].

5.1 FPGA Design Flow

After simulation, the toggle activities of signals, IOs, and logic are collected by value changed dump (VCD) files. And then after synthesis, the practical results are summarized in the third and fourth columns of Table 3. It can be observed that with the same adder design, the higher approximations of the multiplier implementation, the less number of the slices are needed. Similarly, when using the same multiplier, higher approximation of adders consume less number of FPGA slices.

The MOF is decided by the critical path delay. However, since the combinational circuit mapping on FPGA is based on the look-up-table (LUT), the critical paths for all the approximations of implementations are the same, resulting in the same MOF as 271.326 MHz.

Table 2 Slice count and dynamic power and energy

DUT No.	AP Add	AP Mul	Slice of Register	Slice of LUT	Dynamic Power (mW)	Dynamic Energy (pJ)
1	Ex	Ex	700	879	13	4.79
2	Ex	Ap1	684	843	12	4.42
3	Ex	Ap2	652	799	12	4.42
4	Ex	Ap3	640	783	12	4.42
5	Ap1	Ex	698	874	13	4.79

6	Ap1	Ap1	682	838	12	4.42
7	Ap1	Ap2	650	794	12	4.42
8	Ap1	Ap3	638	778	12	4.42
9	Ap2	Ex	538	720	10	3.69
10	Ap2	Ap1	522	692	10	3.69
11	Ap2	Ap2	506	672	10	3.69
12	Ap2	Ap3	494	656	10	3.69

Finally, we use XPower Analyzer to estimate the realistic power consumption. Xilinx Power Analyzer evaluates the power with the Native Circuit Description (NCD) file generated by ISE and the specific simulation VCD file. As the power consumption shown in the sixth column in Table 3, the dynamic power decreases with the increasing of approximations of the adders or multipliers. Some of the power consumptions are the same because the toggle rates of slices are similar to each other in this benchmark.

By simply multiplying dynamic power by the reciprocal of MOF, the dynamic energy is computed in the seventh column. Since the MOF are the same for all the approximate designs, the dynamic energy dissipations have the same trend of the power cost.

5.2 FPGA Slice-Energy cost

In order to find the optimal cost saving in terms of slice number and energy consumption corresponding to the specific quality bound, in this section we present a novel performance metric, denoted as slice-energy cost, as below

$$Slice - Energy = S^x \times E^y \qquad (11)$$

where "S" and "E" represent the FPGA cost in terms of slice count and energy dissipation. "x" is the weight of slice count, and "y" is the weight of FPGA energy consumption. The weights x and y are between 0 and 1, and the summation of the weights is considered as a constant value equal to 1, in order to decide the tradeoff between cost savings. By varying x and y, we can target the performance on a specific application. For example, setting x =y=1/2 leads to equal weighting, x = 1 targets the small-size design, and y = 1 targets the low-energy optimization.

Generally, the slice-energy cost saving should be minimized in order to find the optimal design with different weight configurations. As an example for equally setting the two weights as 1/2, the minimum slice-energy cost occurs at the No. 12 design with the highest approximation of multiplier (AP#3) and the highest approximation of adder (AP#2) as shown in Fig. 10.

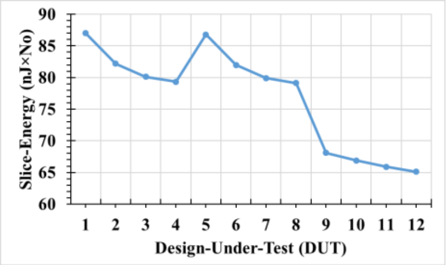


Fig. 10 Slice-Cost VS. Approximations of DUT

6. Conclusion

In this paper, twelve approximations of RGB2Grayscale converters are implemented on an FPGA platform, by proposing and integrating three approximations of multipliers, two approximations of adders, along with exact designs as well. The implementations shown on an FPGA demo provide a wide range of slice-energy saving corresponding to different quality constrains. By a 5.69% quality decreasing for multiplication and 2.85% for addition, the dynamic energy can be reduced to 77.04% and the slice count can be saved to 72.83% compared to the exact design.

Our future work is to implement the face detection algorithm with many approximations, in order to speed up the system and reduce the energy consumption.

7. REFERENCES

[1] H. He, L. Wu, X. Yang, et al, "Dual Long Short-Term Memory Networks for Sub-Character Representation Learning," The 15th Intl. Conference on Information Technology-New Generations (ITNG), PP. 1-6, Jan. 2018.

[2] A. Gajjar, Y. Zhang, and X. Yang, "Demo Abstract: A Smart Building System Integrated with An Edge Computing Algorithm and IoT Mesh Networks," The Second ACM/IEEE Symposium on Edge Computing (SEC2017), No. 35, PP. 1-2, Oct. 2017. DOI=10.1145/3132211.3132462.

[3] X. Yang, W. Wen, and M. Fan, "Improving AES Core Performance via An Advanced IBUS Protocol," ACM Journal on Emerging Technologies in Computing (ACM JETC), Vol. 14, No. 1, PP. 61-63, Jan. 2018. DOI=10.1145/3110713.

[4] X. Yang and W. Wen, "Design of A Pre-Scheduled Data Bus (DBUS) for Advanced Encryption Standard (AES) Encrypted System-on-Chips (SoCs)," The 22nd Asia and South Pacific Design Automation Conference (ASP-DAC 2017), PP. 1-6, Chiba, Japan, Feb. 2017. DOI=10.1109/ASPDAC.2017.7858373

[5] X. Yang, N. Wu, and J. Andrian, "A Novel Bus Transfer Mode: Block Transfer and A Performance Evaluation Methodology," Elsevier, Integration, the VLSI Journal, Vol. 52, PP. 23-33, Jan. 2016. DOI=10.1016/j.vlsi.2015.07.012

[6] X. Yang and J. Andrian, "A Low-Cost and High-Performance Embedded System Architecture and An Evaluation Methodology," IEEE Computer Society Annual Symposium on VLSI (ISVLSI2014), PP. 240-243, Sept. 2014. DOI=10.1109/ISVLSI.2014.20

[7] A. Kahng and S. Kang, "Accuracy-Configurable Adder for Approximate Arithmetic Designs", The 49th Design Automation Conference (DAC2012), PP. 820-825, 2012. DOI=10.1145/2228360.2228509

[8] Z. Kedem, V. Mooney, K. Muntimadugu, et al, "Optimizing Energy to Minimize Errors in Dataflow Graphs Using Approximate Adders", The 2010 Intl. Conference on Compilers, Architectures and Synthesis for Embedded Systems, PP. 177-186, 2010. DOI=10.1145/1878921.1878948

[9] M. Fan, Q. Han, and X. Yang, "Energy Minimization for On-Line Real-Time Scheduling with Reliability Awareness," Elsevier Journal of Systems and Software (JSS), Vol. 127, PP. 168–176, May 2017. DOI=10.1016/j.jss.2017.02.004

[10] X. Yang, N. Wu, and J. Andrian, "Comparative Power Analysis of An Adaptive Bus Encoding Method on The MBUS Structure," Journal of VLSI Design, Vol. 2017, Article ID 4914301, PP. 1-7, May 2017. DOI=10.1155/2017/4914301

[11] X. Yang and J. Andrian, "A High Performance On-Chip Bus (MSBUS) Design and Verification," IEEE Trans. Very Large Scale Integr. Syst. (TVLSI), Vol. 23, Issue: 7, PP. 1350-1354, Sept. 2015. DOI=10.1109/TVLSI.2014.2334351

[12] Y. Zhang, X. Yang, and L. Wu, et al, "Hierarchical Synthesis of Approximate Multiplier Design for Field-Programmable Gate Arrays (FPGA)-CSRmesh System," Intl. Journal of Compt. Applications (IJCA), Vol. 180, No. 17 PP. 1-7, Feb. 2018. DOI=10.5120/ijca2018916380