Energy Efficient GPU Applications Through Computation Skip

Dongning Ma, Xun Jiao
Department of Electrical and Computer Engineering, Villanova University
{dma2,xjiao}@villanova.edu

***Abstract*—With the ever-increasing demand for multimedia processing applications running on GPUs, they have been an important target for energy minimization. In this paper, we propose a method to reduce energy consumption of GPU applications through computation skip. In particular, by exploiting the input operands of floating point operations in GPU applications, we have found many computations can be skipped. This is because their results can be pre-determined without actual executions of floating point units (FPUs). We then classify these computations into different categories and design corresponding circuits to detect them. By integrating the computation skip into two GPU applications, we have reduced the energy consumption by 32.67% - 44.53% on two popular GPU applications.**

I. Introduction

Graphics Processing Units (GPU) have been the primary platform for multimedia applications for their capability of parallel processing. Many multimedia applications are constantly sensitizing the energy-intensive FPUs in GPU pipeline, becoming a primary target for energy minimization. Many efforts have been made to increase the energy efficiency of FPUs in GPU pipeline. A popular approach is to use voltage scaling [3], [1]. While effective, voltage scaling may cause timing errors, resulting in an unacceptable output quality.

In this paper, we reduce the energy consumption of GPUs without introducing any errors. By examining the operands of floating point computations of some GPU applications, we have found the results of many computations can be predetermined without actual FPU executions. For example, if one operand of multiplication is 1, then the result is simply the other operand. We refer these computations as trivial computations and classify the trivial computations in GPU applications into several different cases. Based on this classification, we develop corresponding circuits that can detect and skip such trivial computations. Our contributions are as follows:

- We exploit the distribution of input operands in GPU applications, based on which we find trivial computations and classify them into different cases.
- We implement corresponding circuits that will detect trivial computations and return the results without the FPU. By integrating them with FPUs, energy intensive FPU executions can be avoided.
- We evaluate the effectiveness of computation skip on image processing applications and show that we can save 32.67% - 44.53% energy consumption on FPUs.

II. Proposed Approach

A. Classification of Trivial Computations

Primarily, trivial computations are classified into two categories: full-trivial computations and semi-trivial computations. The result of full trivial computations is fixed and pre-defined, which is not dependent upon the value of operands, while the result of semi-trivial computations is based on one or more operands in this computation.

In this paper, we have explored and classified the trivial computations of two most frequently-used FP operations of GPU applications, addition (ADD) and multiplication (MUL), as Table I displays, in which p and q refer to the operands involved in each operation.

B. Computation Skip

The computation skip runs continuously for each computation in FPUs with the skip circuits integrated with FPUs for the following steps:

- FPU and skip circuit will both read the computation data (operation type and operands).
- With the operands, the skip circuit will identify trivial computations per Table I for the corresponding operation.
- If any trivial computation is identified, the skip circuit will "clock-gate" the FPU to stop its ongoing computation for energy-saving. The multiplexer will then be taken over by the skip circuit to output the result.
- If no trivial computation is identified, there will be no computation skip, and the multiplexer will output result from the FPU.

The diagram of computation skip work-flow is illustrated in Fig. 1. The key component of the proposed computation skip approach is the skip circuit used to detect trivial computations based on operands. We have designed the skip circuits with several modules. This design can detect trivial computations of both ADD and MUL. For example, as Fig. 2 shows, there are three different operand identification modules available in the current design, which are used to identify "0", "1" and two inverse numbers, respectively. For ADD, the operation selection circuit will activate add-relevant modules, namely "0" and "inverse" as these two refer to the corresponding semi-trivial and full-trivial computations for ADD (Fig. I). For MUL, module "0" and "1" will be activated instead.

III. Experimental Results

A. Hit Rate

If one computation is regarded as trivial and subsequently skipped, it will be recorded as a "hit". Therefore the hit rate $r = \frac{N_t}{N}$, represents the ratio of trivial computations amount N_t and total computations N, in one GPU application. We have examined the hit rate of two GPU applications, Sobel Filter and Robert Filter, as shown in Fig. 3,

978-1-7281-2437-7/19/$31.00

TABLE I
CLASSIFICATION OF TRIVIAL COMPUTATIONS IN COMPUTATION SKIP

	Description	Expression	Result
MUL-Full	any of the operands equal to 0	$p = 0$ or $q = 0$	0
MUL-Semi	any of the operands equal to 1 or -1	$\|p\| = 1$ or $\|q\| = 1$	$\pm p$ or $\pm q$
ADD-Full	the two operands are inverse numbers of each other	$p + q = 0$	0
ADD-Semi	any of the operands equal to 0	$p = 0$ or $q = 0$	q or p

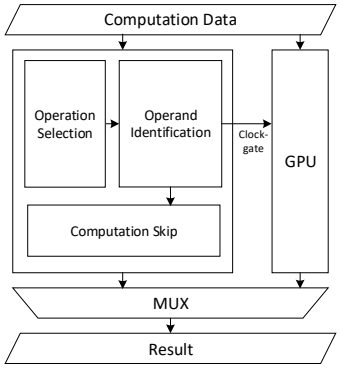

Fig. 1. Diagram of Computation Skip with GPU

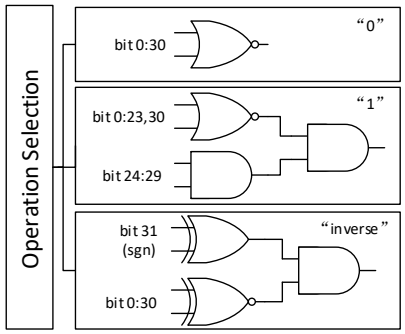

Fig. 2. Modules of Operand Identification Circuit

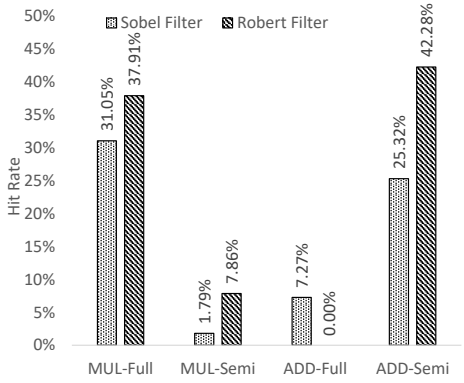

Fig. 3. Hit Rate of Trivial Computations in GPU Applications

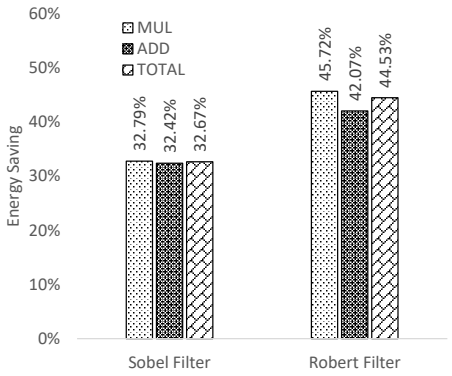

Fig. 4. Energy Savings of GPU Applications using Computation Skip

B. Energy Saving

$$\varepsilon = \frac{r_m(E_{fm} - E_{bm}) + r_a(E_{fa} - E_{ba})}{E_{fm} + E_{fa}} \quad (1)$$

The energy saving is calculated as Eq. 1, where r_m and r_a refer to hit rates of MUL and ADD trivial computations; E_{fm} (9891FJ) and E_{fa} (4742FJ) refer to energy consumption of each MUL and ADD computation of FPU; E_{bm} (12.5FJ) and E_{ba} (23.7FJ) refer to the energy consumption of each skip of the skip circuits. The energy consumption data of FPU are adopted from [2] under 1.0V while the energy of skip circuits are obtained from Synopsys PrimeTime. The energy savings are illustrated in Fig. 4. For both applications, the energy savings surpass 30% while Robert Filter displays a higher energy saving than Sobel Filter.

IV. CONCLUSION

In this paper, we propose a novel method to improve energy efficiency of GPU applications through computation skip. We identify the computation skip opportunities by exploiting the operands of floating point computations. We have found many computations results can be pre-determined without actual hardware execution. We therefore design skip circuits that can detect such "trivial computations" and integrate them with FPUs in GPU pipeline. We have achieved 32.67%-44.53% energy savings on two image processing applications.

REFERENCES

[1] Rong Ge et al. Effects of dynamic voltage and frequency scaling on a k20 gpu. In *2013 42nd International Conference on Parallel Processing*. IEEE, 2013.

[2] Amirali Ghofrani et al. Associative memristive memory for approximate computing in gpus. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2016.

[3] Jungseob Lee et al. Improving throughput of power-constrained gpus using dynamic voltage/frequency and core scaling. In *2011 International Conference on Parallel Architectures and Compilation Techniques*. IEEE, 2011.