A Parameterized Floating-Point Library Applied to Multispectral Image Clustering

Xiaojun Wang, Miriam Leeser, Haiqian Yu
Department of Electrical and Computer Engineering
Northeastern University, Boston, MA
xjwang,mel,hyu@ece.neu.edu, (617)373-5294

1 Introduction

One of the major applications of reconfigurable hardware is to accelerate algorithms which are initially implemented in software that runs on the general purpose processor. To achieve the acceleration, custom datapaths are built in reconfigurable hardware using either fixed-point or floating-point arithmetic. To make more parallel implementations possible and to reduce the power dissipation of the circuit, we want to minimize the bitwidth of each signal in the datapath. Hence, it is important to have fine-grained control over the datapath bitwidth throughout the whole design. Arbitrary fixed-point formats are not difficult to implement and are in common use. However, due to the inherent complexity of the floating-point representation, arbitrary floating-point formats are much harder to implement although they are no less desirable.

2 Related Work

A lot of work has been done on the implementation of floating-point units on FPGAs. Sahin et. al. [1] introduced signals to synchronize the flow of data through pipelines. Recent work by Dido et. al. [2] discussed datapath optimization with flexible floating-point formats that are optimal for each signal in the datapath. However, the formats used for their specific application are not general formats. Belanovic and Leeser [3] presented a parameterized floating-point library and its use in a K-means clustering application. Since floating-point division is not included in their library, the mean updating step of K-mean clustering has to be moved to the host for calculation, which diminishes the speedup of the hardware application. Our library is an extension to this research that adds floating-point division and square root.

3 Key Elements of Our Work

Our parameterized floating-point library is composed of three parts: format control, arithmetic operation, and format conversion. Format control includes modules `denorm` and `rnd_norm`. The first is used for denormalizing (introduction of the implied integer digit) and the second is used for rounding and normalizing. Format conversion includes modules `fix2float` and `float2fix`. The first is used for converting from fixed-point representation (both unsigned and signed) to floating-point representation and the second is in the other direction. Arithmetic operations include modules `fp_add`, `fp_sub` and `fp_mul`, which are used for floating-point addition, subtraction and multiplication respectively. We recently added floating-point division (`fp_div`) and floating-point square root (`fp_sqrt`). For both floating-point division and square root, we use the small table-lookup method with small multipliers [4] [5]. These algorithms are both small and elegant. Our result shows that these algorithms are very well suited to FPGA implementations, and lead to a good tradeoff of area and latency.

Some features of our library are:

- Our parameterized floating-point library is a superset of all the previously published floating-point formats including IEEE standard format.

- Our library is flexible. It supports the creation of custom format floating-point data-paths, as well as hybrid fixed and floating-point implementations.
- Our library is more complete than all other earlier work with a separate normalization unit, rounding with support for both "round to zero" and "round to nearest", and some error handling features.
- Each component in our library has synchronization signals to aid in the creation of pipelines.
- Both the area and the latency for our floating-point division and square root are small. For IEEE single precision format division, it takes 14 clock cycles to generate final results, where the clock period is 10ns. But it only takes approximately 2% of slices on a VirtexXC2V3000. Our IEEE single precision format square root has similar properties.

We implemented a floating-point K-means clustering algorithm and applied it to multispectral satellite images. K-means clustering is an iterative algorithm where the total number of clusters is known in advance. The algorithm works as follows. First means are initialized using a hierarchical method. During each iteration, each pixel of the image is assigned to the closest cluster based on the distance between each pixel and each of the K cluster centers. At the end of one iteration, the new mean of each cluster is calculated based on the new pixel assignments and is used for the next iteration as the center of each cluster. To obtain the new mean of each cluster, an accumulator and a counter are associated with each cluster. Once a pixel is assigned to a certain cluster, the value of the pixel is added to the accumulator and the counter is incremented by one. The new mean can be obtained by dividing the accumulator value by the counter value. In previous work [3] this mean updating step is done on the host because it requires floating-point division. With our new `fp_div` module, we are able to implement the mean updating in FPGA hardware. This greatly reduces the communication between host and FPGA board and further accelerates the runtime.

4 Conclusion

A library of fully parameterized floating-point hardware modules is presented in this paper. It includes floating-point format control, arithmetic operations, as well as the conversions between fixed-point and floating-point formats. Floating-point division and square root are supported. An implementation of the K-means clustering algorithm for multispectral image processing is presented, demonstrating the use of the library and the new added floating-point division module.

References

[1] I. Sahin, C. S. Gloster, and C. Doss, "Feasibility of floating-point arithmetic in reconfigurable computing systems," in *Military and Aerospace Applications of Programmable Devices and Technology International Conference*, 2000.

[2] J. Dido, N. Geraudie, L. Loiseau, O. Payeur, Y. Savaria, and D. Poirier, "A flexible floating-point format for optimizing data-paths and operators in FPGA based DSPs," in *International Symposium on Field-Programmable Gate Arrays*, pp. 50–55, 2002.

[3] P. Belanovic and M. Leeser, "A library of parameterized floating-point modules and their use," in *12th International Conference on Field Programmable Logic and Application*, 2002.

[4] P. Hung, H. Fahmy, O. Mencer, and M. J. Flynn, "Fast division algorithm with a small lookup table," in *Asilomar Conference*, 1999.

[5] M. D. Ercegovac, T. Lang, J.-M. Muller, and A. Tisserand, "Reciprocation, square root, inverse square root, and some elementary functions using small multipliers," in *IEEE Transactions on Computers,*, vol. 2, pp. 628–637, 2000.