

January 01, 2007

Floating-point divide and square root for efficient FPGA implementation of image and signal processing algorithms

Xiaojun Wang
Northeastern University

Miriam Leeser
Northeastern University

Recommended Citation

Wang, Xiaojun and Leeser, Miriam, "Floating-point divide and square root for efficient FPGA implementation of image and signal processing algorithms" (2007). *Research Thrust R3 Presentations*. Paper 8. <http://hdl.handle.net/2047/d10009073>

This work is available open access, hosted by Northeastern University.



This work was supported in part by Gordon-CenSSIS, the Bernard M. Gordon Center for Subsurface Sensing and Imaging Systems, under the Engineering Research Centers Program of the National Science Foundation (Award Number EEC-998621).

Floating-Point Divide and Square Root for Efficient FPGA Implementation of Image and Signal Processing Algorithms

Xiaojun Wang, Miriam Leeser
 xjwang@ece.neu.edu mel@ece.neu.edu

Reconfigurable Computing
 Laboratory



Abstract

Division and square root are important operations in many high performance signal processing applications. We have implemented floating point division and square root based on Taylor series for the variable precision floating point library developed at the Reconfigurable Computing Laboratory at Northeastern. Our result shows that they are very well suited to FPGA implementations, and lead to a good tradeoff of area and latency. We implemented a floating-point K-means clustering algorithm and applied it to multispectral satellite images. The mean update is moved from host to FPGA hardware with the new fp_div module to reduce the communication between host and FPGA board and further accelerate the runtime. We are also working on QR factorization using both floating point divide and square root.

State of the Art

- [1] P. Hung, H. Fahmy, O. Mencer, and M. J. Flynn, "Fast division algorithm with a small lookup table," *Asilomar Conference*, 1999
 - [2] M. D. Ercegovic, T. Lang, J.-M. Muller, and A. Tisserand, "Reciprocation, square root, inverse square root, and some elementary functions using small multipliers," *IEEE Transactions on Computers*, vol. 2, pp. 628-637, 2000
- Both algorithms are simple and elegant
 - Based on Taylor series
 - Use small table-lookup method with small multipliers
 - Very well suited to FPGA implementations
 - BlockRAM, distributed memory, embedded multiplier
 - Lead to a good tradeoff of area and latency
 - Can be fully pipelined
 - Clock speed similar to all other components in the floating point library

QR Factorization

Givens Rotation

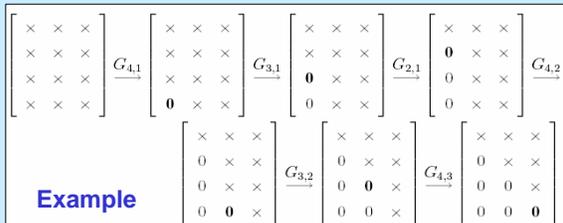
function: $[c, s] = \text{givens}(a, b)$

if $b = 0$
 $c = 1; s = 0$
 else
 if $|b| > |a|$
 $\tau = a/b; s = 1/\sqrt{1+\tau^2}; c = s\tau$
 else
 $\tau = b/a; c = 1/\sqrt{1+\tau^2}; s = c\tau$
 end
 end
 end givens

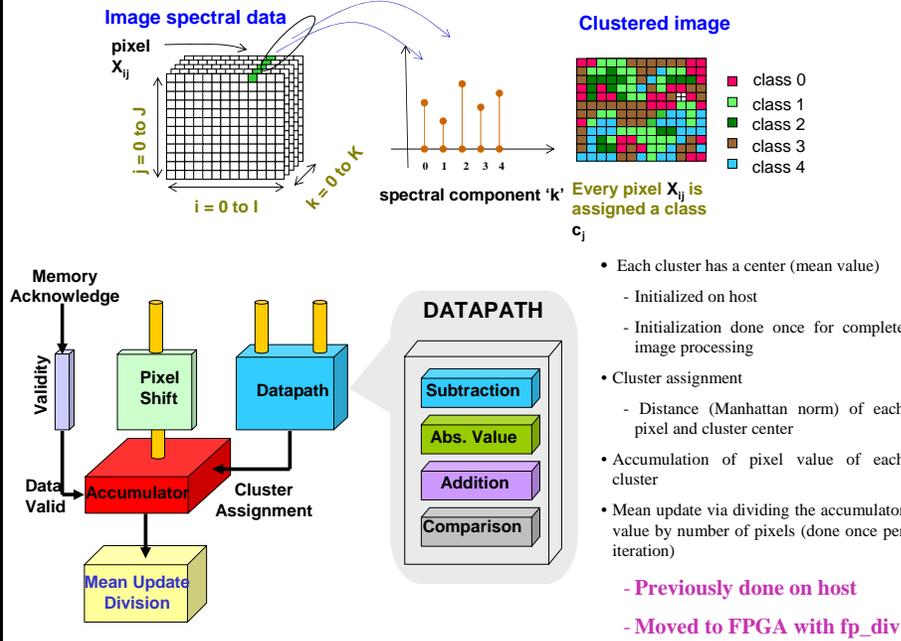
$$r = \sqrt{a^2 + b^2}$$

c: cosine; s: sine

Divide and square root are required



An Application: K-Means Clustering



Experimental Results

Floating Point Divider and Square Root

FP Square Root on a XC2V6000
 (The last two are IEEE single/double precision floating point format)

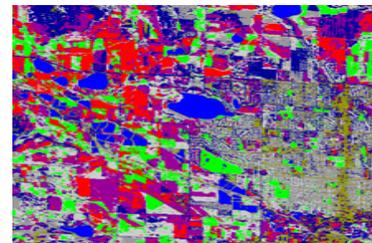
Floating Point Format	8 (2,5)	16 (4,11)	32 (8,23)	64 (11,52)
# of slices	1%	1%	1%	4%
# of BlockRAM	2%	2%	2%	80%
# of Embedded Multiplier	2%	4%	6%	16%
Clock period (ns)	6	7	8	10
Maximum frequency (MHz)	165	139	125	103
Latency # of clock cycles	9	12	13	17
Latency (ns)	55	86	104	165
Throughput (million results/sec)	165	139	125	103

FP square root is small, has small latency and high throughput
 The result for FP Divider is similar

K-means Clustering

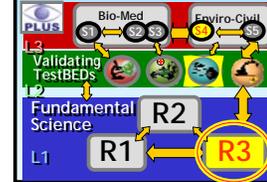
- 8 cluster, 8 channel, 8 bit per channel
- 37% slices, 81% blockRAMs, 44% embedded multipliers of Virtex2V6000
 - More than 2150x faster than software implementation for core computation only
 - 11x faster than software implementation, including time to configure FPGA and move data between board and host PC.

Clustered Output Image



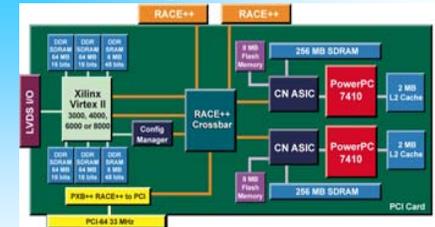
Research Level 1 Thrust R3A

This work is a part of CenSSIS Research Thrust R3A. Due to inherent limitations of the fixed-point representation, it is desirable to perform arithmetic operation in the floating-point format for many image and signal processing algorithms. Our goal is to develop a parameterized floating-point library with reconfigurable hardware to speed up those image and signal processing algorithms such as remote sensing application.



Computer Systems, Inc.
MERCURY
 This project is funded by Mercury Computer Systems, Inc.

Reconfigurable Hardware



- Features of Mercury Atlanta Board:
- One Xilinx Virtex II XC2V6000-5 FPGA (144 on-board BlockRAMs, 144 embedded multipliers)
 - 12MB DDR SRAM and 256 MB DDR SDRAM
 - dual-processor PCI module with two PowerPCs

Technology Transfer

- The floating-point library has been used by many users such as Los Alamos National Laboratory, Sandia National Laboratory, Kodak, Systron Donner, L3 Communications, and Magnetic Analysis Corp since it was provided on the web

Conclusions

- The library includes fully pipelined and parameterized hardware modules for floating point arithmetic
- New module fp_div and fp_sqrt have small area and low latency, are easily pipelined
- Applications using fp_div and fp_sqrt show great speedup vs. software implementation

Further Information

<http://www.ece.neu.edu/groups/rpl/projects/floatingpoint/index.html>
 Email us: xjwang@ece.neu.edu