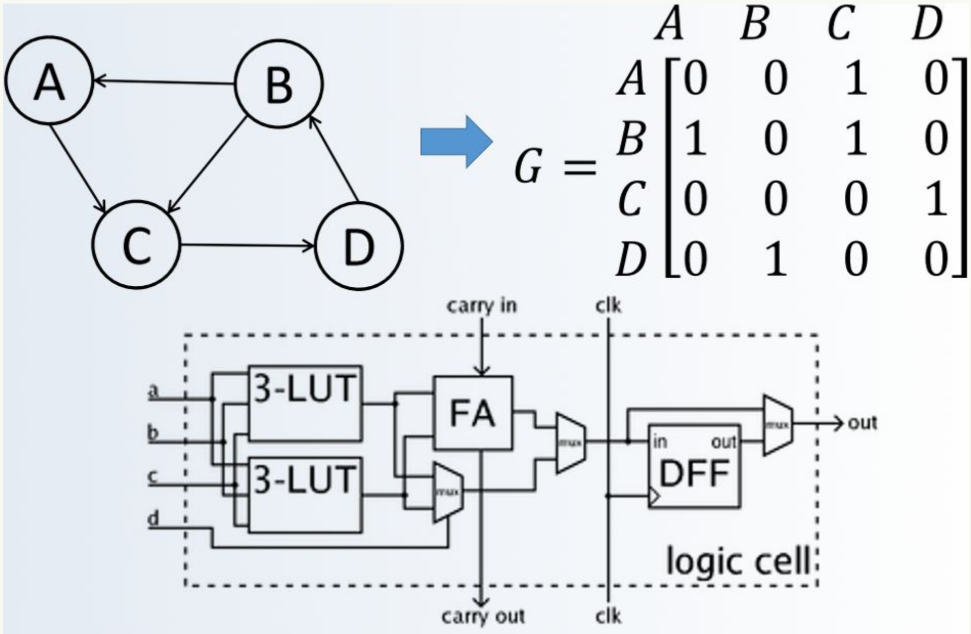
Exploring SPMV Acceleration for heterogeneous CPU-FPGA architecture

Blaise Tine

September 2016



Abstract

This document describes a research project proposal for the Advanced Microarchitecture class. it lays down the background and motivation behind the project and provides some metrics and milestones towards realizing this work.

Background and Motivation

In recent years, the agile development methodology in data centers has created a new application domain for FPGAs due to their reconfigurability and energy efficiency and is capturing the interest of both academia [11] and the industry [12]. New heterogeneous computing platforms [13] have emerged to increase the communication bandwidth and reduce latency between CPU and FPGAs, opening opportunities for new efficient FPGA accelerators architectures for domain specific applications.

Graph analytics remains one of the largest applications in production data center today and the need to increase computational throughput and efficiency to address the growing size of the data remains a challenging task, both at software programming layer and hardware architecture. There is a need for productive, robust implementation of various graph algorithms [1] [2] [3] as well as the need for an efficient hardware abstraction to support it. GraphMat [1] programming model offers an efficient abstraction of graph algorithms using linear algebra operations for efficient computation on single node multi-core platforms. This programming paradigm extensively uses doubly compressed column-based sparse matrices (DCSC) [4] for graph representation and most the computation is based on sparse matrix - sparse vector multiplication (SPMV).

SPMV acceleration has been studied extensively in the past decade and continues to receive improvements or mutation as new hardware ecosystems or technology emerge due to the critical importance of the kernel. Various implementation of SPMV kernels have been studied and implemented on both GPU's [5] [6] and FPGA's [7] [8] [9] [10]. GraphMat running on an heterogeneous single node CPU-FPGA platform offers a new architecture space for SPMV design exploration because of the following reasons:

- A heterogeneous CPU-FPGA platform with cache coherency and low latency data transfer presents a new architecture ecosystem for SPMV FPGA design, different existing PCIe-based implementation.
- GaphMat programming paradigm enforces an iterative computation model in which CPU and FPGA share the sparse bitmap-based input vector for efficient computation on the CPU.
- There is currently no existing work that evaluates the effectiveness of on heterogeneous CPU-FPGA platform for linear algebra or similar computation benchmark. This work could provide new insights about unexpected new challenges when targeting the platform.

The primary objective in the work will be to explore different SPMV architecture designs for accelerating SPMV on the heterogeneous CPU-FPGA platform. The design variation will be primarily based on matrix format (column-based versus row-based), but we will also explore other design configurations such as the number of memory ports, number of compute units and the input buffers width. The implementation will be done using CHDL [14], a C++ hardware description library which provides similar HDL functionality as Chisel [15], BlueSpec [17] or MyHDL [18], with the advantage of native simulation speed and direct debugging of SystemC [16].

Evaluation Methodology

We propose the following methodology for evaluating our FPGA implementations of SPMV.
The experimental setup will consist of the Intel QuickAssist FPGA Platform with the following specifications:

Features	**Specification**
Processor	Intel Xeon ES-2680 v2
Cores	20 cores (x2 threads)
Caches	L1 (2 x 32 KB), L2 (256 KB), L3 (25600 KB)
Memory	32 GB DDR3

FPGA	Altera Stratix V (234K LUTs, 393K FFs, 52MB RAM)
FPGA/CPU Bandwidth	R/W 4913 MB/s, R/only 4370 MB/s, W/only 3374 MB/s
FPGA/CPU Latency	Cached 519.96 ns, Uncached 227.78 ns

The performance will be evaluated using four graph algorithm implemented in GraphMat:

- BFS - Breadth First Search
- PageRank - Web Analytics page ranking
- SSSP - Single Source Shortest Path
- Conjugate Gradient - Linear equation solver

We will be using the graph datasets from Graph500 (http://www.graph500.org)
We will be gathering real-time performance metrics as well as FPGA area costs for the various design configurations.

Milestones

We anticipate the following milestones to provide a reasonable general breakdown of the various tasks associated with this work together with their time estimate.

1. Implement CHDL Support for behavioral verilog codegen
2. First Presentation checkpoint
3. Implement generic SPMV AFU using CHDL
4. Profiling/tuning SPMV models
5. Final Presentation
6. Writing Final Report

References

[1] Sundaram, Narayanan and Satish, Nadathur and Patwary, Md Mostofa Ali and Dulloor, Subramanya R. and Anderson, Michael J. and Vadlamudi, Satya Gautam and Das, Dipankar and Dubey, Pradeep. “GraphMat: High Performance Graph Analytics Made Productive”. In Proc. VLDB ENdow. July 2015.

[2] JSatish, Nadathur and Sundaram, Narayanan and Patwary, Md. Mostofa Ali and Seo, Jiwon and Park, Jongsoo and Hassaan, M. Amber and Sengupta, Shubho and Yin, Zhaoming and Dubey, Pradeep. “Navigating the Maze of Graph Analytics Frameworks Using Massive Graph Datasets ”. Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data.

[3] Fu, Zhisong and Personick, Michael and Thompson, Bryan. “MapGraph: A High Level API for Fast Development of High Performance Graph Analytics on GPUs”. In ACM 2014.

[4] A. Buluc and J. R. Gilbert, "On the representation and multiplication of hypersparse matrices," Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on 2008.

[5] Carl Yang, Yangzihao Wang, John D. Owens. “Fast Sparse Matrix and Sparse Vector Multiplication Algorithm on the GPU”. In Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International.

[6] Greathouse, Joseph L. and Daga, Mayank. “Efficient Sparse Matrix-vector Multiplication on GPUs Using the CSR Storage Format“. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis 2014.

[7] Jeremy Fowers, Kalin Ovtcharov, Karin Strauss, Eric S Chung, Greg Stitt. “A high memory bandwidth fpga accelerator for sparse matrix-vector multiplication”. Field-Programmable Custom Computing Machines (FCCM), 2014 IEEE 22nd Annual International Symposium.

[8] Y. Umuroglu and M. Jahre, “An energy efficient column-major backend for FPGA SpMV accelerators”. In 2014 IEEE 32nd International Conference on Computer Design (ICCD).

[9] Dorrance, Richard and Ren, Fengbo and Markovi, Dejan. “A Scalable Sparse Matrix-vector Multiplication Kernel for Energy-efficient Sparse-blas on FPGAs”. Proceedings of the 2014 ACM/SIGDA International Symposium on Field-programmable Gate Arrays.

[10] Zhuo, Ling and Prasanna, Viktor K. “Sparse Matrix-Vector Multiplication on FPGAs ”. In Proceedings of the 2005 ACM/SIGDA 13th International Symposium on Field-programmable Gate Arrays.

[11] Choi, Young-kyu and Cong, Jason and Fang, Zhenman and Hao, Yuchen and Reinman, Glenn and Wei, Peng. “A Quantitative Analysis on Microarchitectures of Modern CPU-FPGA Platforms ”. Proceedings of the 53rd Annual Design Automation Conference 2016.

[12] Putnam, Andrew and Caulfield, Adrian M. and Chung, Eric S. and Chiou, Derek and Constantinides, Kypros and Demme, John and Esmaeilzadeh, Hadi and Fowers, Jeremy and Gopal, Gopi Prashanth and Gray, Jan and Haselman, Michael and Hauck, Scott and Heil, Stephen and Hormati, Amir and Kim, Joo-Young and Lanka, Sitaram and Larus, James and Peterson, Eric and Pope, Simon and Smith, Aaron and Thong, Jason and Xiao, Phillip Yi and Burger, Doug. “A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services”. In SIGARCH Computing 2014.

[13] Oliver, Neal and Sharma, Rahul R. and Chang, Stephen and Chitlur, Bhushan and Garcia, Elkin and Grecco, Joseph and Grier, Aaron and Ijih, Nelson and Liu, Yaping and Marolia, Pratik and Mitchel, Henry and Subhaschandra, Suchit and Sheiman, Arthur and Whisonant, Tim and Gupta, Prabhat. “A Reconfigurable Computing System Based on a Cache-Coherent Fabric," 2011 International Conference on Reconfigurable Computing and FPGAs”.

[14] Chad D. Kersey, “CHDL: Open-Source C++ Hardware Design Library”,
url: https://github.com/cdkersey/chdl

[15] Jonathan Bachrach, Huy Vo, Brian Richards, Yunsup Lee, Andrew Waterman, Rimas Avižienis, John Wawrzynek, and Krste Asanović. *“Chisel: constructing hardware in a Scala embedded language”*. In Proceedings of the 49th Annual Design Automation Conference (DAC '12).

[16] Preeti Ranjan Panda. SystemC: a modeling platform supporting multiple design abstractions. In Proceedings of the 14th international symposium on Systems synthesis (ISSS '01)

[17] Arvind. *“Bluespec: A language for hardware design, simulation, synthesis and verification”* Invited Talk. In Proceedings of the First ACM and IEEE International Conference on Formal Methods and Models for Co-Design (MEMOCODE '03)

[18] Jan Decaluwe. “*MyHDL: a python-based hardware description language*”. Linux J. 2004, 127 (‘04)