Paper 1

Title: Is HLS ready for business? A computational finance case study

Authors: Gordon Inggs, Shane Fleming, David Thomas and Wayne Luk

Abstract: High Level Synthesis (HLS) tools for Field Programmable Gate Arrays (FPGAs) have made considerable progress, and are now sufficiently mature that a novice developer could create a functionally correct implementation with limited understanding of the target hardware. In this case study, a novice developer considers a benchmark of financial problems for implementation upon FPGA via HLS. This novice starts by extending an existing implementation for a CPU or GPU using tools such as Xilinx's Vivado HLS, the Altera OpenCL SDK or Maxeler's MaxCompiler. When their direct source code translation inevitably didn't meet performance expectations, this developer then applies optimisations such as exploiting task or pipeline parallelism as well as C-slowing. When a combination of these optimisations are considered for a range of devices and process technologies, an acceleration of up to 220 times is achieved using these tools, the sort of acceleration expected of custom architectures. Compared to the 31 times improvement shown by an optimised Multicore CPU implementation, the 60 times improvement by a GPU and 207 times by a Xeon Phi, these results suggest that HLS is indeed ready for industrial adoption.

Paper 2

Title: Low-latency option pricing using systolic binomial trees

Authors: Aryan Tavakkoli and David Thomas

Abstract: This paper presents a novel reconfigurable hardware accelerator for the pricing of American options using the binomial-tree model. The proposed architecture exploits both pipeline and coarse-grain parallelism in a highly efficient and scalable systolic solution, designed to exploit the large numbers of DSP blocks in modern architectures. The architecture can be tuned at compile-time to match user requirements, from dedicating the entire FPGA to low latency calculation of a single option, to high throughput concurrent evaluation of multiple options. On a Xilinx Virtex-7 xc7vx980t FPGA this allows a single option with 768 time steps to be priced with a latency of less than 22 micro-seconds and a pricing rate of more than 100 K options/sec. Compared to the fastest previous reconfigurable implementation of concurrent option evaluation, we achieve an improvement of 65X in latency and 9X in throughput with a value of 10.7 G nodes/sec, on a Virtex-4 xc4vsx55 FPGA.

Paper 3

Title: Efficient FPGA implementation of digit parallel online arithmetic operators

Authors: Kan Shi, David Boland and George A. Constantinides

Abstract: Online arithmetic has been widely studied for ASIC implementation. Online components were originally designed to perform computations in digit serial with most significant digit (MSD) first, resulting in the ability to chain arithmetic operators together for low latency. More recently, research has shown that digit parallel online operators can fail more gracefully when operating beyond the deterministic clocking region in comparison to operators with conventional arithmetic. Unfortunately, the utilization of online arithmetic operators in the past has required a large area overhead for FPGA implementation. In this paper, we propose novel approaches to implement the key primitives of online arithmetic, adders and multipliers, efficiently on modern Xilinx FPGAs with 6-input LUTs and carry resources. We demonstrate experimentally that in comparison to a direct RTL synthesis, the proposed architectures achieve slice savings of over 67% and 69%, and speed-ups of over 1.2× and 1.5× for adders and multipliers, respectively. As a result, the area overheads of using online adders and multipliers in place of traditional arithmetic primitives is reduced from 8.41× and 8.11× to 1.88× and 1.84× respectively. Finally, because an online multiplier generates MSDs first, we also demonstrate the method to create an online multiplier with a reduced precision output that is smaller than a traditional multiplier producing the same result. We show that this can lead to silicon area savings of up to 56%.

Paper 4

Title: Parallel resampling for particle filters on FPGAs

Authors: Shuanglong Liu, Grigorios Mingas and Christos-Savvas Bouganis

Abstract: Particle filters (PFs) are a set of algorithms that implement recursive Bayesian filtering, which represent the posterior distribution by a set of weighted samples. Resampling is a fundamental operation in PF algorithms. It consists of taking a population of samples and reconstructing it based on the weights attached to each sample, favouring the samples with large weights. However, resampling is computationally intensive when the number of samples is large and, most importantly, it is not inherently parallelizable like the other steps of the particle filter. Parallel computing devices such as Graphics Processing Units (GPUs) and Field Programmable Gate Arrays (FPGAs) have been proposed to accelerate resampling. In this paper, we propose novel parallel architectures that map four state-of-the-art resampling algorithms (systematic, residual systematic, Metropolis and Rejection resampling) to a FPGA. FPGA-specific optimisations are introduced to further optimize the performance of the above systems. The proposed architectures are implemented in a Virtex-6 LX240T FPGA device with half-utilization of logic resources. Compared to the respective state-of-the-art implementations on an NVIDIA K20 GPU, the achieved speedups are in the range of 1.7x-49x.