Parallel Programming for FPGAs

Prerequisite(s): ECE4100/ECE6100/CS4290/CS6290, or equivalent
Instructor: Cong (Callie) Hao

Course Objectives

FPGAs have been serving as important computing resources for decades with significant benefits in low-power, high-throughput, and low-latency applications. Especially, in the current AI revolutionary era, the massive computing and big-data processing requirements have tremendously and exponentially increased. As an example, deep neural networks (DNNs), have demonstrated extremely promising results in various areas but also come with high computational demands for both cloud and IoT platforms. Other examples include large-scale graph processing, scientific computing such as physics simulation, and bioinformatic and chemistry applications such as modular modeling. With the emerging needs for low-power high-performance computing in various domains, FPGAs have been attracting ever-growing interests in specialized hardware accelerators in both industry and academia. Examples include Amazon's AWS with F1 FPGA clusters, Microsoft Azure FPGA cloud, Xilinx's FPGA adaptive compute cluster at multiple universities, etc.

This course will present recent advances towards the goal of efficient and high-performance FPGA parallel programming using High-Level Synthesis (HLS) for computation intensive applications. Specifically, it will provide an overview of FPGA architectures, discuss its underlining structures, and highlight key technologies in achieving high-performance parallel programming on FPGA. To improve FPGA development productivity, this course will adopt behavioral-level programming language, C/C++, via HLS tools for agile development. It will discuss recent achievements of FPGA accelerations in multiple domains not limited to DNNs to broaden the view. Multiple design examples will be provided to let learners quickly get started on basic designs, but with large room to improve which is left for exploration. Optionally, this course will include accelerator/algorithm co-design, being an extremely important and promising research topic.

Learning Outcomes

As part of this course, students will: understand the basic architecture of traditional and modern FPGAs and System-on-Chips (SoCs); understand the key techniques of improving the design performance; understand tradeoffs between various hardware architectures and platforms; learn about HLS programming language using C/C++; learn about micro-architectural knobs such as precision, data reuse, memory optimization, and parallelism to architect FPGA accelerators given target area-power-performance metrics; evaluate the implemented design on FPGA boards and iterate on improving the design quality; and understand future trends and opportunities for FPGAs for a diverse range of applications such as GNNs, scientific computing, medical electronics, cybersecurity systems, and wireless communications.

Course Structure

The course will involve a mix of lectures interspersed with heavy paper reading and discussions. A semester long programming-heavy project will focus on developing an FPGA accelerator using HLS for DNN or GNN algorithms.

Course Text

The material for this course will be derived from the following texts:

1. Kastner, Ryan, Janarbek Matai, and Stephen Neuendorffer. "Parallel programming for FPGAs." arXiv preprint arXiv:1805.03648 (2018).

2. Papers from recent FPGA and computer architecture conferences: ISCA, MICRO, HPCA, ASPLOS, DATE, DAC, ICCAD, ICCD.
3. Papers from ML conferences: ICML, NeurIPS, ICLR, CVPR

Syllabus and Outline

1. **Overview of FPGA**
   a. FPGA architecture
   b. FPGA programming tutorial (Vivado)
2. **Overview of High-Level Synthesis (HLS)**
   a. HLS introduction
   b. Vitis HLS tutorial
3. **Overview of Machine Learning**
   a. Deep Neural Networks
   b. Graph Neural Networks
4. **FPGA Design Techniques (I)**
   a. Data precision and model quantization
   b. Loop optimizations and array partitioning
   c. Data reuse and model tiling
5. **FPGA Design Techniques (II)**
   a. Storage and memory access
   b. Data streaming
   c. C/RTL Co-simulation
6. **Software/Hardware Co-design**
   a. Hardware-aware algorithm design
   b. Neural architecture search
7. **Future Trends**
   a. Spatial temporal GNNs
   b. Scientific computing
   c. Analog Accelerators

Course Grading

Lab Assignments	30% (3 Labs - 10% each)
Paper Presentation	10%
Project	60%
	Project proposal: 10%
	Mid-term report: 5%
	Source code: 15%
	Presentation: 10%
	Final report: 20%

Course Policies

Attendance and Absence. Students are expected to attend all lectures and exams. If one has a documented emergency or a university mandated reason, get in touch with the instructor before (preferable) or latest by the day of the exam.

Learning Accommodations. If needed, we will make classroom accommodations for students with disabilities. These accommodations should be arranged in advance and in accordance with the office of Disability Services (http://www.adapts.gatech.edu)

Honor Code. Students are expected to abide by the Georgia Tech Academic Honor Code (http://www.policylibrary.gatech.edu/student-affairs/academic-honor-code). Honest and ethical behavior is always expected. All incidents of suspected dishonesty will be reported to and handled by the office of student affairs. Students will have to do all assignments individually unless explicitly told otherwise. Students may discuss with classmates but may not copy any solution (or any part of a solution).