University of Toronto

ECE532 2018-2019

Final Proposal

Group #	2	Date	2019.01.31

Project Name	Fog GPU Computing
Team Member	Lichen Liu Lin Sun Feiyu Ren
TA	Mohammad Tabrizi

1.0 Introduction

Cloud Computing has been arising these days, it separates the computing resources from clients' actual business logic such that the client only needs to focus on implementing their business ideas without worrying too much about IT side of things. However, the cloud computing is more common to be a service from large companies, where the cloud servers are located remotely in the data center. A gap exists between such kind of cloud computing where throughput is deemed more important than latency, and the kind of computing that the team has foreseen, where latency also matters.

Image a scenario looks like this. Peter and Ray live together in an apartment, and they are both relying on GPUs to do things: Peter wants to train his machine learning models, whereas Ray is doing a lot of video processing. High End video cards are expensive nowadays, so that Peter and Ray would like to share a single GPU card, but they would not want to plug and unplug every time they use it.

The team is proposing another kind of cloud computing solution, Fog(Cloud on ground) GPU Computing, that would perfectly fulfill their needs, that a single GPU is shared with multiple clients using local network connections.

2.0 Project Team

The team is made up of three experienced computer engineering students, having strong background in FPGA-related fields and software programming.

Lichen Liu:

1. Interned as a developer in the placer team for Altera Quartus Fitter.
2. Have hands-on experience and theoretical knowledge on compilers.

Feiyu Ren:

1. Experience in game engine design and shader programming.
2. Have hands-on experience and theoretical knowledge on compilers.

Lin Sun:

1. Experience in FPGA and machine learning deployment on heterogeneous clusters.

The teams' FPGA-related experience can be directly applied to the HDL development on FPGA. The team can also use its knowledge and experiences on compiler to write a simple C-like language to GPU assembly compiler to ease the use of the GPU.

3.0 Project Description

The project is consisted of two subsystems: a GPU computation server, and a client application, which are connected via WiFi. Due to the time limitation, the client side application will only use the GPU computation server to compute graphics-related tasks, as opposed to other tasks such as machine learning training tasks.

The GPU computation server is in charged of the cloud computing, and is built entirely on a FPGA chip. Most of the area on the chip is for the GPU implementation, that supports a simple instruction set architecture. The subsystem has a microblaze processor, used mainly for network connection and GPU scheduling purposes. It also has a WiFi module, for peer-to-peer connection with the client.

The client application side is consisted of a microblaze processor, a WiFi module for peer-to-peer connection with the GPU computation server, and a VGA module to display the output produced by the GPU, e.g. the picture rendered by the GPU.

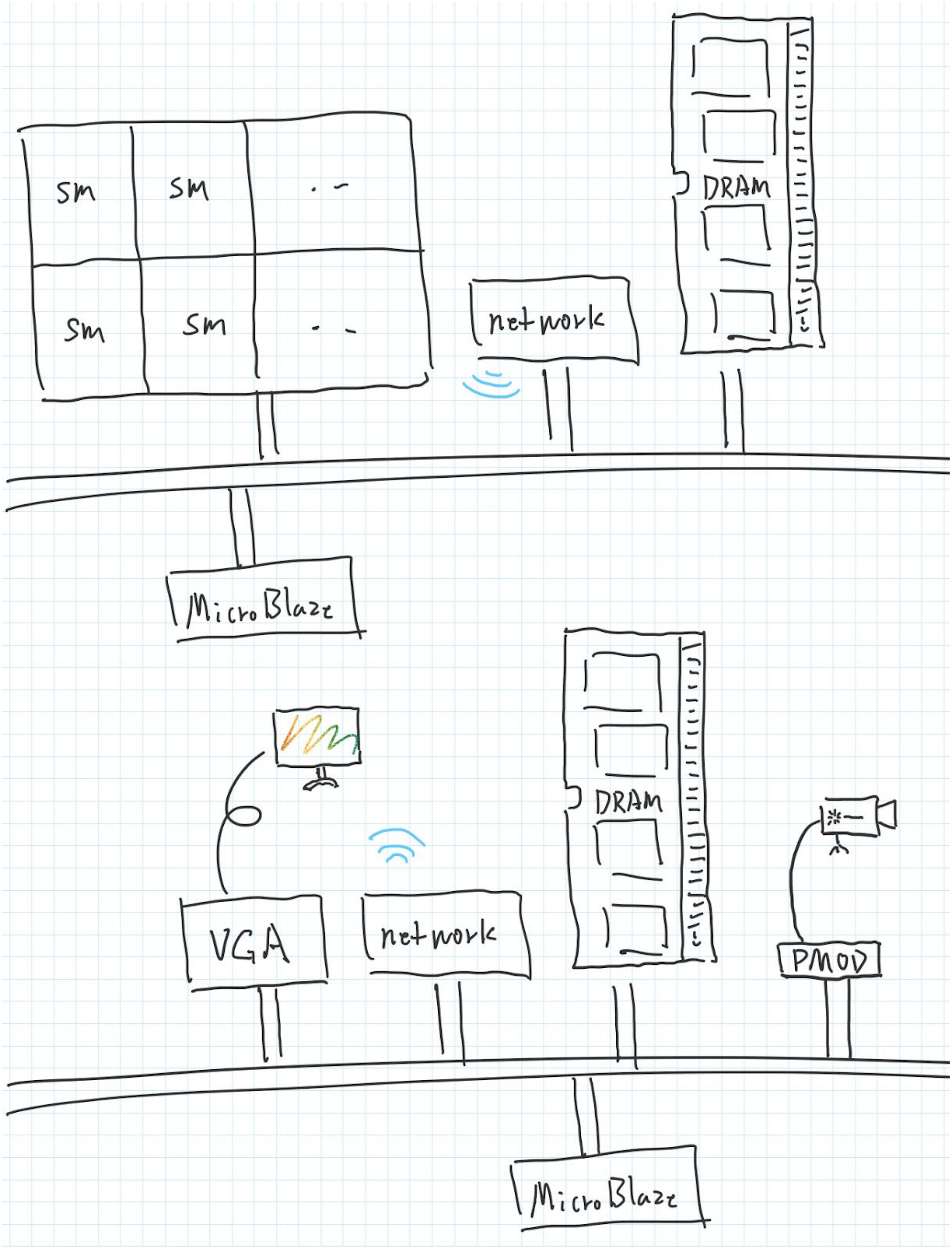
3.1 Functional Requirements and Features

1. GPU server and client must be on different physical machines.
2. GPU server and client must be using network to connect.
3. GPU must be general purposed.
4. Client must be able to display the result of the computation.

3.2 Acceptance Criteria

1. GPU and client are implemented on two different boards
2. GPU communicates with host through networks
3. GPU should have its own instruction set.
4. Client can use VGA to display the rendered picture to the monitor.

3.3 System Block Diagram



4.0 Testing

The testing of our design will be divided into two stages, simulation and real world. For the simulation, the team will first implement the design in verilog and write extensive testbenches to verify the functionality of the design, tools including Verilog and myHDL has been investigated by the team to enhance the efficiency of such verification. For real world testing, the team will implement tests for every functional blocks incrementally and test every block individually, until the complete design is verified.

5.0 Project Complexity

Feature/Operation/Ip core	Points
Microblaze	1
Custom IP - GPU core	1
DDR	1
VGA	1
Network	1
Infrared Light Sensor	1
Compiler for converting C-like language to the customized GPU assembly	TBD

6.0 Risk

A major risk to this project would be the time pressure for each milestone. As all team members are enrolled in the ECE496 Capstone and CSC418 Computer Graphics, which are two LAB-intensive courses, the time management might be a potential risk to this project. To guarantee that all deliverables can be done on time, and minimize the impact of unexpected amount of time of debugging, the team will follow a test-driven development approach, and will start with a simple design that works at the first place and then add things into it. This guarantees that at any moment of time, the team would have a completely-working system, which can reduces impact of unexpected amount of time of debugging.

Another possible risk is the lack of professional knowledges on GPU. The team will do a lot of researches to gain knowledge on it.

7.0 Resources Requirement

The project requires:

1. Two FPGA boards: one Nexys 4 DDR board and one Nexys 4 Video board.
2. VGA monitors.
3. Infrared Light Sensor

8.0 Milestones

Milestone #1

1. Shown testbench with general networking layer working.

Milestone #2

1. Shown testbench with general VGA working.

Milestone #3

1. Get custom processor working in hardware.

Milestone #4

1. Get vertex shader working in hardware with software driver.

Milestone #5

1. Mid-Project Demo show pixel shader working.

Milestone #6

1. Get pixel shader working.
2. Implemented client to connect with GPU server.

Milestone #7

1. Final Demo Done! It all works.