CSEE 4840 Embedded System Project Design

Accelerating VGG16 Network on FPGA

Wenqi Jiang(wj2285)

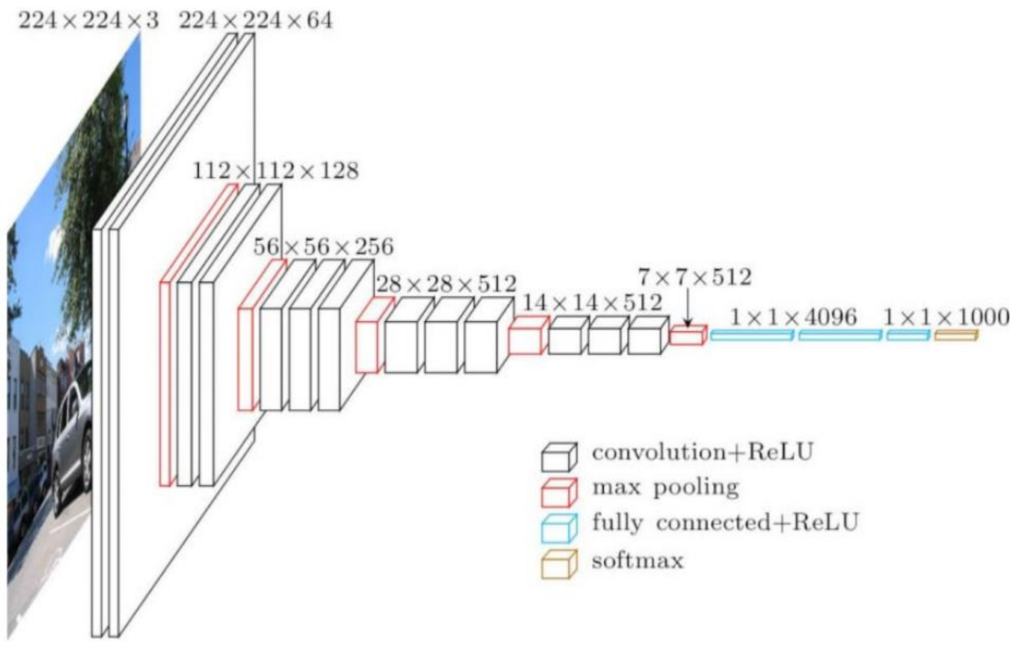
Manqi Yang(my2577)

Ke Xu(kx2141)

Xingyu Hou(xh2371)

1 Introduction

Convolutional Neural Networks (CNN) are widely used in Computer Vision tasks such as Image Classification, Object Detection and Semantic Segmentation. The amount of processing required by CNNs leads to an increasing need on dedicated hardwares. FPGAs, due to its programmable property and hardware acceleration capacity, can serve as the intermediate product accelerating CNNs between General Purpose Graphics Processing Unit (GPGPU) and Application-specific Integrated Circuit (ASIC).
In this project,we will accelerate VGG16 CNN on our FPGA.
VGG16 is a CNN model that achieve 92.7% top-5 test accuracy.Th basic structure of VGG16 is shown below.

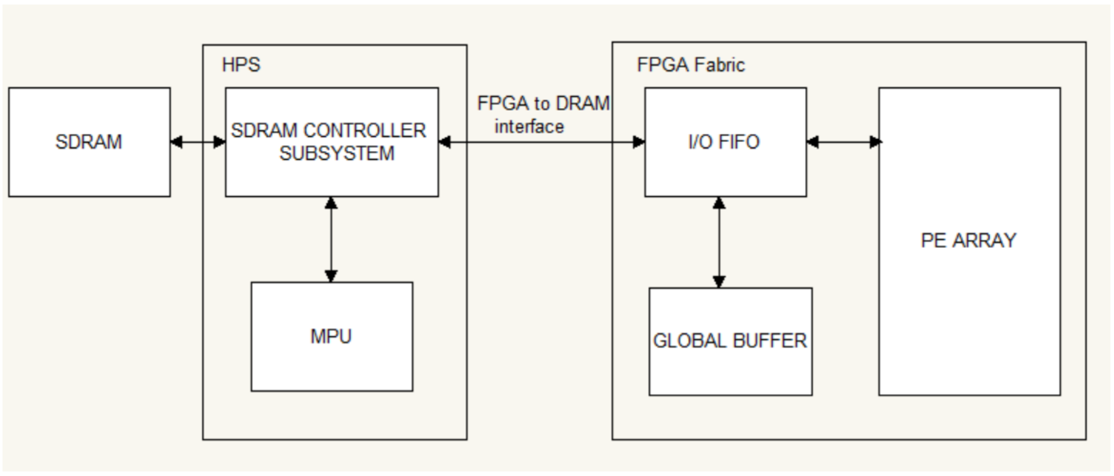


VGG16 Architecture

In the structure,we have four different types of layers as shown above.Over 90% of the computation is done in convolution layer.So the crucial part of acceleration is to build hardware that do convolution computation effieciently.
In this project, we will implement row stationary dataflow method(which we will talk in hardware part later) to accelerate VGG16 network on DE1-SOC board.This project involves both software and hardware design.

2 Implementation

The basic floorplan of the whole system structure is shown below.In our project,most of CNN computation work will be done in hardware.Software will control the dataflow between external SDRAM and FPGA.



2.1 Software

In this project,we need to use software to load weight and input feature from DRAM to FPGA.Since the throughput of the system is extremely large,we need to use DMA method to load data directly from DRAM to FPGA without transferring data to HPS.To achieve this goal,we need to connect our FPGA fabric to FPGA to HPS SDRAM Interface using AXI bus or Avalon-MM bus. Then,we could use MPU to control the SDRAM Controller subsystem to transfer data between FPGA and DRAM depending on the signal back from FPGA.

2.2 Hardware

2.2.1 Global Buffer

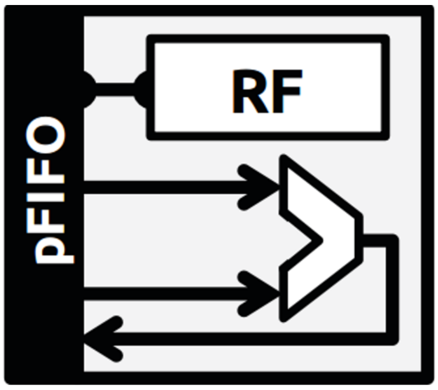
Since the I/O throughput of data is very large in CNN computation,we need to set up a global buffer to save data from/to the external DRAM to decrease the frequency of communication between DRAM and FPGA.Global buffer would be made of Block ram on the FPGA board.The typical size of global buffer should be 128KB for CNN computation.Since we have 4450Kbits embedded memory in the FPGA,it's enough for us to configure the global buffer.

2.2.2 I/O FIFO

For hardware part,we need FIFO to control data communication between global buffer,PE array and external DRAM.In this design,data width of the FIFO should be 16bits since data and weight we use would be 16bits fixed point number.However,for one FIFO,data transfer speed is limited by the clock cycle and data width.To solve this problem,we will try to use multiple FIFOs to make sure we have enough throughput for data loading into the PE array.

2.2.3 PE ARRAY

PE(Processing Element) is the fundamental unit that do convolution computation.Its structure is shown below.

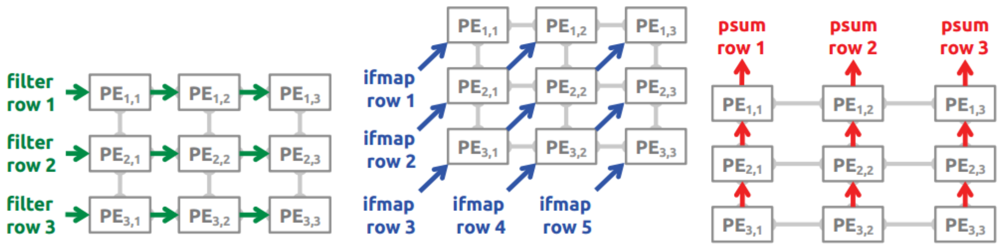


PE is made of register file,MAC(Multiplication and Accumulation unit)and FIFO.For every MAC operation,FIFO load data to input from register file.After calculation of a partial sum,the output of MAC will send back to RF through FIFO.By using this method,we could calculate multiple partial sum and send back to global buffer or DRAM together.The typical size of PE register file is 256Byte.Since we have 500KB flip flop resouces on the FPGA and the PE array size will be about 200,it will be enough flip flop resources for us to use.

PE array is multiple PEs that connected in array that could transfer data among them.The most important part of our hardware design is to set the feasible dataflow that go through PE array.

2.2.4 Row stationary dataflow

In our project,we will implement row stationary dataflow method.The dataflow of PE array is shown below.



The implementation of the RS dataflow is inspired by the idea of applying a strip mining technique in a spatial architecture. It breaks the high-dimensional convolution down into 1D convolution primitives that can run in parallel; each primitive operates on one row of filter weights and one row of ifmap pixels, and generates one row of psums. Psums from different primitives are further accumulated together to generate the ofmap pixels. The inputs to the 1D convolution come from the global buffer or DRAM. Each primitive is mapped to one PE for processing.Therefore, the computation of *each row pair stays stationary* in the PE, which creates convolutional reuse of filter weights and ifmap pixels at the RF level.

3 Milestone

Milestone 1 (Apr 5) Construct the basic building blocks of the whole neural network, including convolutional layers, fully-connected layers, pooling layers and activation function. These blocks are implemented by System Verilog.
Milestone 2 (Apr 19) Implementing the dataflow strategy. Maximize the data reuse to achieve better energy efficiency.
Milestone 3 (May 3) Test the overall throughput and energy efficiency and write up necessary documents.

Reference

[1] Kamel Abdelouahab, Maxime Pelcat, Jocelyn Serot, and Fran¸cois Berry. Accelerating cnn inference on fpgas: A survey. arXiv preprint arXiv:1806.01683, 2018.
[2] Yu-Hsin Chen, Joel Emer, and Vivienne Sze. Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks. In ACM SIGARCH Computer Architecture News, volume 44, pages 367–379. IEEE Press, 2016.
[3] Liqiang Lu, Yun Liang, Qingcheng Xiao, and Shengen Yan. Evaluating fast algorithms for convolutional neural networks on fpgas. In 2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), pages 101–108. IEEE, 2017. [4] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. Efficient processing of deep neural networks: A tutorial and survey. Proceedings of the IEEE, 105(12):2295–2329, 2017