基于高层次融合的卷积神经网络 FPGA 硬件加速实现

魏楚亮 [1*]，陈儒林 [1]，高谦 [2,3]，孙正隆 [2,3]

（1. 汕头大学 电子工程系；

2. 深圳市人工智能与机器人研究院；

3. 香港中文大学（深圳） 理工学院

*通讯作者： clwei@stu.edu.cn）

摘要：深度学习是从大规模原始数据中提取所需信息的一种先进方法。卷积神经网络作为深度学习的代表算法之一，被广泛应用于模式识别、计算机视觉和自然语言处理等领域。可编程门阵列（FPGA）为一种编程自由度高且可开发性强的高速处理集成电路，可以加速卷积神经网络前向传播的计算过程。为了降低 FPGA 的开发成本，开发者可采用 Xilinx 公司的 Vivado HLS 设计复杂的 FPGA 任务。基于 Vivado HLS，开发人员可以通过 C/C++代码而不是硬件描述语言在 FPGA 开发平台上设计硬件体系结构。本文研究了如何在 FPGA 平台上，通过高层次融合进行深度学习网络前向传导过程的硬件加速。经测试表明该系统的性能明显优于传统的 GPU 计算平台。

关键词：可编程逻辑门电路；高层次融合；深度学习；硬件加速电路

FPGA based Hardware Acceleration for CNN Developed by High Level Synthesis

Chuliang Wei[1※]; Rulin Chen[1]; Qian Gao[2,3]; Zhenglong Sun[2,3]

[1]Department of Electronic Engineering, Shantou University

[2]Shenzhen Institute of Artificial Intelligence and Robotics for Society

[3]School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen

※Corresponding Author： clwei@stu.edu.cn

Abstract: Deep learning is a advanced methodology for extracting the desired information from large scale original data. As a representative algorithm of deep learning, convolutional neural network (CNN) has been widely used in pattern recognition, computer vision and natural language processing etc. A flexible high-speed processing integrated circuit, known as Field Programmable Gate Array (FPGA), can accelerate the forward propagation process significantly. To reduce the FPGA development cost, Xilinx Vivado HLS is applied to design the complicated FPGA task. Through Vivado HLS, developers can design hardware architecture on FPGA platform through C/C++ code instead of hardware description language. In this paper, we describe how to implement deep learning network on FPGA platform through High Level Synthesis tool. The evaluation of the proposed system shows better performance than the traditional computing platform GPU.

Keywords: FPGA, High Level Synthesis, Deep Learning, Hardware Acceleration Circuits

1. Introduction

In recent years, Convolutional Neural Networks (CNN) have become an important approach in some informatics or engineering fields such as computer vision [1] [2] [3] , signal processing [4] [5] and robotics [6] [7], which requires complex artificial intelligence. In addition, other complicated interdisciplinary applications [8] [9] including stock price prediction, gas exploration, and medical imaging etc are also in need of CNN.

Graphics Processing Unit (GPU) has been widely used as the accelerator for CNN. Sasanka Potluri et al [10] proposed a real-time Discrete Time CNN system on the basis of GPU developed by Open Computing Language (Open CL), which showed better computing performance compared to the CPU system. Besides, Daniel Strigl et al [11] presented a acceleration framework of CNN based on GPU for some complex problems such as Optical Character Recognition (OCR) or face detection. Other works including car plate recognition system [12] and denoiser prior system for image restoration [13] have been proposed on the basis of GPU. It has been proved that GPU performs 2 to 24 times faster than on the central processing unit (CPU).

A more powerful hardware acceleration circuit, Field-Programmable Gate Array (FPGA), has been proved that it has smaller clock cycle requirement compared to GPU in the same tasks [14] due to the richer embedded utilization resources such as DSP blocks, registers and FIFOs etc [15]. Chen Zhang et al [16] presented a FPGA-based accelerator for CNN, which achieves a peak performance of 61.62 GFLOPS under 100MHz working frequency and outperform other implementations prominently.

However, GPU is widely used as the deep learning computing platform because of its efficient development process while few developers choose Field-Programmable Gate Array (FPGA). According to [14], it took 1 person (postdoctoral level) 2 months to develop a GPU-based real-time phase-based optical processing system while it took 2 persons (postdoctoral level) 15 months to finish the same system on FPGA.

With the development of high level synthesis, Xilinx presented a novel tool, Vivado HLS [17], to design large scale complex FPGA tasks using high level computer languages [18]. Traditionally, developers need to use inefficient low-level hardware description language (HDL) to design at high cost. Through Vivado HLS, developers do not have to program the HDL instead of using C/C++ to design the FPGA architectures. Then the designed C/C++ code can be converted to the register-transfer level (RTL) model and HDL by Vivado HLS automatically. Furthermore, Vivado HLS provides the different directives to optimize the FPGA design to reduce the system latency and interval. The evaluation of the design can also be seen in the HLS.

In this paper, we develop a FPGA based hardware acceleration system for CNN which can further be used in real time processing system. The rest of the paper is organized as follows. Section 2 introduces the architecture of AlexNet. Section 3 illustrates how to develop AlexNet on FPGA by HLS tool and optimize the original model through optimization directives in detail. The computing performance comparison between the proposed FPGA system and GPU platform is detailed in Section 4. In the Section 5, a application scenario of proposed system, UR5 based human-robot interaction system, is illustrated in detail. Finally, Section 6 makes a brief conclusion and looks forward to the implementation for some challenging projects in the future.

2. Architecture of CNN

Here we choose AlexNet as the deep learning model to test. AlexNet is widely used in computer vision tasks [19] [20] [21] because its reasonable tradeoffs between speed and accuracy. The whole network comprises 8 layers with training weight: the first five are convolution layer and the last three are fully-connected. The ReLU non-linearity is implemented to follow every convolutional and fully-connected layer. Besides, there are two normalization layers and three max pooling layers in the AlexNet. The author used softmax function at the end of the network to provide the distribution of different class labels. If we use ImageNet with every image pixel size 227×227×3 as dataset to train the network, the output will be the 1000-way one dimensional vector because this dataset contains 1000 different classes. The overall architecture of AlexNet and detailed information of each layers are shown in Table 1.

Layer Name	**Layer Type**	**Details**
Conv 1	96 11×11×3 Conv olution	Stride [4 4] Padding[0 0 0 0]
Conv 2	256 5×5×48 Convolution	Stride [1 1] Padding[2 2 2 2]
Conv 3	384 3×3×256 Convolution	Stride [1 1] Padding[1 1 1 1]
Conv 4	384 3×3×192 Convolution	Stride [1 1] Padding[1 1 1 1]
Conv 5	256 3×3×192 Convolution	Stride [1 1] Padding[1 1 1 1]
FC 1	4096 fully connec ted layer	
FC 2	4096 fully connec ted layer	
FC 3	1000 fully connec ted layer	

Table.1 The architecture of AlexNet.

3. HLS Based Development Process

Traditionally, FPGA can be developed at either gate level (GL) or register transfer level (RTL). Designing FPGA in traditional way needs developers to arrange the logic gate circuit to satisfy the desired need while taking lots of details into consideration such as bit width and time sequence etc, which requires a lot of developing time even for a experienced developer. In [14], to compare the performance between GPU and FPGA, it took 15 months to achieve the desired task on FPGA for two postdoctoral people and it only took 2 months on GPU for one same level developer.

In order to reduce the development cost on FPGA and meet the requirement of more complicated computing task, hardware need to be design at the algorithmic level, which means developers can only focus on the high level specification of the problem. For this reason, Xilinx produced a new FPGA development kit Vivado for synthesis and analysis of HDL architectures. One of the most important tool of Vivado is High Level Synthesis (HLS), which accepts synthesizable subsets of ANSI, C/C++, SystemC and Matlab. Then the code is analyzed and automatically converted into register-transfer level (RTL) model and HDL, which is traditionally generated by gate-level logic synthesis development software.

The working flow for FPGA development of AlexNet using Vivado HLS is shown in Figure 1. In this system, we use C/C++ as our development language and set all the computations to perform using single floating point data type. First we design the AlexNet using high level language (C/C++) and perform the C/C++ simulation experiments. Once the experiment result meets our requirements, the C/C++ code is converted to HDL and automatically generate the RTL model through High Level Synthesis. Furthermore, Vivado HLS also provides C/RTL co-simulation to simulate different FPGA on-chip environments and evaluate the use of logic gate resources in the proposed system. In this design, all the computations are performed using single floating point data type.

To reduce the latency and interval, HLS has different directives for optimizing the FPGA design. Optimization directive in HLS is another powerful tool to help developers to design FPGA at the algorithmic level. It can produce a micro-architecture that meets the desired requirement and area goals. We apply PIPELINE and ARRAY_PARTITION directives here. Through PIPELINE directive, the next execution do not have to start until the current execution finished which greatly reduce the initiation interval. ARRAY_PARTITION directive can partition large arrays into multiple smaller arrays or into individual registers, improving access to data and remove block RAM bottlenecks which helps in reducing latency. The example use of optimization directives in Vivado HLS is shown in Figure 2.

After optimization, the proposed system can be encapsulated into an IP core. We directly uses the FPGA development platform to call the IP core to complete the process of developing FPGA through HLS from the C/C++ program to the FPGA on-chip system.

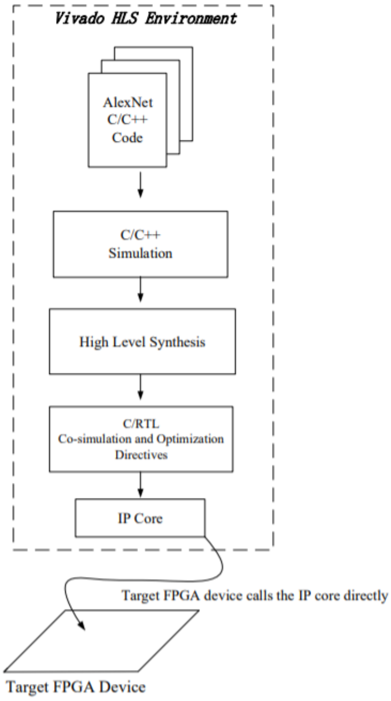
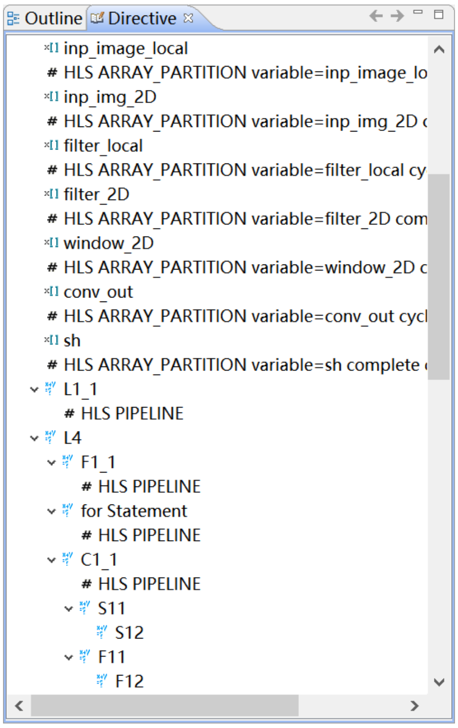


Fig. 1 The development working flow of AlexNet on FPGA in Vivado HLS environment.



```
inp_image_local
  # HLS ARRAY_PARTITION variable=inp_image_lo
inp_img_2D
  # HLS ARRAY_PARTITION variable=inp_img_2D c
filter_local
  # HLS ARRAY_PARTITION variable=filter_local cy
filter_2D
  # HLS ARRAY_PARTITION variable=filter_2D com
window_2D
  # HLS ARRAY_PARTITION variable=window_2D c
conv_out
  # HLS ARRAY_PARTITION variable=conv_out cycl
sh
  # HLS ARRAY_PARTITION variable=sh complete
L1_1
    # HLS PIPELINE
L4
  F1_1
      # HLS PIPELINE
  for Statement
      # HLS PIPELINE
  C1_1
      # HLS PIPELINE
    S11
      S12
    F11
      F12
```

Fig. 2 The use of optimization directives in one convolution layer.

4. Testing performance and analysis

The proposed system implements pre-trained AlexNet model with 60.5k parameters on Xilinx xcvu9p-flgb2104-2-i FPGA device and the development environment is Vivado 2017.4. The operating frequency is set to 100 MHz. By comparison, we implement the same model with the same parameter bit width in the NVIDIA 960m GPU with 12GB memory working environment developed by Matlab 2018b.

The performance comparison between FPGA and GPU is shown in Figure 3. It takes 21.95ms to complete the forward propagation procedure for a 227×227×3 image on FPGA while taking 70ms on GPU. In general, the computing on FPGA platform is about 3 times faster than the traditional GPU platform.

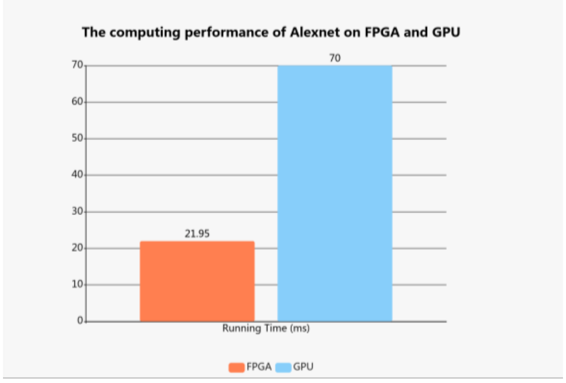


Fig. 3 The performance comparison between the FPGA and GPU platform.

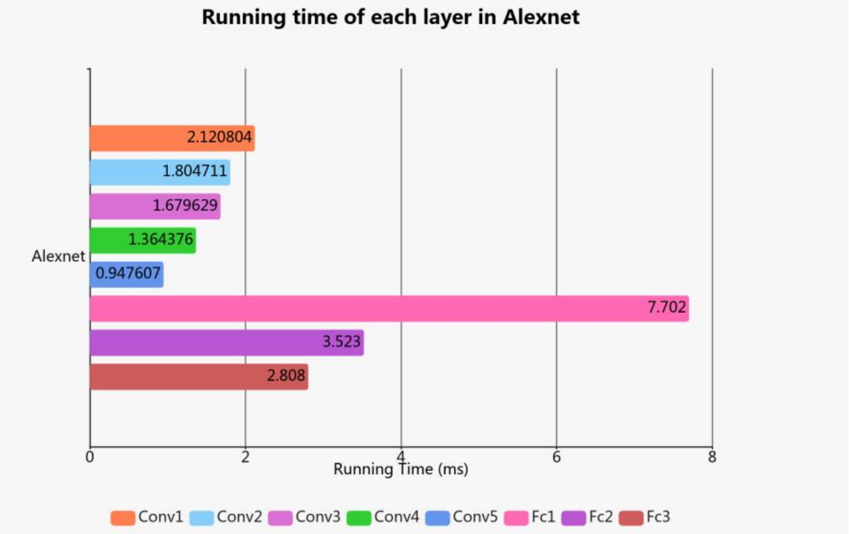


Fig. 4 The running time of each layer in AlexNet.

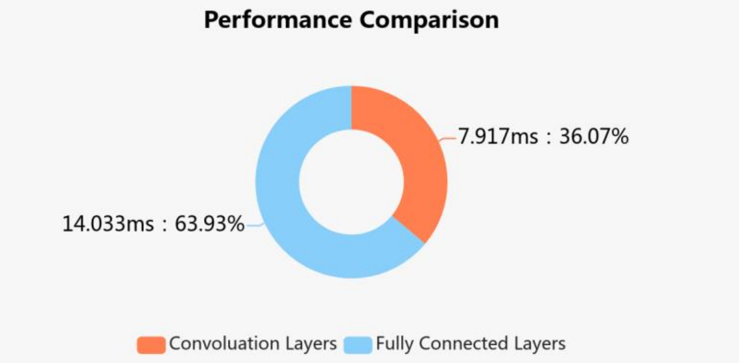


Fig. 5 Performance comparison between convolution layers and fully connected layers.

Resource	Utilization	Available	Utilization(%)
BRAM	1124	4320	26.01%
DSP	6686	6840	97.74%
FF	1404357	2364480	59.39%
LUT	1075078	1182240	90.93%

Table.2 .Resource utilization of Xilinx xcvu9p-flgb2104-2-i.

Moreover, detailed running time of each layer is shown in Figure 4. It is clear that the execution time is decreasing from the first to the last convolution layers, because the number of parameters is reducing after every convolution layer. Though there are only three fully connected layers, it takes 63.93% execution time of whole system to perform them as shown above in Figure 5. Table 2 indicates the resource utilization situation of the proposed system, which is within the limit of the chosen FPGA board.

5. Conclusion

This paper proposed a FPGA based hardware acceleration system for deep learning network. The development tool is the novel Vivado High Level Synthesis instead of traditional hardware description language which can focus on the algorithmic level to reduce the development cost. AlexNet is chosen in the proposed system and the evaluation of the proposed system shows better performance than the traditional computing platform like GPU. Moreover, the proposed system can be further used in some practical projects, such as optical signal processing, image processing, self-driven car and human-robot interaction system, to accelerate the processing procedure while dealing with the large scale complex input data.

In the future, we plan to put the proposed FPGA based AlexNet network system into practice. So far we have built up a whole human-robot interaction system consisting of UR5 robot arm, kinect camera, force sensor and infrared sensor. To achieve a more stable and sensitive system, the speed of image processing should be as fast as possible. Due to the limited resources and fixable circuit design, GPU can not be a prefect processor in this specific task compared to FPGA. Besides, our designed system is divided into separate layers, which means the system could be changed into other similar convolution neural networks and used in other different application scenarios flexibly.

6. Acknowledgements

This work was supported by the Characteristic Innovation Project of Universities in Guangdong Province-Natural Science under Grant 2018KTSCX061, the National Natural Science Foundation of China under Grant 61471228, the Key Project of Guangdong Province Science & Technology Plan under Grant 2015B020233018, and the Project 2019-INT010 from the Shenzhen Institute of Artificial Intelligence and Robotics for Society.

References

[1]Oza, Poojan, and Vishal M. Patel. "Deep CNN-based Multi-task Learning for Open-Set Recognition." arXiv preprint arXiv:1903.03161 (2019).

[2]Y. Zhu, R. Urtasun, R. Salakhutdinov, and S. Fidler, “SegDeepM: Exploiting segmentation and context in deep neural networks for object detection,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, pp. 4703 – 4711, USA, June 2015.

[3]R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR’14), pp. 580 – 587, Columbus, Ohio, USA, June 2014.

[4]Zhang, Dacheng, et al. "A novel in-loop filtering mechanism of HEVC based on 3D sub-bands and CNN processing." Signal, Image and Video Processing (2019): 1-9.

[5]Ye, Hao, Geoffrey Ye Li, and Biing-Hwang Juang. "Power of deep learning for channel estimation and signal detection in OFDM systems." IEEE Wireless Communications Letters 7.1 (2018): 114-117.

[6]Liang, Feng, and Chun Zhang. "Hardware Oriented Vision System of Logistics Robotics." 2018 12th IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID). IEEE, 2019.

[7]Gao, Xiang, and Tao Zhang. "Unsupervised learning to detect loops using deep neural networks for visual SLAM system." Autonomous robots 41.1 (2017): 1-18.

[8]Selvin, Sreelekshmy, et al. "Stock price prediction using LSTM, RNN and CNN-sliding window model." 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE, 2017.

[9]Li, Qing, et al. "Medical image classification with convolutional neural network." 2014 13th International Conference on Control Automation Robotics & Vision (ICARCV). IEEE, 2014.

[10]Potluri, Sasanka, et al. "CNN based high performance computing for real time image processing on GPU." Proceedings of the Joint INDS'11 & ISTET'11. IEEE, 2011.

[11]Strigl, Daniel, Klaus Kofler, and Stefan Podlipnig. "Performance and scalability of GPU-based convolutional neural networks." 2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing. IEEE, 2010.

[12]Lee, Sanghyeop, et al. "Car plate recognition based on CNN using embedded system with GPU." 2017 10th International Conference on Human System Interactions (HSI). IEEE, 2017.

[13]Zhang, Kai, et al. "Learning deep CNN denoiser prior for image restoration." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[14]Pauwels, Karl, et al. "A comparison of FPGA and GPU for real-time phase-based optical flow, stereo, and local image features." IEEE Transactions on Computers 61.7 (2012): 999-1012.

[15]Wang, Xiaofang, and Sotirios G. Ziavras. "Hera: A reconfigurable and mixed-mode parallel computing engine on platform fpgas." 16th International Conference on Parallel and Distributed Computing and Systems (PDCS). 2004.

[16]Zhang, Chen, et al. "Optimizing fpga-based accelerator design for deep convolutional neural networks." Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. ACM, 2015.

[17]Feist, Tom. "Vivado design suite." White Paper 5 (2012): 30.

[18]Wei, Chuliang, Rulin Chen, and Qin Xin. "FPGA Design of Real-Time MDFD System Using High Level Synthesis." IEEE Access 7 (2019): 83664-83672.

[19]Almisreb, Ali Abd, Nursuriati Jamil, and N. Md Din. "Utilizing AlexNet Deep Transfer Learning for Ear Recognition." 2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP). IEEE, 2018.

[20]Lu, Siyuan, Zhihai Lu, and Yu-Dong Zhang. "Pathological brain detection based on AlexNet and transfer learning." Journal of computational science 30 (2019): 41-47.

[21]Nawaz, Wajahat, et al. "Classification Of Breast Cancer Histology Images Using ALEXNET." International Conference Image Analysis and Recognition. Springer, Cham, 2018.