International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

COPY RIGHT

ELSEVIER SSRN

IJIEMR Transactions, online available on 3rd Aug 2019. Link :http://www.ijiemr.org/downloads.php?vol=Volume-08&issue=ISSUE-08

Title AN ADAPTABLE DEEP LEARNING ACCELERATOR UNIT (DLAU) FOR FPGA

Volume 08, Issue 08, Pages: 94–100.

Paper Authors

D.VEERA LAKSHMI, P.NAGARAJU
KAKINADA INSTITUTE OF ENGINEERING AND TECHNOLOGY,KORANGI,ANDHRAPRADESH,INDIA,533461

USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

To Secure Your Paper As Per UGC Guidelines We Are Providing A Electronic Bar Code

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

AN ADAPTABLE DEEP LEARNING ACCELERATOR UNIT (DLAU) FOR FPGA

[1]D.VEERA LAKSHMI, [2]P.NAGARAJU

[1]M.TECH VLSID, DEPT OF E.C.E, KAKINADA INSTITUTE OF ENGINEERING AND TECHNOLOGY,KORANGI,ANDHRAPRADESH,INDIA,533461

[2]ASSOSIATE PROFESSOR, KAKINADA INSTITUTE OF ENGINEERING AND TECHNOLOGY,KORANGI,ANDHRAPRADESH,INDIA,533461

Abstract:
Machine learning plays an important role in the field of artificial intelligence. Deep learning shows excellent ability in solving complex learning problems than that of machine learning. But, the size of the neural networks and the demand of practical applications poses significant challenge to construct a high performance implementations of deep learning neural networks. Hence to improve the performance and maintain the low power cost, in this paper a scalable accelerator deep learning accelerator unit (DLAU), is designed for large-scale deep learning networks using field-programmable gate array (FPGA) as the hardware prototype. The proposed DLAU uses carry save adder in the computation process and further to increase the computation speed, a parallel self timed adder is used in the design. Experimental results on the state-of-the-art Xilinx FPGA board demonstrate that the DLAU accelerator achieved more speed when compared to other devices like ASIC.

Keywords: Deep learning, field-programmable gate array (FPGA), hardware accelerator, neural network.

I.INTRODUCTION

In the past few years, machine learning has become pervasive in various research fields and commercial applications, and achieved satisfactory products. The emergence of deep learning speeded up the development of machine learning and artificial intelligence. Consequently, deep learning has become a research hot spot in research organizations [1]. In general, deep learning uses a multilayer neural network model to extract high-level features which are a combination of low-level abstractions to find the distributed data features, in order to solve complex problems in machine learning. Currently, the most widely used neural models of deep learning are deep neural networks (DNNs) [2] and convolution neural networks (CNNs) [3], which have been proved to have excellent capability in solving picture recognition, voice recognition, and other complex machine learning tasks. However, with the increasing accuracy requirements and complexity for the practical applications, the size of the neural networks becomes explosively large scale, such as the Baidu

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

Brain with 100 billion neuronal connections, and the Google cat-recognizingsystem with one billion neuronal connections. The explosive volume of data makes the data centers quite power consuming. In particular, the electricity consumption of data centers in U.S. are projected to increase to roughly 140 billion kilowatt-hours annually by 2020 [4]. Therefore, it poses significant challenges to implement high performance deep learning networks with low power cost, especially for large-scale deep learning neural network models. So far, the state-of-the-art means for accelerating deep learning algorithms are field-programmable gate array (FPGA), application specific integrated circuit (ASIC), and graphic processing unit (GPU). Compared with GPU acceleration, hardware accelerators like FPGA and ASIC can achieve at least moderate performance with lower power consumption. However, both FPGA and ASIC have relatively limited computing resources, memory, and I/O bandwidths, therefore it is challenging to develop complex and massive DNNs using hardware accelerators.

2. Literature SURVEY

For ASIC, it has a longer development cycle and the flexibility is not satisfying. Chen et al. [6] presented a ubiquitous machine-learning hardware accelerator called DianNao, which initiated the field of deep learning processor. It opens a new paradigm to machine learning hardware accelerators focusing on neural networks. But DianNao is not implemented using reconfigurable hardware like FPGA, therefore it cannot adapt to different application demands. Currently, around FPGA acceleration researches, Ly and Chow [5] designed FPGA-based solutions to accelerate the restricted Boltzmann machine (RBM). They created dedicated hardware processing cores which are optimized for the RBM algorithm. Similarly, Kim et al. [7] also developed an FPGA-based accelerator for the RBM. They use multiple RBM processing modules in parallel, with each module responsible for a relatively small number of nodes. Other similar works also present FPGA-based neural network accelerators [9]. Yu et al. [8] presented an FPGA-based accelerator, but it cannot accommodate changing network size and network topologies. To sum For ASIC, it has a longer development cycle and the flexibility is not satisfying. Chen et al. [6] presented a ubiquitous machine-learning hardware accelerator called DianNao, which initiated the field of deep learning processor. It opens a new paradigm to machine learning hardware accelerators focusing on neural networks. But DianNao is not implemented using reconfigurable hardware like FPGA, therefore it cannot adapt to different application demands. Currently, around FPGA acceleration researches, Ly and Chow [5] designed FPGA-based solutions to accelerate the restricted Boltzmann machine (RBM). They created dedicated hardware processing cores which are optimized for the RBM algorithm. Similarly, Kim et al. [7] also developed an FPGA-based accelerator for the RBM. They use multiple RBM processing modules in parallel, with each module responsible for a relatively small number of nodes. Other similar works also present FPGA-based

neural network accelerators [9]. Yu et al. [8] presented an FPGA-based accelerator, but it cannot accommodate changing network size and network topologies. To sum up, these studies focus on implementing a particular deep learning algorithm efficiently, but how to increase the size of the neural networks with scalable and flexible hardware architecture has not been properly solved.

Deep learning: Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised, semi-supervised or unsupervised. Deep learning is a class of machine learning algorithms that: • Use a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input. • Learn in supervised (e.g., classification) and/or unsupervised (e.g., pattern analysis) manners. • Learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts.

Objective

To tackle these problems, we present a scalable deep learning accelerator unit named DLAU to speed up the kernel computational parts of deep learning algorithms. In particular, we utilize the tile techniques, FIFO buffers, and pipelines to minimize memory transfer operations, and reuse the computing units to implement the largesize neural networks. This approach distinguishes itself from previous literatures with following contributions.

1) In order to explore the locality of the deep learning application, we employ tile techniques to partition the large scale input data. The DLAU architecture can be configured to operate different sizes of tile data to leverage the tradeoffs between speedup and hardware costs. Consequently, the FPGA-based accelerator is more scalable to accommodate different machine learning applications.

2) The DLAU accelerator is composed of three fully pipelined processing units, including tiled matrix multiplication unit (TMMU), part sum accumulation unit (PSAU), and activation function acceleration unit (AFAU). Different networktopologies such as CNN, DNN, or even emerging neural networks can be composed from these basic modules. Consequently, the scalability of FPGA-based accelerator is higher than ASIC-based accelerator.

3.PROPOSED DLAU ARCHITECTURE

The large-scale DNNs include iterative computations which have few conditional branch operations, therefore, they are suitable for parallel optimization in hardware. In this paper, we first explore the hot spot using the profiler. Results in Fig.1 illustrates the percentage of running time including matrix multiplication (MM), activation, and vector operations. For the representative three key operations: 1) feed forward; 2) RBM; and 3) BP, MM play a significant role of the overall execution. In particular, it takes 98.6%, 98.2%, and 99.1%

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

of the feed forward, RBM, and BP operations.

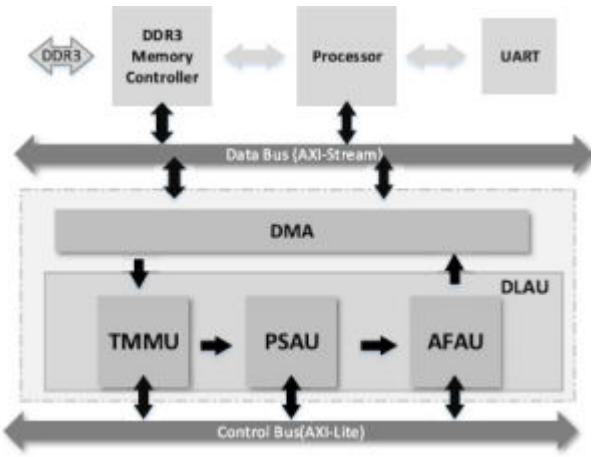

Fig. 1. DLAU accelerator architecture.

Fig.2 describes the DLAU system architecture which contains an embedded processor, a DDR3 memory controller, a DMA module, and the DLAU accelerator. The embedded processor is responsible for providing programming interface to the users and communicating with DLAU via JTAG-UART. In particular it transfers the input data and the weight matrix to internal BRAM blocks, activates the DLAU accelerator, and returns the results to the user after execution. The DLAU is integrated as a standalone unit which is flexible and adaptive to accommodate different applications with configurations. The DLAU consists of three processing units organized in a pipeline manner: 1) TMMU; 2) PSAU; and 3) AFAU. For execution, DLAU reads the tiled data from the memory by DMA, computes with all the three processing units in turn, and then writes the results back to the memory. In particular, the DLAU accelerator architecture has the following key features.

FIFO Buffer: Each processing unit in DLAU has an input buffer and an output buffer to receive or send the data in FIFO. These buffers are employed to prevent the data loss caused by the inconsistent throughput between each processing unit.

Pipeline Accelerator: We use stream-like data passing mechanism (e.g., AXI-Stream for demonstration) to transfer data between the adjacent processing units, therefore, TMMU, PSAU, and AFAU can compute in streaming-like manner. Of these three computational modules, TMMU is the primary computational unit, which reads the total weights and tiled nodes data through DMA, performs the calculations, and then transfers the intermediate part sum results to PSAU. PSAU collects part sums and performs accumulation. When the accumulation is completed, results will be passed to AFAU. AFAU performs the activation function using piecewise linear interpolation methods. In the rest of this section, we will detail the implementation of these three processing units, respectively.

A. TMMU Architecture TMMU is in charge of multiplication and accumulation operations. TMMU is specially designed to exploit the data locality of the weights and is responsible for calculating the part sums. TMMU employs an input FIFO buffer which receives the data transferred from DMA and an output FIFO buffer to send part sums to PSAU. Fig. 2 illustrates the TMMU schematic diagram, in which we set

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

tile size = 32 as an example. TMMU first reads the weight matrix data from input buffer into different BRAMs in 32 by the row number of the weight matrix (n = i%32 where n refers to the number of BRAM, and i is the row number of weight matrix). Then, TMMU begins to buffer the tiled node data.In the first time, TMMU reads the tiled 32 values to registers Reg_a and starts execution. In parallel to the computation at every cycle, TMMU reads the next node from input buffer and saves to the registers Reg_b. Consequently, the registers Reg_a and Reg_b can be used alternately. For the calculation, we use pipelined binary adder tree structure to optimize the performance. As depicted in Fig. 3, the weight data and the node data are saved in BRAMs and registers. The pipeline takes advantage of time-sharing the coarse-grained accelerators. As a consequence, this implementation enables the TMMU unit to produce a part sum result every clock cycle.

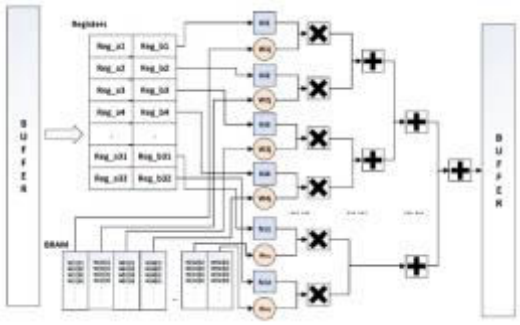

Fig. 2. TMMU schematic.

B. PSAU Architecture PSAU is responsible for the accumulation operation. Fig.3 presents the PSAU architecture, which accumulates the part sum produced by TMMU. If the part sum is the final result, PSAU will write the value to output buffer and send results to AFAU in a pipeline manner. PSAU can accumulate one part sum every clock cycle, therefore the throughput of PSAU accumulation matches the generation of the part sum in TMMU.

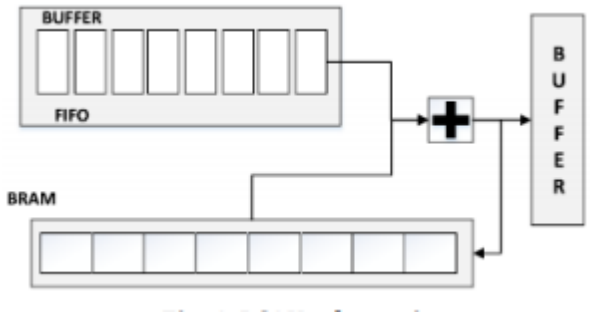

Fig. 3. PSAU schematic.

C. AFAU Architecture Finally, AFAU implements the activation function using piecewise linear interpolation ($y = ai*x+bi$, $x \in [x1, xi+1]$). This method has been widely applied to implement activation functions with negligible accuracy loss when the interval between xi and xi+1 is insignificant. Equation (1) shows the implementation of sigmoid function. For $x > 8$ and $x \leq -8$, the results are sufficiently close to the bounds of 1 and 0, respectively. For the cases in $-8 < x \leq 0$ and $0 < x \leq 8$, different functions are configured.

SIMULATION RESULTS

Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	151	14752	1%
Number of Slice Flip Flops	15	29504	0%
Number of 4 input LUTs	294	29504	0%
Number of bonded IOBs	40	250	16%
Number of GCLKs	1	24	4%

Figure 4:- Design summary

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

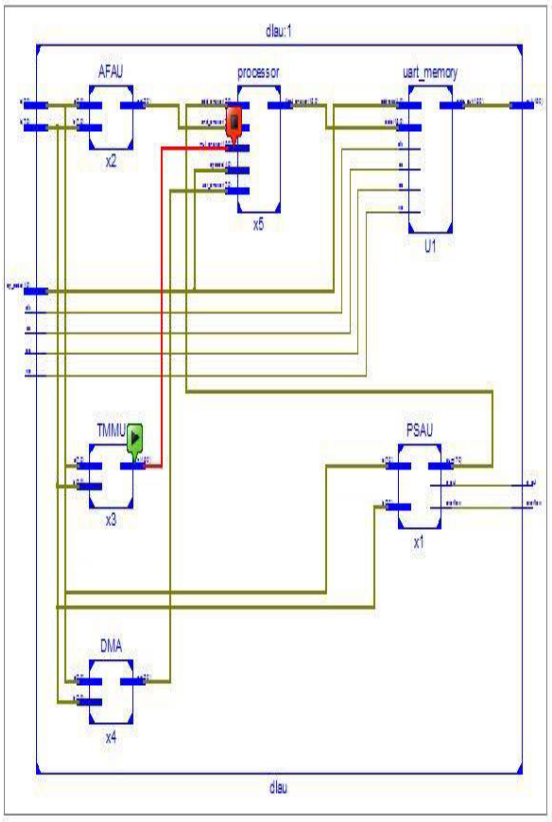

Figure 5:- RTL of DLAU

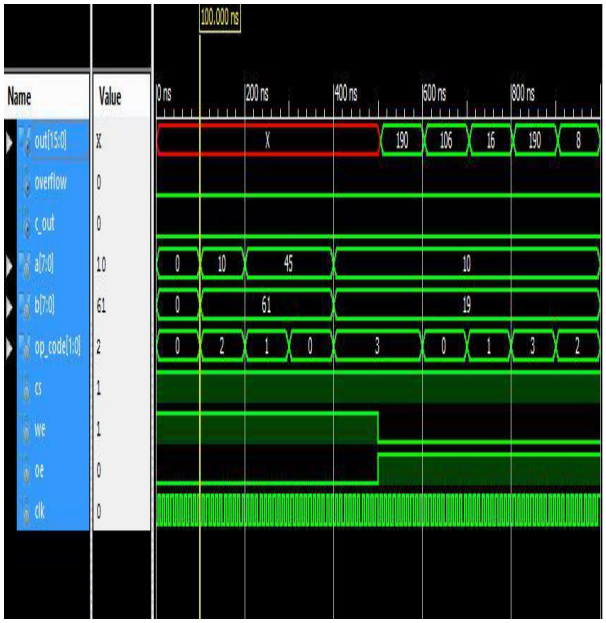

Figure 6:- Simulation of DLAU

```
================================================================================
Timing constraint: Default path analysis
  Total number of paths / destination ports: 32 / 2
--------------------------------------------------------------------------------
Delay:               14.713ns (Levels of Logic = 9)
  Source:            b<0> (PAD)
  Destination:       overflow (PAD)

  Data Path: b<0> to overflow
                                        Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             36   1.218   1.342  b_0_IBUF (b_0_IBUF)
     LUT4:I1->O             2   0.704   0.526  x1/x1/f1/c_out1 (x1/x1/ripple1)
     LUT3:I1->O             2   0.704   0.526  x1/x1/f2/c_out1 (x1/x1/ripple2)
     LUT3:I1->O             2   0.704   0.526  x1/x1/f3/c_out1 (x1/x1/ripple3)
     LUT3:I1->O             2   0.704   0.526  x1/x1/f4/c_out1 (x1/x1/ripple4)
     LUT3:I1->O             3   0.704   0.610  x1/x1/f5/c_out1 (x1/x1/ripple5)
     LUT3:I1->O             2   0.704   0.482  x1/x1/f6/c_out1 (x1/x1/ripple6)
     LUT3:I2->O             8   0.704   0.757  x1/x2/overflow1 (overflow_OBUF)
     OBUF:I->O                  3.272          overflow_OBUF (overflow)
    ----------------------------------------
    Total                      14.713ns (9.418ns logic, 5.295ns route)
                                        (64.0% logic, 36.0% route)

================================================================================
```

Figure 7:- Time Summary

CONCLUSION AND FUTURE WORK

In this paper, we have presented DLAU, which is a scalable and flexible deep learning accelerator based on FPGA. The DLAU includes three pipelined processing units, which can be reused for large scale neural networks. DLAU uses tile techniques to partition the input node data into smaller sets and compute repeatedly by time-sharing the arithmetic logic. Experimental results on Xilinx FPGA prototype show that DLAU can achieve 36.1× speedup with reasonable hardware cost and low power utilization. The results are promising but there are still some future directions, including optimization of the weight matrix and memory access. Also the trade-off analysis between FPGA and GPU accelerators is another promising direction for large scale neural networks accelerations.

REFERENCES

1. Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.

2. J. Hauswald et al., "DjiNN and Tonic: DNN as a service and its implications for future warehouse scale computers," in Proc. ISCA, Portland, OR, USA, 2015, pp. 27–40.
3. C. Zhang et al., "Optimizing FPGA-based accelerator design for deep convolutional neural networks," in Proc. FPGA, Monterey, CA, USA, 2015, pp. 161–170. 4. P. Thibodeau. Data Centers are the New Polluters. Accessed on Apr.
4, 2016. [Online]. Available: http://www.computerworld.com/ article/2598562/data-center/data-centers-are-the-new-polluters.html
5. L. Ly and P. Chow, "A high-performance FPGA architecture for restricted Boltzmann machines," in Proc. FPGA, Monterey, CA, USA, 2009, pp. 73–82.
6. T. Chen et al., "DianNao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," in Proc. ASPLOS, Salt Lake City, UT, USA, 2014, pp. 269–284.
7. S. K. Kim, L. C. McAfee, P. L. McMahon, and K. Olukotun, "A highly scalable restricted Boltzmann machine FPGA implementation," in Proc. FPL, Prague, Czech Republic, 2009, pp. 367–372.
8. Q. Yu, C. Wang, X. Ma, X. Li, and X. Zhou, "A deep learning prediction process accelerator based FPGA," in Proc. CCGRID, Shenzhen, China, 2015, pp. 1159–1162.
9. J. Qiu et al., "Going deeper with embedded FPGA platform for convolutional neural network," in Proc. FPGA, Monterey, CA, USA, 2016, pp. 26–35.