بسم الله الرحمن الرحيم

∞∞∞

تم رفع هذه الرسالة بواسطة / مني مغربي أحمد

بقسم التوثيق الإلكتروني بمركز الشبكات وتكنولوجيا المعلومات دون أدنى مسئولية عن محتوى هذه الرسالة.

ملاحظات: لا يوجد

IMPLEMENTATION OF DEEP CONVOLUTIONAL NEURAL NETWORKS (CNN) ON FPGA/CPU PLATFORM USING XILINX SDSOC

By

Rania Osama Hassan Hassan

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY
in
Electronics and Communications Engineering

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2022

Implementation of Deep Convolutional Neural Networks (CNN) on FPGA/CPU platform using Xilinx SDSoC

By

Rania Osama Hassan Hassan

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY
in
Electronics and Communications Engineering

Under the Supervision of

Associate Prof.
Omar Ahmed Ali Nasr

Associate Professor
Electronics and Communications
Department
Faculty of Engineering, Cairo University

Associate Prof.
Hassan Mostafa Hassan Mostafa

Associate Professor
Electronics and Communications
Department
Faculty of Engineering, Cairo University

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2022

Implementation of Deep Convolutional Neural Networks (CNN) on FPGA/CPU platform using Xilinx SDSoC

By
Rania Osama Hassan Hassan

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY
in
Electronics and Communications Engineering

Approved by the
Examining Committee

_____
Associate Prof. Omar Ahmed Ali Nasr, Thesis Main Advisor

_____
Associate Prof. Hassan Mostafa Hassan Mostafa, Advisor

_____
Prof. Dr. Mohsen Abd El Razik Rashwan, Internal Examiner

_____
Prof. Dr. Ahmed Hassan Madian , External Examiner
- NCRRT, Egyptian Atomic Energy Authority

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2022

Engineer's Name: Rania Osama Hassan Hassan
Date of Birth: 14/4/1988
Nationality: Egyptian
E-mail: Rania.osama@eng.cu.edu.eg
Phone: 01223326553
Address: El waha Nasr city

Registration Date: 1/10/2015
Awarding Date: …./…./2022.
Degree: Doctor of Philosophy
Department: Electronics and Communications Engineering

Supervisors:

Associate Prof. Omar Ahmed Ali Nasr
Associate Prof. Hassan Mostafa Hassan Mostafa

Examiners:

Associate Prof. Omar Ahmed Ali Nasr (Thesis main advisor)
Associate Prof. Hassan Mostafa Hassan Mostafa (advisor)
Prof. Mohsen Abd El Razik Rashwan (Internal examiner)
Prof. Ahmed Hassan Madian (External examiner)
NCRRT, Egyptian Atomic Energy Authority

Title of Thesis:

Implementation of Deep Convolutional Neural Networks (CNN) on FPGA/CPU platform using Xilinx SDSoC

Key Words:
Convolutional Neural Networks (CNNs), Alex-Net, GoogLeNet, SDSoC, HLS.

Summary:

CNNs are the state-of-the-art systems for image classification due to their high accuracy but their computational complexity is very high. Therefore one of the challenges in this field nowadays is the hardware acceleration for real time applications. FPGAs are the target for HW implementation as they have low power consumption and flexible architecture which fits larger CNNs despite the GPUs which consumes large power. This work discusses this problem and provides a solution that compromises between the speed of the CNN and the limited resources of FPGA. This solution depends on using parallelism and pipelining techniques inside some layers for implementing CNN using Xilinx SDSoC tool. The implementation of the design using high level language enhances the design time. In addition, it fits for larger designs compared to using only an FPGA. An Alex-Net CNN and GoogLeNet CNN are implemented successfully on Xilinx SDSoC platform and achieved a very good results.

Disclaimer

I hereby declare that this thesis is my own original work and that no part of it has been submitted for a degree qualification at any other university or institute.

I further declare that I have appropriately acknowledged all sources used and have cited them in the references section.

Name: Rania Osama Hassan Date: …. /..../2022

Signature:

Dedication

To my Father, my Mother, my daughter Maram, my husband, my brothers (Mohamed and Ali), and all my relatives and friends.

Acknowledgments

First of all I must thank ALLAH for his great mercy supporting me all the way till the end. If it weren't for his help, I wouldn't have reached this point.

I would like to express my deep sense of respect and gratitude towards my advisors and guides Dr. Hassan Mostafa and Dr. Omar Nasr for their continuous support, advice, and guidance throughout my work.

I would like to give many thanks to Eng. Mohamed Nafea for his help in solving a lot of issues together and his support in the work and also I would like to thank Eng. Mahmoud kishky for his support and help with the tool.

I am especially indebted to my family for their love, sacrifice, and support specially my father, my mother, and my husband for their continuous support.

I can never forget to thank my friends, Mona Ghazal,Manar Negm, Mona Fouad, Noura Ahmed, Doaa saeed and Doaa Fathy for their encouragement and advices.

TABLE OF CONTENTS

DISCLAIMER ..... **I**

DEDICATION ..... **II**

ACKNOWLEDGMENTS ..... **III**

LIST OF TABLES ..... **VIII**

LIST OF FIGURES ..... **IX**

NOMENCLATURE ..... **XI**

ABSTRACT ..... **XII**

CHAPTER 1: INTRODUCTION **1**

1.1. MOTIVATION 1
1.2. PROBLEM DISCUSSION 4
1.3. THE PROPOSED SOLUTION 6
1.4. ORGANIZATION OF THE THESIS 7

CHAPTER 2: BACKGROUND AND LITERATURE REVIEW **8**

2.1. OVERVIEW OF NEURAL NETWORKS (NNS) 8
2.2. OVERVIEW OF CONVOLUTIONAL NEURAL NETWORKS (CNNS) 9
2.2.1. THE CONVOLUTION LAYER 10
2.2.2. THE POOLING LAYER 11
2.2.3. THE RECTIFIED LINEAR UNIT (RELU) 11
2.2.4. LOCAL RESPONSE NORMALIZATION (LRN) 12
2.2.5. THE FULLY CONNECTED LAYER (FC) 12
2.3. THE CNN DIFFERENT MODELS 13
2.3.1. LENET-5 13
2.3.2. ALEX-NET 14
2.3.3. ZFNET & OVERFEAT 15
2.3.4. GOOGLENET 16
2.3.5. VGGNET 17
2.3.6. RESNET 18
2.4. RELATED WORK 23
2.4.1. SOFTWARE OPTIMIZATIONS 25
2.4.1.1. DATA QUANTIZATION 25
2.4.1.1.1. LINEAR QUANTIZATION 25
2.4.1.1.2. NON-LINEAR QUANTIZATION 26

2.4.1.2. PRUNING……..……...26
2.4.1.3. BINARIZED CNNS……..……...26
2.4.1.4. ENCODING TECHNIQUE……..……..26
2.4.1.5. DATA COMPRESSION……..……27
2.4.1.5. NETWORK DISTILLATION……..……27
2.4.2. HARDWARE FRIENDLY OPTIMIZATIONS……..……..27
2.4.2.1. LOOP REORDERING, UNROLLING, AND PIPELINING…...……...…27
2.4.2.2. MEMORY TILING…….……27
2.4.2.3. BATCHING……..……....28
2.4.2.4. SPATIAL, FREQUENCY-DOMAIN AND WINOGRAD CONV..……28
2.4.3. ARCHITECTURAL OPTIMIZATIONS TECHNIQUES……..29
2.4.3.1. THE LOOP TILING……29
2.4.3.2. PARALLELIZATION……..29
2.4.3.3. EXPLOITING DATA REUSE……29
2.4.3.4. PARTITIONING AND FOLDING……...30
2.5. PLATFORMS OF DL IMPLEMENTATION……30
2.5.1. CENTRAL PROCESSING UNIT (CPU)……31
2.5.2. GRAPHICS PROCESSING UNIT (GPU)……...31
2.5.3. FIELD PROGRAMMABLE GATE ARRAY (FPGA)……..32
2.5.4. APPLICATION SPECIFIC INTEGRATED CIRCUIT (ASIC)……..32
2.6. SUMMARY……..34
CHAPTER 3: PROPOSED HARDWARE ACCELERATION METHODOLOGY SDSOC TOOL 35
3.1. SDSOC OVERVIEW 35
3.2. HIGH LEVEL SYNTHESIS (HLS) FLOW 39
3.2.1. VIVADO HLS TOOL 43
3.2.2. VITIS AI 44
3.2.3. INTEL FPGA SDK FOR OPENCL 44
3.2.4. INTEL HLS COMPILER.....45
3.2.5. CATAPULT HLS PLATFORM 45
3.2.6. STRATUS HLS.....45
3.2.7. LEGUP COMPILER.....45
3.2.8. GAUT.....45

3.3. SDSoC Flow for implementation on ZYNQ UltraScale+104 46
3.4. Pragmas in SDSoC 48
3.5. Design Methodology using SDSoC 52
3.5.1. THE IMPLEMENTATION PROCEDURE USING SDSOC 52
3.5.1.1. APPLICATION PROFILING 52
3.5.1.2. MARKING HARDWARE FUNCTIONS 53
3.5.1.3. LOOP OPTIMIZATIONS 53
3.5.1.4. HLS PRAGMAS FOR OPTIMIZATION 53
CHAPTER 4: ALEX-NET HARDWARE ACCELERATION (CASE STUDY 1)…….....54
4.1. Alex-Net Architecture 54
4.1.1. REDUCING OVER FITTING 56
4.1.1.1. DATA AUGMENTATION 56
4.1.1.2. DROPOUT 56
4.1.2. ALEXNET LAYERS 56
4.1.2.1. CONVOLUTIONAL LAYER 57
4.1.2.2. THE MAX POOLING LAYER 58
4.1.2.3. THE RELU ACTIVATION FUNCTION 59
4.1.2.4. THE ZERO PADDING 60
4.1.2.5. LOCAL RESPONSE NORMALIZATION (LRN) 61
4.1.2.6. DROPOUT 61
4.1.2.7. THE FULLY CONNECTED LAYER 62
4.1.2.8. SOFTMAX LAYER 62
4.2. The proposed Optimization Techniques for Hardware ImplementatioN…….63
4.2.1. PIPELINING AND UNROLLING FOR LOOPS 64
4.2.2. OPTIMIZATION PRAGMAS FOR MEMORY ACCESS 66
4.3. Alex-Net implementation Results 67
4.3.1. ALEX-NET HARDWARE RESOURCES 67
4.3.2. ALEX-NET POWER ESTIMATIONS 68
4.3.3. HARDWARE ACCELERATED CYCLES 69
4.3.4. ALEX-NET IMPLEMENTATION COMBINATIONS 70
4.4. Comparison of our Convolution implementation in Alex-Net with the RTL only implementation 71
CHAPTER 5: GOOGLENET HARDWARE ACCELERATION (CASE STUDY 2)……73

5.1. GoogLeNet Architecture and The Inception Module 73
5.2. HLS Code for GoogLeNet 76
5.2.1. SELECTING REFERENCE MODEL 76
5.2.2. C++ IMPLEMENTATION 78
5.2.3. VERIFYING THE DEVELOPED C++ MODEL 78
5.3. The proposed Technique 78
5.3.1. TILING TECHNIQUE 79
5.4. GoogLeNet Implementation Results 82
5.4.1. THE TILING SIZE EFFECT ON THE PERFORMANCE 83
5.4.1.1. HARDWARE RESOURCES 83
5.4.1.2. HARDWARE ACCELERATED CLOCK CYCLES 84
5.4.1.3. THE REAL EXECUTION ON THE BOARD 84
5.4.1.4. POWER ESTIMATIONS 85
5.5. Comparison with Related Work 86
5.6. Comparison between Convolution implementation with and without the tiling technique for GoogLeNet 88
5.7. The kit working 94
CONCLUSIONS AND FUTURE WORK DIRECTIONS 95
REFERENCES 96
APPENDIX A : THE SDSOC OUTPUT PERFORMANCE ESTIMATIONS REPORTS (GOOGLENET)……..……...….105

List of Tables

Table 1.1 Size and complexity of state-of-the-art models....7
Table 2.1 Architecture of multiple versions of ResNet [26] 21
Table 2.2 Error rates (%) of single-model results on the ImageNet validation set (except† reported on the test set) [26]. 21
Table 2.3 Error rates (%) of ensembles. The top-5 error is on the test set of ImageNet and reported by the test server [26]. 22
Table 3.1: Zynq UltraScale+ MPSoC ZU7EV Features and Resources [57]....38
Table 3.2 State-of-the-art HLS Tools....42
Table 3.3 SDS Pragmas by Type [75] 48
Table 3.4 HLS Pragmas by Type [75]....49
Table 4.1 Architecture of AlexNet 57
Table 4.2 Alex-Net Layers Hardware Resources 68
Table 4.3 Power Estimations for Alex-Net Layers 69
Table 4.4 Alex-Net Hardware accelerated cycles 69
Table 4.5 Hardware Resources Utilization....70
Table 4.6 Implementation performance parameters....71
Table 4.7 Comparison between RTL and SDSoC implementation....71
Table 4.8 Comparison with the Recent Work 72
Table 5.1 Comparison between Different Reference Models 76
Table 5.2 GoogLeNet Overall Hardware Resources....83
Table 5.3 the Hardware Accelerated Clock Cycles....84
Table 5.4 the Real Execution time on the Board....84
Table 5.5 Hardware Resources Utilization (Tiling 32,16) 85
Table 5.6 Hardware Accelerated Clock Cycles (Tiling 32, 16) 85
Table 5.7 Power Estimations (Tiling 32, 16) 86
Table 5.8 Comparison with Hardware Accelerator....87
Table 5.9 Comparison with CPU and GPU Performance 87

List of Figures

Figure 1.1 Artificial Intelligence Overview 3
Figure 1.2 Design space in the development of accelerators for deep neural networks [9] 5
Figure 1.3 the classification accuracy vs. computation requirements (GOPs) for the inference step in recent well-known image classifiers [13]. 6
Figure 2.1 Human Neuron System [19] 9
Figure 2.2 Convolution Operation over one image [21] 10
Figure 2.3 the Convolution Operation on Small Local Window of Image [22].....10
Figure 2.4 The Max. Pooling Operation [22].....11
Figure 2.5 ReLU Function [22].....12
Figure 2.6 fully connected as a part of CNN [22].....13
Figure 2.7 LeNet-5 Architecture [19].....13
Figure 2.8 AlexNet Architecture [11] 14
Figure 2.9 ZFNet Architecture [19].....15
Figure 2.10 GoogLeNet architecture [19] 16
Figure 2.11 VGGNet Architecture [19].....17
Figure 2.12 the training error and test error for CIFAR-10 with different Depth [26]. .18
Figure 2.13 Residual Learning: a Building Block [26]. 19
Figure 2.14 VGG19 and ResNet (34-layers plain and Residual) Architectures [26].....20
Figure 2.15 Top-1 Accuracy of the state of the art models over last years [27]. 22
Figure 2.16 the architecture of LeNet5 [31].....24
Figure 2.17 Pseudo code of conv. layer without tiling (the left one) and after tiling (the right one) [21].....28
Figure 2.18 Use of batching to convert (a) matrix-vector multiplication into (b) matrix-matrix multiplication [21].....28
Figure 2.19 Data Reuse method between PE and BRAM [20] 30
Figure 2.20 SDF partitioning [20].....30
Figure 2.21 GFLOPS comparison between different CPUs and GPUs [35].....31
Figure 2.22 Hardware platforms comparison [1] 33
Figure 3.1: SDSoC Environment Flow [55].....35
Figure 3.2 SDSoC Architecture [55].....37
Figure 3.3 ZCU104 Evaluation Board Block Diagram [57] 38
Figure 3.4 Design flows for high-level code deployment [58]. 39
Figure 3.5 Generic HLS Design Flow [58]. 41
Figure 3.6 GAUT deign Flow [58].....46
Figure 3.7 Vivado HLS Design Flow [74] 47
Figure 3.8 The optimization methodology for the hardware functions [76]. 50
Figure 3.9 Example of defining the data for accelerator [75]. 51
Figure 4.1 ImageNet Large Scale Visual Recognithion Challenge winners [78]. 54
Figure 4.2 Alex-Net Neural Network Architecture [79] 55
Figure 4.3 Alexnet architecture distribution on two GPUs [11]. 56
Figure 4.4 the Convolution Operation [7]. 58
Figure 4.5 Max Pooling (Down sampling) Operation [80]. 59

Figure 4.6 Four-layer convolutional neural network with ReLUs (solid line) reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons (dashed line) [11]. 60
Figure 4.7 Dropout of multiple neurons in hidden layer [82]. 61
Figure 4.8 Multi-layer perceptron with one hidden layer [83]. 62
Figure 4.9 the softmax normalized probability [84]. 63
Figure 4.10 Loop pipeline effect on the throughput [75] 64
Figure 4.11 Sequential call for accelerator [75] 65
Figure 4.12 Pipeline execution of matrix multiplication [75] 65
Figure 4.13 Loop unrolled partially with factor 2 [75] 65
Figure 4.14 the equivalent unrolled loop [75]. 66
Figure 5.1 Inception Module used in GoogLeNet [12] 73
Figure 5.2 GoogLeNet incarnation of the Inception architecture [12] 74
Figure 5.3 GoogLeNet network [12]. 75
Figure 5.4 The Development Process Flow. 77
Figure 5.5 The pseudo Code for Tiling in a CLP [30] 79
Figure 5.6 The tiling Conversion to small Blocks 80
Figure 5.7 The pseudo Code for Tiling in convolution loops 81
Figure 5.8 The pseudo Code for the tiled sub_convkxk 82
Figure 5.9 The estimated performance report for inception3a_5x5_reduce without tiling 88
Figure 5.10 the estimated performance report for inception3a_5x5_reduce with tiling 89
Figure 5.11 The estimated performance report for inception3a_5x5_reduce with tiling (200MHz) 90
Figure 5.12 The estimated performance report for inception3a_5x5 without tiling 91
Figure 5.13 The estimated performance report for inception3a_5x5 with tiling 92
Figure 5.14 The estimated performance report for inception3a_5x5 with tiling (200 MHz) 93
Figure 5.15 The kit working on one sample test image 94
Figure A.1 Performance Output for Tiling 8 105
Figure A.2 Putty Output 105
Figure A.3 Performance Output for Tiling 16 106
Figure A.4 Putty output for tiling 16 106
Figure A.5 Performance Output for Tiling 32 107
Figure A.6 Performance Output for Tiling 32 for Kernels and 16 for Weights 108
Figure A.7 putty output from the board for Tiling 32 for Kernels and 16 for Weights 108
Figure A.8 Performance Output for Tiling 64 109
Figure A.9 putty output from the board 110
Figure A.10 the power estimator for all Conv. Blocks accelerator 111
Figure A.11 the power estimator for FC layer 112
Figure A.12 the power estimator for Softmax layer 113
Figure A.13 the power estimator for FC and all conv. Layers 114
Figure A.14 the power estimator for FC and Softmax Layers 114
Figure A.15 the power estimator for FC, Softmax and all conv. Layers 115