P-3472

FPGA IMPLEMENTATION FOR NEURAL NETWORK BASED IMAGE COMPRESSION

By

KALPANA.M

Reg. No. 1020106007

of

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University, Coimbatore)

COIMBATORE - 641006

A MINI PROJECT REPORT

Submitted to the

FACULTY OF ELECTRONICS AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements

for the award of the degree

of

MASTER OF ENGINEERING

IN

APPLIED ELECTRONICS

MAY 2011

BONAFIDE CERTIFICATE

Certified that this project report titled "**FPGA IMPLEMENTATION FOR NEURAL NETWORK BASED IMAGE COMPRESSION**" is the bonafide work of **KALPANA.M** [Reg. No. 1020106007] who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

[illegible]

Project Guide
Ms.M.Alagumeenaakshi M.E.,
A.P(SRG)

[illegible]

for Head of the Department
Dr. Rajeswari Mariappan

The candidate with university Register No. 1020106007 is examined by us in the project viva-voce examination held on ..23..05..2011.

[illegible] 23/5/11
Internal Examiner

[illegible] 23/5
External Examiner

ACKNOWLEDGEMENT

A project of this nature needs co-operation and support from many for successful completion. In this regards, I am fortunate to express my heartfelt thanks to Chairman **Padmabhusan Arutselvar Dr.N.Mahalingam B.Sc.,F.I.E.,** and Co-Chairman **Dr.B.K.Krishnaraj Vanavarayar B.Com,B.L.,** for providing necessary facilities throughout the course.

I would like to express my thanks and appreciation to the many people who have contributed to the successful completion of this project. First I thank, **Dr.J.Shanmugam, Ph.D,** Director, for providing me an opportunity to carry out this project work.

I would like to thank **Dr.S.Ramachandran, Ph.D,** Principal, who gave her continual support and opportunity for completing the project work successfully.

I would like to thank **Dr.Rajeswari Mariappan, Ph.D,** Prof of Head, Department of Electronics and Communication Engineering, who gave her continual support for us throughout the course of study.

I would like to thank **Ms.M.Alagumeenaakshi M.E (Ph.D),** Assistant Professor(SRG), Project guide for her technical guidance, constructive criticism and many valuable suggestions provided throughout the project work.

My heartfelt thanks to **Ms.R.Hemalatha M.E (Ph.D),** Associate Professor , Project coordinator, for her contribution and innovative ideas at various stages of the project to successfully complete this work.

I express my sincere gratitude to my family members, friends and to all my staff members of Electronics and Communication Engineering Department for their support throughout the course of my project.

ABSTRACT

Image compression is one of the key image processing techniques in signal processing and communication systems. Compression of images leads to reduction of storage space and reduces transmission bandwidth and hence also the cost. Advances in VLSI technology are rapidly changing the technological needs of common man. One of the major technological domains that are directly related to mankind is image compression. Neural networks can be used for image compression. Neural network architectures have proven to be more reliable, robust, and programmable and offer better performance when compared with classical techniques.

In this paper the main focus is to development of new architectures for FPGA implementation of neural network based image compression optimizing power and frequency. The design proposed consists of matrix multiplication of two matrices, one is the input image samples, and the second is the weight matrix obtained after training. This multiplied output is passed through the nonlinear transfer function to obtain the compressed output that gets transmitted or stored in compressed format. On the decompression side, the compressed data in matrix form is multiplied with the weight matrix to get back the original image. The image quality of the decompressed image depends on the weight matrix. The image data of size 16x16 is multiplied by the weight matrix of 4x16 to get a compressed output of 4x16. On the decompressor side 4x16 input matrix (compressed image) is multiplied with the weight matrix of size 16x4 reproduces the original image.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 PROJECT GOAL

Neural networks for image compression and decompression have been adopted as they achieve better compression and also work in noisy environment. Many approaches have been reported in realizing the NN architectures on software and hardware for real-time applications. Today's technological growth, has led to scaling of transistors and hence complex and massively parallel architecture are possible to realize on dedicated hardware consuming low power and less area. This work reviews the neural network approaches for image compression and proposes hardware implementation schemes for NN.

The transport of images across communication paths is an expensive process. Image compression provides an option for reducing the number of bits in transmission. This in turn helps to increase the volume of data transferred in a space of time, along with reducing the cost required. It has become increasingly important to most computer networks, as the volume of data traffic has begun to exceed their capacity for transmission.

A design method of neural networks based on VHDL hardware description language, and FPGA implementation is proposed. A design of a general neuron for topologies using back propagation algorithm is described. The neuron is then used in the design and implementation of a neural network using Xilinx Spartan-2e FPGA. The simulation results are obtained with Xilinx ISE 8.2i software. The results are analyzed in terms of operating frequency and chip utilization.

1.2 OVERVIEW

Artificial Neural Networks (ANNs) have been applied to many problems , and have demonstrated their superiority over traditional methods when dealing with noisy or incomplete data. One such application is for image compression. Neural Networks seem to be well suited to this particular function, as they have the ability to preprocess input patterns to produce simpler patterns with fewer components. This compressed information(stored in a hidden layer)preserves the full information obtained from the external environment. Not only can ANN based techniques provide sufficient compression rates of the data in question, but security is easily maintained. This occurs because the compressed data that is sent along a communication line is encoded and does not resemble its original form.

1.3 SOFTWARE USED

- ModelSim XE 111 6.2g
- Xilinx ISE 9.2i

1.4 ORGANIZATION OF THE REPORT

- **Chapter 2** discusses about data compression methods.
- **Chapter 3** discusses about artificial neural network
- **Chapter 4** discusses about back propagation algorithm .
- **Chapter 5** discusses the back propagation architecture.
- **Chapter 6** discusses the simulation results.
- **Chapter 7** shows the conclusion of the project.

CHAPTER 2

DATA COMPRESSION METHODS

Traditional techniques that have already been identified for data compression include: Predictive Coding, Transform coding and Vector Quantization.

1.2.1 Predictive Coding

Typically, images exhibit a high degree of correlation among neighboring samples. A high degree of correlation implies a high degree of redundancy in the raw data. Therefore, if the redundancy is removed by decorrelating the data, a more efficient and hence compressed coding of the signal is possible. This can be accomplished through the use of predictive coding or differential pulse-code modulation (DPCM).

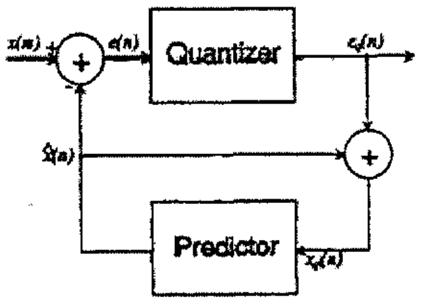


Fig. 1. Block diagram of a DPCM system.

The Predictor uses past samples x(n-l),x(n-2), . . . ,x(n-p), or in the case of images, neighboring pixels, to calculate an estimate, x^(n), of the current sample. It is the difference between the true value and the estimate, namely e(n) = x(n) – x^(n), which is used for storage or transmission. As the accuracy of the predictor increases, the variance of the difference decreases resulting in a higher predictive gain and therefore a higher compression ratio.

The problem, of course, is how to design the predictor. One approach is to use a statistical model of the data to derive a function which relates the value of the neighboring pixels to that of the current one in an optimal manner. An autoregressive model (AR) is one such model which has been successfully applied to images.

1.2.2 Transform Coding

Another approach to image compression is the use of transformations that operate on an image to produce a set of coefficients. A subset of these coefficients is chosen and quantized for transmission across a channel or for storage. The goal of this technique is to choose a transformation for which such a subset of coefficients is adequate to reconstruct an image with a minimum of discernible distortion. A simple, yet powerful, class of transform coding techniques is linear block transform coding. An image is subdivided into nonoverlapping blocks of n x n pixels which can be considered as N-dimensional vectors **x** with N = n x n. A linear transformation, which can be written as an M x N-dimensional matrix **W** with M ≤ N, is performed on each block with the M rows of W, wi being the basis vectors of the transformation. The resulting M-dimensional coefficient vector **y** is calculated as **y = wx.**

If the basis vectors **wi** are orthonormal, that is

$$\mathbf{w}_i^T \mathbf{w}_j = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

then the inverse transformation is given by the transpose of the forward transformation matrix resulting in the reconstructed vector $\mathbf{x} = \mathbf{W}^T\mathbf{y}$. The optimal linear transformation with respect to minimizing the mean squared error (MSE) is the Karhunen-Lotve transformation (KLT). The transformation matrix **W** consists of M rows of the eigenvectors corresponding to the M largest eigenvalues of the sample autocovariance matrix
Є= E[XXT] The KLT also produces uncorrelated coefficients and therefore results in the most efficient coding of the data since the redundancy due to the high degree of correlation between neighboring pixels is removed. The KLT is related to principal components analysis (PCA), since the basis vectors are also the M principal components of the data. Because the KLT is an orthonormal transformation, its inverse is simply its transpose. A number of practical difficulties exist when trying to implement the above approach. The calculation of the estimate of the covariance of an image may be unwieldy and may require a large amount of memory. Inaddition, the solution of the eigenvectors and eigenvalues is computationally intensive. Finally, the calculation of the forward and inverse transforms is of order O(MN) for each image block. Due to these difficulties, fixed-basis transforms suchas the discrete cosine transform (DCT) ,which can be computed in order O(N log N), are typically used when implementing block transforms schemes. The Joint Photographics Expert Group (JPEG) have adopted the linear block transform coding approach for its standard using the DCT as the transformation.

1.2.3 Vector Quantization

The process of quantization maps a signal x(n) into a series of K discrete messages. For the kth message, there exists a pair of thresholds t_k and $t_{k+1}$and an output value Q_k such that $t_k < Q_k \leq t_{k+1}$. For a given set of quantization values, the optimal thresholds are equidistant from the values. The concept of quantizing data can be extended from scalar or one-dimensional data to vector data of arbitrary dimension. Instead of output levels, vector quantization (VQ) employs a set of representation vectors (for the one-dimensional case) or matrices (for the twodimensional case). The set is referred to as the "codebook" and the entries as "codewords." The thresholds are replaced by decision surfaces defined by a distance metric. Typically, Euclidean distance from thecodeword is used. The advantage of vector quantization over scalar quantization is that the high degree of correlation between neighboring pixels can be exploited. Even for a memoryless system, the coding of vectors instead of scalars can theoretically improve performance.

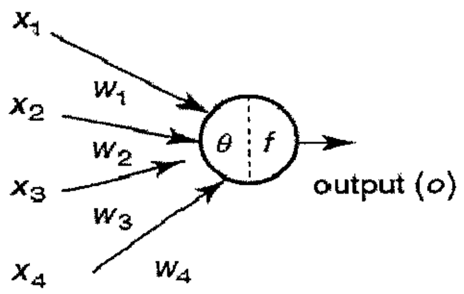
In the coding phase, the image is subdivided into blocks,typically of a fixed size of n x n pixels. For each block, the nearest codebook entry under the distance metric is found and the ordinal number of the entry is transmitted. On reconstruction, the same codebook is used and a simple lookup operation is performed to produce the reconstructed image. Initially, K codebook entries are set to random values. On each iteration, each block in the input space is classified, based on its nearest codeword. Each codeword is then replaced by the mean of its resulting class. The iterations continue until a minimum acceptable error is achieved. This algorithm minimizes the mean squared error over the training set. While the LBG algorithm converges to a local minimum, it is not guaranteed to reach the global minimum. In addition, the algorithm is very sensitive to the initial codebook. Furthermore, the algorithm is slow since it requires an exhaustive search through the entire codebook on each iteration.

With this brief review of conventional image compression techniques at hand, we are ready to consider the role of neural networks as an image compression tools.

CHAPTER 3
ARTIFICIAL NEURAL NETWORK

Artificial Neural Networks (ANNs) are non-linear mapping structures based on the function of the human brain. ANNs can identify and learn correlated patterns between input data sets and corresponding target values. After training, ANNs can be used to predict the outcome of new independent input data. ANNs imitate the learning process of the human brain and can process problems involving non-linear and complex data even if the data are imprecise and noisy.

ANN's consists of simple computational units called neurons, which are highly interconnected. ANNs are parallel computational models comprised of densely interconnected adaptive processing units. These networks are fine-grained parallel implementations of nonlinear static or dynamic systems. ANNs are now being increasingly recognized in the area of classification and prediction, where regression model and other related statistical techniques have traditionally been employed. The most widely used learning algorithm in an ANN is the Backpropagation algorithm.



(a) Artificial neuron

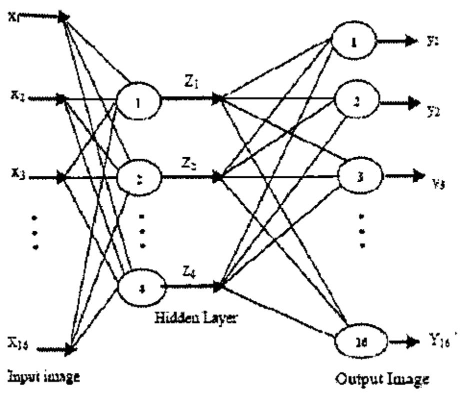
The basic architecture consists of three types of neuronlayers: input, hidden, and output layers. In feed-forward networks, the signal flow is from input to output units, strictly in a feed-forward direction. The data processing can extend over multiple (layers of) units, but no feedback connections are present. Recurrent networks contain feedback connections. Contrary to feed-forward networks, the dynamical properties of the network are important. In some cases, the activation values of the units undergo a relaxation process such that the network will evolve to a stable state in which these activations do not change anymore.

A neural network has to be configured such that the application of a set of inputs produces the desired set of outputs. Various methods to set the strengths of the connections exist. One way is to set the weights explicitly, using a priori knowledge. Another way is to train the neural network by feeding it teaching patterns and letting it change its weights according to some learning rule. The learning situations in neural networks may be classified into three distinct sorts. These are supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, an input vector is presented at the inputs together with a set of desired responses, one for each node, at the output layer. A forward pass is done, and the errors or discrepancies between the desired and actual response for each node in the output layer are found. These are then used to determine weight changes in the net according to the prevailing learning rule.

In unsupervised learning (or self-organization), a (output) unit is trained to respond to clusters of pattern within the input. In this paradigm, the system is supposed to discover statistically salient features of the input population. Unlike the supervised learning paradigm, there is no a priori set of categories into which the patterns are to be classified; rather, the system must develop its own representation of the input stimuli. Reinforcement learning is learning what to do – how to map situations to actions – so as to maximize a numerical reward signal.

CHAPTER 4
BACK PROPAGATION ALGORITHM

By taking the partial derivative of the error of the network with respect to each weight, the direction in which the error of the network moving can be determined. if we take the negative of this derivative (i.e. the rate change of the error as the value of the weight increases) and then proceed to add it to the weight, the error will decrease until it reaches a local minima. This makes sense because if the derivative is positive, this tells us that the error is increasing when the weight is increasing. The obvious thing to do then is to add a negative value to the weight and vice versa if the derivative is negative. Taking of these partial derivatives and then applying them to each of the weights takes place, starting from the output layer to hidden layer weights, then the hidden layer to input layer weights (as it turns out, this is necessary since changing these set of weights requires that we know the partial derivatives calculated in the layer downstream), this algorithm has been called the *backpropagation algorithm.*



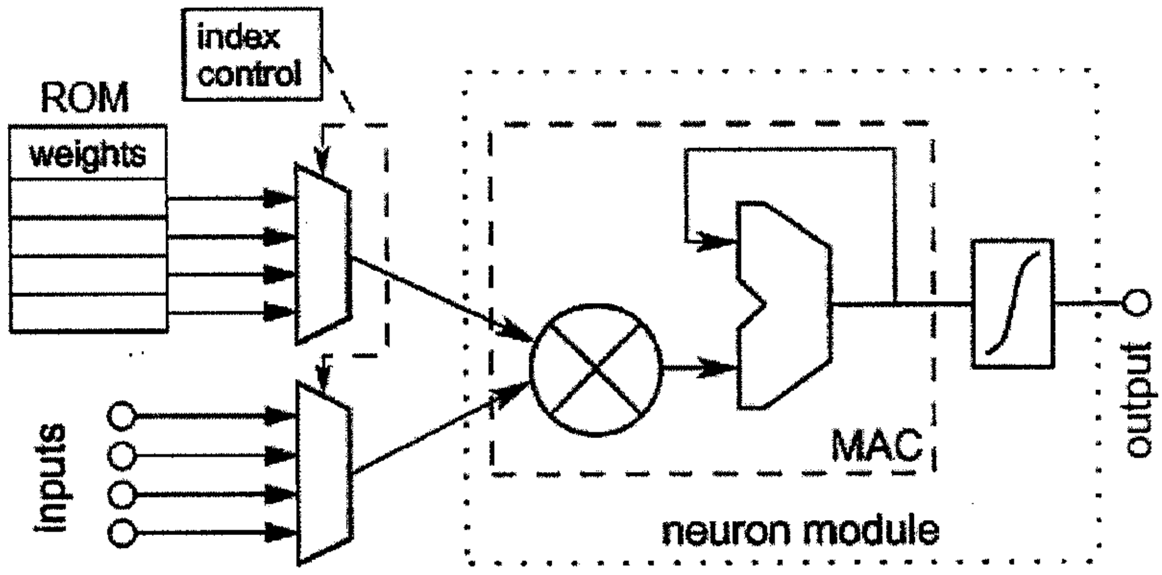
Basic architecture for image compression
Using NN.

A neural network can be trained in two different modes: online and batch modes. The number of weight updates of the two methods for the same number of data presentations is very different. The online method weight updates are computed for each input data sample, and the weights are modified after each sample. An alternative solution is to compute the weight update for each input sample, but store these values during one pass through the training set which is called an *epoch.* At the end of the epoch, all the contributions are added, and only then the weights will be updated with the composite value. This method adapts the weights with a cumulative weight update, so it will follow the gradient more closely. It is called the *batch-training mode.*

Training basically involves feeding training samples as input vectors through a neural network, calculating the error of the output layer, and then adjusting the weights of the network to minimize the error.

Initial weights are choosen at small random values.The initial weights cannot be very high because the sigmoidal activation function used here may get saturated from the beginning itself and the system may stuck at a local minima or at a very flat plateau at the starting point itself. The main concept involved is to update weight in neural network since it is a tedious process to calculate weight in back propagation algorithm as it involves various formulas we are designing the back propagation architecture to calculate new weights.

CHAPTER 5
BACK PROPAGATION ARCHITECTURE



Two distinct neuron implementation were designed using 8-bit and 12-bit binary MAC (multiply accumulate) circuits. The sigmoid activation function is used for implemented neurons in hidden layer, and linear activation function is used for output layer neuron. For all neurons (8-bit and 12-bit), the product of signed input (4-bit / 8-bit) and signed weight (4-bit) form a signed result (8-bit/ 12bit). These products value are accumulated into activation state. The final output value is obtained by applying the activation function. The weight coefficients are stored in a ROM within neurons.

In the figure, the MAC unit which accepts a serial processing of weights and parallel inputs pairs, each pair is multiplied together and a running total is recorded. An index control module controls the multiplexing order. Once all input pairs have been processed, the final sum is passed through the activation function to produce the neuron's output. The main advantage of serial processing is the small constant area required, regardless of topology, to implement one MAC and some routing for one input and one weight contained in the weight ROM module. The obvious disadvantage is the processing speed. If the network involves alarge number of inputs, serial processing will suffer from slow processing.

Sigmoid Activation Function Hardware Design:

A very important part of neuron implementation is activation function hardware design. One of the most frequently used activation function in backpropagation neural networks applications is the hyperbolic tangent (tanh) sigmoid function

$$f(n)=e^{n} -e^{-n}/e^{n}+e^{-n}$$

This function is not suitable for direct digital implementation as it consists of an infinite exponential series. Many implementations use a lookup table for approximation. Many other kinds of activation functions have been proposed and the backpropagation algorithm is applicable to all of them. A differentiable activation function makes the function computed by a neural network differentiable (assuming that the integration function at each node is just the sum of the inputs), since the network itself computes only function compositions. The error function also becomes differentiable. The values entered in the lookup table is between e^{0} to e^{16} since we are using a 16 input neuron.

STEPS INVOLVED IN ARCHITECTURE:

P- 3472

1.Weight values are randomly choosen.
2.Multiply the input pixel values and weight values that is fed as input to the MAC unit.
3. The multiplied output is compared with lookup table values and the difference produced is the error value.
4. The error can be rectified by dividing the difference value with the input image pixel value and the result obtained is the new weight matrix value.
5.By repeating the above process the new weight value for the entire matrix can be obtained.

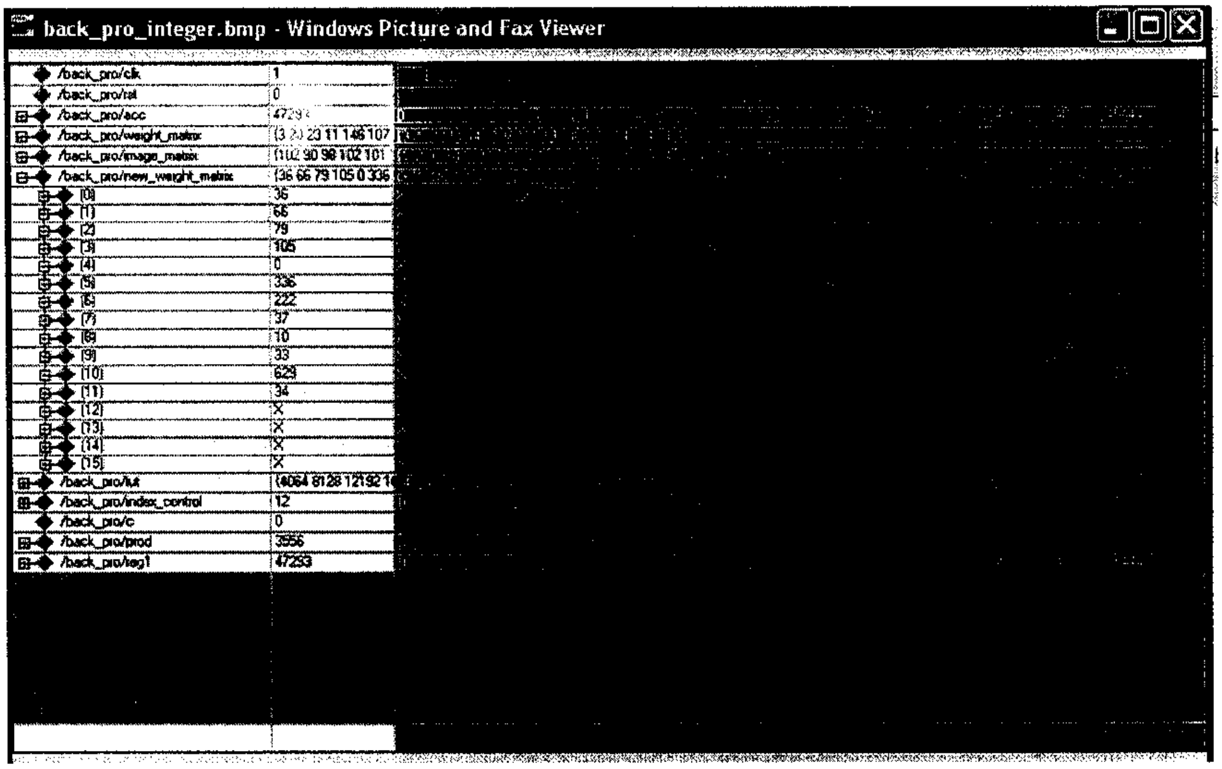
CHAPTER 6
RESULTS AND DISCUSSION

The simulation of this project has been done using MODELSIM XE111 6.2g and XILINX ISE 9.2i. Modelsim is a simulation tool for programming {VLSI} {ASIC}s, {FPGA}s, {CPLD}s, and {SoC}s. Modelsim provides a comprehensive simulation and debug environment for complex ASIC and FPGA designs. Support is provided for multiple languages including Verilog, SystemVerilog, VHDL and SystemC.

Xilinx was founded in 1984 by two semiconductor engineers, Ross Freeman and Bernard Vonderschmitt, who were both working for integrated circuit and solid-state device manufacturer Zilog Corp. The Virtex-II Pro, Virtex-4, Virtex-5, and Virtex-6 FPGA families are particularly focused on system-on-chip (SOC) designers because they include up to two embedded IBM PowerPC cores The ISE Design Suite is the central electronic design automation (EDA) product family sold by Xilinx. The ISE Design Suite features include design entry and synthesis supporting Verilog or VHDL, place-and-route (PAR), completed verification and debug using Chip Scope Pro tools, and creation of the bit files that are used to configure the chip.

The simulation results showing the implementation of the back propagation architecture and the synthesis results are attached here.

6.1 SIMULATION RESULT



back_pro_integer.bmp - Windows Picture and Fax Viewer

```
/back_pro/clk                 1
/back_pro/rst                 0
/back_pro/acc                 4729[illegible]
/back_pro/weight_matrix       {3 2[illegible] 23 11 146 107
/back_pro/image_matrix        {102 90 98 102 101
/back_pro/new_weight_matrix   {36 66 79 105 0 336
   [0]                        36
   [1]                        66
   [2]                        79
   [3]                        105
   [4]                        0
   [5]                        336
   [6]                        222
   [7]                        37
   [8]                        10
   [9]                        33
   [10]                       629
   [11]                       34
   [12]                       X
   [13]                       X
   [14]                       X
   [15]                       X
/back_pro/lut                 {4064 8128 12192 1
/back_pro/index_control       12
/back_pro/c                   0
/back_pro/prod                3556
/back_pro/reg1                47293
```

Simulation result for updation of weight

1st row and 1st column iteration:

1. Initially weight value 3 and input image value 102 is multiplied[3x102=306] and stored in register of MAC unit. Here the weight value is randomly chosen.
2. The multiplied value is compared with the lookup table value and difference produced is the error[4064-306=3758]
3. This error value is rectified by dividing the difference value with the input image value[3758/102=36].Thus the new weight value is computed.
4. The above procedure is performed for the entire matrix and multiplying the input image value with the new weight value gives the compressed image value.

6.2 SYNTHESIS RESULT FOR 64 BIT NEURON

64-BIT ARCHITECTURE:

DELAY REPORT:

Minimum period	:	4.022ns
Maximum Frequency	:	248.633MHz
Minimum input arrival time before clock	:	No path found
Maximum output required time after clock	:	6.347ns
Maximum combinational path delay	:	No path found

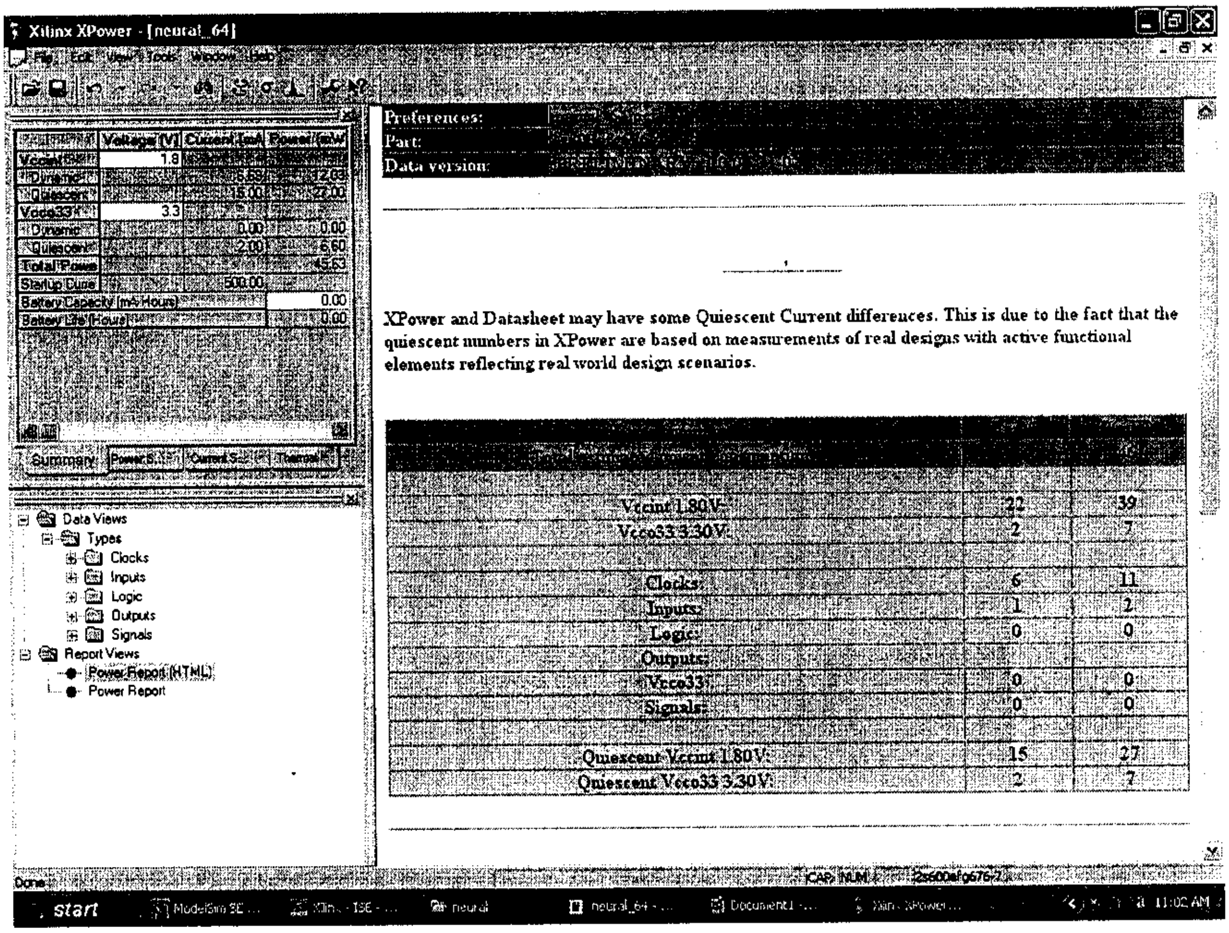
MAP REPORT:

Logic Utilization:

Number of Slice Flip Flops	:	21 out of 13,824	1%
Number of 4 input LUTs	:	2 out of 13,824	1%
Total Number 4 input LUTs	:	18 out of 13,824	1%
Number used as logic	:	2	
Number used as Shift registers	:	16	
Number of bonded IOBs	:	32 out of 510	6%
IOB Flip Flops	:	8	
Number of GCLKs	:	1 out of 4	25%
Number of GCLKIOBs	:	1 out of 4	25%
Total equivalent gate count for design	:	2,292	
Additional JTAG gate count for IOBs	:	1,584	

From the above delay report we obtain that maximum frequency is about 248.633MHz.

6.3 POWER REPORT 64 BIT NEURON



	Voltage [V]	Current [mA]	Power [mW]
Vccint	1.8		
Dynamic		6.69	12.03
Quiescent		15.00	27.00
Vcco33	3.3		
Dynamic		0.00	0.00
Quiescent		2.00	6.60
Total Power			45.63
Startup Current		500.00	
Battery Capacity [mA Hours]			0.00
Battery Life [Hours]			0.00

XPower and Datasheet may have some Quiescent Current differences. This is due to the fact that the quiescent numbers in XPower are based on measurements of real designs with active functional elements reflecting real world design scenarios.

Vccint 1.80V:	22	39
Vcco33 3.30V:	2	7
Clocks:	6	11
Inputs:	1	2
Logic:	0	0
Outputs:		
Vcco33:	0	0
Signals:	0	0
Quiescent Vccint 1.80V:	15	27
Quiescent Vcco33 3.30V:	2	7

From the above result we obtain the maximum power at 25degree centigrade to be 46(mW) which is larger compared to 16 bit neuron.

6.4 SYNTHESIS RESULT OF 16 BIT NEURON

16-BIT ARCHITECTURE:

DELAY REPORT:

Minimum period	:	4.535ns
Maximum Frequency	:	220.507MHz
Minimum input arrival time before clock	:	No path found
Maximum output required time after clock	:	6.140ns
Maximum combinational path delay	:	No path found

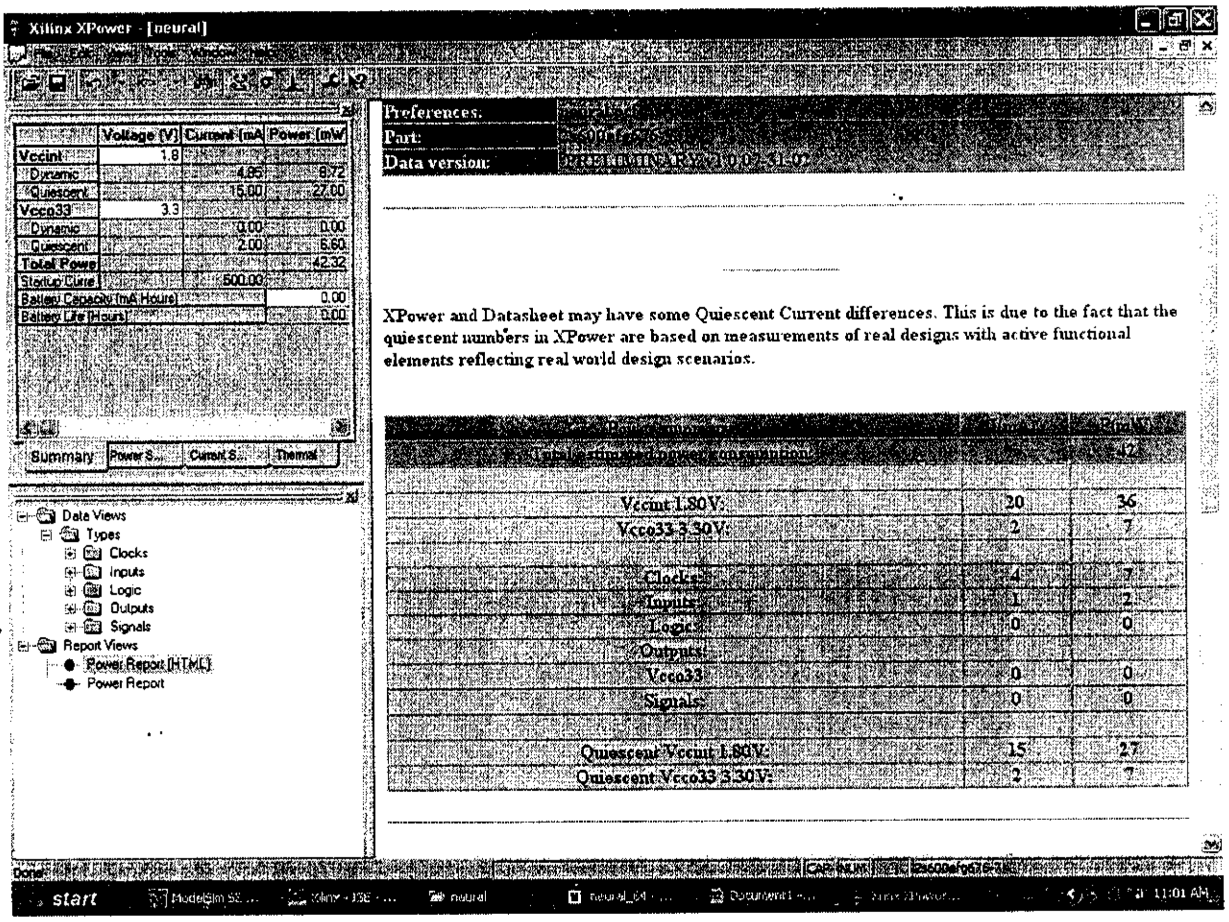
MAP REPORT:

Logic Utilization:

Number of Slice Flip Flops	:	7 out of 13,824 1%
Number of 4 input LUTs	:	4 out of 13,824 1%
Total Number 4 input LUTs	:	12 out of 13,824 1%
Number used as logic	:	4
Number used as Shift registers	:	8
Number of bonded IOBs	:	8 out of 510 1%
IOB Flip Flops	:	8
Number of GCLKs	:	1 out of 4 25%
Number of GCLKIOBs	:	1 out of 4 25%
Total equivalent gate count for design	:	1,168
Additional JTAG gate count for IOBs	:	432

From the above result the maximum frequency obtained is 220.507MHz

6.5 POWER REPORT FOR 16 INPUT NEURON


Xilinx XPower - [neural]

	Voltage (V)	Current (mA	Power (mW
Vccint	1.8		
Dynamic		4.85	8.72
Quiescent		15.00	27.00
Vcco33	3.3		
Dynamic		0.00	0.00
Quiescent		2.00	6.60
Total Powe			42.32
Startup Curre		500.00	
Battery Capacity (mA Hours)			0.00
Battery Life (Hours)			0.00

Preferences:	[illegible]
Part:	[illegible]
Data version:	[illegible]

XPower and Datasheet may have some Quiescent Current differences. This is due to the fact that the quiescent numbers in XPower are based on measurements of real designs with active functional elements reflecting real world design scenarios.

Power summary:	I(mA)	P(mW)
Total estimated power consumption:	[illegible]	42
Vccint 1.80V:	20	36
Vcco33 3.30V:	2	7
Clocks:	4	7
Inputs:	1	2
Logic:	0	0
Outputs:		
Vcco33	0	0
Signals:	0	0
Quiescent Vccint 1.80V:	15	27
Quiescent Vcco33 3.30V:	2	7

From the above result we obtain the maximum power at 25degree centigrade to be 42(mW) which is smaller compared to 64 bit neuron.

COMPARISON RESULTS

SYNTHESIS RESULTS		
Implementation Details	16	64
Maximum operating frequency	220.507MHz	248.633MHz
Maximum power at 25 degree Centigrade	42mW	46mW

Table 1 **COMPARISON OF 16 INPUTS AND 64 INPUT NEURON**

CHAPTER 7
CONCLUSION AND FUTURE SCOPE

This project presents a new architecture for image compression by updating the weights in network. The major advantage in using this method is that power gets reduced. As the design is mapped on FPGA, it supports Reconfigurability. Reconfigurability can be achieved by changing the weight matrix and the input layer for better compression.

Security is easily maintained. This occurs because the compressed data that is sent along a communication line is encoded and does not resemble its original form.

FUTURE SCOPE

Future scope of this project is to develop architecture and hardware design of above architecture using ASIC implementation.

BIBLIOGRAPHY

[1] Dony, R.D., and Haykin, S., Neural Network Approaches to Image Compression, Proceedings of the IEEE, (1995), Vol.23, No.2, pp 289–303.

[2] Namphol, A. et al., Image Compression with a Hierarchical Neural Network, IEEE Transactions on Aerospace and Electronic Systems,(1996), Vol.32, No.1, pp.327–337.

[3] Blumenstein, M., The Recognition of Printed and Handwritten Postal Address using Artificial Neural Networks, (1996), Dissertation, Griffith University, Australia.

[4] Jiang, J., A Neural Network Design for Image Compression and Indexing. International Conference on Artificial Intelligent Expert Systems and Neural Networks, (1996), Hawaii, USA, pp 296–299.

[5] Ivan Vilvovic, "An experience in Image compression using neural networks', 48th International Symposium ELMAR-2006, June 2006, Zadar, Croatia.

[6] K.Venkata Ramanaiah, Dr K.Lal Kishore, and Dr P.Gopal Reddy "Power Efficient Multilayer Neural Network for Image Compression" Information Technology journal 6(8): 1252–1257 ISSN1812-5638.@ 2007 Asian Network for Scientific Information.

[7] K.Venkata Ramanaiah, Dr K.Lal Kishore, and Dr P.Gopal Reddy "New Architecture for NN based Image Compression for optimized Power, Area and Speed". i-managere's Journal on Electrical Engineering. Vol. 1. No. 3. ISSN-0973-8835. Jan- March -2008,754