Australian Journal of Basic and Applied Sciences, 4(3): 379-388, 2010
ISSN 1991-8178

A FPGA Implementation of Neural/Wavelet Face Detection System

[1,2]Hossein Sahoolizadeh, [3]Ahmad Keshavarz

[1]Islamic Azad Universitiy, Dashtestan branch, Iran
[2]Behin Tadbir Intelligent Systems Co, Kermanshah, Iran
[3]Persian Golg University, Bushehr 75169, Iran

Abstract: In this paper the FPGA implementation of composition of an Artificial Neural Network (ANN) architecture and Wavelet transforms for a face detection system is considered. A novel technique is proposed to design the face detection system. First, Wavelet Transforms are applied in the input image in order to extract particular features that are detected by a locally connected Multilayer Perceptron network.. We then propose an FPGA implementation which entails very low logic and memory costs and achieves extremely high processing rates at low clock speeds.

Key words: Face detection system, DWT, Neural network, MLP, FPGA

INTRODUCTION

Face detection in complex environments is a challenging problem which has fundamental importance to model-based video coding, image-database, content-based image retrieval, and face recognition systems. Face detection in real-time processing is an important and preliminary step of a variety of applications requiring intelligent human-computer interaction(Mohabbati, B., M. Shiri, 2005).Numerous approaches for face detection have been proposed in the last decade, many of them described and compared in two interesting recent surveys by Yang *et al*. (2002) and Hjelmas *et al*. (2001). These paradigms include Geometrical Features, Eigenfaces, Template Matching, Graph Matching and Neural Network approaches. Most face detection methods are based on local facial feature detection and classification using statistical and geometric models of the human face. Low level analysis first deals with the segmentation of visual features using image properties such as edges (Govindaraju, V., 1996), intensity (Yang, G. and T.S. Huang, 1994), color (Garcia, C. and G. Tziritas, 1999; Feraud, R., O. Bernier, 2002), motion (Low, B. and M. Ibrahim, 1997), or generalized measures (Lin, C.C. and W.C. Lin, 1996). Other approaches are based on template matching where several correlation templates are used to detect local subfeatures, considered as rigid in appearance (eigenfeatures (Moghaddam, B. and A. Pentland, 1997)) or deformable (Wiskott, L., J. Fellous, 1997; Garcia, C., G. Simandiris, 2001). Then, visual features are organized into a more global concept of face through facial feature and constellation analysis using face geometry constraints (Garcia, C., G. Simandiris, 2001; Yow, K. and R. Cipolla, 1997; Jeng, S., H. Yao, 1998; Maio, D. and D. Maltoni, 2000). Low cost hardware implementation of an accurate face detection system is useful in a wide range of applications. FPGA becomes the most applicable microelectronic technology in many recent applications such as communication, mobile telephone, etc. This is due to the relatively high capacity and low cost of the FPGA and also, short design cycle and short time to market when using EDA tools. The FPGAs provide a new implementation platform for the discrete wavelet transform. FPGAs maintain the advantages of the custom functionality of VLSI ASIC devices, while avoiding the high development costs and the inability to make design modifications after production.Furthermore; FPGAs inherit design flexibility and adaptability of software implementations. Many of the literature has proposed hardware implementation of neural networks, but it does not have learning ability (Botros, N.M., M. Abdul-Aziz, 1994; Faiedh, H., Z. Gafsi, 2004). Some researchers have proposed hardware implementation of neural networks with on-chip learning that uses the BP algorithm (Hikawa, H., 2003). In this paper, we propose an efficient FPGA hardware implementation of system to detect face by combining neural network and wavelet representation properties. First of all, Wavelet Transforms are applied in the input image in order to extract particular features which are detected by a locally connected Multilayer Perceptron network (Daubechies, I., 1992; Haykin, S., 1999). The idea of locally connected neurons was inspired from Hubel and Wiesel's research on the locally sensitive

Corresponding Author: Hossein Sahoolizadeh, Islamic Azad Universitiy, Dashtestan branch, Iran
Email: hossein_sahooli@yahoo.com

neurons in the cat visual system (Hubel, D.H. and T.N. Wiesel, 1962). This work aims to improve and simplify the feature extraction process proposed in (Garcia, C. and M. Delakis, 2004), based in a convolutional neural network architecture which was initially proposed by LeCun *et al* (1998). The convolutional architecture is based on a highly complex process that automatically synthesizes the feature extractors from a training set of face and non-face patterns and classifies them accordingly. The organization of this paper is as follows: In the next section we introduce our face understudied detection system .In the section 3, we review DWT and neural networks as our tools to design the whole face detection system. In the section 4, we consider VHDL-RTL design and FPGA implementation of the system. Results are discussed in section 5 and finally section 6 makes some conclusions.

2-The Face Detection System:

The proposed solution is an intelligent mechanism which recognizes a given image as if being face or not. For that matter, two specific Wavelet functions, the Haar and Daubechies , are used in order to extract features from a given image, which will then feed the Multilayer Perceptron locally connected input fields, responsible for their classification as being face or nonface. Before feeding the network, the input values are normalized to the range of -1 to 1. Resulting from the 3-D Wavelet Transforms in a 32x32 pixel image, come eight 4x4 pixel images, being four of them processed by the Haar function, and the other four by the Daubechies one. Each of these represents, respectively, the low-pass-low-pass (LL), low-pass-high-pass (LH), high-pass-lowpass (HL) and high-pass-high-pass (HH) filtering. In both cases, third level transformations wavelets were applied. The proposed MLP consists of two layers of hyperbolic tangent neurons. The first layer, the hidden one, has eight neurons, where each one of them is locally connected to a specific resulting image from the Wavelet transforms. The second layer consists of only one neuron, fully-connected to the eight ones in the previous layer, which will output positive signal for face patterns and negative signal for the non-face ones. Fig. 1 shows the details of the proposed architecture.

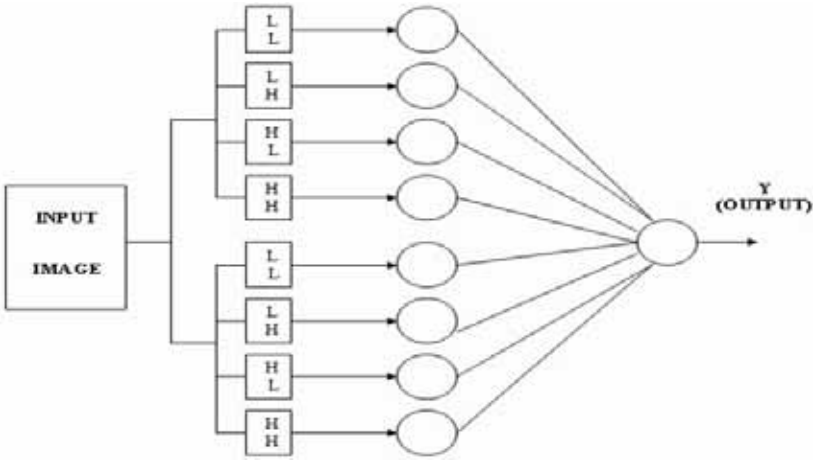

Fig. 1: The proposed networkfor face detection system.

3-Design of the System:

3.1. Wavelet Design:

As mentioned before, we utilize two important wavelet transforms to efficiently implement Face detection system. These discrete wavelet transforms are Haar and Daubechies wavelet transforms. Haar transform decomposes a discrete signal into two subsignals of half its length.One subsignal is a running average or trend; the other subsignal is a running difference or fluctuation. The Haar wavelet transform has a number of advantages (James S. Walker, 1999): It is conceptually simple, fast, and since it can be calculated in place without a temporary array, is memory efficient. The Daubechies wavelet transforms are defined in the same way as the Haar wavelet transform by computing the running averages and differences via scalar products with scaling signals and wavelets the only difference between them consists in how these scaling signals and wavelets are defined(James S. Walker, 1999). The Daubechies wavelet is more complicated than the Haar wavelet. Daubechies wavelets are continuous; thus, they are more computationally expensive to use than the Haar wavelet, this wavelet type has balanced frequency responses but non-linear phase responses. Daubechies

wavelets use overlapping windows, so the high frequency coefficient spectrum reflects all high frequency changes. he discrete wavelet transform (DWT) is simply implemented by a 2- band reconstruction block as shown in Fig. 2 The input signal X(z) is split by two filters H0(z) and H1(z) into a low pass component X0 and a high pass component X1 , oth of which are decimated (down-sampled) by 2:1. In order o reconstruct the signal, a pair of reconstruction filters G0(z) nd G1(z) and usually the filters are designed such that output signal Y(z) is identical to the input X(z). This is known as the condition for perfect reconstruction (PR) (Nick, K. and M. Julian, 1997).The art of finding good wavelet lies in the design of the set of filters, to achieve various tradeoffs between spatial and frequency domain characteristics while satisfying the perfect reconstruction condition. Now we are going to discuss the condition for perfect reconstruction. As shown in figure1 the process of decimation and interpolation by 2:1 at the output of H0 (z) and H1 (z) effectively sets all odd samples of these signals to zero.

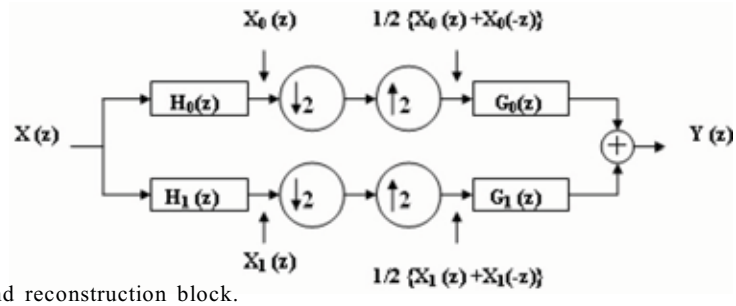

Fig. 2: The 2-band reconstruction block.

3.2. Neural Design of the System:

Multi Layer Perceptron (MLP) Neural Network is a good tool for classification purposes .It can approximate almost any regularity between its input and output (Haykin, S., 1999). The ANN weights are adjusted by supervised training procedure called backpropagation. Backpropagation is a kind of the gradient descent method, which search an acceptable local minimum in the ANN weight space in order to achieve minimal error. Error is defined as a root mean square of differences between real and desired outputs of NN. During the training procedure MLP builds separation hypersurfaces in the input space. After training MLP can successfully apply acquired skills to the previously unseen samples. It has good extrapolative and intrerpolative abilities. Fig. 3 illustrates a sing-output MLP network with inputs X1…Xn and output.

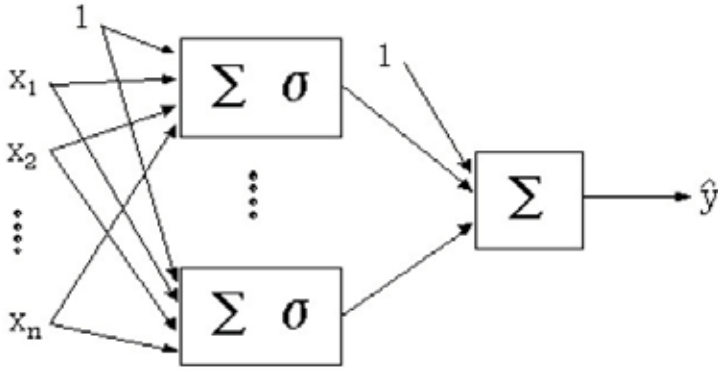

Fig. 3: A MLP network with one hidden layer and one output

Each arrow in the figure symbolizes a parameter in the network. The network is divided into layers. The input layer consists of just the inputs to the network, and then follows a hidden layer, which consists of any number of neurons, or hidden units placed in parallel. Each neuron performs a weighted summation of the inputs, which then passes a nonlinear activation function, σ, also called the neuron function. Mathematically the functionality of a hidden neuron is described by

$$\sigma\left(\sum_{j-1}^{n} w_j x_j + b_j\right)$$

Where the weights $\{w_j, b_j\}$ are symbolized with the arrows feeding into the neuron. The network output is formed by another weighted summation of the outputs of the neurons in the hidden layer. This summation on the output is called the output layer. In Fig. 2 there is only one output in the output layer since it is a single-output problem. Generally, the number of output neurons equals the number of outputs of the related problem. The output of this network is given by

$$\hat{Y}(\theta) = g(\theta, x) = \sum_{i=1}^{nb} w_i^2 \sigma\left(\sum_{j-1}^{n} w_{i,j}^1 x_j + b_{j,i}^1\right) + b^2$$

where n is the number of inputs and nh is the number of neurons in the hidden layer. The variables $\left(W_{i,j}^1 b_{j,i}^1 w_i^2 b^2\right)$ are the parameters of the network model that are represented collectively by the parameter vector, θ. In this paper we use compact notation $g(\theta,x)$ representing the neural network model. The size of the input and output layers are defined by the number of inputs and outputs of the network and, therefore, only the number of hidden neurons has to be specified when the network is defined. The nonlinear activation function in the neuron is usually chosen to be a smooth step function. The default is the standard sigmoid that is shown in Fig. 4.

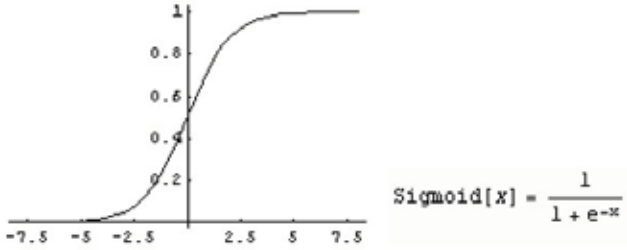

Fig. 4: The sigmoid function that is used as a standard activation function

The number of layers and the number of hidden neurons in each hidden layer are the parameters that will be designed by user. The general rule is to choose these design parameters so that the best possible model with as few parameters as possible is obtained. For many practical applications, one or two hidden layers will suffice. The output neurons in the MLP networks in Fig. 3 are linear; that is, they do not have any nonlinear activation function after the weighted sum. In training the MLP network, its parameters are adjusted incrementally until the training data satisfy the desired mapping as well as possible; that is, until $\hat{Y}(\theta)$ matches the desired output y as closely as possible up to a maximum number of iterations.

3.3. System Design:

Hardware implementation of the proposed system consists of three building blocks: 1-Input image reception, scaling and buffering, 2- 3-D DWT computation, 3-Classification. Fig. 5 shows the basic building blocks of the system.

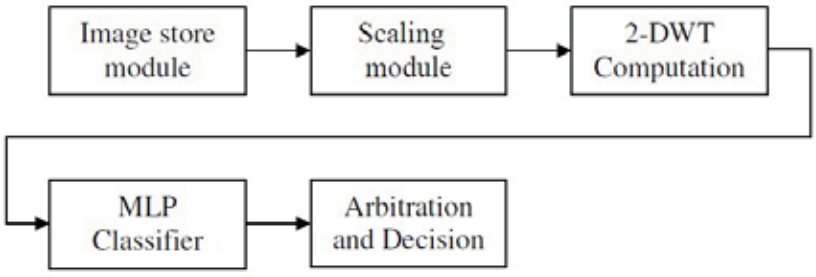

Fig. 5: The basic building block s of the system

The image store module stores the image data arriving from the input. This module transfers the image data to the 3-D DWT computation module based on the scale information from the image scalar module. The image is stored in a BRAM of the FPGA. The images are scaled down based on a scale factor by the image scalar module. The image scalar module generates and transfers the address of the BRAM containing a frame image in the image store module to request image data according to a scale factor. The image store module transfers a pixel data to the classifier module based on the address of BRAM required from the image scalar module. A scaling image technique is used in hardware instead of the scaling sub-window because it does not need a huge cache memory for fast memory access and it is easy to implement in hardware. Since our architecture has a fixed integral image window (32×32 pixels), it needs to scale input images down to detect large faces. To make scaled images, we use a nearest neighbor interpolation algorithm with a factor of 1.2. A pixel value in the scaled images is set to the value of the nearest pixel in the original images. This is the simplest interpolation algorithm that requires a lower computation cost. The number of the scaled images depends on the input image resolution. Our scalar module performs the down-scaling of input images until the height of the scaled image is similar with the size of the image window (32x32 pixels) The Arbitration and Decision module stores the outputs of the MLP for each scale value In a separate array. Values in this array indicate the output of the MLP for that special location of the image. Since the network has some invariance for scaling and positioning, multiple answers may result for a particular face. This module counts the number of other detections within a specified neighborhood of that detection. If the number is above a threshold, then that location is classified as a face. In order to prevent overlaps, once each location is considered as a face any other detection in a specified neighborhood are discarded. The above analysis is performed for each array. Final results are merged into one output array, indicating the location for face objects (Mohammad, S. Sadri *et al*, 2004).

4- The FPGA Implementation of the System:

In general, the process of designing a system will proceed from a behavioral to a physical representation.High level synthesis converts a behavioral specification of a digital system into an equivalent RTL design that meets a set of stated performance Constraint (Gajski, D., N. Dutt, A. Wu, 1992) Programming in hardware is advantageous because gates are low-cost and there is a possibility to guarantee the time behavior of the functions. FPGAs are a form of programmable logic, which offer flexibility in design like software, but with performance speeds closer to Application Specific Integrated Circuits (ASICs). With the ability to be reconfigured an endless number of times after having been manufactured. FPGAs have traditionally been used as a prototyping tool for hardware designers. FPGA are more common than for example ASIC because of their size, cost and performance.

4.1 -Wavelet FPGA Implementation:

The design of our system is started with Forward Haar and Daubechies wavelet transforms implementation. The basic building block of the forward Haar or Daubechies discrete wavelet transform filter bank is the decimator which consists of an FIR filter followed by a down-sampling operator (Vaidyanathan, P., 1993). Down-sampling an input sequence x[n] by an integer value of 2, consists of generating an output sequence y[n] according to the relation y[n] = x[2n]. Accordingly, the sequence y[n] has a sampling rate equal to half of that of x[n]. We implemented the decimator as shown in Fig. 6. An active-high output control pin, labeled load, has been implemented in FIR filter structure and connected directly to the CLK input of a 1-bit counter. The input port of the FIR filter is connected to the input samples source, the input port of the FIR filter is connected to the input samples source, whereas the output port is connected to a parallel-load register. The register loads its input bits in parallel upon receiving a high signal on its load input from the 1-bit counter, and blocks its input otherwise.

Assuming unsigned 8-bit input samples, the decimator operates as follows: When the load signal is activated, every time the FIR completes a filter operation, it triggers the counter to advance to the next state. If the new state is 1, the parallel-load register is activated, and it stores the data received at its input from the FIR filter. If the new state is 0, the register is disabled, and consequently the FIR output is blocked from entering the register, and ultimately discarded. The above procedure repeats, so that when the state machine has 1 on its output, the FIR data is stored, and when it has a 0 on its output, the FIR data is discarded. Fig. 7 shows the VHDL Entity block diagram of the design.

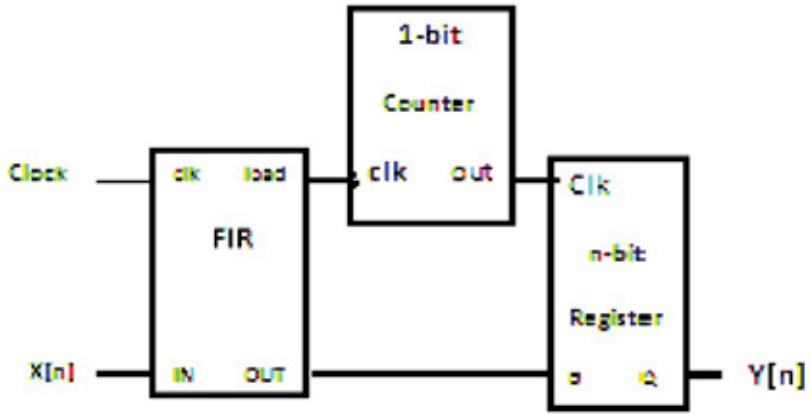

Fig. 6: Implementation of the basic blocks Forward DWT

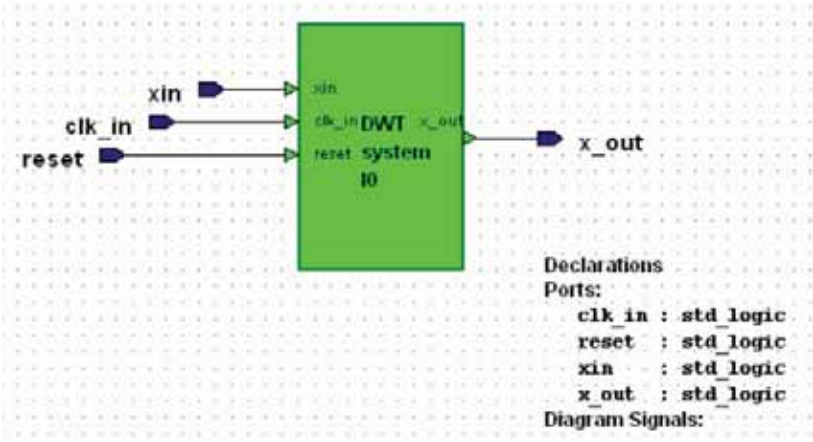

Fig. 7: Entity block diagram of the design

4.2-FPGA Mapping of ANN:

The digital NN hardware implementations are further classified as (i) FPGA-based implementations (ii) DSP-based implementations (iii) ASIC-based implementations (Chen, Y.J., Du Plessis, 2002; Sund Su Kim, Seul Jung, 2004).DSP based implementation is sequential and hence does not preserve the parallel architecture of the neurons in a layer. ASIC implementations do not offer reconfigurablity by the user. FPGA is a suitable hardware for neural network implementation as it preserves the parallel architecture of the neurons in a layer and offers flexibility in reconfiguration. FPGA realization of ANNs with a large number of neurons is a challenging task. Selecting weight precision is one of the important choices when implementing ANNs on FPGAs. Weight precision is used to trade-off the capabilities of the realized ANNs against the implementation cost. A higher weight precision means fewer quantization errors in the final implementations, while a lower precision leads to simpler designs, greater speed and reductions in area requirements and power consumption. One way of resolving the trade-off is to determine the “minimum precision” required. In this work we consider the weights as 8 -bit fixed-point values. Direct implementation for non-linear sigmoid transfer functions is very expensive. As the excitation function is highly nonlinear we adopt the Look Up Table (LUT) method in order to simplify function computation. The LUT is implemented using the inbuilt RAM available in FPGA IC. The use of LUTs reduces the resource requirement and improves the speed. Also the implementation of LUT needs no external RAM since the inbuilt memory is sufficient to implement the excitation function. Fig. 8 describes the basic structure of the functional unit (neuron) that implements the calculations associated with neuron.

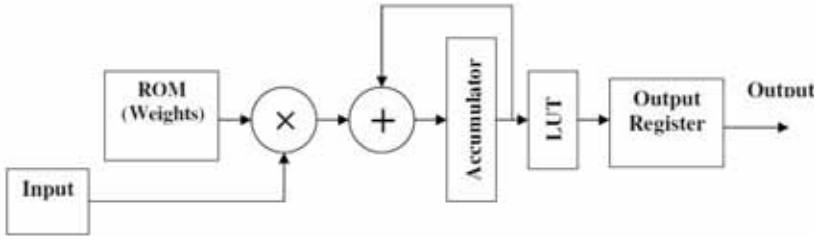

Fig. 8: The neuron RTL block diagram

Each neuron has a ROM, which stores the weights of the connection links between the particular neuron to the neurons of the previous layer. The multiplier performs high speed multiplication of input signals with weights from ROM. A sixteen bit register is used to hold the weights from the ROM and the input signal from the previous layer. The whole MLP implementation is shown in Fig. 9. The network mainly consists of input registers, control unit, neurons and output register. To provide on neuron output to the next stage at each clock cycle a Multiplexer and a counter is used. The training of the network is done in software and the results loaded into hardware .Weights are updated during the training process, but remain constant during the detection process. The Register Transfer Level design of the system has been carried out using standard VHDL as the hardware description language. This language allows three different levels of description. We have chosen RTL to implement this system. The entire design process was done using the ISE development tool, from Xilinx (ISE development). The system physically implemented on Spartan-3 XC3 S4000 XILINX FPGA device. The attributes of this device is summarized in table 1.

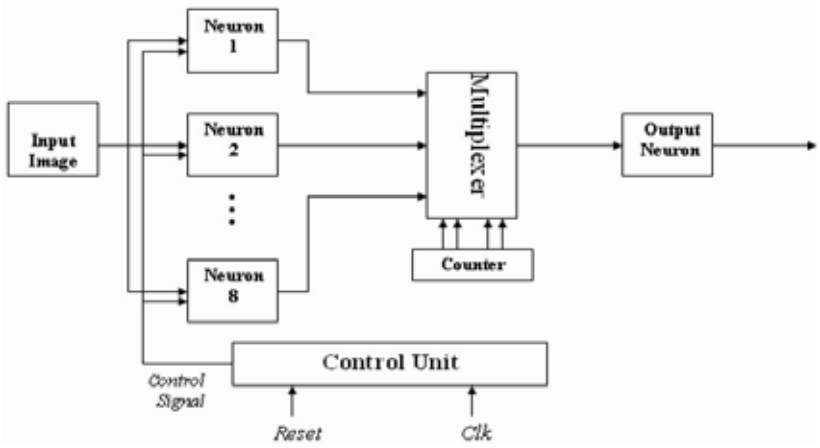

Fig. 9: The RTL block diagram of MLP neural network

Table 1: Summary of Spartan-3 XC3S4000 FPGA Attributes

Device	System Gates	Logic Cells	Array CLB (One CLB = Four Slices)			Distributed RAM (bits1)	Block RAM (bits1)	Dedicated Multipliers	DCMs	Maximum User I/O	Maximum Differential I/O Pairs
			Rows	Columns	Total CLBs						
XC3S4000	4M	62,208	96	72	6,912	432K	1,728K	96	4	712	312

RESULTS AND DISCUSSION

The MLP was trained by using the backpropagation algorithm with 0.00 1 learning rate and 0.5×10 -6 precision, with randomly initialized weights. In order to satisfy the locally-connected input condition, the backpropagation algorithm was modified so that, during the weight adjustment in the hidden layer, only the input values related to its respective neuron were considered. The used training set contained two hundred grayscale images of both face and non-face patterns. The face patterns were taken from the AR Database (Martinez, A.M. and R. Benavente, 1998) with each of them being resized to 32x32 pixels using the nearest neighbor interpolation and manually cropped to contain only features from eyes, nose and mouth. Non-face patterns were created by selecting images from random daily situations, such as landscapes and regular objects. It is also important to mention that the false positives resulting from previous network trainings were fed into the training set, so that the MLP could learn from its own mistakes. Thirty images were used for testing, sixteen of them are faces, and the other fourteen of them are non-face images. The training was done inn

offline mode. The hardware used for the training was an Intel Pentium 4 processor running at 3.2 GHz and 1GB of DDR RAM. The MATLAB package was used for training the network in offline mode. Table2 shows the results of training.

Table 2: Results for training of MLP

Epoch	Total offline training time	Correct answers
1150	17.53	96.8%

Throughout the tests, the network presented a high error rate in non-face patterns. Thus, to improve its performance, the training set was increased with a greater number of non-face patterns, mainly taken from false positives of previous trainings. After training the weights of trained network downloaded to FPGA ROM. It was noticed that, in most false positive cases, the images contained occluded faces, such as people wearing glasses, which reflected the lights and causing confusion in the extraction of such features. In other cases, the faces were too bent to the sides, which caused misplacement of certain important features. The network showed great efficiency in complex images, such as smiling faces, partially occluded images and faces with great lighting variation. The test result for a given image is shown in Fig 10.

Fig. 10.1: U.S. Religious Delegation to Iran

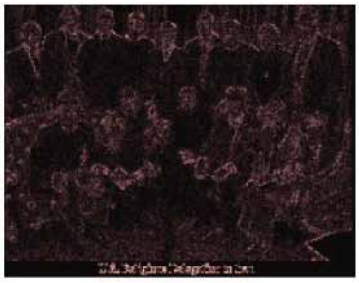

Fig. 10.2:

Fig. 10.3:

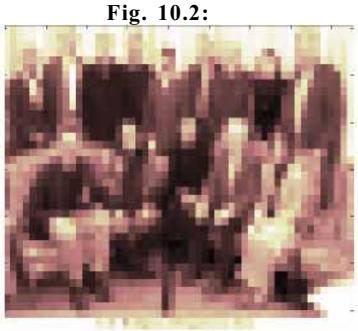

Fig. 10.4:

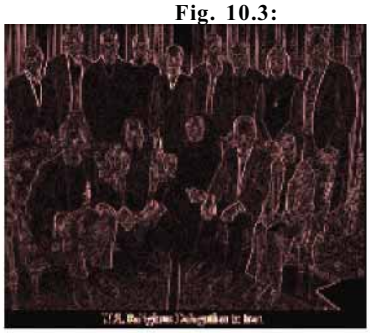

Fig. 10.5:

Fig. 10.6:

Fig.10: Test result for a given image,(10.1) The input image ,(10.2) Horizontal detail coefficient of DWT(HL filtering),(10,3) Diagonal detail coefficient of DWT(HH filtering),(10.4) Vertical detail coefficient of DWT(LH filtering),(10.5)Approximation coefficient of DWT(LL filtering),(10.6)out

6-Conclusion:

In this paper wavelet+Neural Network methods were studied. To verify the circuit, first of all, the NN was designed and verified on the PC using MATLAB. Next, the weights and test bench data were saved to verify the VHDL code. After the simulation of the NN, area and operating speed were obtained by synthesis using ISE XILINX tool. We have chosen an FPGA as our implementation hardware because it combines the reprogrammability advantage of general purpose processors with the parallel processing and speed advantages of custom hardware. The size and speed evaluation of the proposed FPGA circuit revealed its low cost in terms of logic and memory. The proposed architecture is implemented on a sparran-3 CX3S4000 FPGA and its performance is measured and compared with an equivalent software implementation. We find increase of system performance over the equivalent software implementation By the analysis of the results, it can be said that they were well satisfying.

ACKNOWLEDGMENT

The authors would like to express their gratitude to Mr.Behnam Karami, the manager of Behin Tadbir Intelligent Systems Company for providing us measurements.

REFERENCES

Botros, N.M., M. Abdul-Aziz, 1994. Hardware implementation of an artificial neural network using field programmable gate arrays (FPGA's), IEEE Transactions on Industrial Electronics, 41(6): 665-667.

Chen, Y.J., Du Plessis, 2002. "Neural Network Implementation on a FPGA , Proceedings of IEEE Africon, 1: 337-342.

Daubechies, I., 1992. Ten Lectures on Wavelets, CBMS-NSF Regional Conference Series in Applied Mathematics, Rutgers University and AT&T Bell Laboratories, 61.

Faiedh, H., Z. Gafsi, K. Torki, K. Besbes, 2004. Digital hardware implementation of a neural network used for classification, in: The 16th International Conference on Microelectronics, Dec., pp: 551-554.

Feraud, R., O. Bernier, J. Viallet and M. Collobert, 2002. A Fast and Accurate Face Detector Based on Neural Networks,IEEE Trans. Pattern Analysis and Machine Intelligence, 23(1): 42-53.

Gajski, D., N. Dutt, A. Wu, 1992. "High-Level Synthesis: Introduction to Chip and System Design", Kluwer Academic Publishers, Boston.

Garcia, C., G. Simandiris and G. Tziritas, 2001. A Feature-Based Face Detector Using Wavelet Frames, Proc. Int'l Workshop Very Low Bit Coding, pp: 7 1-76.

Garcia, C. and M. Delakis, 2004. Convolucional Face Finder: A Neural Architecture for Fast and Robust Face Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(11): 1408-1422.

Garcia, C. and G. Tziritas, 1999. Face Detection Using Quantized Skin Color Regions Merging and Wavelet Packet Analysis,IEEE Trans. Multimedia, 1(3): 264-277.

Govindaraju, V., 1996. Locating Human Faces in Photographs,"Int'l J.Computer Vision, 19(2): 129-146.

Haykin, S., 1999. Neural Networks: A Comprehensive Foundation, 2nd ed., Prentice-Hall.

Hikawa, H., 2003. A digital hardware pulse-mode neuron with piecewise linear activation function, IEEE Transactions on Neural Networks, 14(5): 1028-1037.

Hjelmas, E. and B.K. Low, 2001. Face Detection: A Survey, Computer Vision and Image Understanding, 83: 236-274.

Hubel, D.H. and T.N. Wiesel, 1962. Interaction and Functional Architecture in the Cat's Visual Cortex, J. Physiology, 160: 106-154.

Inês Ap G. Boaventura, 2005. A face detector using Neural Networks and Discrete Wavelet Transforms, XVIII Brazilian Symposium on Computer Graphics and Image Processing, Natal, Brazil.

ISE development tool, from Xilinx

Jeng, S., H. Yao, C. Han, M. Chern and Y. Liu, 1998. Facial Feature Detection Using Geometrical Face Model: An Efficient Approach, Pattern Recognition, 31(3): 273-282.

James S. Walker, 1999. A Primer on Wavelets and Scientific Applications, CRC; 1 edition, ISBN-10: 0849382769.

LeCun, Y., Y. Bengio and P. Haffner, 1998. "Gradient-Based Learning Applied to Document Recognition", Proceedings of the IEEE, 86(11) 2278-2324.

Lin, C.C. and W.C. Lin, 1996. Extracting Facial Features by an Inhibitory Mechanism Based on Gradient Distributions, Pattern Recognition, 29: 2079-2101.

Low, B. and M. Ibrahim, 1997. A Fast and Accurate Algorithm for Facial Feature Segmentation," Proc. Int'l Conf. Image Processing.

Maio, D. and D. Maltoni, 2000. Real-Time Face Location on Gray-Scale Static Images, Pattern Recognition, 33: 1525-1539.

Martinez, A.M. and R. Benavente, 1998. The AR Face Database, CVC Technical Report no. 24.

Mohabbati, B., M. Shiri, Sh. Kasaei, 2005. An efficient wavelet/neural networks-based face detection algorithm, Internet, 2005 .The First IEEE and IFIP International Conference in Central Asia on Publication Date: 26-29 Sept.

Moghaddam, B. and A. Pentland, 1997. Probabilistic Visual Learning for Object Recognition, IEEE Trans. Pattern Analysis and Machine Intelligence, 19(7): 696-710.

Mohammad, S. Sadri *et al*, 2004. An FPGA Based Fast Face detector, Global Signal Processing Expo & Conference (GSPx), Santa Clara, CA.

Nick, K. and M. Julian, 1997. Wavelet Transform in Image Processing. Signal Processing and Prediction I, Eurasip, ICT press, 23-34.

Sund Su Kim, Seul Jung, 2004. "Hardware Implementation of Real Time Neural Network Controller with a DSP and an FPGA ", IEEE International Conference on Robotics and Automation, 5: 3161-3165.

Vaidyanathan, P., 1993. Multirate Systems and Filter Banks. New Jersey: Prentice Hall. ISBN-10: 0136057187.

Wiskott, L., J. Fellous, N. Kruger and C.V. der Malsburg, 1997. Face Recognition by Elastic Bunch Graph Matching,IEEE Trans. Pattern Analysis and Machine Intelligence, 19(7): 775-779.

Yang, M., D. Kriegman and N. Ahuja, 2002. Detecting Faces in Images: A Survey, IEEE Trans. Pattern Analysis and Machine Intelligence, 24(1): 34-58.

Yang, G. and T.S. Huang, 1994. Human Face Detection in Complex Background,Pattern Recognition, 27(1): 53-63.

Yow, K. and R. Cipolla, 1997. Feature-Based Human Face Detection, Image and Vision Computing, 15(9): 713-735.