FPGA-Based Handwritten Signature Recognition System

Sami El Moukhlis1, Abdessamad Elrharras, Abdellatif Hamdoun

Abstract— In this paper, a method of classification of handwritten signature based on neural networks, and FPGA implementation is proposed. The designed architecture is described using Very High Speed Integrated Circuits Hardware Description Language (VHDL). The proposed application consists of features extraction from handwritten digit images, and classification based on Multi Layer Perceptron (MLP). The training part of the neural network has been done by using MATLAB program; the hardware implementations have been developed and tested on an Altera DE2-70 FPGA.

Keywords— ANN, FPGA, MLP, Recognition, VHDL.

I. INTRODUCTION

Security is a major preoccupation at the international level. Certain technologies, which were still marginal, were then grown considerably. Currently, the signature verification is a very important tool for authentication of an individual, it is also used to authenticate a document, validate a contract or financial transaction, or as a password to view confidential documents. It can be significant value in security and e-commerce applications. Many computer systems are proposed in passports and identity cards, as well as many other uses related to security.

The main difficulty for based authentication on signature is that signatures of the same person are not exactly the same. To ensure that this recognition is accurate, it is necessary that the variation between the signatures of the same person be less than the distance between the signatures of two different people. This fact makes the problem of authenticating a classification problem.

The main purpose of this work is to propose a system identification of handwritten signature offline, which consists in providing solutions to four problems: a) Data acquisition, b) Pre treatment, c) Features extraction and d) Decision.

We used MATLAB to extract characteristics from handwritten signatures and to train the ANN in order to calculate the weights.

The hardware implementation describes the design of a digital system architecture realizing a feed-forward multilayer neural network; the network is implemented in Altera Cyclone II FPGA embedded in Terasic DE2-70 Board [1].

Manuscript received April, 2014.

Sami El Moukhlis, Information processing Department, FSBM / HASSAN II University / Casablanca, Morocco.

Abdessamad Elrharras, Information processing Department, FSBM / HASSAN II University / Casablanca, Morocco.

Abdellatif Hamdoun, Information processing Department, FSBM / HASSAN II University / Casablanca, Morocco.

II. FEATURES EXTRACTION

A. Review Stage

Normally, a digitized signature is represented by a pixels matrix with a large size. To get around the problems of performance and efficiency, we present an effective procedure for pre-processing the signatures images, which is capable of extracting primitives that present, on the one hand a certain geometric invariance, and secondly some statistical invariance. While the geometric invariance signifies tolerance of the operator's translations, rotation and scaling, and the statistical invariance mean a tolerance of inevitable noise. Indeed, after this step, the signature will no longer be represented by a matrix of pixels, but by a feature vector [5]. The segmentation of the image is described by the following steps:

i) Let IM be the signature image with three dimensions: width 'l', languor 'L', and color indices 'RGB'

ii) The conversion from the RGB to gray-level, consist in forming a pixels matrix 'I' with two dimension (l, L) from each pixel IM (i, j), such that:
$I(i,j) = 0.2989 * R + 0.5870 * G + 0.1140 * B$ (1)

iii) Thresholding the image consists in comparing each pixel with a threshold's':
If $I(i, j) > s$ then $I(i, j)=1$ else $I(i, j)=0$

iv) The binary image can be presented by a distinct pair of horizontal and vertical projections. These projections are simply the number of black pixels in each line or column. To harmonize projections with the number of input neurons of the network, it is necessary to resize the Final Stage signature image processed on a grid of 20 × 20 pixel (the size of the grid is arbitrary).

v) In addition to the horizontal and vertical projections, the feature vector also contains static characteristics related to the form:

- Density: Number of black pixels over the total area the image of the signature.
- The centre of gravity: The average position of the black pixels in the image. This average position has expressed as a percentage of x and Y relative to the width and height of the image:

$$G_x = \frac{\sum_{i=1}^{n} x_i}{n \times \text{width}} \quad (2) \qquad G_y = \frac{\sum_{i=1}^{n} y_i}{n \times \text{height}} \quad (3)$$

III. ARTIFICIAL NEURAL NETWORKS

A. Formal neuron

The artificial neural networks (ANN) are trying to mimic the biological neural structures [3].

A formal neuron is a mathematical representation of the biological neuron. The artificial neuron generally has several inputs, and only one output. The actions of excitatory synapses are represented by numerical co-efficient (the synaptic weights) associated with the inputs. The numerical values of these co-efficient are adjusted in a training phase. In its simplest version, an artificial neuron calculates the weighted sum of the received inputs, and then this value applies the activation function.

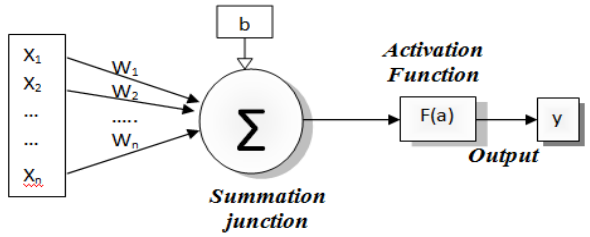

Fig.1 Mathematical model of the formal neuron

The formal neuron that is given in the figure above has n inputs denoted as$\{X_1, X_2 \ldots X_n\}$. Each line that connects these inputs to the summation junction is assigned a weight, denoted as$\{W_1, W_2 \ldots W_n\}$. The neuron activation function F(a) in McCulloch-Pitts model is a threshold function: However, linear and sigmoid functions are also used in different situations. The output y of the neuron is given by the formula:

$$y = f(\sum W_i * X_i + b) \qquad (4)$$

One of the most important parts of a neuron is its activation function. In this work, we have chosen a sigmoid function, because it's nonlinearity of making it possible to approximate any function.

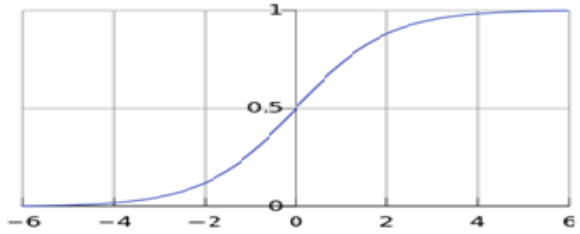

Fig.2 sigmoid function

B. Multi-Layer Perceptron

The multi-layer perceptron (MLP) is a neural network model which is organized in the form of layers. In the input layer, we have found neurons that accept the input variables$\{X_1, X_2 \ldots X_n\}$, then the hidden layer, and an output layer that should indicate the appropriate class for the object presented in the input. The neurons in a layer i get inputs from layer i-1 and feed their output to layer i+1.These type of networks are called feed-forward networks.

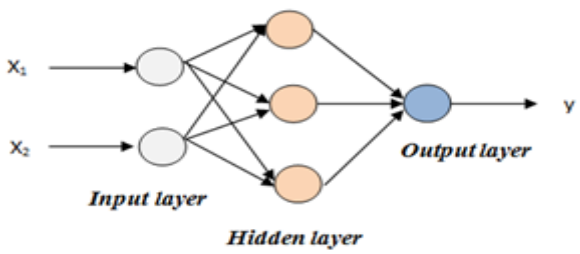

Fig.3 MLP model

MLP are especially trained using the back-propagation algorithm [14], which aims at minimizing the global error measured at the output layer, by the relation bellow:

$$e(t) = y_d(t) - y_m(t) \qquad (5)$$

Where $y_d(t)$ denotes the desired output, and $y_m(t)$ the measured output of the neuron.

The BP algorithm uses an iterative supervised learning procedure, where the MLP is trained with a set of predefined inputs and outputs. And, the global error $E_g(t)$ is calculated by equation (5), this error can be minimized by the gradient descent technique.

$$E_g(t) = \frac{1}{2}\sum_{i=1}^{n}\left(y_{d,i}(t) - y_{m,i}(t)\right)^2 \qquad (6)$$

IV. PROPOSED DESIGN

The proposed design consists of an MLP neural network two layers which is one of the most architecture used in pattern recognition each layer is fully connected with its adjacent layers.

Each layer contains a number of neurons which is the processing element of the ANN, and activation function which is the most important arithmetic operation required for neural networks.

A. Architecture of the neuron

A Neuron can be viewed as processing data in three steps; the weighting of its input values, the summation of them all and their filtering by sigmoid function.

Figure 1 shows the block diagram of the designed neuron, the MAC unit which accepts a serial processing of weights and parallel inputs pairs, each pair is multiplied together and added to the next multiplication until the end. The result is fed in the activation function.

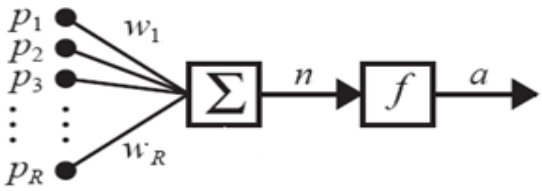

Fig.4: Architecture of neuron.

The implementation of the trained network was made by VHDL language. This design is synthesized by Quartus II. The figure 5 shows the block diagram of a neuron, it s composed of multiplier that multiply the weight and the input and an accumulator that mad the sum of all the multiplication and at the end the result is put in the activation block which validate or not the result.

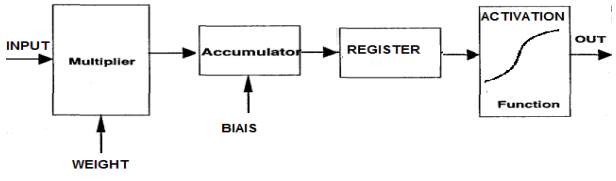

Fig.5 Block diagram of a neuron

We implement our circuit on Cyclone II FPGA from ALTERA, the implementation of the neurons has been done by using 16-bit MAC (Multiply Accumulate) circuit, the sigmoid function is implemented both in the hidden an output layer.

B. Activation function

The important characteristics [2] of the network depend on its structure, the activation function and the learning mechanism.

The activation function [6] is used to limit the value of the neuron output. It's one of the most important parts of an ANN. The sigmoid function (1) is the most frequently used activation function in back-propagation neural networks applications. It is not suitable for direct implementation because it consists of an infinite exponential series. For its implementation function on FPGA, we use Piece-Wise Linear Approximation which consists of a linear approximation of 'logsig' function; the mathematical representation is shown in equation 2. This method has sufficient precision for our implementation.

$$\theta(v) = \frac{1}{(1+e^{-v})} \qquad (7)$$

$$\theta(v) = \begin{cases} 1 & for\ v \geq 4 \\ 0.0625v + 0.75 & for\ 0 < v < 4 \\ 0.5 & for\ v = 0 \\ 0.0625v + 0.25 & for -4 \leq v < 0 \\ 0 & for\ v \leq -4 \end{cases} \qquad (8)$$

The figure 6 shows the RTL hardware circuit of the activation function, it gives an idea about the complexity and calculation that needed in the implementation of this function.

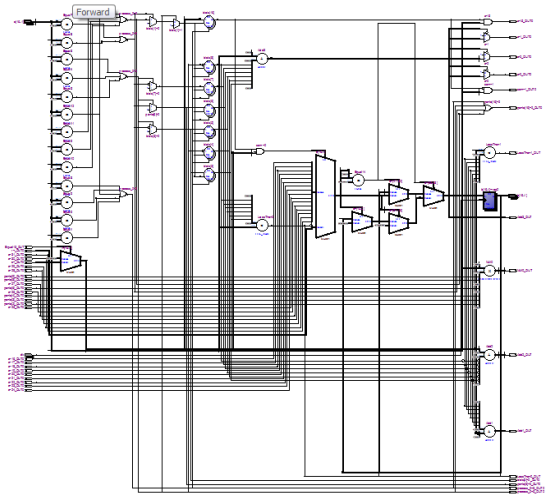

Fig.6 RTL circuit of the sigmoid

V. RESULTS

A. Training Results

The database of signature images used in this work contains 120 images of 2 different persons; these images are digitized by fixed resolution 20x20.

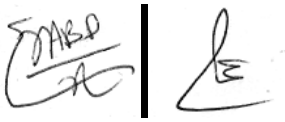

Fig4: samples of images from data-base

The first table shows how we have used these data in both the learning and the test phases.

TABLE I

DATA-BASE USED IN LEARNING AND TESTING

Person	Learning phase	Testing phase
A	40	20
B	40	20

We note that the number of neurones in the input layer is fixed by the number of features extracted which is 43, and we use 1 neurone in the output layer.

The second table the desired outputs during the learning phase.

TABLE III, DESIRED OUTPUTS IN LEARNING PHASE

Output	Person
0	A
1	B

In order to determine the most convenient architecture (number of hidden layers and the number of neurones in each layer), an intuitive method has been used. This method consists in testing experimentally several architectures, with different hidden layers size.

The 3th table summarizes the results obtained for these different architectures.

TABLE IIIII, SIMULATION RESULTS OF DEFERENT ARCHITECTURES IN LEARNING AND TESTING PHASES

Architecture	Classification rate in Learning		Classification rate in testing
	A	B	
43-2-1	94	95	74%
43-5-1	98	97	88%
43-6-1	97	98	93%
43-10-1	98	97	89%
43-15-1	93	88	77%

Analysing the results shown in the table above, it is clear that the best results are obtained by the architecture 43-6-1, for which the Classification rate is 0.93.

B. Implementation Results

In this part the architecture of feed-forward neural network used (43-6-1) layers; the network is composed of 43 inputs, the hidden layer [7] with 6 sigmoid neurons and the output layer with 1 sigmoid neuron. The network uses the parallelism and the rapidity of the FPGA to perform a parallel processing of the data.

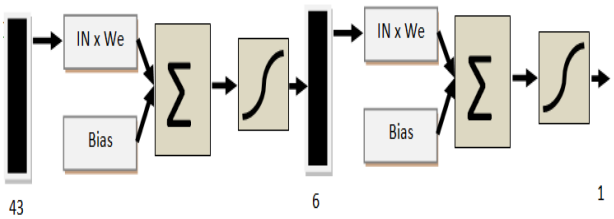

Fig.7: The global network

For the neurons of the hidden layer the product of signed inputs and signed weights form an accumulated result to which we applied the activation function, the final outputs of each neuron (6 outputs) are used as inputs of the next layer (the output layer),

the same process is repeated and the final result is the output of the network that indicates whether or not the handwritten signature belong to the group.

The RTL design of the circuit is shown in the figure 9.

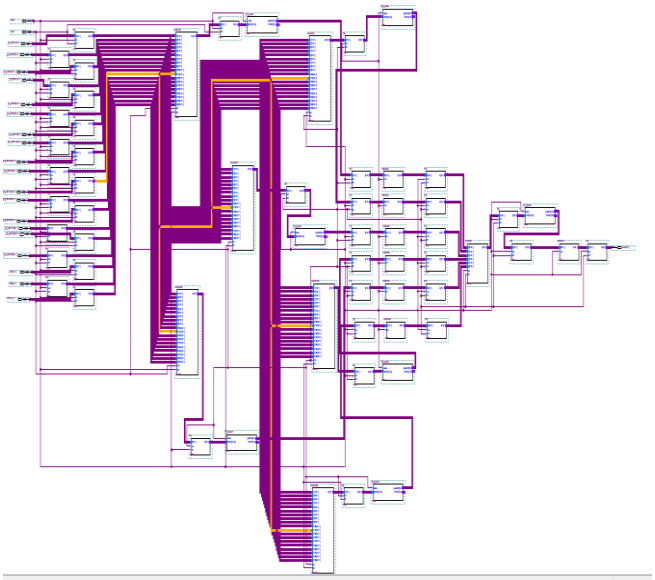

Fig.9: The RTL description of the network

The figure 6 summarize the characteristics of the designed circuit including resource use and performance, we can see that our design occupied 39% of the chip and used only 5% of the embedded Multiplier, this results shows the advantages of using FPGA in the implementation of ANN.

Flow Summary	
Flow Status	Successful - Fri Jan 03 20:00:32 2014
Quartus II 32-bit Version	11.1 Build 216 11/23/2011 SP 1 SJ Web Edition
Revision Name	r1052
Top-level Entity Name	r1052
Family	Cyclone II
Device	EP2C70F896C6
Timing Models	Final
Total logic elements	26,835 / 68,416 (39 %)
Total combinational functions	25,327 / 68,416 (37 %)
Dedicated logic registers	4,271 / 68,416 (6 %)
Total registers	4271
Total pins	322 / 622 (52 %)
Total virtual pins	0
Total memory bits	0 / 1,152,000 (0 %)
Embedded Multiplier 9-bit elements	14 / 300 (5 %)
Total PLLs	0 / 4 (0 %)

Fig.8: The results of the designed circuit

To validate our design we made a simulation by using SIMULINK, considering the processing of the neural network. We feed the network with two vectors, one of a signature that belong to the trained set and on that not, the simulation results of the overall network behavior are shown in figure 10.

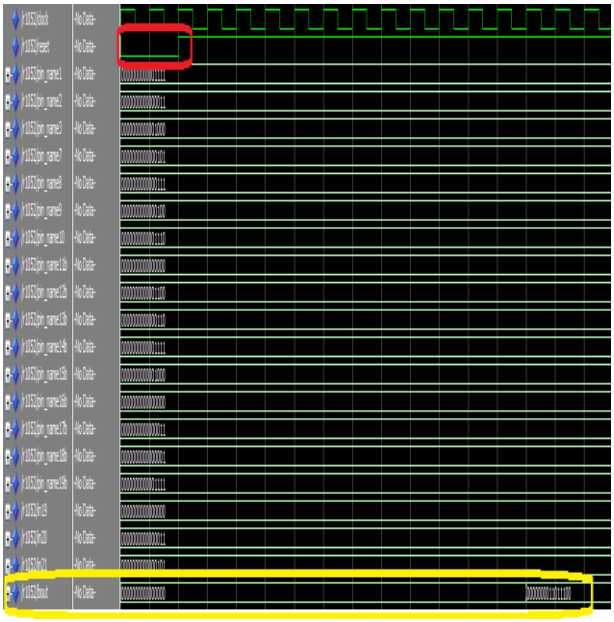

Fig.10 The simulation

VI. CONCLUSIONS

This paper has presented a design solution of neural networks by FPGAs. We implemented a single neuron and proposed a solution for connecting neurons into a multilayer feed-forward BP neural network. The proposed network architecture is modular, being possible to easily increase or decrease the number of neurons as well as layers.

An important obstacle in this work was the hardware implementation for the approximation of sigmoid activation function.

The aim of this work is to be able to use the weights calculated in a software environment (MATLAB) directly in a hardware solution implemented in FPGA.

The result of implementation shows that we still have enough resources for implementing other circuit like the image processing algorithm used to extract features from the handwritten images.

REFERENCES

1. http://www.altera.com
2. M.Gopal, Digital Control and Static Variable Methods. Tata McGraw Hill,New Delhi, 1997.
3. Y. Safi and A. Bouroumi, Prediction of Forest Fires Using Artificial Neural Networks, Applied Mathematical Sciences, vol. 7, no. 6, pp. 271-286, 2013
4. K Tsagkaris, A Katidiotis, P Demestichas, Neural network-based learning schemes for cognitive radio systems. Comp. Commu.31(14),3394–3404(2008)
5. M.Kavianifar and A.Amin, "Preprocessing and Structural Feature Extraction for a Multi-Fonts Arabic / Persian OCR," New South Wales university, Sydney, Australia, 2000.
6. Alin Tisan, Stefen Oniga, Daniel MIC, Attila Buchman, “Digital Implementation of the Sigmoid Function for FPGA Circuits”, ACTA Technica NAPOCENSIS Electronics and Telecommunication, vol.50, no.2, 2009.
7. Sathish Kumar, Neural Networks: A Classroom Approach. Tata McGraw-Hill Publishing Company Limited, New Delhi, 2004.