Computer and Information Science; Vol. 8, No. 1; 2015
ISSN 1913-8989 E-ISSN 1913-8997
Published by Canadian Center of Science and Education

FPGA-Based Fully Parallel PCA-ANN for Spectrum Sensing

A. Elrharras[1,2], S. El Moukhlis[2], R. Saadane[1], M. Wahbi[1] & A. Hamdoun[2]

[1] Engineering system Laboratory, SIRC/LaGeS-Hassania School of Public Works BP 8108 Oasis-Casablanca, Morocco

[2] Information Processing Laboratory, Ben M'sik Faculty of Sciences Hassan II Casablanca University, BP.7955 Sidi Othmane Casablanca, 20702, Morocco

Correspondence: A. Elrharras, Hassania School of Public Works BP 8108 Oasis-Casablanca, Morocco. Tel: 212-663-665-979. E-mail: a.elrharras@gmail.com

Received: January 3, 2015 Accepted: January 6, 2015 Online Published: January 28, 2015

doi:10.5539/cis.v8n1p108 URL: http://dx.doi.org/10.5539/cis.v8n1p108

Abstract

The cognitive radio system is proposed as an optimal way to improve the frequency underutilization. Spectrum sensing is the first and the essential function in this approach. A cognitive user must sense his environment to detect the unused channels, and then he can use the free channel without causing any interference to the primary user. In this article, an innovative technique is proposed for spectrum sensing based on principal component analysis and neural networks in frequency domain. The designed blocks are described using VHSIC Hardware Description Language (VHDL). The suggested application consists of extracting features from the captured signals by PCA; the classification is done by a Multi-Layer Perceptron (MLP). Neural network training part and principal components are done on MATLAB environment; while the hardware implementations are created on an FPGA DE2-70board.

Keywords: cognitive radio, spectrum sensing, features extraction, FFT, PCA, ANN, MLP, VHDL, FPGA

1. Introduction

The frequency spectrum use is becoming increasingly demanded(Datla, Rajbanshi, Wyglinski, & Minden, 2009), as a result of the rising needs of wireless technologies and their various services. The current management of the frequency spectrum is a static allocation which cannot support these growing needs. Within this context, the approach of cognitive radio was introduced by Mitola to end those problems and increase the spectrum efficiency. In this way, the spectrum management will shift from the classic appearance to a dynamic use (Mitola & Maguire Jr, 1999) and (Zeng, Liang, Hoang, & Zhang, 2010). The dynamism given to this management is based on separating users into two categories:

PU: Primary User; that has a license to benefit from a given spectral band.

SU: Secondary User; that does not have a license, but can temporarily profit from a free band.

Thus under the condition, the PU should not suffer harmful external interference, and does not make any modifications to allow coexistence with the SU.

Several methods for spectrum sensing are suggested in the literature, such as Matched filter (Fuchs, 2009), Cyclostationary test (Ning, Sohn, & Kim, 2009), and Energy Detection (Yucek & Arslan, 2009); but the energy detection remains the most chosen, not only for its simple implementation, but since it does not need to know any information in advance about the PU. In this method, the received signal energy is measured, and compared to a predetermined threshold, which presents the noise power of present on the channel, if the energy exceeds the threshold, we declare the PU presence, otherwise it is absent.

The uncertainty of noise, shadowing and the channel fading are problems that decrease the performance of an energy detector (Tandra & Sahai, 2005). In our context, we propose a hybrid architecture for spectrum sensing, which consists of three blocks FFT, PCA, and ANN. MATLAB is used to calculate PCA eigenvectors, besides training the neural network, with the intention of calculating weights and bias. The hardware implementation illustrates the scheme of a numerical system with three stages achieving FFT block with 1024 points, a matrix multiplication block to change the signal basis by PCA, and a feed-forward multilayer neural network to make the decision; the hole process is realized in Altera Cyclone II FPGA integrated inside the DE2-70 Board

(TechnologiesTerasic).

The second section presents the adopted methodology and the used materials, then the third section deals with the different parts of implementation. In the fourth section, the implementation results are discussed. Finally, the fifth section concludes the paper.

2. Methodology and Materials

2.1 Mathematical Formulation of Spectrum Sensing

In this context, we shall detect the presence of the PU that emits at a given time a signal $\mathrm{x(t)}$. The problem of the spectrum sensing can be formulated as follows:

$$\begin{cases} \mathrm{y(t) = n(t)} & : \ \mathrm{H_0} \\ \mathrm{y(t) = \varepsilon * x(t) + n(t)} & : \ \mathrm{H_1} \end{cases} \quad \text{with} \quad 0 < \varepsilon \leq 1 \tag{1}$$

$\mathrm{y(t)}$: The received signal. $\mathrm{x(t)}$: The signal transmitted currently being deterministic or random, but unknown. $\mathrm{n(t)}$: Noise and it is supposed to be additive white Gaussian.

The hypothesis $\mathrm{H_0}$ presents the case where the received signal $\mathrm{y\,(t)}$contains the noise alone. While the hypothesis $\mathrm{H_1}$, refers to the ability of detecting the PU presence.

The performance of the detector is characterized by using the pair ($\mathrm{P_d}$, $\mathrm{P_{fa}}$) (Atapattu, Tellambura, & Jiang, 2010).

$\mathrm{P_d}$: probability of detection $\mathrm{P_{fa}}$: probability of false alarm.

$$P_d = P(decision\ of\ \mathrm{H_1}/\ \mathrm{H_1}) \tag{2}$$

$$P_{fa} = P(decision\ of\ \mathrm{H_1}/\ \mathrm{H_0}) \tag{3}$$

2.2 Dimensionality Reduction by PCA

The Principal Component Analysis (PCA) is commonly used in data exploration. When we have a quantitative database, wherein ‘n’ observations (the captured signals) are described by ‘m’ variables (samples), where ‘m’ is considerably high. It is impossible to understand the data structure and the proximity between the observations by a simple analysis or even a correlation matrix. The PCA can be considered a projection method, which projects the observations from the N-dimensional space to a K-dimensional space ($K \ll N$) (Burges, 2010), such that the maximum of information is retained on the K-first dimensions.

The purpose of PCA is condensing the original data into a new group, so that they have no correlation between them, and are ordered in terms of the percentage of variance contributed by each component. Thus, the first principal component includes the maximum variance, while the second principal component covers following variance, and the process is repeated until the last principal component. In this way, the information loss decreases from one step to the next.

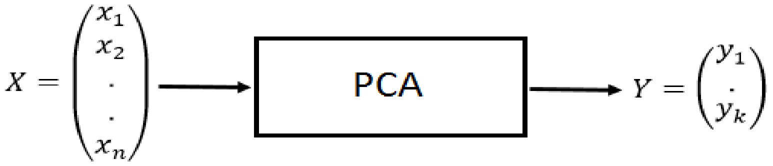
$X = \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{pmatrix}$
PCA
$Y = \begin{pmatrix} y_1 \\ . \\ y_k \end{pmatrix}$

Figure 1. Dimensionality reduction by PCA

2.3 Artificial Neural Networks

Hardware neural network systems (called neuro-hardware) have been developed rapidly over the last decade. Unlike the Von Neumann architecture - Classic -this is sequential in nature, artificial neural networks (ANN) benefit from the massively parallel processing. A variety of materials are designed to exploit the parallelism inherent in the neural network models. Despite the power growth of the numerical calculation in general processors, the approach of neuro-hardware is considered promising in specialized applications such as image processing, voice synthesis, pattern recognition…

2.3.1 Formal Neuron

Studying artificial neural network was inspired mainly from the biological learning system(Basheer & Hajmeer, 2000); the biological model is composed of complex layers of interconnected neurons. In effect, the human brain is composed of approximately 10^{11} neurons, each one have an average of 10^{3} connections. It is believed that the considerable calculation power of the brain is a result of the parallel and distributed processing performed by these neurons.

The artificial neuron has generally multiple inputs and a single output. Actions of excitatory synapses are illustrated by the coefficients called synaptic weights; these weights are coupled with all inputs. The numerical values of these coefficients are adjusted in the learning phase.

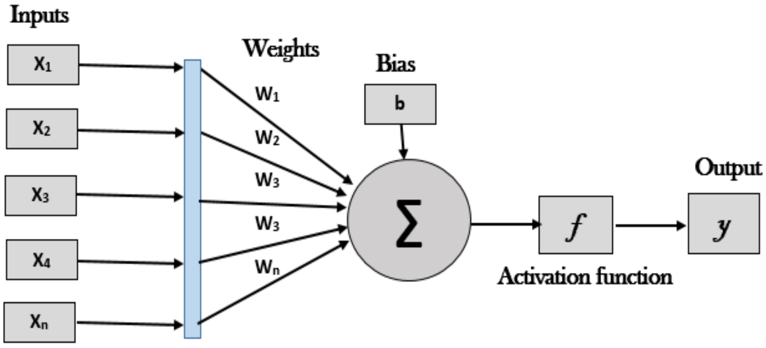

Figure 2. The formal neuron architecture

The second figure illustrates the artificial neuron design, the formal neuron receives n inputs presented by the vector$\{X_1, X_2 \ldots X_n\}$, these inputs are assigned by their weights indicated such as $\{W_1, W_2 \ldots W_n\}$, and the b presents the bias activation. Equation (4) gives the output of this model.

$$y = f(\sum W_{i} * X_i + b) \tag{4}$$

The activation function has a crucial role in the formal neuron architecture. In this paper, a sigmoid function is utilized the since its nonlinearity makes it possible to approximate any function.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{5}$$

2.3.2 Multi-Layer Perceptron

A Multi-Layer Perceptron (MLP) consists of a number of artificial neurons interconnected, this network is organized in the form of layers (Huang, 2003), such as the layer $'i'$receive as input the outputs of the previous layer $'i-1'$ and feed its outputs to the next layer $'i+1'$. This model is called a direct neural network or a feed forward neural network. The first and last layers are called, respectively, the input and output layers. Layers that are neither input nor output are known as hidden layers.

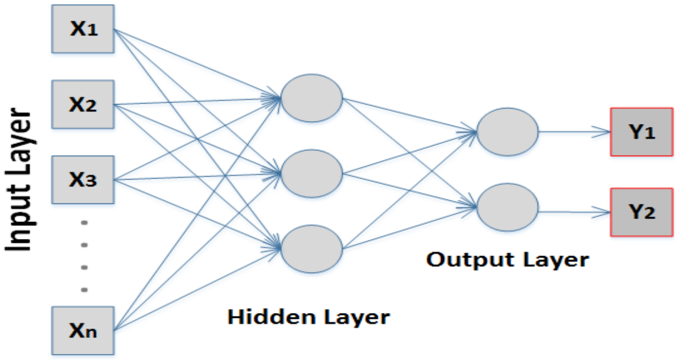

Figure 3. MLP model

The MLP model uses essentially the back-propagation algorithm for the training phase(Shao & Zheng, 2011), this algorithm consist in minimizing the global error E_g which calculated by the relation below.

$y_{d,i}(t)$: Is the desired output for the ith neuron, and $y_{m,i}(t)$ is the measured output for the same neuron

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

2.4 Proposed Design

The design adopted in this paper is guided by the principle to realize an efficient detector based on neural networks, which will be able to determine whether the band being in the test is free or occupied, not only with a satisfactory precision, but also with a minimal number of inputs to the model to save the training time and minimize the area of implementation. The proposed design consists of three stages:

- FFT block to transform captured signals from time to frequency domain.
- Dimensionality reduction with PCA.
- Decision is done by a MLP neural network with two layers in which every layer is completely linked with its closest layers.

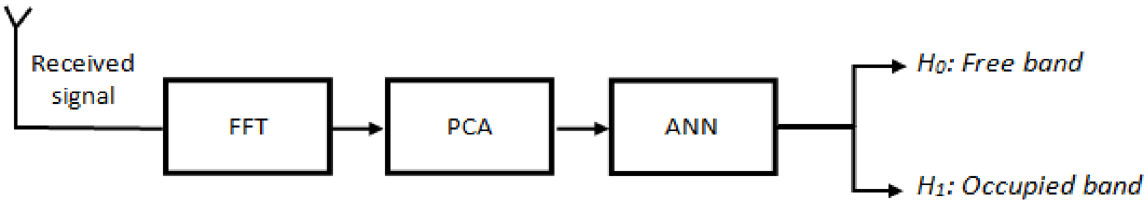

Figure 4. Proposed architecture

2.5 Measurement Platform and Database

The measurement platform employed in this paper consists of two parts. In the first one, the generation of a signal, where we can choose the type of modulation: AM, FM or FSK, by the Agilent 33250A Function / Arbitrary Waveform Generator (*Agilent 33250A User's Guide Ref 33250-90002*, March 2003). In the second, the numerical oscilloscope FI3104 (*InfiniiVision 3000 X-Series Oscilloscopes Datasheet Ref 5990-6619EN*, April 11, 2014) is used to receive signals that are captured by the antenna, the latter is connected to a Personal Computer (PC) with Universal Serial Bus 2 (USB2) interface. The figure below gives a view about the measurement platform used in this experience.

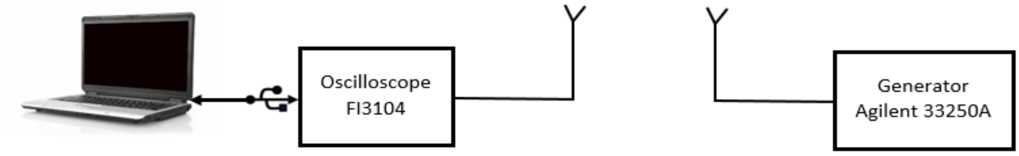

Figure 5. Measurement platform

The signals database used in our work has been collected exploiting the previous platform. It contains 500 signals for different types of modulation (AM, FM and FSK) and 200 signals that include only noise. The following table shows data used in both the learning and testing phases.

Table 1. Data-base used in learning and testing

Signal	Learning phase	Testing phase
Primary signal	400	100
Noise	100	100

2.6 Implementation Platform

The detection system is implemented on an FPGA device, as it offers a high flexibility. VHSIC Hardware Description Language (VHDL) is used to synthesize the design, with Altera Quartus II Web Edition as the place and-route tool. The target device is the Altera DE2-70. The following figure gives more details about the target board.

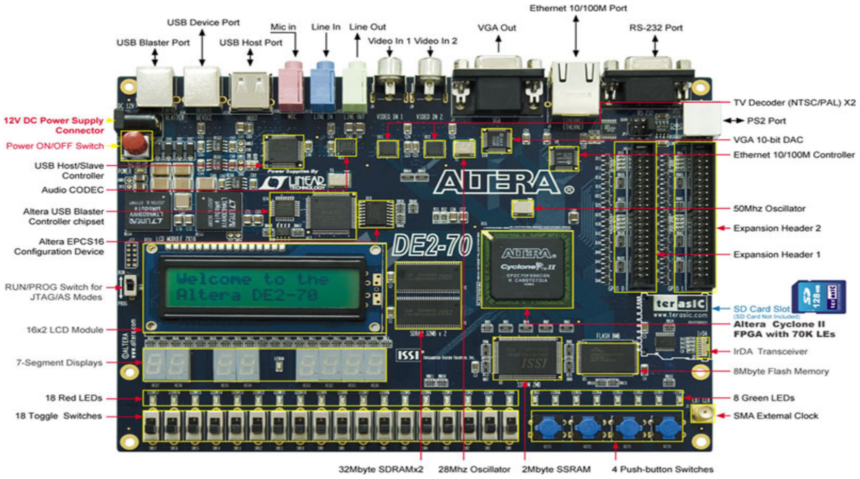

Figure 6. DE2-70 FPGA from Altera

3. Implementation

3.1 FFT Implementation

As a first block of our proposed architecture, we use the *FFT MegaCore* function which is an IP Core for FPGA from Altera (*FFT MegaCore Function User Guide*, August 2014), this function is able to calculate both forward and inverse Fourier transform. This hardware architecture is chosen to optimize the usage of resources, and has the four following parameters: Burst mode, Single output engine, Single instance of engine, 16-bit internal and data input/output precision widths.

3.2 PCA Implementation

The aim of using PCA in this work is to make possible the hardware implementation of the neural network detector. In this work, each captured signal contains 1024 samples. If we use the signal vector directly as an input of the neural network, there would be 1024 input neurons, which would form a very large network in terms of neurons number in the input and hidden layers, also it would increase the computational complexity (registers, multipliers, adders). So the architecture of the detector would be high in terms of hardware cost. In this respect, PCA is used to reduce ANN complexity; but we have to take into account that the classification rate must be kept at a satisfactory level.

The PCA training part, which concerns the eigenvectors matrix calculation, was developed in MATLAB. The hardware implementation has been restricted to features extraction following these two equations (8) and (11).

$$X' = \frac{(X-\bar{X})}{\sigma} Such\ as \bar{X} = \frac{1}{N}\sum_{i=1}^{n} X_i \qquad and \qquad \sigma_j = \frac{1}{N}\sum_{i=1}^{n}(x_{i,j} - \bar{x}_{i,j}) \tag{8}$$

X : Is the signal vector, while $\bar{X}$ is the vector of variable means, and then σ present the variance of variables.

$$X_r = P^T.X' \qquad Such\ as \qquad P = [u_1, \ldots, u_k] \tag{9}$$

$\mathrm{X_r}$: Is the reduced vector, and P is the new basis$\{\mathrm{u}_1, \ldots, \mathrm{u}_k\}$ of the K-largest eigenvalues.

As described in the figure below, the architecture of PCA consists in three blocks: the first stage receives the signal vector already converted in the frequency space, then it subtracts the vector of variable means $\bar{X}$ from the signal vector X, the $\bar{X}$ vector has been stored in a Read-Only Memory (ROM). The second block is used to devise the first block output by the variance σ given by the equation (8). The third and final stage was used to project the signal into the reduced space using the equation (9).

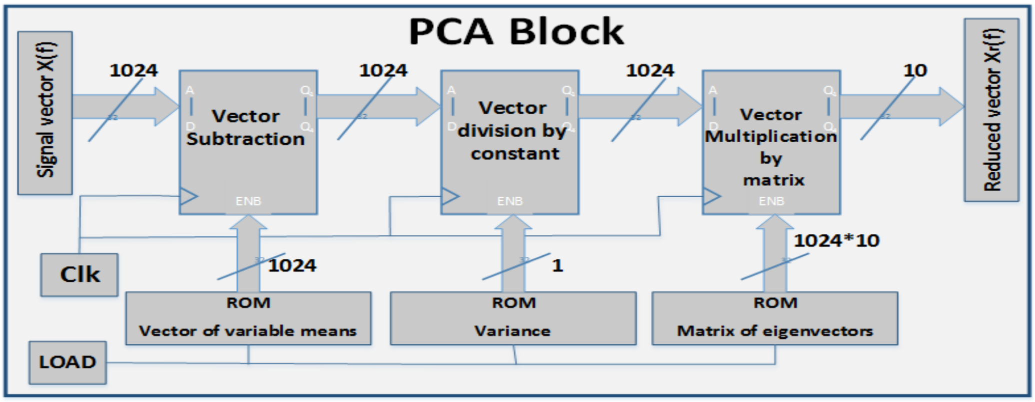

Figure 7. PCA block architecture

3.3 Architecture of the Neuron

The hardware implementation of the decisional phase will be restricted in the trained neural network; the training phase has been done in MATLAB environment. As specified above, this implementation has been developed by VHDL language, and the design has been created in Quartus II environment. Figure 8 explains the block diagram of a neuron, it is consisted of a multiplier block which multiplies the weights by all parallel inputs, then an accumulator which gives the addition result of all the multiplications plus the bias, and finally the score is taken from the output of the activation function.

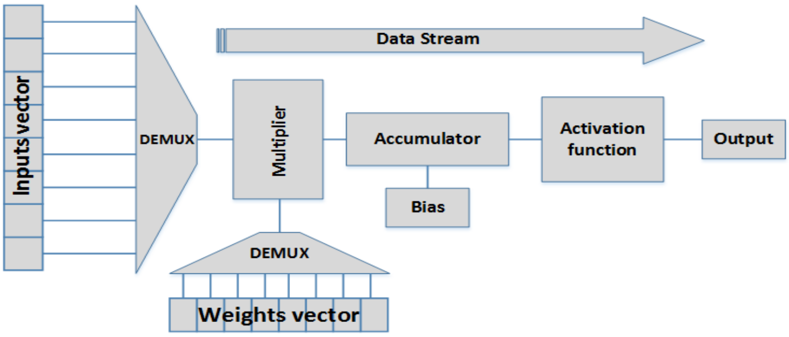

Figure 8. Block diagram of a neuron

3.4 Activation Function

As cited in the section (2.3.1) the activation function has a critical impact in the neuron design, in the Multi-Layer perceptron model the sigmoid function is the most used. The sigmoid function consists of an infinite exponential series, which makes it inappropriate for a direct implementation on FPGA. To bypass this problem, we use the Piece-Wise Linear Approximation (Campo, Finker, Echanobe, & Basterretxea, 2013) , in that approximation the 'logsig' function presented by the equation 10, is divided into five intervals to consider it as linear function in each interval, as shown in the equation 11.

$$\theta(v) = \frac{1}{(1+e^{-v})} \tag{10}$$

$$\theta(v) = \begin{cases} 1 & for\ v \geq 4 \\ 0.0625v + 0.75 & for\ 0 < v < 4 \\ 0.5 & for\ v = 0 \\ 0.0625v + 0.25 & for - 4 \leq v < 0 \\ 0 & for\ v \leq -4 \end{cases} \tag{11}$$

Figure 9 illustrates the approximated sigmoid hardware design; this architecture provides an insight about the complexity degree for a hardware implementation of such function.

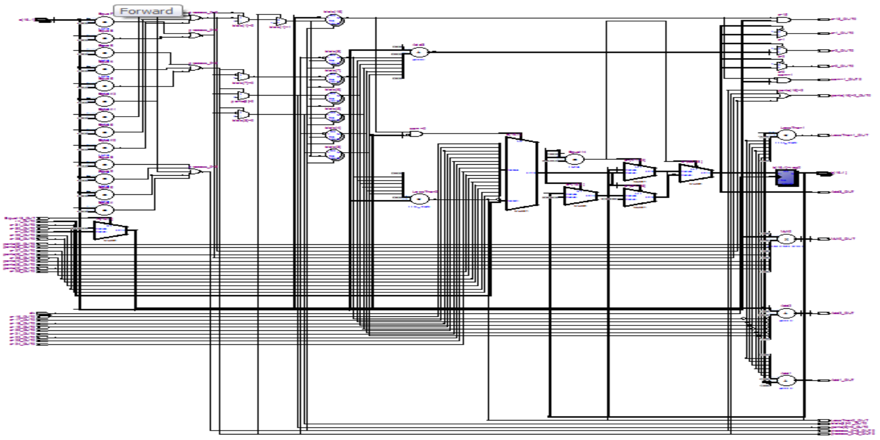

Figure 9. The approximated "Logsig" circuit

4. Results and Discussion

4.1 PCA Results

The PCA training is realized using MATLAB environment, the following graph gives the percentage of the first 35 eigenvalues. From these results, we can see that the first 10 components with a percentage of eigenvalue superior than 2%, together explain 98% of the total data-base variance. The first principal component describes the greater percentage 12.1%, while the second one illustrates about 10.6% of the total variance.

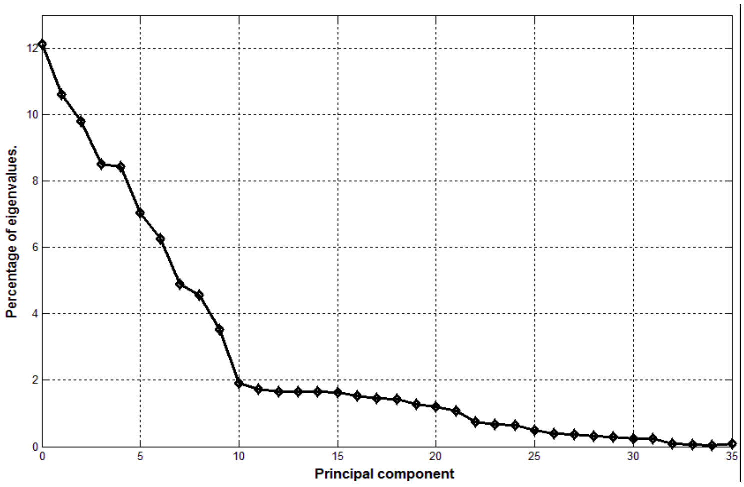

Figure 10. Percentage of the first 35 eigenvalues

4.2 Neural Network Training Results

Knowing that the PCA block extracts 10 features, so in the input layer we use also 10 neurons, and in the output layer we use one neuron to differentiate between H_0 and H_1.

Table 2. Desired outputs in the learning step

Hypothesis	Signal used	Output
H_0	Only noise	0
H_1	Primary signal plus noise	1

Each additional neuron in the hidden layer is used to take into account specific profiles of the input neurons. A larger number makes it possible to better match the data presented, but decreases the capacity of generalization in the network. Here again there is no general rule, but empirical rules (Zhang, Patuwo, & Hu, 2001). The size of the hidden layer must be:

-Equal to that of the input layer.

-Equal to 75% of the input layer.

-Equal to the square root of the product between the number of neurons in the input and output layer.

Should be noted that the last choice reduces the number of freedom degrees left to the network, then the ability to adapt on the training samples in favor of a greater stability / ability to generalize.

In this work, we adopt a statistical approach, it consist in trying different architectures with different numbers of neurons in the hidden layer. The classification rate is calculated for these architectures. The following table summarizes the results of this statistical study:

Table 3. Learning and testing results for different hidden layer sizes

Hidden layer size	Classification rate in Learning		Classification rate in testing
	H_0	H_1	
5	98	97	80%
7	98	99	94%
10	100	99	98%
12	97	100	92%
15	98	100	88%
20	100	98	96%

Analyzing the results shown in the table above, it is obvious that the high classification rate (98%) in testing is achieved by using 10 neurons in the hidden layer.

4.3 Neural Network Implementation Results

The implemented network architecture consists in using 10 neurons at the input layer. Thereafter, 10 neurons are used in the hidden layer, while we exploit one neuron in the output layer to differentiate between the two hypotheses H_0 and H_1. The following figure shows the RTL design of the full network circuit:

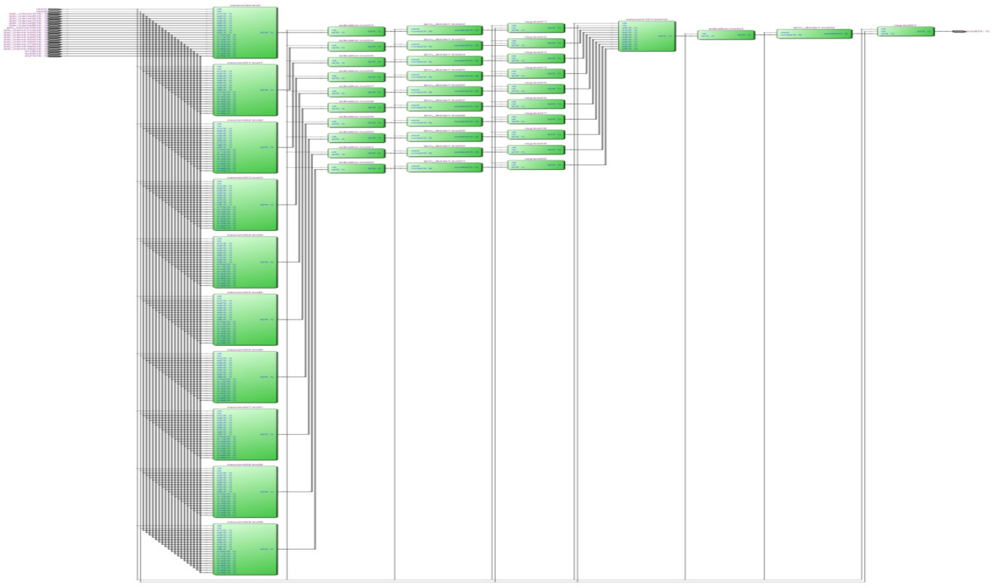

Figure 11. The RTL description of the network circuit

4.4 Full Detector Implementation Results

The subsequent diagram explains the manner in which we connect the two designed blocks (PCA and ANN), with the *FFT MegaCore* IP Core. At the beginning of this architecture, we receive the collected samples. The treatment begins by transforming the signal into frequency domain using the FFT IP core. Next, the PCA block extracts the features, this block takes $X(f)$ with 1024 parameters and sets free $X_r(f)$ with only 10 elements. Finally, the reduced vector is presented to the ANN block to make the decision about the channel stat.

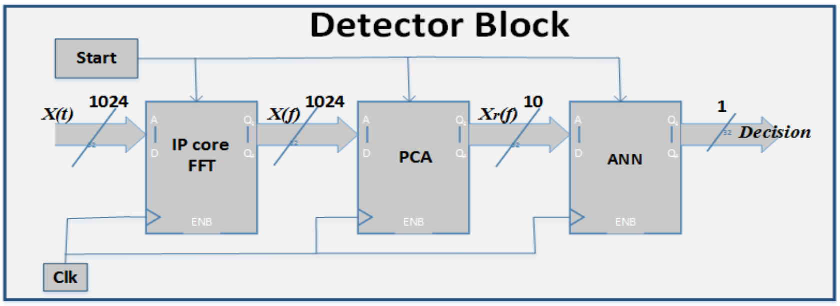

Figure 12. Full detector block architecture

The 4th table summarizes the proposed design characteristics, including the utilization of different resources; we notice that the proposed architecture takes 46% from the total logic elements in the FPGA, while it spends 70% from the embedded 9-bit Multipliers.

Table 4. The designed circuit results

Recourse	available	Used	Percentage
Total logic elements	68,416	31,410	46%
Total pins	622	322	52%
Embedded Multiplier 9-bits	300	210	70%

5. Conclusions

In this work, a new method of spectrum sensing based on PCA and ANN is designed and implemented on an FPGA device. A fully parallel MLP network is trained with a back-propagation algorithm, and the detector error is at a suitable level (2%). The numerical precision of the proposed detector is reasonable, and the inaccuracy of the ANN is very low. The sigmoid activation function is the most essential block of the neuron; it was realized in hardware using the Piece-Wise Linear Approximation with floating-point representation, and the results show that the activation block consumes the major part of the hardware resources. The final proposed network architecture is modular, making it possible to change the number of neurons and layers in a simple manner. In our future work, we will explore ways to integrate the learning phase for both PCA and ANN in the same design. To do this, we have to reduce the FPGA resource usage, while maintaining the performance of detector at an acceptable level.

References

Atapattu, S., Tellambura, C., & Jiang, H. (2010). Analysis of area under the ROC curve of energy detection. *Wireless Communications, IEEE Transactions on, 9*(3), 1216-1225. http://dx.doi.org/10.1109/TWC.2010.03.091085

Basheer, I., & Hajmeer, M. (2000). Artificial neural networks: fundamentals, computing, design, and application. *Journal of microbiological methods, 43*(1), 3-31. http://dx.doi.org/doi:10.1016/S0167-7012(00)00201-3

Burges, C. J. (2010). Geometric methods for feature extraction and dimensional reduction-a guided tour. *Data mining and knowledge discovery handbook*, 53-82. http://dx.doi.org/10.1007/978-0-387-09823-4_4

Campo, I., Finker, R., Echanobe, J., & Basterretxea, K. (2013). Controlled accuracy approximation of sigmoid function for efficient FPGA-based implementation of artificial neurons. *Electronics Letters, 49*(25), 1598-1600. http://dx.doi.org/10.1049/el.2013.3098

Datla, D., Rajbanshi, R., Wyglinski, A. M., & Minden, G. J. (2009). An adaptive spectrum sensing architecture for dynamic spectrum access networks. *Wireless Communications, IEEE Transactions on, 8*(8), 4211-4219. http://dx.doi.org/10.1109/TWC.2009.080989

Fuchs, J. J. (2009). *Identification of real sinusoids in noise, the Global Matched Filter approach.* Paper presented at the 15th Ifac-Ifors symposium on Identification and system Parameter Estimation. http://dx.doi.org/10.3182/20090706-3-fr-2004.00187

InfiniiVision 3000 X-Series Oscilloscopes Datasheet Ref 5990-6619EN. (April 11, 2014). Retrieved from: http://cp.literature.agilent.com/litweb/pdf/5990-6619EN.pdf

Mitola, J., & Maguire Jr, G. Q. (1999). Cognitive radio: making software radios more personal. *Personal Communications, IEEE, 6*(4), 13-18. http://dx.doi.org/10.1109/98.788210

Ning, H., Sohn, S. H., & Kim, J. M. (2009). A blind OFDM detection and identification method based on cyclostationarity for cognitive radio application. *IEICE transactions on communications, 92*(6), 2235-2238. http://dx.doi.org/10.1587/transcom.e92.b.2235

Shao, H., & Zheng, G. (2011). Convergence analysis of a back-propagation algorithm with adaptive momentum. *Neurocomputing, 74*(5), 749-752. http://dx.doi.org/10.1016/j.neucom.2010.10.008

Tandra, R., & Sahai, A. (2005). *Fundamental limits on detection in low SNR under noise uncertainty.* Paper presented at the Wireless Networks, Communications and Mobile Computing, 2005 International Conference on. http://dx.doi.org/10.1109/WIRLES.2005.1549453

Yucek, T., & Arslan, H. (2009). A survey of spectrum sensing algorithms for cognitive radio applications. *Communications Surveys & Tutorials, IEEE, 11*(1), 116-130. http://dx.doi.org/10.1109/SURV.2009.090109

Zeng, Y., Liang, Y. C., Hoang, A. T., & Zhang, R. (2010). A review on spectrum sensing for cognitive radio: challenges and solutions. *EURASIP Journal on Advances in Signal Processing, 2010*, 2.

http://dx.doi.org/10.1155/2010/381465

Zhang, G. P., Patuwo, B. E., & Hu, M. Y. (2001). A simulation study of artificial neural networks for nonlinear time-series forecasting. *Computers & Operations Research, 28*(4), 381-396. http://dx.doi.org/10.1016/s0305-0548(99)00123-9

Technologies Terasic. Altera DE2-70 Board. Retrieved from http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=226

Agilent 33250A User's Guide Ref 33250-90002. (March 2003). Retrieved from http://cp.literature.agilent.com/litweb/pdf/33250-90002.pdf

FFT MegaCore Function User Guide. (August 2014). Retrieved from http://www.altera.com/literature/ug/ug_fft.pdf

Copyrights

Copyright for this article is retained by the author(s), with first publication rights granted to the journal.

This is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/3.0/).