TJES
ISSN: 1813-162X
Tikrit Journal of Engineering Sciences
available online at: http://www.tj-es.com

Hardware Implementation of Artificial Neural Network for Data Ciphering

Sahar L. Kadoory[1] , Toka A. Fatehi[2] , Qutaiba A. Hasan[3]
[1] *Electronics Engineering College, University of Mosul, Mosul, Iraq*
E-mail: saharlazem@yahoo.com
[2] *Electronics Engineering College, University of Mosul, Mosul, Iraq*
E-mail: toka_engineer@yahoo.com
[3] *Petroleum and Minerals Engineering College, Tikrit University, Salahaldeen, Iraq*
E-mail: qutaibaeng@yahoo.com

Abstract

This paper introduces the design and realization of multiple blocks ciphering techniques on the FPGA (Field Programmable Gate Arrays). A back propagation neural networks have been built for substitution, permutation and XOR blocks ciphering using Neural Network Toolbox in MATLAB program. They are trained to encrypt the data, after obtaining the suitable weights, biases, activation function and layout. Afterward, they are described using VHDL and implemented using Xilinx Spartan-3E FPGA using two approaches: serial and parallel versions. The simulation results obtained with Xilinx ISE 9.2i software. The numerical precision is chosen carefully when implementing the Neural Network on FPGA. Obtained results from the hardware designs show accurate numeric values to cipher the data. As expected, the synthesis results indicate that the serial version requires less area resources than the parallel version. As, the data throughput in parallel version is higher than the serial version in rang between (1.13-1.5) times. Also, a slight difference can be observed in the maximum frequency.

Keywords: Back propagation, Ciphering, Encryption, FPGA, Neural Network, VHDL.

التنفيذ المادي للشبكة العصبية الاصطناعية المستخدمة لاغراض تشفير البيانات

الخلاصة

يقدم هذا البحث التصميم و التنفيذ المادي لعدد من تقنيات التشفير متعدد الكتل على مصفوفة البوابات المبرمجة حقليا. لقد تم بناء شبكات عصبية و تدريبها بطريقة الانتشار العكسي لكل من التشفير الابدالي و التعويضي و التشفير بطريقة XOR باستخدم صندوق العدة للشبكات العصبية المتوفرة في برنامج ال MATLAB. دربت هذه الشبكات لتشفير البيانات بعد الحصول على الاوزان والتخطيط و دوال التفعيل المناسبة. تم استخدام الشبكات المدربة لبناء كيان مادي بواسطة لغة وصف الكيان المادي VHDL و طبقت على اداة مصفوفة البوابات المبرمجة حقليا ال Spartan-3E باستخدام منهجيتين هما المتسلسلة والمتوازية. تم الحصول على نتائج المحاكاة بواسطة برنامج ISE 9.2i. انّ الدقّة العددية استخدمت بعناية عندما طبّقت الشبكة العصبية على FPGA. حيث ان النتائج المستحصلة من التصميم المادي تظهر قيم عددية دقيقة لتشفير البيانات. كما هو متوقّع، تشير نتائج التمثيل بأنّ المنهجية المتسلسلة تحتاج الى مساحة اقل من المنهجية المتوازية. كما ان إنتاجية البيانات بالمنهجية المتوازية أعلى من المنهجية المتسلسلة بمقدار (1.5-1.13) مرة. كما ويلاحظ وجود إختلاف طفيف في التردد الأقصى بين المنهجيتين.

الكلمات الدالة: الانتشار العكسي، تشفير، مصفوفة البوابات المبرمجة حقليا، الشبكات العصبية، لغة وصف الكيانات المادية.

Introduction

Artificial Neural Network can be defined as “a parallel, distributed information processing structure consisting of processing elements (which can possess local memory and can carry out localized information processing operations) interconnected via unidirectional signal channels called connections”. ANN becomes more and more popular in the Artificial Intelligence (AI) field these decades since it can be used for security network, image processing, pattern recognition, fingerprint analysis and many other problems’ solving [1].

The problem of protecting information has existed since information has been managed. However, as technology advances and information management systems become more and more powerful, the problem of enforcing information security also becomes more critical. The massive use of the communication networks for various purposes in the past few years has posed new serious security threats and increased the potential damage that violations may cause. A violation to the security of the information may jeopardize the whole system working and cause serious damages. Advances in artificial neural networks (ANNs) provide effective solutions to this problem [2].

Nowadays, feed forward ANN has the widest and most comprehensive application among all the ANN models. Back propagation Neural network (BPNN) is a typical feed forward structure. Nonetheless, there are no definite rules for the choice of how many hidden layers and neurons, no promising convergence and it may take very long time for the iterative training procedures [1].

There are different kinds of electronic implementations of ANN: digital, analog, hybrid, and each one have specific advantages and disadvantages depending on the type and configuration of the network, training method and application. For digital implementations of ANN there are different alternatives: custom design, digital signal processors, and programmable logic. Among them, programmable logic offer low cost, powerful software development tools and true parallel implementations. Field programmable gate arrays (FPGA) are a family of programmable logic devices based on an array of configurable logic blocks (CLB), which give a great flexibility in the development of digital ANNs [3].

The paper focuses on realizing the neural network for multiple blocks ciphering techniques on the FPGA (Field Programmable Gate Arrays). A back propagation neural networks have been trained for substitution, permutation and XOR blocks cipher to encrypt the data, after obtaining the suitable weights, biases, activation function, and layout.

Related Works

Recently, much work has focused on implementing artificial neural networks on reconfigurable computing platforms to solve the security problems. Reconfigurable computing is a means of increasing the processing density (i.e. greater performance per unit of silicon area) above and beyond that provided by general-purpose computing platforms. Field Programmable Gate Arrays (FPGAs) are a medium that can be used for reconfigurable computing and offer flexibility in design like software but with performance speeds closer to Application Specific Integrated Circuits (ASICs) [4]. Some of these works are arranged as follows:

In 2007, Chandra et al. [5], discussed the possibility of employing Neural Networks for identification of Cipher Systems from cipher texts. Cascade Correlation Neural Network and Back Propagation Network have been employed for identification of Cipher Systems. Very large collection of cipher texts were generated using a Block Cipher (Enhanced RC6) and a Stream Cipher (SEAL). Promising results were obtained in terms of accuracy using both the Neural Network models but it was observed that the Cascade Correlation Neural Network Model performed better compared to Back Propagation Network.

In 2008, Sadeghian et al. [6], described an innovative form of cipher design based on the use of recurrent neural networks. The proposed cipher had a relatively simple architecture and, by incorporating neural networks, it released the constraint on the length of the secret key. The design of the symmetric cipher was described in detail and its security was analyzed. The cipher was robust in resisting different cryptanalysis attacks and provides efficient data integrity and authentication services. Simulation results were presented to validate the effectiveness of the proposed cipher design.

In 2009, Abdelhalim et al. [7], approved a modified implementation of Rijndael AES encryption standard based on the fact that any FPGA includes built in memory block; therefore they stored all the results of the fixed operations within the memory modules. The modification gave an 11% reduction in area and 25% increase in speed (Throughput) compared with the original design. Their design gave the highest throughput and area utilization over all the Iterative Looping based FPGA implementations.

In 2010, Sivagurunathan et al.[8], deliberated classical substitution ciphers, namely, Playfair, Vigenère and Hill ciphers. The features of the cipher methods under consideration were extracted and a back-propagation neural network was trained. The network was tested for random texts with random keys of various lengths. The cipher text size was fixed as 1Kb. The results so obtained were encouraging.

In 2011, Siddeeq. Y. Ameen et al.[9], attempted to implement Rijndael AES cryptosystem using ANN. In the design of the NN performs encryption and decryption processes using of symmetric key cipher. The key used in both encryption and decryption processes was the initial weights for neural network and then trained to its final weight with a fast and low cost algorithm, such as Levenberg – Marquardt Algorithm. The final weights of neural network represented the final key that can be used for encryption and decryption processes. Simulation results showed the closeness of the results achieved by the proposed NN-based AES cryptosystem with that of the normal AES.

In 2012, Khaled Alallayah et al.[10], discussed the methods of block Simplified DES (SDES) crypto systems. They constructed a Neuro-Identifier mode to achieve two objectives: the first one is to emulate construction Neuro-model for the target cipher system, while the second is to (cryptanalysis) determine the key from given plaintext-ciphertext pair.

Modern Ciphers

With the advent of the computer, ciphers need to be bit-oriented in addition to character-oriented. This is so because the information to be encrypted is not just text; it can also consist of numbers, graphics, audio, and video data. It is convenient to convert these types of data into a stream of bits, encrypt the stream, and then send the encrypted stream. In addition, when text is treated at the bit level, each character is replaced by 8 (or 16) bits, which means the number of symbols becomes 8 (or 16). A modern symmetric cipher is a combination of simple ciphers. In other words, a modern cipher uses several simple ciphers to achieve its goal. These simple ciphers first will be discussed [11].

XOR Cipher

One of Modern ciphers is called the XOR cipher because it uses the exclusive-or operation as defined in computer science. Figure (1) shows an XOR cipher.

An XOR operation needs two data inputs plaintext, as the first and a key as the second. In other words, one of the inputs is the block to be the encrypted, the other input is a key; the result is the encrypted block. In an XOR cipher, the size of the key, the plaintext, and the cipher text are all the same. XOR ciphers have a very interesting property: the encryption and decryption are the same. This encryption technique is appropriate for binary signals and it encrypts each pixel (bit) individually [12].

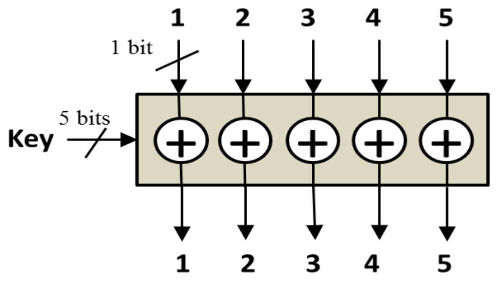

Fig. 1. XOR cipher

Substitution Cipher: S-BOX

The input to an S-box is a stream of bits with length N; the result is another stream of bits with length M. So, N and M are not necessarily the same. Figure (2) shows an Example of S-box.

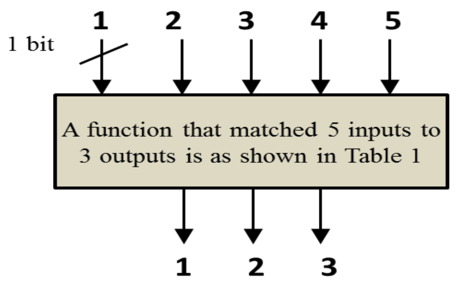
1 bit
1 2 3 4 5
A function that matched 5 inputs to 3 outputs is as shown in Table 1
1 2 3

Fig. 2. An example of S-box

The S-box is normally keyless and is used as an intermediate stage of encryption or decryption. The function that matches the input to the output may be defined mathematically or by a table [11]. Thus S-box which is applied in this paper is the function defined in Table (1) and the output of this S-box is determined as follows:

The first and last bits of input bits represent in base 2 a number in the range 0 to 3. Let that number be i. The middle 3 bits of input bits represent in base 2 a number in the range 0 to 8. Let that number be j. Look up in Table 1 the number in the i'th row and j'th column. It is a number in the range 0 to 8 and is uniquely represented by a 3 bit block. That block is the output of S-box. For example, for input 01011 the row is 01, that is row 1, and the column is determined by 101, that is column 5. In row 1 column 5 appears 5 so that the output is 101 [13, 14].

Table 1. An example of S-box function

	Column No.							
Row No.	1	2	3	4	5	6	7	8
1	0	5	1	6	5	7	6	0
2	0	3	3	5	4	0	6	7
3	1	0	2	0	3	4	7	4
4	4	3	3	0	0	6	3	4

Transposition Cipher: P-BOX

A P-box (permutation box) is for characters and for bits parallels. For characters, their locations are changed, for example, a character in the first position of the plaintext may appear in the tenth position of the cipher text. A character in the eighth position may appear in the first position. In other words, a transposition cipher reorders the symbols in a block of symbols. For bits parallels, it transposes bits. Two types of permutations can have in P-boxes: the straight permutation and expansion permutation as shown in Figure (3).

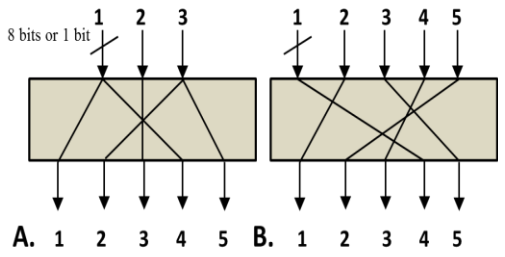

Fig. 3. P-box A. expansion permutation B. straight permutation

A straight permutation cipher or a straight P-box has the same number of inputs as outputs. In other words, if the number of inputs is N, the number of outputs is also N. In an expansion permutation cipher, the number of output ports is greater than the number of input ports [11].

Artificial Neural Networks

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems process information. An interesting feature of the ANN models is their intrinsic parallel processing strategies. However, in most cases, ANN is implemented using sequential algorithms that run on single processor architecture, and does not take advantage of the inherent parallelism. Another important feature of an artificial neural network is its ability to be learned by examples. Every time a neural network is made, training is needed to 'teach' it to give a specific output if a specific input is given [15].

One of the most useful algorithms of ANN training is back propagation network that uses back propagation learning algorithm. The back propagation neural network is essentially a network of simple processing elements working together to produce a complex output. These elements or nodes are arranged into different layers as shown in Figure (4) [2,3]:

- Input layer propagates a particular input vector's components to each node in middle layer.

- Middle layer nodes compute output values, which become inputs to the nodes of output layer.
- The output layer nodes compute the network output for the particular input vector.
-

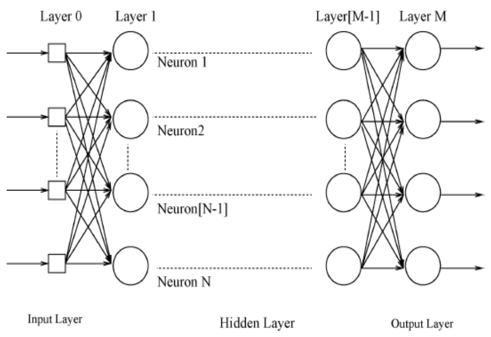

Fig. 4. Back propagation architecture

Simulation of Ciphering System

Multilayer back-propagation neural networks are used to simulate the blocks ciphering that are explained in section 3; the data set has been used with different inputs length. The neural network used in the encryption process is a 3-layer feed-forward network implementing the back propagation algorithm. The number of neurons each layer are shown in Table (2). Neural Network Toolbox in MATLAB program is used to implement the neural network for blocks ciphering. Training was conducted until the mean square error MSE fell below e-20 or reached a maximum iteration limit of 10000 as shown in Figure (5). The mean square error denotes the error limit to stop NN training. The MSE is the average of NN target output subtracted by the desired target output from all the training patterns. Two training algorithms are used; the traingdx which updates weight and bias values according to gradient descent momentum and an adaptive learning rate, the second is trainlm which is a network training function that updates weight and bias states according to Levenberg-Marquardt optimization. The output layer activation function is the pure line activation function. In the learning process, one thousands of data set are used to train the neural network.

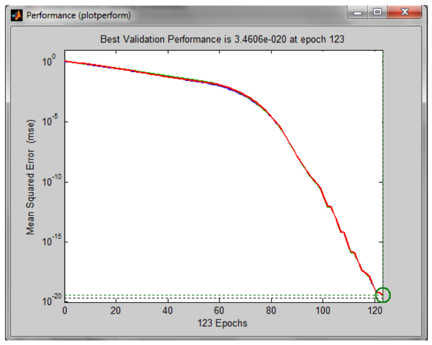

Fig. 5. the training process: the number of epochs with the MSE

After the acquisition of the suitable weights and biases, they are used to build the ANN on the FPGA device. The block diagram of the whole system is shown in Figure (6).

Table 2. Summary of simulation results of ciphering system in MATLAB

Cipher Block	Length of each inputs in bits	No. of inputs	No. of Hidden layer neurons	No. of outputs	Activation function of hidden layer	Training algorithm
Straight 1-bit	1	5	5	5	Pure line	traingdx
Straight 8-bits	8	5	5	5	Pure line	traingdx
Expansion 1-bit	1	3	3	5	Pure line	traingdx
Expansion 8-bits	8	3	3	5	Pure line	traingdx
S-box	1	5	10	3	Log sigmoid	trainlm
XOR	1	5	10	5	Log sigmoid	trainlm

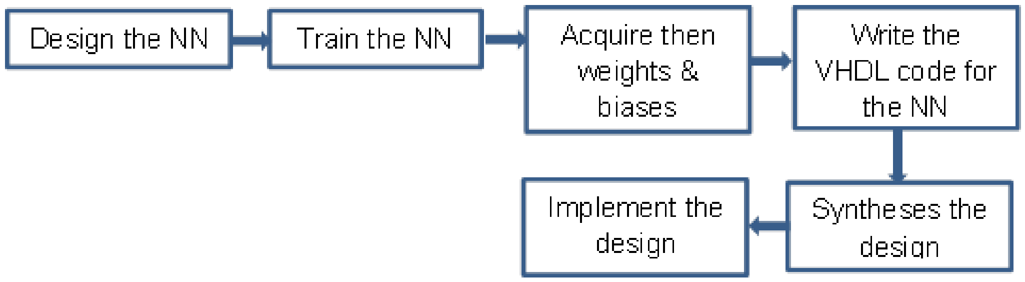

Fig. 6. The block diagram of the proposed system

FPGA Realization of Ciphering System

According to the normal structures of Neural Network, the hardware implementation is grander to the software approach because it can yield improvement of ANN's features, especially parallelism. Moreover, FPGA is a digital device with reconfigurable properties and flexibility. The back propagation neural networks for multiple blocks ciphering are implemented in this paper using one of the largest Xilinx FPGA devices, SPARTAN-3E, XC3S500E. This device has a capacity of (4656) logic slices and can operate at a maximum clock speed of 50MHz.

For the ANN implementation, fixed point computations with two's complement representation and different bit depths are chosen for the stored data (inputs data, weights, outputs, activation function, etc). It is necessary to limit the range of different variables:

- The length of inputs data depend on the type of block cipher as indicated in table 2 (1 or 8 bits are chosen)
- The length of each Weight is 16 bits (5 for integer part and 11 for fraction part).
- The length output from adder and multiplier, and activation function depend on the layer (hidden or output) and the type of block cipher shown in Table (2).

The design is characterized by the hardware description language VHDL as neural network architecture, finally the output of the neural network architecture passes through the output layer. The internal structure of each hardware neuron is composed of set of adders and multipliers to get the cumulative sum of the inputs multiplied by their weights and added to the bias of that neuron. The final cumulative sum then processed by the activation function of that neuron. In the following subsections, both a serial and a parallel version of the ANN architecture are described for 8-bit straight block cipher, the characteristics of this block cipher are shown in Table (2).

Parallel Version

The parallel architecture describes a kind of 'node parallelism', in the sense that it requires one hardware unit per neuron when working at a determined layer. With this strategy, all the neurons of a layer work in parallel and therefore get their outputs simultaneously. This is not a fully parallel strategy because the outputs for different layers are obtained in a serial manner.

Since the data transmission between layers is serial, every functional unit will also need some local storage for output data. The data in the parallel registers are introduced to the single activation unit. The output of the activation unit is either used as output of the neurons in hidden layer or it is stored in the output array RAM.

For the 8-bit straight block cipher, where 5 neurons exist at a hidden layer and 5 at the output layer, 5 hardware units are required. All hardware units will work in parallel when computing the outputs of the hidden layer, and the same ones will work when the output layer is computed. Given that all hardware units work in parallel for each input data, they need access to the associated weights simultaneously, and hence, the weight arrays RAM should be private for each neuron. For this reason, the 3-bit counter is used to address 5 arrays RAM for each layer. The parallel structure of the 8 bit straight block cipher is shown in Figure (7).

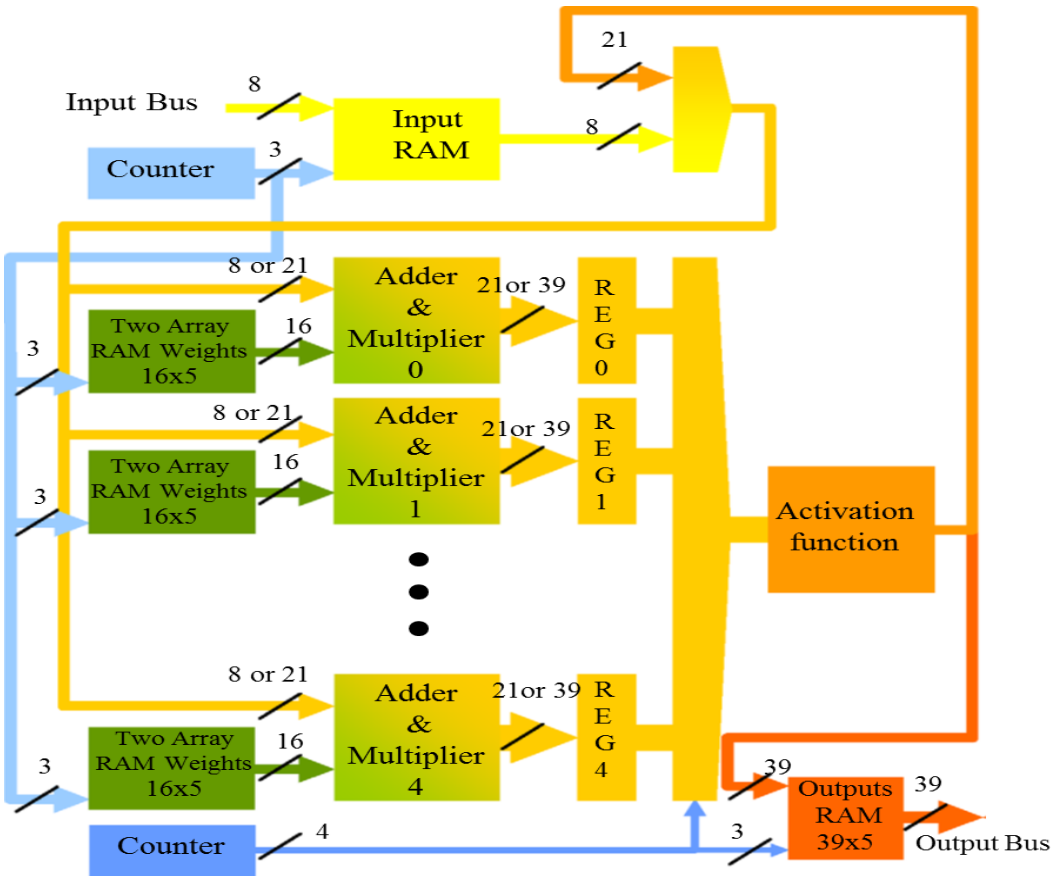

Fig. 7. The structure of the parallel neural network

Serial Version

The serial architecture has been designed to estimate the minimum area required to implement the ANN, although this implies a large execution time. Therefore, this serial version of the ANN consists in a single hardware neuron unit that carries out all the computations for every the neuron. The inputs of the hardware neuron unit are both the input data (plaintext) and their associated weights, which have been stored in separate array RAM. There is a number of array RAM to store the weights for each neuron, the number of these array RAM depend on the number of neurons at hidden and output layers in each block cipher. In addition, two separate ones to store the output of neuron in hidden and output layers. By separating the array RAM, the ANN will be able to read from the array RAM associated with input layer and to write the output of the hidden neurons in the same clock cycle. The activation function output is stored either in the hidden array RAM or in the output array RAM, depending on the layer of the computed neuron.

The addressing arrays RAM have been carried out by two 3-bit counters. The first 3-bit counter addresses the arrays RAM when reading them, and the second 3-bit counter addresses them when arrays RAM writing. The 6-bit address of the weights arrays RAM is computed by merging the addresses of two 3-bit counters. Figure (8) shows the structure of the serial version for straight 8-bit block cipher shown in Table (2).

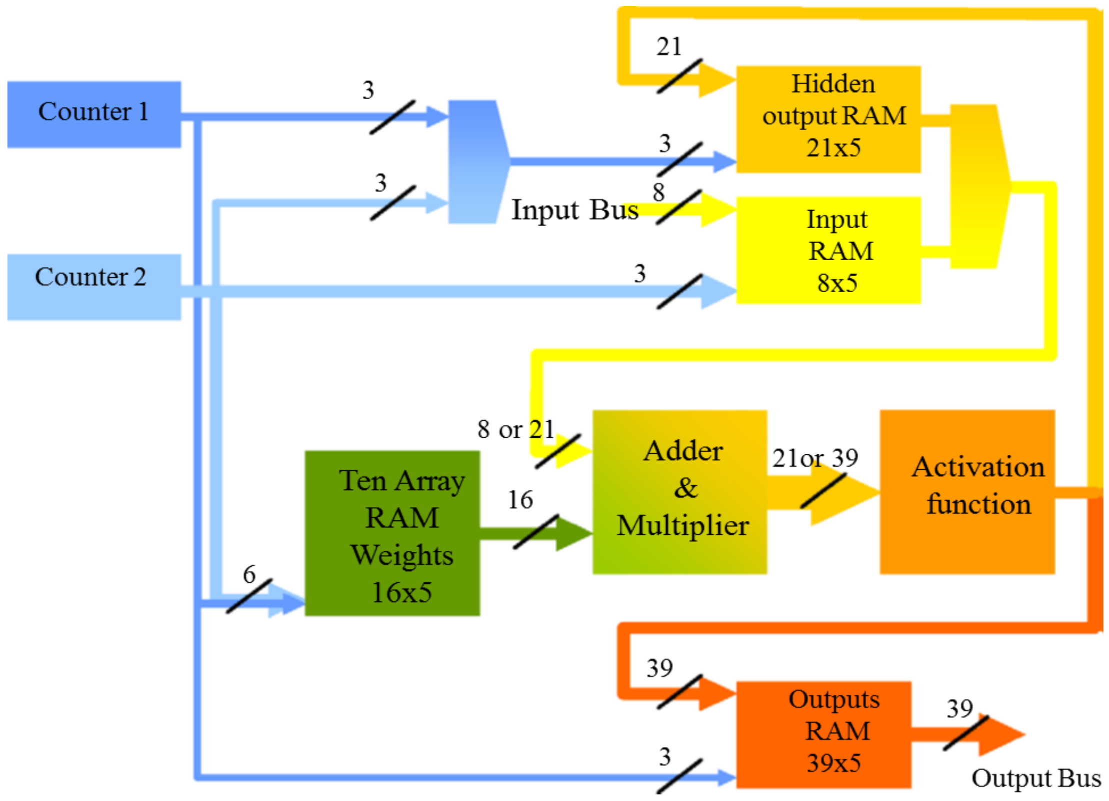

Fig. 8. The structure of the serial neural network

Results

After designing the neural networks, the testing shows that the NNs can be used actively as a tool for data ciphering. In the other hand, the hardware design of the networks gives a low error rate when comparing the numeric values with those obtained in software. Table (3) shows a comparison among real, MATLAB and FPGA numeric values for different ciphering techniques.

Table (4) displays the spent times to run the multiple blocks cipher software on computer (Pentium4, 3.00GHz, 2GB RAM) using MATLAB program. While, Tables (5) and (6) show the implementation results obtained after synthesizing both serial and parallel versions of the ANN for multiple blocks cipher. The results of each implementation can be characterized by the same parameters like (number of slices, maximum no. of clock, etc.) in order to make comparisons between the different implementations easier. As expected, results indicate that the serial version, with only a hardware unit, requires less area resources than the parallel one, where a hardware unit per hidden neuron was included. In the parallel version, the data throughput is higher than the serial version rounding (1.13-1.5) times, mostly due to the reduced number of clock cycles needed for each input data evaluation. After synthesizing the design, a slight difference can be observed in the maximum clock frequency, this is due to the different maximum combinational paths of the two approaches.

Table 3. Comparisons among real, MATLAB and FPGA results for different block ciphering techniques

Cipher block	Real values	MATLAB values	FPGA values	Cipher block	Real values	MATLAB values	FPGA values
Straight 1-bit	0	-0.000044	-0.00012	Straight 8-bits	204	204.0003	203.9930
	1	1.0001	1.0001		234	234.0003	233.9879
	0	-0.000054	-0.00006		107	106.9993	107.0052
	1	0.99983	0.9997		124	124.0006	123.9846
	1	0.99988	0.9998		36	35.99945	35.9879
Expansi-on 1-bit	1	1.000006	1	Expansion 8-bits	17	16.9985	17
	1	1.000006	1		221	221.0048	221
	0	-0.000019	0		148	148.004	147.9926
	1	1.000006	1		17	16.9985	17
	1	1.000006	1		221	221.0048	221
S-box	1	1.000003	0.9883	XOR	1	0.999992	0.9733
	0	0	0		1	1.000042	0.9688
	0	0	0		1	0.99994	0.9420
					0	0.000011	-0.0097
					0	-0.000004	-0.0300

Table 4. The spent times of MATLAB simulation for multiple blocks cipher

	Straight 1-bit	Straight 8-bit	Expansion 1-bit	Expansion 8-bit	S-box	XOR
MATLAB Simulation(sec)	0.035	0.0420	0.054	0.039	0.036	0.042

Table 5. The obtained results of FPGA resources for multiple blocks cipher using the parallel implementation

	Straight 1-bit		Straight 8-bit		Expansion 1-bit		Expansion 8-bit	
Logic Utilization	Used	Utilization	Used	Utilization	Used	Utilization	Used	Utilization
Number of Slices	325	6%	315	6%	205	4%	178	3%
Number of Slice Flip Flops	78	0%	78	0%	78	0%	78	0%
Number of 4 input LUTs	629	6%	609	6%	392	4%	344	3%
Number of MULT18X18SIOs	20	100%	20	100%	12	60%	12	60%
Number of GCLKs	1	4%	1	4%	1	4%	1	4%
Maximum Frequency (MHz)	46.640		46.491		61.712		61.448	
Maximum No. of clocks	7		7		7		7	
Throughputs (Mb/s)	259.8514		259.0213		343.824		342.3531	

Table 6. The obtained results of FPGA resources for multiple blocks cipher using the serial implementation

	Straight 1-bit		**Straight 8-bit**		**Expansion 1-bit**		**Expansion 8-bit**		**S-box**		**XOR**	
Logic Utilization	Used	Utilization	Used	Utilization	Used	Utilization	Used	Utilization	Used	Utilization	Used	Utilization
Number of Slices	188	4%	184	3%	131	2%	104	2v	401	8%	385	8%
Number of Slice Flip Flops	85	0%	85	0%	84	0%	84	0%	45	0%	114	1%
Number of 4 input LUTs	323	3%	317	3%	208	2%	160	1%	787	8%	753	8%
Number of MULT18X18SIOs	10	50%	10	50%	6	30%	6	30%	15	75%	13	65%
Number of GCLKs	1	4%	1	4%	1	4%	1	4%	1	4%	1	4%
Maximum Frequency (MHz)	48.940		49.202		61.866		69.897		30.461		30.487	
Maximum No. of clocks	11		11		9		9		14		16	
Throughputs (Mb/s)	173.5145		174.4435		268.086		302.887		97.91036		85.74469	

From Table (5), it can be seen that the straight block cipher method use all the available multipliers of the Spartan 3e FPGA, the other methods use less number of the available multipliers. In the parallel structure of the neural network, the output of each layer is available in the same clock, this give a high throughput for the input and hidden layers. The final output takes more clock cycles because each output neuron needs one clock cycle to be read from the output array RAM.

From Table (6), it can be noted that XOR and S-box block cipher require large resources area because they have great numbers of neurons in their structures shown in Table (2). In the serial structure of the neural network, the output of each neuron of each layer is available in the one clock, this give a less throughput for layer one and the hidden layer. As, the final output for each neuron in the output layer take one clock cycle to be read from the output array RAM.

Figure (9) shows the time diagram of the parallel implementing for an 8-bit straight ciphering block. While, Figure (10) shows the time diagram of the serial implementing for an 8-bit expansion ciphering block.

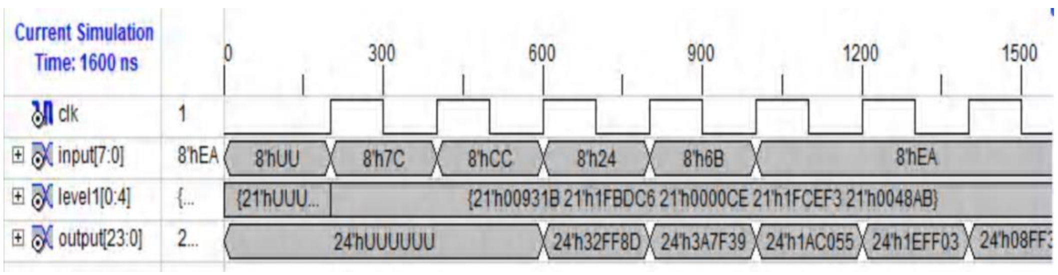

Fig. 9. Time diagram of the parallel implementing for an 8-bit straight ciphering block

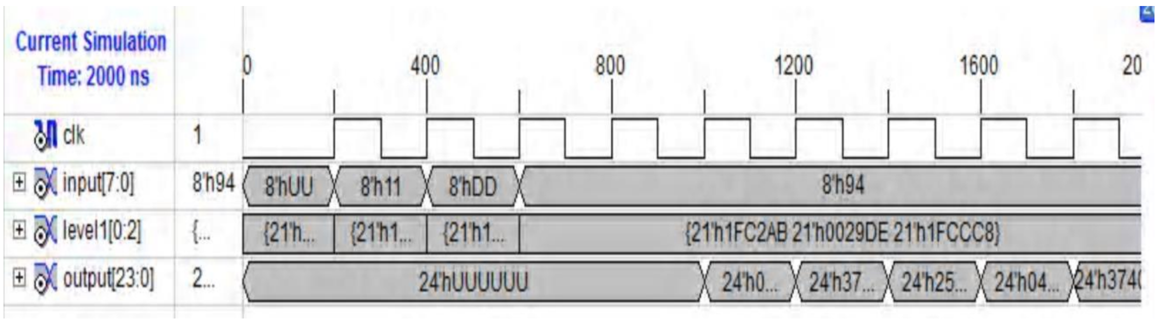

Fig. 10. Time diagram of the serial implementing for an 8-bit expansion ciphering block

Conclusions

Artificial neural networks can be trained to act as a cryptography system; different block ciphering techniques are realized using ANNs. Then the NNs are described with the VHDL and implemented on the FPGAs using two approaches: serial and parallel versions.

Implementing the cryptography system on FPGA makes it faster, reliable and the method of ciphering can be changed easily by altering the weights of the neurons. The numerical precision is chosen carefully when implementing the ANN on FPGAs; higher precision gives more accurate results but takes more FPGA resources. Obtained results from the hardware designs show accurate numeric values to cipher the data. As expected, the results indicate that the serial version, with only a hardware unit, requires less area resources than the parallel one. As, the data throughput in parallel version is higher than the serial version in rang between (1.13-1.5) times. Also, a slight difference can be experimental in the maximum frequency.

References

1- Jihong, L., Baorui, L., and Deqin, L., “Design and Implementation of FPGA-Based Modified BKNN Classifier”, International Journal of Computer Science and Network Security (IJCSNS), Vol. 7, No. 3, pp. 67-71, March 2007.

2- Shihab, K., "A Back Propagation Neural Network for Computer Network Security", Journal of Computer Science, Vol. 2, No. 9, pp. 710-715, 2006.

3- Rafid, A. Kh., “Hardware Implementation of Backpropagation Neural Networks on Field programmable Gate Array (FPGA)”, Al-Rafidain Engineering Journal, Vol. 16, No.3, pp. 62-70, Aug. 2008.

4- Antony, W. S., Medhat, M., and Shawki, A., ”The Impact of Arithmetic Representation on Implementing MLP-BP on FPGAs: A Study”, IEEE Transactions On Neural Networks, Vol. 18, No. 1, pp. 240-252, January 2007.

5- Chandra, B., and Paul, P. V., “Applications of Cascade Correlation Neural Networks for Cipher System Identification”, World Academy of Science, Engineering and Technology, Vol. 26, pp. 311-314, 2007.

6- Arvandi, M., Wu, S., and Sadeghian, A., “On the Use of Recurrent Neural Networks to Design Symmetric Ciphers”, Computational Intelligence Magazine, IEEE, Vol. 3, Issue 2, pp. 42-53, May 2008.

7- Abdelhalim, M. B., Aslan, H. K., Mahmoud, A., and Farouk, H., “A Design for an FPGA Implementation of Rijndael Cipher”, ICGST-PDCS Journal, Vol. 9, Issue 1, pp. 9-15, October 2009.

8- Sivagurunathan, G., Rajendran, V., and Purusothaman, T., “Classification of Substitution Ciphers using Neural Networks”, International Journal of Computer Science and Network Security (IJCSNS), Vol. 10, No. 3, pp. 274-279, March 2010.

9- Siddeeq, Y. A., and Ali H. M., “AES Cryptosystem Development Using Neural Networks”, International Journal of Computer and Electrical Engineering, Vol. 3, No. 2, pp. 315-318, April 2011.

10- Khaled, M. A., Alaa, H. A., Waiel, A., and Mohamed, A., "Applying neural Networks for Simplified Data Encryption Standard (SDES) Cipher System Cryptanalysis", International Arab Journal of Information Technology(IAJIT), Vol. 9, No. 2, pp. 163-169, March 2012.

11- Behrouz A. Forouzan, “Data Communication and Networking”, 4th edition, McGraw-Hill Companies. Inc., New York, pp 1107, 2007.

12- Bo, Z., "XOR Based Optical Encryption with Noise Performance Modeling and Application to Image Transmission over Wireless IP LAN", Master thesis, Peninsula University of Technology, pp 136, 2004.

13- Nimmi, G., “Implementation of Optimized DES Encryption Algorithm upto 4 Round on Spartan 3”, International Journal of Computer Technology and Electronics Engineering (IJCTEE), Vol. 2, Issue 1, pp. 82-86, Jan 2012.

14- Pooja, R., Jaikarn, S., Mukesh, T., and Sanjay, R., “Optimized DES Algorithm Using X-nor Operand Upto 4 Round on Spartan3”, International Journal of Computational Engineering Research (IJCER), Vol. 2, Issue. 8, pp. 193-200, December 2012.

15- Zaid, G. M., "Design and Implementation of Multilayer Neural Network for Speech Recognition Using FPGA", Master Thesis, Technical College of Mosul, pp91, 2007.