Implementation of Neural Network for Image Processing on the FPGA Programmable Systems

KAZIMIERZ WIATR, PAWEL CHWIEJ
Institute of Electronics
AGH Technical University of Cracow
30-059 Krakow, Mickiewicza 30
POLAND

Abstract: - In this paper the implementation of fragment digital Cellular Neural Network (CNN) for image processing on the Field Programmable Gate Array (FPGA) and it's experiment results are present. The high processing speed of the network is use to provide real time processing. Results shows that the architecture CNN and FPGA and implementation has good corespondent.

Key-Words: - Image Processing, Neural Networks, Reconfugurable Systems, Field Programmable Gate Arrays

1. Introduction

Real-time image processing requires large computational powers. It is useful to apply dedicated equipment for the realization of image processing algorithms. The development of programmable systems allows for the exceptionally good application of solutions known as CCM (Custom Compatible Machine). It is beneficial to apply neural network structures in this situation.

The aim of this paper is to investigate the possibilities of the implementation of the chosen neural network for image processing on the FPGA programmable systems. The authors assume that the network learning occurs in the all applications computer while the implementation on the FPGA concerns the neural network already taught.

The kind of neural networks that have been widely applied in various fields is called the cellular neural network. It combines the features of the artificial neural network, that is information processing by using identical elements, simple structures and functions, and the cellular automata model, that is the regular structure and local connections among the elements. It is characteristic that the connections weights are fixed and the network displays the recursive nature. Moreover, the structure matches quite well the FPGA array structure.

2. Applied neural network

The cellular neural network consists of identical elements combined closely together and forming a regular geometrical architecture. Signals are processed simultaneously and the interactions among the cells occur locally. However, despite the local range of these interactions, the cellular neural network is able to perform global processing, which means that the outcome may depend on the values of the cells situated beyond the closest vicinity. This happens because the cell is controlled by the input signal of the neighboring cells, and these cells vary with time. The basic field for the application of such network is image processing. Works on cellular networks concern:

- real image segmentation,
- noise reduction,
- contour detection,
- feature detection.

A cell is the single element of the network. Cells are connected through local channels and form a regular, flat (two-dimensional) grid or lattice. The info introduction into all the cells is parallel. The cell interactions cause dynamic transformation processes in the network. When the network attains its equilibrium, we obtain the input signal which, quite often, is a binary one. Despite the fact that the cells are connected

locally, the processing is global, that is the whole network participates in the process. The cellular neural network structure is regular; every cell is connected to the neighboring cell. The only exception are the boundary cells which are stimulated by artificially generated signals. Their selection depends on the application of a given network. There are solutions that connect the cells from one edge with the cells from the opposite edge creating a toroidal network.

The regularity of the network geometrical structure enables its imple-mentation in the real - time image processing and analysis. It is possible to create such network using the semiconductor technique.

Apart from the output and input signals, every cell has its polarization signal. Polarization signal has a fixed value but is not identical for every cell in the network. The transforming function models cell X_{ij} state into the output state Y_{ij} (fig.1).

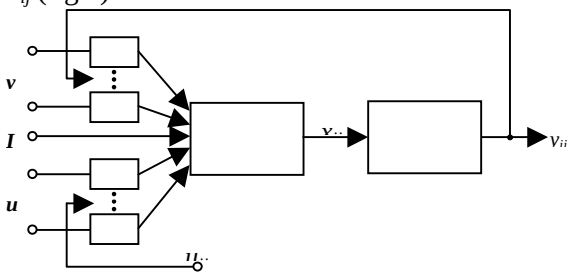

Fig. 1. Plan Cellular Neural Network

This function is segmentally linear, of a unicast slope in the origin of the coordinate system. We can present the time change dynamics of the state and output signals through the nonlinear equations:

$$x(n+1) = \sum A \cdot y(n) \cdot \sum B \cdot u(n) \qquad (1)$$

$$y_{ij} = \begin{cases} -1 & for \quad x < 0 \\ 1 & for \quad x \geq 0 \end{cases} \qquad (2)$$

where:
i,j – indexes referring to the controlled cell,
n – number period.

Every cell is connected to its neighbor only. The neighboring cells interact. One has to find the area intuitively and mark it as N_{ij}. Every cell has its initial value of the state *x* variables, u input (its own as well as its neighboring cells) and y output (its own as well as its neighboring cells). All of the signals control the cell. Every cell processes the signals identically. The outcome is the weighted sum of controlling signals which is nonlinearly transformed. The state of each cell is activated after the time $t > 0$ and then it attains the equilibrium. The cellular neural network always returns one stable state. The network working is presented on the (fig. 2)

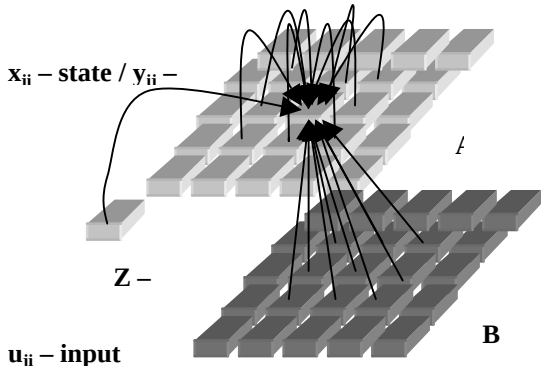

Fig. 2. The network working with model

3. Technological plane

Virtex programmable systems by Xilinx have been applied as a technological plane for the implementation of the neural network. The Virtex FPGA systems enable programming of complicated logic because one of their features is high density data compaction. The deve-lopment of silicon integrated circuit technology made it possible to achieve a very quick circuit which broke the parameter barrier of its predecessors and extended the traditionally applied programmable logic. Drawing from experience of producing previous series of FPGA systems, the designers obtained the system of a high ability to implement the developed and complicated projects. Since the appearance of the first XCV50 system, the number of gates has increased 20 times in the XCV1000 system, and the number of CLB has increased 16 times. After the introduction of the next 1,8 – Volt Virtex E series the program-mable systems became the alternative for the ASIC systems.

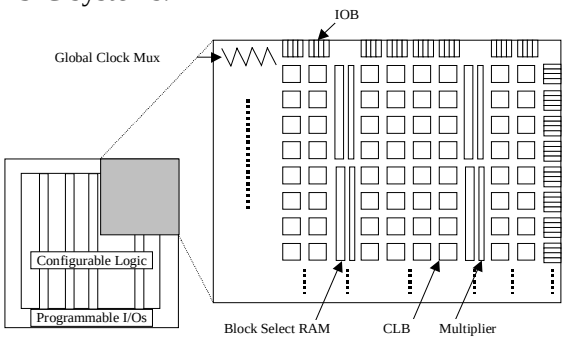

Fig. 3. Architecture VirtexII systems

Five different connection tracts applied in the systems assure the communication on various distances. They also have a considerable impact on the architecture elasticity. It increases the possibility of parallel processing of many signals. Improving the elasticity is also a matter of 16 global clock lines which can be alternatively used as input-output ports. Virtex II systems are equipped with the Select Link interface allowing for quick communication with the other Virtex systems (fig. 3). They also include the JTAG interface compatible with the IEEE1532 standards. Owing to this, there is a possibility of partial reconfiguration of the system structure during work. The system architecture is modeled on the CLB structure. A single cell contains four inputs, one output, one carry and others. In Virtex systems every CLB cell is built of four logical cells organized into two identical slices (fig. 4). Every of the four-input LUT tables can be optionally configured as single or bi-port RAM memory and also as a 16-bits shift register.

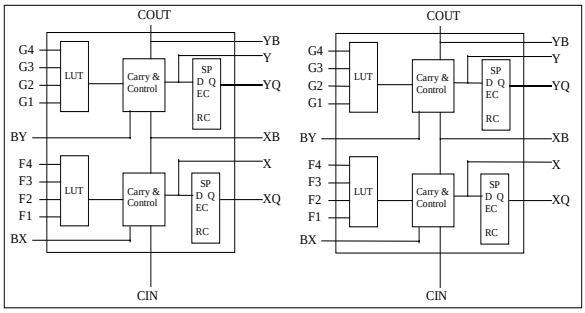

Fig. 4. Two-Slice Virtex CLB

The inner Virtex II system structure is subordinated to obtaining the maximally high work speed. One of the most important elements influencing the resultant productivity is the carry generator among the basic blocks realizing logical functions. The possibility of creating blocks for generating single-level multi-argument functions or horizontal quick carry functions is quite important. Additional ORCY gates situated in every cell of the CLB block have been applied for such realizations in the basic logical cells of the Virtex II systems.

4. Image cognition by cellular NN

The task of the network is to extract from the image a defined object. This object must reach the maximum level in the grayscale, that is, it must consist of elements that differentiate from the background elements. The second condition is that the element belonging to the object must have at least one neighboring element belonging to the background. The object cognition would be realizable when two mutually independent pieces of information are delivered to the network: the information about the wanted object and input data – containing the processed image. In order to realize these assumptions in the cellular neural network, one should define the rules of delivering these two pieces of information. The input image is introduced into the network cells according to the rule telling that we assign for the background pixels “-1”, and for the object pixels “+1”. The wanted object is defined by means of start states. The object has an assigned “+1” value, and the remaining cells have “-1” value.

Transformation rules:

a) If the neural cell has "+1" first state and “+1” input signal, then “+1” output signal should appear.
b) If the cell input is "+1" and the output signal of any of the neighboring cell is "+1", then this cell output will also be “+1”.
c) If the cell input signal equals "-1", then all the cells outputs must adopt “-1” value.

The second transformation rule is called the diffusion rule. The phenomenon occurs when every cell analyses its neighbor’s input signal values. In this rule it is sufficient when the size of the operator A is 3×3. The values of operator A should be positive because only such values lead to the increase of the state variable value.

The controlling operator B defines which cell belongs to the object and which to the background. If the cell belongs to the object, it is necessary that the operator does not increase its values. That is why, it is beneficial when only the central element is non-zero and positive.

Rule a) is reflected in inequality:

$$B + A_{00} - 8A + I > 0 \qquad (3)$$

In order to reflect rule b) we choose the worst case when only the output signal controlling the cell is the element of the object and its value is “+1”:

$$B - A_{00} - 6A + I > 0 \qquad (4)$$

If the element of the object has not been selected, is must remain as the background:

$$B - A_{00} - 8A + I < 0 \qquad (5)$$

The elements belonging to the background remain the background because:

$$-B + A_{00} - 8A + I < 0 \quad (6)$$

$$-B - A_{00} + 8A + I < 0 \quad (7)$$

The following pattern may be the solution of these inequalities:

$$A = \begin{bmatrix} q & q & q \\ q & 1+6q & q \\ q & q & q \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 8q & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad I = 5q \quad (8)$$

5. NN implementation in FPGA

Network implementation is possible only in Virtex systems because the Virtex family predecessors contain too few elements for encompassing the network structure. I have made an analysis for the XCV1000 system implementing two networks of the neighboring area : r = 1 and r = 2 in various configurations (the results are presented in tab.1):

a) network r = 1, 1bit inputs and outputs, 4 bits coefficient with sign,
b) network r = 1, 4 bits inputs and outputs with sign, 4 bits coefficient with sign,
c) network r = 1, 8 bits inputs and outputs with sign, 8 bits coefficient with sign,
d) network r = 2, 1 bit inputs and outputs, 4 bits coefficient whit sing.

I have applied signum activation function into every network. The network structure is toroidal. Apart from the intercellular connections and I/O signals, the cellular neural network has controlling systems and connections in order to exam whether all the cells have completed their calculations and whether it is possible to send the results to the cells output. It is also able to clear its values.

I present below the results acquired after the 3×3 network implementation. The results perform defined functions: angle embossment, edge (border) detection. These are three networks of the same template with the input info variable:

– 2 bits inputs and outputs with sign,
– 3 bits inputs and outputs with sign,
– 8 bits inputs and outputs with sign.

The results are presented in tab. 2. In particular tab. 2 presents the comparison of the system area occupied by the given network and the comparison of the network working speed.

Template:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 4 & -1 \\ -1 & -1 & -1 \end{bmatrix}, \quad I = -5 \quad (9)$$

requires the use of 3 bit coefficients but only 11 out of 19 are non-zero.

The second network type is the network used for edge detection. I have implemented three networks just like in the former example. The results are presented in tab. 3. In particular tab. 3 presents the comparison of the system area occupied by the given network and the comparison of the network working speed.

Network template:

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 7 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad I = 5 \quad (10)$$

As in the former case, the number of non-zero elements equals 10.We have analyzed the cellular networks in various configurations implemented in the XCV1000 Virtex system. The area occupation by the particular networks the percentage system area is presented in fig. 5. Numbers is same as above.

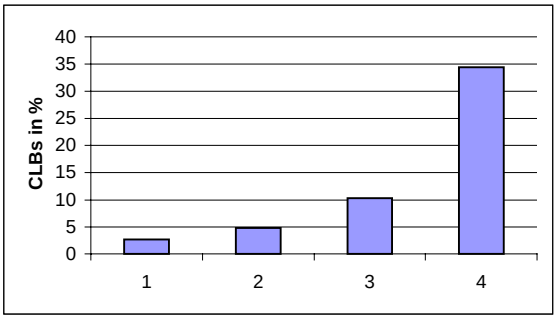

Fig. 5. Waste percentage area system by networks

Network implementation r = 2 occupies a considerable space. Fig. 6 presents the maximum working speed of the system including a given network. First line presents the speed before the optimization, the second line after the optimization. Fig. 7 shows two speed increments (augments) after the optimization. The optimization of the extended network of r =2 did not bring any effect. It is caused by the considerable area occupation and limiting the system maneuvers during the optimization.

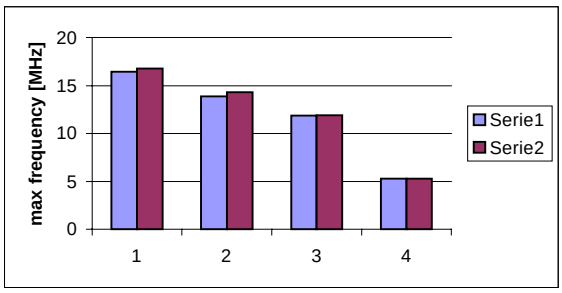

Fig. 6. Max speeds for system, series 1 - optimization area, series 2 - optimization speed

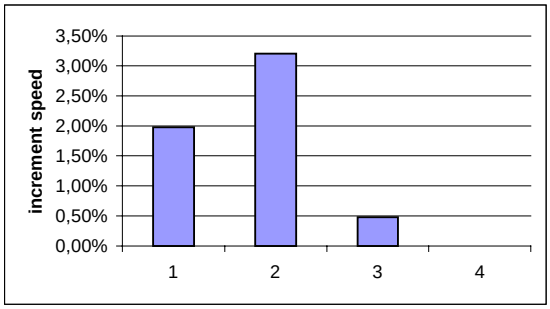

Fig. 7. Increment speed

6. Conclusions

The above presented networks are configured in maximum values because, in reality, not all the coefficients of the pattern are non- zero. It is also unnecessary to record the coefficients in 8 bits. This solution occupies considerable area and decreases the system speed. In the analysis, I have introduced two kinds of network: for angle embossment and edge detection. They have 11 non- zero pattern coefficients and they characterize , in comparison to the preceding networks , in good speeds, at little waste of the system area. The speeds, even at 8 bit input and output do not fall below 20 MHz. For the angle embossment network the speed only slightly decreases during the bit increase. The use of the cellular image processor with the application of these networks gives a real chance for the physical utilization of the network. Summing up the results of my work, I can assert that cellular neural networks are suitable for the implementation in FPGA systems. However, the utilization of a network implemented in FPGA systems has to take place with cooperation with other systems. The construction of too large a neural network and its implementation in FPGA system, despite the possibility of using XC2V8000, is not a good solution because it considerably decreases the system speed. The use of the cellular image processor, which works sequentially, seems to be a better solution. We should also apply a suitable interface making it possible to introduce images into the "digital world". Initial processing can precisely take place in the cellular neural network. The neural network implemented in the FPGA system can serve for various tasks concerning image processing, we should only exchange suitable patterns for given tasks. The authors truly hope for the use of neural networks for the realization on higher levels, and processing images connected with diagnosing the object shapes. The authors' current works basing on the experiments presented in the paper proceed in this direction.

References

[1] L. O. Chua, L. Yang, *Cellular Neural Networks: Theory and Applications*, 1988.

[2] T. Roska, J. Vandewalle, *Cellular Neural Networks*, J. Wiley & Sons 1993.

[3] F. A. Savaci, J. Vandewalle, *On the Stability Analysis of Cellular Neural Networks*, IEEE Trans. on Circuits and System I, vol. 40, March 1993, pp.213-214.

[4] P. P. Civalleri, M. Gilli, L. Pandolf, *On Stability of Cellular Neural Networks with Delay*, 1993.

[5] L. O. Chua, B. Shi, *Multiple Layer Cellular Neural Networks: A Tutorial*, University of California, Berkeley, 1990.

[6] L. O. Chua, *CNN: A Vision of Complexity*, Int. Journal of Bifurcation and Chaos,1997.

[7] K. Wiatr, E. Jamro, *Implementation of Multipliers in FPGA Structure*, IEEE Symp. on Quality Electronic Design, San Jose, CA, IEEE Computer Society 2001, pp. 415- 420.

[8] K. Wiatr, E. Jamro, *Constant Coefficient Multiplication in FPGA Structures*, Euromicro Conf. on Digital Systems Design: Architecture, Methods, and Tools, Maastricht, IEEE Computer Society 2000, pp. 252-259.

Table 1. Values for optimization of area or speed

Elements FPGA	Network $r = 1$, 1 bit inputs and outputs, 4 bits. coefficients, XCV1000		Network $r = 1$, 4 bit inputs and outputs, 4 bits coefficients, XCV1000		Network $r = 1$, 8 bit inputs and outputs, 8 bits coefficients, XCV1000		Network $r = 2$, 1 bit inputs and outputs, 4 bits coefficients, XCV1000	
	speed	area	speed	area	speed	area	speed	area
Slices	746	742	1318	1317	2863	2862	9511	9511
Flops	18	18	36	36	81	81	150	150
Latches	9	9	27	27	72	72	125	125
IOBs	64	64	100	100	217	217	176	176
4 input LUTs	516	508	2173	2173	4459	4459	9565	9565
Total gates	9117	9069	25530	25530	53574	53574	146827	146827
Max frequency [MHz]	16,762	16,437	14,298	13,854	11,887	11,830	5,274	5,274
Max delay [ns]	11,421	14,617	15,276	15,113	16,016	10,257	19,963	19,963

Table 2. Area values for networks

Elements FPGA	2 bits inputs and outputs with sign	3 bits inputs and outputs with sign	8 bits inputs and outputs with sign
Slices	514	547	600
Flops	43	45	90
Laches	34	36	81
IOBs	55	109	244
4 input LUTs	812	885	974
Total gates	9760	10245	11382
Max frequence [MHz]	22,702	22,362	22,142
Max delay [ns]	12,351	9,385	8,621

Table 3. Area values for networks

Elements FPGA	2 bits inputs and outputs with sign	3 bits inputs and outputs with sign	8 bits inputs and outputs with sign
Slices	489	626	733
Flops	27	45	90
Latches	18	36	81
IOBs	53	105	211
4 input LUTs	769	931	1156
Total gates	9294	11175	13542
Max frequency [MHz]	25,401	23,910	21,797
Max delay [ns]	10,542	11,829	8,757