A 145 MHz User-Programmable Gate Array

Eduardo do Valle Simões e Dante Augusto Couto Barone
Universidade Federal do Rio Grande do Sul - Porto Alegre - Brazil

Abstract

This work aims to present the most relevant results derived from the development of a novel FPGA matrix. Nevertheless its capacity, its highlights are low cost and its ability to deal with high frequencies. This prototype matrix, named FLECHA, presents some structural novelties, as a different approach to perform the interconnections between logic cells and I/O Pads and an internal controller.

One of the major contributions of this work deals with the strategy of placing the logic cells in rows, allowing a drastic reduction of the number of switches without reducing interconnection capacity between logic cells and I/O Pads. Following this principle, a new matrix presenting 600 equivalent gates was constructed including 40 logic cells and 40 programmable I/O Pads. The FLECHA matrix is intended to implement simple logic functions as for fast "glue logic" between processors and memories, reaching an operational frequency of 145 MHz.

1: Introduction

Generally, great capacity FPGAs have many electrical structures in the propagation path of internal electrical signals. Many times, it provokes a substantial delay in the logic function generation and makes them inefficient to interface between fast devices like microprocessors and memories. The FLECHA project will attack this problem with the development of a high speed field-programmable gate array, with low-capacity/high-frequency logic cells. This paper will show a new internal interconnection strategy, different from the used cross-bar switch, aiming to improve operational speed with a low-cost design methodology.

A prototype matrix, called FLECHA, was developed according that new design methodology. This matrix has low logic density, but its original logic cells and the internal interconnection strategy allow high speed applications.

1.1: Presentation of the FLECHA matrix

The FLECHA matrix and all of its characteristics are directed to glue logic applications, where gate complexity is not too high, but the number of I/O signals that need to be processed is. This situation may conduct to logic cells waste, in case of high capacity cells utilization. A good solution for this application is to choose 3-input logic cells, once the majority of cases present simple 2-input logic gates. The high performance of 145 MHz allows the employment of FLECHA matrix to provide control logic between microprocessors and memories.

FLECHA matrix was designed presenting the same number of logic cells and I/O cells. This is an efficient way to get the balance between implementation capacity and the great demand of interconnections required by glue logic applications. Aiming to reduce prototyping costs, a 48-pin DIL-type package was chosen. Eight pins are used as control or power Pads, and 40 pins were designed as programmable I/O Pads. This I/O limitation led to the utilization of an array containing 40 programmable logic cells.

The general topology of the FLECHA matrix is shown in figure 1, where its 40 logic cells are distributed in 4 rows or groups with 10 logic cells and 10 I/O Pads that can communicate through the central bus to implement complex large functions. This figure presents the regularity of the chip layout, developed with ES2 CMOS 1.2 μm technology. This topology allows the array to implement a function with up to 10 cells, or many small functions in each row. There are cases that we need to use more than 10 cells to implement a complex function, then we can connect two or more groups of 10 cells in the matrix by the central bus. Thus, user-programmable logic cell arrays are viable alternatives to conventional mask-programmed gate arrays in most applications.

1.2: Design for future expansion

The layout of the matrix was designed according a modular approach, aiming to minimize the necessary effort to future modifications at the original matrix architecture that are essential to the development of new family members. Each row with 10 logic cells and 10 I/O Pads has

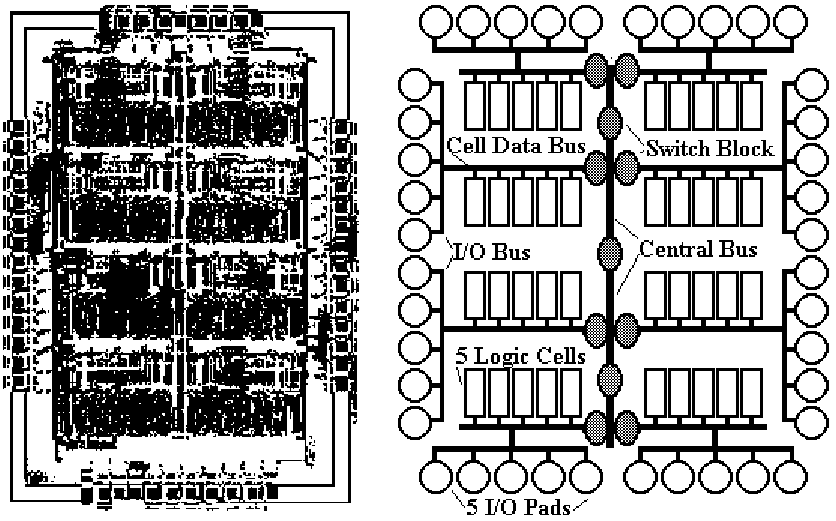


Figure 1 -- Layout and general topology of the FLECHA matrix.

its own switching system and the necessary configuration memory. This modularity allows future utilization of many rows to built high capacity arrays, with the same characteristics, to perform more general applications.

2: Design considerations

The FLECHA matrix architecture has three distinct elements: the programmable logic cells, the configurable I/O Pads and the interconnection network. Ten logic cells are placed in groups with ten I/O Pads to built the rows, and these rows are regularly distributed by the matrix. This solution provides a direct communication between Pads and the internal logic cells. The application of the matrix in real problems has showed the superiority of the implemented solution instead an interior logic cell array involved by an I/O Pad ring.

2.1: Programmable logic cell

Figure 2 shows the architecture of a three-input logic cell that provides the functional elements to construct the user's logic gates. Each logic cell has a Functional Block and an Output Block. A shift-register chain and an 8:1 multiplexer are used to perform the truth table of the functional block of the cell. The FLECHA logic cells are able to implement any Boolean function of up to three inputs, representing the truth table of the function in the internal static memory cells. In that case, the search table based solution showed up an approach more efficient than other tested possibilities, like multiplexers [4] or Universal Logic Modules [7], because of its better performance and future expansion without necessity of grand modifications in the electric circuit of the logic cell.

In the Output Block, a 2:1 multiplexer can connect an internal register to the output of the functional block if it is necessary to implement sequential logic, as it is shown in figure 2. This register is an edge-triggered D-type flip-flop. Therefore, the 40 logic cells are capable of implementing any 40 independent sequential function of up to 3 input variables. Four multiplexers are used to connect the three inputs and the output of the cell to the lines of the data bus, which is used to interconnect each cell side by side, as shown in figure 2. Consequently, each logic cell is an independent logic unit, with its own configuration memory, and can be placed side by side to built arrays with different logic capacities [11].

2.2: Internal configuration memory

The functions of the logic cells, the configuration of the I/O Pads, and the state of the interconnection network are defined by a configuration program stored in internal static memory cells. These cells consist of a shift-register chain that can be loaded automatically at power-up. The process of loading the configuration memory is independent of the user logic functions that are implemented in the matrix.

2.3: Programmable input/output pad

Pointing to a simple design, bi-directional Pads from the available cell library 1.2µm [5] are connected to an independent control circuit to provide the configurable input/output blocks. The I/O Pad architecture allows the separation between the Pad proper circuit (buffer and filter) and the configuration structure. Thus, the Pad circuit can be re-designed separately, by the case of a new process technology becomes available. One external pin and a

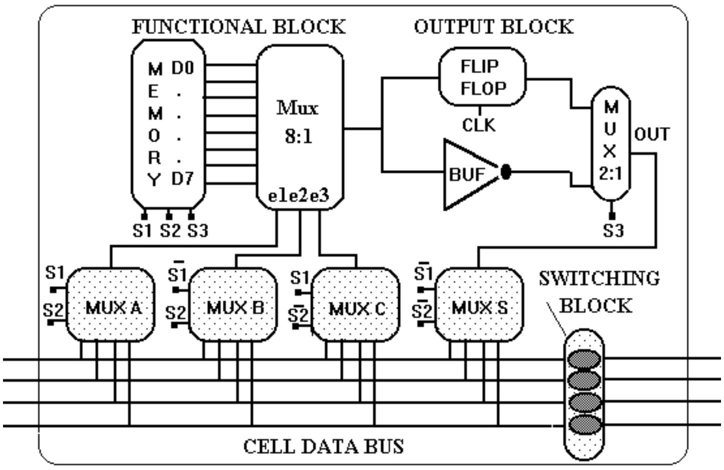


Figure 2 -- Architecture of a three-input logic cell and its lateral interconnections.

portion of the internal configuration memory are associated to each Pad. FLECHA matrix Pads are grouped in groups of 5 and a 5-line bus is used to route each group of Pads to its corresponding logic cell in a row. Some alternative paths were implemented to allow the utilization of the Pads that are not necessary to one row by other logic cells, without busying of the central bus.

2.4: Internal interconnection network: lateral placement technique

The FLECHA interconnection topology was designed according to the lateral placement technique [11], developed to reduce the amount of switches without reducing the interconnection capacity of the logic cells. This technique consists to place side by side all the cells that implement the same logic function (it means that a powerful software tool will be necessary to place and route these cells). Once they are placed in rows, those cells need only to exchange signals with their neighbors, and this fact permits a great simplification in the switching structures. A new row, under or above, should be connected by the central bus utilization, if it is necessary to use more than one row of cells to implement one big logic function. Figure 3 presents this technique applied to 5 logic cells and its I/O Pads in a row.

Commercial FPGAs usually have complex cross-bar switch, allowing a logic cell to communicate with all its neighbors [13][1][2][9]. This system uses a great amount of the total silicon area and many configuration bits are necessary to control its state [12]. The presented solution is directed to reduce the amount of these switches, without disrupting interconnection capacity between logic cells and I/O Pads. The lateral placement technique consists of the utilization of a powerful placement tool, that is able to arrange laterally the logic cells that constitute the same logic function. Once laterally placed, these cells just need to exchange information with their neighbors. Thus, the necessary switching system is hardly simplified, reducing internal delays of the circuit, what improves roughly its speed [8].

This new switching system is also controlled by the internal shift-register chain and permits a great economy of memory cells and switches. The FLECHA I/O Pads were also connected according to this technique, so each row with 10 logic cells is able to communicate with 10 I/O Pads through an independent interconnection system.

Under this perspective, many circuits were implemented to specify the necessary amount of routing paths to get the balance between interconnection capacity and occupied silicon area. Figure 3 represents one of the best results obtained: a simple and efficient approach (performing 296 switches, at all). This solution is able to implement typical glue logic applications with reduced logic cell waste (estimated 75% of utilization with large logic gates, and 90% with simple ones).

2.5: The FLECHA matrix configuration data loading process

FLECHA matrix configuration process is established programming its internal static memory cells. They will specify the function and interconnection of the logic cells and I/O Pads. This internal memory consists of a long shift-register chain. This data is transmitted through the chain under the synchronization of global clock pulses. When each transmitted bit reaches its corresponding position in the chain, the hole matrix is well suited according the implemented device.

Configuration data is loaded from an external source at power-up, or under a reconfiguration command. At the beginning of the configuration process, an internal counter

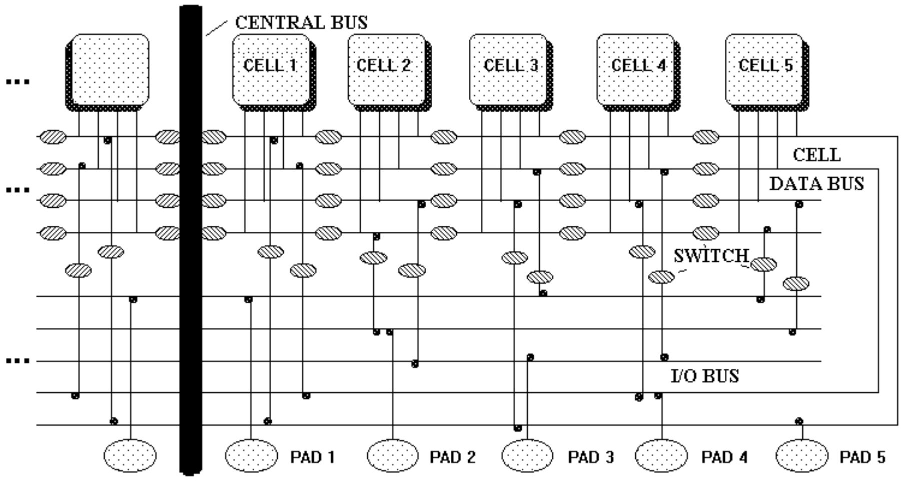


Figure 3 -- Basic architecture of a group of 5 logic cells and 5 I/O Pads placed side by side in a row.

is set to zero. That counter will count the number of global clock pulses applied to the chip.

The loading process of the configuration data is bit-serial, according a specific order. The process will be completed when the counter value is equal to the amount of data to be loaded. Thus, the control circuit resets the logic cells internal flip-flop and the hole matrix is able to work precisely.

Once the bit-serial process was chosen, only two pins are necessary: 1 input pin, and other one to the synchronization of the operation. This configuration facilities the global operation of the matrix and allows an economy of circuit Pads. Some serial CMOS PROMs commercialized by XILINX [13], as XC1736 of 36.288 x 1 bit and XC1765 of 65.536 x 1 bit, can be used to store the FLECHA matrix configuration data.

3: FLECHA matrix control circuit

3.1: Control circuit general configuration

The chosen methodology to load the configuration data originates the necessity of an internal control circuit, responsible to generate internal clock pulses during the charge process and normal operation of the FLECHA matrix. This control circuit is responsible to supervise the charge of the configuration data, and, after that, to reset the internal flip-flops of the logic cells and to perform clock pulses, allowing the implemented device to work adequately. An external reset pin is applied to the control circuit. After the control circuit is adequately powered, the reset reference voltage goes to 5V, driving the control circuit to initiate its operation.

3.2: Clock pulse generation

Only 4 external pins are necessary: 1 pin for connecting the external clock; 1 reset pin; 1 for loading data from the external memory; and 1 external memory enable pin. After the reset command, the control circuit resets its counter and enables the external memory to transmit the configuration data. Then, the counter starts to count the amount of received data and, after that all data is stored correctly in the internal shift-register chain, the control circuit disables the external memory, resets logic cells flip-flop; and enables the logic cell array to start normal operation.

Both shift-registers and flip-flops of the cells need 2-phase clock signals to work. Therefore, control circuit has to generate clock signal and its complement to the internal registers. Figure 4 presents the control circuit, the external signals ***CLK*** and ***RESET***, and the generated internal signals: ***MENABLE***, the internal reset signal ***RST*** and clock pulses ***CK1 - CK2*** e ***CKFF1 - CKFF2*** to shift-registers and flip-flops, respectively.

Figure 4 presents the block responsible to divide the original clock signal: ***CK1 = CLK*** e ***CK2 = NOT (CLK)***. A switch was applied to the clock generation block of the shift-registers to keep a stable state (***CK1*** low and ***CK2*** high) when it is necessary. The output circuit of this block is a powerful buffer. A similar circuit was used to generate ***CKFF1*** and ***CKFF2***, to the logic cell flip-flops.

3.3: Design of the FLECHA matrix control circuit

Figure 4 also presents the internal counter, built with 10 basic counters (D-type flip-flops). This 10-bit counter can count up to 1024 clock pulses. The external signal ***RESET*** sets the counter to zero. The counting process is stopped when bit 9 turns to 1, thus, ***CK1*** and ***CK2*** are interrupted. After a short time, the internal reset signal ***RST*** goes high, releasing logic cell flip-flops and the control of

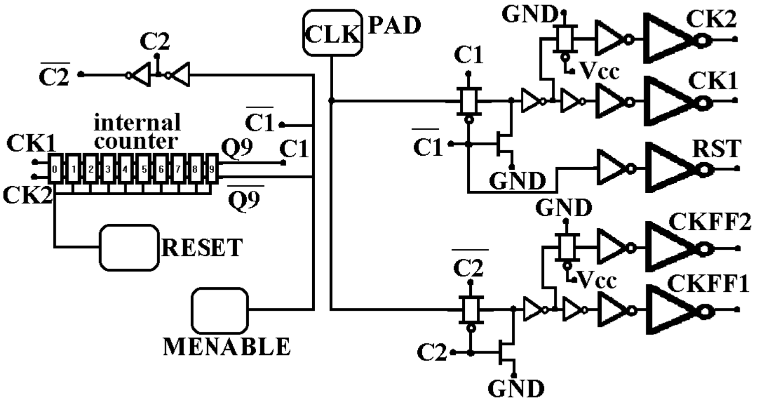


Figure 4 -- Representation of the control circuit and its internal structures.

the block that is responsible for ***CKFF1*** and ***CKFF2*** generation. FLECHA matrix can now initiate its normal operation and the control circuit will maintain this situation until the activation of the external reset command, imposing a new configuration data loading cycle.

A creative solution was adopted aiming the simplification of this circuit, with the elimination of a decoder, responsible to detect the exact moment when the counter reaches the number of configuration bits stored at the shift-register chain. In that case, the counter will ever count up to 1024, once the external memory is loaded with 0s at the first 248 bits. The falling 776 bits will contain the necessary configuration data. It occurs because FLECHA matrix has 776 shift-registers to memorize its internal functions, thus, it is necessary to transmit exactly 776 configuration bits at power-up. The implemented solution provokes the charge of 248 unnecessary bits. This fact does not cause major problems, because the first transmitted data will be abandoned by the shift-register chain when the next 776 arrive. The delay of the configuration data charge will be quite bigger: 10.240ms, in comparison to the necessary 7.760ms.

3.4: Control circuit implementation

The implemented solution is very useful, because allows future modifications in the matrix. If the number of shift-registers is altered, the control circuit does not need to be modified. Only the amount of initial 0s will need to be changed by the software tool. A new flip-flop has to be add to the counter just if the number of shift-registers of the new matrix is greater than 1024. If it is necessary, a new matrix with up to 2048 configuration bits can be built.

The final layout of the control circuit has 182 transistors and final dimensions of 96 x 554mm. The performance of the control circuit is good, once it is capable to load all configuration data in 10.240ms, at matrix power-up.

3.5: FLECHA matrix design software

The main contribution of this research is not restricted to the presentation of an innovative prototype matrix, technically suited to special slices of the market, since the FLECHA project will continue with the development of a software environment of digital circuit design. This environment is constituted of a state-of-the-art placing and routing methodology based on Neural Networks technology [3] and an object-oriented user interface. The initial experiments with the Hopfield model [10] showed us great possibilities in reducing extremely the silicon waste in normal FPGA matrixes due to inadequate routing policies.

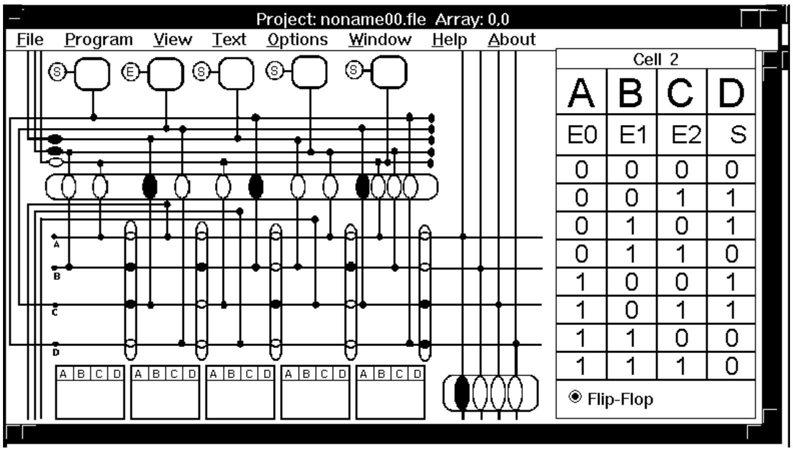
The initial results obtained with neural techniques have improved from 50% to 90% the utilization capacity of the FLECHA matrix structures, in some cases. Figure 5 shows the graphic interface of the software, where the user can specify the circuit to be implemented, test it, and generate the configuration pattern of the implementation.

4: Performance analysis

4.1: Density

The 48 pin DIL-type package was chosen because of its good relation between cost and available pins. This package allows the design of a matrix with 40 logic cells and 40 I/O Pads, performing 600 equivalent gates. Any Boolean function can be implemented in each logic cell, where a large number of registers are provided, and any necessary routing paths can be defined between the logic cells and the I/O Pads.

The evaluation of the number of equivalent gates of an FPGA is very difficult to be founded, because there are many electrical structures in the internal architecture that are not used to implement applications [6]. To specify the logic capacity of the FLECHA matrix, many circuits were


Project: noname00.fle Array: 0,0
File Program View Text Options Window Help About
S
E
S
S
S
A
B
C
D
A B C D
A B C D
A B C D
A B C D
A B C D

Cell 2

A	B	C	D
E0	E1	E2	S
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

● Flip-Flop

Figure 5 -- The object oriented user interface of the design software environment.

implemented to allow comparisons with commercial architectures. Table 1 presents some of the results.

Table 1 -- Number of required logic cells to implement typical logic functions

Logic Function	Logic Cells
1 of 8 Decode with 3 Enables (74138)	12
8 to 1 Multiplexer with Enable and Complementary Output (74151)	14
Serial/parallel 8-bit Shift-Register With reset (74164)	16

A good efficiency was obtained with the development of FLECHA matrix: any Boolean function can be implemented with each logic cell, a great amount of registers is available, and different routing paths can be generated with the internal interconnection system.

The falling analysis is related to two different commercial products: XC3042 of XILINX [13], containing 144 logic cells and 96 I/O Pads; and EPM7032 of ALTERA [1], with 32 logic cells and 36 I/O Pads.

Many circuits, dedicated to interface microprocessors, memories, and device controllers, were implemented to perform this comparison. These circuits were used to fit the three matrixes, allowing the evaluation of the FLECHA matrix equivalent gates. Table 2 shows the number of circuits that is possible to implement with each matrix (Cells/Pads), considering first, the number of necessary logic cells (Cells), and second, the number of I/O Pads (Pads) used by the circuits. These examples have proved the FLECHA matrix efficiency, because it has the best   It is possible to conclude, from the implementation of these circuits, that each considered FPGA has its own characteristics, and this fact improves their performance according the necessities of different applications. FLECHA matrix has the best relation between the number of logic cells and I/O Pads. This characteristic improves its performance with glue logic applications, once it is necessary to consider the great number of signal exchanges. relation logic cells and I/O Pads.

Table 2 -- Comparison between the 3 FPGAs utilized to implement glue logic application

Implementation (Cells/Pads)	FLECHA	XILINX XC3042	ALTERA EPM7032
Circuit 1	4x / 3x	36x / 7x	4x / 2x
Circuit 2	5x / 3x	36x / 7x	4x / 2x
Circuit 3	5x / 4x	48x / 9x	10x / 3x
Circuit 4	10x / 4x	48x / 10x	32x / 4x
Average Values	6x / 3.5x	42x / 8.25x	12.5x / 2.75x

The XILINX matrix has high-capacity 5-input logic cells, that are able to implement up to 2 functions with the same input. This fact leads to a best performance when applications present big logic gates, with the same inputs.

The ALTERA EPLD showed up high efficiency implementing with only one logic cell, by sum of products, many logic gates with a large number of inputs, but needing just one output Pad.

4.2: Operational frequency

Considering a *speed grade* based on the maximum toggle rate of the internal flip-flop of a single logic cell [6], the matrix operational frequency, due to its simplified internal structures, is about 145 MHz, once a single logic cell delay is less than 6.0 ns. The external frequency is often smaller due to the delay of the I/O Pads. With the FLECHA matrix Pads [5], this external frequency is 66 MHz.

The maximum system clock frequency, however, depends on the application and its implementation on the matrix. The number of concatenated logic cells in each specific logic function determines its delay. Table 3 presents some critical timing parameters of FLECHA matrix related to the XILINX 70 MHz XC3000 family [13][6]. This comparison shows the high speed of the simplified structures of the FLECHA matrix.

Table 3 -- Analysis of logic cell(LC) typical timing parameters - worst case (ns)

Parameter Description	XILINX XC3000	FLECHA Matrix
LC Functional Block Combinatorial Delay	9	3.6
LC Register -- Clock to Output Delay	8	3.3
I/O Pad -- Direct Input	7	4.45
I/O Pad -- Output Buffer (50 pF)	10	15.3
LC Input to Output Combinatorial Delay	17	6.9

5: Conclusions and future trends

FLECHA MATRIX presents a reasonable logic capacity of 1.000 equivalent gates, considering the obtained low cost of silicon area and package, without loosing speed related to its operational frequency of 145 MHz. The high performance of the logic cells is due to the combination of basic architectural simplifications and original design techniques, like the lateral placement of the logic cells.

The circuit was implemented in a 44-pin package, a cheap solution that can implement many applications where the other solutions are not well suited. Its high speed allows the utilization of the FLECHA matrix in fast communication systems, like high speed microprocessors, for example. The internal delay of signal propagation, from the input to output of one logic cell, is 6.9ns and the external frequency, due to the delay of the Pads, reaches the 66 MHz mark. The architecture of the programmable I/O Pads permits the separation between the circuit of the Pad (buffer and filter) and its configuration structure. Because of that, the Pads can be redesigned separately, if a new process technology becomes available.

Many circuits have been implemented to perform an evaluation of the programmable structures of the matrix and test the advantages of the lateral placement technique. These examples have proved the functionality of this architecture and pointed a reasonable coefficient of logic cells utilization (80%), despite its simplified switching system.

The proposed switching system is very simple and efficient, allowing complex function routing in the matrix with low waste of logic cells, and its reduced number of switches in the path of logic signals permits a great operational speed. This system allows a large reduction of the circuit area.

Other interesting point to be considered is its simplified control circuit. The internal counter does not need a decoder to accuse the end of the configuration data loading process. This solution permits the development of new matrixes, having different logic densities, without the necessity of changing the control circuit. It is only necessary that the software tool modifies the number of initial 0s, added to the data file.

The future versions will be designed with more cells in the array (an 80-pin package matrix is available) and an internal flip-flop bank, that aims to improve the capacity of sequential logic implementation, without the utilization of the logic cells internal register.

6: References

[1] Altera Corp. Mask-Programmed Logic Devices. In: Product Information Bulletin. Santa Clara, CA: Jan 1992. ver. 1, n. 14.

[2] Actel Corp. ACTEL Logic Optimizer. Sunnyvale, CA, 1991

[3] Ahrens, M. An FPGA Family Optimized for High Densities and Reduced Routing Delay. In: Custom Integrated Circuits Conference, 1990. p. 31.5.1-31.5.4.

[4] Chen, X.; Hurst, S.L. A Consideration of the Minimum Number of Input Terminals on Universal Logic Gates and Their Realization. Internal Journal of Electronic Testing: Theory and Applications, Hinghan, MA, v.50, p.1-13, 1991.

[5] Dossa, M.K. Digital CMOS 1.5um Pads Library. Porto Alegre: CPGCC/UFRGS, RP n. 162, version 1, Set. 1991. 47p.

[6] Hung-Cheng, H.; Dong, K.; JA, J.Y. A 9000- Gate User-Programmable Gate Array. In: IEEE Custom Integrated Circuits Conference, 1988, San Jose, CA. XILINX Inc.

[7] Hurst, S.L. A Survey of Published Information on Universal Logic Arrays. Microelectronic and Reliability, Oxford, England, n.16, p.663-674, 1977.

[8] Lee, H.L.; Sim, B.L.; Samudra, G.S. A Study of Interconnect Capacitance Correlation with a 2-D Capacitance Simulation. In: International Symposium on IC Techinology, Systems & Applications, 5, Singapore. 1993. p.425-429.

[9] Rose, J.S.; Brown, S. Flexibility of Interconnection Structures for Field-Programmable Gate Arrays. IEEE Journal of Solid-State Circuits, New York, NY, v.26, n.3, p.277-282, Mar. 1991.

[10] Shih, P.H.; Chang, K.E.; Feng, W.S. Neural Computation Network For Global Routing, Computed-Aided Design. Oct. 1990.

[11] Simões, E.V.,Uebel, L.F., Barone, D.A.C.: Fast Prototyping of Artificial Neural Network: GSN Digital Implementation. IV International Conference on Microelectronics for Neural Network and Fuzzy Systems, Torino, Italy, Sep. 1994.

[12] Yan, N.; Lim, Y.C.; Samudra, G. et al. A Parallel Clustering Approach on General Connectivity and Its Application to VLSI Placement. In: Internal Symposium on IC Technology, Systems & Applications Singapore, 1993. p.185-190.

[13] Xilinx Inc. The Programmable Gate Array Data Book. San Jose, CA, 1993.