IMPLEMENTING A FUZZY PROCESSOR ON PROGRAMMABLE LOGIC CIRCUITS: MODULARIZATION CRITERIA

Esperanza M. López
E.T.S. de Ingenieros Industriales
Universidad de Murcia
Cartagena. Murcia
espe@plc.um.es

Alfonso Aniorte Carbonell
E.T.S. de Ingenieros Industriales
Universidad de Murcia
Cartagena. Murcia
aac@plc.um.es

Ramón Ruiz Merino
E.T.S. de Ingenieros Industriales
Universidad de Murcia
Cartagena. Murcia
ramon@plc.um.es

Summary

In this communication, we propose a scheme for implementing a fuzzy processor on programmable logic circuits (FPGA), utilizing pipelined structures. We have partitioned the processor structure in several sections, which correspond to different integrated circuits. Such I.C. carry out the operations inherents in a fuzzy process (fuzzifiers, inference engine, defuzzifiers) in a very efficient an flexible manner maintaining a low cost per prototype.

Keywords: FPGA, Fuzzy Processor, ASIC, Digital Design.

1 INTRODUCTION

The need of making compatible the calculation requeriments that Fuzzy Logic imposes, with the limitations in response time of certain applications, in particular the control of processes, implies almost unavoidably to find alternatives to the conventional calculation systems, which are not quite adequate for the efficient treatment of the operative schemes that arise of the linguistic variables type. Fortunately, the modularity and the simplicity of the fuzzy processing algorithms which are adequate for these applications, permit their efficient implementation on dedicated hardware structures [2]. The problem is usually to decide how much flexibility (measured in terms of the [programmability] of the different blocks that constitute a fuzzy system) should be supported in a dedicated fuzzy device. Factors like the speed, complexity, restrictions of response time and interaction requirements with not fuzzy algorithms, will influence finally on the decision [1].

The implementation of digital systems based on regular structures of gates or, more in general, of cells, has revolutionized some conventions of the digital logic and arithmetic. Viewed as specific calculation units, their implementations provide excellent performances in the majority of the applications, providing true generic hardware.

Within this methodological alternative for implementing of electronic digital systems, the introduction by the middle of the 80's of the Field Programmable Gate Array (FPGA) meant a milestone in the field of the design of application specific integrated circuits (ASIC). In fact, at the present time their technology makes them competitive in both complexity and versatility as opposed to other ASIC alternatives for direct implementation on silicon (standard cells or gate array). On the other hand, these circuits present clear advantages in terms of: time of development, flexibility, and specially, cost of the prototypes. To sum up, even though a full custom implementation on silicon usually surpasses the capacity of a generic programmable system and reaches higher speed, very few applications deserve the cost of creating a specific integrated circuit, being more and more attractive the solutions based on FPGA.

We present in this work the general structure of a configurable fuzzy processor, developed to be implemented on several FPGA circuits. Given the flexible nature of this implementation style, the proposal that we make does not constitute a closed solution; rather it pretends to open perspectives in both the modularization of the design and the possibilities of pipelining of the operation, starting from a library of constructive blocks which allow to construct multi-FPGA, specific processors adapted to the dimensions and particularities of the fuzzy computation problems. We have intended to evaluate the possibilities of this methodology of implementation, in a field so demanding and mature as that of the dedicated circuits in fuzzy logic. It will be necessary to take advantage of the characteristics of flexibility, modularity and parallelism of the modern programmable devices in order to achieve competitive implementation features for speed and complexity

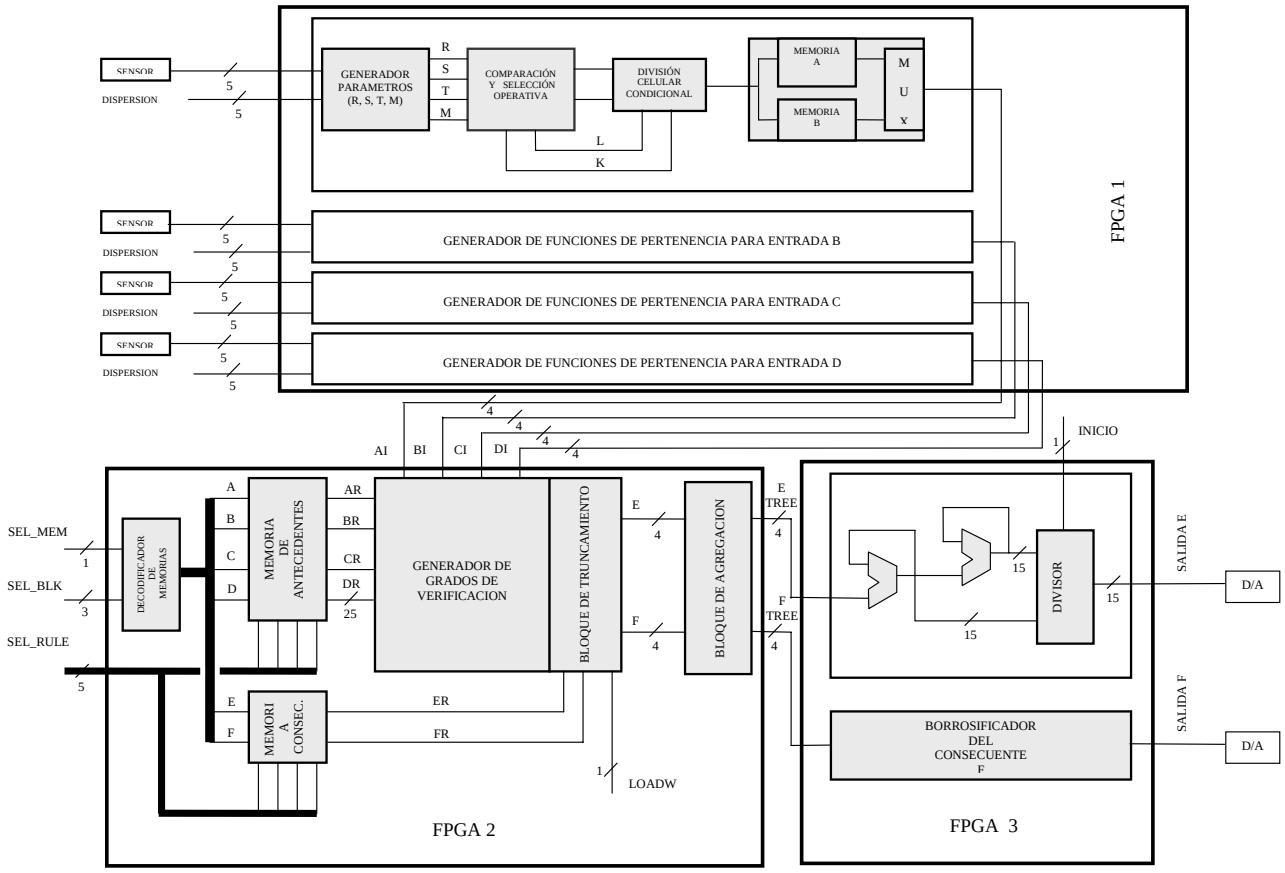

Figure 1: General block diagram of the fuzzy processor.

2 MODULAR STRUCTURE OF PROCESSOR

The processor adjusts to the classical structure, in which parallelism is introduced by means of a pipeline of five stages. The main elements of the architecture are summarized in the general block diagram shown on fig. 1. In this figure we can distinguish three main modules, that correspond to the physical blocks in which we have divided the processor. This modularizationn of the design in physical blocks differentiated in three levels, together with the characteristics of configurability introduced in certain sections of the design, confer the proposed architecture a high grade of flexibility, allowing several options with regard to the number of inputs, number of rules and number of outputs. In this way, the block diagram of the fig. 1 (one integrated circuit by level) only corresponds to the simplest configuration, adequate for systems of rules consisting of 4 antecedents (or 2 in the alternating configuration of structure of rules) and 2 consequents (or 1), containing 25 rules (or 50 when we process rules 2 to 1).

2.1 MODULE OF INPUT GENERATION

This module carries out the translation of the variables measures from of the sensors in possibility distributions consisting of 32 values quantified in 4 bits; such possibilities are the inputs to the inference process. The characteristics of these triangular distributions are determined by both the measurement of the sensor and the value of dispersion that is introduced in order to characterize the uncertainty in the measurement of each input. Both values are quantified in 5 bits.

In order to produce the distributions, we have designed a generator of possibility values that avoids the demands of memory of the systems based on Look up Tables [3], that could constitute a problem when implemented on FPGA, given its limitations as for available RAM. The generation of the membership values is carried out starting from the calculation of four parameters (fig. 2),which are introduced in a module of comparison and selection which determines the function to be applied in each case. In the intervals where it is necessary to carry out a division (to calculate the

lineal interpolation), this is performed by means of a cellular unit whose characteristics of regularity and local connectivity favor their implementation, providing some excellent speed capabilities. In the cases of absence of dispersion in the measurement, the need for these operations of division will are eliminated.

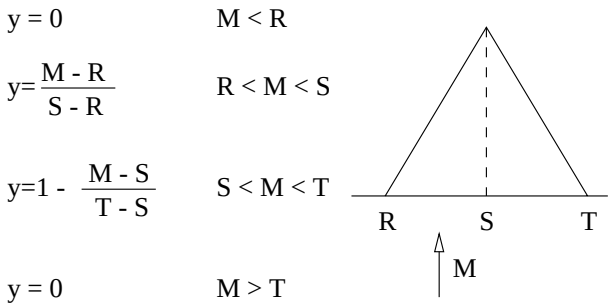

Figure 2: Parameters of the membership values.

The possibility values obtained in this way are stored in one of the two halves of a memory area, from which they will be introduced in a later phase into the module of rules to carry out the fuzzy inference. The inclusion of a double memory structure, where reading and writing of data are alternatively switched, allows us to introduce a first pipeline stage in the controller operation. So, as long as the A half receives the data of the generator in a certain time interval (writing), simultaneously the B half will supply the possibility values (reading) of the previous measurement to the inference.

The moderate complexity of this module allows their implementation in a FPGA of medium size, of the type XC4010 (10000 logical gates and 12800 bits of RAM). This circuit will be occupied about 90%. Concerning to the frequency, the maximum delay in this module corresponds to the structure of division (170 ns). This delay determines the maximum operation frequency of the system.

2.2 MODULE OF RULES (INFERENCES)

In this module the distributions resulting from the input evaluation through the set of rules are calculated. Such rules can be configured as 4 antecedents and 2 consequents rules, or alternatively as 2 antecedents and 1 consequent rules; in the first case to a maximum of 25 can be stored, and in the second 50 ones. The rules are stored in a RAM memory, value by value, so that the total dedicated area to this purpouse is 25 x (4+2) x 32 4-bit words (4800 x 4).

The calculation of the verification degrees among the inputs and the antecedents of the rules is carried out by means of a direct implementation of the Mamdani's algorithms; inputs are compared in parallel to all the antecedents of each rule. The minima of the membership functions are extracted sequentially element by element and then the maxima of this series are stored in registers. In a later stage of the pipeline these degrees of verification will truncate the consequent distributions, passing these truncated distributions to an aggregation tree (MAX), that will produce the output distribution associated with each consequent. The aggregation is made up as a strictly combinational function, and it contains the critic path which determines the maximum operation frequency of the device. This frequency is higher than the one imposed by the module of input generation. The distributions resulting from the aggregation block constitute the input to the following module.

Such a method for storing rules permits a high degree of freedom in the definition of antecedents and consequents distributions. Concerning to the number and structure of the rules, we also have a high versatility; the possibility of chaining rules could be effective just applying the output from one of these modules to the input of circuits a second level. Concerning to the addition of in rules, the disposition of circuits in parallel connected to the same module of generation of inputs will proportion this possibility, being necessary a minimum amount of logic (MAX modules) in order to complete the aggregation tree; such logic can be included within any of the FPGA of this level, as their degree of occupation is not too high. In order to implement these modules, the programmable devices we must use have to be quite complex. Devices of the type XC4052 permit this implementation, being occupied about the 75%. The XC4052 has about 52000 logical gates, and an available RAM of almost 62000 bits. On the other hand, the maximum operation frequency of this module is higher than the one determined by the module of input generation (about the double). So the frequency of the whole system is not limited by this module.

2.3 DEFUZZIFIER MODULE

The last module contains a double defuzzifier structure that simultaneously processes the two output from the module of rules. For the defuzzification, we use the center of gravity method, what makes necessary a new operation of division. In this module, the division has been implemented according to a sequential algorithm without restoration. The use of a division structure different from that of the module of rules is due to the fact that the operative scheme utilized for the calculation of the center of gravity permits to synchronize the output rate of the module of rules with the displacement/accumulation steps involved in this division al-

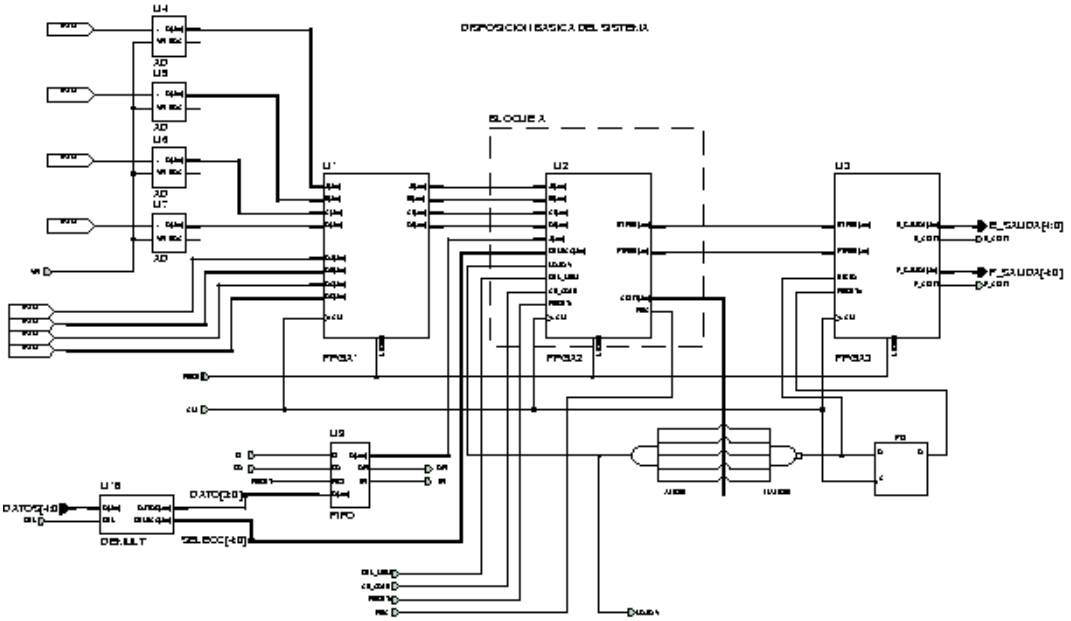

Figure 3: General structure of the controller card.

gorithm. Each of such steps involves a propagation delay much lower than the one of the cellular divisor included in the first module (delay not exceeds 40 ns in this case). This third module doesn't offer problems of complexity, so we can opt again to utilize a FPGA of medium size (type XC4010), that allows to hold the functions of the two defuzzifiers easily.

3 CONTROLLER CARD

The three integrated circuits we have presented constitute a versatile fuzzy processor as a whole, providing and adequate platform to carry out control tasks based on fuzzy rules. However, for this operation it is necessary a certain logic that implements the communication with a ghost computer in order to program the integrated circuits. Complementary logic elements are also needed to configure the integrated circuits and to provide an adequate sequencing of the operations. The fig. 3 represents the disposition of these integrated circuits in a controller card and their connection. The proposal includes the programming of the integrated circuits through a parallel port, that will also serve for storing the rules in the memory of the second integrated circuit.

4 RESULTS AND CONCLUSIONS

The integrated circuits that implement our proposal have been verified using a temporal analyzer, in order to determine the critical path and, consequently, the maximum frequency to which they can operate. The results of this analysis have been commented previously, being the first of the modules who imposes a greater limitation in the operation frequency. A functional analysis has also been carried out. Problems of temporal violation detected in first instance have been solved, and we have verified the correctness of each of the modules as well as and the fuzzy processor as a whole. In summary, we consider satisfactory the results obtained, confirming that the elected approach is a suitable one, competitive in the majority of aspects with ASIC solution of much greater cost per prototype.

References

[1] A. Costa y A. DeGloria (1995). Hardware solutions for Fuzzy Control. *Proc. IEEE* pp 83.

[2] A. Gabrielli et al.(1997). VLSI Design and Realization of a 4 Input High Speed Fuzzy Processor. *6th IEEE Int. Conf. Fuzzy Systems.* Barcelona. July.

[3] H. Watanabe et al.(1990). A VLSI Fuzzy Logic Controller with Reconfigurable, Cascadable, Architecture.*IEEE Journal Solid State Circuits.* n.25.