Design of Very High Speed CMOS Fuzzy Processors for Applications in High Energy Physics Experiments

Davide Falchieri, Alessandro Gabrielli, Enzo Gandolfi, Massimo Masetti

Istituto Nazionale di Fisica Nucleare - Università di Bologna
via Berti Pichat 6/2 - 40127 Bologna Italy
Email: masetti@bo.infn.it

Abstract

We faced the problem of VLSI fuzzy chip design because the processing rate of the fuzzy chips available on the market is too low for trigger applications in High Energy Physics Experiments. When we started, three years ago, we chose the 0.7 micron ES2 technology because it was the fastest one available at an accessible price. The paper describes chip architectures where each rule is processed in one clock period: 20 ns. Our goal was a processing rate of few hundred ns: it was possible to reach such a result only when the number of the inputs is less or equal to four: for two inputs the processing rate is 80 ns while for four inputs 320 ns because only the active rules are processed. For more inputs a genetic rule generator is used because it allows to have a fuzzy system with a very low number of rules: for most applications the number of rules is about 10. The architecture of a 10 input fuzzy chip able to process a "genetic fuzzy system" is described. The above chips have been constructed and now we have redesigned the two input fuzzy chip using the 0 .35 μm technology of the Alcatel Mietec to reach a higher processing rate.

1. Introduction

Neural chips are now used in trigger devices for HEPE [1]. Four years ago we faced the problem of using also fuzzy chip microprocessors [2], because a fuzzy system can work as a neural system and is more flexible [3]. We made then a comparison between the neural and fuzzy approaches reaching the following conclusions:

- fuzzy chips running at a speed suitable for trigger devices are not available on the market [4], [5], therefore one has to design its own VLSI chip while, for the neural solution, one can use commercial chips;
- to develop the fuzzy system, that is the rules and the Membership Functions (MF) related to an application, there are two possibilities: an expert can develop it or SW tools can be used. These are called Rule Generators and are able to develop a fuzzy system related to a specific application by means of Neural Network (NN) [6] and/or Genetic Algorithms (GA) [7];
- the fuzzy solution is more flexible because you know its knowledge basis that is the fuzzy system and you can improve on-line the performances by changing the rules. That is not completely true when a fuzzy system is obtained by a rule generator.

To design a very high speed fuzzy chip we have:

- to select a very advanced CMOS technology: three years ago we used the 0.7 μm CMOS technology and the ES2 foundry standard cells;
- to design a pipeline architecture: 20 ns delay for each pipeline stage with a 50 MHz clock;
- to implement very fast algorithms like the Sugeno inference and defuzzification method [8] that is the final division process, see figure 1, which takes place in parallel to the pipeline stages.

To further improve the processing rate we followed two different solutions:

- for a fuzzy processor with no more than 4 inputs and each input with no more than 7 fuzzy sets (FS) it is possible to process only the active rules. Following this approach we designed and constructed a two input fuzzy processor running at a rate of 80 ns [9], with a 14 square millimetre area and a 4 input fuzzy processor running at a rate of 320 ns, with a 60 square millimetre area [10];
- for more than 4 inputs we designed a 10 input fuzzy processor matched to a fuzzy system obtained by a genetic rule generator where the number of the rules of the fuzzy system is very low. The processing rate is 20 ns times the number of the rules of the fuzzy system to be processed. The chip area is 80 square millimetres. The above realisations can match the processing rate of a second level trigger but now new CMOS technologies, that is 0.5, 0.35 micron by Alcatel Mietec and 0.25 micron by SGS Thomson are available. With these new technologies we are starting to redesign (the previous designs have been made using the VHDL) the above chips with a clock up to 150 - 200 MHz,

therefore they can match the processing rate required by a first trigger level; for example the two input fuzzy chip will run at a rate less than 27 ns with further improvements while the "genetic" fuzzy chip will have a processing rate of less than 7 ns times the number of rules. These chips will be sent to the Alcatel Mietec foundry by the end of this year. In this paper the architecture of the chips are described.

2. The four Input Fuzzy Processor

Here is described, see figure 1, the four input fuzzy processor; the two input fuzzy processor has a similar architecture. The innovative feature of this design is the independence of the processing rate from the fuzzy system. This is obtained in the following way:

- any fuzzy system is converted into a new one where all the rules are present and then loaded in the rule and MF memories,
- an Active Rule Selector (ARS) identifies, without changing the processing rate, the active rules related to each input data set which are to be processed.

The chip architecture involves 12 pipeline stages and each stage takes 20 ns. The main features of the 0.7 micron CMOS fuzzy processor are here summarized:

- four 7 bit inputs, one 7 bit output;
- 7 FSs for each of the 4 input variables, only trapezoidal shapes are allowed;
- maximum overlapping of the input MFs no more than 2;
- 128 crisp MFs for the output variable;
- 4 bits both for the predicates and the premise degree of truth, called respectively α and θ value;
- T-norm implemented by Minimum, (MIN), or Product to obtain the θ value;
- Sugeno order zero [3] inference and defuzzification method;
- 50 Mega Fuzzy Inferences per Second with a 50 MHz clock.

With these features we obtain the following performances:

- *any mathematical function can be approximated with an error of 1%;*
- *each rule is processed in one clock period. The total processing time is:*
  - *number of active rules times the clock period: 16 x 20 = 320 ns* ***plus***
  - *two delays that are due to the number of pipeline stages, 12 x 20 = 240 ns, and to the time required by the division process: 90 ns.*

The main chip architecture is reported in figure 1. The ARS selects the active rules related to the actual values of the input variables; to do that it generates the related addresses for the rule and the MFs shape memories. Then the fuzzification process starts.

The design of the chip has been carried out using the Cadence DFWII Software obtained via Europractice and using the

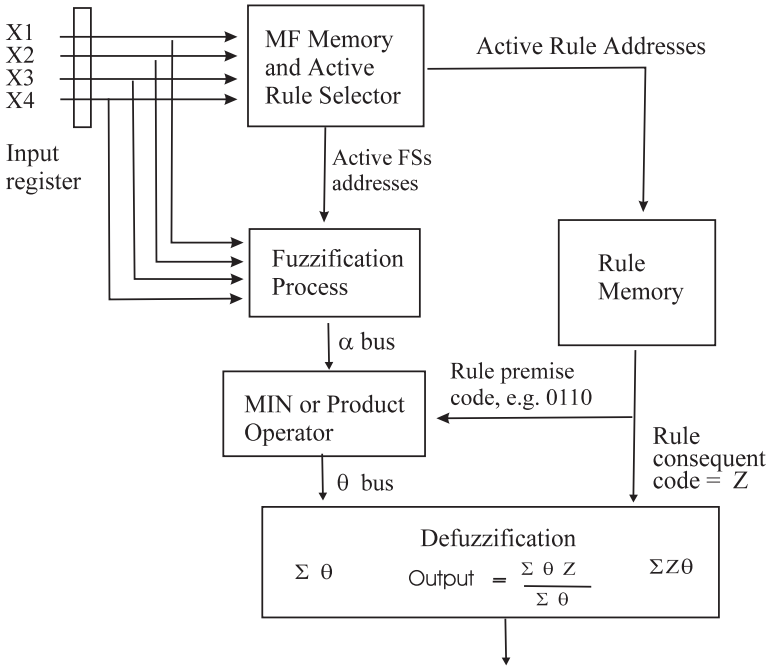

Figure 1. Fuzzy Chip Architecture

Figure 1. The fuzzy chip architecture

VHDL language. The block diagram of the chip layout is reported in figure 3 a), while the chip actual layout is reported in figure 3 b).

2.1. The selection of the active fuzzy rules

To understand what an active rule is let us suppose to have a fuzzy system with 2 input variables, 3 FSs for each input with overlapping of the MFs not higher than 2. A typical fuzzy rule for the above fuzzy system is like:

if (X0 is Low) and (X1 is Medium) then (Z is High).

You can see in figure 2 that only the four rules where X0 is related to L or M FSs and/or X1 to M or H FSs give a non zero contribution to the final result, that is the α values are different from 0: these rules are called active rules. In figure 2 is sketched the α value which is the degree of truth of the predicate X0 is Low. If we have N input variables, K FSs for each input variable, only t-norm operator for the rules and at most an overlap of 2, the active rules are 2^N while all the possible fuzzy rules are K^N. Therefore, for the above example, we can process only 4 rules instead of 9 for a fuzzy system made of all the rules.

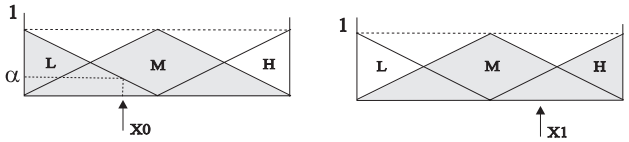

Figure 2. An input data set and the the related MFs

Our 4 input fuzzy processor is able to process only the active rules, which are 16, between all the possible ones, which are 2401; to obtain this result our solution requires:

- to store a fuzzy system made of all the possible rules in the fuzzy chip;
- to use the address code to identify the rule premise.

Then we store the rules starting from the address 0000 where it is stored the rule if (X0 is Low) and (X1 is Low) and (X2 is Low) and (X3 is Low) then (Z is ...) and at the address 0001 the rule if (X0 is Low) and (X1 is Low) (X1 is Low) and (X2 is Low) and (X3 is Medium) then (Z is ...) and so on. The rule code consists of two parts: the first one, related to the premise, where zero means that the related variable is not present in the rule while 1 means the vice versa and the second one, related to the consequent, reports the crisp output fuzzy set. In this way the rule address defines the fuzzy sets of the premise while the rule code confirms or not if the rule was present in the initial fuzzy system (the one without all the rules) and which input variables and output FSs were involved. In figure 1 the premise code 0110 means that the active rule involved comes from a rule in the former fuzzy system where only X_2 and X_3 are involved while the consequent code identifies the crisp value of the output FS. If we had 0000, as a premise code, it would mean that there is not a corresponding rule in the initial fuzzy system, therefore its contribution is zero.

2.2. The chip architecture

The input data set has to be loaded, see the input register in figure 1, into the chip according to the input synchronization handshake signals. The input data set can be ready at any time depending on the external device that generates it: the inputs can be loaded at a maximum rate of 320 ns. The chip architecture is made of the following blocks:

2.2.1. The MF Memory and the Active Rule Selector. In this block there are four memories where the beginning and ending points of the 7 MFs related to the FSs for each variable are stored. Then the ARS can select the two involved MFs related to each input variable. As soon as these FSs are identified another circuit is able to compute the address code where are stored the MF shapes - four points for each trapezoidal MF - and the related active rules. These two addresses are sent to the fuzzification and rule memory blocks.

2.2.2. The Fuzzification Process. In this block there are four memories where are stored the four points related to each MF (the MF shape is trapezoidal) and a MF generator creates the shape in such a way that a four bit α value can be computed for each input value. This circuit receives both the addresses related to all the active MFs and the input data set then computes the related α values. This four values are sent to an operator to compute the related premise grade of truth θ as the MIN (minimum) or Product of the α values.

2.2.3. The Inference and Defuzzification block. This block receives the θ value and, from the rule memory, the consequent code, that is the crisp 7 bit Z_i value and it computes the $\Sigma\,\theta_i$ and $\Sigma\,\theta_i Z_i$ operations. After the last rule is processed a new input data set can enter the chip and the division process starts in order to compute, in parallel to the pipeline processing, the Z_o output value:

$$Z_o = \Sigma\, Z_i\theta_i \,/\Sigma\, \theta_i$$

3. The pipeline timing

We will describe now what elaboration takes place, clock by clock, in each pipeline stage, see table 1, where two processing steps are computed in parallel:

- at the first clock a new input data set is loaded in the input register, this process is off line to the pipeline processing;
- the ARS requires one clock to generate the code of the active MFs to be sent to the fuzzification process and two clocks to compute the premise and consequent code of the

Pipe #	2 Parallel pipelines: processing goals	
Off pipe	Input Data Set Read	*Parallel pipeline processing*
1	Selection of the active MF for each input variable on the base of the input data set	
2	MF memories address generation	
3	MF memories read cycle	Rule memory address generation
4 - 5 - 6	α computations for the 4 input variables	Rule memory cycle: Z and Premise Rule Code
7	Selection/Rejection of the α on the base of the Premise Rule Code	Z Shift Process
8 - 9	α Minimum and Product computation	Z Shift process
10	Selection of the θ value between Minimum and Product computation	Z Shift process
11	$\Sigma\ \theta_i$ computation	θ*Z multiplication
12	$\Sigma\ Z_i\ \theta_i$ computation	
Off pipe:	Computation of the output value: $Z_o = \Sigma\ Z_i\ \theta_i\ /\ \Sigma\ \theta_i$	

Table 1. Pipelines and off line computation steps

active rule: pipeline steps from 1 to 3, see table 1;
- the fuzzification block requires four clock periods to compute the α values: pipeline steps from 4 to 7;
- the MIN block requires three clocks to compute the θ value, there are three identical circuits which perform the same operation on two input variables: pipeline steps from 8 to 10;
- the defuzzification block requires one clock to compute the $\Sigma\ \theta_i$ and the Sugeno multiplication $Z_i\ \theta_i$ operations and another clock to compute the $\Sigma\ Z_i\ \theta_i$ operation: pipeline steps 11 and 12.
- the division process takes place when the last rule has been processed and runs in parallel to the pipeline stages. It takes 90 ns.

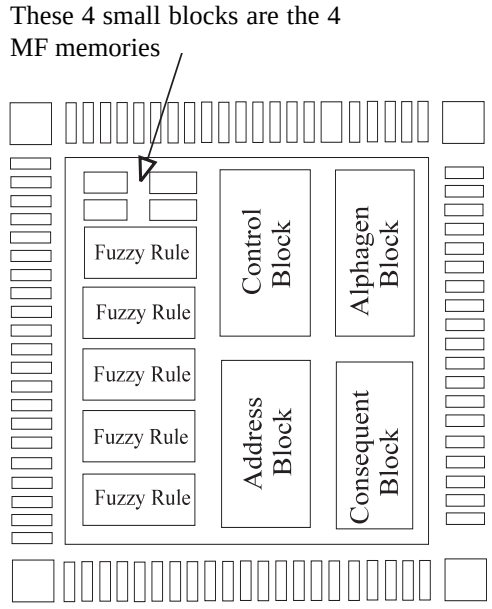

Figure 3 a). The block diagram of the chip layout

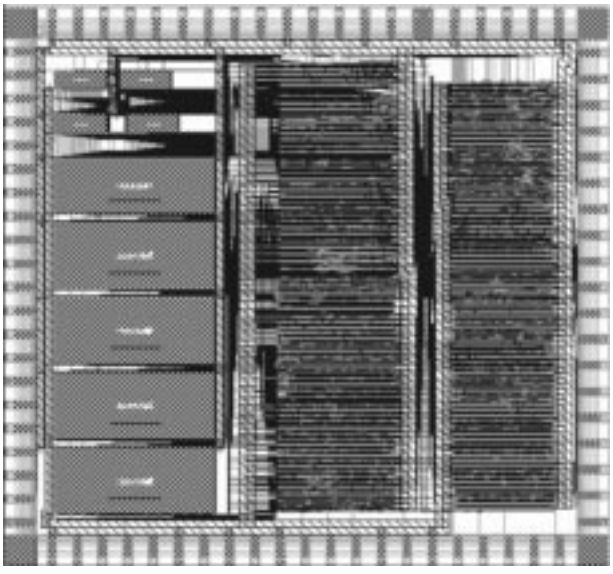

Figure 3 b). The chip actual layout

4. The 10 input fuzzy chip matched to a genetic fuzzy system

The rule structure of a genetic fuzzy system, that is obtained by a genetic rule generator, is completely different from the one of the traditional fuzzy system as each rule has its membership functions. Therefore the fuzzy processor architecture is different: in the previous fuzzy chips we had a rule memory and a MF memory for each input variable, here we only have one rule memory where, in each rule, are coded the related MFs. There is another difference concerning the shape of the MFs which are taken into account. In figure 4 is presented a fuzzy system composed of four rules and the MFs related to each rule are sketched. As a genetic fuzzy system can be made of few rules, our rule memory capacity is made of 60 rules and it is possible to store more than one fuzzy system related to the same input data: our application foresees to have up to four different fuzzy systems that can be selected on line to process the input data. In the figure 4 are shown the input MF shapes that will be approximated by a trapezoidal symmetric shape and the crisp output MFs. The fuzzy processor architecture has been designed according to the following considerations:

- to improve the processing rate it has no meaning to select the active rules because most of the rules are active;
- the number of rules of the genetic fuzzy system is much smaller than the number of rules written by an expert;
- the number of FSs for each variable is equal to the number of rules of the fuzzy system.

The main features of the 0.7 micron CMOS genetic fuzzy processor are:

- maximum number of rules of the fuzzy system: 60 rules;
- up to four different fuzzy systems can process the same input data set (this is required by our application);
- 10 inputs, each of 9 bits, and one 9 bit output;