Hardware Architecture and FPGA Implementation of a Type-2 Fuzzy System

Miguel A. Melgarejo R.
Laboratory for computational intelligence (LAMIC)
Universidad Distrital FJC
Bogotá D.C, Colombia
mmelgarejo@udistrital.edu.co

Carlos A. Peña-Reyes
Logic Systems Laboratory (LSL)
Swiss Federal Institute of Technology at Lausanne, EPFL
Lausanne, Switzerland
carlos.pena@epfl.ch

ABSTRACT.
This paper presents an architectural proposal for a hardware-based interval type-2 fuzzy inference system. First, it presents a computational model which considers parallel inference processing and type reduction based on computing inner and outer bound sets. Taking into account this model, we conceived a hardware architecture with several pipeline stages for full parallel execution of type-2 fuzzy inferences. The architectural proposal is used for specifying a type-2 fuzzy processor with reconfigurable rule base, which is implemented over FPGA technology. Implementation results show that this processor performs more than 30 millions of type-2 fuzzy inferences per second.

Categories and Subject Descriptors
B.7.1 [**Integrated Circuits**]: Types and design styles – *algorithms implemented in hardware, gate arrays.*

General Terms: Design, Experimentation.

Keywords: Type-2 fuzzy logic, interval type-2 fuzzy systems, fuzzy hardware, distributed arithmetic, FPGA.

1 INTRODUCTION
Type-2 fuzzy logic is an emergent research field. It has been demonstrated that type 2 fuzzy systems are capable of dealing with noisy training data, noisy measurements and multiple meanings of linguistic categories [1]. As a matter of fact, type-2 fuzzy logic is a better choice than traditional fuzzy logic for handling phenomena which are both non linear and stochastic [1,2].

Several applications of type-2 fuzzy logic have been recently addressed [3,4]. Some of them require high processing speed so as to operate in real-time. In this sense, it is convenient to implement hardware-based type-2 fuzzy systems.

To the best of our knowledge, currently there is no hardware realization of type-2 fuzzy systems. It has been proposed a computational model and a hardware architecture for an Interval Type-2 Fuzzy Logic System (IT2FLS) [12]. This architecture includes the typical stages of an IT2FLS [1]: fuzzification, inference engine and type reduction, which is based on the computation of inner and outer bound sets, the so called Wu-Mendel closed forms [5].

In this paper, several methodological aspects related to hardware implementation of digital type-1 fuzzy systems are revised [6,7,9] in order to conceive the architecture of the fuzzification and the inference engine stages. For type-reduction, we propose the use of Distributed Arithmetic (DA) to compute the Wu-Mendel closed forms [13]. Due to the inherently parallel nature of an IT2FLS, we consider pipelined parallel organization in order to guarantee the speed performance of the final implementation.

A type-2 fuzzy processor is designed based on the proposed architecture. It is conceived to have up to eight type-2 fuzzy sets per input and nine type-2 singletons sets at the output. It is implemented onto an FPGA (Field Programmable Gate Array).

This paper is organized as follows: Section 2 describes the considered computational model. Our architecture is explained in Section 3. Implementation results are presented in Section 4. We draw conclusions in Section 5.

GLSVLSI'04, April 26-28, 2004, Boston, Massachusetts, USA

2. COMPUTATIONAL MODEL
A type-2 fuzzy system is a rule-based fuzzy system that uses type-2 membership functions to describe its linguistic variables. A fuzzy system is interval type-2 (IT2) as long as any of its antecedent or consequent sets is IT2 [1,2].

Selecting a computational model for processing type-2 fuzzy inferences has a considerable impact on the performance that a hardware implementation could offer. Regarding the block diagram presented in Figure 1, we consider an inference execution model that performs the following tasks:

2.1 Type-2 fuzzification
The first operation in the logic section is fuzzification, which maps a crisp value into a type-2 fuzzy set [1,2]. An example of singleton fuzzification over an IT2 fuzzy set is presented in Figure 2. For each point of the universe of discourse, two membership levels are computed, one called *upper membership value* which is obtained from the *upper membership function*, and the other called *lower membership value* which is obtained from the *lower membership function*. In this computational model, Singleton fuzzification is selected as it is the simplest method. The acquisition of several inputs to be fuzzified, as well as their upper and lower fuzzification can be performed in parallel. As long as the acquisition of each input is independent on the others, they can be acquired and fuzzified simultaneously.

2.2 Inference engine
This stage operates according to a set of rules. The rule base does not depend on the nature of fuzzy sets. Consequently, it is processed exactly as in type-1 fuzzy systems [1,2]. The inference engine combines and computes the rules and gives a mapping from type-2 fuzzy antecedent sets to type-2 fuzzy consequent sets.

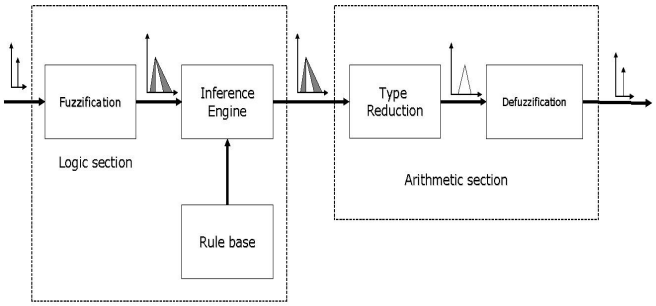

Figure 1. Block diagram of a type-2 fuzzy inference system.

There exists several choices to perform the antecedent and consequent operations required by the inference engine [1]. The inference process for a single rule, shown in Figure 2, proceeds as follows:

1. The upper and lower membership values are t-normed—using the minimum operator—in order to obtain, respectively, the upper and lower firing values, $\bar{f}^i$ and $\underline{f}^i$.
2. The upper and lower membership functions of the consequent fuzzy set are t-normed respectively with the firing values $\bar{f}^i$ and $\underline{f}^i$ so as to obtain an output type-2 fuzzy set.

In our computational model, the antecedent and consequent operations are computed separately for each rule. Here again, a hardware implementation can exploit parallelism as several rules can be processed simultaneously and their firing intervals calculated by parallel computation of the t-norms of the upper and lower membership values.

2.3 Type reduction

Type reduction is obtained by applying Zadeh's extension principle to defuzzification. It represents a mapping of a type-2 fuzzy set into a type-1 fuzzy set [1,2]. In this sense, type reduction searches the best type-1 fuzzy set that represents a type-2 set. It is possible to propose an equivalent type reducer for each defuzzification method.

Type reduction of general type-2 fuzzy sets is computationally expensive. Two algorithmic procedures for computing type reduction have been proposed for IT2 fuzzy sets: the *Karnick-Mendel iterative procedure* [1] and the *Wu-Mendel closed forms*[5].

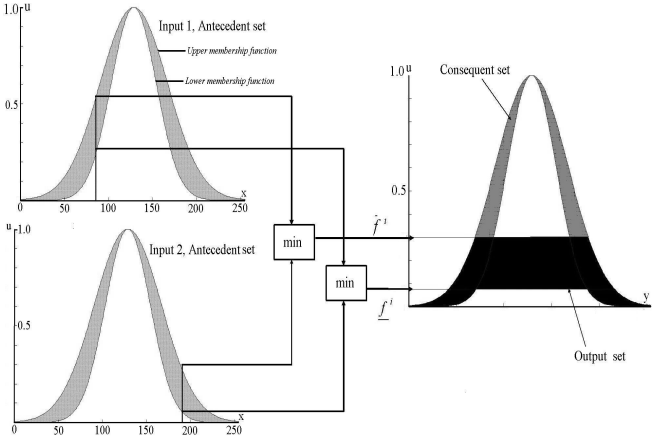

Figure 2. An example of fuzzification and inference process

The former method provides an exact computation of type reduction but it is computationally costly, so it is not well suited for hardware implementation. The latter method finds bound sets for the output type reduced set of an IT2FLS [5]. This computation is performed by means of several closed expressions, so it is a better option for hardware realization.

The result of type reducing an IT2 fuzzy set is an interval type-1 fuzzy set [1,2], so computing type reduction consists on calculating the interval $[y_l, y_r]$. Due to uncertainty, y_r could lie in the interval $[\underline{y}_r, \bar{y}_r]$, while y_l in $[\underline{y}_l, \bar{y}_l]$. Two interval type-1 fuzzy sets can be defined: the *inner bound set* $[\bar{y}_l, \underline{y}_r]$ and the *outer bound set* $[\underline{y}_l, \bar{y}_r]$. The terms y_l and y_r are approximated by:

$$\tilde{y}_l = \frac{\underline{y}_l + \bar{y}_l}{2} \quad (1) \qquad \tilde{y}_r = \frac{\underline{y}_r + \bar{y}^r}{2} \quad (2)$$

The defuzzified point is calculated as the average of $\tilde{y}_l$ and $\tilde{y}_r$. Taking into account that an IT2FLS can be viewed as a collection of *embedded type-1* fuzzy systems [1,5], it is possible to obtain $\tilde{y}_l$ and $\tilde{y}_r$ by computing four of those systems, called *boundary type-1 fuzzy logic* systems. Particularly for height-type reduction these systems are computed as follows:

$$y_l^{(0)} = y_r^{(M)} = \frac{\sum_{i=1}^{M} \underline{f}^i y^i}{\sum_{i=1}^{M} \underline{f}^i} \quad (3)$$

$$y_l^{(M)} = y_r^{(0)} = \frac{\sum_{i=1}^{M} \bar{f}^i y^i}{\sum_{i=1}^{M} \bar{f}^i} \quad (4)$$

The inner bound set is obtained from (3) and (4) as :

$$\bar{y}_l = \min\left\{y_l^{(0)}, y_l^{(M)}\right\} \quad (5)$$

$$\underline{y}_r = \max\left\{y_r^{(0)}, y_r^{(M)}\right\} \quad (6)$$

while the outer bound set is obtained as follows:

$$\underline{y}_l = \bar{y}_l - \frac{\sum_i (\bar{f}^i - \underline{f}^i)}{\sum_i \bar{f}^i \sum_i \underline{f}^i} \times \frac{\sum_i \underline{f}^i (y^i - y^1) \sum_i \bar{f}^i (y^M - y^i)}{\sum_i \underline{f}^i (y^i - y^1) + \sum_i \bar{f}^i (y^M - y^i)} \quad i = 1..M \quad (7)$$

$$\bar{y}_r = \underline{y}_r + \frac{\sum_i (\bar{f}^i - \underline{f}^i)}{\sum_i \bar{f}^i \sum_i \underline{f}^i} \times \frac{\sum_i \bar{f}^i (y^i - y^1) \sum_i \underline{f}^i (y^M - y^i)}{\sum_i \bar{f}^i (y^i - y^1) + \sum_i \underline{f}^i (y^M - y^i)} \quad i = 1..M \quad (8)$$

Our computational model applies height type reduction as it is, the simplest method. Usually y_i is chosen as the point with the highest primary membership value in the upper membership function of the respective consequent set.

If the terms y^L-y^j and y^j-y^1 are considered as the coefficients of linear combinations involved in expressions (7) and (8), there is a coefficient sharing between several of those linear combinations. This is an important fact that should be taken into account for reducing silicon area in parallel-hardware implementation. Even though two dedicated processors are necessary for computing y_l and y_r, they could share hardware components taking advantage of the similarity between the involved expressions.

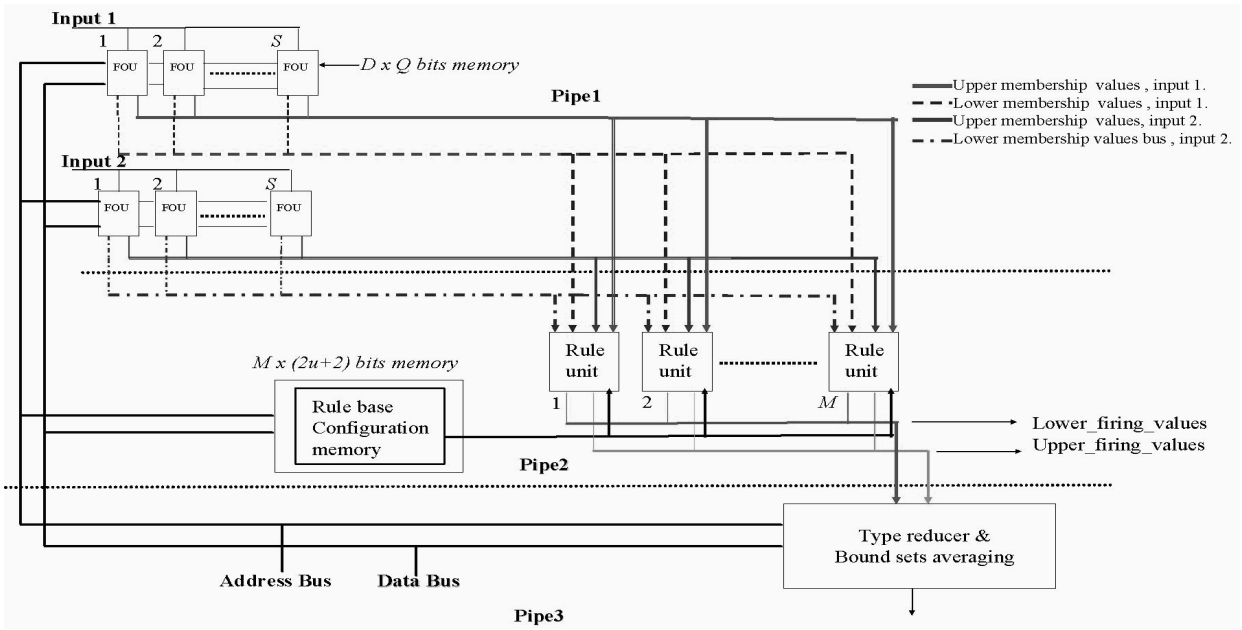

Figure 3. Proposed hardware architecture.

3. HARDWARE ARCHITECTURE

Figure 3 depicts our hardware architecture, conceived for digital hardware implementation, of an IT2FLS. The architecture is specified to have two inputs and one output as well as a rule base of maximum *M* rules. It accepts a maximum of *M* consequent singleton sets and *S* antecedent IT2 fuzzy sets per input. According to the computational model described in the previous section, it is organized as a cascade of three pipeline stages. The first stage consists of the fuzzification units, which perform parallel singleton fuzzification for all antecedent sets. In the second stage, there is an array of *M* configurable rule-computing units, each computing a firing interval. The last pipeline stage computes height type reduction using the Wu-Mendel closed forms. Following, we detail the methodological aspects taken into account to conceive each stage.

3.1 Fuzzification stage

Memory-based fuzzification is the most used computational method in digital type-1 fuzzy processors [6,9]. It allows to implement any membership function shape and provides low fuzzification-computing times being, thus, appropriate for high speed fuzzy processing. However, the depth of the memories growths exponentially with the desired resolution.

Given that most of the recent reconfigurable technologies (i.e. CPLDs, FPGAs, etc) have dedicated memory arrays, we choose a memory-based look up table scheme to perform fuzzification. The width and depth of the memories are selected according to an accuracy criterion [9]. We propose to use one memory (marked FOU in Figure 3) to perform all the entire fuzzification process for one set: each type-2 fuzzy set is represented by means of its upper and lower membership functions. Suppose the memory width is *Q* bits (where *Q* should be a multiple of two), so the values of the lower bound are stored in the less significant *Q/2* bits of the memory, while the upper bound values are stored in the *Q/2* most significant bits.

The values of the two inputs are stored using two registers. The width *W* of those registers depends on the selected memory depth *D*, where $W = Log_2(D)$. Each input is fuzzified by addressing the memories of its associated fuzzification units with the output of the corresponding register. Note that fuzzification for both inputs is performed in parallel over all the antecedents sets.

3.2 Rule-computation units

The rule computation units use minimum t-norm for performing type-2 fuzzy inferences. This minimum t-norm is the most used in digital type-1 fuzzy processors as it is a better option than product t-norm for hardware implementation [6,7]. Indeed, minimum operators demands less hardware resources than multipliers do and they introduce a lower delay in the critical path of the final implementation leading to a better speed performance.

3.3 Type reducer and bound sets averaging

The hardware realization of height-type reduction is the key point of our architectural proposal. It is divided in seven pipelined stages, as shown in Figure 4, producing a total latency of nine clock cycles. The first stage computes the linear combinations required by the Wu-Mendel closed forms. It is divided in five functional blocks, which calculate the linear combinations with shared coefficients and the sums of the upper and lower firing values. Distributed Arithmetic is extended in order to compute these linear combinations [13].

The following stages complete the computation of the closed forms. The boundary systems as well as the right terms of expressions (7) and (8) are computed between pipeline stages Pipe 2 and Pipe 5 (see Figure 4). Here, operations are performed by using fixed point arithmetic, and all components have one cycle latency for full parallel operation. When this kind of arithmetic is used, it is necessary to introduce integer scaling for both divisions and multiplications [11].

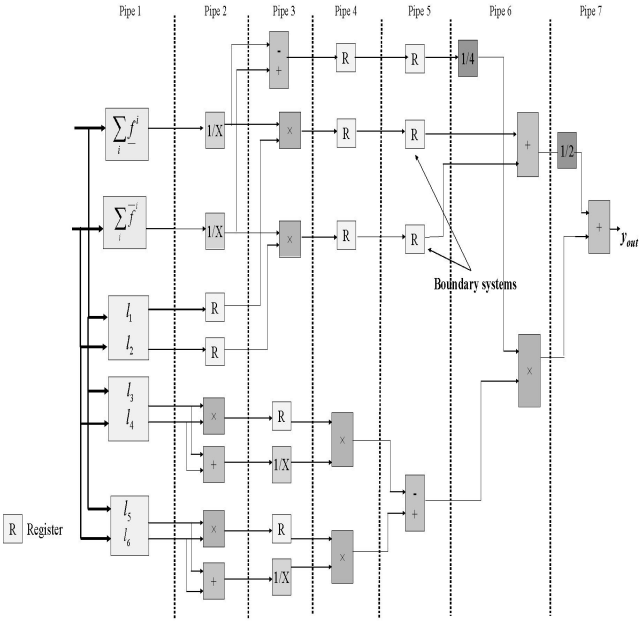

Figure 4. Hardware architecture of the height type reducer.

Table I. Implementation results XC2V3000ff1152-4 FPGA

Parameter	Value
Operation Frequency	33.789 MHz
Slices	2065 (14%)
18Kb RAM BLOCKs Fuzzification DA schemes Reciprocals	56 (58%) 16 24 16
Multipliers (18x18)	56 (58%)
IOBs	44 (6%)

4. A FPGA BASED TYPE-2 FUZZY CO-PROCESSOR

A type-2 fuzzy co-processor was designed according to our architectural proposal. It was specified for two 8-bit inputs, one output (32-bit), a rule base of maximum nine rules and a maximum of nine output singleton sets.

Some performance figures of this processor, referred to the XC2V3000ff1152-4 FPGA, are provided in table I. One can observe the following from the table:

1. Due to the parallel organization considered in the architectural proposal, the speed processing, measured in millions of type-2 fuzzy inferences per second, coincides with the achieved operation frequency.
2. This processor might be implemented into a smaller target FPGA, provided that it has enough memory resources.

5. CONCLUSIONS

We have presented an architectural proposal intended to implement a type-2 fuzzy inference system using reconfigurable hardware.

In a first methodological step, the analysis of an interval type-2 fuzzy system led us to the definition of an inference execution model, whose main features are: (1)division of the inference process in several phases that can be executed concurrently and (2) adoption of the Wu-Mendel closed forms for type reduction, which can be cost-effectively implemented in hardware.

Then, we conceived our architectural proposal taking into account the aforementioned execution model. This architecture has several pipeline stages in which inference processing phases are performed in parallel. We proposed the use of Distributed Arithmetic for implementing type reduction in hardware.

A type-2 fuzzy co-processor was then successfully synthesized and tested over a FPGA target technology. Its processing speed reaches 30 millions of type-2 fuzzy inferences per second.

REFERENCES

[1] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems , introduction and new directions*, Prentice Hall PTR, Upper Saddle River, NJ, 2001.

[2] Q. Liang and J. Mendel, Interval type-2 fuzzy logic systems: theory and design, *IEEE Trans. on Fuzzy Systems*, Vol 8. pp 535-550, Oct. 2000.

[3] Q. Liang, N. Karnik, J.M. Mendel, Connection admission control in ATM networks using survey based type-2 fuzzy logic systems, *IEEE Trans. Systems, Man, and Cybernetics*, Vol 30, pp 329-339, Aug. 2000.

[4] Q. Liang, N. Karnik, J.M. Mendel, Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters, *IEEE. Trans. on Fuzzy Systems*, Vol. 8, No. 5, pp 551-563, Oct 2000.

[5] H. Wu, J. Mendel, Uncertainty bounds and their use in the design of interval type 2 fuzzy logic systems, *IEEE Trans. on Fuzzy Systems*, October 2002, pp. 622–639.

[6] I. Baturone et al.,*Microelectronic design of fuzzy logic based systems*, CRC Press , March , 2000.

[7] G Ascia, V. Catania and M. Russo , " VLSI Hardware Architecture for Complex Fuzzy Systems", *IEEE Transactions on Fuzzy Systems*, October 1999, pp. 553-570.

[8] S. A. White, "Applications of Distributed Arithmetic to Digital Signal Processing: A tutorial Review", *IEEE ASSP Magazine*, Vol. 6 No 3, 1989.

[9] Djuro G., Jaime and Bo, "Hardware implementations of fuzzy membership functions, operations, and inference", *Computers & Electrical Engineering,* Volume 26, Issue 1, 17 January 2000, Pages 85-105.

[10] Kurt Baundendistel, "An improved method of scaling for real-time signal processing applications", IEEE Transactions on Education, Vol. 3, No. 3, August 1994.

[11] *S.F Obermann, M. J Flynn, "*Division algorithms and implementations", *IEEE Transactions on Computers*, Vol. 46, Issue: 8, Aug. 1997, pp 833-854.

[12] M. Melgarejo, C. A. Peña-Reyes, A. García, "Computational model and architectural proposal for a hardware type-2 fuzzy system", *2nd IASTED Conference on Neural Network and Computational Intelligence(NCI 2004*), Grindelwald, Switzerland, February 2004.

[13] M. Melgarejo, A. García, C. A. Peña-Reyes, "Architectural proposal an hardware implementation of a type-2 fuzzy system", *X workshop IBERCHIP IWS 2004*, Cartagena , Colombia, March, 2004.