IMPLEMENTATION OF CMOS FUZZY CONTROLLERS AS MIXED-SIGNAL INTEGRATED CIRCUITS

I. Baturone, S. Sánchez-Solano, A. Barriga, J. L. Huertas.

Instituto de Microelectrónica de Sevilla - Centro Nacional de Microelectrónica
Avda. Reina Mercedes s/n, (Edif. CICA)
E-41012, Sevilla, Spain

*Trans. Fuzzy Systems, (Special Issue on Hardware Implementations)
Vol. 5, n° 1, pp. 1-19, Febrero 1997.*

© 1997 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

This material is presented to ensure timely dissemination of scholarly and technical work. Copyright and all rights therein are retained by authors or by other copyright holders. All persons copying this information are expected to adhere to the terms and constraints invoked by each author's copyright. In most cases, these works may not be reposted without the explicit permission of the copyright holder.

Implementation of CMOS Fuzzy Controllers as Mixed-Signal Integrated Circuits

Illuminada Baturone, Santiago Sánchez-Solano, Ángel Barriga, and José Luis Huertas.

Abstract:

This paper discusses architectural and circuit-level aspects related to hardware realizations of fuzzy controllers. A brief overview on fuzzy inference methods is given focusing on chip implementation. The singleton or zero-order Sugeno's method is chosen since it offers a good trade-off between hardware simplicity and control efficiency. The CMOS microcontroller described herein processes information in the current-domain, but input-output signals are represented as voltage to ease communications with conventional control circuitry. Programming functionalities are added by combining analog and digital techniques, giving rise to a versatile microcontroller, capable of solving different control problems. After identifying the basic component blocks, the circuits used for their implementation are discussed and compared with other alternatives. This study is illustrated with the experimental results of prototypes integrated in different CMOS technologies.

I. INTRODUCTION

At present, most information processing is performed digitally, however, interfacing with the real world remains an analog issue. Besides the fact that A/D and D/A interfaces are essential, analog circuitry can be also found as pre- and post-processors in many applications [1]. In general, analog implementations are preferable for high frequency and low power, although choosing which part of a chip be analog or digital is a complex issue. This reasoning still applies for the specific case of fuzzy circuits, which are becoming an attractive approach to process control. Fuzzy techniques provide a general framework to handle the control of complex processes which are very difficult to describe with mathematical equations but can be controlled by human experts. The often intuitive and ambiguous human knowledge is easily represented by fuzzy rules and can be included into an automated control strategy [2-4].

The application of fuzzy inference techniques to real-time control problems (robotics, image and speech processing, etc.) calls for hardware realizations tailored to the fuzzy paradigm. Many reported fuzzy microcontrollers are digital because of the advantages deriving from sound design techniques, flexible programmability, and easy embedding into digital processing environments [5-7]. However, analog hardware seems to be more suited for the implementation of massively parallel systems that process continuous or multivalued information, like fuzzy systems. Probably, the main drawback of analog implementations is their relatively low accuracy, which does not seem a severe limitation in view of the typical demands of many fuzzy control applications [8-11]. While some analog realizations are designed for specific applications [12-13], we find it more interesting to implement programmable microcontrollers, following the work in [14-16]. Programmability allows the fuzzy chip to be reconfigured to solve different control problems without requiring complex adjusting procedures.

This paper presents mixed-signal transistor-only ICs which combine the best of analog and digital worlds to implement programmable fuzzy microcontroller chips. Typical functionalities are interchanged: the analog circuitry provides the required computing power, while the digital part allows a systematic way to program the fuzzy chip with a simple interface. We focus on CMOS circuits realized with current-mode continuous-time techniques because they are especially suited for direct implementation of the basic fuzzy operators. Working in current-mode, with transistors biased in strong inversion, dynamic range problems can be handled with flexibility and high-speed operation can be attained [17]. The microcontroller described here processes the fuzzy information in the current domain, but to improve the interface capability the I/O signals are represented as voltage, since the majority of existing devices are voltage-processing circuits.

The structure of this paper is as follows: Section II briefly reviews the wide range of possibilities to choose from when designing a fuzzy controller. Studies and results from mathematical and engineering fields are combined with microelectronics criteria to explain the choices made in the presented fuzzy microcontroller. Its ability to provide different control actions is discussed in Section III. Section IV evaluates three types of architecture to implement the chosen inference mechanism (singleton or zero-order Sugeno's method). They are compared in terms of functional block and digital word requirements. The circuits used to realize the different functional blocks are described in Sections V to VII. Experimental results are included to illustrate this analysis. System performance is detailed in Section VIII. The static and dynamic behavior of a programmable two-input one-output controller chip implementing nine rules is illustrated through Hspice simulations.

II. HARDWARE COMPONENTS OF FUZZY CONTROLLERS

A fuzzy controller maps crisp input into crisp output via IF-THEN rules like the following (in the case of single-output controllers):

Rule 1: IF x_l is A_l^I and/or... and/or x_u is A_u^I THEN y is B_l

...

Rule r: IF x_l is A_l^r and/or... and/or x_u is A_u^r THEN y is B_r

where x_i ($i=1, \dots, u$) are controller input, y is the output, and A_j^I, B_j ($j=1, \dots, r$) are linguistic values defined by fuzzy sets on the corresponding universes of discourse. The variables in the IF part are called antecedents, and those in the THEN part are called consequents or conclusions. In essence, a fuzzy controller is formed by two main elements (Figure 1):

- a) An interfacing shell providing the mapping between crisp values and fuzzy sets (fuzzification mechanism) and viceversa (defuzzification mechanism).
- b) A decision making or inference core performing the mathematical transformations to derive control actions from the set of rules.

The fuzzification interface provides a fuzzy partition of the input and output universes of discourse of the different variables, as well as a choice of the membership function which describes each fuzzy set. In general, the larger the number of fuzzy sets, the more complex the fuzzy controller, and the higher its potential capabilities [18-20].

Concerning the input variables, the fuzzification operation is performed by circuits known as Membership Function Circuits (MFCs). Several classes of parameterized functions are used

to define membership functions: triangles and trapezoids are popular in industrial control applications [10-11] due to their simplicity, while gaussian, sigmoidal, or bell-shaped functions are especially used in adaptive neuro-fuzzy controllers [21-22]. We have selected triangle and trapezoidal membership functions because they are very easily implemented by current-mode CMOS circuits (this will be discussed in Section V). Unlike to memory-based MFCs, e.g. those employed in digital realizations [5-7], an analog MFC implements a continuous membership function by its transfer characteristics. Hence, the input signals are directly processed and the input universes of discourse need not have a discrete representation. The parameters required to define a generic trapezoidal/triangle membership function are: (a) the situation in the universe of discourse, (b) the support or interval on the universe of discourse where the degrees of membership are not zero, and (c) the slopes of the trapezoid/triangle that take into account how the support is covered by the function (Figure 2). On the other hand, the output universe of discourse needs a discrete representation since it must be generated on-chip. How the output space is dealt with greatly determines the resulting hardware. To better understand this, let us briefly review some commonly used fuzzy inference mechanisms, focusing on hardware implementation.

Two operations have to be performed in the inference mechanism (Figure 3): the calculation of each rule's output or conclusion and the aggregation of all of them to obtain the overall output. To infer each rule's conclusion, the first value to calculate is the firing strength or activation degree of the rule, which measures the degree of matching between the controller input values and the antecedent part of the rule. This value results from the connection of antecedents via triangular norms (conjunctions "and") or conorms (disjunctions "or") [3]. Among the associated operations (minimum and maximum, algebraic product and sum, bounded product and sum, etc.), minimum (min) and maximum (max) are very easily implemented by integrated circuits (Section VI). They are also very popular in control applications. The operation which combines the antecedent part (represented by the activation degree) and the consequent part of a fuzzy rule is known as fuzzy implication [3].

Most of the first fuzzy controllers used fuzzy sets to define the consequent linguistic variables. The inference methods that they implemented were the "min-max" Mamdani's method, which uses a min operator to perform the fuzzy implication and a max operator to aggregate rules [23], and the "product-sum" method, which uses algebraic product and sum instead of min and max [10]. To defuzzify the global fuzzy output, the traditional strategies were the Middle Of Maxima (MOM) and the Center Of Gravity (COG) methods [3, 24, 25]. This is shown in Figure 4a.

From a microelectronics point of view, the definition of the consequent variable as a fuzzy set is very costly in terms of area or time. The first analog fuzzy chip, proposed by Yamakawa [14], represented the output space with 25 elements. The membership degrees of these elements to a particular fuzzy set were provided in parallel via a bus of 25 lines, resulting in high area occupation. To allow implementing more rules on a single chip, an alternative architecture proposed by the authors [26] employed serial circuits to generate these membership functions, at the cost of a slower inference speed.

The key point is that this hardware complexity is not justified by control efficiency. The studies in [24-25] reveal that the MOM method can make the controller input-output relation non-monotone, discontinuous, and even independent of the input membership functions. Regarding the COG method, some authors [25, 27] have detected the problem that the final output is more influenced by ambiguous rather than by precise consequents, because ambiguous consequents are represented by wider membership functions.

Hardware is significantly reduced if the consequent linguistic variable, B_i , is defined by several parameters instead of a membership function distributed along a discrete output space. These parameters will concentrate the information that a membership function provides about a fuzzy set. As cited for antecedents, this information concerns the situation of the membership function in the universe of discourse (given by the element with maximum degree of pertinence or by the center of area), its grade of fuzziness (represented by the support or width), and the way of covering that support (represented by the shape) (Figure 2). This can be achieved by using inference methods which first defuzzify each rule's output (generally given by a fuzzy set, B_i') to obtain a representative value $f(B_i')$, and then aggregate all of them via a weighted average [24, 25, 28, 29] (Figure 4b). The following general expression shows how these methods derive the controller output:

$$\text{output} = \frac{\sum_{i=1}^r w(h_i, B_i) f(B_i')}{\sum_{i=1}^r w(h_i, B_i)} \quad (1)$$

where h_i is the activation degree of the i -th rule.

In these methods, the information about the situation and the shape of the consequent membership function is included in the value $f(B_i')$, while the influence of consequent supports is included in the weight $w(h_i, B_i)$. In particular, the Weighted Fuzzy Mean (WFM) [24] and Quality Method (QM) [25] take $f(B_i')$ as constants, c_i , namely the center of area of the consequent fuzzy sets, B_i , while the weights are chosen as the activation degrees scaled by a constant parameter, $w(h_i, B_i) = h_i \cdot \gamma_i$. In the first method, the parameter γ_i is proportional to the support of the consequent fuzzy set, so that the global output is similar to that obtained by the product-sum-COG method, thus suffering from the same drawback of ambiguous consequent dominance previously commented. The Quality Method avoids this problem by employing a parameter γ_i inversely proportional to the fuzzy set's support. The QM and WFM method consider only the situation and support of the consequent fuzzy sets, not their shapes. The method proposed by Berenji and Khedkar (BKM) [28] takes them into account. If consequent sets are represented by piece-wise linear functions, the rule's conclusion derived by this method is a crisp value $f(B_i')$ that can be expressed as $(\alpha_i + \beta_i h_i)$, where h_i is the activation degree of the rule and α_i, β_i are constants related with the consequent piece-wise linear function. Unlike the former methods, the value $f(B_i')$ now includes information not only about the situation of the consequent fuzzy set, but also about its shape. This value $f(B_i')$ implicitly depends on the input signals, x_i , via the activation degree h_i . In Takagi and Sugeno's Method (TSM) [29], the value $f(B_i')$ is also influenced by the input signals. In this method, the influence is explicitly expressed by the following relation:

$$f(B_i') = \sum_{k=1}^u \alpha_{ik} x_k + \alpha_{i0}, \quad (2)$$

where α_{ik} are constants. (Table I summarizes all these simplified inference methods).

Finally, when $w(h_i, B_i)$ is equal to h_i and $f(B_i')$ is a constant, c_i , the resulting method is known as Fuzzy Mean, Height Method, Singleton, or zero-order Sugeno's Method [22, 24, 25, 30, 31], which can be considered as a simplification of all the above methods. In this case, Equation (1) is reduced to the following expression:

Table I: Inference methods that use parameterized information of the output space.

	WFM	QM	BKM	TSM
$w(h_i, B_i)$	$h_i \cdot \gamma_i$ ($\gamma_i \propto \text{support}$)	$h_i \cdot \gamma_i$ ($\gamma_i \propto 1/\text{support}$)	h_i	h_i
$f(B_i')$	c_i	c_i	$\alpha_i + \beta_i h_i$	$\sum_{k=1}^u \alpha_{ik} x_k + \alpha_{i0}$

$$\text{output} = \frac{\sum_{i=1}^r h_i \cdot c_i}{\sum_{i=1}^r h_i} \quad (3)$$

From a hardware point of view, this is the method with the least memory requirements (the least number of parameters to store and program). In addition, it does not need multiplier but rather scaler circuits, which are simpler to design and ease digital programmability. From a theoretical point of view, although the shapes and supports of consequent fuzzy sets can not be taken into account, there are enough parameters to choose from so as to ensure efficient control [24].

III. CONTROL SURFACES PROVIDED BY A SINGLETON MICROCONTROLLER

Generally, fuzzy controllers perform a nonlinear mapping between the input and output space. The design problem is to accurately approximate the desired control surface. The microcontroller selected here (with trapezoidal/triangular membership functions to represent antecedents, min as fuzzy conjunction, product as fuzzy implication, and singleton or crisp values to represent consequents) satisfies the conditions stated by Castro [31] for universal approximator. This means that it can uniformly approximate any real function to an arbitrary degree of accuracy, by using enough rules, antecedent fuzzy sets, and consequent singletons.

Although the control of any process is theoretically possible, it is helpful to know the kind of approximation this system performs. Let us consider a single-input single-output (SISO) fuzzy system, where the maximum number of overlapping antecedent membership functions is two. Under this assumption, given an input x , two rules are activated:

If x is A_1 then y is c_1

If x is A_2 then y is c_2

where the fuzzy sets A_1 and A_2 are assumed to be described by triangle membership functions, $\mu(x)$, with slopes m_1 and m_2 , respectively (Figure 5).

According to the expression (3), for an input value x the output y is given by:

$$y = \frac{c_1 h_1 + c_2 h_2}{h_1 + h_2} = \frac{c_1 [1 + m_1(a_1 - x)] + c_2 [1 + m_2(x - a_2)]}{2 + m_1(a_1 - x) + m_2(x - a_2)} \quad (4)$$

From the first and second derivatives of the above expression with respect to x , information can be obtained about the controller output for x belonging to the overlapping interval. This result, summarized in Table II and illustrated in Figure 5, shows that a SISO system may perform nonlinear approximation without resorting to the use of fuzzy consequents, by only changing the shapes and overlapping degree of the input fuzzy sets. Similar outputs (concave or convex nonlinearities) are also obtained when more than two antecedent membership functions overlap each other. This nonlinear behavior is illustrated in Figure 6a, where a cosine function is approximated with four rules, being the absolute error below 3.1%. Figure 6a also shows how smooth output can be achieved in spite of using sharp membership functions such as triangles (Figure 6b).

Table II: Features of the output signal of a SISO controller.

	$c_1 > c_2$	$c_1 < c_2$
$m_1 > m_2$	convex and monotonically decreasing	concave and monotonically increasing
$m_1 < m_2$	concave and monotonically decreasing	convex and monotonically increasing

For the particular case when two membership functions with the same slopes ($m_1=m_2$ for every overlapping interval) overlap, the output is a piece-wise linear function (Figures 6c and d). In this situation, the fuzzy system can be viewed as the piece-wise approximator (weighted sum of basis functions) usually employed in mathematical approximation theory [32]. Figure 6c illustrates this linear behavior. To approximate the same cosine function as in Figure 6a with the same accuracy, eight rules have been used (Figure 6d).

IV. ARCHITECTURE AND REQUIRED BUILDING BLOCKS.

Three functional stages are required to implement the selected singleton microcontroller:

- Antecedent fuzzifier, formed by the membership function circuits (MFCs).
- Connective, containing the circuits to implement min operator (MINs),
- Aggregation, performing the weighted average. It includes the circuits to implement the scaling with singleton consequents (CONSs).

Many analog implementations reported in the literature employ a data path for each rule [13, 14, 33, 34], which means that each rule has its own MFCs, MIN, and CONS blocks. If this architecture is chosen to implement a programmable fuzzy chip, the number of building blocks and parameters to program (associated with antecedents and consequents) are proportional to the number of rules, which is fixed in the chip. The main drawback is that the possible repeti-

tion of linguistic variables in different rules leads to redundant information and replicated blocks.

An alternative is to fix the number of input fuzzy sets and that all rules share the MFCs. Let us consider a controller chip with r rules, u input variables, and q_a fuzzy sets covering them (for simplicity, this number is assumed equal for all the input). In this approach, the number of MFCs is reduced from $u \cdot r$ to $u \cdot q_a$ (r is usually greater than q_a), so that the ratio of MFC saving is given by:

$$\frac{u \cdot r - u \cdot q_a}{u \cdot r} = 1 - \frac{q_a}{r} \quad (5)$$

The total number of bits to program the MFCs is also reduced from $u \cdot r \cdot p_a$ to $u \cdot q_a \cdot p_a$, where p_a is the number of bits required by each antecedent fuzzy set. However, information must be added, defining which MFCs are connected within each rule, implying an additional $u \cdot r \cdot q_a$ bits. The result is that the ratio of bits saved when the MFCs are shared by all the rules can be expressed as:

$$\frac{u \cdot r \cdot p_a - u \cdot q_a \cdot p_a - u \cdot r \cdot q_a}{u \cdot r \cdot p_a} = 1 - \left(\frac{q_a}{r} + \frac{q_a}{p_a} \right) \quad (6)$$

A second alternative is to also share the singleton values of the output variables, that is to share the CONS blocks by all the rules. If the number of possible values is q_c , this means a reduction of CONS blocks from r to q_c . If p_c is the number of bits required to program each singleton value, $(r - q_c) \cdot p_c$ bits are saved. However, $q_c \cdot r$ bits must be added to assign a CONS block to each rule. The expressions of the ratios of CONS blocks and bits saved by this approach together with equations (5) and (6) are summarized in Table III.

The data in Table III help the designer to choose the most suitable architecture for a set of given parameters $\{r, u, q_a, p_a, q_c, \text{ and } p_c\}$. In general, sharing MFCs is advantageous when the number of rules is large compared to the number of input fuzzy sets (controllers with input partitions of low resolution, $r > q_a$). Similarly, sharing consequents is preferable when the number of rules is large compared to the number of output singleton values (output partition of low resolution, $r > q_c$).

In particular, sharing both MFCs and CONSs is advantageous when the complete fuzzy rule set (q_a^u) is implemented on-chip, which is practical for many typical controllers with two or three input and input/output partitions of low resolution. Figure 7 illustrates the resulting matrix-like architecture for a two-input controller. It shows the connection of the three functional stages (antecedent fuzzifier, connective, and aggregation stages) and the digital buses to program the antecedent/consequent parameters and the rule set. In this case, the number of MIN circuits is the same as the number of rules, so that no bits are required to define which pairs of MFCs must be connected. The bits which program the rule set are then aimed to defining the CONS block associated to each rule.

Table III: Design alternatives for fuzzy controllers achitecture.

	Ratio of MFCs saved	Ratio of CONSs saved	Ratio of bits saved
MFCs shared by all the rules	$1 - \frac{q_a}{r}$	-	$1 - \left(\frac{q_a}{r} + \frac{q_a}{p_a} \right)$
CONSs shared by all the rules	-	$1 - \frac{q_c}{r}$	$1 - \left(\frac{q_c}{r} + \frac{q_c}{p_c} \right)$

V. FUZZIFIER STAGE

The fuzzifier stage, which performs the antecedents fuzzification, is composed of MFCs. The MFCs in our design implement trapezoidal/triangular functions that can be digitally programmed. To understand the mathematical operations involved in an MFC, let us discuss three strategies to generate symmetric trapezoidal functions illustrated in Figure 8.

(a) The fuzzifier operation shown in Figure 8a is carried out as follows:

$$\text{fuzzy}(x) = x_{\text{ref}} \ominus [m(x \ominus x_2) + m(x_1 \ominus x)] \quad (7)$$

where $\ominus$ is a rectification or bounded difference operator, defined as:

$$a \ominus b = \begin{cases} a - b & \text{if } a > b \\ 0 & \text{otherwise} \end{cases}$$

Current-mode realizations that implement this operation have been proposed in [12, 35].

(b) The technique illustrated in Figure 8b performs the following transformation:

$$\text{fuzzy}(x) = m(x \ominus x_1) - m(x \ominus x_2) - m(x \ominus x_3) \quad (8)$$

The current-mode circuit used in [36] follows this technique.

(c) The strategy proposed by the authors in [37] employs the parameters depicted in Figure 8c. It realizes the following transformation:

$$\text{fuzzy}(x) = x_{\text{ref}} \ominus m[(x \ominus x_{\text{aux}}) + (x_{\text{aux}} \ominus x) \ominus x_{\text{sat}}] = x_{\text{ref}} \ominus m(|x - x_{\text{aux}}| \ominus x_{\text{sat}}) \quad (9)$$

In any case, the basic mathematical operations required are addition (+), subtraction (-), rectification ($\ominus$), and scaling ($\bullet m$).

The current-mode circuit proposed by the authors following the last technique is shown in Figure 9. Working with currents, addition and subtraction are reduced to connecting wires (in this manner ' $I_{\text{in}} - I_{\text{aux}}$ ' is implemented at the input node). To realize the bounded difference, the

traditional solution is a diode-connected transistor, which can be the input transistor of a current mirror (the operation ' $\ominus I_{\text{sat}}$ ' is implemented by transistor T_m). A double bounded difference operation (' $I_{\text{in}} \ominus I_{\text{aux}}$ ' and ' $I_{\text{aux}} \ominus I_{\text{in}}$ ') is efficiently carried out by transistors T_1 and T_2 , which are never conducting simultaneously [38]. Unlike the current-mode circuits in [12, 35, 36] which must replicate I_{in} twice or three times, and deal with two or three possible large currents (I_1, I_2 and I_3), the circuit we use does not need to replicate I_{in} and I_{aux} , and the only additional current of possibly large value is I_{aux} . The result is better performance in terms of area occupation and power dissipation.

Precision of current-mode circuits is usually limited by mismatching in the current mirrors, due to systematic and random errors [39]. In this sense, an advantage of the proposed MFC regarding precision is the reduced number of replications. The factor which can provoke systematic errors is large voltage swing at the input node. This can be reduced by biasing the gates of transistors T_1 and T_2 with different voltage sources [26], or by including negative feedback via a simple inverter [40] as shown in Figure 9.

An interesting feature of this MFC is that digital programmability can be included in a straightforward manner. D/A converters have been employed to program I_{aux} and I_{sat} , and to perform the scaling by m . In this sense, the current-mode approach allows very small data converters capable of operating at low-power and high-speed, contrary to the high cost in area and/or time of traditional voltage-mode data converters. Another advantage is that these parameters can be changed independently and without affecting the value of the maximum-degree-of-membership parameter (I_{ref}).

We have designed a programmable MFC with 4 bits to fix I_{aux} and 2 bits to fix I_{sat} and m . A microphotograph of the circuit is shown in Figure 10a. The active area is 0.07 mm^2 in $1.5 \mu\text{m}$ CMOS technology. Figure 10b shows the experimental results obtained by programming the slope value m . Figure 10c illustrates a family of 8 triangular membership functions resulting from different digital codes for I_{aux} . Figure 10d shows the test obtained with the same codes in I_{aux} but with different codes for I_{sat} and m , resulting in a family of 8 trapezoidal functions.

To provide this MFC with voltage input, the differential version of the voltage-to-current converter proposed in [41] can be employed.

VI. CONNECTIVE STAGE

Min and max operators are typical connectives. In fact, only one of them is required since they are related via De Morgan's law as follows:

$$\min (x_i) = \overline{\max (\bar{x}_i)} = x_{\text{ref}} - \max (x_{\text{ref}} - x_i) \quad (10)$$

Connecting the MFC's output with a min operator results in:

$$\begin{aligned} \min [\text{fuzzy}_i(x_i)] &= x_{\text{ref}} - \max \{ x_{\text{ref}} - [x_{\text{ref}} \ominus m(|x_i - x_{\text{aux}}| \ominus x_{\text{sat}})] \} = \\ &= x_{\text{ref}} \ominus \max \{ m(|x_i - x_{\text{aux}}| \ominus x_{\text{sat}}) \} \end{aligned} \quad (11)$$

The last equation enables employing a max operator so that the bounded difference with I_{ref} is not performed within each MFC, but after antecedent connection. This manner was chosen since current-mode max operators are less costly to implement than min circuits, as shown in this section.

In general, fuzzy controllers have multiple antecedents, and thus require a multiple-input operator. The different algorithms proposed for multiple-input max operators are defined by the following expressions:

$$x_o = \sum_i (x_i \ominus (x_o - \xi_i)) \quad \text{with} \quad \xi_i = x_i \ominus \sum_{j \neq i} \xi_j \quad (12)$$

$$x_o = A \sum_i (x_i \ominus x_o) \quad (13)$$

where x_i are the input, x_o is the output, and A is a big number.

The first algorithm is analyzed in [42], and a current-mode realization is proposed. Each input current is replicated to inhibit all other input, resulting in an $O(n^2)$ structure with the sequel of large area occupation and slow operation due to many parasitic capacitors. In the other algorithm, proposed in [43], the inhibition signal for each input is represented by the mean value of those input which have not been zeroed out. Thus inhibition signals are simple to generate since they do not involve internal variables. Consequently, structures of $O(n)$ complexity are possible.

From a hardware point of view, the last algorithm is realized using feedback schemes with high-gain blocks to implement the scaling by A . The precision of a max circuit is evaluated by the difference required between input signals to obtain a single dominating one and by the error observed at the output when all the input signals have an equal value. A high gain value of A means that former errors are proportionally reduced. A typical voltage realization is the classical diode gate, used for OR operation in Boolean logic (see Figure 11a). This is the structure employed in the bipolar fuzzy rule chip of [14]. A typical solution to obtain high precision is to include a diode in the feedback loop of an operational amplifier [44, 45], as shown in Figure 11b. This solution leads to large area (because of operational amplifiers) and low operational speed since it requires a dominant pole to achieve stability (if several gain stages are cascaded).

These drawbacks are avoided by the current-mode circuit proposed by the authors in [46]. Its simple structure, shown in Figure 12, is the result of exploiting all the potentialities of MOS transistors. The voltage in the shared node is locally fed back and fixed to the value needed by an MOS transistor to be in saturation conducting the maximum input current, $I_{\max}$. This is the idea proposed by Lazzaro et al. [47] to implement a winner-take-all circuit, which is also used by Sasaki et al. [33] to realize a max circuit. The novelty of the circuit in Figure 12 is that transistors T_i are not only employed as competitive devices, but also as switches that convey the current from low-impedance to high-impedance node, thus avoiding additional cascode or regulated output stages as used in [33]. Transistors T_i act as diodes: if the current through M_x is lower than the current through M_y ($I(M_x) < I(M_y)$), M_x enters ohmic region, so that the voltage at the gate of T_x decreases and T_x goes off; contrarily if ($I(M_x) > I(M_y)$), M_x enters saturation and T_x , acting as a voltage follower, closes the feedback loop which makes $I(M_x) = I(T_x)$. The circuit can be viewed as the connection of improved Wilson current mirrors which share their output diode-connected transistor. This avoids DC matching errors typical of conventional Wilson mirrors [48].

A 3-input max circuit has been integrated in 2.4- μm CMOS technology, and is shown in the microphotograph of Figure 13a. Figure 13b illustrates the static behavior obtained when two of the input currents vary, while the third current is fixed to 0 μA . Precision of 0.23% of the total output current was measured. To illustrate the dynamic behavior, V-to-I and I-to-V converters were added at the input and output, respectively. Then the circuit performs as a volt-

age-mode min operator. Despite these converters, high speed is achieved (transient response of 160ns), as depicted in Figures 13c and 13d.

VII. AGGREGATION STAGE

This stage performs the weighed average of consequent singletons to obtain a crisp output. The mathematical operations involved are addition, scaling, and division (Equation (3)). Division has traditionally been the most difficult arithmetic function to implement with analog computing devices [49]. On one hand, this is due to nature of division which leads to very large output as the denominator approaches to zero (and the numerator is not zero). As a result, an ideal divider must have infinite gain and dynamic range. On the other hand, division derives from a linear relation, that is, to obtain $x_{\text{out}} = x_{\text{ref}} x_{\text{num}} / x_{\text{den}}$, the starting point is to attain $x_{\text{out}} x_{\text{den}} = x_{\text{num}} x_{\text{ref}}$. This is not easy to achieve if working with intrinsically nonlinear devices such as MOS transistors.

Sasaki and Ueno [30] have proposed avoiding division by selecting a particular antecedent connective and a particular input fuzzy partition, so that the denominator in Equation (3) is constant. However this approach can not be applied to multi-input fuzzy controllers. A way of avoiding division in multiple-input controllers is to use product as an antecedents' connective, input fuzzy sets such that for every input the sum of membership values is constant, and a complete rule set. However, the overall result can be an increased rather than reduced hardware cost (compare u -input r multipliers with u -input r max circuits and a divider).

Other solutions may basically follow three strategies:

(a) Normalization of weight parameters (activation degrees) followed by a sum of scalings, according to the structure illustrated in Figure 14a:

$$\text{output} = \sum_{i=1}^r \left(\frac{h_i}{\sum_{i=1}^r h_i} \right) c_i = \sum_{i=1}^r \bar{h}_i c_i \quad (14)$$

(b) Simultaneous weighting and averaging by means of an aggregation circuit as reported by C. Mead [50]. Since Mead's circuit employs OTAs, it implements another simplified method, namely one in which $w(h_i, C_i)$ (Equation (1)) is proportional to $(h_i)^{1/2}$. To realize the Singleton Method, true multipliers must be used in a feedback loop (Figure 14b) to achieve:

$$\sum_{i=1}^r (c_i - \text{output}) h_i = 0 \quad (15)$$

(c) Use of a two-level structure with a first stage of scaler operators, followed by a divider, as shown in Figure 14c.

Two CMOS current-mode schemes that approximate a true normalizer operator have been reported in literature [11, 33, 51]. These approaches do not verify one feature of simplified methods: the importance of each rule must depend only on its activation degree or its consequent parameters (Equation (1)), and not on the normalization operation. In particular, the Normalization Locked Loop proposed in [11, 33] changes the value of the maximum degree of

membership without proper scaling of the input domain. This causes some rules to be activated when they should not be and vice versa. The other reported structure [51] is the implementation of Gilbert's normalizer [52] with CMOS instead of bipolar transistors. The MOS version is nonlinear due to the quadratic behavior of MOS transistors in saturation.

Let us now consider the robustness of strategy b against mismatching and second-order effects. If an ideal multiplier should realize:

$$z_i = h_i(c_i - \text{output}) \quad (16)$$

a real one includes a slope deviation and an offset, which can be modelled as:

$$z_i = (h_i + \delta h_i)(c_i - \text{output}) + \delta z_i \quad (17)$$

All these errors are added when implementing the aggregation operation $\sum z_i = 0$. If the slope deviations (δh_i) are represented by their medium value ϵ^{sl} , and offset (δz_i) by ϵ^{of} , the approximated relative error at the output is:

$$\text{error}_r \approx \frac{\sum c_i \epsilon_i^{sl} + k \epsilon^{of}}{\sum h_i c_i} + \frac{k \epsilon^{sl}}{\sum h_i} \quad (18)$$

where k is the number of connected multipliers.

Variations in multipliers' gain can be counteracted to some extent by adjusting parameters c_i , as in programmable or self-tuning chips. However, offset errors are difficult to overcome. For digitally programmable fuzzy chips, in addition to these multipliers (possibly requiring compensation circuitry), D/A converters are required to program the parameters c_i .

Strategy c allows saving area by exploiting the capability of D/A converters to perform as scalars. Compared to analog multipliers, D/A scalars, in addition to their easier design, offer the advantages of digital programmability and storage [53]. In particular, current-mode D/A scalars do not suffer from offset errors, since they are based on current mirrors. Therefore, only one divider circuit must be compensated for offset. Regarding frequency behavior, the load capacitance affecting the divider is lower than the capacitance in the aggregation circuit, thus strategy c may offer higher operation speed. These advantages lead us to choose strategy c to implement programmable singleton fuzzy chips.

A. Scaling operation

The circuits that implement the fuzzy implication, CONSs, are then current-mode D/A converters, like the circuits which generate I_{aux} , I_{sat} , and m in the MFCs. Figures 15a and b show experimental results of 2- and 4-bit converters. Both display no offset and deviations from ideal values of less than 2% of maximum output current. Figure 15c illustrates the schematic of the circuit corresponding to the results in Figure 15b. It uses the common technique of binary-weighted current sources:

$$I_o = \left(\sum_{i=0}^3 b_i 2^i \right) h_i = c_i \cdot h_i \quad (19)$$

B. Addition operation

To date, we have exploited one of the advantages of current-mode techniques so that currents can be added by simply connecting wires. In particular this operation is now required to obtain the numerator ($\Sigma h_i c_i$) and denominator (Σh_i) of Equation (3), as shown in Figure 14c. It must be pointed out that the precision of this operation depends on the number of currents to sum, the output resistance of current sources (G_{ok}^{-1}), and the input resistance in the adding node (G_i^{-1}). Figure 16 shows a circuit to model the addition of k current sources. Applying the current Kirchhoff's law in the adding node, results that:

$$I_{\text{sum}} \left(1 + \frac{\Sigma_k G_{ok}}{G_i} \right) = \Sigma_k I_k \quad (20)$$

which can be approximated as:

$$I_{\text{sum}} \cong \Sigma_k I_k \left(1 - \frac{\Sigma_k G_{ok}}{G_i} \right) \quad (21)$$

To increase the output resistance of current sources, the current mirrors involved are preferably cascode mirrors.

C. Division operation

This section presents a unified framework to describe different solutions for divider circuits, which implement the equation $x_{\text{out}} = x_{\text{ref}} \cdot x_{\text{num}} / x_{\text{den}}$. As commented above, the starting point is to determine gain-variable linear circuits, $x_{\text{out}} = K(x_{\text{den}}) \cdot x_{\text{num}}$. Linearization techniques have been widely studied by analog designers since CMOS transistors are inherently nonlinear devices. For a survey on this topic, the authors refer to Chapter 5 in [17]. The most simple approach for a linearization technique is to combine two perfectly matched transistors operating in saturation or triode region [17, 41]. However, the resulting gain-variable elements are not easily controlled, which is a problem for our purpose. This was overcome by resorting to the use of four matched transistors, connected as shown in Figure 17. This cell provides a variety of solutions for divider circuits, depending on the region of operation and how input and output are selected. Three types are briefly analyzed in the following.

Type I:

When the cell in Figure 17 is biased in saturation, it can be viewed as a four-transistor translinear loop [54]. On this basis, the circuit depicted in Figure 18a can perform as a squarer or a square-rooter [54, 55] since its transfer characteristic is:

$$I_o = \sqrt{I_1 \cdot I_4} + \frac{I_1^2}{2(\sqrt{I_1} + \sqrt{I_4})^2} \quad (22)$$

where the simple square-law model of the MOS transistor has been assumed, $I = \beta(V_{GS} - V_T)^2/2$ with $\beta = \mu C_{ox} W/L$ (these parameters have their usual meaning).

A current-input current-output divider is then obtained by combining two squarers, as proposed in [54], or cascading a square-rooter with a squarer, as illustrated in Figure 18b. A divider prototype according to the latter has been integrated in a 2.4- μm CMOS process (its active area is 0.0513 mm², without considering biasing current mirrors). The transistor geometries were chosen so that the output of the square-rooter is $(I_X \cdot I_M)^{1/2}$ and the output of the squarer is $4I_1^2/I_C$, thus obtaining the following overall output:

$$I_o = I_X \frac{I_M}{I_C} \quad (23)$$

The main drawback of this divider is that it is very sensitive to deviations from the simple square-law model caused by length-channel modulation, mobility reduction, or mismatching. Experimental results of our prototype confirmed this. Figure 18c shows the output obtained for I_X fixed at 10 μA and denominator current I_C changing from 15 μA to 30 μA in steps of 2.5 μA , while x axis is the numerator current, I_M . Offset about 1.5 μA (10% of the maximum output current) and 3 μA (20%) were typically measured at the input and output, respectively.

Working with currents is more suitable since the numerator and denominator of equation (3) are represented by currents. However, it is preferable that the controller output be given in voltage-mode for interface capabilities. This means that the output of this type-I divider should be converted to voltage, which is performed by a transresistance element. The two following approaches configure the basic four-transistor element in Figure 17 to directly obtain a variable-transresistor with a transfer function $V_o = R \cdot I_{num}$. On one hand, since MOS transistors are basically transconductor devices ($I = V/R$), some kind of feedback must be included to reverse its behavior ($V = R \cdot I$). On the other, since MOS transistors are voltage-controlled devices, an I-to-V converter must be added so that the variable-resistance R is current, rather than voltage controlled. This results in:

$$V_o = R \cdot I_{num} = V_{ref} \cdot I_{num} / I_{den} \quad (24)$$

Type II:

A voltage-controlled transresistance element is obtained if the basic four-transistor cell is biased in saturation as shown in Figure 19a. Feedback is introduced by connecting transistors M_1 and M_3 in a diode configuration. Hence, their gates/drains are low-impedance nodes whose voltages can be taken as output variables. Applying the simple square-law model of MOS transistors, the resulting transfer characteristic is:

$$V_o = V_1 - V_2 = \frac{I_n}{\beta(V_a - V_b)} \quad (25)$$

Whenever feedback is included, care must be taken to ensure that the overall feedback is always negative, that is, the poles of the circuit are on the left-half s-plane; otherwise the circuit is unstable. In this case, the circuit stability is achieved if V_b is greater than V_a .

A symmetric CMOS structure (Figure 19b) has been chosen to provide high input impedance for both control nodes, V_a and V_b , thus avoiding voltage sources of low output impedance. This type of structures is widely used in the design of transconductors and multipliers [56-58]. Two circuits like that in Figure 19b have been combined to obtain an input-current output-voltage divider (see Figure 19c):

$$V_o = \frac{\beta_d}{\beta_n} (V_{ad} - V_{bd}) \frac{I_n}{I_d} \quad (26)$$

A prototype of this divider has been integrated in 2.4- μm CMOS technology, occupying an active area of 0.0648 mm^2 (without considering biasing current mirrors and analog output buffer), shown in the microphotograph of Figure 19d. The output voltage swing is 1.6v for 5v supply voltage and biasing current, I' , of 60 μA , while input ranges are 60 μA for denominator current. These results are shown in Figure 18e, where the different lines correspond to I_d changing from 0 to 60 μA in steps of 10 μA . Measured output offset voltages were typically less than 13mv (0.8% of the total output range), while input offset currents were less than 0.75 μA (1.5% of the input range).

Type III:

A voltage-controlled linear element is also obtained when the four-transistor cell operates in triode-region and the voltages at nodes a and b in Figure 20a have the same value. This circuit has been used in many applications, acting as a multiplier or a resistive network [17, 59-61]. Let us consider the circuit shown in Figure 20b, where the four-transistor cell is in the feedback path of a single-ended differential amplifier. This circuit may be seen as a transresistance amplifier which converts an input current into voltage through a voltage-controlled resistance. Its transfer characteristic is:

$$V_o = V_1 - V_2 = \frac{I_a - I_b}{\beta(V_a - V_b)} \quad (27)$$

where V_a must be greater than V_b to achieve stability.

Several advantages result from biasing the core cell in triode region. In particular, it allows a wide dynamic range, since there are no stacked transistors and voltage drops are usually low since the cell is operating in triode region. Additionally, body effect is inherently cancelled [17]. The most significant second-order effect (mobility reduction) can be reduced if fully-differential amplifiers are employed.

Some authors [60, 61] have used the circuit in Figure 20b with another four-transistor element at the input, so that $I_a - I_b$ is obtained from a voltage difference, thus resulting in a voltage-mode multiplier/divider. The objective of our application is just the opposite: to add an I-to-V converter so that $V_a - V_b$ is obtained from a current, I_d . In particular, a prototype has been also integrated in 2.4- μm CMOS technology, with the same I-to-V converter as used for the type-

II divider. A two-stage single-ended Miller opamp has been employed. The total active area (not including input current mirrors) is 0.0497mm^2 . It is shown in the microphotograph of Figure 20c. The output voltage swing is 1.8v for 5v supply voltage, while input current ranges are $60\mu\text{A}$ for denominator current, I_d . Output offset voltages of less than 10mv (0.5% of the total output range) were measured and input offset currents were less than $1\mu\text{A}$ (1% of the input range). These results are shown in Figure 20d, where the parametric lines correspond to I_d changing from $10\mu\text{A}$ to $60\mu\text{A}$ in steps of $10\mu\text{A}$, with I_b fixed at $10\mu\text{A}$.

Of all the divider circuits analyzed, the last two are good candidates to implement division in the average stage of fuzzy controllers. Design of the saturation-region divider requires a good selection of transistor geometries and careful layout to avoid mismatching. In the case of the triode-region divider, care must be taken in the design of the amplifier used.

VIII. DESIGN OF A FUZZY CONTROLLER CHIP

The circuits described to implement the fuzzy inference (MFCs, MIN/MAX, CONS, and divider circuits) are input-output compatible. They can be managed as library cells to place and route following an efficient architecture. As an example, we have designed a two-input one-output fuzzy chip which implements nine rules. Three fuzzy sets cover the universe of discourse for input and output variables. Digital words to program the antecedent parameters, I_{aux} , I_{sat} , and m , have 5, 3, and 3 bits respectively, while digital words to codify the consequent parameters, c_i , have 5 bits.

According to the discussion of Section IV, three kinds of architecture can be chosen: (a) a data-path for each rule, (b) sharing MFCs, and (c) sharing both MFCs and CONSs. Applying the figures given in Table III to this example ($u=2$, $r=9$, $q_a=q_c=3$, $p_a=11$, and $p_c=5$), shows that:

- Sharing MFCs allows a reduction of 66% in the analog part, $(1 - q_a / r) = 2/3$. Since the complete rule set is implemented, all the MFC combinations are required and no additional bits must be included to define the antecedent part of the rules. As a result, the number of bits is also reduced 66% $(1 - q_a / r)$ as compared with an architecture where the MFCs are associated to each rule.

- Sharing CONS circuits saves 66% of the analog part, $(1 - q_c / r) = 2/3$, and about 6.7% of the digital part, $(1 - q_c / r - q_c / p_c) = 1/15$, as compared with an architecture where CONS circuits are associated to each rule.

Consequently, the third architecture is selected. Figure 21 shows the analog core of the resulting layout corresponding to $2.4\text{-}\mu\text{m}$ CMOS technology. It follows the block diagram of Figure 7, where the average stage contains three 5-bit D/A converters (CONS blocks) and a type-III divider followed by an analog buffer. The area occupied is only $700\mu\text{m} \times 1400\mu\text{m}$.

The circuits described in this paper allow implementing the singleton fuzzy inference method with high approximation. In this sense, software fuzzy tools capable of performing the singleton method can be used to design the appropriate fuzzy controller for a given problem. The result can be down-loaded into the fuzzy chip through programming. For example, let us consider the design of a PI (proportional+integral) fuzzy controller which is usually referenced in the literature (Figure 22). The output of the system under control is compared with a reference signal. The error signal, $e(t)$, and its rate of change, $\Delta e(t)$, are the input of the fuzzy controller. The output of the controller is the change of the control signal that must be applied to the plant, $y(t)$.

The plant chosen for this example is a first-order system with a time constant of $0.5\mu\text{s}$. The rule base, the membership functions of antecedents, and the singleton values of consequents, shown in Figure 23, have been defined with the help of a fuzzy tool called Xfuzzy [62]. The former two-input one-output CMOS chip has been simulated with Hspice, and the resulting control surface (Δy versus e and Δe) (Figure 24a) shows an absolute error below 2.5% (Figure 24b). The transient behavior of the fuzzy chip has been also simulated for different pulse signals at the input (e , Δe) and a load of 5pF at the output (terminal Δy) (Figure 25). The maximum delay is less than 450ns. These static and dynamic features are good enough to ensure the control of the plant, as shown in Figure 26. This figure corresponds to the Hspice simulation of the fuzzy chip, working in the feedback-loop configuration of Figure 22.

IX. CONCLUSIONS

Efficient solutions at architectural and operator levels have been provided to implement programmable fuzzy controllers. In particular, the singleton method has been chosen as the inference mechanism. This method has proven very adequate for hardware realization while maintaining sufficient interpolation ability to approximate any control surface.

This paper has introduced circuits to implement the required functional blocks: (a) membership function circuits to define the input variables of the fuzzy controller, (b) min circuits to perform the antecedent connection, (c) scaler circuits to perform the fuzzy implication, and (d) divider circuits to obtain the control output.

The mathematical operations required in the different functional blocks (addition, subtraction, bounded difference, and scaling) are easily implemented with current-mode techniques, and thus result in very simple circuits. Current-mode D/A converters are used to program antecedent and consequent parameters. Their ability to perform as programmable scalers is exploited in the membership function circuits and the circuits that carry out the fuzzy implication. A design methodology for divider circuits is presented. Three types of realization are obtained from a four-transistor cell capable of implementing gain-variable circuits.

The resulting fuzzy controller is a mixed-signal IC where the computing power is provided by the analog part, while the digital part offers user-friendly programmability. The internal processing is performed in current-mode, but I/O interface is realized in voltage-mode to ease communication with conventional circuitry.

REFERENCES

- [1] E. A. Vittoz, "Analog VLSI signal processing: why, where, and how?", *Analog ICs and Signal Processing*, Kluwer Academic Pub., vol. 6, pp. 27-44, 1994.
- [2] L. A. Zadeh, "Making computers think like people", *IEEE Spectrum*, vol. 21, pp. 26-32, 1984.
- [3] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller", Parts I y II, *IEEE Trans. Syst., Man and Cybern.*, vol. 20, no. 2, pp. 404-432, 1990.
- [4] D. Driankov, H. Hellendoorn, and M. Reinfrank, "An introduction to fuzzy control", Heidelberg: Springer-Verlag, 1993.
- [5] M. Togai and H. Watanabe, "Expert system on a chip: An engine for real-time approximate reasoning" *IEEE Expert Syst. Mag.*, vol. 1, pp. 55-62, 1986.
- [6] H. Watanabe, W. D. Dettloff, and K. E. Yount, "A VLSI fuzzy logic controller with reconfigurable, cascable architecture", *IEEE Jour. of Solid-State Circuits*, vol. 25. no. 2, pp. 376-382, April 1990.
- [7] H. Eichfeld, M. Lohner, and M. Muller, "Architecture of a CMOS fuzzy logic controller with optimized memory organization and operator design", in *Proc. FUZZ-IEEE*, pp. 1317-1323, San Diego, 1992.
- [8] P. J. King and E. H. Mamdani, "The application of fuzzy control systems to industrial processes", *Automatica*, vol. 13, no. 3, pp. 235-242, 1977.
- [9] S. Boverie, B. Demaya, and A. Titli, "Fuzzy logic control compared with other automatic control approaches", in *Proc. of the 30th Conf. on Decision and Control*, pp. 1212-1216, Brighton, 1991.
- [10] M. Mizumoto, "Fuzzy controls by product-sum-gravity method", Advancement of fuzzy theory and systems in China and Japan, International Academic Publishers, 1990.
- [11] T. Yamakawa, "A Fuzzy inference engine in nonlinear analog mode and its application to a fuzzy logic control", *IEEE Trans. Neural Networks*, vol. 4, no. 3, pp. 496-522, May 1993.
- [12] L. Lemaitre, M. J. Patyra, and D. Mlynek, "Analysis and design of CMOS fuzzy logic controller in current mode", *IEEE Jour. of Solid-State Circuits*, vol. 29, no. 3, pp. 317-322, March 1994.
- [13] N. Manaresi, E. Franchi, R. Guerrieri, and G. Baccarani, "A modular analog architecture for fuzzy controllers", in *Proc. European Solid-State Circuits Conf.*, pp. 288-291, 1994.
- [14] T. Yamakawa, "High-Speed Fuzzy Controller Hardware System: The Mega FIPS Machine", Elsevier Science Pub. Comp. Inc., *Information Sciences* 45, pp. 113-128, 1988.
- [15] J. W. Fattaruso, S. S. Mahant-Shetti, and J. B. Barton, "A Fuzzy Logic Inference Processor", *IEEE Jour. of Solid-State Circuits*, vol. 29, no. 4, pp. 397-402, April 1994.
- [16] I. Baturone, S. Sánchez-Solano, A. Barriga and J.L. Huertas: "A switched capacitor based singleton fuzzy controller", 3rd *IEEE Int. Conf. on Fuzzy Systems*, pp. 1303-1307, Orlando, 1994.
- [17] C. Toumazou, F. J. Lidgley, and D. G. Haigh, "Analogue IC design: the current-mode approach", Peter Peregrinus Ltd., 1990.
- [18] T. J. Procyk and E. H. Mamdani, "A linguistic self-organizing process controller", *Automatica*, vol. 15, pp. 15-30, 1979.

- [19] B. Kosko, "Fuzzy systems as universal approximators", *IEEE Trans. Computers*, vol. 43, pp. 1329-1333, 1994.
- [20] X-J. Zeng and M. G. Singh, "Approximation theory of fuzzy systems-MIMO case", *IEEE Trans. Fuzzy Systems*, vol. 3, no. 2, pp. 219-235, May 1995.
- [21] L-X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning", *IEEE Trans. Neural Networks*, vol. 3, no. 5, pp. 807-814, Sept. 1992.
- [22] J-S. R. Jang and C-T. Sun, "Neuro-fuzzy modelling and control", *Proc. of the IEEE*, vol. 83, no. 3, pp. 378-406, March 1995.
- [23] E. H. Mandani and S. Assilan, "An experiment in linguistic synthesis with a fuzzy logic controller", *Int. J. Man-Mach. Studies*, vol. 7, no. 1, pp. 1-13, 1975.
- [24] R. Jager, H. B. Verbruggen, and P. M. Bruhx, "The role of defuzzification methods in the application of fuzzy logic", in *Proc. Symp. on Intelligent Components and Instruments for Control Applications*, pp. 111-116, Málaga, 1992.
- [25] H. Hellendoorn and C. Thomas, "Defuzzification in fuzzy controllers", *Jour. of Intelligent and Fuzzy Systems*, vol. 1, pp. 109-123, 1993.
- [26] J. L. Huertas, S. Sánchez-Solano, A. Barriga and I. Baturone, "A hardware implementation of fuzzy controllers using analog/digital techniques", *Int. Journal of Computers and Electrical Engineering*, vol. 20, no. 5, pp. 409-419, Sept. 1994.
- [27] W. Yu, T. Lim, I. Bae, and Z. Bien, "Fuzzy logic control of a roof crane with conflicting rules", in *Proc 5th IFSA World Congress*, pp. 1370-1373, Seul, 1993.
- [28] H. R. Berenji and P. Khedkar, "Learning and tuning fuzzy logic controllers through reinforcements", *IEEE Trans. Neural Network*, vol. 3, no. 5, 724-740 (1992).
- [29] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modelling and control", *IEEE Trans. Syst., Man, and Cybern.*, vol. 15, no. 1, pp. 116-132, 1985.
- [30] M. Sasaki and F. Ueno, "A fuzzy logic function generator (FLUG) implemented with current-mode CMOS circuits", in *Proc. Int. Symp. on Multiple Valued Logic*, pp. 356-362, 1991.
- [31] J. L. Castro, "Fuzzy logic controllers are universal approximators", *IEEE Trans. Syst., Man, and Cybern.*, vol. 25, no. 4, pp. 629-635, April 1995.
- [32] X-J. Zeng and M. G. Singh, "Approximation theory of fuzzy systems-SISO case", *IEEE Trans. Fuzzy Systems*, vol. 2, no. 2, pp. 162-176, May 1994.
- [33] M. Sasaki, N. Ishikawa, F. Ueno, and T. Inoue, "Current mode analog fuzzy hardware with voltage input interface and normalization locked loop", *IEICE Trans. Fundamentals*, vol. E75-A, no. 6, pp. 650-654, June 1992.
- [34] O. Landolt, "Efficient analog CMOS implementation of fuzzy rules by direct synthesis of multidimensional fuzzy subspaces", in *Proc. FUZZ-IEEE*, pp. 453-458, San Francisco 1993.
- [35] T. Yamakawa, T. Miki, and F. Ueno, "The design and fabrication of the current mode fuzzy logic semi-custom IC in the standard CMOS IC technology", in *Proc. 15th IEEE Int. Symp. Multiple-Valued Logic*, pp. 76-82, May 1985.
- [36] T. Kettner, C. Heite and K. Schumacher, "Analog CMOS realization of fuzzy logic membership functions", *IEEE Jour. of Solid-State Circuits*, vol. 28, no. 7, pp. 857-861, July 1993.

- [37] J. L. Huertas, S. Sánchez-Solano, A. Barriga and I. Baturone, "Serial architecture for fuzzy controllers: hardware implementation using analog/ digital VLSI techniques", in *Proc. 2nd. Int. Conf. on Fuzzy Logic and Neural Networks*, pp. 535-538, Iizuka, 1992.
- [38] Z. Wang, "Novel pseudo RMS current converter for sinusoidal signals using a CMOS precision current rectifier", *IEEE Trans. Instrumentation and Measurement*, vol. 39, no. 4, pp. 670-671, 1990.
- [39] D. G. Nairn and C. A. T. Salama, "Current-mode algorithmic analog-to-digital converters", *IEEE Jour. of Solid-State Circuits*, vol. 25, no. 4, pp. 997-1004, Aug. 1990.
- [40] A. Rodríguez-Vázquez, R. Domínguez-Castro, F. Medeiro, and M. Delgado-Restituto, "High-Resolution CMOS Current Comparators: Design and Application to Current-Mode Function Generation", *Analog ICs and Signal Proc.*, vol. 7, pp. 149-165, 1995.
- [41] K. Bult and H. Wallinga, "A class of analog CMOS circuits based on the square-law characteristic of an MOS transistor in saturation", *IEEE Jour. of Solid-State Circuits*, vol. 22, no. 3, pp. 357-365, June 1987.
- [42] M. Sasaki, T. Inoue, Y. Shirai, and F. Ueno, "Fuzzy multiple-input maximum and minimum circuits in current mode and their analysis using bounded difference equations", *IEEE Trans. on Computer*, vol. 39, no. 6, pp. 768-774, June 1990.
- [43] T. Inoue, F. Ueno, and T. Motomura, "Analysis and design of analog CMOS building blocks for integrated fuzzy inference circuits", in *Proc. Int. Symp. on Circuits and Systems*, pp. 2024-2027, Singapore 1991.
- [44] J. G. Graeme, "*Designing with operational amplifiers*", Mc Graw Hill, 1977.
- [45] M. Hassoun and A. Nabha, "Implementation of $O(n)$ complexity max/min circuits for fuzzy and connectionist computing", in *Proc. IEEE Int. Conf. on Neural Networks*, pp. 998-1003, March 1993.
- [46] I. Baturone, J. L. Huertas, A. Barriga, and S. Sánchez Solano, "Current-mode multiple-input max circuit", *Electronics Letters*, vol. 30, no. 9, pp. 678-680, April 1994.
- [47] J. Lazzaro, S. Ryckebusch, M. A. Mahowald, and C. A. Mead, "Winner-take-all network of $O(n)$ complexity" in D. Touretzky (Ed.): "*Advances in neural information processing systems I*" (Morgan Kaufmann, San Mateo, CA, 1989), pp. 703-711.
- [48] Z. Wang, "Analytical determination of output resistance and DC matching errors in MOS current mirrors", *IEE Proc., Pt. G*, vol. 137, no. 5, pp. 397-404, Oct. 1990.
- [49] Analog Devices, Inc., "Nonlinear circuit handbook", edited by D. H. Sheingold, 1976.
- [50] C. Mead, "*Analog VLSI and neural systems*", Addison-Wesley, 1989.
- [51] F. V. Verdú, A. Rodríguez-Vázquez, and B. Linares-Barranco, "Open loop normalization circuit in current mode", in *Proc. 3rd. Int. Conf. on Fuzzy Logic, Neural Nets and Soft Computing*, pp. 645-646, Iizuka, Aug. 1994.
- [52] B. Gilbert, "A monolithic 16-channel analog array normalizer", *IEEE Jour. of Solid-State Circuits*, vol. 19. no. 6, pp. 956-963, Dec. 1984.
- [53] S. Sánchez-Solano, I. Baturone, A. Barriga, and J. L. Huertas, "A current-mode memory and computing block for advanced fuzzy chips", accepted for publication at *VI Int. Fuzzy Systems Association World Congress*, July 1995.
- [54] R. J. Wiegerink, "Analysis and Synthesis of MOS translinear circuits", Kluwer Academic Pub., Boston, 1993.

- [55] I. Baturone, J. L. Huertas, A. Barriga, and S. Sánchez Solano, "Extending the functionality of a flexible current-mode CMOS circuit", *Electronics Letters*, vol. 31, no. 15, pp. 1231-1232, July 1995.
- [56] E. Seevinck and R. Wassenaar, "A versatile CMOS linear transconductor/square-law function", *IEEE Jour. of Solid-State Circuits*, vol. 22, no. 3, pp. 366-377, June 1987.
- [57] Z. Czarnul y S. Takagi, "Design of linear tunable CMOS differential transconductor cells", *Electronics Letters*, vol. 26, no. 21, pp. 1809-1811, Oct. 1990.
- [58] S. Sakurai and M. Ismail, "High frequency wide range CMOS analogue multiplier", *Electronics Letters*, vol. 28, no. 24, pp. 2228-2229, Nov. 1992.
- [59] B.-S. Song, "CMOS RF circuits for data communication applications", *IEEE Jour. of Solid-State Circuits*, vol. 21, no. 2, pp. 310-317, April 1986.
- [60] Z. Czarnul, "Novel MOS resistive circuit for synthesis of fully integrated continuous-time filters", *IEEE Trans. Circuits and Systems*, vol. 33, no. 7, pp. 718-721, July 1986.
- [61] N. Kachab and M. Ismail, "A nonlinear CMOS analog cell for VLSI signal and information processing", *IEEE Jour. of Solid-State Circuits*, vol. 26, no. 11, pp. 1689-1699, Nov. 1991.
- [62] J. Masa, "Xfuzzy 1.0: A simulation environment for control systems based on fuzzy logic", Technical report, University of Seville, Sept. 1992. (In Spanish).

Figure 1: A conceptual view of a fuzzy controller.

Figure 2: Parameters of a trapezoidal membership function.

Figure 3: Basic operations in a fuzzy controller.

Figure 4: Two ways of dealing with information about the output universe of discourse: (a) consequents defined by fuzzy sets, and (b) consequents represented by several parameters.

Figure 5: Types of output signal that a singleton SISO controller may provide.

Figure 6: (a) Nonlinear output of a singleton controller approximating a cosine function. (b) Antecedents' membership functions of the rules employed. (c) Piece-wise linear output of a singleton controller approximating the same function as in (a). (d) Antecedents' membership functions of the rules employed in (c).

(a)

(b)

(c)

(d)

Figure 7: Matrix-like architecture sharing MFCs and CONS circuits for a two-input programmable fuzzy chip.

Figure 8: Strategies to generate membership functions.

Figure 9: Current-mode MFC for the third strategy.

Figure 10: (a) Microphotograph of a programmable MFC. (b) Programming the slope. (c) Family of 8 triangular membership functions. (d) Family of 8 trapezoidal membership functions.

(a)

(b)

(c)

(d)

Figure 11: Realizations of multi-input max circuits: (a) Classical diode gate circuit. (b) Improved diode circuit.

Figure 12: Proposed current-mode max circuit.

Figure 13: (a) Microphotograph of a 3-input current-mode max circuit. (b) Measured DC behavior. (c) and (d) Measured transient response of a voltage-input voltage-output min circuit based on a circuit like that in (a)

(a)

(b)

(c)

(d)

Figure 14: Block diagrams of architectures for weighted average operators: (a) Strategy a. (b) Strategy b. (c) Strategy c. (d) Symbol of a multiplier. (e) Symbol of a scaler.

Figure 15: (a) Experimental results of a 2-bit current-mode D/A converter. (b) Experimental results of a 4-bit current-mode D/A converter. (c) Consequent block with binary-weighted sources.

(a)

(b)

(c)

Figure 16: Model to illustrate addition operation.

Figure 17: Core cell to implement gain-variable linear circuits.

Figure 18: (a) Circuit able to perform as a square-rooter (SQRT) or as a squarer (SQR). (b) Combination SQRT/SQR to obtain a current-input current-output divider. (c) Experimental results of a prototype implementing the block diagram in (b).

(a)

(b)

(c)

Figure 19: (a) Saturation-region basic cell to implement a voltage-controlled transresistance circuit. (b) CMOS realization of the cell in (a). (c) Block diagram of a current-input voltage-output divider combining two cells like that in (b). (d) Microphotograph of a prototype that implements the scheme in (c). (e) Experimental results of the previous circuit.

(a)

(d)

(b)

(e)

(c)

Figure 20: (a) Triode-region basic cell to implement a voltage-controlled transresistance circuit. (b) Realization of the cell in (a) via a differential amplifier. (c) Microphotograph of a current-input voltage-output divider based on the structure in (b). (d) Experimental results of the previous circuit.

(a)

(c)

(b)

(d)

Figure 21: Layout of the analog core of a programmable chip implementing nine rules.

Figure 22: Block diagram of a PI fuzzy controller.

Figure 23: (a) Employed rule-base. (b) Membership functions for antecedents and singleton values for consequents.

Figure 24: (a) Control surface provided by the CMOS controller (results from Hspice simulations). (b) Error surface (from a comparison with the theoretical control surface).

Figure 25: Different transient responses of the CMOS fuzzy controller (results from Hspice simulations).

Figure 26: Hspice simulation of the fuzzy chip working in the feedback-loop configuration of Figure 22.