Neuro-Fuzzy Chip to Handle Complex Tasks with Analog Performance

Fernando Vidal-Verdú

Dto.Electrónica-Universidad de Málaga
Complejo Tecnológico, Campus Teatinos 29071 Málaga, SPAIN
email: vidal@ctima.uma.es

Rafael Navas-González

Dto.Electrónica-Universidad de Málaga
Complejo Tecnológico, Campus Teatinos
29071 Málaga, SPAIN
email: rnavas@ctima.uma.es

Angel Rodríguez-Vázquez

Dept.of Analog and Mixed-Signal
Circuit Design
C N M - Universidad de Sevilla
Edificio CICA, C/Tarfia s/n,
41012-Sevilla, SPAIN
angel@cnm.us.es

Keywords: Fuzzy-Control, Fuzzy-Hardware, Mixed-Signal.

Paper No. SP018

Corresponding Author:

Dr. Fernando Vidal-Verdú
Dto.Electrónica-Universidad de Málaga
Complejo Tecnológico, Campus Teatinos
29071 Málaga, SPAIN
phone: +34 952 13 33 25
fax: +34 952 13 3324
email:vidal@ctima.uma.es

Neuro-Fuzzy Chip to Handle Complex Tasks with Analog Performance

Abstract

This Paper presents a mixed-signal neuro-fuzzy controller chip which, in terms of power consumption, input-output delay and precision performs as a fully analog implementation. However, it has much larger complexity than its purely analog counterparts. This combination of performance and complexity is achieved through the use of a mixed-signal architecture consisting of a programmable analog core of reduced complexity, and a strategy, and the associated mixed-signal circuitry, to cover the whole input space through the dynamic programming of this core [1]. Since errors and delays are proportional to the reduced number of fuzzy rules included in the analog core, they are much smaller than in the case where the whole rule set is implemented by analog circuitry. Also, the area and the power consumption of the new architecture are smaller than those of its purely analog counterparts simply because most rules are implemented through programming. The Paper presents a set of building blocks associated to this architecture, and gives results for an exemplary prototype. This prototype, called MFCON, has been realized in a CMOS 0.7μm standard technology. It has two inputs, implements 64 rules and features 500ns of input to output delay with 16mW of power consumption. Results from the chip in a control application with a DC motor are also provided.

I. INTRODUCTION

Associative Memory Networks, as the CMAC network, the B-spline Network, Radial Basis Function Network or Fuzzy Systems [2] perform a partition of the input space and generate the output from data related to small areas around the input vector. This fact provides network transparency and allows the introduction of structured knowledge, as in the Fuzzy Systems, which has become a major advantage to design control systems quickly. On the other hand, the number of basis functions (rules in a fuzzy system) grows exponentially with the input space dimension in Associative Memory Networks, which is their main disadvantage.

Implementations of large control algorithms with many variables are usually carried out by software in powerful computers [3][4]. This works for systems where just mechanical or thermal processes are involved, with time constants above one second, thus the input-output delay of the control action is not very demanding. However, if faster processes are going to be faced, as those in motion control and power systems, special purpose hardware could be necessary or perform better than the previous approach. To cope with complex control tasks in the range of milliseconds, general purpose microprocessors or DSPs, or those with a special set of instructions are the best choice [5][6]. However, to manage delays of few milliseconds and down to the microsecond and even nanosecond range, special purpose ASICs are required [7][8][9].

Digital ASICS [10][11] are robust because they work with digital signals, thus they can handle more complex tasks. On the other hand, they need the outer shell of analog circuitry to build the interface with sensors and actuators. Analog circuits [12][13][14][15][16] are considered good candidates to implement neural networks, despite their sensitivity to errors and noise, because the precision requirements are supposed low. However, the latter is not as true in ANN as in other networks with a high redundancy as the multilayer perceptrons, due to the fact that just a few nodes and parameters determine the output, thus errors in these nodes modify the output and are not compensated or corrected by other nodes. Thus, if the input dimension grows and hence the system complexity, errors are difficult to keep bounded. The most complex pure monolithic analog fuzzy controllers implement around 15 rules (basis functions) [13][14]. Nevertheless, since analog circuits provide the best efficiency in terms of area and power/speed ratio, it would be desirable to be able to increase their complexity to manage problems as motor control, where complexities above 25 rules are common [5]. Their straight interface to the plant and faster operation allow them to get better results in terms of less overshoot, smaller settling time, oscillations or ripple voltages or currents [7][9].

In order to build larger analog circuits with bounded errors, we could exploit the inherent tuning capabilities learning procedures have. However, this is only useful if the learning algorithm is implemented on-chip or with the chip-in-a-loop [17]. The first approach increases the requirements on hardware, because we need precise circuits for the learning part as well as fine tunable nodes in the remaining architecture [18]. The second approach still needs tunable nodes but takes advantage of an external computer to perform precise computations. The main drawback here is that the resulting controller is more expensive because of the need of tuning. It is also less flexible, the process to get the controller becomes longer and begins to loss its main appeal. In addition, the resulting controller is less autonomous to be used in embedded applications. Thus, learning algorithms are many times used to get the programming parameters in a conventional computer, then the result is put on silicon without further tuning [9].

Under such conditions, how could we increase the complexity of the networks while processing in analog mode to keep a small delay, power consumption and area?. This paper shows and implementation of the strategy in Fig.1 that exploits the local feature of ANNs to preserve the advantages of analog implementations [1]. Since ANNs provide the output from just a few set of nodes, we implement these nodes in an analog core and make it dynamically programmable to compute the output for any input vector. The basis function identifier performs the input space partition with a set of A/D converters. Note that these converters do not convert the input for further computing, but just perform a coarse clustering to get the set of basis functions that determine the output. This means we usually need only simple, as low as 3bits A/D converters for every input dimension. The output of the converters is used to address a data base which stores the programming data for the analog core. Once the programming data are in the programming bus, the controller is able to provide the output because the core processes the input with analog circuits. Since the input to output signal path is entirely analog, the analog performance is preserved as long as the programming time is just a fraction of the analog core delay. In addition, small analog cores

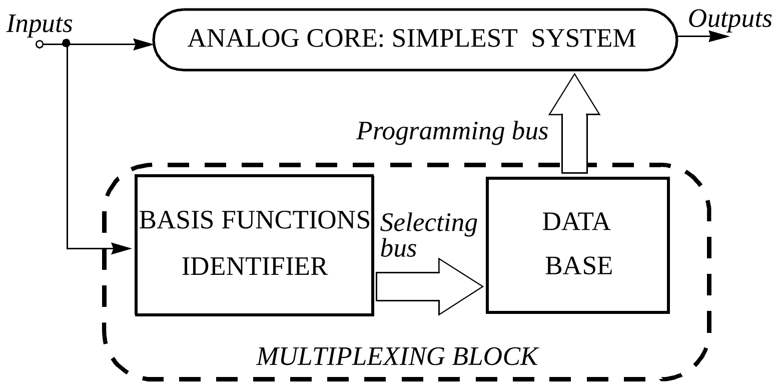


Fig. 1. Strategy to increase the complexity

can be carefully designed to bound the error at output and be multiplexed dynamically to implement a large number of rules. This solves the problem of facing complex tasks while preserves a small input-output delay time, and even good performance in terms of area and power consumption. Section II briefly describes this strategy and the resulting mixed-signal high-complexity fuzzy architecture; Sections III and IV describe the implementation of the high level building blocks in this architecture; and Section V provides experimental results of the MFCON prototype based on the previous approach that has been implemented in standard technology as well as results from a control application example. Finally, conclusions are collected in Section VI.

II. MIXED-SIGNAL HIGH-COMPLEXITY FUZZY CONTROLLER ARCHITECTURE

The proposed controller is based on a zero-order Takagi-Sugeno fuzzy system whose rules,

$$\text{IF } (x_1 \text{ is } \mathrm{A}_{1k}) \text{ AND } (x_2 \text{ is } \mathrm{A}_{2k}) \text{ AND} \ldots (x_M \text{ is } \mathrm{A}_{Mk}) \quad \text{THEN } y = y_k^* \quad (1 \le k \le N)$$

have *singletons* in the consequents. The *surface response* is interpolated from the singletons as

$$y = f(\mathbf{x}) = \sum_{k=1,N} y_k^* \frac{w_k(\mathbf{x})}{\sum_{k=1,N} w_k(\mathbf{x})} = \sum_{k=1,N} y_k^* w_k^*(\mathbf{x}) \tag{1}$$

where the multi-dimensional *basis functions* $w_k(\mathbf{x})$ are evaluated by extracting the *minimum* from the values of the one-dimensional *membership functions* $s_{ik}(x_i)$ associated to the *k*-th rule,

$$w_k(\mathbf{x}) = min\{s_{1k}(x_1), s_{2k}(x_2), ..., s_{Mk}(x_M)\}$$

and $s_{ik}(x_i)$ are chosen to generate a *lattice* partition of the input space [19] [20] – see Fig.2(a) for illustration of lattice partitions.

This type of inference with lattice partitions has been employed in different implementations. Its advantages are simplicity, generality and ease of programmability [21] [22]. As a counterpart, the number of rules needed to perform a good approximation becomes prohibitively large as the number of inputs increases [20]. Since in fully-analog implementations the errors and parasitic capacitances at the global computation nodes grow with the number of rules, these errors become very large, thus degrading the global accuracy and input-output delay.

The architecture used herein overcomes this problem by using the *decomposition property* presented in [23] [24]. The input space is split into subspaces defined by the lattice partition; see Fig.2(a) for illustration. Within each subspace the corresponding piece of the surface response – see drawing at the right in Fig.2(a) – is captured by the *simplest fuzzy system* within this subspace, a system of just two rules per input [23] [24]. Interestingly, the structure of this simplest fuzzy system remains the same for all subspaces; only some parameters must be tuned in order to fit the

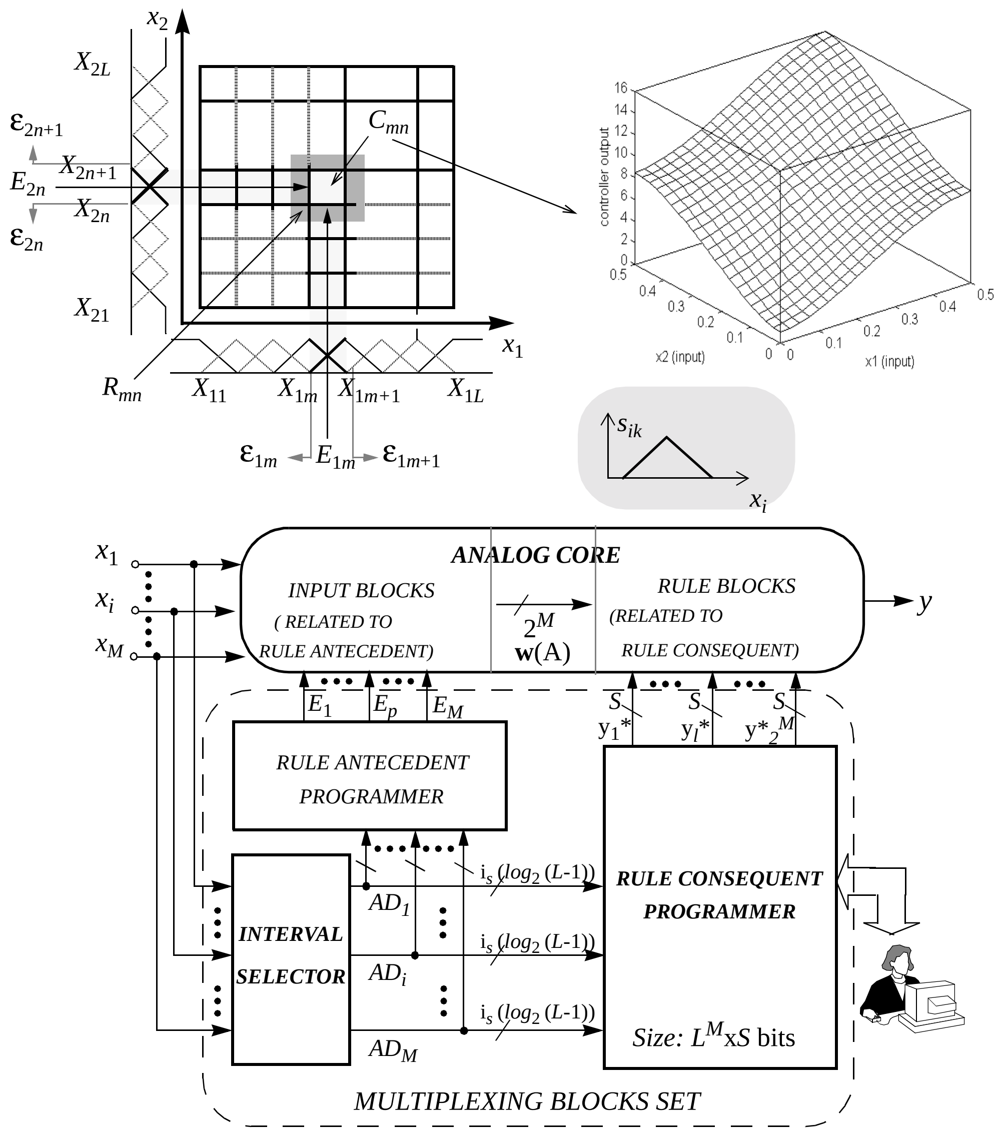


Fig. 2. General Multiplexed Architecture Fuzzy Controller with Analog Core

surface response within each subspace. Thus, the strategy adopted here consists of: a) implementing a *programmable* analog fuzzy *core* for the simplest fuzzy system; b) locating the subspace corresponding to the applied inputs; c) mapping the actual subspace location onto a set of corresponding programming signals for the analog fuzzy core. Outputs are then computed by the analog core driven by the inputs and the subspace programming signals. Since errors and input-output delay are basically determined by the simple analog core, they can be kept bounded even in very complex controllers.

From now on, the generic subspace, labelled C_{mn} in Fig.2(a), will be called *interpolation interval*. Fig.2(b) shows the proposed fuzzy controller architecture for a case with M inputs, L labels per input – thus L^M rules –, and where digital words of S-bits per singleton are used to represent the singleton values. Two parts are clearly identified. The **Analog Core** at the top implements the simplest fuzzy system. The **Multiplexing Blocks Set** delivers the programming signals corresponding to the interpolation interval to which the actual inputs belong.

In the Multiplexing Blocks Set, a battery of simple and fast AD converters – just three bits if seven labels per input are considered – is used to codify the interpolation intervals. The digital word $AD_1 \ldots AD_M$ of $M \times \mathrm{i}_S[\log_2(L-1)]$ [1] bits provided by these converters drives the **Rule Antecedent Programmer**, which provides a set of analog programming values,

$$PA_{C_{k_1 \ldots k_M}} = \{E_1, \ldots, E_p, \ldots, E_M\}, \tag{2}$$

and the **Rule Consequent Programmer**, which provides a set of digital programming values,

$$PC_{C_{k_1 \ldots k_M}} = \left\{ \mathrm{y}_1^*, \ldots, \mathrm{y}_l^*, \ldots, \mathrm{y}_{2^M}^* \right\}. \tag{3}$$

These are used to program the antecedent and the consequent blocks of the Analog Core, respectively.

1. $\mathrm{i}_S(x)$ means the next superior integer of x.

This system operates in asynchronous and continuous-time mode, so that its input/output delay is bounded only by the intrinsic circuit response time. In order to preserve the analog performance, the multiplexing blocks must be designed to minimize their effects on global parameters, such as the input/output delay, errors, etc. In the following, we will describe the implementation of the blocks in Fig.2(b). This description includes considerations pertaining to a general case and details pertaining to the bidimensional case implemented at the MFCON controller prototype.

III. ANALOG CORE

The analog core of Fig.2(b) implements the Takagi-Sugeno algorithm in (1) in a controller with two inputs, therefore four rules, which has a generic interpolation interval as the input space – see Fig.3(a). The circuitry is based on that previously reported by the authors in [14], although some important modifications have been made to incorporate programmability, as well as to save area and power.

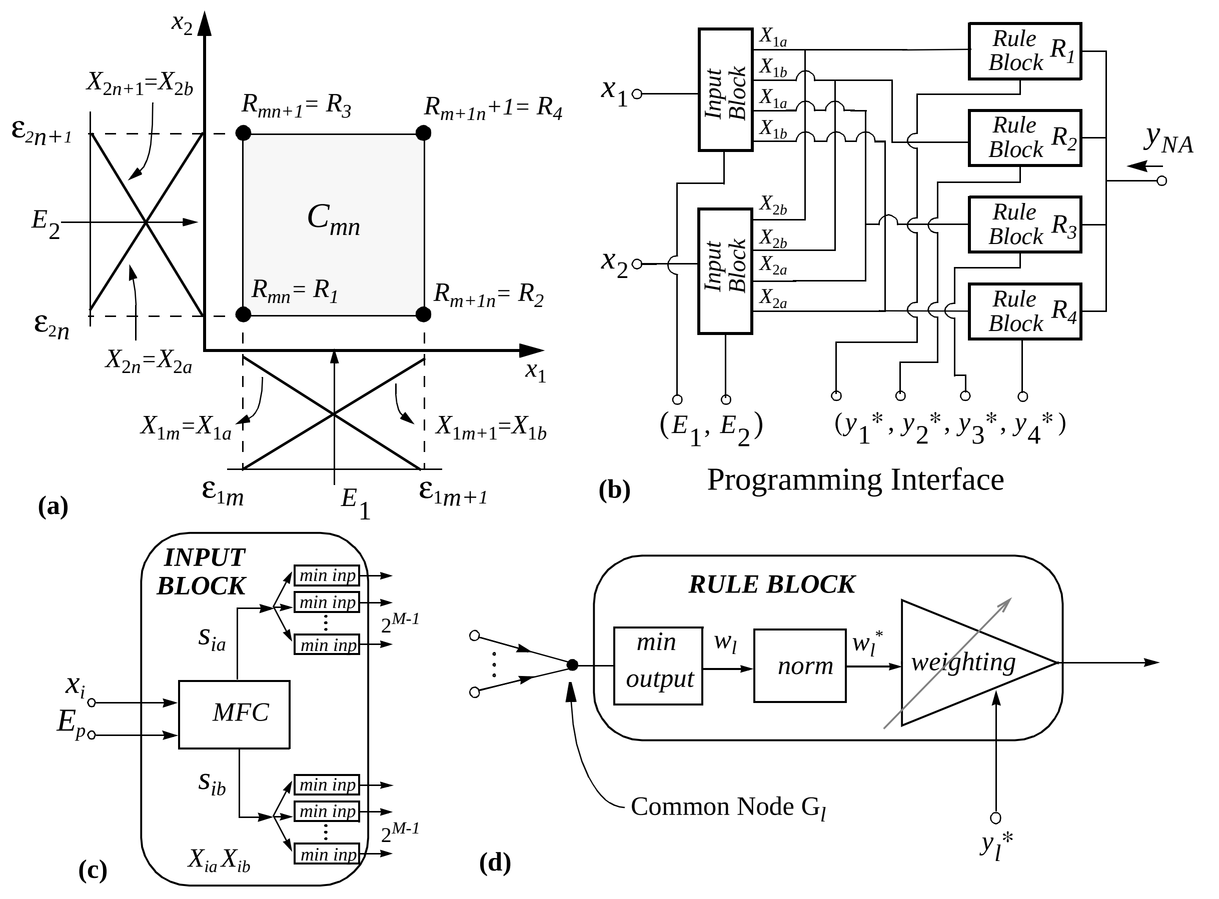


Fig. 3. (a) Interpolation interval and related parameters, (b) analog core architecture, (c) input block, (d) rule block

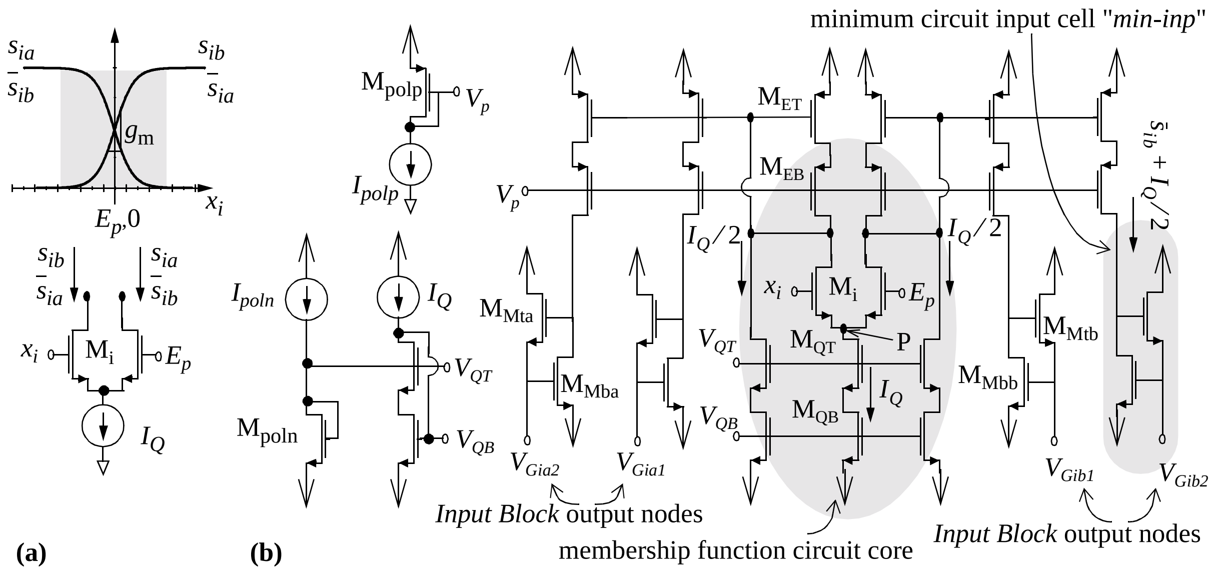
The core is composed of instances of two main building blocks, namely the *Input Block* – see Fig.3(c) – and the *Rule Block* – see Fig.3(d). There is one *Input Block* per input, hence two in total; and one *Rule Block* per rule of the simplest fuzzy system, hence four in total. These six blocks are wired as depicted in Fig.3(b) to form the core. Programming of the input blocks is made through voltages that locate the center of the interpolation interval – see Fig.3(a). On the other hand, programming of the rule blocks is realized through digital signals which codify the singleton values.

Note that rules R_1 and R_3 in Fig.3(a) are rules R_2 and R_4, respectively, in the interval located at left of that depicted in the figure. This means that a rearrangement of the rules is required when there is a change of the interpolation interval and the core is programmed. This rearrangement is realized by analog multiplexors in other programmable architectures [13][25]; here, however, the core architecture is fixed, and the rearrangement is realized by digital multiplexors in the programming interface – details are found in Section IV.E. On the one hand, this strategy is much more robust; on the other, it yields a significant reduction of errors, delays and interferences.

A. Input Block Circuitry

As Fig.3(c) shows, the input block accepts two types of inputs, the set of controller inputs x_i and the set of programming signals E_p; and delivers two sets of 2^{M-1} outputs. Fig.4(b) shows the schematics of the input block. The front-end differential amplifier is employed to obtain the membership functions associated to labels X_{ia} and X_{ib} (see Fig.3(a)). Hence, programmability of the central point location is readily implemented by driving the differential pair with the voltage E_p. Also, since these membership functions are complementary, a simple differential pair suffices to provide both, as Fig.4(a) illustrates. This is exploited to simplify the implementation of the minimum operator by using De Morgan's Law, i.e. with complement plus maximum circuits.

Note in Fig.4(b) that the differential pair outputs are replicated by current mirrors and then used



M_i *(W/L)*	M_{ET} *(W/L)*	M_{EB} *(W/L)*	M_{QT} *(W/L)*	M_{QB} *(W/L)*	$M_{Mta,b}$ *(W/L)*
21.6µm/12µm	24µm/4µm	24µm/0.8µm	12µm/0.8µm	12µm/6µm	9µm/4.2µm
$M_{Mba,b}$ *(W/L)*	M_{poln} *(W/L)*	M_{polp} *(W/L)*	I_{polp}	I_{poln}	I_Q
9µm/4.2µm	5µm/20µm	5µm/18µm	5µA	5µA	20µA

Fig. 4. (a) Differential pair, (b) *Input Block* schematic

to drive the so-called input cells of the minimum circuit [26]. The output cell of this circuit is implemented in the *Rule Block* – see Fig.3(d). For a given set of currents associated to corresponding minimum circuit input cells, the maximum among them is selected by wiring the output nodes of these cells, see Fig.4(b), to the node G_l of Fig.3(d).

Table 1 gives some expressions related to circuit design and performance. $\beta_i = \beta_0 \cdot (W_i/L_i)$

Table 1: Main design equations for the *Input Block*

Input range	$V_{QT} + \sqrt{I_Q/(2\beta_i)} \le x_i \le V_p + V_{T0n} + V_{T0p}$
Membership function shape	$s_{ib}(x_i) = I_Q/2 + \sqrt{I_Q\beta_i/2}(x_i - E_p)\sqrt{1-(x_i-E_p)^2\beta_i/2I_Q}$, $\lvert x_i - E_p\rvert \le \hat{x}$
Power consumption	$P_{input\_block} = 2I_Q(2^{2M-1}+1)V_{DD}$, $M(\text{number of inputs}) = 2$ in Fig.4

in Table 1 is the large signal gain transconductance factor of the M_i transistor in Fig.4(b) and V_{T0n} and V_{T0p} are the NMOS and PMOS transistor threshold voltages, respectively. Note that the *Input Block* constitutes the global input interface, thus its input range is also the controller input range.

Voltages V_p and V_{QT} are generated to keep M_{ET} and M_{EB} in the top mirror, as well as M_{QT} and M_{QB} in the bottom mirror, in the saturation region, which means $V_p \le V_{DD} - (V_{T0p} + \sqrt{I_Q/\beta_{ET}} + \sqrt{I_Q/\beta_{EB}})$ and $V_{QT} \ge V_{T0n} + \sqrt{I_Q/\beta_{QB}} + \sqrt{I_Q/\beta_{QT}}$. These voltages are obtained from the bias circuits in Fig.4(b). The parameter $\hat{x} \le \sqrt{I_Q/\beta_i}$ is chosen to obtain the desired smoothness, which improves for smaller values of $\hat{x}$ – see Fig.5. A bias current $I_Q/2$ is added to the differential pair output in Fig.4(b) to prevent the transistors at the top mirror from entering in weak inversion, which would degrade the dynamic response.

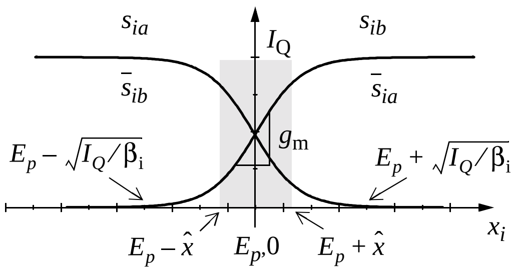


Fig. 5. Membership function support

B. Rule Block and Output

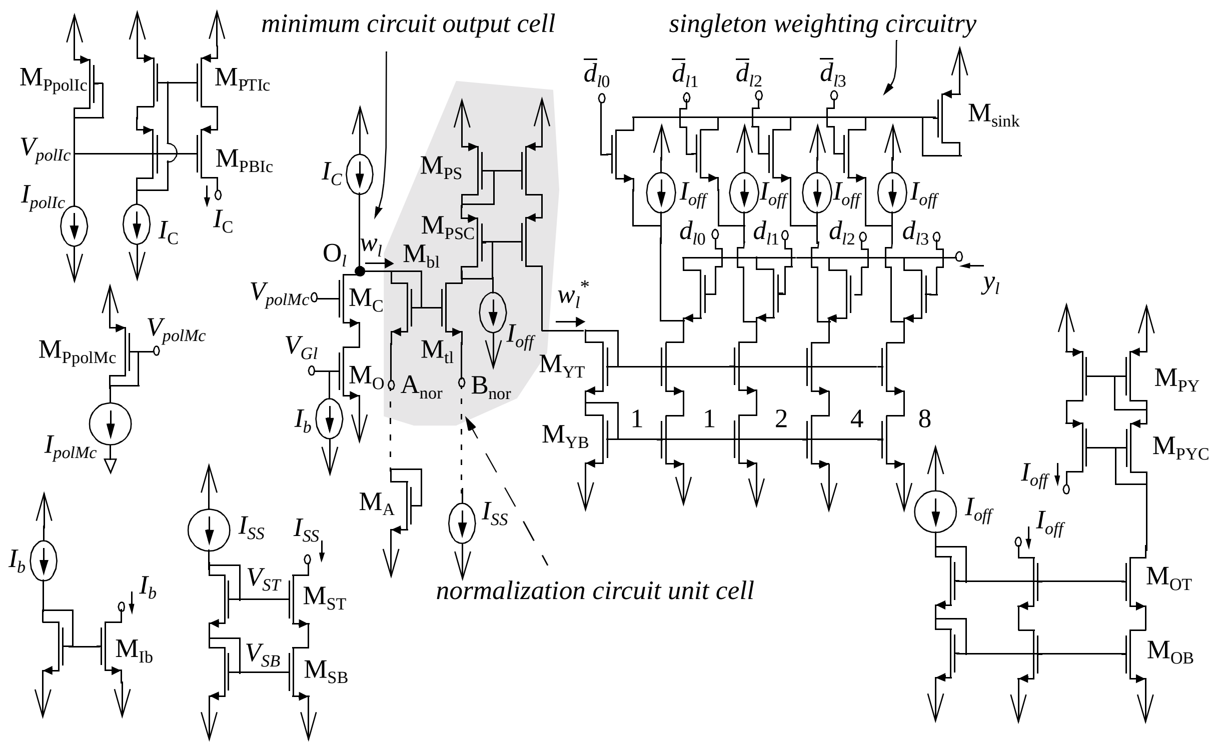
Fig.3(d) shows the operators within the generic high level *Rule Block*, while Fig.6 shows its CMOS implementation in the MFCON chip prototype, and Table 2 its main design equations. First, as already said above, the minimum circuit output cell provides the rule antecedent output w_l in Fig.6, where I_C is needed to perform the complement at output required by De Morgan's law and I_b provides a path to discharge node G_l. Thus, the normalization is performed in every rule block by the normalization circuit unit cell in Fig.6 [14] to obtain w_l^*. The bias current source I_{SS} and the sink transistor M_A are shared by all rule blocks in the analog core. Every normalization circuit output is reflected by a PMOS current mirror and weighted by the singleton value y_l^* – represented by the binary code $d_{l0}, \ldots, d_{l3}$ in Fig.6; also the current offset I_{off} is added to improve the dynamic performance. This weighting is carried out by a digitally programmable current mirror. However,

Table 2: Main design equations for the *Rule Block*.

Rule anteced-ent out-put	$\left[\begin{array}{c} w_l = I_C - I_Q/2 - max\{\overline{s_{1l}(x_1)}, \overline{s_{2l}(x_2)}, ..., \overline{s_{Ml}(x_M)}\} \quad = \\ = \; min\{s_{1l}(x_1), s_{2l}(x_2), ..., s_{Ml}(x_M)\} \end{array}\right]$
Normal-ization circuit output	$w_l^* = \frac{\beta_{\mathrm{tl}}}{\beta_{\mathrm{bl}}} w_l \left[1 + \frac{\eta(\mathbf{w})}{\sqrt{w_l}}\right]^2 \quad \eta(\mathbf{w}) = \frac{\sum_l \sqrt{w_l}}{2^M}\left(\sqrt{1 + \frac{N\frac{\beta_{\mathrm{bl}}}{\beta_{\mathrm{tl}}} I_{SS} - \sum_l w_l}{\sum_l \sqrt{w_l}^2}} - 1\right)$
Power	$[I_b + I_C + (y^* + 2)I_{off}]V_{DD} \le P \le [I_b + I_C + (y^* + 2)(I_{off} + I_{SS})]V_{DD}$
Maxi-mum output rules error contribu-tion	$\sigma^2(y_{I_{ss}}) = I_{SS}^2 \frac{\sigma^2(\beta_{\mathrm{SB}})}{\beta_{\mathrm{SB}}^2} + I_{SS} 4\beta_{\mathrm{SB}} \sigma^2(V_{T\mathrm{SB}}) + 2I_{off}^2 \frac{\sigma^2(\beta_{\mathrm{OB}})}{\beta_{\mathrm{OB}}^2} + I_{off} 8\beta_{\mathrm{OB}} \sigma^2(V_{T\mathrm{OB}}) + $ $+ 2(I_{off} + I_{SS})^2 \frac{\sigma^2(\beta_{\mathrm{PS}})}{\beta_{\mathrm{PS}}^2} + 8(I_{off} + I_{SS})\beta_{\mathrm{PS}} \sigma^2(V_{T\mathrm{PS}}) + 2(I_{off} + I_{SS})^2 \frac{\sigma^2(\beta_{\mathrm{YB}})}{\beta_{\mathrm{YB}}^2} +$ $+ 8(I_{off} + I_{SS})\beta_{\mathrm{YB}} \sigma^2(V_{TYB}) \; + 2I_{off}^2 \frac{\sigma^2(\beta_{\mathrm{PY}})}{\beta_{\mathrm{PY}}^2} + 8I_{off}\beta_{\mathrm{PY}} \sigma^2(V_{T\mathrm{PY}})$
Mini-mum output rules error contribu-tion	$\sigma^2(y_0) = 2(y^* I_{off})^2 \left[\frac{\sigma^2(\beta_{\mathrm{OB}})}{\beta_{\mathrm{OB}}^2} + \frac{4\beta_{\mathrm{SB}}}{I_{off}} \sigma^2(V_{T\mathrm{OB}})\right] + 2(y^* I_{off})^2 \left[\frac{\sigma^2(\beta_{\mathrm{PS}})}{\beta_{\mathrm{PS}}^2} + \frac{4\beta_{\mathrm{PS}}}{I_{off}} \sigma^2(\mathrm{V}_{T\mathrm{PS}})\right] +$ $+ \left(1 + \frac{1}{y^*}\right)(y^* I_{off})^2 \left[\frac{\sigma^2(\beta_{\mathrm{YB}})}{\beta_{\mathrm{YB}}^2} + \frac{4\beta_{\mathrm{YB}}}{I_{off}} \sigma^2(V_{T\mathrm{YB}})\right] + \left(1 + \frac{1}{y^*}\right)(y^* I_{off})^2 \left[\frac{\sigma^2(\beta_{\mathrm{PY}})}{\beta_{\mathrm{PY}}^2} + \frac{4\beta_{\mathrm{PY}}}{I_{off}} \sigma^2(V_{T\mathrm{PY}})\right]$

the dynamical programming of the mirror can cause large current spikes at output unless a special design is made. This design introduces the top branches at Fig.6 that are controlled by $\bar{d}_{l0,\, \dots l3}$, thus they drive some current when their associated switch in the output branch governed by $d_{l0,\, \dots l3}$ is ON, and vice versa. This guarantees that transistors in the current mirror are always in the saturation region, and never in the ohmic region. Since transitions from the ohmic to the saturation region were found to be the cause of the large spikes at output, the latter are reduced drastically with the proposed design.

Since the singleton weighting circuit provides a current as output, and the final processing step of the algorithm is the addition of these currents, we just wire up the rule block outputs, as Fig.3(b)



M_o (W/L)	M_C (W/L)	M_{bl} (W/L)	M_{tl} (W/L)	$M_{PSC}(W/L)$	M_{PS} (W/L)	M_A (W/L)	M_{YB} (W/L)	M_{YT} (W/L)
9µm/4.4µm	9µm/0.8µm	13µm/5µm	5µm/13µm	25µm/0.8µm	25µm/4µm	40µm/10µm	10µm/4µm	10µm/0.8µm
I_{SS}	I_{off}	$M_{PpolIc}(W/L)$	I_{polIc}	$M_{PTIc}(W/L)$	$M_{PBIc}(W/L)$	I_C	$M_{PpolMc}(W/L)$	I_{polMc}
10µA	10µA	5µm/18µm	5µA	16µm/10µm	16µm/0.8µm	30µA	10µm/8µm	5µA
$M_{Ib}(W/L)$	I_b	$M_{PY}(W/L)$	$M_{PYC}(W/L)$	$M_{ST}(W/L)$	$M_{SB}(W/L)$	$M_{OT}(W/L)$	$M_{OB}(W/L)$	
1.2µm/13.6µm	0.5µA	20µm/4µm	20µm/0.8µm	20µm/0.8µm	20µm/4µm	12µm/0.8µm	12µm/6µm	

Fig. 6. *Rule Block* schematic

illustrates, to exploit KCL and obtain a current which enters the global output node. However, we still reflect this output current with a current mirror to get a current that leaves the controller. The output current range is $[0\ ,\ I_{SS} \times (2^S - 1)]$, thus 150µA FSO (Full Scale Output) for the chip of this Paper. Fig.7 shows the output of the analog core for an interpolation interval of MFCON as measured in the laboratory.

With respect to errors, note first that those of systematic nature are minimized by using symmetrical structures and cascode transistors. Hence, most important errors are of random nature, due to transistor mismatches. The boundary at a given fuzzy set core [20] or interpolation point is determined by $\sigma^2(y) = \sigma^2(y_{I_{ss}}) + 3\sigma^2(y_0)$, where $\sigma^2(y_{I_{ss}})$ and $\sigma^2(y_0)$ are found in Table 2. The

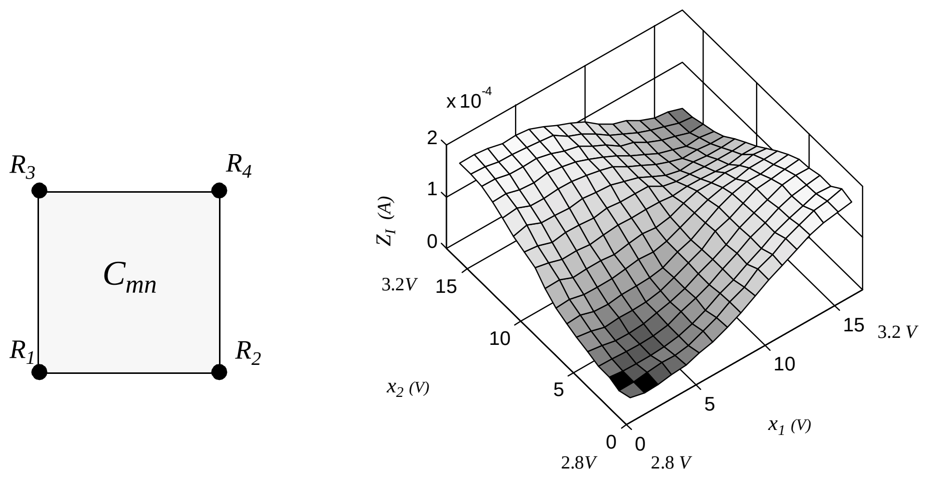


Fig. 7. Analog core output

latter are the variances associated to maximum and minimum output rules, respectively, whose nominal currents are I_{SS} and zero if $(W/L)_{\text{bl}}/(W/L)_{\text{tl}} \geq ((\sqrt{w_l} - \sqrt{w_{l(min)}})/\sqrt{I_{SS}})^2$ fulfils at the normalization circuit, where $w_{l(min)}$ is the minimun value at the normalization circuit input. Since many parameters have to be set to fit the estimated error into a specified boundary, a software mathematical assistant is required. The design of this Paper was made to get 3σ below 10% FSO. The choice in this chip for the offset current in Fig.6 was $I_{off} = I_{SS} = 10\mu A$ and $\sigma = 3.9\mu A$. However, smaller values of this offset current achieve a considerable reduction of the error, while the dynamic response is not much affected.

IV. MULTIPLEXING BLOCK SET

C. Interval Selector

This block comprises a battery of M A/D converter blocks, as Fig.8(a) shows. A simple and fast *A/D flash converter* – see Fig.8(b) – is the best choice since high-speed asynchronous operation is required, and low resolution is enough – note that the number of labels L is seldom higher than seven in most control applications. In addition, the converter comparators are designed to have hysteresis, and a priority coder is used to convert the thermometer code into a Gray code.

It is important to note that these converters are not in the signal path, but in the control path; also, they do not encode the input signal to be digitally processed, but just cluster the input space into regions. Therefore, the proposed fast flash A/D converters can readily accomplish the resolution requirements, and the input-output delay of the overall controller is hardly affected by the programming circuitry around the analog core which processes the input signal. Specifically, the estimated delay for this block in the controller presented in this Paper is 60 ns, for an input overdrive of 50mV from the reference voltage. This delay is around 10% of the measured global controller input-output delay.

There are three basic elements which make up the *Interval Selector* block in Fig.8(b): an array of linear resistors, a Gray encoder and an analog comparator with hysteresis. The *arrays of linear resistors* in the converters provide the reference values ε_{ij} in Fig.8(b), which determine the interpolation interval bounds. In addition, they also generate the whole set of programming values E_{ij} – see Fig.8(b) and Fig.3(b). Note that, if the partition is the same for all input dimensions, only one resistor array may be shared by the converters as long as the comparators have high impedance inputs, which saves area and power consumption. A conventional, serpentine-shaped polysilicon strip has been used to implement this element, while the *Gray encoder* has been designed with

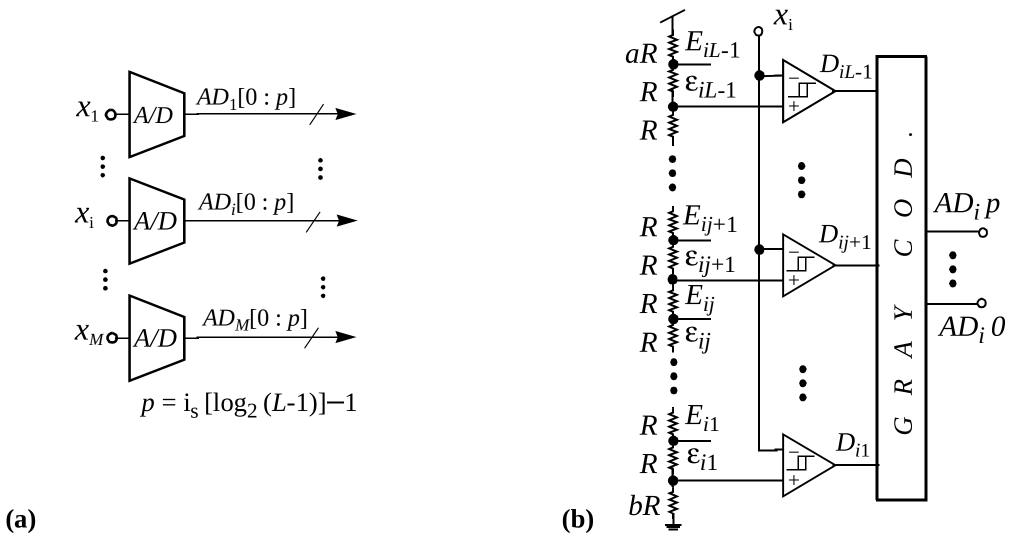
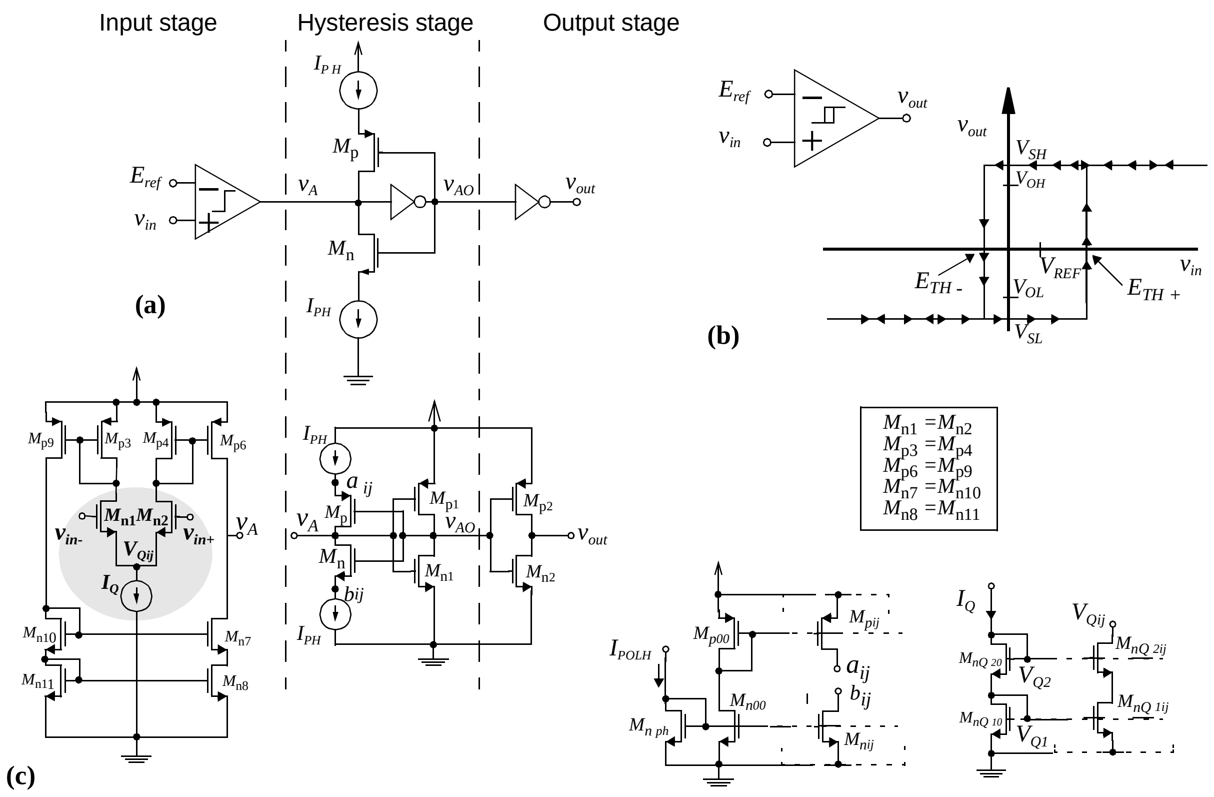


Fig. 8. Interval selector building blocks: (a) battery of converters, (b) A/D flash converter structure



$M_{n1} = M_{n2}$ (W/L)	$M_{p3} = M_{p4}$ (W/L)	$M_{p6} = M_{p9}$ (W/L)	$M_{n7} = M_{n10}$ (W/L)	$M_{n8} = M_{n11}$ (W/L)	$M_{nQ\ 20}$ (W/L)	$M_{nQ\ 10}$ (W/L)
150μm/2μm	30μm/2μm	60μm/2μm	20μm/2μm	20μm/2μm	10μm/2μm	20μm/5μm
$M_{nQ\ 2ij}$ (W/L)	$M_{nQ\ 1ij}$ (W/L)	M_n (W/L)	M_p (W/L)	M_{n1} (W/L)	M_{p1} (W/L)	$M_{n2}(W/L)$
10μm/2μm	20μm/5μm	4μm/6μm	4μm/2μm	2μm/2.4μm	3μm/0.8μm	2μm/0.8μm
M_{p2} (W/L)	M_{nph} (W/L)	M_{p00} (W/L)	M_{n00} (W/L)	M_{pij} (W/L)	M_{nij} (W/L)	
4μm/0.8μm	15μm/3μm	60μm/3μm	15μm/3μm	60μm/3μm	15μm/3μm	

Fig. 9. Comparator with hysteresis: (a) Symbol and ideal transference curve, (b) Multi-stage comparator with hysteresis used in MFCON, (c) Input stage, hysteresis and output stages schematic

CMOS logic gates of minimum size.

On the other hand, Fig.9(a) shows the simple three-stage *comparator with hysteresis* that has been implemented in MFCON, whose schematic is depicted in Fig.9(c). The input stage is a CMOS differential amplifier which provides high impedance inputs. The hysteresis stage, comprises a simple CMOS inverter comparator, and two complementary PMOS and NMOS current switches controlled by the inverter output v_{AO}, which implements the positive feedback to get hysteresis. Finally the output stage, is another simple CMOS inverter.

The switches in the second stage allow us to either connect or disconnect the current sources I_{PH} to the input inverter node v_A; this either adds or substracts current at this node, thereby forcing the reference voltage of the comparator to be either E_{TH+} or E_{TH-} , respectively (see Fig.9(b). First-order calculations obtains,

$$E_{TH+} = \frac{I_{PHn}}{g_m} \qquad E_{TH-} = -\frac{I_{PHp}}{g_m}, \tag{4}$$

where g_m is the small-signal transconductance of the differential stage; and $I_{PHn} \approx I_{PHp} \approx I_{PH}$, if $r_{onPH} \gg r_{DSn_{ON}}$ and $r_{opPH} \gg r_{DSp_{ON}}$, where r_{onPH} and r_{opPH} are the output resistances of the circuitry which implements the current sources I_{PH} – see right part of Fig.9(c), and $r_{DSn_{ON}}$ and $r_{DSp_{ON}}$ are the ON resistances of the NMOS and PMOS switches, respectively. The expression above shows that the hysteresis can be controlled by the current source I_{PH}, which is derived from an external bias current I_{POLH}.

The Table in Fig.9 shows the sizes of the transistors in Fig.9(c) as used in MFCON. They have been designed to cope with the requirements of gain, common mode range, power consumption and errors, from the simulations and analytical expressions reported in [27].

D. Rule Antecedent Programmer

The main requirements for this block are high-speed operation, as well as design simplicity, reliability and compactness. Because every element E_p in $PA_{C_{k_1 \dots k_M}}$ must be selected from the whole set of programming values E_{ij} in Fig.8(b), this block comprises a battery of M digitally controlled analog multiplexor cells, as Fig.10(a) shows. Fig.10(b) shows the internal structure of an analog multiplexor cell, which is composed of analog switches (CMOS transmission gates), and a Gray decoder which provides the digital control signals for the switches. Both elements are well-known CMOS building blocks and their design will not be explained here. The estimated input-output delay for this block, obtained from simulations, is negligible in comparison with the *Interval Selector* input-output delay.

E. Rule Consequents Programmer

Every element y^*_l in $PC_{C_{k_1 \dots k_M}}$ is an S-bit digital value, which must be selected from the whole set of singleton values y^*_k, for $1 \le k \le L^M$, hence this block must store and address efficiently up to $L^M \times S$ data bits in a digital memory. Besides, it must implement two ways to address these data, one for writing or reading data for external programming (i.e. to program the controller with the proper rule set), and another one for internal accesses, to get the set $PC_{C_{k_1 \dots k_M}}$ from the address provided by the *Interval Selector*. This internal read interface must be asynchronous and fast enough to cope with the speed and continuous time requirements of the controller. It must also provide the whole set ($PC_{C_{k_1 \dots k_M}}$) in one step and in the proper order.

Because of the input space lattice partition, every generic fuzzy rule R_{mn} belongs to 2^M different adjacent interpolation intervals, thus the corresponding singleton value y^*_{mn} can belong to different programming sets – see example in Fig.11(a). Besides, this shared rule is in different positions depending on the interval it contributes, hence the corresponding singleton value y^*_{mn}

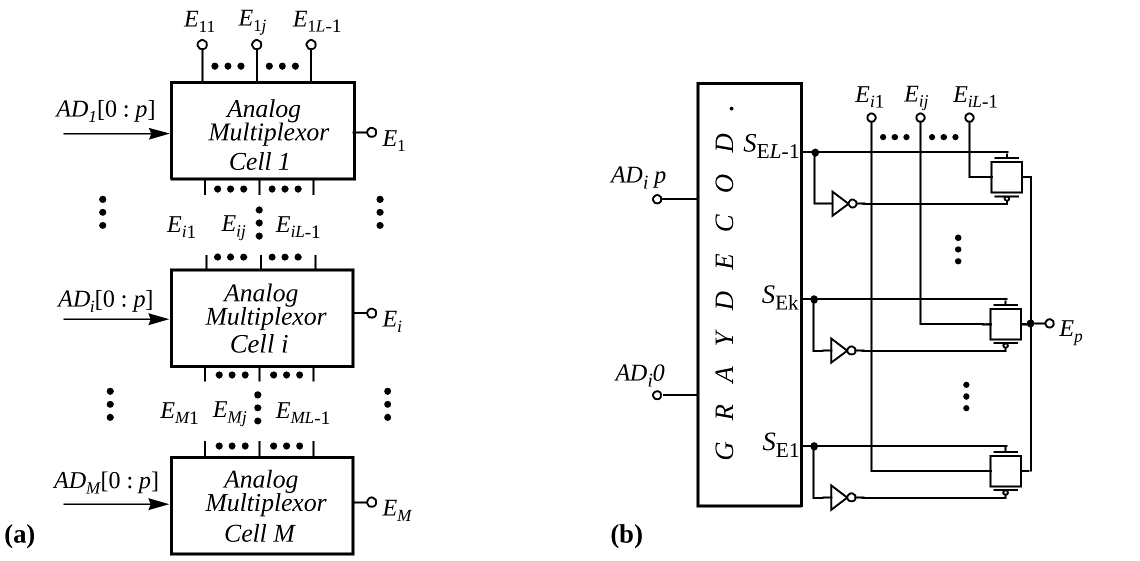


Fig. 10. *Rule Antecedent Programmer*: (a) Arrangement of Analog Multiplexor Cells, (b) Cell internal structure

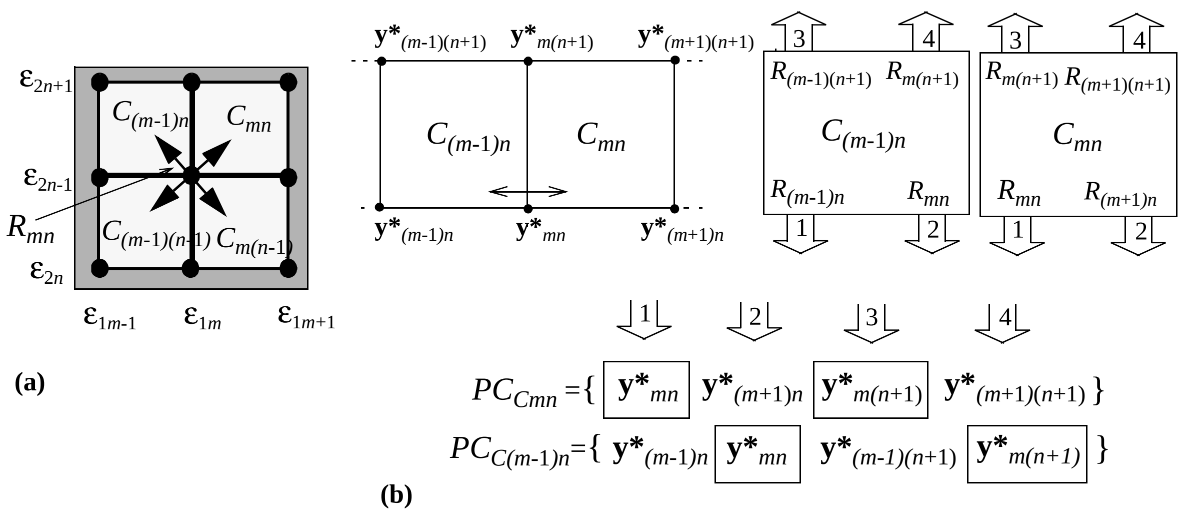


Fig. 11. (a) R_{mn} is active rule in four adjacent intervals: $(C_{mn}, C_{m(n-1)}, C_{(m-1)n}, C_{(m-1)(n-1)})$, (b) Rule order and programming bus example for two adjacent intervals

must appear in different locations in the programming bus – see Fig.11(b), because this bus is fixed in the Analog Core side as discussed in Section III.

The design of this block is based on the generic conceptual architecture for internal accesses depicted in Fig.12(a). The figure shows how the data are distributed into different blocks of *Memory Cells*, which contain subsets of singleton values which will never be addressed simultaneously. For a given address, the *Row Selector* selects one row per *Memory Cell* block simultaneously, thus all needed singleton values are ready to be accessed. At the same time, the

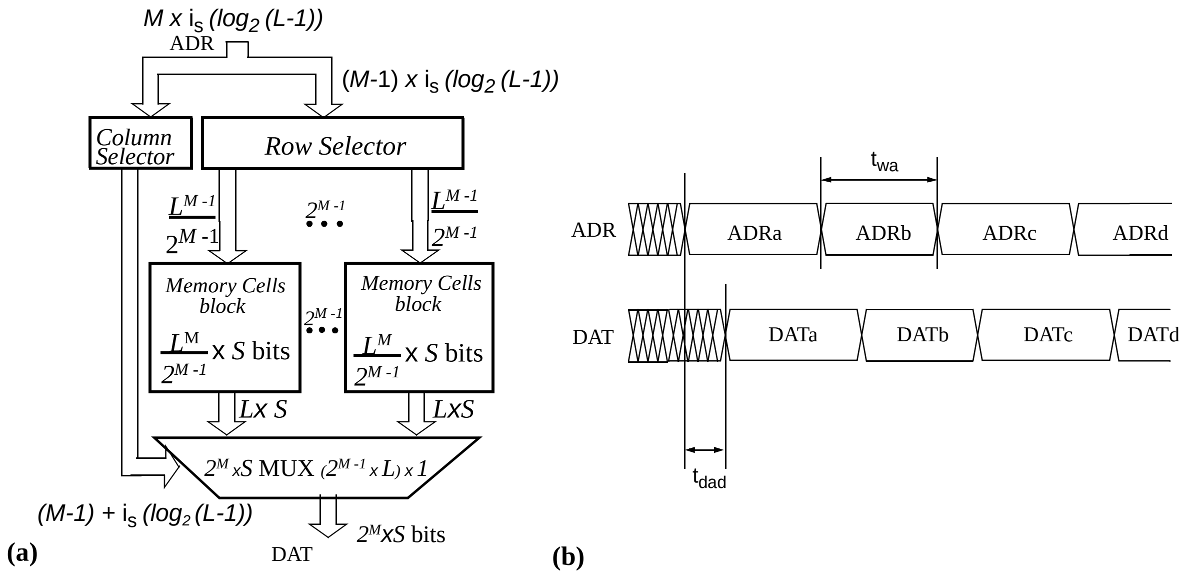


Fig. 12. (a) Rule Consequents Programmer conceptual internal architecture, (b) Internal accesses interface timing

Column Selector controls the multiplexor to locate properly every singleton value in the programming bus. Fig.12(b) illustrates the timing of these accesses.

Fig.13 illustrates the organization and interfaces of the building blocks in the CMOS implementation of the *Rule Consequent Programmer* in MFCON based on Fig.12(a). These blocks are *Memory Cells*, *Row Selector, Column Selector for Internal Accesses* and *Column Selector for External Accesses*.

The basic building block of the *Memory Cell blocks* in MFCON is shown in Fig.14(a). It comprises four one-bit *memory basic cells*, associated to two different *Memory Cell Blocks* in Fig.12(a). Specifically, *row n* in Fig.14(a) comprises one-bit basic memory cells *(m,n)* and *(m+1,n)*, which belong to the *Memory Cell Block 2*, while *row n+1* comprises the cells *(m,n+1)*

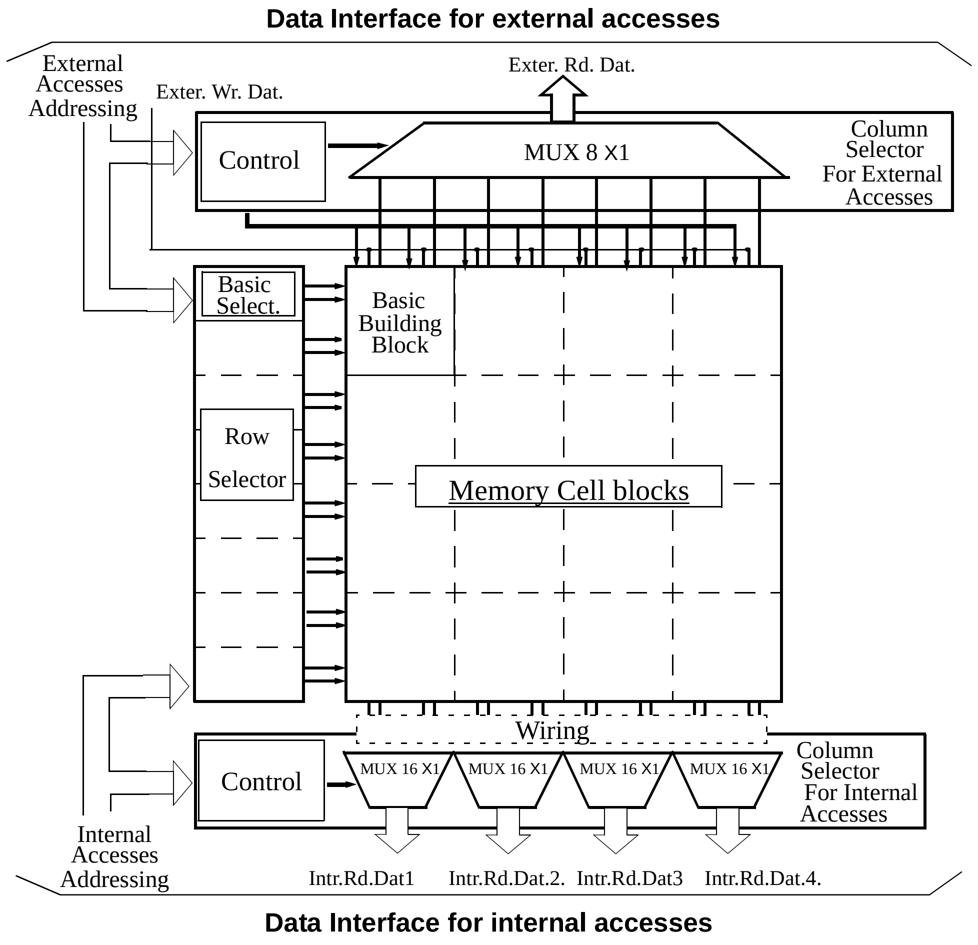


Fig. 13. Organization of the different building blocks used in the CMOS implementation of the *Rule Consequent Programmer* in MFCON.

and *(m+1,n+1)* which belong to the *Memory Cell Block 1.* For internal read accesses, switches s_m and s_{m+1} are open, thus memory cells *(m,n)* and *(m+1,n)* are in different column lines in the *Memory Cell Block 2,* while *(m,n+1)* and *(m+1,n+1)* are in different column lines in the *Memory Cell Block 1.* Therefore, the four stored bits can be read simultaneously if both row lines are selected for internal accesses. However, for external accesses, switches s_m and s_{m+1} are closed, so memory cells *(m,n)* and *(m+1,n)* belong to the same column line, as well as *(m,n+1)* and *(m+1,n+1)*. Thus, the whole block is configured to be accessed as a two-row, two-column conventional memory block, and the memory is a conventional RAM for external accesses. Fig.14(b) shows the elements in one column of the basic building block in Fig.14(a). The *one-bit basic memory cell* used in MFCON (enclosed in dash square in Fig.14(b)), is a single-ended bit-line static CMOS cell [28], which is common in register files and multiport memories. Because of the flexibility in changing its configuration, and its robustness [29], this cell results very suitable for the reconfiguration requirements of this application. Fig.14(b) also shows the switches for reconfiguration and the latches for regenerating the logic levels. The transistor sizes in the memory

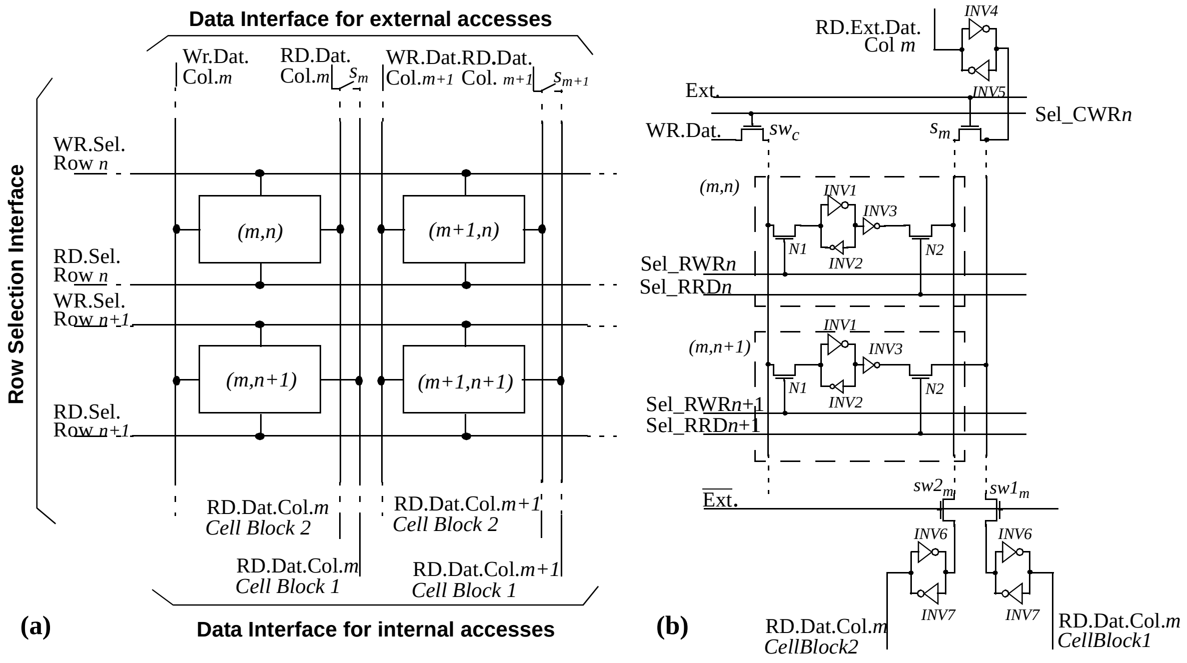


Fig. 14. (a) *Memory Cell* basic building block, (b) Detail of the elements of one column: Basic Memory Cell (enclosed in dashed square), reconfiguration switches and column sensed latches

cell have been determined from the design recommendations in [28].

The *Row Selector* activates the proper row selection lines after decoding the access type (read/ write) and origin (internal/external) and the corresponding subset of address lines – see Fig.13. For external accesses the block works as a conventional binary decoder, while for internal accesses it works as a Gray decoder which activates simultaneously several row selection lines per access, one per *Memory Cell* block considered. Fig.15(a) illustrates the conceptual architecture and interfaces, while Fig.15(b) illustrates the basic building block of the *Row Selector* as implemented in MFCON.

Because internal accesses are always for reading, the *Column Selector for Internal Accesses* comprises a battery of properly sized multiplexors with shared control lines, as Fig.15(c) illustrates, where the conceptual architecture and interfaces of this block in MFCON are shown. Because column data lines are properly wired to the multiplexor inputs, the control block can be

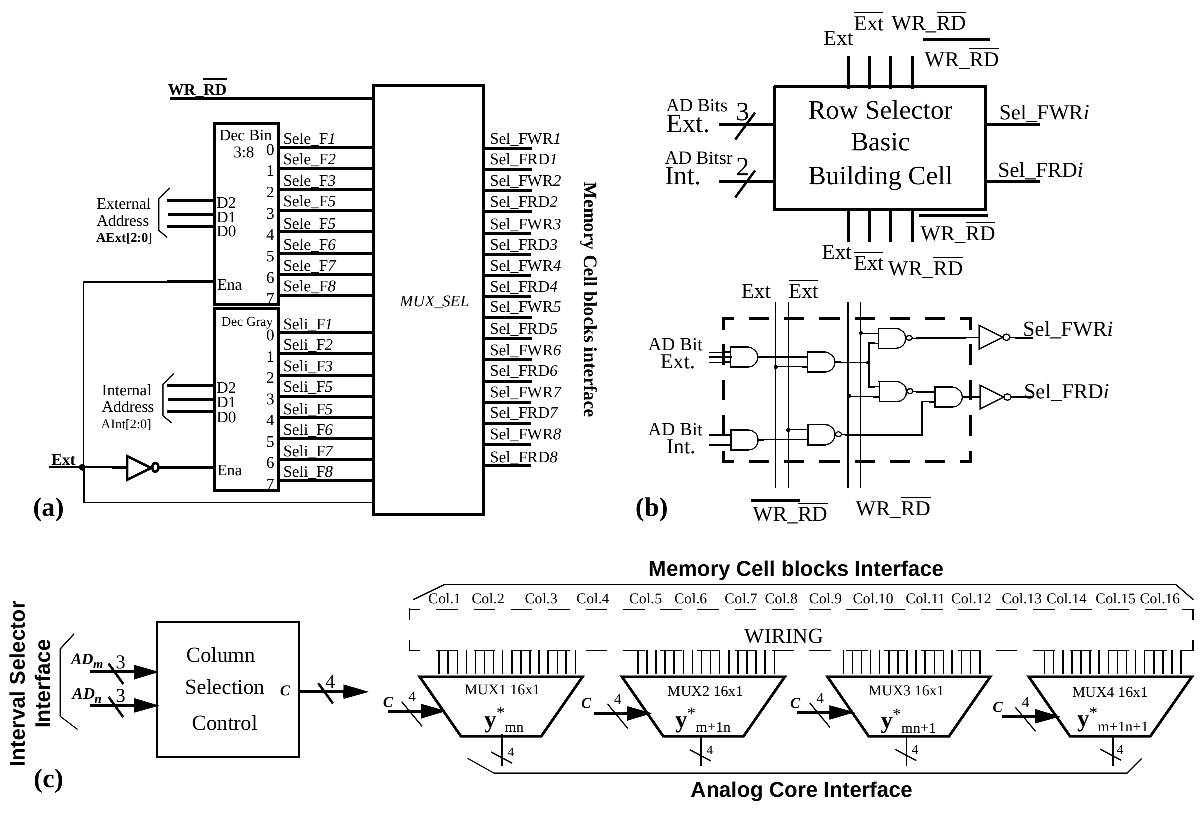


Fig. 15. (a) *Row Selector* conceptual architecture, and (b) Basic building block, (c) *Column Selector for Internal Accesses* conceptual architecture

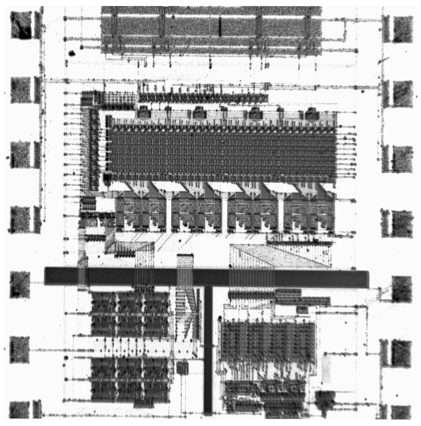
very simple.

The *Column Selector for External Accesses* comprises a control block and a multiplexor per output line – see Fig.13. The control block contains a decoder which activates the proper column selection lines by decoding a subset of the corresponding address lines for write accesses. On the other hand, the multiplexors are controlled by the same subset of address lines to provide the data in read accesses.

V. STATIC AND DYNAMIC PERFORMANCES OF MFCON

The MFCON controller prototype has been integrated in a single-poly, double-metal CMOS 0.7μm technology offered in EUROPRACTICE. The chip was simulated with HSPICE and designed with Design Frame Work II. It implements the architecture of Fig.2 with two inputs ($M = 2$), eight labels per input ($L = 8$), and four bits per singleton ($S = 4$). Fig.16(a) shows a microphotograph of the chip, and Fig.16(b) shows the floorplan of the chip with the blocks and their sizes. Conservative layout strategies have been adopted; particularly, the analog circuitry has been placed far away from the digital one, with large isolation guard rings in between [1]. Also, in order to further reduce interferences between these parts, separate pins and lines have been employed for the analog supply, the digital supply, and the ground [30].

(a)



(b)

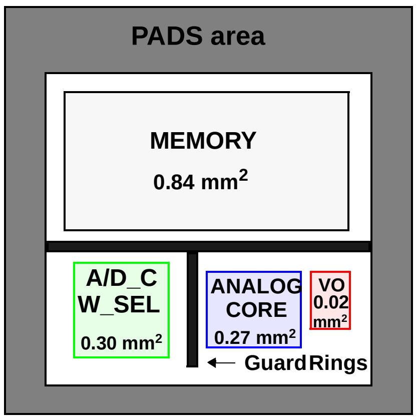


Fig. 16. (a) Microphotography and (b) Floor plan of the chip

1. The area occupation of this chips, around 5mm^2, is much larger than needed. Since 5mm^2 is the minimum area for MPW projects, largely conservative layout floorplanning strategies have been adopted regarding the separation of analog and digital parts.

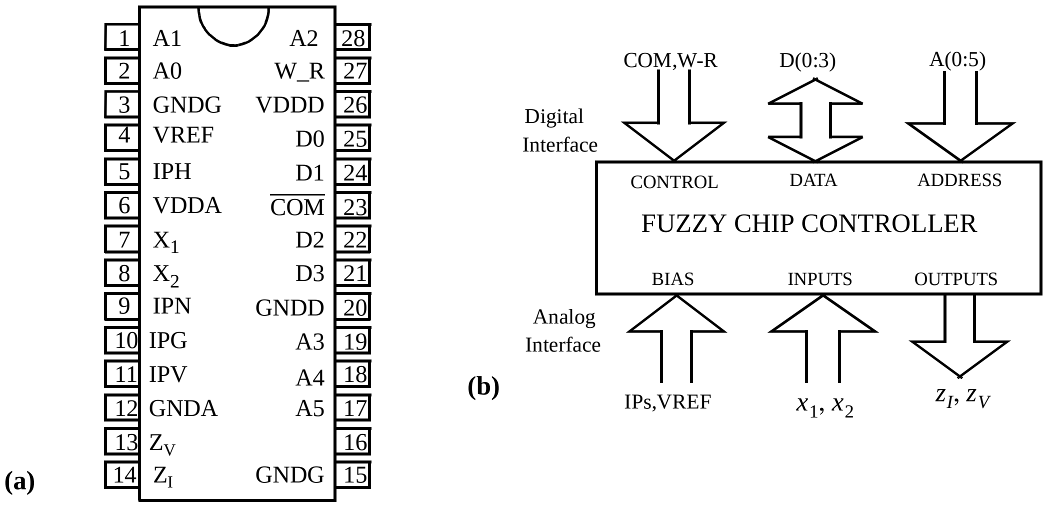


Fig. 17. Fuzzy controller chip: (a) pin-out; (b) interfaces

Fig.17(a) shows the chip pin-out, while Fig.17(b) shows the digital and analog interfaces with the pins grouped in buses. On the one hand, the digital interface corresponds to a typical asynchronous peripheral for microcontroller-based systems, with an input address bus (A{0:5}), a bidirectional data bus (D{0:3}) and a control bus (COM, W-R). On the other hand, the analog interface is composed of the controller input and output signals, and a few off-chip bias signals to simplify testing of the prototype. These signals should be generated on-chip in a marketable final version.

The fuzzy controller inputs are x_1 and x_2, which are driven by voltage mode signals in the range from 2.0V to 4.8 V. The primary fuzzy controller output is z_I – a current signal in the range from 0μA to 150μA. A voltage output is also provided at z_V. This voltage is obtained by applying a replica of z_I to an on-chip polysilicon resistor. Every singleton value y^*_{mn} is a four-bit digital word which encodes sixteen uniformly distributed analog values in the output current range.

Fig.18 and Fig.19 show some experimental results obtained from the test environment. Fig.18(a) depicts a section of a measured DC control surface, and Fig.18(b) shows the transient response to a step input. The former illustrates the response (bottom) to a ramp in one input (top) while the other input remains constant, in a kind of mexican hat surface. The latter corresponds to

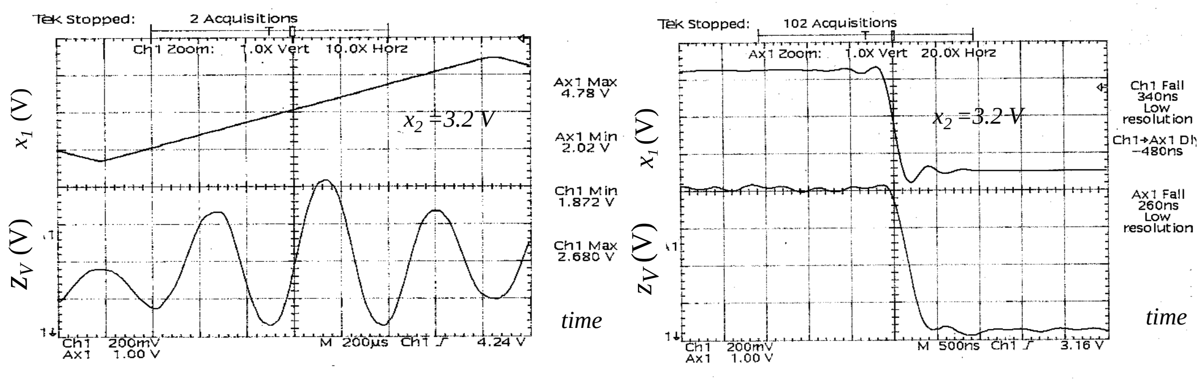
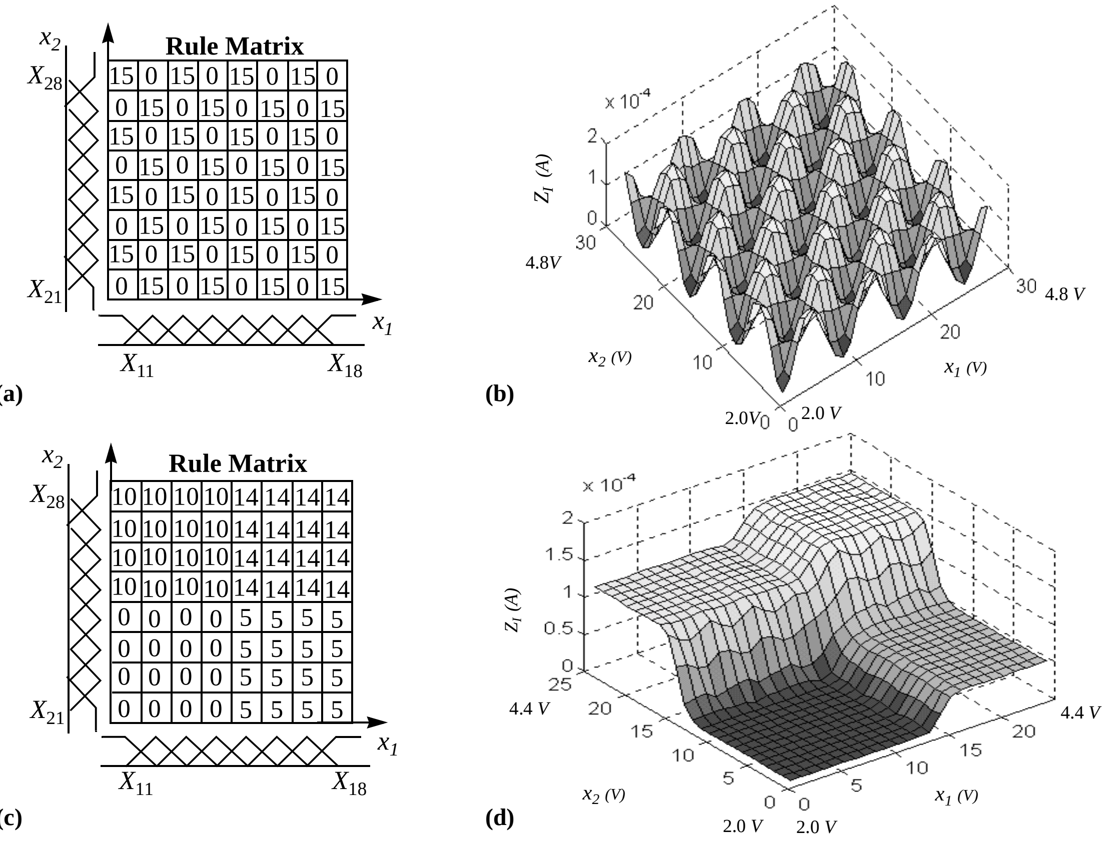


Fig. 18. Measured results: (a) DC nonlinear control surface section, (b) transient falling edge step response.

a falling edge in one input which forces the output to change from its maximum to its minimum value, as well as to jump to a different interpolation interval, which means a dynamic programming of the analog core. The measured delay time is around 500ns. Since the oscilloscope is not able to sense currents, previous measurements are voltages in the output z_V. With regard to Fig.19(b) and Fig.19(d), they are built with data obtained from a data acquisition board, where the current output



15	0	15	0	15	0	15	0
0	15	0	15	0	15	0	15
15	0	15	0	15	0	15	0
0	15	0	15	0	15	0	15
15	0	15	0	15	0	15	0
0	15	0	15	0	15	0	15
15	0	15	0	15	0	15	0
0	15	0	15	0	15	0	15

10	10	10	10	14	14	14	14
10	10	10	10	14	14	14	14
10	10	10	10	14	14	14	14
10	10	10	10	14	14	14	14
0	0	0	0	5	5	5	5
0	0	0	0	5	5	5	5
0	0	0	0	5	5	5	5
0	0	0	0	5	5	5	5

Fig. 19. Control surface measured results: (a) programming rule matrix I, (b) nonlinear control surface I, (c) programing rule matrix II and (d) nonlinear control surface II

of the chip is externally converted into a digital word and processed. Both figures illustrate the ability of the controller to interpolate functions. Finally, the measured chip power consumption was around 16mW, which is obtained by sensing the current from the supply voltage sources.

In addition, Fig.20 shows results from an example application with the chip in a control loop. The task is the start of a DC motor controlled with a PWM DC-DC switching converter at 100kHz. Fig.20(a) shows the control surface, while Fig.20(b) and Fig.20(c) show the motor speed (top) and armature current (bottom) for both, the direct start and controlled soft-start, respectively. Note that the speed rise time is similar in both, direct and controlled cases, while the initial current spike is not present in the controlled case. Fig.20(d) shows how smooth the control is in the range of microseconds. Finally, Fig.20(e) shows that the current is not got under control if the same strategy is implemented with a microcontroller.

VI. CONCLUSIONS

The mixed-signal fuzzy controller chip presented in this Paper attains the performance levels of fully analog controllers while overcoming their inherent limitations in terms of programmability and complexity. This is achieved by employing the multiplexing strategy and architecture presented by the authors in [1]. The data in Table 3 are intended to compare the MFCON chip to

Table 3: CMOS Analog implementation of Fuzzy Controllers

Features/ CMOS Chips	Manaresi [12]	Guo [13]	Vidal [14]	Baturone [15]	MFCON
Complexity	9rules@2inputs @2output	13rules@3inputs @1output	16rules@2inputs @1output	9rules@2inputs @1output	64rules@2inputs @1output
Technology	0.7µm CMOS	2.4µm CMOS	1µm CMOS	2.4µm CMOS	0.7µm CMOS
Power	44mW@5V	550mW@10V	8.6mW@5V	21mW@5V	16mW@5V
In/out Delay	570 ns	160 ns	471 ns	2000 ns	500 ns
Precision	No data	No data	6.5% (3σ)	No data	7.8%(3σ)
Interface (in@out)	volt@volt	volt@volt	volt@current	volt@volt	volt@current
Programmability	HIGH	LOW	HIGH	HIGH	HIGH
Area	1.9 mm^2	16.2 mm^2	1.6 mm^2	1.1 mm^2	2.65 mm^2 without pads

other analog continuos-time CMOS controller chips which implement a similar inference algorithm.

From Table 3 it is seen that the MFCON chip implements much more rules that the others. Note also that this increased number of rules is not accompanied by a significant power consumption increase; neither by an operation speed drop. Actually, regarding power consumption, only the

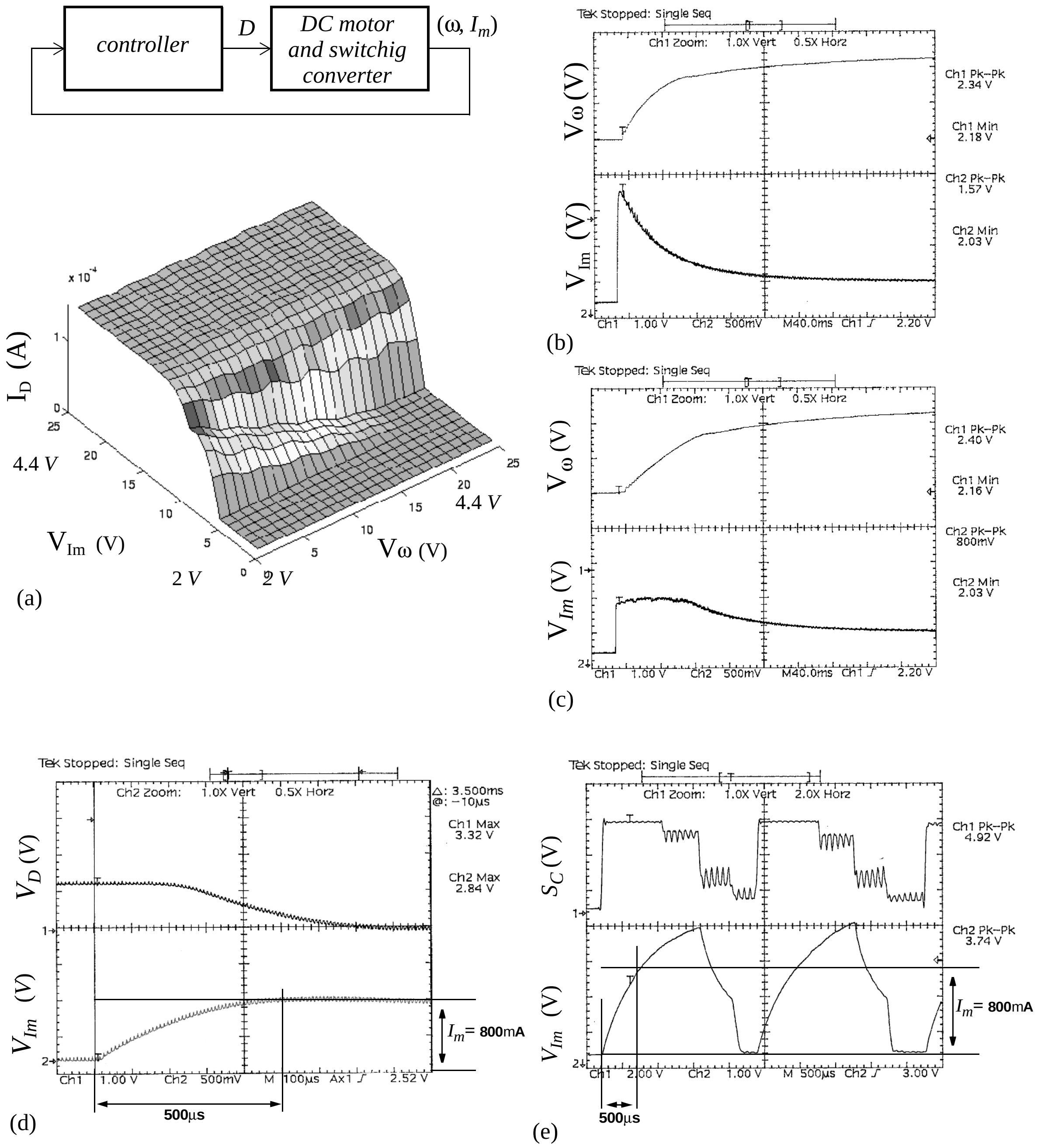


Fig. 20. Start of a DC motor control example: Control surface (a); curves for the speed V_ω and armature current V_{Im} for direct (b) and controlled (c) start; controlled start detail with MFCON (d) and controlled start detail with microcontroller (e).

prototype in [14] consumes less power than the MFCON chip, although it realizes four times less rules. Regarding speed, the prototype in [13] is faster, although its power consumption is much larger and its complexity much smaller.

The data in Table 3 confirms that the proposed strategy actually overcomes the limitations of purely analog controllers while keeping their performance advantages. For instance, a fully analog controller designed by the authors [14] using similar circuitry yields 470ns and 8.6mW for 16 rules, while the MFCON chip yields 500ns and 16mW for 64 rules. Advantages of the proposed architecture become more evident as the number of rules and inputs increases [1]. Furthermore, the proposed architecture is very well suited for the modular generation of complex fuzzy controllers. Since it is based on the dynamical programming of an analog core, whose size – 2^M rules – depends just on the number of inputs M, we could have a reduced set of well-designed analog cores (one input, two inputs, three inputs...) as cells. Every cell is valid for building controllers with a different number of rules, while their performances must be quite similar in terms of errors, power consumption and input-output delay. These cells could even be integrated in conventional microcontrollers which would provide a very good control performance.

REFERENCES

[1] F. Vidal-Verdú, R. Navas-González and A. Rodríguez-Vázquez, "Multiplexing architecture for mixed-signal CMOS fuzzy controllers", *Electronics Letters*, vol.34 no.14 pp. 1437-1438, July 1998

[2] M. Brown and C. Harris, *NeuroFuzzy Adaptive Modeling and Control*, prentice Hall International, 1994.

[3] Werbos et al. "Neural Networks, System Identification and Control in the Chemical Process Industries", in *Handbook of Intelligent Control*, Eds White D. A., Van Nostrand Reinhold, NY, Chapter 10, 1992

[4] K. Hirota y M. Sugeno, *Industrial Applications of Fuzzy Thechnology in the World*. World Scientific. 1995.

[5] B. K. Bose, " Expert Systems, Fuzzy Logic, and Neural Network Applications in Power Electronic and Motion Control" *Proceedings of the IEEE*, Vol. 82, pp. 1303-1323, August 1994.

[6] K. Young-Ho and K. Lark-Kyo, "Design of Neuro-Fuzzy Controller for the Speed Control of a DC Servo Motor", *Proceeding of the V International Conference on Electrical Machines and Systems*, vol. 2, pp. 731-734, August 2001.

[7] J.O.P. Pinto, B.K.Bose, L.E. Borges da Silva, "A Stator-Flux-Oriented Vector-Controllerd

Induction Motor Drive with Space Vector PWM and Flux-Vector Synthesis by Neural Networks", *IEEE Transactions on Industry Applications*, vol. 37, no. 5, Sep./Oct. 2001.

[8] M. Criscione, G. Giustolisi, A. Lionetto, M. Muscarà, G. Palumbo, "A Fuzzy Controller for Step-Up DC/DC converters", *The 8th. IEEE Inter. Conf. on Electronics, Circuits and Systems (ICECS 2001)*, vol. 2 pp.977-980, 2001.

[9] E. Franchi, N. Manaresi, R. Rovatti, A. Bellini, and G. Baccarani, "Analog Synthesis of Nonlinear Functions Based on Fuzzy Logic", *IEEE Journal of Solid-State Circuits*, vol. 33, no. 6, June 1998.

[10] Sanchez-Solano, S., Barriga A., Jiménez C..J., Huertas J.L. "Design and Application of Digital Fuzzy Controllers", *Procc. IEEE Int. Conf. on Fuzzy Systems*, pp. 869-874, Barcelona, 1997.

[11] Gabrielli, A. Gandolfi E., Masetti M., "Design of a Family of VLSI High Speed Fuzzy Processors", Procc. *IEEE Conf on Fuzzy Systems*, pp. 1099-1105, New Orleans, 1996.

[12] N.Manaresi, R. Rovatti, E. Franchi, R. Guerrieri, and G. Baccarani, “A Silicon Compiler of Analog Fuzzy Controllers: From Behavioral Specifications to Layout”. *IEEE Trans. on Fuzzy Systems*, Vol. 4, pp. 418-428, November 1996.

[13] S. Guo, L. Peters, and H. Surmann, “Design and Application of an Analog Fuzzy Logic Controller”. *IEEE Trans. on Fuzzy Systems*, Vol. 4, pp. 429-438, November 1996.

[14] A. Rodríguez-Vázquez, R. Navas, M. Delgado-Restituto and F. Vidal-Verdú, "A Modular Programmable CMOS Analog Fuzzy Controller Chip", *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal processing*, Vol. 46, No.3 pp. 251-265, March 1999.

[15] I. Baturone, S. Sánchez-Solano and J.L. Huertas, "Towards the IC Implementation of Adaptive Fuzzy Systems". *IEICE Transactions on Fundamentals*, Vol. E-81-A, No. 9, pp. 1877-1885, 1998.

[16] Teresa Serrano-Gotarredona, Andreas G. Andreou, and Bernabé Linares-Barranco, "A Programmable VLSI Filter Architecture for Application in Real Time Vision Processing Systems," International Journal of Neural Systems, Special Issue on New Trends in Neural Network Implementations, vol. 10, No. 3, pp. 179-190, June 2000.

[17] G. Cauwenberghs, M. A. Bayoumi, *Learning on Silicon. Adaptive VLSI Neural Systems*, Kluwer Academic Publishers, 1999.

[18] F. Vidal-Verdú, R. Navas-González and A. Rodríguez-Vázquez, "Circuits for on-chip Learning in Neural-Network" in *Learning on Silicon. Adaptive VLSI Neural Systems*, Edited by G. Cauwenberghs, M. A. Bayoumi, Chapter 9, Kluwer Academic Publishers, 1999.

[19] J.S.R. Jang and C.T. Sun, “Neuro-Fuzzy Modeling and Control”. *Proceedings of the IEEE*, Vol. 83, pp. 378-406, March 1995.

[20] L.X.Wang, *A Course in Fuzzy Systems and Control*, Prentice-Hall PTR 1997.

[21] T. Miki, T. Yamakawa, “Fuzzy Inference on an Analog Fuzzy Chip”, *IEEE Micro*, vol. 15, no. 4, pp. 58-66, 1995.

[22] I. Baturone, S. Sanchéz-Solano, A. Barriga and J.L. Huertas, “Implementation of CMOS Fuzzy Controllers as Mixed-Signal Integrated Circuits”, *IEEE Tran. on Fuzzy Systems*, vol.5, no.1,Feb.1997.

[23] X.J.Zeng and M.G.Singh, “Decomposition Property of Fuzzy Systems and Its Applications”, *IEEE Tran. on Fuzzy Systems*, vol. 4, no. 2, pp. 149-165, May1996.

[24] X.J.Zeng and M.G.Singh, “Approximation Accuracy Analysis of Fuzzy Systems as Function Approximators”, *IEEE Tran. on Fuzzy Systems*, vol. 4, no. 1, pp. 44-63, Feb.1996.

[25] I. Baturone, A. Barriga, S. Sanchéz-Solano, and J.L. Huertas, “Mixed Signal Design of a Fully Parallel Fuzzy Processor”, *Electronics Letter*, vol.34, no.5, pp.437-438, March.1998

[26] F. Vidal-Verdú and A. Rodríguez-Vázquez, "Using Building Blocks to Design Analog Neuro-Fuzzy Controllers " *IEEE Micro*, vol. 15, no. 14, pp. 49-57, August 1995.

[27] A.Rodriguez-Vázquez et al. *Encyclopedia of Electrical and Electronics Engineering*, Wiley 1999.

[28] H.Shinohara, N.Matsumoto, K.Fujimori,Y.Tsujihashi,H.Nakao,S.Kato,Y.Horiba, and A.Tada "A Flexible Multiport RAM Compiler for Data Path", *IEEE Journal of Solid-State Circuits*, vol.26, no.3, pp. 343-349, March1991.

[29] N.H.E.Weste and K. Eshraghian, *Principles of CMOS VLSI Design*, Addison-Wesley 1994.

[30] M. Ingels and M.S.J.Steyaert, "Design Strategies and Decoupling Techniques for Reducing the Effects of Electrical Interference in Mixed-Mode IC´s". *IEEE Journal of Solid-State Circuits*, vol. 32, no. 7, July 1997.