An integrated circuit realization for a piecewise linear function

Martín Di Federico, Víctor M. Jiménez-Fernández, Pedro Marcelo Julián and Osvaldo Agamenoni†
Luis Hernández-Matínez and Arturo Sarmiento-Reyes‡

†*Universidad Nacional del Sur, Bahía Blanca, Argentina*
mdife@uns.edu.ar
‡*Instituto Nacional de Astrofísica, Óptica y Electrónica, México.*
luish@inaoep.mx

***Abstract*— In this paper we present an integrated circuit (IC) realization for a three dimensional piecewise linear (PWL) function. The IC is designed and fabricated in a standard CMOS 0.5 μm technology. It includes three analog or 8 bit-coded inputs. The output of the circuit is a digital word with 8-bit precision which represents the value of the PWL function at the three-dimensional input. Programmability is considered in the chip architecture. The PWL function is programmed in an external 4kB RAM memory addressed by a 12-bit word.**

***Keywords*— Piecewise linear function, integrated circuit realization, circuit architecture**

I Introduction

In recent papers [1], [2] two different (analog and mixed-signal,respectively) implementations of the PWL approximation technique proposed by Julián *et al.* has been proposed. In particular, the circuit architecture proposed in [2] provides a piecewise-linear inputs-output relationship based on a weighted sum of the so-called α-functions which are defined over a domain partitioned by simplices. Each α-function is of a local nature, since it is different from "0" only over a reduced number of simplices of the domain. As a consequence, the value of the approximate PWL function can be obtained, for any n-dimensional input vector x, by combining a limited subset of the basis functions weighted by their corresponding coefficients [3], [4]. Then all basis functions perform basically the same operation and the difference between two basis functions is that they operate over two different regions of the domain. Therefore, the evaluation can be done using only one function circuit block and an algorithm to shift the inputs [5], [6], [7]. For every evaluation point, all nonzero basis functions need to be evaluated, weighted and added. This principle has been considered in the architecture of the IC presented in this paper.

II Mathematical background

Let us consider a domain $\mathbf{S}$ subdivided with a simplicial partition H using a grid step δ.

It produces a set of vertices

$$\mathbf{V}_s = \{\mathbf{v} \in \mathbf{R}^n : v_i = -1 + m_i \times \delta, i = 1, \cdots, n\} \tag{1}$$

where $0 \leq m_i \leq m$, and $m \times \delta = 2$. The grid step δ is the size of the division on every coordinate axis.
Also, let us consider a family of PWL functions defined over the simplicial partition. It constitutes a linear vector space $PWL_H(\mathbf{S})$ whose dimension is $q = (m+1)^n$.
Any function $F \in PWL_H[\mathbf{S}]$ can be expressed in vectorial form as

$$F(\mathbf{x}) = \mathbf{c}^T \mathbf{\Lambda}(\mathbf{x}) \tag{2}$$

where $\mathbf{c} \in \mathbf{R}^q$ is the so called vector of parameters, $\Lambda = [\alpha_1, \cdots, \alpha_q]$, and α_i, for $i = 1, \cdots, q$, is a PWL basis function. In the formulation under consideration, each basis function $\alpha_i \in PWL_H[\mathbf{S}]$ is defined as

$$\alpha_i(\mathbf{v}_j) = \left\{ \begin{array}{ll} 1, & \text{if} \quad i = j \\ 0, & \text{if} \quad i \neq j \end{array} \right\} \tag{3}$$

where $\mathbf{v}_j \in V_S$, for every $i = 1, \cdots, q$.
It has been shown in [5], that any point $\mathbf{x} \in \mathbf{S}$ can be decomposed as

$$\mathbf{x} = \sum_{l=1}^{r} \mu_{i_l} \mathbf{v}_{i_l} \tag{4}$$

where the terms in the expansion satisfy $0 \leq \mu_{i_l} \leq 1$, $\mathbf{v}_{i_l} \in V_S$, for every $l = 1, \cdots, r$, with $r = n + 1$, and $\sum_{i_l}^{r} \mu_{i_l} = 1$.
Also in [5], it has been proved, that any function of the basis given by (3) satisfies

$$\left\{ \begin{array}{ll} \alpha_i(\mathbf{x}) = \mu_{i_l}, & \text{for} \quad l = 1, \cdots, r \\ \alpha_p(\mathbf{x}) = 0, & \text{for} \quad p \neq \{i_1, \cdots, i_r\} \end{array} \right\} \tag{5}$$

where $\mathbf{x} \in \mathbf{S}$ is a point in the form of (4).
The evaluation of function (2) at a point (4) gives a result

$$F(\mathbf{x}) = \sum_{i=1}^{q} c_i \alpha_i \left(\sum_{l=1}^{r} \mu_{i_l} \mathbf{v}_{i_l} \right) \tag{6}$$

As F is linear inside simplex S^i, then $F(\mathbf{x}) = \sum_{i=1}^{q} c_i \sum_{l=1}^{r} \mu_{i_l} \alpha_i(\mathbf{v}_{i_l})$, and after exchanging the sumation terms, we have

$$F(\mathbf{x}) = \sum_{l=1}^{r} \mu_{i_l} \sum_{i=1}^{q} c_i \alpha_i(\mathbf{v})_{i_l} \tag{7}$$

Finally, if we consider the relation given in (3), then (7) reduces to

$$F(\mathbf{x}) = \sum_{l=1}^{r} c_{i_l} \mu_{i_l} \tag{8}$$

From this equation, we observe that in order to calculate the value of $F(\mathbf{x})$ at the input $\mathbf{x}$, we need to determine the a weighted sum that involves the parameters c_i) and μ_{il}, for $l = 1, \cdots, r$.

III Chip architecture

The architecture of the IC presented in this paper, represents the circuit implementation of eq.(8). Fig.1 is a block diagram which describes a general scheme for implementing eq.(8).

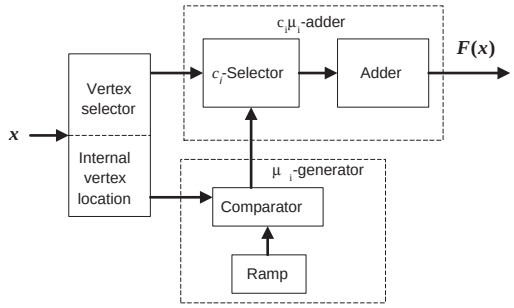

Figure 1: Block diagram architecture for implementing eq.(8)

In references [5] and [6], it was reported a proposal for obtaining the μ parameters. Such proposal consists in a set of comparators which comparate the input signals (x vector), with a ramp. This idea has been considered in the μ_i-generator block of Fig.1. Notice that input x has been decomposed in two subsets, any of them considers all the fuction domain and it selects a specific vertex, the other one, indicates a location inside of the selected vertex. It is important to point out that the $\mu_i c_i$-adder performs a μ_i times addition of c_i. Finally, the output block $F(x)$ indicates the value of the fuction $F(\cdot)$ at the input x.

A Description

In Fig.2 is shown the circuit implementation for the scheme of Fig.1. The output of the IC is a digital word with 8-bit precision. In the present version of the IC, the memory was left outside. There are two alternatives to load the input values into the chip. The first alternative is by presenting three analog values at three input pins. There are three comparators which compare the input signals with an analog ramp and latch the conversion. The second alternative is to load directly the digital values serially. In both cases, the inputs are stored in 8-bit registers. The four most significant bits of the inputs are used to select the simplex the input belongs to and the four less significant bits indicate the input position inside the simplex. The weighting coefficients c_k are kept in the external memory, which is addressed with a 12-bit word.

The value of the PWL function $F(x)$ at each input x is the weighted sum of $n + 1 = 4$ parameter values. The four addresses to the memory positions where the coefficients c_j ($j \in \Im_z$) are stored are obtained by comparing the values of a digital ramp with the four less significant bits (LSB) of the 8-bit registers. This ramp is implemented with a 4-bit digital counter. Each 12-bit address is obtained by juxtaposing $n = 3$ 4 bit strings. The i-th string is equal to the four most significant bits (MSB) of the i-th register if the counter count is greater than the four LSB of the register; otherwise, the i-th string is the value of the four MSB of the register plus one. The comparison between the counter and each register is done using a digital comparator. Each address is calculated by a block called Address Generator, and the weighted sum is done with a 12-bit adder. The weighted sum $\sum_{j\in\Im_z} c_j\mu_j$ is obtained for free, since the memory position of c_j is addressed by the Address Generator for a time proportional to μ_j. Then, it is sufficient the 12-bit adder to perform the whole weighted sum.

B Architecture

The IC has an analog block and a digital block; both are powered up from different sources to allow them working and being tested separately as shown in Fig.2 The analog block consists of three A/D converters, based on an external ramp and an OTA comparator. The analog ramp must be synchronized with the internal counter. The comparator output is used to latch the value of the input so that the A/D conversion is performed at the same time in the three input channels. The comparator outputs are connected to output pads and the Latch signals are connected to input pads. Therefore, an external signal can be used to latch the values in the registers. As was mentioned before, this alternative was used to obtain the experimental results of the IC.

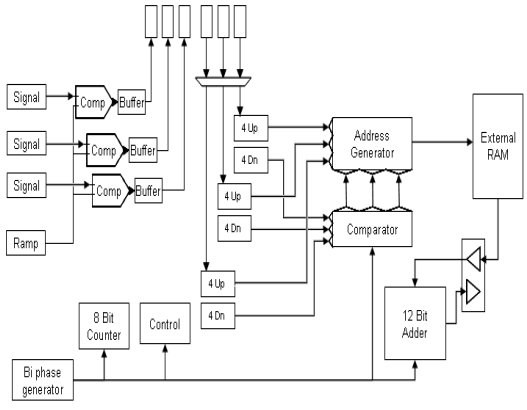

Figure 2: Chip Architecture.

As shown in Fig.2, the digital Block consist of a control block, a 8-bit counter, a 12-bit adder and three sets of 8-Bit Register, Address generator and Comparator. Ap-

Table I

State	EP	PROC
Nothing	1	0
Converting	0	0
Processing	0	1

propriately sized buffers were designed to drive the clock and clear lines.

B 1 Control Block

In order to perform the A/D conversion and the function evaluation, the chip has three different states called Nothing, Converting and Processing, which are coded with two registers in the Control block. In the Nothing state, the I/O bus works as an output bus and shows the value of the function calculated previously. When the input (say SP) is "1", the state machine (FSM) goes to the Converting state, to make the A/D conversion. The FSM stays in this state 256 clock cycles and after that, it goes into the Processing state. While the chip is making the A/D conversion, the signal to latch the value of the counter in the register is generated by the OTA. In the next state (Processing) it should be ensured that the signals to latch do not change, because the register would latch the new counter value. In order to avoid this, a multiplexer was placed in the input of the register which connects the output of the OTA in the Converting state, and sets a "1" in the latch signal in the Processing state. In the Processing state the I/O bus works as an input bus connected to the external RAM. In this state the chip performs the 16 additions reading the PWL parameter values from the external memory.

The two control signals EP (End of Processing) and PROC (Processing) provided by the FSM in each state are summarized in Table I.

B 2 12-Bit Adder

In order to produce the weighted sum, necessary to obtain the value of $F(x)$, the adder adds the sixteen values from the memory and divide it by sixteen. In order to add 16 values of 8 bits, a 12-bit adder is needed; the divide-by-16 operation is easily done by taking only the 8 most significant bits. The 12-bit adder has 8 inputs, so that the 4 most significant bits are connected to "0". The adder circuit is comprised of two modules, one calculates the carry, and another calculates the value of the sum.

B 3 8-Bit Counter

The 8Bit Counter is used for two different functions: To perform the A/D conversion and also to perform the addition of the 16 sums of the values of the memory parameters (ci). This block has a modular Structure and work with a two-phase clock.

B 4 8-Bit Register

Each register is Master-Slave with a two-phase clock, where the Master reads input data with a logic "1" in phase one and locks the data with a logic "0". The slave works in a similar fashion but with the second phase.

B 5 Comparator

It compares the less significant bits of each input register with the digital ramp. Since the ramp is implemented with the 8-bit Counter the 4 LSB of the counter are connected to the comparator and also the 4 LSB of the input register.

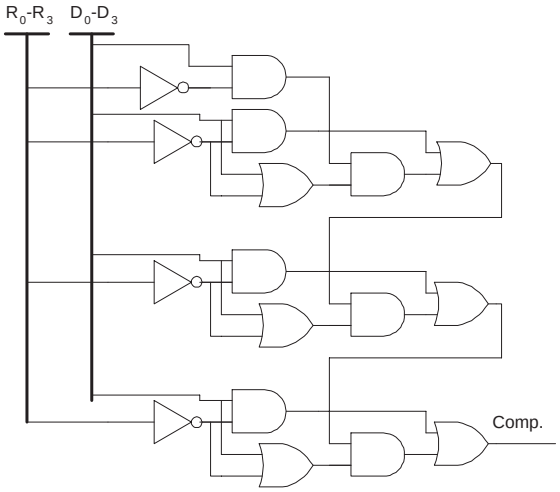

Figure 3: Comparator.

B 6 Address Generator

The Address generator determines the address memory where the (ci) parameter can be found. The inputs of this circuit are the 4 MSB of the input register and the comparator outputs. If a comparator output is 0,then the corresponding address generator output is directly the four more significant bits of that specific input register. If a comparator output is 1, then the corresponding address generator output is given by the consecutive address memory.

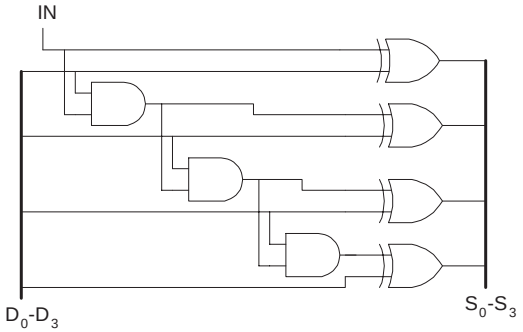

Figure 4: Address generator.

C Numerical example

In order to clarify the mathematical background exposed in section II and the IC performance explained in the

previous section, let us consider a hypothetical two-dimensional example. Suppose the continuous PWL funcion $F(x_1, x_2)$ defined over a simplicial partition with $m_1 = m_2 = 2$ and a unitary grid step ($\delta = 1$), as it is depicted in Fig.5.

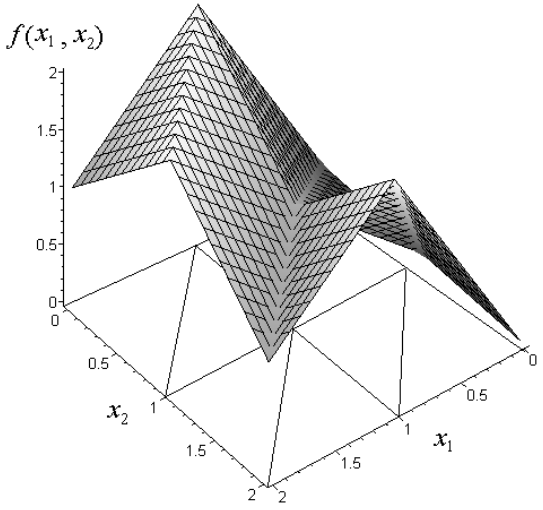

Figure 5: A two-dimensional PWL function.

The value of the PWL funcion, $c_i = F(x_1, x_2)$ at the vertex points, is collected and stored into the RAM memory as it is summarized in Table1.

i	Vertex	Memory Dir.	$\mathbf{c}_i = F(\mathbf{x_1}, \mathbf{x_2})$
0	$(0,0)$	00000000	0
1	$(0,1)$	00000001	2
2	$(0,2)$	00000010	1
3	$(1,0)$	00010000	0
4	$(2,0)$	00100000	0
5	$(1,1)$	00010001	1
6	$(1,2)$	00010010	2
7	$(2,1)$	00100001	2
8	$(2,2)$	00100010	1

Table 1: $c_i = F(\mathbf{x_1}, \mathbf{x_2})$ values.

The evaluation of an arbitrary point, for instance, the point $\mathbf{x} = (1.5, 1.75)$ at the function $F(x_1, x_2)$ is obtained as follows:
As a first step, the point $\mathbf{x}$ is decomposed by

$$\begin{bmatrix} 1.5 \\ 1.75 \end{bmatrix} = 0.5 \begin{bmatrix} 2 \\ 2 \end{bmatrix} + 0.25 \begin{bmatrix} 1 \\ 2 \end{bmatrix} + 0.25 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Now, let us introduce the following notation:

$$X_b = [B_{MSB}.B_{LSB}] = [B_3 B_2 B_1 B_0 . B_{-1} B_{-2} B_{-3} B_{-4}]$$

where X_b indicates a 8 bits number decomposed in two sections separated by a point. B_{MSB} and B_{LSB} indicate the integer and fraction part of the number, respectively and B_n is the 2^n-bit for $n \in \{3, 2, 1, 0, -1, -2, -3, -4\}$. The digital numerical code for x is given by

$$\begin{bmatrix} 0001.1000 \\ 0001.1100 \end{bmatrix} = .1000 \begin{bmatrix} 0010 \\ 0010 \end{bmatrix} + .0100 \begin{bmatrix} 0001 \\ 0010 \end{bmatrix} + .0100 \begin{bmatrix} 0001 \\ 0001 \end{bmatrix}$$

The B_{MSB} and B_{LSB} are in fact, the more and less significant bits of the 8-bits input register.
Notice that the decomposed representation of the point $\mathbf{x} = (1.5, 1.75)$ can also be rewritten as

$$\begin{bmatrix} 1.5 \\ 1.75 \end{bmatrix} = 0.5 \left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\} + 0.25 \left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right\} + 0.25 \left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right\}$$

where $[1\ \ 1]^T$ is a simplex selector term and it corresponds with the 4-bits more significant of the input register.

In the digital format it is given by

$$\begin{bmatrix} 0001.1000 \\ 0001.1100 \end{bmatrix} = .1000 \left\{ \begin{bmatrix} 0001 \\ 0001 \end{bmatrix} + \begin{bmatrix} 0001 \\ 0001 \end{bmatrix} \right\} + .0100 \left\{ \begin{bmatrix} 0001 \\ 0001 \end{bmatrix} + \begin{bmatrix} 0000 \\ 0001 \end{bmatrix} \right\} + .0100 \left\{ \begin{bmatrix} 0001 \\ 0001 \end{bmatrix} + \begin{bmatrix} 0000 \\ 0000 \end{bmatrix} \right\}$$

In accordance with reference [6], from a purely mathematical point of view, the μ_{i_l} parameters are computed as

$$\mu_{i_3} = 0.5$$
$$\mu_{i_2} = 0.25$$
$$\mu_{i_1} = 1 - (\mu_{i_2} + \mu_{i_3}) = 1 - (0.75) = 0.25$$

From a circuit point of view, the μ_{i_l} values indicate the times that a parameter must be added itself. Fig.6 shows the two comparator outputs for our example. Notice that μ_{i3} takes 8 ramp cycles and $\mu_{i2} = \mu_{i1}$, 4 cycles.

Finally, according with equation (8), $F(\mathbf{x})$ is computed by a weighted sum of the $\mu_{i_l} c_{i_l}$ product terms, where c_{i_l} indicates the value of $F(\mathbf{x})$ at the i_l-th vertex.
After substituting the value of $F(\mathbf{x})$ from Table1, at the vertices $[2\ \ 2]^T$, $[1\ \ 2]^T$, and $[1\ \ 1]^T$, it results

$$F(x_1, x_2) = 0.5(1) + 0.25(2) + 0.25(1) = 1.25$$

In a digital format the evaluation of $F(\cdot)$ is obtained directly form the 12-bit adder output. It performs a

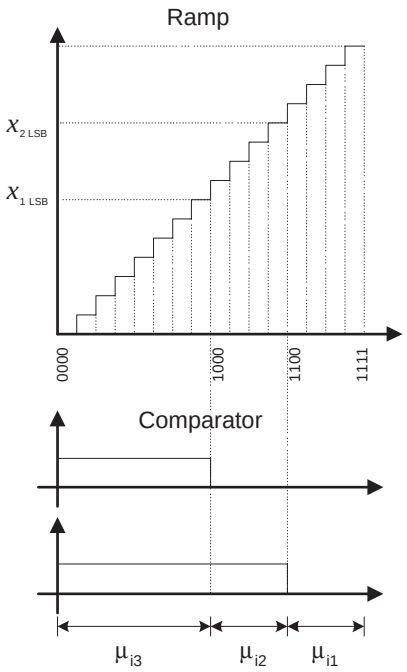

Figure 6: μ parameters.

μ_{il} times sum of the c_i values indicated by the address generator. The memory directions to obtain the c_i values are: $DIR[00100010]$, $DIR[00010010]$, and $DIR[00010001]$. For our example, the value of the PWL function is given by

$$\begin{aligned} F(x_1, x_2) = {} & 000000000001 + 000000000001 + \\ & 000000000001 + 000000000001 + \\ & 000000000001 + 000000000001 + \\ & 000000000001 + 000000000001 + \\ & 000000000010 + 000000000010 + \\ & 000000000010 + 000000000010 + \\ & 000000000001 + 000000000001 + \\ & 000000000001 + 000000000001 \\ = {} & [000000010100] \end{aligned}$$

As the adder result is scaled, then it must be divided by 16 in order to obtain the final result. Such result is obtained by considering the 8 more significant bits of the adder as the integer part of the final result and the 4 less significant as the fraction part.
The evaluation of $F(\cdot)$ at the input $(\mathbf{x_1}, \mathbf{x_2})$ in a digital format is given by

$$F(0001.1000, 0001.1100) = 00000001.0100$$

IV Layouts for the digital sections of the IC

In this section we present the layouts for the digital sections involved into the IC architecture. The layouts were designed aided by the Tanner EDA Tools software. The IC was integrated in a n-well non-silicided CMOS process of $0.5\mu m$. This process has 3 metal layers and 2 poly layers. All the transistors of the digital part are minimum size, being the PMOS of $3\mu m \times 0.6\mu m$ and the NMOS of $1.8\mu m \times 0.6\mu m$. Fig.7 show the comparator layout. The size of this block is $114\mu m \times 57\mu m$. The 1-bit-adder block of the 12-bit-adder is shown in Fig.9. The selector layout of a $30\mu m \times 114\mu m$ size is shown in Fig.8.

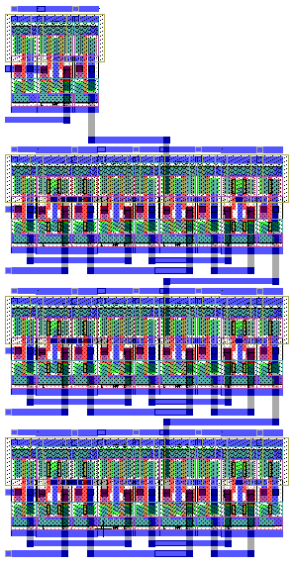

Figure 7: Comparator layout.

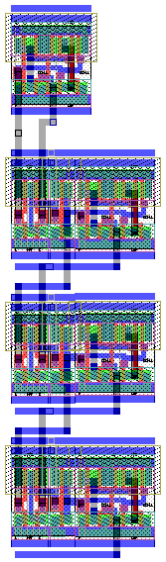

Figure 8: Selector layout.

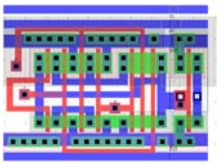

Figure 9: Adder layout.

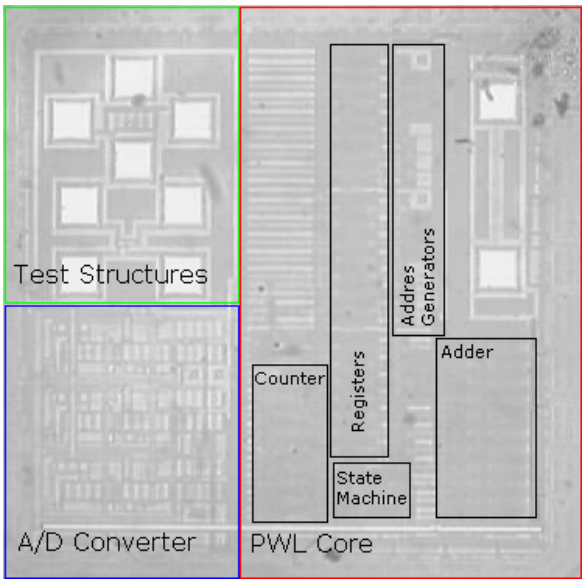

Figure 10: The IC. The main blocks of the circuit are evidenced.

V Conclusions

It has been shown the implementation of a simplicial PWL function evaluator. The proposed IC allows to evaluate with good accuracy a three dimensional PWL function. The block diagram and a detailed explanation of the chip operation is described. The mathematical background is presented and also a simple two-dimensional numerical example lets understanding the chip operation.

VI Aknowledgment

Ph.D. Víctor M. Jiménez Fernández is grateful for the partial economical support that he received by the National Institute for Astrophysics, Optics and Electronics in the Post.Ph.D. visitor position at the Universidad Nacional del Sur, Bahía Blanca, Argentina.
The authors would thank Poggi Tomaso for his help in the chip testing. Also, authors are grateful to “Fundación Universidad Nacional del Sur”’ for the support given by the PICT 2003 No.13468..

REFERENCES

[1] M. Storace and M. Parodi, “Towards analog implementations of PWL two-dimensional non-linear functions,” *International Journal of Circuit Theory and Applications*, vol. 33, no. 2, pp. 147-160, Mar.-Apr. 2005.

[2] M. Parodi, M. Storace, and P. Julián, “Synthesis of multiport resistors with piecewise-linear characteristics: a mixed-signal architecture,” *International Journal of Circuit Theory and Applications*, VOL.33, no. 4, pp. 307–319, Jul.-Aug. 2005.

[3] P. Julián, A. Desages, and B. D’Amico, “ Orthonormal High-Level Canonical PWL Functions with Applications to Model Reduction,” *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, VOL.47, pp. 702-712, May 2000.

[4] P. Julián and O. Agamennoni, “High-Level Canonical Piecewise Linear Representation Using a Simplicial Partition,” *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, VOL.46, pp. 463-480, April 1999.

[5] P. Julián, R. Dogaru, and L. Chua, “A Piecewise-Linear Simplicial Coupling Cell for CNN Gray-Level Image Processing,” *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, VOL.49, pp. 904-913, ¡July 2002.

[6] P. Mandolesi, P. Julián, and A. Andreou, “ A scalable and Programmable Simplicial CNN Digital Pixel Processor Architecture,” *IEEE Transactions on Circuits and Systems-I: Regular papers*, VOL.51, pp. 988-996, May 2004.

[7] M. Di Federico, P. Julián, T. Poggi, and M. Storace, “ A Simplicial PWL Integrated Circuit Realization, accepted in” *IEEE International Symposium on Circuits and Systems ISCAS-2007*, New Orleans, U.S.A., May 2007.