World Applied Sciences Journal 16 (Special Issue on Recent Trends in VLSI Design): 53-62, 2012
ISSN 1818-4952

VLSI Design for Activation and Membership Function for Neuro-Fuzzy Integrated System

[1]A.Q. Ansari and [2]Neeraj Kumar Gupta

[1]Department of Electrical Engg., Jamia Millia Islamia, New Delhi, 110025, India
[2]Department of Electrical & Electronics Engg.,
Krishna Institute of Engg. & Technology, Uttar Pradesh 201206, India

Abstract: In this paper, a Neuro-Fuzzy integrated system, which is based on fuzzy inference system using on-line learningability of neural network, is presented. By using on-line learning procedure, the proposed Neuro-fuzzy integrated system (NFIS) can be used to construct an input-output mapping based on fuzzy if-then rules and the tuning of the parameters of membership function. The activation and membership functions for NFIS have been realized using differential amplifier and operational transconductance amplifier (OTA) respectively. With a simple structure the sigmoidal activation and fuzzy membership function shows good performance in low power and area consumption.

Key words: Neuro-Fuzzy integrated system · Differential amplifier · Operational Transconductance Amplifier · Current Mirror OTA · Multiplier Type OTA

INTRODUCTION

Computational Intelligence combines neural network, fuzzy systems and evolutional computing [1-8]. Neurofuzzy integrated system utilizes features of both Neural and Fuzzy networks together for better results by which we can easily generalize the unseen data from seen data by forming the fuzzy rules and training. Neural networks are composed of a large number of highly interconnected processing elements (nodes), which are connected through the weights. When looking into the structure and parameter learning of neural networks, many common points to the methods used in adaptive processing can be found. The backpropagation algorithm used to train the neural network is a generalized Widrow's least mean square (LMS) algorithm and can be contrasted to the LMS algorithm usually used in adaptive system.

In this paper, neuro-fuzzy integrated system and its analog VLSI circuits for fuzzy membership and neural activationfunctions have been presented. Neuro-Fuzzy integrated system learns system behavior by using system input-output data and so does not use any mathematical modeling.After learning the system's behavior, neuro-fuzzy integrated system automatically generates fuzzy rules and membership functions and thus solves the key problem of fuzzy logic and reduces the design cycle very significantly.

Neuro-Fuzzy integrated system then verifies the solution (generated rules and membership functions). It also optimizes the number of rules and membership functions. Finally, automatic code converter converts the optimized solution (rules and membership functions) into embedded controller's assembly code.

The first fuzzy chip was reported in 1986 at AT&T Bell Laboratories. Since then many different approaches have been suggested [9, 10]. Depending on the design techniques employed they are classified into two groups: digital and analog. Generally a digital fuzzy system is either a fuzzy (co-) processor [11, 12] or a digital ASIC [13], which contains logic circuits to compute the fuzzy algorithm memories to store fuzzy rules and generators or look-up tables for membership functions of the input and output variables. Compared to its analog counterpart, the digital approach has greater flexibility, easier design automation and good compatibility with other digital systems. However, most of the digital systems require A/D and D/A converters to communicate with sensors and/or actuators. Furthermore, the digital systems are

Corressponding Author: A.Q. Ansari, Department of Electrical Engg., Jamia Millia Islamia, New Delhi, 110025, India.
E-mail: aqansari@ieee.org.

more complex and need larger chip area, e.g. the synthesis of a 4-bit maximum operation in [14] results in a CMOS unit of nearly 100 transistors. The research on analog fuzzy systems started with the pioneering work of Yamakawa [15] and was followed by many researchers [9, 10, 16]. With the nonlinear characteristics of the active devices in analog circuit, the fuzzy elements can be implemented in very simple structures. This brings a reduction in the circuit complexity, which implies better speed performance and reduced chip-area consumption. Until now, the main drawback of the analog approaches has been their poor flexibility [17, 18]. To overcome the shortcomings encountered in analog circuit, while still keeping their advantages, analog VLSI circuits for fuzzy membership and neural activation functions have been simulated.

In order to facilitate an easy and systematic understanding of the proposed work, section II discusses neuro-fuzzy integrated system; section III includes analog VLSI circuit for activation functions using differential amplifier;section IV includes analog VLSI circuit for membership functions using operational transconductance amplifier (OTA) and at the end section V presents the conclusions and scope for future research.

Structure of Neuro-fuzzy Integrated System: Fig. 1 shows the function diagram of the proposed neuro-fuzzy integrated system (NFIS). Theneuro-fuzzy integrated systempossessesthe advantages of both the neural and fuzzy systems. It bringsthe low-level learningand computational power of neural networks into fuzzy systems and provides the high-level human-like thinking and reasoning of fuzzy systems into neural networks.

Both structure and parameter learning are used concurrently for the construction of an adaptive neuro-fuzzy integrated system. The structure learning includes the pre-condition and consequent structure identification. The pre-condition structure corresponds to the input space partitioning and is formulated as a combinational optimization problem with objective to minimize the number of rules generated and number of fuzzy sets. The main task of consequent structure identification is to decide when to generate a new membership function and which significant terms (input variables) should be added to the consequent parts. Least mean squares (LMS) or recursive mean squares (RLS) algorithms adjust the parameters of the consequent parts and the pre-condition

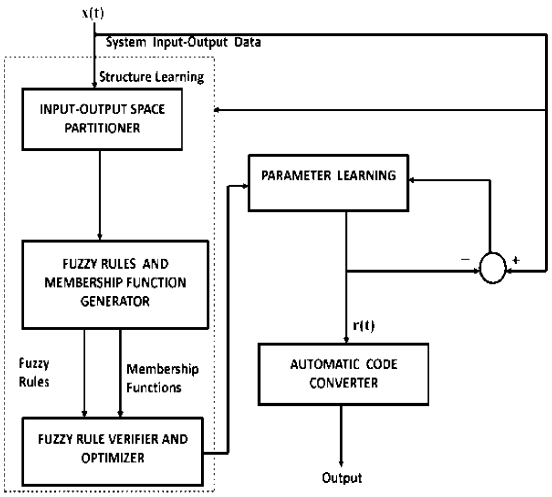

Fig. 1: Functional diagram of the proposed NFIS

part parameters are adjusted by using backpropagation algorithm to minimize the cost function. During the learning process the system can be used anytime for normal operation without repeated training on the input output patterns. Rules are created dynamically on receiving on-line training data by performing following processes:

- Input / Output space partitioning;
- Fuzzy rules and membership function generator;
- Fuzzy rules verifier and optimizer;
- Parameter learning

In this learning process the first three steps belong to structure learning and the last step belong to the parameter learning. The various blocks of an adaptive neuro-fuzzy integrated system can be explained as follows:

Input / Output Space Partitioning: This block determines the number of rules extracted from the training data as well as the number of fuzzy sets on the universe of discourse disclosure of each input variable. A rule corresponds to a cluster in the input space with m_i and p_i representing the center and the variance of the cluster. For the incoming pattern x the strength a rule is fired can be interpreted as the degree of incoming pattern belonging to the corresponding cluster. For computational efficiency, we can use the firing strength as follows:

$$F^i(x) = \prod_i u_i = e^{-[p_i(x-m_i)]^T[p_i(x-m_i)]} \quad (1)$$

Where $F^j \in [0, 1]$ and the argument of the exponential is the distance between x and the cluster i. The above criteria can be used for the generation of new fuzzy rule. Let $x(t)$ be the newly coming pattern then

$$J = \arg \max_{1 \le j \le r(t)} F^j(x) \tag{2}$$

Where $r(t)$ is the number of existing rules at time t. If $F^j \le \bar{F}(t)$, then a new rule is generated where $\bar{F}(t) \in (0,1)$.

Once a new rule is generated the next step is to assign initial centers and widths of the corresponding membership functions, which can be set as follows

$$m_{[r(t)+1]} = x \tag{3}$$

$$p_{[r(t)+1]} = -\frac{1}{\beta} diag\left[\frac{1}{\ln(F^1)} \ldots\ldots \frac{1}{\ln(F^J)}\right] \tag{4}$$

In the system the width is taken into account in degree measure so that a cluster with larger width, fewer rules will be generated in its vicinity than a cluster with smaller width.

Fuzzy Rules and Membership Function Generator: The generation of a new input cluster corresponds to the generation of a new fuzzy rule, with its pre-condition part constructed by the learning algorithm. At the same time, the consequent part of the generated rule is decided. The algorithm is based on the fact that differen tpre-conditions of different rules may be mapped to the same consequent fuzzy set. Since only the center of each output membership function isused for defuzzification, the consequent part of each rule may simply be regarded as a singleton.

Fuzzy Rule Verifier and Optimizer: The generated fuzzy rules and membership functions can be verified by using neuro-fuzzy integrated rule verifier. Both training set as well as a test set should be used for the verification process. If the generated rules and membership functions do not produce satisfactory result, one can easily manipulate appropriate parameters (e.g. more data, smaller error criterion, learning rate etc) so that the neural net learns more about the system behavior and finally produce satisfactory solution. The number of rules and membership function can also be optimized using the neuro-fuzzy integrated rule verifier, which is another very important feature. This reduces memory requirement and execution speeds-the two very desirable features for many applications. The optimization process might lose some accuracy and one can make some trade-offs between accuracy and cost.

Parameter Learning: After the network structure is adjusted according to the current training pattern, the network then enters the parameter learning phase to adjust the parameters of the network optimally based on the same training pattern. Parameter learning is performed on the whole network after structure learning, irrespective of whether the nodes (links) are newly added or are existent originally. Using backpropagation for supervised learning the error function as follow.

$$D = \frac{1}{2}[O(t) - T(t)]^2 \tag{5}$$

Where $T(t)$ is the desired output and $O(t)$ is the current output.

For each training data set, starting at the input nodes, a forward pass is used to compute the activity levels of all the nodes in the network to obtain the current output $O(t)$. Then starting at the output nodes, a backward pass is used to compute $\partial D / \partial w$ for all the hidden nodes. Noting that w is the adjustable parameter in a node, the update rule used is

$$\Delta w \propto -\frac{\partial D}{\partial w} \tag{6}$$

$$w(t+1) = w(t) + \eta\left(-\frac{\partial D}{\partial w}\right) \tag{7}$$

Where η is the learning rate, $\frac{\partial D}{\partial w} = \frac{\partial D}{\partial a} \cdot \frac{\partial a}{\partial w}$ and a is the activation function.

Learning with Fuzzy Input Vectorsfor Classification Problems: In order to classify n-dimensional fuzzy vectors, a multilayer feed forward neural network that has ninput units, n' hidden units and a single output unit have been employed. When the fuzzy input vector $A_p = (A_{pi_1} :\ldots\ldots, A_{p_n})$ is presented to the input layer, the fuzzy input-output relation of each unit of the neural network is defined as follows:

Input units:

$$Y_{pi} = A_{pi}, \quad i = 1,2,\ldots\ldots,n \tag{8}$$

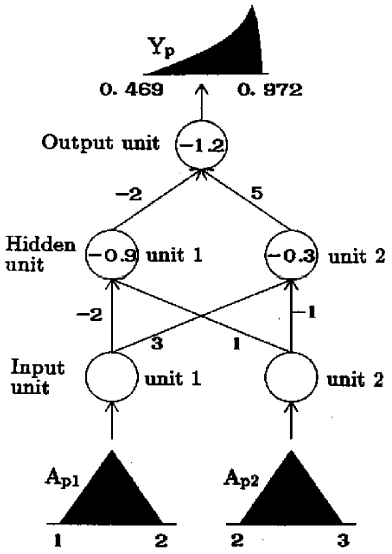

Fig. 2: An example of fuzzy input-output relations of neural networks.

Hidden units:

$$Y_{pj} = f(Net_{pj}), \qquad J = 1,2,.....n' \tag{9}$$

$$Net_{pj} = \sum_{i-1}^{n} \omega_{ji} Y_{pi} + \theta_j \tag{10}$$

Output units: $$Y_p = f(Net_p) \tag{11}$$

$$Net_p = \sum_{j=1}^{n'} \omega_j Y_{pj} + \theta \tag{12}$$

Here the weights ω_{ji}, ω_j and the biases θ_j, θ are real parameters which are updated in the learning phase of the neural network. The inputs A_{pi}, Net_{pi} and Net_p and the outputs Y_{pi}, Y_{pj} and Y_p are fuzzy numbers.

The fuzzy input vector $A_p = (A_{p1},.....A_{pn})$ is mapped to the fuzzy output Y_p, by the neural network defined by (8)-(12). In Fig. 2, a simple example of the neural network where weights and biases are fixed as follows:

$$\omega_{ji} : \omega_{11} = -2, \ \omega_{21} = 3, \ \omega_{22} = -1, w_{12} = 1$$

$$\omega_{ji} : \omega_{11} = -2, \ \omega_2 = 5$$

$$\theta_j : \theta_1 = -0.9, \ \theta_2 = -0.3$$

$$\theta : \theta = -1.2$$

In Fig. 2, the weights and the biases are shown beside the connections and inside the units, respectively. The fuzzy input vector A, = (A_{pl}, A_{pz}) in Fig. 2 is as

$$A_{p1}=(1.5, 0.5)_L, A_{p2} = (2.5, 0.5)_L,$$

Where $A = (a, b)_L$ denotes a symmetric triangular fuzzy number with the center *a*and the spread *b* defined by the membership function:

$$\mu_A(x) = \max\{1 - |x - a|/b, 0\} \tag{13}$$

The fuzzy input vector A_p, = ((1.5, 0.5)$_L$, (2.5, 0.5)$_L$ is mapped to the fuzzy number Y_p, as shown in Fig.2 by the fuzzy input-output relations in (8)-(12). It should be noted that the calculation of the actual output Y_p, is performed on level sets while the input-output relation of the neural network is defined by the extension principle in (8)-(12). Since the level sets of fuzzy numbers are intervals, the calculation of $Y_p$requires only interval arithmetic. In fact, Y_p, in Fig. 2 is drawn by interval arithmetic on 100 level sets corresponding to h = 0.01,0.02,..., 1.00.

The characteristic features of the architecture of neural networks are as follows:

- Fuzzy numbers are propagated through neural networks;
- A single unit is employed for representing a fuzzy number.

These two features stand in contrast to other approaches proposed for dealing with fuzzy sets as input values to neural networks [19-26]. For example, in the studies by Yager [20], Umano and Ezawa [21] and Keller *et al.*[22-24], fuzzy sets on discrete base sets were used as input values to neural networks. For a single fuzzy set, A, on a discrete number base set with nelements, say Ω = (x_i,x_2,.....x_n), n input units were employed for representing A. The membership value of the fuzzy set on x_i, i.e., $\mu A(x_i)$, was the input to the i^{th} input unit. In their architecture, a fuzzy set was represented by its membership values and real numbers were propagated through neural networks. In the study by Uehara and Fujise [25], a fuzzy set is represented by a set of level sets. That is, a fuzzy set A, was represented by its level sets: $[A]_{h1}$,$[A]_{h2}$,.....$[A]_{hm}$ and m was the number of level sets. The input values of neural networks were the upper limits and the lower limits of the m level sets. Therefore 2m input units were required for representing a single fuzzy set. Since the input values were real numbers, real numbers were propagated through the neural networks of Uehara and Fujise [25].

Learning Algorithm of Neural Networks: First we define the target output, t_p, corresponding to the fuzzy input vector $A_p = (A_{pi},.....A_{pn})$ as

$$t_p = \begin{cases} 1 & \text{if } A_p \text{ belongs to class 1} \\ 0 & \text{if } A_p \text{ belong to calss2} \end{cases} \quad (14)$$

Let define a cost function to be minimized in the learning of neural networks using the fuzzy actual output, Yp and the corresponding target output, tp. For the h-level set of Yp, a cost function is defined by the maximum squared error as

$$e_{ph} = \max\left\{(t_p - o_p)^2/2 : o_p \in \left[Y_p\right]_h\right\} \quad (15)$$

If the input vector Ap is a real vector, Yp is a real number. In this case, the cost function is reduced to the squared error used in the BP algorithm.

The h-level set of the fuzzy input vector $A_p = (A_{pi},.....A_{pn})$ is mapped to the h-level set of the fuzzy output Yp. The h-level set of Ap is defined as

$$\left[A_p\right]_h = \left(\left[A_{p1}\right]_h,.....\left[A_{pn}\right] \right)_h \quad (16)$$

Therefore the h-level set $[Y_p]_h$in is calculated from the interval input vector $\left[A_p\right]\left(\left[A_{p1}\right]_h,.....\left[A_{pn}\right] \right)_h$. This means that the cost function [28] can be calculated only from the operations on intervals. That is, the calculation of the exact shape of Ypdefined by the extension principle is not required in the learning of neural networks.

The learning of neural networks involves minimizing the cost function [28]. In a manner similar to that outlined by Rumelhart *et al*. [29, 30], the weights w_j and $w_{ji}$are changed according to the following rules:

$$\Delta\omega_j(t+1) = \eta(-\partial e_{ph}/\partial\omega_j) + \alpha\Delta\omega_j(t) \quad (17)$$

$$\Delta\omega_{ji}(t+1) = \eta(-\partial e_{ph}/\partial\omega_{ji}) + \alpha\Delta\omega_{ji}(t) \quad (18)$$

Where t indexes the presentation number and η and α are a learning constant and a momentum constant, respectively. The biases θ and θ_j, are changed in the same manner as w_j and $W_{ji.}$.

Using the cost function (15), a neural network for a fixed value of h has been trained. If the sum of the cost functions over all the given fuzzy vectors becomes very small by the learning, the h-level sets of the given fuzzy vectors can be correctly classified using the trained neural network.

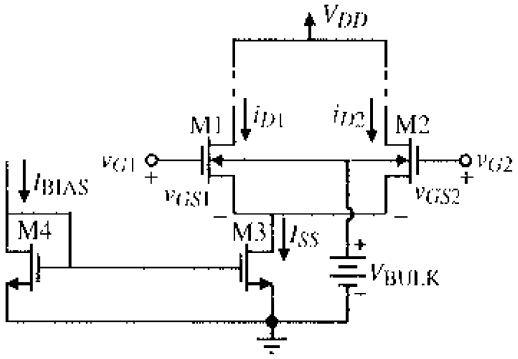

Fig. 3: CMOS differential amplifier using NMOS transistors

Automatic Code Converter: After a satisfactory solution is obtained, automatic code converter converts the solution to an embedded processors assembly code. Various options can be provided to have the code optimized for accuracy, speed and/or memory. The definition and evaluation of instruction set extension for neuro-fuzzy integrated processing have shown [31, 32].

Analog Vlsi Circuit for Activation Function Using Differential Amplifier

Large-Signal Analysis of CMOS Differential Amplifier Using NMOS Transistor: Analysis of the differential amplifier with the large-signal characteristics as in Figure 3 showna CMOS differential amplifier that uses *n*-channel MOSFETs M_1 and M_2 to form a differential amplifier. M_1 and M_2 are biased with a current sink I_{ss} connected to sources of M_1 and M_2. This configuration of M_1 and M_2 is often called a source-coupled pair: $M_3$and M_4 are an example of how the current sink I_{ss}might be implemented.

The large-signal analysis begins by assuming that M_1 and M_2 are perfectly matched. It is also not necessary for us to define the loads of M_1 and M_2 to understand the difference large-signal behavior. The large-signal characteristics can be developed by assuming that M_1 and M_2 of Fig. 3 are always in saturation. This condition is reasonable in most cases and illustrates the behavior even when this assumption is not valid. The pertinent relationship describing large-signal behavior are given as:

$$v_{ID} = v_{GSI} - vGS_2 = \left(\frac{2i_{D1}}{\beta}\right)^{1/2} - \left(\frac{2i_{D2}}{\beta}\right)^{1/2} \quad (19)$$

and

$$I_{SS} = i_{D1} + i_{D2} \tag{20}$$

Substituting (20) into (19) and forming a quadratic allows the solution for i_{D1} and i_{D2} as:

$$i_{D1} = \frac{I_{ss}}{2} + \frac{I_{ss}}{2}\left(\frac{\beta v_{ID}^2}{I_{ss}} - \frac{\beta^2 v_{ID}^4}{4I_{ss}^2}\right)^{1/2} \tag{21}$$

$$i_{D2} = \frac{I_{ss}}{2} - \frac{I_{ss}}{2}\left(\frac{\beta v_{ID}^2}{I_{ss}} - \frac{\beta^2 v_{ID}^4}{4I_{ss}^2}\right)^{1/2} \tag{22}$$

Where these relationship are only useful for $v_{ID} < 2(I_{SS} / \beta)^{1/2}$.

Bipolar Sigmoidal Function: The desired range here is between +1 and –1. This function is related to the hyperbolic tangent function. The bipolar sigmoidal function is given as,

$$b(x) = 2f(x) - 1 \tag{23}$$

$$b(x) = 2 \times \frac{1}{1 + exp(-sx)} - 1 \tag{24}$$

$$b(x) = \frac{1 - exp(-sx)}{1 + exp(-sx)} \tag{25}$$

On differentiating the function b(x),

$$b'(x) = \frac{s}{2}[(1 + b(x))(1 - b(x))] \tag{26}$$

Analog Activation Function Generator Circuit: Fig 5 is built of a coupled differential amplifier which has two differential pairs (DP1 and DP2) and a PMOS M5. The two-reference voltages, low-reference V_{rl} and high-reference V_{r2}, where $V_{rl} < V_{r2}$, define activation function. Due to the coupling effect of the PMOS, the subfunction generation and the final output combination are completed within one stage.

Depending on the relative values between the input voltage V_i and the reference voltages V_{rl} and V_{r2}, the circuit operates in one of the three regions (I-V) shown in Fig. 4. Assume that the current sources I_{ss} in DP1 and DP2 are ideal and equivalent, the input devices in each differential pair are symmetric, the half widths of the transfer regions of DP1 and DP2 are T_1 and $T_{2(T_i=\sqrt{2I_{ss}/\beta_i})}$

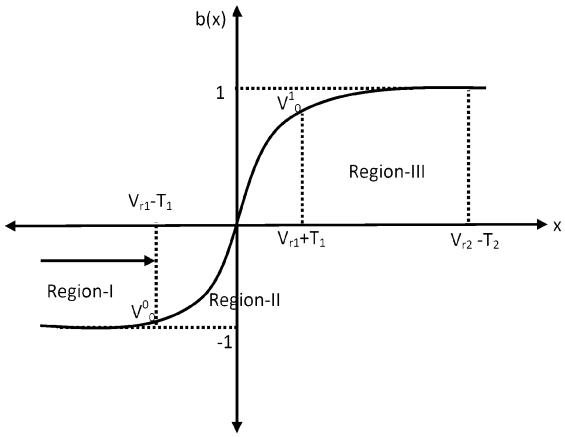

Fig. 4: Bipolar Sigmoidal Function

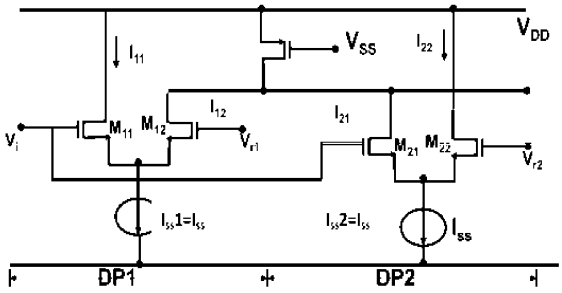

Fig. 5: Shows the schematic diagram of the proposed activation function circuit.

and β_i is the transconductance parameter of the input devices in DPl and DP2 with i = 1 and i = 2, respectively, then the three operating regions of the activation function circuit illustrated in Fig. 4 can be described as follows.

Region I: When $V_i < V_{rl} - T_1$, M_1 and M_3 are in their cutoff regions while M_2 and M_4 are in the saturation states. The current through the PMOS is equal to I_{ss}. The output voltage is $V_{out} = V_{DD} - I_{ss}R = V^0_o$ which corresponds to a activation function value -1.

Region II: When $V_{rl} - T_1 \le V_i \le V_{rl} + T_1$, DP1 is in its transfer region. If $V_{rl} - V_{rl} > T_1 + T_2$, then DP2 keeps the same state as in region I. As V_i, increases, the activation function circuit gives a linear ascending answer between -1 and 1.

Region III: When $V_{rl} + T_1 < V_i < V_{r2} - T_2$, both M_1 and M_4 are in cutoff states, the current flowing through the PMOS is zero. The output voltage is $V_{out} = V_{DD} = V_o^1$, which corresponds to activation function value 1.

Normalized with $(V_{out} - V_o^0)/(V_o^1 + V_o^0)$ the generated activation function can be summarized as

$$b(x) = \begin{cases} 0, \quad Vr_1 - T_1 & \text{region I} \\ \frac{1}{2}\left[1 + \left(\frac{\beta_1(V_i - V_{r1})}{I_{SS}} - \frac{\beta_1^2(V_i - V_{r1})^4}{4I_{SS}^2}\right)^{1/2}\right] & \\ V_{r1} - T_1 \le V_i \le V_{r1} + T_1 & \text{region II} \\ 1, \quad V_{r1} + T_1 \le V_i \le V_{r1} - T_1 & \text{region III} \end{cases} \quad (27)$$

As shown in Fig. 4 and (27), the definition of the implemented activation function is dependent on the reference voltages V_{r1}, V_{r2}, transconductance parameters β_1 and current I_{ss}.

Spice and Matlab simulation were perform to confirm the performance of the proposed circuit equations. Some partially program and glaring results of differential amplifier based circuits generating a bipolar sigmoidal function are presented.

```
diffampr=newff([-1 1],[10 1],
{'tansig' 'purelin'},'traingdm');
diffampr.trainParam.show=50;
diffampr.trainParam.lr=.05;
diffampr.trainParam.mc=.9;
diffampr.trainParam.epochs=500;
diffampr.trainParam.goal=1e-2;
Y=sim(diffampr,a);
plot(a,b,a,Y,'o')
```

- OPERATING POINT INFORMATION:

NAME	M11	M12	M21	M22
MODEL	MOSN	MOSN	MOSN	MOSN
ID	1.00E-04	7.84E-13	5.00E-05	5.00E-05
VGS	7.74E-01	-9.23E+00	6.01E-01	6.01E-01
VDS	7.74E-01	7.74E-01	6.01E-01	6.01E-01
VBS	0.00E+00	0.00E+00	0.00E+00	0.00E+00
VTH	1.80E-01	1.80E-01	1.80E-01	1.80E-01
VDSAT	5.94E-01	0.00E+00	4.21E-01	4.21E-01

- TOTAL POWER DISSIPATION 2.10E-13 WATTS
- CHIP SIZE 0.27 mm × 0.23 mm (0.0621 mm^2)

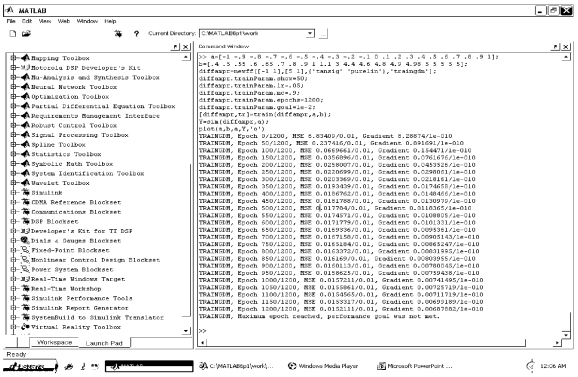

Fig. 6: CMOS Differential Amplifier Using a Current-mirror Load

Fig. 7: *Characteristics of Differential Amplifier*

Simulation Results

Analog Vlsi Circuits for Fuzzy Membership Functions: In our proposed circuit as shown in Fig 11, we use two types of OTA viz current mirror OTA (CMOTA) and multiplier-type OTA (MTOTA). Operational transconductance amplifiers (OTAs) have important building blocks for various analog circuits and systems. Depending on system needs, an OTA must satisfy many design requirements. Fig. 8 shows the basic two stage OTA which is often adopted to attain both the desired gain and output swing. For a two stage amplifier, the first stage is designed to the total gain and second stage contributes a large output swing. To evaluate the power consumption of Fig. 8, we consider a desired gain-bandwidth (GB) and load capacitance (C_L) . Assuming all transistors have same overdrive voltage (V_{OD} =V_{GS}-V_T) and the nondominanat pole at the output is at least three times of the GB, it can be shown that the core current consumption of the two-stage OTA[33] is given by

$$I_{OTA} = 2\pi \times GB \times V_{OV} \times (4C_C + 3C_L) \quad (28)$$

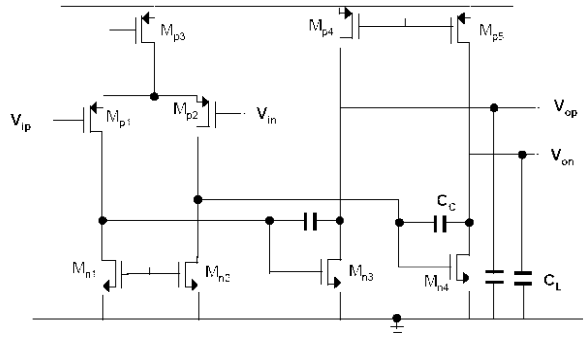

Fig. 8: Two stage OTA

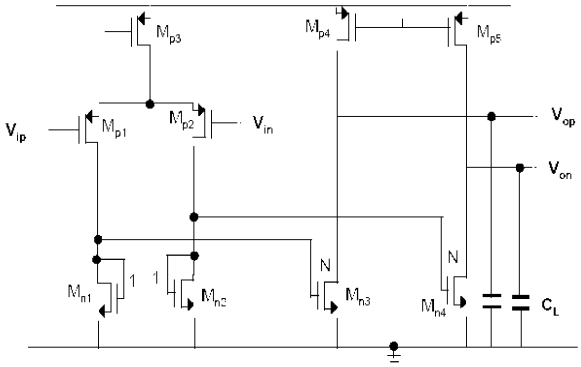

Fig. 9: Current Mirror OTA

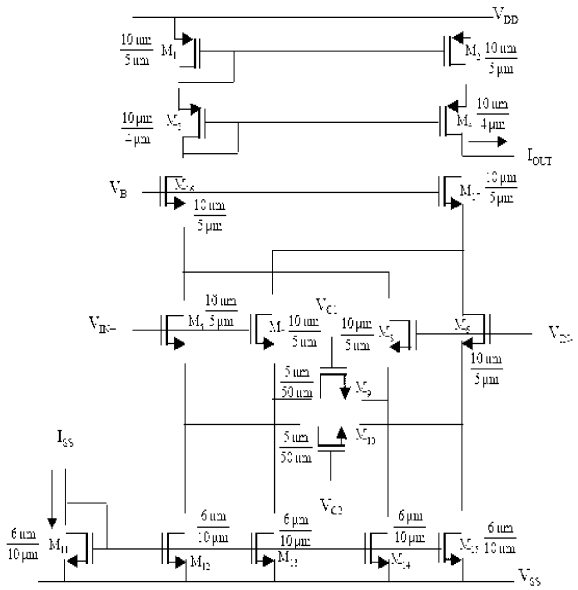

Fig. 10: Multiplier-type OTA

A significant portion of the current is spent on driving the compensation capacitors as seen in (28). A current mirror OTA (CMOTA) is used to avoid the compensation capacitors which is shown in Fig 9. The current consumed by a CMOTA can be expressed as [33, 34].

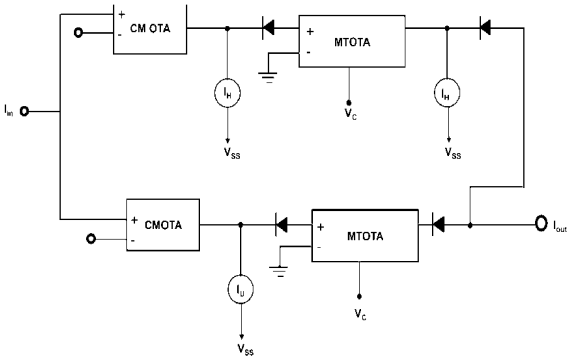

Fig. 11: Circuit for generating a triangle function with independent adjustable slopes

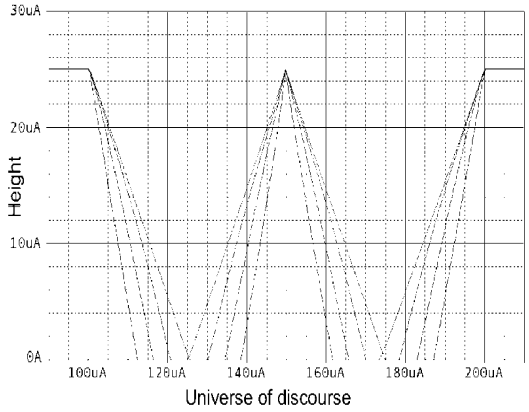

Fig. 12: Triangular, left and right shoulder membership functions

$$I_{CMOTA} = 2\pi \times GB \times V_{OV} \times C_L \times \left(1 + \frac{1}{N}\right) \qquad (29)$$

Where N is the size ratio between mirroring devices, Mn_1 and Mn_3. Comparing with (29), considerable power reduction is achieved by eliminating the compensation capacitors.

Fig. 10 shows a voltage controlled OTA resistor using a multiplier-type OTA. MOSFETs, M9 and M10, are operating in the triode region. Under these conditions, the output current I_{OUT} of the multiplier-type OTA becomes.

$$I_{OUT} = 4KV_C(V_{IN+} - V_{IN-}) \equiv G_{m2}(V_{IN+} - V_{IN-}) \qquad (30)$$

WhereK is the transconductance parameter of the MOSFET's, M9 and M10. From (30), the transconductance Gm2 of this OTA is proportional to the control voltage V_C.

The OTA based proposed circuit to generate membership function is shown in Fig. 11.

Table 1: Node analysis and features of proposed circuit

Node	Voltage	Node	Voltage	Node	Voltage	Node	Voltage
(1)	.6000	(2)	.1825	(3)	5.0000	(4)	-5.0000
(5)	-.0597	(6)	3.1531	(7)	2.9758	(8)	.6500
(9)	-.1825	(10)	0.0000	(X1.2)	4.7686	(X1.3)	4.7510
(X1.4)	4.5371	(X1.6)	2.0000	(X1.7)	1.8733	(X1.9)	-.1136
(X2.2)	4.6921	(X2.3)	4.6847	(X2.4)	4.3842	(X2.6)	2.0000
(X2.7)	1.8405	(X2.9)	.1967	(X3.2)	3.4503	(X3.3)	2.0492
(X3.4)	3.6965	(X3.6)	1.5000	(X3.7)	.4281	(X3.8)	.4170
(X1.10)	-.0614	(X1.12)	-4.6511	(X2.10)	1.9500	(X2.12)	-4.6511
(X3.10)	-.3454	(X3.11)	1.0000	(X3.12)	1.6500	(X3.13)	.2371
(X3.15)	-3.6999	(X3.16)	-.3341	(X3.17)	.2280		

• TOTAL POWER DISSIPATION 2.91E-03 WATTS

• CHIP SIZE 0.78 mm × 0.83mm(0.6475 mm^2)

SPICE simulations were performed to confirm the performance of the proposed circuit. Some glaring result of OTA based circuits generating a triangle function with independent adjustable slopes after being annotated with subcircuit using SPICE are presented in the Table 1.

To verify that the current mode cells work properly, electrical simulations were done in Spice using parameters of the standard AMI, 1·2-micron, n-well, MOS process. The DC characteristic appears in Fig. 12.

Voltages V_C responsible for changing slopes independently, were swept from 0 to 0·8V, keeping the current I_H at 25μA. The left and right shoulders were also simulated in DC, sweeping V_C from 0 to 0·8V. The universe of discourse was 300μA and the power supply voltages were V_{DD}=5V and V_{SS}=0V.

CONCLUSION

In this paper, a neuro-fuzzy integrated system is presented. The NFIS can be trained by numerical data and linguistic information expressed by fuzzy if-then rules. This feature makes the incorporation of *a* priori knowledge into the design of systems possible. Another important feature of the NFIS is that, without any giveninitial structure, the NFIS can construct itself automatically from numerical training data.Neuro-Fuzzy integrated system has been applied in a wide range of fields for imaging, classification, diagnosis, prediction and control in real life ambiguous situations. What is required for setting up such a system is data that represents the past history and performance of the real system and a selection of a suitable model.

Analog VLSI circuit for generating fuzzy membership and activation functions of neuro-fuzzy integrated system (NFIS) has also been presented. Due to compact size and low power consumption of the circuit, they are suitable for VLSI chip implementation. The field of neuro-fuzzy integrated hardware implementations is undoubtedly very vast and completely open for research at this moment. Our contribution is merely a step forward and an effort to explore such important technological field with viable implementation technique.

ACKNOWLEDGMENTS

We would like to thank NirmeshAwasthiof ST Microelectronics, G.Noidafor useful discussions on the related theme.

REFERENCES

1. Albus, J.S., 19975. A new approach to manipulator control: The cerebellar model articulation controller (CMAC), American Society of Mechanical Engineers Trans. Journal of Dynamic Systems, Measurement and Control, 97: 220-227.
2. Amari, S. and N. Kasabov, 1998. Brain-Like Computing and Intelligent Information Systems, New York: Springer-Verlag.
3. Arbib, M., 1995. The Handbook of Brain Theory and Neural Networks, Cambridge, MA: MIT Press.
4. Fukuda, T., Y. Komata and T. Arakawa, 1997. Recurrent neural networks with self-adaptive GAs for biped locomotion robot", in Proc. Int. Conf. Neural Networks ICNN. Piscataway, NJ: IEEE pp: 1710-1715.
5. Furlanello, C., D. Giuliani and E. Trentin, 1995. Connectionist speaker normalization with generalized resource allocation network, in Advances in neural information processing systems (NIPS), D. Toretzky *et al.*, Eds.Cambridge, MA: MIT Press, pp: 1704-1707.
6. D³ugosz, R., V. Kolodyazhniy and W. Pedrycz, 2010. Power efficient hardware implementation of a fuzzy neural network, Proceedings of the 17th International Conference on Mixed Design of Integrated Circuits and Systems, MIXDES, pp: 576-580.
7. Peymanfar, A., A. Khoei and Kh. Hadidi, 2007. A New ANFIS Based Learning Algorithm for CMOS Neuro-Fuzzy Controllers Electronics, 14th IEEE International Conference on Circuits and Systems, ICECS, pp: 890-893.
8. HouZhi-xiang and Li He-qing, 2006. Nonlinear System Identification Based on Adaptive Neural Fuzzy Inference System, Proceedings of International Conference on Communications, Circuits and Systems, 3: 2067-2069.

9. Kettner, T., K. Schumacher and K. Goser, 1993. Realization of a monolithic analog fuzzy logic controller, in Proc. 20th Eur. Solid-State Circuit Conf, Sevilla, Spain, Sept. pp: 66-69.
10. Miki, T., H. Matsumoto, K. Ohto and T. Yamakawa, 1993. Silicon implementation for a novel high-speed fuzzy inference engine: Mega-FLIPS analog fuzzy processor, J. Intell. Fuzzy Syst., 1(1): 27-42.
11. Sasaki, M., F. Ueno and T. Inoue, 1993. 7.5 MEIPS fuzzy microprocessor using SIMD and logic-memory structure, in Proc. 2nd Int. ConfFuzzySyst., San Francisco, CA, Mar. 1: 527-537.
12. Ungering, A. and K. Goser, 1994. Architecture of a 64-bit fuzzy inference processor, in Proc. 3rd IEEE Int. Conf Fuzzy Logic Syst., Orlando, FL, pp: 1776-1780.
13. Eichfeld M. Lohner and M. Muller, 1992. Architecture of a fuzzy logic controller with optimized memory organization and operator design, in Proc. the 1st IEEE lnt. Conf Fuzzy Syst., San Diego, CA, pp: 1317-1323.
14. Watanabe, H., W.D. Dettloff and K.E. Yount, 1990. A VLSI fuzzy logic controller with reconfigurable, cascadable architecture, IEEE J. Solid-Stare Circuits, 25: 376-382.
15. Yamakawa, T. and T. Miki, 1986. The current mode fuzzy logic integrated circuit fabrication by the standard CMOS process, IEEE Trans. on Computers, C-35: 161-167.
16. Lee, C.C., 1990. Fuzzy logic in control systems: fuzzy logic controller, IEEE Trans. Syst., Man, Cybern., 20(2): 404-435.
17. Chun-Fei Hsu, 2007. Self-Organizing Adaptive Fuzzy Neural Control for a Class of Nonlinear Systems, IEEE Transactions on Neural Networks, 18(4): 1232-1241.
18. Jie Su, JiaRen and Haipeng Pan, 2008. An improved self-structuring neuro-fuzzy algorithm, International Conference on Information and Automation, ICIA, pp: 1123-1126.
19. Zadeh, L.A., 1975. The concept of a linguistic variable and its application to approximate reasoning: Part 1, 2 and 3," Inform. Sci., 8: 199-249, pp: 301-357 and 9: 43-80.
20. Yager, R.R., 1991. Modeling and formulating fuzzy knowledge bases using neural networks, Tech. Rep. No. MII-1111, Machine Intelligence Institute, Iona College.
21. Umano, M. and Y. Ezawa, 1991. Execution of approximate reasoning by neural network, in P m .FANSymp., pp: 267-273 (in Japanese).
22. Keller, J.M. and H. Tahani, 1992. Backpropagation neural networks for fuzzy logic, Inform. Sci., 62: 205-221.
23. Keller, J.M. and H. Tahani, 1992. Implementation of conjunctive and disjunctive fuzzy logic rules with neural networks, Int. J. Approx. Reasoning, 6: 221-240.
24. Keller, J.M., R.R. Yager and H. Tahani, 1992. Neural network implementation of fuzzy logic, Fuzzy Sets and Systems, 45: 1-12.
25. Uehara, K. and M. Fujise, 1990. Learning of fuzzy-inference criteria with artificial neural network, in Proc. of Int. Conf. on Fuzzy Logic & Neural Networks, IIZUKA '90, Vol.1, Japan, pp: 193-198.
26. Cohen, M.E. and D.L. Hudson, 1992. Approaches to the handling of fuzzy input date in neural networks, in proc. Fuzz-IEEE, pp:: 93-100.
27. Hayashi, Y., 1992. A neural expert system using fuzzy teaching input, " in Proc. Fuzz-IEEE, pp: 485-491.
28. Lee, S.C. and E.T. Lee, 1975. Fuzzy neural networks, " Math Biosci., 23: 151-177.
29. Rumelhart, D.E., G.E. Hinton and R.J. Williams, 1986. Learning representations by back-propagating errors, Nature, 323: 533-536.
30. Rumelhart, D.E. and J.L. McClelland, 1986. The PDP Research Group, Parallel Distributed Processing, 1. Cambridge, MA: MIT.
31. Ansari, A.Q. and Neeraj Gupta, 2009. MIPS-NF Instructions for fourth Generation Processor, 2nd IEEE International Conference on Computer, Control & Communication, pp: 1-5.
32. Kala, R., A. Shulkla and R. Tiwari, 2009. Fuzzy Neuro Systems for Machine Learning for Large Data Sets, IEEE International on Advance Computing Conference, IACC, pp: 541-545.
33. Yao, L., M. Steyaert and W. Sansen, 2004. A 1-V 140-μW 88-dB audio sigma-delta modulator in 90-nm CMOS, IEEE J. Solid-State Circuits, 39(11): 1809-1818.
34. Ansari, A.Q. and Neeraj Gupta, 2011. Design and Simulation of OTA-Based Activation and Membership Function for NeurofuzzySystems, International Journal of Computational Cognition, 9(1): 100-104.