Technical Report 2007- 2-002

CMOS IMPLEMENTATION OF AN ARTIFICIAL NUERON LEARNABLE TO ARBITRARY LOGICAL FUNCTIONS

V. VARSHAVSKY,
V. MARAKHOVSKY, H. SAITO

September 14, 2007

Computer Organization Laboratory
The University of Aizu
Tsuruga, Ikki-machi, Aizu-Wakamatsu City
Fukushima 965-8580, Japan

Technical Report 2007- 2-002

Title:
CMOS Implementation of an Artificial Neuron Learnable to Arbitrary Threshold Logical Functions

Authors:
V. Varshavsky, V. Marakhovsky, H. Saito

Key Words and Phrases: artificial neuron, CMOS implementation, learnable synapse, excitatory and inhibitory inputs, learning process, learning sequence, refreshment process, test function, threshold logical element, threshold logical function, Horner's scheme, Fibonacci sequence.

Abstract:
The paper offers a new methodology of designing in CMOS technology analog-digital artificial neurons learnable to arbitrary logical threshold functions of some number of variables. The problems of functional ability, implementability restrictions, noise stability, and refreshment of the learned state are formulated and solved. Some functional problems in experiments on teaching an artificial neuron to logical functions are considered. Recommendations on selection of testing functions and formation of teaching sequences are given. All results in the paper are received using SPICE simulation. For simulation experiments with analog/digital CMOS circuits, the transistor models MOSIS BSIM3v3.1, 0.8μm, level 7 are used.

Report Date:
2007/09/14

Written Language:
English

Any Other Identifying Information of this Report:

Distribution Statement:
First Issue: 10 copies

Supplementary Notes:

Computer Organization Laboratory
The University of Aizu
Aizu-Wakamatsu
Fukushima 965-8580
Japan

CMOS Implementation of an Artificial Neuron Learnable to Arbitrary Threshold Logical Functions

V. VARSHAVSKY,
V. MARAKHOVSKY[1], H. SAITO[2]

[1,2] The University of Aizu, Aizu-Wakamatsu City, Fukushima Prefecture, 965-8580 JAPAN
marak@u-aizu.ac.jp, hiroshis@u-aizu.ac.jp

Abstract: - The paper offers a new methodology of designing in CMOS technology analog-digital artificial neurons learnable to arbitrary logical threshold functions of some number of variables. The problems of functional ability, implementability restrictions, noise stability, and refreshment of the learned state are formulated and solved. Some functional problems in experiments on teaching an artificial neuron to logical functions are considered. Recommendations on selection of testing functions and formation of teaching sequences are given. All results in the paper are received using SPICE simulation. For simulation experiments with analog/digital CMOS circuits, the transistor models MOSIS BSIM3v3.1, 0.8μm, level 7 are used.

Key-Words: - artificial neuron, CMOS implementation, learnable synapse, excitatory and inhibitory inputs, learning process, learning sequence, refreshment process, test function, threshold logical element, threshold logical function, Horner's scheme, Fibonacci sequence.

1 Introduction

Hardware implementation of an artificial neuron has a number of well-known advantages over software implementation [1–5]. In its turn, a hardware artificial neuron can be implemented as a special purpose programmable controller or digital/analog circuit (device). Each of these implementations has its advantages, drawbacks, and fields of application. Commercially available neurochips can be of any of these two types. The comparative analysis of characteristics and application fields of various neurochips is beyond the scope of this paper. We will just note that digital/analog implementation has one obvious advantage over all other implementations, which is high performance.

On the other hand, digital/analog implementation, due to its internal analog nature, has rigid limitations on the class of realizable threshold functions. These limitations considerably decrease the functional possibilities of neural nets with fixed number of neurons.

Functional power of neurochip equally depends on the number of neurons that can be placed on one VLSI and functional possibilities of a single neuron. Unfortunately, it has not been properly studied yet how much these parameters affect the functional power of the neurochip. However, it is evident that decreasing area/synapse and extending the functional possibilities of a neuron are prior aims when creating new neurochips.

In [6, 7], a new type of threshold element (β-driven threshold element, β-DTE) has been offered that required one transistor per logical input. Its circuit was based on representing a threshold function in ratio form. In [8–11], a CMOS learnable neuron has been suggested on the base of β-DTE that consisted of synapses, a β-comparator, and an output amplifier. The learnable synapse of this neuron had 5 transistors and one capacitor. The neuron had one remarkable property: its implementability depended only on the threshold value and did not depend on the number of logical inputs and their weights. This fact and relatively low complexity made this neuron very attractive for usage in the next generation of digital-analog neurochips.

Frankly speaking, an artificial neuron destined for implementation of logical threshold functions it is more correct to call learnable threshold element (LTE). During learning, this device forms analog weights for binary (digital) input variables. Obviously, a real artificial neuron can be constructed on the base of LTE.

The goal of this paper is improving the LTE circuit in terms of its learnability to complicated logical threshold functions (with big value of the minimum threshold), noise-stability and ability of keeping the learned state for a long time.

When the function threshold is high, the noise-stability becomes especially important. It is determined by the smallest change of the output voltage $\min \Delta V$ of the β-comparator at the threshold. Bigger $\min \Delta V$ is attained by increasing sharpness of the β-comparator characteristic in the threshold zone. This is provided by incorporating into the β-comparator two extra transistors and selecting their functional modes.

The noise stability and, hence, the implementability of given logical functions by the LTE depends not only on the $\min \Delta V$ value but also on the threshold position of the β-comparator characteristic relatively to the threshold of the output amplifier. In the paper, a way of teaching the LTE to a given logical function is offered. This teaching method allows not only automatic positioning of the amplifier threshold to the middle of $\min \Delta V$ but also increasing $\min \Delta V$ up to $\max \min \Delta V$, which is attained when finding the minimum threshold of the function and is determined by the steepness of the β-comparator characteristic. The method is based on using three output amplifiers with different thresholds, which provide threshold hysteresis. The width of this hysteresis determines the value of $\min \Delta V$ attained during learning.

Some additional problem are stated and solved in the paper. One of the problems is maintaining the LTE in the learned state for a long time (refreshing the analog memory on capacitors). The solution of the problem uses the same idea of applying threshold hysteresis of output amplifiers.

Another problem is connected with possibility to speed up LTE learning to given logical threshold functions. This problem is solved by parallel forming weights of several input variables and by changing learning step value during learning.

The problem of functional abilities of LTE is also stated in the paper. It is obvious that LTE can implement only threshold logical functions. According to the theory of switching functions all threshold functions are monotonous. The minimum representation of monotonous functions coincides with their concise form. If the concise form of a threshold function contains only positive variables, the function is called isotonous (subclass of monotonous functions). The LTE with the simplest synopses, each of which contains only one capacitor as a memory element, can be learned only to isotonous threshold functions. As it will be shown below, using more complicated synapse circuit with two memory elements for keeping positive and negative weights it is possible to construct LTE learnable to arbitrary threshold function of some number of variables.

Finally, some functional problems in experiments on teaching LTE are considered. A set of recommendations how to choose testing functions and how to construct teaching sequences is given.

All results in the paper are received with help of SPICE simulation. For simulation experiments, transistor models MOSIS BSIM3v3.1 0.8μm (level 7) for analog/digital circuits were used. In most experiments on LTE teaching, logical threshold Horner's function of 7 and 10 variables were applied as test functions.

2 LTE Learnable to Isotonous Threshold Functions

2.1 Threshold Element with Controllable Input Weights

The conventional mathematical model of a neuron, starting from the work by McCulloch and Pitts [12], is the threshold function:

$$F = Sign\left(\sum_{j=1}^{n} w_j x_j - T\right); \quad Sign(A) = \begin{cases} 0 \text{ if } A < 0 \\ 1 \text{ if } A \geq 0 \end{cases} \tag{1}$$

where w_j is the weight of the *j*-th input and *T* is the threshold value.

Representing a threshold function as (1) implies that a threshold element is traditionally implemented by the structure shown in Fig.1.

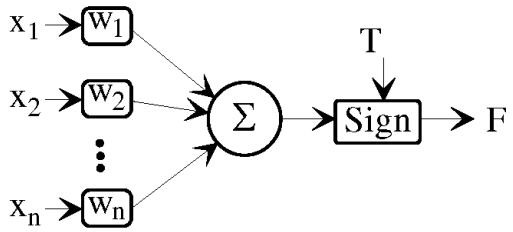

Figure 1. General structure of the neuron threshold model.

It is shown in [6, 7] that any threshold function can be represented in ratio form, as follows:

$$F = Sign\left(\sum_{j=1}^{n} w_j x_j - T\right) = Sign\left(\frac{\sum_{j\notin S} w_j x_j}{\sum_{j\in S} w_j \bar{x}_j} - 1\right) = Rt\left(\frac{\sum_{j\notin S} w_j x_j}{\sum_{j\in S} w_j \bar{x}_j}\right);\ Rt(\mathrm{A/B}) = \begin{cases} 0 \text{ if } A < B \\ 1 \text{ if } A \geq B \end{cases} \qquad (2)$$

where S is a certain subset of indexes[1] such that $\sum_{j\in S} w_j = T$. From (2) it immediately follows that CMOS implementation of a threshold element can be like that in Fig.2.

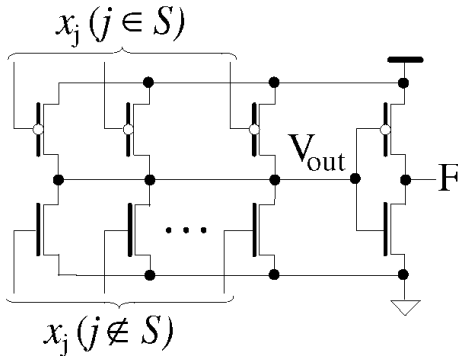

Figure 2. The β-driven threshold element (β-DTE).

The voltage V_{out} at the β-comparator output is determined by the ratio of steepnesses (β_n and β_p) of n- and p-circuits. Namely by this reason, the threshold element is called β-driven (β-DTE). The steepnesses are formed by connecting transistors of respective width in parallel.

In [8, 9], to build a threshold element with controllable input weights, a reduced ratio form is introduced:

$$F = Sign\left(\sum_{j=1}^{n} w_j x_j - T\right) = Rt\left(\frac{\sum_{j=1}^{n} w_j x_j}{T}\right) = Rt\left(\sum_{j=1}^{n} \omega_j x_j\right); \quad \omega_j = w_j / T \qquad (3)$$

that leads to the β-comparator circuit shown in Fig.3a where $\beta_{nj} = \omega_j \beta$; $\beta_n = \beta \sum_{j=1}^{n} \omega_j x_j$; $\beta_p = \beta$.

[1] To construct S it is sufficient to take any hypercube vertex that lies in the separating hyperplane and to include in S indexes of the variables having the value 1 on the vertex.

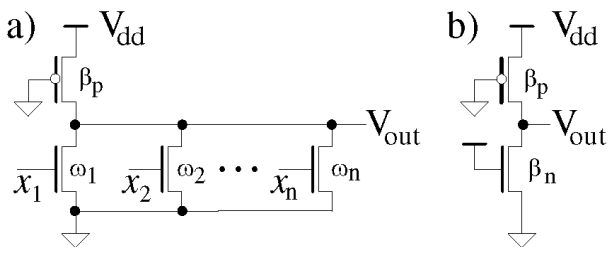

Figure 3. The β-comparator: CMOS implementation (a); equivalent circuit (b).

In Fig.3b, a circuit is shown equivalent to that in Fig.3a. The output voltage of the β-comparator is determined by the value $\alpha = \beta_n / \beta_p$ in the following way:

$$V_{out} = \begin{cases} > V_{dd}/2 \text{ if } \alpha < 1 \\ \le V_{dd}/2 \text{ if } \alpha \ge 1 \end{cases}.$$

If the output voltage of a CMOS couple (Fig.3b) $V_{out} \approx V_{dd}/2$, this means that both the transistors are in non-saturated mode since both of them meet the condition $V_{th} < V_{out} < V_{gs} - V_{th}$, $V_{gs} = V_{dd}$.[2] Hence,

$$I_n = \beta_n \left[(V_{dd} - V_{th}) V_{out} - \frac{V_{out}^2}{2} \right],$$

$$I_p = -\beta_p \left[(V_{dd} - V_{th})(V_{dd} - V_{out}) - \frac{(V_{dd} - V_{out})^2}{2} \right], \quad (4)$$

$$I_n + I_p = 0.$$

In [6] these equations were analyzed and it was shown that the suggested comparator circuit has sensitivity $\frac{dV_{out}}{d\alpha} \approx -2$ V in the point $\alpha = \beta_n / \beta_p = 1$. Hence, at the threshold level $(V_{out} = V_{dd}/2)$ the reaction of the β-comparator to a unit change of the weighted sum $\Delta V_{out} \approx |2/T|$ V, i.e. it linearly decreases as the threshold grows.

The analysis of β-DTE stability to parameter variations made in [5] showed that only β-DTE with small thresholds (≤ 3, 4) can be stably implemented. However, an artificial neuron is a learnable object and variations of many parameters (for example, technological) can be compensated during learning.

The learnable LTE on the base of β-DTE [8, 9] has a sufficiently simple control over the input weight (Fig.4): the control voltage changes the equivalent β of the respective synapse. Since the

[2] For simplicity let's assume that the threshold voltage is the same for the both transistors.

synapse can be in one of two states, conducting or non-conducting, the output voltage V_{out} of the β-comparator is formed only by the synapses which are conducting in this given moment.

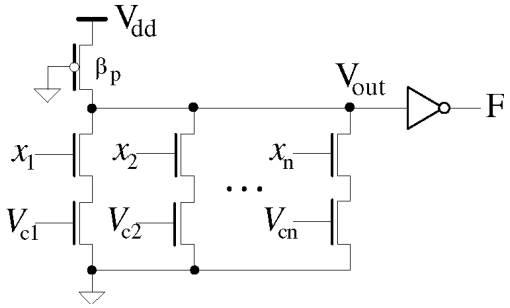

Figure 4. β-driven LTE.

Easy to understand that after the threshold is reached, adding new synapses does not change the LTE output state. It follows from this that the implementability of β-DTE and, hence, of the LTE on its base depends only on the threshold value and does not depend on the number of inputs and sum of their weights (this fact was established in [6]). The essential aspect is the sensitivity of the β-comparator to the current change at the threshold point. Since the range of β-comparator output voltage is limited within $(0 - V_{dd})$, the only way of increasing the β-comparator steepness at the threshold point is increasing non-linearity of the dependency of the β-comparator output voltage on the ratio $\alpha = \beta_n / \beta_p$.

2.2 Increasing β-comparator Sensitivity

To increase sensitivity of the β-comparator, its transistors should be in the saturated mode when the output voltage is in the threshold zone of output amplifier switching. This can be demonstrated by an example of the equivalent circuit in Fig.3b.

Let the gates of both the transistors be fed not by ground and voltage supply but by voltages V_{gs}^{p} and V_{gs}^{n}, such that both the transistors are in the saturated mode when $V_{out} = V_{dd} / 2$. Let us assume for simplicity that $V_{gs}^{p} = V_{gs}^{n} = V_{gs}$, $V_{th}^{p} = V_{th}^{n} = V_{th}$, and $0 < V_{gs} - V_{th} < V_{dd} / 2$. Then the equations for the currents flowing through the transistors can be represented as

$$I_n = \beta_n (V_{gs} - V_{th})^2 (1 + \lambda_n V_{out}),$$
$$I_p = -\beta_p (V_{gs} - V_{th})^2 \left[1 + \lambda_p (V_{dd} - V_{out})\right], \qquad (5)$$
$$I_n + I_p = 0.$$

where the parameters λ_n and λ_p reflect the small increase of the transistor currents that takes place when grows. From these equations we find

$$V_{out} = \frac{1-\alpha+\lambda_p V_{dd}}{\lambda_p + \lambda_n \alpha}, \quad \alpha = \beta_n / \beta_p \tag{6}$$

and

$$\frac{dV_{out}}{d\alpha} = -\frac{\lambda_n + \lambda_p + \lambda_n \lambda_p V_{dd}}{(\lambda_p + \lambda_n \alpha)^2}. \tag{7}$$

Let $\lambda_n = 0.03\,{}^{1}\!/_{\mathrm{V}}$ and $\lambda_p = 0.11\,{}^{1}\!/_{\mathrm{V}}$.[3] For $V_{out} = V_{dd}/2$, it is easy to calculate from (6) that $\alpha = 1.15$. Parameter α does not equal to one at this point since the values of λ_n and λ_p are different. When $V_{dd} = 5\,\mathrm{V}$ and $\frac{dV_{out}}{d\alpha} = -7.5\,\mathrm{V}$. Thus, the sensitivity of the β-comparator has increased by 3.75 times. The less λ_n and λ_p, the more the sensitivity.

In the LTE circuit (Fig.4), every synapse consists of two transistors. The gate of one transistor is fed by the input variable x_j; the gate of the other one is fed by the voltage V_{cj} that controls the variable weight (current in the synapse).

Let us first consider the lower part of the LTE β-comparator where the synapse currents are summed and replace the couples of transistors, which form synapses, by equivalent transistors with characteristics shown in Fig.5. These characteristics were obtained by SPICE simulation.

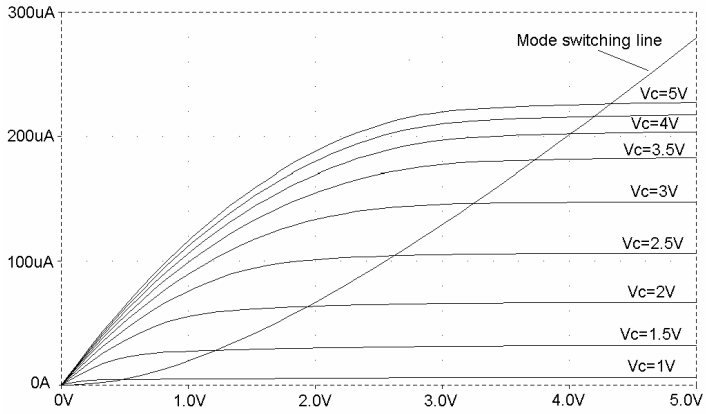

Figure 5. Characteristics of the transistor that is equivalent to the transistor couple.

To the left of the mode switching line, the transistors are in the non-saturated mode; to the right — in the saturated mode. It is easy to see from these characteristics that when $V_{out} = 2.5\,\mathrm{V}$ the equivalent transistors are in the saturated mode, if the control voltage $V_C < 2.5\,\mathrm{V}$, and in the non-saturated mode, if $V_C > 2.5\,\mathrm{V}$. Thus, the saturated mode condition restricts the range of control voltage change. Breaking this restriction leads to decreasing the output signal of the β-comparator

[3] The values of these parameters were taken from the used transistor models.

because the currents are re-distributed among the synapses. Indeed, let the smallest weight corresponds to synapse current $I_{\min}$ and adding this current to the total current of the other synapses must cause the switching of the LTE. If the synapse with the biggest current is not saturated, decreasing V_{out} because of the total current increases makes the current of this synapse smaller. The currents of other non-saturated synapses also decrease. As a result, the total current increases by a value, which is considerably smaller than $I_{\min}$. This leads to decreasing the output signal of the β-comparator.

The range, in which the control voltages of the synapses change, can be extended with help of extra n-channel transistor *M*3 incorporated into the circuit as it is shown in Fig.6.

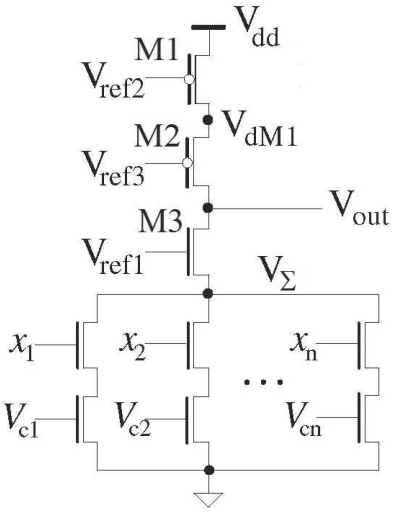

Figure 6. The modified β-comparator.

The gate of this transistor is fed by voltage V_{ref1} such that when the current provides $V_{out} \approx V_{dd}/2$, the transistor is saturated under reaction of the voltage $V_{gs} = V_{ref1} - V_{\Sigma}$. Increasing the total current through the synapses by adding a synapse with the smallest current makes V_{Σ} smaller, so that V_{gs} becomes bigger. The extra transistor opens and the extra increase of the total current compensates the change in V_{Σ}. Thus, due to the negative voltage feedback, the extra transistor stabilizes V_{Σ} and therefore stabilizes the currents through the synapses.

In Fig.6, when the control voltage of the synapse has its maximum value ($V_C = 5\,\text{V}$), the current through the synapse depends on V_{out} as it is shown in Fig.7. It looks like a transistor characteristic having two zones: the linear zone and zone of saturation. It is easy to see that when $V_{out} \approx 2.5\,\text{V}$, the synapse is in the saturated mode. When the voltage V_{ref1} gets smaller, the synapse current decreases and a change of the synapse current range narrows down. When V_{ref1} increases, the synapse current grows and the linear zone of the characteristic becomes wider that may cause the lost of current

stabilization in the working point. Thus, there is an optimum value of V_{ref1}. In all experiments $V_{ref1} = 3\,\text{V}$.

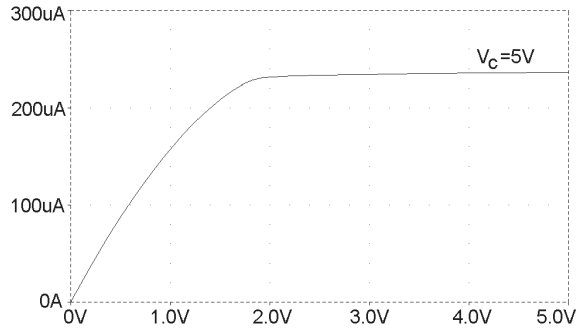

Figure 7. Dependence of the synapse current on V_{out} when $V_C = 5\,\text{V}$.

Now let us consider the *p*-channel part o f the modified *β*-comparator (Fig.6). In the working point ($V_{out} \approx V_{dd}/2$) it should provide a current corresponding to the maximum value of the threshold of realized functions. For this goal, one *p*-channel transistor can be used with the reference voltage V_{ref} providing its saturation in the working point. However, in this case the steepness of the characteristic $V_{out}(I)$ in the working point will be insufficient for good stabilization of the threshold value of the current. By this reason, the modified *β*-comparator circuit (Fig.6) uses the idea of a cascode amplifier [13, p.287]. It has two *p*-channel transistors *M*1 and *M*2 referenced by voltages V_{ref2} and V_{ref3} respectively. These reference voltages are selected so that as the comparator current grows, the transistor *M*l is saturated first and then *M*2 becomes saturated. In SPICE experiments $V_{ref2} = 3.5\,\text{V}$, $V_{ref3} = 2.5\,\text{V}$.

The dependence of voltage V_{dM1} on the current at the drain of *M*l is shown in Fig.8 (Curve 2).

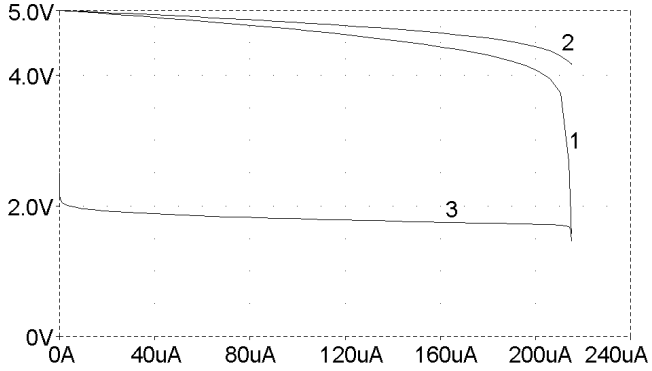

Figure 8. Curve 1 – dependency $V_{out}(I)$; Curve 2 – dependency $V_{dM1}(I)$; Curve 3 – dependency $V_{\Sigma}(I)$.

As soon as $M1$ comes into the saturation zone, the voltage V_{gs} of $M2$ begins to change with higher speed because of $V_{gs} = V_{ref3} - V_{dM1}$. The voltage drop on $M2$ sharply grows increasing the steepness of $V_{out}(I)$ (Curve 1 in Fig.8). Curve 3 in Fig.8 shows rather good stabilization of the voltage drop $V_{\Sigma}(I)$ on the synapses.

For comparison, Fig.9 contains experimental characteristics of the old and new β-comparators adjusted to the function threshold T=89.

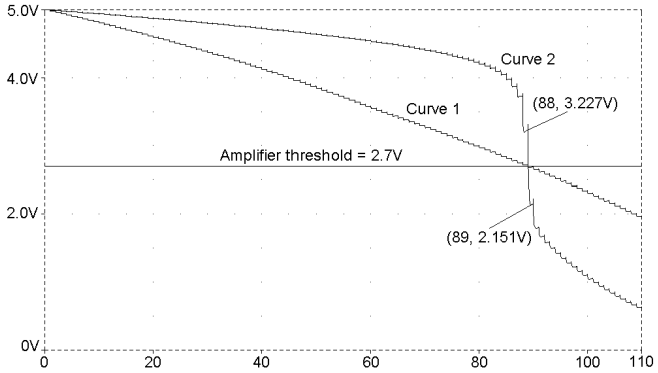

Figure 9. Comparator characteristics: curve 1 for the old comparator; curve 2 for the new one.

This experiment shows how the comparator output V_{out} depends on the number of switched synapses whose control inputs were fed by the voltage $\min V_c$ corresponding to the smallest weight of a variable. For the old comparator (Curve 1), the leap of the output voltage in the threshold point is 32mV. The characteristic of the new comparator has a much higher steepness in the threshold zone; the voltage leap in the threshold point is about 1V.

2.3 LTE Circuit and the Way of Teaching

Circuits used for forming control voltages determining weights of input variables of artificial neurons just slightly depend on the way of synapse implementation. Some of these circuits were published (for example, in [14]) and they are of about the same structure. The difference between them is mainly associated with a type of a memory element they use (a capacitor or a transistor with floating gate) and with a way of representing the values of input binary variables ({0,1} or {-1,+1}). In Fig.10, the full LTE circuit is given, which experiments were conducted with. Every its synapse contains 5 transistors and a capacitor. Two of the transistors form one of the parallel branches of the β-comparator. The input variable arrives at the gate of the lower transistor and the control voltage arrives at the gate of the upper one. This usage of the transistors, as compared to the

opposite, makes the synapse current dependable on the control voltage more linearly, reducing to several times the influence of the input variable change on the control voltage.

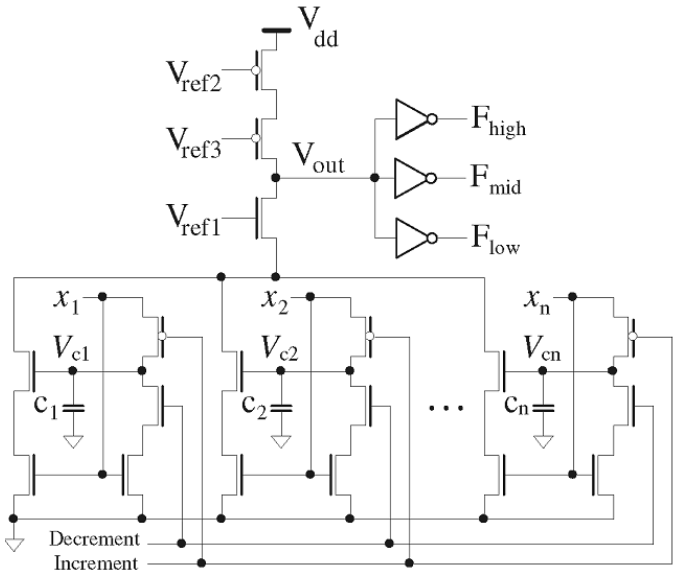

Figure 10. The LTE circuit.

During teaching, the voltage that controls the synapse current (i.e. variable weight) accumulates on a capacitor. The capacitor charge is allowed to change only when the synapse is active, i.e. when the input variable equals to "1". The capacitor charge increase or decrease is realized by approximately the same quanta that determine a learning step. The learning step is appointed on the base of required accuracy of setting the control voltages. Its value can be controlled by choosing amplitude and duration of "increment" and "decrement" signals.

When teaching LTE to fairly complicated threshold functions (with a big value of sum of weights and threshold), the learning step should be small. Usually, algorithms of LTE teaching are built so that as soon as the output signal of the LTE begins to coincide with the value of the learning function, the teaching stops. Due to the small learning step, in cases when the LTE fires after the variable with the smallest weight changes its value, the voltage leap at the output of the β-comparator can exceed the minimum permissible value, which is sufficient for amplifier firing, just by a very small value.

To increase the margin of reliability after the teaching, the LTE circuit has three output amplifiers with different (high, middle and low) sensitivity thresholds. The value of the function produced by the taught LTE is taken from the output F_{mid}. The output signals F_{high} and F_{low} are used only during teaching. After teaching, the voltage leap at the output of the β-comparator that causes switching F_{mid} will be not less than the difference between the threshold voltages of the other two amplifiers.

The control voltages, which have been set during teaching, are kept on the capacitors and can change due to parasitic leakage resistances. In this connection, one should organize the procedure of capacity memory refreshment. The three output amplifiers with different thresholds allow solving this problem, for example, by auto-correcting the control voltages using the output signal F_{mid} as a learning sequence of function's values.

The general structural scheme used when simulating the process of teaching the neuron to a given threshold logical function is shown in Fig.11.

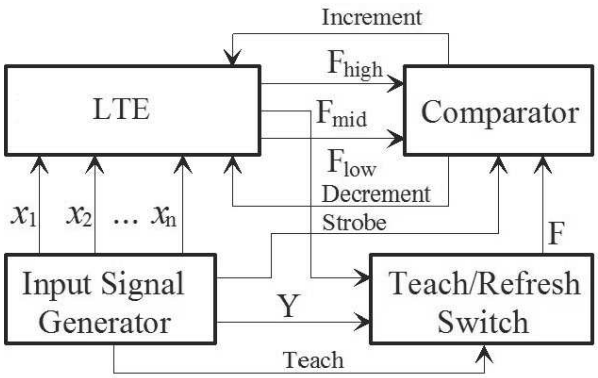

Figure 11. General scheme for experiments.

The generator of input signals periodically produces sequences of value combinations of input variables $x_1, x_2, ..., x_n$ and the sequence of values that the given logical function *Y* takes on these combinations. Teaching/refresh switch passes to its output *F* either the signal *Y* (when teaching) or the output signal F_{mid} (when refreshing). The comparator produces the signals "decrement" and "increment." Passive values of these signals are equal to "0" and "1" respectively. Their logical description looks like

$$Decrement = \overline{Y} \cdot F_{high} \text{ and } Increment = \overline{Y} \vee F_{low} \text{ when teaching;}$$

$$Decrement = \overline{F}_{mid} \cdot F_{high} \text{ and } Increment = \overline{F}_{mid} \vee F_{low} \text{ when refreshing.}$$

Physically, these signals are realized with limited amplitude and duration, determining the learning step.

In experiments with LTE learning, there is an acute problem of selecting threshold functions for teaching what determines simulation time. The duration of experiments is very important because it is often measured in hours and even days. A threshold function for teaching should has

- a short sequence of variable combinations checking all possible switches of the function value,
- a wide range of variable weights,
- a high threshold value for a given number of variables.

In more detail, this problem will be investigated in the last section of the paper. It will be shown that a function, which can be represented by the Horner's scheme $x_n(x_{n-1} \vee x_{n-2}(x_{n-3} \vee x_{n-4}(...)))$, satisfies to these requirements. For such functions, the sequence of integer values of variable weights and threshold with minimum sum forms the Fibonacci sequence. The length of the checking sequence is $n + 1$ for the Horner's function of n variables.

2.4 SPICE Simulation Results of LTD Learning

Two series of experiments on LTE teaching to given threshold functions are represented here.

The goal of the first series was to show the necessity of using a threshold hysteresis when teaching the LTE and when providing the auto-support of the LTE state after the LTE is taught. The threshold hysteresis can be obtained using three output amplifiers whose characteristics have different thresholds as it is shown in Fig. 12.

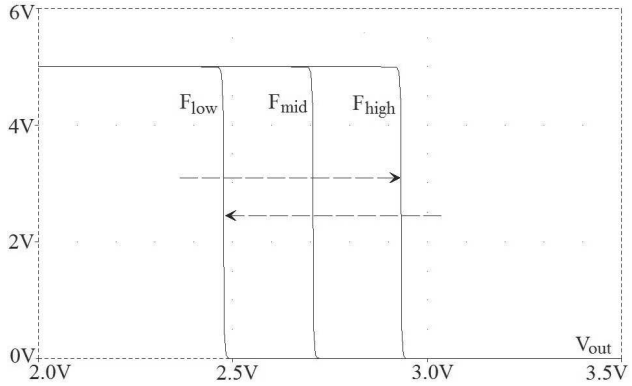

Figure 12. Static characteristics of the output amplifiers.

When the movement to the threshold is from the left, the higher value of the threshold is used for learning; when from the right, the LTE learns to the lower value. This leads to stretching out the minimum leap $\min \Delta V_{out}$ of the β-comparator output voltage in the threshold zone and to automatic positioning of the output amplifier threshold with the middle value F_{mid} into the middle of this leap. Obviously, the hysteresis width should not exceed $\max \min \Delta V_{out}$, which is defined by the parameters of the p-channel part of the β-comparator and by the minimum value $T_{\min}$ of the logical function threshold: $\max \min \Delta V_{out} = f(I_{\max \min} = I_{comp} / T_{\min})$ where I_{comp} is the comparator current in the threshold zone and $I_{\max \min}$ is the maximum current of the synapse with the smallest weight.

For the teaching, it was chosen the Horner's function of 7 variables:

$$Y_7 = x_7(x_6 \vee x_5(x_4 \vee x_3(x_2 \vee x_1))) = x_7x_6 \vee x_7x_5x_4 \vee x_7x_5x_3x_2 \vee x_7x_5x_3x_1,$$

$$\overline{Y_7} = \overline{x}_7 \vee \overline{x}_6(\overline{x}_5 \vee \overline{x}_4(\overline{x}_3 \vee \overline{x}_2\overline{x}_1)) = \overline{x}_7 \vee \overline{x}_6\overline{x}_5 \vee \overline{x}_6\overline{x}_4\overline{x}_3 \vee \overline{x}_6\overline{x}_4\overline{x}_2\overline{x}_1. \quad (8)$$

From all its possible representations in the form $Y_7 = Sign(\sum_{j=1}^{7} w_j x_j - T)$ with integer values of weights and threshold, the representation

$$Y_7 = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 - 21) \quad (9)$$

is optimum by criterion of $\min(\sum_{j=1}^{7} w_j + T) = 54$. For this representation, $T_{\min} = 21$.

The checking sequence for this function contains 8 combinations. Their order is chosen in such a way that the sequence of the corresponding function values is alternate. The combinations in the checking sequence have one remarkable property: if the LTE is taught to the optimum representation of the function, their change must cause the voltage leaps at the comparator output, which are equal in amplitude. The teaching sequence is a periodically repeated checking sequence.

To get illustrative and easily explainable results of the experiment, we had to reduce the steepness of the β-comparator characteristic. In the experiment, the β -comparator parameters were chosen so that when the threshold was 21, $\max \min \Delta V_{out}$ was equal to 0.5V. The teaching was conducted with the step equal to 10mV.

The results of teaching the LTE to the Horner's function of 7 variables with various hysteresis widths are given in Fig. 13.

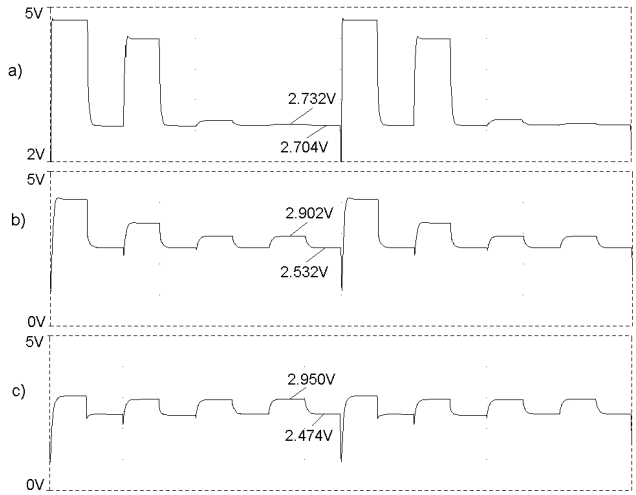

Figure 13. Output signal of the LTE β-comparator learned to the function of 7 variables: without hysteresis (a), with hysteresis of 340mV (b) and of 450mV (c).

When there was no hysteresis (Fig.13a), the LTE is learned to the function representation with threshold T=267 that strongly differed from the optimum. The voltage leaps at the comparator output vary on the checking sequence from $\min \Delta V_{out} = 28\text{mV}\ (2.732 - 2.704)$ up to $\max \Delta V_{out}$ that exceeds 2V. Obviously, after such teaching, the LTE will have a very bad noise-stability.

In the second case (Fig.l3b), the neuron was taught with a 340mV wide hysteresis and learned to the function representation with threshold T=26. The dispersion of the voltage leaps at the comparator output considerable decreased ($\min \Delta V_{0out} = 0.37\text{V}$; $\max \Delta V_{out} = 0.9\text{V}$) and the noise-stability of the LTE significantly increased.

In the third case (Fig.13c), the neuron was taught with the hysteresis of 450mV wide. The LTE was learned to a function representation, which was close to the optimum. All the voltage leaps at the output of the β-comparator were approximately the same ($\min \Delta V_{out} = 0.48V$; $\max \Delta V_{out} = 0.55\text{V}$).

By simulation it was checked the possibility to provide auto-support of the LTE in the learned state on the base of using threshold hysteresis. With this aim, the leakage resistances of control voltage capacitors were explicitly incorporated to the LTE circuit. The LTE was taught to the Horner's function of 7 variables with 450mV wide hysteresis of output amplifiers threshold. After the teaching, the learning mode was replaced by the refreshment mode. The neuron kept stably functioning on the periodically repeated checking sequence. Signals "Increment" and "Decrement" occurred from time to time correcting the control voltages on the capacitors and supporting them within the permissible limits.

The result of correcting the control voltages is easily observable in Fig.14.

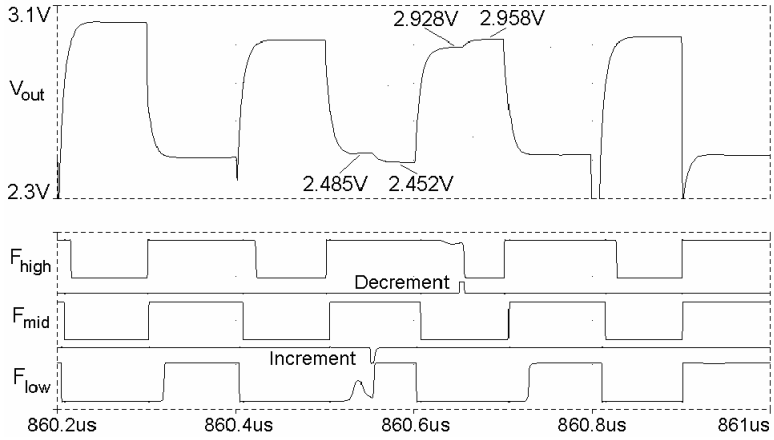

Figure 14. Correction of the control voltages in the refreshment mode of LTE operation.

As it is seen from the picture, voltage level $V_{out} = 2.485\text{V}$ corresponding to the value combination 1010110 of input variables was not sufficient to switch the output signal F_{low}. As a result, the signal "*Increment*" occurred that increased by 10mV the voltages on the synapse capacitors C_7, C_5, C_3, and C_2 causing the decrease of V_{out} by 33mV and switching F_{low}. On the next combination, 1010100, the level $V_{out} = 2.928\text{V}$ was not sufficient to switch F_{high}. The signal "*Decrement*" reduced by 10mV the voltages on C_7, C_5, and C_3 causing the increase of V_{out} by 30mV and

changing the value of F_{high}. In spite of the correction of the control voltages, the values of the output signal F_{mid} still corresponded to the values of the function on all combinations of the checking sequence.

In the second series of the experiments, the LTE was taught to a threshold function of 10 variables:

$$\begin{aligned} &Y_{10} = x_{10}(x_9 \vee x_8(x_7 \vee x_6(x_5 \vee x_4(x_3 \vee x_2x_1)))) = \\ &x_{10}x_9 \vee x_{10}x_8x_7 \vee x_{10}x_8x_6x_5 \vee x_{10}x_8x_6x_4x_3 \vee x_{10}x_8x_6x_4x_2x_1; \\ &\overline{Y}_{10} = \overline{x}_{10} \vee \overline{x}_9(\overline{x}_8 \vee \overline{x}_7(\overline{x}_6 \vee \overline{x}_5(\overline{x}_4 \vee \overline{x}_3(\overline{x}_2 \vee \overline{x}_1)))) = \\ &\overline{x}_{10} \vee \overline{x}_9\overline{x}_8 \vee \overline{x}_9\overline{x}_7\overline{x}_6 \vee \overline{x}_9\overline{x}_7\overline{x}_5\overline{x}_4 \vee \overline{x}_9\overline{x}_7\overline{x}_5\overline{x}_3\overline{x}_2 \vee \overline{x}_9\overline{x}_7\overline{x}_5\overline{x}_3\overline{x}_1. \end{aligned} \tag{10}$$

This function can be represented in the form

$$Y_{10} = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 + 34x_9 + 55x_{10} - 89). \tag{11}$$

The checking sequence for the function must contain not less then 11 combinations, which are defined by the terms of Y_{10} and $\overline{Y}_{10}$ in (10). To make the teaching sequence of function values interchanged it is needed to have the odd number of combinations in the checking sequence. For this purpose it is possible to add any combination, on which the function has the value "1". It is well known that any threshold function is a star. The top vertex of the star is the most convenient candidate to be added to the checking sequence (this improves the learning time). Finally, the checking sequence for the function (10) is

x_{10}	x_9	x_8	x_7	x_6	x_5	x_4	x_3	x_2	x_1	Y_{10}
1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	1
1	0	0	1	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0	0	0	1
1	0	1	0	0	1	1	1	1	1	0
1	0	1	0	1	1	0	0	0	0	1
1	0	1	0	1	0	0	1	1	1	0
1	0	1	0	1	0	1	1	0	0	1
1	0	1	0	1	0	1	0	0	1	0
1	0	1	0	1	0	1	0	1	1	1
1	0	1	0	1	0	1	0	1	0	0
1	1	1	1	1	1	1	1	1	1	1

The checking sequence is implemented as it is shown in Fig.15. The last graphic in it represents strobe-signal *t* that participates in forming "increment" and "decrement" signals.

In the LTE circuit, the β-comparator was adjusted to the maximum sensitivity providing at the threshold T=89 $\max\min \Delta V_{out} \approx 1$V (Fig.9), the hysteresis width was 0.85V, and the learning step was adaptive.

The learning step is defined by amplitude and duration of the "increment" and "decrement" signals. These signals provide charging and discharging the capacitors with the current of 0.15uA

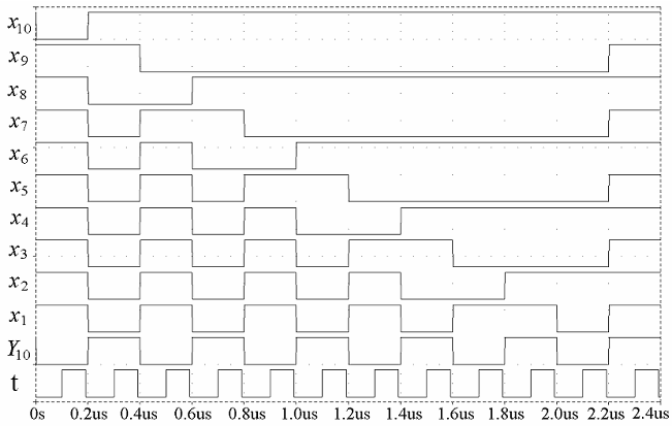

Figure 15. Single checking sequence of signal value combinations.

and can have the maximum duration equal to the duration of the strobe-signal *t* (90ns). It gives for the capacitor of 1pF the maximum learning step equal to 13.5mV.

The change of the control voltages on the synapse capacitors during the learning is shown in Fig.16. The dynamics of the leaning is easily observable. The control voltages stop changing in the

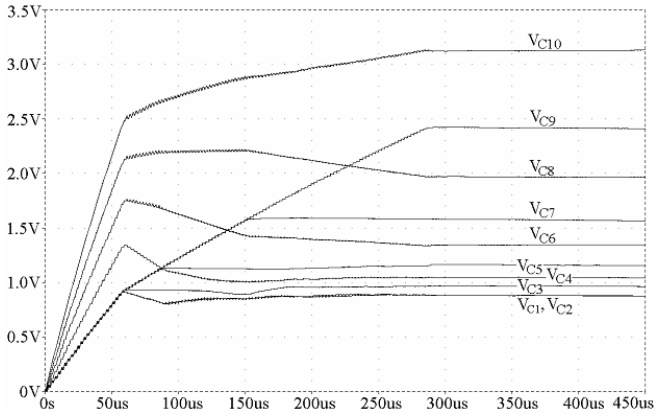

Figure 16. Changes of the control voltages on the synapse capacitors during the learning.

moment 0.28ms. This means that the learning process is over and it is possible to switch teaching mode to refreshing mode. More accurate the moment of mode switching can be defined with special control signal, which sets the switcher into refreshing mode, if the LTE output F_{mid} coincides with the output *F* of the mode switcher on all combinations of some checking sequence. In the refreshing mode, if F_{mid} and *F* not coincide on at least one combination of the checking sequence, this control signal sets the switcher into teaching mode. It means that the LTE lost the learned state and must be taught again. The refreshing mode of operation can be interrupted with evaluation process for calculation the value of logical function on some input combination. Obviously, that refreshing and evaluation have to interchange. During evaluation to receive correct results, the LTE output F_{mid} should be gated.

As it is easy seen from Fig.16, stable values of the control voltages approximately correspond to the weights of the variables in the optimum representation of the threshold function (the values are distributed close to Fibonacci numbers). At the time instance equal to 0.28ms, the teaching mode has been replaced by the refreshment mode. Starting from this moment, only rare signals "*Increment*" and "*Decrement*" appeared, correcting some control voltages.

The output signals of the LTE and the output signal of its β-comparator in one period of the checking sequence of the refreshment mode is given in Fig. 17. One can see that the smallest leap of

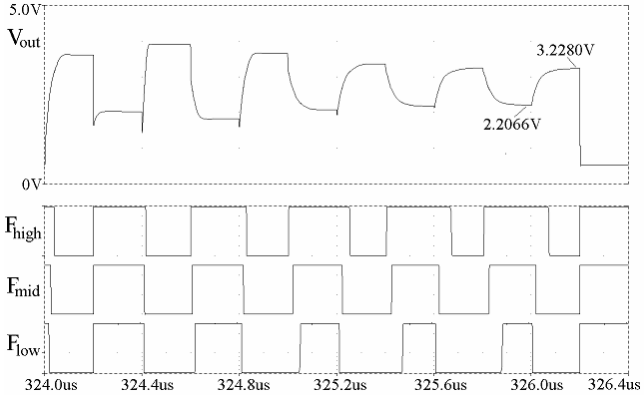

Figure 17. Picture of the signals on the outputs of the β-comparator and the LTE in the refresh mode.

V_{out} at the comparator output is 1V. The output signal V_{mid} represents the values of the realized function. In cases when the output signals F_{high} and F_{low} do not correspond to the function value, the control voltages are corrected.

2.5 Implementability Limits of the LTE

In order to study the functional power of the LTE, a number of experiments were carried out with SPICE simulation of its behavior. For all experiments with learnable threshold elements, the problem of choosing testing threshold functions is crucial. This problem will be discussed in the last section of the paper. As it was already noticed, a threshold test-function should match the following demands:

— to have the short learning sequence,

— to cover a wide range of input weights,

— to have the biggest threshold for the given number of variables.

Monotonous Boolean functions representable by Horner's scheme match all these demands. For such functions of n variables, the sequence of input weights and threshold forms Fibonacci sequence and the length of the shortest learning (checking) sequence is $n+1$ (the number of

combinations of input variable values). Experiments were made with three threshold functions for $n = 10$, 11 and 12:

$$F_{10} = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 + 34x_9 + 55x_{10} - 89),$$

$$F_{11} = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 + 34x_9 + 55x_{10} + 89x_{11} - 144), \quad (12)$$

$$F_{12} = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 + 34x_9 + 55x_{10} + 89x_{11} + 144x_{12} - 233).$$

Since the learning process was not the object of these experiments, the optimum values of control voltages were set on the synapses. The logical inputs of the LTE were fed by checking (learning) sequences.

In the first series of the experiments, $\max\min \Delta V_{out}$ (the maximum of the smallest change of β-comparator output voltage at the threshold level of 2.7V) was determined. The results of the experiments are given in the second column of Table 1. The implementability of the LTD is determined by the signal ΔV_{out} value. According to the table, the LTE learnable to functions of 12 variables is quite near the edge of implementability because of relatively small value of ΔV_{out}.

Table 1: Results of SPICE simulations.

LTE type	ΔV_{out}	(min÷ max)V_{th}	δV_{dd}
F_{10}	1V	1.88÷3.7V	0.3%
F_{11}	0.525V	1.9÷3.68V	0.2%
F_{12}	0.325V	1.97÷3.65V	0.1%

In the second series of the experiments, for fixed parameters of the comparator the range of admissible threshold voltages of the output amplifier F_{mid} has been defined under stipulation that on the range borders the comparator produced $\min \Delta V_{out}$ not less then 100mV when the LTE was in the learned state. The results are given in the third column of Table 1. The conclusion is: deviation of the amplifier threshold (e.g. because of technological parameter variations) does not essentially influence upon LTE implementability. The LTE during learning is adjusted to any threshold of the output amplifier from these ranges.

The other experiments were associated with the question: with what precision the voltages should be maintained for normal functioning of the LTE after learning. First of all, the LTE stability to supply voltage variations should be investigated. With constant values of the reference voltages, when changing the voltage supply at ±0.1% (±5mV), the dependence of the voltage V_{out} from the current flowing through p-transistors of the comparator shifts along the axis of current at ±1.5% as shown in Fig. 10. For the LTE F_{12}, the current in the working point is about $233I_{\min}$; 1.5% of this value is $3.51I_{\min}$, i.e. the shift of the characteristic is 3.5 times more than the minimum current of

the synapse. Evidently, the LTE will not function properly when the working current changes like that.

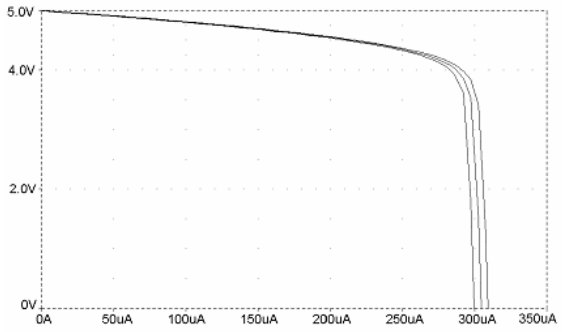

Figure 18. Behavior of the dependency $V_{out}(I_p)$ when the voltage V_{dd} changes in the interval ±0.1%.

On the other hand, taking into account the way of reference voltages producing, it is natural to assume that the reference voltages must change proportionally to the changes of the voltage supply. The effect from reference voltage change is oppositely directed to the effect of supply voltage change and partially compensates it. The experiments carried out under these conditions showed that learned LTE F_{10}, F_{11}, and F_{12} can function properly in respective ranges of supply voltage change shown in the fourth column of Table 1. To fix the borders of the ranges, the following condition was used: signal $\Delta V_{out}/2$ should be more or less than the output amplifier threshold by a value not less than 50mV.

The control voltages of the synapses were set up with the accuracy of 1mV. With what accuracy should they be maintained after the learning? Evidently, the LTE will not function properly if with the same threshold of the output amplifier the total current of the synapses will drift by $I_{\min}/2$ in one or the other side. Experiments were conducted to determine the permissible range $\pm\delta V_C$, in which the control voltage V_C of one of the synapses (with minimum and maximum currents) can change while the control voltages of the other synapses are constant. The condition for fixing the range borders was the same as in the previous series of experiments. The obtained results are given in Table 2.

Table 2: Results of SPICE simulations.

Type	$\delta I_{S\min}$	$\delta V_{C\min}$	$\delta V_{C\max}$
F_{10}	$\pm 0.42 I_{\min}$	±5.3% (±46mV)	±0.60% (±17mV)
F_{11}	$\pm 0.40 I_{\min}$	±4.7% (±40mV)	±0.73% (±27mV)
F_{12}	$\pm 0.34 I_{\min}$	±3.8% (±32mV)	±0.23% (±10mV)

In the second column of the table, the permissible ranges of synapse current change are shown. The third and fourth columns contain the limits of change of the control voltages. These limits define corresponding changes of current in synapses with minimum and maximum weights.

It is possible to make the following conclusion basing on Table 2 data: since all the control voltages of synapses in the LTE should be maintained simultaneously, their maintenance as accurate as units of millivolts should.

3 LTE Learnable to Arbitrary Threshold Functions

A threshold function with positive input weights is an isotonous Boolean function. Such a function can be realized by an artificial neuron (LTE) with only excitatory inputs. However, most problems solved by artificial neural networks either require inhibitory inputs. If the input type (excitatory or inhibitory) is known beforehand, the problem of inverting the weight sign is solved trivially by inverting the respective variable. Otherwise, the neuron should have synapses capable of forming the weight and type of the input during the learning, using only increment and decrement signals. The possibility of building such synapses for the LTE is the subject of this section.

3.1 Statement of the Problem

The behavior of a β-DTE is described by a threshold function in ratio form [6]. To build the LTE, it is convenient to represent threshold functions in reduced ratio form:

$$F = Rt(\sum_{j=1}^{n} w_j x_j / T) = Rt(\sum_{j=1}^{n} \omega_j x_j) \tag{13}$$

where $\omega_j = w_j / T$ and $Rt(A) = \begin{cases} 0 \text{ if } A < 1, \\ 1 \text{ if } A \geq 1. \end{cases}$

The simplest and obvious way of solving this problem is doubling the number of variables (and synapses) feeding the LTE inputs by both x_j and their inversions $\bar{x}_j$ with input weights a_j and b_j respectively. Note that doubling the number of synapses does not lead to cutting down the number of realizable threshold functions because the implementability of LTE depends only on the threshold value and does not depend on the sum of the input weights or number of synapses. Quite the contrary, incorporating extra inverse inputs increases the number of realizable threshold functions of n variables by $2n$ times.

Let in a certain isotonous threshold function $Rt(\sum_{j=1}^{n} \omega_j x_j)$ some variables $x_i \in Y$ are inverted while other variables $x_j \in Z$ $(i \neq j, Z \bigcup Y = X)$ are not. Then

$$F = Sign(\sum_{x_j \in Z} w_j x_j + \sum_{x_i \in Y} w_i \bar{x}_i - T) = Sign(\sum_{x_j \in Z} w_j x_j + \sum_{x_i \in Y} w_i (1 - x_i) - T) =$$

$$Sign(\sum_{x_j \in Z} w_j x_j - \sum_{x_i \in Y} w_i \bar{x}_i - (T - \sum_{x_i \in Y} w_i)) = Rt\left(\frac{\sum_{x_j \in Z} \omega_j x_j - \sum_{x_i \in Y} \omega_i x_i}{1 - \sum_{x_i \in Y} \omega_i} \right) \quad (14)$$

where $\omega_j = w_j / T$. It is easy to see from (14) that negative weights use can be reduced to inverting the variables and vice versa. Normalized threshold of a function represented by Rt-formula with negative weights is equal to $1 - \sum_{x_i \in Y} \omega_i$.

The circuit of a neuron synapse capable of forming both positive and negative weights of an input variable is made of two simple synapses as shown in Fig.19.

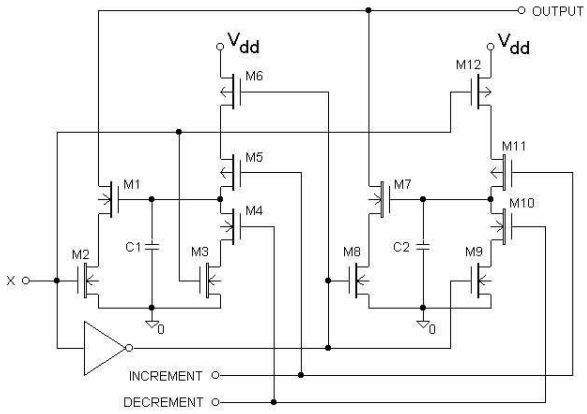

Figure 19. Synapse forming positive and negative weights of the input variable.

It is easy to see that the LTE with such synapses realizes the threshold function

$$Rt\left(\frac{\sum_{j=1}^{n} (a_j - b_j) x_j}{1 - \sum_{j=1}^{n} b_j} \right) \quad (15)$$

where a_j and b_j are weights brought to the threshold. They are defined by voltages on the capacitors C_1 and C_2 for x_j and $\bar{x}_j$ respectively.

On the other hand, for the case of doubling the synapses number, it follows from (14) that the threshold function realized by the LTE must be

$$Rt\left(\frac{\sum_{j=1}^{n} (a_j - b_j) x_j}{1 - \sum_{j=1}^{n} (b_j - a_j) Sign(b_j - a_j)} \right) \quad (16)$$

(for keeping the limitations on weights and thresholds).

It is easy to see that if in every pair (a_j, b_j) one of the weights is equal to zero, then expressions (15) and (16) coincide and have a view:

$$F = Rt(\sum_{x_j \in Z} a_j x_j + \sum_{x_i \in Y} b_i \bar{x}_i) = Rt\left(\frac{\sum_{x_j \in Z} a_j x_j - \sum_{x_i \in Y} b_i x_i}{1 - \sum_{x_i \in Y} b_i}\right). \quad (17)$$

It follows from the above that when teaching a LTE with such synapses it is desirable to change the input weights (a_j, b_j) by such a way that one of the weights in each pare goes to zero. More over, as it will be shown below, this condition provides the maximum level of LTE implementability.

It is difficult to conclude from (15) and (16) that synaptic weights affect the neuron implementability. Let us look how the β-comparator operates (Fig.6). The sizes of *p*-transistors and reference voltages V_{ref2} and V_{ref3} determine the current I_{th} when the output voltage of the β-comparator is equal to the output amplifier threshold. As a first approximation, the smallest change of the current is $I_0 = I_{th} / T$ and $\Delta V_{out} = kI_0$ where *k* is the steepness of the β-comparator voltage-current characteristic at the threshold of the output amplifier. However, if $a_j \neq 0$ and $b_j \neq 0$, then via each *j*-th synapse an additional current flows determined by the value of $\min(a_j, b_j)$. Thus, the approximate value of the smallest current can be obtained from the equation

$$I_0 = \frac{I_{th}}{T} - I_0 \sum_j \min(a_j, b_j)$$

and

$$I_0 = \frac{I_{th}}{T(1 + \sum_j \min(a_j, b_j))}. \quad (18)$$

It follows from (18) that if the value of ΔV_{out} is fixed, the biggest realizable threshold depends on $\min(a_j, b_j)$ as

$$T \leq \frac{kI_{th}}{\Delta V_{out}(1 + \sum_j \min(a_j, b_j))}. \quad (19)$$

Thus, keeping the implementability level requires either increasing I_{th} during the learning (that is actually associated with some difficulties) or providing $\min(a_j, b_j) = 0$ for any *j* by modifying the synapse circuit and changing the learning algorithm.

In [20] several modifications of synapse circuits have been suggested and for each of them existence of stable decisions, which the LTE is able to keep to realize non-isotonous threshold functions, has been proved. Unfortunately, authors could not find on-chip learning algorithms leading to these decisions. One of possible solutions is proposed in the next subsection. This solution provides on-chip learning process, which gives convergence independently from initial conditions and uses modified synapse circuit.

3.2 The Problem Decision

The same general structure scheme, which is shown in Fig.11, is used when simulating the process of teaching the LTE to an arbitrary given threshold logical function. The "Input Signal Generator" periodically produces checking sequence of value combinations of input variables and the sequence of values that the given logical function takes on these combinations. The isotonous logical function (10) depending on 10 variables is chosen as a test function. The checking sequence for this function represented in Fig.15. Non-isotonous functions can be derived from this function, by inverting some variables with help of inverters. The generator is supposed to be implemented separately from the LTE. Other blocks of the general scheme are assumed on-chip implemented. Bellow schematics for all of them are shown. In the schematics, widths of all transistors are pointed out to make experiments repeatable.

The "Teach/Refresh Switch" passes to its output F either the signal Y (when teaching) or the signal F_{mid} (when refreshing) and realizes the logical function

$$F = Y \,\&\, Teach \vee F_{mid} \,\&\, \overline{Teach}\,.$$

Its schematic is very simple and realized in Fig.20.

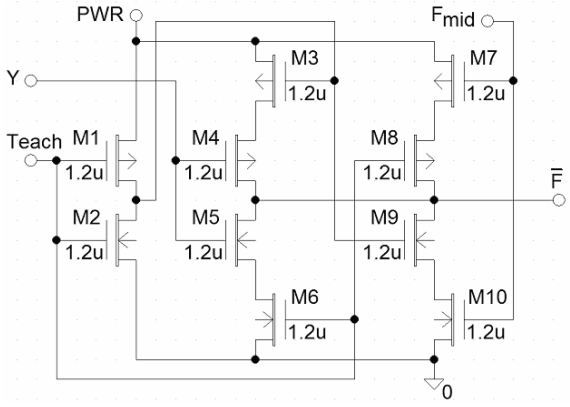

Figure 20. Schematic of the switch.

The "Comparator" produces "Decrement" and "Increment". Its schematic is shown in Fig. 21. In the schematic the "Increment" signal is designated as $incr\_p$ to point out that this signal controls p-channel transistors. Its function together with implementation description is

$$incr\_p = F_{low} \vee \overline{F} \vee \bar{t} = \overline{\overline{F}_{low} \cdot \overline{\overline{F} \vee \bar{t}}}$$

where $\overline{F}$ is the switch output and $\bar{t}$ is the strobe-signal inversion. This function is realized on the transistors M_1 – M_6 and M_{19} – M_{22}. The function logic contains embedded current mirror on the transistors M_3 and M_4, which restricts the "Increment" current through p-channel transistors

controlled by the signal $incr\_p$. The width of the transistor M_3 provides the 0.15uA current through *p*-channel transistor of the minimum width (1.2u).

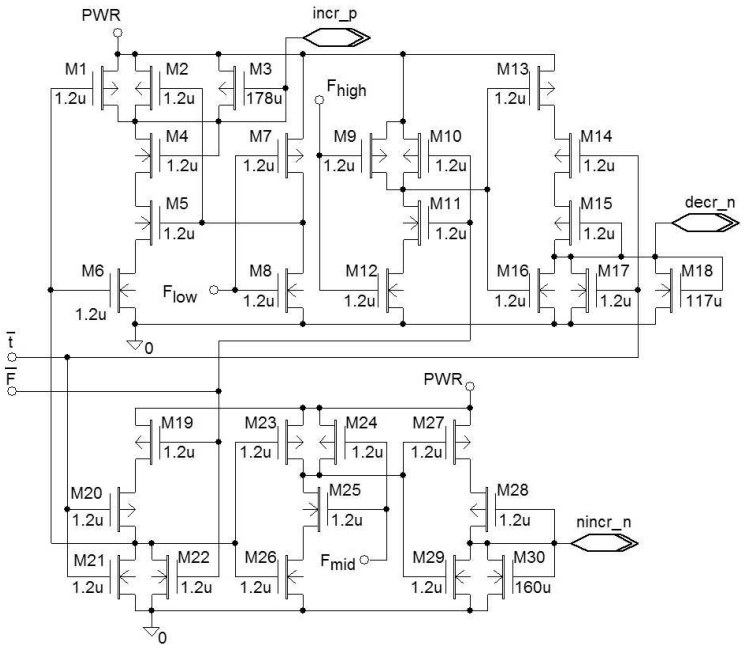

Figure 21. Schematic of the "Comparator".

"Decrement" signals consist of two signals: $decr\_n$ and $nincr\_n$. The main signal $decr\_n$ has the logical function

$$decr\_n = F_{high} \cdot \overline{F} \cdot t = \overline{\overline{F_{high}} \cdot \overline{\overline{F}} \vee \overline{t}}$$

that is implemented on transistors $M_9 - M_{18}$. The function implementation contains the embedded current mirror on the transistors (M_{15}, M_{18}), which restricts the signal $decr\_n$ amplitude. The transistor M_{18} of 117u width provides the "decrement" current 0.15uA through *n*-channel transistors of minimum width (1.2u) controlled by the signal $decr\_n$.

Additional "Decrement" signal $nincr\_n$ is used to create an additional force that pulls down voltages of the capacitors corresponding to $\min(a_i, b_j)$ up to the ground potential during LTE learning. It has the logical function

$$nincr\_n = F_{mid} \cdot F \cdot t = \overline{\overline{F_{mid}} \cdot \overline{\overline{F}} \vee \overline{t}}$$

and is implemented on transistors $M_{19} - M_{30}$. The last stage of the signal implementation contains the incorporated current mirror on transistors (M_{28}, M_{30}) that restricts to 0.11uA the current in *n*-channel transistors of minimum width controlled by the signal.

The LTE itself consists of β-comparator with output amplifiers and synapses. The schematic of the β-comparator is presented in Fig.22.

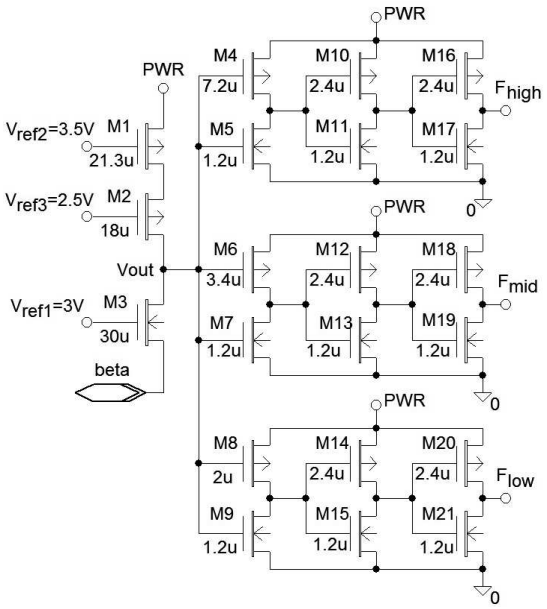

Figure 22. Schematic of the β-comparator with output amplifiers.

In this picture, all parameters of transistors and values of reference voltages are pointed out. All amplifiers are constructed as a serial connection of three inverters. The amplifier with the output F_{mid} has the threshold equal to 2.7V. The thresholds of amplifiers with outputs F_{low} and F_{high} are equal to 2.3V and 3.15V respectively. Thus, the width of threshold hysteresis is 850mV.

The full synapse circuit of the LTE learnable to arbitrary threshold functions is introduced in Fig.23. It is constructed on the base of the synapse circuit in Fig.19. Voltages V_{C1} and V_{C2} on the capacitors control the synapse current flowing through pairs of transistors (M_5, M_{23}) or (M_6, M_{24}). The circuit has to input logical variables x and $\bar{x}$. The variable $\bar{x}$ can be derived with help of an inverter. The voltages V_{C1} and V_{C2} on the capacitors C_1 and C_2 correspond to the positive a and negative b weights respectively. If $V_{C1} < V_{th}$ or $V_{C2} < V_{th}$ (here V_{th} is the threshold voltage of n-channel transistors), it means that $a = 0$ or $b = 0$ because the corresponding transistor pair will be closed.

The signals $incr\_p$ and $decr\_n$ increments and decrements the capacitor voltages through pairs of transistors (M_1, M_3), (M_2, M_4) and (M_7, M_{25}), (M_8, M_{26}) respectively depending on the value of input variables (x, $\bar{x}$).

Two pseudo nMOS inverters on transistor pairs (M_{27}, M_{37}) and (M_{28}, M_{38}) are sensitive elements of capacitor voltages close to V_{th}. Voltage $V_{ref\,4} = 3.9$V fixes the conductivity of their p-channel transistors. Output signals G_1 and G_2 of these elements control conductivity of two pairs of transistors (M_{15}, M_{21}) and (M_{16}, M_{22}) respectively. Each pair of these transistors opens when the voltage on corresponding capacitor (V_{C1} or V_{C2}) exceeds 675mV. Two inverters (transistors M_{29},

M_{39} and M_{30}, M_{40}) with outputs G_3 and G_4 invert the signals G_1 and G_2. These inverters control transistors M_9 and M_{10}, each of which open when the voltage of corresponding capacitor (V_{C1} or V_{C2}) exceeds 635mV. Signals G_3 and G_4 write also the information about the sign of the synapse weight in the latch (outputs Q and $\overline{Q}$) on transistors (M_{31} – M_{36} and M_{41}, M_{42}). The latch keeps information when voltages of both signals G_3 and G_4 exceed the threshold of the latch inputs. When Q=Log.1, the weight sign is positive. If Q=Log.0, the sign is negative.

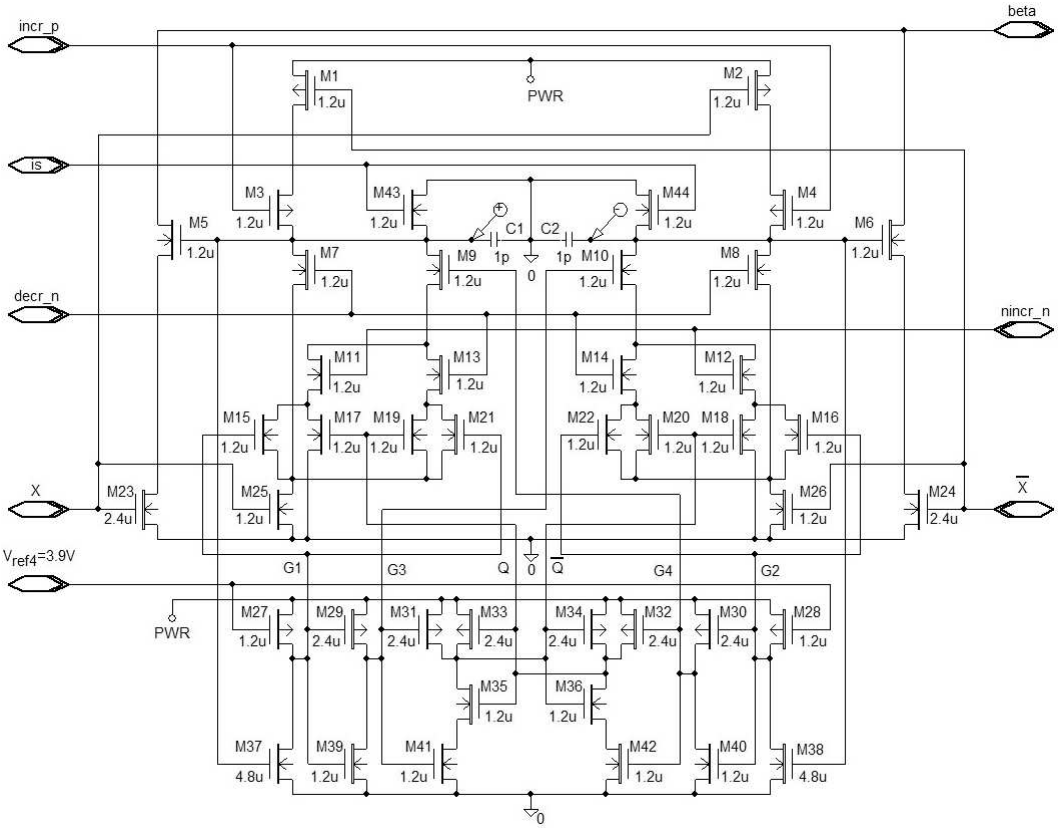

Figure 23. The LTE Synapse implementation.

Signals G_1–G_4, $decr\_n$, $nincr\_n$, and the latch Q, and input variables ($x, \overline{x}$) control conductivity of additional decrement chains for each capacitor. For the capacitor C_1, these chains are described by the expression $M_9(M_{11}(M_{15}M_{25} \vee M_{17}) \vee M_{13}(M_{19} \vee M_{21})M_{25}$. For the capacitor C_2 the expression for chains is $M_{10}(M_{12}(M_{16}M_{26} \vee M_{18}) \vee M_{14}(M_{20} \vee M_{22})M_{26}$.

The initial setting signal is controls the transistors M_{43} and M_{44}, which serve only for initial setting the voltages on the capacitors C_1 and C_2.

During learning, the synapse works by the fallowing way. Let us suppose that initially $V_{C1} < V_{th}$ and $V_{C2} > V_{th}$. Then the signals G_1 and G_2 will be equal to "Log.1" and "Log.0" respectively and the latch Q will be in the state Q=Log.0 (negative weight sign). There are two cases.

First, the sign of input variable weight is negative, i.e. it coincides with the state of the latch. In this case the signal $nincr\_n$ together with twice increased by amplitude signal $decr\_n$ pulls the voltage V_{C2} to the 0V by the chain (M_9, M_{13}, M_{19}, M_{25}) and by the chain (M_9, M_{11}, M_{15}, M_{25}) respectively. At the same time the signals $incr\_p$ and $decr\_n$ try to set the voltage V_{C2} corresponding to the weight of the input variable $\bar{x}$.

Second, the sign of the input variable weight is positive, i.e. it does not coincide with the state of the latch. In this case, the learning sequence will provide that for the capacitor C_1 increment steps caused by the signals $incr\_p$ will prevail decrement steps caused by the signals $decr\_n$, $nicr\_n$ and the voltage V_{C1} will grow. As soon, as V_{C1} will exceed V_{th} the sensitive inverter (M_{27}, M_{37}) closes the transistors M_{15}, M_{21} stopping action of the signal $decr\_n$ by additional chain and the signal $nicr\_n$. This inverter also switches the inverter (M_{29}, M_{39}), which, in its turn, opens the transistor M_{10} enabling action of the signals $decr\_n$ by additional chain and the signal $nicr\_n$. After that, the voltage V_{C1} will continue to grow up to the weight value and the signals $decr\_n$, $nicr\_n$ will pull down the voltage V_{C2} by tree chains: (M_8, M_{26}), (M_{10}, M_{14}, M_{20}, M_{26}), and (M_{10}, M_{12}, M_{18}).

As soon, as the voltage V_{C2} reaches 675mV, the sensitive inverter (M_{28}, M_{38}) opens the transistors M_{16}, M_{22} and, when $V_{C2} = 635mV$, switches the inverter (M_{30}, M_{40}), which, in its turn, closes the transistor M_9 and switches the letch Q into the state Q=Log.1 (positive weight sign). Output signals of the latch close the transistors (M_{18}, M_{20}) and open the transistors (M_{17}, M_{19}). Difference of critical values of the voltage V_{C2}, which lead to switching the sensitive element (M_{28}, M_{38}) and the inverter (M_{30}, M_{40}), is very important because in this case switching of the letch only slightly changes the condition of the capacitor C_2 discharging and voltage V_{C2} continues to go down up to ground potential.

3.3 Results of SPICE Simulation

All experiments on teaching of the LTE to non-isotonous (antitonous) threshold functions were made for functions obtained by inverting some variables in the isotonous threshold Horner's function (10) of 10 variables. Below results of SPICE simulating LTE learning are presented only for two test-functions: for the isotonous function (10) and for the non-isotonous function (antitonous) derived from (10) by inverting variables with even indexes

$$Y_{10} = x_{10}\bar{x}_9 \vee x_{10}x_8\bar{x}_7 \vee x_{10}x_8x_6\bar{x}_5 \vee x_{10}x_8x_6x_4\bar{x}_3 \vee x_{10}x_8x_6x_4x_2\bar{x}_1;$$

$$\bar{Y}_{10} = \bar{x}_{10} \vee x_9\bar{x}_8 \vee x_9x_7\bar{x}_6 \vee x_9x_7x_5\bar{x}_4 \vee x_9x_7x_5x_3\bar{x}_2 \vee x_9x_7x_5x_3x_1. \qquad (20)$$

Another form of the function (20) representation is

$$Y_{10} = Sign(-x_1 + x_2 - 2x_3 + 3x_4 - 5x_5 + 8x_6 - 13x_7 + 21x_8 - 34x_9 + 55x_{10} - 34)\,. \qquad (21)$$

The checking sequence for these two functions represented in Fig.15. In experiments, the duration of keeping one combination is 200ns and the checking sequence takes 2.4us. The learning sequence is cyclically repeated checking sequence.

Fig.24 shows the LTE learning process to the function (10) starting from the initial state, in which $V_{C1} = V_{C2} = 0\text{V}$ for all synapses. In this figure, designations ΔV_j of curves denote voltage difference $V_{C1} - V_{C2}$ for the synapse of *j*-th variable.

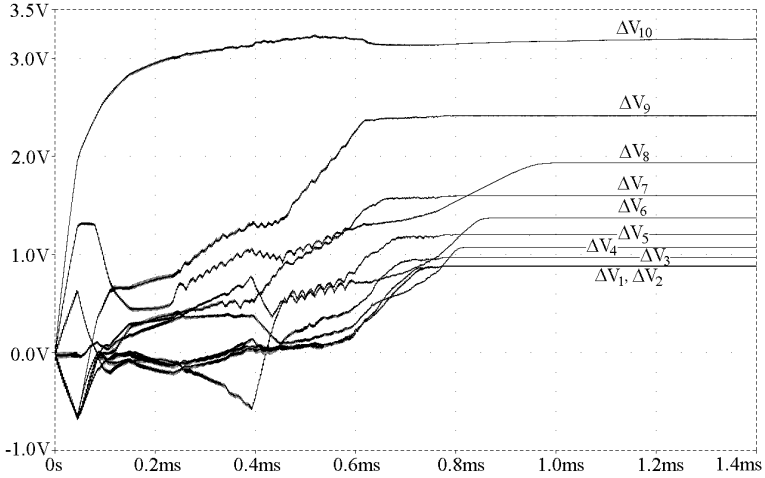

Figure 24. The LTE learning to the function (10).

It is easy to see from Fig.24 that all curves reach stable states for the time equal to 0.95ms or for 395 learning cycles and the weights of all variables are positive. As measurements show, the decision is found for 0.75ms (313 cycles). It means that up to this time all $\min(a_j, b_j) < V_{th}$ and do not act on producing the output signal of the β-comparator. At the time instant, equal to 0.95ms, all $\min(a_j, b_j) = 0\text{V}$.

Fig.25 illustrates the process of LTE learning to the function (20) starting from the same initial state.

It is possible to conclude, analyzing Fig.25, that this learning process has the same time parameters as the process in Fig.24. Signs of variable weights and their values are determined correctly: all odd variables have positive weights and all weights of even variables are negative.

For the proposed procedure of on-chip LTE learning to arbitrary threshold functions of some number of variables, it is incomprehensible how to prove its convergence analytically. A plenty of SPICE-simulation experiments have been done to be sure that this procedure possess of convergence and doesn't depend from a type of threshold functions and initial conditions.

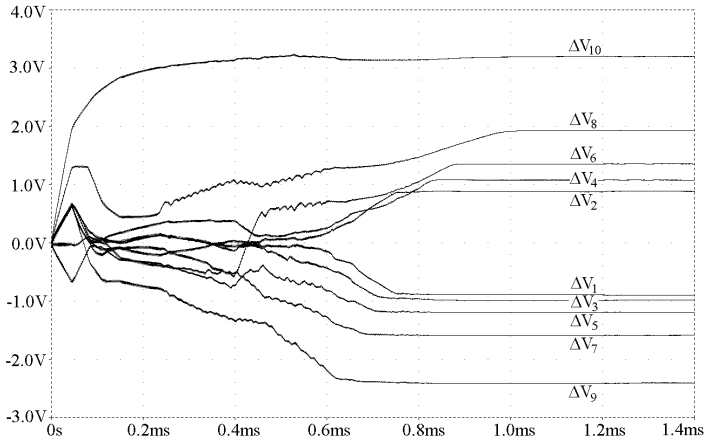

Figure 25. The LTE learning to the function (20)

In Fig.26 the process of LTE learning to the function (10) is presented for one of the worst cases when from the initial state, in which all synaptic weights are the least negative ($V_{C1} = 0\text{V}$, $V_{C2} = 5\text{V}$), the LTE is taught to all positive weights. The picture shows correct result of the learning.

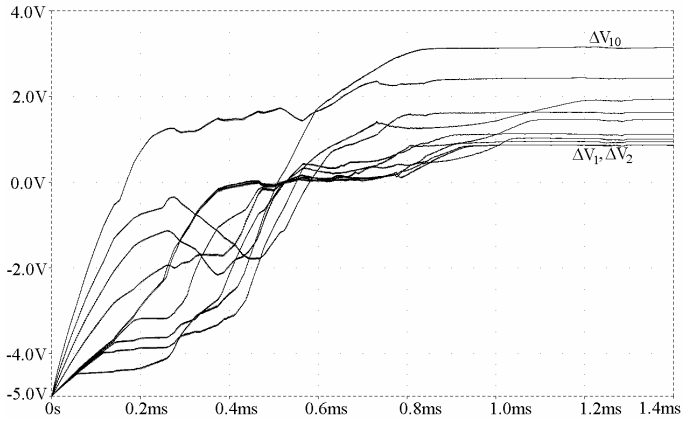

Figure 26. The process of LTE learning to the function (10) from the initial state of the least negative weights.

Now it is possible to confirm that, if the LTE is taught to realize some threshold function, using its state as initial, it can be repeatedly taught to any other one.

4 Some Functional Problems in Experiments with LTE Learning

Developing a hardware implementation (e.g. CMOS) of artificial neurons with critical parameters such as threshold value of realizable functions, number of inputs, values and sum of input weights is a hard technical problem that nevertheless has to be solved [1-8]. While tasks of purely logical design can be solved more or less efficiently in an analytic way, tasks of physical

design necessarily require computer simulation (e.g. SPICE simulation). Computer simulation can and must answer the following questions:

– What are the limiting parameters of an artificial neuron of certain type?

– Are these parameters attainable during the teaching?

A modern learnable artificial neuron, which is implemented as a hardware device and oriented to reproducing complicated threshold functions (sum of input weights and threshold >1000), is just a sophisticated analog-digital device, whose maximum functional power is attained, in many cases, by using effects of the second and even the third order in transistor behavior. This, in its turn, requires using models of higher levels (e.g. BSIM3v3.1). Therefore, the dependencies of neuron behavior on the synapse parameters are, generally speaking, non-linear. Because of this, the neuron for simulation should have a wide range of synaptic weights that covers all the range of values, which is studying. This, in its turn, requires that the neuron behavior should be simulated under all necessary combinations of the input signals. It should also be taken into account that simulation results strongly (and sometimes crucially) depend on the parameters of the transistor model in usage. Hence, to get results with a certain level of generality, a number of simulations using different models (e.g. of different manufacturers) should be conducted. Thus, to get an answer to the question about the maximum functional power of the artificial neuron, a serious experimental work is needed, which volume obviously depends linearly on the number of variable value combinations used in every experiment.

The existence of a number of neuron circuit parameters, providing reproducing a threshold function of limiting complexity, does not mean yet that the voltages controlling the input weights and threshold for this function can be attained during the teaching. If a control voltage V can change in the interval $V_{\min} \le V \le V_{\max}$ and $w_{\max}$ is the maximum value of corresponding input weight (or threshold), then $\Delta V = (V_{\max} - V_{\min}) / w_{\max}$ is the value determining required precision of teaching. Taking into account the possible non-linearity of the dependence $w(V)$ and necessity to compensate during teaching technological variations of transistor parameters (geometrical sizes, thresholds, etc.), precision of setting the control voltages should be $\delta V = k\Delta V$ (k < 1). A teaching system can be considered as a complex non-linear analog-discrete control system with feedback delay. Its analytic study is very difficult, so again one arrives to the necessity of computer simulation as the basic tool of his research.

In order to provide required precision, the increment of the voltage controlling the synaptic weight in one-step of teaching (exposing one combination of variable values) should be $\le \delta V$. Hence, to simulate the process of teaching the artificial neuron to reproduce a threshold function with $w_{\max} = 100 \div 200$, every combination from the learning sequence should be exposed about

1000 times or even more, without respect to selecting the teaching strategy. Because of this, SPICE simulation of the teaching process takes hours. Naturally, duration of the simulation process linearly depends on the length of the learning sequence.

Thus, if it is desirable to get reasonable simulation time for highly complicated artificial neurons, for test tasks one should look for threshold functions with learning sequences of the minimum length and with fixed values of $w_{\max}$. Values of input weights should cover all the value range as tightly as possible. Functions of this type are the subject of the section. Some results of threshold logic that have been known many years ago, at least as scientific folklore, will be used.

4.1 Bearing Sets of Threshold Functions and Checking Sequences

The geometrical model of threshold functions $Y = Sign(\sum_{j=1}^{n} w_j x_j - \eta)$ is a separating hyperplane with the equation $\sum_{j=1}^{n} w_j x_j - \eta = 0$. If the combinations of variable values correspond to vertexes of a unit hypercube, the threshold function has the value "Log.1" on the vertices that have positive distance to the separating hyperplane (*T* set) and the value "Log.0" on those, whose distance to the separating hyperplane is negative (*F* set). Traditionally, the task of synthesizing a threshold element by given sets *T* and *F* is reduced to solving a linear programming task:

$$\text{finding } \min\left(\sum_{j=1}^{n} w_j + \eta\right) \text{ at the conditions } \begin{cases} \sum_{x_j \in T} w_j x_j - \eta \geq 0 \\ \sum_{x_j \in F} w_j x_j - \eta < 0 \end{cases}.$$

Note that the system of inequalities is redundant since some inequalities majorize some others.

Without loss of generality, only threshold functions with $w_j > 0$ will be discussed further. Such threshold functions correspond to isotonous Boolean functions (monotonous functions with only positive variables) and can be realized by neurons with only excitatory inputs. By definition, for monotonous functions $f(X)$, from $X_j > X_i$ it follows that $f(X_j) \geq f(X_i)$ where X_j and X_i are certain combinations of variable values. Or, for isotonous threshold functions

$$\sum_{x_k = 1 \in X_j} w_k x_k - \eta > \sum_{x_k = 1 \in X_i} w_k x_k - \eta .$$

A monotonous Boolean function in a unit hypercube corresponds to a "star", i.e. a set of subcubes that have at least one common vertex (star vertex) [23]. For an isotonous function, the star vertex is $X_{\max} = \{1,1,...,1\}$; for an antitonous function (inversion of the isotonous one) – $X_{\min} = \{0,0,...,0\}$. The vertex lying on the maximum diagonal of a subcube (at the maximum distance) from the top of the star will be referred to as bearing vertex. The set of bearing vertices for

star T will be called bearing set T_0 and for star F – bearing set F_0. It is easy to see that vertices $X \in T_0$ are the minimum and vertexes $X \in F_0$ are the maximum in the respective subcubes. Hence, to solve the linear programming task, it is enough to use only the inequalities corresponding to the bearing sets.[4]

A subcube of dimension *m* in an *n*-dimensional hypercube corresponds to a conjunction of range $n-m$ in the concise form of a Boolean function.[5] This conjunction determines the bearing vertex, namely: for the set T_0, coordinate x_j has the value "Log.l", if x_j appears in the conjunction and "Log.0" otherwise; for the set F_0, coordinate x_j has the value "Log.0", if x_j appears in the conjunction and "Log.l" otherwise. For example, for $n=7$, $x_1x_3x_6 \Leftrightarrow 1010010$, $\bar{x}_1\bar{x}_3\bar{x}_4\bar{x}_6 \Leftrightarrow 0100101$. Thus, the number of vertices in the sets T_0 and F_0 equals to the number of terms in the minimum Boolean forms of the threshold function and its inversion.

It obviously follows from above that, if the artificial neuron is taught to recognize bearing sets, it recognizes corresponding threshold function as well. A learning sequence that consists of input variable value combinations belonging to bearing sets will be referred to as a bearing learning sequence. The length of the bearing learning sequence can vary in a wide range: from $n+1$ for the function $Y = Sign\left(\sum_{j=1}^{n} x_j - n\right)$ up to $2C_n^{n/2} = \dfrac{2 \cdot n!}{(n/2)!(n/2)!}$ for the function $Y = Sign\left(\sum_{j=1}^{n} x_j - n/2\right)$ with odd *n*.

4.2 Test Functions

The length of the test sequence as a function of the number of variables means nothing, if it is not correlated with complexity of the threshold function. A natural question arises about estimating threshold function complexity. For simulation tasks, which are discussed here, this complexity is associated with implementability of an artificial neuron. It varies depending on a circuit solution. For a *v*-CMOS artificial neuron [3-5], its implementability and, hence, threshold function complexity estimation is determined by the sum of the input weights. For a β-driven artificial neuron [6-8], implementability and complexity are determined only by the threshold value. Keeping in mind these two types of complexity estimation for threshold functions, we will use two criteria of efficiency for test functions, namely: $C_1 = L(n)/\eta$ and $C_2 = L(n)/\sum_{j=1}^{n} w_j$ where $L(n)$ is the

[4] This result has been known in the threshold logics since late 50's or early 60's. As the times are remote, giving any particular references is difficult.

[5] For a monotonous function, the concise form coincides with the minimum form.

length of the learning sequence in the number of bearing combinations. The less values of these criteria the more efficient the function will be in teaching.

Let us start from a simple example considering two threshold functions:

$$Y_1 = sign\left(\sum_{j=n}^{n} x_j - n\right);\ \mathrm{Y}_1 = \bigwedge_{j=1}^{n} x_j;\ \overline{\mathrm{Y}}_1 = \bigvee_{j=1}^{n} \overline{x}_j \text{ and}$$

$$Y_2 = Sign\left((n-1)x_n + \sum_{j=1}^{n-1} x_j - n\right);\ Y_2 = x_n \cdot \bigvee_{j=1}^{n-1} x_j;\ \overline{\mathrm{Y}}_2 = \overline{x}_n \vee \bigwedge_{j=1}^{n-1} \overline{x}_j .$$

Both the functions have the same number of the bearing combinations and $L(n) = n+1$. Since the threshold is the same for Y_1 and Y_2, both functions also have the same value of the first efficiency criteria C_1, $C_1(Y_1) = C_1(Y_2) = 1 + 1/n$. The only advantage of Y_2 is that there is an input with the weight $n-1$. At the same time $C_2(Y_1) = 1 + 1/n$, $C_2(Y_2) = (1/2) + 1/(n-1)$. Since $C_2(Y_1) > C_2(Y_2)$, Y_2 is more preferable as a test function. There arises a question, if it possible to derive test functions with the highest efficiency for the both criteria.

Let us consider Boolean functions that can be represented in Horner's scheme:

$$H_1(n) = x_n \vee x_{n-1}(x_{n-2} \vee x_{n-3}(x_{n-4} \vee ...)),$$

$$H_2(n) = x_n(x_{n-1} \vee x_{n-2}(x_{n-3} \vee x_{n-4}(...))),$$

and call them Horner's functions of the first and second type respectively. Note that according to De Morgan's law, inverted functions of the first type are functions of the second type relatively inverted variables, and vice versa, inverted functions of the second type are functions of the first type relatively inverted variables.

Let $N[T_{01}(n)] = N[F_{02}(n)]$, $N[T_{02}(n)] = N[F_{01}(n)]$ be the numbers of vertices in the bearing sets of Horner's functions of *n* variables of the first and the second type. It is easy to see that $H_1(n) = x_n \vee H_2(n-1)$ and $H_2(n) = x_n H_1(n-1)$. Therefore, $N[T_{01}(n)] = N[F_{01}(n-1)] + 1$, $N[F_{01}(n)] = N[T_{01}(n-1)]$ and $L(n) = L(n-1) + 1 = n + 1$. Hence, a Horner's function of *n* variables has the shortest bearing learning sequence.

Statement 1: The first and the second type of Horner's functions of *n* variables are threshold functions with the same vectors of input weights $W = \{w_n, w_{n-1}, ..., w_1\}$ differing only in their threshold values.

It is easy to find, directly applying De Morgan's theorem, that Horner's functions of the first and the second type are dual[6], i.e. $H_1(n) = H_2^d(n)$.

---

[6] Functions $f(X)$ and $\varphi(X)$ are called dual, if $f(X) = \overline{\varphi(\overline{X})}$.

For any threshold function $f(X) = Sign(\sum_{j=1}^{n} w_j x_j - \eta)$, the following is true:

$$\overline{f(X)} = Sign\left(-\sum_{j=1}^{n} w_j x_j + \eta - 1\right) \text{ and}$$

$$f^d(X) = \overline{f(\bar{X})} = Sign\left(-\sum_{j=1}^{n} w_j (1 - x_j) + \eta - 1\right) =$$

$$Sign\left[\sum_{j=1}^{n} w_j x_j - \left(\sum_{j=1}^{n} w_j - \eta + 1\right)\right]$$

that proves the statement 1.

Statement 2: Input weights of a threshold functions represented by Horner's scheme form the Fibonacci sequence.

It follows directly from the minimum form for Horner's functions that

$$w_n = \eta_1(n) = \eta_2(n-1) = w_{n-1} + w_{n-2}$$

where $\eta_1(n)$ and $\eta_2(n)$ are thresholds for Horner's functions of *n* variables of the first and the second type respectively. Solving the difference equation with the initial conditions $w_1 = w_2 = 1$, we get

$$w_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right];$$

$$\eta_1(n) = w_n;\ \eta_2(n) = w_{n+1};\ \sum_{j=1}^{n} w_j = w_{n+1} - 1.$$

At the first glance, Horner's functions look like functions of *n* variables with extreme parameters (sum of input weights, threshold, etc.). However, this is not so, as it is possible to see from simple examples. Already for 4 variables there is a threshold function

$$x_1 x_2 x_3 \vee (x_2 \vee x_3) x_4 = Sign(x_1 + 2x_2 + 2x_3 + 3x_4 - 5) \quad (22)$$

with the sum of input weights more than that of Horner's functions. From the function dual to (20), by deleting inversions of variables and multiplying it by x_5 the next function is derived

$$(x_2 x_3 \vee (x_1 \vee x_2 \vee x_3) x_4) x_5 = Sign(x_1 + 2x_2 + 2x_3 + 3x_4 + 5x_5 - 9). \quad (23)$$

This function has the threshold more than that of the second type Horner's function of 5 variables. However, both functions (22) and (23) have bearing learning sequences of the length equal to *n*+3. Note that within practically interesting values of maximum input weights, thresholds, and sums of input weights (100-1000), Horner's functions are an excellent example of test functions.

Finally, the following Horner's functions can be recommended as test functions with the shortest learning sequences:

$$Y_8 = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 - 21),$$

$$Y_9 = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 + 34x_9 - 34),$$

$$Y_{10} = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 + 34x_9 + 55x_{10} - 55),$$

$$Y_{11} = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 + 34x_9 + 55x_{10} + 89x_{11} - 89),$$

$$Y_{12} = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 + 34x_9 + 55x_{10} + 89x_{11} + 144x_{12} - 144),$$

$$Y_{13} = Sign(x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 13x_7 + 21x_8 + 34x_9 + 55x_{10} + 89x_{11} + 144x_{12} + 233x_{13} - 233).$$

Table 3 contains bearing sets T_{0j} and F_{0j} for functions Y_j. In this table, Combinations of variable values $\{x_1, x_2, ..., x_n\}$ corresponds to decimal equivalents of binary numbers $\sum_{j=1}^{2} x_j 2^{j-1}$.

Table 3: Bearing sets for Horner's functions of the first type

n	T_0	F_0
8	85,86,88,96,128	63,79,83,84
9	171,172,176,192,256	127,159,167,169,170
10	341,342,344,352,384,512	255,319,335,339,340
11	683,684,688,704,768,1024	511,639,671,679,681,682
12	1365,1366,1368,1376,1408,1536,2048	1023,1279,1343,1359,1363,1364
13	2731,2732,2736,2752,2816,3072,4096	2047,2559,2687,2719,2727,2729,2730

5 Conclusion

The proposed LTE has many attractive features. It is rather simple for hardware implementation in CMOS technology. Its β-comparator has a very high sensitivity to current changes providing the possibility of getting the smallest voltage leap at the comparator output equal to 1V when the threshold of the realized function is 89 and to 325mV, if the threshold is 233. The implementability does not depend on sum of input weights and is determined only by the function threshold. Such an LTE can perform very complicated functions, for example, logical threshold functions of 12 variables. Carried out experiments confirms this for functions of 10 variables. More over, during LTE learning all dispersions of technological and functional parameters of the LTE circuit are compensated.

For enhancement of functional abilities the new circuit of the LTE synapse has been proposed, which gives to LTE opportunity to have both excitatory and inhibitory inputs. The LTE with such synapses can be taught to arbitrary threshold function of some number of variables in the case when it is not known beforehand which inputs are inhibitory and which are excitatory. The on-chip

learning procedure, which allows teaching the neuron to arbitrary threshold function of 10 or less variables, has been also proposed. The solution is based on the well-known fact that any Boolean function of *n* variables can be represented as an isotonous function of 2*n* variables (x_j and $\bar{x}_j$). The circuit in Fig.23 realizes this idea in the pure form. The function looks like

$$y = Sign(\sum_{j=1}^{n} w_j x_j + \sum_{j=1}^{n} v_j \bar{x}_j - T) = Rt(\sum_{j=1}^{n} a_j x_j + \sum_{j=1}^{n} b_j \bar{x}_j)$$

where, if $a_j \neq 0$, then $b_j = 0$ and visa versa.

Ability to determine the type of logical variable inputs during the learning increases the number of realizable functions in 2*n* times as compared with the isotonous LTE implementation (*n* is a number of variables).

We believe that the proposed LTE and its learning procedure can be very useful in many important applications including development of real artificial neurons. Functional power of neurochips depends on not only the number of neurons, which can be placed on one VLSI, but also on functional possibilities of a single neuron. It is evident that extending functional possibilities of a neuron is the prior aim when creating new neurochips. Especially it is very important in the case of a neuron implementation as a digital/analog circuit.

The main drawback of the proposed LTE is high requirements to stability of the supply voltage. This drawback looks to be peculiar to all circuits with high resolution, for example, digital-analog and analog-digital converters. It is natural to assume that rather slow changes of the supply voltage (with period not less then tens milliseconds) will be compensated during learning and refreshing. Its fast changes can cause the lost of the learned information. It would be reasonable to study the possibility of compensating the operational parameters LTE by circuit facilities.

References

[1] C. Mead, *Analog VLSI and Neural Systems*, Addison-Wesley, 1989.

[2] S.M. Fakhraie, K.C. Smith, *VLSI-Compatible Implementations for Artificial Neural Networks*, Kluwer, Boston-Dordrecht-London, 1997.

[3]. T. Shibata, T. Ohmi, "Neuron MOS binary-logic integrated circuits: Part 1, Design fundamentals and soft-hardware logic circuit implementation", *IEEE Trans. on Electron Devices*, Vol.40, No.5, 1993, pp. 974 – 979.

[4] T. Ohmi, T. Shibata, K. Kotani, "Four-Terminal Device Concept for Intelligence Soft Computing on Silicon Integrated Circuits", *Proc. of IIZUKA'96*, 1996, pp. 49 – 59.

[5] T. Ohmi T "VLSI reliability through ultra clean processing". *Proc. of the IEEE*, Vol.81, No.5, 1993, pp. 716-729

[6] V. Varshavsky, “Beta-Driven Threshold Elements”, *Proceedings of the 8-th Great Lakes Symposium on VLSI, IEEE Computer Society*, Feb. 19-21, 1998, pp. 52 – 58.

[7] V. Varshavsky, “Threshold Element and a Design Method for Elements”, filed to Japan's Patent Office, Jan.30, 1998, JPA H10-54079 (under examination).

[8] V. Varshavsky, “Simple CMOS Learnable Threshold Element”, *International ICSC/IFAC Symposium on Neural Computation*, Vienna, Austria, Sept.23-25, 1998.

[9] V. Varshavsky, “CMOS Artificial Neuron on the Base of Beta-Driven Threshold Elements”, *IEEE International Conference on Systems, Man and Cybernetics*, San Diego, CA, October 11-14, 1998, pp. 1857 – 1861.

[10] V. Varshavsky, “Synapse, Threshold Circuit and Neuron Circuit”, filed to Japan's Patent Office on Aug. 7, 1998, the application number is JPA-H10-224994.

[11] V. Varshavsky, “Threshold Element”, filed to Japan's Patent Office on Aug. 12, 1998, the application number is JPA-H10-228398.

[12] S. McCulloch, W. Pitts, “A Logical Calculus of the Ideas Imminent in Nervous Activity”, *Bulletin of Mathematical Biophysics*, 5, 1943, pp.115 – 133.

[13] F.E. Allen, D.R. Holberg, *CMOS Analog Circuit Design*, Oxford University Press, 1987.

[14] A. Montalvo, R. Gyurcsik, J. Paulos, “Toward a General-Purpose Analog VLSI Neural Network with On-Chip Learning” , *IEEE Transactions on Neural Networks*, Vol.8, No.2, March 1997, pp. 413 – 423.

[15] V. Varshavsky, V. Marakhovsky, “Learning Experiments with CMOS Artificial Neuron”, Lecture *Notes in Computer Science 1625, ed. by Bernard Reusch, Computational Intelligence Theory and Applications. Proceedings of the 6th Fuzzy Days International Conference on Computational Intelligence*, Dortmund, Germany, May 25-27, Springier, 1999, pp. 706 – 707.

[16] V. Varshavsky, V. Marakhovsky, “Beta-CMOS implementation of artificial neuron”, *SPIE's 13th Annual International Symposium on Aerospace/Defense Sensing, Simulation, and Controls. Applications and Science of Computational Intelligence II*, Orlando, Florida, April 5-8, 1999, pp. 210 – 221.

[17] V. Varshavsky, V. Marakhovsky, “Beta-CMOS Artificial Neuron and Implementability Limits”, *Lecture Notes in Computer Science 1607, Jose Mira, Juan V. Sanchez-Andves (Eds), Engineering Applications of Bio-Inspired Artificial Neural Networks. Proceedings of International Work-Conference on Artificial and Natural Neural Networks (IWANN'99)*, Spain, June 2-4, Springier, Vol. 11, 1999, pp. 117 – 128.

[18] V. Varshavsky, V. Marakhovsky, “The Simple Neuron CMOS Implementation Learnable to Logical Threshold Functions”, *Proceedings of International Workshop on Soft Computing in Industry (IWSCI'99)*, June 16-18, 1999, Muroran, Hokkaido, Japan, IEEE, 1999, pp. 463 – 468.

[19] V. Varshavsky, V. Marakhovsky, "Implementability Restrictions of the Beta-CMOS Artificial Neuron", *The Sixth International Conference on Electronics, Circuits and Systems (ICECS'99)*, Pafos, Cyprus, September 5-8, IEEE, 1999, pp. 401 – 405.

[20] V. Varshavsky, V. Marakhovsky, "Non-Isotonous Beta-Driven Artificial Neuron", *Proceedings of SPIE, Applications and Science of Computational Intelligence III*, April 24-27, Orlando, Florida, Volume 4055, 2000, pp. 250 – 257.

[21] V. Varshavsky, V. Marakhovsky, "Artificial Neuron Learnable to Threshold Functions", *WSEAS Transactions on Circuits and Systems*, Issue 1, Vol. 3, Jan. 2004, pp. 141 – 148.

[22] V. Varshavsky, V. Marakhovsky, H. Saito, "Some Functional Problems in Experiments on teaching Artificial Neurons", 8-*th International Conference on Humans and Computers (HC-2005)*, Aizu-Wakamatsu, Japan, August 31 – September 2, 2005, pp. 339 – 344.

[23] R.H. Urbano, R.K. Mueller, "A topological method for the determination of the minimal forms of a Boolean function", *IRE Transactions on Electronic Computers*, Vol.5, 1956, pp. 126 – 132.