CMOS Implementation of an Artificial Neuron Training on Logical Threshold Functions

V. VARSHAVSKY, V. MARAKHOVSKY[1], H. SAITO[2]
[1]Saint-Petersburg State Polytechnic University,
Polytechnicheskaya str., 29, Saint Petersburg,
RUSSIA
[2] The University of Aizu
Aizu-Wakamatsu City, Fukushima Prefecture, 965-8580
JAPAN
marak@aivt.ftk.spbstu.ru, hiroshis@u-aizu.ac.jp

Abstract: - This paper offers a new methodology for designing in CMOS technology analog-digital artificial neurons training on 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 logical functions to an artificial neuron are considered. Recommendations are given on selecting testing functions and generating teaching sequences. All results in the paper are obtained using SPICE simulation. For simulation experiments with analog/digital CMOS circuits, 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]. The hardware implementation of an artificial neuron can take the form of either a special purpose programmable controller or digital/analog circuit (device). Each type of implementations has its advantages, drawbacks, and fields of application. Although analog/digital implementation has the advantage of high performance, there are rigid limitations on the class of realizable threshold functions due to its analog nature. These limitations considerably decrease the functional possibilities of neural nets that have a fixed number of neurons.

The functional power of a neurochip depends equally on the number of neurons that can be placed on one VLSI and the functional possibilities of a single neuron. Unfortunately, the effects of these parameters on the functional power of the neurochip have not been studied. However, before creating new neurochips, it is necessary to decrease the area/synapse and extend the functional possibilities of a neuron.

In [6, 7], a new type of threshold element (β-driven threshold element, β-DTE) was 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 was proposed on the base of β-DTE that consisted of synapses, a β-comparator, and an output amplifier. The learnable synapse of this neuron had five transistors and one capacitor. The neuron had one remarkable property: its implementability depended only on the threshold value and not on the number of logical inputs or their weights. This fact coupled with its relatively low complexity made this neuron very attractive for use in the next generation of digital-analog neurochips.

An artificial neuron designed for implementation of logical threshold functions it is more correctly called a learnable threshold element (LTE). During learning, this device creates analog weights for binary (digital) input variables. Obviously, an actual artificial neuron can be constructed based on LTE.

The goal of this paper is to improve the LTE circuit in terms of its learnability for complicated logical threshold functions (with a large value of the minimum threshold), noise-stability, and ability to maintain 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. Larger $\min\Delta V$ is attained by increasing the sharpness of the *β*-comparator characteristic in the threshold zone. This is achieved 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 relative to the threshold of the output amplifier. This paper offers a method to teach the LTE a given logical function. This teaching method allows not only automatic positioning of the amplifier threshold to the middle of $\min\Delta V$ but also increases $\min\Delta V$ 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 uses 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 issues are addressed in the paper. One problem 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 the threshold hysteresis of output amplifiers. The second issue concerns the possibility of speeding up LTE learning on given logical threshold functions. This problem is solved by forming weights of several input variables simultaneously and by changing the learning step values during learning. The problem of functional abilities of LTE is also examined in this 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 (a subclass of monotonous functions). The LTE with the simplest synopses, each of which contains only one capacitor as a memory element, can be taught only to isotonous threshold functions. As will be shown below, by using more complicated synapse circuits with two memory elements for keeping positive and negative weights it is possible to construct an LTE that is learnable for an arbitrary threshold function of some number of variables. Finally, some functional problems in experiments on teaching LTE are considered and a set of recommendations is given on how to choose testing functions and construct teaching sequences.

All results in the paper were obtained using SPICE simulation. For the 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 for 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:

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

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.

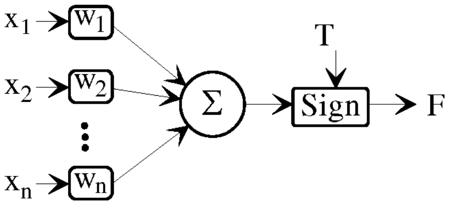


Fig. 1 General structure of the neuron threshold model.

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

$$F = Sign\left(\sum_{j=1}^{n} w_j x_j - T\right) = Sign\left(\frac{\sum_{j\notin S} w_j x_j}{\sum_{j\in S} w_j \bar{x}_j} - 1\right) =$$

$$Rt\left(\frac{\sum_{j\notin S} w_j x_j}{\sum_{j\in S} w_j \bar{x}_j}\right);\ Rt(A/B) = \begin{cases} 0 \text{ if } & A < B \\ 1 \text{ if } & A \ge B \end{cases} \quad (2)$$

where *S* is a certain subset of indexes[1] such that

[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.

$\sum_{j\in S} w_j = T$. From (2) it immediately follows that CMOS implementation of a threshold element can be like Fig.2.

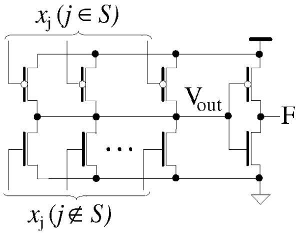


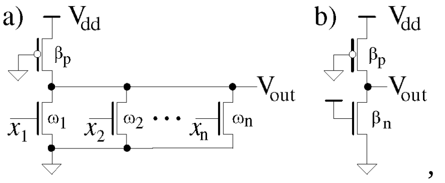
Fig. 2 β-driven threshold element (β-DTE).

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. For this reason, the threshold element is called β-driven (β-DTE). The steepnesses are formed by connecting transistors of corresponding widths in parallel.

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

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

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



,

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

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) $V_{out} \approx V_{dd}/2$, it means that both 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,

$$I_n = \beta_n\left[(V_{dd} - V_{th})V_{out} - \frac{V_{out}^2}{2}\right],$$
$$I_p = -\beta_p\left[(V_{dd} - V_{th})(V_{dd} - V_{out}) - \frac{(V_{dd} - V_{out})^2}{2}\right], \qquad (4)$$
$$I_n + I_p = 0.$$

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 at 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 increases.

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

The learnable LTE based on 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.

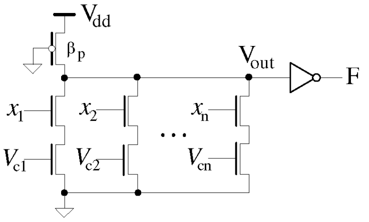


Fig. 4 β-driven LTE.

Since the 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.

Clearly, once 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

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

essential aspect is the sensitivity of the β-comparator to changes in the current at the threshold point. Since the range of β-comparator output voltage is restricted within ($0-V_{dd}$), the only way to increase the β-comparator steepness at the threshold point is to increase the 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 the 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 using the example of the equivalent circuit in Fig.3b.

Let the gates of both 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

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

where the parameters λ_n and λ_p reflect the small increase in the transistor currents that takes place when V_{ds} increases. From these equations we find

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

and

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

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 1 at this point since the values of λ_n and λ_p are different. When $V_{dd}=5\,\mathrm{V}$ and $\alpha=1.15$, $\dfrac{dV_{out}}{d\alpha}=-7.5\,\mathrm{V}$. Thus, the sensitivity of the β-comparator has increased 3.75 times. The smaller the values of λ_n and λ_p, the greater 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 (the 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 using SPICE simulation.

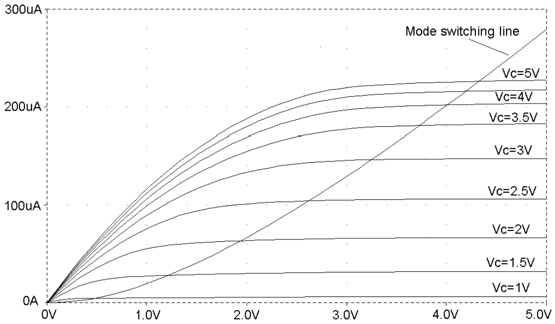


Fig. 5 Characteristics of the transistor that is equivalent to the transistor couple.

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

The range in which the control voltages of the synapses change can be extended by incorporating an extra n-channel transistor M_3 into the circuit as

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

shown in Fig.6. 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 the reaction of the voltage $V_{gs} = V_{ref1} - V_{\Sigma}$. Increasing the total current through the synapses by adding a synapse that has the smallest current makes V_{Σ} smaller, so that V_{gs} becomes larger. The extra transistor opens and the increase in the total current compensates for 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.

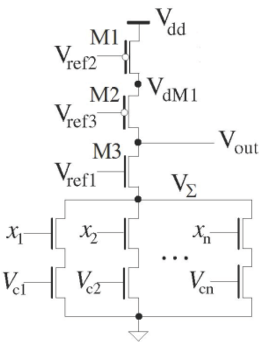


Fig. 6 Modified β-comparator.

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

When V_{ref1} increases, the synapse current grows and the linear zone of the characteristic widens, which may cause the loss of current stabilization at the working point. Thus, there is an optimum value of V_{ref1}. In all experiments $V_{ref1} = 3$ V.

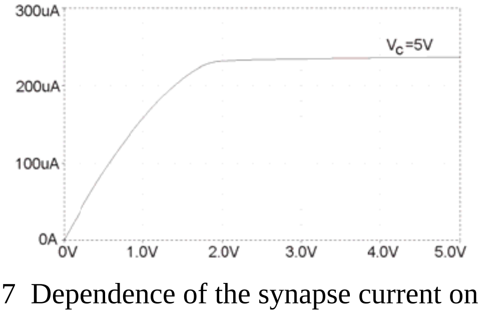


Fig. 7 Dependence of the synapse current on V_{out} when $V_C = 5$ V.

Now let us consider the p-channel part in the modified β-comparator (Fig.6). At 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. To achieve this goal, one p-channel transistor can be used with the reference voltage V_{ref} providing its saturation at the working point. However, in this case the steepness of the characteristic $V_{out}(I)$ at the working point will be insufficient for good stabilization of the threshold value of the current. For this reason, the modified β-comparator circuit (Fig.6) uses the notion of a cascode amplifier [13, p.287] that 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 increases, first the transistor M_1 is saturated and then M_2 becomes saturated. In SPICE experiments $V_{ref2} = 3.5$ V, $V_{ref3} = 2.5$ V.

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

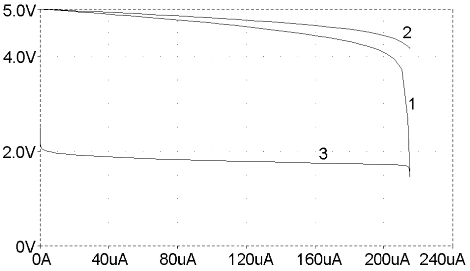


Fig. 8 Curve 1 – dependency $V_{out}(I)$; Curve 2 – dependency $V_{dM1}(I)$; Curve 3 – dependency $V_{\Sigma}(I)$.

As soon as M_1 comes into the saturation zone, the voltage V_{gs} of M_2 begins to change at a higher speed because $V_{gs} = V_{ref3} - V_{dM1}$. The voltage drop on M_2 sharply grows, thereby 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.

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

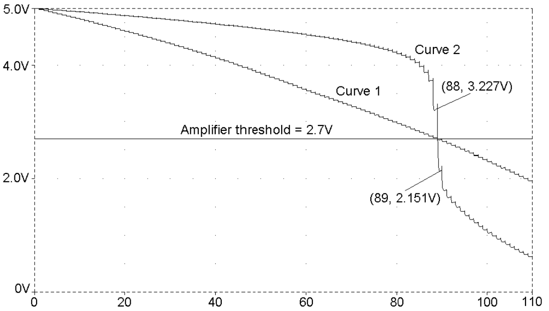


Fig. 9 Comparator characteristics: Curve 1 for the old comparator; Curve 2 for the new one.

new comparator has a much higher steepness in the threshold zone; the voltage leap at the threshold point is about 1V.

2.3 LTE Circuit and the Method of Teaching

Circuits used to create control voltages that determine the weights of input variables of LTE depend slightly on the way synapses are implemented. Some of these circuits were published (for example, in [14]) and they have similar structures. Their main difference lie in the type of memory element they use (a capacitor or a transistor with floating gate) and with the way the values of input binary variables ({0,1} or {-1,+1}) are represented.

Fig.10 shows the complete LTE circuit used in the experiments. Every its synapse contains five 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 order of the transistor connection makes the synapse current dependency on the control voltage more linearly, reducing by several times the influence of the input variable change on the control voltage.

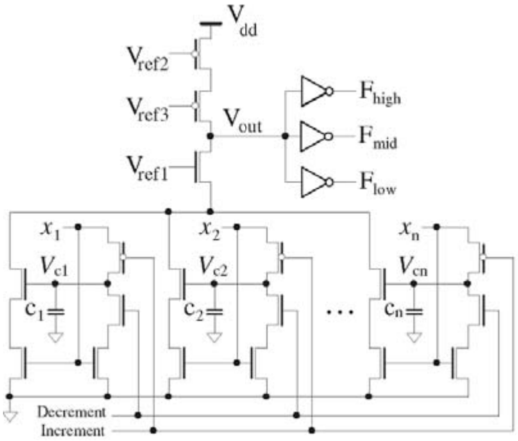


Fig. 10 The LTE circuit.

During teaching, the voltage that controls the synapse current (i.e., the 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 "1". The capacitor charge increase or decrease is realized by approximately the same quanta that determine a learning step. The learning step is appointed based on the accuracy required to set the control voltages. Its value can be controlled by choosing the amplitude and duration of "increment" and "decrement" signals.

When teaching LTE fairly complicated threshold functions (with a large value of sum of weights and thresholds), the learning step should be small. LTE teaching algorithms are usually 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, by a very small value.

To increase the margin of reliability after the teaching, the LTE circuit has three output amplifiers with different sensitivity thresholds: high, middle and low. 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 refreshing capacity memory. The three output amplifiers with different thresholds allow one to solve this problem, for example, by auto-correcting the control voltages using the output signal F_{mid} as a learning sequence of the 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.

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. The teaching/refresh switch passes to

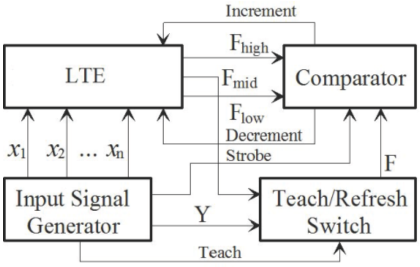


Fig. 11 General scheme for experiments.

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

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

when teaching;

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

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 in selecting threshold functions for teaching, which 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 have:

– a short sequence of variable combinations checking all possible switches of the function value,

– a wide range of variable weights, and

– a high threshold value for a given number of variables.

This problem will be investigated in greater detail in the final section of the paper. It will be shown that a function that 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 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 for given threshold functions are described 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 to the LTE state after the LTE is taught. The threshold hysteresis can be obtained using three output amplifiers whose characteristics have different thresholds as shown in Fig.12.

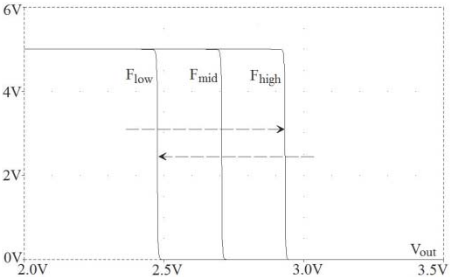


Fig. 12 Static characteristics of the output amplifiers.

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, Horner's function of seven variables was chosen:

$$\begin{aligned} &Y_7 = x_7(x_6 \vee x_5(x_4 \vee x_3(x_2 \vee x_1))) = \\ &x_7x_6 \vee x_7x_5x_4 \vee x_7x_5x_3x_2 \vee x_7x_5x_3x_1; \\ &\overline{Y}_7 = \overline{x}_7 \vee \overline{x}_6(\overline{x}_5 \vee \overline{x}_4(\overline{x}_3 \vee \overline{x}_2\overline{x}_1)) = \\ &\overline{x}_7 \vee \overline{x}_6\overline{x}_5 \vee \overline{x}_6\overline{x}_4\overline{x}_3 \vee \overline{x}_6\overline{x}_4\overline{x}_2\overline{x}_1. \end{aligned} \tag{8}$$

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

$$\begin{aligned} Y_7 = Sign(&x_1 + x_2 + 2x_3 + 3x_4 + 5x_5 \\ &+ 8x_6 + 13x_7 - 21) \end{aligned} \tag{9}$$

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

The checking sequence for this function contains eight combinations. Their order is chosen such that the sequence of the corresponding function values alternates. 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 obtain illustrative and easily explainable results from 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 seven variables with various hysteresis widths are given in Fig. 13.

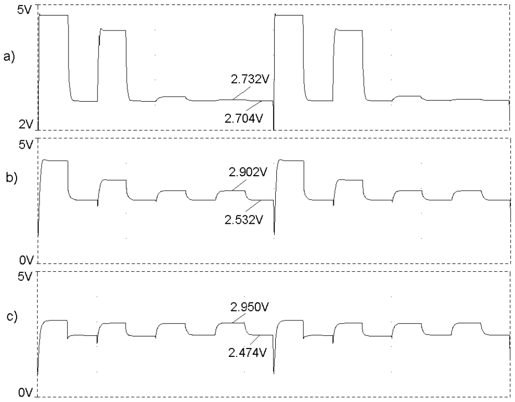


Fig. 13 Output signal of the LTE β-comparator learned to the function of 7 variables:
(a) without hysteresis,
(b) with hysteresis of 340mV, and
(c) with hysteresis of 450mV.

When there was no hysteresis (Fig.13a), the LTE learned 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 poor 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 decreased considerably ($\min\Delta V_{0ut} = 0.37\text{V}$; $\max\Delta V_{out} = 0.9\text{V}$) 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 learned a function representation that was close to the optimum. All the voltage leaps at the output of the β-comparator were approximately the same ($\min\Delta V_{out} = 0.48V$; $\max\Delta V_{out} = 0.55\text{V}$).

Using simulation, we checked the possibility of providing auto-support of the LTE in the learned state based on of using threshold hysteresis. With this aim, the leakage resistances of control voltage capacitors were explicitly incorporated into the LTE circuit. The LTE was taught the Horner's function of seven 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.

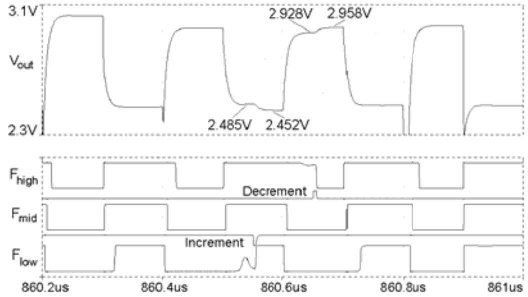


Fig. 14 Correction of the control voltages in the refreshment mode of LTE operation.

As seen from the figure, the 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 experiments, the LTE was taught to a threshold function of 10 variables:

$$Y_{10} = x_{10}(x_9 \vee x_8(x_7 \vee x_6(x_5 \vee x_4(x_3 \vee x_2 x_1)))) = x_{10}x_9 \vee x_{10}x_8x_7 \vee x_{10}x_8x_6x_5 \vee x_{10}x_8x_6x_4x_3 \vee x_{10}x_8x_6x_4x_2x_1;$$
$$\overline{Y}_{10} = \overline{x}_{10} \vee \overline{x}_9(\overline{x}_8 \vee \overline{x}_7(\overline{x}_6 \vee \overline{x}_5(\overline{x}_4 \vee \overline{x}_3(\overline{x}_2 \vee \overline{x}_1)))) = \overline{x}_{10} \vee \overline{x}_9\overline{x}_8 \vee \overline{x}_9\overline{x}_7\overline{x}_6 \vee \overline{x}_9\overline{x}_7\overline{x}_5\overline{x}_4 \vee \overline{x}_9\overline{x}_7\overline{x}_5\overline{x}_3\overline{x}_2 \vee \overline{x}_9\overline{x}_7\overline{x}_5\overline{x}_3\overline{x}_1. \quad (10)$$

This function can be represented in the form

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

The checking sequence for the function must contain not less than 11 combinations that are defined by the terms of Y_{10} and $\overline{Y}_{10}$ in (10). To make the teaching sequence of function values interchangable it is necessary to have an 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 shown in Fig.15. The final 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 $\max\min \Delta V_{out} \approx 1\text{V}$ (Fig.9); the hysteresis width was 0.85V, and the learning step was adaptive.

The learning step is defined by the amplitude and duration of the "increment" and "decrement" signals. These signals ensure charging and discharging the capacitors with the current of 0.15uA and can have a maximum duration equal to the duration of the strobe-signal *t* (90ns). It gives for the capacitor of 1pF the maximum learning step that is equal to 13.5mV.

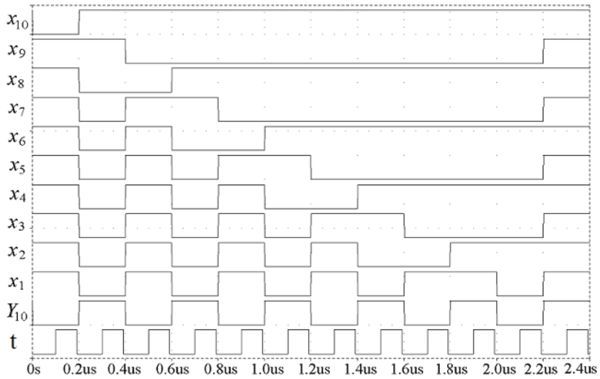


Fig. 15 Single checking sequence of signal value combinations.

The change of the control voltages on the synapse capacitors during the learning is shown in Fig.16.

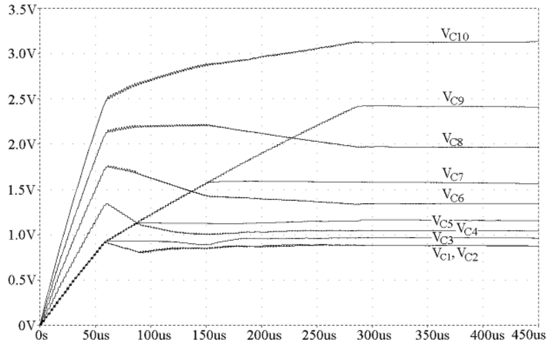


Fig. 16 Changes of the control voltages on the synapse capacitors during the learning.

The dynamics of the learning are easily observable. The control voltages stop changing at the moment 0.28ms. This means that the learning process is over and it is possible to switch from the teaching mode to the refreshing mode. The most accurate moment of mode switching can be defined with a special control signal that 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* do not coincide on at least one combination of the checking sequence, this control signal sets the switcher into teaching mode. This means that the LTE has lost the learned state and must be taught again. The refreshing mode of operation can be interrupted with an evaluation process that calculates the value of the logical function on some input combination. Obviously, refreshing and evaluation have to interchange. During evaluation, to receive correct results the LTE output F_{mid} should be gated.

As is easily 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 time equal to 0.28ms, the teaching mode has been replaced by the refreshment mode. Starting from this moment, only rare signals of "*Increment*" and "*Decrement*" appear 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 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.

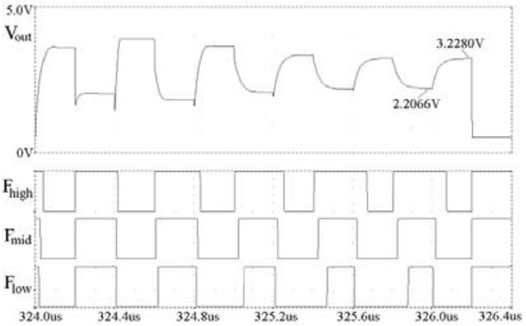


Fig. 17 Picture of the signals on the outputs of the *β*-comparator and the LTE in the refresh mode.

2.5 Implementability Limits of the LTE

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

– to have a short learning sequence,
– to cover a wide range of input weights,
– to have the largest 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 the Fibonacci sequence and the length of the shortest learning (checking) sequence is $n+1$ (the number of combinations of input variable values). Experiments were conducted with three threshold functions for $n=10$, 11 and 12:

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

Since the learning process was not the objective 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 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 for functions of 12 variables is near the edge of implementability because of relatively small values of ΔV_{out}.

Table 1: Results of SPICE simulations.

LTE type	ΔV_{out}	$(\min\div\max)V_{th}$	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 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 than 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 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 should the voltages be maintained for normal functioning of the LTE after learning? First, 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. 18. 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 in this way.

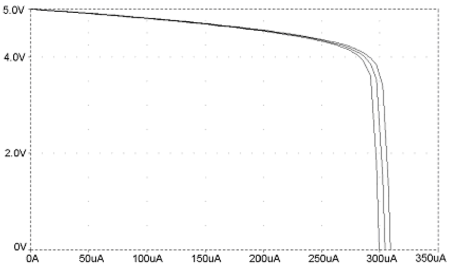


Fig. 18 Behaviour of the dependency $V_{out}(I_p)$ when the voltage V_{dd} changes in the interval ±0.1%.

On the other hand, taking into account the method of producing reference voltages, 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 in the reverse direction 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 either more or less than the output amplifier threshold by a value of not less than 50mV.

The control voltages of the synapses were set up with an 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 drifts by $I_{\min}/2$ on either 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}	$\pm 0.42I_{\min}$	±5.3% (±46mV)	±0.60% (±17mV)
F_{11}	$\pm 0.40I_{\min}$	±4.7% (±40mV)	±0.73% (±27mV)
F_{12}	$\pm 0.34I_{\min}$	±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 based on Table 2 data: since all the control voltages of synapses in the LTE should be maintained simultaneously, their maintenance should be as accurate as units of millivolts.

3. LTE Learnable for 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 also 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 behaviour 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:

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

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 way of solving this problem is by 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. On the contrary, incorporating extra inverse inputs increases the number of realizable threshold functions of *n* variables by 2*n* times.

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

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

where $\omega_j = w_j / T$. It is easy to see from (14) that the use of negative weights can be reduced to inverting the variables and vice versa. The 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.

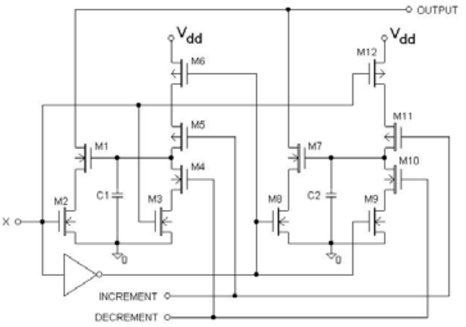


Fig. 19 Synapse forming positive and negative weights of the input variable.

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

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

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

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

(to maintain 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 the form:

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

It follows from the above that when teaching an LTE with such synapses it is desirable to change the input weights (a_j, b_j) in such a way that one of the weights in each pair goes to zero. Moreover, 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 at 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 that is determined by $\min(a_j, b_j)$. 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

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

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

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

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 the existence of stable decisions, which the LTE is able to keep realizing non-isotonous threshold functions, has been proved. Unfortunately, the authors could not find on-chip learning algorithms leading to these decisions. One possible solution is proposed in the next subsection. This solution provides an on-chip learning process, which gives convergence independent of initial conditions and uses a modified synapse circuit.

3.2 The Problem Decision

The same general structure scheme that is shown in Fig.11, is used when simulating the process of teaching the LTE an arbitrary given logical threshold 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 is represented in Fig.15. Non-isotonous functions can be derived from this function by inverting some variables with the help of inverters. The generator is supposed to be implemented separately from the LTE. Other blocks of the general scheme are assumed to be implemented on-chip. In the schematics (shown below), the widths of all transistors are pointed out to make the experiments replicable.

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 is realized in Fig.20.

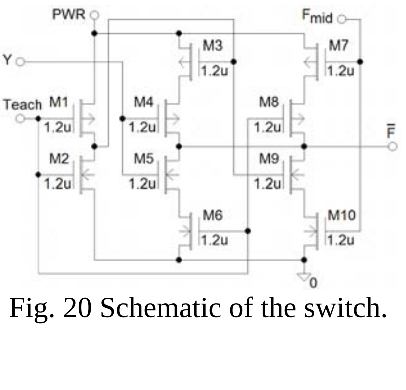


Fig. 20 Schematic of the switch.

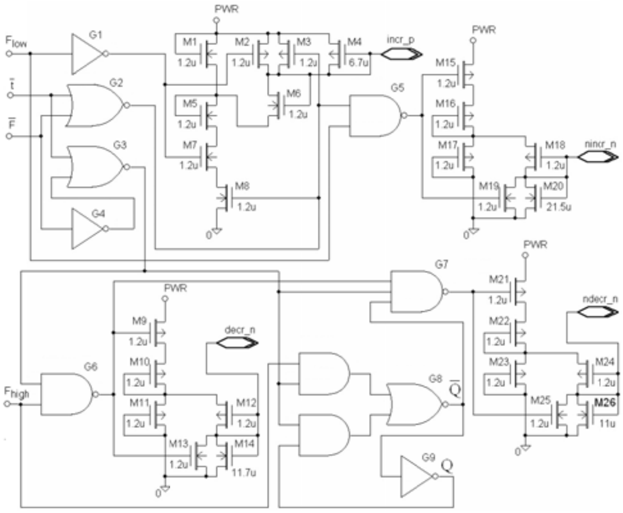


Fig. 21 Schematic of the "Comparator".

The "Comparator" produces "Decrement" and "Increment" signals. 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 its implementation description is

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

where $\overline{F}$ is the switch output and $\bar{t}$ is the strobe-signal inversion. Passive value of this signal is "Log.1" and active value is "Log.0". The function is realized on the gates G$_1$, G$_2$, and transistors M$_1$ – M$_8$. The output stage of the function implementation is a NAND gate (transistors M$_2$, M$_3$, M$_7$, and M$_8$) combined with an embedded voltage divider (transistors M$_1$– M$_5$) and a current mirror on the transistors M$_4$ and M$_5$, which restricts the "Increment" current through p-channel transistors controlled by the signal $incr\_p$. The width of the transistor M$_4$ provides the 0.15uA current through a p-channel transistor of the minimum width (1.2u). Using the voltage divider allowed to reduce the width of the transistor M$_4$ more than twenty times.

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

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

that is implemented on the gates G$_3$, G$_4$, G$_6$, and transistors M$_9$ – M$_{14}$. Active value of this signal is "Log.1". The last stage of the function implementation is an inverter (transistors M$_9$, M$_{13}$) that contains the embedded voltage divider (transistors M$_{10}$, M$_{11}$) and the current mirror on the transistors M12, M$_{14}$, which restricts the amplitude of the signal $decr\_n$. The transistor M$_{14}$ of 11.7u width provides the current 0.15uA through *n*-channel transistors of minimum width (1.2u) controlled by the signal $decr\_n$.

Two additional "Decrement" signals ($nincr\_n$ and $ndecr\_n$) are used when teaching the LTE synapses input weight signs. Each of them creates an additional force that pulls down voltages of the synapse capacitors corresponding to $\min(a_j, b_j)$ up to the ground potential during LTE learning. The signal $nincr\_n$ is alternative to the signal $incr\_p$ and has the logical function

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

that is implemented on gates G$_2$, G$_5$, and transistors M$_{15}$ – M$_{20}$. The last stage of the implementation is analogous to the last stage of the signal $decr\_n$. The transistor M$_{20}$ of 21.5u width provides current 81.6nA through *n*-channel transistors of minimum width (1.2u) controlled by the signal $nincr\_n$.

The signal $ndecr\_n$ is alternative to the signal $decr\_n$ and has one additional restriction: if during the strobe-signal $t =$"Log.1" the signal $decr\_n =$ "Log.1" is finished, the signal $ndecr\_n =$ "Log.1" cannot be produced. The function of the signal $ndecr\_n$ is defined as

$$ndecr\_n = \overline{F}_{high}\cdot\overline{F}\cdot t\cdot\overline{\mathrm{Q}}$$

where $\overline{\mathrm{Q}}$ is output of the latch keeping the value of the signal $decr\_n$ up to the end of the strobe signal. In Fig.21 the latch Q is constructed from gates G$_8$, G$_9$ and has excitation functions

$$S = F_{high}\cdot\overline{\bar{t}\vee\overline{\overline{F}}};\quad \overline{R} = \overline{\bar{t}\vee\overline{\overline{F}}}.$$

The signal $ndecr\_n$ is implemented as

$$ndecr\_n = \overline{\overline{\overline{\bar{t}\vee\overline{\overline{F}}}\cdot\overline{\overline{\overline{\bar{t}\vee\overline{\overline{F}}}\cdot F_{high}}}\cdot\overline{\mathrm{Q}}}}$$

on gates G$_3$, G$_4$, G$_6$, G$_7$, and transistors M$_{21}$ – M$_{26}$. The last stage of the implementation is the same as in the realisation of the signal $decr\_n$. The transistor M$_{26}$ of the current mirror with 11u width provides current 0.16uA through *n*-channel transistors of minimum width (1.2u) controlled by the signal $ndecr\_n$.

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

In this figure, 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 a 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.

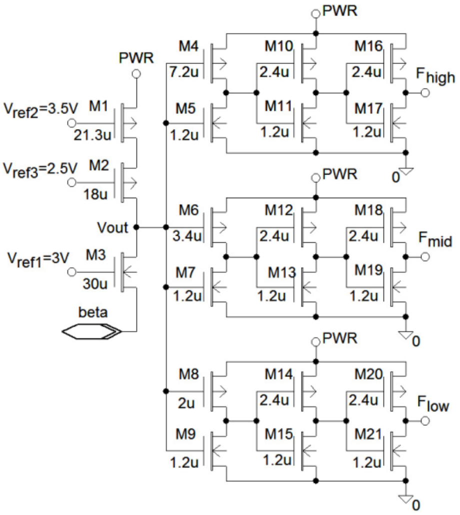


Fig.22 Schematic of the β-comparator with output amplifiers.

The full synapse circuit of the LTE learnable for arbitrary threshold functions is introduced in Fig.23. It is constructed on the basis of the synapse circuit in Fig.19. Voltages V_{C1} and V_{C2} on the capacitors control the synapse currents flowing through pairs of transistors (M$_5$, M$_{21}$) or (M$_6$, M$_{22}$). The circuit has two input logical variables x and $\bar{x}$. The variable $\bar{x}$ can be derived with the help of an inverter. The voltages 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 $decr\_n$ increase and decrease the capacitor voltages through pairs of transistors (M$_1$, M$_3$), (M$_2$, M$_4$) and (M$_7$, M$_{23}$), (M$_8$, M$_{24}$) respectively depending on the value of input variables (x, $\bar{x}$).

Two pseudo *n*-MOS inverters on transistor pairs (M$_{29}$, M$_{39}$) and (M$_{30}$, M$_{40}$) are sensitive elements of capacitor voltages close to V_{th}. Voltage V_{ref4} = 3.9V fixes the conductivity of their *p*-channel transistors. Output signals G$_1$ and G$_2$ of these elements control the conductivity of two pairs of transistors (M$_{15}$, M$_{17}$) and (M$_{16}$, M$_{18}$) respectively. Transistors of each pare can pass currents only when the voltage on the corresponding capacitor (V_{C1} or V_{C2}) exceeds 675mV. Two inverters (M$_{31}$, M$_{41}$ and M$_{32}$, M$_{42}$) with outputs G$_3$ and G$_4$ invert the signals G$_1$ and G$_2$. Signals G$_3$ and G$_4$ control

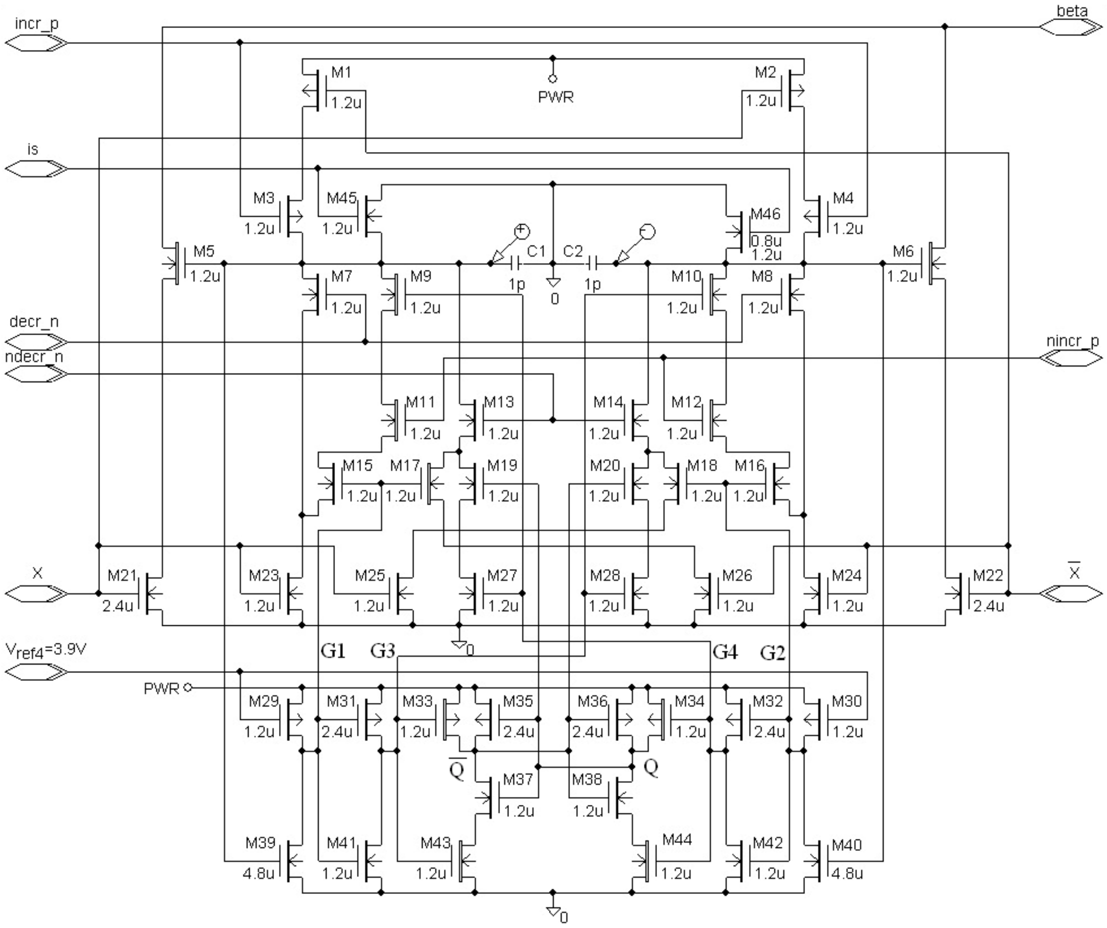


Fig. 23 The LTE Synapse implementation.

transistors M_9 and M_{10}, each of which opens when the voltage of the corresponding capacitor (V_{C1} or V_{C2}) exceeds 635mV. Difference between these threshold voltages (675–635=40mV) is very important because it provide correct learning behaviour of synaptic weights in the region close to the threshold of n-channel transistors.

Signals G_3 and G_4 also write information about the sign of the synaptic weight in the latch (outputs Q and $\overline{\mathrm{Q}}$) on transistors M_{33} – M_{38}, M_{43}, M_{44}. 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.

Signals G_1 – G_4, $ndecr\_n$, $nincr\_n$, the latch Q, and input variables ($x, \bar{x}$) control the conductivity of additional decrement chains for each capacitor. For the capacitor C1, these chains are described by the expression

$$\mathrm{M}_9\mathrm{M}_{11}\mathrm{M}_{15}\mathrm{M}_{23} \vee \mathrm{M}_{13}(\mathrm{M}_{19}\mathrm{M}_{27} \vee \mathrm{M}_{17}\mathrm{M}_{26}).$$

For the capacitor C_2 the expression for chains is

$$\mathrm{M}_{10}\mathrm{M}_{12}\mathrm{M}_{16}\mathrm{M}_{24} \vee \mathrm{M}_{14}(\mathrm{M}_{20}\mathrm{M}_{28} \vee \mathrm{M}_{18}\mathrm{M}_{25}).$$

It should be noted that transistors M_{27}, M_{28} cannot be replaced by transistors M_{23}, M_{24} because of appearance of parasitic dependency between voltages V_{C1} and V_{C2} through parasitic capacitances of these transistors.

The initial setting signal is controls the transistors M_{45} and M_{46}, which serve only for the initial setting of the voltages on the capacitors C_1 and C_2.

During learning, the synapse works in the following 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 the input variable weight is negative, i.e., it coincides with the state of the latch. In this case the signals $nincr\_n$ and $ndecr\_n$ pull together the voltage V_{C1} to 0V by the chain (M_9, M_{11}, M_{15}, M_{23}) and by the chain (M_{13}, M_{17}, M_{26}) 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 over decrement steps caused by the signals $decr\_n$, $ndecr\_n$, $nicr\_n$ and the voltage V_{C1} will grow. As soon, as V_{C1} exceeds V_{th} the sensitive inverter (M$_{29}$, M$_{39}$) closes the transistors M$_{15}$, M$_{17}$ halting the action of the signals $ndecr\_n$, $nicr\_n$ by the additional chains. This inverter also switches the inverter (M$_{31}$, M$_{41}$), which, in its turn, opens the transistors M$_{10}$ and M$_{28}$ enabling action of the signals $ndecr\_n$ by the additional chain. After that, the voltage V_{C1} continues to rise to the weight value and the signals $decr\_n$, $ndecr\_n$ will pull down the voltage V_{C2} by two chains: (M$_8$, M$_{24}$), and (M$_{14}$, M$_{20}$, M$_{28}$).

As soon as the voltage V_{C2} reaches 675mV, the sensitive inverter (M$_{30}$, M$_{40}$) opens the transistors M$_{16}$, M$_{18}$ and, when $V_{C2} = 635mV$, switches the inverter (M$_{32}$, M$_{42}$), which, in its turn, closes the transistors M$_9$, M$_{27}$ and switches the latch Q into the state Q = Log.1 (positive weight sign). Output signals of the latch close the transistor M$_{20}$ and open the transistor M$_{19}$. The difference in critical values of the voltage V_{C2}, which leads to switching the sensitive element (M$_{30}$, M$_{40}$) and the inverter (M$_{32}$, M$_{42}$), is very important because in this case switching of the latch only slightly changes the condition of the capacitor C$_2$ discharging and the voltage V_{C2} continues decreasing down to ground potential due to opening the additional chain (M$_{14}$, M$_{18}$, M$_{25}$) that partly compensates closing the chain (M$_{14}$, M$_{20}$, M$_{28}$).

3.3 Results of SPICE Simulation

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

$$\begin{aligned} Y_{10} &= x_{10}\bar{x}_9 \vee x_{10}x_8\bar{x}_7 \vee x_{10}x_8x_6\bar{x}_5 \vee \\ &\quad x_{10}x_8x_6x_4\bar{x}_3 \vee x_{10}x_8x_6x_4x_2\bar{x}_1; \\ \bar{Y}_{10} &= \bar{x}_{10} \vee x_9\bar{x}_8 \vee x_9x_7\bar{x}_6 \vee x_9x_7x_5\bar{x}_4 \vee \\ &\quad x_9x_7x_5x_3\bar{x}_2 \vee x_9x_7x_5x_3x_1. \end{aligned} \tag{20}$$

Another form of the function (20) representation is

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

The checking sequence for these two functions is 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 for 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.

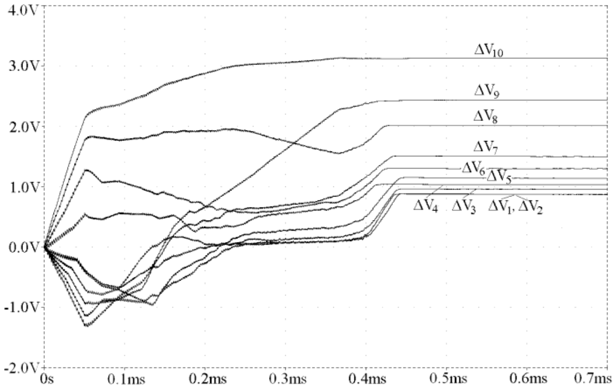


Fig. 24 The LTE learning of the function (10).

It is easy to see from Fig.24 that all curves reach stable states for the time equal to 0.44ms or for 183 learning cycles and the weights of all variables are positive.

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

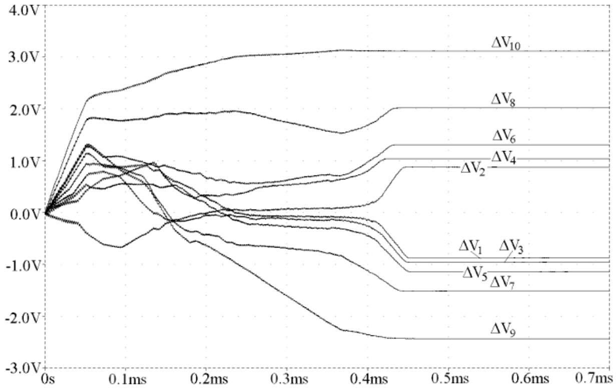


Fig. 25 The LTE learning of the function (20).

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 even variables have positive weights and all weights of odd variables are negative.

In Fig.26 the process of LTE learning of 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.

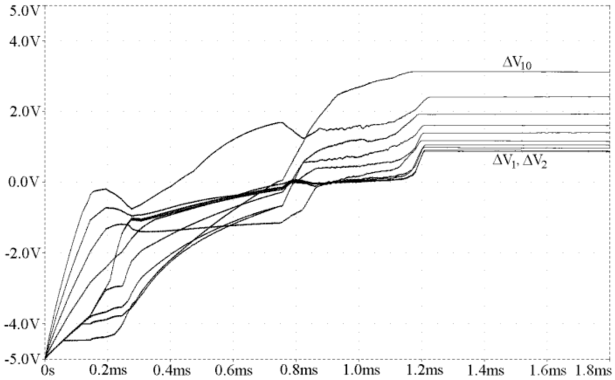


Fig.26 The LTE learning of the function (10) from the initial state of the least negative weights.

The picture shows the correct result of the learning. The decision is found for 1.2ms (500 cycles).

Fig.27 illustrates the process of LTE learning of the function (20) from the initial state ($V_{C1} = 0\text{V}$, $V_{C2} = 5\text{V}$) for all synapses.

Analyzing Fig.25, it is possible to conclude, that this learning process is finished for 1.45ms (604 cycles). Signs of variable weights and their values are determined correctly: all odd variables have negative weights and all weights of even variables are positive.

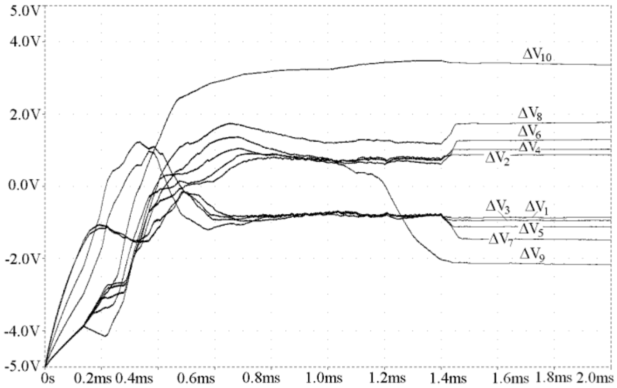


Fig.27 The LTE learning of the function (20) from the initial state of the least negative weights.

For the proposed procedure of on-chip LTE learning for arbitrary threshold functions of some number of variables, it is not clear how to prove its convergence analytically. Sufficient SPICE-simulation experiments have been done to establish that this learning procedure possesses of convergence from initial states ($V_{C1} = 0\text{V}$, $V_{C2} = 0\text{V}$), ($V_{C1} = 0\text{V}$, $V_{C2} = 5\text{V}$), ($V_{C1} = 5\text{V}$, $V_{C2} = 0\text{V}$),($V_{C1} = 5\text{V}$, $V_{C2} = 5\text{V}$), which may be different for different synapses. To provide the procedure convergence from these initial states it is necessary to chose very accurate current amplitudes of the signals $ndecr\_n$ and $nicr\_n$.

Nevertheless it is impossible to be sure that there is no an initial state, from which does not exist the algorithm convergence. By this reason it does not possible to confirm that, if the LTE is taught to realize some threshold function, using its state as initial, it can be repeatedly taught any other function. May be it is so, but this fact did not proved.

In any case, if accept the restriction that any learning process is started from the reasonable initial state ($V_{C1} = 0\text{V}$, $V_{C2} = 0\text{V}$), which is the same for all synapses, the accuracy diapason of setting for the signals $ndecr\_n$, $nicr\_n$ become much wider.

By this reason, it can be recommended after losing by the LTE of a learned state to do the initial setting of the LTE before new act of learning.

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 difficult 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 a certain type?

– Are these parameters attainable during 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 merely a sophisticated analog-digital device, whose maximum functional power is attained, in many cases, by using the effects of the second and even the third order transistor behaviour. This, in its turn, requires using models of higher levels (e.g. BSIM3v3.1). Therefore, the dependencies of neuron behaviour 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 under consideration. This, in turn, requires that the neuron behaviour 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., from different manufacturers) should be conducted. Thus, to get an answer to the question about the maximum functional power of the artificial neuron, serious experimental work is needed, the volume of which obviously depends linearly on the number of variable value combinations used in every experiment.

The existence of a number of neuron circuit parameters, which provide reproducing a threshold function of limiting complexity, does not mean 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 the corresponding input weight (or threshold), then $\Delta V = (V_{\max} - V_{\min}) / w_{\max}$ is the value determining the required precision of teaching. Taking into account the possible non-linearity of the dependence $w(V)$ and necessity of compensating during teaching technological variations of transistor parameters (geometrical sizes, thresholds, etc.), the 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 at the necessity of computer simulation as the basic tool of this research.

In order to provide the 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 or more times, regardless of selecting the teaching strategy. Because of this, SPICE simulation of the teaching process takes hours. Naturally, the duration of the simulation process linearly depends on the length of the learning sequence.

In order to obtain reasonable simulation time for highly complicated artificial neurons, the test tasks should be 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 ranges as tightly as possible. Functions of this type are the subject of the section. Some results of threshold logic that have been known for several tens of years, 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 a positive distance from the separating hyperplane (*T* set) and the value "Log.0" on those whose distance from 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:

$$\text{finding } \min\left(\sum_{j=1}^{n} w_j + \eta\right) \text{ at the conditions}$$

$$\begin{cases} \sum\limits_{x_j \in T} w_j x_j - \eta \ge 0 \\ \sum\limits_{x_j \in F} w_j x_j - \eta < 0 \end{cases}.$$

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

Without loss of generality, threshold functions with $w_j > 0$ will be only 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 $f(X_j) \ge f(X_i)$ 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 a 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 is equal 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 $Y = Sign(\sum_{j=1}^{n} x_j - n)$ up to $2C_n^{n/2} = \dfrac{2 \cdot n!}{(n/2)!(n/2)!}$ for the function $Y = Sign(\sum_{j=1}^{n} x_j - n/2)$ 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 *ν*-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 efficiency criteria 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 length of the learning sequence in the number of bearing combinations. The lower the values of these criteria, the more efficient the function will be in teaching.

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

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

$$Y_2 = Sign\left((n-1)x_n + \sum_{j=1}^{n-1} x_j - n\right);$$

$$Y_2 = x_n \cdot \bigvee_{\mathrm{j=1}}^{n-1} x_j;\ \overline{\mathrm{Y}}_2 = \bar{x}_n \vee \bigwedge_{j=1}^{n-1} \bar{x}_j\,.$$

Both functions have the same number of 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 preferable as a test function; the question arises of whether it is possible to derive test functions with the highest efficiency for 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 with respect to inverted variables, and vice versa, i.e., inverted functions of the second type are functions of the first type with respect to inverted variables.

[4] This result has been known in the threshold logics since the late 50's or early 60's, but it is difficult to give specific reference.

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

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,

$$N[T_{01}(n)] = N[F_{01}(n-1)] + 1,$$
$$N[F_{01}(n)] = N[T_{01}(n-1)] \text{ and}$$
$$L(n) = L(n-1) + 1 = n + 1.$$

Hence, a Horner's function of n variables has the shortest bearing learning sequence.

Statement 1: The first and second types 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. $H_1(n) = H_2^d(n)$.

For any threshold function

$$f(X) = Sign(\textstyle\sum_{j=1}^{n} w_j x_j - \eta),$$

the following is true:

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

and

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

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 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 fore variables there is a threshold function

$$\begin{aligned} &x_1 x_2 x_3 \vee (x_2 \vee x_3) x_4 = \\ &Sign(x_1 + 2x_2 + 2x_3 + 3x_4 - 5) \end{aligned} \qquad (22)$$

with the sum of input weights greater than that of Horner's functions. From the function dual to (22), by deleting inversions of variables and multiplying it by x_5 the next function is derived

$$\begin{aligned} &(x_2 x_3 \vee (x_1 \vee x_2 \vee x_3) x_4) x_5 = \\ &Sign(x_1 + 2x_2 + 2x_3 + 3x_4 + 5x_5 - 9). \end{aligned} \qquad (23)$$

This function has a threshold greater than that of the second type Horner's function of five variables. However, both functions (22) and (23) have bearing learning sequences of the length equal to n+3. Note that for values within practical interest 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 $\{x_1, x_2, ..., x_n\}$ correspond to decimal equivalents of binary numbers $\sum_{j=1}^{2} x_j 2^{j-1}$.

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

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 simple for hardware implementation in CMOS technology. Its β-comparator has very high sensitivity to current changes; this makes it possible to obtain 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 the 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. The experiments confirm this result for functions of 10 variables. Moreover, during LTE learning all dispersions of technological and functional parameters of the LTE circuit are compensated.

For enhancement of functional abilities a new circuit of the LTE synapse has been proposed, which gives to LTE the opportunity to have both excitatory and inhibitory inputs. The LTE with such synapses can be taught 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. An on-chip learning procedure, which allows the LTE to learn arbitrary threshold functions of 10 or less variables, has also been 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 $2n$ 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 vice versa.

The ability to determine the type of logical variable inputs during the learning increases the number of realizable functions by 2^n times compared with the isotonous LTE implementation (n is the number of variables).

We believe that the proposed LTE and its learning procedure can be very useful in many important applications including development of actual artificial neurons. The functional power of neurochips depends not only on the number of neurons that 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, particularly in the case of a neuron implementation as a digital/analog circuit.

The main drawback of the proposed LTE is the high stability requirements for the supply voltage. This drawback appears 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 periods of not less than tens milliseconds) will be compensated during learning and refreshing. Its rapid changes can cause the loss of learned information. It would be reasonable to study the possibility of compensating the operational LTE parameters using circuit facilities.

Somebody can note that PSPICE simulations was oriented to very old CMOS technology of 0.8μm. Frankly speaking it does not matter what technology to use. We used old technology because of the problems stated in the paper have been studding during 15 years. There is no doubt that all simulations can be repeated for new technologies oriented on digital/analog implementations. Unfortunately submicron technologies are not appropriate because of big values of leakage currents.

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.