Manuscript received Nov. 12, 2007; revised Feb. 18, 2007

Multiple-valued Logic Approach to Fuzzy Controllers Implementation

V. VARSHAVSKY, V. MARAKHOVSKY[1], I. LEVIN[2], H. SAITO[3]
[1,3] The University of Aizu
Aizu-Wakamatsu City, Fukushima Prefecture, 965-8580
JAPAN
vbmarak@gmail.com, hiroshis@u-aizu.ac.jp
[2] Tel Aviv University
Ramat Aviv, Tel Aviv 69978
ISRAEL
ilia1@post.tau.ac.il

Abstract: - This paper offers a new technique for designing fuzzy controllers as analog hardware devices on bases of CMOS implementation of multi-valued logical functions. This approach is based on using a summing amplifier with saturation as a building block that can be considered as a multi-threshold logical element. The functional completeness in an arbitrary-valued logic of a summing amplifier with saturation is proven. This fact gives a theoretical background for an analog implementation of fuzzy devices. In contrast with the traditional software approach to fuzzy controller implementations based on explicit fuzzification, fuzzy inference, and defuzzification procedures, hardware implementations of fuzzy controllers as analog devices have advantages of higher speed, lower power consumption, smaller die area and more. Universal and proper design methods for such type of hardware are offered. The paper illustrates design examples for real industrial fuzzy controllers and provides SPICE simulation results of their functioning.

Key-Words: - Fuzzy Logic, Fuzzy Controller, Fuzzy rules, Fuzzy inference, Fuzzification and Defuzzification, Multi-Valued Logic, Multi-threshold Element, Functional Completeness, Summing Amplifier.

1 Introduction

Fuzzy logic control is a methodology bridging *artificial intelligence* and traditional *control theory*. This methodology is usually applied in the only cases when accuracy is not of high necessity or importance. On the other hand, as it is stated in [1], "Fuzzy Logic can address complex control problems, such as robotic arm movement, chemical or manufacturing process control, antiskids braking systems or automobile transmission control with more precision and accuracy, in many cases, than traditional control techniques … . Fuzzy Logic is a methodology for expressing operational laws of a system in linguistic terms instead of mathematical equations."

Wide spread of the fuzzy control and high effectiveness of its applications in a great extend is determined by formalization opportunities of necessary behavior of a controller as a "fuzzy" (flexible) representation. This representation usually is formulated in the form of logical (fuzzy) rules under linguistic variables of a type "If A then B".

The Fuzzy Logic methodology [2, 3] comprises three phases:

1. The *fuzzification* is a transformation of analog (continuous) input variables to linguistic ones, e.g., transformation of temperature into the terms *cool*, *warm*, *hot* or transformation of speed into the terms *negative big* (*NB*), *negative small* (*NS*), *zero* (*Z*)", *positive small* (*PS*), *positive big* (*PB*). Such transformation is realized by introduction of so-called *membership functions*, which define both a range of value and a degree of membership. For linguistic variables it is important not only which membership function a variable belongs to, but also a relative degree (weight) to which it is a member. A variable can have a weighted membership in several membership functions at the same time.
2. The *fuzzy inference* maps input linguistic variables onto output linguistic variables on the base a system of fuzzy rules of the type "IF A THEN B" For instance: "IF the temperature is *worm* THEN the speed is *Positive Small* (*PS*)" or "IF the speed is *Negative Big* (*NB*) THEN force is *ZERO*". Since input linguistic variables are weighted, the output linguistic variables can be obtained weighted as well. Traditional fuzzy logic approach comprises Mamdani-type and Sugeno-type inference methods.

Manuscript received Nov. 12, 2007; revised Feb. 18, 2007

The Mamdani-type method is more intuitive and assumes the output variables as a fuzzy set. Fuzzy rules in it contain a *fuzzy precondition* part (after IF) and a *fuzzy consequence* part (after THEN). The Sugeno-type method expects the output variables to be singletons or dealing with consequents that are equations. So it is better suited for mathematical analysis, nonlinear system modeling and interpolation.

3. In the *defuzzification* phase, the weighted values of output linguistic variables obtained as a result of fuzzy inference have to be transformed to analogue (continuous) variables. This procedure is also based on membership functions. Two major methods are used for defuzzification:

– The *maximum* defuzzification method, wherein an output value is determined by the linguistic variable with the maximum weight;
– The *centroid* calculation defuzzification method, wherein an output value is determined by the weighted influence of all the active output membership functions.

As a rule, or at least in a great part of applications, a fuzzy controller is a transformer of input analog signals into an analog output signal. A linguistic variable is a *subjective* characteristic of an input analog variable, values of which are transformed on bases of given membership functions into a set of weighted values of corresponding linguistic variables. This procedure is called a fuzzification and it contains as its composite part the analog-digital transformation.

A set of combinations of weighted linguistic variables corresponds to each value combination of input analog variables. On bases of a system of fuzzy inference rules it is possible to receive the set of weighted output linguistic variables. Using these variables and their membership functions, with help of one of well known defuzzification methods it is possible to form values of the analog output variable. The defuzzification procedure also includes digital-analog transformation.

At present the most wide-spread way of fuzzy logic control implementation is using the programmable fuzzy controllers, which are available on the market together with the means of computer aided programming (e.g. Motorola's 8-bit 68HC11 and 16-bit 68HC12 microcontrollers or specialized fuzzy processors of Siemens 80C517/80C535 families). However, in spite of the implementation evidence and fuzzy controllers' accessibility this approach to controller implementation possesses some disadvantages, e.g. such as high cost and low throughput (that is especially important when fuzzy control in the control contour is used) etc.

This work shows that for a sufficient wide set of applications, fuzzy controllers can be implemented as rather simple CMOS devices, which can be used in embedded systems or as an IP core. What is the basic idea of the proposal?

A fuzzy controller is a deterministic device, for which one and only one value of the output analog variable corresponds to each value combination of the input analog variables. It means that the fuzzy controller should realize an analog function $Y = f(x_1, x_2, ..., x_n)$ [1].

There are two important questions:

1. How to transit from a standard specification of a fuzzy logic function to the specification of corresponding analog function?
2. How to transit from an analog function specification and/or from a standard specification of a fuzzy logic function to corresponding CMOS implementation?

First of all, let us address to membership functions. In most cases [2 – 4], membership functions have a triangle or trapeze form (see Fig.1).

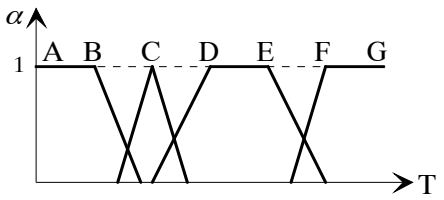

Fig.1 Types of membership functions.

In Fig.1 linguistic points (variables) A and B are *cold,* C is *fresh*, D and E are *worm*, F and G are *hot*. These points determine the connection of the linguistic variables with values of the analog variable T (T is *temperature*). Relatively to these points and similar points for other analog input variables we can compose a table of fuzzy rules connecting combinations of input linguistic variables with output linguistic variables.

On bases of membership functions we can put into accordance to the input and output linguistic variables a set of integer numbers splitting by appropriate way all diapason of changing of corresponding analog variables. Then the table of fuzzy rules will to determine by obvious way the function of multi-valued logic, values of which define the digit representation of the output

[1] It should be noticed that in suppressing majority of publications on fuzzy controllers, this function is given as a response surface and practically without exception this surface has a piecewise linear form.

linguistic variable on chosen value combinations of multi-valued input variables.

In other words, according to our concept, for a broad class of fuzzy controller specifications it is possible to construct corresponding tables connecting input and output membership functions. Frequently membership functions evenly divide the ranges of output variables' variations. If it is not so, the membership functions can be brought to even scale by increasing the number of gradations or, as it will be shown later, by introducing a certain equalization procedure for logical levels. Therefore, specification tables represent nothing but tables determining a specific multi-valued logical function. And what is more, for a number of implementations it is possible to neglect weighting and determining input linguistic variables and simply to use continuous-valued variables.

The above idea was in the focus of our research. We dealt with searching for simple basic multi-valued functions, which, from the one hand, would present a complete functional basis in the multi-valued logic, and from the other hand, could be efficiently implemented by CMOS technology.

2 Hardware Implementation of Fuzzy Controllers

2.1 Summing Amplifier as a Multi-Valued Logical Element

Summing amplifier's behavior, accurate to the members of the infinitesimal order that is determined by the amplifier's gain factor in disconnected condition (Fig.2), is described as follows:

$$V_{out} = \begin{cases} V_{dd} & \text{if } \sum_{j=1}^{n} \frac{R_0}{R_j}(V_j - \frac{V_{dd}}{2}) \le -\frac{V_{dd}}{2} \\ \frac{V_{dd}}{2} - \sum_{j=1}^{n} \frac{R_0}{R_j}(V_j - \frac{V_{dd}}{2}) & \text{in other cases} \\ 0 & \text{if } \frac{V_{dd}}{2} \le \sum_{j=1}^{n} \frac{R_0}{R_j}(V_j - \frac{V_{dd}}{2}) \end{cases} \quad (1)$$

where V_{dd} is the supply voltage, V_j is the voltage on j^{th} input, R_j is the resistance of j^{th} input, R_0 is the feedback resistance, and $V_{dd}/2$ is the midpoint of the supply voltage. Dependence of V_{out} on $\sum_{j=1}^{n} \frac{R_0}{R_j} \cdot (V_j - \frac{V_{dd}}{2})$ is shown in Fig.3 (a).

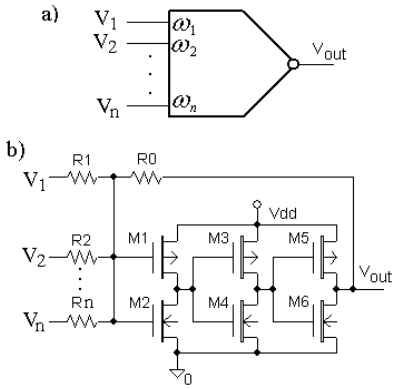

Fig.2 Summing amplifier: a) general designation, b) CMOS implementation using symmetrical invertors.

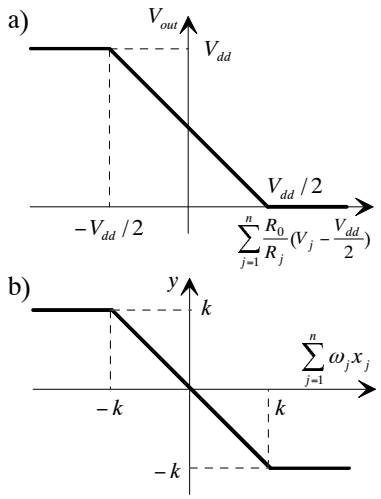

Fig.3 Summing amplifier's behavior: a) within voltage coordinates; b) within multi-valued variable coordinates.

Let us split the source voltage V_{dd} on $m = 2k + 1$ voltage levels. Then replacing the input voltages $V_j - V_{dd}/2$ by m-valued logical variables $x_j = (2 \cdot V_j - V_{dd})k / V_{dd}$ and the output voltage V_{out} by m-valued variable y and designating $R_0 / R_j = \omega_j$ the system (1) can be represented as (2).

$$y(X) = S(\sum_{j=1}^{n} \omega_j \cdot x_j) = \begin{cases} +k & \text{if } \sum_{j=1}^{n} \omega_j \cdot x_j \le -k \\ -\sum_{j=1}^{n} \omega_j \cdot x_j & \text{if } k > \sum_{j=1}^{n} \omega_j \cdot x_j > -k \\ -k & \text{if } \sum_{j=1}^{n} \omega_j \cdot x_j \ge +k \end{cases} \quad (2)$$

Graphical view of (2) is shown in Fig.3 (b).

Later on, we will call the functional element, whose behavior is determined by the system (2), a multi-valued threshold element. When $\omega_j = 1,\ j = 1,2,3$, we will call it a majority element and designate as $maj(x_1, x_2, x_3)$.

2.2 Functional Completeness of the Threshold Element

The basic operation (or a set of basic operations) is called functionally completed in arbitrary-valued logic, if any function of this logic can be represented as superposition of the basic operations.

There are some known functionally complete sets of functions. It is clear, that for proving the functional completeness of a certain new function it is sufficient to show that every function of the known functionally complete set can be represented as a superposition of the considered function. One of functionally complete functions in *m*-valued logic is the Webb's function [8]:

$$w(x, y) = [\max(x, y) + 1]_{\mathrm{mod}\, m}. \qquad (3)$$

Therefore, for proving functional completeness of the threshold operation in multi-valued logic it is sufficient to show how the Webb's function can be represented through this operation [5 – 7].

First, let us represent the function $\max(x_1, x_2)$ by threshold functions. To do this let us consider the function $f_a(x)$, such as

$$f_a(x) = \max(x, a) = \begin{cases} a & \text{if} \quad a \geq x \\ x & \text{if} \quad x > a \end{cases}, \qquad (4)$$
$$|x| \leq k, |a| \leq k.$$

The diagram of this function is shown in Fig.4(a).

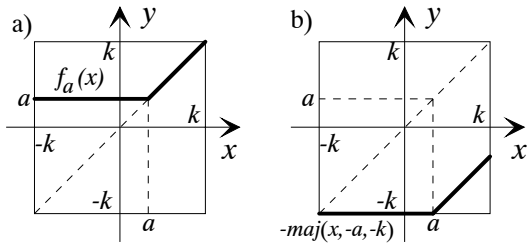

Fig. 4 Diagrams of the functions a) $f_a(x)$ and b) $-maj(x,-a,-k)$.

The $-maj(x,-a,-k)$ function diagram is shown in Fig.4(b). Actually, as far as $x < a$ $x - a - k < -k$ and $-maj(x,-a,-k) = -k$. Note that for all values of x,

$$f_a(x) = -maj(x,-a,-k) + a + k$$

as it follows from Fig.4, hence

$$f_a(x) = -maj(-maj(x,-a,-k), a, k). \qquad (5)$$

Taking into consideration

$$-maj(a,b,c) = maj(-a,-b,-c),$$

it follows from (5) that

$$\max(x_1, x_2) = maj(maj(x_1, -x_2, -k), -x_2, -k). \qquad (6)$$

Now let us consider the representation of the function $y = (x+1)_{\mathrm{mod}\, m}$, $x \geq 0$, $0 \leq y \leq m\text{-}1$ through threshold functions. First of all we designate $m = 2k+1$ and change the beginning of coordinates so that the function will have a form $y = (x + k + 1)_{\mathrm{mod}(2k+1)} - k$, $x \geq -k$, $-k \leq y \leq +k$. To implement this function on threshold elements let us turn to the sequence of pictures in Fig.5.

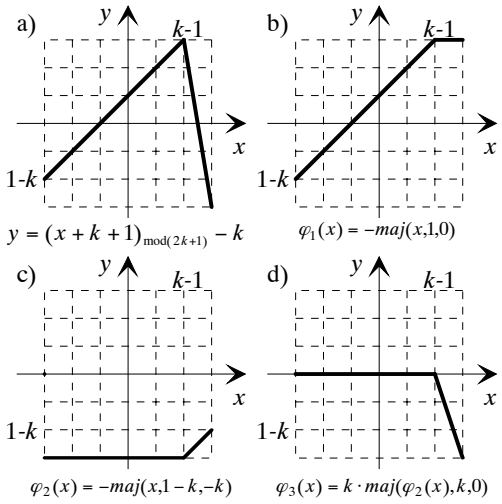

Fig. 5 Implementation of the function $y = (x + k + 1)_{\mathrm{mod}(2k+1)} - k$.

It is easy to see that

$$(x + k + 1)_{\mathrm{mod}(2k+1)} - k = \varphi_1(x) + 2\varphi_3(x)$$

and obviously, this function can also be implemented on threshold elements as

$$y = maj(maj(x,1,0), k \cdot maj(maj(x,1-k,-k),-k,0),$$
$$k \cdot maj(maj(x,1-k,-k),-k,0)).$$

Hence, the functional completeness of the summing amplifier in arbitrary-valued logic is shown. The proof procedure of functional completeness naturally does not give information about methods of effective synthesis. Some methods of a circuit design in the proposed basis will be developed later.

Manuscript received Nov. 12, 2007; revised Feb. 18, 2007

2.3 Fuzzy Devices as Multi-Valued and Analog Circuits

Conventional implementation of fuzzy devices usually has the structure shown in Fig.6. Analog variables $X = \{x_1, x_2, ..., x_n\}$ enter the fuzzy device input. Fuzzifier converts a set of analog variables x_j into sets of weighted linguistic (digital) variables $A = \{a_1, a_2, ..., a_n\}$.

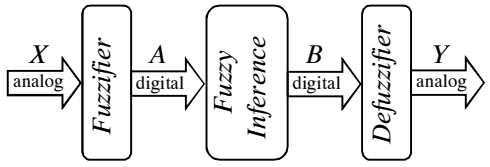

Fig.6 Conventional structure of a fuzzy device implementation.

Fuzzy Inference block generates based on the fuzzy rules a set of weighted linguistic variables values $B = \{b_1, b_2, ..., b_k\}$.

Defuzzifier converts sets of weighted linguistic (digital) variables $B = \{b_1, b_2, ..., b_k\}$ into a set of output analog variables $Y = \{y_1, y_2, ..., y_k\}$.

As a rule, fuzzifier and defuzzifier include AD and DA (analog-digital and digital-analog) converters and are implemented on both levels (hardware and software). Fuzzy inference is usually implemented on the level of microprocessor software.

It is easy to see that each set of values of output analog variables unambiguously corresponds to some set of input analog variable values; hence a fuzzy device could be specified as a functional analog of a signal converter

$$Y(X) = \{y_1(X), y_2(X), ..., y_k(X)\}$$

and its output Y determines a system of n-dimensional surfaces. In cases of sufficient simple membership functions (in known publications such functions are in majority), for fuzzy controller implementations as analog devices it is sufficient to provide a piecewise-linear approximation between a couples of points calculated as adjacent values of a multi-valued logic function.

Let $m = 2k + 1$ linguistic variables a_j ($a_j \in A$) correspond to values of analog variable x_j ($x_j \in X$). Then basing on a system of fuzzy rules, we can specify a system of m-valued logic functions, as follows:

$$B(A) = \{b_1(A), b_2(A), ..., b_k(A)\}. \quad (7)$$

Note that most publications describing fuzzy controllers contain tables specifying fuzzy controllers' behaviour as (7) and a plenty of publications contain piecewise-linear approximations of the corresponding surfaces.

The apparent conclusion can be made from the things mentioned above: if a fuzzy controller is represented as (7), it can be implemented as superposition of multi-valued threshold elements. In this case, owing linear behavior of the threshold element in the zone between the saturation levels ((2) and Fig.3 (b)), natural piecewise linear approximation appears between the discrete points of specification.

In the last subsection of this section some illustrations will be given to show that for a number of real applications the offered approach can provides simple and efficient circuits of controllers.

2.4 Fuzzy Controller Implementations as Circuits from Threshold Elements

2.4.1 Example 1

Let us consider the example, which is taken from [9, pp. 81 – 86]: "Design of a Rule-Based Fuzzy Controller for the Pitch Axis of an Unmanned Research Vehicle".

The fuzzy control rules for the considered device depend on the error value $e = ref - output$ and changing of error $ce = \dfrac{old\ e - new\ e}{sampling\ period}$. Fuzzifier gives seven linguistic variables for each of input analog variables (NB – negative big; NM – negative middle; NS – negative small; ZO – zero; PS – positive small; PM – positive middle; PB – positive big). The output has the same seven gradations. Corresponding 49 fuzzy rules are represented in Table 1.

Table 1: Table of Fuzzy Rules

	e							
ce		NB	NM	NS	ZO	PS	PM	PB
	NB	ZO	PS	PM	PB	PB	PB	PB
	NM	NS	ZO	PS	PM	PB	PB	PB
	NS	NM	NS	ZO	PS	PM	PB	PB
	ZO	NB	NM	NS	ZO	PS	PM	PB
	PS	NB	NB	NM	NS	ZO	PS	PM
	PM	NB	NB	NB	NM	NS	ZO	PS
	PB	NB	NB	NB	NB	NM	NS	ZO

Let us split evenly the source voltage (e.g. 3.5V) onto seven logical levels corresponding to linguistic levels and enumerate them with integer numbers from -3 to +3. Then Table 2 will represent Table 1 as the function of seven-valued logic.

Manuscript received Nov. 12, 2007; revised Feb. 18, 2007

Table 2: The Seven-Valued Function

		e						
ce	0	-3	-2	-1	0	1	2	3
	-3	0	1	2	3	3	3	3
	-2	-1	0	1	2	3	3	3
	-1	-2	-1	0	1	2	3	3
	0	-3	-2	-1	0	1	2	3
	1	-3	-3	-2	-1	0	1	2
	2	-3	-3	-3	-2	-1	0	1
	3	-3	-3	-3	-3	-2	-1	0

It is seen from Table 2 that the function is symmetric with respect to "North-West – South-East" diagonal and its values can be calculated as $e - ce$. This dependency is shown in Fig.7.

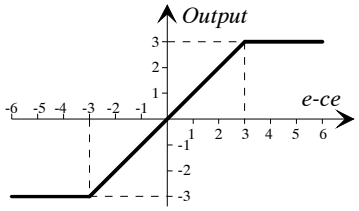

Fig.7 Graphical representation of the function specified by Table 2.

It apparently follows from comparison of Fig.3 (b) and Fig.7 that in order to reproduce the function specified by Table 2 it is sufficient to have one two-input summing amplifier and one one-input amplifier that will be called inverter.

Note that inversion of logic variables lying within $-k \div +k$ interval is the operation of diametric negation $\bar{x} = -x$; the operation $\overline{V_{out}} = V_{dd} - V_{in}$ corresponds to it in the terms of summing amplifier's input and output voltages. Thus CMOS circuit containing 12 transistors and 5 resistors, which implements our function, is shown in Fig.8.

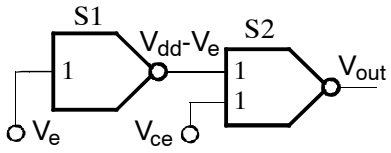

Fig.8 Implementation of the fuzzy controller specified by Table 2.

2.4.2 Example 2

This example is taken from [9, pp. 168 – 172]: "Manipulator for Man-Robot Cooperation (Control Method of Manipulator/Vehicle System with Fuzzy Inference)".

In the considered example the experimental manipulator has two force/torque sensors. One of them is the operational force sensor F_h; the other is "the environmental force sensor" ω. Each of input and output variables of the manipulator controller is represented with three linguistic variables – S (small), M (middle) and B (big). The controller has five fuzzy rules, as it follows:

If $\omega = S$ then Output=B;
If $\omega = B$ then Output=S;
If $\omega = M$ and $F_h = S$ then *Output*=S;
If $\omega = M$ and $F_h = M$ then *Output*=M;
If $\omega = M$ and $F_h = B$ then *Output*=B;

The controller *Output* is three-valued logic function specified in Table 3.

Table 3: The ternary function

		F_h		
		−1	0	+1
ω	−1	+1	+1	+1
	0	−1	0	+1
	+1	−1	−1	−1

It can be simply proved by trivial substitution that $Output = maj(2\omega, -F_h, 0)$ and CMOS implementation coincides with the circuit shown in Fig.8, if make substitutions $V_e = V_{F_h}$, $V_{ce} = V_\omega$ and change the weight of the input V_ω to 2.

2.4.3 Example 3. Fuzzy Controller for Washing Machine

This example is taken from Aptronix Incorporated (http://www.aptronix.com/fuzzynet).

A. Controller specification
Input variables:
Dirtiness of clothes: *Large* (*L*), *Medium* (*M*), and *Small* (*S*);
Type of dirtiness: *Greasy* (*G*), *Medium* (*M*), and *Not Greasy* (*NG*).

Output variable is *washing time* (minutes): *Very Long* (*VL*), *Long* (*L*), *Medium* (*M*), *Short* (*S*), and *Very Short* (*VS*).

Fuzzy rules are represented in Table 4.

Table 4: Matrix of linguistic variables

Wash. time		*Dirtiness of clothes*		
		S	*M*	*L*
Type of dirt.	*NG*	*VS*	*S*	*M*
	M	*M*	*M*	*L*
	G	*L*	*L*	*VL*