# Design and Application of an Analog Fuzzy Logic Controller

Shuwei Guo, Liliane Peters and Hartmut Surmann

GMD-SET, Schloss Birlinghoven, D-53754 St. Augustin, Germany

**Abstract — In this paper we present an analog fuzzy logic hardware implementation and its application to an autonomous mobile system. With a simple structure the fabricated fuzzy controller shows good performance in processing speed and area consumption. Accomplished with 13 reconfigurable rules, a speed of up to 6 MFLIPS has been achieved. To stress the advantages of the new architecture - speed and flexibility - the same control strategy is implemented on the new analog fuzzy controller and on a digital multi-purpose microcontroller in software. The results of the two implementations show that the analog approach is not only faster but also enough flexible to compete digital fuzzy approaches.**

## I. INTRODUCTION

Motivated by Zadeh, explored and validated by Mamdani, fuzzy logic control (FLC) has been used successfully in numerous control systems. Most applications rely on conventional digital computers or microprocessors programmed with a sequential calculation of the fuzzy logic quantities to perform the logic inferences. It limits generally applications to low-speed problems. Real-time systems often require very short time responses. In these cases, a hardware implementation seems to be the only solution.

The first fuzzy chip was reported in 1986 at AT&T Bell Lab [21]. Since then many different approaches have been suggested [6][9][26]. Depending on the design techniques employed they are classified into two groups: digital and analog. The digital approach originated from Togai and Watanabe's work [21] and resulted in some useful chips [15][20][24]. Generally a digital fuzzy system is either a fuzzy (co-)processor [14][22] or a digital ASIC [3][18], which contains logic circuits to compute the fuzzy algorithm, memories to store fuzzy rules, and generators or look-up tables for membership functions of the input and output variables. Compared to its analog counterpart, the digital approach has greater flexibility, easier design automation, and good compatibility with other digital systems. However, most of the digital systems require A/D and D/A converters to communicate with sensors and/or actuators. Furthermore, the digital systems are more complex and need larger chip area, e.g. the synthesis of a 4-bit maximum operation in [24] results in a CMOS unit of nearly 100 transistors.

The research on analog fuzzy systems started with the pioneering work of Yamakawa [25][26], and was followed by many researchers [4][6][8][9][13]. With the nonlinear characteristics of the active devices in analog circuit, the fuzzy elements can be implemented in very simple structures. This brings a reduction in the circuit complexity which implies better speed performance and reduced chip area consumption. Until now the main drawback of the analog approaches has been their poor flexibility.

Some programmable analog fuzzy chips have been suggested. Yamakawa *et al.* programs the membership functions and the fuzzy rules over digital registers which needs a rather large chip area. With a 2µm BiCMOS technology, the die size of the fuzzy inference part (rule chip) is 11.0 x 10.9 $mm^2$ while the defuzzifier chip takes an area of 3.4 x 2.9 $mm^2$ with a 3 µm bipolar technology. The processing speed of the fuzzy system is 1,400,000 fuzzy inference/s without defuzzification and 625,000 fuzzy inference/s including defuzzification [9]. Kettner *et al.* generates the layout from existing generic macrocells which are programmable only during the prototyping phase [6]. Ungering *et al.* introduce a mixed-mode fuzzy controller with a digital inference unit to achieve more flexibility. To avoid A/D and D/A converters, they used sorted pointers instead of the analog membership values. Unfortunately, this limits the number of the overlapping membership functions. By using FPGAs (XILINX 4010) for the digital part and Opamps for the analog part, an operation speed of some 10KHz was achieved [23].

To overcome these drawbacks, we propose a novel architecture for the hardware implementation of fuzzy logic control. After a short summary of the fuzzy inference algorithm, which forms the theoretical background of our hardware, we give a description to the new analog approach in section III. To get a better understanding of the advantages of the architecture presented, a real-time collision avoidance for an autonomous system is chosen. In section IV a brief description to the autonomous system is given. For this kind of application not only the processing speed is of great importance but also the flexibility of the implemented rule-base. With changing goals and/or environment new rule-bases are required. The system implementations of the control strategy in analog and digital techniques are presented in section V. A comparison between the two approaches concludes the paper.

## II. FUZZY INFERENCE

The fuzzy inference engine is the kernel of any fuzzy logic controller. Its dynamic behaviour is generally characterized by a set of fuzzy rules of the form:

*if* (a set of conditions are satisfied)
*then* (a set of consequences can be inferred).

The *if*-clause, an antecedent, is a condition in the application domain; the *then*-clause, a consequent, is a control action given to the process under control. With a set of fuzzy rules the fuzzy inference engine is able to derive a control action for a given set of input values. In other words, a control action is determined by the observed inputs which represent the state of the process to be controlled with the control rules. The approach used in fuzzy control is based on the approximate reasoning method of the generalized modus ponens (GMP). For example, for a two-input single-output n-rule fuzzy system, the GMP states:

| | |
|---|---|
| Premise: | is A' and y is B' |
| Implication $R_1$: | if x is $A_1$ and y is $B_1$ then z is $C_1$ |
| | ..... |
| also $R_i$: | if x is $A_i$ and y is $B_i$ then z is $C_i$ |
| | ..... |
| also $R_n$: | if x is $A_n$ and y is $B_n$ then z is $C_n$ |
| Conclusion: | z is C', |

in which *x*, *y* and *z* are linguistic variables and represent two inputs (process states or sensor measurements) and one output (control action). $A_i$, $B_i$ and $C_i$ are fuzzy sets defined on the appropriate universe U, V and W, respectively, with i = 1, 2, …, n. The fuzzy conditions in the antecedents are combined by the connective "*and*", while the sentence connective "*also*" links the rules into a rule set, or equivalently a fuzzy rule base. It should be noted that $A_i$, $B_i$, $C_i$, as well as A', B' are *apriori* known but C' will be deduced.

Basically a fuzzy inference process involves two concepts, the fuzzy implication and the compositional rule of inference. For the fuzzy rule "*if x is $A_i$ and y is $B_i$ then z is $C_i$*", the fuzzy relation can be expressed as:

$$R^i = (A_i \times B_i) \rightarrow C_i, \tag{1}$$

where × represents the Cartesian product; → denotes an operator for fuzzy implication. For a *n*-rule system, the fuzzy relation R is therefore a *n*-ary matrix:

$$R = \{R^1, R^2, \ldots, R^i, \ldots, R^n\}. \tag{2}$$

If we have the observations A' and B', then the fuzzy conclusion C' can be inferred by

$$C' = (A', B') \circ R, \tag{3}$$

where "∘" represents the compositional operator. This fuzzy reasoning is called the compositional rule of inference.

It is proved [7] that if the sentence connective "*also*" in the rule base is interpreted as union operation

$$R = \bigcup_{i=1}^{n} R^i, \tag{4}$$

then the fuzzy control action inferred from the complete set of fuzzy control rules is equivalent to the aggregated results derived from individual control rule:

$$C' = (A', B') \circ \bigcup_{i=1}^{n} R^i = \bigcup_{i=1}^{n} (A', B') \circ R^i. \tag{5}$$

Within the numerous inference strategies, Mamdani's technique is the most commonly used in the existing fuzzy control systems owing to its simplicity. In this method, the minimum operation is adopted for computing a fuzzy implication relation, and the *max-min* as the compositional rule of inference.

If we interpret the sentence connective "*and*" in the antecedent of the fuzzy rule as the intersection operation, the Cartesian product is realized by the minimum operation:

$$\mu_{A_i \times B_i}(x, y) = \mu_{A_i}(x) \wedge \mu_{B_i}(y) . \tag{6}$$

The membership function for the *i*th fuzzy relation can then be calculated as:

$$\mu_{R^i}(x, y, z) = \mu_{A_i}(x) \wedge \mu_{B_i}(y) \wedge \mu_{C_i}(z) . \tag{7}$$

The conclusion deduced from the *i*th rule $C'_i$ for the inputs A' and B' can be computed with the max-min compositional rule of inference:

$$\mu_{C_i'}(z) = \max_{x \in U} \max_{y \in V} \{ (\mu_{A'}(x) \wedge \mu_{B'}(y)) \wedge \mu_{R^i}(x, y, z) \}$$

$$= \alpha_i \wedge \mu_{C_i}(z) \tag{8}$$

where

$$\alpha_i = \max_{x \in U} (\mu_{A'}(x) \wedge \mu_{A_i}(x)) \wedge \max_{y \in V} (\mu_{B'}(y) \wedge \mu_{B_i}(y)) \tag{9}$$

$\alpha_i$ is called the firing strength (or weight) of the *i*th rule for the inputs A' and B'. In practical applications, the inputs are usually singletons, e.g., the measurements from sensors, A' = $x_0$ and B' = $y_0$. The related membership functions $\mu_{A'}(x)$ and $\mu_{B'}(y)$ are equal to zero except at the point $x = x_0$, and $y = y_0$, where $\mu_{A'}(x_0) = 1$, and $\mu_{B'}(y_0)=1$. In this case, the firing strength can be reduced to:

$$\alpha_i = \mu_{A_i}(x_0) \wedge \mu_{B_i}(y_0) . \tag{10}$$

The *n* fuzzy rules in the rule base are aggregated with the union operation. The overall output is then computed by combining the individual result from each rule in the rule base:

$$\mu_{C'}(z) = \bigcup_{i=1}^{n} \{\mu_{C_i}(z)\} . \tag{11}$$

For simplification in hardware implementations, a set of singletons is usually adopted in the consequent part of the fuzzy rule. Assume this set consists of *k* terms, i.e., $\mu_{C_i}(z) = \mu_C(z_j) = 1$ (i = 1, 2, …, n, while j = 1, 2, …, or *k*), then the inferred control action as a response to the actual input ($x_0$, $y_0$) can be written as

$$\mu_{C'}(z_j) = \bigcup_{i=1}^{n} \alpha_{ij}, \ (j = 1, 2, \ldots, k) \tag{12}$$

where $\alpha_{ij} = \alpha_i \wedge \mu_{C_i}(z)\big|_{z = z_j} = \alpha_i$ is the contribution of the *i*th rule to the *j*th term in the consequent part.

Experiments and theoretical investigations proved that Mamdani's technique yields better control results than that of other methods in fuzzy control applications [7]. Moreover, the operators used in this approach are very easy to implement in hardware.

## III. ANALOG FUZZY LOGIC CONTROLLER

A fuzzy logic controller is a system realization of the fuzzy control algorithm. Basically it consists of three function blocks: a *fuzzification interface,* a *fuzzy inference engine,* and a *defuzzification interface*. As mentioned earlier in the introduction, several problems exist in the implementation of fuzzy logic controllers with analog technology. To overcome these problems we propose a new architecture with the following main features:

- an adjustable membership function circuit;
- a reconfigurable inference engine; and
- a small area defuzzification block.

A short description of each block with the stress on the architecture novelty is described in the following.

### *A. Fuzzification circuit (MFC)*

The input data of a fuzzy logic controller are usually crisp values acquired from sensor measurement. Therefore, a fuzzification interface is needed to calculate the belonging (membership) of the observed inputs to the defined linguistic terms in the preconditions of the fuzzy rules. This conversion is done by so called membership function circuits (MFCs). The output of a MFC gives the grade of membership of the input to a related linguistic term in current or voltage mode.

Basically, a fuzzy term can be defined by a membership function of trapezoidal shape as shown in Fig. 1. The core of the function corresponds to the region which has a full membership ($\mu = 1$); the ascending and descending boundaries are the regions with partial matching or membership ($0 < \mu < 1$); the support of the defined fuzzy term is then the sum of the core and the boundaries in which $\mu > 0$. Commonly four types of membership functions are adopted in the hardware design of fuzzy systems: S-shaped function, Z-shaped function, triangular-shaped function and trapezoidal-shaped function. The first three types of function can be considered as special shapes of the last one.

Various approaches have been proposed to generate analog nonlinear membership functions in either current or voltage mode. As a general case in the solutions reported in [6][9][10], a MFC is built of two stages: the first for S- and Z-subfunctions generation and the second for the combination of these subfunctions. The positive and negative slopes of the generated membership function are regulated within the S- and Z-subfunction circuits, while the position and the type of the function are assigned with reference voltages or currents in the combination part.

Fig. 2 shows the schematic diagram of the proposed membership function circuit. In is built of a coupled differential amplifier which has two differential pairs (DP1 and DP2) and a common load R. The two reference voltages, low reference $V_{r1}$ and high reference $V_{r2}$, where $V_{r1} < V_{r2}$, define the membership function. Thanks to the coupling effect of the load resistor R, the subfunction generation and the final output combination are completed within one stage.

Depending on the relative values between the input voltage $V_i$ and the reference voltages $V_{r1}$ and $V_{r2}$, the circuit operates in one of the five regions (I to V) shown in Fig. 1. Assume

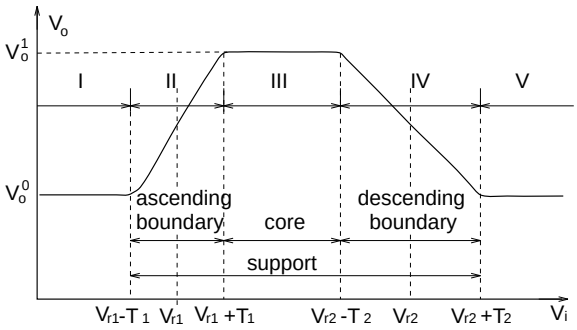



Fig. 1. Trapezoidal shape of a shape membership function

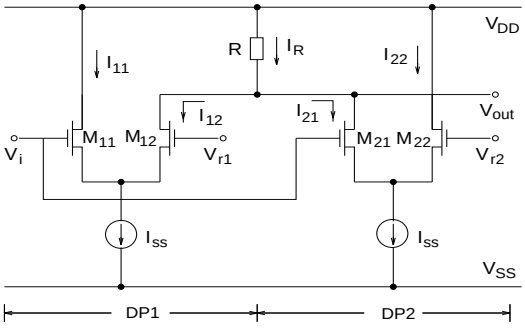



Fig. 2. The schematic diagram of the MFC.

that the current sources $I_{ss}$ in DP1 and DP2 are ideal and equivalent, the input devices in each differential pair are symmetric, the half widths of the transfer regions of DP1 and DP2 are $T_1$ and $T_2$ ($T_i = \sqrt{2I_{ss}/\beta_i}$, and $\beta_i$ is the transconductance parameter of the input devices in DP1 and DP2 with i = 1 and i= 2 respectively, then the five operating regions of the MFC illustrated in Fig. 1 can be described as follows:

- *Region I*: when $V_i < V_{r1} - T_1$, $M_{11}$ and $M_{21}$ are in their cutoff regions while $M_{12}$ and $M_{22}$ are in the saturation states. The current through the load resistance R is equal to $I_{ss}$. The output voltage is $V_{out} = V_{DD} - I_{ss}R = V_o^0$, which corresponds to a membership function value 0.
- *Region II*: when $V_{r1} - T_1 \le V_i \le V_{r1} + T_1$, DP1 is in its transfer region. If $V_{r2} - V_{r1} > T_1 + T_2$, then DP2 keeps the same state as in region I. As $V_i$ increases, the membership function circuit gives a linear ascending answer between 0 and 1.
- *Region III*: when $V_{r1} + T_1 < V_i < V_{r2} - T_2$, both $M_{11}$ and $M_{22}$ are in cutoff states, the current flowing through the load R is zero. The output voltage is $V_{out} = V_{DD} = V_o^1$ which corresponds to a membership function value 1.
- *Region IV*: when $V_{r2} - T_2 \le V_i \le V_{r2} + T_2$, DP1 keeps the same state as in region III and DP2 operates in its transfer region, antisymmetric to the case in region II. As$V_i$ increases, the membership function circuit gives a linear descending answer between 1 and 0.
- *Region V*: when $V_i > V_{r2} + T_2$, the coupled differential amplifier operates as in region I but DP1 and DP2 have a

reverse role. The output voltage is $V_o^0$ which corresponds to a membership function value 0.

Normalized with $\left(V_{out}-V_O^0\right)/\left(V_O^1-V_O^0\right)$ the generated membership function can be summarized as:

$$\mu = \begin{cases} 0, & V_i < V_{r1}-T_1 \quad \text{region I} \\ \frac{1}{2}\left[1+\left(\frac{\beta_1 (V_i-V_{r1})^2}{I_{ss}} - \frac{\beta_1^2 (V_i-V_{r1})^4}{4I_{ss}^2}\right)^{1/2}\right] & \\ & V_{r1}-T_1 \le V_i \le V_{r1}+T_1 \quad \text{region II} \\ 1, & V_{r1}+T_1 \le V_i \le V_{r2}-T_2 \quad \text{region III} \\ \frac{1}{2}\left[1-\left(\frac{\beta_2 (V_i-V_{r2})^2}{I_{ss}} - \frac{\beta_2^2 (V_i-V_{r2})^4}{4I_{ss}^2}\right)^{1/2}\right] & \\ & V_{r2}-T_2 \le V_i \le V_{r2}+T_2 \quad \text{region IV} \\ 0, & V_i > V_{r2}+T_2 \quad \text{region V} \end{cases} \tag{13}$$

As shown in Fig. 1 and Eq. 13, the definition of the implemented membership function is dependent on the reference voltages $V_{r1}$, $V_{r2}$, transconductance parameters $\beta_1$ and $\beta_2$, and current $I_{ss}$. By changing the values of these parameters in the circuit presented in Fig. 2, the membership function has different shape types, slopes, and position on the voltage axis. For instance, from the general membership function with trapezoidal shape (Fig. 3-d), we get a triangular shape when $V_{r1} + T_1 = V_{r2} - T_2$ (Fig. 3-c). The same function changes to a S- or Z-shape type if $V_{r1}$ is equal to the highest potential in the circuit (i.e., $V_{DD}$) or $V_{r2}$ is equal to the lowest potential (i.e., $V_{ss}$) respectively (see Fig. 3-a and -b).

The positive and negative slopes of the membership function are mainly determined by the transconductance parameters of the input devices in DP1 and DP2, respectively. Fig. 4

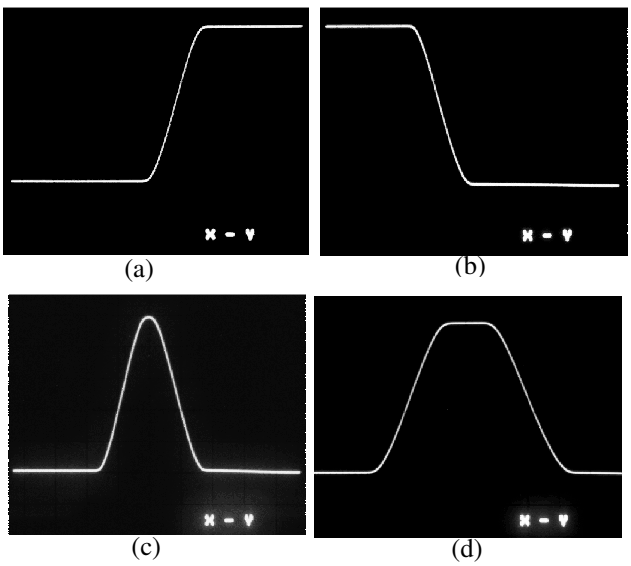



Fig. 3. Four types of membership functions realized by the MFC shown in Fig. 2. (a) S-function, (b) Z-function, (c) triangular function and (d) trapezoidal function.

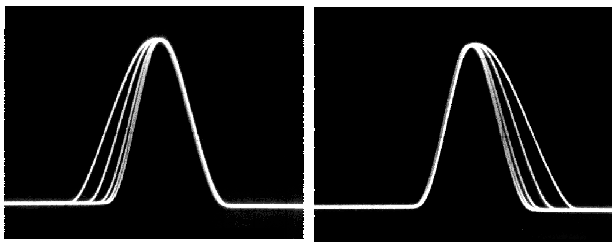



Fig. 4. Membership functions with different slopes (a) positive and (b) negative.

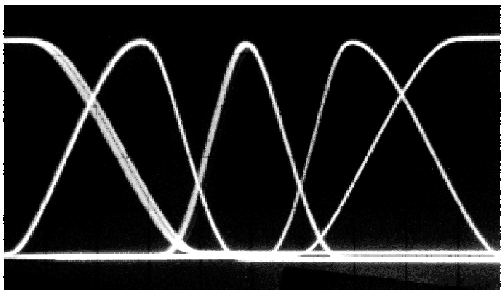


Fig. 5. Fuzzification with 5 fuzzy terms.

presents the change in slopes of the membership function when the widths of the input devices are varied from 20μm to 100μm in DP1 (Fig. 4-a) and DP2 (Fig. 4-b). To conclude the description of the membership function circuit, an example is presented in Fig. 5. It gives a possible 5-term fuzzification distribution for *negative large (NL), negative small (NS), …, positive large (PL).* The complete set of linguistic terms with different definitions are built by choosing different values for $V_{r1}$, $V_{r2}$, $\beta_1$ and $\beta_2$ with the same MFC.

### *A. Reconfigurable inference engine*

The next block in a fuzzy controller is the fuzzy inference engine. With Mamdani's inference technique, the inference is completed by a set of intersection and union operations. In our proposed architecture, these operations are realized with minimum (*min*) and maximum (*max*) circuits, which are implemented with a CMOS analog technology in voltage-mode.

Fig. 6-a shows the basic structure of a two-input *max* circuit. It consists of two NMOS devices $M_1$, $M_2$ and a current source $I_{ss}$. Two inputs are connected to the gates of $M_1$ and $M_2$ separately. The output $V_{max}$, which is connected to the common source of the input devices, has always the bigger value of the two inputs $V_1$ and $V_2$ with an offset voltage

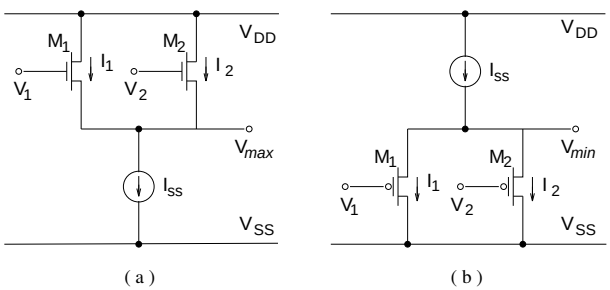
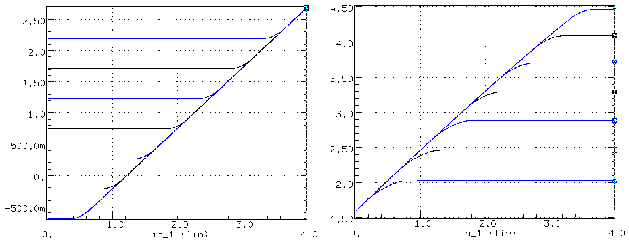



Fig. 6. Basic structure of (a) *max* and (b) *min* operators.



(a) (b)

Fig. 7. Simulation results of (a) *max* and (b) *min* operators.

$V_{off_{max}}$:

$$V_{max} = max\,(V_1, V_2) - V_{off_{max}} \tag{14}$$

Symmetrically, a *min* operator can be easily implemented by using PMOS devices. As indicated in Fig. 6-b, the output voltage of this circuit is:

$$V_{min} = min\,(V_1, V_2) + V_{off_{min}} \tag{15}$$

where $V_{off_{min}}$ is an offset voltage in the circuit.

The HSPICE simulation results of the two-input *max* and *min* circuits are shown in Fig. 7-a and -b, respectively. The basic *max* or *min* circuit can be extended to a multi-input circuit by putting several input transistors in parallel.

For the purpose of rule reconfiguration, analog multiplexers (MUXs) have been introduced in the inference engine. Fig. 8 shows the block diagram of a three-input *n*-rules reconfigurable inference engine. If the consequent part of the fuzzy rules are labelled with *k* linguistic terms, then the possible combination of the rules is *n*k*. Inserting *n* MUXs between the *if* and *then* parts, the outputs of *n* antecedent parts are connected to *k* consequent parts following the configuration of the rule base. By means of the control pins $S_1$,..., $S_n$, the MUX for each rule is defined individually. Depending on the selected voltage, the switch connects the weight calculated in the *if* part to the chosen *then* part. For example, the rule *i* can be changed from

$R_i$: *if A is $A_i$ and B is $B_i$ and C is $C_i$ then z is NL*

into

$R_i$: *if A is $A_i$ and B is $B_i$ and C is $C_i$ then z is Z*

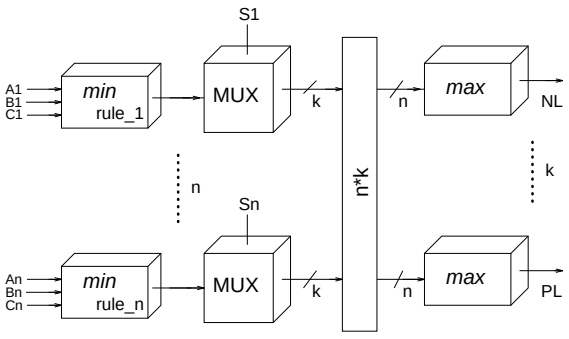



Fig. 8. A reconfigurable fuzzy inference engine.

just by changing the voltage applied to the control pin $S_i$. In the first case the pin $S_i$ was connected to a voltage corresponding to *NL* and in the second case to a voltage corresponding to *Z*. Tests have shown that the circuit is very robust to fluctuations related to the applied control voltage. Up to 10% error in the applied voltage were neglected by the switch mechanism.

### *A. Defuzzification Circuit*

The output of a fuzzy inference engine is a fuzzy set which represents the possible distribution of the control action. For practical use, a crisp control output is usually required. Thus a defuzzification interface is necessary to convert the inferred fuzzy control action into a nonfuzzy (crisp) value. Among the suggested defuzzification strategies, the centre of gravity (COG) method is the most commonly used. In the case of a discrete universe, the output can be calculated as

$$z = \frac{\sum_{j=1}^{k} \mu_{C'}(z_j) \cdot z_j}{\sum_{j=1}^{k} \mu_{C'}(z_j)} \tag{16}$$

where *k* is the number of discrete fuzzy elements; $\mu_{C'}(z_j)$ is the inferred (or final) membership function related to the *j*th singleton term $z_j$ in the consequent z, i.e., the weight of $z_j$ for the final output computation. The theoretical analysis and experiment results proved that the COG strategy has a good steady-state performance. A FLC based on the COG method generally yields a lower mean square error than that based on other methods [7].

Several solutions have been proposed to build defuzzification circuits with COG method [9], [27]. In most cases a division circuit is used which necessitates a relatively large design area. To avoid this, we developed a new defuzzification circuit which calculates the center of gravity without employing a division circuit [5]. As shown in Fig. 9, the proposed implementation is based on the principle of a voltage follower-aggregation circuit. For a *k*-term defuzzification, *k* transconductance amplifiers ($A_1$ to $A_k$) are used to aggregate the inputs $V_1$ to $V_k$ . They represent the values of *k* singleton terms in the consequence part of the fuzzy inference system. Assume that the transconductance of an amplifier $A_j$ is $G_j$, i.e., $G_1$ for $A_1$, $G_2$ for $A_2$ ,..., $G_k$ for $A_k$, then the current

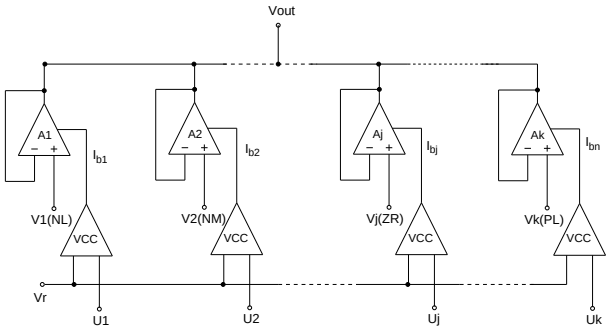



Fig. 9. The schematic of the defuzzifier circuit.

from the *j*th amplifier $A_j$ to the common point $V_{out}$ is $I_j = G_j (V_j - V_{out})$. Based on Kirchhoff's current law, the sum of the currents $I_j$ coming from the *k* amplifiers is zero. Thus we have:

$$V_{out} = \frac{\sum_{j=1}^{k} G_j V_j}{\sum_{j=1}^{k} G_j}. \quad (17)$$

This means that the output voltage of the circuit $V_{out}$ is the average of the inputs $V_j$. The contribution of each input to the output is weighted by the transconductance of the corresponding amplifier $G_j$.

From circuit analysis it is known that the transconductance $G_j$ of an amplifier is proportional to the root of the current source $I_{ss}$ when the amplifier operates in its linear region, and that a voltage-current converter (VCC) circuit converts an input voltage $U_j$ into a current $I_j$ with a square law. If we use a voltage-current converter to drive the current source of a transconductance amplifier and set the inputs of the converters to the outputs of the inference engine, then the proposed circuit in Fig. 9 represents the defuzzified value of the inference process:

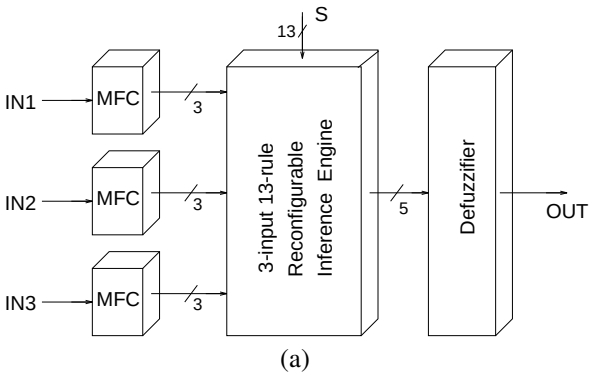
$$V_{out} = \frac{\sum_{j=1}^{k} U_j V_j}{\sum_{j=1}^{k} U_j}, \quad (18)$$

where $V_j$ is the value of the *j*th singleton term in the consequent, and $U_j$ is inferred membership value related to $V_j$.

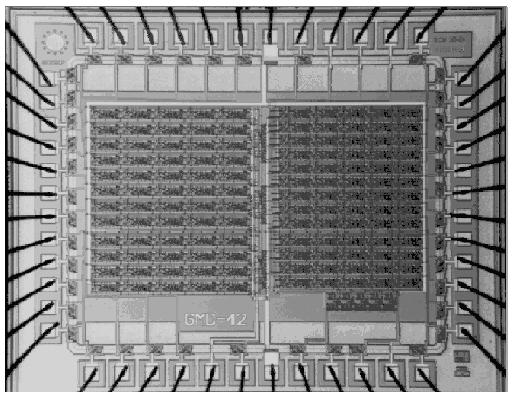
### *A. Chip Realization*

To confirm the performance of the presented architecture, and fit the requirements of the application presented in section IV, a three-input one-output reconfigurable FLC has fabricated. Fig. 10-a presents the block diagram of the controller. The input variables are fuzzified with three linguistic terms: *small*, *medium* and *large*, while the output variable are characterised by 5 singletons. Due to the overlap of the membership functions, 13 well-distributed fuzzy rules cover the whole control surface. Thus, for each point on the control surface there is at least one rule activated [11].

This design was fabricated with a 2.4µm CMOS process, and has an area of 3.6 x 4.5 $mm^2$ (Fig. 10-b). The test measurements of the chip show for a pulse input a rise time of 160ns and a decay time of 140ns (Fig. 11) . This corresponds to a speed of 6M FLIPS (Fuzzy Logic Inferences Per Second) including the defuzzification process. The power consumption is about 550 mW for the supply voltage of ±5V.




(a)



(b)

Fig. 10. A fuzzy logic controller;
(a) the block diagram, and (b) the photomicrograph

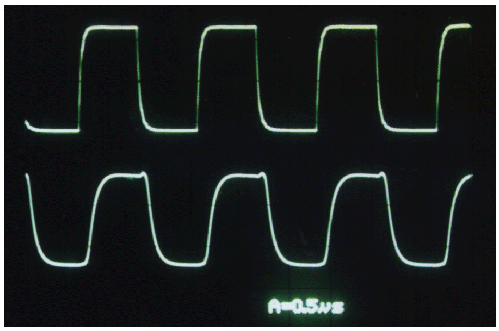


Fig. 11. Pulse response of the fuzzy chip.

## IV. A FUZZY LOGIC AUTONOMOUS DRIVEN SYSTEM

### *A. The Platform MORIA*

An autonomous mobile robot (AMR) has to cope with uncertain, incomplete or approximate information. Moreover it has to identify sudden perceptual situations and to manoeuvre in real-time. Therefore an AMR has been often used as a testbed for fuzzy logic strategies [1][12][16]. Our proposed hardware solution has several advantages for a successful implementation like:

- fast response time, real-time behaviour;
- analog I/O, direct control of the actuators (motors).
- reconfigurable rule-base, flexibility.

Our in-house autonomous system MORIA (Fig. 12) was used as a testbed for the proposed analog fuzzy chip. "MORIA" is an industrial mobile vehicle driven by two motors, one for forward / backward movement and the other for steering angle turning. The vehicle has a length of 175 cm (including the bumper with a length of 45 cm), a width of 73 cm and a height of 60 cm. It can transport a payload of 150 kg by a natural weight of 400 kg. Six sonar sensors mounted in two groups on

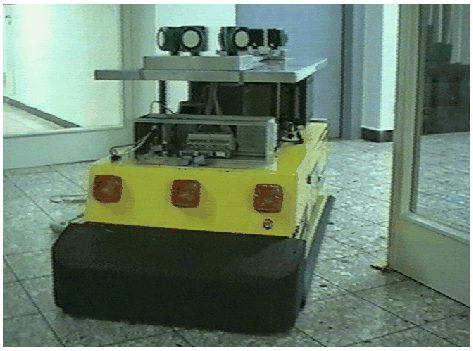


Fig. 12. Platform MORIA.

the platform - three at the front and three at the back - form the sensory input for the recognition of the environment. In addition the system has a communication link (infrared) by which a remote computer keeps in touch with the platform. The role of this link is to test the behaviour of the robot when the navigation commands (goals) come from a central unit, e.g., coordinating several robots, and the robot itself has just a collision avoidance strategy based on the local environment information.

### A. The control strategy

The developed fuzzy logic strategy represents the collision free navigation of an autonomous system for an unstructured real world environment. The heart of the system is a set of context dependent fuzzy rules. For a dynamic system such as the autonomous robot, the mapping of the output - moving speed and steering angle - depends not only on the current input - sonar values -, but also on the targets given by a planner.

The control strategy is divided into two modules. The first module is the navigator which insures a collision free navigation based on the local information given by sensors. The input variables to the controller are the data given by the sonor sensors and a command given by the planner. From the available six sensors only the three placed in the moving direction are estimated. Thus with each change in direction the evaluated sensor group is switched. The sensors provide an inaccurate measured distances between the robot and the obstacles. This information is fuzzified with the fuzzy terms like "*near*", "*far*", *etc*. By this means the system becomes robust against noise and measurement inaccuracies, which are covered by the membership function corresponding to each label.

The planner gives the global goal which is a list of linguistic commands enabling the robot to reach any given point in the environment. For example a list of commands could be: "*turn first left*", "*straight ahead*", "*next turn right*", "*stop*". These commands can either be sent by a central computer or by the user through the infrared communication link. It gives an additional input to the navigation module which changes the behaviour of the autonomous system independent of the given goal. For a detailed description of the avoidance strategy, see also [19].

## V. IMPLEMETATION

### A. Analog Implementation

For the analog implementation of the navigation strategy, we use prototypes of the presented analog fuzzy logic controller (see Section III. A).The input data has been obtained from the sonor sensors. The analog outputs drive the two motors and thus control the speed and direction of the platform. Fig. 13 shows the block diagram of the implemented control strategy in analog hardware.

Two fabricated fuzzy chips were used to implement the control strategy. As the fuzzy controller needs to evaluate the sensors only in the driving direction, a switch system selects the sensor groups, connecting the inputs of the controller either to the three sensors in front or at the rear of the vehicle. As shown in Fig. 14, each input variable is fuzzified with three membership functions. The output variables are characterized by five singleton terms.

The control actions left turn (LT) , right turn (RT) generated by the fuzzy chips for the obstacle avoidance can be regarded as the direct response to the local environment (sensor data), and these are based on the predefined fuzzy rules indicated in Fig. 15. The global linguistic command sent by the planer changes the final orientation control by assigning different weights to LT and RT. For example, the command *turn right*, implies a bigger weight (m >1) to RT; for the command *go ahead*, both RT and LT get the same weight.

It should be noted that in some cases local reactions superseed the global command in some cases. For instance, a command *turn next left,* which implies a change in weight for LT into big, will not be executed, if the output of the fuzzy controller based on the local environment estimation requires a weak action , which corresponds to a small weight for LT. Thus when the robot travels along a corridor, the global command *turn next left* is not executed at once if the robot recognizes a wall on the left side. After giving the turn left command, the robot will turn a litle bit to the left and follow the wall on the left side until it is possible to execute the given command, without hitting the wall. This implementation insures a collision free driving.

### A. Digital Implementation

As mentioned in the introduction, a digital implementation has usually a greater flexibility and an easier design automation than analog approaches. To compare the speed and the flexibility of our analog controller with a digital approach we implemented the same control strategy - same number of rules and same membership function - on a multi-purpose digital controller.

For a more realistic comparison between two approaches, we consider data dependencies of the fuzzy controller algorithm and available operations of today's microprocessors. The optimized digital calculation of the fuzzy algorithm is done in four steps [17]:

- All calculation operations are executed with integer numbers (8-bit).
- The membership functions are stored in look-up tables.
- Since the FRBS is a universal approximator [2] the most suitable algorithm for the digital processor is used.
- Data dependencies, especially the property of the minimum operator (min(x, 0) = 0) reduces the number of operations drastically; if one sub-premise of the antecedent of a rule is zero, then the output of the rule is zero and thus further cal-

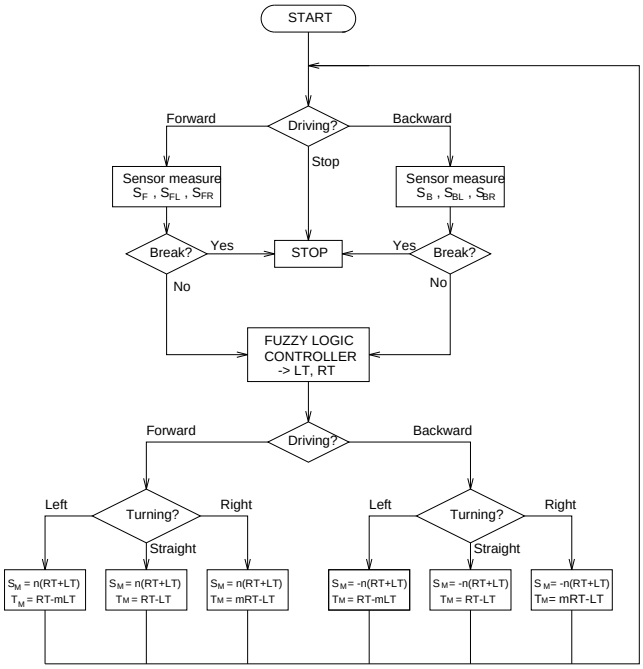



Fig. 13. Control strategy for the MORIA navigator

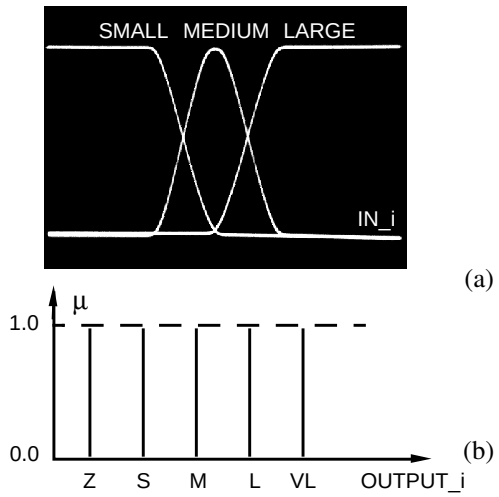



Fig. 14. Fuzzy set definitions for (a) the input and (b) the output variables.

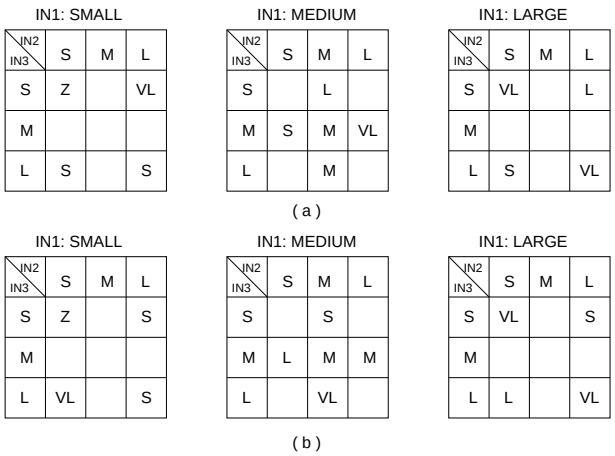



Fig. 15. Fuzzy rule sets for the (a)right turning (RT); (b) left turning (LT) strategy.

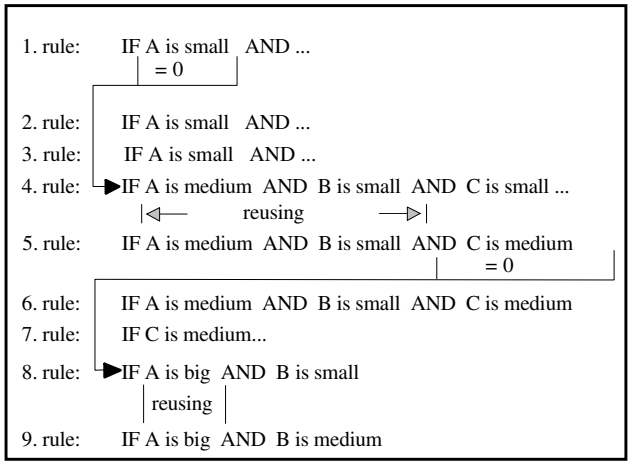



Fig. 16. Optimized antecedent evaluation. Rules 2 and 3 are skipped when the first premise is found to be zero.

culation are unnecessary; a sub-premises is calculated just once, and reused whenever appropriate (see Fig. 16). Based on these optimizations only a few rules have to be evaluated completely.

Table 1 shows a benchmark of the rule base with the optimization described above. The C code is compiled with the GNU compiler on two different general purpose processors. Although the above mentioned optimization were applied, the proposed analog hardware implementation is about 100-1000 times faster then the software implementation on general purpose microprocessors.

**Table 1: Inference speed (FLIPS) of the fuzzy controller for different processors**

| # activated rules | SUN SPARC 2 PC 486/33 | SUN SPARC 10 |
|---|---|---|
| 2 | 120K | 352K |
| 6 | 68K | 240K |
| 9 | 26K | 200K |
| 13 | 23K | 171K |

## VI. CONCLUSIONS

The distinctive features of the analog controller can be summarized as follows:

- ***Monolithic structure.*** Employing min-max operations and the centre of gravity defuzzification method, the whole fuzzy control algorithm is integrated on a single chip with a standard CMOS technology.
- ***Parallel inference engine.*** Working in voltage-mode, the fuzzy inference engine can be organized in parallel. Therefore the inference speed is independent of the number of implemented fuzzy rules.
- ***Simple circuitry.*** Implemented with the nonlinear characteristics of active devices in an analog technique, each building block contains only a few components. Con-

structed with such building blocks, a fuzzy controller takes a very small design area and has low power consumption.

- ***High-speed processing.*** The simple circuitry and the parallel structure of the inference engine make it possible to reach processing speeds of 6 MFLIPS.
- ***Reconfigurable rule base.*** Through the added control pins an on-line rule change is possible.

The presented experimental results show that it is possible to build a real-time navigation system with low-price hardware and inaccurate sensors. The presented analog and digital implementation have both fulfilled the real-time reaction requirement of 20ms. Some classical problems still remain, as for example oscillation. We attribute them partly to the limitation of the small rule bases and the small number of available sensors.

By implementing the same control strategy in software on a multi-purpose microprocessor and in hardware with a new fuzzy chip, we have shown that not only the analog approach is faster (with an inference time less than 160ns) but also enough flexible (reconfigurable) to compete digital fuzzy approaches.

## VII. REFERENCES


[1] H.R. Beom and H.S. Cho. *A Sensor-based Obstacle Avoidance Controller for a Mobile Robot Using Fuzzy Logic and Neural Networks*, in IEEE International Conference on Intelligent Robots and Systems, pp. 1470 - 1475, 6.1992.

[2] J. L. Castro, *Fuzzy logic controllers are universal approximators*, Technical Report #DECSAI-93101, University of Grenada, Department of Computer Science and Artificial Intelligence, 1993.

[3] H. Eichfeld, M. Löhner and M. Müller, Architecture of a Fuzzy Logic Controller with optimized memory organisation and operator design, in Proceedings of the 1st IEEE International Conference on Fuzzy Systems, pp. 1317-1323, San Diago, 1992.

[4] S. Guo, L. Peters and R. Camposano, *Yet Another Analog Fuzzy Controller*, Vorträge der ITG-Fachtagung: Mikroelektronik für die Informationstechnik, pp. 67-70, Berlin, März 1994.

[5] S. Guo and L. Peters, *A New Hardware Implementation of the Centre of Gravity Defuzzification Method*, Vorträge der 3. GI/ITG/GME-Fachtagung: Rechnergestützter Entwurf und Architektur mikroelektronischer Systeme, pp. 145- 150, Oberwiesenthal, Mai 1994.

[6] T. Kettner, K. Schumacher and K. Goser, *Realization of a Monolithic Analog Fuzzy Logic Controller*, in Proceeding of the 20th European Solid-state Circuit Conference, Sevilla, pp. 66-69, 1993.

[7] C. C. Lee, *Fuzzy Logic in Control Systems: Fuzzy Logic Controller*, IEEE Transaction on Systems, Man, and Cybernetics, vol. 20, no. 2, pp. 404-435, 1990.

[8] L. Lemaitre, M. J. Patyra and D. Mlynek, *Analysis and Design of CMOS Fuzzy Logic Controller in Current Mode*, IEEE Journal of Solid-state Circuits, vol. 29, no. 3, pp. 317-322 1994.

[9] T. Miki, H. Matsumoto, K. Ohto and T. Yamakawa, *Silicon Implementation for a Novel High-speed Fuzzy Inference Engine: Mega-FLIPS Analog Fuzzy Processor*, Journal of Intelligent and Fuzzy System, vol.1, no. 1, pp. 27-42, 1993.

[10] L. Peters, S. Guo and R. Camposano, *An Analog Fuzzy Logic Inference Engine*, in Proceedings of the 11th European Conference on Circuit Theory and Design--ECCTD'93, pp. 891-894, Switzerland, 1993.

[11] L. Peters, K. Beck and R. Camposano, *Adaptive Fuzzy Controller Improves Comfort*, IEEE International Conference on Fuzzy Systems, San Francisco, CA, pp. 512-515, 1993.

[12].A. Saffiotti, et al., *A Multivalued Logic Approach to Integrating Planning and Control*, Artificial Intelligence Center, SRI International, Technical Note No. 533, Menlo Park, CA 94025, USA, 4/1994.

[13] M. Sasaki, N. Ishikawa, F. Ueno and T. Inoue, *Current mode analog fuzzy hardware with voltage input interface and normalization locked loop*, IEEE International Conference on Fuzzy Systems, San Diego, CA, pp. 431-442, 1992.

[14] M. Sasaki, F. Ueno and T. Inoue, *7.5MFLIPS Fuzzy Microprocessor Using SIMD and Logic-Memory Structure*, in the Proceedings of the 2nd International Conference on Fuzzy Systems, pp. 527-537, San Francisco, 1993.

[15] K. Shimizu, M. Osumi and F. Imae, *Digital Fuzzy Processor FP-5000*, in Proceedings of the 2nd International Conference on Fuzzy Logic and Neural Networks, Iizuka, Japan, pp. 539-542, 1992.

[16] M. Sugeno and K. Murakami, *An Experimental Study on Fuzzy Parking Control Using a Model Car*, in *Industrial Applications of Fuzzy Control*, M. Sugeno (ed.), Elsevier Science Publishers B.V., North-Holland, pp. 125-138, 1985.

[17] H. Surmann, A. P. Ungering, T. Kettner and K. Goser, *What kind of hardware is necessary for a fuzzy rule based system*, in the Proceedings of the 3rd IEEE International Conference on Fuzzy Systems, pp. 274-278, Orlando, 1994.

[18] H. Surmann, A. Ungering and K. Goser, *Optimized Fuzzy Controller Architecture for Field Programmable Gate Arrays*, in Lecture Notes on Computer Science Nr. 705, pp.124-133, Springer, 1993.

[19] H. Surmann, J. Huser and L. Peters, *A Fuzzy System for Indoor Mobile Robot Navigation*, Proceedings of the IEEE Conference on Fuzzy Systems, Yokohama, March 1995, pp. 83-88.

[20] M. Togai and S. Chiu, *A Fuzzy Accelerator and a Programming Environment for Real-time Fuzzy Control*, in Proceedings of the 2nd IFSA Congress, Tokyo, pp. 147-151, 1987.

[21] M. Togai and H. Watanabe, *A VLSI Implementation of a Fuzzy Inference Engine: Toward an Expert System on a Chip*, Information Science, vol. 38, pp. 147-163, 1986.

[22] A. Ungering and K. Goser, *Architecture of a 64-bit Fuzzy Inference Processor,* in Proceedings of the 3rd IEEE International Conference on Fuzzy Logic Systems, pp. 1776-1780, Orlando, 1994.

[23] A. Ungering, D. Herbst, A. Weyergraf and K. Goser, *Architecture of a Mixed Mode Fuzzy Controller*, in proceedings of the 4th IEEE International Conference on Fuzzy Systems. pp. 1191-1196. Yokohama, Japan, 1995

[24] H. Watanabe, W. D. Dettloff and K. E. Yount, *A VLSI Fuzzy Logic Controller with Reconfigurable, Cascadable Architecture,* IEEE Journal of Solid-state Circuits, vol. SC- 25, no. 2, pp. 376-382, 1990.

[25] T. Yamakawa and T. Miki, T*he Current Mode Fuzzy Logic Integrated Circuits Fabricated by the Standard CMOS Process*, IEEE Transaction on Computers, vol. C-35, no. 2, pp. 161-167, 1986.

[26] T. Yamakawa, *High Speed Fuzzy Controller Hardware System*, in Proceedings of the 2nd Fuzzy System Symposium, pp. 122-130, Japan, 1986

[27] T. Yamakawa, *A Fuzzy Inference Engine in Nonlinear Analog Mode and Its Application to a Fuzzy Logic Controller*, IEEE Transaction on Neural Networks, vol. 4, no. 3, pp. 496-522, 1993.