Synthesis and VHDL Implementation of Fuzzy Logic Controller for Dynamic Voltage and Frequency Scaling (DVFS) Goals in Digital Processors

Hamid Reza Pourshaghaghi,
Juan Diego Echeverri Escobar and José Pineda de Gyvez
Electronic Systems Group,
Eindhoven University of Technology, Eindhoven,
the Netherlands

1. Introduction

The concept of power consumption is becoming the primary concern in modern high performance processors, and in digital circuits and system on chips (SoCs). While CMOS technology has been scaling towards smaller feature sizes, the performance of digital systems has been exponentially increasing as clock frequency increases. Also the computational workload and hence the activity of a digital circuit may change substantially and it exposes a lot of breakthroughs in the exploitation of adaptive low power methodologies. Dynamic voltage and frequency scaling (DVFS) is a popular system level power management technique that dynamically scales the supply voltage and clock frequency level of device (Rabaey, 2010). A DVFS system can be considered as a closed loop control system: contingent on the observed workload, supply voltage and operational speed gets adjusted. Since the changes in supply voltages do not occur instantaneously due to the fact that some delays are involved the large capacitance on the supply rails, the main real challenge in the design of such a system lies in how to measure and predict the workload of processor to change supply voltage accurately. The efficiency of DVFS strongly depends on the accuracy of the workload estimation, and note that misestimating can substantially reduce the effectiveness of such closed loop systems.

Most previous DVFS methods focused on offline profiling to learn the average case execution time or worst case execution time. Different closed loop adaptive controllers have been proposed to deal with time varying workloads. Most of them are based on conventional PID controllers and their variants e.g. PI controller or I controller. These kinds of configurations need offline profiling to tune the coefficients of the controller to be able to track or predict the workload variations. However, for different shapes of workload variations, it is necessary to do the off-line tuning again and determine the coefficients another time. So they are not considered as general solutions for adjusting voltage and frequency. Some other estimation methods e.g. adaptive filters have proposed for predicting workload and controlling the behavior of the power and energy savings. Unfortunately, most of them need offline profiling and/or the applications are limited to some specific periodic workload variations.

In this chapter, we discuss an on-line adaptive fuzzy logic controller for DVFS that is able to accurately and robustly predict and track the workload variations even when those variations are highly nonstationary or soft. Furthermore, we describe comprehensively how one can build the controller in VHDL and use it as the power management controller unit. We propose a new method to use for the defuzzification part of the fuzzy controller that makes the circuit faster. The fuzzy controller can be applicable to different kinds of workload variations with regards to real-time constraints, and can adaptively change the supply voltage and frequency of a processor. The proposed controller can be easily upgraded by adding new rules or adding new features to improve performance. In this chapter, all the practical limitations and real-time constraints for designing the fuzzy logic controller as the DVFS method will be discussed during design procedure.

2. Related works over power management techniques

So far a lot of research has been done to explore different approaches for performing DVFS. Many of the previous works are categorized in task level algorithms that use offline profiling to obtain the average-case execution time (ACET) or worst-case execution time (WCET) as models for the workloads of the given application. For example, one of the earliest works was presented in (Yao et al., 1995) where they assumed that the arrival time, deadline of workloads, and task execution time based on CPU cycle are given to designers as constants. The works proposed in (Im et al., 2006) and (Jejurikar & Gupta, 2006) are two more examples where in (Im et al., 2006) they proposed a technique to reduce the energy consumption based on WCET workload model using buffers; and in (Jejurikar & Gupta, 2006) a dynamic voltage scaling (DVS) method in the presence of task synchronization based on WCET workload model in multiprocessor environment was proposed. This kind of approaches cannot deal with the time-varying workload especially when the workload shows a large variation with nonstationary property. Another category of researches related to DVFS comprise techniques which require either application or compiler support to perform (Azevedo et al., 2002; Yang et al., 2001; Chung et al., 2002). Generality and offline profiling for different workload variations is still a big drawback existing in these classes of works.

Using adaptive approaches for DVFS leads to save more power and energy in comparison to the conventional techniques. Proposing closed loop system architectures started by introducing self-timed adaptive supply-voltage scaling for asynchronous circuits in (Nielsen et al., 1994) where in their architecture, first input-first output (FIFO) buffers are used in both inputs and outputs of the processor. The FIFO-buffers average the computational workload to adjust the supply voltage and frequency. The feedback is based on actual path delays of the circuit. The feedback signal controls the DC-DC converter based on the information derived from the FIFO's. After this architecture, other researchers used a similar configuration to adapt the power supply voltage to lower the power consumption in digital signal processors DSPs (Gutnik & Chandrakasan, 1997). The structure of the proposed system architecture in (Gutnik & Chandrakasan, 1997) is the same as (Nielsen et al., 1994), but they designed a configuration for synchronous designs, and variations in the computational workload were taken into account as well. Like the self-timed variable voltage system of (Nielsen et al., 1994), input data is buffered into a FIFO type of buffer to enable averaging of the workload. Then, the control loop controls the processing rate to avoid queuing overflow and underflow of the FIFOs. The controller in this methodology consists of a voltage regulator, a ring oscillator, a rate-compare block and a programmable look-up table (LUT). The controller block decides to change the voltage and frequency based on the processing rate and existent LUT. Disadvantage of this configuration is that it comes with extra latency as buffer utilization is only one measure of workload.

An evolution to closed loop control configuration, in the estimation of workload and adaptive control methods become main challenge for design efficient DVFS. These techniques aimed at estimating time varying workload using adaptive filters that most of them were based on a conventional proportional, integral and derivative (PID) controller. One famous PID based approach is presented in (Wei & Horowitz, 2003) where voltage samples are used to control a VCO to change the frequency as feedback signal for the buck converter. The reference signal and feedback signal come into the controller as variable frequency clocks, both feed into counters, and the number of transitions is counted for a fixed period of time. A PID controller, based on the calculated error value between its inputs, decides to change the voltage and frequency of the circuit. Some example of using PID controller to estimate workload variations are proposed in (Hughes and Adve, 2003; Gu & Chakraborty, 2008; Wu et al., 2005; Lu et al., 2002, 2003). In (Hughes and Adve, 2003), the PID controller is used to estimate the frame decoding time in multimedia applications and it was used in (Gu & Chakraborty, 2008) for 3-D games. In (Wu et al., 2005) and (Lu et al., 2003), the PI controller, which is a variant of the PID method, is applied to estimate the buffer occupancy for DVS targeting data buffered systems. In (Lu et al., 2002), an integral controller which is also a variant of PID method is designed and used to estimate workload estimation for performing DVS. Despite the PID controller is an adaptive filter, it suffers from possible overshooting and undershooting, depending on the selected coefficients. Also the PID controller for estimating application is useful when the designer select coefficient for specific workload variations and if the shape of workload changes, the coefficient should be defined again based on new workload variations. Hence, the tuning of coefficients critically determines the prediction accuracy.

Beside PID based controllers, some other estimators and adaptive filters are proposed to forecast workload variations in order to adjust voltage and/or frequency. In (Sinha & Chandrakasan, 2001), an adaptive approach for dynamic voltage scheduling on processors is presented based on workload prediction by filtering a trace history. In this work, they examine some conventional filters and evaluate their accuracy based on power saving amounts. They concluded that adaptive LMS filtering is the most powerful one and can be used to predict workload variations. Also in (Bang et al., 2009), they proposed a Kalman-filter based on-line estimator to predict and track the workload variation that can be applicable to periodic applications with soft real-time constrains.

Compared to previous approaches, the fuzzy logic controller is similar to adaptive filters in the sense of estimating the workload variations. However, the fuzzy logic controller can work as an on-line methodology without updating any parameters during run-time adaptations and also without any other information about the nature of the workload variation. The controller can estimate and track any kind of workload variations accurately and it does not require any coefficient tuning through offline profiling.

3. Principles of Dynamic Voltage and Frequency Scaling (DVFS)

The most important key to save power and energy of a digital circuit or processor is to reduce supply voltage and clock frequency according to the performance requirements. The power consumption of a clocked digital CMOS circuit is given by the well-known formula:

$$Power = \underbrace{\alpha.C.V_{dd}^{2}.F_{ck}}_{Dynamic\ Power} + \underbrace{I_{leak}.V_{dd}}_{leakage\ power} \tag{1}$$

where $\alpha.C$ is the total switched capacitance, V_{dd} is the supply voltage, F_{ck} is the clock frequency and I_{leak} is the off-state current of the circuit. By reducing the supply voltage and clock frequency, considerable power can be saved while $\alpha.C$ is generally fixed for a specific application. Over the years, researchers have proposed different hardware adaptive power management infrastructures to construct low power system-on-chip (SoC) integrated circuits. Among all the methods, DVFS methods are the most effective ones to save power consumption in processors. Conceptually, online DVFS problem for a digital CMOS circuit e.g. a processor is to scale voltage and frequency based on performance variation demands. The general block diagram of a dynamic supply voltage and frequency scaled system is shown in Fig. 1(Nielsen et al., 1994; Gutnik & Chandrakasan, 1997).

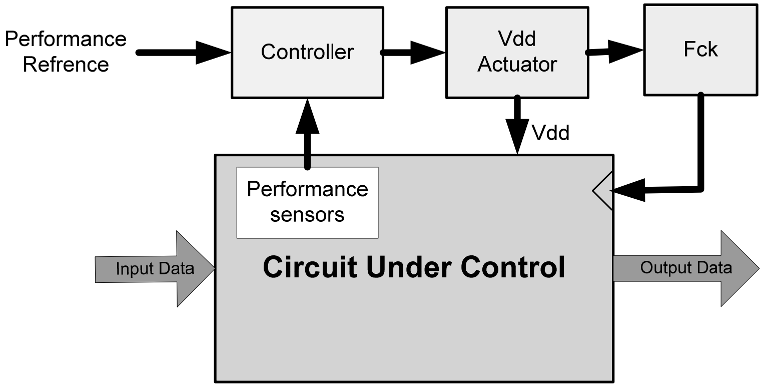


Fig. 1. General block diagram of a DVFS system

In this block diagram, there are three main components. The first component is a performance sensor that monitors the main specification of the processor e.g. average of supply current, temperature and supply voltage variations. The second component is the controller. This controller block works based on an input data received from the sensors by comparing it with the reference performance received from the power management unit or software to decide the change in supply voltage when necessary. The third block is the supply voltage actuator that can be on-chip or off-chip, e.g. a DC-DC converter and clock frequency actuators that can be a PLL. Since reducing the supply voltage causes increasing the delay of circuits, controlling the voltage and frequency of a processor dramatically depends on the accuracy of the controller.

Since there is a strong correlation between the supply current and the workload of a processor (Benini et al., 1999), the controller is designed based on observing and tracking of the average of current variations. The most important purpose is how to predict and track supply current variations of the processor and to drive it to operate at the lowest possible voltage and corresponding minimum frequency, for which a specific application can meet all of its deadlines under specific timing constraints. If the supply current tracking can perform in a proper way, the supply voltage and clock frequency of the processor can be adjusted w.r.t output predicted current signal. Supply voltage variations are same with variations of the predicted supply current signal. For determining clock frequency in each control time, the proper look up table corresponding to the delay-voltage model can be used. The delay of a CMOS gate can be modeled as

$$\tau = \frac{C_{gate}V_{dd}}{K(V_{dd}-V_{th})^{\beta}} \tag{2}$$

where K and β are technological parameters, and V_{th} is device threshold voltage. The cycle time of a design is modeled as a function of the critical path delay given as $T_C = L_D\tau$ where L_D is the logic depth in number of (equivalent) gates in the critical path. Therefore, the clock frequency for satisfying all timing deadlines of the circuit can be determined as

$$f_{ck}(t) = 1/T_c(t). \tag{3}$$

The relation of the normalized operating frequency versus normalized supply voltage of a sample CMOS digital circuit is shown in Fig. 2. As mentioned before, changing the processor clock frequency can be done by the available PLL in the circuit. PLL can only provide some limited clock frequencies, for instance suppose that a sample PLL can provide six different clock frequencies, like $\{f_{ck1}, f_{ck2}, \dots, f_{ck6}\}$ shown in Fig. 2.

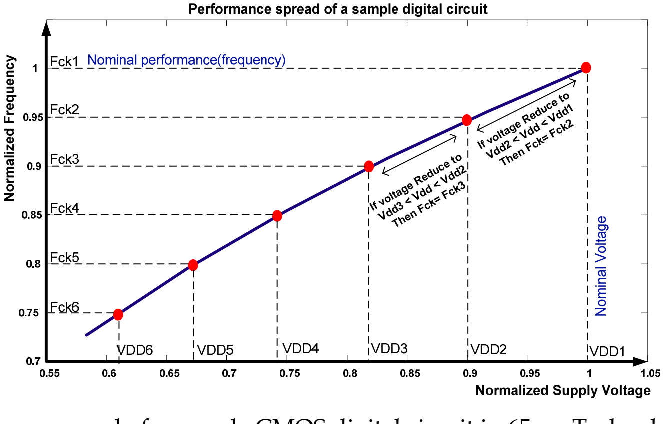


Fig. 2. Performance spread of a sample CMOS digital circuit in 65nm Technology.

Imagine a specific application is running with a constant frequency f_{ck1} at its nominal supply voltage V_{dd1}without any voltage scaling. Now suppose that the supply current is such that there is opportunity to save power by reducing the supply voltage. However, observe that when the supply voltage reduces (e.g. to a point between V_{dd1} and V_{dd2} as shown in Fig. 2), the frequency of operation would reduce as well to f_{ck2}. If the supply voltage goes for a value between V_{dd2} and V_{dd3}, then the frequency can switch to the f_{ck3} value. In this way, adjusting supply voltage to the lowest allowable value together with frequency scaling will ensure that the application is properly executed and the maximum possible power is saved. For switching the supply voltage to different possible values, it is needed to use voltage actuators like on-chip or off-chip DC-DC converters. In most DC-DC converters as voltage regulators, switching between voltage output levels takes a few tens of microseconds. For doing safe voltage and frequency switching, voltage and clock frequency changes may not be done in parallel. While the supply current is going to decrease, the frequency should first be decreased and subsequently the voltage should be lowered to the appropriate value. On the contrary, when the supply current is going to increase, the circuit requires the voltage to be increased first followed by the frequency update. This ensures that the voltage supply to the processor is never lower than the minimum required for the current operating frequency and avoids data corruption due to circuit failure.

4. DVFS based on fuzzy logic controller

The block diagram of the proposed dynamic voltage and frequency scaling configuration is shown in Fig. 3 (Pourshaghaghi & Pineda de Gyvez, 2009).

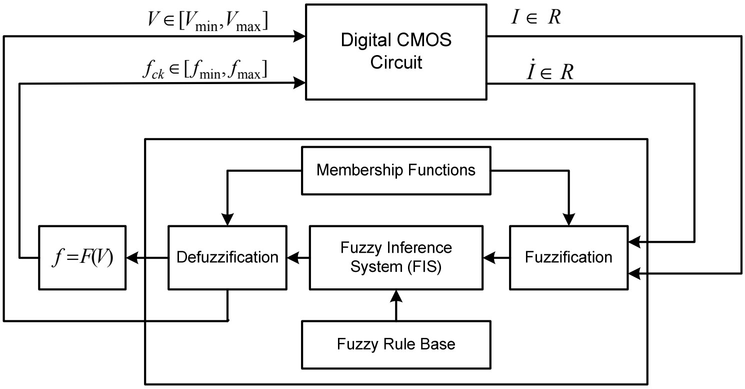
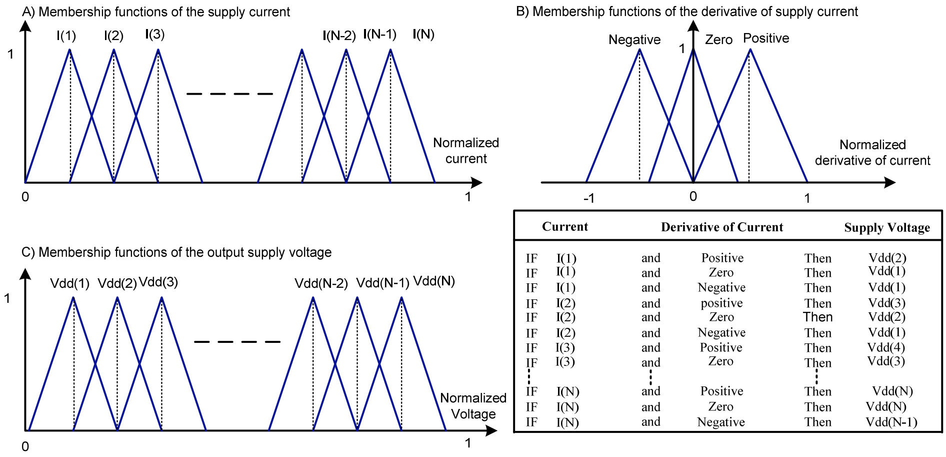


Fig. 3. Dynamic voltage and frequency scaling configuration based on supply-current tracking by fuzzy logic controller.

In this block diagram, the supply current and also the derivative of the supply current are observed as two inputs of the fuzzy logic block. The reason for using the derivative of the supply current is that it helps to predict the variations of the workload. If one can predict variations of the supply current, then it is easier for the actuators to act sooner. Consequently, the amount of saved power can be increased significantly, not to mention finishing the executing task on time. Given a specific value for the supply current, if the derivative is positive, it implies that the supply current is increasing. Otherwise, the supply current is decreasing. Therefore, the fuzzy *if-then* rules should be defined to follow this concept. It should be taken into account that, for having more precision to predict supply current variations, it is possible to compute the second derivative of the current. Thus, the fuzzy logic block receives two inputs: supply-current and its derivative. Based on these two inputs, the fuzzy logic block, as an expert system, can decide about the voltage and frequency of the processor. Actually, by this method, the fuzzy logic controller is tracking the supply current to decide upon the new voltage of the digital circuit. Actuators for supply voltage can be an on-chip or off-chip DC-DC converters. The same procedure can be done for determining the frequency of the processor. But for deciding about the final frequency value, it should be taken into account that the frequency obtained by fuzzy logic controller has to be greater than the frequency obtained by worst case execution time. Also the frequency can be defined based on a proper predefined look up table.

Based on performing different experiments, the proposed internal structure of fuzzy controller was resulted to have membership functions and fuzzy rules like ones shown in Fig. 4. In this structure, if N membership functions are defined for the supply current and 3 membership functions are defined for its derivative, then $N \times 3$ rules should define the fuzzy logic rule-base block. The rules should be defined in a way that the supply voltage tracks the variations of the supply current. Therefore, the proposed controller predicts first the supply current variations and then it decides on how to change the voltage and frequency pair. Using fuzzy logic sets, the fuzzy inference system (FIS) formulates the process of getting the output based on the defined input membership functions and the fuzzy if-then rules. Mamdani FIS is the most commonly useful methodology for applying fuzzy logic controllers on practical systems and we recommend using it for DVFS goals (Lee, 1999).


A) Membership functions of the supply current
I(1)
I(2)
I(3)
I(N-2)
I(N-1)
I(N)
1
0
1
Normalized current
B) Membership functions of the derivative of supply current
Negative
Zero
Positive
1
-1
0
1
Normalized derivative of current
C) Membership functions of the output supply voltage
Vdd(1)
Vdd(2)
Vdd(3)
Vdd(N-2)
Vdd(N-1)
Vdd(N)
1
0
1
Normalized Voltage

	Current		Derivative of Current		Supply Voltage
IF	I(1)	and	Positive	Then	Vdd(2)
IF	I(1)	and	Zero	Then	Vdd(1)
IF	I(1)	and	Negative	Then	Vdd(1)
IF	I(2)	and	positive	Then	Vdd(3)
IF	I(2)	and	Zero	Then	Vdd(2)
IF	I(2)	and	Negative	Then	Vdd(1)
IF	I(3)	and	Positive	Then	Vdd(4)
IF	I(3)	and	Zero	Then	Vdd(3)
⋮		⋮		⋮	
IF	I(N)	and	Positive	Then	Vdd(N)
IF	I(N)	and	Zero	Then	Vdd(N)
IF	I(N)	and	Negative	Then	Vdd(N-1)

Fig. 4. The defined membership functions of states, A) Supply current as the first input of the fuzzy controller. It has N membership functions named by I(1), I(2),…, I(N) , B) Derivative of supply current as the second input of the fuzzy controller, it has three membership functions named by Negative, Zero, and Positive, C) Supply voltage as the output of the fuzzy controller, it has N membership functions named by Vdd(1), Vdd(2),…, and Vdd(N). Fuzzy *if-then* rules are defined in the table.

Several experiments have been conducted to evaluate different aspects of the controller. In the first simulation, we designed a controller and implemented it on a sampled supply current of a processor near to reality. This supply current is shown in Fig. (5.a) and its derivative is shown in Fig. (5.b). Based on the internal fuzzy system structure described in Fig. 4, we have considered nine Triangular membership functions for supply current. These functions are defined between $150\mu A$ to $600\mu A$, without losing the generality, with a symmetrical shapes and widths. Each supply current membership function has 50% overlaps with its neighbor membership function (functions). We have considered five Triangular membership functions for derivative of supply current from $-250\,{}^{mA}/_{sec}$ to $250\,{}^{mA}/_{sec}$. Consequently, we defined 27 *if-then* rules based on the rules shown in Fig. 4. Nine symmetrical triangular membership functions for supply voltage have been considered as well. These membership functions have 50% overlap with each other and have same widths too. We used also the centre of area as the defuzzification method. The result of this simulation is shown in Fig. (5.c). As one can see from the supply voltage values, the fuzzy logic can track the variation of supply current very well. The output surface of fuzzy logic controller is shown in Fig. (5.d). In this figure, the entire span of supply voltage based upon the entire span of supply current and its derivative is displayed. It shows pseudo continuity of the output voltage with variations of workload.

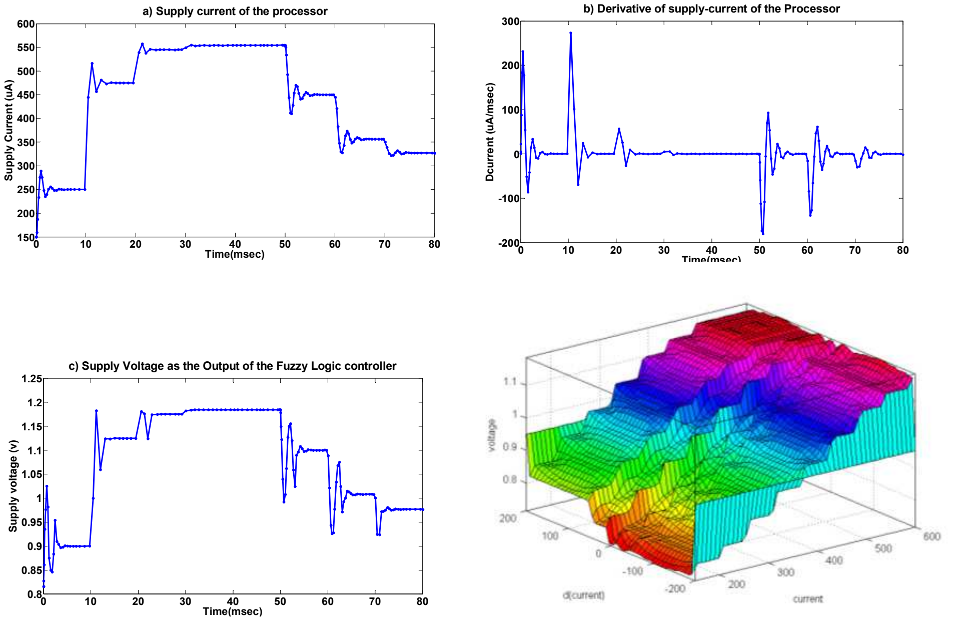


Fig. 5. Simulation results of applying fuzzy Logic controller (FLC) on a sampled supply current. a) Supply current of a sample processor, the variation of current is based on different applications, b) Derivative of the supply current, c) Voltage (output) of the FLC that goes to DC-DC converter, d) Output surface of the controller which shows variations rate of the voltage (output) regarding to input variations.

We simulated a PID controller on another supply current signal and compared the results with the fuzzy controller. Suppose that we have a supply current signal like the one shown in Fig. (6.a). We trained the PID controller with some simulation testing to find out what coefficients are the best for the proportional, integration and derivative part of the controller. Finally, with a trial and error method we found that with $k_p = 100$, $k_i = 2$ and $k_d = 1$, it can track the supply current very well. The tracking result is shown in Fig. (6.b). But when the shape of the supply current changed, similar to the supply current shown in Fig. (6.c), the PID could not track the variations with the same coefficients and we have to change coefficients again. The output of PID block in the second experiment is shown in Fig. (6.d). It is also important to mention that the fuzzy logic controller works well regardless of the system's inputs, while the PID controller requires the mathematical formulation of the system to adapt its coefficients to be able to work properly. One of the main advantages of the fuzzy logic controller is that the hardware implementation is easy because everything here is digital. Another advantage is that this controller can work on-line to track all workload circumstances with high speed and less error in comparison with other traditional control methods.

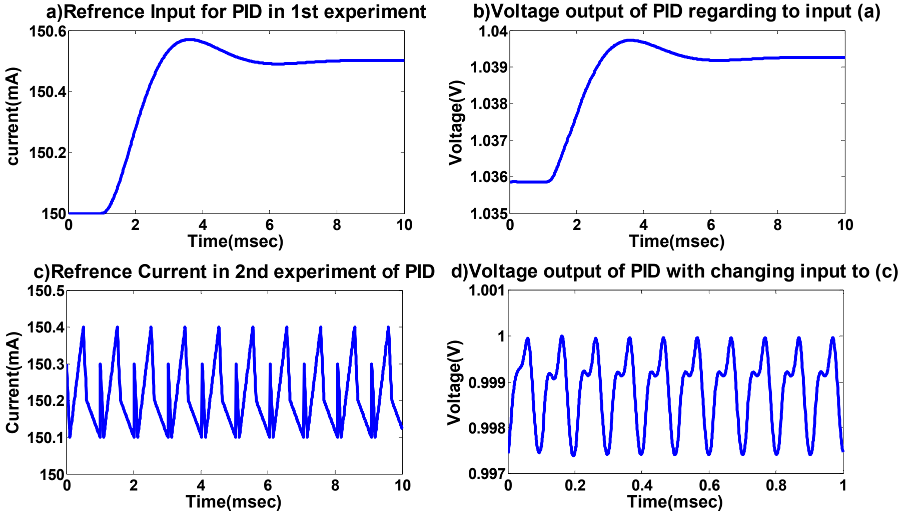


Fig. 6. Simulation of PID controller on two different input supply current signals. a,b) the supply voltage track supply current variations well. c,d) when the current variations changes as c, then supply voltage cannot track the new variations with old PID coefficients.

5. VHDL implementation of the fuzzy logic controller

The general architecture of the fuzzy logic controller to track supply current variations of a processor is shown in Fig. 7 where the information flows from left to right. The fuzzy logic controller is designed based on the Mamdani fuzzy inference system (FIS). The first step to implement the controller as a digital circuit is to convert analogue input values, supply current and its derivative, to digital ones. For this purpose an analogue to digital converter (A/D) is necessary to digitize the input crisp values. The resolution of the selected A/D depends on the desired accuracy for supply current, derivative of supply current and supply voltage data. For example, suppose that the supply current variations of a processors change between 0mA and 100mA and one has selected an 8 bit A/D. In this case, the resolution of the supply current samples is as follows:

$$2^8 = \frac{I_{max}-I_{min}}{res.(I)} \rightarrow res.(I) \cong 0.4mA \tag{4}$$

Hence if a voltage actuator e.g. a DC-DC converter has been selected to regulate the processor’s voltage between 0.7V and 1.2V, the output supply voltage made by the fuzzy logic controller has steps of 1.95 mV for supply voltage. In this section, we design the controller in VHDL based on an 8 bits resolution for digital values, and without loss of generality one can extend the design to other resolutions.

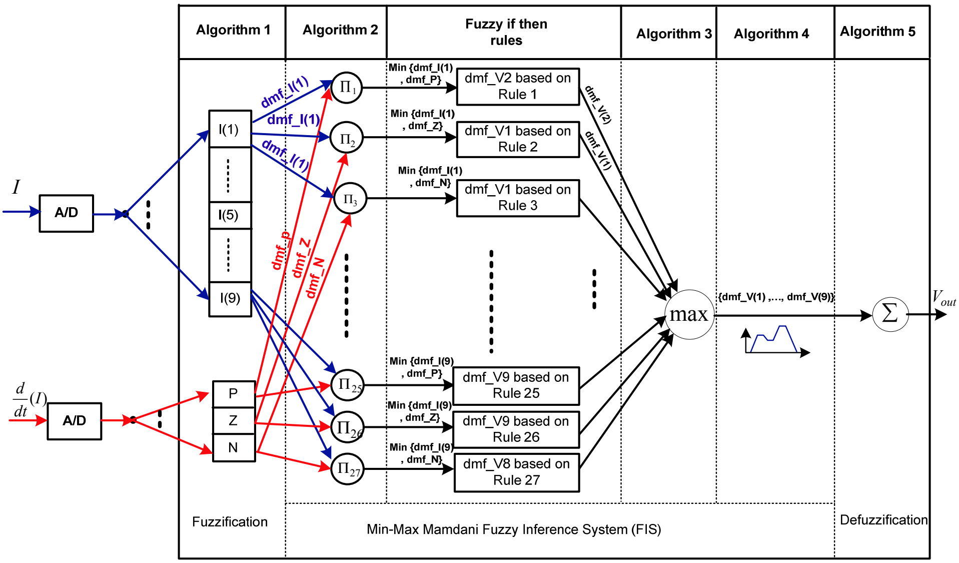


Fig. 7. Architecture of the Mamdani Fuzzy Inference System (FIS) for the supply voltage computation in VHDL implementation

5.1 Implementation of the fuzzification stage

After digitizing the input crisp values, the first step is to define membership functions for the current, derivative of the current and the supply voltage. We consider nine membership functions for the supply current variations, three membership function for its derivative, and nine membership functions for the output supply voltage. The numbers of the membership functions are obtained based on executing different experiments and evaluating the accuracy of the controller with different supply current signatures. These functions are defined in the triangular shapes like the ones presented in Fig. 4 combined with the same corresponding table of fuzzy if-then rules. First we start to design the membership functions (MFs) of the supply current. Since we have used an 8 bits resolution for A/D, the input range of current should map between 0($0) and 255 ($FF). Consider defined MFs of the current as the ones shown in Fig. 8. In this figure, the Y axis shows the degree of membership function as a value between 0 and 1 and the X axis shows the supply current universe of discourse. All these parameters need to be mapped between 0 and 255. Each MF in Fig. 8 is represented by four parameters: point1 (P1), the positive slope value, point2 (P2), and the negative slope value.

For each current value as the input, the degree of membership function (dmf) depends on the location of the current value regarding to these four parameters. The pseudo code to calculate the degree of membership function for a specific input current value is presented in Algorithm 1. In this pseudo code, it is supposed that the slopes of all triangular membership functions have value 8. With this assumption, one can avoid using multipliers in the circuit to calculate the degree of membership functions and increase the speed of the circuit.

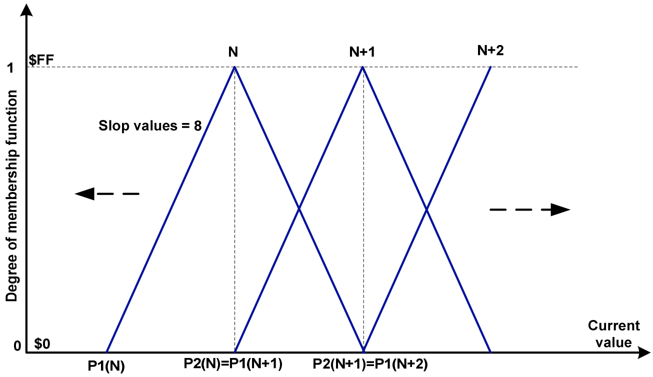


Fig. 8. Triangular membership functions for the supply current variations

To calculate the degree of membership functions of the derivative of supply current, an algorithm similar to Algorithm 1 can be used. The only differences are 1) three MFs are defined (N=3) for the derivative of current, 2) different slopes are defined for the MFs of the derivate of current as slope (MF1) =slope (MF3) =8 and slope (MF2) =16.

Algorithm 1 – Fuzzification : triangular MFs for supply current and calculate dmf

```
 Data( $P_1$, $P_2$, slope, $I\_MFS$, $dmf\_I$ : std_logic_vector(7 downto 0))
 $N$: Counter for the Number of membership function for supply current variations
 $P_1(N)$: Point 1 of the membership function N shown in Fig.8
 $P_2(N)$: Point 2 of the membership function N (middle point in each MF)
 Slope (N): 8
 I_MFS: Number of membership function for supply current variations (here: 9)
 dmf_I: Degree of membership functions
 I : input supply current
For N=1 to I_MFS
    IF I < $P_1(N)$
        dmf_I = 0
    Elseif I < $P_2(N)$
        dmf_I = (I - $P_1(N)$ ) * Slope (N)
    Elseif I < $P_2(N+1)$
        dmf_I = 255 - (I - $P_1(N)$ ) * Slope (N)
    Else I > $P_2(N+1)$
        dmf_I = 0
End
```

5.2 Rule evaluation: Implementation of fuzzy inference system

Considering Fig. 7 and since there are 9 MFs for the current and 3 MFs for its derivative, 27 fuzzy if-the rules are defined to correspondingly calculate the fuzzy supply voltage output values. The Mamdani FIS is used to evaluate the fuzzy if-then rules. To design Mamdani FIS in VHDL, let's consider the first fuzzy if-the rules defined in Fig. 7:

IF the supply current belongs to I(1) **AND** the derivative of current belongs to P (Positive) **Then** voltage is VDD(2).

For this rule, the AND operator should be applied to obtain one value out of two degree of membership functions (current and its derivative) which represents the result of the antecedent part in rule 1. This value actually represents a weighting factor for this specific rule. In the Mamdani FIS, the AND operator is a fuzzy operator to find the minimum value between two degrees of MFs: current and its derivative. The following minimum function, Algorithm 2, is used to implement the fuzzy AND operator in VHDL. In Fig. 7, this stage is called product layer which is a part of the min-max Mamdani FIS.

Algorithm 2 – Minimum Function

```
Function Minimum(a, b: std_logic_vector(7 downto 0))
variable min: std_logic_vector (7 downto 0) := (others => '0');
    Begin
        If a < b Then
            min := a;
        Else
            min := b;
        End If;
    Return min;
End Minimum;
```

-**Note**: In this function, a represents the dmf_I (degree of membership function of the current) and b represents dmf_di.

The output value of the minimum function is a value that determines the degree of MF for Vdd(2) in rule 1. For all the defined 27 rules, there may exist different degrees of MF values for each fuzzy voltage set. Therefore, one final value for each MF of the supply voltage should be determined. The min-max Mamdani FIS uses the maximum operator to calculate the final degree of the membership function for Vdd(1) to Vdd(9). The fuzzy maximum operator models one fuzzy set with the maximum values returned by the output fuzzy set of each rule. In VHDL, one can use the maximum function presented in Algorithm 3.

Algorithm 3 – Maximum Function

```
Function Maximum(a, b: std_logic_vector(7 downto 0))
variable max: std_logic_vector(7 downto 0) := (others => '0');
        Begin
                If a > b then
                        max := a;
                Else
                        max := b;
                End If;
        Return max;
End Maximum;
```

-Note: In this function, a and b represent the dmf_V values (degree of membership function of the voltage). This function should be called for each dmf_V(1), …,dmf_v(9) separately.

5.3 Implementation of the defuzzification stage

The last step is to perform the defuzzification process that converts the obtained fuzzy set into a single number as the output supply voltage. The aggregate output fuzzy set consists of a range of voltage output values and has to be defuzzified to determine a single output supply voltage value. For the defuzzification method, the centroid calculation is used to compute the final value. The centroid method computes the center of area under the curve of the fuzzy output set. From the min-max FIS, nine degrees of membership functions for each voltage set is obtained (dmf_V(1), …, dmf_V(9)). For each input value of the current and its derivative, there are a maximum of 3 *dmf* that have a nonzero value. Suppose that the aggregated output fuzzy set is as the one shown in Fig. 9.

To compute the output voltage value, as one can see from eq. (5), the following functions are needed to use: summation, multiplier and divider. Since implementing a divider block results in a circuit that occupies more area, we propose to use a look up table (LUT) stored in the memory of processor. This LUT needs to be filled out by the designer. Under this approach, the data stored in the memory estimates the center of gravity of the output fuzzy set obtained by the min-max Mamdani FIS. Here, we explain the required size of the memory and how to address and access to data in the LUT.

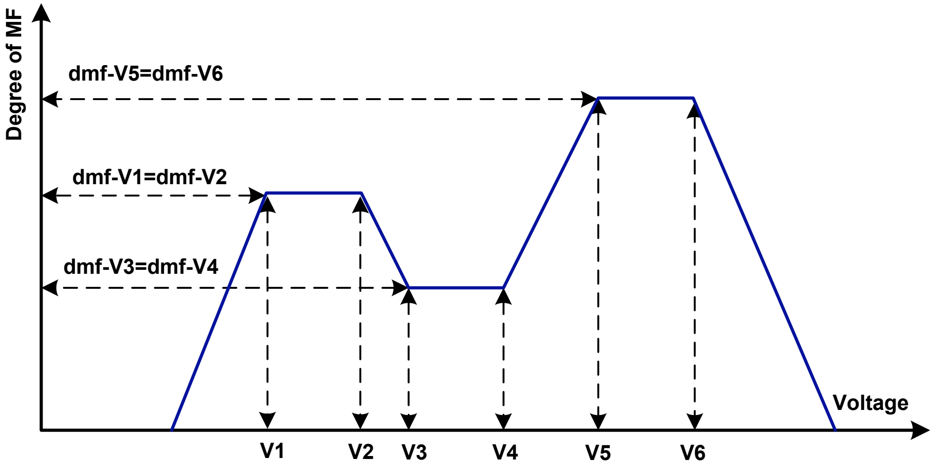


Fig. 9. Defuzzication and calculation of the final supply voltage value

If the centroid method for the defuzzification is applied, the output voltage value is as follows:

$$V = \frac{\sum_{i=1}^{6} Vi*dmf\_Vi}{\sum_{i=1}^{9} dmf\_Vi} \quad (5)$$

To track supply current variations, for each pair of fuzzy inputs (supply current and its derivative) at a specific time, there is a maximum of three adjacent membership functions MFs for the voltage which have degree of membership function *dmf* value distinct from zero. Therefore, one can use Algorithm 4 to first find those involved voltage MFs and then use the LUT to calculate the final voltage value.

Algorithm 4 – Specifying active voltage membership functions in the defuzzification stage

Data:

V_MFS =9: Number of membership function for supply voltage (here: 9)

N[1:9]: Counter for the number of membership functions of supply voltage

dmf_V [0:255]: Degree of MF

rb [0 or1]: a bit to specify which MF is involved in calculating the final voltage value

```
For N=1 to V_MFS
     IF  dmf_V(N) = 0 Then
          rb(N) = 0
     Else rb(N) = 1
     End
End
```

To construct the LUT, we only use the first 3 most significant bits (MSB) of each voltage membership function. Since there is a maximum of three membership functions involved in calculating the final crisp voltage value, one needs to consider $2^{3+3+3} = 512$ words of the memory to make the desired LUT. Suppose we want to consider the whole 8 bits of each degree of voltage membership function value, the number of words in the memory changes to 2^{24}. For now, let's assume we have considered 3 MSBs for each degree of MF. Depending on the number of the active voltage membership functions and corresponding degree of membership functions obtained by the Mamdani FIS, one can access the corresponding word in the memory to access the output voltage value stored in it. The VHDL algorithm to access the proper memory address in the defuzzification part of the designed fuzzy controller is shown in Algorithm 5.

Now each address of the memory should be filled out by a proper value to estimate the centre of gravity accordingly. We simulated all the corresponding possible situations for the aggregated fuzzy output voltage sets in MATLAB and estimate the output voltages. Then we stored all the corresponding values into the 512 bytes considered memory.

Algorithm 5 – How to access the data of the memory in the defuzzification stage

Data

address: address of the memory

N[1:9]: the number of membership function for supply voltage

rb [0 or1]: a bit to specify which MF is involved in calculating the final voltage value

Vout: output supply voltage value

```
Counter = 0
For N = 2 to V_MFS
    Counter = Counter + 1
        IF rb(N) = 1
            address = concatenate(r(N-1), r(N), r(N+1))
            break
        End IF
End For
Vout = 32*(Counter - 1) + LUT(address)
```

5.4 Synthesis results

We have implemented the proposed fuzzy logic controller in a CMOS 90nm technology and synthesized it with Cadence RC compiler to measure its power consumption and area. For benchmarking purposes, the synthesis of the circuit is done with different speeds. Synthesis specifications are mentioned in Table 1.

Synthesis and the Library specifications:
CMOS 90 nm HVT-TSMC
Supply Voltage: 1.2 V
PVT Typical corner
Temperature: 25 degree
Frequencies : {20, 40,60,80,100,200,333} MHz

Table 1. Library specification for synthesizing the fuzzy logic controller

The synthesis results are shown in Fig. 10. Since the fuzzy logic controller is a digital controller, its circuit does not consume much power and it does not occupy much area as shown in Fig. 10.

The main differences between the proposed VHDL implementation of the fuzzy controller and the other already implemented VHDL fuzzy controllers (Vuong et al., 2006; Vasantha et al., 2005; Sakthivel et all., 2010; Daijin, 2000) is about the speed of the controller. In the proposed implementation strategy, there are no multiplier and divider circuits used, and also we have considered a fixed slope value for the membership functions. For these reasons, the circuit naturally works faster. Since we have used the memory to store the defuzzification data, it is worth to mention that the power consumption of the proposed circuit is probably higher than previously reported ones.

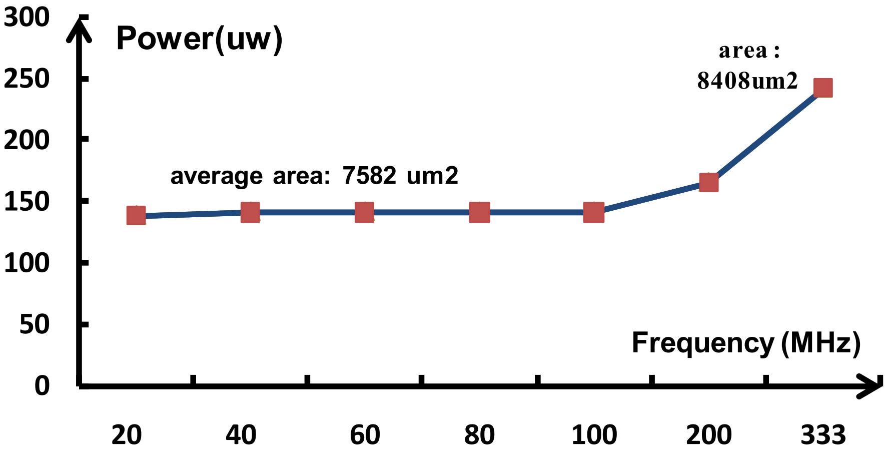


Fig. 10. Synthesis results of the fuzzy logic controller

As way of example, we test the fuzzy logic circuit with the supply current profile of a processor when it executes a MPEG2-decoding application. The output result of the fuzzy logic circuit implemented in VHDL is shown in Fig. 11. The output signal of the fuzzy controller can accurately track the supply current variations. This output signal can be used to scale and adjust the supply voltage of the processor based on current variations for dynamic voltage scaling goals. Also in Fig. 11, the simulation result of the fuzzy controller implemented in Matlab is presented.

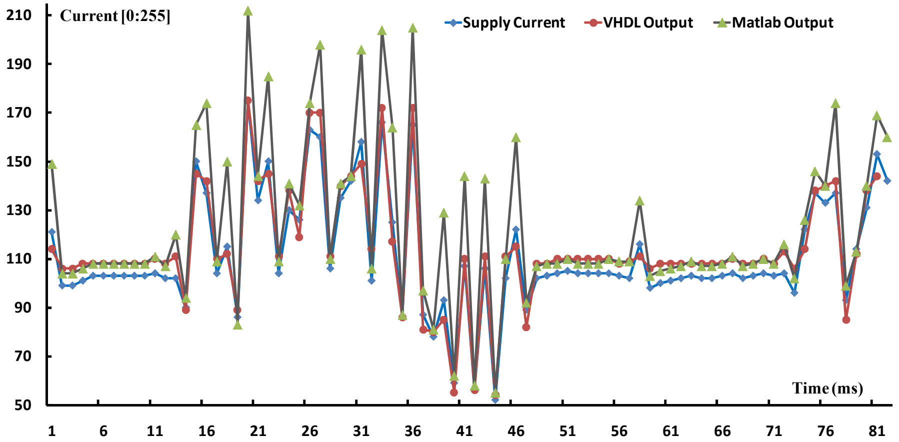


Fig. 11. Comparison between the tracking results of the implemented VHDL fuzzy circuit and Matlab simulation

6. Conclusion

In this chapter, a dynamic fuzzy logic controller based on supply-current variation tracking for dynamic voltage and frequency scaling purposes was proposed. In the proposed method, the fuzzy logic controller decides about changing the supply voltage of the circuit under control by observing and predicting the supply-current variations. The simulation results showed the effectiveness of the proposed configuration in comparison to a PID controller. Furthermore, in this chapter, we described how to implement the proposed controller in VHDL. Also a new method for implementing the defuzzification stage in VHDL was proposed. The synthesized results of the implemented fuzzy controller in a CMOS 90nm technology, using Cadence RC compiler, evaluated in this chapter based on its power consumption and area.

7. Acknowledgment

This work was supported by the Dutch Technical Science Foundation (STW), under the agreement 363120-427.

8. References

Azevedo, A.; Issenin, I.; Cornea, R.; Gupta, R.; Dutt, N.; Veidenbaum, A. & Nicolau, A. (2002). Profile-Based Dynamic Voltage Scheduling Using Program Checkpoints, *Proceedings of the conference on Design, Automation and Test in Europe (DATE 2002)*, pp. 168-175.

Bang, S.; Bang, K.; Yoon, S. & Chung, E.Y. (2009). Run-time adaptive workload estimation for dynamic voltage scaling, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* Vol.28, No.9, pp. 1334-1347.

Benini, L.; Bogliolo, A.; Paleologo, G.A. & De Micheli, G. (1999). Policy optimization for dynamic power management, *IEEE Trans. On Computer-Aided Design of Integrated Circuits and Systems,* Vol.18, No.6, pp. 813-833.

Chung, E.Y.; De Micheli, G. & Benini, L. (2002). Contents provider-assisted dynamic voltage scaling for low energy multimedia applications, *Proc. of the 2002 international symposium on Low power electronics and design*, Monterey, California, USA, pp. 42-47.

Daijin, K. (2000) An Implementation of Fuzzy Logic Controller on the Reconfigurable FPGA System, *IEEE Transactions on Industrial Electronic*, Vol.47, No.3, pp. 703 – 715.

Gu, Y. & Chakraborty, S., (2008). Control theory-based DVS for interactive 3D games, in *Proc. DAC*, 2008, pp. 740–745.

Gutnik, V. & Chandrakasan, A.P. (1997). Embedded power supply for low-power DSP, *IEEE Trans. On VLSI Syst.*, Vol.5, No.4, pp. 425-435.

Hughes, C.J. & Adve, S.V. (2003). A formal approach to frequent energy adaptations for multimedia applications, in *Proc. Int. Conf. Comput. Des.*, pp. 489–496.

Im, C.; Kim, H. & Ha, S. (2006). Dynamic voltage scheduling technique for low-power multimedia applications using buffers, *Proc. ISLPED,* pp. 34-39.

Jejurikar, R. & Gupta, R. (2006). Energy-aware task scheduling with task synchronization for embedded real-time systems, *IEEE Trans. Comput.- Aided Design Integr. Circuits Syst.*, Vol.25, No.6, pp. 1024–1037.

Lee, C.-C., (1990) Fuzzy Logic in Control Systems: Fuzzy Logic Controller-Parts 1 and 2, *IEEE Trans.on Systems, Man, and Cybernetics*, Vol.20, No.2, pp. 404-435.

Lu, Z.; Hein, J.; Humphrey, M.; Stan, M.; Lach, J. & Skadron, K. (2002). Control-theoretic dynamic frequency and voltage scaling for multimedia workloads, in *Proc. Int. Conf. Compilers, Architecture, Synthesis Embed. Syst.*, pp. 156–163.

Lu, Z.; Lach, J.; Stan, M. & Skadron, K. (2003). Reducing multimedia decode power using feedback control, in *Proc. Int. Conf. Comput. Des.*, pp. 489–496.

Nielsen, L.S.; Nielssen, C.; Sparsø, J. & Van Berkel, K. (1994). Low-power operation using self-timed circuits and adaptive scaling of the supply voltage, *IEEE Trans. VLSI Syst.*, Vol.2, pp. 391–397.

Pourshaghaghi, H.R. & Pineda de Gyvez, J., (2009). Dynamic Voltage Scaling Based on Supply Current Tracking Using Fuzzy Logic Controller, *Proc. of the 16th IEEE Int. Conf. on Electronics, Circuits, and Systems, (ICECS 2009),* pp. 779-782.

Rabaey J, (2010) *Low Power Design Essentials*. Springer, pp. 249-288.

Sakthivel, G.; Anandhi, T.S. & Natarajan, S.P. (2010) Real time implementation of a fuzzy logic controller on FPGA using VHDL for DC-Motor speed control, *International Journal of Engineering Science and Technology,* Vol.2, No.9, pp. 4511-4519.

Sinha, A. & Chandrakasan, A.P. (2001). Dynamic voltage scheduling using adaptive filtering of workload traces, *Proc. of the 14th International Conference on VLSI Design (VLSID '01)*, pp.221-226.

Vasantha Rani, S.P.J.; Kanagasabapathy, P. & Sathish Kumar, A. (2005). Digital Fuzzy Logic Controller using VHDL, *INDICON2005,* pp. 463-466.

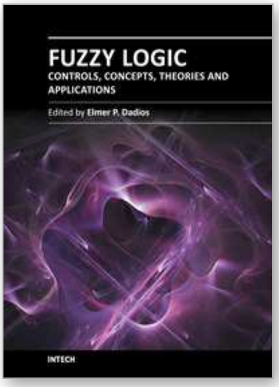
Vuong, P.T.; Madni, A.M. & Vuong, J.B. (2006). VHDL implementation for a fuzzy logic controller, *Automation Congress, 2006. WAC '06. World,* pp. 1-8.

Wei, G. & Horowitz, M. (1999). A fully digital energy-efficient adaptive power supply regulator, *IEEE Journal of solid-state Circuits*, Vol.34, No.4, pp. 520-528.

Wu, Q.; Juang, P.; Martonosi, M. & Clark, D.W. (2005). Formal control techniques for power-performance management, *IEEE Micro*, Vol.25, No.5, pp. 52–62.

Yang, P.; Wong, C.; Marchal, P.; Catthoor, F.; Desmet, D.; Verkest, D. & Lauwereins, R. (2001). Energy-Aware Runtime Scheduling for Embedded-Multiprocessor SOCs, *IEEE Design & Test*, Vol.18, No.5, pp.46-58.

Yao, F.; Demers, A. & Shenker, S. A. (1995). Scheduling model for reduced CPU energy, *Proc. Found. Comput. Sci.,* pp. 374-382.



Fuzzy Logic - Controls, Concepts, Theories and Applications
Edited by Prof. Elmer Dadios

ISBN 978-953-51-0396-7
Hard cover, 428 pages
Publisher InTech
Published online 28, March, 2012
Published in print edition March, 2012

This book introduces new concepts and theories of Fuzzy Logic Control for the application and development of robotics and intelligent machines. The book consists of nineteen chapters categorized into 1) Robotics and Electrical Machines 2) Intelligent Control Systems with various applications, and 3) New Fuzzy Logic Concepts and Theories. The intended readers of this book are engineers, researchers, and graduate students interested in fuzzy logic control systems.

Hamid Reza Pourshaghaghi, Juan Diego Echeverri Escobar and José Pineda de Gyvez (2012). Synthesis and VHDL Implementation of Fuzzy Logic Controller for Dynamic Voltage and Frequency Scaling (DVFS) Goals in Digital Processors, Fuzzy Logic - Controls, Concepts, Theories and Applications, Prof. Elmer Dadios (Ed.), ISBN: 978-953-51-0396-7, InTech, Available from: http://www.intechopen.com/books/fuzzy-logic-controls-concepts-theories-and-applications/synthesis-and-vhdl-implementation-of-fuzzy-logic-controller-for-dynamic-voltage-and-frequency-scalin

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821