VNU Journal of Science: Comp. Science & Com. Eng., Vol. 31, No. 2 (2015) 56–65

Design and Modeling of a Voltage-Frequency Controller for Network-on-Chip Routers based on Fuzzy-Logic

Hai-Phong Phan, Xuan-Tu Tran

SIS Laboratory, VNU University of Engineering and Technology,
144 Xuan Thuy road, Cau Giay district, Hanoi, Vietnam

Abstract

Network-on-Chip (NoC) paradigm allows designers to integrate efficiently more intellectual properties (IPs) into a single chip system. However, the power consumption has become one of the most critical issues for designing such large complex systems. Low power design can be achieved by scaling the voltage and frequency of the target components. The question is how to make the voltage-frequency scaling adaptable to the required performance of the system at run-time while reducing as much as possible the power consumption. In this paper, we present the design and modeling of a Voltage-Frequency Controller for NoC routers based on fuzzy-logic processing. The communication traffic of a network router will be predicted by a fuzzy-logic algorithm. Then the voltage and frequency of the router will be dynamically scaled according to the predicted results in order to get power consumption optimal for the network router. The Voltage-Frequency Controller is then modeled at register-transfer level using VHDL – a hardware description language. The most important part of the proposed controller, the fuzzy-logic processor (FLP) designed with the famous Sugeno model, has been successfully implemented on FPGA devices.

© 2015 Published by VNU Journal of Sciences.
Manuscript communication: received 20 May 2015, revised 16 June 2015, accepted 25 June 2015.
Corresponding author: Hai-Phong Phan, phongph.de12@vnu.edu.vn

Keywords: Low Power, Network-On-Chip, DVFS, Fuzzy Logic.

1. Introduction

The fuzzy set theory was first proposed by L.A Zadeh in 1965 [1]. It has been widely applied in many application fields, from control theory to artificial intelligence. The typical fields in which the fuzzy logic theory has been successfully applied include automation control, power saving, data processing, signal processing, especially in robotic design. Many researches focus on the design of robots with human behaviors based on fuzzy logic. By using fuzzy logic algorithms, the behavior of robots can be implemented through the decision "IF–THEN" which is similar to human thinking. Therefore, robots can have "thinking" and action as humans more than the previous robots generation [2, 3].

In the medical field, fuzzy logic theory has been studied and applied in many different subjects, especially in the field of biomedical signal processing. In [4], Y. C. Yeh *et al.* introduced a simple method to study the effectiveness of ECG signals using fuzzy logic theory. Reference [5] also introduced a technique based on fuzzy logic theory to handle the MRI images effectively.

Previously, fuzzy logic systems are usually implemented by software. The advantages of this method are the ability of quick deployment, easy modification, time reduction in development, and low cost. However, one of the most disadvantages is that the processing and computation speed are too slow. This affects the implementation of fuzzy logic in real-time systems.

Fuzzy logic theory has prompted the researchers to develop fuzzy logic systems with faster processing and calculating speed, and more power efficiency. One of the recent trends for developing fuzzy logic systems is to deploy the system as a hardware core to increase the processing speed of the system [6, 7, 8].

Besides, the power consumption is also an important factor in designing complex systems, especially in designing Network-on-Chip (NoC) based systems. The NoC paradigm has been recently known as an emerging solution for designing large, complex system-on-chips (SoCs) [9]. In the NoC based systems, computing units (i.e. Intellectual Properties or IPs) communicates with each other using a micro network that is composed of network routers and network links. Low power techniques for this kind of systems are based on the fact that the system never works at its maximum power capacity. There often exists some idle or low speed operating parts of the system during operating time.

In this paper, we focus on designing a voltage-frequency controller for NoC routers. The role of the proposed controller is to adjust the frequency and voltage of the target router according to its workload (the communication traffic going through). Therefore, the power consumption of each router as well as the whole system can be reduced while keeping system performance suitable. To do that, we apply a fuzzy logic algorithm using Sugeno model [10] in the controller. The design is then modeled and implemented using VHDL at register-transfer level.

The remaining part of the paper is organized as follows. Section 2 presents the proposed Voltage-Frequency Controller for NoC routers. The modeling of this VFC is discussed in Section 3. The simulation and experimental results are given in Section 4. Finally, conclusions and remarks will be provided in Section 5.

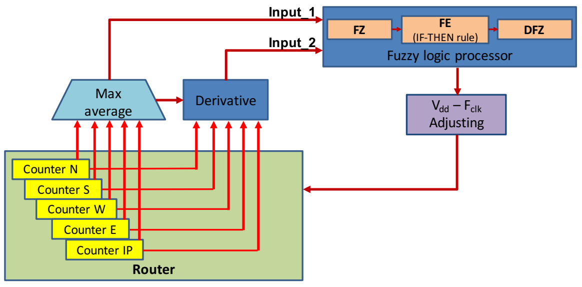

Fig. 1: The proposed Voltage-Frequency Controller.

2. The Proposed Voltage-Frequency Controller

In this paper, we assume that the traffic going through a router is also a quantity that reflects the activities of this router. If the router has a large communication traffic, it must be supplied a higher frequency, as well as a higher voltage, to meet the high data transmission rate and vice versa.

Therefore, we propose to use a voltage-frequency controller to scaling voltage and frequency of the router according to the activities of it in order to reduce the power consumption of a router in a NoC based system. To do that the controller will monitor the traffic through the router, then predict the change of traffic to make a decision to increase or decrease the values of voltage and frequency accordingly.

To simplify the structure and reduce hardware resources of the system, we propose a Voltage-Frequency Controller (VFC) as shown in Figure 1. In this controller, we use a fuzzy logic processor to predict the communication traffic and make decision about the values of frequency and voltage.

In this architecture, each input port of the target router will be equipped with a traffic counter. These counters count the data flits passing through the router in certain clock cycles (average traffic) based on the corresponding response signals from the router. Since the router normally has 5 input/output ports [11], there will be 5 communication traffic values from the router. The maximum traffic value passing through the router is then decided by the Max Average (MA) block. In fact, the MA will compare and find the maximum value of the five average traffic values given by the router. Finally, this information will be sent to the Input_1 of the FLP for being processed.

The Derivative (DER) block calculates the derivative of traffics obtained from the counters. To do that, it receives the traffic values from the counters and store these values to buffers. The derivation of traffic will be calculated by the present value and the previous value. The DER determines the derivative value of traffic according to the maximum traffic value decided by MA block and then gives it to Input_2 of the FLP for further processes.

The Fuzzy Logic Processor (FLP) will process the given information (maximum traffic value and derivative value) to predict the next communication load passing through the router and decide the suitable voltage and frequency supplied to the router. It is constructed from the three sub-blocks: the Fuzzification (FZ), the Fuzzy Engine (FE) and the Defuzzification (DFZ), as described in Figure 1. As mentioned above, the operation of FLP is based on Sugeno model to simplify the process of modeling and calculation. As a result, this leads to the reduction of hardware resources required for implementing the whole voltage-frequency controller.

The Voltage-Frequency Adjusting (VFA) block controls the voltage and the frequency supplied to the router. In this design, the router is supplied by three pairs of frequency voltage values (low, medium, high). When the frequency is changed, the voltage will be also adjusted to new level corresponding to the new frequency. The change of frequency is determined by a control signal at the output of FLP.

3. Modeling the Voltage-Frequency Controller

3.1. The Counter

The model of Counter_x is described in Figure 2. In this model, the Clock_counter block is used to count a number of the clock signal - (*clk*) - events. The Signal_counter block counts events from the *resp_in* signal (a handshaking signal at the router indicating a flit transaction complete). When the Clock_counter reach a fixed value, the number of events in Signal_count block is sent to the output of Counter_x as a value of traffic.

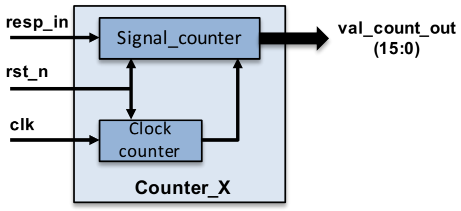

Fig. 2: The model of Counter_x.

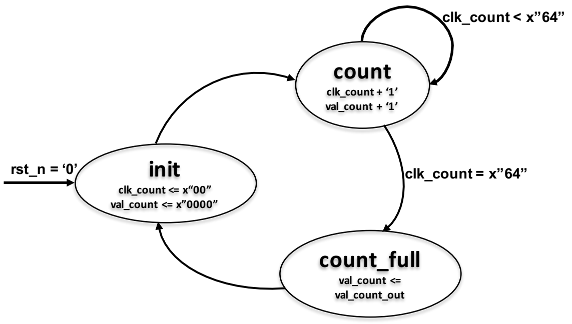

Fig. 3: The finite state machine of Counter_x.

The activities of Counter_x are modeled as a finite state machine (FSM) with three states: *init_st*, *count_st* and *count_full_st* (Figure 3). In the *init_st* state, all of signals will be reset to the initial values. The next state of the *init_st* will be *count_st* state. The *count_st* state will count the events of signal *resp_in* and signal *clk*. If the number of *clk*'s events equals 0x64, the next state of FSM is *count_full_st* state. In the *count_full_st* state, number events of *resp_in* signal will be sent to the output as a value of traffic and FSM come back to the *init_st* state. At this state, a signal (*end_count*) is also sent to the DER to warn that a counting process has been finished.

3.2. The Max Average

The MA block receives traffic values from the Counter_x at five ports of the router. By comparing those values, this block will chose the maximum value between them and send it to *input_1* of the FLP. Simply, the structure of this block is the combination of 16-bit comparators (COMP) as in Figure 4.

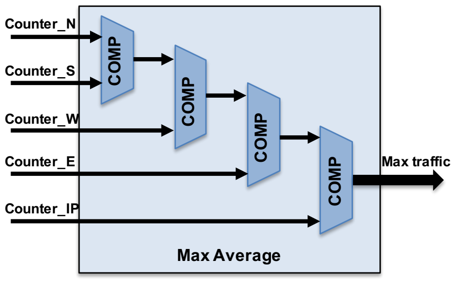

Fig. 4: The diagram of Max Average block.

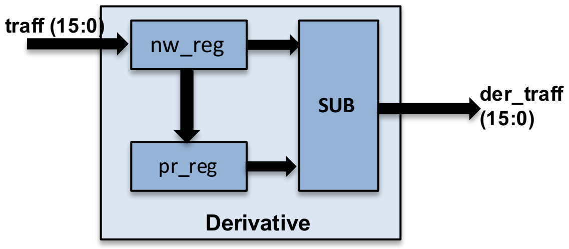

Fig. 5: The block diagram of the Derivative.

3.3. *The Derivative*

The main structure of the DER is composed of two 16-bit registers. One register stores the traffic value at present time. The other one stores the value of a frame of time before. The derivative of traffic is calculated as an absolute of subtraction between those registers.

The source code of Process describes the DER block as below:

```
der_process : process (end_in , rst_n) is
begin
if rst_n = '0' then
 dev_traf_out <= x"0000";
 reg_nx <= x"0000";
 reg_pr <= x"0000";
elsif end_in'event and end_in='1' then
 reg_pr  <= val_traf_in;
 reg_nx <= reg_pr;
 if reg_pr > reg_nx then
  dev_traf_out <= reg_pr - reg_nx;
 else
  dev_traf_out <= reg_nx - reg_pr;
 end if;
end if;
end process der_process;
```

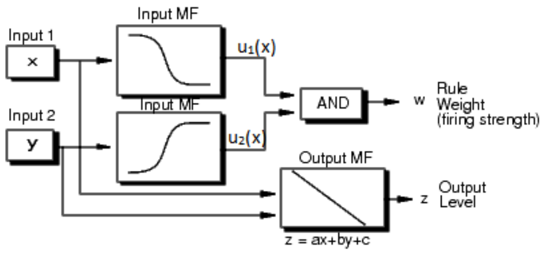

Fig. 6: The operations of a Sugeno model.

3.4. *The Fuzzy-Logic Processor*

3.4.1. *An introduction about Sugeno-type fuzzy inference*

Model Sugeno, or Takagi-Sugeno-Kang, is one of methods of fuzzy inference. Introduced in 1985 [10], it is similar to the Mamdani method in many respects. The first two parts of the fuzzy inference process, fuzzifying the inputs and applying the fuzzy operator, are exactly the same. The main difference between Mamdani and Sugeno is that the Sugeno output membership functions are either linear or constant.

A typical rule in a Sugeno fuzzy model has the form:

If *Input_1* = x and *Input_2* = y, then Output is $z = ax + by + c$

For a zero-order Sugeno model, the output level z is a constant ($a = b = 0$).

A Sugeno rule operates as shown in the diagram depicted in Figure 6.

The fuzzification converts a clean value of input to a fuzzy value based on the membership functions (MSF). This value is characterized by the degree of MSF - $\mu(x)$ and depends on the shape of MSF, the number of partitions of MSF, and the correlation membership functions.

For calculating the degree of MFS, different shapes of membership functions have been already proposed. Trapezoidal, Gaussian, triangular are some of the common shapes for membership functions.

The output level z_i of each rule is weighted by the firing strength w_i of the rule. For example, for an AND rule with *Input_1* = x and *Input_2* = y, the firing strength is

$$w_i = AND_Rule(\mu_1(x), \mu_2(y)) \qquad (1)$$

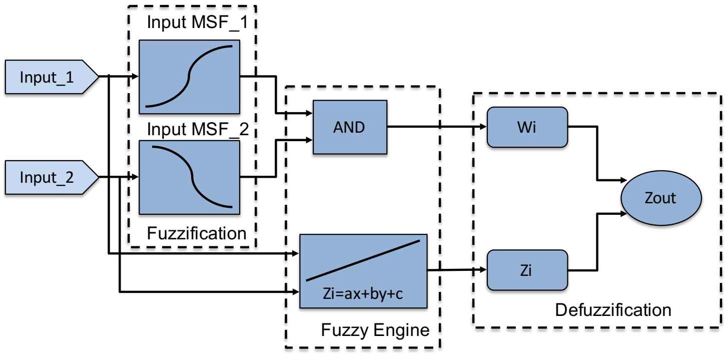

Fig. 7: Fuzzy-logic processor (FLP) model.

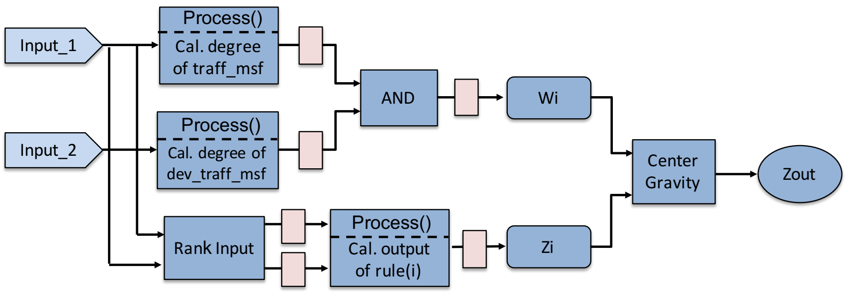

Fig. 8: Processing diagam of the fuzzy-logic processor.

The final output of the system is the weighted average of all rule outputs, computed as

$$Final\_Output = \frac{\sum_{i=1}^{n} w_i.z_i}{\sum_{i=1}^{n} w_i} \quad (2)$$

where N is the number of rules.

3.4.2. *Modeling the Fuzzy-Logic Processor*

The proposed Fuzzy-Logic Processor (FLP) is a fuzzy logic system with two inputs and an output based on Sugeno model. The FLP is implemented by three blocks as depicted in Figure 7.

- The Fuzzification (FZ) block is composed of two sub-blocks: input_MSF1 and input_MSF2. Each sub-block is a process to calculate the degree of each input (*input_1* and *input_2*) based on the membership functions.
- The Fuzzy Engine (FE) block includes two sub-blocks: the AND-rule is used for calculation of the firing strength w_i and the Z_i is used to calculate the output level z_i of each rule.
- The Defuzzification (DE) block is a process to calculate the final output of system based on the weighted average of all rule outputs.

The processing diagram of the FLP is described as Figure 8. More detail about the design of the FLP has been presented in [12].

3.4.3. *Fuzzification*

To simplify the explanation of MSFs mathematical formulation, and without loss of generality, let us consider only triangular/trapezoidal MSF.

A trapezoidal MSF is described by a set of parameters (*point_1, b, c, point_2*) as in Figure 9. The triangle MSF is the simplification of trapezoidal MSF when the parameters *point_2* and *b* are the same.

To describe the triangle MSF easily, the set of parameters becomes (*point_1, slope_1, point_2,*

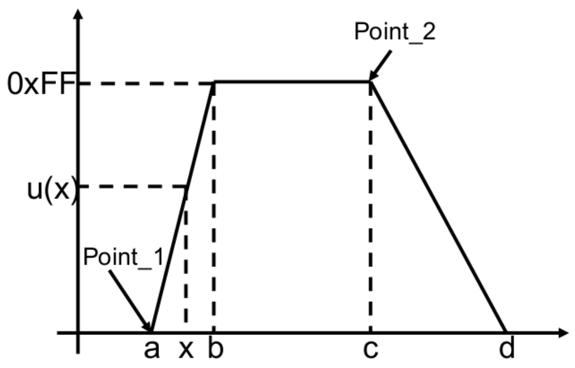

Fig. 9: The description of trapezoidal membership functions.

slope_2). The degree of membership is calculated as below.

- If *input_1* $\in [0, a]$ or *input_1* $> d$ then $\mu(x) = 0$.
- If *input_1* $\in [b, c]$ then $\mu(x) = 1$
- If *input_1* $\in [a, b]$ or *input_1* $\in [c, d]$ then

$$\mu(x) = \begin{cases} \text{(input\_1 - point\_1)} * \text{slope\_1}, \\ \qquad \textit{if input\_1} < \textit{point\_2} \\ \text{0xFF - (input\_1 - point\_2)*slope\_2}, \\ \qquad \textit{if input\_1} > \textit{point\_2} \end{cases} \tag{3}$$

The degree of membership is discrete with 8-bit resolution. Therefore, $\mu(x) = 1$ equals 0xFF in hexadecimal. In VHDL, the MSF is described as a set of parameters by a Record data type as below: The fuzzification of the traffic values at

```
type traffic_type is (term_of_mfs);
type traffic_membership is
record
term: traffic_type;
point_1: std_logic_vector(7 downto 0);
slope_1: std_logic vector(7 downto 0);
point_2: std_logic_vector(7 downto 0);
slope_2: std_logic_vector(7 downto 0);
end record;
```

input_1 is implemented through five membership functions named: *vlow, low, medium, high, vhigh*.

The shape of membership functions used in this model is the triangular. The values of parameters are described in Figure 10. The maximum value of the membership function *vhigh* is 0xC0, corresponding to the maximum value of traffic is $192 Mflits/s$. This value is selected in accordance with the maximum communication speed of a router has been designed in [11], approximately $180 Mflits/s$. The values of parameters *slope_1* and *slope_2* are 0x08 for both slopes of all membership functions. The purpose is to ensure that the calculation will be done with minimum error possibility.

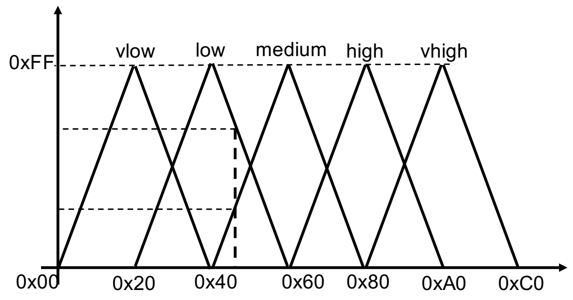

Fig. 10: The membership functions of *input_1*.

The value of *input_2* is the derivative of maximum traffic going through the router. This value is defined as an absolute value of the traffics change in a unit of time. The degree of membership at *input_2* is calculated through three membership functions with linguistic variables respectively: *slow, normal, fast*. The membership functions are also triangular functions, which are described in Figure 11. The maximum value of the MSF *high* is 0x3C. This value is corresponded to the maximum value of traffic variability $60 Mflits/s^2$. The values of *slope_1* and *slope_2* of the membership functions are 0x11.

The output Z_{out} of FLP is a constant value, corresponding to the operating frequency of the router. Therefore, the membership functions of output are singletons. The output is described by three membership functions: *low*, *normal*, and *high*. The membership functions of output are described as 8-bit constants in VHDL (Figure 12).

3.5. Evaluation rule

By applying the Sugeno model, an evaluation rule usually takes the IF-THEN statement as

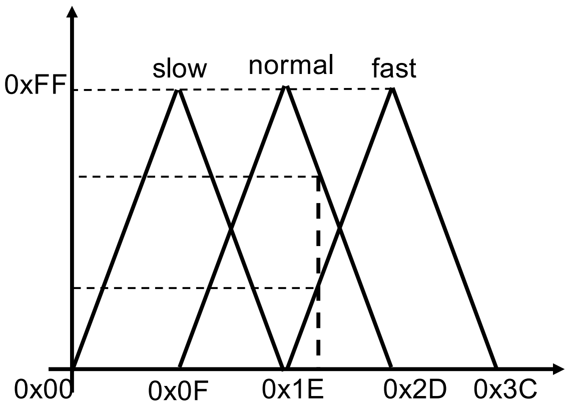

Fig. 11: The membership functions of *input_2*.

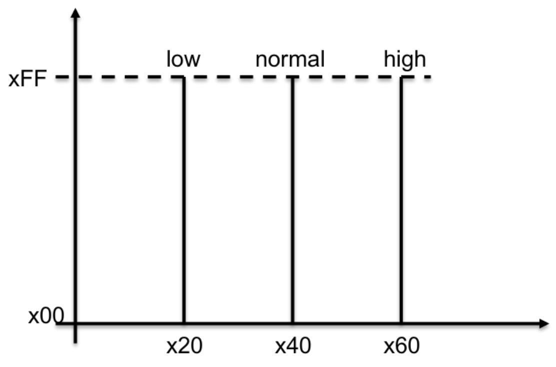

Fig. 12: The membership functions of *output*.

follows:

"**IF** *input_1* = *x* **AND** *input_2* = *y* **THEN** *Output* = *ax* + *by* + *c*"

In the case $a = b = c = 0$ and the output is a constant, we have a Sugeno model with zero order. In this model, the value of output z_i is characterized by the firing strength w_i of each rule. This firing strength is based on the rule evaluations of the combination of linguistic variables. Assuming that, we apply the AND rule with *input_1*=x and *input_2*=y then this value is

$$w_i = MIN(\mu(x), \mu(y)) \tag{4}$$

The calculation of the firing strength w_i is done by AND-rule block in the proposed model. This process is described by a source code to find the minimum value as below:

In this source code, *u_x* and *u_y* are the MSF's degree of *input_1* and *input_2*.

```
rule_weight: process (u_x, u_y) is
  variable i, j: integer;
begin
  for i in 1 to n loop
    for j in 1 to m loop
      if (u_x(i) < u_y(j)) then
        rule_w(i,j) <= u_x(i);
      else
        rule_w(i,j) <= u_y(j);
      end if;
    end loop;
  end loop;
end process rule_weight;
```

Table 1: Evaluation Rules

Traffic	Der_traffic	Frequency
vlow	slow	low
low	slow	low
medium	slow	low
high	slow	normal
vhigh	slow	normal
vlow	normal	low
low	normal	low
medium	normal	normal
high	normal	high
vhigh	normal	high
vlow	fast	normal
low	fast	normal
medium	fast	high
high	fast	high
vhigh	fast	high

With the membership functions described in Section 3.4.3, we have total 5×3 evaluation rules as in Table 1.

3.6. Defuzzification

The defuzzification is a process to calculate the exact values of FLPs output. With the output value of each rule z_i and the firing strength value of the rule w_i, the final output of the FLP is the weighted average of all rule outputs, which is computed as:

$$Z_{out} = \frac{\sum_{i=1}^{n} w_i.z_i}{\sum_{i=1}^{n} w_i} \tag{5}$$

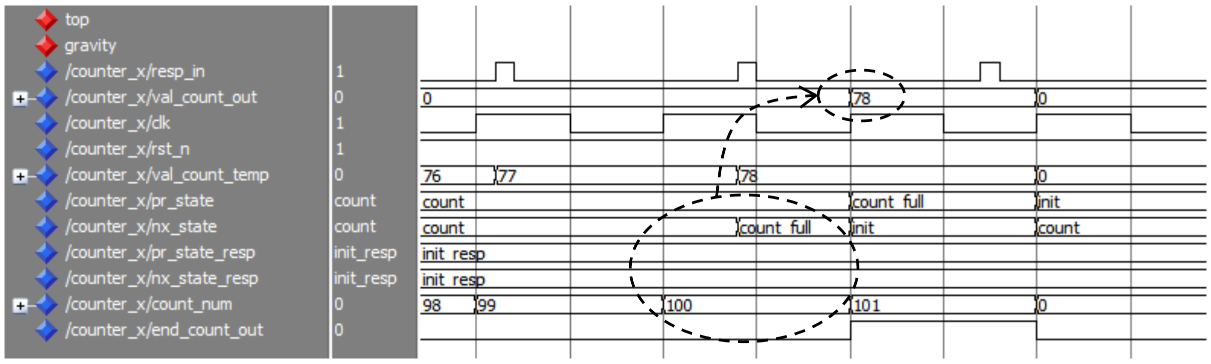
top
gravity
/counter_x/resp_in 1
/counter_x/val_count_out 0
/counter_x/clk 1
/counter_x/rst_n 1
/counter_x/val_count_temp 0
/counter_x/pr_state count
/counter_x/nx_state count
/counter_x/pr_state_resp init_resp
/counter_x/nx_state_resp init_resp
/counter_x/count_num 0
/counter_x/end_count_out 0
0
78
0
76
77
78
0
count
count_full
init
count
count_full
init
count
init_resp
init_resp
98
99
100
101
0

Fig. 13: The simulation waveform of the Count_x.

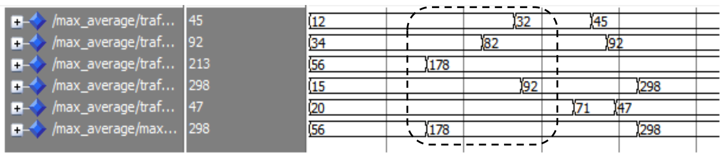

Fig. 14: The simulation waveform of the MA.

Where *n* is the number of rules. The VHDL code of this calculation is described as below:

```
for i in 1 to n loop
 for j in 1 to m loop
  w_t:= unsigned(w(i,j));
  z_t:= unsigned(z(i,j));
  upper:= upper + (w_t*z_t);
  lower:= lower + w_t;
 end loop;
end loop;
z_out:= divide(upper,lower);
```

Where *n* and *m* are the numbers of MSFs of at *input_1* and *input_2*.

4. Simulation and Implementation Results

After all of blocks of the controller had been modeled at RTL level, we simulate the operators of those blocks by ModelSim software. A testbench of each blocks will generate the random data for inputs. By observing the simulation waveforms and comparing the simulation results with the calculation results, we can conclude about the operations of those blocks.

Figure 13 is a short waveform of the Counter_x. We can see, when the number of clock events reaches value 100, then the next state of FSM will be *count_full_st* state and the output *val_count_out* gets the value 78.

One of simulation results of the MA is shown in Figure 14. With five values of inputs, the output returns value 178 – the maximum value between those values.

Figure 15 shows the simulation results of the DER. In the dash-line rectangle, we see the value of the input is 4161. The last value of input is 8449, so the result is an absolution of subtraction, equal to 3824.

The simulation of the FLP is shown in a short waveform as in Figure 16. In this waveform, we can see when the value of each input is 0x28 and then the output has a value of 0x4F. This result is accordant with the calculation results. All testing results have proved that the operations of FLP are in accordance with the proposed model.

After being successfully modeled and verified, the FLP has been implemented on FPGA devices (Spartan 3E-xc3s500e-5vq100) by using Xilinx ISE tool suite. The implementation results are described in Table 2.

As shown in the Table 3, the implementation overhead of our proposed architecture is slightly less than the existing ones (there is no ROM block in our design) while the output resolution

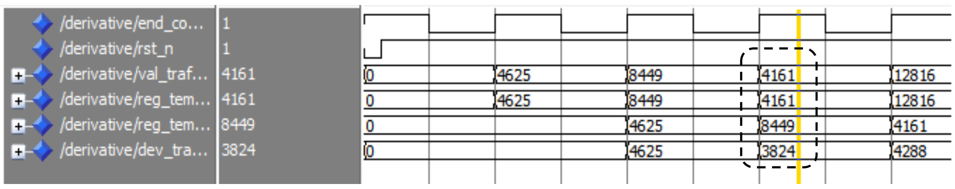
/derivative/end_co... 1
/derivative/rst_n 1
/derivative/val_traf... 4161
/derivative/reg_tem... 4161
/derivative/reg_tem... 8449
/derivative/dev_tra... 3824

Fig. 15: The simulation waveform of the DER.

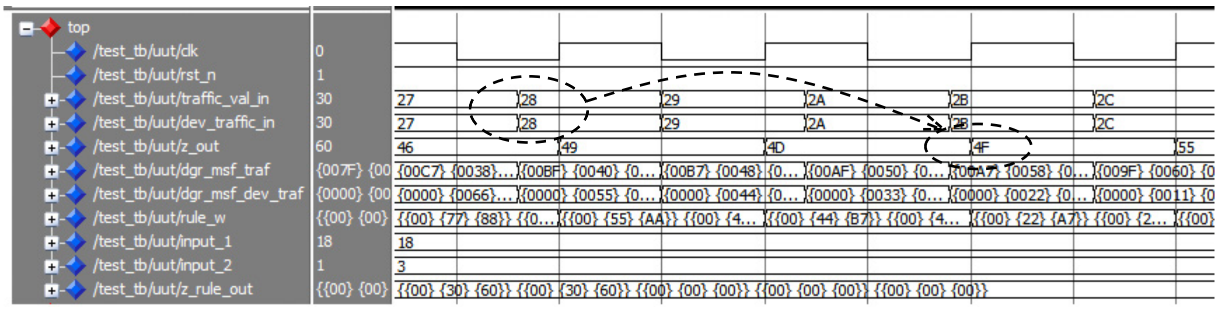

Fig. 16: The simulation waveform of the FLP.

Table 2: Implementation results on FPGA devices (Spartan 3E-xc3s500e-5vq100).

Logic Utilization	Used	Available	Utilization
Slices	711	4656	15%
Flip flop Slices	197	9312	2%
4 input LUTs	1325	9312	14%
Bonded IOBs	26	66	39%
Number of MULT 18X18SIOs	19	20	95%
GCLKs	1	24	4%

Table 3: Comparison with some related works

	This work	DFLC with arithmetic MF generator [7]	DFLC with ROM MF generator [13]
Model	Takagi Sugeno zero-order	Takagi Sugeno zero-order	Takagi Sugeno zero-order
Inputs	2	2	2
Inputs resolution	8 bits	8 bits	8 bits
Output	1	1	1
Output resolution	16 bits	12 bits	12 bits
Number of fuzzy inference rules	15	49	49
Defuzzification method	Weighted average	Weighted average	Weighted average
Slice flip flops	197	344	238
4 input LUTs	1325	406	419
Bonded IOBs	26	30	30
16x1 ROMs	0	0	128
MULT 18x18SIOs	19	6	3
GCLKs	1	2	2
Maximum Frequency	209MHz	200MHz	200MHz

is higher (16 bits instead of 12 bits). However, in this design we are using more multiplications than the other works. These multiplications can be replaced by shifts and additions to reduce the area cost in the future. In addition, as the FLP takes 03 clock cycles to complete a calculation, with the maximum frequency 209MHz the calculation speed of FLP is about 69 million calculations per second.

5. Conclusions

In this paper, the design of a voltage-frequency controller using fuzzy logic algorithm for NoC routers has been presented. The controller analyzes the communication traffic and the variation of this traffic at the target network router and makes decision to increase/ decrease the frequency and voltage applied to the router. The design of the voltage-frequency controller using VHDL is also presented. Some main simulation and implementation results have been presented and discussed.

Acknowledgment

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2013.17.

References

[1] L. Zadeh, Fuzzy sets, Information and Control 8 (3) (1965) 338 – 353.

[2] D. H. Song, S. Jung, Geometrical analysis of inverse kinematics solutions and fuzzy control of humanoid robot arm under kinematics constraints, in: Proceedings of the 2007 International Conference on Mechatronics and Automation (ICMA), 2007, pp. 1178–1183. doi:10.1109/ICMA.2007.4303715.

[3] D. Song, S. Jung, Neural compensation technique for fuzzy controlled humanoid robot arms: Experimental studies, in: Proceedings of the IEEE 22nd International Symposium on Intelligent Control (ISIC), 2007, pp. 424–429.

[4] Y.-C. Yeh, W.-J. Wang, C. W. Chiou, Heartbeat case determination using fuzzy logic method on ECG signals., International Journal of Fuzzy Systems 11 (4).

[5] A. Lakdashti, M. S. Moin, K. Badie, Reducing the semantic gap of the MRI image retrieval systems using a fuzzy rule based technique, International Journal of Fuzzy Systems 11 (4) (2009) 232–249.

[6] P. Vuong, A. Madni, J. Vuong, VHDL implementation for a fuzzy logic controller, in: Proceedings of the 2006 World Automation Congress (WAC), 2006, pp. 1–8.

[7] K. Deliparaschos, F. Nenedakis, S. Tzafestas, Design and implementation of a fast digital fuzzy logic controller using FPGA technology, Journal of Intelligent and Robotic Systems 45 (1) (2006) 77–96.

[8] S. Uppalapati, D. Kaur, Design and implementation of a mamdani fuzzy inference system on an FPGA, in: Proceedings of the 2009 annual Meeting of the North American on Fuzzy Information Processing Society (NAFIPS), 2009, pp. 1–6.

[9] W. Dally, B. Towles, Route packets, not wires: On-chip interconnection networks, in: Proceedings of the 2011 Design Automation Conference (DAC), 2001, pp. 684–689.

[10] M. Sugeno, An introductory survey of fuzzy control, Information Sciences 36 (1) (1985) 59–83.

[11] N.-K. Dang, T.-V. Le-Van, X.-T. Tran, FPGA implementation of a low latency and high throughput network-on-chip router architecture, in: Proceedings of the 2011 International Conference on Integrated Circuits and Devices in Vietnam (ICDV), 2011, pp. 112–116.

[12] H.-P. Phan, X.-T. Tran, A fuzzy-logic based voltage-frequency controller for network-on-chip routers, in: Proceedings of the 11th Conference on PhD Research in Microelectronics and Electronics (IEEE PRIME), IEEE, Glasgow, Scotland, 2015.

[13] K. Deliparaschos, F. Nenedakis, S. Tzafestas, A fast digital fuzzy logic controller: FPGA design and implementation, in: Proceedings of 10th IEEE Conference on Emerging Technologies and Factory Automation (ETFA), Vol. 1, 2005, pp. 4 pp.–262.