Center of Largest Area Defuzzifier Vnit VLSI Architecture
Asim Majeed Murshid , Ahmed Chalak Shakir , Sarmed Nihad Mohammed
Department of Computer Science , College of Science , University of Kirkuk , Kirkuk , Iraq
Asim.majeed2001@gmail.com

Abstract
Defuzzification unit combines fuzzy variables and fuzzy decisions to form a corresponding real (crisp or non-fuzzy) signal which can be used then to actuate some processes. Since the fuzzy data cannot be used directly for the real time applications, therefore it has to be converted into a crisp value. In designing the defuzzification unit in this work, the center of largest area has been used, because of its computationally efficient nature. The Center of Largest Area (COLA) of defuzzification is simple and is being generally used in comparison with more complex center of sums defuzzification method. The proposed architecture has been modeled in Very High Description Language (VHDL) and implemented in XILINX and Spartan field programmable gate arrays (FPGA). The proposed architecture is more efficient in area and the speed of operation in comparison with more complex architecture used for the Center of sums. The functional analysis has revealed that the proposed architecture is implementing COLA based defuzzifier efficiently and accurately.

Keywords: Defuzzification; Fuzzy processor; Center of largest area; VLSI design.

Introduction
The modern concept of fuzzy sets was introduced by Lotfi Zadeh in his work "Fuzzy Sets" which described the mathematics of fuzzy set theory nearly three decades ago. Although a relatively new theory, fuzzy logic has been used in many engineering applications because being considered as a simplistic solution available for the specific problems. In 1920, Łukasiewicz extended the conventional bi-valued logic of Aristotle and proposed a new tri-valued logic in his paper entitled "On three-valued logic"[1-2]. He added a new truth value to the truth logic 0 and false logic 1 and can be best translated as "possible," with a numerical values assigned between True and False [3-5].

The advantages of fuzzy logic over the traditional solutions is that it allows computers to be able to reason more like humans, respond effectively to complex inputs to deal with notations like "too hot", "too cold", "too high" or "just right". Further, there is no need of advanced mathematics or theory to develop or implement a fuzzy logic system. The fuzzy data obtained from the fuzzification process is not suitable for the real time applications and have to be converted into crisp form. The conversion of data from fuzzy form to crisp form is known as the defuzzification, also called as "rounding off". The defuzzification has the capability to reduce a fuzzy quantity to a crisp single valued quantity. It reduces the collection of membership function values into a single quantity [6-9].

Related works
The various fuzzy systems are realized by different researcher for different applications. The original digital realization of fuzzy inference processor was performed by Toga and Watanabe [10-11]. H. Peyravi et al. [12] have proposed reconfigurable inference engine for the analog fuzzy logic controller, based on Mamdani inference technique. J.M. Jou et al. [13], R. d’Amore [14] and N. E. Evmorfopoulos et al. [15] have proposed different architecture for the fuzzy inference processor. A significant improvement is reducing power and reducing redundancy has been obtained in these structures. The inference engine performance is an important issue which needs to be addressed.

Many researchers have done a work on the defuzzification fuzzy processors. Roberto d’Amore et al [12] has developed a two input one output bit scalable architecture for fuzzy processors. In this work, a model has been developed from which the defuzzified value has been obtained manually using the COLA technique initially. An architecture has been designed and developed for the model and the functional analysis has been performed using VHDL and implemented in Vertex-4 field programmable gate array (FPGA). In this work, three models have been designed to test the functionality of the proposed fuzzy processor. It has been observed that a complete match exists between the manual calculation of defuzzified values and the values calculated by using the proposed architecture. This shows that the architecture developed is calculating the defuzzified values correctly and efficiently.

Defuzzification process
The conversion of data from fuzzy form to crisp form is known as the defuzzification, also called as "rounding off". It reduces the collection of membership function values into a single quantity. The different defuzzification methods used in literature are, (1) Max membership principle. 2. Centroid method. 3. Weighted average method. 4. Mean-max membership. 5. Centre of sums. 6. Center of largest area. 7. First of maxima or last of maxima. These methods have their own applications, advantages and disadvantages [1-4].

The researchers have developed various architectures of defuzzifiers depending on what application the fuzzy processor or the fuzzy controller is being designed for. Figure 1 shows the block diagram of a defuzzifier circuit. C1X1, C1X2, C1X3, C1X4, C2X1, C2X2, C2X3, C2X4, C2X4, HC1, HC2, HC4 are the fuzzy inputs to the defuzzifier, which comprises of elements and
The defuzzifier block is having an architecture intact based the defuzzification techniques as mentioned above to extract the defuzzified or the crisp value. The crisp output value only can be used to control various processes or mechanisms. The defuzzification technique used in this work is the Center of largest area.

This method uses the overall output or union of all individual output fuzzy sets, also is the most accurate method among all. The researchers have generally designing the Very Large Scale Integration (VLSI) architecture of the COLA defuzzifier; however, in this work an efficient COLA based defuzzifier is designed and implemented. Equation 1 shows the model for the COLA based defuzzification [1-4].

\[ Z^* = \frac{\int \mu(x) dx}{\int \mu(x) dx} \quad (1) \]

In this work, we have used a model to study and realize the defuzzification architecture as shown in Figure 2. In this model two rules are being fired and Mamdani implication has been used for inference. We have used aggression of rules, as the overall consequent is being obtained from the two individual consequent in each set. In each set, the minimum of two antecedent membership values is propagated to the consequent and truncates the membership function for the consequent of each rule.

The truncated membership functions from each rule are aggregated according to the following equation used for conjunctive system of rules [3].

\[ \mu_y(y) = min(\mu_{y1}(y), \mu_{y2}(y), \ldots, \mu_{yR}(y)) \quad (2) \]

Each rule comprises of two antecedents (A1, B1 and A2, B2) and one consequent (C1, C2) and is represented as

\[
\text{IF}(X \text{ is } A1) \ \text{AND} \ (Y \text{ is } B1) \ \text{then} \ (Z \text{ is } C1) \quad (3) \\
\text{IF}(X \text{ is } A2) \ \text{AND} \ (Y \text{ is } B2) \ \text{then} \ (Z \text{ is } C2) \quad (4)
\]

In MODEL 1, as shown in Figure 2, FA1 and FB1 are the first and second fuzzy antecedents of the first rule, respectively, and C1 refers to the fuzzy consequent of that fuzzy rule. Since; the antecedents are connected by logical "AND", therefore, the minimum membership value of the antecedents propagate through to the consequent and truncates the membership function for the consequent of each rule. Similarly, FA2 and FB2 are the first and second fuzzy antecedents of the second rule, respectively, and C2 refers to the fuzzy consequent of the second fuzzy rule. This process is the inference and it follows Mamdani’s implication method, which is the most common in practice and in the literature. The union of two consequents C1 and C2 is shown in row 3 of Figure 2. From this union the defuzzified value is being obtained by using the Center of Largest Area (COLA) technique, as given in equation 1.

Calculation of defuzzified values

The defuzzified or crisp value for MODEL 1 can be calculated first manually and then using the proposed architecture. It is important that the two values must match; otherwise the proposed architecture is not well designed. In this study, Center of Largest Area (COLA) is being used to calculate the defuzzified value. Figure 3 again shows the defuzzification for the MODEL 1 as given by Figure 2. The application of equation 1 to Figure 3 will generate the defuzzified value of model 1. The defuzzified value obtained out of COLA defuzzifier (Y_{COLA}) is again given by equation (1’).

The calculations go as follows

\[ Y_{COLA} = Z^* = \frac{CR1 + AR1 + CR2 + AR2}{AR1 + AR2} \quad (1’) \]

Where CR1 and CR2 are the Center of region R1 and R2 respectively, AR1 and AR2 are the Area of region R1 and R2 respectively.

Figure 1: Block diagram of a defuzzifier
Model (1):

- **Generator C1**

- **Generator C2**

- **Adding C1 & C2**

**Figure 2:** Fuzzification, Inference and Defuzzification using Center of Largest Area for model 1

**Figure 3:** Defuzzification for MODEL 1

In the center of largest area method we first determine the areas of the two individual convex fuzzy output sets. After selecting Largest area we find the center of largest area.

Area of trapezoid = \( \frac{1}{2} \times \text{height} \times (L1 + L2) \)

Hence, \( L1 \) is longer than \( L2 \) in trapezoid C1 or C2.

Where \( L1 = (C1X4 - C1X1) \) and \( L2 = (C1X3 - C1X2) \) for trapezoid C1, therefore;

\[
L1 + L2 = [(C1X4 + C1X3) - (C1X1 + C1X2)] = [C1X4 - C1X3 - C1X1 - C1X2] = C1X
\]

Meaning \( L1 + L2 = C1X \)

Where C1X1, C1X2, C1X3, C1X4 and C2X1, C2X2, C2X3, C2X4 are the representation point of trapezoid MFs C1 and C2 respectively, also HC1 and HC2 are the Height value of trapezoid MFs C1 and C2 respectively.
Height of C1 \( (HC_1) = 0.5 = A \ H. \)
C1X1 = 0, C1X2 = 2, C1X3 = 6, C1X4 = 8
Center of C1 = CC1 = \( \frac{C1X4 - C1X1}{2} + C1X1 = \frac{8 - 0}{2} + 0 \)
= 4.
C1X3 = C1X3 + C1X4 = 6 + 8 = 14, C1X12 = C1X1 + C1X2 = 0 + 2 = 2
C1X = C1X34 = C1X12 = 14 - 2 = 12
VC1 = \( \frac{C1X}{2} = \frac{12}{2} = 6. \)

Area of trapezoid C1 = height \( \times \frac{L1 + L2}{2} = HC1 \times \frac{C1X}{2} \)
= HC1 \( \times \) VC1
AC1 = HC1 \( \times \) VC1 = 10 \( \times \) 6 = 60.
height of C2 = HC2 = 0.25 = 5H.
C2X1 = 6, C2X2 = 7, C2X3 = 13, C1X4 = 14

Height of C2 \( (HC_2) = 0.5 = A \ H. \)
C2X1 = 6, C2X2 = 8, C2X3 = 12, C1X4 = 14
Center of C2 = CC2 = \( \frac{C2X4 - C2X1}{2} + C2X1 = \frac{14 - 6}{2} + 0 \)
+ 6 = 10.
C2X3 = C2X3 + C2X4 = 12 + 14 = 26, C2X12 = C2X1 + C2X2 = 6 + 8 = 14
C2X = C2X34 - C2X12 = 26 - 14 = 12.
VC2 = \( \frac{C2X}{2} = \frac{14}{2} = 7. \)

AC2 = HC2 \( \times \) VC2 = 10 \( \times \) 7 = 35.
AC1 > AC2 = 60 > 35
\n\n\textit{\textbullet} \ C2 is Largest Area
\textit{\textbullet} Center of Largest Area is Center of C2 (CC2) = 10.

The same procedure has been used for other defuzzified model, not shown in a figure. The defuzzified value obtained is 5H, all results of models 1, 2 and 3 as shown in table 1.

\textbf{Hardware realization of the defuzzification process}

In section III, the defuzzified values have been manually calculated using the Center of Largest Area, as shown in Figure 3 and Figure 4. However, manual calculation is not sufficient, a real hardware is needed which will automatically calculate the defuzzified values. Therefore, in this section, a novel architecture of a defuzzifier based on C.O.L.A. technique has been designed and simulated. The proposed architecture has been modeled in VHDL and has been implemented in field programmable gate array (FPGA). The defuzzified output in the C.O.L.A. method is given by the following equation

\begin{equation}
Z^* = \frac{\int \mu_C(z)dz}{\int \mu_C(z)dz}
\end{equation}

\textit{Figure 4: Defuzzification for MODEL = A}
TABLE 1: Calculation of Defuzzification parameters

<table>
<thead>
<tr>
<th>parameters</th>
<th>Model (1)</th>
<th>Model (2)</th>
<th>Model (3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1X1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C1X2</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>C1X3</td>
<td>6</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>C1X4</td>
<td>8</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>Height of C1</td>
<td>10</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>Center of C1</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Area of C1</td>
<td>60</td>
<td>35</td>
<td>105</td>
</tr>
<tr>
<td>C2X1</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>C2X2</td>
<td>7</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>C2X3</td>
<td>13</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>C2X4</td>
<td>14</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>Height of C2</td>
<td>5</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Center of C2</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Area of C2</td>
<td>35</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>Largest Area</td>
<td>C1</td>
<td>C2</td>
<td>C1</td>
</tr>
<tr>
<td>Center of Largest Area</td>
<td>4</td>
<td>10</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 5 are the architecture realization of the proposed architecture of equation (1). The VHDL modeling of the proposed architectures have been performed. The functional analysis is shown in Figure 6. It is clear from the functional analysis that there is a clear cut match between the results obtained manually and results generated by the architecture. In the first T-state of the functional analysis, the output of the defuzzifier ‘O’ is 4H, which is same as obtained manually for Figure 3. Similarly, for T-State second and third, the outputs of the defuzzifier is AH and 5H respectively, which is same as calculated manually. This shows the proposed architecture is realizing the defuzzifier action efficiently and accurately.
Fpga implementation of the defuzzifier

A Spartan 3E XC3S100E FPGA platform from XILINX has been used to implement the proposed architecture. The FPGA has around 1920 4-input look up tables (LUT) and 66 bonded input/output buffers (IOB). The FPGA logic resource used in an implementation are shown in Table 2, schematic and netlist generated of the proposed defuzzifier shown in Figure 7 and 8 respectively. The implementation results shown in Table 2 show that there is further scope for improvement in the proposed structures by incorporting more parallelism in the architecture.

![Schematic of the proposed defuzzifier](image)

**Figure 7: Schematic of the proposed defuzzifier**

![Netlist generated by the synthesis tools](image)

**Figure 8: Netlist generated by the synthesis tools.**

<table>
<thead>
<tr>
<th>TABLE 2: FPGA Implementation results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Utilization Summary (FPGA: Spartan 3E XC3S100E)</td>
</tr>
<tr>
<td>Logic Utilization</td>
</tr>
<tr>
<td>Number of 4 input LUTs</td>
</tr>
<tr>
<td>Number of occupied Slices</td>
</tr>
<tr>
<td>Number of Slices containing only related logic</td>
</tr>
<tr>
<td>Number of Slices containing unrelated logic</td>
</tr>
<tr>
<td>Total Number of 4 input LUTs</td>
</tr>
<tr>
<td>Number of bonded IOBs</td>
</tr>
<tr>
<td>Number of MULT 18X18SIOs</td>
</tr>
<tr>
<td>Average Fanout of Non-Clock Nets</td>
</tr>
</tbody>
</table>

**Conclusion**

The proposed architecture is based on Center of Largest Area. The COLA is the most important and simplest of the defuzzification methods. The defuzzified values have been initially manually calculated and finally have been obtained by the proposed architecture. The VHDL modeling and Spartan 3E FPGA implementation of the proposed architecture has been done. It has been seen that the proposed architecture realizes COLA based defuzzifier efficiently, as there is a complete match between the results obtained manually and through the architecture.
معمارية الدوائر المتكاملة الفائقة المدى لوحدة إعادة التنصيب

عاصم مجيد مرشد، أحمد جالاك شاكر، سيد ناهد محمد
قسم علوم الحاسوب، كلية العلوم، جامعة كركوك، كركوك، العراق
Asim.majeed2001@gmail.com

المتخصّص

وحدة فك الضبابية يجمع بين المتغيرات العشوائية والقرارات العشوائية لتشکّل الإشارات التعرفة والتي يمكن استخدامها بعد ذلك ل atrav مكمات بعض العمليات. بما إن البيانات العشوائية لا يمكن استخدامها مباشرة في إجراءات زمنية التشفير، فعلية فإنه لا بد من تحليلها إلى قيمة واضحة. في هذا العمل تم تصميم وحدة فك الضبابية بغرض تحويله للكتابة باستخدام طرق أكثر تدخلا من طريقة مركز المجموع، مركز الـ COLA ونضيفها

الضبابية بسيطة وتم استخدامها بشكل عام بالمقارنة مع طرق أخرى تدعي من طريقة مركز المجموع، يتم فك الهيكل المفتتح في Xilinx ونضيفها

العديد المماثل ليس مصروفات البيانات القابلة للبرمجة (FPGA) في المحالمة، بعد أن تجنيب المركز المفتتح هو أكثر كفاءة في المنطقة وسرعة العملية

الكفاءة مع بنية أكثر تدعي من مركز المجموع، وقد تبين من التحليل الوظيفي أن الهيكل المفتتح يتضمن

VLSI