Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)
Vol. 3, No. 1, Maret 2016, hlm. 75-82

IMPLEMENTASI LOGIKA FUZZY PADA SISTEM BERBASIS FIELD PROGRAMMABLE GATE ARRAY (FPGA)

Mochammad Hannats Hanafi Ichsan[1], Eko Setiawan[2], Mochamad Afief Hamidi[3]

Fakultas Ilmu Komputer, Universitas Brawijaya, Malang
Email : hanas.hanafi@ub.ac.id, ekosetiawan@ub.ac.id, afief.luffy@gmail.com

(Naskah masuk: 18 Februari 2016, diterima untuk diterbitkan: 17 Maret 2016)

Abstrak

Penggunaan perangkat mikrokontroller dewasa ini semakin banyak dipergunakan seperti arduino, atmega, FPGA dan lain sebagainya. Salah satu perangkat tersebut adalah FPGA (Field Programmable Gate Array). Bahasa yang digunakan pada FPGA adalah VHDL atau VHSIC (Very High Speed Integrated Circuit Hardware Description Language) merupakan salah satu jenis bahasa yang digunakan untuk mendeskripsikan fungsi rangkaian digital. Pada penelitian ini akan dijelaskan tentang implementasi tentang dasar Logika Fuzzy pada VHDL. Sehingga memiliki keuntungan jika dilakukan implementasi ini akan didapatkan rancang bangun logika fuzzy yang memungkinkan untuk diimplementasikan dengan cepat pada perangkat seperti Xilinx, Synosis dan lain sebagainya. Pada penelitian ini berhasil diimplementasikan, proses pengujian dilakukan dengan membandingkan perhitungan matematis dengan hasil keluaran sistem yang didapatkan akurasi sebesar 80%. Akan tetapi proses waktu eksekusi total untuk semua proses dalam Logika Fuzzy sebesar 145 ns.

Kata Kunci : *Logika Fuzzy, VHDL, FPGA*

Abstract

The use of microcontroller devices nowadays more and more used as arduino, atmega, FPGA and so forth. One such device was FPGA (Field Programmable Gate Array). The language used in the FPGA was VHDL or VHSIC (Very High Speed Integrated Circuit Hardware Description Language) is a kind of language used to describe functions of a digital circuit. This research would be explained on the basis of the implementation of Fuzzy Logic in VHDL. So it has an advantage when it is done this implementation will be obtained fuzzy logic design that allows to implemented quickly in devices such as Xilinx, Synosis and others. In this research successfully implemented, the testing process is done by comparing the mathematical calculations, the results obtained system output accuracy of 80%. But the total execution time for all processes in the Fuzzy Logic amounted to 145 ns.

***Keywords** : Fuzzy Logic, VHDL, FPGA*

1. PENDAHULUAN

Dewasa ini berbagai macam alat digunakan menggunakan teknologi mikroprosesor. Sebagian besar alat yang ada menggunakan arduino maupun atmega. Sedangkan untuk FPGA sedikit pengguna yang memakai alat tersebut (Ashenden, 2002). FPGA (*Field-programmable Gate Arrays*) merupakan gerbang digital dengan IC (*Integrated Circuit*) dimana masing-masing gerbang dapat memiliki interkoneksi dan dapat dilakukan konfigurasi antar bagiannya (Solano, 2006).Penggunaan FPGA memiliki keuntungan dalam implementasinya yang bertujuan untuk memberikan efisiensi rancangan dengan cara mengurangi pemakaian perangkat keras. (Nasrullah, 2009). Koreksi error yang dimiliki FPGA sangat kecil dan adalah teknologi yang dapat digunakan untuk implementasi menggunakan berbagai algoritma (IEEE, 1988).

Logika fuzzy telah banyak diterapkan dalam bidang kendali otomatis dan juga sistem pendukung keputusan (Kusumadewi, 2004). Pengambilan keputusan sebaiknya dilakukan berdasarkan pertimbangan-pertimbangan tertentu agar keputusan yang diambil dapat bersifat objektif, karena pengambilan keputusan yang dilakukan tanpa pertimbangan yang benar dapat mengakibatkan keputusan yang diambil menjadi kurang objektif (Zhu, 2007).

FPGA menggunakan pemrograman bahasa VHDL. VHDL atau VHSIC *Very High Speed Integrated Circuit Hardware Description Language* merupakan salah satu jenis bahasa HDL yang digunakan untuk mendeskripsikan fungsi rangkaian digital seperti gerbang logika, flip-flop dan lain sebagainya (IEEE, 1988) (Raychev, 2005) (Salapura, 2005). Akan tetapi pada VHDL semua bilangan desimal akan diubah pada bilangan biner, heksadesimal ataupun boolean (Sakthivel, 2010), hal ini akan berpengaruh pada proses yang ada dalam Logika Fuzzy karena angka yang diproses didalam Logika Fuzzy berupa angka desimal (Olivas, 2008).

Konversi inilah yang menyebabkan perbedaan selisih nilai yang diproses oleh VHDL.

Pada penelitian sebelumnya yang dilakukan oleh Asad dkk (Asad, August, 1995) implementasi logika fuzzy dengan VHDL untuk kontrol temperatur Pendingin Ruangan. Pada penelitian lain yang pernah dilakukan oleh Tigaeru dan Ursaru hanya mensimulasikan Logika Fuzzy pada VHDL (Tigaeru, May, 2004). Serta pada buku yang ditulis oleh Dadios (Dadios, 2012) juga tidak disebutkan apakah hasil akhir dari pengolahan Logika Fuzzy tepat sesuai harapan perhitungan manual. Dengan penelitian yang sama oleh Dadios, Maldonado dan Castillo (Maldonado, 2013) tidak melakukan proses koreksi untuk membandingkan dengan perhitungan manual. Oleh karena itu penulis akan meneliti seberapa akurat dan seberapa cepat proses Logika Fuzzy pada VHDL karena pada proses FPGA, berbeda dengan bahasa pemrograman karena berbasis gerbang logika. Serta pengujian keakuratan hasil akhir sangat berpengaruh terhadap keluaran sistem.

2. METODOLOGI

Pada sub bab ini akan dijelaskan bagian-bagian yang akan dibahas pada penelitian beserta perancangan yang akan diimplementasikan.

3. FPGA (Field-Programmable Gate Arrays)

Gerbang digital yang memiliki interkoneksi antar masing-masing gerbang tersebut serta dapat dikonfigurasi antar bagiannya disebut dengan FPGA. Sebuah IC pada FPGA terdiri dari 3 komponen pendukung utama yaitu *Configurable Logic Block*, *Input Output Block* dan *Programmable Interconnect* (Ashenden, 2002).

Configurable Logic Block (CLB) merupakan komponen dasar yang membentuk IC FPGA berupa matrikmatrik yang saling terhubung oleh PI, yang dapat dikonfigurasikan untuk melakukan rangkaian logika kombinasional, shift register, atau RAM.

Input Output Block (IOB) merupakan penghubung atau sebagai antarmuka antara pin-pin terminal IC FPGA dengan jalur koneksi di luar IC, IOB dikelompokkan ke dalam beberapa I/O Bank yang sesuai dengan standar I/O.

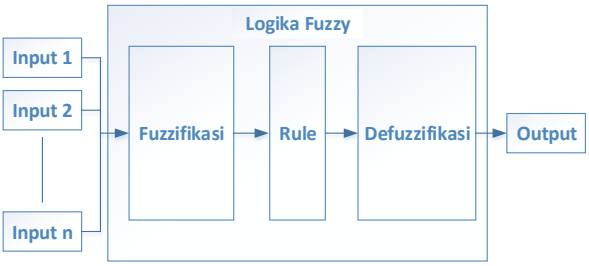
Programmable Interconncect (PI) adalah komponen yang berperan sebagai saklar penghubung yang dapat dikonfigurasikan dan akan menghubungkan antar blok-blok CLB maupun dengan IOB.

3.1. VHDL

VHDL (VHSIC *Hardware Description Languange*) merupakan bahasa pemrograman yang digunakan pada FPGA. Secara umum struktur bahasa pemrograman VHDL terdiri dari tiga bagian pokok yaitu deklarasi *Library*, *Entity*, *dan Architecture* (IEEE, 1988). Deklarasi Library merupakan kumpulan potongan kode yang sering digunakan, Sebuah entity berisi deklarasi Input/Output dari suatu sistem, sedangkan deklarasi arsitektur berisi deskripsi dari suatu rangkaian atau fungsi logika yang akan diimplementasikan (Salapura, 2005). Di dalam arsitektur dapat menangani baik rangkaian sekuensial maupun rangkaian kombinasional dan dalam 1 (satu) modul VHDL dimungkinkan untuk menggunakan lebih dari 1 (satu) arsitektur (Ashenden, 2002). Semua bilangan pada VHDL akan diproses dengan bilangan boolean karena digunakan gerbang logika yang akan memproses input sampai output.

3.2. Blok Diagram Sistem

Proses logika fuzzy masih dilakukan secara umum yaitu mulai dari input, proses *fuzzification*, rule, *defuzzification* beserta output yang dapat dilihat pada Gambar 1.



Gambar 1. Proses Logika Fuzzy

Pengambilan keputusan dalam teknik fuzzy dilakukan dalam beberapa tahap yaitu: *fuzzification*, penentuan m*embership function, rule evaluation* dan *defuzzification* (Kusumadewi, 2004). Proses pertama pada Logika Fuzzy adalah *membership function*, pada proses ini *membership function* akan memproses masukan atau input untuk diolah kedalam fungsi keanggotaan, sampai didapatkan input tersebut masuk dalam fungsi keanggotaan yang telah ditentukan. Penentuan ini berdasarkan nilai yang telah diset sebelumnya. Kemudian dilanjutkan proses *rule evaluation*.

Pada rule evaluation akan dilakukan pencarian fuzzy output dari fuzzy input dengan proses dimana nilai fuzzy input yang didapatkan proses *fuzzification* kemudian dimasukkan kedalam sebuah rule yang telah dibuat untuk dijadikan sebuah fuzzy output. Rule evaluation merupakan bagian utama dari fuzzy karena *rule evaluation* akan menjadi dasar untuk menentukan sistem menjadi pintar atau tidak. Rule yang sudah ditentukan kemudian diaplikasikan kedalam fungsi implikasi. Secara umum ada 2 fungsi implikasi yaitu: Max-Min dan Max-Prod, pada penelitian ini akan dipergunakan fungsi implikasi yang sering digunakan dalam pengambilan keputusan adalah Max-Min.

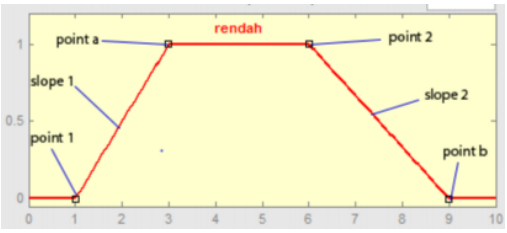
3.3. Logika Fuzzy

Pada penelitian ini, yang akan dilakukan untuk pengujian adalah sebuah simulasi sistem kontrol sederhana untuk mendeteksi gas *hidrocarbon* (hc) dan *carbon* (co) yang asumsinya didapatkan dari

sensor. Namun pada penelitian ini input tidak didapatkan dari masukan sensor, tetapi dari masukan pada simulator. Masing-masing input akan diberikan fungsi keanggotaan, untuk *hidrocarbon* adalah rendah, sedang, dan tinggi. Sedangkan untuk input *carbon* juga akan diberikan himpunan sedang, rendah dan tinggi yang masing-masing memiliki fungsi keanggotaan yang berbeda. Kemudian akan diolah kedalam *rule evaluation* yang akan mengolah nilai input dan terakhir akan diproses dengan *defuzzification* yang menggunakan Min-Max.

3.4. *Fuzzification*

Pada proses ini didefinisikan derajat keanggotaan. Derajat keanggotaan didapatkan untuk setiap himpunan, pada VHDL harus diubah ke boolean (Maldonado, 2013) sehingga 00H untuk derajat keanggotaan 0 dan FFH untuk derajat keanggotaan 1 seperti pada Gambar 2.

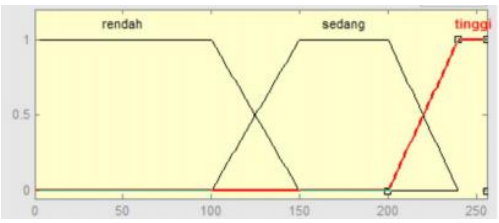


Gambar 2. Derajat Keanggotaan

Dari Gambar 2 dapat dilihat terdiri dari 2 slope. Point merupakan titik pada tiap trapesium keanggotaan (Raychev, 2005). Sedangkan slope merupakan sisi miring pada trapesium keanggotaann. Untuk menentukan slope didapat dari Persamaan 1.

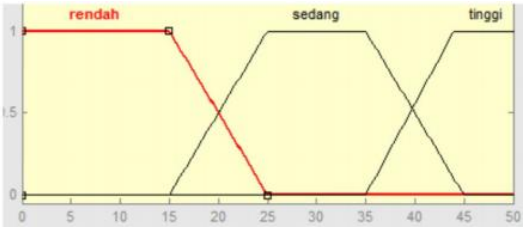
$$Slope = \frac{(y2-y1)}{(x2-x1)} \qquad (1)$$

Untuk *hidrocarbon* fungsi keanggotaannya bernilai 1 atau rendah pada masukan 0 sampai 100, untuk nilai 0 jika masukan lebih dari 150. Sedangkan grafik turun dari rendah mulai 100-150. Untuk kondisi sedang bernilai 1 diantara 150-200, bernilai 0 diantara 0 sampai 100 dan diatas 240, sedangkan grafik naik diantara 100-150 dan grafik turun diantara 200-240. Sedangkan untuk fungsi keanggotaan tinggi bernilai 0 dari masukan bernilai 0 sampai 200, fungsi keanggotaan naik dari 200-240 dan bernilai 1 jika masukan diatas 240 seperti dapat dilihat pada Gambar 2.



Gambar 3. Himpunan *Hidrocarbon* (hc)

Untuk *carbon* fungsi keanggotaannya bernilai 1 atau rendah pada masukan 0 sampai 15, untuk nilai 0 jika masukan lebih dari 25. Sedangkan grafik turun dari rendah mulai 15-25. Untuk kondisi sedang bernilai 1 diantara 25-35, bernilai 0 diantara 0 sampai 15 dan diatas 45, sedangkan grafik naik diantara 15-25 dan grafik turun diantara 35-45. Sedangkan untuk fungsi keanggotaan tinggi bernilai 0 dari masukan bernilai 0 sampai 35, fungsi keanggotaan naik dari 35-45 dan bernilai 1 jika masukan diatas 45 seperti pada Gambar 3.



Gambar 4. Himpunan *Carbon* (co)

Dikarenakan menggunakan 8 bit maka µ = 1 sama dengan 255 atau $FF (tanda $ mengindikasikan pada penulisan hexadesimal). Sebagai contoh point1 bernilai 1 dan point 2 bernilai 6 maka slope 1 dan slope 2 menggunakan rumus (3.0) maka perhitungannya pada Persamaan 2 dan 3 dan untuk persamaan yang lain diletakkan pada *Configurable Logic Block*.

$$Slope\ 1 = \frac{1}{(3-1)} = \frac{\$FF}{2} = \frac{255}{2} = 127 = \$7F \qquad (2)$$

$$Slope\ 2 = \frac{1}{(9-6)} = \frac{\$FF}{3} = \frac{255}{3} = 85 = \$55 \qquad (3)$$

Dengan proses yang sama dilakukan perhitungan matematis pada VHDL untuk fungsi 2 fungsi yang lain misalkan untuk memproses masukan *hidrocarbon* dan *carbon*.

```
type input is (hcrendah,hcsedang,hctinggi);
type membership is record term: input;
point1: std_logic_vector (7 downto 0);
slope1: std_logic_vector (7 downto 0);
point2: std_logic_vector (7 downto 0);
slope2: std_logic_vector (7 downto 0);
end record;

type membership_functions is array(natural range<>) of membership;
constant mfs: membership_functions:=
((term => hcrendah, point1 => "00000000", slope1 => x"00", point2 => x"64", slope2 => x
(term => hcsedang, point1=> x"6E", slope1=> x"06", point2 => x"C8", slope2 => x"06"),
(term => hctinggi, point1=> x"C8", slope1=> x"06", point2 => x"F0", slope2 => x"FF"));
```

Gambar 5. Implementasi Representasi Fungsi Keanggotaan *hidrocarbon* pada VHDL

Dari Gambar 5, dapat dilihat bahwa domain untuk nilai keanggotaan rendah adalah [0 0 100 150]. Pada gambar membership terlihat bahwa range yang dimiliki adalah 0 sampai 256 dikarenakan input diskalakan dikarenakan di vhdl menggunakan input 8-bit, untuk fungsi keanggotaan tinggi memiliki nilai maksimal 256.

```
type input is (corendah,cosedang,cotinggi);
type membership is record term: input;
point1: std_logic_vector (7 downto 0);
slope1: std_logic_vector (7 downto 0);
point2: std_logic_vector (7 downto 0);
slope2: std_logic_vector (7 downto 0);
end record;

type membership_functions is array(natural range<>) of membership;
constant mfs: membership_functions:=
((term => corendah, point1=> x"00", slope1=> x"00", point2 => x"0F", slope2 =>
(term => cosedang, point1=> x"0F", slope1=> x"19", point2 => x"23", slope2 =>
(term => cotinggi, point1=> x"23", slope1=> x"19", point2 => x"2D", slope2 =>
```

Gambar 6. Implementasi Representasi Fungsi Keanggotaan *Carbon* pada VHDL

Hal yang sama terjadi pada input *carbon*, semua nilai pada fungsi keanggotaan dikonversi pada bilangan heksadesimal.

3.5. *Rule evaluation*

Setelah menentukan derajat keanggotaan dalam tahap fuzzifikasi. Langkah selanjutnya adalah membuat aturan untuk memutuskan tindakan apa yang harus diambil dalam himpunan fungsi keanggotaan. Ada standar operator fuzzy yang dapat digunakan untuk mendefinisikan rule yaitu "AND", "OR" dan "NOT".

```
function minimum (a,b: std_logic_vector) RETURN std_logic_vector
   variable min: std_logic_vector(7 downto 0):= (others => '0');

   begin
      if a<b then min := a;
      elsif b<a then min := b;

      end if;
   RETURN min;
end minimum;
```

Gambar 7. Representasi Fungsi Minimum

Pada penelitian ini dipergunakan fungsi "AND" seperti ditunjukkan pada Gambar 4. Untuk "AND" => C = minimum (A, B).

```
--RULE BASE
rule1 : position(0)<= minimum(u1(0),u4(3));
-- UNTUK KONDISI HC RENDAH DAN CO RENDAH

rule2 : position(1)<= minimum(u1(0),u5(4));
-- UNTUK KONDISI HC RENDAH DAN CO SEDANG

rule3 : position(2)<= minimum(u1(0),u6(5));
-- UNTUK KONDISI HC RENDAH DAN CO TINGGI
```

Gambar 8. Representasi *Rule evaluation*

Fungsi minimum dan maksimum yang digunakan untuk memperoleh hasil dari setiap *Rule evaluation*. Setelah masing-masing nilai hasil dari fungsi keanggotaan, diproses dengan *Rule evaluation* seperti pada Gambar 8.

Tabel 1. Fuzzy Rule

Rule	Hc	Co	Position
1	Rendah	Rendah	Position(0)
2	Rendah	Sedang	Position(1)
3	Rendah	Tinggi	Position(2)
4	Sedang	Rendah	Position(3)
5	Sedang	Sedang	Position(4)
6	Sedang	Tinggi	Position(5)
7	Tinggi	Rendah	Position(6)
8	Tinggi	Sedang	Position(7)
9	Tinggi	Tinggi	Position(8)

Pada Gambar 8 tersebut adalah potongan dari rule yang dibuat, untuk detail dari rule yang dirancang ada pada Tabel 1, karena fungsi keanggotaan *hidrocarbon* sebanyak tiga dan fungsi keanggotaan *carbon* tiga sehingga dihasilkan 9 rule.

3.6. *Defuzzification*

Setelah mengetahui keluaran dari rule maka selanjutnya adalah mengkombinasikannya menjadi satu output. Pada proses ini dilakukan fungsi implikasi, pada fungsi implikasi yang sering digunakan adalah fungsi Max-Min karena metode ini adalah metode yang paling umum digunakan seperti pada Persamaan 4. Metode Max (Maksimum) mengambil nilai yang diperoleh dengan cara mengambil nilai maksimum.

$$If\ a\ is\ Ai\ and\ b\ is\ Bi\ then\ c\ is\ Ci = f(a,b) \quad (4)$$

$$\mu\, x(i) = \mu\, x(i) = max\,(\mu sf\,(xi)\,, \mu kf\,(xi)\,) \quad (5)$$

Fungsi implikasi menggunakan metode min max untuk menentukan keluaran output dengan nilai terbesar. Fungsi implikasi tersebut akan memilih hasil keluaran dari *rule evaluation* yang telah diproses sebelumnya dengan menggunakan Persamaan 5.

```
product <= max(product9a,product9b);
```

Gambar 9. Representasi *Defuzzification*

Kode VHDL untuk proses defuzzifikasi ditunjukkan pada Gambar 9. Setelah proses defuzzifikasi selesai maka nilai output yang dihasilkan adalah hasil akhir dari proses Logika Fuzzy.

4. HASIL DAN PEMBAHASAN

Pada bagian hasil dan pembahasan akan dibahas tentang hasil yang didapatkan dari dua macam pengujian melalui simulator. Pertama pengujian waktu eksekusi secara keseluruhan, kedua dilakukan pengujian dengan cara membandingkan dengan perhitungan manual.

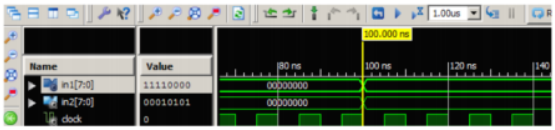
4.1. Pengujian Waktu Eksekusi

Pada pengujian ini dimasukkan nilai set sebagai asumsi bahwa input didapatkan dari nilai akuisisi data sensor dengan input 240 dikonversi pada bilangan biner "11110000" untuk nilai hc dan 21 dikonversi juga "00010101" untuk nilai kadar co seperti pada Gambar 10, syntax tersebut dimasukkan dalam *Programmable Interconncect*.

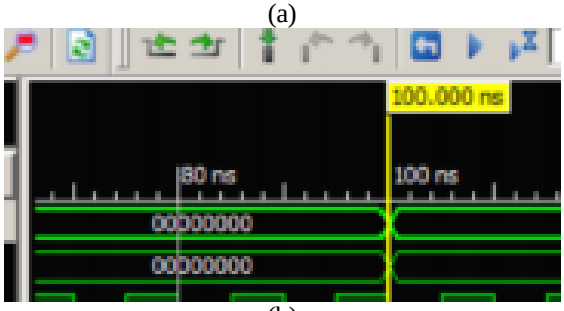
```
-- Stimulus process
stim_proc: process
begin
   -- hold reset state for 100 ns.
   wait for 100 ns;

   -- insert stimulus here
   in1 <= "11110000"; --240
   in2 <= "00010101"; --21
```

Gambar 10. Masukan Nilai

Dari potongan syntax diatas dapat diketahui bahwa input akan masuk pada detik ke 100 ns. Hasil run program seperti pada Gambar 11 (a). Sedangkan untuk hasil running secala jelas yang menunjukkan waktu sebesar 100 ns ada pada Gambar 11 (b). Masing-masing pengerjaan untuk proses *membership function*, *rule evaluation* sampai *defuzzification* dikerjakan pada *Input Output Block.*



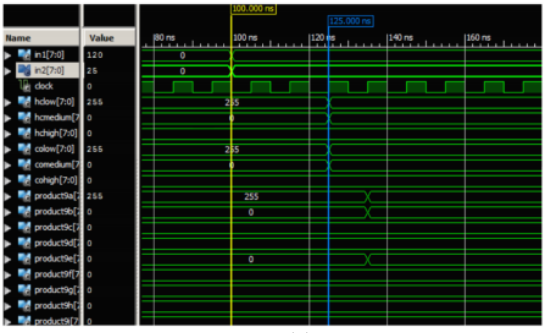
(a)



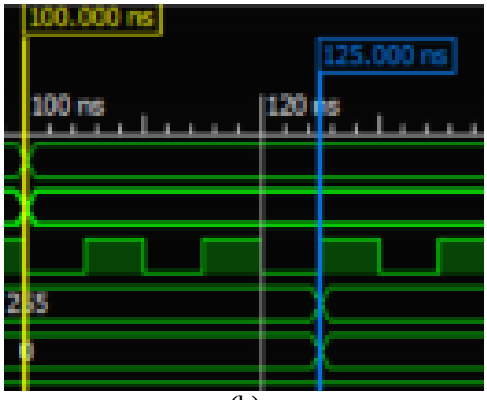
(b)

Gambar 11. (a)Tampilan Masukan Pengujian Pada Simulator (b)Detail Waktu Pengujian

Nilai "in1" dan "in2" pada Gambar 11 adalah representasi dari pengambilan nilai sensor yang dapat diubah sesuai nilai yang diinginkan. Masing-masing memiliki slot untuk proses yang berbeda dan diproses secara bersamaan. Pengujian waktu proses *fuzzification* dibutuhkan waktu 25 ns untuk menentukan keanggotaan. Hasil waktu eksekusi ditunjukkan pada Gambar 12.



(a)



(b)

Gambar 12. (a)Pengujian Waktu Proses *Fuzzification* Pada Simulator (b) Detail Waktu Yang Dilakukan

Dari Gambar 12 (a) dapat diketahui garis warna kuning merupakan total proses input pada sistem, sedangkan garis warna biru merupakan output fungsi keanggotaan yang diperoleh dari input. Pada Gambar 11 (b) ditunjukkan warna biru sebesar 125 ns atau total waktu dari awal proses sampai proses *defuzzification*. Sehingga total waktu untuk *defuzzification* adalah sebesar 25 ns seperti pada Persamaan 6.

$$125\,ns - 100\,ns = 25\,ns \qquad (6)$$

Untuk proses selanjutnya yaitu *rule evaluation* dibutuhkan waktu 10 ns untuk keluar nilai output. Hasil waktu eksekusi dapat dilihat pada Gambar 13.

(a)

(b)

Gambar 13. Pengujian Waktu Eksekusi *Rule evaluation*

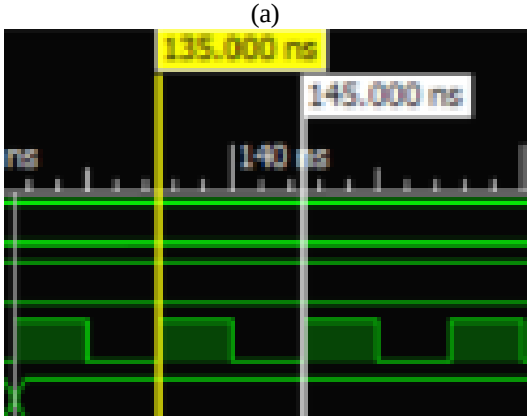
Dari Gambar 13 (a) dapat diketahui garis warna putih merupakan hasil output dari rule evaluation. Seperti dibahas sebelumnya, warna

kuning adalah total dari awal proses *membership function*, warna biru adalah total awal proses sampai *defuzzification* sedangkan warna putih adalah total dari awal proses sampai *rule evaluation* seperti pada Gambar 13 (b). Sehingga waktu yang dipergunakan untuk proses *rule evaluation* sendiri adalah sebesar 10 ns seperti pada Persamaan 7.

$$135\,ns - 125\,ns = 10\,ns \qquad (7)$$

Setelah proses rule evaluation dilanjutkan pada proses *defuzzification* yaitu menentukan output akhir dari sistem tersebut. Hasil waktu eksekusi dapat dilihat pada Gambar 14.

(a)



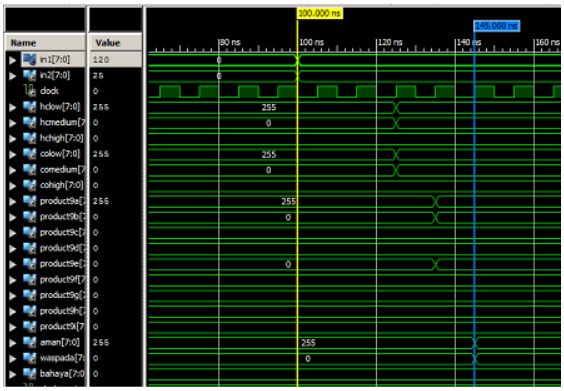
(b)

Gambar 14. (a) Pengujian Waktu *Defuzzification* Pada Simulator (b) Detail Waktu Gambar 14(a)

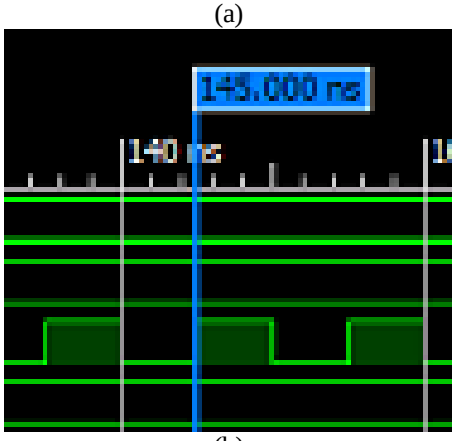
Dari gambar diatas dapat diketahui garis kuning merupakan hasil output dari awal proses sampai dengan proses rule evaluation, sedangkan garis berwarna putih merupakan hasil output dari *defuzzification*. Sehingga untuk waktu yang dibutuhkan untuk proses *rule evaluation* sebesar 10 ns seperti pada Persamaan 8.

$$145\,ns - 135\,ns = 10\,ns \qquad (8)$$

Dari semua proses tersebut dapat diketahui berapa waktu yang dibutuhkan sistem dari awal input sampai hasil akhir keluar. Berikut adalah hasil waktu eksekusi dapat dilihat pada Gambar 15 (a) adalah detail pengujian pada simulator seperti digambarkan dengan garis dengan warna biru.



(a)



(b)

Gambar 15. (a) Hasil Akhir Pengujian Pada Simulator (b) Detail Hasil Akhir Pengujian

Sedangkan pada Gambar 15 (b) digambarkan dengan detail dari gambar yang dimiliki oleh Gambar 15(a) dimana waktu akhir adalah 145 ns. Sehingga dari awal running, dijumlahkan masing-masing per bagian proses Logika Fuzzy didapatkan waktu total sebesar 145 ns, atau dapat dilihat pada Gambar 15 pada garis yang berwarna biru.

4.2. Pengujian Perbandingan dengan Perhitungan Manual

Pada pengujian keseluruhan untuk perhitungan manual dilakukan proses dan rumus Logika Fuzzy secara umum. Pada pengujian akhir ini nilai input telah diskalakan dan diolah dalam nilai desimal. Pada pengujian akhir perhitungan secara manual akan dicocokkan dengan output dari simulator VHDL yang ditunjukkan pada tabel pada Tabel 2.

Tabel 2. Pengujian dengan Perhitungan Manual

No.	input		Manual	VHDL
	in1(hc)	in2(co)		
1	80	10	255	255
2	80	20	130	130
3	80	30	255	255
4	80	40	130	130
5	80	50	255	255

No.	input		Manual	VHDL
	in1(hc)	in2(co)		
6	120	10	195	195
7	120	20	130	130
8	120	30	195	155*
9	120	40	195	130*
10	120	50	195	155*
11	160	10	255	255
12	160	20	130	130
13	160	30	255	255
14	160	40	130	130
15	210	50	255	255
16	210	10	195	179*
17	210	20	130	130
18	210	30	195	179*
19	210	40	130	130
20	210	50	195	195
21	240	10	255	255
22	240	20	130	130
23	240	30	255	255
24	240	40	130	130
25	240	50	255	255

5. KESIMPULAN

Logika fuzzy dapat dimplementasikan pada FPGA dengan menggunakan pengkodean yang cukup kompleks. Perhitungan waktu eksekusi dapat dilakukan dengan rincian untuk proses *membership function* memakan waktu 125 ns, pada proses ini cukup memakan waktu yang sangat panjang karena memiliki paling banyak proses untuk menentukan pada fungsi keanggotaan manakah nilai input yang dimasukkan. Kedua adalah proses *rule evaluation* yang membutuhkan waktu 10 ns dan proses *defuzzification* membutuhkan waktu 10 ns karena pada proses ini tidak membutuhkan proses yang kompleks. Pada proses *rule evaluation* dan *defuzzification* hanya memasukkan nilai yang dihasilkan dari *membership function* pada rule yang telah disediakan. Sehingga total waktu yang dipergunakan untuk proses dari awal sampai akhir adalah 145 ns. Untuk beberapa kali pengujian didapatkan panjang waktu yang sama untuk masing-masing input yang berbeda.

Sedangkan untuk akurasi sistem, sebanyak 25 kali percobaan didapatkan 5 data output yang tidak sesuai dengan perhitungan matematis, hal ini disebabkan oleh konversi data dari bentuk desimal dan diubah kedalam heksadesimal maupun biner untuk dapat diolah pada VHDL sehingga tingkat akurasi dari sistem yang didapatkan sebesar 80%. Sehingga dapat disimpulkan bahwa keluaran dari VHDL tidak selalu sama dengan perhitungan manual. Setelah dilakukan pengecekan, ketidaksesuaian ini terdapat pada awal proses konversi data pada proses *membership function*, karena pada proses ini akurasi dan presisi nilai dilakukan dengan detail.

6. DAFTAR PUSTAKA

Asad M., Wan, L. A., Kuo, D., and Vuong, J. Sensor Based Microcontroller Unit with Built In Fuzzy Inference for an Endometrium Ablator. 3rd European Congress on Intelligent Techniques and Soft Computing (EUFIT ’95). - pp. 1621-1624 : [s.n.], August, 1995.

Ashenden P.J., Peterson, G.D., Teegarden, D. The System Designer’s Guide VHDL-AMS. - San Francisco, US : Morgan Kaufman Publishers, 2002.

Dadios E., Pourshaghaghi, H.R., Escobar, J.D.E., and JGyvez, J.P.de. ynthesis 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.) - Croatia : InTech, 2012. - 978-953-51-0396-7.

IEEE IEEE Standard VHDL Language Reference Manual. IEEE. - New York : IEEE Standard 1076-1987, 1988.

Kusumadewi S., Purnomo, S. Aplikasi Logika Fuzzy untuk Pendukung Keputusan. - Yogyakarta : Graha Ilmu, 2004.

Maldonado Y., and Castillo, O. Particle Swarm Optimization of Interval Type-2 Fuzzy Systems for FPGA Applications. Applied Soft Computing. - 2013. - 1 : Vol. 13. - hal. 496–508.

Nasrullah E. Raharjo, J. Rancang Bangun Alat Pemantau Kualitas Udara Sekitar Berbasis Mikrokontroler AVR ATMega 8 Dengan Penampil Dot Matrix. Electrician, Jurnal Rekayasa dasn Teknilogi Elektro. - 2009. - Vol. 3, No 1. - hal. 1-9.

Olivas J., Sepulveda, R., and Castillo, O. Development of an Embedded Simple Tuned Fuzzy Controller. FUZZ-IEEE. (IEEE World Congress on Computational Intelligence). - Hong Kong : IEEE, 2008. - 978-1-4244-1818-3.

Raychev R., Mtibaa, A., and Abid, M. VHDL Modelling of a Fuzzy Co-Processor Architecture. International Conference on Computer Systems and Technologies - CompSysTech’2005.. - Varna, Bulgaria : [s.n.], 2005.

Sakthivel G., Anandhi, T.S., and Narajan, S.P. Real Time Implementation of a Fuzzy Logic Controller On FPGA Using VHDL for DC Motor Speed Control. International Journal of

Engineering Science and Technology. - 2010. - Vol. 9. - hal. 4511–4519.

Salapura V., Hamann, V. Implementing Fuzzy Control Systems Using VHDL and Statecharts. Technische Universitat Wien. - Wien-Austria : [s.n.], 2005.

Solano S.S., Brox, P., Cabrera, A., and Baturone, I. Modelling and implementation of fuzzy systems based on VHDL. International Journal of Approximate Reasoning. - 2006. - 2 : Vol. 41. - hal. 164-178.

Tigaeru L., Ursaeru, O. A VHDL Based Approach To Model Fuzzy Logic Systems. 7th International Conference on Development and Application Systems. - Suceava, Romania : [s.n.], May, 2004.

Zhu H., Yang, D. Fuzzy Systems and Knowledge Discovery. Changsa, China : Springer, 2007.