BAB IV

IMPLEMENTASI

Bagian ini menjelaskan program-program untuk mengendalikan *microcontroller* Arduino.

4.1 Program Arduino *Fuzzy*

4.1.1 Program Inisialisasi Arduino *Fuzzy*

```
#include <DHT.h>
#include <SPI.h>
#include <Ethernet.h>
#include "fis_header.h"
#include "RBDdimmer.h"

#define outputPinKipas  9
#define outputPinLampu  10
#define outputPinPupuk  11
#define outputPinKapur  12
#define zerocross  2

dimmerLamp dimmer1(outputPinKipas);
//dimmerLamp dimmer2(outputPinLampu);
```

Gambar 4.1 Program Inisialisasi Arduino *Fuzzy*

Gambar 4.1 Tahap program inisialisasi dari Arduino *fuzzy*. Program `#include <DHT.h>` berfungsi untuk pemanggilan suatu

library. Program `#define outputPinKipas 9` dan `#define zerocross 2` berfungsi untuk mendeklarasikan *channel dimmer* sesuai dengan *pin* pada *microcontroller* Arduino. Program `const int fis_gcI` berfungsi untuk mendeklarasikan *input*, yaitu DHT11 dan YL-69. Program `const int fis_gcO` berfungsi untuk mendeklarasikan *output*, yaitu Kipas Angin. Program `const int fis_gcR` berfungsi untuk mendeklarasikan *rules* untuk kondisi *output* dan *input*.

4.1.2 Program Arduino *Fuzzy* Tahap Setup

```
void setup()
{
    while (!Serial)
    {
      ;
    }

    Serial.begin(9600);
    dimmer1.begin(NORMAL_MODE, ON);
    //dimmer2.begin(NORMAL_MODE,ON);
    //dimmer3.begin(NORMAL_MODE, ON);
    //dimmer4.begin(NORMAL_MODE, ON);

    pinMode(powerPin, OUTPUT);
    digitalWrite(powerPin, LOW);
    dht.begin();

    Ethernet.begin(mac, ip, gateway, subnet);
    Serial.print("server is at ");
    Serial.println(Ethernet.localIP());}
```

Gambar 4.2 Program Tahap Setup Arduino Fuzzy

Gambar 4.2 Tahap program *setup* dari Arduino *fuzzy*. Program `while (!Serial)` berfungsi untuk menunggu Arduino siap berkomunikasi dengan komputer/laptop. Program

`dimmer1.begin(NORMAL_MODE, ON);` berfungsi untuk inisialisasi dimmer agar bisa menyala. Program `Ethernet.begin(mac, ip, gateway, subnet)` berfungsi untuk inisialisasi ethernet dan setting jaringan.

4.1.3 Program Arduino *Fuzzy* Tahap Looping

```
void loop()
{
    if (client.connect(server2, 80))
    {
      g_fisInput[0] = dht.readTemperature();
      double hasil = bacaSensor();
      g_fisInput[1] = hasil / 1023 * 100;
      Serial.println("Terhubung...");

      client.print( "GET /dht11/index.php?");
      client.print("s=");
      client.print(g_fisInput[0]);
      client.print("&k=");
      client.print(g_fisInput[1]);

      client.println();
      client.println( " HTTP/1.1");
      client.print( "Host: " );
      client.println(server2);
      client.println( "Connection: close" );
      client.println();
      client.stop();
      delay(5000);
      client.stop();
    }
```

```
else
{
  Serial.println("Connection Failed");
}

g_fisOutput[0] = 30;
g_fisOutput[1] = 60;
g_fisOutput[2] = 90;
fis_evaluate();

  if (g_fisInput[0] < 24 && g_fisInput[1] < 50)
  {
      dimmer1.setPower(g_fisOutput[0]);
      //dimmer2.setPower(g_fisOutput[2]);
      //dimmer3.setPower(g_fisOutput[2]);
      //dimmer4.setPower(g_fisOutput[0]);
      delay(500);
  }
  if (g_fisInput[0] < 24)
  {
    if (g_fisInput[1] >= 50 && g_fisInput[1] <= 70)
    {
      dimmer1.setPower(g_fisOutput[0]);
      //dimmer2.setPower(g_fisOutput[2]);
      //dimmer3.setPower(g_fisOutput[1]);
      //dimmer4.setPower(g_fisOutput[1]);
      delay(500);
    }
  }
```

```
if (g_fisInput[0] < 24 && g_fisInput[1] > 70)
{
    dimmer1.setPower(g_fisOutput[0]);
   // dimmer2.setPower(g_fisOutput[2]);
   // dimmer4.setPower(g_fisOutput[2]);
   // dimmer3.setPower(g_fisOutput[0]);
   delay(500);
}

if (g_fisInput[0] >= 24 && g_fisInput[0] <= 27)
{
  if (g_fisInput[1] < 50)
  {
    dimmer1.setPower(g_fisOutput[1]);
   // dimmer2.setPower(g_fisOutput[1]);
   // dimmer3.setPower(g_fisOutput[2]);
   // dimmer4.setPower(g_fisOutput[0]);
   delay(500);
  }
}
if (g_fisInput[0] >= 24 && g_fisInput[0] <= 27)
{
  if (g_fisInput[1] >= 50 && g_fisInput[1] <= 70)
  {
      dimmer1.setPower(g_fisOutput[1]);
      // dimmer2.setPower(g_fisOutput[1]);
      // dimmer3.setPower(g_fisOutput[1]);
      // dimmer4.setPower(g_fisOutput[1]);
         delay(500);}}
```

```
if (g_fisInput[0] >= 24 && g_fisInput[0] <= 27)
     {
        if (g_fisInput[1] > 70)
        {
          dimmer1.setPower(g_fisOutput[1]);
         // dimmer2.setPower(g_fisOutput[1]);
         // dimmer3.setPower(g_fisOutput[0]);
          //dimmer4.setPower(g_fisOutput[2]);
          delay(500);
        }
     }

     if (g_fisInput[0] > 27 && g_fisInput[1] < 50)
      {
          dimmer1.setPower(g_fisOutput[2]);
         // dimmer2.setPower(g_fisOutput[0]);
         // dimmer3.setPower(g_fisOutput[2]);
         // dimmer4.setPower(g_fisOutput[0]);
         delay(500);
      }
      if (g_fisInput[0] > 27)
      {
        if (g_fisInput[1] >= 50 && g_fisInput[1] <= 70)
        {
          dimmer1.setPower(g_fisOutput[2]);
         // dimmer2.setPower(g_fisOutput[0]);
         // dimmer3.setPower(g_fisOutput[1]);
         // dimmer4.setPower(g_fisOutput[1]);
         delay(500);}}
```

```
      if (g_fisInput[0] > 27 && g_fisInput[1] > 70)
      {
          dimmer1.setPower(g_fisOutput[2]);
         // dimmer2.setPower(g_fisOutput[0]);
        //  dimmer3.setPower(g_fisOutput[0]);
         // dimmer4.setPower(g_fisOutput[2]);
         delay(500);
      }
 delay(1000);
}

double bacaSensor()
{
  digitalWrite(powerPin, HIGH);
  delay(500);
  double nilaiSensor = analogRead(sensorPin);
  digitalWrite(powerPin, LOW);
  return 1023 - nilaiSensor;
}
```

Gambar 4.3 Program Tahap Looping Arduino Fuzzy

Gambar 4.3 Tahap program looping dari Arduino *fuzzy*. Program `if (client.connect(server2, 80))` berfungsi untuk pemeriksaan apakah *client* sudah terhubung dengan *ip server* dan *port* yang sesuai. Program `client.print("GET /dht11/index.php?")` berfungsi untuk mengakses data dari *web services* yang terletak pada *web server*. Program `fis_evaluate()` berfungsi untuk pemanggilan

method *fuzzy inference system* agar menghasilkan *output* yang ditampung di variabel g_fisOutput. Program `bacaSensor()` berfungsi untuk method dalam menentukan nilai sensor YL-69.

4.1.4 Program *Fuzzy Inference System*

```
//*************************************************************
*********

// Support functions for Fuzzy Inference System

//*************************************************************
*********

// Triangular Member Function

FIS_TYPE fis_trimf(FIS_TYPE x, FIS_TYPE* p)

{

    FIS_TYPE a = p[0], b = p[1], c = p[2];

    FIS_TYPE t1 = (x - a) / (b - a);

    FIS_TYPE t2 = (c - x) / (c - b);

    if ((a == b) && (b == c)) return (FIS_TYPE) (x == a);

    if (a == b) return (FIS_TYPE) (t2*(b <= x)*(x <= c));

    if (b == c) return (FIS_TYPE) (t1*(a <= x)*(x <= b));

    t1 = min(t1, t2);

    return (FIS_TYPE) max(t1, 0);

}


FIS_TYPE fis_min(FIS_TYPE a, FIS_TYPE b)

{

    return min(a, b);

}


FIS_TYPE fis_max(FIS_TYPE a, FIS_TYPE b)

{

    return max(a, b);

}
```

```
FIS_TYPE fis_array_operation(FIS_TYPE *array, int size,
_FIS_ARR_OP pfnOp)
{
    int i;
    FIS_TYPE ret = 0;

    if (size == 0) return ret;
    if (size == 1) return array[0];

    ret = array[0];
    for (i = 1; i < size; i++)
    {
        ret = (*pfnOp)(ret, array[i]);
    }

    return ret;
}
//************************************************************
********
// Data for Fuzzy Inference System
//************************************************************
********
// Pointers to the implementations of member functions
_FIS_MF fis_gMF[] =
{
    fis_trimf
};
// Count of member function for each Input
int fis_gIMFCount[] = { 3, 3 };
```

```
// Count of member function for each Output
int fis_gOMFCount[] = { 3, 3, 3, 3 };


// Coefficients for the Input Member Functions
FIS_TYPE fis_gMFI0Coeff1[] = { 20, 20, 24 };
FIS_TYPE fis_gMFI0Coeff2[] = { 23, 25.5, 28 };
FIS_TYPE fis_gMFI0Coeff3[] = { 27, 33, 33 };
FIS_TYPE* fis_gMFI0Coeff[] = { fis_gMFI0Coeff1, fis_gMFI0Coeff2,
fis_gMFI0Coeff3 };
FIS_TYPE fis_gMFI1Coeff1[] = { 0, 0, 50 };
FIS_TYPE fis_gMFI1Coeff2[] = { 40, 60, 80 };
FIS_TYPE fis_gMFI1Coeff3[] = { 70, 100, 100 };
FIS_TYPE* fis_gMFI1Coeff[] = { fis_gMFI1Coeff1, fis_gMFI1Coeff2,
fis_gMFI1Coeff3 };
FIS_TYPE** fis_gMFICoeff[] = { fis_gMFI0Coeff, fis_gMFI1Coeff };


// Coefficients for the Output Member Functions
FIS_TYPE fis_gMFO0Coeff1[] = { 0, 0, 0.5 };
FIS_TYPE fis_gMFO0Coeff2[] = { 0, 0.5, 1 };
FIS_TYPE fis_gMFO0Coeff3[] = { 0.5, 1, 1 };
FIS_TYPE* fis_gMFO0Coeff[] = { fis_gMFO0Coeff1, fis_gMFO0Coeff2,
fis_gMFO0Coeff3 };
FIS_TYPE fis_gMFO1Coeff1[] = { 0, 0, 0.5 };
FIS_TYPE fis_gMFO1Coeff2[] = { 0, 0.5, 1 };
FIS_TYPE fis_gMFO1Coeff3[] = { 0.5, 1, 1 };
FIS_TYPE* fis_gMFO1Coeff[] = { fis_gMFO1Coeff1, fis_gMFO1Coeff2,
fis_gMFO1Coeff3 };
FIS_TYPE fis_gMFO2Coeff1[] = { 0, 0, 0.5 };
FIS_TYPE fis_gMFO2Coeff2[] = { 0, 0.5, 1 };
FIS_TYPE fis_gMFO2Coeff3[] = { 0.5, 1, 1 };
```

```
FIS_TYPE* fis_gMFO2Coeff[] = { fis_gMFO2Coeff1, fis_gMFO2Coeff2,
fis_gMFO2Coeff3 };

FIS_TYPE fis_gMFO3Coeff1[] = { 0, 0, 0.5 };

FIS_TYPE fis_gMFO3Coeff2[] = { 0, 0.5, 1 };

FIS_TYPE fis_gMFO3Coeff3[] = { 0.5, 1, 1 };

FIS_TYPE* fis_gMFO3Coeff[] = { fis_gMFO3Coeff1, fis_gMFO3Coeff2,
fis_gMFO3Coeff3 };

FIS_TYPE** fis_gMFOCoeff[] = { fis_gMFO0Coeff, fis_gMFO1Coeff,
fis_gMFO2Coeff, fis_gMFO3Coeff };


// Input membership function set

int fis_gMFI0[] = { 0, 0, 0 };

int fis_gMFI1[] = { 0, 0, 0 };

int* fis_gMFI[] = { fis_gMFI0, fis_gMFI1};


// Output membership function set

int fis_gMFO0[] = { 0, 0, 0 };

int fis_gMFO1[] = { 0, 0, 0 };

int fis_gMFO2[] = { 0, 0, 0 };

int fis_gMFO3[] = { 0, 0, 0 };

int* fis_gMFO[] = { fis_gMFO0, fis_gMFO1, fis_gMFO2, fis_gMFO3};


// Rule Weights

FIS_TYPE fis_gRWeight[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1 };


// Rule Type

int fis_gRType[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1 };
```

```
// Rule Inputs

int fis_gRI0[] = { 1, 1 };

int fis_gRI1[] = { 1, 2 };

int fis_gRI2[] = { 1, 3 };

int fis_gRI3[] = { 2, 1 };

int fis_gRI4[] = { 2, 2 };

int fis_gRI5[] = { 2, 3 };

int fis_gRI6[] = { 3, 1 };

int fis_gRI7[] = { 3, 2 };

int fis_gRI8[] = { 3, 3 };

int* fis_gRI[] = { fis_gRI0, fis_gRI1, fis_gRI2, fis_gRI3,
fis_gRI4, fis_gRI5, fis_gRI6, fis_gRI7, fis_gRI8 };


// Rule Outputs

int fis_gRO0[] = { 3, 3, 0, 0 };

int fis_gRO1[] = { 3, 2, 0, 2 };

int fis_gRO2[] = { 3, 0, 0, 3 };

int fis_gRO3[] = { 2, 3, 2, 0 };

int fis_gRO4[] = { 2, 2, 2, 2 };

int fis_gRO5[] = { 2, 0, 2, 3 };

int fis_gRO6[] = { 0, 3, 3, 0 };

int fis_gRO7[] = { 0, 2, 3, 2 };

int fis_gRO8[] = { 0, 0, 3, 3 };

int* fis_gRO[] = { fis_gRO0, fis_gRO1, fis_gRO2, fis_gRO3,
fis_gRO4, fis_gRO5, fis_gRO6, fis_gRO7, fis_gRO8 };


// Input range Min

FIS_TYPE fis_gIMin[] = { 20, 0 };
```

```
// Input range Max
FIS_TYPE fis_gIMax[] = { 33, 100 };


// Output range Min
FIS_TYPE fis_gOMin[] = { 0, 0, 0, 0 };


// Output range Max
FIS_TYPE fis_gOMax[] = { 1, 1, 1, 1 };


//************************************************************
*********
// Data dependent support functions for Fuzzy Inference System
//************************************************************
*********
FIS_TYPE fis_MF_out(FIS_TYPE** fuzzyRuleSet, FIS_TYPE x, int o)
{
    FIS_TYPE mfOut;
    int r;


    for (r = 0; r < fis_gcR; ++r)
    {
        int index = fis_gRO[r][o];
        if (index > 0)
        {
            index = index - 1;
            mfOut = (fis_gMF[fis_gMFO[o][index]])(x,
fis_gMFOCoeff[o][index]);
        }
        else if (index < 0)
```

```
      {
                        index = -index - 1;
                        mfOut = 1 -
(fis_gMF[fis_gMFO[o][index]])(x, fis_gMFOCoeff[o][index]);
             }
        else
        {
            mfOut = 0;
        }

        fuzzyRuleSet[0][r] = fis_min(mfOut, fuzzyRuleSet[1][r]);
    }
    return fis_array_operation(fuzzyRuleSet[0], fis_gcR,
fis_max);
}

FIS_TYPE fis_defuzz_centroid(FIS_TYPE** fuzzyRuleSet, int o)
{
    FIS_TYPE step = (fis_gOMax[o] - fis_gOMin[o]) /
(FIS_RESOLUSION - 1);
    FIS_TYPE area = 0;
    FIS_TYPE momentum = 0;
    FIS_TYPE dist, slice;
    int i;

    // calculate the area under the curve formed by the MF
outputs
    for (i = 0; i < FIS_RESOLUSION; ++i){
        dist = fis_gOMin[o] + (step * i);
        slice = step * fis_MF_out(fuzzyRuleSet, dist, o);
```

```
        area += slice;
        momentum += slice*dist;
    }

    return ((area == 0) ? ((fis_gOMax[o] + fis_gOMin[o]) / 2) :
(momentum / area));
}

//************************************************************
********
// Fuzzy Inference System
//************************************************************
********
void fis_evaluate()
{
    FIS_TYPE fuzzyInput0[] = { 0, 0, 0 };
    FIS_TYPE fuzzyInput1[] = { 0, 0, 0 };
    FIS_TYPE* fuzzyInput[fis_gcI] = { fuzzyInput0, fuzzyInput1,
};
    FIS_TYPE fuzzyOutput0[] = { 0, 0, 0 };
    FIS_TYPE fuzzyOutput1[] = { 0, 0, 0 };
    FIS_TYPE fuzzyOutput2[] = { 0, 0, 0 };
    FIS_TYPE fuzzyOutput3[] = { 0, 0, 0 };
    FIS_TYPE* fuzzyOutput[fis_gcO] = { fuzzyOutput0,
fuzzyOutput1, fuzzyOutput2, fuzzyOutput3, };
    FIS_TYPE fuzzyRules[fis_gcR] = { 0 };
    FIS_TYPE fuzzyFires[fis_gcR] = { 0 };
    FIS_TYPE* fuzzyRuleSet[] = { fuzzyRules, fuzzyFires };
    FIS_TYPE sW = 0;
```

```
// Transforming input to fuzzy Input
    int i, j, r, o;
    for (i = 0; i < fis_gcI; ++i)
    {
        for (j = 0; j < fis_gIMFCount[i]; ++j)
        {
            fuzzyInput[i][j] =
                (fis_gMF[fis_gMFI[i][j]])(g_fisInput[i],
fis_gMFICoeff[i][j]);
        }
    }

    int index = 0;
    for (r = 0; r < fis_gcR; ++r)
    {
        if (fis_gRType[r] == 1)
        {
            fuzzyFires[r] = FIS_MAX;
            for (i = 0; i < fis_gcI; ++i)
            {
                index = fis_gRI[r][i];
                if (index > 0)
                    fuzzyFires[r] = fis_min(fuzzyFires[r],
fuzzyInput[i][index - 1]);
                else if (index < 0)
                    fuzzyFires[r] = fis_min(fuzzyFires[r], 1 -
fuzzyInput[i][-index - 1]);
                else
                    fuzzyFires[r] = fis_min(fuzzyFires[r], 1);
            }
```

```
        }
        else
        {
            fuzzyFires[r] = FIS_MIN;
            for (i = 0; i < fis_gcI; ++i)
            {
                index = fis_gRI[r][i];
                if (index > 0)
                    fuzzyFires[r] = fis_max(fuzzyFires[r],
fuzzyInput[i][index - 1]);
                else if (index < 0)
                    fuzzyFires[r] = fis_max(fuzzyFires[r], 1 -
fuzzyInput[i][-index - 1]);
                else
                    fuzzyFires[r] = fis_max(fuzzyFires[r], 0);
            }
        }
        fuzzyFires[r] = fis_gRWeight[r] * fuzzyFires[r];
        sW += fuzzyFires[r];
    }
}
```

Gambar 4.4 Program Fuzzy Inference System

Gambar 4.4 Program *Fuzzy Inference System* hasil konversi file .fis yang diperoleh dari Matlab Fuzzy Logic Designer Toolbox ke Arduino C melalui *website* *http://www.makeproto.com/projects/fuzzy/matlab_arduino_FIST/index.php*

4.2 Program Website

4.2.1 Program Memasukkan Data Sensor Arduino ke Database

```
<?php
      $con = mysqli_connect("localhost", "root", "" ,"arduino" );
      $suhu = $_GET["s"];
      $humid = $_GET["k"];
      $sql = "INSERT INTO suhu(Suhu, Kelembaban) values ('$suhu',
'$humid')";
      $data = mysqli_query($con, $sql);
?>
```

Gambar 4.5 Program PHP Input Data Sensor Arduino

Gambar 4.5 Program untuk memasukkan data sensor Arduino, yaitu sensor suhu dan sensor kelembaban. `$_GET["s"]` dan `$_GET["k"]` merupakan program untuk mengambil variabel data sensor suhu dan kelembaban pada Arduino.

4.2.2 Program Tampilan Tabel Data Sensor Arduino

```
<html>

      <head>

            <script type="text/javascript">

                  function ajaxrunning()

                  {

                        if (window.XMLHttpRequest)

                        {

                              xmlhttp=new XMLHttpRequest();

                        }

                        else

                        {

            xmlhttp =new ActiveXObject("Microsoft.XMLHTTP");

                        }

                        xmlhttp.onreadystatechange=function()

                        {

                              if (xmlhttp.readyState==4 &&
xmlhttp.status==200)                              {


      document.getElementById("inbox").innerHTML =
xmlhttp.responseText;

                              }

                        }
```

```
                        xmlhttp.open("GET","run.php");
                        xmlhttp.send();

                        setTimeout("ajaxrunning()", 1000)

                    }

            </script>

            <style type="text/css">

            h1 {font-family: Verdana;}

            body {font-family: Verdana;

                  font-size: 12px;}

            td, th {font-size: 12px;}

            </style>

</head>

     <body onload="ajaxrunning()">

            <center><h2>DATA SENSOR</h2></center>

            <center><p>Menampilkan 50 data terakhir dari sensor
yang ada dalam database.</p></center>

            <center><div id="inbox"></div></center>

      </body>

</html>
```

Gambar 4.6 Program Tampilan Tabel Data Sensor Arduino

Gambar 4.6 Program untuk menampilkan daftar data sensor Arduino ke dalam tabel. Pada program *function ajaxrunning* terdapat

`window.XMLHttpRequest` yang berfungsi untuk pemeriksaan browser *modern* seperti Chrome, Mozilla Firefox, Internet Explorer 7+, Safari dan Opera. Sedangkan `ActiveXObject("Microsoft.XMLHTTP")` berfungsi untuk pemeriksaan browser jadul seperti Internet Explorer 5 dan 6. Program `xmlhttp.readyState==4 && xmlhttp.status==200` berfungsi untuk pemeriksaan apakah respon server sudah siap diproses atau belum.

4.2.3 Program *Input* Data Sensor dari Database ke Tabel

```
<?php
function getData($sql)
    {
            $con = mysqli_connect("localhost", "root", "",
"arduino");
            $data = mysqli_query($con, $sql);
            return $data;
      }
$query = "SELECT suhu, kelembaban, waktu FROM `suhu` order by no
DESC limit 50 ";
$hasil = getData($query);
echo "<table border='1'>";
echo
"<tr><th>No</th><th>Suhu</th><th>kelembaban</th><th>waktu</th></
tr>";
$a = 0;
while ($data = mysqli_fetch_row($hasil))
{
      $a++;
```

```
        $b = $data[0];

        $c = $data[1];

        $d = $data[2];

        echo
"<tr><td>".$a."</td><td>".$b."</td><td>".$c."</td><td>".$d."</td><
/tr>";

}

echo "</table>";

?>
```

Gambar 4.7 Program Input Data Sensor Arduino ke Tabel

Gambar 4.7 Program yang berfungsi untuk menarik data sensor Arduino dari database ke dalam tabel *website*. Program `"SELECT suhu, kelembaban, waktu FROM `suhu` order by no DESC limit 50 "` berfungsi untuk mengambil data Suhu, Kelembaban, dan Waktu berdasarkan 50 data terakhir. Program `mysqli_fetch_row($hasil)` berfungsi untuk memasukkan data Suhu, Kelembaban, dan Waktu ke dalam baris tabel berdasarkan 50 data terakhir.