BAB IV

HASIL DAN PEMBAHASAN

4.1 Kebutuhan *Hardware* dan *Software*

Untuk dapat menguji sistem ini, diperlukan perangkat keras dan perangkat lunak.

4.1.1 Kebutuhan Perangkat Keras (*Hardware)*

Penulis membutuhkan perangkat keras sebagai berikut:

1. Komputer atau laptop
2. Arduino uno SMD R3
3. Sensor Ultrasonik PING
4. *Real time Clock* (RTC) modul
5. Xbee *series* 2 dengan Xbee *shield*

4.1.2 Kebutuhan Perangkat Lunak (*Software)*

Perangkat lunak yang digunakan dalam pengujian sistem ini menggunakan *software* arduino IDE.

4.2 Pengujian Sistem

Pengujian sistem yang dilakukan penulis merupakan pengujian terhadap perangkat keras dan perangkat lunak secara keseluruhan dan digunakan untuk mengetahui apakah sistem dapat berjalan sesuai yang diharapkan.

4.3 Pengujian *Output* Sensor Ultrasonik PING dan Modul RTC

Pengujian sensor ultrasonik dan modul RTC dilakukan untuk mengetahui informasi ketinggian air beserta waktu disetiap *node* pada prototipe. Terdapat dua *node* sensor yang diletakan pada tempat atau kondisi yang berbeda pada saat pengukuran. Data yang dihasilkan oleh sensor dan RTC diolah oleh arduino supaya dapat menghasilkan nilai ketinggian air dan waktu pada saat data ketinggian air itu diambil.

4.3.1 Peralatan

Peralatan yang dibutuhkan pada setiap *node* dalam pengujian ini adalah sebagai berikut:

1. Arduino uno
2. Sensor ultrasonik PING
3. Modul RTC
4. Komputer atau laptop
5. Kabel USB
6. *Software* arduino IDE
7. Meteran

4.3.2 Prosedur Pengujian

Langkah – langkah dalam prosedur pengujian ini adalah sebagai berikut:

1. Hubungkan sensor ultrasonik dan modul RTC ke arduino uno menggunakan kabel pada pin yang telah ditentukan.
2. Hubungkan arduino uno (*node router*) pada komputer menggunakan kabel USB.

3. Hubungkan *node coordinator* dengan komputer menggunakan kabel USB, *node coordinator* digunakan untuk mengirimkan *request* data ke *node* sensor agar *node* sensor dapat melakukan pembacaan terhadap sensor.
4. Aktifkan komputer atau laptop dan jalankan *software* arduino IDE.
5. *Upload* skrip yang telah dibuat ke dalam arduino uno (*node router*) yang digunakan untuk membaca *output* sensor dan modul RTC.
6. Buka serial monitor pada *software* arduino IDE untuk melihat hasil *output* dari sensor dan RTC
7. Amati dan bandingkan *output* sensor dengan pengukuran manual menggunakan meteran.

4.3.3 Hasil Pengujian

Pada pengujian ini sensor ultrasonik digunakan untuk m*end*apatkan ketinggian air pada prototipe. Selain itu juga melakukan pengukuran ketinggian air secara manual menggunakan meteran. Sensor ultrasonik dan RTC akan mulai membaca ketinggian air dan waktu saat data ketinggian air tersebut diambil ketika arduino uno (*node router*) dinyalakan dan telah menerima *request* data dari *node coordinator*. Pengujian dapat dilakukan seperti Gambar 4.1.

Gambar 4.1 Pengujian Sensor Ultrasonik

Untuk m*end*apatkan hasil pengukuran dari sensor digunakan skrip berikut ini pada setiap *node router*:

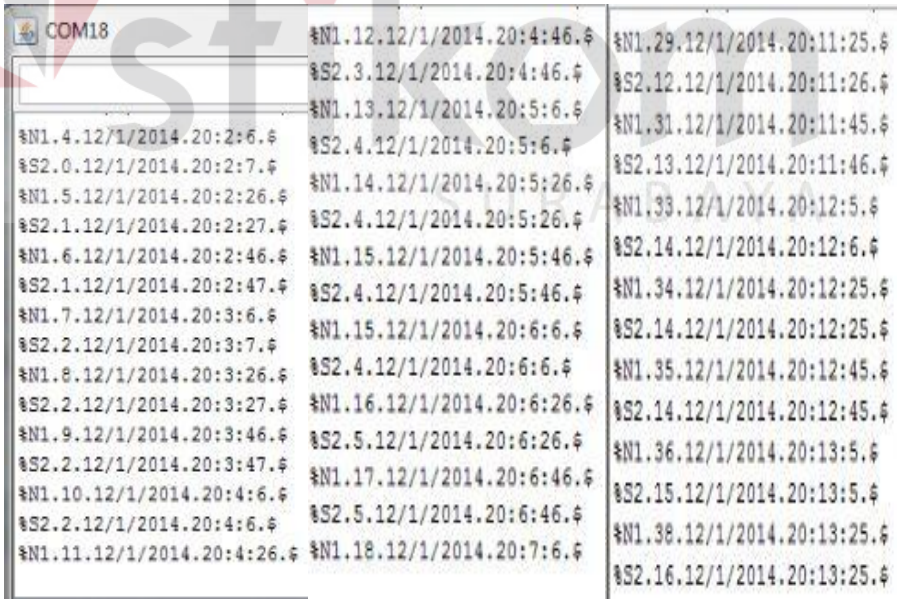
1. *Node router* 1

```
#include <Wire.h>
#include "RTClib.h"
String id,indata,tampung;
const int pingPin = 7;
RTC_DS1307 RTC;

void setup() {
  // initialize serial communication:
  Serial.begin(9600);
  Wire.begin();        // Start the Wire (I2C communications)
  RTC.begin();         // start RTC
  RTC.adjust(DateTime(__DATE__, __TIME__));
  //mengatur waktu pada RTC sesuai dengan waktu pada komputer
  // perintah ini dilakukan hanya sekali untuk RTC
}

long microsecondsToCentimeters(long microseconds)
{
  return microseconds / 29 / 2;
}

void loop(){
  while(Serial.available()>0){
    char data_masuk=(char)Serial.read();
     id += data_masuk;
    if(data_masuk =='\n')
    {
    if((id[0]=='%') && (id[1]=='1'))
     {
       delay(1000);
       sph();
       id="";
     }
     else if ((id[0]=='%') &&  (id[1]=='N'))
      {
        delay(4500);
        sensor2();
      }
     id="";
     }
  }
}

void sph() //fungsi pembacaan sensor ultrasonik dan RTC
 {
    //baca sensor1
    long duration, cm, i;
    pinMode(pingPin, OUTPUT);
    digitalWrite(pingPin, LOW);
    delayMicroseconds(2);
    digitalWrite(pingPin, HIGH);
    delayMicroseconds(5);
```

```
    digitalWrite(pingPin, LOW);
    pinMode(pingPin, INPUT);
    duration = pulseIn(pingPin, HIGH);
    cm = microsecondsToCentimeters(duration);
    i = 40-(cm-7);
    if ((i==47) || (i < 0))
    {
      i=0;
    }
    //kirim data sensor1
    Serial.print('%');
    Serial.print("N1.");
    Serial.print(i);
    Serial.print('.');
    //date
    DateTime now = RTC.now();
    Serial.print(now.day(), DEC);
    Serial.print('/');
    Serial.print(now.month(), DEC);
    Serial.print('/');
    Serial.print(now.year(), DEC);
    //time
    Serial.print('.');
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.print(now.second(), DEC);
    Serial.print('.');
    Serial.print('$');
    Serial.print('\n');
}

void sensor2 () //fungsi sensor2
{
    if (id[0]=='%')
    {
      if ((id[1]=='N') && (id[2]=='2'))
      {
        id[1]='S';
        indata=id;

      }
    Serial.print(id);
    }
  id="";
}
```

Skrip atau teks yang tercetak tebal merupakan program yang dilakukan secara berulang – ulang. Skrip tersebut akan membaca data yang masuk melalui komunikasi nirkabel, jika *node router* 1 menerima pesan “%1” yang dimana pesan tersebut merupakan *request* data dari *node coordinator*, maka setelah delay 1000 ms atau 1 detik, *node router* 1 akan membaca ketinggian air

dan waktu pada prototipe dan mengirimkan data tersebut secara *broadcast* ke *coordinator* dan *node router* 2. Fungsi sph ( ) merupakan skrip pembacaan ketinggian air yang dilakukan oleh sensor dan pembacaan waktu yang dilakukan oleh modul RTC yang kemudian data tersebut dikirimkan secara *broadcast* oleh xbee *series* 2. Data ketinggian air dan waktu pada *node router* 1 memiliki ID “N1”. Jika *node router* 1 menerima pesan “%N” yang mana pesan data tersebut dari *node router* 2, maka dengan delay 4500 ms atau 4,5 detik, *node router* 1 membaca data dari *node router* 2 yang kemudian memforwardnya ke *node coordinator*. Pembacaan data dari *node router* 2 dilakukan pada skrip fungsi sensor2 ( ). Fungsi skrip sensor2 ( ) adalah jika data atau pesan yang diterima mempunyai awalan “%N2”, maka karakter “N” akan diubah menjadi “S”. Gambar 4.2 menunjukkan hasil pengukuran ketinggian air oleh sensor ultrasonik dan waktu oleh RTC pada *node router* 1.



Gambar 4.2 Hasil Pengukuran Sensor Ultrasonik dan Pembacaan Waktu RTC pada *Node router* 1

Format data pada *node router* 1 sebagai berikut %N1.4.12/1/2014.20:2:6.$ dimana % adalah *header* data, N1 adalah ID untuk *node router* 1, 4 adalah ketinggian air dalam cm, 12/1/2014 adalah tanggal, 20:2:6 adalah waktu, dan $ adalah *tail* dari data.

2. *Node router* 2

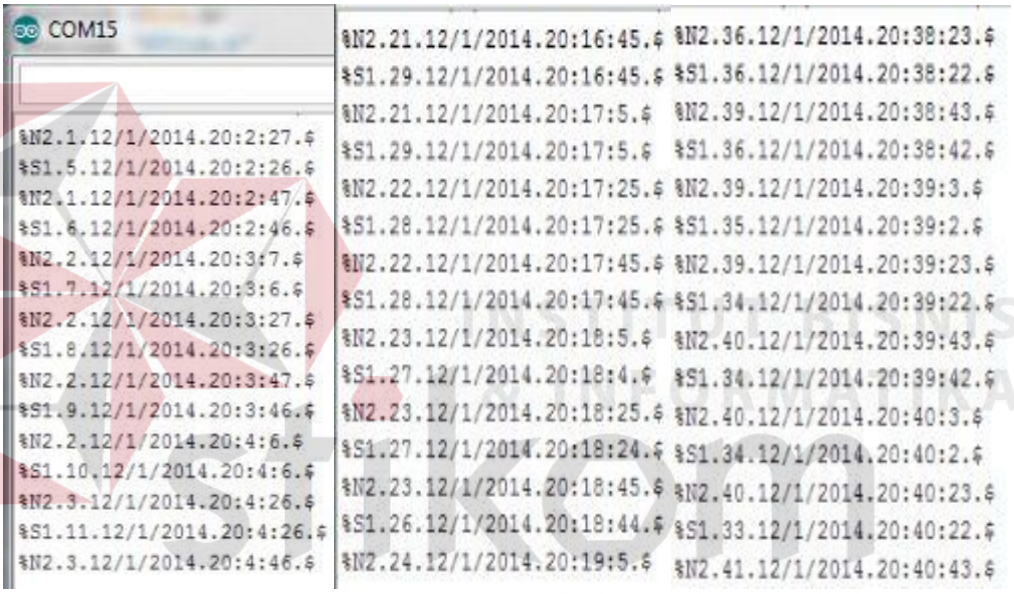
Skrip yang digunakan pada *node router* 2 hampir sama dengan *node router* 1, perbedaannya hanya terletak ID dan delay. ID yang diberikan untuk *node router* 2 adalah "N2" dan perubahan ID *node router* 1 yang melewati *node router* 2 adalah "S1" yang akan dikirim ke *node coordinator*. Delay pada *node router* 2 berbeda dengan *node router* 1, ini dikarenakan agar pada setiap pengiriman data melalui nirkabel (Xbee) tidak bertabrakan satu dengan yang lainnya.

```
#include <Wire.h>
#include "RTClib.h"
String id,indata,tampung;
const int pingPin = 7;
RTC_DS1307 RTC;

void setup() {
  // initialize serial communication:
  Serial.begin(9600);
  Wire.begin();        // Start the Wire (I2C communications)
  RTC.begin();         // start RTC
  RTC.adjust(DateTime(__DATE__, __TIME__));
  //mengatur waktu pada RTC sesuai dengan waktu pada komputer
  // perintah ini dilakukan hanya sekali untuk RTC
}

long microsecondsToCentimeters(long microseconds)
{
  return microseconds / 29 / 2;
}

void loop(){
  while(Serial.available()>0){
    char data_masuk=(char)Serial.read();
     id += data_masuk;
    if(data_masuk =='\n')
    {
    if((id[0]=='%') && (id[1]=='1'))
```

```
      {
        delay(1500);
        sph();
        id="";
      }
      else if ((id[0]=='%') &&  (id[1]=='N'))
       {
         delay(4000);
         sensor2();
       }
      id="";
      }
  }
}

void sph() //fungsi pembacaan sensor ultrasonik dan RTC
 {
    //baca sensor2
    long duration, cm, i;
    pinMode(pingPin, OUTPUT);
    digitalWrite(pingPin, LOW);
    delayMicroseconds(2);
    digitalWrite(pingPin, HIGH);
    delayMicroseconds(5);
    digitalWrite(pingPin, LOW);
    pinMode(pingPin, INPUT);
    duration = pulseIn(pingPin, HIGH);
    cm = microsecondsToCentimeters(duration);
    i = 40-(cm-7);
    if ((i==47) || (i < 0))
    {
      i=0;
    }
    //kirim data sensor2
    Serial.print('%');
    Serial.print("N2.");
    Serial.print(i);
    Serial.print('.');
    //date
    DateTime now = RTC.now();
    Serial.print(now.day(), DEC);
    Serial.print('/');
    Serial.print(now.month(), DEC);
    Serial.print('/');
    Serial.print(now.year(), DEC);
    //time
    Serial.print('.');
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.print(now.second(), DEC);
    Serial.print('.');
    Serial.print('$');
    Serial.print('\n');
}

void sensor1 () //fungsi sensor1
{
```

```
    if (id[0]=='%')
    {
      if ((id[1]=='N') && (id[2]=='1'))
      {
        id[1]='S';
        indata=id;

      }
    Serial.print(id);
    }
  id="";
}
```

Gambar 4.3 menunjukkan hasil pengukuran ketinggian air oleh sensor ultrasonik dan waktu oleh RTC pada *node router* 2.



Gambar 4.3 Hasil Pengukuran Sensor Ultrasonik dan Pembacaan Waktu RTC pada *Node router* 2

Format data pada *node router* 2 sebagai berikut %N2.1.12/1/2014.20:2:27.$ dimana % adalah *header* data, N2 adalah ID untuk *node router* 2, 1 adalah ketinggian air dalam cm, 12/1/2014 adalah tanggal, 20:2:27 adalah waktu, dan $ adalah *tail* dari data.

Dari hasil pengujian sensor ultrasonik pada *node router* 1 dan 2 dapat diketahui bahwa pengukuran ketinggian air menggunakan sensor ultrasonik

terdapat sedikit perbedaan dengan pengukuran secara manual menggunakan meteran. Hasil persentase kesalahan dapat dihitung dengan rumus :

$$error(\%) = \frac{(JarakPembacaanSensor - JarakSebenarnya)}{JarakPembacaanSensor} \times 100$$

Tabel 4.1 merupakan selisih hasil pengujian sensor ultrasonik pada *node router* 1 dan 2 dengan pengujian menggunakan meteran.

Tabel 4.1 Hasil Pengujian Sensor Ultrasonik dan Alat Ukur

	Node router 1	Alat Ukur	Selisih	*Node router* 2	Alat Ukur	Selesih
1	4 cm	4 cm	0	1 cm	1 cm	0
2	8 cm	8 cm	0	2 cm	2 cm	0
3	13 cm	13 cm	0	4 cm	4 cm	0
4	18 cm	18 cm	0	9 cm	9 cm	0
5	20 cm	21 cm	1	19 cm	20 cm	1
6	21 cm	22 cm	1	24 cm	25 cm	1
7	33 cm	34 cm	1	29 cm	30 cm	1
8	35 cm	36 cm	1	36 cm	37 cm	1
9	37 cm	38 cm	1	39 cm	41 cm	2
10	38 cm	39 cm	1	41 cm	43 cm	2
			2.09%			2.56%

Berdasarkan hasil pengujian pada Tabel 4.1 dapat diketahui bahwa pengukuran ketinggian air menggunakan sensor ultrasonik pada prototipe ini mempunyai rata-rata persentase kesalahan 2,09% pada *node router* 1 dan 2,56% pada *node router* 2. Dalam penentuan nilai persentase kesalahan pembacaan sensor ultrasonik dengan rumus dianggap mutlak bernilai positif walaupun dalam hasil perhitungan bernilai negatif karena dalam perhitungan ini penulis hanya ingin mengetahui selisih nilai pembacaan sensor ultrasonik dengan pengukuran secara manual menggunakan meteran. Persentase kesalahan masih dapat ditoleransi untuk sebuah sistem yang bekerja secara cepat dan *real time*. Secara

umum, semakin jauh jarak pembacaan sensor semakin kecil persen kesalahan. Perbedaan pembacaan sensor dengan pengukuran dengan meteran (jarak sebenarnya) dapat disebabkan oleh adanya *noise*. *Noise* dapat berupa gangguan gelombang dari luar yang mempunyai interferensi gelombang dengan frekuensi yang sama, daya pantul objek dan lain-lain sehingga mempengaruhi selisih waktu pemancaran dan penerimaan pantulan gelombang.

4.4 Pengujian Protokol Komunikasi

Pengujian protokol komunikasi ini bertujuan untuk mengetahui apakah pengiriman data dari setiap *node* ke *node* lainnya yang dituju benar dan diterima dengan baik oleh sisi penerima.

4.4.1 Peralatan

Peralatan yang digunakan pada setiap *node* dalam pengujian ini adalah sebagai berikut:

1. Arduino uno dan Xbee *series* 2 beserta shield
2. Sensor ultrasonik PING dan modul RTC (hanya digunakan pada *node router* 1 dan *router* 2)
3. Komputer atau laptop
4. Kabel USB
5. *Software* arduino IDE

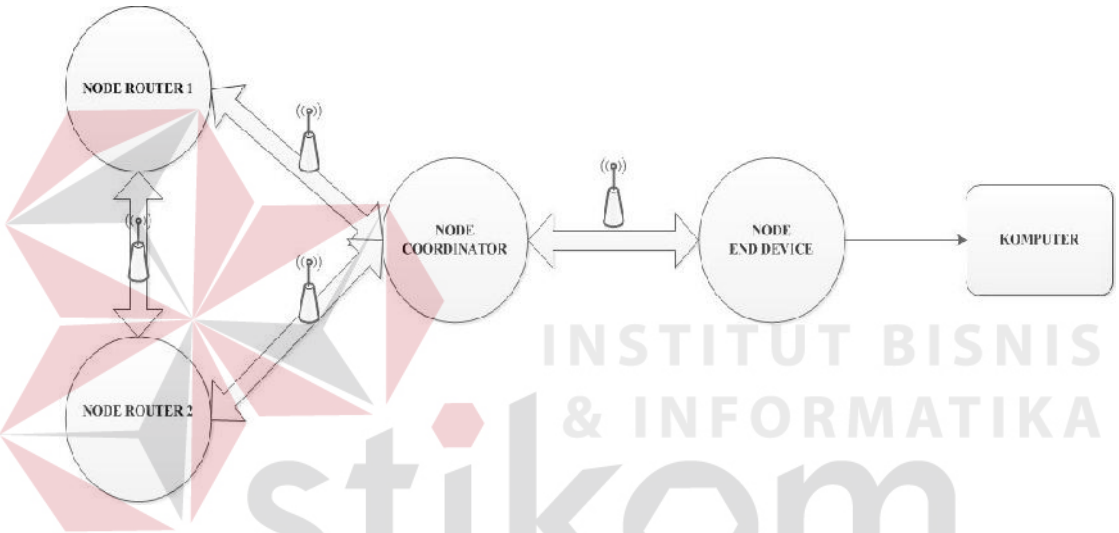
4.4.2 Prosedur Pengujian

Langkah – langkah dalam prosedur pengujian ini adalah sebagai berikut:

1. Hubungkan setiap *node* pada komputer menggunakan kabel USB
2. Aktifkan komputer dan jalankan *software* arduino IDE

3. *Upload* masing-masing skrip yang digunakan untuk setiap *node*nya
4. Buka serial monitor pada *software* arduino IDE untuk melihat data yang dikirim dan diterima di setiap *node*
5. Amati apakah ada data yang *crash* atau bertabrakan sehingga tidak diterima atau sampai pada *node* yang dituju.

4.4.3 Hasil Pengujian



Gambar 4.4 Komunikasi dalam Sistem WSN Peringatan Dini Banjir

Gambar 4.4 merupakan komunikasi yang terjadi dalam sistem WSN untuk peringatan dini terhadap banjir. Terdapat beberapa tahapan proses komunikasi yang terjadi pada setiap *node*.

Proses pertama adalah *node coordinator* akan mengirimkan pesan yang berisi *request* data secara *broadcast* ke *node router* 1 dan 2 selama 20 detik sekali. Berikut cuplikan skrip pada *node coordinator*:

```
void kirim(int dat)
{
  if (dat == 1)
  {
    tampung1="";
    tampung2="";
```

```
    tampung3="";
    tampung4="";
  Serial.print('%');
  Serial.print(dat);
  Serial.print(" : request data :");
  Serial.print('\n');
  }
}

void loop(){
b=1;
while(b<5){
  kirim(b);
  while(Serial.available()>0){
      char data_masuk=(char)Serial.read();
      indata += data_masuk;
      if(data_masuk=='\n'){
        baca();
        indata="";
      }
    }
    indata="";
    if (b == 4)
    {
      banding();
    }
delay(5000);
b++;
}
}
```

Proses kedua, ketika *node router* 1 dan 2 menerima pesan *request* data dari *coordinator*, maka *node router* 1 tersebut akan mengirimkan data ketinggian air dan waktu dengan ID “N1” secara *broadcast*. *Node router* 1 akan mengirimkan data tersebut secara *broadcast* ke *node coordinator* dan *node router* 2. Sedangkan *node router* 2 akan mengirimkan data dengan ID “N2” secara *broadcast* ke *node coordinator* dan *node router* 1. Ketika *node router* 2 menerima data dari *node router* 1, maka *router* 2 merubah ID data dari *node router* 1 menjadi “S1” dan akan memforward data tersebut ke *node coordinator*. Begitu juga dengan *node router* 1 ketika menerima data dari *node router* 2 akan merubah ID data menjadi “S2” dan selanjutnya akan memforward data tersebut ke *node coordinator*.

Proses ketiga, jika semua data yang dikirimkan oleh *node router* 1 dan 2 telah diterima pada *node coordinator*, maka ID semua data tersebut akan diubah

menjadi “C1” dan “C2”. Selanjutnya *node coordinator* akan membandingkan data tersebut. Kemudian hasil dari perbandingan data tersebut akan dikirimkan ke *node router end* device. Berikut cuplikan skrip atau program untuk mengubah ID semua data dan setelah itu membandingkannya:

```
void seleksi()
{
  if ((indata[0]=='%') && (indata[1]=='N') )
  {
      //pemberian ID baru C
        indata[1]='C';
      if (indata[2]=='1')
      {
        tampung1=indata;// tampung data untuk N1
        indata="";// mengkosongkan data
       Serial.print("N1");
        Serial.print(tampung1);
      }
      if (indata[2]=='2')
      {
        tampung3=indata;// tampung data untuk N2
        indata="";// mengkosongkan data
        Serial.print("N2");
        Serial.print(tampung3);
      }
    }
    else if ((indata[0]=='%') && (indata[1]=='S'))
    {
      //pemberian ID baru C
      indata[1]='C';
      if (indata[2]=='1')
      {
        tampung2=indata;// tampung data N1 melalui N2
        indata="";// mengkosongkan data
       Serial.print("N1viaN2");
        Serial.print(tampung2);
      }
      if (indata[2]=='2')
      {
        tampung4=indata;// tampung data N2 melalui N1
        indata="";// mengkosongkan data
       Serial.print("N2viaN1");
        Serial.print(tampung4);
      }
    }
  else
  {
    indata="";
  }
}

void banding()
{
  Serial.println("Hasil Perbandingan :");
```

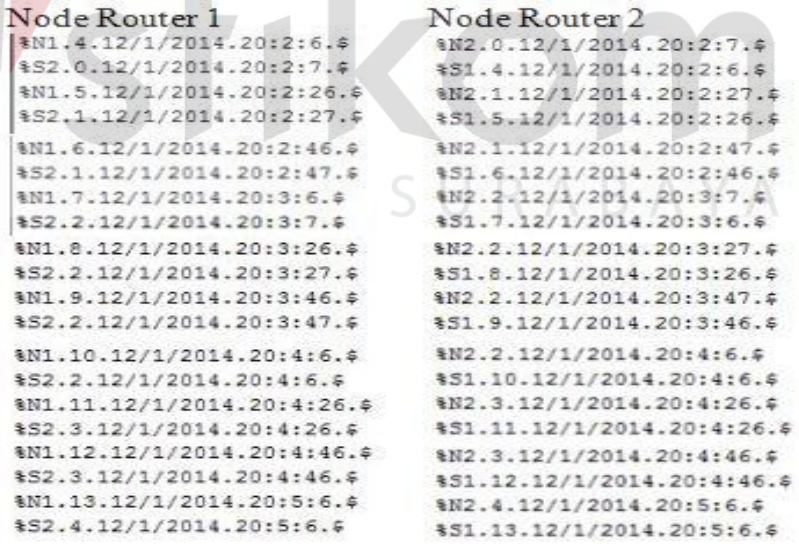
```
  //jika seluruh tampung tidak terisi atau node coordinator tidak
  //menerima data
if ((tampung1 == "") && (tampung2 == ""))
{
  //tampung0 berisi info error pada node 1
  Serial.println(tampung0);
}
if ((tampung3 == "") && (tampung4 == ""))
{
  //tampung berisi info error pada node 2
  Serial.println(tampung);
}
  // jika tampung terisi 1
if ((tampung1 != "") && (tampung2 == ""))
{
  Serial.print(tampung1);
}
if ((tampung1 == "") && (tampung2 != ""))
{
  Serial.print(tampung2);
}
if ((tampung3 != "") && (tampung4 == ""))
{
  Serial.print(tampung3);
}
if ((tampung3 == "") && (tampung4 != ""))
{
  Serial.print(tampung4);
}
// dua tampung terisi
if ((tampung1 != "") && (tampung2 != ""))
{
  if (tampung1 == tampung2)
  {
    Serial.print(tampung1);
  }
}

if ((tampung3 != "") && (tampung4 != ""))
{
  if (tampung3 == tampung4)
  {
    Serial.print(tampung3);
  }
}
}
```

Dari skrip tersebut maka kondisi perbandingan yang terjadi adalah seperti berikut:

1. Jika *node coordinator* tidak menerima data dari *node router* 1 atau 2, maka *node coordinator* akan mengirimkan pesan *error* yang terjadi pada *node router* 1 atau 2 ke *node router end device*.

2. Jika *node coordinator* hanya menerima salah satu data dari *node router* 1 atau 2, maka *node* ini akan membandingkan data tersebut. Jika salah satu tampung terisi, *node coordinator* akan mengirim data sesuai dengan tampung yang diterima ke *node router end device*.
3. Jika *node coordinator* menerima semua data dari *node router* 1 dan 2 atau tampung1, 2, 3, 4 terisi, maka akan dilakukan perbandingan data. Jika data tampung1 sama dengan tampung2, maka *node* ini akan mengirimkan data salah satu dari tampung ke *node router end device* (dalam skrip ini akan mengirimkan data tampung1). Jika data tampung3 sama dengan tampung4, maka *node* ini akan mengirimkan data salah satu dari tampung ke *node router end device* (dalam skrip ini akan mengirimkan data tampung3).

Gambar 4.5 menunjukkan hasil pengiriman dan penerimaan data antar *node router* yang ditampilkan di serial monitor pada setiap *node*.



Gambar 4.5 Pengiriman dan Penerimaan Data antar *Node router*

Berdasarkan Gambar 4.5 dapat dilihat bahwa data ketinggian air dan waktu yang berasal dari *node router* 1 memiliki ID “N1” dan *node router* 2

memiliki ID “N2”. Kemudian data dengan ID “S2” pada *node router* 1 merupakan data yang berasal dari *node router* 2 yang diterima oleh *node router* 1, perhatikan nilai ketinggian air dan waktu dengan ID “N2” pada *node router* 2 dan ID “S2” pada *node router* 1 akan bernilai sama. Begitu juga data dengan ID “S1” pada *node router* 2 merupakan data yang berasal dari *node router* 1 yang diterima oleh *node router* 2, perhatikan nilai ketinggian air dan waktu dengan ID “N1” pada *node router* 1 dan ID “S1” pada *node router* 2 akan bernilai sama. Data tersebut selanjutnya akan diteruskan atau dikirimkan ke *node coordinator*.

Untuk hasil pengiriman dan penerimaan data antara *node router* dan *node coordinator* serta hasil perbandingan data diperlihatkan pada Gambar 4.6

Gambar 4.6 Hasil Pengiriman/Penerimaan/Perbandingan Data pada *Node coordinator*

Gambar 4.6 merupakan hasil pengiriman / penerimaan dan perbandingan data pada *node coordinator*. Pada *node coordinator* terdapat beberapa data yang diterima dan dikirimkan antara lain:

1. %1 : request data : merupakan request data yang dikirimkan secara broadcast ke *node router* 1 dan 2.
2. Data N1%C1.4.12/1/2014.20:2:6.$ merupakan data yang diterima berasal dari *node router* 1.
3. Data N2%C2.0.12/1/2014.20:2:7.$ merupakan data yang diterima berasal dari *node router* 2.
4. Data N1viaN2%C1.4.12/1/2014.20:2:6.$ merupakan data yang diterima berasal dari *node router* 1 melalui *node router* 2.
5. Data N2viaN1%C2.0.12/1/2014.20:2:7.$ merupakan data yang diterima berasal dari *node router* 2 melalui *node router* 1.
6. Data %C1.4.12/1/2014.20:2:6.$ dan %C2.0.12/1/2014.20:2:7.$ merupakan hasil perbandingan data yang dikirimkan ke *node router end* device.

Berdasarkan Gambar 4.5 dan Gambar 4.6 dapat diketahui apakah terjadi data *crash* atau tabrakan data pada saat pengiriman data antar *node router* 1, 2, dan *node coordinator* sehingga mempengaruhi penerimaan data pada *node* yang dituju. Hasil dalam pengujian protokol komunikasi ini tidak ada data yang *crash* atau bertabrakan karena skrip atau program disetiap *node* mempunyai delay yang berbeda-beda untuk mengirimkan datanya.

4.5 Pengujian Jarak Kemampuan Pengiriman dan Penerimaan Xbee

Pada pengujian ini bertujuan untuk mengetahui kemampuan jangkauan atau jarak pengiriman dan penerimaan data pada WSN menggunakan Xbee S2.

4.5.1 Peralatan

Peralatan yang digunakan pada setiap *node* dalam pengujian ini adalah sebagai berikut:

1. Arduino uno
2. Xbee pro S2 dan Xbee S2 beserta *shield*
3. Sensor ultrasonik PING dan modul RTC pada *node router*
4. Kabel USB
5. *Node coordinator* terhubung pada komputer
6. Baterai 9v (pada *node router*)

4.5.2 Prosedur Pengujian

Langkah-langkah dalam pengujian ini sebagai berikut:

1. Hubungkan *node coordinator* pada komputer dengan kabel USB.
2. aktifkan komputer dan jalankan program arduino IDE.
3. *Upload* skrip atau program sebagai *coordinator* yang digunakan pada *node coordinator.*
4. Setelah *upload* program selesai, buka serial monitor.
5. *Upload* skrip atau program untuk *router* 1 dan *router* 2 seperti pada saat *upload* program *coordinator*.
6. Hubungkan *node router* 1 dan *router* 2 dengan catu daya atau baterai pada masing-masing *node.*

7. Ukur jarak antar Xbee, dan carilah jangkauan atau jarak maksimal pengiriman dan penerimaan data.

4.5.3 Hasil Pengujian

Dari langkah – langkah pengujian komunikasi data pada Xbee yang telah dilakukan di luar ruangan (*Outdoor Area*) dan di dalam ruangan (*Indoor Area*) didapatkan hasil jarak atau jangkauan pengiriman dan penerimaan data sebagai berikut:

Tabel 4.2 Hasil Pengujian Jarak Pengiriman dan Penerimaan data pada Xbee (*Outdoor*)

No.	Jarak (meter)	Keterangan
1.	10	Ok
2.	20	Ok
3.	30	Ok
4.	40	Ok
5.	50	Ok
6.	60	Ok
7.	70	Ok
8.	80	Ok
9.	90	Ok
10.	100	Ok
11.	101	Gagal
12.	102	Gagal
13.	103	Gagal
14.	104	Gagal
15.	105	Gagal

Tabel 4.3 Hasil Pengujian Jarak Pengiriman dan Penerimaan data pada Xbee (*indoor*)

No.	Jarak (meter)	Keterangan
1.	10	Ok
2.	20	Ok
3.	30	Ok
4.	40	Ok
5.	41	Gagal
6.	42	Gagal

Pada kondisi *outdoor* dengan jarak 1 - 100 meter pengiriman dan penerimaan data antar *node* dapat berkomunikasi dengan baik, tetapi pada jarak 101 meter tidak dapat berkomunikasi. Artinya komunikasi antar *node* tersebut terputus, sehingga *node* yang dituju tidak dapat menerima data yang dikirimkan oleh *node* lain.

Sedangkan pada kondisi *indoor* dengan jarak 1 – 40 meter dapat berkomunikasi dengan baik. Setelah jarak melebihi 40 meter komunikasi terputus. Dengan demikian hasil yang didapat sama dengan spesifikasi pada *datasheet* xbee S2.