

BAB III

PERANCANGAN DAN PEMBUATAN SISTEM

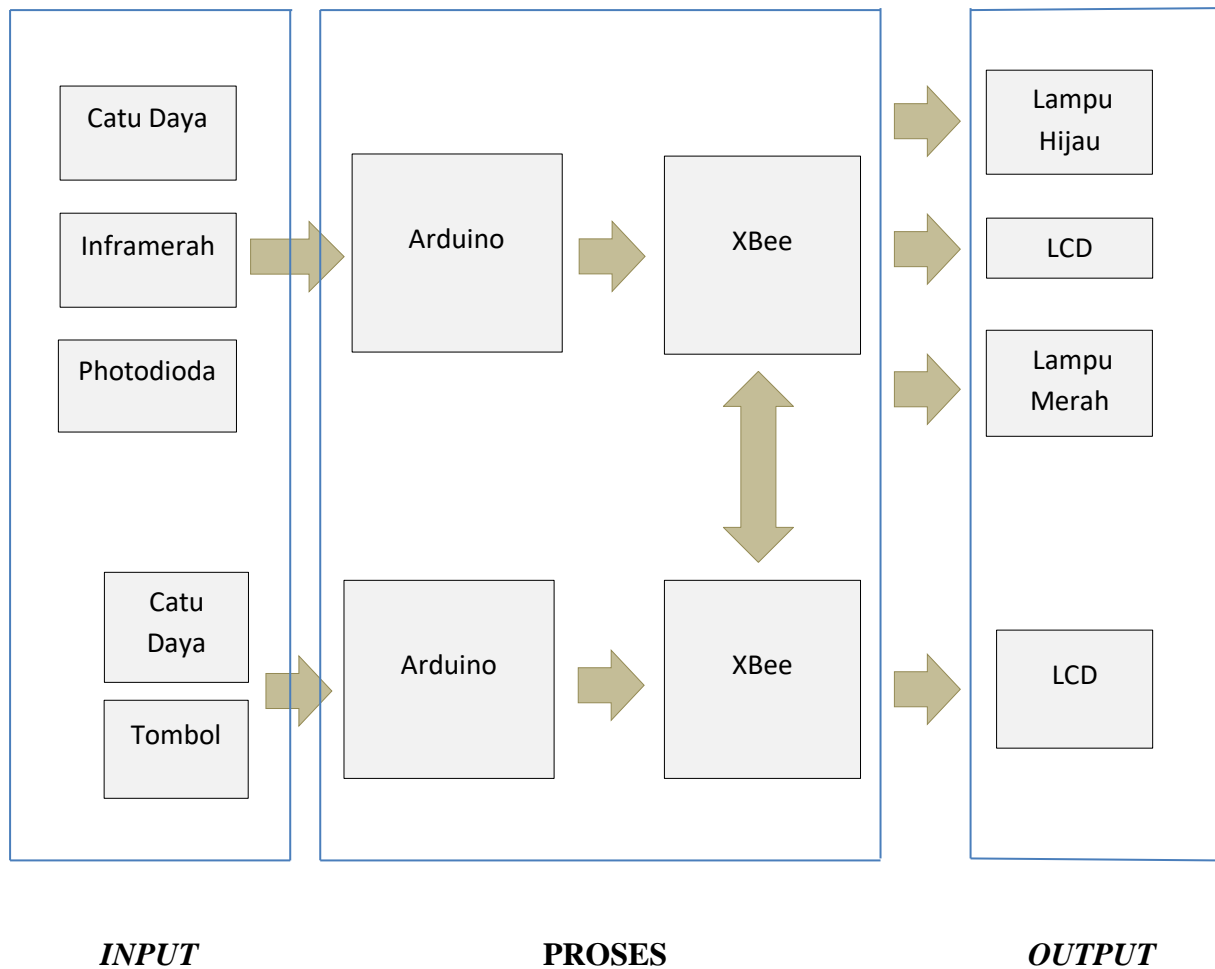
Pada bab III ini membahas tentang perancangan dan pembuatan sistem pengendali pada lampu lalu lintas menggunakan komunikasi ZigBee menggunakan sensor infra merah dan *photodiode* sebagai penerima cahaya infra merah berbasis Arduino sebagai *master* dan *slave* berupa perancangan perangkat keras (*hardware*) maupun perangkat lunak (*software*).

Perangkat keras meliputi perancangan sensor infra merah dan *photodiode*, perancangan sistem minimum Arduino, perancangan *Light Emitting Diode* (LED), perancangan LCD 16x2, dan perancangan catu daya.

3.1 PERANCANGAN SISTEM

Perancangan sistem rancang bangun pengendali lampu lalu lintas menggunakan komunikasi ZigBee berbasis Arduino jika diimplementasikan bertujuan untuk mengurangi terjadinya kemacetan lalu lintas, meminimalkan pemborosan waktu dan biaya, penghematan bahan bakar kendaraan, meminimalkan polusi akibat gas buang kendaraan bermotor serta membantu aparat lalu lintas dalam mengatur di perempatan jalan.

Pembuatan sistem lalu lintas yang menggunakan komunikasi komunikasi ZigBee ini terdiri dari beberapa rangkaian yaitu rangkaian sensor infra merah, rangkaian LED, dan rangkaian LCD 16x2. Untuk catu daya menggunakan *transformator* dengan *input* AC 110V/220V dan *output* DC 5 V yang dihubungkan dengan *stop* kontak. Berdasarkan gambar 3.1 merupakan blok diagram *traffic lights system*. Pada blok diagram terdiri dari *input*, proses, dan *output*. Untuk rangkaian *input*, dapat dilihat pada blok diagram terdiri dari sensor infra merah yang berfungsi untuk mendeteksi adanya obyek yang melintas dan pada *fase* ini akan dilakukan penghitungan. Sebagai penerima cahaya infra merah menggunakan *photodiode*.



Gambar 3.1 Blok Diagram Sistem

Untuk rangkaian proses menggunakan Mikrokontroler yang berfungsi sebagai pemroses dan pengendali seluruh sistem. Dalam hal ini menggunakan Arduino sebagai *master* dan *slave*. *Master* difungsikan sebagai pengendali seluruh sistem dan *slave* bertindak sebagai bawahannya. *Master* akan memproses data *inputan* dari pendeteksian sensor infra merah dan setelah diolah sedemikian rupa, maka *master* akan mengirimkan hasilnya berupa nyala lampu hijau ke *slave*.

Untuk rangkaian *output*-nya menggunakan LED dan LCD. LED dan digunakan sebagai indikator nyala lampu dan waktu nyala lampu. Untuk

LCD akan menampilkan berupa jumlah antrian di ruas jalan setelah proses selesai.

Sesuai gambar 3.1 diatas dapat dilihat penjelasannya pada Tabel 3.1 bagian *Master* dan tabel 3.2 terdapat penjelasan pada bagian *Slave*, sebagai berikut :

Tabel 3.1 Penjelasan diagram blok sistem *Master*

No	Bagian	Keterangan
1	Catu Daya	Sebagai sumber listrik <i>Master</i>
2	Arduino Uno R3	Sebagai pusat pengendali untuk <i>Slave</i>
3	Modul XBee	Sebagai alat komunikasi antara <i>Master</i> ke <i>Slave</i>
4	LCD	Sebagai tampilan durasi waktu
5	Tombol	Untuk men- <i>setting</i> manual durasi waktu yang dibutuhkan pada <i>Slave</i> , menambah dan mengurangi rasio kendaraan, menyimpan durasi waktu dan menyimpan rasio kendaraan yang telah kita tentukan.

Tabel 3.2 Penjelasan diagram blok sistem *Slave*

No	Bagian	Keterangan
1	Catu Daya	Sebagai sumber listrik <i>Master</i>
2	Arduino Uno R3	Sebagai pusat pengendali untuk <i>Slave</i>
3	Modul XBee	Sebagai alat komunikasi antara <i>Master</i> ke <i>Slave</i>
4	LCD	Sebagai tampilan durasi waktu
5	Infra merah dan Photodioda	Untuk mendeteksi objek yang melintas pada infra merah dan <i>photodioda</i>

No	Bagian	Keterangan
6	LED Warna Hijau	Sebagai indikator nyala LED warna hijau
7	LED Warna Merah	Sebagai indikator nyala LED warna merah

Cara kerja dari sistem pengendali lampu lalu lintas ini adalah mikrokontroler *master* akan mengambil data dari pendeteksian sensor infra merah yang dipasang di ruas jalan dan sensor infra merah akan bekerja apabila ada obyek yang melintas. Setelah antrian sudah didapatkan, maka mikrokontroler *master* akan memproses data tersebut, apakah berubah atau tidak. Penentuan waktu lampu dilihat dari kriteria hasil *scanning* yang ada. Dengan *setting default* lampu hijau menyala adalah 10 detik dan lampu merah menyala adalah 13 detik. Hasil dari pem-prosesan berupa data akan diteruskan ke mikrokontroler *slave* berupa nyala lampu hijau serta *timer*-nya. Setelah satu siklus berjalan maka jumlah antrian di ruas akan ditampilkan oleh LCD. Proses penghitungan obyek adalah pada saat lampu merah menyala.

3.2 PERANCANGAN DAN PEMBUATAN *HARDWARE*

Perancangan hardware meliputi pembuatan perangkat keras sistem pengendali pada lampu lalu lintas menggunakan sensor infra merah sebagai pendeteksian obyek yang melintas meliputi pembuatan rangkaian secara *schematic* dan perhitungan dari komponen yang digunakan. Perancangan dimulai dengan perancangan rangkaian sensor infra merah, perancangan rangkaian LED, perancangan rangkaian LCD 16x2, perancangan catu daya, serta rangkaian keseluruhan.

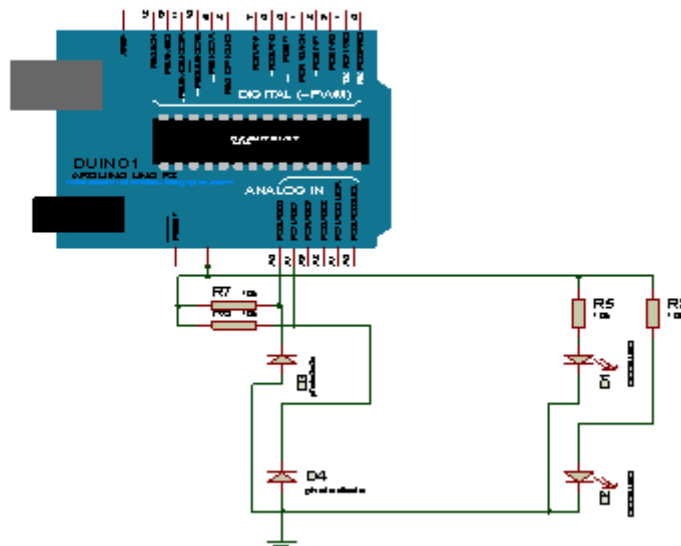
3.2.1. PERANCANGAN RANGKAIAN *SLAVE*

3.2.1.1 Perancangan Rangkaian Sensor Infra Merah

Pada sistem pengendali lampu lalu lintas ini menggunakan sensor infra merah. Sensor ini bekerja sama dengan *photodiode*. *Photodiode* adalah perangkat yang sangat sensitif dengan sinar infra

merah, sehingga jika ada yang menghalangi pancaran sinar infra merah ke *photodiode* maka akan bekerja dan dalam hal ini akan dihitung satu begitu seterusnya.

Prinsip kerja dari sensor infra merah tersebut adalah untuk mendeteksi objek yang melintas. Dimana *photodiode* yang sensitif akan pancaran infra merah tidak menerima pancaran infra merah, sedangkan kondisi *photodiode* masih menerima pancaran infra merah sehingga kondisi ini disebut dengan kondisi normal. Pada gambar 3.2 merupakan skematik dari rangkaian sensor infra merah.



Gambar 3.2 Skematik Arduino dan Rangkaian Sensor Infra Merah

Sesuai dengan gambar 3.2, dapat dilihat penjelasannya pada tabel 3.3, sebagai berikut :

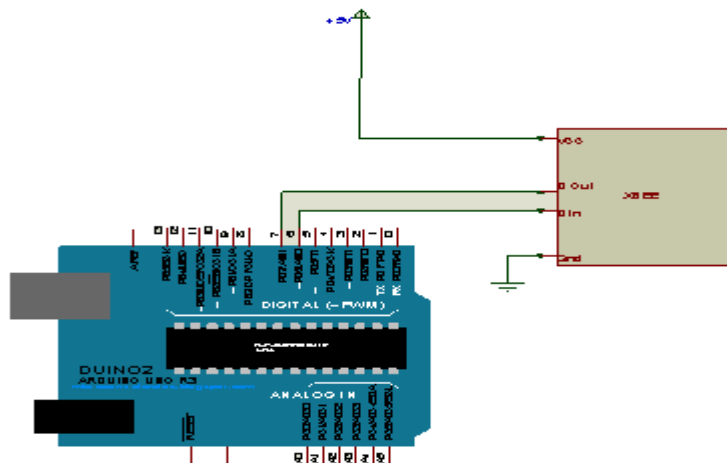
Tabel 3.3 Koneksi Antara Arduino dengan Sensor Inframerah dan *Photodiode*

No	Kaki Arduino	<i>Photodiode</i>
1	A0	D3
2	A1	D4

Dapat dijelaskan pada saat intensitas infra merah yang diterima *Photodiode* besar maka tahanan *Photodiode* menjadi kecil, sedangkan jika intensitas Infra merah yang diterima *Photodiode* kecil maka tahanan yang dimiliki *photodiode* besar.

3.2.1.2 Perancangan Rangkaian XBee Slave

Pada perancangan XBee tersebut terdapat suatu komunikasi yaitu komunikasi ZigBee. Pada tugas akhir ini komunikasi ZigBee berfungsi untuk saling mengetahui antara dari segi *master* dan segi *slave*. Dari segi master sendiri komunikasi ZigBee berfungsi untuk mengirim program durasi waktu yang akan dilakukan pada segi *slave*, sedangkan dari segi slave tersebut fungsi dari komunikasi ZigBee yaitu untuk menerima settingan *default* dari *master* yang akan diproses oleh *slave*, kemudian dari segi *slave* tersebut memproses terhadap objek yang melintas pada ruas jalan kemudian *slave* mengirim keadaan ruas jalan tersebut kepada *master* yang menggunakan perantara komunikasi ZigBee. Gambar 3.3 merupakan skematik XBee dari segi *slave*.



Gambar 3.3 Skematik Arduino dan Rangkaian XBee *Slave*.

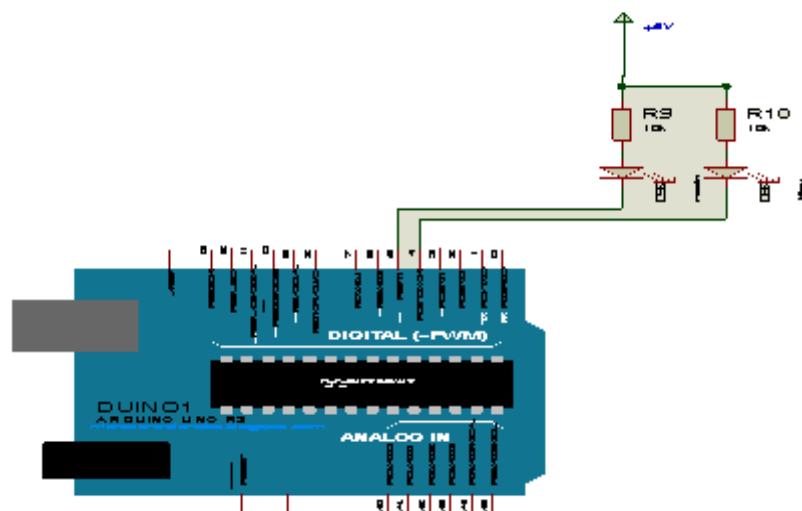
Sesuai dengan gambar 3.3, dapat dilihat penjelasannya pada tabel 3.4, sebagai berikut :

Tabel 3.4 Koneksi antara Arduino dengan XBee *Slave*.

No	Kaki Arduino	Kaki XBee
1	5 Volt	Masukan pada XBee di port Vcc
2	PD7	Masukan pada XBee di port Dout
3	PD6	Masukan pada XBee di port Din
4	GND	Masukan pada XBee di port GND

3.2.1.3 Perancangan Rangkaian LED

Pada sistem pengendali lampu lalu lintas ini, LED digunakan sebagai indikator *traffic lights*. Pada gambar 3.3 yaitu rangkaian LED yang digunakan sebagai indikator *traffic lights* di ruas jalan. LED akan menyala sesuai dengan perintah yang telah diinstruksikan oleh mikrokontroler. Dari rangkaian tersebut menggunakan 2 buah resistor dan 2 buah LED dengan warna merah dan warna hijau. Pada LED warna hijau terhubung pada PD4 sedangkan lampu warna merah terhubung pada PD5.



Gambar 3.4 Skematik Arduino dan Rangkaian LED

Pada gambar 3.4 dapat dilihat penjelasannya pada tabel 3.5, sebagai berikut :

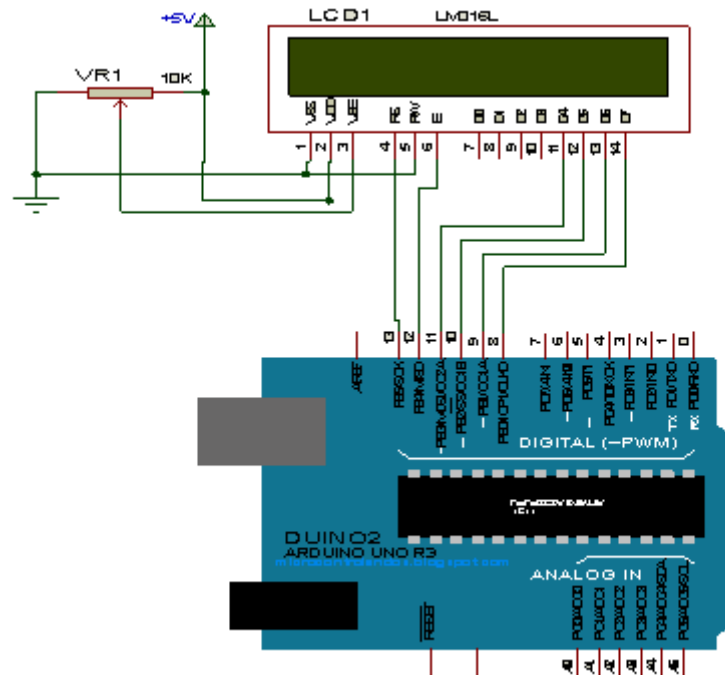
Tabel 3.5 Koneksi antara Arduino dengan Rangkaian LED warna merah dan warna hijau

No	Kaki Arduino	Kaki LED
1	PD5	LED warna merah
2	PD4	LED warna hijau

Pada skematik rangkaian LED mengenai rancang bangun pengendali lampu lalu lintas menggunakan komunikasi ZigBee berbasis Arduino ini, terdapat 2 buah LED, masing-masing warna hijau dan warna merah.

3.2.1.4 Perancangan Rangkaian LCD 16x2 dengan Arduino *Slave*

Untuk pembuatan rancang bangun sistem pengendali lampu lalu lintas pada Tugas Akhir ini menggunakan LCD 16x yang terdiri dari 16 karakter dua baris. Dapat dikatakan bahwa LCD ini menampilkan 32 karakter, dengan masing-masing baris terdiri dari 16 karakter. Pada Tugas Akhir ini LCD dipasang pada sisi *master* yang berfungsi sebagai *display* yang akan menampilkan berupa *timer traffic* dua fasa yaitu *timer* merah dan hijau, serta antrian masing-masing di ruas jalan yang telah dideteksi oleh sensor infra merah. Gambar 3.5 Merupakan rangkaian LCD 16x2 pada *Slave*.



Gambar 3.5 Skematik Arduino dan Rangkaian LCD Pada *Slave*.

LCD 16x2 memiliki 16 pin, untuk dapat mengkonfigurasi LCD 16x2 dengan mikrokontroler Arduino maka pin I/O mikrokontroler harus dikonfigurasi dengan PIN LCD 16x2. Selain untuk hubungan antarmuka dengan mikrokontroler, pada LCD terdapat PIN Vcc yang dihubungkan dengan *power* sebesar 5 volt dan pin *ground* yang dihubungkan ke *ground*. Pada gambar 3.5 dapat dilihat penjelasannya pada tabel 3.6, sebagai berikut :

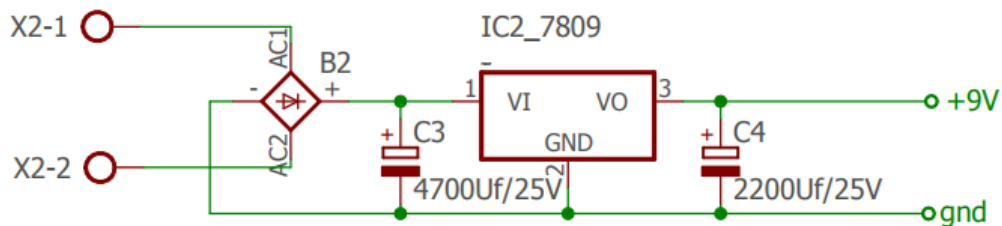
Tabel 3.6 Koneksi antara Arduino dengan LCD *Slave*

No	Kaki Arduino	Kaki Pada LCD
1	PB0	D7
2	PB1	D6
3	PB2	D5
4	PB3	D4
5	PB4	E

No	Kaki Arduino	Kaki Pada LCD
6	PB5	RS

3.2.1.5 Perancangan Catu Daya *Slave*

Catu daya merupakan energi yang diperlukan oleh sistem pengendali lampu lalu lintas ini agar dapat bekerja. Catu daya yang digunakan adalah sebesar 9 volt dengan mengadopsi tegangan kerja mikrokontroler Arduino. Sumber energi catu daya didapatkan dari PLN yang kemudian diubah menjadi tegangan DC dengan menggunakan adapter DC sebesar 9 volt dan rangkaian IC Regulator untuk menstabilkan tegangan menjadi 9 volt. Gambar 3.6 merupakan rangkaian catu daya pada Tugas Akhir ini.



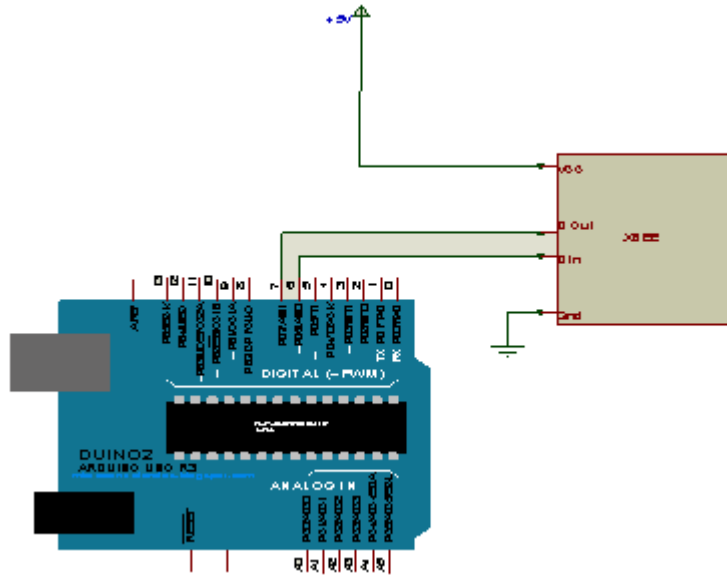
Gambar 3.6 Rangkaian Catu Daya *Slave*

3.2.2. PERANCANGAN RANGKAIAN *MASTER*

3.2.2.1. Perancangan Rangkaian XBee *Master*

Pada perancangan XBee tersebut terdapat suatu komunikasi yaitu komunikasi ZigBee. Pada tugas akhir ini komunikasi ZigBee berfungsi untuk saling mengetahui antara dari segi *master* dan segi *slave*. Dari segi master sendiri komunikasi ZigBee berfungsi untuk mengirim program durasi waktu yang akan dilakukan pada segi *slave*, sedangkan dari segi *slave* tersebut fungsi dari komunikasi ZigBee yaitu untuk menerima *setting-an* dari *master* yang akan

diproses oleh *slave*, kemudian dari segi *slave* tersebut memproses terhadap objek yang melintas pada ruas jalan kemudian *slave* mengirim keadaan ruas jalan tersebut kepada *master* yang menggunakan perantara komunikasi ZigBee. Gambar 3.7 merupakan skematik XBee dari segi *master* dan gambar 3.7 merupakan skematik dari segi *Master*.



Gambar 3.7 Skematik Arduino dan Rangkaian XBee *Master*

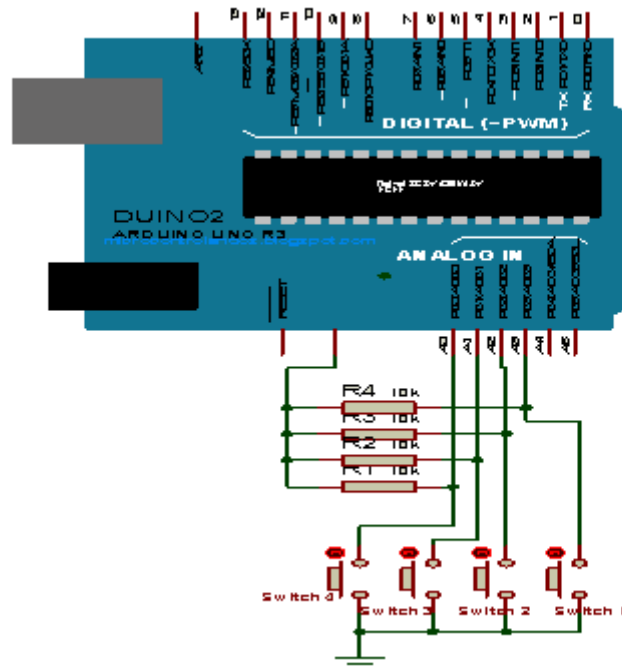
Sesuai dengan gambar 3.7, dapat dilihat penjelasannya pada tabel 3.7, sebagai berikut :

Tabel 3.7 Koneksi antara Arduino dengan XBee *Master*

No	Kaki Arduino	Kaki XBee
1	5 Volt	Masukan pada XBee di port Vcc
2	PD7	Masukan pada XBee di port Dout
3	PD6	Masukan pada XBee di port Din
4	GND	Masukan pada XBee di port GND

3.2.2.2. Perancangan Rangkaian Tombol

Pada tugas akhir ini tombol yang ada terdapat pada bagian *master*. Fungsi tombol pada bagian master yaitu untuk mengatur durasi waktu terhadap lampu merah dan lampu hijau yang tergantung pada kondisi ruas jalan tersebut. Gambar 3.8 terdapat skematik rangkaian tombol.



Gambar 3.8 Skematik Arduino dan Rangkaian Tombol

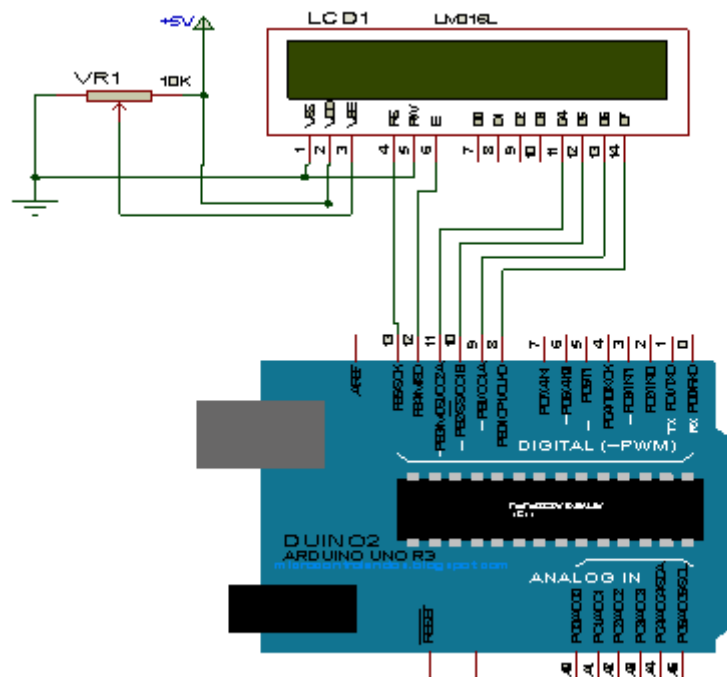
Dari gambar 3.8 dapat dilihat penjelasannya pada tabel 3.8, sebagai berikut :

Tabel 3.8 Koneksi antara Arduino dengan *Switch*

No	Kaki Arduino	Keterangan <i>Switch</i>	Fungsi
1	A0	<i>Switch 1</i>	Edit Rasio
2	A2	<i>Switch 2</i>	<i>Menu dan Exit</i>
3	A3	<i>Switch 3</i>	Mengurangi Waktu dan Rasio
4	A4	<i>Switch 4</i>	Tambah Waktu dan Simpan

3.2.2.3 Perancangan Rangkaian LCD 16x2 dengan Arduino Master

Untuk pembuatan rancang bangun sistem pengendali lampu lalu lintas pada Tugas Akhir ini menggunakan LCD 16x yang terdiri dari 16 karakter dua baris. Dapat dikatakan bahwa LCD ini menampilkan 32 karakter, dengan masing-masing baris terdiri dari 16 karakter. Pada Tugas Akhir ini LCD dipasang pada sisi *master* yang berfungsi sebagai *display* yang akan menampilkan berupa *timer traffic* dua fasa yaitu *timer* merah dan hijau, serta antrian masing-masing di ruas jalan yang telah dideteksi oleh sensor infra merah. Gambar 3.9 Merupakan rangkaian LCD 16x2 pada *Master*.



Gambar 3.9 Skematik Arduino dan Rangkaian LCD 16x2 Pada *Master*

LCD 16x2 memiliki 16 pin, untuk dapat mengkonfigurasi LCD 16x2 dengan mikrokontroler Arduino maka pin I/O mikrokontroler harus dikonfigurasi dengan PIN LCD 16x2. Selain untuk hubungan antarmuka dengan mikrokontroler, pada

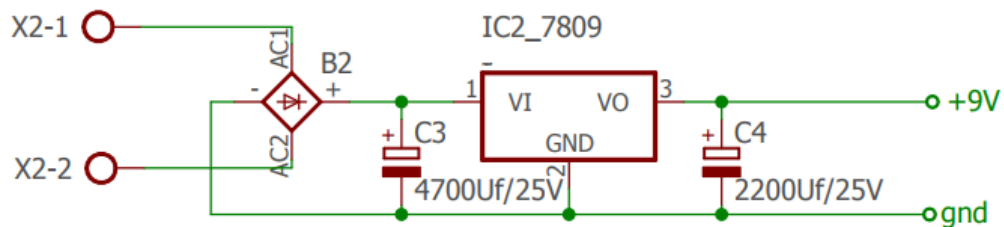
LCD terdapat PIN Vcc yang dihubungkan dengan *power* sebesar 5 volt dan pin *ground* yang dihubungkan ke *ground*. Pada gambar 3.9 dapat dilihat penjelasannya pada tabel 3.9, sebagai berikut :

Tabel 3.9 Koneksi antara Arduino dengan LCD *Master*

No	Kaki Arduino	Kaki Pada LCD
1	PB0	D7
2	PB1	D6
3	PB2	D5
4	PB3	D4
5	PB4	E
6	PB5	RS

3.2.2.4. Perancangan Catu Daya *Master*

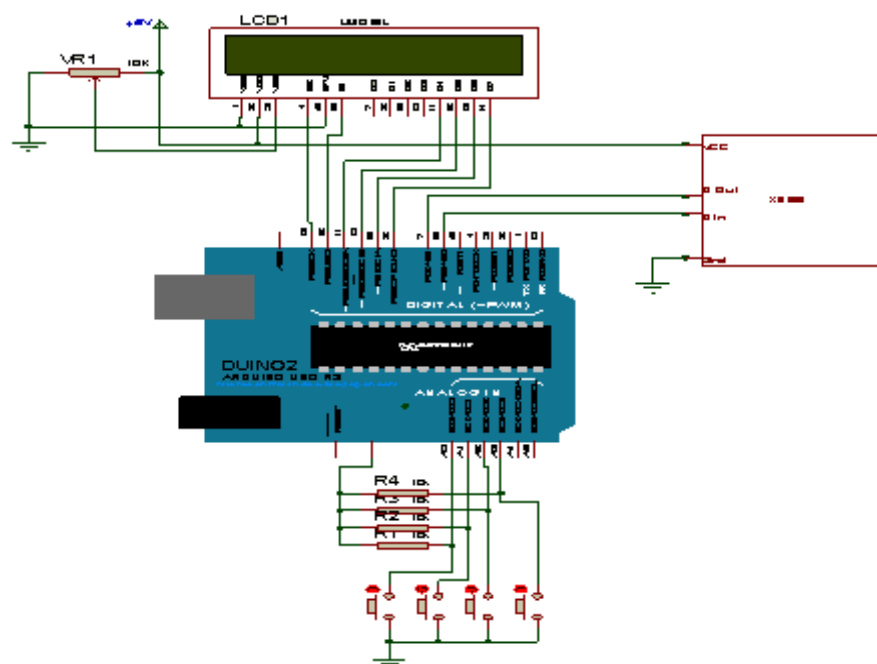
Catu daya merupakan energi yang diperlukan oleh sistem pengendali lampu lalu lintas ini agar dapat bekerja. Catu daya yang digunakan adalah sebesar 5 volt dengan mengadopsi tegangan kerja mikrokontroler Arduino. Sumber energi catu daya didapatkan dari PLN yang kemudian diubah menjadi tegangan DC dengan menggunakan adapter DC sebesar 9 volt dan rangkaian IC Regulator untuk menstabilkan tegangan menjadi 5 volt. Gambar 3.11 merupakan rangkaian catu daya pada Tugas Akhir ini.



Gambar 3.10 Rangkaian Catu Daya *Master*

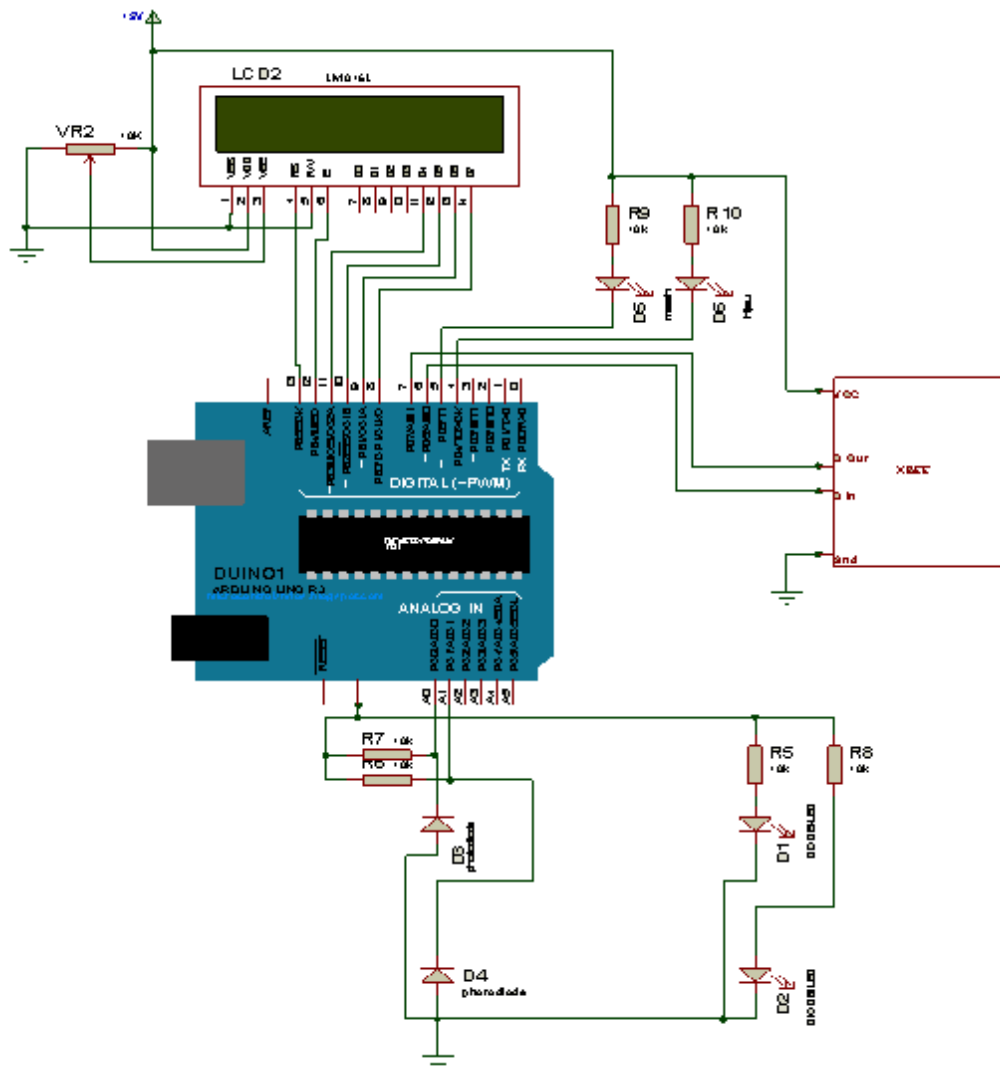
3.2.3. Rangkaian Sistem Keseluruhan

Setelah perancangan tiap rangkaian pada sistem selesai dibuat maka dapat dibuat rancangan sistem secara keseluruhan. Perancangan rangkaian sistem keseluruhan pada *Master* di tugas akhir mengenai sistem pengendali lampu lalu lintas ini dapat dilihat pada gambar 3.11.



Gambar 3.11 Skematik Keseluruhan Pada *Master*

Adapun mengenai skematik keseluruhan pada bagian *Slave* seperti pada gambar 3.12.

Gambar 3.12 Skematik Keseluruhan Pada *Slave*

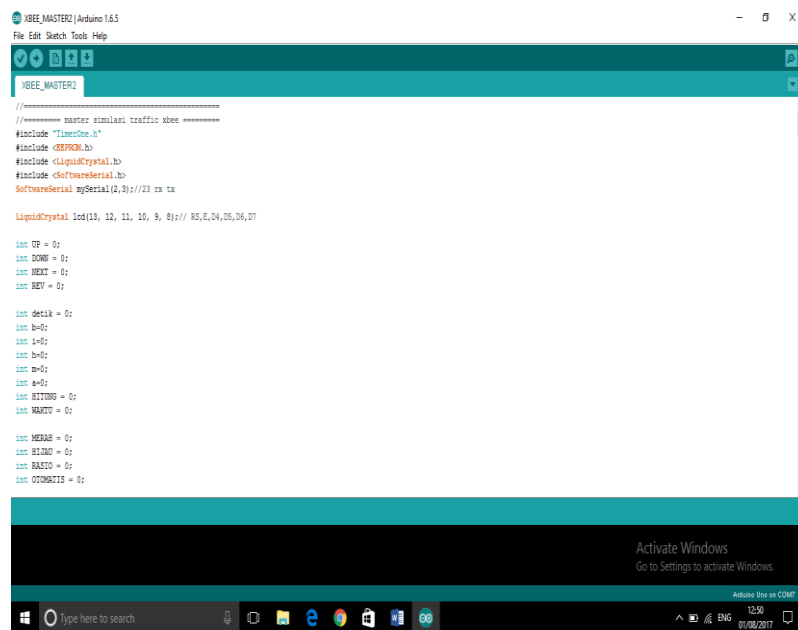
3.3 PERANCANGAN DAN PEMBUATAN PROGRAM

Pada rancang bangun pengendali sistem lampu lalu lintas tersebut dapat meliputi pembuatan *flowchart* program, perancangan program pada sisi *master* dan perancangan program pada sisi *slave*. Pada sisi *master*, perancangan program meliputi perancangan program mengirim data, perancangan program pemrosesan data, serta perancangan program penampil LCD. Sedangkan pada sisi *slave*, perancangan program meliputi

program menerima data, perancangan program LED. *Software* yang digunakan sebagai alat bantu dalam menulis program pada Tugas Akhir ini adalah menggunakan Arduino 1.0 .

3.3.1. *Software* Arduino 1.0 sebagai Program Arduino Uno

Arduino 1.0 adalah *software* yang membantu pembuatan program mikrokontroler dengan menggunakan bahasa pemrograman C. Adanya *software* arduino ini akan mempermudah kerja untuk mengetahui sistem bangun pengendali lampu lalu lintas ini. Gambar 3.13 merupakan tampilan dari Arduino 1.0.



```
YBEE_MASTER1
//===== MASTER simulasi traffic xbee =====
#include "TimerOne.h"
#include <EEPROM.h>
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2,3);//23 rx tx

LiquidCrystal lcd(13, 12, 11, 10, 9, 8);// RS,E,D4,D5,D6,D7

int UP = 0;
int DOWN = 0;
int NEXT = 0;
int REV = 0;

int dht11 = 0;
int b=0;
int l=0;
int b=0;
int m=0;
int a=0;
int HIJUNG = 0;
int WAKTU = 0;

int MERAH = 0;
int KIRI = 0;
int RASIO = 0;
int COGNETIS = 0;
```

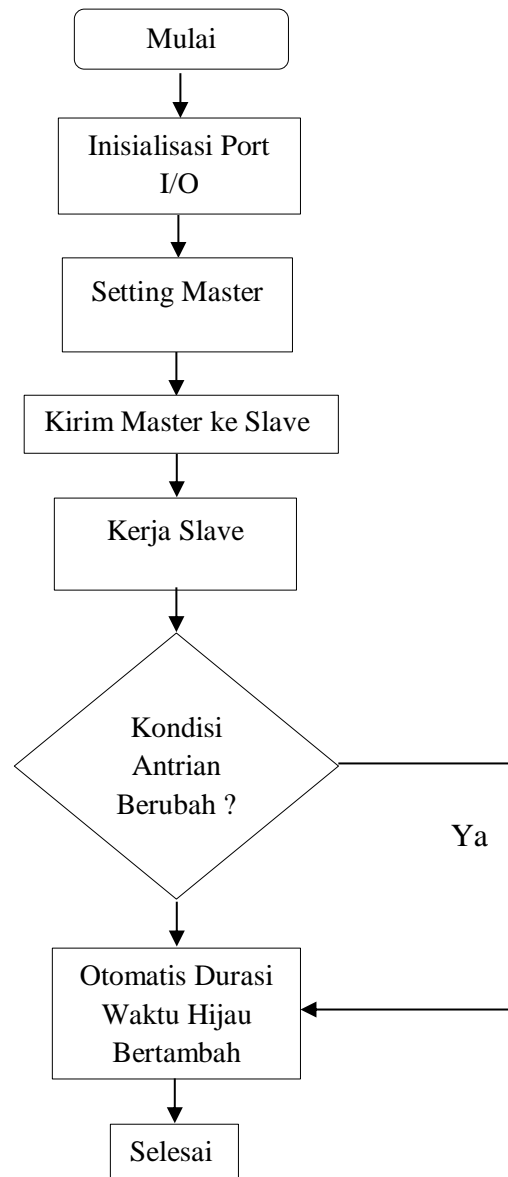
Gambar 3.13 Tampilan *Software* Arduino.

Pada *software* Arduino sangat *simple*, tidak ada pengaturan-pengaturan untuk mengganti mikrokontroler yang akan digunakan. Jadi langsung saja tuliskan program untuk menjalankannya.

3.3.2. Pembuatan dan Penjelasan Program

Penjelasan masing-masing program sistem pengendali lampu lalu lintas menggunakan komunikasi ZigBee berbasis Arduino akan dijelaskan berdasarkan sistem kerja. Sistem kerja ini akan menjelaskan semua jalur atau urutan kerja mulai dari awal sampai akhir.

Pada mikrokontroler *master* akan mengambil data pendeteksian objek dari sensor infra merah yang terpasang di ruas jalan. Sensor infra merah adalah sebagai eksekutor pengecekan antrian dengan menghitung jumlah kendaraan yang melintas. Setelah antrian sudah didapatkan selanjutnya akan tampil pada LCD, dan mikrokontroler *master* akan memproses data tersebut, apakah berubah atau tidak. Penentuan waktu lampu dilihat dari kriteria hasil *scanning* yang ada. Dengan *setting* lampu hijau menyala misalnya adalah 10 detik dan lampu merah menyala adalah 13 detik. Setelah proses pada *master* telah selesai maka akan dikirimkan ke *slave*. Program pada tugas akhir ini meliputi dua bagian, yaitu pada sisi *master* dan pada sisi *slave*. Pada sisi *master* meliputi program inisialisasi *port I/O*, serta program utama. Sedangkan pada sisi *slave* meliputi program inisialisasi *port I/O*, serta program utama. Di bawah ini merupakan flowchart 3.14 pembuatan program pada Tugas Akhir ini, sebagai berikut :

Flowchart 3.14 Pembuatan Program *Master* dan *Slave*

Dengan melihat flowchart 3.14 ini kemungkinan besar bisa mengetahui proses-proses atau instruksi kerja dari sebuah program yang nantinya akan dijalankan. Program pada tugas akhir ini meliputi dua bagian, yaitu pada sisi *master* dan pada sisi *slave*. Pada tugas akhir ini meliputi *slave*

dan *master*. Berikut pembahasan mengenai program dari kerja alat Tugas Akhir.

3.3.3. Program

Inisialisasi pada program ini dibagi menjadi 3 bagian. Inisialisasi bagian pertama yaitu inisialisasi untuk mengaktifkan *input* dan *output*. Bagian kedua inisialisasi yaitu inisialisasi pin yang digunakan. Inisialisasi bagian ke tiga yaitu inisialisasi untuk tampilan awal program. Dari ke tiga jenis inisialisasi yang ada, dijelaskan sebagai berikut :

a. Inisialisasi Pin Yang Digunakan

Inisialisasi pin digunakan untuk menetapkan bagian pin mana yang akan dipakai. Pin yang digunakan untuk RX dan TX pada master yaitu pin 2 dan 3. LCD 16x2 menggunakan pin 13,12, 11, 10, 9, 8. Dari masing-masing pin yang digunakan tersebut, perintah yang digunakan yaitu seperti berikut ini :

```
SoftwareSerial mySerial(2,3); //23 rx tx
LiquidCrystal lcd(13, 12, 11, 10, 9, 8); //
RS,E,D4,D5,D6,D7
```

b. Inisialisasi Input Dan Output

Inisialisasi *input* dan *output* berfungsi untuk mengaktifkan *input* dan *output* yang digunakan dalam pembuatan tugas akhir. *Input* dan *output* tersebut akan dikendalikan oleh mikrokontroler agar dapat bekerja. Berikut merupakan program inisialisasi I/O yang digunakan.

```
Serial.begin(9600);
mySerial.begin(9600);
lcd.begin(16, 2);
lcd.clear();
```

c. Inisialisasi Program Master

1). Pada inisialisasi program master ini, berfungsi untuk menyiapkan penempatan *port*, *memory* dan *library*. Pada program berikut terdapat untuk inisialisasi *port* LCD dan fungsi tombol. Di tombol sendiri memiliki berbagai banyak fungsi, seperti halnya *setting* lampu warna hijau, lampu warna merah, rasio kendaraan yang melintas per menit dan *setting* durasi waktu lampu hijau otomatis apabila rasio kendaraan melebihi objek per menitnya. Perintah yang digunakan untuk melakukan penjalanan program sebagai berikut :

```
#include "TimerOne.h"

#include <EEPROM.h>

#include <LiquidCrystal.h>

#include <SoftwareSerial.h>

SoftwareSerial mySerial(2,3);//23 rx tx

LiquidCrystal lcd(13, 12, 11, 10, 9, 8);//
RS,E,D4,D5,D6,D7

int UP = 0;

int DOWN = 0;

int NEXT = 0;

int REV = 0;

int detik = 0;

int b=0;

int i=0;

int h=0;

int m=0;

int a=0;
```

```
int HITUNG = 0;

int WAKTU = 0;

int MERAH = 0;

int HIJAU = 0;

int RASIO = 0;

int OTOMATIS = 0;

int MERAH_E = 10;

int HIJAU_E = 20;

int RASIO_E = 30;

int OTOMATIS_E = 40;

int MERAH_SEBELUMNYA = 11;

int HIJAU_SEBELUMNYA = 21;

char INCHAR;

char IN_KEND[4];

int KENDARAAN = 0;

int KENDARAAN1 = 0;

int KENDARAAN2 = 0;

int KENDARAAN3 = 0;

void setup ()

{

    Timer1.initialize(500000);          // initialize
timer1, and set a 1/2 second period

    Timer1.attachInterrupt(callback); // attaches
callback() as a timer overflow interrupt

    Serial.begin(9600);

    mySerial.begin(9600);
```

```
lcd.begin(16, 2);
```

2. Pada pembahasan berikut mengenai program tampilan awal pada sisi *master*. Pada sisi awal terdapat perintah untuk melakukan *setting* tulisan awalan pada LCD *master* beserta durasi waktu 2 detik yang digunakan pada saat tampilan awal muncul sampai berakhirnya tulisan tersebut. Pada awalan LCD akan terdapat tampilan tulisan dengan format program `lcd.print("MASTER CONTROLER");` dan berikut pada baris ke dua akan muncul tulisan dengan program `lcd.print("XBEE COMMUNICATION");` berikut programnya :

```
lcd.clear();  
lcd.setCursor(0, 0);  
lcd.print("MASTER CONTROLER");  
lcd.setCursor(0, 1);  
lcd.print("XBEE COMMUNICATION");  
delay (2000);  
lcd.clear();  
a=0;  
detik=0;  
}
```

3. Pada pembahasan berikut mengenai program *interrupt*. *Interrupt* pada program yaitu terdapat pada `void callback()` yang merupakan merupakan suatu penjeda atau menghentikan aktivitas program pada *master* maupun *slave* selama setengah detik sekali, kemudian akan kembali seperti semula yaitu program bekerja. Fungsi dari `i++;` yaitu untuk memiliki arti sebagai perhitungan dengan menekan satu kali tombol. Berikut programnya:

```
void callback()
{
    i++;
    if(i==2)
    {
        a++;
        i=0;
        detik++;
    }
}
```

4. Pada penjelasan ini yaitu mengenai program proses penyalaan durasi lampu merah. Pada proses tersebut tidak hanya *setting*-an lampu merah, ada juga mengenai program tampilan di LCD apabila lampu merah sedang mengalami proses perhitungan atau detikan lampu merah bekerja, berikut program yang digunakan pada penyalaan lampu merah :

```
void loop ()
{
    BACA_EEPROM();
    // delay (200);
    do
    {
        BACA_MENU();
        BACA_SERIAL();
        lcd.setCursor(0,0);
        lcd.print("COUNT : ");
        lcd.print(a);
    }
```



```
lcd.print(" ");  
  
if(MERAH >= a)  
{  
    lcd.setCursor(0,1);  
    lcd.print("M : ");  
    lcd.print(MERAH);  
    lcd.print(" ");  
}
```

5. Pada bagian berikut merupakan format data dan program bagaimana cara mengirim data atau perintah yang digunakan untuk mengirimkan lampu hijau dan lampu merah ke *slave* yaitu dengan format M[NilaiMerah]*H[NilaiHijau]#, yang mempunyai arti yaitu bahwa huruf “M” merupakan singkatan dari kata “Merah” dengan maksud apabila nilai merah dikirimkan ke *slave*, maka tidak akan tertukar dengan nilai warna hijau, dan juga sebaliknya yaitu tanda “*” berfungsi sebagai pemisah antara nilai hijau dan nilai merah.

```
    if(detik >= 60)          // kirim perubahan otomatis  
    setiap 1 menit  
    {  
        detik = 0;  
        mySerial.print("M");  
        mySerial.print(MERAH);  
        mySerial.print("*");  
        delay(500);  
        mySerial.print("H");  
        mySerial.print(HIJAU);  
        mySerial.print("#");  
        lcd.setCursor(6,1);  
        lcd.print("KIRIM DATA");
```

```
        delay(2000);
        lcd.clear();
        a=0;
    }
}while(a !=MERAH);
a=0;
```

6. Berikut merupakan penjelasan dan program mengenai penyalaan durasi lampu hijau. Bahwa tidak hanya program *setting*-an lampu hijau, ada juga mengenai letak kolom dan baris tampilan yang digunakan untuk menghitung durasi waktu pada LCD. Berikut programnya :

```
BACA_MENU();
BACA_SERIAL();
lcd.setCursor(0,0);
lcd.print("COUNT : ");
lcd.print(a);
lcd.print(" ");
lcd.print(" ");
if(HIJAU >= a)
{
    lcd.setCursor(0,1);
    lcd.print("H : ");
    lcd.print(HIJAU);
    lcd.print(" ");
}
```

7. Pada bagian berikut merupakan format data dan program bagaimana cara mengirim data atau perintah yang digunakan untuk mengirimkan lampu hijau dan lampu merah ke *slave* yaitu dengan format

M[NilaiMerah]*H[NilaiHijau]#, yang mempunyai arti yaitu bahwa huruf “M” merupakan singkatan dari kata “Merah” dengan maksud apabila nilai merah dikirimkan ke *slave*, maka tidak akan bertukar dengan nilai warna hijau, dan juga sebaliknya yaitu tanda “*” berfungsi sebagai pemisah antara nilai hijau dan nilai merah.

```
    if(detik >= 60)           // kirim perubahan otomatis
setiap 1 menit
    {
        detik = 0;
        mySerial.print("M");
        mySerial.print(MERAH);
        mySerial.print("*");
        delay(500);
        mySerial.print("H");
        mySerial.print(HIJAU);
        mySerial.print("#");
        lcd.setCursor(6,1);
        lcd.print("KIRIM DATA");
        delay(2000);
        lcd.clear();
        a=0;
    }
}while(a !=MERAH);
a=0;
```

8. Berikut penjelasan dalam program `void BACA_MENU()` yaitu proses pembacaan master terhadap tombol dengan melakukan setting di dalam menu seperti “NEXT” yaitu dengan program `if (NEXT <= 100)` dengan penjelasan bahwa apabila melakukan 1 kali pengeklikan pada tombol akan bernilai 100 dalam rutin program baca tombol. Yang dimaksud tersebut yaitu pada bagian tombol terdapat 4 tombol. Masing-masing tombol memiliki fungsi antara lain sebagai berikut seperti

penyettingan durasi lampu hijau, penyettingan lampu merah, atur rasio kendaraan per menit kemudian akan ada menu yang menyajikan penambahan otomatis pada lampu warn hijau kemudian otomatis. Pada penjelasan pada bagian tombol apabila penyettingan telah dilakukan, secara otomatis program tersebut akan bekerja secara terus menerus sebelum adanya pen-*setting*-an ulang. Berikut program yang rutin baca tombol, sebagai berikut :

```
void BACA_MENU()
{
    BACA_TOMBOL();
    if (NEXT <= 100)
    {
        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print(" MENU ^ rasio");
        lcd.setCursor(0,1);
        lcd.print("< penyalaan");
        MERAH = EEPROM.read(MERAH_E);    // BACA MEMORI
                                           LAMANYA MERAH
        HIJAU = EEPROM.read(HIJAU_E);
        delay(350);
        do
        {
            BACA_TOMBOL();
            if(UP <= 100)
            {
                lcd.clear();
```

```
lcd.setCursor(0,0);  
lcd.print("RASIO");  
lcd.setCursor(0,1);  
lcd.print("^ + v -");  
do  
{  
  BACA_TOMBOL();  
  lcd.setCursor(0,1);  
  if(UP <=100)           // tambah rasio  
{  
    RASIO ++;  
    lcd.setCursor(0,0);  
    lcd.print("RASIO = ");  
    lcd.print(RASIO);  
    lcd.print(" /min");  
    lcd.print("  ");  
    lcd.setCursor(0,1);  
    lcd.setCursor(0,1);  
    lcd.print("< SAVE ^ v ATUR");  
    lcd.print("  ");  
    delay(250);  
  }  
  BACA_TOMBOL();  
  if(UP <=100)           // tambah NYALA  
                          HIJAU OTOMATIS  
{
```

```
        OTOMATIS ++;

        lcd.setCursor(0,1);

        lcd.print("H+ = ");

        lcd.print(OTOMATIS);

        lcd.print("  ");

        lcd.print(" > save");

        lcd.print("  ");

        delay(250);

    }

}while(NEXT !=0);    // EXIT

    b = 0;

    lcd.clear();

    lcd.setCursor(0,0);

    lcd.print("---- EXIT -----");

    delay(500);

{

    MERAH = EEPROM.read(MERAH_E);        // BACA MEMORI
LAMANYA MERAH

    HIJAU = EEPROM.read(HIJAU_E);        // BACA MEMORI
LAMANYA HIJAU

    RASIO = EEPROM.read(RASIO_E);        // BACA MEMORI
RASIO

    OTOMATIS = EEPROM.read(OTOMATIS_E); // BACA MEMORI
OTOMATIS

}
```

d. Inisialisasi Program *Slave*

1. Pada inisialisasi program *master* ini, berfungsi untuk menyiapkan penempatan *port*, *memory* dan *library*. Pada program `#include "TimerOne.h"` di mana bahwa pada `#include` merupakan suatu library dan pada `"TimerOne.h"` merupakan pemilihan pada timer 1. Berikut terdapat untuk inisialisasi *port* LCD dan fungsi tombol. Pada penjelasan memiliki berbagai banyak fungsi, seperti halnya lampu hijau, lampu merah, sensor dan objek kendaraan yang melintas sensor. Dalam menyiapkan mengenai lampu hijau, lampu merah, sensor dan objek kendaraan yang melintas sensor perlu dipersiapkan tipe data dan variabel, dan berikut programnya :

```
#include "TimerOne.h"
#include <EEPROM.h>
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2,3); //23 rx tx
LiquidCrystal lcd(13, 12, 11, 10, 9, 8); //
                    RS,E,D4,D5,D6,D7

int SENSOR1 = A0;
int SENSOR2 = A1;
int LAMP_MERAH = 7;
int LAMP_HIJAU = 6;
int VALUE1 = 0;
int VALUE2 = 0;
int JML_KEND = 0;
int JML_KEND_E = 10;
int out =0;
char INCHAR;
char IN_MERAH[4];
char IN_HIJAU[4];
```

```
int KON_MERAH = 0;
int KON_MERAH1 = 0;
int KON_MERAH2 = 0;
int KON_MERAH3 = 0;
int KON_HIJAU = 0;
int KON_HIJAU1 = 0;
int KON_HIJAU2 = 0;
int KON_HIJAU3 = 0;
int M = 0;
int H = 0;
int b = 0;
int i=0;
int a=0;
int detik = 0;
int HITUNG = 0;
int WAKTU = 0;
int MERAH = 0;
int HIJAU = 0;
int RASIO = 0;
int MERAH_E = 12;
int HIJAU_E = 20;
int RASIO_E = 30;
```

2. Pada bagian ini terdapat penjelasan dan program yang berhubungan mengenai nyala lampu hijau dan nyala lampu merah, dan pada apabila tegangan listrik mendapatkan sumber dari PLN akan secara otomatis lampu merah dan hijau menyala dengan program `digitalWrite(LAMP_MERAH, HIGH);` dan `digitalWrite(LAMP_HIJAU, HIGH);` kemudian pada program `pinMode(LAMP_MERAH, OUTPUT);` merupakan lampu merah berfungsi sebagai output dan `pinMode(LAMP_HIJAU, OUTPUT);` lampu hijau sebagai output yang

sama. Penampilan karakter atau tulisan pada LCD yang digunakan, berikut programnya :

```
void setup ()
{
pinMode(LAMP_MERAH, OUTPUT);
pinMode(LAMP_HIJAU, OUTPUT);
digitalWrite(LAMP_MERAH, HIGH);
digitalWrite(LAMP_HIJAU, HIGH);
Serial.begin(9600);
mySerial.begin(9600);
lcd.begin(16, 2);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("SLAFE CONTROLER");
lcd.setCursor(0, 1);
lcd.print("XBEE COMMUNICATION");
delay (2000);
lcd.clear();
}
```

3. Berikut penjelasan mengenai penyalan kedua buah lampu tersebut yaitu hijau dan merah dengan maksud yaitu apabila lampu hijau menyala maka lampu merah pun akan mati, dengan program `digitalWrite(LAMP_MERAH, LOW);` dan `digitalWrite(LAMP_HIJAU, HIGH);` atau sering disebut dengan pergantian penyalan kedua warna lampu tersebut. Kemudian program dalam melakukan pen-*setting*-an tulisan pada LCD *slave* dan cara kerja sensor apabila tidak terhalang dan apabila sensor tersebut terhalang. Kemudian setelah sensor membaca objek, sensor akan membaca dan menampilkan jumlah kendaraan yang melintas. Berikut programnya :

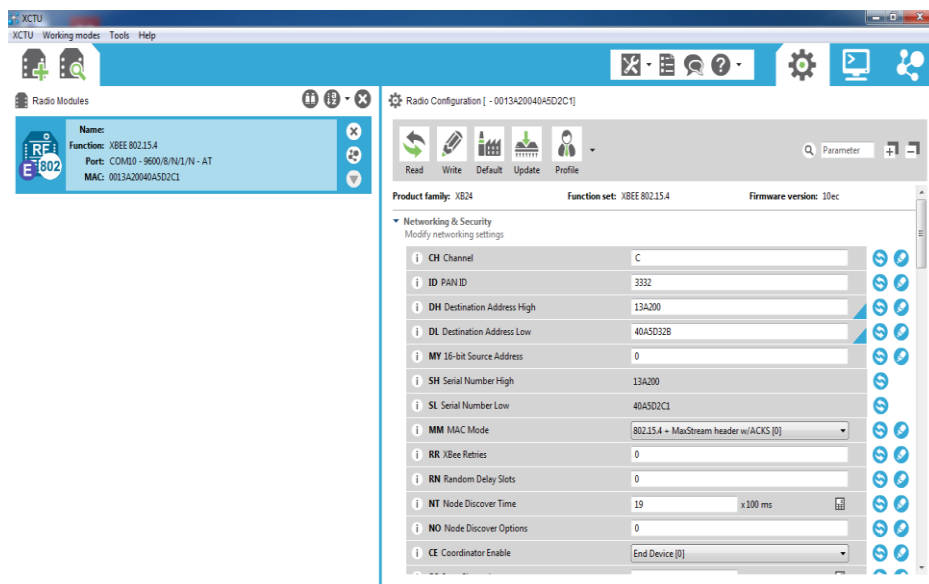
```
digitalWrite(LAMP_MERAH, LOW);
digitalWrite(LAMP_HIJAU, HIGH);
lcd.setCursor(0,0);
lcd.print("CNT:");
lcd.print(a);
lcd.print(" ");
lcd.setCursor(9,0);
lcd.print("Knd=");
lcd.print(JML_KEND);
lcd.print(" ");
if(MERAH >= a)
{
    lcd.setCursor(0,1);
    lcd.print("M : ");
    lcd.print(MERAH);
    lcd.print(" ");
}

if (VALUE1 >= 600 && VALUE2 >= 600)
{
    do
    {
        DETEC:
        BACA_SENSOR();
        if (VALUE1 <= 200 || VALUE2 <= 200) //
        {
            JML_KEND ++; // tambahkan jumlah kendaraan
            //lcd.clear();
            lcd.setCursor(9,0);
            lcd.print("Knd=");
            lcd.print(JML_KEND);
            lcd.print(" ");
        }
    }
}
```

```
        out = 1;
    }
    if (VALUE1 >= 600 && VALUE2 >= 600)
    {
        lcd.setCursor(0,0);
        lcd.print("CNT:");
        lcd.print(a);
        lcd.print(" ");
        goto DETEC;
    }
    KIRIM_DATA();
}while(out != 1); // tunggu sampai mobil lewat
out = 0;
}
lcd.setCursor(9,0);
lcd.print("Knd=");
lcd.print(JML_KEND);
lcd.print(" ");
}while(a != MERAH);
a=0;
BACA_EEPROM();
// delay(700);
```

3.3.4 Konfigurasi XBee

Pengaturan komunikasi antar zigbee menggunakan sistem *unicast* atau lebih dikenal *peer to peer* dengan men-*setting destination* ke alamat tujuan atau ke Zigbee yang lain. Cara ini hanya dapat digunakan untuk komunikasi antar alamat tujuan dan pengirim. *Software* yang digunakan untuk mengatur zigbee adalah Digi XCTU. Berikut gambar 3.15 merupakan *setting* Zigbee Tx dan Rx.



Gambar 3.15 Konfigurasi Zigbee Penerima

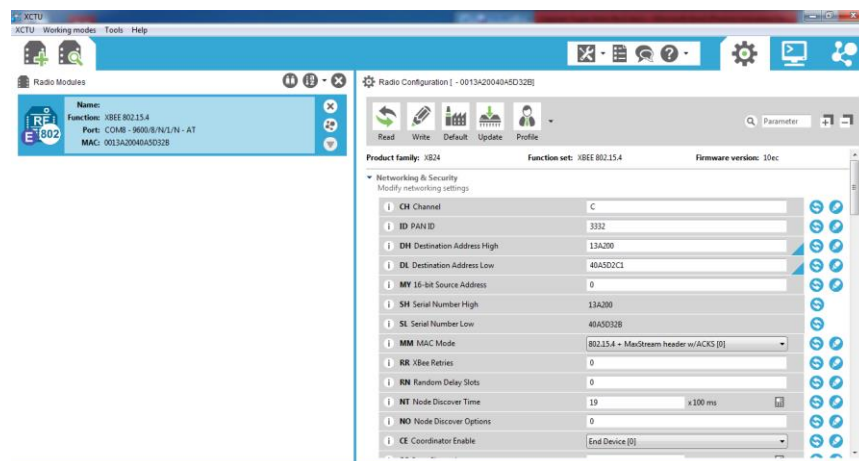
Dan berikut tabel 3.10 merupakan *Networking* dan *Security* pada konfigurasi ZigBee penerima.

Tabel 3.10 *Networking* dan *Security* Penerima.

No	<i>Modify Networking Setting</i>	<i>Setting Modify Networking Setting</i>
1	<i>CH Channel</i>	C
2	ID PAN ID	3332
3	<i>Destination Address High</i>	13A200
4	<i>Destination Address Low</i>	40A5D32B
5	<i>16-bit Source Address</i>	0

Dalam konfigurasi tersebut, Channel (CH) yang digunakan disamakan dengan Zigbee Penerima, demikian pula dengan Pan ID (ID). Konfigurasi Destination High (DH) dan Konfigurasi Destination Low (DL) menggunakan MAC ID yang berada di Zigbee Penerima. Sedangkan untuk konfigurasi di Zigbee Rx adalah kebalikan dari Zigbee Pengirim. Yaitu dengan mengkonfigurasi Channel (CH) yang sama kemudian PAN ID, konfigurasi DH dan DL merupakan kebalikan dari konfigurasi dari Zigbee Tx yaitu dengan DH dari MAC Address Zigbee pengirim dan DL dari MAC Address Zigbee pengirim.

Berikut contoh gambar 3.16 konfigurasi Zigbee penerima :



Gambar 3.16 Konfigurasi Zigbee Pengirim

Dan berikut tabel 3.11 merupakan *Networking* dan *Security* pada konfigurasi ZigBee pengirim.

Tabel 3.11 *Networking* dan *Security* Pengirim.

No	<i>Modify Networking Setting</i>	<i>Setting Modify Networking Setting</i>
1	CH Channel	C
2	ID PAN ID	3332
3	<i>Destination Address High</i>	13A200
4	<i>Destination Address Low</i>	40A5D2C1
5	16-bit <i>Source Address</i>	0

3.3.5 Pendefinisian Penggunaan XBee

Zigbee sendiri merupakan sebuah protokol komunikasi tingkat tinggi dengan menggunakan standar IEEE 802.15.4 dengan *wireless personal area networks* (WPANs). Teknologi ini dimanfaatkan untuk penggunaan pengiriman data yang dilakukan secara wireless dengan tingkat transmisi data rendah dan konsumsi daya yang rendah juga. Dan dari sisi harga relatif lebih murah dibandingkan dengan WPANs yang lain seperti *Bluetooth*. Teknologi Zigbee sendiri sering diaplikasikan pada sistem *embedded* (sistem tertanam). Zigbee memiliki kecepatan transfer 250Kbps, jauh lebih rendah dari protokol lain seperti *Bluetooth* yang mencapai 1 Mbps akan tetapi dari sisi jarak Zigbee lebih unggul menyentuh angka 76 m. Dan alat ini menggunakan konsumsi daya baterai yang dayanya mencapai setengah tahun.

Tabel 3.12 perbedaan Xbee S1 dan Xbee Pro

Spesifikasi	Xbee S1	Xbee Pro
Daya	1 -2mW	50 -63mW
Jangkauan indoor	40 m	60 -90m
Jangkauan Loss	120 m	1,5 – 3,2 Km
Sensivitas Penerima	-96 dBm	-102 dBm
Tx Peak Current	40 mA @2,2 V	170-295dBm
Rx Current	40 mA @2,2 V	45 mA @3,3 V
Power Down Current	< 1uA	3,5 uA

Protokol Zigbee hanya memiliki tiga lapisan yang terdiri dari lapisan *networking*, *application* dan *security*. Akan tetapi memiliki lapisan *physical* paling bawah dan MAC yang mengadopsi pada IEEE 802.15.4 yang digunakan untuk mendefinisikan pengaturan daya, pengalamatan, kesalahan, format pesan dan *point to point* komunikasi radio. Zigbee menggunakan 3 frekuensi yang digunakan secara berbeda – beda. Untuk saat ini frekuensi yang digunakan 915 MHz untuk Amerika, 868 MHz di Eropa dan 2,4 GHz di Jepang dan lainnya.

3.3.6 Program Pengiriman Data

Pada program pengiriman data yaitu bagaimana program yang terdapat pada *master* kemudian diterima oleh *slave* tergantung apa yang telah di-*setting* oleh *master* kemudian bagaimana hasil yang diterima oleh *slave*. Berikut program dan penjelasan mengenai pengiriman data dari *master* dan *slave* dengan cara mengenai *setting* penyalan lampu merah, sebagai berikut :

3.3.6.1 Program Master

Di bawah ini merupakan program penyalaaan lampu merah, dan berikut programnya. Pada `int MERAH = 0;` merupakan *integer* dari lampu warna merah, dan `int MERAH = 0;` merupakan suatu inisialisasi awalan pada lampu warna merah kemudian terdapat `int MERAH_E = 10;` yaitu *integer* warna merah dengan nilai variabel 10, kemudian dilanjutkan dengan `int MERAH_SEBELUMNYA = 11;` yaitu bahwa *integer* lampu merah yang sebelumnya memiliki nilai variabel 11 dengan `Serial.begin(9600);` yaitu mengatur *boudrate serial* yang diisi dengan *boudrate* 9600 kemudian akan muncul pada lampu merah pada LCD dengan program :

```
if(MERAH >= a)
{
    lcd.setCursor(0,1);
    lcd.print("M : ");
    lcd.print(MERAH);
    lcd.print(" ");
}
if(detik >= 60) // kirim perubahan otomatis
setiap 1 menit
{
    detik = 0;
    mySerial.print("M");
    mySerial.print(MERAH);
    mySerial.print("*");
    delay(500);
}
```

Dan apabila lampu merah melakukan perhitungan selama lebih dari 1 menit yaitu terdiri dari 60 detik, maka akan secara otomatis lampu merah pun melakukan proses pergantian dengan lampu lainnya. Dan apabila lampu merah melakukan proses kerja selama 1 menit maka secara otomatis lampu merah pun akan mengirimkan data ke *slave* dengan program :


```
lcd.setCursor(6,1);
  lcd.print("KIRIM DATA");
  delay(2000);
  lcd.clear();
  a=0;
}
}while(a !=MERAH);
a=0;
```

3.3.6.2 Program Slave

Pada pembahasan program *slave* yang perlu diketahui bahwa data yang diterima oleh *slave* berupa *setting*-an dari *master*, dan pada pembahasan berikut merupakan hanya pada lampu warna merah. Lampu merah sebagai *output* beserta keadaan menyala dalam proses penerimaan dengan program `pinMode(LAMP_MERAH, OUTPUT);` dan `digitalWrite(LAMP_MERAH, HIGH);` dengan menggunakan *boudrate serial* 9600 `Serial.begin(9600); mySerial.begin(9600);`, kemudian hasil yang diterima oleh *slave* pada lampu merah sebagai berikut :

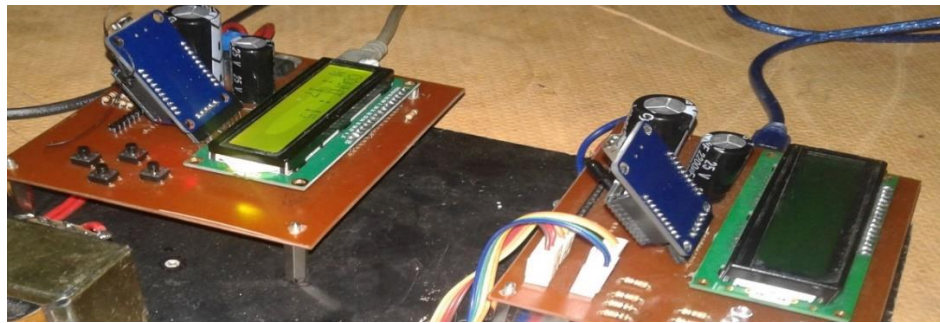
```
do
{
  BACA_SERIAL();
  KIRIM_DATA();
  BACA_SENSOR();

  digitalWrite(LAMP_MERAH, LOW);
do
{
  BACA_SERIAL();
  KIRIM_DATA();
  BACA_SENSOR();
```

```
digitalWrite(LAMP_MERAH, LOW);
```

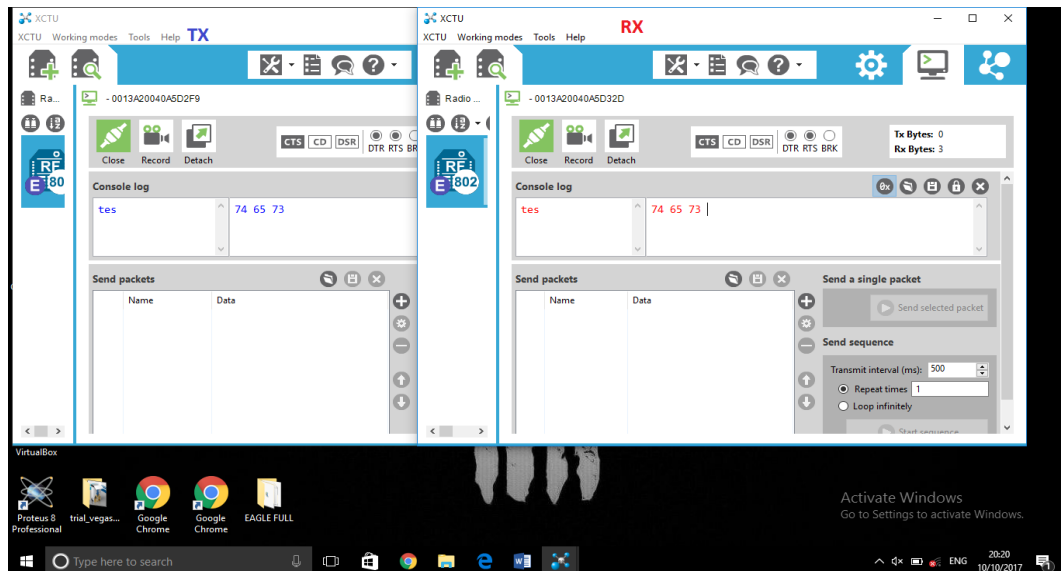
3.3.7 Data Pengiriman XBee *Master* ke *Slave*

Berikut gambar keterangan pada alat tugas akhir yang berjudul “Rancang Bangun Pengendali Lampu Lalu Lintas Menggunakan Komunikasi ZigBee Berbasis Arduino”, sebagai berikut gambar 3.17 merupakan data pengiriman XBee *Master* dan *Slave*.



Gambar 3.17 Data Pengiriman XBee *Master* dan XBee *Slave*

Di bawah ini pada gambar 3.18 merupakan gambar proses pengiriman pada komunikasi ZigBee antara pengirim dan penerima, sebagai berikut:



Gambar 3.18 Proses Komunikasi ZigBee dari pengirim (TX) ke penerima (RX)