**BAB IV**

# HASIL DAN PEMBAHASAN

## Unit RTC ( Real Time Clock )

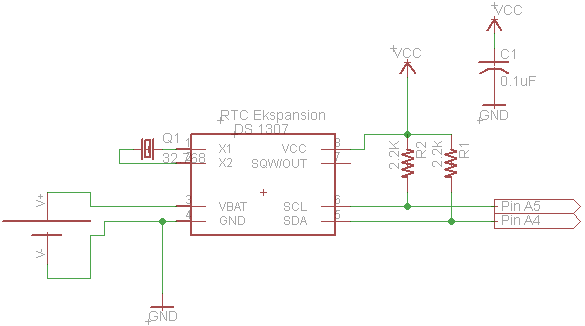
Unit RTC digunakan untuk mengirimkan data waktu pembacaan sensor. Hal ini diperlukan untuk mengetahui waktu pembacaan data sensor, dan koreksi terhadap adanya paket data yang hilang.

### Skematik Rangkaian RTC

DS1307 merupakan IC RTC yang menggunakan protokol I2C (*Inter IC Bus* ) yang sesuai dengan protokol TWI (*2 wire serial interfaces*) pada mikrokontroler AVR. Dalam komunikasi menggunakan protokol I2C hanya diperlukan 2 jalur data yakni SCL (*Serial Clock*) dan SDA (*Serial Data*).

*Arduino board* yang digunakan adalah deumilanove dengan mikrokontrolernya adalah Atmega328 yang mana pin SCL terletak pada kaki no 28 pada Atmega328 dan dalam *arduino board* disebut sebagai pin A5, sedangkan pin SDA terletak pada kaki no 27 pada Atmega328 dan dalam *arduino board* disebut sebagai pin A4.

Skematik rangkaian percobaan untuk modul RTC ditunjukan oleh gambar 4.1.

**

*Gambar 4.1 Skematik rangkaian modul RTC dengan IC DS1307*

### Algoritma Penggunaan RTC ( *Real Time Clock*) dengan IC DS1307

Pada penggunaan IC DS1307 dilakukan 2 proses yakni menulis data ke IC DS1307 ( dalam hal ini adalah seting waktu ) dan yang kedua adalah membaca data waktu dari IC DS1307. Prosedur kedua proses ini mengacu pada protokol I2C agar komunikasi berjalan sebagai mana mestinya.

Sesuai dengan protokol I2C, prosedur penulisan data dilakukan dengan langkah – langkah sebagai berikut.

1. *Master* mengirim *start* *sequence*
2. *Master* mengirim alamat *slave* yang akan ditulis data dengan bit R/W low (alamat genap)
3. *Master* mengirim alamat dari internal register yang ingin ditulis
4. *Master* mengirim byte data
5. Jika data belum tertulis semua, setelah *slave* mengirim ACK=0 maka byte data berikutnya dikirim
6. *Master* mengirim *stop* *sequence*.

Sedangkan prosedur pembacaan data dilakukan dengan langkah – langkah sebagai berikut.

1. *Master* mengirim *start* *sequence*
2. *Master* mengirim alamat *slave* yang akan ditulis data dengan bit R/W low (alamat genap)
3. *Master* mengirim alamat dari internal register yang ingin dibaca
4. *Master* mengirim *start* *sequence*
5. *Master* mengirim alamat *slave* yang akan ditulis data dengan bit R/W high (alamat ganjil)
6. *Master* membaca byte data dari *slave*
7. *Master* mengirim *stop* *sequence*.

Dengan konfigurasi *Arduino board* sebagai *master* dan modul RTC sebagai *slave* maka prosedur penulisan data dan pembacaan data sesuai protokol I2C diatas dapat disusun dengan notasi algoritmik berikut.

*{-procedure menulis data ke DS 1307 untuk set waktu-}*

Procedure setDS1307 (input= second,minute,hour,dayOfWeek,dayOfMonth,Year)

Set second as BCD *{--ubah ke BCD format--}*

Set minute as BCD

Set hour as BCD

Set dayOfWeek as BCD

Set dayOfMonth as BCD

Set Year as BCD

Set I2C address 0x68 *{--alamat DS1307--}*

Send (0) *{--Start Sequence--}*

Send (0x68) *{-kirim alamat slave-}*

Write (0x00=second) *{--tulis second @00h--}*

Write (0x01=minute) *{--tulis minute @01h--}*

Write (0x02=hour) *{--tulis hour @ 02h--}*

Write (0x03=dayOfWeek) *{tulis dayOfWeek @03h}*

Write (0x04=dayOfMonth*){--tulis dayOfMonth @ 04h--}*

Write (0x05=Year) *{--tulis Year @ 05h--}*

Send (0) *{--Stop sequence--}*

{--end procedure--}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*{--procedure membaca data dari DS 1307 --}*

Procedure getDS1307(output= second,minute,hour,dayOfWeek,dayOfMonth,Year)

Send (0)  *{--Start Sequence--}*

Send (0x68) *{--kirim alamat slave--}*

Send (0)  *{--Start Sequence--}*

Send (0x68 + 0x01) *{--Read data--}*

Second=read(0x00) *{--baca alamat 00h--}*

Minute=read(0x01) *{--baca alamat 01h--}*

Hour=read(0x02) *{--baca alamat 02h--}*

dayOfWeek=read(0x03) *{--baca alamat 03h--}*

dayOfMonth=read(0x04) *{--baca alamat 04h--}*

year=read(0x05) *{--baca alamat 05h--}*

Second=set second as Dec *{-------convert BCD ke Desimal-------}*

Minute=set minute as Dec

Hour=set hour as Dec

dayOfWeek=set dayOfWeek as Dec

dayOfMonth=set dayOfMonth as Dec

year=set year as Dec

send(0) *{--stop sequence--}*

*{--end procedure--}*

Didalam software arduino IDE versi 0022 telah tersedia *library* “wire.h” yang berfungsi mempermudah pengguna yang akan melakukan komunikasi *master* – *slave* menggunakan sistem I2C. *Library* ini mengacau pada aturan komunikasi I2C dengan kata kunci untuk memanggil prosedure dan funsi dari *library* “wire.h” sebagai berikut.

Wire.begin();

Wire.beginTransmission(alamat yang dituju);

Wire.endTransmission();

Wire.requestFrom(alamat + bit read);

Wire.send(data bit dalam BCD);

Wire.receive();

Wire.onReceive();

Wire.onRequest() ;

Untuk masing – masing prosedure penulisan dan pembacaan data ditunjukan oleh source code dibawah dengan terlebih dahulu memanggil *library* “wire.h” dan mendeklarasikan alamat DS1307 pada 0x68 di bagian inisialisasi.

#include "Wire.h"

#define DS1307\_I2C\_ADDRESS 0x68

*//\*\*\*\*\*menulis data set waktu ke DS1307\*\*\*\*\*//*

void setDateDs1307(byte second,byte minute,byte hour,byte dayOfWeek,byte dayOfMonth,byte month, byte year){

Wire.beginTransmission(DS1307\_I2C\_ADDRESS);

Wire.send(0); *// start clock dengan mengisi bit 7 = 0*

Wire.send(decToBcd(second));

Wire.send(decToBcd(minute));

Wire.send(decToBcd(hour));

Wire.send(decToBcd(dayOfWeek));

Wire.send(decToBcd(dayOfMonth));

Wire.send(decToBcd(month));

Wire.send(decToBcd(year));

Wire.endTransmission(); }

*//\*\*\*\*\*mengambil data dari DS1307\*\*\*\*\*//*

void getDateDs1307(byte \*second,byte \*minute,byte \*hour,byte \*dayOfWeek,byte \*dayOfMonth,byte \*month,byte \*year){

Wire.beginTransmission(DS1307\_I2C\_ADDRESS);

Wire.send(0);

Wire.endTransmission();

Wire.requestFrom(DS1307\_I2C\_ADDRESS, 7);

\*second = bcdToDec(Wire.receive() & 0x7f);

\*minute = bcdToDec(Wire.receive());

\*hour = bcdToDec(Wire.receive() & 0x3f);

\*dayOfWeek = bcdToDec(Wire.receive());

\*dayOfMonth = bcdToDec(Wire.receive());

\*month = bcdToDec(Wire.receive());

\*year = bcdToDec(Wire.receive());

}

Penulisan data ke DS1307 dilakukan dengan memanggil prosedur setDateDS1307(second, minute,hour, dayOfWeek, dayOfMonth, month, year) dan ini dilakukan pada saat pertama menggunakan IC DS1307, setelah itu data tersimpan dan perhitungan waktu akan terus berjalan.

Selama suplai tegangan pada DS1307 aktif dan tidak ada kerusakan pada rangkaian, maka data waktu pada DS1307 dapat diambil kapan saja. Apabila suplai tegangan pada DS1307 hilang, maka perlu dilakukan penulisan ulang waktu dan tanggal karena data yang tersimpan diregister internal DS1307 merupakan data terakhir IC ini bekerja secara normal (mendapat suplai tegangan).

Contoh pemanggilan prosedur untuk menulis data ke DS1307 adalah sebagai berikut.

void **loop**(){

second = 30;

minute = 7;

hour = 3;

dayOfWeek = 4;

dayOfMonth = 18;

month = 8;

year = 11;

setDateDs1307(second, minute, hour, dayOfWeek, dayOfMonth, month, year);

}

Setelah, data tertulis ke DS1307, langkah selanjutnya adalah pengambilan data kalender pada DS1307. Contoh pemanggilan prosedur untuk membaca data pada DS1307 adalah sebagai berikut.

void **loop**() {

byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;

getDateDs1307(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month, &year);

if(hour>0){

if(hour<=12){

Serial.print(hour, DEC);}

else {

Serial.print(hour-12, DEC);} }

else{

Serial.print("12");}

Serial.print(":");

if (minute < 10) {

Serial.print("0");}

Serial.print(minute, DEC);

Serial.print(":");

if (second < 10) {

Serial.print("0");}

Serial.print(second, DEC);

if(hour<12){

Serial.print("am");}

else{

Serial.print("pm");}

Serial.print(dayOfMonth,DEC);

delay(1000); }

Serial.print("-");

Serial.print(month,DEC);

Serial.print("-");

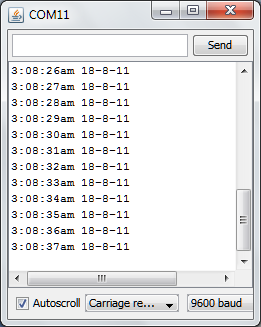
Serial.print(year,DEC);

Serial.println();

delay(1000);

}

Hasil eksekusi program pembacaan data DS1307 diatas diamati menggunakan serial monitor dengan seting *baud rate* 9600 bps. Hasil pengujian ditunjukan oleh gambar 4.2.

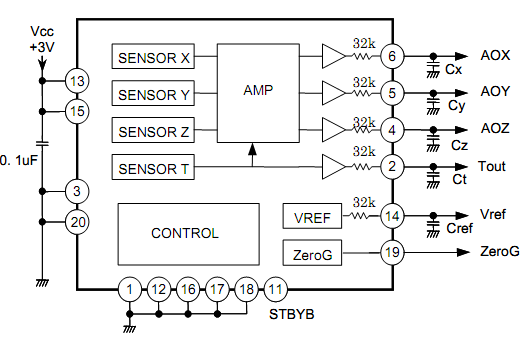
**

*Gambar 4.2 Pembacaan data DS1307*

## Metode Akusisi Data Percepatan

Dalam penelitian ini, penbacaan percepatan pergerakan tanah menggunakan modul *acelerometer* buatan Parallax.Inc dimana dalam modul ini sensor yang digunakan adalah H48C dan telah terintegrasi dengan *4 chanel 12-bit A/D converter with serial interface* MCP3204.

Blok diagram dari H48C ditunjukan oleh gambar 4.3. Dengan AOX sebagai analog output sumbu x, AOY sebagai analog output sumbu y, AOZ sebagai analog output sumbu y, dan Vref sebagai tegangan referensi. Maka, untuk mengetahui besarnya percepatan yang dinyatakan dalam level tegangan analog untuk tiap sumbu adalah sebagai berikut.

**

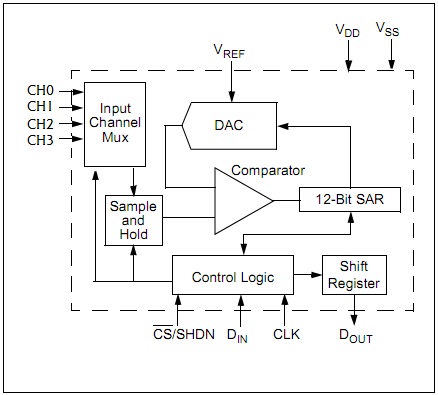
*Gambar 4.3 Blok diagram H48C*

gX = AOX – Vref (mV) ................................................................(4.1)

gY = AOY – Vref (mV) ................................................................(4.2)

gZ = AOZ – Vref (mV) .................................................................(4.3)

Pada modul 3 *axis accelerometer* buatan Parallax.Inc output tegengan analog tiap sumbu dan output Vref dihubungkan dengan MCP3204 untuk mengkonversi data analog menjadi digital dan mengirimkanya ke kontroler secara serial.

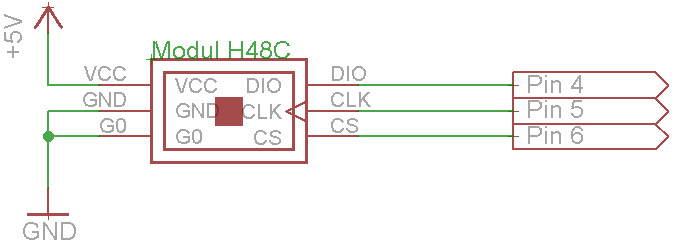


*Gambar 4.4 Diagram Blok MCP3204*

Diagram blok MCP3204 ditunjukan oleh gambar 4.4. Pada modul accelerometer, pin AOX pada H48C dihubungkan ke CH0 dari MCP3204, AOY pada H48C terhubung dengan CH1 dari MCP3204, AOZ pada H48C terhubung dengan CH2 dari MCP3204 dan Vref pada H48C terhubung dengan CH3 dari MCP3204.

### Skematik Rangkaian *Accelerometer*

Rangkaian pengujian menggunakan modul *3 axis accelerometer H48C* dan kontrolernya menggunaka *Arduino board* tipe Deumilanove. Skematik rangkaian pengujian ditunjukan oleh gambar 4.5.

**

*Gambar 4.5 Skematik rangkaian pengujian modul accelerometer*

Dari modul accelerometer hanya dibutuhkan 3 pin untuk melakukan pembacaan, yakni :

1. Pin DIO sebagai jalur digital input / output, terhubung dengan pin 4 *Arduino board*.
2. Pin CLK untuk pemberian *clock triger* sensor dan terhubung dengan Pin 5 *Arduino board*.
3. Pin CS (*Chip Select*) untuk aktifasi modul, terhubung dengan Pin 6 *Arduino board.*

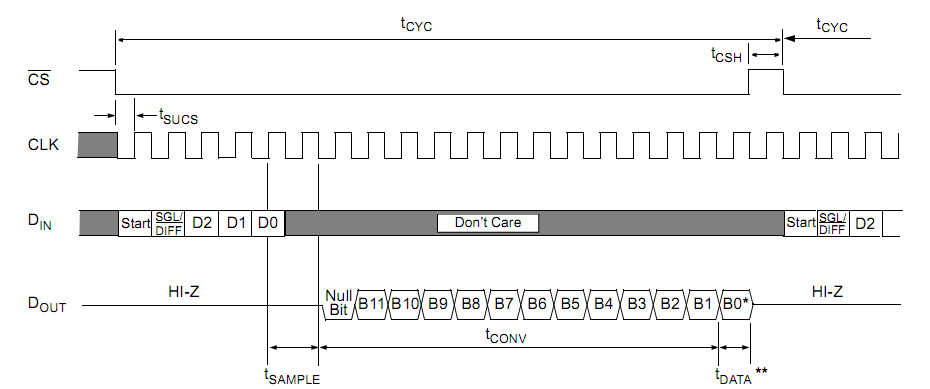
### Algoritma Pembacaan Percepatan Tiap Sumbu

Pembuatan algoritma pembacaan tiap sumbu didasarkan pada timing diagram dari MCP3204 (gambar 4.6). Frekuensi *clock* optimal dari MCP3204 adalah sebesar 10 KHz sehingga periode 1 *clock* yang paling optimal dapat dihitung dengan persamaan dibawah(persamaan 4.4).

...........................................................................(4.4)

Sehingga periode *clock* ) optimal adalah 0,0001 s atau sebesar 100 µs dan dalam satu pulsa *clock* terdapat 1 periode pulsa bernilai rendah (*)* dan 1 periode pulsa bernilai tinggi () sehingga masing – masing dan memiliki selang waktu yakni sebesar 50 µs.

Dalam modul *accelerometer* yang digunakan, jalur DIN dan DOUT digabungkan menjadi 1 dan disebut pin DIO.



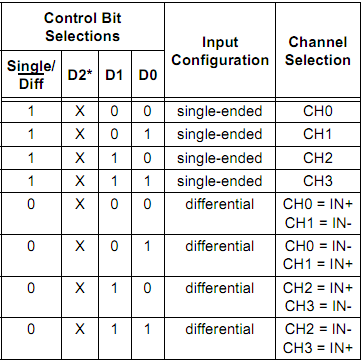
*Gambar 4.6 Timing Diagram MCP3204*

Karena pin DIN dan DOUT digabungkan menjadi 1 pin, maka untuk memungkinkan pembacaan data sensor diperlukan kontroler untuk mengatur kondisi pin DIO sebagai output pada saat pengiriman *control bit* dan sebagai input pada saat pembacaan data hasil konversi MCP3204.

Konfigurasi kontrol bit untuk pembacaan tiap *chanel* sesuai dengan tabel 4.1.

Mengingat bahwa CH0 = AOX, CH1 = AOY, CH2 = AOZ, CH3 = Vref maka alternatif konfigurasi input yang paling mudah adalah dengan input *single – ended* yakni dengan men-set bit Single/Diff bernilai 1.

Tabel 4.1. Konfigurasi Bit Kontrol MCP3204



Notasi algoritmik untuk pembacaan modul *accelerometer* adalah sebagai berikut .

**Procedure** getH48C ( input=D1,D0)

Set DIO sebagai Output

Set CLK sebagai output

Set /CS sebagai output

{--Kirim start bit--}

CLK ke 1 (*Falling edge),*CS=0,DIO=1

{--Kirim kontrol bit--}

CLK ke 2, DIO=1 *{--single ended--}*

CLK ke 3, DIO=0 *{--D2 (don’t care)--}*

CLK ke 4, DIO=D1

CLK ke 5, DIO=D0

*{--D1.D0 = 00 🡪 sb x*

*D1.D0 = 01 🡪 sb y*

*D1.D0 = 10 🡪 sb z*

*D1.D0 = 11 🡪 sb Vref--}*

CLK ke 6 *{--sample time--}*

CLK ke 7 *{--nul bit--}*

Set DIO sebagai Input

*{--ambil data dan simpan di array--}*

CLK ke 8, dValue[11] = DIO

CLK ke 9, dValue[10] = DIO

CLK ke 10, dValue[9] = DIO

CLK ke 11, dValue[8] = DIO

CLK ke 12, dValue[7] = DIO

CLK ke 13, dValue[5] = DIO

CLK ke 14, dValue[6] = DIO

CLK ke 15, dValue[4] = DIO

CLK ke 16, dValue[3] = DIO

CLK ke 17, dValue[2] = DIO

CLK ke 18, dValue[1] = DIO

CLK ke 19, dValue[0] = DIO

**For** i=11 to i=0 **begin**

Data += dValue[i]\*2^i

**End**.

CS=1

**Return** data

*{--end procedure--}*

Dari notasi algoritmik diatas, kemudian dikonversi kedalam bahasa C versi arduino karena kontroler yang digunakan adalah *arduino board*. Listing program procedure gatH48C(D1,D0) pada arduino IDE adalah sebagai berikut.

int getH48C(int D1,int D0){

digitalWrite(Cs, LOW);

pinMode(Dio, OUTPUT);

//\*\*\*1

digitalWrite(Clk, LOW);

digitalWrite(Dio, HIGH);// start bit

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*2

digitalWrite(Clk, LOW);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*3

digitalWrite(Clk, LOW);

digitalWrite(Dio, LOW);// D2 = 0

delayMicroseconds(1);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*4

digitalWrite(Clk, LOW);

if (D1==0){ digitalWrite(Dio, LOW);}

else{ digitalWrite(Dio, HIGH);}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*5

digitalWrite(Clk, LOW);

if (D0==0){ digitalWrite(Dio, LOW);}

else{ digitalWrite(Dio, HIGH);}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*6

digitalWrite(Clk, LOW);

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*7

pinMode(Dio, INPUT);

digitalWrite(Clk, LOW);

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*8

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[11]= 1 ;}

else {

dValue[11]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*9

digitalWrite(Clk, LOW);

if (D1==0){ digitalWrite(Dio, LOW);}

else{ digitalWrite(Dio, HIGH);}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*5

digitalWrite(Clk, LOW);

if (D0==0){ digitalWrite(Dio, LOW);}

else{ digitalWrite(Dio, HIGH);}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*6

digitalWrite(Clk, LOW);

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*7

pinMode(Dio, INPUT);

digitalWrite(Clk, LOW);

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*8

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[11]= 1 ;}

else {

dValue[11]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*9

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[10]= 1 ;}

else {

dValue[10]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*10

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[9]= 1 ;}

else {

dValue[9]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*11

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[8]= 1 ;}

else {

dValue[8]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*12

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[7]= 1 ;}

else {

dValue[7]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*13

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[6]= 1 ;}

else {

dValue[6]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*14

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[5]= 1 ;}

else {

dValue[5]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*15

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[4]= 1 ;}

else {

dValue[4]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*16

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[3]= 1 ;}

else {

dValue[3]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*17

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[2]= 1 ;}

else {

dValue[2]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*18

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[1]= 1 ;}

else {

dValue[1]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

//\*\*\*19

digitalWrite(Clk, LOW);

if (digitalRead(Dio)==HIGH){

dValue[0]= 1 ;}

else {

dValue[0]= 0 ;}

delayMicroseconds(50);

digitalWrite(Clk, HIGH);

delayMicroseconds(50);

data=(dValue[11]\*2048)+(dValue[10]\*1024)+(dValue[9]\*512)+(dValue[8]\*256)+(dValue[7]\*128)+(dValue[6]\*64)+(dValue[5]\*32)+(dValue[4]\*16)+(dValue[3]\*8)+(dValue[2]\*4)+(dValue[1]\*2)+(dValue[0]\*1);

digitalWrite(Cs,HIGH);

return data;

}

Untuk pembacaan nilai tegangan dilakukan dengan memanggil prosedur getH48C(D1,D0) dengan D1 = 1 dan D0 = 0 untuk membaca Vref, D1 = 0, D0 = 0 untuk membaca nilai sumbu X, D1 = 0, D0 = 1 untuk membaca nilai sumbu Y, dan D1 = 1 dan D0 = 0 untuk membaca nilai pada sumbu Z. Contoh pemanggilan prosedur untuk pembacaan nilai Vref, sumbu X, sumbu Y, dan sumbu Z adalah sebagai berikut.

int dX, dY, dZ = 0;

ref = getH48C(1,1);

dX = getH48C(0,0);

dY = getH48C(0,1);

dZ = getH48C(1,0);

Sesuai dengan persamaan 4.1, 4.2, dan 4.3 maka level tegangan untuk tiap sumbu dinyatakan dengan mengeksekusi perintah berikut.

float aX, aY, aZ = 0.00;

aX = (((float)dX) - ((float)ref));

aY = (((float)dY) - ((float)ref));

aZ = (((float)dZ) - ((float)ref));

Nilai gX, gY, dan gZ masih dalam level tegangan yang dinyatakan dengan nilai ADC 12 bit (0 hingga 4095). Untuk menyatakan kedalam nilai g sesuai dengan jangkauan pembacaan sensor (-3 g hingga +3g) maka digunakan persamaan berikut.

*g* ...........................................(4.5)

Berdasarkan persamaan 4.5, nilai 4095 adalah nilai maksimum ADC 12 bit, 3.3 adalah tegangan suplai H48C dan 0.3663 adalah nilai tegangan keluaran 1g. Persamaan diatas disederhanakan menjadi persamaan 4.6 berikut.

*g = (data Axis – Vref) x 0.0022* ...........................................(4.6)

Sehingga, *source code* untuk pembacaan nilai g untuk tiap sumbu dengan satuan *g* adalah sebagai berikut.

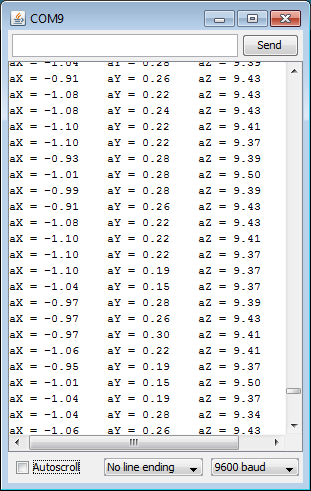
float aX, aY, aZ = 0.00;

aX = (((float)dX) - ((float)ref)) \*0.0022;

aY = (((float)dY) - ((float)ref)) \*0.0022;

aZ = (((float)dZ) - ((float)ref)) \*0.0022;

Hasil pengujian program untuk pembacaan nilai g menggunakan *source code* diatas dilihat menggunakan serial monitor dengan *baud* *rate* 9600 bps sesuai dengan gambar 4.7.

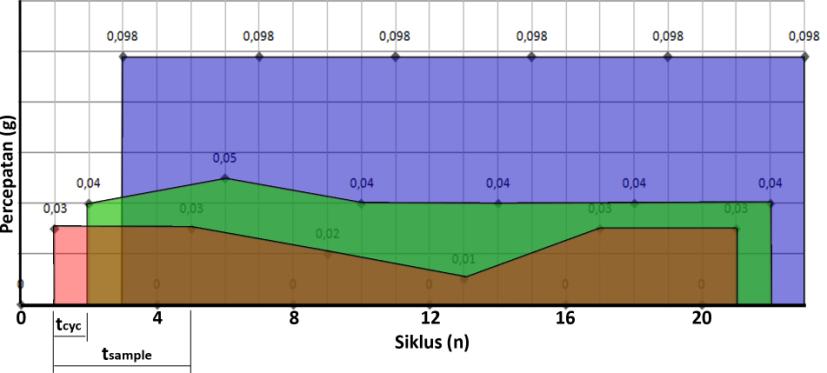
**

*Gambar 4.7 Hasil pembacaan modul accelerometer*

Pembacaan diatas memberikan hasil berupa percepatan sesaat tiap sampel. Dengan mengetahui dan jumlah *clock* dalam satu siklus pengambilan data, maka dapat dihitung periode sampel dengan cara berikut.

* = 100 µs dalam satu siklus pengambilan data terdapat 20 *clock* sehingga periode 1 siklus (tcycle) sebesar 20 kali 100 µs sehingga tcycle = 2000 µs atau 2 ms
* Algoritma pengambilan data dilakukan secara bergantian dari Vref, dX, dY, dan dZ dan masing – masing proses pengambilan data merupakan 1 suklus pengambilan data sehingga periode sampel (tsample) tiap sumbu sebesar 4 kali tcycle sehingga tsample = 8 ms.

Sebagai ilustrasi ditunjukan oleh grafik dibawah (gambar 4.8) dimana untuk mengambil data sebanyak 6 sampel dari tiap sumbu diperlukan sebanyak 24 siklus pengambilan data. Siklus pertama (n = 0) diawali dengan pengambilan data Vref, kemudian pada n = 1 diambil data percepatan sumbu x (daerah berwana merah) kemudian n = 2 diambil data percepatan sumbu y ( daerah berwarna hijau) kemudian dilanjutkan untuk n = 3 diambil data sumbu z (daerah berwarna biru) dan pada n = 4 diulangi lagi untuk pengambilan data Vref dan seterusnya.



*Gambatr 4.8 Ilustrasi penga mbilan* data *sebanyak 6 sampel untuk tiap – tiap sumbu*

Setelah mengetahui periode sampel tiap sumbu adalah 8 ms maka jumlah sampel dalam 1 detik sebanyak 125 sampel.

### Karakteristik Akselerometer H48C

Akselerometer H48C dapat mengukur percepatan pada sumbu X, Y dan Z baik percepatan statik maupun percepatan dinamik.

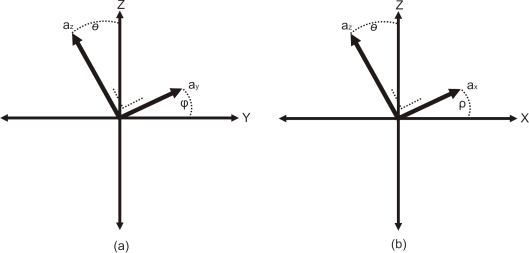
Percepatan statik merupakan percepatan yang dipengaruhi oleh gaya gravitasi. Sedangkan percepatan dinamik dipengaruhi oleh gaya dari luar selain gaya gravitasi.

1. **Percepatan Statik**

Pada kondisi pembacaan percepatan statik, nilai percepatan tiap sumbu dipengaruhi oleh kemiringan sensor terhadap bidang referensi dalam hal ini adalah gaya gravitasi. Sehingga kondisi statik ideal untuk aplikasi pembacaan kemiringan suatu objek.

Nilai percepatan yang diberikan merupakan representasi posisi dari sensor terhadap pusat gravitasi dan magnitude vektor percepatan dari 3 sumbu yang diberikan sama dengan 1g. Kemiringan sensor memiliki sudut *pitch* (ρ), *roll* (φ) dan *theta* (θ).

*Ρitch* (ρ) merupakan sudut antara sumbu X relatif terhadap bumi. *Roll* (φ) merupakan sudut antara sumbu Y relativ terhadap bumi. Sedangkan *theta* (θ) merupakan sudut antara sumbu Y relatif terhadap gravitasi. Sudut ρ , φ, dan θ masing – masing ditunjukan oleh gambar 4.9 dalam representasi 2 dimensi.

**

*Gambar 4.9. Sudut roll (φ) dalam plot dua dimensi antara sumbu Y dan Z (gambar a), Sudut pitch (ρ) dalam dua dimensi antara sumbu X dan Z.*

Akselerometer mengukur percepatan statis pada tiap sumbu dan memberikan data berupa vektor percepatan sehingga kita dapat menghitung besar sudut yang terbentuk pada masing – masing sudut ρ, φ, dan θ dengan persamaan 4.8, 4.9 dan 4.10 berikut.

............................................................(4.8)

...........................................................(4.9)

.........................................................(4.10)

1. **Pengujian Percepatan Statik**

Pengujian pengukuran statik dilakukan untuk mengetahui karakteristik akselerometer H48C pada saat bergerak statis.

Dalam pengujian ini dilakukan dua skema yaitu :

1. Pengambilan sudut ρ dan θ dengan cara memutar sensor pada sumbu X ( sebesar satu putaran penuh (3600 ) dan kemudian memutar sensor sebesar ¼ putaran pada arah positif kemudian kembali lagi ketitik awal ( ¼ putaran arah negatif).
2. Pengambilan sudut φ dan θ dengan cara memutar sensor pada sumbu Y sebesar satu putaran penuh (3600 ) dan kemudian memutar sensor sebesar ¼ putaran pada arah positif kemudian kembali lagi ketitik awal ( ¼ putaran arah negatif)

Nilai M (resultan) diisi dengan menggunakan persamaan . Nilai M diperlukan untuk menguji apakah sensor bekerja statis atau tidak. Dapat dilihat bahwa dari data percepatan yang diberikan akselerometer pada pengujian pertama, tampak bahwa M ≈ 1g. Nilai resultan vektor percepatan pada kondisi statik tidak melebihi dari nilai percepatan gravitasi sedangkan adanya nilai yang lebih besar dari 1 dari percobaan pertama diatas dikarenakan sensor melakukan pembacaan pada saat sensor diputar yang artinya sensor menerima gaya dari luar yang nilainya kecil.

Apabila dari tabel diatas dibuat grafik maka vektor percepatan sumbu X (Ax) merupakan fungsi *sinus* dan vektor percepatan pada sumbu Z (Az) merupakan fungsi *cosinus*.

*Gambar 4.10 Grafik percepatan pada akselerometer bergerak statik dengan Ay≈0 bergerak 3600*

Pengujian kedua adalah menggerakan modul pada sumbu X (Sumbu Y = 0) sebesar 900 kearah positif kemudian kembali lagi sebesar 900 kearah negatif. Hasil pembacaan ditunjukan pada gambar 4.11 dibawah.

*Gambar 4.11 Grafik percepatan pada akselerometer bergerak statik dengan Ay≈0 bergerak 900 bolak – balik.*

Skema pengujian kondisi statik yang kedua adalah dengan memutar sensor pada sumbu Y (X=0) sebesar 3600 dan dilanjutkan dengan memutar sensor pada sumbu Y sebesar 900 bolak balik. Didapat hasil sesuai gambar 4.12 dan 4.13.

*Gambar 4.12 Grafik percepatan pada akselerometer bergerak statik dengan Ax≈0 bergerak 3600.*

*Gambar 4.13 Grafik percepatan pada akselerometer bergerak statik dengan Ax≈0 bergerak 900 bolak – balik.*

Baik vektor Ax maupun vektor Ay keduanya merupakan fungsi *sinus* sedangkan vektor Az merupakan fungsi *cosinus* sehingga sudutρ,φ dan θ dapat dicari dengan menggunakan persamaan 4.8, 4.9 dan 4.10.

Akan tetapi fungsi *arctan* akan memberikan nilai pada jangkauan -900 hingga 900 sehingga persamaan 4.8, apabila diterapkan untuk menghitung sudut ρ dari hasil pembacaan akselerometer pada tabel 4.2 akan memberikan nilai dari -900 hingga 900 seperti pada gambar 4.14. Padahal pergerakan sesungguhnya adalah dari 00 hingga 3600. Fungsi *arctan* juga akan memberikan hasil yang sama untuk nilai sudut pada kuadran I dan kuadran III sebagai contoh untuk sudut 2250, nilai *tan*(2250)=1 sedangkan hasil dari fungsi *arctan* 1 = 450. Persamaan 4.8, 4.9 dan 4.10 juga akan memberikan nilai yang salah apabila modul berputar pada kuadran II dan IV.

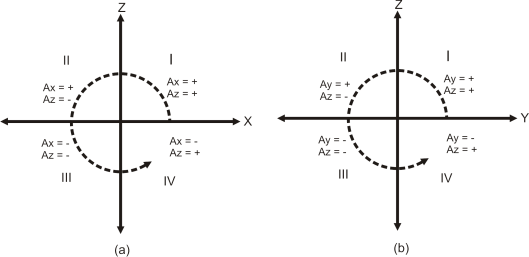
*Gambar 4.14 Nilai yang didapat apabila menggunakan persamaan 4.8 untuk menghitung sudut ρ dari data akselerometer pada tabel 4.2.*

Sudut – sudut pada kuadran II hingga kuadran IV dapat diperoleh dengan memodifikasi persamaan 4.8, 4.9, dan 4.10 sebagai berikut.

* Kuadran I
* Kuadran II
* Kuadran III
* Kuadran IV

Untuk mengetahui dikuadran mana modul bergerak dapat dilakukan dengan membaca tanda bilangan positif (+) atau negatif (-) pada masing – masing data vektor percepatan.

Penentuan daerah kuadran dapat dilakukan dengan analisis vektor percepatan pada tiap dua sumbu yaitu X-Z dan Y-Z. Berdasarkan analisa pada gerak melingkar 3600 pad grafik yang ditunjukan oleh gambar 4.10 dan 4.12 maka daerah kiadran I – IV dapat diketahui jika tanda bilangan vektor percepatan Ax, Ay dan Az sesuai gambar 4.15.



*Gambar 4.15 Penentuan daerah kuadran dengan menganalisa data vektor percepatan sumbu X dan Y terhadap sumbu Z.*

Setelah mengetahui karakteristik perubahan nilai percepatan pada perpitaran dalam masing – masing kuadran langkah selanjutnya adalah mengaplikasikan persamaan yang didapat kedalam data percepatan hasil pengujian. Hasil perhitungan menggunakan persamaan yang telah dimodifikasi untuk data pada tabel 4.2 ditunjukan grafik pada gambar 4.16 dibawah. Sedangkan data selengkapnya mengenai pengujian percepatan statis dapat dilihat di lampiran 1.

*Gambar 4.16 Nilai sudut ρ yang dihasilkan apabila pososisi kuadran diketahui*

1. **Percepatan Dinamik**

Akselerometer H48C selain dapat mengukur keadaan percepatan statik juga dapat mengukur percepatan dinamik. Pada pengukuran percepatan dinamik, nilai percepatan yang didapat dipengaruhi oleh gaya gravitasi berikut gaya dari luar yang menyebabkan modul akselerometer bergerak.

Sehingga pada kondisi modul bergerak nilai resultan gaya dari akselerometer yang dibaca lebih besar dari 1g. Untuk mengetahui besar gaya yang bekerja pada modul yang menyebabkan modul bergerak dapat menggunakan persamaan berikut.

..............................................................(4.11)

Dengan Ftotal merupakan jumlah keseluruhan gaya yang bekerja pada modul, Fg merupakan gaya gravitasi dan Feks merupakan gaya eksternal yang bekerja pada modul. Dengan mengingat bahwa F = m.a maka persamaan 4.11 dapat dirubah menjadi.

.............................................................(4.12)

Dimana, *atotal* adalah percepatan total yang bekerja pada benda, *ag*adalah percepatan benda akibat pengaruh gaya gravitasi dan *aeks* merupakan percepatan benda akibat gaya eksternal seperti dorongan atau tarikan.

Karena daerah kerja dari sensor merupakan bidang tiga dimensi maka persamaan 4.12 berlaku untuk masing – masing sumbu x, y dan z. Sehingga:

Ketika dalam kondisi diam vektor percepatan yang bekerja pada benda hanya karena pengaruh gaya gravitasi sedangkan ketika benda bergerak, vektor percepatan benda merupakan jumlah dari vektor percepatan akibat gaya gravitasi dan vektor percepatan akibat gaya eksternal. Sehingga percepatan akibat gaya eksternal dapat dicari dengan mengurangi nilai percepatan pada saat benda bergerak dengan percepatan pada saat benda diam. Yang mana percepatan pada saat benda diam adalah *ag*. Akan tetapi nilai percepatan eksternal tidaklah murni percepatan yang mengakibatkan benda bergerak, didalamya terdapat vektor percepatan yang disebabkan oleh gaya gesekan benda, vibrasi, derau internal sensor dan lain sebagainya. Nilai percepatan ini dapat dikatakan sebagai faktor kesalahan (*error*). Sehingga percepatan benda bergerak sesungguhnya adalah sesuai persamaan 4.13 berikut.

.................................................(4.13)

1. **Pengujian Percepatan Dinamik**

Untuk pengujian percepatan dinamik dilakukan dengan mengambil data yang resultan vektor percepatanya lebih besar dari pada resultan vektor percepatan pada saat modul diam. Sehingga pada pengujian ini diperlukan data nilai percepatan pada saat modul tidak bergerak sebagai nilai acuan.

Data nilai percepatan awal pada saat modul diam diambil dari rata – rata nilai percepatan pada 50 sampel. Untuk proses inisialisasi ini diperlukan penambahan prosedur pada mikrokontroler dan menyimpan nilai resultan percepatan percepatan sebagai nilai acuan pengiriman data ke komputer.

Data yang dikirim ke komputer adalah data percepatan benda bergerak yang didalamnya terdapat faktor kesalahan atau *error* akan tetapi untuk proses pengujian ini nilai *error* masih diikut sertakan untuk analisa lebih lanjut.

Prosedur inisialisasi data menggunakan *source code* sebagai berikut.

float x0 = 0;

float y0 = 0;

float z0 = 0;

float m0 = 0;

void inisialisasi(){

for(int i=0;i<=50;i++){

tX = getH48C(50,0,0);

Xinit[i] =(((float)(tX-2048)))\*0.0022;

tY = getH48C(50,0,1) ;

Yinit[i] =(((float)(tY-2048)))\*0.0022;

tZ = getH48C(50,1,0) ;

Zinit[i] =(((float)(tZ-2048)))\*0.0022;

res[i] = sqrt(sq(Xinit[i])+sq(Yinit[i])+sq(Zinit[i]));

m0 = max(m0 , res[i]);

x0 = x0 + Xinit[i];

y0 = y0 + Yinit[i];

z0 = z0 + Zinit[i];

}

x0 = x0 \* 0.02;

y0 = y0 \* 0.02;

z0 = z0 \* 0.02;

}

Prosedure inisialisasi ini akan memberikan nilai pada variabel global x0,y0,z0 dan m0. Sedangkan data yang dikirim ke komputer merupakan data nilai percepatan yang resultan vektor percepatanya lebih besar dari m0 yakni dengan memanfaatkan percabangan *if(....)* sebagai berikut.

M=sqrt(sq(aX)+sq(aY)+sq(aZ));

if (M > m0){

aX = aX - x0; //nilai percepatan dikurangi

aY = aY - y0; // percepatan awal

aZ = aZ - z0;

Serial.print(aX);

Serial.print(9,BYTE); //karakter Tab

Serial.print(aY);

Serial.print(9,BYTE);

Serial.print(aZ);

Serial.print(9,BYTE);

Serial.println(M);

Kemudian setelah meng-*upload* *source code* ke mikrokontroler dilakukan percobaan menggrekan modul pada tiap sumbu sejauh jarak tertentu untuk mendapatkan karakteristik pergerakan modul. Tiap sumbu memiliki arah positif dan negatif sehingga pada uji coba dilakukan pengambilan data tiap sumbu sebanyak dua buah data yakni pada pegerakan positif dengan jarak tertentu dan pada pergerakan negatif pada jarak tertentu.

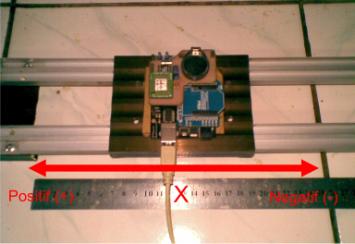
Pengujian untuk tiap sumbu di ilustrasikan oleh gambar 4.17. Untuk pengujian pergerakan tiap sumbu diambil data masing – masing sumbu untuk masing – masing arah pada perpindahan sejauh 10 cm, 20 cm, 30cm, 40cm serta 50 cm kemudian dibuat grafik untuk mempermudah analisa.

Data pengujian pergerakan pada sumbu x dengan arah positif ditunjukan pada tabel 4.3 , sedangkan grafik yang terbentuk ditunjukan oleh gambar 4.18.

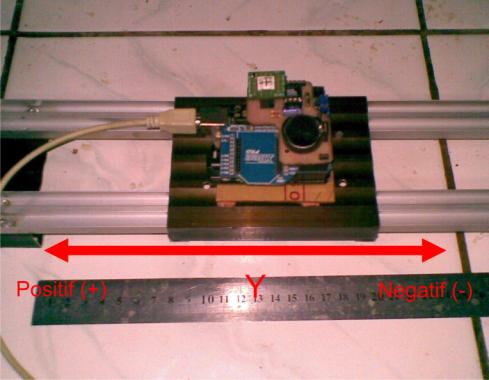
Data pengujian pergerakan pada sumbu x dengan arah negatif ditunjukan pada tabel 4.4 , sedangkan grafik yang terbentuk ditunjukan oleh gambar 4.19.

Data pengujian pergerakan pada sumbu x dengan arah negatif ditunjukan pada tabel 4.4 , sedangkan grafik yang terbentuk ditunjukan oleh gambar 4.19.

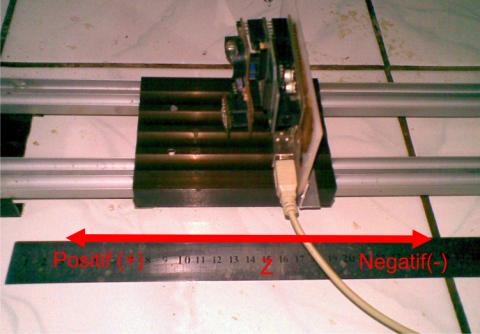
Data pengujian pergerakan pada sumbu y dengan arah positif ditunjukan pada tabel 4.5 , sedangkan grafik yang terbentuk ditunjukan oleh gambar 4.20.



(a)



(b)

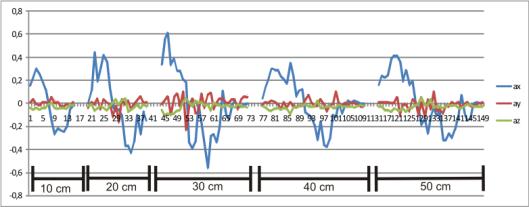


(c)

*Gambar 4.17Posisi modul pada saat pengujian pergerakan tiap sumbu. (a) sumbu x, (b) sumbu y dan (c) sumbu z*

Tabel. 4.3 Data pengujian pergerakan pada sumbu x arah positif

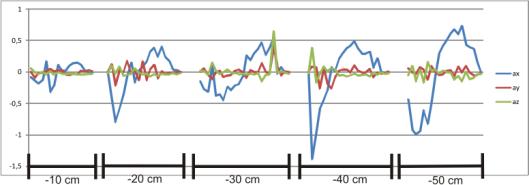
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 cm | | | 20 cm | | | 30 cm | | | 40 cm | | | 50 cm | | |
| ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az |
| 0.15 | 0 | -0 | 0 | -0 | -0 | 0.34 | -0 | -0.1 | 0.04 | -0 | -0 | 0.16 | 0.01 | -0 |
| 0.22 | 0.03 | -0.1 | 0.11 | 0.01 | -0 | 0.55 | 0.02 | -0.1 | 0.14 | 0.01 | -0 | 0.24 | 0.01 | -0 |
| 0.3 | -0 | -0 | 0.44 | -0 | -0.1 | 0.61 | 0.06 | -0.1 | 0.21 | 0 | -0.1 | 0.22 | 0.01 | -0.1 |
| 0.25 | -0 | -0.1 | 0.19 | 0.04 | -0.1 | 0.33 | -0.1 | -0.1 | 0.3 | 0.02 | -0.1 | 0.24 | 0.01 | -0.1 |
| 0.19 | 0.01 | -0 | 0.3 | -0.1 | -0 | 0.39 | 0.05 | -0.1 | 0.29 | 0.01 | -0.1 | 0.38 | 0 | -0.1 |
| 0.12 | 0.01 | -0 | 0.42 | 0.03 | -0 | 0.28 | 0.08 | -0.1 | 0.29 | -0 | -0 | 0.41 | 0 | -0.1 |
| -0 | -0 | -0 | 0.36 | 0 | -0.1 | 0.28 | -0 | -0.1 | 0.23 | -0.1 | -0.1 | 0.41 | 0.05 | -0.1 |
| -0.2 | 0.05 | 0 | 0.13 | -0 | -0 | 0.21 | 0.1 | 0.01 | 0.21 | 0.01 | -0 | 0.36 | -0.1 | -0.1 |
| -0.3 | 0 | -0 | -0 | -0.1 | 0 | 0.18 | -0.2 | 0.04 | 0.17 | -0 | -0.1 | 0.19 | -0 | -0.1 |
| -0.2 | 0.01 | -0.1 | -0.1 | -0 | 0.03 | -0.3 | -0 | 0.01 | 0.35 | 0.01 | -0.1 | 0.29 | -0 | -0.1 |
| -0.2 | -0 | -0 | -0.1 | -0.2 | -0 | -0.4 | 0.04 | -0 | 0.23 | -0 | -0 | 0.16 | -0.1 | -0 |
| -0.3 | -0 | -0 | -0.1 | 0.03 | 0.02 | -0.3 | -0.1 | 0.02 | 0.06 | -0 | -0 | 0.19 | 0.03 | -0 |
| -0.2 | -0 | -0.1 | -0.4 | -0 | 0.04 | -0 | -0.1 | 0 | 0.11 | -0 | -0 | 0.01 | 0.01 | -0 |
| -0.1 | -0 | -0 | -0.4 | 0.02 | -0.1 | -0.2 | 0.08 | 0.01 | 0.12 | -0 | -0 | -0 | -0.1 | -0 |
| -0 | 0.01 | -0 | -0.4 | -0.1 | -0.1 | -0.4 | -0 | -0 | -0.1 | 0.02 | -0 | -0.2 | 0.01 | 0.05 |
|  |  |  | -0.3 | -0 | -0 | -0.6 | 0.07 | -0 | -0.1 | 0.07 | -0 | -0.1 | -0.1 | -0 |
|  |  |  | -0.1 | 0.02 | -0 | -0.3 | 0.09 | -0.1 | -0.1 | 0 | -0 | -0.3 | 0.03 | -0 |
|  |  |  | -0.3 | 0.06 | -0.1 | -0.3 | -0.1 | 0 | -0.2 | -0 | -0 | -0.1 | -0.1 | 0.03 |
|  |  |  | -0.1 | 0 | -0 | -0.3 | 0 | -0 | -0.3 | -0 | 0.02 | -0 | 0.1 | -0 |
|  |  |  | -0.2 | 0.01 | -0 | -0.2 | 0.04 | -0 | -0.2 | 0.03 | -0 | -0.1 | -0.1 | -0 |
|  |  |  |  |  |  | 0.11 | 0.04 | -0 | -0.4 | -0 | -0 | -0.1 | -0.1 | -0 |
|  |  |  |  |  |  | -0.1 | 0 | -0 | -0.4 | -0.1 | -0 | -0.3 | -0.1 | -0.1 |
|  |  |  |  |  |  | -0.1 | 0 | -0 | -0.3 | -0 | -0 | -0.3 | -0 | -0 |
|  |  |  |  |  |  | -0 | 0.03 | -0 | -0.1 | -0 | -0 | -0.3 | -0 | -0 |
|  |  |  |  |  |  | 0.02 | 0.03 | -0 | -0.1 | 0.1 | -0 | -0.3 | 0.01 | -0 |
|  |  |  |  |  |  | -0 | -0.1 | -0 | 0.01 | -0 | -0 | -0.2 | -0 | -0 |
|  |  |  |  |  |  | -0 | 0.03 | -0 | -0.1 | -0 | -0 | -0.1 | 0.01 | -0 |
|  |  |  |  |  |  | -0 | 0.06 | -0 | -0 | 0 | -0 | 0.07 | -0 | -0 |
|  |  |  |  |  |  | 0 | 0.05 | -0 | 0.02 | -0 | -0 | -0.1 | -0 | -0 |
|  |  |  |  |  |  |  |  |  | 0 | 0.01 | -0 | -0.2 | 0.01 | -0 |
|  |  |  |  |  |  |  |  |  | 0.02 | 0 | -0 | -0.1 | -0 | -0 |
|  |  |  |  |  |  |  |  |  | 0.01 | -0 | -0 | 0 | -0 | -0 |
|  |  |  |  |  |  |  |  |  | 0 | -0 | -0 | 0.01 | -0 | -0 |
|  |  |  |  |  |  |  |  |  | 0 | -0 | -0 | 0 | 0 | -0 |
|  |  |  |  |  |  |  |  |  |  |  |  | 0.01 | 0 | -0 |



*Gambar 4.18 Grafik pengujian pergerakan modul pada sumbu x positif*

Tabel 4.4 Data pengujian pergerakan sumbu x pada arah negatif

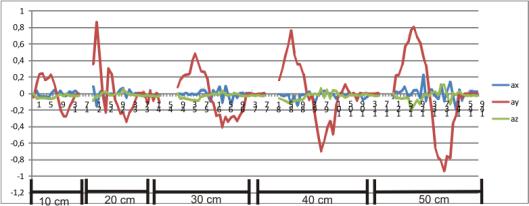
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| - 10 cm | | | - 20 cm | | | - 30 cm | | | - 40 cm | | | - 50 cm | | |
| ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az |
| -0.1 | 0 | 0.06 | 0 | -0 | -0 | -0.2 | 0 | 0.02 | 0 | 0 | -0 | -0.4 | 0.06 | -0.1 |
| -0.1 | -0.1 | -0 | -0.4 | 0.12 | 0.1 | -0.3 | 0.06 | 0.02 | -1.4 | 0.08 | 0.39 | -0.9 | -0 | 0.09 |
| -0.2 | -0 | -0 | -0.8 | -0.2 | -0 | -0.3 | -0.1 | -0.1 | -1 | 0.07 | -0.2 | -1 | -0 | -0.2 |
| -0.1 | -0 | 0 | -0.6 | 0.03 | 0.08 | 0.09 | -0.1 | 0.15 | -0.6 | -0.3 | -0.1 | -1 | 0.07 | 0.04 |
| 0.17 | 0.01 | -0.1 | -0.4 | -0 | -0.1 | -0.4 | -0 | 0.03 | -0.5 | 0.06 | 0.08 | -0.6 | 0.04 | -0.2 |
| -0.3 | 0.05 | 0.01 | -0.1 | 0.18 | -0.1 | -0.4 | 0.02 | -0 | -0.3 | -0.2 | 0.01 | -0.8 | -0.1 | 0.16 |
| -0.2 | 0.01 | -0.1 | 0.15 | -0 | -0 | -0.5 | -0.1 | -0 | 0.07 | -0.3 | 0.06 | -0.5 | 0.09 | 0.03 |
| 0.11 | 0.03 | -0 | -0.2 | 0.02 | 0.06 | -0.2 | -0.1 | 0.04 | 0.01 | -0.1 | 0.04 | -0.1 | 0 | 0.04 |
| 0.03 | 0.03 | -0 | 0.02 | -0.1 | -0 | -0.3 | -0.1 | -0 | 0.2 | 0.02 | -0.1 | 0.28 | -0.1 | -0 |
| 0.02 | -0 | -0 | 0.07 | 0.17 | -0 | -0.2 | 0.02 | -0 | 0.29 | 0.04 | -0.1 | 0.41 | -0 | 0.02 |
| 0.09 | -0.1 | -0 | 0.16 | -0.1 | -0 | -0.2 | 0.04 | -0 | 0.37 | 0.04 | -0.1 | 0.5 | 0.02 | 0 |
| 0.14 | -0 | -0 | 0.32 | 0.04 | -0.1 | -0.1 | -0 | -0 | 0.41 | -0 | -0.1 | 0.61 | 0.03 | -0 |
| 0.15 | -0.1 | -0 | 0.38 | 0.11 | -0 | 0.26 | -0.1 | -0 | 0.49 | 0.1 | -0 | 0.67 | -0.1 | -0.1 |
| 0.11 | 0.02 | -0 | 0.24 | -0.1 | -0.1 | 0.19 | 0.03 | -0 | 0.37 | 0.02 | -0.1 | 0.6 | 0.08 | -0.1 |
| 0.03 | 0 | -0 | 0.4 | 0.01 | -0 | 0.29 | 0 | -0.1 | 0.29 | -0 | -0.1 | 0.73 | -0 | -0.2 |
| -0 | 0.03 | -0 | 0.23 | -0 | -0 | 0.36 | 0 | -0 | 0.29 | -0 | -0 | 0.43 | 0.09 | 0 |
| -0 | 0.01 | -0 | 0.17 | -0 | -0 | 0.47 | 0.14 | -0.2 | 0.32 | 0.05 | -0.1 | 0.39 | -0 | -0.1 |
|  |  |  | 0.04 | -0 | -0 | 0.27 | 0 | -0.1 | 0.06 | -0.1 | -0 | 0.36 | -0.1 | -0.1 |
|  |  |  | 0.03 | 0 | -0 | 0.37 | -0 | -0 | 0.21 | -0 | -0.1 | 0.15 | -0.1 | -0 |
|  |  |  | -0 | 0 | -0 | 0.53 | 0.39 | 0.65 | 0 | -0 | -0 | -0 | -0 | -0 |
|  |  |  |  |  |  | -0 | 0.11 | -0 | -0 | -0 | -0 |  |  |  |
|  |  |  |  |  |  | 0.07 | -0.1 | 0.01 | 0 | 0.01 | -0 |  |  |  |



*Gambar 4.19 Grafik pengujian pergerakan modul pada sumbu x negatif*

Tabel 4.5 Data pengujian pergerakan sumbu y pada arah positif

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 cm | | | 20 cm | | | 30 cm | | | 40 cm | | | 50 cm | | |
| ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az |
| -0 | -0 | 0 | 0.08 | 0.36 | -0.1 | 0 | 0.08 | 0 | -0 | 0.16 | -0.1 | -0.1 | -0 | 0.01 |
| 0.03 | 0.14 | -0.1 | -0.2 | 0.87 | -0 | 0 | 0.16 | -0.1 | -0 | 0.3 | -0.1 | 0.02 | 0.22 | -0.1 |
| -0 | 0.24 | -0.1 | 0.01 | 0.41 | -0 | -0 | 0.21 | -0.1 | -0 | 0.45 | -0.1 | 0.01 | 0.23 | -0.1 |
| -0 | 0.25 | -0.1 | 0.04 | 0.01 | 0.01 | 0.01 | 0.23 | -0.1 | -0 | 0.59 | -0.1 | 0.03 | 0.35 | -0.1 |
| -0 | 0.16 | -0.1 | -0.1 | -0.2 | 0.02 | -0 | 0.24 | -0.1 | -0.1 | 0.77 | -0.1 | 0.08 | 0.56 | -0.1 |
| -0.1 | 0.19 | -0.1 | 0 | 0.31 | 0 | 0 | 0.38 | -0.1 | -0 | 0.49 | -0.1 | -0 | 0.62 | -0.1 |
| 0 | 0.23 | -0.1 | 0 | 0.24 | -0 | -0 | 0.49 | -0.1 | -0.2 | 0.36 | -0.1 | -0 | 0.76 | -0.2 |
| 0.04 | 0.12 | -0.1 | -0 | -0.1 | -0 | 0 | 0.37 | -0.1 | -0 | 0.36 | -0.1 | 0.04 | 0.81 | -0.2 |
| -0 | -0.1 | -0 | 0 | -0.1 | 0 | 0.04 | 0.27 | -0 | -0 | 0.24 | -0.1 | 0.04 | 0.68 | -0.1 |
| 0.03 | -0.2 | -0 | 0.05 | -0.3 | 0 | 0.04 | 0.26 | -0 | -0 | 0.04 | -0 | -0.1 | 0.62 | -0.1 |
| -0 | -0.3 | -0 | 0.07 | -0.2 | 0.05 | 0 | 0.19 | -0 | 0.02 | -0.1 | 0 | 0.23 | 0.44 | -0 |
| -0 | -0.3 | 0 | -0.1 | -0.3 | 0.01 | -0.1 | -0.1 | -0 | -0 | -0 | -0 | -0.1 | 0.35 | -0 |
| 0.02 | -0.2 | -0 | 0.05 | -0.2 | -0 | 0.04 | -0.1 | 0 | -0.1 | -0.3 | 0 | 0.03 | 0.01 | -0 |
| 0.01 | -0.1 | -0 | -0 | -0.2 | -0 | -0 | -0.3 | 0 | 0 | -0.5 | -0 | 0.05 | -0.2 | 0.01 |
| 0.03 | -0 | -0 | 0.01 | -0.1 | -0 | 0.08 | -0.3 | -0 | 0 | -0.7 | 0.07 | -0 | -0.6 | 0.02 |
| -0 | 0 | -0 | -0 | 0.01 | -0 | -0.1 | -0.4 | 0.03 | 0.03 | -0.6 | -0.1 | -0.1 | -0.8 | -0 |
|  |  |  | -0 | 0.01 | -0 | 0.09 | -0.3 | -0 | 0.01 | -0.4 | 0.05 | 0.11 | -0.8 | 0.1 |
|  |  |  | -0 | 0.02 | -0 | -0 | -0.3 | -0 | 0 | -0.3 | -0.1 | -0.1 | -0.9 | 0.11 |
|  |  |  | -0 | -0.1 | -0 | -0.1 | -0.3 | -0 | 0 | -0.5 | 0.04 | 0.01 | -0.8 | 0 |
|  |  |  | -0 | 0.04 | -0 | 0.01 | -0.3 | 0 | -0.1 | -0.1 | 0.03 | 0.14 | -0.8 | -0.1 |
|  |  |  | -0 | -0.1 | -0 | -0.1 | -0.3 | 0 | -0 | -0 | -0 | -0 | -0.4 | 0.01 |
|  |  |  | 0 | 0.01 | -0 | 0.01 | -0.3 | -0 | 0.01 | 0.02 | -0 | -0.2 | -0.3 | -0 |
|  |  |  | -0 | -0.1 | -0 | 0.01 | -0.2 | -0 | -0 | 0.11 | -0.1 | 0.05 | 0.01 | 0.02 |
|  |  |  |  |  |  | -0 | 0.01 | -0 | 0 | 0.04 | -0 | -0.1 | -0.1 | -0 |
|  |  |  |  |  |  | 0.01 | 0 | -0 | -0 | -0 | -0 | 0 | 0 | -0 |
|  |  |  |  |  |  | 0 | 0 | -0 | -0 | 0.01 | -0 | -0 | 0 | -0 |
|  |  |  |  |  |  | -0 | -0 | -0 | -0 | -0.1 | -0 | 0.03 | 0 | -0 |
|  |  |  |  |  |  | -0 | 0 | -0 | 0 | 0.01 | -0 | 0.02 | 0 | -0 |
|  |  |  |  |  |  | 0 | 0 | -0 | -0.1 | 0 | -0 | 0.02 | 0.01 | -0 |
|  |  |  |  |  |  | 0.01 | 0 | -0 | -0.1 | 0.01 | 0 |  |  |  |
|  |  |  |  |  |  |  |  |  | -0 | 0 | -0 |  |  |  |
|  |  |  |  |  |  |  |  |  | -0 | 0 | -0 |  |  |  |
|  |  |  |  |  |  |  |  |  | 0 | 0 | -0 |  |  |  |
|  |  |  |  |  |  |  |  |  | -0 | 0.01 | -0 |  |  |  |

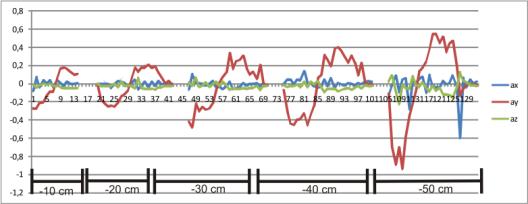


*Gambar 4.20 Grafik pengujian pergerakan modul pada sumbu y positif*

Data pengujian pergerakan pada sumbu y dengan arah negatif ditunjukan pada tabel 4.6 , sedangkan grafik yang terbentuk ditunjukan oleh gambar 4.21.

Tabel 4.6 Data pengujian pergerakan sumbu y pada arah negatif

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -10 cm | | | -20 cm | | | -30 cm | | | -40 cm | | | -50 cm | | |
| ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az |
| -0.1 | -0.3 | -0 | 0 | -0 | 0 | -0.1 | -0.4 | 0.01 | -0 | -0.1 | -0 | -0.1 | -0.1 | 0.02 |
| 0.08 | -0.3 | -0.1 | 0 | -0.2 | -0 | 0.11 | -0.5 | -0 | 0.05 | -0.3 | -0 | -0 | -0.7 | 0.09 |
| -0 | -0.2 | -0 | 0 | -0.2 | -0 | 0.01 | -0.2 | 0.07 | 0.05 | -0.4 | -0 | 0.09 | -0.9 | -0.1 |
| 0.04 | -0.2 | 0 | 0 | -0.3 | -0 | 0.01 | -0.3 | -0 | -0 | -0.5 | -0 | -0.1 | -0.7 | -0.2 |
| 0 | -0.1 | -0 | -0.1 | -0.2 | -0 | 0 | -0.3 | -0 | 0.05 | -0.4 | -0 | 0.05 | -0.9 | -0 |
| 0.04 | -0.1 | 0 | -0 | -0.3 | 0 | -0 | -0.3 | 0.01 | 0.04 | -0.4 | -0 | 0.06 | -0.6 | -0.1 |
| -0 | -0.1 | -0 | 0.03 | -0.2 | -0 | 0 | -0.3 | -0 | 0.14 | -0.3 | 0.02 | -0.3 | -0.4 | -0.1 |
| 0.04 | 0.06 | -0 | 0.02 | -0.1 | 0.02 | 0.03 | -0.2 | -0 | 0 | -0.5 | -0 | 0.05 | -0.2 | 0.08 |
| 0.01 | 0.17 | -0 | 0.01 | -0.1 | -0 | -0 | -0.1 | 0.02 | 0.01 | -0.3 | 0.02 | 0.01 | -0.1 | -0 |
| -0 | 0.18 | -0.1 | -0 | -0.1 | -0 | -0 | 0.01 | -0 | -0 | -0.3 | -0 | 0.05 | 0.13 | -0 |
| 0.02 | 0.16 | -0.1 | 0 | 0.04 | -0 | -0.1 | 0.07 | 0.02 | -0.1 | 0.06 | 0.03 | 0.08 | 0.17 | 0.01 |
| 0 | 0.13 | -0.1 | 0.05 | 0.18 | -0.1 | -0 | 0.12 | -0 | -0 | 0.09 | -0 | -0.1 | 0.24 | -0.1 |
| 0 | 0.1 | -0.1 | -0.1 | 0.13 | 0.07 | 0.02 | 0.34 | -0.1 | 0.05 | 0.32 | -0.1 | 0.1 | 0.37 | -0.1 |
| 0.01 | 0.11 | -0.1 | -0 | 0.18 | -0.1 | -0.1 | 0.16 | -0.1 | -0 | 0.26 | -0.1 | -0 | 0.55 | -0 |
|  |  |  | 0.02 | 0.18 | -0.1 | -0.1 | 0.25 | -0.1 | 0.06 | 0.19 | -0.1 | -0 | 0.55 | -0.1 |
|  |  |  | -0 | 0.21 | -0.1 | -0 | 0.26 | -0.1 | -0.1 | 0.39 | -0.1 | -0 | 0.45 | -0.1 |
|  |  |  | 0.02 | 0.17 | -0.1 | 0.01 | 0.31 | -0.1 | 0.01 | 0.4 | -0.1 | 0.03 | 0.52 | -0.1 |
|  |  |  | -0 | 0.19 | -0.1 | -0 | 0.17 | -0.1 | -0 | 0.36 | -0.1 | -0 | 0.35 | -0.1 |
|  |  |  | 0.03 | 0.12 | -0 | 0 | 0.1 | -0.1 | -0 | 0.3 | -0.1 | 0.01 | 0.44 | -0.1 |
|  |  |  | -0 | 0.07 | -0 | 0 | 0.13 | -0.1 | 0.01 | 0.23 | -0.1 | 0.09 | 0.47 | -0.2 |
|  |  |  | 0.01 | 0 | -0 | 0.01 | 0.05 | -0 | 0 | 0.31 | -0.1 | -0 | 0.19 | -0.1 |
|  |  |  | -0 | 0.03 | -0 | -0 | 0.21 | -0.1 | -0 | 0.23 | -0.1 | -0.6 | -0.1 | 0.13 |
|  |  |  | 0 | -0 | -0 | 0.01 | -0 | -0 | 0.01 | 0.09 | -0 | 0.07 | -0.1 | 0.03 |
|  |  |  |  |  |  | -0 | -0 | -0 | -0 | 0.23 | -0.1 | -0 | -0 | 0.01 |
|  |  |  |  |  |  |  |  |  | 0.02 | -0 | -0 | 0.05 | 0 | -0 |
|  |  |  |  |  |  |  |  |  | 0.03 | 0 | -0 | 0 | -0 | -0 |
|  |  |  |  |  |  |  |  |  | 0.02 | 0 | -0 | 0.02 | -0 | -0 |

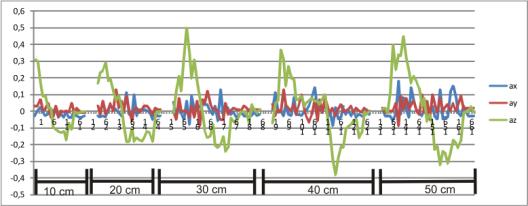


*Gambar 4.21 Grafik pengujian pergerakan modul pada sumbu y negatif*

Data pengujian pergerakan pada sumbu z dengan arah positif ditunjukan pada tabel 4.7 , sedangkan grafik yang terbentuk ditunjukan oleh gambar 4.22.

Tabel 4.7 Data pengujian pergerakan sumbu z pada arah negatif

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 cm | | | 20 cm | | | 30 cm | | | 40 cm | | | 50 cm | | |
| ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az |
| -0 | 0.03 | 0.31 | -0 | 0.03 | 0.17 | 0.05 | 0 | -0 | 0.01 | 0.04 | -0.1 | 0.01 | 0.02 | 0.01 |
| 0 | 0.03 | 0.3 | -0 | -0 | 0.23 | -0 | -0.1 | 0.12 | 0.11 | -0 | 0.04 | -0 | -0 | 0.02 |
| 0.02 | 0.07 | 0.17 | -0 | 0.06 | 0.24 | 0.03 | 0.08 | 0.21 | -0 | 0.01 | 0.06 | -0 | 0 | -0 |
| 0 | 0.01 | 0.09 | -0 | -0 | 0.29 | -0 | 0 | 0.12 | 0 | -0 | 0.37 | -0 | 0 | 0.11 |
| -0 | 0.01 | 0.09 | 0 | 0.01 | 0.18 | -0.1 | 0 | 0.32 | -0 | 0.13 | 0.31 | -0.1 | 0 | 0.4 |
| -0 | 0.05 | 0.02 | 0.03 | 0.05 | 0.19 | 0.06 | 0 | 0.5 | 0.04 | 0.04 | 0.15 | -0 | 0.05 | 0.25 |
| 0.01 | -0 | 0.02 | 0.02 | 0 | 0.12 | -0.1 | -0 | 0.35 | -0 | 0.02 | 0.27 | 0.18 | -0.1 | 0.34 |
| 0 | -0 | -0.1 | 0.01 | 0.13 | 0.06 | 0.09 | 0.03 | 0.2 | -0 | 0 | 0.2 | -0 | 0.09 | 0.33 |
| 0.01 | 0.03 | -0.1 | -0 | 0.02 | 0.09 | 0.01 | -0 | 0.31 | 0.08 | 0.04 | 0.19 | 0.08 | 0.01 | 0.45 |
| -0 | -0 | -0.1 | -0 | 0.06 | 0.03 | 0.01 | 0.06 | 0.18 | -0 | -0 | 0.07 | 0.01 | 0.08 | 0.32 |
| -0 | 0 | -0.1 | 0 | -0 | -0.1 | 0.02 | -0.1 | 0.08 | -0 | 0.03 | 0.07 | -0 | 0.01 | 0.24 |
| -0 | -0 | -0.1 | 0.11 | 0.09 | -0.2 | 0.06 | 0.07 | -0 | -0 | 0.1 | 0.09 | 0.14 | 0.03 | 0.17 |
| -0 | 0 | -0.2 | -0 | 0 | -0.2 | 0.04 | 0.05 | -0.1 | -0 | 0.01 | 0.03 | 0.06 | 0.06 | 0.21 |
| -0 | -0 | -0.1 | -0 | -0.1 | -0.1 | 0.04 | 0.12 | -0.2 | 0 | -0 | 0.05 | 0.01 | 0 | 0.18 |
| -0 | 0.05 | -0.1 | 0.1 | 0.01 | -0.2 | 0 | 0.05 | -0.2 | 0.03 | -0.1 | 0.06 | -0 | -0 | 0.14 |
| 0 | -0 | 0 | -0 | 0.02 | -0.2 | -0 | 0.01 | -0.1 | 0.08 | -0 | 0.08 | -0 | 0.1 | 0.03 |
| -0 | 0.02 | -0 | 0.01 | -0 | -0.2 | 0.03 | 0.02 | -0.2 | 0.14 | 0.07 | 0.06 | 0 | -0 | 0.08 |
| -0 | 0 | -0 | 0 | 0.01 | -0.2 | -0.1 | 0 | -0.2 | 0 | -0.1 | 0.06 | 0.05 | 0.06 | -0.1 |
| -0 | -0 | 0 | 0.01 | 0.03 | -0.2 | 0.13 | -0.1 | -0.2 | 0.03 | 0.07 | 0.04 | -0 | -0.1 | -0 |
| -0 | -0 | 0 | 0 | 0.03 | -0.1 | -0 | -0.1 | -0.2 | -0 | -0 | 0.07 | 0.03 | 0.1 | -0.1 |
|  |  |  | -0 | 0.01 | -0.1 | 0.02 | 0.05 | -0.2 | 0.05 | 0.08 | 0.1 | -0 | 0.05 | -0.2 |
|  |  |  | -0.1 | -0 | -0.2 | 0.02 | -0 | -0.2 | -0 | 0.02 | -0.1 | -0.1 | -0 | -0.2 |
|  |  |  | 0.01 | 0.02 | -0 | -0 | 0 | -0 | -0 | -0 | -0.1 | 0.13 | 0.08 | -0.3 |
|  |  |  | -0 | 0.01 | 0 | -0 | 0.02 | -0.1 | -0.1 | 0.02 | -0.3 | -0.1 | 0 | -0.2 |
|  |  |  | -0 | 0.01 | 0 | -0 | 0.01 | -0.1 | 0.02 | -0 | -0.4 | -0 | 0.02 | -0.3 |
|  |  |  |  |  |  | -0 | 0.01 | -0 | -0.1 | 0.01 | -0.3 | -0 | 0.04 | -0.3 |
|  |  |  |  |  |  | -0 | 0 | 0 | -0 | 0.04 | -0.2 | 0.11 | 0 | -0.3 |
|  |  |  |  |  |  | -0 | 0.03 | 0 | 0.03 | -0 | -0.2 | 0.15 | 0.05 | -0.2 |
|  |  |  |  |  |  | -0 | 0.02 | -0 | -0 | 0 | -0.1 | 0.08 | 0 | -0.2 |
|  |  |  |  |  |  | -0 | 0 | -0.1 | 0 | 0.01 | -0.1 | -0 | 0.03 | -0.2 |
|  |  |  |  |  |  | -0 | 0.02 | 0.04 | -0 | 0.03 | -0.1 | -0 | 0.09 | -0.2 |
|  |  |  |  |  |  | -0 | 0.01 | 0.01 | 0 | -0 | -0.1 | 0.03 | 0.01 | -0 |
|  |  |  |  |  |  | -0 | 0.01 | -0 | -0 | 0.02 | -0.1 | -0 | 0.02 | 0 |
|  |  |  |  |  |  | -0 | 0.01 | 0.01 | -0 | 0 | -0.2 | -0 | 0.02 | 0 |
|  |  |  |  |  |  |  |  |  | -0 | -0 | -0.1 | -0 | 0.01 | 0.03 |
|  |  |  |  |  |  |  |  |  | -0 | 0.02 | 0 | -0 | 0.02 | 0 |
|  |  |  |  |  |  |  |  |  | 0.01 | -0 | -0 |  |  |  |
|  |  |  |  |  |  |  |  |  | -0 | 0 | 0 |  |  |  |

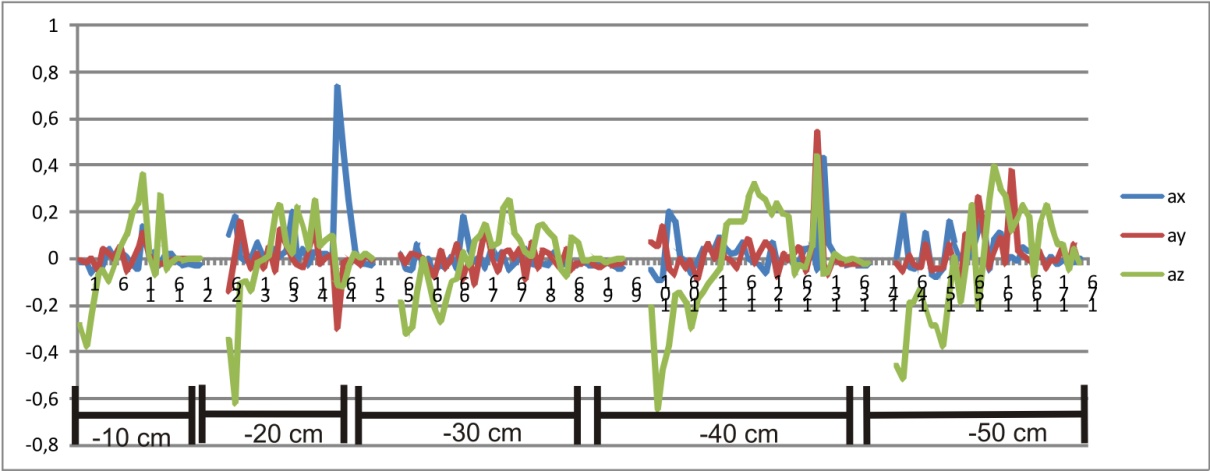


*Gambar 4.22 Grafik pengujian pergerakan modul pada sumbu z positif*

Data pengujian pergerakan pada sumbu z dengan arah negatif ditunjukan pada tabel 4.8 , sedangkan grafik yang terbentuk ditunjukan oleh gambar 4.23.

Tabel 4.8 Data pengujian pergerakan sumbu y pada arah negatif

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -10 cm | | | -20 cm | | | -30 cm | | | -40 cm | | | -50 cm | | |
| ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az | ax | ay | az |
| -0 | -0 | -0.3 | 0.1 | -0.1 | -0.4 | 0.02 | 0.01 | -0.2 | -0.1 | 0.07 | -0.2 | 0.01 | -0 | -0.5 |
| -0 | -0 | -0.4 | 0.18 | 0.02 | -0.6 | -0 | -0 | -0.3 | -0.1 | 0.05 | -0.7 | 0.19 | -0.1 | -0.5 |
| -0.1 | 0 | -0.2 | 0.01 | 0.16 | -0.1 | -0.1 | 0.02 | -0.3 | -0.1 | 0.14 | -0.5 | -0 | 0.01 | -0.2 |
| -0 | -0.1 | -0.1 | -0 | 0.01 | -0.1 | 0.06 | 0.01 | -0.2 | 0.2 | -0 | -0.4 | -0.1 | -0 | -0.2 |
| -0 | 0.04 | -0.1 | 0 | -0 | -0.1 | -0 | 0 | 0 | 0.16 | -0.1 | -0.2 | -0 | -0 | -0.1 |
| 0.04 | 0.02 | -0.1 | 0.07 | 0.02 | -0 | 0 | -0 | -0.1 | 0.01 | 0 | -0.2 | 0.11 | 0.06 | -0.2 |
| 0.01 | 0 | -0 | 0.01 | -0 | -0 | -0.1 | -0.1 | -0.2 | -0 | -0 | -0.2 | -0.1 | -0.1 | -0.3 |
| 0.03 | 0.05 | 0.04 | 0 | 0.05 | 0.01 | -0 | 0.03 | -0.3 | -0.1 | -0 | -0.3 | -0.1 | -0 | -0.3 |
| 0.01 | -0.1 | 0.1 | 0.04 | -0.1 | 0.18 | -0 | -0 | -0.2 | -0 | -0.1 | -0.2 | -0.1 | -0 | -0.4 |
| -0 | -0 | 0.18 | 0.01 | 0.12 | 0.23 | 0.01 | -0 | -0.1 | 0.04 | 0.02 | -0.2 | 0.16 | 0.06 | -0.2 |
| -0.1 | 0.03 | 0.23 | 0.03 | 0.06 | 0.06 | -0 | 0.06 | -0.1 | 0.04 | 0.06 | -0.1 | 0.07 | 0 | 0.01 |
| 0.14 | 0.11 | 0.36 | 0.2 | 0 | 0.02 | 0.18 | -0.1 | 0.03 | 0.04 | 0 | -0.1 | 0 | -0.1 | -0.2 |
| 0.01 | 0.01 | 0.02 | -0 | -0 | 0.22 | 0.04 | -0 | -0 | 0.09 | 0.08 | -0.1 | -0 | 0.1 | -0 |
| 0.03 | -0 | -0.1 | 0.04 | -0 | 0.14 | -0 | -0.1 | 0.07 | 0.04 | 0 | 0.14 | 0.02 | 0.05 | 0.23 |
| -0 | -0 | 0.27 | -0 | 0.02 | 0.04 | 0.01 | 0.03 | 0.1 | 0.02 | -0 | 0.16 | 0.12 | 0.26 | -0.2 |
| 0.02 | -0 | -0.1 | 0.03 | 0.13 | 0.25 | -0 | 0.14 | 0.15 | 0.03 | -0 | 0.16 | 0.18 | 0.14 | -0 |
| 0.02 | -0 | -0 | 0.02 | -0 | 0.05 | 0.05 | 0.01 | 0.05 | 0.07 | 0.04 | 0.16 | -0.1 | -0 | 0.25 |
| -0 | 0 | 0 | 0.02 | 0 | 0.08 | 0 | -0.1 | 0.06 | 0 | 0.08 | 0.26 | 0.08 | 0.02 | 0.4 |
| -0 | 0 | -0 | 0 | 0.01 | 0.1 | 0.03 | 0.02 | 0.21 | -0 | -0 | 0.32 | 0.11 | 0.09 | 0.3 |
| -0 | 0 | 0 | 0.74 | -0.3 | -0.1 | -0.1 | 0.03 | 0.25 | -0 | 0.02 | 0.27 | -0 | -0 | 0.27 |
| -0 | 0 | 0 | 0.45 | -0 | -0.1 | -0 | 0 | 0.11 | -0.1 | 0.07 | 0.25 | 0.01 | 0.37 | 0.12 |
| -0 | 0 | 0 | 0.25 | -0 | -0 | 0 | 0.04 | 0.08 | 0.07 | 0.03 | 0.19 | -0 | 0.03 | 0.17 |
|  |  |  | 0.04 | -0 | 0.02 | 0.04 | -0.1 | 0.03 | -0 | -0.1 | 0.24 | 0.05 | 0.01 | 0.23 |
|  |  |  | -0 | -0 | 0 | -0 | 0.07 | 0.01 | 0 | 0.02 | 0.19 | 0.03 | -0 | 0.18 |
|  |  |  | -0 | 0.01 | 0.02 | -0 | -0 | 0.14 | 0.01 | -0 | 0.18 | 0.02 | -0 | -0.1 |
|  |  |  | -0 | -0 | 0 | -0 | 0.03 | 0.15 | -0 | 0.02 | -0.1 | 0.02 | 0.03 | 0.15 |
|  |  |  |  |  |  | -0 | 0.02 | 0.11 | -0 | 0.05 | -0 | -0 | -0 | 0.23 |
|  |  |  |  |  |  | 0.03 | -0 | 0.09 | 0.04 | -0.1 | -0 | 0.01 | -0 | 0.14 |
|  |  |  |  |  |  | -0 | -0 | -0 | 0.05 | 0.06 | 0.05 | -0 | -0 | 0.06 |
|  |  |  |  |  |  | -0 | 0.04 | -0.1 | -0.1 | 0.54 | 0.44 | -0 | 0.04 | 0.06 |
|  |  |  |  |  |  | -0 | -0.1 | 0.09 | 0.43 | 0 | -0.1 | 0 | -0 | -0.1 |
|  |  |  |  |  |  | -0 | -0 | 0.07 | 0.07 | -0.1 | -0.1 | -0 | 0.06 | 0.04 |
|  |  |  |  |  |  | -0 | -0 | -0 | 0.02 | 0 | 0.02 | 0 | -0 | -0 |
|  |  |  |  |  |  | -0 | -0 | -0 | -0 | -0 | 0 |  |  |  |
|  |  |  |  |  |  | -0 | -0 | 0 | -0 | -0 | -0 |  |  |  |
|  |  |  |  |  |  | -0 | -0 | 0 | -0 | -0 | 0 |  |  |  |
|  |  |  |  |  |  | -0 | -0 | -0 | -0 | -0 | -0 |  |  |  |
|  |  |  |  |  |  | -0 | -0 | -0 | -0 | -0 | -0 |  |  |  |
|  |  |  |  |  |  | -0 | -0 | 0 | -0 | -0 | -0 |  |  |  |
|  |  |  |  |  |  | -0 | -0 | 0 |  |  |  |  |  |  |



*Gambar 4.23 Grafik pengujian pergerakan modul pada sumbu z negatif*

1. **Analisa Hasil Pengujian Percepatan Dinamik**
2. **Pergeseran ke arah negatif dan arah positif**

Dapat diamati berdasarkan grafik yang diperoleh dari pengujian percepatan dinamik pada tiga sumbu diperoleh perbedaan bentuk grafik pada pergeseran pada arah positif dan arah negatif adalah sebagai berikut (gambar 4.24).

*Gambar 4.24 Perbedaan arah pergerakan benda pada arah positif (kiri) dan arah negatif (kanan)*

Gambar 4.24 merupakan contoh grafik yang terbentuk dari data percepatan pada perpindahan data sumbu y sejauh 40 cm pada arah positif dan 40 cm arah negatif. Pada pergerakan kearah positif, grafik perceptan yang terbentuk sepertihalnya gelombang sinus sedangkan pada arah pergerakan negatif grafik yang terbentuk sepertihalnya gelombang sinus yang mengalami pergeseran fasa 1800.

Dari grafik tersebut apabila di integrasikan akan didapat nilai kecepatan yang berbeda, yakni untuk pergerakan positif nilai kecepatan yang didapat bernilai positif sedangkan untuk pergerakan kearah negatif kecepatan yang didapat bernilai negatif. Integrasi data percepatan menjadi data kecepatan digunakan persamaan 2.4 yaitu

Dengan percepatan ke-n adalah *an* dan kecepatan ke-n adalah *vn*sehinggapersamaan 2.4 diatas menjadi.

Sedangkan nilai *∆t* adalah 8 ms sesuai dengan interval pengambilan data tiap sumbu sedangkan nilai *v0* adalah 0. Grafik pada gambar 4.24 diatas jika diintegrasikan menjadi seperti ditunjukan oleh gambar 4.25.

*Gambar 4.25 Hasil integrasi data percepatan pada pergerakan benda pada arah positif (kiri) dan arah negatif (kanan)*

Integrasi kedua akan menghasilkan data jarak perpindahan, parameter yang di integrasi adalah data percepatan sehingga persamaan yang digunakan adalah sebagai berikut.

Untuk perhitungan ini nilai *∆t* masih sama dengan yang sebelumnya yaitu 8 ms sedangkan inisialisasi jarak perpindahan pada sample ke-0 adalah sejauh 0 m sehingga s0=0. Hasil integrasi ditunjukan oleh gambar 4.26 dengan total jarak perpindahan yang terjadi pada benda adalah pada sampel terakhir.

*Gambar 4.26 Hasil integrasi data kecepatan pada pergerakan benda pada arah positif (kiri) dan arah negatif (kanan)*

Dari gambar diatas dapat dilihat bahwa perpindahan benda hasil perhitungan mendekati perpindahan benda sesungguhnya serta dapat disimpulkan bahwa sensor mampu mendeteksi pergerakan linear pada arah positif maupun negatif dengan melihat nilai jarak perpindahan yang didapat yakni pada pergerakan kearah positif jarak perpindahan yang didapat bernilai positif sedangkan pada pergerakan kearah negatif jarak perpindahan yang didapat bernilai negatif.

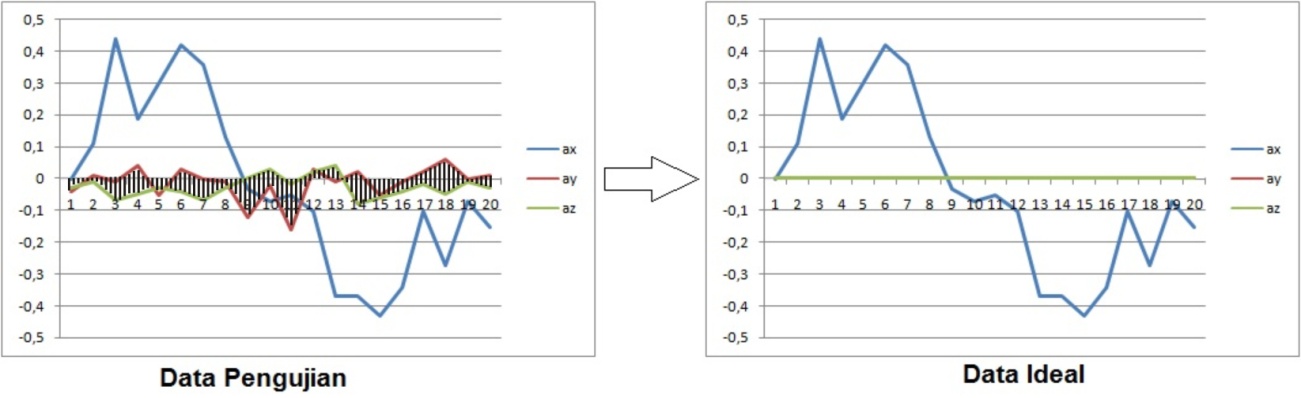
1. **Pengaruh Pergerakan Satu Sumbu Terhadap Sumbu Yang Lain**

Idealnya, jika akselerometer digerakan hanya pada satu sumbu secara linear maka nilai percepatan pada sumbu yang lain bernilai tetap karena tidak mengalami perubahan posisi. Misalnya jika sensor digerakan pada sumbu x kearah positif, maka nilai percepatan yang berubah seharusnya hanya percepatan pada sumbu x saja sedangkan nilai percepatan pada sumbu y dan z harusnya tetap.

Akan tetapi pada pengujian ini ternyata memberikan hasil yang berbeda karena meskipun digerakan pada satu sumbu tetapi nilai percepatan sumbu yang lain ternyata ikut berubah. Hal ini terjadi karena beberapa faktor diantaranya:

1. Efek vibrasi pada saat modul bergerak.
2. Gaya gesek media yang digunakan pada pengujian.
3. Efek dari *Micro Electro-Mechanical System* (MEMS) yang diadopsi oleh akselerometer H48C.
4. Pengaruh *noise* internal akselerometer.

Gambar 4.27 menunjukan data percepatan yang didapat pada pengujian dan perbandinganya dengan data ideal yang seharusnya didapat. Daerah yang diarsir adalah nilai percepatan pada sumbu y dan z yang ikut berubah ketika modul digerakan pada sumbu x saja.



*Gambar 4.27 Perbandingan hasil pengujian dengan data ideal, meskipun digerakan hanya pada sumbu x akan tetapi nilai percepatan pada sumbu y dan sumbu z juga ikut berubah.*

Efek vibrasi dan pengaruh gaya gesek media pengujian merupakan kesalahan yang terjadi pada saat proses pengujian. Kesalahan ini merupakan faktor eksternal termasuk didalamnya adalah *human error*. Untuk meminimalisasi kesalahan ini diperlukan media pengujian yang benar – benar bagus diantaranya adalah media bersifat licin sehingga meminimalisasi gaya gesek antara media yang diam dengan media yang bergerak. Selain itu pada proses menggerakan benda juga diusahakan benda benar – benar terbebas dari dorongan maupun tekanan pada sumbu yang lain.

Efek dari *Micro Electro-Mechanical System* (MEMS) dapat dianalogikan dengan sistem pegas pada gambar 4.28. Sebuah kubus dengan masa M terhubung melalui dua buah pegas pada kedua sumbunya terhadap dinding sebuah balok. Pada kondisi diam, panjang regangan pegas pada masing – masing sumbu adalah *x* dan *y*. Sedangkan apabila balok diberi gaya dorong kesalah satu sumbu, misalnya sumbu *x* maka pada kubus M berlaku hukum *Newton I* yaitu ‘*setiap benda memiliki sifat kelembaman’* yang mengakibatkan terjadinya perubahan regangan pada pegas *x* sejauh *dx*, akan tetapi pada pegas *y* juga mengalami perubahan panjang sejauh *dy* karena terjadi simpangan pada kubus M. Fenomena ini sama halnya jika sistem dalam representasi 3 dimensi. Sedangkan dalam teknik pengukuran inersial, fenomena ini disebut *cross axis efect.*



*Gambar 4.28 Analogi Micro Electro-Mechanical System yang diadopsi oleh akselerometer H48C*

1. **Tingkat Akurasi Pembacaan Akselerometer**

Tabel 4.8 menunjukan data hasil pengujian akselerometer yang digerakan pada sumbu x, y, dan z berikut selisih pembacaan dengan data sebenarnya.

Dari hasil yang ditunjukan pada tabel 4.8 menunjukan bahwa tingkat akurasi pembacaan akselerometer H48C tergolong cukup tinggi karena untuk keseluruhan pengujian rata – rata error dibawah 1.5 cm. Akan tetapi data satu sumbu diatas sebenarnya didalamnya masih terdapat noise yang disebutkan pada poin 2 diatas yang seharusnya tidak diikut sertakan dalam perhitungan. Hal ini dapat dibuktikan dengan melakukan perhitungan yang sama untuk sumbu yang lain yang seharusnya nilainya mendekati 0. Hasil perhitungan ditunjukan oleh tabel 4.9 dibawah.

Tabel 4.8 Data hasil pengujian pergerakan akselerometer pada sumbu x, y, dan z

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Jarak (cm) | X | | Y | | Z | |
| Terukur | *Error* | Terukur | *Error* | Terukur | *Error* |
| 10 | 7.7 | 2.3 | 9.2 | 0.8 | 9.4 | 0.6 |
| 20 | 21.4 | 1.4 | 21.2 | 1.2 | 19.5 | 0.5 |
| 30 | 28.5 | 1.5 | 30.7 | 0.7 | 29.8 | 0.2 |
| 40 | 41.8 | 1.8 | 39.7 | 0.3 | 40.6 | 0.6 |
| 50 | 51.5 | 1.5 | 51.7 | 1.7 | 52.9 | 2.9 |
| -10 | -8 | 2 | -10.7 | 0.7 | -9.5 | 0.5 |
| -20 | -19.6 | 0.4 | -20.1 | 0.1 | -22.3 | 2.3 |
| -30 | -30.2 | 0.2 | -32.3 | 2.3 | -31.8 | 1.8 |
| -40 | -39.4 | 0.6 | -40.4 | 0.4 | -38.1 | 1.9 |
| -50 | -51.1 | 1.1 | -53.9 | 3.9 | -53.4 | 3.4 |
| Error rata - rata | | 1.28 |  | 1.21 |  | 1.47 |

Tabel 4.9 Hasil *double* integrasi data percepatan pada sumbu yang tidak bergerak.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Jarak (cm) | Pergeseran sumbu x | | Pergeseran sumbu y | | Pergeseran sumbu z | |
| y | z | x | z | x | y |
| 10 | 0.3 | 4.6 | -1.6 | -5.2 | -2.6 | 2.3 |
| 20 | 2.7 | 6.4 | -1.9 | -4.1 | 5.7 | 5.8 |
| 30 | 0.4 | 12.1 | 0 | 2.3 | 3.8 | 1.7 |
| 40 | 4.4 | 12.3 | -2.5 | -15.5 | 6 | 11.7 |
| 50 | -4.5 | -14.7 | 9 | -9 | 0 | 1.9 |
| -10 | -1 | -2 | 0 | -2.5 | -1.6 | -0.2 |
| -20 | 4.8 | -4.1 | 2.4 | 0.4 | -4.7 | 2.5 |
| -30 | -1.9 | 1.7 | -1.8 | -5.2 | -0.9 | -2.4 |
| -40 | -7.3 | 1.1 | 0.4 | -7.3 | -1.4 | -2.8 |
| -50 | 1 | -4.1 | 1.4 | -11 | 13 | 8.7 |

Bentuk grafik dari data percepatan, kecepatan dan jarak perpindahan dari hasil pengujian pergerakan satu sumbu pada akselerometer H48C dapat dilihat pada lampiran 2.

### Reduksi Error Pada Pengukuran Dan Perhitungan

Jika hanya menggunakan metode pengukuran dan perhitungan diatas, dapat diamati bahwa tingkat kesalahan dari pengukuran dan perhitungan cukup tinggi jika dibandingkan dengan nilai sesungguhnya.

Untuk meminimalisir tingkat kesalahan dilakukan perbaikan proses pengolahan sinyal lebih lanjut. Dimaulai dari sinyal yang didapat dari sensor akselerometer.

Sinyal dari akselerometer terdapat banyak derau untuk itu diperlukan filter. Terdapat banyak macam filter digital yang dapat diaplikasikan untuk sinyal dari akselerometer ini. Pemilihan filter didasarkan pada sinyal yang didapat.

Data percepatan dari akselerometer merupakan sinyal yang informasinya direpresentasikan dalam runtun waktu diskret. Sehingga dipilih filter IIR (*Infinite Impulse Response*) dan *prototype* filter analog yang dipilih adalah LPF *Bessel Filter*. Fungsi rekursi untuk IIR ditunjukan oleh persamaan 4.14.

.......................(4.14)

Langkah selanjutnya adalah menentukan parameter – parameter fiter yaitu frekuensi sampling, frekuensi *cutt off*, dan orde filter.

Pengambilan data akselerometer dalam 1 detik sebanyak 125 data sehingga frekuensi sampling adalah 125 Hz, sedangkan frekuensi *cutt off* dibuat 5 Hz.

Respon frekuensi dari filter digital adalah pada [0,1] sedangkan 1 merepresentasikan π yang merupakan frekuensi *Nyquist* sehingga π = 62.5 Hz. Frekuensi *cutt off* dari filter digital adalah ωc = *fc / fs* = 0.04 rad/s.

Untuk mencari koefisien rekursi dari fungsi alih filter yang akan dibuat pada pengujian ini dilakukan dengan menggunakan program *Filter* *helper* yang tersedia pada website <http://www-users.cs.york.ac.uk/~fisher/mkfilter> dengan memasukan parameter frekuensi sampling, frekuensi *cut off* (*corner frequency*) dan orde filter.

* + 1. **Penentuan Orde Filter**

Untuk menentukan orde filter yang optimal dilakukan pengujian terhadap data yang didapat dari pembacaan akselerometer. Tabel 4.10 menunjukan koefisien rekursi untuk tiap orde dari orde 1 hingga orde 10 pada filter Bessel dengan frekuensi *sampling* 125 Hz dan frekuensi *cut off* 5 Hz.

Tabel 4.10 Koefisien rekursi filter besel orde 1 hingga 10 dengan frekuensi *cut off* 5 Hz dan frekuensi *sampling* 125 Hz

|  |  |  |  |
| --- | --- | --- | --- |
| Orde | gain | Koefisien Rekursi | |
| 1 | 8.916 | 1  1 | 0.776 |
| 2 | 5.051e+01 | 1  2  1 | -0.573  1.494 |
| 3 | 2.711e+02 | 1  3  3  1 | 0.423  -1.655  2.203 |
| 4 | 1.331e+03 | 1  4  6  4  1 | -0.304  1.596  -3.191  2.887 |
| 5 | 5.908e+03 | 1  5  10  10  5  1 | 0.212  -1.399  3.753  -5.110  3.539 |
| 6 | 2.396e+04 | y[n] =(x[n-6])+(6 \* x[n-5])+(15 \* x[n-4])+ (20 \* x[n-3])+(15 \* x[n-2])+(6 \* x[n-1])+(x[n-0])+(-0.1422802283 \* y[n-6]) +(1.1395670964 \* y[n-5])+ (-3.8512763692 \* y[n-4])+ (7.0384217426 \* y[n-3])+ (-7.3466431593 \* y[n-2])+ (4.1595397650 \* y[n-1]) |  |
| 7 | 9.019e+04 | y[n] =(x[n-7])+(7 \* x[n-6])+(21 \* x[n-5])+ (35 \* x[n-4])+(35 \* x[n-3])+ (21 \* x[n-2])+(7 \* x[n-1])+ (x[n-0])+(0.0926657569 \* y[n-7])+ (-0.8744780076 \* y[n-6])+ (3.5784963076 \* y[n-5])+ (-8.2397847975 \* y[n-4])+ (11.5430252792 \* y[n-3])+ (-9.8512532460 \* y[n-2])+ (4.7499094369 \* y[n-1]) |  |
| 8 | 3.194e+05 | y[n] =(x[n-8])+(8 \* x[n-7])+(28 \* x[n-6])+ (56 \* x[n-5])+(70 \* x[n-4])+(56 \* x[n-3])+(28 \* x[n-2])+(8 \* x[n-1])+ (x[n-0])+(-0.0586383381 \* y[n-8])+ (0.6386514185 \* y[n-7])+ (-3.0770659006 \* y[n-6])+ (8.5735247895 \* y[n-5])+ (-15.1242655856 \* y[n-4])+ (17.3167118101 \* y[n-3])+ (-12.5831540162 \* y[n-2])+ (5.3134342636 \* y[n-1]) |  |
| 9 | 1.074367989e+06 | y[n] =(x[n-9])+(9 \* x[n-8])+(36 \* x[n-7])+ (84 \* x[n-6])+(126 \* x[n-5])+(126 \* x[n-4])+(84 \* x[n-3])+(36 \* x[n-2])+ (9 \* x[n-1])+(x[n-0])+(0.0361333089 \* y[n-9])+(-0.4469788612 \* y[n-8])+ (2.4835321686 \* y[n-7])+ (-8.1411261514 \* y[n-6])+ (17.3659057828 \* y[n-5])+ (-25.0218054775 \* y[n-4])+ (24.3792419665 \* y[n-3])+ (-15.5077379518 \* y[n-2])+ (5.8523586558 \* y[n-1]) |  |
| 10 | 3.457e+06 | y[n] =(x[n-10])+(10 \* x[n-9])+(45 \* x[n-8])+ (120 \* x[n-7])+(210 \* x[n-6])+(252 \* x[n-5])+(210 \* x[n-4])+(120 \*x[n-3])+ (45 \* x[n-2])+(10 \*x[n-1])+(x[n-0])+ (-0.0217192171 \* y[n-10])+ (0.3012846437 \* y[n-9])+ (-1.8998620952 \* y[n-8])+ (7.1765621146 \* y[n-7])+ (-17.9969646061 \* y[n-6])+ (31.3334899657 \* y[n-5])+ (-38.3927239217 \* y[n-4])+ (32.7257173947 \* y[n-3])+ (-18.5945171308 \* y[n-2])+ (6.3684366602 \* y[n-1]) |  |

Setelah mendapatkan persamaan rekursi dan gain dari tiap – tiap orde, langkah selanjutnya adalah melakukan pengambilan data.

Data yang diambil adalah data pergerakan pada sumbu x sejauh -30 cm. Data yang diambil sebanyak 10 kali kemudian dari sekumpulan data percobaan dilakukan integrasi kedua untuk memperoleh data perpindahan. Dari data perpindahan dapat diambil nilai error dalam rMSE (root Mean Square Error).

Orde filter yang dipilih adalah orde filter yang memberikan nilai rMSE paling kecil. Hasil pengujian filter tiap orde ditunjukan oleh tabel 4.11 dibawah. Sedangkan grafik data percepatan sebelum dan sesudah menggunakan filter dari orde 1 hingga 10 ditunjukan oleh gambar 4.29.

Tabel 4.11 Nilai rMSE pada pengujian filter dari orde 1 hingga 10 dan data yang tidak difilter.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Filter | Percobaan ke - | | | | | | | | | | rMSE |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Tidak difilter | -0.255 | -0.386 | -0.276 | -0.239 | -0.309 | -0.368 | -0.340 | -0.337 | -0.348 | -0.355 | 0.0474 |
| Orde 1 | -0.256 | -0.381 | -0.279 | -0.243 | -0.309 | -0.364 | -0.339 | -0.335 | -0.347 | -0.352 | 0.0450 |
| Orde 2 | -0.257 | -0.379 | -0.280 | -0.244 | -0.309 | -0.363 | -0.339 | -0.335 | -0.346 | -0.351 | 0.0441 |
| Orde 3 | -0.257 | -0.377 | -0.281 | -0.246 | -0.310 | -0.362 | -0.339 | -0.334 | -0.345 | -0.350 | 0.0432 |
| Orde 4 | -0.258 | -0.375 | -0.283 | -0.248 | -0.310 | -0.361 | -0.339 | -0.333 | -0.345 | -0.349 | 0.0423 |
| Orde 5 | -0.259 | -0.374 | -0.284 | -0.249 | -0.310 | -0.360 | -0.340 | -0.333 | -0.344 | -0.349 | 0.0416 |
| Orde 6 | -0.260 | -0.372 | -0.285 | -0.251 | -0.310 | -0.359 | -0.340 | -0.332 | -0.344 | -0.348 | 0.0409 |
| Orde 7 | -0.261 | -0.371 | -0.286 | -0.252 | -0.310 | -0.358 | -0.340 | -0.332 | -0.344 | -0.347 | 0.0404 |
| Orde 8 | -0.262 | -0.370 | -0.286 | -0.253 | -0.310 | -0.357 | -0.340 | -0.332 | -0.344 | -0.347 | 0.0398 |
| Orde 9 | -0.262 | -0.369 | -0.287 | -0.254 | -0.310 | -0.356 | -0.340 | -0.332 | -0.343 | -0.346 | 0.0393 |
| Orde 10 | -0.263 | -0.369 | -0.288 | -0.255 | -0.310 | -0.356 | -0.340 | -0.331 | -0.343 | -0.346 | 0.0389 |

1. Tidak di filter
2. Orde 1
3. Orde 2
4. Orde 3
5. Orde 4
6. Orde 5
7. Orde 6
8. Orde 7
9. Orde 8
10. Orde 9
11. Orde 10

*Gambar 4.29 Grafik nilai percepatan sebelum dan sesudah menggunakan filter orde 1 hingga 10*

Grafik nilai rMSE ditunjukan oleh gambar 4.30 dibawah. Tampak bahwa filter orde 10 memberikan nilai rMSE yang paling kecil.

*Gambar 4.30 Grafik nilai rMSE hasil integrasi kedua (data perpindahan) pada tiap – tiap orde filter dan data yang tidak menggunakan filter.*

Selain itu dapat diamati juga *cross axis effect* dari setiap percobaan. Tabel 4.12 menunjukan nilai rata – rata *cross axis effect* pada pengujian tidak menggunakan filter dan menggunakan filter orde 1 hingga 10 sedangkan grafiknya ditunjukan oleh gambar 4.31.

Dari hasil yang didapat tampak bahwa filter orde 10 memberikan nilai rata – rata pergerakan pada sumbu Y dan sumbu Z paling kecil. Sehingga dapat disimpulkan bahwa *cross axis effect* pada penggunaan filter orde 10 paling kecil.

Berdasarkan nilai rMSE dan *cross axis effect* maka dipilih filter dengan orde 10.

Tabel 4.12 *Cross Axis Effect* pada pengujian tidak menggunakan filter dan menggunakan filter orde 1 hingga 10.

|  |  |  |
| --- | --- | --- |
| **Filter** | **Rata - rata Y** | **Rata - rata Z** |
| Tidak difilter | 0.0265 | 0.0280 |
| Orde 1 | 0.0243 | 0.0266 |
| Orde 2 | 0.0235 | 0.0261 |
| Orde 3 | 0.0226 | 0.0256 |
| Orde 4 | 0.0218 | 0.0251 |
| Orde 5 | 0.0211 | 0.0247 |
| Orde 6 | 0.0205 | 0.0244 |
| Orde 7 | 0.0200 | 0.0240 |
| Orde 8 | 0.0195 | 0.0237 |
| Orde 9 | 0.0191 | 0.0234 |
| Orde 10 | 0.0186 | 0.0232 |

*Gambar 4.31 Grafik cross axis effect pada pengujian tidak menggunakan filter dan menggunakan filter orde 1 hingga 10*

* + 1. **Penerapan Pada Algoritma Program**

Pada tabel 4.10 ditunjukan persamaan rekursi dari fungsi alih filter yang dihasilkan dengan x[n] adalah sebagai input dan y[n] sebagai output. Langkah selanjutnya adalah menerapkanya kedalam program arduino.

Berikut fungsi filter IIR pada arduino yang diigunakan.

// LPF Bessel, fc = 5 Hz Fs = 125 Hz Orde=10

#define GAIN 3.457215320e+06

**float** IIR(**float** value, **float** xv[], **float** yv[]) {

xv[0] = xv[1];

xv[1] = xv[2];

xv[2] = xv[3];

xv[3] = xv[4];

xv[4] = xv[5];

xv[5] = xv[6];

xv[6] = xv[7];

xv[7] = xv[8];

xv[8] = xv[9];

xv[9] = xv[10];

xv[10] = value / GAIN;

yv[1] = yv[2];

yv[2] = yv[3];

yv[3] = yv[4];

yv[4] = yv[5];

yv[5] = yv[6];

yv[6] = yv[7];

yv[7] = yv[8];

yv[8] = yv[9];

yv[9] = yv[10];

yv[10] = (xv[0]+xv[10]) + 10\*(xv[1]+xv[9]) + 45\*(xv[2]+xv[8]) + 120\*(xv[3]+xv[7]) + 210\*(xv[4]+xv[6]) + 252\*xv[5] + (-0.0217192171\*yv[0]) + (0.3012846437\*yv[1]) + (-1.8998620952\*yv[2]) + (7.1765621146\*yv[3]) + (-17.9969646061\*yv[4]) + (31.3334899657\*yv[5]) + (-38.3927239217\*yv[6]) + (32.7257173947\*yv[7]) + (-18.5945171308\*yv[8]) + (6.3684366602\*yv[9]);

**return** yv[10];

}

Untuk memanggil fungsi IIR diatas dilakukan dengan cara sebagai berikut.

//============ Filtering data akselerometer==========//

**float** accelXIn[11], accelXIn[11], accelXIn[11];

**float** accelXOut[11], accelXOut[11], accelXOut[11];

**float** accelX = IIR(Ax, accelXIn, accelXOut);

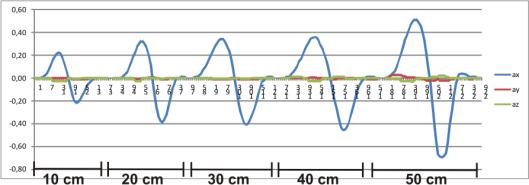
**float** accelY = IIR(Ay, accelYIn, accelYOut);

**float** accelZ = IIR(Az, accelZIn, accelZOut);

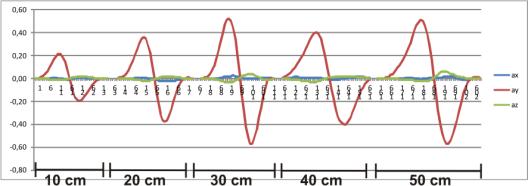
* + 1. **Hasil Pengukuran Setelah Menggunakan Filter**

Pengukuran kembali dilakukan dengan metode mengukur percepatan pada masing-masing sumbu dengan arah pergerakan positif dan negatif untuk mengetahui efek penggunaan filter pada modul.

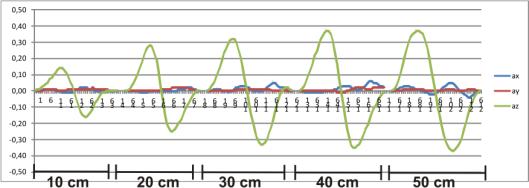
Hasil yang didapat pada pengukuran sumbu X, Y, dan Z pada arah positif maupun negatif ditunjukan oleh gambar 4.32 berikut.



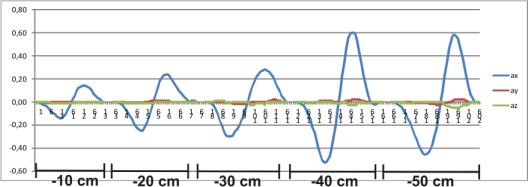
(a)



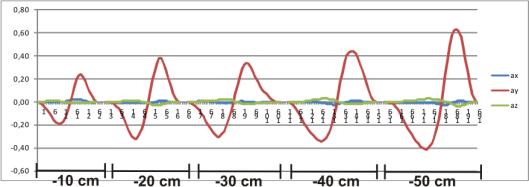
(b)



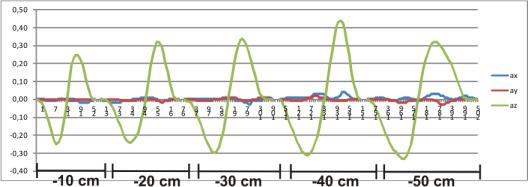
(c)



(d)



(e)



(f)

*Gambar 4.32 Hasil pengujian akselerometer setelah difilter (a) Sumbu x arah positif, (b) Sumbu x arah positif, (c) Sumbu z arah positif, (d)Sumbu x arah negatif,(e) Sumbu y arah negatif,(f) Sumbu z arah negatif*

Berdasarkan grafik yang ditunjukan oleh gambar 4.32 diatas tampak bahwa hasil yang didapatkan setelah menerapkan filter pada akusisi data akselerometer menunjukan kurva pergerakan lebih halus dan lebih baik. Tampak bahwa noise pada tiap – tiap sumbu dapat dikurangi. Hal ini menunjukan bahwa filter bekerja sesuai dengan yang diharapkan.

Langkah selanjutnya adalah melakukan double integral untuk mendapatkan data jarak perpindahan. Untuk hasil masing – masing pergerakan sumbu, hasil double integral ditunjukan oleh tabel 4.13 dibawah.

Tabel 4.13 Data hasil pengujian pergerakan akselerometer pada sumbu x, y, dan z setelah difilter

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Jarak (cm) | X | | Y | | Z | |
| Terukur | *Error* | Terukur | *Error* | Terukur | *Error* |
| 10 | 11.02 | 1.02 | 11.24 | 1.24 | 11.75 | 1.75 |
| 20 | 20.03 | 0.03 | 20.94 | 0.94 | 20.79 | 0.79 |
| 30 | 33.75 | 3.75 | 31.99 | 1.99 | 32.29 | 2.29 |
| 40 | 45.08 | 5.08 | 43.09 | 3.09 | 42.71 | 2.71 |
| 50 | 54.10 | 4.10 | 51.95 | 1.95 | 53.18 | 3.18 |
| -10 | -9.61 | 0.39 | -10.43 | 0.43 | -11.45 | 1.45 |
| -20 | -19.70 | 0.30 | -21.81 | 1.81 | -21.80 | 1.80 |
| -30 | -31.79 | 1.79 | -30.41 | 0.41 | -29.56 | 0.44 |
| -40 | -43.04 | 3.04 | -42.74 | 2.74 | -39.92 | 0.08 |
| -50 | -50.10 | 0.10 | -54.22 | 4.22 | -53.10 | 3.10 |
| Error rata - rata | | 1.96 |  | 1.88 |  | 1.76 |

Sedangkan pengaruh pergerakan terhadap sumbu yang tidak digerakan ditunjukan oleh tabel 4.14 berikut.

Tabel 4.14 Hasil *double* integrasi data percepatan pada sumbu yang tidak bergerak setelah difilter

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Jarak (cm) | Pergeseran Sumbu X | | Pergeseran Sumbu Y | | Pergeseran Sumbu Z | |
| Y | Z | X | Z | X | Y |
| 10 | -1.19 | -5.47 | 0.45 | -0.16 | 1.90 | 3.63 |
| 20 | -0.55 | -2.17 | -0.87 | -1.00 | -0.33 | 1.83 |
| 30 | -4.81 | -4.69 | 2.67 | -1.63 | 6.17 | 2.16 |
| 40 | -0.21 | -3.45 | 4.04 | -3.50 | 3.94 | 1.39 |
| 50 | 3.22 | -6.91 | 2.52 | 0.62 | 5.34 | 2.74 |
| -10 | -0.53 | -1.45 | 0.85 | 0.15 | -1.34 | -0.31 |
| -20 | -0.68 | -2.64 | -2.61 | -0.53 | -3.52 | -1.69 |
| -30 | -2.64 | -1.03 | -0.47 | 1.84 | -1.55 | -0.97 |
| -40 | 1.07 | -2.79 | -2.35 | 2.69 | 5.64 | -0.33 |
| -50 | -3.19 | -4.78 | -5.65 | 3.01 | 3.03 | -5.66 |

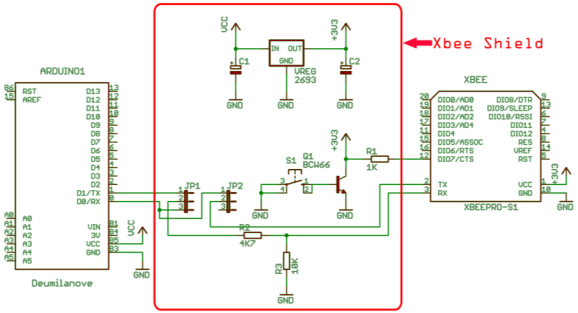
Jika dibandingkan dengan data perhitungan yang belum menggunakan filter, data hasil perhitungan yang menggunakan filter jauh lebih baik walaupun pengaruh gravitasi dan *cross axis efect* belum bisa dihilangkan.

## Unit Komunikasi (modul RF)

Unit komunikasi merupakan bagian yang bertugas untuk mengirim dan menerima data dari tiap *node*. Dalam pembuatan jaringan sensor nirkabel ini terdapat 3 buah *node* yang terdiri dari 2 buah *router* dan 1 buah *gateway.* Antara *router*dan *gateway* memiliki perbedaan skematik dalam hal jalur komunikasi dengan mikrokontroler serta perbedaan algoritma yang digunakan. Sehingga dalam unit komunikasi dibagi menjadi dua buah yaitu *router* dan *gateway.*

### Skematik rangkaian modul RF pada router

Skematik rangkaian perantaraan modul RF (Xbee Pro Series 1) dengan mikrokontroler (Arduino Deumilanove) ditunjukan oleh gambar 4.28.



*Gambar 4.28 Skematik perantaraan Xbee* Pro *Series 1 dengan Arduino Deumilanove pada router*

Pada router digunakan *Xbee Shield* (kotak merah) sebagai perantara antara arduino dengan modul Xbee. Penggunaan *Xbee Shield* memberikan keuntungan sebagai berikut.

1. Instalasi xbee dengan arduino dipermudah karena tinggal menancapkan modul pada soket yang tersedia.
2. Tersedia regulator tegangan 3,3 Volt yang sesuai dengan tegangan suplai yang dibutuhkan modul Xbee sehingga tidak perlu membuat penyesuai tegangan lagi.
3. Dapat digunakan sebagai perantara untuk pemrograman Xbee dengan cara melepas mikrokontroler yang ada pada modul arduino.
4. Terdapat penyesuai level tegangan antara pin TX arduino dengan pin RX Xbee.

Bentuk *Xbee* Shield beserta Xbee Pro Series 1 yang telah terpasang pada *Xbee Shield* ditunjukan oleh gambar 4.30.



*Gambar 4.30. Xbee Shield yang telah terpasang modul Xbee Pro Series 1.*

Pada *router* , jalur komunikasi antara modul Xbee dengan arduino sama dengan jalur komunikasi yang digunakan untuk komunikasi serial arduino dengan komputer melalui perantara konverter *USB to Serial* menggunakan FT232. Sehingga apabila *Xbee Shield* terpasang pada arduino dan akan melakukan proses *download* program dari komputer ke arduino harus mengubah posisi sambungan pada jumper JP2 (gambar 4.8) yakni pin nomor 2 dihubungkan dengan pin no 3. Sehingga hubungan pin RX dari arduino dengan pin TX Xbee terputus dan program dapat ter*-download* dengan sempurna. Jika tidak dilakukan pemindahan posisi jumper JP2 maka akan menyebabkan *error* pada proses penulisan program dari komputer ke arduino.

### Algoritma Pengiriman Data Pada Router

Pembuatan algoritma program pengiriman data pada *router* menggunakan bantuan *library* untuk xbee (file “Xbee.h”) yang ditulis oleh Andrew Rapp dari <http://fsf.org/> (2009). Sebelum membuat program terlebih dahulu dilakukan seting parameter pada modul Xbee sebagai berikut.

|  |  |
| --- | --- |
| Parameter | Nilai |
| BD (*Baud rate*) | 9600 |
| DL (*Destination Address Low*) | 7E63 |
| AP (*API* *mode*) | 2 |

Setelah parameter di seting sesuai tabel diatas, langkah selanjutnya adalah penyusunan algoritma program. Pembuatan algoritma *router* mengacu pada *library* yang digunakan serta penyesuaian dengan karakteristik modul Xbee. Langkah – langkah pembuatan algoritma *router* adalah sebagai berikut.

1. Memasukan *library* Xbee.h kedalam program.
2. Membuat variabel bertipe Xbee.
3. Membuat variabel *buffer* untuk menampung data yang akan dikirim. Variabel ini bertipe integer positif 8 bit (1 *byte*) dan berbentuk *array* dengan besarnya *array* tergantung dari banyaknya data yang akan dikirim.
4. Membuat variabel yang bertipe *XbeeAddress64* dan nilainya diisi dengan *XbeeAdress64(DH,DL)* dimana DH merupakan 32 bit MSB dari alamat Xbee yang dituju dan DL merupakan 32 bit LSB dari alamat Xbee yang dituju.
5. Membuat variabel bertipe *Tx16Request* untuk menampung data keseluruhan yang yang akan dikirim yang mana variabel ini akan diisi dengan hasil dari fungsi *Tx16Request(addr16,payload, sizeof(payload)). Addr16* merupakan 16 bit LSB alamat pengirim.
6. Membuat variabel bertipe *TxStatusResponse*. Variabel ini akan menampung hasil dari fungsi *TxStatusResponse().*
7. Membuat algoritma pengiriman data yakni mengisi *payload* yang telah dibuat dengan data yang akan dikirim.

Pada program pengiriman data pada *router* dibuat variabel dengan nama *xbee* yang bertipe *Xbee* kemudian variabel *buffer* untuk menampung data yang akan dikirim bernama *payload* dengan tipe uint8\_t (*unsigned integer 8 bit*) atau 1 *byte* dan berbentuk *array* dengan panjang *array* 13. Panjang *array* ditentukan dengan menghitun banyaknya *byte* data yang akan dikirim. Data – data yang akan dikirim diantaranya adalah sebagai berikut.

1. Data dari unit RTC berupa data jam, menit, detik, tanggal, bulan dan tahun yang masing – masing 1 *byte* sehingga panjang data dari RTC berjumlah 6 *byte.*
2. Data jarak perpindahan modul hasil integrasi data percepatan yang dibaca oleh akselerometer. Data jarak perpindahan terdiri dari 3 sumbu yang tiap – tiap sumbu terdiri dari 2 *byte* sehingga jumlah data yang dikirim sebanyak 6 *byte*. Alasan mengapa data perpindahan membutuhkan alokasi 2 *byte* dikarenakan data perpindahan ada yang bernilai negatif sedangkan mengirimkan bilangan negatif pada komunikasi serial sering menimbulkan masalah pada perhitungan sehingga *payload* bertipe uint8\_t yang memiliki jangkauan nilai dari 0 hingga 255. Agar dapat mengirimkan data negatif maka ditambahkan 1 *byte* lagi sebagai penanda data positif atau negatif. Jika data positif , *byte* ini akan bernilai 0x00 sedangkan jika datanya negatif, *byte* ini bernilai oxff. Data perpindahan sendiri bertipe float dengan ketelitian 2 angka dibelakang koma sehingga agar data perpindahan dapat dikirim maka dilakukan konversi ke uint8\_t dengan dikalikan 100 terlebih dahulu.
3. Jumlah keseluruhan data yang akan dikirim adalah 12 *byte* yakni 6 *byte* data dari RTC dan 6 *byte* data jarak perpindahan. Dengan mengirimkan data sebanyak 12 *byte* maka panjang data (*data length*) yang diterima oleh *gateway* seharusnya adalah 17 *byte* karena mendapat tambahan 5 *byte* yang terdiri dari 1 *byte* data *API identifier,* 2 *byte* data *ID* *router*, 1 *byte* data RSSI (*Ricieve Signal Strength*) dan 1 *byte* data *options byte.* Desimal 17 dalam format heksadesimal adalah 0x11.
4. Dalam format API (*Aplication Programming Interface*) mode 2 terdapat *escape byte* yakni *byte – byte* yang berfungsi sebagai karakter kusus dalam format ASCI diantaranya 0x7E (karakter ~ / *start delimeter* pada jaringan Zigbee),0x11(karakter Xon) dan 0x13 (karakter Xoff / enter). Sehingga jika *data length* adalah 0x11 maka akan terdeteksi sebagai *escape byte* . Salah satu cara untuk menghindari *escape byte* adalah dengan menambahkan 1 *byte* data pada *payload* sehingga jumlah *payload* menjadi 13 dan *data length* menjadi 0x12.

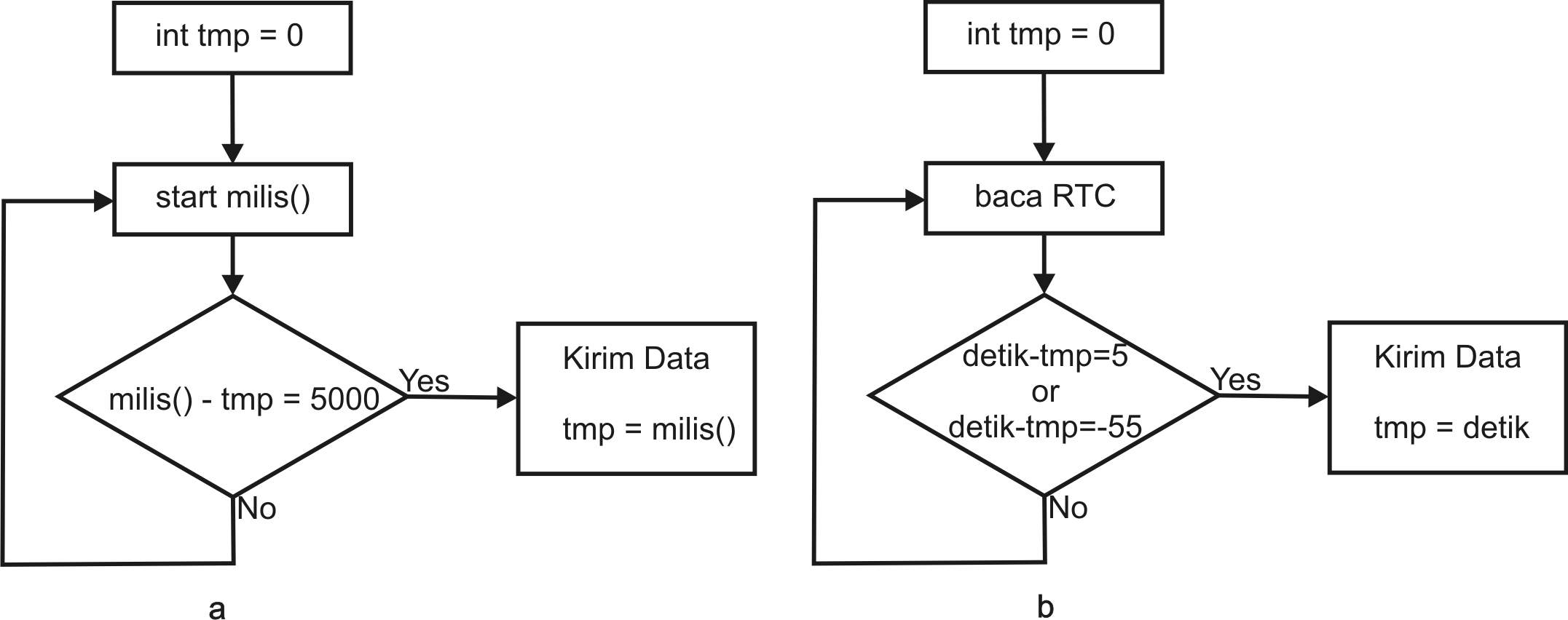
Setelah menentukan format data yang dikirim, langkah selanjutnya adalah menentukan periode pengiriman data.

Untuk menentukan periode pengiriman data dapat dilakukan dengan dua alternatif yaitu memanfaatkan *timer/counter* internal yang dimiliki oleh mikrokontroler atau dengan memanfaatkan *timer* eksternal yaitu dari modul RTC.

Jika menggunakan *timer/counter* internal mikrokontroler pada arduino dapat memanfaatkan fungsi milis()atau micros()*.*fungsi milis() merupakan fungsi penghitung naik (*count up* ) dengan nilai keluaran bertipe unsigned long sehingga jangkauan *counter* ini adalah dari 0 hingga 4.294.967.295 atau 4 *byte* (232 – 1). Nilai *counter* akan bertambah satu setiap 1 milidetik sehingga nilai maksimal 4.294.967.295 akan tercapai setelah 49 hari lebih 17 jam 2 menit 47 detik setelah arduino dijalankan. Dan pada saat ini *counter* akan mengalami *over flow* dan kembali ke – 0. Sedangkan fungsi micros() merupakan fungsi yang serupa dengan milis() akan tetapi nilai *counter* bertambah setiap 1 mikrodetik. Fungsi ini juga bertipe unsigned long sehingga akan mengalami *over flow* setelah 71 menit 35 detik arduino dijalankan.

Alternatif pewaktuan kedua adalah dengan memanfaatkan data yang diberikan oleh unit RTC. Data yang diberikan oleh RTC berupa data tahun, bulan, tanggal, hari, jam, menit dan detik. Untuk mengatur selang waktu pengiriman data dapat memanfaatkan data waktu ini.

Perbedaan algoritma perhitungan selang waktu pengiriman data menggunakan *timer/counter* internal dengan *timer* dari unit RTC ditunjukan pada flowchart pada gambar 4.30 dibawah. Kedua program sama - sama mengatur selang waktu pengiriman data tiap 5 detik.

**

*Gambar 4.30 Flowchart algoritma pewaktuan pengiriman data. (a) menggunakan counter internal, (b) menggunakan RTC*

Kelemahan dari pewaktuan menggunakan *counter* internal yakni pada akurasi pewaktuan. Karena arduino menggunakan crystal 16 MHz tingkat akurasi *counter* sebesar 4 mikrodetik sehingga tiap penambahan nilai maka *error* sebesar 4 mikrodetik ini ikut bertambah juga. *Counter* internal ini juga akan mengalami gangguan jika mikrokontroler mengalami *hang* pada proses perhitungan ataupun proses transfer data antar register.

Sedangkan pemakaian unit RTC sebagai penentu waktu pengiriman data memiliki tingkat akurasi yang lebih baik selama tidak ada gangguan eksternal pada rangkaian. Penggunaan RTC eksternal juga lebih stabil karena sumber *clock* terpisah dari sistem mikrokontroler sehingga tidak terpengaruh oleh proses yang sedang terjadi pada internal mikrokontroler.

Agar pewaktuan pengiriman data berjalan konstan, proses perhitungan selang waktu pengiriman data dibuat dalam subrutin interupsi sehingga apabila nilai selang waktu telah didapat maka program interupsi berjalan dan menunda proses yang sedang terjadi pada program utama.

Program utama pada router terdiri dari pengambilan data dari akselerometer, perhitungan jarak perpindahan serta menyimpan nilai – nilai variabel yang akan dikirim kedalam *payload*. Jadi ketika rutin interupsi berjalan, nilai yang dikirim merupakan nilai terakhir yang disimpan didalam *payload*. Setelah menjalankan rutin interupsi, nilai – nilai pada variabel yang dikirim tadi kemudian direset kembali dan *payload*  diisi dengan nilai yang baru.

Programutama pengiriman data pada *router* dalam format bahasa C arduino adalah sebagai berikut.

void **loop**() {

getDateDs1307(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month, &year);

//==================routine interupsi==================//

noInterrupts();

if((second-tmp == 5) || (second-tmp == -55)){

xbee.send(tx);

Vx, Vy, Vz, Dx, Dy, Dz = 0;

flashLed(statusLed, 1, 300);

tmp = second;

}

interrupts();

//==================Data Waktu==========================//

payload[0] = hour & 0xff;

payload[1] = minute & 0xff;

payload[2] = second & 0xff;

payload[3] = dayOfMonth & 0xff;

payload[4] = month & 0xff;

payload[5] = year & 0xff;

//=================Mengukur Percepatan================//

Ax = ((float(getH48C(0,0) - 2048))\*0.0022)-Xawal;

Ay = ((float(getH48C(0,1) - 2048))\*0.0022)-Yawal;

Az = ((float(getH48C(1,0) - 2048))\*0.0022)-Zawal;

//====remove vibrasi dengan filter IIR ===============//

accelX = IIR(Ax, accelXInput, accelXOutput);

accelY = IIR(Ay, accelYInput, accelYOutput);

accelZ = IIR(Az, accelZInput, accelZOutput);

// ============ hitung kecepatan ==========//

Vx += (accelX \* 0.01);

Vy += (accelY \* 0.01);

Vz += (accelZ \* 0.01);

// =========== hitung perpindahan ========== //

Dx += (Vx \* 0.01);

Dy += (Vy \* 0.01);

Dz += (Vz \* 0.01);

//===========konversi float ke uint8\_t==============//

if (Dx < 0 ){

byteDx = byte(Dx\*(-100));

payload[6] = 0xff;

payload[7] = byteDx & 0xff;

}

else {

byteDx = byte(Dx\*100);

payload[6] = 0x00;

payload[7] = byteDx & 0xff;

}

if (Dy < 0 ){

byteDy = byte(Dy\*(-100));

payload[8] = 0xff;

payload[9] = byteDy & 0xff;

}

else {

byteDy = byte(Dy\*100);

payload[8] = 0x00;

payload[9] = byteDy & 0xff;

}

if (Dz < 0 ){

byteDz = byte(Dz\*(-100));

payload[10] = 0xff;

payload[11] = byteDz & 0xff;

}

else{

byteDz = byte(Dz\*100);

payload[10] = 0x00;

payload[11] = byteDz & 0xff;

}

payload[12] = 0x00;

}

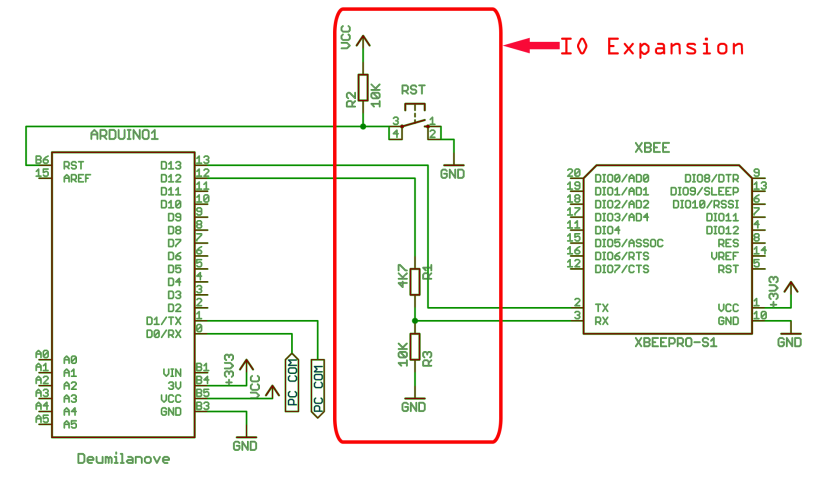
### Skematik rangkaian modul RF pada gateway

Fungsi utama dari *gateway* adalah untuk menerima data yang dikirim dari *router* serta mengirim data ke komputer server. Permasalahan yang dihadapi adalah perangkat modul RF dengan komputer server keduanya menggunakan mode komunikasi UART dengan arduino deumilanove sedangkan jalur komunikasi UART pada arduino deumilanove hanya terdapat 1 buah yakni pin 0 dan pin 1 (pin RX dan pin TX).

Apabila menggunakan Xbee Shield seperti halnya *router* maka diperlukan pemindahan jumper JP1 dan JP2 (gambar 4.28) secara terus menerus sehingga jalur komunikasi UART pada arduino dapat digunakan secara bergantian dan transfer data baik dari modul RF ke arduino atau dari arduino ke komputer server dapat dilakukan. Hal ini sangat tidak efisien apabila dilakukan secara manual. Apabila menggunakan *switching* secara elektronis misalnya menggunakan transistor atau optocoupler maka pewaktuan proses pensaklaran tergantung dari karakteristik komponen yang digunakan sehingga dapat dimungkinkan adanya *error* pada proses transfer data.

Salah satu solusi dari permasalahan tersebut adalah dengan menggunakan pendekatan *software* yakni dengan memanfaatkan *library* NewSoftSerial yang telah tersedia pada paket arduino IDE 0022. *Library* NewSoftSerial dibuat oleh David A. Mellis (2006). Dengan menggunakan *library* NewSoftSerial maka dapat membuat pin arduino yang bukan jalur komunikasi UART berfungsi sebagai jalur komunikasi UART sehingga dalam satu modul arduino deumilanove dapat dimungkinkan terdapat lebih dari satu jalur komunikasi UART.

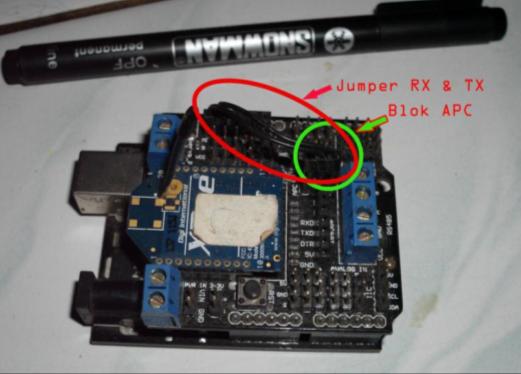
Skematik rangkaian pada *gateway* sesuai dengan gambar 4.31 berikut.



**Blok APC**

*Gambar 4.31 Skematik* rangkaian *pada gateway*

Pada *gateway*, perantaraan Xbee dengan arduino dibantu menggunakan arduino IO Expansion. Posisi pin TX dan RX modul Xbee terhubung pada blok APC yang terlebih dahulu melalui penyesuai level tegangan pada jalur RX Xbee. Untuk menghubungkan pin TX dan pin RX yang terdapat pada blok APC dari arduino IO Expansion dengan pin 12 dan pin 13 arduino deumilanove dilakukan dengan menambahkan kabel *jumper* seperti pada gambar 4.32.

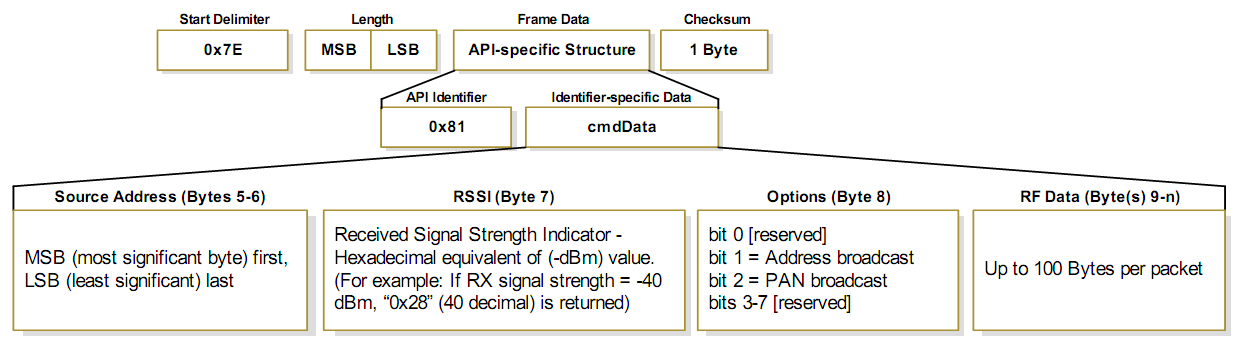


*Gambar 4.32 Jumper RX dan TX* pada *blok APC dari arduino IO Expansion dengan pin 12 dan pin 13*

### Algoritma pembacaan data pada gateway

Fungsi utama *gateway* adalah sebagai penerima data yang dikirim secara *wireless* oleh *router* dan mengirimkan kembali ke komputer server melalui komunikasi serial. Tugas ini sebenarnya dapat dilakukan dengan cara melewatkan langsung data yang diterima oleh Modul RF kemudian mengirimkan ke komputer server. Akan tetapi data yang diterima merupakan data yang ter-enkripsi dalam format API, sedangkan pengamatan terhadap data yang ter-enkripsi relatif lebih sulit jika dibandingkan dengan data yang tidak ter-enkripsi. Sehingga proses enkode frame data yang diterima dilakukan pada modul *gateway*.

Langkah pertama proses enkode frame data yang diterima adalah dengan mengenali tipe frame yang dikirim oleh *router*. Pengalamatan modul yang digunakan adalah dengan menggunakan 16 bit alamat sehingga tipe frame API pada protokol zigbee dengan mode 16 bit alamat adalah 0x81 (Maxstream.Inc, 2007). Tipe frame data pada API *identifier* = 0x81 ditunjukan pada gambar 4.33 berikut.



*Gambar 4.33 Tipe frame API untuk Xbee RX packet pada mode pengalamatan 16 bit (Maxstream.Inc, 2007)*

Dapat dilihat format frame data pada tipe frame 0x81 bahwa dalam satu frame terdiri dari *start delimeter*(*byte* ke-1), data *length* (*byte* ke-2 dan ke-3), *Frame Data* (*byte* ke-4 hingga ke-n) dan *checksum* (*byte* terakhir).

Setelah mengetahui tipe frame API yang dikirim oleh router, langkah selanjutnya adalah menganalisa format frame data yang diterima. Frame data terdiri dari *Api Identifier* (*byte* ke-4) dan *identifier Specific Data* yang terdiri dari *Source Address* (*byte* ke-5 dan ke-6), *RSSI* (*byte* ke-7), *Options* (*byte* ke-8), serta *RF data* (*byte* ke-9 hingga ke-n). Didalam *RF Data* inilah data yang kita kirim berada.

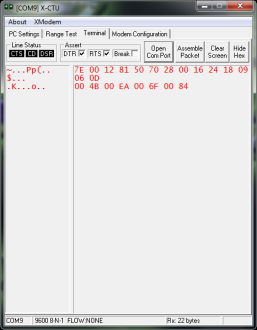
Data yang kita kirim merupakan data yang disimpan dalam variabel *payload* pada modul *router* yang telah kita bahas sebelumnya. Sehingga kita telah mengetahui jumlah *payload* yakni sebanyak 13 *byte*. Sehingga data dari *payload* akan menempati *byte* ke-9 hingga ke-21.

Sampai disini kita dapat menentukan format frame data yang akan diterima oleh *gateway* yaitu sebagai berikut.

|  |  |  |
| --- | --- | --- |
| *byte* ke-1 | = | 0x7E |
| *byte* ke-2 | = | 0x00 |
| *byte* ke-3 | = | 0x12 |
| *byte* ke-4 | = | 0x81 |
| *byte* ke-5 | = | MSB address |
| *byte* ke-6 | = | LSB address |
| *byte* ke-7 | = | RSSI |
| *byte* ke-8 | = | 0x00 |
| *byte* ke-9 | = | jam |
| *byte* ke-10 | = | menit |
| *byte* ke-11 | = | detik |
| *byte* ke-12 | = | tanggal |
| *byte* ke-13 | = | bulan |
| *byte* ke-14 | = | tahun |
| *byte* ke-15 | = | flag Dx |
| *byte* ke-16 | = | Dx |
| *byte* ke-17 | = | flag Dy |
| *byte* ke-18 | = | Dy |
| *byte* ke-19 | = | Flag Dz |
| *byte* ke-20 | = | Dz |
| *byte* ke-21 | = | 0x00 |
| *byte* ke-22 | = | checksum |

Langkah selanjutnya adalah melakukan uji coba membaca data yang diterima oleh modul Xbee, dengan cara menghubungkan modul Xbee ke komputer server kemudian mengamati data yang diterima pada *softwere* terminal. Untuk uji coba ini dapat memanfaatkan X-CTU , CoolTerm atau softwere terminal yang lain yang dapat menampilkan bilangan heksadesimal dari *byte* data yang diterima.

Hasil pengujian menggunakan X-CTU dapat dilihat pada gambar 4.33 dibawah.



*Gambar 4.33 Data yang diterima oleh modul Xbee*

Dari data yang diterima oleh modul Xbee yang ditunjukan pada gambar 4.33 dapat dianalisa sebagai berikut.

*Byte* ke-1 bernilai 0x7E yang berarti*start delimeter*. Kemudian dilanjutkan dengan *byte* ke-2 dan ke-3 secara berturut – turut merupakan MSB dan LSB dari *Data Length* yaitu jumlah *byte* dari *API Identifier*  hingga *bye* terakhir sebelum *checsum* atau dari *byte* ke-4 hingga *byte* ke-21 yang berjumlah 17 *byte*. (dalam format heksadesimal = 0x12).

Modul Xbee yang digunakan pada router terdiri dari 2 buah yang masing – masing memiliki alamat 5070 dan 4D36 sehingga jika *byte* ke-5 digeser ke kiri 8 bit kemudian ditambahkan dengan *byte* ke-6 akan memiliki dua kemungkinan nilai yaitu 0x5070 atau 0x4D36. Nilai ini tergantung dari modul mana data ini dikirimkan. Karena data menunjukan 0x5070 maka modul Xbee pengirim mempunyai alamat 0x5070.

*Byte* ke-7 bernilai 0x28, sedangkan *byte* ini merupakan nilai RSSI negatif dalam skala dBm (desibell miliwat). Sehingga pada pengujian diatas menunjukan bahwa RSSI bernilai -40 dBm.

*Byte* ke-8 bernilai 0x00 menunjukan bahwa opsi pengiriman yang digunakan oleh pengirim secara *reserved* yang artinya data yang dikirim hanya untuk alamat tertentu saja*.* Jika *byte* ini bernilai 0x01 maka pengiriman yang digunakan secara broadcast dalam satu PAN (*Personal Area Network*) sedangkan jika bernilai 0x02 maka opsi pengiriman yang digunakan oleg pengirim adalah secara broadcast antar PAN.

*Byte* ke-9 hingga 14 merupakan data waktu pengiriman data yang didapat dari unit RTC pada modul *router*. Secara berturut – turut data ini menujukan waktu pukul 22:36 .24 tanggal 9/6/13.

*Byte* ke-15, ke-17, dan ke-19 merupakan *flag* atau penanda nilai bilangan pada *byte* selanjutnya yang telah diset sebelumnya pada program pengiriman data pada *router*. Sehingga dari data yang diterima didapat informasi bahwa *byte* ke-16, ke-18 dan ke-20 semuanya bernilai positif.

*Byte* ke-16, ke-18 dan ke-20 secara berturut – turut merupakan data perpindahan pada sumbu X, Sumbu Y dan sumbu Z dan nilainya menunjukan perpindahan sebesar Dx = 75 cm, Dy = 234 cm dan Dz = 111 cm.

*Byte* ke-21 merupakan *byte* yang didapat dari *payload[13]* pada program *router* yang berfungsi untuk menghindari adanya *escape byte*. Sedangkan *byte* ke-22 adalah *byte* terakhir atau *checksum.* *Byte* ini dapat digunakan sebagai pengkoreksi data yang dikirim benar atau tidak. Untuk menguji jikadata dikirim dengan sempurna dilakukan dengan cara menjumlahkan seluruh *byte* yang diterima kecuali *byte* ke-1 sampai *byte* ke-3. Data yang dikirim tidak cacat jika hasil penjumlahan pada 1 *byte* LSB bernilai 0xff. Jadi hasil verifikasi dari data yang diperoleh diatas adalah sebagai berikut.

0x81 + 0x50 + 0x70 + 0x28 + 0x00 + 0x16 + 0x24 + 0x18 + 0x06 + 0x0D + 0x00+ 0x4B + 0x00 + 0xEA + 0x00 + 0x6F + 0x00 + 0x84 = 0x3FF

Nilai 1 *byte* LSB dari hasil perhitungan diatas adalah 0xFF sehingga dapat dipastikan bahwa data yang dikirim tidak ada cacat.

Langkah selanjutnya adalah menyusun algoritma yang akan digunakan untuk memprogram mikrokontroler. Penyusunan algoritma program pada *gateway* didasarkan pada 2 pokok permasalahan yaitu :

1. Jalur komunikasi.
2. Informasi apa saja yang akan dikirim ke komputer server.

Penentuan jalur komunikasi telah dibahas sebelumya yaitu dengan pendekatan software dengan memanfaatkan *library* NewSoftSerial sehingga diawal program perlu dimasukan header NewSoftSerial.h. Setelah memasukan *header* kemudian membuat *object* dengan *class* NewSoftSerial dan menentukan pin Rx dan Tx yang digunakan untuk komunikasi UART. *Class* NewSoftSerial memiliki beberapa *public methods* berikut.

NewSoftSerial(uint8\_t receivePin, uint8\_t transmitPin, bool inverse\_logic = false);

~NewSoftSerial();

void begin(long speed);

void end();

int read();

uint8\_t available(void);

bool active()

bool overflow()

static int library\_version()

static void enable\_timer0(bool enable);

void flush();

Jalur komunikasi yang terbentuk akan digunakan untuk transfer data antara modul Xbee dengan mikrokontroler sehingga *object* ini diberi nama xbee dengan pin Tx pada pin 12 dan pin Rx pada pin 13. Sampai disini kita telah memiliki jalur komunikasi UART pada pin 12 dan pin 13.

Potongan listing program untuk konstruktor jalur UART ini sebagai berikut.

#include <NewSoftSerial.h>

NewSoftSerial xbee (13,12);

Selanjutnya adalah menyiapkan variabel untuk menampung *byte* data yang diterima dari modul Xbee. Mengingat bahwa didalam 1 frame terdapat 22 *byte* data sehingga variabel ini berbentuk *array* dari uint8\_t dengan lebar *array* 22.

Langkah selanjutnya adalah mengisi *array*  dengan *byte* data yang diterima dari modul Xbee dan kemudian memilih data mana saja yang akan dikirim ke komputer server. Proses ini dilakukan dengan membaca *byte* data satu persatu dan memasukanya kedalam tiap – tiap indeks *array*.

Informasi yang perlu ditampilkan di komputer server diantaranya adalah alamat pengirim data, data RSSI, data waktu pengiriman, serta data perpindahan sehingga *array* yang akan diproses nilainya pada indeks 5,6,7,9 hingga 20. Mengingat kembali bahwa data akselerometer memiliki *flag* pada *byte* ke-15, 17 dan 19 maka untuk menentukan tanda bilangan pada data percepatan dilakukan dengan memanfaatkan instruksi if (conditional) dengan parameter yang diuji adalah array yang menunjukan *flag* untuk data perpindahan yakni pada indeks ke-14, 16 dan 18 (indek *array* pertama adalah 0). Potongan program pada algoritma utama penerimaan data dari modul Xbee dan mengirimkan data ke komputer server adalah sebagai berikut.

void **loop**() {

int sX, sY, sZ;

while (xbee.available() > 0) {

int b = readByte();

if (b == 0x7E) {

packet[0] = b;

packet[1] = readByte();

packet[2] = readByte();

int Length = ((packet[1] << 8) | packet[2]);

for(int i=1;i<=(Length+1);i++) {

packet[2+i] = readByte();

}

printPacket(Length+4);

int frametype = packet[3];

if (frametype == 0x81) {

int router = (packet[4] << 8) | packet[5];

**Serial**.print("Router : ");

**Serial**.println(router,HEX);

int RSSI = packet[6];

**Serial**.print("RSSI :");

**Serial**.println(RSSI, HEX);

int hour = packet[8];

int minute = packet[9];

int second = packet[10];

int dayOfMonth = packet[11];

int month = packet[12];

int year = packet[13];

**Serial**.print(hour,DEC); **Serial**.print(":");

**Serial**.print(minute,DEC); **Serial**.print(":");

**Seria**l.print(second,DEC); **Serial**.print(" ");

**Serial**.print(dayOfMonth,DEC); **Serial**.print("/");

**Serial**.print(month,DEC); **Serial**.print("/");

**Serial**.println(year,DEC);

if (packet[14] == 0x00){

sX = packet[15] ; }

if (packet[14] == 0xff){

sX = packet [15]\*(-1);}

if (packet[16] == 0x00){

sY = packet[17] ; }

if (packet[16] == 0xff){

sY = packet [17]\*(-1);}

if (packet[18] == 0x00){

sZ = packet[19] ; }

if (packet[18] == 0xff){

sZ = packet [19]\*(-1);}

**Serial**.print("Perpindahan X :");

**Serial**.println(sX,DEC);

**Serial**.print("Perpindahan Y :");

**Serial**.println(sY,DEC);

**Serial**.print("Perpindahan Z :");

**Serial**.println(sZ,DEC);

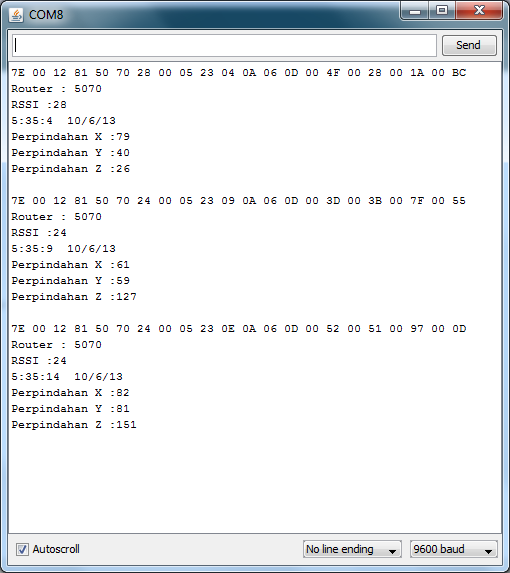
}

}

}

}

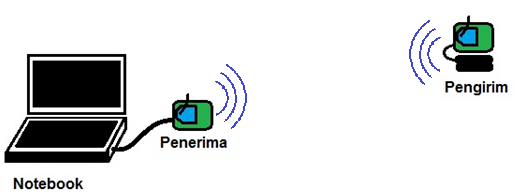
Hasil yang didapat pada program pembacaan dan pengiriman data pada *gateway* ditunjukan oleh gambar 4.34 dibawah ini.



*Gambar 4.34 Hasil runing program pada gateway*

## Pengujian RSSI (*Receive Signal Strength Indicator*)

Untuk pengujian RSSI dilakukan dengan 2 cara yakni menguji kekuatan sinyal pada daerah yang tidak ada halangan atau LOS (*Line Of Sight*) dan menguji kekuatan sinyal pada daerah yang terdapat halangan. Skematik yang digunakan untuk pengujian ini adalah sebagai berikut (gambar 4.35 )



*Gambar 4.35 Skematik rangkaian pengujian RSSI*

Perangkat yang dibutuhkan untuk pengujian ini adalah sebagai berikut.

* Notebook sebagai media penampil dan penyimpan data.
* Modul penerima terdiri dari modul Xbee Pro Series 1 dan Arduino.
* Modul pengirim terdiri dari modul Xbee Pro Series 1, Arduino dan batere *lithium* 7,4 Volt sebagai catu daya modul.
* Meteran gulung
* Tali/benang dengan panjang 20 meter.
* Tiang penyangga modul dengan panjang 1,3 meter hingga 1,5 meter.

Untuk seting parameter dari tiap modul Xbee Pro yang digunakan sesuai dengan tabel 4.10 berikut.

Tabel 4.10 Seting parameter Xbee untuk pengujian RSSI

|  |  |  |
| --- | --- | --- |
| Parameter | Pengirim | Penerima |
| BD (*Baud Rate*) | 3 (9600 bps) | 3 (9600 bps) |
| DL (*Destination Address Low)* | 7E63 | FFFF |
| MY (*16 bit Source Address*) | 4D36 | 7E63 |
| PL (*Power Level*) | 4 (10 dBm) | 4 (10 dBm) |

Untuk pengujian RSSI ini berkaitan erat dengan *Power Level*dari modul yang digunakan. Pada parameter PL (Power *Level*) diseting 4 dimana modul yang digunakan adalah Xbee Pro yang varian internasional sehingga nilai 4 berarti *power level* dari modul diseting 10 dBm.

Algoritma yang digunakan untuk pengujian RSSI ini adalah sebagai berikut.

Inisialisasi

Xbee.Avaliable >0

Xbee Kirim Data ke Penerima

Yes

No

Inisialisasi

Xbee.Avaliable >0

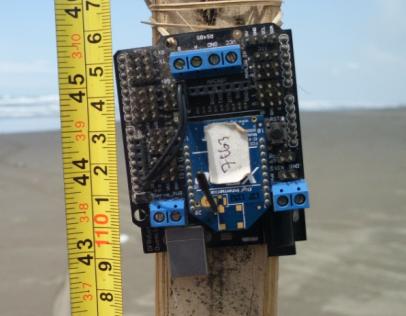
Cetak ke Port Serial COM

Yes

No

*Gambar 4.36 Flowchart algoritma modul penerima (kiri) dan modul pengirim (kanan) yang digunakan pada pengujian RSSI*

Untuk pengujian LOS dilakukan di pantai Widarapayung kecamatan Binangun kabupaten Cilacap. Pengujian dilakukan hingga jarak 500 meter dengan data yang diambil per 20 meter. Untuk menghindari adanya halangan sehingga antara modul pengirim dan penerima benar – benar LOS, tiap modul diletakan pada ketinggian 1 meter hingga 1,3 meter diatas tanah (gambar 4.37).



*Gambar 4.37 Tinggi modul dari* permukaan *tanah (kiri : Pengirim, kanan : Penerima)*

Data yang diperoleh dari pengujian ini adalah sebagai berikut (Tabel 4.11).

Tabel 4.11 Data pengujian RSSI modul terhubung secara LOS

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NO | Jarak (m) | RSSI Sample ke -n (dalam Heksadesimal) | | | | | | | | | | | RSSI | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Rata - Rata | Rata - Rata | |
| 1 | 20 | 3C | 3C | 3C | 3C | 3C | 3C | 3C | 3C | 3C | 3C | 3C | -60 | dBm |
| 2 | 40 | 49 | 4C | 4D | 4D | 4D | 4D | 4D | 4D | 4E | 4A | 4C | -76 | dBm |
| 3 | 60 | 4F | 4D | 4F | 4F | 50 | 50 | 4F | 55 | 53 | 4D | 4F | -79 | dBm |
| 4 | 80 | 54 | 57 | 58 | 62 | 5F | 5D | 59 | 58 | 62 | 53 | 5A | -90 | dBm |
| 5 | 100 | 5D | 59 | 62 | 5A | 59 | 5F | 62 | 65 | 57 | 5D | 5D | -93 | dBm |
| 6 | 120 | 62 | 5F | 62 | 62 | 5D | 5F | 5D | 5F | 65 | 5F | 60 | -96 | dBm |
| 7 | 140 | 56 | 5A | 62 | 58 | 62 | 5A | 65 | 5D | 62 | 65 | 5E | -94 | dBm |
| 8 | 160 | 65 | 62 | 62 | 62 | 62 | 62 | 65 | 5F | 65 | 65 | 62 | -98 | dBm |
| 9 | 180 | 5D | 65 | 62 | 62 | 62 | 65 | 62 | 62 | 5C | 62 | 61 | -97 | dBm |
| 10 | 200 | 5D | 62 | 65 | 65 | 62 | 5F | 62 | 65 | 62 | 62 | 62 | -98 | dBm |
| 11 | 220 | 62 | 65 | 65 | 65 | 65 | 65 | 65 | 65 | 65 | 65 | 64 | -100 | dBm |
| 12 | 240 | 62 | 59 | 54 | 56 | 54 | 54 | 55 | 5C | 58 | 54 | 57 | -87 | dBm |
| 13 | 260 | 65 | 5C | 58 | 58 | 59 | 5A | 5C | 5C | 5A | 5A | 5B | -91 | dBm |
| 14 | 280 | 53 | 57 | 52 | 57 | 54 | 52 | 51 | 50 | 4F | 52 | 52 | -82 | dBm |
| 15 | 300 | 59 | 59 | 57 | 57 | 57 | 57 | 57 | 54 | 54 | 56 | 56 | -86 | dBm |
| 16 | 320 | 5D | 5A | 5C | 5A | 5A | 5D | 5F | 5C | 5D | 62 | 5C | -92 | dBm |
| 17 | 340 | 55 | 53 | 50 | 56 | 58 | 5A | 62 | 5C | 5F | 5D | 59 | -89 | dBm |
| 18 | 360 | 62 | 5D | 62 | 5F | 5F | 5D | 5D | 5F | 62 | 5F | 5F | -95 | dBm |
| 19 | 380 | 65 | 5D | 5F | 62 | 62 | 5C | 5D | 5C | 5D | 5C | 5E | -94 | dBm |
| 20 | 400 | 62 | 62 | 5D | 65 | 65 | 65 | 65 | 65 | 65 | 65 | 63 | -99 | dBm |
| 21 | 420 | 65 | 65 | 62 | 62 | 65 | 5D | 62 | 62 | 62 | 65 | 62 | -98 | dBm |
| 22 | 440 | 5F | 5C | 62 | 62 | 62 | 5F | 5F | 62 | 5D | 5F | 5F | -95 | dBm |
| 23 | 460 | 62 | 62 | 5F | 62 | 65 | 65 | 65 | 62 | 62 | 65 | 62 | -98 | dBm |
| 24 | 480 | 65 | 62 | 62 | 62 | 5F | 62 | 62 | 62 | 55 | 62 | 60 | -96 | dBm |
| 25 | 500 | 65 | 65 | 65 | 65 | 62 | 62 | 62 | 65 | 62 | 65 | 63 | -99 | dBm |

Berdasarkan *data sheet* Xbee Pro Series 1 (Maxstream, 2007) sensitivitas penerimaan sinyalnya adalah dari -36 dBm hingga -100 dBm sehingga dari data yang ada dapat dibuat grafik persentase RSSI terhadap jarak dengan perhitungan sebagai berikut.

Dan hasilnya adalah sebagai berikut.(gambar 4.38)

*Gambar 4.38 Grafik perbandingan nilai RSSI terhadap jarak*

Pengujian RSSI berikutnya adalah pengujian modul pada daerah berpenghalang. Pengujian ini dilakukan di pesisir Pantai Widara Payung dimana daerah tersebut merupakan daerah perkebunan yang ditanami pohon kelapa dan dibawahnya ditanami palawija serta ada beberapa yang tidak ditanami palawija sehingga ditumbuhi tumbuhan liar dan perdu. Gambar 4.39 menunjukan lokasi pengujian RSSI untuk daerah yang berpenghalang.



*Gambar 4.39 Lokasi pengujian nilai RSSI pada medan berpenghalang*

Data RSSI yang diambil tiap 10 meter dan dilakukan hingga tidak ada data yang terkirim. Dari hasil percobaan, jarak terjauh data dapat diterima oleh modul penerima adalah 140 meter.

Data hasil pengujian ditunjukan oleh tabel 4.12 dan grafik perbandingan nilai RSSI terhadap jarak ditunjukan oleh gambar 4.40.

Tabel 4.12 Data hasil pengujian RSSI pada medan berpenghalang

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NO | Jarak (m) | RSSI Sample ke -n (dalam Heksadesimal) | | | | | | | | | | | RSSI | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Rata - Rata | Rata - Rata | |
| 1 | 10 | 42 | 42 | 42 | 3F | 3F | 40 | 3C | 43 | 42 | 47 | 41 | -65 | dBm |
| 2 | 20 | 4F | 49 | 4C | 4E | 46 | 4E | 47 | 46 | 44 | 4F | 4A | -74 | dBm |
| 3 | 30 | 4F | 50 | 50 | 58 | 4A | 49 | 4D | 51 | 49 | 4E | 4E | -78 | dBm |
| 4 | 40 | 58 | 56 | 53 | 52 | 52 | 59 | 53 | 52 | 5F | 53 | 55 | -85 | dBm |
| 5 | 50 | 54 | 5F | 57 | 5F | 56 | 54 | 4D | 62 | 59 | 57 | 58 | -88 | dBm |
| 6 | 60 | 4F | 58 | 59 | 56 | 59 | 4C | 5A | 52 | 53 | 5F | 55 | -85 | dBm |
| 7 | 70 | 62 | 5D | 65 | 65 | 62 | 59 | 65 | 65 | 65 | 59 | 61 | -97 | dBm |
| 8 | 80 | 59 | 5D | 5A | 5F | 59 | 5D | 5C | 65 | 5F | 65 | 5D | -93 | dBm |
| 9 | 90 | 5D | 5D | 59 | 5D | 5A | 62 | 58 | 65 | 57 | 5A | 5C | -92 | dBm |
| 10 | 100 | 5A | 5F | 58 | 5D | 5A | 5A | 5C | 5C | 5D | 53 | 5A | -90 | dBm |
| 11 | 110 | 65 | 5F | 65 | 65 | 5F | 65 | 62 | 62 | 5F | 62 | 62 | -98 | dBm |
| 12 | 120 | 62 | 5D | 65 | 5F | 58 | 5C | 5D | 62 | 62 | 5D | 5E | -94 | dBm |
| 13 | 130 | 65 | 65 | 5F | 65 | 62 | 65 | 65 | 62 | 62 | 65 | 63 | -99 | dBm |
| 14 | 140 | 65 | 5F | 5F |  |  |  |  |  |  |  | 61 | -97 | dBm |

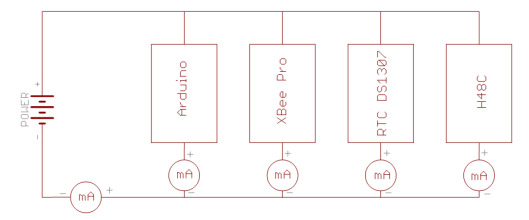
*Gambar 4.40 Grafik perbandingan nilai RSSI terhadap jarak pada medan berpenghalang*

## Konsumsi Daya

Salah satu aspek terpenting dalam jaringan sensor nirkabel adalah pengetahuan mengenai konsumsi daya dari modul yang digunakan. Hal ini dikarenakan *node* akan diletakan didaerah tertentu dan terkadang jauh jala – jala PLN. Sehingga informasi konsumsi daya penting untuk menentukan tipe catu daya yang akan digunakan khususnya konsumsi daya *router.*

Dalam sebuah *node router* terdiri dari 4 bagian utama yaitu kontroler, unit RF, unit RTC dan Sensor. Sehingga untuk mengetahui konsumsi daya sebuah *router* diperlukan pengukuran pada masing – masing bagian.

Pengukuran daya dapat dilakukan dengan mengukur konsumsi arus modul. Skematik pengukuran arus dari sebuah *router* ditunjukan oleh gambar 4.41 dibawah.



*Gambar 4.41 Skematik pengukuran daya dari sebuah router*

Pada pengukuran digunakan catu daya berupa 2 buah sel baterai *Lithium Polimer* dengan tegangan masing – masing 3.6 V dengan kapasitas 2600 mAH dan disusun seri sehingga tegangan catu daya keseluruhan adalah 7.2 V.

Pengukuran dilakukan pada modul yang sedang menjalankan program *router* dan mengirimkan data melalui modul RF ke *gateway* dengan interval pengiriman data 5 detik.

Hasil yang didapat dari pengukuran ditunjukan pada tabel 4.13 dibawah.

Tabel 4.13 Hasil pengukuran konsumsi arus modul *router*

|  |  |
| --- | --- |
| Unit | Arus |
| Kontroler ( Arduino) | 20 mA |
| RF ( Xbee Pro ) | 62 mA |
| RTC ( DS1307 ) | 2 mA |
| Akselerometer ( H48C ) | 1 mA |
| Arus Total | 85 mA |

Dengan arus total 85 mA dan tegangan kerja adalah 7.2 V sehingga konsumsi daya modul adalah 612 mW. Dengan kapasitas baterai 2600 mAH idealnya baterai dapat mensuplai modul selama 30.5 jam akan tetapi pada kenyataanya dapat bervariasi.

Jika menggunakan baterai sebagai catu daya modul, kapasitas baterai sangat menentukan lamanya waktu kerja modul. Sehingga untuk membuat modul bekerja lebih lama dapat dilakukan dengan beberapa cara seperti menambah kapasitas baterai, meminimalisir kerja kontroler, serta memperpanjang jeda waktu pengiriman data.

Menambah kapasitas baterai dapat dilakukan dengan memilih baterai yang kapasitasnya tinggi atau menggabungkan beberapa sel baterai secara paralel. Akan tetapi baterai memiliki hambatan dalam yang menyebabkan baterai mengalami *self-discharge* meskipun baterai tidak digunakan.

Semakin tinggi kerja kontroler menyebabkan kontroler mengkonsumsi arus lebih banyak sehingga untuk memperkecil konsumsi arus dapat dilakukan dengan membuat kerja kontroler seringan mungkin. Ini dapat dilakukan dengan beberapa cara yaitu membuat algoritma yang lebih ringkas, menghindari adanya operasi matematikal yang kompleks, mengganti operasi pembagian dengan operasi perkalian atau dengan operasi penggeseran bit jika mungkin, mengurangi penggunaan tipe *float* karena tipe ini membutuhkan alokasi 32 bit untuk *single precision* dan 64 bit untuk *double precision* sehingga melibatkan operasi bilangan pada *float* akan lebih lama dari pada operasi pada bilangan bulat (*integer*).

Selang waktu pengiriman data berpengaruh sangat besar pada konsumsi daya dari sebuah modul karena modul RF membutuhkan arus yang tinggi pada saat mengirim data. XbeePro sendiri membutuhkan arus 250 mA untuk melakukan pengiriman data dan hanya 55 mA pada saat *idle*. Sehingga semakin cepat interval pengiriman data akan berakibat pada semakin besarnya konsumsi daya dari modul.