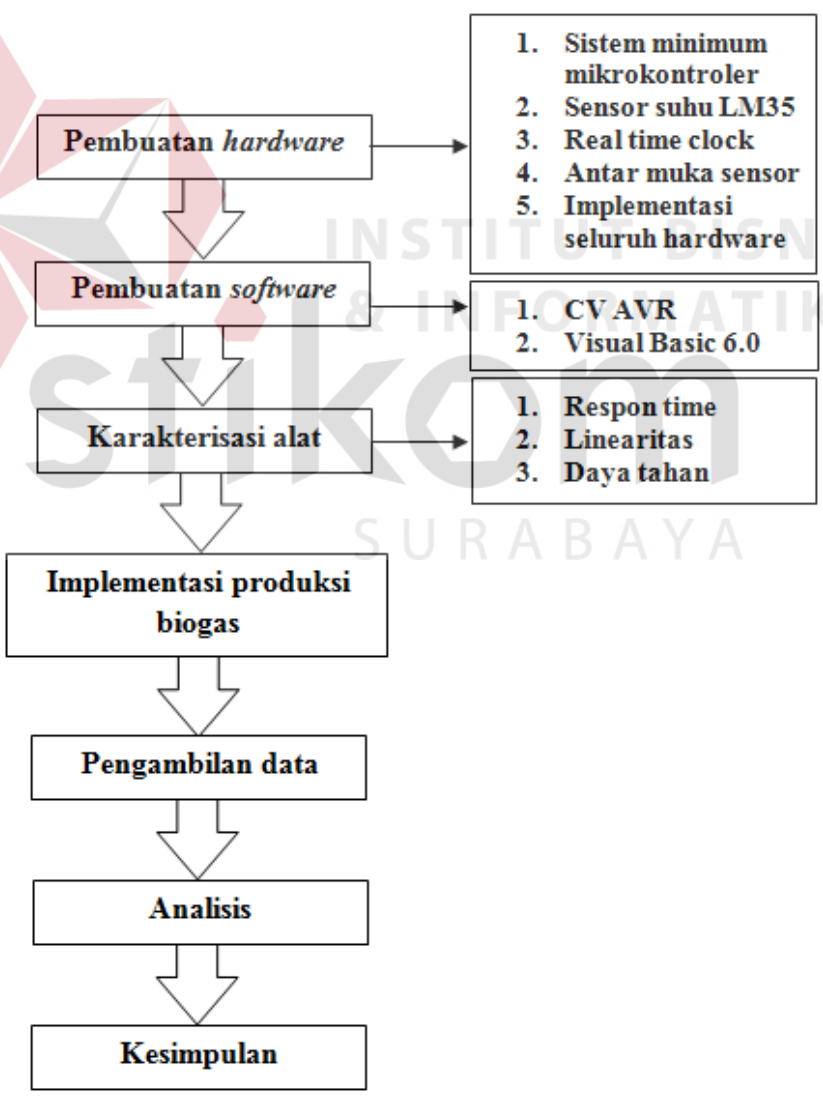
BAB III

METODE PENELITIAN

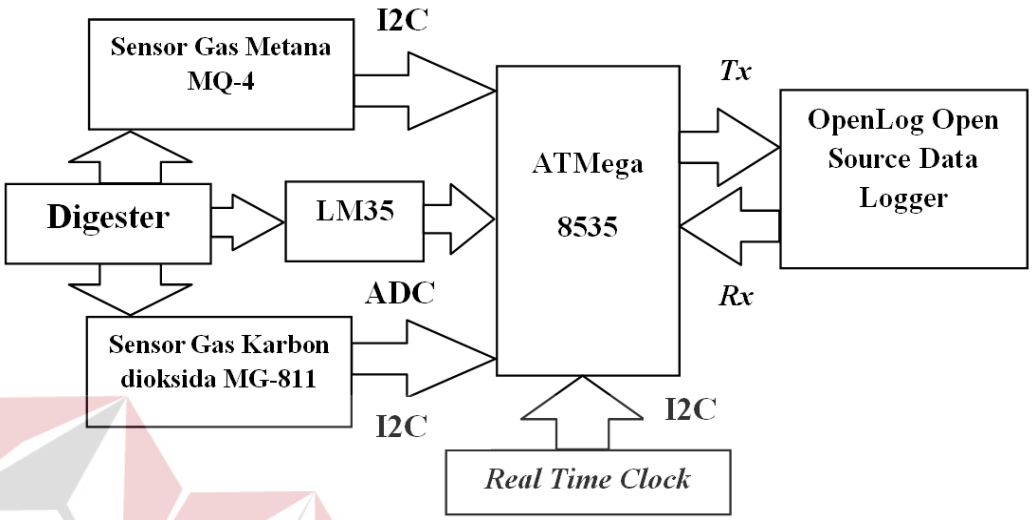
3.1 Model Penelitian

Penelitian dilakukan berdasarkan tahapan yang telah disusun. Tahapan umum penelitian dibuat dalam sebuah skema kerja yang menggambarkan alur pengerjaan penelitian. Tahapan penelitian secara umum dijelaskan dalam skema pada Gambar 3.1.

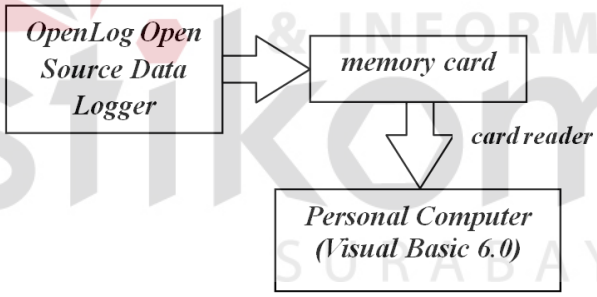


Gambar 3.1 Skema umum penelitian

Pada penelitian terdapat dua tahapan utama yaitu bagian akuisisi data yang dijelaskan oleh blok diagram pada Gambar 3.2 dan bagian analisis data yang dijelaskan oleh blok diagram pada Gambar 3.3.



Gambar 3.2 Diagram blok sistem



Gambar 3.3 Diagram blok analisis data

Pada bagian akuisisi data, terdapat tiga sensor, sensor suhu LM35, sensor gas metana dan sensor gas karbon dioksida yang ada di dalam *digester*. Di luar digester terdapat rangkaian *real time clock* yang berfungsi sebagai *time generator* untuk proses *data logging*. Sebagai kontrol dari sistem, terdapat Atmega8535 yang mempunyai fungsi sebagai pengolah data dari *real time clock* dan sensor-sensor yang ada di dalam *digester*. Selanjutnya, data-data yang berupa data suhu,

data gas metana dan karbon diokisda akan dikirimkan melalui komunikasi serial ke modul *data logger*. Modul *data logger* akan membuat file .TXT yang berisi semua hasil akuisisi data sensor. Pengiriman data sensor tersebut disertai juga dengan data tanggal dan jam untuk menambah *reability* data. Proses ini dilakukan setiap 15 menit sekali dengan mempertimbangkan kondisi suhu yang tidak mudah berubah.

Bagian analisis data merupakan bagian untuk mengelola data yang sudah ada untuk mendapatkan kesimpulan. Dengan tujuan untuk mempermudah proses analisis data, dibuatlah aplikasi analisis data dengan Visual Basic 6.0. Aplikasi ini bertujuan untuk membaca semua data yang tersimpan dalam file .TXT. Semua data yang berjumlah ratusan tersebut akan diseleksi sesuai dengan jenisnya, yaitu data suhu, data gas metana dan data karbon dioksida. Pada tahap terakhir, dibuatlah grafik sesuai dengan jenis datanya untuk melihat pola dan kecenderungan gas metana dan gas terhadap pengaruh suhu.

Untuk menunjang proses analisis, *digester* mendapat 3 jenis perlakuan suhu. *Digester* akan diletakkan pada 3 tempat dengan suhu yang berbeda-beda. *Digester* akan diberi pelakuan pertama yaitu suhu kamar normal dengan cara diletakkan pada kamar yang tidak memiliki pendingin. Kamar tersebut memiliki suhu antara 28^0 sampai 32^0 Celsius. *Digester* diletakkan pada salah satu sudut kamar seperti yang dapat dilihat pada Gambar 3.4.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Gambar 3.4 *Digester* diletakkan pada salah satu sudut kamar

Setelah itu, *digester* akan diberi perlakuan suhu kedua yaitu suhu yang lebih rendah dengan cara dimasukkan ke dalam lemari pendingin. Lemari pendingin tersebut memiliki suhu antara 18^0 sampai 22^0 Celsius. *Digester* diletakkan di dalam lemari pendingin seperti yang dapat dilihat pada Gambar 3.5.

Gambar 3.5 *Digester* diletakkan di dalam lemari pendingin

Setelah 2 perlakuan suhu tersebut *digester* diberi perlakuan terakhir, yaitu suhu tinggi. *Digester* akan diletakkan di dalam sebuah wadah berbentuk tabung tanpa tutup yang memiliki suhu antara 34^0 sampai 44^0 Celsius. Di dasar tabung

tersebut terdapat bohlam 5 Watt yang berfungsi sebagai pemanas seperti yang dapat dilihat pada Gambar 3.6.

Gambar 3.6 Tabung dengan bohlam sebagai pemanas *digester*

3.2. Perancangan Perangkat Keras

3.2.1. Perancangan Digester

Dalam proses pembuatan biogas diperlukan *digester* atau tanki pencerna. Fungsi utama *digester* adalah untuk menampung kotoran sapi dan melakukan proses fermentasi secara anaerob. Agar *digester* dapat mendukung penelitian ini, *digester* dibuat dari bahan penghantar panas dan dingin yang baik. Berikut ini bahan dan spesifikasi dari *digester*.

Bentuk dimensi

Tabung

Ukuran dimensi

Ukuran *digester* : Tabung dengan jari-jari alas 7 cm dan tinggi 25 cm.

Volume = 3.14x7cmx7cm x25cm

= 3.84 liter

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Struktur material

Bahan material yang digunakan : alumunium

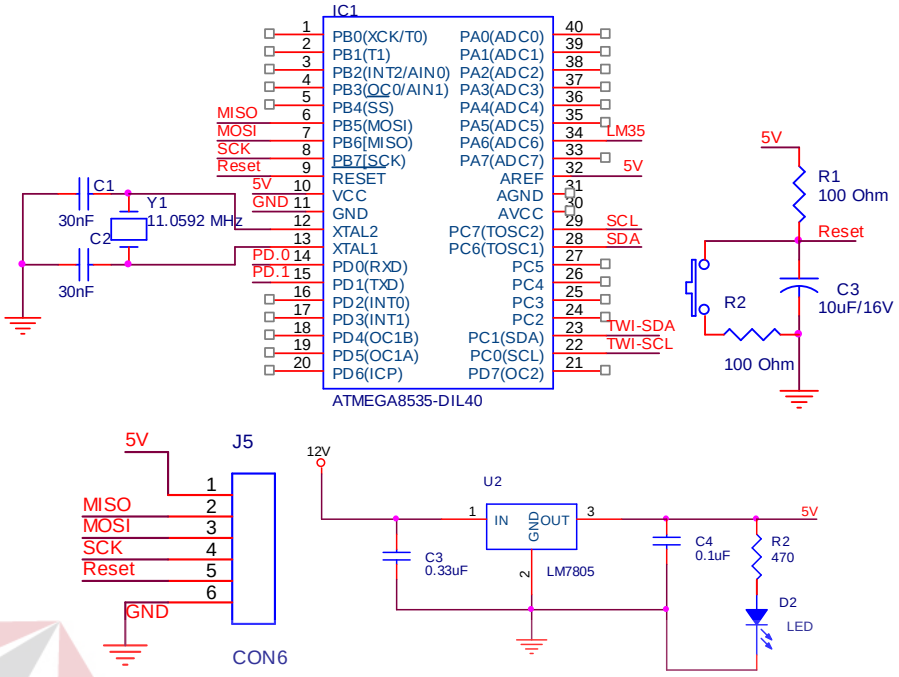
Bentuk fisik *digester* secara detail dapat dilihat pada Gambar 3.7.

Gambar 3.7 *Digester*

3.2.2. Perancangan Sistem Minimum

Mikrokontroler yang digunakan adalah Atmega8535. Mikrokontroler digunakan sebagai pengolah data dari sensor-sensor dan sebagai pengontrol sistem secara keseluruhan. Mikontroler Atmega8535 membutuhkan rangkaian pendukung atau sistem minimum untuk dapat bekerja. Rangkaian sistem minimum terdiri dari rangkaian *clock generator*, rangkaian *reset*, rangkaian *voltage regulator*, dan rangkaian *downloader*. Rangkaian sistem minimum dapat dilihat pada Gambar 3.8.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA



Gambar 3.8 Sistem minimum

Untuk penggunaaan masing-masing pin dari Atmega8535 dapat dilihat pada Tabel 3.1.

Tabel 3.1 Pin I/O Sistem Minimum

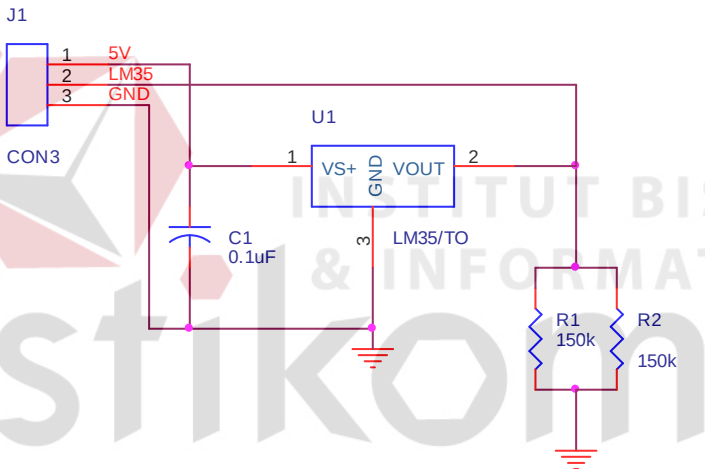
Pin I/O	Fungsi
Vcc	sumber tegangan 5V
Gnd	*Ground*
PD0/RX	disambungkan dengan pin TX *OpenLog*
PD1/TX	disambungkan dengan pin RX *OpenLog*
PC62/SDA	terhubung secara paralel pada jalur SCL (*clock*) dari 2 modul DT sense gas Sensor
PC7/SCL	terhubung secara paralel pada jalur SDA (*data*) dari 2 modul DT sense gas Sensor
Reset	pin reset Atmega8535
PA6	pin masukan ADC dari sensor suhu LM35
PC0/SCL	pin masukan SCL dari TWI yang digunakan oleh RTC
PC1/SDA	pin masukan SDA dari TWI yang digunakan oleh RTC

3.2.3. Perancangan Sensor Suhu LM35

Untuk mengetahui suhu di dalam digester maka sensor yang digunakan adalah LM35. Sensor suhu LM35 mempunyai 3 pin kaki yaitu, pin masukan

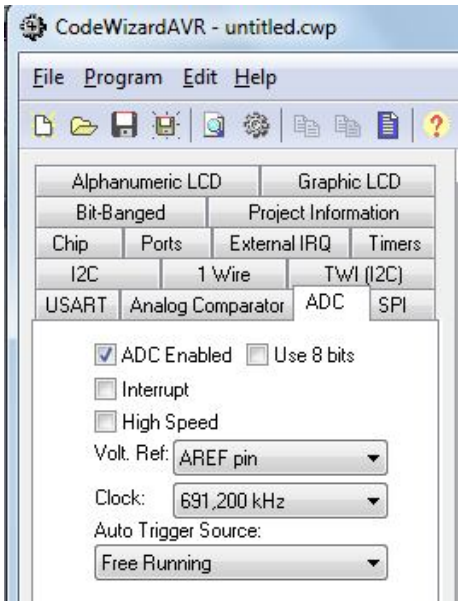
tegangan, pin keluaran data yang dimasukkan ke dalam pin *analog to digital converter* mikrokontroler dan pin *ground*.

Seperti kebanyakan *micropower circuits* lainnya, LM35 mempunyai keterbatasan dalam menangani *capacitive loads* yang besar. Untuk itu perlu ditambahkan *damper* untuk menambah kemampuan toleransi terhadap *capacitive loads* yang besar. Maka dari itu, keluaran dari LM35 diparalel dengan resistor yang disusun secara seri untuk mencegah beban yang berlebih. Lalu, ditambahkan juga kapasitor pada pin tegangan masukan dan *ground* sebagai *bypass capasitor*. Rangkaian sensor suhu LM35 dapat dilihat pada Gambar 3.9.



Gambar 3.9 Rangkaian sensor suhu LM35

Sedangkan untuk konfigurasi *analog to digital converter* untuk LM35 pada CVAVR ditunjukkan seperti pada Gambar 3.10.



Gambar 3.10 Konfigurasi sensor suhu LM35

Volt Ref merupakan sumber pemilihan tegangan referensi ADC, tegangan referensi yang digunakan untuk pemilihan penggunaan tegangan referensi ADC, antara AVCC dan VREF. *Clock* adalah banyaknya frekuensi sampling ADC. Dan *Auto Trigger Source* merupakan mode ADC yang akan digunakan.

LM35 memiliki range pengukuran antara -55^0 sampai dengan 150^0 Celsisus dengan tingkat akurasi 0.5^0 Celsius. Sensor ini memiliki keluaran yang linear sebesar 10mV/^{0} Celcius. Jadi, tiap kenaikkan 10mV maka suhu bertambah 1^0 Celsius.

Dengan menggunakan rangkaian dasar, atau yang disebut *Basic Centigrade Temperature Sensor* yang ada pada *datasheet* LM35, maka LM35 memiliki range pengukuran 0mV – 10mV/^{o}C. Jika dibuatkan kesetaraaan antara tegangan dengan suhu, maka akan terlihat seperti pada Tabel 3.2.

Tabel 3.2 Perbandingan tegangan dan suhu

Tegangan	Suhu
0V	0 ^{o}C
10mv	1 ^{o}C
100mV	10 ^{o}C
1000mV	100 ^{o}C
1500mV	150 ^{o}C

Dengan melihat karakteristik tersebut, maka dengan menggunakan ADC (*Analog to Digital Convertion*) kita bisa melakukan konversi dari tegangan ke suhu menggunakan mikrokontroler ATmega8535 yang mempunyai ADC internal berjumlah 8 channel 10 bit. Jarak tegangan dari 0 sampai dengan tegangan maksimum sama dengan nilai 0 sampai dengan 1024 (n^{10}).

Secara internal, mikrokontroler ATmega8535 menggunakan rumus tegangan masukan dikali 1024 dibagi dengan tegangan referensi. Di mana hasilnya adalah hasil konversi ADC. Dalam penggunaan ADC dengan LM35 sebagai masukan, maka tegangan masukan untuk ADC adalah tegangan keluaran dari LM35. Hubungan tegangan LM35 dan keluaran ADC dijelaskan pada Tabel 3.3.

Tabel 3.3 LM35 dan keluaran ADC

Keluaran LM35	Rumus	Perhitungan Matematis	Keluaran ADC
0 V	0*1024/5000	0	0
1 mV	1*1024/5000	0.2048	0
10 mV	10*1024/5000	2.048	2
1000 mV	1000*1024/5000	204.8	205

Untuk mengubah data hasil pembacaan LM35 menjadi data suhu yang sebenarnya maka cara pembacaan rumus dibalik. Setiap mendapatkan output digital dari mikrokontroler Atmega8535, hasil konversi dalam bentuk tegangan

setiap satu digit LSB yang dikeluarkan oleh ADC mikrokontroler akan bernilai sebesar :

1 LSB = Vref / (2^n – 1)

Jika Vref adalah 5000 mV maka nilai 1 LSB kurang lebih = 4.9 mV (pembulatan). Sehingga rumus konversinya adalah:

Suhu = (Keluaran ADC * Kenaikan satu LSB) / Volt per Celcius

Di mana:

1. Keluaran ADC adalah hasil pembacaan ADC mikrokontroler.
2. Kenaikan satu LSB = 4.9 (jika Vref = 5V)
3. Volt per celcius = 10 (karakteristik LM35, 10mV/^{o}C)

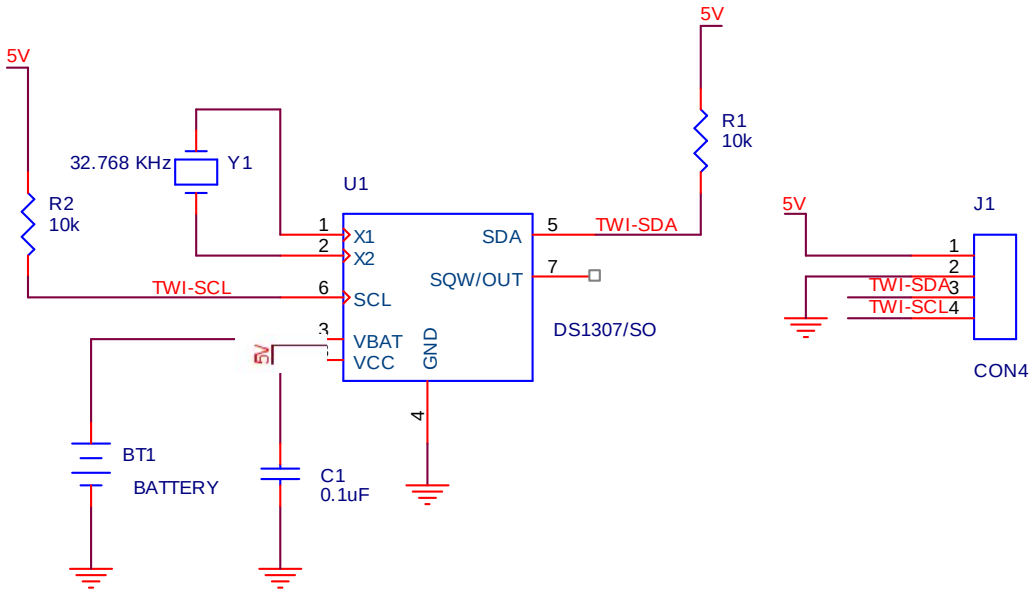
Berikut ini program untuk membaca data ADC dari LM35 berikut perintah konversi agar nilainya menunjukkan nilai suhu sebenarnya.

```
temp = read_adc(6);
temp = (temp * 4.8) / 10;
itoa(temp,suhu);
delay_ms(10);
```

3.2.4. Perancangan *Real Time Clock* DS1307

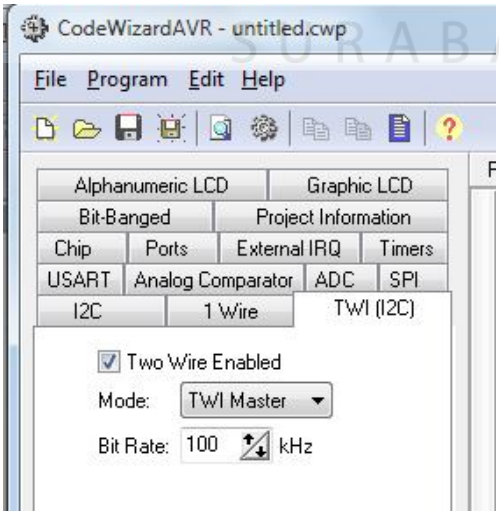
Pada saat sensor melakukan akuisisi data diperlukan data waktu dan tanggal guna mendukung proses *logging* data. Untuk kebutuhan tersebut, maka dirancanglah sebuah *real time clock*, dengan menggunakan DS1307. DS1307 membutuhkan beberapa komponen pendukung yaitu *crystal*, kapasitor, *battery backup* dan *pull up resistor* guna mendukung kinerjanya. Rangkaian DS1307 berserta komponen-komponen pendukungnya dijelaskan pada Gambar 3.11.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA



Gambar 3.11 Rangkaian RTC DS1307

DS1307 menyediakan pin *battery backup* untuk dihubungkan pada baterai lithium 3V atau sumber energi lain sehingga ketika *supply* daya utama mati, *battery backup* mengambil alih *supply* energi pada RTC sehingga DS1307 tetap berjalan. Protokol komunikasi yang digunakan adalah komunikasi TWI. Konfigurasi *real time clock* pada CVAVR dapat dilihat pada Gambar 3.12.



Gambar 3.12 Konfigurasi TWI untuk RTC DS1307

Selanjutnya untuk konfigurasi akses data pada *real time clock* dengan TWI sudah bisa memanfaatkan *library* dari *ds1307_twi.h*. Dari *library* tersebut sudah disediakan fungsi-fungsi yang dibutuhkan untuk membaca data dari *real time clock*.

Agar dapat dipakai, terlebih dahulu harus dilakukan inisialaisi awal berupa pengaturan jam dan tanggal yang sesuai, lalu *download* program ke mikrokontroler. Setelah sudah diberikan inisialisasi awal maka baris program pengaturan jam dan tanggal dapat dihilangkan kemudian *download* program lagi ke mikrontroler. Berikut baris program untuk melakukan inisialisasi awal.

```
rtc_set_time(17,07,07);   //set time 17:07:07
rtc_set_date(0,13,07,13); //set date 13:07:2013
```

Jika inisialisasi awal sudah dilakukan, maka untuk membaca data dapat dilakukan dengan fungsi yang sudah disediakan oleh *library* TWI. Berikut menunjukkan baris perintah untuk membaca dari RTC.

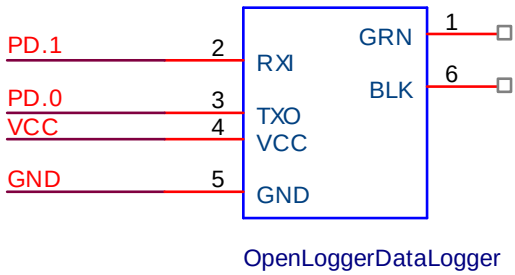
```
rtc_get_time(&h,&m,&s);
rtc_get_date(0,&dd,&mm,&yy);
```

3.2.5. Konfigurasi *Openlog Data Logger*

Untuk keperluan *data logging* maka digunakan *Openlog data logger*. Modul ini mempunyai 6 buah pin yaitu pin RX, TX, VCC, GND, GRN, dan BLK. Namun untuk penggunaan pada Atmega 8535 hanya 4 buah pin yang dipakai.

Komunikasi dengan modul ini menggunakan komunikasi serial USART. Untuk itu pin RX pada modul ini disambungkan dengan pin TX mikrokontroler dan pin TX pada modul disambungkan dengan pin RX pada mikrokontroler. Dan yang paling utama, pin GND harus disambungkan pada GND yang sama yang

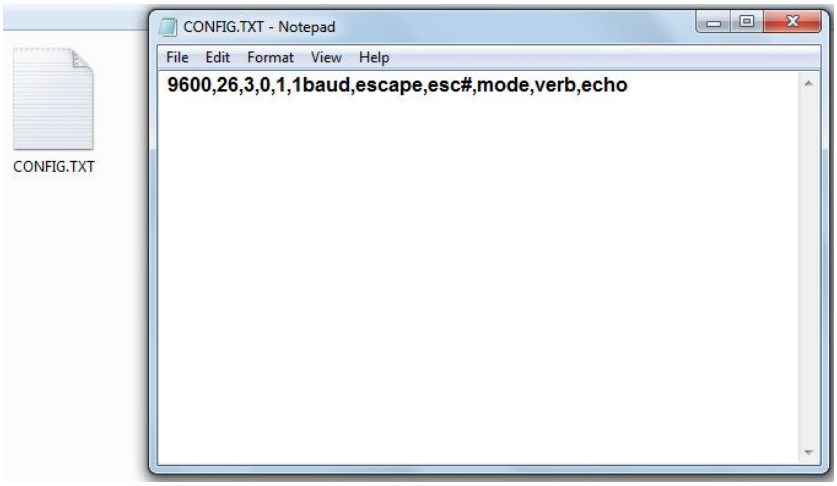
dipakai mikrokontroler guna mendukung komunikasi serial. Skematik dari konfigurasi modul *Openlog data logger* dapat dilihat pada Gambar 3.13.



Gambar 3.13 Rangkaian *Openlog data logger*

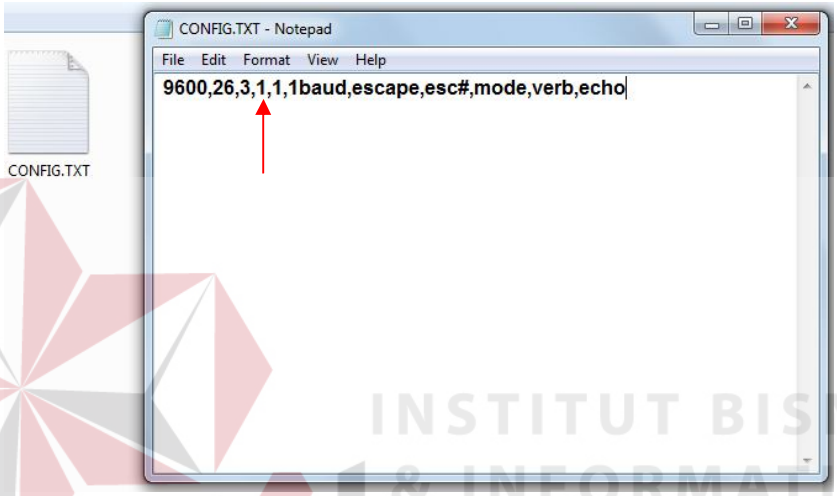
Modul *Openlog* dapat diatur konfigurasinya dengan mengubah isi file CONFIG.TXT yang terdapat pada *memory card* yang sebelumnya sudah dimasukkan ke dalam modul *Openlog* yang sudah diberi catu daya. Konfigurasi yang diubah disesuaikan dengan kebutuhan pengerjaan Tugas Akhir. Konfigurasi yang diubah adalah adalah penggunaan mode *Sequential Log*.

Untuk dapat mengubah konfigurasi ke mode *Sequential Log* maka masukkan *memory card* pada computer dan buka file *CONFIG.TXT* yang ada pada *memory card* dengan notepad. Setelah itu akan muncul serangkaian baris kode perintah seperti yang ditunjukkan pada Gambar 3.14.



Gambar 3.14 Pengaturan CONFIG.TXT

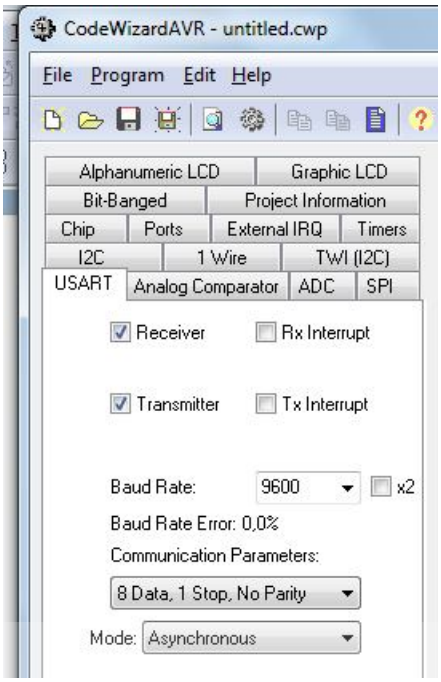
Setiap 1 jenis konfigurasi dibatasi dengan tanda " , " (koma). Konfigurasi *Sequential Log* didapatkan dengan cara mengubah angka secara manual dengan keyboard pada konfigurasi keempat atau setelah tanda koma ketiga. Ubah angka 0 (nol) menjadi 1 (satu) seperti yang ditunjukkan pada Gambar 3.15. Setelah konfigurasi selesai, tekan CTRL+S untuk menyimpan perubahan. Setelah itu keluar dari program notepad dan lakukan *safely remove* pada *memory card*.



Gambar 3.15 Konfigurasi *Sequential Log*

Jika proses tersebut sudah dilakukan, maka masukkan lagi *memory card* pada *OpenLog* dan berikan catu daya. Setelah itu, *led* pada Openlog akan berkedip selama beberapa kali, yang menandakan proses konfigurasi berhasil dilakukan. Kemudian, untuk memeriksa apakah konfigurasi berhasil maka matikan lagi catu daya *OpenLog*. Ambil *memory card* dan masukkan lagi ke dalam komputer. Jika proses konfigurasi berhasil, maka akan ada dua file *.TXT* yaitu *SEQLOG00.TXT* dan *CONFIG.TXT*.

Kemudian, untuk melakukan pengiriman data maka diperlukan konfigurasi USART pada *CVAVR* yang ditunjukkan seperti pada Gambar 3.16.



Gambar 3.16 Konfigurasi USART *Openlog data logger*

Jika konfigurasi pada *CVAVR* sudah selesai, maka untuk mengirimkan data hasil pembacaan sensor digunakan perintah *printf();* yang ada pada *library stdio.h*. Berikut ini barisan program yang mengirimkan data pembacaan sensor-sensor, serta data jam dan tanggal.

```
printf("**");
delay_ms(10);
printf("Jam: %02i:%02i:%02i",h,m,s);
delay_ms(10);
printf("--");
delay_ms(10);
printf("Tgl: %02i:%02i:%02i",dd,mm,yy);
delay_ms(10);
printf("--");
delay_ms(10);
printf("Suhu: %02d",temp);
delay_ms(10);
printf("--");
delay_ms(10);
printf("CO2: %i",sensor);
delay_ms(10);
printf("--");
delay_ms(10);
printf("CH4: %i",sensor4);
delay_ms(10);
printf("**");
```

Setiap awal dan akhir satu paket data diberikan karakter ** guna menandai sepaket data yang satu dengan sepaket data yang lain. Sedangkan untuk antar data diberikan karakter -- untuk membedakan jenis data yang satu dengan yang lain. Pemberiaan karakter khusus ini juga untuk memudahkan proses analisa data dengan program Visual Basic 6.0.

Kemudian, untuk pengiriman antar data memerlukan *delay* sebesar 10 ms. Hal ini merupakan pengaturan yang disarankan oleh pabrikan modul jika mode *Sequential Log* digunakan.

3.2.6. Konfigurasi *DT Sense Gas Sensor*

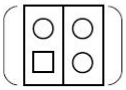
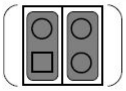
Modul *DT sense gas sensor* digunakan guna mendapatkan hasil pembacaan data sensor yang lebih akurat. Pada Tugas Akhir ini digunakan dua modul *DT sense gas sensor*. Modul *DT sense gas sensor* digunakan untuk masing-masing sensor MQ-4 (metana) dan MG-811 (karbon dioksida). Kedua modul tersebut mempunyai pin SDA dan SCL. Pin SDA dari satu modul diparalel dengan pin SDA lalu disambungkan dengan pin SDA mikrokontroler. Hal yang sama juga berlaku pada pin SCL. Konfigurasi konektor J3 yang dipakai untuk modul *DT sense gas sensor* dapat dilihat pada Tabel 3.4.

Tabel 3.4 Konektor *interface* J3 (Manual rev1, 2013)

Pin	Nama	Fungsi
1	GND	Titik referensi ground untuk catu daya input
2	VCC	Terhubung ke catu daya (5 Volt)
3	SDA	I2C-bus data input / output
4	SCL	I2C-bus clock input

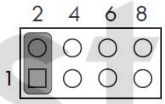
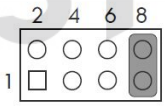
Pada tiap modul terdapat *jumper* untuk mengaktifkan *pull up resistor* untuk pin SDA dan SCL. Karena ada dua modul yang dihubungkan pada satu jalur I2C, maka hanya satu *jumper* dari kedua modul yang diaktifkan. Satu modul dengan

jumper terpasang, dan modul yang lain dengan *jumper* yang dilepas. Konfigurasi untuk *jumper* J1 ditunjukkan pada Gambar 3.17.

Jumper SCL SDA (J1)	Fungsi
	*Pull-up* tidak aktif (*jumper* terlepas)
	*Pull-up* aktif (*jumper* terpasang)

Gambar 3.17 Konfigurasi *jumper* SDA SCL J1

Tiap sensor memiliki konfigurasi *jumper RLOAD* atau *jumper* yang mengatur nilai resistor beban masing-masing. Pengaturan *jumper* pada modul sesuai dengan jenis sensor (MQ-4 atau MG-811) yang dipakai dapat dilihat pada Gambar 3.18.

Jumper RLOAD (J7)	Nilai Resistor Beban dan Rekomendasi Sensor
	Nilai Resistor 2K2 Ohm Sensor MQ-4 dan MQ-135
	Nilai Resistor 100K Ohm Sensor MG-811

Gambar 3.18 Konfigurasi *jumper RLOAD* J7

Kedua modul *DT sense gas sensor* memiliki alamat I2C yang sama yaitu 0xE0. Oleh karena itu alamat salah satu modul harus diganti. Pada sistem ini, alamat yang diganti adalah modul dengan sensor metana MQ-4. Untuk penggantian alamat modul digunakan antar muka UART. Pin TX dan RX yang ada pada modul disambungkan pada pin TX dan RX. Berikut pengaturan komunikasi serial untuk menggati alamat sensor .

```
// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 9600
UCSRA=0x00;
UCSRB=0x18;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x47;
```

Sedangkan untuk menggati alamat sensor digunakan program seperti berikut ini. Alamat *default* dari modul adalah 0xE0 kemudian diganti menjadi 0xE2.

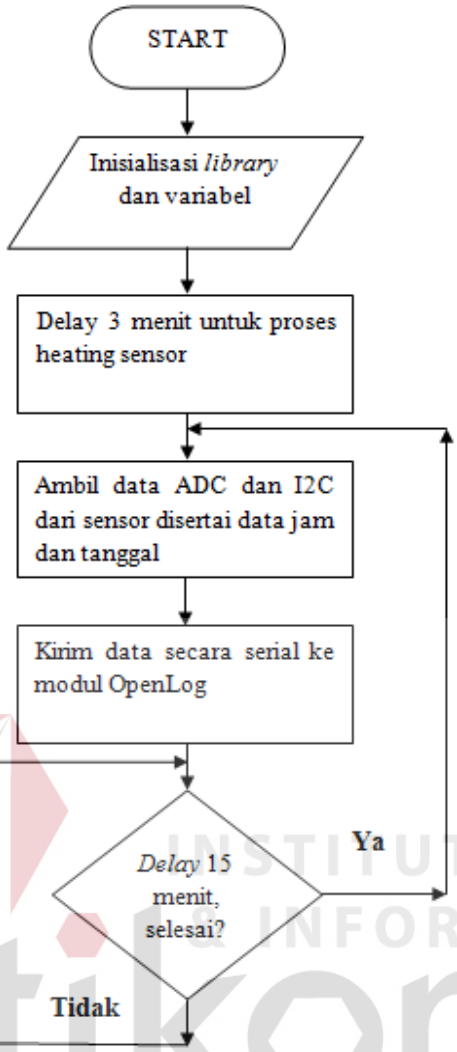
```
while(1)
{
 delay_ms(1000);
 putchar(0x53); //perintah untuk mengubah alamat I2C
 putchar(0xAA); //paramater untuk mengubah alamat I2c
 putchar(0x55); //paramater untuk mengubah alamat I2c
 putchar(0xE2); //alamat baru
 delay_ms(1500);
}
```

3.3. Perangkat Lunak

Perangkat lunak yang digunakan terdiri dari dua jenis. Perangkat lunak yang digunakan untuk mikrokontroler Atmega8535, yang melakukan proses akuisisi data dari sensor-sensor serta *logging* data. Yang kedua, perangkat lunak pada Visual Basic *6.0* yang melakukan proses pembacaan data dengan format file *.TXT* yang merupakan file *logging* dari modul *Openlog*.

3.3.1. Program Mikrokontroler

Untuk melakukan proses *compile* program pada mikrokontroler Atmega8535 digunakan program *CVAVR 2.053*. Diagram alir yang menjelaskan alur program mikrokontroler dengan *CVAVR* dijelaskan pada Gambar 3.19.



Gambar 3.19 Diagram alir *CVAVR*

Dalam melakukan pemrograman dengan *CVAVR* yang harus dilakukan pertama kali adalah inisialisasi *library* dan variabel yang digunakan. Penggunaan *library* diimplementasikan dengan menuliskan *#include* disertai *header* pada awal program. Berikut ini daftar inisialisasi *library* dan variabel yang digunakan.

```
#include <mega8535.h>
#include <ds1307_twi.h>
#include <delay.h>
#include <i2c.h> // I2C Bus functions
#include <alcd.h> // Alphanumeric LCD functions
#include <stdio.h> // Standard Input/Output functions
#include <stdlib.h>
#define ADC_VREF_TYPE 0x00
#include <twi.h> // TWI functions
```

Sensor gas memerlukan waktu *heating* ketika pertama kali diberi sumber daya agar data yang keluar nilainya stabil. Oleh karena itu, perlu diberikan waktu untuk proses *heating* pada sensor gas dengan cara memberikan waktu *delay* selama tiga menit. Perintah untuk memberikan waktu untuk *heating* ini hanya dieksekusi satu kali ketika mikrokontroler pertama kali menerima sumber daya, oleh karena itu pada CVAVR program tersebut dituliskan sebelum *while(1)*. Berikut program untuk memberikan waktu *heating* pada sensor gas.

```
//-----------Heating-----------//
delay_ms(60000);
delay_ms(60000);
delay_ms(60000);
// Global enable interrupts
#asm("sei")
```

Setelah proses inisialisasi variabel, *library* dan perintah *heating* selesai, maka langkah selanjutnya adalah membaca data dari modul sensor gas dengan menggunakan komunikasi I2C. Berikut ini perintah untuk membaca data dari modul sensor gas metana dan karbon dioksida.

```
//Karbon Dioksida
i2c_start(); // Start Condition
i2c_write(0xE0); // Write to DT-SENSE module
i2c_write(0x41); // “Read Sensor” Command
i2c_stop(); // Stop Condition

delay_us(10); // 10 us delay

i2c_start(); // Start Condition
i2c_write(0xE1); // Read from DT-SENSE module
temp1 = i2c_read(1); // Data Sensor MSB
temp2 = i2c_read(0); // Data Sensor LSB
i2c_stop(); // Stop Condition

sensor = (temp1 * 256) + temp2 ;
itoa(sensor,tampil1);

delay_ms(10);

//Metana
i2c_start(); // Start Condition
i2c_write(0xE2); // Write to DT-SENSE module
i2c_write(0x41); // “Read Sensor” Command
i2c_stop(); // Stop Condition
```

```
delay_us(10); // 10 us delay

i2c_start(); // Start Condition
i2c_write(0xE3); // Read from DT-SENSE module
temp14 = i2c_read(1); // Data Sensor MSB
temp24 = i2c_read(0); // Data Sensor LSB
i2c_stop(); // Stop Condition

sensor4 = (temp14 * 256) + temp24 ;
itoa(sensor4,tampil4);
delay_ms(10);
```

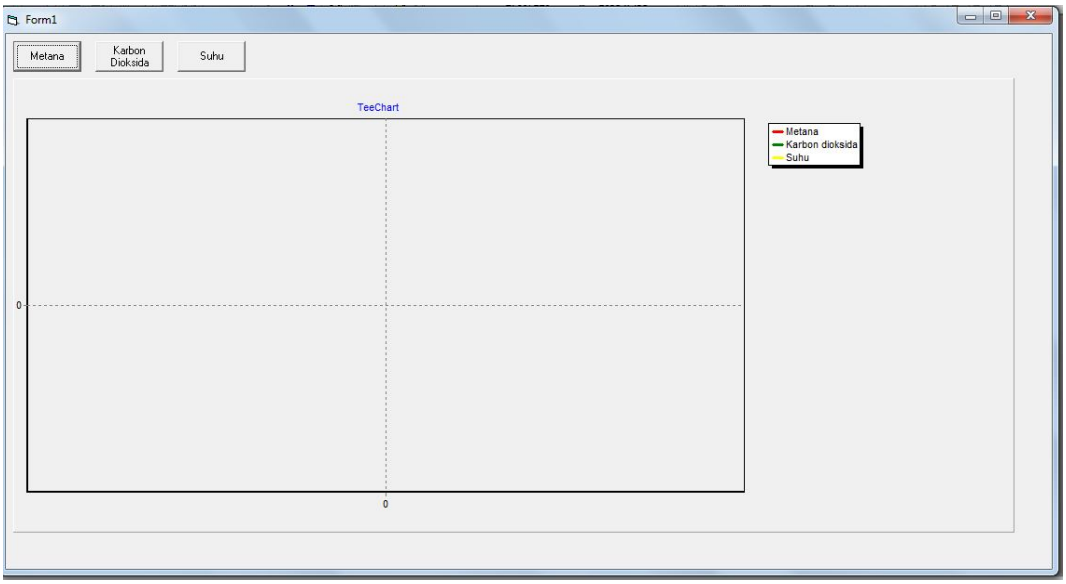
Setelah proses pengiriman data ke *OpenLog* secara serial selesai maka langkah selanjutnya adalah membuat waktu tunda selama 15 menit. Berikut adalah perintah yang digunakan untuk membuat waktu tunda selama 15 menit.

```
for(i=1;i<=15;i++)   //delay 15 menit
{
  delay_ms(60000);
}
delay_ms(100);
```

3.3.2. Program Visual Basic 6.0

Mikrokontroler mengirim data hasil akuisisi sensor-sensor kemudian mengirimkannya secara serial ke modul *OpenLog*. Modul *OpenLog* kemudian menuliskan data-data yang dikirimkan tersebut ke dalam file dengan format *.TXT*. Nama file tersebut menjadi SEQLOG00.TXT karena dalam mode *sequential logging*. Kemudian, ratusan data dalam fomat *.TXT* tersebut dianalisa untuk dapat menemukan pola tertentu sehingga dapat diambil kesimpulan. Tampilan utama menu dari program analisis data yang dibuat dengan Visual Basic *6.0* dapat dilihat pada Gambar 3.20.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA



Gambar 3.20 Tampilan progam Visual Basic *6.0*

Tujuan analisis data dengan Visual Basic *6.0* adalah membaca isi file .TXT dan menampilkannya dalam bentuk grafik (*TeeChart*). Yang pertama kali yang harus dilakukan adalah membuka file .TXT tersebut untuk kemudian dibaca isinya. Setelah itu dilakukan perulangan untuk membaca isi dari file tersebut per karakter sampai menemukan *end of file* (EOF). Isi dari file dibaca sampai didapatkan jenis data yang dimaksud dengan cara menghitung jumlah '–' (strip) yang merupakan pemisah data yang satu dengan data yang lain. Setelah semua proses selesai sampai ditemukan EOF maka yang harus dilakukan adalah memberikan perintah untuk menutup file.TXT yang dibuka. Hal ini merupakan sintaks dari operasi pembacaan file dengan Visual Basic *6.0*.