

BAB II

KAJIAN PUSTAKA

2.1 Gambaran Umum *Nebulizer Compressor*

Nebulizer Compressor merupakan salah satu alat dalam bidang kesehatan, alat ini berfungsi untuk mengurangi dampak yang diterima pada penderita masalah saluran pernafasan seperti asma, batuk atau PPOK (Penyakit Paru-Paru Obstruksi Kronis), karena keuntungan dari alat ini mengubah obat cair menjadi uap (*aerosol*) lalu dihirup, sehingga bagi penderita masalah saluran pernafasan sangatlah tepat dan efektif menggunakan alat ini karena obat langsung masuk ke paru-paru.[12] Obat-obat cair yang digunakan banyak jenis dan fungsinya, yaitu sebagai berikut :

1. *NaCL* dan *Bisolvon* cair

Berfungsi untuk mengencerkan dahak.

2. *Atroven* dan *Berotex*

Berfungsi untuk melonggarkan saluran pernafasan.

3. *Inflamid*

Berfungsi untuk anti radang.

4. *Combiven* dan *Meptin*

Berfungsi untuk melonggarkan saluran pernafasan.

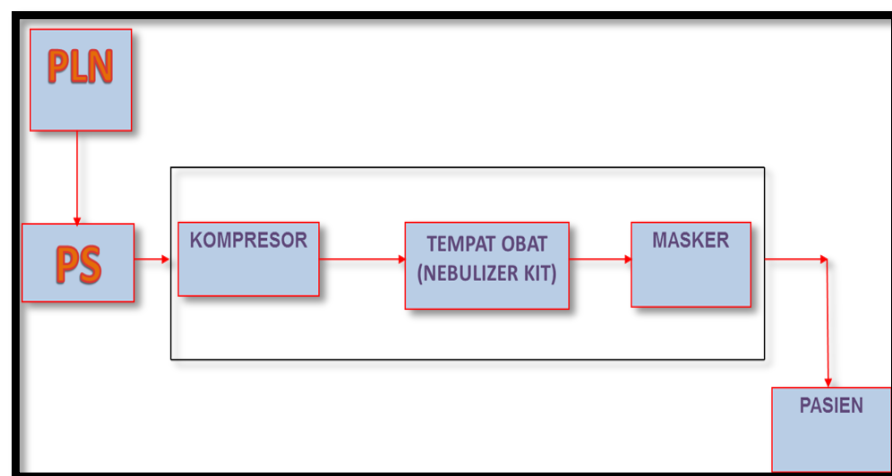
2.1.1 Prinsip Dasar

Prinsip dasar alat ini memanfaatkan proses pengkabutan (*nebuliza*) yaitu proses partikel air yang pecah menjadi kabut karena adanya tekanan udara tinggi yang menekan partikel air. Tetapi dalam

menggunakan alat *Nebulizer Compressor*, tidak menggunakan air melainkan diganti dengan obat cair yang sudah memiliki fungsi masing-masing.[12]

2.1.2 Blok Diagram

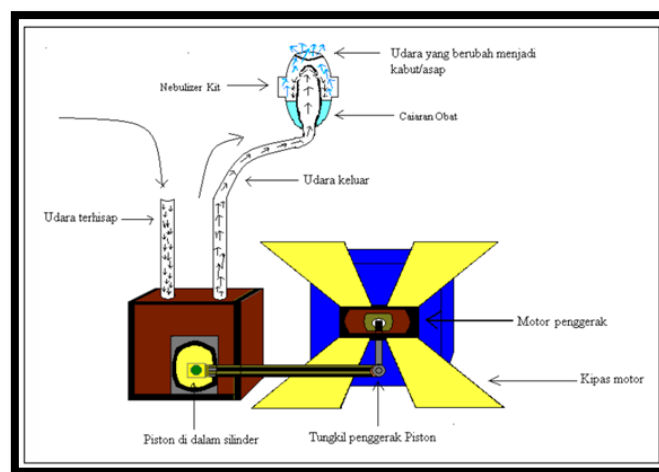
Nebulizer Compressor memiliki blok diagram yang sangat sederhana, penggambarannya seperti dibawah ini :



Gambar 2.1. Blok Diagram *Nebulizer Compressor*

Cara kerja blok diagram alat *Nebulizer Compressor* yaitu ketika motor *compressor* bekerja yang mengakibatkan kipas motor bergerak, dan ditengah kipas terdapat tungkil, dimana ketika motor berputar maka tungkil tersebut akan ikut bergerak mengikuti arah berputarnya motor sehingga tungkil dapat menggerakkan piston bergerak maju dan mundur. Piston berada didalam sebuah silinder yang terdapat dua buah lubang yang berguna untuk menghisap udara dari luar dan untuk mengeluarkannya. Lubang udara yang keluar

dihubungkan dengan selang udara yang nantinya akan masuk ke wadah obat cair (*Nebulizer Kit*). Kemudian udara yang masuk ke dalam *Nebulizer Kit* dimampatkan dan dikeluarkan melewati celah-celah kecil sehingga tekanannya menjadi tinggi dan obat cair yang berada didalam *nebulizer kit* tertekan sehingga molekul-molekul cairan obat menjadi pecah dan berubah menjadi kabut.[12]



Gambar 2.2. Cara kerja *Nebulizer Compressor*

2.2 Perbandingan Alat

Perbandingan alat bertujuan untuk mengetahui dan memahami kelebihan dan kekurangan alat yang pernah dirancang bangun sebelumnya, agar sebagai acuan dalam memodifikasi alat. Perbandingan alat yang menurut saya tepat yaitu sebagai berikut :

2.2.1 *Nebulizer Compressor* Berbasis Mikrokontroler AT89S52

Referensi ini dari hasil laporan Tugas Akhir yang disusun oleh Alif Istiaji Nugroho (07.01.008), Teknik Elektromedik, Politeknik

Muhammadiyah Yogyakarta, 2010. Alat ini menggunakan sistem mikrokontroler *AT89S52* dengan tampilan *Seven segment* dan *Driver Motor ULN 2803*.

Tabel 2.1. Perbandingan alat modifikasi dengan alat lain

NO	NAMA ALAT	KELEBIHAN	KEKURANGAN
1	<i>Nebulizer Compressor Type KQW-4A Berbasis ATmega 328</i>	<p><i>Software IDE Arduino Uno</i>, karena <i>downloader</i> telah tertanam pada <i>board Arduino Uno</i> sehingga tidak perlu rangkaian <i>dowloader</i> tambahan dari luar.</p> <p><i>LCD</i>, karena tampilan menjadi lebih mudah dilihat dengan aneka ragam warna cerah dan tidak terlalu mengurangi tempat/<i>box</i>.</p> <p><i>Keypad</i>, karena mudah dalam penggunaannya hanya dengan menekan tombol atau angka yang diinginkan dengan melihat gambar pada <i>keypad</i>.</p>	<p><i>Tidak adanya Setting Timer</i>, karena belum adanya pengatur waktu.</p> <p><i>Tidak adanya Pengatur tekanan udara</i> sehingga tekanan udara tidak dapat diatur lebih tinggi atau rendah.</p>

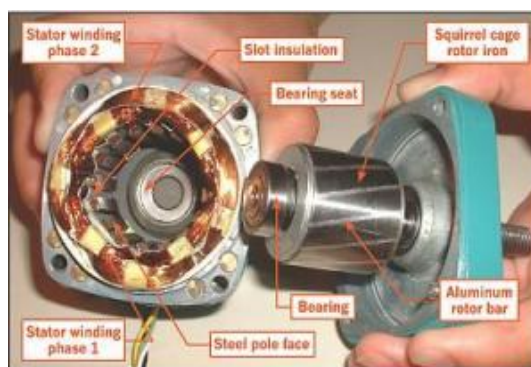
2	Nebulizer Compressor Berbasis AT89S52	Driver Motor ULN 2803 , karena untuk mengontak aktif atau non-aktifkan motor, tegangan stabil. Seven segment.	Setting waktu , karena pengatur waktunya telah diseting tetap, oleh sebab itu tidak dapat dirubah sesuai keinginan. Tidak adanya Setting ukuran obat.
---	--	--	--

2.3 Motor Kompresor

Jantung dari alat *Nebulizer Compressor* adalah motor yang berfungsi untuk menggerakkan kipas pada motor, dan ditengah kipas terdapat tungkil, dimana ketika motor berputar maka tungkil tersebut akan ikut bergerak mengikuti arah berputarnya motor sehingga tungkil dapat menggerakkan piston bergerak maju dan mundur. Motor kompresor ini menggunakan motorr induksi satu *fase*, dimana *fluks magnet* hanya berganti-ganti arah mengakibatkan motor sukar sewaktu akan dijalankan. Untuk memperbesar daya bagi perputaran motor sewaktu dijalankan medan magnet baru yang berbeda *fase* dengan medan magnet utama. Dalam hal ini, berarti harus ada aliran listrik baru yang tidak *se-fase* dengan aliran listrik yang mengalir pada kumparan utama. Untuk membentuk dua buah arus listrik yang berbeda *fase*, menggunakan rangkaian kumparan induktor atau kapasitor secara seri pada kumparan bantu.[3] Di bawah ini gambar dan bagian-bagian dari motor[11] :



Gambar 2.3. Motor Kompresor



Gambar 2.4. Bagian-bagian Motor

2.3.1 *Rotor* adalah bagian yang bergerak/berputar. Bagian-bagian *rotor* yaitu sebagai berikut :

1. Poros Jangkar (*Shaft*) adalah titik pusat putaran dari jangkar dan sebagai penghubung secara mekanis mesin dengan piranti luar.
2. Inti Jangkar (*Rotor Core*) adalah bagian berputar yang digunakan untuk membawa belitan jangkar.

3. Belitan Jangkar (*Rotor Windings*) adalah sekelompok kumparan yang saling dihubungkan sehingga menghasilkan tegangan yang diinginkan.
4. Kipas Pendingin (*Cooling Fan*) adalah bagian yang berfungsi sebagai pendingin mesin.

2.3.2 *Stator* adalah bagian yang diam/statis. Bagian-bagian *stator* yaitu sebagai berikut :

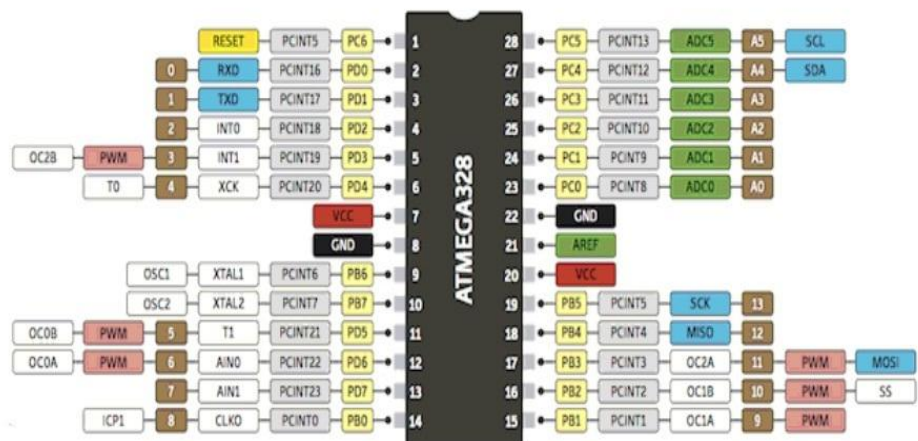
1. Kerangka Mesin (*Frame*) adalah bagian yang berfungsi membawa *flux* magnet yang diproduksi oleh kutub.
2. Sepatu Kutub (*Field Pole*) adalah bagian yang berfungsi untuk mengarahkan medan magnet utama agar menyebar sesuai dengan bentuk jangkar.
3. Bantalan Poros (*Ball Bearings*) adalah tempat meletakkan poros.
4. Sikat Arang (*Carbon Brush*) adalah bagian yang berfungsi untuk mengumpulkan arus dari *komutator* dan menghubungkan dengan sumber listrik ke beban.
5. *Slip Rings* adalah bagian yang berfungsi untuk menghubungkan belitan/kumparan dengan sikat arang.

2.4 Mikrokontroler

Mikrokontroler merupakan suatu *IC (Integrated Circuit)* yang di dalamnya berisi *CPU, ROM, RAM*, dan *I/O*. Dengan adanya *CPU* tersebut maka mikrokontroler dapat melakukan proses berfikir berdasarkan program yang telah diberikan kepadanya. Mikrokontroler dapat disebut sebagai komputer yang berukuran kecil yang berdaya rendah sehingga sebuah baterai dapat memberikan daya.

2.4.1 Mikrokontroler *ATMega 328*

1. Arsitektur dan Konfigurasi *Pin ATMega 328*



Gambar 2.5. Konfigurasi Pin *ATMega 328*

Mikrokontroler *ATMega 328* memiliki beberapa kriteria standar yaitu memiliki *32 KB Flash Programmable* dan *1 KB EEPROM* yang dapat di program ulang sekitar *1000 kali write* atau *erase cycle*, *2 KB SRAM*, *14 jalur I/O*, *6 pin analog*, dua buah *16 bit timer/counter*, dengan arsitektur *4 level interrupt*, *serial port*,

on-chip oscillator dan *on-chip timer/counter*. Beberapa *pin-pin* *ATMega 328* :

a. *PORT B (PB7-PB0)*

PORT B adalah *port I/O* dua arah (*bidirectional*) 8-bit dengan *resistor pull up internal* yang dapat dipilih. Ketika digunakan sebagai *input*, *pin* yang *pull down* secara eksternal akan memancarkan arus jika *resistor pull up*-nya diaktifkan. *Pin-pin PORT B* akan berada pada kondisi *tri-state* ketika *Reset* aktif meskipun *clock*, tidak akan *running*.

b. *Port C (PC5-PC0)*

PORT C adalah *port I/O* dua arah (*bidirectional*) 8-bit dengan *resistor pull up internal* yang dapat dipilih. Ketika digunakan sebagai *input*, *pin* yang di *pull down* secara eksternal akan memancarkan arus jika *resistor pull up*-nya diaktifkan. *Pin-pin PORT C* akan berada pada kondisi *tri-state* ketika *Reset* aktif meskipun *clock*, tidak akan *running*. Jika *Fuse RSTDISBL* diprogram, maka *PC6* berfungsi sebagai *pin I/O* akan tetapi dengan karakteristik yang berbeda dengan *PC5-PC0*. Jika *Fuse RSTDISBL* tidak diprogram, maka *PC6* berfungsi sebagai masukan *Reset*.

c. *Port D (PD7-PD0)*

PORT D adalah *port I/O* dua arah (*bidirectional*) 8-bit dengan *resistor pull up internal* yang dapat dipilih. Ketika

digunakan sebagai *input*, *pin* yang di *pull down* secara eksternal akan memancarkan arus jika resistor *pull up*-nya diaktifkan. *Pin-pin PORT D* akan berada pada kondisi *tri-state* ketika *Reset* aktif meskipun *clock*, tidak akan *running*.

d. *Reset*

Pin Reset berfungsi untuk *me-restart* program yang diletakkan pada *board* dengan sebuah tombol yang berfungsi untuk mengkondisikan awal suatu program. Sinyal *LOW* pada *pin* ini akan membawa mikrokontroler ke kondisi *Reset*, meskipun *clock* tidak *running*.

e. *AREF*

Pin Analog berfungsi untuk tegangan *input* analog atau *ADC*.

f. *VCC (Voltage Common Colector)*

Pin pemberi tegangan positif (+) yang besar tegangan sekitar 4,5volt-5,5 volt.

g. *GND (Ground)*

Pin pemberi tegangan negatif (-).

h. *TXD* dan *RXD*

Merupakan jalur data komunikasi serial. *TX (Transmitter)* berfungsi untuk mengirim data serial dan *RX (Rectifier)* berfungsi untuk menerima data serial.

i. *INT (INT0 dan INT1)*

Interrupt merupakan *pin* dengan fungsi khusus untuk interupsi *hardware*. Interupsi biasanya digunakan sebagai selaan dari program, misalkan pada saat program berjalan kemudian terjadi interupsi *hardware/software* maka program utama akan berhenti dan akan menjalankan program interupsi.

j. *XCK (External Clock)*

Merupakan *pin* yang dapat berfungsi sebagai sumber *clock external* untuk *TX* dan *RX*.

k. *XTAL (XTAL dan XTAL2)*

Merupakan *pin* yang berfungsi sebagai sumber *clock* utama mikrokontroler.

l. *TOSC (Timer Oscillator 1 dan 2)*

Merupakan *pin* dapat berfungsi sebagai sumber *clock* eksternal untuk *timer*.

m. *T0 dan T1 (Timer)*

Pin yang berfungsi sebagai masukan *counter* eksternal untuk *timer* 0 dan *timer* 1.

n. *AIN (Analog Comparator Negative)*

Merupakan *pin* yang berfungsi sebagai masukan *input* untuk analog *comparator*.

m. *ICP (Input Capture Pin)*

Pin yang berfungsi sebagai Timer Counter

o. *SCK (Serial Clock)*

Merupakan *pin* yang berfungsi sebagai jalur komunikasi pemrograman serial.

p. *MISO (Master Input Slave Output)*

Merupakan *pin* yang berfungsi sebagai jalur komunikasi pemrograman serial.

q. *MOSI (Master Output Slave Input)*

Merupakan *pin* yang berfungsi sebagai jalur komunikasi pemrograman serial.

r. *SS (Slave Select)*

Merupakan *pin* yang berfungsi sebagai jalur komunikasi pemrograman serial.

s. *OC (Output Compare 1A, 1B dan 2)*

Merupakan *pin* yang dapat berfungsi sebagai keluaran *PWM (pulse width modulation)*.

t. *AVCC*

Pin pemberi tegangan untuk *ADC, PC3-PC0, dan ADC7-ADC6*. *Pin* ini harus dihubungkan dengan *VCC*, meskipun *ADC* tidak digunakan. Jika *ADC* digunakan, *VCC* harus dihubungkan ke *AVCC* melalui *low pass filter* untuk

mengurangi *noise*. *ADC7-ADC6*, *ADC* (*Analog Digital Converter*) adalah analog *input*.

2.5 *Arduino Uno*

Arduino Uno adalah pengendali *single board* mikrokontroler yang bersifat *open source*, yang dirancang untuk memudahkan penggunaan elektronik dalam berbagai bidang. Perangkat kerasnya memiliki prosesor *atmel AVR* (*Alf and Vegard's Risc Processor*) dan perangkat lunaknya memiliki bahasa pemrograman sendiri. *Arduino uno*, dapat diartikan juga papan rangkaian elektronik *open source* yang didalamnya terdapat komponen utama yaitu sebuah *chip* mikrokontroler dengan jenis *AVR*. Mikrokontroler itu sendiri adalah *chip* atau *IC* (*integrated Circuit*) yang bisa diprogram menggunakan komputer. Mikrokontroler berfungsi sebagai otak yang mengendalikan *input*, proses *input*, dan *output* sebuah rangkaian elektronik.

2.5.1 *Software*

Arduino Uno dibuat untuk para pemula bahkan yang tidak memiliki dasar bahasa pemrograman sama sekali karena menggunakan bahasa *C++* yang telah dipermudah dengan *library*. *Arduino Uno* menggunakan *software processing* yang digunakan untuk menulis program. *Processing* sendiri merupakan penggabungan antara bahasa *C++* dan *Java*. *Software* ini terdiri dari 3 bagian yaitu sebagai berikut :

1. *Editor programmer*

Modul yang berfungsi untuk menulis dan mengubah program dalam bahasa *processing*. *Listing* program pada *Arduino Uno* disebut *Sketch*.

2. *Compiler*

Modul yang berfungsi untuk mengubah bahasa kode program (*processing*) kedalam kode *biner* karena kode *biner* adalah satu-satunya bahasa program yang dipahami mikrokontroler.

3. *Uploder*

Modul yang berfungsi untuk memasukkan kode *biner* kedalam memori mikrokontroler.

Struktur perintah pada *Arduino Uno* secara garis besar terdiri dari 2 bagian yaitu sebagai berikut :

1. *Void setup*

Berisi perintah yang akan dijalankan hanya satu kali saja sejak *Arduino Uno* dihidupkan.

2. *Void loop*

Berisi perintah yang akan berjalan berulang-ulang selama *Arduino Uno* dinyalakan.

```
void setup() {  
    // perintah-perintah untuk  
    konfigurasi dan inisialisasi  
    arduino board  
}  
  
void loop() {  
    // perintah - perintah utama  
    arduino board  
}
```

Gambar 2.6. Program *Void setup* dan *Void loop*

Baik *void setup* *loop* () maupun *function* harus diberi tanda kurung kurawal buka “{” sebagai tanda awal program dan kurung kurawal tutup “}” sebagai tanda akhir program. Pada proses *Uploader*, dimana pada proses ini mengubah bahasa pemrograman yang nantinya di *compile* oleh *avr-gcc* (*avr-gcc compiler*) yang hasilnya akan disimpan kedalam papan *Arduino Uno*. *Avr-gcc compiler* merupakan suatu bagian penting untuk *software* bersifat *open source*. Dengan adanya *avr-gcc compiler*, maka akan membuat bahasa pemrograman dapat dimengerti oleh mikrokontroler. Proses terakhir ini sangat penting, karena dengan adanya proses ini maka akan membuat proses pemrograman mikrokontroler menjadi sangat mudah.

2.5.2 Hardware



Gambar 2.7. Papan Arduino Uno

Single board mikrokontroler yang bersifat *open source* yang dikembangkan untuk mikrokontroler *AVR 8 bit* dan *ARM 32 bit* seperti *ATMega 8*, *ATMega 168*, *ATMega 328*, *ATMega 1280*, dan *ATMega 2560* dengan menggunakan kristal osilator *16 MHz*, namun ada beberapa tipe *Arduino* yang lain menggunakan kristal osilator *8 MHz*. *Hardware* pada *Arduino Uno* terdiri dari 20 bagian yaitu sebagai berikut :

1. 14 *Pin I/O* Digital (*Pin 0-13*)

Sejumlah *pin* digital dengan nomor 0-13 dapat dijadikan *input* atau *output*. Pada saat *input* ataupun *output*, terdapat kondisi 2 yaitu bila mendapat tegangan *5 volt*, *pin* akan *High* (nyala) dan *Low* (padam) jika mendapat *0 volt*, contohnya seperti menggunakan suatu sensor. Pada *Arduino Uno* terbagi menjadi 3 program yaitu sebagai berikut :

a. *pinMode* ()

pinMode () digunakan dalam *void setup* () untuk menentukan *pin* sebagai *Input* atau *Output*. Istilah *pin* mengartikan angka dari *pin* digital yang akan digunakan sedangkan *mode* mengartikan pilihan *input* atau *output*.

```
pinMode (pin, Output);    // mengset pin
sebagai output
```

Gambar 2.8. Program *pinMode* ()b. *digitalRead* ()

digitalRead () digunakan untuk membaca nilai *pin* yang telah ditentukan dengan hasil *High* atau *Low*.

```
int ledPin = 13; // LED terhubung ke pin digital 13
int inPin = 7; // pushbutton terhubung ke pin digital 7
int val = 0; // variable untuk menyimpan sebuah nilai
void setup() {
  pinMode(ledPin, OUTPUT); // set pin digital 13 sebagai
keluaran
  pinMode(inPin, INPUT); // set pin digital 13 sebagai masukan
}
void loop(){
  val = digitalRead(inPin); // baca nilai pin input
```

Gambar 2.9. Program *digitalRead* ()

c. *digitalWrite ()*

digitalWrite () digunakan untuk menulis *pin* digital.

```
digitalWrite (pin, High); // set
pin to High
```

Gambar 2.10. Program *digitalWrite ()*

2. 6 *Pin Input Analog (Pin 0-5)*

Sejumlah *pin* analog bernomor 0-5 yang digunakan untuk membaca nilai *input*, memiliki nilai analog yang dapat mengubah kedalam angka antara 0 dan 1023. Pada *Arduino* terbagi menjadi 2 program yaitu sebagai berikut :

a. *analogRead ()*

analogRead () digunakan untuk membaca nilai *pin* analog.

Fungsi ini hanya bekerja pada *pin* analog (0-5).

```
Value = analogRead(pin); // mengset
'value' sama dengan nilai analog pin
```

Gambar 2.11. Program *analogRead ()*

b. *analogWrite ()*

analogWrite () digunakan untuk menulis nilai *pin* analog.

```
analogWrite(pin, value); // menulis
ke pin analog
```

Gambar 2.12. Program *analogWrite ()*

3. 6 Pin Output Analog (Pin 3, 5, 6, 9, 10 dan 11)

Sejumlah *pin* yang sebenarnya merupakan *pin* digital tetapi sejumlah *pin* tersebut dapat diprogram kembali menjadi *pin output* analog.

2.5.3 Socket USB

Socket USB adalah *socket* kabel *USB* yang disambungkan ke komputer atau laptop. Yang berfungsi untuk mengirimkan program ke *Arduino Uno* dan juga sebagai *port* komunikasi serial. Kelebihan *Arduino* tidak membutuhkan *flash programmer* eksternal karena didalam *chip* mikrokontroler telah diisi dengan *bootloader* yang membuat proses *upload* menjadi lebih sederhana. Jadi untuk koneksi pada komputer/laptop dapat menggunakan *RS232 to TTL converter* atau *chip* *USB* ke serial *converter* seperti *FTDI FT232*.

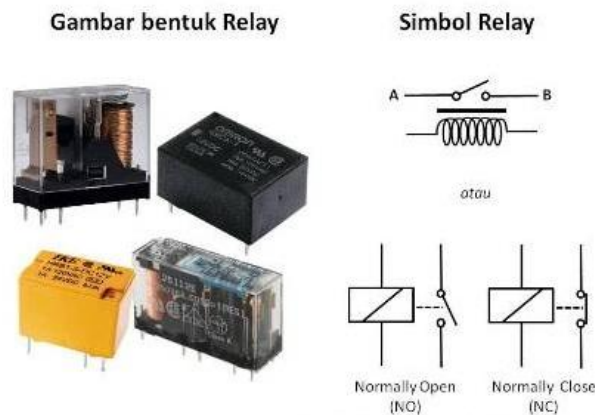
2.5.4 Catu Daya

Pin catu daya adalah *pin* yang memberikan tegangan untuk komponen atau rangkaian yang dihubungkan dengan *Arduino Uno*. Pada bagian catu daya ini *pin* *Vin* dan *Reset*. *Vin* digunakan untuk memberikan tegangan langsung kepada *Arduino Uno* tanpa melalui tegangan pada *USB* atau *adaptor*, sedangkan *Reset* adalah *pin* untuk memberikan sinyal *reset* melalui tombol atau rangkaian eksternal. Catu daya yang diperlukan untuk memberi tegangan pada minimum sistem cukup dengan tegangan 5 *VDC*.

2.5.5 Input-Output Digital dan Input Analog

Input-output digital atau *pin* digital adalah *pin* untuk menghubungkan *Arduino Uno* dengan komponen atau rangkaian digital. Contohnya jika ingin membuat *LED* berkedip maka *LED* tersebut bisa dipasang pada salah satu *pin input* atau *output* digital dan *ground*. Komponen lain yang menghasilkan *output* digital atau menerima *input* digital bisa disambungkan ke *pin* ini. *Input* analog atau analog *pin* adalah *pin* yang berfungsi untuk menerima sinyal dari komponen atau rangkaian analog. Contohnya, *sensor* suhu, *sensor* cahaya, dll.

2.6 Relay



Gambar 2.13. Bentuk dan Simbol *Relay*

Relay adalah saklar yang dioperasikan dengan listrik dan merupakan komponen yang terdiri dari 2 bagian utama yakni Elektromagnet (*Coil*) dan kontak saklar (*Mechanical*). *Relay* menggunakan prinsip elektromagnetik untuk menggerakkan kontak saklar sehingga dengan arus listrik yang kecil

dapat menghantarkan listrik yang bertegangan lebih tinggi. Penggunaan *relay* perlu memperhatikan tegangan pengontrolnya serta kekuatan *relay* mengontak arus/tegangan. Biasanya ukurannya tertera pada *body relay*, misalnya *relay 12 VDC/4A 220V*, artinya tegangan yang diperlukan sebagai pengontrolnya adalah *12 Volt DC* dan mampu mengontak arus listrik maksimal sebesar *4 Ampere* pada tegangan *220 Volt*. [9]

2.6.1 Fungsi *Relay*

Beberapa fungsi umum yang telah banyak digunakan kedalam peralatan elektronika yaitu sebagai berikut :

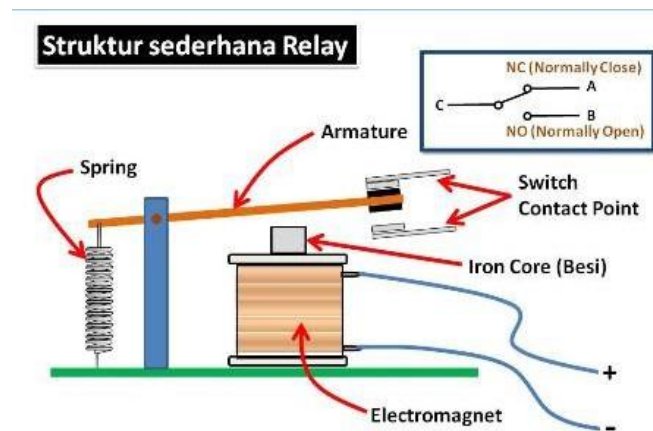
1. Untuk menjalankan fungsi logika (*Logical Function*).
2. Untuk memberikan fungsi penundaan waktu (*Time Delay Function*).
3. Untuk melindungi motor ataupun komponen lain dari kelebihan tegangan maupun hubungan singkat (*Short*).

2.6.2 Prinsip Kerja *Relay*

Komponen dasar pembentuk sebuah *relay* sederhana ada 4 yaitu sebagai berikut :

1. Elektromagnet (*Coil*)
2. *Armature*
3. *Switch Contact Point* (Kontak Saklar)
4. *Spring*

Penggambaran struktur *relay* sederhana yaitu sebagai berikut :



Gambar 2.14. Struktur sederhana *Relay*

Kontak Saklar *relay* terdiri dari 2 jenis yaitu sebagai berikut :

1. NC (*Normally Close*)

Normally Close adalah kondisi awal sebelum diaktifkan akan selalu berada diposisi tertutup (*Close*).

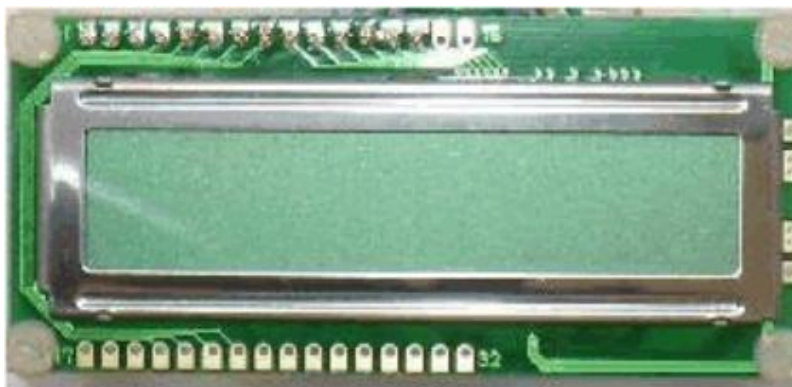
2. NO (*Normally Open*)

Normally Open adalah kondisi awal sebelum diaktifkan akan selalu berada diposisi terbuka (*Open*).

Berdasarkan gambar diatas, prinsip kerja *relay* yaitu sebuah besi (*Iron Core*) yang dililit oleh kumparan *coil* yang berfungsi untuk mengendalikan besi tersebut. Apabila kumparan *coil* diberikan arus listrik, maka akan timbul gaya elektromagnet yang kemudian menarik *armature* untuk berpindah dari posisi sebelumnya (*Normally Close*) ke posisi baru (*Normally Open*) sehingga menjadi saklar yang dapat menghantarkan arus listrik. Pada saat tidak dialiri listrik, *armature*

akan kembali lagi ke posisi awal (*Normally Close*). *Coil* membutuhkan arus listrik yang relatif kecil untuk mengaktifkan elektromagnet dan menarik kontak saklar ke posisi *close*.

2.7 *Liquid Crystal Display*



Gambar 2.15. *LCD* Karakter 16x2

Liquid Crystal Display (*LCD*) adalah komponen yang dapat menampilkan berbagai macam karakter. *LCD* pada umumnya memiliki berbagai jenis tapi yang digunakan saat ini yaitu *LCD* karakter 16x2, dengan lebar tampilan 2 baris 16 kolom dan memiliki 16 *pin* konektor.[5] Fungsi dari ke 16 *pin* tersebut yaitu sebagai berikut :

Tabel 2.2. Fungsi-fungsi *pin LCD*

PIN	NAME	FUNCTION
1	VSS	Ground Voltage
2	VCC	5+ Voltage
3	VEE	Contrast Voltage
4	RS (Register Select)	0 = Register Select 1 = Data Register
5	R/W (Read/Write)	0 = Write Mode 1 = Read Mode

6	<i>Enable</i>	<i>0 = Start to data LCD character 1 = Disable</i>
7	<i>DB0</i>	<i>Data Bus 0</i>
8	<i>DB1</i>	<i>Data Bus 1</i>
9	<i>DB2</i>	<i>Data Bus 2</i>
10	<i>DB3</i>	<i>Data Bus 3</i>
11	<i>DB4</i>	<i>Data Bus 4</i>
12	<i>DB5</i>	<i>Data Bus 5</i>
13	<i>DB6</i>	<i>Data Bus 6</i>
14	<i>DB7</i>	<i>Data Bus 7</i>
15	<i>BPL</i>	<i>Black Panel Light</i>
16	<i>GND</i>	<i>Ground Voltage</i>

Tampilan karakter pada *LCD* diatur oleh *Pin EN*, *RS* dan *R/W*. Penjelasan yaitu jalur *EN* dinamakan *Enable*, jalur ini digunakan untuk memberitahu *LCD* bahwa anda sedang mengirimkan sebuah data. Untuk mengirimkan data ke *LCD*, maka program *EN* harus dibuat logika *LOW* “0” dan atur pada dua jalur kontrol yang lain yaitu *RS* dan *R/W*.

Jalur *RS* adalah jalur *Register Select*, jalur ini digunakan untuk menentukan jenis data yang masuk. Ketika *RS* berlogika *LOW* “0”, data akan dianggap sebagai sebuah perintah seperti *clear screen*, posisi kursor dll. Dan ketika *RS* berlogika *HIGH* “1”, data yang dikirim adalah data teks yang akan ditampilkan pada layar *LCD*. Sebagai sebuah contoh, untuk menampilkan huruf “T” pada layar *LCD* maka *RS* harus diatur logika *HIGH* “1”.

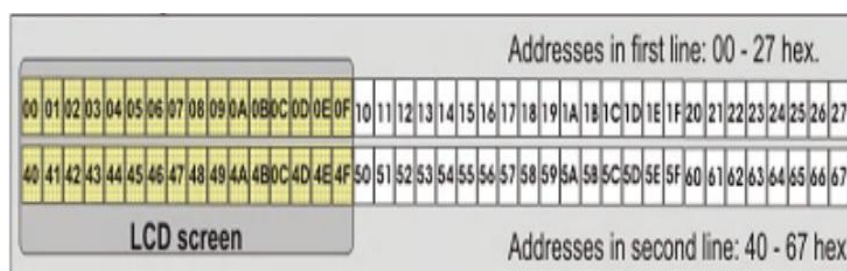
Jalur *R/W* adalah jalur kendali *Read/Write*. Ketika *R/W* berlogika *LOW* “0”, maka informasi pada data *bus* akan tertuliskan pada layar *LCD*. Ketika *R/W* berlogika *HIGH* ”1”, maka program akan melakukan

pembacaan memori dari *LCD*. Sedangkan pada aplikasi umum *pin R/W* selalu diberi logika *LOW* "0".[13]

2.7.1 Memory LCD

1. *DDRAM*

Display Data Random Access Memory (DDRAM) merupakan memori untuk menyimpan karakter yang akan ditampilkan.



Gambar 2.16. Peta Memori *LCD* Karakter 16x2

Pada peta memori tersebut, daerah yang berwarna kuning (00 s/d 0F dan 40 s/d 4F) adalah tampilan yang tampak. Jumlah sebanyak 16 karakter per baris dengan dua baris. Angka pada setiap kotak adalah alamat memori yang bersesuaian dengan posisi dari layar.

2. *CGRAM*

Character Generator Random Access Memory (CGRAM) merupakan memori untuk menggambarkan pola sebuah karakter yang dapat dirubah sesuai keinginan.

3. *CGROM*

Character Gennerator Read Only Memory (CGROM) berfungsi untuk menggambarkan pola sebuah karakter dimana pola tersebut adalah karakter dasar yang telah ditentukan secara permanen oleh pabrik sehingga pengguna hanya tinggal mengambil sesuai alamat memori dan tidak dapat merubah karakter dasar *CGROM*.

2.7.2 *Register LCD*

1. *Register Perintah*

Register perintah merupakan *register* yang berisi perintah-perintah dari mikrokontroler ke *LCD* pada saat proses penulisan data.

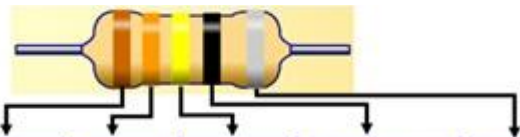
2. *Register Data*

Register data merupakan *register* untuk menuliskan atau membaca data dari atau ke *DDRAM*. Penulisan data pada *register* akan menempatkan data tersebut ke *DDRAM* sesuai dengan alamat yang sudah diatur sebelumnya.

2.8 *Resistor*

Resistor adalah komponen yang berfungsi untuk membatasi jumlah arus yang mengalir dalam suatu rangkaian. Kemampuan *resistor* dalam menghambat arus listrik sangat beragam disesuaikan dengan nilai *resistansi* *resistor* tersebut. *Resistor* bersifat *resistif* dan umumnya terbuat dari bahan karbon. Satuan *resistansi* dari suatu *resistor* disebut *Ohm* atau

dilambangkan dengan simbol Ω (*Omega*). Nilai *resistansi* pada *resistor* dapat diketahui dari warna gelang pada bagian luar *resistor*, setiap warna gelang memiliki harga dan fungsi yang berbeda-beda. Untuk warna gelang 1 dan 2 menunjukkan nilai kemudian 3 menunjukkan faktor kali dan 4 menunjukkan nilai toleransi. Umumnya *resistor* dipasaran kebanyakan hanya memiliki 4 gelang warna tapi ada juga 5 sampai 6, warna gelang 5 dan 6 menunjukkan nilai sensitifitas terhadap suhu sekitarnya. Untuk mengetahui besar nilai masing-masing warna dapat dilihat pada gambar berikut:



Warna	Gelang 1	Gelang 2	Gelang 3	Multiplier	Toleransi
	Gelang 1	Gelang 2	Gelang 3	Gelang 4	Gelang 5
Hitam		0	0	1 Ohm	
Coklat	1	1	1	10 Ohm	$\pm 1\%$
Merah	2	2	2	100 Ohm	$\pm 2\%$
Orange	3	3	3	1 K Ohm	
Kuning	4	4	4	10 K Ohm	
Hijau	5	5	5	100 K Ohm	$\pm 0,5\%$
Biru	6	6	6	1 M Ohm	$\pm 0,25\%$
Ungu	7	7	7	10 M Ohm	$\pm 0,10\%$
Abu-abu	8	8	8		$\pm 0,05\%$
Putih	9	9	9		
Emas				0,1 Ohm	$\pm 5\%$
Perak				0,01 Ohm	$\pm 10\%$

Gambar 2.17. Warna dan Nilai gelang pada *resistor*

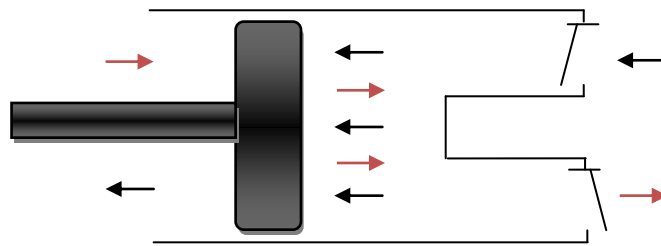
Tabel 2.3 Hasil uji *transistor* dengan *ohm* meter

Kaki I	Kaki II	Kaki III	Gejala
M	H	-	ON
M	-	H	ON
H	M	-	OFF
H	-	M	OFF

Hasil pengujian, ditemukan bahwa kaki I adalah kaki **Base**, dimana selama pengukuran harus ada kaki acuan dan menunjukkan gejala *ON*, *ON* kemudian bila dibalik polaritasnya menunjukkan gejala *OFF*, *OFF* maka kaki basis *ON* pada saat dipasang polaritas negatif atau *OFF* saat dipasang polaritas positif maka jenis *transistor* adalah PNP sedangkan untuk menentukan kaki **Emitter** dan **Collector**, kita harus menghitung nilai hambatan yang dimiliki oleh emitor dan kolektor. Apabila kaki II hambatannya lebih besar dari kaki III maka dapat kita simpulkan bahwa kaki II merupakan kolektor dan kaki III merupakan emitor. Prinsip *transistor* sebagai penghubung (saklar) yaitu ketika *transistor* dioperasikan pada titik jenuh (*Saturation*) atau dalam keadaan tertutup/terhubung, maka arus *basis* harus sama dengan arus *basis saturation*. Jika arus *basis* lebih besar maka *transistor* tetap pada titik jenuh sehingga jika ingin pada titik sumbat (*Cutoff*) atau dalam keadaan terbuka/tidak terhubung maka arus *basis* paling kecil harus sama dengan nol (0). Kondisi tertutup/terhubung yaitu antara kolektor dan emitor terjadi hubungan singkat.[2]

2.10 Kompresor

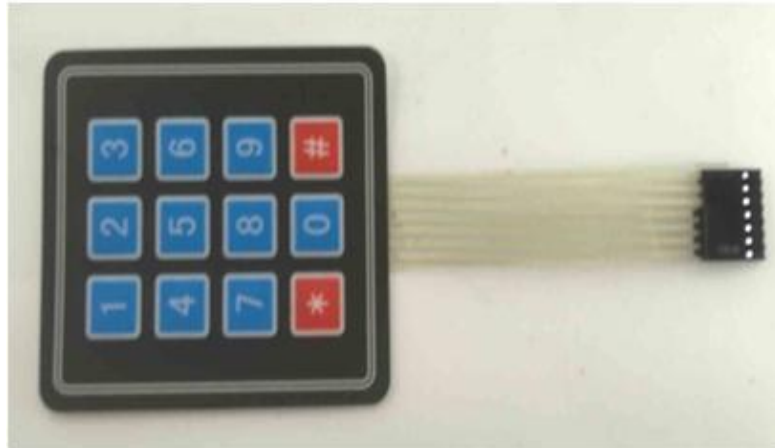
Kompresor torak yang menggunakan sebuah membran (*diafragma*) akan terpisah letak toraknya. Udara yang masuk dan keluar tidak langsung berhubungan. Adanya pemisahan ruangan ini udara akan lebih terjaga dan bebas dari uap air dan pelumas/oli. Oleh karena itu kompresor *diafragma* banyak digunakan pada industri bahan makanan, farmasi, obat-obatan dan kimia.[]



Gambar 2.19. Proses udara masuk-keluar dengan *diafragma*

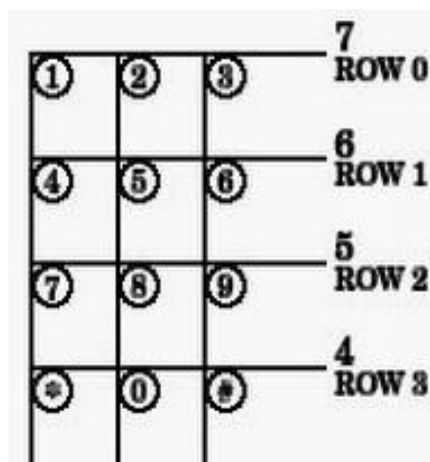
Prinsip kerjanya yaitu torak pada kompresor *diafragma* tidak secara langsung menghisap dan menekan udara, tetapi menggerakkan sebuah membran (*diafragma*) dahulu. Dari gerakan *diafragma* yang kembang Kempis itulah yang akan menghisap dan menekan udara dengan tekanan yang lebih tinggi.

2.11 Keypad



Gambar 2.20. Keypad membrane 4x3

Keypad biasanya digunakan pada beberapa peralatan yang berbasis mikrokontroler. Pada penggunaannya *keypad* terdiri dari beberapa saklar, yang saling terhubung jika dilakukan penekanan pada bagian *keypad* sehingga antara kolom dan baris akan terhubung. Agar mikrokontroler dapat melakukan *scan keypad* harus diberikan logika *LOW* “0” ketika tombol *keypad* tidak ditekan dan logika *HIGH* “1” pada saat tombol *keypad* ditekan. *Keypad membrane* yang digunakan adalah *keypad* dengan jumlah kolom 3 dan jumlah baris 4 yang dapat digunakan, rangkaian *keypad* 4x3 dapat dilihat pada gambar berikut :



Gambar 2.21. Pin Output Keypad membrane 4x3

2.12 Kalibrasi

Secara umum kalibrasi memiliki pengertian sebagai rangkaian kegiatan membandingkan hasil pengukuran suatu alat dengan alat standar yang sesuai untuk menentukan besarnya koreksi pengukuran alat serta ketidakpastiannya. Dalam pengertian ini alat standar yang digunakan juga harus terkalibrasi dibuktikan dengan sertifikat kalibrasi. Dengan demikian maka besarnya koreksi pengukuran alat dapat ditelusurkan ke standar nasional atau standar internasional dengan suatu mata rantai kegiatan kalibrasi yang tidak terputus.

Alat ukur yang telah dikalibrasi tidak akan secara terus menerus berlaku masa kalibrasinya, karena peralatan tersebut selama masa pemakaiannya mengalami perubahan spesifikasi akibat pengaruh frekuensi pemakaian, lingkungan penyimpanan, cara penggunaan, dan sebagainya. Untuk itulah selama berlakunya masa kalibrasi alat bersangkutan perlu dipelihara dengan cara merawat dan cek secara periodik. Rangkaian

kegiatan kalibrasi secara sederhana yaitu kegiatan persiapan kalibrasi, pelaksanaan kalibrasi, perhitungan data kalibrasi, penentuan ketidakpastian dan penerbitan laporan kalibrasi.

2.12.1 Kegiatan persiapan kalibrasi antara lain sebagai berikut :

1. Persiapan alat standar dan alat yang akan dikalibrasi

Alat yang akan dikalibrasi dan alat standar dikondisikan pada kondisi yang sama sesuai metode kalibrasi, hal ini diperlukan untuk menghindarkan perbedaan hasil ukur akibat pengaruh lingkungan.

2. Metode kalibrasi

Metode kalibrasi dapat mengacu kepada metode standar internasional maupun metode standar lainnya misalnya *text book*, jurnal, buletin, dan manual peralatan, namun perlu diperhatikan bahwa acuan tersebut harus merupakan publikasi yang diakui masyarakat luas. Selain itu dari beberapa pilihan metode kalibrasi dapat dipilih metode yang mudah dilaksanakan, karena sulitnya mengikuti metode kalibrasi dapat berakibat kesalahan dalam pengambilan data kalibrasi.

2.12.2 Kegiatan pelaksanaan kalibrasi antara lain sebagai berikut :

1. Pengamatan awal

Jika alat yang dikalibrasi berupa instrumen, pastikan bahwa alat tersebut dapat beroperasi normal. Jika alat berupa objek ukur pastikan bahwa alat mempunyai bentuk sempurna. Pada

prinsipnya pelaksanaan kalibrasi tidak bertujuan untuk memperbaiki alat, karenanya alat yang tidak normal seyogyanya tidak boleh dikalibrasi. Alat demikian harus diperbaiki dulu oleh petugas yang khusus menangani perbaikan alat hingga alat tersebut diyakini beroperasi normal.

2. Penyetelan

Penyetelan alat yang akan dikalibrasi biasanya diperlukan untuk menghindari kesalahan titik nol. Penyetelan dapat berupa menyetel kedataran, pembersihan alat dari kotoran, menyetel titik nol, dalam hal misalnya kalibrasi neraca elektronik penyetelan dapat berupa kalibrasi internal sesuai prosedur dalam manual.

3. Pengamatan kewajaran hasil ukur

Pengamatan ini dimaksudkan untuk memastikan kewajaran penunjukan alat. Jika alat menunjukkan hasil ukur yang tidak wajar, mungkin perlu penyetelan kembali atau perlu dicari penyebab ketidakwajaran penunjukan alat tersebut.

4. Pengukuran

Pengukuran dilakukan pada titik ukur tertentu seperti dinyatakan dalam dokumen acuan kalibrasi sesuai kapasitas alat atau rentang ukur tertentu yang biasa digunakan oleh pengguna alat. Jika dokumen acuan kalibrasi tidak menyatakan titik ukur, biasanya pengukuran dilakukan dalam selang 10% dari kapasitas

ukur alat. Titik ukur harus dibuat mudah dibaca oleh pengguna alat. Pada waktu pengukuran hanyalah melakukan pengambilan data dan tidak boleh melakukan kegiatan lainnya yang mungkin menyebabkan pembacaan atau pencatatan menjadi salah.

5. Pencatatan

Pencatatan hasil ukur harus berdasar kepada apa yang dilihat bukan kepada apa yang dirasakan. Pencatatan dilakukan seobjektif mungkin menggunakan *format* yang telah dirancang dengan teliti sesuai dengan ketentuan metode kalibrasi. Selain data ukur hal yang perlu dicatat adalah identitas alat selengkapnya serta faktor yang mempengaruhi kalibrasi seperti suhu ruangan, kelembaban, tekanan udara dan sebagainya.

6. Perhitungan

Data kalibrasi yang diperoleh dihitung sesuai metode kalibrasi. Perhitungan biasanya melibatkan pekerjaan mengkonversi satuan, menghitung nilai maksimum-minimum, nilai rata-rata, standar deviasi, atau menentukan persamaan regresi. Hasil perhitungan akan menjadi dasar dalam penarikan kesimpulan dan penentuan ketidakpastian kalibrasi.

a. Rata-rata

Rata-rata adalah bilangan yang di dapat dari hasil pembagian jumlah nilai data oleh banyaknya data dalam

kumpulan tersebut. Secara matematis rata-rata dapat dirumuskan :

$$\text{Rata-rata} = (\bar{X}) = \frac{\sum Xi}{n}$$

b. Simpangan (*Error*)

Simpangan (*Error*) adalah selisih dari rata-rata nilai terhadap masing-masing nilai yang diukur. Rumus simpangan (*Error*) yaitu :

$$\text{Simpangan} = \bar{X} - X_n$$

c. % *Error*

% *Error* adalah nilai prosentase dari simpangan (*Error*) terhadap nilai yang dikehendaki. Rumus % *Error* yaitu :

$$\% \text{ Error} = \frac{X_n - \bar{X}}{X_n} \times 100\%$$

d. Standar Deviasi

Standar Deviasi adalah suatu nilai yang menunjukkan tingkat (derajat) variasi kelompok data atau ukuran standar penyimpangan. Jika standar deviasinya semakin kecil maka data tersebut akan semakin presisi. Secara matematis standar deviasi dapat dirumuskan :

$$SD = \sqrt{\frac{(X_1 - \bar{X})^2 + (X_2 - \bar{X})^2 + \dots + (X_5 - \bar{X})^2}{n - 1}}$$