BAB IV

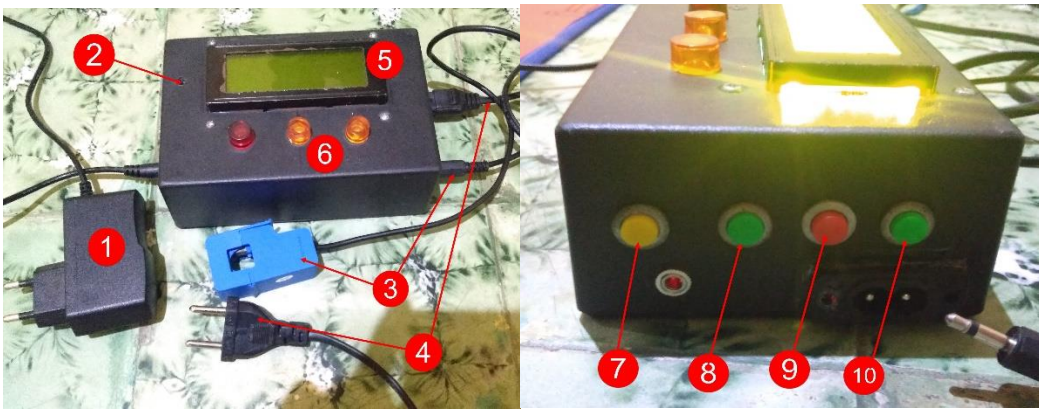
PENGUJIAN DAN PEMBAHASAN

Pada bab ini akan dijelaskan tentang sistem kerja dan pengujian keseluruhan alat *monitoring* yang telah dibuat. Untuk kategori pengujian dibagi menjadi 2 kategori utama yaitu pengujian perangkat keras *(hardware)* dan pengujian perangkat lunak *(software)*, dengan tujuan untuk mengetahui seberapa besar performa alat, apakah sesuai seperti dengan yang diharapkan atau belum.

Pada pengujian perangkat keras *(hardware)* dilakukan dengan cara mengetes perkomponen terpisah seperti pengujian sistem minimum Arduino uno R3, kemudian pengujian komponen sensor terhadap *board* Arduino atau komponen *output* led dengan *buzzer* terhadap *board* Arduino bahkan dari segi catu daya juga dilakukan pengujian apakah sudah maksimal kinerjanya atau belum.

Sedangkan pada pengujian perangkat lunak *(software)* yaitu pengujian mengenai sistem alur *script program* yang ditanamkan kedalam sistem minimum *chip* mikrokontroller apakah sudah berjalan sesuai dengan yang diharapkan atau justru belum sesuai kinerjanya sehingga dianggap perlu untuk dilakukan penyempurnaan dari segi *script program*. Selain itu pengujian dari segi *software* juga meliputi tentang kalibrasi sensor yang digunakan yaitu YHDC SCT 013-000 dan ZMPT101b yang memberikan sinyal *output* berupa sinyal analog kemudian dikonversikan oleh Arduino uno R3 menjadi sinyal digital menggunakan fitur ADC *(Analog to Digital Converter)*. Agar nilai akurasi parameternya benar-benar tepat

perlu dilakukan perhitungan nilai ADC dengan tepat pula. Setelah itu akan diuraikan pula tentang cara mengoperasikan alat monitoring ini kedalam bentuk tutorial pengoperasian nya seperti pengaksesan setiap submenu dan fungsi dari setiap tombol alat *monitoring*.



Gambar 4.1 Bentuk Alat *Monitoring* Daya Rumah Tangga

Keterangan gambar:

1. Catu daya/ *power supply* Arduino uno R3, spesifikasi input 220-250V AC 50Hz dan *output* 9V DC/ 500mA.
2. *Buzzer ventilator*, spesifikasi *input* 5V DC frekuensi *output* 1-5 kHz.
3. Sensor SCT 013-000, spesifikasi *input* dapat mengukur arus maksimal 100 A dan akan menghasilkan gelombang analog *output* sebesar 50 mV. *Turn ratio* 100 A : 0.05 A.
4. Sensor ZMPT101b, spesifikasi *input* dapat mengukur tegangan AC maksimal 250V.
5. *Alphanumeric LCD*, dengan penambahan fitur I2C dan dapat menampung karekter sebanyak 20 x 4 karakter dalam sekali tampil.

6. LED *indicator*, led ukuran 5mm, tegangan *input* maksimal 3V DC.
7. *Push button* kuning, untuk melakukan opsi *back* pada saat menu pilihan ditampilkan pada *alphanumeric LCD*.
8. *Push button* hijau 1, untuk melakukan opsi *down* pada saat memilih menu pilihan yang akan ditampilkan pada *alphanumeric LCD.*
9. *Push button* merah, untuk melakukan opsi *enter/ ok* pada saat memilih menu pilihan pada *alphanumeric LCD*.
10. *Push button* hijau 2, untuk melakukan opsi *up* pada saat memilih menu pilihan yang akan ditampilkan pada *alphanumeric LCD.*

4.1 Pengujian Perangkat Keras *(Hardware)*

Pengujian perangkat keras dilakukan dengan tujuan untuk melakukan uji coba sistem apakah telah berfungsi dengan semestinya atau belum, beberapa pengujian yang dilakukan pada alat *monitoring* daya ini adalah sebagai berikut:

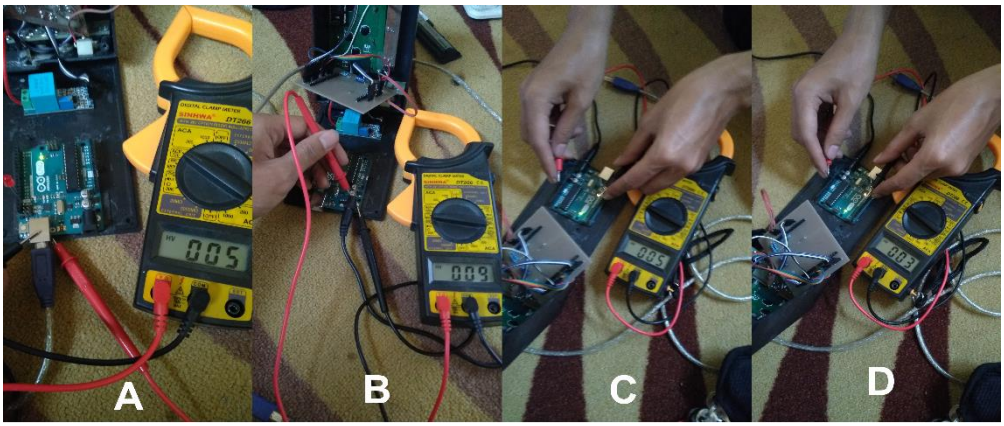
1. Pengujian catu daya/ *power supply*.
2. Pengujian sistem minimum *board* Arduino R3.
3. Pengujian *Alphanumeric LCD* 20 x 4.
4. Pengujian *piezo buzzer* dan LED sebagai sistem *alarm*.
5. Pengujian *push button* sebagai pengontrol arah menu yang dipilih.
6. Pengujian sensor SCT 013-000.
7. Pengujian sensor ZMPT101b.

4.1.1 Pengujian Catu Daya/ *Power Supply*

Catu daya merupakan sistem yang sangat berperan penting dalam memaksialkan kerja alat *monitoring* ini karena ia berfungsi sebagai pengubah dan penyuplai tegangan keseluruh komponen yang ada pada alat *monitoring* seperti *board* Arduino, sistem *alarm*, dan LCD *alphanumeric*. Pada alat *monitoring* ini terdapat 4 jenis catu daya/ *power supply* diantaranya catu daya USB 5V DC, catu daya adaptor 9V DC, catu daya 5V DC dan catu daya 3,3 V DC dari regulator tegangan *board* Arduino. Berikut ini merupakan tabel pengujian tegangan pada alat *monitoring*:

Tabel 4.1 Pengujian Catu Daya/ *Power Supply*

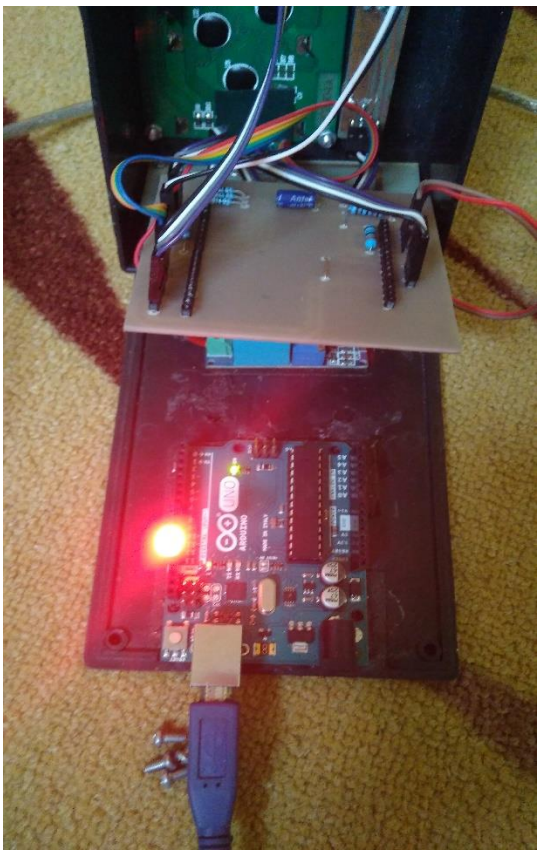
No	Jenis Catu Daya	Tegangan Terukur
1	USB komputer	5 V
2	Adaptor	9 V
3	Regulator Arduino 5V	5 V
4	Regulator Arduino 3,3V	3 V



Gambar 4.2 Pengukuran Tegangan (A) USB Komputer, (B) Adaptor 9V, (C)Regulator 5V, (D) Regulator 3,3V

4.1.2 Pengujian Sistem Minimum Arduino R3

Pengujian sistem minimum Arduino R3 ini adalah dengan memberikan tegangan pada Arduino lalu mengukur nilai tegangan pada setiap bagian sistem seperti *alphanumeric* LCD, sistem *alarm*, *push button* dll. Rangkaian sistem minimum dapat dikatakan berfungsi dengan baik apabila *script program* yang sudah dibuat di komputer dapat di-*upload* ke *board* Arduino dan programnya berjalan dengan baik tanpa terkendala.



Gambar 4.3 Pengujian Sistem *Minimum Board* Arduino

Pada pengujian program untuk *board* Arduino ini diuji dengan menggunakan *script program example* dari *software* Arduino IDE, lalu dilakukan

proses *compile* dan upload ke sistem minimum Arduino. Pada uji coba sistem minimum Arduino ini dapat dilihat seperti gambar di bawah ini bahwa *pindigital* 13 yang terhubung ke LED menyala selama 1 detik dan mati selama 1 detik, dengan menggunakan *script program* berikut ini:

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

4.1.3 Pengujian *Alphanumeric* LCD 20 x 4

Alphanumeric LCD 20 x 4 merupakan *user interface* yang digunakan oleh alat *monitoring* ini sebagai *display* untuk menampilkan parameter-parameter seperti tegangan, arus, dan konsumsi daya yang digunakan oleh beban. *Alphanumeric* LCD 20 x 4 ini, dihubungkan dengan board Arduino menggunakan 2 jalur komunikasi data yakni pin SCL dan SDA dari pin Arduino.

Untuk melakukan pengujian kinerja *Alphanumeric* LCD 20 x 4 terlebih dahulu dilakukan pengecekkan tegangan input pada LCD apakah sudah sesuai 5V atau jauh dari batas nilai toleransi. Karena ketika nilai tegangan yang masuk pada *alphanumeric* LCD tidak sesuai atau jauh dari batas toleransi 5V maka kinerja dari LCD tidak akan maksimal nantinya.

Tabel 4.2 Pengujian Tegangan *Alphanumeric* LCD

No	Pengujian	V terukur
1	Tegangan pin input LCD	5 V

Setelah dilakukannya pengecekkan tegangan dan ternyata dinyatakan telah sesuai maka dilanjutkan dengan pengujian secara fungsional yakni memasukkan *script program* untuk menguji coba menampilkan karakter pada *alphanumeric* LCD. Berikut adalah contoh *script* yang digunakan untuk menguji *alphanumeric* LCD:

```
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27,2,1,0,4,5,6,7);

void setup()
{
  lcd.begin (20,4);
  lcd.setBacklightPin(3,POSITIVE);
  lcd.setBacklight(HIGH);
}

void loop()
{

  lcd.home();
  lcd.print("   ERICK PRATAMA  ");
  lcd.setCursor(0,1);
  lcd.print("       UMY        ");
  lcd.setCursor(0,2);
  lcd.print("  SEMANGAT SKRIPSI  ");
  lcd.setCursor(0,3);
  lcd.print("      BISMILLAH     ");
  lcd.setCursor(0,4);
  delay(1000);

}
```

Gambar 4.4 Pengujian *Alphanumeric* LCD Untuk Menampilkan Karakter

4.1.4 Pengujian *Piezo Buzzer*

Piezo buzzer yang digunakan adalah *buzzer* jenis umum dengan *supply* tegangan input maksimal 5V DC *(Direct Current)*. *Buzzer* ini digunakan sebagai peranti *alarm* pada alat *monitoring* yang akan berbunyi ketika nilai daya yang terukur telah mencapai titik *trip*-nya MCB *(Miniature Circuit Breaker)*. *Buzzer* ini terhubung dengan pin digital no 3 pada *board* Arduino, dan diberikan resistor dengan nilai 100Ω ½ watt, toleransi 5% sebagai pengaman untuk mengatasi kemungkinan kelebihan arus yang masuk pada *buzzer* sehingga akan lebih mengamankan *buzzer* dari kerusakan.

Untuk melakukan pengujian pada *buzzer* maka terlebih dahulu membuat *script program* seperti di bawah ini lalu di *upload*-kan ke *board* Arduino.

```
//script untuk menguji logika 0
void setup()
    {pinMode(3, OUTPUT);}

    void loop()
    {digitalWrite(3, LOW);}
```

```
//script untuk menguji logika 1
void setup()
    {pinMode(3, OUTPUT);}

    void loop()
    {digitalWrite(3, HIGH);}
```

Pengujian pada peranti *buzzer* ini meliputi dua pengujian diantaranya pengujian pada saat diberikan logika 0 pada pin digital nomer 3 (pin yang terhubung dengan *buzzer*) setelah itu diukurlah berapa nilai tegangan yang terukur. Lalu selanjutnya untuk pengujian yang kedua yaitu dengan memberikan logika 1 pada pin digital nomer 3, setelah itu kembali mengukur nilai tegangan pada kaki-kaki *buzzer* tadi, dan memperhatikan kondisi bunyi pada *buzzer* untuk setiap logika 0 dan 1 yang diberikan. Perhatikan tabel 4.3 di bawah ini:

Tabel 4.3 Pengujian Tegangan *piezo buzzer*

No	Jenis Pengujian	V Terukur	Kondisi Buzzer
1	Pengujian pada logika 0	0 V	*Off*
2	Pengujian pada logika 1	4 V	*On*

Gambar 4.5 Pengujian Tegangan *Buzzer* Pada Saat *On*

4.1.5 Pengujian LED Indikator

Led indikator ini merupakan susunan 3 buah led biasa dengan ukuran 5mm yang dilengkapi resistor sebesar 220Ω ½ watt toleransi 1% yang dipasangkan disetiap led dan terdiri dari 2 buah warna kuning dan 1 buah warna merah sebagai bentuk peringatan *visual* pada alat *monitoring* ketika mendekati titik *trip*-nya MCB.

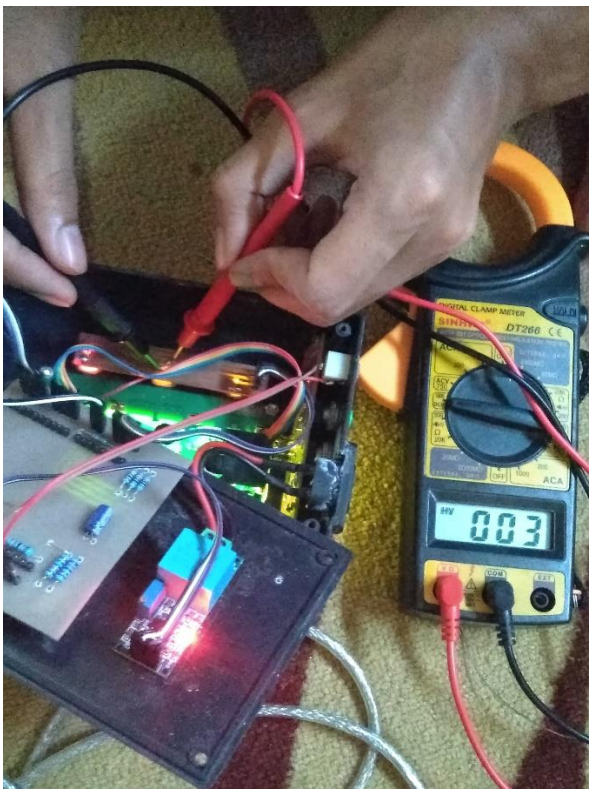
Untuk melakukan pengujian kondisi led indikator terlebih dahulu membuat *script* program seperti di bawah ini:

```
//script untuk menguji logika 0
    void setup()
    {
      pinMode(4, OUTPUT);
      pinMode(5, OUTPUT);
      pinMode(6, OUTPUT);
    }

    void loop()
        {
        digitalWrite(4, LOW);
        digitalWrite(5, LOW);
        digitalWrite(6, LOW);
        }


//script untuk menguji logika 1
void setup()
    {
      pinMode(4, OUTPUT);
      pinMode(5, OUTPUT);
      pinMode(6, OUTPUT);
    }

    void loop()
        {
        digitalWrite(4, HIGH);
        digitalWrite(5, HIGH);
        digitalWrite(6, HIGH);
        }
```

Lalu selanjutnya bentuk pengujian yang akan dilakukan pada pembahasan ini meliputi pengujian ketika led ini diberikan logika 0 dan logika 1 lalu dilakukan pengukuran tegangan pada setiap kaki-kaki led.

Tabel 4.4 Pengujian Tegangan Led Indikator Pada Alat *Monitoring*

No	Led ke-	Jenis Pengujian	V Terukur	Kondisi Led
1	Led 1	Pada logika 0	0 V	Mati
2	Led 1	Pada logika 1	3 V	Hidup
3	Led 2	Pada logika 0	0 V	Mati
4	Led 2	Pada logika 1	3 V	Hidup
5	Led 3	Pada logika 0	0 V	Mati
6	Led 3	Pada logika 1	3 V	HIdup



Gambar 4.6 Pengujian Tegangan LED Pada Saat *On*

4.1.6 Pengujian *Push Button*

Push button yang digunakan adalah *push button* jenis biasa dengan mode *switching* (mengirimkan sinyal satu per satu ketika ditekan). Cara sederhana untuk menguji coba *push button* masih layak atau tidak di tinjau dari segi fisiknya dapat diuji coba menggunakan multimeter dengan selector ke arah ohm meter atau *continuity*. *Push button* ini digunakan untuk mengontrol menu pada bagian *alphanumeric* LCD, pada alat *monitoring* ini terdapat 4 buah *push button* yakni *push button up, down, enter,* dan *back.* Untuk melakukan pengujiannya maka terlebih dahulu melakukan proses *upload script program* pada *board* Arduino seperti berikut ini agar dapat tampil pada *alphanumeric* LCD dan dapat langsung diuji coba pada *hardware*-nya. Berikut adalah potongan *script* untuk menguji coba *push button*:

```
#define _LCDML_CONTROL_digital_low_active        1
  #define _LCDML_CONTROL_digital_enable_quit     1
  #define _LCDML_CONTROL_digital_enter           9
  #define _LCDML_CONTROL_digital_up              10
  #define _LCDML_CONTROL_digital_down            8
  #define _LCDML_CONTROL_digital_quit            12
```

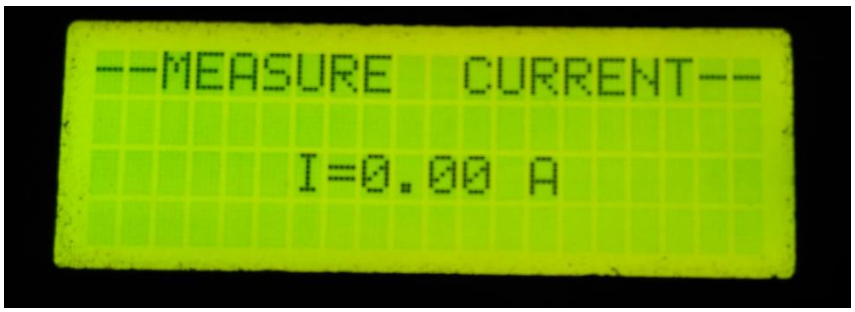
Gambar 4.7 Pengujian *Push Button* Dalam Akses Menu Alat *Monitoring*

4.1.7 Pengujian Sensor Arus SCT 013-000

Pada sensor arus seri SCT 013-000 ini untuk melakukan pengkalibrasiannya terlebih dahulu ditentukan nilai *burden* resistor seperti yang telah diuraikan pada bab III tentang perhitungan nilai kalibrasi dan rangkaian konverter dari pin *analog* menuju *jack* sensor SCT 013-000. Untuk melakukan pengujiannya maka terlebih dahulu membuat *script* programnya seperti di bawah ini:

```
#include "EmonLib.h"
EnergyMonitor emon1;

void setup()
{
  LiquidCrystal_I2C  lcd(0x27,  2,  1,  0,  4,  5,  6,  7,  3,  POSITIVE);
  emon1.current(0, 60);
}

void loop()
{
  emon1.calcVI(20,2000);
  float Irms           = emon1.Irms;
  lcd.setCursor(8,2);lcd.print(Irms);

  lcd.print(F("--MEASURE  CURRENT--"));
  lcd.setCursor(6,2);
  lcd.print(F("I="));
  lcd.setCursor(13,2);
  lcd.print(F("A"));
}
```

Setelah membuat *script* seperti di atas maka proses selanjutnya adalah melakukan *upload script* ke *board* Arduino, ketika di *upload* akan tampil parameter arus pada *serial monitor software* Arduino IDE seperti tampak pada gambar di bawah ini:



Gambar 4.8 Tampilan Parameter Arus Pada Alat *Monitoring*

Untuk menguji cobanya cukup meng-klemkan sensor SCT 013-000 pada salah satu kawat penghantar (boleh kawat fasa atau kawat netral) yang penting salah satu diantara keduannya, lalu memasangkan *jack* sensor ke *jack female* pada body alat *monitoring*. Apabila kawat penghantar tadi diberikan beban maka akan tampil nilai pengukuran dari sensor SCT 013-000 pada *alphanumeric* LCD alat *monitoring* seperti tampak pada gambar di bawah ini:

Gambar 4.9 Mengukur Arus Menggunakan Alat *Monitoring*

Tabel 4.5 Pengujian Tingkat Akurasi Sensor Arus Seri SCT 013-000

No	Jenis Beban	Media Pengujian		error pengukuran (%)
		Multimeter FLUKE 287 (Ampere)	Sensor SCT 013-000 (Ampere)	
1	Minigrinder 45Watt	0,22	0,21	4,54
2	Kipas Angin 35Watt	0,24	0,24	0
3	Solder 30Watt	0,12	0,11	8,3
4	*Charger Handphone*	0,04	0,04	0
5	*Speaker*	0,12	0,12	0

Dari data pengukuran di lapangan nilai arus yang didapatkan pada setiap beban dengan kapasitas daya yang berbeda-beda, mendapatkan hasil pengukuran pada sensor SCT 013-000 dengan nilai yang tidak terlalu jauh akurasinya dibandingkan dengan hasil pengukuran dari multimeter. Berdasarkan pada tabel perhitungan di atas maka didapatkan hasil perhitungan *error* pada sensor SCT 013-000:

1. Minigrinder 45 Watt

$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$
$$= \left|\frac{0.22 - 0.21}{0.22}\right| \times 100\%$$
$$= \left|\frac{0.01}{0.22}\right| \times 100\%$$
$$= 0.045 \times 100\%$$
$$= 4{,}54\ \%$$

2. Kipas Angin 35 Watt

$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$
$$= \left|\frac{0.24 - 0.24}{0.24}\right| \times 100\%$$
$$= \left|\frac{0}{0.24}\right| \times 100\%$$
$$= 0 \times 100\%$$
$$= 0\ \%$$

3. Solder 30 Watt

$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$
$$= \left|\frac{0.12 - 0.11}{0.12}\right| \times 100\%$$
$$= \left|\frac{0.01}{0.12}\right| \times 100\%$$
$$= 0.083 \times 100\%$$
$$= 8{,}3\ \%$$

4. *Charger Handphone*

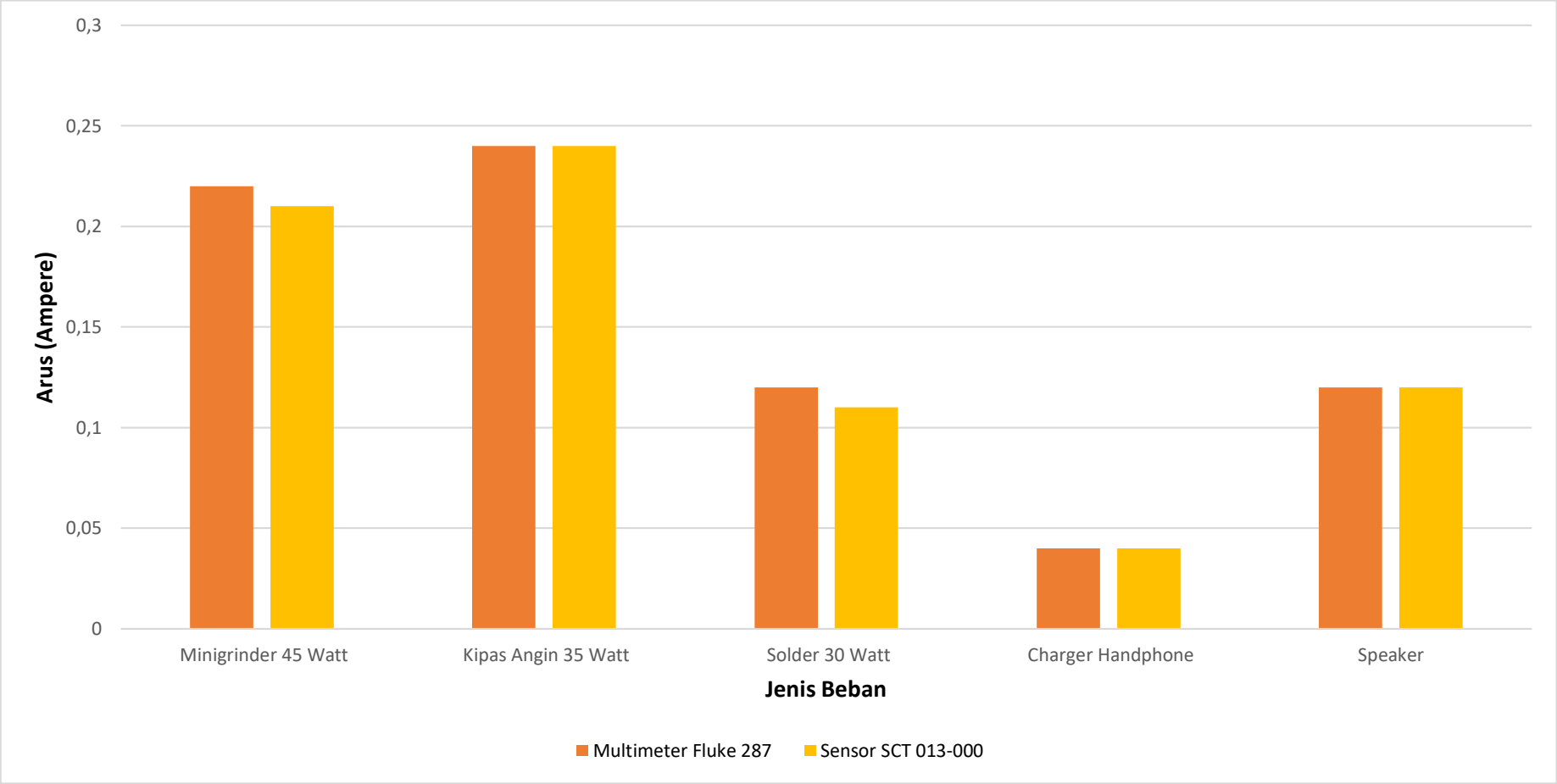
$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$
$$= \left|\frac{0.04 - 0.04}{0.04}\right| \times 100\%$$
$$= \left|\frac{0}{0.04}\right| \times 100\%$$
$$= 0 \times 100\%$$
$$= 0\ \%$$

5. *Speaker*

$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$
$$= \left|\frac{0.12 - 0.12}{0.12}\right| \times 100\%$$
$$= \left|\frac{0}{0.12}\right| \times 100\%$$
$$= 0 \times 100\%$$
$$= 0\ \%$$

Dari data hasil pengukuran dan uji coba sensor SCT 013-000 di atas maka dapat dihitung nilai error rata-rata dari setiap pengujian sebagai berikut:

$$Error\ rata - rata = \frac{Jumlah\ nilai\ error}{Banyaknya\ error\ terjadi}$$
$$= \frac{(4{,}54\ +\ 0\ +\ 8{,}31\ +\ 0\ +\ 0)\%}{5}$$
$$= 2{,}57\ \%$$



Gambar 4.10 Diagram Uji Coba Sensor Arus Terhadap Multimeter Fluke 287

4.1.8 Pengujian Sensor Tegangan ZMPT101b

Sensor ZMPT101b ini merupakan sensor tegangan yang menggunakan *transformer step down* sebagai media untuk mengkonversikan parameter tegangan sebenaarnya ke parameter tegangan yang akan dibaca oleh Arduino untuk nantinya diproses lebih lanjut. Untuk melakukan kalibrasi dari sensor ini dapat dilakukan dengan cara memutar *variable resistor* yang ada pada sensor sampai mendapatkan perbandingan nilai yang pas terhadap alat ukur yang sekiranya lebih presisi, namun sebelum melakukan perlu dilakukan uji coba terlebih dahulu maka dituliskanlah *script program* pada *software* Arduino IDE seperti di bawah ini:

```
#include "EmonLib.h"
EnergyMonitor emon1;

void setup()
{
  LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);
  emon1.voltage(1, 195.5, 1.2);
}

void loop()
{
  emon1.calcVI(20,2000);
  float Vrms           = emon1.Vrms;
  lcd.setCursor(8,2);lcd.print(Vrms);

  lcd.print(F("--MEASURE  VOLTAGE--"));
  lcd.setCursor(6,2);
  lcd.print(F("V="));
  lcd.setCursor(14,2);
  lcd.print(F("V"));
}
```

Setelah menuliskan *script* seperti di atas maka selanjutnya melakukan proses *upload* program pada *board* Arduino, sehingga akan ditampilkan parameter tegangan seperti pada gambar dibawah ini:

Gambar 4.11 Mengukur Arus Menggunakan Alat *Monitoring*

Untuk melakukan uji coba sensor ini cukup dengan menghubungkan kabel dari sensor terhadap stop kontak 220 VAC pada installasi rumah, maka akan secara otomatis membaca nilai tegangan pada installasi rumah kita tersebut, yang selanjutnya nilai inilah yang akan diproses lagi lebih lanjut untuk mendapatkan parameter daya yang digunakan pada installasi di rumah kita.

Gambar 4.12 Mengukur Tegangan Menggunakan Alat *Monitoring*

Tabel 4.6 Pengujian Tingkat Akurasi Sensor Tegangan Seri ZMPT101b

No	Waktu Pengukuran	Media Pengujian		error pengukuran (%)
		Multimeter FLUKE 287 (Volt)	Sensor ZMPT101b (Volt)	
1	06:00	215,5	215,3	0,092
2	13:35	211,2	211,1	0,047
3	20:00	212,3	212,2	0,047
4	01:14	213,5	213,4	0,046
5	04:23	214,6	214,5	0,046

Dari data pengukuran di lapangan nilai tegangan yang didapatkan pada setiap waktu yang berbeda-beda, mendapatkan hasil pengukuran pada sensor ZMPT101b dengan nilai yang tidak terlalu jauh akurasinya dibandingkan dengan hasil pengukuran dari multimeter. Berdasarkan pada tabel perhitungan di atas maka didapatkan hasil perhitungan *error* pada sensor ZMPT101b:

1. Pengukuran tegangan pukul 06:00

$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$
$$= \left|\frac{215{,}5 - 215{,}3}{215{,}5}\right| \times 100\%$$
$$= \left|\frac{0{,}2}{215{,}5}\right| \times 100\%$$
$$= (9{,}28e - 4) \times 100\%$$
$$= 0{,}092\ \%$$

2. Pengukuran tegangan pukul 13:35

$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$

$$= \left|\frac{211{,}2 - 211{,}1}{212{,}2}\right| \times 100\%$$
$$= \left|\frac{0.1}{212{,}2}\right| \times 100\%$$
$$= (4{,}71e - 4) \times 100\%$$
$$= 0{,}047\ \%$$

3. Pengukuran tegangan pukul 20:00

$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$
$$= \left|\frac{212{,}3 - 212{,}2}{212{,}3}\right| \times 100\%$$
$$= \left|\frac{0{,}1}{212{,}3}\right| \times 100\%$$
$$= (4{,}71e - 4) \times 100\%$$
$$= 0{,}047\ \%$$

4. Pengukuran tegangan pukul 01:14

$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$
$$= \left|\frac{213{,}5 - 213{,}4}{213{,}5}\right| \times 100\%$$
$$= \left|\frac{0{,}1}{213{,}5}\right| \times 100\%$$
$$= (4{,}68e - 4) \times 100\%$$
$$= 0{,}046\ \%$$

5. Pengukuran tegangan pukul 04:23

$$Nilai\ Error = \left|\frac{Nilai\ Acuan - Percobaan}{Nilai\ Acuan}\right| \times 100\%$$
$$= \left|\frac{214{,}6 - 214{,}5}{214{,}6}\right| \times 100\%$$

$$= \left|\frac{0{,}1}{214{,}6}\right| \times 100\%$$
$$= (4{,}65e-4) \times 100\%$$
$$= 0{,}046\ \%$$

Dari data hasil pengukuran dan uji coba sensor SCT 013-000 di atas maka dapat dihitung nilai error rata-rata dari setiap pengujian sebagai berikut:

$$Error\ rata-rata = \frac{Jumlah\ nilai\ error}{Banyaknya\ error\ terjadi}$$

$$= \frac{(0{,}092\ +\ 0{,}047\ +\ 0{,}047 +\ 0{,}046\ +\ 0{,}046)\%}{5}$$

= 0,0556%

Gambar 4.13 Diagram Uji Coba Sensor Tegangan ZMPT101b Terhadap Multimeter Fluke 287

4.2 Pengujian Perangkat Lunak *(Software)*

Pada pengujian yang kedua ini adalah dilakukannya pengujian yang ditinjau dari segi *software*-nya, atau dengan bahasa lain pengujian dari segi *script program* yang ditanamkan pada *chip* mirokontroller Arduino. Pengujian ini berisi tentang instruksi-instruksi yang digunakan dalam hal pengaksesan *alphanumeric* LCD, sensor SCT 013-000 dan sensor ZMPT101b, serta instruksi *push button* dan pemanggilan fungsi *alarm* pada alat *monitoring* ini.

Pada pengujian *software* ini digunakan 3 buah *library* tambahan untuk mengakses perangkat keras *(hardware)* yang digunakan, diantaranya *library emonlib* yang merupakan singkatan dari *energy monitor library*, tujuan digunakannya *library* ini adalah untuk memudahkan dalam hal pengaksesan sensor SCT 013-000 dan ZMPT101b, kemudian ada *library* untuk mengakses *alphanumeric* LCD dengan tambahan fitur I2C yang juga menggunakan library khusus yaitu *library new liquid crystal*, dan yang terakhir digunakan pula *library LCD menu lib* untuk mempermudah *programmer* dalam membuat menu dan fungsi-fungsi yang akan ditampilkan pada LCD nantinya kemudian dengan mudah dapat dikontrol oleh *user* menggunakan 4 buah *push button* yang telah ditentukan setiap fungsinya masing-masing.

Untuk penulisan *script program*, programmer sengaja memisahkkannya menjadi 5 file terpisah, diantaranya *LCDML_lcd_i2c*, *LCDML_CONTROL*, *LCDML_DISP*, *LCDML_FUNC_DISP*, dan *LCDML_FUNC_MEASURE*. Tujuannya dipisah-pisah adalah untuk memfokuskan penulisan *script program* pada pokok tertentu saja sehingga *programmer* dapat dengan mudah menganalisis

error nantinya, serta orang lain juga dapat dengan mudah memahaminya apabila akan dikembangkan atau disempurnakan untuk lebih lanjut.

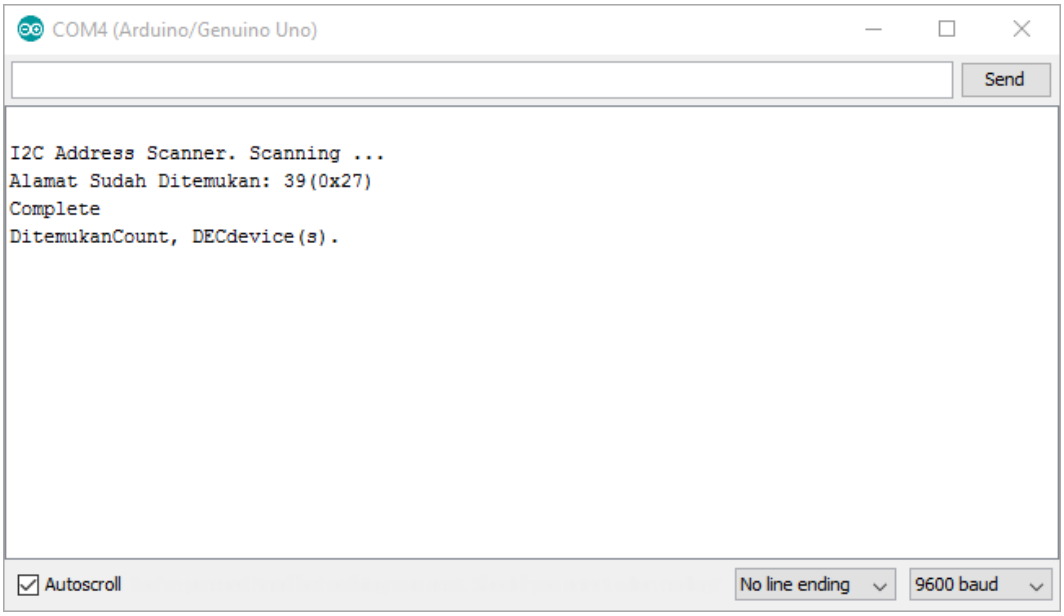
4.2.1 Pembahasan *Script Program Alphanumeric LCD* dengan I2C

Pada *alphanumeric* LCD dengan fitur tambahan *chip* I2C dalam penulisan *script program* sedikit berbeda dengan *alphanumeric* LCD biasa tanpa I2C, perbedaannya terletak pada pengalamatan LCD tersebut, dan diperlukan *script* tambahan untuk mengetahui alamat dari LCD tersebut sebab terkadang alamatnya berbeda-beda untuk setiap LCD maka dari itu diperlukan *script* seperti dibawah ini:

```
#include <Wire.h>
void setup() {
 Serial.begin(9600);
  while(!Serial)
  {
  }
  Serial.println ();
  Serial.println ("I2C Address Scanner. Scanning ...");
  byte count=0;
  Wire.begin();
  for(byte i=1;i<120;i++)
  {
    Wire.beginTransmission(i);
    if(Wire.endTransmission()== 0)
  {
    Serial.print("Alamat Sudah Ditemukan: ");
    Serial.print(i,DEC);
    Serial.print("(0x");
    Serial.print(i,HEX);
    Serial.println(")");
    count++;
    delay(1);
  }
}

Serial.println("Complete");
Serial.print("Ditemukan");
```

```
Serial.print("Count, DEC");
Serial.println("device(s).");
}
void loop()
{
}
```

Ketika meng-*upload script* maka buka *tab tool* pada *software* Arduino IDE, atau dapat dilakukan menggunakan shortcut ctrl + shift + m, akan tampil alamat dari I2C *alphanumeric* LCD tersebut di *serial monitor*. Lihat gambar dibawah ini:



Gambar 4.14 Proses *Scanning* Alamat I2C *Alphanumeric* LCD

Setelah mengetahui alamat dari *alphanumeric* LCD tersebut adalah 0x27 maka kita tinggal memasukkannya seperti pada *script* di bawah ini, kemudian LCD sudah dapat diakses untuk menampilkan nilai parameter atau menampilkan karakter sesuai keinginan *programmer*. Pada alat *monitoring* ini *programmer* menuliskan karakter dan fungsi pada *alphanumeric* LCD. Ketika dikontrol menggunakan *push button* dan menu yang disediakan telah dipilih maka akan menjalankan fungsi

sesuai instruksi yang telah dipilih tadi. Lihatlah kutipan *script* di bawah ini untuk lebih jelasnya:

```
#define _LCDML_DISP_cols             20
#define _LCDML_DISP_rows             4

LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);

void LCDML_DISP_setup(LCDML_FUNC_about_this)
{
  lcd.setCursor(0, 0);
  lcd.print(F(" For Monitor Energy"));
  lcd.setCursor(0, 1);
  lcd.print(F("450VA  900VA  1300VA "));
  lcd.setCursor(0, 2);
  lcd.print(F("--------------------"));
  lcd.setCursor(0, 3);
  lcd.print(F("Powered by: EmonLib"));
}

void LCDML_DISP_loop(LCDML_FUNC_about_this)
  {

  }

void LCDML_DISP_loop_end(LCDML_FUNC_about_this)
  {

   }
```

4.2.2 Pembahasan *Script* Pembuatan Menu

Pada alat *monitoring* ini digunakan menu untuk mengakses setiap instruksi yang diinginkan oleh *user*, misalkan *user* ingin mengakses menu untuk mengukur arus saja maka telah disediakan *menu*-nya dan dapat diakses dengan cara melakukan *select* menggunakan *push button*. Dalam penyusunan menu dan sub menu di dalamnya ini digunakan *library* khusus yaitu LCD Menu Lib sebagai media untuk membantu menuliskan *script*-nya.

Alat monitoring ini menggunakan 4 buah menu utama dengan 6 buah sub menu di dalamnya. Berikut adalah potongan *script* dalam membentuk menu pada alat *monitoring* ini:

```
#define _LCDML_DISP_cnt    10

LCDML_DISP_init(_LCDML_DISP_cnt);
LCDML_DISP_add (0,_LCDML_G1,LCDML_root   , 1  , "About This"     ,
LCDML_FUNC_about_this);
LCDML_DISP_add (1,_LCDML_G1,LCDML_root   , 2  , "Alarm Test"     ,
LCDML_FUNC_alarm_test);
LCDML_DISP_add (2,_LCDML_G1,LCDML_root   , 3  , "Measure"        ,
LCDML_FUNC);
LCDML_DISP_add (3,_LCDML_G1,LCDML_root_4 , 1  , "Current"        ,
LCDML_FUNC_current);
LCDML_DISP_add (4,_LCDML_G1,LCDML_root_4 , 2  , "Voltage"        ,
LCDML_FUNC_voltage);
LCDML_DISP_add (5,_LCDML_G1,LCDML_root_4 , 3  , "Apparent Power" ,
LCDML_FUNC_apparent_power);
LCDML_DISP_add (6,_LCDML_G1,LCDML_root   , 5  , "Select Monitor" ,
LCDML_FUNC);
LCDML_DISP_add (7,_LCDML_G1,LCDML_root_5 , 1  , "Power 450  VA"  ,
LCDML_FUNC_450);
LCDML_DISP_add (8,_LCDML_G1,LCDML_root_5 , 2  , "Power 900  VA"  ,
LCDML_FUNC_900);
LCDML_DISP_add (9,_LCDML_G1,LCDML_root_5 , 3  , "Power 1300 VA"  ,
LCDML_FUNC_1300);
LCDML_DISP_createMenu(_LCDML_DISP_cnt);
```

Dari potongan script di atas dapat dilihat bahwa secara keseluruhan alat monitoring ini tersusun atas 10 bagian menu dan keseluruhannya di masukkan ke dalam 1 grup yang diberi tanda berupa *LCDML_G1* yang berarti LCD *Menu Library Group* 1, kemudian untuk menentukan apakah menu yang kita buat tadi merupakan menu utama atau bagian dari sub menu maka dapat dibedakan dengan penulisan *scrip*t *LCDML_root* (merupakan menu utama) sedangkan *LCDML_root_4* dan *LCDML_root_5* (merupakan sub menu) dari menu utama Measure dan Select Monitor. Dalam menentukan penyusunan menu ini diperlukan urutan yang sesuai, maksudnya untuk setiap menu utama maka dimulai dengan

angka 1, 2, 3, dan seterusnya namun untuk sub menu harus mengulangi urutan dari 1,2,3 dan seterusnya. Sebagai contoh ketika *programmer* ingin membuat sub menu di dalam menu utama yang ke 3 maka sub menu tadi harus dimulai dari angka 1 lagi tidak boleh dilanjut ke angka 4. Perhatikan kembali penulisan *script* diatas.

Kemudian untuk memberikan nama pada setiap menu utama atau setiap sub menu maka digunakan tanda (“ ”) sebagai penanda bahwa ia merupakan karakter yang akan ditampilkan pada LCD nantinya dalam pilihan menu utama. Setelah itu dideklarasikan pula *LCDML_FUNC_XXXX* yang merupakan fungsi ketika menu tersebut dipilih maka akan melakukan instruksi sesuai dengan perintah pada bagian file baru di *tab LCDML_FUNC* (file program dipisahkan menjadi 5 bagian utama).

4.2.3 Pembahasan *Script* Pengaksesan *Push Button*

Pada alat monitoring ini digunakan 4 buah *push button* sebagai pengontrol menu, ia merupakan peranti yang berperan penting dalam hal pengaksesan fungsi sesuai keinginan *user*. Untuk menggunakannya agar dapat berfungsi sesuai fungsinya yang diinginkan maka digunakanlah potongan *script* seperti di bawah ini:

```
#if(_LCDML_CONTROL_cfg == 2)

  #define _LCDML_CONTROL_digital_low_active      1
  #define _LCDML_CONTROL_digital_enable_quit     1
  #define _LCDML_CONTROL_digital_enter           9
  #define _LCDML_CONTROL_digital_up              10
  #define _LCDML_CONTROL_digital_down            8
  #define _LCDML_CONTROL_digital_quit            12

void LCDML_CONTROL_setup()
{
```

```
  pinMode(_LCDML_CONTROL_digital_enter      , INPUT_PULLUP);
  pinMode(_LCDML_CONTROL_digital_up         , INPUT_PULLUP);
  pinMode(_LCDML_CONTROL_digital_down       , INPUT_PULLUP);
  # if(_LCDML_CONTROL_digital_enable_quit == 1)
    pinMode(_LCDML_CONTROL_digital_quit     , INPUT_PULLUP);
  # endif
}
```

Terlebih dahulu setiap *push button* pada alat *monitoring* dideklarasikan terhubung ke pin mana untuk setiap *push button* terhadap *board* Arduino, lalu diberikan *pullup internal* dari *chip* Arduino tujuannya agar *push button* tetap berada dalam kondisi konstan walaupun posisi nya sudah dalam keadaan tidak ditekan oleh *user* sehingga pemilihan menu dapat dilakukan dengan mudah.

Script untuk mendeklarasikan *push button* dapat dilihat pada *#define _LCDML_CONTROL_digital_low_active 1* hingga akhir kurung kurawal untuk definisi dan konfigurasi setiap pin push button di sesuaikan dengan skematik yang telah dirancang sebelumnya pada bab III, gambar 3.7. Dan untuk mengakses *internal pullup* pada *push button* dapat dilihat dari tulisan *pinMode (_LCDML_CONTROL_digital_enter, INPUT_PULLUP)*, sampai ke akhir kurung kurawal.

4.2.4 Pembahasan *Script* Pengaksesan *Alarm*

Dalam mengakses alarm sebagai *output* dari alat *monitoring* ini maka pada *script program* dibagi menjadi 3 bagian utama, yaitu *alarm level 1, alarm level 2,* dan *alarm level 3.* Berikut ini merupakan potongan *script program* yang telah dirancang oleh *programmer*:

```
void alarm_level_1()
```

```
{
  digitalWrite (led3    , HIGH);
  digitalWrite (led2    , HIGH);
  digitalWrite (buzzer  , HIGH);
  delay(1000);

  digitalWrite (led3    , LOW);
  digitalWrite (led2    , LOW);
  digitalWrite (buzzer  , LOW);
  delay(300);
}

void alarm_level_2()
{
  digitalWrite (led3    , HIGH);
  digitalWrite (led2    , HIGH);
  digitalWrite (buzzer  , HIGH);
  delay(1000);

  digitalWrite (led3    , LOW);
  digitalWrite (led2    , LOW);
  digitalWrite (buzzer  , LOW);
  delay(300);
}

void alarm_level_3()
{
  digitalWrite (led1    , HIGH);
  digitalWrite (led2    , HIGH);
  digitalWrite (led3    , HIGH);
  digitalWrite (buzzer  , HIGH);
  delay(1000);

  digitalWrite (led1    , LOW);
  digitalWrite (led2    , LOW);
  digitalWrite (led3    , LOW);
  digitalWrite (buzzer  , LOW);
  delay(300);
}
```

Pada *script alarm level 1*, akan menghidupkan 1 buah led sebagai peringatan dalam bentuk *visual* pada alat *monitoring* dan akan membunyikan *buzzer*. Begitu pula halnya untuk *script alarm level 2* dan *level 3.*

4.2.5 Pembahasan *Script* Pengaksesan Sensor SCT 013-000

Dalam mengakses sensor arus seri SCT 013-000 diperlukan beberapa *setting* parameter nilai kalibrasi seperti yang telah dijelaskan pada bab III, kemudian juga diperlukan tambahan *library* yaitu emonlib *(Energy Monitor Library)* sebagai penunjang untuk memudahkan akses sensor terhadap *board* Arduino. Berikut ini merupakan potongan *script* dalam mengakses sensor SCT 013-000:

```
void measure_current()
{
  emon1.calcVI(20,2000);
  double Irms           = emon1.Irms;
  lcd.setCursor(8,2);lcd.print(Irms);
}
```

Pada script di atas terdapat *script emon1.calcVI(20,2000)*, itu merupakan baris *script* untuk melakukan kalkulasi nilai arus, tegangan, daya aktif, daya semu hingga cos phi sekaligus, yang keseluruhannya dapat dipanggil menggunakan fungsi emon1.Vrms untuk mengakses tegangan, emon1.Irms untuk mengakses arus dan emon1.apparentPower untuk mengkalkulasi nilai daya semu dan emon1.powerFactor untuk akses nilai cos phi.

4.2.6 Pembahasan *Script* Pengaksesan Sensor ZMPT101b

Dalam mengakses sensor tegangan seri ZMPT101b juga memerlukan *library* emonlib sebagai penggerak utama dalam melakukan pengukuran parameter tegangan, pada prinsipnya memanfaatkan nilai *analog* yang terbaca pada sensor lalu dikirimkan pada *board* Arduino sebagai media kalkulasi sekaligus konverter

dalam memantau perkembangan nilai tegangan yang selalu berubah-ubah, lalu dikalkulasikan lagi lebih lanjut untuk melakukan perhitungan daya yang terpakai pada installasi rumah. Berikut ini merupakan potongan *script* untuk mengakses sensor ZMPT101b:

```
void measure_voltage()

{ emon1.calcVI(20,2000);
  double Vrms           = emon1.Vrms;
  lcd.setCursor(8,2);lcd.print(Vrms); }
```

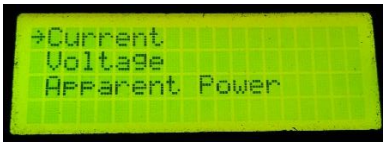
Pada prinsipnya karena sensor ZMPT101b menggunakan *library* yang sama dengan sensor SCT 013-000 yakni sama-sama menggunakan *library* emonlib maka dalam hal pengaksesan tidak jauh berbeda, perbedaannya hanya terletak pada *setting* pin *analog*, nilai kalibrasi, dan *phase shift* saja yang dideklarasikan pada menu *void* utama program.

4.3 Pengujian Alat *Monitoring* Secara Keseluruhan

Pengujian keseluruhan adalah pengujian alat *monitoring* dari mulai pertama kali dihidupkan hingga sampai melakukan *monitoring* daya, seluruh pengujian ini dituangkan kedalam bentuk tabel di bawah ini:

Tabel 4.7 Pengujian Alat *Monitoring* Secara Keseluruhan

No	Item Pengujian	Hasil Pengujian	Keterangan
1	Tampilan awal alat *monitoring*		Pada saat alat monitoring pertama kali dihidupkan maka akan tampil pilihan menu pada LCD yang dapat melakukan *scroll.*

No	Item Pengujian	Hasil Pengujian	Keterangan
2	Menu *about this* alat *monitoring*		Menu ini berfungsi untuk memberikan keterangan singkat pada *user* mengenai fungsi alat ini.
3	Menu *alarm test* alat *monitoring*		Menu ini berfungsi untuk menguji coba keseluruhan komponen *alarm* yang terdiri dari led dan *buzzer*.
4	Menu *Measure* alat *monitoring*		Menu ini berfungsi untuk melakukan pengukuran arus, tegangan, dan daya yang dikonsumsi/ *apparent power* tanpa melakukan *monitoring* pada installasi rumah.
5	Menu *Select Monitor* alat *monitoring*		Menu ini berfungsi untuk melakukan *monitoring* pada 3 *range* daya PLN diantaranya, pelanggan 450VA, 900VA, dan 1300VA yang nantinya akan memberitahu *user* ketika MCB akan *trip* lewat bunyi *alarm*

4.4 Pengujian Alat *Monitoring* Terhadap *Range* Daya Pelanggan PLN

Karena alat ini fungsi utamanya ditujukan sebagai alat untuk melakukan *monitoring* konsumsi daya pada sebuah rumah dan nantinya akan membunyikan *alarm* ketika MCB akan *trip*, maka fokus pengujian kali ini berada pada *range* daya pelanggan 900 VA sebagai salah satu sampel untuk membuktikan kinerja alat secara fungsional telah berfungsi maksimal, walaupun alat ini sebenarnya juga

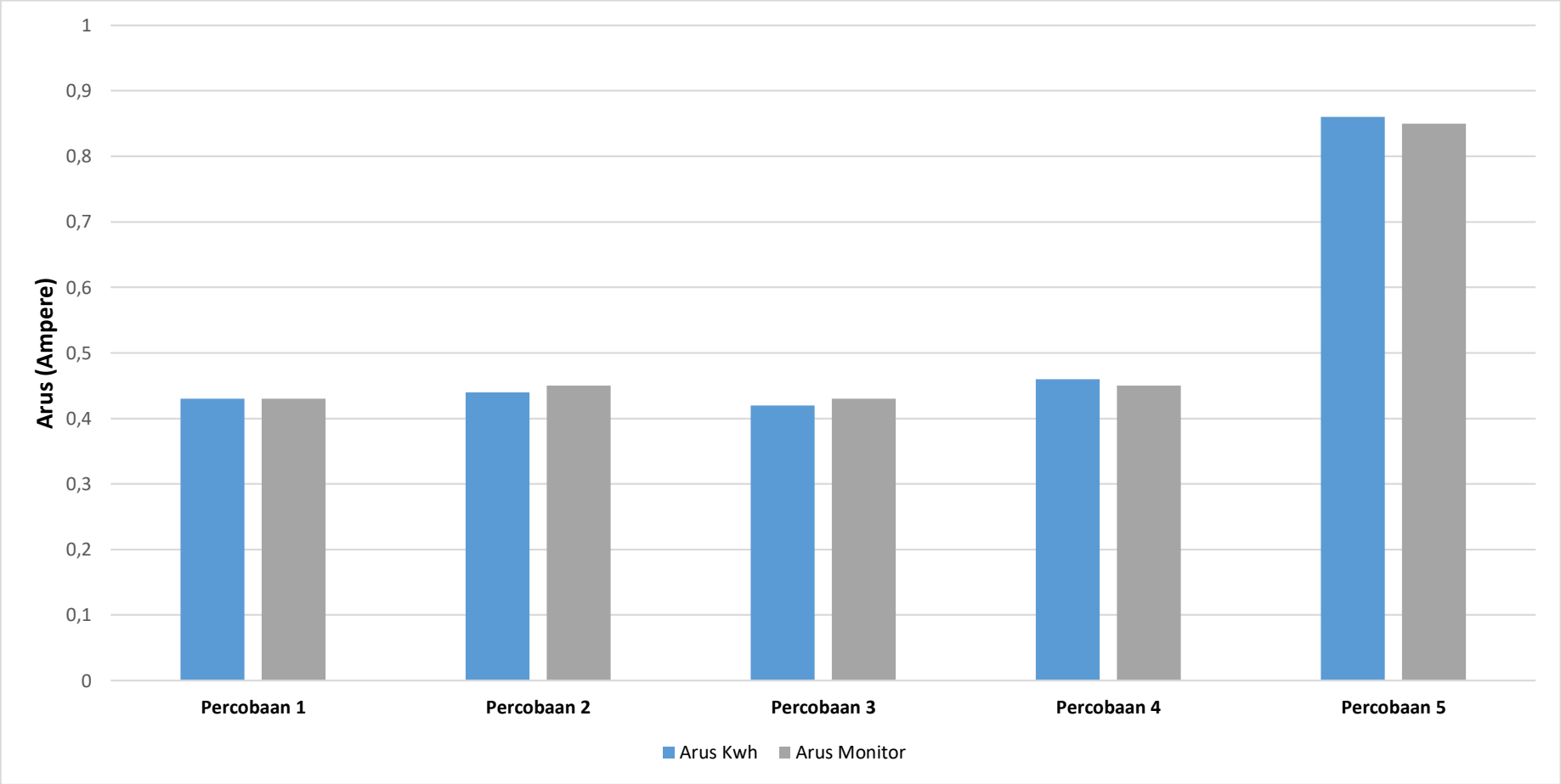
mampu untuk melakukan pemonitoringan pada *range* daya pelanggan PLN 900 VA dan 1300 VA.

4.4.1 Pengujian Pada *Range* Daya PLN 450 VA

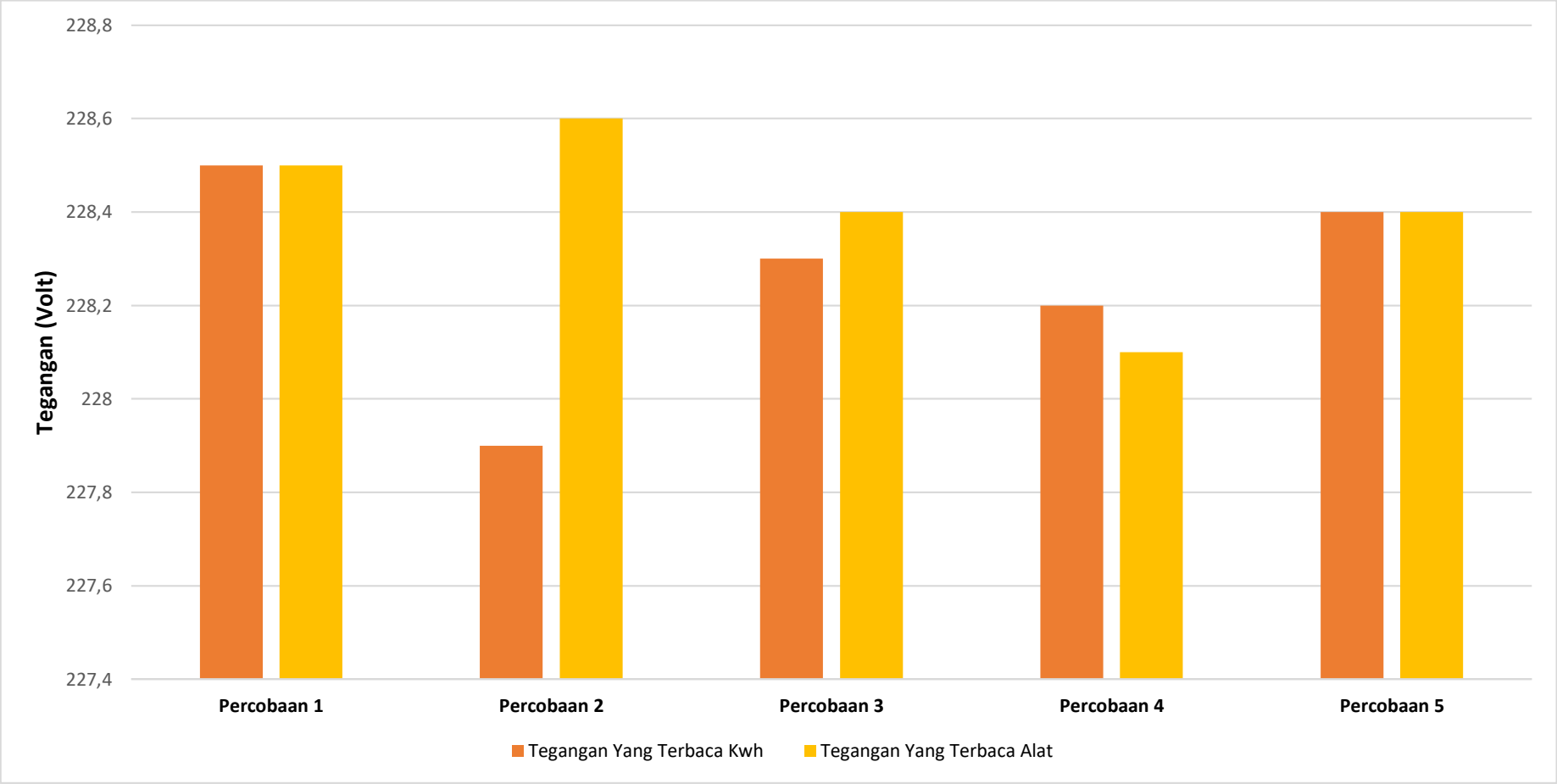
Untuk membuktikan kinerja dari alat monitoring ini sudah maksimal atau belum maka dilakukanlah pengujian khusus pada *range* daya pelanggan PLN 450 VA yang menggunakan Kwh meter digital, sebab Kwh meter digital ini dapat menampilkan nilai arus dan tegangan yang digunakan pada installasi rumah tersebut sehingga, akan lebih memudahkan dalam hal perbandingan nilai akurasi pengukuran dan *monitoring*.

Tabel 4.8 Pengujian Tingkat Akurasi Sensor Terhadap Kwh Meter Digital

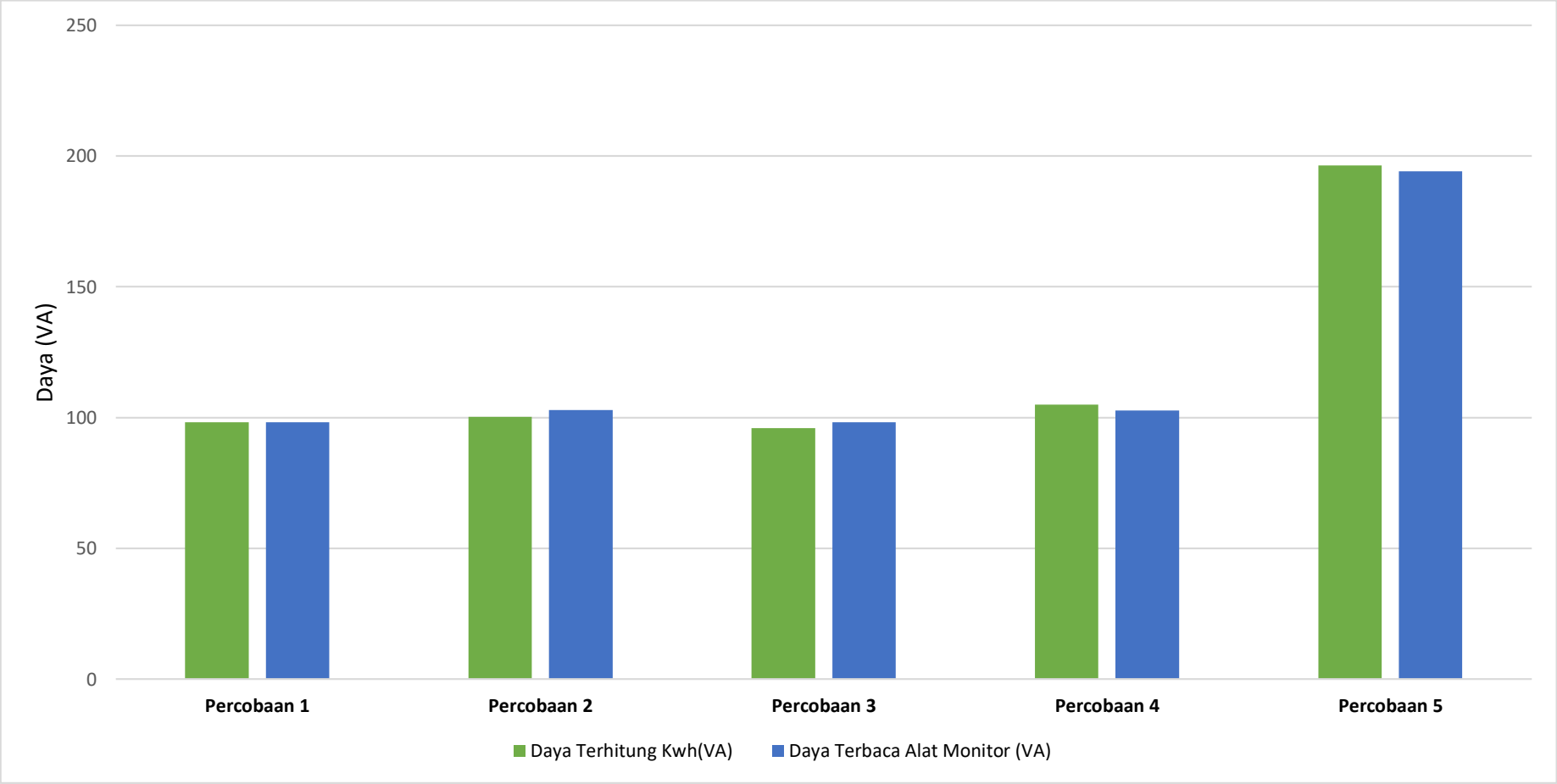
No	Kwh Meter		Daya Kwh (VA)	Alat Monitoring		Daya Monitoring (VA)	Keterangan
	Arus (A)	Tegangan (V)		Arus (A)	Tegangan (V)		
1.	0,43	228,5	98,26	0,43	228,5	98,26	Tidak bunyi
2.	0,44	227,9	100,28	0,45	228,6	102,87	Tidak bunyi
3.	0,42	228,3	95,89	0,43	228,4	98,21	Tidak bunyi
4.	0,46	228,2	104,97	0,45	228,1	102,65	Tidak bunyi
5.	0,86	228,4	196,42	0,85	228,4	194,14	Bunyi



Gambar 4.15 Diagram Uji Coba Sensor Arus Alat Monitoring Terhadap Kwh Meter Digital



Gambar 4.16 Diagram Uji Coba Sensor Tegangan Alat Monitoring Terhadap Kwh Meter Digital



Gambar 4.17 Diagram Uji Coba Daya Terbaca Pada Alat Monitoring Terhadap Kwh Meter Digital