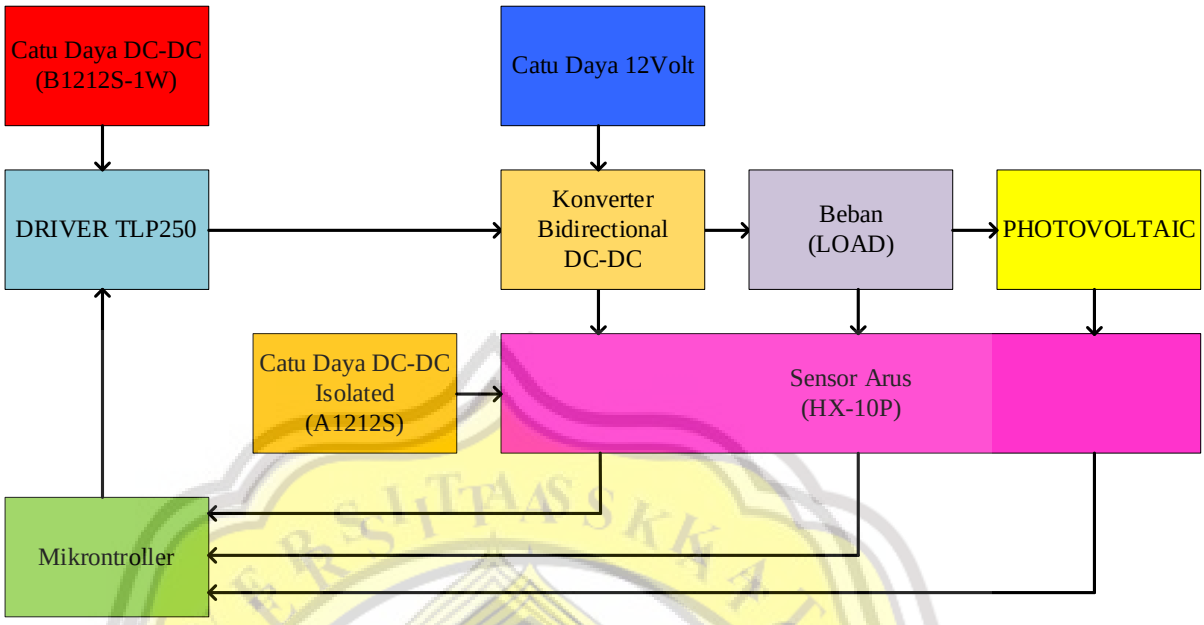
# BAB III

# DESAIN DAN IMPLEMENTASI

## 3.1 Pendahuluan

Strategi kontrol yang telah disajikan akan dibuktikan dengan simulasi yang dibuat lalu dari simulasi tersebut akan diimplementasikan menjadi sebuah *hardware* untuk membuktikan hasil dari simulasi dengan dasar teori yang ada. Desain dari topologi terbaru yang telah dibuat dalam simulasi dengan menggunakan bantuan PSIM yaitu aplikasi *Power* Simulator dimana parameter dalam simulasi tersebut disamakan dengan nilai parameter pada pembuatan *hardware*nya. Nilai parameter disajikan seperti pada Tabel 3.1. Pada Gambar 3.1 merupakan implementasi *hardware* yang menjadi sebuah acuan dan menjelaskan hubungan komponen yang digunakan pada konverter DC-DC ini. Pada Gambar 3.1 merupakan diagram blok dari cara kerja *hardware* yang telah dibuat pada simulasi dan alat.

Pada diagram blok ini dapat dilihat hubungan atau koneksi pada suplai catu daya, *driver*, serta beban yang akan digunakan. Pada *hardware* ini menggunakan berbagai macam catu daya untuk memenuhi kebutuhan tegangan yang diinginkan sehingga dapat dengan mudah memenuhi kebutuhan tegangan pada *hardware*. Pada diagram blok dapat dilihat bahwa, rangkaian mencakup suplai catu daya, *driver*, serta beban yang digunakan. Catu daya yang digunakan sendiri terdapat bermacam – macam disesuaikan dengan kebutuhan dan kemudahan dalam pembuatan *hardware*.



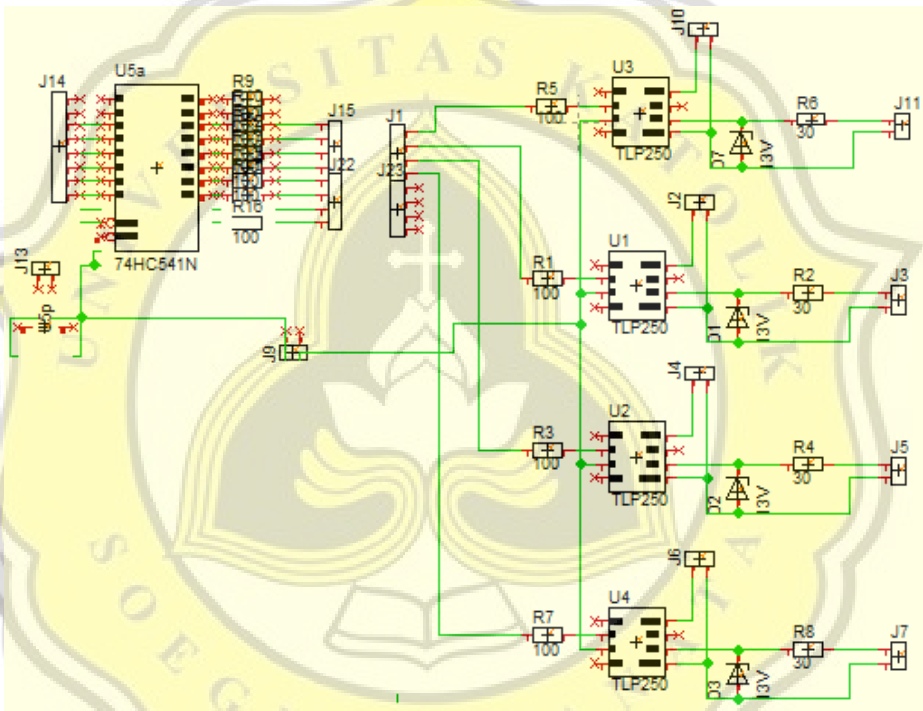

Gambar 3. 1 Diagram Blok Konverter *Bidirectional* DC-DC

Tabel 3. 1 Parameter Simulasi dan *Hardware*

| **Parameter** | **Value** |
|---|---|
| *DC Microgrid Voltage* | 48 $V_{DC}$ |
| *Battery Voltage* | 60 $V_{DC}$ |
| *Inductor Filter* | 5 mH |
| *Capacitor Filter C1, C2* | 220 uF |
| *Load Resistance* | 30 Ohm |
| *Switching Frequency* | 25 kHz |

## 3.2 Rangkaian *Driver* TLP250

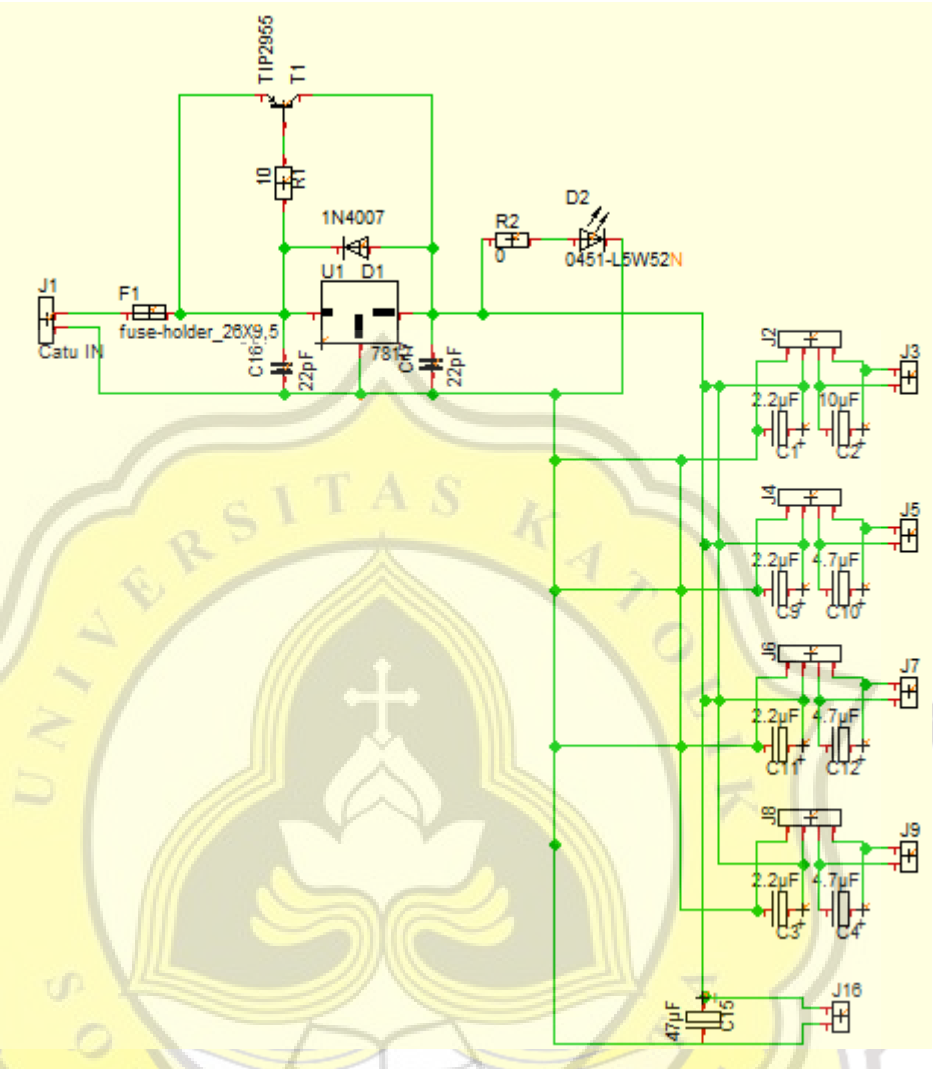
Rangkaian *Driver* TLP250 merupakan rangkaian penting yang digunakan dalam konverter ini. Rangkaian *driver* TLP250 berguna untuk menghubungkan antara *microgrid* dan saklar daya. Rangkaian *driver* TLP250 yang disajikan pada Gambar 3.2 merupakan rangkaian yang dibutuhkan pada konverter yang diimplementasikan dengan jumlah 4 buah.




Gambar 3. 2 Rangkaian *Driver* TLP250

## 3.3 Rangkaian Sirkuit B1212S

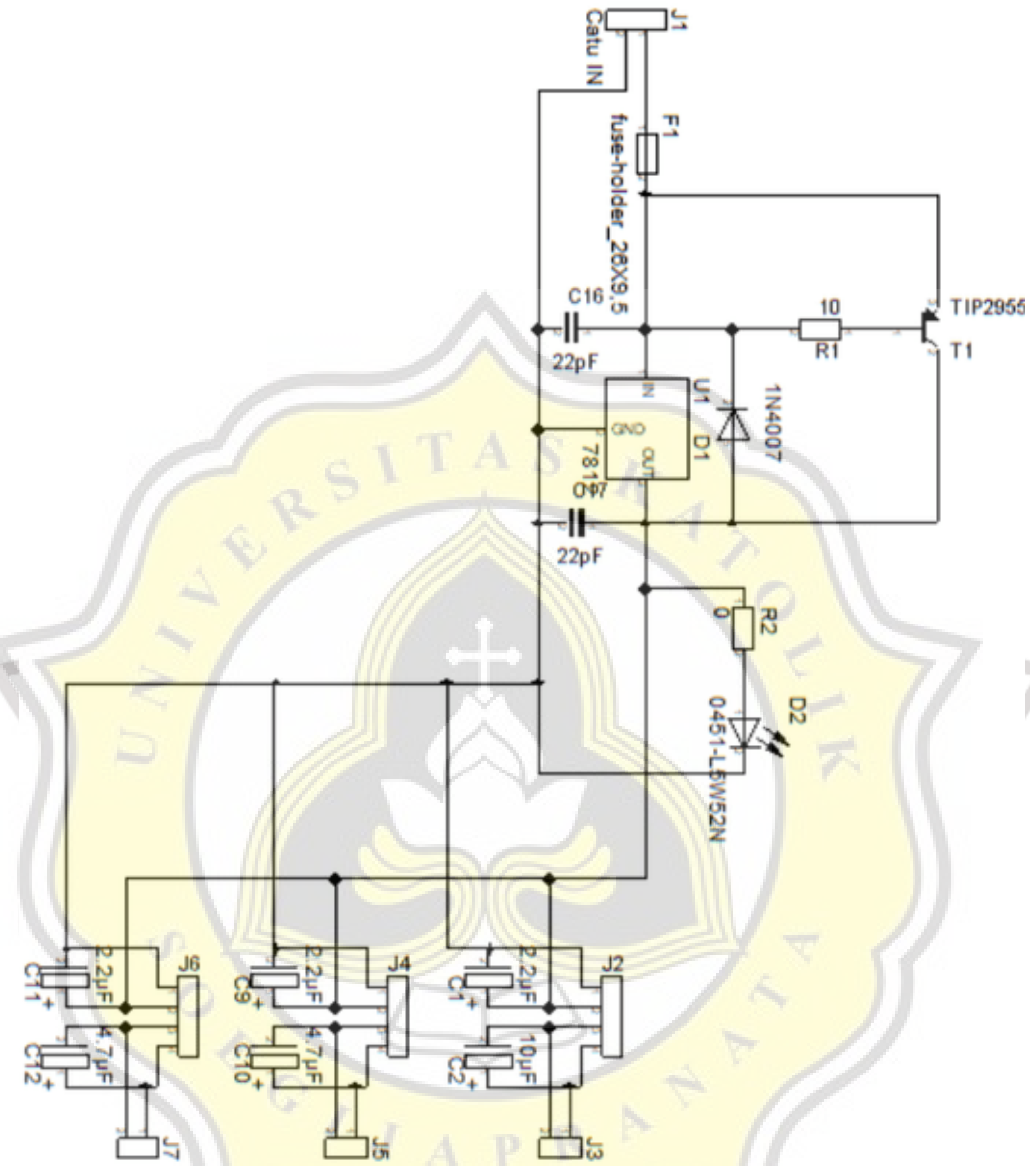
Rangkaian sirkuit B1212S merupakan rangkaian dari suplai daya yang memiliki output tegangan 12 Volt dan 0 Volt. Suplai daya dari B1212S ini akan digunakan untuk memberi suplai tegangan pada rangkaian *driver* TLP250 agar Tlp250 dapat beroperasi dengan baik.




Gambar 3. 3 Rangkaian Sirkuit B1212S

## 3.4 Rangkaian Sirkuit A1212S

Rangkaian sirkuit A1212S merupakan catu daya yang memiliki output tegangan 12V, 0V, dan -12V. A1212S merupakan rangkaian catu daya yang terisolasi sehingga memiliki tiga macam keluaran. Rangkaian ini digunakan untuk mensuplai tegangan yang dibutuhkan pada sensor yang digunakan seperti sensor arus.

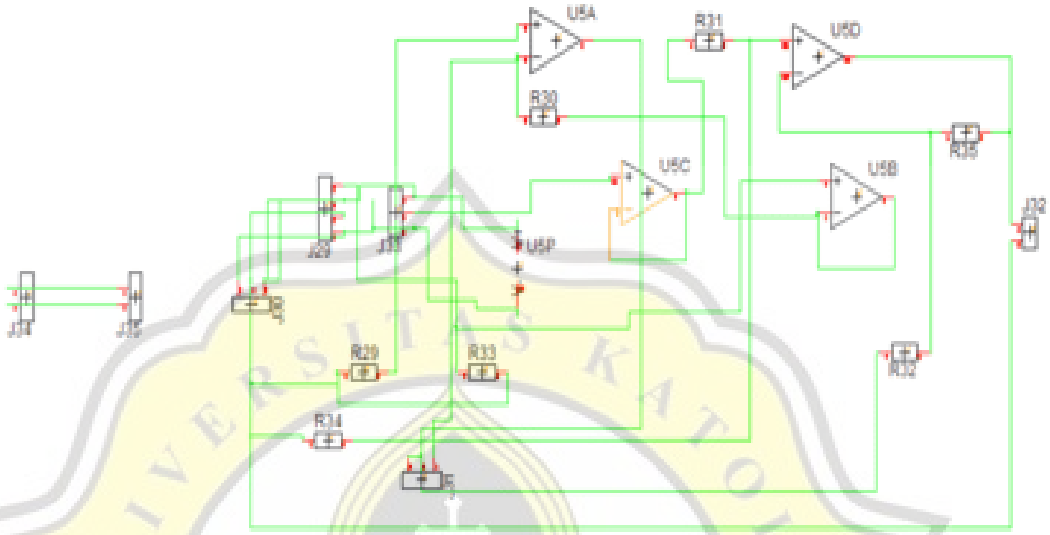



Gambar 3. 4 Rangkaian Sirkuit A1212S

## 3.5 Rangkaian Sensor Arus HX-10P

Rangkaian sensor arus HX-10P digunakan untuk membaca nilai arus keluaran secara aktual yang dihasilkan oleh PV, beban, dan konverter *bidirectional* DC-DC. Sensor ini bekerja pada arus maksimal hingga 10 A dengan frekuensi maksimal 50kHz. Sensor ini membutuhkan suplai

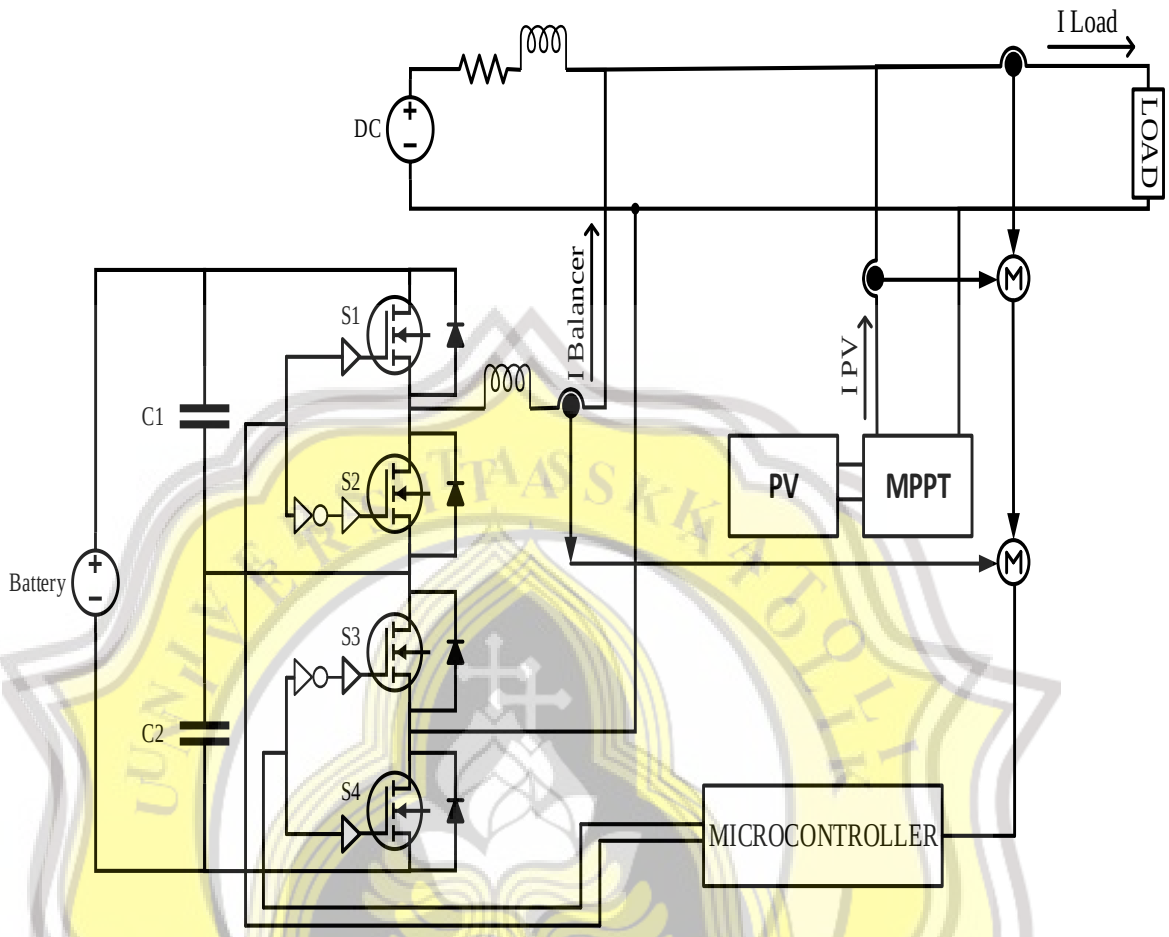
tegangan dengan tiga input tegangan agar sensor ini dapat bekerja yaitu 12Volt, 0Volt, dan -12Volt.




Gambar 3. 5 Rangkaian Sensor Arus HX-10P

## 3.6 Rangkaian Topologi Konverter *Bidirectional* DC-DC

Rangkaian topologi dari konverter *bidirectional* DC-DC yang disajikan merupakan interkoneksi antara *microgrid* DC, beban, PV, dan konverter *bidirectional* DC-DC. Pada konverter ini bekerja untuk memproses arus keluaran yang terdeteksi oleh sensor arus pada arus beban, arus konverter, dan arus PV. Konverter ini bekerja dengan mengkombinasikan *energy balancer* dimana membutuhkan 4 mosfet dan 2 kapasitor pada konverter. Rangkaian topologi dari konverter *bidirectional* DC-DC dapat dilihat pada Gambar 3.6.

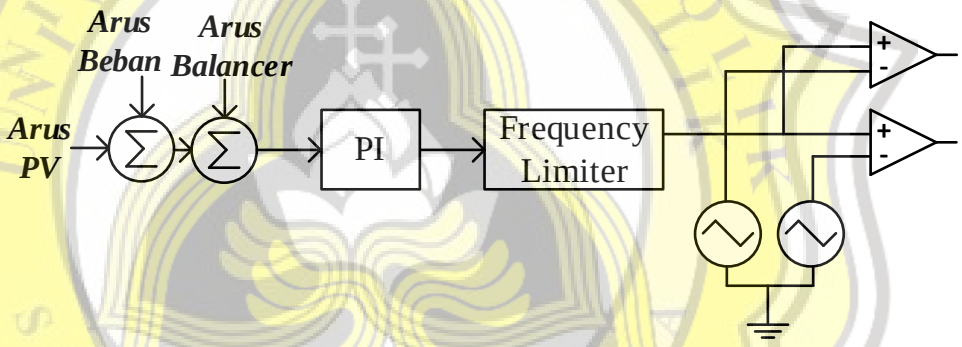



Gambar 3. 6 Rangkaian Topologi Konverter *Bidirectional* DC-DC

## 3.7 Rancangan Strategi Kontrol

Algoritma kontrol konverter penyeimbang energi dapat dilihat pada Gambar 3.7. Kontroler membaca arus keluaran sebagai umpan balik, arus dari beban, dan arus dari PV. Di sini ada pengontrol proporsional-integral (PI) yang digunakan. Pertama, pembacaan arus dari beban dan PV dihitung, menghasilkan sinyal arus kesalahan pertama. Sinyal arus kesalahan pertama dihitung dengan arus keluaran dari keluaran untuk menghasilkan sinyal arus kesalahan kedua yang diproses oleh pengontrol PI. Sinyal keluaran dari

pengontrol PI harus diberikan pembatas. Oleh karena itu, arus mengikuti kemampuan baterai agar tidak merusak baterai. Kemudian sinyal keluaran tersebut diproses untuk modulasi sinyal pembawa untuk menghasilkan PWM, yang digunakan untuk mengontrol saklar energi dari konverter dua arah. Sinyal yang dihasilkan dari konverter melewati *filter* sehingga sinyal keluaran memiliki noise yang lebih sedikit. Kp adalah *gain proporsional*, Ki adalah *gain integrator*, e(t) adalah nilai *error*, dan dt adalah perubahan waktu. Dengan menggunakan pengontrol PI, kesalahan kondisi tunak mendekati nol.

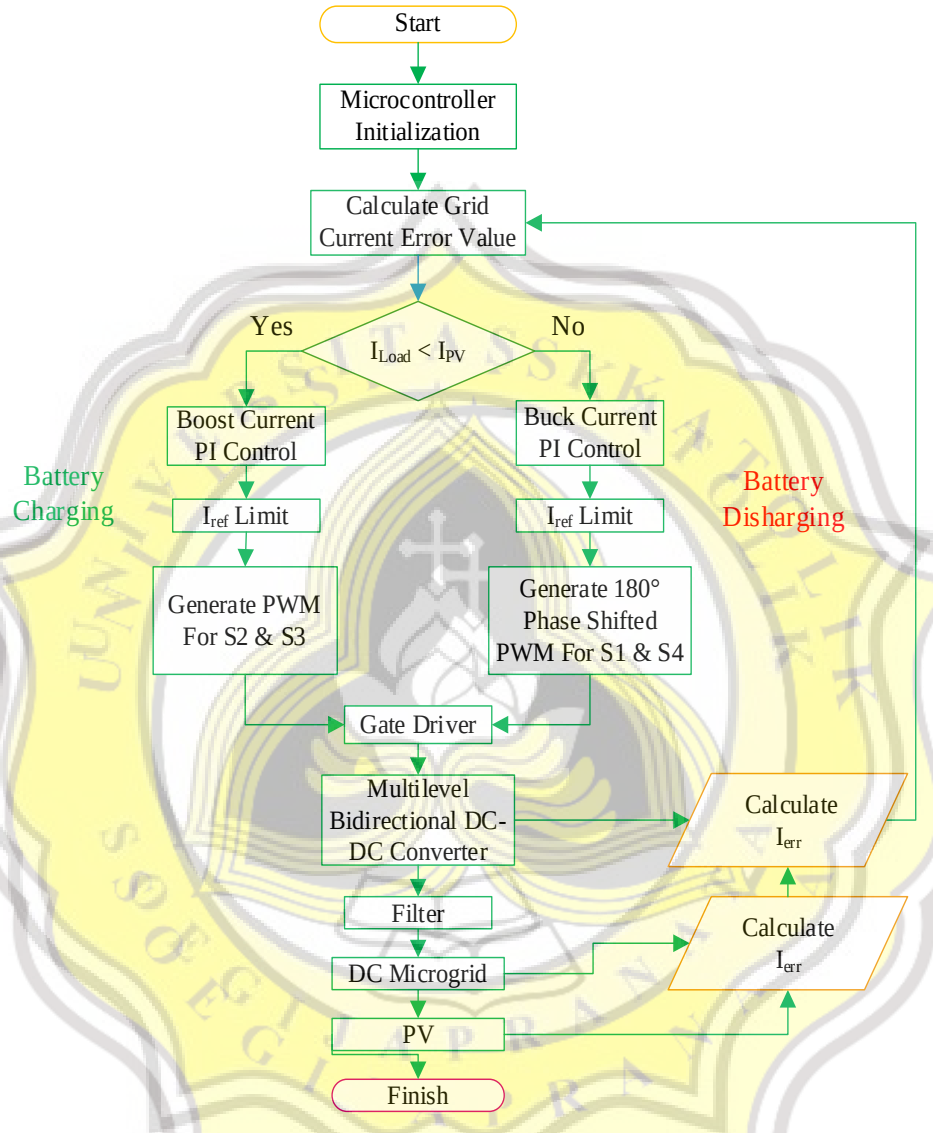



Gambar 3. 7 Strategi kontrol

### 3.7.1 Flowchart Pemrograman

Pada Gambar 3.8 merupakan flowchart dari pemrograman yang merupakan algoritma kontrol yang digunakan untuk mengontrol arus keluaran agar dapat dikontrol dengan baik sehingga menghasilkan kestabilan dan hasil keluaran yang stabil dan baik. Algoritma kontrol ini dikendalikan dengan menggunakan bahasa pemrograman. Hasil dari flowchart pemrograman ini akan memberikan output dari *microgrid* berupa sinyal *Pulse Width*

*Modulation* yang digunakan untuk mengatur saklar daya yang digunakan.




Gambar 3. 8 Flowchart Pemrograman

### 3.7.2 Pemrograman

Pada bagian ini merupakan bagian pembahasan mengenai program yang digunakan pada implementasi *hardware* konverter

*bidirectional* DC-DC yang digunakan pada *microgrid* dalam mengontrol konverter tersebut.

```
#include <STM32F4ADC.h>
#define D1 PE0
#define D2 PE1
#define D3 PB9
#define D4 PB8

int i=1000;
int S1,S2,S3,S4,S5,S6;
STM32ADC inADC(ADC1);
const int setOverflow = 4000;
int count,car1,car2;
int refv,iact,iload,ipv,actv,err,mod_final,car,pot,iref;
double itg,lastitg,pi,mod,Bmod,P,I;
uint16_t analog_pins[] = {PA0, PA1, PA2, PA3, PA4, PA5, PA6};

//control
float kp=0.2;
float ki=50;

void PINMODE() {
 pinMode(D1, OUTPUT);
 pinMode(D2, OUTPUT);
 pinMode(D3, OUTPUT);
 pinMode(D4, OUTPUT);

}
void setup() {
 Serial.begin(9600);
 PINMODE();
 for (uint8_t x = 0; x<sizeof(analog_pins); x++)
 pinMode(analog_pins[x], INPUT_ANALOG);

 Timer2.init();
 Timer2.pause();
 Timer2.setMasterMode(TIMER_MASTER_MODE_UPDATE);
 Timer2.setPeriod(3000);
 Timer2.setMode(TIMER_CH2, TIMER_OUTPUT_COMPARE);
 Timer2.setCompare(TIMER_CH2, 1);
 Timer2.attachInterrupt(TIMER_CH2,INT1);
```

Inisialisasi *Header Library*

Deklarasi Variabel dan Konstanta

Inisialisasi Nilai KP dan KI

Inisialisasi Port Input Output

Inisialisasi Timer Counter 2 dan 3

```
Timer2.refresh();
Timer2.resume();

Timer3.init(); //PWM timer
Timer3.setPeriod(20);
Timer3.refresh();
Timer3.resume();

delayMicroseconds(19); //carrier phase delay

Timer4.init();
Timer4.setPeriod(20);
Timer4.refresh();
Timer4.resume();

pinMode(PB0,PWM);
pinMode(PB6,PWM);
pinMode(PB7,PWM);
pinMode(PB8,PWM);

inADC.setSamplingTime(ADC_SMPR_3);
inADC.enableDMA();
}
void loop() {
 while(1){
sensor();
bidirect();
 }
//Serial.println(R_acti);
}
void sensor(){
refv = map(analogRead(PA0),0,4095,-2000,2000);//referensi voltage
iact = map(analogRead(PA1),0,4095,-2000,2000);//act arus
iload = map(analogRead(PA2),0,4095,-2000,2000);//act arus
ipv = map(analogRead(PA3),0,4095,-2000,2000);//act arus
actv = map(analogRead(PA4),0,4095,-2000,2000);//actual
pot = map(analogRead(PA5),0,4095,-2000,2000);//potensio
}
void bidirect(){
//   refv = 4000;
  iref= iload-ipv;
  err = iref-iact;
  P = kp*err;
  itg = lastitg+err*0.0001;
  I=ki*itg;
  pi=P+I;
```

Inisialisasi Timer Counter 4

Inisialisasi pin PWM

Menjalankan ADC dengan DMA

Pembacaa
Nilai AD

Perhitungan Nilai PI