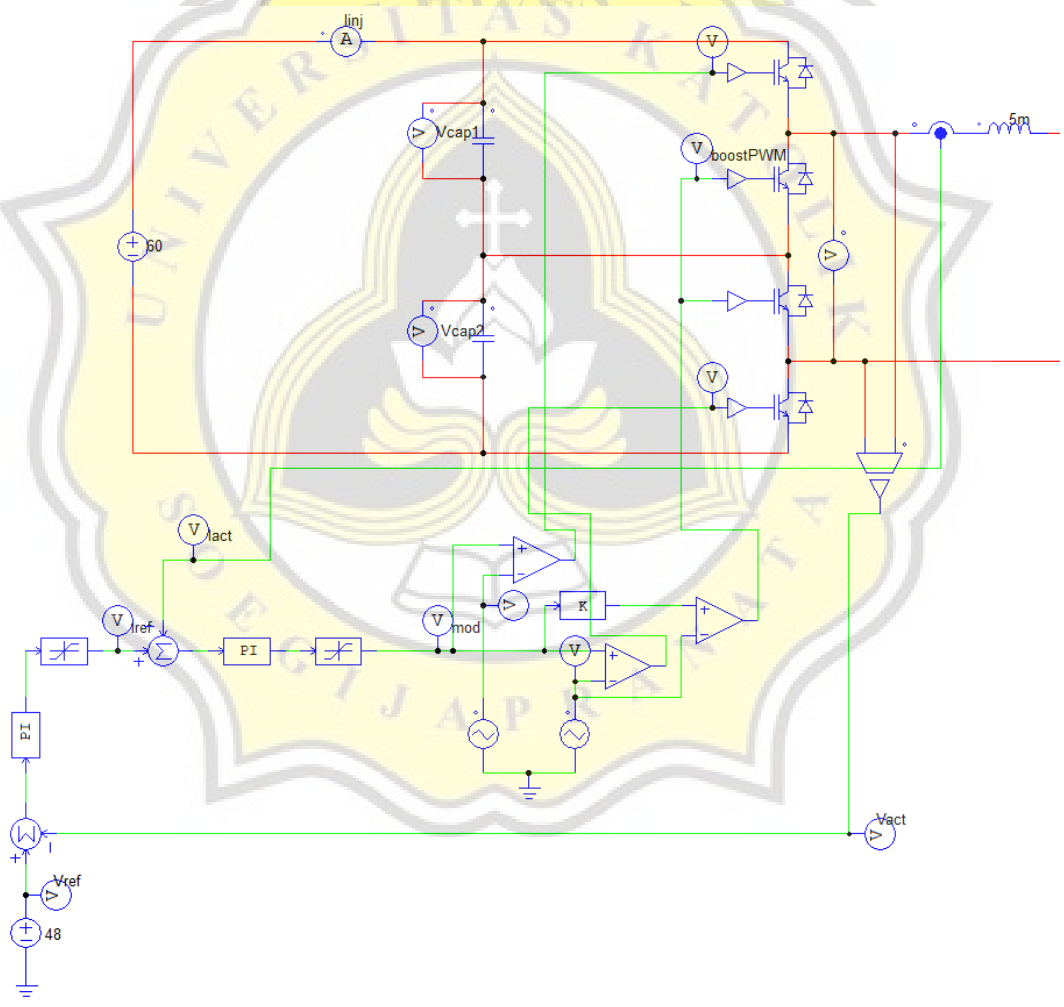
BAB III

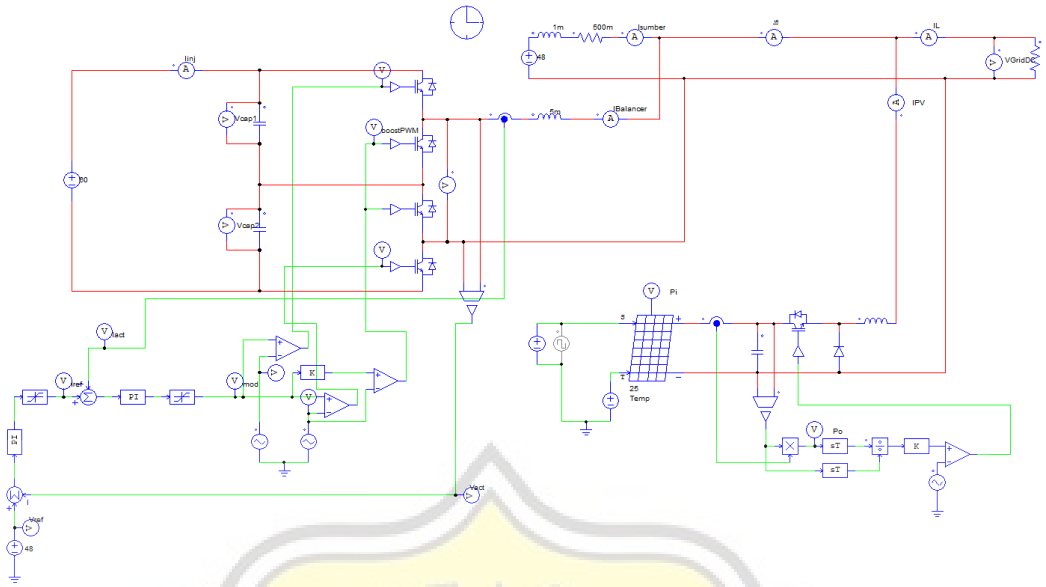
IMPLEMENTASI DESAIN

3.1 Diagram Simulasi dan *Hardware*

Sistem penyeimbang daya *microgrid* DC ini disimulasikan menggunakan aplikasi PSIM. Tujuan dibuatnya simulasi untuk menganalisa sistem dapat berfungsi sehingga pembuatan *hardware* jauh lebih mudah. Sistem yang dibuat pada simulasi adalah sistem analog seperti pada gambar dibawah.

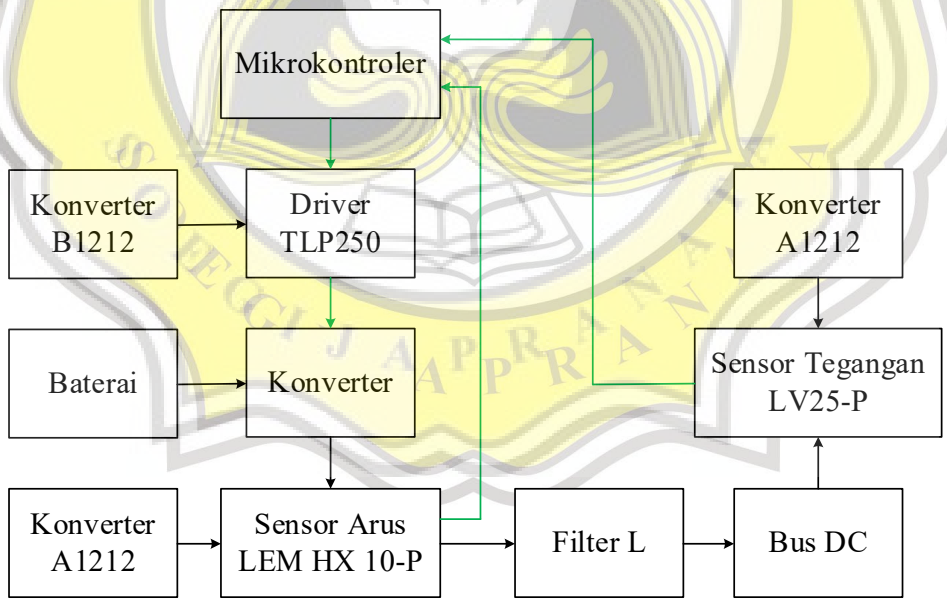


Gambar 3. 1. Rangkaian Simulasi Konverter Pada *Power Simulator*



Gambar 3. 2. Rangkaian Simulasi Sistem Microgrid DC Pada *Power Simulator*

Setelah simulasi desain berhasil, desain tersebut diimplementasikan pada sebuah *hardware* dan seluruh sistem kendalinya adalah digital karena lebih mudah pengaplikasiannya dan terjangkau.



Gambar 3. 3. Diagram blok *hardware*

Gambar 3.3 menunjukkan diagram blok sistem pada prototipe *hardware*. Pada implementasi *hardware*, nilai referensi disetel dengan menentukan nilai tegangan *bus* DC

saat tidak berbeban sehingga memerlukan proses kalibrasi nilai referensi. 4 buah TLP250 digunakan sebagai *driver* MOSFET yang masing-masing mengendalikan satu. TLP250 disuplai daya oleh konverter B1212 karena perlu terisolasi dengan TLP250 lainnya. Pada keluaran konverter terdapat sensor arus LEM HX 10-P yang dirangkai seri. Setelah itu juga terdapat tapis L yang dirangkai seri untuk menghaluskan riak arus. Tegangan *bus* DC dibaca oleh sensor tegangan LV 25-P yang terhubung paralel dengan *bus* DC. Prototipe ini menggunakan kedua sensor ini karena memberikan isolasi antara mikrokontroler dengan terminal yang diukur, sehingga mikrokontroler dapat lebih aman dari tegangan tinggi. Kedua sensor ini disuplai oleh konverter A1212 karena membutuhkan tegangan positif dan negatif dan juga memberikan isolasi. Mikrokontroler yang mengendalikan seluruh sistem kontrol prototipe penyeimbang daya adalah STM32F407VET6 karena memiliki performa yang dibutuhkan. Tabel 3.1 merupakan parameter dari desain simulasi dan implementasi *hardware*.

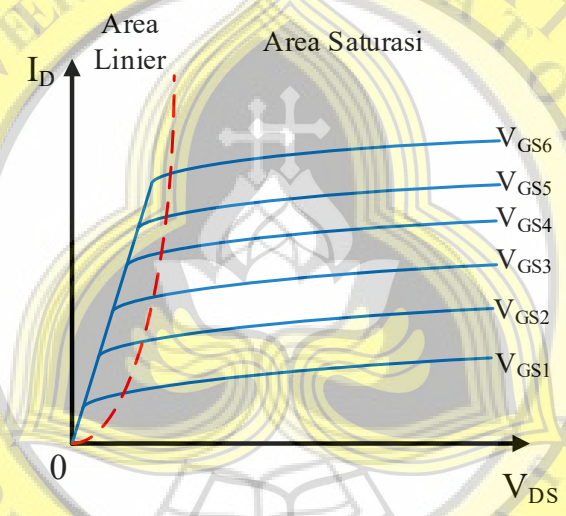
Tabel 3.1. Parameter Penelitian

Parameter	Nilai
Tegangan *microgrid* DC	48 V_{DC}
Tegangan Baterai	60 V_{DC}
Filter Induktor	5 mH
Filter Kapasitor C1, C2	220 uF
Hambatan Beban	30 Ohm
Frekuensi *Switching*	25 kHz

3.2 MOSFET (Metal–Oxide–Semiconductor Field-Effect Transistor)

Metal–Oxide–Semiconductor Field-Effect Transistor (MOSFET) merupakan perangkat semikonduktor daya yang memiliki tiga terminal yaitu *gate, drain,* dan *source.* MOSFET biasa digunakan sebagai sakelar daya elektronik yang mampu melakukan

switching yang cepat dengan efisiensi tinggi. Tegangan terminal *gate* menentukan konduktivitas antara terminal *drain* dan *source*. Untuk melakukan *switching* diperlukan tegangan tertentu sehingga hambatan *drain-source* dapat sekecil mungkin sehingga meminimalkan rugi-rugi daya. MOSFET digunakan karena dapat mengendalikan aliran daya secara efisien. Untuk menggunakan MOSFET sebagai *switching* pada rangkaian maka MOSFET bekerja dalam kondisi ON atau OFF. Jika diberikan tegangan positif antara terminal *gate* dan *source* maka MOSFET menjadi kondisi ON dan jika terminal *gate* diberi tegangan negatif atau nol, maka MOSFET akan menjadi kondisi OFF. Hal ini sama seperti *switching* pada BJT dan IGBT.



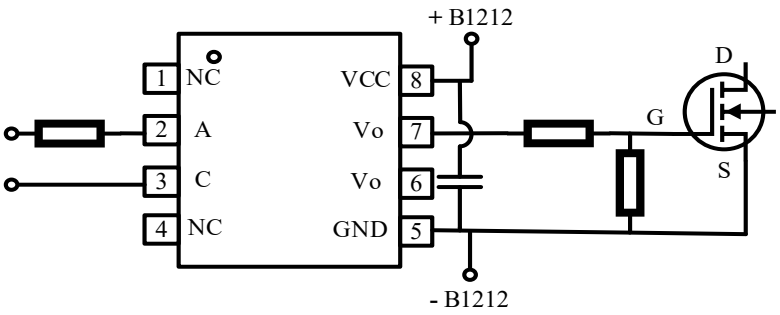
Gambar 3. 4. Kurva karakteristik MOSFET

Pada gambar 3.4 di atas menunjukkan karakteristik arus dan tegangan tergantung pada tegangan gerbangnya atau V_{GS}. Sumbu X menunjukkan tegangan *drain-source* atau V_{DS} dan sumbu Y menunjukkan arus *drain.* Selama kondisi OFF, arus yang melewati *drain* dan tegangan pada *gate* adalah nol. Dengan menaikkan tegangan V_{GS} maka menaikkan arus *drain* seperti pada gambar kurva di atas. Jenis MOSFET yang digunakan untuk implementasi *hardware* ini adalah seri IRFP460.

3.3 TLP250 *Gate Driver*

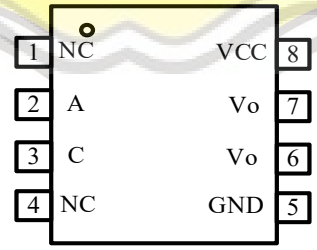
Untuk menjalankan MOSFET dibutuhkan *driver* karena sakelar daya ini merupakan perangkat terkendali tegangan. Pada terminal *gate* perangkat ini terdapat kapasitansi yang harus diisi ke tegangan tertentu supaya dapat menjadi kondisi ON dan seringkali tegangan tersebut sulit diraih oleh tegangan logika keluaran dari mikrokontroler. Pada *switching* kecepatan tinggi karena kapasitansi pada *gate* diisi dan dibuang secara cepat, maka semakin tinggi frekuensi *switching* semakin besar pula daya yang dikonsumsi oleh MOSFET /tersebut. Maka daya keluaran logika dari mikrokontroler juga sering kali tidak mampu menyuplai daya yang dibutuhkan oleh sakelar daya tersebut. Oleh karena itu rangkaian *driver* diperlukan.

TLP250 dapat digunakan sebagai *gate driver* MOSFET yang sinyal pensakelarannya terisolasi oleh rangkaian daya menggunakan *optocoupler*. *Driver* ini mengatur pensakelaran MOSFET dengan suplai daya dari B1212 yang mendapatkan sinyal dari mikrokontroler melewati LED dan fotodioda. Seperti yang diketahui untuk mengaktifkan MOSFET diperlukan tegangan antara *gate* dan *source*, maka untuk menjalankan sakelar daya yang dirangkai seri sebanyak 4 buah diperlukan sumber yang terisolasi untuk setiap MOSFETnya. Oleh karena itu terdapat 4 buah pasang B1212 dan TLP250 untuk mengendalikan 4 buah MOSFET. Rangkaian penggunaan TLP250 ditunjukkan pada Gambar 3.5 di bawah.



Gambar 3. 5. Rangkaian Penghubungan *Driver* TLP250 Dengan MOSFET

Terdapat 8 buah terminal pada TLP250, namun hanya 6 yang memiliki fungsi. Pada sisi primer, pin 2 merupakan anoda untuk LED di dalam TLP250 yang terhubung ke *port* I/O pada mikrokontroler. Karena merupakan LED, maka diperlukan sebuah resistor untuk menghambat arus. Katoda pin 3 terhubung ke terminal gnd pada mikrokontroler. Pada sisi sekundernya, VCC pin 8 terhubung ke terminal positif B1212 yang merupakan sumber tegangan yang akan mensuplai terminal *gate* pada MOSFET. GND pin 5 terhubung pada terminal *source* dari MOSFET dan terminal negatif B1212. Pin Vo 6&7 saling terhubung dan memiliki fungsi yang sama yang terhubung ke terminal *gate* untuk memberi sinyal pensakelaran berdasarkan sinyal dari fotodioda. Resistor digunakan untuk mencegah arus *inrush* yang masuk ke terminal *gate* MOSFET dan resistor *pull-down* memastikan MOSFET dalam keadaan OFF saat sinyal pensakelaran mati. Gambar 3.6 di bawah merupakan pin terminal dari TLP250.



Gambar 3. 6. Terminal pada TLP250

3.4 Mikrokontroler STM32F407VET6

Mikrokontroler adalah perangkat yang berfungsi mengerjakan perintah pemrograman yang dibuat dengan tujuan tertentu. Suatu mikrokontroler memiliki satu atau lebih inti *Central Processing Unit* (CPU) bersama memori dan peripheral I/O yang bisa diprogram. Mikrokontroler STM32F407VET6 merupakan mikrokontroler dari STMicroelectronics performa tinggi dengan dengan menggunakan prosesor ARM Cortex-M4 32bit RISC *core* yang beroperasi pada frekuensi sampai 168MHz. STM32F407VET6 memiliki memori tertanam berkecepatan tinggi. Mikrokontroler ini digunakan karena kecepatan proses yang tinggi dan pembacaan ADC yang cepat dan akurat.

Gambar 3. 7. *Board* STM32F407VET6

Fitur-fitur STM32F407VET6 antara lain:

a) CPU Arm® 32-bit Cortex®-M4 dengan FPU, akselerator Adaptive *real-time* (ART *Accelerator*) memungkinkan eksekusi kondisi tunggu 0 dari memori *flash*, frekuensi sampai dengan 168 MHz, unit proteksi memori, 210 DMIPS atau 1.25DMIPS/MHz (*Dhrystone* 2.1), dan instruksi DSP.
b) Memori *flash* hingga 1 Mbyte.

c) SRAM hingga 192+4 Kbyte termasuk 64-Kbyte CCM (*core coupled memory*) RAM data.

d) Memori OTP 512 byte.

e) Kontroler memori statik fleksibel mendukung *Compact Flash*, SRAM, PSRAM, NOR dan memori NAND.

f) Antarmuka paralel LCD.

g) Operasi daya rendah.

h) Memiliki 3×12-bit, 2.4 MSPS ADC: hingga 24 *channel* dan 7.2 MSPS dalam mode *triple interleaved.*

i) 2×12-bit DAC.

j) Kontroler DMA 16-aliran dengan dukungan FIFO dan *burst*.

k) Memiliki 17 *timer* yaitu 12 *timer* 16-bit dan 2 *timer* 32-bit kecepatan hingga 168 MHz, masing masing dengan 4 IC/OC/PWM atau *counter* pulsa dan *input encoder* quadratur (*incremental*).

l) Memiliki 140 pot I/O dengan kemampuan interupsi, terdiri dari 136 fast I/O hingga 84 MHz dan 138 I/O 5V-toleran.

m) Memiliki 15 antarmuka komunikasi yaitu 4 USART, 3 SPI, 3 I2C, 2 UART, SDIO, dan 2 CAN.

n) Antarmuka kamera paralel 8-bit sampai 14-bit hingga 54 Mbyte/s.

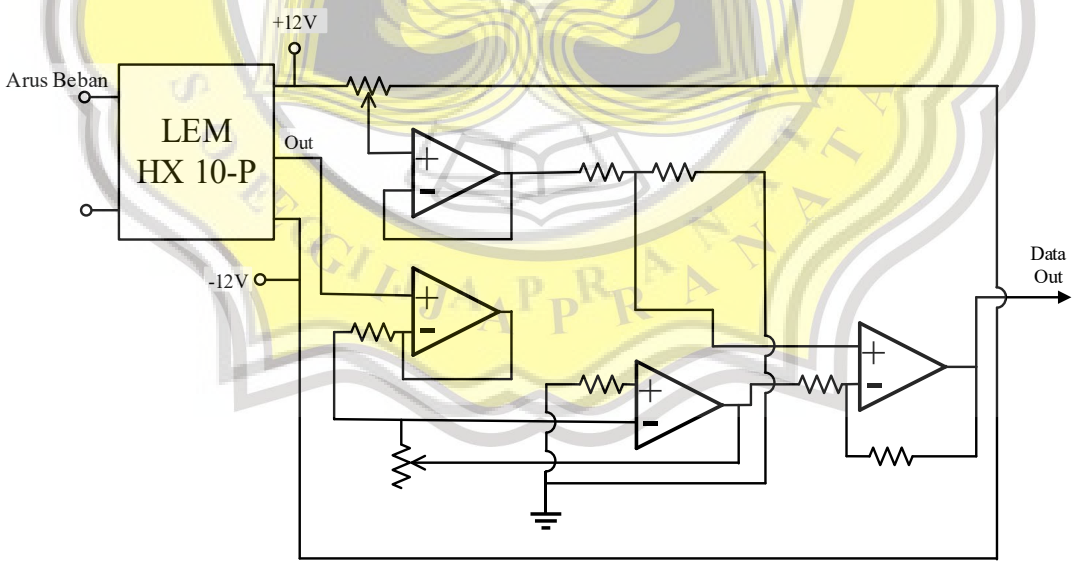
o) Generator angka acak.

p) Unit kalkulasi CRC

q) RTC dengan akurasi *subsecond,* kalender *hardware*.

3.5 Sensor Arus LEM HX 10-P

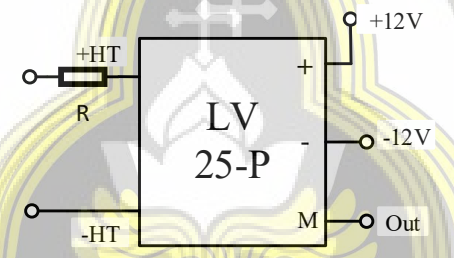
Pada sistem kendali konverter ini supaya mikrokontroler dapat membaca besarnya arus aktual pada terminal *output* konverter, maka dibutuhkan sensor arus. LEM HX 10-P digunakan sebagai sensor arus pada konverter ini dengan *rating* pengukuran hingga 10 Amper. Sensor arus ini mengubah besaran arus yang diukur menjadi suatu besaran tegangan supaya dapat didukung oleh ADC mikrokontroler. Sensor arus ini memberikan isolasi antara sisi *input* dengan sisi *output* nya sehingga mikrokontroler aman dari tegangan tinggi atau lonjakan tegangan dari keluaran konverter. Sensor arus ini masih membutuhkan rangkaian *Op-Amp* untuk menguatkan sinyal keluarannya dan dengan rangkaian *Op-Amp* ini offset dan amplitudo dari sinyal aktual tersebut dapat diatur supaya berada di dalam jangkauan pembacaan ADC mikrokontroler. Rangkaian sensor arus ini membutuhkan catu daya +12V dan -12V untuk dapat beroperasi. Gambar 3.8 di bawah merupakan rangkaian penguat sensor arus LEM HX 10-P beserta dengan *Op-Amp* nya.



Gambar 3. 8. Rangkaian Penguat Pada LEM HX 10-P

3.6 Sensor Tegangan LV25-P

Supaya mikrokontroler dapat dapat membaca besarnya tegangan aktual, maka sensor tegangan LV25-P ini digunakan. Sensor tegangan ini memberikan isolasi antara rangkaian yang diukur dengan keluarannya, sehingga mikrokontroler dapat dengan aman mengukur tegangan tinggi. Rentang pengukuran sensor ini bergantung pada besarnya resistor R. Sensor ini memiliki *rating* arus masukannya sebesar 10 mA, sehingga jika batas tegangan maksimal yang ingin diukur adalah 100V, maka diperlukan resistor 10 kΩ. Supaya keluaran sensor ini dapat mendukung tegangan pada ADC mikrokontroler, maka diperlukan rangkaian *amplifier* pada keluaran sensor ini. Diagram sensor tegangan LV25-P ditunjukkan pada Gambar 3.9.



Gambar 3. 9. Diagram Sensor Tegangan LV25-P

3.7 B1212S & A1212 Isolated Converter

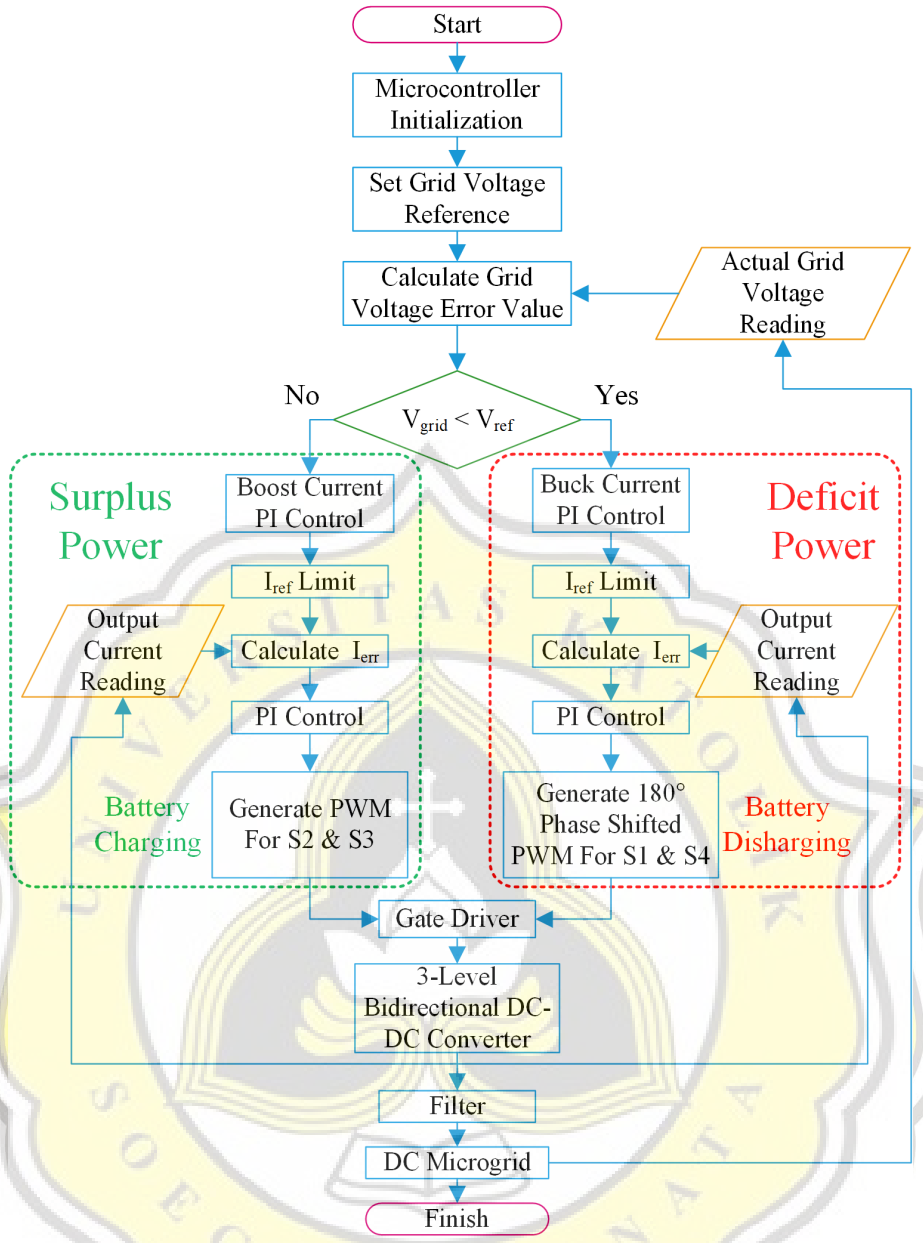
Konverter ini berfungsi untuk menghasilkan tegangan DC yang terisolasi dengan sumbernya. Keluaran dari konverter ini belum teregulasi sehingga masih dibutuhkan rangkaian. Rangkaian regulator dapat diberikan pada bagian *input* nya atau pada bagian *output*. Beban dari konverter ini tidak boleh kurang dari 10% dari beban *rating* nya, sehingga untuk beban yang sangat kecil diperlukan menambahkan resistor beban pada bagian *output*.

Pada implementasi *hardware,* rangkaian konverter ini berfungsi untuk menghasilkan tegangan +12V dan -12V yang dibutuhkan oleh sensor arus. Karena

keluaran dari konverter ini terisolasi dari sumbernya, sehingga terminal positif sumber dapat dihubungkan ke terminal negatif *output* konverter ini atau sebaliknya untuk menghasilkan tegangan positif dan negatif dengan terminal ground pada sambungan terminal sumber dan terminal *output* konverternya yang telah dihubungkan. Perbedaan antara B1212 dengan A1212 terletak pada terminal *output.* A1212 memberikan tegangan *output* negatif dengan total 3 buah terminal (+12, 0, -12) sedangkan B1212 hanya memiliki 2 buah terminal (+12, 0). Karena sensor yang digunakan memerlukan tegangan negatif, maka digunakan konverter A1212.

3.8 Algoritma Pemrograman

Seluruh sistem kendali penyeimbang daya dibuat secara digital ke dalam suatu program yang diproses oleh suatu mikrokontroler. Gambar 3.10 menggambarkan alur cara kerja program dalam suatu *flowchart*.



Gambar 3. 10. *Flowchart* pemrograman sistem

Awalnya, referensi tegangan *bus* DC perlu dikalibrasi agar penyeimbangan daya dapat dilakukan dengan akurat. Mikrokontroler membaca variabel tegangan dan arus aktual menggunakan *analog to digital converter* (ADC) internal bawaannya. Ketika tegangan *bus* DC di bawah referensi, ia akan memasuki operasi mode *buck* untuk memasok daya ke sistem. Sebalikya akan memasuki operasi mode *boost* dan mengisi daya baterai. Nilai *error* tegangan diproses menggunakan kontroler PI yang kemudian

digunakan sebagai nilai referensi arus. Nilai referensi arus tersebut dibatasi terlebih dahulu nilai maksimalnya sesuai dengan yang diperlukan untuk keamanan. Peregulasian arus dilakukan menggunakan kontroler PI kedua yang kemudian digunakan untuk menghasilkan sinyal PWM untuk pensakelaran. Untuk operasi *buck*, ini akan menghasilkan dua PWM yang bergeser fasa. Sinyal PWM yang dihasilkan oleh mikrokontroler diteruskan ke *gate driver* untuk amplifikasi dan dapat mengontrol sakelar daya semikonduktor dari konverter.

Pemrograman tersebut ditulis dalam bahasa C/C++ pada *software* Arduino IDE yang disajikan di bawah ini.

```
include <STM32F4ADC.h>
```

Pemanggilan *Header Library*

```
#define set PD5 //saklar set
#define vrefup PD3 //saklar + vref
#define vrefdown PD2 //saklar - vref
```

Penerjemahan nama *pin*

```
STM32ADC inADC(ADC1);
```

Inisialisasi ADC

```
int iact,iload,ipv,vact,err,pot,iref;
int err_v;
double itg,lastitg,pi,mod,Bmod,P,I;
double vref= 2500, itg_v, lastitg_v, pi_v, P_v,I_v;
uint8_t analog_pins[] = {PA0, PA1, PA2, PA3,
PA4, PA5, PA6};
```

Deklarasi data variabel

```
//control
float kp_v=0.5;
float ki_v=20;
float kp_i=0.5;
float ki_i=80;
```

Deklarasi konstanta setelan kontrol PI

```
void setup() {
```

Program *setup*

```
Serial.begin(9600);
```

Inisialisasi *bitrate port serial*

```
pinMode(set,INPUT_PULLUP);
pinMode(vrefup,INPUT_PULLUP);
pinMode(vrefdown,INPUT_PULLUP);
for (uint8_t x = 0; x<sizeof(analog_pins);
x++)
```

Inisialisasi mode *port I/O*

```
Timer5.init();
Timer5.pause();
Timer5.setPeriod(10000);
Timer5.setMode(TIMER_CH2, TIMER_OUTPUT_COMPARE);
Timer5.setCompare(TIMER_CH2, 1);
Timer5.attachInterrupt(TIMER_CH2,ITR);
Timer5.refresh();
Timer5.resume();
```

Inisialisasi *Timer* 5 untuk *Interrupt*

```
Timer3.init(); //PWM timer
Timer3.setPeriod(20);
Timer3.refresh();
Timer4.init();
Timer4.setPeriod(20);
Timer4.refresh();
Timer4.setCount(1689);
Timer3.resume();
Timer4.resume();
```

Inisialisasi *Timer* 3 & 4 untuk PWM

Timer 4 diberi setCount 1689 karena dengan periode 20 uS dengan *clock speed* 168 MHz maka nilai atas *timer count* adalah 3360, sehingga timer 4 akan menghitung mulai dari nilai tengah dan membuat seolah-olah menggeser sinyal *carrier sawtooth* 180° antara *timer count* 4 dan *timer count* 3, maka kedua timer ini berfungsi untuk membangkitkan

sinyal PWM yang tergeser 180°. Namun timer 3 dan 4 hanya memiliki frekuensi *clock* setengah dari *clock* utama, namun nilai atas *count* tetap, sehingga frekuensi PWM yang dihasilkan menjadi setengahnya. Dengan periode 20 uS maka frekuensi PWMnya 50 kHz / 2 menjadi 25 kHz.

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

Inisialisasi *port* timer PWM

```
}
```

```
void ITR(){
```

Program *Timer Interrupt*

```
if (digitalRead(set)==0) // set v reference
 {
  vref = map(analogRead(PA4),0,4095,0,4000);
//v actual, for reference before voltage drop
 }
 if(digitalRead(vrefup)==0)
 {vref++;}
  if(digitalRead(vrefdown)==0)
 {  vref--; }
 if(vref>3900)
 {vref=3900;}
 if(vref<0)
 {vref=0;}
```

Input kalibrasi referensi tegangan

```
 Serial.print(vref);
 Serial.print("  ");
 Serial.print(iact);
 Serial.print("  ");
 Serial.println(mod);
```

Tampilkan variabel melalui komunikasi serial

```
}
```

```
void loop() {
```

Program *Loop*

```
  while(1){
  sensor(); //sensor read
  bidirect(); //control process
  }
```

Program *While* untuk pembacaan sensor dan proses kendali

```
}
```

```
void sensor(){
 iact = map(analogRead(PA1),0,4095,-
2000,2000); //i actual
 vact = map(analogRead(PA4),0,4095,0,4000);
//v actual
}
```

Program pembacaan sensor

```
void bidirect() {
```

Program kendali dan pensakelaran

```
   err_v = vref-vact;
   P_v= kp_v*err_v;
   itg_v = lastitg_v + err_v*0.0001;
   I_v=ki_v*itg_v;
   pi_v= P_v+I_v;
```

Loop PI tegangan

```
if(pi_v>-2000 && pi_v<2000)  //current anti
windup
   {lastitg_v=itg_v;}
```

Anti windup PI tegangan

Anti windup berfungsi menghentikan integrasi saat mencapai nilai batas sehingga mengurangi terjadinya *overshoot*.

```
iref= pi_v ;
err = iref-iact;
P = kp_i*err;
itg = lastitg+err*0.0001;
I=ki_i*itg;
pi=P+I;
```

Loop PI arus

```
mod = pi;
if(mod<-2500)  //limiter
mod= -2500;
if(mod>-2500 && mod<3360)//mod anti windup
  {lastitg=itg;}
```

Anti windup PI arus

```
Bmod = -mod; //boost mod
```

Nilai invers modulasi untuk konversi *boost*

```
pwmWrite(PB6,mod); //S1
pwmWrite(PB7,Bmod); //S2
pwmWrite(PB8,Bmod); //S3
pwmWrite(PB0,mod); //S4
```

Eksekusi *timer PWM*

```
}
```