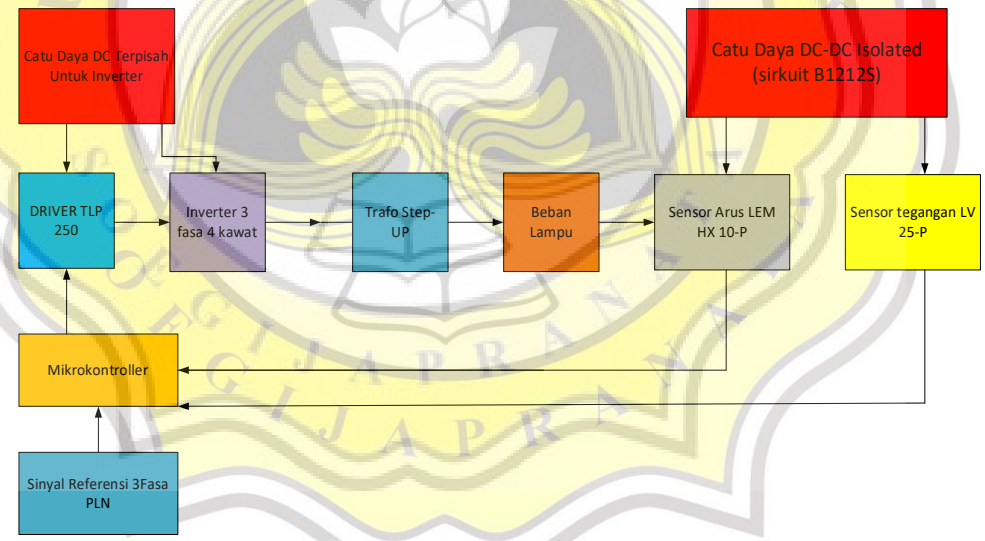
BAB III

DESAIN DAN IMPLEMENTASI

3.1 Pendahuluan

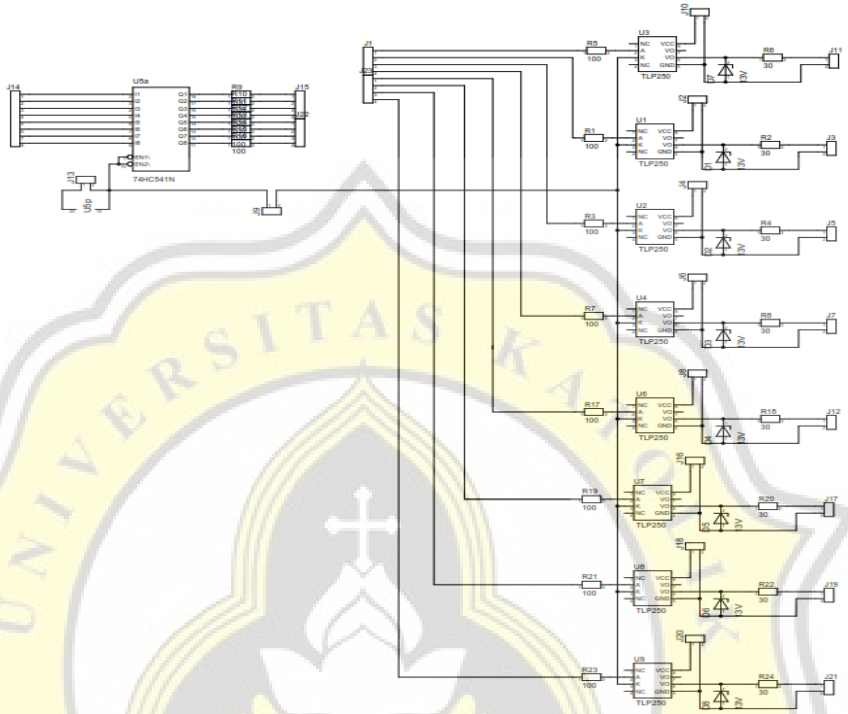
Pada tugas akhir ini akan membahas tentang perancangan dan implementasi inverter 3 Fasa 4 kawat (3F4K) dan beberapa rangkaian yang mendukung dalam berjalannya alat. Rangkaian pendukung tersebut terdiri dari rangkaian *optocoupler driver* TLP 250, rangkaian sensor arus LEM HX- 10 P, Rangkaian tegangan LV-25P, kemudiang rangkaian Catudaya B1212S, dan Rangkaian Inverter tiga fasa empat kawat.



Gambar 3. 1 Diagram Blok Inverter perancangan inverter

Berdasarkan gambar 3.1, sistim yang dipakai dalam *double-loop control* pada pengaplikasian Inverter 3 fasa 4 kawat diatas adalah dengan menggunakan sistim kendali *closed-loop* dimana kita dapat mengatur frekuensi sesuai yang kita inginkan melalui perintah dari mikrokontroller STM32F407. Untuk catu daya yang digunakan disini adalah catudaya tegangan DC untuk mensuplai *Driver* TLP 250 yang nantinya telah berisi algoritma pemrograman yang bersasal dari mikrokontroller dan sinyal referensi yang berasal dari sumber PLN mikrokontroller juga terhubung dengan sensor arus dan tegangan agar nantinya kita bisa meregulasi tegangan dan arus dengan program, dari *driver* akan dialirkan menuju rangkaian inverter agar terjadi keluaran sinyal AC yang dimana sinyal AC tersebut akan diringkatkan tegangannya dari 30V menjadi 220V dengan *trafo step-up*, kemudian diaplikasikan pada beban lampu.

3.2 Rangkaian TLP 250



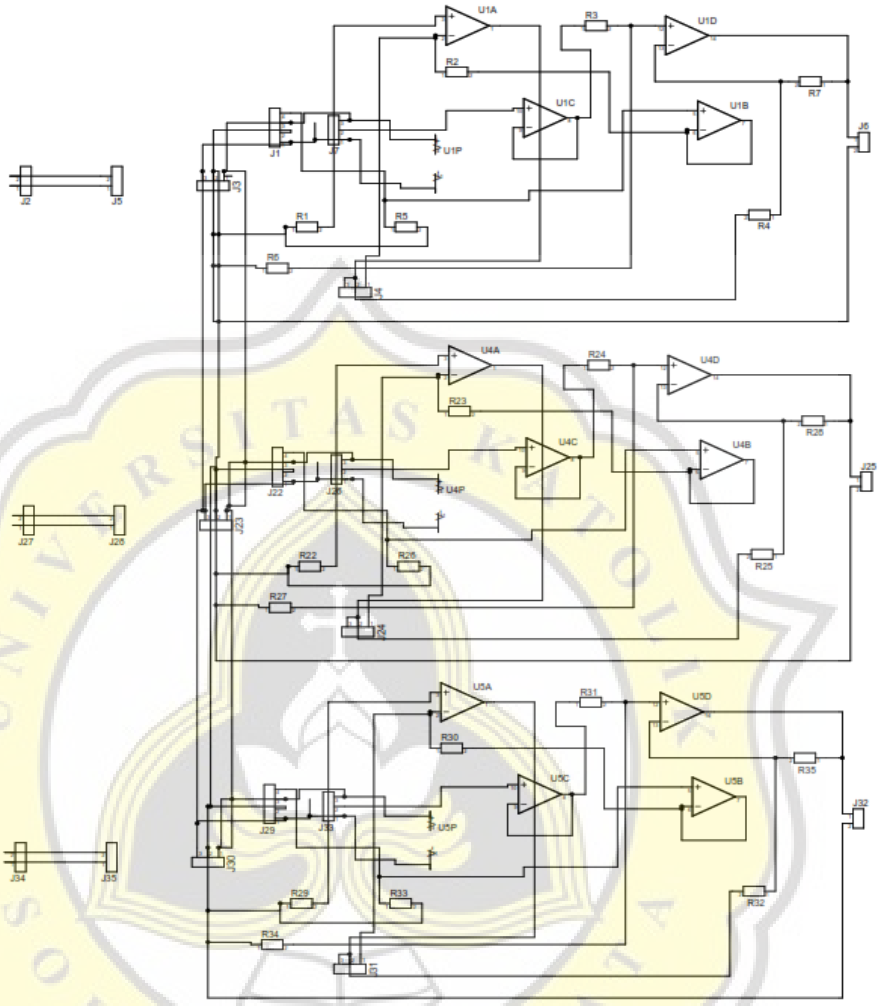
Gambar 3. 2 Rangkaian Driver TLP 250 Pada Inverter 3F4K

Rangkaian TLP 250 atau dapat disebut *optocoupler driver* digunakan sebagai bentuk *gate drive* pada IGBT karena memerlukan teganagan masukan 10VDC. Rangkaian ini juga berfungsi untuk membuat tegangan ter-*isolated* agar nantinya tegangan pada STM32F407 dan rangkaian daya menjadi terpisah. Kemudian gelombang yang dihasilkan akan dihubungkan dengan penguat atau *buffer*, dari rangkaian *driver* TLP 250 akan dihubungkan pada rangkaian kaki *gate* pada IGBT. Dapat kita lihat pada pin HIN dan LIN pada *driver* dapat dikoneksikan dengan gelombang keluaran yang berasal dari

buffer, kemudian pada bagian HO dan LO akan dihubungkan pada kaki MOSFET Seperti gambar 3.2.

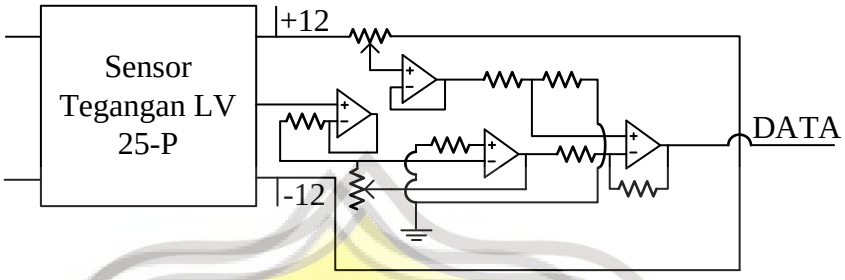
3.3 Rangkaian Sensor Arus LEM HX-10P

Pada inverter 3F4K sensor arus LEM HX-10P digunakan untuk deteksi arus yang nantinya digunakan sebagai komparasi antara gelombang referensi dan gelombang aktual. Pada jenis sensor arus LEM HX-10P dapat membaca arus makmsimal hingga 10A, pada sensor LEM HX- 10P dapat menghasilkan keluaran tegangan ketika dialiri dengana arus. Akan tetapi tegangan yang dihasilkan sangatlah kecil karena dibutuhkan peningkatan tegangan sehingga dibutuhkan *buffer* agar terjadinya penguatan tegangan sehingga nantinya sensor dapat diregulasi sesuai dengan tegangan yang diinginkan. Untuk tegangan yang dibutuhkan sensor arus adalah +12 VDC -12VDC, ground.



Gambar 3. 3 Rangkaian Sensor LEM HX 10-P

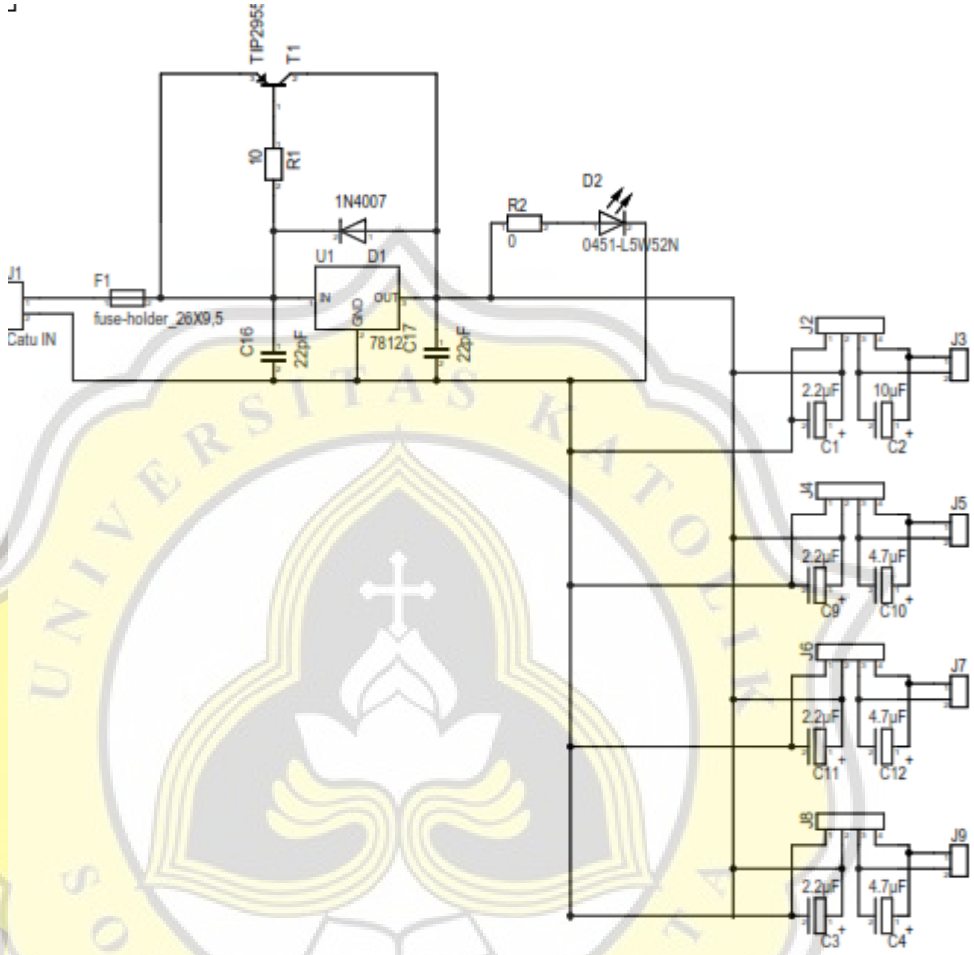
3.4 Rangkaian Sensor Tegangan LEM LV 25-P



Gambar 3. 4 Rangkaian Catudaya B1212S

Pada inverter 3F4K sensor tegangan LEM LV-25P digunakan untuk deteksi tegangan yang nantinya digunakan sebagai komparasi antara gelombang referensi dan gelombang aktual .Hampir sama seperti sensor arus pada jenis sensor tegangan LEM LV-25P dapat membaca tegangan maksimal hingga 25V, pada sensor LEM LV- 25P dapat menghasilkan keluaran tegangan ketika dialiri dengan arus. Akan tetapi tegangan yang dihasilkan sangatlah kecil karena dibutuhkan peningkatan tegangan sehingga dibutuhkan *buffer* agar terjadinya penguatan tegangan sehingga nantinya sensor dapat diregulasi sesuai dengan tegangan yang diinginkan. Untuk tegangan yang dibutuhkan sensor arus adalah +12 VDC -12VDC, ground. Ditampilkan pada gambar 3.4

3.5 Rangkaian Catu daya B1212S

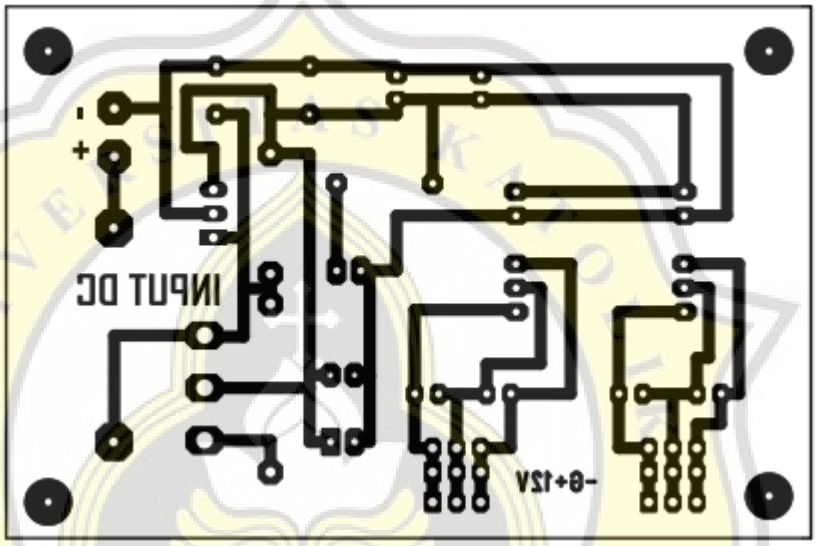


Gambar 3. 5 Rangkaian Catudaya B1212S

Pada Inverter 3F4K menggunakan rangkaian catu daya B1212S sebagai penyuplai daya tegangan secara relative konstan pada gambar 3.5 yang ditampilkan menunjukan rangkaian B1212S sebagai catudaya dengan tiga keluaran yaitu +12VDC, -12VDC dan +5VDC. Disini dikarenakan mikrokontroler harus menggunakan suplai tegangan 5V kita menggunakan

sumber +5VDC untuk dialirkan pada mikrokontroller dan untuk 12 V akan disuplai menuju *octocoupler*.

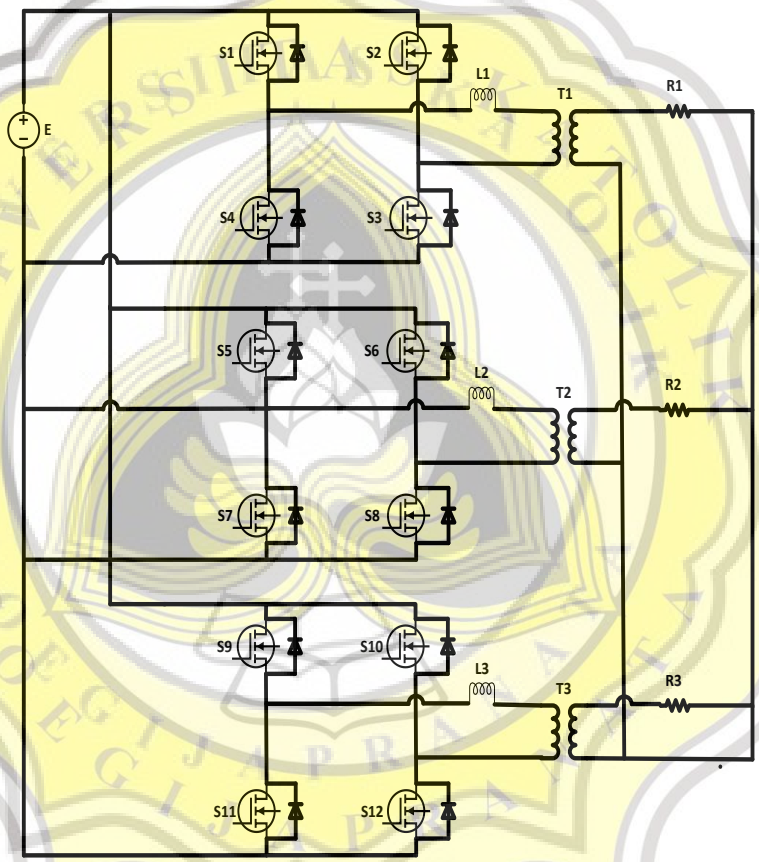
3.6 Rangkaian Daya Catudaya A1212S



Gambar 3. 6 Rangkaian Catudaya B1212S

Inverter 3F4K menggunakan rangkaian A1212 sebagai converter catu daya terisolasi pada gambar 3.6 ditampilkan bahwa terdapat 6 output keluaran pada rangkaian A1212S yang terdiri dari +12V, ground, -12 pada projek ini A1212S digunakan untuk mensuplai tegangan pada sensor arus dan tegangan agar dapat mengirimkan tengan secara konstan dan tidak mengalami voltage drop sehingga diperlukan rangkaian catudaya isolated A1212S.

3.7 Rangkaian Daya *Inverter* tiga fasa empat kawat

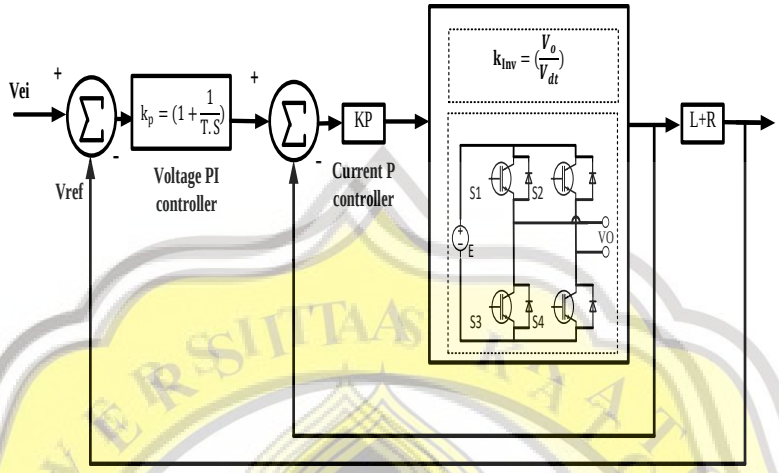


Gambar 3. 7Rangkaian daya Inverter 3F4K

Rangkaian daya *Inverter* tiga fasa empat kawat (3F4K) menggunakan mosfet FGH40N60SFD sebagai sakelar daya dengan *rate* tegangan maksimum yang dapat dicapai hingga 600V dan arus maksimal 80A. Rangkaian daya *inverter*

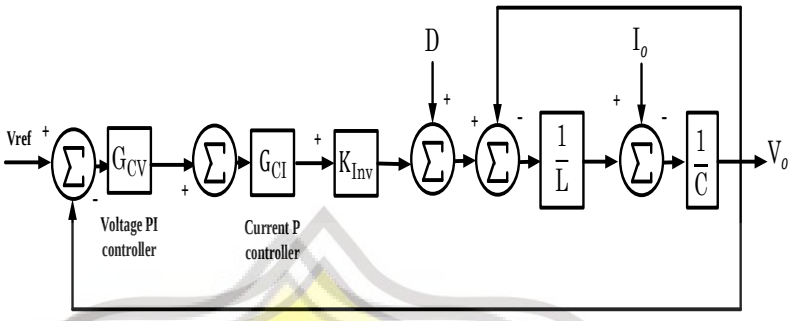
3F4K menggunakan satu buah sumber DC yang dipasang secara seri. Tegangan keluaran setiap *inverter* akan dihubungkan menuju lilitan trafo *step-up* dan pada masing masing sisi lilitang primer pada trafo akan dihubungkan sebuah beban lampu sebesar 100 Watt. Untuk pembentukan tiap fasa menggunakan *inverter* 3F4K yang disusun secara pararel , sengan sebuah sinyal referensi yang saling bergeser 120˚. Tegangan keluaran yang dihasilkan setiap inverter akan dihubungkan dengan trafo *step-up* yang berfungsi untuk menaikan tegangan menjadi 220V serta menggabungkan setiap fasa yang terpisah menjadi satu. Pembentukan fasa netral terdapat pada keluaran trafo yang dihubungkan secara pararel sehingga membentuk tiga fasa empat kawat.

3.8 Strategy Kontrol *Double-loop*

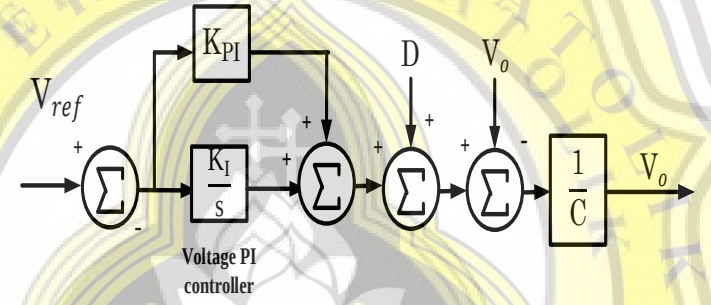


Gambar 3. 8 Rangkaian control double loop

Sistem kendali *double-loop* pada inverter 3F4K dapat dilihat pada gambar 3.8 dimana tegangan dan arus saling teregulasi untuk menghasilkan gelombang sinusoidal yang maksimal pada Inverter 3F4K. Prinsip kerja pada rangkaian ini adalah saat adanya tegangan sumber DC maka akan mengalir menuju sensor tegangan dimana terjadi pembacaan nilai tegangan menggunakan algoritma PI dan dibandingkan dengan tegangan referensi sehingga penjumlahan nilai tersebut dialirkan menuju sensor arus untuk diproses dengan algoritma P yang dimana nilai tersebut digunakan untuk proses pada *switching inverter* sehingga gelombang keluaran akan di filter (l+R) dan membentuk gelombang sinusoidal.



Gambar 3. 9 Model System Block Diagram



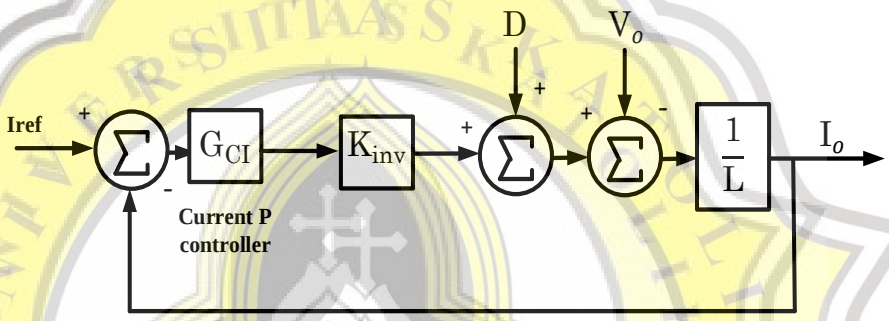
Gambar 3. 10 Model System Voltage Control

Pada gambar 3.10 ditampilakn sistem model dari block diagram *double loop control* dimana tegangan dikontrol dengan kendali PI (G_{CV}) kemudian terdapat komparasi nilai P (G_{CI}) untuk memproses sinyal output PWM keluaran dari *inverter* K_{INV} sehingga di proses dengan filter untuk membentuk gelombang AC. Pada gambar 3.11 ditampilkan sebuah kendali tegangan block diagram yang dimana dapat digunakan sebagai acuan untuk

membuat tegangan yang stabil untuk itu ditampilkan sebuah persamaan sebagai berikut.

$$\frac{V_O}{V_{ref}} = \frac{SK_p + K_i}{S^3tC + S^2C + S.K_P + K_i} \tag{3.1}$$

$$\frac{V_O}{V_D} = \frac{S^2t + S}{S^3tC + S^2C + SK_p + K_i} \tag{3.2}$$



Gambar 3. 11 Model System Current Control

Nilai dari hasil komparasi kendali P (G_{CI}) digunakan untuk memproduksi arus teregulasi yang dialiri pada *inverter* sehingga menampilkan sebuah persamaan sebagai berikut.

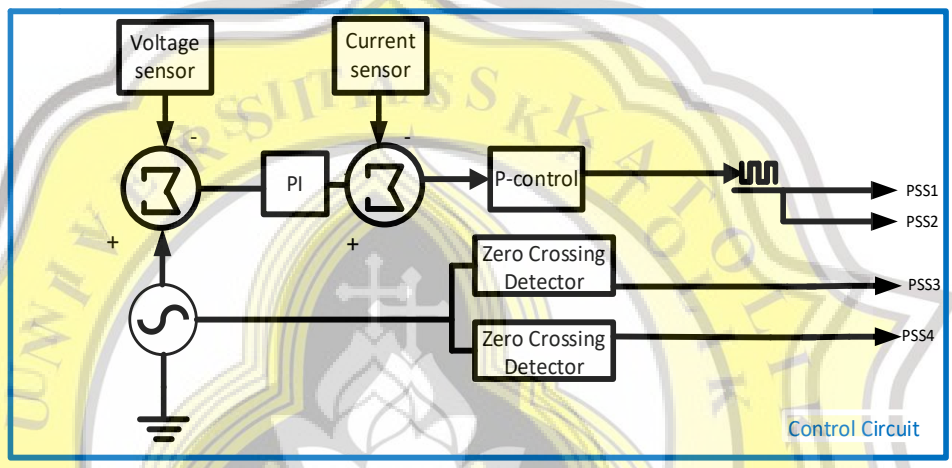
$$\frac{I_O}{I_{ref}} = \frac{G_{CI}.K_{INV}}{L + G_{CI}.K_{INV}} \tag{3.3}$$

$$\frac{I_o}{I_{ref}} = \frac{K_{INV}.K_P}{L + K_p.K_{INV}} \tag{3.4}$$

Dari kalkulasi nilai arus tersebut dapat kita komparasi pada inverter untuk memproduksi gelombang sinyal keluaran dengan persamaan.

$$K_{INV}=(\frac{V_o}{V_{d_t}}) \quad (3.5)$$

3.9 Algoritma Pemrograman

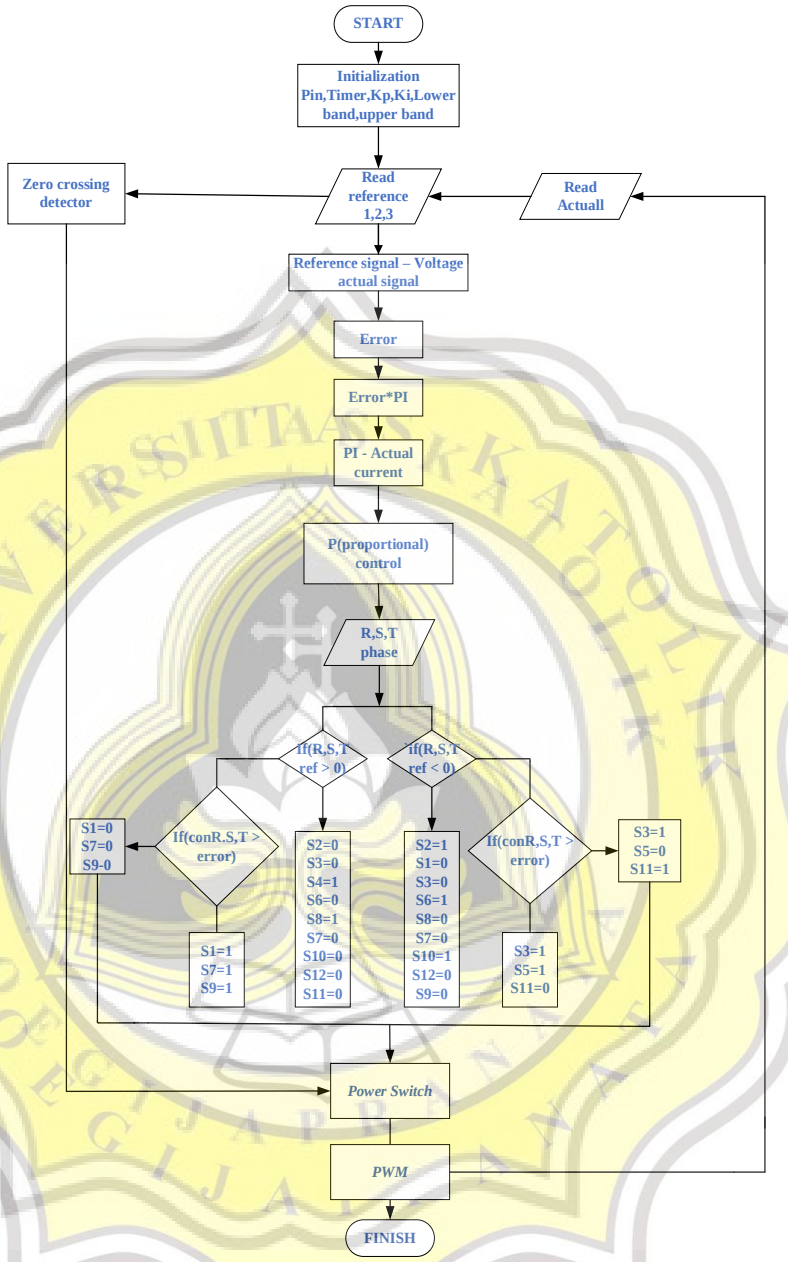


Gambar 3. 12 Kendali Inverter Double-loop

Rangkaian kendali dibutuhkan agar terjadinya proses pensaklaran pada inverter pada gambar 3.12 dapat kita lihat bahwa strategi kontrol terbagi menjadi dua yaitu rangkaian daya dan rangkaian kendali, rangkaian daya terdiri dari empatt sakelar (PSS1-PSS4), dapat kita lihat pada gambar 3.12 bahwa sumber DC akan dibaca oleh sensor tegangan dengan algoritma PI yang dimana nantinya angka dari algoritma tersebut akan di komparasikan. Agara nantinya nilai arus yang terkendali bisa di proses dengan kendali proportional (P) agar terjadi proses *switching* pada PSS1 dan PSS2 kemudian

melalui algoritma makan PSS3 dan PSS4 akan beroprasi dengan *zero crossing detector* agar menghasilkan mode unipolar pada inverter.

Gambar 3.13 menampilkan gambar flowchart pada *inverter3F4K double-loop control* dimana nilai referensi yang diberikan dari sumber PLN dibandingkan dengan nilai aktual tegangan dan juga dalam pembacaan referensi tejadi logika ZCD agar menghasilkan mode unipolar pada inverter. Kemudian hasil komparasi tersebut dikalkulasikan dengan strategi kontrol PI lalu hasil tersebut dikurangi dengan nilai aktual arus. Sehingga hasil akhir akan diproses meenggunakan strategi kontrol proportional (P) untuk mengatur pensaklaran.



Gambar 3. 13Flowchart Program Inverter Double-loop

```
#include <STM32F4ADC.h>

#define H1 PB9

#define L1 PB8

#define H3 PB7

#define L3 PB6

#define H5 PD14

#define L5 PD13

#define H2 PD7

#define L2 PD6

#define H4 PD5

#define L4 PD4

#define H6 PD12

#define L6 PD11
```

Header identifier

```
STM32ADC inADC(ADC1);

int count,car,conr,cons,cont;

int R_ref,S_ref,T_ref;

int R_actv,S_actv,T_actv;

int R_acti,S_acti,T_acti;

int err,I,lastI,pi,f;

float kp=1;
```

Deklarasi Variable dan Konstanta

```
float ki=1;

int SB = 1;

uint16_t analog_pins[] = {PA0, PA1, PA2, PA3, PA4, PA5, PA6, PA7, PC4,
PC5};
```

```
void PINMODE() {
 pinMode(H1, OUTPUT);
 pinMode(H2, OUTPUT);
 pinMode(L1, OUTPUT);
 pinMode(L2, OUTPUT);
 pinMode(H3, OUTPUT);
 pinMode(H4, OUTPUT);
 pinMode(L3, OUTPUT);
 pinMode(L4, OUTPUT);
 pinMode(H5, OUTPUT);
 pinMode(H6, OUTPUT);
 pinMode(L5, OUTPUT);
 pinMode(L6, OUTPUT);
}
```

Inisialisasi mode pin pada sakelar

```
}
void setup() {
 //Serial.begin(9600);
 PINMODE();
 for (uint16_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(20);
 Timer2.setMode(TIMER_CH2,
TIMER_OUTPUT_COMPARE);
 Timer2.setCompare(TIMER_CH2, 1);
 Timer2.attachInterrupt(TIMER_CH2,INT1);
 Timer2.refresh();
```

Sub Program Timer Counter

```
inADC.setSamplingTime(ADC_SMPR
_3);
 inADC.enableDMA();
}
void loop() {
 R();
T();
}
 S();
```

Pembacaan ADC dan time sampling

```
int kontrol(int ref, int act, int
acti){
 err = ref - act;
 I = ki*err*lastI*0.00001;
 lastI = I;
 pi = (kp*err) + I;
 f = pi - acti;
}
```

Deklarasi Sistem Kendali

```
void INT1(void){

 R_ref = map(analogRead(PC4),0,4095,-
4000,4000);//referensi

 R_actv = map(analogRead(PA6),0,4095,-
4000,4000);//actual

 R_acti = map(analogRead(PA2),0,4095,-4000,4000);

 conr = kontrol(R_ref,R_actv,R_acti);


 S_ref = map(analogRead(PC5),0,4095,-
2000,2000);//referensi

 S_actv = map(analogRead(PA5),0,4095,-
4000,4000);//actual

 S_acti = map(analogRead(PA1),0,4095,-4000,4000);

 cons = kontrol(S_ref,S_actv,S_acti);


 T_ref = map(analogRead(PA3),0,4095,-
2000,2000);//referensi

 T_actv = map(analogRead(PA4),0,4095,-
4000,4000);//actual

 T_acti = map(analogRead(PA0),0,4095,-4000,4000);

 cont = kontrol(T_ref,T_actv,T_acti);

}
```

Pembacaan nilai ADC Pada tiap fasa

```
void R(){
  if (R_ref > 0){
    digitalWrite(H2,0);
    digitalWrite(L2,1);
    digitalWrite(L1,0);
    if(conr > SB)
      digitalWrite(H1,1);
    if(conr < -SB)
     digitalWrite(H1,0);
  }
  if (R_ref < 0){
    digitalWrite(H2,1);
    digitalWrite(L2,0);
    digitalWrite(H1,0);
    if(conr > SB)
      digitalWrite(L1,0);
    if(conr < -SB)
      digitalWrite(L1,1);
  }
}
```

Pensaklaran saklar R

```
void S(){
digitalWrite(H3,0);
 digitalWrite(H4,0);
 digitalWrite(L3,0);
 digitalWrite(L4,0);
 if (S_ref > 0){
  digitalWrite(H4,0);
  digitalWrite(L4,1);
  digitalWrite(L3,0);
  if(cons > SB)
   digitalWrite(H3,1);
  if(cons < -SB)
   digitalWrite(H3,0);
 }
 if (S_ref < 0){
  digitalWrite(H4,1);
  digitalWrite(L4,0);
  digitalWrite(H3,0);
  if(cons > SB)
    digitalWrite(L3,0);
  if(cons < -SB)
    digitalWrite(L3,1);
 }
```

Pensaklaran saklar S

```
}
void T(){
  digitalWrite(H5,0);
  digitalWrite(H6,0);
  digitalWrite(L5,0);
  digitalWrite(L6,0);
  if (T_ref > 0){
    digitalWrite(H6,0);
    digitalWrite(L6,1);
    digitalWrite(L5,0);
    if(cont > SB)
     digitalWrite(H5,1);
    if(cont < -SB)
     digitalWrite(H5,0);
  }
  if (T_ref < 0){
    digitalWrite(H6,1);
    digitalWrite(L6,0);
    digitalWrite(H5,0);
    if(cont > SB)
      digitalWrite(L5,0);
    if(cont < -SB)
    digitalWrite(L5,1);
}
```

Pensaklaran saklar T