# BAB III

# DESAIN RANCANGAN DAN IMPLEMENTASI INVERTER 5-TINGKAT

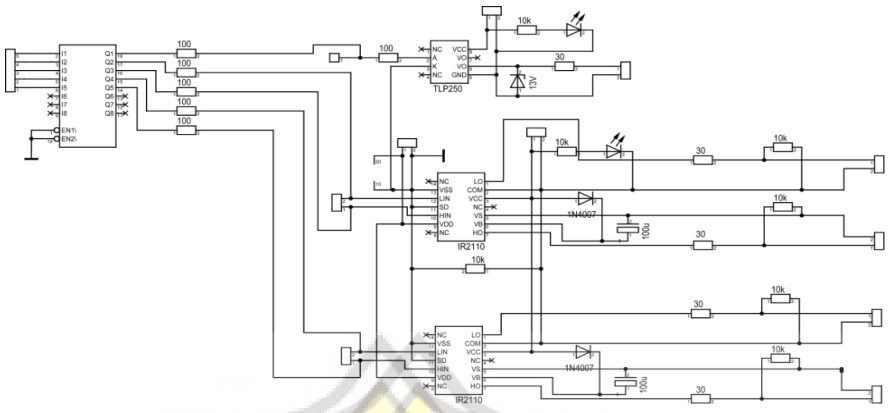
## 3.1 Pendahuluan

Pada Bab ini menjelaskan beberapa Perancangan dan Implementasi *Inverter* 5-Tingkat satu fasa dan beberapa rangkaian pendukung. Beberapa rangkaian pendukung dalam Implementasi yaitu rangkaian *Optocoupler Driver*, Rangkaian Sensor arus LEM HX-10 P, Rangkaian catu daya, dan rangkaian daya *inverter* 5-Tingkat.

## 3.2 Rangkaian *Optocoupler Driver*

*Optocoupler Driver* digunakan sebagai *drive gate* pada MOSFET karena memerlukan tegangan minimal 10 VDC. Selain untuk *drive* berfungsi untuk mengisolasi antara tegangan sehingga tegangan antara mikrokontroler dan rangkaian daya terpisah. Sinyal yang dikeluarkan oleh mikrokontroler dihubungkan ke rangkaian buffer, setelah itu dihubungkan ke rangkaian *optocoupler* dan keluaran pada rangkaian *optocoupler* dihubungkan ke kaki gate pada MOSFET.

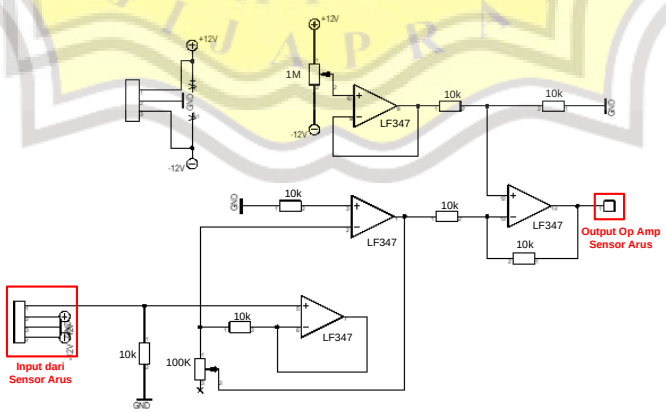
Pada pin HIN dan LIN dihubungkan dengan sinyal keluaran dari rangkaian buffer, dan pada bagian HO dan LO di hubungkan pada kaki gate MOSFET seperti Gambar 3.1.




Gambar 3.1 Rangkaian *Optocoupler Driver*

## 3.3 Rangkaian Sensor Arus LEM HX-10P

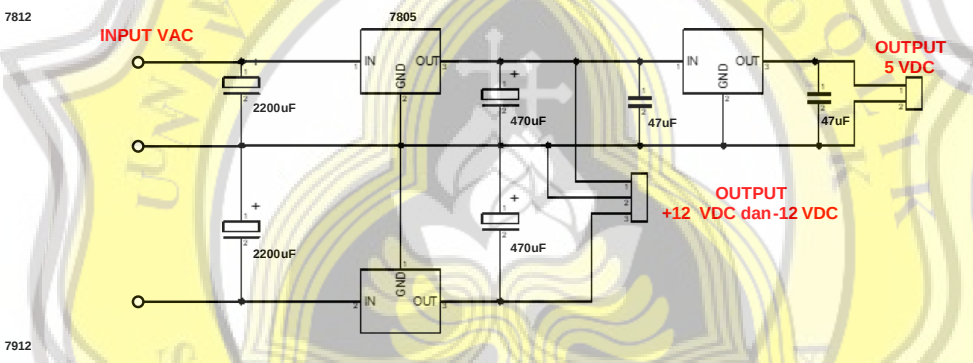
Sensor arus digunakan untuk mendeteksi arus yang digunakan untuk pembanding antara sinyal referensi dan sinyal aktual. Sensor Arus LEM berjenis sensor arus CT yang mampu mendeteksi arus maksimal 10A, sensor arus ini akan menghasilkan tegangan keluaran ketika dialiri oleh arus. Agar tegangan yang dihasilkan sensor arus dapat digunakan perlu adanya rangkaian op-amp seperti Gambar 3.2. Dengan rangkaian op-amp tersebut keluaran dari sensor arus dapat diperbesar dan diberi tegangan bias sesuai dengan keinginan. Sensor arus LEM HX10 P memerlukan tegangan operasi +12VDC -12VDC dan Ground.




Gambar 3.2 Rangkaian Op-Amp Sensor Arus LEM HX-10P

## 3.4 Rangkaian Catu Daya

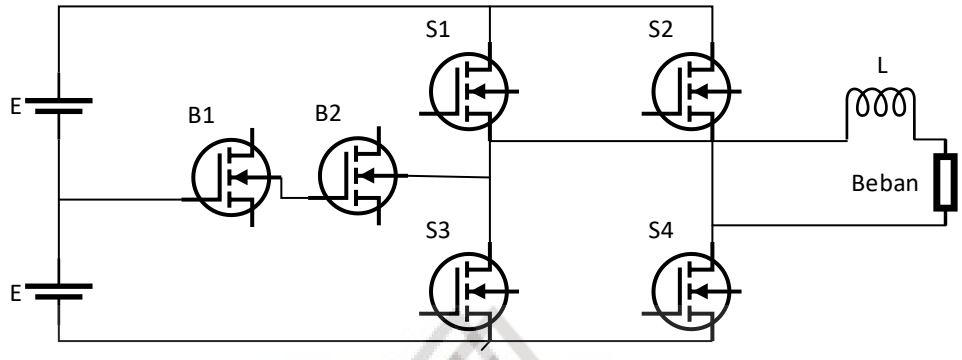
Rangkaian catu daya sebagai penyuplai daya untuk komponen elektronika memerlukan tegangan yang realtif konstan. Gambar 3.3 (a) menunjukan rangkaian catu daya dengan tiga output yaitu, +12VDC, -12VDC dan +5VDC. Sumber +5VDC digunakan sebagai penyuplai mikrokontroler, dan yang lainnya digunakan untuk mensuplai sensor arus LEX HX-10 P dan Gambar 3.3 (b) diagram blok catu daya *optocoupler*.




Gambar 3.3 Rangkaian Catu Daya Sensor dan Mikrokontroler

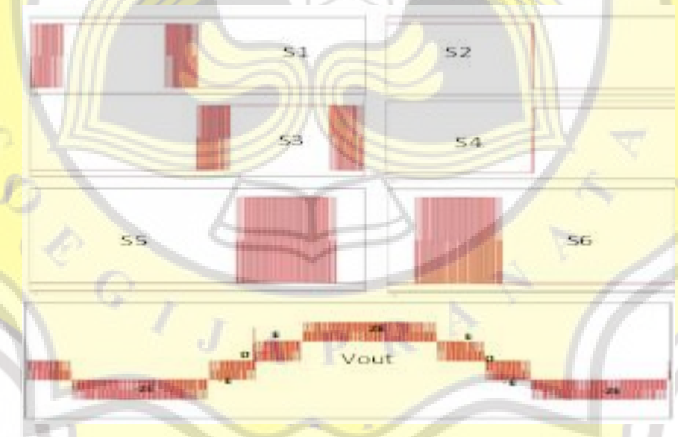
## 3.5 Rangkaian Daya *Inverter* 5-Tingkat

Rangkaian daya pada *inverter* 5-tingkat menggunakan MOSFET IRFP260N sebagai saklar daya dan dioda *fast recovery*, dengan rating tegangan maksimum 200V dan arus maksimal 50A. Rangkaian daya *inverter* 5-tingkat ditunjukan pada Gambar 3.4. *Inverter* 5-tingkat menggunakan dua sumber DC yang dipasang secara seri. Tegangan keluaran tiap *inverter* dihubungkan ke lilitan sekunder trafo *step-up* dan pada sisi lilitan primer dihubungkan pada beban yang berupa lampu 100 Watt.



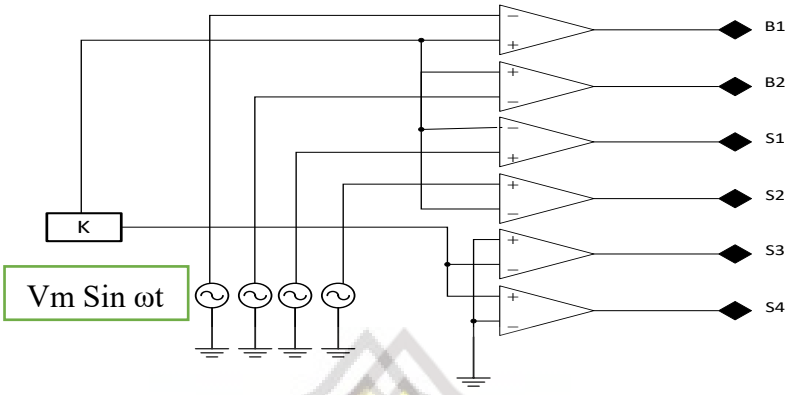

Gambar 3.4 Rangkaian *Inverter* 5-Tingkat pada simulasi PSIM

Pensaklaran setiap saklar dilakukan dengan menggunakan sinyal SPWM seperti Gambar 3.5 sehingga dapat membentuk tegangan keluaran bertingkat. Pensaklaran tiap *inverter* menggunakan metode yang sama dengan sinyal referensi yang saling bergeser 120º. Pembentuk fasa ditunjukan pada Gambar 3.5




Gambar 3.5 Sinyal SPWM S1 – S6

Berdasarkan Gambar 3.5 maka dapat dibentuk rangkaian komparator untuk memodulasi sinyal referensi menjadi sinyal SPWM. Untuk membentuk level kedua pada *inverter* 5-tingkat dilakukan dengan membuat Rangkaian kendali Analog Comparator seperti Gambar 3.6.




Gambar 3.6 Rangkaian kendali Analog Comparator Pada Simulasi

Dari Gambar 3.6 Rangkaian kendali Analog Comparator Pada Simulasi Diatas B1 – S4 dapat disimpulkan dan dibuat konfigurasi pensaklaran seperti Tabel 3.1.

Tabel 3.1 Konfigurasi Pensaklaran *Inverter* 5-Tingkat

| **B1** | **B2** | **S1** | **S2** | **S3** | **S4** | **Vo** |
|---|---|---|---|---|---|---|
| ON | ON | OFF | OFF | OFF | ON | C |
| OFF | OFF | ON | OFF | OFF | ON | 2C |
| OFF | OFF | ON | ON | OFF | OFF | 0 |
| OFF | OFF | OFF | OFF | ON | ON | 0 |
| ON | ON | OFF | ON | OFF | OFF | -C |
| OFF | OFF | OFF | ON | ON | OFF | -2C |

## 3.6 Algoritma Pemrograman

Algoritma pemrograman menggunakan mikrokontroler STM32F407 menggunakan 18 pin *output* digital, 3 pin *input* ADC dan menggunakan 3 *timer counter*. Pada awal program menginisialisasi pin mikrokontroler yang digunakan. Setelah itu menginisialisasi *timer counter*. Pengambilan data dari ADC pada mikrokontroler dilakukan berdasarkan *interrupt timer*. Data ADC digunakan sebagai referensi yang kemudian dimodulasi dengan *timer counter* (sinyal segitiga) dan hasil modulasi tersebut dihubungkan ke dalam gerbang logika. Untuk membentuk polaritas pada *inverter*, dilakukan dengan metode *zero crossing* yang digunakan untuk pensaklaran saklar B1, B2, S1, S2, S3, dan S4. Dan program dalam bahasa C dapat kita konfigurasikan dibawah ini serta Alur pemrograman ditunjukan pada Gambar 3.7.

```
#include <STM32F4ADC.h>
STM32ADC inADC(ADC1);
// Pensaklaran
#define B1 PD0
#define B2 PD1
#define S1 PD5
#define S3 PD4
#define S2 PD7
#define S4 PD6
```

*Header identifier*

```
const int setOverflow = 4000;

const int DT = 50;

int count,data,data2;

uint8_t analog_pins[] = {PA0, PA1, PA2};

int vsin,act,vact,arus,iact,err,err_v,iref;

double itg, lastitg, pi, mod, Bmod, P, I;

double vref, itg_v, lastitg_v, pi_v, P_v, I_v;

int car1,car2,car3,car4;                    Pembacaan ADC dan time sampling


//control

float kp_v = 1 ;

float ki_v = 0.1;

float kp =  0.1;

float ki = 1;                               Deklarasi Sistem Kendali


void PINMODE() {

 pinMode(B1, OUTPUT);

 pinMode(B2, OUTPUT);

 pinMode(S1, OUTPUT);

 pinMode(S3, OUTPUT);

 pinMode(S2, OUTPUT);

 pinMode(S4, OUTPUT);

}                                           Inisialisasi Mode Pin Pada Sakelar

void serial()

{
```

```
 Serial.print(vref);
 Serial.print(" ");
 //Serial.print(car1);
 //Serial.print(" ");
 //Serial.print(car2);
 //Serial.print(" "););
 //Serial.print(car3);
 //Serial.print(" ");
 Serial.print(iact);
 Serial.print(" ");
 Serial.println(vact);
}
```

*Serial Print*

```
void setup() {
 Serial.begin(9600);

 PINMODE();
 for (uint8_t x = 0; x<sizeof(analog_pins); x++)
 pinMode(analog_pins[x], INPUT_ANALOG);
```

*Pin Mode*

```
 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();


Timer3.init();
Timer3.pause(); // stop timer
Timer3.setMasterMode(TIMER_MASTER_MODE_UPDATE);
Timer3.setPrescaleFactor(3); //5.2Khz
Timer3.setOverflow(setOverflow);
Timer3.setCount(0);
Timer3.setMode(TIMER_CH1, TIMER_PWM);
Timer3.refresh();
Timer2.resume();
Timer3.resume();
inADC.setSamplingTime(ADC_SMPR_3);
inADC.enableDMA();
        }
void loop()
{
if (Timer3.getCount() >= 2000)
count = setOverflow-Timer3.getCount();
else
count = Timer3.getCount();
car1 = map(count, 0, 2000, 0, 2000);
car2 = map(count, 0, 2000, 2000, 4000);
car3 = map(count, 0, 2000, -2000, 0);
```

Sub Program Timer Counter

```
car4 = map(count, 0, 2000, -2000, -4000);
```

*Membentuk carrier segitiga & carrier bertingkat*

```
R();
serial();
}
void INT1(void){
vsin = map(analogRead(PA2), 0, 4095, -4000, 4000);
vref = map(vsin, -4000, 4000, 4000, -4000);

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

arus = map(analogRead(PA1), 0, 4095, -4000, 4000);
iact = map(arus, -4000, 4000, 4000, -4000);
}
```

*Nterrupt & Membaca ADC*

```
void R(){

err_v = vref - vact;
P_v = kp_v * err_v;
itg_v = lastitg_v + err_v * 0.00001;
I_v = ki_v * itg_v;
pi_v = P_v + I_v;
```

*Kendali Pi Tegangan*

```
if (pi_v > -4000 && pi_v < 4000) // current anti windup
```

```
 {
   lastitg_v = itg_v;
 }


 iref = pi_v;
 err = iref - iact;
 P = kp * err;
 itg=lastitg+err*0.00001;
 I =ki * itg;
 pi = P + I;
```

*Kendali Pi Arus*

```
//Bidirect 1
 if (pi >= car1){
  digitalWrite(B1, 1);
 }
 else {
  digitalWrite(B1, 0);
 }
//Saklar 1
 if (pi >= car2){
  digitalWrite(S1, 1);
 }
 else{
  digitalWrite(S1, 0);
 }
```

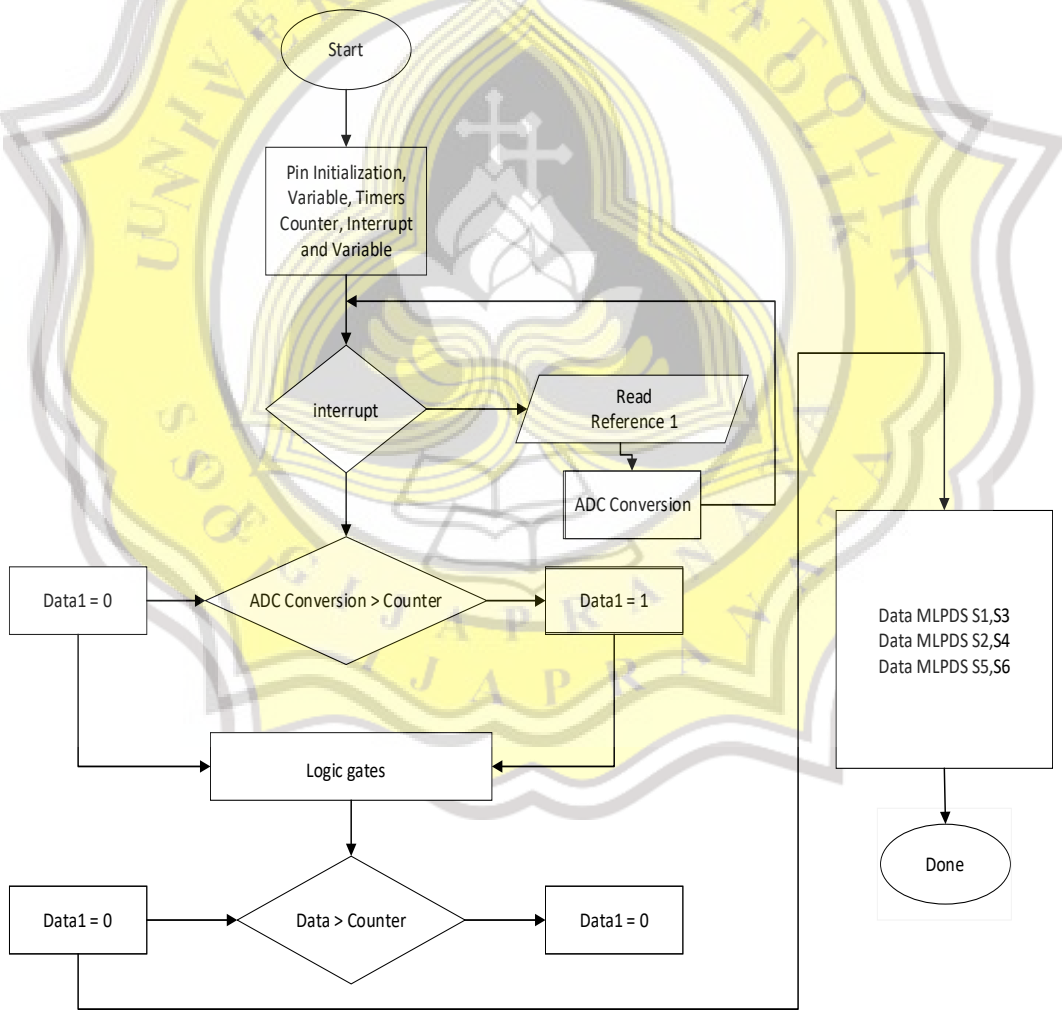
*Pensaklaran B1*

```
//Bidirect 2
 if (pi >= car3){
  digitalWrite(B2, 0);
 }
 else{
  digitalWrite(B2, 1);
 }
//Saklar 2
 if (pi >= car4){
  digitalWrite(S2, 0);
 }
 else{
  digitalWrite(S2, 1);
 }

////ZC////
 if (pi >= DT)     //zero crossing
 {
 digitalWrite(S3, 0);
 digitalWrite(S4, 1);
 }
 else
 digitalWrite(S4, 0);


 if (pi <= DT)     //zero crossing
 {
```

*Pensaklaran B2*

```
digitalWrite(S3, 1);

digitalWrite(S4, 0);

}

else

digitalWrite(S3, 0);

}
```

*Zero Crossing*




Gambar 3.7 Alur Pemrograman