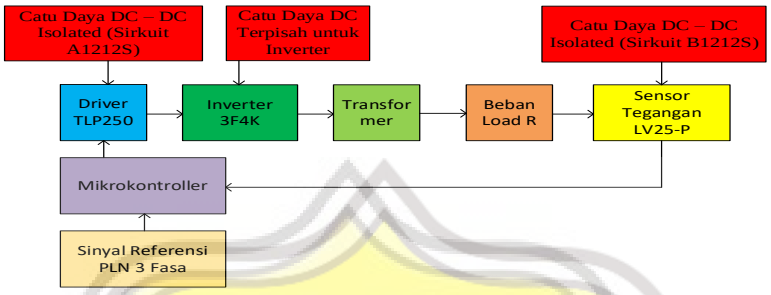
BAB III
DESAIN DAN IMPLEMENTASI

3.1 Pendahuluan

Strategi kontrol yang disajikan dibuktikan melalui simulasi secara komputasional lalu diimplementasikan pada sebuah hardware untuk membuktikan keabsahan dari strategi kontrol tersebut beserta dengan topologi yang disajikan. Desain dari topologi yang disajikan disimulasikan secara waktu nyata menggunakan bantuan dari aplikasi PSIM dari Power SimTech dimana nilai parameter yang digunakan pada simulasi tersebut disamakan dengan nilai parameter pada implementasi di hardware. Nilai parameter tersebut disajikan pada tabel 3.1. Pada gambar 3.1 digunakan pada implementasi hardware serta simulasi dimana melalui gambar tersebut menjadi sebuah acuan dan menjelaskan hubungan antara tiap komponen pada inverter 3F4K. Pada Gambar 3.1, disajikan diagram blok bagi sistem kerja pada simulasi dan Hardware Alat. Gambar tersebut menjadi acuan untuk pembuatan simulasi dan implementasi.

Pada diagram blok dapat dilihat bahwa, rangkaian mencakup supply catu daya, driver, serta beban yang digunakan. Catu daya yang digunakan sendiri terdapat bermacam – macam disesuaikan dengan kebutuhan dan kemudahan dalam pembuatan hardware. Pada diagram blok terdapat

beberapa rangkaian yang mana dibahas pada bab ini sehingga dapat menjadi satu kesatuan rangkaian inverter 3F4K secara utuh.



Gambar 3. 1 Diagram Blok Inverter Tiga Fasa Empat Kawat

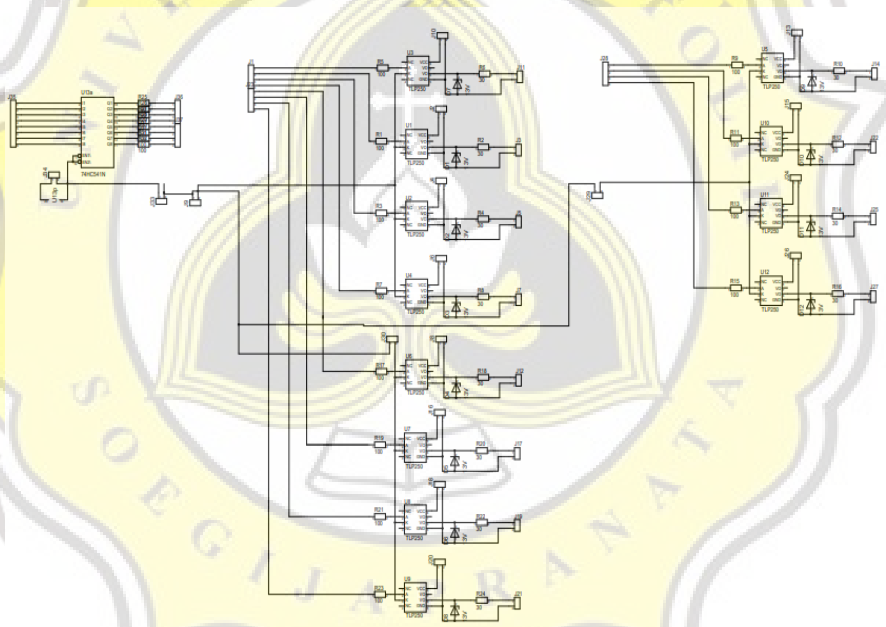
Tabel 3. 1

Parameter Simulasi dan Hardware

Alat	***Nilai***
DC input E	24 V
Filter Induktansi	2 mH
Beban Load Resistor	8 Ω
Hysteresis Batas Atas	0.01 V
Hysteresis Batas Bawah	– 0.01 V
Frekuensi Clock	25 kHz
Frekuensi Interrupt 1	25 kHz
Frekuensi Interrupt 2	25 kHz
Puncak Amplituda Referensi AC	2 V
Frekuensi Referensi AC	50 Hz
Print Time	0.04

3.2 Rangkaian Driver TLP250

Rangkaian driver TLP250 merupakan komponen penting dalam implementasi hardware dikarenakan pada rangkaian berfungsi sebagai sebuah driver atau penggerak yang mana menghubungkan antara mikontroller dengan saklar daya. Rangkaian driver TLP250 yang disajikan pada gambar merupakan desain khusus dimana menyesuaikan dengan kebutuhan untuk mengontrol saklar daya yang digunakan pada inverter 3F4K.

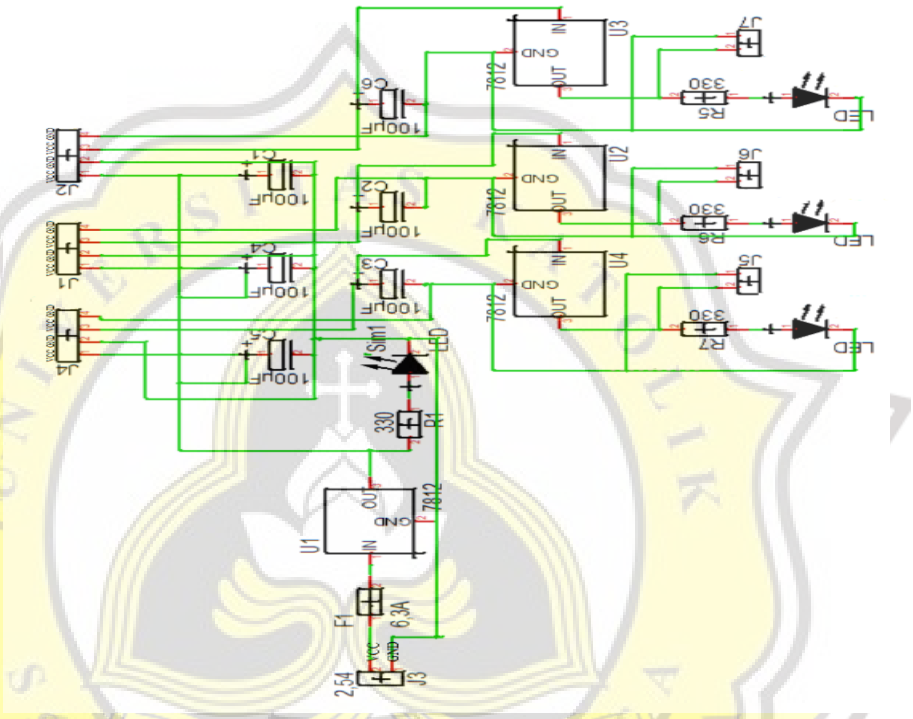


Gambar 3. 2 Rangkaian Driver TLP250

3.3 Rangkaian Sirkuit B1212S

Rangkaian sirkuit B1212S pada implementasi hardware digunakan sebagai sumber untuk menyuplai energi listrik bernilai 12 Volt ke rangkaian

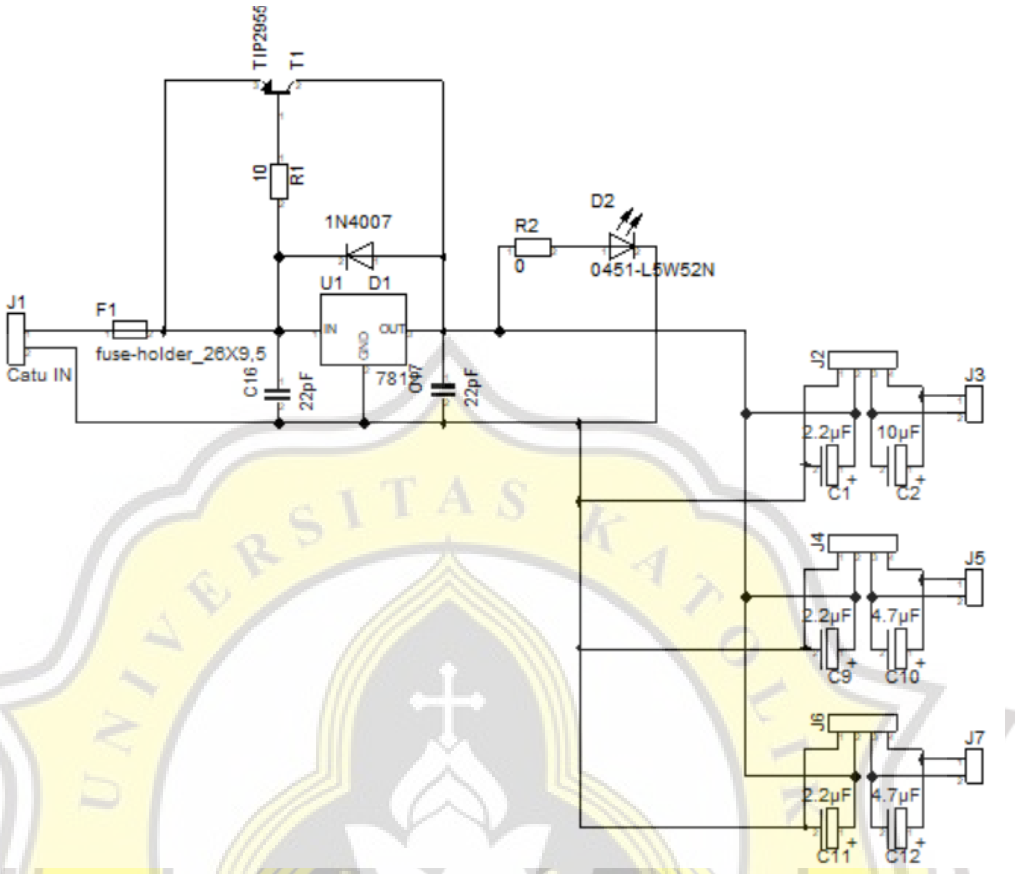
driver TLP250, melalui rangkaian ini dapat dihasilkan tegangan positif dan negatif yang mana dibutuhkan oleh TLP250 untuk dapat bekerja tanpa adanya rangkaian sirkuit dapat menimbulkan beberapa kesulitan yang pada cara untuk menyuplai daya pada TLP250.



Gambar 3. 3 Rangkaian Sirkuit B1212S

3.4 Rangkaian Sirkuit A1212S

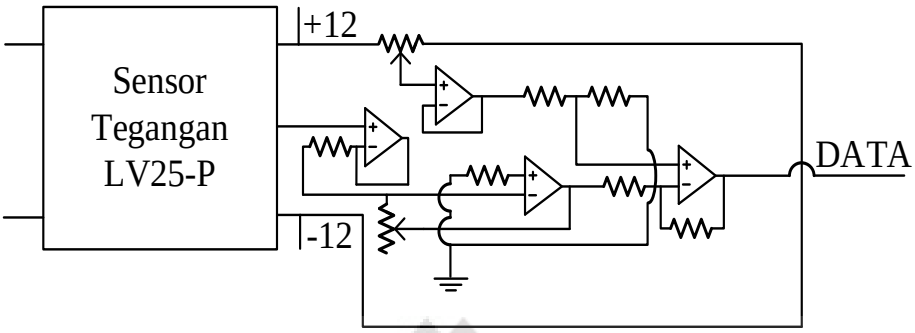
Rangkaian sirkuit A1212S memiliki peran yang hampir sama seperti Rangkaian sirkuit B1212S akan tetapi rangkaian ini juga menghasilkan tegangan nol selain negatif dan positif sehingga rangkaian ini cocok digunakan untuk menyuplai rangkaian sensor tegangan maupun arus dimana biasanya dibutuhkan tiga macam tegangan yaitu positif, negatif, dan nol.



Gambar 3. 4 Rangkaian Sirkuit A1212S

3.5 Rangkaian Sensor Tegangan LV25-P

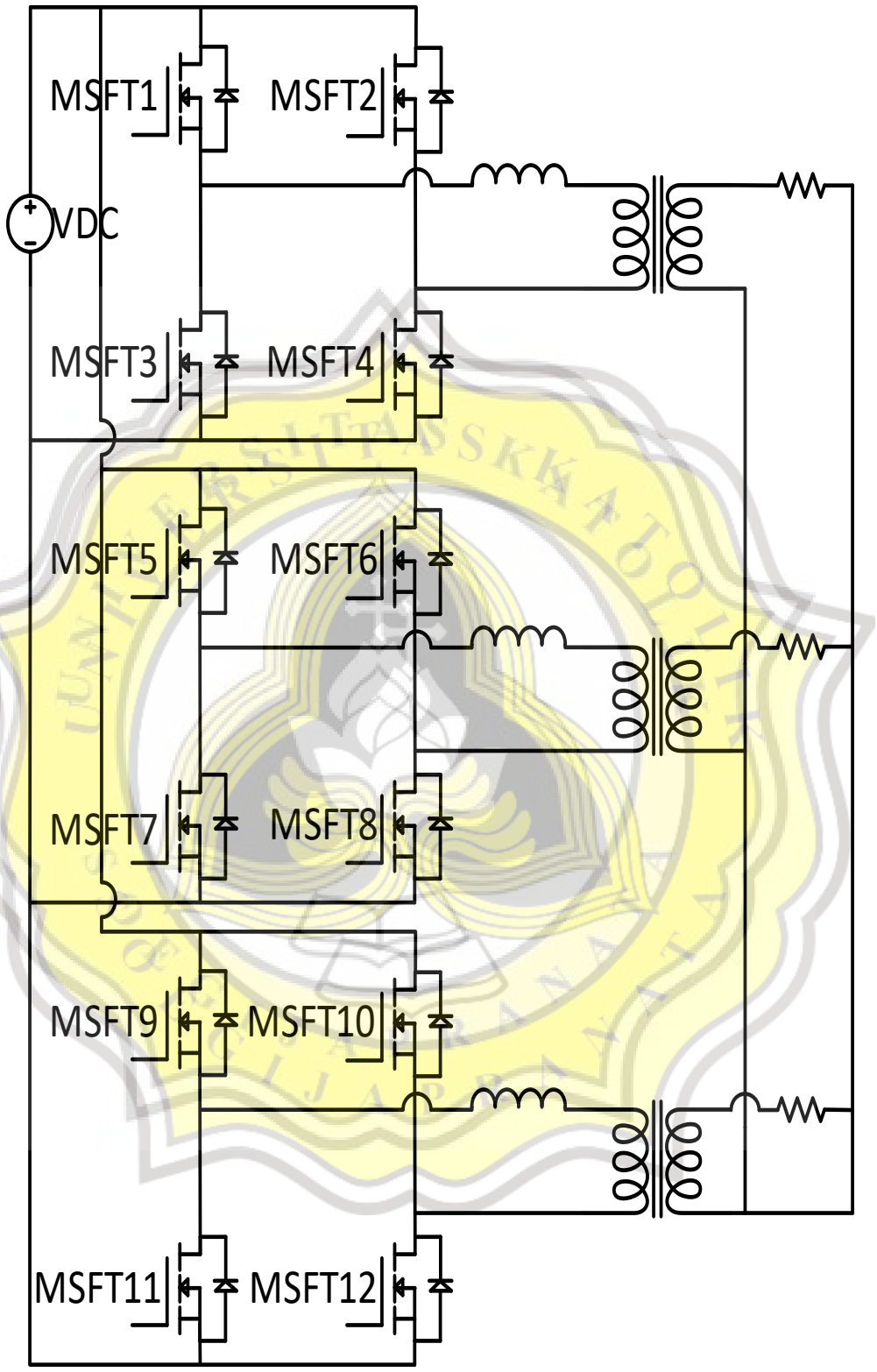
Rangkaian sensor tegangan LV25-P digunakan untuk membaca nilai tegangan keluaran aktual yang dihasilkan oleh inverter 3F4K. Dengan adanya sensor ini dimungkinkan untuk dapat melakukan pembacaan nilai tegangan dengan lebih mudah. Prinsip kerja dari sensor ini adalah dengan membaca nilai tegangan yang mana mencapai nilai maksimum untuk pembacaan berkisar 500 Volt lalu diturunkan menggunakan transformator sehingga menjadi nilai tegangan yang dapat dibaca secara aman oleh mikrokontroller. Sensor ini membutuhkan tiga macam tegangan untuk dapat bekerja yaitu positif, negatif, dan nol.



Gambar 3. 5Rangkaian Sensor Tegangan LV25-P

3.6 Rangkaian Topologi Inverter Tiga-Fasa Empat-Kawat

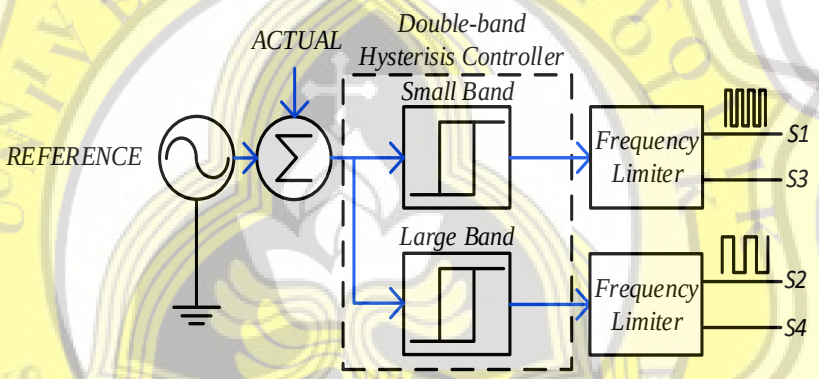
Rangkaian topologi dari inverter 3F4K yang disajikan terdiri atas tiga buah rangkaian topologi inverter satu fasa full bridge dimana diinterkoneksikan secara paralel. Pada setiap inverter full bridge satu fasa memproduksi gelombang AC satu fasa dimana setiap fasanya memiliki perbedaan 120 derajat terhadap masing – masing. Pada bagian sisi keluaran dari setiap inverter dikoneksikan dengan bagian primer dari transformator. Fasa dan titik netral dikoneksikan membentuk hubungan star pada bagian sekunder dari transformer. Dengan topologi yang disajikan, sistem 3F4K dapat didapatkan. Rangkaian topologi dapat dilihat pada gambar 3.6.



Gambar 3. 6 Rangkaian Topologi Inverter Tiga Fasa Empat Kawat

3.7 Rancangan Strategi Kontrol

Rancangan strategi kontrol merupakan pengembangan dari strategi kontrol sederhana yaitu hysteresis dimana dikombinasikan dengan sebuah limiter lalu dikembangkan lagi menjadi hysteresis pita ganda dengan pembatas nilai frekuensi. Strategi kontrol yang disajikan dapat dibentuk dalam sebuah skema pada gambar 3.7. Dengan menggunakan bantuan pembatas frekuensi pada gambar. Strategi kontrol yang disajikan bisa didapatkan.



Gambar 3. 7 Skema Strategi Kontrol yang Disajikan

Pada kondisi positif atau negative, arus yang mengalir pada waktu mati adalah hasil dari perbedaan nilai pada arus dari kondisi awal dengan arus total. Kondisi ini diekspresikan pada persamaan (19) sampai (23). Sehingga, tegangan keluaran dan arus bisa diekspresikan pada persamaan (23) dan (24). Koneksi antara tegangan keluaran dengan arus dideskripsikan pada persamaan (25).

$$\left(V_{DC} - V_{(out)}\right) . T_{(on)} = V_{(out)} . T_{(off)} \quad (19)$$

$$V_{DC} . T_{(on)} - V_{(out)} . T_{(on)} = V_{(out)} . T_{(off)} \quad (20)$$

$$V_{DC} \cdot T_{(on)} = V_{(out)} \cdot T_{(on)} + V_{(out)} \cdot T_{(off)} = V_{(out)} \cdot (T_{ON} + T_{OFF}) \quad (21)$$

$$V_{DC} \cdot T_{(on)} = V_{(out)} \cdot T_s \quad (22)$$

$$V_{(out)} = V_{DC} \frac{T_{(on)}}{T_s} = D \cdot V_{DC} \quad (23)$$

$$I_{(out)} = I_m \sin \theta \quad (24)$$

$$\frac{I_{(m)} \cdot Z}{V_{DC}} = \frac{V_{(out)}}{V_{DC}} \quad (25)$$

Pita kecil hysteresis mengontrol dan menentukan nilai kesalahan pada tegangan keluaran secara keseluruhan. Ketika pita kecil tidak dapat mengatasi nilai kesalahan yang terlalu besar, pita besar akan bereaksi. Pita kecil membentuk tegangan keluaran, sedangkan pita besar mengganti polaritas pada tegangan keluaran. Pensaklaran PWM unipolar dihasilkan pada tegangan keluaran. Pada waktu nyata, frekuensi pensaklaran yang dihasilkan oleh strategi ini tidak dapat diatur, dan bisa melebihi nilai maksimum yang dapat dikontrol oleh komponen. Hal ini menyebabkan adanya ketidakseimbangan frekuensi pada sistem sehingga menghasilkan rugi daya yang besar. Oleh karena itu, frekuensi pensaklaran harus delimitasi mencapai batas maksimum pada komponen untuk menciptakan sistem yang seimbang.

Pada gambar 3.8 menunjukkan adanya dua buah siklus pada inverter. Gelombang diatas level nol tegangan menciptakan nilai tegangan keluaran positif dengan memproduksi sinyal PWM. Pembatas frekuensi diimplementasikan pada salah satu saklar daya untuk melimitasi frekuensi yang dihasilkan. Ketika gelombang berada dibawah tingkat nol tegangan,

tegangan keluaran negatif dihasilkan. Hal tersebut memiliki cara kerja yang sama: pembatas frekuensi melimitasi frekuensi pada salah satu atau kedua saklar daya tersebut. Nilai periode pensaklaran dapat dikalkulasikan melalui persamaan (26). Sehingga, frekuensi pensaklaran diekspresikan pada persamaan (27) dan (28).

$$T_s = T_{ON} + T_{OFF} = \frac{V_{DC} L \Delta i_{(L)}}{V_{(out)}\left(V_{DC} - V_{(out)}\right)} \tag{26}$$

$$f_s = \frac{L\Delta i_{(L)}}{8V_{(o)}C} = \frac{\left(V_{DC} - V_{(out)}\right)D}{L\Delta i_{(L)}} \tag{27}$$

Berdasarkan pada persamaan (28), frekuensi pensaklaran yang dihasilkan harus sama dengan persamaan (30). Sensitivitas frekuensi pensaklaran dikalkulasikan dengan mengganti fungsi menjadi bentuk derivative seperti pada persamaan (29), dan harus memiliki nilai sama dengan nol seperti pada persamaan (30) untuk mendapatkan nilai frekuensi pensaklaran yang maksimum.

Berdasarkan persamaan (30), sudut derajat maksimum harus sesuai dengan persamaan (31), dan nilai dari D membutuhkan nilai seperti pada persamaan (32) untuk mendapatkan nilai nol. Dengan menggunakan persamaan (31) dan (32) ke persamaan (28), frekuensi pensaklaran bisa diekspresikan seperti pada persamaan (33). Nilai dari frekuensi pensaklaran ditentukan melalui sistem modulasi. Jika sistem modulasi dari strategi kontrol hanya menggunakan satu buah gelombang sinusoidal, maka frekuensi pensaklaran yang dihasilkan sama dengan persamaan (33). Jika sistem modulasi dari strategi kontrol menggunakan dua buah sinyal

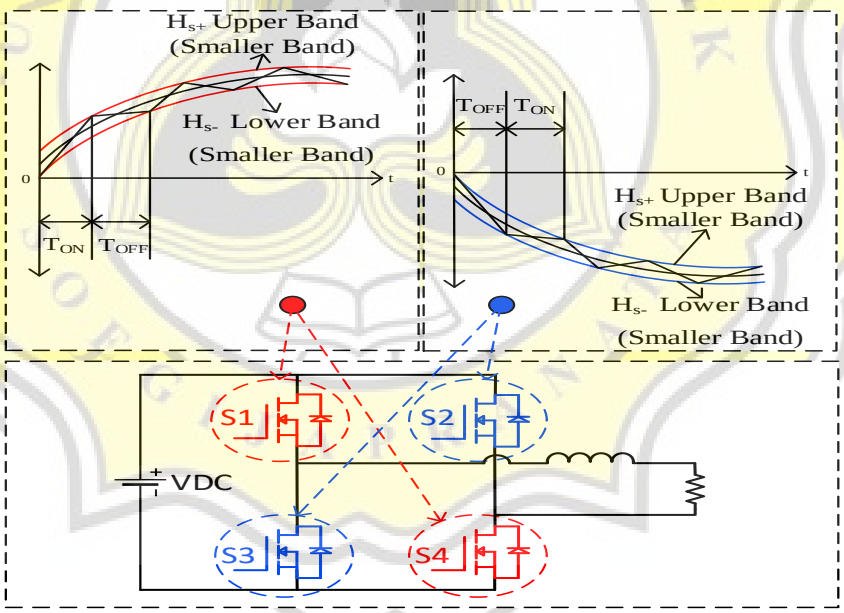
sinusoidal dimana salah satunya bernilai terbalik, hal ini menyebabkan frekuensi pensaklaran memiliki nilai dua kali dari persamaan (33) dan bisa dikalkulasikan sesuai dengan persamaan (34).

$$\sin\theta = \frac{1}{2D} \tag{31}$$

$$D = \frac{1}{2} \tag{32}$$

$$f_s = \frac{1}{4L\Delta i_{(L)}} \tag{33}$$

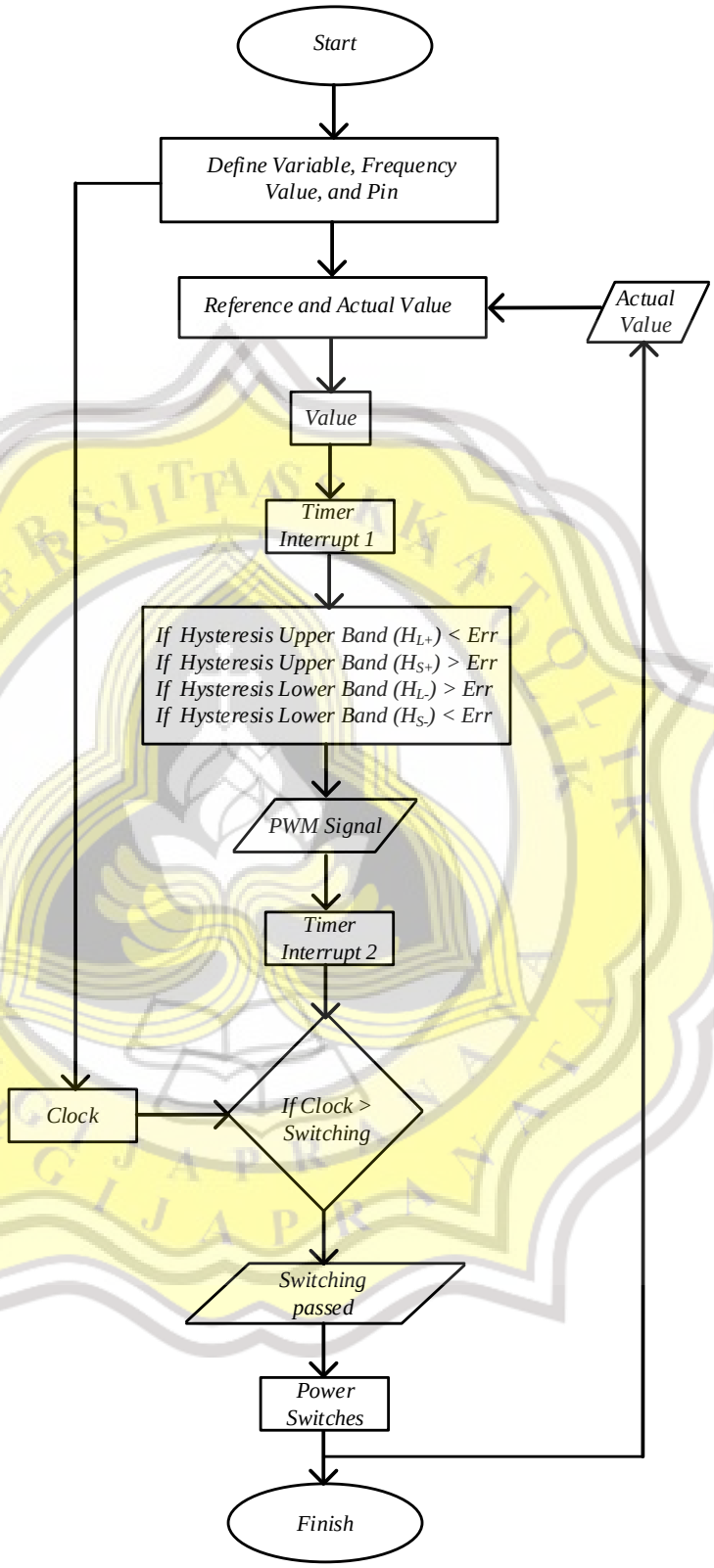
$$f_{maximum} = \frac{1}{2.4L\Delta i_{(L)}} = \frac{1}{8L\Delta i_{(L)}} \tag{34}$$



Gambar 3. 8 Double band hysteresis voltage controller.

3.7.1 Flowchart Pemrograman

Pada gambar diperlihatkan flowchart pemrograman yang menjadi landasan pola pikir untuk implementasi strategi kontrol melalui Bahasa pemrograman. Pada bagian awal flowchart dilakukan inisialisasi terhadap komponen – komponen penyusun program lalu pada program utama terdapat proses untuk mendapatkan nilai kesalahan dimana hal ini yang menjadi acuan untuk diproses dengan menggunakan strategi kontrol yang telah disajikan. Hasil pada flowchart pemrograman ini adalah sebuah sinyal Pulse Width Modulation yang digunakan untuk mengatur pensaklaran pada saklar daya yang digunakan.



Gambar 3. 9 Flowchart Pemrograman

3.7.2 Implementasi Pemrograman

Pada bagian ini dijabarkan pemrograman yang mana merupakan implementasi dari strategi kontrol yang disajikan untuk digunakan oleh mikokontroller dalam mengontrol inverter 3F4K.

```
#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

STM32ADC inADC(ADC1);
int R_ref,S_ref,T_ref;
int R_act,S_act,T_act;
int R_err,S_err,T_err;
int SB = 1; //small band
int LB = 250; //large band
uint16_t analog_pins[] = {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 *Header Library*

Inisialisasi *Pin Variable*

Deklarasi Variabel dan Konstanta

Inisialisasi Input Output Port

```
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(15);
 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_MO
 Timer3.setPeriod(15);
 Timer3.setMode(TIMER_CH3, TIMER_OUTPUT_COMPARE);
 Timer3.setCompare(TIMER_CH3, 3);
 Timer3.attachInterrupt(TIMER_CH3,INT2);
 Timer3.refresh();
 Timer2.resume();
 Timer3.resume();

 inADC.setSamplingTime(ADC_SMPR_3);
 inADC.enableDMA();
}

void loop() {
 //Serial.println(R_ref);
 }
void INT1(void){
 R_ref = map(analogRead(PC4),0,4095,-2000,2000);//referensi
 R_act = map(analogRead(PA6),0,4095,-4000,4000);//actual
 R_err = R_ref - R_act;

 S_ref = map(analogRead(PC5),0,4095,-1700,1700);//r
 S_act = map(analogRead(PA5),0,4095,-4000,4000);//actual
 S_err = S_ref - S_act;

 T_ref = map(analogRead(PA3),0,4095,-2000,2000);//referensi
 T_act = map(analogRead(PA4),0,4095,-4000,4000);//actual
 T_err = T_ref - T_act;
}
```

Inisialisasi Timer Counter 2

Menjalankan Fitur ADC dengan DMA

Program Pembacaan ADC

```
void INT2(void){ //jkff
 R();
 S();
 T();
 }
void R(){
 if (R_err > SB){
  digitalWrite(H1,1);
  digitalWrite(L1,0);
 }
 if (R_err < -SB){
  digitalWrite(H1,0);
  digitalWrite(L1,1);
 }

 if (R_err > LB){
  digitalWrite(H2,0);
  digitalWrite(L2,1);
 }
 if (R_err < -LB){
  digitalWrite(H2,1);
  digitalWrite(L2,0);
 }
}
void S(){
// digitalWrite(H3,0);
// digitalWrite(H4,0);
// digitalWrite(L3,0);
// digitalWrite(L4,0);
 if (S_err > SB){
  digitalWrite(H3,1);
  digitalWrite(L3,0);
 }
 if (S_err < -SB){
  digitalWrite(H3,0);
  digitalWrite(L3,1);
 }

 if (S_err > LB){
  digitalWrite(H4,0);
  digitalWrite(L4,1);
 }
 if (S_err < -LB){
  digitalWrite(H4,1);
  digitalWrite(L4,0);
 }
```

Timer Interrupt

Program Fasa R

Program Fasa S

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

 if (T_err > LB){
  digitalWrite(H6,0);
  digitalWrite(L6,1);}
 if (T_err < -LB){
  digitalWrite(H6,1);
  digitalWrite(L6,0);
 }
}
```

Program Fasa T