BAB III

PERANCANGAN PARALEL INVERTER SUMBER TEGANGAN UNTUK MEMPERBESAR DAYA DAN MEMPERBAIKI TEGANGAN KELUARAN

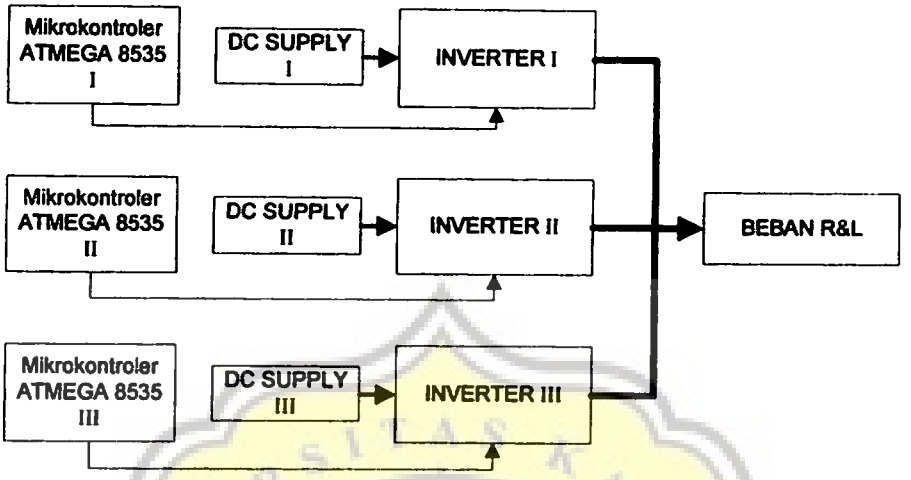
3.1 Pendahuluan

Teknik pengendalian inverter secara digital sudah marak digunakan. Metode yang digunakan adalah menggunakan metode *look up table*. Metode *look up table* adalah suatu metode pengisian data-data ke dalam mikrokontroler dengan mengambil data-data *sampling* yang membentuk suatu gelombang analog. Penggunaan metode *look up table* dengan pengambilan sampling 1 (satu) periode.

Karena dalam pengambilan *sampling* satu periode dirasa masih kurang linear, maka digunakan konsep pemrograman dengan pengambilan data *sampling* ¼ λ (¼ panjang gelombang)[5].

Teknik ini kemudian digunakan pada topologi paralel inverter yang dirancang dengan menghubungkan 3 buah inverter secara paralel. Dengan menggunakan 3 buah mikrokontroler dikembangkan lagi mode inverter yang bisa diubah fasanya untuk digabungkan sehingga dapat memperbaiki kualitas daya[6].

3.2 Perancangan alat

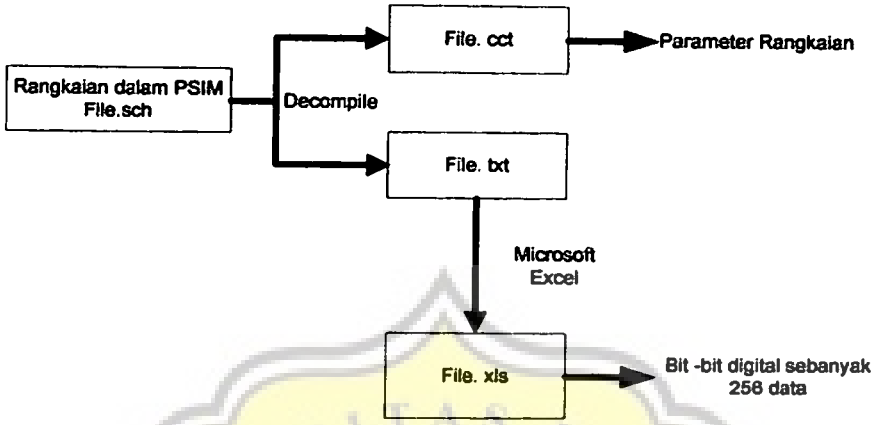


Gambar 3.1. Diagram blok Perancangan Paralel Inverter

Sistem yang digunakan dalam pengendalian inverter satu fasa terprogram ¼ λ adalah dengan menggunakan sistem *Open Loop* (rangkaian terbuka / tanpa umpan balik). Dalam perancangan suatu inverter diperlukan tegangan DC sebagai masukannya sehingga tegangan AC yang disediakan oleh PLN perlu disearahkan terlebih dahulu. Untuk itu diperlukan penyearah sebagai konverter AC ke DC.

Inverter disusun secara terpisah dan dirangkai secara paralel. Dengan program yang kita masukan mode inverter I, inverter II, dan inverter III diatur untuk menghasilkan keluaran tegangan yang lebih baik.

3.3 Metode Pengambilan Data ¼ λ



Gambar 3.2. Diagram pengolahan data

Metode yang digunakan dalam pengambilan sampling ¼ λ adalah dengan simulasi menggunakan *software Power simulator*, dengan mengkomparasikan sinyal sinus 3V, 50 Hertz dengan sinyal segitiga 3V, 5000 Khz diperoleh pola pensaklaran SPWM. Dalam pemakaian *Software* ini perlu pengaturan dalam *time step* supaya dapat menghasilkan data sebanyak 256.

Pada dasarnya, *software* ini akan memuat suatu hasil simulasi dalam bentuk 3 macam file, yaitu:

- .sch

  File .sch merupakan rangkaian yang terdapat pada layar saat *software* dijalankan.

- .cct

  File .cct merupakan parameter yang digunakan dalam simulasi.

- .txt

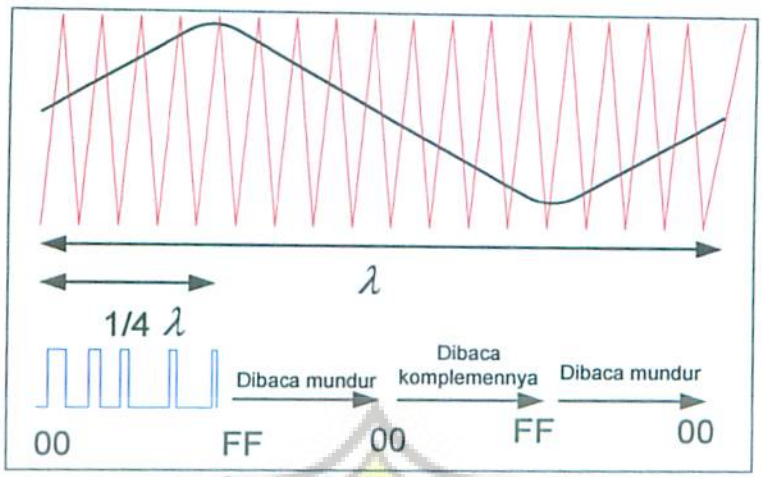
  File .txt merupakan data hasil simulasi yang tertuang dalam desimal.

Selanjutnya, data yang digunakan adalah data dari file .txt. Data yang ada dalam file .txt merupakan data yang berbentuk hexa decimal sebanyak 256 data. Dalam pemrograman mikrokontroler data yang digunakan adalah berupa data digital / biner (0, dan 1). Untuk itu perlu adanya konversi dari hexa decimal ke dalam bentuk biner. Konversi data tersebut dilakukan dalam *microsoft excel.* Setelah data berada dalam *microsoft excel,* data sudah berbentuk biner dan siap dimasukkan dalam program mikrokontroler.

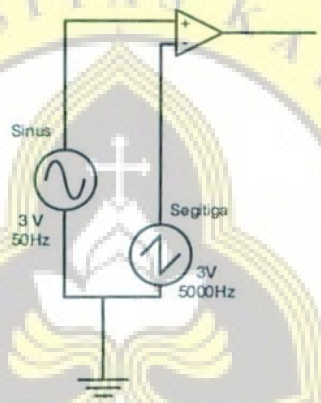
Berikut adalah simulasi dan akuisisi data yang digunakan untuk mendapatkan data SPWM ¼ λ

Teknik pemrograman ¼ λ yang dimaksud adalah sebagai berikut :

- Gelombang sinusoidal yang dipakai adalah ¼ λ
- Sedangkan sinyal pembawa tidak ada perubahan.
- Hasil simulasi merupakan modulasi lebar pulsa sinusoidal ¼ λ dari sinyal yang dimodulasi dimasukkan ke memori pada mikrokontroler
- Teknik yang dipakai untuk menghasilkan satu gelombang adalah pembacaan dari ¼ λ gelombang kemudian dibaca terbalik, kemudian akan dibaca mundur sehingga akan menghasilkan satu gelombang penuh, seperti pada gambar.



Gambar 3.3 Sinyal keluaran SPWM



Gambar 3.4 Simulasi SPWM analog

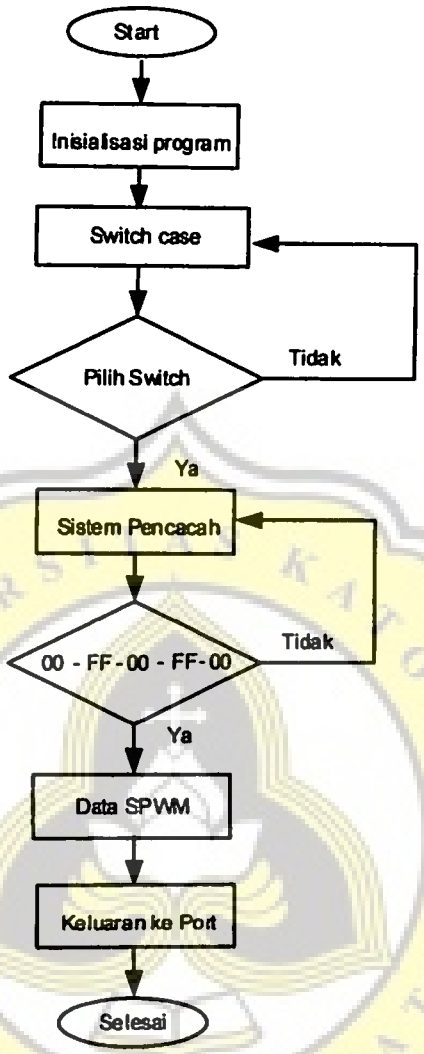
3.4 Pemrograman Mikrokontroler ATMEGA8535

Data hasil simulasi komputasi dengan menggunakan *power simulator* diambil sebanyak 256 data, data tersebut kemudian dikonversi ke *microsoft excel*. Pemrograman mikrokontroler ATMEGA8535 menggunakan bahasa C dengan menggunakan *Code Vision AVR* (CVAR). Metode yang digunakan adalah memasukkan data *sampling* sebanyak 256 data dari hasil simulasi yang telah dilakukan kedalam *look up table.* Untuk menghindari kesalahan dalam penghitungan *delay* atau perbedaan penghitungan waktu dengan mikrokontroler, setiap data dari *look up table* akan dikontrol.

Sinyal SPWM akan dibangkitkan dengan cara memanggil data *look up table* pada memori mikrokontroler. Pulsa *sampling* yang dimasukkan ke dalam program adalah sebanyak 256 pulsa dalam rentang ¼ periode. Inilah yang dimaksud dengan pemrograman ¼ λ. Digunakan ¼ λ dengan tujuan supaya pembebanan komputasi pada mikrokontroler tidak terlalu berat, dengan kata lain, memori yang digunakan menjadi semakin sedikit.

Untuk ¼ periode = 256 pulsa, berarti untuk membentuk satu gelombang penuh, mikrokontroler akan menghasilkan pulsa sebanyak 256 X 4 = 1024 pulsa. Dengan kata lain dengan menggunakan metode ini sinyal keluaran yang dihasilkan mikrokontroler akan menjadi semakin baik. Berikut ini adalah aturan main pemrograman ¼ λ.

Data kemudian diolah dalam program agar menjadi 4 mode *switching* pada PORT A yaitu, normal, fasa tergeser 180^0, 120^0, dan -120^0 dan dapat dipanggil lewat perintah masukan yang sudah dibuat dalam program pada PORT D:



Gambar 3.5 Diagram *Flowchart* Mikrokontroler

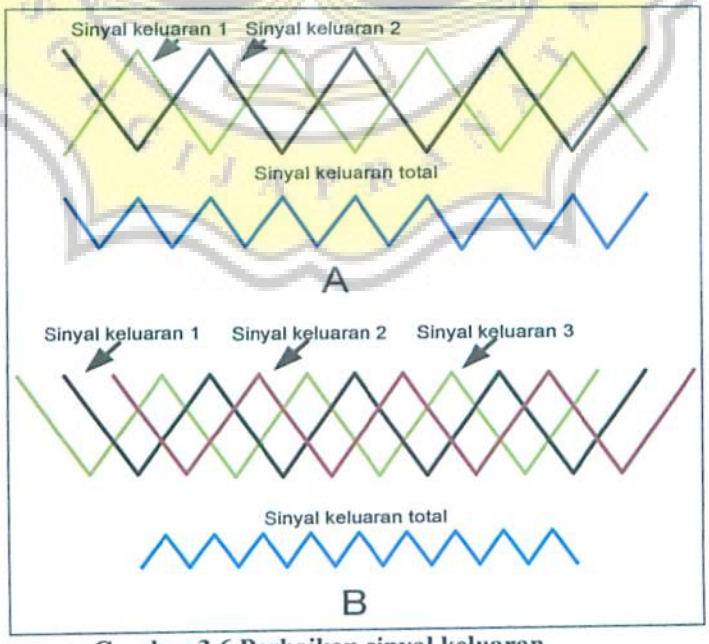
3.5 Kerja Paralel Inverter

Suatu sumber energi listrik dapat dikerjakan paralel seperti generator jika memenuhi kriteria tertentu. Untuk menghasilkan sumber daya dan tegangan yang lebih baik mode pengaturan fasa inverter juga harus diperhatikan. Pengubahan fasa secara sembarangan dapat menyebabkan kerusakan pada alat. Untuk menghindari hal ini dibuat aturan paralel inverter[8].

Tabel 3.1. Aturan paralel inverter

Swicth case	Inverter	Keterangan
A	Inverter A, Inverter B, Inverter C	Kerja Mandiri, sinyal pembawa tidak tergeser
A dan B	Inverter A dan B atau Inverter A dan C atau Inverter B dan C	Paralel dua inverter, sinyal pembawa saling tergeser 180^0
A, B dan C	Inverter A, B dan C	Paralel tiga inverter, sinyal pembawa saling tergeser 120^0

Perbaikan sumber daya dan tegangan dapat dilihat dari perubahan sinyal yang telah dijumlahkan, semakin banyak sinyal yang dijumlahkan semakin baik nilai keluaran totalnya



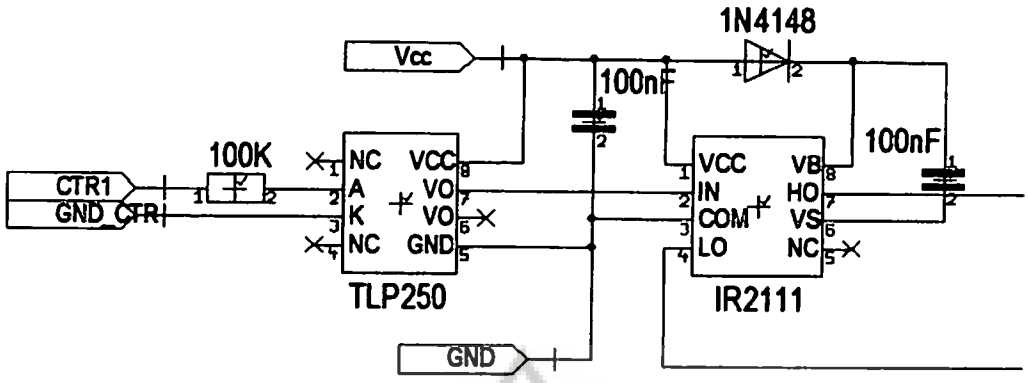
Gambar 3.6 Perbaikan sinyal keluaran

3.6 Rangkaian Driver

Saklar statis yang diimplementasikan dengan MOSFET memiliki tiga buah terminal, yaitu G (*gate*), D (*drain*) dan S (*source*). Pada implementasi MOSFET model NPN jenis *enchantment* arus mengalir dari D ke S jika tegangan G-S melebihi nilai ambangnya (*threshold*), jika tidak maka arus akan ditahan (*blocking*).

Saklar daya sejenis MOSFET atau IGBT bekerja berdasar pulsa pemicuan dari rangkaian kontrol pada *gate*-nya tetapi bekerja pada orde daya yang lebih tinggi sehingga untuk mengendalikan setiap saklar daya diperlukan rangkaian *driver*. Pada aplikasi rangkaian inverter ini isolasi dilakukan dengan menggunakan opto coupler jenis TLP 250 yang terpasang dengan rangkaian *driver*. Rangkaian *driver* berfungsi untuk memindahkan sinyal picu dari sistem kontrol ke sistem daya dengan memisahkan bagian *ground* daya dari *ground* kontrol, karena keduanya bekerja pada catu tegangan yang berbeda. Untuk itu pada setiap rangkaian *driver* harus dicatu dengan catu daya tersendiri yang sesuai dengan tegangan yang dibutuhkan oleh terminal G (*gate*) MOSFET yang digunakan.

Kerusakan saklar statis MOSFET juga sering terjadi karena panas yang ditimbulkan dari gesekan pulsa yang melewatkan arus pada saklar tersebut, untuk itu demi keamanan saklar daya tersebut rangkaian *driver* juga dilengkapi dengan *death time* untuk mengatur perpindahan pulsa pemicuan pada setiap saklar dalam satu lengan.



Gambar 3.7 Rangkaian isolasi dan *driver*

IR2111 berfungsi sebagai *driver* untuk satu lengan inverter dengan kapasitas *switching* yang tinggi. Karena IR2111 memiliki keluaran *high* dan *low* yang memiliki *internal death time* terjadinya saklar hidup secara bersamaan dapat dihindari. Penggunaan IR2111 mengurangi penggunaan *external death time* dan menyederhanakan bentuk rangkaian.