II. TINJAUAN PUSTAKA

A. Pengenalan DC *Chopper*

Chopper adalah suatu alat yang mengubah sumber tegangan arus searah tetap menjadi sumber tegangan arus searah yang bersifat variable. Pengubah daya DC-DC (DC-DC *Converter*) tipe peralihan atau dikenal juga dengan sebutan *DC Chopper* ini dimanfaatkan terutama untuk penyediaan tegangan keluaran DC yang bervariasi besarannya sesuai dengan permintaan pada beban (Adriadi, Y).

Daya masukan dari proses DC-DC tersebut adalah berasal dari sumber daya DC yang biasanya memiliki tegangan masukan yang tetap. Pada dasarnya, penghasilan tegangan keluaran DC yang ingin dicapai adalah dengan cara pengaturan lamanya waktu penghubungan antara sisi keluaran dan sisi masukan pada rangkaian yang sama. Komponen yang digunakan untuk menjalankan fungsi penghubung tersebut tidak lain adalah switch (*solid state electronic switch*) seperti misalnya Thyristor, MOSFET, IGBT, GTO. Secara umum ada dua fungsi pengoperasian dari DC Chopper yaitu penaikan tegangan dimana tegangan keluaran yang dihasilkan lebih tinggi dari tegangan masukan, dan penurunan tegangan dimana tegangan keluaran lebih rendah dari tegangan masukan.

Chopper digunakan pada regulator tegangan dc, dan juga digunakan pada penghubung dengan induktor, untuk membangkitkan sumber arus dc, terutama untuk pembalik sumber arus. Pada banyak aplikasi industri, diperlukan untuk mengubah sumber tegangan DC tetap menjadi sumber tegangan dc yang bersifat variabel. DC *chopper* mengubah secara langsung dari DC ke DC dan biasanya hal ini disebut konverter dc ke dc. *Chopper* dapat disebut sebagai DC, sama dengan *transformator* AC dengan perbandingan putaran yang terus menerus. Seperti *transformator*, *chopper* dapat digunakan untuk menaikkan dan menurunkan sumber tegangan DC (Fauzan, A).

.

B. Klasifikasi *Chopper*

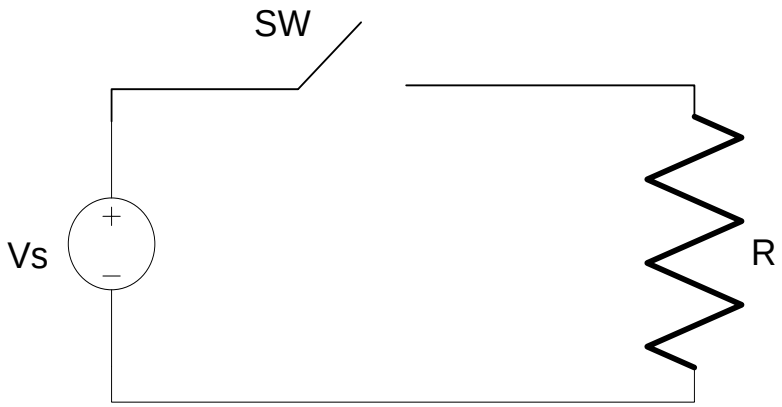
Chopper dapat diklasifikasikan menjadi lima jenis berdasarkan pada arah arus dan tegangan yang mengalir yaitu (Rashid, M):

- *Chopper* kelas A
- *Chopper* kelas B
- *Chopper* kelas C
- *Chopper* kelas D
- *Chopper* kelas E

Dalam tugas akhir ini yang akan dirancang adalah klasifikasi DC *chopper* kelas A, kelas B, dan kelas C.

1. *Chopper* Kelas A

Chopper kelas A ini hanya mengirimkan daya dari sumber ke beban.



Gambar 1. Rangkaian *Chopper* Kelas A

Chopper kelas a ini adalah *chopper* kuadaran kesatu, karena kuadran kerja yang dihasilkannya berada pada kuadran kesatu. Tegangan dan arus beban nya adalah positif.

Tegangan keluaran rata-rata diberikan oleh

$$V_a = \frac{I}{T}\int_0^{t_1} V_o dt = \frac{t_1}{T}V_s = f t_1 V_s = kV_s$$

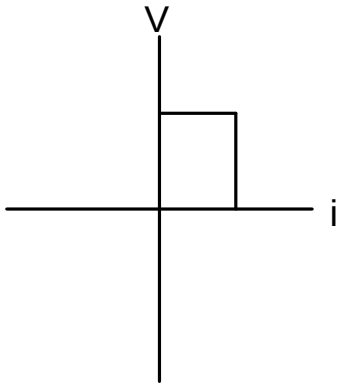
Dan arus beban rata-rata,

$I_a = \frac{V_a}{R} = \frac{kV_s}{R}, dengan\ T\ adalah\ periode\ chopping, k = \frac{t_1}{T}$ adalah duty cycle *chopper,* dan f adalah frekuensi chopping. Nilai rms tegangan keluaran ditentukan dari

$$V_0 = (\frac{1}{T}\int_0^{kT} v^2 0\, dt)^{1/2} = \sqrt{k}\, V_s$$

Dengan mengasumsikan bahwa tidak ada rugi-rugi pada pada *chopper* maka daya masukan pada *chopper* sama dengan daya keluaran yang diberikan dengan

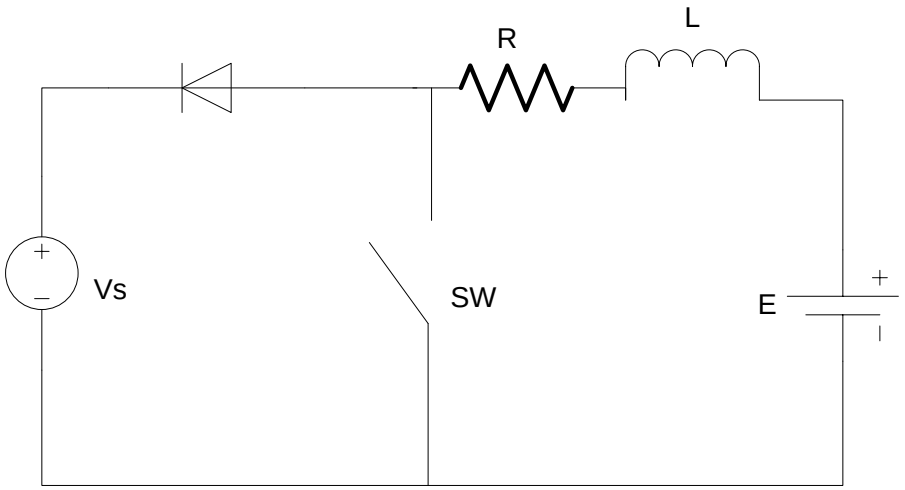
$$P_1 = \frac{1}{T}\int_0^{kT} v_0 i\, dt = \frac{1}{T}\int_0^{kT} \frac{v^2 0}{R} dt = k\frac{v^2 s}{R}$$



Gambar 2. Bentuk Kuadran Kerja *Chopper* Kelas A

2. *Chopper* Kelas B

Chopper kelas B ini arus bebannya keluar dari beban, dengan baterai E adalah bagian dari beban dan dapat menjadi emf balik motor dc.



Gambar 3. Rangkaian *Chopper* Kelas B

Chopper ini dapat juga disebut sebagai *chopper* kuadran kesatu, tetapi bekerja pada kuadran kedua. Tegangan beban positif, tetapi arus beban negatif.

Bila saklar S_w di on-kan, tegangan E menghasilkan arus melalui induktor L dan tegangan beban v_L menjadi nol. Arus i_l, yang meningkat, dinyatakan sebagai berikut:

$$0 = L\frac{di_L}{dt} + Ri_L + E$$

Dengan kondisi mula $i_L(t = 0)I_1$, memberikan

$$i_L = I_1 e^{-\left(\frac{R}{L}\right)t)} \qquad atau \quad 0 \leq t \leq kT$$

Pada t = t_1

$$i_L(t = t_1 = kT) = I_2$$

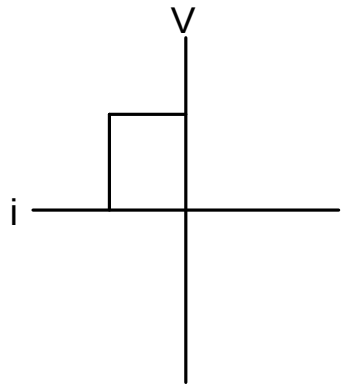
Ketika saklar S_1 di-off-kan, jumlah energi yang disimpan pada induktor L dikembalikan pada sumber V_s melalui diode D_1. Arus beban i_L akan jatuh. Dengan mendefinisikan ulang waktu mula t = 0, arus beban i_L dinyatakan oleh

$$V_s = L\frac{di_L}{dt} + Ri_L + E$$

Dengan kondisi mula $i(t = t_2) = I_2, menghasilkan$

$$i_L = \frac{V_s - E}{R}\left(1 - e^{-\left(\frac{R}{L}\right)t}\right) \quad atau\ 0 \leq t \leq t_2$$

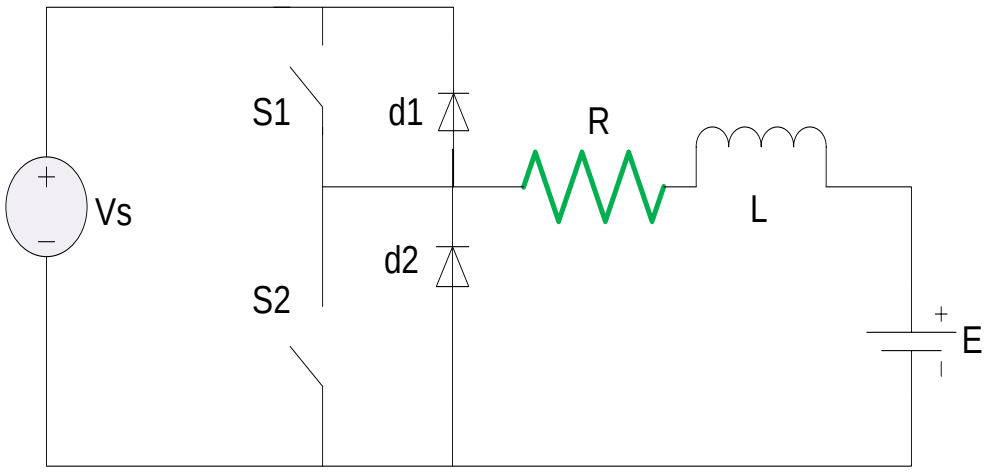
Dengan $t_2 = (1 - k)T. Pada\ t = t_2$



Gambar 4. Bentuk Kuadran Kerja *Chopper* Kelas B

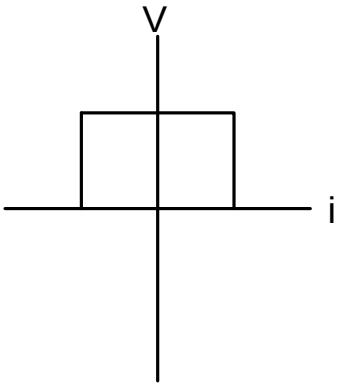
3. *Chopper* Kelas C

Chopper kelas C dapat juga disebut sebagai kombinasi *chopper* kelas A dan *chopper* kelas B. S_1 dan D_2 bekerja seperti *chopper* kelas A, S_2 dan D_1 bekerja seperti *chopper* kelas B. Harus dijaga hati-hati agar dua saklar tersebut tidak bekerja secara bersama-sama;bila hal ini terjadi maka sumber V_s akan mengalami hubung singkat. *Chopper* kelas C ini dapat bekerja sebagai penyearah atau pembalik (*inverter).*



Gambar 5. Rangkaian *Chopper* Kelas C

Arus beban dapat positif atau negatif, sedangkan tegangan beban selalu positif. Ini disebut chopper kuadran kedua.



Gambar 6. Bentuk Kuadran Kerja *Chopper* Kelas C

C. Mikrokontroler

Mikrokontroler adalah prosesor mikro yang yang terdiri CPU ditambah dengan RAM, ROM, I/O *ports*, dan *timer* yang jumlahnya tetap dan dikemas dalam satu chip. *Mikrokontroler* merupakan salah satu bagian dasar dari suatu sistem

komputer. Meskipun mempunyai bentuk elemen-elemen dasar yang sama. Secara sederhana, komputer akan menghasilkan *output* spesifik berdasarkan *input* yang diterima dan program yang dikerjakan. Seperti umumnya komputer, *mikrokontroler* adalah alat yang mengerjakan instruksi-instruksi yang diberikan kepadanya. Artinya, bagian terpenting dan utama dari suatu sistem terkomputerisasi adalah program itu sendiri yang dibuat oleh seorang *programmer*.Program ini menginstruksikan komputer untuk melakukan jalinan yang panjang dari aksi-aksi sederhana untuk melakukan tugas yang lebih kompleks yang diinginkan oleh *programmer*.yang jauh lebih kecil dari suatu komputer pribadi dan komputer *mainframe.*

Mikrokontroler AVR (*Alf and vegard's Risc processor*) merupakan mikrokontroler Atmel yang mempunnyai arsitektur RISC 8-bit,dimana semua instruksi dikemas dalam kode 16-bit dan sebagian besar instruksi dieksekusi dalam 1 siklus *clock,* berbeda dengan instruksi MCS51 yang membutuhkan 12 siklus *clock.* Tentu saja itu terjadi karena kedua jenis mikrokontroler tersebut memiliki arsitektur yang berbeda. AVR berteknologi RISC (*reduced Instruction set computing*), sedangkan seri MCS51 berteknologi CISC (*Complex Instruction Set Computing*). Secara umum, AVR dapat dikelompokkan menjadi 4 kelas, yaitu keluarga ATtiny, keluarga AT90Sxx, keluarga ATmega, dan AT86RFxx. Pada dasarnya yang membedakan masing-masing kelas adalah memori, *peripheral*, dan fungsinya. Dari segi arsitektur dan instruksi yang digunakan, mereka bisa dikatakan hampir sama.

ATMega8535

ATMega8535 memiliki fitur-fitur sebagai berikut:

a. Saluran I/O sebanyak 32 buah yaitu *port* A, *port* B, *port* C, dan *port* D.

b. ADC 10 bit sebanyak 8saluran.

c. Tiga buah *Timer*/*Counter* dengan kemampuan pembanding.

d. CPU yang terdiri atas 32 buah register.

e. *Watchdog Timer* dengan osilator internal.

f. SRAM sebesar 512byte.

g. Memori *Flash* sebesar 8 KB dengan kemampuan *Read While Write*.

h. Unit interupsi internal dan eksternal.

i. *Port* antarmuka SPI.

j. EEPROM sebanyak 512 byte yang dapat diprogram saat operasi.

k. Antarmuka komparator analog.

l. *Port* USART komunikasi serial.

Kapabilitas detail dari ATMega8535 adalah sebagai berikut:

a. Sistem *mikroprosesor* 8 bit berbasis RISC dengan kecepatan maksimal 16 MHz.

b. Kapabilitas memori *flash* 8 Kb, SRAM sebesar 512 byte, dan EEPROM sebesar 512 byte.

c. ADC internal dengan fidelitas 10 bit sebanyak 8 *channel*.

d. *Port* komunikasi serial (USART) dengan kecepatan 2,5 Mbps.

e. Enam pilihan *mode sleep* menghemat penggunaan daya listrik.

Dalam penelitian ini dibutuhkan lebih dari satu buah saluran I/O yang dihubungkan dengan rangkaian konverter dc ke dc. Sehingga *mikrokontroller* yang digunakan harus mempunyai fitur dan port I/O yang memadai. Atas dasar ini maka digunakan *mikrokontroler* ATMega8535 sebagai pengendali utama.

D. Bahasa C

a. Format Penulisan Bahasa C

Untuk mengenal lebih jauh tentang bahasa C, kita terlebih dahulu harus mengenal struktur atau format penulisan program. Sehingga kita memiliki gambaran tentang bagian-bagian dalam pembuatan program (Wardhana, L).

- Bagian Komentar Keterangan Program

  Komentar digunakan untuk memberikan keterangan pada program agar mudah dibaca dan akan diabaikan oleh *compailer*.

  Cara kerja:

  /*……*/ untuk komentar bentuk paragrap

  // untuk komentar bentuk per baris (sebelum enter)

- Bagian *preprocessor*

  Preprocessor #include biasanya digunakan untuk menyertakan file header(.h) atau file library. File include berguna untuk memberitahu *compiler* agar membaca file yang di-include-kan lebih dahulu agar mengenali definisi-definisi yang digunakan dalam program agar tidak dianggap *error*.

Cara penulisan:

#include <……> untuk lokasi standar *file* yang telah di-*setting* oleh tools biasanya pada *folder* include atau *folder* direktori *compiler*.

#include "……" untuk lokasi file yang kita tentukan sendiri

File header io.h adalah *file* yang berisi segala informasi atau definisi tentang register-register fungsi khusus (SFR) dan bit-bit atau pin-pin *mikrokontroller* AVR.

- Bagian deklarasi variable global

  Variable global dideklarasikan di luar semua fungsi termasuk fungsi utama dan letaknya harus di atas. Sifat variabel global yaitu dapat diakses aleh semua pernyataan dalam program.

- Bagian *Prototype* Fungsi

  Berfungsi untuk mendeklarasikan fungsi yang di bawah fungsi "*main*". Jika kita membuat fungsi di atas fungsi "*main*" maka kita tidak usah mendeklarasikan fungsi tersebut langsung tulis saja seperti fungsi "x_pangkat_y".

- Bagian Fungsi Utama/main

  Fungsi utama adalah fungsi pertama yang akan dieksekusi dengan urutan dari atas ke bawah dan akan loncat-loncat bergantung pada instruksi loncatan "goto" atau instruksi panggilan fungsiatau terjadi interupsi apabila interupsi diaktifkan

  Bagian Subprogram fungsi fungsi yang telah di*prototype*kan ditulis di bawah fungsi main. *Prototype* fungsi berguna untuk memudahkan

programmer dalam penulisan program yang besar. Jika kita membuat sebuah fungsi tanpa kita prototypekan maka harus ditulis diatas fungasi maindan ini menyulitkan untuk dibaca dan diperbarui sehingga kita butuh *prototype* fungsi.

b. Komentar

Komentar tidak akan dikompilasi oleh *compiler*. Komentar berguna bagi kita untuk membantu programmer mengingat kembali fungsi tiap pernyataan-pernyataan program.

c. *Preprocessor*

- *Preprocessor* #include

  Biasanya digunkan untuk menyertakan *file header* (.h) atau *file library*.

- *Preprocessor* #define

  Digunakan untuk mendefinisikan konstanta atau makro.

- *Preprocessor* #if-#endif

  Digunakan untuk mengetes ekspresi yang valid untuk mengolah kode program dibawahnya hingga #endif.

- *Preprocessor* #if-#else-#endif

  Digunakan untuk mengetes ekspresi yang valid untuk mengolah kode program di bawahnya atau jika tidak valid maka kode program di bawah #else hingga #endif.

- *Preprocessor* #ifdef-#endif

Digunakan untuk mencari tahu apakah *indetifief* sudah didefinisikan? Jika ya akan mengeksekusi program di bawahnya.

- *Preprocessor* #ifndef-#endif

  Digunakan untuk mencari tahu apakah indetifier sudah didefinisikan? Jika belum maka akan mengeksekusi program dibawahnya.

- *Preprocessor* #undef

  Digunakan untuk menghilangkan *identifier* yang telah kita definisikan dengan #define, sehingga dapat didefinisikan ulang.

d. Pengenal (*inditifier*)

Pengenal digunakan untuk member nama variable, fungsi, konstanta dan lain-lain. Bahasa C bersifat *case sensitive* (huruf capital dan kecil dianggap berbeda). Konstruksi pengenal: huruf angka garis bawah(_). Tiap pengenal bisa menggunakan ketiga hal tersebut dengan catatan tidak boleh diawali dengan angka.

e. Variable

Variable adalah tempat untuk menyimpan dan mengakses data yang mewakili memori dalam *mikrokontrole*r. Variable harus dideklarasikan dengan tipe data bersama nama variabel yang akan digunakan.. tiap tipe data mempunyai jangkauan bilangan yang dapat disimpan, hal ini akibat dari byte memori yang dipesan dan bentuk bilangan bertanda atau tidak. Misalnya *unsigned char* oleh *compiler* disediakan 1 byte memori ram sehingga hanya bisa menampung

bilangan dari 0 sampai dengan 255 sedangkan jika bertanda -128 sampai dengan 127.

f. Larik (*Array*)

Array adalah variable yang berisi sekumpulan data yang mempunyai tipe data yang sama berbentuk matriks tunggal atau matrik multidimensional.

g. Operator

Operator adalah karakter-karakter khusus yang digunakan untuk manipulasi variable. Operator aritmatika, penugasan (*ossigment*), logika dan bit, relasi, pinter.