

PENGENDALIAN KECEPATAN MOTOR DC DENGAN METODE *LOOK UP TABLE* BERBASIS MIKROKONTROLER AT89C51

Muchlas, Sunardi, Tri Antoro

Program Studi Teknik Elektro Fakultas Teknologi Industri Universitas Ahmad Dahlan
Kampus III UAD, Jl. Prof. Dr. Soepomo, S.H., Janturan, Umbulharjo, Yogyakarta
Telp 0274-379418 psw 101/220, Fax 0274-381523
e-mail: muchlas@lycos.com, sunar_gm@yahoo.com, om_trian@yahoo.com

Abstrak

*Pengaturan kecepatan motor DC dapat dilakukan dengan tiga cara, yaitu: dengan mengubah tahanan rangkaian jangkar (R_a), mengubah fluks magnet (ϕ) dan mengubah tegangan (V). Tujuan penelitian ini adalah merancang sistem pengendali kecepatan motor DC dengan mengubah tegangan catu motor secara *look up table* berbasis mikrokontroler AT89C51 yang diharapkan bisa memberikan fleksibilitas yang lebih baik. Perancangan sistem dimulai dari perancangan rangkaian konversi digital ke analog, pengendali motor, sensor kecepatan, tombol keypad, driver LCD, catu daya, interkoneksi hardware dan dilanjutkan perancangan perangkat lunak pengatur sistem kerja pengendali motor. Sensor yang digunakan adalah optocoupler, konversi digital ke analog menggunakan DAC 0808, driver motor menggunakan transistor BD 137 dan pengatur sistem kerja menggunakan mikrokontroler AT89C51. Hasil penelitian menunjukkan bahwa pengendali motor yang dirancang dapat bekerja dengan akurasi 97,4%.*

Kata kunci: Kecepatan motor DC, Look Up Table, Mikrokontroler AT89C51

1. PENDAHULUAN

Motor listrik sebagai penggerak sangat luas penggunaannya, mulai dari alat-alat elektronik rumah tangga yang berdaya rendah sampai pada penggunaan di industri dengan daya ribuan dan jutaan watt. Berdasar pada catu daya listrik yang dipakainya, motor listrik dapat dibedakan menjadi motor DC dan motor AC yang masing-masing memiliki kelebihan. Pada motor DC relatif lebih mudah dalam hal pengendaliannya, tetapi rumit untuk perawatannya, sedangkan motor AC lebih sulit untuk dikendalikan dan karakteristiknya cukup non linier yang membuat tugas analisis lebih sulit.

Sistem kendali pada awalnya masih bersifat analog yang menggunakan kontaktor sebagai pengendalinya, namun dalam perkembangannya sistem kendali sudah bersifat digital artinya unit pengendalinya sudah menggunakan sistem digital seperti komputer dan mikrokontroler. Sistem kendali yang berbasis mikrokontroler sangat fleksibel, karena mempunyai kemampuan untuk mengubah metode atau sifat pengendali suatu proses tanpa mengubah perangkat kerasnya. Perubahan algoritma pengendali tersebut cukup dengan mengubah perangkat lunaknya, hal ini berarti sangat menghemat biaya dan juga tingkat efisiensinya cukup tinggi dibanding sistem kendali analog.

Pada industri-industri yang ada, perangkat-perangkat sistem kendali dibeli dalam bentuk jadi atau paket yang harganya sangat mahal. Keuntungan penggunaan mikrokontroler adalah bentuknya yang kecil, konsumsi daya yang rendah, dan sistem yang handal serta teliti. Selain itu mikrokontroler dapat dimodifikasi isi programnya tanpa harus mengubah perangkat kerasnya sehingga sangat fleksibel dengan keinginan.

Nugroho [3] dalam penelitiannya telah dapat mewujudkan sebuah pengendalian kecepatan motor DC dengan mikrokontroler Atmel AVR AT90S2313 dengan cara mengatur lebar pulsa/*PulseWidth Modulation* (PWM) dan menggunakan sensor *optocoupler* sebagai pencacah. Untuk menampilkan nilai putaran Nugroho [3] menggunakan LCD (*Liquid Crystal Display*).

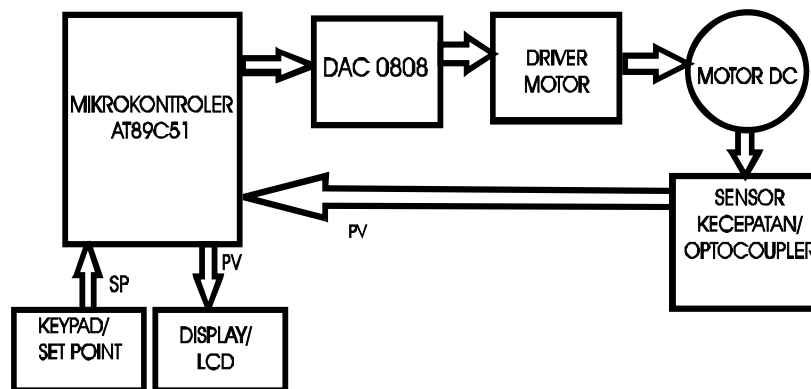
Pada penelitian ini akan dirancang pengendali kecepatan motor DC berbasis mikrokontroler AT89C51 dengan cara mengatur besar tegangan yang masuk pada motor yang diseting melalui *keypad* dan hasil pembacaan ditampilkan dengan LCD. Dalam penelitian ini pengaturan kecepatan dapat dilakukan melalui 11 tombol *keypad*, yang masing-masing tombol telah ditetapkan besarnya kecepatan motor. Metode yang digunakan dalam penelitian ini adalah metode *lookup table* atau tabel tengok, yaitu membandingkan kecepatan putar motor hasil pengukuran pengendali yang dibuat dengan hasil pengukuran menggunakan tachometer terhadap variasi *set point*.

2. METODE PENELITIAN

Pengendali kecepatan motor DC dirancang dengan menggunakan komponen-komponen yang spesifikasinya ditunjukkan pada Tabel 1. Diagram blok pengendali kecepatan motor DC pada penelitian ini ditunjukkan pada Gambar 1.

Tabel 1. Spesifikasi komponen penyusun pengendali kecepatan motor DC

No	Komponen	Spesifikasi
1	Mikrokontroler	AT89C51
2	Tombol keypad	Matrik 3x4
3	DAC	IC 0808
4	LCD	2 x 16 karakter
5	Catu daya	5V, -15V dan +15V/ 1A
6	Motor DC	12V – 18V
7	Sensor kecepatan	Optocoupler
8	Driver motor	Transistor BD137
9	Roda cacah	Diameter 5,3 cm



Gambar 1. Diagram blok pengendalian kecepatan motor DC

Agar rancangan alat pada Gambar 1 di atas dapat bekerja, di perlukan suatu perintah dalam bentuk perangkat lunak (program *assembly*). Tahapan perancangan perangkat lunak pengendalian kecepatan motor DC dilakukan sebagai berikut:

a) Proses pendefinisian variabel

Pendefinisian variabel diperlukan dalam pembuatan program khususnya program yang banyak membutuhkan tempat/lokasi untuk penyimpanan dan pengambilan data pada suatu alamat memori data tertentu, dengan kata lain variabel berfungsi untuk menggantikan register–register yang dimiliki mikrokontroler. Contoh potongan program pendefinisian variabel adalah sebagai berikut:

```

counter_5ms equ 30h
s_p equ 31h
p_v equ 32h
keyport equ p2
keybounc equ 33h
keydata equ 34h
ribuan equ 35h
ratusan equ 37h
puluhan equ 38h
satuan equ 39h
temp1 equ 3ah
temp2 equ 3bh
temp3 equ 3ch
temp4 equ 3dh
temp5 equ 3eh
pulrib equ 3fh
kolom1 bit p2.4
kolom2 bit p2.5
kolom3 bit p2.6
baris1 bit p2.0
baris2 bit p2.1
baris3 bit p2.2
baris4 bit p2.3

```

Penggunaan memori data dimulai pada alamat 30h. Untuk tiap variabel menempati lokasi 1 bit memori data internal mikrokontroler. Pendefinisian variabel secara lengkap dapat dilihat pada lampiran.

b) Proses inisialisasi

Inisialisasi dibagi menjadi dua macam yaitu inisialisasi variabel dan inisialisasi LCD. Inisialisasi variabel berfungsi untuk memberikan nilai awal terhadap suatu variabel yang sudah dibuat. Program inisialisasi variabel yang dibuat adalah sebagai berikut:

```

init_awal:
mov     p1,#00h
mov     p0,#00h
mov     keydata,#00h
mov     s_p,#00h
mov     p_v,#00h
ret

```

c) Inisialisasi LCD

Inisialisasi LCD diperlukan untuk mendeteksi dan mempersiapkan LCD sebagai penampil karakter sesuai dengan data ASCII yang dikirim. Potongan program inisialisasi LCD adalah sebagai berikut:

```

init_lcd:
mov     R1,#38h
lcall  tulisperintah
lcall  tulisperintah
mov     R1,#0Ch
lcall  tulisperintah
mov     R1,#06h
lcall  tulisperintah
mov     R1,#01h
lcall  tulisperintah
ret

```

d) Proses pembacaan tombol keypad

Cuplikan dari program pembacaan tombol keypad adalah sebagai berikut:

```

keypad:
    mov     keybounc,#50
    mov     keyport,#0FFH
    clr     kolom1
ul1:     jb     baris1,key1
        djnz    keybounc,ul1
        acall   set1
        acall   mulai1
        ret
key1:    jb     baris2,key2
        djnz    keybounc,key1
        acall   set2
        mov     keydata,#68
        ret
key2:    jb     baris3,key3
        djnz    keybounc,key2
        acall   set3
        mov     keydata,#138
        ret

```

Pertama yang dilakukan adalah pembacaan data pada keyport atau port2 karena tombol keypad dimasukkan pada port2. Data yang terbaca kemudian di bandingkan untuk membedakan tombol satu dengan yang lain. Untuk mengurangi getaran mekanis pada tombol keypad, tiap-tiap tombol keypad diberi perintah keybounc. Data pertama di bandingkan, jika belum sesuai dilanjutkan pada data pembanding kedua dan selanjutnya sampai menemukan pembanding yang sama. Jika sama maka akan menyimpan data jumlah putaran pada alamat keydata. Proses membandingkan ini merupakan proses untuk membaca tombol keypad yang di aktifkan. Program pembacaan tombol *keypad* secara lengkap dapat dilihat pada lampiran bagian *keypad*.

e) Proses delay/tunda

Proses tunda dibuat dengan perintah sebagai berikut:

```

tunda:
    MOV     R1,#0FFH
tunda1:  MOV     R2,#0
        DJNZ    R2,$
        DJNZ    R1,tunda1
        RET

```

Perintah pada baris ke 3 dikerjakan sebanyak $(255 \times 255) = 65025$ kali, karena instruksi tersebut dikerjakan selama 2 siklus maka waktu totalnya $(65025 \times 2) = 130050$ siklus, masih ditambah dengan pengulangan yang kedua sebanyak $(255 \times 3) = 765$ siklus, sehingga total siklus adalah $(130050 + 765) = 130815$ siklus. pada penggunaan osilator kristal sebesar 12 MHz, satu siklus mesin dikerjakan dalam waktu $1 \mu\text{S}$. Jadi lamanya penundaan yang di hasilkan oleh sub rutin diatas adalah $130815 \text{ siklus} \times 1 \mu\text{S} = 0,1 \text{ detik}$. Dalam baris ketiga terdapat huruf \$ mempunyai arti jika syarat terpenuhi kerjakan lagi instruksi bersangkutan.

f) Proses pembacaan timer/counter

Pada penelitian ini 2 buah *timer/counter* mikrokontroler AT89C51 digunakan untuk membaca pulsa yang datang dari sensor/optocoupler. Mode yang digunakan adalah mode 16 bit counter. *Timer 0* diaktifkan sebagai pencacah dan *timer 1* diaktifkan sebagai pewaktu yang akan menghentikan proses pencacahan pada *timer 0*. *Timer 0* menampung limpahan pulsa dari sensor yang disimpan di TL0 dan TH0 secara terus menerus sampai pencacahan dihentikan pada saat interupsi *timer 1* aktif (TF1=1) kemudian pulsa disimpan di alamat memori yang berlabel p_v.

Contoh program pencacah 16 bit

```

cacah:
    mov             counter_5mS,#200
Tunggu_1S:
    push          TMOD
    mov           TMOD,#00010101B; timer 16 bit
    mov           TH1,#0CH
    mov           TL1,#77H
    Setb         TR0
    setb         TR1
Tunggu_5mS:
    Jbc          TF1,sudah_5mS
    Ajmp         Tunggu_5mS
Sudah_5mS:
    CLR          TF0
    CLR          TF1
    CLR          TR0
    CLR          TR1
    Pop          TMOD
    Djnz        Counter_5mS,Tunggu-1S
    mov          a,TL0
    mov          p_v,a
    mov          TL0,#00h
    ret
  
```

Timer 0 difungsikan sebagai pencacah yang menampung jumlah cacahan pulsa (*event counter*) sedangkan timer 1 difungsikan sebagai interval *counter* (interval waktu penyamplangan), sehingga diperoleh periode penyamplangan (T) sebagai berikut:

$$T = \frac{12(\text{interval_counter})}{\text{Frekuensi_osilator}}$$

Karena frekuensi osilator yang digunakan adalah 12 MHz dan interval *counter* ditentukan yaitu 5000 *cycle* diulang sebanyak 200 kali, maka diperoleh interval *counter* = 5000 x 200 = 10⁶ *cycle*, sehingga:

$$T = \frac{12 \times 10^6}{12 \times 10^6} = 1 \text{ s}$$

g) Proses Konversi rps ke rpm

Peintah untuk konversi rps ke rpm adalah sebagai berikut:

```

kon_rps2rpm:
    mov          A,p_v
    mov          B,#10
    div         AB
    mov         temp1,A
    mov         temp2,B
    mov         A,#60
    mov         B,#10
    div         AB
    mov         temp3,A
    mov         satuan,B
    mov         A,temp3
    mov         B,temp1
    mul         AB
    mov         B,#10
    div         AB
    mov         ribuan,A
    mov         temp4,B
    mov         A,temp3
  
```

```

        mov     B,temp2
        mul     AB
        mov     B,#10
        div     AB
        mov     puluhan,B
        mov     temp5,A
        mov     A,temp5
        add     A,temp4
        jc     tambahan
        mov     ratusan,A
tambahan:
        mov     A,temp4
        add     A,temp5
        mov     B,#10
        div     AB
        mov     ratusan,B
        add     A,ribuan
        jc     tambahan1
        mov     ribuan,A
tambahan1:
        mov     a,ribuan
        mov     b,#10
        div     AB
        mov     pulrib,A
        mov     ribuan,B
        ret

```

Setelah nilai pulsa dimasukkan konversi rps ke rpm ini akan menghitung nilai yang diterima. Proses penghitungan ini dapat di jelaskan sebagai berikut: misal nilai bit 145,

- 1) Langkah pertama
Konversi akan membagi 145 dengan angka 10 (10 sebagai angka pembagi) dihasilkan 14,5. Angka 14 akan disimpan pada alamat temp1 dan angka 5 pada temp2.
- 2) Langkah kedua
Angka 60 (didapat dari 1menit adalah 60 detik) dibagi dengan angka 10 sehingga dihasilkan angka 6,0. Angka 6 akan disimpan pada alamat temp3 dan angka 0 disimpan sebagai satuan.
- 3) Langkah ketiga
Selanjutnya adalah temp3 dikalikan dengan temp1 (6×14) = 84, langkah selanjutnya membagi angka 84 dengan angka 10 dihasilkan 8,4. Angka 8 disimpan sebagai ribuan, kemudian angka 4 disimpan pada alamat temp4.
- 4) Langkah keempat
Langkah selanjutnya adalah mengalikan temp3 dengan temp2 (6×5) = 30, selanjutnya membagi angka 30 dengan angka 10 dihasilkan 3,0. Angka 0 disimpan sebagai puluhan, kemudian angka 3 disimpan pada alamat temp5.
- 5) Langkah kelima
Langkah ini adalah menambahkan temp5 dengan temp4 ($3 + 4$) = 7, angka 7 ini digunakan sebagai ratusan. Sehingga didapatkan hasil akhir 8700 rpm. Apabila dengan menggunakan cara cepat $145 \times 60 = 8700$ rpm.

h) Pengendali Kecepatan Putar Motor DC

Program pengendali kecepatan putar motor DC adalah gabungan dari semua program yang telah diujikan. Program ini terdiri dari program inialisasi variabel dan LCD, pembacaan tombol *keypad*, pembacaan sensor, konversi rps ke rpm, dan ditambah dengan program delay/tunda. Setelah semua piranti atau rangkaian disambungkan dengan mikrokontroler program ini kemudian *download* ke EVBU AT89C51 untuk dijalankan. Untuk mengoperasikan pengendali ini, pertama kali yang harus dilakukan adalah menghidupkan catu daya pengendali motor dengan menekan saklar *on/off* yang ada pada papan. Untuk menjalankan motor dengan menekan salah satu tombol pilihan yang sudah disediakan. 12 tombol keypad yang disediakan salah satunya adalah tombol *stop*, yaitu pada tombol 1. Untuk menghentikan putaran motor langsung ditekan tombol *stop*, dan bila akan menambah kecepatan putar motor bisa langsung ditekan tombol *keypad* yang mempunyai

angka *set point* lebih besar. Dan apabila ingin mengurangi kecepatan putar motor, cukup ditekan tombol keypad yang mempunyai angka *set point* lebih kecil. Kecepatan putar motor DC ditampilkan oleh LCD 2x16 karakter yang ada pada papan.

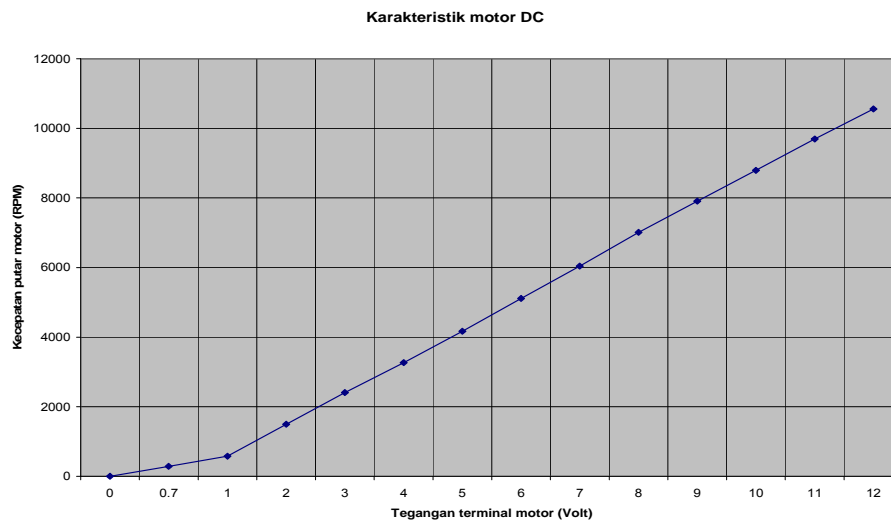
3. HASIL DAN PEMBAHASAN

Data yang pertama diambil dan dianalisis adalah tentang penentuan karakteristik motor DC. Penentuan karakteristik motor DC dilakukan dengan memberikan tegangan variabel pada terminal motor. Pemberian tegangan dibuat step dari 0 sampai 12 volt. Pengukuran kecepatan motor menggunakan tachometer dalam satuan rpm. Pengambilan data dilakukan sebanyak 5 kali (data pada Tabel 2).

Tabel 2. Data kecepatan putar motor terhadap tegangan terminal motor

Tegangan motor (volt)	
Data I	Data II
Data III	Data IV
Data V	Rata-rata
Data V	Rata-rata
0.7	1
2	3
4	5
6	7
8	9
10	11
12	

Hasil pengukuran diambil rata-ratanya sehingga menghasilkan grafik hubungan antara perubahan kecepatan putar terhadap kenaikan tegangan. Grafik dapat dilihat pada Gambar 2. Dari grafik Gambar 2 tersebut dapat dilihat bahwa kecepatan putar motor mengalami kenaikan yang stabil menyesuaikan besarnya tegangan yang diterima motor. Motor akan mulai berputar pada saat $V \geq 0,7$ volt.



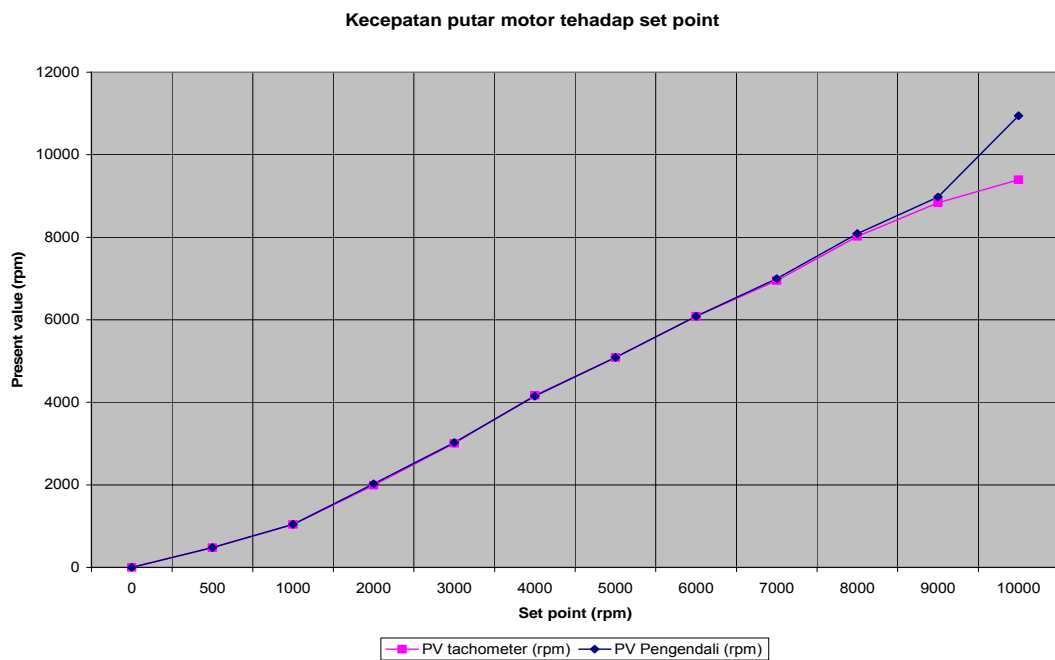
Gambar 2. Grafik karakteristik motor DC.

Data kedua yang diambil dan dianalisis adalah data rata-rata perbandingan kecepatan putar motor hasil pengukuran pengendali yang dibuat dengan hasil pengukuran menggunakan tachometer terhadap variasi *set point*. Pengambilan data dilakukan sebanyak 5 kali percobaan, kemudian diambil nilai rata-ratanya hasil pengukuran tersebut. Data dapat dilihat pada Tabel 3.

Tabel 3. Data rata-rata perbandingan kecepatan putar motor dengan *set point*.

<i>Set point</i> (rpm)	PV (Pengendali) rpm	PV (Tachometer) rpm
0	0	0,0
500	480	475,8
1000	1044	1040,0
2000	2024	1991,8
3000	3024	3003,0
4000	4152	4166,8
5000	5088	5083,2
6000	6084	6083,8
7000	6996	6948,2
8000	8088	8020,6
9000	8976	8833,2
10000	10944	9393,6

Dari data Tabel 3 diperoleh grafik perbandingan antara kecepatan putar motor hasil pengukuran pengendali yang dibuat dengan hasil pengukuran menggunakan tachometer terhadap variasi *set point*. Grafik dapat disajikan pada Gambar 3.



Gambar 3. Perbandingan kecepatan putar motor dengan *set point*

Hasil pengukuran kecepatan putar motor berdasarkan pengendali yang dibuat dengan pengukuran kecepatan putar motor berdasarkan tachometer tidak begitu jauh selisihnya dengan *set point* seperti ditunjukkan pada grafik Gambar 3. Namun pada *set point* 10000 rpm hasil pembacaan tidak bisa stabil (PV naik turun terlalu tinggi). Hal ini disebabkan oleh roda cacah yang hanya memiliki 1 takik/lubang. Pada kecepatan tinggi optocoupler akan sulit membaca kecepatan. Selain hal itu, takik/lubang yang ada pada roda cacah juga terlalu kecil. Sehingga dalam kecepatan tinggi menyulitkan optocoupler menerima cahaya dari inframerah. Kesalahan/error antara kecepatan putar motor yang diinginkan (SP) dengan kecepatan putar motor (PV) hasil pembacaan pengendali disajikan pada Tabel 4.

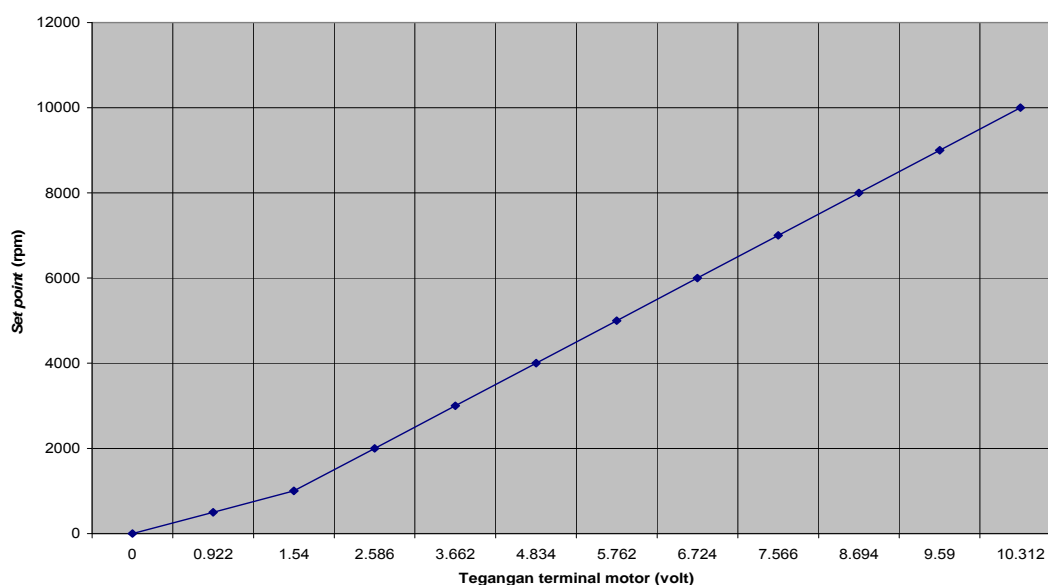
Tabel 4. Data kesalahan/error *set point* terhadap *present value*.

<i>Set point</i> (SP)	<i>Present value</i> (PV)	Tingkat kesalahan/error (%)
0	0	0,00
500	480	4,10
1000	1044	4,20
2000	2024	1,80
3000	3024	0,70
4000	4152	3,60
5000	5088	1,70
6000	6084	1,30
7000	6996	0,05
8000	8088	1,00
9000	8976	0,20
10000	10944	8,60
Tingkat kesalahan rata-rata		2,40

Tabel 5. Data rata-rata pengukuran tegangan terminal motor.

<i>Set point</i> Kecepatan rpm	Pengukuran tegangan terminal motor (volt)					Rata-rata
	Data I	Data II	Data III	Data IV	Data V	
0	0,00	0,00	0,00	0,00	0,00	0,000
500	0,91	0,92	0,93	0,92	0,92	0,922

1000	1,53	1,55	1,52	1,55	1,55	1,540
2000	2,58	2,59	2,58	2,59	2,59	2,586
3000	3,65	3,67	3,66	3,67	3,66	3,662
4000	4,82	4,84	4,83	4,84	4,84	4,834
5000	5,75	5,77	5,76	5,77	5,76	5,762
6000	6,69	6,73	6,73	6,74	6,73	6,724
7000	7,53	7,57	7,57	7,58	7,58	7,566
8000	8,64	8,72	8,68	8,71	8,72	8,694
9000	9,55	9,61	9,59	9,60	9,60	9,590
10000	10,30	10,31	10,32	10,31	10,32	10,312

Hubungan tegangan terhadap *set point*Gambar 4. Grafik tegangan terhadap *set point*

Data yang terakhir yang diambil dan dianalisis adalah pengukuran tegangan pada terminal motor dengan menggunakan voltmeter. Hasil pengukuran tegangan tersebut disajikan pada Tabel 5. Berdasarkan Tabel 5, selanjutnya dibuat grafik hubungan tegangan terhadap *setpoint*, dan diperoleh grafik seperti ditunjukkan pada Gambar 4. Dari grafik ini dapat ditunjukkan bahwa kenaikan tegangan adalah berpola menyesuaikan besarnya *set point* kecepatan motor. Semakin besar *set point* yang dimasukkan, akan mengikuti pula besarnya tegangan yang diterima motor.

4. KESIMPULAN

Berdasarkan hasil dan pembahasan penelitian pengendalian kecepatan motor DC berbasis mikrokontroler di atas dapat diambil kesimpulan sebagai berikut:

1. Telah dapat direalisasikan pengendali kecepatan putar motor DC dengan metode *lookup table* berbasis mikrokontroler AT89C51 dalam mode *single chip*.
2. Tingkat akurasi pengendalian berdasarkan *set point* (SP) terhadap *present value* (PV) adalah sebesar 97,6%.

DAFTAR PUSTAKA

1. Putra, E.A., "Belajar Mikrokontroler AT89C51/52/55 Teori dan Aplikasi", Gava Media, Yogyakarta, 2002.
2. Nugroho, "Pengendalian Kecepatan Motor DC dengan Mikrokontroler Atmel AVR AT90S2313", Skripsi S-1, Institut Sains dan teknologi Akprind, Yogyakarta, 2003.

3. Sumanto, M.A., **"Mesin Arus Searah (Generator Arus Searah, Motor Arus Searah)"**, Andi Offset, Yogyakarta, 1991.
4. Benjamin, C.K., **"Teknik Kontrol Automatik- Automatic Control System"**, seventh edition, PT. Aditya Media, Yogyakarta, 1995.
5. Nalwan A.P., **"Panduan Praktis Teknik Antarmuka dan Pemrograman Mikrokontroler AT89C51"**, Elex Media Computindo, Jakarta, 2003.