DESAIN DAN IMPLEMENTASI KONTROLER P dan PI UNTUK PENGATURAN KECEPATAN MOTOR DC dengan INTERFACE PCI 4716

1. TUJUAN PERCOBAAN

Percobaan ini bertujuan untuk :

- Melakukan identifikasi plant motor DC dengan PCI 4716
- Mendesain dan mengimplementasikan kontroler P dan PI untuk mengatur kecepatan putar motor DC dengan PCI 4716.

2. ALAT-ALAT YANG DIBUTUHKAN

Berikut beberapa alat yang harus dipersiapkan:

- *Servo Amplifier* SA150D
- Motor DC M150F
- *Tachometer* GT 150X
- *Power supply* PS150E
- Modul PCI 4716
- *Multimeter*
- Komputer (PC) yang di-*install software* LabVIEW 2013 dan DAQ untuk PCI 4716
- Kabel penghubung

3. DASAR TEORI

3.1. Sistem *Servo Modular* MS 150

Sistem *servo modular* MS 150 merupakan blok rangkaian elektronik yang digunakan untuk pengendalian kecepatan dan posisi dari motor servo DC. Sistem *servo modular* MS 150 terdiri dari suatu pemakain sumber tenaga (*power supply*), *servo amplifier*, dua unit motor DC, *reduction gear tacho unit*, dan beberapa transducer dan modul.

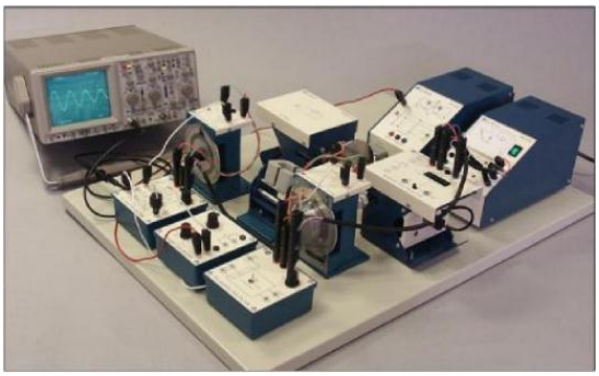
Bagian-bagian sistem *servo modular* MS150 yang digunakan untuk mengidentifikasi motor DC, yaitu :

a. Power Supply

Masukan yang disediakan sebesar 115V dan 230V, 50/60 Hz, 40VA. Dengan keluaran yang dihasilkan sebesar 24V DC, 2A yang diatur untuk menyuplai motor.

b. Motor DC

Motor DC yang digunakan yaitu motor magnet permanen yang mempunyai *shaft* yang dipanjangkan. *Shaft* motor diperpanjang untuk bantalan atau ganjalan secara langsung dari pengereman magnetik dan inersia.



Gambar 1. *Servo Modular* MS 150

c. *Reduction Gear Tacho Unit*

Modular feedback tachogenerator GT150X berfungsi untuk mengkonversi besaran mekanik menjadi besaran listrik.

d. *Servo Amplifier*

Modular *feedback servo amplifier* SA150D terdiri dari rangkaian *transistor* yang dapat menggerakkan motor DC dengan dua arah putaran.

Pengaturan kecepatan dapat dilakukan dengan mengatur sinyal masukan (tegangan) pada motor karena kecepatan motor dipengaruhi oleh besar tegangan masukan yang diberikan. Sistem pengaturan yang demikian biasa disebut sistem pengaturan kecepatan *loop* terbuka. Dalam keadaan beban yang konstan atau dalam keadaan tanpa beban, sistem pengaturan *loop* terbuka masih mungkin digunakan. Akan tetapi, dalam keadaan beban yang berubah-ubah, sistem *loop* terbuka tersebut sudah tidak dapat diandalkan.

3.2. Sistem pengaturan *loop* terbuka

Sistem pengaturan *loop* terbuka merupakan suatu sistem pengaturan yang keluarannya tidak mempunyai pengaruh terhadap aksi kontrol. Pada sistem pengaturan *loop* terbuka tidak terdapat jaringan umpan balik. Oleh karena itu sistem pengaturan *loop* terbuka hanya dapat

digunakan jika hubungan antara masukan dan keluaran sistem diketahui dan tidak terdapat gangguan internal maupun eksternal.

3.3. Sistem pengaturan *loop* tertutup

Sistem pengaturan *loop* tertutup merupakan sistem pengaturan dimana sinyal keluaran mempunyai pengaruh langsung terhadap sinyal kontrol (aksi kontrol). Pada sistem pengaturan *loop* tertutup terdapat umpanbalik (*feedback*) karena itu sistem pengaturan *loop* tertutup seringkali disebut sebagai sistem pengaturan umpanbalik.

3.4. Pengenalan PCI 4716

PCI merupakan instrument *amplifier* dengan *gain* yang dapat diprogram yang memungkinkan membuat sebuah sinyal input tanpa adanya pengkondisian sinyal eksternal. Di dalam PCI terdapat *buffer* FIFO yang menyediakan transfer data dengan kecepatan tinggi dan memori SRAM yang memungkinkan melakukan konversi A/D multichannel.

Beberapa fungsi yang diperoleh dengan menggunakan PCI 4716 antara lain :

1. Konversi A/D 12 bit
2. Konversi D/A
3. Input Digital
4. Output Digital
5. Counter atau Timer

Berikut ini merupakan fitur-fitur yan dimiliki PCI 4716 :

- *16 single-ended/ 8 differential or combination analog input channels*
- *16-bit resolution A/D converter, with up to 200 kS/s sampling rate*
- *8 digital input & 8 digital output channels (TTL Level)*
- *2 analog output channels*
- *16-bit programmable counter/timer x 1*
- *Programmable gain for each analog input channel*
- *Automatic channel/gain scanning*
- *Bus-powered*
- *Device status LED indicator*
- *Removable on-module wiring terminal*
- *Supports high-speed USB 2.0*
- *Auto calibration function*

Pengertian PCI (*Peripheral Component Interconnect*) adalah *bus* yang didesain untuk menangani beberapa perangkat keras. PCI (*Peripheral Component Interconnect*) merupakan bus yang tidak tergantung pada prosesor dan *bandwidth* tinggi yang berfungsi sebagai *bus* periferal. PCI memberikan sistem yang lebih baik bagi subsistem I/O berkecepatan tinggi. PCI dirancang untuk memenuhi kebutuhan I/O dan lebih banyak digunakan untuk mendukung bermacam-macam konfigurasi berbasis microprosesor.

Untuk struktur *bus*, pada umumnya PCI dapat dikonfigurasikan sebagai *bus* 32 bit atau 64 bit. Sedangkan setiap transfer data pada *bus* PCI terdiri dari fase alamat dan fase data. PCI juga memanfaatkan aturan-aturan sentral dan sinkron yang masternya memiliki *request* unik (REQ) dan *signal grant* (GNT).

Gambar 2. Bentuk fisik PCI 4716

3.5. DAQ

Akuisisi data merupakan bagian dari proses pengambilan sample dari kondisi real untuk menghasilkan data yang dapat dimanipulasi oleh komputer. Sistem akuisisi data (**DAS** atau **DAQ)** biasanya mengubah gelombang analog menjadi digital. Komponen pada sistem akuisisi data meliputi:

- Sensor, untuk mengkonversi parameter fisik menjadi sinyal listrik.
- Rangkaian *signal conditioning*, untuk mengkonversi sinyal sensor menjadi bentuk yang dapat dikonversi ke nilai digital.
- *Analog to Digital converter*, untuk mengkonversi sinyal analog menjadi sinyal digital.

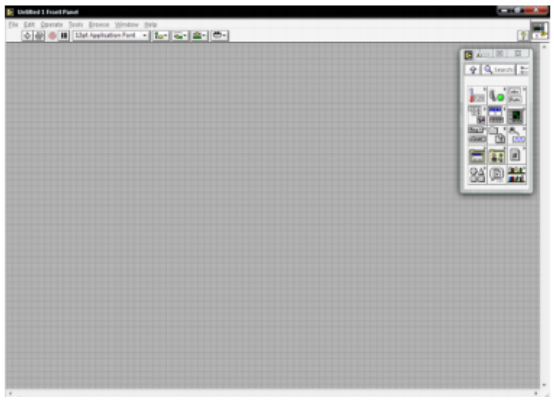
Hardware DAQ biasanya berupa modul yang dapat dihubungkan ke *port* komputer (paralel, seri, USB) atau *card* yang terhubung ke slot (S-100 *bus*, *AppleBus*, ISA, MCA, PCI, PCI-E) pada *motherboard* . Biasanya ruang di bagian belakang *card* PCI terlalu kecil untuk semua koneksi yang diperlukan, sehingga diperlukan modul tambahan untuk koneksi I/O. *Card* DAQ mengandung beberapa komponen (*multiplexer*, ADC, DAC, TTL-IO, *timer* kecepatan tinggi, RAM). Semuanya dapat diakses melalui *bus* oleh mikrokontroler yang dapat menjalankan program-program sederhana. Sebuah kontroler akan lebih fleksibel daripada *logic* dengan kabel, namun lebih murah daripada CPU.

3.6. LabVIEW

LabVIEW adalah sebuah *software* pemrograman yang diproduksi oleh *National Instruments* dengan konsep yang berbeda. Seperti bahasa pemograman lainnya yaitu C++, MATLAB atau Visual basic, LabVIEW juga mempunyai fungsi dan peranan yang sama, perbedaannya bahwa LabVIEW menggunakan bahasa pemrograman berbasis grafis atau blok diagram sementara bahasa pemrograman lainnya menggunakan basis text. Program LabVIEW dikenal dengan sebutan "VI" atau *Virtual Instruments* karena penampilan dan operasinya dapat meniru sebuah instrument. Pada LabVIEW, user pertama-tama membuat *user interface* atau *front panel* dengan menggunakan kontrol dan indikator, yang dimaksud dengan kontrol adalah *knobs*, *push buttons*, *dials* dan peralatan *input* lainnya sedangkan yang dimaksud dengan indikator adalah *graph*, LED dan peralatan *display* lainnya. Setelah menyusun *user interface*, lalu *user* menyusun blok diagram yang berisi kode-kode VI untuk mengatur *front panel*. *Software* LabVIEW terdiri dari tiga komponen utama, yaitu :

1. *Front Panel*

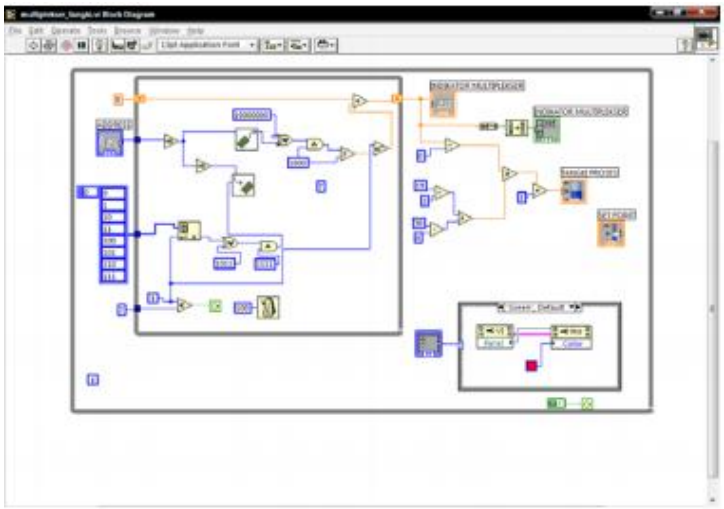
Front panel adalah *window* yang berlatar belakang abu-abu serta mengandung kontrol dan indikator. *Front panel* digunakan untuk membangun sebuah VI, menjalankan program dan men-*debug* program. Tampilan dari *front panel* dapat dilihat pada Gambar 3.



Gambar 3. *Front Panel*

2. Blok diagram dari VI

Blok diagram adalah *window* yang berlatar belakang putih berisi *source code* yang dibuat dan berfungsi sebagai instruksi untuk *front panel*. Tampilan dari blok diagram dapat lihat pada Gambar 4.



Gambar 4. Blok Diagram

3. *Control* dan *Functions Pallete*

Control dan *Functions Pallete* digunakan untuk membangun sebuah VI.

a. Control Pallete

Control Pallete merupakan tempat beberapa kontrol dan indikator pada *front panel*, *control pallete* hanya tersedia di *front panel*, untuk menampilkan *control pallete* dapat

dilakukan dengan mengklik *windows >>show control pallete* atau klik kanan pada *front panel*. Contoh *control pallete* ditunjukkan pada Gambar 5.

Gambar 5. *Control Palette*

b. Functions Pallete

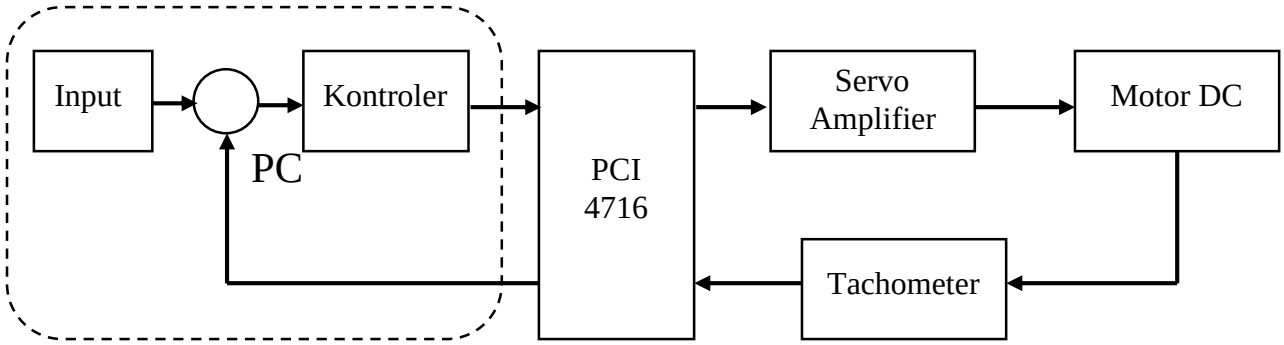
Functions Pallete digunakan untuk membangun sebuah blok diagram, *functions pallete* hanya tersedia pada blok diagram, untuk menampilkannya dapat dilakukan dengan mengklik *windows >> show control pallete* atau klik kanan pada lembar kerja blok diagram. Contoh dari *functions pallete* ditunjukkan pada Gambar 6.

Gambar 6. *Functions pallete*

4. PROSEDUR PRAKTIKUM

Pada Praktikum ini, terdapat tiga percobaan yang dilakukan, yaitu :

- Identifikasi sistem :
  - Identifikasi Statis
  - Identifikasi Dinamis
- *Close loop* dengan kontroler P
- *Close loop* dengan kontroler PI



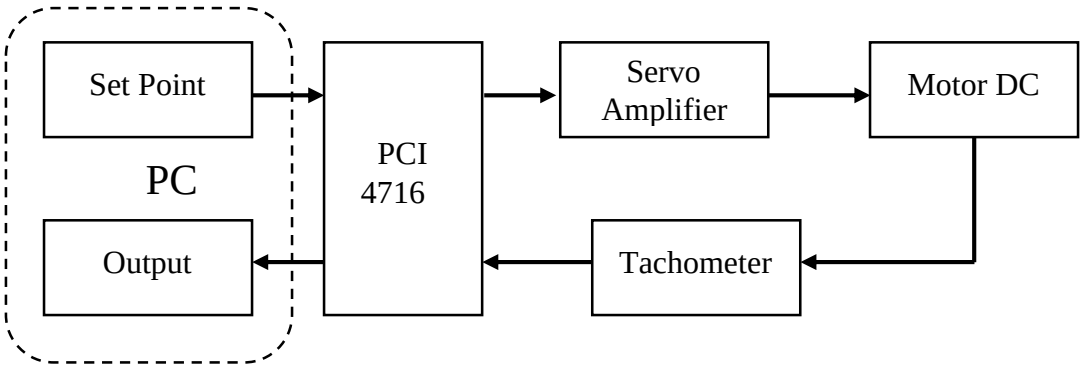
Gambar 7. Konfigurasi Sistem Pengaturan Digital Motor DC

4.1. Percobaan Identifikasi Statis dan Dinamis

4.1.1. Membuat Sistem *Loop* Terbuka

Dalam melakukan identifikasi statis dan dinamis perlu dirancang rangkaian dengan sistem *loop* terbuka. Berikut langkah-langkah dalam melakukan identifikasi.

1. Buat rangkaian sistem *loop* terbuka seperti yang ditunjukkan pada gambar 8.



Gambar 8. Konfigurasi Sistem Pengaturan *Loop* Terbuka Motor DC

2. Hubungkan PC dengan modul PCI 4716.
3. Hubungkan tiap-tiap modul dengan modul *Power supply* PS150E.

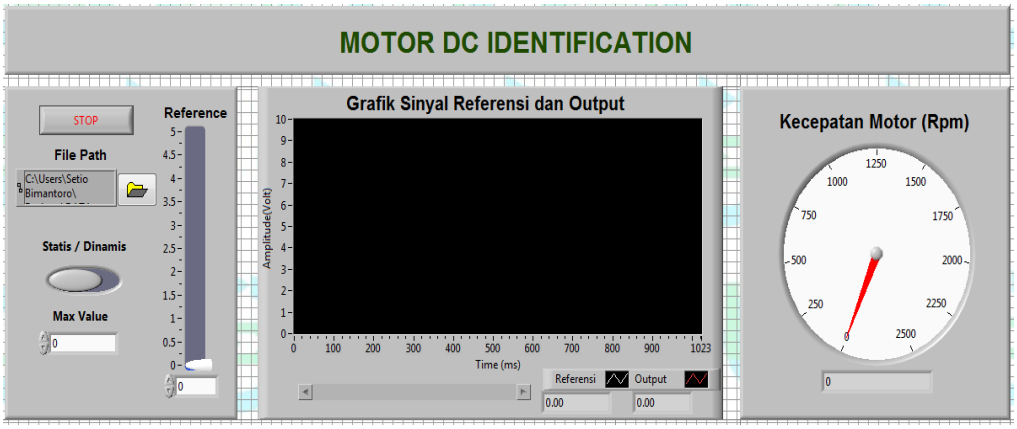
   Perhatian : Modul *Servo Amplifier* SA150D sudah terhubung catu daya sehingga tidak perlu dihubungkan ke modul *power supply*.
4. Hubungkan *Analog Output* (AO) pada PCI board dengan pin 2 modul *Servo Amplifier* SA150D.
5. Hubungkan pin 1 modul *Tachometer* dengan *Analog Input* (AI) PCI *board* dan menghubungkan pin 2 modul *Tachometer* dengan *ground*.

   Perhatian : Pada rangkaian sistem *loop* terbuka, jangan hubungkan pin 1 modul *Tachometer* dengan pin 2 *Operational Unit* OU150A.

6. Hubungkan pin *ground* PCI *board* dengan ground rangkaian motor DC.
7. Ikuti prosedur percobaan menggunakan *software* LabVIEW.

4.1.2. Prosedur Penggunaan *Software* LabVIEW

1. Buka LabVIEW 2013.
2. Buka *file* "Identifikasi Motor DC.vi".



Gambar 9. Tampilan Program Identifikasi Motor DC LabVIEW

4.1.2.1. Prosedur Memasukkan DAQ pada LabVIEW

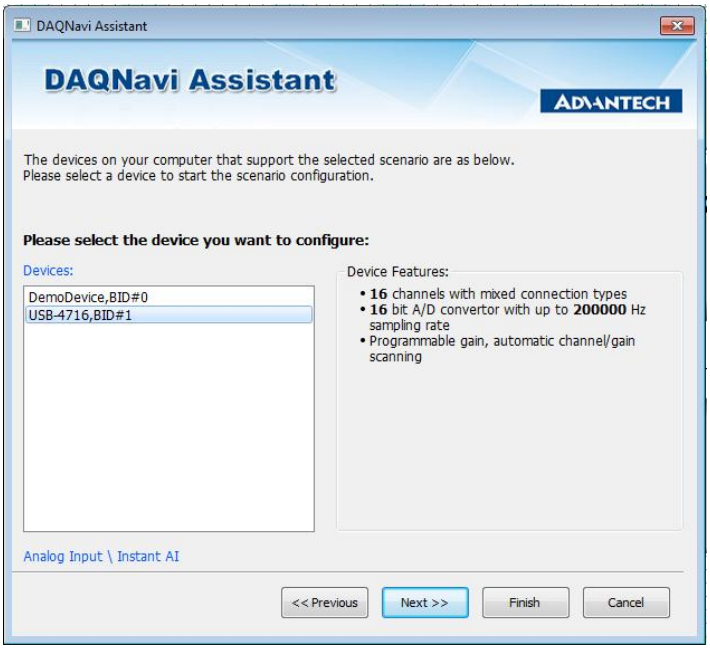
1. Klik kanan pada jendela blok diagram LabVIEW.
2. Pilih *Measurement* I/O → DAQ *Navi* → DAQ *Assistant*.
3. Posisikan DAQ pada tempat yang diinginkan.
4. Tentukan *analog input/output*, seperti pada gambar 10.

Gambar 10. Konfigurasi Fungsi

5. Pilih *Instant* AI/AO.

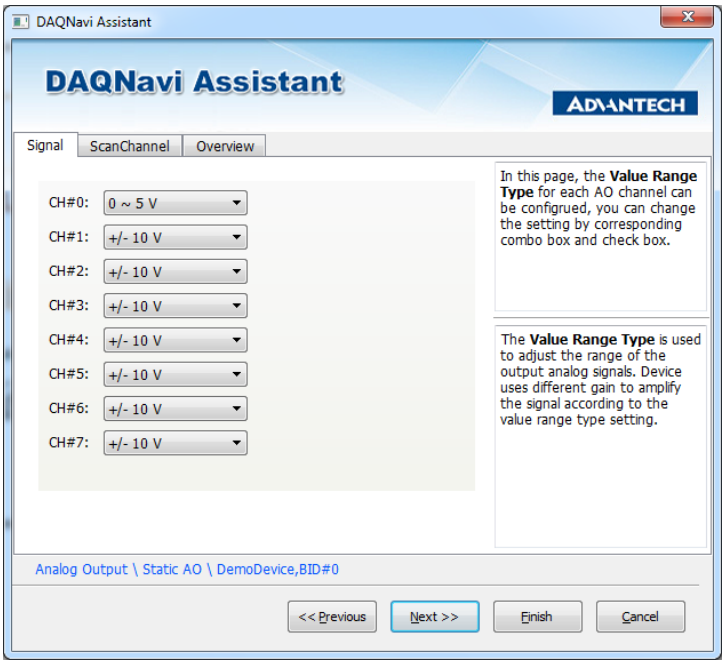
Gambar 11. Konfigurasi Skenario Fungsi

6. Untuk *device* yang ingin dikonfigurasi, pilih PCI-4716, BID#0



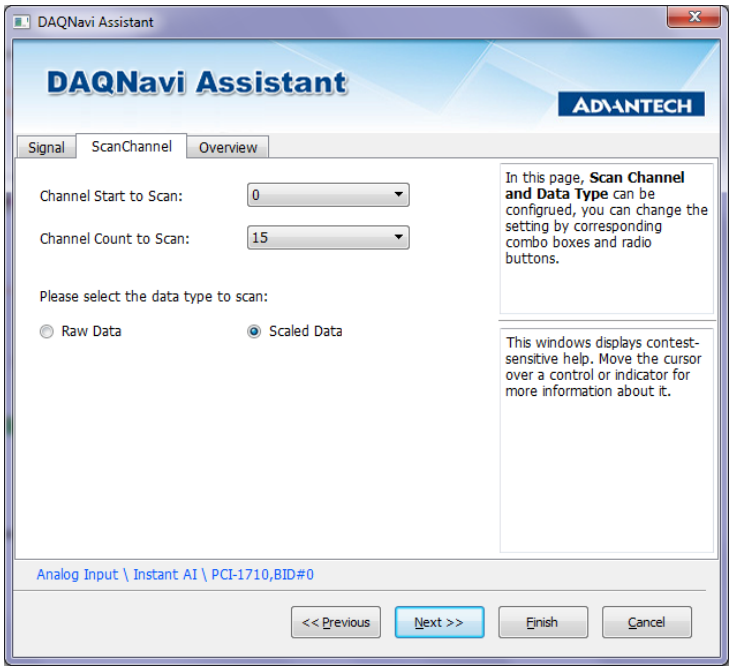
Gambar 12. Konfigurasi *Device* Fungsi

7. Pilih *channel* yang diinginkan, pada kolom *signal* pilih *range* tegangan 0 ~ 5 V.



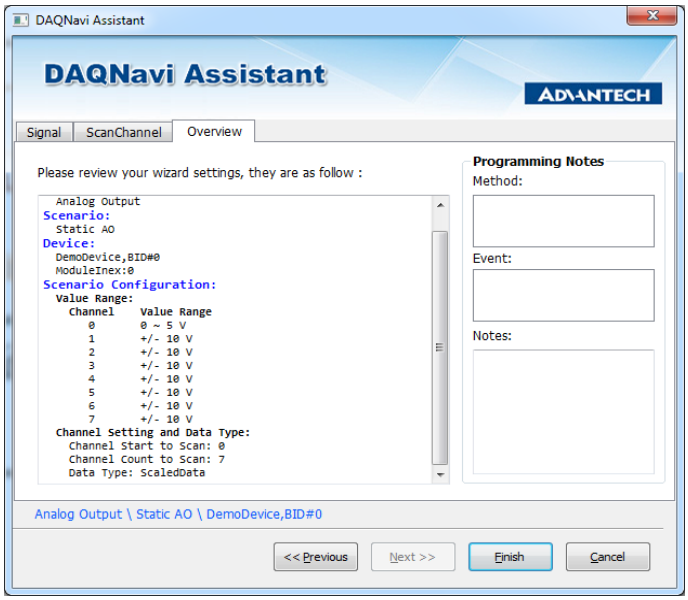
Gambar 13. Konfigurasi *Channel* Fungsi (1)

8. Atur *Scan channel* = 0, *Start* = 0, *Count* = 15, *Scaled Data*



Gambar 14. Konfigurasi *Channel* Fungsi (2)

9. *Overview* konfigurasi yang telah ditentukan sebelumnya akan muncul, lalu pilih *Finish*.



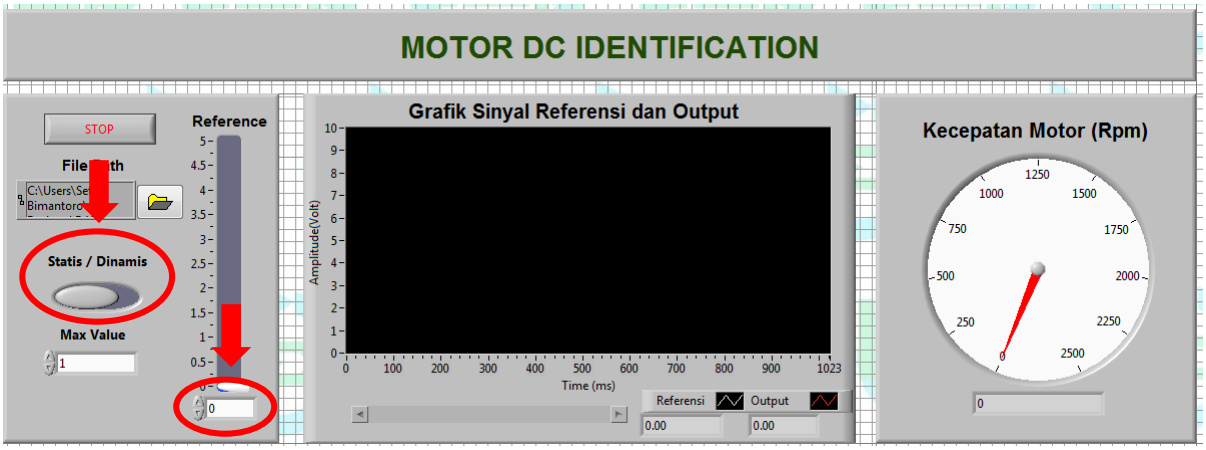
Gambar 15. Konfigurasi *Channel* Fungsi (3)

Perhatian :

Untuk konfigurasi *analog input* ditampilkan sebagai *Data Out* (DDT) dan *analog output* ditampilkan sebagai *Data In* (DDT) pada DAQ Navi

4.1.2.2. Percobaan Identifikasi Sistem Statis

1. Pilih posisi saklar pada posisi Statis.
2. Untuk identifikasi statis, masukan untuk Motor DC berupa tegangan referensi yang dapat diubah dan maksimal masukan sebesar 1V ! Seperti gambar 16.



Gambar 16. Tampilan program LabView pada Identifikasi Statis

3. Pada bagian file path, buat file berekstensi *.txt, kemudian masukkan file tersebut untuk menyimpan data hasil percobaan.
4. Jalankan program selama beberapa detik untuk pengambilan data percobaan identifikasi.
5. Apabila konfigurasi benar, maka motor DC akan berputar dan grafik dari pergerakan motor tersebut akan muncul.
6. Ubah data hasil percobaan menjadi grafik (waktu dengan tegangan input dan output) dengan menggunakan perangkat lunak MATLAB.
7. Dapatkan pemodelan menggunakan identifikasi statis.

Analisa !

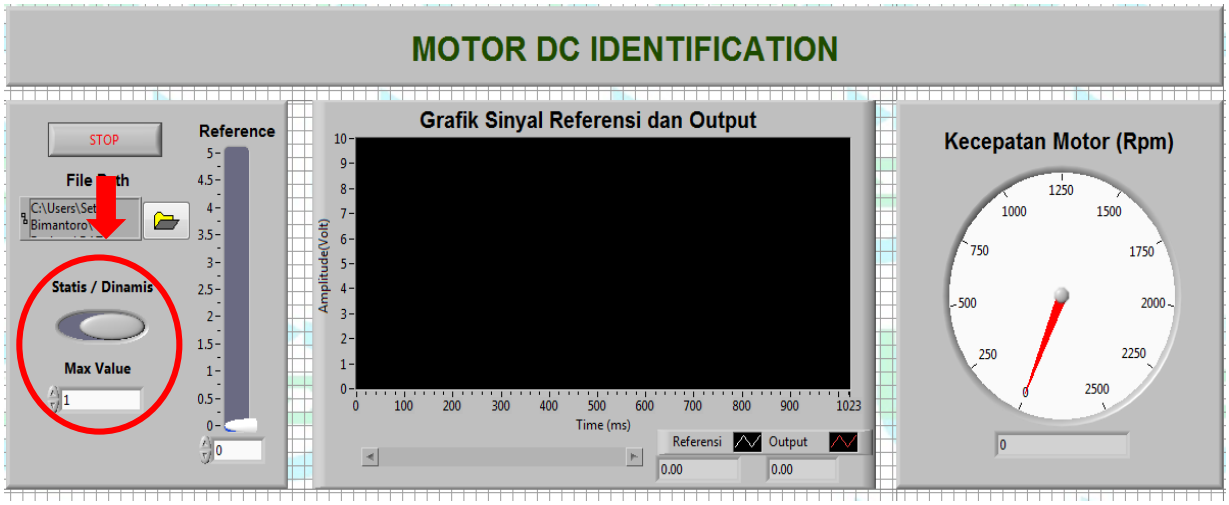
Lakukan analisa terhadap grafik tersebut! Lakukan identifikasi statis dengan cara mendapatkan model matematis plant motor DC berupa fungsi alih yang didapat dari grafik!

Tabel 1. Format Keluaran dari *Notepad*

Iterasi	Waktu (ms)	Tegangan Referensi (Volt)	Tegangan Motor (Volt)	Kecepatan Motor (RPM)
…..	…..	…..	…..	…..

4.1.2.3. Percobaan Identifikasi Sistem Dinamis

1. Pilih posisi saklar pada posisi dinamis
2. Untuk identifikasi dinamis , masukan untuk Motor DC terdapat pada *Max Value* yang dapat diubah dan maksimal masukan sebesar 1V ! Seperti gambar 17.



Gambar 17. Tampilan program LabView pada Identifikasi Dinamis

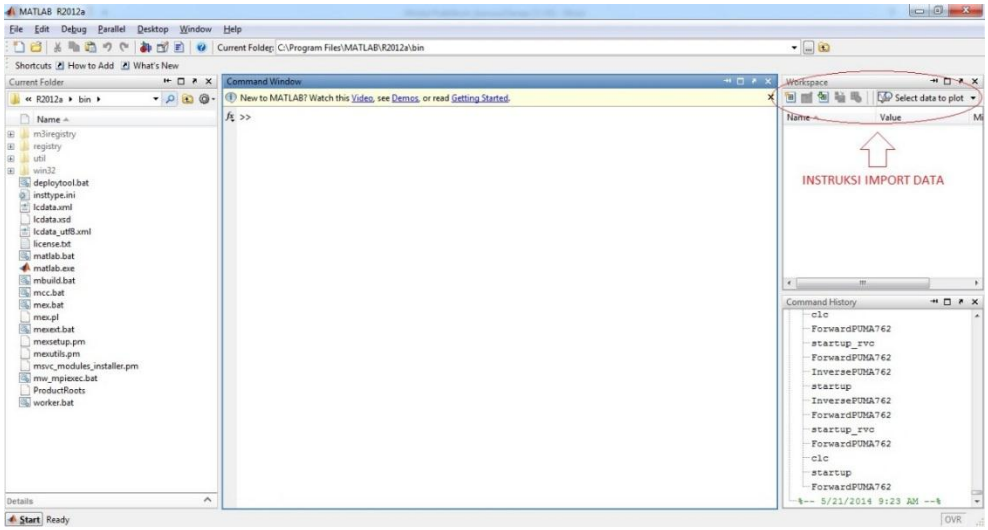
3. Pada bagian *file path*, buat *file* berekstensi *.txt, kemudian masukkan file tersebut untuk menyimpan data hasil percobaan.
4. Jalankan program selama beberapa detik untuk pengambilan data percobaan identifikasi.
5. Apabila konfigurasi benar, maka motor DC akan berputar dan grafik dari pergerakan motor tersebut akan muncul.
6. Ubah data hasil percobaan menjadi grafik dengan menggunakan perangkat lunak MATLAB. Format keluaran pada *notepad* sama seperti Tabel 1.
7. Dengan menggunakan *toolbox simulink*, dapatkan pemodelan menggunakan item identifikasi dinamis.

Pelajari !

Pelajari teori dasar identifikasi dinamis! Mengapa identifikasi dinamis menggunakan sinyal input berupa PRBS?

4.1.3. Prosedur Penggunaan Perangkat Lunak MATLAB

1. Buka *file* *.txt yang telah terisi data hasil percobaan dengan menggunakan *notepad*, ubah tanda koma (‘,’) dengan titik (‘.’) dengan menggunalan Ctrl+F → Replace all.
2. Buka program MATLAB.
3. Pada jendela *workspace*, *import data* percobaan yang telah didapatkan sebelumnya, instruksi bisa ditemukan seperti Gambar 18.



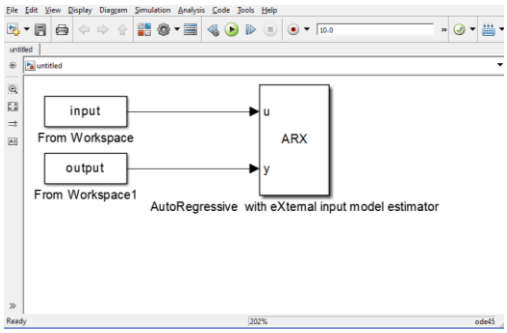
Gambar 18. Tampilan *Import Data* MATLAB

4. Format dari *notepad* yang dimasukkan sama seperti pada format keluaran LabVIEW, dan akan dibentuk data *array* baru untuk *file workspace* pada MATLAB.
5. Ubah waktu sampling ke dalam satuan detik dan pisahkan data input dan output, dengan cara menjalankan program "setup.m" yang berisi *script* seperti di bawah ini.

```
data(:,2) = data(:,2)-data(1,2); % Waktu dikurangi waktu sebelumnya
data(:,2) = data(:,2)/1000;      % Merubah satuan dari ms ke s
plot(data(:,2),[data(:,3) data(:,4) data(:,5)]);%Grafik referen & output
input = [data(:,2) data(:,3)];   % Membuat data input
output = [data(:,2) data(:,4)];  % Membuat data output
```

Perhatian : Fungsi "data" diganti sesuai nama *workspace* yang diambil.

6. Buat plot grafik untuk sinyal tegangan referensi, sinyal *error* dan sinyal tegangan keluaran.
7. Buka data percobaan identifikasi dinamis, pastikan input dan output merupakan variabel yang benar, lalu *run program* (identifikasi sistem dinamis).
8. Buka *Simulink* dan buat blok seperti gambar 19.



Gambar 19. Program Identifikasi Dinamis dengan ARX

- ARX dapat diperoleh di *System Identification Tools*.
- *From Workspace* dapat diperoleh di *Sources*.

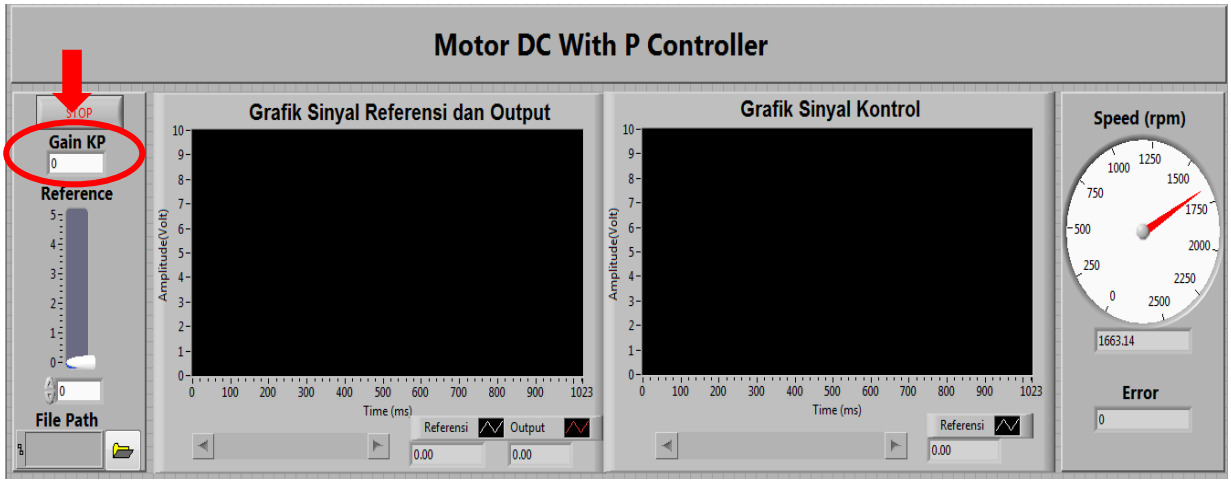
9. Setelah *Simulink* dijalankan maka akan didapatkan *transfer function* yang digunakan untuk menentukan model matematis motor DC.

4.2. Percobaan Sistem *Loop* Tertutup Menggunakan Kontroler P

Langkah kerja :

1. Rencanakanlah kontroler proporsional yang mempunyai respon sistem dengan konstanta waktu 2x lebih cepat dari semula dan hitung pula *ess* dari sistem hasil rancangan.
2. Gunakan rangkaian motor DC sistem *loop* tertutup.

3. Buka LabVIEW 2013 dan buka *project* "P Control Motor DC.vi" serta isi *notepad* untuk bagian *file* kontroler P.
4. Lakukan perhitungan untuk mencari nilai Kp berdasarkan data yang didapatkan dari percobaan identifikasi statis.
5. Masukkan nilai Kp yang telah diperoleh pada kolom gain Kp LabVIEW seperti pada Gambar 20. Kemudian jalankan LabVIEW dan lihat perubahan grafik yang dihasilkan.



Gambar 20. Blok Diagram LabVIEW kontroler P

6. Ubah data hasil percobaan menjadi grafik dengan menggunakan perangkat lunak MATLAB. Lalu lakukan analisa terhadap grafik data percobaan tersebut.

Tabel 2 . Format Keluaran dari *Notepad* kontroler P

Iterasi	Waktu (ms)	Tegangan Referensi (Volt)	Eror (Volt)	Tegangan Kontrol (Volt)	Tegangan Motor (Volt)	Kecepatan Motor (RPM)
…..	…..	…..	…..	…..		

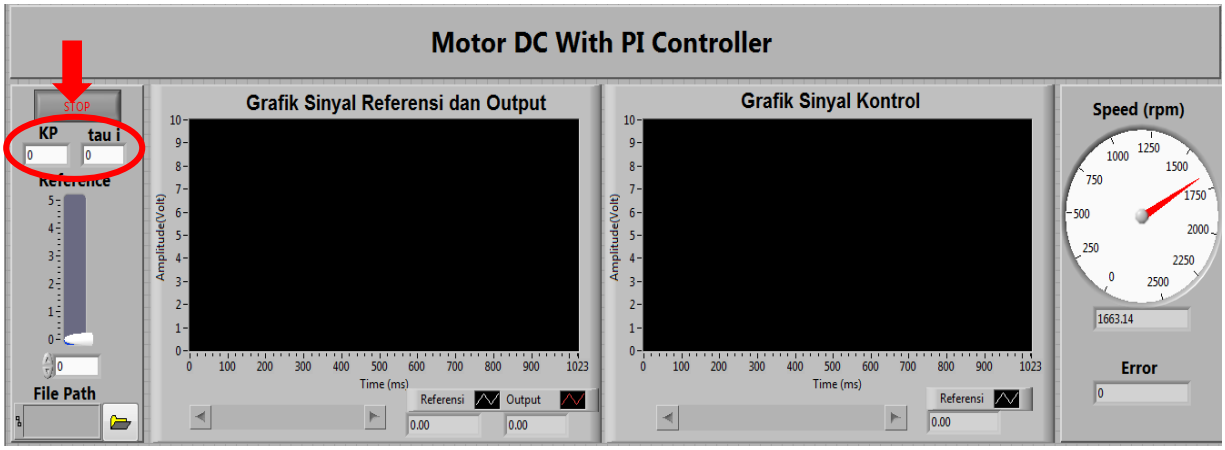
Analisa !

Lakukan analisis terhadap kontroler tipe P ! Bandingkan hasil perhitungan dengan hasil percobaan, apakah hasil percobaan telah sesuai dengan kriteria yang ditentukan!

4.3. Percobaan Sistem *Loop* Tertutup Menggunakan Kontroler PI

Langkah kerja :

1. Rencanakanlah kontroler PI untuk pengaturan kecepatan putar motor DC sedemikian hingga keluaran sistem hasil rancangan mempunyai waktu tunak *ts* (± 5%) sekitar 0.6 detik, *ess* = 0 (*zero offset*), dan tidak memiliki *overshoot*.
2. Gunakan rangkaian motor DC sistem *loop* tertutup.
3. Buka LabVIEW 2013 dan buka *project* "PI Control Motor DC.vi" serta isi *notepad* untuk bagian *file* kontroler PI.
4. Lakukan perhitungan untuk mencari nilai Kp dan Ki berdasar data yang didapatkan dari percobaan identifikasi statis.
5. Masukkan nilai Kp dan τi yang telah dihitung pada kolom Kp dan tau_i seperti yang ditunjukkan pada Gambar 21. Jalankan LabVIEW dan lihat perubahan grafik yang dihasilkan



Gambar 21. Blok Diagram LabVIEW kontroler PI

6. Ubah data hasil percobaan menjadi grafik dengan menggunakan perangkat lunak MATLAB. Lalu lakukan analisa terhadap grafik data percobaan tersebut. Format keluaran dari *notepad* sama seperti pada tabel 2.

Analisa !

Lakukan analisis terhadap Kontroler tipe PI! Bandingkan hasil perhitungan dengan hasil percobaan, apakah hasil percobaan telah sesuai dengan kriteria yang ditentukan! Apa perbedaan Kontroler tipe P dan tipe PI?