SISTEM PENGENDALIAN KECEPATAN ALIRAN UDARA PADA *WIND TUNNEL* DENGAN UMPAN BALIK KECEPATAN ALIRAN UDARA MENGGUNAKAN KONTROLER PID

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI TEKNIK KONTROL

Diajukan untuk Memenuhi Persyaratan
Memperoleh Gelar Sarjana Teknik

Disusun oleh :

RANDY MUHAMMAD

NIM. 105060300111047 - 63

UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2016

LEMBAR PENGESAHAN

SISTEM PENGENDALIAN KECEPATAN ALIRAN UDARA PADA *WIND TUNNEL* DENGAN UMPAN BALIK KECEPATAN ALIRAN UDARA MENGGUNAKAN KONTROLER PID

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI TEKNIK KONTROL

Diajukan Untuk Memenuhi Persyaratan

Memperoleh Gelar Sarjana Teknik

Disusun Oleh:

RANDY MUHAMMAD
NIM. 105060300111047 - 63

Skripsi ini telah direvisi dan disetujui oleh dosen pembimbing

pada tanggal 18 Maret 2016

Dosen Pembimbing I	**Dosen Pembimbing II**
Dr. Ir. Bambang Siswojo, MT.	**Ir. Purwanto, MT**
NIP. 19621211 198802 1 001	**NIP. 19540424 198601 1 001**

Mengetahui,

Ketua Jurusan Teknik Elektro

M. Aziz Muslim, ST., MT., Ph.D

NIP. 19741203 200012 1 001

JUDUL SKRIPSI:

SISTEM PENGENDALIAN KECEPATAN ALIRAN UDARA PADA *WIND TUNNEL* DENGAN UMPAN BALIK KECEPATAN ALIRAN UDARA MENGGUNAKAN KONTROLER PID

Nama Mahasiswa : RANDY MUHAMMAD

NIM : 105060300111047

Program Studi : TEKNIK ELEKTRO

Konsentrasi : TEKNIK KONTROL

Komisi Pembimbing :

Ketua : Dr. Ir. BAMBANG SISWOJO, MT. .....

Anggota : Ir. PURWANTO, MT .....

TIM DOSEN PENGUJI :

Dosen Penguji 1 : Dr. Ir. ERNI YUDANINGTYAS, MT .....

Dosen Penguji 2 : M. AZIZ MUSLIM, ST., MT., Ph.D .....

Dosen Penguji 3 : Ir. MOCH. RUSLI, Dipl.-Ing. .....

Tanggal Ujian : 18 Maret 2016

SK Penguji : No.343/UN10.6/SK/2016

PERNYATAAN ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya dan berdasarkan hasil penelusuran berbagai karya ilmiah, gagasan dan masalah ilmiah yang diteliti dan diulas di dalam Naskah Skripsi ini adalah asli dari pemikiran saya. Tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu Perguruan Tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah Skripsi ini dapat dibuktikan terdapat unsur-unsur jiplakan, saya bersedia Skripsi dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, pasal 25 ayat 2 dan pasal 70).

Malang, 18 Januari 2016

Mahasiswa,

(Materai Rp 6.000,-)

(Tanda Tangan)

Randy Muhammad

NIM. 105060300111047

KATA PENGANTAR

Alhamdulillâh, segala puji hanya bagi Allâh Subhanahu Wa Taála, Rabb alam semesta. Dialah Allâh, Tuhan Yang Maha Satu, Yang Maha Pengasih lagi Maha Penyayang. Dialah Sebaik baik Penolong dan Sebaik baik Pelindung. Shalawat dan salâm kepada Nabi Muhammad Rasulullâh Shallallâhu Alaihi Wa Salâm, Sang pembawa kabar gembira dan sebaik baik suri tauladan bagi yang mengharap Rahmat dan Hidayah-Nya.

Sungguh hanya melalui Pertolongan dan Perlindungan Allâh SWT semata sehingga saya dapat menyelesaikan skripsi ini. Dengan seizin Allâh SWT, di kesempatan yang baik ini saya ingin menghaturkan rasa terima kasih dan penghargaan yang sebesar besarnya atas bantuan sehingga terselesainya skripsi ini kepada:

- Allah SWT yang telah memberikan kelancaran, kemudahan dan hidayah-Nya.
- Keluarga tercinta, kedua orang tua Rakhmad Susatyo dan Retno Setyowati yang selalu memberikan kasih sayang dan doanya yang tiada akhir. Serta kedua saudara Dandy Muhammad dan Pahlevi Muhammad yang selalu memberikan dukungan.
- Bapak M. Aziz Muslim, ST., MT., Ph.D selaku Ketua Jurusan Teknik Elektro Universitas Brawijaya.
- Bapak Hadi Suyono, ST.,MT., Ph.D selaku Sekretaris Jurusan Teknik Elektro Universitas Brawijaya.
- Bapak Dr. Ir. Bambang Siswojo, MT sebagai Dosen Pembimbing I atas segala bimbingan, pengarahan, gagasan, ide, saran, motivasi yang telah diberikan, serta waktu yang diluangkan untuk bimbingan.
- Bapak Ir. Purwanto, MT. selaku KKDK Teknik Kontrol dan sebagai Dosen Pembimbing II atas segala bimbingan, pengarahan, gagasan, ide, saran, motivasi yang telah diberikan, serta waktu yang diluangkan untuk bimbingan.
- Bapak Ibu Dosen, karyawan, staf recording dan RBTE atas segala bantuan dan kemudahan.
- Teman - teman seperjuangan dalam mengerjakan skripsi Ariski, Dandy, Bagus terima kasih telah berbagi pengalaman dan pengetahuan, serta canda dan tawa.
- Teman - teman “MaGiC”, Hanip, Luthfi, Gilang, Rainer, Rangga, Mukson, Dandy, Ulit, Dany, Dugal, Ernanda, Hesta, dan Kadek terima kasih telah berbagi kesenangan, pelajaran hidup, serta canda dan tawa.

- Teman dekat sejak sekolah menengah Joko, Hamdani, Adin, Sesa, dan Afif yang telah meluangkan waktu untuk bercanda dan bercengkerama disaat penat mengerjakan skripsi, serta saudari Saktya Pinastiti yang secara tidak langsung memberi motivasi dan dorongan untuk tetap semangat dalam mengerjakan skripsi.
- Keluarga besar angkatan 2010 MAGNET'10 atas do'a, semangat, serta dukungan yang diberikan pada penulis.
- Semua pihak yang telah memberikan bantuan serta dukungan baik secara langsung maupun tidak langsung atas penyusunan skripsi ini.

Sekiranya Allâh SWT mencatat amalan ikhlas kami dan semua pihak yang turut membantu sehingga skripsi ini terselesaikan. Akhirnya, kami menyadari bahwa skripsi ini masih jauh dari sempurna namun semoga skripsi ini dapat memberikan manfaat bagi kita semua. Allâhumma Amîn.

Malang, Januari 2016

Penulis

DAFTAR ISI

Halaman

LAMPIRAN

DAFTAR GAMBAR

DAFTAR TABEL

RINGKASAN

Randy Muhammad, Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya, Januari 2016, *Sistem Pengendalian Kecepatan Aliran Udara Pada Wind Tunnel dengan Umpan Balik Kecepatan Aliran Udara Menggunakan Kontroler PID,* Dosen Pembimbing: Bambang Siswojo dan Purwanto

Wind tunnel atau terowongan angin adalah alat riset dikembangkan untuk membantu dalam menganalisis efek angin yang bergerak atau di sekitar objek padat. Untuk membangkitkan aliran udara pada *wind tunnel* yaitu dengan cara mengatur putaran *propeller* pada *fan* yang digerakkan menggunakan motor penggerak yang kemudian perubahan aliran udara tersebut dapat dikontrol dengan mengatur putaran *propeller* dengan kecepatan tertentu. Aliran udara tersebut dialirkan menggunakan *propeller* yang terpasang pada sebuah saluran tertutup berbentuk silinder.

Dalam sebuah plant *wind tunnel* kecepatan aliran udara yang dihasilkan tidak selalu sesuai dengan nilai kecepatan yang diinginkan karena adanya gangguan yang menghalangi putaran *propeller.* Sehingga kecepatan aliran udara yang dihasilkan perlu dikendalikan secara elektrik dengan mengatur putaran propeller dengan memasang sebuah sensor *air flow* pada *wind tunnel.*

Solusi dari hal ini yaitu dengan mengendalikan kecepatan putaran *propeller* melalui aktuator motor induksi 3 fasa secara otomatis menggunakan kontroler PID. Diharapkan dengan adanya sistem pengendalian ini, *error* kecepatan aliran udara yang dihasilkan pada *wind tunnel* dapat dikurangi, sehingga kecepatan aliran udara yang dihasilkan dapat sesuai pada nilai yang diinginkan. Dari hasil perancangan dan pengujian alat yang telah dilakukan, didapatkan parameter PID dengan metode *Ziegler-Nichols 1* yang paling baik yaitu Kp=4,67; Ki=7,78; dan Kd=0,7005 dengan *settling time* 6,1 detik.

Kata kunci: *wind tunnel, PID, propeller, air flow, Arduino Mega, motor induksi 3 fasa*

SUMMARY

Randy Muhammad, Department of Electrical Engineering, Faculty of Engineering University of Brawijaya, January 2016, *Air Flow Speed Control System On Wind Tunnel with Feedback Air Flow Speed Using PID Controller*, Academic Supervisor: Bambang Siswojo and Purwanto.

Wind tunnel is a research tool developed to assist in analyzing the effect of wind that moves or around solid objects. To generate airflow in the wind tunnel that is by regulating the rotation on the fan propeller driven using the motor which then changes the air flow can be controlled by adjusting the propeller rotation with a certain speed. The air stream flows using a propeller mounted on a cylindrical closed channels.

Airflow velocity that generated by wind tunnel are not always correspond to the value of the desired speed because of the disruption that prevents rotation propeller. So the speed of air flow being generated electrically controlled by adjusting the propeller rotation by installing a air flow sensor in the wind tunnel.

The solution of this is that by controlling the engine speed using 3 phase induction motor actuator automatically using a PID controller. Expected by this control system, air flow rate error produced at the wind tunnel can be reduced, so that the speed of the airflow generated can match the desired value. From the results of the design and testing tools that have been done, obtained by the PID parameter from Ziegler-Nichols 1 method is Kp = 4.67; Ki = 7.78; and Kd = 0.7005 with a settling time of 6.1 seconds.

Keywords: *wind tunnel, PID, propeller, air flow, Arduino Mega, 3-phase induction motor*

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Pesawat tanpa awak (*Unmanned Aerial Vehicle* atau disingkat UAV), adalah sebuah mesin terbang dapat dikendalikan secara jarak jauh oleh pilot atau mampu mengendalikan dirinya sendiri secara otomatis (*autopilot*). Prinsip terbang menggunakan hukum aerodinamika melalui sayap untuk mengangkat dirinya. Pesawat tanpa awak bisa digunakan untuk membawa muatan baik peralatan elektronik maupun muatan lainnya. Penggunaan UAV biasanya digunakan sebagai penyalur hobi aeromodeling atau untuk mengobservasi lapangan dimana medan yang diobservasi tidak memungkinkan manusia untuk melakukannya, seperti penjelajahan gunung atau laut, pengeboran minyak, pengeksplorasian hasil tambang dan mineral. (Krisnanda, F., 2014)

Udara Aerodinamis yaitu pesawat udara yang lebih berat dari udara. Pesawat Udara Aerodinamis terdiri dari 2 kelompok yaitu pesawat bermotor dan tidak bermotor. Yang bermotor terdiri dari bersayap tetap (*fixed wing*) dan sayap putar (*rotary wing*). Pada penulisan kali ini dipilih untuk melakukan pengukuran gaya aerodinamis pada model *fixed wing*, karena pesawat berjenis ini memiliki platform yang lebar sehingga relatif lebih stabil saat melakukan penerbangan. Pada pesawat bersayap tetap kekuatan pertama dihasilkan oleh aliran udara di permukaan sayapnya yang membentuk sudut datang tertentu dengan flap yakni sayap kecil di belakang sayap yang posisinya ditegakkan sehingga aliran udara mengalir deras ke belakang bisa diarahkan balik ke atas. Pada kondisi gerak planing sering dihadapi masalah ketidakstabilan gerak, maka dari itu perlu diperhatikan faktor-faktor seperti konfigurasi sayap terhadap badan pesawat dan kecepatan aliran udara yang mempengaruhi gerak pesawat maupun gaya-gaya yang timbul lainnya. Tekanan dan kecepatan adalah besaran dasar dalam konsep ilmu aerodinamika. Kedua parameter tersebut menjadi landasan konsep serta aplikasi aerodinamika. Fenomena gerakan fluida yang melewati sebuah benda kerap kali menimbulkan suatu masalah dalam perancangan pada industri yang bergerak dalam bidang aerodinamika.

Untuk mendesain pesawat tanpa awak, didahului dengan mendesain frame pesawat yang terdiri dari badan pesawat (*fuselage*), sayap (*wing*), *stabilizer* horizontal dan vertikal. Untuk mendapatkan karakteristik frame berupa daya angkat diperlukan alat bantu uji berupa terowongan angin (*wind tunnel*). Udara yang mengalir di permukaan sayap bagian bawah menekan permukaan sayap yang relatif datar itu ikut menekan ke atas menimbulkan gaya angkat dan menyebabkan pesawat terangkat ke atas.

Wind tunnel atau terowongan angin adalah alat riset dikembangkan untuk membantu dalam menganalisis efek angin yang bergerak atau di sekitar objek padat. *Wind tunnel* sebagai alat uji aerodinamika terdiri dari beberapa bagian penting yaitu bagian seksi uji (*Test Section*), yang transparan, *honeycomb* sebagai penyearah aliran udara yang masuk kedalam seksi uji, kipas penyedot angin berupa motor penggerak dan *fan*, serta *external balance* sebagai alat ukur gaya yang terjadi pada benda kerja. Untuk membangkitkan aliran udara pada *wind tunnel* yaitu dengan cara mengatur putaran *propeller* pada *fan* yang digerakkan menggunakan motor penggerak yang kemudian perubahan aliran udara tersebut dapat dikontrol dengan mengatur putaran *propeller* dengan kecepatan tertentu. Aliran udara tersebut dialirkan menggunakan *propeller* yang terpasang pada sebuah saluran tertutup berbentuk silinder. Kemudian kecepatan aliran udara yang dihasilkan tadi dapat dideteksi dengan memasang sebuah sensor pada saluran tersebut untuk mengukur kecepatan aliran udara yang melewati *wind tunnel*.

Tidak terprediksinya gangguan yang masuk pada *plant wind tunnel* membuat putaran *propeller* menjadi berputar tidak sesuai pada nilai kecepatan yang diinginkan. Sehingga pada skripsi ini dibuatlah sebuah miniatur *wind tunnel* dengan metode kontrol PID. PID adalah kontroler yang merupakan gabungan dari kontroler proporsional, integral, dan differensial. Kontroler proporsional memiliki kelebihan dalam mempercepat performa sistem, kontroler integral digunakan untuk menghilangkan *offset*, serta kontroler differrensial digunakan untuk mengurangi osilasi. Gabungan dari ketiga kontroler tersebut jika diterapkan pada *wind tunnel* ini diharapkan dapat menghasilkan performa sistem yang cepat, menghilangkan *offset*, dan mengurangi osilasi sehingga keluaran sistem sesuai dengan nilai yang diinginkan.

1.2 Rumusan Masalah

Dari latar belakang di atas dapat diperoleh rumusan masalah sebagai berikut:

1. Bagaimana mendesain model uji kontrol *air flow* untuk menjaga kestabilan putaran *propeller* saat terjadi gangguan berupa aliran udara pada *wind tunnel* berbasis kontrol PID?
2. Bagaimana merancang *hardware* dan *software* alat uji kontrol *air flow* pada *wind tunnel* menggunakan PID?

1.3 Batasan Masalah

Mengacu pada permasalahan pada skripsi ini, maka akan dibatasi pada:

1. Pemodelan alat uji kontrol *air flow* yang dibuat merupakan miniatur terowongan angin.
2. Pembahasan ditekankan pada aplikasi Kontrol PID pada sistem pengontrolan.
3. Arduino Mega sebagai pusat pengendali sistem
4. Gangguan diberikan melalui perubahan aliran udara pada terowongan angin/ *wind tunnel.*
5. Kinerja driver dan rangkaian elektrik tidak dibahas mendalam

1.4 Tujuan

Merancang sistem pengendalian *air flow* untuk menjaga kestabilan putaran *propeller* pada *plant* miniatur *wind tunnel* agar sesuai dengan kecepatan yang diinginkan saat terjadi gangguan berupa aliran udara menggunakan *wind tunnel* berbasis kontroller PID.

1.5 Sistematika Pembahasan

Sistematika penulisan yang digunakan dalam skripsi ini yang terdiri dari enam bab dengan sistematika pembahasan sebagai berikut:

BAB I Pendahuluan

Membahas latar belakang, rumusan masalah, batasan masalah, tujuan dan sistematika pembahasan.

BAB II Tinjauan Pustaka

Membahas teori-teori yang mendukung dalam perancangan dan pembuatan alat.

BAB III Metode Penelitian

Membahas metode penelitian dan perencanaan alat.

BAB IV Perancangan dan Pembuatan Alat

Membahas perancangan alat yang meliputi spesifikasi, perencanaan blok diagram, prinsip kerja, dan pembuatan alat. Setelah itu, bagaimana penerapannya dalam sistem secara keseluruhan.

BAB V Pengujian dan Analisis

Membahas hasil pengujian sistem yang sudah dibuat dan analisis hasil yang diperoleh.

Bab VI Penutup

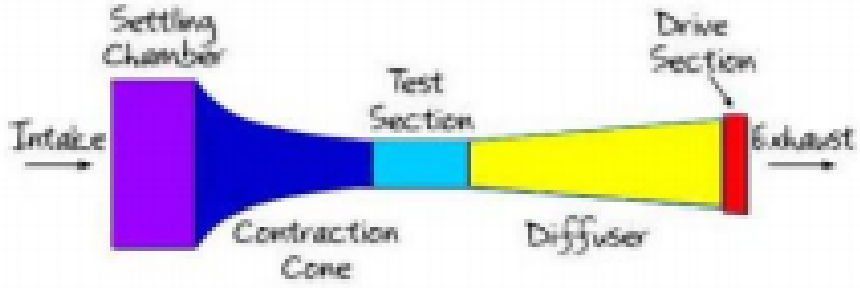
Membahas kesimpulan penelitian dan saran-saran yang diperlukan untuk pengembangan penelitian selanjutnya

BAB II
TINJAUAN PUSTAKA

2.1 Wind Tunnel

Wind tunnel atau terowongan angin adalah alat riset yang dikembangkan untuk membantu dalam menganalisis efek angin yang bergerak atau di sekitar objek padat. Pada umumnya, perancangan terowongan angin berdasarkan dari data-data hasil eksperimen. *Wind tunnel* sebagai alat uji aerodinamika terdiri dari beberapa bagian penting yaitu bagian seksi uji (*Test Section*), yang transparan, *honeycomb* sebagai penyearah aliran udara yang masuk kedalam seksi uji, kipas penyedot angin berupa motor penggerak dan fan, serta external balance sebagai alat ukur gaya yang terjadi pada benda kerja.

Dibagian paling depan terdapat *settling chamber* yang di dalamnya terdapat *honeycomb*. Fungsi *settling chamber* yaitu untuk menyeragamkan aliran udara. Karena aliran turbulen dapat menyebabkan gaya menjadi tidak dapat diperkirakan dan diukur di dalam seksi uji. Di dalam *settling chamber* terdapat *honeycomb*, yang berfungsi untuk mengembangkan atau menghasilkan aliran udara yang halus pada seksi ujinya sehingga aliran udaranya dapat lebih linier. *Honeycomb* ini harus digunakan karena pengaruhnya dalam mengembangkan aliran udara sangat besar. Kemudian udara masuk melalui *contraction cone* yang berfungsi untuk mengambi udara yang memiliki kecepatan tinggi bervolume kecil. Semakin kecil ukuran *contraction cone* semakin tinggi kecepatan udaranya. Setelah itu udara menuju bagian *Test Section* yang berfungsi sebagai tempat untuk menempatkan model sebuah sayap atau pesawat atau benda yang ingin diuji. Saat aliran udara berdasarkan kecepatan yang diinginkan, sensor untuk mengukur gaya dalam hal ini *external balance*, seperti gaya *lift* dan *drag* ditempatkan juga di *test section*. Kemudian oleh *diffuser* laju udara diperlambat sebelum menuju keluaran. Pada *drive section* disediakan gaya yang dapat menyebabkan udara bergerak melewati terowongan Secara umum bentuk model *wind tunnel* ditunjukkan dalam Gambar 2.1.



Gambar 2.1 Model *Wind tunnel Open Circuit*

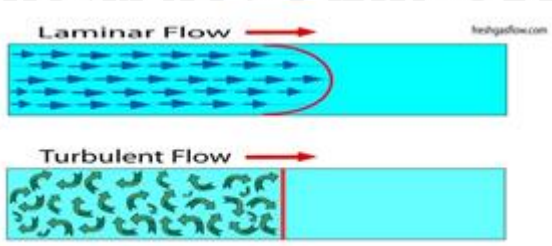
Sumber: Muchammad, 2006

2.2 Tipe Aliran Fluida

Fluida yang bergerak dapat diklasifikasikan ke dalam beberapa kategori. Apakah alirannya steadi atau tak steadi, apakah fluidanya kompresibel (dapat mampat) atau inkompresibel (tak dapat mampat), apakah fluidanya viskos atau non-viskos, atau apakah aliran fluidanya laminar atau turbulen. Jika fluidanya steadi, kecepatan partikel fluida pada setiap titik tetap terhadap waktu. Fluida pada berbagai bagian dapat mengalir dengan laju atau kecepatan yang berbeda, tetapi fluida pada satu lokasi selalu mengalir dengan laju atau kecepatan yang tetap. Aliran fluida dapat dibedakan menjadi aliran laminar dan aliran turbulen, tergantung pada jenis garis alir yang dihasilkan oleh partikel-partikel fluida. Jika aliran dari seluruh partikel fluida bergerak sepanjang garis yang sejajar dengah arah aliran (atau sejajar dengan garis tengah pipa, jika fluida mengalir di dalam pipa), fluida yang seperti ini dikatakan laminar.

Fluida laminar kadang-kadang disebut dengan fluida viskos atau fluida garis alir (streamline). Kata laminar berasal dari bahasa latin lamina, yang berarti lapisan atau plat tipis. Sehingga, aliran laminar berarti aliran yang berlapis-lapis. Lapisan-lapisan fluida akan saling bertindihan satu sama lain tanpa bersilangan seperti dalam Gambar 2.2 (atas).

Jika gerakan partikel fluida tidak lagi sejajar, mulai saling bersilang satu sama lain sehingga terbentuk pusaran di dalam fluida, aliran yang seperti ini disebut dengan aliran turbulen, seperti yang ditunjukkan daam Gambar 2.2 (bawah). (Khamdani, F., 2012)



Gambar 2.2 Aliran laminar (atas) dan aliran turbulen (bawah)

Sumber: Khamdani, F., 2012

2.3 Kontroler

Keberadaan kontroler dalam sebuah sistem kontrol mempunyai kontribusi yang besar terhadap perilaku sistem. Pada prinsipnya hal itu disebabkan oleh tidak dapat diubahnya komponen penyusun sistem tersebut. Artinya, karakteristik *plant* harus diterima sebagaimana adanya, sehingga perubahan perilaku sistem hanya dapat dilakukan melalui penambahan suatu sub sistem, yaitu kontroler.

Salah satu fungsi komponen kontroler adalah mengurangi sinyal kesalahan, yaitu perbedaan antara nilai referensi/nilai yang diinginkan dan nilai aktual. Hal ini sesuai dengan tujuan sistem kontrol dimana mendapat nilai sinyal keluaran sama dengan nilai yang diinginkan referensi. Semakin kecil kesalahan yang terjadi, semakin baik kinerja sistem kontrol yang diterapkan.

Apabila perbedaan antara nilai referensi dengan nilai keluaran relatif besar, maka kontroler yang baik seharusnya mampu mengamati perbedaan ini untuk segera menghasilkan sinyal keluaran untuk mempengaruhi *plant*. Dengan demikian sistem secara cepat mengubah keluaran plant sampai diperoleh selisih dengan nilai referensi sekecil mungkin.

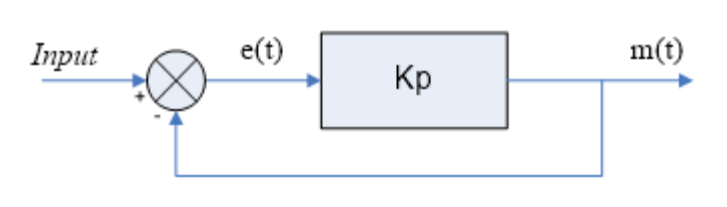
Prinsip kerja kontroler adalah membandingkan nilai aktual keluaran *plant* dengan nilai referensi, kemudian menentukan nilai kesalahan dan akhirnya menghasilkan sinyal kontrol untuk meminimalkan kesalahan (Ogata, K., 1997).

2.3.1 Kontroler Proporsional

Kontroler proporsional memiliki keluaran yang sebanding/proporsional dengan besarnya sinyal kesalahan/*error*. Dapat dikatakan bahwa keluaran kontroler proporsional

merupakan perkalian antara konstanta proporsional dengan masukanya. Perubahan sinyal masukan akan segera menyebabkan sistem secara langsung mengubah keluaranya sebesar konstanta pengalinya.

Untuk lebih jelasnya dapat ditunjukkan pada blok diagram dalam Gambar 2.3.



Gambar 2.3 Diagram blok kontroler proporsional

Sumber: Ogata, K., 1997

Pada Gambar 2.3 menunjukan blok diagram yang menggambarkan hubungan antara *input* (besaran yang diinginkan),besaran aktual dengan besaran keluaran kontroler proporsional,dan besaran kesalahan *(error)*. Sinyal kesalahan *(error)* merupakan selisih antara besaran *setting* dengan besaran aktualnya.

Pada pengendali proporsional hubungan antara keluaran kontroler m(t) dan sinyal kesalahan e(t) adalah sebagai berikut:

$$m(t) = Kp\ e(t) \qquad (2\text{-}1)$$

Dengan Kp adalah penguatan proporsional, keluaran m(t) hanya bergantung pada Kp dan *error*,semakin besar *error* maka semakin besar koreksi yang dilakukan. Penambahan Kp akan menaikan penguatan sistem sehingga dapat digunakan untuk memperbesar kecepatan respon dan mengurangi kesalahan keadaan mantap.

2.3.2 Kontroler Integral

Kontroler integral berfungsi mengurangi kesalahan keadaan mantap yang dihasilkan pada kontroler proporsional sebelumnya. Kalau sebuah *plant* tidak memiliki unsur integrator (1/s), kontroler proporsional tidak akan mampu menjamin keluaran sistem dengan kesalahan keadaan mantapnya nol.

Kontroler integral memiliki karakteristik seperti halnya sebuah integral. Keluaran kontroler sangat dipengaruhi oleh perubahan yang sebanding dengan nilai sinyal

kesalahan. Keluaran kontroler ini merupakan jumlahan yang terus menerus dari perubahan masukannya. Kalau sinyal kesalahan tidak mengalami perubahan, keluaran akan menjaga keadaan seperti sebelum terjadinya perubahan masukan. Gambar 2.4 menunjukkan blok diagram kontroler integral.

Gambar 2.4 Blok diagram kontroler integral

Sumber: Ogata, K., 1997

Nilai keluaran kontroler *m(t)* sebanding dengan integral sinyal kesalahan e(t), Sehingga

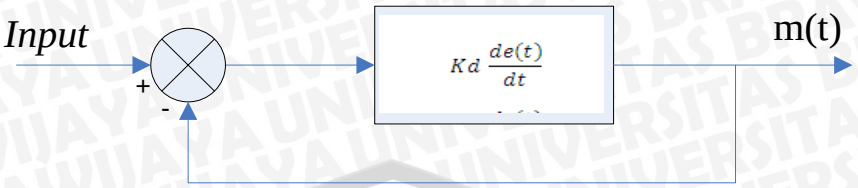
$$\frac{dm(t)}{dt} = Ki.e(t) \qquad (2\text{-}2)$$

$$m(i) = Ki \int_0^t e(t)dt \qquad (2\text{-}3)$$

dengan Ki adalah konstanta integral. Jika sinyal kesalahan e(t)=0, maka laju perubahan sinyal kendali integral $\frac{dm(t)}{dt} = 0$ atau sinyal keluaran kendali akan tetap berada pada nilai yang dicapai sebelumnya. Aksi kontrol integral digunakan untuk menghilangkan kesalahan posisi dalam keadaan mantap (*error steady state*) tanpa memperhitungkan kecepatan respon.

2.3.3 Kontroler Differensial

Kontroler differensial memiliki sifat seperti halnya suatu operasi differensial. Perubahan yang mendadak pada masukan kontroler, akan mengakibatkan perubahan yang sangat besar dan cepat. Gambar 2.5 berikut menunjukkan blok diagram pada kontroler differensial.



Gambar 2.5 Blok diagram kontroler differensial

Sumber: Ogata, K., 1997

Nilai keluaran kontroler m(t) sebanding laju sinyal kesalahan $\frac{de(t)}{dt}$. Hubungan ini dapat ditulis sebagai:

$$m(t) = Kd\ \frac{de(t)}{dt} \qquad (2\text{-}4)$$

Kontroler diferensial akan memberikan sinyal kendali keluaran m(t)= 0, untuk sinyal kesalahan e(t) yang konstan sehingga kontroler diferensial tidak mempengaruhi keadaan mantap. Kontroler diferensial digunakan untuk memperbaiki atau mempercepat respon transien sebuah sistem serta dapat meredam osilasi.

Berdasarkan karakteristik kontroler tersebut, kontroler diferensial umumnya dipakai untuk mempercepat respon awal suatu sistem, tetapi tidak memperkecil kesalahan pada keadaan tunaknya. Kerja kontroler diferensial hanyalah efek dari lingkup yang sempit, yaitu pada periode peralihan. Oleh sebab itu kontroler differensial tidak bisa digunakan tanpa ada kontroler lain.

2.3.4 Kontroler Proporsional Integral (PI)

Aksi kontrolnya dinyatakan dalam persamaan:

$$\text{m(t)} = Kp.e(t) + \frac{Kp}{Ti}\int_0^t e(t)dt \qquad (2\text{-}5)$$

kontroler ini menghasilkan sinyal kesalahan $\int e(t)dt$ kemudian ditambahkan dengan sinyal kesalahan *e(t).*

2.3.5 Kontroler Proportional Differensial (PD)

Aksi kontrolnya dinyatakan dalam persamaan:

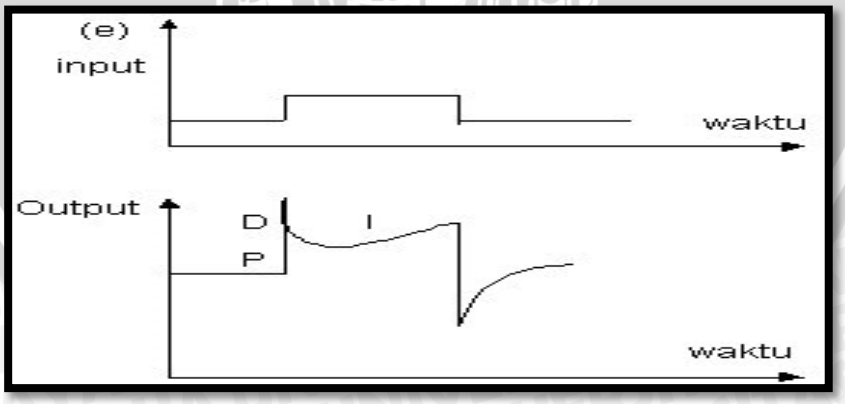
$$\mathrm{m(t)} = \mathrm{Kp.\,e(t)} + Kp.Td\ \frac{de(t)}{dt} \qquad (2\text{-}6)$$

Kontroler PD selalu mengukur kemiringan (*slope*) sinyal kesalahan $\frac{de(t)}{dt}$ dan memperkirakan akan besar *overshoot* yang akan terjadi serta memberikan koreksi sebelum terjadi lewatan sebenarnya sehingga diperoleh *maximum overshoot* yang kecil.

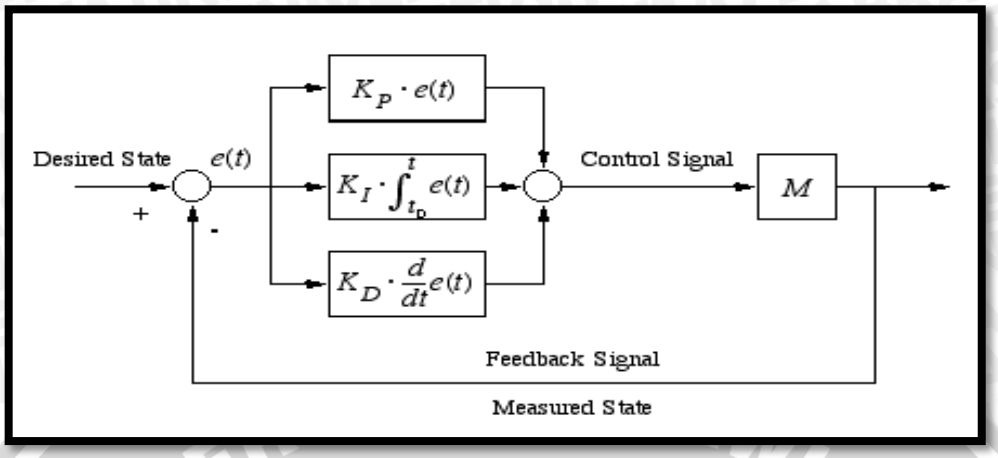
Jika kesalahan keadaan mantap tidak berubah terhadap waktu maka turunanya terhadap waktu sama dengan nol, sehingga kontroler PD tidak mempunyai pengaruh terhadap kesalahan keadaan mantap,tetapi jika terdapat perubahan kesalahan, kontroler PD akan mengurangi besar kesalahan keadaan mantap. Jadi kontrol PD digunakan untuk memperbaiki suatu sistem pengendalian yang tanggapan peralihanya mempunyai *maximum overshoot* yang berlebihan tanpa memperhitungkan kecepatan responnya.

2.3.6 Kontroler Proportional Integral Differensial (PID)

Setiap kekurangan dan kelebihan masing-masing kontroler P, I dan D dapat saling menutupi dengan menggabungkan ketiganya secara paralel menjadi kontroler proporsional integral differensial (PID). Elemen-elemen P, I, dan D masing-masing secara keseluruhan bertujuan untuk mempercepat reaksi sebuah sistem, menghilangkan *offset* dan menghasilkan perubahan awal yang besar (Gunterus, 1994, 8-10). Kontroler PID memiliki diagram kendali seperti yang ditunjukan dalam Gambar 2.6.



Gambar 2.6 Hubungan fungsi waktu antara sinyal keluaran dan masukan kontroler PID



Gambar 2.7 Diagram Blok Kontroler PID

Sumber: Ogata, K., 1997

Aksi kontrolnya dapat dinyatakan sebagai berikut:

$$m(t) = K_p e(t) + \frac{Kp}{Ti} \int_0^t e(t)dt + K_p T_d \frac{de(t)}{dt} \qquad (2\text{-}7)$$

Jenis kontroler ini digunakan untuk memperbaiki kecepatan respon dan mencegah terjadinya kesalahan keadaan mantap serta mempertahankan kestabilan.

Keluaran kontroler PID merupakan penjumlahan dari keluaran kontroler proporsional, integral dan differensial. Gambar 2.7 menunjukan hubungan tersebut. Karakteristik kontroler PID sangat dipengaruhi oleh kontribusi besar dari ketiga parameter P, I, dan D. Penyetelan konstanta Kp, Ki, dan Kd akan mengakibatkan penonjolan sifat dari masing-masing elemen. Satu atau dua dari ketiga konstanta tersebut dapat disetel lebih menonjol dibandingkan yang lain. Konstanta yang menonjol itulah yang akan memberikan kontribusi pengaruh pada respon sistem secara keseluruhan (Gunterus, 1994, 8-10).

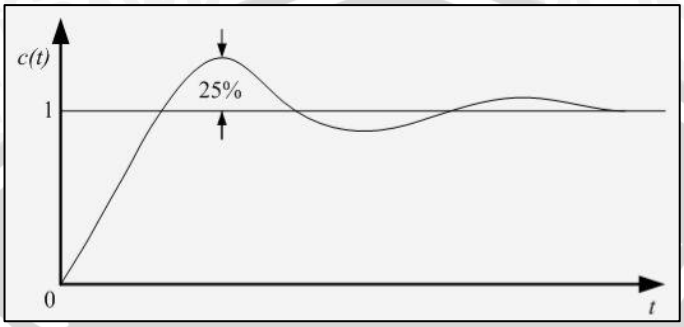
2.4 Metode *Tuning* PID

Metode Ziegler-Nichols.

Ziegler dan Nichols mengemukakan aturan-aturan untuk menentukan nilai dari gain proporsional Kp, waktu integral Ti, dan waktu derivatif Td berdasarkan karakteristik respon transien dari *plant* yang diberikan. Penentuan parameter kontroler PID atau

penalaan kontroler PID tersebut dapat dilakukan dengan bereksperimen dengan plan.(Ogata, K., 1997)

Terdapat dua metode yang disebut dengan aturan penalaan Ziegler-Nichols, pada kedua metode tersebut memiliki tujuan yang sama yaitu untuk mencapai 25% *maximum overshoot* pada respon unit step, seperti ditunjukkan dalam Gambar 2.8.

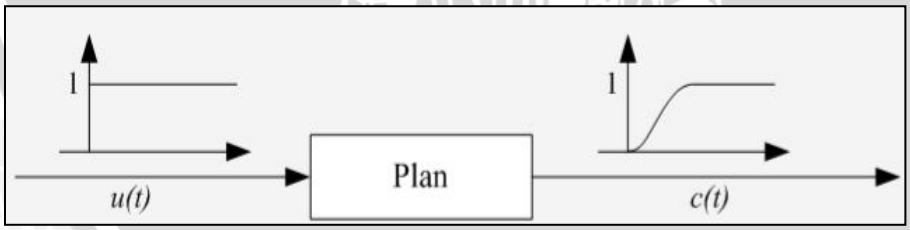


Gambar 2.8 Kurva Respon *Unit Step* yang Menunjukkan 25% *Maximum Overshoot*

Sumber: Ogata, K., 1997

a) Metode Pertama

Metode pertama atau sering disebut metode kurva reaksi, respon dari *plan* dapat dapat diperoleh secara eksperimental dengan masukan berupa unit step, seperti yang ditunjukkan dalam Gambar 2.9.

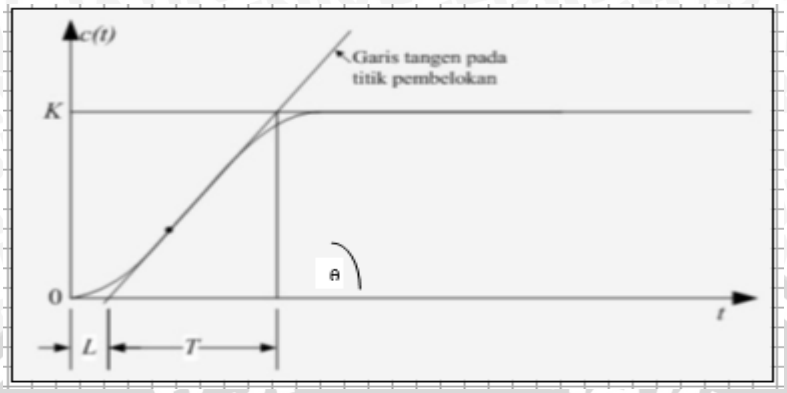


Gambar 2.9 Respons Plan Terhadap Masukan Berupa *Unit Step*

Sumber: Ogata, K., 1997

Jika dalam plan tersebut terdapat integrator atau *dominan complex-conjugate poles*, maka kurva respons unit *step* berbentuk seperti huruf S, seperti dalam Gambar 2.10.

Jika respons tidak memberikan bentuk kurva S, maka metode ini tidak berlaku. (Ogata, K., 1997).



Gambar 2.10 Respons Plan berbentuk S

Sumber: Ogata, K., 1997

Kurva berbentuk S tersebut dapat dikarakteristikkan menjadi dua konstanta yaitu waktu tunda *L* dan konstanta waktu *T*. Waktu tunda dan konstanta waktu ditentukan dengan menggambar sebuah garis tangen pada titik pembelokan dari kurva S, dan menentukan perpotongan antara garis tangen dengan sumbu waktu *t* dan sumbu *c(t) =K*, Fungsi alih *C(s)/U(s)* dapat dilakukan pendekatan dengan sistem orde satu dengan persamaan sebagai berikut:

$$\frac{C(s)}{U(s)} = \frac{Ke^{-Ls}}{Ts+1} \qquad (2\text{-}8)$$

Ziegler dan Nichols menyarankan untuk menentukan nilai-nilai dari Kp, Ti dan Td berdasarkan pada formula yang ditunjukkan dalam Tabel 2.1 (Ogata, K., 1997)

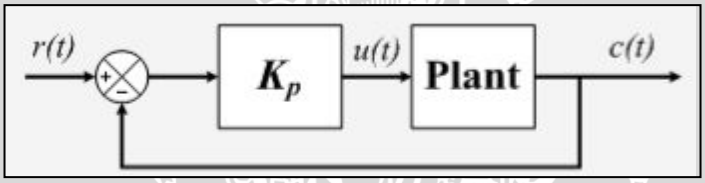
Tabel 2.1 Aturan Penalaran *Ziegler-Nichols* Berdasarkan Respons Unit Step Dari Plan

Tipe Kontroler	K_p	T_i	Td
P	$\frac{T}{L}$	∞	0
PI	$0{,}9\frac{T}{L}$	$\frac{L}{0{,}3}$	0
PID	$1{,}2\,\frac{T}{L}$	$2L$	$0{,}5\,L$

Sumber: Ogata, K., 1997

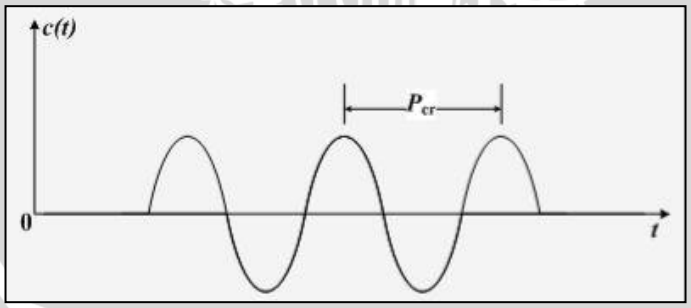
b) Metode Kedua

Dalam metode kedua *Ziegler-Nichols*, mula-mula yang dilakukan adalah membuat Ti = 0 dan Td = 0. Kemudian hanya dengan menggunakan tindakan kontrol proporsional, harga ditingkatkan dari nol ke suatu nilai kritis *K*cr disini mula-mula keluaran memiliki osilasi yang berkesinambungan (Jika keluaran tidak memiliki osilasi berkesinambungan untuk nilai Kp manapun yang telah diambil, maka metode ini tidak berlaku). Dari keluaran yang berosilasi secara berkesinambungan, penguatan kritis *K*cr dan periode *P*cr dapat ditentukan. Diagram blok sistem loop tertutup dengan kontroler proporsional dapat dilihat dalam Gambar 2.11 dan untuk osilasi berkesinambungan dengan periode Pcr dapat dilihat dalam Gambar 2.12 Ziegler dan Nichols menyarankan penyetelan nilai parameter Kp,Ti ,Td dan berdasarkan rumus yang diperlihatkan dalam Tabel 2.2. (Ogata, K., 1997)



Gambar 2.11 Sistem *Loop* Tertutup dengan Kontroler Proporsional

Sumber: Ogata, K., 1997



Gambar 2.12 Osilasi Berkesinambungan dengan periode *Pcr*

Sumber: Ogata, K., 1997

Tabel 2.2 Aturan Dasar Ziegler-Nichols Berdasarkan *Critical GainKcr* dan *Critical PeriodPcr*

Tipe Kontroler	K_p	T_i	Td
P	0.5 Kcr	∞	0
PI	0.45 Kcr	$\frac{1}{1,2}$ Pcr	0
PID	0.60 Kcr	0.5 Pcr	0.125 Pcr

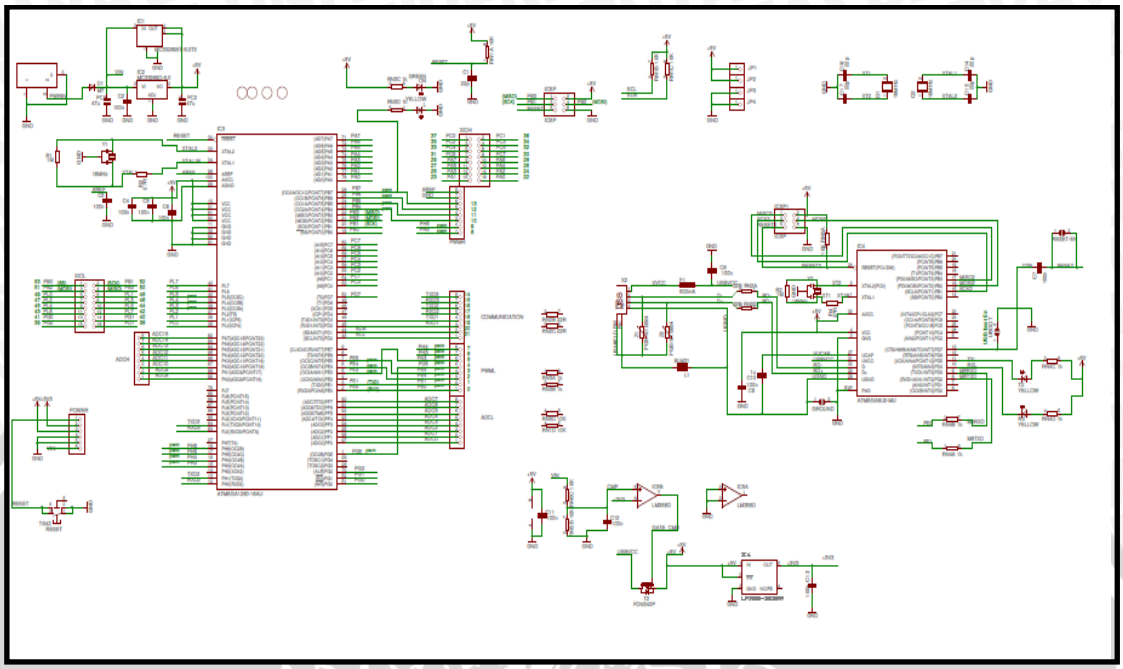
Sumber: Ogata, K., 1997

2.5 Mikrokontroler Arduino mega 2560

Arduino Mega2560 adalah *board* mikrokontroler berbasis ATmega2560. Arduino Mega memiliki 54 pin Input/Output digital (dimana 15 pin diantaranya dapat digunakan sebagai output PWM), 16 input analog, 4 UARTs (port serial perangkat keras), 16 MHz osilator Kristal, koneksi USB, *power jack*, ICSP *header*, dan tombol reset. Untuk mendukung mikrokontroler agar dapat digunakan, cukup hanya menghubungkan Arduino Mega2560 ke komputer dengan menggunakan kabel USB atau dengan memberi daya dengan adaptor AC-*to*-DC atau baterai agar dapat bekerja. Arduino Mega2560 juga cocok menggunakan sebagian besar *shield* yang didesain untuk Arduino Duemilanove.

Arduino Mega2560 berbeda dengan *board* Arduino sebelumnya dimana *board* sebelumnya menggunakan FTDI USB-*to*-serial *driver chip*. Pada Arduino Mega2560 terdapat ATmega16U2 (sebagai ganti dari ATmega8U2 pada board sebelumnya) yang deprogram sebagai USB-*to*-serial *converter*. Arduino Mega2560 juga memiliki RESET sirkuit yang lebih kuat dibandingkan *board* sebelumnya. Fitur baru yang terdapat pada Arduino Mega2560 yaitu penambahan pin SDA dan SCL yang berdekatan dengan pin AREF dan dua pin baru lainnya diletakkan dekat pin RESET, pin pertama yaitu IOREF yang memperbolehkan shield untuk beradaptasi dengan tegangan yang disediakan oleh *board*. Kedepannya, *shield* akan bekerja sangat baik dengan *board* yang menggunakan AVR yang dapat beroperasi dengan 5V dan dengan 3,3V (seperti pada Arduino Duemilanove). Pin kedua adalah pin yang tidak terhubung yang disediakan untuk tujuan

yang akan datang. Gambar 2.13 menunjukkan konfigurasi kaki I/O dari Arduino Mega 2560.



Gambar 2.13 Desain sistem Arduino Mega2560-R3(arduino.cc)

2.5.1 Daya

Arduino Mega2560 dapat diaktifkan melalui koneksi USB atau dengan menggunakan catu daya eksternal (otomatis). Daya eksternal (non-USB) dapat berasal dari adaptor AC-*to*-DC atau baterai. Adaptor dapat dihubungkan dengan menancapkan plug jack dengan pusat-positif sebesar 2,1mm pada *Power jack* pada *board*. Untuk baterai, ujung kepala baterai dapat dimasukkan pada pin Gnd dan Vin pada header di bagian konektor POWER. Board Arduino Mega2560 ini dapat beroperasi dengan catu daya eksternal antara 6 sampai dengan 20 volt. Jika diberi daya kurang dari 7V, kemungkinan pin 5V tetap dapat beroperasi tetapi tidak stabil. Jika menggunakan daya lebih dari 12V, maka regulator tegangan akan panas dan dapat merusak *board* Arduino Mega2560. *Range* daya yang direkomendasikan antara 7 sampai dengan 20 volt.

Pin POWER pada *board* Arduino Mega2560 diantaranya adalah:

- **VIN**. Tegangan masukan untuk *board* Arduino ketika menggunakan catu daya eksternal (berbeda dengan 5V yang berasal dari konektror USB atau sumber tegangan yang telah disesuaikan).
- **5V**. Pin output ini mengeluarkan output sebesar 5V yang telah disesuaikan menggunakn regulator yang berasal dari *board* Arduino. *Board* Arduino dapat dicatu dengan daya yang berasal dari *power jack* DC (7-12V), konektor USB (5V), atau pin VIN yang terdapat pada *board* (7-12V). Mencatu daya pada pin 5V dan 3,3V akan merusak regulator dan *board* Arduino.
- **3,3V**. Merupakan catu daya sebesar 3,3V yang dihasilkan oleh regulator pada *board* Arduino.
- **GND**. Merupakan pin *ground*.
- **IOREF**. Pada board *Arduino*, pin ini menyediakan tegangan referensi yang dioperasikan oleh mikrokontroler. *Shield* yang telah dikofigurasi dengan baik dapat membaca tegangan pin IOREF dan dapat memilih catu daya yang sesuai atau dapat mengaktifkan tegangan translasi pada output yang bekerja pada 5V atau 3,3V.

2.5.2 Memori

ATmega2560 memiliki *flash memory* sebesar 256KB untuk penyimpanan kode (dimana 8KB digunakan sebagai bootloader), 8KB untuk SRAM dan 4KB untuk EEPROM (yang dapat dibaca dan ditulis dengan *library* EEPROM).

2.5.3 *Input* dan *Output*

Setiap pin dari 54 pin digital pada Arduino Mega2560 dapat digunakan sebagai input dan output menggunakan fungsi *pinMode(), digitalWrite(),* dan *digitalRead()*. Setiap pin beroperasi pada tegangan 5V. Masing-masing pin dapat menyediakan atau menerima arus maksimal sebesar 40 mA dan memiliki resistor *pull-up* internal dengan 20-50k Ω. Selain itu, beberapa pin memiliki fungsi special seperti:

- **Serial 0: 0 (RX) dan 1 (TX); Serial 1: 19 (RX) dan 18 (TX); Serial 2: 17 (RX) dan 16 (TX); Serial 3: 15 (RX) dan 14 (TX).** Pin (RX) digunakan untuk menerima dan pin (TX) untuk mentransmisikan data TTL serial. Pin 0 dan 1 juga terhubung dengan pin koresponden dari ATmega16U2 USB-to-TTL serial chip.

- **Eksternal Interrupt: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), dan 21 (interrupt 2).** Pin tersebut dapat dikonfigurasikan untuk memicu interrupt pada kondisi *LOW*, tepi naik atau turun, dan pada kondisi *HIGH*. Untuk lebih jelasnya dapat dilihat pada fungsi *attachInterrupt()*.
- **PWM: 2-13 dan 44-46**. Menyediakan *output* PWM 8-bit dengan fungsi *analogWrite()*.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS)**. Pin tersebut mendukung komunikasi SPI dengan menggunakan *library* SPI. Pin SPI juga terdapat pada ICSP header yang secara fisik dapat bekerja dengan baik untuk Uno, Duemilanove, Diecimila.
- **LED: 13**. Merupakan LED yang terpasang pada board dan terhubung dengan pin *digital* 13. LED akan menyala saat nilai *HIGH* dan akan mati ketika pin bernilai *LOW*.
- **TWI: 20 (SDA) dan 21 (SCL)**. Mendukung komunikasi TWI menggunakan *Wire library*. Pin tersebut tidak terletak pada posisi yang sama dengan pin TWI pada Duemilanove atau Diecimila.

Arduino Mega2560 memiliki 16 analog input yang masing-masing menyediakan resolusi 10bit (1024 yang memiliki nilai berbeda). Pada dasarnya nilai yang terukur dari *ground* hingga 5V, yang titik tertingginya dapat diubah menggunakan pin AREF dan fungsi *analogReference()*.

Berikut adalah beberapa pin *analog* yang terdapat pada board:

- **AREF.** Merupakan tegangan referensi untuk *input analog*. digunakan dengan *analogReference()*.
- **Reset.** Membawa garis *LOW* untuk me-reset mikrokontroler. Pada umumnya digunakan untuk menambahkan tombol reset untuk membatasi tombol reset yang berada pada *board*.

2.5.4 Komunikasi

Arduino Mega2560 memiliki beberapa fasilitas untuk berkomunkasi dengan komputer, Arduino yang lain, atau mikrokontroler lain. ATmega2560 menyediakan empat perangkat UART untuk komunikasi *serial* TTL (5V). ATmega16U2 pada *board* merupakan saluran untuk USB dan menyediakan *port* com virtual untuk *software* pada

komputer. Pada Windows diperlukan sebuah file dengan tipe .inf; tetapi tidak pada OSX atau Linux, kedua *Operating System* ini akan menidentifikasi atau mengenali secara otomatis *board* Arduino sebagai *port* COM. *Software* Arduino menyediakan serial monitor yang memperbolehkan data tekstual untuk dikirimkan baik ke *board* ataupun dari *board*. LED RX dan TX pada *board* akan berkedip saat data dikirimkan melalui chip ATmega8U2/ ATmega16U2 dan koneksi USB ke konektor (namun tidak untuk komunikasi serial pada pin 0 dan 1).

Software Serial library memperbolehkan komunikasi serial antara beberapa pin *digital* pada Arduino Mega2560. ATmega2560 juga mendukung komunikasi TWI dan SPI. *Software* Arduino menyediakan *Wire library* untuk memudahkan bus TWI dan komunikasi SPI.

2.6 Sensor *Air Flow*

Pengukuran aliran mulai dikenal sejak tahun 1732 ketika Henry Pitot mengatur jumlah fluida yang mengalir. Dalam pengukuran fluida perlu ditentukan besaran dan vektor kecepatan aliran pada suatu titik dalam fluida dan bagaimana fluida tersebut berubah dari titik ke titik.

Jenis pengukur aliran yang paling luas digunakan adalah pengukuran tekanan diferensial. Pada prinsipnya beda luas penampang melintang dari aliran dikurangi dengan yang mengakibatkan naiknya kecepatan, sehingga menaikan pula energi gerakan atau energi kinetis. Karena energi tidak bisa diciptakan atau dihilangkan (Hukum perpindahan energi), maka kenaikan energi kinetis ini diperoleh dari energi tekanan yang berubah. Lebih jelasnya, apabila fluida bergerak melewati penghantar (pipa) yang seragam dengan kecepatan rendah, maka gerakan partikel masing-masing umumnya sejajar disepanjang garis dinding pipa. Kalau laju aliran meningkat, titik puncak dicapai apabila gerakan partikel menjadi lebih acak dan kompleks. Kecepatan kira-kira di mana perubahan ini terjadi dinamakan kecepatan kritis dan aliran pada tingkat kelajuan yang lebih tinggi dinamakan turbulen dan pada tingkat kelajuan lebih rendah dinamakan laminer.

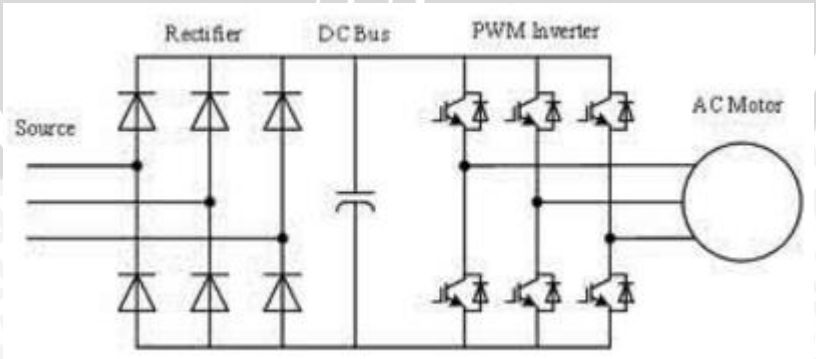
Pengukuran aliran metoda ini dapat dilakukan dengan banyak cara misalnya: menggunakan pipa *venturi*, pipa pitot, *orifice* plat (lubang sempit), *turbine flow meter*,

rotameter, cara thermal, menggunakan bahan radio aktif, elektromagnetik, *ultra sonic* dan *flowmeter gyro*. Cara lain dapat dikembangkan sendiri sesuai dengan kebutuhan proses.

2.7 *Variable Frequency Drive*

Inverter adalah suatu peralatan elektronika daya yang berfungsi untuk mengubah listrik DC menjadi AC. *Inverter* seringkali disebut sebagai *Variable Frequency Drive* (VFD).

Prinsip kerja inverter adalah mengubah input motor (listrik AC) menjadi DC dan kemudian dijadikan AC lagi dengan frekuensi yang dikehendaki sehingga motor dapat dikontrol sesuai dengan kecepatan yang diinginkan. Untuk mengubah tegangan AC menjadi DC dibutuhkan penyearah (*converter* AC-DC) dan biasanya menggunakan penyearah tidak terkendali (*rectifier dioda*) namun juga ada yang menggunakan penyearah terkendali *(thyristor rectifier*). Setelah tegangan sudah diubah menjadi DC maka diperlukan perbaikan kualitas tegangan DC dengan menggunakan tandon kapasitor sebagai perata tegangan. Kemudian tegangan DC diubah menjadi tegangan AC kembali oleh inverter dengan teknik PWM (*Pulse Width Modulation*). Dengan teknik PWM ini bisa didapatkan amplitudo dan frekuensi keluaran yang diinginkan. Selain itu teknik PWM juga menghasilkan harmonisa yang jauh lebih kecil dari pada teknik yang lain serta menghasilkan gelombang sinusoidal, dimana kita tahu kalau harmonisa ini akan menimbulkan rugi-rugi pada motor yaitu cepat panas. Maka dari itu teknik PWM inilah yang biasanya dipakai dalam mengubah tegangan DC menjadi AC *(Inverter*). Skema rangkaian VFD ditunjukkan dalam gambar 2.14.



Gambar 2.14 Skema rangkaian *Variable Frequency Drive*

BAB III
METODE PENELITIAN

Kajian dalam skripsi ini merupakan penelitian yang bersifat aplikasi, yaitu dengan merancang suatu pengendalian menggunakan kontroler P, I, dan D yang bertujuan dapat mendapatkan performansi sistem yang diharapkan.

Langkah – langkah yang perlu dilakukan untuk merealisasikan alat yang akan dibuat adalah sebagai berikut:

1. Studi Literatur
2. Spesifikasi alat
3. Perancangan dan realisasi pembuatan alat
4. Pengujian alat
5. Pengambilan kesimpulan

3.1 Spesifikasi Alat

Spesifikasi alat secara umum ditetapkan terlebih dahulu sebagai acuan dalam perancangan selanjutnya. Spesifikasi alat yang direncanakan adalah sebagai berikut :

- Desain alat berbentuk persegi dengan ukuran 150x150cm. Pada bagian tengah terpasang sebuah *propeller* dengan diameter 147 cm.
- Menggunakan satu buah motor induksi 3 fasa sebagai penggerak.
- Menggunakan sensor *air flow*.
- Menggunakan *variable frequency drive* sebagai inverter.
- Mikrokontroler yang digunakan Arduino Mega 2560.
- Kontroler yang digunakan adalah kontroler PID.

3.2 Studi Literatur

Studi literatur dilakukan untuk mendapatkan pengetahuan dasar tentang segala sesuatu yang mendukung perancangan serta pembuatan sistem pengendalian kecepatan

aliran udara pada *wind tunnel* dengan umpan balik kecepatan aliran udara menggunakan kontroler PID.

Data-data yang dibutuhkan dalam pembuatan alat ini diambil dari buku, jurnal, artikel laporan penelitian dan situs-situs di internet untuk mengetahui karakteristik komponen, prinsip kerja serta teori yang menunjang, antara lain :

- Hal-hal yang berhubungan dengan proses kerja sistem pengendalian kecepatan aliran udara pada *wind tunnel*
- Karakteristik sensor-sensor yang digunakan, yaitu sensor aliran udara dan putaran motor
- Prinsip kerja kontroler P, I dan D
- Informasi tentang Arduino Mega 2560

3.3 Realisasi Pembuatan Sistem

3.3.1 Perancangan Perangkat Keras dan Realisasi Pembuatan Alat

a. Pembuatan diagram blok
b. Penentuan dan Perhitungan komponen yang akan digunakan dalam perancangan alat
c. Merakit perangkat keras (*hardware*) untuk masing-masing blok.

3.3.2 Perancangan dan Perhitungan Komponen yang akan Digunakan

Setelah merancang perangkat keras, maka langkah selanjutnya adalah merancang perangkat lunak guna mengendalikan dan mengatur kerja daripada alat. Desain dan parameter yang telah dirancang kemudian diterapkan pada Arduino Uno dengan menggunakan software Arduino ERW 1.0.5.

3.3.3 Perancangan Perangkat Lunak

Perancangan perangkat lunak dilakukan setelah mengetahui nilai parameter Proporsional (P), Integral (I) dan Deferensial (D). Perancangan dimulai dari pembuatan *flowchart,* kemudian penulisan *listing code*.

3.4 Pengujian dan Analisis Data

Setelah semua komponen pada alat sudah terhubung sesuai dengan diagram blok sistem yang telah dirancang dan program *software* sudah dibuat, maka diadakan pengujian

dan analisis alat. Performansi sistem yang diinginkan dari pengendalian suhu dan kelembaban ruang ini adalah respon sistem tanpa *overshoot* dengan *error steady state* seminimal mungkin.

Pengujian dan analisis yang dilakukan adalah sebagai berikut.

1. Pengujian setiap blok rangkaian.
2. Penggabungan semua blok rangkaian menjadi sebuah sistem.
3. Pengujian alat secara keseluruhan.
4. Evaluasi dan analisis pengujian sistem yang didapat.

3.5 Pengambilan Kesimpulan

Kesimpulan serta saran dapat diambil berdasarkan data yang telah didapat dari hasil pengujian sistem secara keseluruhan. Apabila hasil yang telah didapatkan sesuai dengan yang direncanakan sebelumnya, maka sistem kendali tersebut telah berhasil memenuhi harapan dan dapat dikembangkan untuk penelitian selanjutnya untuk disempurnakan.

BAB IV
PERANCANGAN DAN PEMBUATAN ALAT

Dalam bab ini membahas mengenai perancangan dan pembuatan sistem pengendali kecepatan putaran *propeller* pada *wind tunnel* dengan *feedback* kecepatan aliran udara menggunakan kontroler PID. Perancangan perangkat tersebut meliputi perancangan perangkat keras maupun perancangan perangkat lunak. Sedangkan pembuatan bertujuan untuk menghasilkan semua perangkat pendukung maupun alat secara keseluruhan.

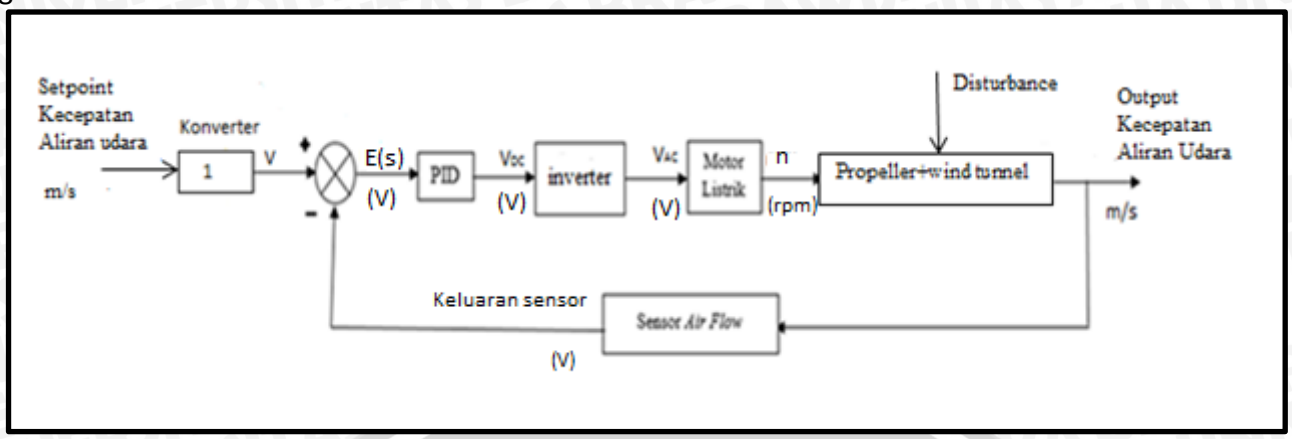
4.1 Perancangan Sistem

Perancangan alat ini dilakukan bertahap dalam bentuk diagram blok sehingga memudahkan dalam analisis pada setiap bloknya maupun secara keseluruhan sistem. Perancangan ini terdiri atas:

1. Perancangan perangkat keras *propeller* (baling-baling) dan motor induksi 3 fasa sebagai penggerak aktuator.
2. Perancangan perangkat lunak (perancangan algoritma kontrol PID pada software arduino mega).

4.2 Diagram Blok Sistem

Dalam skripsi ini dibuat diagram blok agar dalam pengerjaan dapat dilakukan sesuai dengan rancangan sistem. Adapun diagram blok tersebut dapat dilihat pada Gambar 4.1:



Gambar 4. 1 Diagram Blok Sistem

Keterangan dari diagram blok dalam Gambar 4.1:

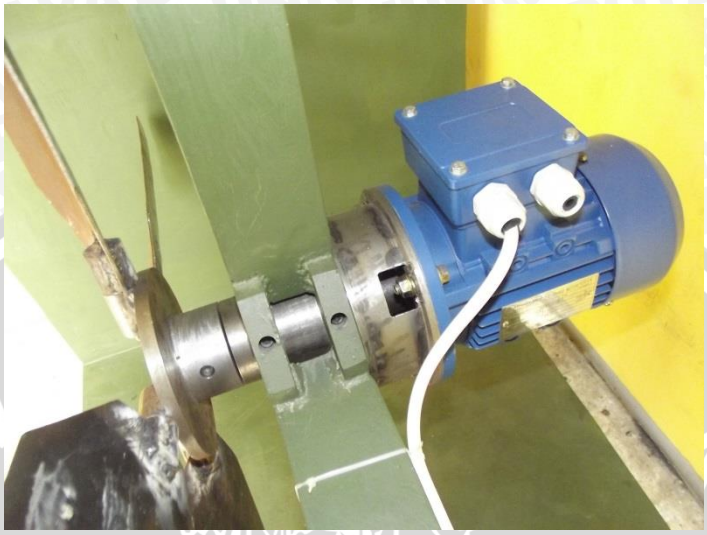
- Masukan / *setpoint* berupa kecepatan putaran diberikan melalui *program* pada Arduino Mega.
- Kemudian input diolah dan menghasilkan sinyal kontrol berupa PWM yang kemudian akan menjadi masukan untuk menggerakkan motor induksi 3 fasa.
- Sinyal dari motor induksi 3 fasa tadi kemudian menggerakkan *propeller* pada *wind tunnel* sehingga mengatur putaran sesuai *setpoint*.
- Keluaran putaran berupa aliran udara yang dihasilkan oleh putaran *propeller* tadi dibaca oleh sensor *air flow* yang kemudian diproses hingga mendapatkan keluaran dalam bentuk digital.
- Keluaran dari mekanik berupa aliran udara kemudian dibaca oleh sensor *air flow*. Keluaran dari sensor yang berupa analog kemudian di ubah menjadi digital oleh *converter* ADC.
- Hasil akhir pembacaan sensor kemudian di kurangkan dengan *input* / *setpoint* sehingga mikrokontroler mampu mengkompensasi *error* yang terjadi.

4.3 Perancangan Perangkat Keras

4.3.1 *Propeller*

Propeller yang digunakan adalah *propeller* dengan diameter 147 cm. Pemilihan ini didasarkan pada ukuran *wind tunnel* yaitu 150x150 cm. *Propeller* digunakan untuk menghasilkan aliran udara pada alat uji ini sehingga didapatkan parameter-parameter yang

akan diukur. Gambar *propeller* diperlihatkan dalam gambar 4.2. Sedangkan model mekanik alat ditunjukkan dalam gambar 4.3



Gambar 4.2 *Propeller*

Gambar 4.3 Model Alat

4.3.2 Sensor *Air Flow*

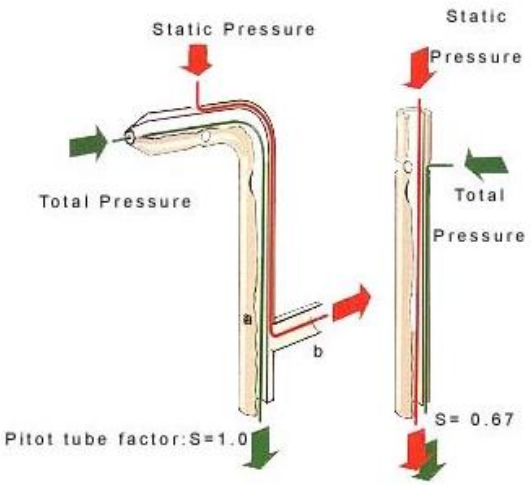
Air flow sensor adalah perangkat yang mengukur aliran udara, yaitu berapa banyak udara mengalir melalui tabung. Ini tidak mengukur volume udara yang lewat melalui

tabung, mengukur kecepatan yang sebenarnya dari udara yang mengalir melalui perangkat dalam segmen waktu yang ditetapkan. Dengan demikian sensor aliran udara ini hanya sebuah aplikasi pengukuran aliran pada suatu media khusus.

Tekanan total diukur dengan menggunakan pipa bagian dalam dari tabung *pitot* dan tekanan statis diukur dengan menggunakan pipa luar dari tabung *pitot*. Ujung tabung luar dan dalam disambungkan ke mikrokontroler. Tekanan kecepatan (yaitu perbedaan antara tekanan total dan tekanan statis), dikonversi menjadi tegangan masukan bagi sensor. Pipa yang mengukur tekanan statis terletak secara radial pada batang yang dihubungkan ke mikrokontoler (pstat). Tekanan pada ujung pipa di mana fluida masuk merupakan tekanan stagnasi(p0). Gambar diagram aliran udara yang masuk pada pipa ditunjukkan dalam Gambar 4.4.

$$p_o = p_{stat} + \frac{1}{2}\rho V^2 \qquad 4\text{-}1$$

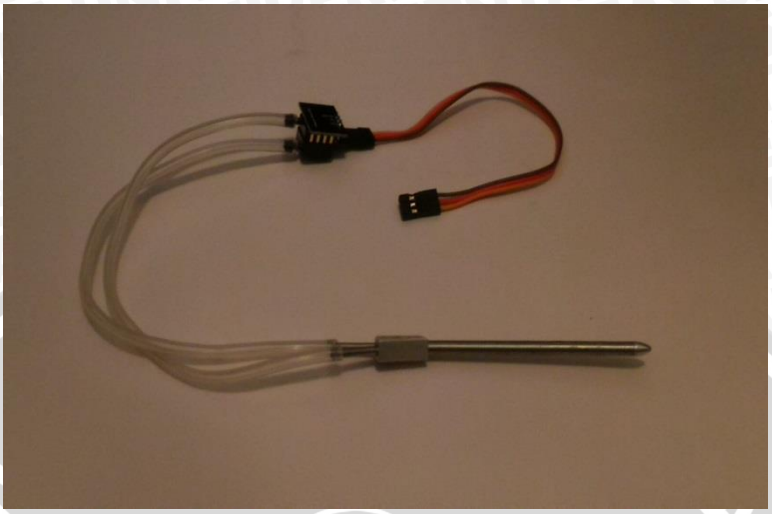
$$V = \sqrt{2(p_0 - p_{stat})/\rho} \qquad 4\text{-}2$$



Gambar 4.4 Diagram aliran udara pada tabung pitot

Sumber: https://yefrichan.wordpress.com/2010/08/02/cara-menghitung-daya-blowerfan/

Sensor yang digunakan dalam pengujian ini ditunjukkan dalam gambar 4.5.



Gambar 4.5 Sensor *Air Flow*

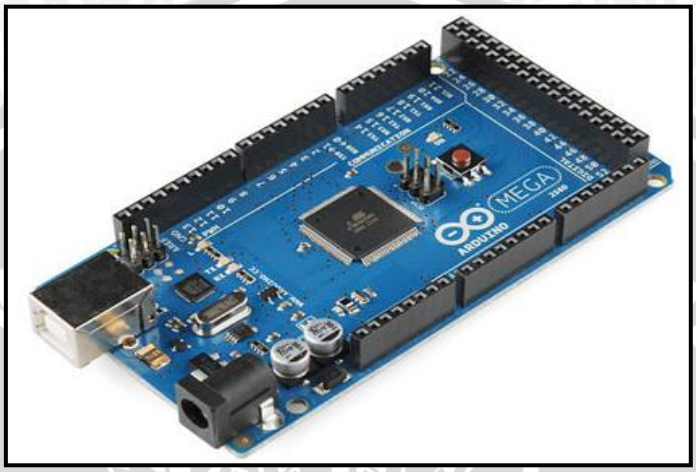
4.3.3 Motor Induksi 3 Fasa

Motor induksi 3 fasa yang digunakan dalam perancangan kali ini berguna sebagai penggerak baling – baling (*propeller*). Motor induksi 3 fasa ini juga dapat langsung terhubung ke Arduino mega tanpa menggunakan driver karena bekerja pada maksimum tegangan masukan 4,8 V dan dengan putaran yang konstan. Gambar motor induksi 3 fasa diperlihatkan pada gambar 4.6.

Gambar 4.6 Motor Induksi 3 Fasa

4.3.4 Modul Arduino Mega 2560

Pada alat ini digunakan Arduino Mega 2560 sebagai pengolah dalam proses pengaturan putaran motor induksi 3 fasa dalam menggerakkan *propeller*. Tampak depan Arduino mega 2560 ditunjukkan dalam Gambar 4.7.



Gambar 4.7 Tampak depan Arduino mega 2560

Arduino Mega 2560 adalah merupakan *board* mikrokontroler berbasis ATMega 2560. Modul ini memiliki 54 digital *input/output* di mana 14 digunakan untuk *output* PWM dan 16 digunakan sebagai analog *input*, 4 untuk UART, 16 MHz osilator kristal, koneksi USB, *power jack*, *ICSP Header*, dan tombol reset.

Modul ini memiliki segalanya yang dibutuhkan untuk memprogram mikrokontroler seperti kabel USB dan sumber daya melalui Adaptor ataupun baterai. Pin masukan dan keluaran Arduino Mega 2560 pada perancangan ini akan difungsikan sesuai Tabel 4.1

Tabel 4.1 Fungsi Pin Arduino Mega 2560

No	Pin	Fungsi
1	A0	Masukan *Pre-Amplifier*
2	8	Masukan motor
3	GND	Jalur masukan GND seluruh sistem
4	Vin	Jalur masukan 5V seluruh sistem

4.3.5 *Variable Frequency Drive*

Dalam penelitian ini, *Variable Frequency Drive* digunakan sebagai *inverter*. *Inverter* adalah suatu peralatan elektronika daya yang berfungsi untuk mengubah listrik DC menjadi AC. *Inverter* seringkali disebut sebagai *Variable Frequency Drive* (VFD).

Variable Frequency Drive mengubah input motor (listrik AC) menjadi DC dan kemudian dijadikan AC lagi dengan frekuensi yang dikehendaki sehingga motor dapat dikontrol sesuai dengan kecepatan yang diinginkan. Untuk mengubah tegangan AC menjadi DC dibutuhkan penyearah (*converter* AC-DC) dan biasanya menggunakan penyearah tidak terkendali (*rectifier dioda*) namun juga ada yang menggunakan penyearah terkendali *(thyristor rectifier*). Setelah tegangan sudah diubah menjadi DC maka diperlukan perbaikan kualitas tegangan DC dengan menggunakan tandon kapasitor sebagai perata tegangan. Kemudian tegangan DC diubah menjadi tegangan AC kembali oleh *inverter* dengan teknik PWM (*Pulse Width Modulation*). Setelah ini bisa didapatkan amplitudo dan frekuensi keluaran yang diinginkan. Gambar alat *Variable Frequency Drive* diperlihatkan pada gambar 4.8.

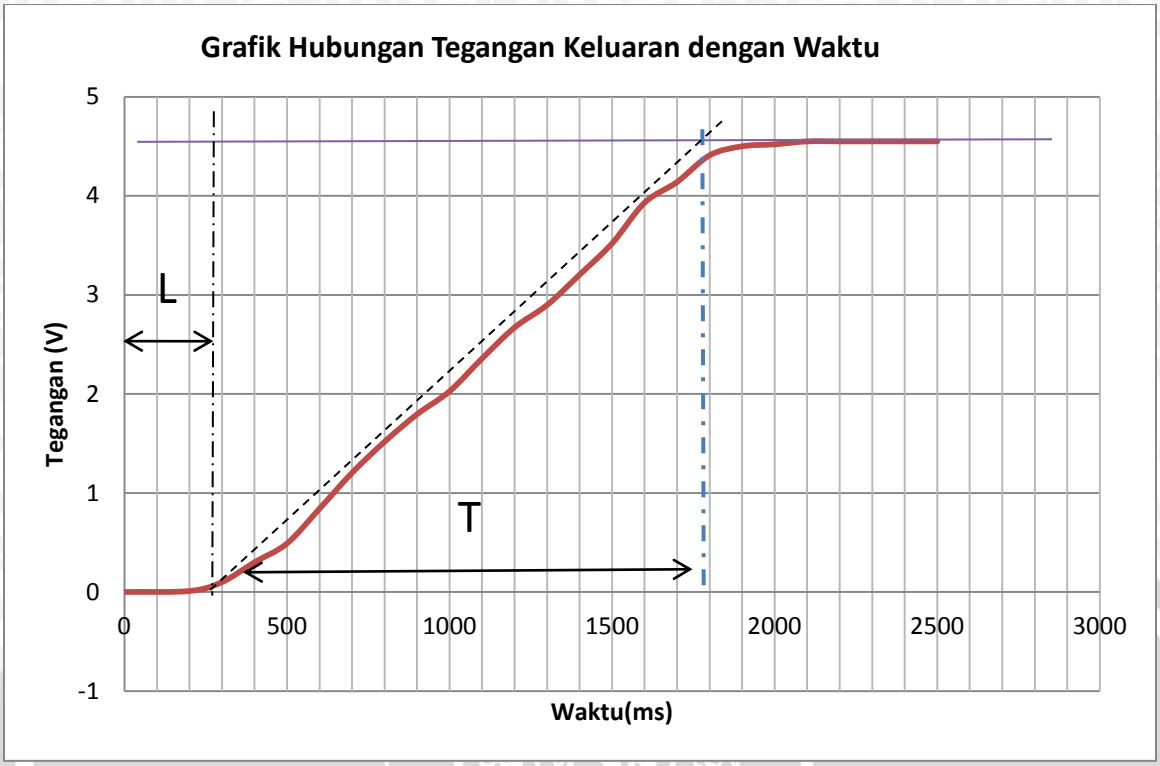
Gambar 4.8 Bentuk fisik *Variable Frequency Drive*

4.4 Perancangan Kontrol PID

Pengambilan data dilakukan dengan pembacaan sensor yang masuk dalam serial monitor Arduino mega. *Tuning* kontroler PID adalah dengan menggunakan metode *Ziegler-Nichols 1* yang telah dimasukkan pada Arduino Mega yang digunakan. Metode ini dipilih karena respon *plant* yang menghasilkan kurva berbentuk S. Kurva tanggapan *plant* digunakan untuk mencari waktu tunda L dan konstanta waktu T yang diperlihatkan dalam Gambar 4.9 sehingga diperoleh nilai dari tiga buah parameter yang terdapat pada kontroler PID yaitu konstanta proporsional (Kp), konstanta integral (Ki) dan konstanta diferensial (Kd).

Dari hasil pengujian *open loop* yang diperlihatkan dalam Gambar 5.3 pada bab selanjutnya berupa kurva S akan digunakan untuk menentukan parameter *tuning* PID dengan metode 1 *Ziegler-Nichols*. Adapun langkah-langkah yang dilakukan sebagai berikut:

1. Menarik garis tangent pada titik infleksi grafik karakteristik *open loop* seperti yang diperlihatkan dalam Gambar 4.9
2. Menentukan perpotongan garis tangent terhadap sumbu waktu t untuk mendapatkan L
3. Menentukan perpotongan garis tangent terhadap sumbu *steady* untuk mendapatkan nilai T
4. Nilai L dan T digunakan untuk menentukan nilai Kp Ki dan Kd sesuai dengan Tabel 4.2.



Gambar 4.9 Metode 1 *Ziegler-Nichols* (hasil pengujian)

Tabel 4.2 Aturan Metode 1 *Ziegler-Nichols* (Ogata K., 1997)

Tipe Kontrol	Kp	τ_i	τ_d
P	$\frac{\tau}{L}$	∞	0
PI	$0{,}9\ \frac{\tau}{L}$	$\frac{L}{0{,}3}$	0
PID	$1{,}2\ \frac{\tau}{L}$	2L	0,5L

Dalam Gambar 4.9, besarnya *steady state* ketika keluaran sensor adalah maksimal yaitu sebesar 4,5 Volt yang ekivalen dengan 8000 RPM. sehingga diperoleh besarnya L = 300mS dan T = 1800-300 = 1500mS.

Dengan demikian parameter kontroler diperoleh sebagai berikut:

$$K_p = 1{,}2\ \frac{\tau}{L} = 1{,}2 \times \frac{1{,}400S}{0{,}300S} = 4{,}67$$

$\tau_i = 2L = 2 \times 0{,}300S = 0{,}6$

$\tau_d = 0{,}5L = 0{,}5 \times 0{,}300S = 0{,}15$

Selanjutnya akan diperoleh Ki dan Kd sebagai berikut:

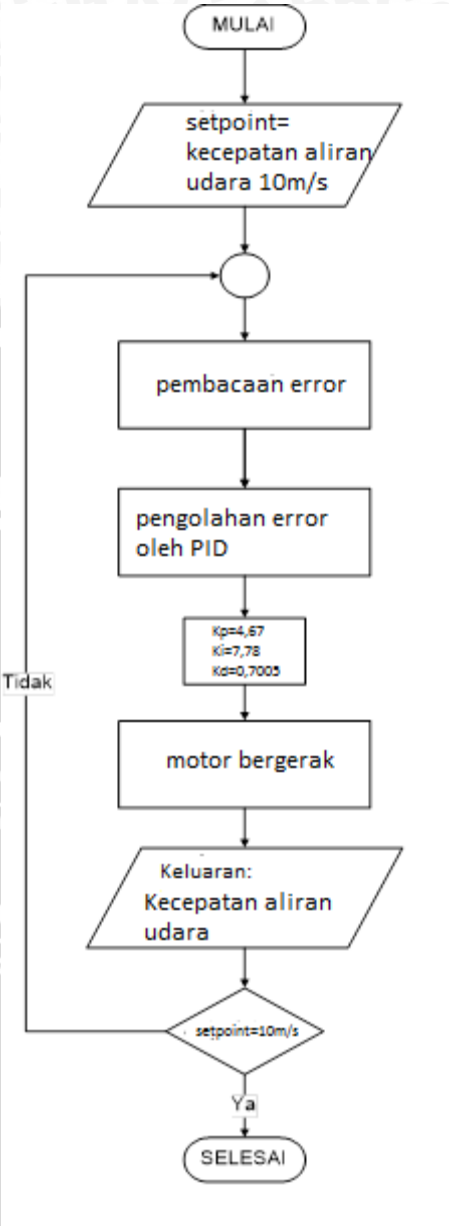
$K_i = \frac{Kp}{\tau_i} = \frac{4{,}67}{0{,}6} = 7{,}78$

$K_d = Kp \times \tau_d = 4{,}67 \times 0{,}2 = 0{,}7005$

4.5 Perancangan Perangkat Lunak

Perancangan perangkat lunak pada pengendalian ini menggunakan bahasa pemrograman C++ dengan menggunakan *software* Arduino ERW 1.0.5. *Flowchart* perancangan perangkat lunak dapat dilihat pada Gambar 4.10. Penjelasan *flowchart* dalam Gambar 4.10 sebagai berikut:

- *Setpoint* berupa kecepatan aliran udara sebesar 10 m/s
- Keluaran dari sensor dibandingkan dengan setpoint menghasilkan error dimana *error* adalah selisih antara *setpoint* dengan *output*
- Kemudian oleh PID, *error* dikurangi dengan memasukkan nilai-nilai parameter PID
- Keluaran dari PID menghasilkan masukan bagi VFD sehingga motor bergerak pada kecepatan tertentu
- Motor menggerakkan *propeller* sehingga menghasilkan aliran udara dengan kecepatan tertentu
- Jika kecepatan aliran udara belum sesuai dengan *setpoint* maka sistem kembali ke pembacaan *error* hingga diperoleh *error* terkecil, jika sudah diperoleh *error* terkecil maka sistem sudah berjalan sesuai dengan *setpoint* dengan baik.



Gambar 4.10 *Flowchart* Perangkat Lunak

BAB V
PENGUJIAN DAN ANALISIS

Tujuan pengujian sistem ini adalah untuk menentukan apakah alat yang telah dibuat berfungsi dengan baik dan sesuai dengan perancangan. Pengujian pada sistem ini meliputi pengujian setiap blok maupun pengujian secara keseluruhan. Pengujian setiap blok ini dilakukan untuk menemukan letak kesalahan dan mempermudah analisis pada sistem apabila alat tidak bekerja sesuai dengan perancangan. Adapun langkah – langkah pengujian yang dilakukan adalah:

1. Pengujian kontrol *air flow* terhadap putaran *propeller*
2. Pengujian respon *open loop plant wind tunnel*
3. Pengujian keseluruhan sistem

5.1 Pengujian Sensor *Air Flow*

a. Tujuan

Menguji tingkat kelinieran *output* sensor *air flow* dalam membaca perubahan kecepatan aliran udara yang dihasilkan terhadap putaran *propeller*.

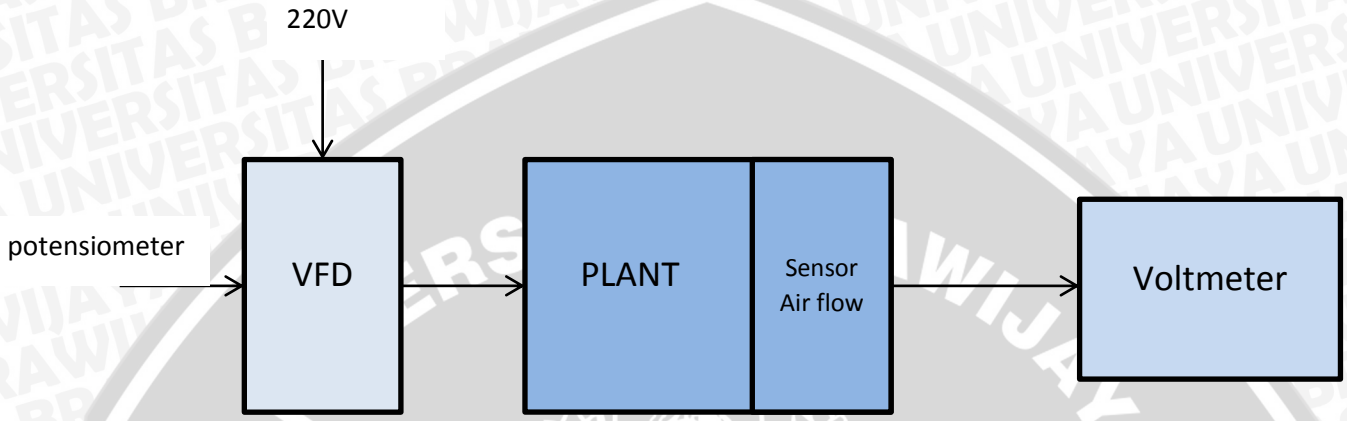
b. Peralatan yang digunakan

- Laptop
- Sensor *Air Flow*
- Arduino Mega
- Mekanikal alat uji *wind tunnel*
- Modul *Variable Frequency Drive*
- Program dan software Arduino ERW 1.0.5
- Data *logger*

c. Langkah Pengujian

1. Menghubungkan *Plant* dengan sensor *air flow*, rangkaian *Variable Frequency Drive*, dan Arduino Mega.
2. Menghidupkan motor dan tentukan batas minimal mesin menyala.
3. Kecepatan putaran maksimal sesuai dengan batas yang telah ditentukan yaitu 11.040 RPM.

4. Sensor akan membaca sesuai dengan kecepatan aliran udara.
5. Mencatat hasil keluaran RPM dan tegangan keluaran sensor kemudian membuat hubungan antara keduanya.



Gambar 5.1 Diagram Blok Pengujian Sensor

d. Hasil Pengujian

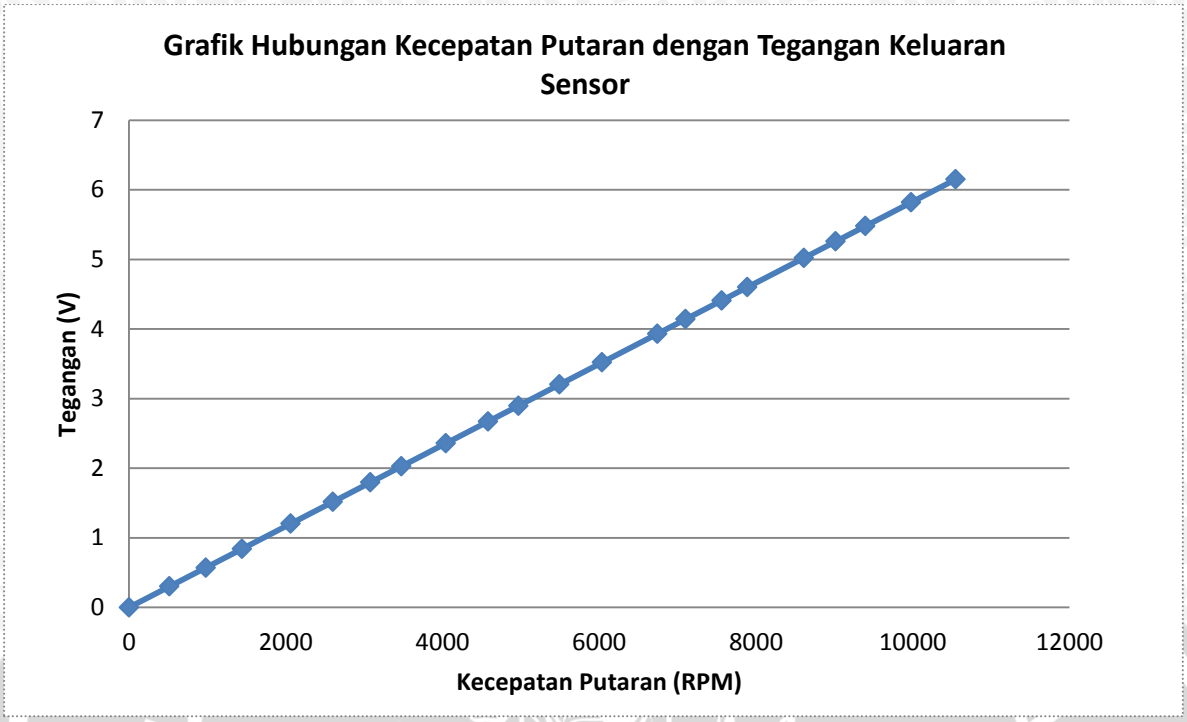
Adapun hasil pengujian sensor *air flow* diperlihatkan dalam tabel 5.1.

Tabel 5. 1 Hasil Pengujian Sensor

No.	Kecepatan putaran (rpm)	Kecepatan *air flow* (m/s)	Tegangan Keluaran (Volt)
1	0	0	0
2	515	0.60	0.30
3	979	1.14	0.57
4	1443	1.68	0.84
5	2063	2.41	1.20
6	2601	3.03	1.52

7	3078	3.59	1.79
8	3476	4.05	2.03
9	4044	4.72	2.36
10	4581	5.34	2.67
11	4970	5.80	2.90
12	5495	6.41	3.20
13	6038	7.04	3.52
14	6742	7.86	3.93
15	7101	8.28	4.14
16	7562	8.82	4.41
17	7891	9.20	4.60
18	8614	10.04	5.02
19	9019	10.52	5.26
20	9398	10.96	5.48
21	9982	11.64	5.82
22	10549	12.30	6.15

Sedangkan grafik hubungan antara kecepatan putaran motor dengan tegangan keluaran diperlihatkan pada gambar 5.2. Dari hasil pengujian yang dilakukan, sensor dapat bekerja dengan maksimal dan terlihat kelinieran yang baik sehingga ideal untuk digunakan sebagai pendeteksi kecepatan putaran pada *propeller*.



Gambar 5. 2 Grafik Hubungan Kecepatan Putaran *Propeller* dengan Tegangan Keluaran Sensor

5.2. Pengujian respon *open loop plant wind tunnel*

a. Tujuan

Menguji respon kecepatan aliran udara yang dihasilkan sistem secara *open loop* dari *wind tunnel*.

b. Peralatan yang digunakan

- Sensor *Air Flow*
- Motor Induksi 3 Fasa ½ HP
- Mekanikal alat uji *wind tunnel*
- Catu daya 12 V DC
- Modul *VFD (Variable Frequency Drive)*
- *Voltmeter*
- *Oscilloscope* digital

c. Langkah pengujian

1. Menghubungkan *Variable Frequency Drive* dengan motor induksi 3 fasa.

2. Mengatur putaran motor induksi 3 fasa pada 0 rpm
3. Menaikkan putaran motor sampai 8000 rpm
4. Mencatat tegangan keluaran sensor yang ditampilkan *oscilloscope* digital

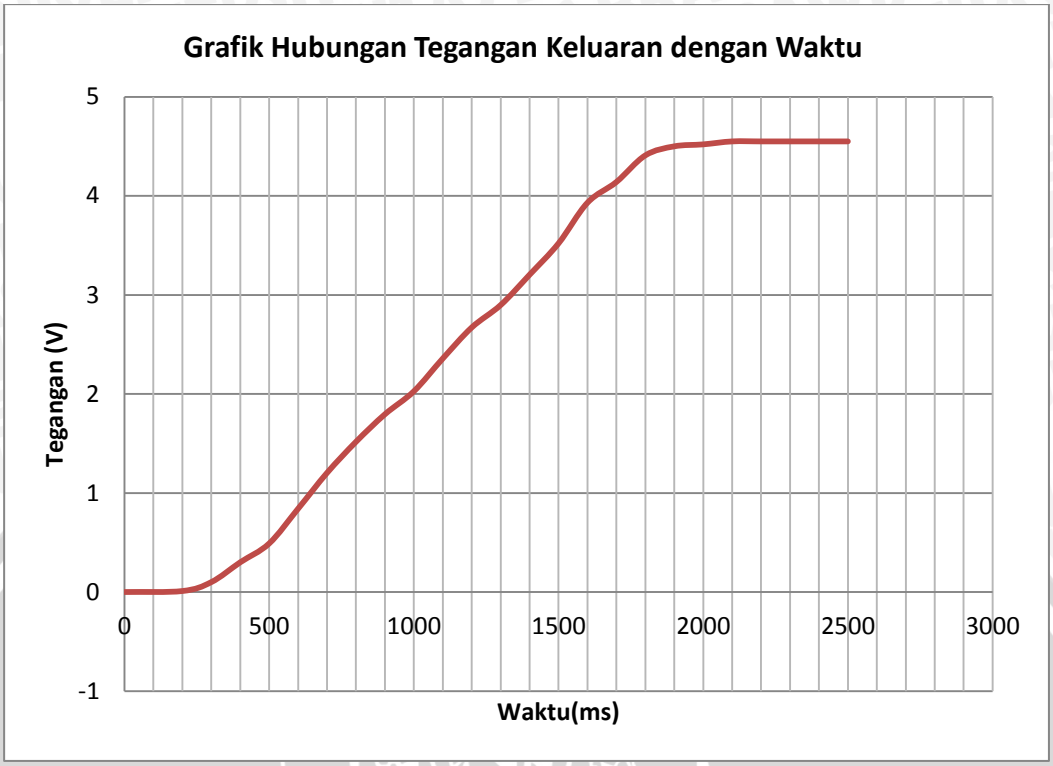
d. Hasil Pengujian

Adapun hasil pengujian aktuator diperlihatkan pada tabel 5.2.

Tabel 5.2 Hasil pengujian aktuator

vout	t(ms)
0	0
0.001	200
0.17	300
0.30	400
0.47	500
0.69	600
0.84	700
1.20	800
1.52	900
1.79	1000
2.03	1100
2.36	1200
2.67	1300
2.90	1400
3.20	1500
3.52	1600
3.93	1700
4.14	1800
4.41	1900
4.50	2000
4.52	2100
4.55	2200

Setelah melakukan pengujian secara *open loop* sesuai dengan langkah diatas, didapatkan hubungan antara tegangan keluaran terhadap waktu ditunjukkan dalam Gambar 5.3.



Gambar 5. 3 Grafik respon aktuator

5.3 Pengujian Keseluruhan Sistem

Pengujian sistem secara keseluruhan ini dilakukan untuk mengetahui kinerja perangkat keras dan perangkat lunak serta mengetahui respon sistem secara *close loop*.

a. Tujuan

Pengujian ini bertujuan untuk mengetahui bagaimana kinerja sistem secara *close loop* dan mengamati respon yang dihasilkan setelah diberi nilai-nilai parameter kontroler agar sistem bekerja sesuai *setpoint* yang diinginkan.

b. Peralatan yang digunakan

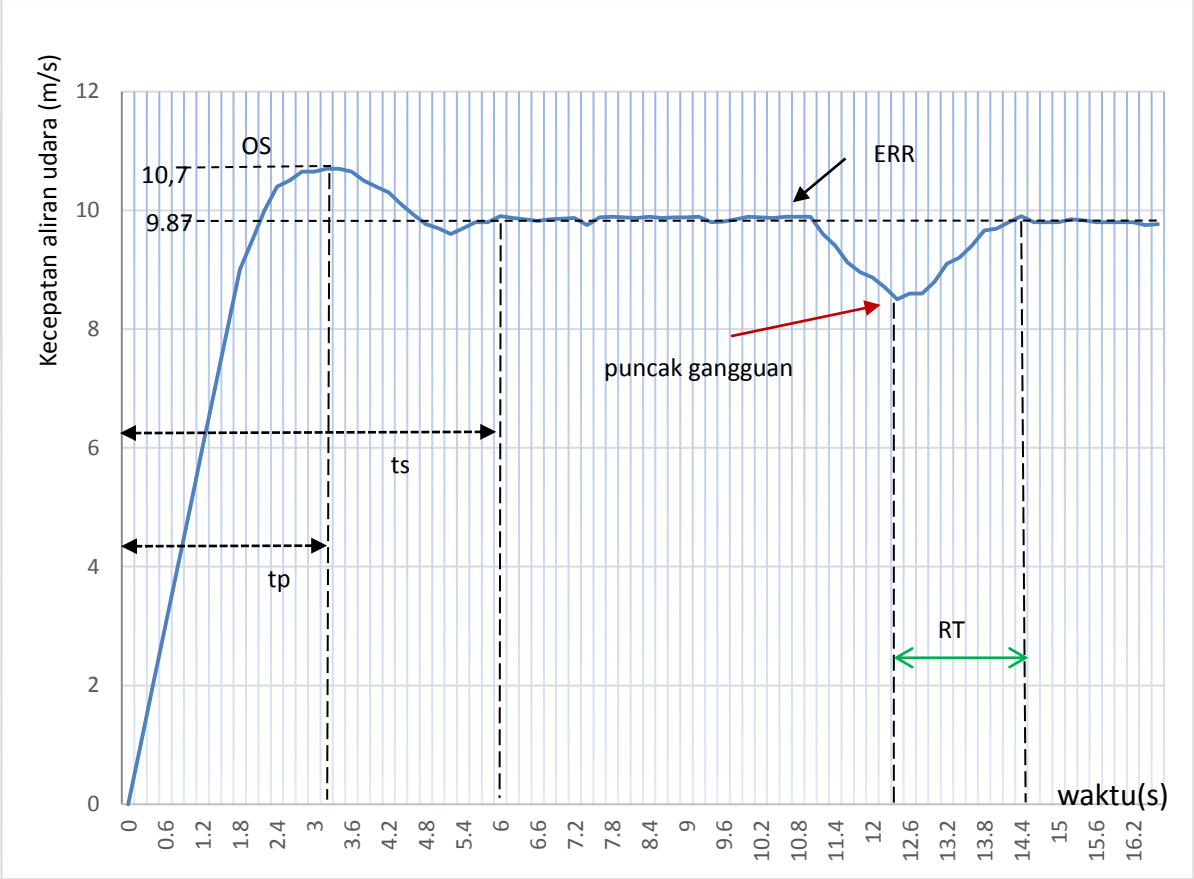
- Catu daya 5 V
- Mekanikal alat uji *wind tunnel*
- Arduino Mega
- Motor induksi 3 fasa
- *Variable Frequency Drive*
- PC / laptop.
- Program dan *software* Arduino ERW 1.0.5

c. Langkah Pengujian

1. Mengunduh program dengan *setpoint* kecepatan aliran udara 10 m/s beserta program KLF pada *software* Arduino ERW 1.0.5.
2. Merekam respon dengan *oscilloscope* digital
3. Memberikan gangguan dengan menghalangi aliran udara
4. Mengamati hasil keluaran nilai kecepatan aliran udara kemudian membuat grafik kecepatan aliran udara terhadap waktu.

d. Hasil Pengujian

Setelah melakukan prosedur pengujian, didapatkan grafik respon ditunjukkan dalam Gambar 5.4 dan hasil respon ditunjukkan dalam Tabel 5.3.



Gambar 5.4 Pengujian Sistem dengan *Setpoin*t kecepatan aliran udara sebesar 10 m/s

Dari hasil pengujian secara *closed loop* seperti yang diperlihatkan dalam Gambar 5.4 dapat diperoleh parameter unjuk kerja sistem sebagai berikut:

1. *Peak Time* (t_p) terjadi pada sekitar 3,1 detik
2. *Steady state* terjadi pada sekitar 6,1 detik
3. *Maximum Overshoot* sebesar $\frac{10,7-9,87}{9,87} \times 100\% = 8,4\%$
4. Besarnya *Error steady state* sebesar $\frac{9,89-9,87}{9,89} \times 100\% = 0,22\%$
5. *Recovery Time* setelah adanya gangguan sebesar 14400-12500 = 1900ms.

Tabel 5.3 Hasil pengujian keseluruhan

Waktu (s)	Kecepatan aliran udara (m/s)
0	0
0.2	1
0.4	2
0.6	3
0.8	4
1	5
1.2	6
1.4	7
1.6	8
1.8	9
2	9.5
2.2	10
2.4	10.4
2.6	10.5
2.8	10.65
3	10.65
3.2	10.7
3.4	10.7
3.6	10.65
3.8	10.5
4	10.4
4.2	10.3
4.4	10.1
4.6	9.93
4.8	9.77
5	9.7
5.2	9.6
5.4	9.7
5.6	9.8
5.8	9.8
6	9.9
6.2	9.87
6.4	9.85
6.6	9.82
6.8	9.85
7	9.86
7.2	9.87
7.4	9.75
7.6	9.88
7.8	9.89
8	9.88
8.2	9.87
8.4	9.89
8.6	9.87
8.8	9.88
9	9.88
9.2	9.89
9.4	9.8
9.6	9.81
9.8	9.85
10	9.89
10.2	9.88
10.4	9.87
10.6	9.89
10.8	9.89
11	9.89
11.2	9.6
11.4	9.4
11.6	9.12
11.8	8.96
12	8.87
12.2	8.7
12.4	8.5
12.6	8.6
12.8	8.6
13	8.8
13.2	9.1
13.4	9.2
13.6	9.4
13.8	9.66
14	9.69
14.2	9.8
14.4	9.9
14.6	9.8
14.8	9.8
15	9.8
15.2	9.85
15.4	9.83
15.6	9.8
15.8	9.8
16	9.8
16.2	9.8
16.4	9.75
16.6	9.77

BAB VI
PENUTUP

6.1 Kesimpulan

Dari perancangan kemudian dilakukan pengujian dari keseluruhan sistem, selanjutnya dapat disimpulkan sebagai berikut:

1. Dari hasil perancangan dan pengujian alat yang telah dilakukan, maka diperoleh nilai konstanta parameter kontrol PID yaitu Kp = 4,67, Ki = 7,78 dan Kd = 0,7005.
2. Pada pengujian dengan memberikan gangguan dengan menghalangi aliran angin, terjadi perubahan pada kecepatan putaran *propeller* dengan nilai *time overshoot* sebesar 3,1 detik, *settling time* atau waktu pencapaian *steady state* yaitu 6,1 detik, *overshoot* sebesar 8,4%, dan *error* sebesar 0,22%, serta secara keseluruhan sistem dapat kembali pada keadaan *steady* dan mampu memberikan respon sistem yang baik ketika terjadinya gangguan dengan *recovery time* sebesar 1900ms.

6.2 Saran

Sebagai pengembangan selanjutnya terdapat beberapa saran-saran sebagai berikut:

1. Sistem dapat dikembangkan dengan menggunakan metode kontroler selain PID

DAFTAR PUSTAKA

Gunterus, F. 1994. *Falsafah Dasar : Sistem Pengendalian Proses*. Jakarta: Elex Media Komputindo.

Krisnanda, F. 2014. *Pengendalian Kecepatan Putaran Gas Engine Pada RC Airplane Menggunakan Kontroller Proporsional Integral Deferensial (PID) Berbasis Mikrokontroller Atmega 328.* Malang: Teknik Elektro Universitas Brawijaya.

Khamdani, F. 2012. *Studi Eksperimental Aliran Campuran Air-Crude Oil yang Melalui Pipa Pengecilan Mendadak Horizontal Berpenampang Lingkaran.* Teknik Mesin Universitas Diponegoro

Muchammad. 2006. *Perhitungan Gaya Drag Pada Benda Uji Pelat Persegi Datar Menggunakan Low Speed Wind Tunnel*. Teknik Mesin Universitas Diponegoro

Ogata, K. 1997. *Teknik Kontrol Automatik Jilid 1*. Jakarta: Penerbit Erlangga.

Ogata, K. 1997. *Teknik Kontrol Automatik Jilid 2*. Jakarta: Penerbit Erlangga.

Santoso, A. 2014. Analisis Pengaruh Kecepatan Motor Pada Quadcoper Terhadap Kestabilan Multicopter Aerial Cam. Teknik Elektro Universitas Pancasila.

Setiawan, I. 2008. Kontrol PID untuk Proses Industri. Elex Media Komputindo, Jakarta.

Welander, P. 2010. "Understanding Derivative in PID Control", Control Engineering, 2, 24-27..

Yefri. 2010. "Cara Menghitung Daya Blower Fan". https://yefrichan.wordpress.com/2010/08/02/cara-menghitung-daya-blowerfan/

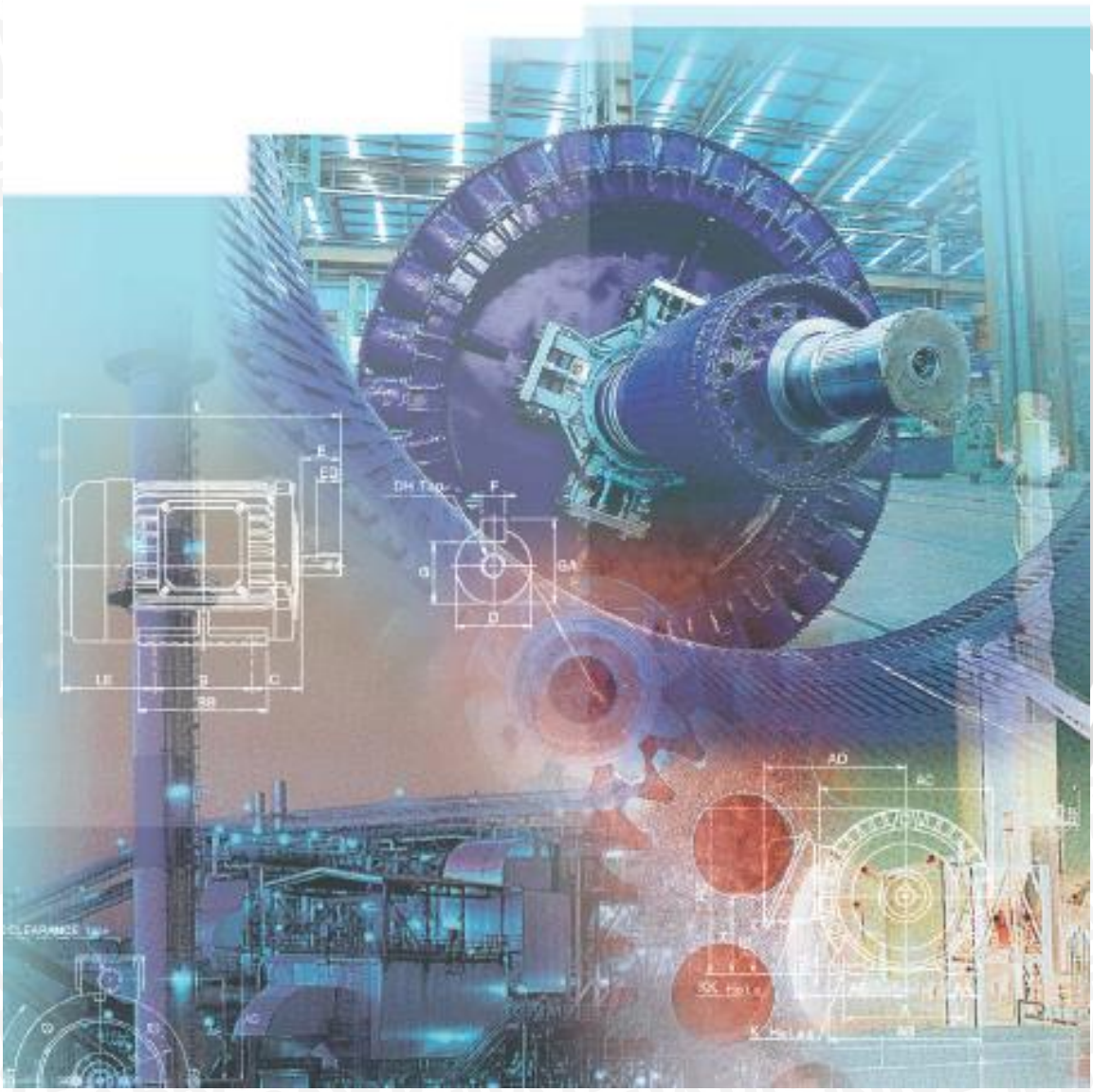
LAMPIRAN 1

DATASHEET

TECO

3-PHASE INDUCTION MOTORS

AEEF · AEVF



SPECIFICATION TABLE
TYPE:AEEF, AEVF
LOW VOLTAGE SQUIRREL CAGE

ITEM		STANDARD SPECIFICATION
RATING	Kind of Motors	Squirrel - Cage Induction Motors (SCIM).
	Design standards	IEC, CNS, JIS.
	Voltages	220V, 380V, or 380V, 400V, 415V, 440V, 460V.
	Frequency	50Hz or 60Hz.
	Output Range	0.25 HP ~ 270 HP (0.18 kW ~ 200 kW).
	Time Duty	Continuous, S1, S.F. : 1.0.
	Frame Nom.	63 ~ 315M.
	Protection Enclosure	Totally Enclosed Fan Cooled (IP 54).
	Cooling Method	Self External Fan, Surface Cooling (IC 411).
	Mounting	Horizontal Foot Mounting (IM 1001), Flange Mounting (IM3011).
	Environment Conditions	Place : Shadow, Non-Hazardous. Ambient Temperature : -15°C ~ 40°C. Relative Humidity : Less Than 90% RH (Non-Condensation). Altitude : Less Than 1000M.
	Drive Method	Belt Service, However, 2 Pole 30 HP and Up Coupling Service is the Way.
	Direction of Rotation	Bi - Directional.
	Method of Starting	Full Voltage Direct On Line or Y - Δ Starting.
APPLICATION	Shaft	Carbon Steel, Cylindrical Single Extension with Keyway and Key.
	Bearing	Bracket Mounting, Vacuum De - Gassed High Quality Open Bearings for Frame No. 250 ~ 315, Grease Pre - Packed Shielded Rolling Bearings for the Others.
	Lubrication	Mineral Oil, Li - Base Grease (Frame 63 ~ 250 MULTEMP SRL, Frame 280 ~ 315 SHELL ALVANIA RL3).
CONSTRUCTION	Terminal Box	Pressed Steel, Larger Size, Can be Set 90° Apart, With Conduit Hole Cable Entrance At Left Side View from the Drive End. Option : Cable Grand.
	Lead Terminal	Solderless Lug Terminals. Option : Wire Connection Seat.
	Stator Insulation	Class E Insulation System for Frame No. 63 ~ 112M Class B Insulation System for Frame No. 132S ~ 180M Class F Insulation System for Frame No.180L ~ 315M
	Rotor Winding	Squirrel Cage, Copper Bar Brazed or Aluminium Conductor with End - Ring and Wafter Blades Integrally Cast.
	Painting	Phenolic Rust Proof Base Plus Lacquer Surface Finished Painting In Blue - Gray Color (Munsell 7.5B 5.5 / 0.5).
	Name Plate	Stainless Steel Plate.
	Bolt Thread	ISO Metric System.
	Grounding Terminal	Be Set Inside the Terminal Box.
PERFORMANCE	Test Procedure	IEC 60034, CNS 10919 (C3192) And Full Voltage Measuring Starting Performance.
	Temperature Rise	Winding Temperature Rise (by Resistance Method) Class E Insulation Not to Exceed 75°C. Class B Insulation Not to Exceed 80°C. Class F Insulation Not to Exceed 100°C.

PERFORMANCE DATA

220V 60Hz

OUTPUT		FULL LOAD RPM	FRAME NO.	EFFICIENCY			POWER FACTOR			CURRENT		TORQUE			ROTOR GD² kg-m²
HP	kW			FULL LOAD (%)	3/4 LOAD (%)	1/2 LOAD (%)	FULL LOAD (%)	3/4 LOAD (%)	1/2 LOAD (%)	FULL LOAD (A)	LOCKED LOAD (A)	FULL LOAD (kg-m)	LOCKED LOAD (%F.L.T.)	PULL OUT (%F.L.T.)	
0.25	0.18	3335	63	61.0	59.5	52.0	77.0	67.5	55.5	1.0	6	0.054	400	370	0.002
		1640	63	67.0	66.0	61.0	65.5	56.0	44.0	1.1	6	0.111	250	240	0.002
		1120	71	64.0	57.5	49.5	60.0	53.0	43.0	1.3	6	0.162	260	250	0.007
0.5	0.37	3400	71	75.0	74.5	70.0	85.0	79.0	68.5	1.5	12	0.107	340	310	0.002
		1660	71	70.0	66.5	60.5	71.0	60.5	48.0	2.0	12	0.216	200	250	0.005
		1135	80	68.0	64.0	57.5	67.0	56.0	45.0	2.2	12	0.330	200	240	0.009
1	0.75	3395	80	77.0	77.5	75.0	87.0	83.5	73.5	3.0	19	0.214	220	280	0.005
		1710	80	78.0	74.5	70.0	76.5	67.5	54.5	3.4	19	0.424	250	280	0.009
		1140	90L	76.0	75.5	71.5	71.0	62.5	50.0	3.6	19	0.637	200	240	0.017
2	1.5	3425	90L	80.0	81.5	80.0	86.0	83.0	74.5	5.5	40	0.424	250	280	0.010
		1715	90L	79.0	79.0	75.0	81.0	70.5	57.0	6.1	40	0.846	220	280	0.017
		1140	100L	78.0	77.5	74.0	74.0	66.0	54.0	6.8	40	1.273	180	220	0.033
3	2.2	3450	90L	83.0	84.5	82.0	89.0	85.5	76.5	8.0	68	0.631	220	280	0.015
		1735	100L	82.0	83.5	80.5	82.5	76.5	66.5	8.7	68	1.266	210	280	0.033
		1160	112M	82.0	83.0	79.5	77.0	67.5	56.0	9.3	68	1.877	180	270	0.059
5	3.7	3485	112M	84.5	86.0	83.5	90.0	86.5	82.0	12.9	110	1.041	240	340	0.038
		1745	112M	85.0	85.5	83.0	85.0	80.0	70.0	13.5	110	2.080	220	300	0.059
		1160	132S	84.0	83.0	79.5	77.0	68.5	58.0	15.1	110	3.129	180	230	0.161
7.5	5.5	3505	132S	85.0	85.5	84.0	90.0	86.5	81.5	19.2	160	1.553	220	280	0.063
		1750	132S	87.0	87.0	85.0	84.0	78.5	68.5	20.1	160	3.111	230	280	0.104
		1160	132M	85.0	86.0	83.5	77.0	72.0	60.5	22.2	160	4.993	200	230	0.217
10	7.5	3510	132S	88.5	88.0	85.5	90.0	87.0	81.0	25.1	200	2.068	200	270	0.078
		1760	132M	88.5	89.0	87.5	86.0	85.5	76.0	25.1	200	4.148	220	250	0.143
		1175	160M	87.0	86.0	84.0	90.0	86.0	82.5	28.1	200	6.176	260	300	0.400
15	11	3540	160M	88.0	87.5	86.5	90.0	88.0	84.0	37.1	290	3.076	220	300	0.147
		1760	160M	90.0	89.5	88.0	89.0	86.0	78.5	38.7	290	6.166	220	250	0.297
		1170	160L	89.5	88.5	88.0	84.0	75.0	70.5	39.1	290	9.306	240	250	0.588
20	15	3520	160M	89.5	90.0	89.0	91.0	90.0	87.5	48.1	390	4.124	210	280	0.183
		1760	160L	90.5	90.5	90.5	86.0	84.0	76.5	50.3	390	8.348	230	280	0.381
		1170	180MC	90.2	91.4	90.5	85.0	79.5	72.5	51.2	390	12.408	220	250	1.054
25	18.5	3530	160L	90.0	91.5	90.5	89.5	90.5	87.0	60.8	480	5.141	240	290	0.237
		1760	180MC	91.5	91.0	90.0	85.5	83.0	77.0	62.9	440	10.310	210	240	0.571
		1170	180LC	90.2	91.2	90.5	84.5	80.0	72.0	64.4	440	15.510	220	250	1.233
30	22	3540	180MA	90.5	91.5	89.5	90.0	88.5	83.5	72.1	550	6.151	210	250	0.302
		1765	180MC	91.5	92.5	91.5	88.0	83.5	77.0	72.9	550	12.338	210	250	0.706
		1175	180LC	91.0	92.0	91.0	84.0	78.5	69.5	75.8	550	18.533	200	260	1.438
40	30	3520	180LA	90.5	91.0	89.5	91.0	90.0	86.0	95.1	680	8.348	230	350	0.358
		1760	180LC	92.0	93.0	93.0	89.0	84.5	78.0	98.7	680	16.497	220	240	0.810
		1170	200LC	92.0	92.5	92.5	85.0	80.0	73.5	100	620	24.816	150	200	1.919
50	37	3545	200LA	91.0	93.0	87.5	89.0	90.5	87.0	121	800	10.239	150	210	0.602
		1770	200LC	92.0	93.0	93.0	89.0	86.5	81.5	124	950	20.604	240	250	1.422
		1175	200LC	92.5	93.0	92.5	84.0	78.0	70.5	126	850	30.888	210	230	2.419
60	45	3545	200LA	91.5	91.5	90.0	90.0	86.5	82.5	143	950	12.285	170	220	0.633
		1765	200LC	92.0	93.0	92.5	89.0	87.0	84.0	144	1060	24.675	210	230	1.643
		1180	225SC	92.5	93.0	93.0	86.0	84.0	78.5	148	980	36.808	220	220	3.203
75	55	3560	225SA	92.0	93.0	91.5	92.0	91.5	87.5	174	1220	15.335	150	210	1.187
		1775	225SC	92.5	93.5	92.0	86.5	85.5	79.5	184	1220	30.670	180	230	1.979
		1175	A250SC	93.0	93.5	93.0	85.5	85.0	78.0	185	1220	46.331	200	230	4.893
		885	280S	92.0	91.5	90.0	77.0	73.5	68.0	202	1215	63.5	135	210	15.0
100	75	3560	A250SA	93.5	93.5	93.0	91.5	89.5	86.5	229	1600	20.447	150	240	1.578
		1775	A250SC	93.5	92.5	91.0	89.0	87.0	83.0	235	1600	40.893	170	270	4.228
		1175	A250MC	93.5	94.0	93.5	86.0	83.5	78.5	244	1600	61.775	200	250	6.582
		885	280M	92.5	93.0	93.5	77.5	74.0	69.0	276	1660	82.5	135	210	19.6
125	90	3555	A250MA	94.0	93.5	92.0	91.5	90.5	86.5	285	2150	25.522	150	250	2.014
		1770	A250MC	94.0	94.0	93.0	92.0	92.0	89.5	283	2150	51.261	190	230	5.161
		1180	280S	94.0	92.8	92.0	85.0	81.5	73.5	298	1990	74.9	130	230	19.8
		885	315S	93.0	93.5	93.5	75.0	74.0	69.0	326	1990	93.0	130	210	27.6
150	110	3565	280S	94.0	93.3	91.5	89.5	88.5	84.0	343	2435	30.0	100	220	4.0
		1770	280S	94.7	94.2	92.7	87.0	85.0	78.0	350	2435	60.5	130	230	7.6
		1180	280M	94.3	93.6	92.0	85.5	82.0	73.5	358	2435	90.8	130	230	19.2
		885	315M	93.0	93.5	90.5	75.0	74.0	69.0	398	2435	121.0	130	210	25.6
175	132	3565	280M	94.5	93.8	92.0	89.0	88.0	83.5	412	2900	35.1	100	220	4.5
		1770	280M	94.9	94.4	92.9	88.0	86.0	79.0	415	2900	72.6	130	230	8.6
		1185	315S	94.5	94.0	92.5	86.0	83.0	77.0	426	2900	108.5	130	230	19.5
200 215	160	3570	315S	94.0	94.1	92.0	90.5	89.8	86.5	489	3540	43.8	100	220	5.7
		1775	315S	95.0	94.5	92.9	88.5	87.5	80.5	494	3540	87.8	130	230	11.2
		1185	315M	94.6	94.2	92.7	86.5	83.5	80.0	513	3540	131.5	130	230	21.4
250 270	200	3570	315M	95.0	94.2	92.0	91.2	90.5	88.5	605	4425	54.7	100	220	7.2
		1775	315M	95.0	94.2	92.0	90.0	88.0	86.5	612	4425	108.7	120	230	14.2

NOTE: 1. The above are typical values based on test, 2. Tolerance According to IEC 34-1,
3. Efficiency, power factor, speed and torque are the same for other voltages. Current values vary inversely with voltage.
4. Data subject to change without notice,

PERFORMANCE DATA

380V 50Hz

OUTPUT		FULL LOAD RPM	FRAME NO.	EFFICIENCY			POWER FACTOR			CURRENT		TORQUE			ROTOR GD² kg-m²
HP	kW			FULL LOAD (%)	3/4 LOAD (%)	1/2 LOAD (%)	FULL LOAD (%)	3/4 LOAD (%)	1/2 LOAD (%)	FULL LOAD (A)	LOCKED LOAD (A)	FULL LOAD (kg-m)	LOCKED LOAD (%FLT)	PULL OUT (%FLT)	
0.25	0.18	2725	63	66.0	59.5	54.0	76.5	71.0	59.0	0.66	3.5	0.067	400	330	0.002
		1345	63	64.0	60.0	57.0	66.5	59.5	46.5	0.65	3.5	0.135	290	240	0.002
		910	71	60.0	54.5	49.0	65.0	50.0	42.0	0.79	3.5	0.199	260	270	0.007
0.5	0.37	2815	71	75.0	74.5	70.5	65.0	76.0	67.0	0.89	7	0.129	320	260	0.002
		1370	71	66.5	66.0	60.5	70.0	62.0	49.0	1.24	7	0.265	200	230	0.005
		930	80	69.0	61.5	55.0	67.0	57.0	45.5	1.24	7	0.390	200	250	0.009
1	0.75	2800	80	76.5	79.0	76.0	67.0	80.0	73.0	1.70	11	0.259	220	260	0.005
		1395	80	73.2	73.0	68.5	74.0	67.0	54.0	2.13	11	0.520	250	260	0.009
		950	90L	71.0	71.5	66.5	70.0	61.0	49.0	2.28	11	0.764	190	230	0.017
2	1.5	2840	90L	80.0	82.0	80.0	86.0	84.0	75.5	3.22	25	0.511	250	260	0.010
		1400	90L	78.5	78.5	75.0	78.5	74.0	60.5	3.66	25	1.037	220	260	0.017
		930	100L	76.0	76.0	71.0	71.5	66.0	52.5	4.17	25	1.561	160	220	0.033
3	2.2	2845	90L	80.5	84.5	83.0	88.5	84.0	75.5	4.80	39	0.765	250	260	0.015
		1425	100L	81.0	81.5	79.0	83.0	74.0	60.0	5.12	39	1.517	210	260	0.033
		955	112M	79.0	79.0	75.0	75.0	67.5	55.0	5.66	39	2.280	160	220	0.059
5	3.7	2880	112M	85.5	86.5	84.5	90.0	87.0	79.0	7.36	63	1.250	240	320	0.038
		1445	112M	84.5	84.5	82.5	83.5	76.5	65.0	8.03	63	2.512	220	290	0.059
		960	132S	82.0	82.0	77.0	75.5	69.5	53.5	9.15	63	3.781	160	220	0.131
7.5	5.5	2905	132S	86.5	87.5	85.5	88.5	86.5	81.0	11.1	93	1.874	210	260	0.063
		1445	132S	86.0	86.0	83.0	82.5	77.0	65.5	12.0	93	3.767	220	260	0.104
		960	132M	84.5	84.0	81.0	77.5	71.0	60.0	13.0	93	5.671	200	230	0.217
10	7.5	2905	132S	88.5	88.5	87.0	88.5	86.0	78.0	14.5	116	2.499	200	260	0.078
		1450	132M	87.5	88.0	86.5	85.5	81.5	71.0	15.2	116	5.006	220	250	0.143
		975	160M	86.0	86.5	84.0	80.0	71.5	59.0	16.5	116	7.445	270	300	0.400
15	11	2940	160M	89.5	89.5	87.0	90.0	87.5	81.5	21.3	169	3.703	210	260	0.147
		1460	160M	89.5	90.0	89.0	86.0	84.0	76.0	21.6	169	7.457	220	250	0.297
		970	160L	88.5	89.5	88.0	84.0	79.5	69.5	22.9	169	11.225	220	250	0.588
20	15	2920	160M	90.0	91.0	90.5	91.0	90.0	87.5	27.7	229	4.972	210	260	0.183
		1465	160L	90.5	90.5	89.0	88.5	85.0	77.5	28.3	229	9.909	230	260	0.591
		975	180MC	89.0	90.5	90.0	83.5	79.0	71.0	33.3	229	14.889	210	250	1.054
25	18.5	2930	160L	90.0	91.0	90.0	89.5	91.5	88.0	35.2	288	6.100	240	250	0.237
		1455	180MC	91.0	91.5	91.0	86.5	82.0	76.0	36.0	288	12.472	210	240	0.571
		975	180LC	90.0	90.5	90.0	83.5	78.0	70.0	38.2	288	18.613	200	240	1.233
30	22	2940	180MA	91.5	91.5	90.0	90.0	87.5	82.5	41.3	319	7.407	210	250	0.282
		1465	180MC	90.5	92.0	92.0	85.5	82.0	75.0	43.9	319	14.864	210	240	0.708
		975	180LC	90.0	90.5	89.0	83.0	76.0	68.0	46.1	319	22.334	200	240	1.438
40	30	2930	180LA	92.0	92.0	91.0	91.0	90.0	86.0	54.2	388	9.943	210	240	0.358
		1465	180LC	91.5	91.5	91.0	85.0	82.0	75.0	58.3	388	19.855	220	230	0.810
		970	200LC	91.5	91.5	91.0	83.0	80.0	74.0	59.7	388	29.932	190	200	1.919
50	37	2940	200LA	92.0	92.0	90.5	87.5	87.5	84.5	70.4	463	12.345	150	210	0.662
		1470	200LC	92.0	92.5	92.5	86.0	86.0	82.0	71.8	463	24.689	190	210	1.422
		975	200LC	92.0	92.0	91.0	80.0	76.0	67.0	77.0	463	37.223	250	250	2.419
60	45	2955	200LA	92.5	92.5	90.0	87.0	85.0	80.0	84.5	582	14.738	160	220	0.633
		1465	200LC	92.5	93.0	93.0	88.5	87.0	81.5	83.1	582	29.728	190	200	1.643
		980	225SC	92.5	92.5	91.5	84.0	82.0	73.0	87.5	582	44.443	220	250	3.023
75	55	2945	225SA	93.0	93.0	91.5	92.0	91.5	89.0	99.4	725	18.485	140	260	1.157
		1470	225SC	93.0	93.0	92.5	85.0	82.5	75.0	108	725	37.034	180	220	1.979
		980	A250SC	93.5	94.0	93.5	86.0	83.0	76.5	106	725	55.550	200	220	4.923
		735	280S	91.5	91.2	89.5	75.0	69.0	58.0	120	733	73.8	140	210	15.0
100	75	2960	A250SA	94.0	94.0	92.0	90.0	88.0	83.5	134	972	24.822	150	260	1.578
		1475	A250SC	94.0	94.0	93.5	88.0	87.0	82.0	136	927	49.211	240	250	4.490
		975	A250MC	93.5	93.5	92.5	87.0	84.0	78.0	140	927	74.447	220	250	6.282
		735	280M	91.5	91.2	89.5	75.0	69.0	58.0	164	983	99.4	140	210	19.6
125	90	2950	A250MA	94.0	94.0	93.0	93.5	89.5	86.0	167	1245	30.757	140	260	2.014
		1475	A250MC	94.0	94.0	93.5	92.0	92.5	88.5	164	1245	61.513	190	230	5.181
		975	280S	93.8	93.6	92.0	81.0	76.5	66.5	190	1150	93.0	130	230	13.8
		735	315S	92.0	91.6	89.0	77.0	70.0	60.0	198	1150	119.8	140	210	21.6
150	110	2950	280S	94.0	93.9	91.5	87.5	85.5	81.5	203	1400	36.3	100	220	4.0
		1475	280S	94.5	94.1	92.5	83.0	81.0	74.0	213	1400	73.6	130	230	7.6
		975	280M	94.0	93.6	92.0	81.0	76.5	66.5	219	1400	110.9	130	230	16.2
		735	315M	92.0	91.6	89.0	77.0	70.0	60.0	236	1400	146.7	140	210	25.6
175	132	2960	280M	94.5	93.8	92.0	87.0	85.0	81.0	244	1660	43.6	100	220	4.5
		1475	280M	94.8	94.3	92.8	83.5	81.5	74.0	254	1660	87.3	130	230	8.6
		980	315S	94.2	93.7	92.2	83.5	76.5	66.5	266	1660	131.0	130	230	18.6
200 215	160	2960	315S	94.6	94.2	92.5	90.0	88.5	83.5	295	2050	52.6	100	220	5.7
		1480	315S	94.9	94.4	92.8	88.5	86.5	79.5	290	2050	105.3	130	230	11.2
		980	315M	94.5	94.0	92.4	85.5	81.5	73.5	301	2050	156.0	130	230	21.6
250 270	200	2960	315M	94.6	94.4	92.7	90.0	88.5	83.5	358	2560	65.8	100	220	7.2
		1480	315M	94.9	94.5	92.8	88.5	86.5	79.5	362	2560	131.6	130	230	14.2

NOTE: 1. The above are typical values based on test, 2. Tolerance According to IEC 34-1,
3. Efficiency, power factor, speed and torque are the same for other voltages. Current values vary inversely with voltage,
4. Data subject to change without notice.

HORIZONTAL FOOT MOUNTED

Totally Enclosed Fan Cooled, Squirrel Cage Rotor,

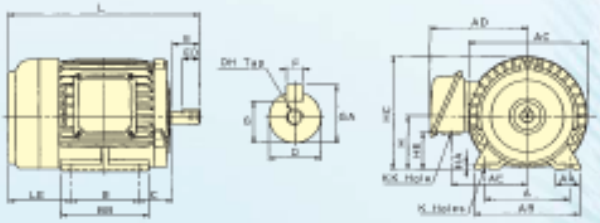


Fig.1

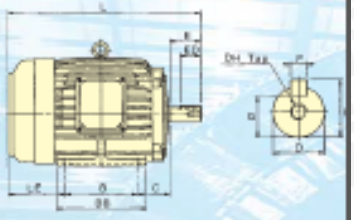


Fig.2

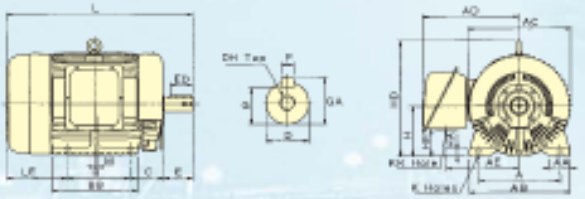


Fig.4

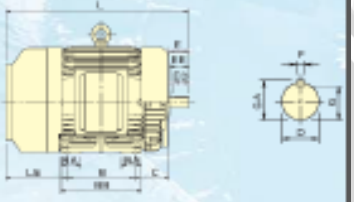


Fig.5

OUTPUT(HP)				FRAME NO.	FIG NO.	A	AA	AB	AC	AD	AE	AF	B	BA	M	BB	C	H	HA	HC	HI
2P	4P	6P	8P																		
1/4	1/6	—	—	63	1	100	28	120	144	123	60	—	80	—	—	100	40	63	8	135	—
1/2	1/2	1/4	—	71		112	35,5	140	162	133	100	—	90	—	—	115	45	71	8	152	—
1	1	1/2	1/6	80		125	35,5	155	177	144	112	—	100	—	—	130	50	80	9	168,5	—
2 3	2	1	1/2	90L		140	35,5	170	200	157	125	—	125	—	—	150	56	90	10	190	—
—	3	2	1	100L	2	160	45	195	219	180	145	—	140	—	—	175	63	100	12,5	—	3
5	5	3	2	112M		190	45	224	238	189	154	—	140	—	—	175	70	112	14	—	2
7,5 10	7,5	5	3	132S		216	45	250	273	225	180	—	140	—	—	175	89	132	16	—	3
—	10	7,5	5	132M		216	45	250	273	225	180	—	178	—	—	212	89	132	16	—	3
15 20	15	10	7,5	160M		254	50	300	334	263	218	—	210	—	—	250	108	160	18	—	3
25	20	15	10	160L		254	50	300	334	263	218	—	254	—	—	300	108	160	18	—	3
30	—	—	—	180MA		279	75	355	382	305	250	—	241	—	—	297	121	180	20	—	4
—	25 30	20	15	180MC		279	75	355	382	305	250	—	241	—	—	297	121	180	20	—	4
40	—	—	—	180LA	3	279	75	355	382	305	250	—	279	—	—	335	121	180	20	—	4
—	40	25 30	20	180LC		279	75	355	382	305	250	—	279	—	—	335	121	180	20	—	4
50 60	—	—	—	200LA		318	80	400	420	342	279	—	305	—	—	365	133	200	25	—	4
—	50 60	40 50	25 30	200LC		318	80	400	420	342	279	—	305	—	—	365	133	200	25	—	4
75	—	—	—	225SA		356	90	450	458	386	312	—	286	—	—	350	149	225	30	—	5
—	75	60	40	225SC		356	90	450	458	386	312	—	286	—	—	350	149	225	30	—	5
100	—	—	—	A2S5SA	4	406	100	500	510	479	384	—	311	—	19	425	168	250	36	—	5
—	100	75	50	A2S3SC		406	100	500	510	479	384	—	311	—	19	425	168	250	36	—	5
125	—	—	—	A2S0MA		406	100	500	510	479	384	—	349	—	28,5	480	168	250	36	—	5
—	125	100	60	A2S0MC		406	100	500	510	479	384	—	349	—	28,5	480	168	250	36	—	5
150	—	—	—	2B3S	5	457	110	560	525	610	455	305	368	110	—	445	190	280	36	—	7
—	150	125	75	2B3S	6	457	110	560	525	610	455	305	368	110	—	445	190	280	36	—	7
175	—	—	—	2B0M	5	457	110	560	525	610	455	305	419	130	—	495	190	280	36	—	7
—	175	150	100	2B0M	6	457	110	560	525	610	455	305	419	130	—	495	190	280	36	—	7
200 215	—	—	—	3155	5	508	115	615	625	610	455	305	406	115	—	490	216	315	40	—	7
—	200 215	175	125	3155	6	508	115	615	625	610	455	305	406	115	—	490	216	315	40	—	7
250 270	—	—	—	315M	5	508	115	615	625	610	455	305	457	115	—	540	216	315	40	—	7
—	250 270	200 215	150	315M	6	508	115	615	625	610	455	305	457	115	—	540	216	315	40	—	7

Note: 1. Tolerance of shaft end diameter D: ø11: ø28 : j6, ø38: ø48 : k6, ø55: ø95 : m6 2. Tolerance of shaft center height h: +0, -0,5 for 250mm and under, +0, -1 for 280mm

5. Data Subject to change without notice

IEC Dimensions

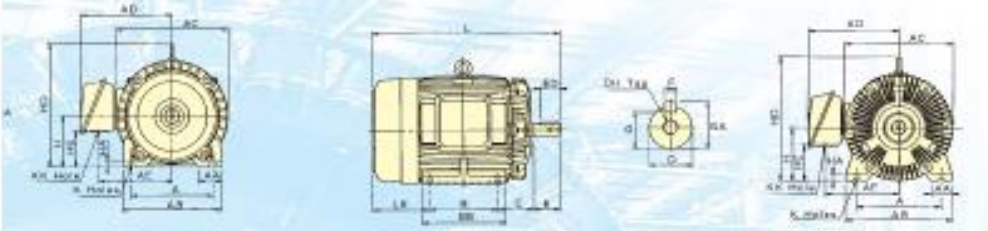


Fig.3

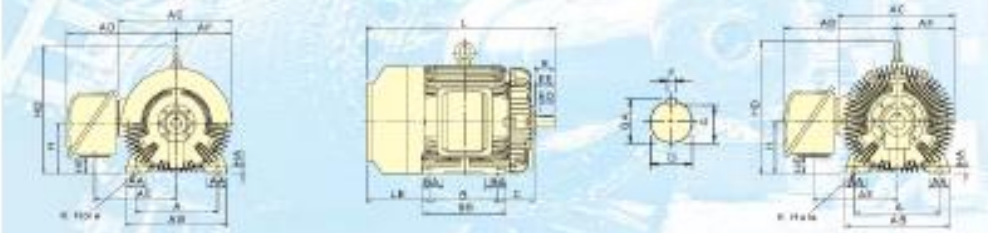


Fig.6

Dimension : mm

	HE	K	KK	L	LE	SHAFT EXTENSION								BEARINGS		APPROX WEIGHT KGS
						D	E	ED	EE	F	G	GA	DH	DRIVE END	OPP. DRIVE END	
	29	7	22	219	76	11	23	18	—	4	8.5	12.5	M4×8	*6201zz	*6201zz	8.5
	54	7	22	250.5	95.5	14	30	24	—	5	11	16	M5×10	*6202zz	*6202zz	12
	55	10	22	282.5	92.5	19	40	28	—	6	15.5	21.5	M6×12	*6204zz	*6204zz	14
	65	10	22	332.5	101.5	24	50	32	—	8	20	27	M8×16	*6205zz	*6205zz	24.5
	71	12	28	374.5	111.5	28	60	40	—	8	24	31	M10×20	*6206zz	*6206zz	31
	83	12	28	391.5	121.5	28	60	40	—	8	24	31	M10×20	*6306zz	*6306zz	42
	83	12	35	464	145	38	80	64	—	10	33	41	M12×24	*6308zz	*6306zz	67
	83	12	35	492	145	38	80	64	—	10	33	41	M12×24	*6308zz	*6306zz	78
	108	14.5	35	606	180	42	110	80	—	12	37	45	M16×32	*6308zz	*6307zz	120
	108	14.5	35	622	180	42	110	80	—	12	37	45	M16×32	*6308zz	*6307zz	144
	118	14.5	52	672	200	48	110	80	—	14	42.5	51.5	M16×32	*6311zzC3	*6310zzC3	185
	118	14.5	52	672	200	48	110	80	—	14	42.5	51.5	M16×32	*6311zz	*6310zz	182
	118	14.5	52	710	200	55	110	80	—	16	49	59	M20×40	*6312zzC3	*6310zzC3	213
	118	14.5	52	710	200	55	110	80	—	16	49	59	M20×40	*6312zz	*6310zz	215
	128	18.5	65	770	222	55	110	80	—	16	49	59	M20×40	*6312zzC3	*6212zzC3	282
	128	18.5	65	800	222	60	140	110	—	18	53	64	M20×40	*6314zzC3	*6212zzC3	315
	153	18.5	92	796	241	55	110	80	—	16	49	59	M20×40	*6312zzC3	*6212zzC3	345
	153	18.5	92	816	241	65	140	110	—	18	58	69	M20×40	*6315zz	*6213zz	373
	139	34	92	886.5	331.5	55	110	80	—	16	49	59	M20×40	6313C3	6313C3	502
	139	34	92	926.5	331.5	75	140	110	—	20	67.5	79.5	M20×40	NU316	6313	515
	139	34	92	947.5	320.5	55	110	80	—	16	49	59	M20×40	6313C3	6313C3	508
	139	34	92	977.5	320.5	75	140	110	—	20	67.5	79.5	M20×40	NU316	6313	520
	91	34	—	1013	344	55	110	80	104	16	49	59	—	6314C3	6314C3	700
	91	34	—	1079	344	85	170	140	157	22	76	90	—	NU320C3	6316	720 / 730 / 780
	91	34	—	1082	343	55	110	80	104	16	49	59	—	6314C3	6314C3	750
	91	34	—	1122	343	85	170	140	157	22	76	90	—	NU320C3	6316	830 / 830 / 860
	126	28	—	1101	369	55	110	80	104	16	49	59	—	6314C3	6314C3	800
	126	28	—	1161	369	85	170	140	157	25	86	100	—	NU320C3	6316	950 / 950 / 920
	126	28	—	1152	369	55	110	80	104	16	49	59	—	6314C3	6314C3	1020
	126	28	—	1212	369	85	170	140	157	25	86	100	—	NU320C3	6316	1040 / 1050 / 1000

and above, 3, Grease Pre-Packed shielded Rolling Bearings 4, Frequency 50Hz and 50/60Hz of center height 250mm and under are suitable for CE marking

FLANGE TYPE

Totally Enclosed Fan Cooled, Squirrel Cage Rotor.

Fig.1

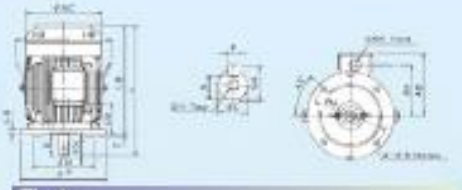


Fig.2

Fig.3

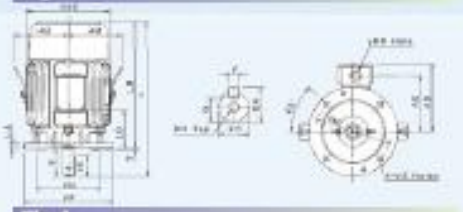


Fig.4

Fig.5

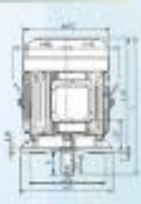


Fig.6

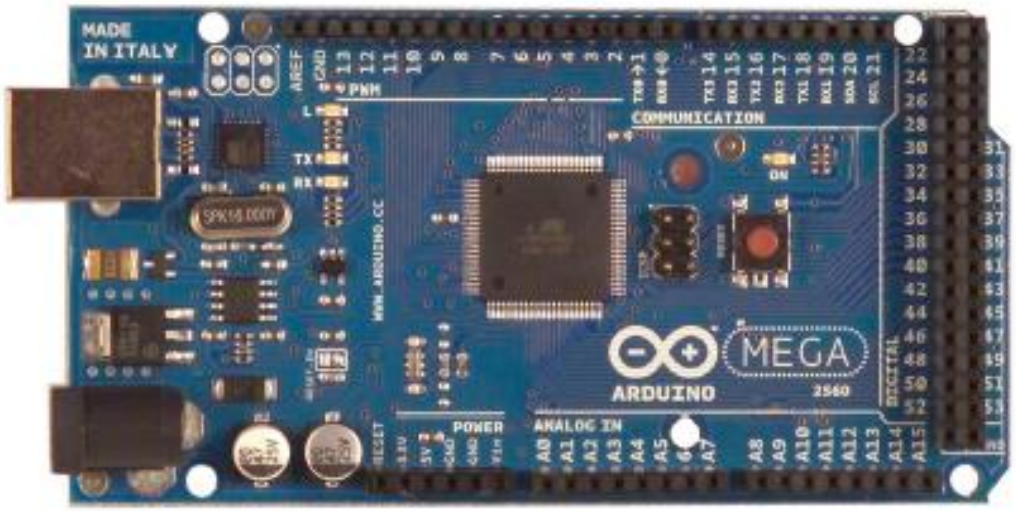
Fig.7

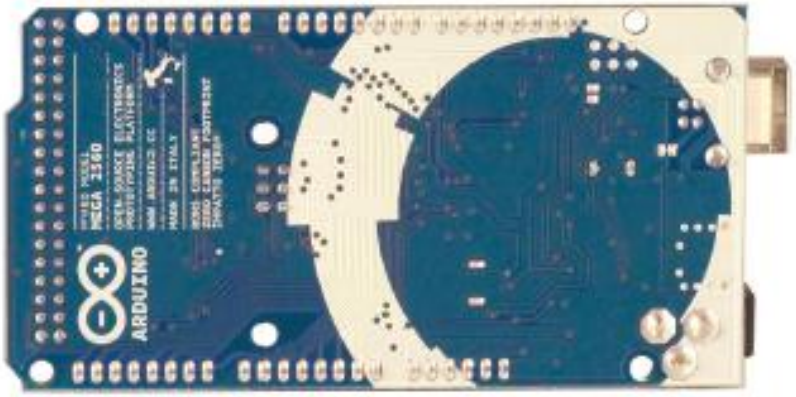
Dimension : mm

OUTPUT(HP)				FRAME NO.	FRAME NO.	AC	AD	AE	HB	KK	L	LA	LB	LD	M	N	P	S	T	SHAFT EXTENSION						
2P	4P	6P	8P																	D	E	ED	F	G	GA	DH
1/4	1/4	—	—	56	1	140	140	90	—	20	249	12	205	76	130	116	160	10	3.5	11	23	16	4	8.5	12.5	M 4 x 8
1/2	1/2	1/4	—	71	1	152	133	103	—	22	277.5	12	247.5	82	130	116	160	10	3.5	14	30	24	5	11	16	M 5 X 10
1	1	1/2	1/4	80	2	177	144	112	—	22	282	12	242	80	165	136	200	12	3.5	19	40	35	6	15.5	21.5	M 6 X 15
2 3	2	1	1/2	90L	3	200	157	125	—	22	321.5	12	281.5	113	165	136	200	12	3.5	24	50	32	8	20	27	M 8 X 19
—	3	2	1	100L	3	210	160	140	140	26	374.5	16	314.5	88	215	180	250	14.5	4	28	60	40	8	24	31	M10 X 22
5	5	3	2	112M	3	245	185	154	140	26	401	16	321	106	215	180	250	14.5	4	28	60	40	8	24	31	M10 X 22
7.5 10	7.5	5	3	132S	4	270	224	180	160	26	454	20	374	97	265	230	300	14.5	4	38	80	64	10	33	41	M12 X 28
—	10	7.5	5	132M	4	270	224	180	160	26	492	20	412	115	265	230	300	14.5	4	38	80	64	10	33	41	M12 X 28
15 20	15	10	7.5	160M	4	334	263	218	217	35	606	20	486	151	300	250	350	16.5	5	42	110	80	12	37	45	M16 X 36
25	20	15	10	160L	4	334	263	218	217	35	692	20	542	173	300	250	350	16.5	5	42	110	80	12	37	45	M16 X 36
30	—	—	—	180MA	5	382	305	256	241	52	672	20	562	170.5	350	300	400	16.5	5	48	110	80	14	42.5	51.5	M16 X 36
—	25 30	20	15	180MC	5	382	305	256	241	52	672	20	562	170.5	350	300	400	16.5	5	48	110	80	14	42.5	51.5	M16 X 36
40	—	—	—	180LA	5	382	305	256	241	52	710	20	580	189.5	350	300	400	16.5	5	55	110	80	16	49	59	M20 X 42
—	40	25 30	20	180LC	5	382	305	256	241	52	710	20	580	189.5	350	300	400	16.5	5	55	110	80	16	49	59	M20 X 42
50 60	—	—	—	200LA	6	420	342	278	260	65	770	20	680	194.5	400	350	450	16.5	5	55	110	80	16	49	59	M20 X 42
—	50 60	40 50	25 30	200LC	6	420	342	278	260	65	800	20	680	194.5	400	350	450	16.5	5	60	140	110	18	53	64	M20 X 42
75	—	—	—	225SA	6	458	384	312	298	60	788	22	678	190	500	450	550	16.5	5	55	110	80	16	49	59	M20 X 42
—	75	60	40	225SC	6	458	384	312	298	60	816	22	678	190	500	450	550	16.5	5	60	140	110	18	53	64	M20 X 42
100	—	—	—	A250SA	7	510	479	364	312	60	930.5	22	780.5	201.5	500	450	550	16.5	5	65	110	80	16	49	59	M20 X 42
—	100	75	50	A250SC	7	510	479	364	312	60	930.5	22	780.5	201.5	500	450	550	16.5	5	75	140	110	20	67.5	79.5	M20 X 42
125	—	—	—	A280MA	7	510	479	364	312	60	947.5	22	837.5	230	500	450	550	16.5	5	55	110	80	16	49	59	M20 X 42
—	125	100	60	A280MC	7	510	479	364	312	60	977.5	22	837.5	230	500	450	550	16.5	5	75	140	110	20	67.5	79.5	M20 X 42

Note: 1. Tolerance of shaft end diameter D : ø11 ~ ø28 : js, ø28 ~ ø48 : k6, ø55 ~ ø75 : m6 2. Tolerance of N: h7 3. Data Subject to change without notice

Arduino Mega 2560





Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Schematic & Reference Design

EAGLE files: arduino-mega2560-reference-design.zip

Schematic: arduino-mega2560-schematic.pdf

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Power

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.

- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.

- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.

- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.

- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

- **I^2C: 20 (SDA) and 21 (SCL).** Support I^2C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I^2C pins on the Duemilanove or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega2560's digital pins.

The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. For SPI communication, use the SPI library.

Programming

The Arduino Mega can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The ATmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega8U2 firmware source code is available in the Arduino repository. The ATmega8U2 is loaded with a DFU bootloader, which can be activated by connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can

have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega2560 contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

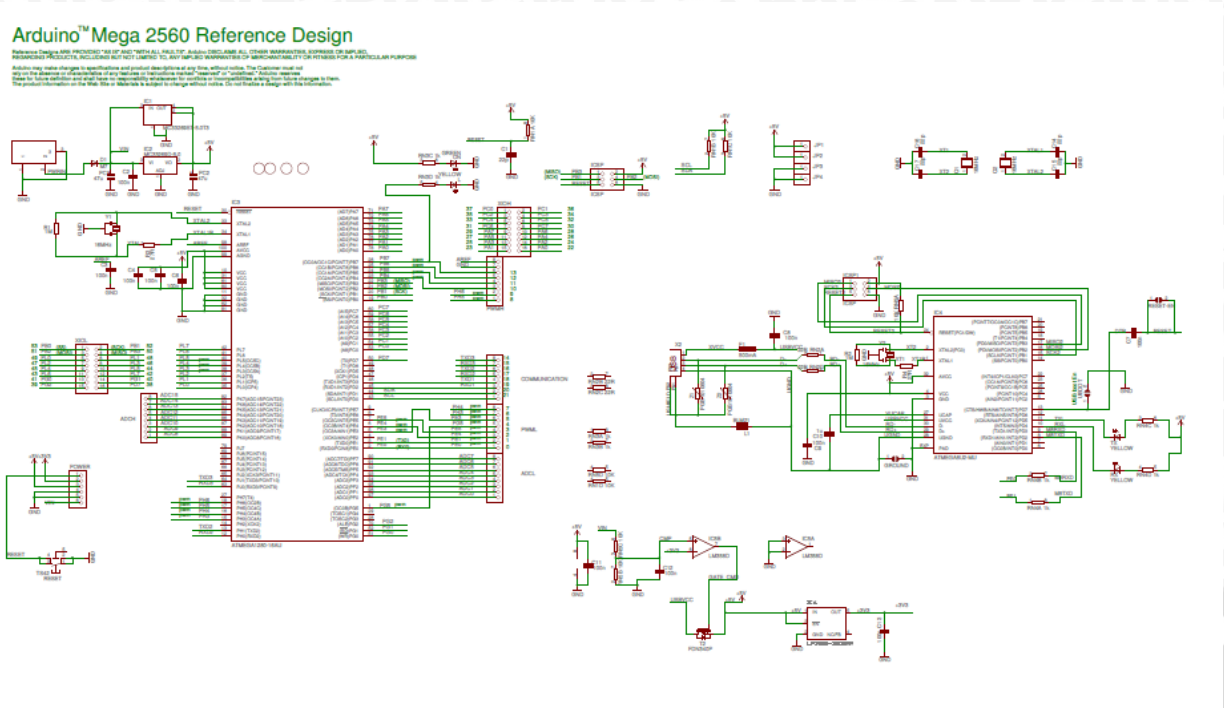
USB Overcurrent Protection

The Arduino Mega2560 has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility

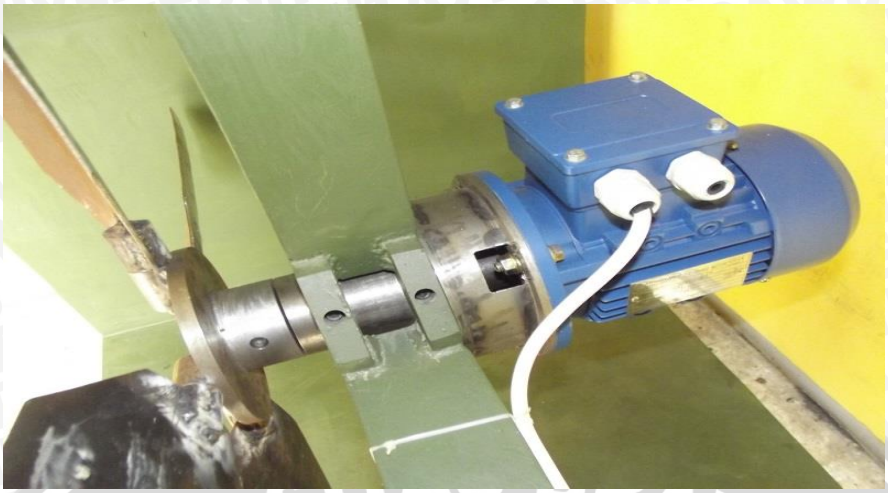
The maximum length and width of the Mega2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega2560 and Duemilanove / Diecimila. *Please note that I^2C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).*


Arduino™ Mega 2560 Reference Design

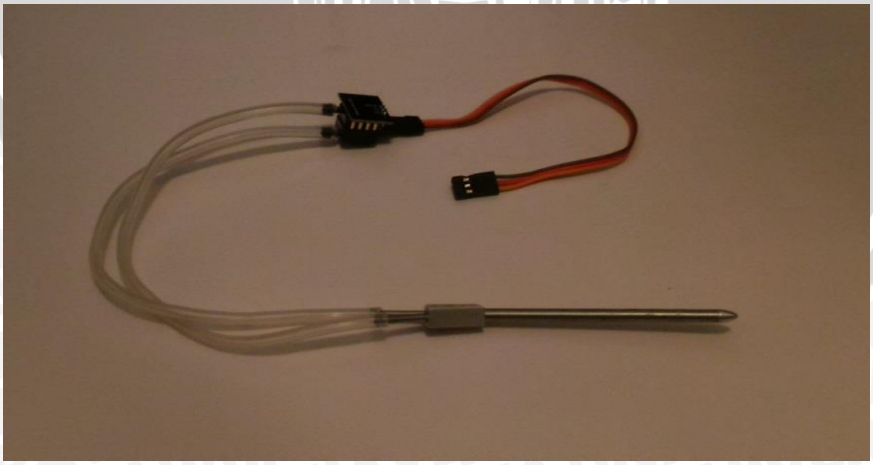
LAMPIRAN 2

FOTO ALAT



Motor Induksi

Alat Tampak Depan



Sensor Air Flow

LAMPIRAN 3

LISTING PROGRAM

```
/*****************************************************************
 * TUGAS AKHIR
 * JUDUL:
* SISTEM PENGENDALIAN KECEPATAN ALIRAN UDARA PADA *WIND TUNNEL* DENGAN UMPAN BALIK KECEPATAN ALIRAN UDARA MENGGUNAKAN KONTROLER PID
 * NAMA: RANDY MUHAMMAD
 * NIM : 105060300111047
 *****************************************************************/

#include <PID_v1.h>
#include "Arduino.h"
#include <Wprogram.h>
#else
#endif

// Mendefinisikan kanal adc setpoint Input dan Output
double Setpoint, Input, Output;
double Kp, Ki, Kd;
double Error=0, Error1=0, sError=0, dError=0, lError=0;
double now=0, dTime=0, lTime=0;


// setting parameter PID
Setpoint = 10;
Kp = 4.67;
Ki = 7.78;
Kd = 0.7005;
PID myPID(&Input, &Output, &Setpoint,Kp,Ki,Kd, DIRECT);

  //turn the PID on
```

```
 myPID.SetMode(AUTOMATIC);
}

void loop()
{
 // baca feedback
 Input = analogRead(0);
 // baca setpoint
 Setpoint = analogRead(1);
 myPID.Compute();
 // keluarkan hasilnya
 analogWrite(3,Output);
}

// deklarasi sensor //
int sensorpin = A1; //pin sensor = A1
int sensorvalue; // nilai sensor
int banyakData;
float v_sen; // tegangan sensor

void loop()
{
 {
   sensorvalue = analogRead(sensorpin);
   v_sen = sensorvalue*(5.0/1023.0);
   airflow = 2.0*v_sen;

   banyakData++;

   Serial.println(rpm);
```

```
  banyakData=0;
 }
  delay(20);

//perhitungan error//
 now=millis();
 if(lTime!=0){dTime =(double)(now-lTime);}

 sensors.requestAirFlow();
 error = setpoint-airflow;
 sError =(sError+error);          //jumlah error = jumlaherror+error
 dError = (error-lError);         // deferensial error = error-last error
/*perhitungan*/

//------- kontrol PID -------//
//Rumus pid

 output = (Kp*Error) + ((Ki*sError)*(dTime/1000))  +((Kd*dError)/(dTime/1000));
  if(PID<=0){PID=0;}
  else{PID+=50;}
  PID=(int)PID;

  lError = Error;
  lTime = now;
}

//sinyal PID perintah untuk aktuator

    if (output > 0) // melebihi setpoint
```

```
    {
     analog.write(0);   //putaran motor berkurang

    }

    else if (output < 0)// melebihi setpoint
    {
     analog.write (400);  //putaran motor bertambah
    }

    else if (output=0)
    {
    analog.write(150);
    }

}
```

```
#ifndef PID_v1_h
#define PID_v1_h
#define LIBRARY_VERSION	1.1.1

class PID
{
  public:

  //Constants used in some of the functions below
  #define AUTOMATIC	1
  #define MANUAL	0
  #define DIRECT  0
  #define REVERSE  1

  //commonly used functions
**************************************************************************
   PID(double*, double*, double*,        // * constructor.  links the PID to the Input, Output, and
      double, double, double, int);     //   Setpoint.  Initial tuning parameters are also set here

   void SetMode(int Mode);               // * sets PID to either Manual (0) or Auto (non-0)

   bool Compute();                       // * performs the PID calculation.  it should be
                                         //   called every time loop() cycles. ON/OFF and
                                         //   calculation frequency can be set using SetMode
                                         //   SetSampleTime respectively

   void SetOutputLimits(double, double); //clamps the output to a specific range. 0-255 by default, but
                                         //it's likely the user will want to change this depending on
                                         //the application
```

```
  //available but not commonly used functions
*************************************************************
   void SetTunings(double, double,      // * While most users will set the tunings once in the
                   double);             //   constructor, this function gives the user the option
                                        //   of changing tunings during runtime for Adaptive control
  void SetControllerDirection(int);  // * Sets the Direction, or "Action" of the controller. DIRECT
                                     //   means the output will increase when error is positive. REVERSE
                                     //   means the opposite.  it's very unlikely that this will be needed
                                     //   once it is set in the constructor.
   void SetSampleTime(int);          // * sets the frequency, in Milliseconds, with which
                                     //   the PID calculation is performed.  default is 100

  //Display functions ****************************************************************
          double GetKp();            // These functions query the pid for interal values.
          double GetKi();            //  they were created mainly for the pid front-end,
          double GetKd();            // where it's important to know what is actually
          int GetMode();             //  inside the PID.
          int GetDirection();        //

  private:
          void Initialize();

          double dispKp;             // * we'll hold on to the tuning parameters in user-entered
          double dispKi;             //   format for display purposes
          double dispKd;             //

          double kp;                 // * (P)roportional Tuning Parameter
    double ki;                       // * (I)ntegral Tuning Parameter
```

```
  double kd;                // * (D)erivative Tuning Parameter

        int controllerDirection;

  double *myInput;          // * Pointers to the Input, Output, and Setpoint variables
  double *myOutput;         //   This creates a hard link between the variables and the
  double *mySetpoint;       //   PID, freeing the user from having to constantly tell us
                            //   what these values are.  with pointers we'll just know.

        unsigned long lastTime;
        double ITerm, lastInput;

        unsigned long SampleTime;
        double outMin, outMax;
        bool inAuto;
};
#endif
```

```
/**********************************************************************************************
 * Arduino PID Library - Version 1.1.1
 * by Brett Beauregard <br3ttb@gmail.com> brettbeauregard.com
 *
 * This Library is licensed under a GPLv3 License
 **********************************************************************************************/

#if ARDUINO >= 100
  #include "Arduino.h"
#else
  #include "WProgram.h"
#endif

#include <PID_v1.h>

/*Constructor (...)*********************************************************
 *    The parameters specified here are those for for which we can't set up
 *    reliable defaults, so we need to have the user set them.
 *********************************************************************************/
PID::PID(double* Input, double* Output, double* Setpoint,
         double Kp, double Ki, double Kd, int ControllerDirection)
{

    myOutput = Output;
    myInput = Input;
    mySetpoint = Setpoint;
            inAuto = false;
```

```
    PID::SetOutputLimits(0, 255);          //default output limit corresponds to
                                           //the arduino pwm limits

  SampleTime = 100;                        //default Controller Sample Time is 0.1 seconds

  PID::SetControllerDirection(ControllerDirection);
  PID::SetTunings(Kp, Ki, Kd);

  lastTime = millis()-SampleTime;
}
/* Compute() **********************************************************
 *   This, as they say, is where the magic happens.  this function should be called
 *  every time "void loop()" executes.  the function will decide for itself whether a new
 *  pid Output needs to be computed.  returns true when the output is computed,
 *  false when nothing has been done.
**********************************************************************/
bool PID::Compute()
{
  if(!inAuto) return false;
  unsigned long now = millis();
  unsigned long timeChange = (now - lastTime);
  if(timeChange>=SampleTime)
  {
   /*Compute all the working error variables*/
        double input = *myInput;
   double error = *mySetpoint - input;
   ITerm+= (ki * error);
   if(ITerm > outMax) ITerm= outMax;
   else if(ITerm < outMin) ITerm= outMin;
   double dInput = (input - lastInput);
```

```
   /*Compute PID Output*/
   double output = kp * error + ITerm- kd * dInput;

         if(output > outMax) output = outMax;
   else if(output < outMin) output = outMin;
         *myOutput = output;

   /*Remember some variables for next time*/
   lastInput = input;
   lastTime = now;
         return true;
  }
  else return false;
}
/* SetTunings(...)*************************************************************
 * This function allows the controller's dynamic performance to be adjusted.
 * it's called automatically from the constructor, but tunings can also
 * be adjusted on the fly during normal operation
 ******************************************************************************/
void PID::SetTunings(double Kp, double Ki, double Kd)
{
  if (Kp<0 || Ki<0 || Kd<0) return;

  dispKp = Kp; dispKi = Ki; dispKd = Kd;

  double SampleTimeInSec = ((double)SampleTime)/1000;
  kp = Kp;
  ki = Ki * SampleTimeInSec;
  kd = Kd / SampleTimeInSec;
```

```
 if(controllerDirection ==REVERSE)
  {
    kp = (0 - kp);
    ki = (0 - ki);
    kd = (0 - kd);
  }
}
/* SetSampleTime(...) *********************************************************
 * sets the period, in Milliseconds, at which the calculation is performed
 ******************************************************************************/
void PID::SetSampleTime(int NewSampleTime)
{
  if (NewSampleTime > 0)
  {
    double ratio  = (double)NewSampleTime
                    / (double)SampleTime;
    ki *= ratio;
    kd /= ratio;
    SampleTime = (unsigned long)NewSampleTime;
  }
}

/* SetOutputLimits(...)****************************************************
 *     This function will be used far more often than SetInputLimits.  while
 *  the input to the controller will generally be in the 0-1023 range (which is
 *  the default already,)  the output will be a little different.  maybe they'll
 *  be doing a time window and will need 0-8000 or something.  or maybe they'll
 *  want to clamp it from 0-125.  who knows.  at any rate, that can all be done
 *  here.
```

```
 **********************************************************************/
void PID::SetOutputLimits(double Min, double Max)
{
  if(Min >= Max) return;
  outMin = Min;
  outMax = Max;

  if(inAuto)
  {
          if(*myOutput > outMax) *myOutput = outMax;
          else if(*myOutput < outMin) *myOutput = outMin;

          if(ITerm > outMax) ITerm= outMax;
          else if(ITerm < outMin) ITerm= outMin;
  }
}

/* SetMode(...)****************************************************************
 * Allows the controller Mode to be set to manual (0) or Automatic (non-zero)
 * when the transition from manual to auto occurs, the controller is
 * automatically initialized
 ******************************************************************************/
void PID::SetMode(int Mode)
{
   bool newAuto = (Mode == AUTOMATIC);
   if(newAuto == !inAuto)
   {  /*we just went from manual to auto*/
      PID::Initialize();
   }
   inAuto = newAuto}
```