



TUGAS AKHIR - TE145561

TELEMETERING PENGUKURAN KECEPATAN PADA MOTOR DC BERBEBAN

Anandita Ghasani
NRP 10311500010037

Dosen Pembimbing
Ir. Josaphat Pramudijanto, M.Eng.

Program Studi Elektro Industri
Departemen Teknik Elektro Otomasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018



TUGAS AKHIR - TE145561

**TELEMETERING PENGUKURAN KECEPATAN PADA MOTOR
DC BERBEBAN**

Anandita Ghasani
NRP 10311500010037

Dosen Pembimbing
Ir. Josaphat Pramudijanto, M.Eng.

Program Studi Elektro Industri
Departemen Teknik Elektro Otomasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018



ITS
Institut
Teknologi
Sepuluh Nopember

FINAL PROJECT - TE145561

***TELEMETERING OF SPEED MEASUREMENT
OF LOADED DC MOTOR***

Anandita Ghasani
NRP 10311500010037

Supervisor
Ir. Josaphat Pramudijanto, M.Eng.

*Electrical Industry Study Program
Electrical and Automation Engineering Department
Vocational Faculty
Institut Teknologi Sepuluh Nopember
Surabaya 2018*

PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "**Telemetry Pengukuran Kecepatan Pada Motor DC Berbeban**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Juli 2018



Anandita Ghasani
NRP 10311500010037

-- *Halaman ini sengaja dikosongkan* --

**TELEMETERING PENGUKURAN KECEPATAN PADA
MOTOR DC BERBEBAN**

TUGAS AKHIR

**Diajukan Guna Memenuhi Sebagian Persyaratan
Memperoleh Gelar Ahli Madya Teknik
Pada
Program Studi Elektro Industri
Departemen Teknik Elektro Otomasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember**

Menyetujui :

Dosen Pembimbing,



**Jr. Josaphat Pramudijanto, M.Eng.
NIP. 19621005 199003 1 003**

**SURABAYA
AGUSTUS, 2018**

-- *Halaman ini sengaja dikosongkan* --

TELEMETERING PENGUKURAN KECEPATAN PADA MOTOR DC BERBEBAN

Nama Mahasiswa : Anandita Ghasani
Nrp : 10311500010037
Pembimbing : Ir. Josaphat Pramudijanto. M.Eng.
NIP : 19621005 199003 1 003

ABSTRAK

Dalam pelaksanaan praktikum yang terkait tentang performansi motor DC di mana informasi mengenai besaran kecepatan, tegangan input, dan beban sangat diperlukan. Besaran-besaran tersebut diukur dan dicatat oleh praktikan sebagai data performansi motor DC. Untuk mempermudah asisten melakukan pengawasan saat pengambilan data tersebut maka digunakan metode telemetering dimana informasi mengenai tegangan, kecepatan motor DC, dan besarnya beban yang diberikan pada saat motor berputar dapat didapatkan dari jarak jauh.

Untuk merealisasikan pengambilan data performansi motor DC berbeban secara telemetering, maka dibuat perangkat berupa modul motor DC yang pada porosnya diberi beban rem magnetik dan juga sensor kecepatan, dengan Arduino Mega 2560 sebagai mikrokontrolernya. Untuk telemetering maka ditambahkan *Ethernet Shield* yang kemudian dihubungkan dengan *access point* menggunakan kabel RJ45. *Access point* akan mengirimkan data ke PC secara nirkabel. Hasil pengukuran modul ini diawasi dari jarak jauh dengan menggunakan *software* LabView sebagai *interface*.

Setelah dilakukan pengujian, hasilnya menunjukkan bahwa pada proses pengiriman data dari *access point* menuju PC tanpa penghalang dapat terus dilakukan tanpa ada *loss* hingga jarak 160 meter, sedangkan dengan penghalang dinding dapat terus dilakukan tanpa ada *loss* hingga jarak 20m. Pada saat tanpa beban motor dapat berputar pada kecepatan maksimal sebesar 1215 rpm dengan *error* terbesar 0,016%, sedangkan pada kecepatan maksimal motor dapat dibebani hingga 80% dengan kecepatan 315 rpm dan *error* terbesar 0,01%. Saat beban konstan 25% motor dapat berputar dengan kecepatan maksimal 1065 rpm, dan saat beban konstan 100% motor berputar maksimal 450 rpm.

Kata Kunci : Motor DC, Telemetering, LabView, Ethernet.

-- *Halaman ini sengaja dikosongkan* --

TELEMETERING OF SPEED OF LOADED DC MOTOR

Student's Name : Anandita Ghasani
Registration Number : 10311500010037
Supervisor : Ir. Josaphat Pramudijanto. M.Eng.
ID : 19621005 199003 1 003

ABSTRACT

In practice implementation of DC motor performance where information about speed, input voltage, and load is required. The quantities are measured and recorded by the practitioner as DC motor performance data. To facilitate the assistant to supervise during the data retrieval, telemetering method is used where the information about voltage, DC motor speed, and the amount of load given at the time of motor rotation can be obtained from long distance.

To realize the performance data from a loaded DC motor with telemetering method, then created a device in the form of a DC motor module on the axis given the magnetic brake load and also the speed sensor, with Arduino Mega 2560 as microcontroller. For telemetering then added Ethernet Shield which is then connected with access point using RJ45 cable. Access point will send data to PC wirelessly. The measurement results of this module are monitored remotely by using LabView software as an interface.

After the test, the process of sending data from the access point to the PC without a barrier in the open space can continue to be done without any loss up to a distance of 160 meters, while with a wall barrier in a closed room can continue without loss up to 20m. While the results show that in a no-load state, the motor can rotate at a maximum speed of 1215 rpm with the largest error of 0.016%, while at maximum speed the motor can be loaded up to 80% with a speed of 315 rpm and the biggest error of 0.01%. When the load is constant at 25%, the motor can rotate with a maximum speed of 1065 rpm, and When the load is constant at 100% the motor can rotate with a maximum speed of 450 rpm.

Keywords : *DC Motor, Telemetering, LabView, Ethernet*

-- *Halaman ini sengaja dikosongkan* --

KATA PENGANTAR

Puji syukur penulis panjatkan kehadiran Allah SWT yang selalu memberikan rahmat dan hidayah-Nya sehingga Tugas Akhir ini dapat terselesaikan dengan baik. Shalawat serta salam semoga selalu dilimpahkan kepada Rasulullah Muhammad SAW, keluarga, sahabat, dan umat muslim yang senantiasa meneladani beliau.

Tugas Akhir ini disusun untuk memenuhi sebagian persyaratan guna menyelesaikan pendidikan Diploma-3 pada Program Studi Teknik Elektro Industri, Departemen Teknik Elektro Otomasi, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember Surabaya dengan judul:

“TELEMETERING PENGUKURAN KECEPATAN PADA MOTOR DC BERBEBAN”

Tugas Akhir ini merupakan sebagian syarat untuk menyelesaikan mata kuliah dan memperoleh nilai pada Tugas Akhir.

Dengan selesainya Tugas Akhir ini penulis menyampaikan terima kasih sebesar-besarnya kepada:

1. Orang Tua atas limpahan doa, kasih sayang, dukungan dan dorongan baik berupa moril atau materil bagi penulis.
2. Bapak Joko Susila, Ir., MT. selaku Ketua Departemen Teknik Elektro Otomasi, FV-ITS.
3. Bapak Ir. Josaphat Pramudijanto, M.Eng. selaku Dosen Pembimbing.
4. Semua pihak yang telah banyak membantu untuk menyelesaikan Tugas Akhir ini yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari masih banyak kekurangan dalam Tugas Akhir ini. Kritik dan saran untuk perbaikan tugas ini sangat diperlukan. Akhir kata semoga tugas ini dapat bermanfaat bagi kita semua.

Surabaya, Juli 2018

Anandita Ghasani
10311500010037

-- *Halaman ini sengaja dikosongkan* --

DAFTAR ISI

HALAMAN JUDUL	i
LEMBAR PENGESAHAN	vii
ABSTRAK	ix
ABSTRACT	xi
KATA PENGANTAR	xiii
DAFTAR ISI	xv
DAFTAR GAMBAR	xix
DAFTAR TABEL	xxi
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Permasalahan.....	1
1.3 Batasan Masalah.....	2
1.4 Tujuan	2
1.5 Sistematika Penulisan.....	2
1.6 Relevansi	3
BAB II TEORI PENUNJANG	5
2.1 <i>Access Point</i>	5
2.2 <i>User Datagram Protocol</i>	6
2.3 Motor DC	7
2.4 Driver L298N	9
2.4.1 Fungsi <i>Pin Driver</i> Motor DC L298N	10
2.4.2 Spesifikasi Modul <i>Driver</i> Motor DC L298N	10
2.5 Potensiometer <i>Wirewound</i>	11
2.6 Sensor <i>Rotary Encoder</i>	12
2.7 Pengereman Magnetik.....	13
2.8 Mikrokontroler Arduino MEGA 2560	11
2.9 <i>Ethernet Shield</i>	16
2.10 LabView.....	17
BAB III PERANCANGAN DAN PEMBUATAN ALAT	19
3.1 Diagram Fungsional Sistem	19
3.2 Perancangan Mekanik	21
3.3 Perancangan Perangkat Keras (<i>Hardware</i>)	22
3.3.1 Koneksi Potensiometer <i>Wirewound</i>	22
3.3.2 Koneksi Sensor Kecepatan	23

3.3.3 Koneksi <i>Driver</i> Motor L298N dengan Arduino Mega 2560	23
3.3.4 Perancangan <i>Shield</i> Arduino Mega 2560	24
3.3.5 Koneksi <i>Ethernet</i> dengan Arduino Mega 2560	24
3.3.6 Koneksi <i>Ethernet</i> dengan <i>Access Point</i>	25
3.4 Perancangan Perangkat Lunak (<i>Software</i>)	26
3.4.1 Pemrograman <i>Software</i> Arduino untuk Sensor Kecepatan.....	26
3.4.2 Pemrograman <i>Software</i> Arduino untuk Pengereman	27
3.4.3 Perancangan <i>Software</i> LabView	28
BAB IV PENGUJIAN DAN ANALISA	31
4.1 Pengujian <i>Input/Output</i> Arduino Mega 2560.....	31
4.2 Pengujian Komunikasi <i>Ethernet</i>	34
4.2.1 Pengujian Koneksi <i>Access Point</i> dan PC	34
4.2.2 Pengujian pada <i>Board</i> Arduino Mega 2560	36
4.3 Pengujian Koneksi Terhadap Jarak	38
4.3.1 Pengujian Koneksi Dengan Penghalang	38
4.3.2 Pengujian Koneksi Tanpa Penghalang	41
4.4 Pengujian dengan <i>Software</i> Wireshark	45
4.5 Kalibrasi Sensor Kecepatan	47
4.6 Pengujian pada <i>Software</i> LabView	49
4.6.1 Pengujian <i>Software</i> LabView pada Motor DC Tanpa Beban.....	51
4.6.2 Pengujian <i>Software</i> LabView pada Motor DC dengan Beban Konstan 25%.....	53
4.6.3 Pengujian <i>Software</i> LabView pada Motor DC dengan Beban Konstan 50%.....	54
4.6.4 Pengujian <i>Software</i> LabView pada Motor DC dengan Beban Konstan 75%.....	55
4.6.5 Pengujian <i>Software</i> LabView pada Motor DC dengan Beban Konstan 100%.....	57
BAB V PENUTUP	59
5.1 Kesimpulan	59
5.2 Saran.....	60

DAFTAR PUSTAKA	61
LAMPIRAN A TABEL HASIL PENGUJIAN.....	A-1
LAMPIRAN B <i>LISTING PROGRAM</i>.....	B-1
LAMPIRAN C <i>DATASHEET</i>	C-1
DAFTAR RIWAYAT HIDUP	D-1

-- *Halaman ini sengaja dikosongkan* --

DAFTAR GAMBAR

Gambar 2.1	Pemanfaatan <i>Access Point</i>	5
Gambar 2.2	Prinsip Kerja Motor DC	7
Gambar 2.3	Arah Putar Motor DC Magnet Permanen	8
Gambar 2.4	Motor Berputar CW dan CCW	9
Gambar 2.5	<i>Driver</i> L298N	10
Gambar 2.6	Konstruksi Sederhana <i>Wirewound Potentiometer</i>	11
Gambar 2.7	<i>Voltage Divider</i>	11
Gambar 2.8	Potensiometer <i>Wirewound</i>	12
Gambar 2.9	<i>Rotary encoder</i>	12
Gambar 2.10	Modul IR <i>Optocoupler</i> FC-03	13
Gambar 2.11	Gaya Pengereman Yang Dihasilkan Oleh Arus Melingkar Eddy	14
Gambar 2.12	Arduino Mega 2560	15
Gambar 2.13	<i>Setting Serial Port</i>	16
Gambar 2.14	<i>Ethernet Shield</i>	17
Gambar 2.15	<i>Front Panel</i> LabView	18
Gambar 2.16	Blok Diagram LabView	18
Gambar 3.1	Blok Diagram Perancangan	19
Gambar 3.2	<i>Wiring</i> Keseluruhan	20
Gambar 3.3	Desain Modul Tampak Samping	21
Gambar 3.4	Desain Modul Tampak Atas	21
Gambar 3.5	Skema Rangkaian Potensiometer <i>Wirewound</i>	22
Gambar 3.6	Skema Rangkaian Modul IR <i>Optocoupler</i> FC-03	23
Gambar 3.7	Skema Rangkaian Modul L298N	23
Gambar 3.8	Skema Rangkaian <i>Shield</i>	24
Gambar 3.9	Koneksi <i>Ethernet Shield</i> Pada Arduino	25
Gambar 3.10	Koneksi <i>Ethernet Shield</i> Pada <i>Access Point</i>	25
Gambar 3.11	<i>Flowchart</i> Sensor Kecepatan pada Arduino	27
Gambar 3.12	<i>Flowchart</i> Pengereman pada Arduino	28
Gambar 3.13	<i>Flowchart</i> Arduino LabView	29
Gambar 3.14	<i>Front Panel</i> LabView	30
Gambar 4.1	Program Pengujian I/O Arduino <i>Active High</i>	31
Gambar 4.2	Skema Pengujian I/O Arduino	32
Gambar 4.3	Program Pengujian I/O Arduino <i>Active Low</i>	33
Gambar 4.4	Skema Pengujian I/O Arduino	33
Gambar 4.5	<i>Setting IP Address Access Point</i> Pada Laptop	35
Gambar 4.6	<i>Command Prompt</i> dengan <i>IP Address Access Point</i>	35
Gambar 4.7	Program <i>Setting IP</i> Arduino	36
Gambar 4.8	<i>Setting IP address</i> untuk board Arduino	37

Gambar 4.9	Tes Ping Pada <i>IP Address Board</i> Arduino	37
Gambar 4.10	Ilustrasi Pengujian Koneksi Dengan Penghalang	38
Gambar 4.11	Tes Ping 5 Meter Dengan Penghalang	39
Gambar 4.12	Tes Ping 10 Meter Dengan Penghalang	39
Gambar 4.13	Tes Ping 17 Meter Dengan Penghalang	40
Gambar 4.14	Tes Ping 20 Meter Dengan Penghalang	40
Gambar 4.15	Ilustrasi Pengujian Koneksi Tanpa Penghalang.....	41
Gambar 4.16	Tes Ping 5 Meter Tanpa Penghalang.....	42
Gambar 4.17	Tes Ping 15 Meter Tanpa Penghalang.....	42
Gambar 4.18	Tes Ping 30 Meter Tanpa Penghalang.....	43
Gambar 4.19	Tes Ping 163 Meter Tanpa Penghalang.....	43
Gambar 4.20	Jarak PC dan <i>Access Point</i> Pada <i>Google Maps</i>	44
Gambar 4.21	Tampilan Wireshark	45
Gambar 4.22	Program Mengirimkan 18.000 Data.....	46
Gambar 4.23	<i>Throughput</i> Pada Wireshark.....	46
Gambar 4.24	Tampilan LabView Motor Diberi Beban	47
Gambar 4.25	Kalibrasi Sensor Kecepatan.....	48
Gambar 4.26	Potongan Program Membatasi PWM Beban.....	49
Gambar 4.27	Pengukuran Kecepatan Menggunakan Tachometer ..	49
Gambar 4.28	Tampilan LabView Motor Belum Diaktifkan	50
Gambar 4.29	Gabungan Respon Kecepatan.....	50
Gambar 4.30	Tampilan LabView Motor Tanpa Beban.....	51
Gambar 4.31	Pengujian Kecepatan Motor Tanpa Beban	52
Gambar 4.32	Pengukuran Kecepatan Menggunakan Tachometer ..	52
Gambar 4.33	Pengereman Konstan 25%	53
Gambar 4.34	Pengujian Kecepatan Beban Konstan 25%	53
Gambar 4.35	Pengereman Konstan 50%	54
Gambar 4.36	Pengujian Kecepatan Beban Konstan 50%	55
Gambar 4.37	Pengereman Konstan 75%	56
Gambar 4.38	Pengujian Kecepatan Beban Konstan 75%	56
Gambar 4.39	Pengereman Konstan 100%	57
Gambar 4.40	Pengujian Kecepatan Beban Konstan 100%	58

DAFTAR TABEL

Tabel 2.1	<i>Port yang Digunakan UDP</i>	6
Tabel 2.2	Spesifikasi Modul IR <i>Optocoupler FC-03</i>	13
Tabel 2.3	Deskripsi Arduino Mega 2560	14
Tabel 4.1	Hasil Pengukuran Per <i>Pin Saat Active High</i>	32
Tabel 4.2	Hasil Pengukuran Per <i>Pin Saat Active Low</i>	34
Tabel 4.3	Pengujian Koneksi Dengan Penghalang.....	41
Tabel 4.4	Pengujian Koneksi Tanpa Penghalang	44
Tabel 4.5	Jumlah Data yang Diterima Wireshark	A1
Tabel 4.5	Kalibrasi Sensor Kecepatan.....	A2
Tabel 4.7	Pengujian Kecepatan Motor Tanpa Pengereman pada LabView	A2
Tabel 4.8	Pengujian Kecepatan Motor Dengan Beban Konstan 25%	A3
Tabel 4.9	Pengujian Kecepatan Motor Dengan Beban Konstan 50%	A3
Tabel 4.10	Pengujian Kecepatan Motor Dengan Beban Konstan 75%	A4
Tabel 4.11	Pengujian Kecepatan Motor Dengan Beban Konstan 100%	A4

-- *Halaman ini sengaja dikosongkan* --

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam proses pembelajaran, terutama pada kegiatan praktikum mengenai karakteristik motor DC, di mana informasi yang dibutuhkan adalah mengenai tegangan input, kecepatan motor DC, dan besarnya beban yang diberikan ke motor DC. Pelaksanaan praktikum yang ada selama ini, setelah praktikan merangkai perangkat praktikum kemudian diikuti pencatatan dan pengambilan data hasil pengamatan. Saat pencatatan data tersebut sering kali terjadi kesalahan oleh praktikan. Demikian juga saat praktikan mengambil data, asisten perlu mendatangi dan memeriksa apakah data yang diambil praktikan sudah tepat. Namun kegiatan praktikum di laboratorium dilakukan oleh banyak praktikan dalam satu sesi sehingga pada proses pemeriksaan oleh asisten, untuk menghindari kekeliruan pengambilan data diperlukan *record* data agar saat asisten memeriksa laporan pengambilan data, data tersebut dapat dipastikan *valid*. Selain itu juga untuk memudahkan mekanisme pengawasan oleh asisten maka timbul gagasan untuk mengaplikasikan metode telemetering ke modul pengukuran kecepatan pada motor DC berbeban.

Telemetering dapat diartikan sebagai suatu pengukuran di suatu perangkat dan mengirimkan data hasil pengukuran tersebut ke penerima yang jaraknya jauh. Penerapan fungsi telemetering dapat dimanfaatkan untuk memantau kecepatan motor DC dan beban pada motor DC yang diberikan oleh praktikan. Pemantauan kecepatan dan beban pada motor DC dengan metode telemetering diperlukan untuk mengetahui kecepatan dan beban pada motor DC dari jarak jauh secara *realtime*.

Dalam Tugas Akhir ini, akan dirancang sistem yang mampu memantau kecepatan motor DC. Data pengendalian kecepatan dan beban motor DC yang tersimpan di arduino akan ditampilkan ke komputer dengan menggunakan jaringan komunikasi *Ethernet*. Dengan sistem ini memungkinkan pemantauan kecepatan motor DC dari jarak jauh.

1.2 Permasalahan

Permasalahan yang diangkat pada Tugas Akhir ini adalah belum tersedianya mekanisme telemetering besaran kecepatan dan tegangan yang merepresentasikan beban motor DC saat berputar.

1.3 Batasan Masalah

Dari perumusan masalah di atas, maka batasan masalah dari Tugas Akhir ini adalah :

1. Sensor yang digunakan sebagai pembaca kecepatan motor DC adalah *optocoupler* dan *encoderdisk*.
2. Pengukuran yang dilakukan hanya terbatas pada pengukuran kecepatan dan beban motor DC yang kemudian ditampilkan pada *personal computer* pada *software* LabView melalui komunikasi *Ethernet* untuk dimonitor.

Dengan adanya batasan masalah ini diharapkan hasil akhir atau tujuan dari Tugas Akhir ini dapat dicapai dengan baik.

1.4 Tujuan

Pembuatan Telemetering Pengukuran Kecepatan pada Motor DC Berbeban bertujuan untuk:

1. Merancang komunikasi untuk menampilkan data pengukuran kecepatan pada motor DC yang tersimpan di dalam arduino hingga dapat ditampilkan pada komputer.
2. Membuat sebuah *Human Machine Interface* pada *software* LabView yang dapat memonitor kecepatan motor DC berbeban.
3. Membuat objek berupa motor DC berbeban yang digunakan untuk telemetering.

1.5 Sistematika Laporan

Pembahasan Tugas Akhir ini akan dibagi menjadi lima bab dengan sistematika sebagai berikut:

Bab I Pendahuluan

Bab ini meliputi latar belakang, permasalahan, tujuan penelitian, sistematika laporan, dan relevansi.

Bab II Teori Penunjang

Bab ini menjelaskan tentang tinjauan pustaka, konsep dari *access point*, UDP, motor DC, *driver* L298N, potensiometer *wirewound*, sensor *rotary encoder*, rem magnetik, Arduino Mega 2560, *Ethernet Shield*, dan LabView.

Bab III Perancangan dan Pembuatan Alat

Bab ini membahas perancangan sistem *hardware* maupun *software* pada Telemetering Pengukuran

Kecepatan pada Motor DC Berbeban berdasarkan teori penunjang pada Bab II.

Bab IV Pengujian dan Analisa

Bab ini membahas tentang pengukuran, pengujian, serta analisa terhadap prinsip kerja dan proses dari suatu alat yang dibuat.

Bab V Penutup

Bab ini menjelaskan tentang kesimpulan dari Tugas Akhir dan saran – saran untuk pengembangan alat ini lebih lanjut.

1.6 Relevansi

Tugas Akhir ini diharapkan dapat dimanfaatkan untuk memudahkan pengambilan informasi mengenai kecepatan motor DC. Sehingga pengambilan data kecepatan dapat langsung dilakukan dengan melihat pada tampilan komputer. Selain itu pemanfaatan metode telemetering ini juga dapat dikaitkan dengan pengaplikasiannya pada kebutuhan lain yang memerlukan pengamatan dan pengambilan data dari jarak jauh.

-- *Halaman ini sengaja dikosongkan* --

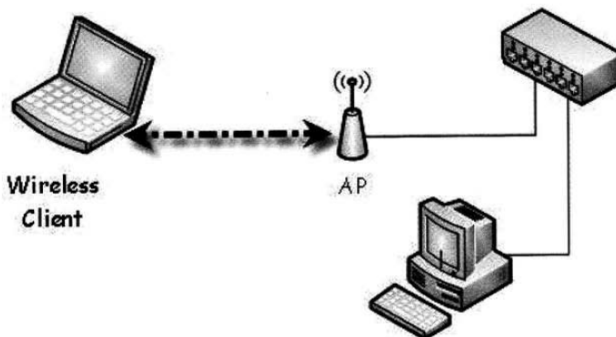
BAB II

TEORI PENUNJANG

Pada Bab II ini akan dijelaskan mengenai teori-teori dasar yang menunjang dan berhubungan dalam pengerjaan Tugas Akhir ini. Teori dasar ini diharapkan mampu membantu dalam pengerjaan Tugas Akhir dan dapat dijadikan referensi nantinya.

2.1 Access Point [1]

Inti dari sebuah jaringan *wireless* modus *infrastructure* adalah penggunaan AP atau *Access Point* yang juga sering di singkat menjadi WAP atau *Wireless Access Point*. Selain sebagai pusat dari jaringan *wireless*, sebuah AP biasanya juga mempunyai port UTP yang dapat digunakan untuk berhubungan langsung dengan jaringan *ethernet* yang telah ada. Dengan menghubungkan sebuah AP dengan jaringan kabel *wireless client* secara otomatis juga terhubung ke dalam jaringan kabel. Dengan cara ini, *wireless client* dapat tetap berhubungan dengan komputer lain yang masih menggunakan kabel, saling berbagi file, berbagi koneksi internet dan menggunakan *resource* jaringan yang lain. Sebuah AP biasanya sudah ditambahkan berbagai kemampuan tambahan yang tidak standar, seperti fungsi *router firewall* dan lain-lain. Gambar 2.1 menunjukkan pemanfaatan *access point*.



Gambar 2.1 Pemanfaatan Access Point

2.2 User Datagram Protocol [2]

UDP adalah protokol yang sangat sederhana dengan *overhead* yang minimum, jika suatu proses perlu untuk mengirim pesan yang relatif kecil dan tidak terlalu mementingkan kehandalan, tepat jika menggunakan UDP. Pengiriman pesan kecil menggunakan UDP membutuhkan interaksi antara pengirim dan penerima lebih sedikit dibandingkan bila menggunakan TCP. Tabel 2.1 menunjukkan beberapa nomor *port* yang sering digunakan oleh UDP. Beberapa *port* dapat juga digunakan baik oleh TCP maupun UDP.

Tabel 2.1 *Port yang Digunakan UDP*

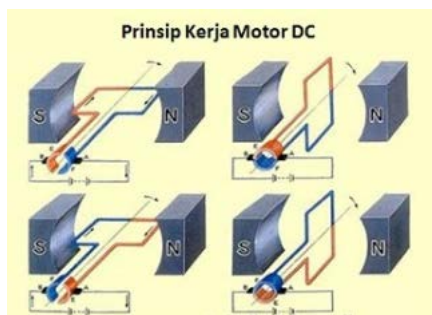
<i>Port</i>	Protokol	Deskripsi
7	Echo	Mengirim balik datagram ke pengirim
9	Discard	Membatalkan semua datagram yang datang
11	Users	Pengguna yang aktif
13	Daytime	Mengembalikan waktu an tanggal
17	Quote	Mengembalikan kutipan hari
19	Chargen	Mengembalikan serangkaian karakter
53	Nameserver	<i>Domain Name Server</i>
67	BOOTPs	<i>Port Server</i> untuk mendownload informasi <i>boot strapping</i>
68	BOOTPc	<i>Port Client</i> untuk mendownload informasi <i>boot strapping</i>
69	TFTP	<i>Trivial File Transfer Protocol</i>
111	RPC	<i>Remote Procedure Call</i>
123	NTP	<i>Network Time Protocol</i>
161	SNMP	<i>Simple Network Management</i>
162	SNMP	<i>Simple Network Management Protocol (trap)</i>

UDP memberikan layanan *connectionless services*, yang artinya setiap *user datagram* yang dikirim adalah *datagram* yang *independent*, sehingga tidak ada hubungan antara *datagram* yang berbeda meskipun mereka berasal dari proses yang sama dan program tujuan yang sama. *User datagram* juga tidak dinomori, serta tidak ada koneksi yang dibangun dan tidak ada koneksi yang diputuskan, seperti pada TCP, ini artinya setiap *user datagram* dapat menuju tujuan yang berbeda-beda.

2.3 Motor DC [3]

Motor listrik merupakan perangkat elektromagnetis yang mengubah energi listrik menjadi energi mekanik. Motor DC memerlukan suplai tegangan yang searah pada kumparan medan untuk diubah menjadi energi mekanik. Terdapat dua bagian utama dari motor DC yaitu stator dan rotor. Stator adalah bagian motor yang tidak berputar sedangkan rotor adalah bagian yang berputar.

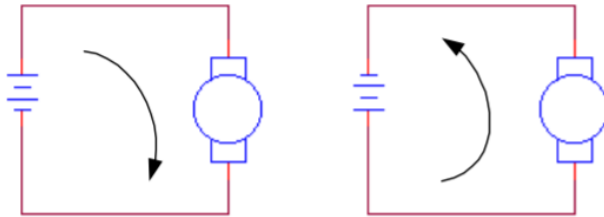
Pada prinsipnya motor listrik DC menggunakan fenomena electromagnet untuk bergerak, ketika arus listrik diberikan ke kumparan, permukaan kumparan yang bersifat utara akan bergerak menghadap ke magnet yang berkutub selatan dan kumparan yang bersifat selatan akan bergerak menghadap ke utara magnet. Saat ini, karena kutub utara kumparan bertemu dengan kutub selatan magnet maupun kutub selatan kumparan bertemu dengan kutub utara magnet maka akan terjadi saling tarik menarik yang menyebabkan pergerakan kumparan berhenti. Untuk menggerakkannya kembali, tepat pada saat kutub kumparan berhadapan dengan kutub magnet arah arus pada kumparan dibalik. Dengan demikian, kutub utara kumparan akan berubah menjadi kutub selatan dan kutub selatannya akan berubah menjadi kutub utara. Pada saat perubahan kutub tersebut terjadi, kutub selatan kumparan akan berhadapan dengan kutub selatan magnet dan kutub utara kumparan akan berhadapan dengan kutub utara magnet. Karena kutubnya sama, maka akan terjadi tolak menolak sehingga kumparan bergerak memutar hingga utara kumparan berhadapan dengan selatan magnet dan selatan kumparan berhadapan dengan utara magnet. Siklus ini akan berulang-ulang hingga arus listrik pada kumparan diputuskan. Prinsip kerja motor DC ditunjukkan pada Gambar 2.2.



Gambar 2.2 Prinsip Kerja Motor DC

Dalam aplikasinya seringkali sebuah motor digunakan untuk arah yang searah dengan jarum jam maupun sebaliknya. Untuk mengubah putaran dari sebuah motor dapat dilakukan dengan mengubah arah arus

yang mengalir melalui motor tersebut. Secara sederhana seperti yang ada pada Gambar 2.3, hal ini dapat dilakukan hanya dengan mengubah polaritas tegangan motor.



Gambar 2.3 Arah Putar Motor DC Magnet Permanen

Dikarenakan arah putaran motor tergantung dari arah aliran arus, dan untuk mengubah arah aliran arus perlu menukar posisi polaritas tegangan motor maka diperlukan rangkaian *H-bridge* agar perubahan polaritas tidak dilakukan secara manual. Cara kerja rangkaian *H-bridge* akan dijelaskan pada sub bab selanjutnya.

Kepesatan motor dc dapat dikontrol dengan berbagai cara, baik di sisi stator dan rotor. Persamaan 2.1 menunjukkan persamaan dalam kepesatan motor.

$$n = \frac{V_t - I_a R_a}{K\phi} \dots\dots\dots(2.1)$$

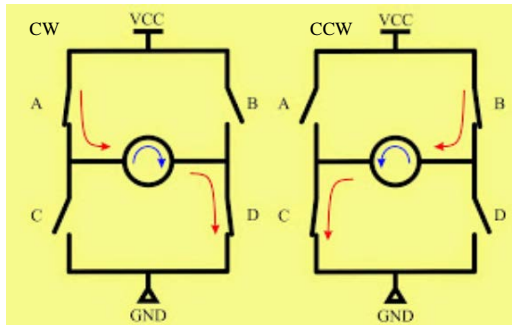
Dari Persamaan 2.1 dapat dilihat bahwa kepesatan motor n bergantung pada empat variabel yaitu medan Φ , tahanan rangkaian jangkar R_a , tegangan terminal V_t , dan arus jangkar I_a . Pengendalian kepesatan motor dapat dilakukan dengan mengubah nilai tiga dari variabel-variabel yang ada pada Persamaan 2.1. Arus jangkar I_a ditentukan oleh besarnya beban yang sedang dicatu oleh jangkar motor, oleh sebab itu tidak dapat digunakan untuk pengendalian kepesatan motor. Sehingga tiga metode dasar pengendalian adalah pengendalian fluks medan, pengendalian tahanan rangkaian jangkar R_a , atau pengendalian tegangan V_t . Dikarenakan motor yang digunakan pada Tugas Akhir ini adalah motor magnet permanen yang fluks medannya adalah konstan maka yang dilakukan untuk mengendalikan kepesatannya adalah dengan mengubah tegangan yang dikenakan pada rangkaian jangkar V_t . Pengaturan tegangan masukan ke motor DC dilakukan dengan menggunakan teknik modulasi lebar pulsa (*pulse-width modulation/PWM*). Mikrokontroler memiliki fitur PWM, sehingga pengaturan tegangan masukan DC motor yang secara otomatis dapat dilakukan dengan menggunakan mikrokontroler. Besar tegangan keluaran dari mikrokontroler 8-bit ditentukan dengan Persamaan 2.2.

$$V_{out} = \frac{\text{Nilai PWM}}{255} \times V_{in} \dots\dots\dots(2.2)$$

dimana V_{out} dan V_{in} masing-masing adalah tegangan keluaran mikrokontroler dan tegangan masukan mikrokontroler.

2.4 Driver L298N [3]

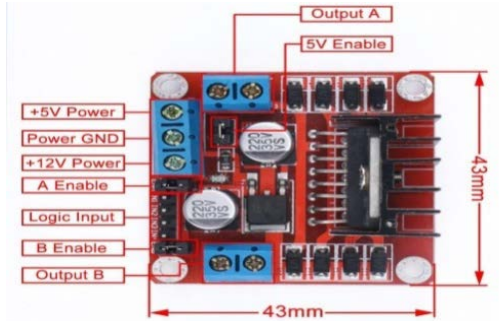
Driver motor L298N merupakan driver motor yang paling populer digunakan untuk mengontrol atau mengendalikan kecepatan dan arah pergerakan motor terutama untuk motor DC. Untuk IC utama yaitu L298N merupakan sebuah IC tipe *H-bridge* yang mampu mengendalikan beban-beban induktif seperti *relay*, *solenoid*, motor DC dan motor *stepper*. Fungsi utama dari rangkaian *H-bridge* adalah untuk mengubah arah arus listrik pada motor apakah mengalir dari kiri atau dari kanan. Perubahan arah arus tersebut digunakan untuk mengubah putaran motor, searah jarum jam atau *clock wise* (CW) atau berbalik arah jarum jam atau *counter clock wise* (CCW).



Gambar 2.4 Motor Berputar CW dan CCW

Gambar 2.4 sebelah kiri menunjukkan saklar A dan D aktif, sehingga arus listrik dari VCC mengalir dari arah kiri motor dan menyebabkan motor berputar CW. Sedangkan pada Gambar 2.4 sebelah kanan saklar yang aktif adalah B dan C, sehingga arus listrik mengalir dari arah kanan motor dan menyebabkan motor berputar sebaliknya (CCW).

Pada IC L298N terdiri dari *transistor-transistor logic (TTL)* dengan gerbang *nand* yang berfungsi untuk memudahkan dalam menentukan arah putaran suatu motor DC maupun motor *stepper*. Kelebihan akan modul driver motor L298N ini yaitu dalam hal kepresisian dalam mengontrol motor sehingga motor lebih mudah untuk dikontrol. Berikut gambat modul driver motor tersebut. Konstruksi pin driver motor DC L298N dapat dilihat pada Gambar 2.5.



Gambar 2.5 Driver L298N

2.4.1. Fungsi Pin Driver Motor DC L298N

Fungsi *pin driver* motor DC sebagai berikut:

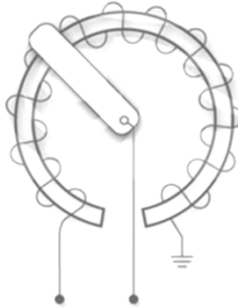
- *Output A* : digunakan untuk dihubungkan ke motor A
- *Output B* : digunakan untuk dihubungkan ke motor B
- *A Enable* : mengaktifkan kendali PWM motor A yang dihubungkan ke Arduino
- *B Enable* : mengaktifkan kendali PWM motor B yang dihubungkan ke Arduino
- *5v Enable* : mengaktifkan tegangan masukan yaitu 5 Vdc, jika tidak *dijumper* maka akan digunakan tegangan *direct* dari +12 V *power*
- *Logic Input* : digunakan untuk pengendali arah putaran motor A dan B

2.4.2. Spesifikasi Modul Driver Motor DC L298N

- Menggunakan IC L298N (*Double H bridge Drive Chip*)
- Tegangan minimal untuk masukan *power* antara 5V-35V
- Tegangan operasional : 5V
- Arus untuk masukan antara 0-36mA
- Arus maksimal untuk keluaran per *Output A* maupun B yaitu 2A
- Daya maksimal yaitu 25W
- Dimensi modul yaitu 43mm x 43mm x 26mm
- Berat : 26g

2.5 Potensiometer *Wirewound* [4]

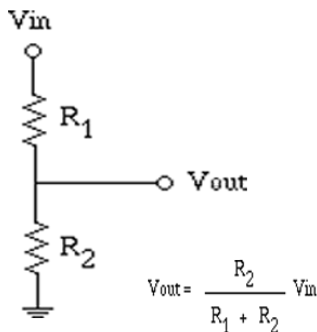
Potensiometer merupakan resistor tiga terminal dengan sambungan geser yang membentuk pembagi tegangan dapat diatur. Potensiometer yang sering dijumpai biasanya memiliki konstruksi *wirewound*. *Wirewound* potensiometer merupakan potensiometer yang berisi lilitan kawat yang dibuat melingkar sesuai dengan jejak kaki penggeser yang dibuat di dalamnya. Konstruksi dari *wirewound* potensiometer dapat dilihat pada Gambar 2.6.



Gambar 2.6 Konstruksi Sederhana *Wirewound Potentiometer*

Jumlah lilitan pada kawat konstruksi di atas merupakan faktor penentu besaran maksimal hambatan pada potensiometer *wirewound*. Perubahan dapat diatur sesuai dengan pergerakan dari poros pemutar yang juga berfungsi sebagai kaki tengah potensiometer.

Potensiometer menggunakan prinsip pembagi tegangan, dimana resistansi dari kaki 1 sampai kaki 2 dianggap R_1 dan resistansi dari kaki 2 sampai kaki 3 dianggap R_2 . Prinsip pembagian tegangan dapat dilihat pada Gambar 2.7 dan Gambar 2.8 menunjukkan potensiometer *wirewound* yang digunakan pada Tugas Akhir ini.



Gambar 2.7 *Voltage Divider*



Gambar 2.8 Potensiometer *Wirewound*

2.6 Sensor *Rotary Encoder* [5]

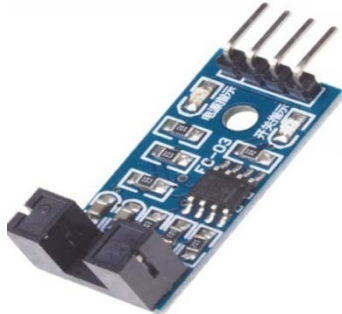
Rotary encoder adalah suatu komponen elektro mekanis yang memiliki fungsi untuk memonitoring posisi angular pada suatu poros yang berputar. Dari perputaran benda tersebut data yang termonitor akan diubah ke dalam bentuk data digital oleh *rotary encoder* berupa lebar pulsa kemudian akan dihubungkan ke mikrokontroler. Gambar 2.9 menunjukkan *rotary encoder* dengan *encoder disk* yang memiliki lubang disepanjang sisinya.



Gambar 2.9 *Rotary Encoder*

Konstruksi *rotary encoder* berupa piringan tipis yang biasanya di kopel dengan poros yang berputar. Piringan tipis tersebut terdapat lubang di sepanjang pinggir lingkarannya. Di bagian sisi-sisi piringan terdapat sebuah led dan *phototransistor* di bagian bersebrangan. Fungsi dari lubang-lubang yang berada di sepanjang pinggir lingkaran tersebut akan menghantarkan cahaya led ke *phototransistor*, sebaliknya jika cahaya led tidak menembus lubang piringan maka cahaya akan tertahan. Piringan tersebut akan berputar sesuai dengan kecepatan putaran motor sehingga *phototransistor* akan saturasi ketika cahaya led menembus

lubang-lubangnya. Pada Tugas Akhir ini sensor kecepatan yang digunakan adalah berupa modul IR *Optocoupler* FC-03. Modul ini menggunakan *comparator wide voltage* LM393. Modul IR *Optocoupler* dapat dilihat pada Gambar 2.10.



Gambar 2.10 Modul IR *Optocoupler* FC-03

Spesifikasi sensor kecepatan IR *Optocoupler* dapat dilihat pada Tabel 2.2.

Tabel 2.2 Spesifikasi Modul IR *Optocoupler* FC-03

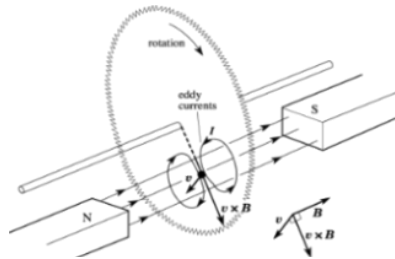
No	Spesifikasi	Nilai
1.	Lebar Celah	5 mm
2.	Arus	>15mA
3.	Tegangan Kerja	3,3-5 V
4.	Format <i>Output</i>	0 dan 1 (<i>Digital Output</i>)
5.	Dimensi	3,2 cm x 1,4 cm

2.7 Pengereman Magnetik [5]

Rem adalah suatu alat yang digunakan untuk melakukan aksi deselerasi yang akan menurunkan kecepatan dalam selang waktu yang ditentukan. Tipe rem yang umumnya digunakan adalah rem yang menggunakan gaya gesek untuk memberikan gaya lawan terhadap gaya gerak. Ada juga tipe rem lain yang tidak memanfaatkan gesekan dua permukaan untuk menghasilkan gaya lawan terhadap gaya penyebab gerak, yaitu rem yang menggunakan gaya magnet untuk menimbulkan gaya lawan. Rem ini disebut Rem Arus Eddy (Rem Magnetik).

Prinsip dasar rem magnetik ini menggunakan hukum Faraday dan hukum Lenz yang sudah terkenal di dunia elektromagnetik. Kedua hukum ini menimbulkan arus eddy yang melingkar dan menginduksi medan magnet yang melawan medan magnet penyebabnya. Hukum-hukum ini berlaku bila ada permukaan yang memotong medan magnet,

dengan artian gaya lawan hanya dihasilkan apabila permukaan tersebut memiliki kecepatan. Semakin tinggi kecepatan maka gaya lawan yang dihasilkan juga semakin besar. Namun semakin rendah kecepatan makagaya lawan akan semakin kecil. Gambar 2.11 menunjukkan ilustrasi gaya pengereman yang dihasilkan oleh arus melingkar Eddy.



Gambar 2.11 Gaya Pengereman Dihasilkan Oleh Arus Melingkar Eddy

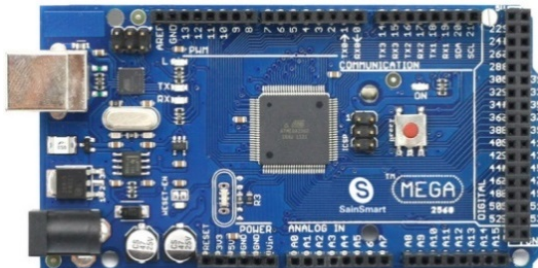
2.8 Mikrokontroler Arduino Mega 2560 [6]

Arduino Mega 2560 adalah *board* arduino yang merupakan perbaikan dari *board* arduino Mega sebelumnya. Arduino Mega awalnya memakai *chip* ATmega1280 dan kemudian diganti dengan *chip* ATmega2560, oleh karena itu namanya diganti menjadi arduino Mega 2560. Gambar 2.12 menunjukkan Arduino Mega 2560 yang digunakan pada Tugas Akhir ini. Dan berikut Tabel 2.3 merupakan spesifikasi arduino Mega 2560.

Tabel 2.3 Deskripsi Arduino Mega 2560

No.	Spesifikasi	Nilai
1.	Mikrokontroler	RISC ATmega 2560
2.	<i>Operating Voltage</i>	5 V
3.	<i>Input Voltage (recommended)</i>	7 – 12 V
4.	<i>Input Voltage (limit)</i>	6 – 20 V
5.	<i>Digital I/O Pins</i>	54 (15 diantaranya <i>input</i> PWM)
6.	<i>Analog Input Pins</i>	16
7.	<i>DC Current per I/O Pin</i>	40 mA
8.	<i>DC Current for 3,3V Pin</i>	50 mA
9.	<i>Flash Memory</i>	256 KB (8 KB sebagai <i>bootloader</i>)

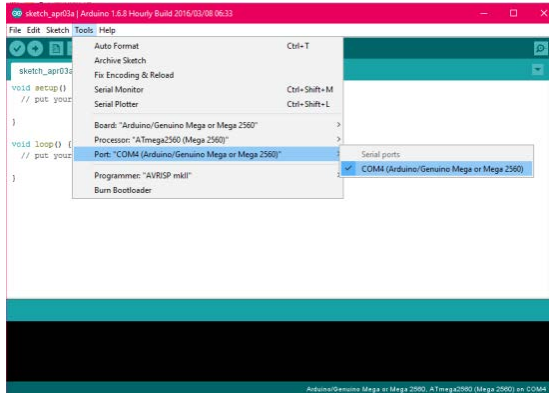
No.	Spesifikasi	Nilai
10.	SRAM	8 KB
11.	EEPROM	4 KB
12.	<i>Clock Speed</i>	16 MHz
13.	<i>USB Host Chip</i>	MAX3421E
14.	<i>Length</i>	101,52 mm
15.	<i>Width</i>	53,3 mm
16.	<i>Weight</i>	37 g



Gambar 2.12 Arduino Mega 2560

Untuk memprogram arduino juga harus dilakukan beberapa tahapan sebagai berikut :

1. **Setting Board Arduino.** Dalam pemrograman *software* arduino harus di *setting* terlebih dahulu *board* arduino agar penggunaan arduino tepat. Dalam purwarupa kali ini arduino menggunakan arduino Mega 2560. Untuk *setting board* arduino bisa masuk ke *tools – board* – setelah itu pilihlah *board* arduino yang sesuai.
2. **Setting Serial.** *Serial* ini merupakan kabel arduino yang dihubungkan kepada komputer atau laptop. *Serial* ini mempunyai dua fungsi yang bisa digunakan. Pertama *serial port* digunakan untuk *mendownload* program dari arduino yang kedua *serial* digunakan sebagai komunikasi *serial* pada arduino dengan komputer. *Setting serial* bisa masuk *tools – serial* - lali pilih COM yang sesuai dengan arduino yang terpasang. Untuk lebih jelasnya dapat dilihat pada Gambar 2.13.



Gambar 2.13 *Setting Serial Port*

3. Apabila program tidak dapat di *download* karena *serial port*, maka cek terlebih dahulu *serial* yang benar pada *device manager*. Lalu dalam *software* arduino untuk memilih *serial port*nya samakan dengan *serial port* untuk arduino dalam *device manager* tersebut. Untuk masuk ke *device manager* dapat masuk start *windows* – lalu ketika *device manager* klik dua kali dan masuk ke *COM*.

2.9 Ethernet Shield [7]

Ethernet Shield merupakan suatu perangkat yang dapat menambah kemampuan Arduino untuk terhubung ke jaringan komputer. *Ethernet Shield* berbasiskan cip *Ethernet Wiznet W5100*. Cip *Ethernet Wiznet W5100* ini menyediakan jaringan internet (IP) baik TCP dan UDP. Dan didukung oleh 4 soket koneksi yang simultan. Penggunaan perangkat ini mengacu pada *library Ethernet Shield* untuk penulisan programnya.

Pada *Ethernet Shield* terdapat sebuah slot *micro-SD (Secure Digital)*, yang dapat digunakan untuk menyimpan *file* yang diakses melalui jaringan. *Board* Arduino dapat berkomunikasi dengan *chip* Winzet W5100 dan *SD card* menggunakan bus ICSP. Pin ICSP ini terdiri dari MOSI, MISO, SCK, VCC, GND, dan RESET. Pin-pin yang sudah disebutkan sebelumnya tidak dapat digunakan untuk *input/output* umum ketika kita menggunakan *Ethernet Shield*. Gambar 2.14 menunjukkan *Ethernet Shield* yang digunakan pada Tugas Akhir ini.

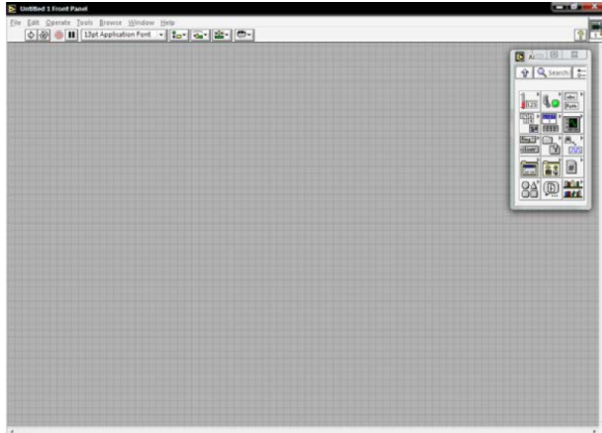


Gambar 2.14 *Ethernet Shield*

2.10 LabView [8]

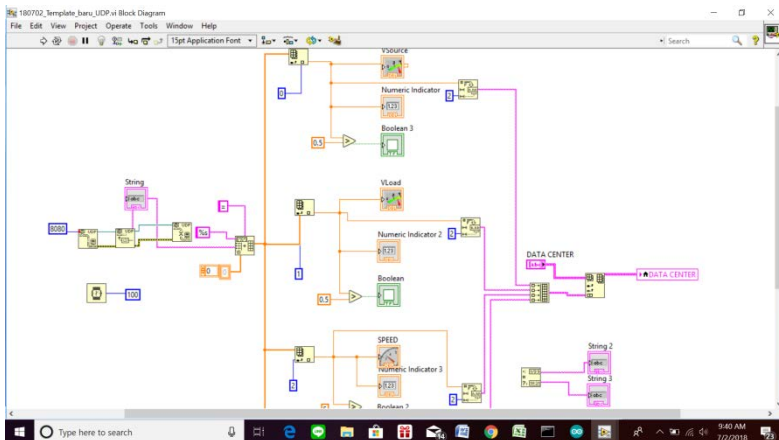
LabView adalah sebuah *software* pemrograman yang diproduksi oleh *National Instruments* dengan konsep yang berbeda. Seperti bahasa pemrograman lainnya yaitu C++, matlab atau *Visual basic*, LabView juga mempunyai fungsi dan peranan yang sama, perbedaannya bahwa LabView menggunakan bahasa pemrograman berbasis grafis atau blok diagram sementara bahasa pemrograman lainnya menggunakan basis *text*. Program LabView dikenal dengan sebutan VI atau *Virtual Instruments* karena penampilannya dan operasinya dapat meniru sebuah *instrument*. Pada LabView, *user* pertama-tama membuat *user interface* atau *front panel* dengan menggunakan *control* dan indikator, yang dimaksud dengan *control* adalah *knobs*, *push buttons*, *dials* dan peralatan input lainnya sedangkan yang dimaksud dengan indikator adalah *graphs*, LEDs dan peralatan display lainnya.

Setelah menyusun *user interface*, lalu *user* menyusun blok diagram yang berisi kode-kode VIs untuk mengontrol *front panel*. *Software* LabView terdiri dari tiga komponen utama, yaitu *front panel*, blok diagram dari Vi, *Control* dan *Functions Pallette*. *Front panel* adalah bagian *window* yang berlatar belakang abu-abu serta mengandung *control* dan indikator. *Front panel* digunakan untuk membangun sebuah VI, menjalankan program dan *debug* program. Tampilan dari *front panel* dapat dilihat pada Gambar 2.15.



Gambar 2.15 Front Panel LabView

Blok diagram adalah bagian *window* yang berlatar belakang putih berisi *source code* yang dibuat dan berfungsi sebagai instruksi untuk *front panel*. Tampilan dari blok diagram dapat dilihat pada Gambar 2.16.



Gambar 2.16 Blok Diagram LabView

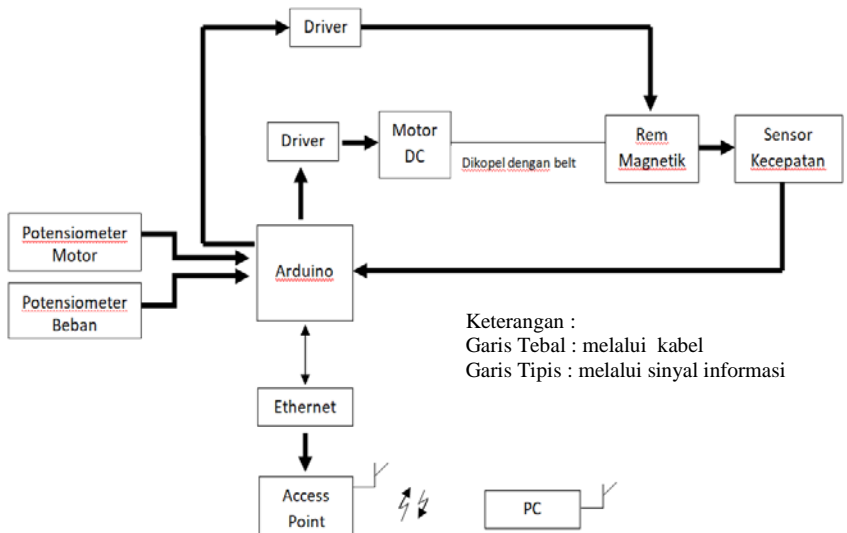
BAB III PERANCANGAN ALAT

Pada bab ini akan dibahas mengenai perancangan serta pembuatan “Telemetering Pengukuran Kecepatan pada Motor DC Berbeban”, baik perancangan perangkat elektronik (*hardware*), perancangan dan pembuatan perangkat lunak (*software*) yang meliputi :

1. Perancangan Mekanik
2. *Wiring Hardware* terdiri dari :
 - a. Koneksi Potensiometer *Wirewound*
 - b. Koneksi Sensor Kecepatan
 - c. Koneksi Modul *Driver* Motor L298N
 - d. Perancangan *Shield* Arduino Mega 2560
 - e. Koneksi *Ethernet Shield* dengan Arduino Mega 2560
 - f. Koneksi *Ethernet Shield* dengan *Access Point*
3. Perancangan *Software* yang berupa *flowchart* terdiri dari :
 - a. Pemrograman *Arduino IDE*
 - b. Pemrograman *LabView*

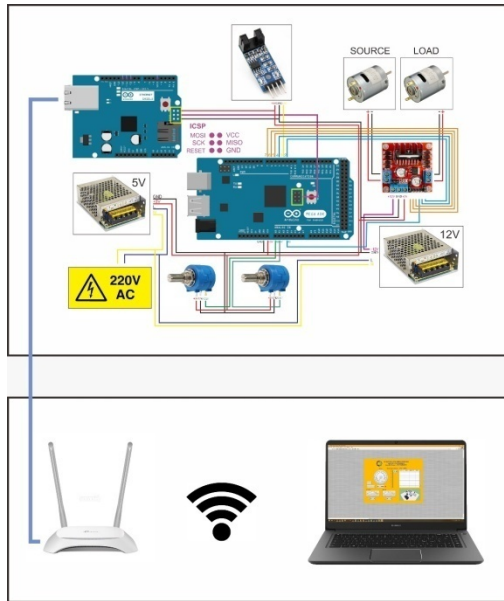
3.1 Diagram Fungsional Sistem

Perancangan sistem secara keseluruhan dalam pembuatan alat yang digunakan dapat diperlihatkan pada Gambar 3.1.



Gambar 3.1. Diagram Fungsional Perancangan

Dari Gambar 3.1 dapat dipahami bahwa jika motor DC dan *power supply* diberi daya maka Arduino Mega akan menyala dan motor DC berputar, sensor kecepatan akan bekerja membaca kecepatan dari motor DC dan potensiometer memasukkan berapa tegangan yang diperlukan agar terjadi pengereman pada motor DC. Pembacaan sensor kecepatan dan tegangan ke beban rem diolah sedemikian rupa hingga didapatkan data kecepatan dan pengereman yang ditampilkan pada *personal computer* (PC) melalui perangkat lunak LabView dengan komunikasi *ethernet compatible* Arduino.



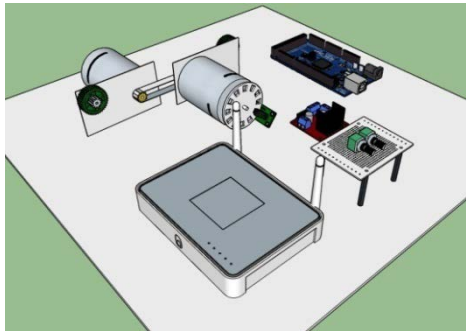
Gambar 3.2 Wiring Keseluruhan

Pada Gambar 3.2 menunjukkan *wiring* keseluruhan pada alat yang digunakan pada Tugas Akhir ini mulai dari sistem pengukuran hingga sistem pengiriman dan penerimaan data. *Pin* yang menghubungkan Arduino dan *Ethernet Shield* adalah *pin* ICSP, *pin* yang menghubungkan potensiometer dan arduino adalah *pin* A0 dan A5. *Pin* yang menghubungkan driver motor L298N dan arduino adalah *pin* digital 2, 4, 5, 6, 7, dan A3. *Pin* yang digunakan Arduino untuk koneksi dengan *pin* DO sensor kecepatan IR *Optocoupler* adalah *pin* 3. Sedangkan Arduino di *supply* dengan *power supply* bertegangan 5V.

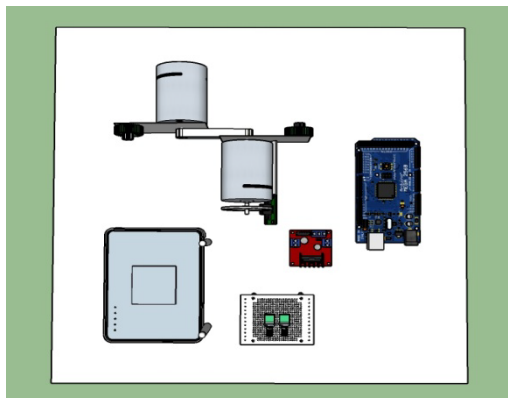
Pada sistem pengiriman dan penerimaan data, *Ethernet Shield* dihubungkan dengan *access point* dengan kabel RJ45 dan *access point* terkoneksi dengan PC secara nirkabel.

3.2 Perancangan Mekanik

Gambar 3.3 menunjukkan desain modul kontrol kecepatan motor DC secara keseluruhan untuk tampak samping dan Gambar 3.4 menunjukkan desain modul kontrol kecepatan motor DC untuk tampak atas.



Gambar 3.3 Desain Modul Tampak Samping



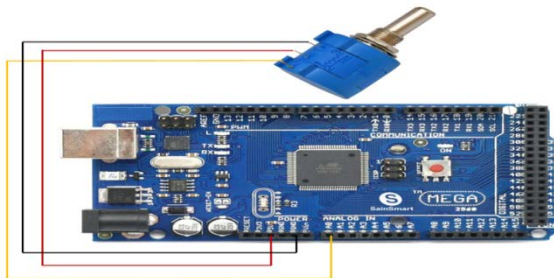
Gambar 3.4 Desain Modul Tampak Atas

3.3 Perancangan Perangkat Keras (*Hardware*)

Pada perancangan *hardware* pada Tugas Akhir ini yang dibahas terdiri dari koneksi keseluruhan *prototype* simulator pengendalian kecepatan motor DC berbeban dan koneksi elektrik berupa rangkaian sensor beserta komunikasinya.

3.3.1. Koneksi Potensiometer *Wirewound*

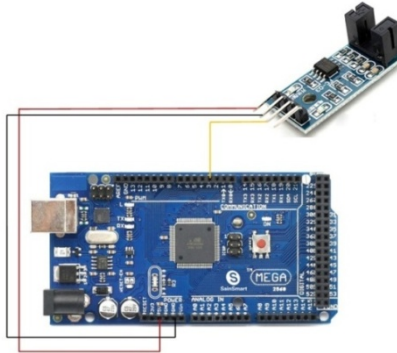
Tugas Akhir ini menggunakan dua potensiometer *wirewound* yang mana potensiometer motor digunakan sebagai input tegangan untuk motor DC sedangkan potensiometer beban digunakan sebagai input tegangan beban motor DC. Penggunaan potensiometer *wirewound* dikarenakan potensiometer jenis ini memiliki resolusi yang tinggi atau 10 kali putaran sehingga perubahan nilai tegangan yang dikeluarkan sangat *smooth* atau berubah sedikit demi sedikit. Pada Tugas Akhir ini potensiometer motor bernilai 10 k Ω sedangkan potensiometer beban bernilai 50 k Ω . Perbedaan kedua resistansi yang digunakan potensiometer tidak berpengaruh pada tegangan yang di keluarkan potensiometer dikarenakan menggunakan prinsip pembagia tegangan. Pada prinsip pembagian tegangan Gambar 2.5 ketika potensiometer beban diputar hingga mencapai persentase 50% dimana potensiometer akan mengeluarkan tegangan 2,5V, nilai R1 adalah 25 k Ω dan R2 adalah 25 k Ω . Sedangkan saat persentase beban 75% dimana potensiometer akan mengeluarkan tegangan 3,75V, nilai R1 adalah 12,5 k Ω dan R2 adalah 37,5 k Ω . Masing-masing potensiometer memiliki 3 kaki dimana salah satu kakinya adalah *output* yang akan terhubung pada *pin* analog Arduino. Potensiometer membutuhkan *supply* yang mana diambil dari 5 Volt DC dari Arduino. Keluaran dari potensiometer akan dihubungkan ke pin 2 dan pin 7, dimana pin 2 dan pin 7 adalah pin PWM yang juga terhubung ke *driver* L298N dengan pin EnA dan EnB. Gambar 3.5 menunjukkan *wiring* potensiometer *wirewound* dengan Arduino Mega 2560.



Gambar 3.5 Skema Rangkaian Potensiometer *Wirewound*

3.3.2. Koneksi Sensor Kecepatan

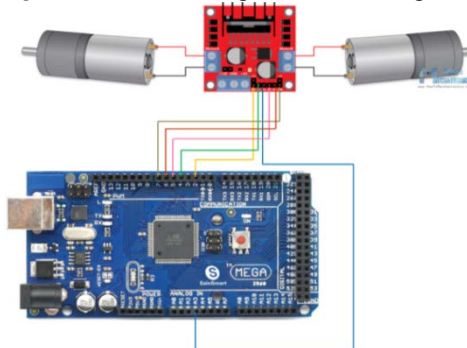
Pada Tugas Akhir ini, untuk mengetahui kecepatan dari motor DC digunakan modul IR *Optocoupler* FC-03. Modul ini dapat langsung dihubungkan ke Arduino. *Pin Vcc* pada modul sensor kecepatan IR *Optocoupler* FC-03 ini terhubung pada *pin 5 Volt* Arduino Mega, *Pin GND* dihubungkan pada *pin Ground* Arduino Mega dan *pin DO* dihubungkan pada *pin 3* Arduino. Gambar 3.6 menunjukkan *wiring* modul sensor kecepatan IR *Optocoupler* FC-03 dengan Arduino Mega.



Gambar 3.6 Skema Rangkaian Modul IR *Optocoupler* FC-03

3.3.3. Koneksi Driver Motor L298N dengan Arduino Mega 2560

Modul L298N adalah modul yang digunakan sebagai *driver* motor DC pada Tugas Akhir ini. Modul ini digunakan sebagai pengendali kecepatan dan arah putar motor DC. Pada driver L298N ini juga terdapat dua *channel* sehingga dalam satu modul dapat mengendalikan kecepatan dan arah putar dua motor DC. Modul L298N ini dapat langsung dihubungkan ke Arduino. Pada Gambar 3.7 menunjukkan *wiring* modul L298N dengan Arduino Mega 2560.

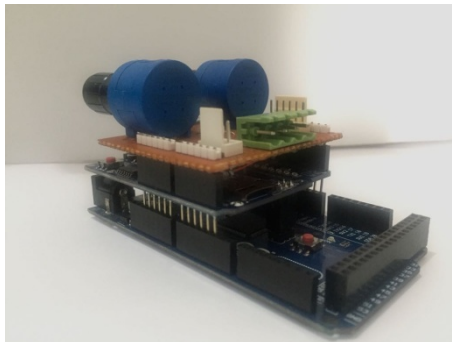


Gambar 3.7 Skema Rangkaian Modul L298N

Pin enA dan enB adalah untuk pengaturan PWM yang masuk ke *pin* 2 dan 5 pada Arduino Mega 2560. Sedangkan *pin* in1 dan in2 adalah *pin* yang berfungsi sebagai pengatur arah putar motor yang dihubungkan ke *pin* 4 dan *pin* A3 pada Arduino Mega 2560. *Pin* in3 dan in4 adalah *pin* yang berfungsi sebagai pengatur arah putar beban yang dihubungkan ke *pin* 5 dan *pin* 6 pada Arduino Mega 2560.

3.3.4. Perancangan *Shield* Arduino Mega 2560

Pada perancangan *shield* untuk Arduino Mega 2560, *shield* diletakkan di atas *Ethernet Shield* Arduino Mega 2560. *Pin* yang digunakan untuk menghubungkan *shield* dengan *Ethernet Shield* adalah mengacu pada *pin-pin* yang digunakan pada sensor kecepatan IR *Opticoupler*, potensiometer, dan *pin power* seperti VCC dan GND pada Arduino Mega 2560. Pemasangan *shield* yang berupa PCB di atas *Ethernet Shield* bertujuan untuk meminimalisir penggunaan kabel *jumper*. Gambar 3.8 menunjukkan pemasangan *shield* di atas *Ethernet Shield*.



Gambar 3.8 Skema Rangkaian *Shield*

3.3.5. Koneksi *Ethernet Shield* dengan Arduino Mega 2560

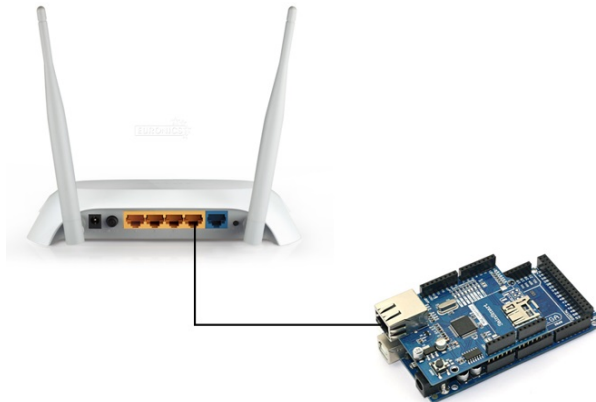
Pada perangkat komunikasi digunakan *Ethernet Shield* yang telah compatible dengan Arduino Mega 2560. Sehingga pemasangan *Ethernet Shield* pada Arduino Mega 2560 hanya dengan digabungkan pada bagian atas Arduino Mega 2560 saja. Gabungan antara Arduino Mega 2560 dan *Ethernet Shield* sering dinamakan Arduino *Web Server*. Arduino menggunakan *pin* ICSP untuk berkomunikasi dengan *Ethernet Shield* W5100. *Pin* ICSP ini terdiri dari MOSI, MISO, SCK, VCC, GND, dan RESET. *Pin* ini tidak dapat digunakan untuk I/O seperti *pin* lainnya. Gabungan *Ethernet shield* yang dipasang di atas Arduino Mega 2560 dan terhubung pada *pin* ICSP dapat dilihat pada Gambar 3.9.



Gambar 3.9. Koneksi *Ethernet Shield* Pada Arduino

3.3.6. Koneksi *Ethernet Shield* dengan *Access Point*

Pada koneksi *Ethernet shield* dengan *access point* dapat langsung dihubungkan dengan menggunakan kabel RJ45. *Access point* yang digunakan pada Tugas Akhir ini adalah TP Link TL-WR480N. Pada *Ethernet shield* telah terdapat slot RJ45 begitu pula dengan *access point*. Pada *access point*, kabel RJ45 dipasang pada slot LAN. Gambar 3.10 menunjukkan koneksi *Ethernet Shield* dengan *access point*.



Gambar 3.10. Koneksi *Ethernet Shield* Pada *Access Point*

3.4 Perancangan Perangkat Lunak (*Software*)

Pada perancangan perangkat *software* pada Tugas Akhir ini yang dibahas terdiri dari *flowchart* pemrograman pada *software* Arduino IDE untuk mengirimkan data kecepatan motor DC ke *software* LabView dan perancangan *software* LabView.

Agar motor dapat dimonitor dan hasil putaran dapat dilihat di PC maka perlu dirancang di sebuah program yang mampu mengelola data kinerja peralatan. *Software* merupakan program berisi perintah-perintah yang dieksekusi oleh Arduino MEGA 2560 sehingga sistem dapat bekerja sesuai dengan alur dan tujuan yang dirancang dan dapat ditampilkan pada komputer dari jarak jauh dengan komunikasi *Ethernet*.

3.4.1. Pemrograman *Software* Arduino untuk Sensor Kecepatan

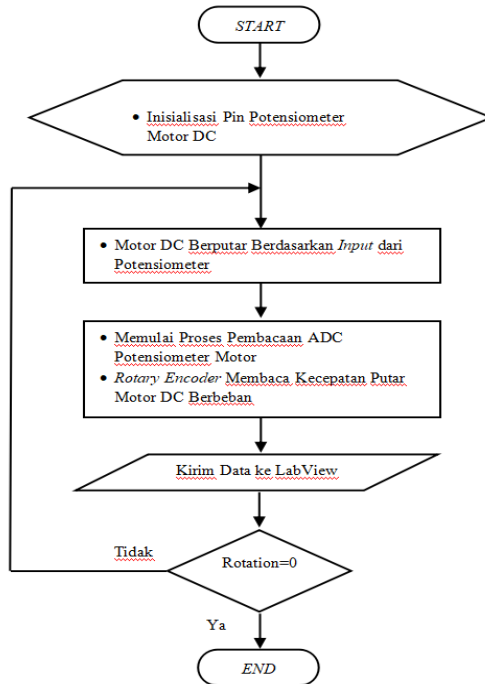
Dalam perancangan program pada *software* arduino dengan fungsi terkait yang dibutuhkan, diperlukan beberapa tahapan yang harus dilakukan terlebih dahulu. Tahapan tersebut adalah membuat algoritma dari alat yang akan dijalankan. Berikut ini algoritma program utama dari pengendalian kecepatan:

1. Modul dapat dioperasikan setelah sistem terpasang dengan benar seperti motor DC beserta beban, *driver* motor DC, catu daya dan potensiometer.
2. Modul dapat bekerja jika rangkaian kontrol sudah terpasang dan sudah dijalankan.
3. Potensiometer motor digunakan untuk mengatur *range* yang akan di suplai ke driver motor DC yang berupa tegangan dan akan diubah menjadi digital melalui ADC Arduino.
4. Potensiometer beban akan mengatur tegangan yang masuk ke beban yang berfungsi untuk memperlambat putaran motor.
5. Arduino akan menyimpan data kecepatan dan pengereman kemudian akan di tampilkan ke PC dengan *software* LabView melalui *Ethernet*.

Pemrograman *software* arduino dirancang dengan menggunakan *software* yang bernama Arduino IDE. Bahasa pemrograman arduino lebih mudah dan sederhana karena didalam arduino IDE sudah terdapat beberapa library yang dapat digunakan untuk merancang pemrograman yang diinginkan. Pada Gambar 3.11 berikut menunjukkan *flowchart* yang digunakan pada Arduino Mega 2560 untuk pembacaan sensor kecepatan pada Tugas Akhir ini.

Pada awal program dijalankan, perlu dilakukan inisialisasi *pin* yang dihubungkan pada *port* arduino kemudian setelah semua *pin* yang digunakan diinisialisasi, dilanjutkan dengan menginisialisasi data

kecepatan motor DC dari sensor *Rotary Encoder*. Kemudian ketika motor dijalankan berdasarkan *input* dari potensiometer dan *Rotary Encoder* membaca kecepatan motor DC, arduino akan memulai proses pembacaan kecepatan motor DC dan kemudian data kecepatan akan dikirimkan ke LabView. Gambar 3.11 menunjukkan *flowchart* yang difungsikan pada Arduino untuk *monitoring* kecepatan motor DC.

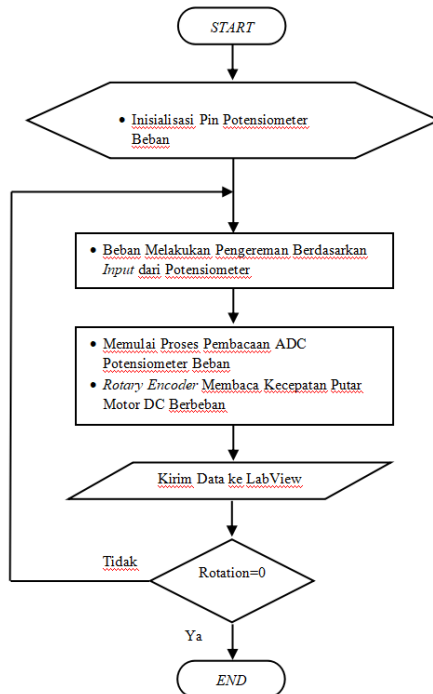


Gambar 3.11. *Flowchart* Sensor Kecepatan pada Arduino

3.4.2. Pemrograman *Software* Arduino untuk Pengereman

Pada pemrograman untuk pengereman yang pertama dilakukan adalah inisialisasi *pin* yang dihubungkan pada *port* arduino, kemudian setelah semua *pin* yang digunakan untuk pengereman terinisialisasi, dilanjutkan dengan menginisialisasi data *range* beban. Kemudian beban mulai melakukan pengereman berdasarkan *input* dari potensiometer, arduino akan membaca tegangan yang dimasukkan pada beban dan kemudian data beban rem akan dikirimkan ke LabView.

Gambar 3.12 menunjukkan *flowchart* yang difungsikan pada Arduino untuk *monitoring* pengereman motor DC.



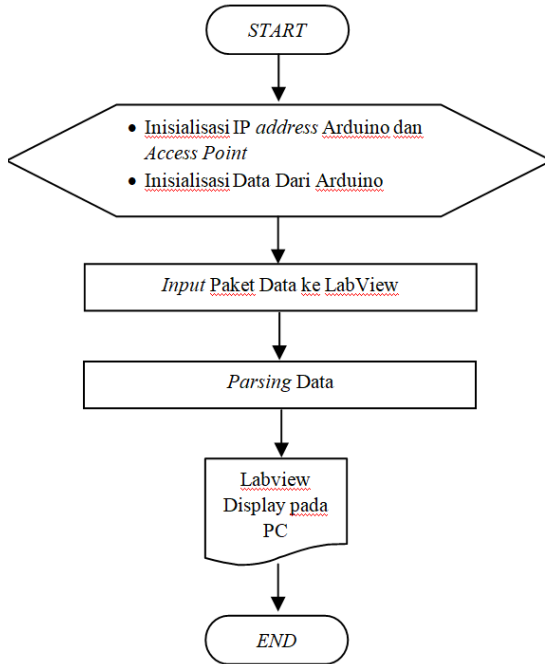
Gambar 3.12. *Flowchart* Pengereman pada Arduino

3.4.3. Perancangan *Software* LabView

Pada perancangan *software* LabView yang pertama dirancang yaitu membuat *block diagram* setelah itu merancang *front panel*. *Front Panel Window* merupakan *user interface window* pada saat VI dijalankan, sedangkan *Block Diagram Window* merupakan bagian yang akan melakukan kalkulasi dan mengeksekusi algoritma. Gambar 3.13 menunjukkan *flowchart* yang difungsikan pada LabView untuk *monitoring* kecepatan dan pengereman pada motor DC.

Pada perancangan blok diagram *software* LabView protokol komunikasi yang digunakan adalah UDP. Saat program dijalankan data masuk ke UDP *open* kemudian data di *write* ke LabView pada UDP *write*. Kemudian data yang berupa satu paket di *parsing* dengan

separator berupa tanda sama dengan (=). Kemudian setelah di *parsing*, data akan terpisah dan tampil pada *front panel* LabView.



Gambar 3.13. Flowchart Blok Diagram LabView

Pada perancangan *front panel software* LabView yang mana akan tampil sebagai *interface*, akan menunjukkan tegangan yang diberikan pada motor dan tegangan yang diberikan ke beban akan ditampilkan dengan tampilan Meter beserta *digital display*nya. Pembacaan tegangan yang diberikan ke motor didapatkan dari Persamaan 3.1 dan pembacaan tegangan yang diberikan ke beban didapatkan dari Persamaan 3.2.

$$V_{motor} = \frac{(ADC_{potensiometer\ motor} \times 5,2 V)}{1023} \dots \dots \dots (3.1)$$

Pada Persamaan 3.1 menunjukkan bahwa tegangan yang diberikan ke motor adalah nilai ADC oleh potensiometer motor dikalikan dengan 5,2V yang merupakan tegangan maksimal yang di keluarkan potensiometer dan dibagi dengan 1023 yang merupakan nilai maksimal ADC 10 bit.

$$V_{beban} = \frac{(ADC_{potensiometer\ beban} \times 5,2 V)}{1023} \dots \dots \dots (3.2)$$

Pada Persamaan 3.2 menunjukkan bahwa tegangan yang diberikan ke beban adalah nilai ADC oleh potensiometer beban dikalikan dengan 5,2V yang merupakan tegangan maksimal yang di keluarkan potensiometer dan dibagi dengan 1023 yang merupakan nilai maksimal ADC 10 bit.

Untuk interface kecepatan menggunakan *gauge* beserta digital displaynya yang mana nilainya diambil dari pembacaan digital sensor kecepatan. Kemudian pada *front panel* juga terdapat *data center* yang berfungsi mencatat semua data dan ditampilkan dalam bentuk tabel. Pada bagian pojok kanan atas *front panel* terdapat fungsi waktu yang diambil langsung dari PC.

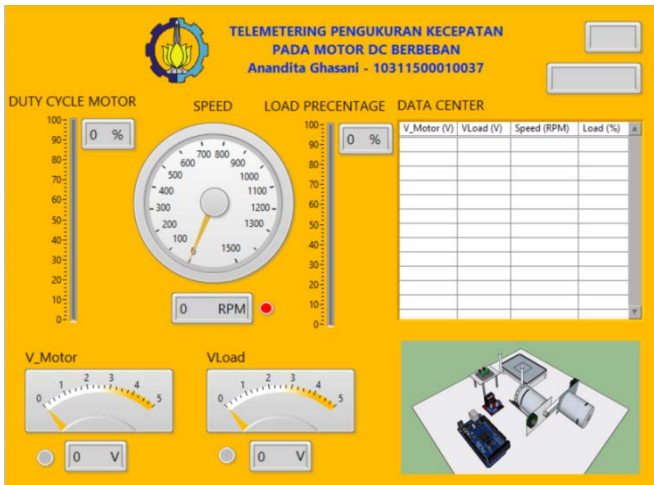
Persentase rem ditampilkan dalam bentuk *vertical progress bar* beserta *digital display* dimana nilainya didapat dari Persamaan 3.3 dimana persentase beban adalah ADC beban dibandingkan dengan tegangan maksimal 5V kemudian dikalikan 100%.

$$\%Brake = \frac{(ADC \text{ beban} \times 100\%)}{5V} \dots\dots\dots(3.3)$$

Duty cycle motor ditampilkan dalam bentuk *vertical progress bar* beserta *digital display* dimana nilainya didapat dari Persamaan 3.4 dimana persentase beban adalah nilai PWM motor dibandingkan dengan nilai maksimum PWM 8 bit yaitu 255 kemudian dikalikan 100%.

$$Duty \text{ Cycle Motor} = \frac{(PWM \text{ motor} \times 100\%)}{255} \dots\dots\dots(3.3)$$

Gambar 3.14 menunjukkan tampilan *front panel* Labview pada Tugas Akhir ini.



Gambar 3.14. *Front Panel* LabView

BAB IV PENGUJIAN DAN ANALISA ALAT

4.1 Pengujian *Input/Output* Arduino Mega 2560

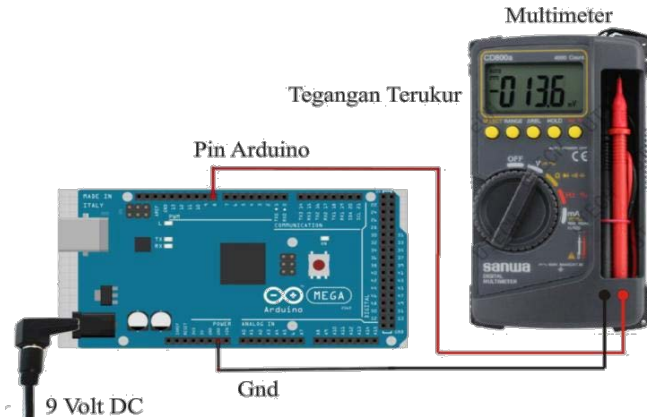
Pengujian *input/output* pada Arduino dilakukan pada rangkaian *board* Arduino Mega 2560. Pengujian dilakukan untuk mengetahui tegangan *output* pada setiap *pin* Arduino Mega 2560 jika diberi *input high* dan *input low* dengan menggunakan multimeter. Langkah yang harus dilakukan untuk mengetahui tegangan *output* saat *input high* adalah :

1. Menyiapkan Arduino Mega 2560 dan Multimeter
2. Memberi *supply* tegangan pada Arduino
3. Mengunggah program pada Arduino yakni memberikan perintah *high* atau logika 1 pada setiap *pin* Arduino. Gambar 4.1 menunjukkan potongan program pengujian I/O Arduino Mega 2560.

```
void setup() {  
  // initialize digital pin |  
  pinMode(A0, OUTPUT);  
  pinMode(A3, OUTPUT);  
  pinMode(A5, OUTPUT);  
  pinMode(2, OUTPUT);  
  pinMode(3, OUTPUT);  
  pinMode(4, OUTPUT);  
  pinMode(5, OUTPUT);  
  pinMode(6, OUTPUT);  
  pinMode(7, OUTPUT);  
  pinMode(10, OUTPUT);  
  pinMode(11, OUTPUT);  
  pinMode(12, OUTPUT);  
  pinMode(13, OUTPUT);  
}  
  
// the loop function runs over and over again forever  
void loop() {  
  digitalWrite(A0, HIGH);  
  digitalWrite(A0, HIGH);  
  digitalWrite(A3, HIGH);  
  digitalWrite(A5, HIGH);  
  digitalWrite(2, HIGH);  
  digitalWrite(3, HIGH);  
  digitalWrite(4, HIGH);  
  digitalWrite(5, HIGH);  
}
```

Gambar 4.1 Program Pengujian I/O Arduino *Active High*

4. Kemudian mengukur tegangan setiap *pin* pada Arduino dengan multimeter. Skema pengukuran ditunjukkan pada Gambar 4.2.



Gambar 4.2 Skema Pengujian I/O Arduino

Hasil pengukuran pada tiap *pin* dari *board* Arduino Mega 2560 dapat dilihat pada Tabel 4.1.

Tabel 4.1. Hasil Pengukuran Per *Pin* Saat *Active High*

<i>Pin Analog</i>	Tegangan (Volt)	<i>Pin Digital</i>	Tegangan (Volt)
<i>Pin A0</i>	5,01	<i>Pin 2</i>	5,01
<i>Pin A3</i>	5,01	<i>Pin 3</i>	5,01
<i>Pin A5</i>	5,01	<i>Pin 4</i>	5,01
		<i>Pin 5</i>	5,01
		<i>Pin 6</i>	5,01
		<i>Pin 7</i>	5,01

Dari hasil Tabel 4.1 dapat disimpulkan bahwa ketika *board* Arduino Mega 2560 diberi *input high* pada tiap pin maka tegangan yang dihasilkan bernilai rata-rata 5,01 Volt yang artinya tegangan yang dikeluarkan telah maksimal dan pin tersebut dapat berfungsi sebagai *supply* untuk beban yang diinginkan karena *output board* Arduino Mega 2560 bernilai *high* dengan *range* tegangan 2,4-5,5 Volt.

Selanjutnya adalah pengujian *board* Arduino untuk mengetahui tegangan *output* Arduino Mega 2560 saat diberi tegangan *input low*. Langkah-langkah yang harus dilakukan adalah sebagai berikut :

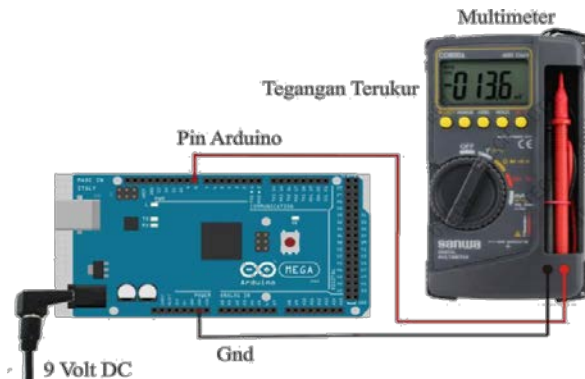
1. Menyiapkan Arduino Mega 2560 dan Multimeter
2. Memberi *supply* tegangan pada Arduino

3. Mengunggah program pada Arduino yakni memberikan perintah *low* atau logika 0 pada setiap *pin* Arduino. Gambar 4.3 menunjukkan potongan program pengujian I/O Arduino Mega 2560.

```
void setup() {  
  // initialize digital pin  
  pinMode(A0, OUTPUT);  
  pinMode(A3, OUTPUT);  
  pinMode(A5, OUTPUT);  
  pinMode(2, OUTPUT);  
  pinMode(3, OUTPUT);  
  pinMode(4, OUTPUT);  
  pinMode(5, OUTPUT);  
  pinMode(6, OUTPUT);  
  pinMode(7, OUTPUT);  
  pinMode(10, OUTPUT);  
  pinMode(11, OUTPUT);  
  pinMode(12, OUTPUT);  
  pinMode(13, OUTPUT);  
}  
  
// the loop function runs over and over again forever  
void loop() {  
  digitalWrite(A0, LOW);  
  digitalWrite(A0, LOW);  
  digitalWrite(A3, LOW);  
  digitalWrite(A5, LOW);  
  digitalWrite(2, LOW);  
}
```

Gambar 4.3 Program Pengujian I/O Arduino *Active Low*

4. Kemudian mengukur tegangan setiap *pin* pada Arduino dengan multimeter. Skema pengukuran ditunjukkan pada Gambar 4.4.



Gambar 4.4 Skema Pengujian I/O Arduino Mega 2560

Hasil pengukuran pada tiap *pin* dari *board* Arduino Mega 2560 dapat dilihat pada Tabel 4.2.

Tabel 4.2. Hasil Pengukuran Per *Pin* Saat *Active Low*

<i>Pin Analog</i>	Tegangan (mV)	<i>Pin Digital</i>	Tegangan (mV)
<i>Pin A0</i>	0,8	<i>Pin 2</i>	1,2
<i>Pin A3</i>	0,8	<i>Pin 3</i>	1,2
<i>Pin A1</i>	0,8	<i>Pin 4</i>	1,2
		<i>Pin 5</i>	1,1
		<i>Pin 6</i>	1,1
		<i>Pin 7</i>	1,2

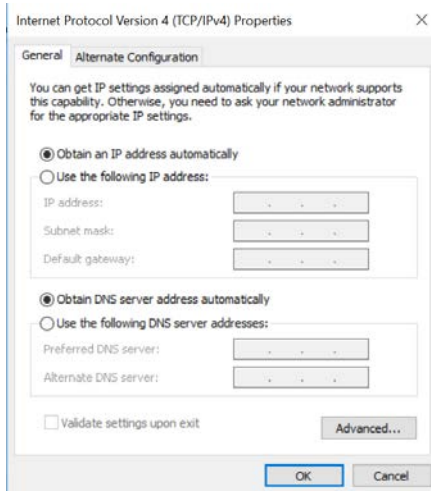
Dari hasil Tabel 4.2 dapat disimpulkan bahwa ketika *board* Arduino Mega 2560 diberi *input low* pada tiap *pin* maka tegangan yang dihasilkan bernilai rata-rata 1,76 mV yang artinya tegangan yang dikeluarkan sangat kecil dan *pin* tersebut dapat berfungsi sebagai *ground* karena *output board* Arduino Mega 2560 bernilai *low* dengan *range* tegangan 0–800 mV.

4.2 Pengujian Komunikasi *Ethernet*

Pengujian ini dilakukan pada modul *Ethernet Shield* yang telah dipasang pada bagian atas *board* Arduino Mega 2560. Pada pengujian komunikasi ini dibagi menjadi dua, yaitu pengujian komunikasi *Ethernet* pada *board* Arduino Mega 2560 dan pengujian komunikasi *Ethernet* dengan *software* LabView.

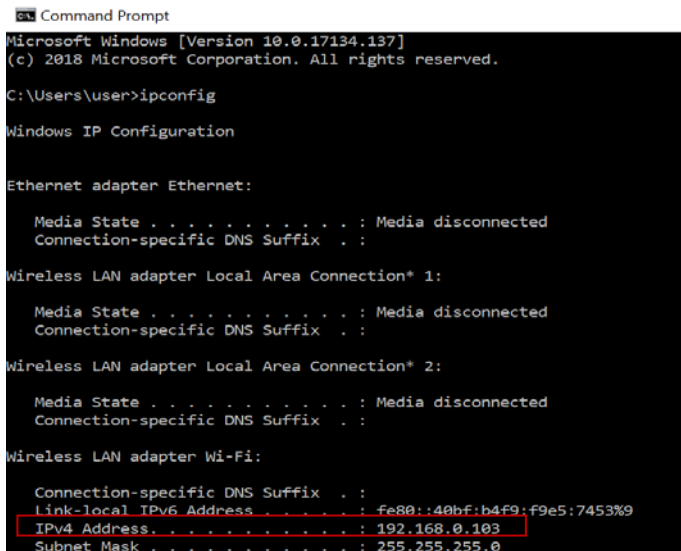
4.2.1 Pengujian Koneksi *Access Point* dan PC

Tahapan awal untuk memulai komunikasi antara *Ethernet* dan PC adalah dengan mengkoneksikan *access point* dengan PC. Kemudian mengatur IP *address access point* pada *setting Internet Protocol Version 4* (TCP/IPv4) menjadi otomatis. Pengaturan IP *address access point* menjadi otomatis dapat dilihat pada Gambar 4.5.



Gambar 4.5. *Setting IP Address Access Point Pada Laptop*

Kemudian setelah melakukan *setting IP address access point*, koneksinya dapat diuji dengan melakukan *ipconfig* pada *command prompt* pada laptop. Pengujian koneksi *access point* pada *command prompt* dapat dilihat pada Gambar 4.6.



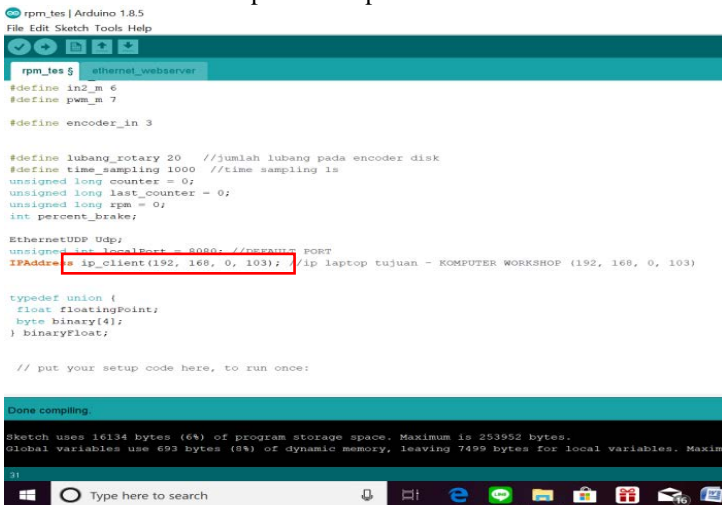
Gambar 4.6. *Command Prompt dengan IP Address Access Point*

Pada Gambar 4.6 dapat dilihat bahwa *Access Point* yang tersambung pada laptop telah siap digunakan. Ketika mengirim perintah *ipconfig command prompt* maka laptop akan memunculkan informasi nomor IP address *Access Point* yang menandakan PC dan *access point* telah terhubung.

4.2.2 Pengujian pada *Board Arduino Mega 2560*

Sebelum menggunakan komunikasi *Ethernet*, sebaiknya dilakukan pengujian komunikasi *Ethernet* pada *board Arduino Mega 2560* terlebih dahulu. Pengujian ini dilakukan dengan cara sebagai berikut :

1. Menyamakan *setting IP (Internet Protocol) address* yang telah terdaftar pada *ipconfig* di *command prompt* PC seperti Gambar 4.7 ke program pada Arduino IDE. Penyamakan IP address antara yang terdaftar pada *command prompt* dengan *software Arduino IDE* dapat dilihat pada Gambar 4.7.



```
rpm_tes | Arduino 1.8.5
File Edit Sketch Tools Help

rpm_tes $ ethernet_webserver

#define in2_m 6
#define pwm_m 7

#define encoder_in 3

#define lubang_rotary 20 //jumlah lubang pada encoder disk
#define time_sampling 1000 //time sampling is
unsigned long counter = 0;
unsigned long last_counter = 0;
unsigned long rpm = 0;
int percent_brake;

EthernetUDP Udp;
unsigned int localPort = 8080; //DEFAULT PORT
IPAddress ip_client(192, 168, 0, 103); //ip laptop tujuan - KOMPUTER WORKSHOP (192, 168, 0, 103)

typedef union {
float floatingPoint;
byte binary[4];
} binaryfloat;

// put your setup code here, to run once:

Done compiling
Sketch uses 16134 bytes (6%) of program storage space. Maximum is 253952 bytes.
Global Variables use 693 bytes (8%) of dynamic memory, leaving 7499 bytes for local variables. Maximum available is 24500 bytes.
```

Gambar 4.7. Program *Setting IP* Arduino

2. Kemudian mendaftarkan IP address untuk *board Arduino* ke program di *software Arduino IDE*. Gambar 4.8 menunjukkan *setting IP address board Arduino* pada *software Arduino IDE*. Terlihat bahwa pada *software Arduino IDE* IP address yang didaftarkan adalah 192.168.0.177.

```
rpm_test - ethernet_webserver.ino | Arduino 1.8.5
File Edit Sketch Tools Help

rpm_test ethernet_webserver
Analog inputs attached to pins A0 through A5 (optional)

created 18 Dec 2009
by David A. Mellis
modified 9 Apr 2012
by Tom Igoe

*/

#include <SPI.h>
#include <Ethernet.h>

// Menentukan MAC, IP dan port Ethernet
byte mac[] = { 0x0F, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E }; //Default
IPAddress ip(192, 168, 0, 177); //Sesuai setingan router
EthernetServer server(80); //default WEB Server port

void setup_ethernet()
{
  Ethernet.begin(mac, ip);
  server.begin();
}
```

Gambar 4.8 Setting IP address untuk board Arduino

3. Untuk mengetahui Arduino telah terhubung atau belum dengan PC maka perlu dilakukan tes Ping pada *command prompt*. Caranya adalah dengan mengetik ping 192.168.0.177 kemudian tekan tombol *enter*. Gambar 4.9 menunjukkan tes ping pada *command prompt* untuk IP address board Arduino Mega 2560.

```
Command Prompt

:\Users\user>ping 192.168.0.177

Pinging 192.168.0.177 with 32 bytes of data:
Reply from 192.168.0.177: bytes=32 time=9ms TTL=128
Reply from 192.168.0.177: bytes=32 time=3ms TTL=128
Reply from 192.168.0.177: bytes=32 time=7ms TTL=128
Reply from 192.168.0.177: bytes=32 time=6ms TTL=128

Ping statistics for 192.168.0.177:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    approximate round trip times in milli-seconds:
        Minimum = 3ms, Maximum = 9ms, Average = 6ms

:\Users\user>
```

Gambar 4.9 Tes Ping Pada IP Address Board Arduino

Saat *command prompt* menyatakan bahwa *connection loss* adalah 0%, hal ini menandakan bahwa *board* Arduino sudah terhubung dengan PC.

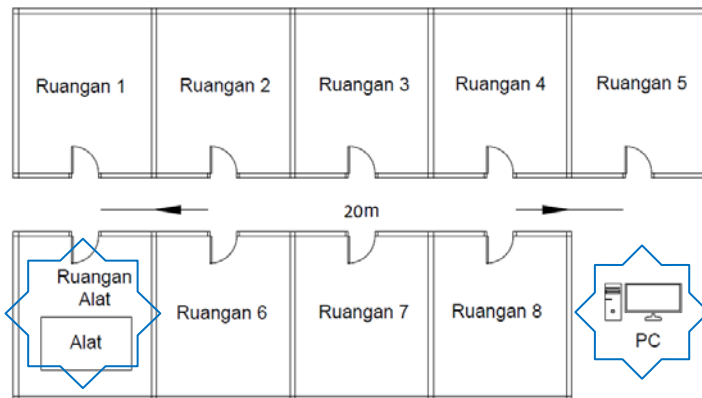
4.3 Pengujian Koneksi Terjadap Jarak

Fungsi telemetering adalah memungkinkan pengukuran jarak jauh dan pelaporan informasi kepada operator sistem. Untuk mengetahui stabilitas koneksi terhadap jarak antara PC dan *access point* maka perlu dilakukan tes ping terhadap IP *address* Arduino.

4.3.1. Pengujian Koneksi Dengan Penghalang

Untuk mengetahui jarak maksimal yang dapat dicapai dari koneksi *access point* dan PC dengan penghalang, maka dilakukan langkah-langkah pengujian sebagai berikut :

1. Memastikan PC dan *access point* telah terhubung. Hal ini dapat dilihat pada Gambar 4.9.
2. Meletakkan PC dan *access point* dengan jarak 1 meter, 5 meter, 10 meter, 17 meter, dan 20,4 meter secara berurutan dengan penghalang.
3. Melakukan tes ping IP *address* pada setiap jarak. Sketsa pengujian koneksi dapat dilihat pada ilustrasi Gambar 4.10.



Gambar 4.10 Ilustrasi Pengujian Koneksi Dengan Penghalang

Gambar pengujian koneksi pada *command prompt* dapat dilihat pada Gambar 4.11, Gambar 4.12, Gambar 4.13 dan Gambar 4.14.

```
C:\WINDOWS\system32\cmd.exe
```

```
C:\Users\user>ping 192.168.0.177

Pinging 192.168.0.177 with 32 bytes of data:
Reply from 192.168.0.177: bytes=32 time=2ms TTL=128
Reply from 192.168.0.177: bytes=32 time=2ms TTL=128
Reply from 192.168.0.177: bytes=32 time=2ms TTL=128
Reply from 192.168.0.177: bytes=32 time=2ms TTL=128

Ping statistics for 192.168.0.177:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 2ms, Maximum = 2ms, Average = 2ms
```

Gambar 4.11. Tes Ping 5 Meter Dengan Penghalang

Dapat dilihat pada Gambar 4.11 menunjukkan bahwa *Ethernet Shield* yang terhubung pada laptop masih belum mengalami gangguan pada sistem komunikasinya, dibuktikan dari hasil ping pada *command prompt* di laptop pada jarak antara laptop dan *access point* sejauh 5 meter dengan penghalang dinding ruangan.

```
C:\Users\user>ping 192.168.0.177

Pinging 192.168.0.177 with 32 bytes of data:
Reply from 192.168.0.177: bytes=32 time=5ms TTL=128
Reply from 192.168.0.177: bytes=32 time=7ms TTL=128
Reply from 192.168.0.177: bytes=32 time=8ms TTL=128
Reply from 192.168.0.177: bytes=32 time=54ms TTL=128

Ping statistics for 192.168.0.177:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 5ms, Maximum = 54ms, Average = 18ms
```

Gambar 4.12 Tes Ping 10 Meter Dengan Penghalang

Gambar 4.12 menunjukkan bahwa *Ethernet Shield* yang terhubung pada laptop masih belum mengalami gangguan pada sistem komunikasinya, dibuktikan dari hasil ping pada *command prompt* di laptop pada jarak antara laptop dan *access point* sejauh 10 meter dengan penghalang dinding ruangan.

```
C:\Users\user>ping 192.168.0.177

Pinging 192.168.0.177 with 32 bytes of data:
Request timed out.
Reply from 192.168.0.177: bytes=32 time=42ms TTL=128
Reply from 192.168.0.177: bytes=32 time=7ms TTL=128
Reply from 192.168.0.177: bytes=32 time=27ms TTL=128

Ping statistics for 192.168.0.177:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 7ms, Maximum = 42ms, Average = 25ms
```

Gambar 4.13 Tes Ping 17 Meter Dengan Penghalang

Gambar 4.13 menunjukkan bahwa *Ethernet Shield* yang terhubung pada laptop mengalami gangguan pada sistem komunikasinya, dibuktikan dari hasil ping pada *command prompt* di laptop pada jarak antara laptop dan *access point* sejauh 17 meter dengan penghalang dinding ruangan.

```
Command Prompt

C:\Users\aaכותib>ping 192.168.0.177

Pinging 192.168.0.177 with 32 bytes of data:
Reply from 192.168.0.177: bytes=32 time=4ms TTL=128
Request timed out.
Request timed out.
Reply from 192.168.0.177: bytes=32 time=2ms TTL=128

Ping statistics for 192.168.0.177:
    Packets: Sent = 4, Received = 4, Lost = 0 (50%loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 4ms, Average = 1ms
```

Gambar 4.14 Tes Ping 20 Meter Dengan Penghalang

Dapat dilihat pada Gambar 4.14 menunjukkan bahwa *Ethernet Shield* yang terhubung pada laptop mengalami gangguan pada sistem komunikasinya, dibuktikan dari hasil ping pada *command prompt* di laptop pada jarak antara laptop dan *access point* sejauh 20 meter dengan penghalang dinding ruangan.

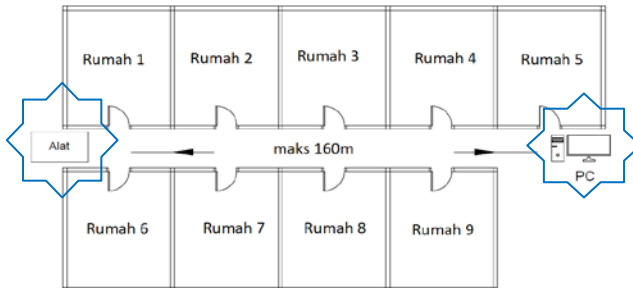
Tabel 4.3. Pengujian Koneksi Dengan Penghalang

Jarak (Meter)	Koneksi <i>Loss</i> (%)	<i>Time Average</i>
5	0	1ms
10	0	18ms
17	25	25ms
20	50	1ms

4.3.2. Pengujian Koneksi Tanpa Penghalang

Untuk mengetahui jarak maksimal yang dapat dicapai dari koneksi *access point* dan PC tanpa penghalang, maka dilakukan langkah-langkah pengujian sebagai berikut :

1. Memastikan PC dan *access point* telah terhubung. Hal ini dapat dilihat pada Gambar 4.9.
2. Meletakkan PC dan *access point* dengan jarak 5 meter, 15 meter, 30 meter, dan 160 meter secara berurutan tanpa penghalang.
3. Melakukan tes ping *IP address* pada setiap jarak. Sketsa pengujian koneksi dapat dilihat pada ilustrasi Gambar 4.15.



Gambar 4.15 Ilustrasi Pengujian Koneksi Tanpa Penghalang

Gambar pengujian koneksi pada *command prompt* dapat dilihat pada Gambar 4.16, Gambar 4.17, Gambar 4.18, dan Gambar 4.19.

```
C:\Users\user>ping 192.168.0.177

Pinging 192.168.0.177 with 32 bytes of data:
Reply from 192.168.0.177: bytes=32 time=5ms TTL=128
Reply from 192.168.0.177: bytes=32 time=5ms TTL=128
Reply from 192.168.0.177: bytes=32 time=6ms TTL=128
Reply from 192.168.0.177: bytes=32 time=8ms TTL=128

Ping statistics for 192.168.0.177:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 5ms, Maximum = 8ms, Average = 6ms
```

Gambar 4.16 Tes Ping 5 Meter Tanpa Penghalang

Terlihat pada Gambar 4.16 saat *access point* dan PC diletakkan pada jarak 5m tanpa penghalang, koneksi *access point* dan PC saat dilakukan tes ping pada *command prompt* masih sangat bagus dan tidak ada koneksi yang *loss*.

```
Command Prompt

C:\Users\user>ping 192.168.0.177

Pinging 192.168.0.177 with 32 bytes of data:
Reply from 192.168.0.177: bytes=32 time=7ms TTL=128
Reply from 192.168.0.177: bytes=32 time=51ms TTL=128
Reply from 192.168.0.177: bytes=32 time=7ms TTL=128
Reply from 192.168.0.177: bytes=32 time=5ms TTL=128

Ping statistics for 192.168.0.177:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 5ms, Maximum = 51ms, Average = 17ms
```

Gambar 4.17 Tes Ping 15 Meter Tanpa Penghalang

Terlihat pada Gambar 4.17 saat *access point* dan PC diletakkan pada jarak 15m tanpa penghalang, koneksi *access point* dan PC saat dilakukan tes ping pada *command prompt* masih bagus dan tidak ada koneksi yang *loss*. Hanya saja waktu yang dibutuhkan untuk menerima data sedikit lebih lambat.

```
C:\Users\user>ping 192.168.0.177

Pinging 192.168.0.177 with 32 bytes of data:
Reply from 192.168.0.177: bytes=32 time=4ms TTL=128
Reply from 192.168.0.177: bytes=32 time=13ms TTL=128
Reply from 192.168.0.177: bytes=32 time=15ms TTL=128
Reply from 192.168.0.177: bytes=32 time=5ms TTL=128

Ping statistics for 192.168.0.177:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 4ms, Maximum = 15ms, Average = 9ms
```

Gambar 4.18 Tes Ping 30 Meter Tanpa Penghalang

Terlihat pada Gambar 4.18 saat *access point* dan PC diletakkan pada jarak 30m tanpa penghalang, koneksi *access point* dan PC saat dilakukan tes ping pada command prompt masih bagus dan tidak ada koneksi yang *loss*.

```
C:\Users\user>ping 192.168.0.177

Pinging 192.168.0.177 with 32 bytes of data:
Request timed out.
General failure.
General failure.
General failure.

Ping statistics for 192.168.0.177:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

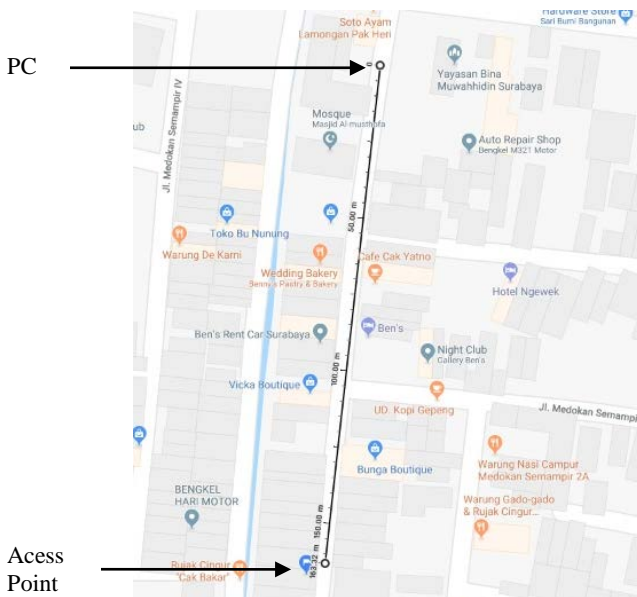
Gambar 4.19 Tes Ping 160 Meter Tanpa Penghalang

Dapat dilihat pada Gambar 4.19 menunjukkan bahwa *Ethernet Shield* yang terhubung pada laptop mengalami gangguan pada sistem komunikasinya, dibuktikan dari hasil ping pada *command prompt* di laptop pada jarak antara laptop dan *access point* sejauh 160 meter tanpa penghalang di ruang terbuka.

Tabel 4.4. Pengujian Koneksi Tanpa Penghalang

Jarak (Meter)	Koneksi <i>Loss</i> (%)	<i>Time Average</i>
5	0	6ms
15	0	17ms
30	0	9ms
160	100	2ms

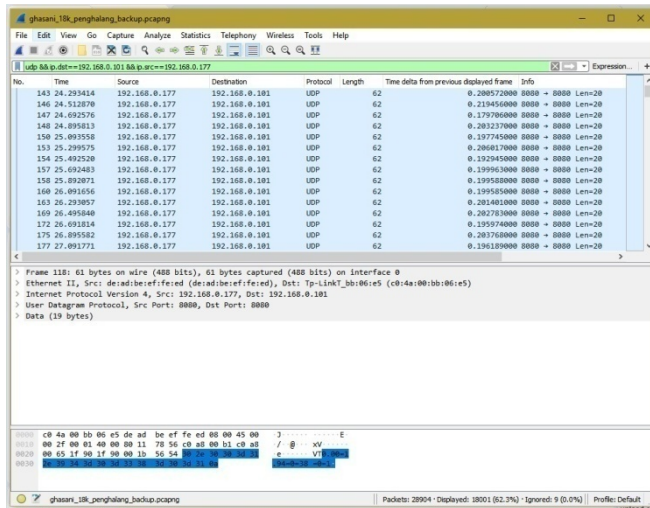
Gambar 4.20 menunjukkan jarak antara PC dan access point dilihat dari *Google Maps*.



Gambar 4.20 Jarak PC dan Access Point Pada *Google Maps*

4.4 Pengujian Dengan Software Wireshark

Selain menggunakan fasilitas *Command Prompt* yang sudah ada pada masing masing PC, terdapat sebuah *software* yang dapat mendeteksi *packet loss*, *throughput* dan juga *delay* dari data yang di dikirim yaitu *software* Wireshark. Pada pengujian *packet loss* menggunakan *software* Wireshark ini pengujian hanya dilakukan pada jarak 5m dengan penghalang dinding ruangan dengan ilustrasi seperti Gambar 4.10. Tampilan pada *software* Wireshark dapat dilihat pada Gambar 4.21.



Gambar 4.21 Tampilan Wireshark

Packet loss adalah banyaknya paket yang hilang pada suatu jaringan paket yang disebabkan oleh tabrakan (*collision*), penuhnya kapasitas jaringan, dan penurunan paket yang disebabkan oleh habisnya TTL (*Time To Live*) paket. Cara menghitung *packet loss* dapat digunakan Persamaan 4.1 :

$$Packet\ Loss = \frac{Data\ yang\ dikirim - Data\ yang\ diterima}{Data\ yang\ dikirim} \times 100\% \dots(4.1)$$

Pada Tugas Akhir ini banyaknya paket data yang hilang adalah 0% dikarenakan interval yang digunakan adalah 200ms sehingga data yang dikirimkan setiap detik adalah 5 data dan lamanya waktu pengambilan data adalah selama 1 jam sehingga data yang dikirimkan adalah sebanyak 18000 data per jam. Pada Arduino diberi program untuk mengirimkan 18.000 data sehingga yang akan di *capture* oleh Wireshark hanya 18.000 data saja. Gambar 4.22 berikut menunjukkan banyaknya data yang dikirimkan oleh

Arduino. Jumlah data yang diterima pada Wireshark akan ditunjukkan pada Tabel 4.5 di halaman Lampiran A.

```

////////////////////////////////////SEND DATA////////////////////////////////////

if(ghasani <= 18000 && voltage_r > 1)
{
dtostrf(voltage_m, 4, 2, str_temp); // mengubah data float voltage jadi string
sprintf(char_buff,"%s", str_temp); // menyimpan string di var char_buff
}

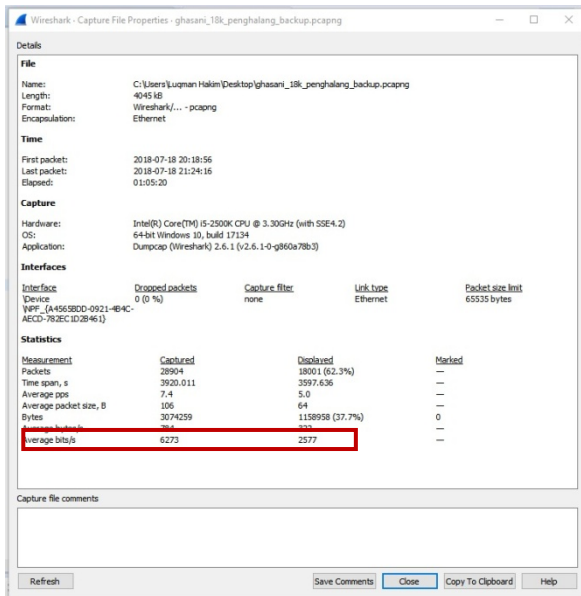
```

Gambar 4.22 Program Mengirimkan 18.000 Data

Throughput adalah kecepatan rata-rata data yang diterima oleh suatu *node* dalam selang waktu pengamatan tertentu. *Throughput* merupakan *bandwidth* aktual saat itu juga dimana kita sedang melakukan koneksi. Satuan yang dimilikinya sama dengan *bandwidth* yaitu bps. Cara menghitung *throughput* dapat digunakan Persaman 4.2:

$$Throughput = \frac{\text{Jumlah data yang dikirim}}{\text{Waktu pengiriman data}} \dots\dots\dots(4.2)$$

Pada Tugas Akhir ini *throughput* nya adalah 0,002577 Mb/s dapat dilihat pada Gambar 4.23 yang diberi tanda merah.



Gambar 4.23 *Throughput* Pada Wireshark

Delay adalah waktu tunda saat paket yang diakibatkan oleh proses transmisi dari satu titik menuju titik lain yang menjadi tujuannya. *Delay* diperoleh dari selisih waktu kirim antara satu paket UDP dengan paket lainnya yang direpresentasikan dalam satuan *seconds*. Perhitungan *delay* dapat dilihat pada Persamaan 4.3

$$Delay = \frac{Total\ Delay}{Total\ Paket\ yang\ Diterima} \dots\dots\dots(4.3)$$

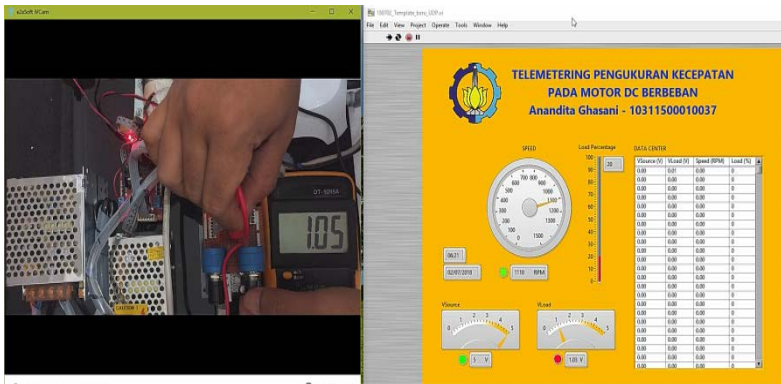
Pada Wireshark, *delay* didapatkan dari merata-rata *time delta from previous displayed frame*, dengan melihat Tabel 4.5 pada Lampiran A, sehingga didapatkan *delay* nya adalah 0,199954 s.

4.5 Kalibrasi Sensor Kecepatan

Sebelum melakukan pengujian maka dilakukan kalibrasi pada sensor kecepatan. Pada Gambar 4.24 menunjukkan motor yang berputar telah diberi beban sehingga pada tampilan LabView tegangan motor muncul 5V dimana merupakan tegangan maksimal dan tegangan beban diberikan 1V dimana persentase pengeremannya adalah sebesar 29% dengan kecepatan 960 rpm.

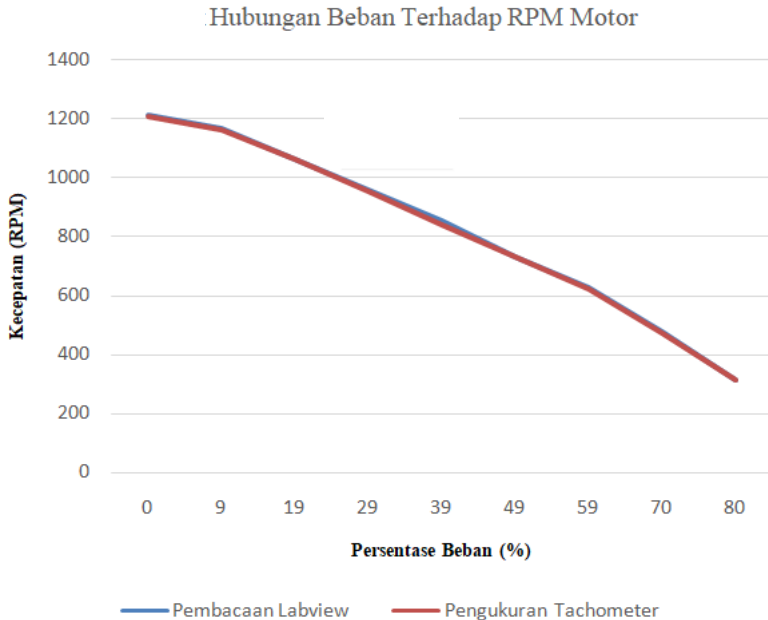
Dengan menggunakan rumus di Persamaan 4.4 dapat diambil nilai persen *error* dari setiap nilai yang diambil dari sensor kecepatan IR *Optocoupler*, tegangan motor dan tegangan beban.

$$\%Error = \frac{\left(\frac{Nilai\ Terbaca}{Labview - Nilai\ Terukur} \right)}{Alat\ Ukur} \times 100\% \dots\dots\dots(4.4)$$



Gambar 4.24 Tampilan LabView Motor Diberi Beban

Data hasil kecepatan motor DC dengan pengereman akan ditunjukkan pada Gambar 4.25.



Gambar 4.25 Kalibrasi Sensor Kecepatan

Berdasarkan data pada Gambar 4.25 hasil pengukuran kecepatan motor DC berbeban, dapat dilihat bahwa pada saat tegangan motor berada pada tegangan maksimal 5,02V, pengereman dapat dilakukan mulai 9% atau dengan tegangan beban sebesar 0,5V. Pada kondisi kecepatan konstan pada tegangan motor 5,02V pengereman maksimal yang dapat dilakukan adalah sebesar 80% dengan kecepatan 315 rpm. Untuk data berupa tabel dapat dilihat pada Tabel 4.6 pada halaman Lampiran A. Nilai *error* terbesar pada beban rem adalah 2,56% didapat dari Persamaan 4.4.

Tegangan pada beban dan kecepatan motor DC berbanding terbalik sehingga saat tegangan beban terus dinaikkan, kecepatan motor DC juga semakin menurun. Saat proses pengambilan data kecepatan motor DC berbeban, ketika pengereman sebesar 80% motor mulai mengalami kesulitan saat dilakukan pengereman dan *driver* L298N mulai panas akibat arus yang ditimbulkan motor DC berbeban.

Berdasarkan hasil yang ditunjukkan pada Gambar 4.25, untuk menghindari kerusakan pada *driver* L298N dan motor DC maka pada

pengujian selanjutnya diberikan batas pada PWM beban maksimal hanya mengeluarkan 70% saja sehingga pada saat pengereman maksimal terbaca di LabView, motor tidak akan berhenti dan *driver* tidak akan panas. Gambar 4.26 menunjukkan potongan program untuk membatasi nilai PWM beban diberi tanda merah.

```
int val_sensor_m = analogRead(pot_m); //membaca nilai adc pot_m
int val_pwm_m = map(val_sensor_m, 0, 1023, 0, 255); //mengubah val_sensor dari 10bit adc menjadi 8bit pwm
float voltage_m = val_sensor_m * (5.2 / 1023.0); //mengubah nilai adc menjadi nilai teg yang ditampilkan

float max = 0.7*val_pwm_m; // 70% nilai max pwm beban

int val_sensor_r = analogRead(pot_r);
int val_pwm_r = map(val_sensor_r, 0, 1023, 0, max); //membatasi nilai pwm beban sehingga tidak lebih dri nilai max
float voltage_r = val_sensor_r * (5.2 / 1023.0);
```

Gambar 4.26 Potongan Program Membatasi PWM Beban

Gambar 4.27 menunjukkan cara pengukuran kecepatan motor DC menggunakan tachometer digital pada pengambilan data Tabel 4.5.



Gambar 4.27 Pengukuran Kecepatan Menggunakan Tachometer

4.6 Pengujian pada *Software* LabView

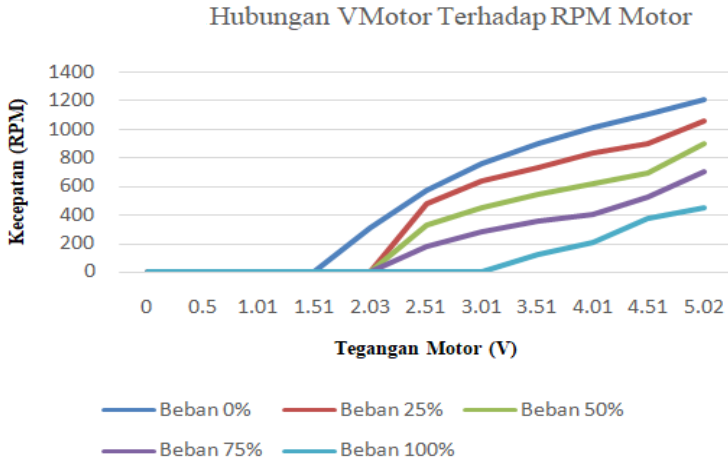
Setelah pengujian komunikasi *Ethernet* pada Arduino Mega 2560, pengujian komunikasi *Ethernet* perlu dilakukan pada *software* LabView untuk mengetahui apakah *Ethernet Shield* dapat mengirimkan data pada *software* LabView atau tidak.

Setelah dilakukan kalibrasi maka pengujian *software* LabView dapat dilakukan dengan menguji kecepatan motor DC tanpa beban, beban konstan 25%, beban konstan 50%, beban konstan 75% dan beban konstan 100%. Tampilan LabView pada motor saat belum diaktifkan dapat dilihat pada Gambar 4.28. Gambar 4.28 menunjukkan dalam keadaan belum aktif dimana tegangan yang diberikan ke motor adalah 0V, tidak terjadi putaran motor dimana *gauge* pada LabView menunjukkan 0 rpm.



Gambar 4.28 Tampilan LabView Motor Belum Diaktifkan

Dapat dilihat pada Gambar 4.28 motor belum dalam keadaan bergerak sehingga nilai pembacaan kecepatan dan tegangan belum muncul pada LabView. Gambar 4.29 menunjukkan perbandingan kecepatan motor tanpa beban dan kecepatan motor dengan beban konstan. Dapat dilihat bahwa titik awal motor berputar adalah pada tegangan motor 1,51V jika tanpa beban sedangkan titik awal pembebanan pada beban konstan 25%, 50% dan 75% dimulai dari titik yang sama yaitu pada tegangan motor 2,03V sedangkan pada beban konstan 100%, titik awal pembebanan adalah pada tegangan motor 3,01V. Untuk pengujian secara terpisah akan ditunjukkan pada sub bab selanjutnya.



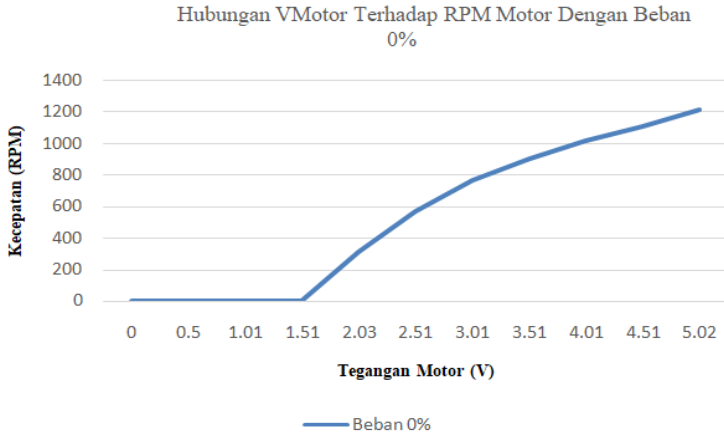
Gambar 4.29 Gabungan Respon Kecepatan

4.6.1. Pengujian *Software* LabView pada Motor Tanpa Beban



Gambar 4.30 Tampilan LabView Motor Tanpa Beban

Gambar 4.30 menunjukkan tampilan LabView pada motor saat tanpa beban dimana pada saat diberikan tegangan ke motor 3,5V dan tampil di *gauge* pada LabView dengan kecepatan 900 rpm dan tegangan yang diberikan ke beban 0V dengan persentase beban 0%. Hasil pengujian *software* LabView pada motor tanpa beban dapat dilihat pada Gambar 4.31.



Gambar 4.31 Pengujian Kecepatan Motor Tanpa Beban

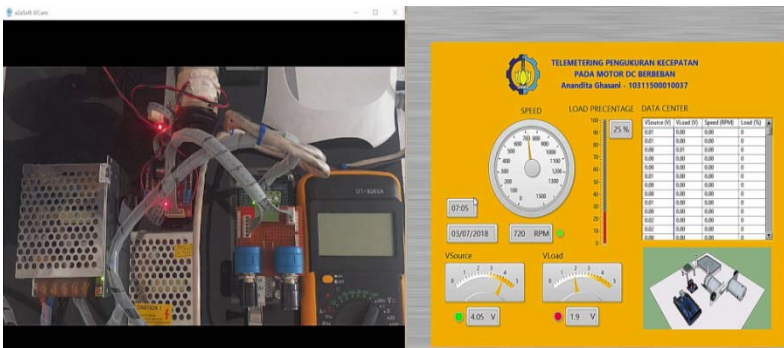
Saat pengambilan data kecepatan motor DC tanpa beban, motor DC mulai bergerak saat tegangan motor sebesar 1,8V namun sangat lambat dan masih belum terbaca oleh sensor kecepatan IR Optocoupler. Berdasarkan Gambar 4.31, motor DC baru akan berputar saat diberi tegangan motor sebesar 2,03V tanpa beban. Tegangan motor dan kecepatan motor DC berbanding lurus sehingga saat tegangan motor terus dinaikkan, kecepatan motor DC juga semakin bertambah. Kecepatan maksimal motor DC tanpa beban yang diberi tegangan maksimal 5,02V adalah sebesar 1215 rpm terbaca pada LabView. Untuk data berupa tabel dapat dilihat pada Tabel 4.7 pada halaman Lampiran A. Gambar 4.32 menunjukkan cara pengukuran kecepatan motor DC menggunakan tachometer digital pada pengambilan data Tabel 4.7.



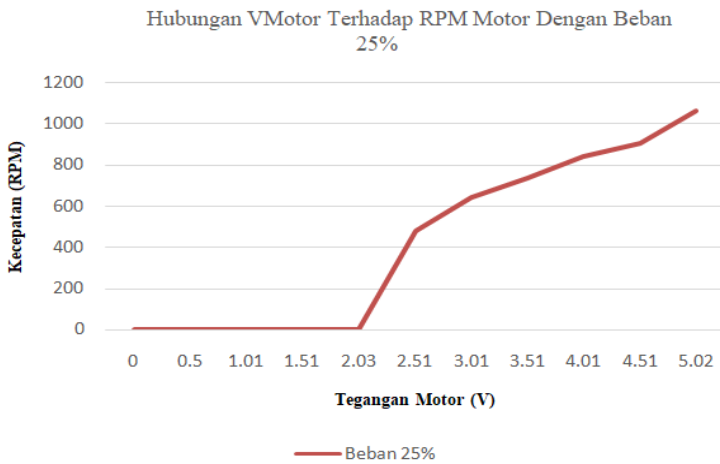
Gambar 4.32 Pengukuran Kecepatan Menggunakan Tachometer

4.6.2. Pengujian *Software* LabView pada Motor dengan Beban Konstan 25%

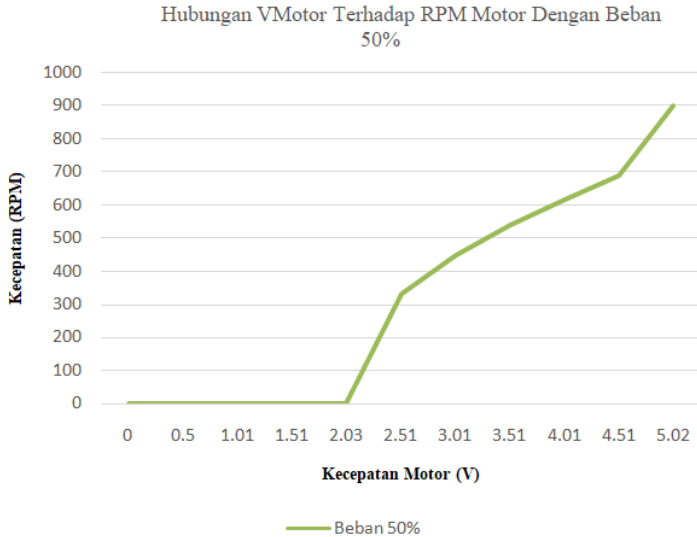
Selanjutnya adalah pengambilan data kecepatan motor DC berbeban dengan beban konstan dan kecepatan motor diubah-ubah. Pada Gambar 4.34 menunjukkan data hasil kecepatan motor DC dengan beban konstan 25% dimana 25% adalah tegangan yang keluar dari potensiometer beban adalah sebesar 1,3V dengan R1 bernilai 37,5 kΩ dan R1 bernilai 12,5 kΩ. Pada Gambar 4.33 menunjukkan persentase pengereman yang tampil pada *front panel* LabView adalah 25% dengan tegangan beban 1,25V saat tegangan motor sebesar 4,05V membuat motor berputar 720 rpm.



Gambar 4.33 Pengereman Konstan 25%



Gambar 4.34 Pengujian Kecepatan Motor Beban Konstan 25%

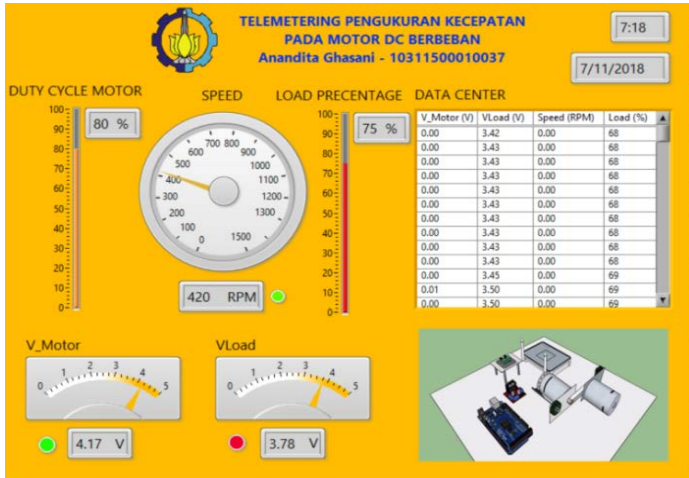


Gambar 4.36 Pengujian Kecepatan Motor Beban Konstan 50%

Saat pengambilan data kecepatan motor DC berbeban konstan 50%, motor baru akan berputar saat diberikan tegangan motor sebesar 2V namun putarannya sangat lambat sehingga belum terdeteksi oleh sensor IR Optocoupler. Berdasarkan Gambar 4.36 saat diberikan tegangan 2,51V motor bergerak dan putarannya dapat terdeteksi oleh sensor IR Optocoupler sebesar 330 rpm. Pada beban 50% motor DC dapat berputar dengan kecepatan maksimal 900 rpm. Untuk data berupa tabel dapat dilihat pada Tabel 4.9 pada halaman Lampiran A.

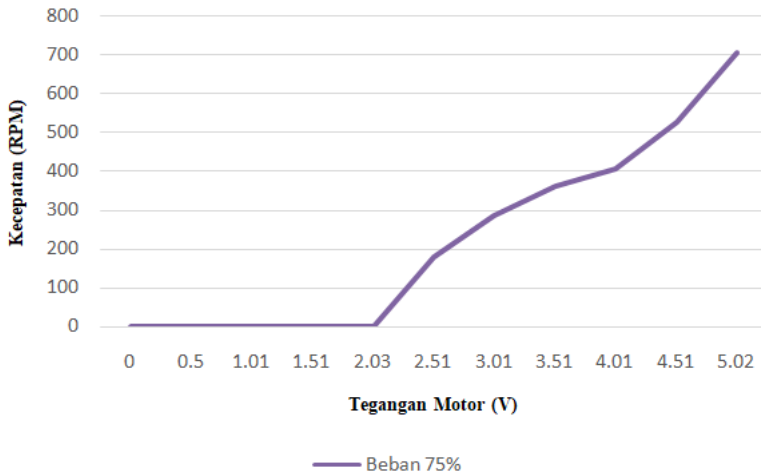
4.6.4. Pengujian *Software* LabView pada Motor Dengan Beban Konstan 75%

Selanjutnya adalah pengambilan data kecepatan motor DC berbeban dengan beban konstan dan kecepatan motor diubah-ubah. Pada Gambar 4.38 menunjukkan data hasil kecepatan motor DC dengan beban konstan 75% dimana 75% adalah tegangan yang keluar dari potensiometer beban adalah sebesar 3,75V dengan R1 bernilai 12,5 kΩ dan R1 bernilai 37,5 kΩ. Pada Gambar 4.37 menunjukkan persentase pengereman yang tampil pada *front panel* LabView adalah 75% dengan tegangan beban 3,7V saat tegangan motor sebesar 4,17V membuat motor berputar 420 rpm.



Gambar 4.37 Pengereman Konstan 75%

Hubungan VMotor Terhadap RPM Motor Dengan Beban 75%



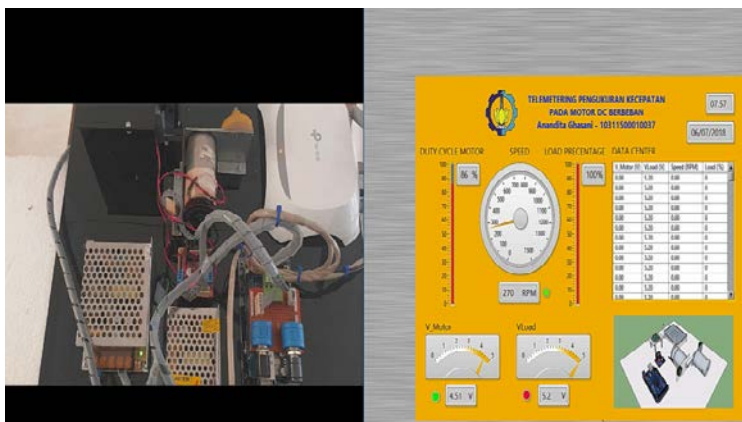
Gambar 4.38 Pengujian Kecepatan Motor Beban Konstan 75%

Saat pengambilan data kecepatan motor DC berbeban konstan 75%, motor baru akan berputar saat diberikan tegangan motor sebesar 2,38V namun putarannya sangat lambat sehingga belum terdeteksi oleh sensor IR *Optocoupler*. Berdasarkan Gambar 4.38, saat diberikan tegangan

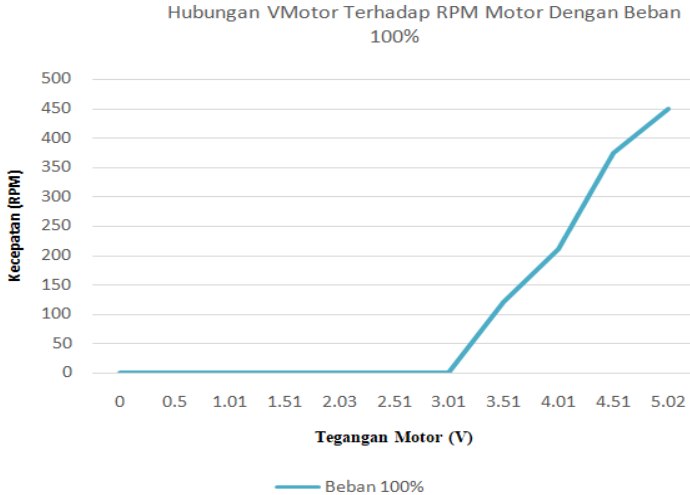
motor 2,51V motor bergerak dan putarannya terdeteksi oleh sensor IR Optocoupler sebesar 180 rpm. Pada beban 75% motor DC dapat berputar dengan kecepatan maksimal 705 rpm. Untuk data berupa tabel dapat dilihat pada Tabel 4.10 pada halaman Lampiran A.

4.6.5. Pengujian *Software* LabView pada Motor Dengan Beban Konstan 100%

Selanjutnya adalah pengambilan data kecepatan motor DC berbeban dengan beban konstan dan kecepatan motor diubah-ubah. Pada Gambar 4.40 menunjukkan data hasil kecepatan motor DC dengan beban konstan 100% dimana 100% adalah tegangan yang keluar dari potensiometer beban adalah sebesar 5V dengan R1 dan R2 masing-masing bernilai 50 kΩ. Pada Gambar 4.39 menunjukkan persentase pengereman yang tampil pada *front panel* LabView adalah 100% dengan tegangan beban maksimal 5V saat tegangan motor sebesar 4,51V membuat motor berputar 370 rpm.



Gambar 4.39 Pengereman Konstan 100%



Gambar 4.40 Pengujian Kecepatan Motor Beban Konstan 75%

Saat pengambilan data kecepatan motor DC berbeban konstan 100%, motor baru akan berputar saat diberikan tegangan motor sebesar 3V namun putarannya sangat lambat sehingga belum terdeteksi oleh sensor IR Optocoupler. Berdasarkan Gambar 4.40 saat diberikan tegangan 3,5V motor bergerak dan putarannya dapat terdeteksi oleh sensor IR Optocoupler sebesar 120 rpm. Pada beban 100% motor DC dapat berputar dengan kecepatan maksimal 450 rpm. Untuk data berupa tabel dapat dilihat pada Tabel 4.11 pada halaman Lampiran A.

BAB V

PENUTUP

5.1 Kesimpulan

Dari seluruh tahapan yang sudah dilaksanakan pada penyusunan Tugas Akhir ini, mulai dari studi literatur, perancangan dan pembuatan sampai pada pengujiannya maka dapat disimpulkan bahwa :

1. Proses pengiriman data dari *access point* menuju PC dengan penghalang dapat terus dilakukan dengan *loss* 0% mulai jarak modul ke PC berjarak 1m dan tetap stabil dengan *loss* 0% sampai jarak 10m, namun saat jarak PC dan modul sejauh 17m data *loss* sebesar 25% dan saat berjarak sejauh 20% data *loss* sebesar 50% dengan penghalang dinding ruangan. Sedangkan proses pengiriman data dari *access point* menuju PC tanpa penghalang di ruang terbuka dapat terus dilakukan dengan *loss* 0% mulai jarak modul ke PC berjarak 1m dan tetap stabil dengan *loss* 0% sampai jarak 150m, namun saat jarak PC dan modul sejauh 160m data langsung hilang 100%. Dan saat dilakukan koneksi ulang di jarak 160m, PC sudah tidak mendeteksi *access point* lagi.
2. Dari pengujian Quality of Service menggunakan *software* Wireshark didapatkan paket data yang hilang adalah 0% dengan *throughput* senilai 0,002577 Mb/s dan *delay* sebesar 0,199954 s pada jarak 5m dengan penghalang dinding ruangan.
3. Motor DC tanpa beban baru berputar saat diberi tegangan sebesar 2,03V dengan kecepatan 315 rpm terbaca di LabView dan dengan tegangan maksimal dapat berputar hingga kecepatan 1215 rpm terbaca di LabView. Sedangkan saat berbeban konstan 25% motor DC baru berputar saat diberi tegangan 2,51V dengan kecepatan 480 rpm dan kecepatan maksimal mencapai 1065. Saat motor DC diberi beban konstan 50%, motor DC baru berputar saat diberi tegangan motor sebesar 2,51V dengan kecepatan 330 rpm dan kecepatan maksimal mencapai 900 rpm. Saat motor DC diberi beban konstan 75%, motor DC baru berputar saat diberi tegangan motor sebesar 2,51V dengan kecepatan 180 rpm dan kecepatan maksimal mencapai 705 rpm. Sedangkan saat berbeban 100% motor baru berputar saat diberi tngangan sebesar 3,5V dengan

kecepatan 150 rpm dan kecepatan maksimal mencapai 450 rpm.

4. Pada tegangan motor maksimal 5,02V, motor DC mulai dapat dilakukan pengereman pada persentase pengereman sebesar 9% atau saat diberi tegangan beban 0,5V dengan kecepatan menjadi 1170 rpm.

5.2 Saran

Untuk lebih memperbaiki dan menyempurnakan kinerja dari alat ini, maka perlu disarankan :

1. Diharapkan kedepannya jika ingin memperluas jarak pengiriman data, sebaiknya menggunakan *access point* yang memiliki kemampuan pengiriman lebih baik daripada TP Link TL-WR 840N seperti yang digunakan pada tugas akhir ini.
2. Diharapkan kedepannya jika ingin memantau kecepatan motor yang memiliki kecepatan lebih maka sebaiknya digunakan motor BLDC yang memiliki kecepatan tinggi.
3. Diharapkan kedepannya alas peletakan alat menggunakan akrilik yang lebih tebal agar komponen di atasnya menjadi kokoh.

DAFTAR PUSTAKA

- [1] S'to. "**Wireless Kung Fu Networking and Hacking**", Jasakom, Jakarta Barat, 2007.
- [2] Dicky Hadiyuwono. "*User Datagram Protocol*", **Makalah**, Departemen Teknik Elektro dan Teknologi Informasi, Fakultas Teknik, Universitas Gajah Mada, Jogjakarta, 2011.
- [3] Fadila Lingga Dewi. "*Alat Penghitung Jumlah Barang Menggunakan Barcode ITF-14*", **Tugas Akhir**, Departemen Teknik Elektro Otomasi FV-ITS, Surabaya, 2017.
- [4] Pradipta R. "*Instrumen Penghitung Nilai Koefisien Pemuaian Linear Logam Berbasis Mikrokontroler*", **Tugas Akhir**, Departemen Fisika, Fakultas Matematika Ilmu Alam, Universitas Indonesia, Depok, 2011.
- [5] Intan Nur Robi Annisa, Zaka Perwira. "*Pembuatan Modul Kontrol Kecepatan Motor Brushless DC Dengan Mikrokontroler*", **Tugas Akhir**, Departemen Teknik Elektro Otomasi, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember, Surabaya, 2016.
- [6] Bachtiar, Muhammad Fachri, Priyatna, Alif Gigah. "*Perancangan Rem Magnetik Pada Motor DC Dengan Menggunakan Arduino*", **Tugas Akhir**, D3 Teknik Elektro, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember, Surabaya, 2015.
- [7] Farrah Fadhilah. "*Telemetry Kebocoran Pipa Pada Distribusi Air Dengan Komunikasi Ethernet*", **Tugas Akhir**, Departemen Teknik Elektro Otomasi, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember, Surabaya, 2018.
- [8] Stevani Agnesia Sigiro, Takdir Tamba, Mester Sitepu, Andi Setiono, "*Instrumentasi Virtual Menggunakan Labview Dan Soundcard*", **Jurnal Penelitian**, Jurusan Fisika, Fakultas Matematika Ilmu Alam, Universitas Sumatra Utara, Medan, 2010.

-- *Halaman ini sengaja dikosongkan* --

LAMPIRAN A

TABEL HASIL PENGUJIAN

Tabel 4.5. Jumlah Data yang Diterima Wireshark

1	No.	Time	Source	Destination	Protocol	Length	Time delta from previous displayed Info
2	118	22.493424	192.168.0.177	192.168.0.101	UDP	61	1.735338 8080 > 8080 Len=19
3	119	22.496783	192.168.0.177	192.168.0.101	UDP	61	0.003359 8080 > 8080 Len=19
4	123	22.694528	192.168.0.177	192.168.0.101	UDP	61	0.197745 8080 > 8080 Len=19
5	124	22.916507	192.168.0.177	192.168.0.101	UDP	61	0.221979 8080 > 8080 Len=19
6	126	23.093874	192.168.0.177	192.168.0.101	UDP	61	0.177367 8080 > 8080 Len=19
7	128	23.295796	192.168.0.177	192.168.0.101	UDP	61	0.201922 8080 > 8080 Len=19
8	129	23.493798	192.168.0.177	192.168.0.101	UDP	61	0.198002 8080 > 8080 Len=19
9	138	23.696402	192.168.0.177	192.168.0.101	UDP	61	0.202604 8080 > 8080 Len=19
10	140	23.893193	192.168.0.177	192.168.0.101	UDP	61	0.196791 8080 > 8080 Len=19
11	142	24.092842	192.168.0.177	192.168.0.101	UDP	62	0.199649 8080 > 8080 Len=20
12	143	24.293414	192.168.0.177	192.168.0.101	UDP	62	0.200572 8080 > 8080 Len=20
13	146	24.51287	192.168.0.177	192.168.0.101	UDP	62	0.219456 8080 > 8080 Len=20
14	147	24.692576	192.168.0.177	192.168.0.101	UDP	62	0.179706 8080 > 8080 Len=20
15	148	24.895813	192.168.0.177	192.168.0.101	UDP	62	0.203237 8080 > 8080 Len=20
16	150	25.093558	192.168.0.177	192.168.0.101	UDP	62	0.197745 8080 > 8080 Len=20
17	153	25.299575	192.168.0.177	192.168.0.101	UDP	62	0.206017 8080 > 8080 Len=20
18	154	25.49252	192.168.0.177	192.168.0.101	UDP	62	0.192945 8080 > 8080 Len=20
19	157	25.692483	192.168.0.177	192.168.0.101	UDP	62	0.199963 8080 > 8080 Len=20
20	158	25.892071	192.168.0.177	192.168.0.101	UDP	62	0.199588 8080 > 8080 Len=20
21	160	26.091656	192.168.0.177	192.168.0.101	UDP	62	0.199585 8080 > 8080 Len=20
22	163	26.293057	192.168.0.177	192.168.0.101	UDP	62	0.201401 8080 > 8080 Len=20
23	169	26.49584	192.168.0.177	192.168.0.101	UDP	62	0.202783 8080 > 8080 Len=20
24	172	26.691814	192.168.0.177	192.168.0.101	UDP	62	0.195974 8080 > 8080 Len=20
•							
•							
•							
17978	28049	3615.333565	192.168.0.177	192.168.0.101	UDP	65	0.20074 8080 > 8080 Len=23
17979	28050	3615.572725	192.168.0.177	192.168.0.101	UDP	65	0.23916 8080 > 8080 Len=23
17980	28051	3615.732583	192.168.0.177	192.168.0.101	UDP	65	0.159858 8080 > 8080 Len=23
17981	28052	3615.945681	192.168.0.177	192.168.0.101	UDP	65	0.213098 8080 > 8080 Len=23
17982	28054	3616.13188	192.168.0.177	192.168.0.101	UDP	65	0.186199 8080 > 8080 Len=23
17983	28055	3616.33183	192.168.0.177	192.168.0.101	UDP	65	0.19995 8080 > 8080 Len=23
17984	28056	3616.53205	192.168.0.177	192.168.0.101	UDP	65	0.20022 8080 > 8080 Len=23
17985	28057	3616.782277	192.168.0.177	192.168.0.101	UDP	65	0.250227 8080 > 8080 Len=23
17986	28058	3616.981165	192.168.0.177	192.168.0.101	UDP	65	0.148888 8080 > 8080 Len=23
17987	28060	3617.134449	192.168.0.177	192.168.0.101	UDP	65	0.203284 8080 > 8080 Len=23
17988	28061	3617.331506	192.168.0.177	192.168.0.101	UDP	65	0.197057 8080 > 8080 Len=23
17989	28064	3617.531382	192.168.0.177	192.168.0.101	UDP	65	0.199876 8080 > 8080 Len=23
17990	28065	3617.730432	192.168.0.177	192.168.0.101	UDP	65	0.19905 8080 > 8080 Len=23
17991	28068	3617.939636	192.168.0.177	192.168.0.101	UDP	65	0.209204 8080 > 8080 Len=23
17992	28070	3618.132931	192.168.0.177	192.168.0.101	UDP	65	0.193295 8080 > 8080 Len=23
17993	28071	3618.333312	192.168.0.177	192.168.0.101	UDP	65	0.200381 8080 > 8080 Len=23
17994	28074	3618.532903	192.168.0.177	192.168.0.101	UDP	65	0.199591 8080 > 8080 Len=23
17995	28075	3618.732785	192.168.0.177	192.168.0.101	UDP	65	0.199882 8080 > 8080 Len=23
17996	28078	3618.932072	192.168.0.177	192.168.0.101	UDP	65	0.199287 8080 > 8080 Len=23
17997	28080	3619.131825	192.168.0.177	192.168.0.101	UDP	65	0.199753 8080 > 8080 Len=23
17998	28081	3619.363895	192.168.0.177	192.168.0.101	UDP	65	0.23207 8080 > 8080 Len=23
17999	28084	3619.550922	192.168.0.177	192.168.0.101	UDP	65	0.187027 8080 > 8080 Len=23
18000	28085	3619.73018	192.168.0.177	192.168.0.101	UDP	65	0.179258 8080 > 8080 Len=23
18001	28088	3619.930313	192.168.0.177	192.168.0.101	UDP	65	0.200133 8080 > 8080 Len=23

Tabel 4.6. Kalibrasi Sensor Kecepatan

Tegangan Motor (Volt)		Tegangan Beban (Volt)		Error (%)	Persentase Beban	Kecepatan (RPM)		Error (%)
Pembacaan LabView	Pengukuran Multimeter	Pembacaan LabView	Pengukuran Multimeter			Pembacaan LabView	Pengukuran Tachometer	
5,02	4,93	0	0	0,00	0	1215	1211	0,003
5,02	4,93	0,5	0,5	0,99	9	1170	1166	0,003
5,02	4,93	1,02	1,01	0,67	19	1065	1066	0,000
5,02	4,93	1,51	1,50	2,03	29	960	957	0,003
5,02	4,93	2,01	1,97	2,03	39	855	845	0,011
5,02	4,93	2,51	2,46	2,03	49	735	734	0,001
5,02	4,93	3,01	2,95	2,03	59	630	623	0,011
5,02	4,93	3,51	3,44	2,03	70	480	477	0,006
5,02	4,93	4,01	3,91	2,56	80	315	316	0,003

Tabel 4.7. Pengujian Kecepatan Motor Tanpa Pengereman pada LabView

Tegangan Motor (Volt)		Error (%)	Tegangan Beban (Volt)		Kecepatan (RPM)		Error (%)
Pembacaan LabView	Pengukuran Multimeter		Pembacaan LabView	Pengukuran Multimeter	Pembacaan LabView	Pengukuran Tachometer	
0	0	0,00	0	0	0	0	0
0,5	0,51	1,96	0	0	0	0	0
1,01	1,02	0,98	0	0	0	0	0
1,51	1,51	0,00	0	0	0	0	0
2,03	2,01	1,00	0	0	315	310	0,016
2,51	2,48	1,21	0	0	570	573	0,005
3,01	2,96	1,69	0	0	765	764	0,001
3,51	3,47	1,15	0	0	900	902	0,002
4,01	3,95	1,52	0	0	1020	1019	0,000
4,51	4,46	1,12	0	0	1110	1114	0,003
5,02	4,93	1,83	0	0	1215	1211	0,003

Tabel 4.8 Pengujian Kecepatan Motor Dengan Beban Konstan 25%

Duty Cycle Motor (%)	Tegangan Motor (Volt)	Kecepatan Motor (RPM)
0	0	0
9	0,5	0
19	1	0
28	1,5	0
38	2	0
48	2,51	480
57	3	645
67	3,5	735
76	4,01	840
86	4,5	915
96	5	1065

Tabel 4.9 Pengujian Kecepatan Motor Dengan Beban Konstan 50%

Duty Cycle Motor (%)	Tegangan Motor (Volt)	Kecepatan Motor (RPM)
0	0	0
9	0,52	0
19	1	0
27	1,51	0
37	2,01	0
47	2,51	330
58	3,02	450
67	3,5	540
76	4,0	615
86	4,51	690
96	5	900

Tabel 4.10 Pengujian Kecepatan Motor Dengan Beban Konstan 75%

Duty Cycle Motor (%)	Tegangan Motor (Volt)	Kecepatan Motor (RPM)
0	0	0
9	0,52	0
19	1	0
27	1,51	0
37	2,01	0
47	2,51	180
57	3,02	285
67	3,5	360
76	4,0	405
86	4,51	525
96	5	705

Tabel 4.11 Pengujian Kecepatan Motor Dengan Beban Konstan 100%

Duty Cycle Motor (%)	Tegangan Motor (Volt)	Kecepatan Motor (RPM)
0	0	0
9	0,5	0
19	1	0
28	1,5	0
38	2,0	0
48	2,5	0
57	3,0	0
67	3,5	120
76	4,0	210
86	4,5	375
96	5	450

LAMPIRAN B

LISTING PROGRAM

LISTING PROGRAM ARDUINO MEGA 2560

```
#include <EthernetUdp.h>
#include <stdio.h>

char char_buff[100]; //karakter max 100 karakter yang bisa disimpan di
char buff
#define interval 200 //interval yang digunakan 200ms, interval =
waktu untuk 1 periode pengambilan data
unsigned long lastmillis = 0;

#define pot_r A0
#define pot_m A5

#define pwm_r 2
#define in1_r 4
#define in2_r A3

#define in1_m 5
#define in2_m 6
#define pwm_m 7

#define encoder_in 3

#define lubang_rotary 20 //jumlah lubang pada encoder disk
#define time_sampling 1000 //time sampling 1s
unsigned long counter = 0;
unsigned long last_counter = 0, dita=0; //last counter = nilai counter
pada interval sebelumnya
int D_counter;
unsigned long rpm = 0;
int percent_brake;
int percent_motor;
```



```
EthernetUDP Udp;
unsigned int localPort = 8080; //DEFAULT PORT
IPAddress ip_client(192, 168, 0, 100); //ip laptop tujuan - KOMPUTER
WORKSHOP (192, 168, 0, 100)
```

```
typedef union {
  float floatingPoint;
  byte binary[4];
} binaryFloat;
```

```
// put your setup code here, to run once:
```

```
void hitung_rpm()
{ //pembacaan data encoder ir optocoupler
  //counter = counter / lubang_rotary; // untuk mendapat 1 putaran
  berapa
  D_counter = last_counter - counter;
  D_counter = abs(D_counter);
  if (D_counter <= 1) //untuk mengabaikan perubahan pembacaan 1
  lubang tiap 1 interval (yg diabaikan perubahan lubangnya)
  {
    counter = last_counter;
  }
  rpm = counter * 1000 / interval ; //supaya menjadi perdetik, dikalikan
  1000, dibagi interval 200
  rpm = rpm * 60 / lubang_rotary ; //supaya menjadi permenit,
  dikalikan 60
  last_counter = counter;
  dita = counter;
  counter = 0;
}
```

```
void setup()
{
  setup_ethernet();
  Udp.begin(localPort);
}
```

```

Serial.begin(9600);

pinMode(encoder_in, INPUT_PULLUP);
// digitalWrite(3, HIGH); // pull up resistor

//attachInterrupt(digitalPinToInterrupt(encoder_in), rpm_sensor_m,
FALLING); //3 No Pin arduino bukan nomer Interupt
// attachInterrupt(1, rpm_sensor_m, FALLING);

pinMode(in1_m, OUTPUT);
pinMode(in2_m, OUTPUT);
pinMode(pwm_m, OUTPUT); //enableA

pinMode(in1_r, OUTPUT);
pinMode(in2_r, OUTPUT);
pinMode(pwm_r, OUTPUT); //enableB
//pinMode(30, OUTPUT);

}
boolean flag_tick = false; //yang diambil perubahan counter abis dari 0
jadi 1
void loop()
{
  //digitalWrite(30,HIGH);
  if (digitalRead(encoder_in) == LOW)
  {
    if (flag_tick == false)
    {
      counter++;
      flag_tick = true;
    }
  }
  else
  flag_tick = false;

  if (millis() - lastmillis >= interval )
  {

```

```

lastmillis = millis();

int val_sensor_m = analogRead(pot_m); //membaca nilai adc pot_m
int val_pwm_m = map(val_sensor_m, 0, 1023, 0, 255); //mengubah
val_sensor dari 10bit adc menjadi 8bit pwm
float voltage_m = val_sensor_m * (5.2 / 1023.0); //mengubah nilai
adc menjadi nilai teg yang ditampilkan

float max = 0.7*val_pwm_m; // 70% nilai max pwm beban

int val_sensor_r = analogRead(pot_r);
int val_pwm_r = map(val_sensor_r, 0, 1023, 0, max); //membatasi
nilai pwm beban sehingga tidak lebih dri nilai max
float voltage_r = val_sensor_r * (5.2 / 1023.0);

percent_brake = voltage_r * 100 / 5; //persentase pengereman=pwm
(ADC) rem dibandingkan dengan pwm(ADC) motor *100%
if (percent_brake >= 100) percent_brake = 100;
if (percent_brake <= 0) percent_brake = 0;

percent_motor = val_pwm_m * 100 / 255; //persentase motor=pwm
motor dibagi 255 nilai max * 100%
if (percent_motor <=0) percent_motor = 0;
if (percent_motor >=100) percent_motor = 100;

analogWrite(pwm_m, val_pwm_m);
//CW
digitalWrite(in1_m, HIGH);
digitalWrite(in2_m, LOW);
//CCW
// digitalWrite(in1_m, LOW);
// digitalWrite(in2_m, HIGH);

analogWrite(pwm_r, val_pwm_r);
//CCW
digitalWrite(in1_r, LOW);
digitalWrite(in2_r, HIGH);

```

```

//CW
// digitalWrite(in1_r, HIGH);
// digitalWrite(in2_r, LOW);

//last_counter=counter;
hitung_rpm();

Serial.print("Volt m=\t");
Serial.print(voltage_m, 2); //val_sensor_r , val_pwm_r
Serial.print("\tVolt r=\t");
Serial.print(voltage_r, 2);
Serial.print("\tRPM =\t");
Serial.print(rpm);
Serial.print("\tpercent =\t");
Serial.print(percent_brake);
Serial.print("\tpercent motor =\t");
Serial.print(percent_motor);
Serial.print("\n");

//s2_ethernet(voltage_m, voltage_r, rpm, percent_brake);
Udp.beginPacket(ip_client, localPort);
//sprintf()
binaryFloat hi;
hi.floatingPoint = voltage_m;
//Serial.write(hi.binary,4);

char str_temp[6];

/* 4 is minimum width, 2 is precision; float value is copied onto
str_temp*/

////////////////////////////////////SEND DATA////////////////////////////////////
dtostrf(voltage_m, 4, 2, str_temp); // mengubah data float voltage jadi
string
sprintf(char_buff,"%s", str_temp); // menyimpan string di var
char_buff

```

```

Udp.write(char_buff); //mengirim string char_buff ke UDP
Udp.write("=");

dtostrf(voltage_r, 4, 2, str_temp);
sprintf(char_buff,"%s", str_temp);

Udp.write(char_buff);
Udp.write("="); //separator data

sprintf(char_buff,"%d",rpm); //sprintf untuk mengirimkan data rpm
kedalam buff untuk disimpan sementara
Udp.write(char_buff); //buff untuk mengirim data buff ke ethernet;
nyimpan data sementara. data di simpan dulu baru di write
Udp.write("=");

sprintf(char_buff,"%d",percent_brake);
Udp.write(char_buff);
Udp.write("="); //penting, separator data, tanpa separator datanya
tergabung (menumpuk)

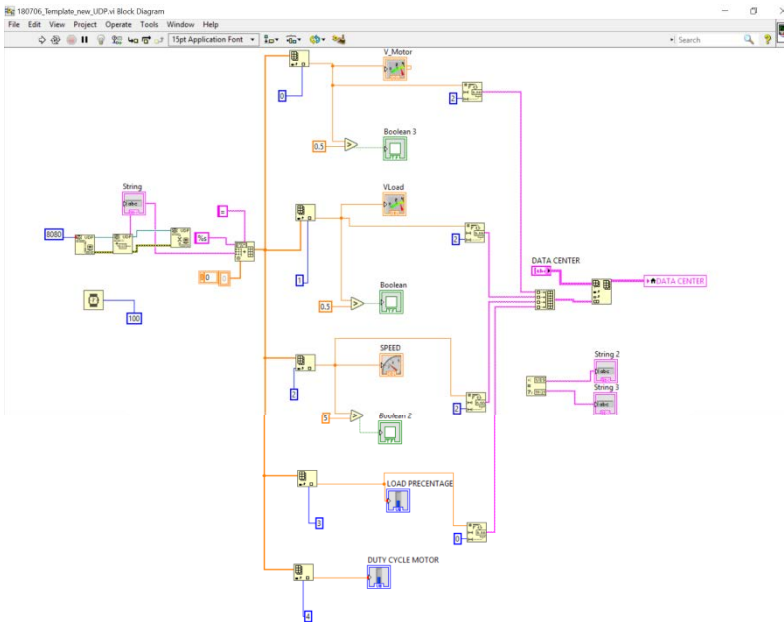
sprintf(char_buff,"%d",percent_motor);
Udp.write(char_buff);

Udp.write("\n");
Udp.endPacket();
}
}

void rpm_sensor_m()
{
counter++;
}

```

BLOK DIAGRAM LABVIEW



-- *Halaman ini sengaja dikosongkan* --

LAMPIRAN C

DATASHEET

1. Datasheet Arduino Mega 2560



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.

✦ **I²C:** **20 (SDA) and 21 (SCL).** Support I²C (TWI) communication using the [Wire library](#) (documentation on the [Wiring website](#)). Note that these pins are not in the same location as the I²C 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 it is 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.

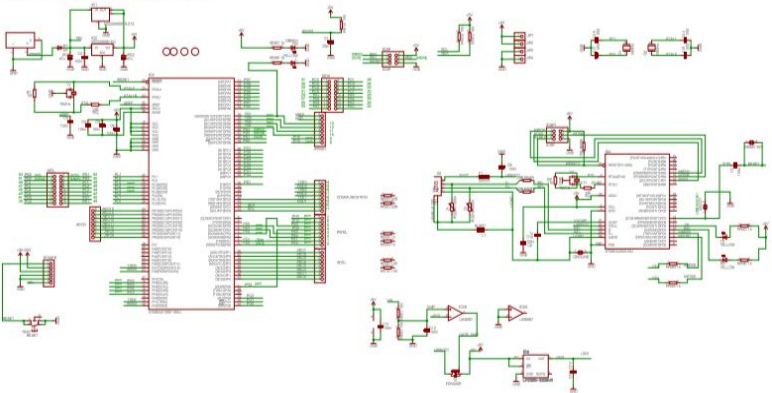
Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega2560 PCB are 4 and 2.4 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 I²C 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

© 2008-2012 Arduino LLC. All rights reserved. This document is the property of Arduino LLC. No part of this document may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, without the prior written permission of Arduino LLC.



2. Datasheet Driver Motor L298N

L298 Dual H-Bridge Motor Driver

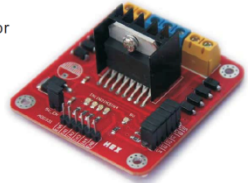
User's Guide

Overview

The Motor Shield is based on the L298, which is a dual full-bridge driver designed to drive inductive loads such as relays, solenoids, DC and stepping motors. It lets you drive two DC motors, controlling the speed and direction of each one independently.

Summary

Operating Voltage 4V to 35V
Motor controller L298N, Drives 2 DC motors or 1 stepper motor
Max current 2A per channel or 4A max
Free running stop and brake function
Chip: ST L298N
Logic power supply:5v
Max power:25w
Weight: 35g
Size:55mm x 60mm x 30mm
Storage temperature:-25 to +135



L298 Dual H-Bridge Motor Driver

User's Guide



CSA: Between this pin and ground is connected the sense resistor to control the current of the load.
Enable----- Ignore current detection function
CSB: Between this pin and ground is connected the sense resistor to control the current of the load.
Enable----- Ignore current detection function



5V-EN: Enable----78M05 worked ,output DC 5V
Disable----78M05 do not work . Need input DC 5V
The module need DC 5V always, for logic supply.



The pull-up resistor enabled.
U1---Enable In1 pull-up resistor [10k].
U2---Enable In2 pull-up resistor [10k].
U3---Enable In3 pull-up resistor [10k].
U4---Enable In4 pull-up resistor [10k].



IN1 IN2 :TTL Compatible Inputs of the Bridge A
IN3 IN4 :TTL Compatible Inputs of the Bridge B.
ENA ENB:TTL Compatible Enable Input: the L state disables the bridge A(enable A) and/or the bridge B (enable B).



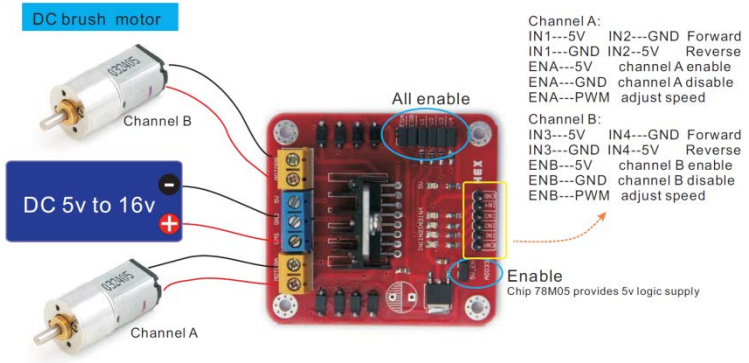
Logic power indicator



5V_EN:
Enable: [5V] can output DC 5V.
Disable:[5v] need input DC 5V.

L298 Dual H-Bridge Motor Driver

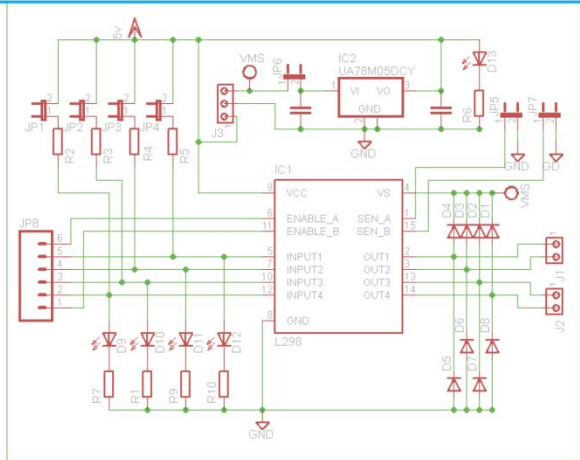
User's Guide



L298 Dual H-Bridge Motor Driver

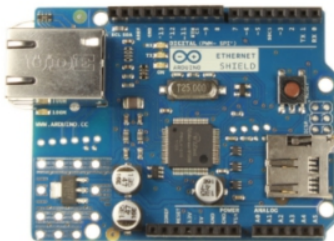
User's Guide

Schematic:

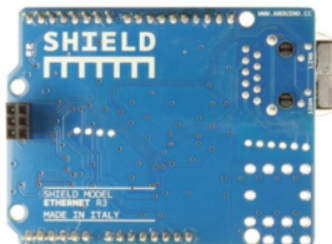


3. Datasheet Ethernet Shield

Arduino Ethernet Shield V1



Arduino Ethernet Shield R3 Front



Arduino Ethernet Shield R3 Back

Overview

NOTE: this product is currently retired and the documentation will not be kept up-to-date

The Arduino Ethernet Shield V1 connects your Arduino to the internet in mere minutes. Just plug this module onto your Arduino board, connect it to your network with an RJ45 cable (not included) and follow a few simple instructions to start controlling your world through the internet. As always with Arduino, every element of the platform – hardware, software and documentation – is freely available and open-source. This means you can learn exactly how it's made and use its design as the starting point for your own circuits. Hundreds of thousands of Arduino boards are already fueling people's creativity all over the world, everyday. Join us now, Arduino is you!

- Requires an Arduino board (not included)
- Operating voltage 5V (supplied from the Arduino Board)
- Ethernet Controller: W5100 with internal 16K buffer
- Connection speed: 10/100Mb
- Connection with Arduino on SPI port

Description

The Arduino Ethernet Shield V1 allows an Arduino board to connect to the internet. It is based on the [Wiznet W5100](#) ethernet chip ([datasheet](#)). The Wiznet W5100 provides a network (IP) stack capable of both TCP and UDP. It supports up to four simultaneous socket connections. Use the [Ethernet library](#) to write sketches which connect to the internet using the shield. The ethernet shield connects to an Arduino board using long wire-wrap headers which extend through the shield. This keeps the pin layout intact and allows another shield to be stacked on top.

The most recent revision of the board exposes the 1.0 pinout on rev 3 of the Arduino UNO board.

The Ethernet Shield V1 has a standard RJ-45 connection, with an integrated line transformer and Power over Ethernet enabled.

There is an onboard micro-SD card slot, which can be used to store files for serving over the network. It is compatible with all the Arduino/Genuino boards. The on-board micro SD card reader is accessible through the SD Library. When working with this library, SS is on Pin 4. The original revision of the shield contained a full-size SD card slot; this is not supported.

The shield also includes a reset controller, to ensure that the W5100 Ethernet module is properly reset on power-up. Previous revisions of the shield were not compatible with the Mega and need to be manually reset after power-up.

Download: [arduino-ethernet-shield-06-schematic.pdf](#), [arduino-ethernet-shield-06-reference-design.zip](#)

The current shield has a Power over Ethernet (PoE) module designed to extract power from a conventional twisted pair Category 5 Ethernet cable:

- IEEE802.3af compliant
- Low output ripple and noise (100mVpp)
- Input voltage range 36V to 57V
- Overload and short-circuit protection
- 9V Output
- High efficiency DC/DC converter: typ 75% @ 50% load
- 1500V isolation (input to output)

NB: the Power over Ethernet module is proprietary hardware not made by Arduino, it is a third party accessory. For more information, see the [datasheet](#)

The shield does not come with the PoE module built in, it is a separate component that must be added on.

Arduino communicates with both the W5100 and SD card using the SPI bus (through the ICSP header). This is on digital pins 10, 11, 12, and 13 on the Uno and pins 50, 51, and 52 on the Mega. On both boards, pin 10 is used to select the W5100 and pin 4 for the SD card. These pins cannot be used for general I/O. On the Mega, the hardware SS pin, 53, is not used to select either the W5100 or the SD card, but it must be kept as an output or the SPI interface won't work.

Note that because the W5100 and SD card share the SPI bus, only one can be active at a time. If you are using both peripherals in your program, this should be taken care of by the corresponding libraries. If you're not using one of the peripherals in your program, however, you'll need to explicitly deselect it. To do this with the SD card, set pin 4 as an output and write a high to it. For the W5100, set digital pin 10 as a high output.

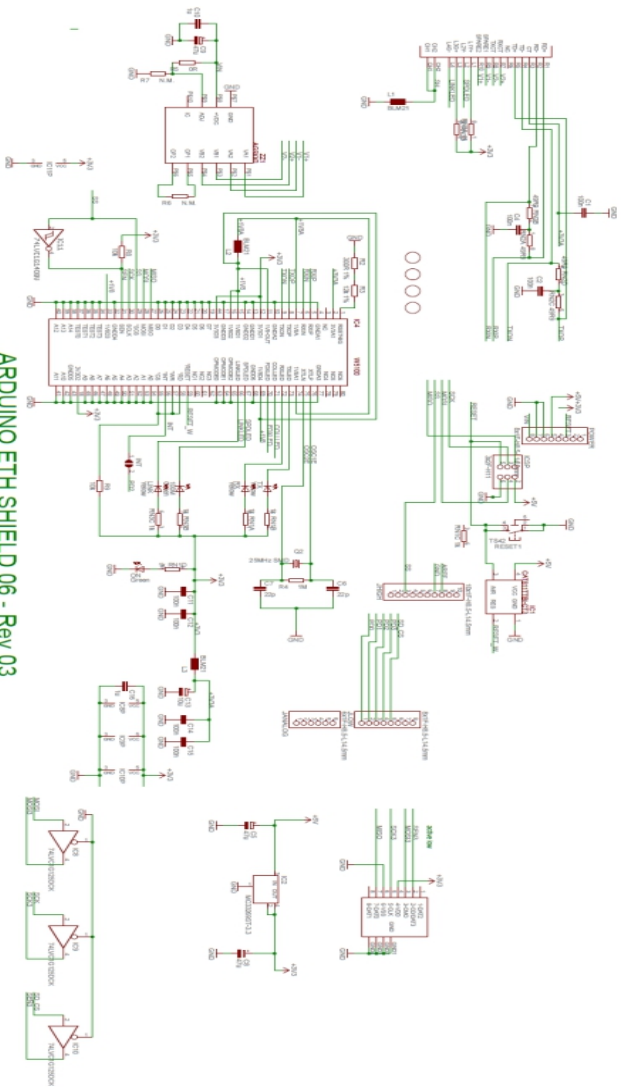
The shield provides a standard RJ45 ethernet jack.

The reset button on the shield resets both the W5100 and the Arduino board.

The shield contains a number of informational LEDs:

- PWR: indicates that the board and shield are powered
- LINK: indicates the presence of a network link and flashes when the shield transmits or receives data
- FULLD: indicates that the network connection is full duplex
- 100M: indicates the presence of a 100 Mb/s network connection (as opposed to 10 Mb/s)
- RX: flashes when the shield receives data
- TX: flashes when the shield sends data
- COLL: flashes when network collisions are detected

The solder jumper marked "INT" can be connected to allow the Arduino board to receive interrupt-driven notification of events from the W5100, but this is not supported by the Ethernet library. The jumper connects the INT pin of the W5100 to digital pin 2 of the Arduino.



ARDUINO ETH SHIELD 06 - Rev 03

Reference Designs ARE PROVIDED AS IS AND WITH ALL FAULTS. Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED REGARDING ANY PRODUCTS INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Reference Designs ARE PROVIDED AS IS AND WITH ALL FAULTS. Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED REGARDING ANY PRODUCTS INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Reference Designs ARE PROVIDED AS IS AND WITH ALL FAULTS. Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED REGARDING ANY PRODUCTS INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

ARDUINO is a registered trademark.

W5100 Datasheet

The W5100 is a full-featured, single-chip Internet-enabled 10/100 Ethernet controller designed for embedded applications where ease of integration, stability, performance, area and system cost control are required. The W5100 has been designed to facilitate easy implementation of Internet connectivity without OS. The W5100 is IEEE 802.3 10BASE-T and 802.3u 100BASE-TX compliant.

The W5100 includes fully hardwired, market-proven TCP/IP stack and integrated Ethernet MAC & PHY. Hardwired TCP/IP stack supports TCP, UDP, IPv4, ICMP, ARP, IGMP and PPPoE which has been proven in various applications for several years. 16Kbytes internal buffer is included for data transmission. No need of consideration for handling Ethernet Controller, but simple socket programming is required.

For easy integration, three different interfaces like memory access way, called direct, indirect bus and SPI, are supported on the MCU side.

Target Applications

The W5100 is well suited for many embedded applications, including:

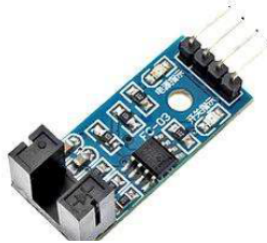
- Home Network Devices: Set-Top Boxes, PVRs, Digital Media Adapters
- Serial-to-Ethernet: Access Controls, LED displays, Wireless AP relays, etc.
- Parallel-to-Ethernet: POS / Mini Printers, Copiers
- USB-to-Ethernet: Storage Devices, Network Printers
- GPIO-to-Ethernet: Home Network Sensors
- Security Systems: DVRs, Network Cameras, Kiosks
- Factory and Building Automations
- Medical Monitoring Equipments
- Embedded Servers

4. Datasheet LM393 Speed Measuring Sensor Module

Rajguru Electronics

www.rajguruelectronics.com

LM393 Motor Speed Measuring Sensor Module For Arduino



GUARD-30A

Widely used in motor speed detection, pulse count, the position limit, etc. The DO output interface can be directly connected to a micro-controller IO port, if there is a block detection sensor, such as the speed of the motor encoder can detect.

DO modules can be connected to the relay, limit switch, and other functions, it can also with the active buzzer module, compose alarm.

Main technical characteristics:

- Dimensions: 32 x 14 x 7mm.
- The sensor reading slot has a width of 5mm.
- Two outputs, one Digital and one Analog.
- LED power indicator.
- LED indicator of the output pulses of pin DO.

Features

- Using imported trough type optical coupling sensor, groove width 5 mm.
- The output state light, lamp output level, the output low level light.
- Covered : output high level; Without sunscreen : the output low level.
- The comparator output, signal clean, good waveform, driving ability is strong, for more than 15 ma.
- The working voltage of 3.3 V to 5 V
- Output form: digital switch output (0 and 1)
- A fixed bolt hole, convenient installation
- Small board PCB size: 3.2 cm x 1.4 cm
- Use the LM393 wide voltage comparator Module

USES:

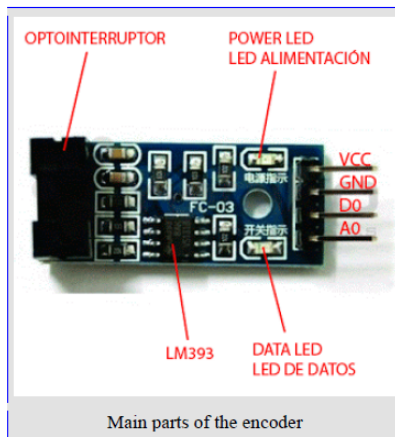
- The module without slot, the receiving tube conduction, module DO output low level, shade, the DO output high level;

- the DO output interface can be directly connected to a microcontroller IO port, if there is a block detection sensor, such as the speed of the motor encoder can detect.
- DO modules can be connected to the relay, limit switch, and other functions, can also with the active buzzer module, alarm. Product connection details: The positive 3.3-1, VCC power supply 5 v2, GND connect power cathode
- DO TTL switch signal output4, AO the module does not work Shipping list: A speed measuring sensor module

This IR speed module sensor with the comparator LM393, we can calculate the speed of rotation of the wheels of our robot. If we place a ring gear that rotates attached to our wheel. It could also be used as an optical switch.

The basic operation of this sensor is as follows; If anything is passed between the sensor slot, it creates a digital pulse on the D0 pin. This pulse goes from 0V to 5V and is a digital TTL signal. Then with Arduino we can read this pulse.

Here are the different parts:



Wiring specification:

- VCC Connect the positive 3.3 5 v power supply
- GND Connect power negative
- DO TTL switch signal output
- AO This module does not work
- The module do not have Analog output.

-- Halaman ini sengaja dikosongkan --

DAFTAR RIWAYAT HIDUP



Nama : Anandita Ghasani
TTL : Medan, 19 Agustus 1997
Kelamin: Perempuan
Agama : Islam
Alamat : Jl. Bank Raya 4 No.3 Palembang
Telp/HP: 081219250323
E-mail : ananditaghasani@gmail.com

RIWAYAT PENDIDIKAN

1. 2003 – 2009 : SD Muhammadiyah 14 Palembang
2. 2009 – 2012 : SMP Negeri 8 Padang
3. 2012 – 2015 : SMA Negeri 1 Palembang
4. 2015 – 2018 : Bidang Studi Elektro Industri, Departemen Teknik Elektro Otomasi, ITS

PENGALAMAN KERJA

1. Kerja Praktek di PT. PLN PERSERO Sektor Dalkit Keramasan, Palembang

-- *Halaman ini sengaja dikosongkan* --